Universidad de Valladolid
FACULTAD DE CIENCIAS

TRABAJO FIN DE GRADO

Grado en Estadística

Desarrollo de una aplicación web de Análisis Clúster robusto.

Autor:
Federico Pérez Rosado

Tutor:
Luis Ángel García Escudero

Índice

Resumen

La actual explosión de información ha dado lugar a un nuevo concepto conocido en la como Big Data, que a su vez ha despertado el interés por tomar decisiones objetivas basadas en los datos. Por ello, resulta de gran interés disponer de aplicaciones estadísticas accesibles para cualquier usuario y desde cualquier dispositivo y distintos entornos de trabajo.

Basado en herramientas de análisis estadístico como R o Python y aplicaciones de desarrollo web como Django, este trabajo desarrolla una aplicación de técnicas de análisis clúster robusto, permitiendo un manejo sencillo que cubre casos comunes para distintos usuarios. Se han utilizado las k-medias recortadas como técnica de Análisis Clúster capaz de resistir el efecto de una cierta proporción de observaciones anómalas.

Abstract

Nowadays, the explosion of data information has created a new concept frequently known as Big Data and has raised up the interest for taking objective decisions based on data. In consequence, there is a great interest on developing new applications, capable of approaching Big Data to any user and from any device.

Based on statistical analysis tools such as R or Python and web development applications such as Django, this study develops an application of robust cluster analysis techniques with easy functionality that solves common cases for different users. The trimmed k-means method has been applied as robust clustering technique because of its ability to resist certain fraction of outlying observations.

Palabras Clave

k-medias, k-medias recortadas, aplicación web, R, Python, Django

1- Introducción

En los últimos años debido al avance del sector tecnológico se ha producido un aumento constante del volumen de datos, dando lugar a un nuevo concepto conocido en la actualidad como Big Data. Existen un montón de artículos e información que le hacen referencia, pero es difícil comprender todo lo que conlleva el termino Big Data, para conocer más sobre el necesitamos remontarnos al momento de la llamada “explosión de la información”.

El término “explosión de la información” apareció por primera vez en la década de 1950 produciendo el inicio de la llamada Revolución Digital o Tercera Revolución Industrial, y el inicio a la transformación digital.

El nacimiento de la tecnología digital llevo de manera intrínseca la generación de datos que se produce con la interacción entre persona y máquina, lo que en la actualidad conocemos como el “Internet de las Personas”, que no es otra cosa que la digitalización de la vida de las personas.

La llegada de internet en 1969, la web en 1990 y los teléfonos inteligentes en la década del 2000 han sido importantes puntos de inflexión respecto al aumento del volumen de datos que generamos en la actualidad.

Según un estudio de la Facultad de Ciencias Sociales de la Universidad Nottingham Trent usamos de media el teléfono 5 horas al día, consultándolo unas 85 veces al día de media, con la consiguiente información que generamos, pero no únicamente desde el teléfono generamos un gran volumen de información, cada vez que realizamos una compra, cuando subimos al bus, cuando consultamos el saldo del banco, etc., es decir estamos hiperconectados y con la llegada de los objetos inteligentes el aumento de la cantidad de datos que generamos se acrecienta cada día.

Pero las personas no somos las únicas que generamos información, los llamados objetos digitales también generan información de ahí nace el concepto de “Internet de las Cosas”, haciendo referencia a la digitalización de los objetos físicos.

Por tanto, deducimos que el volumen de datos va siempre en constante aumento y de ahí nace la definición de Big Data.

> *El Big data o macrodatos es un término que hace referencia a una cantidad de datos tal que supera las capacidades del software tradicional para ser capturados, administrados y procesados en un tiempo razonable. El volumen de los datos masivos crece constantemente gracias al aumento y abaratamiento de los dispositivos de almacenamiento y procesado de éstos, que han posibilitado un gran avance en la digitalización de las empresas y de la sociedad en general.*
>
> -Según Datapedia LUCA

La generación de tal volumen de información produce la necesidad de analizar y comprender dicha información en todos los ámbitos de la sociedad que permita tomar decisiones “Data Driven” en base a hechos objetivos y no subjetivos. En torno a esta nueva tendencia surgen herramientas que junto a la capacidad computacional actual permiten manejar y analizar grandes volúmenes de información.

De este movimiento general nace la idea de conformar aplicaciones web capaces de aplicar técnicas de analítica avanzadas modernas sobre un conjunto de datos, permitiendo su uso en todo tipo de dispositivos. Para ello la ejecución de los algoritmos no se realizarán en el propio dispositivo de los usuarios sino se realizarán desde el entorno en el cual está desplegada la aplicación web, permitiendo realizar analíticas de alto coste computacional desde cualquier dispositivo.

En los próximos capítulos se abordará algunas técnicas agrupación o análisis clúster como el algoritmo de k-medias, una variación de este como es el procedimiento k-medias recortadas, así como su desarrollo en alguna de las herramientas de análisis estadístico más populares de la actualidad como son R y Python.

Partiendo de esta búsqueda de aplicaciones y herramientas, el objetivo de este trabajo de fin de grado es desarrollar una aplicación web funcional capaz de realizar técnicas de análisis clúster robusto sobre un conjunto de datos de manera sencilla e intuitiva para cualquier tipo de usuario.

2- Metodología

A lo largo de este capítulo se realizará una breve presentación del análisis clúster, sus objetivos y métodos, así como se realizará una exposición de la definición de robustez, la cual se tratará en capítulos posteriores.

2.1 Análisis Clúster

El análisis clúster o como se denomina en otros entornos, clustering, reconocimiento de patrones, taxonomía numérica, morfométrica o clasificación no supervisada, es una técnica estadística multivariante que tiene el objetivo principal de agrupar observaciones, datos o individuos similares basándose en sus características. De tal forma que se obtiene la máxima homogeneidad posible entre las observaciones que conforman un mismo grupo y maximizando la diferencia entre grupos.

El objetivo de las técnicas clúster no posee un único objetivo de agrupación, sino que trata de dar solución a problemas como:

- Obtener una mejor visión de los individuos.
- Encontrar agrupaciones o individuos similares.
- Encontrar patrones en los datos.
- Generar hipótesis.
- Obtener modelos de clasificación para futuras observaciones que se incluyan en un dataset

Es aplicable una gran cantidad de campos o áreas de trabajo, como, por ejemplo:

- Arqueología	- Antropología	- Agricultura	
- Economía	- Educación	- Geografía	
- Geología	- Genética	- Telecomunicaciones	
- Medicina	- Ciencias políticas	- Análisis de mercados	
- Psiquiatría	- Psicología	- Sociología	...

Debido a los diferentes enfoques existen distintos procedimientos de agrupación que se categorizan en 2 tipos de métodos principalmente:

- Métodos Jerárquicos
- Métodos No Jerárquicos

Métodos Jerárquicos

Los procedimientos de tipo jerárquicos tienen como objetivo agrupar clústers para formar uno nuevo o dividir uno ya existente en dos nuevos clústers. Realizando este proceso de manera sucesiva se minimiza alguna distancia o se maximiza alguna medida de similitud, por ejemplo, el método de Ward, método de la mediana… El resultado de estas técnicas de clasificación es construir una sucesión de particiones encajadas.

El dendograma es la representación utilizada en los métodos jerárquicos dado que permite visualizar la jerarquía de los grupos de acuerdo con la distancia existente entre las observaciones.

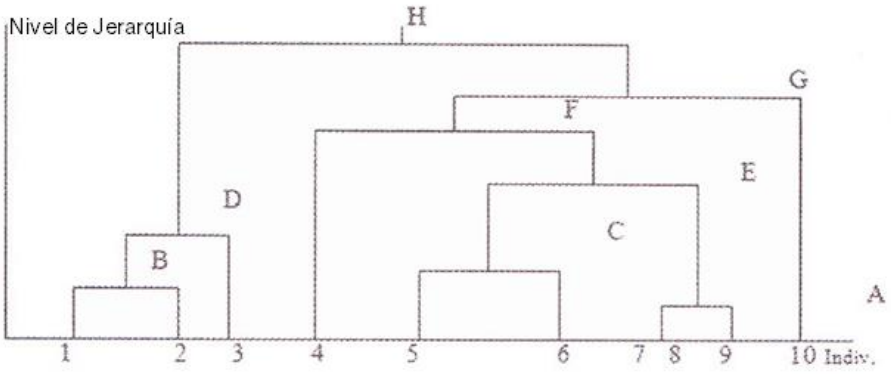


Figura 1: Dendograma sobre una muestra de 10 individuos.

Los métodos Jerárquicos a su vez se dividen en aglomerativo o ascendentes y disociativos o descendentes.

- Método Aglomerativo o Ascendente

El proceso Aglomerativo o ascendente se inicia con tantos grupos como observaciones existan, posteriormente se realizan fusiones sucesivas hasta obtener un único grupo que contenga todas las observaciones.

- Método Disociativo o Descendente

El proceso disociativo o descendente realiza un procedimiento inverso al anterior. Se inicia con un único grupo que contiene todas las observaciones, posteriormente se realizan divisiones de manera consecutiva hasta obtener un numero de grupos igual al número de observaciones.

Métodos No Jerárquicos

Los métodos no jerárquicos también denominados de optimización únicamente producen una sola partición. Por ello es necesario especificar el número de clústers a priori, existen diferentes metodologías como los métodos de reasignación o de agregación entorno a los centros móviles, como por ejemplo el método de las nubes dinámicas, el método de Forgy o métodos basados en modelos (“model-based” clustering) mediante verosimilitud. El método de k-medias será descrito con mayor detalle en próximas secciones.

2.2 Robustez Estadística

Frecuentemente, en el análisis de datos es habitual la existencia de algunas observaciones que poseen un comportamiento muy distinto dentro de un conjunto de datos. Estos valores denominados comúnmente como outliers o valores atípicos pueden afectar de manera significativa en los análisis debido a que poseen una gran influencia en algunos estimadores o procedimientos estadísticos clásicos.

Un ejemplo es el cálculo de la media, los datos atípicos tienen una influencia mayor que los datos cercanos a la media, por ello un único valor puede influenciar enormemente en el cálculo de la media. Lo siguiente es ejemplo de ello:

> *En una muestra* $X = \{0, 56, 61, 57, 58.5\}$ *en la que se observa un valor atípico, en este caso el 0, el valor de su media es* $\bar{X} = 46.5$

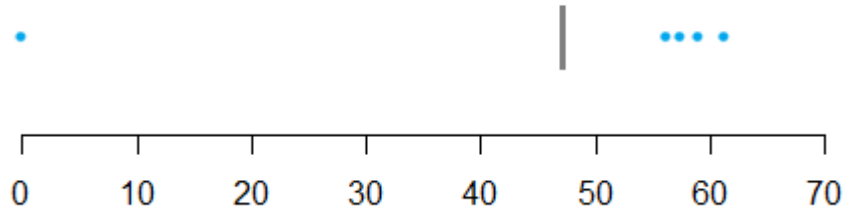


Figura 2: Representación de la media y las observaciones de la muestra X.

Como se puede observar a simple vista, el outlier altera el resultado de manera significativa. Ante esta problemática existen dos opciones principalmente:

- Utilizar procedimientos estadísticos robustos, que se caracterizan por no verse afectados de manera significativa o relativa por la presencia de outliers o variables atípicas. Un ejemplo de estimador robusto es la mediana.

> *En una muestra* $X = \{0, 56, 61, 57, 58.5\}$ *en la que se observa un valor atípico, en este caso el 0, el valor de su mediana* $\tilde{X} = 57$

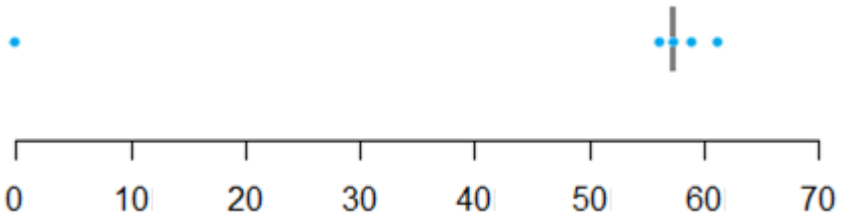


Figura 3: Representación de la mediana y las observaciones de la muestra X.

- Detectar los outliers o valores atípicos de la muestra y eliminarlos o tratarlos para que no produzcan alteraciones significativas en la muestra. A continuación, este procedimiento se realiza en la muestra analizada anteriormente.

