Universidad de Valladolid

FACULTAD DE CIENCIAS

Trabajo Fin de Grado

Grado en Estadística

Análisis comparativo de resultados electorales a nivel de distrito municipal con relación a factores sociodemográficos en Castilla y León

Autor:

Óscar Ferrer Domingo

Tutores:

Alfonso Gordaliza Ramos

Francisco Rodríguez Redondo

Curso 2023/2024

ÍNDICE

Resúmen 5
Abstract 5
1 - INTRODUCCIÓN 6
2 – DATOS 8
2.1 OBTENCIÓN DE DATOS 9
2.2 UNIÓN DE DATOS 11
2.3 DEPURACIÓN DE DATOS 14
3 – ANÁLISIS DE DATOS 15
3.1 ANÁLISIS DESCRIPTIVO 15
3.2 REDES BAYESIANAS 23
3.3 ANÁLISIS DE COMPONENTES PRINCIPALES 27
3.4 ANÁLISIS CLÚSTER 37
3.5 ANÁLISIS MULTIVARIANTE 43
4 - CONCLUSIONES 46
Bibliografía 48
LISTA DE FIGURAS 49
LISTA DE TABLAS 50
ANEXO DE CÓDIGOS 51
UNIÓN DE DATOS 64
DEPURACIÓN DE DATOS 65
ANÁLISIS DESCRIPTIVO 66
REDES BAYESIANAS 68
ANÁLISIS DE COMPONENTES PRINCIPALES 72
ANÁLISIS CLÚSTER 82
ANÁLISIS MULTIVARIANTE 86

Resúmen

El objetivo principal de este proyecto ha sido el de analizar los resultados correspondientes tanto a las elecciones a las Cortes de Castilla y León, el 13 de febrero de 2022, como a las elecciones generales del 23 de julio de 2023 circunscritos a la comunidad autónoma de Castilla y León. Habiéndose obtenido los datos de fuentes oficiales se buscan posibles relaciones entre la evolución del voto de una elección a otra (variables electorales) y los datos sociodemográficos disponibles (variables sociodemográficas). Los resultados se han obtenido mediante utilizando el lenguaje Python y el de R en el entorno de desarrollo R-Studio.

Abstract

The main objective of this project has been to analyze the results of both the elections to the Cortes of Castilla y León on February 13, 2022, and the general elections on July 23, 2023, within the autonomous community of Castilla y León. Using data obtained from official sources, the project seeks to identify possible relationships between the evolution of the vote from one election to another (electoral variables) and the available sociodemographic data (sociodemographic variables). The results have been obtained by using the Python language and R in the R-Studio development environment.

1 - INTRODUCCIÓN

La finalidad de este trabajo consiste en buscar relaciones, de existir, entre la evolución del voto de las elecciones a las Cortes de Castilla y León del 13 de febrero a las elecciones generales del 23 de julio de 2023, y diferentes factores sociodemográficos que se tendrán en cuenta.

El descubrimiento de relaciones entre la evolución electoral y las variables sociodemográficas que definen a distintas zonas del territorio será de interés a la hora de mejorar las predicciones electorales en futuras elecciones. Este conocimiento también podrá utilizarse para mejorar el diseño de las campañas electorales, ayudando a un uso más eficiente de los fondos destinados a estas.

Tanto la información de la parte electoral en ambas elecciones, como la correspondiente a los datos sociodemográficos se ha obtenido de fuentes oficiales, como son: el Instituto Nacional de Estadística (INE), la Junta de Castilla y León (jcyl) y el Ministerio del Interior (MIR). A la hora de recopilar los datos anteriores se ha elegido seleccionar las variables, tanto electorales como sociodemográficas que se han recogido a nivel de mesa electoral o sección censal, que luego se han agrupado para trabajar a nivel de distrito municipal. Se ha utilizado una variable que identifica cada sección censal para unir los archivos originales de distintas fuentes, realizando los cambios necesarios para que la tabla resultante contenga variables útiles. Por último, se ha procedido a la depuración de los datos, corrigiendo los errores producidos después de unir los archivos individuales, principalmente en cuanto a diferencias en la calidad y formato de los datos recogidos en los distritos municipales observados.

Las observaciones provienen de distritos municipales de todo Castilla y León, sumando un total de **2362 distritos municipales** distintos. Al hacer un recuento del origen de estos tenemos que un 11.09% (262 distritos) provienen de **Ávila**, un 16.34% (386 distritos) de Burgos, un 9.82% (232 distritos) de **León**, un 8.47% (200 distritos) de **Palencia**, un 15.83% (374 distritos) de **Salamanca**, un 9.36% (221 distritos) de **Segovia**, un 7.83% (185 distritos) de **Soria**, un 10.37% (245 distritos) de **Valladolid**, y un 10.88% (257 distritos) de **Zamora**.

Una vez recogidos y tratados los datos de las diferentes fuentes se han realizado varias técnicas. En primer lugar, se ha reducido el conjunto final de variables mediante un estudio de las correlaciones entre las variables sociodemográficas recogidas, lo que ha disminuido significativamente el número de variables en el conjunto final.

Para tener un entendimiento inicial de las influencias que tienen las covariables en cada una de las variables electorales de diferencia porcentual de voto, se han obtenido redes bayesianas que clarifican sus relaciones condicionadas. Esto nos permite sacar conclusiones a partir de las independencias condicionadas observadas entre las variables. Destaca la identificación de las ‘variables frontera’ como aquellas entre las electorales con mayor interacción con las covariables.

Para reducir aún más el número de variables intentando perder la menor cantidad de información posible se ha llevado a cabo un Análisis de Componentes Principales. El propósito de este tipo de análisis dentro del trabajo ha sido encontrar de forma gráfica alguna de las relaciones buscadas.

Al aplicarse el análisis anterior, en el grupo inicial de distritos municipales se han encontrado pocas relaciones de interés entre los dos tipos de variables.

Además, se ha realizado un Análisis Clúster y un Análisis Multivariante para interpretar los valores de los clústeres obtenidos.

El análisis clúster tiene como objetivo dividir los datos en particiones de manera que los individuos dentro de cada una sean lo más similares posible entre sí y difieran lo máximo posible respecto a los individuos de otras particiones. En este caso, se utilizaron el método k-medoids y el método jerárquico. Una de las diferencias clave entre estos métodos es que, en el primero, es necesario establecer a priori el número de grupos que se espera obtener, y en el segundo se obtiene una jerarquía sobre la que cortar a posteriori. En este trabajo, se consideraron 9 clústeres en ambos métodos, ya que era el número óptimo obtenido para k-medoids, seleccionando finalmente el clustering jerárquico con complete linkage como el más adecuado.

Para interpretar los resultados del análisis clúster se realiza una regresión multinomial. Es una extensión de la regresión logística que se utiliza cuando la variable dependiente es categórica y tiene más de dos niveles o categorías, a diferencia de la regresión logística binaria puede manejar múltiples categorías simultáneamente. Con esta regresión se observan las variables que son especialmente decisivas en la clasificación de los distritos municipales en los distintos grupos y lo que esto puede implicar. Se destacan en su influencia las ‘variables frontera’ anteriormente mencionadas.

2 – DATOS

Este capítulo consistirá en una explicación detallada de los pasos seguidos para obtener los datos que se utilizarán en los análisis. Comenzará con la recopilación de tablas proporcionadas por diversas fuentes y concluirá con la unión de todas ellas y su posterior depuración.

A lo largo del documento distinguiremos los datos con los que se va a trabajar agrupándolos en dos conjuntos diferentes, contando cada uno de ellos con diferentes variables.

Por un lado, variables **electorales**, o de diferencia en el porcentaje de votos:

- Diferencia porcentual de Votos Nulos
- Diferencia porcentual de Votos en Blanco
- Diferencia porcentual de Cs (solo primer análisis)
- Diferencia porcentual de ESPAÑA VACIADA
- Diferencia porcentual de PODEMOS/SUMAR
- Diferencia porcentual del PP
- Diferencia porcentual del PSOE
- Diferencia porcentual de Soria Ya
- Diferencia porcentual de UPL
- Diferencia porcentual de VOX
- Diferencia porcentual de XAV
- Diferencia porcentual en Abstención
- Diferencia porcentual de Partidos Minoritarios

Y como covariables las variables **sociodemográficas**:

- Censo
- Medida de desigualdad de distribución de la renta P80/P20
- Índice de Gini
- Edad media de la población
- Población total
- Porcentaje de población española
- Tamaño medio del hogar
- Mediana de la renta por unidad de consumo
- Renta neta media por hogar
- Renta neta media por persona
- Ingresos por ‘otras prestaciones’
- Ingresos por ‘otros ingresos’
- Ingresos por ‘pensiones’
- Ingresos por ‘prestaciones por desempleo’
- Ingresos por ‘salario’

Además, se contará con una variable que actúe como **identificador** de las diferentes observaciones, para facilitar la unión de datos llamada IDM.

Las diferencias porcentuales entre los resultados electorales de los partidos se definen para cada partido como:

DiferenciaPorcentual = VotoPorcentualGenerales – VotoPorcentualCortes

Por lo que un valor positivo significara una mayor proporción del voto en las elecciones generales, y uno negativo una mayor proporción en las elecciones a las cortes.

2.1 OBTENCIÓN DE DATOS

Es importante que queden reflejadas las fuentes de las que se han obtenido los datos, así como su referencia temporal, puesto que, de producirse algún fallo a la hora de trabajar con ellos, podrán revisarse de forma sencilla y rápida. Teniendo en cuenta que se utilizarán los resultados de las elecciones a las Cortes del 13 de febrero de 2022 en Castilla y León y las elecciones generales de 23 de julio de 2023, todos los datos sociodemográficos se seleccionarán tan próximos a la actualidad como sea posible.

Los datos iniciales, se han obtenido mediante consultas personalizadas a partir de las siguientes fuentes, seleccionando para cada una de las 9 provincias de Castilla y León. Se han recogido a nivel de distrito municipal o sección censal:

1. (Junta de Castilla y León, 2022)
    - 13 de febrero de 2022
    - Datos electorales Cortes:
        - Código de mesa
        - Nombre de Provincia
        - Código de Municipio
        - Nombre de Municipio
        - Código de Distrito
        - Código de sección
        - Letra de mesa
        - Censo electoral (unidad personas)
        - Votos en primer avance (unidad votos)
        - Votos en segundo avance (unidad votos)
        - Total de votantes (unidad personas)
        - Votos Nulos (unidad votos)
        - Votos Blancos (unidad votos)
        - Nombre de Partido
        - Número de votos (unidad votos)

2. (Ministerio del Interior, 2023)
    - 23 de julio de 2023
    - Datos electorales generales:
        - Código de la Comunidad Autónoma
        - Código de Provincia
        - Nombre de Provincia
        - Código de Municipio
        - Nombre de Municipio
        - Código del distrito municipal
        - Código de sección censal
        - Código de mesa
        - Total censo electoral (unidad personas)

        - Votos Primer Avance (unidad votos)
        - Votos Segundo Avance (unidad votos)
        - Votos válidos (unidad votos)
        - Votos en blanco (unidad votos)
        - Votos nulos (unidad votos)
        - Nombre partido (unidad votos)
3. (INE, 2021)
    - Año 2021
    - Desigualdad de renta por sección censal:
        - Índice de Gini
        - Distribución de la renta P80/P20
    - Distribución de renta de los hogares por sección censal:
        - Edad media de la población
        - Población
        - Porcentaje de menores de 18
        - Porcentaje de 65 años o más
        - Tamaño medio del hogar
        - Porcentaje de población española
        - Renta neta media por hogar
        - Renta neta media por persona
        - Renta bruta media por hogar
        - Renta bruta media por persona
        - Media de la renta por unidad de consumo
        - Mediana de la renta por unidad de consumo
        - Distribución por fuentes de ingresos
4. (INE, 2023)
    - Año 2023
    - Población por sexo y edad (grupos quinquenales)
        - Secciones
        - Sexo
        - Edad (grupos quinquenales)
        - Total (habitantes)

Para simplificar el problema en cuanto a la gran variedad de partidos encontrados en pequeños municipios se ha aplicado el corte del 5%, como se suele hacer en las elecciones locales para la aplicación de la Ley D'hondt. De esta forma se mantienen los datos de aquellos partidos que representen más de un 5% de los votos en su provincia, y se agrupa el resto como 'Partidos Minoritarios'.

Cabe destacar la problemática que supone la diferencia de datos del partido Ciudadanos. Puesto que se presentó a las elecciones de las Cortes de Castilla y León pero no a las elecciones generales, nos vemos obligados a tratarlo de forma especial. El hecho de que no sea un partido minoritario, pues supera el corte asumido, se añade a la complicación.

En el inicio del trabajo la intención era recoger todas las observaciones a nivel de sección censal, pero la existencia de varios problemas obligó a subir el nivel de representación hasta el de distrito censal, como fueron:

- Creación y borrado de secciones censales entre los periodos de ambas elecciones.
- Dificultad de conocer con exactitud las consecuencias que estos cambios suponen en cuanto a la recogida de datos.
- Gran cantidad de missings existentes en muchas variables sociodemográficas a dicho nivel.
- Heterogeneidad marcada en los formatos de los datos iniciales y la escasez de estos a ese nivel.

Este cambio en el nivel final de las variables puso fin a muchas operaciones de preprocesado que se presentaban exageradas a la hora de extraer covariables limpias para el estudio.

2.2 UNIÓN DE DATOS

Una vez descargados los datos que se emplearán en los análisis posteriores, el siguiente paso es juntarlos en una misma tabla, puesto que se encuentran en archivos diferentes. En nuestro caso primero compilaremos dos tablas, una para cada conjunto de variables, según los hemos definido anteriormente. Una vez que tengamos ambas tablas las juntaremos en un dataset final sobre el que realizar los distintos análisis.

Esta tarea se realizará mediante una variable común en todos los archivos llamada **IDM** (Identificador de Distrito Municipal). Esta variable ha sido creada para actuar como índice en las distintas tablas, de forma que haga simple la agrupación de observaciones en cuanto a los distritos municipales en los análisis posteriores. La dificultad que se encuentra en la unión es doble. Por un lado, algunos de los datos vienen dados por código de mesa, por lo que se necesitará una agrupación correcta de estas para ordenar las observaciones por sección censal y agrupar estas secciones en distritos municipales. Y, por el otro, los datos electorales generales diferencian una observación para los resultados de cada partido en cada mesa, por lo que se deberán crear nuevas variables para cada partido de dichas elecciones, traduciendo los datos correctamente. A continuación, se explican todas las soluciones que se han ido aplicando hasta poder realizar la unión de todas las tablas por dicha variable.

El primer lugar, se crea la variable IDM en el formato:

- **Código Provincia, Código Municipio y Código Distrito Municipal**

La creación de esta variable se basó en el uso de los datos recogidos desde ministerio del interior de cadenas de caracteres numéricos con el fin de obtener identificadores únicos los cuales, con ayuda de etiquetas, contuvieran mucha información. Más específicamente se trata de números de 7 dígitos formados por la concatenación de los códigos especificados arriba, estos son los identificadores utilizados por el INE en cada caso. Esta unión se realiza de la siguiente forma:

Ilustración 1.IDM

Una vez se ha construido esta variable en las distintas tablas se utiliza como índice común en la unión de estas. Se utilizará primero una unión interior en la que se guardan las observaciones coincidentes a la hora de juntar los datos electorales de ambas elecciones. Este tipo de unión tiene el riesgo de no recoger todas las observaciones en caso de no haber coincidencias, lo que conllevaba perdidas a nivel de sección censal, pero no es un problema con los distritos municipales.

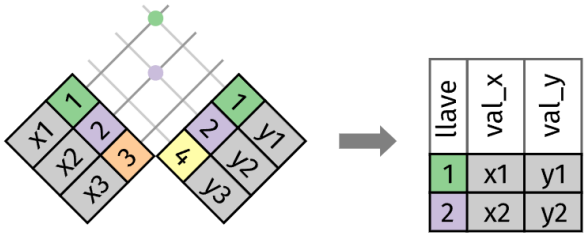

Ilustración 2.Unión interior

En segundo lugar, se utilizará una unión exterior izquierda que relacione los datos electorales con el resto de las variables, lo que implica observaciones adicionales en caso de no coincidencia.

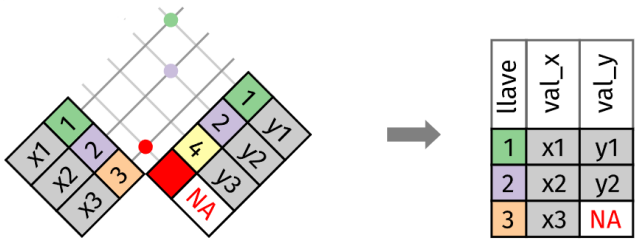

Ilustración 3.Unión exterior izquierda

Todos estos cambios se realizarán mediante códigos en lenguaje Python, utilizando archivos csv como entradas y salidas.

Extraemos los datos de las elecciones generales, y extendemos los partidos políticos como nuevas variables:

> Interpretamos los datos crudos del ministerio del interior en sus variables.

- Utilizamos la función *'read.fwd'* de pandas, y especificando las posiciones de cada variable en la cadena de caracteres, según nos indican las etiquetas correspondientes.

> Aislamos observaciones de Castilla y León.

- Utilizamos la función *'loc'* para aislar las observaciones con código de comunidad autónoma igual al de Castilla y León.
- Aplicamos el *'merge'* para añadir las siglas de los partidos al dataframe.

> Creamos IDM en la tabla de datos.

- Concatenamos los códigos especificados anteriormente, asegurándonos de que la conversión no elimina los ceros intermedios.

> Pasamos distintos partidos de observaciones duplicadas a nuevas variables.

- Realizamos un pivot utilizando *'pivot_table'* utilizando las siglas de los partidos como nuevas columnas, y repartiendo el número de votos entre ellas. Esto es posible al utilizar la nueva variable IDM como índice, pues esta repetida por cada mesa en cada sección censal.

> Agrupamos las observaciones resultantes por secciones censales.

- Hacemos *'groupby'* en torno a la variable índice IDM para colapsar el resto de las variables en los valores correspondientes a secciones censales y a distritos municipales.

Extracción datos elecciones Cortes de Castilla y León:

> Hacemos pivot por código de mesa

- De nuevo utilizamos *'pivot_table'* siendo el nuevo índice el código de mesa.

> Nos deshacemos de partidos duplicados

- Utilizando *'drop_duplicates'*

> Añadido variable IDM en elecciones cortes:

- Seleccionamos el tipo de cadena del que partimos en Código de mesa con *'extract('^([\d-]+)')'*
- Eliminamos los caracteres '-' dejando los dígitos efectivamente concatenados.
- Escogemos los primeros 7 caracteres de la variable resultante.

Extracción de covariables:

> Creamos variables sexo y edad

- Filtramos número de individuos por sexo, eliminando observaciones a nivel superior a sección censal, obtenemos 'TotalHombres' y 'TotalMujeres'.
- Agrupamos secciones en distritos municipales.
- Creamos nuevas variables para el número de individuos para cada intervalo quinquenal por edades.
- Agrupamos edades en variables para intervalos de 10 años.

> Tratamiento de missings

- Son sustituidos por la media ponderada de observaciones en cada variable, de forma que se faciliten el análisis posterior al dataset final.

Una vez hemos tratado todas las variables y las tenemos en sus respectivas tablas finales se utilizan en los distintos análisis.

2.3 DEPURACIÓN DE DATOS

La principal herramienta de depuración de los datos más allá de las comentadas en el tratamiento de unión del apartado anterior ha sido la criba de variables mediante la introducción de un máximo de correlación entre las variables sociodemográficas a incluir en los análisis.

Antes del cribado de variables contaba con 34 covariables (sin contar las variables electorales), que eran:

Censo, Tvotos, Distribución de la renta P80/P20, Índice de Gini, Edad media de la población, Población, Porcentaje de población de 65 y más años, Porcentaje de población española, Porcentaje de población menor de 18 años, Tamaño medio del hogar, Media de la renta por unidad de consumo, Mediana de la renta por unidad de consumo, Renta bruta media por hogar, Renta bruta media por persona, Renta neta media por hogar, Renta neta media por persona, Fuente de ingreso: otras prestaciones, Fuente de ingreso: otros ingresos, Fuente de ingreso: pensiones, Fuente de ingreso: prestaciones por desempleo, Fuente de ingreso: salario, 0-9 años, 10-19 años, 20-29 años, 30-39 años, 40-49 años, 50-59 años, 60-69 años, 70-79 años, 80-89 años, 90-99 años, 100+ años, TotalHombres y TotalMujeres.

Sacamos una matriz de correlaciones entre estas covariables tomando como **baremo una correlación igual o superior a 0.8** entre las variables. Esto se hace para disminuir el número de variables en el conjunto final de datos, lo que ayudará a no exagerar la dimensionalidad de los análisis y a obtener resultados gráficos más claros. No tener covariables altamente correlacionadas ha influido también negativamente en la efectividad de alguno de los análisis, principalmente del de componentes principales.

Tras aplicar el baremo nos quedaremos con las variables sociodemográficas agrupadas anteriormente, eligiendo deshacernos de:

Tvotos, Porcentaje de población de 65 y más años, Porcentaje de población menor de 18 años, Media de la renta por unidad de consumo, Renta bruta media por hogar, Renta bruta media por persona, 0-9 años, 10-19 años, 20-29 años, 30-39 años, 40-49 años, 50-59 años, 60-69 años, 70-79 años, 80-89 años, 90-99 años, 100+ años, TotalHombres y TotalMujeres.

3 – ANÁLISIS DE DATOS

En este capítulo, se buscará cumplir con el objetivo del trabajo: establecer relaciones entre las variables sociodemográficas y electorales. Una vez obtenidos los datos necesarios, se utilizarán técnicas multivariantes conocidas para identificar dichas relaciones.

En primer lugar, se llevará a cabo un análisis descriptivo de los datos, donde se presentarán las características principales de cada variable en la tabla.

Acto seguido se construirán varias redes bayesianas en busca de conclusiones iniciales sobre las relaciones entre variables.

Posteriormente, se aplicarán técnicas multivariantes al conjunto inicial de distritos municipales, como el Análisis de Componentes Principales para intentar establecer relaciones gráficas, un Análisis Clúster sobre los distritos municipales y un Análisis Multivariante utilizando las características de los clústeres obtenidos.

Con estos resultados se espera poder llegar a conclusiones suficientes sobre el nivel de relación existente entre las variables.

3.1 ANÁLISIS DESCRIPTIVO

Esta fase inicial del análisis de datos se centra en la utilización de técnicas de estadística descriptiva para facilitar la interpretación de los datos. Se crearán gráficos y tablas que permitirán formular conclusiones preliminares sobre las posibles relaciones entre las variables sociodemográficas y electorales.

Los objetivos principales de esta etapa son, por un lado, presentar las características individuales de las variables y, por otro, descubrir patrones y relaciones entre las mismas. El análisis exploratorio se suele estructurar en función del tipo de variables con las que se trabaja, diferenciando entre variables categóricas y numéricas. La tabla final contiene únicamente variables numéricas.

En nuestro caso tratamos con distintos tipos de covariables en el nivel de precisión final de distrito municipal, como son las siguientes.

Variables Numéricas

Censo

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
6	76	165.2	819.0	411	56160

Tabla 1.Censo

La mayoría de distritos municipales parece rondar los cientos de habitantes, con excepciones en tanto por arriba como por abajo, aunque los outliers más marcados son los habitantes en las capitales de provincia, que a veces se cuentan en decenas de miles de habitantes.

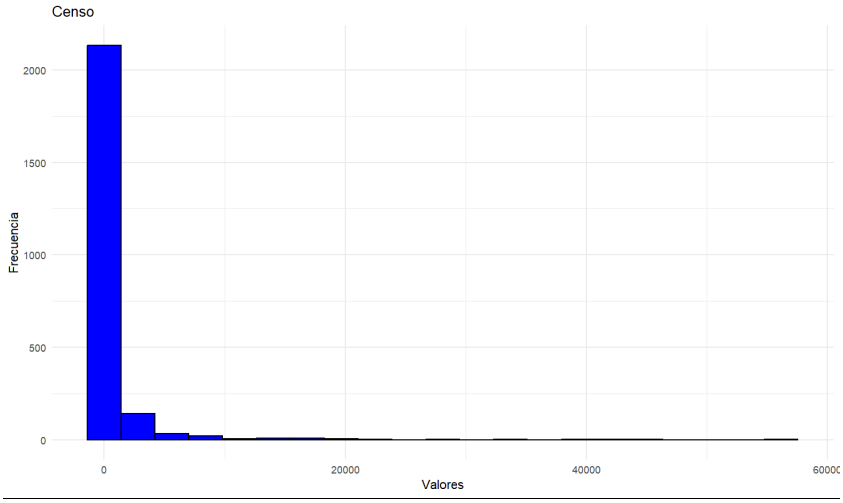

Ilustración 4.Censo

Distribución de la renta P80/P20

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
1.80	2.30	2.41	2.41	2.50	4.20

Tabla 2.Distribución de la renta P80/P20

La distribución de esta variable es centrada con una gran cantidad de valores muy próximos a la media en 2.41, lo que indica una pendiente alta en la evolución de esta en el histograma.

Vemos que con un valor máximo de 4.20, existen observaciones superiores a la media, pero teniendo en cuenta que el 75% de las observaciones están por debajo de 2.50, los valores tan altos se podrían considerar outliers.

