Universidad de Salamanca
Grado en Matemáticas

REVISIÓN DE MÉTODOS MULTIVARIANTES SUPERVISADOS Y NO SUPERVISADOS

Trabajo Fin de Grado

VNiVERSiDAD D SALAMANCA

Alumno: Pedro Ángel Fraile Manzano

Tutoras: Ana Belén Nieto Librero y Nerea González García

Salamanca, Julio de 2023

Índice general

Capítulo 1

Métodos Supervisados

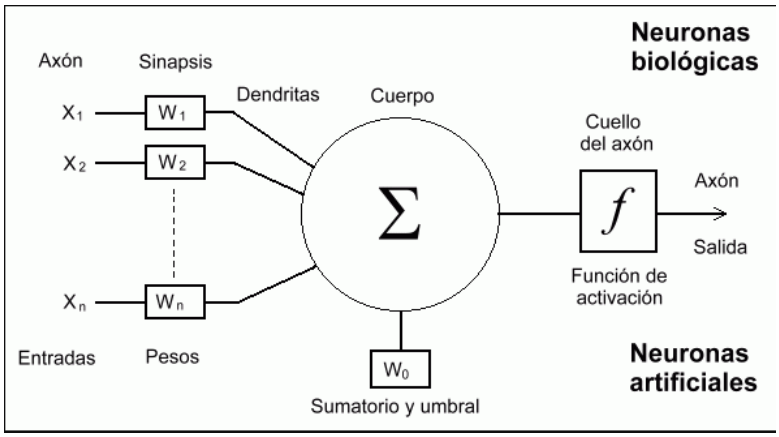
1.1. Introducción

1.2. Redes Neuronales

Las redes neuronales artificiales, redes neuronales simplemente a partir de ahora, se basan en el funcionamiento básico de las neuronas biológicas. Este tipo de células recogen señales externas, las procesan y producen una respuesta en consecuencia.

En el caso de una neurona artificial, dados los valores de entrada de una observación, $\{x_i\}_{i=1}^p$, la neurona tiene asociada a cada uno de ellos un peso, $\{\omega_i\}_{i=1}^p$, y opcionalmente un sesgo b. Además la neurona tiene una función de activación $g(x)$.

Una vez recogidos los valores, se hace una suma ponderada de ellos $z = b + \sum_{i=1}^p \omega_i x_i$ y la neurona produce el valor $g(z)$.



Una vez calculada la salida de la neurona, esta puede conectarse a otra, sirviendo los datos de salida de la primera neurona como datos de entrada de la siguiente. De manera podemos tener capas de m neuronas para producir m salidas que sirvan como datos de entrada para la siguiente capa de neuronas.

Por último, puede haber capas al principio y al final para tareas como estandarizar, centrar los datos y deshacer dichas operaciones para producir una salida coherente.

La siguiente imagen es un esquema de una red neuronal con 7 capas de neuronas interconectadas donde la primera capa es de escalado y la última de desescalado. Se puede observar que se predicen las variables y_1, y_2, y_3 usando como entrada las variables x_1, x_2, x_3, x_4

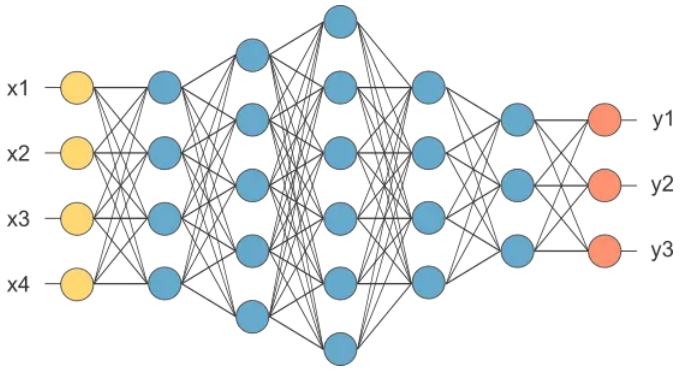


Figura 1.1: Imagen extraída directamente de www.neuraldesigner.com

Una vez detallado el proceso que se da dentro de una neurona hay que detallar los conceptos y elementos de la misma.

Definición 1.2.1. Se llama neurona a cada una de las transformaciones de los datos, $g_m(\omega_m^T \mathbf{x})$

Definición 1.2.2. Al vector ω_m se le denomina *vector de pesos* de la neurona, en particular, ω_{m0} se denomina *sesgo* de la neurona.

Definición 1.2.3. Se define una capa de neuronas como un conjunto de ellas que producen un vector de salida, ya sea para servir como entrada de una capa posterior o como salida final.

Por tanto, utilizando las conexiones entre neuronas y entre sucesivas capas tenemos un algoritmo de regresión por búsqueda de proyecciones mediante la composición y combinación de distintas neuronas con pesos y funciones de activación distintas.

1.2.1. Proyection Pursuit Regression

Sean un vector aleatorio $\mathbf{x}$ de longitud p, una variable objetivo Y y una familia de vectores de parámetros de longitud p, $\{\omega_m\}_{m=1}^M$. Entonces el modelo de Regresión por Búsqueda de Proyecciones *(PPR en inglés)* es de la forma :

$$f(\mathbf{x}) = \sum_{m=1}^{M} g_m(\omega_m^T \mathbf{x}) \tag{1.1}$$

En este modelo las funciones g_m no son especificadas y son estimaciones a lo largo de las direcciones de los vectores $\omega_m^T \mathbf{x}$. Según Hastie, Tibshirani y Friedman [4] para un M lo suficiente grande y utilizando las g_m apropiadas este modelo puede ayudar a la predicción de cualquier función continua real.

