Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

Máster en Ingeniería Industrial

MASTER EN INGENIERÍA INDUSTRIAL

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

TRABAJO FIN DE MÁSTER

DESARROLLO DE UNA HERRAMIENTA DE DIAGNÓSTICO DE FALLOS EN MOTORES DE INDUCCIÓN MEDIANTE LA TÉCNICA ADABOOST

Autor: D. Juan Obregón Sandoval
Tutor: D. Óscar Duque Pérez

Valladolid, Septiembre, 2016

RESUMEN

El objetivo de este Trabajo Fin de Máster es el de valorar la utilidad de los métodos de aprendizaje automático conocidos como AdaBoost para realizar una clasificación de motores de inducción en función del estado de las barras del rotor. De esta forma, se trata de ser capaces de identificar la avería de barra rota a partir de distintos predictores obtenidos del análisis de la corriente de alimentación del motor.

Para conseguir tal fin se utilizará el software comercial MATLAB, desarrollando una serie de herramientas que nos permitan realizar las tareas de clasificación y análisis necesarias.

PALABRAS CLAVE

Motor de Inducción. Fallo por rotura de barra. Clasificación. AdaBoost. MATLAB.

ABSTRACT

The aim of this Master Thesis is to assess the usefulness of the machine learning methods known as AdaBoost for classifying induction motors according to the state of the rotor bars. Thus, it comes to be able to identify the broken bar fault using different predictors obtained from the analysis of the motor supply current.

To achieve this purpose, the commercial software MATLAB is used, developing tools that allow us to perform the necessary classification and analysis tasks.

KEYWORDS

Induction motor. Bar breakage failure. Classification. AdaBoost. MATLAB.

ÍNDICE

CAPÍTULO 1

JUSTIFICACIÓN Y OBJETIVOS

1 JUSTIFICACIÓN Y OBJETIVOS

1.1 Justificación

El motor de inducción es una de las máquinas más utilizadas en la actualidad, de gran importancia en la industria y múltiples actividades humanas. Es un motor robusto, sencillo y que requiere de poco mantenimiento ya que es capaz de operar satisfactoriamente prácticamente hasta el colapso del mismo, momento en el que en el pasado simplemente se sustituía por un motor nuevo.

La gran competencia empresarial existente en la actualidad, así como las mayores exigencias de calidad y fiabilidad, han propiciado que las técnicas de mantenimiento cobren gran importancia respecto al pasado, incluyendo el caso que nos atañe. Las técnicas de análisis de fallos predictivas se han desarrollado lo suficiente como para que su utilidad práctica empiece a ser utilizada con resultados positivos, aplicándose a mantenimientos predictivos de tal manera que se consigue mayor ahorro y fiabilidad que nunca.

Dentro de los fallos propios de estos motores se encuentra el fallo por rotura de barra, que es uno de los fallos más comunes y mejor estudiados a día de hoy, por lo que la elección del mismo de cara a este tipo de mantenimientos es clara.

El siguiente paso lógico consiste en aprovechar estos conocimientos y trasladarlos al área del aprendizaje automático, de tal modo que sea un algoritmo el que realice el mantenimiento de manera automática sin depender del ser humano más allá de su desarrollo, todo ello bajo la sencillez del método que caracteriza a este tipo de aprendizaje.

Dentro del aprendizaje automático existen multitud de técnicas, cada una con una serie de características, ventajas e inconvenientes. El método AdaBoost es una de ellas, y está caracterizado por ser un método sencillo, eficiente, estable y computacionalmente rápido, siempre que se ajuste el modelo adecuadamente y se trabaje con una cantidad de datos y características acordes al mismo.

Este algoritmo se puede programar en cualquier lenguaje de programación, pero se ha elegido MATLAB para llevarlo a cabo. La elección de este lenguaje es debida a que MATLAB es a día de hoy uno de los referentes en cuanto a programación en ingeniería, y proporciona una gama de herramientas muy amplia para múltiples campos, siendo este de clasificación que nos trata uno de ellos.

Otros lenguajes como C proporcionan la mayor potencia, velocidad y rendimiento posibles, si bien habría que desarrollar todas las herramientas, funciones y tratamiento de datos que ya proporciona MATLAB de base, siendo la mejora para este caso concreto prácticamente insignificante, a tenor de los resultados conseguidos en MATLAB.

1.2 Objetivos

El objeto principal de este TFM es el de conseguir la detección de fallos por rotura de

barrasen motores de inducción basada en el método AdaBoost, y utilizando el lenguaje de programación comercial MATLAB.

Esta detección vendría dada por la clasificación de los datos a analizar, correspondientes a una serie de coeficientes, indicadores y parámetros obtenidos del análisis temporal y frecuencial de la corriente de alimentación de un motor sobre el que se han ido provocando fallos en el rotor artificiales.

Los datos a analizar incluyen la clasificación de cada uno de ellos como uno de 5 posibles estados del motor en relación a este tipo de fallos: desde motor completamente sano a motor con barra rota.

De esta forma, habrá que desarrollar una herramienta software que, una vez entrenada, sea capaz de proporcionar una clasificación suficientemente válida, concepto que ya se estudiará a lo largo de esta memoria.

Para ello se han desarrollado una serie de scripts en MATLAB destinados a la evaluación del método AdaBoost, creación de métodos complejos de clasificación que utilicen varios clasificadores basados en AdaBoost tratando de conseguir un mejor resultado que aplicando el algoritmo AdaBoost directamente, así como la elaboración de scripts para poder realizar dicha evaluación de los métodos empleados teniendo una herramienta de clasificación funcional, versátil y flexible.

1.2.1 Estructura de la memoria

El presente documento consta de 12 capítulos y un Anexo cuya temática se recoge a continuación:

- **Capítulo 1:** en este capítulo se analiza la utilidad del proyecto, justificando su realización y estableciéndose los objetivos que marcarán el alcance del trabajo de fin de máster.
- **Capítulo 2:** se describe y muestra la teoría detrás del motor de inducción, abarcando desde su historia a aspectos constructivos, funcionamiento, etc.
- **Capítulo 3:** se detallan los tipos de fallo que afectan a estos motores, así como sus consecuencias.
- **Capítulo 4:** se detallan las estrategias de mantenimiento existentes, indicándose cuáles tienen mayor importancia en la actualidad y la forma en que se aborda para este tipo de motores.
- **Capítulo 5:** se presenta una visión general de la monitorización orientada a motores de inducción, desde la relativa a los fenómenos físicos sobre los que se puede realizar un seguimiento hasta los conceptos de tratamiento de señales y técnicas de toma de decisiones.
- **Capítulo 6:** se trata en profundidad el análisis de señales orientado a la técnica seguida en el banco de ensayos: el análisis de la corriente de alimentación del motor para detectar la rotura de barra. Se indican los coeficientes, indicadores y parámetros utilizados como predictores en el proceso de clasificación estudiado.
- **Capítulo 7:** se presenta de forma descriptiva el proceso por el que se han obtenido los ensayos relativos a fallo del motor, así como los equipos utilizados para ello.

- **Capítulo 8:** se describe de manera general primero, centrándonos en nuestro caso posteriormente, la teoría detrás de los métodos de clasificación.
- **Capítulo 9:** se describe cómo utilizar en MATLAB las herramientas de clasificación de que se dispone, así como una breve explicación de las posibilidades de las mismas.
- **Capítulo 10:** se describe la metodología de trabajo seguida, así como se justifican distintas decisiones que se han tomado antes y/o durante el proceso de desarrollo del software. Asimismo se explica el esquema de funcionamiento general de las herramientas desarrolladas.
- **Capítulo 11:** se recogen y analizan los distintos ensayos que se han ido realizando para comprobar la validez de las herramientas software desarrolladas. Se comparan los resultados obtenidos y se estudian modificaciones al código en función de las observaciones realizadas.
- **Capítulo 12:** se hace una valoración de los resultados, aportando una serie de conclusiones e indicando posibles líneas de investigación futuras a partir de las bases de este trabajo.
- **Capítulo 13:** se indica la bibliografía consultada y utilizada para la realización del presente trabajo.

- **Anexos:** en ellos se incluyen el código desarrollado y utilizado, en el que están recogidos comentarios explicativos del mismo. También se incluyen el total de las gráficas obtenidas durante la realización de los ensayos, ya que en el capítulo correspondiente no se recogen todas por motivos de utilidad y espacio.

CAPÍTULO 2

EL MOTOR DE INDUCCIÓN

2 EL MOTOR DE INDUCCIÓN

2.1 Introducción

El motor de inducción es un motor eléctrico de corriente alterna de construcción simple y robusta, destacando el de rotor de jaula de ardilla, por lo que proporciona un excelente servicio con un mantenimiento reducido. Esto explica que hoy en día suponga entorno al 80% de los motores eléctricos industriales, sobre todo a partir del desarrollo de la electrónica industrial necesaria para permitirle trabajar en circunstancias que requerían anteriormente de otro tipo de motor [Fraile Mora, J. 2008].

Se trata de una máquina asíncrona que basa su funcionamiento en el concepto de campo magnético giratorio. Su concepción se realizó a finales del S.XIX de manera paralela entre Galileo Ferarris en Italia y Nikola Tesla en Estados Unidos, logrando Tesla un motor de mejores prestaciones, lo que explica que se considere a éste último como el inventor principal. Estos motores asíncronos primitivos eran bifásicos (sistema eléctrico existente en los EEUU en la época), por lo que habría que esperar hasta 1890 a que Dolivo Dobrowolsky inventara el primer motor asíncrono trifásico.

En la máquina asíncrona la corriente que circula por uno de los devanados (generalmente el rotor) es inducida por el flujo magnético del otro, por lo que se denominan máquinas de inducción. La otra posible denominación (máquina asíncrona) se debe que la velocidad de giro del rotor no es la de sincronismo propia de la red.

2.2 Aspectos Constructivos

El motor de inducción está compuesto por dos partes principales diferenciadas separadas por un espesor de aire llamado entrehierro [Fraile Mora, J. 2008]:

- **Estator**

  Parte fija de la máquina, de forma cilíndrica hueca. El estator está rodeado y unido a la carcasa, mientras que en su interior irá alojada la otra parte del motor: el rotor. Está formado por chapas de acero al silicio apiladas (para disminuir las corrientes de Foucault) distribuidas en unas ranuras de su periferia interior, donde se alojará el devanado trifásico inductor (figura 1.1).

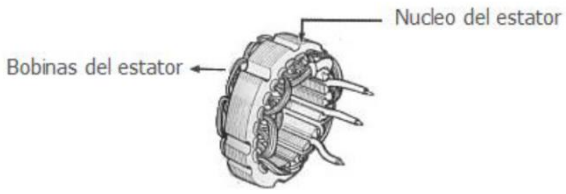


Figura 1.1: Estator de un motor de inducción. [Rodríguez Oraá, Bárbara Mª. 2013]

- **Rotor**

Es la parte móvil de la máquina, donde se inducirán corrientes debido al flujo magnético generado en el estator que acabarán produciendo en conjunto la rotación del mismo entorno a su eje axial. Está formado al igual que el estator por un cilindro de chapas apiladas, con unas ranuras en su periferia exterior (la más próxima al estator) donde se coloca el devanado. Según el devanado existen dos tipos posibles de rotor:

- **Rotor en jaula de ardilla o en cortocircuito:** conductores de cobre/aluminio macizos puestos en cortocircuito por dos anillos laterales (figura1.2). Es el más utilizado.

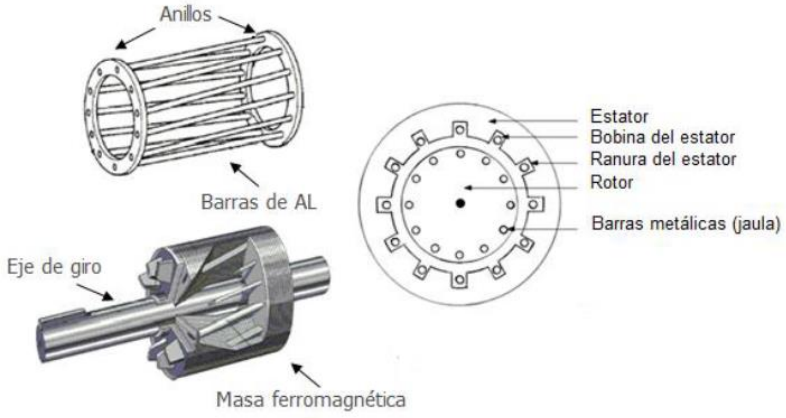


Figura 1.2: Rotor de jaula de ardilla. [Rodríguez Oraá, Bárbara Mª. 2013]

- **Roto devanado o con anillos:** arrollamiento trifásico unido en un extremo por una conexión en estrella, mientras que el extremo restante de las fases se conecta a unos anillos aislados entre sí (figura1.3). Esta configuración se usa para incluir resistencias externas al rotor y así poder variar ciertas características del mismo. Está en desuso debido a la electrónica industrial.

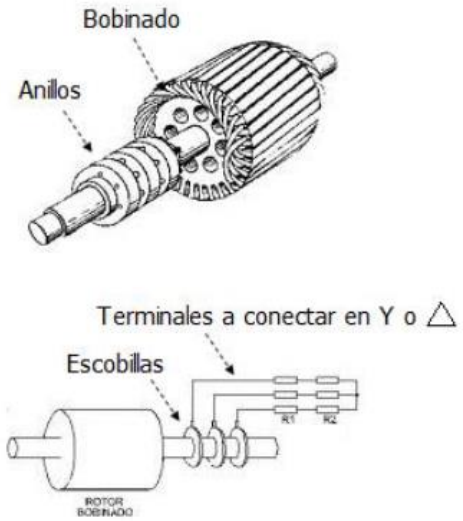


Figura 1.3: Rotor bobinado. [Rodríguez Oraá, Bárbara Mª. 2013]

Además del estator y del rotor, un motor de inducción deberá constar de otro elementos mecánicos y eléctricos para su funcionamiento: tapas, rodamientos, carcasa, ventilador (si fuera necesario por su diseño), caja de bornes, etc. A la caja de bornes se lleva la alimentación de red así como los extremos de los bobinados del estator, donde se conectarán según la disposición requerida (estrella o triángulo).

2.3 Principio de Funcionamiento

En este apartado nos referiremos al principio de funcionamiento de un motor de inducción trifásico. Supongamos que tenemos un estator constituido por tres devanados desfasados 120º eléctricos en el espacio y 2p polos. Estos devanados se conectan a una red trifásica de frecuencia f. Si el sistema está equilibrado por los devanados circularán corrientes alternas senoidales desfasadas 120º en el tiempo entre sí. Cada una de estas bobinas generará un campo magnético estático en el espacio, de dirección coincidente al eje magnético de la bobina y amplitud variable en el tiempo.

La combinación de estos tres campos pulsantes magnéticos se traducirá, según el *Teorema de Ferraris*, en un campo magnético giratorio de amplitud constante y velocidad de rotación constante, tal y como se ve en la figura 1.4[Fraile Mora, J. 2008].

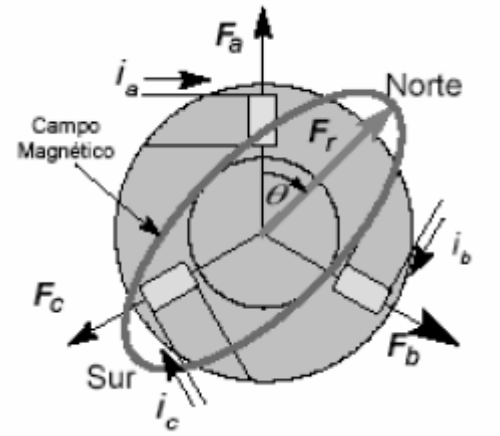


Figura 1.4: Distribución senoidal del campo magnético rotatorio. [Rodríguez Oraá, Bárbara Mª. 2013]

La velocidad de giro de este campo magnético se denomina *velocidad de sincronismo*, y viene dada por la ecuación 2.1:

$$n = \frac{60 \cdot f}{p} \qquad [rpm] \qquad (2.1)$$

Donde:

- n: velocidad de sincronismo.
- p: pares de polos.
- f: frecuencia de la red de alimentación.

Este campo magnético giratorio inducirá un f.e.m en el devanado del rotor debido a la *Ley de inducción eléctrica de Faraday* (ecuación 2.2 para el caso de un único conductor), lo que provocará la circulación de corrientes eléctricas por el mismo.

$$e = \int (v \times B) dl = (v \times B) \cdot L = -N \frac{d\Phi}{dt} \qquad (2.2)$$

Donde:

- e: fuerza electromotriz inducida.
- υ: velocidad relativa del rotor respecto al campo magnético.
- B: densidad de flujo magnético del campo.
- L: longitud del conductor.
- N: número de espiras del conductor.
- Φ: flujo magnético.

Las corrientes inducidas en el rotor tendrán un sentido tal que generen un campo magnético que se oponga a aquel que las ha producido (*Ley de Lenz*), lo que a su vez producirá en el rotor pares de fuerzas en las barras del rotor (ecuación 2.3: *Fuerza de Lorenzt*) que tendrán un momento resultante tal que provocará el giro del rotor en el mismo sentido que el campo magnético del estator, siguiendo a los polos móviles.

$$F = \int i \cdot (dl \times \mathrm{B}) = i \cdot (L \times \mathrm{B}) \qquad (2.3)$$

Donde:

- F: fuerza electromotriz inducida.
- i: corriente que circula por el conductor.
- B: densidad de flujo magnético del campo.
- L: longitud del conductor.

Este movimiento del rotor se producirá a una velocidad inferior a la del campo magnético (velocidad de sincronismo), ya que de igualarla desaparecería la causa que lo generó, por lo que estamos hablando de un régimen de giro asíncrono.

La fuerza electromotriz inducida en el rotor dependerá de la proximidad de la velocidad del rotor a la velocidad de sincronismo. Se obtendrá entonces una mayor fuerza electromotriz cuanto más nos alejemos del sincronismo, por lo que tendremos mayores corrientes inducidas y mayor par electromagnético en el motor. Para plasmar esta proximidad al sincronismo del que dependen muchas variables, se usa el parámetro conocido con el nombre de deslizamiento [Alonso Martínez, Alberto. 2015]:

$$s = \frac{n_1 - n}{n_1} \qquad (2.4)$$

Donde:

- s: deslizamiento.
- n: velocidad de giro del rotor.
- n_1: velocidad de sincronismo.

Este parámetro tiene valores del 3-8% en motores industriales funcionando a plena carga. Cuando aumenta la carga mecánica exigida al motor el par resistente supera al par interno,

aumentando el deslizamiento, lo que provoca a su vez un aumento de las corrientes del rotor y, por tanto, del par motor, restableciéndose el equilibrio dinámico de ambos momentos[Fraile Mora, J. 2008].

El deslizamiento nos da además la relación existente en cada momento entre las frecuencias de las corrientes del estator y las del rotor, tal y como se ve en la ecuación 2.5.

$$f_2 = sf_1 \tag{2.5}$$

Donde:

- s: deslizamiento.
- f_1: frecuencia de las corrientes del estator.
- f_2: frecuencia de las corrientes del rotor.

Se ha de decir también que los motores de inducción guardan especial parecido con los transformadores. El estator equivaldría al primario del transformador y el rotor (bloqueado) sería el secundario del transformador. Sin embargo, una diferencia importante se encuentra en el desfase de la f.e.m de cada espira del motor de inducción. Este desfase se produce como consecuencia de estar distribuidas las espiras a lo largo de las ranuras de la periferia del rotor. Por lo tanto, la f.e.m que se obtendrá en rotor es la suma vectorial de cada una de las f.e.m. generadas en cada espira. Mientras tanto en el transformador se tiene que las espiras forman un devanado concentrado que da lugar a una f.e.m en cada espira en fase con las demás [Alonso Martínez, Alberto. 2015].

Entrando más al detalle, en el análisis de las f.e.m tenemos las expresiones 2.6 y 2.7:

$$E_1 = 4.44K_1 f_1 N_1 \Phi_m \tag{2.6}$$

Donde:

- E_1: valor eficaz de la f.e.m. inducida por fase en el estator.
- K_1: coeficiente del devanado del estator.
- f_1: frecuencia de las corrientes del estator.
- N_1: número de espiras por fase del estator.
- Φ_m: flujo magnético máximo.

$$E_2 = 4.44K_2 f_2 N_2 \Phi_m = 4.44K_2 sf_1 N_2 \Phi_m \tag{2.7}$$

Donde:

- E_2: valor eficaz de la f.e.m. inducida por fase en el rotor.
- K_2: coeficiente del devanado del rotor.
- f_1: frecuencia de las corrientes del estator.
- f_2: frecuencia de las corrientes del rotor.
- s: deslizamiento.
- N_2: número de espiras por fase del rotor.
- Φ_m: flujo máximo que atraviesa el rotor.

Esta última f.e.m. producirá unas corrientes en el rotor de frecuencia f_2 que crearán a su vez un campo magnético giratorio, cuya velocidad respecto a su propio movimiento de rotación se obtendrá de la expresión 2.8, suponiendo que estator y rotor tienen el mismo número de polos:

$$n_2 = \frac{60 \cdot f_2}{p} \qquad (2.8)$$

Si unimos las expresiones 2.1 y 2.5 tendremos:

$$f_2 = sf_1 = \frac{n_1 - n}{n_1} \cdot \frac{p \cdot n_1}{60} = \frac{p(n_1 - n)}{60} \qquad (2.9)$$

que al compararlo con 2.8, y teniendo en cuenta que la velocidad del campo giratorio del rotor respecto a un punto de referencia fijo será n_2+n, llegamos a la expresión 2.10.

$$n_2 + n = (n_1 - n) + n = n_1 \qquad (2.10)$$

que nos indica que el campo del rotor gira en sincronismo con el campo del estator. Para que las f.m.m. de los devanados del estator y del rotor interaccionen produciendo un flujo resultante en el entrehierro, ambas f.m.m han de girar a la misma velocidad, por lo que la condición previamente propuesta de que estator y rotor tengan el mismo número de polos ha de ser una exigencia constructiva. [Fraile Mora, J. 2008]

2.4 Ventajas y desventajas

Las ventajas de los motores de inducción son varias:

- Construcción simple y robusta, sin problemas de estabilidad ante variaciones bruscas de la carga (el motor recupera la velocidad ante variaciones de carga). Concretamente, el rotor de jaula de ardilla es muy robusto.
- Principio de reversibilidad: Debido a este principio podemos hablar tanto de motores como de generadores asíncronos. En nuestro caso el punto de vista a tener en cuenta será el del motor, aunque no por ello se puede descartar su uso como generador.
- No es necesario el uso de escobillas o de elementos rozantes: Gracias a que no existe corriente conducida al rotor. Tienen par de arranque, y sólo una alimentación eléctrica, que se recibe a través de una línea trifásica. Excepto en el caso del rotor devanado (o bobinado).

Por el contrario, también tiene una serie de inconvenientes de los motores de inducción:

- Regulación complicada de la velocidad: Esto es debido a que es difícil que el motor mueva la carga a una velocidad fuera de la velocidad para la que estaba preparado, sin embargo mediante convertidores electrónicos de frecuencia se ha superado completamente este inconveniente.

- El motor absorbe una elevada intensidad en el arranque: Durante el arranque se absorbe una intensidad entre tres y seis veces mayor que la intensidad nominal, cuando el arranque es directo a la tensión nominal.

CAPÍTULO 3

FALLOS EN MOTORES DE INDUCCIÓN

3 FALLOS EN MOTORES DE INDUCCIÓN

3.1 Introducción

Debido a la importancia y uso extendido de este tipo de motores los distintos tipos de averías que se pueden producir a lo largo de su vida útil son factores de gran importancia: la avería puede producir tanto daños materiales como humanos, junto con una inevitable pérdida económica. Resulta por tanto de gran importancia conocer los distintos tipos de averías que se pueden producir para así poder adelantarnos al fallo crítico de la máquina y/o paliar las consecuencias en caso de que se produjera el mismo.

Estas averías surgen debido a la aparición de esfuerzos de distinta naturaleza en la máquina que superan en magnitud los parámetros de diseño con los que fue construida. Asimismo debido al envejecimiento de la propia máquina acabarán apareciendo fallos y averías inevitablemente, si bien el correcto mantenimiento y uso debería de evitar la aparición de fallos prematuros.

3.2 Fallos que afectan al estator y al rotor

3.2.1 Fallos en el estator

Los lugares del estator donde se pueden producir fallos son:

- **Núcleo magnético:**

  Está formado por láminas magnéticas apiladas separadas entre sí por aislante. La degradación de dicho aislante provoca la aparición de corrientes que generan un sobrecalentamiento localizado, lo cual puede producir a su vez el deterioro del asilamiento del devanado del estator.

  Se pueden producir por:

  - Montaje incorrecto.
  - Intrusión de elementos extraños durante el montaje.
  - Vibraciones del motor que afecten al aislamiento de los pernos de apriete del núcleo.

- **Aislamiento del devanado del estator:**

  Es donde se produce la mayoría de fallos del estator. Los distintos fallos que pueden aparecer vienen recogidos en la siguiente tabla.

Tabla 3.1: Tipo de fallos y sus consecuencias en los devanados del estator [Alonso Martínez, Alberto. 2015]

TIPO DE FALLO	CONSECUENCIA
Cortocircuito entre espiras	Funcionamiento por tiempo limitado.
Cortocircuito entre bobinas de la misma fase	Funcionamiento por tiempo limitado.
Cortocircuito entre fases	Fallo del motor. Actuación del equipo de protección.
Derivación fase-tierra	Fallo del motor. Actuación del equipo de protección.
Circuito abierto en una fase	Funcionamiento o no según carga y circuito de protección.

Todos estos fallos pueden originarse por esfuerzos/solicitaciones de muy diversa naturaleza, que se exponen en los siguientes puntos [Alonso Martínez, Alberto. 2015].

3.2.1.1 Esfuerzos eléctricos

- Descargas parciales
  - *Descargas internas*

    Durante el proceso de fabricación del aislamiento se crean burbujas de aire dentro del mismo debido a la imposibilidad de la resina aislante de ocupar todo el volumen del aislamiento, a la evaporación de volátiles de la misma, o a la deslaminación.

    Cuando el campo eléctrico que afecta a estas burbujas iguala o supera la rigidez dieléctrica del aire se producirán descargas eléctricas, cuya acción continuada podrán ocasionar la perforación del aislamiento.

  - *Efecto corona*

    El efecto corona es la ionización del aire circundante a los bobinados. Este efecto se produce cuando el gradiente de tensión entre el conductor y el aire que le rodea supera el valor de rigidez dieléctrica del aire, apareciendo cuando se alcanzan valores de 21kV/cm para condiciones normales de presión y temperatura.

    Inicialmente este efecto sólo tiene carácter sonoro, si bien a medida que el gradiente de tensión aumenta lo mismo ocurre con la intensidad de este efecto, produciéndose el aumento de la zona ionizada así como el efecto visual del mismo, que es la aparición de una corona violácea que rodea al conductor.

    Finalmente se puede ocasionar un arco eléctrico entre el conductor y los elementos sin tensión de la máquina, que puede producir la destrucción de los aislamientos.

    Las zonas críticas son las cabezas de las bobinas y las zonas donde se separan estas del núcleo magnético, produciéndose distorsiones en el campo. Para impedir la aparición del efecto corona se asegura la uniformidad del campo eléctrico por

medio de pinturas semiconductoras (usualmente de grafito) aplicadas en la superficie exterior de las bobinas.

- *Efecto Tracking*

  Es la aparición de un camino eléctrico superficial en los aislantes de tal manera que la corriente circula libremente del conductor a la masa. Estas corrientes originan puntos calientes en el aislante que lo deterioran.

  Este efecto se produce cuando la superficie del aislante está sometida a un gran estrés eléctrico, o cuando está recubierta por humedad, polvo, o cualquier otro tipo de contaminación, dándose principalmente en las cabezas de las bobinas.

- **Histéresis dieléctrica**

Produce unas pérdidas en el aislamiento en forma de calor, el cual si no es disipado correctamente provocará un aumento localizado de la temperatura del aislante que podrá llegar a perforar al mismo.

- **Transitorios eléctricos**

Son sobretensiones de duración reducida que pueden acabar dañando tanto al aislamiento como al bobinado. Se originan debido a múltiples motivos: interrupciones, conmutaciones, cortocircuitos, descargas eléctricas, etc.

3.2.1.2 Esfuerzos mecánicos

- **Fuerzas electromagnéticas**

Estas fuerzas generadas por las corrientes de los bobinados alcanzan grandes valores durante arranques, cortocircuitos y conmutaciones, provocando una oscilación en las bobinas, lo que a su vez puede ocasionar desde microfracturas a rupturas mecánicas importantes en los aislamientos.

Son de más magnitud en motores de media y alta tensión y en motores de alta velocidad.

- **Vibraciones**

Intrínsecas de cualquier máquina rotativa, causan la fatiga, rotura y separación de las láminas de mica presentes en los materiales aislantes.

Si los bobinados no están apropiadamente fijados pueden deteriorarse debido al rozamiento con otros elementos de la máquina.

- **Rozamiento rotor-estator**

Ya sea por contacto intermitente/parcial (durante una fracción del giro) o permanente/completo, siendo el más habitual el primero. Asimismo el roce puede ser radial, axial o combinado desde el punto de vista de la dirección de la fuerza normal que se produce.

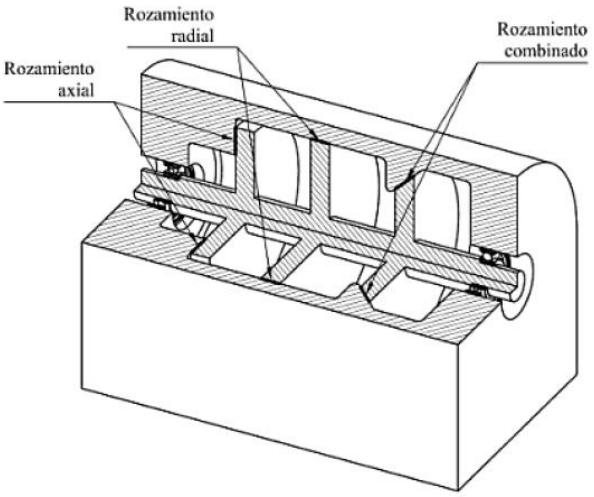


Figura 3.1: Tipos de rozamiento estator-rotor. [Rodríguez Oraá, Bárbara Mª. 2013]

Se debe principalmente a la existencia de otro tipo de problemas como desalineación, el fallo de los rodamientos, desequilibrio del campo magnético, doblamiento del eje, resonancia, dilataciones diferenciales, etc., que provocan un cambio en la trayectoria del rotor.

Como este rozamiento suele tener lugar entre superficies no lubricadas los daños ocasionados suelen ser de consideración importante.

- **Introducción de objetos extraños**

Objetos introducidos accidentalmente al entrehierro a través del sistema de ventilación pueden ocasionar daños en el aislamiento, en las chapas magnéticas así como en el propio conductor tanto de las barras como de los bobinados y anillos.

La gravedad de los daños dependerá del tamaño del objeto así como del espesor del entrehierro existente en la máquina.

3.2.1.3 Esfuerzos térmicos

- **Sobrecarga térmica**

Afectan negativamente a la vida del aislante. Pueden producirse durante los arranques, en régimen permanente o en condiciones de rotor bloqueado, siendo ésta última la que puede provocar los daños más importantes.

Las causas más habituales para las sobrecargas térmicas son las siguientes:

- Arranques.
- Ventilación y/o refrigeración insuficientes.
- Aumento del par resistente.
- Desequilibrio entre fases.
- Variaciones en las tensiones de alimentación.
- Temperatura ambiente muy elevada.

- **Dilatación térmica**

  La diferencia entre los coeficientes de dilatación térmicos del conductor y de los materiales que componen el aislante puede provocar incrementos de temperatura así como el deslizamiento entre capas de material, pudiendo acabar en rotura de las láminas del aislante.

  El deslizamiento entre conductor y aislante puede causar la aparición de bolsas de aire que podrían ocasionar descargas parciales, generando daños.

- **Envejecimiento térmico**

  Afecta principalmente a los aislantes. Sigue la ley de Montsinger, por la que por cada incremento de 10°C en la temperatura su vida útil se reduce a la mitad. Aproximadamente se puede determinar la vida del aislamiento mediante la ley de Arrhenius:

$$L = A \cdot e^{-b \cdot T} \qquad (3.1)$$

  Donde:

  - L: vida del aislamiento [h].
  - A: constante que depende de la clase térmica del conductor[h].
  - b: constante que depende de la clase térmica del conductor [K^{-1}].
  - T: temperatura absoluta de funcionamiento [K].

  Para reducir los efectos de este envejecimiento se suele incrementar la clase de material aislante así como monitorear la temperatura de operación.

3.2.1.4 Esfuerzos de origen ambiental

- **Contaminación**

  El polvo, agua, aceite, etc. que se deposita sobre las superficies de los aislamientos acaba formando una capa que dificulta la disipación de calor y favorece la aparición de corrientes, reduciéndose la vida de los mismos.

  Asimismo esta contaminación afecta negativamente a la grasa de los rodamientos, degradándola.

- **Condensación**

  La condensación de la humedad produce una pérdida de las propiedades del aislamiento y la posible aparición de derivaciones a masa.

- **Altitud**

  Según la norma CEI 60034-1 los motores están diseñados para trabajar a potencia nominal a una altitud menor de los 1000m, por lo que a una altitud mayor se produce un aumento de las temperaturas de trabajo, reduciendo la vida de los aislantes.

- **Temperatura**

  Según la norma CEI 60034-1 los motores están diseñados para trabajar a potencia nominal en ambientes con temperaturas comprendidas entre los -15°C y los 40°C. Para temperaturas superiores se reduce la vida útil del aislamiento, mientras que a temperaturas inferiores aparecen problemas de condensación.

3.2.2 Fallos en el rotor

Según el tipo de rotor se presentarán unas problemáticas u otras. Los fallos aparecerán en el núcleo magnético (fallos comunes para cualquier motor de inducción) y en el bobinado o en la jaula y/o anillos de cortocircuito, según sea el motor.

Los fallos en el núcleo magnético son tales como agrietamientos por altas temperaturas o corrosión debido a impurezas [Alonso Martínez, Alberto. 2015].

- **Rotor bobinado:** los fallos se suelen presentar en el núcleo magnético o en el bobinado, que es semejante al del estator pero de mayor rigidez debido a las fuerzas centrífugas a las que está sometido, por lo que los fallos que se presentan en el mismo serán semejantes a los que se presentan en el estator.

- **Rotor de jaula de ardilla:** los fallos se suelen presentar en el núcleo magnético, en las barras o en el anillo cortocircuitado. Los fallos en barras y anillo aparecen principalmente durante el arranque debido a dos fenómenos:
  - Grandes temperaturas alcanzadas.
  - Grandes esfuerzos al final de las barras, en su conexión con el anillo de cortocircuito.

    Esto es debido a que la corriente de arranque circula por el exterior del conductor de las barras debido a un efecto pelicular, reduciendo su superficie útil y provocando que las barras se expandan axialmente más por dicha zona.

  Estos dos fenómenos se ven producidos o agravados principalmente por errores y/o malos diseños durante su fabricación, como son: elección incorrecta de los materiales de la jaula, malas uniones entre las barras de la jaula y los anillos de cortocircuito, o la utilización de una matriz del rotor de fundición de baja calidad.

3.2.2.1 Esfuerzos electromagnéticos

- **Efecto electromagnético**

  La existencia conjunta de campos magnéticos y corrientes eléctricas tiene como efecto la aparición de fuerzas electromagnéticas unidireccionales, que provocan flexiones en las barras, dando lugar a su vez a vibraciones. Estas vibraciones pueden producir finalmente la rotura por fatiga de la barra y, si la vibración es excesiva, efectos de chispeo que erosionen el material.

  Las mayores vibraciones aparecen en el arranque del motor, debido al gran valor de la corriente.

- **Atracción electromagnética desequilibrada**

  Idealmente el rotor se encuentra perfectamente centrado y las fuerzas electromagnéticas que se le aplican están exactamente balanceadas.

  En la realidad el rotor no se encuentra centrado (peso, desgaste en rodamientos, alineación imperfecta, etc.), lo que origina que haya una zona donde la distancia del entrehierro sea menor de la deseada y otra donde esta distancia sea mayor. En la zona de menor entrehierro el flujo magnético será mayor debido a la menor reluctancia, y en la de mayor entrehierro se producirá el efecto contrario.

  Todo esto origina la aparición de un sistema desequilibrado de fuerzas que puede producir a la larga la flexión del rotor, que en el peor de los casos puede acarrear el contacto físico entre rotor y estator.

3.2.2.2 Esfuerzos dinámicos

- **Fuerzas centrífugas**

  El rotor está diseñado para no sobrepasar una velocidad máxima de giro de cara a evitar el desplazamiento del eje, las chapas magnéticas o de la jaula respecto al núcleo del rotor.

  De cara a evitar este tipo de problemas se trabaja con sistemas de control de la velocidad

- **Esfuerzos transitorios**

  En ocasiones el eje del motor se ve sometido a esfuerzos mecánicos que sobrepasan los valores asociados al funcionamiento normal de régimen permanente, asociados a los arranques o al bloqueo de los accionamientos.

  Por diseño el motor deberá de soportar los sobreesfuerzos que se deban a condiciones normales de funcionamiento (arranques), pero cualquier esfuerzo que supere los esfuerzos máximos de diseño puede acarrear fallos graves en el motor.

- **Esfuerzos cíclicos**

  Se producen por desalineamiento del eje, eje flexionado, carga fluctuante, holgura de los rodamientos, correas de transmisión sobretensadas, etc.

  La acción de esfuerzos cíclicos acarrea fallos por fatiga del material, que pueden tener diversos grados de severidad.

3.2.2.3 Esfuerzos térmicos

- **Desequilibrio térmico**

  Puede ser debido al propio proceso de fabricación o a condiciones de operación que excedan los valores de diseño. Es sencillo identificar los desequilibrios térmicos, no así encontrar su causa.

Las causas más habituales de fallos por este desequilibrio son las siguientes:

- Arranques frecuentes del motor.
- Gradientes de temperatura debido a corrientes desequilibradas producidas por rotura de barras.
- Transferencia de calor desigual entre el centro y las barras.

- **Sobrecarga térmica**

Puede producirse durante el arranque, en régimen permanente o en situaciones excepcionales como la de rotor bloqueado, que es excepcionalmente peligrosa en este caso.

Las causas más habituales de fallas por este desequilibrio son las siguientes:

- Gran número de arranques consecutivos.
- Rotor bloqueado.
- Fricción entre rotor y estator.
- Rotura de barras del rotor.
- Ventilación insuficiente.

- **Chispeos**

Pueden tener carácter no destructivo, ocurriendo incluso durante el régimen de funcionamiento nominal. Sin embargo, el chispeo preocupante tiene lugar durante el arranque y por motivo de la vibración de las barras del rotor, que provocan respectivamente la circulación de corriente por el núcleo magnético y la aparición de arcos eléctricos entre barras y láminas del núcleo.

Las chispas que se observan en el entrehierro son partículas incandescentes procedentes del núcleo magnético o de las barras. El chispeo desaparecerá tras varios ciclos de arranque si estas partículas se produjeron durante la fabricación o montaje, pero si su origen es el chispeo intermitente puede que se mantengan durante toda la vida útil del motor.

- **Puntos calientes y pérdidas**

Son lugares del motor en los que aparecen temperaturas superiores a las esperadas y a las de su entorno. Son debidos a una fabricación del motor que se aleja del diseño ideal o a un mal diseño:

- Diseño incorrecto de la laminación del núcleo magnético.
- Holguras no constantes.
- Uniones defectuosas.
- Chispas en las laminaciones.
- Distribución de pérdidas no uniforme.
- Barras y láminas acortadas desigualmente.

Son problemas que se pueden detectar en los procesos de control de calidad durante la fabricación del motor, así como durante el funcionamiento del mismo a través de

análisis de corriente, de vibraciones, etc.

- **Esfuerzos de origen ambiental**

  Esfuerzos adicionales originados por la entrada de objetos o partículas externos, que pueden provocar abrasión o corrosión, así como impedir el correcto funcionamiento del sistema de ventilación.

3.2.2.4 Fallos por rotura de barras

Debido a su importancia, tanto por gravedad como por estar orientado este documento a la detección del mismo en gran medida, en este punto se hablará con más detalle de la problemática de la rotura de barras del rotor, que afecta a los motores de inducción de rotor de jaula de ardilla.

El agrietamiento y rotura de barras del rotor es uno de los principales fallos de los motores de inducción, especialmente en aquellos que arrancan frecuentemente bajo carga.

Durante el arranque circula una corriente muy elevada por las barras del rotor debido a que la velocidad de éste es mucho menor que la velocidad síncrona, lo que origina un fuerte calentamiento y una expansión de las barras con relación al rotor. Debido a que la resistencia eléctrica de cada una de las barras es diferente el calentamiento y expansión en cada una de ellas será diferente a su vez, llevando todo ello al agrietamiento de las juntas donde las barras se unen al anillo de cortocircuito [Rodríguez Oraá, Bárbara Mª. 2013].

La aparición de una grieta aumenta la resistencia eléctrica de la barra correspondiente, lo cual incrementa el calentamiento y agrava la grieta. En las barras vecinas las corrientes aumentan debido a que en la barra con grieta se ha reducido, lo cual produce la rotura en cadena de la barras.

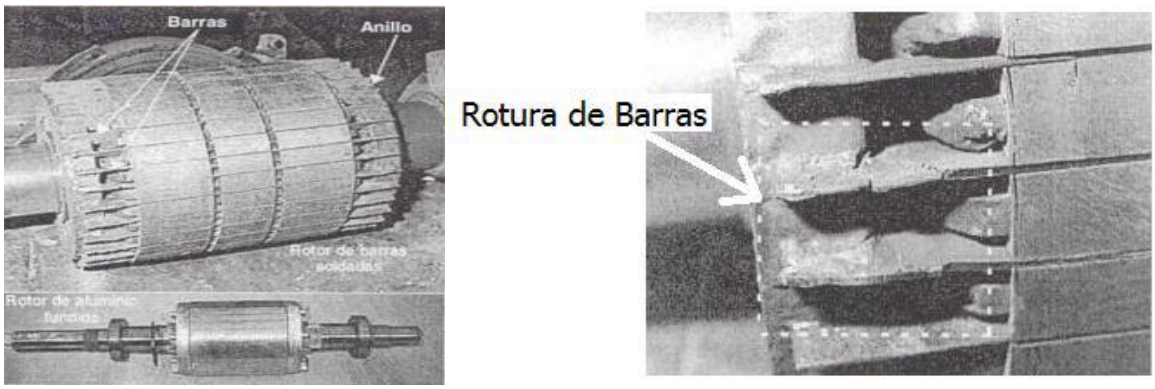


Figura 3.2: Imagen izquierda: rotores de motores de inducción (superior de cobre e inferior de aluminio fundido). Imagen derecha: Barras rotas en la proximidad del anillo de cortocircuito. [Rodríguez Oraá, Bárbara Mª. 2013]

3.3 Fallos en el eje

Si bien hay múltiples causas que originan fallos en el eje de los motores de inducción (fatiga, corrosión, sobrecarga, etc.), principalmente es debido a la fatiga.

Los fallos por fatiga se pueden diferenciar según el tipo de carga soportada:

- **Fatiga axial**

  Si bien la carga axial probablemente provoque el fallo en los rodamientos, originando en ellos el descorchamiento en el camino de rodadura (spalling).

- **Fatiga flectora**

  Fallos debidos a la alternancia de los esfuerzos de compresión y tracción en del eje.

- **Fatiga torsional**

  Fallos relacionados con el valor del par transmitido por el eje.

La zona crítica de los fallos se encuentra en los cambios bruscos de sección, ya sea por tamaño o por geometría.

3.4 Fallos por excentricidad

Es una de las causas más habituales de fallo del rotor, sólo detrás de la rotura de barras.

La excentricidad conlleva un entrehierro no uniforme, lo que origina un campo magnético desequilibrado (asimétrico), vibraciones, ruido, introducción de armónicos en la corriente de alimentación, circulación de corriente eléctrica a través de los rodamientos, y rozamiento entre rotor y estator en los casos más extremos. Por todo esto la detección de este tipo de fallos es vital de cara a la vida útil del motor [Alonso Martínez, Alberto. 2015].

El campo magnético asimétrico es efecto y una de las causas de este fenómeno.

Existen cuatro tipos de excentricidades:

- Excentricidad estática.
- Excentricidad dinámica.
- Excentricidad mixta.
- Excentricidad axial.

3.4.1 Excentricidad estática

En este caso el centro de rotación coincide con el centro geométrico del rotor, estando desplazado respecto al centro geométrico del perímetro interno del estator. El espesor del entrehierro dependerá únicamente del ángulo de giro del rotor, por lo que la posición de mínimo entrehierro se encuentra fija en el espacio.

Esta excentricidad genera una atracción magnética desequilibrada estable en una dirección, que puede ocasionar la flexión del eje del rotor, desgaste en los rodamientos, así como producir la excentricidad dinámica.

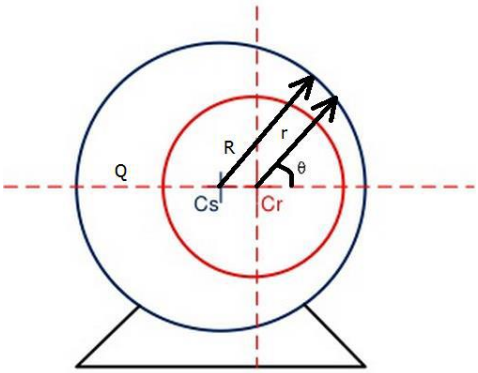


Figura 3.3: Excentricidad estática (Q=f(θ)).[Rodríguez Oraá, Bárbara Mª. 2013]

Las causas de una excentricidad estática son:

- Núcleo del estator de forma ovalada.
- Desalineación de los rodamientos.
- Desgaste de los rodamientos.
- Acoples mecánicos desalineados.
- Excesiva tolerancia.

3.4.2 Excentricidad dinámica

El centro de rotación coincide con el centro geométrico del perímetro interno del estator, no coincidiendo con el centro geométrico del rotor. El espesor del entrehierro será variable en el tiempo y no uniforme, dependiendo del ángulo de giro del rotor y del tiempo, por lo que las posiciones de mínimo entrehierro giran con el rotor.

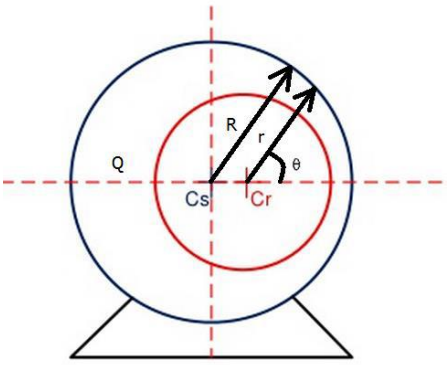


Figura 3.4: Excentricidad dinámica (Q=f(θ, t)).[Rodríguez Oraá, Bárbara Mª. 2013]

Las causas de una excentricidad dinámica son:

- Velocidad de giro del centro del rotor diferente a la de la máquina.
- Excentricidad estática que haya derivado en la excentricidad dinámica.
- Resonancia mecánica a velocidad crítica.

3.4.3 Excentricidad mixta

Es la superposición de las excentricidades estática y dinámica. Es lo que suele ocurrir en las máquinas, donde el rotor no gira alrededor de su centro geométrico ni alrededor del centro

geométrico del estator, si no alrededor de algún punto situado entre ambos centros.

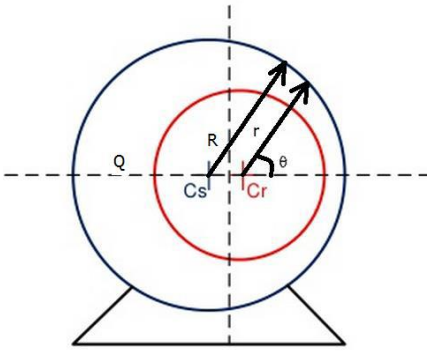


Figura 3.5: Excentricidad mixta. [Rodríguez Oraá, Bárbara Mª. 2013]

3.4.4 Excentricidad axial

La excentricidad axial no es más que la representación de las anteriores excentricidades a lo largo del eje de la máquina. Se da en los casos donde el eje de giro del rotor no es paralelo al eje geométrico del estator y da lugar a una excentricidad distinta en cada sección del eje.Por tanto, el eje de giro del rotor no es paralelo al eje geométrico del estator y da lugar a una excentricidad distinta en cada sección.

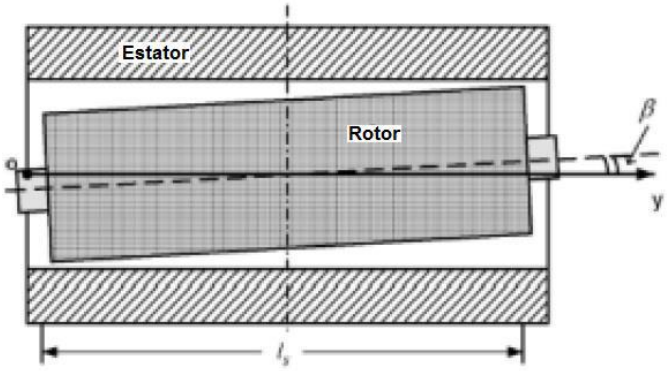


Figura 3.6: Excentricidad axial. [Alonso Martínez, Alberto. 2015]

3.5 Fallos en los rodamientos

Loas rodamientos son elementos mecánicos que sirven de apoyo y reducen la fricción entre dos piezas por medio de la rodadura. Se colocan entre dos componentes de una máquina con un eje de rotación común, por lo que existe un movimiento relativo entre ambos elementos. Los componentes principales de los rodamientos se muestran en la figura 3.7.

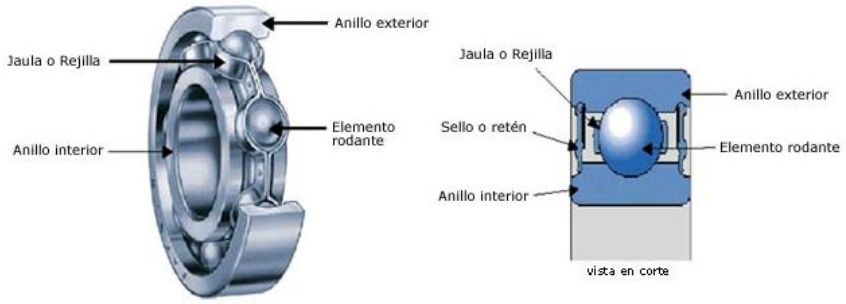


Figura 3.7: Elementos constitutivos de un rodamiento de bolas. [Manuales SKF. 2016]

Los elementos rodantes pueden tener múltiples formas, tales como bolas, cilindros o conos truncados, que se utilizarán según la aplicación lo requiera. Estos elementos, así como los anillos, se fabrican en aceros duros con gran resistencia a la fatiga, mientras que para las jaulas se recurre a aceros más blandos, latón, bronce, resina fenólica o poliamida, ya que su función únicamente es la de mantener los elementos rodantes en las pistas de rodadura.

En la figura 3.8 se recogen las causas de fallo en rodamientos así como su frecuencia relativa, que se procederán a explicar posteriormente.

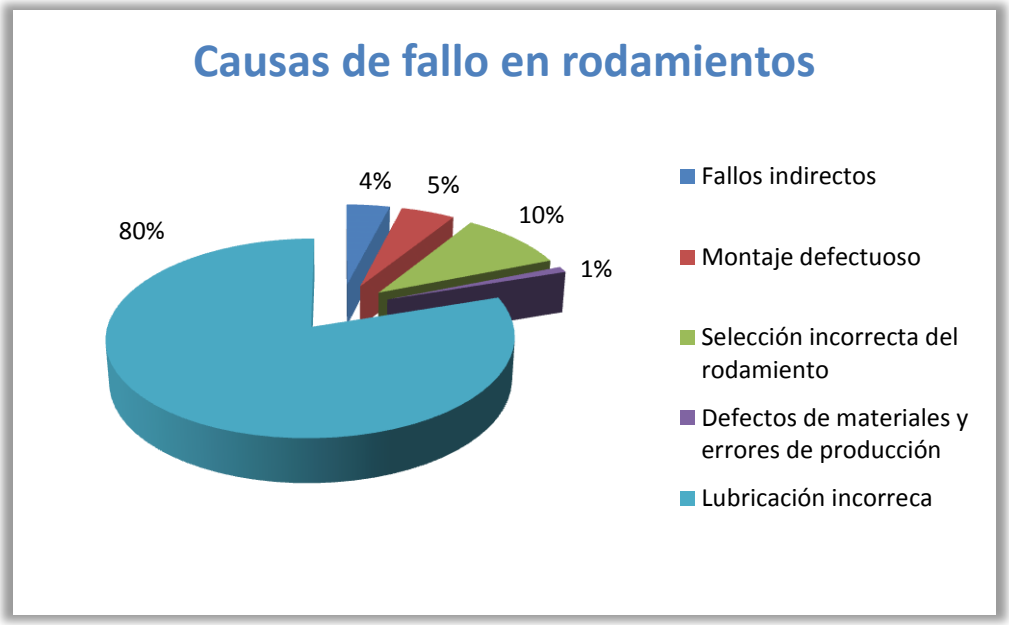


Figura 3.8: Causas de fallo en rodamientos. [Alonso Martínez, Alberto. 2015]

3.5.1 Fallos indirectos

- **Sobrecarga**

  Las precargas incorrectas, ajustes prietos o cargas excesivas pueden acarrear la fatiga prematura de los elementos de rodadura.

Figura 3.9: Deformación plástica del anillo interior de un rodamiento por sobrecarga. [Alonso Martínez, Alberto. 2015]

- ## Descarga eléctrica

Distribuciones asimétricas de corriente en el rotor o en el estator, así como fenómenos electrostáticos, inducen tensiones en el eje del motor que originan corrientes que circularán por el mismo y por los rodamientos.

Estas corrientes provocan el desprendimiento de partículas metálicas, así como la degradación del lubricante. El desprendimiento de material se refleja visualmente como pequeños cráteres, o marcas paralelas al eje si las corrientes son pequeñas, en el camino de rodadura (figura 3.10).

Figura 3.10: Consecuencias de la descarga eléctrica en el anillo interior de un rodamiento.[Alonso Martínez, Alberto. 2015]

- ## Vibración

Las vibraciones estacionarias pueden producir daños en los rodamientos, que visualmente se muestran como depresiones brillantes o manchas rojizas comunes a la corrosión. Las marcas guardan una separación entre ellas idéntica a la separación existente entre los elementos rodantes.

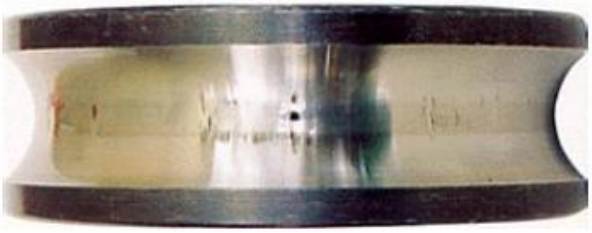


Figura 3.11: Consecuencias de las vibraciones en el anillo interior de un rodamiento. [Alonso Martínez, Alberto. 2015]

- **Alta temperatura**

  Son un factor de riesgo para la vida del rodamiento en tanto en cuando afectan al comportamiento y estabilidad del lubricante. En función del tipo de lubricante elegido existe una temperatura a partir de la cual éste empieza a oxidarse (71°C en el caso de aceites minerales de gran calidad).

3.5.2 Montaje defectuoso

Un desalineamiento de 0,01/10mm es suficiente para causar un aumento importante de la vibración y de la temperatura del rodamiento, lo que se traduce en un gran desgaste del mismo [Alonso Martínez, Alberto. 2015].

Además, en caso de montajes que utilicen calentadores de inducción para la expansión del anillo interior será necesario desmagnetizar el rodamiento antes de su instalación para evitar el fallo debido a la atracción magnética.

Otras posibles causas de un montaje defectuoso son una temperatura excesiva o irregular del rodamiento, así como una carga desequilibrada.

3.5.3 Selección incorrecta del rodamiento

Esta posibilidad ha de ser considerada después de analizar el resto de causas de fallos, ya que se presupone un diseño de la instalación correcto.

3.5.4 Defectos de materiales y errores de producción

Este tipo de causa cada día tiene menor importancia, debido a la mejora de la tecnología de materiales, de los procesos de fabricación y de los instrumentos de control de calidad de la industria.

3.5.5 Lubricación incorrecta

La correcta lubricación de los rodamientos es fundamental para reducir la fricción entre los componentes y, por tanto, la generación de calor, siendo además la principal causa de las averías producidas en los rodamientos. Otras funciones complementarias de la lubricación son la disipación de calor, eliminación de partículas sólidas desgastadas de la superficie de rodadura o protección contra la corrosión.

Las causas de una mala lubricación pueden ser múltiples, tal y como se muestra a continuación y se recoge en la figura 3.12.

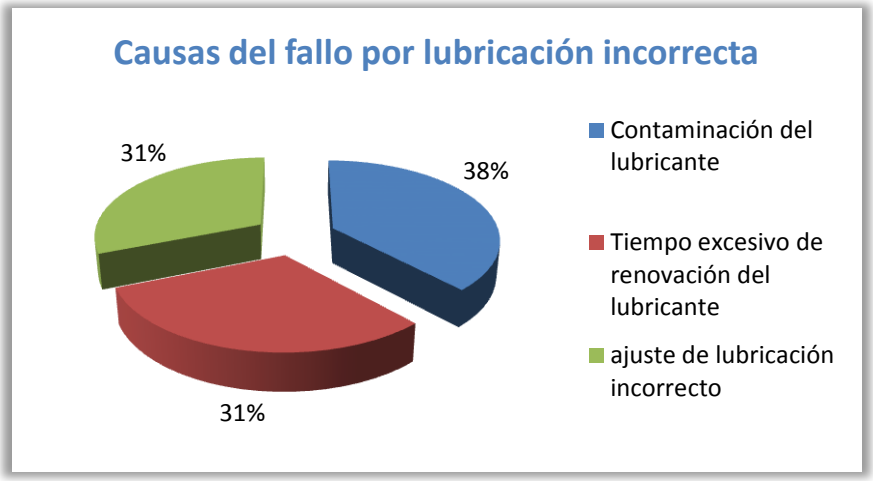


Figura 3.12: Causas de fallo en rodamientos. [Alonso Martínez, Alberto. 2015]

- **Tiempo excesivo sin renovación del lubricante**
- **Ajuste incorrecto de lubricación**

Es resultado de la selección de un refrigerante que no se adapta a las condiciones de operación. Las principales causas de un ajuste incorrecto son a su vez:

- Viscosidad del lubricante incorrecta.
- Cantidad de lubricante insuficiente
- Cantidad de lubricante excesiva

- **Contaminación del lubricante**

La contaminación del lubricante por partículas metálicas desgastadas origina la aparición de microgrietas en el rodamiento que pueden generar esfuerzos locales, reduciéndose la vida del mismo.

Esta reducción de vida depende de diversos factores:

- Tipo, dureza, tamaño y cantidad de partículas contaminantes.
- Ancho de la capa de lubricante.
- Tamaño del rodamiento.

CAPÍTULO 4

ESTRATEGIAS DE MANTENIMIENTO EN MOTORES DE INDUCCIÓN

4 ESTRATEGIAS DE MANTENIMIENTO EN MOTORES DE INDUCCIÓN

4.1 Introducción

Se entiende por mantenimiento la actividad humana que garantiza la funcionalidad y prestaciones de un sistema o dispositivo dentro de una calidad esperada.

Su objetivo, por tanto, es el de prolongar la vida útil de los equipos e instalaciones, afectando en la menor medida de lo posible a su servicio (las interrupciones suponen por si mismas un costo en las actividades económicas, ya sea por el tiempo de parada o por la necesidad de contar con duplicidades de equipos).

Todo buen mantenimiento debe:

- Efectuar las inspecciones y reparaciones con la mayor brevedad posible y utilizando los métodos apropiados.
- Sugerir y proyectar mejoras con el objeto de disminuir la probabilidad de futuras averías.
- Controlar y ajustar el coste directo del mantenimiento. Esto se logra mediante un uso correcto y eficiente del tiempo, materiales, servicios y personal humano.

Todo esto lleva a que la tarea del mantenimiento tenga que mantener en equilibrio dos principios fundamentales: el de fiabilidad y el de coste. Un mantenimiento muy frecuente e intrusivo (interrupciones) originará una gran fiabilidad del motor durante su funcionamiento pero acarreará muchos costes. Por el contrario, un mantenimiento poco frecuente tendrá poco coste directo pero repercutirá en la fiabilidad y vida útil de los equipos.

4.2 Mantenimiento de motores de inducción

Para el caso específico de motores de inducción, las funciones concretas a realizar para llevar un correcto mantenimiento de los mismos son las siguientes:

- Planificación del mantenimiento: tanto del motor, como de equipos eléctricos e instalaciones relacionadas.
- Ejecución del mantenimiento previamente planificado.
- Control, documentación y seguimiento de todo el proceso de mantenimiento.
- Evaluación del mantenimiento realizado.
- Reducción al mínimo de los costes de mantenimiento dentro de la eficiencia y condiciones de operación deseadas, buscando que la vida útil de los dispositivos sea la más adecuada, lo cual no siempre es sinónimo de máxima (obsolescencia, costes muy elevados, etc.).
- Investigación de las causas de las averías, y desarrollo de planes de actuación.

Hay que tener en cuenta que los motores de inducción tienen una importancia crítica en los procesos productivos, por lo que una avería en ellos puede provocar el paro parcial o total del

proceso si no se han tomado las medidas adecuadas.

Hoy en día existen multitud de herramientas y técnicas de gestión y mantenimiento de equipos, el uso de las cuales repercute positivamente en el mantenimiento. Estas técnicas pueden ser herramientas de soporte de decisiones, monitorización de equipos, diseño de equipos orientado a un sencillo mantenimiento, etc.

4.3 Estrategias de mantenimiento

La mayor exigencia de calidad ha motivado que se destinen cada vez más recursos a los planes de mantenimiento, y a que se utilicen nuevas técnicas de mantenimiento. Las estrategias de mantenimiento que se tratarán en este apartado se recogen en la figura 4.1.

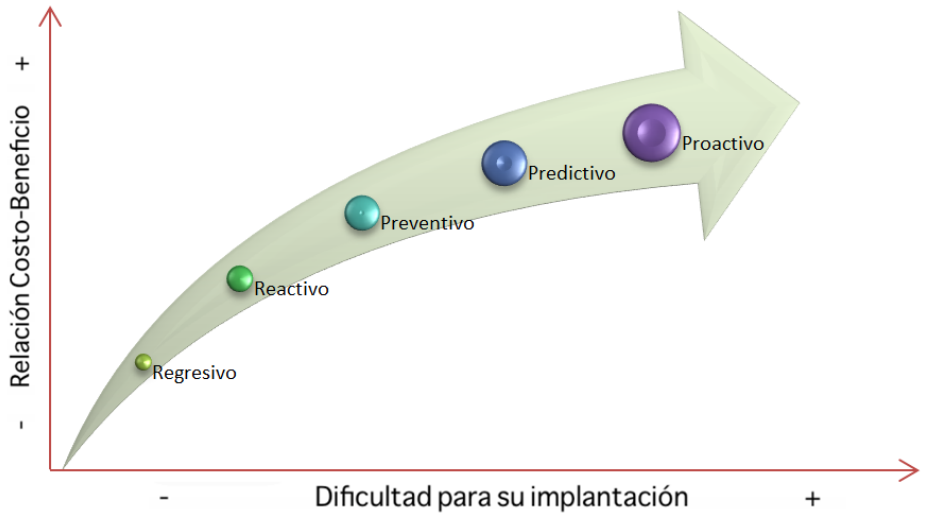


Figura 4.1: Estrategias de mantenimiento. Se muestra la relación que guardan respecto a la dificultad y a la relación costo-beneficio de su implantación. [Rodríguez Oraá, Bárbara Mª. 2013]

Por otra parte, las intervenciones de mantenimiento pueden clasificarse como programadas o no programadas. En la figura 4.2 se muestra a qué tipo corresponden las tres estrategias de mantenimiento más utilizadas: mantenimiento reactivo, preventivo y predictivo [Rodríguez Oraá, Bárbara Mª. 2013].

Figura 4.2: Tipos de intervenciones de mantenimiento según se trate de un mantenimiento reactivo, preventivo o predictivo. [Rodríguez Oraá, Bárbara Mª. 2013]

4.3.1 Mantenimiento regresivo

Realmente no se puede considerar una estrategia, ya que no existe ninguna previsión de la aparición de averías. Sigue la filosofía de "usar y tirar", donde la reparación es antieconómica. Es un método muy utilizado, ya sea por no haber analizado correctamente las ventajas de seguir otra estrategia de mantenimiento o porque para algún caso concreto sea la estrategia más económica y preferible.

4.3.2 Mantenimiento reactivo o correctivo

Se basa en no programar tareas de mantenimiento hasta la aparición de averías en el motor que puedan acabar en la interrupción del proceso. Se puede llevar a cabo en el mismo instante de detección de la avería o programarse para un momento futuro más adecuado.

A su favor tiene su fácil implementación, la gran eficiencia que mantiene sobre sus activos y los ahorros en el aspecto de la planificación. Por contra, es un método sólo adecuado para activos no críticos ya que puede provocar interrupciones de la producción y problemas de seguridad o emisiones, puede dar lugar a reparaciones de baja fiabilidad, exige grandes almacenes de repuestos, y requiere de personal cualificado dedicado en exclusividad a solventar emergencias.

4.3.3 Mantenimiento preventivo

Consiste en la realización de paradas planificadas durante las cuales se sustituyen los elementos más críticos y susceptibles a averías. El elemento crítico en esta estrategia de mantenimiento es la determinación de los períodos de tiempo entre paradas, que se determinan utilizando criterios estadísticos [Rodríguez Oraá, Bárbara Mª. 2013].

El objetivo de este mantenimiento es minimizar la probabilidad de avería o pérdida de rendimiento de las máquinas, maximizándose por tanto la calidad y seguridad de la producción, a la par que se planifican las intervenciones de manera que se ajusten en la medida de lo posible a la vida útil del elemento intervenido. De esta forma se reducen las paradas no planificadas por avería.

Como aspecto negativo importante hay que destacar que no se aprovecha en su totalidad la vida útil de los elementos, lo cual genera un coste económico y de tiempo.

Ciertos estudios indican que hasta el 90% de los componentes sustituidos mediante este mantenimiento podría seguir prestando servicio satisfactoriamente, si bien el ahorro medio respecto al mantenimiento correctivo alcanza valores del 60%.

4.3.4 Mantenimiento predictivo

Consiste en el seguimiento continuo (supervisión) de ciertas variables que determinan el estado del sistema, comparándose con patrones preestablecidos para determinar el instante óptimo en el que realizar la intervención de mantenimiento.

Requiere una fuerte inversión inicial, tanto en equipos como en técnicos, si bien es la que proporciona los mejores resultados ya que maximiza el tiempo de vida de los equipos a la par

que asegura la máxima fiabilidad y seguridad operacional. Otra gran ventaja es que mediante el análisis de los datos recopilados se puede inferir la causa de la avería, lo que permitirá modificar condiciones de trabajo y/o de la instalación para evitar la repetición de dichas averías en el futuro.

La parte más importante para llevar esta estrategia de mantenimiento a buen puerto es elegir acertadamente las variables a monitorizar, ya que de ellas dependerá el posterior diagnóstico del motor. Estas variables requieren de una monitorización no invasiva con la maquina en funcionamiento, lo cual reduce considerablemente el abanico de posibilidades.

El método de monitorización condiciona en gran medida el alcance de la aplicación del mantenimiento, así como de la fiabilidad de los resultados. Todo esto obliga a que los sensores utilizados sean rápidos, fiables, precisos, autocorrectivos, no intrusivos, de fácil calibración, de diseño simple, compactos y preparados para operar bajo ambientes agresivos [Rodríguez Oraá, Bárbara Mª. 2013].

Desde un punto de vista técnico una actividad de mantenimiento será predictiva siempre que cumpla los siguientes requisitos:

- Medición no intrusiva: se ha de realizar con el equipo operando normalmente.
- El resultado de la medición se expresará en unidades físicas o mediante índices adimensionales correlacionados.
- La variable medida ofrezca una buena repetitividad.
- La variable predictiva ofrezca alguna capacidad, es decir, pueda ser analizada representando algún modo típico de fallo.

Desde un punto de vista organizativo, un sistema de gestión de mantenimiento será predictivo siempre que cumpla los siguientes requisitos:

- Medición periódica de variables.
- El sistema permita la coordinación entre los servicios de verificación predictivas y el de planificación del mantenimiento.
- Las organizaciones de mantenimiento y de producción estén preparadas para reaccionar ante un diagnóstico crítico, por motivos de operación y de seguridad.

Las técnicas predictivas más implantadas a día de hoy son el análisis de vibraciones, las inspecciones infrarrojas, análisis de aceites y detección de ultrasonidos. Otra técnica que está cobrando importancia es el análisis de corrientes de alimentación.

Cada una de estas técnicas detecta y diagnostica un conjunto determinado de fallos, no siendo excluyentes unas de otras, por lo que su uso combinado mejorará la fiabilidad del diagnóstico cuando detecten el mismo tipo de fallo.

Las grandes ventajas de esta estrategia de mantenimiento son la minimización de paradas, maximización de la vida útil de los equipos y de la fiabilidad, la flexibilidad en la planificación de mantenimiento, disminución de averías importantes derivadas de una detección de fallo tardía, y la posibilidad de efectuar un control y análisis a distancia.

Como inconvenientes tiene asociado el alto coste de implantación y la limitación de los fallos a detectar, que se reducen a los que las técnicas empleadas permitan. Este último es un

aspecto que día a día va minimizándose, ya que las técnicas asociadas siguen siendo objeto de investigación y de mejora.

4.3.5 Mantenimiento proactivo

También conocido como ingeniería del mantenimiento. Consiste en la investigación de las causas de las averías (RCA: Root Cause Analysis), buscando remedios para evitar que se repitan. De esta manera se aumenta la fiabilidad de los equipos.

Las prácticas más frecuentes para los motores de inducción son el equilibrado dinámico de rotores y la alineación de precisión de acoplamientos. Otras prácticas son los análisis estructurales de tipo ODS (Operating Deflection Shape) o el análisis modal experimental, ambos aplicados a la modificación de bancadas y elementos estructurales, así como al rediseño operativo del equipo.

Esta estrategia se ve potenciada gracias al uso de nuevas tecnologías y herramientas desarrolladas para el diagnóstico predictivo, ya que hace más sencillo y facilita las modificaciones constructivas y operativas de los equipos.

CAPÍTULO 5

MONITORIZACIÓN EN MOTORES DE INDUCCIÓN

5 MONITORIZACIÓN EN MOTORES DE INDUCCIÓN

5.1 Introducción

Como ya se indicó en el capítulo anterior, el mantenimiento predictivo requiere de la observación de ciertas variables del motor, lo que se conoce como monitorización. Dichas variables a observar han de proporcionar información del estado de la máquina, pudiendo asociarse a un determinado tipo de fallo las variaciones significativas de sus valores.

Para llegar al diagnóstico a partir de la monitorización se requiere seguir los pasos recogidos en la figura 5.1.

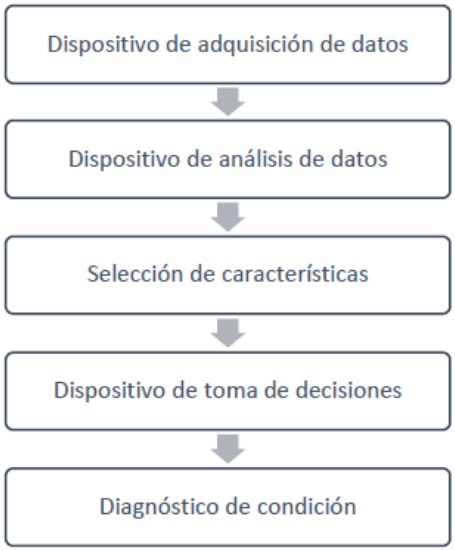


Figura 5.1: Esquema de funcionamiento de la teoría generalizada de la monitorización de estado. [Alonso Martínez, Alberto. 2015]

Inicialmente se requiere de dispositivos de adquisición de datos para poder recabar la información relativa a las variables de estudio, información que será tratada posteriormente mediante el dispositivo de análisis de datos. El siguiente paso es encontrar una característica cuantificada que se relacione con un determinado tipo de fallo, que será evaluada a continuación mediante el dispositivo de toma de decisiones, que no es sino la metodología usada para evaluar la característica seleccionada, que finalmente nos indicará el diagnóstico del estado del motor.

5.2 Variables y métodos utilizados en la adquisición de datos

Se clasifican según el efecto físico que medirán. Los efectos físicos más empleados se recogen a continuación [Alonso Martínez, Alberto. 2015]:

- **Flujo magnético**

  La distorsión en la densidad del flujo magnético del entrehierro debida a efectos del estator establece un flujo axial homopolar en el eje que puede medirse mediante una

bobina de búsqueda instalada en el propio eje, o mediante la medición de la tensión a través de dos bobinas ubicadas en posiciones concretas del motor. A partir de esas tensiones se puede obtener una señal independiente de la caída de tensión en el estator.