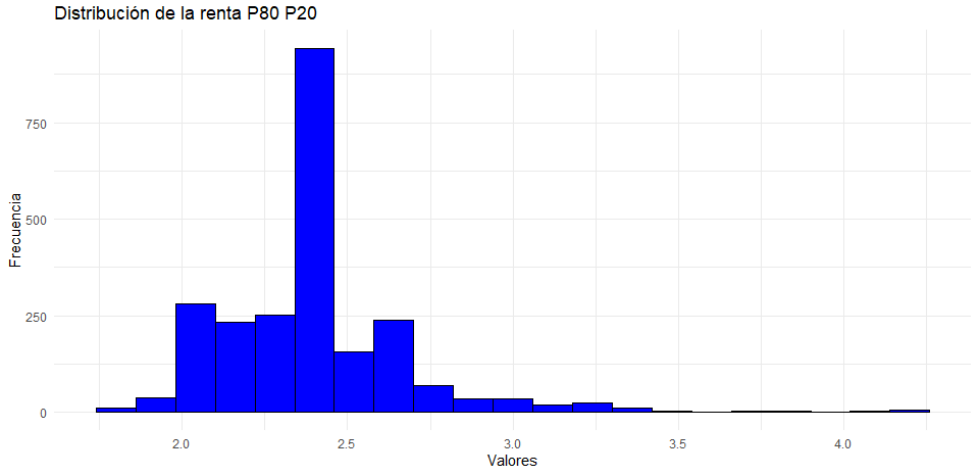

Ilustración 5.Distribución de la renta P80/P20

Índice de Gini

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
20.30	26.80	28.15	28.15	29.00	43.30

Tabla 3.Índice de Gini

La distribución de esta variable, que al igual que la anterior es una medida de desigualdad, es casi perfectamente centrada respecto al valor de la media en 28.15, donde se centran la mayoría de los valores, pues la mediana vale lo mismo. La forma de campana del histograma es más clara en este caso.

Se observan también escasos valores cercanos al máximo, probablemente pertenecientes a las mismas observaciones que aquellos en la variable anterior.

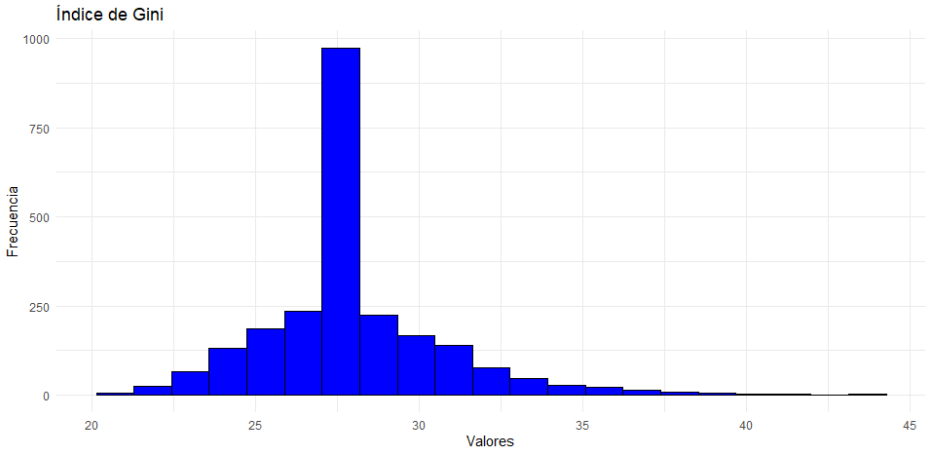

Ilustración 6.Índice de Gini

Edad media de la población

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
33.60	50.90	55.20	54.69	58.80	76.00

Tabla 4.Edad media de la población

Distribución centrada con cola a la izquierda, con media y mediana muy próximas. Su asimetría se hace clara al observar una mayor distancia entre los valores de la moda y el primer cuartil, y la moda y el tercer cuartil. Las observaciones en los extremos no parecen contar con outliers.

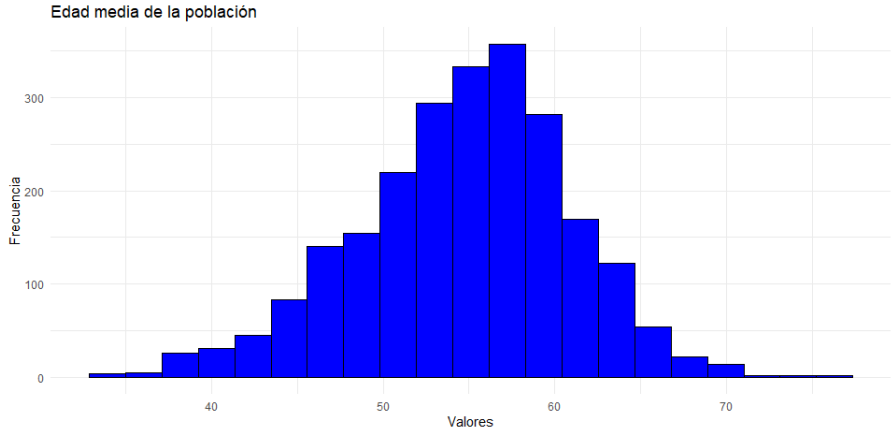

Ilustración 7.Edad media de la población

Población

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
1.00	80.25	174.00	915.91	442.00	67266.00

Tabla 5.Población

La distribución es centrada con una cola asimétrica a la derecha. El valor de la media se ve afectado por los claros outliers de valor superior a 15.000. La mayoría de los valores de población en los distritos municipales son inferiores a 1.000, como se observa en los valores de los cuartiles y la mediana, el tercer cuartil es inferior a 500.

La gran diferencia entre recuentos de población en distritos municipales como pueblos, y aquellos más poblados en las ciudades provinciales, es clara con estos outliers.

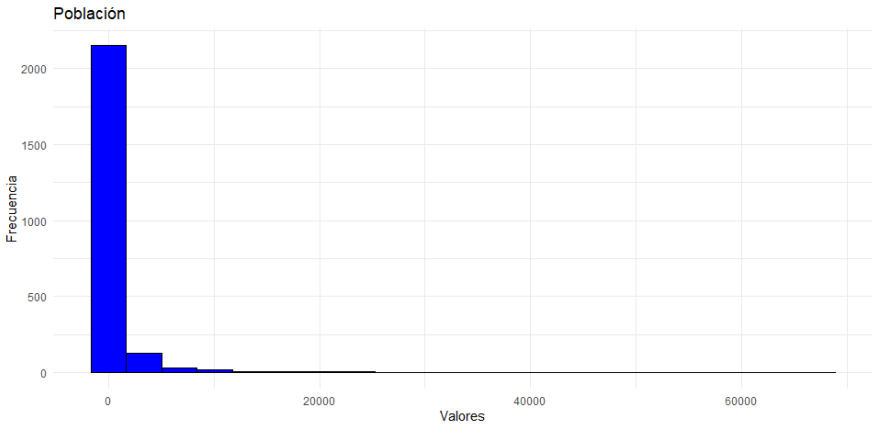

Ilustración 8.Población

Porcentaje de población española

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
56.20	95.08	95.08	95.08	95.08	100.00

Tabla 6.Porcentaje de población española

Los datos referentes al porcentaje de población española se vieron especialmente afectados por la existencia de missings, lo que obligó a tratarlos mediante una sustitución con la media de valores en el resto de la comunidad autónoma. El resultado es una distribución un tanto más pronunciada en torno a la media (95.08), pero que influirá en la relación con otras variables en el análisis mucho menos en comparación a sustituirlos con ceros, como se había optado en los datos originales.

En general se observa una distribución con valores muy próximos al 100%, la gran mayoría de estos entre el 90% y el 100%. Tanto la media como la mediana (antes de la sustitución sin los missings, y después da esta) caen en esta franja.

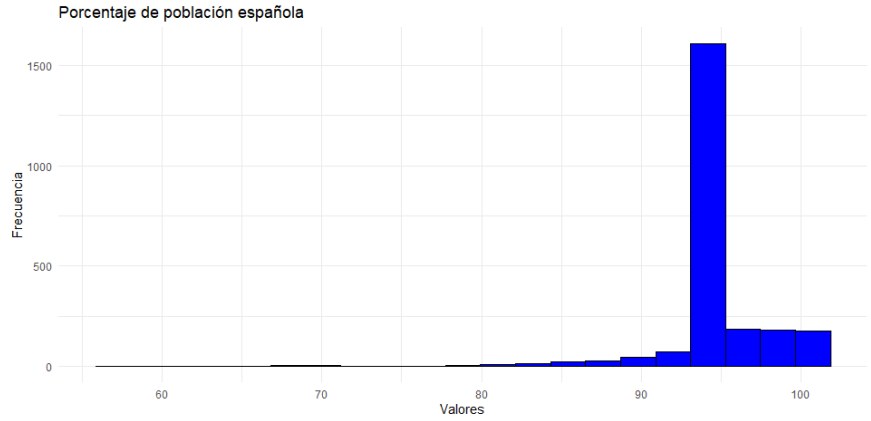

Ilustración 9.Porcentaje de población española

Tamaño medio del hogar

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
1.130	1.920	2.100	2.105	2.280	7.000

Tabla 7.Tamaño medio del hogar

El tamaño medio del hogar tiene una distribución centrada que gira en torno a un valor de 2 en general, como se puede observar por los valores de la media y mediana, cercanos a 2.1, y a los valores de ambos cuartiles también cercanos a 2. Existen observaciones extremas con valores superiores a 3 llegando hasta el 7 de tamaño medio que observamos en el máximo.

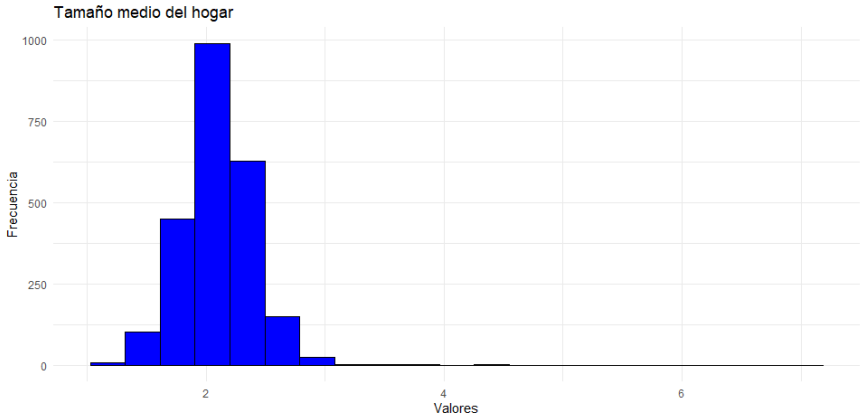

Ilustración 10.Tamaño medio del hogar

Mediana de la renta por unidad de consumo

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
10150	15750	16559	16559	17150	29750

Tabla 8.Mediana de la renta por unidad de consumo

La mediana de la renta por unidad de consumo tiene una distribución que se concentra fuertemente alrededor de 16.559, con una baja dispersión alrededor de este valor. Los valores están relativamente distribuidos de manera simétrica alrededor de la media y la mediana, lo cual sugiere que no hay un sesgo significativo en los datos.

Hay pocos valores extremos en ambos extremos del rango, pero no son lo suficientemente numerosos como para afectar significativamente la media y la mediana.

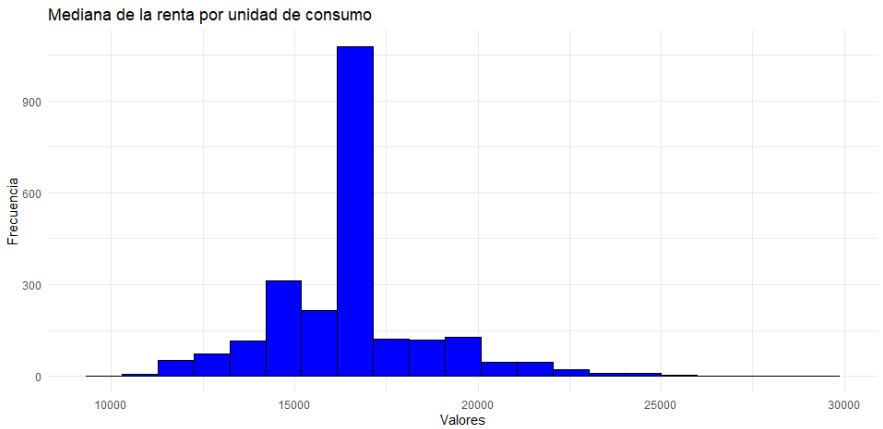

Ilustración 11.Mediana de la renta por unidad de consumo

Renta neta media por hogar

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
1903	24250	28803	28281	31035	71987

Tabla 9.Renta neta media por hogar

La variable posee una distribución que se concentra fuertemente alrededor de 28.803, con una ligera asimetría a la izquierda. La media es un poco menor que la mediana, lo que sugiere la presencia de algunos valores bajos que pueden estar afectando la media. Los valores están relativamente distribuidos de manera simétrica alrededor de la mediana, pero hay unos pocos valores extremos altos que afectan la distribución.

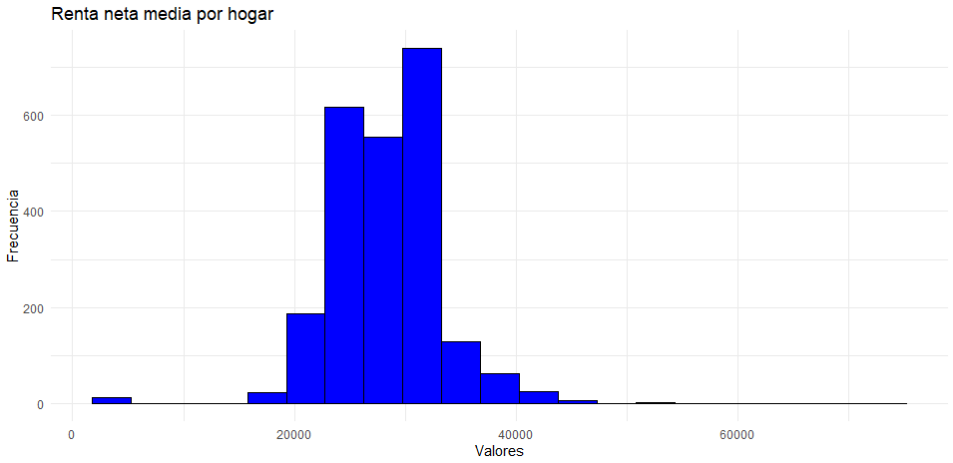

Ilustración 12.Renta neta media por hogar

Renta neta media por persona

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
1008	12109	13094	13409	14790	24109

Tabla 10.Renta neta media por persona

La distribución se concentra principalmente alrededor de 13.094, con una leve inclinación hacia la derecha. La media es ligeramente superior a la mediana, lo que indica la presencia de algunos valores altos que podrían estar influyendo en la media. Los valores están distribuidos de forma relativamente simétrica alrededor de la mediana, aunque estos valores extremos altos afectan la distribución.

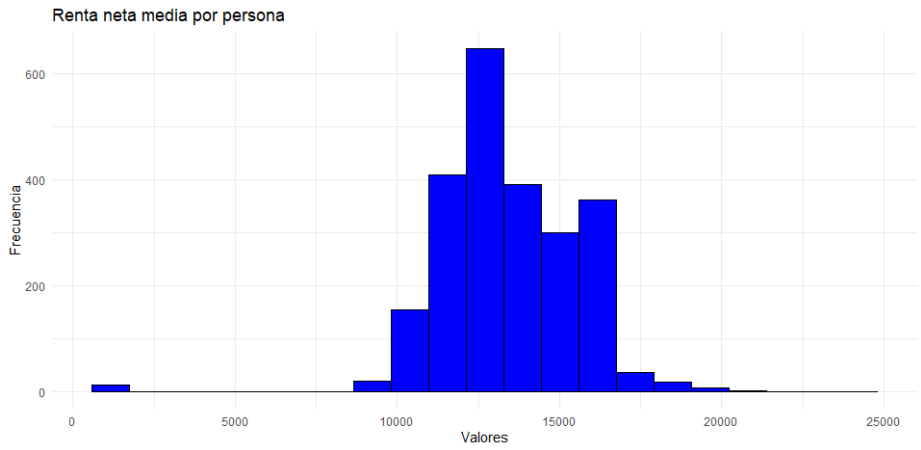

Ilustración 13.Renta neta media por persona

Fuente de ingreso: otras prestaciones

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
191.0	563.2	653.5	653.5	698.0	1336.0

Tabla 11.Fuente de ingreso por otras prestaciones

La distribución es centrada con una frecuencia superior no muy destacada que se ve hinchada por la sustitución de los missings con el valor de la media.

Existen valores atípicos por los dos extremos que afectan a la distribución, pero son lo suficientemente comunes para no ser tratados como outliers.

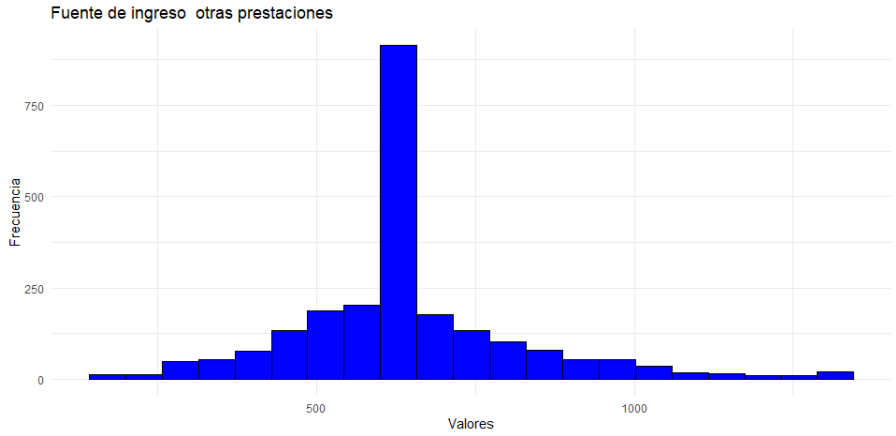

Ilustración 14.Fuente de ingreso por otras prestaciones

Fuente de ingreso: otros ingresos

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
101	1782	2415	2415	2528	10664

Tabla 12.Fuente de ingreso por otros ingresos

La distribución se concentra principalmente alrededor de 2.400, con una leve inclinación hacia la derecha. La media es igual a la mediana, lo que indica la presencia de una frecuencia muy alta en ese valor. Los valores están distribuidos de forma relativamente simétrica alrededor de la mediana, aunque estos valores extremos altos afectan la distribución.

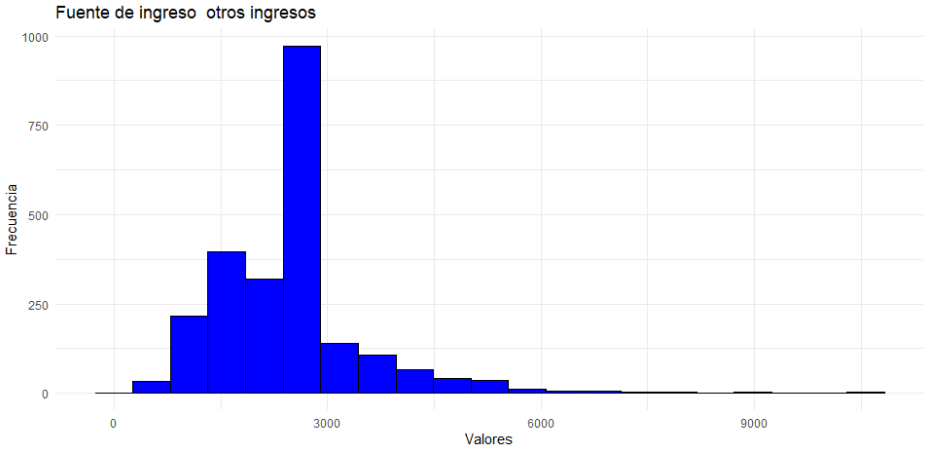

Ilustración 15.Fuente de ingreso por otros ingresos

Fuente de ingreso: pensiones

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
1162	3898	4397	4397	4719	8623

Tabla 13.Fuente de ingreso por pensiones

La distribución se concentra principalmente alrededor de 4.397, con una simetría característica de la centralidad. La media es igual a la mediana, lo que indica la presencia de una frecuencia muy alta en ese valor. Los valores están distribuidos de forma relativamente simétrica alrededor de la mediana, con valores extremos tanto por encima como por debajo de esta.

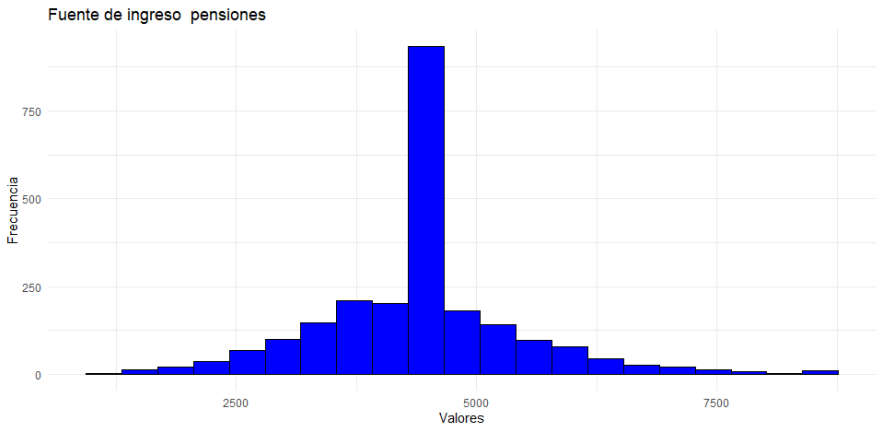

Ilustración 16.Fuente de ingreso por pensiones

Fuente de ingreso: prestaciones por desempleo

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
146.0	331.0	380.5	380.5	420.0	1277.0

Tabla 14.Fuente de ingreso por prestaciones por desempleo

La distribución se concentra principalmente alrededor de 380,5, con una leve inclinación hacia la derecha. La media es igual a la mediana, lo que indica la presencia de una frecuencia muy alta en ese valor. Los valores están distribuidos de forma relativamente simétrica alrededor de la mediana, aunque la existencia de valores extremos altos afecta la distribución.

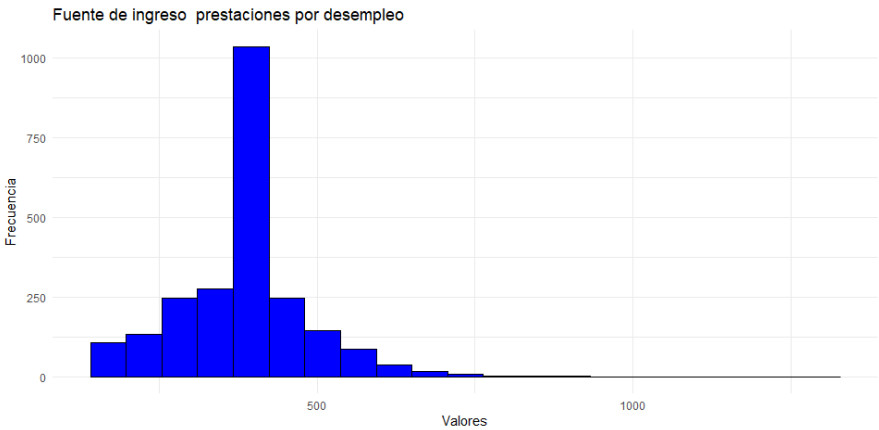

Ilustración 17.Fuente de ingreso por prestaciones por desempleo

Fuente de ingreso: salario

Min.	1º cuartil	Mediana	Media	3º cuartil	Max.
2461	6292	7310	7310	7727	16165

Tabla 15.Fuente de ingreso por salario

La distribución se concentra principalmente alrededor de 7.310, con una notable inclinación hacia la derecha. La media es igual a la mediana, lo que indica una frecuencia muy alta en ese valor. Los valores están distribuidos de manera relativamente simétrica alrededor de la mediana, aunque la presencia de valores extremos altos afecta la distribución.

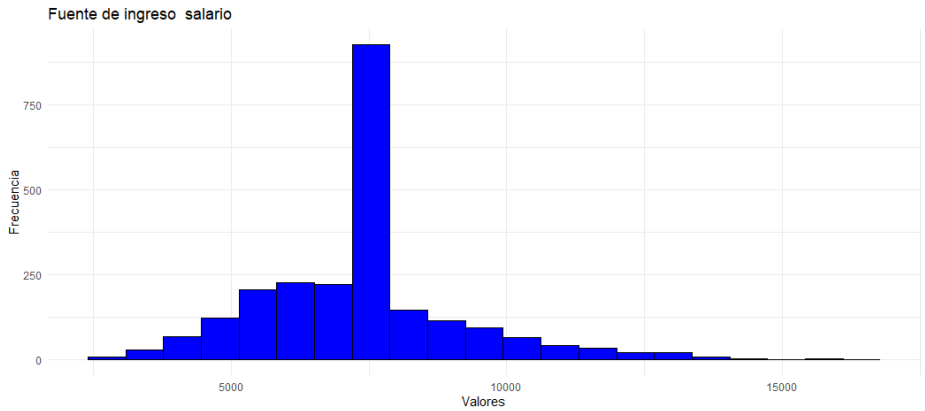

Ilustración 18.Fuente de ingreso por salario

3.2 REDES BAYESIANAS

Para observar las interacciones iniciales entre las variables de nuestro modelo vamos a construir redes bayesianas, observando las relaciones de las variables de diferencia de voto con el resto. Formalmente, las redes bayesianas son grafos dirigidos acíclicos cuyos nodos representan variables aleatorias. Las aristas representan dependencias condicionales; los nodos que no se encuentran conectados representan variables las cuales son condicionalmente independientes de las otras. Esto nos permite interpretar las relaciones de dependencia entre las variables al observar los grafos resultado.