El uso de este modelo es complejo, debido a que hay una gran cantidad de parámetros de distintos tipos, obligando a utilizar distintos métodos de optimización. Asimismo, este tipo de modelos se utilizan solamente cuando se quiere predecir una variable, debido a su complejidad, que los hace difíciles de interpretar.

Sobre este modelo se sustentan las redes neuronales, que establecen previamente las funciones de activación g_m.

En el caso de las redes neuronales, el método de ajuste varía al modelo previamente descrito ya que las funciones están prefijadas

1.2.2. Red Neuronal de 2 capas

En pos de la sencillez de los resultados, se detallará la de una red neuronal de dos capas. El resto de casos $m \geq 2$, el proceso es análogo. Añadir que la situación es aquella en la que se quiere predecir K variables objetivo a partir de observaciones de p variables.

La primera capa tiene los siguientes elementos:

- Como datos de entrada cada una de las observaciones $\mathbf{x}$ de tamaño p e incluimos en cada uno el término inicial $x_0 = 1$ de tal manera que hay $p + 1$ datos de entrada.
- Un total de M unidades lo que provocará un conjunto $\{z_m\}$ de datos de salida.
- Cada unidad de las M tiene unos pesos α_m de dimensión $p + 1$, donde la primera componente α_{m0} se denomina **sesgo**.
- Una función de activación *(Que puede o no ser lineal)* $g^{(1)}$.

De esta manera, tenemos que los datos de salida de esta primera capa son de la forma:

$$z_m = g^{(1)}(\alpha_m^T \mathbf{x}) \tag{1.2}$$

De manera análoga tenemos una segunda capa de neuronas con los siguientes elementos:

- Como datos de entrada los M resultados de la capa anterior, z_m al que añadimos $z_0 = 1$ y denotaremos como el vector $\mathbf{z}$ de longitud $M + 1$.
- Un total de K unidades lo que provocará un conjunto $\{t_k\}$ de datos de salida.
- Cada unidad de las K, tiene unos pesos β_m de dimensión $M + 1$ igual que en la anterior.
- Una función de activación *(Que puede o no ser lineal)* $g^{(2)}$ que puede ser la misma que en la anterior o no.

De esta manera, tenemos que los datos de salida de esta primera capa son de la forma:

$$t_k = g^{(2)}(\beta_k^T \mathbf{z}) \tag{1.3}$$

Hay que destacar que los datos deben estar escalados, o pueden serlo mediante una capa que los escale per sé.

Definición 1.2.4. Sea una matriz de datos $\mathbf{X}$ de tamaño $n \times p$, resultado de observar las variables $X_1 \dots X_p$, se llama *capa de escalado* a la capa de neuronas con los siguientes elementos:

- Un vector de pesos con una sola componente igual a 1 y el resto nulas.
- Una función de activación $g(z) = \frac{z-\mu_i}{\sigma_i}$ donde μ_i, σ_i son las media y desviación típica muestrales de cada variable.

El resultado de que la matriz de datos sea procesada por este tipo de capa es una matriz $\mathbf{X}'$ centrada y estandarizada.

1.2.3. Ajuste y uso de una Red Neuronal

Una vez formulado el funcionamiento de una red neuronal, el ajuste de los pesos y sesgos de esta se puede plantear como un problema de minimización de la pérdida en función de los propios pesos, utilizando métodos de optimización numérica como el método del gradiente o el método de Quasi-Newton.

Hay que tener en cuenta para seleccionar el modelo, es decir, el número de neuronas y capas, que las redes neuronales son proclives al sobre ajuste. Esto es debido a la gran cantidad de parámetros que se pueden tener en una red con una complejidad media. Es por ello que normalmente se introduce una penalización a términos muy grandes y hace que los pesos sean parecidos a 0.

Una vez ajustados los pesos el modelo resultante puede ser complejo de interpretar, pero proporciona predicciones que potencialmente pueden ser todo lo precisas que se requieran según el número de capas y neuronas que utilicemos *[insertar el artículo de roberto]*.

En este trabajo se han detallado los tipos más simples de redes neuronales y de neuronas, con las cuales ya se obtiene una gran capacidad de predicción. Sin embargo, hay estructuras neuronales como las redes convolucionales o las capas LSTM *(Llamadas así por sus siglas en inglés Long-Short Term Memory)* que permiten modelizar sistemas en que los estados futuros son afectados por los estados anteriores como pudiera ser el precio de una acción bursátil o el tiempo atmosférico.

Capítulo 2

Métodos no supervisados

2.1. Introducción

2.2. Análisis de Componentes Principales

El análisis de componentes principales fue en primera instancia desarrollado a principios del siglo XX por el estadístico Pearson (1901). Fue una de las primeras técnicas de análisis multivariante. Como muchos otros métodos de este tipo no tuvo un verdadero desarrollo y expansión hasta que la capacidad de computación fue suficiente para manejar cantidades de datos considerables.

2.2.1. Definición y cálculo de las Componentes

Sea un vector aleatorio $\mathbf{x}^T = [X_1, \dots X_p]$ con vector de medias μ y matriz de covarianzas Σ.

Definición 2.2.1. Las componentes principales son combinaciones lineales de las variables $X_1 \dots X_p$

$$\mathbf{z}_j = a_{1j}X_1 + \dots a_{pj}X_p = \mathbf{a}_j^T\mathbf{x} \tag{2.1}$$

Donde $\mathbf{a}_j$ es un vector de constantes y la variable $\mathbf{z}_j$ cumple lo siguiente:

- Si $j = 1$ $Var(\mathbf{z}_1)$ es máxima restringido a $\mathbf{a}_1^T\mathbf{a}_1 = 1$
- Si $j > 1$ debe cumplir:
  - $Cov(\mathbf{z}_j, \mathbf{z}_i) = 0 \quad \forall i \neq j$
  - $\mathbf{a}_j^T\mathbf{a}_j = 1$
  - $Var(\mathbf{z}_j)$ es máxima.