En una muestra $X = \{0, 56, 61, 57, 58.5\}$ *vamos a detectar si existe algún valor atípico*

Se calcularán los valores atípicos a partir de un gráfico de cajas mediante el cálculo de los cuartiles Q_1, Q_2 *y* Q_3 *, el rango intercuartílico* RIC *y los "bigotes" inferior y superior*

$Q_1 = 52.25$
$Q_2 = \tilde{X} = 57$
$Q_3 = 58.5$

$RIC = (Q_3 - Q_1) = 58.5 - 56 = 2.5$
$inf = (Q_1 - 1.5 * RIC) = 52.25$
$sup = (Q_3 + 1.5 * RIC) = 58.25$

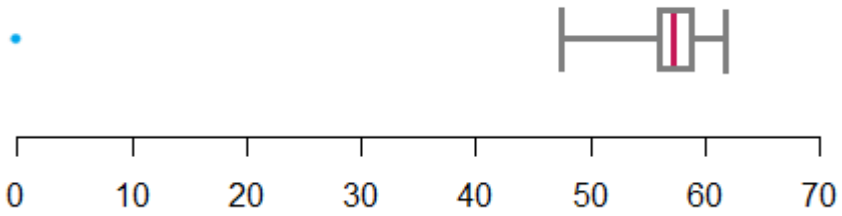


Figura 4: Diagrama de cajas de la muestra X

Considerando el valor 0 como atípico, vamos a suprimirlo y como su nueva media $\bar{X} = 58.125$ *es mas representativa para la muestra.*

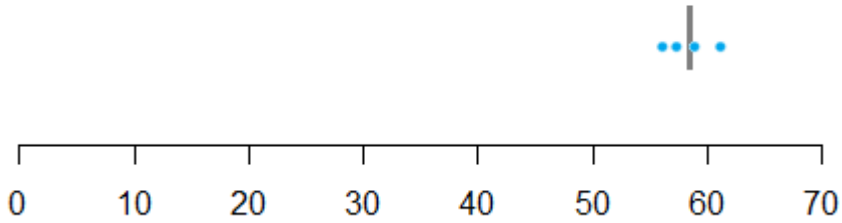


Figura 4: Representación de la media y las observaciones de la muestra X sin el outliers.

Como se evidencia en los ejemplos anteriores tanto con el uso de un estimador robusto, como eliminando los outliers mediante procedimientos robustos adecuados se produce una mejora respecto al cálculo de la media inicial, obteniendo un estimador más representativo de la muestra.

En algunas ocasiones detectar los valores atípicos es muy complejo cuando existen unas grandes cantidades de datos o frente a datos multivariantes.

Con la definición de robustez en los próximos capítulos se aplicarán mejoras sobre el algoritmo k-medias dando lugar a un proceso más robusto.

2.3 Medias Recortadas

Las medias recortadas tienen la finalidad de evitar distorsiones que provocan los valores atípicos como se comenta en el capítulo anterior, para ello se recorta un determinado porcentaje α de las observaciones situadas en los extremos.

> *En una muestra* $X = \{0, 56, 61, 57, 58.5\}$ *y* $\alpha = 0.4$*, se calcula su media recortada.*
>
> *Con el* $\alpha = 0.4$ *se recortan* $\alpha * n = 0.4 * 5 = 2$ *datos, es decir, una observación en cada extremo de la muestra. En la muestra los valores extremos son 0 y 61, calculándose la media con las observaciones restantes.*
>
> $$\bar{X}rec = 57.17$$

Se evidencia en este ejemplo que la media recortada X con el $\alpha = 0.4$, sí logra evitar la distorsión que los valores atípicos que causan en la media. Pero este estimador no ofrece garantía de robustez, por ejemplo, si el valor fijado es $\alpha = 0$, la media recortada es equivalente a la media aritmética, y por tanto el estimador puede verse influenciado por los valores atípicos. Recortes más grandes de lo necesario pueden provocar perdida de la eficiencia del método estadístico aplicado.

Aplicar estos recortes es complicado en problemas multivariantes donde no existe un orden natural como ocurre con datos multivariantes en $\mathbb{R}$ y resulta más complejo en análisis clúster donde puede ser necesario eliminar valores atípicos situadas entre grupos (“Bridge-points”) y que no son atípicos en ninguna de las variables.

3- K-medias

Los métodos de agrupación como el procedimiento k-medias surgieron en base a las investigaciones de aspectos teóricos y algorítmicos basados en la suma de los cuadrados (Cox 1957 y Fisher 1958) y fue tomando una mayor relevancia durante las décadas de 1960 y 1970 debido a la publicación del estudio “Principles of numerical taxonomy” (Sokal y Sneath 1963). Dicha publicación dio lugar a amplio número de investigaciones y artículos centrados en el estudio de los métodos de agrupación como el método de Forgy (1965) que constituyo la aproximación más simple al clustering particional y el método de k-medias de McQueen (1967) que utiliza el concepto de centroides.

3.1 Procedimiento k-medias

El algoritmo k-medias es una técnica no jerárquica, que trabaja de forma iterativa asignando las observaciones a cada uno de los k grupos, y se asignan basándose en la distancia del entre la observación y el centro del grupo en base a sus características, de tal forma que tras un numero de iteraciones suficientes las observaciones pertenecientes a un mismo grupo sean los más homogéneos posibles y lo más distinto posible a las observaciones de otros grupos.

3.2 Algoritmo k-medias

Sea $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ una muestra, los pasos necesarios para llevar a cabo el algoritmo k-medias son:

Paso 0

Determinar un conjunto inicial de k centroides aleatoriamente. Algunos de los procedimientos más habituales para obtener estos centroides son:

- Usar una submuestra de k observaciones elegidas al azar.
- Asignar las n observaciones de manera aleatoria a los k grupos y seleccionar las medias de las observaciones en estos.

Paso 1

Asignar cada observación al centroide más cercano.

- Siendo $\{\boldsymbol{m}_1, \boldsymbol{m}_2, \dots, \boldsymbol{m}_k\} \subset \mathbb{R}^d$ los k centroides fijados en el paso anterior, por cada observación se calcula la distancia euclídea al k centroide más cercano:

$$d_i = \min_{j=1,\dots k} \|x_i - m_j\|, \qquad i = 1, \dots n.$$

- Cada observación se asigna al centroide más cercano basándose en la distancia euclídea:

$$H_j = \left\{ i: \left\| x_i - m_j \right\|^2 = \min_{1 \le j \le k} \left\| x_i - m_j \right\|^2 \right\}, \qquad j = 1, \dots, k$$

La muestra X queda particionada en los k grupos $\{ H_1, \dots H_k \}$

Paso 2

Calcula los nuevos centroides como la media de las observaciones que han sido asignadas a cada uno de los grupos:

$$m_j = \frac{1}{n_j} \sum_{i \in H_j} x_i \qquad j = 1, \dots, k$$

Donde n_j es el total de observaciones en H_j , $j = 1, \dots, k$

El paso anterior de asignación de las observaciones y cálculo de nuevos centroides se realizan de manera iterativa para evaluar en la función objetivo:

$$\sum_{i=1}^{n} \left(\begin{matrix} inf \\ 1 \le j \le k \end{matrix} \left\| x_i - m_j \right\| \right)^2$$

o análogamente la función objetivo es:

$$\sum_{j=1}^{k} \sum_{x_j \in H_k} \left\| x_i - m_j \right\|^2$$

Posteriormente, se denominará en el desarrollo del algoritmo a estas iteraciones como “Ksteps”.

Es necesario inicializar muchas veces el algoritmo desde el paso inicial para almacenar soluciones y utilizar aquella que minimice la función objetivo, estos reinicios se denominaran posteriormente en el desarrollo como “Niter”.

Uno de los métodos utilizados para detectar el número de clústers o validar que son correctos es el método del “codo”, que emplea los valores de la inercia, que es el valor de la función objetivo, para determinar el número de clústers óptimo

El método de “codo” se representa mediante un gráfico de línea en el cual se detectará el número de clústers optimo cuando se visualice un cambio significativo (debido a la estabilización) en la evolución de la inercia.

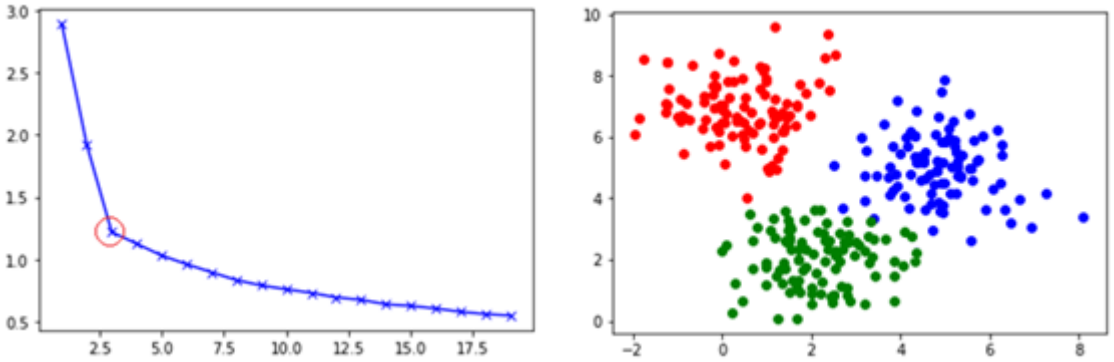


Figura 5: Ejemplo del gráfico del “codo” sobre un conjunto de datos con 3 clústers.

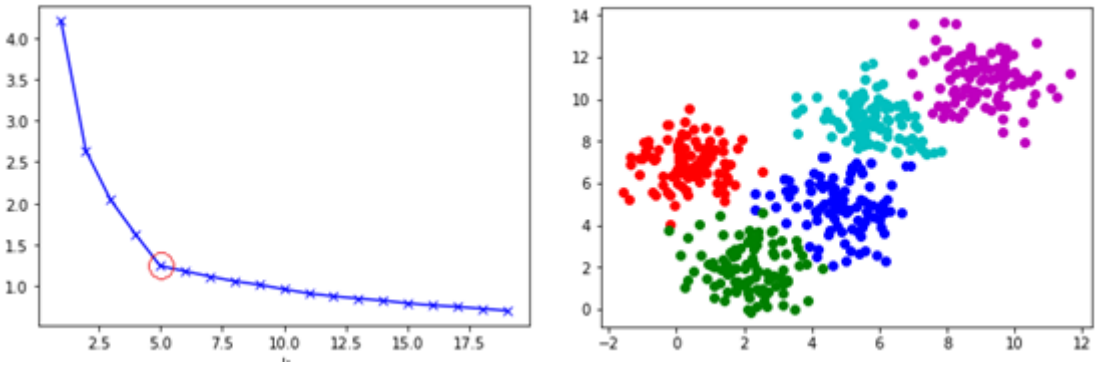


Figura 6: Ejemplo del gráfico del “codo” sobre un conjunto de datos con 5 clústers.

4- K-medias recortado

Esta variación del método clásico trata así de corregir la falta de robustez del procedimiento clásico k-medias dejando una proporción de observaciones (las más atípicas) sin clasificar, tal y como se detalla en las publicaciones “Trimmed K-means: An attempt to robustify quantizers” (Cuesta-Albertos, Gordaliza & Matrán 1997) y “Robustness Properties of k medias and trimmed k-medias” (García-Escudero & Gordaliza 1999).

4.1 Procedimiento k-medias recortado

El *procedimiento de k-medias recortado* se basa en el anterior procedimiento k-medias y en la metodología de "recorte imparcial", visto como una forma de obtener un conjunto no-recortado con la menor “variación posible” y un tamaño de recorte α.

4.2 Algoritmo k-medias recortado

Sea $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ una muestra y un nivel de recorte $\alpha \in (0,1)$. Los pasos necesarios para llevar a cabo el algoritmo k-medias recortadas son:

Paso 0

Determinar un conjunto inicial de k centroides aleatoriamente:

- Por ejemplo, usando una submuestra de k observaciones elegidas al azar.

Paso 1

Asignar cada observación al centroide más cercano.

- Siendo $\{\boldsymbol{m}_1, \boldsymbol{m}_2, \dots, \boldsymbol{m}_k\} \subset \mathbb{R}^d$ los k centroides fijados en el paso anterior, por cada observación se calcula la distancia euclídea al k centroide más cercano:

$$d_i = \min_{j=1,\dots k} \|x_i - m_j\|, \qquad i = 1, \dots n.$$

- Las $(n * \alpha)$ observaciones con mayor distancia d_i se recortan y se asignan a H_0. La condición $d_i \geq d_{([n(1-\alpha)])}$ es conocida como “radio óptimo” y determinando cuales son las observaciones que recortar.

$$H_0 = \{ i: d_i \geq d_{([n(1-\alpha)])} \}, \quad i = 1, \dots n.$$

- Cada observación se asigna al centroide más cercano basándose en la distancia euclídea y dejando a la proporción α de los más alejados fuera,

es decir los pertenecientes a H_0 .

$$H_j = \left\{ i: \left\| x_i - m_j \right\|^2 = \min_{1 \le j \le k} \left\| x_i - m_j \right\|^2 \quad y \quad i \notin H_0 \right\}$$

La muestra H queda particionada en los k grupos $\{ H_1, \dots H_k \}$ y el grupo con las observaciones recortadas H_0.

Paso 2

Calcula los nuevos centroides como la media de las observaciones que han sido asignadas al grupo y no han sido recortadas:

$$m_j = \frac{1}{n_j} \sum_{i \in H_j} x_i \,, \qquad j = 1, \dots k$$

El paso anterior de asignación de las observaciones y cálculo de nuevos centroides se realizan de manera iterativa para evaluar en la función objetivo, posteriormente se denominará en el desarrollo del algoritmo a estas iteraciones también llamados “Ksteps” o pasos de “concentración”

$$\frac{1}{[n(1-\alpha)]} \sum_{i \notin H_0} \left(\begin{matrix} inf \\ 1 \le j \le k \end{matrix} \quad \left\| x_i - m_j \right\| \right)^2$$

o análogamente

$$\frac{1}{[n(1-\alpha)]} \sum_{j=1}^{k} \sum_{x_j \in H_k} \left\| x_i - m_j \right\|^2$$

Es necesario inicializar muchas veces el algoritmo desde el paso inicial para almacenar soluciones y utilizar aquella que minimice la función objetivo, estos reinicios se denominaran posteriormente en el desarrollo como “Niter”.

