



Programa de formación
**MACHINE LEARNING
AND DATA SCIENCE MLDS**

Facultad de
INGENIERÍA





Módulo 2

Introducción al Machine

Learning con *Python*

Unidad 5

**Aprendizaje NO supervisado:
Reducción de la dimensionalidad,
preprocesamiento y pipelines de ejecución**

Clase sincrónica

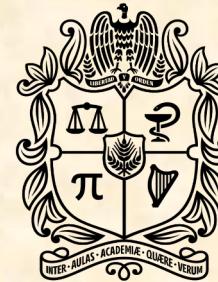


Bienvenida

Fabio Augusto Gonzalez, PhD.

<https://dis.unal.edu.co/~fgonza/>

fagonzalezo@unal.edu.co



UNIVERSIDAD
NACIONAL
DE COLOMBIA

Departamento de Ingeniería de Sistemas e Industrial
Facultad de Ingeniería
Universidad Nacional de Colombia
Sede Bogotá



Tabla de contenidos



1 Reducción de la Dimensionalidad



Maldición de la dimensionalidad



Aplicaciones



Enfoques



2 Análisis de Componentes Principales (PCA)



3 Preprocesamiento y *Pipelines* de Ejecución

Objetivos de aprendizaje



Unidad 5 - Aprendizaje no supervisado: reducción de la dimensionalidad, preprocessamiento y pipelines de ejecución

Al finalizar la unidad usted deberá ser capaz de:



1

Conocer los fundamentos del algoritmo PCA.



2

Implementar modelos de reducción de dimensionalidad con ayuda de la librería *scikit-learn*.

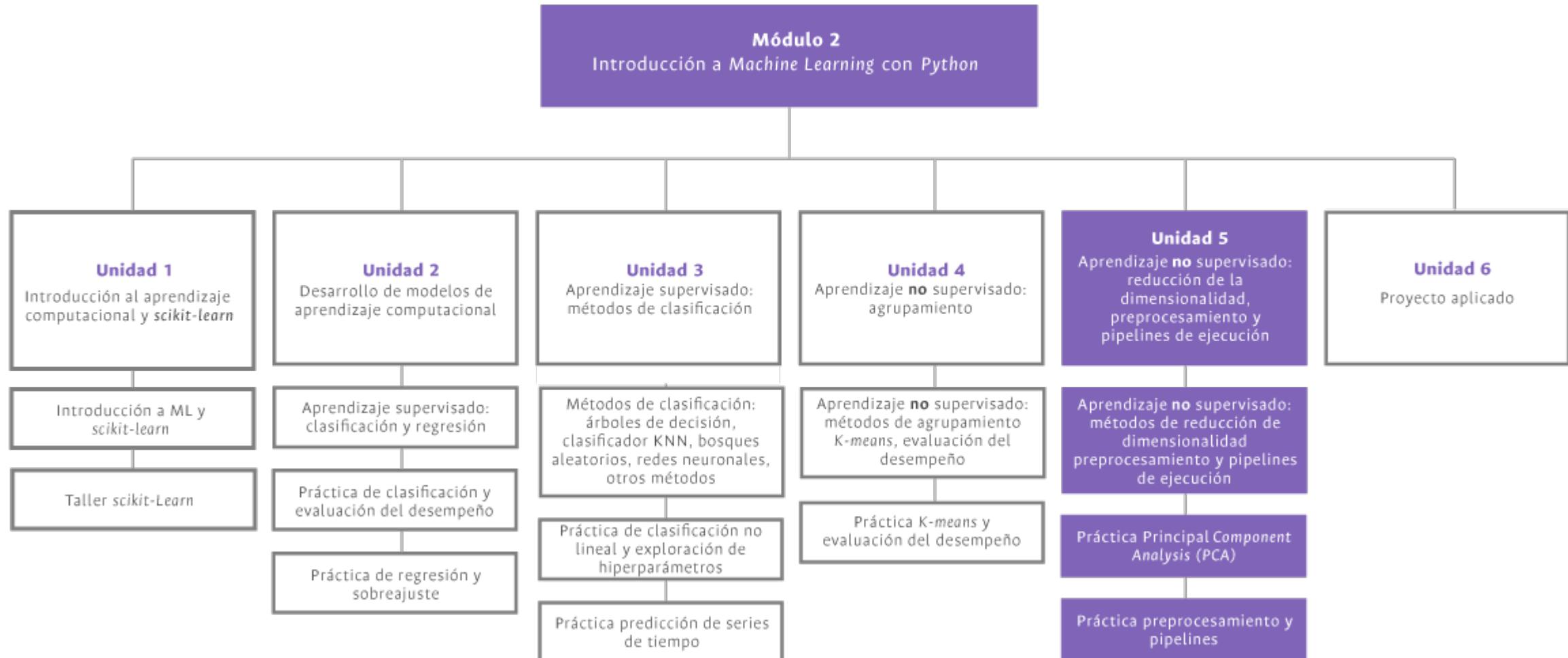


3

Definir un pipeline de ejecución en *scikit-learn* que integre procesamiento de datos, entrenamiento de modelos y predicción.