Para construir las redes utilizaremos el paquete “bnlearn” de R y los algoritmos de aprendizaje de estructura basados en restricciones que nos permite utilizar. Estos algoritmos construyen la estructura de una red bayesiana basada en las independencias condicionales entre las variables, las cuales son inferidas directamente de los datos. Tras utilizar algoritmos tanto exactos como basados en heurísticas en el conjunto de datos nos quedamos con los resultados de la búsqueda TABU.

El algoritmo Tabú (Busqueda Tabu, s.f.) es una búsqueda basada en el vecino con memoria que explora el espacio de posibles estructuras de redes bayesianas. Se le llama "Tabu" porque utiliza una lista de movimientos tabú para evitar ciclos y mejorar la exploración del espacio de búsqueda. Los pasos principales del algoritmo son:

1. Inicializa una estructura de red inicial vacía, y una lista tabú que guarda movimientos prohibidos temporalmente para evitar ciclos y estancamientos en óptimos locales.

2. Genera un conjunto de estructuras vecinas mediante movimientos locales en la red actual, como agregar, eliminar o invertir un arco. Evalúa cada estructura vecina con una puntuación
3. Selecciona la estructura vecina con la mejor puntuación que no esté en la lista Tabu o que, si está, mejora la mejor solución encontrada hasta el momento.
4. Actualiza la red actual y añade o remueve los movimientos realizados a la lista Tabú.
5. Repite los tres pasos anteriores hasta llegar al criterio de parada, un número máximo de iteraciones.

Utilizando este algoritmo obtenemos la siguiente red.

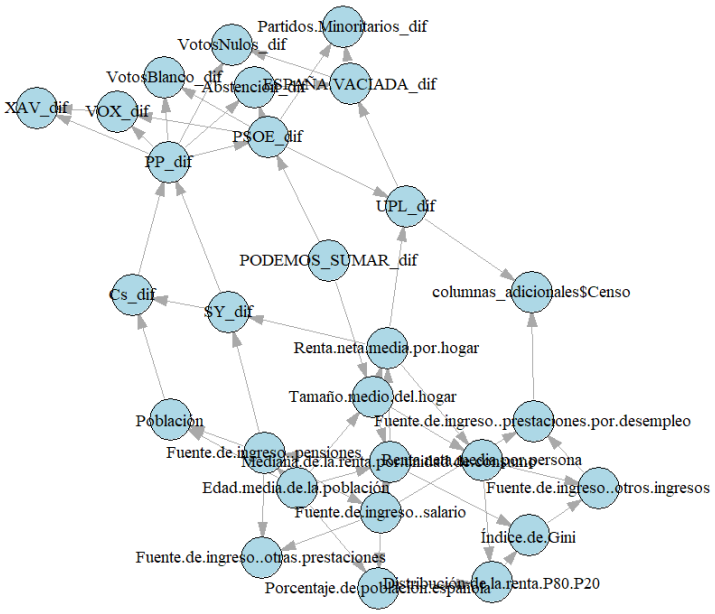

Ilustración 19.Red General Tabu

La red separa nuestras variables principalmente en dos grupos diferenciados, uno con las variables sociodemográficas, en la parte inferior, y otro con la mayoría de las variables de diferencia de votos. Nos fijamos en las variables diferencia que separan ambos grupos, actuando como frontera, pues es donde se ven más claramente las independencias generales. Las relaciones que buscamos son aquellas cuyas aristas partan de variables sociodemográficas y acaben en las variables diferencia, en este caso **UPL_dif, SY_dif y Cs_dif**. Estas variables son casos atípicos, ya sea por tratarse de partidos regionales o por la anomalía en los datos que supone la retirada de ciudadanos de las elecciones generales. En el caso de PODEMOS_SUMAR_dif la relación observada con las variables sociodemográficas es de padre, por lo que no se puede asumir la relación.

Obtenemos los grafos de los tres casos atípicos para visualizar sus relaciones aisladas del resto de variables electorales.

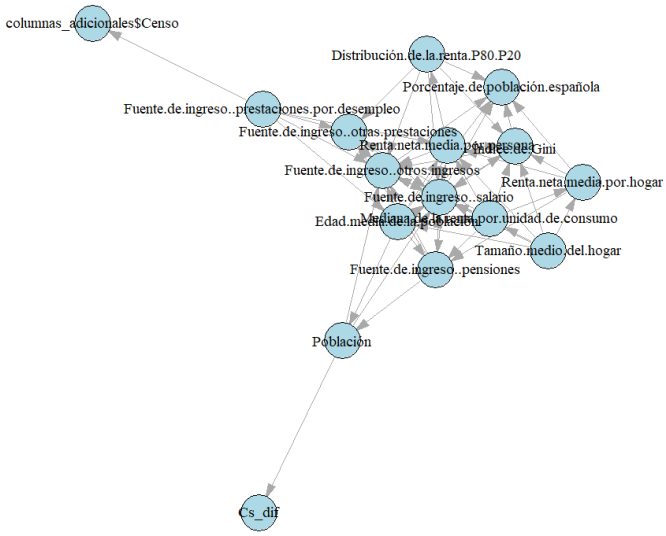

Ilustración 20.Red Cs Tabu

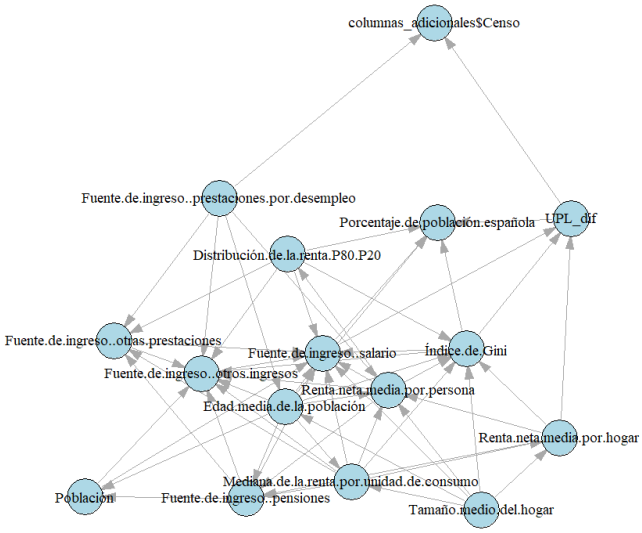

Ilustración 21.Red UPL Tabu

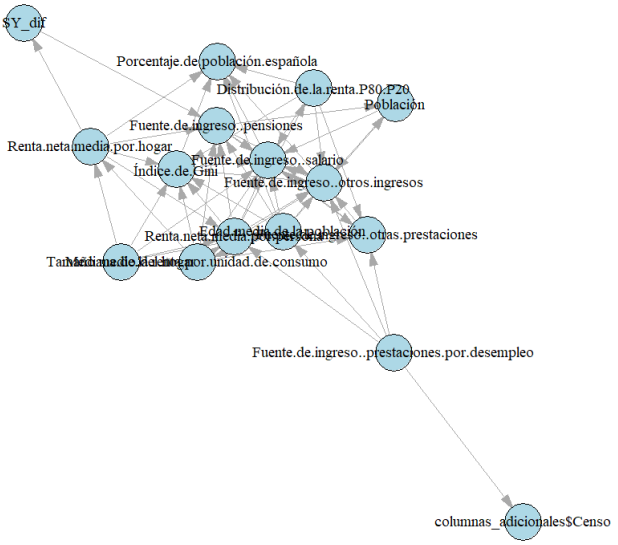

Ilustración 22.Red SY Tabu

La variable de Cs_dif parece ser independiente de las variables sociodemográficas condicionada al valor de la variable Población. Tanto SY_dif como UPL_dif están mucho más integradas en la red de variables, especialmente al tener aristas entrando y saliendo de sus nodos, por lo que no se pueden sacar conclusiones claras sobre su independencia con las variables sociodemográficas, solo que están mucho más relacionadas que el resto de los partidos. Esto puede deberse a su carácter regional, lo que disminuye la participación de voto en estos partidos cuando se contempla el impacto del voto en unas elecciones generales. La gran correlación de estas variables con las covariables explicaría la detección de la diferencia de dichas regiones con el resto de Castilla y León. Llamaremos a estas variables electorales con interacción **'variables frontera'** cuando este resultado sea de interés más adelante.

Hay que destacar además uno de los resultados obtenidos al probar algoritmos exactos en los datos completos, el obtenido al utilizar Fast Incremental Association (Fast-IAMB) (Saratha Sathasivam, 2020) limitado a 5 conexiones por nodo.

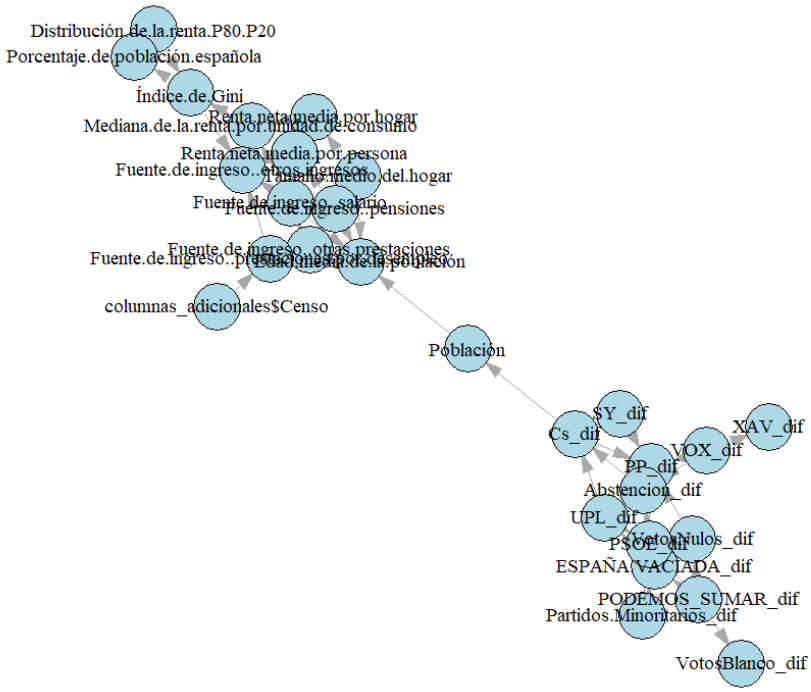

Ilustración 23.Red Global Fast-IAMB

El resultado obtenido sugiere que, **tenida en cuenta la variable Población, el resto de las variables sociodemográficas son independientes de las variables de diferencia de votos**. La separación clara en dos grupos y la dirección de las aristas dejan muy clara esta independencia condicional.

3.3 ANÁLISIS DE COMPONENTES PRINCIPALES

El primer paso por seguir será realizar un análisis de componentes principales, con el objetivo de identificar gráficamente algunas de las relaciones buscadas. Esta técnica pertenece al aprendizaje no supervisado, ya que se desea explorar la posible existencia de subgrupos entre las variables u observaciones sin generar predicciones. Su propósito es reducir el número de variables o dimensiones mientras se conserva la mayor cantidad posible de información, medida como el porcentaje de varianza explicada. Cada componente principal representará una combinación lineal de las variables originales, siendo todas independientes entre sí, y se expresará de la siguiente manera:

$$Componente_p = Coeficiente_{1p}\ Variable_1 + Coeficiente_{2p}\ Variable_2 + \ldots + Coeficiente_{pp}\ Variable_p$$

donde ***p*** es el número de la variable.

En nuestro caso realizaremos dos análisis debido a la problemática de Ciudadanos, comprobando si la inclusión de Ciudadanos como partido minoritario, o el tratamiento del mismo en las elecciones generales como un partido con cero votos supone una diferencia significativa en la composición de las componentes principales.

Por lo tanto, en este caso específico, se obtendrán 27 y 28 componentes principales, cada una formada por una combinación lineal de las 27 y 28 variables con las que se va a trabajar respectivamente.

1. Censo
2. Diferencia porcentual de Votos Nulos
3. Diferencia porcentual de Votos en Blanco
4. Diferencia porcentual de Cs (solo primer análisis)
5. Diferencia porcentual de ESPAÑA VACIADA
6. Diferencia porcentual de PODEMOS/SUMAR
7. Diferencia porcentual del PP
8. Diferencia porcentual del PSOE
9. Diferencia porcentual de Soria Ya
10. Diferencia porcentual de UPL
11. Diferencia porcentual de VOX
12. Diferencia porcentual de XAV
13. Diferencia porcentual en Abstención
14. Diferencia porcentual de Partidos Minoritarios
15. Medida de desigualdad de distribución de la renta P80/P20
16. Índice de Gini
17. Edad media de la población
18. Población total
19. Porcentaje de población española
20. Tamaño medio del hogar
21. Mediana de la renta por unidad de consumo
22. Renta neta media por hogar
23. Renta neta media por persona
24. Ingresos por ‘otras prestaciones’
25. Ingresos por ‘otros ingresos’
26. Ingresos por ‘prestaciones por desempleo’
27. Ingresos por ‘pensiones’

28. Ingresos por ‘salario’

La definición de las variables electorales ayudará a identificar que cambios han sido los más marcados entre las elecciones a través del peso de dicha diferencia en la composición de las componentes principales, y su relación con los pesos de otras variables de la composición.

En el primer análisis tomamos la diferencia de votos de Ciudadanos como una variable de peso, y observamos los resultados de la siguiente tabla. Esto lo hacemos computando los votos de Ciudadanos en las elecciones generales con un valor de 0, pues no se presentaron. En este caso, los autovalores del análisis serán 28, puesto que se trabaja con 28 variables numéricas, y por lo tanto con 28 componentes principales. Además, al tratarse de un análisis normado, debido a las diferencias en las respectivas escalas, la suma de todos ellos será 28:

	Autovalor	%Varianza	%Varianza Acumulada
1	3.94	14.08	14.08
2	2.68	9.56	23.65
3	2.10	7.51	31.15
4	1.86	6.64	37.79
5	1.50	5.35	43.15
6	1.43	5.11	48.26
7	1.30	4.64	52.90
8	1.23	4.39	57.29
9	1.16	4.15	61.44
10	1.04	3.71	65.15
11	1.02	3.65	68.80
12	1.00	3.58	72.38
13	0.92	3.29	75.67
14	0.89	3.18	78.85
15	0.86	3.06	81.91
16	0.76	2.73	84.64
17	0.73	2.61	87.25
18	0.69	2.47	89.71
19	0.63	2.25	91.97
20	0.57	2.05	94.01
21	0.55	1.98	95.99
22	0.36	1.29	97.27
23	0.27	0.98	98.25
24	0.20	0.73	98.98
25	0.20	0.72	99.70
26	0.06	0.20	99.90
27	0.03	0.10	100.00
28	0.00	0.00	100.00

Tabla 16.Varianza Explicada Opción1

Observando la tabla superior, se puede ver claramente que la primera componente es la que presenta mayor autovalor y porcentaje de varianza explicada (14.08%). Podrían seleccionarse hasta 12 componentes para superar el 70%, para alcanzar el 72.38%. Sin embargo, el porcentaje de varianza explicada que aporta cada componente es muy bajo

en general y disminuye rápidamente. Elegimos quedarnos con las seis primeras componentes, que en total representan prácticamente la mitad de la variabilidad (48.26%). Escoger más para aspirar a un porcentaje mayor se aleja de nuestro objetivo de disminuir la dimensionalidad, por lo que habría que seleccionar demasiadas dimensiones para superar una mayoría como el 80%.

Los porcentajes bajos se deben principalmente a la baja correlación entre covariables que ha causado nuestra criba anterior, ya que, como vimos en las redes bayesianas, las variables electorales no parecen tener vínculos estrechos con ellas a priori.

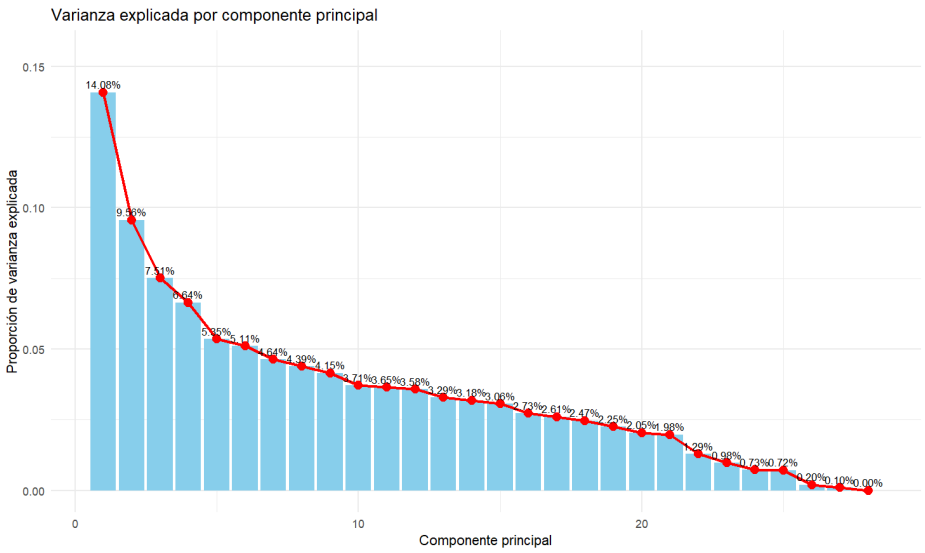

Ilustración 24.Varianza Explicada

Analizando el gráfico superior de autovalores se llega a la misma conclusión. Parece que 6 componentes son suficientes, puesto que es el número en el que la representación empieza a descender casi linealmente, aunque, con esta elección, únicamente se obtendría un 48.26% de varianza explicada. Además, la selección de 6 dimensiones sigue siendo una mejora en cuanto a la reducción de dimensionalidad, pues partimos de 27 variables.

Mediante el autovector correspondiente a la primera componente, que determina los coeficientes, se observa que las mayores contribuciones a esta son de variables económicas y demográficas. PC1 parece estar fuertemente influenciada por **la renta y la fuente de ingreso (especialmente el salario), así como por la edad media de la población**.

PC1 = 0.1426Censo + 0.0094VotosNulos_dif + 0.0025VotosBlanco_dif - 0.1324Cs_dif - 0.1547ESPAÑA.VACIADA_dif + 0.0995PODEMOS_SUMAR_dif + 0.1027PP_dif - 0.2536PSOE_dif - 0.0489SY_dif + 0.1110UPL_dif + 0.0151VOX_dif + 0.0448XAV_dif + 0.2010Abstencion_dif + 0.0798Partidos.Minoritarios_dif + 0.0995Distribución.de.la.renta.P80.P20 + 0.0702Índice.de.Gini - 0.2384Edad.media.de.la.población + 0.1437Población - 0.0478Porcentaje.de.población.española + 0.1982Tamaño.medio.del.hogar + 0.3784Mediana.de.la.renta.por.unidad.de.consumo + 0.4089Renta.neta.media.por.hogar + 0.3046Renta.neta.media.por.persona - 0.1764Fuente.de.ingreso..otras.prestaciones + 0.1729Fuente.de.ingreso..otros.ingresos - 0.1187Fuente.de.ingreso..pensiones + 0.0587Fuente.de.ingreso..prestaciones.por.desempleo + 0.4171Fuente.de.ingreso..salario

Para la segunda componente, se encuentran signos tanto positivos como negativos en los coeficientes, pero en general parece estar capturando variaciones en la participación electoral y la estructura de la población. Se destacan en su composición las variables **Censo, Población y Tamaño del hogar** con coeficientes negativos, y la **edad media de la población, renta por persona e ingresos** en positivo.

PC2 = -0.3434Censo + 0.0244VotosNulos_dif + 0.0026VotosBlanco_dif + 0.0298Cs_dif - 0.1012ESPAÑA.VACIADA_dif - 0.0648PODEMOS_SUMAR_dif + 0.1103PP_dif - 0.0852PSOE_dif - 0.1210SY_dif + 0.0753UPL_dif - 0.0821VOX_dif + 0.0049XAV_dif + 0.0900Abstencion_dif + 0.0837Partidos.Minoritarios_dif + 0.1422Distribución.de.la.renta.P80.P20 + 0.1298Índice.de.Gini + 0.4164Edad.media.de.la.población - 0.3375Población + 0.1076Porcentaje.de.población.española - 0.2988Tamaño.medio.del.hogar + 0.1433Mediana.de.la.renta.por.unidad.de.consumo + 0.0905Renta.neta.media.por.hogar + 0.3405Renta.neta.media.por.persona + 0.0576Fuente.de.ingreso..otras.prestaciones + 0.3230Fuente.de.ingreso..otros.ingresos + 0.3072Fuente.de.ingreso..pensiones - 0.1619Fuente.de.ingreso..prestaciones.por.desempleo - 0.0798Fuente.de.ingreso..salario

La tercera componente parece estar influenciada **principalmente por algunas diferencias de voto como UPL, PSOE y Abstencion,** aunque también recoge aspectos sociodemográficos como los **ingresos por pensiones, la población y la mediana de la renta por unidad de consumo**.

PC3 = 0.2764Censo - 0.0444VotosNulos_dif - 0.0088VotosBlanco_dif + 0.1012Cs_dif + 0.1157ESPAÑA.VACIADA_dif - 0.0910PODEMOS_SUMAR_dif - 0.0005PP_dif + 0.2981PSOE_dif - 0.1189SY_dif - 0.3560UPL_dif - 0.0348VOX_dif + 0.0653XAV_dif - 0.1822Abstencion_dif - 0.0598Partidos.Minoritarios_dif - 0.2300Distribución.de.la.renta.P80.P20 - 0.2526Índice.de.Gini + 0.0999Edad.media.de.la.población + 0.2695Población + 0.1477Porcentaje.de.población.española - 0.0377Tamaño.medio.del.hogar + 0.2913Mediana.de.la.renta.por.unidad.de.consumo + 0.1917Renta.neta.media.por.hogar + 0.2303Renta.neta.media.por.persona + 0.2861Fuente.de.ingreso..otras.prestaciones - 0.0652Fuente.de.ingreso..otros.ingresos + 0.3506Fuente.de.ingreso..pensiones - 0.0571Fuente.de.ingreso..prestaciones.por.desempleo + 0.0908Fuente.de.ingreso..salario

La cuarta componente parece estar influenciada **principalmente por algunas diferencias de voto como UPL, y la desigualdad económica (Índice de Gini y distribución de la renta 80/20)**.

PC4 = -0.3203Censo - 0.0244VotosNulos_dif + 0.0244VotosBlanco_dif - 0.0586Cs_dif - 0.0860ESPAÑA.VACIADA_dif + 0.0381PODEMOS_SUMAR_dif + 0.0161PP_dif - 0.1153PSOE_dif - 0.0274SY_dif + 0.0944UPL_dif + 0.0187VOX_dif + 0.0204XAV_dif + 0.1062Abstencion_dif + 0.0492Partidos.Minoritarios_dif - 0.5463Distribución.de.la.renta.P80.P20 - 0.5741Índice.de.Gini - 0.0146Edad.media.de.la.población - 0.3185Población + 0.2129Porcentaje.de.población.española + 0.1041Tamaño.medio.del.hogar + 0.0829Mediana.de.la.renta.por.unidad.de.consumo + 0.0498Renta.neta.media.por.hogar - 0.0033Renta.neta.media.por.persona - 0.0807Fuente.de.ingreso..otras.prestaciones - 0.1579Fuente.de.ingreso..otros.ingresos - 0.1126Fuente.de.ingreso..pensiones + 0.0601Fuente.de.ingreso..prestaciones.por.desempleo + 0.0568Fuente.de.ingreso..salario

La quinta componente está relacionada con **las diferencias electorales en general,** además de con **el censo, la población, la edad media y el tamaño del hogar**, sugiriendo

que esta componente puede estar capturando factores sociodemográficos que afectan a las diferencias de voto en los distritos municipales.

PC5 = 0.2888Censo + 0.1919VotosNulos_dif - 0.0329VotosBlanco_dif + 0.0081Cs_dif - 0.3088ESPAÑA.VACIADA_dif + 0.0983PODEMOS_SUMAR_dif - 0.2898PP_dif - 0.2840PSOE_dif + 0.1604SY_dif + 0.1745UPL_dif + 0.1576VOX_dif - 0.0421XAV_dif + 0.3293Abstencion_dif + 0.2749Partidos.Minoritarios_dif - 0.0884Distribución.de.la.renta.P80.P20 - 0.0997Índice.de.Gini + 0.1798Edad.media.de.la.población + 0.2980Población + 0.0706Porcentaje.de.población.española - 0.3006Tamaño.medio.del.hogar - 0.0455Mediana.de.la.renta.por.unidad.de.consumo - 0.1499Renta.neta.media.por.hogar + 0.0297Renta.neta.media.por.persona + 0.0779Fuente.de.ingreso..otras.prestaciones - 0.1655Fuente.de.ingreso..otros.ingresos + 0.1985Fuente.de.ingreso..pensiones + 0.0632Fuente.de.ingreso..prestaciones.por.desempleo - 0.0751Fuente.de.ingreso..salario

La sexta componente parece estar también influenciada por diferencias en el voto, especialmente a los partidos políticos del espacio electoral de derechas (**PP y VOX**).