De esta manera lo que se busca es una nueva base que reúna las direcciones de máxima variación

El cálculo de la primera componente principal se lleva a cabo con un proceso de optimización de la función $Var(\mathbf{z}_1)$ sujeto a la restricción de que $\mathbf{a}_1^T\mathbf{a}_1 = 1$.

Aplicando el método de los multiplicadores de Lagrange, dada una función $f(\mathbf{x}) = f(x_1, \ldots, x_p)$ diferenciable con una restricción $g(\mathbf{x}) = g(x_1, \ldots, x_p) = c$, existe una constante λ de manera que la ecuación:

$$\frac{\partial f}{\partial x_i} - \lambda \frac{\partial g}{\partial x_i} = 0 \quad i = 1, \ldots p \tag{2.2}$$

Tiene como solución los puntos estacionarios de $f(\mathbf{x})$. Además, si se define la función $L(\mathbf{x}) = f(\mathbf{x}) - \lambda[g(\mathbf{x}) - c]$ es posible simplificar la expresión anterior a:

$$\frac{\partial L}{\partial \mathbf{x}} = 0 \tag{2.3}$$

Para el caso de las componentes principales, la función objetivo es la varianza de la combinación lineal, es decir, $f(\mathbf{x}) = \mathbf{x}^T\Sigma\mathbf{x}$ y la restricción aplicada es $g(\mathbf{x}) = \mathbf{x}^T\mathbf{x} = 1$.

Tomando $\mathbf{x} = \mathbf{a}_1$ se puede establecer $L(\mathbf{a}_1) = \mathbf{a}_1^T\Sigma\mathbf{a}_1 - \lambda[\mathbf{a}_1^T\mathbf{a}_1 - 1]$. Que al derivarla se obtiene:

$$\begin{aligned}\frac{\partial L}{\partial \mathbf{a}_1} &= 2\Sigma\mathbf{a}_1 - 2\lambda\mathbf{a}_1 \\ &= 2(\Sigma - \lambda)\mathbf{a}_1\end{aligned}$$

Igualando a 0 tenemos la siguiente ecuación:

$$(\Sigma - \lambda I)\mathbf{a}_1 = 0 \tag{2.4}$$

Para que $\mathbf{a}_1$ sea un vector no trivial, se elige λ de tal manera que $|\Sigma - \lambda I| = 0$, es decir, λ es un vector propio de la matriz de covarianzas, Σ. Al ser ésta una matriz semidefinido positiva y simétrica, los valores propios son reales y positivos. Por tanto, $\mathbf{a}_1$ es un vector propio de la matriz de covarianza.

La función a maximizar es $Var(\mathbf{z}_1) = Var(\mathbf{a}_1^T\mathbf{x}) = \mathbf{a}_1^T\Sigma\mathbf{a}_1 = \mathbf{a}_1^T\lambda\mathbf{a}_1$, y para maximizarla basta tomar $\lambda = \text{máx}\{\lambda_1 \ldots \lambda_p\}$. reordenando si es necesario, se tiene que $\lambda = \lambda_1$

Una vez calculada la primera componente principal $\mathbf{z}_1$, la segunda componente se calcula de manera análoga, maximizando $Var(\mathbf{z}_2) = Var(\mathbf{a}_2^T\mathbf{x})$ condicionada por $\mathbf{a}_2^T\mathbf{a}_2 = 1$. A esta restricción tenemos que añadir la restricción $Cov(\mathbf{z}_1, \mathbf{z}_2) = 0$

Proposición 2.2.1. La condición $Cov(\mathbf{z}_1, \mathbf{z}_2) = 0$ equivale a la condición $\mathbf{a}_2^T\mathbf{a}_1 = 0$.

Demostración. Utilizando que $\mathbf{z}_j = \mathbf{a}_j^T\mathbf{x} \quad \forall j$, se tiene entonces que :

$$\begin{aligned}Cov(\mathbf{z}_2, \mathbf{z}_1) &= Cov(\mathbf{a}_2^T\mathbf{x}, \mathbf{a}_1^T\mathbf{x}) \\ &= \mathbb{E}(\mathbf{a}_2^T(\mathbf{x} - \mu)(\mathbf{x} - \mu)^T\mathbf{a}_1) \\ &= \mathbf{a}_2^T\mathbb{E}((\mathbf{x} - \mu)(\mathbf{x} - \mu)^T)\mathbf{a}_1 \\ &= \mathbf{a}_2^T\Sigma\mathbf{a}_1 \\ &= \mathbf{a}_2^T\lambda_1\mathbf{a}_1\end{aligned}$$

De manera que, si $a_2^T\lambda_1 a_1 = 0 \Rightarrow a_2^T a_1 = 0$, luego son vectores ortogonales entre sí. □

Observación: Esta proposición se puede extender de manera simple al caso de tener que calcular la i-ésima componente principal habiendo calculado las anteriores de las cuales se sepan los valores propios asociados.

Corolario 2.2.1. Las componentes principales son todas ortogonales entre sí.