El procedimiento de k-medias recortadas es muy similar al k-medias del capítulo anterior, pero en este desarrollo nótese que las observaciones en el grupo H_0, es decir las observaciones recortadas no se consideran en el cálculo de la función objetivo.

5- Comparativa k-medias vs k-medias recortadas

En apartados previos se analizó la robustez estadística, en este nuevo apartado se demostrará como bajo ciertas condiciones el procedimiento de k-medias recortado es un procedimiento más robusto que el clásico procedimiento de k-medias.

Compararemos una muestra simulada de 208 observaciones siguiendo las distribuciones normales bivariantes:

100 observaciones	100 observaciones	8 observaciones
$N_2\left(\begin{pmatrix}15\\10\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}20\\15\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}0\\0\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 0\end{pmatrix}\right)$

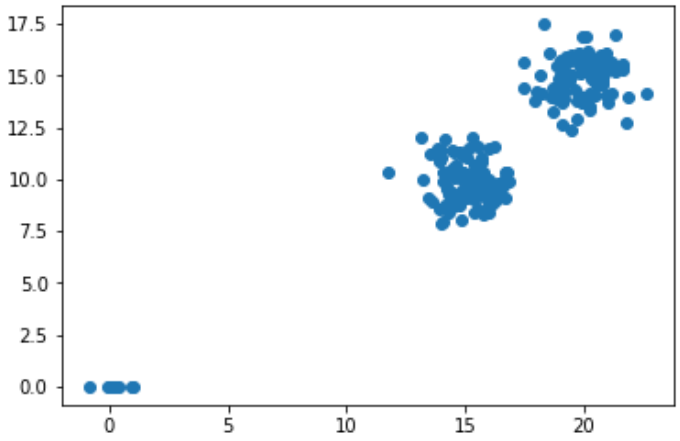


Figura 7: Muestra simulada de 208 observaciones.

Tras aplicar el algoritmo k-medias clásico sobre la muestra simulada de 208 observaciones, fijando el número de clúster en k=2, el resultado final difiere mucho del resultado que se pudiera esperar sobre las asignaciones de las observaciones.

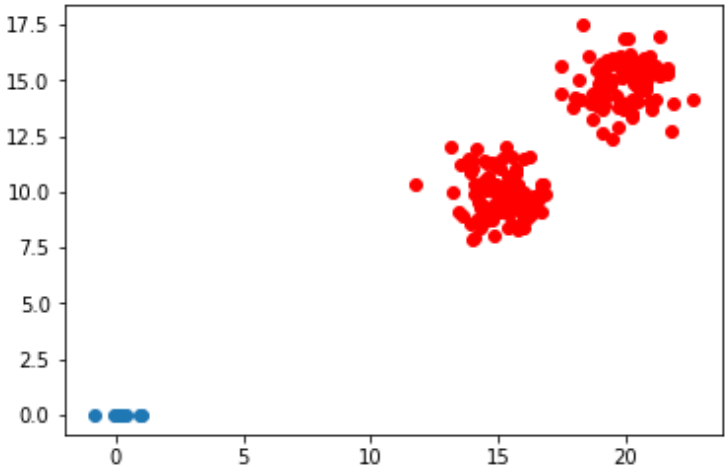


Figura 8: Muestra simulada de 208 observaciones aplicado el procedimiento k-medias.

Como se observa las 8 observaciones son lo suficientemente atípicas para que únicamente estas 8 observaciones conformen un único clúster, quedando el resto de las observaciones en único clúster. Este resultado que puede parecer inesperado a priori es debido a la falta de robustez ya comentada en capítulos previos.

El resultado de aplicar el método k-medias recortadas sobre la misma muestra simulada de 208 observaciones fijando el número de clúster en k=2 y el nivel de recorte en α= 0.05, si produce un resultado similar a las ideas que existieran preconcebidas a priori.

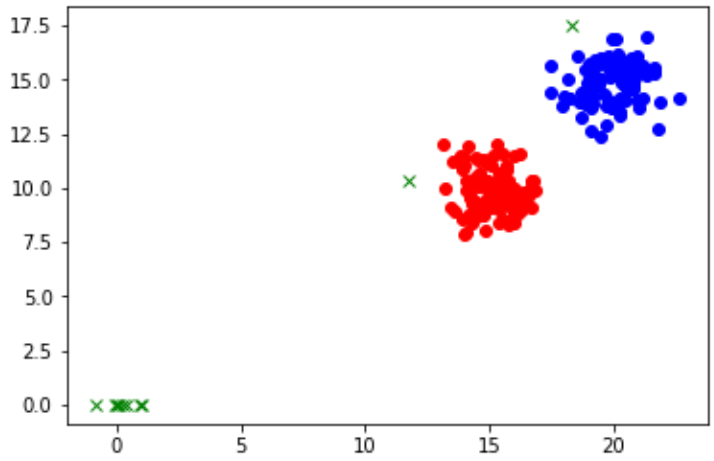


Figura 9: Muestra simulada de 208 observaciones aplicado el procedimiento k-medias recortado.

Este resultado es debido a que el método de k-medias recortado es un procedimiento de mayor robustez que el k-medias clásico. Este procedimiento de k-medias recortado es por tanto el que se aplicara en los desarrollos de los próximos capítulos, debido a que ofrece mayores garantías de resistencia a errores de transcripción, outliers… todo ello debido a su propiedad de robustez frente al procedimiento k-medias clásico.

6- Implementación en Python

A lo largo de este capítulo se hará un breve repaso sobre las características del lenguaje de programación Python, posteriormente se empleará dicho lenguaje de programación como herramienta para desarrollar el algoritmo k-medias.

6.1 Características del Lenguaje

Python es un lenguaje de programación de alto nivel diseñado por Guido van Rossum en 1990 en el Centrum Wiskunde & Informatica de Amsterdam, es usado mayoritariamente tanto en investigación como en el ámbito privado debido a su simplicidad y su filosofía de código abierto.

6.2 Módulos de código abierto

Vamos a instalar y posteriormente importar diferentes librerías de código abierto que permiten potenciar y añadir nuevas funcionalidades a Python:

Math - Librería perteneciente a la biblioteca estándar de Python que proporciona acceso a funciones matemáticas.

```
import math as math
```

Numpy - Librería de Python que proporciona acceso a funciones matemáticas sofisticadas, así como trabajar con objetos N-dimensionales.

```
install numpy
import numpy as np
```

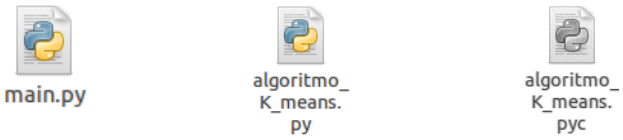
Mathplotlib - Librería de representación gráfica en 2D para Python. En este caso únicamente se importará el módulo pyplot que proporciona un framework similar a MATLAB.

```
install matplotlib
import matplotlib.pyplot as plt
```

6.3 Desarrollo del Algoritmo

Como se ha comentado con anterioridad, Python permite varios paradigmas de la programación incluyendo la programación orientada a objetos, la cual se utilizará a lo largo de este capítulo para el desarrollo del algoritmo k-medias.

El diseño modular del programa que ejecutará el algoritmo k-medias será el siguiente:



- Main.py – Archivo que contiene la función principal del programa y realizará la ejecución de dicho programa basándose en funciones pertenecientes al módulo algoritmo_K_medias.py y los parámetros de entrada definidos.

- Algortimo_K_medias.py – Archivo que contiene las funciones y tareas necesarias para el funcionamiento del algoritmo k-medias, contiene la función principal del algoritmo la cual recibe los parámetros definidos en el Main.py.

- Algoritmo_K_medias.pyc – Archivo compilado que se genera sobre los módulos importados tras la primera ejecución, mejorando la performance del script final.

Con esta estructura planteada se importara el módulo Algoritmo_K_Medias.py sobre el módulo principal Main.py, siendo el primero el que alberga las funciones de cálculo del algoritmo k-medias, para el desarrollo del algoritmo se importará en dicho módulo las librerías openSource citadas en el capítulo anterior; math, numpy y matplotlib.pyplot.

Dentro del módulo Algoritmo_K_Medias se declara la función principal, que obtiene los centroides y realiza las asignaciones de las observaciones a los correspondientes clústers más probables. Para declarar funciones en Python se utiliza la palabra reservada “def” continuado del nombre de la propia función, en este caso trimm, seguido de los parámetros necesarios para el funcionamiento de dicha función y añadiendo dos puntos (“:”) al final.

```
    def trimm(X,K,alpha,**kwargs):
```

Los argumentos definidos como necesarios para la ejecución de la función son tres argumentos convencionales y tres argumentos opcionales, únicamente los opcionales están sujetos a un nombre especifico.

Los argumentos convencionales son:

- X- matriz multidimensional que contiene p + 1 columnas, p que son las dimensiones de la propia observación y una columna donde se guardara la asignación de la observación a un clúster.
- K- valor que determina el número de clústers o grupos.
- Alpha- valor que determina el nivel de recorte aplicado.

Los argumentos opcionales son:

- Niter- número de iteraciones aleatorias, en caso de no introducir ningún valor, por defecto se define el valor 200.

- Ksteps- número de pasos K-mean, en caso de no introducir ningún valor, por defecto se define el valor 10.

- Vopt- valor de inicio suficiente para obtener la función objetivo, en caso de no introducir ningún valor por defecto se define el valor 10^30 (simplemente se considera un valor suficientemente grande).

Se genera la siguiente muestra simulada de 220 observaciones, siguiendo las distribuciones normales bivariantes, para las próximas ejecuciones:

100 observaciones	100 observaciones	20 observaciones
$N_2\left(\begin{pmatrix}0\\5\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}5\\0\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}2.5\\2.5\end{pmatrix},\begin{pmatrix}50 & 0\\0 & 50\end{pmatrix}\right)$

En lo referente al algoritmo el primer paso es la elección de k centroides, siendo k el número de clústers definido previamente, para fijar los primeros centroides se seleccionar 2 observaciones aleatoriamente de la muestra.

```
# Elección aleatoria de los k centroides iniciales
      cini=X[np.random.choice(range(n), size=K, replace=False, p=None)]
```

Con los centros fijados se calcula la distancia de cada observación al k centroide más cercano.

```
# Distancias de cada punto al centroide más cercano
          for h in range(n):
              for k in range(K):
                  ll[k]=sum((X[h,]-cini[k,])**2)
              dist[h]=min(ll)
              #print(dist)
              ind[h]=np.argmin(ll)
```

Las observaciones con mayor distancia son recortadas y se guardan en xmod.

```
# Datos modificados (Xmod) con los puntos no recortados y la última columna con a las asignaciones
  qq=[i for i, x in enumerate([nn <= sorted(dist)[notrim-1] for nn in dist]) if x]
  xmod= np.hstack([X[qq], np.array(ind).reshape(len(ind),1)[qq]])
```

Con las observaciones asignadas y no recortadas se calculas los nuevos k centroides.

```
#Cálculo de los nuevos k centros
          for k in range(K):
              ni=sum(xmod[:,p]==k)
              if ni>1:
                cini[k,]=(xmod[xmod[:,p]==0,0:p-1].mean(axis=0))
```

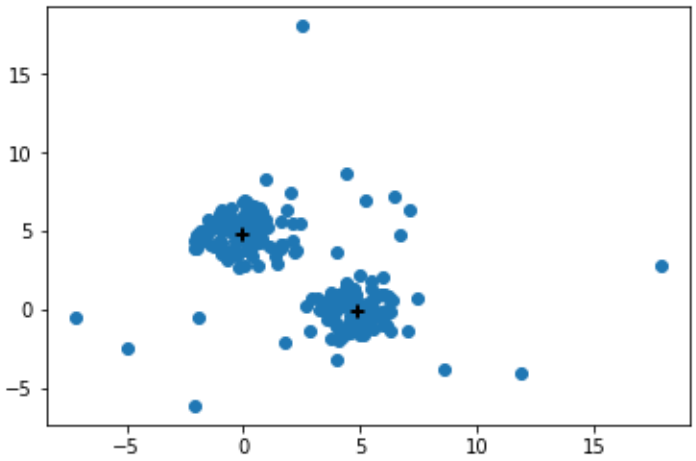


Figura 10: Dos centroides calculados aplicando el procedimiento k-medias recortadas.

Todo el procedimiento explicado anteriormente estará contenido en un iterador que se aplicará Kstep veces, almacenando los resultados para luego evaluarlo en la función objetivo

```
#Cálculo de la k-varianza recortada
        obj=0
        for l in range(notrim):

            obj=obj+sum((xmod[l,0:p]-cini[int(xmod[l,p]),:])**2)

        obj=obj/notrim

        # Cambiar el valor y los centros óptimos (copt)

        if obj<vopt:
            vopt=obj
            # Se imprimen mejoras en las funciones objetivas

            copt=cini
```

Como se desarrolló en el apartado teórico del algoritmo es necesario inicializar muchas veces el algoritmo anterior para obtener un valor mínimo de la función objetivo, el número de reinicios viene determinado por Niter, cuanto mayor sea el número de reinicios mayor precisión tendrá el algoritmo.