A partir de la utilización de al menos cuatro bobinas montadas asimétricamente en el accionamiento del eje se pueden encontrar además espiras cortocircuitadas.

- **Vibración**

Usualmente utilizado para detectar fallos mecánicos como el desalineamiento del rotor y fallos en cojinetes. También sirve para detectar cortocircuitos en el estator, barras rotas y desequilibrio en las tensiones de alimentación.

La medición se realiza mediante transductores situados en los cojinetes o en el estator, según interese.

- **Ruido**

El espectro de ruido en motores de inducción está dominado por el ruido EM, el ruido de ventilación y el ruido acústico. El ruido de ventilación se debe a la turbulencia del aire; el ruido EM es debido a las tensiones de Maxwell que actúan sobre superficies de hierro bajo acción de un campo magnético. Estas fuerzas inducen vibraciones en la estructura del estator, provocando ruido radiado.

- **Velocidad angular instantánea**

Nos informa de la dinámica del motor, siendo de interés porque las pulsaciones de par que son originadas por fallos en el motor alteran la velocidad del rotor. Permiten detectar fallos de asimetría, desequilibrios en la alimentación y fallos en el bobinado a través de la vibración del núcleo del estator.

- **Temperatura**

Hay gran variedad de técnicas basadas en esta variable, que permiten detectar entre otros el fallo por rotura de barras a partir de imágenes termográficas a través de la carcasa. También se pueden montar sensores en el bobinado o en el aislamiento, donde para determinar la temperatura se recurre al modelo térmico y/o al modelo de resistencia estatórica.

- **Par del entrehierro**

Es combinación del flujo inducido y de las corrientes, siendo sensible a cualquier desequilibrio, ya sean los producidos por defectos o por desequilibrio en la alimentación. El interés de esta variable es que la mayoría de fallos provocan alteraciones específicas en la misma. Como ejemplo, analizando la forma del par en el entrehierro se puede distinguir entre el fallo por barras rotas y el fallo por desequilibrio en el bobinado del estator.

El gran inconveniente es la imposibilidad de medición directa de esta variable, lo cual limita su uso, ya que las mediciones realizadas por sensores acoplados al soporte del

estator o al eje difieren del valor real del par, debido a que el conjunto del motor y carga mecánica constituye un sistema con su propia frecuencia natural.

- **Corriente**

En el análisis de la huella de la corriente (MCSA) se utilizan los resultados del análisis espectral de la corriente del estator para determinar los fallos.

La corriente de alimentación de un motor de inducción en perfecto estado tiene una sola componente. Cuando se produce un fallo se producen asimetrías en las corrientes y en el flujo magnético, que inducen corrientes armónicas en el estator, superponiéndose a las corrientes estatóricas originales. La técnica de MCSA puede detectar y analizar estas componentes armónicas, si bien fenómenos como la saturación, corrientes entre barras y asimetría magnética determinan la efectividad del diagnóstico.

- **Tensión inducida**

El método de análisis espectral de la tensión entre la línea de alimentación y el neutro del estator se utiliza para detectar el fallo de barra rota, ya que el mismo provoca asimetrías en la inductancia mutua de la máquina, aumentado los componentes armónicos de la tensión línea-neutro.

Esta técnica no sólo presenta las ventajas y sencillez de la técnica MCSA, sino que además presenta una mayor sensibilidad al fallo de barra rota. El inconveniente es que es una técnica aún incipiente, quedando mucho desarrollo por delante para su uso funcional.

- **Potencia**

La potencia instantánea es definida como el producto de la tensión entre dos terminales del estator y la corriente que circula por uno de los dos terminales. Se determina midiendo dichas tensión y corriente, efectuando su cálculo con dicha información. Hay tres tipos de potencias instantáneas: la pot. inst. parcial referida a la componente constante de la potencia, la pot. inst. parcial referida a la componente fundamental, y la pot. inst. total. El espectro de dichas potencias consta de varias componentes, apareciendo bandas laterales alrededor de las mismas con motivo del fallo por barra rota.

Hay otra técnica para detectar el mismo tipo de fallo referida al análisis espectral de la potencia reactiva.

- **Descarga parcial**

El test analizador de descargas parciales (PDA) es una de las primeras técnicas utilizadas en el mantenimiento predictivo de máquinas eléctricas, siendo fácilmente utilizado para el devanado del estator, ya que un devanado deteriorado sufre 30 veces o más el fenómeno de descargas parciales.

- **Análisis de gases**

Sirve para detectar la degradación del aislamiento, ya que al degradarse produce monóxido de carbono, que pasa al circuito de refrigeración de aire y se puede detectar

mediante una técnica de absorción infrarroja (IR).

La utilización de modulación por ancho de impulso (PWM) genera picos elevados de tensión que deterioran el aislamiento, debido a que debido a esta técnica los conductores empiezan a arrancar electrones del aire circundante, lo que origina la formación de ozono. Este ozono se combina con el nitrógeno del aire produciéndose diversas formas de óxido nitroso, que es altamente corrosivo.

- **Ensayo de sobretensión**

Método para el diagnóstico de fallos en el bobinado. Es un ensayo de comparación, donde dos pulsos de tensiones idénticas de gran amplitud y frecuencia se establecen simultáneamente en dos fases del bobinado del motor, estando la tercera fase conectada a tierra.

Se utiliza un osciloscopio para comparar el reflejo de los pulsos, que nos mostrarán fallos en el aislamiento de los bobinados si lo hubiera, que pueden ser localizados posteriormente mediante un dispositivo denominado "Surge Tester".

La detección de fallos en el aislamiento por este método es previa a que se produzca el fallo por cortocircuito, indicándosenos el debilitamiento del asilamiento.

- **Análisis del circuito del motor (MCA)**

Determina las variaciones dentro del motor, lo cual dará lugar a identificar los defectos, midiendo las propiedades electromagnéticas del motor como si fuese un circuito eléctrico. Se aplica una cantidad pequeña de energía con respuestas amplificadas, que son las que ayudan a evaluar la condición de los bobinados comparando las lecturas recibidas del rotor y del estator.

5.3 Tipos de señales

El procesamiento de las señales varía en función de su tipología. En este apartado se explica una clasificación de señales fenomológica (figura 5.2).

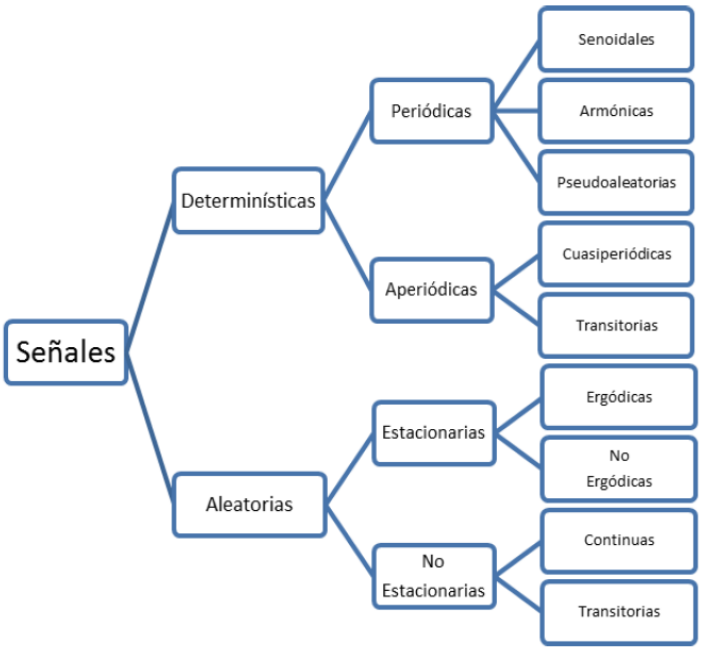


Figura 5.2: Clasificación de señales. [Alonso Martínez, Alberto. 2015]

5.3.1 Señales Deterministas

Son aquellas cuyos valores pueden ser predichos con exactitud, ya que su evolución se puede ajustar por un modelo matemático una vez conocidas las condiciones previas.

- **Señales periódicas**

  Una señal se denomina periódica si cumple con la condición (5.1) para toda *n*, donde *N* es el período de una señal discreta, siendo su valor mínimo el período fundamental. Su frecuencia es a su vez un número racional.

$$x(n) = x(n + N) \tag{5.1}$$

  - *Senoidales*

    Están definidas por la función seno (ecuación 5.2).

$$x(n) = sen(\omega n) \tag{5.2}$$

  - *Armónicas*

    Son aquellas señales periódicas no senoidales que pueden ser descompuestas en una serie de componentes de diferentes amplitudes y frecuencias, siendo estas últimas múltiplos enteros de la frecuencia original. A la primera frecuencia se le denomina primer armónico o armónico fundamental, y los siguientes reciben el nombre de segundo, tercero,…, armónicos.

  - *Pseudoaleatorias*

    Toda señal aleatoria que se repite periódicamente.

- **Señales aperiódicas**

Se engloba como tal a cualquier señal determinística que no sea periódica.

  - *Cuasiperiódicas*

    Señales no periódicas en su totalidad.

  - *Transitorias*

    Señales que tienen un principio y un final, correspondientes al nivel cero. Su duración puede ser tanto muy breve como extremadamente larga.

5.3.2 Señales aleatorias

En una señal aleatoria existe una incertidumbre sobre los valores que puede tomar la señal en los siguientes instantes, por lo que sólo pueden ser descritas de una manera estadística. Una realización de un proceso aleatorio difiere de los otros en su descripción temporal, aunque poseen las mismas propiedades estáticas. Estas señales se caracterizan, por tanto, por sus propiedades estadísticas y espectrales.

- **Señales estacionarias**

Señales de parámetros estadísticos constantes en el tiempo.

  - *Ergódicas*

    Señales donde es posible intercambiar medias temporales con medias estadísticas.

  - *No ergódicas*

    Señales donde no es posible intercambiar medias temporales con medias estadísticas.

- **Señales no estacionarias**

Señales de parámetros estadísticos no constantes en el tiempo.

  - *Continuas*

    Señales cuyo valor varía a lo largo de todo el período de observación.

  - *Transitorias*

    Tienen un principio y un fin, correspondientes al valor cero, dentro del período de observación.

5.4 Técnicas de análisis de señales

La obtención de información a simple vista de una señal no procesada es tarea casi imposible, y eso sin tener siquiera en cuenta el ruido inherente a la señal obtenida por medio de un instrumento de medición.

El procesamiento de señales es un conjunto de técnicas que analizan y transforman la señal original para obtener una representación significativa de parte de la información contenida

originalmente en la señal. Todo procesamiento busca igualmente reducir los efectos del ruido.

La elección del tipo de procesamiento depende de la naturaleza de la señal y de la información que se quiere obtener.

- **Técnicas de procesamiento en el dominio del tiempo**

  Las características relativas a los fallos pueden ser extraídas calculando indicadores estadísticos. Además, se puede recurrir a gráficos y coeficientes que muestren la variación de la señal a lo largo de tiempo.

- **Técnicas de procesamiento en el dominio de la frecuencia**

  Se utilizan para extraer la información contenida en la frecuencia de la señal, realizando una transformación para trasladar la señal al dominio del tiempo.

- **Técnicas de procesamiento en el dominio de tiempo-frecuencia**

  Su principal objetico es el de desarrollar una función que describa la densidad de energía de una señal tanto en el tiempo como en la frecuencia, lo que permite establecer la fracción de energía existente en un determinado intervalo de tiempo y de frecuencias.

Todo esto se tratará con mayor profundidad en el capítulo siguiente.

5.5 Técnicas de toma de decisiones

Su objetivo es valorar los parámetros, coeficientes o indicadores vinculados con los fallos hallados en el paso de selección de características, de tal manera que se pueda determinar la gravedad del fallo ante el que nos encontramos [Alonso Martínez, Alberto. 2015].

5.5.1 Técnicas estadísticas

- **Análisis discriminante**

  Método sencillo y eficiente desde el punto de vista computacional. Una vez seleccionadas las características y entrenado el método sólo queda calcular una ecuación como la ecuación 5.3para clasificar un nuevo ensayo.

$$y_{km} = u_0 + u_1 X_{1km} + \cdots + u_n X_{nkm} \qquad (5.3)$$

  Como la separación entre clases es lineal, la clasificación cuando hay presentes relaciones no lineales entre características es incorrecta en ciertos casos.

  - *Análisis discriminante descriptivo*

    Tiene por objeto determinar en qué medida un conjunto de características permite extraer dimensiones que diferencian a las clases, y cuáles de estas características son las que presentan mayor poder de discriminación. Dichas características reciben el nombre de variables discriminantes.

- *Análisis discriminante predictivo*

  Su objetivo es determinar una o más ecuaciones matemáticas (funciones discriminantes) que permitan la clasificación de nuevos casos a partir de la información que se tiene sobre ellos. Estas funciones combinan ciertas características de modo que permite la identificación del elemento a clasificar como perteneciente a la clase más parecida a sus características.

- **Regresión logística**

Es un método básico siempre que el análisis de datos tenga por objeto la descripción de la relación entre una variable respuesta y una o más variables explicativas. Solventa el problema que presenta la regresión lineal a la hora de trabajar con una variable de salida discreta que toma dos o más valores.

Otra gran ventaja son las prestaciones que ofrece la utilización de la función logística (ecuación 5.4), ya que puede equipararse a una probabilidad. Z (ecuación 5.5) es la suma lineal de las variables independientes X_i cuyos coeficientes a determinar son α y β_i.

$$f(x) = \frac{1}{1 + e^{-z}} \tag{5.4}$$

$$z = \alpha + \beta_1 X_1 + \cdots + \beta_k X_k \tag{5.5}$$

Además de todo esto, posibilita la consideración de relaciones no lineales, por lo que puede establecer límites de separación con curvatura entre las clases consideradas.

5.5.2 Redes neuronales

Las redes de neuronas artificiales (ANN) son un paradigma de procesamiento de información inspirado en el modo de operación del cerebro, siendo su estructura el elemento clave. Parten del modelo matemático de neurona propuesto por McCulloch y Pitts en 1943.

Están compuestas por cierto número de elementos de procesamiento (neuronas) que trabajan al unísono para resolver un problema específico. Las neuronas se encuentran conectadas entre sí, presentando esta conexión una intensidad variable, que viene determinada por los pesos sinápticos. Cada neurona recibe un conjunto de entradas x_j devolviendo una única salida, estando afectada cada entrada por un peso w_{kj} (figura 5.3).

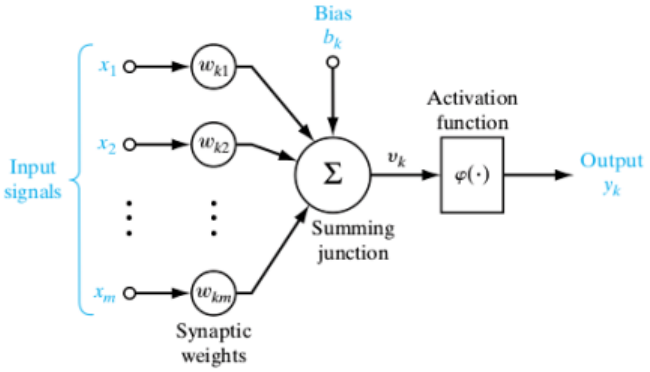


Figura 5.3: Modelo neuronal de McCulloch-Pitts. [Tenório, Eduardo. 2014]

La suma ponderada de las entradas, conocida como activación de la neurona v_k, es el primer paso para obtener la salida (ecuación 5.6). El parámetro w_{k0} es un umbral que se utiliza para compensar la diferencia entre el valor medio de las entradas y el valor medio de las salidas deseadas (figura 5.4).

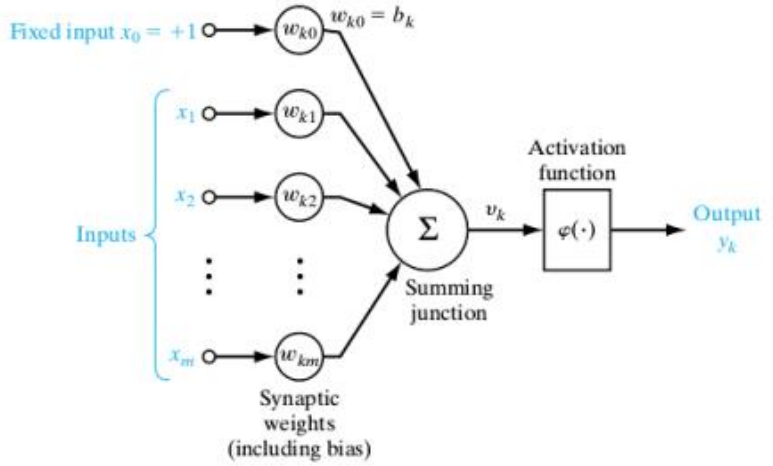


Figura 5.4: Modelo neuronal de McCulloch-Pitts. [Tenório, Eduardo. 2014]

$$v_k = u_k + b_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \qquad (5.6)$$

Finalmente, a partir de la activación de la entrada se obtiene la salida y_k de la neurona mediante la aplicación de la función de activación/transferencia φ (figura 5.3 y ecuación 5.7).

$$y_k = \varphi(v_k) = \varphi\left(\sum_{j=1}^{m} w_{kj} x_j + b_k\right) = \varphi\left(\sum_{j=0}^{m} w_{kj} x_j\right) \qquad (5.7)$$

CAPÍTULO 6

ANÁLISIS DE SEÑALES

6 ANÁLISIS DE SEÑALES

6.1 Introducción

Tal y como se comentó en su momento brevemente en el capítulo anterior, el análisis de señales a emplear depende de la naturaleza de las mismas así como de lo que esperemos obtener de dicho análisis.

No siempre es posible aplicar todas las técnicas a cualquier señal, y aunque fuese posible el resultado sería completamente infructuoso si los resultados no son de aplicación útil, habiéndose perdido tiempo y recursos en dicha tarea.

En este capítulo nos centraremos en los análisis de la señal de corriente de alimentación de los motores de inducción estudiados que nos proporcionan los indicadores, coeficientes y otros parámetros que serán utilizados posteriormente de predictores en el proceso de clasificación.

6.2 Análisis en el dominio del tiempo

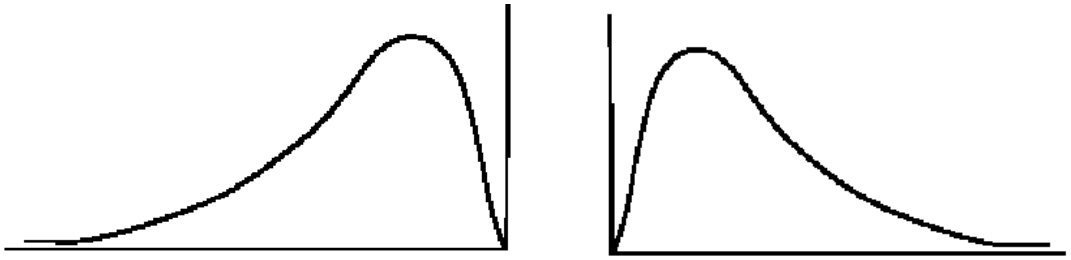
El análisis en el dominio del tiempo existe desde la aparición del osciloscopio. Permite establecer características básicas de las formas de onda.

Diversos autores han propuesto la utilización de características estadísticas obtenidas directamente de la señal en el dominio del tiempo, tanto de vibración como de corriente. Algunas de estas características tienen una interpretación física o geométrica clara mientras que otras poseen un poder discriminante entre tipos de fallos que las hacen útiles para el diagnóstico sin poseer una interpretación geométrica o física que facilite su visualización.

A continuación se explican los estadísticos utilizados, cuya interpretación física o geométrica resulta sencilla.

- **Coeficiente de asimetría o Skewness:**

  Una distribución es simétrica cuando a ambos lados de la media existen la misma cantidad de valores, equidistantes dos a dos de la media, y con la misma frecuencia. En caso contrario, decimos que la distribución es asimétrica, y entonces puede ser de dos tipos, como se muestra en la figura 6.1.



a). b)

Figura 6.1: a) Curva asimétrica a la izquierda. B) Curva asimétrica a la derecha. [Exploración de Datos. 2013]

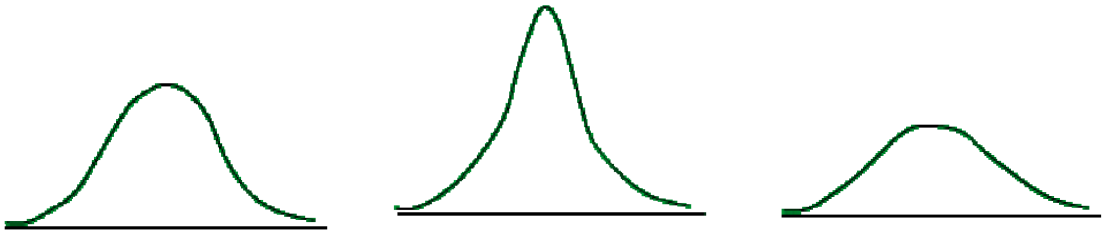
Dependiendo del valor del coeficiente de asimetría (skew) la distribución puede ser:

- Si skew> 0, la distribución es asimétrica positiva o a la derecha.
- Si skew = 0, la distribución es simétrica.
- Si skew< 0, la distribución es asimétrica negativa o a la izquierda

- **Coeficiente de Curtosis:**

Pretende comparar la curva de una distribución con la curva de la variable Normal, en función de la cantidad de valores extremos de la distribución.

Una distribución es Mesocúrtica si la distribución de sus datos es la misma que la de la variable Normal. En ese caso, su coeficiente de Curtosis es 3. La distribución es Leptocúrtica si está más apuntada que la Normal. En ese caso, su coeficiente de Curtosis vale más que 3. Si la distribución está menos apuntada que la Normal, entonces es Platicúrtica, y su coeficiente de Curtosis vale menos que 3.



a) Mesocúrtica. b) Leptocúrtica. c) Platicúrtica.

Figura 6.2: Clasificación de una distribución de datos en función de su coeficiente de Curtosis. [Exploración de Datos. 2013]

El factor de cresta para una onda sinodal es siempre $\sqrt{2}$. Un factor de cresta muy elevado implica sobre-intensidades puntuales importantes.

El factor de forma de una onda es la relación entre el valor eficaz y el valor medio, con lo que se obtiene una forma aproximada de la función. Tiene siempre un valor mayor o igual a 1. Si el factor de forma es pequeño, indica que la onda tiene poco nivel de alterna mientras que si es muy grande la onda tiene poca componente de continua y mucha de alterna.

- **Características estadísticas en el dominio del tiempo**

Varias características estadísticas tomadas directamente de la onda temporal, tanto de señales de vibración como de corriente, han sido propuestas como indicadores de fallos en motores de inducción.

De estas características, se han elegido 15 valores estadísticos para ser utilizados en el diagnóstico de barras rotas en motores de inducción. Estas características, con sus respectivas fórmulas, se pueden ver en la Tabla 6.1.

Algunas de las características estadísticas consideradas tienen una interpretación geométrica sencilla, mientras que otras son útiles para el diagnóstico pero no tienen una interpretación sencilla.

Tabla 6.1: Características estadísticas seleccionadas.

	Característica estadística	Ecuación	Nomenclatura
*	Momento de 1er Orden.	$m_1 = \frac{1}{n}\sum(x)$	1er M
*	Momento de 2º Orden.	$m_3 = \frac{1}{n}\sum(x-\bar{x})^2$	2º M
1	Momento de 3er Orden.	$m_3 = \frac{1}{n}\sum(x-\bar{x})^3$	3er M
2	Momento de 4to Orden.	$m_4 = \frac{1}{n}\sum(x-\bar{x})^4$	4to M
3	Cumulante de 1er Orden.	$C_1 = m_1$	1er C
4	Cumulante de 2do Orden.	$C_2 = m_2 - {m_1}^2$	2do C
5	Cumulante de 3er Orden.	$C_3 = m_3 - 3m_1m_2 + 2{m_1}^3$	3er C
6	Cumulante de 4to Orden.	$C_4 = m_4 + m_3m_1 - 3{m_2}^2 + 12m_2{m_1}^2 - 6{m_1}^4$	4to C
7	Cumulante de 6to Orden.	$c_6 = \frac{1}{n}\sum\frac{(x-\bar{x})^6}{{m_3}^2}$	6º C
8	Skewness o asimetría.	$Skew = \frac{m_3}{(\sqrt{m_2})^3}$	Skew
9	Curtosis.	$Curt = \frac{m_4}{(\sqrt{m_2})^4}$	Curt
10	Promedio de valores absolutos.	$\lvert\bar{X}\rvert = \frac{1}{n}\sum\lvert x\rvert$	PVA
11	Máximo valor absoluto.	$X_p = \max\lvert x\rvert$	MVA
12	Valor cuadrático medio.	$X_r = \left(\frac{1}{n}\sum\sqrt{\lvert x\rvert}\right)^2$	VCM
13	Valor eficaz o RMS (*Root Mean Square*)	$x_{rms} = \sqrt{\frac{1}{n}\sum(x-\bar{x})^2}$	RMS
14	Factor de Cresta.	$F_c = \frac{x_p}{x_{rms}}$	FC
15	Factor de forma.	$F_f = \frac{x_{rms}}{\lvert\bar{x}\rvert}$	FF

* No se utilizan como predictores para la clasificación.

6.3 Análisis en el dominio de la frecuencia

El análisis en el dominio de la frecuencia tiene una importancia muy relevante a la hora de analizar señales, si bien tiene ciertas limitaciones derivadas de su propia definición que hacen necesario un análisis conjunto en el dominio del tiempo. Esto es así porque en el momento de transferir la señal del dominio del tiempo al dominio de la frecuencia se pierde parte de la información contenida en la señal previamente.

Es una herramienta clásica en la teoría de control debido a su sencillez de comprensión respecto al dominio temporal, aunque en la actualidad se han venido desarrollado herramientas computacionales que posibilitan una fácil simulación en el dominio del tiempo.

6.3.1 Técnicas de procesamiento

6.3.1.1 Transformada de Fourier (TF)

Todas las transformaciones de señales entre el dominio del tiempo y el de la frecuencia se basan en la Transformada de Fourier (ecuación 6.1) y en la Transformada inversa de Fourier (ecuación 6.2). Se denota con letra mayúscula la representación de la función en el dominio de la frecuencia, y con minúscula en el del tiempo.

$$X(f) = \int_{-\infty}^{+\infty} x(t) e^{-j2\pi f t} dt \tag{6.1}$$

$$x(f) = \int_{-\infty}^{+\infty} X(t) e^{-j2\pi f t} df \tag{6.2}$$

Hay que tener en consideración la estacionalidad de la señal *x(t)*. Será estacionaria si sus componentes frecuenciales existen para todo *t* en que esté definida *x(t)*.

6.3.1.2 Densidad espectral de potencia (PSD)

La densidad espectral de potencia (ecuación 6.3) contiene información relacionada con la distribución en potencia de la señal *x(t)* en el dominio de la frecuencia, y se mide en V^2/Hz. Su principal aplicación es el análisis espectral de señales aleatorias o señales periódicas contaminadas con ruido.

$$|X(f)|^2 = X(f) \cdot X^*(f) \tag{6.3}$$

6.3.1.3 Transformada discreta de Fourier (DFT)

La Transformada de Fourier implica operar con una formulación analítica de la señal considerando un período de muestreo infinito, lo cual nunca se cumple en la práctica ya que se trabaja con señales muestreadas no disponiéndose en muchos casos de la formulación analítica de la misma. Si consideramos la señal muestreada se define la Transformada discreta de Fourier, cuya expresión se recoge en la ecuación 6.4.

$$X(m\Delta f) = \int_{-\infty}^{+\infty} x(t) e^{-j2\pi \Delta f t} dt \tag{6.4}$$

Donde:

- m=0, ±1, ±2,….
- Δf: espacio entre líneas de frecuencia.

Además, los métodos de integración numérica convierten el cálculo de la integral en una serie de sumas con la forma que se recoge en la ecuación 6.5, conocida como la Transformada discreta de Fourier en suma de series.

$$X_x(m\Delta f) \approx \Delta t \sum_{-\infty}^{+\infty} x(n\Delta t) e^{-j2\pi n \Delta f \Delta t} \tag{6.8}$$

Donde:

- Δt: intervalo de tiempo entra muestras consecutivas.

Como hemos de considerar el sumatorio con un valor finito, la expresión anterior queda como la ecuación 6.6.

$$X_x(m\Delta f) \approx \Delta t \sum_{n=0}^{N-1} x(n\Delta t) e^{-j2\pi n \Delta f \Delta t} \qquad (6.8)$$

6.3.1.4 *Transformada rápida de Fourier (FFT)*

El cálculo digital de la Transformada de Fourier es computacionalmente costoso debido al gran número de operaciones matemáticas a realizar. Para solventar este problema se desarrolló por Cooley y Tukey en 1965 la Transformada rápida de Fourier, que escoge un número de muestras *N* que sea potencia de dos para aprovechar ciertas propiedades de simetría y conseguir calcular el resultado de la Transformada de Fourier con mayor rapidez y eficiencia.

La menor frecuencia que se va a conseguir con un analizador FFT va a depender de la longitud del registro en el tiempo, ya que si el período de la señal es mayor que el registro en el tiempo no se podrá determinar completamente dicho período. Por tanto, la frecuencia inferior del espectro debe ocurrir a la inversa de la longitud del período de muestreo *PM* (ecuación 6.9).

$$f_{min} = \frac{1}{PM} \qquad (6.9)$$

La resolución del espectro (en Hz) será 1/PM, que es la menor distancia a la que dos componentes del espectro pueden ser separados. La mayor frecuencia del espectro se obtiene al multiplicar la frecuencia mínima por el número de líneas de frecuencia (ecuación 6.10).

$$f_{max} = \frac{N}{2}\frac{1}{PM} \qquad (6.10)$$

Donde:

- N: número de muestras por segundo. Ha de ser múltiplo de 2 por las características inherentes de esta técnica.- Δt: intervalo de tiempo entra muestras consecutivas.

6.3.2 Diagnóstico mediante análisis espectral de la corriente de alimentación

Para el caso que nos ocupa el análisis espectral se aplicará a la corriente de alimentación del motor, lo que se conoce como técnica MCSA. Si bien nos proponemos únicamente la clasificación de ensayos caracterizados por fallo debido a rotura de barras, a continuación se explicarán todos los tipos de fallos que se pueden detectar mediante esta técnica, así como las limitaciones de la misma.

La existencia de asimetrías en el rotor (por ejemplo, una barra rota) provoca la fluctuación de los valores de la corriente absorbida. Está demostrado que aplicando una secuencia positiva de tensiones a los devanados del estator se inducen corrientes en el rotor a la frecuencia de deslizamiento, lo cual deriva en campos giratorios de sentido directo e inverso en el

entrehierro. Cuando el devanado del rotor es perfectamente simétrico los campos directos se suman y los inversos se anulan, pero cuando el rotor posee alguna asimetría los campos inversos no se cancelan entre sí, apareciendo una componente giratoria inversa en el campo que induce corrientes a una determinada frecuencia en el estator.

Se utiliza la Transformada rápida de Fourier (FFT) para obtener el espectro de las señales medidas.

6.3.2.1 Armónicos de corriente originados por excentricidad

Se puede detectar la excentricidad del entrehierro identificando en la corriente patrones anormales y analizando la tendencia en el tiempo. Los armónicos producidos por este tipo de asimetría tienen unas frecuencias que siguen la expresión recogida en la ecuación 6.11:

$$f_{ec} = f_s \cdot \left[(R \pm n_d) \left(\frac{1-s}{p} \right) \pm n_{ws} \right] \qquad (6.11)$$

Donde:

- f_s: frecuencia principal.

- p: pares de polos.

- n_s: ±1.

- n_{ws}: 1, 2, 5, 7...

- R: número de ranuras del rotor.

- s: deslizamiento.

6.3.2.2 Armónicos de corriente originados por fallo de rotura de barras

Son indicadores que utilizaremos en nuestro problema de clasificación. Derivan de la aplicación de la transformada rápida de Fourier (FFT), ya comentada con anterioridad, a la corriente de alimentación y sirven para determinar el caso de rotura de barras en un motor de inducción de jaula de ardilla.

La rotura de barras inducirá armónicos en la corriente del rotor, cuyas frecuencias se determinan según las expresiones 6.12 y 6.13 [Pons Llinares, Joan; Climente Alarcón, Vicente; Puche Panadero, Rubén; Antonino Daviu, Jose A.]:

$$f_{bb} = f_s \cdot |1 \pm 2ks| \qquad (6.12)$$

$$f_{bb} = f_s \cdot \left[\frac{k}{p} (1-s) \pm s \right] \qquad (6.13)$$

Donde:

- f_s: frecuencia principal.

- p: número de pares de polos.

- En (7.1) *k* es cualquier entero positivo mientras que en (7.2) *k/p*=1,3,5,….

De entre todos los armónicos los más importantes son aquellos que se obtienen para k=1: el

armónico de banda lateral inferior (**LSH**) y el *armónico de banda lateral superior* (**USH**).

Tabla 8.1: Indicadores del dominio de la frecuencia elegidos.

	Indicador FFT	Frecuencia del armónico	Nomenclatura
1	Lower Sideband Harmonic	$f_{LSH} = f_s \cdot \lvert 1 - 2s \rvert$	LSH
2	Upper Sideband Harmonic	$f_{USH} = f_s \cdot (1 + 2s)$	USH

En la figura 6.3 se muestra el espectro esperable para el fallo de barras rotas, con la componente fundamental en el centro y dos armónicos destacables a ambos laterales de la misma.

Figura 6.3: Espectro de barras rotas. [Rodríguez Oraá, Bárbara Mª. 2013]

6.3.2.3 Armónicos de corriente originados por cortocircuito en espiras del estator

Las componentes armónicas originadas por el cortocircuito de espiras tienen frecuencias que siguen la expresión mostrada en la ecuación 6.14.

$$f_{st} = f_s \cdot \left[\frac{n}{p}(1 - s) \pm k\right] \qquad (6.11)$$

Donde:

- f_s: frecuencia principal.

- p: pares de polos.

- n: 1, 2, 3,...

- k: 1, 3, 5, 7...

- s: deslizamiento.

6.3.2.4 Armónicos de corriente originados por influencias mecánicas

La variación de la excentricidad origina variaciones en la longitud del entrehierro, modificándose la forma del flujo magnético a su paso. Esto puede inducir componentes armónicas en la corriente del estator dadas por:

- **La influencia de reductores**

  En el caso de presencia de reductores se originan armónicos de frecuencia similar a los producidos por fallo de barra rota, lo que hay que tener en cuenta si se trata de mantener un

- **Problemas en rodamientos**

  Los fabricantes de rodamientos proporcionan los valores de las frecuencias de fallo, por lo que analizando el espectro de la corriente se pueden detectar los mismos. El grado de deterioro de los rodamientos será función de la amplitud y frecuencia de los armónicos originados.

6.3.2.5 Limitaciones de la técnica

Hay una serie de factores que influirán sobre la presencia de los armónicos en el espectro de la corriente y sobre su amplitud, limitando los resultados de la técnica.

- **Nivel de carga**

  Niveles de carga muy reducidos implican deslizamientos cercanos al valor 0, lo que provoca que las bandas se encuentren muy cercanas a la componente fundamental, lo que requerirá de instrumentos de captación y análisis de suficiente resolución.

- **Oscilaciones e inercia de motor y carga**

  En cualquier motor que trabaje con un par de carga variable respecto a la posición se inducirán corrientes de diversas frecuencias en el estator, superponiéndose entre sí.

- **Corrientes interlaminares**

  Son corrientes que aparecen entre barras, circulando a través del núcleo magnético. Reducen la amplitud de los armónicos generados por asimetrías en el rotor.

- **Sistema de alimentación**

  Las amplitudes de los armónicos característicos de falta son función de la gravedad del fallo, de los parámetros constructivos del rotor y de las características del sistema de alimentación.

- **Efectos del ranurado**

  La distribución de los devanados en ranuras causa armónicos de fuerza magnetomotriz que a su vez originan armónicos de corriente, que pueden estar próximos a los armónicos característicos de falta.

- **Efectos de la saturación**

  La saturación del campo magnético origina la aparición de armónicos en el espectro de la corriente de alimentación, que pueden superponerse con los armónicos característicos de falta.

CAPÍTULO 7

BANCO DE ENSAYOS

7 BANCO DE ENSAYOS

7.1 Introducción

En este capítulo se procederá a describir brevemente los equipos y herramientas utilizados para obtener los ensayos de motores, de donde se obtuvieron posteriormente los indicadores que se utilizarán para el proceso de clasificación de los mismos. El objetivo es mostrar el funcionamiento del banco de ensayo, así como la metodología de ensayo y análisis seguida.

7.2 El banco de ensayos

Es el conjunto de equipos utilizados para realizar los ensayos, es decir, donde obtener, tratar y almacenar los datos recopilados.

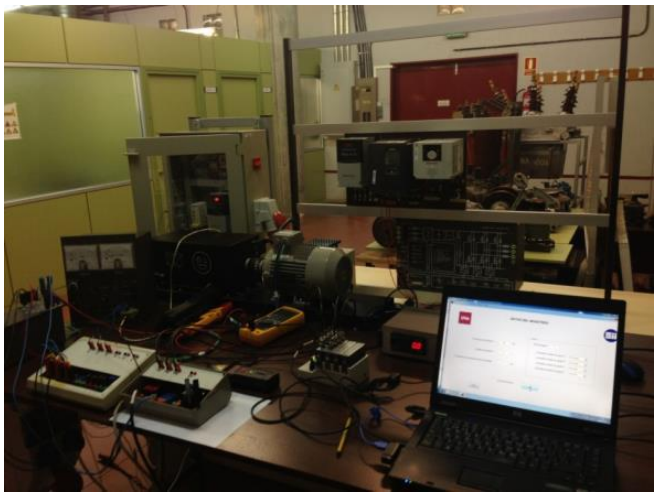


Figura 7.1: Banco de ensayos. [Zamora Pérez, Carlos. 2013]

El banco de ensayos se encuentra situado en el Departamento de Ingeniería Eléctrica de la EII de la Universidad de Valladolid.

7.2.1 Alimentación

El banco de ensayos puede ser alimentado de dos formas:

- Alimentación del motor directamente de la red a 50 Hz (figura 7.2).

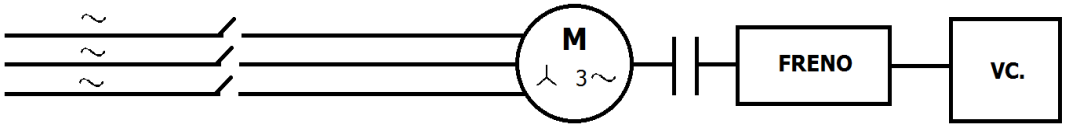


Figura 7.2: Banco de ensayos con alimentación de motor directa de la Red. [Zamora Pérez, Carlos. 2013]

- Alimentación del motor a través de un variador de frecuencia o convertidor (figura 7.3).

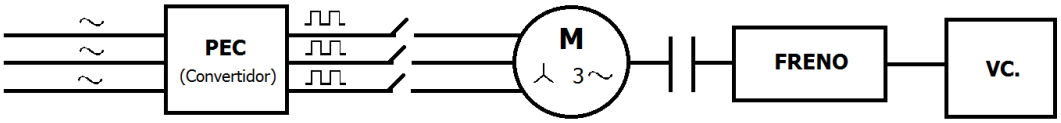


Figura 7.3: Banco de ensayos con alimentación de motor a través de un convertidor. [Zamora Pérez, Carlos. 2013]

Para ello se utiliza el siguiente convertidor que se encuentra en el laboratorio: PowerFlex 40 de Allen Bradley. Se caracteriza por disponer de una función específica para la elección de la frecuencia de conmutación en un rango entre 3 y 8kHz con una precisión de 0,1kHz.

Figura 7.4: Convertidor AB PowerFlex 40. [Zamora Pérez, Carlos. 2013]

7.2.2 Motor

En el laboratorio se dispone de distintos motores de inducción, sobre los que se van provocando los distintos tipos de fallos que se desea estudiar, con la severidad de fallo deseada.

En nuestro caso concreto se han realizado ensayos relacionados con la rotura de barra, donde se ha ido procediendo a taladrar de forma artificial las barras de aluminio de la jaula correspondientes al motor M8 (figura 7.5).

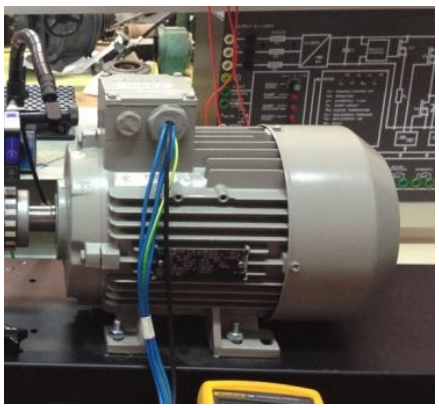


Figura 7.5: Motor de inducción M8. [Zamora Pérez, Carlos. 2013]

Los distintos estados del motor con los que se han realizado los ensayos se recogen en la tabla 7.3, mostrándose en la figura 7.6.

Tabla 7.1: Estados del motor supuestos en función de la gravedad del daño en las barras del rotor.

Estado del motor	Estado cualitativo de las barras
Motor sano (R1)	Barras en aparente perfecto estado.
R2	Perforación de 4,2 mm de profundidad, 2,5 mm de diámetro
R3	Perforación de 9,4 mm de profundidad, 2,5 mm de diámetro
R4	Perforación de 18 mm de profundidad, 2,5 mm de diámetro
R5	Perforación de 18 mm de profundidad, 3,5 mm de diámetro

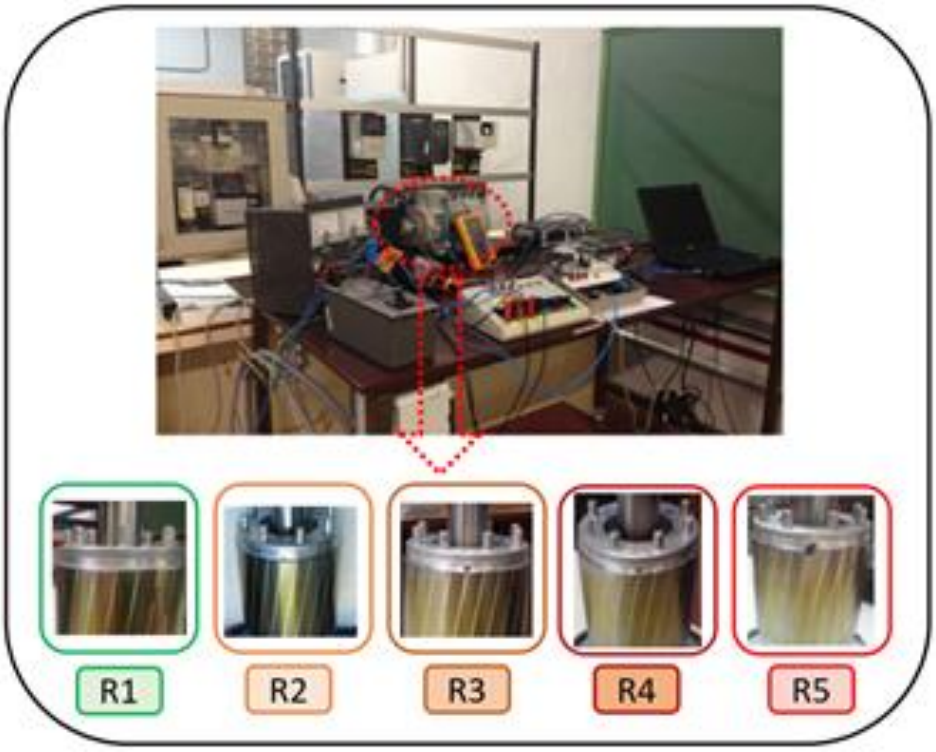


Figura 7.6: Estados del motor supuestos en función de la gravedad del daño en las barras del rotor.

Las características del motor nº8 empleado para los ensayos son las siguientes:

Motor Asíncrono Siemens modelo: 1LA7 090 4AA 10

- Motor estándar, Eficiencia aumentada
- Grado de protección IP55
- Versión aluminio
- 4 polos
- Potencia: 1,1 kW. 1,5CV
- Factor de Potencia: 0,81.
- Tensión de Alimentación: 230/400V.

- Velocidad Nominal (50Hz): 1500 r.p.m.

7.2.3 Freno

Se ha empleado un freno de polvo magnético así como su controlador asociado (figura 7.7), con el objeto de generar un par de carga modificable de cara a realizar distintos ensayos.

En nuestro caso se ha trabajado con dos niveles de carga:

- Nivel de carga 1: correspondiente a baja carga (deslizamiento ≈3%).
- Nivel de carga 2: correspondiente a plena carga (deslizamiento ≈5%).

Figura 7.7: Izquierda: controlador del freno. Derecha: freno de polvo magnético. [Zamora Pérez, Carlos. 2013]

7.2.4 Sensores

Se han utilizado dos tipos de sensores diferentes:

- Sensor de velocidad o tacómetro

  En el laboratorio se dispone de dos tipos de tacómetros (figura 7.8): un tacómetro láser o un tacómetro digital, requiriendo este último de un sensor óptico para la adquisición de datos (figura 7.9). En nuestro caso se ha utilizado el tacómetro láser de la marca Compact y modelo CT6/LSR, cuyas características se recogen en la tabla 7.2.

Figura 7.8: Izquierda: tacómetro láser. Derecha: tacómetro digital. [Zamora Pérez, Carlos. 2013]

Figura 7.9: Sensor óptico del tacómetro digital. [Zamora Pérez, Carlos. 2013]

Tabla 7.2: Características técnicas del tacómetro láser utilizado.

Módulo láser		Escalas de velocidad	
Escala óptica	25-2000mm	Precisión	±1 dígito
Ángulo óptico	±80°	Resolución	0.001
Longitud de onda	670nm		

- Sensor de corriente

Se utiliza para medir la intensidad eléctrica, estando basadas en efecto hall.

7.2.5 Transductores

Los transductores son aparatos que transforman un determinado tipo de energía de entrada en otro tipo a la salida. La señal de entrada es la procedente de los sensores correspondientes. Para la realización de los ensayos se cuenta con los siguientes tipos de transductores:

- Transductores de intensidad

Convierten la corriente alterna (I/t) en tensión alterna proporcional a dicha intensidad.

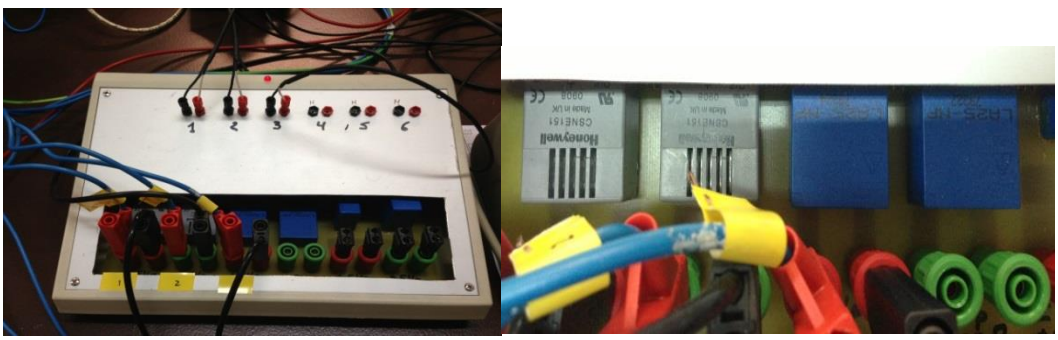


Figura 7.10: Izquierda: transductor de intensidad. Derecha: detalle del transductor de intensidad. [Zamora Pérez, Carlos. 2013]

- Transductores de tensión

Convierten una tensión de 380V en otra mucho menor del orden de ±5V o ±10V.

Figura 7.11: Izquierda: transductor de tensión. Derecha: detalle del transductor de tensión. [Zamora Pérez, Carlos. 2013]

7.2.6 Tarjeta de adquisición de datos

Transforma una entrada analógica (la señal procedente de los transductores) en digital, enviándola al puerto USB 2.0 del ordenador para su correspondiente almacenamiento. En este ordenador se procederá posteriormente a obtener el espectro de corriente gracias al software de adquisición de datos.

Para nuestro caso se empleará la tarjeta de la marca National Instruments, modelo PCI-6250 (figura 7.12). Sus especificaciones técnicas se recogen en la tabla 7.3.

Tabla 7.3: Características técnicas de la tarjeta de adquisición de datos.

Entrada analógica		E/S digitales	
Número de Canales	16 SE/8 DI	Número de Canales	24 DIO
Velocidad de Muestreo	1.25 MS/s	Niveles Lógicos	TTL
Resolución	16 bits	Máximo Rango de Entrada	0..5 V
Rango de Voltaje (Max)	-10..10 V	Máximo Rango de Salida	0..5 V
Precisión del Rango	1920 µV	Entrada de Flujo de Corriente	Sinking, Sourcing
Sensibilidad del Rango	112 µV	Salida de Flujo de Corriente	Sinking, Sourcing
Rango de Voltaje (Mín)	-100..100 mV	Capacidad de Corriente	24 mA/448 mA

Figura 7.12: Tarjeta de adquisición de datos. [Zamora Pérez, Carlos. 2013]

7.3 Metodología para la realización de ensayos

7.3.1 Etapas a seguir en cada ensayo

A continuación se realiza una detallada descripción de las etapas llevadas a cabo para la realización de los ensayos.

- Conexión del ordenador a través de la toma de seguridad e inicio del programa empleado para la adquisición de datos: MATLAB.
- Conexión de todos los instrumentos de medida y equipos comprobando que la misma sea adecuada. Este proceso se realiza sin la conexión a la red.
- Arranque del motor con el freno al mínimo y ajuste de la frecuencia del variador. Anotación del valor de tensión y temperaturas del motor y del freno a través de los instrumentos de medida (amperímetro, voltímetro, etc.).
- Con el motor en vacío, es decir, con el freno en la posición de mínimo en la que no ejerce ningún par resistente. Se realizan una serie de ensayos anotando para cada ensayo los valores obtenidos mediante los instrumentos de medida: revoluciones por minuto, corriente consumida por el motor, temperatura del freno y temperatura del motor.
- Dejar enfriar el motor y a continuación seguir con el resto de ensayos.
- Concluidos los ensayos de vacío y enfriados el motor y el freno, se realizan los ensayos a plena carga.
- Con el motor a plena carga, es decir elevando el par de carga impuesto por el freno hasta que la corriente que circula por una de las fases del motor alcance la intensidad de referencia. Se realizan una serie de ensayos anotando para cada ensayo los valores obtenidos mediante los instrumentos de medida: revoluciones por minuto, corriente consumida por el motor, temperatura del freno y temperatura del motor.
- Terminados los ensayos, reducir al mínimo el freno y apagar el motor, registrando las temperaturas correspondientes. Hasta que las temperaturas no son lo suficientemente bajas, no se debe continuar con la realización de ensayos.
- Adquisición y tratamiento de datos mediante el programa Matlab para su posterior estudio.
- Recopilación de los datos obtenidos en una hoja de Excel.

7.3.2 Sistema y metodología de adquisición de datos

Este subapartado trata de describir el sistema y metodología empleados para la realización del proceso de adquisición de datos.

Para realizar el ensayo se miden las tres intensidades (intensidades de línea) y también se miden las tres tensiones (tensiones de línea). Para medir las intensidades y las tensiones emplearemos una serie de sensores de efecto Hall o transductores (uno por fase) como se ilustra en el esquema de la figura 7.13.

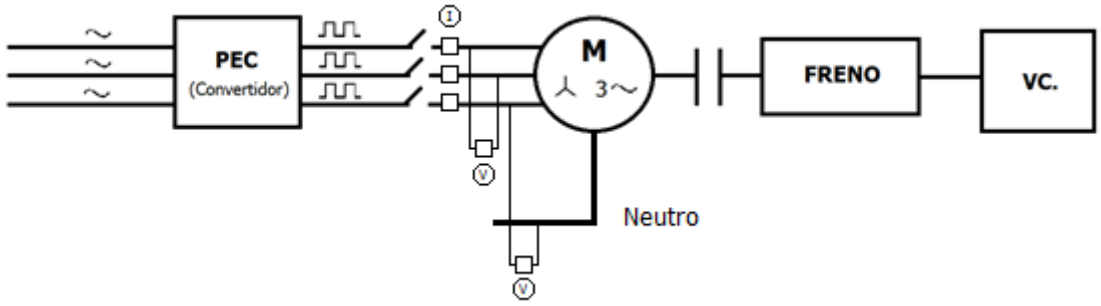


Figura 7.13: Esquema de la disposición de las conexiones de los transductores. [Zamora Pérez, Carlos. 2013]

Los transductores de intensidad convierten la corriente alterna (I/t) en tensión alterna proporcional a la intensidad. Los transductores tienen una masa en común y se toman medidas respecto a esa masa. Se conectan en serie en cada fase (dos iguales y uno diferente pero equivalente). Tienen un patillaje por donde hacemos circular una I (del orden amperios [A]) en la entrada de manera que por la por la salida donde tenemos una resistencia obtendremos una intensidad más pequeña (aprox. 25mA) (Rango de la tarjeta de adquisición de datos); lo que se traduce en una caída de tensión en una resistencia externa.

Los transductores de tensión transforman una tensión de 380V en otra mucho más pequeña de aproximadamente ±5V o ±10V (rango de la tarjeta de adquisición de datos).

Una vez conseguido mediante los transductores transformar la señal de entrada en una señal de salida adecuada al rango de nuestra tarjeta de adquisición de datos, se registra en una tarjeta de National Instruments que es de tipo modular y la cual admite entradas BNC.

La tarjeta recibe las señales (continuas en el tiempo) y las muestrea a la frecuencia que le indiquemos y durante el tiempo requerido; es decir mide las señales de entrada que recibe y las transmite al ordenador mediante un cable USB.

La tarjeta también recibe una señal de velocidad a través de un sensor óptico de contraste B/N cuya salida es de tipo BNC. Del sensor se obtiene otra salida que va a un tacómetro que nos proporciona la velocidad en rpm. En caso de fallo de este tacómetro, también se dispone de otro portátil que realiza la medida mediante laser.

Una vez adquiridos los datos, se ejecuta un programa desarrollado en el entorno Matlab. Dicho programa realiza una serie de llamadas a subrutinas de LabVIEW para el registro de datos, y a continuación aplica una Transformada DFT para obtener el espectro de frecuencia de la señal de corriente del estator del motor.

Para el manejo de las señales recibidas en el ordenador se ha empleado los siguientes valores:

- La frecuencia de muestreo en nuestros ensayos será de 80kHz (Cuanto mayor sea la frecuencia de muestreo mayor será el número de puntos por ciclo).
- El tiempo de muestreo T de nuestros ensayos será de 10s. Esto permite tener en el dominio de la frecuencia una resolución de 0,1 Hz ($f_{resolución}$= 1/T=1/10=0,1Hz).
- Desde el ordenador se indica a la tarjeta la frecuencia y el tiempo de muestreo. La tarjeta procesa los cálculos y MATLAB los guarda en un archivo *.mat* y se calcula el espectro con una Transformada DFT. También se guardan las ondas temporales registradas.

CAPÍTULO 8

CLASIFICACIÓN

8 CLASIFICACIÓN

8.1 Introducción

La clasificación dentro de las ciencias de la computación hace frente a dos tipos de problemáticas de enfoques diferentes: conocer de antemano el número de clases o grupos en los que desea clasificar una serie de datos, de modo que hay que establecer reglas mediante las cuales clasificar futuras observaciones; o bien establecer la existencia de cierto número de clases dentro de un grupo de datos facilitados.

En nuestro caso concreto utilizaremos una clasificación con el enfoque de clasificar futuras observaciones dentro de un número de clases conocidas.

Hay tres grandes corrientes de investigación al respecto:

8.1.1 Enfoque estadístico

La primera fase, conocida como clásica, deriva de los trabajos de Fisher en discriminación lineal. La segunda fase o fase moderna utiliza modelos más flexibles, muchos de los cuales tratan de proporcionar una estimación de la distribución conjunta de las características de cada clase, que a su vez puede dar lugar a una regla de clasificación.

Los métodos estadísticos están caracterizados generalmente por tener un modelo de probabilidad explícito subyacente, que proporciona la probabilidad de pertenecer a cada clase en vez de una simple clasificación. Además, se supone que las técnicas serán utilizadas por estadísticos, por lo que se supone una cierta intervención humana con respecto a la selección y transformación de variables, así como a la estructuración general del problema.

8.1.2 Aprendizaje automático

El aprendizaje automático o *machine learning* se utiliza generalmente para crear procedimientos de computación automáticos basados en operaciones lógicas o binarias, que aprenden a realizar una tarea a partir de una serie de ejemplos o muestras. Al proceso en el que el algoritmo crea una serie de reglas a partir de los ejemplos proporcionados se le denomina *entrenamiento* del clasificador.

Su objetivo es el de generar expresiones de clasificación lo suficientemente simples como para ser comprendidas fácilmente por el ser humano. Deben imitar lo suficiente el razonamiento humano como para proporcionar una visión global del proceso de clasificación.

La mayor limitación de este tipo de clasificación reside en su propio planteamiento: a mayor complejidad del problema de clasificación mayor cantidad de datos hay que proporcionar previamente para su entrenamiento, si bien han ido surgiendo distintos métodos para resolver las distintas problemáticas que van surgiendo: pocos datos de muestra, cantidad de atributos variables, etc.

Como el enfoque estadístico, estas técnicas requieren de un conocimiento de las mismas para llevar a cabo su desarrollo, si bien la operación se asume que es sin intervención humana.

Se abordará este tipo de clasificación con más detalle en los apartados siguientes, ya que la técnica de clasificación elegida pertenece a esta corriente.

8.1.3 Redes neuronales

Ya se comentó el concepto de redes neuronales en capítulos anteriores, por lo que no se va a repetir su funcionamiento. Su empleo en el campo de la clasificación es interesante ya que al igual que en el caso del aprendizaje automático trata de llevar a cabo esta tarea sin intervención humana durante el proceso.

8.2 Aprendizaje automático

8.2.1 Introducción

Como ya se indicó anteriormente, el objetivo es el de conseguir un aprendizaje automático para ser capaces de hacer predicciones precisas basándonos en observaciones pasadas.

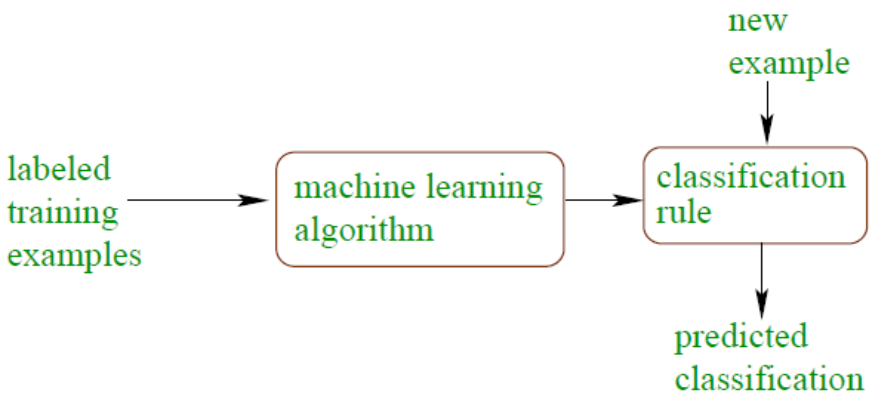


Figura 8.1: Proceso de clasificación desde el entrenamiento hasta la propia acción de clasificación. [Schapire, Rob]

Las características del aprendizaje automático moderno son las siguientes:

- Alta exactitud en las predicciones.
- Métodos de uso general y totalmente automáticos (si bien es necesaria la intervención y conocimientos humanos durante su desarrollo).
- Métodos preparados para trabajar con gran cantidad de datos.
- Interacción entre la teoría y la práctica.

Las principales ventajas de estos métodos son las siguientes:

- Resultados más precisos que la mayoría de los conseguidos por reglas elaboradas por humanos.
- Dificultad de expresar parte del conocimiento humano en forma de reglas (Ej: reconocimiento de letras).
- Dificultad para expresar.
- No es necesario un experto.
- Métodos baratos y flexibles.

Mientras que sus principales desventajas son:

- Necesitan una gran cantidad de datos.
- Imposibilidad de conseguir precisión perfecta.

Hay que destacar que se puede considerar 3 tipos de error en la clasificación:

- Error de entrenamiento o de resustitución: es la fracción de ejemplos de entrenamiento mal clasificados.
- Error de test: es la fracción de ejemplos de prueba mal clasificados.
- Error generalizado: es la probabilidad de clasificar incorrectamente una nueva muestra aleatoria.

8.2.2 Tipos

Según del tipo de salida que se produzca y de cómo se aborde el tratamiento de los ejemplos, los diferentes algoritmos de aprendizaje automático se pueden agrupar en [Sancho Caparrini, Fernando]:

- **Aprendizaje supervisado**

Se genera una función que establece la relación entre las entradas y las salidas deseadas del sistema, estando la base de conocimientos del sistema formada por ejemplos etiquetados a priori. Es el tipo de clasificación que tratamos en este documento.

- **Aprendizaje no supervisado**

El proceso de modelado se lleva a cabo sobre un conjunto de ejemplos formados únicamente por entradas al sistema, sin conocer su clasificación correcta. Se busca que el sistema sea capaz de reconocer patrones para poder etiquetar las nuevas entradas.

- **Aprendizaje semisupervisado**

Es la combinación de los dos algoritmos anteriores, teniendo en cuenta ejemplos clasificados y no clasificados.

- **Aprendizaje por refuerzo**

El algoritmo aprende observando el mundo que le rodea y con un continuo flujo de información en las dos direcciones (del mundo a la máquina, y de la máquina al mundo), realizando un proceso de ensayo-error, y reforzando aquellas acciones que reciben una respuesta positiva en el mundo.

- **Transducción**

Similar al aprendizaje supervisado, pero su objetivo no es construir de forma explícita una función, sino únicamente tratar de predecir las categorías de los siguientes ejemplos basándose en los ejemplos de entrada, sus respectivas categorías y los ejemplos nuevos al sistema. Es decir, estaría más cerca del concepto de aprendizaje supervisado dinámico.

- **Aprendizaje multitarea**

Engloba todos aquellos métodos de aprendizaje que usan conocimiento previamente aprendido por el sistema de cara a enfrentarse a problemas parecidos a los ya vistos.

8.2.3 Algoritmos basados en el machine learning

Hay múltiples tipos de algoritmos de clasificación basados en aprendizaje automático. A la hora de elegir uno u otro hay que tener en cuenta ciertas consideraciones:

- Precisión
- Tiempo de entrenamiento
- Linealidad
- Número de parámetros
- Número de características
- Casos especiales

Lo que se trata de conseguir es un buen equilibrio entre las anteriores consideraciones, si bien habrá casos en los que primarán unas u otras.

Nosotros explicaremos cuatro algoritmos debido a su relevancia actual:

8.2.3.1 Árboles de decisión

El algoritmo tiene una única entrada y múltiples salidas, una por cada clase. Los datos a clasificar van recorriendo el algoritmo, que desde su entrada única va realizando bifurcaciones en función de una única regla por nodo (figura 8.2), de tal modo que al llegar a un nodo el dato tomará un camino u otro en función de la característica que se analice.

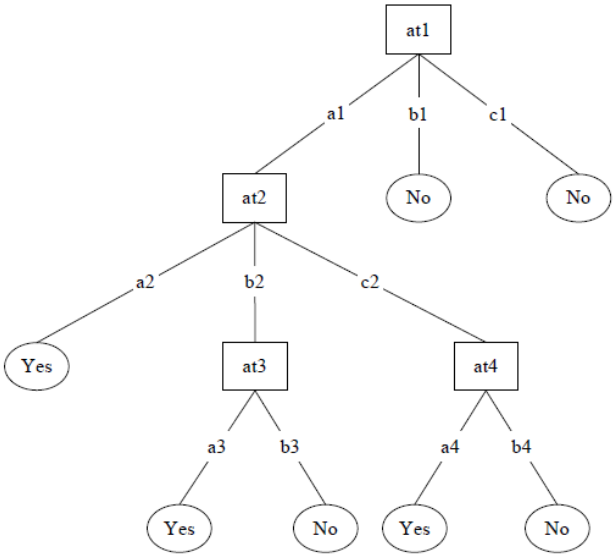


Figura 8.2: Ejemplo de árbol de decisión. [Kotsiantis, S. B.]

Cada nodo representa una característica de una instancia que pretenda ser clasificada, y cada rama representa un valor que el nodo puede asumir. Al primer nodo se le conoce como nodo raíz.

Son fáciles de interpretar y de explicar, pero requieren de la reconstrucción del árbol cuando se añaden ejemplos al clasificador y pueden sufrir fácilmente de sobreajuste (figura 8.3) si los árboles son muy grandes, que es la situación por la que un clasificador está sobreentrenado. Cuando un sistema se entrena demasiado o se entrena con datos extraños, el algoritmo de aprendizaje puede quedar ajustado a unas características muy específicas de los datos de entrenamiento que no tienen relación causal con la función objetivo.

Durante la fase de sobreajuste el éxito al responder las muestras de entrenamiento sigue incrementándose mientras que su actuación con muestras nuevas va empeorando.

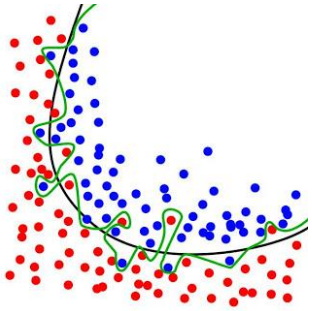


Figura 8.3: Ejemplo de sobreajuste de un clasificador. [Manzanilla, Orestes]

En estos casos el error de entrenamiento se reducirá mientras que el de validación aumentará.

8.2.3.2 Boosting

Es un proceso iterativo que combina las salidas de varios clasificadores débiles (poco mejor que un clasificador aleatorio puro, que tiene una precisión del 50%) para obtener un clasificador más eficiente.

Dentro del boosting hay distintos algoritmos diferentes: AdaBoost (Freund&Schapire, 1997), SAMME,….

La idea deriva de que en ocasiones establecer reglas de predicción de alta precisión es algo muy complejo y difícil, siendo muy sencillo establecer pequeñas reglas simples, que combinadas ofrecen los mismos resultados que una regla muy precisa. Se pueden alcanzar precisiones del orden del 99% si los datos son suficientes.

Las características de este método son las siguientes:

- Rápido (pero no tanto como otros).
- Simple y fácil de programar.
- Flexible: puede combinarse con cualquier algoritmo de aprendizaje.
- Tiende a no sobre-ajustarse.
- Tiene muchas aplicaciones.

El algoritmo utilizado en este trabajo (AdaBoost) pertenece a este método.

8.2.3.3 Máquinas de vectores soporte (SVM)

Tienen su origen en los trabajos sobre la teoría del aprendizaje estadístico y fueron introducidas en los años 90 por Vapnik y sus colaboradores. Originariamente las SVMs fueron pensadas para resolver problemas de clasificación binaria, aunque actualmente se

utilizan para resolver otros tipos de problemas (regresión, agrupamiento, multiclasificación). [Carmona Suárez, Enrique J. 2014]

Pertenecen a la categoría de los clasificadores lineales, puesto que inducen separadores lineales o hiperplanos, ya sea en el espacio original de los ejemplos de entrada, si éstos son separables o cuasi-separables (ruido), o en un espacio transformado (espacio de características), si los ejemplos no son separables linealmente en el espacio original. La búsqueda del hiperplano de separación en estos espacios transformados, normalmente de muy alta dimensión, se hará de forma implícita utilizando las denominadas funciones kernel.

Mientras la mayoría de los métodos de aprendizaje se centran en minimizar los errores cometidos por el modelo generado a partir de los ejemplos de entrenamiento (error empírico), el sesgo inductivo asociado a las SVMs radica en la minimización del denominado riesgo estructural.

La idea es seleccionar un hiperplano de separación que equidiste de los ejemplos más cercanos de cada clase para, de esta forma, conseguir lo que se denomina un margen máximo a cada lado del hiperplano (figura 8.4). Además, a la hora de definir el hiperplano sólo se consideran los ejemplos de entrenamiento de cada clase que caen justo en la frontera de dichos márgenes. Estos ejemplos reciben el nombre de vectores soporte.

Desde un punto de vista práctico, el hiperplano separador de margen máximo ha demostrado tener una buena capacidad de generalización, evitando en gran medida el problema del sobreajuste a los ejemplos de entrenamiento.

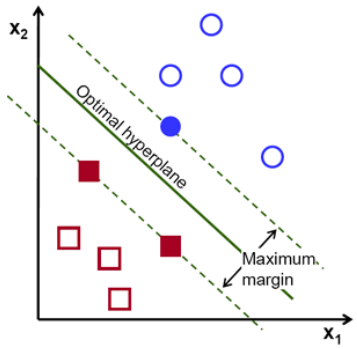


Figura 8.4: Creación de tres clasificadores débiles mediante método AdaBoost. [OpenCv]

Desde un punto de vista algorítmico, el problema de optimización del margen geométrico representa un problema de optimización cuadrático con restricciones lineales que puede ser resuelto mediante técnicas estándar de programación cuadrática. La propiedad de convexidad exigida para su resolución garantiza una solución única, en contraste con la no unicidad de la solución producida por una red neuronal artificial entrenada con un mismo conjunto de ejemplos.

Tiene dos inconvenientes importantes:

- Problema estadístico: la cantidad de datos necesaria es a menudo proporcional al número de dimensiones.
- Problema computacional: muy costoso en tiempo y memoria cuando se trabaja con grandes dimensiones.

8.2.3.4 Naive Bayes

Es uno de los clasificadores más utilizados por su simplicidad y rapidez. Se trata de una técnica de clasificación y predicción supervisada que construye modelos que predicen la probabilidad de posibles resultados. Constituye una técnica supervisada porque necesita tener ejemplos clasificados para que funcione.

Está basada en el Teorema de Bayes (teorema de la probabilidad condicionada, figura 8.5).

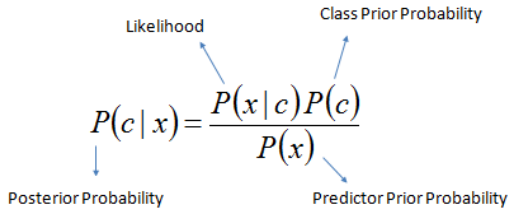


Figura 8.5: Teorema de Bayes aplicado al método de Naive Bayes. [Sayad, Saed]

El algoritmo tienes dos partes:

1. Crear el modelo.
    a) Calcular las probabilidades a priori de cada clase.
    b) Para cada clase, realizar un recuento de los valores de atributos que toma cada ejemplo. Se debe distribuir cada clase por separado para mayor comodidad y eficiencia del algoritmo.
    c) Aplicar la Corrección de Laplace, para que los valores "cero" no den problemas.
    d) Normalizar para obtener un rango de valores [0,1].

2. Clasificar un nuevo ejemplo.
    a) Para cada clase disponible, se determinan los valores de probabilidad de cada valor de los atributos del nuevo ejemplo.
    b) Aplicar la fórmula de Naïve Bayes.

8.3 Método AdaBoost

AdaBoost es el acrónimo de “Adaptative Boosting”, es decir, Boosting adaptativo, es un algoritmo de aprendizaje automático formulado por Yoav Freund y Robert Schapire.

Cuenta con múltiples versiones distintas: AdaBoost, AdaBoost.M1, AdaBoost.M2,… siendo estas tres muy conocidas debido a ser creadas por los autores del algoritmo original.

Como todo método boosting, consiste en entrenar iterativamente una serie de clasificadores débiles de tal forma que cada nuevo clasificador preste mayor atención a los datos mal clasificados en los clasificadores anteriores, para finalmente combinar todos estos clasificadores débiles y obtener un rendimiento similar al de los clasificadores fuertes.

Al inicio se considera una distribución uniforme para los pesos de las observaciones, si bien para conseguir que cada nuevo clasificador preste mayor atención a los datos clasificados erróneamente por los anteriores se emplean funciones que ponderan la importancia de cada dato durante el entrenamiento del clasificador. De este modo, los datos que han sido bien clasificados pierden peso a favor de los que fueron mal clasificados, tratando de conseguir que los nuevos clasificadores se centren en dichos datos mal clasificados (figura 8.6).

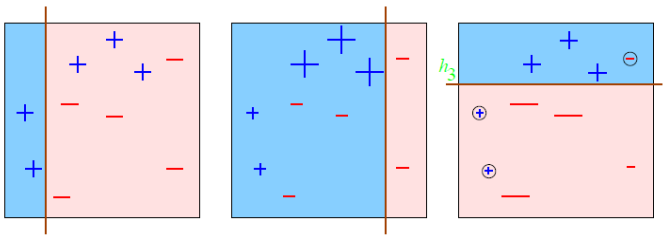


Figura 8.6: Creación de tres clasificadores débiles mediante método AdaBoost. [Schapire, Rob]

Al final del algoritmo se combinarán todos los clasificadores débiles, teniendo mayor peso en la votación final aquellos que hayan mostrado mejor rendimiento (figura 8.7). Como combinan todos los clasificadores, la clasificación del conjunto tenderá hacia una clasificación correcta de todos los datos.

