UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE CIENCIAS

ESCUELA PROFESIONAL DE MATEMÁTICA

SEMINARIO DE TESIS DE MATEMÁTICA PURA Y APLICADA I

MACHINE LEARNING

y una aplicación en

Reconocimiento Facial.

Un proyecto presentado por Anthony Enrique Huertas Quispe

Supervisado por:
Edgard Kenny Venegas Palacios

Lima, Perú
2016

Este trabajo esta dedicado
en especial a mi madre quien
confía en mis éxitos profesionales.

Índice general

Introducción

En la actualidad, gran parte de la tecnología está siendo desarrollada mediante procesos autónomos inteligentes, es decir que se están diseñando programas computacionales o sistemas que adquieren un aprendizaje automatizado (Machine Learning) lo cual permite que posteriormente se realicen análisis mucho más inteligentes para una tarea específica. Por este hecho, es que Machine Learning se ha vuelto necesario para la tecnológia de información pues con uso de técnicas estadísticas, optimización, aplicaciones del álgebra lineal permite que un proceso aprenda de sí mismo y elabore mecanismos que desarrollen trabajos complejos en el campo de la programación.

Maching Learning, o en latín *aprendizaje automatizado*, tiene una serie de aplicaciones como la de reconocimiento facial, huellas digitales o de escritura, procesamiento de lenguaje además de desarrollarse en medicina, ingeniería y muchas más disciplinas.

En este trabajo, el aprendizaje automatizado será definido desde varios puntos de vista por diferentes autores; luego se darán a conocer técnicas y algoritmos de aprendizaje que se implementan para el desarrollo de modelos de un tipo básico de inteligencia artificial, en base a conceptos matemáticos que serán vistos. Posteriormente nuestro enfoque se centrará en técnicas de reducción de dimensionalidad, como lo es el análisis de componentes principales, con el fin de entender y analizar los procesos que se llevan a cabo en un aplicación directa del aprendizaje automatizado.

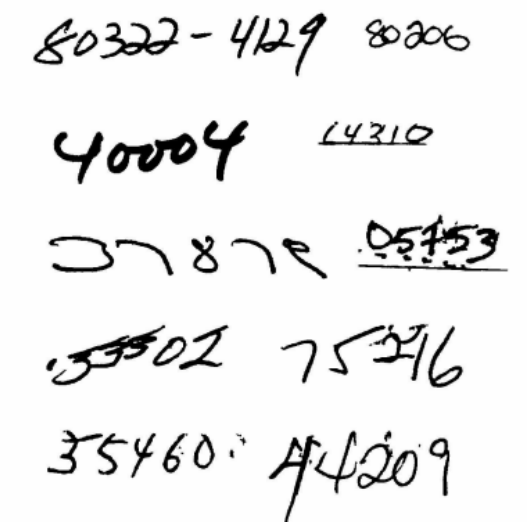


Figura 1: Reconocimiento de escritura

En el capitulo final, se dará a conocer la importancia del reconocimiento facial y como se ve aplicado el aprendizaje automatizado con el objeto de diseñar un sistema que permita procesar en forma óptima el desarrollo de dicha tarea mediante la definición de características significantes denominadas **eigenfaces**. La elaboración de este sistema se verá facilitada por la implementación del análisis de componentes principales (PCA) correspondiente a la reducción de dimensionalidad que a su vez está clasificado como un aprendizaje no supervisado, conjuntamente con una técnica de descomposición matricial (SVD) lo cual se definirá posteriormente; he ahí la importancia capítulo a capítulo en este trabajo.

Figura 2: Reconocimiento facial

Capítulo 1

Machine Learning

En esta sección, el objetivo será comprender el significado de **Machine Learning**, o aprendizaje automatizado, tanto desde una visión matemática como computacional. En primera instancia, Christopher Bishop, subgerente en el laboratorio de investigación de Microsoft en Cambridge, enunció lo siguiente:
"Pattern recognition has its origins in engineering, whereas machine learning grew out of computer science. However, these activities can be viewed as two facets of the same field...", en español, *"Los patrones de reconocimiento tienen sus orígenes en la ingeniería, mientras que el aprendizaje automatizado proviene de la ciencia de la computación. Sin embargo, estas actividades pueden ser vistas como dos facetas de un mismo campo..."*
lo cual nos da a entender que, desde su perspectiva, los métodos que se desarrollaron en la ciencia de la computación aprovechan métodos semejantes a los que utiliza la ingeniería y viceversa; sin embargo, si la Informática o tecnología de la información satisface nuestro requerimiento eso es llamado aprendizaje automatizado.

Por otra parte, Arthur Samuel, pionero americano en el campo de aprendizaje automatizado, lo define como:
"Field of study that gives computers the ability to learn without being explicitly programmed", en español, *"Campo de estudio que da a las computadoras la habilidad de aprender sin ser programadas explícitamente."*
Arthur inició estos estudios cuando decidió enseñar a una máquina llamada *Defense Calculator* a jugar el famoso "*Juego de Damas*"; el análisis implementado fue de tal modo que el sistema no necesite calcular el número de jugadas posibles y optar por la mejor, pues resultaría una complejidad, sino que logre hacer un conteo de piezas y mediante instrucciones básicas ejecute la que genere como resultado más piezas de ventaja sobre el adversario; esto revolucionó el razonamiento científico pues hoy en día un aprendizaje automatizado facilita la complejidad de un resultado.

La idea que se opta hoy en día es la que fue enunciada por Tom Mitchell sobre el

problema de aprendizaje bien planteado:
"*A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E*", en español, "*Se dice que un programa de computadora aprende de una experiencia E con respecto a alguna tarea T y una medida de rendimiento P, si el rendimiento en la tarea T, con uso de la medida P, mejora con la experiencia E.*"
Un ejemplo sencillo, es cuando un sistema aprende como filtrar correos Spam, en este caso la experiencia E sería la observación sobre los correos que uno como usuario decidió marcarlas como tal, la tarea T del sistema sería la de clasificar correos como Spam o no en base a la experiencia E, y la medida de rendimiento P vendría a ser el número de correos que correctamente han sido clasificados como Spam.
En este capítulo, daremos a conocer algunas técnicas de aprendizaje automatizado, en donde los dos primeros son los más sobresalientes para este proyecto.

1.1. Aprendizaje Supervisado

El aprendizaje supervisado (*Supervised Learning*) es implementado en un sistema fundamentando su utilidad en el campo de la predicción, basando sus experiencias en características que generan efectos reales, valores de salida (numéricos o etiquetas), mediante un *algoritmo de Machine Learning*, y aprende de ellas para la elaboración de un modelo; consiguientemente con un *algoritmo de predicción* basado en nuestro modelo generamos los valores de salida predichos para nuevos datos (datos de prueba); en otras palabras, se enseña a un programa cómo realizar una tarea y que haga uso de aquel conocimiento adquirido para problemas futuros. Las aplicaciones son multivariadas, pero clasificadas por cómo se enfoquen dichos efectos.

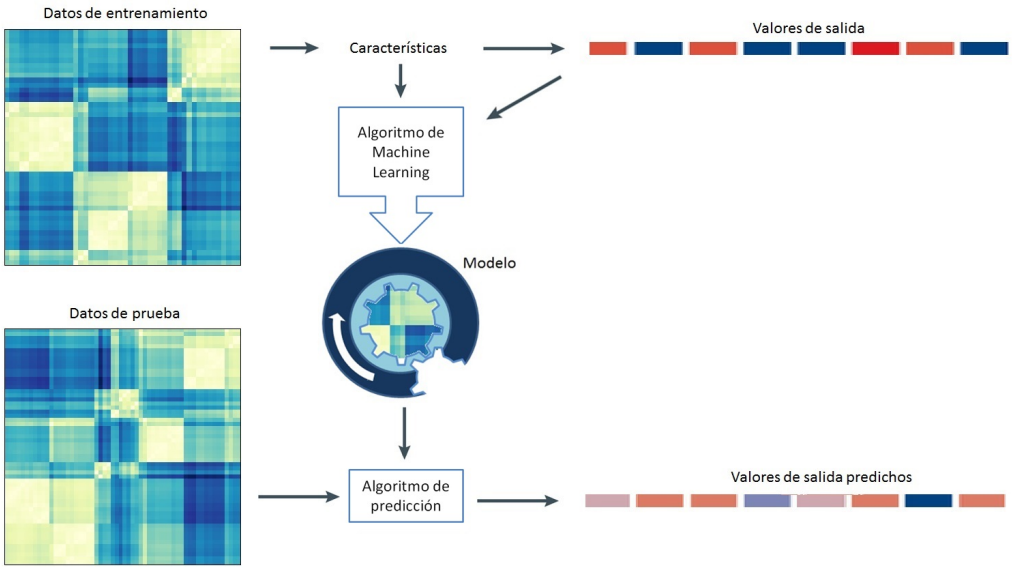


Figura 1.1: Aprendizaje Supervisado (Modelo Gráfico).

1.1.1. Regresión

Esta técnica es efectuada en un sistema para predecir uno o múltiples valores numéricos de salida de datos en base a sus características, y a la experiencia adquirida sobre un conjunto de datos con sus valores de salida reales. Un ejemplo de aprendizaje automatizado por regresión, es el de predecir el progreso de la enfermedad de diabetes en base a variables psicológicas (edad, sexo, peso, presión sanguínea).

1.1.1.1. Algoritmo principal de Regresión

- **Regresión Lineal:** La técnica de regresión lineal, es muy útil y sencilla de elaborar pues establece un modelo lineal de la forma $y = \mathbf{x}^T\theta$ que nos permite relacionar linealmente los valores de salida numéricos dependientes de las características de los datos. A continuación, se visualiza el diseño de un modelo de regresión lineal para 442 pacientes sobre el ejemplo de diábetes planteado anteriormente; la base de datos se encuentra en la librería scikit-learn.datasets.

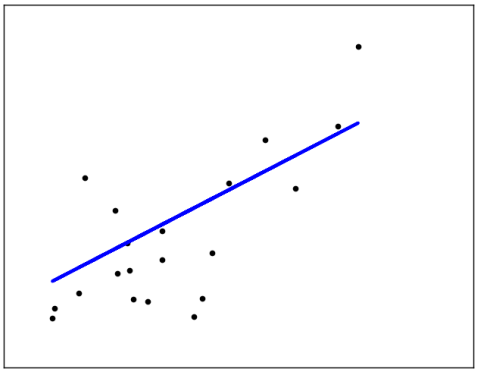


Figura 1.2: Regresión Lineal.

1.1.2. Clasificación

Consiste en enseñar a un sistema que, bajo experiencias en datos que ya se encuentran clasificados de acuerdo a ciertas características, continúe un proceso de clasificación con nuevos datos. Un ejemplo de un sistema que realice esta tarea, es el de contar con la longitud y ancho de sépalos de flores iris que se encuentran clasificadas como Setosa, Versicolor o Virgínica, y en base a ello continuar la clasificación para nuevos datos con tales características.

1.1.2.1. Algoritmos principales de Clasificación

- **Regresión Logística**[1]**:** A pesar del término "regresión", este algoritmo de clasificación no corresponde a la técnica de Regresión, propiamente dicho. El análisis mediante regresión logística es la de establecer una relación entre las características de ciertos datos y valores discretos, permitiéndo modelar un sistema que prediga consiguientemente este tipo de relaciones para nuevos datos. A continuación se visualiza la clasificación de datos con los valores discretos $-\mathbf{1}$ y $\mathbf{1}$.

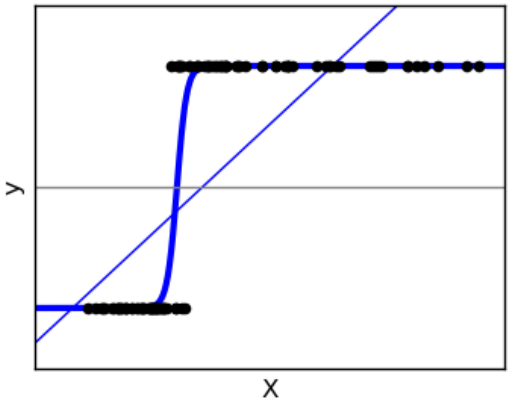


Figura 1.3: Regresión Logística.

- **Árboles de Decisión:** Esta técnica clasifica variables tanto continuas independientes como cualitativas[2] agrupándolas en un diagrama árbol[3], estableciendo un camino predictivo para un dato de entrada basándose en sus características.

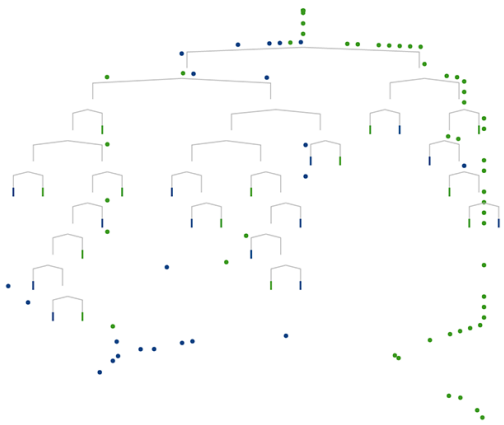


Figura 1.4: Arbol de decisión, modelado por los datos visualizados como puntos.

[1] Algunos autores establecen este algoritmo como parte de la técnica de Regresión.
[2] Variables que pueden ser medidas o presentar un orden.
[3] Diagrama estructurado por categorías.

- **Support Vector Machine (SVM)**[4]**:** Esta técnica distribuye datos $k-$ *dimensionales* en un espacio $\mathbb{R}^k$, donde cada eje de coordenadas se denomina vector soporte, para luego establecerse una geometría de separación aproximada sobre los datos. A continuación, se visualiza zonas de clasificación establecidas sobre dos características.

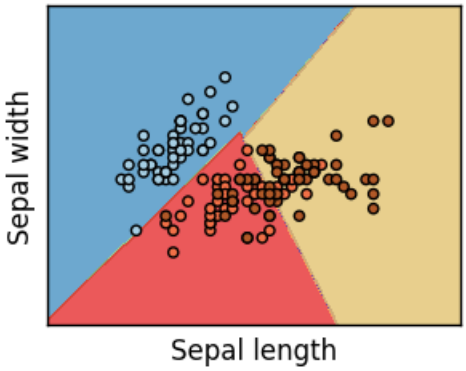


Figura 1.5: SVM.

- **Naive Bayes:** Esta técnica hace uso del Teorema de Bayes, lo cual nos establece lo siguiente:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

donde P es una función de probabilidad; la idea de este resultado es predecir "y" en base al conjunto de datos "X", lográndose esto cuando $P(y|X)$ sea el máximo valor en comparación con cualquier otro valor $P(y'|X)$.

- **K-vecinos más cercanos (KNN)**[5]**:** Esta técnica clasifica a un dato en base a las clasificación de los K datos más cercanos[6] que lo rodean.

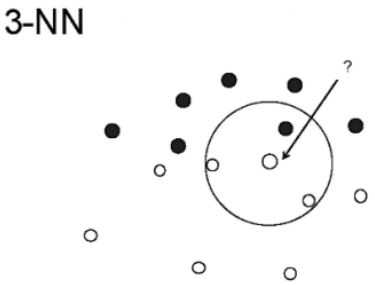


Figura 1.6: KNN.