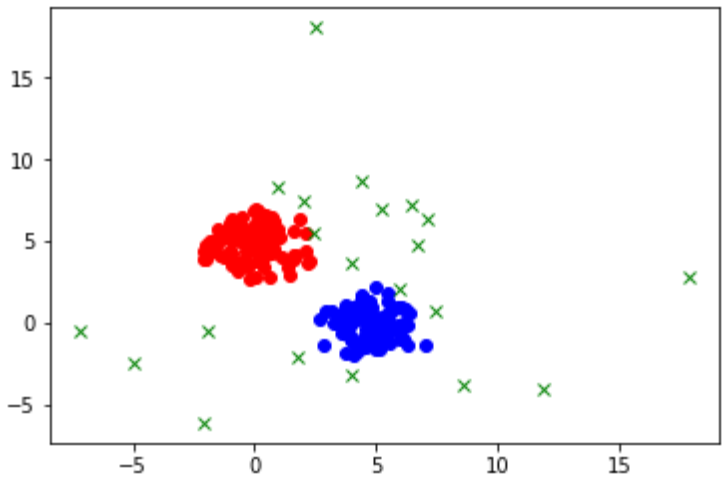


Figura 11: Asignación de las observaciones tras aplicar el procedimiento k-medias recortado.

Finalmente, fijadas las posiciones ultimas de los k centroides recortados, se obtiene el radio óptimo de los clústers y se asignan nuevamente las observaciones a sus correspondientes clústers, no asignando aquellas observaciones que no estén localizadas dentro del radio óptimo de los clústers.

La función definida en Python completa del algoritmo k-medias recortadas, que asigna cada observación a los k clústers es la siguiente:

```
def trimm(X,K,alpha,**kwargs):

    n=len(X)

    p=X.ndim

    notrim=math.floor(n*(1-alpha))

    ll=list(range(K))
    ind=list(range(n))
    dist=list(range(n))


    # Número de reinicios aleatorios

    niter=kwargs.get('niter',1000)

    # Número de pasos k-medias

    Ksteps=kwargs.get('Ksteps',10)

    # Inicia la función objetivo en un valor suficientemente grande
    vopt=kwargs.get('vopt',10**30)

    #Reinicios aleatorios
    for i in range(niter):
        # Elección aleatoria de los k centroides iniciales
        cini=X[np.random.choice(range(n), size=K, replace=False, p=None)]
        dim_cini=[K,p]

        #C-steps
        for t in range(Ksteps):

            # Distancias de cada punto al centroide más cercano
            for h in range(n):
                for k in range(K):
                    ll[k]=sum((X[h,]-cini[k,])**2)
                dist[h]=min(ll)
                #print(dist)
                ind[h]=np.argmin(ll)

            # Datos modificados (Xmod) con los puntos no recortados y la última columna igual a las
asignaciones de grupos

            qq=[i for i, x in enumerate([nn <= sorted(dist)[notrim-1] for nn in dist]) if x]

            xmod= np.hstack([X[qq], np.array(ind).reshape(len(ind),1)[qq]])
            #print(xmod)
            #print(np.size(xmod,0))
            #Cálculo de los nuevos k centros

            for k in range(K):
                ni=sum(xmod[:,p]==k)

                if ni>1:
                  cini[k,]=(xmod[xmod[:,p]==0,0:p-1].mean(axis=0))

             #print(cini)
         #Cálculo de la k-varianza recortada
        obj=0
        for l in range(notrim):
```

```
            obj=obj+sum((xmod[l,0:p]-cini[int(xmod[l,p]),:])**2)

        obj=obj/notrim

        # Cambiar el valor y los centros óptimos (copt)

        if obj<vopt:
            vopt=obj
            # Se imprimen mejoras en las funciones objetivas

            copt=cini

    # Obtener las asignaciones finales

    asig=ind

    for h in range(n):
        for k in range(K):
            ll[k]=math.sqrt(sum((X[h,]-copt[k,])**2))
        dist[h]=min(ll)
        ind[h]=np.argmin(ll)

    ord=sorted(dist)

    # Calcula el radio optimo

    ropt=ord[notrim]

    # Asigna cada observación a su cluster y -1 para las observaciones recortadas

    for h in range(n):
        if dist[h] > ropt:
            asig[h]= -1
        else:
            asig[h]=ind[h]



    # La function devuelve las observaciones asignadas a cada cluster
    asig=np.asarray(asig)
    asig=asig.reshape(len(asig),1)

    Xnew= np.hstack((X,asig))
    return Xnew
```

6.4 Ejecución del Algoritmo

Una vez estructurado modularmente el código, como se explicó en el capítulo 6.3, se creará un entorno virtual donde ejecutar el proceso que permite ejecutar el algoritmo en un entorno aislado donde otras librerías o versiones no interfieren en la ejecución.

Desde la propia consola de Windows se realiza la instalación de la herramienta capaz de crear entornos virtuales en Python.

- Instalar la herramienta de creación de entornos virtuales

```
pip install virtualenv
```

- Creación del entorno virtual “temp” en la ubicación de los ejecutables.

```
virtualenv temp
```

- Activar el entorno virtual con la ubicación de la ruta del archivo “active.bat”

```
“C.\Users\...\temp\Scripts\activate.bat”
```

- Instalar los módulos nombrados anteriormente desde el pip dentro del entorno virtual

Con estos pasos el entorno virtual ya está preparado para ejecutar archivo main.py desde el propio terminal de MS-DOS.

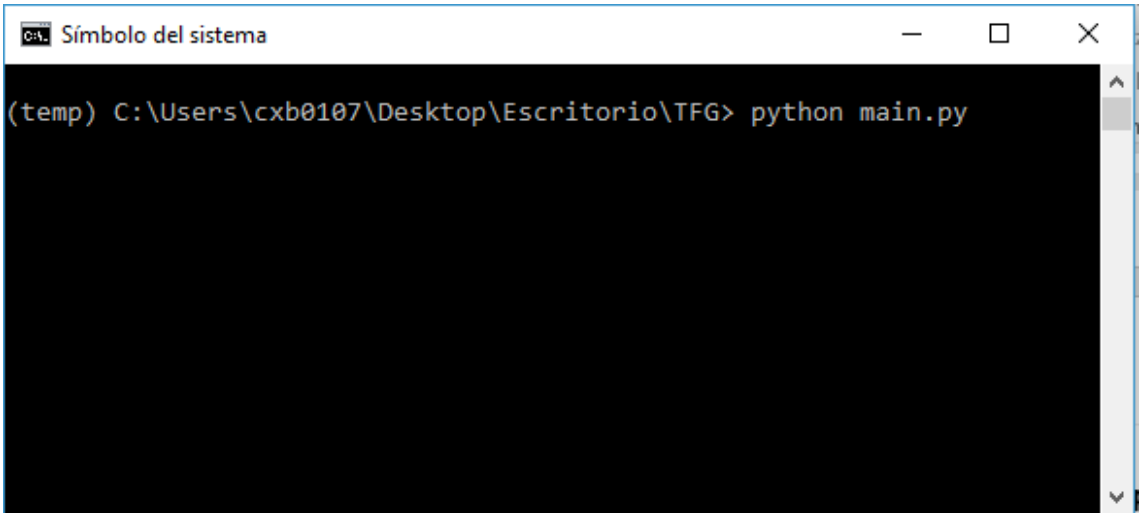


Figura 12: Ejecución del archivo main.py desde el virtualenv temp v.

Con los parámetros definidos para la ejecución en el archivo main son; K= 2, Alpha= 0.1, Niter= 200, Ksteps= 10, Vopt= 10^30 y sobre una muestra de 220 individuos con las siguientes características:

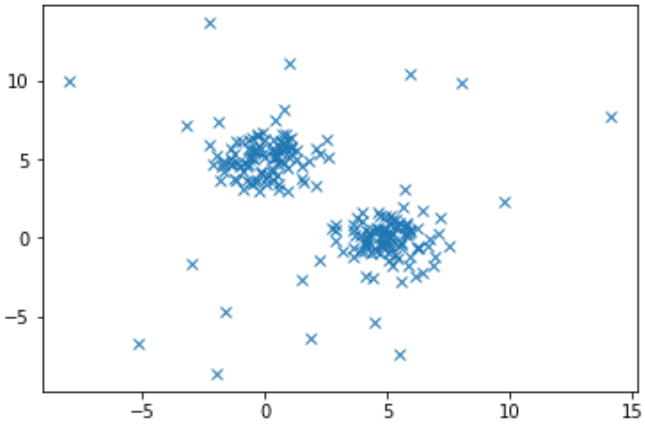


Figura 13: Datos de muestra con 220 observaciones en Python.

El resultado final de la asignación de las observaciones tras la ejecución del archivo main sobre esta muestra de datos de 220 individuos se distribuyen gráficamente sobre el plano:

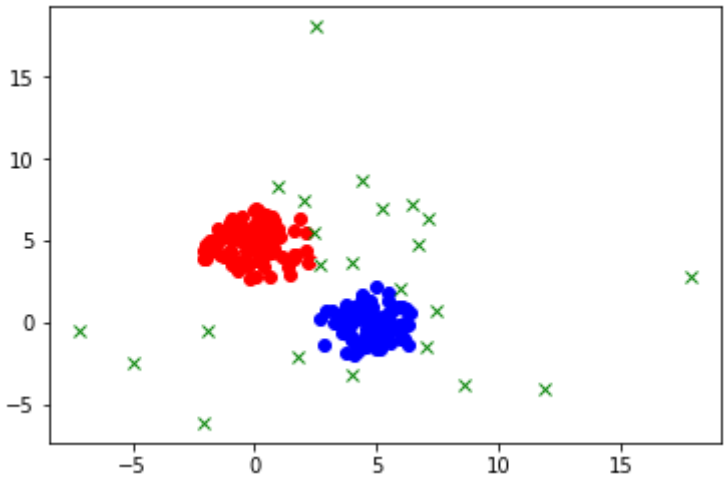


Figura 14: Asignación en 2 clústers aplicado k-medias recortadas de 220 observaciones en Python.

Las asignaciones de las 220 observaciones de las en 2 clústers son 99 individuos en el primer clúster y 90 observaciones en el segundo clúster y 22 observaciones recortadas.

7- Implementación en R

En este capítulo se nos presenta las principales características del lenguaje R, seguido de los desarrollos del algoritmo k-medias en dicho en entorno.

7.1 Características del Lenguaje

R es un lenguaje de programación de análisis estadístico desarrollado basado en el lenguaje S por Ross Ihaka y Robert Gentleman en 1993, licenciado bajo GPL y capaz de compilar y ejecutar en diversas plataformas UNIX, Windows, MacOS… permitiendo que tenga una gran repercusión y alcance.

7.2 Desarrollo del Algoritmo

Al estar basado en el lenguaje de programación S, R hereda la programación orientada a objetos, los desarrollos se realizan en scripts con extensión .r, documento con órdenes o códigos que se ejecutan o compilan de manera secuencial.

El script que denominado como “k-medias.r” que contiene únicamente una función capaz realizar las asignaciones de las observaciones al correspondiente clúster más probable a partir de otras funciones definidas en la librería MASS citada anteriormente.

Las funciones en R también son objetos y por tanto también es necesario darles un nombre, en este caso se denominará “trimm”, para definir funciones en R es necesario invocar a function() con los argumentos o valores de entrada entre paréntesis “()”, seguido entre llaves “{ }”, del cuerpo y el resultado, siendo el cuerpo las operaciones que se ejecutan en base a los argumentos definidos y el resultado la última línea del código, que será el valor devuelto como resultado del total de operaciones.

```
trimm <- function(X,K,alpha){
…
}
```

Los argumentos definidos en la función son:

- X- matriz de p columnas numéricas, tantas como dimensiones caracterizan a las observaciones.
- K- valor que determina el número de clústers o grupos.
- Alpha- valor numérico que determina el nivel de recorte aplicado.

En lo referente al cuerpo de la función, se definen los parámetros de manera fija aquellos que fueron definidos como opcionales en el desarrollo del algoritmo en Python, los valores usados como predeterminados en dicho algoritmo es el valor fijado en este caso.

Los valores fijos en el cuerpo de la función son:

- Niter- número de iteraciones aleatorias, con valor fijado en 200.
- Ksteps- número de pasos K-mean, definido como 10.
- Vopt- valor de inicial para obtener la función objetivo, fijado al inicio en 10^30.

En lo referente al desarrollo de las operaciones necesarias para obtener el resultado del algoritmo k-medias el primer paso es de nuevo al igual que ocurría en el algoritmo desarrollado en Python el cálculo de los centroides. Para obtener dichos k centros, siendo k el número de clústers definido previamente como un argumento, se realizarán 200 iteraciones correspondientes al número definido en la variable “Niter” y seleccionando aleatoriamente los centros iniciales. A lo largo de estas iteraciones se calculan las distancias de cada observación a su centro más cercano, para finalmente empleando las observaciones asignadas a cada clúster obtener el centroide sin usar el procedimiento de recorte en las observaciones al igual que en el procedimiento usado en el algoritmo en Python.

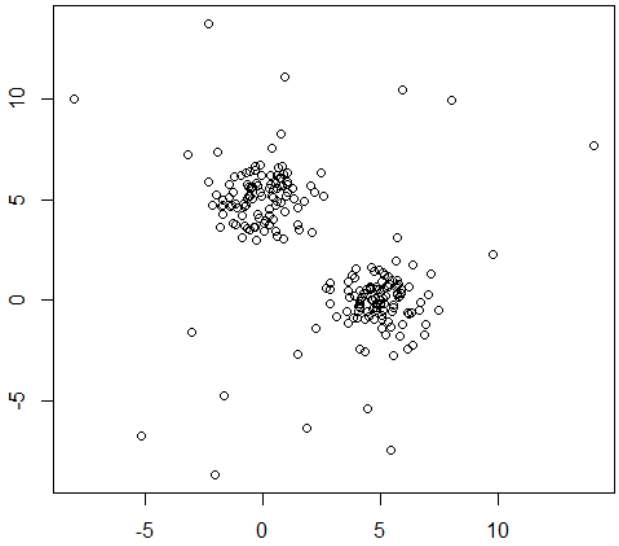


Figura 15: Datos de muestra con 220 observaciones en R.