PC6 = -0.2234Censo + 0.2566VotosNulos_dif + 0.0557VotosBlanco_dif + 0.0841Cs_dif - 0.0877ESPAÑA.VACIADA_dif - 0.0771PODEMOS_SUMAR_dif - 0.5625PP_dif + 0.1913PSOE_dif + 0.0103SY_dif - 0.2231UPL_dif + 0.4758VOX_dif + 0.1923XAV_dif - 0.0368Abstencion_dif + 0.1259Partidos.Minoritarios_dif + 0.0655Distribución.de.la.renta.P80.P20 + 0.0702Índice.de.Gini - 0.1151Edad.media.de.la.población - 0.2308Población - 0.2294Porcentaje.de.población.española + 0.1128Tamaño.medio.del.hogar + 0.0804Mediana.de.la.renta.por.unidad.de.consumo + 0.1010Renta.neta.media.por.hogar + 0.0561Renta.neta.media.por.persona + 0.0732Fuente.de.ingreso..otras.prestaciones + 0.0202Fuente.de.ingreso..otros.ingresos + 0.0156Fuente.de.ingreso..pensiones + 0.0773Fuente.de.ingreso..prestaciones.por.desempleo + 0.0638Fuente.de.ingreso..salario

Calculamos las contribuciones, correlaciones y representacion de las variables originales con estas componentes principales elegidas.

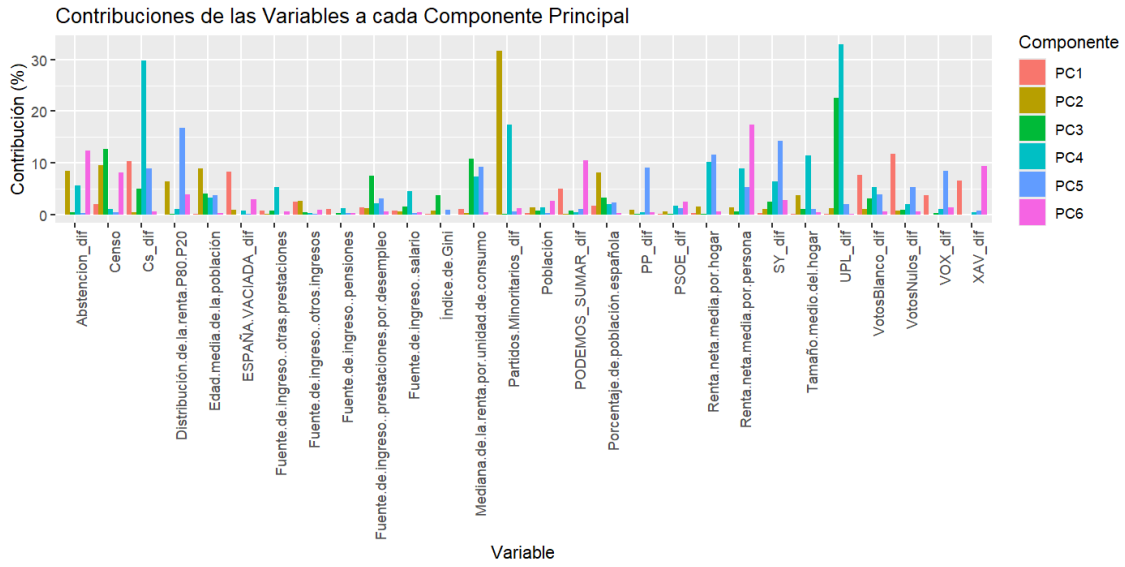

Ilustración 25.Contribuciones PCA

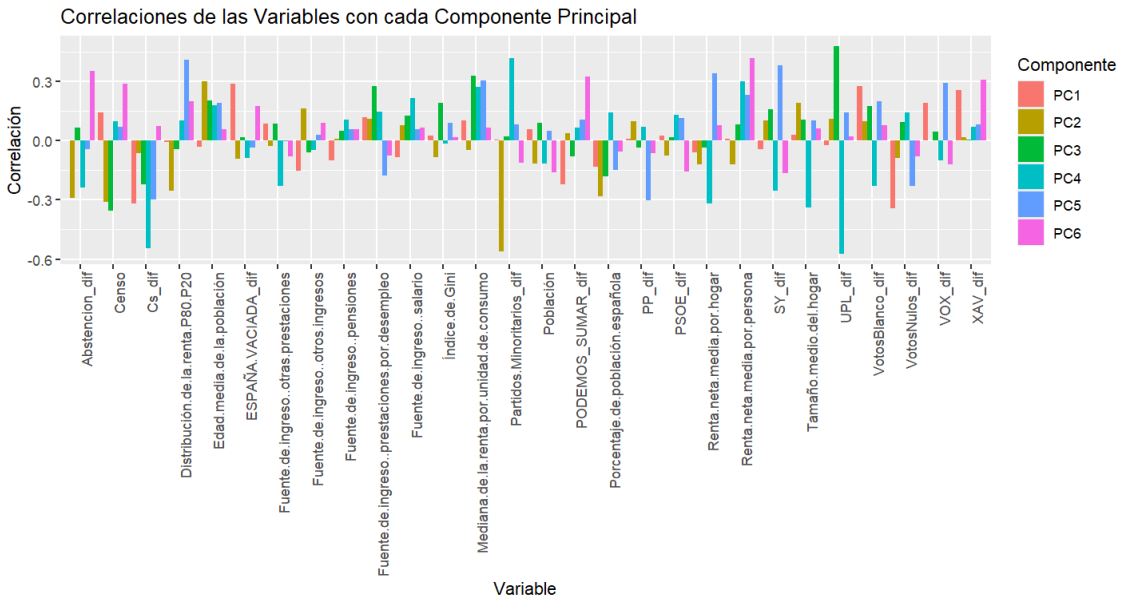

Ilustración 26.Correlaciones PCA

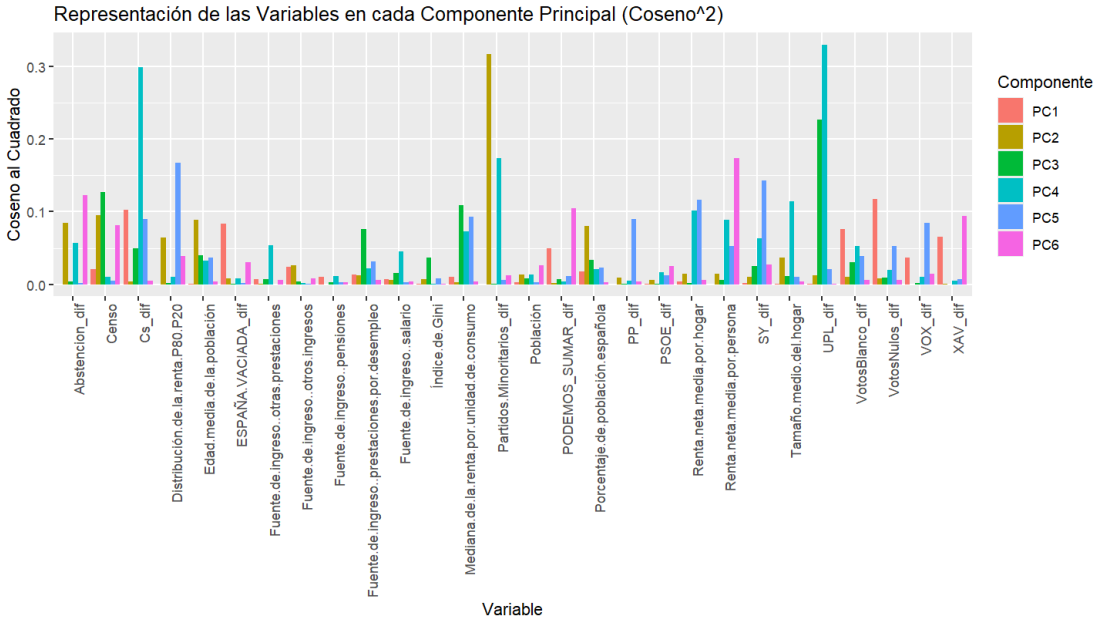

Ilustración 27.Representaciones PCA

En cuanto a las contribuciones, observamos que:

- La mayor contribución es la de la variable **Partidos_Minoritarios_dif** para con la segunda componente.
- A la primera componente contribuyen especialmente algunas variables de diferencia de votos como la **abstención**, los **votos nulos** y los votos de **Cs**.
- A la cuarta componente contribuyen principalmente **UPL_dif y Cs_dif**. Ambas observadas como variables frontera en las redes bayesianas.
- A la tercera componente está formada por **UPL_dif** y algunas covariables.

Las correlaciones más destacadas ya las hemos interpretado al estudiar la composición de cada una de las 6 componentes principales.

Por otro lado, los cosenos cuadrados informan de la calidad de la representación de cada variable en la dimensión respectivamente, en ellos se observa:

- Las variable **Partidos_minoritarios_dif** tiene una representación muy alta en la segunda componente principal (PC2), como se muestra por la barra amarilla alta. Esto sugiere que esta variable es clave para PC2.
- **UPL_dif** es la mejor representada en PC3 con diferencia.
- **UPL_dif, Cs_dif y Partidos_Minoritarios_dif** son las mejor representadas en la cuarta componente, todos ellos partidos con participaciones bajas**.**
- La medida de desigualdad **Distribución.de.la.renta.P80.P20** tiene alta representación en la quinta componente principal al igual que otras variables económicas como las **rentas netas** y la **mediana de la renta por unidad de consumo**. El hecho de que **PP_dif, VOX_dif y SY_dif** también estén bien representadas puede estar indicando que estas componentes capturan información importante relacionada con el impacto de las covariables en estas diferencias de voto.
- La sexta componente principal representa la **renta neta media por persona** y algunas diferencias de voto.

Como conclusión de este **análisis de componentes principales**, observamos que las primeras seis componentes principales representan un porcentaje limitado, pero aceptable, de la varianza total, lo que justifica su selección. Las características de estas componentes conllevan varias conclusiones sobre ellas:

1. **Variables Económicas y Demográficas**: Las variables económicas, como la renta neta media por hogar y por persona, y las fuentes de ingreso (especialmente el salario), tienen una fuerte influencia en las primeras componentes principales. Esto también sucede con variables demográficas, como la edad media de la población, la población total y el tamaño medio del hogar, indicando que las diferencias económicas y demográficas juegan un papel crucial en la variabilidad de los datos.
2. **Interacciones complejas**: Las componentes principales muestran que las interacciones entre variables económicas, demográficas y electorales son complejas, pero existen. Por ejemplo, la renta y la fuente de ingreso no solo están relacionadas con factores económicos directos, sino que también influyen en el comportamiento electoral y en la estructura poblacional.
3. **Preferencias Políticas**: PC6 está dominada por las diferencias en el apoyo a los partidos políticos, especialmente PP y VOX.

En cuanto al **segundo análisis** de componentes principales, el que opta por incluir a Ciudadanos entre los partidos minoritarios en lugar de incluir su diferencia como una nueva variable, apoya la decisión de elegir seis componentes principales, pues aumenta el porcentaje de varianza que explican hasta el 49.70%.

	Autovalor	%Varianza	%Varianza Acumulada
1	3.88	14.36	14.36
2	2.68	9.91	24.28
3	2.09	7.73	32.01
4	1.85	6.87	38.88

5	1.49	5.51	44.39
6	1.43	5.31	49.70

Tabla 17.Varianza Explicada Opción2

Aun así, la contribución de la diferencia porcentual de voto de los partidos minoritarios disminuye al incluir a Ciudadanos en su grupo, por lo que optamos por seguir con los ajustes del primer análisis. Resultados de Partidos.Minoritarios_dif en el segundo análisis:

Partidos.Minoritarios_dif	PC1	PC2	PC3	PC4	PC5	PC6
Contributions	1.0717	2.0694	0.8781	0.4181	5.4971	4.4314
Correlations	0.0866	-0.1165	-0.0736	0.0193	-0.4872	-0.2341
Cos2	0.0075	0.0136	0.0054	0.0004	0.2374	0.0548

Un aumento tangencial en el porcentaje de varianza representada no justifica la perdida de importancia de la diferencia de voto en dichos partidos.

Mediante el análisis de componentes principales centrado en las seis primeras dimensiones no se obtienen las relaciones directas claras buscadas entre variables sociodemográficas y diferencias electorales. Para comprobar si existen diferencias pasadas por alto hasta ahora, se generarán gráficos que muestren las coordenadas de las variables en pares de componentes.

Al combinar las seis componentes obtenemos un total de quince gráficos de coordenadas para pares de componentes.

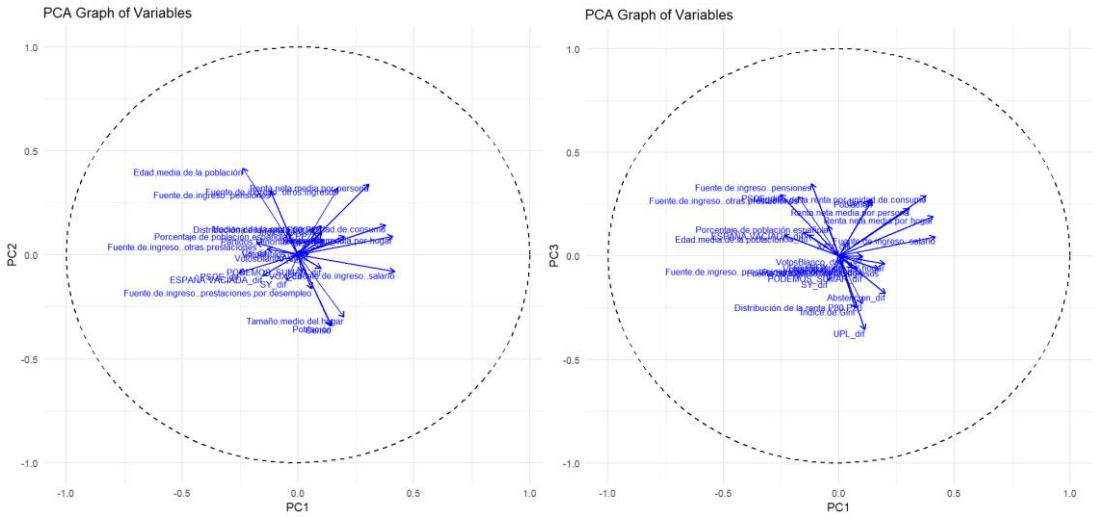

Ilustración 28.Gráficos de coordenadas 1 y 2

En el gráfico que compara PC1 con PC2 se observa que hay una alta densidad de variables cerca del centro, lo que sugiere que muchas variables no contribuyen significativamente a la varianza explicada por los primeros dos componentes principales, esto es común en los gráficos obtenidos. Además, las variables **Población** y **Censo** están cercanas, lo que sugiere que tienen una relación similar con PC1 y PC2. **Edad media de la población y Tamaño medio del hogar** tienen relación con ambas componentes, pues están en la diagonal del gráfico.

En cuanto al grafico que compara PC1 y PC3 se puede destacar la diferencia de representación entre **Ingresos por salario y Renta neta media por hogar**, más inclinado hacia PC1, y **UPL_dif y Ingresos por pensiones** hacia PC3.

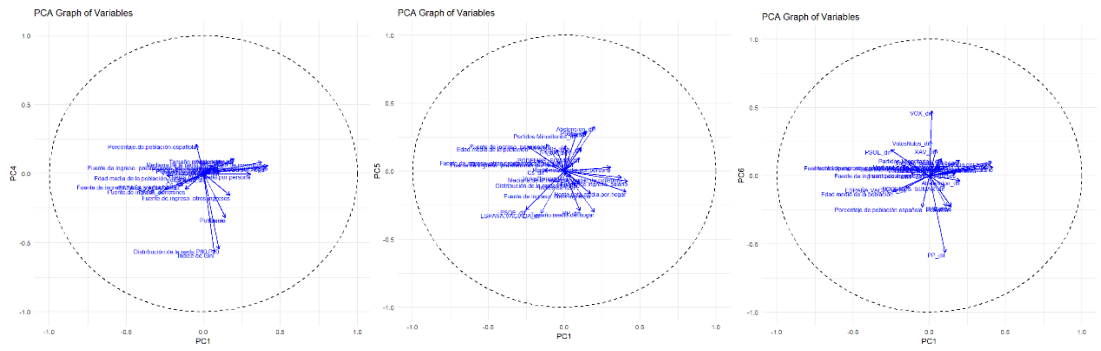

Ilustración 29.Gráficos de coordenadas 3, 4 y 5

En el primer grafico se observa que **Distribucion de la renta P80/P20** y **Índice de Gini** están más cercanas y pueden estar relacionadas. Se confirma además la clara correlación de las variables de desigualdad con la cuarta componente principal, con sus autovectores negativos, claramente más grandes que el resto del gráfico.

El grafico comparación de PC5 con PC1 no deja en claro ninguna relación entre las variables más allá de que la mayoría están correlacionadas en menor medida en el mismo.

En cuanto a las dimensiones PC6 y PC1, destaca claramente la diferencia en la relación de las variables. La gran mayoría están mucho mejor correlacionadas con PC1, y solo las variables de diferencia de voto de los partidos de derechas, **VOX_dif** y **PP_dif**, muestran una gran correlación con la sexta componente principal, lo que las convierte en un grupo diferenciado del resto.

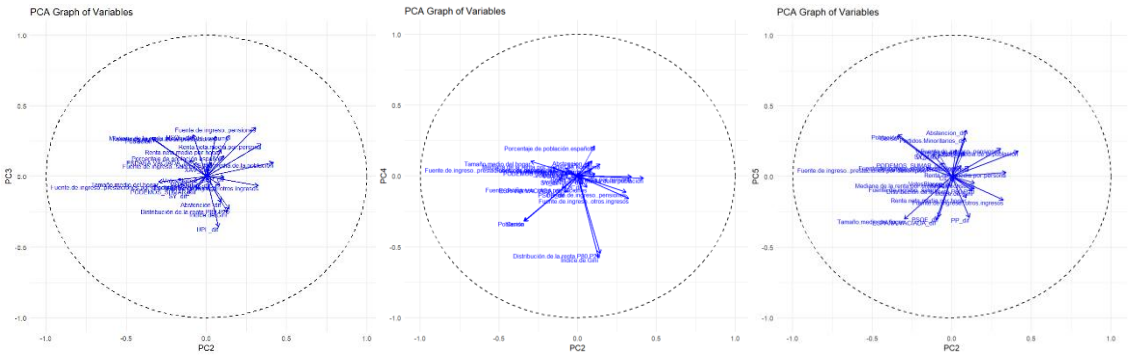

Ilustración 30.Gráficos de coordenadas 6, 7 y 8

En este caso en estos gráficos se repiten observaciones ya comentadas como la importancia de **UPL_dif** en PC3, la cercanía de los autovectores de las variables de desigualdad (**Índice de Guini** y **Distribucion P80/P20**) y su importancia en PC4, y de las variables de recuento poblacional (**Población** y **Censo**). Cabe destacar que el autovector de estas últimas es diagonal en el gráfico que compara PC4 y PC2, lo que habla de su representación en ambas componentes. Estas variables de recuento poblacional parecen ser importantes a través de todas las componentes.

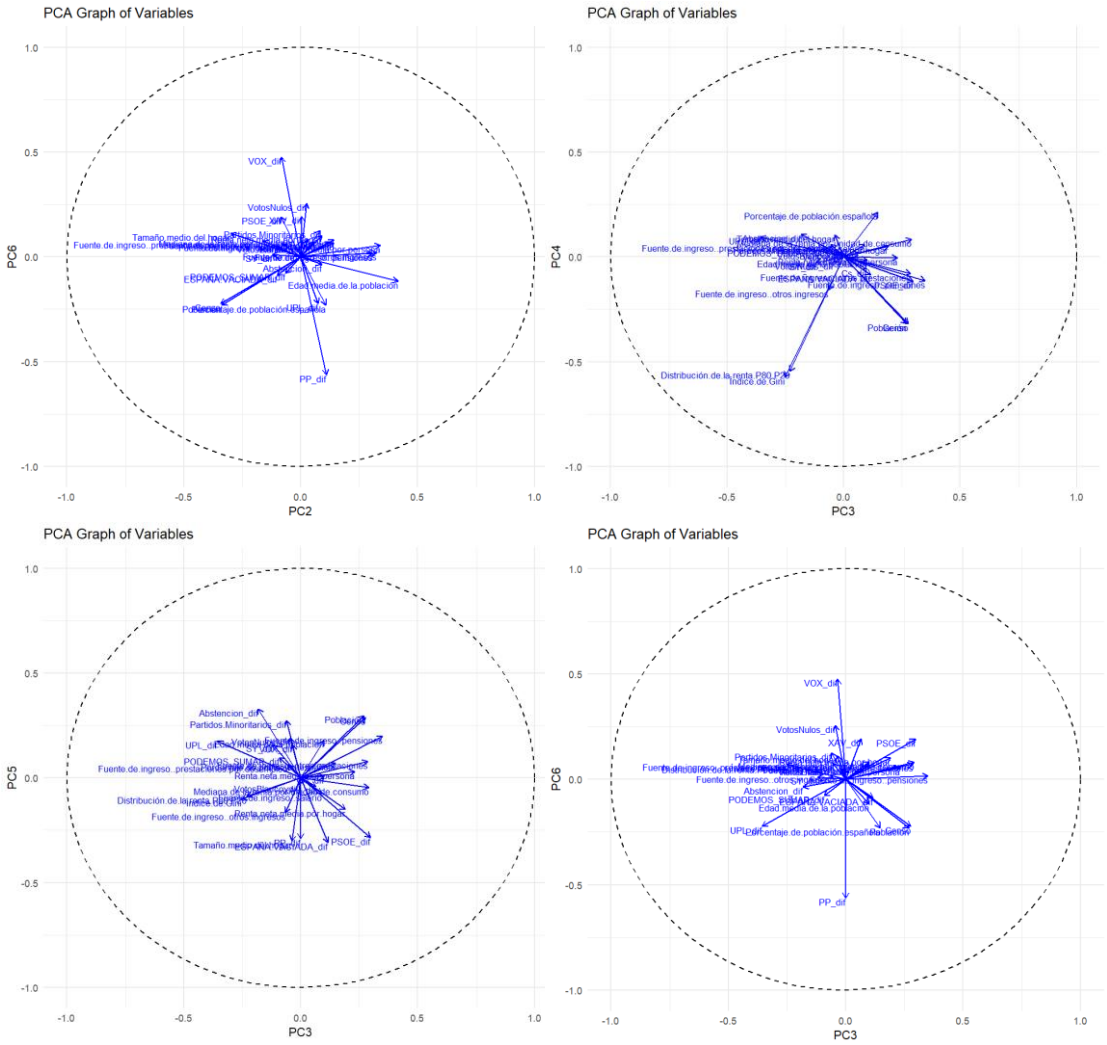

Ilustración 31.Gráficos de coordenadas 9, 10, 11 y 12

En estos gráficos se observan las mismas inclinaciones en las correlaciones referentes a la diferencia de voto en los partidos de derecha en PC6, y la influencia de las variables de desigualdad en PC4. Vuelven a estar presentes las variables de recuento poblacional con valores próximos a 0.25.

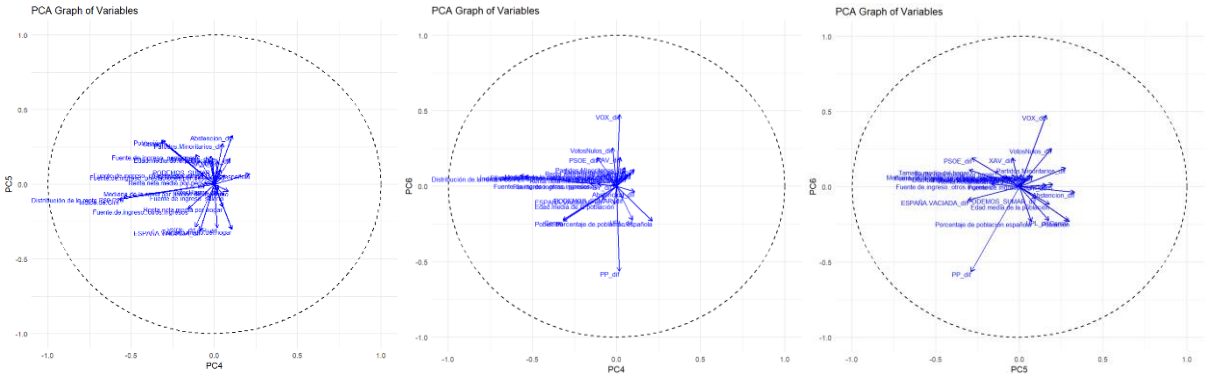

Ilustración 32.Gráficos de coordenadas 13, 14 y 15

En el grafico que compara las componentes cuarta y quinta vemos que las variables de **desigualdad económica** no están relacionadas con PC5, lo que apoya que su relación principal es con PC4. Más allá de esto tanto **Población** como **Censo** parecen tener correlación conjunta con ambas dimensiones.

En cuanto a las correlaciones en PC6 con PC4, se observa que las variables que dominan PC6, **VOX_dif** y **PP_dif**, no se correlacionan con la componente cuatro, como parece que si lo hacen ligeramente con la cinco en el gráfico siguiente. Además, la diferencia de votos de **UPL** parece estar levemente correlacionada con ambos. El resto de las variables o no son significativas, con correlaciones claramente menores a 0.25, o pertenecen al grupo que se correlaciona a la cuarta componente con valores negativos.

En cuanto al último gráfico, se observa una mayor correlación de las variables de diferencia de voto con la componente 5 en comparación a la 4 del gráfico anterior. Esto se nota claramente en la inclinación de los autovectores de **VOX_dif** y **PP_dif**, como ya se ha comentado, además de tener correlaciones con variables de diferencia de votos en otros partidos con valores positivos, en comparación con los valores negativos de la cuarta componente, como es el caso de **UPL, Abstención, Partidos Minoritarios y Votos Nulos**.

3.4 ANÁLISIS CLÚSTER

Se realizará un análisis clúster (Kassambara, 2017) sobre los distritos municipales. El objetivo es determinar si las características de los grupos obtenidos permiten deducir alguna relación entre las variables sociodemográficas y electorales.