[4]A pesar de ser una técnica de clasificación, puede implementarse en técnicas de regresión lineal.
[5]Está técnica también tiene implementaciones en técnicas de Regresión
[6]Esto refiere a la implementación de una métrica que actue sobre los valores característicos de los datos

1.2. Aprendizaje no Supervisado

El aprendizaje no supervisado *(Unsupervising Learning)* es implementado en un sistema cuando se necesita estructurar datos que no poseen valores de salida o etiquetas; en otras palabras, representa una serie de técnicas que hace que un sistema aprenda a resolver tareas en base a experiencias que son adquiridas por sí mismo.

Este tipo de aprendizaje es fundamental en la estructuración de los datos cuando no se tiene una información inicial sobre estos, a diferencia del aprendizaje supervisado, que basa sus experiencias a causa de efectos reales. Por tanto la elaboración de un *algoritmo de Machine Learning*, en este caso, estudia las características de datos (datos de entrenamiento), y las estructura en base a ciertas similitudes que se puedan encontrar en la distribución, por lo que consiguientemente con un *algoritmo de estructuración* basado en nuestro modelo, se estructuran nuevos datos (datos de prueba).

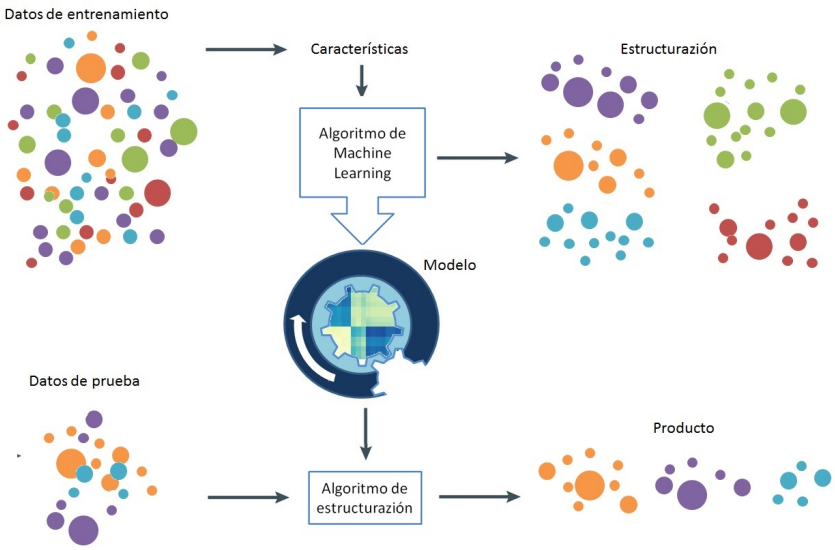


Figura 1.7: Aprendizaje no Supervisado (Modelo Gráfico).

1.2.1. Agrupamiento (Clustering)

A esta técnica del aprendizaje no supervisado se le denomina *Clustering* y agrupa datos que posean similitud en base a la distribución de sus características logrando diferenciar los datos unos de otros. De forma similar a los algoritmos de *Clasificación*, Clustering logra en ciertos aspectos clasificar datos; sin embargo, el proceso es sin tener un conocimiento a priori de algún valor de salida ni etiquetas sobre los datos.

1.2.1.1. Algoritmos principales de Clustering

- **K-means:** Esta técnica asume K agrupamientos (clusters), regidos bajo una característica central, sobre los datos en donde cada una de ellas mantengan cierta homogeneidad con respecto a evaluar datos pertenecientes a distintos grupos, manteniéndose heterogeneidad entre grupos. La idea es obtener el K óptimo que satisfaga lo explicado anteriormente.

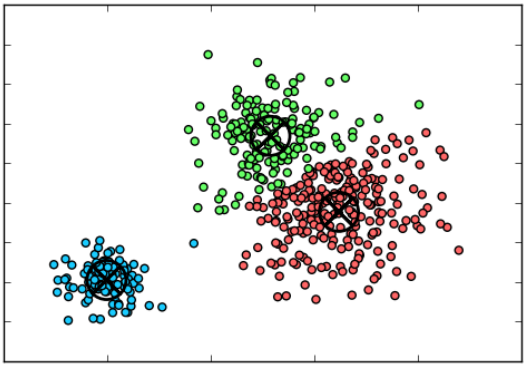


Figura 1.8: K-means.

1.2.2. Estimación de densidad

Este método basa su construcción en un estimador que modele datos observados, formando distribuciones aproximadas.

1.2.2.1. Algoritmos principales de Estimación de densidad

- **Estimación de Densidad de Kernel (KDE):** Esta técnica estima la función de probabilidad de una variable, en este caso de dimensión (característica) de los datos mediante estadística inferencial en base a una cantidad de datos finito.

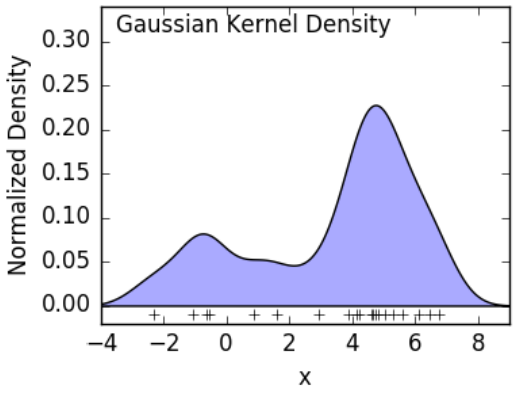


Figura 1.9: Estimación de densidad.

1.2.3. Reducción de Dimensionalidad

El uso correspondiente de esta técnica tiene una alta importancia al trabajar en dimensiones muy elevadas pues dado datos con un número alto de carácterísticas, la reducción de dimensionalidad procesa el aprendizaje basando su estudio en una dimensión reducida. A continuación, se visualizará un conjunto de datos, en donde, en una dirección, puede ser representando como una nube de puntos tridimiensional mientras, que en otra, se aproxima a un plano bidimensional; he aquí una muy buena aplicación de la reducción de dimensionalidad.

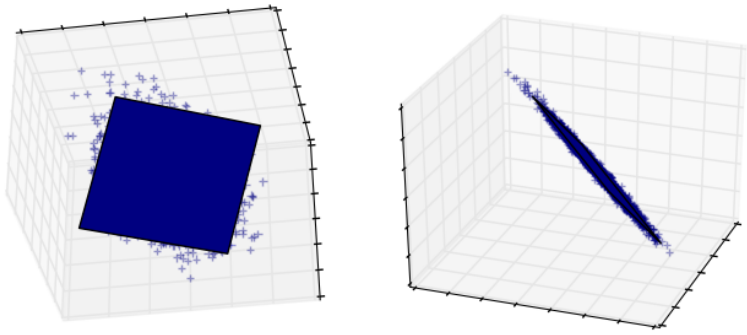


Figura 1.10: Reducción de dimensionalidad.

1.2.3.1. Algoritmos principales de Reducción de Dimensionalidad

- **Descomposición de Valores Singulares (SVD)[7]:** Está técnica en particular es un resultado propio del álgebra lineal; sin embargo, es importante representarlo como una técnica de reducción de dimensionalidad pues por su construcción reduce cálculos de dimensiones altas a dimensiones de menor valor generando un menor costo computacional. Esta técnica descompone una matriz $A \in \mathbb{R}^{m \times n}$ como $A = U\Sigma V^T$ donde U, V son autovectores de las matrices AA^T y A^TA respectivamente, y Σ matriz diagonal que mantiene los valores singulares de A.

- **Análisis de Componentes Principales (PCA):** El procedimiento llevado por esta técnica es la de establecer un subespacio de menor dimensión que los datos, tal que la pérdidad de información sobre los datos establecida en dicho subespacio, sea insignificante.

[7]Existen múltiples técnicas al igual que SVD que reducen de forma similar el costo computacional como lo es la descomposición QR, Cholesky, LU, etc.

1.3. Aprendizaje por Refuerzo

El aprendizaje por refuerzo *(Reinforcement Learning)* es uno de los más importantes en su uso sobre el campo de la inteligencia artificial. Esta técnica de aprendizaje es mucho más compleja pues implementa una relación **recompensa - castigo** en su proceso, es decir, que mientras el sistema desarrolle una tarea y se vea bien realizada se le recompensará asignándole un valor positivo, en caso contrario se le castigará, asignándole un valor negativo; por lo que nuestro sistema mejora rendimientos, analiza situaciones y realiza nuevas tareas quizás desarrolladas por sí mismo.

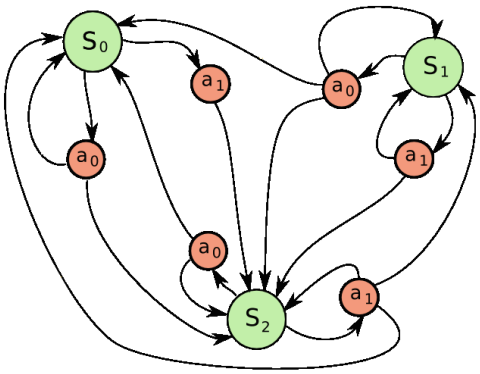


Figura 1.11: Aprendizaje por Refuerzo (Modelo Gráfico).

Capítulo 2

Nociones Básicas

Con el fin de contar con las herramientas básicas para el análisis correspondiente de la reducción de dimensionalidad ya introducido en el capítulo anterior, se definirán algunos conceptos matemáticos y estadísticos que influirán adecuadamente.

2.1. Álgebra lineal

Evitando la complejidad de los resultados, el espacio en el que se enfocará es en R^n, por tanto los conceptos dados se limitan a este espacio.

2.1.1. Matrices y Vectores

Definición 2.1. (Matriz - vector - dimensión). *Denominaremos* **matriz** *a un conjunto de números reales agrupados en n filas y m columnas, denotado como* $A \in \mathbb{R}^{n\times m}$*, siendo* $n \times m$ *su* **dimensión***, y presenta la siguiente estructura*

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & \ddots & \ddots & a_{2m} \\ \vdots & \ddots & \ddots & \vdots \\ a_{n1} & \dots & \dots & a_{nm} \end{bmatrix} = [a_{ij}]_{n\times m}$$

Se denominará **vector** *cuando* $m = 1$ *y se denota como* $v \in \mathbb{R}^n$*. Su representación es*

$$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

Operaciones con matrices:

- Suma: $A + B = [a_{ij} + b_{ij}]_{n\times m}$ donde $A = [a_{ij}]_{n\times m}$ y $B = [b_{ij}]_{n\times m}$.
- Producto: $AB = \left[\sum_{k=1}^{p} a_{ik}b_{kj}\right]_{n\times m}$ donde $A = [a_{ij}]_{n\times p}$ y $B = [b_{ij}]_{p\times m}$.
  Este producto solo está definido si el número de columnas de A es el número de filas de B.
- Producto: $cA = [ca_{ij}]_{n\times m}$ donde $A = [a_{ij}]_{n\times p}$ y $c \in \mathbb{R}$.

Propiedades del producto de matrices:

- Asociatividad: $(AB)C = A(BC)$
- Distributividad: $A(B + C) = AB + AC$
- No Conmutatividad: Si $A = [a_{ij}]_{n\times p}$ y $B = [b_{ij}]_{p\times m}$, luego $AB = [c_{ij}]_{n\times m}$; sin embargo, BA no está definido si $n \neq m$. En caso $m = n$, entonces $AB = [c_{ij}]_{n\times n}$ y $BA = [c_{ij}]_{p\times p}$ pero no necesariamente $n = p$, es decir $AB \neq BA$ en general.

Tipos matrices:

- **Matriz cuadrada**: Sea $A \in \mathbb{R}^{m\times n}$, diremos que A es una matriz cuadrada si $m = n$.
- **Matriz diagonal** Sea $A \in \mathbb{R}^{n\times n}$, diremos que $A = [a_{ij}]_{n\times n}$ es una matriz diagonal si $a_{ij} = 0$ para todo $i \neq j$.
- **Matriz identidad**: La matriz identidad $\mathbb{I} \in \mathbb{R}^{n\times n}$, donde $\mathbb{I}_{n\times n} = [a_{ij}]_{n\times n}$, se define como una matriz diagonal tal que $a_{ij} = 1$ para todo $i = j$.
- **Matriz transpuesta**: Sea $A \in \mathbb{R}^{n\times n}$, donde $A = [a_{ij}]_{m\times n}$, definimos su transpuesta como $A^T = [a_{ji}]_{n\times m}$.
- **Matriz simétrica**: Sea $A \in \mathbb{R}^{m\times n}$, diremos que A es una matriz simétrica si $A = A^T$.
- **Matriz inversa**: Sea $A \in \mathbb{R}^{n\times n}$, diremos que A tiene inversa si existe una matriz $B \in \mathbb{R}^{n\times n}$ tal que $AB = I$. L matriz inversa se denota como A^{-1}.
  (Se cumple la conmutatividad $AA^{-1} = A^{-1}A = I$).
- **Matriz ortogonal.**: Sea $A \in \mathbb{R}^{n\times n}(n \geq 2)$, es matriz ortogonal si $A^{-1} = A^T$.

Definición 2.2. (Traza). *Sea $A \in \mathbb{R}^{n\times n}$, se define la traza de A como*

$$Tr(A) = \sum_{i=1}^{n} a_{ii}, \quad \textit{donde } A = [a_{ij}]_{n\times n}.$$

Propiedades de la Traza:

- Sea $A, B \in \mathbb{R}^{n\times m}$, entonces $Tr(A+B) = Tr(A) + Tr(B)$
- Sea $A \in \mathbb{R}^{n\times m}$ y $c \in \mathbb{R}$, entonces $cTr(A) = Tr(cA)$
- Sea $A \in \mathbb{R}^{n\times m}$ y $B \in \mathbb{R}^{m\times n}$, entonces $Tr(AB) = Tr(BA)$.

Proposición 2.1. *Si $A \in \mathbb{R}^{n\times m}$, luego AA^T y A^TA son matrices cuadradas simétricas.*

Demostración. Dado que $A \in \mathbb{R}^{n\times m}$, tenemos que $A^T \in \mathbb{R}^{m\times n}$ entonces $AA^T \in \mathbb{R}^{n\times n}$ y $AA^T \in \mathbb{R}^{m\times m}$, matrices cuadradas.
Además $(AA^T)^T = (A^T)^TA^T = AA^T$ por tanto AA^T es simétrica; análogamente A^TA es simétrica. □

Definición 2.3. (Producto Punto). *Se define el producto punto de $u, v \in \mathbb{R}^n$ como*

$$u \cdot v = u^T v = \sum_{i=1}^{n} u_i v_i, \quad \text{donde } u = [u_i]_{n\times 1}, v = [v_i]_{n\times 1}.$$

Propiedades del Producto Punto:

- Definida positiva: $(v \cdot v) \geq 0$. Además $v \cdot v = 0$ si y solo si $v = 0$.
- Simetría: $u \cdot v = v \cdot u$.
- Linealidad: $(au + bv) \cdot w = a(u \cdot w) + b(v \cdot w)$.

Definición 2.4. (Norma). *Se denota la norma de un vector $v \in R^n$ por $\|v\|$ y se define como*

$$\|v\| = \sqrt{v \cdot v} = \sqrt{\sum_{i=1}^{n} v_i^2}, \quad \text{donde } v = [v_i]_{n\times 1},$$

representando su longitud.