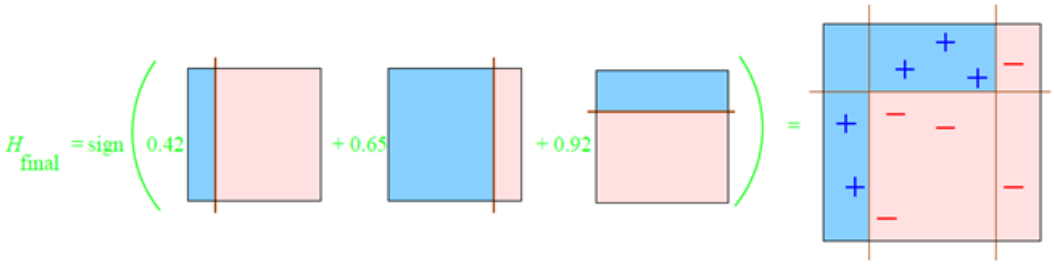


Figura 8.7: Combinación de los tres clasificadores débiles para obtener el clasificador final. [Schapire, Rob]

8.3.1 Algoritmo AdaBoost

Se trata del algoritmo original que dio lugar a la familia de algoritmos AdaBoost. Es el algoritmo de boosting más conocido para bases de datos con dos clases, y fue creado en 1995.

El funcionamiento ya se ha explicado en el apartado anterior, mostrándose en la figura 8.8 el algoritmo con más detalle [Freund, Yoav; Schapire, Robert E]:

1. Se toma como entrada del algoritmo los datos correspondientes a la muestra (S) de entrenamiento, donde se indicarán los valores de los atributos de cada ensayo (X) y la clase a las que pertenecen (Y), y se inicializa el peso de las observaciones.
2. A continuación se hacen T iteraciones, en cada una de las cuales se entrena un algoritmo de aprendizaje débil que realiza una hipótesis con un determinado error (ε). Por último se actualizan los pesos de los ensayos según haya resultado la clasificación y se repite la operación desde el paso del entrenamiento del algoritmo de aprendizaje.
3. Una vez se hayan llevado a cabo las T iteraciones se obtiene el clasificador final, donde el error de entrenamiento (figura 8.9) se ha minimizado.

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t} [h_t(x_i) \neq y_i] .$$

- Choose $\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right) .$$

Figura 8.8: Algoritmo AdaBoost original. [Freund, Yoav; Schapire, Robert E]

$$\epsilon_t = \Pr_{i \sim D_t} [h_t(x_i) \neq y_i] = \sum_{i : h_t(x_i) \neq y_i} D_t(i)$$

Figura 8.9: Error de la hipótesis final (H(x)) del algoritmo AdaBoost. [Freund, Yoav; Schapire, Robert E]

8.3.2 Algoritmo AdaBoost.M1

Esta versión permite una clasificación multiclase, si bien no se consigues resultados demasiado precisos. El funcionamiento del algoritmo es el siguiente (figura 8.10) [Freund, Yoav; Schapire, Robert E]:

1. Se toma como entrada del algoritmo los datos correspondientes a la muestra (S) de entrenamiento, donde se indicarán los valores de los atributos de cada ensayo (X) y la clase a las que pertenecen (Y), y se inicializa el peso de las observaciones.
2. A continuación se hacen T iteraciones, en cada una de las cuales el algoritmo hace llamadas a un algoritmo débil de aprendizaje (WeakLearn), construyéndose un clasificador débil con el conjunto de entrenamiento ponderado que minimiza el error de entrenamiento (ε). Por último se actualizan los pesos de los ensayos según haya resultado la clasificación y se repite la operación desde el paso de la llamada al WeakLearn.
3. Una vez se hayan llevado a cabo las T iteraciones se obtiene el clasificador final, con un error de entrenamiento minimizado (figura 8.11).

Algorithm AdaBoost.M1
Input: sequence of m examples $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$
with labels $y_i \in Y = \{1, \ldots, k\}$
weak learning algorithm **WeakLearn**
integer T specifying number of iterations

Initialize $D_1(i) = 1/m$ for all i.
Do for $t = 1, 2, \ldots, T$:

1. Call **WeakLearn**, providing it with the distribution D_t.
2. Get back a hypothesis $h_t : X \to Y$.
3. Calculate the error of h_t: $\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$.
   If $\epsilon_t > 1/2$, then set $T = t - 1$ and abort loop.
4. Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$.
5. Update distribution D_t:
   $$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t & \text{if } h_t(x_i) = y_i \\ 1 & \text{otherwise} \end{cases}$$
   where Z_t is a normalization constant (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$h_{fin}(x) = \arg\max_{y \in Y} \sum_{t:h_t(x)=y} \log \frac{1}{\beta_t}.$$

Figura 8.10: AlgoritmoAdaBoost.M1. [Freund, Yoav; Schapire, Robert E]

$$\frac{|\{i : h_{fin}(x_i) \neq y_i\}|}{m} \leq \prod_{t=1}^{T} \sqrt{1 - 4\gamma_t^2} \leq \exp\left(-2 \sum_{t=1}^{T} \gamma_t^2\right)$$

Figura 8.11: Error de la hipótesis final (h_{fin}) del algoritmoAdaBoost.M1. [Freund, Yoav; Schapire, Robert E]

8.3.3 Algoritmo AdaBoost.M2

Respecto a AdaBoost.M1 trata de mejorar la comunicación entre el algoritmo de boosting y el algoritmo débil de aprendizaje, a la par que se confiere a este último mayor capacidad para realizar hipótesis discriminantes más complejas, de tal manera que se solvente las desventajas de la anterior versión del algoritmo. Se comporta exactamente igual que AdaBoost.M1 para el caso binario, funcionando de manera distinta cuando se trabaja con varias clases (mejores resultados) [Freund, Yoav; Schapire, Robert E].

En la figura 8.12 se muestra el algoritmo en cuestión.

Algorithm AdaBoost.M2
Input: sequence of m examples $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$
with labels $y_i \in Y = \{1, \ldots, k\}$
weak learning algorithm **WeakLearn**
integer T specifying number of iterations

Let $B = \{(i, y) : i \in \{1, \ldots, m\}, y \neq y_i\}$
Initialize $D_1(i, y) = 1/|B|$ for $(i, y) \in B$.
Do for $t = 1, 2, \ldots, T$

1. Call **WeakLearn**, providing it with mislabel distribution D_t.
2. Get back a hypothesis $h_t : X \times Y \rightarrow [0, 1]$.
3. Calculate the pseudo-loss of h_t:
$$\epsilon_t = \frac{1}{2} \sum_{(i,y) \in B} D_t(i, y) \left(1 - h_t(x_i, y_i) + h_t(x_i, y)\right).$$
4. Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$.
5. Update D_t:
$$D_{t+1}(i, y) = \frac{D_t(i, y)}{Z_t} \cdot \beta_t^{(1/2)(1 + h_t(x_i, y_i) - h_t(x_i, y))}$$
where Z_t is a normalization constant (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:
$$h_{fin}(x) = \arg\max_{y \in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) h_t(x, y).$$

Figura 8.12: AlgoritmoAdaBoost.M2. [Freund, Yoav; Schapire, Robert E]

El error de entrenamiento se puede calcular de acuerdo a la expresión que se recoge en la figura 8.13., donde k hace referencia al número de clases.

$$\begin{aligned} \frac{|\{i : h_{fin}(x_i) \neq y_i\}|}{m} &\leq (k-1) \prod_{t=1}^{T} \sqrt{1 - 4\gamma_t^2} \\ &\leq (k-1) \exp\left(-2 \sum_{t=1}^{T} \gamma_t^2\right) \end{aligned}$$

Figura 8.13: Error de la hipótesis final (h_{fin}) del algoritmo AdaBoost.M2. [Freund, Yoav; Schapire, Robert E]

8.4 Métodos de evaluación de la clasificación

Se trata de uno de los procesos más difíciles en el ámbito de la clasificación, ya que se pueden dar muchos casos posibles en los que un clasificador no tiene una validez adecuada: podemos tener un clasificador que se haya ajustado a la perfección a la muestra de entrenamiento (bajo error del clasificador en el entrenamiento) pero luego muestre un gran error a la hora de clasificar la muestra de test (gran error de clasificación).También puede darse el caso de un clasificador que tenga un bajo error de clasificación pero que sea completamente incapaz de clasificar correctamente una clase en concreto, siendo un clasificador a evitar. Otro caso podría ser el que las muestras de testeo y de entrenamiento no sean representativas.

Como vemos entran en juego muchas posibilidades, por lo que se han de proporcionar las herramientas necesarias para juzgar la validez y rendimiento de los clasificadores empleados.

Las herramientas elegidas en nuestro caso son las siguientes:

8.4.1 Técnica Hold-Out

Es una técnica de evaluación de sistemas de aprendizaje donde se trabaja con muestras de entrenamiento y de testeo independientes y elegidas al azar dentro de la muestra total (en función de un porcentaje elegido en cada caso). Por tanto, de esta manera al realizar la clasificación de la muestra de testeo los resultados serán más fiables que si trabajásemos con una muestra única tanto para el entrenamiento como el testeo, aunque siempre puede darse el caso que la partición de la muestra de lugar a particiones similares en cada ejecución, por propia probabilidad estadística.

Hay que indicar se trabaja con particiones aproximadamente estratificadas: el peso de cada clase es prácticamente idéntico en ambas particiones.

8.4.2 Técnica de validación cruzada de k iteraciones (K-foldcross-validation)

Es una técnica utilizada para evaluar los resultados de un análisis estadístico para garantizar que son independientes de la partición entre datos de entrenamiento y datos de prueba. Es una mejora del método Hold-Out (método de retención), ya que se divide la muestra total en k subconjuntos (*folds*), usándose uno de ellos como datos de entrenamiento y otros como datos de prueba, y repitiéndose todo durante k iteraciones (generalmente 10, al haberse determinado en multitud de casos como el número de iteraciones más apropiado). Finalmente se calcula la media aritmética del error de clasificación obtenido de las distintas clasificaciones efectuadas.

La elección de los subconjuntos puede ser aleatoria en cada iteración o simplemente mediante una fracción de la muestra original (figura 8.14). El *Leave-one-out* es un caso particular de la validación cruzada en el que el número de subconjuntos es igual al número de ejemplos y el número de repeticiones es 1.

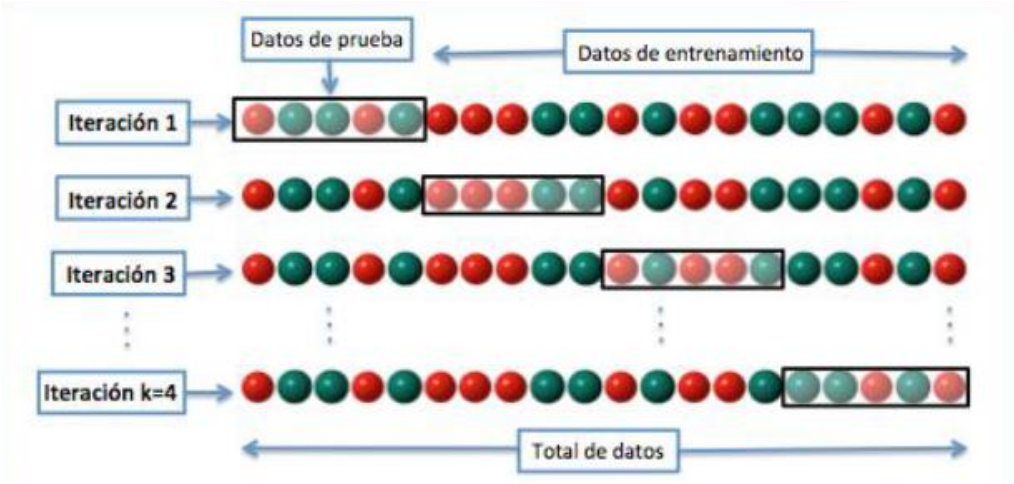


Figura 8.14: Esquema de funcionamiento del método de validación cruzada, con 4 subconjuntos (folds). [Rodríguez, Oldemar]

Cada una de las estrategias tiene sus ventajas e inconvenientes. En nuestro caso hemos optado por el método de los k subconjuntos.

8.4.3 Error de clasificación por resustitución:

Es el error del clasificador obtenido de la muestra de entrenamiento. Por sí sólo este error no es determinante, ya que para muestras posteriores presentará un error de clasificación más elevado, de magnitud desconocida. A priori es deseable un error lo más bajo posible.

8.4.4 Matriz de confusión

También conocida como matriz de error, es una matriz que incluye información sobre el rendimiento de la clasificación realizada por el algoritmo. Básicamente se trata de un tipo específico de tabla de contingencia de dos variables (clases reales de la muestra de testeo y clases obtenidas mediante el clasificador para dicha muestra) donde se muestran los verdaderos positivos (**TP**), falsos positivos (**FP**), verdaderos negativos (**TF**) y falsos negativos (**FN**) que se han obtenido en el proceso de clasificación para cada clase.

A partir de los datos mostrados en la matriz de confusión se pueden definir varios parámetros, donde para explicarlos recurriremos al ejemplo de clasificación binaria, tal y como se muestra en la figura 8.15 y en la tabla 8.1.

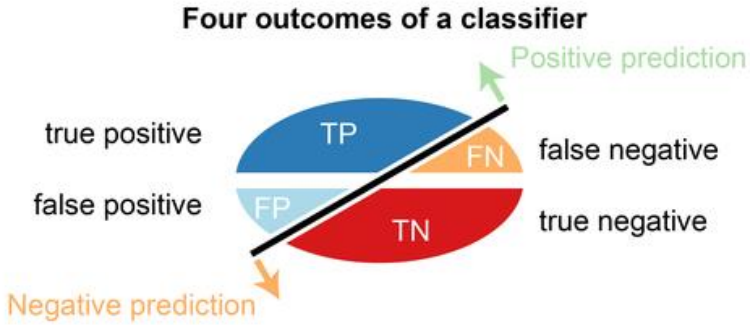


Figura 8.15: Salidas obtenidas en la clasificación de una clase: verdaderos positivos, falsos positivos, verdaderos negativos y falsos positivos. [Classeval]

Tabla 8.1: Ejemplo de matriz de confusión para una clasificación binaria de una muestra.

		Clase Predicha	
		P	***N***
Clase Real	***A***	TP (# de TPs)	FN (# de FNs)
	B	FP (# de FPs)	TN (# de TNs)

Como se deduce de la tabla 8.1, para el caso de una clasificación perfecta la matriz de confusión sería una matriz diagonal.

Algunos de los parámetros que podemos definir son los siguientes.

- **Tasa de error (ERR):** proporción de predicciones incorrectas respecto al total de la muestra. Valor comprendido entre 0 y 1.
- **Exactitud (ACC):** proporción de predicciones correctas respecto al total de la muestra. Valor comprendido entre 0 y 1.
- **Sensibilidad (SN):** también llamada *Ratio de verdaderos positivos* (**TPR**), es la proporción de las predicciones correctas positivas respecto al total original de positivos. Valor comprendido entre 0 y 1.

- **Especificidad (SP):** también llamada *Ratio de verdaderos negativos* (**FPR**), es la proporción de las predicciones correctas negativas respecto al total original de negativos. Valor comprendido entre 0 y 1.
- **Precisión (PREC):** también llamada *Valor predictivo positivo*, es la proporción de las predicciones correctas positivas respecto al total de positivos predichos. Valor comprendido entre 0 y 1.
- **Ratio de falsos positivos (FPR):** es la proporción de las predicciones incorrectas negativas respecto al total original de negativos. Valor comprendido entre 0 y 1.
- **Coeficiente de correlación de Matthews (MCC):** es un coeficiente de correlación calculado utilizando los cuatro valores que proporciona la matriz de confusión.
- **F-score (F_β):** es la media armónica de la precisión y la sensibilidad.

Tabla 8.2: Parámetros definidos a partir de la matriz de confusión.

	Parámetro	Ecuación	Nomenclatura
1	Tasa de error	$\mathrm{ERR} = \frac{(\mathrm{FP}+\mathrm{FN})}{(\mathrm{TP}+\mathrm{TN}+\mathrm{FN}+\mathrm{FP})} = \frac{(\mathrm{FP}+\mathrm{FN})}{\mathrm{P}+\mathrm{N}}$	ERR
2	Exactitud	$\mathrm{ACC} = \frac{(\mathrm{TP}+\mathrm{TN})}{\mathrm{P}+\mathrm{N}} = 1-\mathrm{ERR}$	ACC
3	Sensibilidad	$\mathrm{SN} = \frac{\mathrm{TP}}{(\mathrm{TP}+\mathrm{FN})} = \frac{\mathrm{TP}}{\mathrm{P}}$	SN o TPR
4	Especificidad	$\mathrm{SP} = \frac{\mathrm{TN}}{(\mathrm{TN}+\mathrm{FN})} = \frac{\mathrm{TN}}{\mathrm{N}}$	SP o FPR
5	Precisión	$\mathrm{PREC} = \frac{\mathrm{TP}}{(\mathrm{TP}+\mathrm{FP})}$	PREC
*	Ratio de falos positivos	$\mathrm{FPR} = \frac{\mathrm{FP}}{(\mathrm{TN}+\mathrm{FP})} = 1-\mathrm{SP}$	FPR
6	Coeficiente de correlación de Matthews	$\mathrm{MCC} = \frac{(\mathrm{TP}\cdot\mathrm{TN}-\mathrm{FP}\cdot\mathrm{FN})}{\sqrt{((\mathrm{TP}+\mathrm{FP})\cdot(\mathrm{TP}+\mathrm{FN})\cdot(\mathrm{TN}+\mathrm{FP})\cdot(\mathrm{TN}+\mathrm{FN}))}}$	MCC
7	F-score	$\mathrm{F}_\beta = \frac{((1+\beta^2)\cdot(\mathrm{PREC}+\mathrm{SN}))}{(\beta^2\cdot\mathrm{PREC}+\mathrm{SN})}$ (*)	F_β

* Donde β toma usualmente los valores de 0.5, 1 o 2, siendo el más habitual el caso de β=1.

De todos estos parámetros mostrados los dos últimos gozan de menos popularidad, y en la mayor parte de los casos se puede estimar bastante bien la validez de la clasificación únicamente con los primeros.

8.4.5 Curva ROC (Característica Operativa Relativa)

Es una representación gráfica de la razón o ratio de verdaderos positivos (**TPR**) frente a la razón o ratio de falsos positivos (**FPR**, que es equivalente a **1-SP**) según se varía el umbral de discriminación. En el fondo equivalente a representar gráficamente parte de la información proporcionada por la matriz de confusión, para una identificación más visual. En concreto se utilizan los parámetros de especificidad y de sensibilidad, ya que la primera mide el rendimiento de toda la parte negativa de la muestra mientras que la segunda mide el rendimiento de toda la parte positiva.

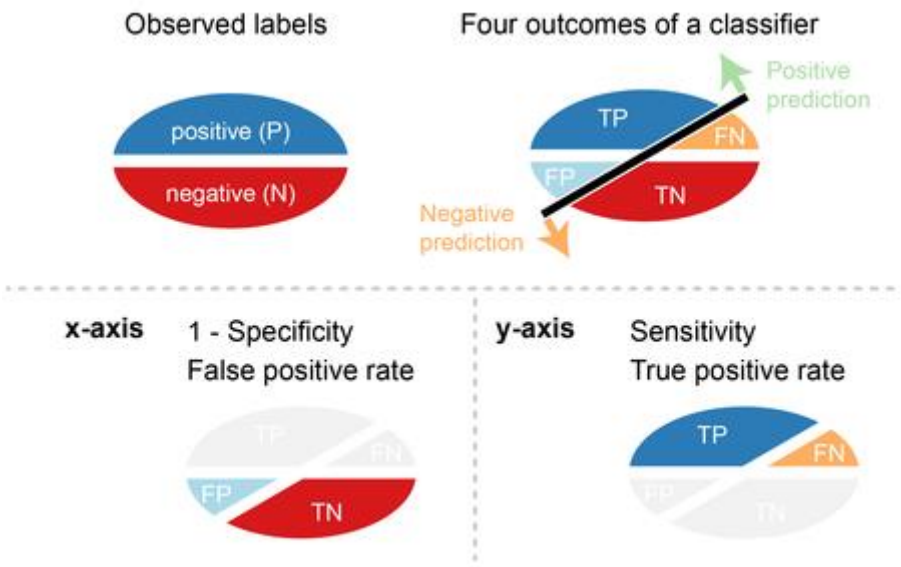


Figura 8.16: Construcción de las curvas ROC. Se parte de la especificidad y de la precisión de la clasificación realizada, obtenidas de la matriz de confusión. [Classeval]

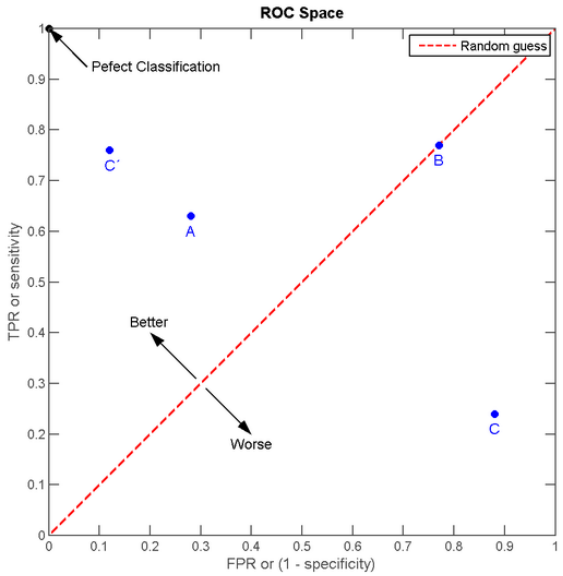


Figura 8.17: Curva ROC. Caso A: clasificación aceptable. Caso B: clasificación aleatoria. Caso C: clasificación muy mala. Caso C': clasificación buena obtenida de invertir la clasificación realizada en el caso C. [Classeval]

En el caso de MATLAB esto se representa mediante rectas que unen los puntos (0,0) y (1,1) con el punto en el espacio correspondiente a la clasificación realizada, habiendo un punto para cada una de las clases. De esta manera, cuanto mayor sea el área bajo nuestra "curva" mejor será la clasificación global, siendo el área de 1 el caso de clasificación perfecta y el área de 0.5 el de clasificación aleatoria.

La carencia de este tipo de gráficos es que son incapaces de mostrar exactamente qué clases se han clasificado erróneamente entre sí en el caso de clasificaciones multiclase, dato que si aparece en la matriz de confusión, además de otras limitaciones como la difícil comparación en ocasiones que se tiene de dos clasificadores diferentes (dos curvas ROC diferentes pueden tener el mismo área bajo la curva), además de que no es factible comparar clasificadores que han trabajado con muestras de diferente tamaño.

CAPÍTULO 9

CLASIFICACIÓN EN MATLAB

9 CLASIFICACIÓN EN MATLAB

9.1 Introducción

En MATLAB las herramientas para describir, analizar y modelar datos mediante estadística y aprendizaje automático vienen incluidas en Statistics and Machine Learning Toolbox™.

Dicha Toolbox permite utilizar estadística, descriptiva y trazados para el análisis exploratorio de datos, ajustar distribuciones de probabilidad a datos, generar números aleatorios para simulaciones Monte Carlo y realizar pruebas de hipótesis. Los algoritmos de regresión y clasificación permiten extraer inferencias de datos y crear modelos predictivos.

Para el análisis de datos multidimensionales permite identificar variables o características clave que repercuten en su modelo con selección de características secuencial, regresión por pasos, análisis de componentes principales, regularización y otros métodos de reducción de dimensionalidad [MathWorks 2016].

Hay dos opciones:

- Utilizar la app que ofrece MATLAB para realizar estas tareas.
- Utilizar el sistema convencional de funciones y script. Este último es el que hemos utilizado al considerarlo más potente aunque complejo, al requerir de conocer las distintas funciones.

La versión utilizada es la que se recoge en la figura 9.1.

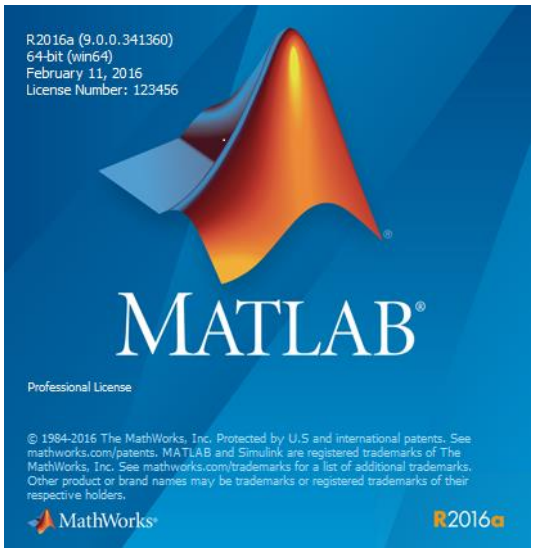


Figura 9.1: Versión de MATLAB utilizada.

9.2 Pasos

Los pasos generales y comunes a los métodos que vamos a utilizar son los siguientes:

- **Preparar los datos**

Los datos de entrada al método de clasificación han de estar preparados de la manera exigida por los métodos que se van a utilizar.

- **Elegir un algoritmo de clasificación**

Dependerá de las necesidades del usuario. En nuestro caso se trabajará con los algoritmos *ClassificationorRegressionEnsembles* y *ClassificationTrees*. El primero es donde se encuentran los métodos AdaBoost, y en el segundo el método de clasificación mediante árbol de decisiones, cuyo uso se explicará más adelante.

En la figura 9.2 se recogen los distintos algoritmos de clasificación con que cuenta MATLAB.

Algorithm	Fitting Function
Classification Trees	`fitctree`
Regression Trees	`fitrtree`
Discriminant Analysis (classification)	`fitcdiscr`
k-Nearest Neighbors (classification)	`fitcknn`
Naive Bayes (classification)	`fitcnb`
Support Vector Machines (SVM) for classification	`fitcsvm`
SVM for regression	`fitrsvm`
Mutliclass models for SVM or other classifiers	`fitcecoc`
Classification or Regression Ensembles	`fitensemble`
Classification or Regression Tree Ensembles (e.g., random Forests [9]) in Parallel	`TreeBagger`

Figura 9.2: Algoritmos de clasificación y sus funciones de ajuste en MATLAB. [MathWorks 2016]

En la figura 9.3 se reflejan las características típicas de varios de estos algoritmos, para facilitar una elección inicial que se ajuste lo más posible a las necesidades del usuario. Los resultados que aparecen reflejados en ella provienen 7000 observaciones, 80 predictores y 50 clases.

Classifier	Multiclass Support	Categorical Predictor Support	Prediction Speed	Memory Usage	Interpretability
Decision trees — `fitctree`	Yes	Yes	Fast	Small	Easy
Discriminant analysis — `fitcdiscr`	Yes	No	Fast	Small for linear, large for quadratic	Easy
SVM — `fitcsvm`	No. Combine multiple binary SVM classifiers using `fitcecoc`.	Yes	Medium for linear. Slow for others.	Medium for linear. All others: medium for multiclass, large for binary.	Easy for linear SVM. Hard for all other kernel types.
Naive Bayes — `fitcnb`	Yes	Yes	Medium for simple distributions. Slow for kernel distributions or high-dimensional data	Small for simple distributions. Medium for kernel distributions or high-dimensional data	Easy
Nearest neighbor — `fitcknn`	Yes	Yes	Slow for cubic. Medium for others.	Medium	Hard
Ensembles — `fitensemble`	Yes	Yes	Fast to medium depending on choice of algorithm	Low to high depending on choice of algorithm.	Hard

Figura 9.3: Algoritmos de clasificación más utilizados y descripción de las características habituales de estos algoritmos. [MathWorks 2016]

Siendo:

Tabla 9.1: Significado de los términos Velocidad y Memoria en la figura 9.2

Velocidad		Memoria	
Rápida	0.01s	Reducida	1MB
Media	1s	Media	4MB
Lenta	100s	Grande	100MB

- **Ajustar un modelo**

Que dependerá del algoritmo de clasificación utilizado. En las funciones de llamada de los mismos se pueden incluir múltiples entradas para modificar su comportamiento, ajuste, etc.

- **Elegir un método de validación**

Donde se propone analizar el error de resustitución, realizar una validación cruzada o examinar el error out-of-bag para los árboles de decisión creados mediante el método bagging.

- **Examinar el ajuste y reajustar**

Donde se recomienda que antes de la validación se estudie:

- Cambiar parámetros de ajuste para mejorar el modelo.
- Cambiar parámetros de ajuste para conseguir un modelo más pequeño.
- Probar un algoritmo distinto.

- **Usar el modelo de clasificación para realizar predicciones**

Donde se nos indica utilizar la función con la siguiente estructura:

```
Ypredicted=predict(modelo clasificación, datos a clasificar)
```

- **Validar los resultados**

Analizando los indicadores considerados en el capítulo anterior.

- **Compactar el clasificador**

Es un paso opcional, que consiste en eliminar los datos de entrenamiento del objeto que se ha creado con el modelo clasificador. De esta forma aligeramos el tamaño en memoria de este objeto, si bien ya no podremos trabajar con los datos que tenía almacenados.

9.3 Funciones de clasificación empleadas

El objeto de este trabajo es utilizar el método AdaBoost en MATLAB para la clasificación de los ensayos de fallos por rotura de barra en motores de inducción. Inicialmente se trabajó exclusivamente con este método, pero se encontró una complicación inesperada: si se obtiene un error de entrenamiento nulo en la primera iteración del algoritmo AdaBoost, el modelo de clasificación entrenado es incapaz de realizar predicciones válidas.

Es debido a la manera en que está programado el algoritmo, por lo que las únicas opciones son escribir el algoritmo por cuenta propia o utilizar otro método de clasificación.

Se ha optado por lo segundo, ya que un error nulo en la primera iteración implica la clasificación más sencilla posible, por lo que realmente no es necesario recurrir a métodos de boosting, obteniendo resultados perfectos mediante los clasificadores de árbol de decisiones, que es lo que se ha utilizado en dichos casos.

Ambas funciones nos permiten emplear los métodos de validación de la clasificación propuestos en el apartado anterior (validación cruzada, error de resustitución, matriz de confusión y curva ROC), estando definidas la forma de hacerlo en el código recogido en los anexos.

9.3.1 Fitctree

Es la función para realizar modelos de árboles binarios de clasificación para clasificación multiclase. Tienen la forma que se observa en la figura 9.4, siendo la estructura de su llamada como la que se encuentra a continuación, donde *acción especial* hace referencia a utilizar alguna de las posibilidades que nos permite la función (validación cruzada, etc.):

```
tree = fitctree(predictores,clasesrespuesta,acciónespecial,valor de dicha opción)
```

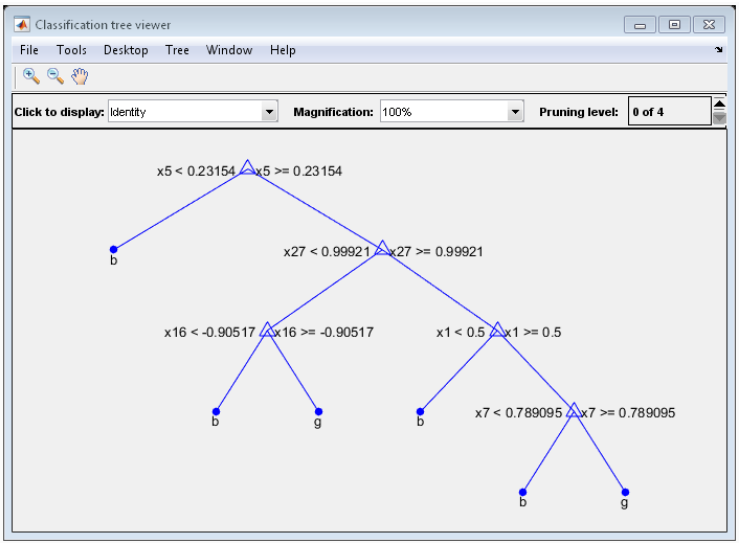


Figura 9.4: Ejemplo de árbol de decisión binario en MATLAB. [MathWorks 2016]

9.3.2 Fitensemble

Es la función para realizar clasificadores compuestos, formados por múltiples clasificadores débiles. Para crear estos clasificadores se necesita en esencia la información mostrada en la figura 9.5.

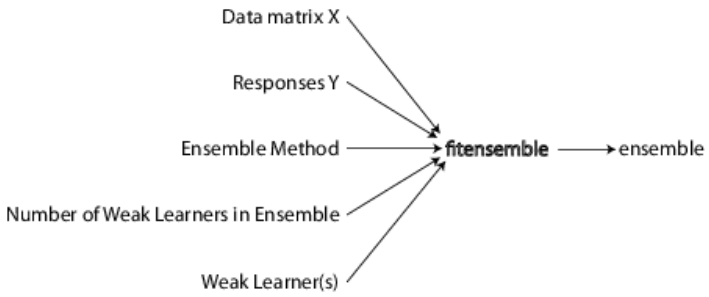


Figura 9.5: Información mínima necesaria para obtener los clasificadores compuestos en MATLAB. [MathWorks 2016]

Es la función que utilizaremos para aplicar el método AdaBoost. Los pasos para utilizarla son los siguientes:

1. Poner los predictores en una matriz

   Donde las filas representan cada ensayo y las columnas un predictor distinto.

2. Preparar los datos de respuesta en un vector

   Donde cada elemento representa la clase a la que pertenece cada ensayo en cuestión.

3. Elegir el método

   En nuestro caso se nos ofrece AdaBoostM1 para la clasificación de dos clases y AdaBoostM2 para la clasificación de tres o más clases. En la figura 9.6 se muestran algunos de estos métodos posibles, y el uso para el que se destinan.

Algorithm	Regress.	Binary Classif.	Binary Classif. Multi- Level Preds.	Classif. 3+ Classes	Class Imbalance	Stop	Sparse
Bag	×	×		×			
AdaBoostM1		×					
AdaBoostM2				×			
LogitBoost		×	×				
GentleBoost		×	×				
RobustBoost		×					
LPBoost		×		×		×	×
TotalBoost		×		×		×	×
RUSBoost		×		×	×		
LSBoost	×						
Subspace		×		×			

Figura 9.6: Tabla con ciertas características de distintos métodos de clasificación mediante la función fitensemble. [MathWorks 2016]

4. Establece el número de componentes del clasificador

   Es decir, elegir el tamaño del clasificador, siendo un compromiso entre velocidad y precisión (cuidado con el sobreajuste).

5. Preparar los algoritmos de aprendizaje débil

   Actualmente son de tres tipos:

   - Discriminantes.
   - KNN.
   - Árbol: es el tipo elegido, ya que los otros son recomendados para otros algoritmos.

6. Realizar la llamada a la función

La función tiene la forma siguiente (varía en función de las entradas suministradas):

```
ens=fitensemble(predictores,clasesrespuesta,modelo,número de
clasificadores algoritmo de aprendizaje,acciónespecial,valor de
dicha opción)
```

CAPÍTULO 10

SOLUCIÓN IMPLEMENTADA EN MATLAB

10 SOLUCIÓN IMPLEMENTADA EN MATLAB

10.1 Introducción

Como se indicó previamente, el objeto de este trabajo es el de obtener una herramienta software basada en MATLAB que permita llevar a cabo una clasificación mediante el método AdaBoost.

Se ha optado por crear inicialmente tres scripts con diferente enfoque:

- Un script denominado **'Evaluador_Clasif_AdaBoostM1'**con la idea de evaluar el funcionamiento del método de clasificación de MATLAB *AdaBoostM1* respecto a los datos de que disponemos.

- Un script denominado **'Evaluador_Clasif_AdaBoostM2'**con la idea de evaluar el funcionamiento del método de clasificación de MATLAB *AdaBoostM2* respecto a los datos de que disponemos.

- Un script denominado **'Metodos_Clasif_Multi_AdaBoost'** donde se pueden seleccionar diversos parámetros que permiten elegir entre los distintos sistemas de clasificación, así como afectar al mismo proceso de clasificación en sí.
  Esto se ha hecho así para poseer una herramienta software versátil que podrá realizar tanto el entrenamiento de los clasificadores como la acción de clasificar, eligiéndose por el usuario la acción deseada.

De esta manera se espera que según la situación requerida se recurra a uno u otro código, ya que en esencia se trata del mismo algoritmo preparado para ser utilizado de una u otra manera.

Todo el código utilizado se muestra en los ANEXOS de este documento. Se ha cuidado de que este código esté lo suficientemente detallado y explicado (explicaciones incorporadas al propio código) como para que no se entre aquí en un detalle excesivamente profundo, sino más bien para indicar:

- Esquema y funcionamiento de los distintos métodos/sistemas de clasificación utilizados. No confundirlo con el método de clasificación propiamente dicho (*AdaBosstM1*, *AdaBosstM2*, *Tree*,…), ya que en este caso hace referencia al uso aislado o en conjunto de las funciones de clasificación de MATLAB.
- Esquema y funcionamiento global de los scripts creados.

En el apartado de *RESULTADOS OBTENIDOS Y ANÁLISIS* es donde se entrará a analizar los distintos ensayos y resultados obtenidos, limitándose en este a explicar los distintos clasificadores utilizados, los distintos métodos ideados para utilizar dichos clasificadores, ya sea de manera individual o como una combinación de varios, y el funcionamiento de cada caso.

10.2 Explicación de la solución propuesta

El software está diseñado para utilizar como datos de entrada un archivo de Excel que tenga un formato equivalente al Excel con el que se ha trabajado en este caso (figuras 10.1 y 10.2).

En él vienen recogidos los ensayos realizados, indicando en cada caso bajo qué condiciones se han realizado, cuál es el estado del motor (lo que serán las clases a la hora de la clasificación) y distintos indicadores y características que serán los predictores utilizados para entrenar el clasificador así como para realizar posteriormente la clasificación. Cada fila del Excel corresponderá a un ensayo diferente.

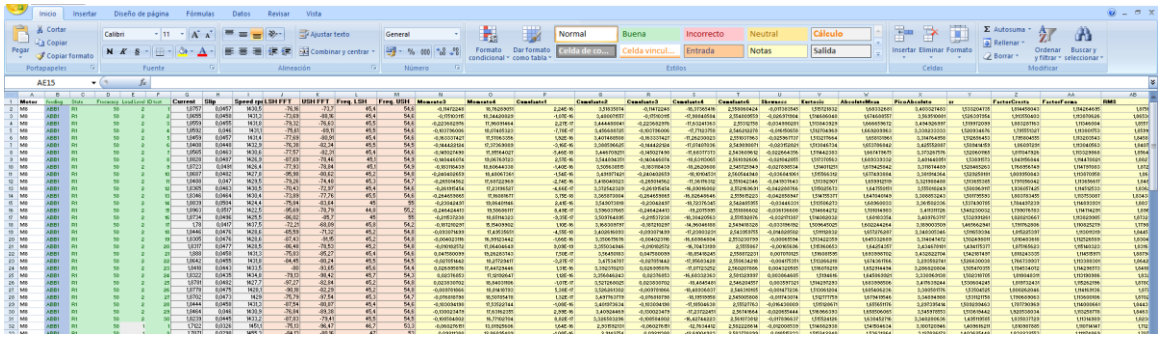


Figura 10.1: Visión general del formato de Excel con el que se va a trabajar.

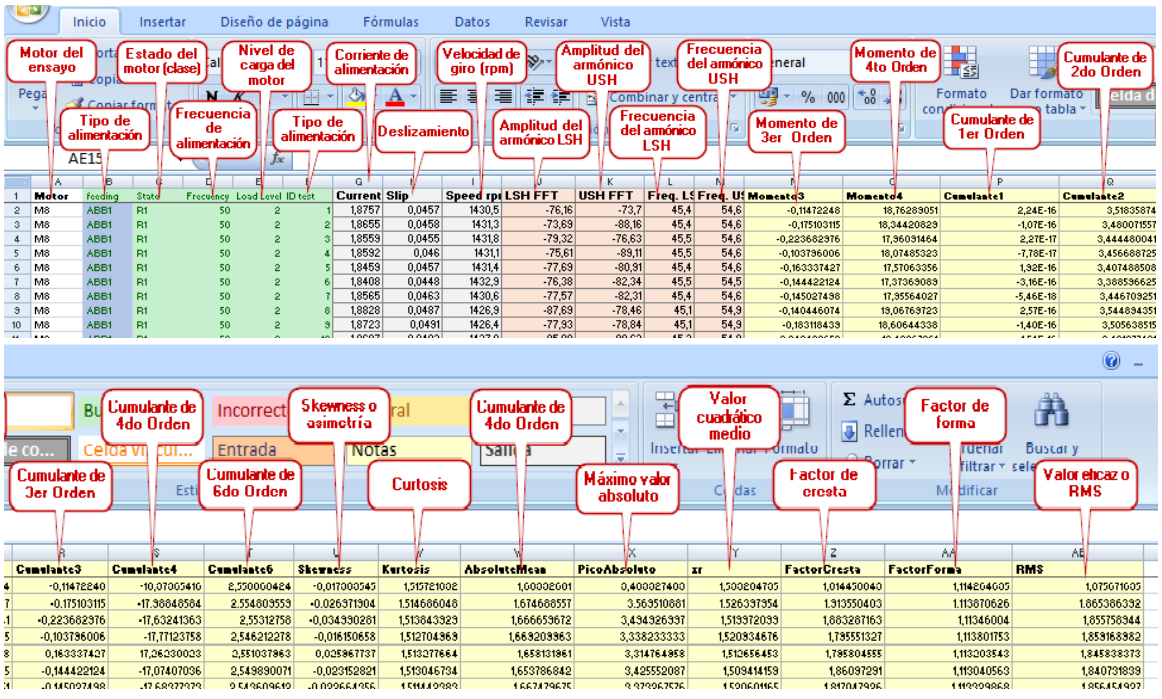


Figura 10.2: Tipo de información esperable en cada columna del Excel. Parte superior: columnas de la A a la Q. Parte inferior: columnas de la R a la AB.

En el algoritmo se ha considerado que las clases de clasificación serán siempre 5 (siendo su denominación la letra "R" seguida de un número del 1 al 5), por lo que si hubiera un cambio al respecto habría que modificar para ello el código, añadiendo o eliminando clasificadores, exceptuando en la clasificación mediante un único clasificador multiclase (*AdaBoostM2*), o aquellos otro métodos en los que participe un clasificador multiclase, si bien el rendimiento de la clasificación se vería resentido.

La herramienta software desarrollada permite seleccionar aquellos datos de la muestra total que tengan una determinada alimentación (de red o mediante variador) o nivel de carga (nivel de carga 1 o 2, que corresponden a media y plena carga respectivamente), pudiéndose trabajar con todos los datos si se requiere.

Se ha optado por combinar clasificadores binarios (*AdaBosstM1*) y clasificadores multiclase (*AdaBosstM2*) con la idea de tener clasificadores más robustos, así como para ir analizando el comportamiento, potencial y validez de los mismos.

Como se parte de las funciones de clasificación propias de MATLAB, primeramente ha sido necesario analizar el funcionamiento y validez de las mismas, para posteriormente utilizarlas de manera conjunta y conseguir el objetivo de lograr una clasificación multiclase.

La combinación de varios clasificadores requerirá de más tiempo y potencia de cálculo debido tanto al entrenamiento necesario de cada uno de ellos como al algoritmo creado para la clasificación de la partición de testeo, del procesamiento de los resultados, etc. si bien se ha comprobado que los tiempos con los que se trabaja son lo suficientemente reducidos (menores al minuto en el ordenador en el que se han realizado los experimentos) como para ser una opción viable y satisfactoria siempre y cuando supongan alguna mejora en la clasificación.

Hay que tener en cuenta que algunas de las ventajas del método AdaBoost son la sencillez y relativa rapidez del mismo, por lo que añadir excesiva complejidad y tiempo de cálculo va en contra de la misma filosofía del método.

10.3 Análisis de las funciones de clasificación de MATLAB

Inicialmente se ha optado por realizar la clasificación individual de cada clase mediante clasificadores binarios, y posteriormente la clasificación multiclase mediante el método *AdaBoostM2*. El objetivo de esto es analizar de forma más sencilla e independiente el comportamiento de las funciones de clasificación que se utilizarán.

10.3.1 Clasificación simple binaria

Se utiliza la función de MATLAB *fitensemble* con *AdaBoostM1*, entrenándose el clasificador de manera que separe la clase deseada (R1, R2, R3, R4 o R5) del resto.

Como se indicó previamente, el método *AdaBoostM1* no está diseñado para que el error del clasificador en el primer ciclo sea nulo, por lo que si se produce dicho caso el clasificador creado no tendrá utilidad. Que el error del clasificador sea 0 indica que realmente no es necesario aplicar un método iterativo como es AdaBoost, por lo que se recurrirá a una clasificación del tipo *Tree* cuando esto ocurra.

10.3.2 Clasificación simple multiclase

Se utiliza la función de MATLAB *fitensemble* con *AdaBoostM2*, siendo un clasificador multiclase, por lo que deberá identificar cualesquiera de los ensayos introducidos en el test como su correspondiente clase, dentro del error de clasificación que se produzca.

10.4 Métodos de clasificación multiclase propuestos

Estos métodos son los utilizados en los scripts facilitados, y están preparados para clasificar con mayor o menor acierto el conjunto de los ensayos en sus respectivas clases. Es decir, son algoritmos diseñados para la clasificación multiclase. Se ha buscado mejorar la clasificación "simple" que se ha descrito en los apartados anteriores (*AdaBoostM2*) a la par que crear algoritmos que se puedan servir de clasificadores binarios para efectuar clasificaciones multiclase.

10.4.1 Método 1

En esencia se trata de utilizar una cascada de 4 clasificadores binarios, orientados a R1, R2, R3 y R4 respectivamente, donde se va determinando la clase de cada ensayo a través de la aplicación sucesiva de los clasificadores (empezando por el correspondiente a R1 hasta llegar al de R4) siempre y cuando el clasificador previo haya clasificado ese ensayo como uno perteneciente a otra clase.

Es decir, partiendo de un clasificador enfocado a la clase R1, si obtuviéramos un resultado negativo (es decir: "ensayo perteneciente a otra clase distinta de R1") entraría en juego un clasificador enfocado a la clase R2, y así sucesivamente hasta llegar al clasificador enfocado a la clase R4, ya que en caso de resultado negativo de este último clasificador indicaría inequívocamente que se trataría de un ensayo perteneciente a la clase R5.

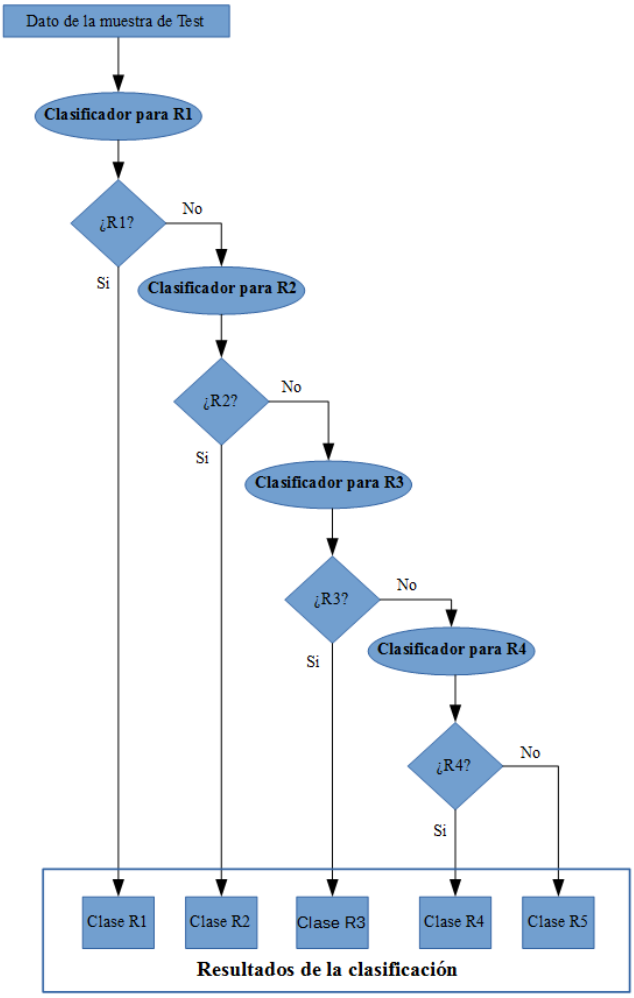


Figura 10.3: Esquema de funcionamiento del método 1.

10.4.2 Método 2

Es equivalente al caso de clasificación simple multiclase ya tratado anteriormente, por lo que no se entrará en mayor detalle.

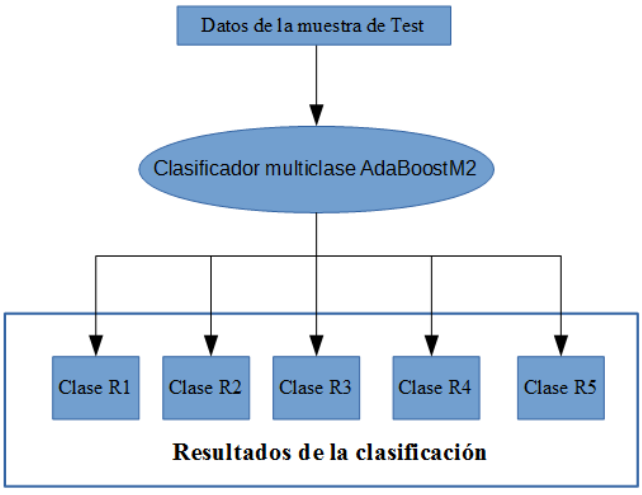


Figura 10.4: Esquema de funcionamiento del método 2.

10.4.3 Método 3

Es similar al caso de la cascada de clasificadores binarios, utilizándose un clasificador multiclase (*AdaBosstM2*) para determinar las clases R2, R3, R4 y R5.

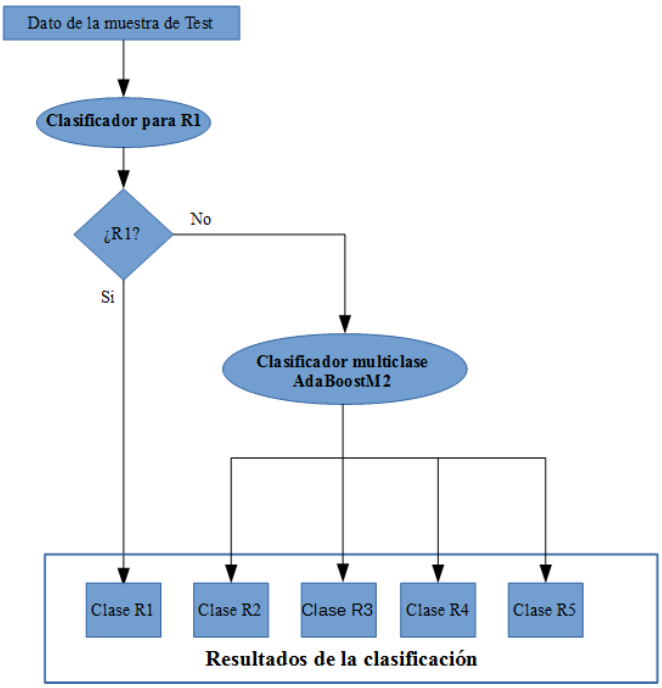


Figura 10.5: Esquema de funcionamiento del método 3.

10.4.4 Método 4

Es similar al caso de la cascada de clasificadores binarios, utilizándose un clasificador multiclase (*AdaBosstM2*) para determinar las clases R3, R4 y R5.

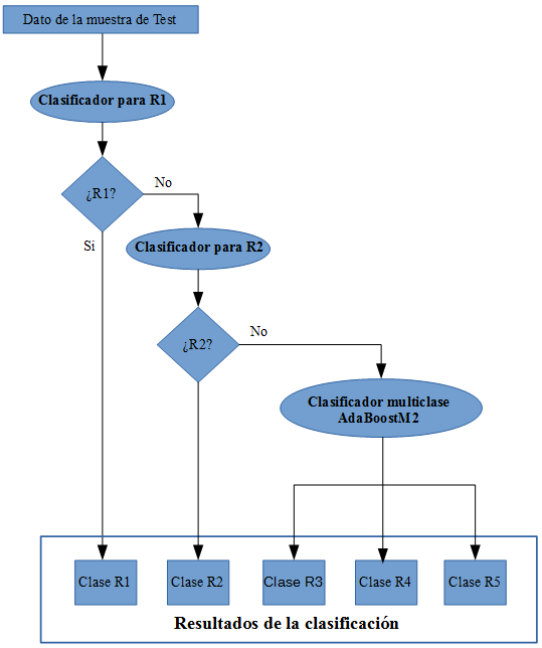


Figura 10.6: Esquema de funcionamiento del método 4.

10.4.5 Método 5

Se aplica simultáneamente a cada ensayo un clasificador binario para identificar cada clase. En los casos en que haya discrepancias (un ensayo clasificado como positivo por varios clasificadores, o clasificado como negativo por todos) se utilizará un clasificador multiclase (*AdaBosstM2*) para determinar la clase correspondiente.

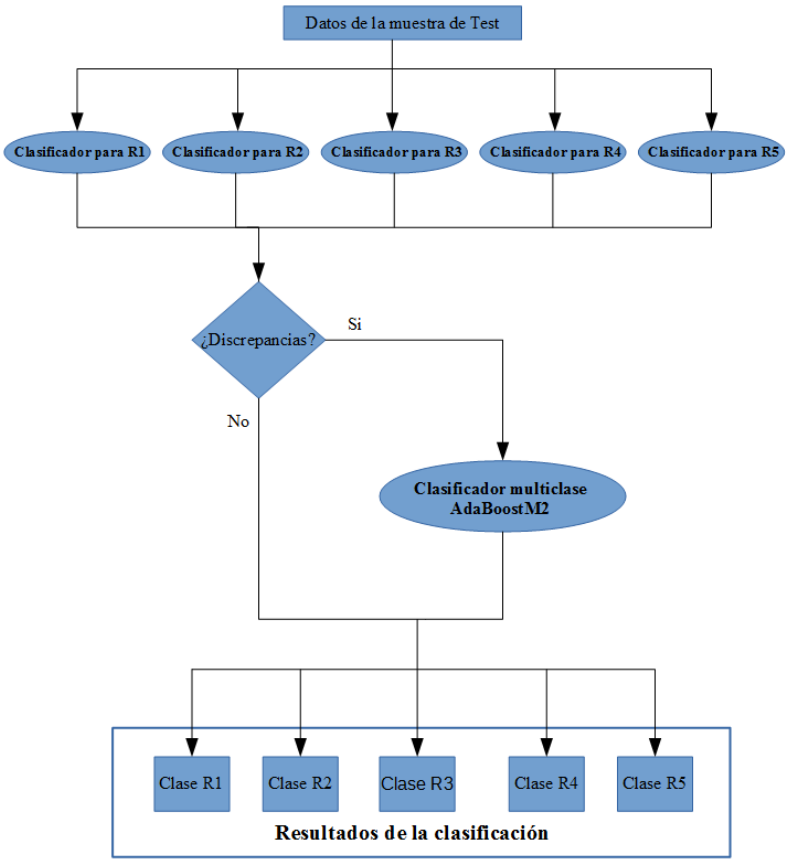


Figura 10.7: Esquema de funcionamiento del método 5.

10.4.6 Método 6

Es similar al método anterior, con la salvedad de que se aplica simultáneamente a cada ensayo un clasificador binario correspondiente a cada clase y un clasificador multiclase (*AdaBosstM2*), de tal manera que se priorizan los resultados correspondientes a la clase R1.

Sólo cuando el clasificador binario enfocado a R1 y/o el multiclase clasifican el ensayo como correspondiente a una clase distinta de R1 entran en juego las clasificaciones correspondientes al resto de clasificadores binarios. Si por la razón que fuera hay alguna discrepancia entre estos últimos, podría deberse a dos motivos:

- Que varios clasificadores binarios presenten resultados positivos. De ser así se utiliza el clasificador multiclase para el desempate de los resultados.
- Que ninguno muestre un resultado positivo. Entonces se consulta al clasificador multiclase. Si indica que la clase es R1, se tomará como clase R4, ya que es un caso de clasificación errónea muy común, además de que se ha descartado que se trate de la clase R1 con el clasificador binario.

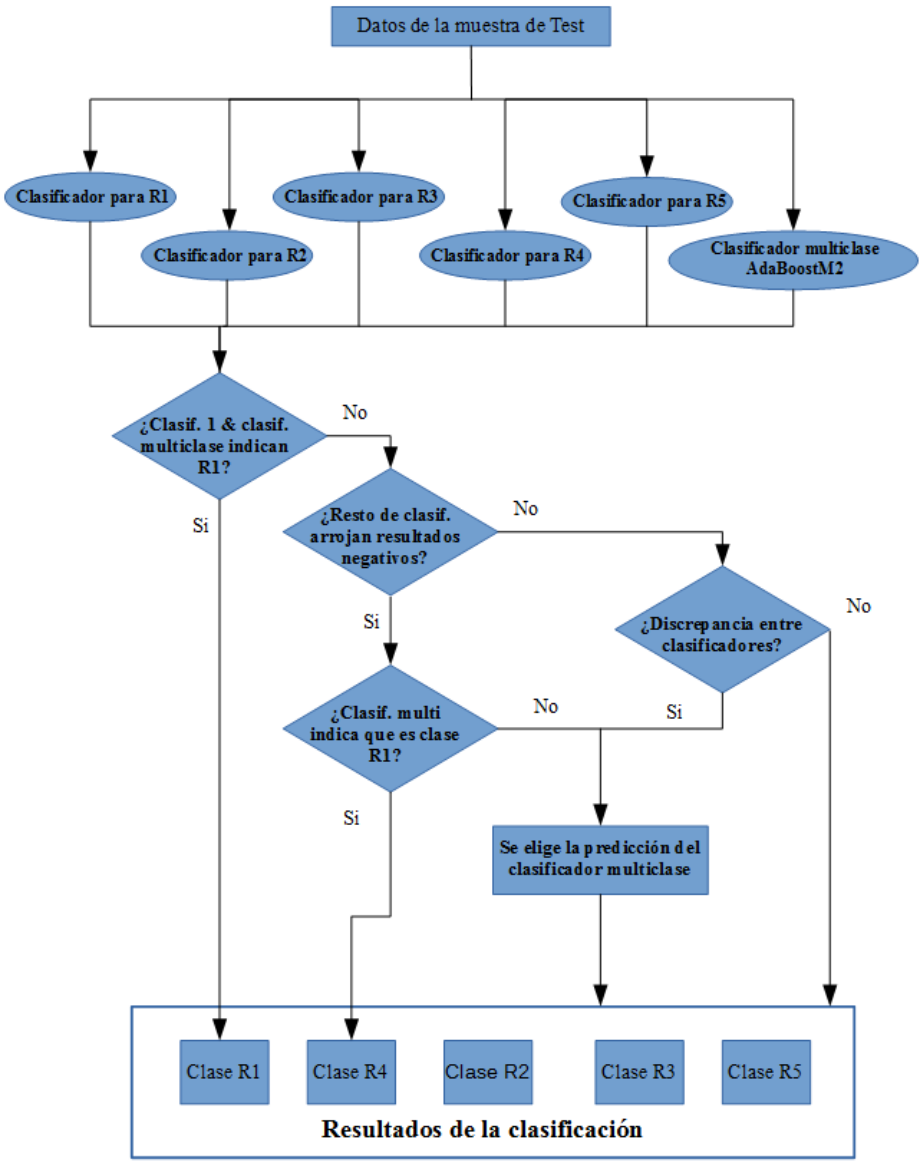


Figura 10.8: Esquema de funcionamiento del método 6.

10.5 Esquemas y funcionamiento global de los scripts

10.5.1 Evaluador_Clasif_AdaBoostM1 yEvaluador_Clasif_AdaBoostM2

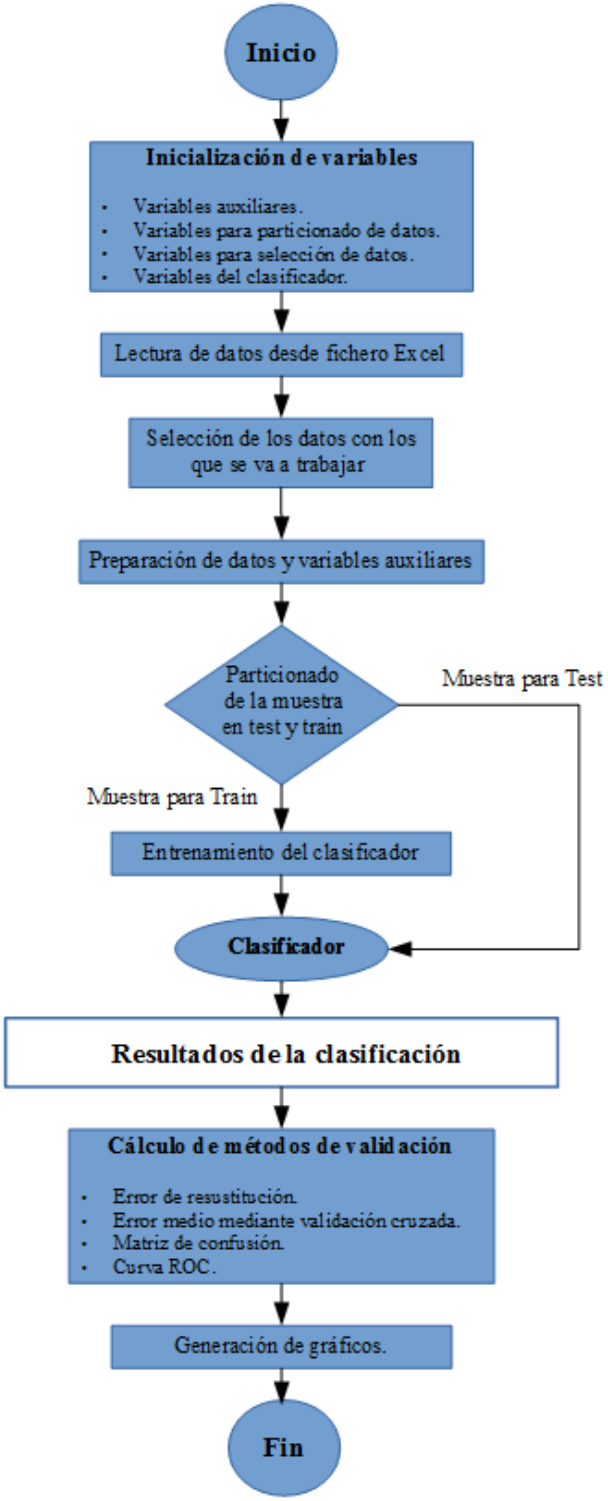


Figura 10.9: Esquema de funcionamiento de los scripts '*Evaluador_Clasif_AdaBoostM1*'y *'Evaluador_Clasif_AdaBoostM2'*.

10.5.2 Metodos_Clasif_Multi_AdaBoost

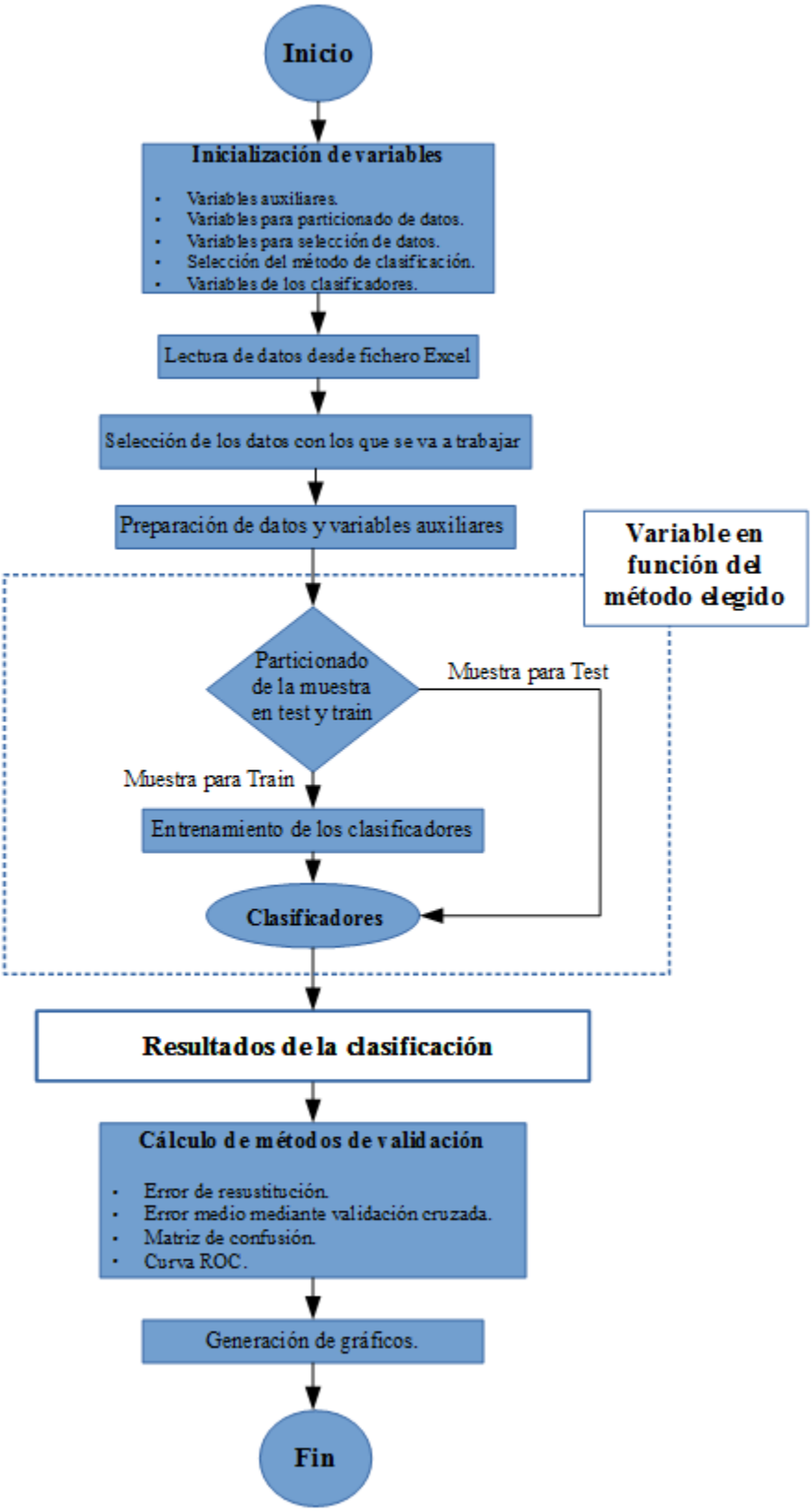


Figura 10.10: Esquema de funcionamiento del script '*Metodos_Clasif_Multi_AdaBoost*'.

CAPÍTULO 11

RESULTADOS OBTENIDOS Y ANÁLISIS

11 RESULTADOS OBTENIDOS Y ANÁLISIS

11.1 Introducción

En este capítulo se analizan los resultados obtenidos en cada uno de los ensayos realizados, analizándose de cara a obtener las conclusiones finales recogidas en el capítulo 12 de esta memoria. Todos los gráficos obtenidos se recogen en los ANEXOS, mostrándose aquí únicamente aquellos considerados interesantes. Las conclusiones y análisis realizados se obtienen de los gráficos recogidos en el mismo, así de las experiencias recopiladas a la hora de realizar y depurar el código del software.

11.2 Clasificadores Binarios

11.2.1 Introducción

Se ha utilizado en todos los casos una semilla de generación de número aleatorio igual a 1, de cara a permitir la reproducción de los ensayos realizados.

El número de iteraciones (cantidad de clasificadores débiles que forman el clasificador) se ha tomado inicialmente de 50, en función de cuándo se estabiliza el error mediante validación cruzada, de que el tiempo de computación no sea muy elevado y que se evite el efecto del sobreajuste (figura 11.1). Sin embargo, para estudiar con más detalle el error de validación cruzada se trabajará en algún caso con 150 iteraciones, indicándose cuando sea así, y siempre habiéndose comprobado que no afecte a la precisión del clasificador.

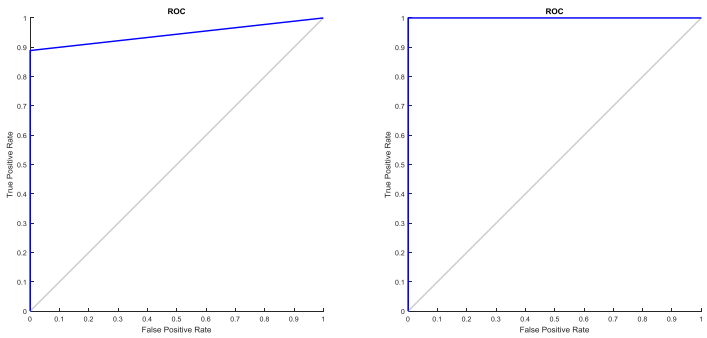


Figura 11.1: Comparación de curvas ROC al variar el número de clasificadores débiles. Clasificación binaria clase R1. Izquierda: 500 clasificadores débiles. Derecha: 50 clasificadores débiles.

El número de subconjuntos (folds) tomado a la hora de realizar la validación cruzada es de 20, ya que el tiempo de computación para 80 folds es de 52 segundos, para 50 de 32 segundos, y 16 segundos para 20folds. Además, se comprueba en todos estos casos como el error obtenido de validación cruzada prácticamente el mismo en todos estos casos (figura 11.2). El número de folds habitual en estos casos suele ser del orden de 10, habiendo elegido 20 para asegurar una mejor validación.

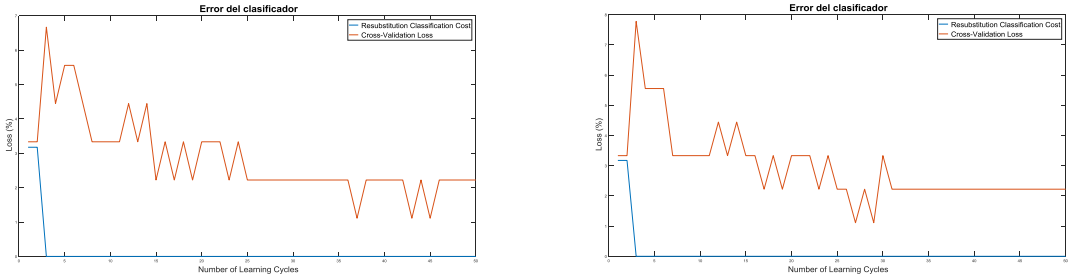


Figura 11.2: Comparación de los errores al variar el número de folds en la validación cruzada. Clasificación binaria clase R1. Izquierda: 20 folds. Derecha: 80 folds.

11.2.2 Clasificador Binario para clase R1

11.2.2.1 Resultados al variar la alimentación

Se observan mejores resultados al trabajar con alimentación de red que con alimentación vía variador de frecuencia, tanto para la precisión de la clasificación concreta estudiada (figura 11.3) como analizando el error mediante validación cruzada (figura 11.5).

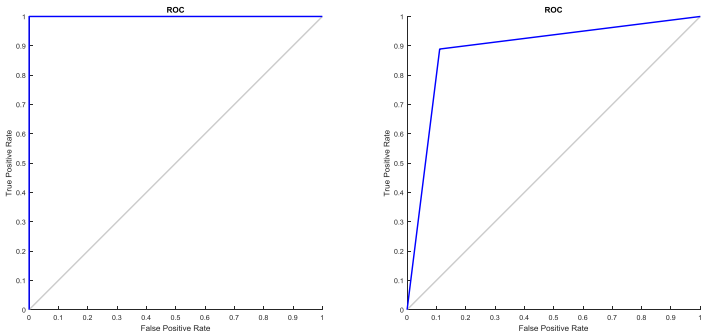


Figura 11.3: Comparación de curvas ROC al variar la alimentación. Clasificación binaria clase R1, nivel de carga 1. Izquierda: alimentación de Red. Derecha: alimentación mediante convertidor.

En el caso de los errores, para la alimentación mediante convertidor se ha observado como el error medio (validación cruzada) empeora tanto en forma (variación no escalonada, con múltiples máximos y mínimos relativos) como cuantitativamente (error mucho mayor en todos los ciclos de entrenamiento.). En el caso del error de resustitución se observa como al clasificador le cuesta ajustar más el modelo que en caso de alimentación por Red.

Al trabajar con cualquier tipo de alimentación al modelo le cuesta más ciclos ajustarse. Los mejores resultados se obtienen para el caso de alimentación de Red, y los peores para la clasificación conjunta de alimentación de Red y mediante convertidor (figura 11.4).

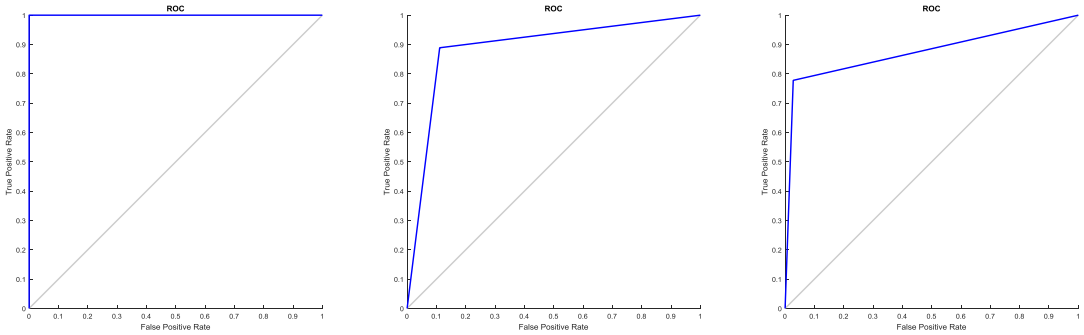


Figura 11.4: Comparación de curvas ROC al variar la alimentación. Clasificación binaria clase R1, grado de carga 1r. Izquierda: alimentación de Red. Centro: alimentación mediante convertidor. Derecha: alimentación conjunta.