Para $k = 2$, se dan dos restricciones, $\mathbf{a}_2^T\mathbf{a}_2 = 1$ y además $\mathbf{a}_1^T\mathbf{a}_2 = 0$. Para este caso existen λ, ϕ de manera que la función a maximizar es:

$$L(\mathbf{a}_2) = \mathbf{a}_2^T\Sigma\mathbf{a}_2 - \lambda[\mathbf{a}_2^T\mathbf{a}_2 - 1] - \phi(\mathbf{a}_1^T\mathbf{a}_2) \tag{2.5}$$

Que al ser derivado respecto $\mathbf{a}_2$ obtenemos:

$$2\Sigma\mathbf{a}_2 - 2\lambda\mathbf{a}_2 - \phi\mathbf{a}_1 = 0 \tag{2.6}$$

Que al multiplicar todo por $\mathbf{a}_1^T$ resulta que $\phi = 0$. De esta manera, se obtiene en la ecuación lo mismo que en el cálculo de la primera.

Por tanto, $\lambda = \lambda_2$ que es el segundo valor propio más grande, y $\mathbf{a}_2$ es el vector propio de valor propio λ_2.

Un proceso similar se puede seguir para calcular el resto de componentes principales.

Por ende, se obtiene que las componentes principales vienen dadas por los vectores propios de la matriz de covarianzas Σ. Además sabemos que $Var(\mathbf{a}_k^T\mathbf{x}) = \lambda_k$ donde λ_k es el k-ésimo valor propio más grande.

Sea ahora la matriz $\mathbf{A}$ cuyas columnas son los $\mathbf{a}_k$. Entonces el vector $\mathbf{z}$ que contiene a las componentes principales viene dado por la transformación:

$$\mathbf{z} = \mathbf{A}^T\mathbf{x} \tag{2.7}$$

Se deduce rápidamente que la matriz $\mathbf{A}$ es ortonormal, de manera que $\mathbf{A}^T\mathbf{A} = \mathbf{I}$

2.2.2. PCA en matrices de datos

Sea $\mathbf{x}$ el vector aleatorio de longitud p, tomando n observaciones de ese vector se obtienen $\mathbf{x}_1 \dots \mathbf{x}_n$. Esta recopilación de observaciones nos permite construir la matriz de datos $\mathbf{X}$ cuyas filas son cada una de las observaciones. Esta matriz $\mathbf{X}$ es de tamaño $n \times p$. Para este caso, las componentes principales se definen de manera análoga:

Definición 2.2.2. Dado un un vector aleatorio de longitud p del cual hemos extraído n observaciones se definen las componentes principales como:

$$\widetilde{z}_{ij} = \mathbf{a}_j^T \mathbf{x}_i \tag{2.8}$$

Donde $\mathbf{a}_j$ es un vector de constantes de longitud p que cumple lo siguiente:

- Si $j = 1$, entonces $\mathbf{a}_1$ maximiza la varianza de la muestra, es decir máximiza $\frac{1}{n-1}\sum_{i=1}^{n}(\widetilde{z}_{i1} - \overline{z}_1)^2$. Además debe cumplir que $\mathbf{a}_1^T\mathbf{a}_1 = 1$
- Si $j > 1$ debe cumplir:
  - Los vectores $\mathbf{a}_j$ son ortogonales entre sí.
  - $\mathbf{a}_j^T\mathbf{a}_j = 1$
  - La varianza muestral es máxima.

Es decir el factor $\widetilde{z}_{ij}$ es la transformación de la observación j-ésima por la i-ésima componente principal.

Por tanto, el proceso que se detalla para un vector aleatorio $\mathbf{x}$ con matriz de covarianzas Σ se puede extender a este caso en el conocemos la matriz de covarianzas muestrales $\mathbf{S}$.

Con el objetivo de hacer las demostraciones más sencillas y compactas tomaremos la matriz $\mathbf{X}$ como la matriz centrada $\overline{\mathbf{X}}$, es decir:

$$\overline{x}_{ij} = x_{ij} - \overline{x}_j \tag{2.9}$$

Donde $\overline{x}_j$ es la media muestral de la j-ésima variable. Esto hace que $\mathbf{S} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X}$. Esto permite hablar de los valores y vectores propios de $\mathbf{S}$ y de $\mathbf{X}^T\mathbf{X}$ indistintamente, ya que los vectores propios son los mismos y los valores propios son proporcionales.

2.2.3. Reducción de la dimensionalidad

Uno de los objetivos de las componentes principales es reducir la dimensionalidad de la matriz de datos de tamaño $n \times p$, $\mathbf{X}$. Esta matriz de datos se puede interpretar como un conjunto de puntos del espacio $\mathbb{R}^p$.

La intención final es buscar una proyección sobre una subvariedad de dimensión $m < p$ que reduzca la pérdida de información de la matriz y que brinde una mayor capacidad de interpretación de los datos, ya que en el caso de que $m = 2$ o $m = 3$ se podrán hacer representaciones gráficas de manera sencilla . En virtud de conseguir esto se deben definir los siguientes conceptos:

Definición 2.2.3. Dada una matriz $\mathbf{X} \in \mathbb{M}_{n\times p}(\mathbb{R})$ existe la descomposición en valores singulares *(SVD en inglés)*:

$$\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^T \tag{2.10}$$

Donde:

- $\mathbf{U}$ matriz ortogonal y de tamaño $n \times n$
- Σ matriz de tamaño $n \times p$ diagonal, cuyos elementos no nulos son los valores singulares $\sigma_1 \geq \ldots \geq \sigma_r \geq 0$ que son los valores propios de la matriz $\mathbf{X}^T\mathbf{X}$ y $r = rg(\mathbf{X})$
- $\mathbf{V}$ matriz ortogonal y de tamaño $p \times p$

Proposición 2.2.2. La matriz $\mathbf{V}$ de tamaño $(p \times p)$ es la matriz que contiene los vectores para hacer la combinación lineal que definen las componentes principales.