Esta técnica implica un aprendizaje no supervisado, ya que los grupos iniciales a los que pertenece cada individuo son desconocidos, y se busca crear subconjuntos similares entre sí en relación con una determinada medida de similitud entre los distritos municipales. Por lo tanto, su propósito es obtener particiones de los datos de tal manera que los distritos municipales dentro de cada una sean lo más parecidos posibles entre sí y que difieran lo máximo posible de los distritos municipales de otras particiones.

Se emplearán dos métodos de clúster diferentes para posteriormente comparar sus resultados: el método k-medoids y el método jerárquico, estandarizando nuevamente las variables para que todas tengan una importancia equivalente en el método utilizado y no dependan de la escala en la que están representadas.

Método k-medoids

Este método establece K particiones de los datos que son distintas y no se solapan, aunque es necesario seleccionar el número de grupos deseado antes de aplicar el algoritmo. Denotando n como el número de individuos disponibles y K como el número de grupos que se desean formar, el algoritmo k-medoids proporcionará un óptimo local de entre las K^n formas posibles de agrupar todas las observaciones:

- En primer lugar, se asigna aleatoriamente uno de los K grupos buscados a cada individuo.
- Se repiten los siguientes dos pasos hasta que el resultado no cambie, indicando que se ha alcanzado el óptimo buscado:
    1. Se selecciona un objeto representativo (medoide) para cada grupo, que minimiza la distancia total a los otros objetos del mismo clúster.
    2. Se asigna cada individuo al grupo cuyo medoide tenga más cerca.

A diferencia de un centroide, que es una media de todas las características de los puntos en un clúster (como en k-means), el medoide es un punto real del conjunto de datos que tiene la menor distancia total a todos los demás puntos del clúster.

El primer paso por realizar es estimar el número óptimo de clústeres. Para esta tarea se han utilizado el método de la estadística gap, utilizando 'cluster::clara' como función de partición. Este método puede automatizarse mediante la función de R "fviz_nbclust" que además de obtener el número de grupos buscado, genera una representación gráfica de los resultados:

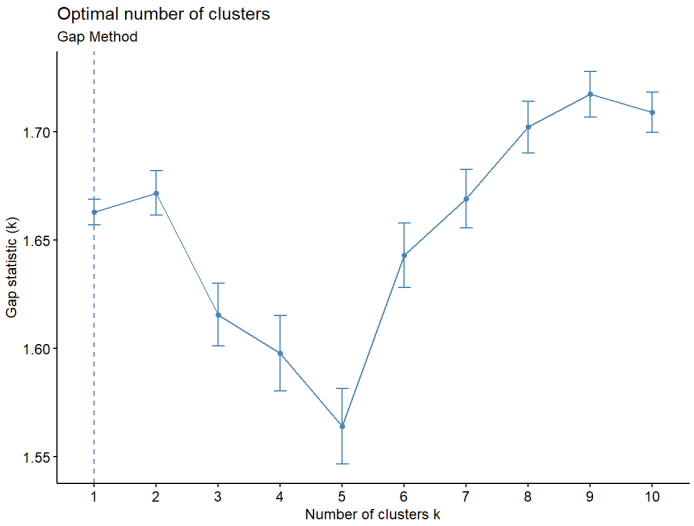

Ilustración 33.Gap Statistic por Número de Clústeres

Como se puede observar en el gráfico superior, el número óptimo de clústeres para los datos es 9 usando el filtro del máximo global, y tras aplicar el algoritmo k-medoids para el número óptimo de grupos se obtiene:

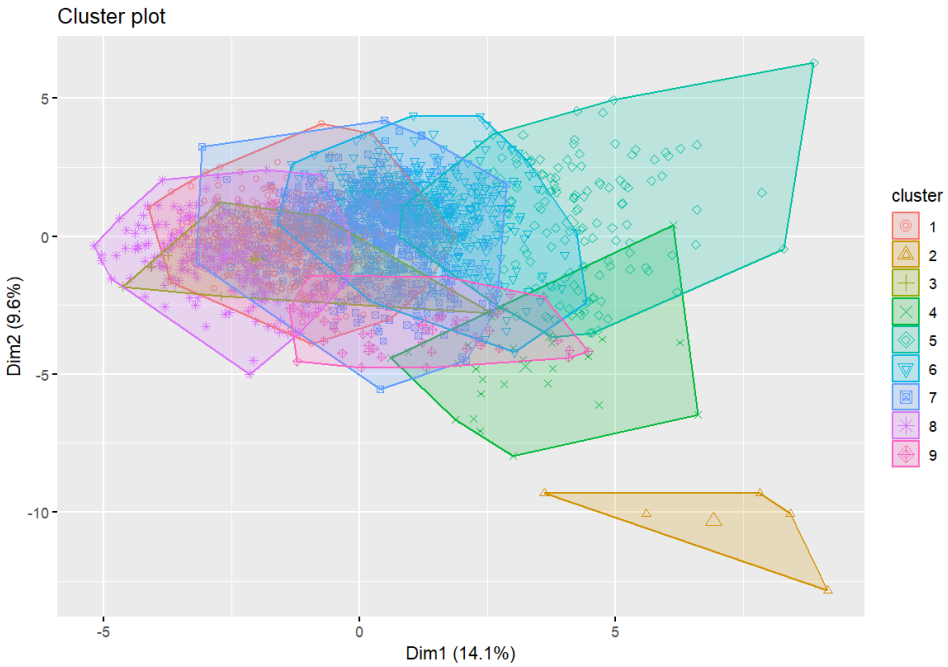

Ilustración 34.Cluster plot

Método jerárquico

Este método ofrece una ventaja sobre k-medoids al no requerir la especificación previa del número de grupos deseados. Los métodos jerárquicos generan representaciones en forma de árbol llamadas dendrogramas, que ilustran la estructura jerárquica entre los distritos municipales. A medida que se asciende en el dendrograma, las ramas se unen, agrupando distritos municipales similares dentro de cada unión. Las uniones en la parte inferior del dendrograma contienen distritos municipales más similares que las ubicadas más arriba. Por lo tanto, las conclusiones derivadas de estos gráficos se centran en la altura a la cual se unen los distritos municipales; esta línea horizontal indica qué distritos municipales pertenecen a cada grupo.

Por otro lado, hay que elegir la medida de similitud, en este caso elegiremos la distancia euclídea:

- Para dos individuos i y j, la distancia euclidiana Dij se define como:

$$Dij = \sqrt{\sum_{k=1}^{n} (Xki - Xkj)^2}$$

  Siendo Xki el valor de la variable xk para el individuo i y Xkj el valor de la variable xk para el individuo j.

Finalmente, hay que seleccionar el algoritmo que medirá la distancia entre clústeres, contemplamos tres posibilidades:

- Complete Linkage: Calcula la distancia entre los elementos de dos grupos diferentes y selecciona la máxima como la distancia entre los grupos.
- Average Linkage: Similar al método anterior, pero la distancia entre grupos es el promedio de todas las distancias entre pares de individuos.
- Single Linkage: Utiliza el mismo enfoque, pero selecciona la distancia mínima entre todas las distancias calculadas.

Una vez elegidos los métodos para medir las distancias entre individuos y entre clústeres, el dendrograma se construirá de manera iterativa. En este proceso, inicialmente cada elemento se considera como un clúster individual y, en cada paso, los individuos más similares se unen gradualmente hasta que todos pertenezcan a un único grupo, completando así el gráfico.

Al aplicar el método, el primer paso ha sido calcular la matriz de distancias euclidianas a partir de la matriz de datos después de estandarizarlos y, posteriormente, aplicar los 3 métodos Linkage explicados para seleccionar aquel que obtenga mejores resultados.

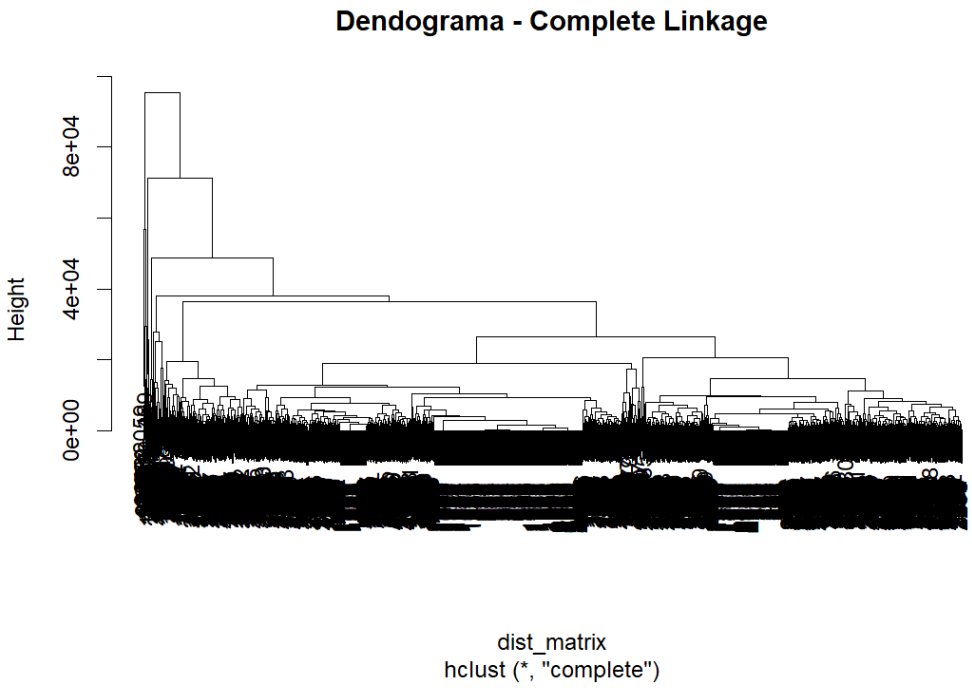

Ilustración 35.Dendograma Complete Linkage

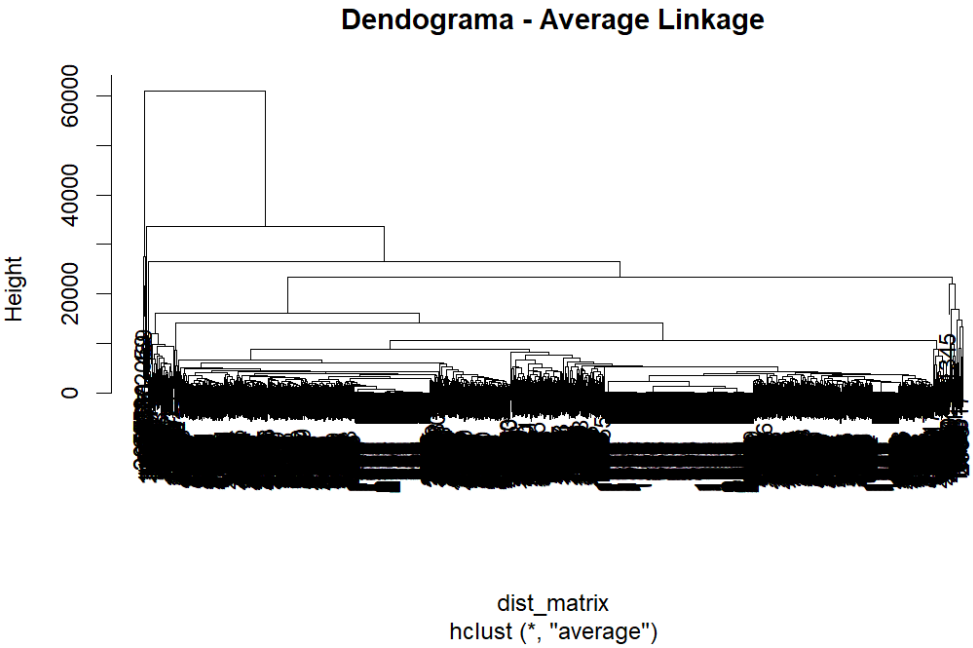

Ilustración 36.Dendograma Average Linkage

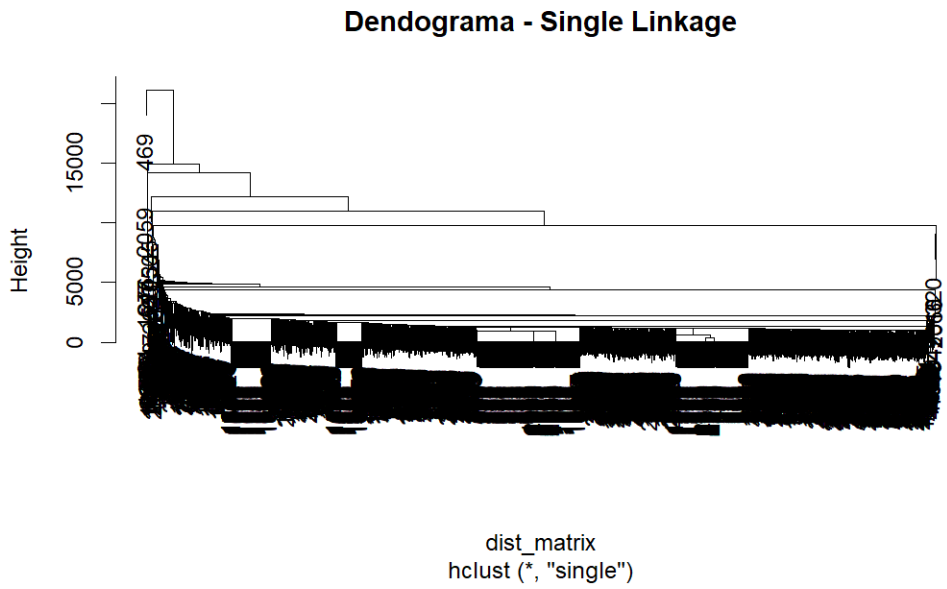

Ilustración 37.Dendograma Single Linkage

Como se observa, el método de Complete Linkage parece mostrar un dendrograma más equilibrado. Por lo tanto, hemos decidido utilizar este método para aplicar el clustering jerárquico a los datos, utilizando la distancia euclidiana para medir las distancias entre los distritos municipales. El corte del dendrograma se establecerá en la altura que genere 9 grupos, que es el número óptimo obtenido para k-medoids. De esta manera, podremos comparar los resultados obtenidos entre ambos métodos.

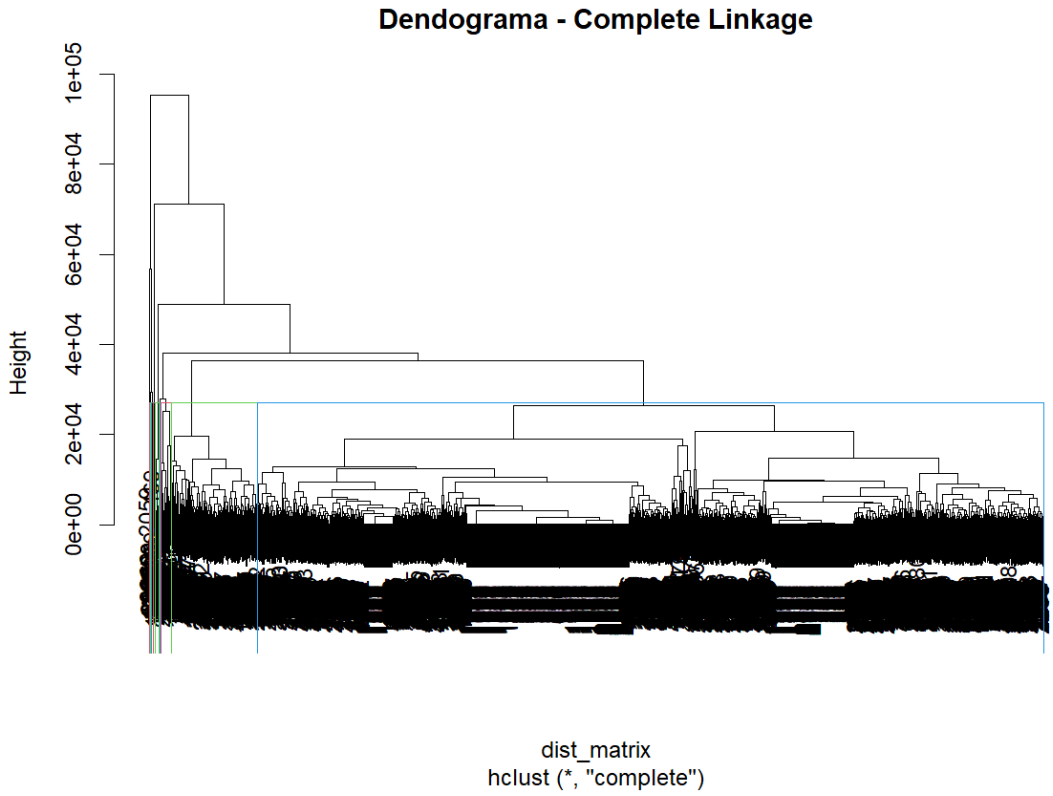

Ilustración 38.Complete Linkage Corte 9 Clústeres

Los resultados obtenidos utilizando el método jerárquico difieren con respecto al método k-medoids, por lo que el siguiente paso será compararlos para seleccionar aquel que obtenga mejores resultados. Para seleccionar el mejor algoritmo, utilizaremos tres métricas diferentes. Primero, evaluaremos la conectividad de cada método, donde un valor más bajo indica un mejor rendimiento. En contraste, tanto el índice de Dunn como el ancho de la silueta deben ser lo más altos posibles para reflejar un rendimiento superior del algoritmo.

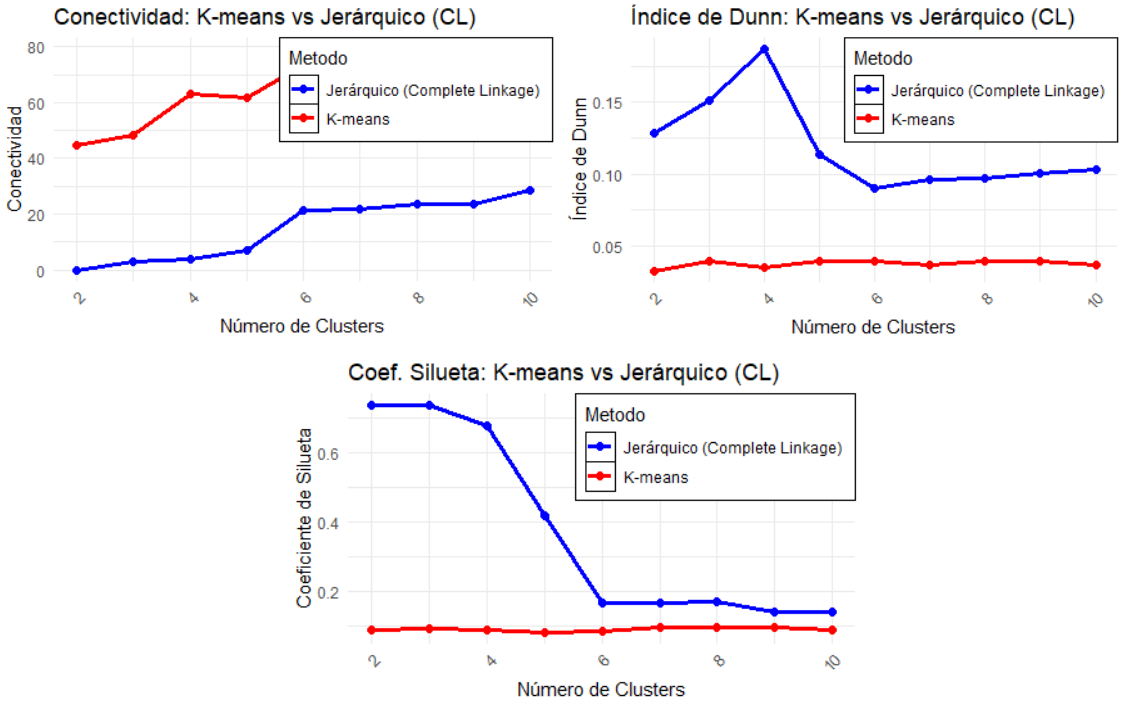

Ilustración 39.Gráficos Metricas Comparativas

Como se observa en los gráficos anteriores, la conectividad es menor para el método jerárquico, independientemente del número de clústeres seleccionados. Además, tanto el índice de Dunn como el coeficiente de la silueta son consistentemente mayores para este método. Por lo tanto, se optará por el clustering jerárquico con distancia euclidiana y enlace completo (Complete Linkage) para la creación de los 9 grupos de distritos municipales.

3.5 ANÁLISIS MULTIVARIANTE

Un análisis multivariante se refiere al estudio de múltiples variables simultáneamente para comprender las relaciones complejas entre ellas. En el contexto de datos categóricos, como los obtenidos a partir de un análisis de clúster, el objetivo principal es identificar patrones y agrupaciones dentro de los datos basados en las características observadas.

En nuestro caso lo utilizaremos sobre la variable resultado del análisis clúster. Este factor resultante se utilizará como variable dependiente en el modelo para explorar cómo las variables predictoras afectan la pertenencia a cada clúster. Al tratar con una variable factor con 9 niveles elegimos realizar una **regresión multinomial**, puesto que es especialmente apropiada cuando la variable de interés (en este caso, el factor resultado del análisis de clúster) tiene más de dos categorías.

El modelo de la **regresión multinomial** sigue la siguiente formula:

$$\log\left(\frac{P(Y = Kt \mid X)}{P(Y = k \mid X)}\right) = \beta k0 + \beta k1 * X1 + \beta k2 * X2 + \cdots + \beta kp * Xp$$

$$para\ k = 1{,}2, \ldots, Kt - 1$$

Donde:

- $P(Y = k \mid X)$ es la probabilidad de que la observación pertenezca a la categoría k dado el vector de variables independientes X.
- $\beta k0, \beta k1, \ldots, \beta kp$ son los coeficientes del modelo para la categoría k.
- Kt es el número total de categorías en la variable dependiente Y.
- $X1, X2, \ldots, Xp$ son las variables independientes, las predictoras.

En nuestro caso la criba por correlación hecha anteriormente ayuda a cumplir que estas variables no estén muy correlacionadas entre sí, facilitando la interpretación.

Esta regresión permite además predecir la probabilidad de pertenencia a cada categoría del factor resultante del análisis de clúster como veremos a continuación.

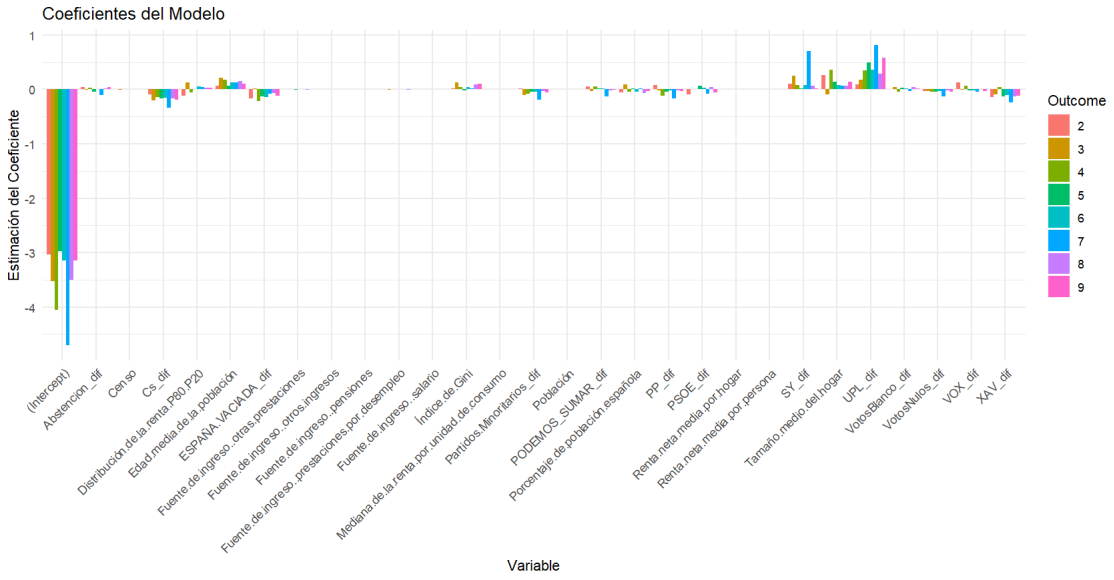

Ilustración 40.Coeficientes Regresión Multinomial

Al observar los coeficientes obtenidos del análisis se observan varias cosas. En general los coeficientes de cada variable tienen el mismo signo para todos los grupos, lo que se puede interpretar como un efecto continuo en su influencia.

Las variables con coeficientes de mayor valor absoluto y por lo tanto que más contribuyen a la clasificación de los distritos municipales en uno de los clústeres son **SY_dif, UPL_dif, Cs_dif y Tamaño medio del hogar**. Cabe destacar que vuelven a aparecer las variables de diferencia de votos tomadas como frontera entre las variables electorales y las sociodemográficas en las redes bayesianas construidas anteriormente con el algoritmo tabu.

Los **intercept** son negativos y de valor significativo pero similares, por lo que la clasificación no se decanta en exceso por alguno de los grupos en ausencia de las variables predictoras. Lo que sí significan estos valores tan altos es que la base de comparación del resto de grupos está considerablemente alejada del grupo 1, por lo que la diferencia se encuentra principalmente entre el grupo 1 y el resto.