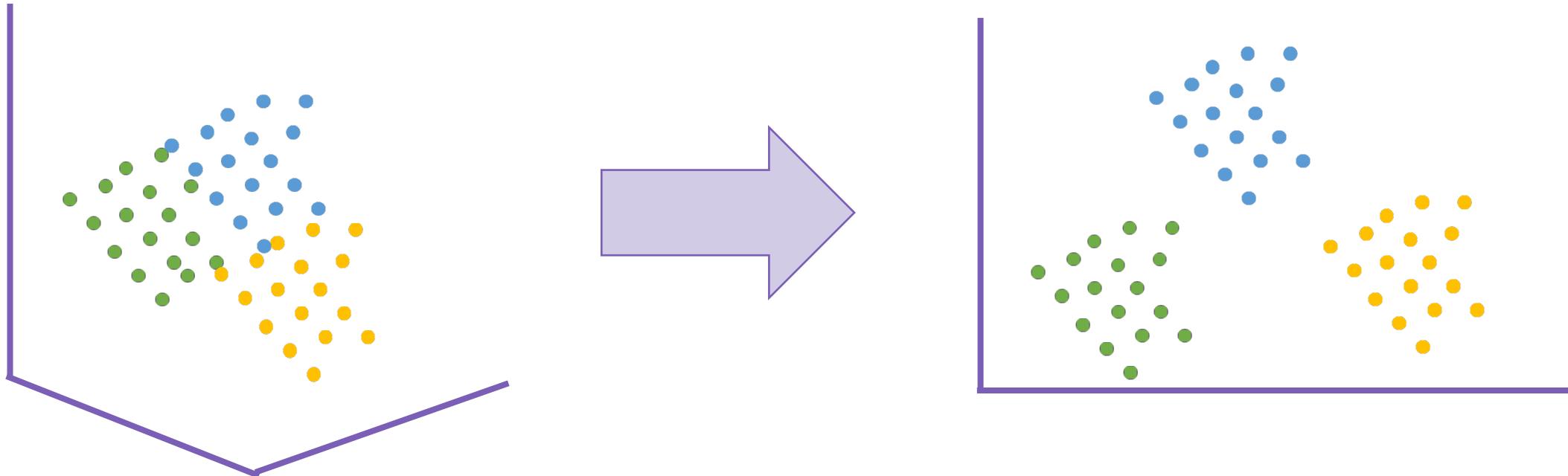


Mapa de contenidos de la unidad





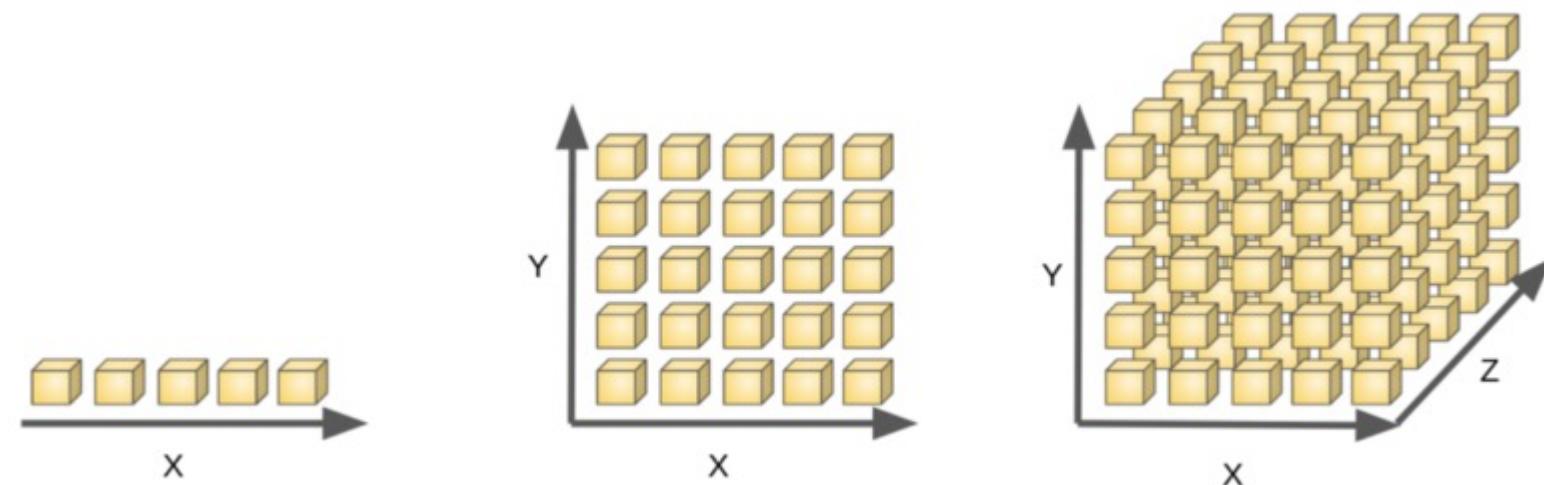
Reducción de la Dimensionalidad



Reducción de la dimensionalidad

 Maldición de la dimensionalidad

- La **maldición de la dimensionalidad** es la dificultad inherente para analizar, interpretar y visualizar datos a medida que el número de características o dimensiones aumenta.
- Visualizar datos con más de 3 dimensiones es una tarea extremadamente difícil y almacenar vectores con muchas dimensiones puede ser costoso.
- Cuando el número de dimensiones aumenta el espacio crece de manera exponencial.



Reducción de la dimensionalidad



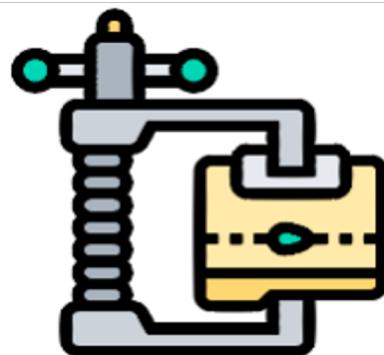
- Los datos altamente dimensionales pueden tener muchas características que son **redundantes** y podrían ser explicadas con la combinación de otras dimensiones.
- Las características de datos altamente dimensionales suelen estar **correlacionadas** y poseen una estructura intrínseca de menor dimensión.
- La reducción de la dimensionalidad aprovecha la redundancia y las correlaciones de los datos para crear una representación más compacta de los datos, sin perder información o perdiendo muy poca información.

Reducción de la dimensionalidad

 Aplicaciones

Compresión

La reducción de la dimensionalidad puede ser aplicada como una técnica de compresión. Este principio se aplica en formato como el jpeg y mp4, los cuales son algoritmos de compresión para imágenes y música.