Demostración. La matriz $\mathbf{X}^T\mathbf{X}$ es la matriz de covarianzas $(p \times p)$ por la descomposición en valores singulares tenemos que:

$$\begin{aligned}\mathbf{X}^T\mathbf{X} &= (\mathbf{U}\Sigma\mathbf{V}^T)^T(\mathbf{U}\Sigma\mathbf{V}^T)\\ &= \mathbf{V}\Sigma^T\mathbf{U}^T\mathbf{U}\Sigma\mathbf{V}^T\\ &= \mathbf{V}\Sigma^T\Sigma\mathbf{V}^T\end{aligned}$$

Donde la matriz $\Sigma^T\Sigma$ es una matriz diagonal de tamaño $p \times p$ cuyos elementos son los cuadrados de los valores singulares de $\mathbf{X}$, que son a su vez los valores propios de $\mathbf{X}^T\mathbf{X}$.

Añadiendo la condición de ortogonalidad de $\mathbf{V} \Rightarrow \mathbf{V}^{-1} = V^T$ es fácil ver que la matriz $\mathbf{V}$ es la matriz cuyas columnas son los vectores propios de $\mathbf{X}^T\mathbf{X}$ □

Corolario 2.2.2. El cálculo de las componentes principales de la matriz de datos $\mathbf{X}$ es equivalente a calcular la descomposición en valores singulares de la misma.

Definición 2.2.4. Sea $\mathbf{A} \in \mathbb{M}_{n\times p}(\mathbb{R})$ definimos la *norma de Frobenius* de la matriz $\mathbf{A}$ como :

$$||\mathbf{A}||_F = (tr(\mathbf{A}^T \cdot \mathbf{A}))^{\frac{1}{2}} = \left(\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}^2\right)^{\frac{1}{2}} \tag{2.11}$$

Proposición 2.2.3. La norma de Frobenius es invariante a transformaciones ortogonales

Demostración. Sea $\mathbf{U}$ una matriz ortogonal, que cumple $\mathbf{U}^T \cdot \mathbf{U} = \mathbf{U} \cdot \mathbf{U}^T = \mathbf{I}$, sea una matriz cualquiera $\mathbf{A}$, entonces:

$$\begin{aligned} ||\mathbf{U} \cdot \mathbf{A}||_F^2 &= tr((\mathbf{UA})^T \cdot (\mathbf{UA})) \\ &= tr((\mathbf{A}^T\mathbf{U}^T) \cdot \mathbf{UA})) \\ &= tr(\mathbf{A}^T\mathbf{A}) \\ &= ||\mathbf{A}||_F^2 \end{aligned}$$

□

Se ha elegido la norma de frobenius se ha elegido por la siguiente propiedad.

Proposición 2.2.4. Dada una matriz de datos $\mathbf{X}$ de tamaño $n \times p$ entonces

$$||\mathbf{X}||_F^2 = (n-1)\sum_{i=1}^{p} s_{ii}^2 \tag{2.12}$$

Donde las s_{ii}^2 son las varianzas muestrales.

Demostración. Debido a la centralidad impuesta a la matriz $\mathbf{X}$, sabemos que la matriz de covarianzas es $\mathbf{S} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X}$ por tanto,se tiene que utilizar la definición de la norma:

$$\begin{aligned} ||\mathbf{X}||_F^2 &= tr(\mathbf{X}^T\mathbf{X}) \\ &= (n-1)tr(\mathbf{S}) \\ &= (n-1)\sum_{i=1}^{p} s_{ii}^2 \end{aligned}$$

□

Por tanto, la norma de Frobenius da una imagen del tamaño de la matriz de datos en función de la varianza total de los datos, lo que concuerda con la idea de buscar una matriz que aproxime la matriz de datos con la mínima pérdida de variación de los datos.

Definición 2.2.5. Se llama matriz reducida de orden $m \leq p$ de $\mathbf{X}$ y se denota como $\mathbf{X}_m$, a la matriz $n \times p$ resultado de:

$$\mathbf{X}_m = \mathbf{U}_m\Sigma_m\mathbf{V}_m^T \tag{2.13}$$

Donde:

- $\mathbf{U}_m$ matriz ortogonal de tamaño $n \times m$, resultado de tomar de $\mathbf{U}$ únicamente la matriz las m primeras columnas.

- Σ_m matriz cuadrada de tamaño m diagonal con los m primeros valores singulares.
- $\mathbf{V}_m$ matriz ortogonal de tamaño $p \times m$ obtenida al tomar las m primeras columnas de $\mathbf{V}$.

Teorema 2.2.1 (De Eckart-Young)**.** Sea $\mathbf{A}$ una matriz de coeficientes reales de tamaño $n \times p$ y rango r entonces se cumple que:

$$||\mathbf{A} - \mathbf{B}||_F \leq ||\mathbf{A} - \mathbf{A}_m||_F \quad \forall \mathbf{B}/rg(\mathbf{B}) = m \leq r \tag{2.14}$$

Por tanto, la matriz reducida brinda la mejor aproximación de la matriz de datos teniendo un criterio de aproximación basado en la variación de los datos. En consecuencia se puede

Como conclusión, se puede definir un criterio para elegir el orden de la matriz reducida m. Se puede entonces definir la variación acumulada de la siguiente manera:

$$t_m = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{p} \lambda_i} \tag{2.15}$$

Donde los λ_i son los valores propios de la matriz $\mathbf{S} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X}$.

Cuanto más cercana sea t_m a 1 manteniendo m lo más pequeña posible mejor por que implicaría que con pocas componentes principales podemos "explicar" la mayor parte de la variación de los datos. Según Jollife [5] entre un 0,8 y 0,9 es lo más habitual.