La función en R definida que asigna cada observación es la siguiente:

```
trimm <- function(X,K,alpha){
        tiempo <- proc.time()
        n <- dim(X)[1]
        p <- dim(X)[2]
        no.trim <- floor(n*(1-alpha))
        ll <- (1:K)
        ind <- (1:n)
        dist <- ind

        # Número de reinicios aleatorios
        niter <- 200
```

```
        # Número de pasos k-medias
        Ksteps <- 10

        #Inicia la función objetivo en un valor suficientemente grande
        vopt <- 10^30

        #Reinicios aleatorios
        for (iter in 1:niter){

                # Elección aleatoria de K-centroides iniciales
                cini <- X[sample(1:n,size=K,replace=F),]
                dim(cini) <- c(K,p)

         #C-steps
                for (t in 1:Ksteps){

                        # Distancia de cada observación al centroide mas cercano
                        for (h in 1:n){
                                for (k in 1:K){
                                        ll[k] <- sum((X[h,]-cini[k,])^2)
                                        }
                                dist[h] <- min(ll)
                                ind[h] <- which.min(ll)
                                }

                        # Datos modificados (Xmod) con los puntos no recortados y la última
                        # columna igual a las asignaciones de grupos
                        qq <- (1:n)[dist<=sort(dist)[no.trim]]
                        xmod <- as.matrix(cbind(X[qq,],ind[qq]))

                        #Cálculo de los nuevos k centros
                        for (k in 1:K){
                                ni <- sum(xmod[,p+1]==k)
                                if (ni>1){
                                        cini[k,]<-apply(xmod[xmod[,p+1]==k,1:p],2,mean)
                                        }
                        }

                }

        # Cálculo de la k-varianza recortada
        obj <- 0
        for (l in 1:no.trim){
                obj <- obj+sum((xmod[l,1:p]-cini[xmod[l,p+1],])^2 )
        }
        obj <- obj/no.trim

        # Cambiar el valor y los centros óptimos (copt)
        if (obj <vopt){
                vopt <- obj
                # Se imprimen mejoras en las funciones objetivas
                #print(vopt)
                copt <- cini
                }
        }
## Obtener las asignaciones finales

asig <- ind
for (h in 1:n){
        for (k in 1:K){
                ll[k] <- sqrt(sum((X[h,]-copt[k,])^2))
                }
        dist[h] <- min(ll)
        ind[h] <- which.min(ll)
                }
ord <-sort(dist)
```

```
# Calcula los radios
ropt <- ord[no.trim]

# Asigna cada observacion a su cluster y pone un 0 a las observaciones recortadas
for (h in 1:n){
        ifelse( dist[h]>ropt, asig[h] <-0, asig[h] <-ind[h])
        }

for (k in 1:K){
        # Group
        #print(paste("Cluster ",k,":"))
        #print(c(1:n)[asig==k])
        }
        # Trimmed observations
        #print("Trimmed observations:")
        #print(c(1:n)[asig==0])

return(list(vopt=vopt,copt=copt,ropt=ropt,asig=asig))
}
```

7.3 Ejecución del Algoritmo

Ya definida la función con el algoritmo k-medias y con los parámetros definidos para la ejecución del algoritmo en Python que son; K= 2, Alpha= 0.1, Niter= 200, Ksteps= 10, Vopt= 10^30 y sobre una muestra de 220 individuos, se realizara la misma para el algoritmo desarrollado en R.

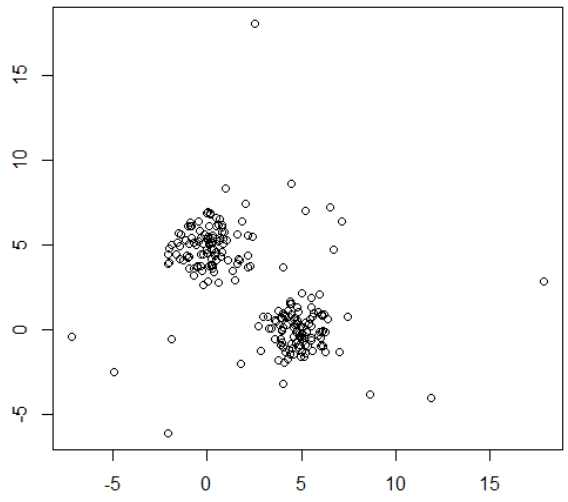


Figura 16: Datos de muestra con 220 observaciones en R.

El algoritmo se ejecuta desde el propio entorno de R, en la R-Consola con el consiguiente resultado:

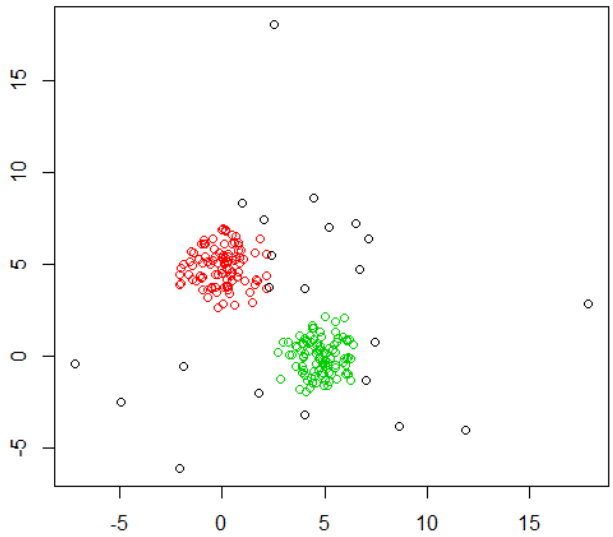


Figura 17: Asignación en 2 clústers aplicado k-medias recortadas de 220 observaciones en R.

Las asignaciones de las 220 observaciones de las en 2 clústers son 99 individuos en el primer clúster y 90 observaciones en el segundo clúster y 22 observaciones recortadas.

8- Comparativa R vs Python

En este capítulo se realiza una comparativa de los lenguajes utilizados para el desarrollo del algoritmo k-medias en capítulos anteriores.

Es difícil priorizar un lenguaje frente al otro, en este capítulo vamos a tratar las bondades de ambos, la última versión de Python disponible es la 3.7.3 del 25 marzo de 2019 frente a R que se encuentra en su versión 3.5.3 del 11 de marzo de 2019.

R-project

R se centra en el análisis de datos, estadísticas y visualización de datos, es mas utilizado en el ámbito académico y la investigación, permite crear complejos modelos analíticos en cuestión de minutos debido a sus características.

Python

Python es un lenguaje de propósito general centrado en la productividad y legibilidad del código, es utilizado por desarrolladores que desean profundizar en la ciencia de datos debido a si sintaxis fácil de entender y su integración con otras aplicaciones es sencilla.

A grandes rasgos se puede apreciar que la fortaleza de R es su capacidad para realizar análisis y modelado de datos de forma muy cómoda y específica, debido a ser una herramienta desarrollada con el propósito de investigación en la Ciencia de los Datos.

Por el contrario, Python es un lenguaje de propósito general y por tanto permite realizar grandes proyectos, dado que permite integrar otras herramientas de propósito general como por ejemplo Django, Flask, Tensorflow, Spark, etc. y que no aplican exclusivamente a la Ciencia de los Datos.

Realizando una comparativa los desarrollos en R y Python, mencionados anteriormente, y utilizando las mismas condiciones y la misma muestra de 220 observaciones. Tal y como se espera para algoritmos de este tipo con una cantidad de iteraciones suficientes, el resultado es el mismo considerando las mismas inicializaciones.

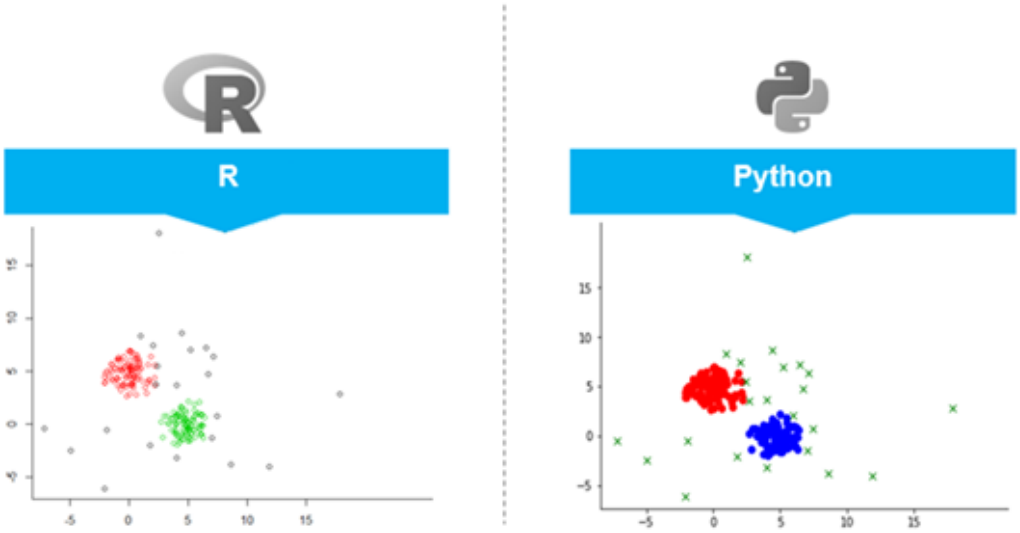


Figura 18: Resultado de las asignaciones de los algoritmos en R y Python en 220 observaciones.

Se realizará una prueba de los algoritmos sobre una muestra simulada de 1000 observaciones con un nivel de α = 0,01, siguiendo las distribuciones normales bivariantes, para las próximas ejecuciones:

990 observaciones	990 observaciones	20 observaciones
$N_2\left(\begin{pmatrix}20\\15\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}30\\25\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}20\\20\end{pmatrix},\begin{pmatrix}50 & 0\\0 & 50\end{pmatrix}\right)$

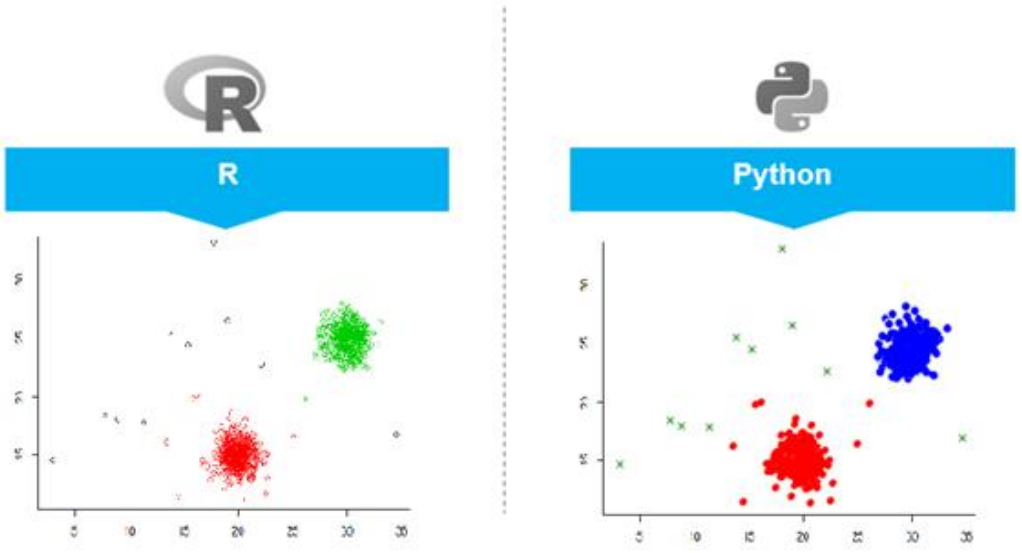


Figura 19: Resultado de las asignaciones de los algoritmos en R y Python en 1000 observaciones.

Por último, se realizará nuevamente una ejecución con una muestra de 200200 observaciones con un nivel de recorte $\alpha = 0.05$.