Definición 2.5. (Distancia entre vectores) *Se define la distancia entre $u, v \in \mathbb{R}^n$ como*

$$d(u, v) = \|u - v\|.$$

Definición 2.6. (Ortogonalidad). *Sean $u, v \in \mathbb{R}^n$, se dice que u y v son ortogonales si $u \cdot v = 0$. Además, un conjunto $\{u_1, u_2, \ldots, u_m\} \subset \mathbb{R}^n$ $(m \leq n)$ es un conjunto ortogonal si $u_i \cdot u_j = 0$ para todo $i \neq j$.*

Definición 2.7. (Vector Normal). *Un vector $v \in \mathbb{R}^n$ es normal, o unitario, si $\|v\| = 1$.*

Definición 2.8. (Ortonormalidad). *Un conjunto* $\{u_1, u_2, \ldots, u_m\} \subset \mathbb{R}^n$ $(m \leq n)$ *es ortonormal si es un conjunto ortogonal y además* $\|u_i\| = 1$ *para todo* $i = 1, \ldots, n$.

Corolario 2.1. *Sea* $U \in \mathbb{R}^{n\times n}$*, sus vectores columna* $u_1, \ldots, u_n$ *forman una ortonormal de* R^n *si y solo si* U *es ortogonal, es decir* $U^T = U^{-1}$.

Demostración. Veamos la representación matricial de U^TU

$$U^TU = \begin{bmatrix} u_1^T \\ \vdots \\ u_n^T \end{bmatrix} \begin{bmatrix} u_1 & u_2 & \ldots & u_n \end{bmatrix} = \begin{bmatrix} u_1^Tu_1 & \ldots & u_1^Tu_n \\ \vdots & \ddots & \vdots \\ u_n^Tu_1 & \ldots & u_n^Tu_n \end{bmatrix} \tag{2.1}$$

Por tanto,

$$\{u_1, \ldots, u_n\} \text{ base ortonormal de } R^n \Leftrightarrow u_i^Tu_j = \begin{cases} 1 & , i = j \\ 0 & , i \neq j \end{cases} \Leftrightarrow U^TU = \mathbb{I}_{n\times n} \tag{2.2}$$

□

Definición 2.9. (Linealidad independiente) *Un conjunto* $\{u_1, u_2, \ldots, u_m\} \subset \mathbb{R}^n$ $(m \leq n)$ *es linealmente independiente si la siguiente igualdad*

$$\sum_{i=1}^{m} c_iu_i = 0 \quad \text{con } c_i \in \mathbb{R} \text{ para todo } i = 1, \ldots, m$$

solo se satisface cuando $c_i = 0$ *para todo* $i = 1, \ldots, m$.

Definición 2.10. *(***Rango.)** *Sea* $A \in \mathbb{R}^{n\times m}$*, se define el rango de* A *como el número de columnas de* A *linealmente independientes, y denotado por* $Rang(A)$.

Proceso de Ortogonalización de Gram-Schmidt

Sea un conjunto linealmente independiente de vectores $\{v_1, \ldots, v_n\}$ que generen un subespacio S, entonces se pueden construir un conjunto ortogonal $\{u_1, \ldots, u_n\}$ que generen el mismo subespacio, de la siguiente forma

$$u_i \quad = \quad v_i - \sum_{k=1}^{i-1} \frac{u_k \cdot v_i}{\|u_k\|^2} u_k, \quad \text{para todo } i = 1, \ldots, n. \tag{2.3}$$

La determinación de un conjunto ortonormal sería $\left\{ \frac{u_1}{\|u_1\|}, \ldots, \frac{u_n}{\|u_n\|} \right\}$.

2.1.2. Eigenvectores y Eigenvalores

Definición 2.11. (Eigenvector-Eigenvalor)*: Sea* $A \in \mathbb{R}^{n\times n}$ *y* $v \in \mathbb{R}^n$ *distinto de cero, se dice que* v *es eigenvector de* A *si*

$$Av = \lambda v, \quad \text{para algún } \lambda \in \mathbb{R}.$$

Se dice que λ *es el eigenvalor de* A*, correspondiente a* v.

Proposición 2.2. *Si v es un eigenvector, con valor propio $\lambda \neq 0$, de A^TA, luego Av es un eigenvector con el mismo valor propio λ de AA^T.*

Demostración. Sea v es un eigenvector de A^TA y $\lambda \neq 0$ su eigenvalor correspondiente, entonces $A^TAv = \lambda v$.
Luego multiplicando por A a la izquierda

$$\begin{aligned} A(A^TAv) &= A(\lambda v) \\ \Rightarrow \quad AA^T(Av) &= \lambda(Av). \end{aligned}$$

Por lo que Av es eigenvector de AA^T y $\lambda \neq 0$ su eigenvector correspondiente. □

Proposición 2.3. *Sea $A \in \mathbb{R}^{n\times n}$, entonces A^TA no tiene autovalores negativos.*

Demostración. Sea u un eigenvector de A^TA y λ su eigenvalor correspondiente, entonces $A^TAu = \lambda u$, luego

$$\begin{aligned} u^T(A^TAu) &= u^T(\lambda u) \\ (u^TA^T)Au &= \lambda u^Tu \\ (Au)^TAu &= \lambda\|u\| \\ \|Au\|^2 &= \lambda\|u\|^2 \Rightarrow \lambda = \frac{\|Au\|^2}{\|u\|^2} \geq 0. \end{aligned}$$

□

Definición 2.12. (Valores Singulares). *Sea $A \in \mathbb{R}^{n\times m}$, se definen los valores singulares de A como las raíces cuadradas de los eigenvalores de A^TA.*
La buena definición, es a causa de la Proposición 2.1.

Definición 2.13. (Diagonalización) *Sea $A \in \mathbb{R}^{n\times n}$, se dice que A es diagonalizable si y solo si A puede ser descompuesta como*

$$A = UDU^{-1} \quad \text{donde } U = \begin{bmatrix} u_1 & \dots & u_n \end{bmatrix} \text{ y } D = diag(\lambda_1, \dots, \lambda_n).$$

si U es matriz ortogonal, entonces se dice que A es ortogonalmente diagonalizable, por tanto $A = UDU^T$.

Proposición 2.4. *Sea la matriz $A \in \mathbb{R}^{n\times n}$ con autovalores $\lambda_1, \lambda_2, \dots, \lambda_n \in \mathbb{R}$ entonces*

$$\sum_{i=1}^{n} \lambda_i = Tr(A).$$

Demostración. Como $A = UDU^{-1}$, usando las propiedades de la traza tenemos que

$$Tr(A) = Tr(UDU^{-1}) = Tr(U^{-1}UD) = Tr(D) = \sum_{i=1}^{n} \lambda_i$$

□

2.1.3. Teoría Espectral

Teorema 2.1. (Teorema Espectral). *Si $A \in \mathbb{R}^{n\times n}$ es simétrica, entonces existe una base ortonormal de R^n formada por eigenvectores de A.*

Demostración. Primero veamos que si una matriz A es ortogonalmente diagonalizable entonces existe una base ortonormal de $\mathbb{R}^n$ formada por eigenvectores de A; y luego veamos que siendo A simétrica entonces es ortogonalmente diagonalizable para concluir la prueba.

- Si A es ortogonalmente diagonalizable entonces $A = UDU^T$, donde U es matriz ortogonal y $D = diag(\lambda_1, \ldots, \lambda_n)$. Además $U^T AU = D$.
  Ahora veamos que los vectores columna de $U = [u_1 \ldots u_n]$, siendo estos

$$u_i = \begin{bmatrix} u_1 & \ldots & u_i & \ldots & u_n \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 1 \leftarrow i \\ \vdots \\ 0 \leftarrow n \end{bmatrix} = Ue_i,$$

  son eigenvectores de A.

$$Pu_i = PUe_i = P(P^T PUe_i) = P(De_i) = P(\lambda_i e_i) = \lambda_i Pe_i = \lambda_i u_i.$$

  Por lo que u_i es eigenvector de A para todo $i = 1, \ldots, n$ y por corolario 2.2, $\{u_1, \ldots, u_n\}$ es una base de R^n formada por eigenvectores de A.

- Sea A simétrica, y u_1 un eigenvector normal con λ_i su correspondiente eigenvalor. Usando el método de Gram-Schmidt, determinamos un conjunto ortonormal $u_1, \ldots, u_n$ de R^n y tomamos $U = [u_1 \ \ldots \ u_n]$.
  El análisis será por inducción, es decir suponiendo que una matriz simétrica $M \in \mathbb{R}^{n-1\times n-1}$ es ortogonalmente diagonalizable, entonces veamos que una matriz simétrica $A \in \mathbb{R}^{n\times n}$ también lo es.
  Si $U = [u_1 \ \ldots \ u_n]$, siendo esta matriz ortogonal por Corolario 2.1, y por el hecho de que $Ue_1 = u_1$ y $U^T u_1 = e_1$ tenemos la primera columna de $U^T AU$:

$$U^T AUe_1 = U^T Au_1 = U^T \lambda u_1 = \lambda_1 e_1 = \begin{bmatrix} \lambda_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Además $(U^TAU)^T = U^TAU$, es decir U^TAU es matriz simétrica, por lo tanto

$$U^TAU = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \mathrm{M} \end{bmatrix}$$

siendo $M \in \mathbb{R}^{(n-1\times(n-1))}$ matriz simétrica; por lo supuesto M es ortogonalmente diagonalizable, entonces $VMV^T = D_M$ donde V es matriz ortogonal y D_M es matriz diagonal.
Veamos que

$$W = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V} \end{bmatrix} U^T$$

es matriz ortogonal y que WAW^T es diagonal, usando el hecho de que $U^TU = \mathbb{I}_{n\times n}$

$$WW^T = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V} \end{bmatrix} U^TU \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V}^T \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V}^T \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mathbb{I}_{(n-1)\times(n-1)} \end{bmatrix}$$

por lo que W es matriz ortogonal. Ahora

$$\begin{aligned} W^TAW &= \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V} \end{bmatrix} U^TAU \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V}^T \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V} \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \mathrm{M} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{V}^T \end{bmatrix} \\ &= \begin{bmatrix} \lambda_1 & 0 \\ 0 & \mathrm{D}_M \end{bmatrix} \end{aligned}$$

Por tanto A es ortogonalmente diagonalizable.

□

Como se ha visto en la demostración de la proposición anterior, tenemos el siguiente corolario,

Corolario 2.2. *Sea $A \in \mathbb{R}^{n\times n}$ simétrica, entonces es ortogonalmente diagonalizable por sus eigenvectores.*

2.1.4. Proyección

Definición 2.14. *Sea $v \in \mathbb{R}^n$ y P un subespacio de $\mathbb{R}^n$, se define la proyección ortogonal de v sobre P como $w = proy_P(v) \in S$ tal que*

$$\|w - v\| \leq \|z - v\| \quad \textit{para todo } z \in S.$$

Teorema 2.2. (Matriz de proyección). *Dado* $\{u_1, u_2, \ldots, u_m\}$ *base de un subespacio* $S \subset \mathbb{R}^n$ *y sea* $v \in R^n$*, entonces la proyección ortogonal de* v *sobre* S *es*

$$proy_S(v) = U(U^TU)^{-1}U^Tv \quad \text{donde } U = \begin{bmatrix} u_1 & \ldots & u_m. \end{bmatrix}$$

$P = U(U^TU)^{-1}U^T$ *se conoce como la matriz de proyección.*

Teorema 2.3. (Descomposición Ortogonal). *Dado* $\{u_1, u_2, \ldots, u_m\}$ *base ortogonal de un subespacio* $S \subset \mathbb{R}^n$ *y sea* $v \in R^n$*, entonces la proyección ortogonal de* v *sobre* S *es*

$$proy_S(v) = \frac{v \cdot u_1}{\|u_1\|^2}u_1 + \frac{v \cdot u_2}{\|u_2\|^2}u_2 + \ldots + \frac{v \cdot u_m}{\|u_m\|^2}u_m.$$

Si $\{u_1, u_2, \ldots, u_m\}$ *es una base ortonormal entonces:*

$$proy_S(v) = (v \cdot u_1)u_1 + (v \cdot u_2)u_2 + \ldots + (v \cdot u_m)u_m.$$

Demostración. Por Teorema 2.2, $proy_S(v) = U(U^TU)^{-1}U^Tv$, donde $U = \begin{bmatrix} u_1 & \ldots & u_m \end{bmatrix}$. Dado que el conjunto $\{u_1, u_2, \ldots, u_m\}$ es ortogonal entonces $u_i^Tu_j = 0$ para todo $i \neq j$. Ahora veamos lo siguiente

$$(U^TU)^{-1} = \left(\begin{bmatrix} u_1^T \\ \vdots \\ u_m^T \end{bmatrix} \begin{bmatrix} u_1 & \ldots & u_m \end{bmatrix} \right)^{-1} = \left(\begin{bmatrix} \|u_1\|^2 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \|u_m\|^2 \end{bmatrix} \right)^{-1} = \begin{bmatrix} \frac{1}{\|u_1\|^2} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \frac{1}{\|u_m\|^2} \end{bmatrix}$$

Luego,

$$proy_S(v) = \begin{bmatrix} u_1 & \ldots & u_m \end{bmatrix} \begin{bmatrix} \frac{1}{\|u_1\|^2} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \frac{1}{\|u_m\|^2} \end{bmatrix} \begin{bmatrix} u_1^T \\ \vdots \\ u_m^T \end{bmatrix} v = \begin{bmatrix} u_1 & \ldots & u_m \end{bmatrix} \begin{bmatrix} \frac{u_1^Tv}{\|u_1\|^2} \\ \vdots \\ \frac{u_m^Tv}{\|u_m\|^2} \end{bmatrix}$$

Por tanto $proy_S(v) = \sum_{i=1}^{m} \frac{v \cdot u_i}{\|u_i\|^2}u_i$. Si $\{u_1, \ldots, u_m\}$ es un conjunto ortogonal entonces $\|u_i\| = 1$, por lo que $proy_S(v) = \sum_{i=1}^{m}(v \cdot u_i)u_i$. □

Veamos que siendo $\{u_1, u_2, \ldots, u_m\}$ un conjunto ortonormal de $\mathbb{R}^n$ entonces estos vectores determinarían un eje de coordenadas de dimensión m, en donde las nuevas coordenadas para $proy_S(v)$ sean $\left\{ \frac{v \cdot u_i}{\|u_i\|} \right\}_{i=1}^{m}$; para entender este resultado partiremos por lo siguiente

Teorema 2.4. (Ley de Variación de los Componentes). *Dado* $\{u_1, u_2, \ldots, u_n\}$ *una base ortonormal de* R^n *y sea* $p \in \mathbb{R}^n$*. Si los vectores unitarios* e_i*, del sistema de coordenadas, se desplazan hacia los vectores* u_i *respectivamente, formando un nuevo sistema, entonces* p *en este nuevo sistema de coordenadas viene dado por*

$$p_2 = Up \quad \textit{donde } U = \begin{bmatrix} u_1 & \ldots & u_n \end{bmatrix} \textit{ se denomina matriz de rotación.}$$

Ahora si S es el subespacio generado por $\{u_1, \ldots, u_m\}$, extendiendola a una base ortonormal de R^n, sea $\{u_1, \ldots, u_n\}$ entonces la proyección de $v \in \mathbb{R}^n$ sobre S, por el Teorema 2.3, se tiene que

$$\begin{aligned} proy_{\,S}(v) &= (v \cdot u_1)u_1 + \ldots + (v \cdot u_n)u_m + 0u_{m+1} \ldots + 0u_n. \\ &= \begin{bmatrix} u_1 & \ldots & u_m & u_{m+1} & \ldots & u_n \end{bmatrix} \begin{bmatrix} u_1^T v \\ \vdots \\ u_m^T v \\ 0 \\ \vdots \\ 0 \end{bmatrix} \end{aligned}$$

Por la "ley de variación de los componentes", el vector $(u_1^T v, \ldots, u_m^T v, 0, \ldots, 0) \in \mathbb{R}^n$ representa el vector $proy_{\,S}(v)$ en otro sistema de coordenadas, pero como se observa, tal vector puede incorporarse en un sistema de menor dimensión

$$proy_{\,S}(v) \in \mathbb{R}^n \equiv (u_1^T v, \ldots, u_m^T v, 0, \ldots, 0) \in \mathbb{R}^n \equiv (u_1^T v, \ldots, u_m^T v) \in \mathbb{R}^m \tag{2.4}$$

Por lo que hemos determinado que $proy_{\,S}(v)$ puede ser representado en un espacio $\mathbb{R}^m$ como $U^T v$ donde $U = \begin{bmatrix} u_1 & \ldots & u_m \end{bmatrix}$, teniendose por tanto la siguiente proposición.