Reducción de la dimensionalidad

 Aplicaciones

Agrupamiento

Un algoritmo de agrupamiento se aplica a los datos con dimensión reducida, para mejorar el desempeño de la visualización e identificación de grupos

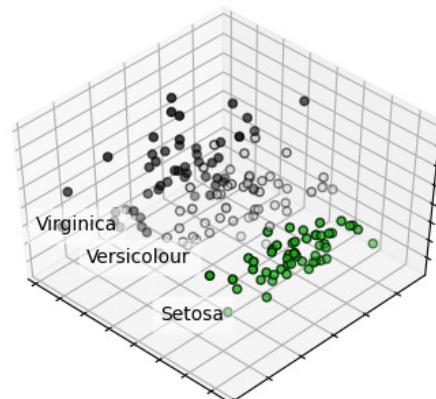


Reducción de la dimensionalidad

 Aplicaciones

Visualización

Al tratar datos altamente dimensionales, no se visualizan en su dimensión original, sino que se utiliza alguna técnica de reducción de la dimensionalidad, para reducir el número de estas a 2 o 3 y así ser visualizados.



Reducción de la dimensionalidad



Enfoques

Selección de características:

- Escogen un subconjunto de las características para reducir la dimensión y construir modelos.
- Generalmente requieren conocer las etiquetas de los datos,
- Los enfoques principales son los siguientes: de *filtro*, de *wrapper*, e *intrínsecos*.

Proyección de características

- Buscan transformar un conjunto de características a una representación más compacta.
- Las características originales se transforman a nuevas características que corresponden a combinaciones de las características iniciales.
- El algoritmo más popular es el Análisis de Componentes Principales (PCA, por sus siglas en inglés)
- Otros ejemplos incluyen: análisis de componentes independientes (ICA, por sus siglas en inglés), t-SNE y *Autoencoders*.

Reducción de la dimensionalidad

 Selección de características

Los métodos de **filtro** buscan subconjuntos de características basados en su relación con la etiqueta u objetivo.

Los métodos de **wrapper** buscan subconjuntos de características que se desempeñen bien en la tarea en cuestión (guiados por métricas de desempeño).

Los métodos **intrínsecos** son aquellos donde el algoritmo de aprendizaje realiza selección de características de manera implícita, por ejemplo, árboles de decisión.

Reducción de la dimensionalidad

 Selección de características

Información Mutua

La información mutua mide la cantidad de información que se puede obtener de una variable aleatoria dada otra. Se puede aplicar como una técnica de selección de características de filtro y se calcula la información mutua entre las características y la etiqueta, para seleccionar las características que aporte mayor información con respecto a la etiqueta.

Las clases `mutual_info_classif` y `mutual_info_regression` del paquete `sklearn.feature_selection` implementan esta técnica para etiquetas discretas y continuas, respectivamente.

Reducción de la dimensionalidad

 Proyección de características**Análisis de componentes principales**

Es un método de reducción de la dimensionalidad lineal. Encuentra los componentes que explican la mayor variación de los datos.

Análisis de componentes independientes (ICA)

Encuentra componentes estadísticamente independientes uno del otro.

t-SNE

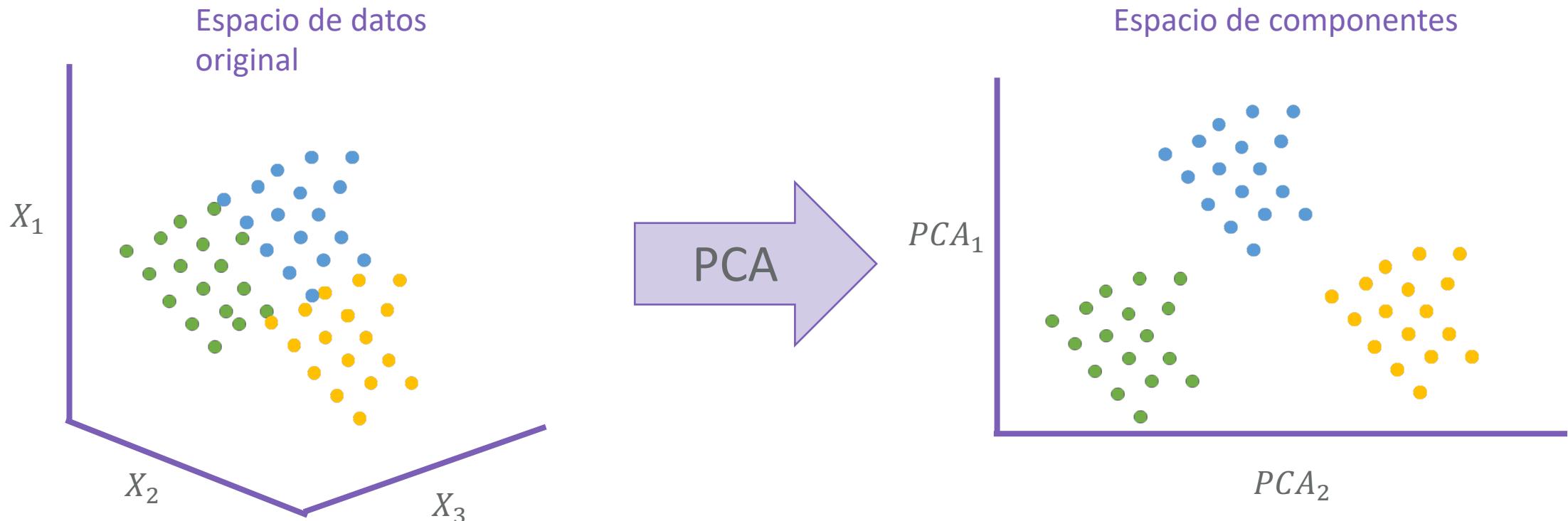
Es un algoritmo de reducción de la dimensionalidad no lineal especializado para la visualización de datos.

Autocodificadores o Autoencoders

Modelos de redes neuronales que buscan reconstruir los datos de entrenamiento. Un autocodificador reduce un ejemplo x a un vector con menos dimensiones z . A partir de z se reconstruye el vector original.