200000 observaciones	200000 observaciones	200 observaciones
$N_2\left(\begin{pmatrix}20\\15\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}30\\25\end{pmatrix},\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$	$N_2\left(\begin{pmatrix}25\\20\end{pmatrix},\begin{pmatrix}50 & 0\\0 & 50\end{pmatrix}\right)$

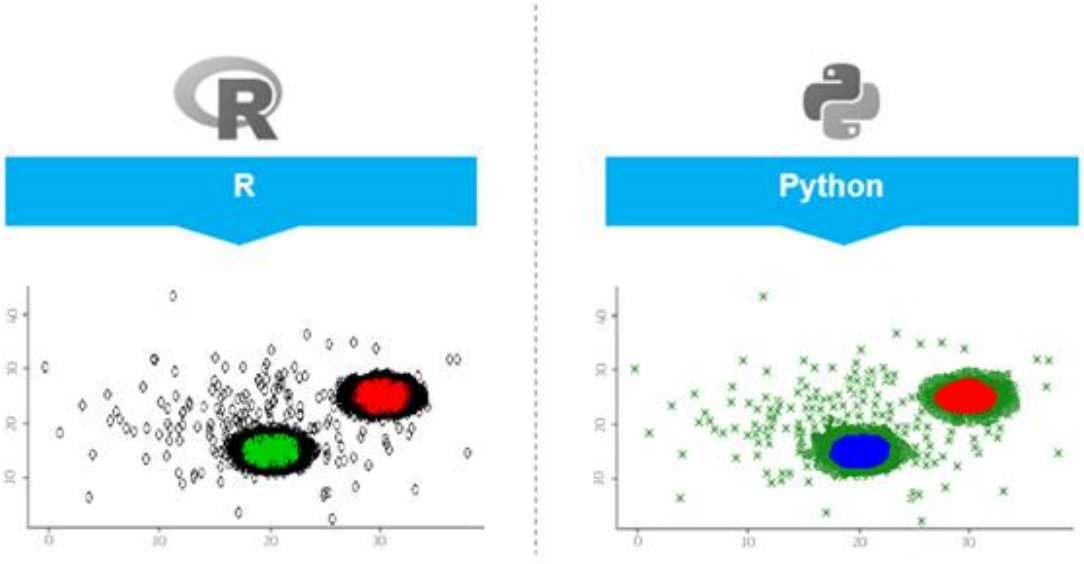


Figura 20: Resultado de las asignaciones de los algoritmos en R y Python en 200200 observaciones.

Tras realizar 5 ejecuciones sobre cada muestra aplicando el algoritmo de k-medias recortadas en R y en Python los resultados de tiempo promedios son los siguientes:

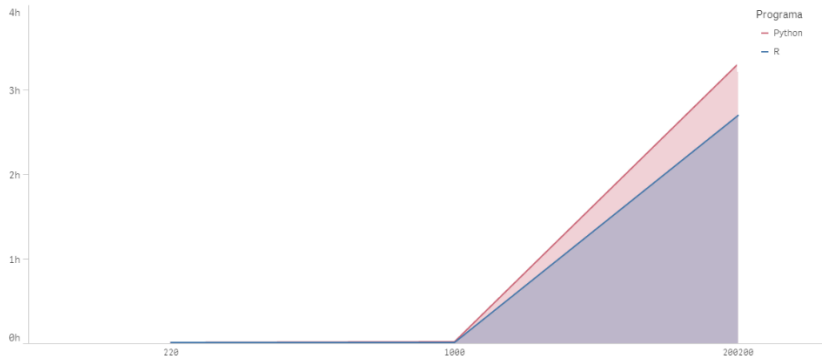


Figura 21: Grafico que representa el tiempo de ejecución en R y Python de las muestras.

	220 obs.	1000 obs.	200200 obs.
R	4.36 seg.	32 seg	2h y 52 min.
Python	8.14 seg.	45 seg	3h y 12 min

Los resultados obtenidos por el algoritmo de k-medias recortadas en R y Python son similares a la hora de aplicar los recortes y las asignaciones, pero como si es significativamente apreciable, el tiempo de ejecución en Python es superior en crudo al de R para muestras de mayor tamaño.

Por otra parte, es destacable que no se han explorado en este TFG otras líneas de investigación para optimizar las ejecuciones, como la posibilidad de la paralelización y la concurrencia, que ofrece grandes ventajas en grandes conjuntos de datos, donde Python dispone de gran cantidad de librerías.

9- Aplicación web de k-medias recortadas

Las aplicaciones web y aplicaciones cloud han tomado una mayor relevancia en los últimos tiempos debido al nacimiento de los smartphones, así como las plataformas de cloud como AWS, AZURE, GOOGLE CLOUD… es por lo que las aplicaciones desktop tradicionales han comenzado a migrar sus plataformas al cloud.

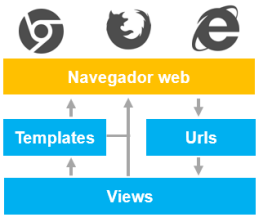
Basado en la exposición del capítulo anterior en el cual se presenta a Python como un lenguaje de propósito general y compatible con gran cantidad de herramientas que no aplican exclusivamente a la Ciencia de Datos, será el principal motor de la aplicación.

9.1 Herramientas aplicadas

Para el desarrollo de la aplicación web es necesario el uso de algunas tecnologías web.

Django

Django es un framework de desarrollo web de alto nivel y código abierto escrito en Python. Su función principal es proporcionar componentes que permiten un desarrollo rápido y limpio de entornos web.



Django está basado en el modelo ***Models-Views-Templates***.

En el desarrollo de este proyecto no se utiliza el esquema tradicional. El esquema de la aplicación web no utiliza los *Models*, dado que no es necesario almacenar cálculos y resultados para futuras ejecuciones.

Los resultados se almacenarán en un .csv exportable.

JavaScript

JavaScript es un lenguaje de programación que se utiliza de manera habitual en el desarrollo de páginas y aplicaciones web, esto se debe a que los navegadores web modernos son capaces de interpretar el código JavaScript, y se ejecutan en el cliente permitiendo realizar efectos y funcionalidades web.

Visualización D3

D3 es una librería de JavaScript que se comenzó a distribuir en 2011 con licencia BSD de software libre, permite visualizar gráficos dinámicos e interactivos utilizando estándares web. La librería D3 combina potentes técnicas de visualización haciendo uso de distintas tecnologías SVG, Canvas, HTML y CSS soportado por los distintos navegadores web.

HTML

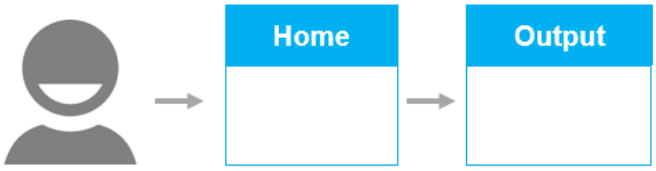
HTML (HyperText Markup Language) es un lenguaje de marcado utilizado en el desarrollo de páginas web, permite ordenar el contenido de una web mediante la utilización de etiquetas.

CSS

Las plantillas CSS o hojas de estilo en cascada permite establecer el diseño visual, estilos fuentes, espaciado, colores, etc… que se aplican sobre un lenguaje de marcado.

9.2 Desarrollo de la aplicación

La aplicación web cuyo desarrollo se expone a lo largo de este capítulo dispone de dos páginas *"urls.py",* la primera "*url"* home en la cual el usuario cargará sus datos para posteriormente en una segunda "*url"* output visualizará el resultado de aplicar el algoritmo k-medias recortados sobre sus datos, también podrá descargar los datos y la asignación correspondiente al clúster.



Django es la principal herramienta en la cual se construye la aplicación web, por ello el primer paso es iniciar un proyecto en Django con el nombre de la aplicación ***Cluster_Trimm***

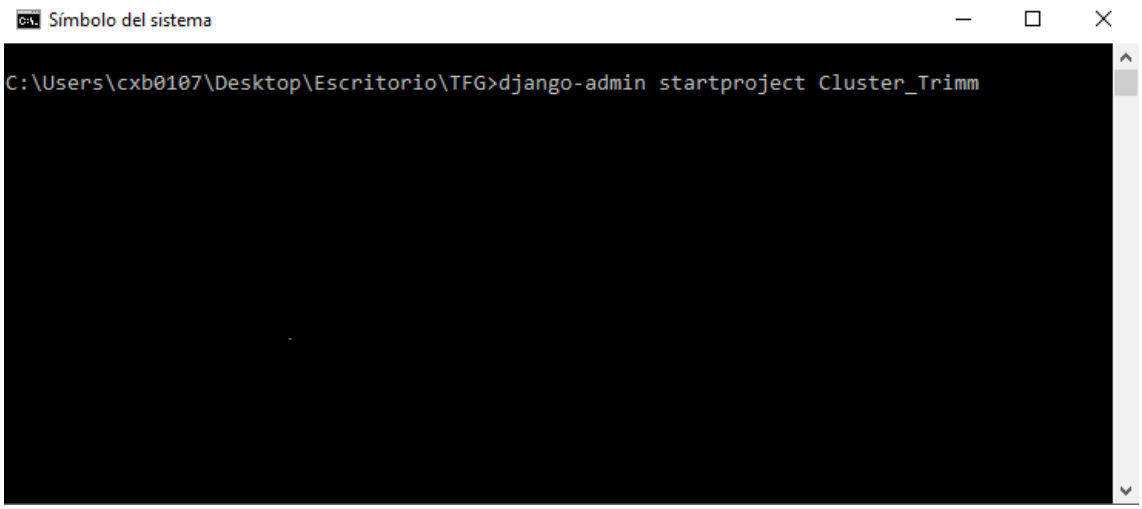


Figura 19: Inicio del proyecto Cluster_Trimm en Django.

Una vez iniciado el proyecto en django sea configura el archivo de settings.py para que el funcionamiento entre vistas, *urls*, plantillas y los elementos estáticos sea el

adecuado. Para ello se declara la carpeta *templates* como el directorio de las plantillas html y la carpeta *static* como el directorio de csv, imágenes y javascripts.

```
# Añadimos configuracion (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.1/howto/static-files/

STATIC_URL = '/static/'

# Añadimos el directorio a archivos estaticos
STATICFILES_DIRS = (
    os.path.join(BASE_DIR, 'static'),
)

STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')
```

Figura 20: Configuración del directorio static en Django.

Una vez configurado el setting.py del proyecto se prosigue configurando el views.py, en él se declaran 2 funciones:

- button: esta función devolverá la primera plantilla home.html.

```
def button(request):
    return render(request,'home.html')
```

- output: esta función realiza la lectura de los datos cargados en formato csv por un usuario en la primera plantilla home, en la propia función se integra el algoritmo de k-medias recortado implementado en Python, la propia función almacena el resultado en un csv y devuelve la segunda plantilla html output.html, así como el número de observaciones asignadas a cada clúster y los outliers. Los parámetros k y α se definirán en la propia la aplicación web.

```
def output(request):
    csv_file = request.FILES['csv_file']
    X=genfromtxt(csv_file, delimiter=',')
    try:                        K = request.POST['k-cluster']
    except MultiValueDictKeyError:      K = 2
    try:                        alpha = request.POST['alpha']
    except MultiValueDictKeyError:      alpha = 0.1
    K=int(float(K))
    alpha=float(alpha)
    n=len(X)
    p=X.ndim
    notrim=math.floor(n*(1-alpha))
    ll=list(range(K))
    ind=list(range(n))
    dist=list(range(n))
    niter=10
    Ksteps=200
    vopt=10**30
```

```
    for i in range(niter):

        cini=X[np.random.choice(range(n), size=K, replace=False, p=None)]

        for t in range(Ksteps):

            for h in range(n):
                for k in range(K):
                    ll[k]=sum((X[h,]-cini[k,])**2)
                dist[h]=min(ll)
                ind[h]=np.argmin(ll)


            qq=[i for i, x in enumerate([nn <= sorted(dist)[notrim-1] for nn in
dist]) if x]

            xmod= np.hstack([X[qq], np.array(ind).reshape(len(ind),1)[qq]])

        obj=0
        for l in range(notrim):

            obj=obj+sum((xmod[l,0:p]-cini[int(xmod[l,p]),:])**2)

        obj=obj/notrim

        if obj<vopt:
            vopt=obj
            copt=cini

    asig=ind

    for h in range(n):
        for k in range(K):
            ll[k]=math.sqrt(sum((X[h,]-copt[k,])**2))
        dist[h]=min(ll)
        ind[h]=np.argmin(ll)

    ord=sorted(dist)
    ropt=ord[notrim]

    for h in range(n):
        if dist[h] > ropt:
            asig[h]= -1
        else:
            asig[h]=ind[h]
    asig=np.asarray(asig)
    asig=asig.reshape(len(asig),1)

    Xnew= np.hstack((X,asig))

    num_k1=(len(X[Xnew[:,2]==0]))
    num_k2=(len(X[Xnew[:,2]==1]))
    outliers=(len(X[Xnew[:,2]==-1]))
```

```
np.savetxt("datos_exportados.csv", Xnew, delimiter=",",fmt='%f')

    return
render(request,'output_tfg.html',{'num_k1':num_k1,'num_k2':num_k2,'outliers':outliers})
```

El siguiente paso es configurar el archivo url.py en cual se define la raíz, donde se situará la vista *button* que se ejecuta al iniciar el servidor. También define la url output donde se sitúa la propia vista *output* que se ejecuta al accionar una etiqueta html con nombre *"script".*

```
from django.conf.urls import url
from django.contrib import admin

from . import views

urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^$', views.button),
    url(r'^output', views.output, name="script"),
]
```

Figura 21: Configuración del directorio url.py en Django.

Tras la configuración de los elementos principales del proyecto en Django, se utiliza una visualización D3 en la cual se representará mediante un diagrama de dispersión los resultados del algoritmo de k-medias recortadas. La visualización se basa en los desarrollos de Jonas Petersson, Mike Bostock, Michele Weigle y Richard Westenra.

La visualización del diagrama de dispersión D3 requiere de los siguientes archivos Javascript para su funcionamiento, que se almacenaran entre los archivos estáticos.

d3.v3min.js – Librería JavaScript para de manipulación de documentos basados en datos, en este caso corresponde a la tercera versión mínima del paquete d3.

d3.tip.v.0.6.3 – Librería JavaScript que permite añadir tooltips a las visualizaciones D3, desarrollada por Justin Palmer desde 2013.

scatter.js – es un archivo de texto plano que contiene el script de Javascript que permite realizar la representación visual, leerá las asignaciones desde un archivo csv, para ello utiliza funciones de las librerías d3.v3min y d3.tip anteriores.