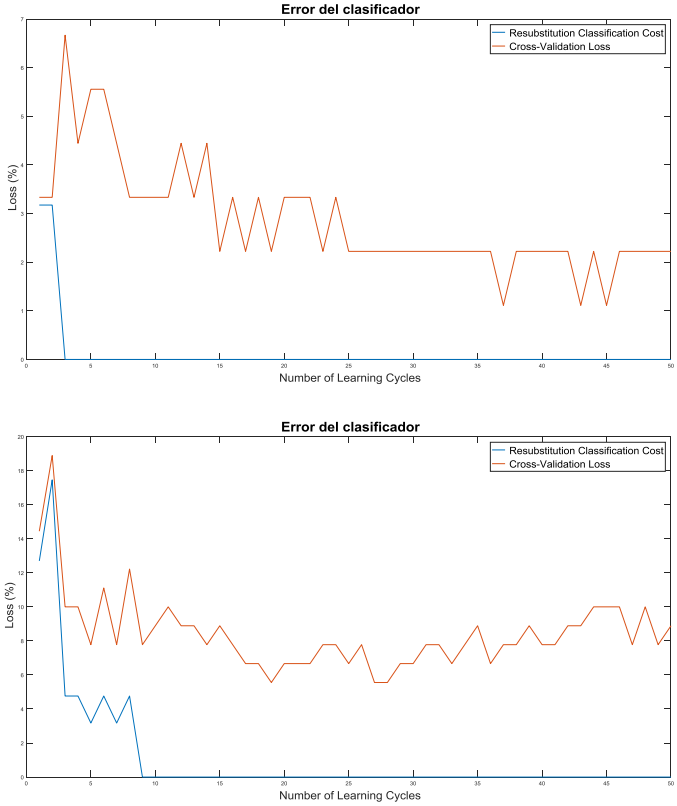


Figura 11.5: Comparación de los errores al variar la alimentación. Clasificación binaria clase R1, nivel de carga 1. Arriba: alimentación de Red. Abajo: alimentación mediante convertidor.

11.2.2.2 Resultados al variar el grado de carga

Con grados de carga de nivel 1 el modelo del clasificador se ajusta más fácilmente que para grados de carga de nivel 2 (plena carga).

El error medio de clasificación para el caso de nivel de carga 1 presenta un comportamiento monótamente decreciente hasta alcanzar un valor en el que se mantiene prácticamente estable, obviando pequeños picos, mientras que para el caso de nivel de carga 2 el error medio tiene valores más reducidos para un pequeño número de iteraciones que el que toma posteriormente (figura 11.7). El valor absoluto de este error es mayor en el caso de un nivel de carga 2.

Los ensayos realizados con ambos grados de carga simultáneamente presentan mejores

resultados que aquellos correspondientes al grado de carga 2, pero peores a los del grado de carga 1 (figura 11.6).

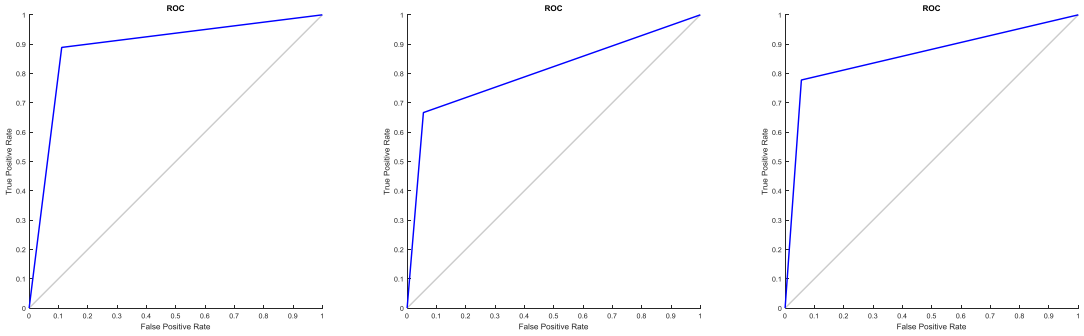


Figura 11.6: Comparación de curvas ROC al variar el grado de carga. Clasificación binaria clase R1, alimentación mediante convertidor. Izquierda: grado de carga 1. Centro: grado de carga 2. Derecha: grados de carga 1 y 2.

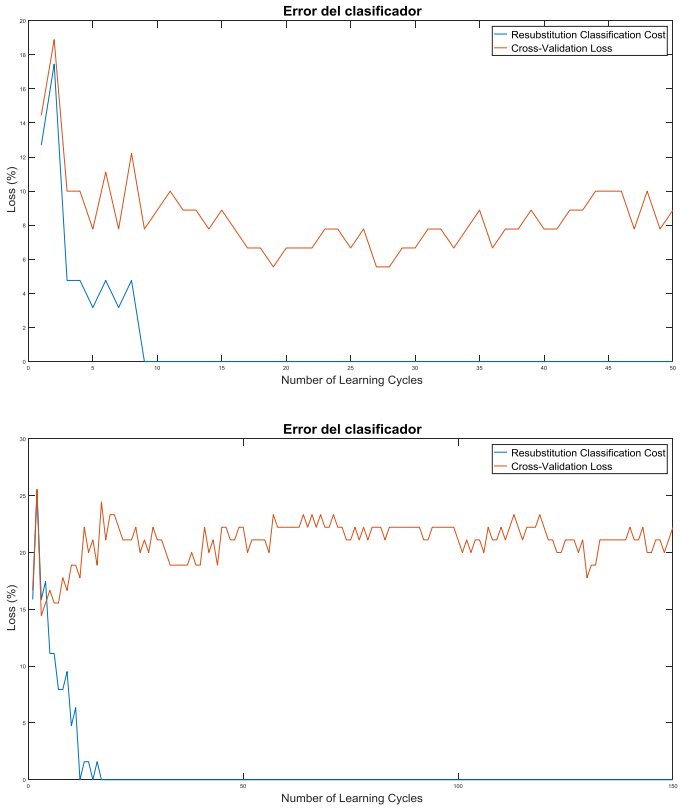


Figura 11.7: Comparación de los errores al variar el grado de carga. Clasificación binaria clase R1, alimentación mediante convertidor. Arriba: grado de carga 1. Abajo: grado de carga 2.

11.2.2.3 Conclusiones

En líneas generales los mejores resultados y menores errores se obtienen para los casos de alimentación de Red y nivel de carga 1. Por el contrario, los casos de alimentación mediante convertidor y de nivel de carga 2 presentan los peores resultados.

Sorprendentemente los casos mixtos presentan mejores resultados (obviando que necesitan más iteraciones para ajustar el modelo del clasificador) que los más desfavorables, entendiéndose que se palia en cierta medida el comportamiento de estos al incorporar los

ensayos de alimentación de Red y de nivel de carga 1.

Un hecho a destacar es que a priori se esperaba una mejor clasificación en el caso del grado de carga 2 (plena carga), ya que el método de análisis de la corriente de alimentación del motor con el que se obtuvieron los predictores USH y LSH presenta mejores resultados con grados de carga altos. Se ha observado que esto no es así, a lo cual se encuentran dos posibles explicaciones: que no hay demasiada diferencia en estos dos predictores según los grados de carga utilizados y/o que el uso del resto de estimadores, obtenidos del dominio del tiempo, camufla cualquier posible mejoría, siempre refiriéndonos al caso particular de esta clase, ya que en otras se siguen comportamientos completamente opuestos.

11.2.3 Clasificador Binario para clase R2

Presenta un comportamiento similar al caso del clasificador binario para R1, si bien todos los resultados que se consiguen son peores que en ése caso, tanto en error absoluto (figura 11.9) como en precisión del caso que nos ocupa (figura 11.9).

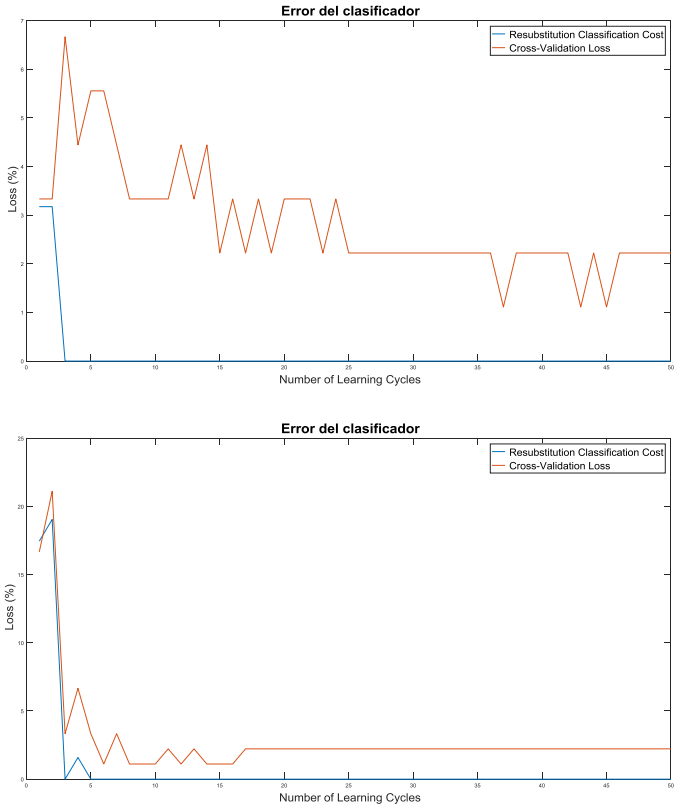


Figura 11.8: Comparación del error de clasificación. Grado de carga 1 y alimentación de Red. Arriba: clasificación clase R1. Abajo: clasificación clase R2.

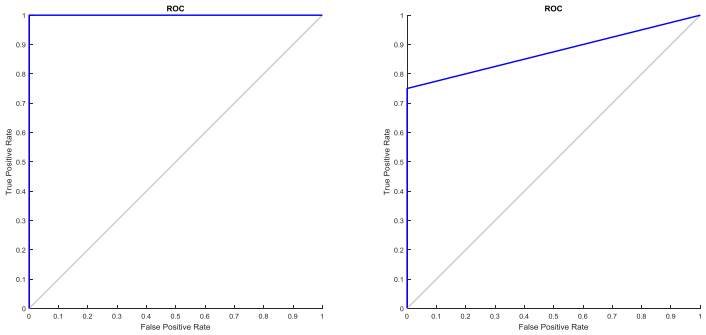


Figura 11.9: Comparación de la precisión de clasificación. Grado de carga 1 y alimentación de Red. Izquierda: clasificación clase R1. Derecha: clasificación clase R2.

11.2.4 Clasificador Binario para claseR3

La clasificación de la clase R3 presenta mejores resultados que las clasificaciones de R1 y R2, tanto por precisión de este ensayo concreto como por el valor absoluto y comportamiento del mismo (figuras 11.10 y 11.11).

Hay que destacar que no sigue el comportamiento de las clases R1 y R2, ya que los mejores resultados se suelen obtener con el nivel de carga 2. En la mayoría de los casos el error medio conseguido tiene un valor del orden del 2%.

Como las curvas de los errores medios siguen un comportamiento muy constante, se han representado en muchos casos 150 iteraciones para observar el grado de variabilidad de las mismas, no habiéndose encontrado.

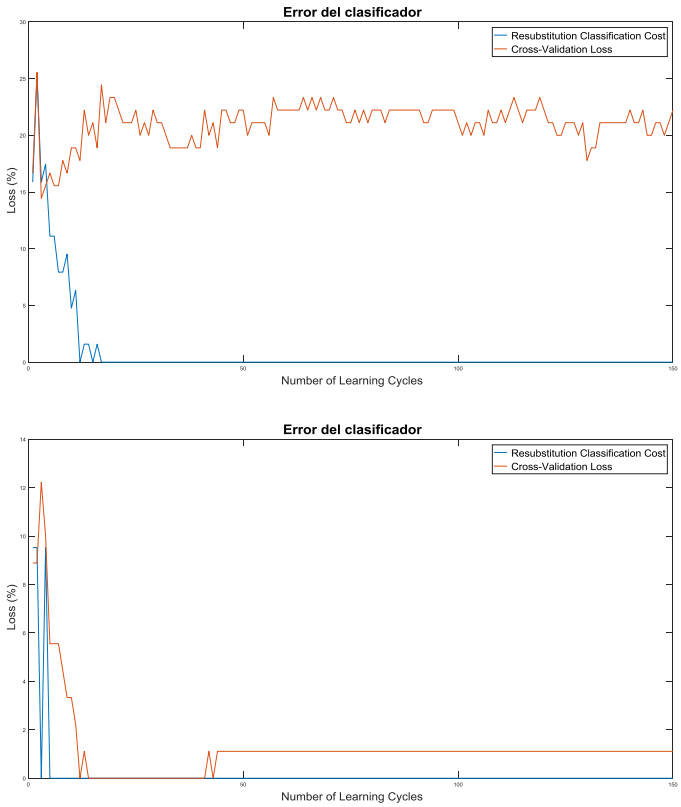


Figura 11.10: Comparación del error de clasificación. Grado de carga 1 y alimentación mediante convertidor. Arriba: clasificación clase R1. Abajo: clasificación clase R3.

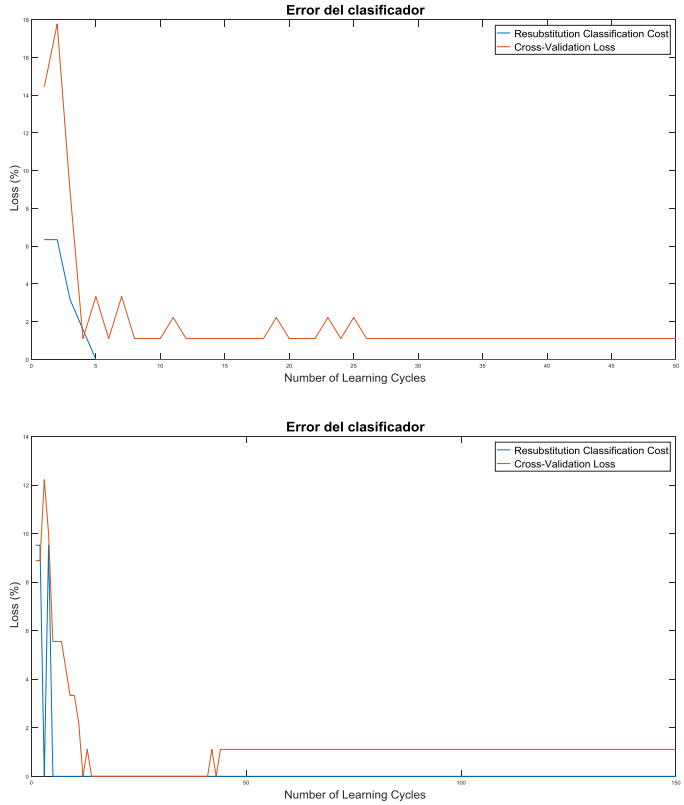
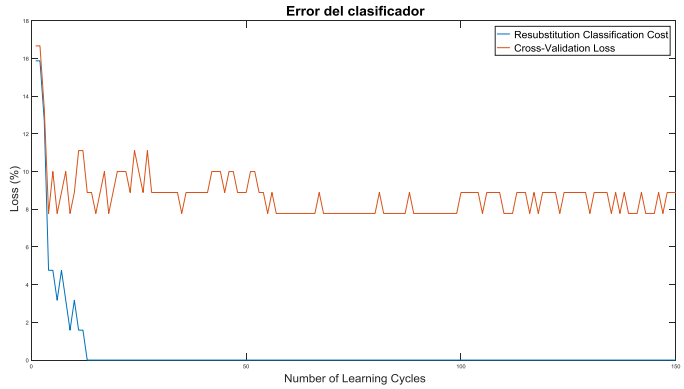


Figura 11.11: Comparación del error de clasificación. Clasificación clase R3. Grado de carga 2. Arriba: alimentación de Red. Abajo: alimentación mediante convertidor.

11.2.5 Clasificador Binario para clase R4

Se observa como la clasificación de esta clase presenta bajos errores, en algunos casos nulos al cabo de un determinado número de iteraciones del algoritmo de entrenamiento del clasificador. La clasificación de ensayos pertenecientes a alimentación mediante convertidor suele arrojar mejores resultados y menores errores que mediante RED (figura 11.12).



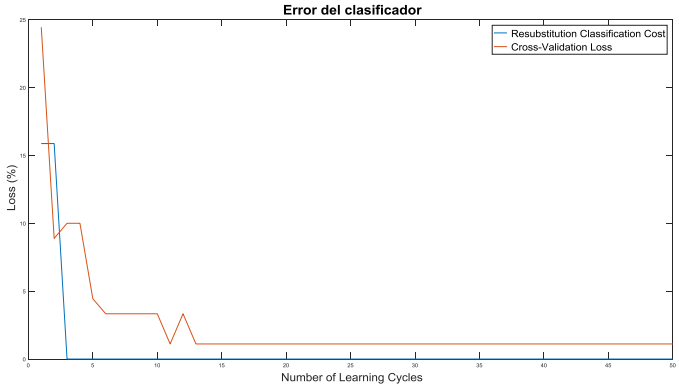


Figura 11.12: Error de clasificación. Clasificación clase R4. Grado de carga 1. Arriba: alimentación de Red. Abajo: alimentación mediante convertidor.

Asimismo, se obtienen resultados más positivos cuando se trabaja con grado de carga 2, siendo los casos con grado de carga 1 los que peores resultados obtienen (figura 11.13).

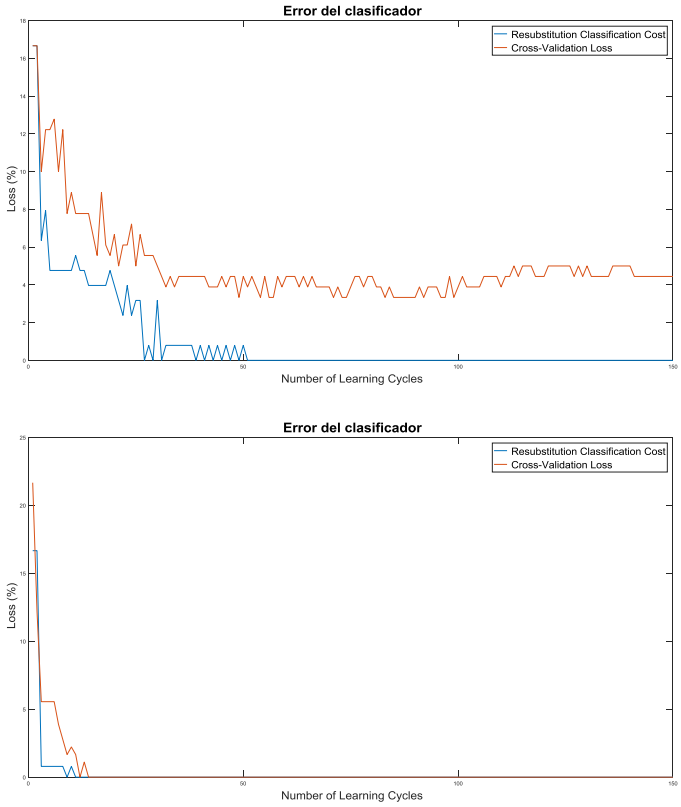


Figura 11.13: Error de clasificación. Clasificación clase R4. Cualquier tipo de alimentación. Arriba: grado de carga 1. Abajo: grado de carga 2.

11.2.6 Clasificador Binario para clase R5

Es la clasificación que mejores resultados arroja: en todos los casos el primer error del clasificador durante su entrenamiento fue nulo, por lo que se han utilizado siempre clasificadores basados en árboles de decisión, con resultados de clasificación prácticamente perfectos. Además, el error medio calculado presenta siempre valores muy reducidos, del orden de 0-1%.

Los mejores resultados se obtienen en los ensayos de grado de carga 2, sin encontrarse apenas influencia del tipo de alimentación utilizada.

11.3 Clasificador Multiclase mediante AdaBoostM2

11.3.1 Introducción

Al igual que en el caso de clasificación binara, se ha utilizado en todos los casos una semilla de generación de número aleatorio igual a 1, de cara a permitir la reproducción de los ensayos realizados.

El número de iteraciones se ha tomado igualmente de 50. Sin embargo, para estudiar con más detalle el error de validación cruzada se trabajará en algún caso con 150 iteraciones, indicándose cuando sea así, y siempre habiéndose comprobado que no afecte a la precisión del clasificador.

El número de subconjuntos (folds) tomado a la hora de realizar la validación cruzada también es de 20.

11.3.2 Resultados al variar la alimentación

Con la alimentación mediante variador de frecuencia el clasificador es incapaz de clasificar correctamente la clase 2, sea cual sea la cantidad de iteraciones realizadas al entrenar el clasificador, clasificándola siempre como clase 1 (figura 11.14). Esto también ocurre cuando se trabaja con ambos tipos de alimentación simultáneamente.

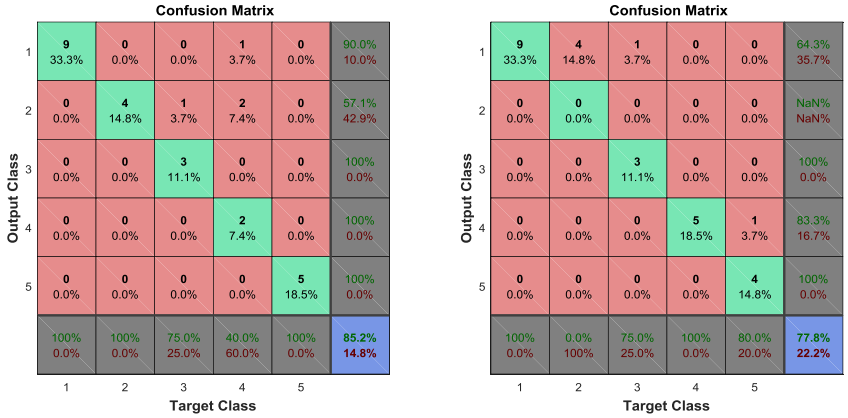


Figura 11.14: Clasificación multiclase. Grado de carga 1. Izquierda: alimentación de Red. Derecha: alimentación mediante convertidor.

Los errores son cercanos del doble en el caso de alimentación mediante convertidor que en el caso de alimentación por Red, observándose además que la forma de la curva del error es mucho más escalonada en el primer caso (figura 11.15).

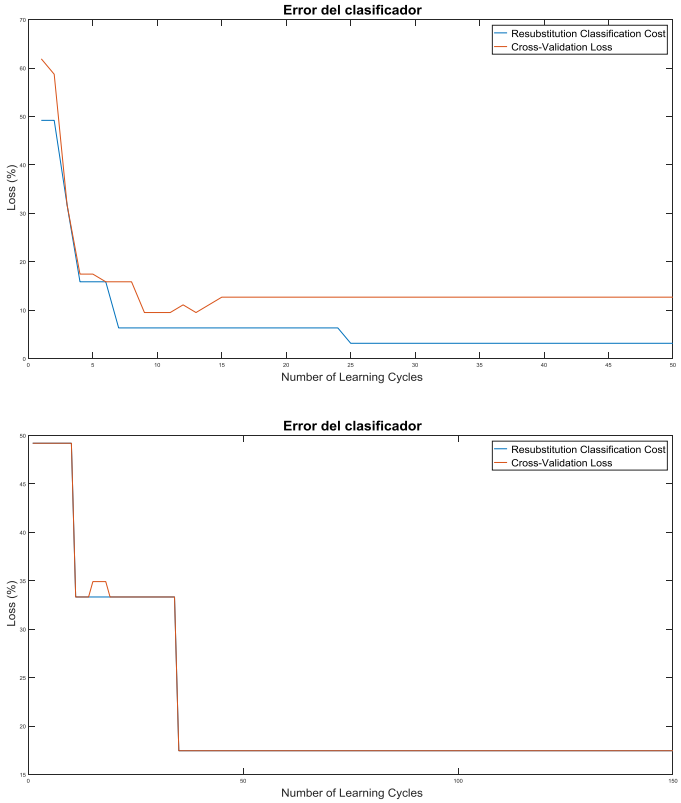


Figura 11.15: Error de clasificación. Clasificación multiclase. Grado de carga 1. Arriba: aumentación de Red. Abajo: alimentación mediante convertidor.

11.3.3 Resultados al variar el grado de carga

El error medio es mucho menor en el caso del grado de carga 2, y cuando se trabaja con cualquier nivel de carga los resultados se aproximan a los del nivel de carga 2, con excepción del caso de alimentación por Red, cuyos resultados de clasificación al trabajar con cualquier nivel de carga son mucho peores que los del cada nivel de carga por separado, ya que se clasifican los ensayos pertenecientes a la clase R2 como si fueran de clase R1 (figura 11.16).

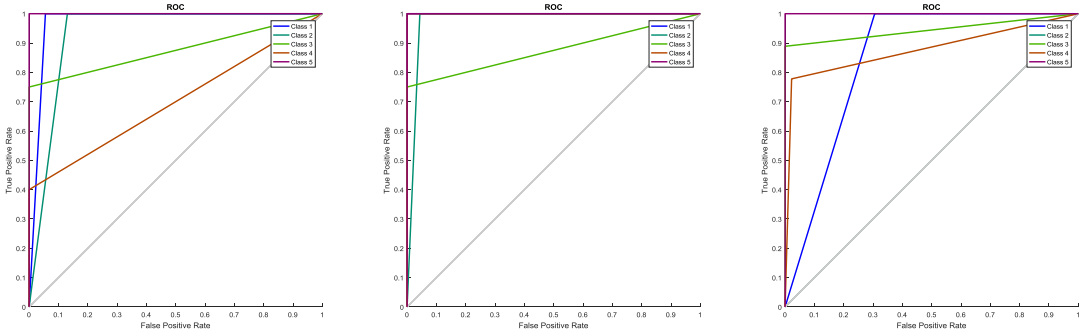


Figura 11.16: Clasificación multiclase. Alimentación de Red. Izquierda: grado de carga 1. Centro: grado de carga 2. Derecha: cualquier grado de carga.

11.3.4 Conclusiones

Se observa como en ningún caso el modelo se ajusta 100% a la muestra de entrenamiento, a diferencia de lo que ocurría en el caso de los clasificadores binarios. Además, el error que presenta este método de clasificación es notablemente superior al de la clasificación binaria.

Los errores más habituales de clasificación son clasificar como clase R1 muestras pertenecientes a la clase R2, lo cual no es de extrañar teniendo en cuenta que se trata de un fallo incipiente. Las clases R3 y R4 también presentan en ocasiones una mala clasificación, generalmente como clase R1. La clase R5 rara vez se clasifica erróneamente.

11.4 Métodos de clasificación multiclase

La variación de los resultados en función del nivel de carga y del tipo de alimentación ya se ha recogido en los apartados anteriores, por lo que de cara a evaluar cada uno de los métodos de clasificación propuestos se trabaja con un nivel de carga y de alimentación dados (Tabla 11.2).

Tabla 11.2: Tipo de alimentación y nivel de carga elegidos para los ensayos de los métodos.

Tipo de alimentación	Red
Nivel de carga	1

Además, se trabajará con los parámetros de número de clasificadores débiles y folds para validación cruzada elegidos anteriormente para cada clasificador.

Esto se ha hecho así debido a que la evaluación de los distintos métodos, para distintos valores de alimentación y de carga, así como repitiendo los ensayos para distintos conjuntos de test y de entrenamiento exigiría una cantidad de combinaciones enorme, no aportándose nada nuevo ya que el comportamiento de los clasificadores utilizados en los distintos métodos ya se ha analizado, y sólo queda comparar cuál de los métodos de clasificación multiclase creados tiene un mejor funcionamiento.

Para realizar los ensayos se utilizarán distintos valores de semilla de generación de números aleatorios (tabla 11.3). Finalmente se comparan los resultados de los diferentes métodos, fundamentalmente las matrices de confusión obtenidas, para establecer cuáles son los más indicados.

Tabla 11.3: Valores de semilla de generación de números aleatorios elegidos para los ensayos de los métodos.

Nº Ensayo	Valor de la semilla
Ensayo 1	1
Ensayo 2	7
Ensayo 3	50
Ensayo 4	102
Ensayo 5	320

En todos los casos se analizan los errores de clasificación y de validación cruzada de los clasificadores individuales. El error total de cada método vendrá dado por estos errores y por cómo sea el algoritmo en cada caso.

11.4.1 Apreciaciones al método 1

Hay dos maneras posibles de entrenar los clasificadores: entrenamiento mediante la muestra total de entrenamiento o entrenamiento mediante una muestra que sólo englobe aquellas clases que, en caso del buen funcionamiento del algoritmo en cascada, vaya a tener que analizar a analizar (Ej: el clasificador R2vsR3R4R5 se entrenaría para clasificar la clase R2 en una muestra de clases R3, R4 y R5).

En la figuras 11.17 y 11.18 se observan los resultados al realizar un entrenamiento con muestra completa o con muestra reducida en la totalidad de los clasificadores, deduciéndose que en la mayoría de los casos es más aconsejable la clasificación mediante muestra completa, exceptuando en el caso del clasificador R4vsR5.

Por tanto, la configuración más adecuada y para la que se han obtenido los resultados es para el entrenamiento reducido del clasificador R4vsR5 (se entra únicamente para clasificar la clase R4 a partir de una muestra de clases R4 y R5) y el entrenamiento completo para el resto de clasificadores (se entran para clasificar la clase correspondiente en una muestra que contenga el resto de clases).

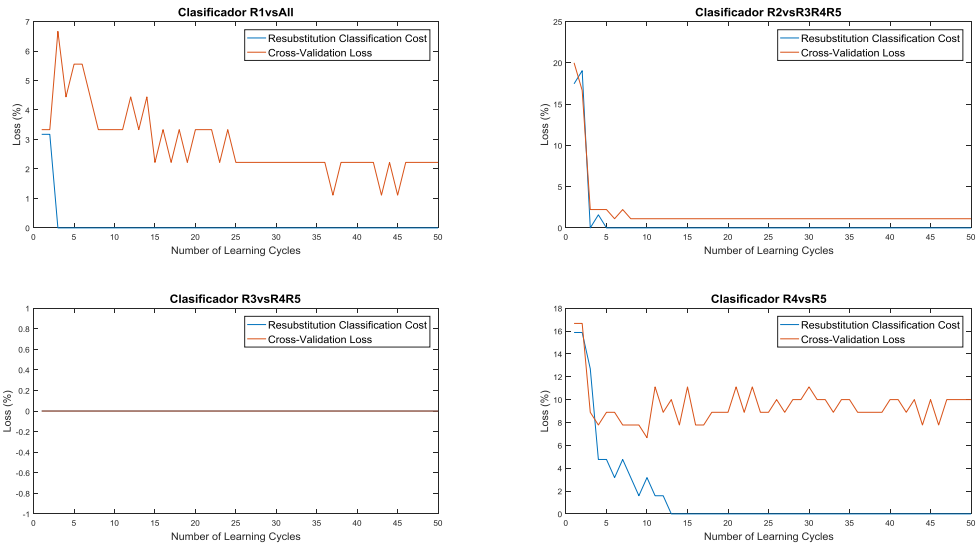


Figura 11.17: Errores de clasificación en el método de clasificación 1. Semilla de valor 1, alimentación Red, grado de carga 1. Entrenamiento de clasificadores mediante muestras completas.

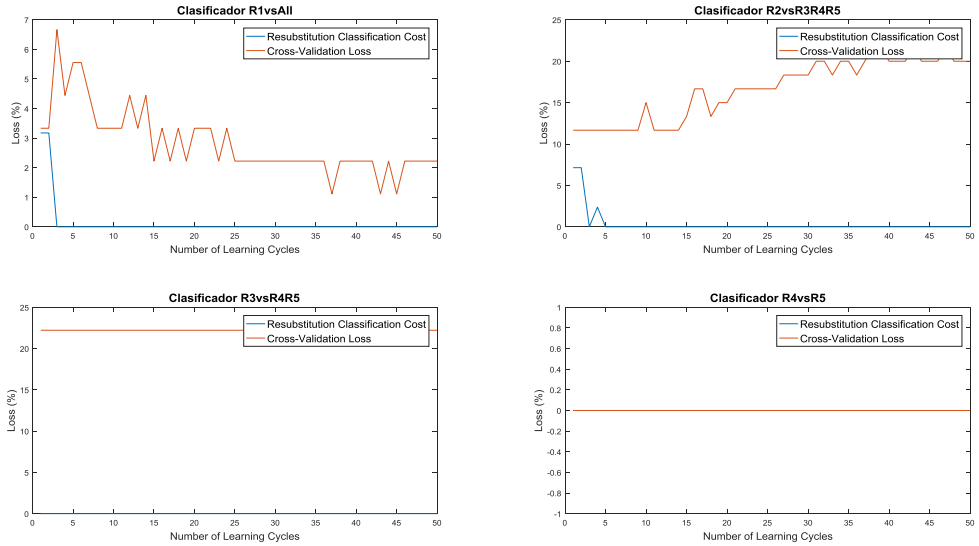


Figura 11.18: Errores de clasificación en el método de clasificación 1. Semilla de valor 1, alimentación Red, grado de carga 1. Entrenamiento de clasificadores mediante muestras reducidas.

11.4.2 Apreciaciones al método 2

Como ya se ha comentado, se trata del caso de la utilización de la función de clasificación mediante el método *AdaBoostM2*. Se observan las mismas limitaciones a la hora de clasificar las distintas clases relatadas anteriormente.

11.4.3 Apreciaciones al método 3

Al igual que ocurría en el método 1, hay dos maneras posibles de entrenar el clasificador multiclase: entrenamiento mediante la muestra total de entrenamiento o entrenamiento mediante una muestra que sólo englobe aquellos casos que se van a analizar (que no contemplarían a la clase R1). Se ha observado que es más conveniente la clasificación mediante muestras completas (figura 11.19 y 11.20).

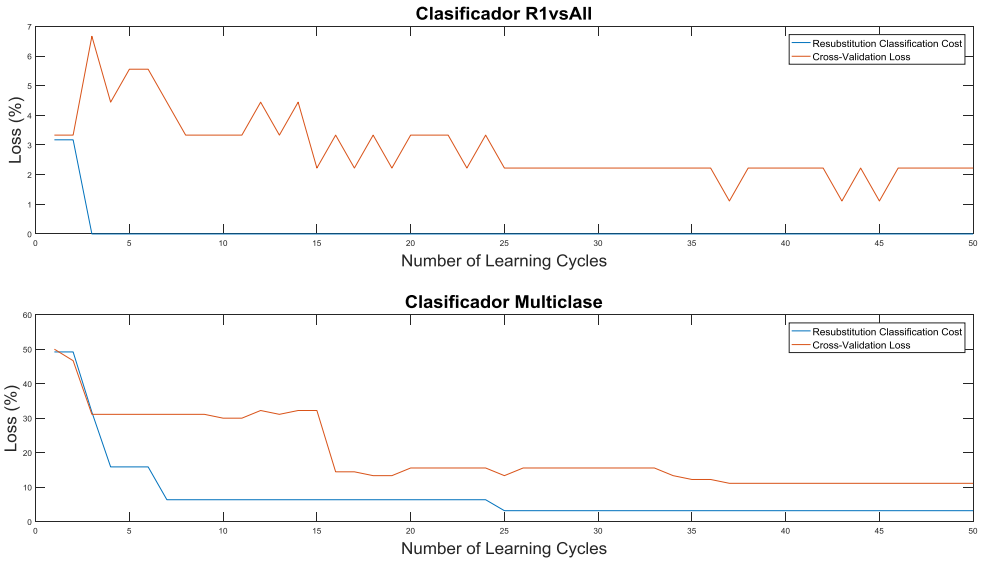


Figura 11.19: Errores de clasificación en el método de clasificación 3. Semilla de valor 1, alimentación Red, grado de carga 1. Entrenamiento del clasificador multiclase mediante muestra completa.

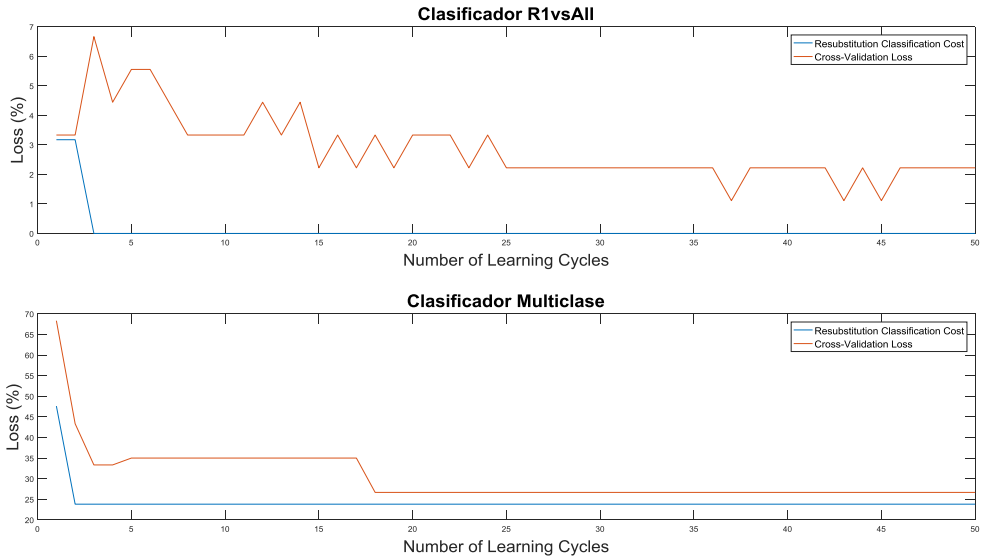


Figura 11.20: Errores de clasificación en el método de clasificación 3. Semilla de valor 1, alimentación Red, grado de carga 1. Entrenamiento del clasificador multiclase mediante muestra reducida.

11.4.4 Apreciaciones al método 4

Ocurre exactamente lo mismo que en el método 3, con la adicción a mayores de la posibilidad de entrenar mediante muestra total o parcial al clasificador encargado de detectar la clase R2. Como era de esperar a tenor de los resultados obtenidos para los métodos 1 y 3, los mejores resultados se obtienen del entrenamiento mediante muestras completas.

11.4.5 Comparación de los métodos

El método 2 presenta unas precisiones en los ensayos realizados del orden del 90%, con un error medio es del orden del 12%. Sin embargo, presenta un elevado error de clasificación habitual para la clase R4, así como para las clases R2 y R3. Esto repercutía en los casos predichos como pertenecientes a la clase R1, lo cual es algo bastante negativo según los objetivos que se han planteado, ya que R1 es un motor sano y R4 presenta un fallo en barra de rotor bastante acusado.

El método 3 presenta una precisión muy variable, con unos resultados dependientes en gran medida del clasificador multiclase utilizado. Realmente no se consigue apenas mejora respecto al método 2, lo cual es negativo ya que el coste computacional es mayor en este caso.

El método 4 presenta unos resultados exactos a los obtenidos en el método 3, lo cual nos indica que la inclusión de un clasificador destinado a la clase R2 no nos añade ningún beneficio, resultando más caro desde el punto de vista del procesamiento del algoritmo.

El método 5 presenta unos resultados muy buenos comparándolos con los métodos 2, 3 y 4, alcanzando precisiones del orden del 100% en algún caso de los analizados. Pero igualmente que en anteriores casos, el método tiene tendencia a clasificar erróneamente la clase R4, lo cual es un problema serio en el caso que se trata. Si bien este método a priori resultaba prometedor, los resultados obtenidos no los son tanto, aún más si se tiene en cuenta el coste computacional de entrenar 6 clasificadores diferentes, con sus respectivas validaciones cruzadas.

Como ya se indicó anteriormente, se creó el método 6 con el objeto de solventar la tendencia del clasificar erróneamente la clase R4, prediciéndola en muchos casos como clase R1. Se ha conseguido este objetivo en gran medida, como se puede observar en la figura 11.21, si bien es un error que se sigue cometiendo. Como ya se indicó anteriormente, el método 5 tiene tendencia a clasificar algunas muestras pertenecientes a la clase R4 como clase R1, lo cual es bastante serio teniendo en cuenta que R1 es un motor sano y R4 presenta un fallo en barra de rotor bastante acusado.

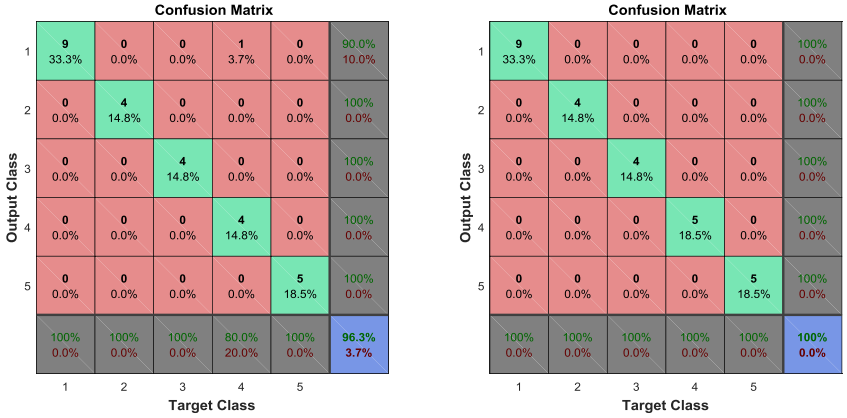


Figura 11.21: Comparación de resultado de los métodos 5 y 6. Valor de semilla 1, alimentación Red y grado de carga 1. Izquierda: método 5. Derecha: método 6.

Analizando los resultados obtenidos se ve claramente que los métodos 1 y 6 presentan de lejos los mejores resultados, con precisiones en algunos casos del 100%, siendo ligeramente mayor la precisión del método 6 para algunos casos, mientras que en la mayoría de las situaciones es idéntica.

Esto se ha observado al estudiar el comportamiento de estos dos algoritmos para diferentes valores de alimentación y carga posibles, observándose los mismos comportamientos (ya descritos en apartados anteriores, al analizar los clasificadores individualmente):

- Los peores resultados se tienen al analizar las muestras provenientes de una alimentación mediante convertidor.
- Las mejores mediciones se dan de media en el caso de alimentación de Red y de grado de carga 1.

A mayores de todo esto, suelen ser las mismas clases las que presentan problemas de predicción:

- Clase R1 clasificada como R2 o R4.
- Clase R2 clasificada como R1 o R4.
- Clase R4 clasificada como R1.

El principal problema radica en predecir situaciones de fallo (R2 y R4) como situaciones de motor sano, más que la situación inversa.

A tenor de los resultados, por mucho que se intenten combinar y ajustar los distintos métodos AdaBoost proporcionados por MATLAB se seguirán presentando los mismos problemas. Si bien se podría intentar ajustar los modelos de los clasificadores a la perfección mediante

métodos de minimización de errores, es algo que habría que realizar para cada combinación de alimentación y de grado de cargo, además que, tal y como se vio al estudiar el número de clasificadores débiles a utilizar, el método AdaBoost de MATLAB presenta limitaciones para realizar la correcta clasificación que se trata en esta memoria, y existiría el riesgo de sobreajuste de los modelos.

11.5 Mejora de los métodos propuestos

A raíz de los resultados no demasiado positivos obtenidos para los casos de clasificación para cualquier tipo de alimentación y cualquier tipo de carga se ha procedido a modificar el código, empezando por permitir la variación de la tasa de aprendizaje. Inicialmente se trabajaba con una tasa de valor unidad, por lo que se ha decidido utilizar una de valor de 0,1.

Se han elegido los métodos 1 y 2 para probar esta modificación, porque el primero es uno de los que mejor resultados ha proporcionado y el segundo porque utiliza el método *AdaBoostM2*, que es el método AdaBoost utilizado por MATLAB para la clasificación multiclase, es decir, es un método diseñado específicamente para la clasificación multiclase, por lo que es esperable que tenga unos resultados más acertados de los obtenidos hasta ahora.

En las figuras 11.22 y 11.23 se recogen los resultados obtenidos para la tasa de aprendizaje anteriormente utilizada (1) y la actual (0,1) utilizando el método 2, viéndose que el error de clasificación empeora considerablemente, mientras que el error medio alcanza un valor similar, pero necesitando más clasificadores para ello.

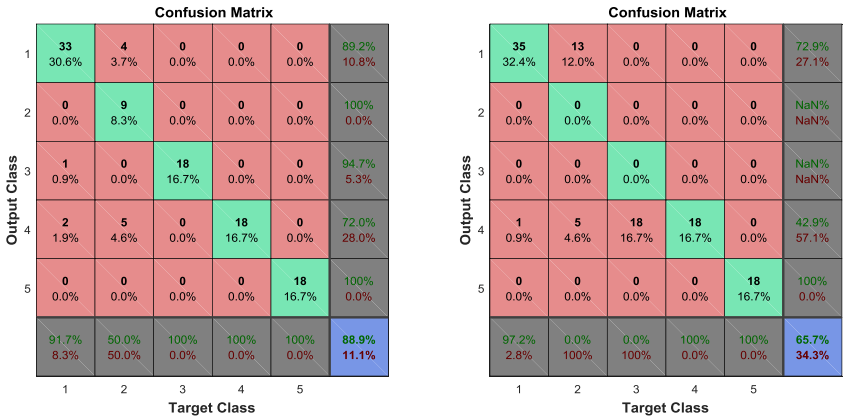


Figura 11.22: Comparación de resultado para el método 2. Valor de semilla 320, cualquier alimentación, cualquier nivel de carga. Izquierda: tasa de aprendizaje de valor 1. Derecha: tasa de aprendizaje de valor 0,1.

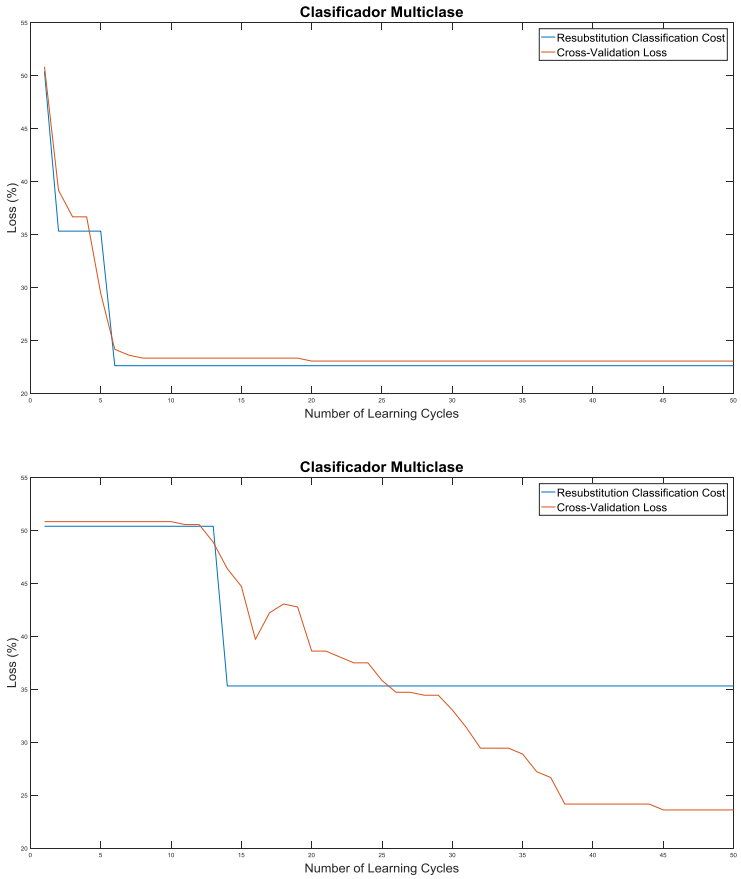
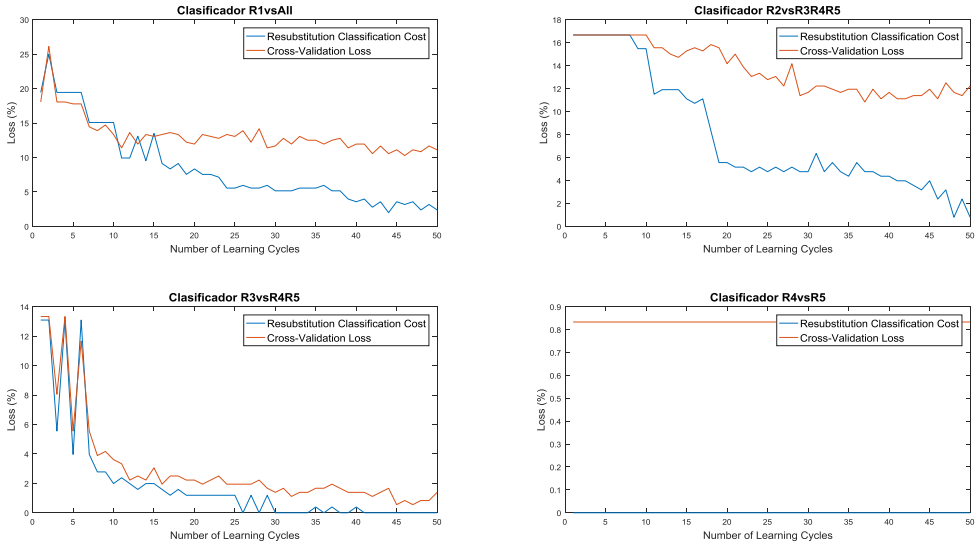


Figura 11.23: Comparación del error medio para el método 2. Valor de semilla 320, cualquier alimentación, cualquier nivel de carga. Arriba: tasa de aprendizaje de valor 1. Abajo: tasa de aprendizaje de valor 0,1.

En el caso del método 1, utilizar la tasa de aprendizaje de 0,1 con la configuración actual del clasificador empeora los resultados obtenidos para el ensayo actual. El error medio aumenta ligeramente para el clasificador R2vsR3R4R5 y notablemente para el clasificador R3vsR4R5, tal y como se recoge en la figura 11.24.



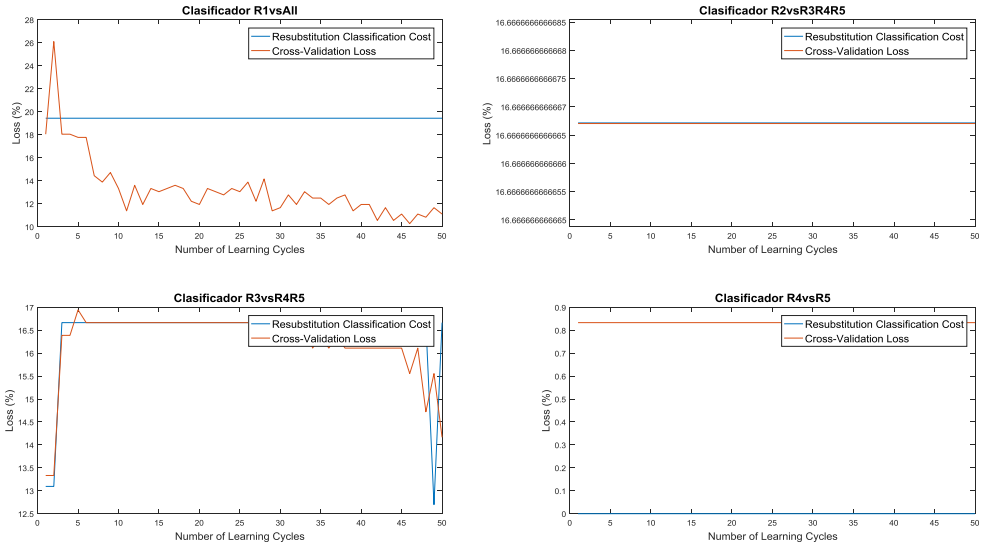


Figura 11.24: Comparación del error medio para el método 1. Valor de semilla 1, cualquier alimentación, cualquier nivel de carga. Arriba: tasa de aprendizaje de valor 1. Abajo: tasa de aprendizaje de valor 0,1.

En resumen, los resultados obtenidos manteniendo la configuración de los clasificadores utilizados a excepción de la tasa de aprendizaje, cuyo valor se ha fijado en 0,1 (por defecto es 1) empeora los resultados de clasificación, clasificando las clases R2 y R3 como R1 con mayor frecuencia.

Los resultados obtenidos para la tasa de aprendizaje nos indican que la limitación encontrada a la hora de clasificar reside en el algoritmo de aprendizaje: se ha utilizado el algoritmo que trae la función *fitensemble* por defecto, que permite trabajar con árboles de una única escisión. La solución radica en crear un algoritmo de aprendizaje mediante las funciones de MATLAB *templateTree*, *templateKNN*, o*templateDiscriminant*.

Para confirmar esta suposición se ha utilizado un algoritmo de aprendizaje en árbol, como los que veníamos utilizando. Para ello se ha recurrido a la siguiente función:

```
t=templateTree('MaxNumSplits',MaxNumSplits,'Surrogate','on');
```

Que será introducida en la llamada de la propia función *fitensemble*:

```
Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,'AdaBoostM2',
NLearn_Multi,t,'LearnRate',LearnRate);
```

Los resultados obtenidos para el método *AdaBoostM2*se recogen en las figuras 11.25 y 11.26, donde se observa una mejora extraordinaria, pasando de un error medio del 23% a uno del 8%. Además se solventa el problema de la incapacidad mostrada por este método para clasificar correctamente la clase R2 se han solventado en gran medida.

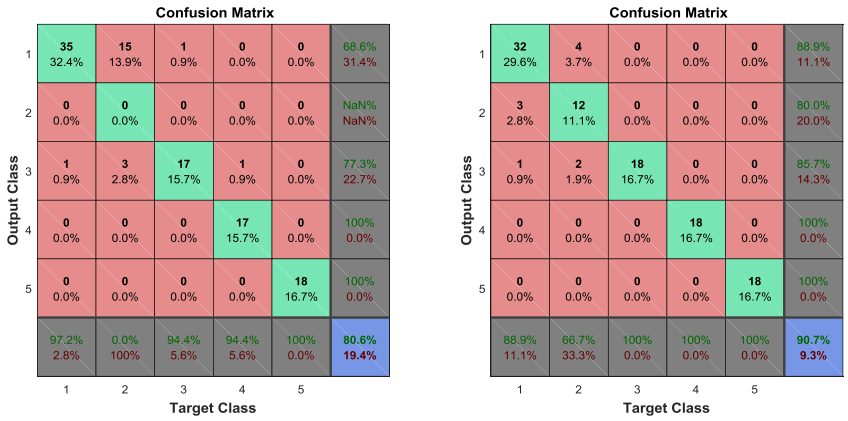


Figura 11.25 Comparación de resultado para el método 2. Valor de semilla 1, tasa de aprendizaje de valor 1, cualquier alimentación, cualquier nivel de carga. Izquierda: número de escisiones máximas=1. Derecha: número de escisiones máximas=30.

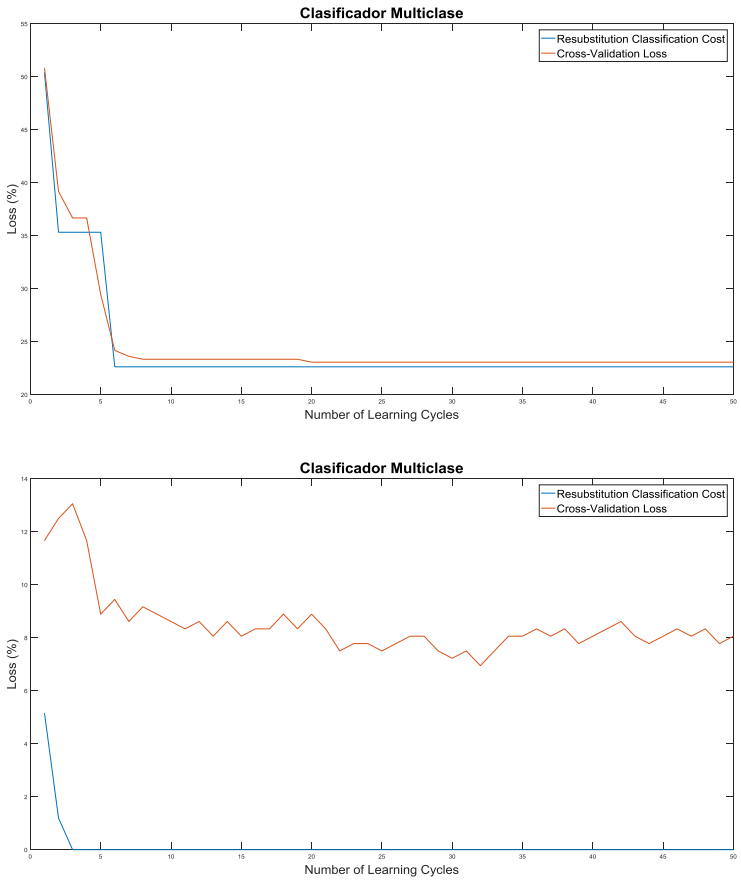


Figura 11.26: Comparación del error medio para el método 2. Valor de semilla 1, cualquier alimentación, cualquier nivel de carga. Arriba: número de escisiones máximas=1. Abajo: número de escisiones máximas=30.

La dificultad de utilizar este método de manera correcta es trabajar con unos parámetros del modelo del clasificador que arrojen los mejores resultados para la situación requerida, por lo que la forma correcta de abordar este problema en el futuro será optimizar los clasificadores utilizados para el caso concreto tratado, evitando el problema de sobreajuste que se deriva del uso de excesivos niveles en los árboles de decisión utilizados.

Por esta razón se ha creado el script *Metodos_Clasif_Multi_AdaBoost_Mejorado*, donde se han incluido en los parámetros configurables la tasa de aprendizaje y el número de divisiones

máximas a realizar para el entrenador de los clasificadores de los distintos métodos. El script sigue el mismo esquema de funcionamiento que el script *Metodos_Clasif_Multi_AdaBoost*, con la salvedad de que los errores medios y de resustitución de los clasificadores se muestran mediante diagramas de barras (figura 11.27).

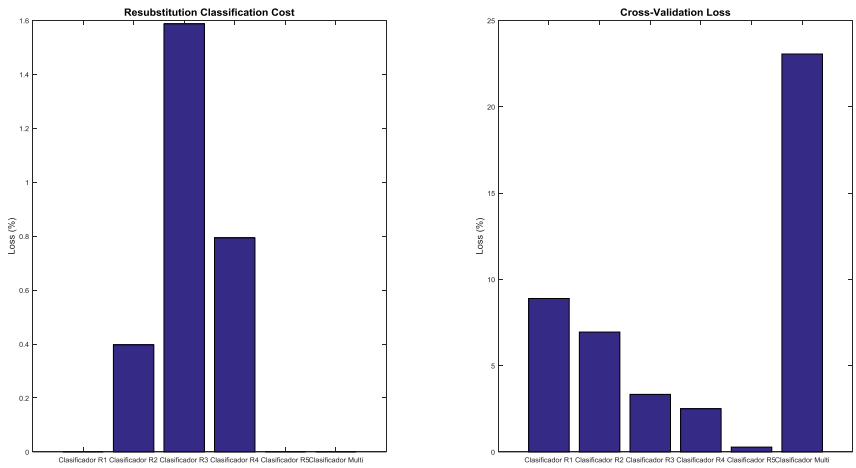


Figura 11.27: Nueva visualización de los errores de resustitución y de validación cruzada en el script *'Metodos_Clasif_Multi_AdaBoost_Mejorado'*.

Esto se ha tenido que hacer debido a que al trabajar con árboles de decisión de más niveles el algoritmo AdaBoost puede crear menos clasificadores débiles que el número de iteraciones realizadas, porque representar dichos errores de la forma anterior llevaría a error. De esta manera se muestra el error final conseguido por los clasificadores, y el error final medio estimado mediante validación cruzada.

Mediante el nuevo script se consiguen resultados mucho mayores que anteriormente, con especial relevancia del método 2, como se indicó anteriormente. El menor tiempo de cálculo de este método frente al resto hace sea éste el preferible cuando se desee menor coste computacional, ya que la diferencia de resultados es muy reducida (figura 11.28).

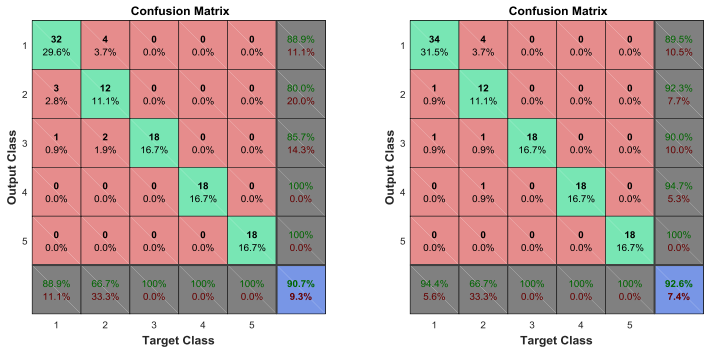


Figura 11.28 Comparación de resultados para métodos 2 y 5. Valor de semilla 320, tasa de aprendizaje de valor 1, 20 escisiones máximas, 50 iteraciones, 20 folds para validación cruzada, cualquier alimentación, cualquier nivel de carga. Izquierda: método 2. Derecha: método 5.

En la figura anterior se ve como los principales errores consisten en clasificar muestras R2 como R3 o R1, y muestras R1 como R2 o R3. El segundo caso es el menos preocupante, ya que únicamente implicaría realizar costes de mantenimiento innecesarios, mientras que el primero implicaría no realizar o programar tareas de mantenimiento necesarias. Sin embargo,

como la clasificación de clases R4 y R5 es prácticamente perfecta (error medio menor al 5%, figura 11.29) en la inmensa mayoría de los casos se detectaría el fallo en etapas posteriores.

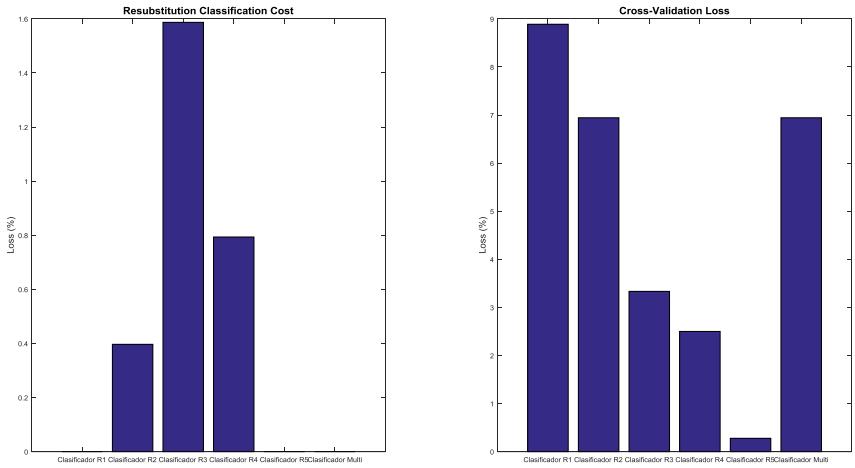


Figura 11.29 Error de resustitución y error generalizado un ensayo mediante el método 5. Valor de semilla 320, tasa de aprendizaje de valor 1, 20 escisiones máximas, 50 iteraciones, 20 folds para validación cruzada, cualquier alimentación, cualquier nivel de carga.

Por tanto, queda patente de que los métodos AdaBoost en general, y *AdaBoostM2* en particular debido a su rapidez y sencillez, permiten una predicción muy satisfactoria de las clases tratadas en esta memoria, bajo el uso de los predictores con los que se ha tratado.

CAPÍTULO 12

CONCLUSIONES

12 CONCLUSIONES

12.1 Introducción

En este capítulo se hará un breve repaso del proceso de análisis hasta la obtención de los resultados, incidiendo sobre todo en los métodos aplicados. Además, se resumirá en términos generales las conclusiones obtenidas en el análisis de los motores de inducción, y por último, se indicarán las posibles líneas de investigación que ya se mostraron en el cuarto capítulo.

12.2 Resumen de resultados obtenidos

A lo largo de esta memoria se ha tratado de realizar la correcta clasificación de muestras provenientes del análisis de la corriente de alimentación de un motor de inducción. Estas muestras corresponden a distintos ensayos realizados sobre un motor con un grado de avería en las barras del rotor, de tal manera que habría un total de 5 estados o clases a clasificar, desde el motor con rotor sano hasta el caso de rotor muy dañado.

Para realizar la clasificación se ha recurrido al software comercial MATLAB, mediante el uso de los métodos AdaBoost que proporciona, de cara a analizar el potencial de los mismos en el caso que se trata.

De cara a validar éste método se ha procedido a realizar la clasificación de las distintas clases, analizando cómo influye el grado de carga y el tipo de alimentación con el que se realizaron los ensayos del motor a esta clasificación. Los métodos de validación han sido 4: validación cruzada, error de resustitución, matriz de confusión y curva ROC.

Se han creado distintos scripts con el fin de realizar lo propuesto: un script para analizar la clasificación binaria mediante el método *AdaBoostM1* de cada clase, otro para analizar la clasificación multiclase mediante el método *AdaBoostM2* y otro para analizar distintos métodos de clasificación multiclase formados mediante el uso de los métodos *AdaBoostM1* y *AdaBoostM2*.

Finalmente se ha creado otro script donde se recogen los distintos métodos aplicando las mejoras encontradas durante el análisis.

12.3 Conclusiones

Si bien cada clase se comporta de manera un poco distinta, a grosso modo se han obtenido resultados muy satisfactorios (errores del 0-8% de media) de clasificación en las muestras pertenecientes a alimentación de red y nivel de carga 1 (deslizamiento del 3%), empeorándose la clasificación al utilizar alimentación mediante variador de frecuencia, tal y como era de esperar, así como al trabajar con un nivel de carga 2 (deslizamiento del 5%).

Además, se observa como las clase R5 (fallo crítico) no presenta apenas situaciones de clasificación errónea, siendo el resto de clases más sensibles, sobre todo las clases R4, R2 y R1 (es ese orden). La clasificación de una muestra correspondiente a un estado de fallo como

clase R1 es el mayor error que se puede cometer, ya que daría pie a considerar un motor con cierto grado de deterioro como un motor sano.

El ajuste de los clasificadores, por tanto, se tendría que realizar, para cada una de las combinaciones posibles de alimentación y de grado de carga, si bien estamos limitados al posible sobreajuste de los clasificadores y a no recurrir a algoritmos de mucho coste computacional, porque de ser factible el uso de grandes tiempos de cálculo hay mejores técnicas que AdaBoost.

Se ha tratado de solventar las carencias de los clasificadores "simples" que proporciona MATLAB mediante su uso combinado para obtener una clasificación multiclase, consiguiéndose este objetivo dentro de las limitaciones observadas del método, que se hacen siempre patentes.

Los mejores resultados inicialmente eran de la combinación de distintos clasificadores binarios orientados a una clase en concreto, si bien mediante la utilización de árboles de decisión con mayores niveles para el aprendizaje de los clasificadores se ha conseguido reducir drásticamente el error medio del clasificador multiclase*AdaBoostM2*, haciendo que la diferencia de precisión para casos concretos sea del orden del 2-3%, alcanzando la precisión general de todos los métodos órdenes del 90%.

De esta forma, se ha conseguido prácticamente eliminar las clasificaciones erróneas de las clases R4 y R5, siendo el error más habitual clasificar la clase R2 como R1, presentado valores de clasificaciones correctas del 66% para esta clase. Sin embargo, al tratarse de un estado de fallo incipiente era un resultado de esperar, cuya importancia se reduce aún más si cabe al considerar que de progresar el fallo éste será detectado más adelante casi con total seguridad.

12.4 Futuras líneas de investigación

El método AdaBoost ha presentado unos resultados muy buenos en los casos estudiados, por lo que habría que analizar si es posible su mejora mediante el uso de otros predictores (o bien mediante un uso selectivo de los utilizados) para las clases que más fallos presentan, o bien utilizar algún otro método de clasificación, ya sea en conjunto con AdaBoost o de forma independiente.

El uso de múltiples clasificadores se ha mostrado como una herramienta muy potente, tanto por el fácil ajuste de los modelos como por los resultados obtenidos y la sencillez del algoritmo creado, si bien requiere de un tiempo de computación alto comparándolo con el método *AdaBoostM2*.

El uso conjunto de diversos métodos de clasificación podría dar lugar a múltiples algoritmos donde mediante el análisis de errores medios, y de intervalos de confianza calculados para los diferentes clasificadores, dichos clasificadores se relacionaran para lanzar la predicción final, de forma similar a los métodos planteados en éste trabajo pero teniendo en cuenta a mayores la probabilidad de error en la predicción de los clasificadores individuales.

Otro factor interesante a tener en cuenta a la hora de realizar un algoritmo de clasificación es el coste de realizar una clasificación errónea: no tiene igual gravedad clasificar un motor sano como averiado, donde simplemente se adelantarían las labores de mantenimiento, que clasificar un motor averiado como sano, ya que de aplicar exclusivamente mantenimiento predictivo se podría producir el fallo total del motor.

De esta manera, dos situaciones que presenten un error de clasificación medio idéntico, con los resultados similares para los ensayos realizados pueden suponer situaciones y costes diametralmente opuestos, siendo preferible en ocasiones un modelo de clasificación menos ambicioso y/o con resultados medios globales peores que otro modelo que pudiese dar lugar a situaciones más catastróficas y caras.

CAPÍTULO 13

BIBLIOGRAFÍA

13 BIBLIOGRAFÍA

13.1 Libros

Fraile Mora, Jesús (2008). "*Máquinas eléctricas. 6ª edición"*. Editorial Mc Graw Hill.

Michie, D.; Spiegelhalter, D.J.; Taylor, C.C. (1994). "*Machine Learning, Neural and Statistical Classification"*.

13.2 Artículos

Freund, Yoav; Schapire, Robert E.*"A Short Introduction to Boosting"*. AT&TLabs. Journal of Japanese Society for Artificial Intelligence, 14(5):771-780, Septiembre, 1999.

Freund, Yoav; Schapire, Robert E. *"Experiments with a New Boosting Algorithm"*. AT&T Labs. Machine Learning: Proceedings of the Thirteenth International Conference, 1996.

Kotsiantis,S. B.. "*Supervised Machine Learning: A Review of Classification Techniques"*. Universidad del Peloponeso, Grecia. Julio, 2007.

Pons Llinares, Joan; Climente Alarcón, Vicente; Puche Panadero, Rubén; Antonino Daviu, Jose A. *"Bar breakage detection on Squirrel Cage Induction Motors via Transient Motor Current Signal Analysis based on the Wavelet Transform. A Review"*. Universidad Politécnica de Valencia

Reiss, Attila; Stricker, Didier; Hendeby, Gustaf. "*Confidence-based multiclass AdaBoost for physical activity monitoring"*. 2013 International Symposium on Wearable Computers.

13.3 Proyectos Final de Carrera

Rodríguez Oraá, Bárbara Mª. "*Detección de fallo de barras en motores de inducción utilizando histogramas"*. Universidad de Valladolid. 2013.

Alonso Martínez, Alberto. "*Aplicación del método MCSA en el análisis del estado de motores de inducción de una industria ferroviaria"*. Universidad de Valladolid. 2015.

Zamora Pérez, Carlos. "*Estudio de aplicación de la técnica de estimación espectral MUSIC para la detección de fallos en motores de inducción"*. Universidad de Valladolid., 2013.

13.4 Documentos

Aler Mur, Ricardo. "*Evaluación de técnicas de aprendizaje-I"*. Universidad Carlos III de Madrid.

Carmona Suárez, Enrique J. *"Tutorial sobre Máquinas de Vectores Soporte (SVM)"*. Universidad Nacional de Educación a Distancia (UNED). Última versión: Julio, 2014.

Cortizo Pérez, José Carlos. "*Evaluación de los SINAI"*. Universidad Europea de Madrid.

Meir, Ron. *"Boosting Tutorial"*. Machine Learning Summer School 2002.

Schapire, Rob. "*Machine learning algorithms for classification"*. Universidad de Princeton.

13.5 Webs

Clasificador NaïveBayes. . Último acceso: 31/08/2016, de http://naivebayes.blogspot.com.es/

Classifier evaluation with imbalanced datasets. Último acceso: 31/08/2016, de https://classeval.wordpress.com/.

Exploración de Datos: Introducción a la estadística descriptiva, Recursos Educativos para los Profesores de la Universidad Católica de Valparaíso. Último acceso: 2013, de http://www.ucv.cl/web/estadistica/index.htm.

Mathworks. "help>stats". Último acceso: 31/08/2016, de http://es.mathworks.com/help/stats/.

Manuales SKF. Tipos y diseños de rodamientos. "*Spain>Productos>Rodamientos,..> Rodamientos...>Principios para la selección y...>Nociones básicas sobre rodamientos>Tipos y diseños de rodamientos"*. Último acceso: 31/08/2016, de http://www.skf.com/es/products/bearings-units-housings/roller-bearings/principles/bearing-basics/bearing-types-and-designs/index.html.