2

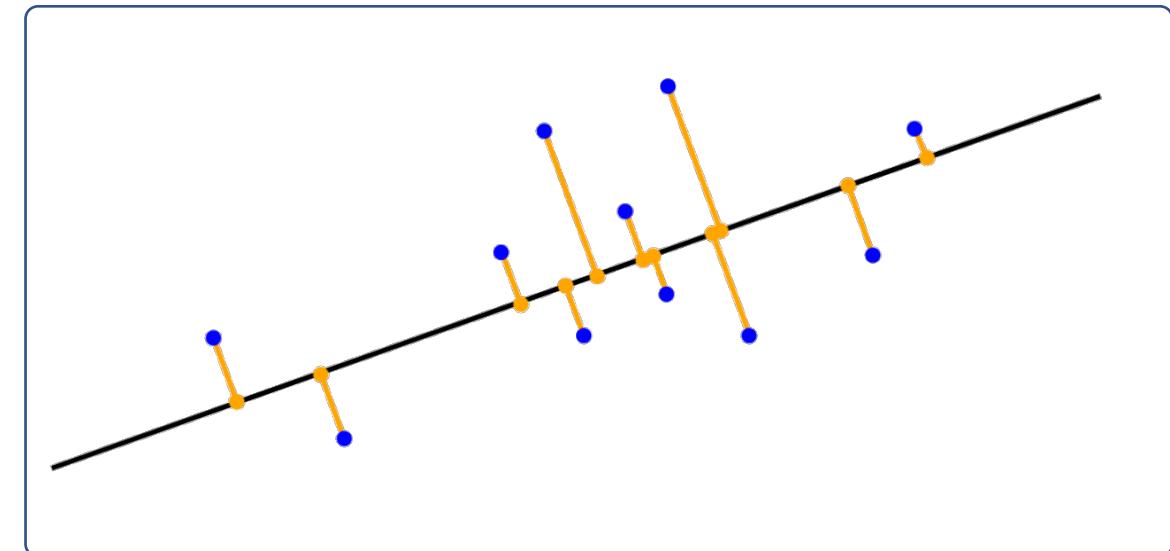
Análisis de Componentes Principales (PCA)



Análisis de Componentes Principales (PCA)

Generalidades

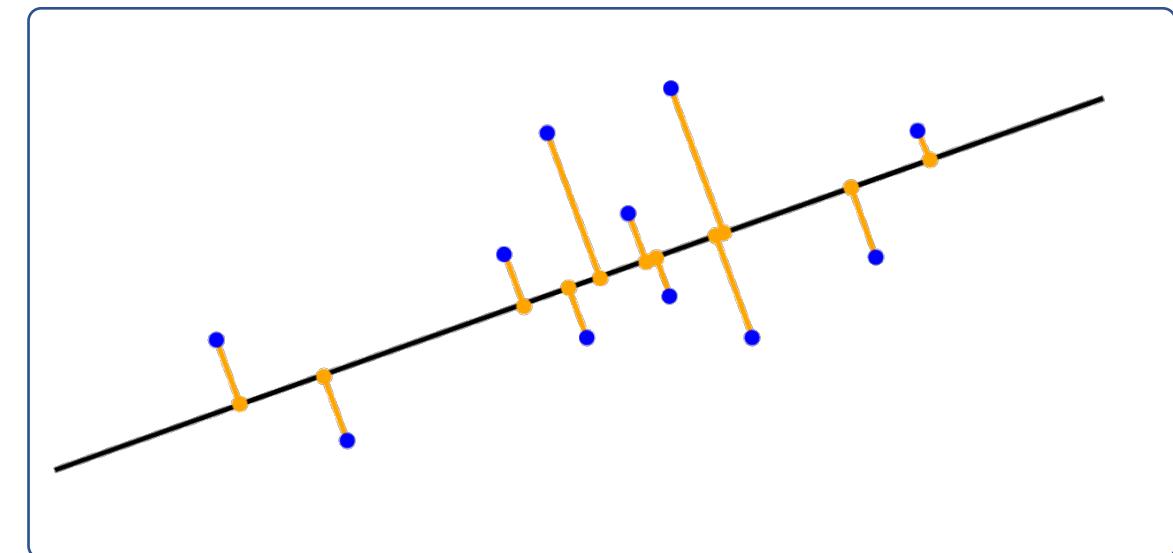
- PCA, por sus siglas en Inglés (*Principal Component Analysis*).
- Propuesto por Pearson (1901) y Hotelling (1933), lleva más de 100 años desde su introducción y es el método más popular.
- Los dos principales mecanismos que usa son la **rotación** y **proyección**.
- El método busca las direcciones (rotación) con varianza máxima.
- Estas direcciones corresponden a la base de un subespacio que mantiene tanta variación (separación de los datos) como sea posible.
- Los datos son proyectados a ese subespacio.



Análisis de Componentes Principales (PCA)

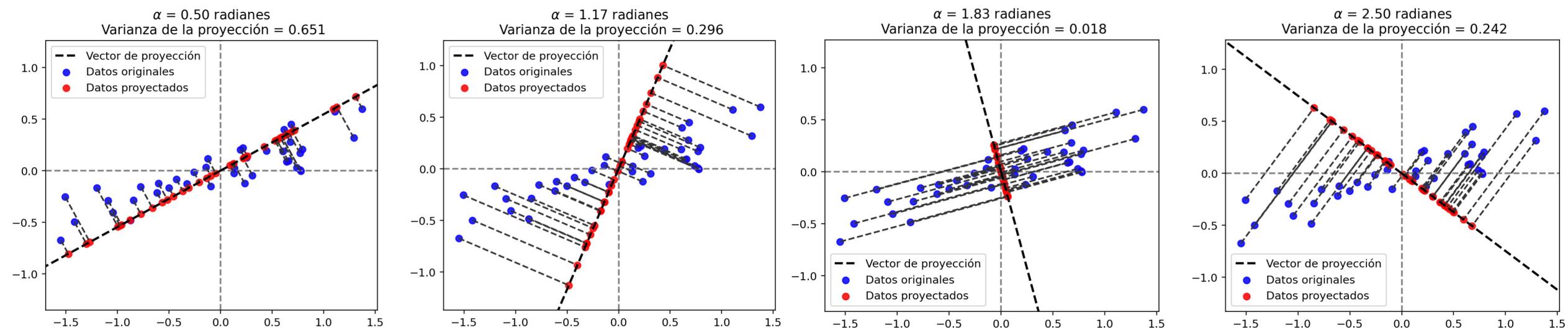
 Generalidades

- La primera dirección encontrada corresponde al primer **componente principal**.
- El proceso puede ser repetido, lo que permite encontrar vectores perpendiculares (no correlacionados), para encontrar los siguientes componentes principales.