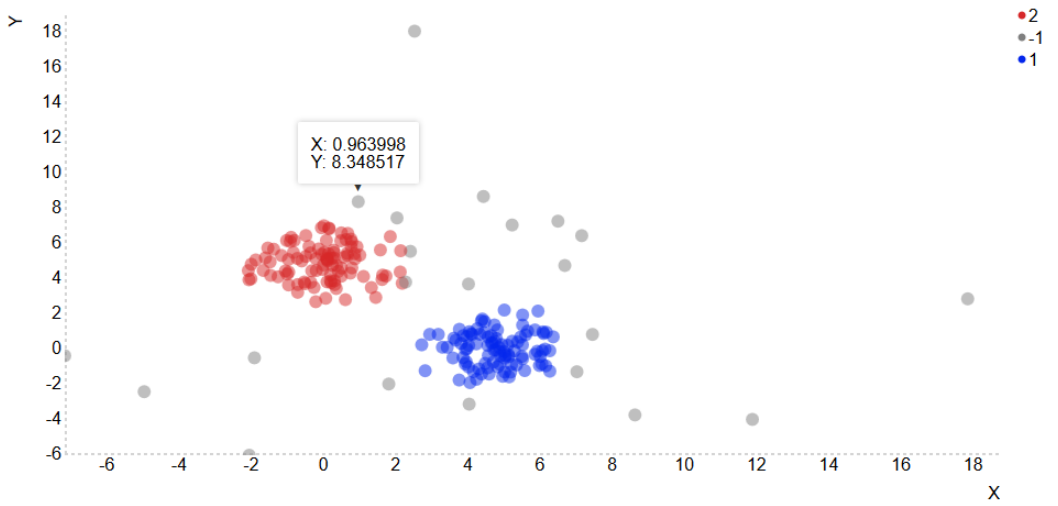


Figura 22: Visualización interactiva en D3 de 2 clústers aplicado k-medias recortado.

Por último, se especifican las plantillas, así como los archivos estáticos necesarios para la aplicación web.

Home.html

La primera plantilla y únicamente dispondrá de la funcionalidad que permite al usuario cargar un csv en la aplicación web, para su posterior análisis.

```
<!DOCTYPE html>
<html>

<head>
	<title>
		UVA | K-Medias recortadas App
	</title>
	{% load static %}
	<link rel="stylesheet" type="text/css" href="{% static "css/home.css"
%}" />
	<link rel="shortcut icon" type="image/png" href="{% static
'img/favicon.ico' %}"/>
</head>

<body>
	<header>
		<div class="header-white">

			<div class="head-left"><img src="{% static "img/uva.png"
%}" height="80px">
			</div>
			<div class="title">K-Medias recortadas</div>
		</div>

	</header>
```

```
        <form method="post" action="{% url 'script' %}"
enctype="multipart/form-data">
                {% csrf_token %}
        <div class="header-upload">
                <h2>Carga tu archivo csv</h2>

                <div class="body-upload">

                <div class="content-upload">

                        <div class="content-upload-left">
                                <img src="{% static "img/upload.svg" %}"
height="100px"><br><br>
                                <input id="upload" type="file" name="csv_file">
                        </div>

                        <div class="content-upload-right">
<b>Configura los parámetros</b><br><br>
                                        k-número de clusters<br><input type="text"
name="k-cluster" value="2">
                                        <br><br>
                                        &alpha;-nivel de recorte<br><input type="text"
name="alpha" value="0.1">
                                        <br><br>
                                        <b>¿Necesitas ayuda con el formato de tu
CSV?</b><br><br>
                                        Descarga nuestro
                                                <a class="model csv" href="{% static
"data/data_prueba.csv" %}" style="color: #00aaee; font-weight: bold; text-
decoration:none;" title="modelo.csv">
                                                        modelo de csv
                                                </a>
                                        y sigue su formato para tener un resultado
satisfactorio.
                        </div>
                </div>
                        <div class="send">
                                <input class="open" type="submit" value="Cargar
csv"/>
                        </div>
                </div>
        </div>

        <div class="popup-overlay" style="background-color: rgba(0,0,0,0.5);">
                <div class="popup-content">

                        <div class="contloa">

                                        <div class="message"><img src="{% static
"img/spinner.svg" %}" height="100px"><br><b>Cargando tus
datos</b></div>

                        </div>
                </div>
```

```
        </div>
        <script src="{% static 'js/jquery.min.js' %}"></script>
        <script src="{% static 'js/popup.js' %}"></script>
</form>
</body>

</html>
```

Output.html

La plantilla final y mostrara un diagrama de dispersión interactivo desarrollado en D3, que muestra el resultado de aplicar el algoritmo de k-medias recortadas sobre el conjunto de datos cargado en la plantilla anterior.

```
<!DOCTYPE html>
<html>

<head>
        <title>
                UVA | K-Medias recortadas App
        </title>
        {% load static %}
        <link rel="stylesheet" type="text/css" href="{% static
"css/output_tfg.css" %}" />
        <link rel="stylesheet" type="text/css" href="{% static "css/scatter.css"
%}" />
        <link rel="shortcut icon" type="image/png" href="{% static
'img/favicon.ico' %}"/>

        <script src="{% static 'js/d3.v3.min.js' %}"></script>
        <script src="{% static 'js/d3.tip.v0.6.3.js' %}"></script>
        <script>

          var margin = { top: 50, right: 50, bottom: 50, left: 50 },
              outerWidth = 900,
              outerHeight = 500,
              width = outerWidth - margin.left - margin.right,
              height = outerHeight - margin.top - margin.bottom;

          var x = d3.scale.linear()
              .range([0, width]).nice();

          var y = d3.scale.linear()
              .range([height, 0]).nice();

           var xCat = "X",
              yCat = "Y",
              colorCat = "Cluster";


          d3.csv(src="{% static 'data/data_prueba.csv' %}", function(data) {
            data.forEach(function(d) {
            d.X = +d.X;
            d.Y= +d.Y;
```

```
});

var xMax = d3.max(data, function(d) { return d[xCat]; }) * 1.05,
    xMin = d3.min(data, function(d) { return d[xCat]; }),
    xMin = xMin > 0 ? 0 : xMin,
    yMax = d3.max(data, function(d) { return d[yCat]; }) * 1.05,
    yMin = d3.min(data, function(d) { return d[yCat]; }),
    yMin = yMin > 0 ? 0 : yMin;

    x.domain([xMin, xMax]);
    y.domain([yMin, yMax]);

var xAxis = d3.svg.axis()
    .scale(x)
    .orient("bottom")
    .tickSize(-height);

 var yAxis = d3.svg.axis()
     .scale(y)
     .orient("left")
     .tickSize(-width);

 var color = d3.scale.category10();
     color.range(['#980A41', '#7f7f7f', '#00aaee']);
//var color = colores_cluster();

var tip = d3.tip()
   .attr("class", "d3-tip")
   .offset([-10, 0])
   .html(function(d) {
       return xCat + ": " + d[xCat] + "<br>" + yCat + ": " + d[yCat];
     });

var zoomBeh = d3.behavior.zoom()
   .x(x)
   .y(y)
   .scaleExtent([0, 500])
   .on("zoom", zoom);

var svg = d3.select("#scatter")
  .append("svg")
  .attr("width", outerWidth)
  .attr("height", outerHeight)
  .append("g")
  .attr("transform", "translate(" + margin.left + "," + margin.top + ")")
  .call(zoomBeh);

svg.call(tip);
svg.append("rect")
  .attr("width", width)
  .attr("height", height);

svg.append("g")
```

```
            .classed("x axis", true)
            .attr("transform", "translate(0," + height + ")")
            .call(xAxis)
            .append("text")
            .classed("label", true)
            .attr("x", width)
            .attr("y", margin.bottom - 10)
            .style("text-anchor", "end")
            .text(xCat);

          svg.append("g")
            .classed("y axis", true)
            .call(yAxis)
            .append("text")
            .classed("label", true)
            .attr("transform", "rotate(0)")
            .attr("y", -margin.left/2) //.attr("y", -margin.left)
            .attr("dy", ".71em")
            .style("text-anchor", "end")
            .text(yCat);

        var objects = svg.append("svg")
          .classed("objects", true)
          .attr("width", width)
          .attr("height", height);

          objects.append("svg:line")
          .classed("axisLine hAxisLine", true)
         .attr("x1", 0)
         .attr("y1", 0)
         .attr("x2", width)
         .attr("y2", 0)
         .attr("transform", "translate(0," + height + ")");

          objects.append("svg:line")
         .classed("axisLine vAxisLine", true)
         .attr("x1", 0)
         .attr("y1", 0)
         .attr("x2", 0)
         .attr("y2", height);

        objects.selectAll(".dot")
            .data(data)
          .enter().append("circle")
            .classed("dot", true)
            .attr("r", function (d) { return 6 ; }) //.attr("r", function (d) { return 6 *
Math.sqrt(d[rCat] / Math.PI); })
            .attr("transform", transform)
            .style("fill", function(d) { return color(d[colorCat]); })
            .on("mouseover", tip.show)
            .on("mouseout", tip.hide);

        d3.select("input").on("click", change);
```

```
        function zoom() {
          svg.select(".x.axis").call(xAxis);
          svg.select(".y.axis").call(yAxis);
          svg.selectAll(".dot")
             .attr("transform", transform);
          }

        function transform(d) {
          return "translate(" + x(d[xCat]) + "," + y(d[yCat]) + ")";
        }
  });

        </script>
</head>

<body>
      <header>
            <div class="header-white">

                  <div class="head-left"><img src="{% static "img/uva.png"
%}" height="80px">
                  </div>
                  <div class="title">K-Medias recortadas</div>
            </div>

      </header>

      <div class="header-upload">
            <h2>2 Grupos contruidos con K-medias recortadas</h2>

            <div class="body-upload">
              <div class="content-scatter-left">
                <div id="scatter" style="float:left;"></div>

              </div>
              <div class="content-scatter-right">
                <b>Resumen</b>
                <br><br>
                <span class="dot" style="background-color:
#00aaee;"></span>
                  <b>Grupo 1:</b> <span style="color: #00aaee; font-
weight: bold;">{{num_k1}}</span> observaciones <br>
                <span class="dot" style="background-color:
#980A41;"></span>
                  <b>Grupo 2:</b> <span style="color: #00aaee; font-
weight: bold;">{{num_k2}}</span> observaciones <br><br>
                <span class="dot" style="background-color: #7f7f7f;"></span>
                  <b>Outliers:</b> <span style="color: #00aaee; font-
weight: bold;">{{outliers}}</span> observaciones <br>
                <a class="model csv" href="{% static "data/data_prueba.csv"
%}" style="color: #00aaee; font-weight: bold; text-decoration:none;"
title="modelo.csv">
                  <input class="open" type="submit" value="Download
Cluster.csv"/>
```

```
                </a>
                  </div>

            </div>
        </div>
</body>
</html>
```

9.3 Ejecución de la aplicación

Con el desarrollo anterior del proyecto Cluster_trimm en Django, se iniciará el servidor desarrollo mediante el comando runserver.

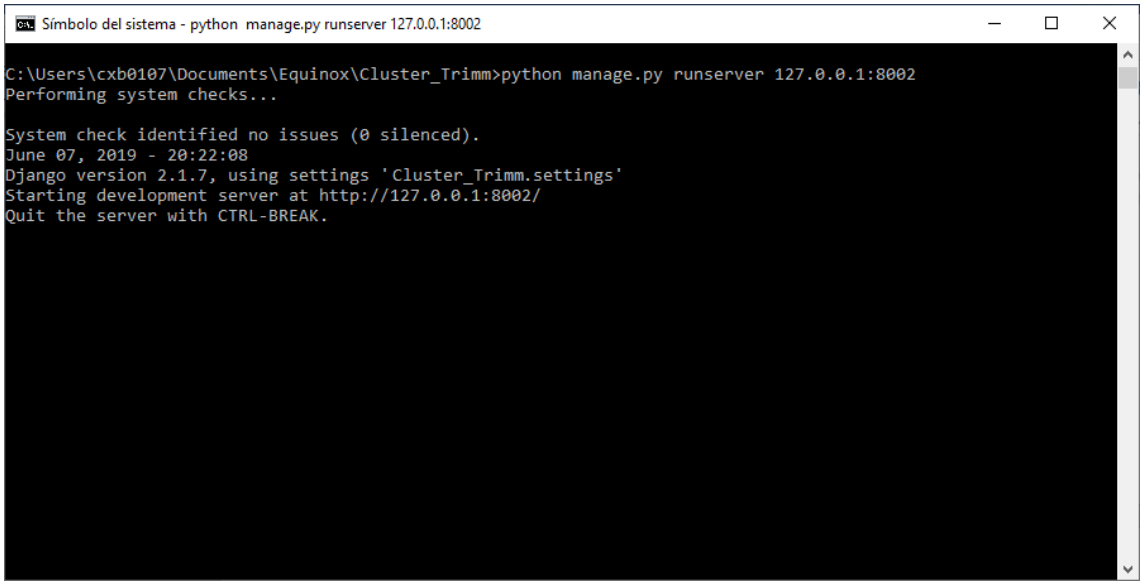


Figura 23: Visualización interactiva de dos clústers aplicado el procedimiento k-medias recortado.