En cuanto a los coeficientes más destacados para cada grupo (el 1 es la referencia):

- El grupo 2 tiene como principal coeficiente el **Tamaño medio del hogar**.
- El grupo 3 tanto **SY_dif**, la **Edad media de la población**, y **UPL_dif**.
- El grupo 4 sobre todo **UPL_dif** y el **Tamaño medio del hogar**.
- El grupo 5 principalmente **UPL_dif** y el **Tamaño medio del hogar**, pero con un coeficiente menor al del grupo 4.
- El grupo 6 tambien **UPL_dif** y en menor medida la **Edad media de la población**.
- El grupo 7 destaca con los coeficientes de **UPL_dif, SY_dif y Cs_dif**. Los tres tienen valores especialmente grandes para este grupo.
- El grupo 8 también esta principalmente influenciado por **UPL_dif**.
- El grupo 9 encuentra un coeficiente importante en **UPL_dif** y en **SY_dif** en menor medida.

Por lo tanto, la variable UPL_dif parece troncal en la decisión de asignar distritos municipales a clústeres distintos al más grande, el grupo 1 que ha servido como referencia.

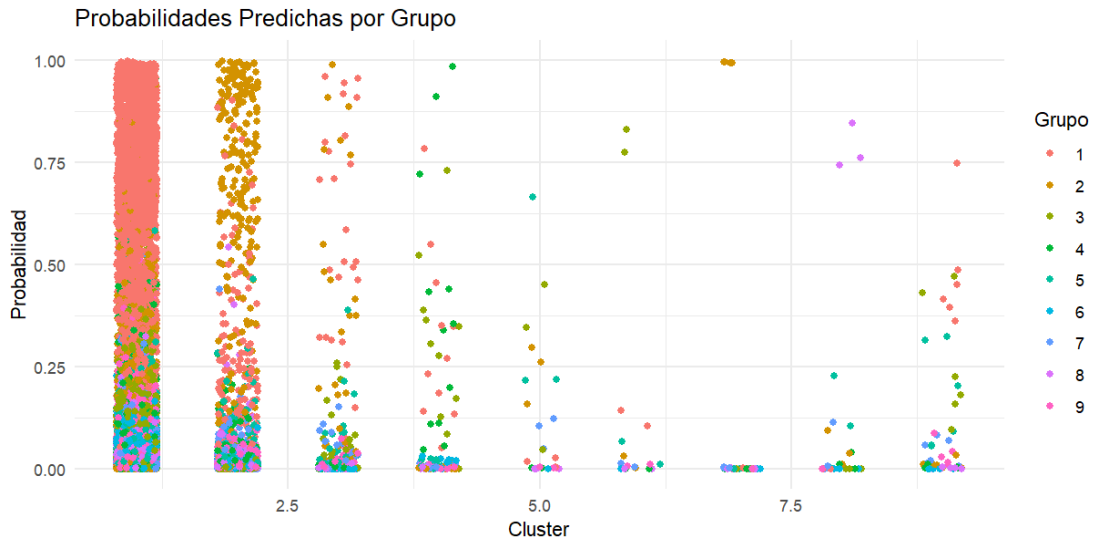

Ilustración 41.Probabilidades Predichas Regresión Multinomial

En cuanto a las probabilidades predichas para cada grupo hemos obtenido las predicciones mostradas en el grafico superior. Cabe destacar en este gráfico que la mayoría de los puntos en los clústeres 1 y 2 tienen probabilidades cercanas a 1 de predecir su asignación a sus grupos correctos. Esto sugiere una fuerte clasificación de observaciones en ambos grupos con alta certeza.

En cuanto al resto de los grupos, del 3 al 9, las probabilidades son más bajas y mas dispersas en general. Hay algunos puntos que se destacan (probabilidades altas) en clústeres superiores (como en los clústeres 4 y 7), lo que indica que, aunque la mayoría de las observaciones tienen bajas probabilidades de pertenecer a estos grupos, hay algunas que el modelo clasifica con alta certeza.

En definitiva, el análisis muestra una tendencia clara de alta probabilidad de pertenencia a los primeros dos grupos, con los otros grupos teniendo una distribución de probabilidades más baja y dispersa.

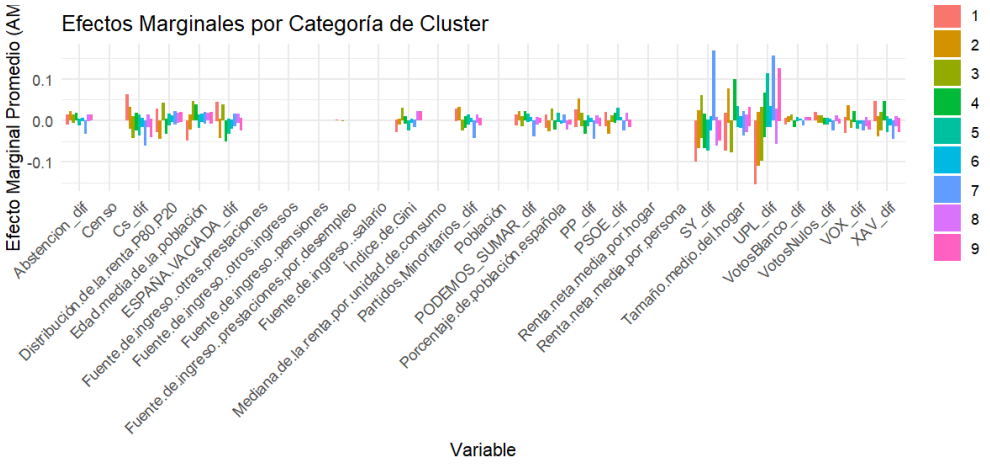

Ilustración 42.Efectos Marginales Regresión Multinomial

En el grafico relativo a los efectos marginales de las diferentes variables con respecto a los clústeres volvemos a observar el impacto que tienen principalmente **UPL_dif, SY_dif y el Tamaño medio del hogar**. Con una menor influencia se observan también valores que llegan a 0.05 como valor absoluto en **XAV_dif, Cs_dif, PP_dif** y la **Edad media de la población**.

A la hora de evaluar el ajuste del modelo de regresión multinomial hemos tomado dos indicadores estadísticos, el Log-Likelihood y el AIC. Obtenemos los siguientes valores.

	Diagnóstico	Valor
1	Log-Likelihood	-1425.043
2	AIC	3298.085

El Log-Likelihood (logaritmo de la verosimilitud) es una medida de qué tan bien el modelo explica los datos observados, mide bondad de ajuste del modelo. Por otro lado, el AIC es un criterio de selección de modelos que penaliza la complejidad del modelo.

En nuestro caso una log-likelihood de -1425.043 sugiere que el modelo tiene un buen ajuste en términos relativos a los datos que hemos utilizado. El valor del AIC, sin embargo, de 3298.085 indica que, aunque el modelo tiene un buen ajuste puede haber cierta penalización por la complejidad del modelo (número de variables predictoras). Esto tiene sentido al considerar que estamos tratando con un total de 28 variables predictoras.

4 – CONCLUSIONES Y TRABAJO FUTURO

Tras aplicar las diferentes técnicas multivariantes a la tabla obtenida después de la obtención y el tratamiento de los datos, no se ha encontrado una relación clara entre las variables sociodemográficas analizadas y la diferencia de voto en general. Sin embargo, si se han observado algunas conclusiones recurrentes en los distintos análisis.

Tras construir varias **Redes Bayesianas** se han observado dos inclinaciones en la relación entre las variables consideradas. En primer lugar, utilizando Fast Incremental Association (Fast-IAMB), la independencia entre las variables electorales y las variables sociodemográficas condicionada a la presencia de la variable Población. Y en segundo lugar la relación de las variables sociodemográficas con unas pocas variables electorales a las que nos hemos referido como 'variables frontera' con el grupo de variables sociodemográficas. Estas variables son **UPL_dif, SY_dif** y **Cs_dif**.

Utilizando el **Análisis de Componentes Principales**, se ha observado que la variable Población tiene autovectores notables en todas las componentes principales, lo que habla de una influencia global de fondo y una posible relación inversamente proporcional entre la diferencia de votos entre PP y VOX. Los resultados destacan sobre todo la complejidad de las relaciones entre variables que logran observarse, ya sea debido a las bajas contribuciones de estas en general o a la dificultad de detectar relaciones claras.

En cuanto al **Análisis Clúster** realizado que comparó el método k-medoids con el método jerárquico, se eligió el método jerárquico con complete linkage cortando el árbol resultado en **9 grupos**. Esto diversificó la clasificación lo suficiente como para estudiar las variables que influían en la separación de los distritos municipales en los distintos clústeres.

Interpretamos los resultados de la clasificación mediante un **Análisis Multivariante**, más específicamente una Regresión Multinomial, que permitió identificar las variables que más influían en la clasificación de los distritos municipales. Interpretando los coeficientes y los efectos marginales destacan las variables **UPL_dif, SY_dif, Cs_dif y Tamaño medio del hogar**, lo que incluye a las 'variables frontera' destacadas anteriormente junto con una variable sociodemográfica. La importancia repetida de estas variables parece apoyar las relaciones vistas en la red bayesiana.

En definitiva, los dos grupos de variables son **independientes condicionados a la presencia de la variable Población**, pero existen relaciones entre las llamadas 'variables frontera' y el grupo de variables sociodemográficas en caso contrario. La particularidad común en las variables frontera, pues se trata de diferencias electorales en partidos con fuerte carácter regional como son UPL y SY, es que históricamente observan una disminución en el compromiso electoral de sus votantes al tratarse de elecciones generales.

La diferencia que esto podría suponer en la diferencia electoral, fuera de nuestras variables, junto con la particularidad de que Cs pertenezca también a este grupo sin haberse presentado a las elecciones generales, nos hace pensar que estas relaciones pueden no ser de gran importancia teniendo en cuenta a la independencia condicional anteriormente mencionada.

En cuanto al **posible trabajo futuro**, se podría reforzar el estudio de relaciones tomando diferencias con nuevas elecciones realizadas, y buscando relaciones partiendo de las

conclusiones a las que se ha llegado en este caso. Se podría también colocar más el foco en el papel de la variable Población sobre la que se sustenta la independencia condicional, y como se da esta relación.

Bibliografía

Aragón Ruiz, M. (2021). Recuperado el 13 de Julio de 2024, de https://uvadoc.uva.es/handle/10324/50482

bnlearn. (s.f.). Recuperado el 11 de Julio de 2024, de https://www.bnlearn.com/

Busqueda Tabu. (s.f.). Recuperado el 13 de Julio de 2024, de https://es.wikipedia.org/wiki/B%C3%BAsqueda_tab%C3%BA

INE. (2021). Recuperado el 11 de Julio de 2024, de https://www.ine.es/dynt3/inebase/index.htm?padre=7132

INE. (2023). Recuperado el 11 de Julio de 2024, de https://www.ine.es/dynt3/inebase/index.htm?padre=10358&capsel=10358

Junta de Castilla y León. (13 de Febrero de 2022). Recuperado el 11 de Julio de 2024, de https://datosabiertos.jcyl.es/web/jcyl/set/es/sector-publico/elecciones_2022/1285131898209

Kassambara, A. (2017). *Practical Guide to Cluster Analysis in R: Unsupervised Machine Learning*. Recuperado el 13 de Julio de 2024, de https://books.google.es/books?hl=es&lr=&id=plEyDwAAQBAJ&oi=fnd&pg=PP2&dq=cluster+analysis+in+R&ots=xeGTnCiLYw&sig=h95Pk396s2tOLaf5ibEB4aA1E6U#v=onepage&q&f=false

Liang J, B. G. (8 de Agosto de 2020). *Multinomial and ordinal Logistic regression analyses with multi-categorical variables using R*. Recuperado el 13 de Julio de 2024, de https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7475459/

Michael Greenacre, P. J. (22 de Diciembre de 2022). *Principal component analysis*. Recuperado el 13 de Julio de 2024, de https://www.nature.com/articles/s43586-022-00184-w

Ministerio del Interior. (23 de Julio de 2023). Recuperado el 11 de Julio de 2024, de https://infoelectoral.interior.gob.es/es/elecciones-celebradas/area-de-descargas/

Saratha Sathasivam, P. C. (Junio de 2020). *Learning Bayesian Networks: The Combination*. Recuperado el 13 de Julio de 2024, de https://d1wqtxts1xzle7.cloudfront.net/96066465/D7568049420-libre.pdf?1671501962=&response-content-disposition=inline%3B+filename%3DLearning_Bayesian_Networks_The_Combinati.pdf&Expires=1720827303&Signature=T2hYe1FeWiLVMdSvrIwsZGNrRGPZc~hZlEh15AvHfzHWJNTNOs

LISTA DE FIGURAS

Ilustración 1.IDM 12
Ilustración 2.Unión interior 12
Ilustración 3.Unión exterior izquierda 12
Ilustración 4.Censo 16
Ilustración 5.Distribución de la renta P80/P20 16
Ilustración 6.Índice de Gini 17
Ilustración 7.Edad media de la población 17
Ilustración 8.Población 18
Ilustración 9.Porcentaje de población española 18
Ilustración 10.Tamaño medio del hogar 19
Ilustración 11.Mediana de la renta por unidad de consumo 19
Ilustración 12.Renta neta media por hogar 20
Ilustración 13.Renta neta media por persona 20
Ilustración 14.Fuente de ingreso por otras prestaciones 21
Ilustración 15.Fuente de ingreso por otros ingresos 21
Ilustración 16.Fuente de ingreso por pensiones 22
Ilustración 17.Fuente de ingreso por prestaciones por desempleo 22
Ilustración 18.Fuente de ingreso por salario 23
Ilustración 19.Red General Tabu 24
Ilustración 20.Red Cs Tabu 25
Ilustración 21.Red UPL Tabu 25
Ilustración 22.Red SY Tabu 25
Ilustración 23.Red Global Fast-IAMB 26
Ilustración 24.Varianza Explicada 29
Ilustración 25.Contribuciones PCA 31
Ilustración 26.Correlaciones PCA 32
Ilustración 27.Representaciones PCA 32
Ilustración 28.Gráficos de coordenadas 1 y 2 34
Ilustración 29.Gráficos de coordenadas 3, 4 y 5 35
Ilustración 30.Gráficos de coordenadas 6, 7 y 8 35
Ilustración 31.Gráficos de coordenadas 9, 10, 11 y 12 36
Ilustración 32.Gráficos de coordenadas 13, 14 y 15 36
Ilustración 33.Gap Statistic por Número de Clústeres 38
Ilustración 34.Cluster plot 38
Ilustración 35.Dendograma Complete Linkage 40
Ilustración 36.Dendograma Average Linkage 40
Ilustración 37.Dendograma Single Linkage 41
Ilustración 38.Complete Linkage Corte 9 Clústeres 41
Ilustración 39.Gráficos Metricas Comparativas 42
Ilustración 40.Coeficientes Regresión Multinomial 43
Ilustración 41.Probabilidades Predichas Regresión Multinomial 44
Ilustración 42.Efectos Marginales Regresión Multinomial 45

LISTA DE TABLAS

Tabla 1.Censo.....15
Tabla 2.Distribución de la renta P80/P20.....16
Tabla 3.Índice de Gini.....16
Tabla 4.Edad media de la población.....17
Tabla 5.Población.....17
Tabla 6.Porcentaje de población española.....18
Tabla 7.Tamaño medio del hogar.....19
Tabla 8.Mediana de la renta por unidad de consumo.....19
Tabla 9.Renta neta media por hogar.....20
Tabla 10.Renta neta media por persona.....20
Tabla 11.Fuente de ingreso por otras prestaciones.....21
Tabla 12.Fuente de ingreso por otros ingresos.....21
Tabla 13.Fuente de ingreso por pensiones.....22
Tabla 14.Fuente de ingreso por prestaciones por desempleo.....22
Tabla 15.Fuente de ingreso por salario.....23
Tabla 16.Varianza Explicada Opción1.....28
Tabla 17.Varianza Explicada Opción2.....34

ANEXO DE CÓDIGOS

La obtención y unión de datos se han realizado con Python, y el resto de los códigos con R.

OBTENCIÓN DE DATOS

Extracción elecciones generales

```
import numpy as np
import pandas as pd

#Candidaturas/partidos
candidaturas =
pd.read_fwf("02202307_MESA/03022307.DAT",widths=[2,4,2,6,50,150,6,6,6],header=None,
encoding='latin-1')
candidaturas.columns = ['Tipo','Ano','Mes','Codcan','Sigla','Denominacion',
'Cod_Prov','Cod_CCAA','Cod_Nacion']
candidaturas

# creo el fichero de votos
votos = pd.read_fwf("02202307_MESA/10022307.DAT", widths=[2,4,2,1,2,2,3,2,4,1,6,7], header=
None, encoding='latin1')
votos.columns =
['TIPO','Anio','Mes','Nvuelta','Ccaa','Cprov','Cmun','Cdist','Csecc','Cmesa','Codcan','Tvotos']
votos
# Nos quedamos solo con Castilla y León
votoscyl = votos.loc[votos.Ccaa==8]
# comprobamos que sólo tenemos las provincias de CyL
votos.Cprov.unique()
#hacemos merge para obtener nombre candiadturas
votoscyl =votoscyl.merge(candidaturas,how='left', on='Codcan' )
votoscyl

#Añadimos IDSC al dataframe
columnas_a_combinar = ['Cprov', 'Cmun', 'Cdist', 'Csecc']
for col in columnas_a_combinar:
    longitud = 2  # longitud predeterminada
    if col == 'Cmun':
        longitud = 3
    elif col == 'Csecc':
        longitud = 3
    votoscyl[col] = votoscyl[col].astype(str).str.zfill(longitud)
votoscyl['IDSC'] = votoscyl[columnas_a_combinar].apply(lambda row: ''.join(map(str, row)),
axis=1)
#for col in columnas_a_combinar:
#    votoscyl[col].astype(int)
votoscyl
#votoscyl['Sigla'].unique()
print(votoscyl.columns)
```

```
#Primero quitamos las columnas innecesarias y dejamos 'Siglas' como variable que pasar a ancho
#Pasamos partidos a ancho
votoscyl = votoscyl.drop(columns=['Denominacion', 'Cod_Prov', 'Cod_CCAA', 'Cod_Nacion'])
votoscyl.head()

#Pasamos partidos a ancho
pivot_votoscyl = pd.pivot_table(votoscyl, index='IDSC', columns='Sigla',values='Tvotos',
aggfunc='sum', fill_value=0)
pivot_votoscyl = pivot_votoscyl.reset_index()
pivot_votoscyl

#Agrupamos por secciones censales en votoscyl antes de hacer el merge
votoscyl = votoscyl.drop(columns=['Sigla'])
votoscyl = votoscyl.drop(columns=['Cmesa'])
columnas_a_sumar = ['Tvotos']
#votoscyl = votoscyl.groupby('IDSC').sum().reset_index()
votoscyl = votoscyl.groupby('IDSC').agg({**{col: 'sum' for col in columnas_a_sumar}, **{col:
'first' for col in votoscyl.columns if col not in columnas_a_sumar}})
votoscyl

#Con los datos de los partidos volvemos a poner en dataset en conjunto
votoscyl = votoscyl.drop(columns=['IDSC'])
votoscyl =votoscyl.merge(pivot_votoscyl,how='left', on='IDSC' )
votoscyl

#Guardamos el csv
votoscyl.to_csv(r'C:\...\Datos Ambas Elecciones\votoscyl.csv', sep=';', index=False,
encoding='latin-1')
```

Extracción elecciones cortes de Castilla y León

```
import numpy as np
import pandas as pd
#Cogemos los datos de CyL
datos_CyL = pd.read_csv(r'C:\...\Datos Ambas Elecciones\ResultadosElectorales2022 (1).csv',
sep=';',encoding='latin-1')

#Para evitar repeticiones de mesas por partido, haremos pivot al DataFrame de CyL
pivot_cyl = pd.pivot_table(datos_CyL, index='Código de mesa', columns='Partido',values='Nº
Votos', aggfunc='sum', fill_value=0)
pivot_cyl = pivot_cyl.reset_index()
pivot_cyl.to_csv(r'C:\...\Datos Ambas Elecciones\pivotcyl.csv', sep=';', index=False,
encoding='latin-1')

#Dropeamos las columnas de partidos en el dataframe original
datos_CyL.drop(['Partido', 'Nº Votos'], axis=1, inplace=True)
datos_CyL = pd.merge(datos_CyL, pivot_cyl, left_on='Código de mesa', right_on='Código de mesa',
how='inner')
```

```
datos_CyL.drop_duplicates(subset='Código de mesa', inplace=True)

datos_CyL.to_csv(r'C:\...\Datos Ambas Elecciones\datoscyl.csv', sep=';', index=False,
encoding='latin-1')

Creación IDSC en tabla de elecciones cortes de Castilla y León
dataCyL = pd.read_csv('datoscyl.csv',sep=';',encoding='latin-1')
#Ahora colapsamos el codigo de seccion censal, quitando las mesas
# Extraer los primeros 11 caracteres de la columna 'IDGen' (sin contar las letras)
dataCyL['Secciones'] = dataCyL['Código de mesa'].str.extract('^([\d-]+)')
dataCyL['Secciones'] = dataCyL['Secciones'].apply(lambda x: x.rstrip('-'))
dataCyL['Secciones'] = dataCyL['Secciones'].str.replace('-', '')  # Elimina los guiones
dataCyL['Secciones'] = dataCyL['Secciones'].apply(lambda x: x[:5] + '0' + x[5:] if len(x) < 10
else x)  # Añade '0' en la sexta posición
# Colapsar el DataFrame por la columna 'Secciones' y sumar el resto de valores
dataCyL = dataCyL.groupby('Secciones').sum().reset_index()

dataCyL.to_csv('datoscyl2.csv',sep=';',encoding='latin-1')
```

Colapso de variables electorales en Distritos Municipales

```
import numpy as np
import pandas as pd


dataGen = pd.read_csv('votoscyl(generales).csv',sep=';',encoding='latin-1')
dataGen
dataCortes = pd.read_csv('datoscyl2(cortes).csv',sep=';',encoding='latin-1')
dataCortes


#Creamos la nueva variable IDM (Identificador Distrito Muestral) en ambas bases de datos

for i in range(len(dataGen)):
    first_digit = int(str(dataGen['IDSC'].iloc[i])[0])
    if first_digit == 5 or first_digit == 9:
        dataGen.at[i, 'IDM'] = str(dataGen['IDSC'].iloc[i])[:6]
    else:
        dataGen.at[i, 'IDM'] = str(dataGen['IDSC'].iloc[i])[:7]

for i in range(len(dataCortes)):
    first_digit = int(str(dataCortes['Secciones'].iloc[i])[0])
    if first_digit == 5 or first_digit == 9:
        dataCortes.at[i, 'IDM'] = str(dataCortes['Secciones'].iloc[i])[:6]
    else:
        dataCortes.at[i, 'IDM'] = str(dataCortes['Secciones'].iloc[i])[:7]

columnas_a_sumar = ['Tvotos', '3e', 'CJ', 'ESCAÑOS EN BLANCO', 'ESPAÑA VACIADA',
       'EV-PCAS-TC', 'FE de las JONS', 'FO', 'FUERZA CÍVICA', 'GITV', 'PACMA',
       'PCTE', 'PP', 'PREPAL', 'PSOE', 'PUM+J', 'RECORTES CERO', 'SUMAR', 'SY',
```

```
       'U.P.L.', 'Ud.Ca', 'VB', 'VOX', 'VP', 'XAV', 'Zsi', 'Censo',
       'VotosBlanco', 'VotosNulos', 'TotalVotosCandidaturas', 'Abstencion']
dataGen = dataGen.groupby('IDM').agg({**{col: 'sum' for col in columnas_a_sumar}, **{col:
'first' for col in dataGen.columns if col not in columnas_a_sumar}})
dataGen

columnas_a_sumar = ['Censo', 'Tvotos', 'Votos Nulos',
       'Votos Blanco', 'C.Bierzo - BEX', 'Cs', 'DESPIERTA', 'EB',
       'ESPAÑA VACIADA', 'FE de las JONS', 'P.S.O.E.', 'PACMA', 'PCAS-TC-RC',
       'PCAS-TC-Recortes Cero', 'PCTE', 'PODEMOS-IU-AV', 'POR ZAMORA', 'PP',
       'PP.CC.AL', 'PREPAL', 'PSOE', 'PUEDE', 'PUM+J', 'PYC (ANULADA)', 'SY',
       'TAB', 'UNIÓN REGIONALISTA', 'UPL', 'VOLT', 'VOX', 'XAV',
       'ZAMORA DECIDE', 'centrados', 'Abstencion']
dataCortes = dataCortes.groupby('IDM').agg({**{col: 'sum' for col in columnas_a_sumar}, **{col:
'first' for col in dataCortes.columns if col not in columnas_a_sumar}})
dataCortes

#dropeamos los IDSC y guardamos los csv
dataGen.drop(columns=['IDSC'])
dataCortes.drop(columns=['Secciones'])

dataGen.to_csv('votosGen.csv', sep=';', index=False, encoding='latin-1')
dataCortes.to_csv('votosCortes.csv', sep=';', index=False, encoding='latin-1')
```