Proposición 2.5. *Sea* S *subespacio de* R^n *con una base ortonormal* $u_1, \ldots, u_m$ *que la genere y sea* $v \in \mathbb{R}^n$*, entonces* $proy_{\,S}(v)$ *puede ser incorporado en* R^n*, mediante una rotación de su sistema de coordenadas, como el vector*

$$\begin{bmatrix} u_1^T \\ \vdots \\ u_m^T \end{bmatrix} v$$

2.2. Estadística Descriptiva

La teoría de estadística descriptiva que nos concierne en esta sección, nos facilitá las descripciones de los problemas que tratemos en el estudio de la técnica PCA y consiguientemente de la aplicación en reconocimiento facial.

2.2.1. Variables aleatorias

Los resultados de un experimento pueden tener naturaleza tanto numérica como no numérica, en este último se les asociaría un valor númerico a cada distinto resultado. El concepto de variable aleatoria es el de establecerse un conjunto numérico de salidas para todo tipo de experimentos.
Por tanto, una variable aleatoria X es una función $X : \Omega \to \mathbb{R}$. Una idea sencilla de variable aleatoria es $X = \{1, 2, \ldots, 6\}$ representando los resultados del experimento al lanzar un dado, siendo este de naturaleza numérica. Si la variable aleatoria es $X = \{0, 1\}$ representado los resultados del experimento al lanzar una moneda(0 si sale cara 1 si sale sello), este sería de naturaleza no numérica.
Además se define una **variable aleatoria discreta** si esta puede tomar una cantidad finita de valores; y una **variable aleatoria continua** si puede tomar una cantidad infinita de valores.

Una variable aleatoria para nuestro estudio representan las dimensiones de los datos, es decir, de poseer n características, corresponden a n variables aleatorias.

2.2.2. Tendencias centrales y dispersión

Dado que es necesario la determinación de un valor típico como resultado de una serie de resultados por cada variable aleatoria, se define la *media*, la cual determina un valor promedio de una cantidad específica de datos. Además definimos cantidades relacionadas a las distribuciones con el objetivo de medir sus dispersiones.

Definición 2.15. (Media). *Sea X una variable aleatoria correspondiente a un conjunto de M datos, definimos la media como*

$$\overline{x} \quad = \quad \frac{1}{M} \sum_{i=1}^{M} x_i$$

donde x_i son los valores que toma la variable X en cada uno de los datos.

La media nos es útil para determinar el valor que uno intuye que pudiese ocurrir; sin embargo este valor no indica certeza por lo que se necesita tener en cuenta que existe la posibilidad de que un valor resultante se desvíe de la media; una de las variables que nos permite cuantificar esta medida de riesgo es la varianza.

Definición 2.16. (Varianza). *Sea X una variable aleatoria correspondiente a un conjunto de M datos, definimos la varianza como*

$$Var(X) = \frac{1}{M}\sum_{i=1}^{M}(x_i - \overline{x})^2.$$

donde $\overline{x}$ es la media de los datos, x_i valores que toma la variable X en cada uno de los datos.

La varianza nos establece el tipo de variabilidad de una variable aleatoria pues mide el riesgo de que un resultado no sea el valor promedio. En cierto aspectos se hace uso de la **desviación estándar** definida como la raíz cuadrada de la varianza.

Definición 2.17. (Covarianza). *Sean M datos con dos variables X, Y, entonces la covarianza se define como*

$$Cov(X, Y) = \frac{1}{M}\sum_{i=1}^{M}(x_i - \overline{x})(y_i - \overline{y}).$$

donde $\overline{x}, \overline{y}$ son la medias de los datos respectivamente a las variables X, Y.

Cuyo análisis es el siguiente:

1. Si la covarianza es altamente positiva entonces existen grandes valores en una variable y grandes valores en la otra,

2. Si la covarianza es positiva pero baja entonces existen valores bajos en una variable y de igual manera en la otra,

3. Si la covarianza es altamente negativa entonces existen grandes valores en una variable y bajos valores en la otra.

Como notamos, esta medida de variabilidad solo está definida para relacionar dos variables aleatorias, para ello definiremos una matriz que relaciona simultáneamente las varianzas entre diversas variables aleatorias y, que por su simetría, trae consigo importantes resultados.

Definición 2.18. (Matriz de Covarianza). *Sean M datos con n variables $X_1, \ldots, X_n$, entonces la matriz de covarianza se define como*

$$\Sigma = \begin{bmatrix} Cov(X_1, X_1) & \cdots & Cov(X_1, X_n) \\ \vdots & \ddots & \vdots \\ Cov(X_n, X_1) & \cdots & Cov(X_n, X_n) \end{bmatrix}$$

	ϕ_1	ϕ_1	$\dots$	ϕ_M
X_1	x_{11}	x_{12}	$\dots$	x_{1M}
X_2	$\vdots$	$\ddots$		$\vdots$
$\vdots$				
X_n	x_{n1}	x_{n2}	$\dots$	x_{nM}

Cuadro 2.1: M datos con n variables

Proposición 2.6. *Sea $\{\Phi_1, \dots, \Phi_M\}$ un conjunto de M datos con n variables $X_1, \dots, X_n$ con media igual a cero en cada una de ellas, entonces la matriz de covarianza viene determinana como*

$$\Sigma = \frac{1}{M} \begin{bmatrix} \Phi_1 & \dots & \Phi_M \end{bmatrix} \begin{bmatrix} \Phi_1^T \\ \vdots \\ \Phi_M^T \end{bmatrix}$$

Demostración. Sean los M datos como se presentan en el Cuadro 2.1. Suponiendo que la media es igual a cero en cada variable entonces tenemos que

$$cov(X_i, X_j) = \frac{1}{M} \sum_{k=1}^{M} x_{ik} y_{jk} \text{ para todo } i, j = 1, \dots, n.$$

Ahora veamos lo siguiente, para concluir la prueba,

$$\begin{aligned}
\frac{1}{M} \begin{bmatrix} \Phi_1 & \dots & \Phi_M \end{bmatrix} \begin{bmatrix} \Phi_1^T \\ \vdots \\ \Phi_M^T \end{bmatrix} &= \frac{1}{M} \begin{bmatrix} x_{11} & \dots & x_{1M} \\ \vdots & \ddots & \vdots \\ x_{n1} & \dots & x_{nM} \end{bmatrix} \begin{bmatrix} x_{11} & \dots & x_{n1} \\ \vdots & \ddots & \vdots \\ x_{1M} & \dots & x_{nM} \end{bmatrix} \\
&= \frac{1}{M} \begin{bmatrix} \sum_{k=1}^{M} (x_{1k})^2 & \dots & \sum_{k=1}^{M} x_{1k} x_{nk} \\ \vdots & \ddots & \vdots \\ \sum_{k=1}^{M} x_{nk} x_{1k} & \dots & \sum_{k=1}^{M} (x_{nk})^2 \end{bmatrix} \\
&= \begin{bmatrix} Cov(X_1, X_1) & \dots & Cov(X_n, X_1) \\ \vdots & \ddots & \vdots \\ Cov(X_n, X_1) & \dots & Cov(X_n, X_n) \end{bmatrix} \\
&= \Sigma \text{ (matriz de covarianza).}
\end{aligned}$$

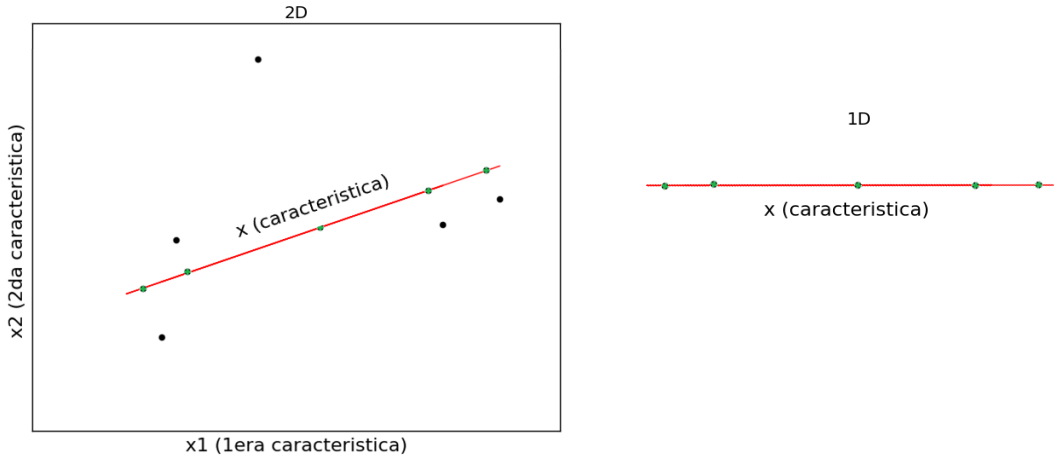
□

Capítulo 3

Reducción de Dimensionalidad

Este método de aprendizaje no supervisado, nos permite que la multidimensionalidad de los datos sea comprimida, es decir, que el número de características, correspondientes a los datos, se vean reducidas en base a proyectarlas hacia un espacio de menor dimensión.

La idea de esta reducción de dimensional es que nos resuma la data sin perder información relevante además de producir la optimización de un costo computacional. Para una primera visión de este método de aprendizaje, supongamos que tenemos datos con dos características, donde una represente la habilidad humana en una cierta tarea T y otra característica que cuantifique el *placer* al realizarla, ello puede ser resumido en una sola característica que represente *aptitud* (ver figura 3.1); esto es reducción de dimensionalidad.



(a) Datos en 2 dimensiones (b) Datos en 1 dimensión

Figura 3.1: Reducción de dimensionalidad

3.1. Descomposición de Valores Singulares (SVD)

La descomposición de valores singulares, *Singular Value Decomposition* (SVD), representa una técnica, que está relacionada directamente con la diagonalización de una matriz simétrica, cuya importancia es fundamental en una serie de aplicaciones del álgebra lineal. En particular esta técnica, será usada en el análisis de componentes principales; sin embargo, es importante tener que cuenta que existen múltiples técnicas en lugar de SVD, como lo es la descomposición QR, Cholesky, LU y demás.
La idea es representar una matriz $A \in \mathbb{R}^{m\times n}$ de la siguiente forma:

$$A = U\Sigma V^T \tag{3.1}$$

donde $U \in \mathbb{R}^{m\times m}$, $V \in \mathbb{R}^{n\times n}$ son matrices ortogonales, y $\Sigma \in \mathbb{R}^{m\times n}$ es una matriz que toma valores $\Sigma_{ii} = \sigma_i$ y $\Sigma_{ij} = 0$ $(i \neq j)$. Además, cada elemento σ_i son los valores singulares de la matriz A, y por ello la denominación de la técnica.

3.1.1. Construcción

Tomamos el mín$\{m, n\}$, sin pérdida de generalidad supongamos $n \leq m$, ya que en caso contrario, se aplicaría la descomposición sobre A^T y tomando la transpuesta se obtendría el SVD de A.
Primero analizemos A^TA, de dimensión $n \times n$, siendo esta simétrica , por Corolario 2.2, entonces

$$A^TA = VDV^T = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_n^T \end{bmatrix} \tag{3.2}$$

donde $\{v_1, v_2, \dots, v_n\}$ es un conjunto ortogonal de eigenvectores de A^TA y $\lambda_1, \lambda_2, \dots, \lambda_n$ sus eigenvalores correspondientes, los cuales no son negativos (Proposición 2.3).
Ahora veamos lo siguiente:

$$Av_i \cdot Av_j = (Av_i)^T Av_j = v_i^T A^T Av_j = v_i^T \lambda_j v_j = \lambda_j v_i \cdot v_j \tag{3.3}$$

Por lo que $\{Av_1, Av_2, \dots, Av_n\}$ es un conjunto ortogonal de R^n, cuyos vectores distintos de cero formarían una base ortogonal de $Im(A)$; sin pérdidad de generalidad sea esta base $\{Av_1, Av_2, \dots, Av_k\}$. Luego, $Av_i = 0$ para todo $i = k+1, \dots, n$ y por (3.3) tenemos que $\lambda_i = \|Av_i\|^2 = 0$. Asumiendo que los eigenvalores λ_i se encuentran en orden decreciente, entonces

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_k > 0 \quad \text{y} \quad \lambda_{k+1} = \dots = \lambda_n = 0 \tag{3.4}$$

Además, $\|Av_i\| = \sqrt{\lambda_i} > 0$ para todo $i = 1, \ldots, k$, por lo que una base ortonormal de $Im(A)$ vendría determinada como $\{u_1, u_2, \ldots, u_k\}$ donde

$$u_i = \frac{Av_i}{\|Av_i\|} = \frac{Av_i}{\sqrt{\lambda_i}} \tag{3.5}$$

Por definición $\sigma_i = \sqrt{\lambda_i}$ son los valores singulares positivos de A, y luego

$$\sigma_i u_i = Av_i \quad \text{para todo } i = 1, \ldots, k \tag{3.6}$$

Ahora si $k < m$, extendemos la base $\{u_1, u_2, \ldots, u_k\}$ de $Im(A)$ a una base ortonormal $\{u_1, u_2, \ldots, u_m\}$ de R^m; por lo que tenemos las siguiente igualdades:

$$\begin{aligned} Av_i &= \sigma u_i \quad && \text{para todo } i = 1, \ldots, k. \\ Av_i &= 0u_i = 0 \quad && \text{para todo } i = k+1, \ldots, n. \end{aligned}$$

Obteniendo la siguiente representación matricial

$$A \begin{bmatrix} v_1 & \ldots & v_n \end{bmatrix}_{n\times n} = \begin{bmatrix} u_1 & \cdots & u_m \end{bmatrix}_{m\times m} \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & 0_{(k)\times(n-k)} \\ & & \sigma_k & \\ \hline \multicolumn{3}{c|}{0_{(m-k)\times(k)}} & 0_{(m-k)\times(n-k)} \end{array}\right]_{m\times n}$$

$$A = \begin{bmatrix} u_1 & \cdots & u_m \end{bmatrix}_{m\times m} \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & 0_{(k)\times(n-k)} \\ & & \sigma_k & \\ \hline \multicolumn{3}{c|}{0_{(m-k)\times(k)}} & 0_{(m-k)\times(n-k)} \end{array}\right]_{m\times n} \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}_{n\times n} \tag{3.7}$$

siendo éste el SVD de A.