Análisis de Componentes Principales (PCA)

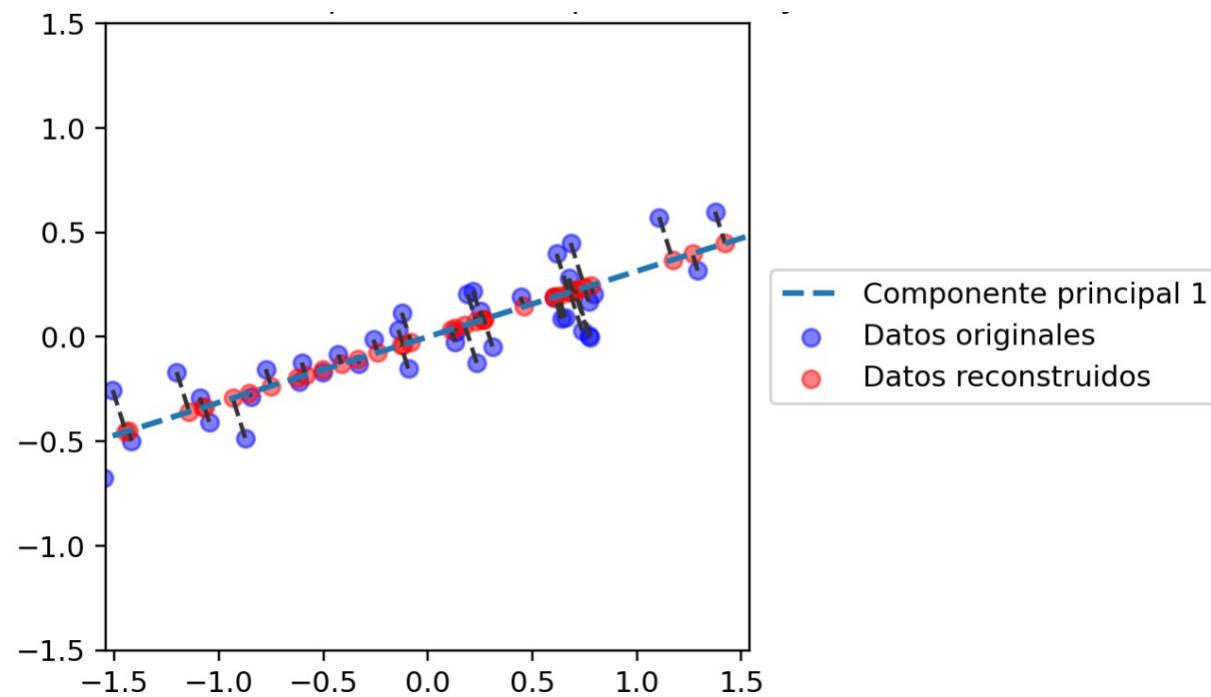
Varianza para diferentes direcciones de proyección



Análisis de Componentes Principales (PCA)

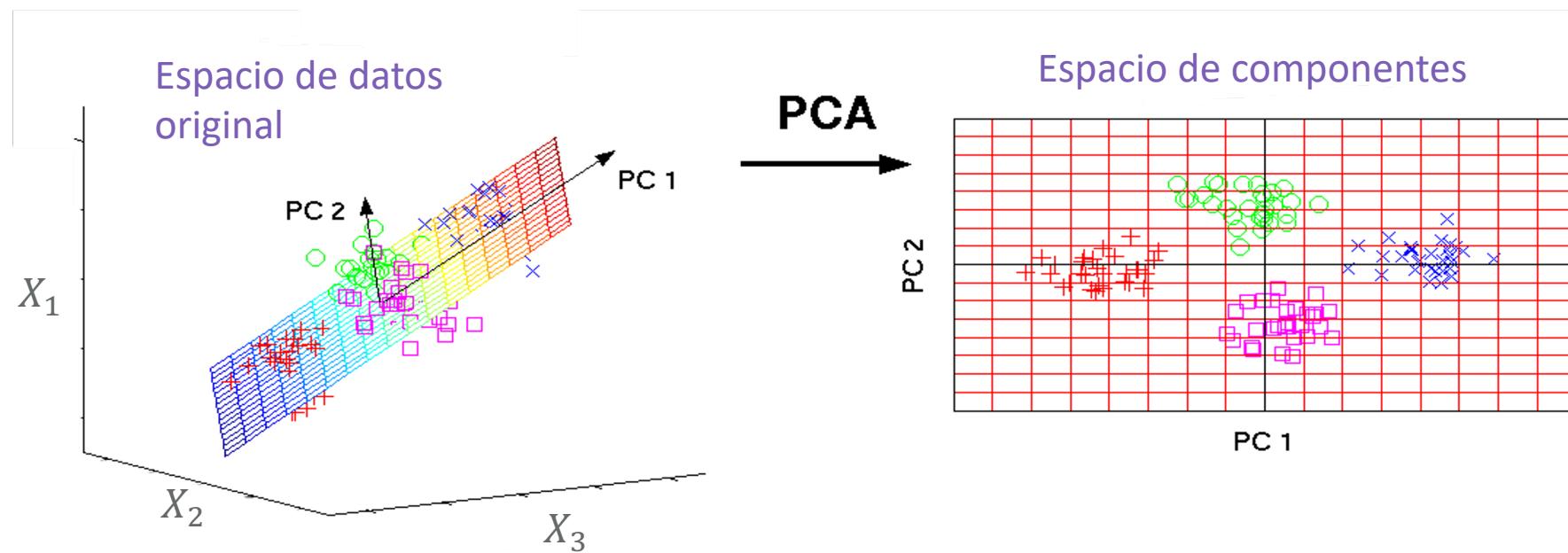
Dirección de mayor varianza

La siguiente figura muestra un conjunto de datos (azul) proyectado (rojo) a una línea la cual corresponde a la dirección de mayor varianza de los datos. Este sería el primer componente principal.