Extracción covariables de renta

```
import numpy as np
import pandas as pd


def filtrar2021(archivocsv):
    data = pd.read_csv(archivocsv,sep=';')
    data = data.loc[data['Periodo'] == 2021]
    data_2021 = data[(data['Distritos'].notna()) & (data['Secciones'].isna())]
    data_2021
    return data_2021


#AVILA
#Indice de guini y P80/P20
guini = filtrar2021("Avila\\37683.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Avila\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Avila\\30877.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
```

```
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Avila\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Avila\\30869.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Avila\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Avila\\30870.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Avila\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Avila\\Total.csv", sep=';', index=False, encoding='latin-1')
totalAvila = total


#BURGOS
#Indice de guini y P80/P20
guini = filtrar2021("Burgos\\37687.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Burgos\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Burgos\\30934.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Burgos\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Burgos\\30926.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Burgos\\Renta.csv", sep=';', index=False, encoding='latin-1')
```

```
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Burgos\\30927.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Burgos\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Burgos\\Total.csv", sep=';', index=False, encoding='latin-1')
totalBurgos = total


#LEON
#Indice de guini y P80/P20
guini = filtrar2021("Leon\\37703.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Leon\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Leon\\31078.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Leon\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Leon\\31070.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Leon\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Leon\\31071.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Leon\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
```

```
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Leon\\Total.csv", sep=';', index=False, encoding='latin-1')
totalLeon = total


#PALENCIA
#Indice de guini y P80/P20
guini = filtrar2021("Palencia\\37708.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Palencia\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Palencia\\31150.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Palencia\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Palencia\\31142.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Palencia\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Palencia\\31143.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Palencia\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Palencia\\Total.csv", sep=';', index=False, encoding='latin-1')
totalPalencia = total


#SALAMANCA
#Indice de guini y P80/P20
guini = filtrar2021("Salamanca\\37713.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
```

```
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Salamanca\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Salamanca\\31186.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Salamanca\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Salamanca\\31178.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Salamanca\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Salamanca\\31179.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Salamanca\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Salamanca\\Total.csv", sep=';', index=False, encoding='latin-1')
totalSalamanca = total


#SEGOVIA
#Indice de guini y P80/P20
guini = filtrar2021("Segovia\\37715.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Segovia\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Segovia\\31204.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
```

```
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Segovia\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Segovia\\31196.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Segovia\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Segovia\\31197.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Segovia\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Segovia\\Total.csv", sep=';', index=False, encoding='latin-1')
totalSegovia = total


#SORIA
#Indice de guini y P80/P20
guini = filtrar2021("Soria\\37717.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Soria\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Soria\\31222.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Soria\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Soria\\31214.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Soria\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
```

```
ingreso = filtrar2021("Soria\\31215.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Soria\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Soria\\Total.csv", sep=';', index=False, encoding='latin-1')
totalSoria = total


#VALLADOLID
#Indice de guini y P80/P20
guini = filtrar2021("Valladolid\\37722.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Valladolid\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Valladolid\\31267.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Valladolid\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Valladolid\\31259.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Valladolid\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Valladolid\\31260.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Valladolid\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
```

```
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Valladolid\\Total.csv", sep=';', index=False, encoding='latin-1')
totalValladolid = total

#ZAMORA
#Indice de guini y P80/P20
guini = filtrar2021("Zamora\\37723.csv")
guini['Secciones'] = guini['Distritos'].str[:7]
guini = pd.pivot_table(guini, index='Distritos', columns='Índice de Gini y Distribución de la
renta P80/P20', values='Total',aggfunc='sum', fill_value='.',dropna=False)
guini = guini.reset_index()
guini.to_csv("Zamora\\Guini.csv", sep=';', index=False, encoding='latin-1')
#Indicadores demográficos
demograficos = filtrar2021("Zamora\\31276.csv")
demograficos['Secciones'] = demograficos['Distritos'].str[:7]
demograficos = pd.pivot_table(demograficos, index='Distritos', columns='Indicadores
demográficos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
demograficos = demograficos.reset_index()
del demograficos['Porcentaje de hogares unipersonales']
demograficos.to_csv("Zamora\\Demograficos.csv", sep=';', index=False, encoding='latin-1')
#Indicadores renta media y mediana
renta = filtrar2021("Zamora\\31268.csv")
renta['Secciones'] = renta['Distritos'].str[:7]
renta = pd.pivot_table(renta, index='Distritos', columns='Indicadores de renta media y mediana',
values='Total',aggfunc='sum', fill_value='.',dropna=False)
renta = renta.reset_index()
renta.to_csv("Zamora\\Renta.csv", sep=';', index=False, encoding='latin-1')
#Distribución por fuentes de ingreso
ingreso = filtrar2021("Zamora\\31269.csv")
ingreso['Secciones'] = ingreso['Distritos'].str[:7]
ingreso = pd.pivot_table(ingreso, index='Distritos', columns='Distribución por fuente de
ingresos', values='Total',aggfunc='sum', fill_value='.',dropna=False)
ingreso = ingreso.reset_index()
del ingreso['Renta bruta media por persona']
ingreso.to_csv("Zamora\\Ingreso.csv", sep=';', index=False, encoding='latin-1')
#Juntamos todas las variables
total = pd.merge(guini, demograficos, on='Distritos')
total = pd.merge(total, renta, on='Distritos')
total = pd.merge(total, ingreso, on='Distritos')
total.to_csv("Zamora\\Total.csv", sep=';', index=False, encoding='latin-1')
totalZamora = total

#Tabla final CyL
dataframes = [totalAvila, totalBurgos, totalLeon, totalPalencia, totalSalamanca, totalSegovia,
totalSoria, totalValladolid, totalZamora]
totalFinal = pd.concat(dataframes, ignore_index=True)

for i in range(len(totalFinal)):
```

```
    totalFinal.at[i, 'IDM'] = str(totalFinal['Distritos'].iloc[i])[:7]

columnas = ['IDM'] + [col for col in totalFinal.columns if col != 'IDM']
totalFinal = totalFinal[columnas]
totalFinal = totalFinal.drop(columns=['Distritos'])
#Guardamos
totalFinal.to_csv("CyLRentasFinal.csv", sep=';', index=False, encoding='latin-1')
```

Extracción covariables demográficas

```
import numpy as np
import pandas as pd


data=pd.read_csv('61441.csv', header=0, sep=';',encoding='utf-8')
data


#Nos deshacemos de las que no son Castilla y Leon usando un filtro de provincias de CyL
cyl =['05','09','24','34','37','40','42','47','49']

index_eliminar=[]

for i in range(len(data['Provincias'])):
    provincia = data['Provincias'].loc[i]
    if pd.isna(provincia):
        index_eliminar.append(i)
    elif provincia[:2] not in cyl:
        index_eliminar.append(i)

data = data.drop(index=index_eliminar)


#Quitamos las observaciones que no son seccion muestral
data.reset_index(drop=True, inplace=True)
index_eliminar=[]
for i in range(len(data)):
    seccion = data['Secciones'].iloc[i]
    if pd.isna(seccion):
        index_eliminar.append(i)

data = data.drop(index=index_eliminar)
data.to_csv('data.csv', index=False, sep=';')

#Creamos IDM
for i in range(len(data['Secciones'])):
    #print(data['Secciones'].iloc[i][:7])
    data['Secciones'].iloc[i] = data['Secciones'].iloc[i][:7]

data = data.rename(columns={'Secciones': 'IDM'})
data = data.drop(columns=['Total Nacional','Provincias','Municipios'])
```

```
data = pd.pivot_table(data, index='IDM', columns=['Sexo', 'Edad (grupos quinquenales)'],
values='Total', aggfunc='sum', fill_value=0)
data = data.reset_index()

data.columns = ['_'.join(col).strip() for col in data.columns.values]

# Definir los intervalos de edades
intervalos = {
    '0-9 años': ['De 0 a 4 años', 'De 5 a 9 años'],
    '10-19 años': ['De 10 a 14 años', 'De 15 a 19 años'],
    '20-29 años': ['De 20 a 24 años', 'De 25 a 29 años'],
    '30-39 años': ['De 30 a 34 años', 'De 35 a 39 años'],
    '40-49 años': ['De 40 a 44 años', 'De 45 a 49 años'],
    '50-59 años': ['De 50 a 54 años', 'De 55 a 59 años'],
    '60-69 años': ['De 60 a 64 años', 'De 65 a 69 años'],
    '70-79 años': ['De 70 a 74 años', 'De 75 a 79 años'],
    '80-89 años': ['De 80 a 84 años', 'De 85 a 89 años'],
    '90-99 años': ['De 90 a 94 años', 'De 95 a 99 años'],
    '100+ años': ['100 y más años']
}

# Crear un DataFrame para almacenar los datos colapsados
collapsed_data = pd.DataFrame()
collapsed_data['IDM'] = data['IDM_']

# Sumar los valores de los intervalos definidos
for intervalo, columnas in intervalos.items():
    col_names = [f'Ambos sexos_{col}' for col in columnas if f'Ambos sexos_{col}' in
data.columns]
    if col_names:
        collapsed_data[intervalo] = data[col_names].sum(axis=1)

# Agregar las columnas de 'Total' para 'Mujer' y 'Hombre'
collapsed_data['TotalHombres'] = data['Hombre_Total']
collapsed_data['TotalMujeres'] = data['Mujer_Total']

collapsed_data['TotalHombres'] = collapsed_data['TotalHombres'].astype(int)

collapsed_data['TotalMujeres'] = collapsed_data['TotalMujeres'].astype(int)

collapsed_data.to_csv('data4.csv', index=False, sep=';',encoding='latin-1')
```

UNIÓN DE DATOS

Creación Tabla Final

```
import numpy as np
import pandas as pd


data1 = pd.read_csv('../Renta de los hogares/CyLRentasFinal.csv', header=0,
sep=';',encoding='latin-1')
data2 = pd.read_csv('..//Proporciones edad (tramos) y sexo/data4.csv', header=0,
sep=';',encoding='latin-1')


data1.drop(columns=['Distritos'], inplace=True)

# Hacer merge por la columna 'IDM'
data = pd.merge(data1, data2, on='IDM')

cols = data.columns.tolist()
cols = ['IDM'] + [col for col in cols if col != 'IDM']
data = data[cols]
data.to_csv('data.csv', index=False, sep=';',encoding='latin-1')
```

Correcciones Tabla Final

```
import numpy as np
import pandas as pd

data1 = pd.read_csv('votosCortes.csv', header=0, sep=';',encoding='latin-1')
data2 = pd.read_csv('votosGen.csv', header=0, sep=';',encoding='latin-1')

data1 = data1.drop(columns=["Unnamed: 0","Secciones"])

df1_filtrado = data1[data1['IDM'].isin(data2['IDM'])]

# Filtrar df2 para conservar solo las filas con IDM presente en df1
df2_filtrado = data2[data2['IDM'].isin(data1['IDM'])]

#Guardamos los archivos con IDM igual en ambos
df1_filtrado.to_csv('votosCortes.csv', index=False, sep=';',encoding='latin-1')
df2_filtrado.to_csv('VotosGen.csv', index=False, sep=';',encoding='latin-1')
```

DEPURACIÓN DE DATOS

Correlación de variables

```
library(tidyverse)

datos <- read.csv("data.csv", sep=';', dec = ',',na.strings = '.', fileEncoding =
'latin1',header = TRUE)
datos$Población <- as.numeric(datos$Población)
datos$Fuente.de.ingreso..otras.prestaciones <-
as.numeric(datos$Fuente.de.ingreso..otras.prestaciones)

#Variables eliminadas poco a poco por correlacion
datos <- subset(datos, select = -
c(X100..años,Porcentaje.de.población.menor.de.18.años,Renta.bruta.media.por.hogar,Renta.bruta.me
dia.por.persona,Media.de.la.renta.por.unidad.de.consumo,Porcentaje.de.población.de.65.y.más.años
,X10.19.años,TotalHombres,TotalMujeres,X0.9.años,X20.29.años,X30.39.años,X40.49.años,X50.59.años
,X60.69.años,X70.79.años,X80.89.años,X90.99.años))

# Calcular la matriz de correlación
columnas_con_na <- colnames(datos)[apply(datos, 2, anyNA)]
for (col in columnas_con_na) {
  datos[[col]][is.na(datos[[col]])] <- mean(datos[[col]], na.rm = TRUE)
}
cor_matrix <- cor(datos, use = "complete.obs")

# Convertir la matriz de correlación a un dataframe largo (tidy)
cor_long <- as.data.frame(as.table(cor_matrix))

# Filtrar los pares de variables con correlación superior a 0.8 (y diferente de 1)
high_cor_pairs <- cor_long %>%
  filter(Var1 != Var2 & abs(Freq) > 0.8) %>%
  arrange(desc(abs(Freq)))

high_cor_pairs

write.csv2(datos, file = "covariables.csv", row.names = FALSE)
```

ANÁLISIS DESCRIPTIVO

Distribución Censo

```
library(ggplot2)
library(dplyr)
# Leer el archivo CSV
data <- read.csv("gruposCortes.csv", sep = ";", dec=',', fileEncoding="latin1", header = TRUE)
# Definir el número de intervalos para los histogramas
num_intervals <- 20
# Obtener el nombre de la columna actual
column_name <- names(data)[1]

# Calcular estadísticas descriptivas básicas
print(paste("Summary of", column_name, ":"))
print(summary(data[[1]]))

# Crear un dataframe temporal para la columna actual
df <- data.frame(Value = data[[1]])
# Crear un título sin puntos para el gráfico
tituloSinPuntos <- gsub("\\.", " ", column_name)

# Generar el gráfico de distribución
p <- ggplot(df, aes(x = Value)) +
  geom_histogram(binwidth = (max(df$Value, na.rm = TRUE) - min(df$Value, na.rm = TRUE)) /
num_intervals, fill = "blue", color = "black") +
  labs(title = tituloSinPuntos, x = "Valores", y = "Frecuencia") +
  theme_minimal()

# Mostrar el gráfico
print(p)
```

Distribución Resto de Covariables

```
library(ggplot2)
library(dplyr)

# Leer el archivo CSV
data <- read.csv("covariables.csv", sep = ";", dec=',', fileEncoding="latin1", header = TRUE)
# Definir el número de intervalos para los histogramas
num_intervals <- 20

# Iterar sobre cada columna del dataframe
for (i in 1:ncol(data)) {
  # Obtener el nombre de la columna actual
  column_name <- names(data)[i]

  # Calcular estadísticas descriptivas básicas
  print(paste("Summary of", column_name, ":"))
  print(summary(data[[i]]))

  # Crear un dataframe temporal para la columna actual
  df <- data.frame(Value = data[[i]])
  # Crear un título sin puntos para el gráfico
  tituloSinPuntos <- gsub("\\.", " ", column_name)

  # Generar el gráfico de distribución
  p <- ggplot(df, aes(x = Value)) +
    geom_histogram(binwidth = (max(df$Value, na.rm = TRUE) - min(df$Value, na.rm = TRUE)) /
num_intervals, fill = "blue", color = "black") +
    labs(title = tituloSinPuntos, x = "Valores", y = "Frecuencia") +
    theme_minimal()
  # Mostrar el gráfico
  print(p)
}
```

REDES BAYESIANAS

Construcción de redes

```
library(ggplot2)
library(bnlearn)
library(igraph)
library(infotheo)

# Cargar los datos de los resultados electorales para dos periodos
resultados_periodo1 <- read.csv("gruposCortes.csv", sep=';', header = TRUE, fileEncoding =
"Latin1")
resultados_periodo2 <- read.csv("gruposGen.csv", sep=';', header = TRUE, fileEncoding =
"Latin1")

# Cargar los datos de las covariables
covariables <- read.csv("covariables.csv", sep=';', dec = ',',na.strings = '.',header = TRUE,
fileEncoding = "Latin1")

excluir_columnas <- c("IDM","Censo","Tvotos")

# Calcular porcentaje de votos por partido en cada periodo
resultados_periodo1_porcentaje <- resultados_periodo1[, !names(resultados_periodo1) %in%
excluir_columnas] /
  rowSums(resultados_periodo1[, !names(resultados_periodo1) %in% excluir_columnas]) * 100

resultados_periodo2_porcentaje <- resultados_periodo2[, !names(resultados_periodo2) %in%
excluir_columnas] /
  rowSums(resultados_periodo2[, !names(resultados_periodo2) %in% excluir_columnas]) * 100

# Extraer las columnas IDM, Censo y Tvotos de las tablas originales
columnas_adicionales <- resultados_periodo1[, c("IDM", "Censo", "Tvotos"), drop = FALSE]

#Tomamos ciudadanos como 0 en
resultados_periodo2_porcentaje$Cs <- 0

datos_op1 <- data.frame( #Lo definimos como generales-cortes
  VotosNulos_dif = resultados_periodo2_porcentaje$VotosNulos -
resultados_periodo1_porcentaje$Votos.Nulos,
  VotosBlanco_dif = resultados_periodo2_porcentaje$VotosBlanco -
resultados_periodo1_porcentaje$Votos.Blanco,
  Cs_dif = resultados_periodo2_porcentaje$Cs - resultados_periodo1_porcentaje$Cs,
  ESPAÑA.VACIADA_dif = resultados_periodo2_porcentaje$ESPAÑA.VACIADA -
resultados_periodo1_porcentaje$ESPAÑA.VACIADA,
  PODEMOS_SUMAR_dif = resultados_periodo2_porcentaje$SUMAR -
resultados_periodo1_porcentaje$PODEMOS, #Estandarizamos SUMAR como podemos por desigualdad+
similitud de voto
  PP_dif = resultados_periodo2_porcentaje$PP - resultados_periodo1_porcentaje$PP,
  PSOE_dif = resultados_periodo2_porcentaje$PSOE - resultados_periodo1_porcentaje$PSOE,
```

```
  SY_dif = resultados_periodo2_porcentaje$SY - resultados_periodo1_porcentaje$SY,
  UPL_dif = resultados_periodo2_porcentaje$UPL - resultados_periodo1_porcentaje$UPL,
  VOX_dif = resultados_periodo2_porcentaje$VOX - resultados_periodo1_porcentaje$VOX,
  XAV_dif = resultados_periodo2_porcentaje$XAV - resultados_periodo1_porcentaje$XAV,
  Abstencion_dif = resultados_periodo2_porcentaje$Abstencion -
resultados_periodo1_porcentaje$Abstencion,
  Partidos.Minoritarios_dif = resultados_periodo2_porcentaje$Partidos.Minoritarios -
resultados_periodo1_porcentaje$Partidos.Minoritarios
  )

#Añadimos Censo a las covariables
covariables<-cbind(covariables,columnas_adicionales$Censo)

#Estandarizado de datos
standarize<-function(variable){
  return (variable-mean(variable))/sd(variable)
}
#covariables<-as.data.frame(apply(covariables,2,standarize))

#Total
data1 <- cbind(covariables,datos_op1)
#names(data1)[names(data1) == "datos_op1$VOX_dif"] <- "VOX_dif"
data1 <- subset(data1, select = -c(IDM))
data1_scaled<-as.data.frame(apply(data1,2,standarize))
data1_scaled<-sapply(data1_scaled,as.numeric)
data1_scaled<-as.data.frame(data1_scaled)

pdag <- tabu(data1_scaled,maxp=2)

g <- graph_from_adjacency_matrix(amat(pdag), mode="directed", diag=FALSE)
# Personalizar el gráfico
plot(g,
     vertex.label = V(g)$name,
     vertex.size = 15,
     vertex.label.cex = 0.8,
     vertex.label.color = "black",
     vertex.color = "lightblue",
     edge.arrow.size = 0.1,
     layout = layout_with_fr)

fitted = bn.fit(pdag, data1_scaled,method="mle-g")

#Curiosidad Independencia de variables por población con el fast.imab
pdag <- fast.iamb(data1_scaled,max.sx=5)
dag <- cextend(pdag)
g <- graph_from_adjacency_matrix(amat(dag), mode="directed", diag=FALSE)
# Personalizar el gráfico
plot(g,
     vertex.label = V(g)$name,
```

```
     vertex.size = 15,
     vertex.label.cex = 0.8,
     vertex.label.color = "black",
     vertex.color = "lightblue",
     edge.arrow.size = 0.1,
     layout = layout_with_fr)

fitted = bn.fit(dag, data1_scaled,method="mle-g")

#Creamos las redes para cada una de las variables diferencia que hacen de frontera
#Cs
data2 <- cbind(covariables,datos_op1$Cs_dif)
names(data2)[names(data2) == "datos_op1$Cs_dif"] <- "Cs_dif"
data2 <- subset(data2, select = -c(IDM))
data2_scaled<-as.data.frame(apply(data2,2,standarize))
data2_scaled<-sapply(data2_scaled,as.numeric)
data2_scaled<-as.data.frame(data2_scaled)

pdag <- tabu(data2_scaled)
g <- graph_from_adjacency_matrix(amat(pdag), mode="directed", diag=FALSE)
# Personalizar el gráfico
plot(g,
     vertex.label = V(g)$name,
     vertex.size = 15,
     vertex.label.cex = 0.8,
     vertex.label.color = "black",
     vertex.color = "lightblue",
     edge.arrow.size = 0.1,
     layout = layout_with_fr)

fitted = bn.fit(pdag, data2_scaled,method="mle-g")

#UPL
data3 <- cbind(covariables,datos_op1$UPL_dif)
names(data3)[names(data3) == "datos_op1$UPL_dif"] <- "UPL_dif"
data3 <- subset(data3, select = -c(IDM))
data3_scaled<-as.data.frame(apply(data3,2,standarize))
data3_scaled<-sapply(data3_scaled,as.numeric)
data3_scaled<-as.data.frame(data3_scaled)

pdag <- tabu(data3_scaled)
g <- graph_from_adjacency_matrix(amat(pdag), mode="directed", diag=FALSE)
# Personalizar el gráfico
plot(g,
     vertex.label = V(g)$name,
     vertex.size = 15,
     vertex.label.cex = 0.8,
     vertex.label.color = "black",
     vertex.color = "lightblue",
```

```
     edge.arrow.size = 0.1,
     layout = layout_with_fr)

fitted = bn.fit(pdag, data3_scaled,method="mle-g")

#SY
data4 <- cbind(covariables,datos_op1$SY_dif)
names(data4)[names(data4) == "datos_op1$SY_dif"] <- "SY_dif"
data4 <- subset(data4, select = -c(IDM))
data4_scaled<-as.data.frame(apply(data4,2,standarize))
data4_scaled<-sapply(data4_scaled,as.numeric)
data4_scaled<-as.data.frame(data4_scaled)

pdag <- tabu(data4_scaled)
g <- graph_from_adjacency_matrix(amat(pdag), mode="directed", diag=FALSE)
# Personalizar el gráfico
plot(g,
     vertex.label = V(g)$name,
     vertex.size = 15,
     vertex.label.cex = 0.8,
     vertex.label.color = "black",
     vertex.color = "lightblue",
     edge.arrow.size = 0.1,
     layout = layout_with_fr)

fitted = bn.fit(pdag, data4_scaled,method="mle-g")
```