3.1.2. Equivalencia

La descomposición matricial observada en (3.7), puede particionarse mediante operaciones simples de la siguiente forma

$$A = \begin{bmatrix} u_1 & \cdots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_k^T \end{bmatrix} + \begin{bmatrix} u_{k+1} & \cdots & u_m \end{bmatrix} \begin{bmatrix} 0 \end{bmatrix} \begin{bmatrix} v_{k+1}^T \\ \vdots \\ v_n^T \end{bmatrix}$$

por lo que el SVD de A tomaría la forma equivalente

$$A = \begin{bmatrix} u_1 & \cdots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_k^T \end{bmatrix} \tag{3.8}$$

3.1.3. Algoritmo SVD

Observación: Se trabaja $\min n, m$*, para que el costo computacional disminuya al realizarse el SVD trabajando con una matriz* $A^T A$ *de menor dimensión.*

Por la construcción previamente desarrollada, resumimos la técnica SVD a continuación,

- Siendo una matriz $A \in \mathbb{R}^{m \times n}$ con $n \leq m$, se diagonalizará la matriz $A^T A = VDV^T$ obteniéndose una matriz V cuyas columnas $v_1, \ldots, v_n$ forman un conjunto ortonormal de eigenvectores de $A^T A$ que corresponden a eigenvalores $\lambda_1, \ldots, \lambda_n$, que sin pérdida de generalidad estén ordenados de mayor a menor. (En caso $n > m$ aplicar el proceso para A^T y luego tomar la transpuesta).
- Sean $\lambda_1, \ldots, \lambda_k$ los eigenvalores distintos de cero, luego formar una matriz D de dimensión $m \times n$ donde $D_{ii} = \sigma_i = \sqrt{\lambda_i}$ para todo $i = 1, \ldots, k$, y cero sus otros elementos.
- Tomar $u_i = \dfrac{Av_i}{\sigma_i}$ para todo $i = 1, \ldots, k$. Ademas por Proposición 2.2, $u_1, \ldots, u_k$ son eigenvectores de AA^T asociados a los mismos eigenvalores.
- Extender el conjunto ortonormal $\{u_1, \ldots, u_k\}$ a una base ortonormal $\{u_1, \ldots, u_m\}$ de R^m, y formar la matriz U asignando los vectores u_i como sus vectores filas.

Algoritmo 1: SVD

Datos: $A \in \mathbb{R}^{m \times n}$
Resultado: U, S, Vh `/* donde` $A = USVh$ `*/`

Function *SVD (A):*
  si $n \leq m$ **entonces**
    `/*` $V = [v_1, \ldots, v_n]$ `matriz ortogonal de eigenvectores de` $A^T A$`, y` D `lista de eigenvalores` λ_i `ordenadas de mayor a menor, con correspondencia` $v_i \longmapsto \lambda_i$ `*/`
    $V, D \leftarrow eig(A^T A)$;
    `/* Sean` $\lambda_1, \ldots, \lambda_k$ `distintos de cero. */`
    $U \leftarrow \left[\dfrac{Av_1}{s_1}, \ldots, \dfrac{Av_k}{s_k}\right]$; $S \leftarrow \{\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_k}\}$; $Vh \leftarrow [v_1, \ldots, v_k]^T$;
  en otro caso
    `/*` $V = [v_1 \ldots v_m]$ `matriz ortogonal de eigenvectores de` AA^T`, y` D `lista de eigenvalores` λ_i`, con correspondencia` $v_1 \longmapsto \lambda_i$ `*/`
    $V, D \leftarrow eig(AA^T)$;
    `/* Sean` $\lambda_1, \ldots, \lambda_k$ `distintos de cero. */`
    $U \leftarrow [v_1, \ldots, v_k]$; $S \leftarrow \{\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_k}\}$; $Vh \leftarrow \left[\dfrac{A^T v_1}{s_1}, \ldots, \dfrac{A^T v_k}{s_k}\right]^T$;
  fin si
  devolver *U, diag(S), Vh* `/* diag(S) forma una matriz diagonal */`

3.2. Análisis de Componentes Principales (PCA)

El análisis de componentes principales, *Principal Component Analysis* (PCA), representa una herramienta para el desarrollo de este aprendizaje no supervisado, el cual simplifica estructuras que se encuentren en grandes dimensiones, como lo es en la aplicación de compresión de imágenes; por ejemplo, una imagen puede tener un tamaño 256×256, lo que significa que esta se ve dividida en 65536 características, por tanto imágenes en dicho tamaño representan datos con una muy alta dimensionalidad, y es aquí cuando es muy útil la técnica.

PCA reestructura datos en un espacio de menor dimensión, denominado **espacio reducido**, manteniéndose un alto porcentaje de la variabilidad de los datos iniciales. Estos datos reestructurados son conocidos como **patrones de clasificación**, y son determinados por vectores que se denominan **componentes principales**.

3.2.1. Datos Ajustados (Media Normalizada)

Sea una cantidad M de datos n-dimensionales, es decir con n características, dados en el Cuadro 3.1.

	Muestras			
	ϕ_1	ϕ_2	$\cdots\cdots\cdots$	ϕ_M
Características	x_{11}	x_{12}	$\cdots\cdots\cdots$	x_{1M}
	x_{21}			x_{2M}
	$\vdots$		$\ddots$	$\vdots$
	$\vdots$			$\vdots$
	x_{n1}	x_{n2}	$\cdots\cdots\cdots$	x_{nM}

Cuadro 3.1: M datos, n dimensionales, iniciales; donde $\overline{\phi} = \dfrac{1}{M}\displaystyle\sum_{k=1}^{M} \phi_k = \begin{bmatrix} \overline{x_1} \\ \vdots \\ \overline{x_n} \end{bmatrix}$ contiene los valores medios de cada característica.

Normalizar la media, indica generar datos ajustados con media 0 como se muestra en el Cuadro 3.2, permitiendo centrar la distribución y manteniendo la variabilidad para cada una de las características.

	Muestras			
	Φ_1	Φ_2	$\cdots\cdots\cdots$	Φ_M
Características	$x_{11} - \overline{x}_1$	$x_{12} - \overline{x}_1$	$\cdots\cdots\cdots$	$x_{1M} - \overline{x}_1$
	$x_{21} - \overline{x}_2$			$x_{2M} - \overline{x}_2$
	$\vdots$		$\ddots$	$\vdots$
	$\vdots$			$\vdots$
	$x_{n1} - \overline{x}_n$	$x_{n2} - \overline{x}_n$	$\cdots\cdots\cdots$	$x_{nM} - \overline{x}_n$

Cuadro 3.2: M datos, n dimensionales, normalizados.
Siendo $\Phi_k = \phi_k - \overline{\phi}$, $k = 1, \ldots, M$.
Por tanto, $\overline{\Phi} = \dfrac{1}{M}\displaystyle\sum_{k=1}^{M} \Phi_k = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$.

En efecto, la variabilidad en cada dimensión de los datos iniciales (Cuadro 3.1) y los datos ajustados (Cuadro 3.2) es la misma, esto ocurre por lo siguiente:
Siendo $x_{i1}, \ldots, x_{iM}$ los valores correspondientes a la característica X_i de los datos iniciales,

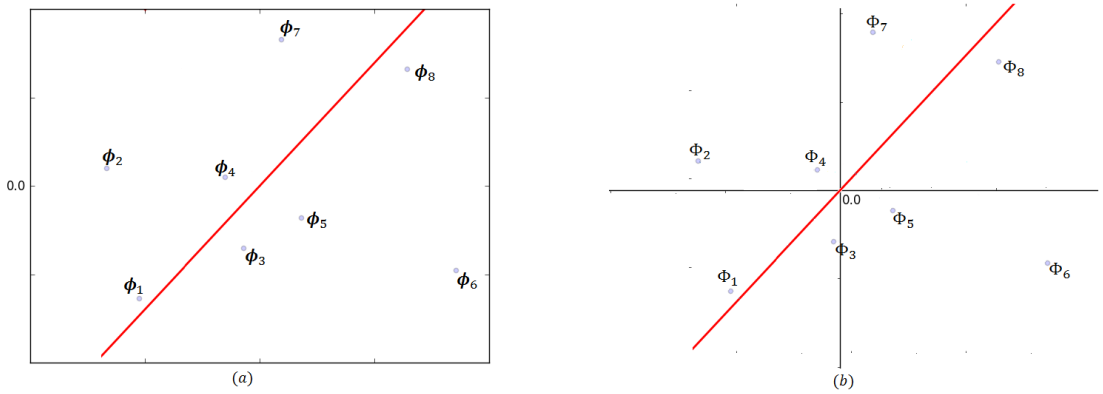


Figura 3.2: (a) representan datos no normalizados, (b) representan datos con la media normalizada.

entonces la varianza correspondiente es

$$Var(X_i) = \sum_{k=1}^{M} \frac{(x_{ik} - \overline{x_i})^2}{M}$$

y dado que la media es cero para los datos ajustados, donde $x_{i1} - \overline{x_i}, \ldots, x_{iM} - \overline{x_i}$ son los valores correspondientes a su característica X_i, entonces la varianza toma el mismo valor dado anteriormente.

3.2.2. Espacio reducido

Primero fijemos m, una dimensión menor que n; nuestro análisis solo bastaría en determinar un subespacio m-dimensional P de R^n en donde la distribución de nuevos datos, producto de la proyección de los datos ajustados hacia dicho subespacio, contenga el mayor porcentaje posible de la variabilidad en R^n; es decir, que reestructurando los datos ajustados n-dimensionales en nuevos datos m-dimensionales sobre un subespacio P, se logre minimice la pérdida de la varianza.

Para ello, si fijamos m, veamos que el subespacio P m-dimensional que satisfaga lo mencionado, es la generada por m eigenvectores ortonormales de la matriz de covarianza de los datos, que correspondan a los m eigenvalores de mayor valor. No es muy difícil pensar que la matriz de covarianza posea un conjunto de eigenvectores ortonormales, pues siendo ésta una matriz simétrica entonces se cumple lo indicado en el Teorema 2.1.

Optimización.

Nuestro problema es términos teóricos es el siguiente

minimizar la pérdida de variabilidad a causa de la reducción de dimensionalidad de n a m	$\equiv$	*maximizar la variabilidad generada por los datos reestructurados en la dimensión m.*

Sean $u_1, u_2, \ldots, u_m$ vectores ortonormales, a determinar, tal que generen el subespacio P.

$$P \;=\; \langle u_1, u_2, \ldots, u_m \rangle \,. \tag{3.9}$$

Luego por Teorema 2.3, se tiene que

$$proy_{\,P}(\Phi_k) \;=\; \sum_{i=1}^{m} (u_i^T \Phi_k) u_i \tag{3.10}$$

y por Proposición 2.5 se establecen las $proy_{\,P}(\Phi_k)$ en un nuevo sistema de coordenadas de dimensión m como

$$\begin{bmatrix} \Theta_1 & \Theta_2 & \cdots & \Theta_M \end{bmatrix} = U \begin{bmatrix} \Phi_1 & \Phi_2 & \cdots & \Phi_M \end{bmatrix} \;\text{, con } U = \begin{bmatrix} u_1^T \\ \vdots \\ u_m^T \end{bmatrix} \tag{3.11}$$

donde U es la matriz de rotación, estableciéndose un espacio reducido R^m, donde las nuevas coordenadas de cada dato reestructurado están dadas en el Cuadro 3.3.

	Muestras			
	Θ_1	Θ_2	$\cdots\cdots\cdots$	Θ_M
Características	$u_1^T\Phi_1$	$u_1^T\Phi_2$	$\cdots\cdots\cdots$	$u_1^T\Phi_M$
	$u_2^T\Phi_1$			$u_2^T\Phi_M$
	$\vdots$		$\ddots$	$\vdots$
	$\vdots$			$\vdots$
	$u_m^T\Phi_1$	$u_M^T\Phi_2$	$\cdots\cdots\cdots$	$u_m^T\Phi_M$

Cuadro 3.3: M datos, m dimensionales, reestructurados.

$$\overline{\Theta} \;=\; \frac{1}{M}\sum_{k=1}^{M} \Theta_k \;=\; \begin{bmatrix} \frac{1}{M}\sum_{k=1}^{M} u_1^T \Phi_k \\ \vdots \\ \frac{1}{M}\sum_{k=1}^{M} u_1^T \Phi_k \end{bmatrix} \;=\; \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Se tiene la media normalizada.

Dado que este conjunto de datos reestructurados mantiene la media normalizada, luego la varianza en cada dimensión de estos datos reestructurados, viene dada por

$$\lambda_i \;=\; \frac{1}{M}\sum_{k=1}^{M} (u_i^T \Phi_k)^2, \quad \text{para todo } i = 1, \ldots, m. \tag{3.12}$$

y por tanto la varianza total en R^m, como

$$\sum_{i=1}^{m} \lambda_i = \sum_{i=1}^{m} \frac{1}{M} \sum_{k=1}^{M} (u_i^T \Phi_k)^2, \quad \text{(Varianza total de los datos transformados)} \tag{3.13}$$

Ahora corresponde determinar el conjunto ortonormal $\{u_i\}_{i=1}^m$ tal que se maximicen las varianzas de (3.12), con el objetivo de maximizar la variabilidad total de (3.13). Esto es,

$$\text{máx} \left\{ \lambda_i = \frac{1}{M}\sum_{k=1}^{M} (u_i^T \Phi_k)^2 \right\} \text{ para todo } i = 1, \ldots, m. \tag{3.14}$$

Determinación del subespacio de dimensión reducida

Extendamos la base $\{u_1, \ldots, u_m\}$ ortonormal de P a una base ortonormal de R^n, siendo esta $\{u_1, \ldots, u_n\}$. Ahora usando el hecho de que $u_i^T u_i = 1$, veamos la relación que precisa (3.12).

$$\begin{aligned}
\lambda_i &= \frac{1}{M}\sum_{k=1}^{M}(u_i^T\Phi_k)(u_i^T\Phi_k) \\
\lambda_i u_i^T u_i &= \frac{1}{M}u_i^T\sum_{k=1}^{M}\Phi_k\Phi_k^T u_i \\
\begin{bmatrix} u_1^T \\ \vdots \\ u_i^T \\ \vdots \\ u_n^T \end{bmatrix}\lambda_i u_i &= \begin{bmatrix} u_1^T \\ \vdots \\ u_i^T \\ \vdots \\ u_n^T \end{bmatrix}\frac{1}{M}\sum_{k=1}^{M}\Phi_k\Phi_k^T u_i \\
\begin{bmatrix} u_1 & \ldots u_n \end{bmatrix}\begin{bmatrix} u_1^T \\ \vdots \\ u_n^T \end{bmatrix}\lambda_i u_i &= \begin{bmatrix} u_1 & \ldots u_n \end{bmatrix}\begin{bmatrix} u_1^T \\ \vdots \\ u_n^T \end{bmatrix}\frac{1}{M}\sum_{k=1}^{M}\Phi_k\Phi_k^T u_i \\
\lambda_i u_i &= \frac{1}{M}\sum_{k=1}^{M}\Phi_k\Phi_k^T u_i
\end{aligned}$$

$$\Rightarrow \quad \lambda_i u_i = \frac{1}{M}\begin{bmatrix}\Phi_1 & \ldots & \Phi_M\end{bmatrix}\begin{bmatrix}\Phi_1 \\ \vdots \\ \Phi_M\end{bmatrix}u_i \quad \text{donde } A = \begin{bmatrix}\Phi_1 & \ldots & \Phi_M\end{bmatrix} \tag{3.15}$$

Por tanto los vectores ortonormales $u_1, \ldots, u_n$ son eigenvectores de $C = \frac{1}{M}AA^T$, con $\lambda_1, \ldots, \lambda_n$ sus eigenvalores correspondientes. Con objetivo de maximizar (3.14), entonces $u_1, \ldots, u_m$ tendrán que ser correspondidos a los eigenvalores de mayor valor. Además, C es la matriz de covarianza de los datos (Proposición 3.6).