Con el servidor de desarrollo iniciado la aplicación ya estará funcionando en la url: http://127.0.0.1:8002/ y accesible desde cualquier navegador.

Al acceder a la aplicación web la vista de la página principal es la siguiente

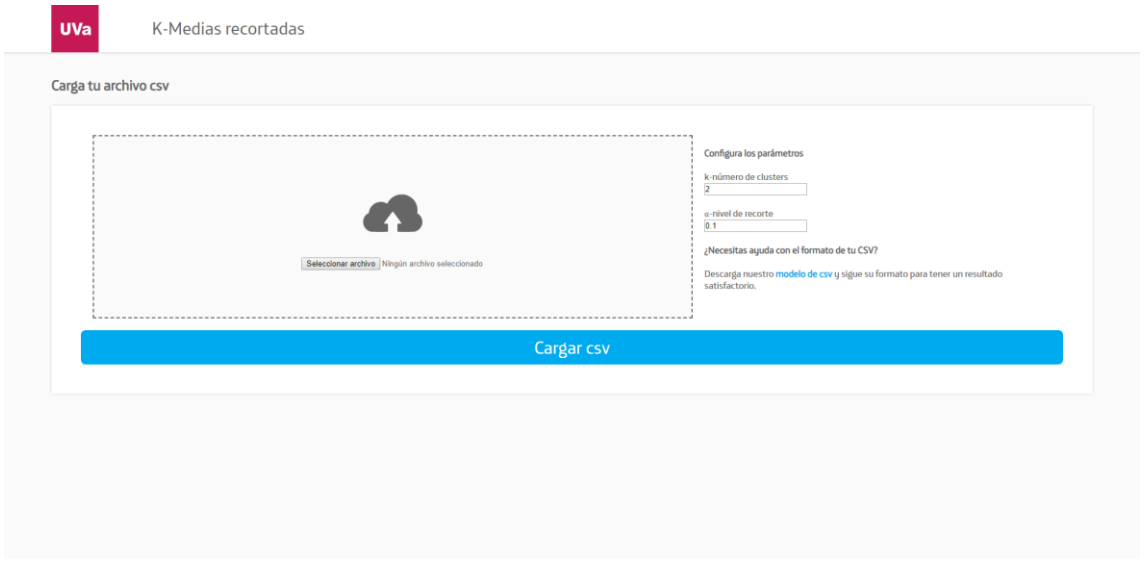


Figura 23: Página principal de la aplicación web.

El usuario podrá cargar un csv con un formato determinado, así como configurar los parámetros k y α, en la zona derecha de la aplicación se dispone de un enlace a un csv de ejemplo aclaratorio para los usuarios.

Al hacer clic en cargar csv se inicia el algoritmo de k-medias recortados sobre el csv seleccionado por el usuario.

Figura 24: Tiempo de carga y ejecución del algoritmo tras cargar el csv.

El mensaje de Cargando datos se mostrará hasta la finalización de toda la ejecución del algoritmo de k-medias recortadas.

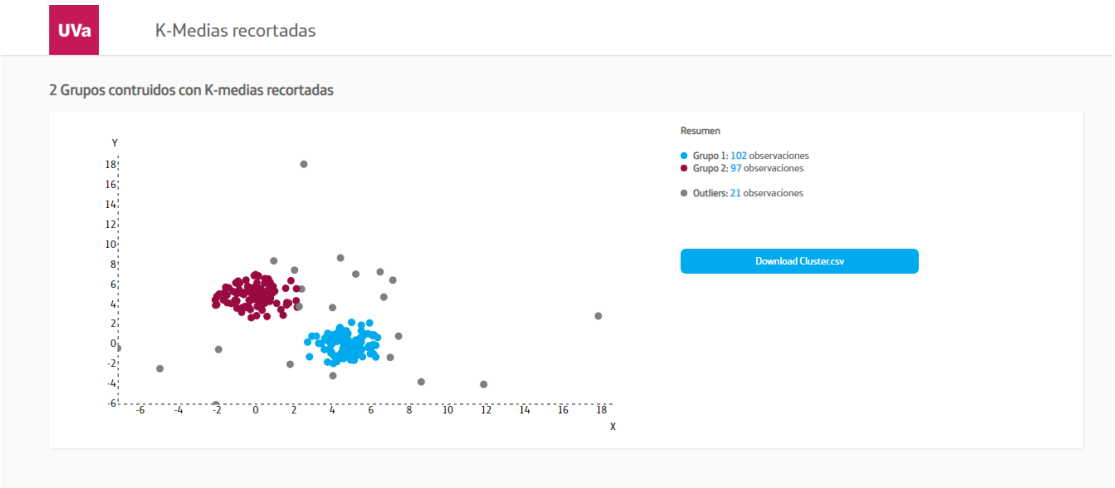


Figura 25: Pagina que muestra los resultados de aplicar el algoritmo y permite su descarga.

Al finalizar la ejecución se muestra visualmente el resultado del algoritmo sobre los datos cargados mediante un csv por el usuario, así como el número de observaciones totales asignadas a cada clúster y las observaciones consideradas como outliers por el procedimiento de recorte.

El usuario puede descargar el csv con el detalle de cada observación ya sea asignado a un clúster o considerado outlier.

10- Conclusiones y líneas futuras

El objetivo principal en el cual se centra este trabajo de fin de grado es desarrollar una aplicación web funcional mediante el uso de herramientas opensource de uso actual y de mayor uso en el campo de la ciencia de datos, dotando a la aplicación del uso de un algoritmo de Análisis Clúster Robusto de manera simple. En este sentido, se han alcanzado las siguientes conclusiones:

1. En lo referente al sistema de Análisis Clúster Robusto

   Se ha comprobado mediante simulaciones que el procedimiento k-medias recortadas es útil para realizar Análisis Clúster en conjuntos de datos que tienen una fracción de observaciones contaminantes. El procedimiento es suficientemente rápido para su implementación a través de una aplicación web. Disponer de herramientas de Análisis Clúster robusta que sea fácilmente accesible pensamos que puede ser de interés en la práctica estadística ya que la presencia de observaciones atípicas suele ser la regla más que la excepción.

2. En lo referente a la selección de herramientas

   Otra de las vías de investigación que se presenta en este TFG es el estudio de herramientas de análisis estadístico con mayor uso en el periodo actual, como son R y Python. El análisis de ambas no reveló grandes diferencias en su uso a nivel de resultados, ni tampoco en materia de tiempos de ejecución. **La opción elegida finalmente para el desarrollo de la aplicación web fue Python**, debido principalmente su naturaleza de lenguaje de propósito general, que permite el uso de herramientas que no son utilizadas de manera exclusiva en la ciencia de datos, como por ejemplo Django que se encuentra también de forma frecuente en el contexto del desarrollo web.

3. Avances alcanzados con desarrollo de la aplicación

   La aplicación web dota de un **entorno amigable** que permite a usuarios poco avanzados en materia de programación y machine learning aplicar un sistema de clasificación avanzado sobre un conjunto de datos bidimensional de manera práctica y sencilla.

   Finalmente permitirá que los usuarios carguen de manera inmediata sus datos y puedan exportarlos, **agilizando los estudios** que necesiten aplicar un algoritmo de clasificación de manera rápida y sencilla.

4. Principales problemas enfrentados durante el desarrollo de la aplicación

   Al ser Python un lenguaje utilizado de manera poco común en el ámbito académico, una de las grandes dificultades enfrentadas a la hora de desarrollar el algoritmo aplicado en este trabajo fin de grado es la elección óptima de los objetos aplicados, debido a que **algunas librerías externas cuyo uso resulta imprescindible pueden ralentizar las ejecuciones** que lleva a cabo la aplicación, por no ser estos objetos nativos.

Otra de las grandes dificultades planteadas ha sido el uso de un entorno Django. Dado que **no se trata de una herramienta de la ciencia de datos**, su documentación, aunque extensa, está orientada en su gran mayoría hacia proyectos pertenecientes a la rama de la ingeniería informática, lo que supone un esfuerzo adicional a la hora de integrarlo en un sistema de esta naturaleza.

5. Puntos de mejora de cara a un mayor desarrollo de la aplicación

Debido al extenso número de campos de estudio en los que resulta implementable esta aplicación web, cabe esperar que se produzcan posteriores desarrollos que permitan **adaptarla a circunstancias de análisis específicas**.

Una posible evolución de la actual aplicación pasaría por proporcionar herramientas que ayuden al usuario a **elegir los parámetros k y α basándose en los propios datos a analizar.** Algunas herramientas de carácter gráfico ya están disponibles en la literatura.

Por otra parte, otras líneas de investigación aplicable en investigaciones futuras es la optimización de ejecuciones, así como la posibilidad de la paralelización y la concurrencia, que ofrece grandes ventajas en grandes conjuntos de datos, donde Python y R disponen de gran cantidad de librerías pensadas para este fin.

Finalmente, otra de las futuras líneas de estudio derivadas de este trabajo podría ser la inclusión de **distintos algoritmos de Análisis Clúster Robusto en esta misma aplicación**, dando al usuario la opción de realizar la selección y establecer comparaciones entre las diferentes opciones. Por ejemplo, sería interesante adaptar a este entorno la metodología “tclust” y que permitiría trabajar con clústers no necesariamente esféricos y con la misma dispersión.

11- Bibliografía

Artículos y libros

Bock, H-H. 2008, Origins and extensions of the k-medias algorithm in cluster analysis, Institute of Statistics, RWTH Aachen University, D-52056 Aachen, Germany

Chatfield, C. y Collins, A.J. 1980, Introduction to Multivariate Analysis, Chapman & Hall

Cuesta-Albertos, J. Gordaliza ,A. Matrán, C. 1997, Trimmed K-means: An attempt to robustify quantizers, The Annals of Statistics, 553-576, Vol. 25

De la Fuente Fernández, S. 2011, Análisis Conglomerados, Facultad de Ciencias Económicas y Empresariales UNAM

Gallardo San Salvador, J.A. 2018, Análisis de Datos Multivariantes, Universidad de Granada

García-Escudero, L.A. Gordaliza, A. Mayo-Iscar A. 2008, A general trimming approach to robust cluster analysis, 1324-1345, Vol. 36

García-Escudero, L.A. y Gordaliza, A. 1999, Robustness Properties of k Medias and Trimmed k Medias, Journal of the American Statistical Association, 956-969, Vol. 94

García-Escudero, L.A., Gordaliza, A., Matrán, C. y Mayo-Iscar, A., 2010, A review of robust clustering methods, Advances in Data Analysis and Classification, 89-109, Vol. 4

García-Escudero, L.A. Gordaliza, A. y Matrán, C. 2003, Trimming Tools in Exploratory Data Analysis, Journal of Computational and Graphical Statistics, 434-449, Vol. 12

Guojun Gan, Chaoqun Ma, Jianhong Wu, 2007, Data Clustering: Algorithms and Applications, Society for Industrial and Applied Mathematics

Peña, D. 2002, Análisis de Datos Multivariantes, Mc Graw Hill.

Srivastava, M.S. 2002, Methods of Multivariate Statistics, Willey

Sitios Web

Manero-Bastin, A. (17 de octubre de 2018), R vs Python: Usability, Popularity, Pros & Cons, Jobs, and Salaries, [Datasciencecentral.com]

https://www.datasciencecentral.com/profiles/blogs/r-vs-python-meta-review-on-usability-popularity-pros-amp-cons

Andrews S. (2 de noviembre de 2015), People check their smartphones 85 times a day, [Ntu.ac.uk]

https://www.ntu.ac.uk/about-us/news/news-articles/2015/11/people-check-their-smartphones-85-times-a-day-and-they-dont-even-know-theyre-doing-it

Bostock, M. (22 de marzo de 2018), Data-Driven Documents, [D3js.org]

https://d3js.org/

Data-Driven Science (30 de enero de2018), Python vs R for Data Science: And the winner is..., [Medium.com]

https://medium.com/@data_driven/python-vs-r-for-data-science-and-the-winner-is-3ebb1a968197

Datapedia, [Luca-d3.com]

https://luca-d3.com/es/diccionario-tecnologico/index.html

Mínguez Salido, R. Estadística Robusta, [Wolterskluwer.es]

http://diccionarioempresarial.wolterskluwer.es/Content/Documento.aspx?params=H4sIAAAAAAAEAMtMSbF1jTAAASNjM2MztbLUouLM_DxbIwMDS0NDA1OQQGZapUt-ckhlQaptWmJOcSoAvvhbpzUAAAA=WKE

Petersson, J. (19 de octubre de 2018) D3 Zoomable Scatterplot, [bl.ocks.org]

http://bl.ocks.org/peterssonjonas/4a0e7cb8d23231243e0e

(12 de septiembre de 2016) Selección del número óptimo de Clústers, [Jarroba.com]

https://jarroba.com/seleccion-del-numero-optimo-clusters/

(27 de enero de 2018) Python 3.7 documentation, [Python.org]

https://docs.python.org/3.7/

(11 de febrero de 2019) Documentación Django, [Djangoproject.com]

https://docs.djangoproject.com/es/2.1/

(18 de marzo de 2019) JavaScript, [Mozilla.org]

https://developer.mozilla.org/es/docs/Web/JavaScript

What is R? [R-project.org]

https://www.r-project.org/about.html