2.3. Análisis de Clusters

Según [7] el análisis de clusters tiene como objetivo conseguir una clasificación de las observaciones en subconjuntos que tengan sentido de acuerdo al contexto de la investigación. El clustering es un tipo de clasificación de los datos que tiene las siguientes características:

- *Exclusividad:* Una observación no puede pertenecer a más de un cluster a la vez.
- *Es intrínseco:* No hay ninguna etiqueta que permita clasificar las observaciones, son las características de las propias muestras las que servirán como diferenciadores. Lo que hace que sea un método *no supervisado*.

Además de estas características el clustering puede ser jerárquico *(concepto que se desarrollará más adelante)* o particional.
Por otro lado, los algoritmos se pueden clasificar según sean *aglomerativos*, en los que se empieza teniendo cada una de las observaciones y se van formando los clusters hasta tener un solo cluster con todas las muestras. Por el contrario, los *algoritmos divisivos*, empiezas con un solo cluster y se van haciendo subconjuntos del mismo.

Otra clasificación puede venir dada por el modo en el que se fijan en las variables, si es *Monotético*, en cada paso se centra en una variable, mientras que si es *Politético* se utilizan todas las variables a la vez.

Con el objetivo de formalizar lo anterior, de ahora en adelante, sea el caso en el que se han tomado n observaciones $\omega_1, \ldots, \omega_n$, se denota como $\Omega = \{\omega_1, \ldots, \omega_n\} = \{1, \ldots, n\}$ para abreviar.

Crear una clasificación entre las observaciones de Ω es establecer una relación de equivalencia $\mathcal{R}$ sobre Ω. De esta manera, podemos establecer que:

$$\Omega = \bigsqcup_{i=1}^{m} c_i \tag{2.16}$$

Donde c_i son las clases de equivalencia de la relación $\mathcal{R}$.

Definición 2.3.1. Se llama *clustering* a la partición que provoca la relación $\mathcal{R}$ y se definen los *clusters* como las clases de equivalencia de dicha relación, $\{c_i\}$.

2.3.1. Estructuración jerárquica de Ω

En un proceso de clustering jerárquico se crea una sucesión de particiones sobre el mismo conjunto donde los sucesivos clusterings se obtienen agrupando clusters.

Para poder formalizar el clustering jerárquico debemos definir el concepto de *jerarquía indexada*

Definición 2.3.2. Una *jerarquía indexada* (C, α) sobre un conjunto Ω, es una colección de clusters $C \subset \mathcal{P}(\Omega)$ y un índice que cumplen:

- *Intersección.* Si $c, c' \in C$ entonces $c \cap c' \in \{c, c', \emptyset\}$. Es decir, dos clusters o están contenido el uno en el otro o son disjuntos.
- *Reunión* Cada cluster se puede caracterizar como la unión de los clusters que contiene. Es decir, $c \in C \Rightarrow c = \cup\{c'/c' \in C, \quad c' \subset c\}$
- El conjunto de todos los clusters es el conjunto total.

El índice es una aplicación $\alpha : C \to \mathbb{R}$ que cumple lo siguiente :

$$\alpha(i) = 0, \forall i \in \Omega \qquad \alpha(c) \leq \alpha(c') \text{ si } c \subset c' \tag{2.17}$$

De esta manera, el índice puede medir como de heterogéneos son los clusters, es decir, puede otorgar una medida de lo similares o no que son las observaciones dentro de un mismo cluster.

Proposición 2.3.1. Para todo $x \geq 0$ la relación $\mathcal{R}_x$ sobre Ω:

$$i\mathcal{R}_x j \quad \text{si} \quad i, j \in c \quad \text{siendo} \quad \alpha(c) \leq x \tag{2.18}$$

Es una relación de equivalencia.

Es decir, con esta relación se establece que dos elementos son "iguales" cuando hay un cluster de un nivel de heterogeneidad menor que un umbral que los contiene. Es decir, se fija un criterio por el cual dos observaciones son lo suficientemente similares.

Definición 2.3.3. Un espacio ultramétrico es una pareja (Ω, u) donde Ω es un conjunto finito y u una función distancia sobre $\Omega \times \Omega$ que verifica para cada elemento de $i, j, k \in \Omega$

- $u(i, j) \geq u(i, i) = 0$
- $u(i, j) = u(j, i)$
- *Propiedad ultramétrica:* $u(i, j) \leq \sup\{u(i, k), u(j, k)\}$

Se puede ver que toda distancia ultramétrica cumple la desigualdad triangular. Por tanto, todo espacio ultramétrico es métrico. Cómo añadido, juntando los elementos próximos de Ω se mantiene la propiedad ultramétrica.

Teorema 2.3.1. Supongamos un clustering con m clusters de $\Omega = \bigsqcup_{i=1}^{m} c_i$ sobre el que se tiene una distancia ultramétrica u.
Tomando c_i, c_j los dos clusters más cercanos y uniéndolos podemos definir una nueva distancia u' sobre los $m-1$ clusters nuevos.

Demostración. Sea $k \neq i, j$ por la propiedad ultramétrica se tiene que:

$$\begin{aligned} u(i,k) &\leq sup\{u(i,j), u(j,k)\} = u(j,k) \\ u(j,k) &\leq sup\{u(i,j), u(i,k)\} = u(i,k) \end{aligned} \tag{2.19}$$

En consecuencia, $u(i,k) = u(j,k)$ y por tanto, $u(c_k, c_i) = u(c_k, c_j)$.
Definiendo de la distancia ultramétrica u':

$$\begin{aligned} u'(c_k, c_i \cup c_j) = u(c_k, c_i) = u(c_k, c_j) \quad k \neq i, j \\ u'(c_a, c_b) = u(c_a, c_b) \quad a, b \neq i, j \end{aligned} \tag{2.20}$$

Con esta nueva distancia se cumple la propiedad ultramétrica. Para elementos c_a, c_b con $a, b \neq i, j$ es evidente, pues no hemos cambiado la definición de la distancia para esos clusters. Ahora sea el siguiente caso:

$$\begin{aligned} u'(c_a, c_i \cup c_j) = u(c_a, c_i) &\leq \sup\{u(c_a, c_b), u(c_b, c_i)\} = \\ &= \sup\{u'(c_a, c_b), u'(c_b, c_i \cup c_j)\} \end{aligned} \tag{2.21}$$

Por tanto, haciendo uniones entre los elementos más cercanos no modifica la estructura de espacio ultramétrico.