Análisis de Componentes Principales (PCA) / Método

- Los componentes principales forman una base ortogonal de los datos, que pueden ser encontrados a través de la descomposición de valores propios de la matriz de covarianza de los datos.
- La variación que explica un componente (vector propio) es proporcional al valor propio.
- Los componentes con mayor valor propio son los componentes principales.

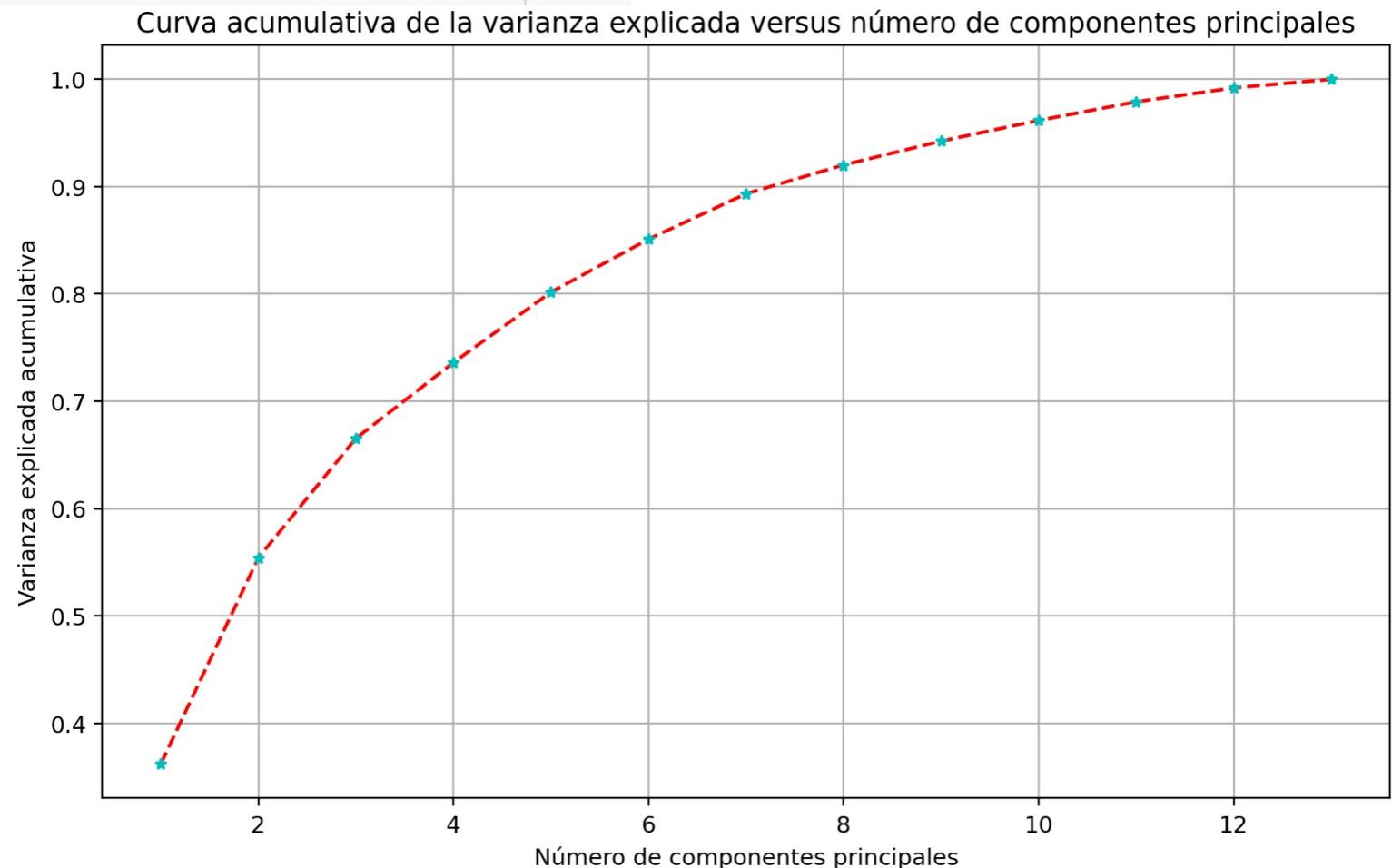


Análisis de Componentes Principales (PCA)