Como se observó en (3.12), cada eigenvector u_i mantiene una varianza λ_i, y la técnica SVD es útil para determinarlas aplicandose sobre A. De forma que u_i sea más representativa, λ_i asumimos que están ordenados en forma decreciente. (Ver Figura 3.5)

$$\begin{array}{ccccccc}
\lambda_1 & \geq \ldots \geq & \lambda_m & \geq \ldots \geq & \lambda_n & \geq 0 \\
\uparrow & & \uparrow & & \uparrow & & \\
u_1 & & u_m & & u_n & &
\end{array} \tag{3.16}$$

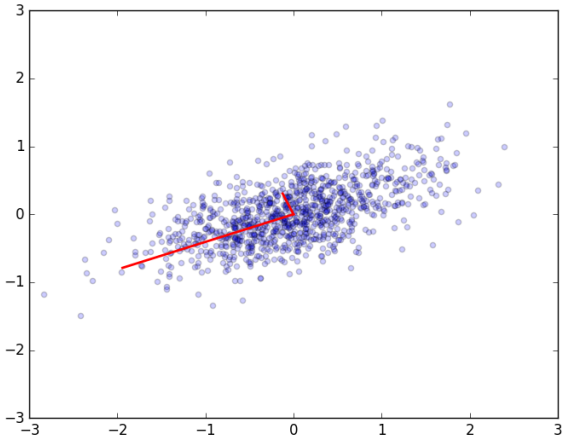


Figura 3.3: Visualización de la varianza explicada por los eigenvectores de la matriz de covarianza de datos bidimensionales, donde el eigenvector que mejor explica la varianza es la que corresponde al eigenvalor más alto.

3.2.3. Patrones de clasificación

Los datos n dimensionales reestructurados en datos m dimensionales, en base al espacio reducido, son denominados como patrones de clasificación.
Estos datos se presentan en el Cuadro 3.3 y, por su estructura matricial su cálculo se determina como:

$$\begin{bmatrix} \Theta_1 & \Theta_2 & \cdots & \Theta_M \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \\ \vdots \\ u_m^T \end{bmatrix} \begin{bmatrix} \Phi_1 & \Phi_2 & \cdots & \Phi_M \end{bmatrix} \tag{3.17}$$

Donde $u_1, \ldots, u_m$ son los m eigenvectores de la matriz de covarianza de los datos n dimensionales, que corresponden a los m eigenvalores de mayor valor.

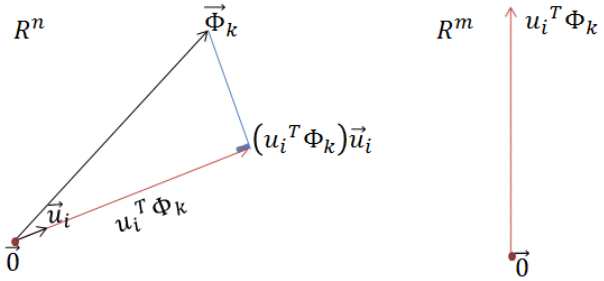


Figura 3.4: $\Phi_k \in R^n$. Los vectores ortonormales $u_1, u_2, \ldots, u_m$ establecen un espacio de dimensión reducida R^m, donde la coordenada i-ésima de Φ_k es $u_i^T \Phi_k$.

3.2.4. Pérdida de variabilidad

Ahora análizaremos la proporción de la varianza total que se ve perdida por la reducción de dimensionalidad.
Por (3.12) y (3.13), tenemos que la varianza total de los patrones de clasificación es

$$\sum_{i=1}^{m} \lambda_i \quad \text{(Varianza total en } R^m\text{)} \tag{3.18}$$

Luego siendo C la matriz de covarianza, con X_i representando las variables de los datos n dimensionales, entonces $Traza(C) = \sum_{i=1}^{n} Var(X_i)$ indica la variabilidad total de los datos. Además Proposición 2.4, $Tr(C) = \sum_{i=1}^{n} \lambda_i$, donde λ_i son los eigenvalores de C. Por lo que

$$\sum_{i=1}^{n} \lambda_i \quad \text{(Varianza total en } R^n\text{)} \tag{3.19}$$

Por tanto el porcentaje de la varianza que se mantiene en el espacio reducido es

$$\frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \times 100\,\% \tag{3.20}$$

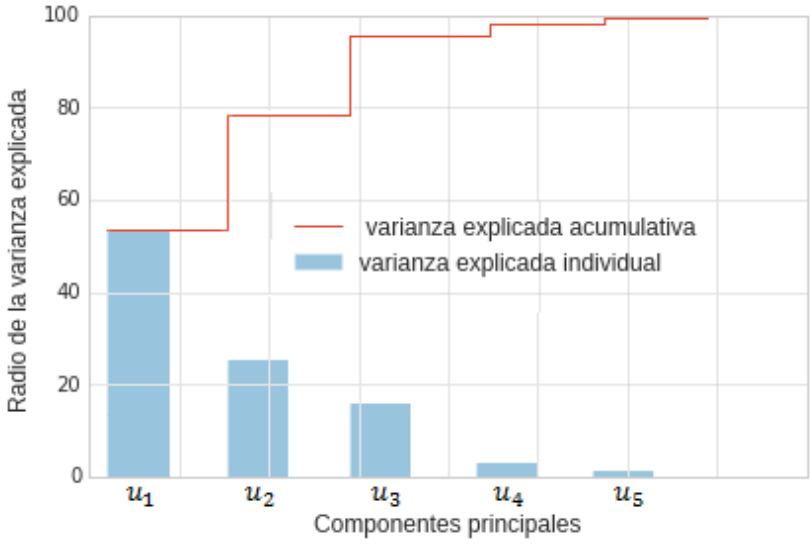


Figura 3.5: Porcentaje de varianza explicada por los eigenvectores de la matriz de covarianza de datos con 5 características. El análisis es el siguiente: A menor valor del eigenvalor, el eigenvector correspondiente mantiene una menor varianza.

3.2.5. Número de Componentes Principales

Anteriormente, se fijo m y el análisis se enfocó en determinar el mejor subespacio de dimensión m que maximice la variabilidad de los datos; ahora nuestro siguiente análisis será determinar el mínimo valor m, que será denominado como número de componentes principales, tal que el subespacio de dimensión m mantenga una alto porcentaje de la variabilidad (3.20), digamos no menos del 98 %.
Como se había determinado, el mejor subespacio de dimensión m será el generado por los m eigenvectores de la matriz de covarianza de los datos, que correspondan a los m eigenvalores de mayor valor (3.16).
Por tanto, para mantener un porcentaje de varianza(3.20) no menos del 98 %, tenemos que determinar el menor m tal que

$$\frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \times 100\,\% \geq 98\,\% \tag{3.21}$$

Es aquí cuando la técnica SVD fundamenta su utilidad, para determinar los eigenvalores u_i y los autovalores λ_i de tal forma que se mantenga tal porcentaje de varianza. La técnica nos dice que

$$A = \begin{bmatrix} \Phi_1 & \dots & \Phi_M \end{bmatrix} = \begin{bmatrix} u_1 & \dots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_k^T \end{bmatrix} \tag{3.22}$$

donde $\{u_1, \dots, u_k\}$ es un conjunto ortonormal de k eigenvectores de AA^T, cada una correspondiente a su eigenvalor σ_i^2 distinto de cero, asumiendo que están ordenados en forma decreciente, donde k es el rango de A. Dado que la matriz de covarianza es $\frac{1}{M}AA^T$, entonces los eigenvalores correspondientes serían $\lambda_i = \frac{1}{M}\sigma_i^2$. Además por (3.4), $u_{k+1}, \dots, u_n$ corresponderían a eigenvalores cero, por lo que estos vectores no mantienen algún porcentaje de variabilidad. Ahora para satisfacer (3.21), el porcentaje de variabilidad para un subespacio de dimensión m, tomaría la siguiente forma

$$\frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \times 100\,\% = \frac{\sum_{i=1}^{m} \frac{1}{M}\sigma_i^2}{\sum_{i=1}^{n} \frac{1}{M}\sigma_i^2} \times 100\,\% = \frac{\sum_{i=1}^{m} \sigma_i^2}{\sum_{i=1}^{n} \sigma_i^2} \times 100\,\% \geq 98\,\% \tag{3.23}$$

y solo bastaría con determinar el mínimo m tal que se cumpla lo anterior.

En Figura 3.5, podemos observar que los eigenvectores u_1, u_2, u_3 formarían el subespacio de dimensión 3 que maximice la variabilidad de los datos, pero observamos que solo acumularía no más del 95 %, sin embargo $u_1, \ldots, u_4$ mantienen una varianza acumulada quizas no menos del 98 %, por lo que el número de componentes principales sería 4.

3.2.6. Relación Variabilidad - Distancia

El análisis de componentes principales se enfoca en minimizar la pérdida de variabilidad, veamos que esta pérdida está directamente relacionada con la distancia total de los datos hacia su proyección en el subespacio P (3.9).

De (3.12), $M\lambda_i = \sum_{k=1}^{M} (u_i^T.\Phi_k)^2$ en donde λ_i alcancen valores máximos para $i = 1, \ldots, m$.

Esto último, nos determina que cada $\sum_{k=1}^{M} (u_i^T \Phi_k)^2$ alcance valores máximos. Además,

$$\sum_{i=1}^{m}\sum_{k=1}^{M}(u_i^T\Phi_k)^2 = \underbrace{\sum_{k=1}^{M}(u_1^T\Phi_k)^2}_{M\lambda_1} + \ldots + \underbrace{\sum_{k=1}^{M}(u_m^T\Phi_k)^2}_{M\lambda_m}$$

donde $\{u_1, \ldots, u_m\}$ es un conjunto ortonormal. Dado que $M\lambda_i \geq 0$ entonces, el problema de maximización vendría determinado como

$$\begin{cases} \text{máx}\left\{\sum_{i=1}^{m}\sum_{k=1}^{M}(u_i^T\Phi_k)^2\right\} & \text{con } u_i \in \mathbb{R}^n \text{ para todo } i = 1, \ldots, m \\ \text{donde } u_i^T.u_j = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases} \end{cases} \tag{3.24}$$

y en forma equivalente, dado que $\sum_{k=1}^{M} \|\Phi_k\|^2$ es un valor fijo, el problema de minimización siguiente

$$\begin{cases} \text{mín}\left\{\sum_{k=1}^{M}\|\Phi_k\|^2 - \sum_{i=1}^{m}\sum_{k=1}^{M}(u_i^T\Phi_k)^2\right\} & \text{con } u_i \in \mathbb{R}^n \text{ para todo } i = 1, \ldots, m \\ \text{donde } u_i^T u_j = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases} \end{cases} \tag{3.25}$$

Además, veamos lo siguiente

$$\begin{aligned}\sum_{k=1}^{M}\|\Phi_k\|^2-\sum_{i=1}^{m}\sum_{k=1}^{M}(u_i^T\Phi_k)^2 &= \sum_{k=1}^{M}\left(\|\Phi_k\|^2-\sum_{i=1}^{m}(u_i^T\Phi_k)^2\right)\\ &= \sum_{k=1}^{M}\left(\Phi_k-\sum_{i=1}^{m}(u_i^T\Phi_k)u_i\right)^T\left(\Phi_k-\sum_{i=1}^{m}(u_i^T\Phi_k)u_i\right)\\ &= \sum_{k=1}^{M}\left\|\Phi_k-\sum_{i=1}^{m}(u_i^T\Phi)u_i\right\|^2\end{aligned}$$

Por Teorema 2.3, tenemos que $\sum_{i=1}^{m}(u_i^T\Phi)u_i = proy_P(\Phi_k)$, entonces

$$\sum_{k=1}^{M}\|\Phi_k\|^2-\sum_{i=1}^{m}\sum_{k=1}^{M}(u_i^T\Phi_k)^2 = \sum_{k=1}^{M}\|\Phi_k-proy_P(\Phi_k)\|^2 \tag{3.26}$$

El problema enfocado a determinar el conjunto ortonormal $\{u_1,u_2,\ldots,u_m\}$ que maximice la variabilidad, tomaría un enfoque equivalente a

$$\begin{cases}\text{mín}\left\{\sum_{k=1}^{M}\|\Phi_k-proy_P(\Phi_k)\|^2\right\}\\ P=\langle u_1,u_2,\ldots,u_m\rangle \text{ con } u_i\cdot u_j=\begin{cases}1 & \text{si } i=j\\ 0 & \text{si } i\neq j\end{cases}\end{cases} \tag{3.27}$$

Por tanto se concluye que maximizar la variabilidad en un espacio reducido corresponde a minimizar la suma de distancias cuadradas de los datos hacia el subespacio.

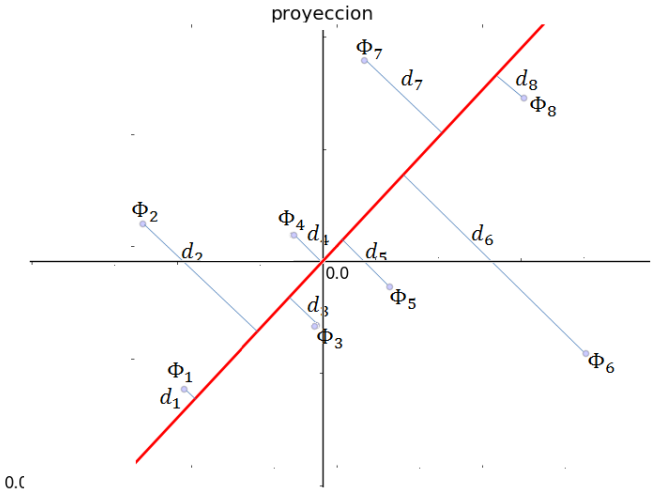


Figura 3.6: minimizar $\sum_{k=1}^{8}d_k=\sum_{k=1}^{8}\|\Phi_k-proy_P(\Phi_i)\|$ maximiza la variabilidad de los datos en el subespacio de menor dimensión (Recta).