□

Dado estos resultados se puede seguir el siguiente procedimiento para obtener una jerarquía indexada.

Algoritmo fundamental de clasificación

Dado un espacio ultramétrico (Ω, u) utilizando el teorema anterior se pueden ir juntando los clusters más próximos conservamos la distancia ultramétrica.

---

1. Se empieza con el clustering $\Omega = \{1\} \sqcup \ldots \sqcup \{n\}$.
2. Calculamos todas las distancias $u(c_a, c_b)$ y unimos los clusters que más cercanos estén utilizando la misma distancia ultramétrica como en el teorema anterior.
3. Se considera la nueva partición

$$\Omega = \{1\} \sqcup \ldots \sqcup \{i, j\} \sqcup \ldots \sqcup \{n\} \tag{2.22}$$

   Cada vez que se une c_i con c_j se define el índice $\alpha(c_i \cup c_j) = u(c_i, c_j)$.

Este procedimiento nos brinda una forma sencilla de dada una distancia ultramétrica construir una jerarquía indexada.

En el caso contrario en el que se disponga de una jerarquía indexada, se puede crear un espacio ultramétrico con la siguiente proposición.

Proposición 2.3.2. Dado un conjunto Ω finito sobre el que tenemos una jerarquía indexada (C, α) podemos establecer sobre Ω una estructura de espacio ultramétrico.

Demostración. Definiendo la distancia

$$u(i,j) = \alpha(c_{ij}) \tag{2.23}$$

Donde c_{ij} es el mínimo cluster que contiene a ambos i, j. Sean ahora los clusters mínimos que contienen a $\{i,k\}, \{j,k\}, c_{ik}, c_{jk}$ respectivamente, entonces:

$$c_{ik} \cap c_{jk} \neq \emptyset \tag{2.24}$$

Por ser una jerarquía indexada tenemos dos posibilidades de inclusión uno en el otro:

$$\begin{cases} c_{ik} \subset c_{jk} \Rightarrow i,j,k \in c_{jk} \Rightarrow c_{ij} \subset c_{jk} \Rightarrow u(i,j) = \alpha(c_{ij}) \leq u(j,k) = \alpha(c_{jk}) \\ c_{jk} \subset c_{ik} \Rightarrow i,j,k \in c_{ik} \Rightarrow c_{ij} \subset c_{ik} \Rightarrow u(i,j) = \alpha(c_{ij}) \leq u(i,k) = \alpha(c_{ik}) \end{cases} \tag{2.25}$$

Por tanto, se obtiene que $u(i,j) \leq \sup\{u(i,k), u(j,k)\}$

□

2.3.2. Algoritmos basados en jerarquías

Sea ahora una matriz de datos $\mathbf{X}$ resultante de hacer n observaciones sobre p variables aleatorias. Sea una distancia δ *(Se detallará más adelante como elegirla en función del tipo de datos que tengamos.)* sobre el espacio de observaciones y Δ la matriz $\Delta = (\delta(i,j))$ de tamaño $n \times n$, la matriz que contiene todas las distancias entre observaciones.

En el caso de que δ sea ultramétrica se puede utilizar el algoritmo fundamental, en el caso contrario hay que realizar una modificación.

Algoritmo de clasificación de observaciones

Lo que se busca en este tipo de algoritmos es convertir δ en una distancia últramétrica, por ende dado el espacio métrico (Ω, δ) el algoritmo es el siguiente:

---

1. Se empieza con el clustering $\Omega = \{1\} \sqcup \ldots \sqcup \{n\}$.
2. Calculamos todas las distancias $u(c_a, c_b)$ y unimos los clusters que más cercanos estén y se define la distancia de un elemento k al cluster de la siguiente manera:
$$\delta(k, \{i,j\}) = f(\delta(i,k), \delta(j,k)) \tag{2.26}$$
Donde la función f hace que δ cumpla la propiedad ultramétrica
3. Se considera la nueva partición
$$\Omega = \{1\} \sqcup \ldots \sqcup \{i,j\} \sqcup \ldots \sqcup \{n\} \tag{2.27}$$
Cada vez que se une c_i con c_j se define el índice $\alpha(c_i \cup c_j) = \delta'(c_i, c_j)$.

De esta manera se obtiene también una jerarquía indexada (C, α)

Dependiendo de la función f que se elija, se tienen distintos algoritmos:

- Si $f(x, y) = min(x, y)$ se obtiene el algoritmo de *single linkage*.
- Si $f(x, y) = min(x, y)$ se obtiene el algoritmo de *complete linkage*.
- Si $f(x, y) = \dfrac{x + y}{2}$ se obtiene el algoritmo de *average linkage*.

Es decir, tomando distintas f obtenemos distintos algoritmos. En estos casos, los algoritmos todos son algoritmos aglomerativos.