Manzanilla, Orestes. Optimization & Machine Learning. *"Overfitting o Sobre-ajuste"*. Último acceso: 31/08/2016, de http://optimachine.blogspot.com.es/2008/03/overfitting-o-sobre-ajuste.html.

OpenCv. *"Introduction to Support Vector Machines"*. Último Acceso: 31/08/2016 de http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html.

Rodríguez, Oldemar. *"Validación Cruzada (cross-validation) y Remuestreo (bootstrapping)"*. Último acceso: 31/08/2016, de http://www.oldemarrodriguez.com/yahoo_site_admin/assets/docs/Presentaci%C3%B3n_-_CV.293124233.pdf.

Rohrer, Brandon. Microsoft Azure. *"How to choose algorithms for Microsoft Azure Machine Learning"*. Último acceso: 31/08/2016, de https://azure.microsoft.com/en-us/documentation/articles/machine-learning-algorithm-choice/.

Sancho Caparrini, Fernando. *"Introducción al Aprendizaje Automático"*. Universidad de Sevilla. Último acceso: 31/08/2016, de http://www.cs.us.es/~fsancho/?e=75.

Sayad, Saed. Naïve Bayesan. *"Map>DataMining>PredictingtheFuture>Modeling>Classification>NaiveBayesian*". Último acceso: 31/08/2016, de http://www.saedsayad.com/naive_bayesian.htm.

Tenório, Eduardo. The Men Who Stare at Codes. "*Neural networks in a nutshell"*. 2014. Último acceso: 31/08/2016, de https://themenwhostareatcodes.wordpress.com/2014/03/02/neural-networks-in-a-nutshell/comment-page-1/.

ANEXOS

1 CÓDIGO FUENTE DE LOS SCRIPTS

1.1 Clasificador Binario mediante AdaBoostM1 y Árbol de decisión

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%             TRABAJO FIN DE MASTER              %%%%%%%%%%%%
%
%     DESARROLLO DE UNA HERRAMIENTA DE DIAGNOSTICO DE FALLOS EN MOTORES
%                  DE INDUCCION MEDIANTE LA TECNICA ADABOOST
%
%                           JUAN OBREGÓN SANDOVAL
%%%%%%%%%%%%                                                 %%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%--------------------------------------------------------------------------%

%   ENTRENAMIENTO Y ENSAYO DE CLASIFICADORES BINARIOS MEDIANTE ADABOOST

%--------------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear variables;
clc;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                       INICIALIZACION DE VARIABLES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Parametros auxiliares
Num_caract=7;                   % Longitud de busqueda de strings para
                                % elegir datos acordes a Alimentacion y a
                                % Nivel_Carga

eliminar=[1 2 3 4 5 6 9 10]';   % Filas a eliminar de Datos (datos no
                                % utilizados, no son estimadores ni clases)

Reproduc=1;     % Si =1--> Se fija el valor de la semilla para
                % reproductibilidad
Seed=1;         % Inicializacion semilla

% Parametros para particionado de datos
Partition=0.3;                  % Porcentaje datos usados como test


% Parametros para seleccion de valores de medicion
Restric_Alim=1;     % Si =1--> Se trabajara con la alimentacion elegida
Alimentacion='RED'; % Se determina el tipo de alimentacion al motor elegida

Restric_Carga=1;    % Si =1--> Se trabajara con la carga elegida
Nivel_Carga=1;      % Se determina el nivel de carga de funcionamiento del
motor elegida

Clase=5;            % Se determina la clase a estudiar elegida
```

```
% Parametros del clasificador
NLearn=50;                      % Numero de ciclos clasificador
KFold=20;                       % Subconjuntos para el CV del clasificador

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                       CARGA Y PREPARADO DE DATOS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Carga de valores del excel
[Datos_Inicio, Texto_Inicio, Todo_Inicio]=xlsread('datos_ignacio.xlsx');

Datos=Datos_Inicio;
Texto=Texto_Inicio;
Todo=Todo_Inicio;

% Tratamiento de los datos segun parametros elegidos previamente
Texto(1,:)=[];              % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
Todo(1,:)=[];               % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
if Restric_Alim==1
    % Buscar valores correspondientes a alimentacion elegida
    a=find(strncmp(Alimentacion,Texto(:,2),Num_caract));
    % Matriz con valores correspondientes a la alimentacion elegida
    Datos=Datos(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Texto=Texto(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Todo=Todo(a,:);
else
end

if Restric_Carga==1
    % Buscar valores numericos correspondientes a carga elegida
    b=find(Datos(:,2)==Nivel_Carga);
    % Matriz con valores correspondientes a los de la carga elegida
    Datos=Datos(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
    Texto=Texto(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
    Todo=Todo(b,:);
else
end

Datos(:,eliminar)=[];
Texto(:,eliminar+3)=[];
Todo(:,eliminar+3)=[];

% Busqueda de las medidas de cada clase
Clases=unique(Todo(:,3));
Num_Clases=length(Clases);  % Se determina el numero de clases existentes

R1=find(strncmp('R1',Todo(:,3),Num_caract));
R2=find(strncmp('R2',Todo(:,3),Num_caract));
R3=find(strncmp('R3',Todo(:,3),Num_caract));
R4=find(strncmp('R4',Todo(:,3),Num_caract));
R5=find(strncmp('R5',Todo(:,3),Num_caract));

X=Datos;
```

```
Y_Clases=Todo(:,3);
Long_Y=length(Y_Clases);
Y=ones([Long_Y,1]);
Y(R1)=1;
Y(R2)=2;
Y(R3)=3;
Y(R4)=4;
Y(R5)=5;

% Preparamos los datos de la muestra reducida
if Reproduc==1
    rng(Seed,'twister') % Para reproducibilidad de las mediciones
else
end

cvpart=cvpartition(Y,'holdout',Partition);

X_Training=X(cvpart.training,:);
X_Test=X(cvpart.test,:);

Y_Training=Y(cvpart.training,:);
Y_Test=Y(cvpart.test,:);

Long_X_Test=length(X_Test);
Y_Binario=zeros(Num_Clases,Long_Y);

for i=1:Num_Clases
    Y_Binario(i,:)=(Y==i);
end

% Preparamos los datos test para representar la matriz de confusion
Y_Test_Mat_Conf=zeros(Num_Clases,Long_X_Test);
for i=1:Num_Clases
    Y_Test_Mat_Conf(i,:)=(Y_Test==i);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                        ENTRENAMIENTO DEL CLASIFICADOR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Error_Clasif_AdaBoostM1=0;      % Variable para detectar cuando aplicar
                                % clasificacion mediante arbol de decision

% Entrenamiento del clasificador
Y_Training_Clasif_Binario=(Y_Training==Clase);

Clasificador=fitensemble(X_Training,Y_Training_Clasif_Binario,'AdaBoostM1',
NLearn,'Tree');

if isempty(Clasificador.TrainedWeights)
    Clasificador=fitctree(X_Training,Y_Training_Clasif_Binario);
    Clasificador_Aux=fitctree(X,Y_Binario(Clase,:));
    display('Error de clasificacion 0 Clasificador')
else
    Error_Clasif_AdaBoostM1=1;
   Clasificador_Aux=fitensemble(X,Y_Binario(Clase,:),'AdaBoostM1',
   NLearn,'Tree');
end
```

```
% Cross Validation
Clasificador_CV=crossval(Clasificador_Aux,'KFold',KFold);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                CLASIFICACIÓN DE LA PARTICION DE PRUEBA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Clasificacion=predict(Clasificador,X_Test);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     ERROR DE RESISTITUCION Y ERROR DE CLASIFICACION MEDIANTE CV
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if Error_Clasif_AdaBoostM1==1
    RsLoss=resubLoss(Clasificador,'Mode','Cumulative');
    GenError=kfoldLoss(Clasificador_CV,'Mode','Cumulative');
else
    RsLoss=resubLoss(Clasificador)*ones(1,NLearn)';
    GenError=kfoldLoss(Clasificador_CV)*ones(1,NLearn)';
end
RsLoss=RsLoss*100;
GenError=GenError*100;

    figure;
    plot((1:1:NLearn),RsLoss,(1:1:NLearn),GenError);
    title('\fontsize{24} Error del clasificador');
    legend('\fontsize{18} Resubstitution Classification Cost',
    '\fontsize{18} Cross-Validation Loss');
    xlabel('\fontsize{20} Number of Learning Cycles');
    ylabel('\fontsize{20} Loss (%)');
    ylim([0 inf]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    MATRIZ DE CONFUSION Y CURVA CARACTERISTICA OPERATIVA DEL RECEPTOR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Representamos la matriz de confusion
figure
plotconfusion(Y_Test_Mat_Conf(Clase,:),Clasificacion')

% Representamos la curva caracteristica operativa del receptor
figure
plotroc(Y_Test_Mat_Conf(Clase,:),Clasificacion')
```

1.2 Clasificador Multiclase mediante AdaBoostM2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%              TRABAJO FIN DE MASTER              %%%%%%%%%%%%%
%
%     DESARROLLO DE UNA HERRAMIENTA DE DIAGNOSTICO DE FALLOS EN MOTORES
%                   DE INDUCCION MEDIANTE LA TECNICA ADABOOST
%
%                            JUAN OBREGÓN SANDOVAL
%%%%%%%%%%%%%                                                 %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%

%   ENTRENAMIENTO Y ENSAYO DE CLASIFICADOR MULTICLASE MEDIANTE ADABOOST

%-------------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear variables;
clc;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                     INICIALIZACION DE VARIABLES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Parametros auxiliares
Num_caract=7;                   % Longitud de busqueda de strings para
                                % elegir datos acordes a Alimentacion y a
                                % Nivel_Carga

eliminar=[1 2 3 4 5 6 9 10]';   % Filas a eliminar de Datos (datos no
                                % utilizados, no son estimadores ni clases)

Reproduc=1;     % Si =1--> Se fija el valor de la semilla para
                % reproductibilidad
Seed=1;         % Inicializacion semilla

% Parametros para particionado de datos
Partition=0.3;                  % Porcentaje datos usados como test

% Parametros para seleccion de valores de medicion
Restric_Alim=0;      % Si =1--> Se trabajara con la alimentacion elegida
Alimentacion='ABB1'; % Se determina el tipo de alimentacion al motor
elegida

Restric_Carga=0;    % Si =1--> Se trabajara con la carga elegida
Nivel_Carga=2;      % Se determina el nivel de carga de funcionamiento del
motor elegida

% Parametros del clasificador
NLearn=150;                      % Numero de ciclos clasificador
KFold=20;                       % Subconjuntos para el CV del clasificador
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                        CARGA Y PREPARADO DE DATOS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Carga de valores del excel
[Datos_Inicio, Texto_Inicio, Todo_Inicio]=xlsread('datos_ignacio.xlsx');

Datos=Datos_Inicio;
Texto=Texto_Inicio;
Todo=Todo_Inicio;

% Tratamiento de los datos segun parametros elegidos previamente
Texto(1,:)=[];              % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
Todo(1,:)=[];               % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
if Restric_Alim==1
    % Buscar valores correspondientes a alimentacion elegida
    a=find(strncmp(Alimentacion,Texto(:,2),Num_caract));
    % Matriz con valores correspondientes a la alimentacion elegida
    Datos=Datos(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Texto=Texto(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Todo=Todo(a,:);
else
end

if Restric_Carga==1
    % Buscar valores numericos correspondientes a carga elegida
    b=find(Datos(:,2)==Nivel_Carga);
    % Matriz con valores correspondientes a los de la carga elegida
    Datos=Datos(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
    Texto=Texto(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
    Todo=Todo(b,:);
else
end

Datos(:,eliminar)=[];
Texto(:,eliminar+3)=[];
Todo(:,eliminar+3)=[];

% Busqueda de las medidas de cada clase
Clases=unique(Todo(:,3));
Num_Clases=length(Clases);        % Se determina el numero de clases
existentes

R1=find(strncmp('R1',Todo(:,3),Num_caract));
R2=find(strncmp('R2',Todo(:,3),Num_caract));
R3=find(strncmp('R3',Todo(:,3),Num_caract));
R4=find(strncmp('R4',Todo(:,3),Num_caract));
R5=find(strncmp('R5',Todo(:,3),Num_caract));

X=Datos;
Y_Clases=Todo(:,3);
Long_Y=length(Y_Clases);

Y=ones([Long_Y,1]);
```

```
Y(R1)=1;
Y(R2)=2;
Y(R3)=3;
Y(R4)=4;
Y(R5)=5;
% Preparamos los datos de la muestra reducida
if Reproduc==1
    rng(Seed,'twister') % Para reproducibilidad de las mediciones
else
end

cvpart=cvpartition(Y,'holdout',Partition);

X_Training=X(cvpart.training,:);
X_Test=X(cvpart.test,:);

Y_Training=Y(cvpart.training,:);
Y_Test=Y(cvpart.test,:);

Long_X_Test=length(X_Test);

Y_Binario=zeros(Num_Clases,Long_Y);

for i=1:Num_Clases
    Y_Binario(i,:)=(Y==i);
end

% Preparamos los datos test para representar la matriz de confusion
Y_Test_Mat_Conf=zeros(Num_Clases,Long_X_Test);
for i=1:Num_Clases
    Y_Test_Mat_Conf(i,:)=(Y_Test==i);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                       ENTRENAMIENTO DEL CLASIFICADOR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Entrenamiento del clasificador
Clasificador=fitensemble(X_Training,Y_Training,'AdaBoostM2',NLearn,'Tree');

% Cross Validation
Clasificador_Aux=fitensemble(X,Y,'AdaBoostM2',NLearn,'Tree');
Clasificador_CV=crossval(Clasificador,'KFold',KFold);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                 CLASIFICACIÓN DE LA PARTICION DE PRUEBA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Clasificacion=predict(Clasificador,X_Test);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       ERROR DE RESISTITUCION Y ERROR DE CLASIFICACION MEDIANTE CV
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


RsLoss=100*resubLoss(Clasificador,'Mode','Cumulative');
GenError=100*kfoldLoss(Clasificador_CV,'Mode','Cumulative');
```

```
    figure;
    plot((1:1:NLearn),RsLoss,(1:1:NLearn),GenError);
    title('\fontsize{24} Error del clasificador');
    legend('\fontsize{18} Resubstitution Classification Cost',
    '\fontsize{18} Cross-Validation Loss');
    xlabel('\fontsize{20} Number of Learning Cycles');
    ylabel('\fontsize{20} Loss (%)');
    ylim([0 inf]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    MATRIZ DE CONFUSION Y CURVA CARACTERISTICA OPERATIVA DEL RECEPTOR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Preparamos los datos test para representar la matriz de confusion
Y_Clasificacion_Mat_Conf=zeros(Num_Clases,Long_X_Test);
for i=1:Num_Clases
    Y_Clasificacion_Mat_Conf(i,(Clasificacion==i))=1;
end

% Representamos la matriz de confusion
figure
plotconfusion(Y_Test_Mat_Conf,Y_Clasificacion_Mat_Conf)

% Representamos la curva caracteristica operativa del receptor
figure
plotroc(Y_Test_Mat_Conf,Y_Clasificacion_Mat_Conf(:,:))
```

1.3 Script de evaluación de métodos de clasificación

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%            TRABAJO FIN DE MASTER             %%%%%%%%%%%%%
%
%     DESARROLLO DE UNA HERRAMIENTA DE DIAGNOSTICO DE FALLOS EN MOTORES
%                   DE INDUCCION MEDIANTE LA TECNICA ADABOOST
%
%                           JUAN OBREGÓN SANDOVAL
%%%%%%%%%%%%%                                              %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%

%  ENTRENAMIENTO Y ENSAYO DE METODOS DE CLASIFICACION MULTICLASE MEDIANTE
%   COMBINACION DE CLASIFICADORES BINARIOS Y MULTICLASE ADABOOST Y ARBOL

%-------------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear variables;
clc;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      INICIALIZACION DE VARIABLES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Parametros auxiliares

Num_caract=7;                   % Longitud de busqueda de strings para
                                % elegir datos acordes a Alimentacion y a
                                % Nivel_Carga

eliminar=[1 2 3 4 5 6 9 10]';   % Filas a eliminar de Datos (datos no
                                % utilizados, no son estimadores ni clases)

Reproduc=1;     % Si =1--> Se fija el valor de la semilla para
                % reproductibilidad
Seed=320;       % Inicializacion semilla

% Parametro para particionado de datos

Partition=0.3;                  % Porcentaje datos usados como test


% Parametros para seleccion de valores de medicion

Restric_Alim=0;       % Si =0--> Se trabajara con todas las alimentaciones
Alimentacion='ABB1';  % Se determina el tipo de alimentacion al motor
                      % elegida

Restric_Carga=0;     % Si =0--> Se trabajara con todas las cargas
Nivel_Carga=2;       % Se determina el nivel de carga de funcionamiento del
                     % motor elegida
```

```
% Eleccion del metodo de clasificación

Metodo=6;           % Si =1--> Cascada de clasificadores binarios
                    % Si =2--> Clasificador Clasificador multiclase
                    % Si =3--> Clasificador binario R1vsAll + Clasificador
                    % multiclase
                    % Si =4--> Clasificador binario R1vsAll + Clasificador
                    % binario R2vsResto + Clasificador multiclase
                    % Si =5--> Clasificadores binarios y clasificador
                    % multiclase utilizados simultaneamente
                    % Si =6--> Clasificadores binarios y clasificador
                    % multiclase utilizados simultaneamente, con
preponderancia
                    % a la clasificación como 1

Muestra_Reducida=0;             % Si =1--> Clasificadores entrenan con los
                                % datos correspondientes reducidos

% Parametros de los clasificadores

NLearn_R1=50;             % Numero de ciclos clasificador para R1
NLearn_R2=50;             % Numero de ciclos clasificador para R2
NLearn_R3=50;             % Numero de ciclos clasificador para R3
NLearn_R4=50;             % Numero de ciclos clasificador para R4
NLearn_R5=50;             % Numero de ciclos clasificador para R5

NLearn_Multi=50;          % Numero de ciclos clasificador multiclase

KFold=20;                 % Subconjuntos para el CV del clasificador

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                        CARGA Y PREPARADO DE DATOS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Carga de valores del excel
[Datos_Inicio, Texto_Inicio, Todo_Inicio]=xlsread('datos_ignacio.xlsx');

Datos=Datos_Inicio;
Texto=Texto_Inicio;
Todo=Todo_Inicio;

% Tratamiento de los datos segun parametros elegidos previamente
Texto(1,:)=[];              % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
Todo(1,:)=[];               % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
if Restric_Alim==1
    % Buscar valores correspondientes a alimentacion elegida
    a=find(strncmp(Alimentacion,Texto(:,2),Num_caract));
    % Matriz con valores correspondientes a la alimentacion elegida
    Datos=Datos(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Texto=Texto(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Todo=Todo(a,:);
else
end

if Restric_Carga==1
```

```
    % Buscar valores numericos correspondientes a carga elegida
    b=find(Datos(:,2)==Nivel_Carga);
    % Matriz con valores correspondientes a los de la carga elegida
    Datos=Datos(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
    Texto=Texto(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
    Todo=Todo(b,:);
else
end

Datos(:,eliminar)=[];
Texto(:,eliminar+3)=[];%Sobra
Todo(:,eliminar+3)=[];%Sobra

% Busqueda de las medidas de cada clase

Clases=unique(Todo(:,3));
Num_Clases=length(Clases);        % Se determina el numero de clases
existentes

R1=find(strncmp('R1',Todo(:,3),Num_caract));
R2=find(strncmp('R2',Todo(:,3),Num_caract));
R3=find(strncmp('R3',Todo(:,3),Num_caract));
R4=find(strncmp('R4',Todo(:,3),Num_caract));
R5=find(strncmp('R5',Todo(:,3),Num_caract));

X=Datos;
Y_Clases=Todo(:,3);
Long_Y=length(Y_Clases);

Y=ones([Long_Y,1]);
Y(R1)=1;
Y(R2)=2;
Y(R3)=3;
Y(R4)=4;
Y(R5)=5;

% Preparamos los datos de la muestra reducida

if Reproduc==1
    rng(Seed,'twister') % Para reproducibilidad de las mediciones
else
end

cvpart=cvpartition(Y,'holdout',Partition);

X_Training=X(cvpart.training,:);
X_Test=X(cvpart.test,:);

Y_Training=Y(cvpart.training,:);
Y_Test=Y(cvpart.test,:);

Long_X_Test=length(X_Test);

X_Training_Clasif_R2vsR3R4R5=X_Training(Y_Training>1,:);
X_Training_Clasif_R3vsR4R5=X_Training(Y_Training>2,:);
X_Training_Clasif_R4vsR5=X_Training(Y_Training>3,:);
```

```
Y_Training_Clasif_R2vsR3R4R5=Y_Training(Y_Training>1);
Y_Training_Clasif_R3vsR4R5=Y_Training(Y_Training>2);
Y_Training_Clasif_R4vsR5=Y_Training(Y_Training>3);

Y_Binario=zeros(Num_Clases,Long_Y);
for i=1:Num_Clases
    Y_Binario(i,:)=(Y==i);
end

% Preparamos los datos test para representar la matriz de confusion
Y_Test_Mat_Conf=zeros(Num_Clases,Long_X_Test);
for i=1:Num_Clases
    Y_Test_Mat_Conf(i,:)=(Y_Test==i);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                ENTRENAMIENTO DE CLASIFICADORES Y PREDICCION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Error_Clasif_AdaboostM1=zeros(1,Num_Clases);

if Metodo==1
% Cascada de clasificadores binarios (uno vs restantes)

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
    Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
    'AdaBoostM1',NLearn_R1,'Tree');
    if isempty(Clasificador_R1.TrainedWeights)
        Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
        Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
        display('Error de clasificacion 0 Clasificador R1vsAll')
    else
        Error_Clasif_AdaboostM1(1)=1;
        Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
        NLearn_R1,'Tree');
    end
    % Cross Validation
    Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

    if Muestra_Reducida==1

        % Clasificador R2vsR3R4R5
        Y_Training_Clasif_Binario_R2=(Y_Training_Clasif_R2vsR3R4R5==2);
        Clasificador_R2=fitensemble(X_Training_Clasif_R2vsR3R4R5,
        Y_Training_Clasif_Binario_R2,'AdaBoostM1',NLearn_R2,'Tree');
        if isempty(Clasificador_R2.TrainedWeights)
            Clasificador_R2=fitctree(X_Training_Clasif_R2vsR3R4R5,
            Y_Training_Clasif_Binario_R2);
            Clasificador_Aux_R2=fitctree(X(Y>1),(Y(Y>1))==2);
            display('Error de clasificacion 0 Clasificador R2vsR3R4R5')
        else
            Error_Clasif_AdaboostM1(2)=1;
            Clasificador_Aux_R2=fitensemble(X(Y>1),(Y(Y>1))==2,
            'AdaBoostM1',NLearn_R2,'Tree');
        end
        % Cross Validation
        Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);
```

```
        % Clasificador R3vsR4R5
        Y_Training_Clasif_Binario_R3=(Y_Training_Clasif_R3vsR4R5==3);
        Clasificador_R3=fitensemble(X_Training_Clasif_R3vsR4R5,
        Y_Training_Clasif_Binario_R3,'AdaBoostM1',NLearn_R3,'Tree');
        if isempty(Clasificador_R3.TrainedWeights)
            Clasificador_R3=fitctree(X_Training_Clasif_R3vsR4R5,
            Y_Training_Clasif_Binario_R3);
            Clasificador_Aux_R3=fitctree(X(Y>2),(Y(Y>2))==3);
            display('Error de clasificacion 0 Clasificador R3vsR4R5')
        else
            Error_Clasif_AdaboostM1(3)=1;
            Clasificador_Aux_R3=fitensemble(X(Y>2),(Y(Y>2))==3,
            'AdaBoostM1',NLearn_R3,'Tree');
        end
        % Cross Validation
        Clasificador_CV_R3=crossval(Clasificador_Aux_R3,'KFold',KFold);

        % Clasificador R4vsR5
        Y_Training_Clasif_Binario_R4=(Y_Training_Clasif_R4vsR5==4);
        Clasificador_R4=fitensemble(X_Training_Clasif_R4vsR5,
        Y_Training_Clasif_Binario_R4,'AdaBoostM1',NLearn_R4,'Tree');
        if isempty(Clasificador_R4.TrainedWeights)
            Clasificador_R4=fitctree(X_Training_Clasif_R4vsR5,
            Y_Training_Clasif_Binario_R4);
            Clasificador_Aux_R4=fitctree(X(Y>3),(Y(Y>3))==4);
            display('Error de clasificacion 0 Clasificador R4vsR5')
        else
            Error_Clasif_AdaboostM1(4)=1;
            Clasificador_Aux_R4=fitensemble(X(Y>3),(Y(Y>3))==4,
            'AdaBoostM1',NLearn_R4,'Tree');
        end
        % Cross Validation
        Clasificador_CV_R4=crossval(Clasificador_Aux_R4,'KFold',KFold);
    else
        % Clasificador R2vsR3R4R5
        Y_Training_Clasif_Binario_R2=(Y_Training==2);
        Clasificador_R2=fitensemble(X_Training,
        Y_Training_Clasif_Binario_R2,'AdaBoostM1',NLearn_R2,'Tree');
        if isempty(Clasificador_R2.TrainedWeights)
            Clasificador_R2=fitctree(X_Training,
            Y_Training_Clasif_Binario_R2);
            Clasificador_Aux_R2=fitctree(X,Y_Binario(2,:));
            display('Error de clasificacion 0 Clasificador R2vsR3R4R5')
        else
            Error_Clasif_AdaboostM1(2)=1;
            Clasificador_Aux_R2=fitensemble(X,Y_Binario(2,:),'AdaBoostM1',
            NLearn_R2,'Tree');
        end
        % Cross Validation
        Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);

        % Clasificador R3vsR4R5
        Y_Training_Clasif_Binario_R3=(Y_Training==3);
        Clasificador_R3=fitensemble(X_Training,Y_Training_Clasif_Binario_R3
        ,'AdaBoostM1',NLearn_R3,'Tree');
        if isempty(Clasificador_R3.TrainedWeights)
            Clasificador_R3=fitctree(X_Training,
            Y_Training_Clasif_Binario_R3);
            Clasificador_Aux_R3=fitctree(X,Y_Binario(3,:));
            display('Error de clasificacion 0 Clasificador R3vsR4R5')
        else
            Error_Clasif_AdaboostM1(3)=1;
```

```
            Clasificador_Aux_R3=fitensemble(X,Y_Binario(3,:),'AdaBoostM1',
            NLearn_R3,'Tree');
        end
        % Cross Validation
        Clasificador_CV_R3=crossval(Clasificador_Aux_R3,'KFold',KFold);

        % Clasificador R4vsR5
        Y_Training_Clasif_Binario_R4=(Y_Training==4);
         Clasificador_R4=fitensemble(X_Training,
        Y_Training_Clasif_Binario_R4,'AdaBoostM1',NLearn_R4,'Tree');
        if isempty(Clasificador_R4.TrainedWeights)
            Clasificador_R4=fitctree(X_Training,
            Y_Training_Clasif_Binario_R4);
            Clasificador_Aux_R4=fitctree(X,Y_Binario(4,:));
            display('Error de clasificacion 0 Clasificador R4vsR5')
        else
            Error_Clasif_AdaboostM1(4)=1;
            Clasificador_Aux_R4=fitensemble(X,Y_Binario(4,:),'AdaBoostM1',
            NLearn_R4,'Tree');
        end
        % Cross Validation
        Clasificador_CV_R4=crossval(Clasificador_Aux_R4,'KFold',KFold);
    end

    % Cascada de clasificadores. Validacion mediante particion de prueba
    Clasificacion=zeros([Long_X_Test,1]);

    for i=1:Long_X_Test
        Clasificacion(i)=predict(Clasificador_R1,X_Test(i,:));
        if Clasificacion(i)==1
        else
            Clasificacion(i)=predict(Clasificador_R2,X_Test(i,:));
            if Clasificacion(i)==1
                Clasificacion(i)=2;
            else
                Clasificacion(i)=predict(Clasificador_R3,X_Test(i,:));
                if Clasificacion(i)==1
                    Clasificacion(i)=3;
                else
                    Clasificacion(i)=predict(Clasificador_R4,X_Test(i,:));
                    if Clasificacion(i)==1
                        Clasificacion(i)=4;
                    else
                        Clasificacion(i)=5;
                    end
                end
            end
        end
    end

elseif Metodo==2
% Clasificador Adaboost Multiclase (todas las clases)

    % Entrenamiento del clasificador
    Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,'AdaBoostM2',
    NLearn_Multi,'Tree');
    % Cross Validation
     Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',NLearn_Multi,
    'Tree');
    Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
    'KFold',KFold);
```

```
    % validacion mediante particion de prueba
    Clasificacion=predict(Clasificador_AdaboostM2,X_Test);

elseif Metodo==3
% Aplicacion de clasificador Adaboost Biclase para R1vsAll, y Adaboost
% Multiclase para el resto de clases

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
     Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
'AdaBoostM1',NLearn_R1,'Tree');
    if isempty(Clasificador_R1.TrainedWeights)
        Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
        Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
        display('Error de clasificacion 0 Clasificador R1vsAll')
    else
        Error_Clasif_AdaboostM1(1)=1;
        Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
        NLearn_R1,'Tree');
    end
    % Cross Validation
    Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

    % Clasificador Multiclase
    if Muestra_Reducida==1
        Clasificador_AdaboostM2=fitensemble(X_Training_Clasif_R2vsR3R4R5,
        Y_Training_Clasif_R2vsR3R4R5,'AdaBoostM2',NLearn_Multi,'Tree');
        % Cross Validation
        Clasificador_AdaboostM2_Aux=fitensemble(X(Y>1,:),Y(Y>1,:),
        'AdaBoostM2',NLearn_Multi,'Tree');
        Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
        'KFold',KFold);
    else
        Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,
        'AdaBoostM2',NLearn_Multi,'Tree');
        % Cross Validation
        Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',
        NLearn_Multi,'Tree');
        Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
        'KFold',KFold);
    end

    % Cascada de clasificadores. Validacion mediante particion de prueba
    Clasificacion=zeros([Long_X_Test,1]);

    for i=1:Long_X_Test
        Clasificacion(i)=predict(Clasificador_R1,X_Test(i,:));
        if Clasificacion(i)==1
        else
            Clasificacion(i)=predict(Clasificador_AdaboostM2,X_Test(i,:));
        end
    end

elseif Metodo==4
% Aplicacion de clasificador Adaboost Biclase para R1vsAll y R2vsR3R4R5 y
% Adaboost Multiclase para el resto de clases

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
   Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
```

```
'AdaBoostM1',NLearn_R1,'Tree');
if isempty(Clasificador_R1.TrainedWeights)
    Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
    Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
    display('Error de clasificacion 0 Clasificador R1vsAll')
else
    Error_Clasif_AdaboostM1(1)=1;
    Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
    NLearn_R1,'Tree');
end
% Cross Validation
Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

if Muestra_Reducida==1
    % Clasificador R2vsR3R4R5
    Y_Training_Clasif_Binario_R2=(Y_Training_Clasif_R2vsR3R4R5==2);
    Clasificador_R2=fitensemble(X_Training_Clasif_R2vsR3R4R5,
    Y_Training_Clasif_Binario_R2,'AdaBoostM1',NLearn_R2,'Tree');
    if isempty(Clasificador_R2.TrainedWeights)
        Clasificador_R2=fitctree(X_Training_Clasif_R2vsR3R4R5,
        Y_Training_Clasif_Binario_R2);
        Clasificador_Aux_R2=fitctree(X(Y>1),(Y(Y>1))==2);
        display('Error de clasificacion 0 Clasificador R2vsR3R4R5')
    else
        Error_Clasif_AdaboostM1(2)=1;
        Clasificador_Aux_R2=fitensemble(X(Y>1),(Y(Y>1))==2,
        'AdaBoostM1',NLearn_R2,'Tree');
    end
    % Cross Validation
    Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);

    Clasificador_AdaboostM2=fitensemble(X_Training_Clasif_R2vsR3R4R5,
    Y_Training_Clasif_R2vsR3R4R5,'AdaBoostM2',NLearn_Multi,'Tree');
    % Cross Validation
    Clasificador_AdaboostM2_Aux=fitensemble(X(Y>1,:),Y(Y>1,:),
    'AdaBoostM2',NLearn_Multi,'Tree');
    Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
    'KFold',KFold);
else
    % Clasificador R2vsR3R4R5
    Y_Training_Clasif_Binario_R2=(Y_Training==2);
    Clasificador_R2=fitensemble(X_Training,
    Y_Training_Clasif_Binario_R2,'AdaBoostM1',NLearn_R2,'Tree');
    if isempty(Clasificador_R2.TrainedWeights)
        Clasificador_R2=fitctree(X_Training,
        Y_Training_Clasif_Binario_R2);
        Clasificador_Aux_R2=fitctree(X,Y_Binario(2,:));
        display('Error de clasificacion 0 Clasificador R2vsR3R4R5')
    else
        Error_Clasif_AdaboostM1(2)=1;
        Clasificador_Aux_R2=fitensemble(X,Y_Binario(2,:),'AdaBoostM1',
        NLearn_R2,'Tree');
    end
    % Cross Validation
    Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);

    Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,
    'AdaBoostM2',NLearn_Multi,'Tree');
    % Cross Validation
    Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',
    NLearn_Multi,'Tree');
```

```
        Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
        'KFold',KFold);
    end

    % Cascada de clasificadores. Validacion mediante particion de prueba
    Clasificacion=zeros([Long_X_Test,1]);

    for i=1:Long_X_Test
        Clasificacion(i)=predict(Clasificador_R1,X_Test(i,:));
        if Clasificacion(i)==1
        else
            Clasificacion(i)=predict(Clasificador_R2,X_Test(i,:));
            if Clasificacion(i)==1
                Clasificacion(i)=2;
            else
                Clasificacion(i)=predict(Clasificador_AdaboostM2,
                X_Test(i,:));
            end
        end
    end

elseif Metodo==5||Metodo==6
% Metodo 5: Aplicacion de todos los clasificadores binarios, multiclase
% para desempatar resultados

% Metodo 6: Aplicacion de todos los clasificadores binarios y multiclase,
% primando redultados correspondientes a la clase 1

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
    Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
    'AdaBoostM1',NLearn_R1,'Tree');

    if isempty(Clasificador_R1.TrainedWeights)
        Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
        Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
        display('Error de clasificacion 0 Clasificador R1vsAll')
    else
        Error_Clasif_AdaboostM1(1)=1;
        Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
        NLearn_R1,'Tree');
    end
    % Cross Validation
    Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

    % Clasificador R2vsAll
    Y_Training_Clasif_Binario_R2=(Y_Training==2);
    Clasificador_R2=fitensemble(X_Training,Y_Training_Clasif_Binario_R2,
    'AdaBoostM1',NLearn_R2,'Tree');

    if isempty(Clasificador_R2.TrainedWeights)
        Clasificador_R2=fitctree(X_Training,Y_Training_Clasif_Binario_R2);
        Clasificador_Aux_R2=fitctree(X,Y_Binario(2,:));
        display('Error de clasificacion 0 Clasificador R2vsAll')
    else
        Error_Clasif_AdaboostM1(2)=1;
        Clasificador_Aux_R2=fitensemble(X,Y_Binario(2,:),'AdaBoostM1',
        NLearn_R2,'Tree');
    end
    % Cross Validation
    Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);
```

```
% Clasificador R3vsAll
Y_Training_Clasif_Binario_R3=(Y_Training==3);
Clasificador_R3=fitensemble(X_Training,Y_Training_Clasif_Binario_R3,
'AdaBoostM1',NLearn_R3,'Tree');

if isempty(Clasificador_R3.TrainedWeights)
    Clasificador_R3=fitctree(X_Training,Y_Training_Clasif_Binario_R3);
    Clasificador_Aux_R3=fitctree(X,Y_Binario(3,:));
    display('Error de clasificacion 0 Clasificador R3vsAll')
else
    Error_Clasif_AdaboostM1(3)=1;
    Clasificador_Aux_R3=fitensemble(X,Y_Binario(3,:),'AdaBoostM1',
    NLearn_R3,'Tree');
end
% Cross Validation
Clasificador_CV_R3=crossval(Clasificador_Aux_R3,'KFold',KFold);

% Clasificador R4vsAll
Y_Training_Clasif_Binario_R4=(Y_Training==4);
Clasificador_R4=fitensemble(X_Training,Y_Training_Clasif_Binario_R4,
'AdaBoostM1',NLearn_R4,'Tree');

if isempty(Clasificador_R4.TrainedWeights)
    Clasificador_R4=fitctree(X_Training,Y_Training_Clasif_Binario_R4);
    Clasificador_Aux_R4=fitctree(X,Y_Binario(4,:));
    display('Error de clasificacion 0 Clasificador R4vsAll')
else
    Error_Clasif_AdaboostM1(4)=1;
    Clasificador_Aux_R4=fitensemble(X,Y_Binario(4,:),'AdaBoostM1',
    NLearn_R4,'Tree');
end
% Cross Validation
Clasificador_CV_R4=crossval(Clasificador_Aux_R4,'KFold',KFold);

% Clasificador R5vsAll
Y_Training_Clasif_Binario_R5=(Y_Training==5);
Clasificador_R5=fitensemble(X_Training,Y_Training_Clasif_Binario_R5,
'AdaBoostM1',NLearn_R5,'Tree');

if isempty(Clasificador_R5.TrainedWeights)
    Clasificador_R5=fitctree(X_Training,Y_Training_Clasif_Binario_R5);
    Clasificador_Aux_R5=fitctree(X,Y_Binario(5,:));
    display('Error de clasificacion 0 Clasificador R5vsAll')
else
    Error_Clasif_AdaboostM1(5)=1;
    Clasificador_Aux_R5=fitensemble(X,Y_Binario(5,:),'AdaBoostM1',
    NLearn_R5,'Tree');
end
% Cross Validation
Clasificador_CV_R5=crossval(Clasificador_Aux_R5,'KFold',KFold);

% Clasificador Multiclase
Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,'AdaBoostM2',
NLearn_Multi,'Tree');
% Cross Validation
Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',NLearn_Multi,
'Tree');
Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
'KFold',KFold);
```

```
    Clasificacion_Tot=zeros([6,Long_X_Test]);

    Clasificacion_Tot(1,:)=predict(Clasificador_R1,X_Test);
    Clasificacion_Tot(2,:)=predict(Clasificador_R2,X_Test);
    Clasificacion_Tot(3,:)=predict(Clasificador_R3,X_Test);
    Clasificacion_Tot(4,:)=predict(Clasificador_R4,X_Test);
    Clasificacion_Tot(5,:)=predict(Clasificador_R5,X_Test);

    Clasificacion=zeros([Long_X_Test,1]);

    if Metodo==5
        % Validacion mediante particion de prueba
        for i=1:Long_X_Test
            if sum(Clasificacion_Tot(:,i))==1
                for k=1:Num_Clases
                    if (Clasificacion_Tot(k,i))==1
                        Clasificacion(i)=k;
                    else
                    end
                end
            else
                Clasificacion_Tot(6,i)=predict(Clasificador_AdaboostM2,
                X_Test(i,:));
                 Clasificacion(i)=Clasificacion_Tot(6,i);
            end
        end
    else
        Clasificacion_Tot(6,:)=predict(Clasificador_AdaboostM2,X_Test);

        % Validacion mediante particion de prueba
        for i=1:Long_X_Test
            if (Clasificacion_Tot(1,i)==0)||(Clasificacion_Tot(6,i)~=1)
                if (sum(Clasificacion_Tot([2 3 4 5],i)>0))==0
                    if (Clasificacion_Tot(6,i)~=1)
                        Clasificacion(i)=Clasificacion_Tot(6,i);
                    else
                        Clasificacion(i)=4;
                    end
                elseif (sum(Clasificacion_Tot([2 3 4 5],i)>0))>1
                    Clasificacion(i)=Clasificacion_Tot(6,i);
                else
                    Clasificacion(i)=1+find(Clasificacion_Tot([2 3 4
                    5],i)>0);
                end
            else
                Clasificacion(i)=1;
            end
        end
    end

else
    display('Introduzca un valor para seleccionar el método de clasificación
    comprendido entre 0 y 4')
    Error_Clasif_AdaboostM1=9999; % Valor de control, por si el usuario ha
    elegido un método inexistente
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      ERROR DE RESISTITUCION Y ERROR DE CLASIFICACION MEDIANTE CV
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if Metodo==1
    if Error_Clasif_AdaboostM1(1)==1
        RsLoss_R1=resubLoss(Clasificador_R1,'Mode','Cumulative');
        GenError_R1=kfoldLoss(Clasificador_CV_R1,'Mode','Cumulative');
    else
        RsLoss_R1=resubLoss(Clasificador_R1)*ones(1,NLearn_R1)';
        GenError_R1=kfoldLoss(Clasificador_CV_R3)*ones(1,NLearn_R1)';
    end
    RsLoss_R1=RsLoss_R1*100;
    GenError_R1=GenError_R1*100;

    if Error_Clasif_AdaboostM1(2)==1
        RsLoss_R2=resubLoss(Clasificador_R2,'Mode','Cumulative');
        GenError_R2=kfoldLoss(Clasificador_CV_R2,'Mode','Cumulative');
    else
        RsLoss_R2=resubLoss(Clasificador_R2)*ones(1,NLearn_R2)';
        GenError_R2=kfoldLoss(Clasificador_CV_R3)*ones(1,NLearn_R2)';
    end
    RsLoss_R2=RsLoss_R2*100;
    GenError_R2=GenError_R2*100;

    if Error_Clasif_AdaboostM1(3)==1
        RsLoss_R3=resubLoss(Clasificador_R3,'Mode','Cumulative');
        GenError_R3=kfoldLoss(Clasificador_CV_R3,'Mode','Cumulative');
    else
        RsLoss_R3=resubLoss(Clasificador_R3)*ones(1,NLearn_R3)';
        GenError_R3=kfoldLoss(Clasificador_CV_R3)*ones(1,NLearn_R3)';
    end
    RsLoss_R3=RsLoss_R3*100;
    GenError_R3=GenError_R3*100;

    if Error_Clasif_AdaboostM1(4)==1
        RsLoss_R4=resubLoss(Clasificador_R4,'Mode','Cumulative');
        GenError_R4=kfoldLoss(Clasificador_CV_R4,'Mode','Cumulative');
    else
        RsLoss_R4=resubLoss(Clasificador_R4)*ones(1,NLearn_R4)';
        GenError_R4=kfoldLoss(Clasificador_CV_R4)*ones(1,NLearn_R4)';
    end
    RsLoss_R4=RsLoss_R4*100;
    GenError_R4=GenError_R4*100;

    figure;
    subplot(2,2,1);
    plot((1:1:NLearn_R1),RsLoss_R1,(1:1:NLearn_R1),GenError_R1);
    title('\fontsize{14}Clasificador R1vsAll');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(2,2,2);
    plot((1:1:NLearn_R2),RsLoss_R2,(1:1:NLearn_R2),GenError_R2);
    title('\fontsize{14}Clasificador R2vsR3R4R5');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
```

```
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(2,2,3);
    plot((1:1:NLearn_R3),RsLoss_R3,(1:1:NLearn_R3),GenError_R3);
    title('\fontsize{14}Clasificador R3vsR4R5');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(2,2,4);
    plot((1:1:NLearn_R4),RsLoss_R4,(1:1:NLearn_R4),GenError_R4);
    title('\fontsize{14}Clasificador R4vsR5');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

elseif Metodo==2
    RsLoss_Multiclase=100*resubLoss(Clasificador_AdaboostM2,'Mode',
    'Cumulative');
    GenError_Multiclase=100*kfoldLoss(Clasificador_AdaboostM2_CV,'Mode',
    'Cumulative');
    plot((1:1:NLearn_Multi),RsLoss_Multiclase,(1:1:NLearn_Multi),
    GenError_Multiclase);
    title('\fontsize{24}Clasificador Multiclase');
    legend('\fontsize{18}Resubstitution Classification Cost',
    '\fontsize{18}Cross-Validation Loss');
    xlabel('\fontsize{20}Number of Learning Cycles');
    ylabel('\fontsize{20}Loss (%)');
    ylim([0 inf]);

elseif Metodo==3
    if Error_Clasif_AdaboostM1(1)==1
        RsLoss_R1=resubLoss(Clasificador_R1,'Mode','Cumulative');
        GenError_R1=kfoldLoss(Clasificador_CV_R1,'Mode','Cumulative');
    else
        RsLoss_R1=resubLoss(Clasificador_R1)*ones(1,NLearn_R1)';
        GenError_R1=kfoldLoss(Clasificador_CV_R1)*ones(1,NLearn_R1)';
    end
    RsLoss_R1=RsLoss_R1*100;
    GenError_R1=GenError_R1*100;

    RsLoss_Multiclase=100*resubLoss(Clasificador_AdaboostM2,'Mode',
    'Cumulative');
    GenError_Multiclase=100*kfoldLoss(Clasificador_AdaboostM2_CV,'Mode',
    'Cumulative');

    figure;
    subplot(2,1,1);
    plot((1:1:NLearn_R1),RsLoss_R1,(1:1:NLearn_R1),GenError_R1);
    title('\fontsize{22}Clasificador R1vsAll');
    legend('\fontsize{12}Resubstitution Classification Cost',
    '\fontsize{12}Cross-Validation Loss');
    xlabel('\fontsize{20}Number of Learning Cycles');
    ylabel('\fontsize{20}Loss (%)');
    ylim([0 inf]);
```

```
    subplot(2,1,2);
    plot((1:1:NLearn_Multi),RsLoss_Multiclase,(1:1:NLearn_Multi),
    GenError_Multiclase);
    title('\fontsize{22}Clasificador Multiclase');
    legend('\fontsize{12}Resubstitution Classification Cost',
    '\fontsize{12}Cross-Validation Loss');
    xlabel('\fontsize{20}Number of Learning Cycles');
    ylabel('\fontsize{20}Loss (%)');
    ylim([0 inf]);

elseif Metodo==4
    if Error_Clasif_AdaboostM1(1)==1
        RsLoss_R1=resubLoss(Clasificador_R1,'Mode','Cumulative');
        GenError_R1=kfoldLoss(Clasificador_CV_R1,'Mode','Cumulative');
    else
        RsLoss_R1=resubLoss(Clasificador_R1)*ones(1,NLearn_R1)';
        GenError_R1=kfoldLoss(Clasificador_CV_R1)*ones(1,NLearn_R1)';
    end
    RsLoss_R1=RsLoss_R1*100;
    GenError_R1=GenError_R1*100;

    if Error_Clasif_AdaboostM1(2)==1
        RsLoss_R2=resubLoss(Clasificador_R2,'Mode','Cumulative');
        GenError_R2=kfoldLoss(Clasificador_CV_R2,'Mode','Cumulative');
    else
        RsLoss_R2=resubLoss(Clasificador_R2)*ones(1,NLearn_R2)';
        GenError_R2=kfoldLoss(Clasificador_CV_R2)*ones(1,NLearn_R2)';
    end
    RsLoss_R2=RsLoss_R2*100;
    GenError_R2=GenError_R2*100;

    RsLoss_Multiclase=100*resubLoss(Clasificador_AdaboostM2,'Mode',
    'Cumulative');
    GenError_Multiclase=100*kfoldLoss(Clasificador_AdaboostM2_CV,'Mode',
    'Cumulative');

    figure;
    subplot(2,2,1);
    plot((1:1:NLearn_R1),RsLoss_R1,(1:1:NLearn_R1),GenError_R1);
    title('\fontsize{14}Clasificador R1vsAll');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(2,2,2);
    plot((1:1:NLearn_R2),RsLoss_R2,(1:1:NLearn_R2),GenError_R2);
    title('\fontsize{14}Clasificador R2vsR3R4R5');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(2,2,[3,4]);
    plot((1:1:NLearn_Multi),RsLoss_Multiclase,(1:1:NLearn_Multi),
    GenError_Multiclase);
    title('\fontsize{14}Clasificador Multiclase');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
```

```
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

elseif Metodo==5||Metodo==6
    if Error_Clasif_AdaboostM1(1)==1
        RsLoss_R1=resubLoss(Clasificador_R1,'Mode','Cumulative');
        GenError_R1=kfoldLoss(Clasificador_CV_R1,'Mode','Cumulative');
    else
        RsLoss_R1=resubLoss(Clasificador_R1)*ones(1,NLearn_R1)';
        GenError_R1=kfoldLoss(Clasificador_CV_R1)*ones(1,NLearn_R1)';
    end
    RsLoss_R1=RsLoss_R1*100;
    GenError_R1=GenError_R1*100;

    if Error_Clasif_AdaboostM1(2)==1
        RsLoss_R2=resubLoss(Clasificador_R2,'Mode','Cumulative');
        GenError_R2=kfoldLoss(Clasificador_CV_R2,'Mode','Cumulative');
    else
        RsLoss_R2=resubLoss(Clasificador_R2)*ones(1,NLearn_R2)';
        GenError_R2=kfoldLoss(Clasificador_CV_R2)*ones(1,NLearn_R2)';
    end
    RsLoss_R2=RsLoss_R2*100;
    GenError_R2=GenError_R2*100;

    if Error_Clasif_AdaboostM1(3)==1
        RsLoss_R3=resubLoss(Clasificador_R3,'Mode','Cumulative');
        GenError_R3=kfoldLoss(Clasificador_CV_R3,'Mode','Cumulative');
    else
        RsLoss_R3=resubLoss(Clasificador_R3)*ones(1,NLearn_R3)';
        GenError_R3=kfoldLoss(Clasificador_CV_R3)*ones(1,NLearn_R3)';
    end
    RsLoss_R3=RsLoss_R3*100;
    GenError_R3=GenError_R3*100;

    if Error_Clasif_AdaboostM1(4)==1
        RsLoss_R4=resubLoss(Clasificador_R4,'Mode','Cumulative');
        GenError_R4=kfoldLoss(Clasificador_CV_R4,'Mode','Cumulative');
    else
        RsLoss_R4=resubLoss(Clasificador_R4)*ones(1,NLearn_R4)';
        GenError_R4=kfoldLoss(Clasificador_CV_R4)*ones(1,NLearn_R4)';
    end
    RsLoss_R4=RsLoss_R4*100;
    GenError_R4=GenError_R4*100;

    if Error_Clasif_AdaboostM1(5)==1
        RsLoss_R5=resubLoss(Clasificador_R5,'Mode','Cumulative');
        GenError_R5=kfoldLoss(Clasificador_CV_R5,'Mode','Cumulative');
    else
        RsLoss_R5=resubLoss(Clasificador_R5)*ones(1,NLearn_R5)';
        GenError_R5=kfoldLoss(Clasificador_CV_R5)*ones(1,NLearn_R5)';
    end
    RsLoss_R5=RsLoss_R5*100;
    GenError_R5=GenError_R5*100;

    RsLoss_Multiclase=100*resubLoss(Clasificador_AdaboostM2,'Mode',
    'Cumulative');
    GenError_Multiclase=100*kfoldLoss(Clasificador_AdaboostM2_CV,'Mode',
    'Cumulative');
```

```
    figure;
    subplot(3,2,1);
    plot((1:1:NLearn_R1),RsLoss_R1,(1:1:NLearn_R1),GenError_R1);
    title('\fontsize{14}Clasificador R1vsAll');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(3,2,2);
    plot((1:1:NLearn_R2),RsLoss_R2,(1:1:NLearn_R2),GenError_R2);
    title('\fontsize{14}Clasificador R2vsAll');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(3,2,3);
    plot((1:1:NLearn_R3),RsLoss_R3,(1:1:NLearn_R3),GenError_R3);
    title('\fontsize{14}Clasificador R3vsAll');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(3,2,4);
    plot((1:1:NLearn_R4),RsLoss_R4,(1:1:NLearn_R4),GenError_R4);
    title('\fontsize{14}Clasificador R4vsAll');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);

    subplot(3,2,5);
    plot((1:1:NLearn_R5),RsLoss_R5,(1:1:NLearn_R5),GenError_R5);
    title('\fontsize{14}Clasificador R4vsAll');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);
    subplot(3,2,6);
    plot((1:1:NLearn_Multi),RsLoss_Multiclase,(1:1:NLearn_Multi),
    GenError_Multiclase);
    title('\fontsize{14}Clasificador Multiclase');
    legend('\fontsize{13}Resubstitution Classification Cost',
    '\fontsize{13}Cross-Validation Loss');
    xlabel('\fontsize{13}Number of Learning Cycles');
    ylabel('\fontsize{13}Loss (%)');
    ylim([0 inf]);
else
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    MATRIZ DE CONFUSION Y CURVA CARACTERISTICA OPERATIVA DEL RECEPTOR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if Error_Clasif_AdaboostM1==9999
else
    [Confusion_Matrix,order]=confusionmat(Y_Test,Clasificacion);

    % Preparamos los datos test para representar la matriz de confusion
    Y_Clasificacion_Mat_Conf=zeros(Num_Clases,Long_X_Test);
    for i=1:Num_Clases
        Y_Clasificacion_Mat_Conf(i,(Clasificacion==i))=1;
    end

    % Representamos la matriz de confusion
    figure
    plotconfusion(Y_Test_Mat_Conf,Y_Clasificacion_Mat_Conf)

    % Representamos la curva caracteristica operativa del receptor
    figure
    plotroc(Y_Test_Mat_Conf,Y_Clasificacion_Mat_Conf)
end
```

1.4 Script mejorado

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%              TRABAJO FIN DE MASTER           %%%%%%%%%%%%%
%
%     DESARROLLO DE UNA HERRAMIENTA DE DIAGNOSTICO DE FALLOS EN MOTORES
%                  DE INDUCCION MEDIANTE LA TECNICA ADABOOST
%
%                            JUAN OBREGÓN SANDOVAL
%%%%%%%%%%%%%                                                %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%

%  ENTRENAMIENTO Y ENSAYO DE METODOS DE CLASIFICACION MULTICLASE MEDIANTE
%   COMBINACION DE CLASIFICADORES BINARIOS Y MULTICLASE ADABOOST Y ARBOL

%-------------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear variables;
clc;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                     INICIALIZACION DE VARIABLES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Parametros auxiliares
Num_caract=7;                  % Longitud de busqueda de strings para
                               % elegir datos acordes a Alimentacion y a
                               % Nivel_Carga

eliminar=[1 2 3 4 5 6 9 10]';  % Filas a eliminar de Datos (datos no
                               % utilizados, no son estimadores ni clases)

Reproduc=1;          % Si =1--> Se fija el valor de la semilla para
                     % reproductibilidad
Seed=320;            % Inicializacion semilla

% Parametro para particionado de datos
Partition=0.3;                 % Porcentaje datos usados como test


% Parametros para seleccion de valores de medicion
Restric_Alim=0;       % Si =0--> Se trabajara con todas las alimentaciones
Alimentacion='ABB1'; % Se determina el tipo de alimentacion al motor
elegida

Restric_Carga=0;     % Si =0--> Se trabajara con todas las cargas
Nivel_Carga=2;       % Se determina el nivel de carga de funcionamiento del
motor elegida

% Eleccion del metodo de clasificacion
Metodo=5;            % Si =1--> Cascada de clasificadores binarios
                     % Si =2--> Clasificador Clasificador multiclase
                     % Si =3--> Clasificador binario R1vsAll + Clasificador
```

```
                    % multiclase
                    % Si =4--> Clasificador binario R1vsAll + Clasificador
                    % binario R2vsResto + Clasificador multiclase
                    % Si =5--> Clasificadores binarios y clasificador
                    % multiclase utilizados simultaneamente
                    % Si =6--> Clasificadores binarios y clasificador
                    % multiclase utilizados simultaneamente, con
                    % preponderancia a la clasificación como 1

% Parametros de los clasificadores
NLearn_R1=50;             % Numero de ciclos clasificador para R1
NLearn_R2=50;             % Numero de ciclos clasificador para R2
NLearn_R3=50;             % Numero de ciclos clasificador para R3
NLearn_R4=50;             % Numero de ciclos clasificador para R4
NLearn_R5=50;             % Numero de ciclos clasificador para R5

NLearn_Multi=50;          % Numero de ciclos clasificador multiclase

LearnRate=1;              % Tasa de entrenamiento de los clasificadores
MaxNumSplits=20;          % Maximo número de escisiones en el arbol de
                          % decisión del algoritmo de aprensizaje

KFold=20;                 % Subconjuntos para el CV del clasificador

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                       CARGA Y PREPARADO DE DATOS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Carga de valores del excel
[Datos_Inicio, Texto_Inicio, Todo_Inicio]=xlsread('datos_ignacio.xlsx'); %
Cargamos valores del excel

Datos=Datos_Inicio;
Texto=Texto_Inicio;
Todo=Todo_Inicio;

% Tratamiento de los datos segun parametros elegidos previamente
Texto(1,:)=[];              % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
Todo(1,:)=[];               % Eliminamos fila para cuadrar dimensiones con
                            % "Datos"
if Restric_Alim==1
    % Buscar valores correspondientes a alimentacion elegida
    a=find(strncmp(Alimentacion,Texto(:,2),Num_caract));
    % Matriz con valores correspondientes a la alimentacion elegida
    Datos=Datos(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Texto=Texto(a,:);
    % Matriz con valores correspondientes a la alimentacion elegida
    Todo=Todo(a,:);
else
end
if Restric_Carga==1
    % Buscar valores numericos correspondientes a carga elegida
    b=find(Datos(:,2)==Nivel_Carga);
    % Matriz con valores correspondientes a los de la carga elegida
    Datos=Datos(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
    Texto=Texto(b,:);
    % Matriz con valores correspondientes a los de la carga elegida
```

```
    Todo=Todo(b,:);
else
end

Datos(:,eliminar)=[];
Texto(:,eliminar+3)=[];
Todo(:,eliminar+3)=[];

% Busqueda de las medidas de cada clase
Clases=unique(Todo(:,3));
Num_Clases=length(Clases);        % Se determina el numero de clases
existentes

R1=find(strncmp('R1',Todo(:,3),Num_caract));
R2=find(strncmp('R2',Todo(:,3),Num_caract));
R3=find(strncmp('R3',Todo(:,3),Num_caract));
R4=find(strncmp('R4',Todo(:,3),Num_caract));
R5=find(strncmp('R5',Todo(:,3),Num_caract));

X=Datos;
Y_Clases=Todo(:,3);
Long_Y=length(Y_Clases);

Y=ones([Long_Y,1]);
Y(R1)=1;
Y(R2)=2;
Y(R3)=3;
Y(R4)=4;
Y(R5)=5;

% Preparamos los datos de la muestra reducida
if Reproduc==1
    rng(Seed,'twister') % Para reproducibilidad de las mediciones
else
end

cvpart=cvpartition(Y,'holdout',Partition);

X_Training=X(cvpart.training,:);
X_Test=X(cvpart.test,:);

Y_Training=Y(cvpart.training,:);
Y_Test=Y(cvpart.test,:);

Long_X_Test=length(X_Test);
X_Training_Clasif_R2vsR3R4R5=X_Training(Y_Training>1,:);
X_Training_Clasif_R3vsR4R5=X_Training(Y_Training>2,:);
X_Training_Clasif_R4vsR5=X_Training(Y_Training>3,:);

Y_Training_Clasif_R2vsR3R4R5=Y_Training(Y_Training>1);
Y_Training_Clasif_R3vsR4R5=Y_Training(Y_Training>2);
Y_Training_Clasif_R4vsR5=Y_Training(Y_Training>3);

Y_Binario=zeros(Num_Clases,Long_Y);
for i=1:Num_Clases
    Y_Binario(i,:)=(Y==i);
end

% Preparamos los datos test para representar la matriz de confusion
```

```
Y_Test_Mat_Conf=zeros(Num_Clases,Long_X_Test);
for i=1:Num_Clases
    Y_Test_Mat_Conf(i,:)=(Y_Test==i);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                 ENTRENAMIENTO DE CLASIFICADORES Y PREDICCION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Error_Clasif_AdaboostM1=zeros(1,Num_Clases);
t=templateTree('MaxNumSplits',MaxNumSplits,'Surrogate','on');

if Metodo==1
% Cascada de clasificadores binarios (uno vs restantes)

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
    Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
    'AdaBoostM1',NLearn_R1,t,'LearnRate',LearnRate);
    if isempty(Clasificador_R1.TrainedWeights)
        Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
        Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
        display('Error de clasificacion 0 Clasificador R1vsAll')
    else
        Error_Clasif_AdaboostM1(1)=1;
        Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
        NLearn_R1,t,'LearnRate',LearnRate);
    end
    % Cross Validation
    Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

    % Clasificador R2vsR3R4R5
    Y_Training_Clasif_Binario_R2=(Y_Training==2);
     Clasificador_R2=fitensemble(X_Training,Y_Training_Clasif_Binario_R2,
    'AdaBoostM1',NLearn_R2,t,'LearnRate',LearnRate);
    if isempty(Clasificador_R2.TrainedWeights)
        Clasificador_R2=fitctree(X_Training,Y_Training_Clasif_Binario_R2);
        Clasificador_Aux_R2=fitctree(X,Y_Binario(2,:));
        display('Error de clasificacion 0 Clasificador R2vsR3R4R5')
    else
        Error_Clasif_AdaboostM1(2)=1;
        Clasificador_Aux_R2=fitensemble(X,Y_Binario(2,:),'AdaBoostM1',
        NLearn_R2,t,'LearnRate',LearnRate);
    end
    % Cross Validation
    Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);

    % Clasificador R3vsR4R5
    Y_Training_Clasif_Binario_R3=(Y_Training==3);
    Clasificador_R3=fitensemble(X_Training,Y_Training_Clasif_Binario_R3,
    'AdaBoostM1',NLearn_R3,t,'LearnRate',LearnRate);
    if isempty(Clasificador_R3.TrainedWeights)
        Clasificador_R3=fitctree(X_Training,Y_Training_Clasif_Binario_R3);
        Clasificador_Aux_R3=fitctree(X,Y_Binario(3,:));
        display('Error de clasificacion 0 Clasificador R3vsR4R5')
    else
        Error_Clasif_AdaboostM1(3)=1;
        Clasificador_Aux_R3=fitensemble(X,Y_Binario(3,:),'AdaBoostM1',
        NLearn_R3,t,'LearnRate',LearnRate);
    end
    % Cross Validation
```

```
    Clasificador_CV_R3=crossval(Clasificador_Aux_R3,'KFold',KFold);

    % Clasificador R4vsR5
    Y_Training_Clasif_Binario_R4=(Y_Training_Clasif_R4vsR5==4);

    Clasificador_R4=fitensemble(X_Training_Clasif_R4vsR5,Y_Training_Clasif_B
    inario_R4,'AdaBoostM1',NLearn_R4,t,'LearnRate',LearnRate);
    if isempty(Clasificador_R4.TrainedWeights)

        Clasificador_R4=fitctree(X_Training_Clasif_R4vsR5,Y_Training_Clasif
        _Binario_R4);
        Clasificador_Aux_R4=fitctree(X(Y>3),(Y(Y>3))==4);
        display('Error de clasificacion 0 Clasificador R4vsR5')
    else
        Error_Clasif_AdaboostM1(4)=1;
        Clasificador_Aux_R4=fitensemble(X(Y>3),(Y(Y>3))==4,'AdaBoostM1',
        NLearn_R4,t,'LearnRate',LearnRate);
    end
    % Cross Validation
    Clasificador_CV_R4=crossval(Clasificador_Aux_R4,'KFold',KFold);

    % Cascada de clasificadores. Validacion mediante particion de prueba
    Clasificacion=zeros([Long_X_Test,1]);

    for i=1:Long_X_Test
        Clasificacion(i)=predict(Clasificador_R1,X_Test(i,:));
        if Clasificacion(i)==1
        else
            Clasificacion(i)=predict(Clasificador_R2,X_Test(i,:));
            if Clasificacion(i)==1
                Clasificacion(i)=2;
            else
                Clasificacion(i)=predict(Clasificador_R3,X_Test(i,:));
                if Clasificacion(i)==1
                    Clasificacion(i)=3;
                else
                    Clasificacion(i)=predict(Clasificador_R4,X_Test(i,:));
                    if Clasificacion(i)==1
                        Clasificacion(i)=4;
                    else
                        Clasificacion(i)=5;
                    end
                end
            end
        end
    end

elseif Metodo==2
% Clasificador Adaboost Multiclase (todas las clases)

    % Entrenamiento del clasificador
    Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,'AdaBoostM2',
    NLearn_Multi,t,'LearnRate',LearnRate);
    % Cross Validation
    Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',NLearn_Multi,
    t,'LearnRate',LearnRate);
    Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
    'KFold',KFold);

    % validacion mediante particion de prueba
    Clasificacion=predict(Clasificador_AdaboostM2,X_Test);
```

```
elseif Metodo==3
% Aplicacion de clasificador Adaboost Biclase para R1vsAll, y Adaboost
% Multiclase para el resto de clases

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
    Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
    'AdaBoostM1',NLearn_R1,t,'LearnRate',LearnRate);

    if isempty(Clasificador_R1.TrainedWeights)
        Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
        Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
        display('Error de clasificacion 0 Clasificador R1vsAll')
    else
        Error_Clasif_AdaboostM1(1)=1;
        Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
        NLearn_R1,t,'LearnRate',LearnRate);
    end
    % Cross Validation
    Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

    % Clasificador Multiclase
    Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,
    'AdaBoostM2',NLearn_Multi,t,'LearnRate',LearnRate);
    % Cross Validation
    Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',NLearn_Multi,
    t,'LearnRate',LearnRate);
    Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
    'KFold',KFold);

    % Cascada de clasificadores. Validacion mediante particion de prueba
    Clasificacion=zeros([Long_X_Test,1]);

    for i=1:Long_X_Test
        Clasificacion(i)=predict(Clasificador_R1,X_Test(i,:));
        if Clasificacion(i)==1
        else
            Clasificacion(i)=predict(Clasificador_AdaboostM2,X_Test(i,:));
        end
    end

elseif Metodo==4
% Aplicacion de clasificador Adaboost Biclase para R1vsAll y R2vsR3R4R5 y
% Adaboost Multiclase para el resto de clases

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
    Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
    'AdaBoostM1',NLearn_R1,t,'LearnRate',LearnRate);

    if isempty(Clasificador_R1.TrainedWeights)
        Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
        Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
        display('Error de clasificacion 0 Clasificador R1vsAll')
    else
        Error_Clasif_AdaboostM1(1)=1;
        Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
        NLearn_R1,t,'LearnRate',LearnRate);
    end
```

```
    % Cross Validation
    Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

    % Clasificador R2vsR3R4R5
    Y_Training_Clasif_Binario_R2=(Y_Training==2);
    Clasificador_R2=fitensemble(X_Training,Y_Training_Clasif_Binario_R2,
    'AdaBoostM1',NLearn_R2,t,'LearnRate',LearnRate);

    if isempty(Clasificador_R2.TrainedWeights)
        Clasificador_R2=fitctree(X_Training,Y_Training_Clasif_Binario_R2);
        Clasificador_Aux_R2=fitctree(X,Y_Binario(2,:));
        display('Error de clasificacion 0 Clasificador R2vsR3R4R5')
    else
        Error_Clasif_AdaboostM1(2)=1;
        Clasificador_Aux_R2=fitensemble(X,Y_Binario(2,:),'AdaBoostM1',
        NLearn_R2,t,'LearnRate',LearnRate);
    end
    % Cross Validation
    Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);

    % Clasificador Multiclase
    Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,
    'AdaBoostM2',NLearn_Multi,t,'LearnRate',LearnRate);
    % Cross Validation
    Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',NLearn_Multi,
    t,'LearnRate',LearnRate);
    Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
    'KFold',KFold);

    % Cascada de clasificadores. Validacion mediante particion de prueba
    Clasificacion=zeros([Long_X_Test,1]);

    for i=1:Long_X_Test
        Clasificacion(i)=predict(Clasificador_R1,X_Test(i,:));
        if Clasificacion(i)==1
        else
            Clasificacion(i)=predict(Clasificador_R2,X_Test(i,:));
            if Clasificacion(i)==1
                Clasificacion(i)=2;
            else
                Clasificacion(i)=predict(Clasificador_AdaboostM2,
                X_Test(i,:));
            end
        end
    end

elseif Metodo==5||Metodo==6
% Metodo 5: Aplicacion de todos los clasificadores binarios, multiclase
% para desempatar resultados

% Metodo 6: Aplicacion de todos los clasificadores binarios y multiclase,
% primando redultados correspondientes a la clase 1

    % Clasificador R1vsAll
    Y_Training_Clasif_Binario_R1=(Y_Training==1);
    Clasificador_R1=fitensemble(X_Training,Y_Training_Clasif_Binario_R1,
    'AdaBoostM1',NLearn_R1,t,'LearnRate',LearnRate);
    if isempty(Clasificador_R1.TrainedWeights)
        Clasificador_R1=fitctree(X_Training,Y_Training_Clasif_Binario_R1);
        Clasificador_Aux_R1=fitctree(X,Y_Binario(1,:));
        display('Error de clasificacion 0 Clasificador R1vsAll')
```

```
else
    Error_Clasif_AdaboostM1(1)=1;
    Clasificador_Aux_R1=fitensemble(X,Y_Binario(1,:),'AdaBoostM1',
    NLearn_R1,t,'LearnRate',LearnRate);
end
% Cross Validation
Clasificador_CV_R1=crossval(Clasificador_Aux_R1,'KFold',KFold);

% Clasificador R2vsAll
Y_Training_Clasif_Binario_R2=(Y_Training==2);
Clasificador_R2=fitensemble(X_Training,Y_Training_Clasif_Binario_R2,
'AdaBoostM1',NLearn_R2,t,'LearnRate',LearnRate);
if isempty(Clasificador_R2.TrainedWeights)
    Clasificador_R2=fitctree(X_Training,Y_Training_Clasif_Binario_R2);
    Clasificador_Aux_R2=fitctree(X,Y_Binario(2,:));
    display('Error de clasificacion 0 Clasificador R2vsAll')
else
    Error_Clasif_AdaboostM1(2)=1;
    Clasificador_Aux_R2=fitensemble(X,Y_Binario(2,:),'AdaBoostM1',
    NLearn_R2,t,'LearnRate',LearnRate);
end
% Cross Validation
Clasificador_CV_R2=crossval(Clasificador_Aux_R2,'KFold',KFold);

% Clasificador R3vsAll
Y_Training_Clasif_Binario_R3=(Y_Training==3);
Clasificador_R3=fitensemble(X_Training,Y_Training_Clasif_Binario_R3,
'AdaBoostM1',NLearn_R3,t,'LearnRate',LearnRate);
if isempty(Clasificador_R3.TrainedWeights)
    Clasificador_R3=fitctree(X_Training,Y_Training_Clasif_Binario_R3);
    Clasificador_Aux_R3=fitctree(X,Y_Binario(3,:));
    display('Error de clasificacion 0 Clasificador R3vsAll')
else
    Error_Clasif_AdaboostM1(3)=1;
    Clasificador_Aux_R3=fitensemble(X,Y_Binario(3,:),'AdaBoostM1',
    NLearn_R3,t,'LearnRate',LearnRate);
end
% Cross Validation
Clasificador_CV_R3=crossval(Clasificador_Aux_R3,'KFold',KFold);

% Clasificador R4vsAll
Y_Training_Clasif_Binario_R4=(Y_Training==4);
Clasificador_R4=fitensemble(X_Training,Y_Training_Clasif_Binario_R4,
'AdaBoostM1',NLearn_R4,t,'LearnRate',LearnRate);
if isempty(Clasificador_R4.TrainedWeights)
    Clasificador_R4=fitctree(X_Training,Y_Training_Clasif_Binario_R4);
    Clasificador_Aux_R4=fitctree(X,Y_Binario(4,:));
    display('Error de clasificacion 0 Clasificador R4vsAll')
else
    Error_Clasif_AdaboostM1(4)=1;
    Clasificador_Aux_R4=fitensemble(X,Y_Binario(4,:),'AdaBoostM1',
    NLearn_R4,t,'LearnRate',LearnRate);
end
% Cross Validation
Clasificador_CV_R4=crossval(Clasificador_Aux_R4,'KFold',KFold);

% Clasificador R5vsAll
Y_Training_Clasif_Binario_R5=(Y_Training==5);
    Clasificador_R5=fitensemble(X_Training,Y_Training_Clasif_Binario_R5
    , 'AdaBoostM1',NLearn_R5,t,'LearnRate',LearnRate);
```

```
if isempty(Clasificador_R5.TrainedWeights)
    Clasificador_R5=fitctree(X_Training,Y_Training_Clasif_Binario_R5);
    Clasificador_Aux_R5=fitctree(X,Y_Binario(5,:));
    display('Error de clasificacion 0 Clasificador R5vsAll')
else
    Error_Clasif_AdaboostM1(5)=1;
    Clasificador_Aux_R5=fitensemble(X,Y_Binario(5,:),'AdaBoostM1',
    NLearn_R5,t,'LearnRate',LearnRate);
end
% Cross Validation
Clasificador_CV_R5=crossval(Clasificador_Aux_R5,'KFold',KFold);

% Clasificador Multiclase
Clasificador_AdaboostM2=fitensemble(X_Training,Y_Training,'AdaBoostM2',
NLearn_Multi,t,'LearnRate',LearnRate);
% Cross Validation
Clasificador_AdaboostM2_Aux=fitensemble(X,Y,'AdaBoostM2',NLearn_Multi
,t,'LearnRate',LearnRate);
Clasificador_AdaboostM2_CV=crossval(Clasificador_AdaboostM2_Aux,
'KFold',KFold);

% Algoritmo de determinacion de clases
Clasificacion_Tot=zeros([6,Long_X_Test]);

Clasificacion_Tot(1,:)=predict(Clasificador_R1,X_Test);
Clasificacion_Tot(2,:)=predict(Clasificador_R2,X_Test);
Clasificacion_Tot(3,:)=predict(Clasificador_R3,X_Test);
Clasificacion_Tot(4,:)=predict(Clasificador_R4,X_Test);
Clasificacion_Tot(5,:)=predict(Clasificador_R5,X_Test);

Clasificacion=zeros([Long_X_Test,1]);

if Metodo==5

    for i=1:Long_X_Test
        if sum(Clasificacion_Tot(:,i))==1
            for k=1:Num_Clases
                if (Clasificacion_Tot(k,i))==1
                    Clasificacion(i)=k;
                else
                end
            end
        else
            Clasificacion_Tot(6,i)=predict(Clasificador_AdaboostM2,
            X_Test(i,:));
            Clasificacion(i)=Clasificacion_Tot(6,i);
        end
    end
else
    Clasificacion_Tot(6,:)=predict(Clasificador_AdaboostM2,X_Test);

    for i=1:Long_X_Test
        if (Clasificacion_Tot(1,i)==0)||(Clasificacion_Tot(6,i)~=1)
            if (sum(Clasificacion_Tot([2 3 4 5],i)>0))==0
                if (Clasificacion_Tot(6,i)~=1)
                    Clasificacion(i)=Clasificacion_Tot(6,i);
                else
                    Clasificacion(i)=4;
                end
            elseif (sum(Clasificacion_Tot([2 3 4 5],i)>0))>1
                Clasificacion(i)=Clasificacion_Tot(6,i);
```

```
                else
                    Clasificacion(i)=1+find(Clasificacion_Tot([2 3 4
                    5],i)>0);
                end
            else
                Clasificacion(i)=1;
            end
        end
    end
else
    display('Introduzca un valor para seleccionar el método de clasificación
    comprendido entre 0 y 4')
    Error_Clasif_AdaboostM1=9999; % Valor de control, por si el usuario ha
    elegido un método inexistente
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     ERROR DE RESISTITUCION Y ERROR DE CLASIFICACION MEDIANTE CV
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if Metodo==1
    RsLoss_R1=resubLoss(Clasificador_R1);
    GenError_R1=kfoldLoss(Clasificador_CV_R1);
    RsLoss_R2=resubLoss(Clasificador_R2);
    GenError_R2=kfoldLoss(Clasificador_CV_R2);
    RsLoss_R3=resubLoss(Clasificador_R3);
    GenError_R3=kfoldLoss(Clasificador_CV_R3);
    RsLoss_R4=resubLoss(Clasificador_R4);
    GenError_R4=kfoldLoss(Clasificador_CV_R4);

    GenError=100*[GenError_R1 GenError_R2 GenError_R3 GenError_R4];
    RsLoss=100*[RsLoss_R1 RsLoss_R2 RsLoss_R3 RsLoss_R4];

    figure;
    subplot(1,2,1);
    bar(RsLoss)
    Labels={'Clasificador R1','Clasificador R2','Clasificador R3',
    'Clasificador R4'};
    set(gca,'XTick', 1:4,'XTickLabel',Labels)
    title('\fontsize{14}Resubstitution Classification Cost');
    ylabel('\fontsize{13}Loss (%)');

    subplot(1,2,2);
    bar(GenError)
    Labels={'Clasificador R1','Clasificador R2','Clasificador R3',
    'Clasificador R4'};
    set(gca,'XTick', 1:4,'XTickLabel',Labels)
    title('\fontsize{14}Cross-Validation Loss');
    ylabel('\fontsize{13}Loss (%)');

elseif Metodo==2
    RsLoss_Multiclase=100*resubLoss(Clasificador_AdaboostM2);
    GenError_Multiclase=100*kfoldLoss(Clasificador_AdaboostM2_CV);
    Error=[RsLoss_Multiclase RsLoss_Multiclase];

    figure;
    bar(Error);
    Labels={'Resubstitution Classification Cost','Cross-Validation Loss'};
    set(gca,'XTick', 1:2,'XTickLabel',Labels)
    title('\fontsize{14}Clasificador Multiclase');
    ylabel('\fontsize{13}Loss (%)');
```

```
    ylabel('\fontsize{20}');

elseif Metodo==3
    RsLoss_R1=resubLoss(Clasificador_R1);
    GenError_R1=kfoldLoss(Clasificador_CV_R1);
    RsLoss_Multiclase=resubLoss(Clasificador_AdaboostM2);
    GenError_Multiclase=kfoldLoss(Clasificador_AdaboostM2_CV);

    GenError=100*[GenError_R1 GenError_Multiclase];
    RsLoss=100*[RsLoss_R1 RsLoss_Multiclase];

    figure;
    subplot(1,2,1);
    bar(RsLoss)
    Labels={'Clasificador R1','Clasificador Multi'};
    set(gca,'XTick', 1:2,'XTickLabel',Labels)
    title('\fontsize{14}Resubstitution Classification Cost');
    ylabel('\fontsize{13}Loss (%)');

    subplot(1,2,2);
    bar(GenError)
    Labels={'Clasificador R1','Clasificador Multi'};
    set(gca,'XTick', 1:2,'XTickLabel',Labels)
    title('\fontsize{14}Cross-Validation Loss');
    ylabel('\fontsize{13}Loss (%)');

elseif Metodo==4
    RsLoss_R1=resubLoss(Clasificador_R1);
    GenError_R1=kfoldLoss(Clasificador_CV_R1);
    RsLoss_R2=resubLoss(Clasificador_R2);
    GenError_R2=kfoldLoss(Clasificador_CV_R2);
    RsLoss_Multiclase=resubLoss(Clasificador_AdaboostM2);
    GenError_Multiclase=kfoldLoss(Clasificador_AdaboostM2_CV);

    GenError=100*[GenError_R1 GenError_R2 GenError_Multiclase];
    RsLoss=100*[RsLoss_R1 RsLoss_R2 RsLoss_Multiclase];

    figure;
    subplot(1,2,1);
    bar(RsLoss)
    Labels={'Clasificador R1','Clasificador R2','Clasificador Multi'};
    set(gca,'XTick', 1:3,'XTickLabel',Labels)
    title('\fontsize{14}Resubstitution Classification Cost');
    ylabel('\fontsize{13}Loss (%)');

    subplot(1,2,2);
    bar(GenError)
    Labels={'Clasificador R1','Clasificador R2','Clasificador Multi'};
    set(gca,'XTick', 1:3,'XTickLabel',Labels)
    title('\fontsize{14}Cross-Validation Loss');
    ylabel('\fontsize{13}Loss (%)');

elseif Metodo==5||Metodo==6
    RsLoss_R1=resubLoss(Clasificador_R1);
    GenError_R1=kfoldLoss(Clasificador_CV_R1);
    RsLoss_R2=resubLoss(Clasificador_R2);
    GenError_R2=kfoldLoss(Clasificador_CV_R2);
    RsLoss_R3=resubLoss(Clasificador_R3);
    GenError_R3=kfoldLoss(Clasificador_CV_R3);
    RsLoss_R4=resubLoss(Clasificador_R4);
```

```
    GenError_R4=kfoldLoss(Clasificador_CV_R4);
    RsLoss_R5=resubLoss(Clasificador_R5);
    GenError_R5=kfoldLoss(Clasificador_CV_R5);
    RsLoss_Multiclase=resubLoss(Clasificador_AdaboostM2);
    GenError_Multiclase=kfoldLoss(Clasificador_AdaboostM2_CV);

   GenError=100*[GenError_R1 GenError_R2 GenError_R3 GenError_R4
   GenError_R5 GenError_Multiclase];
   RsLoss=100*[RsLoss_R1 RsLoss_R2 RsLoss_R3 RsLoss_R4 RsLoss_R5
   RsLoss_Multiclase];

    figure;
    subplot(1,2,1);
    bar(RsLoss)
   Labels={'Clasificador R1','Clasificador R2','Clasificador R3',
   'Clasificador R4','Clasificador R5','Clasificador Multi'};
    set(gca,'XTick', 1:6,'XTickLabel',Labels)
    title('\fontsize{14}Resubstitution Classification Cost');
    ylabel('\fontsize{13}Loss (%)');

    subplot(1,2,2);
    bar(GenError)
   Labels={'Clasificador R1','Clasificador R2','Clasificador R3',
   'Clasificador R4','Clasificador R5','Clasificador Multi'};
    set(gca,'XTick', 1:6,'XTickLabel',Labels)
    title('\fontsize{14}Cross-Validation Loss');
    ylabel('\fontsize{13}Loss (%)');
else
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    MATRIZ DE CONFUSION Y CURVA CARACTERISTICA OPERATIVA DEL RECEPTOR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if Error_Clasif_AdaboostM1==9999
else
    [Confusion_Matrix,order]=confusionmat(Y_Test,Clasificacion);

    % Preparamos los datos test para representar la matriz de confusion
    Y_Clasificacion_Mat_Conf=zeros(Num_Clases,Long_X_Test);
    for i=1:Num_Clases
        Y_Clasificacion_Mat_Conf(i,(Clasificacion==i))=1;
    end

    % Representamos la matriz de confusion
    figure
    plotconfusion(Y_Test_Mat_Conf,Y_Clasificacion_Mat_Conf)

    % Representamos la curva caracteristica operativa del receptor
    figure
    plotroc(Y_Test_Mat_Conf,Y_Clasificacion_Mat_Conf)
end
```