Varianza explicada

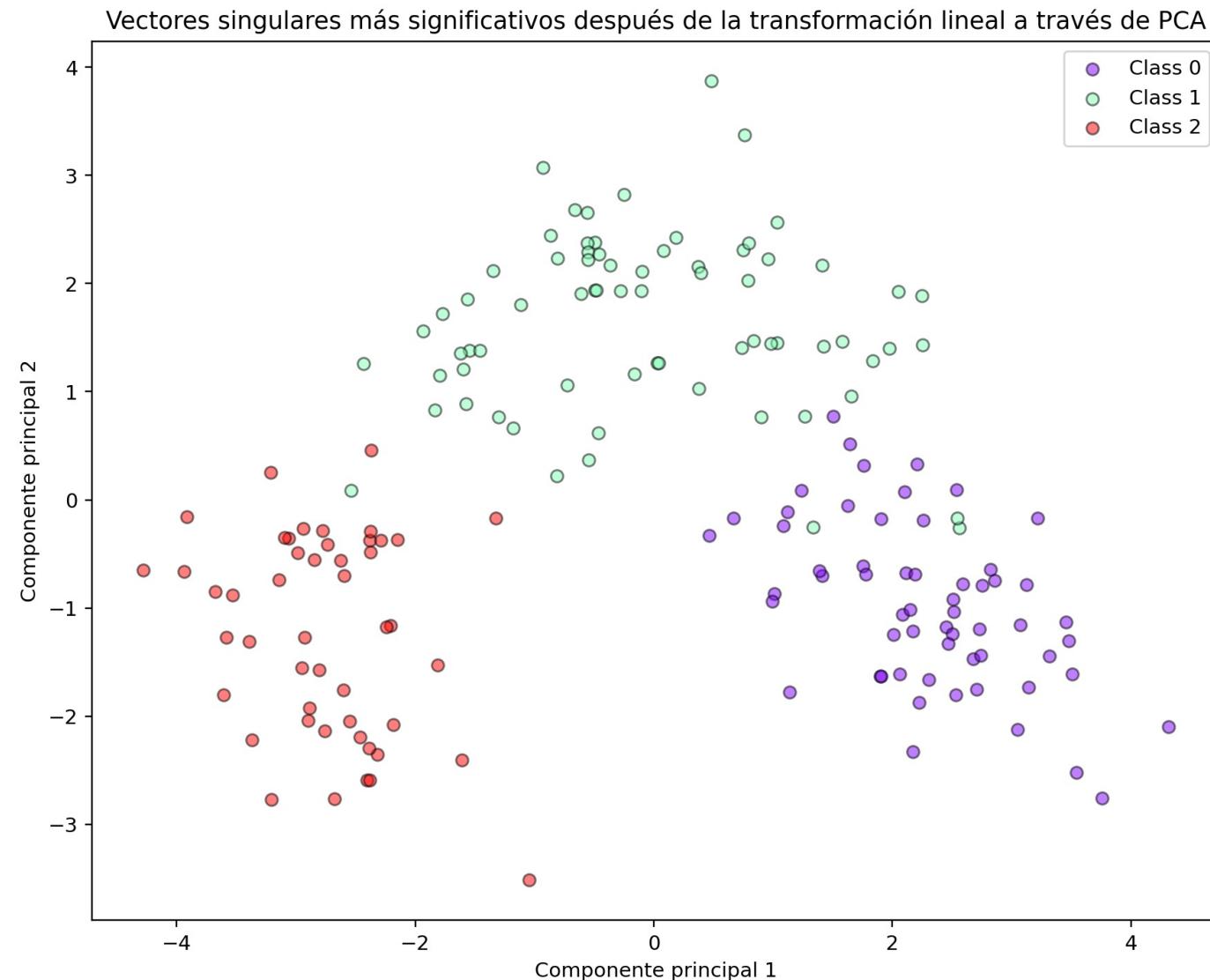
```
1 pca = PCA() #Si no se indica el número de componentes se usa la cantidad de columnas.  
2 transf = pca.fit_transform(X_scaled)  
3  
4 varianza_expl = pca.explained_variance_ratio_  
5  
6 print(varianza_expl)
```

```
[0.36198848 0.1920749 0.11123631 0.0706903 0.06  
0.04238679 0.02680749 0.02222153 0.01930019 0.01  
0.00795215]
```