3.2.7. Algoritmo PCA

El diseño del algoritmo de análisis de componentes principales, paso a paso se resume de la siguiente forma

- Contar con **datos multidimensionales**; es decir con n características ($n \geq 2$).
- **Normalizar la media** de los datos, con el objetivo de centrar la distribución sin perder la variabilidad. Para ello, se necesitará determinar la media en cada dimensión de los datos y luego substraer estos valores en tales dimensiones, obteniéndose **datos ajustados**.
- Aplicar la *descomposición de valores singulares* (SVD) sobre la matriz A cuyos vectores columna sean los datos ajustados, obteniéndose $A = U\Sigma V^T$.
  Donde Σ contiene en su diagonal los n valores singulares $\sigma_i \geq 0$ de A ordenados en forma decreciente. (σ_i^2 son eigenvalores de la matriz AA^T).
- Calcular el **número de componentes principales**, esto es calcular el menor valor m tal que
$$\frac{\sum_{i=1}^{m} \sigma_i^2}{\sum_{i=1}^{n} \sigma_i^2} \geq 0.98$$
- Calcular los **componentes principales**, estos son los m primeros vectores columna de la matriz U del SVD de A, y formaremos la matriz U_m cuyas columnas sean los componentes principales (en el mismo orden de como se encuentran en U).
- Calcular los **patrones de clasificación**, estos son los datos reestructurados en dimensión $m < n$, y se obtienen mediante $U_m A$.

Algoritmo 2: PCA

Datos: $X = [\phi_1, \ldots, \phi_M]$

/* donde ϕ_i son n dimensionales, $X = \begin{bmatrix} x_{11} & \ldots & x_{1M} \\ \vdots & \ddots & \vdots \\ x_{n1} & \ldots & x_{nM} \end{bmatrix}$ */

Function *PCA(X):*

- $Y \leftarrow \frac{1}{M} \sum_{i=1}^{M} \phi_i$;/* Media de los datos. */
- $\Phi_i \leftarrow \phi_i - Y$; /* Normalizando los datos. */
- $A \leftarrow [\Phi_1, \ldots, \phi_M]$
- $U, S, Vh \leftarrow SVD(A)$;
- /* $U = [u_1 \ \ldots \ u_n]$ donde u_i son los eigenvectores de AA^T, $S = diag(\sigma_1, \ldots, \sigma_k)$ donde $s_{ii}^2 = \sigma_i^2$ son los eigenvalores de AA^T, con correspondencia $u_i \longmapsto \sigma_i^2$ */
- Total= 0;
- **para** $i = 1$ *hasta* $i = k$ **hacer**
  - Total $\leftarrow$ Total + s_{ii}^2;
- **fin para**
- $m = 1$;
- **mientras** $\frac{\sum_{i=1}^{m} s_{ii}^2}{Total} < 0.98$ **hacer**
  - $m \leftarrow m + 1$;
- **fin mientras**
- $U_m \leftarrow [u_1 \ldots u_m]$;
- /* Determinando los patrones de clasificación */
- $\Theta \leftarrow U_m^T A$

devolver Y, A, m, U_m, Θ

Capítulo 4

Reconocimiento Facial

En este capítulo, enfocaremos el aprendizaje automatizado hacia el diseño de un modelo computacional que posea la habilidad del reconocimiento facial de un individuo. Como ha de intuirse, los modelos que realizan esta tarea resultan de gran interés en aplicación a una serie de problemas que incluyen identificación criminal, procesamiento de imágenes y, aún con mayor estudio, en la interacción humano-computadora.

Se podría pensar que desarrollar un tipo de modelo computacional que opte este aprendizaje representaría una tarea de alto nivel a causa de que el rostro humano es multidimensional y complejo, pero en este trabajo se presentará una forma sencilla, rápida y precisa en la que se pudiese llevar a cabo dicha tarea cuyo aprendizaje consta en codificar información relevante de un conjunto de imágenes faciales de individuos y almacenarlas en una base de datos que servirá como uso de comparación para información codificada de nuevas imágenes.

El método se desarrolla con un aprendizaje no supervisado y se basa en descomponer imágenes faciales en un conjunto de imágenes características, que para ser calculadas es necesario inicialmente representar cada imagen, estandarizada en escala de grises, como una matriz $n \times m$ en donde sus elementos corresponden a valores de píxeles que indican las intensidades de color de cada pixel según posición.

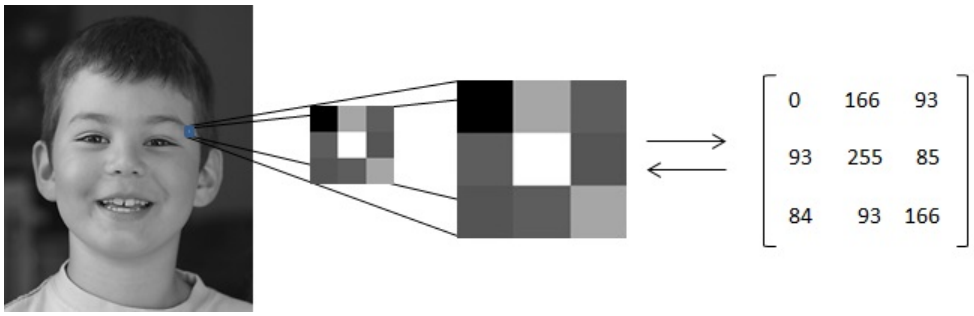


Figura 4.1: *Biyección* “Escala de grises - 0:255 pixeles”

Estas imágenes características serán llamadas *eigenfaces*, las cuales funcionan como componentes principales generadoras de un nuevo subespacio llamado *face space*; este nuevo espacio servirá para que nuevas imágenes faciales sean proyectadas y posterioremente clasificadas mediante patrones de clasificaciones que serán determinadas en el transcurso de este capítulo.

4.1. Eigenface y face-space

La técnica que motivó el uso de *eigenfaces* fue diseñada por *Sirovich y Kirby (1987)* mediante la implementación del análisis del componente principal (PCA), estudiada en el capítulo anterior.

La definición informal de los eigenfaces es que estos representan los eigenvectores, o vectores propios, de la matriz de covarianza de un conjunto de datos, que funcionan como datos de entrenamiento para nuestro modelo, por lo que nos mide de cierta forma las variaciones entre imágenes. En un inicio, se explicó que una imagen facial corresponde biyectivamente a una matriz de píxeles, que por asignación computarizada sus elementos varían de 0 a 255 (Figura 4.1); esto nos ofrece, matemáticamente, una idea base de lo que significa un eigenface, cuya función intrínseca será la caracterización entre imágenes faciales, haciendo uso de las matrices de píxeles.

Sea una imagen de tamaño $m \times n$ entonces su *matriz de píxeles* y *vector de píxeles*, el cual viene a ser la extensión vectorizada de la matriz de píxeles, son de dimensiones $n \times m$ y se representan como

$$\underset{\textit{matriz de píxeles}}{\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a_{n1} & \dots & \dots & a_{nm} \end{bmatrix}_{n\times m}} \equiv \underset{\textit{vector de píxeles}}{\{a_{11}, \dots, a_{1m}, a_{21}, \dots, a_{2m}, a_{31} \dots a_{nm}\}}$$

Como ha de saberse, varias de las imágenes faciales de individuos se presentan en distintos colores y tamaños, es por ello que en el anexo a este proyecto, se ha implementado un código Python que realice las tareas de tranformar una imagen en escala de grises, obteniéndose su correspondiente vector de píxeles, y viceversa; además de configurar el tamaño de la imagen con el objetivo que se trabajen con vectores de píxeles de igual dimensión.
(*Ver Códigos en* **Anexo-Pg. 47**).

Para un mejor lenguaje matemático; de ahora en adelante, una imagen facial será entendida como su matriz de píxeles y para el análisis haremos uso de K imágenes faciales.

Definición 4.1. (Media facial). *Sean $\phi_1, \phi_2, \ldots, \phi_K$ imágenes faciales, definimos la media facial Ψ, como*

$$\Psi = \frac{1}{K} \sum_{i=1}^{K} \phi_i \tag{4.1}$$

Definición 4.2. (Contrastes faciales). *Sean $\Phi_1, \ldots, \Phi_K$ imágenes faciales, con media facial Ψ, definimos los contrastes faciales como*

$$\Phi_i = \phi_i - \Psi. \tag{4.2}$$

Como ha de entenderse, los constrastes faciales representarían un conjunto de datos con la media normalizada (ver Cuadro 3.2).

Con objeto de la reducción de dimensional $n \times m$ usando la técnica (PCA), se determinarán los eigenvectores de la matriz de covarianza que generen un subespacio de características de dimensión menor en donde se minimice la pérdida de información; sin embargo veamos se puede reducir la dimensión manteniéndose el 100 % de la variabilidad.

Definición 4.3. (Eigenfaces). *Se denominan eigenfaces a los eigenvectores, de la matriz de covarianza de las imagenes faciales $\phi_1, \ldots, \phi_K$, que generan el subespacio de dimensión reducida usando el análisis de componentes principales.*

Definición 4.4. (Face-space). *Se denomina face-space al subespacio generado por los eigenfaces, se denotará como F.*

En primera instancia, tenemos que la matriz de covarianza vendría determinada, por Proposición 2.6, de la siguiente forma:

$$C = \frac{1}{K} \begin{bmatrix} \Phi_1 & \Phi_2 & \ldots & \Phi_K \end{bmatrix}_{(n\times m)\times K} \begin{bmatrix} \Phi_1^T \\ \Phi_2^T \\ \vdots \\ \Phi_K \end{bmatrix}_{K\times(n\times m)} \tag{4.3}$$

Si optando con imágenes faciales de tamaño $n \times m$, entonces la matriz de covarianza tendría una dimensión $(n \times m)^2$, y determinar los eigenvectores y autovalores tendría un gran costo computacional. Sin embargo, dado que la cantidad de datos es mucho menor

que tal dimensión, es decir $K \ll (n \times m)$, usando la Proposición 2.2,

$$D = \frac{1}{K} \begin{bmatrix} \Phi_1^T \\ \Phi_2^T \\ \vdots \\ \Phi_K \end{bmatrix}_{K \times (n \times m)} \begin{bmatrix} \Phi_1 & \Phi_2 & \dots & \Phi_K \end{bmatrix}_{(n \times m) \times K} \tag{4.4}$$

, donde tenemos que los eigenvalores $\lambda_1, \dots, \lambda_r$ $(r \leq K)$ distintos de cero de la matriz D , asociados a sus eigenvectores $\{v_1, v_2, \dots, v_r\}$, son los mismos autovalores distintos de cero de la matriz C asociados a sus eigenvectores $Av_1, Av_2, \dots, Av_r$, siendo $A = [\Phi_1, \dots, \Phi_K]$. Denotemos $u_i = Avi$. Sin pérdida de generalidad, supongamos $r = K$ y que los eigenvalores están ordenados en orden decreciente, para que el análisis de componentes principales mantenga su importancia.

Ahora por (3.18) y (3.19) tenemos la varianza total en el subespacio K dimensional, y la varianza total en el espacio $R^{n \times m}$.

$$\sum_{i=1}^{K} \lambda_i \qquad \text{(Varianza total en } R^K\text{)} \tag{4.5}$$

$$\sum_{i=1}^{n \times m} \lambda_i = \sum_{i=1}^{K} \lambda_i \qquad \text{(Varianza total en } R^{n \times m}\text{)} \tag{4.6}$$

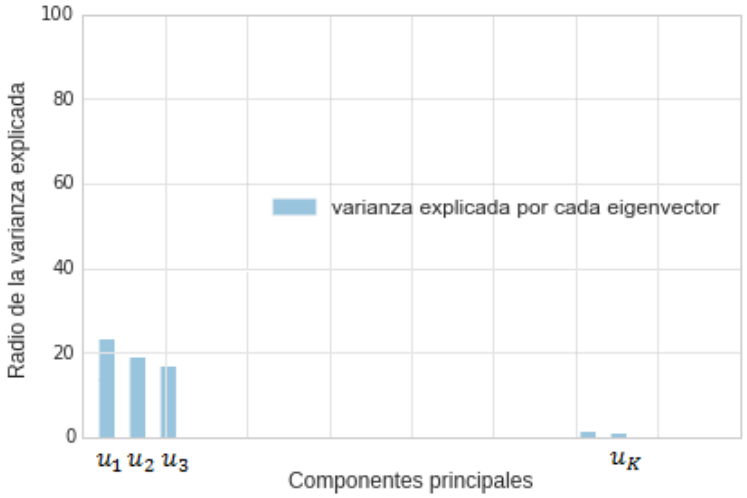


Figura 4.2: Varianza explicada.

Lo último se deduce, dado que los eigenvalores $\lambda_{K+1}, \dots, \lambda_{n \times m}$ de C son ceros, y por tanto sus eigenvectores correspondientes explican 0 % de la variabilidad.

Por tanto el porcentaje de varianza que se mantiene en el subespacio de dimensión R^K es

$$\frac{\text{Varianza total en } R^K}{\text{Varianza total en } R^{n\times m}} \times 100\,\% \quad = \quad 100\,\%. \tag{4.7}$$

¿Podríamos entonces decir que los eigenvectores $\{u_1, u_2, \ldots, u_K\}$ son los eigenfaces?
No necesariamente pues, como se ve en Figura 4.2, es posible que el porcentaje de variabilidad total que se mantenga en los eigenvectores correspondientes a eigenvalores de menor valor sea insignificante. En la práctica realizada por Sirivich y Kirby (1987), se observó que tomando los eigenvectores $u_1, u_2, \ldots, u_{115}$ que mantengan el 100 % de variabilidad en sus datos, solo los primeros 40 eigenvectores fueron suficientes para una excelente representación pues se generaba una pérdida del 2 % de variabilidad. Por este hecho, es útil el PCA para determinar el número de componentes principales que no genere pérdida de variabilidad significante, digamos no más del 2 %. Sea entonces K', el número de componentes principales.

4.2. Clasificación de las Imágenes Faciales

Determinado ya los eigenfaces $u_1, u_2, \ldots, u_{K'}$, nuestro siguiente paso constará en determinar los datos transformados, por (3.11),

$$\begin{bmatrix} \Theta_1 & \Theta_2 & \cdots & \Theta_K \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \\ \vdots \\ u_{K'}^T \end{bmatrix} \begin{bmatrix} \Phi_1 & \Phi_2 & \cdots & \Phi_K \end{bmatrix} \tag{4.8}$$

Estos datos transformados representan **los patrones de caracterización** para una imagen ϕ de entrada, permitiendo las siguientes clasificaciones:

1. **Reconocimiento de una imagen facial**: Se clasifica la imagen de entrada como una imagen facial caraterizada por el patrón correspondiente a una de las K imágenes faciales que modelan el aprendizaje.

2. **No reconocimiento de una imagen facial**: Se clasifica la imagen de entrada como una imagen facial que no es caracterizada por ninguno de los patrones de las K imágenes faciales que modelan el aprendizaje.

3. **Imagen no facial**: No se clasifica la imagen de entrada como una imagen facial.