2 GRÁFICOS OBTENIDOS DURANTE LOS ENSAYOS

2.1 Clasificadores Binarios

2.1.1 Introducción

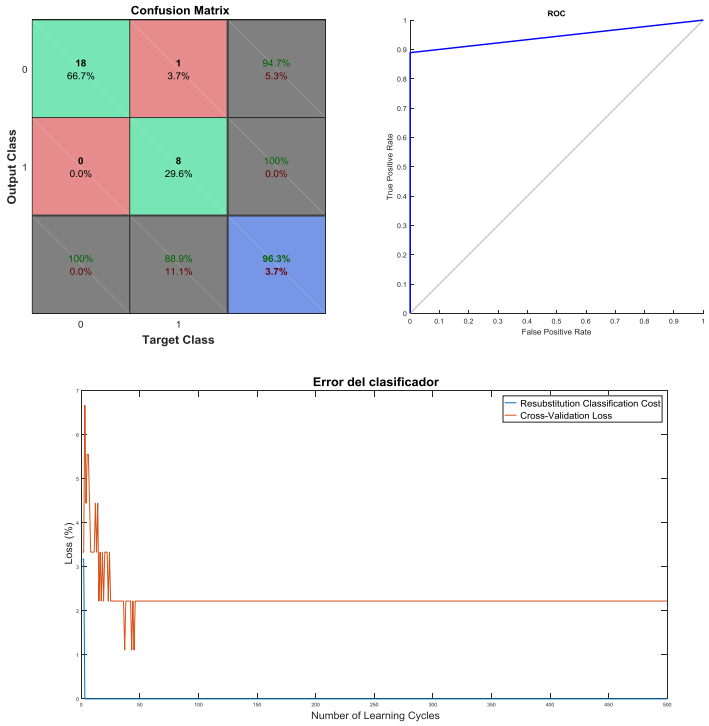


Figura A2-1: Ensayo con 500 clasificadores débiles y 20 subgrupos para validación cruzada. Clasificación binaria clase R1.

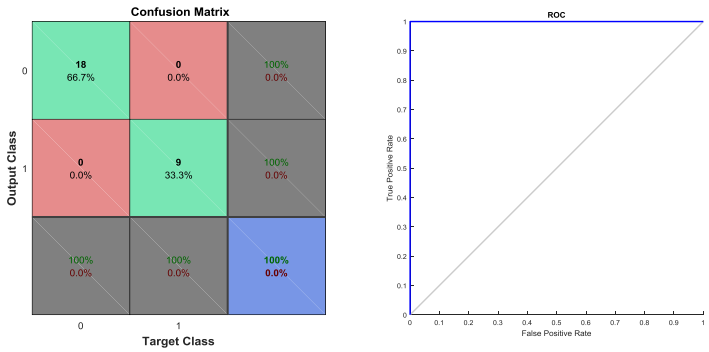


Figura A2-2:Ensayo con 50 clasificadores débiles y 20 subgrupos para validación cruzada. Clasificación binaria clase R1.

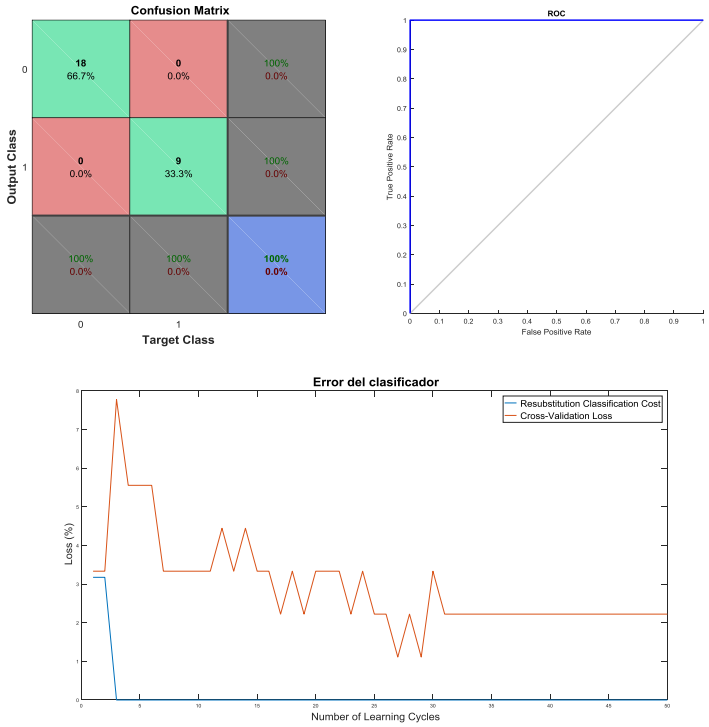


Figura A2-3: Ensayo con 50 clasificadores débiles y 80 subgrupos para validación cruzada. Clasificación binaria clase R1.

2.1.2 Clasificador Binario para clase R1

2.1.2.1 Ensayos para alimentación de Red

- **Nivel de carga 1**

Figura A2-2.

- ## Nivel de carga 2

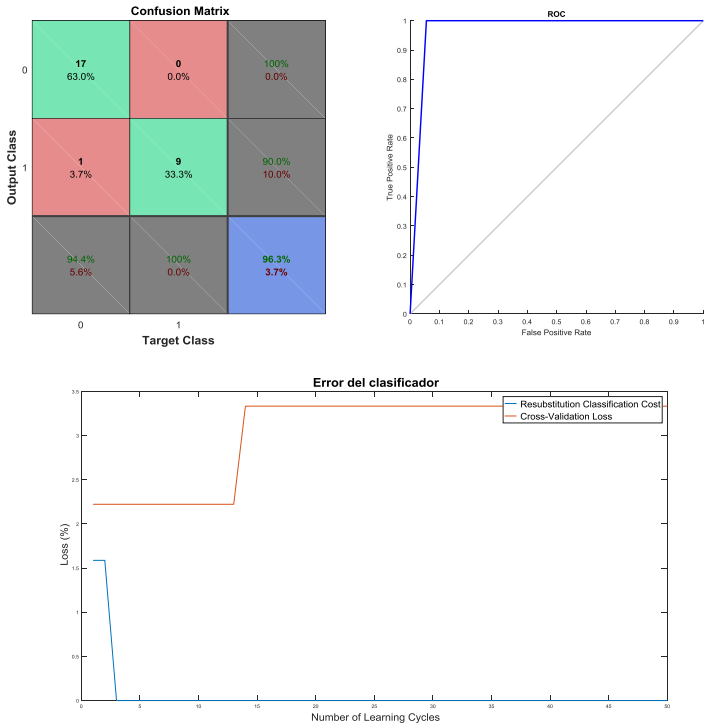


Figura A2-4: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación de Red. Clasificación binaria clase R1.

- ## Cualquier nivel de carga

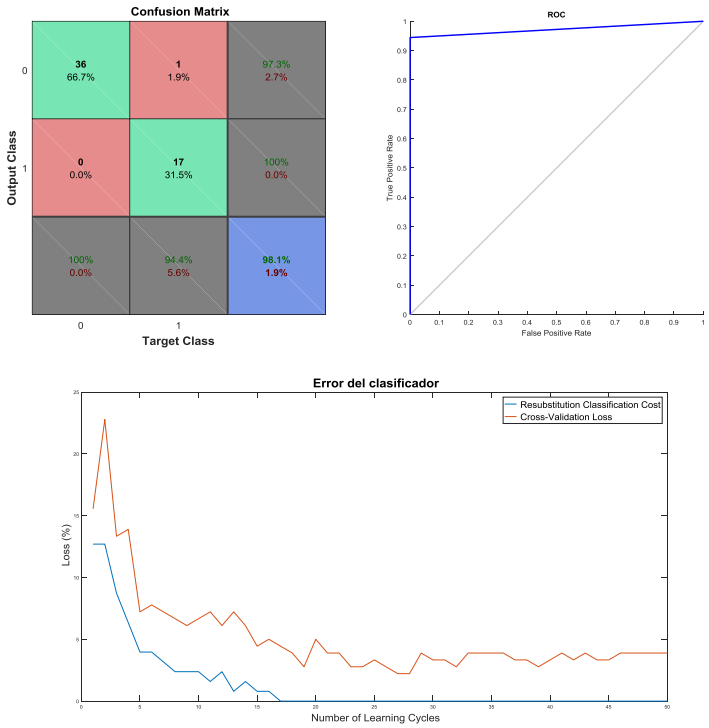


Figura A2-5: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación de Red. Clasificación binaria clase R1.

2.1.2.2 Ensayos para alimentación mediante convertidor

- **Nivel de carga 1**

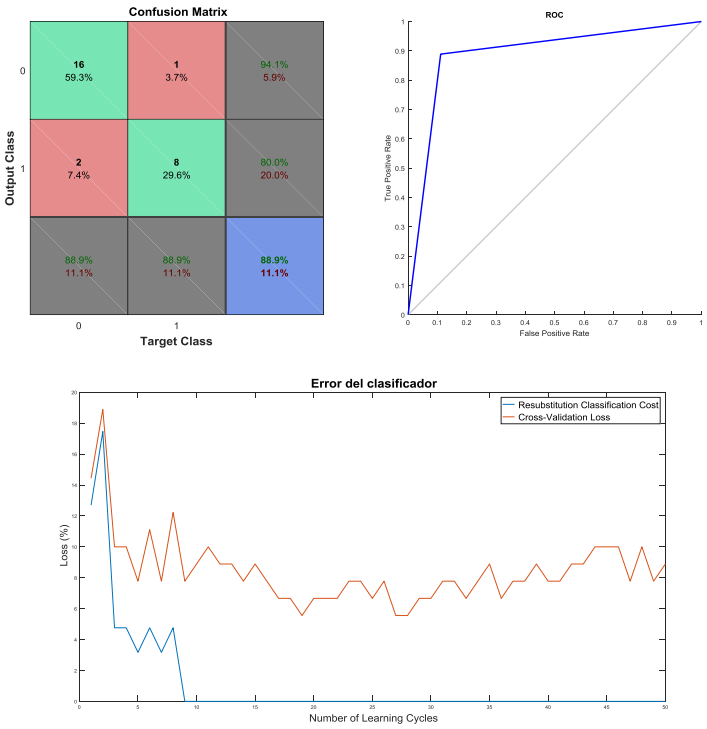


Figura A2-6: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación mediante Convertidor. Clasificación binaria clase R1.

- **Nivel de carga 2**

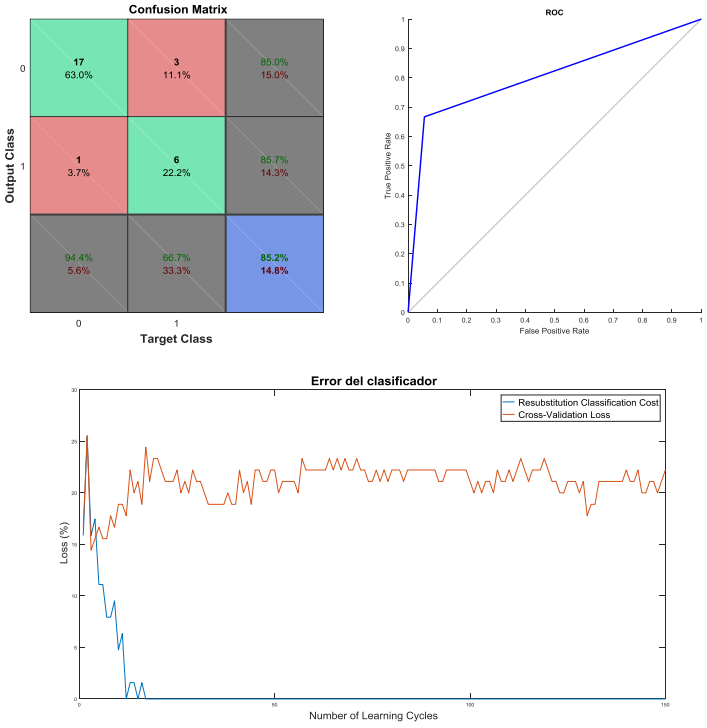


Figura A2-7: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación mediante Convertidor. Clasificación binaria clase R1.

- **Cualquier nivel de carga**

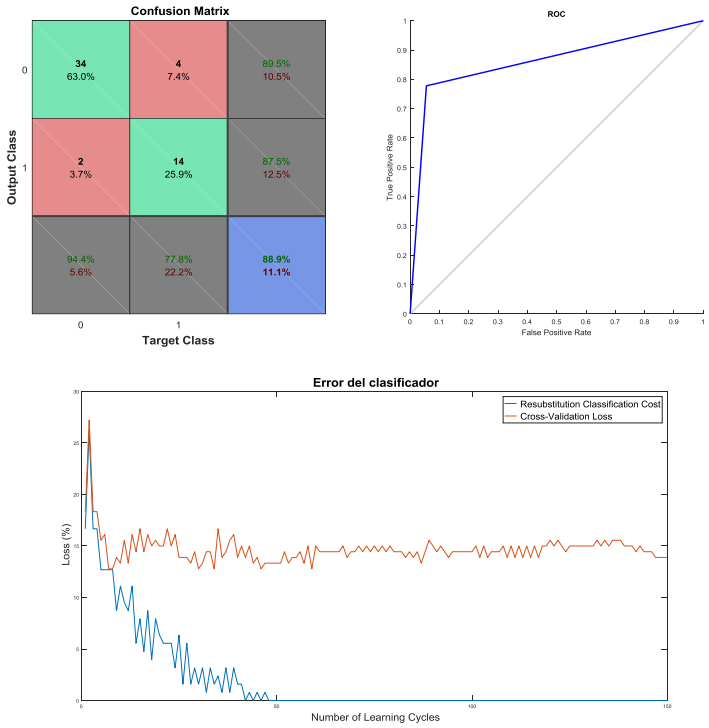


Figura A2-8: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación mediante Convertidor. Clasificación binaria clase R1.

2.1.2.3 Ensayos para cualquier tipo de alimentación

- **Nivel de carga 1**

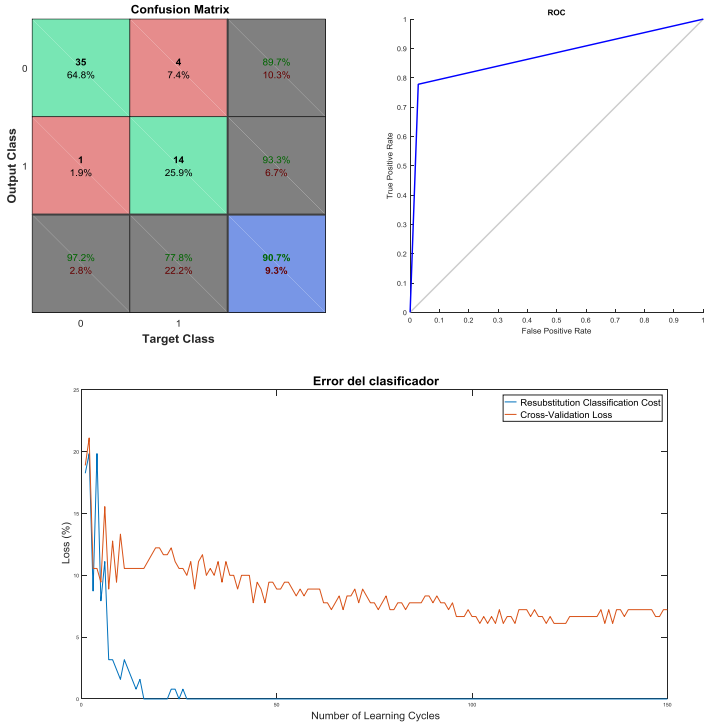


Figura A2-9: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y cualquier tipo de alimentación. Clasificación binaria clase R1.

- ## Nivel de carga 2

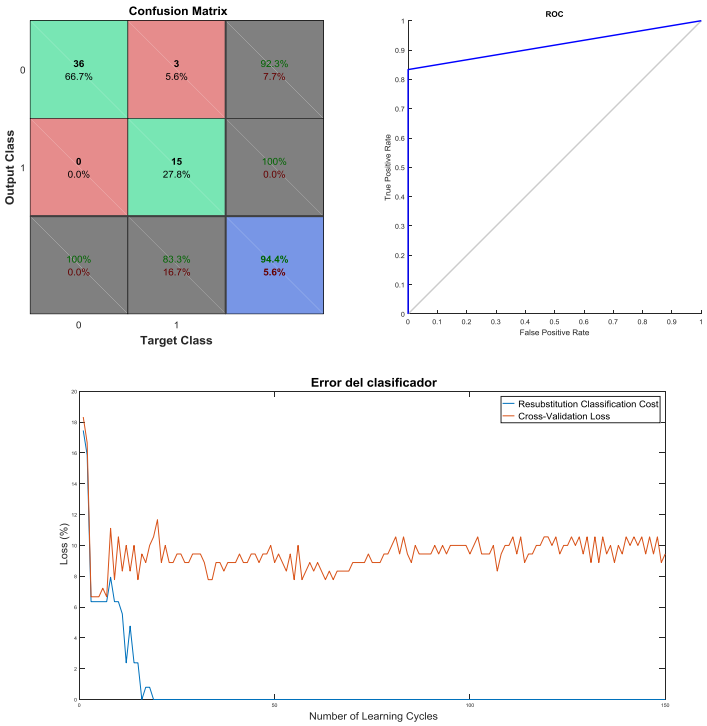


Figura A2-10: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y cualquier tipo de alimentación. Clasificación binaria clase R1.

- ## Cualquier nivel de carga

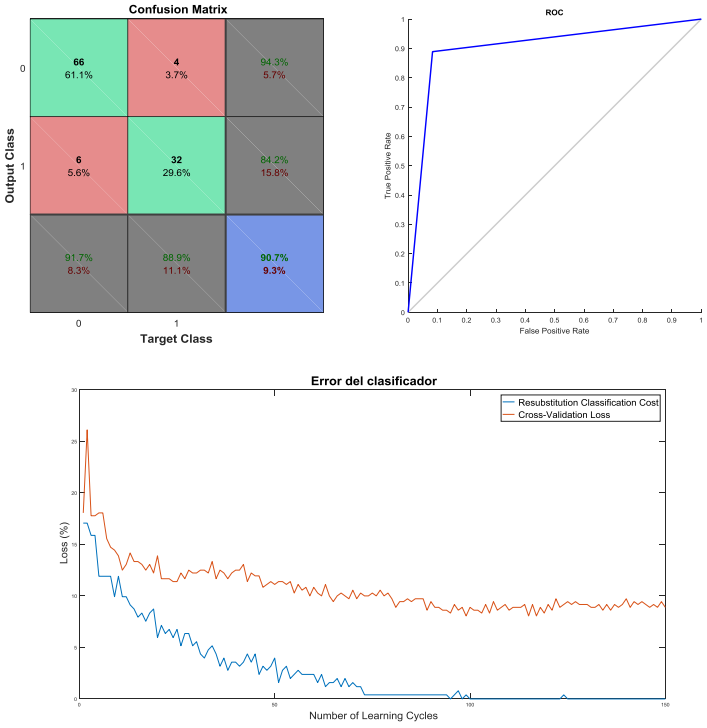


Figura A2-11: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y cualquier tipo de alimentación. Clasificación binaria clase R1.

2.1.3 Clasificador Binario para clase R2

2.1.3.1 Ensayos para alimentación de Red

- **Nivel de carga 1**

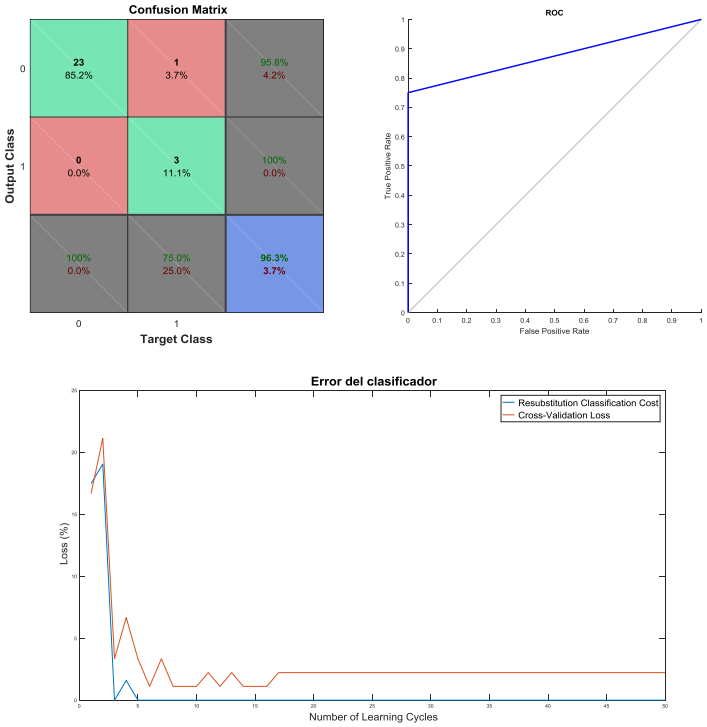


Figura A2-12: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación de Red. Clasificación binaria clase R2.

- **Nivel de carga 2**

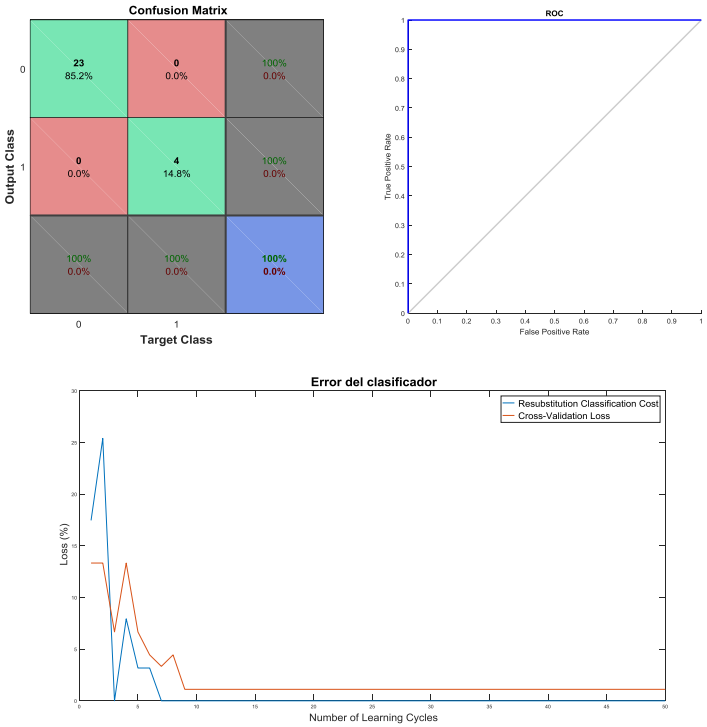


Figura A2-13: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y

alimentación de Red. Clasificación binaria clase R2.

- Cualquier nivel de carga

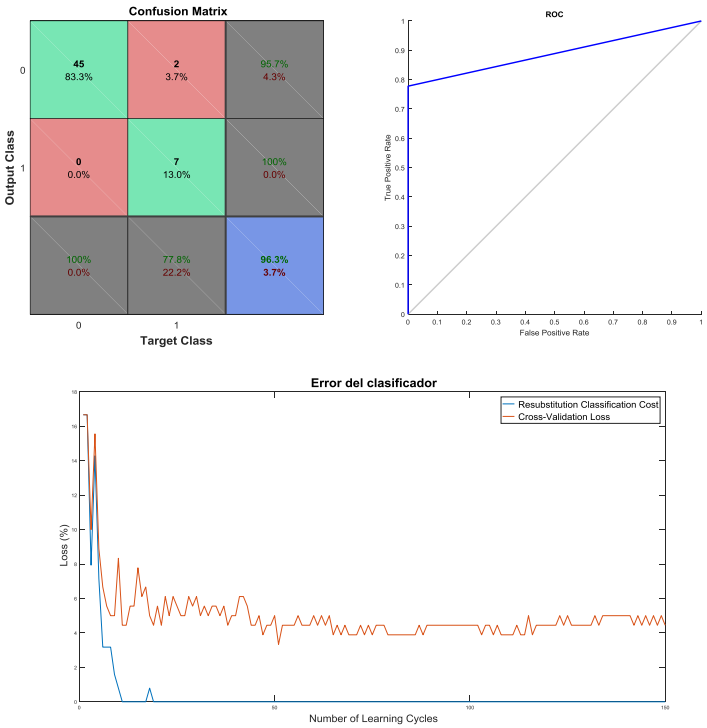


Figura A2-14: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación de Red. Clasificación binaria clase R2.

2.1.3.2 *Ensayos para alimentación mediante convertidor*

- Nivel de carga 1

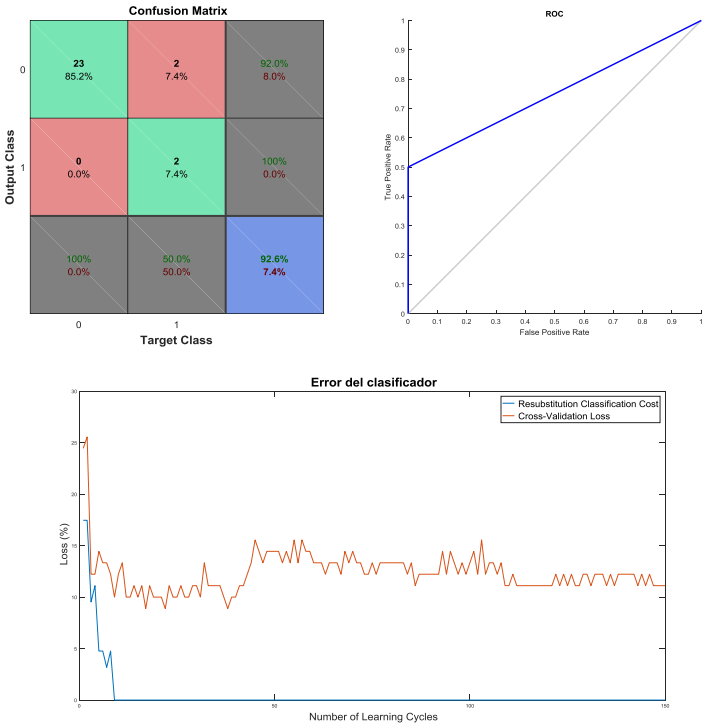


Figura A2-15: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y

alimentación mediante convertidor. Clasificación binaria clase R2.

- **Nivel de carga 2**

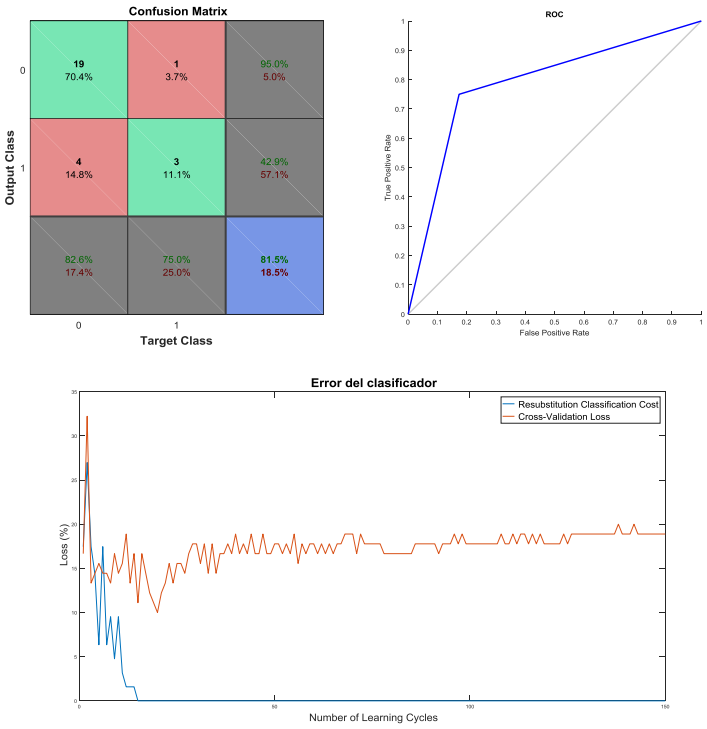


Figura A2-16: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación mediante convertidor. Clasificación binaria clase R2.

- **Cualquier nivel de carga**

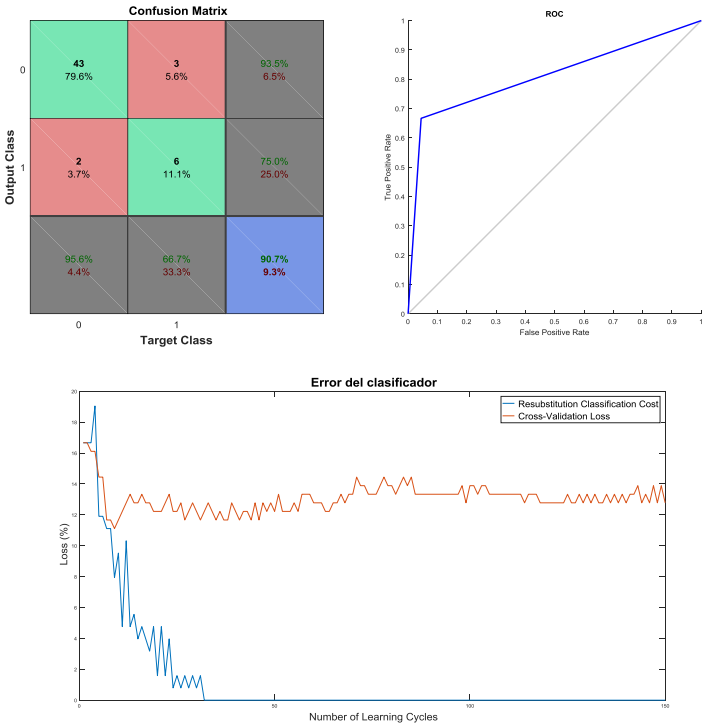


Figura A2-17: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga

y alimentación mediante convertidor. Clasificación binaria clase R2.

2.1.3.3 *Ensayos para cualquier tipo de alimentación*

- **Nivel de carga 1**

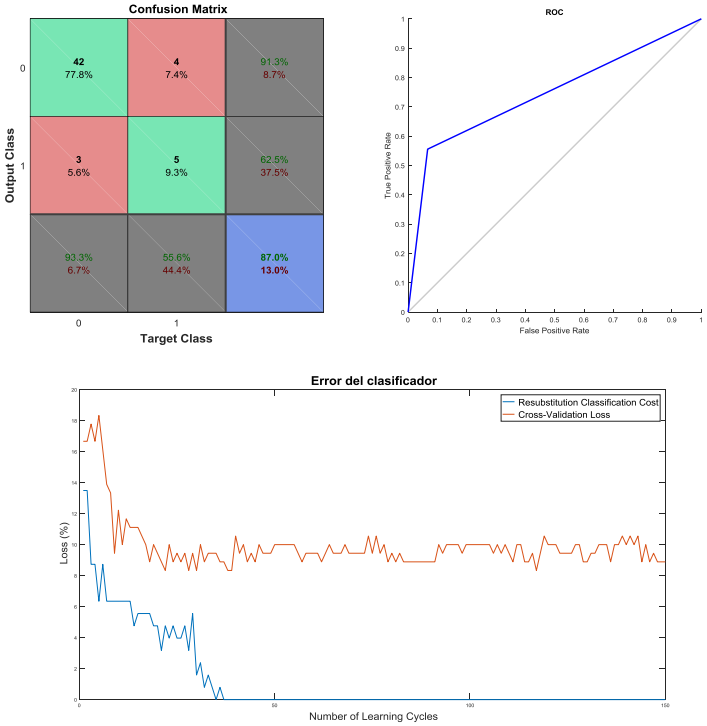


Figura A2-18: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y cualquier tipo de alimentación. Clasificación binaria clase R2.

- **Nivel de carga 2**

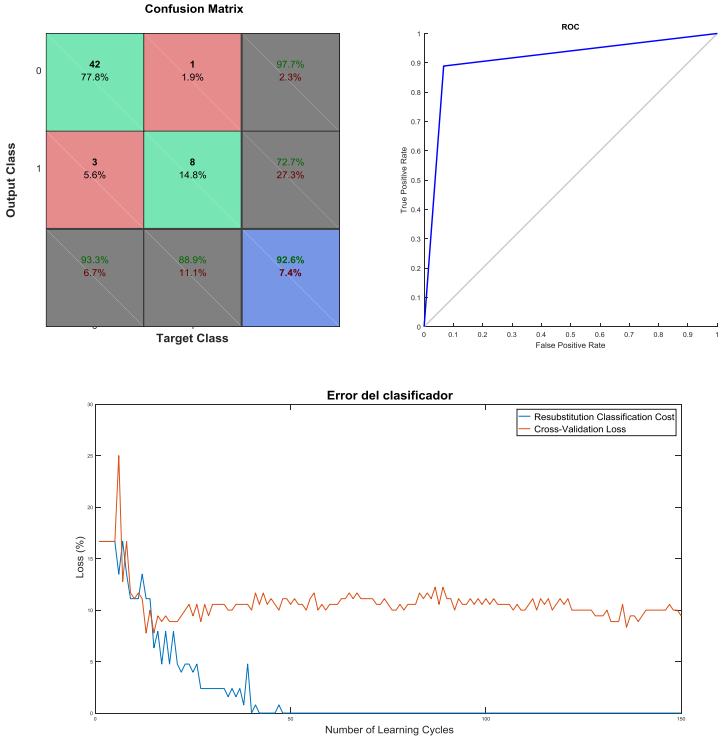


Figura A2-19: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y

cualquier tipo de alimentación. Clasificación binaria clase R2.

- **Cualquier nivel de carga**

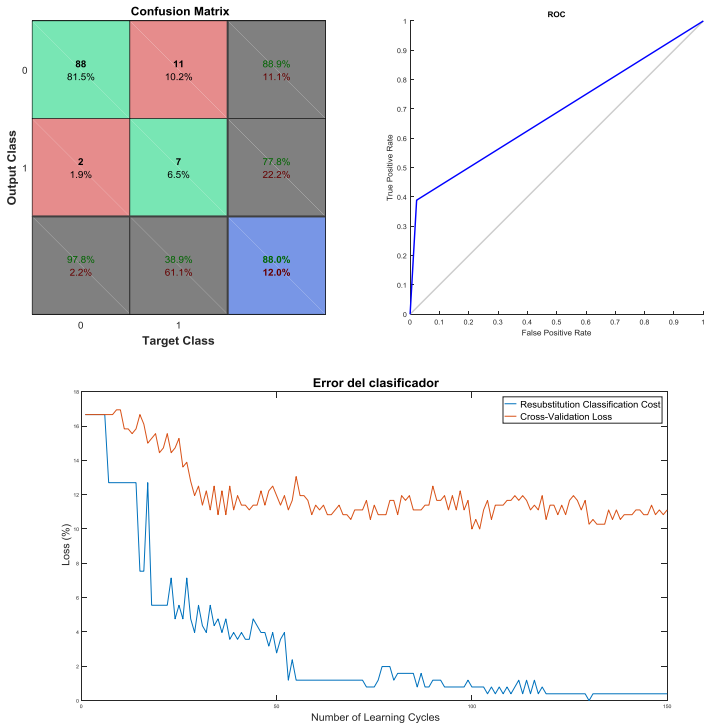
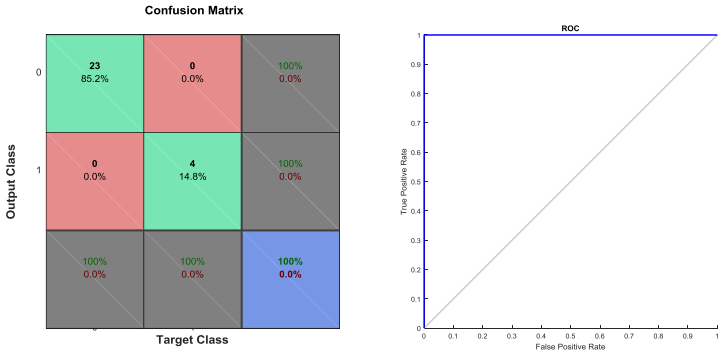


Figura A2-20: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y cualquier tipo de alimentación. Clasificación binaria clase R2.

2.1.4 Clasificador Binario para clase R3

2.1.4.1 Ensayos para alimentación de Red

- **Nivel de carga 1**



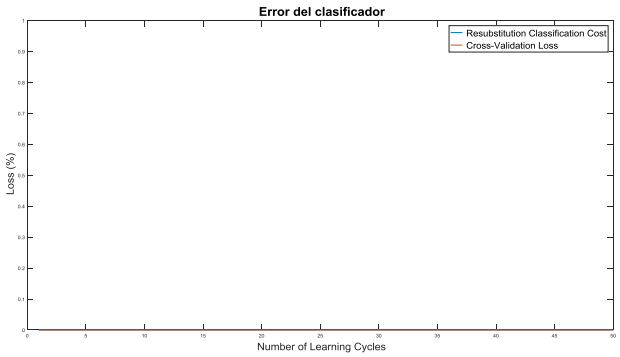
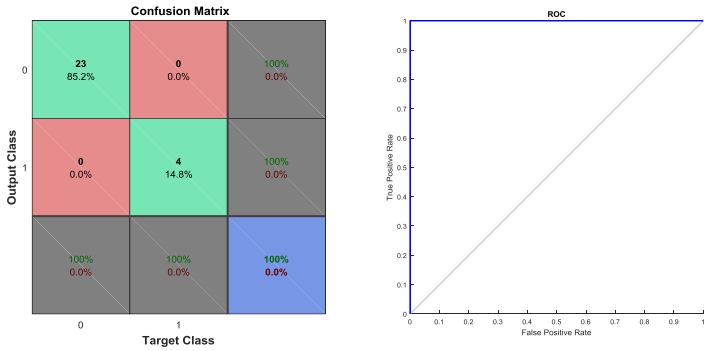


Figura A2-21: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación de Red. Clasificación binaria clase R3.

- ## Nivel de carga 2



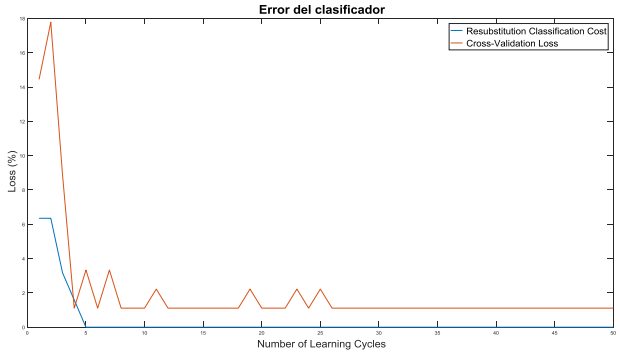
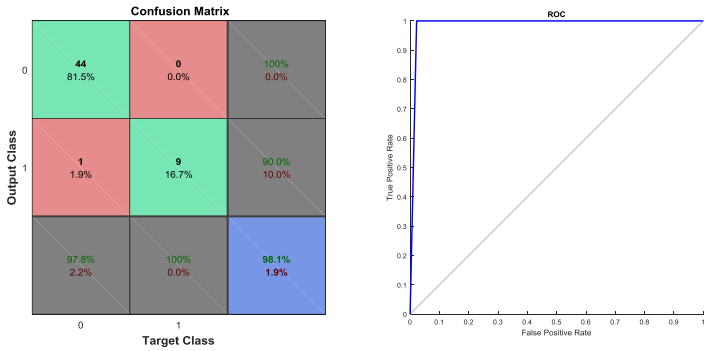


Figura A2-22: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación de Red. Clasificación binaria clase R3.

- ## Cualquier nivel de carga



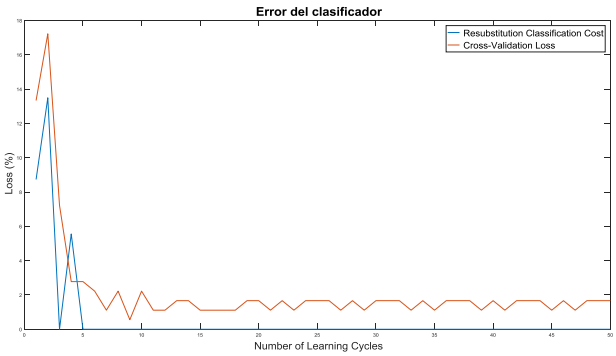


Figura A2-23: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación de Red. Clasificación binaria clase R3.

2.1.4.2 Ensayos para alimentación mediante convertidor

- **Nivel de carga 1**

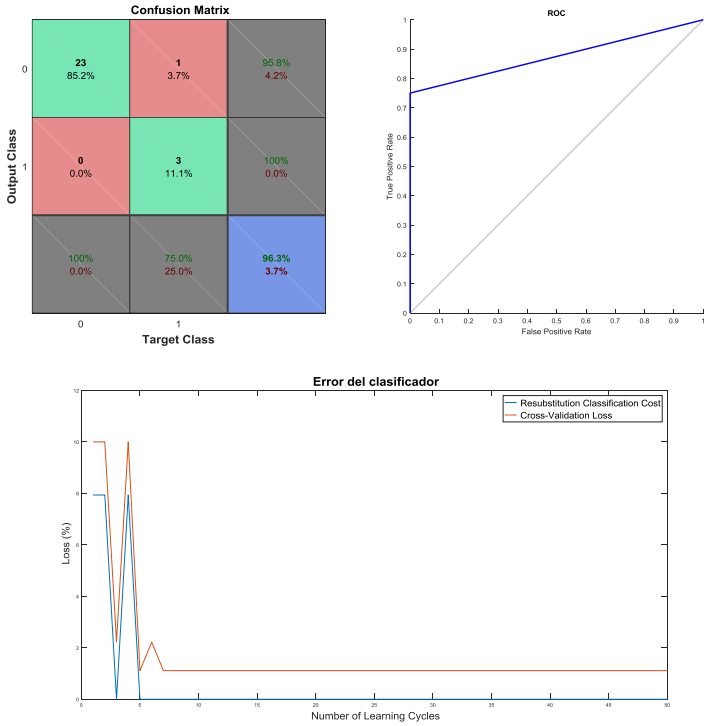
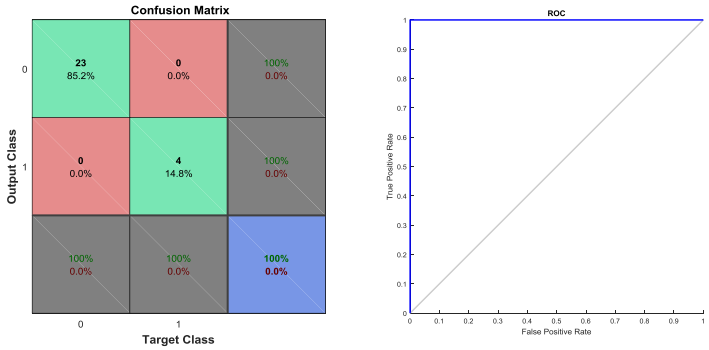


Figura A2-24: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación mediante convertidor. Clasificación binaria clase R3.

- **Nivel de carga 2**



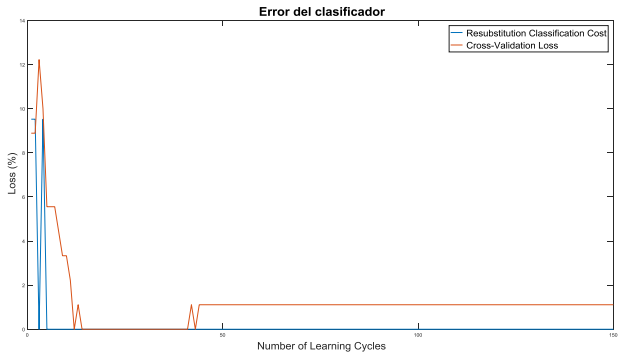


Figura A2-25: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación mediante convertidor. Clasificación binaria clase R3.

- **Cualquier nivel de carga**

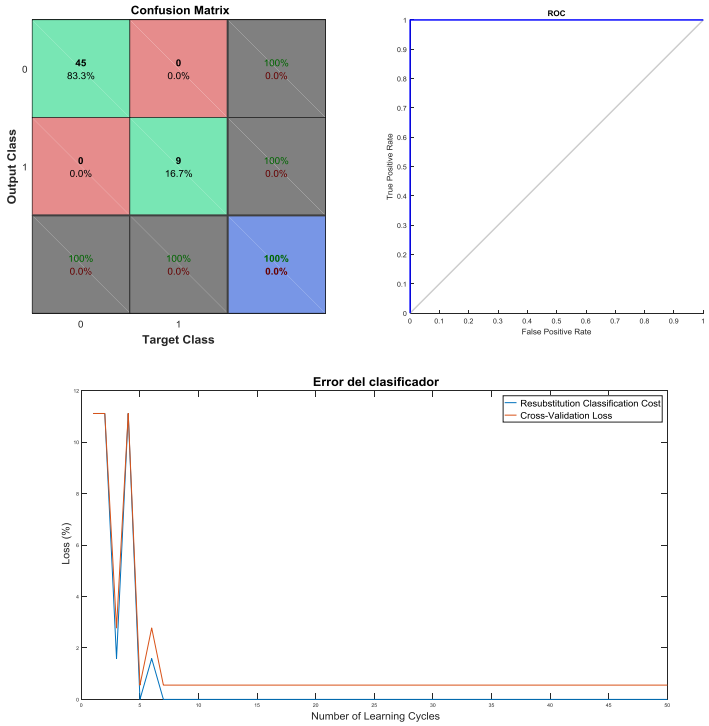
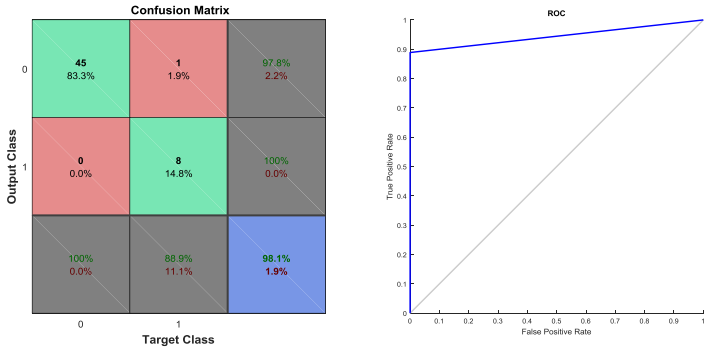


Figura A2-26: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación mediante convertidor. Clasificación binaria clase R3.

2.1.4.3 Ensayos para cualquier tipo de alimentación

- **Nivel de carga 1**



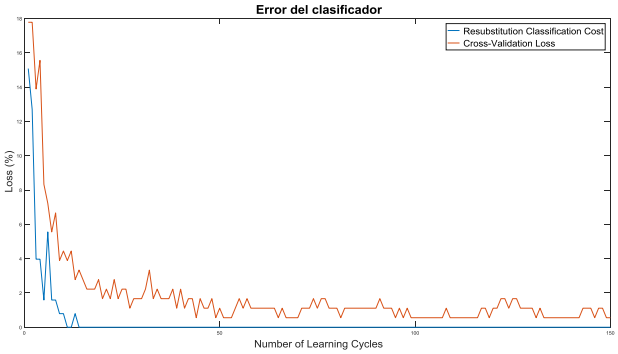
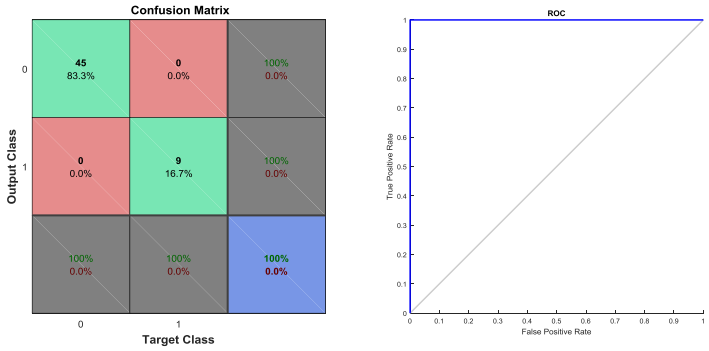


Figura A2-27: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y cualquier tipo de alimentación. Clasificación binaria clase R3.

- ## Nivel de carga 2



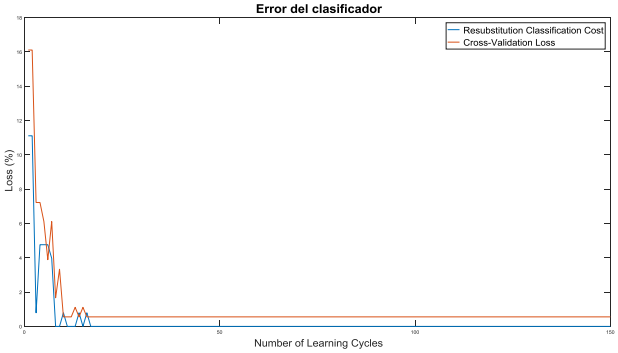
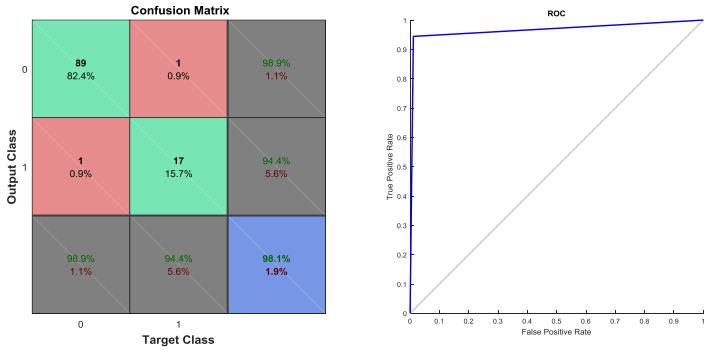


Figura A2-28: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y cualquier tipo de alimentación. Clasificación binaria clase R3.

- ## Cualquier nivel de carga



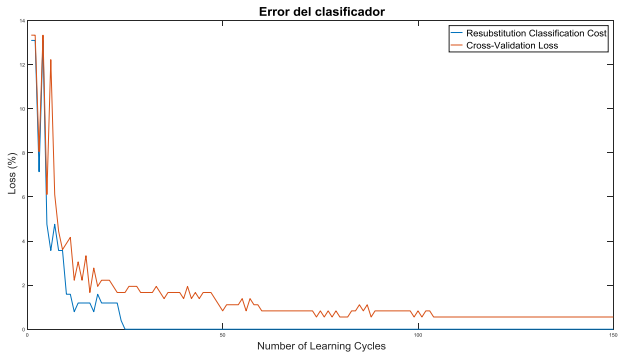


Figura A2-29: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y cualquier tipo de alimentación. Clasificación binaria clase R3.

2.1.5 Clasificador Binario para clase R4

2.1.5.1 Ensayos para alimentación de Red

- **Nivel de carga 1**

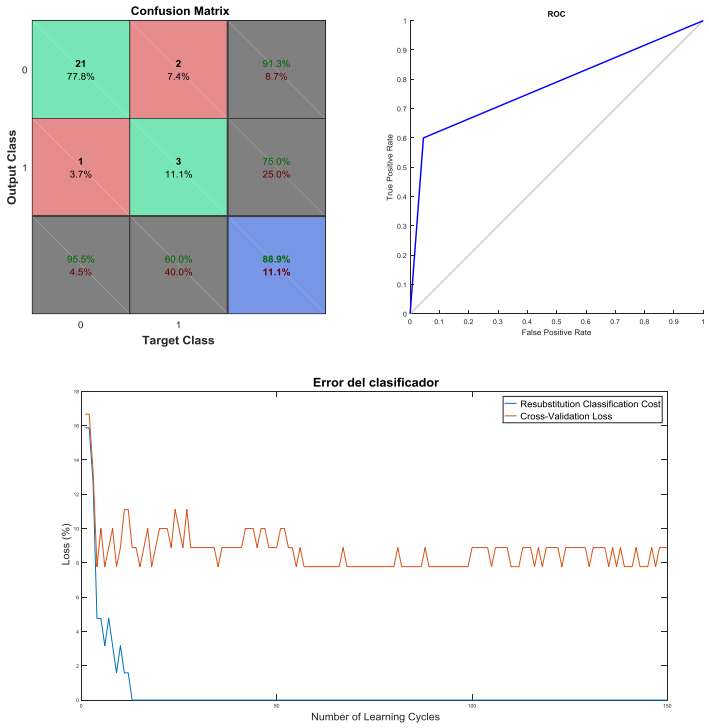


Figura A2-30: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación de Red. Clasificación binaria clase R4.

- ## Nivel de carga 2

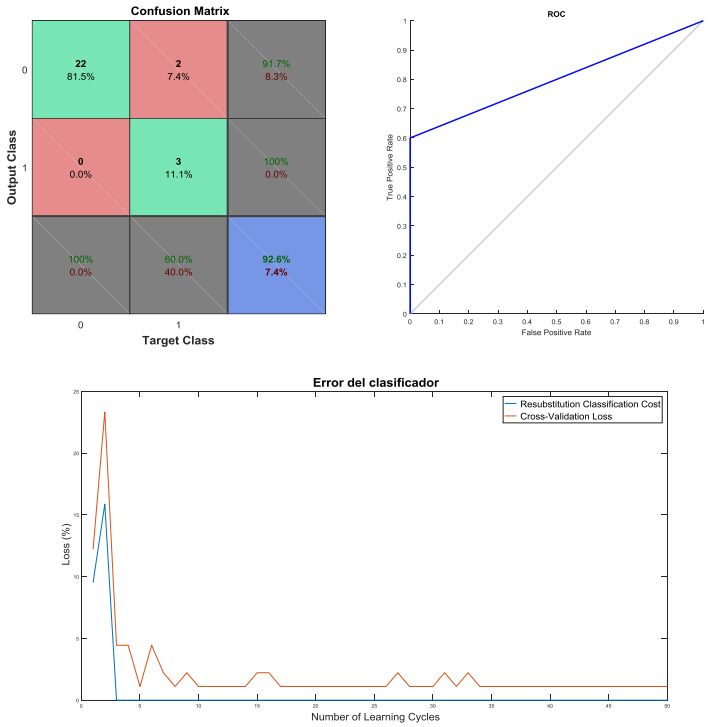


Figura A2-31: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación de Red. Clasificación binaria clase R4.

- ## Cualquier nivel de carga

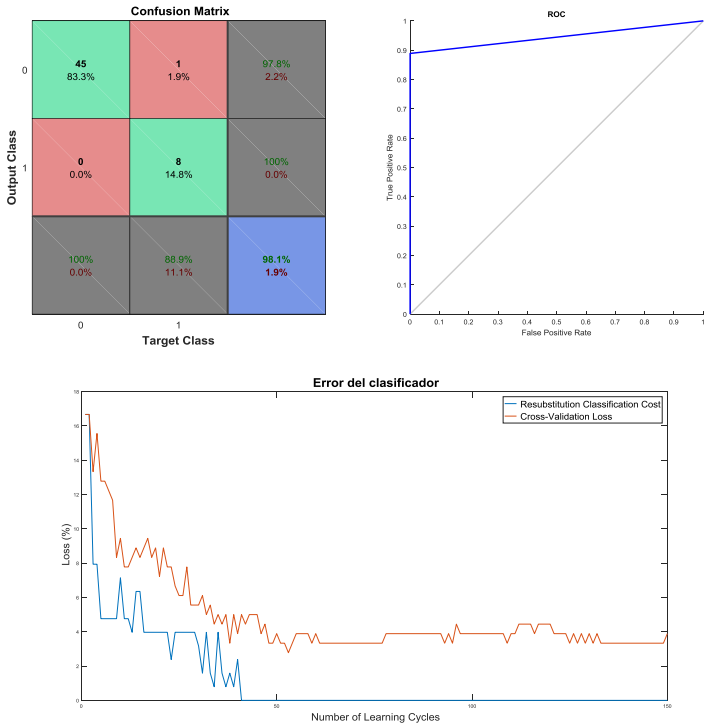


Figura A2-32: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación de Red. Clasificación binaria clase R4.

2.1.5.2 Ensayos para alimentación mediante convertidor

- **Nivel de carga 1**

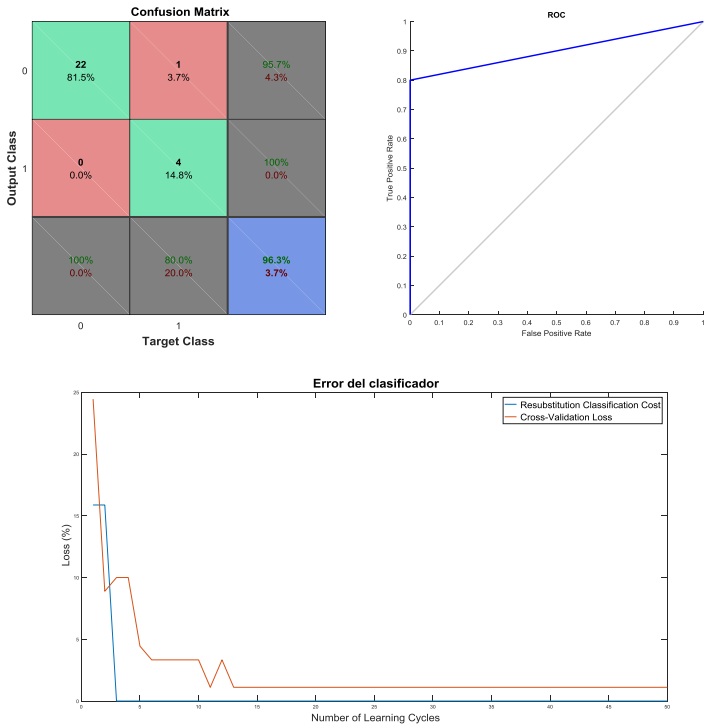


Figura A2-33: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación mediante convertidor. Clasificación binaria clase R4.

- **Nivel de carga 2**

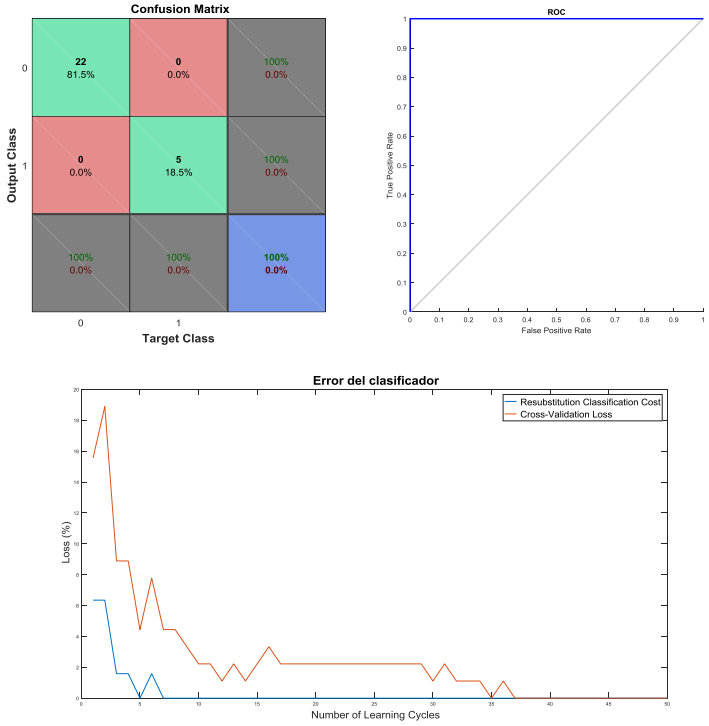


Figura A2-34: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación mediante convertidor. Clasificación binaria clase R4.

- **Cualquier nivel de carga**

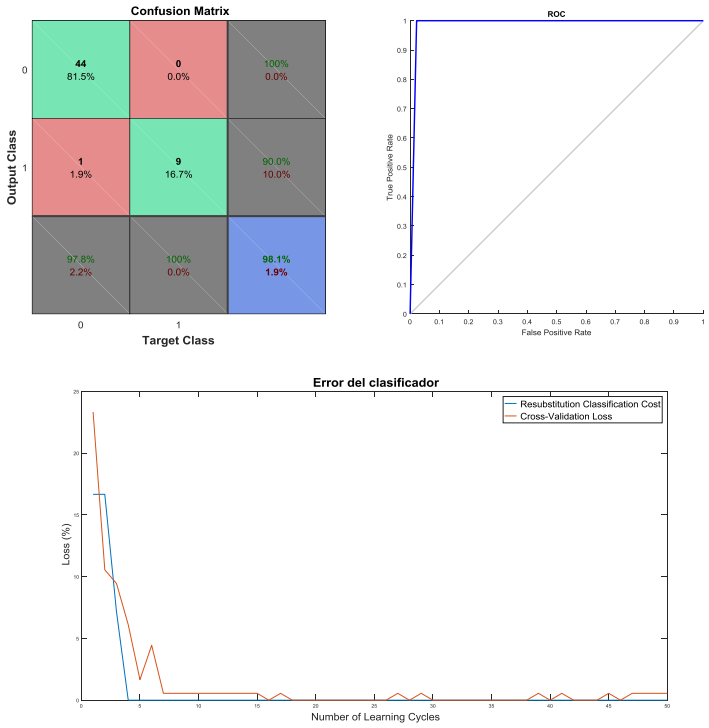


Figura A2-35: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación mediante convertidor. Clasificación binaria clase R4.

2.1.5.3 Ensayos para cualquier tipo de alimentación

- **Nivel de carga 1**

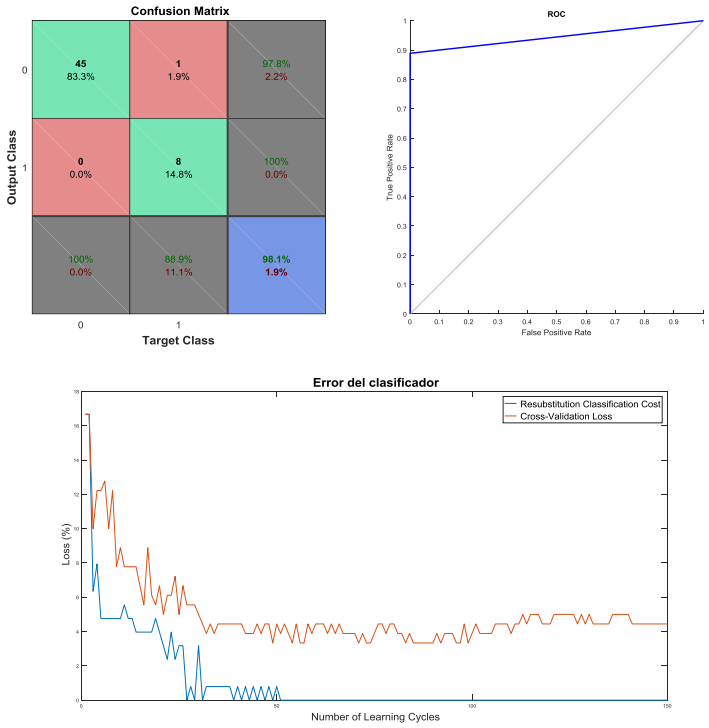


Figura A2-36: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y cualquier tipo de alimentación. Clasificación binaria clase R4.

- ## Nivel de carga 2

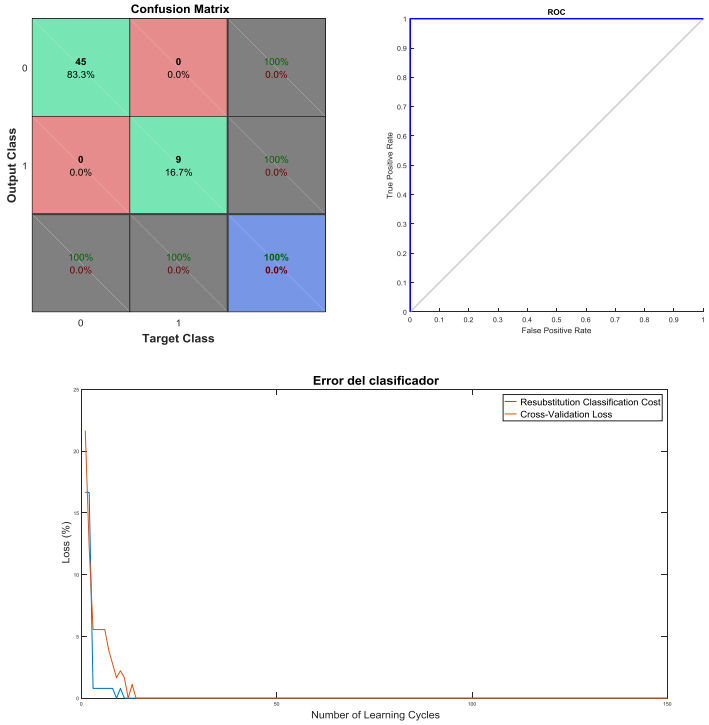


Figura A2-37: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y cualquier tipo de alimentación. Clasificación binaria clase R4.