ANÁLISIS DE COMPONENTES PRINCIPALES

Análisis de Componentes Principales

```
library(ggplot2)
library(ggfortify)
# Cargar los datos de los resultados electorales para dos periodos
resultados_periodo1 <- read.csv("gruposCortes.csv", sep=';', fileEncoding="latin1", header =
TRUE)
resultados_periodo2 <- read.csv("gruposGen.csv", sep=';', fileEncoding="latin1", header = TRUE)

# Cargar los datos de las covariables
covariables <- read.csv("covariables.csv", sep=';', dec = ',', fileEncoding="latin1", na.strings
= '.',header = TRUE)

excluir_columnas <- c("IDM","Censo","Tvotos")

# Calcular porcentaje de votos por partido en cada periodo
resultados_periodo1_porcentaje <- resultados_periodo1[, !names(resultados_periodo1) %in%
excluir_columnas] /
  rowSums(resultados_periodo1[, !names(resultados_periodo1) %in% excluir_columnas]) * 100

resultados_periodo2_porcentaje <- resultados_periodo2[, !names(resultados_periodo2) %in%
excluir_columnas] /
  rowSums(resultados_periodo2[, !names(resultados_periodo2) %in% excluir_columnas]) * 100

# Extraer las columnas IDM, Censo y Tvotos de las tablas originales
columnas_adicionales <- resultados_periodo1[, c("IDM", "Censo", "Tvotos"), drop = FALSE]


opcionCs <- 1

if (opcionCs == 1) {
  #Opcion1: Ciudadanos como 0
  resultados_periodo2_porcentaje$Cs <- 0

  datos_op1 <- data.frame( #Lo definimos como generales-cortes
    VotosNulos_dif = resultados_periodo2_porcentaje$VotosNulos -
resultados_periodo1_porcentaje$Votos.Nulos,
    VotosBlanco_dif = resultados_periodo2_porcentaje$VotosBlanco -
resultados_periodo1_porcentaje$Votos.Blanco,
    Cs_dif = resultados_periodo2_porcentaje$Cs - resultados_periodo1_porcentaje$Cs,
    ESPAÑA.VACIADA_dif = resultados_periodo2_porcentaje$ESPAÑA.VACIADA -
resultados_periodo1_porcentaje$ESPAÑA.VACIADA,
    PODEMOS_SUMAR_dif = resultados_periodo2_porcentaje$SUMAR -
resultados_periodo1_porcentaje$PODEMOS, #Estandarizamos SUMAR como podemos por desigualdad+
similitud de voto
    PP_dif = resultados_periodo2_porcentaje$PP - resultados_periodo1_porcentaje$PP,
    PSOE_dif = resultados_periodo2_porcentaje$PSOE - resultados_periodo1_porcentaje$PSOE,
    SY_dif = resultados_periodo2_porcentaje$SY - resultados_periodo1_porcentaje$SY,
```

```
    UPL_dif = resultados_periodo2_porcentaje$UPL - resultados_periodo1_porcentaje$UPL,
    VOX_dif = resultados_periodo2_porcentaje$VOX - resultados_periodo1_porcentaje$VOX,
    XAV_dif = resultados_periodo2_porcentaje$XAV - resultados_periodo1_porcentaje$XAV,
    Abstencion_dif = resultados_periodo2_porcentaje$Abstencion -
resultados_periodo1_porcentaje$Abstencion,
    Partidos.Minoritarios_dif = resultados_periodo2_porcentaje$Partidos.Minoritarios -
resultados_periodo1_porcentaje$Partidos.Minoritarios
  )
  datos_op1<-cbind(columnas_adicionales,datos_op1)
} else {
  #Opcion2: Ciudadanos en Partidos Minoritarios
  #Lo añadimos a partidos minoritarios
  resultados_periodo1_porcentaje$Partidos.Minoritarios <-
resultados_periodo1_porcentaje$Partidos.Minoritarios + resultados_periodo1_porcentaje$Cs
  resultados_periodo1_porcentaje <- resultados_periodo1_porcentaje[,
!(names(resultados_periodo1_porcentaje) %in% "Cs")]

  #Ahora creamos el dataset
  datos_op1 <- data.frame( #Lo definimos como generales-cortes
    VotosNulos_dif = resultados_periodo2_porcentaje$VotosNulos -
resultados_periodo1_porcentaje$Votos.Nulos,
    VotosBlanco_dif = resultados_periodo2_porcentaje$VotosBlanco -
resultados_periodo1_porcentaje$Votos.Blanco,
    ESPAÑA.VACIADA_dif = resultados_periodo2_porcentaje$ESPAÑA.VACIADA -
resultados_periodo1_porcentaje$ESPAÑA.VACIADA,
    PODEMOS_SUMAR_dif = resultados_periodo2_porcentaje$SUMAR -
resultados_periodo1_porcentaje$PODEMOS, #Estandarizamos SUMAR como podemos por desigualdad+
similitud de voto
    PP_dif = resultados_periodo2_porcentaje$PP - resultados_periodo1_porcentaje$PP,
    PSOE_dif = resultados_periodo2_porcentaje$PSOE - resultados_periodo1_porcentaje$PSOE,
    SY_dif = resultados_periodo2_porcentaje$SY - resultados_periodo1_porcentaje$SY,
    UPL_dif = resultados_periodo2_porcentaje$UPL - resultados_periodo1_porcentaje$UPL,
    VOX_dif = resultados_periodo2_porcentaje$VOX - resultados_periodo1_porcentaje$VOX,
    XAV_dif = resultados_periodo2_porcentaje$XAV - resultados_periodo1_porcentaje$XAV,
    Abstencion_dif = resultados_periodo2_porcentaje$Abstencion -
resultados_periodo1_porcentaje$Abstencion,
    Partidos.Minoritarios_dif = resultados_periodo2_porcentaje$Partidos.Minoritarios -
resultados_periodo1_porcentaje$Partidos.Minoritarios
  )
  datos_op1<-cbind(columnas_adicionales,datos_op1)
}


# Asumiendo que los datos tienen una columna común como 'IDM' o similar para hacer la unión
datos_completos <- merge(datos_op1, covariables, by = "IDM", all.x = TRUE)
#Estandarizado de datos
standarize<-function(variable){
  return (variable-mean(variable))/sd(variable)
}
```

```r
datos_completos<-as.data.frame(apply(datos_completos,2,standarize))

#Quitamos IDM pues es solo el identificador, y Tvoto por correlacion 0.99 con Censo
datos_completos <- subset(datos_completos, select = -c(IDM,Tvotos))

if (opcionCs == 1) {
  #Guardamos el dataset completo para luego
  write.csv2(datos_completos, file = "C:/Users/oscat/Documents/6º/TFG Est/Análisis/4 - Analisis
Cluster General/datosCompletosOp1.csv", row.names = FALSE, fileEncoding = "latin1")
}
# Seleccionar las columnas numéricas para el PCA
datos_numericos <- datos_completos[, sapply(datos_completos, is.numeric)]

#Quitamos NAs poniendo las medias para no modificar el análisis
columnas_con_na <- colnames(datos_numericos)[apply(datos_numericos, 2, anyNA)]

for (col in columnas_con_na) {
  datos_numericos[[col]][is.na(datos_numericos[[col]])] <- mean(datos_numericos[[col]], na.rm =
TRUE)
}

# Aplicar PCA a las variables numéricas
pca_result <- prcomp(datos_numericos, center = TRUE, scale. = TRUE)

# Obtener los componentes principales y la varianza explicada
pca_components <- pca_result$rotation  # Componentes principales (vectores propios)
pca_varianza_explicada <- pca_result$sdev^2 / sum(pca_result$sdev^2)  # Varianza explicada por
cada componente


#Grafico screeplot de varianza explicada con ggplot2
componente <- 1:length(pca_varianza_explicada)
ggplot() +
  geom_col(aes(x = componente, y = pca_varianza_explicada), fill = "skyblue") +
  geom_point(aes(x = componente, y = pca_varianza_explicada), color = "red", size = 3) +
  geom_line(aes(x = componente, y = pca_varianza_explicada, group = 1), color = "red", size = 1)
+
  geom_text(aes(x = componente, y = pca_varianza_explicada, label = sprintf("%.2f%%",
pca_varianza_explicada * 100)),
            vjust = -0.5, size = 3) +
  labs(title = "Varianza explicada por componente principal",
       x = "Componente principal",
       y = "Proporción de varianza explicada") +
  ylim(0, max(pca_varianza_explicada) * 1.1) +
  theme_minimal()

# Calcular los autovalores (varianza explicada por cada componente)
autovalores <- pca_result$sdev^2
```

```r
# Calcular el porcentaje de varianza explicada por cada componente
porcentaje_varianza <- autovalores / sum(autovalores) * 100

# Calcular el porcentaje de varianza acumulada
porcentaje_varianza_acumulada <- cumsum(porcentaje_varianza)

# Imprimir los resultados en el formato específico
cat("Componente\n")
cat(1:length(autovalores), "\n\n")

cat("Autovalor\n")
cat(sprintf("%.2f", autovalores), "\n\n")

cat("%Varianza\n")
cat(sprintf("%.2f", porcentaje_varianza), "\n\n")

cat("%Varianza Acumulada\n")
cat(sprintf("%.2f", porcentaje_varianza_acumulada), "\n")

# Ver las cargas de las 6 primeras componentes principales
print(pca_components[, 1])
print(pca_components[, 2])
print(pca_components[, 3])
print(pca_components[, 4])
print(pca_components[, 5])
print(pca_components[, 6])

options(scipen = 999)
#Sacamos las contribuciones de las variables a PC1 y PC2.
loadings <- pca_result$rotation
contribuciones<-t(t(abs(loadings[, 1:6])) / rowSums(t(abs(loadings[, 1:6])))) * 100
round(contribuciones,4)
#Correlaciones y cosenos
correlaciones <- loadings[, 1:6] * pca_result$sdev[1:6]
round(correlaciones, 4)
cosenos <- correlaciones^2
round(cosenos,4)

#Composicion de los PC en graficos
contribuciones <- t(t(abs(loadings))/colSums(abs(loadings))) * 100

# Graficar las contribuciones de PC1
ggplot(as.data.frame(contribuciones), aes(x = reorder(rownames(contribuciones), -
contribuciones[, 1]), y = contribuciones[, 1])) +
  geom_bar(stat = "identity", fill = "skyblue") +
  coord_flip() +
  labs(title = "Contribuciones a PC1",
       x = "Variable",
       y = "Contribución (%)") +
```

```
  theme_minimal()

# Graficar las contribuciones de PC2
ggplot(as.data.frame(contribuciones), aes(x = reorder(rownames(contribuciones), -
contribuciones[, 2]), y = contribuciones[, 2])) +
  geom_bar(stat = "identity", fill = "coral") +
  coord_flip() +
  labs(title = "Contribuciones a PC2",
       x = "Variable",
       y = "Contribución (%)") +
  theme_minimal()

#Dimension 4 diferencia de derechas
# Graficar las contribuciones de PC4
ggplot(as.data.frame(contribuciones), aes(x = reorder(rownames(contribuciones), -
contribuciones[, 4]), y = contribuciones[, 4])) +
  geom_bar(stat = "identity", fill = "green") +
  coord_flip() +
  labs(title = "Contribuciones a PC4",
       x = "Variable",
       y = "Contribución (%)") +
  theme_minimal()


# Cálculo del círculo unidad
theta <- seq(0, 2 * pi, length.out = 100)
circle <- data.frame(x = cos(theta), y = sin(theta))

# Crear los gráficos PCA con ggplot2 para todas las combinaciones de PC
ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 1], yend = pca_result$rotation[,
2]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 1], y = pca_result$rotation[, 2], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC1", y = "PC2") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 1], yend = pca_result$rotation[,
3]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 1], y = pca_result$rotation[, 3], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC1", y = "PC3") +
```

```
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 1], yend = pca_result$rotation[,
4]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 1], y = pca_result$rotation[, 4], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC1", y = "PC4") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 1], yend = pca_result$rotation[,
5]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 1], y = pca_result$rotation[, 5], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC1", y = "PC5") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 1], yend = pca_result$rotation[,
6]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 1], y = pca_result$rotation[, 6], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC1", y = "PC6") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 2], yend = pca_result$rotation[,
3]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 2], y = pca_result$rotation[, 3], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC2", y = "PC3") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
```

```
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 2], yend = pca_result$rotation[,
4]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 2], y = pca_result$rotation[, 4], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC2", y = "PC4") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 2], yend = pca_result$rotation[,
5]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 2], y = pca_result$rotation[, 5], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC2", y = "PC5") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 2], yend = pca_result$rotation[,
6]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 2], y = pca_result$rotation[, 6], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC2", y = "PC6") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 3], yend = pca_result$rotation[,
4]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 3], y = pca_result$rotation[, 4], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC3", y = "PC4") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 3], yend = pca_result$rotation[,
5]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 3], y = pca_result$rotation[, 5], label =
rownames(pca_result$rotation)),
```

```
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC3", y = "PC5") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 3], yend = pca_result$rotation[,
6]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 3], y = pca_result$rotation[, 6], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC3", y = "PC6") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 4], yend = pca_result$rotation[,
5]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 4], y = pca_result$rotation[, 5], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC4", y = "PC5") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 4], yend = pca_result$rotation[,
6]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 4], y = pca_result$rotation[, 6], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC4", y = "PC6") +
  theme_minimal()

ggplot() +
  geom_path(aes(x = cos(theta), y = sin(theta)), color = "black", linetype = "dashed") +
  geom_segment(aes(x = 0, y = 0, xend = pca_result$rotation[, 5], yend = pca_result$rotation[,
6]),
               arrow = arrow(length = unit(0.2, "cm")), color = "blue") +
  geom_text(aes(x = pca_result$rotation[, 5], y = pca_result$rotation[, 6], label =
rownames(pca_result$rotation)),
            hjust = 1, vjust = 1, color = "blue", size = 3) +
  labs(title = "PCA Graph of Variables", x = "PC5", y = "PC6") +
  theme_minimal()
```

Gráfico 6 primeras componentes

```
library(ggplot2)

data <- datos_numericos

# Realizar PCA
pca_result <- prcomp(data, scale. = TRUE)

# Obtener las cargas factoriales (correlaciones)
cargas <- pca_result$rotation[, 1:6]

# Calcular el coseno al cuadrado de las variables
cos2 <- cargas^2

# Calcular las contribuciones (cargas al cuadrado dividido por la suma de las cargas al cuadrado de cada componente)
contribuciones <- sweep(cos2, 2, colSums(cos2), "/") * 100

# Crear un dataframe con las variables y sus medidas
variables <- rownames(cargas)
medidas <- data.frame(
  Variable = rep(variables, 6),
  Componente = rep(paste0("PC", 1:6), each = length(variables)),
  Contribucion = as.vector(t(contribuciones)),
  Correlacion = as.vector(t(cargas)),
  Coseno2 = as.vector(t(cos2))
)

# Generar gráficos
# Contribuciones
ggplot(medidas, aes(x = Variable, y = Contribucion, fill = Componente)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Contribuciones de las Variables a cada Componente Principal",
       x = "Variable",
       y = "Contribución (%)") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

# Correlaciones
ggplot(medidas, aes(x = Variable, y = Correlacion, fill = Componente)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Correlaciones de las Variables con cada Componente Principal",
       x = "Variable",
       y = "Correlación") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

# Coseno al cuadrado
ggplot(medidas, aes(x = Variable, y = Coseno2, fill = Componente)) +
  geom_bar(stat = "identity", position = "dodge") +
```

```
  labs(title = "Representación de las Variables en cada Componente Principal (Coseno^2)",
       x = "Variable",
       y = "Coseno al Cuadrado") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

#comparacion con Opcion2 (ejecutar despues de ejecutar Opcion 2 en componentesPrincipales.R)

# Nombre de la variable de interés
variable_of_interest <- "Partidos.Minoritarios_dif"

# Valores de contribución, correlación y representación
contribuciones <- (pca_result$rotation[, 1:6])^2
correlaciones <- contribuciones / rowSums(contribuciones)
representacion <- contribuciones / rowSums(contribuciones^2)

# Extraer los valores para la variable de interés
contribucion_var <- contribuciones[variable_of_interest, ]
correlacion_var <- correlaciones[variable_of_interest, ]
representacion_var <- representacion[variable_of_interest, ]

# Crear un data frame con los resultados
resultados <- data.frame(
  Componente = colnames(contribuciones),
  Contribucion = contribucion_var,
  Correlacion = correlacion_var,
  Representacion = representacion_var
)
resultados
```

ANÁLISIS CLÚSTER

Análisis Clúster

```
library(factoextra)
library(NbClust)
library(cluster)
library(ggplot2)
library(fpc)

# Cargar el dataset
data_scaled <- read.csv("datosCompletosOp1.csv", sep=';', dec=',', fileEncoding="latin1",
header=TRUE)

# Determinar el número óptimo de clusters para k-means
#fviz_nbclust(data_scaled, cluster::clara, method = "gap_stat") + labs(subtitle = "Gap Method")
#fviz_nbclust(data_scaled, cluster::clara, method = "silhouette") + labs(subtitle = "Silhouette
Method")
gap_results <- fviz_nbclust(data_scaled, cluster::clara, method = "gap_stat") + labs(subtitle =
"Gap Method")
gap_results

# Extraer los valores de la silueta
gap_results <- gap_results$data
# Extraer el número óptimo de clusters sugerido por la mayoría de los métodos
optimal_clusters <- maxSE(gap_results$gap, gap_results$SE.sim, method = "globalmax")
optimal_clusters <- as.numeric(optimal_clusters)
cat("Número óptimo de clusters:", optimal_clusters, "\n")

# Realizar k-medoids con el número óptimo de clusters
set.seed(123)
kmeans_result <- kmeans(data_scaled, centers = optimal_clusters, nstart = 25)
fviz_cluster(kmeans_result, data = data_scaled, geom = "point")

# Realizar clustering jerárquico con diferentes métodos de enlace
dist_matrix <- dist(data_scaled)

# Método Complete Linkage
hc_complete <- hclust(dist_matrix, method = "complete")
plot(hc_complete, main = "Dendograma - Complete Linkage")

# Método Average Linkage
hc_average <- hclust(dist_matrix, method = "average")
plot(hc_average, main = "Dendograma - Average Linkage")

# Método Single Linkage
hc_single <- hclust(dist_matrix, method = "single")
plot(hc_single, main = "Dendograma - Single Linkage")
```

```r
#Grafico Complete Linkage con borde en k optimo
hc_complete <- hclust(dist_matrix, method = "complete")
plot(hc_complete, main = "Dendograma - Complete Linkage")
rect.hclust(hc_complete, k = optimal_clusters, border = 2:4)

#Ahora comparamos los métodos hallando la conectividad, el índice de Dunn y el coeficiente de
Silueta de ambos

# Función para calcular conectividad personalizada
calculate_connectivity <- function(clusters, dist_matrix, k = 10) {
  connectivity <- 0
  n <- length(clusters)

  for (i in 1:n) {
    dist_i <- sort(dist_matrix[i, ])
    neighbors <- dist_i[2:(k + 1)]  # Excluir la distancia a sí mismo (primera en la lista)
    neighbor_indices <- as.numeric(names(neighbors))

    same_cluster <- clusters[i] == clusters[neighbor_indices]
    connectivity <- connectivity + sum(!same_cluster)
  }

  return(connectivity)
}

# Inicializar listas para almacenar resultados
results <- data.frame(
  clusters = 2:10,
  connectivity_kmeans = numeric(9),
  connectivity_hc_complete = numeric(9),
  dunn_kmeans = numeric(9),
  dunn_hc_complete = numeric(9),
  silhouette_kmeans = numeric(9),
  silhouette_hc_complete = numeric(9)
)

dist_matrix <- dist(data_scaled)

for (k in 2:10) {
  # K-means
  set.seed(123)
  kmeans_result <- kmeans(data_scaled, centers = k, nstart = 25)
  results$connectivity_kmeans[k-1] <- clValid::connectivity(data_scaled, cl =
kmeans_result$cluster)
  results$dunn_kmeans[k-1] <- fpc::cluster.stats(dist_matrix, kmeans_result$cluster)$dunn
  results$silhouette_kmeans[k-1] <- mean(cluster::silhouette(kmeans_result$cluster,
dist_matrix)[, "sil_width"])

  # Hierarchical clustering (Complete Linkage)
```

```
  hc_complete <- hclust(dist_matrix, method = "complete")
  cutree_complete <- cutree(hc_complete, k = k)
  results$connectivity_hc_complete[k-1] <- clValid::connectivity(data_scaled, cl =
cutree_complete)
  results$dunn_hc_complete[k-1] <- fpc::cluster.stats(dist_matrix, cutree_complete)$dunn
  results$silhouette_hc_complete[k-1] <- mean(cluster::silhouette(cutree_complete,
dist_matrix)[, "sil_width"])
}

# Verificar resultados
print(results)


# Data frame para conectividad
data_connectivity <- data.frame(
  Clusters = rep(2:10, 2),
  Conectividad = c(results$connectivity_kmeans, results$connectivity_hc_complete),
  Metodo = rep(c("K-means", "Jerárquico (Complete Linkage)"), each = 9)
)

# Gráfico de conectividad
ggplot(data_connectivity, aes(x = Clusters, y = Conectividad, color = Metodo)) +
  geom_line(size = 1.2) +
  geom_point(size = 2) +
  labs(x = "Número de Clusters", y = "Conectividad",
       title = "Conectividad: K-means vs Jerárquico (CL)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = c(1, 1),
        legend.justification = c(1, 1),
        legend.background = element_rect(fill = "white", color = "black"),
        legend.key = element_rect(fill = "white", color = "black")) +
  scale_color_manual(values = c("blue", "red"))



# Data frame para índice de Dunn
data_dunn <- data.frame(
  Clusters = rep(2:10, 2),
  Dunn = c(results$dunn_kmeans, results$dunn_hc_complete),
  Metodo = rep(c("K-means", "Jerárquico (Complete Linkage)"), each = 9)
)

# Gráfico de índice de Dunn
ggplot(data_dunn, aes(x = Clusters, y = Dunn, color = Metodo)) +
  geom_line(size = 1.2) +
  geom_point(size = 2) +
  labs(x = "Número de Clusters", y = "Índice de Dunn",
       title = "Índice de Dunn: K-means vs Jerárquico (CL)") +
```

```
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = c(1, 1),
        legend.justification = c(1, 1),
        legend.background = element_rect(fill = "white", color = "black"),
        legend.key = element_rect(fill = "white", color = "black")) +
  scale_color_manual(values = c("blue", "red"))



# Data frame para coeficiente de silueta
data_silhouette <- data.frame(
  Clusters = rep(2:10, 2),
  Silueta = c(results$silhouette_kmeans, results$silhouette_hc_complete),
  Metodo = rep(c("K-means", "Jerárquico (Complete Linkage)"), each = 9)
)

# Gráfico de coeficiente de silueta
ggplot(data_silhouette, aes(x = Clusters, y = Silueta, color = Metodo)) +
  geom_line(size = 1.2) +
  geom_point(size = 2) +
  labs(x = "Número de Clusters", y = "Coeficiente de Silueta",
       title = "Coef. Silueta: K-means vs Jerárquico (CL)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = c(1, 1),
        legend.justification = c(1, 1),
        legend.background = element_rect(fill = "white", color = "black"),
        legend.key = element_rect(fill = "white", color = "black")) +
  scale_color_manual(values = c("blue", "red"))


#Guardamos la variable resultado del complete linkage
data_scaled$Cluster <- cutree(hc_complete, k = optimal_clusters)
write.csv2(data_scaled, "C:/Users/oscat/Documents/6º/TFG Est/Análisis/5 - Regresión
Multivariante/data_results_clusters.csv", row.names = FALSE, fileEncoding = "latin1")
```

ANÁLISIS MULTIVARIANTE

Regresión Multinomial

```
library(ggplot2)
library(nnet)
library(tidyr)
library(broom)
library(dplyr)

data <- read.csv("data_results_clusters.csv", sep=';', dec=',', fileEncoding="latin1",
header=TRUE)

# Asegurarse de que la variable de clúster sea un factor
data$Cluster <- as.factor(data$Cluster)

# Definir la fórmula del modelo
#Usamos modelo multinomial pues el factor tiene mas de 2 opciones
modelo <- multinom(Cluster ~ Censo + VotosNulos_dif + VotosBlanco_dif + Cs_dif +
ESPAÑA.VACIADA_dif + PODEMOS_SUMAR_dif + PP_dif + PSOE_dif + SY_dif + UPL_dif + VOX_dif +
XAV_dif + Abstencion_dif + Partidos.Minoritarios_dif + Distribución.de.la.renta.P80.P20 +
Edad.media.de.la.población + Población + Porcentaje.de.población.española +
Tamaño.medio.del.hogar + Mediana.de.la.renta.por.unidad.de.consumo + Renta.neta.media.por.hogar
+ Renta.neta.media.por.persona + Fuente.de.ingreso..otras.prestaciones +
Fuente.de.ingreso..otros.ingresos + Fuente.de.ingreso..pensiones +
Fuente.de.ingreso..prestaciones.por.desempleo + Fuente.de.ingreso..salario + Índice.de.Gini,
data = data)

# Resumen del modelo
summary(modelo)

#Predecir usando el modelo
predicciones <- predict(modelo, newdata = data)
head(predicciones)

# Extraer coeficientes del modelo y convertir a data frame
coef_df <- as.data.frame(t(coef(modelo)))
coef_df$Variable <- rownames(coef_df)
coef_df <- gather(coef_df, key = "Outcome", value = "Estimate", -Variable)

# Crear un gráfico de barras de los coeficientes
ggplot(coef_df, aes(x = Variable, y = Estimate, fill = Outcome)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Coeficientes del Modelo",
       x = "Variable",
       y = "Estimación del Coeficiente") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```r
# Predecir probabilidades
predicciones_prob <- predict(modelo, newdata = data, type = "probs")
pred_df <- as.data.frame(cbind(data$Cluster, predicciones_prob))
colnames(pred_df) <- c("Cluster", levels(data$Cluster))
# Reestructurar datos para ggplot
pred_df_long <- pivot_longer(pred_df, cols = -Cluster, names_to = "Grupo", values_to =
"Probabilidad")

# Crear un gráfico de puntos de probabilidades predichas por grupo
ggplot(pred_df_long, aes(x = Cluster, y = Probabilidad, color = Grupo)) +
  geom_point(position = position_jitter(width = 0.2, height = 0)) +
  labs(title = "Probabilidades Predichas por Grupo",
       x = "Cluster",
       y = "Probabilidad") +
  theme_minimal()

# Calcular efectos marginales manualmente usando diferencias finitas
calcular_efectos_marginales <- function(modelo, data, variable, delta = 1e-5) {
  data_plus <- data
  data_minus <- data
  data_plus[[variable]] <- data[[variable]] + delta
  data_minus[[variable]] <- data[[variable]] - delta

  preds_plus <- predict(modelo, newdata = data_plus, type = "probs")
  preds_minus <- predict(modelo, newdata = data_minus, type = "probs")

  efecto_marginal <- (preds_plus - preds_minus) / (2 * delta)
  return(efecto_marginal)
}

variables <- colnames(data)[!colnames(data) %in% c("Cluster")]
efectos_marginales_list <- lapply(variables, function(var) {
  efectos <- calcular_efectos_marginales(modelo, data, var)
  efectos_df <- as.data.frame(efectos)
  efectos_df$Variable <- var
  efectos_df
})

efectos_marginales_df <- bind_rows(efectos_marginales_list)
efectos_marginales_long <- pivot_longer(efectos_marginales_df, cols = -Variable, names_to =
"Grupo", values_to = "Efecto Marginal")

# Crear un gráfico de barras de los efectos marginales por categoría de Cluster
ggplot(efectos_marginales_long, aes(x = Variable, y = `Efecto Marginal`, fill = Grupo)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Efectos Marginales por Categoría de Cluster",
       x = "Variable",
       y = "Efecto Marginal Promedio (AME)") +
  theme_minimal() +
```

```r
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Diagnóstico del modelo
log_lik <- logLik(modelo)
aic <- AIC(modelo)
diagnosticos <- data.frame(
  Diagnóstico = c("Log-Likelihood", "AIC"),
  Valor = c(log_lik, aic)
)
# Diagnóstico del modelo
log_lik <- logLik(modelo)
aic <- AIC(modelo)

# Crear un dataframe con los diagnósticos
diagnosticos <- data.frame(
  Diagnóstico = c("Log-Likelihood", "AIC"),
  Valor = c(log_lik, aic)
)
diagnosticos

# Crear un gráfico de barras de los diagnósticos
ggplot(diagnosticos, aes(x = Diagnóstico, y = Valor, fill = Diagnóstico)) +
  geom_bar(stat = "identity") +
  labs(title = "Diagnóstico del Modelo",
       x = "Diagnóstico",
       y = "Valor") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```