Sea nuestra imagen de entrada ϕ, para iniciar su clasificación, primero determinaremos la correspondiente imagen contrastada, $\Phi = \phi - \Psi$, y su correspondiente dato reestructu-

rado por el subespacio de menor dimensión

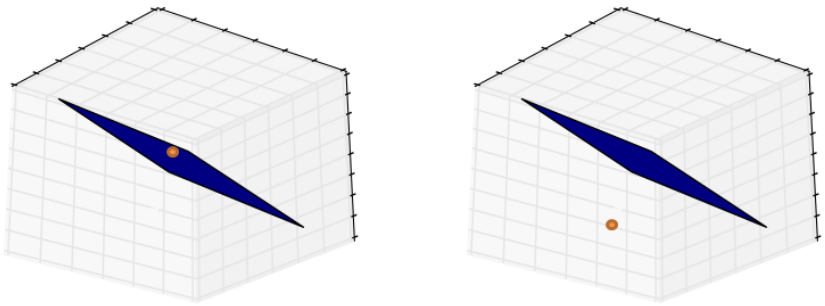
$$\Theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_{K'} \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \\ \vdots \\ u_{K'}^T \end{bmatrix} \Phi = \begin{bmatrix} u_1^T \Phi \\ u_2^T \Phi \\ \vdots \\ u_{K'}^T \Phi \end{bmatrix} \tag{4.9}$$

4.2.1. Imagen Facial - No Facial

Nuestra verificación inicial es determinar si la imagen de entrada es una imagen facial o no; esto es si el face-space la clasifica. Con esto último, nos referimos de que la imagen no pierda tanta variabilidad al ser proyectada hacia el face-space. Por (3.27), tenemos que tal pérdida está relacionada directamente con la distancia de la imagen hacia su proyección (A menor distancia, menor pérdida de variabilidad). Por medios empíricos, se establece un límite δ a esta distancia para su clasificación. (Ver figura 4.4)

$$\varepsilon^2 = \|\Phi - proy_F \Phi\|^2 \leq \delta \quad (Imagen\ facial) \tag{4.10}$$

donde por (4.1): $proy_F(\Phi) = \sum_{i=1}^{K'} \theta_i u_i = \sum_{i=1}^{K'} (u_i \cdot \Phi) u_i$.



(a) El subespacio clasifica el dato. (b) El subespacio no clasifica el dato

Figura 4.3: Ejemplo de un subespacio bidimensional, caracterizando un dato tridimensional.

4.2.2. Reconocimiento Facial

Luego habiéndose verificado que nuestra imagen de entrada es una imagen facial, buscaremos el patron Θ_i (4.8) que mejor caracterice a Θ (4.9), usando el mismo análisis anterior, es decir que la distancia entre ambos sea la menor.

$$\epsilon^2 = \text{mín}\{\epsilon_i^2 : \ \|\Theta - \Theta_i\|^2 = \epsilon_i^2, i = 1, \ldots, K'\}. \tag{4.11}$$

Se obtendrá el patrón que mejor clasifique la nueva imagen, pero aún no podemos concluir un reconocimiento facial por tanto bajo medios empíricos, se establece el límite δ a este valor mínimo.

$$\epsilon^2 \leq \delta \quad (Reconocimiento\ Facial) \tag{4.12}$$

Por tanto, de determinarse lo anterior mendiante el patrón Θ_i, entonces se dice que habrá **reconocimiento facial** de la imagen facial ϕ_i sobre ϕ.

En caso contrario se clasificaría como un "no reconocimiento" de una imagen facial.

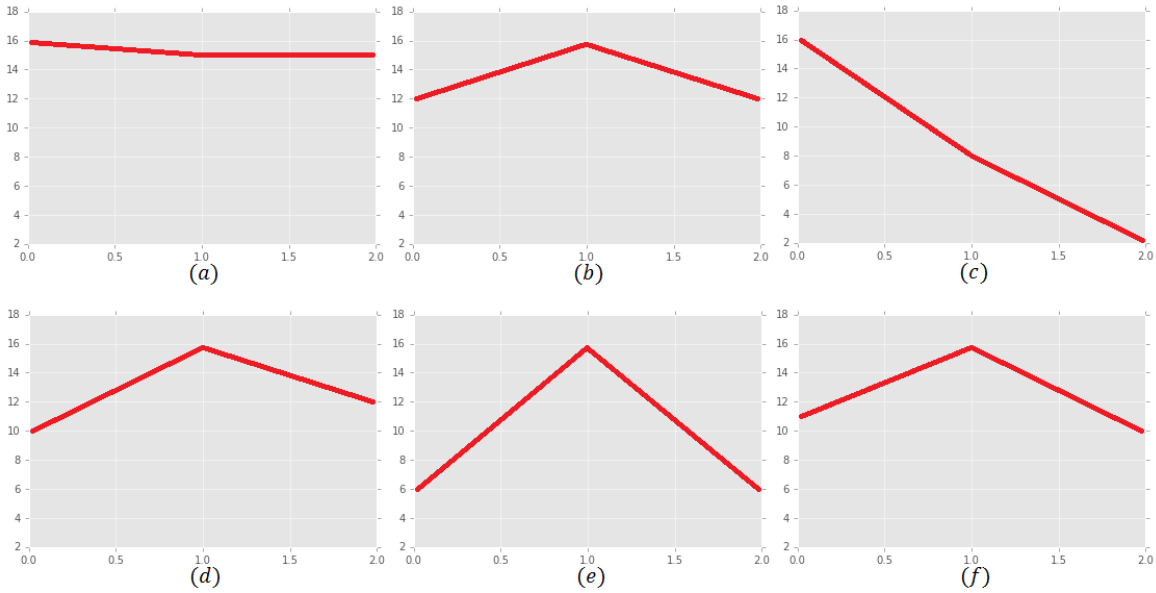


Figura 4.4: X: δ;Y: Número de clasificaciones correctas. Tomándose 16 imágenes faciales, de las K imágenes faciales de entradas, pero variando su: (a) iluminación - (b) inclinación de su rostro - (c) tamaño de cabeza - (d) iluminación e inclinación - (e) iluminación y tamaño de cabeza - (f) inclinación y tamaño de cabeza.

4.3. Modelamiento del aprendizaje

En resumen del análisis determinado en las secciones anteriores se modela el aprendizaje de la siguiente forma.

Entrenamiento de Modelo

1. Tomar K imágenes faciales, como datos de entrenamiento para el aprendizaje.
2. Transformar las imágenes a escala de grises y determinar su vector de píxeles correspondiente a cada una de ellas. Para un mejor lenguaje, el término "imágen facial" hará referencia a su vector de píxeles, siendo estas $\phi_1, \ldots, \phi_K$ por cada imagen facial.
3. Normalizar las imágenes facial, obteniendo lo que denominaremos imágenes contrastadas: $\Phi_1, \ldots, \Phi_K$, donde $\Phi = \phi - \Psi$. (Ver 4.2)
4. Calcular el número K' y los componentes principales $u_1, \ldots, u_{K'}$ tal que se mantenga una varianza no menos del 98 %, usando la técnica PCA con aplicación del SVD sobre $A = [\Phi_1, \ldots, \Phi_K]$. los componentes principales son denominados *eigenfaces* y el espacio generado por los eigenfaces será denominado como $face - space$.
5. Determinar los K datos transformados, patrones de clasificación: $\Theta_1, \ldots, \Theta_K$. (ver 4.8)

Tarea de reconocimiento:

6. Tomar una imagen de entrada para iniciar el reconocimiento facial.
7. Transformar la imagen de entrada en escala de grises y determinar su vector de píxeles ϕ y constrastarla mediante $\Phi = \phi - \Psi$.
8. Obtener el dato transformado de la imagen de entrada: Θ. (ver 4.8)
9. Determinar si la imagen de entrada clasifica como "imagen facial" (ver 4.10).
10. De clasificar como imagen facial, determinar el patrón Θ_i que lo caracterice estableciéndose el reconocimiento facial.

Tareas Opcionales

- Si se establece el reconocimiento facial de una imagen de entrada, entonces actualizar el modelo de aprendizaje, incorporando esta imagen en los datos de entrenamiento. El objetivo de esta tarea adicional es el de mejorar el reconocimiento facial sobre imágenes que este realmente relacionadas con una imagen facial de los datos de entrenamiento.

- Si la imagen es facial pero no se establece un reconocimiento facial, entonces actualizar el modelo de aprendizaje, incorporando esta imagen en los datos de entrenamiento. El objetivo de esta tarea adicional es el de aumentar la capacidad de reconocimiento facial para nuevas imágenes faciales que no se encuentren relacionadas con los datos de entrenamiento.

- Si la imagen es facial pero no se establece un reconocimiento facial, incorporar solo su dato transformado en la lista de patrones de clasificación. El objetivo de esta tarea adicional es es de contar con un nuevos patrones de clasificación que clasifiquen nuevas imágenes faciales.

Algoritmo 3: Reconocimiento Facial

Datos: K imágenes faciales como vectores de píxeles: $X = [\phi_1, \phi_2, \ldots, \phi_K]$

$\Psi, A, K', U_{K'}, B \leftarrow PCA(X)$

/* Ψ media facial, */

/* $A = [\Phi_1, \ldots, \Phi_K]$ contrastes faciales, */

/* K' componentes principales, */

/* $U_{K'} = [u_1, \ldots, u_{K'}]$, */

/* $B = [\Theta_1, \ldots, \Theta_{K'}]$ patrones de clasificación. */

Input: Imagen con vector de píxeles : ϕ

$\Phi \leftarrow \phi - \Psi$;/* Dato ajustado */

$\Theta \leftarrow U_{K'}^T \Phi$;/* Dato transformado */

$proy_F(\Phi) \leftarrow (u_1^T \Phi) u_1$;

para $i = 1$ *hasta* $i = K'$ **hacer**

  $proy_F(\Phi) \leftarrow proy_F(\Phi) + \sum_{i=2}^{K'} (u_i^T \Phi) u_i$

fin para

si $\|\Phi - proy_F(\Phi)\| \leq \delta$ **entonces**

  $j \leftarrow \min_{i \in \{1,\ldots,K'\}} \{i : \|\Theta - \Theta_i\|\}$;

  si $\|\Theta - \Theta_j\| \leq \delta$ **entonces**

    etiqueta $\leftarrow 1$

  en otro caso

    etiqueta $\leftarrow 2$

  fin si

en otro caso

  etiqueta$\leftarrow 0$

fin si

si *etiqueta* $== 0$ **entonces**

  No es imagen facial;

en otro caso

  si *etiqueta* $== 1$ **entonces**

    Existe reconocimiento facial de ϕ sobre ϕ_j;

  en otro caso

    Es imagen facial pero no existe reconocimiento facial;

  fin si

fin si

Capítulo 5

Conclusiones y Recomendaciones

La implementación de una técnica capaz de establecer reconocimiento facial, ha logrado determinarse en mayor parte con un buen uso del álgebra lineal y la estadística. Es claro que teóricamente, el diseño puede realizarse sin reducción de la dimensionalidad; sin embargo, desarrollar ello es fundamental pues el objetivo principal se basa en reducir tareas de cálculo que un sistema puediese realizar para disminuir su costo computacional permitiéndose diseñar un modelo de aprendizaje de una forma rápida y precisa.

El estudio de la técnica PCA, que fundamentó el presente trabajo sobre reconocimiento facial, nos permite concluir que existe la posibilidad de enfocar estudios en menores dimensiones, obteniendo resultados con pérdida de información insignificante.

Por como se construyen los *eigenfaces* se intuye que procesos análogos pueden tomarse en cuenta viéndose aplicadas tal vez en reconocimiento de estructuras celulares, huellas digitales, y demás; asignándoles los tipos de clasificación en forma similar a lo desarrollado.

La técnica desarrollada, nos permite capacitar a un sistema de una forma básica pero no representa una solución general pues una imagen en la vida real suele tener varias imágenes faciales incorporadas de individuos, además de encontrarse en varios entornos; por ello, enseñar a un sistema que desarrolle tareas con el objetivo de localizar una imagen facial resulta interesante. Más aún, si las imágenes faciales suelen tener inclinaciones y hasta giros.

Se recomienda un análisis más profundo en cuanto a la clasificación de las imágenes faciales pues técnicas como SVM, pueden ser implementadas para la realización de dicha tarea. Además con respecto a la función distancia que básicamente se usa para dicha tarea de clasificación, no debe verse limitada en hacer uso de ella pues existen funciones distancias alternativas que el álgebra lineal proporciona.

Anexos (Códigos)

Los códigos están elaborados en el lenguaje de programación Python

Imagen → Escala de grises - Vector de pixeles

```
from PIL import Image
import numpy as np

def gray_size_pixel(imagen,pix_ancho=180,pix_largo=200):
  #Escala de grises
  img = Image.open(imagen).convert('L')

  #Cambio de tamaño de la imagen
  img = img.resize((pix_ancho,pix_largo), Image.ANTIALIAS)

  #Vector de pixeles
  a=list(np.asarray(img))
  a=np.array(a)
  a=np.reshape(a, (np.product(a.shape),))

  return img, a

#Funcionamiento: Cambiar formato (jpg, jpe, png, ...) de ser necesario
imagen = 'imagen.jpe'
img,a = gray_size_pixel(imagen)
img.show()

#Guardado de la imagen en escala de grises
img.save('imagenGris.jpg')
```

Vector de pixeles → Imágen Escala de grises

```
def image_from_gray_pixel(a,pix_ancho=180,pix_largo=200):
  #Vector a Matriz
  a=a.reshape(pix_largo,pix_ancho)

  #Transformación a matriz de píxeles
  b = np.zeros((pix_largo,pix_ancho, 3), dtype=np.uint8)
  for i in range(pix_largo):
    for j in range(pix_ancho):
      for k in range(3):
        b[i][j][k] = a[i][j]

  #Imagen en Escala de grises
  img = Image.fromarray(b, 'RGB')
  return img

#Funcionamiento y guardado de la imagen
b = image_from_gray_pixel(a,pix_ancho=180,pix_largo=200)
b.save('imagenformada.jpg')
b.show()
```

Bibliografía

[1] Bishop, C. (2006). *Pattern Recognition and Machine Learning.* Cambridge University, United Kingdom.

[2] Smola, A. and Vishwanathan, S. (2008). *Introduction to Machine Learning.* Cambridge University, United Kingdom.

[3] Maleki, A. and Do, T (2014). *Review of Probability Theory.* Standford University, United States.

[4] Hastie. T.; Tibshirani, R. and Friedman, J. (2008). *The Elements of Statistical Learning - Data Mining, Inference and Prediction.* Standford University, United States.

[5] Anderson, D.; Sweeney, D. y William, T. (2008). *Estadística para Administración y Economía.* University of Cincinnati, United States.

[6] Turk, M. and Pentland, A. (1991). *Eigenfaces for Recognition.* Massachusetts Institute of Technology, United States.

[7] Smith, L. (2002). *A tutorial on Principal Component Analysis.* University of Otago, New Zealand.

[8] Sagitov, S. (2013). *Probability and Random Processes.* Chalmers University of Technology and Gothenburg University, Sweden.

[9] Cinlar, E. (2013). *Introduction to Stochastic processes.* Princeton University, United States.

[10] Kalman, D. (2002). *A Singularly Valuable Decomposition: The SVD of a Matrix.* The American University, United States.