- ## Cualquier nivel de carga

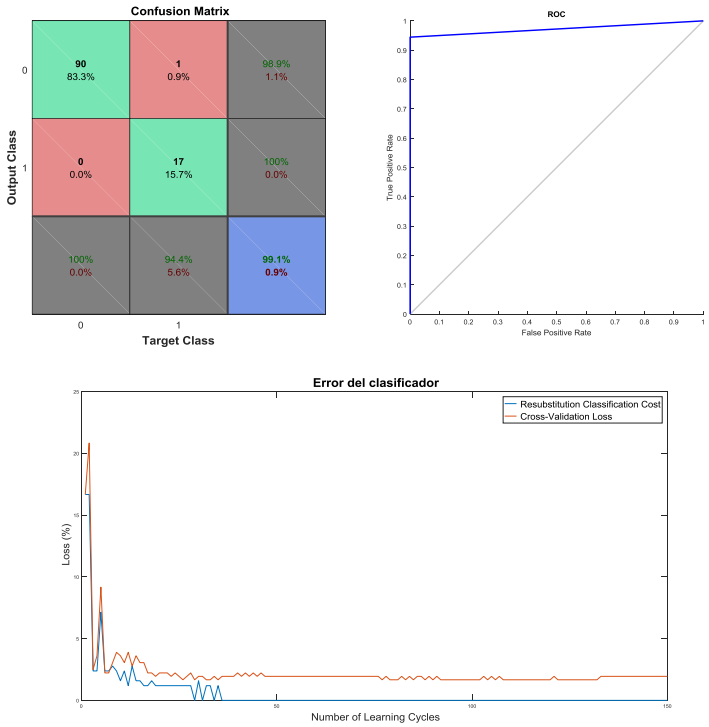


Figura A2-38: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y cualquier tipo de alimentación. Clasificación binaria clase R4.

2.1.6 Clasificador Binario para clase R5

2.1.6.1 Ensayos para alimentación de Red

- **Nivel de carga 1**

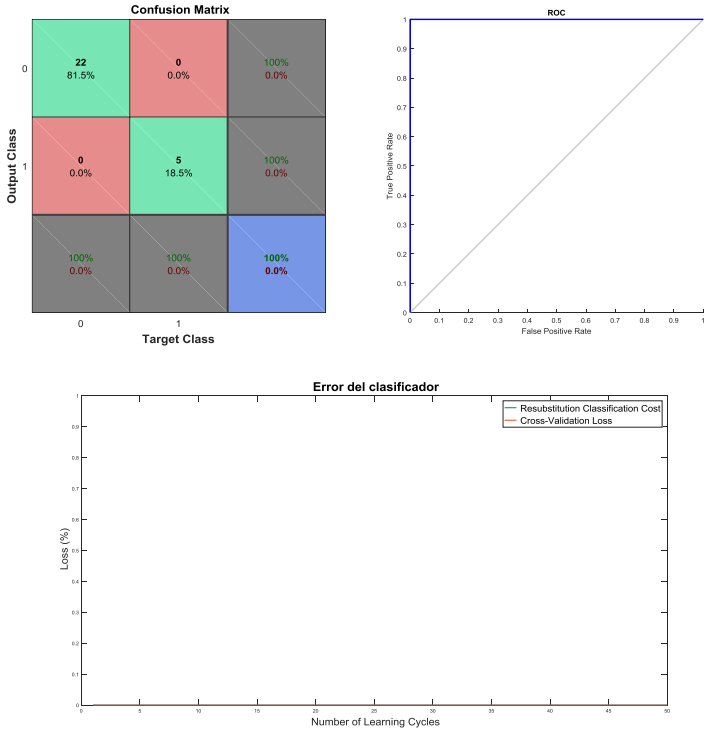


Figura A2-39: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación de Red. Clasificación binaria clase R5.

- **Nivel de carga 2**

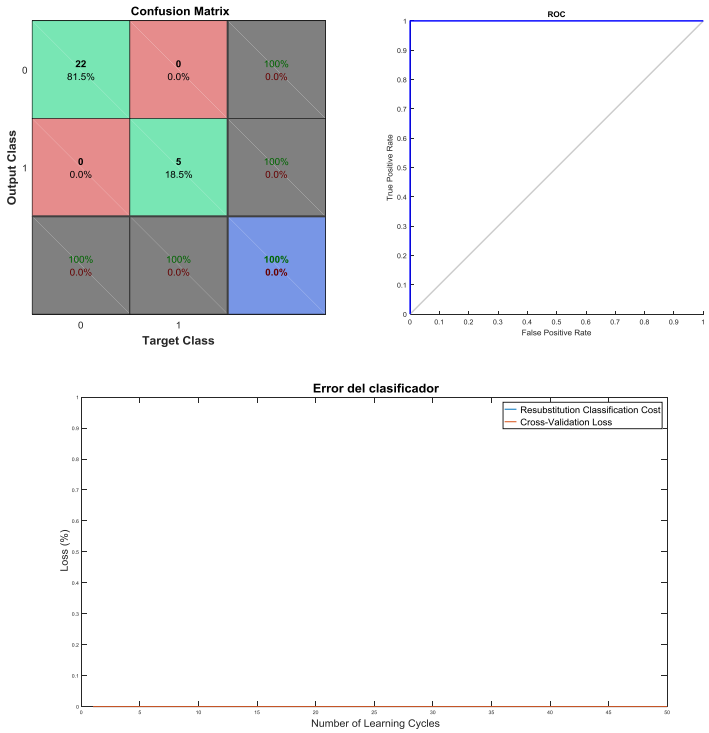


Figura A2-50: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y

alimentación de Red. Clasificación binaria clase R5.

- **Cualquier nivel de carga**

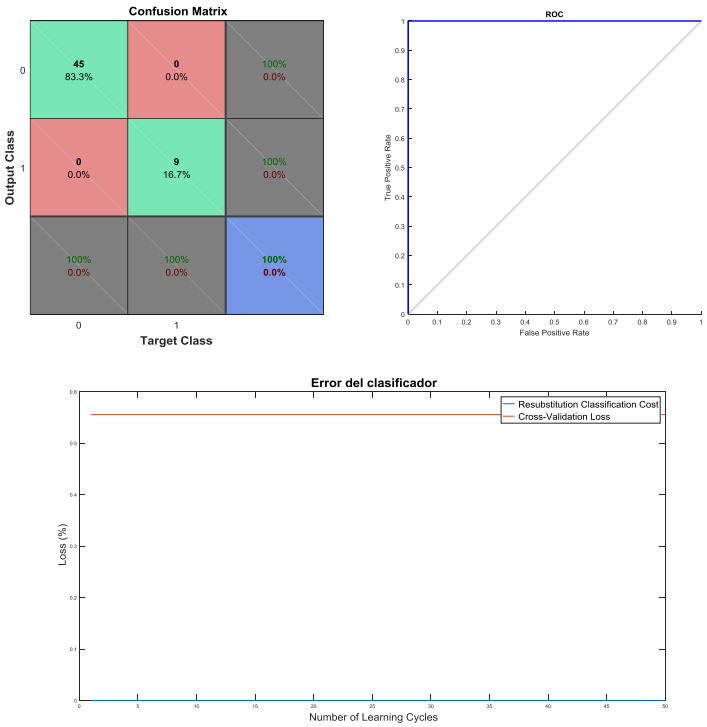


Figura A2-51: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación de Red. Clasificación binaria clase R5.

2.1.6.2 Ensayos para alimentación mediante convertidor

- **Nivel de carga 1**

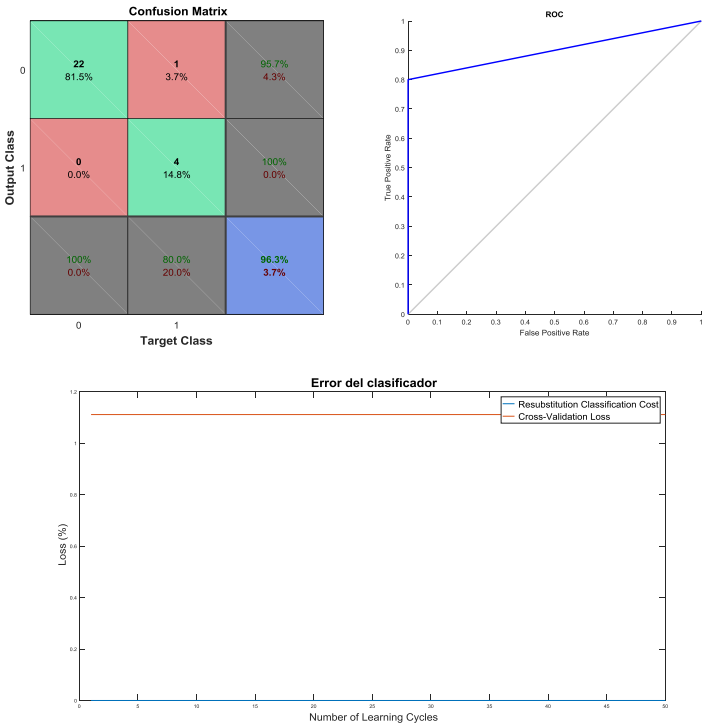


Figura A2-52: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y

alimentación mediante convertidor. Clasificación binaria clase R5.

- **Nivel de carga 2**

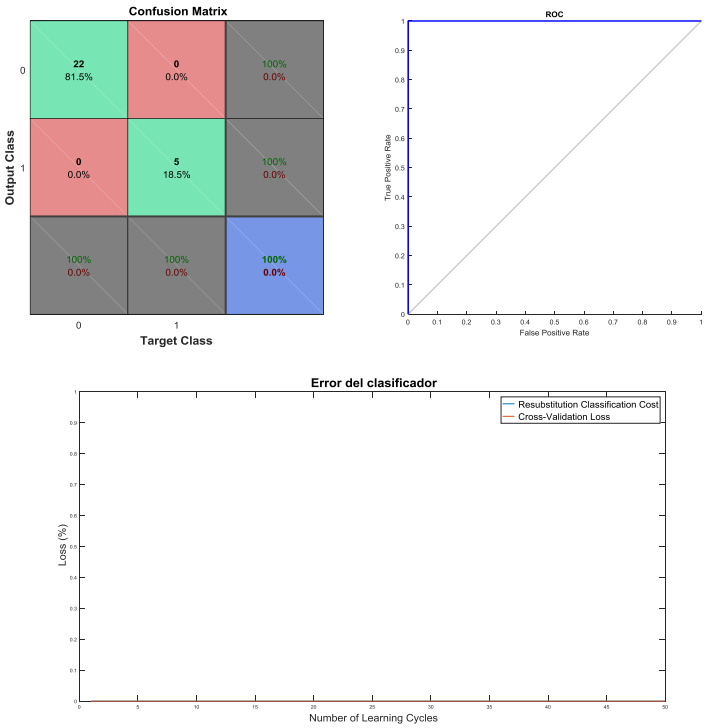


Figura A2-53: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación mediante convertidor. Clasificación binaria clase R5.

- **Cualquier nivel de carga**

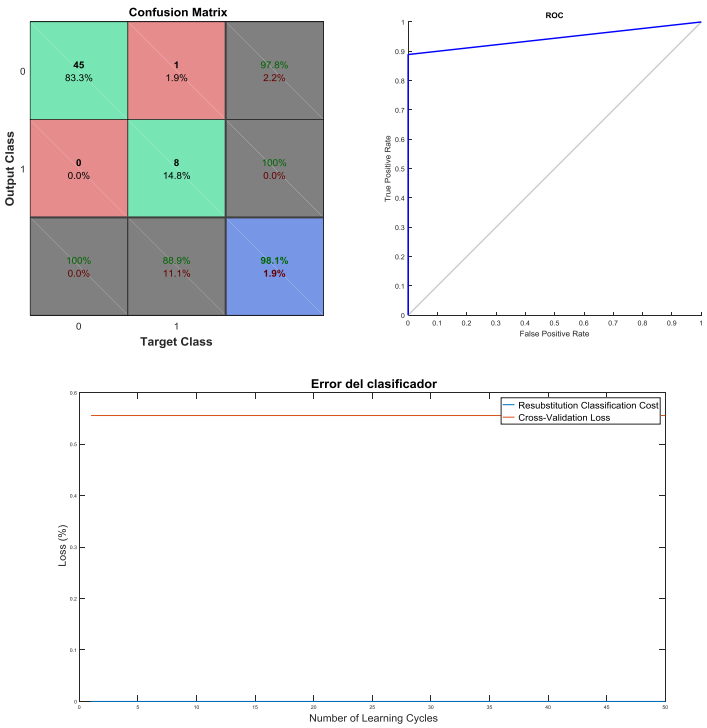


Figura A2-54: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación mediante convertidor. Clasificación binaria clase R5.

2.1.6.3 Ensayos para cualquier tipo de alimentación

- **Nivel de carga 1**

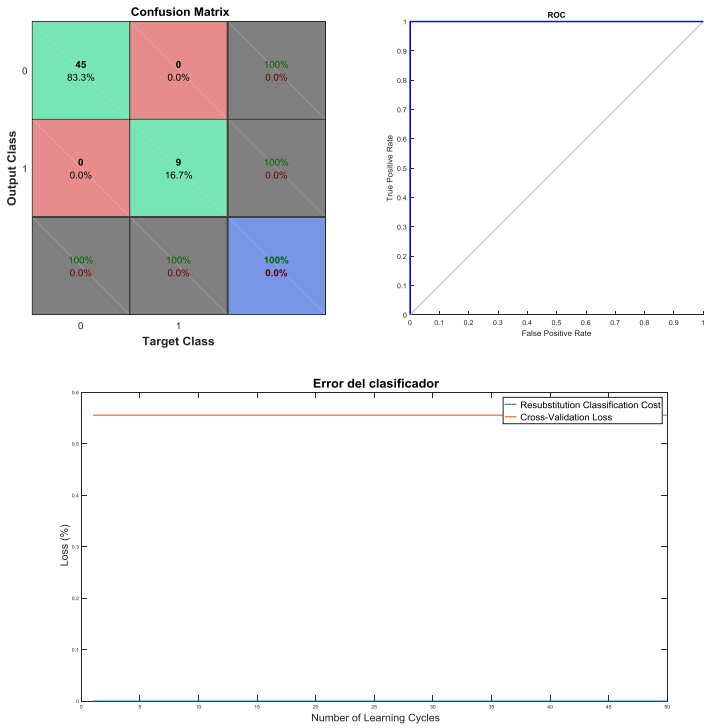


Figura A2-55: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y cualquier tipo de alimentación. Clasificación binaria clase R5.

- **Nivel de carga 2**

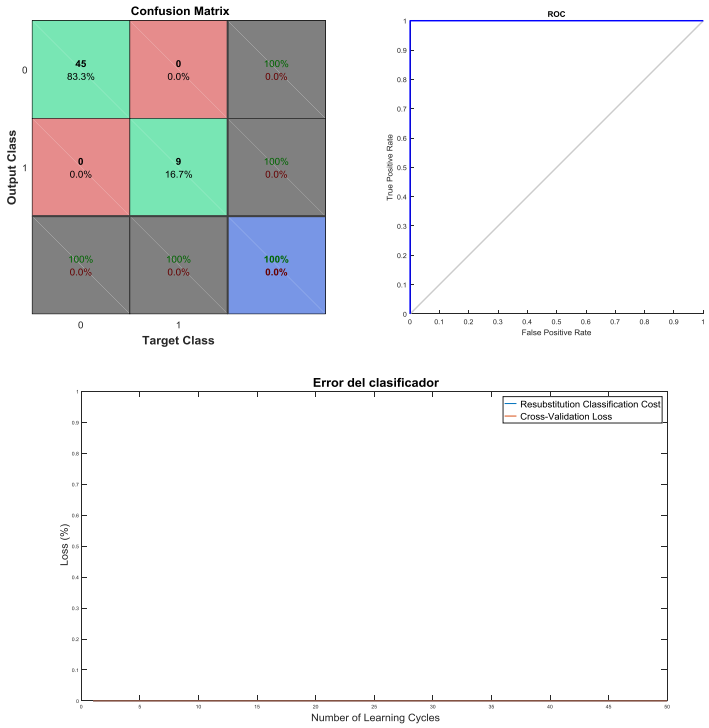


Figura A2-56: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y cualquier tipo de alimentación. Clasificación binaria clase R5.

- **Cualquier nivel de carga**

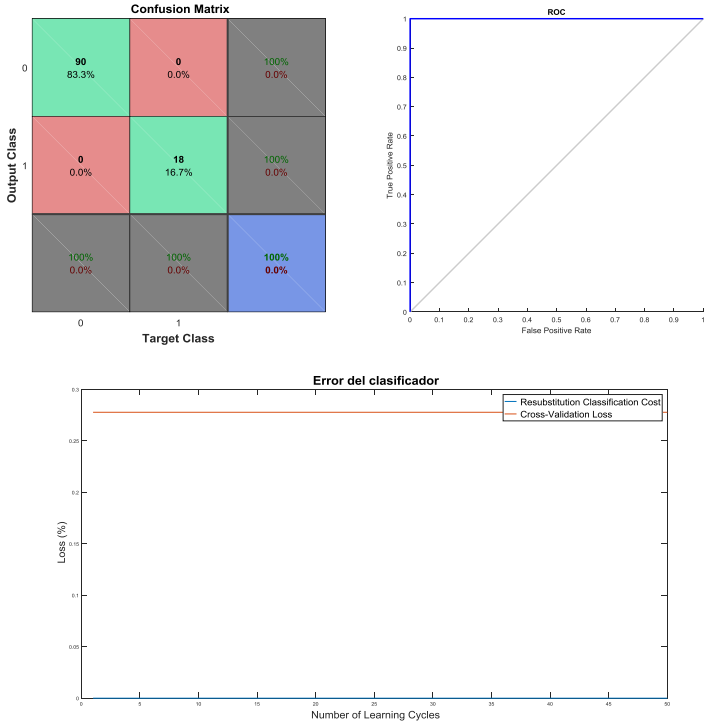


Figura A2-57: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y cualquier tipo de alimentación. Clasificación binaria clase R5.

2.1.7 Clasificador multiclase

2.1.7.1 Ensayos para alimentación de Red

- **Nivel de carga 1**

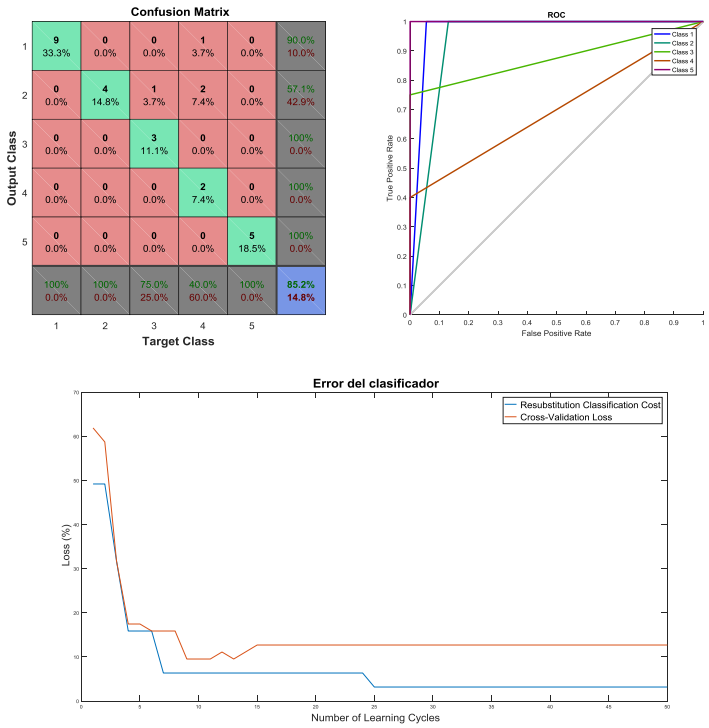


Figura A2-58: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y

alimentación de Red. Clasificación multiclase.

- ## Nivel de carga 2

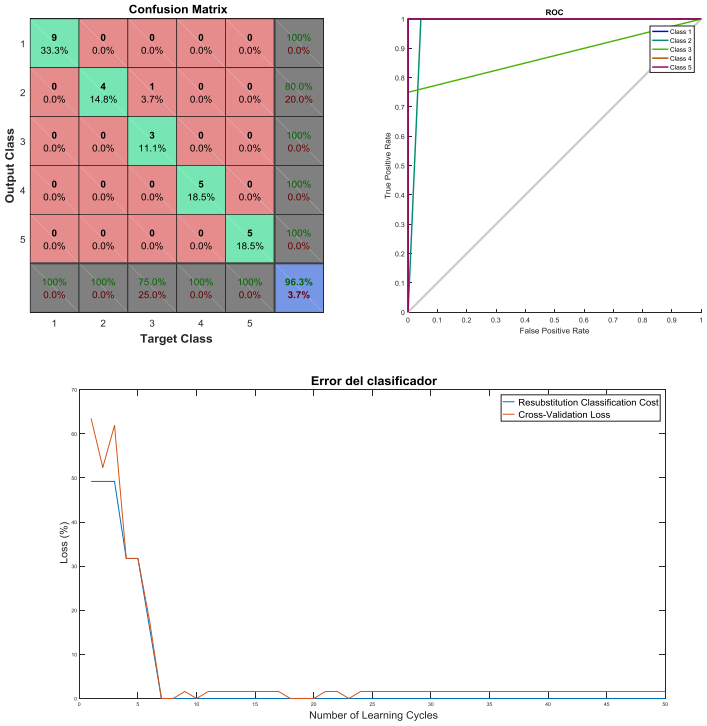


Figura A2-59: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación de Red. Clasificación multiclase.

- ## Cualquier nivel de carga

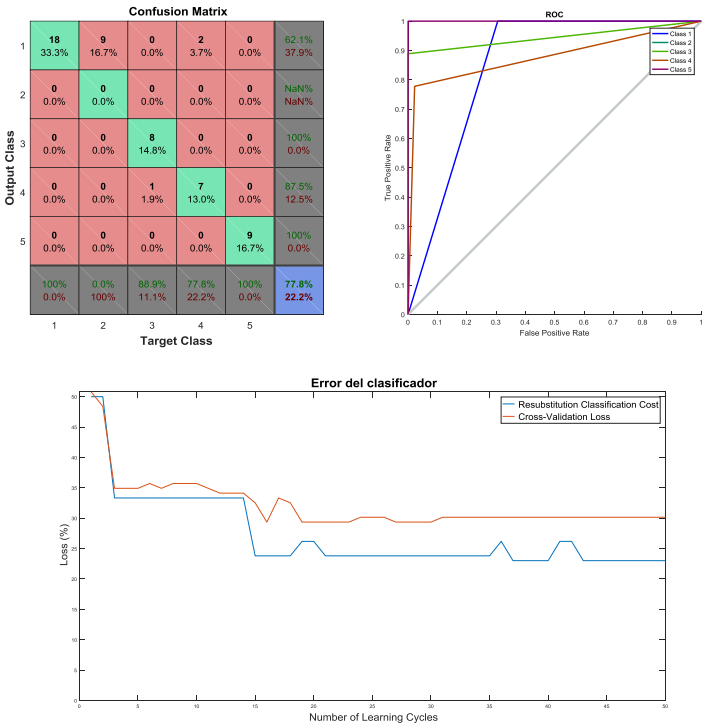


Figura A2-60: Ensayo con 50 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación de Red. Clasificación multiclase.

2.1.7.2 Ensayos para alimentación mediante convertidor

- **Nivel de carga 1**

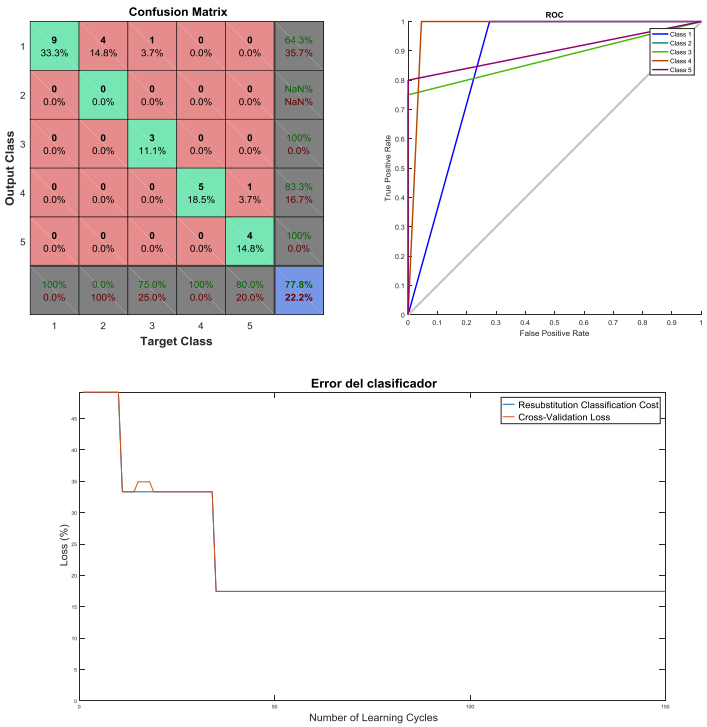


Figura A2-61: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y alimentación mediante convertidor. Clasificación multiclase.

- **Nivel de carga 2**

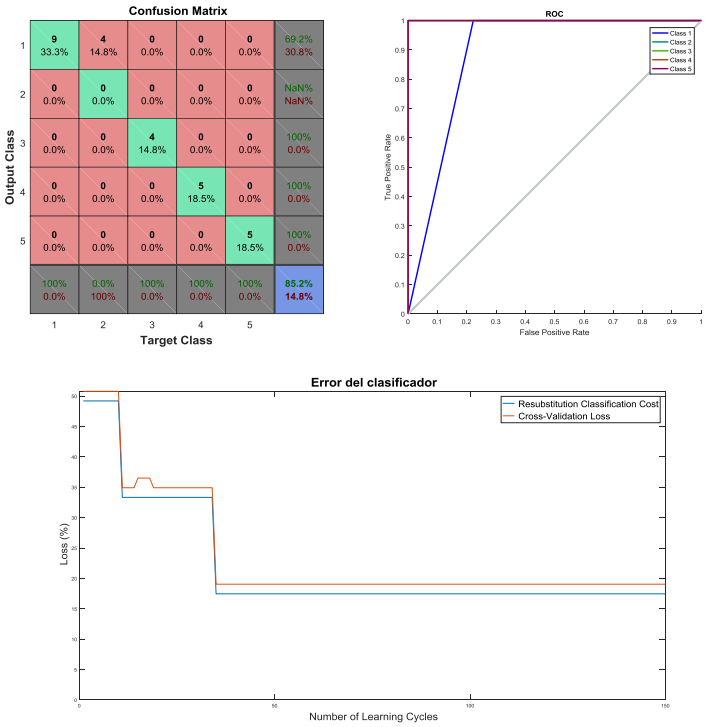


Figura A2-62: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y alimentación mediante convertidor. Clasificación multiclase.

- **Cualquier nivel de carga**

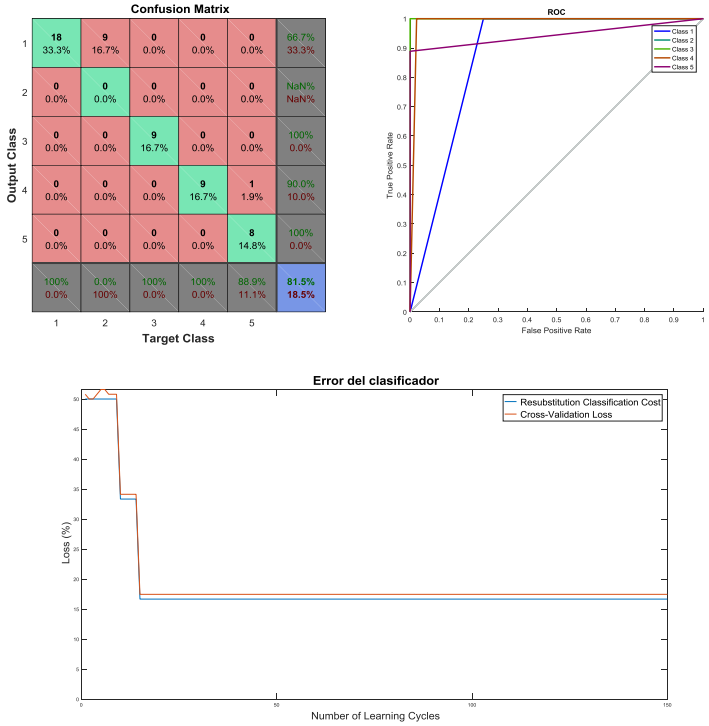


Figura A2-63: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y alimentación mediante convertidor. Clasificación multiclase.

2.1.7.3 Ensayos para cualquier tipo de alimentación

- **Nivel de carga 1**

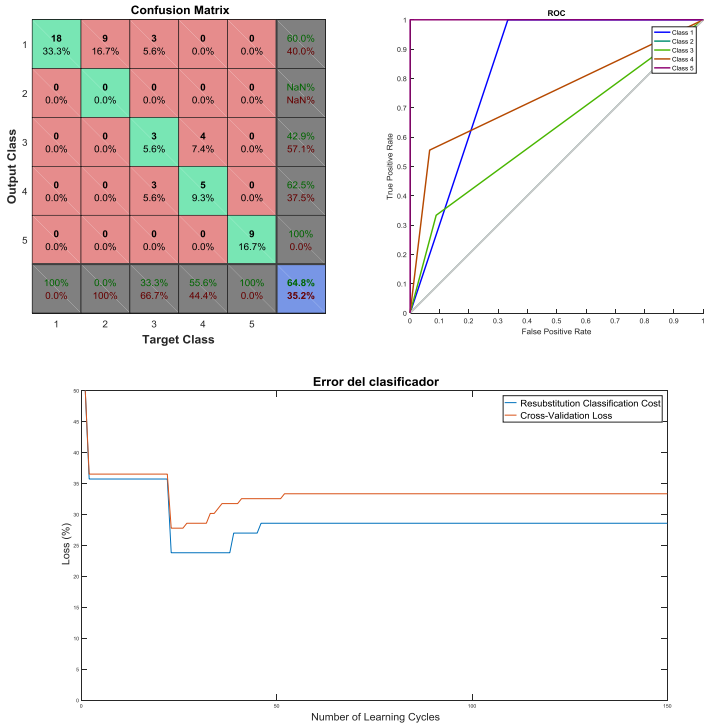


Figura A2-64: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 1 y cualquier tipo de alimentación. Clasificación multiclase.

- ## Nivel de carga 2

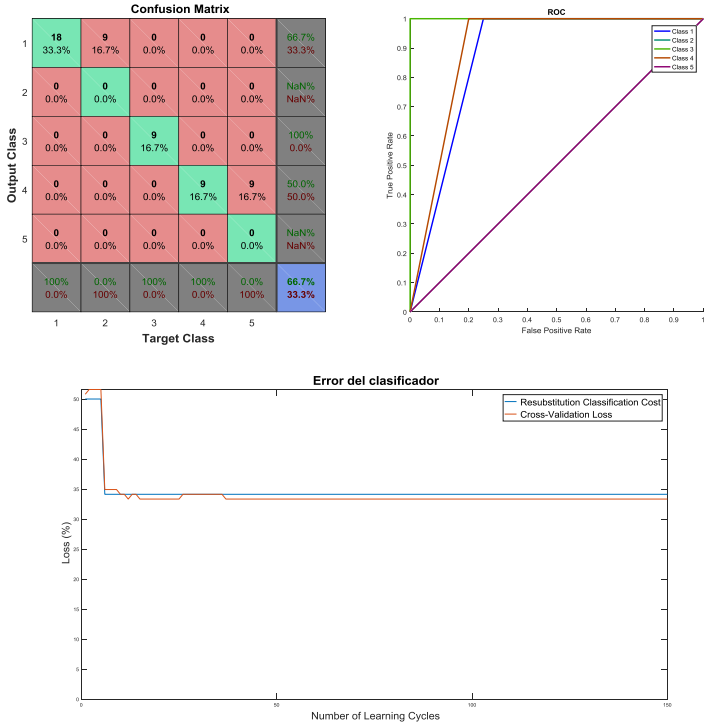


Figura A2-65: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, nivel de carga 2 y cualquier tipo de alimentación. Clasificación multiclase.

- ## Cualquier nivel de carga

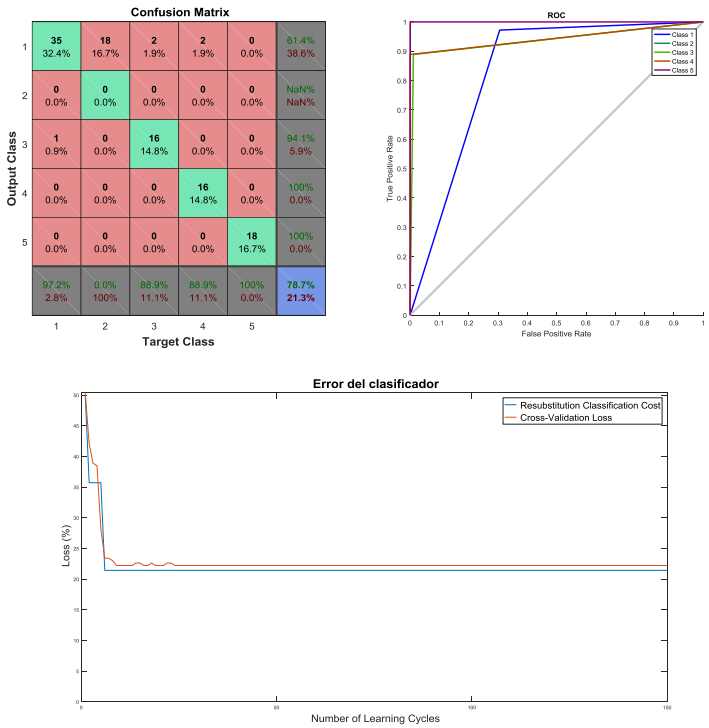


Figura A2-66: Ensayo con 150 clasificadores débiles, 20 subgrupos para validación cruzada, cualquier nivel de carga y cualquier tipo de alimentación. Clasificación multiclase.

2.2 Métodos de clasificación multiclase

2.2.1 Método 1

2.2.1.1 Ensayo 1

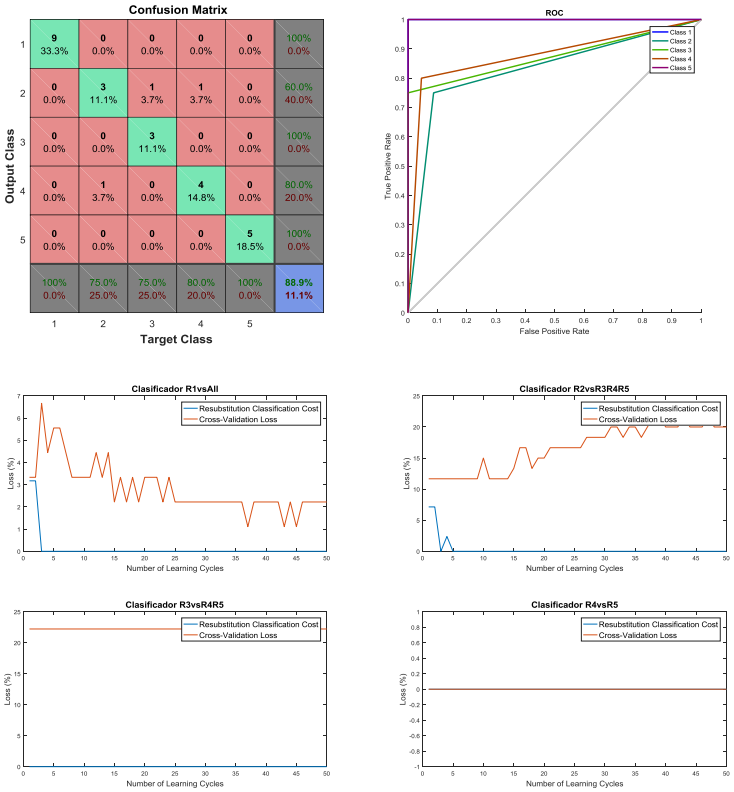
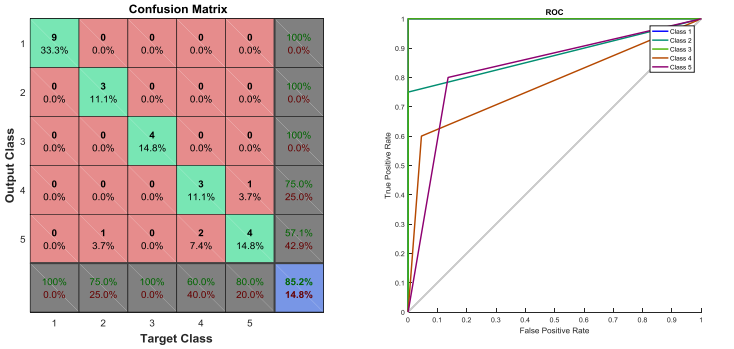


Figura A2-67: Ensayo mediante método 1 de clasificación. Semilla de valor 1. Clasificadores entrenados mediante muestra de entrenamiento reducida.



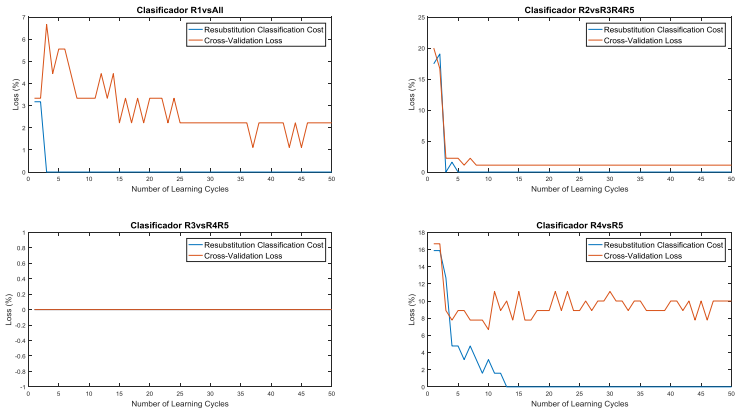


Figura A2-68: Ensayo mediante método 1 de clasificación. Semilla de valor 1. Clasificadores entrenados mediante muestra de entrenamiento completa.

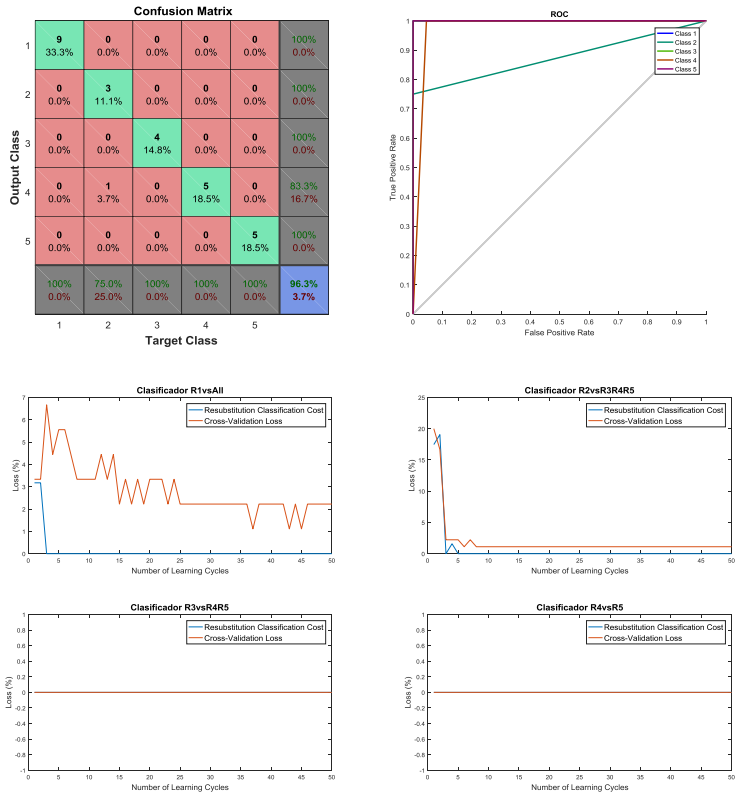


Figura A2-69: Ensayo mediante método 1 de clasificación. Semilla de valor 1. Clasificadores entrenados mediante muestra de entrenamiento completa, excepto el clasificador R4vsR5.

2.2.1.2 Ensayo 2

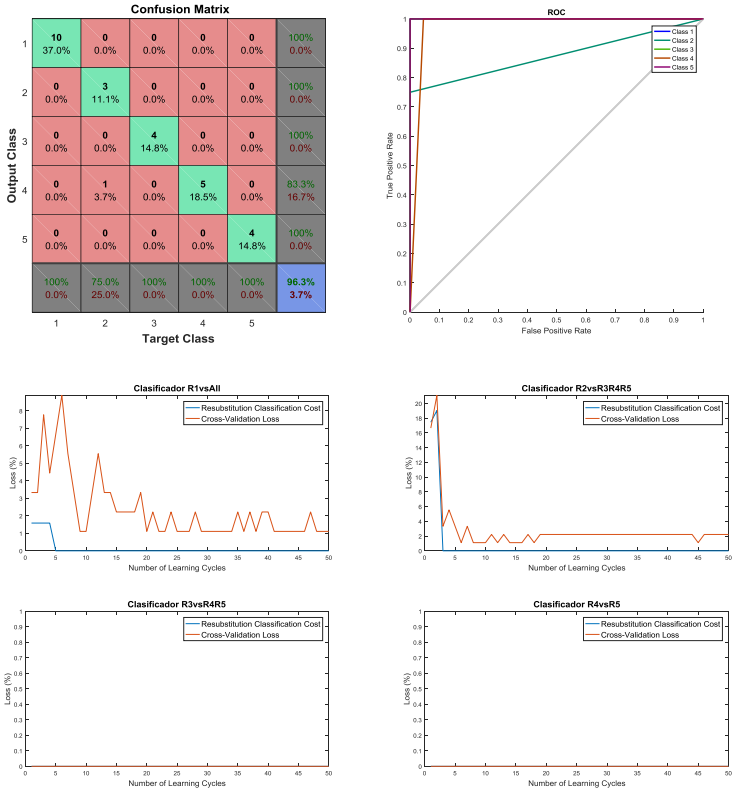


Figura A2-70: Ensayo mediante método 1 de clasificación. Semilla de valor 7. Clasificadores entrenados mediante muestra de entrenamiento completa, excepto el clasificador R4vsR5.

2.2.1.3 Ensayo 3

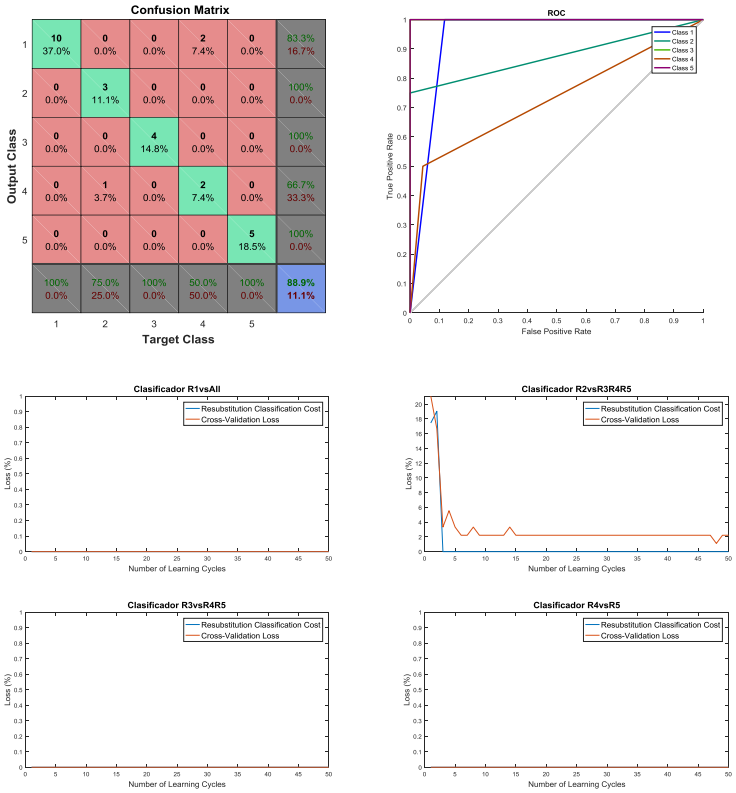


Figura A2-71: Ensayo mediante método 1 de clasificación. Semilla de valor 50. Clasificadores entrenados mediante

muestra de entrenamiento completa, excepto el clasificador R4vsR5.

2.2.1.4 Ensayo 4

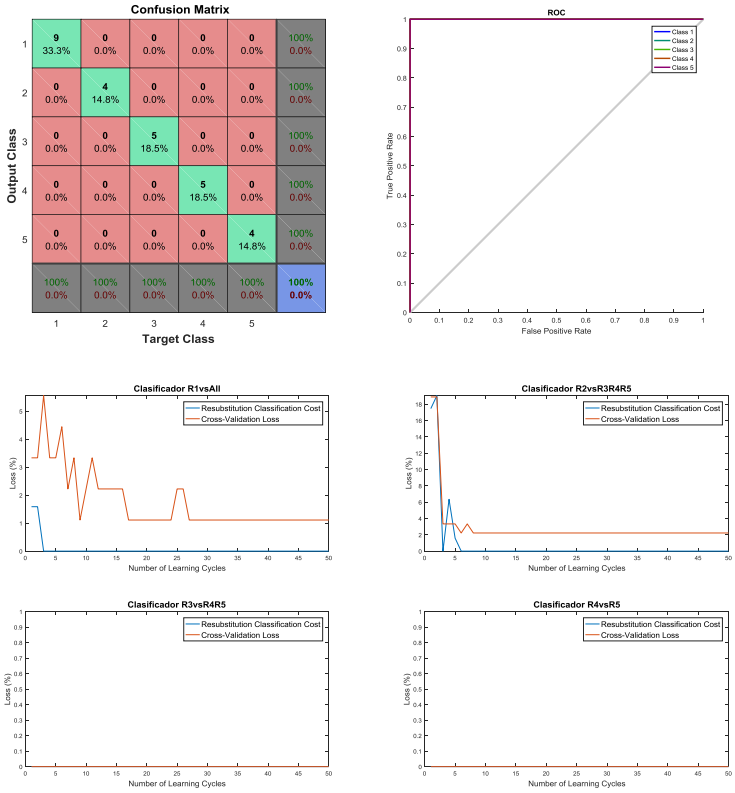
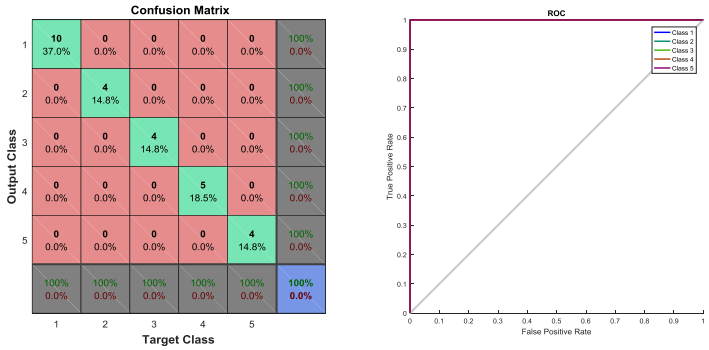


Figura A2-72: Ensayo mediante método 1 de clasificación. Semilla de valor 102. Clasificadores entrenados mediante muestra de entrenamiento completa, excepto el clasificador R4vsR5.

2.2.1.5 Ensayo 5



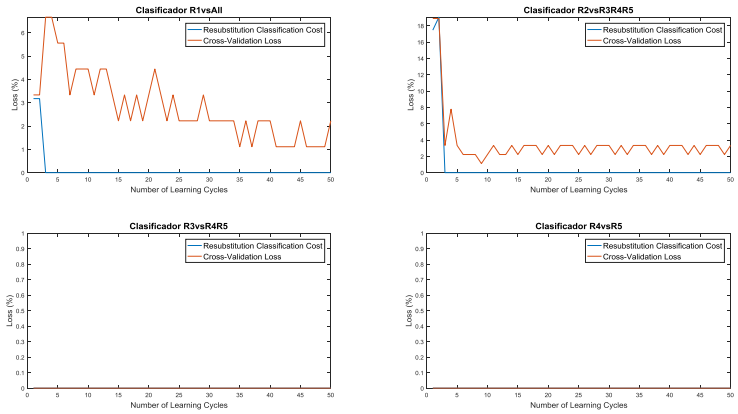


Figura A2-73: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Clasificadores entrenados mediante muestra de entrenamiento completa, excepto el clasificador R4vsR5.

2.2.2 Método 2

2.2.2.1 Ensayo 1

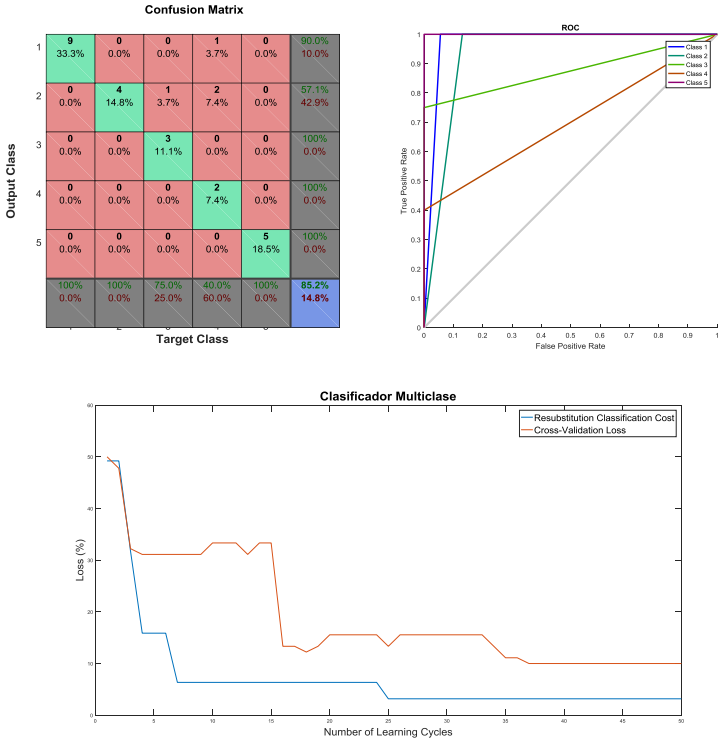


Figura A2-74: Ensayo mediante método 2 de clasificación. Semilla de valor 1.

2.2.2.2 Ensayo 2

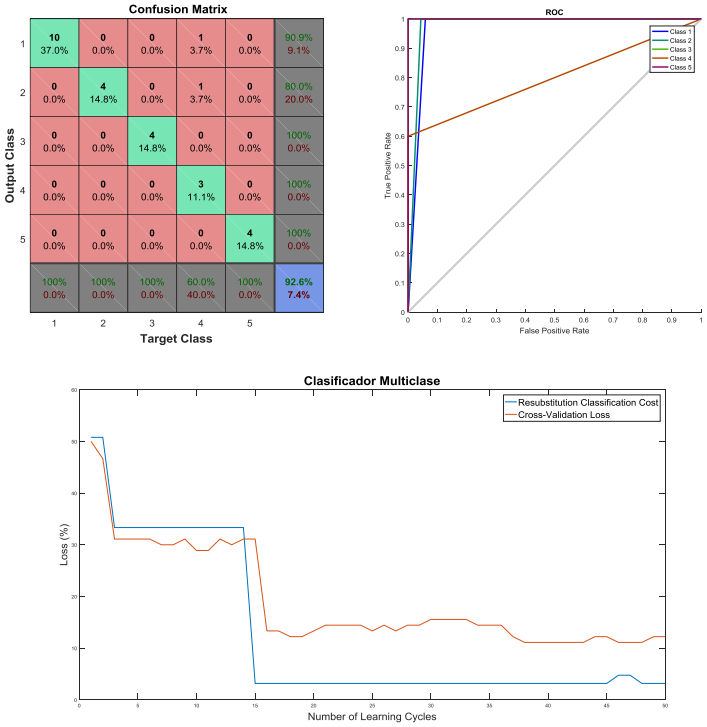


Figura A2-75: Ensayo mediante método 2 de clasificación. Semilla de valor 7.

2.2.2.3 Ensayo 3

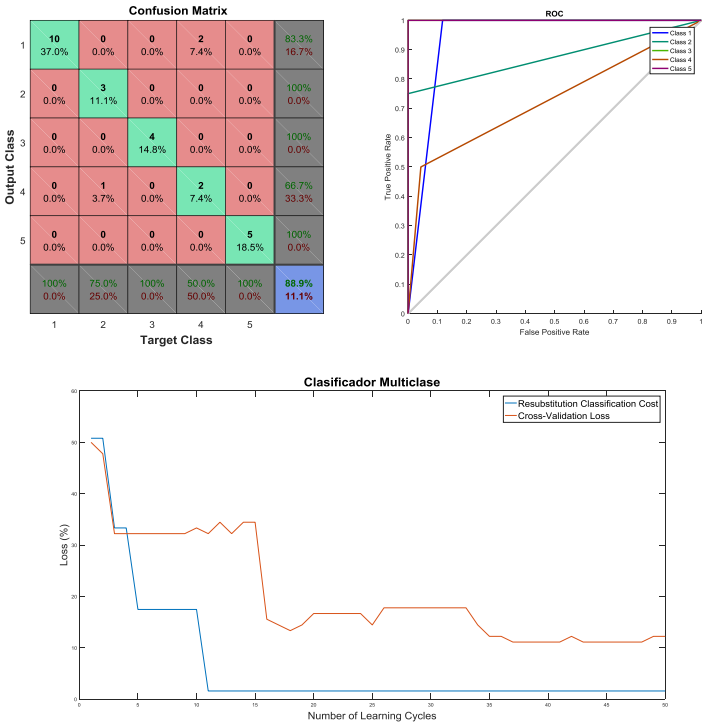


Figura A2-76: Ensayo mediante método 2 de clasificación. Semilla de valor 50.

2.2.2.4 Ensayo 4

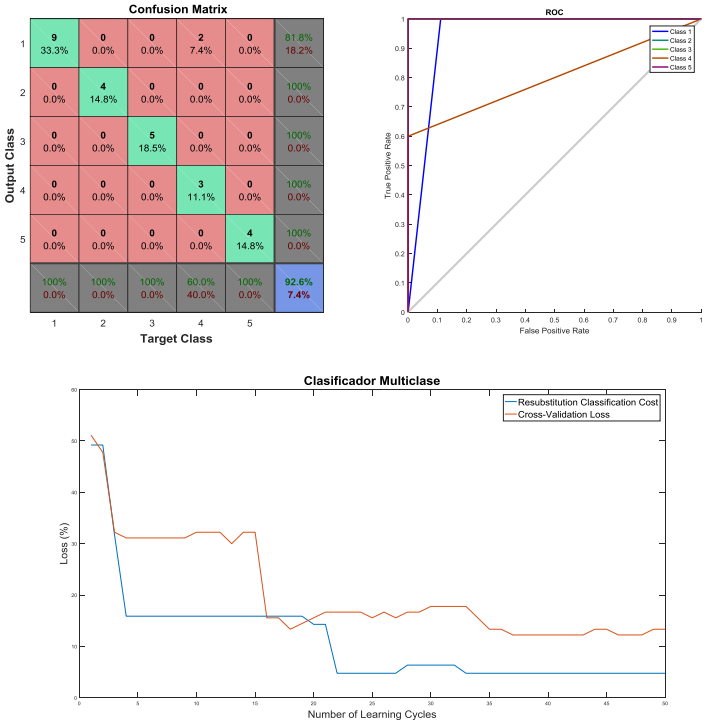


Figura A2-77: Ensayo mediante método 2 de clasificación. Semilla de valor 102.

2.2.2.5 Ensayo 5

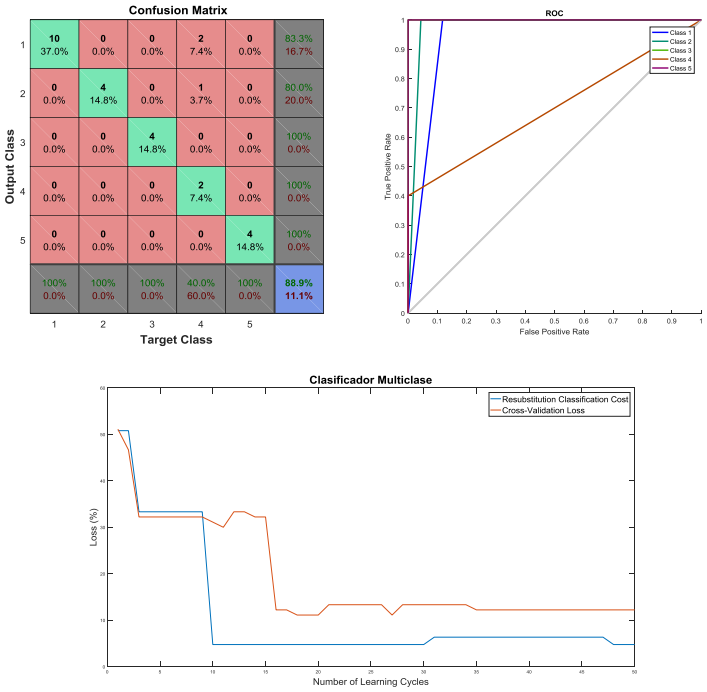


Figura A2-78: Ensayo mediante método 2 de clasificación. Semilla de valor 320.

2.2.3 Método 3

2.2.3.1 Ensayo 1

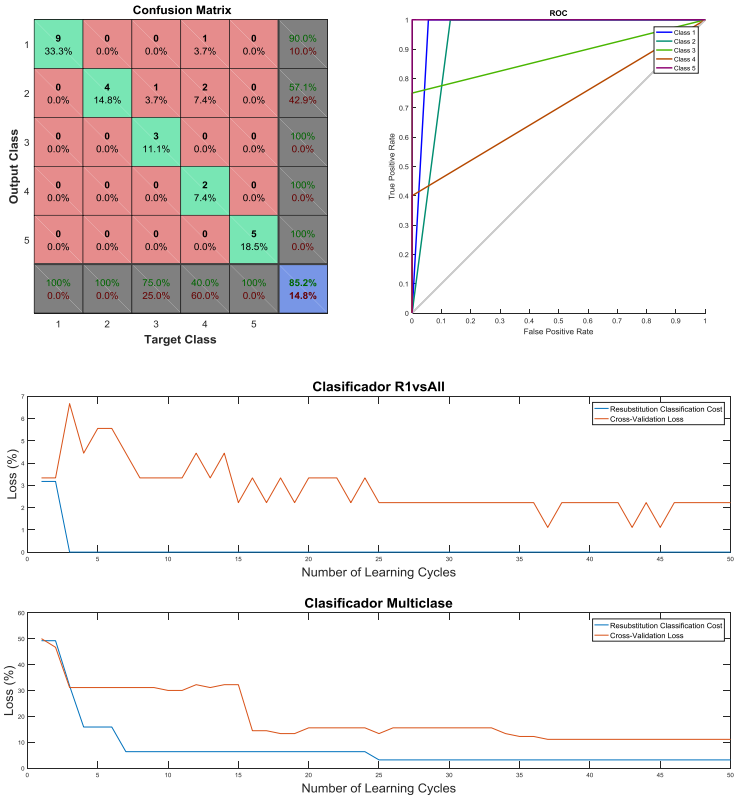


Figura A2-79: Ensayo mediante método 3 de clasificación. Semilla de valor 1. Clasificador multiclase entrenado mediante muestra de entrenamiento completa.

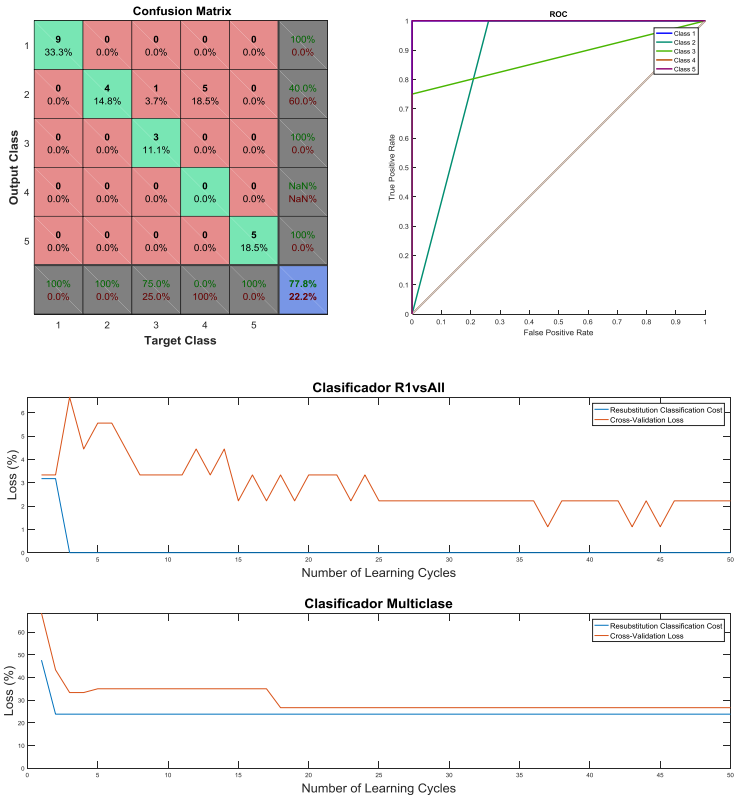


Figura A2-80: Ensayo mediante método 3 de clasificación. Semilla de valor 1. Clasificador multiclase entrenado mediante muestra de entrenamiento reducida.

2.2.3.2 *Ensayo 2*

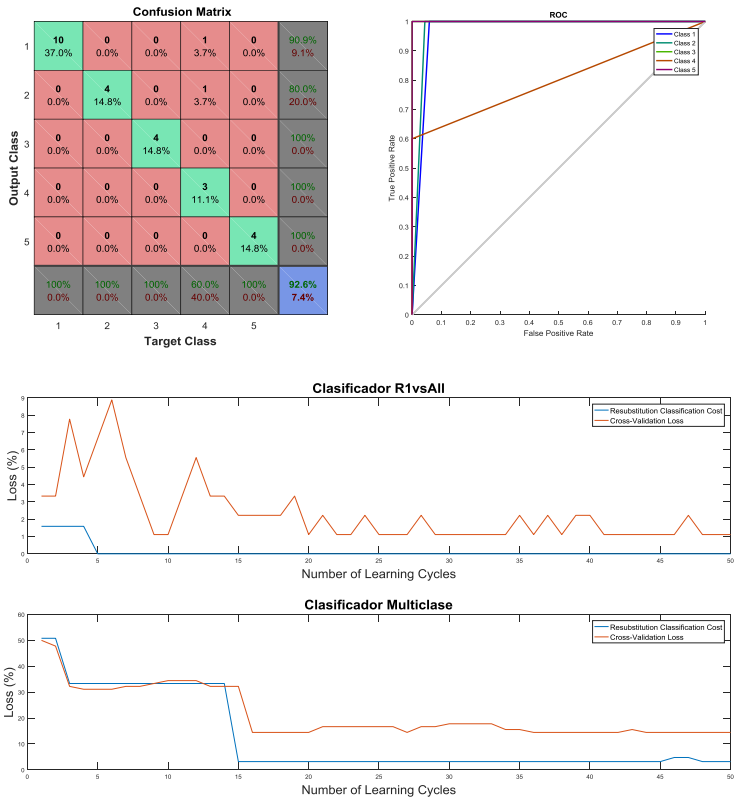


Figura A2-81: Ensayo mediante método 3 de clasificación. Semilla de valor 7. Clasificadores multiclase entrenado mediante muestra de entrenamiento completa.

2.2.3.3 Ensayo 3

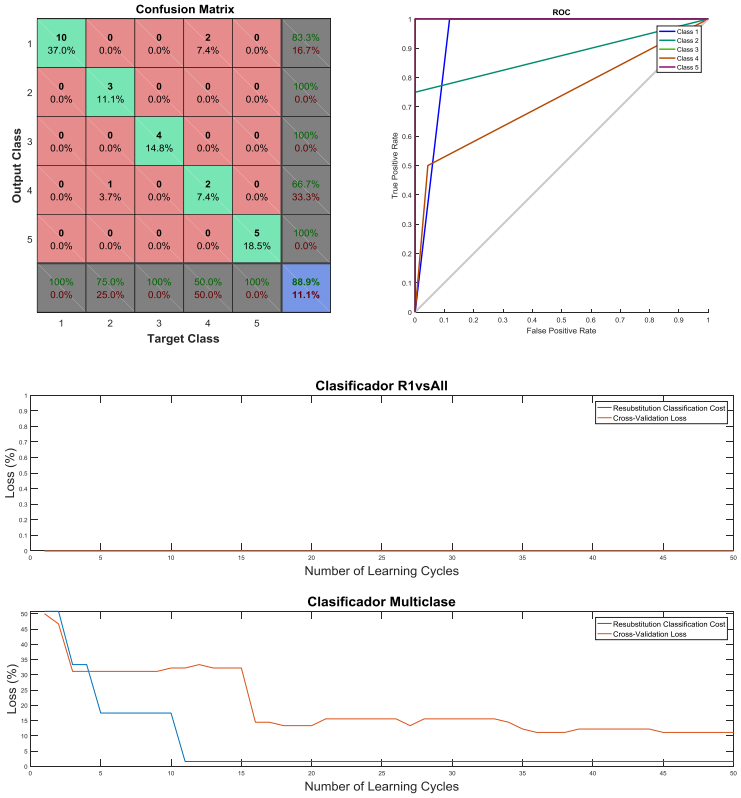


Figura A2-82: Ensayo mediante método 3 de clasificación. Semilla de valor 50. Clasificadores multiclase entrenado mediante muestra de entrenamiento completa.

2.2.3.4 Ensayo 4

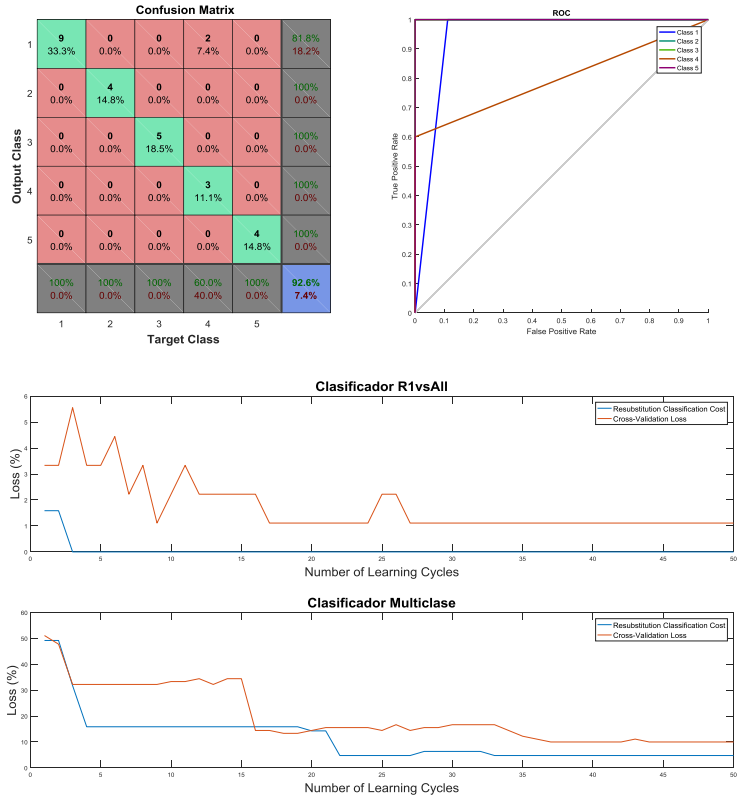


Figura A2-83: Ensayo mediante método 3 de clasificación. Semilla de valor 102. Clasificadores multiclase entrenado

mediante muestra de entrenamiento completa.

2.2.3.5 Ensayo 5

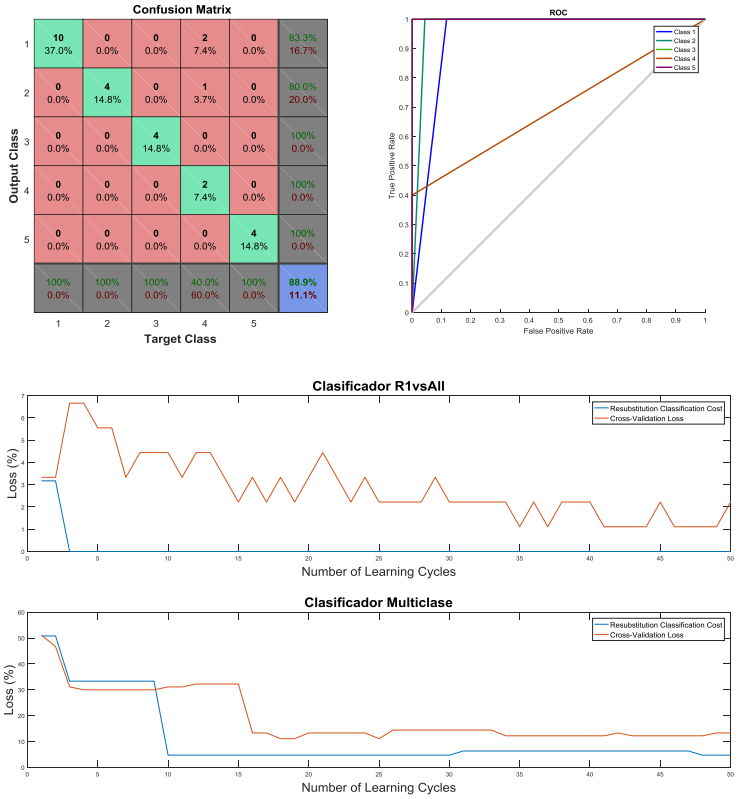
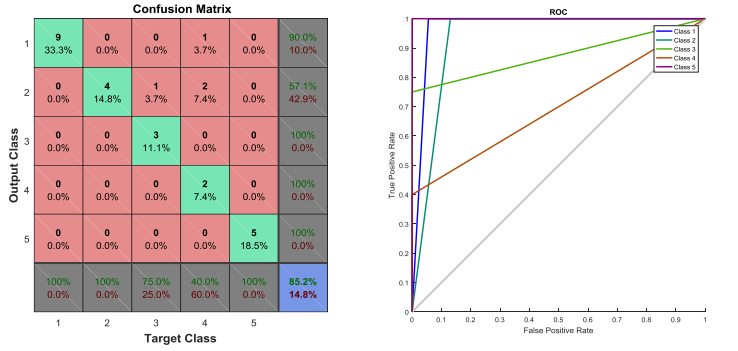


Figura A2-84: Ensayo mediante método 3 de clasificación. Semilla de valor 320. Clasificadores multiclase entrenado mediante muestra de entrenamiento completa.

2.2.4 Método 4

2.2.4.1 Ensayo 1



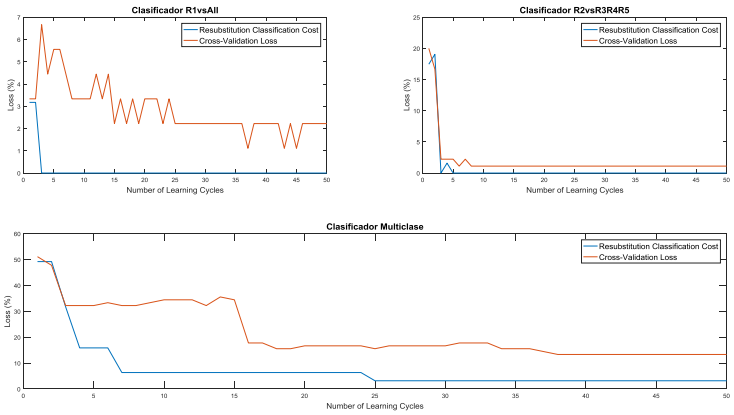


Figura A2-85: Ensayo mediante método 4 de clasificación. Semilla de valor 1. Clasificadores entrenado mediante muestra de entrenamiento completa.