2.3.3. Elección de las diferencias

Añadir a esto que el mayor problema que conllevan este tipo de algoritmos es elegir y definir una distancia. Sea un vector aleatorio $\mathbf{x}$ de longitud p sobre el que se realizan n observaciones, entonces la forma más habitual de definir la distancia entre dos observaciones es:

$$D(x_i, x_j) = \sum_{k=1}^{p} \omega_k \cdot d_k(x_{ik}, x_{jk}); \quad \sum_{k=1}^{p} \omega_k = 1 \tag{2.28}$$

Donde las d_k son las distancias adaptadas al tipo de variable que sea la variable aleatoria X_k. En particular:

***Variables Cuantitativas*:** Basta con que sea una función monótona creciente del valor absoluto de la diferencia entre las dos observaciones. De manera general, se utilizaría la distancia euclídea, pero esta tiene el problema de que no tiene en cuenta las escalas de los datos es por eso que tenemos que estandarizar y centrar los datos para que no haya problemas de escalas. Una medida de distancia que suple esto es la distancia de Mahalanobis.

Definición 2.3.4. Dadas dos muestras u observaciones $\mathbf{x}_i, \mathbf{x}_j$ sacadas de la misma población de un total de n de las cuales se observan p variables aleatorias, con matriz de covarianzas muestrales $\mathbf{S}$. Se define la distancia de Mahalanobis de la siguiente manera:

$$\mathbf{M}^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i, \mathbf{x}_j)^T \mathbf{S}^{-1} (\mathbf{x}_i, \mathbf{x}_j) \tag{2.29}$$

Variables Ordinales: Para una variable categórica cuyos M valores tienen un orden se puede hacer la siguiente transformación para tratarlas como una variable cuantitativa:

$$\frac{i - \frac{1}{2}}{M} \quad i = 1 \dots M \tag{2.30}$$

Variables Categóricas: En este caso, suponiendo que la variable tiene M posibles valores, lo habitual es definir una matriz $\mathbf{L}$ de tamaño $M \times M$ simétrica , de manera que el elemento $L_{r,r'} = L_{r',r}$ es la distancia entre la categoría r y la r' que habitualmente se suele tomar como 1 y la diagonal igual a 0.

De esta manera, podemos definir de manera general una distancia entre las observaciones.

2.3.4. Algoritmos no jerárquicos

Teniendo en cuenta como deben ser los clusterings el objetivo es conseguir g clusters que provocan una partición exhaustiva del conjunto de observaciones. Sea el espacio métrico (Ω, d) se puede considerar que la *dispersión total de las muestras* dada un clustering con g clusters tiene la siguiente expresión:

$$\mathbf{T} = \mathbf{B} + \mathbf{W} \tag{2.31}$$

Donde B es la distancia inter-clusters y W es la distancia intra-cluster del clustering realizado.

Por tanto, de acuerdo a nuestro objetivo un clustering en g clusters debe ser aquel que minimice W y maximice B

El número de posibles clusterings de un mismo conjunto de n observaciones es de $\frac{g^n}{g!}$. Hay muchas de esas posibles particiones que no son óptimas y hay algunas que son mínimos locales. El algoritmo más habitual es el de K-medias o K-medoides.

Algoritmo de K-medias y K-medoides
Este algoritmo requiere que se prefije el número de clusters que se van a utilizar, además de que las variables sean cuantitativas,*(El algoritmo de K-medoides es una modificación de este que soluciona dicha restricción)* .

---

1. Se inicializa con g puntos aleatorios del espacio $\mathbb{R}^p$. Se asigna a cada observación un cluster de acuerdo a cual de los g puntos es más cercano.

2. Se calculan los puntos medios de cada cluster y se reasignan las observaciones de acuerdo a cual está más próximo

3. Se repite el paso anterior, calculando W en cada paso y se detiene el proceso cuando no haya mejora ninguna o se de un criterio de parada.

En este algoritmo ocurre que la distancia al cuadrado de los puntos de cada cluster al centroide disminuye en cada paso debido a como está formulado.

Además del problema de elegir el número de clusters, se da el problema que sólo se puede utilizar con variables cuantitativas. Haciendo una modificación en el algoritmo de K-medias obtenemos el algoritmo de K-medoides.

---

1. Para una asignación de clusters se calculan las distancias entre los puntos de cada cluster. Y se toma de cada cluster el que minimice la suma las distancias del cluster. Que serán los medoides de cada cluster.

2. Se asigna a cada punto el cluster cuyo medoide sea más cercano a dicha observación

3. Se repiten los pasos anteriores hasta que las asignaciones no cambien.

Por tanto, los puntos de referencia de este algoritmo son observaciones, no como las medias que pueden no serlo. Además es mucho más costoso a nivel computacional.

Adicionalmente, este es uno de los algoritmos afectados por la maldición de la dimensionalidad en la que al aumentar la dimensión del espacio de posibles observaciones cada vez es menos denso en información y necesitamos más muestras.

Bibliografía

[1] **Abdi, H., y Williams, L. J. (2010).** *Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics* 2(4), 433–459.

[2] **Chatfield,C y Collins A.J (1989)**. *Introduction to multivariate analysis*, Chapman and Hall.

[3] **Cuadras, C.M. (2014)**, *Nuevos métodos de Análisis Multivariante*, CMC Editions, Barcelona.

[4] **Hastie, T., Tibshirani, R. y Friedman J. (2001)**, *The Elements of Statistical Learning, Data Mining, Inference and Prediction* Springer

[5] **Jollife I.T.(1986)**. *Principal Component Analysis*, Springer-Verlag.

[6] Neural Designer `https://www.neuraldesigner.com/`

[7] **Jain, A.K., Richard, C.D., (1988)** *Algorithms for clustering data.* Prentice-Hall, Inc.

[8] *Scikit Learn Documentation, Clustering Methods.* `https://scikit-learn.org/stable/modules/clustering.html#clustering`