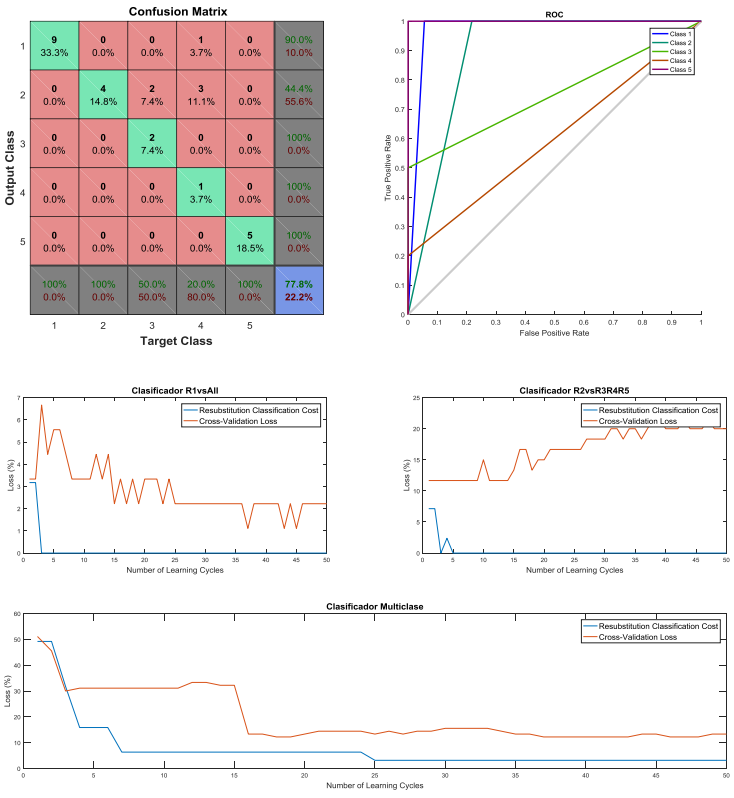
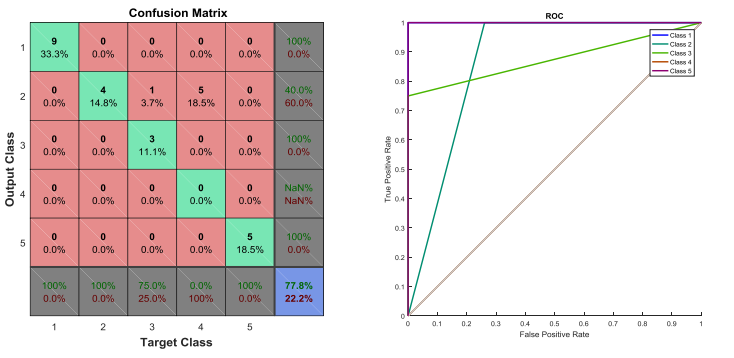


Figura A2-86: Ensayo mediante método 4 de clasificación. Semilla de valor 1. Clasificador binario para R2 entrenado mediante muestra de entrenamiento reducida.



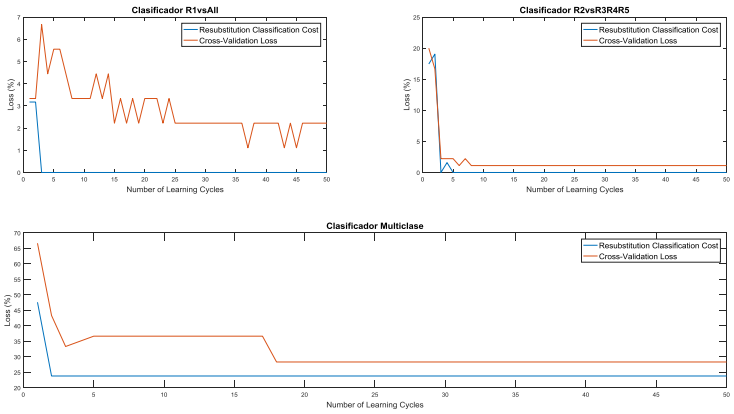


Figura A2-87: Ensayo mediante método 4 de clasificación. Semilla de valor 1. Clasificador multiclase entrenado mediante muestra de entrenamiento reducida.

2.2.4.2 Ensayo 2

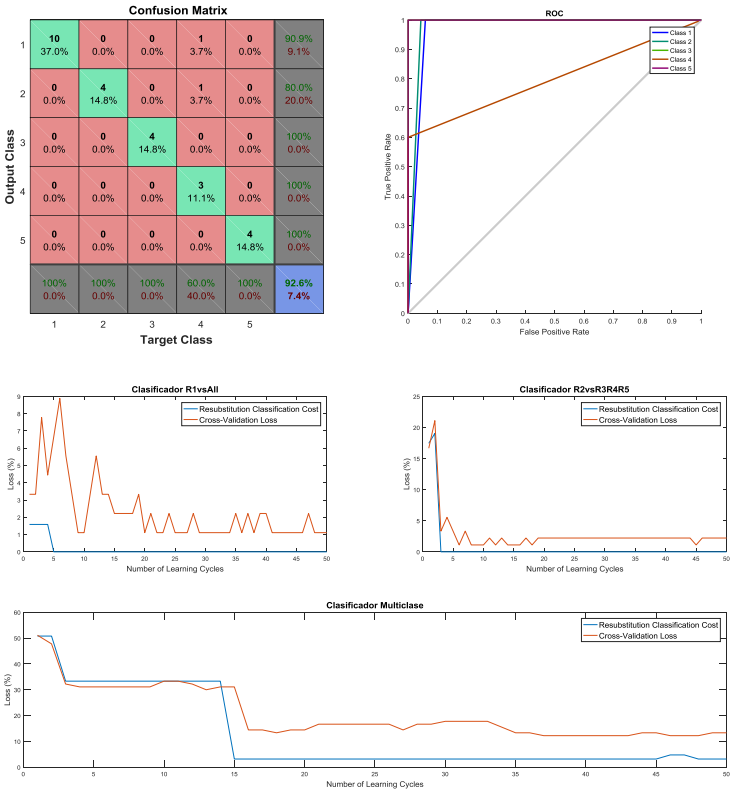


Figura A2-88: Ensayo mediante método 4 de clasificación. Semilla de valor 7. Clasificadores entrenado mediante muestra de entrenamiento completa.

2.2.4.3 Ensayo 3

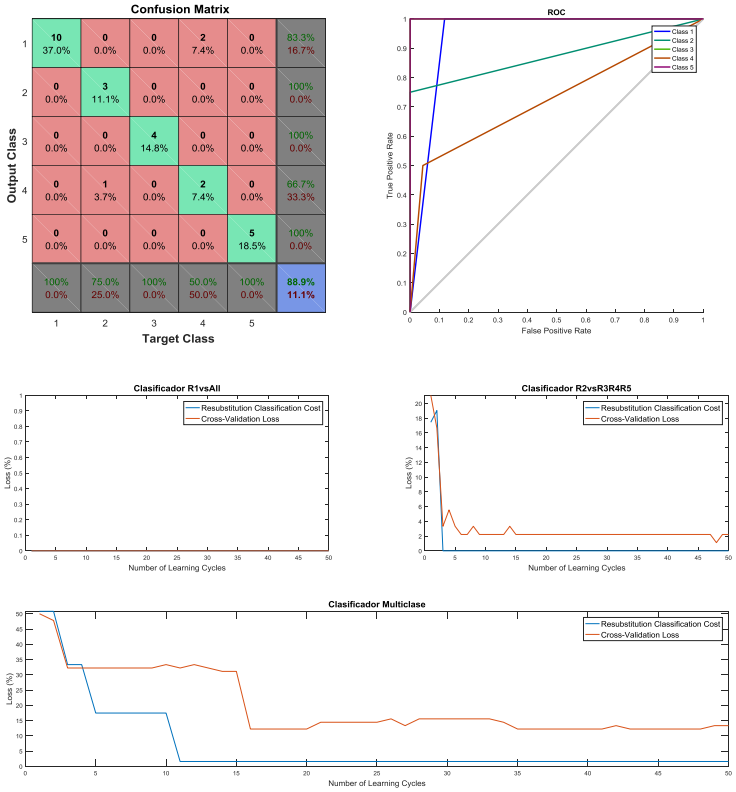


Figura A2-89: Ensayo mediante método 4 de clasificación. Semilla de valor 50. Clasificadores entrenado mediante muestra de entrenamiento completa.

2.2.4.4 Ensayo 4

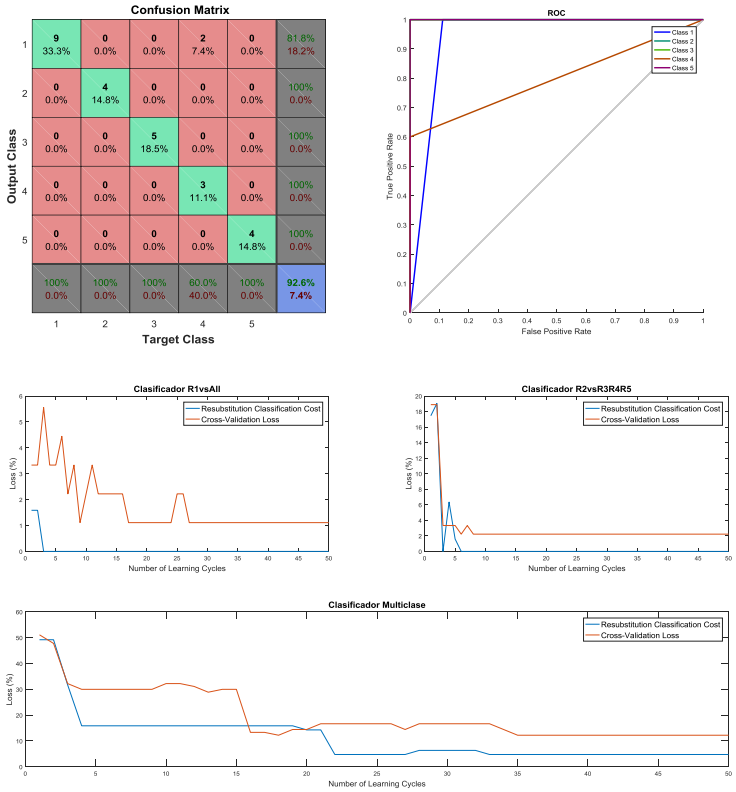


Figura A2-90: Ensayo mediante método 4 de clasificación. Semilla de valor 102. Clasificadores entrenado mediante

muestra de entrenamiento completa.

2.2.4.5 Ensayo 5

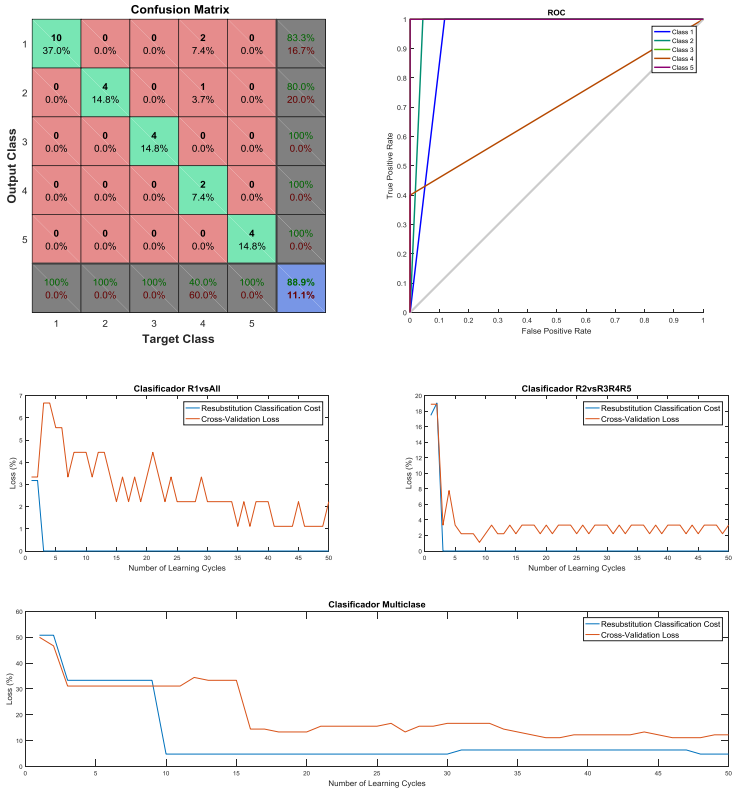
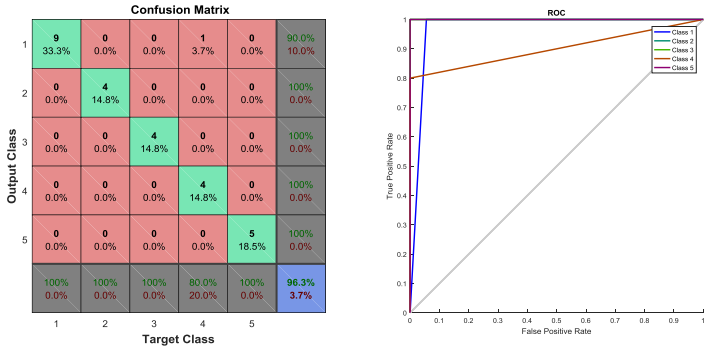


Figura A2-91: Ensayo mediante método 4 de clasificación. Semilla de valor 320. Clasificadores entrenado mediante muestra de entrenamiento completa.

2.2.5 Método 5

2.2.5.1 Ensayo 1



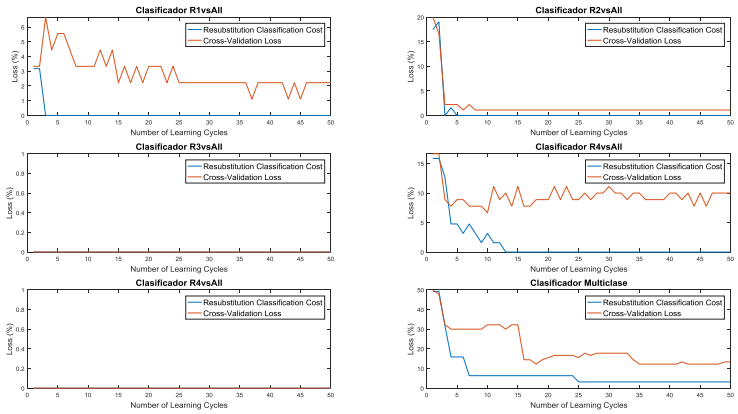


Figura A2-92: Ensayo mediante método 5 de clasificación. Semilla de valor 1.

2.2.5.2 Ensayo 2

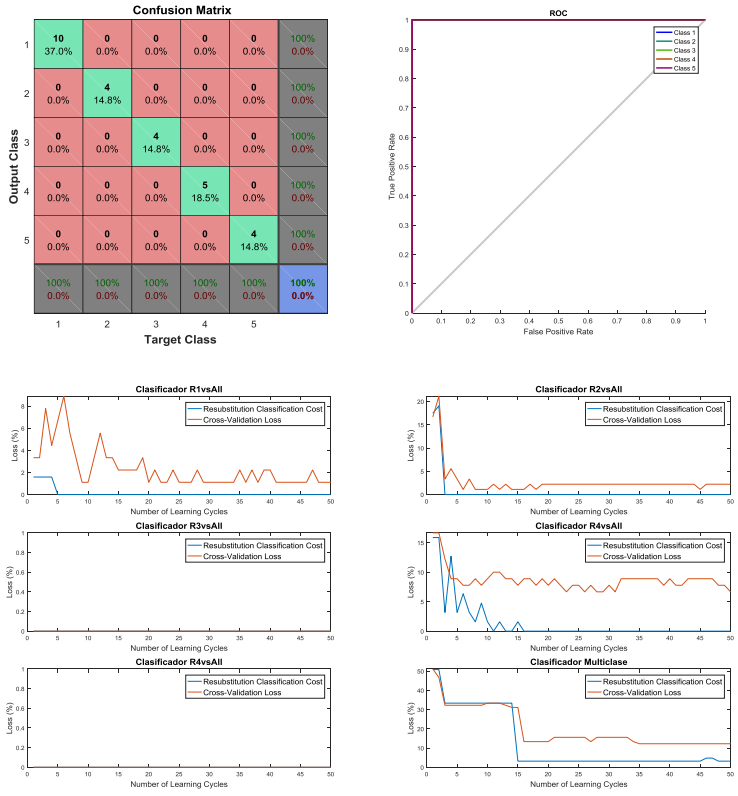
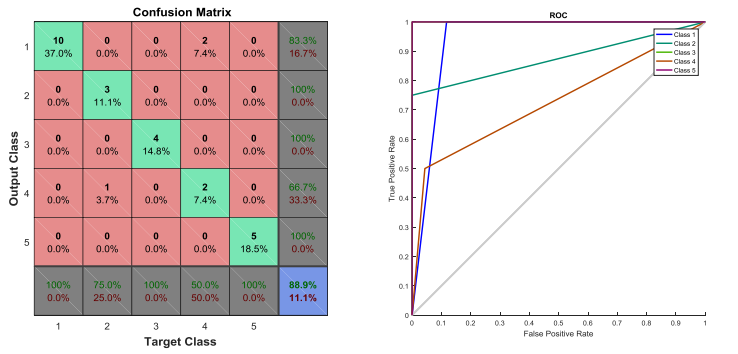


Figura A2-93: Ensayo mediante método 5 de clasificación. Semilla de valor 7.

2.2.5.3 Ensayo 3



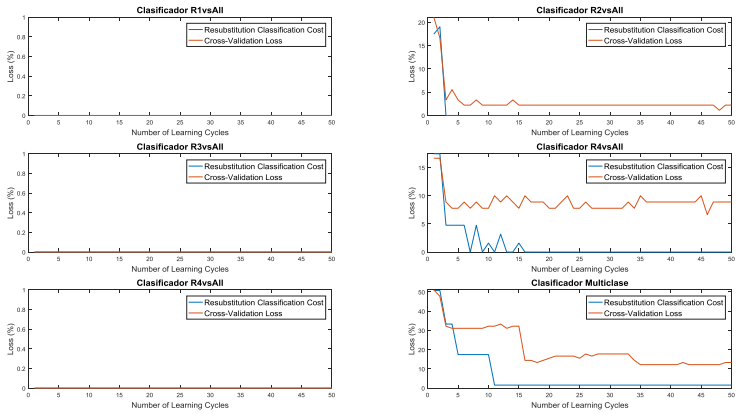


Figura A2-94: Ensayo mediante método 5 de clasificación. Semilla de valor 50.

2.2.5.4 Ensayo 4

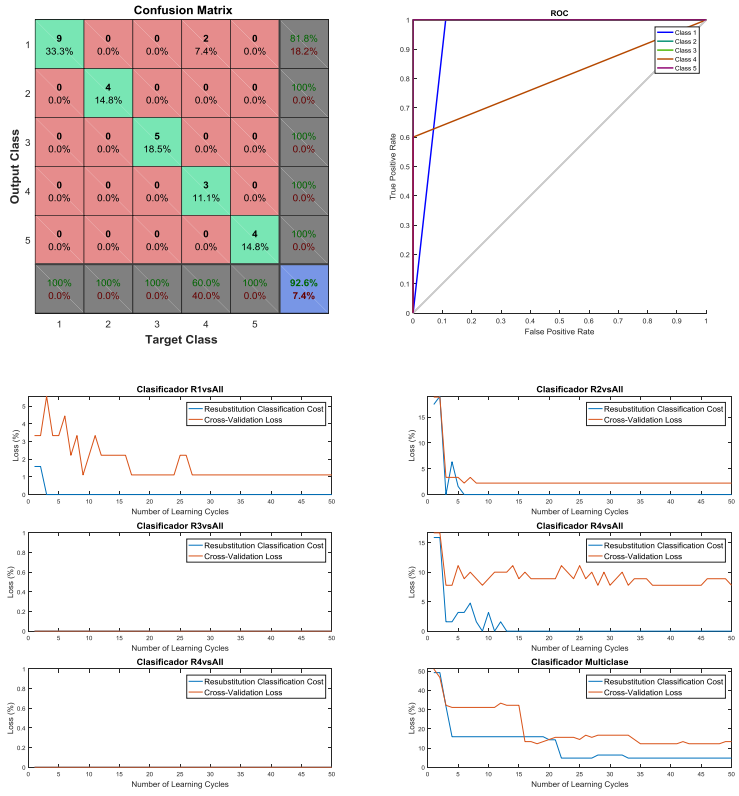
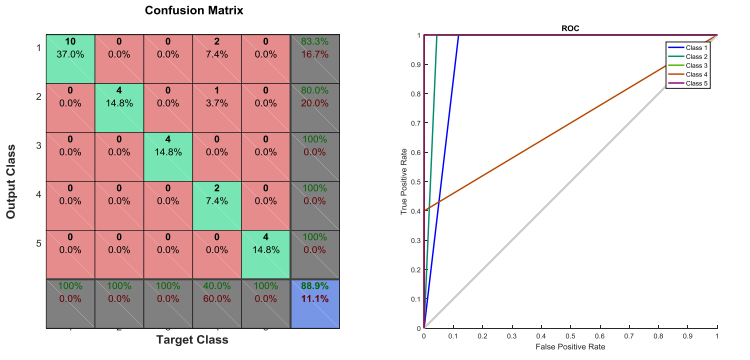


Figura A2-95: Ensayo mediante método 5 de clasificación. Semilla de valor 102.

2.2.5.5 Ensayo 5



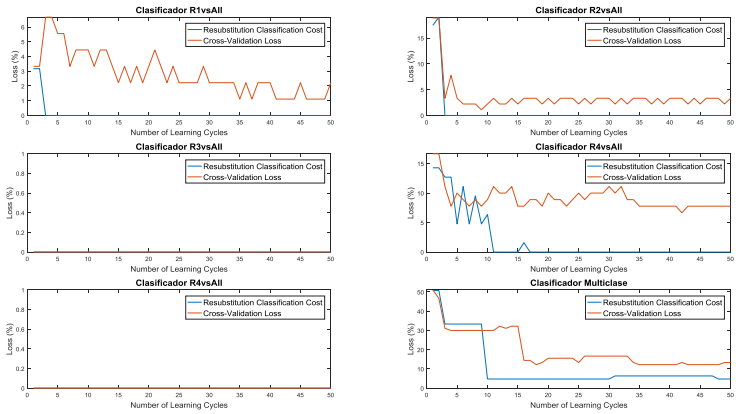


Figura A2-96: Ensayo mediante método 5 de clasificación. Semilla de valor 320.

2.2.6 Método 6

2.2.6.1 Ensayo 1

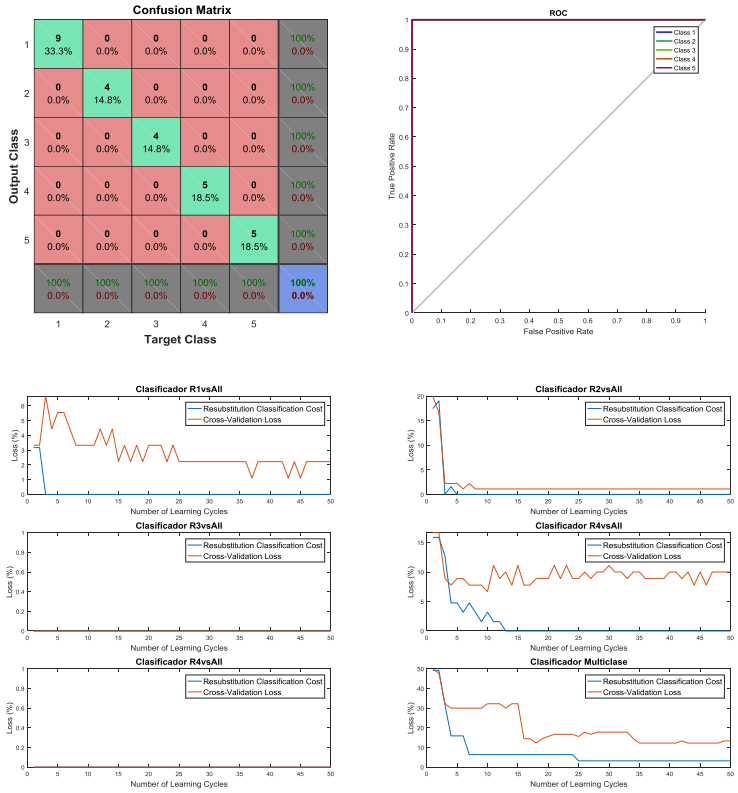


Figura A2-97: Ensayo mediante método 6 de clasificación. Semilla de valor 1.

2.2.6.2 Ensayo 2

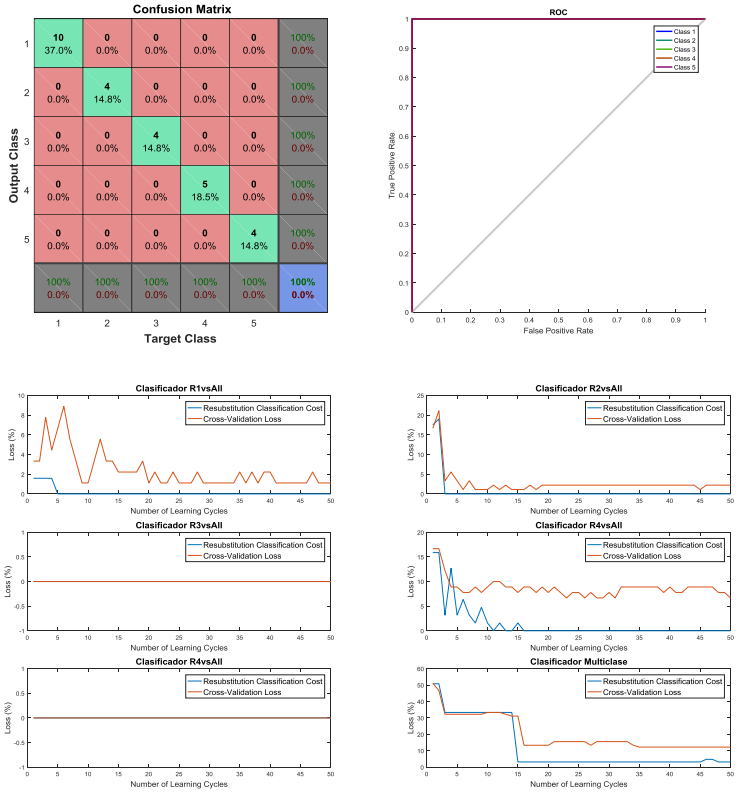


Figura A2-98: Ensayo mediante método 6 de clasificación. Semilla de valor 7.

2.2.6.3 Ensayo 3

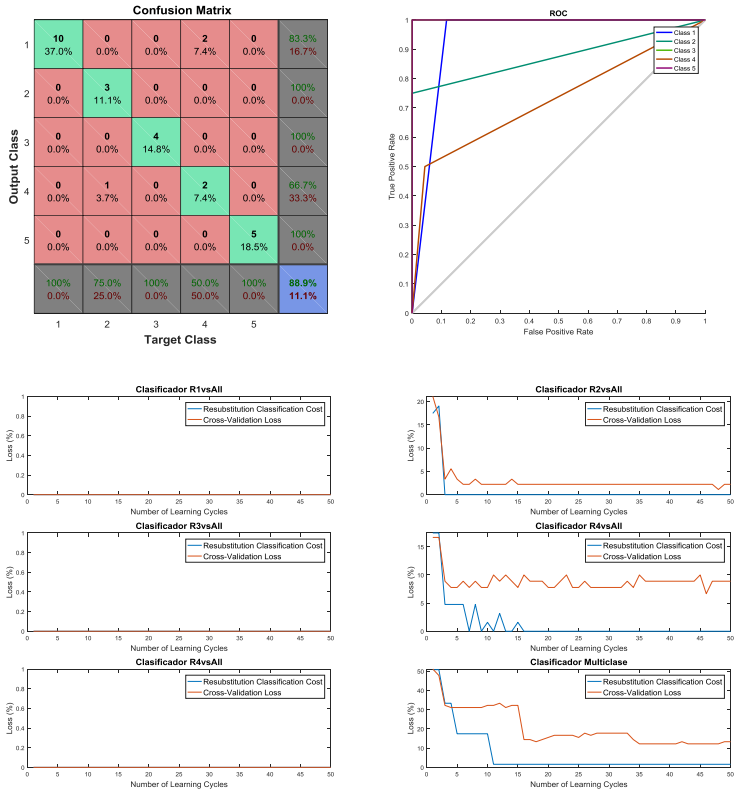


Figura A2-99: Ensayo mediante método 6 de clasificación. Semilla de valor 50.

2.2.6.4 Ensayo 4

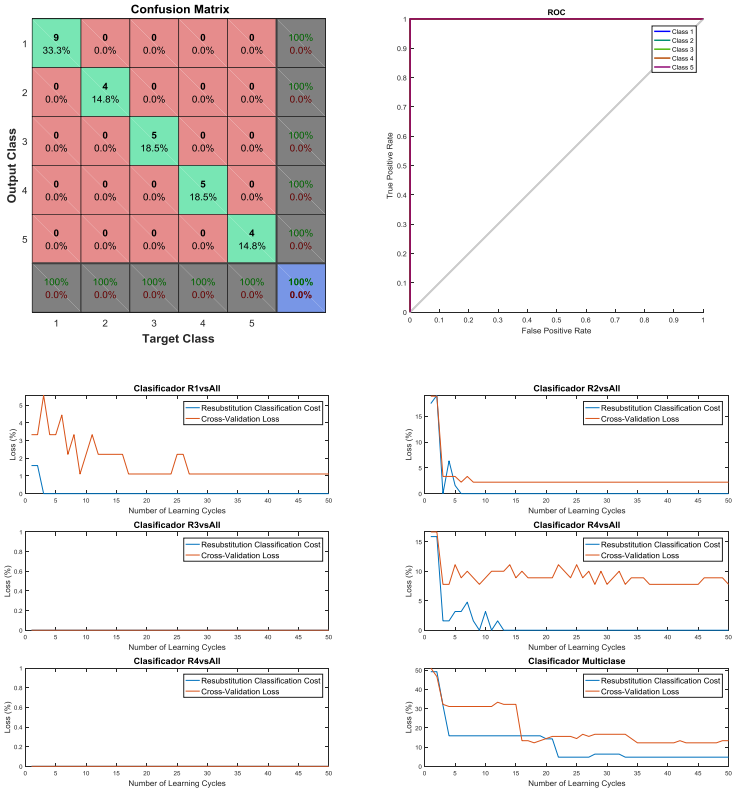


Figura A2-100: Ensayo mediante método 6 de clasificación. Semilla de valor 102.

2.2.6.5 Ensayo 5

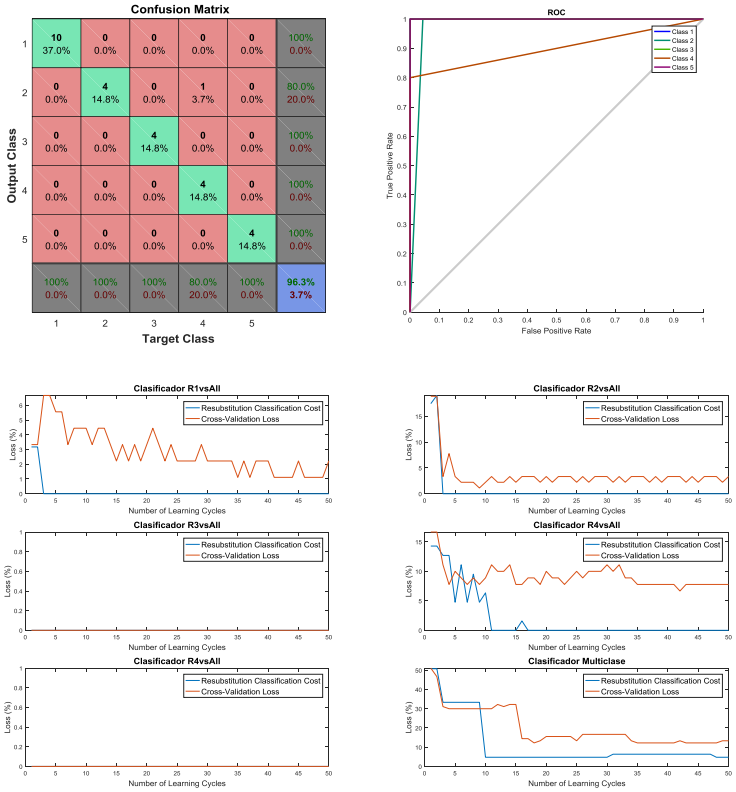


Figura A2-101: Ensayo mediante método 6 de clasificación. Semilla de valor 320.

2.2.7 Ensayos adicionales mediante el método 1

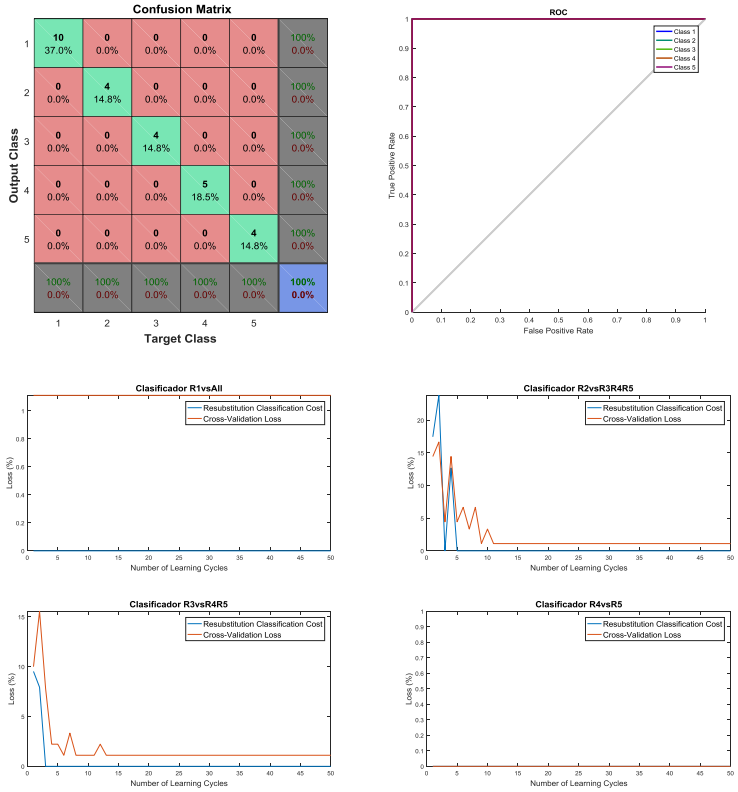


Figura A2-102: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Alimentación Red. Nivel de carga 2.

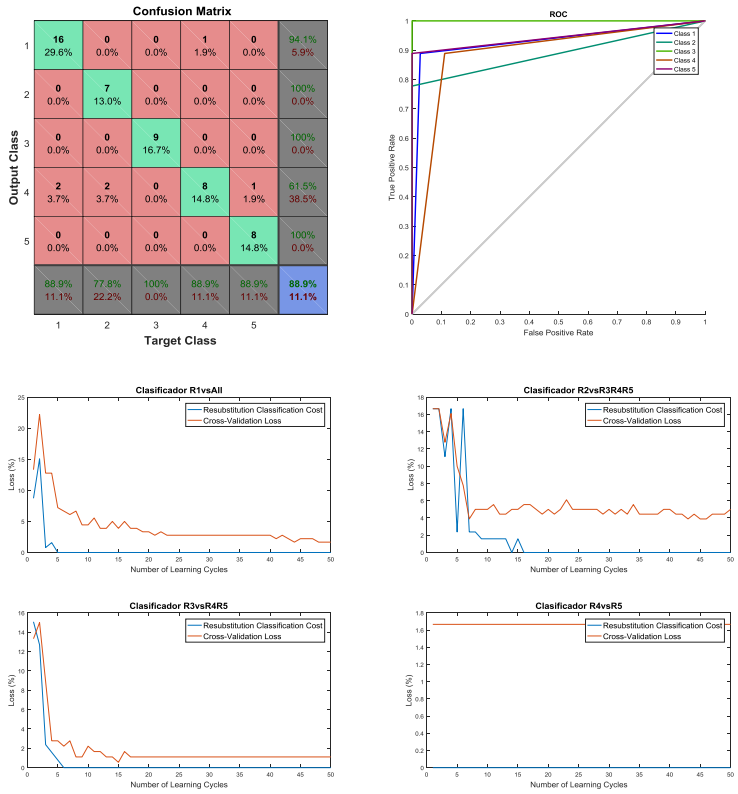


Figura A2-103: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Alimentación Red. Cualquier nivel de carga.

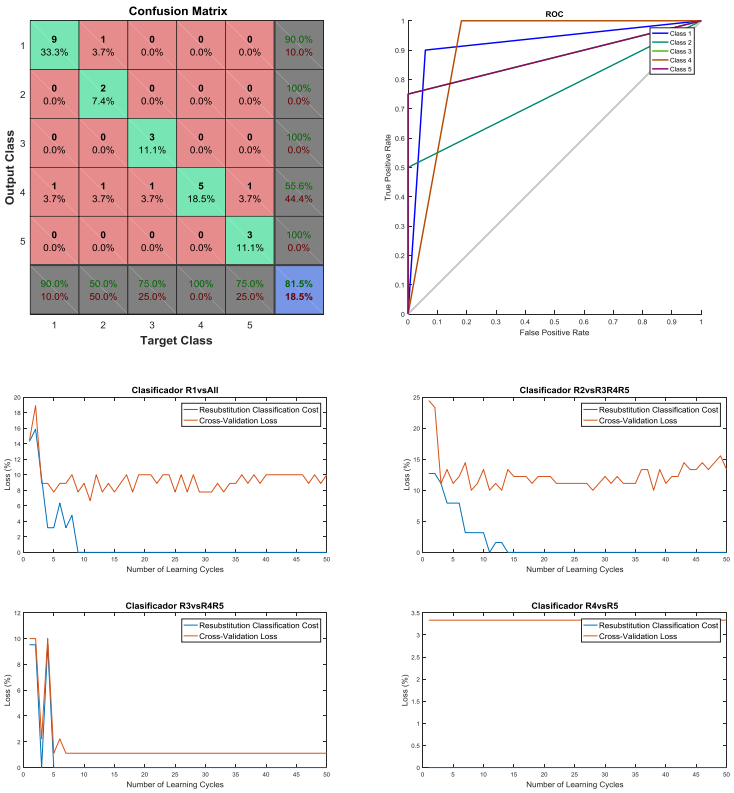


Figura A2-104: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Alimentación mediante convertidor. Nivel de carga 1.

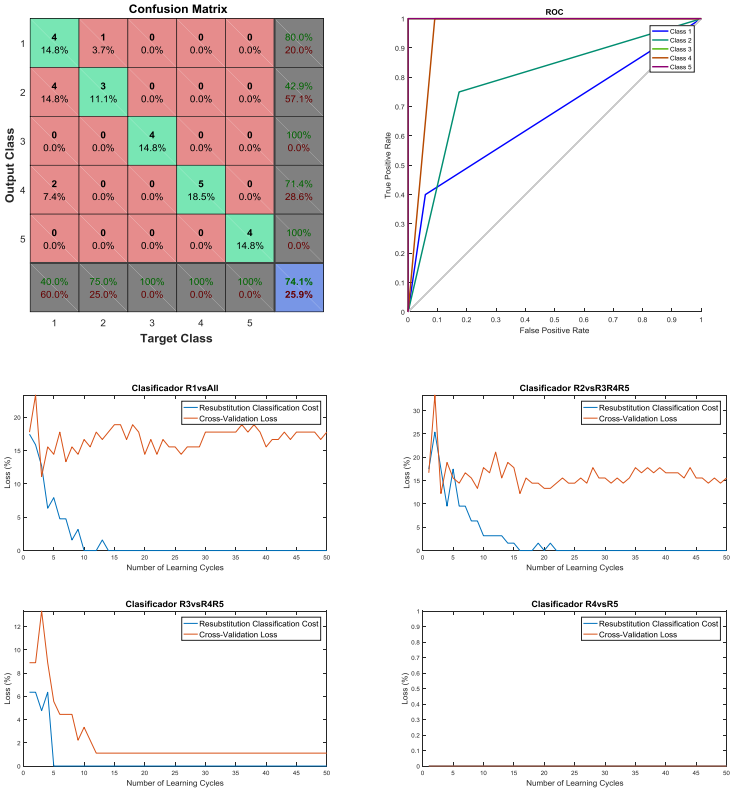


Figura A2-105: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Alimentación mediante convertidor. Nivel de carga 2.

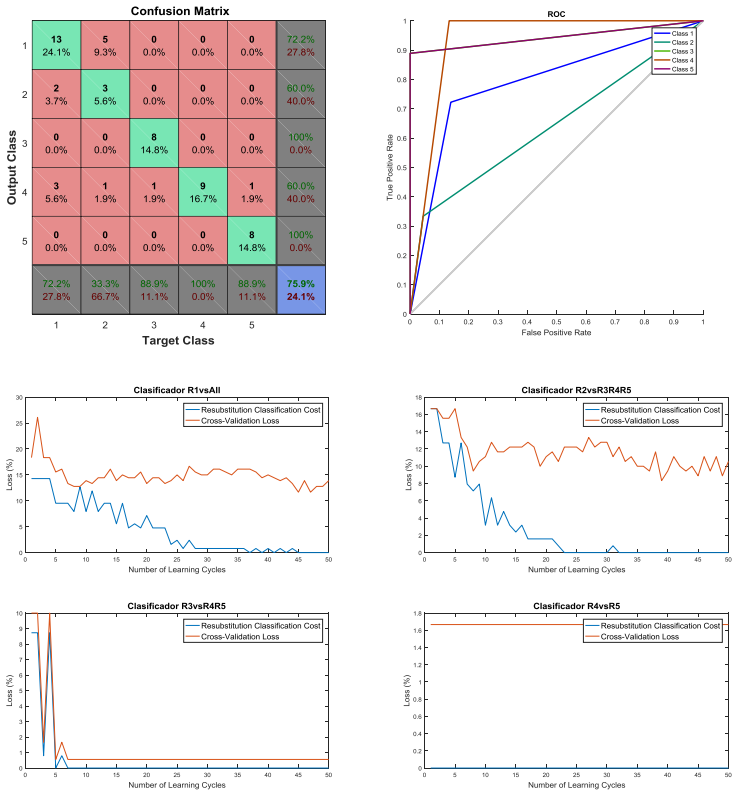


Figura A2-106: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Alimentación mediante convertidor. Cualquier nivel de carga.

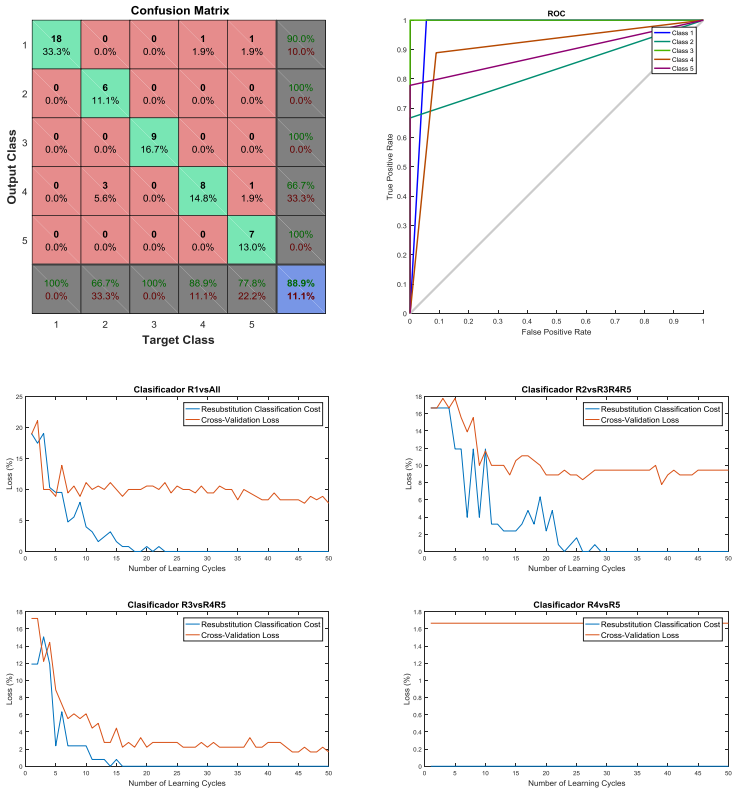


Figura A2-107: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Cualquier tipo de alimentación. Nivel de carga 1.

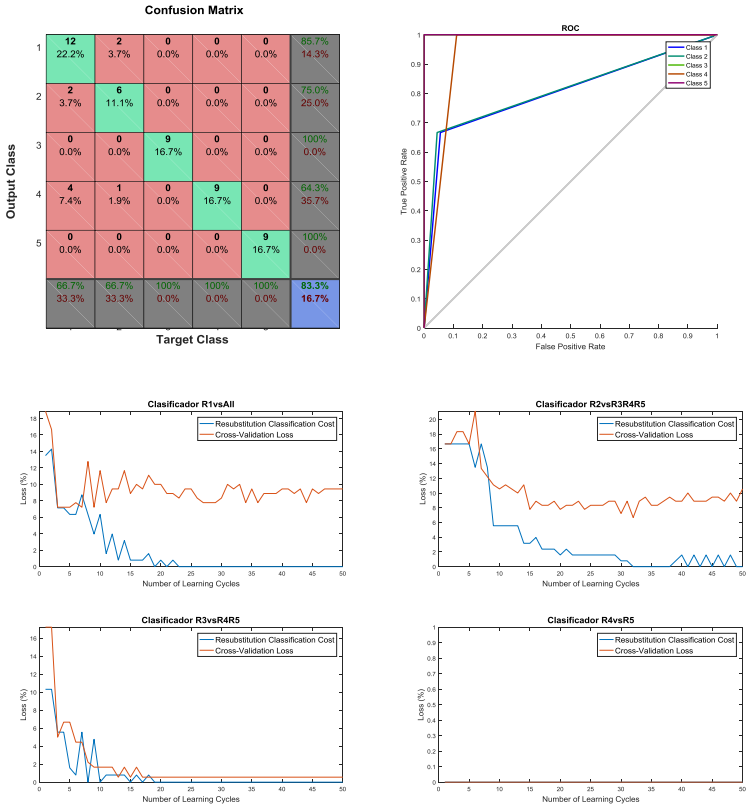


Figura A2-108: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Cualquier tipo de alimentación. Nivel de carga 2.

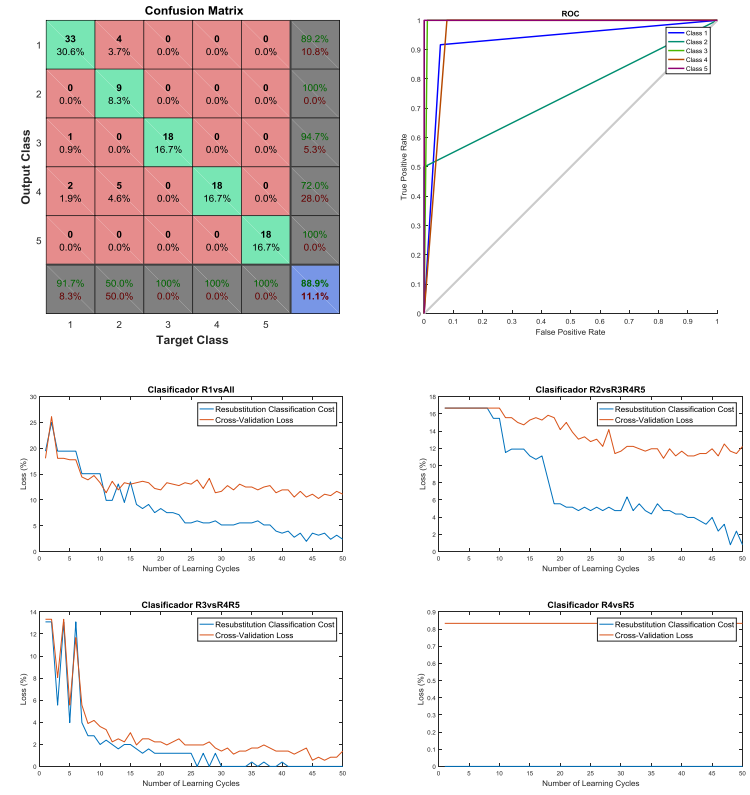


Figura A2-109: Ensayo mediante método 1 de clasificación. Semilla de valor 320. Cualquier tipo de alimentación. Cualquier nivel de carga.

2.2.8 Ensayos adicionales mediante el método 6

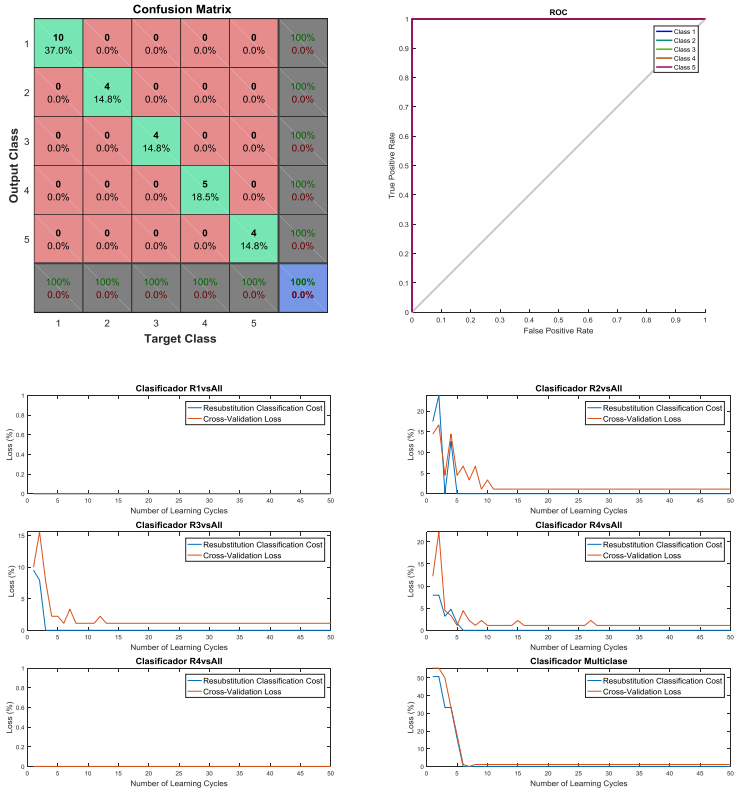


Figura A2-110: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Alimentación Red. Nivel de carga 2.

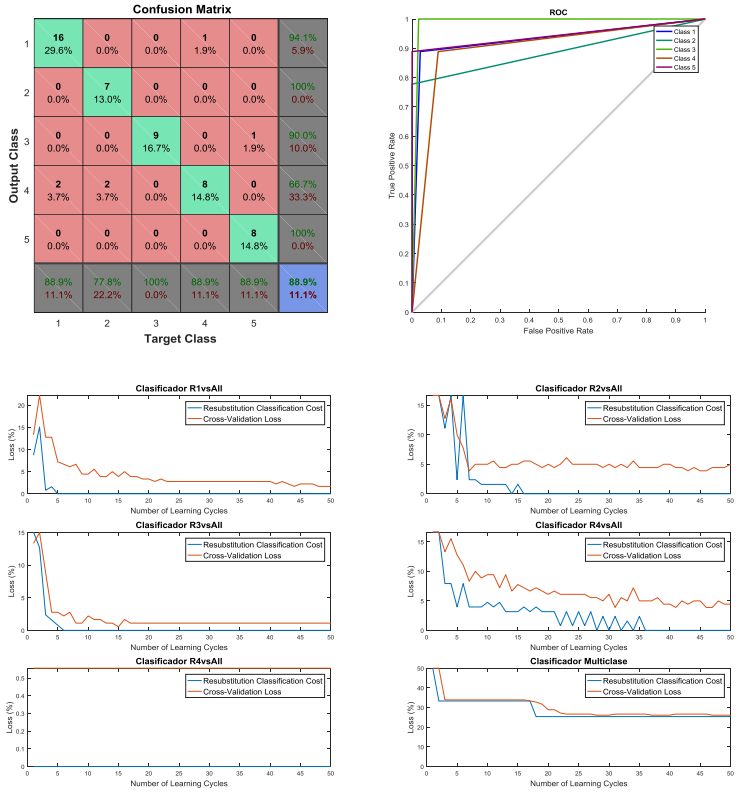


Figura A2-111: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Alimentación Red. Cualquier nivel de carga.

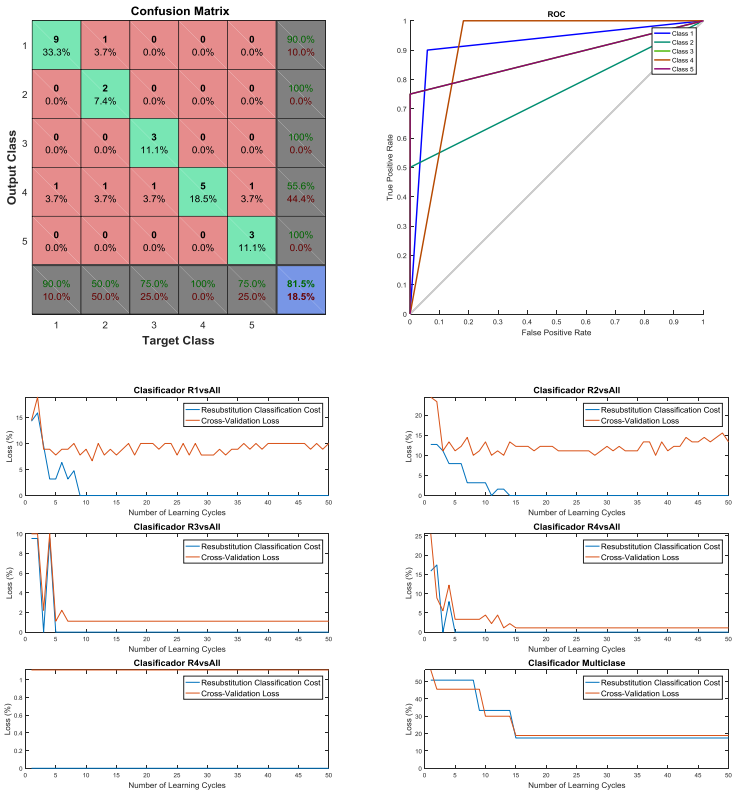


Figura A2-112: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Alimentación mediante convertidor. Nivel de carga 1.

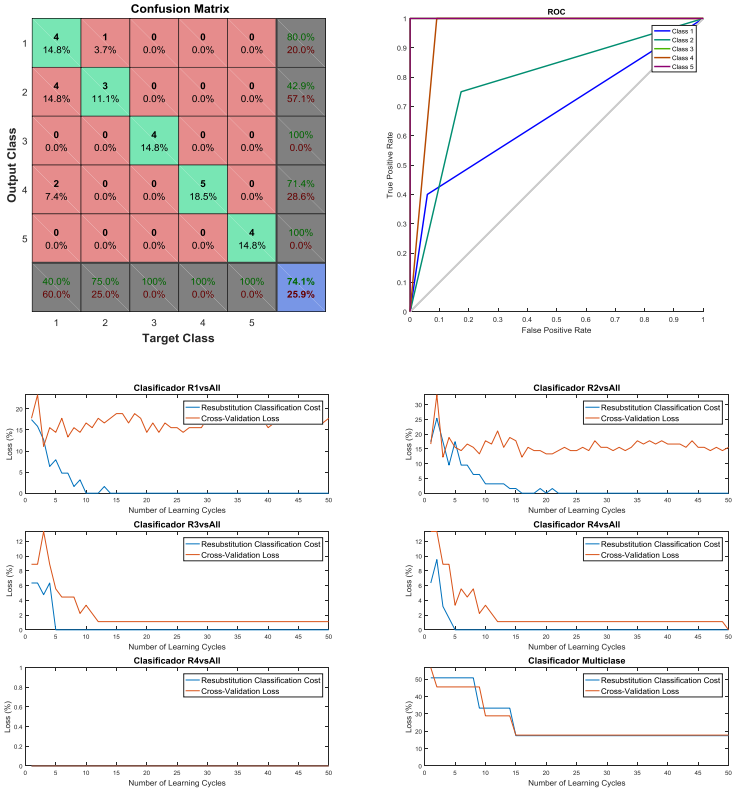


Figura A2-113: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Alimentación mediante convertidor. Nivel de carga 2.

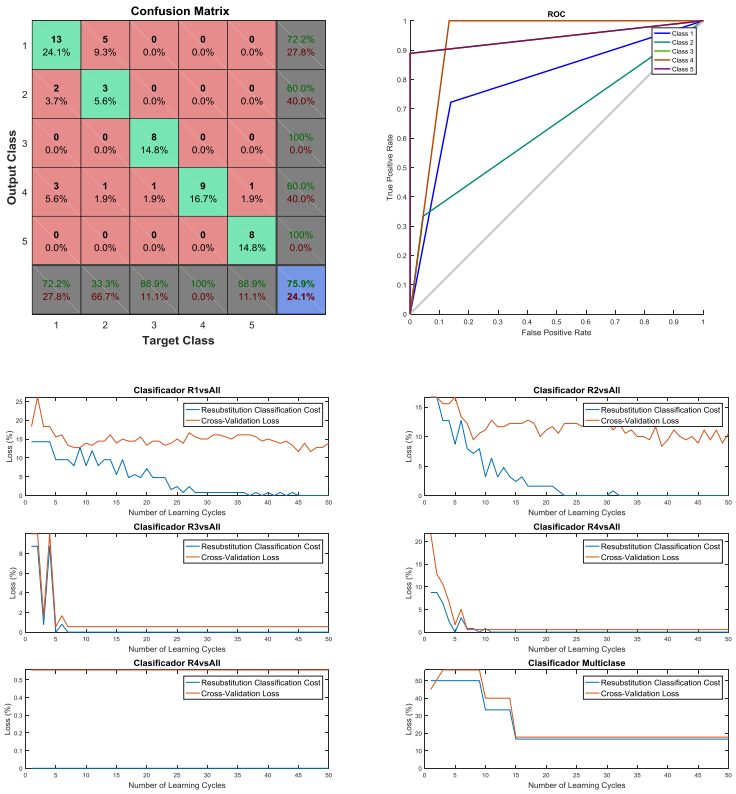


Figura A2-114: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Alimentación mediante convertidor. Cualquier nivel de carga.

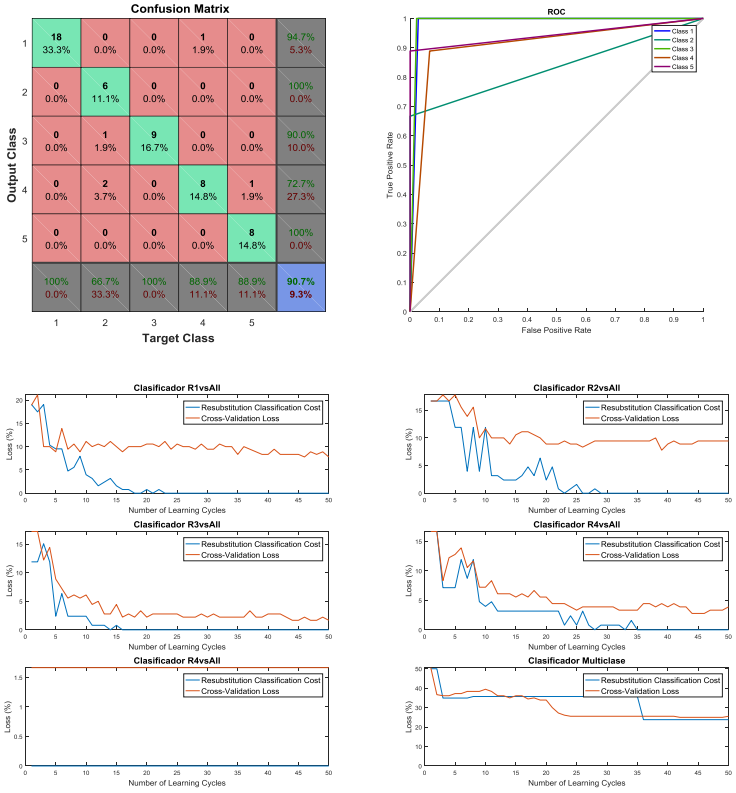


Figura A2-115: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Cualquier tipo de alimentación. Nivel de carga 1.

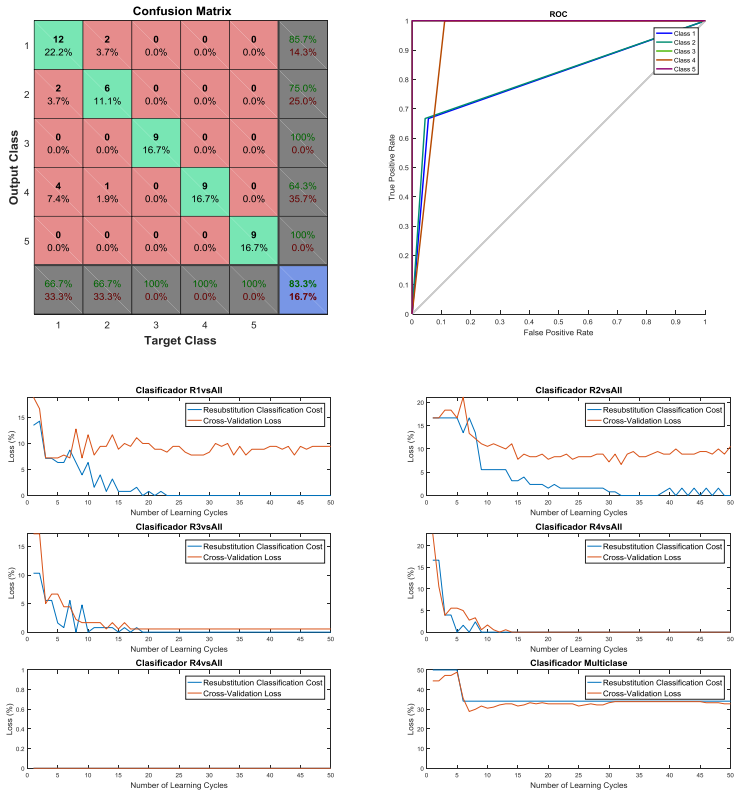


Figura A2-116: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Cualquier tipo de alimentación. Nivel de carga 2.

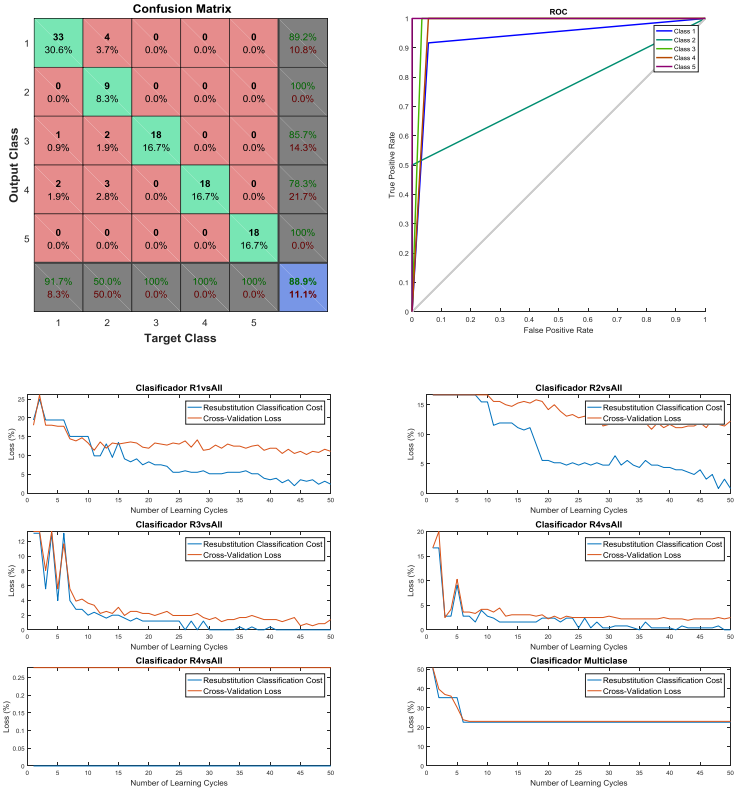


Figura A2-117: Ensayo mediante método 6 de clasificación. Semilla de valor 320. Cualquier tipo de alimentación. Cualquier nivel de carga.

2.2.9 Ensayos con tasa de aprendizaje de valor 0,1

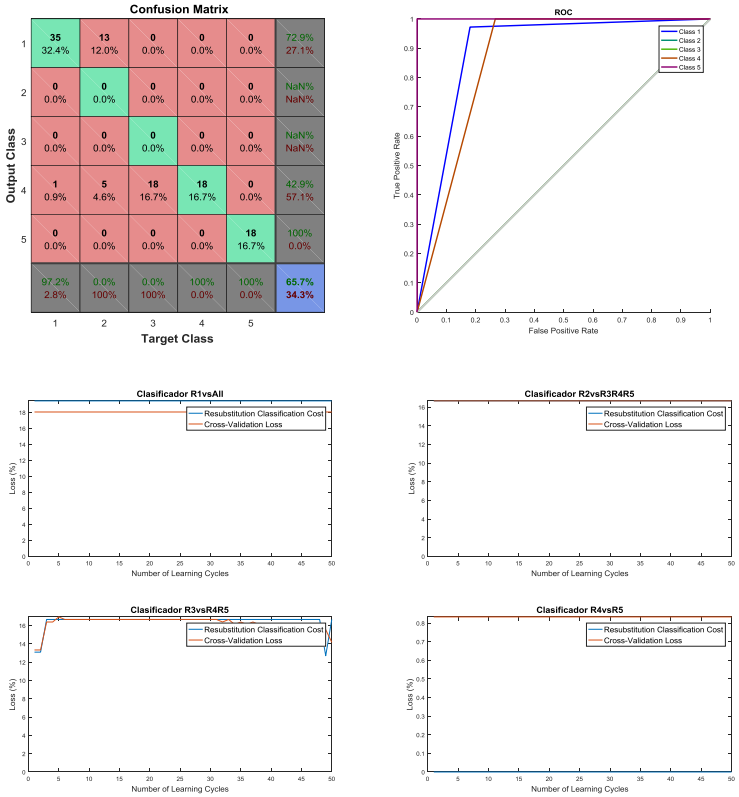


Figura A2-118: Ensayo mediante método 1 de clasificación con tasa de aprendizaje de valor 0,1. Semilla de valor 320. Cualquier alimentación. Cualquier nivel de carga.

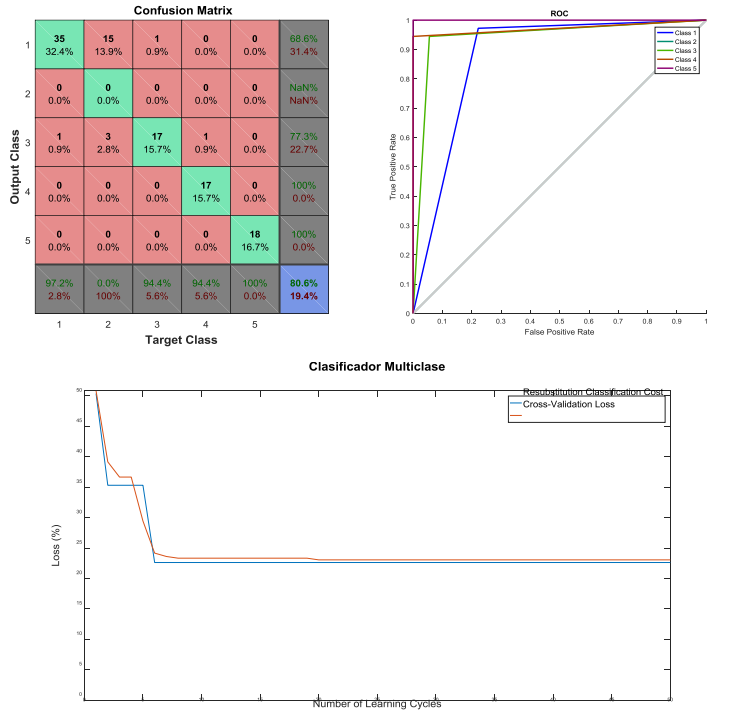


Figura A2-119: Ensayo mediante método 2 de clasificación con tasa de aprendizaje de valor 1. Semilla de valor 320. Cualquier alimentación. Cualquier nivel de carga.

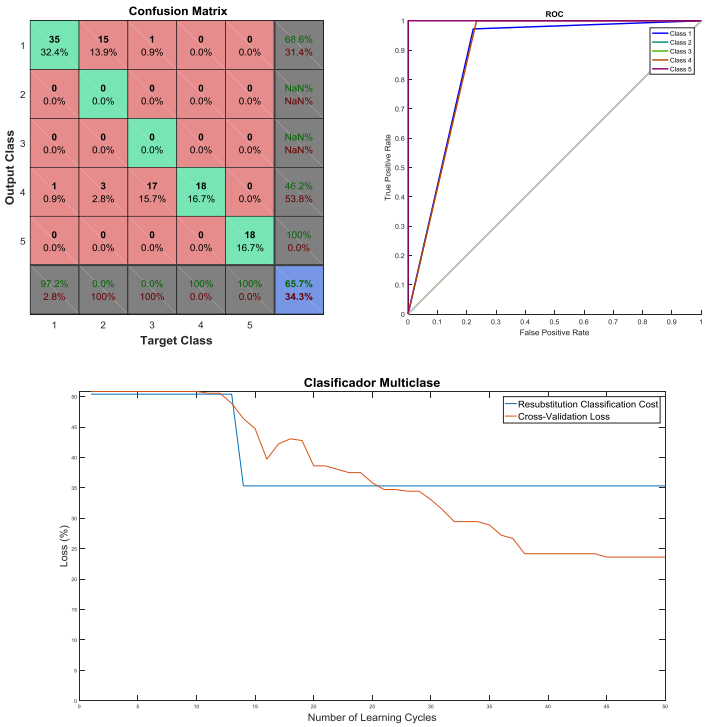


Figura A2-120: Ensayo mediante método 2 de clasificación con tasa de aprendizaje de valor 0,1. Semilla de valor 320. Cualquier alimentación. Cualquier nivel de carga.

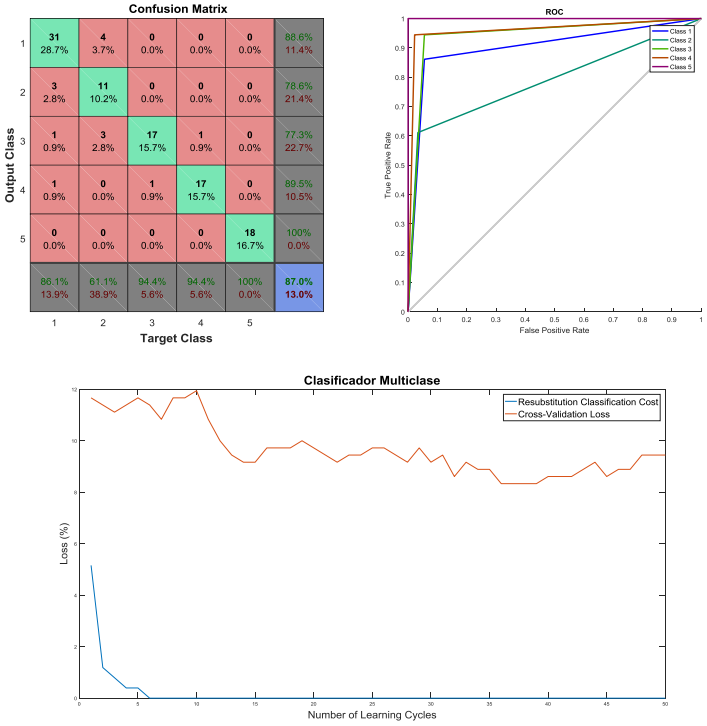


Figura A2-121: Ensayo mediante método 2 de clasificación con tasa de aprendizaje de valor 0,1 y máximo número de escisiones en el algoritmo de aprendizaje de 20. Semilla de valor 320. Cualquier alimentación. Cualquier nivel de carga.

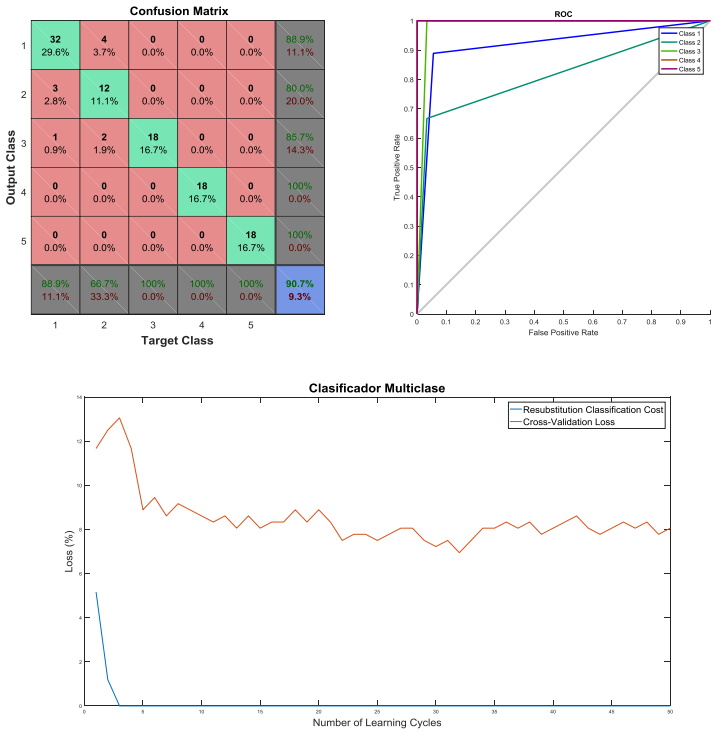


Figura A2-122: Ensayo mediante método 2 de clasificación con tasa de aprendizaje de valor 1 y máximo número de escisiones en el algoritmo de aprendizaje de 20. Semilla de valor 320. Cualquier alimentación. Cualquier nivel de carga.