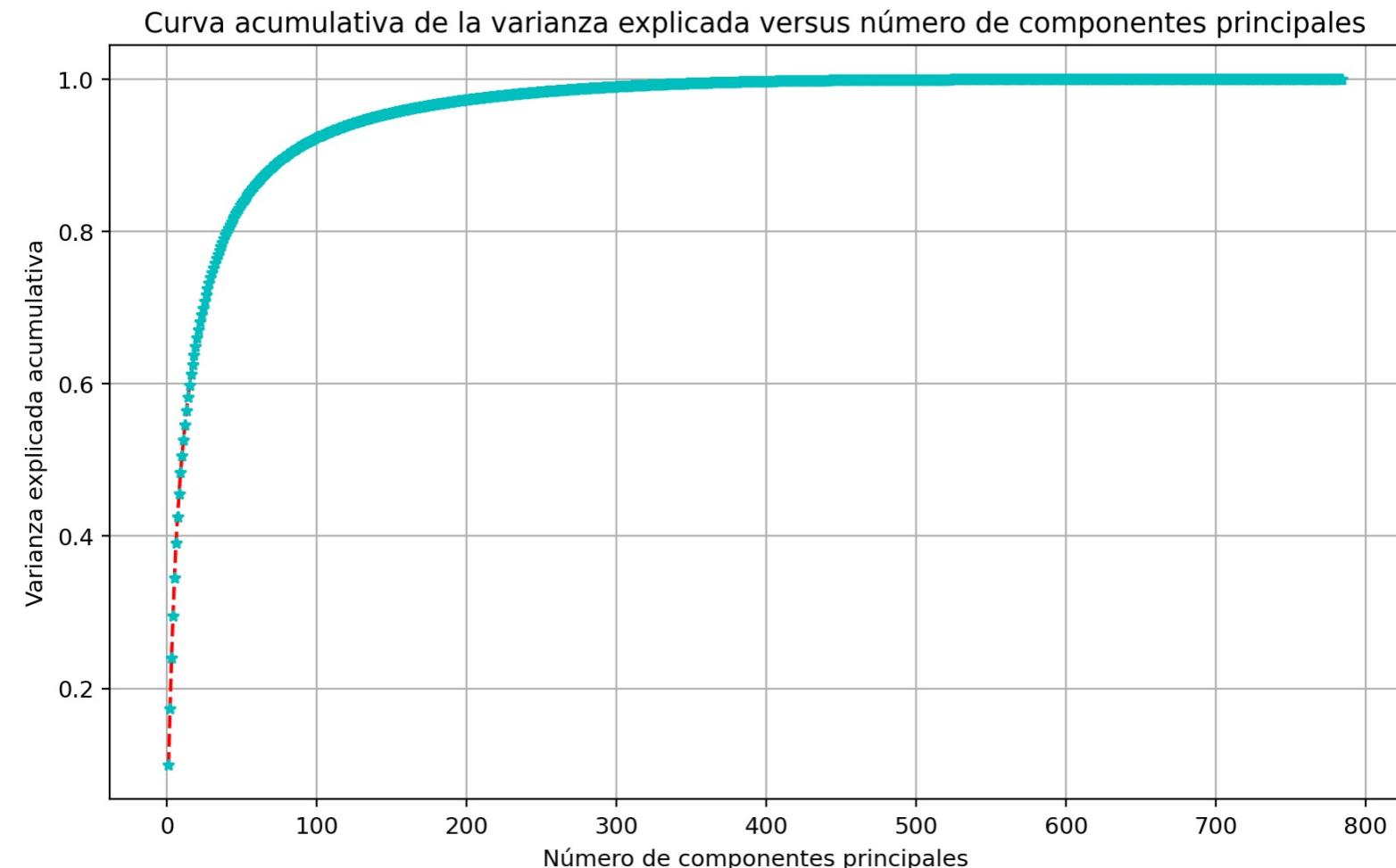
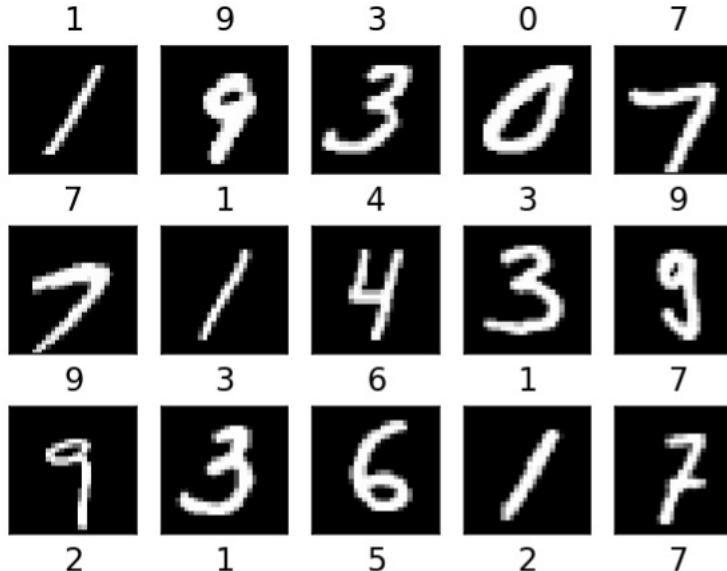
Análisis de Componentes Principales (PCA)

Visualización de los dos primeros componentes principales



Análisis de Componentes Principales (PCA)

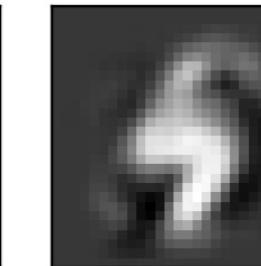
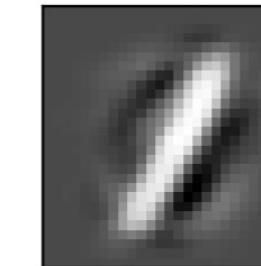
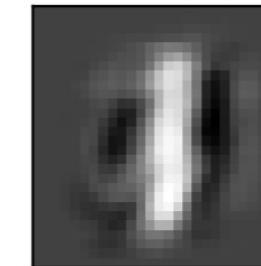
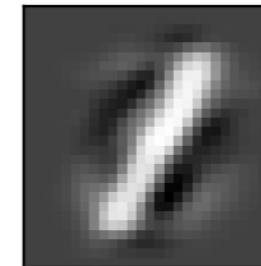
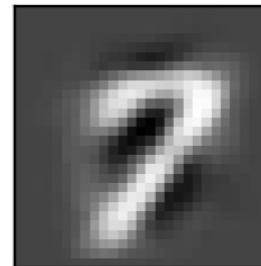
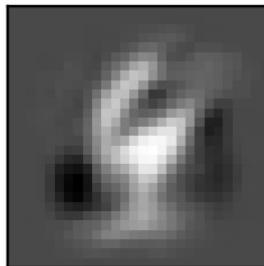
PCA con imágenes de dígitos



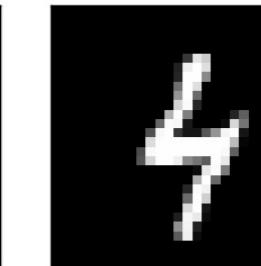
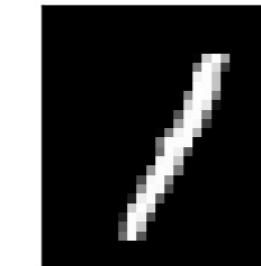
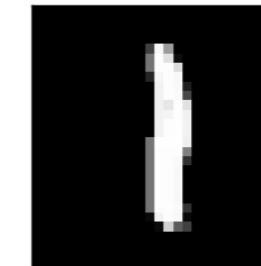
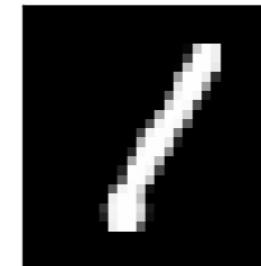
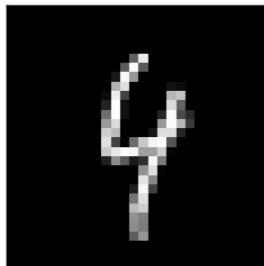
Análisis de Componentes Principales (PCA)

Reconstrucción (10 componentes)

```
1 pca = PCA(n_components=10)
2 Xp = pca.fit_transform(X_mnist)
3 Xr = pca.inverse_transform(Xp)
4
5 show_img_matrix_pca(X_mnist, Xr)
```



Reconstruida

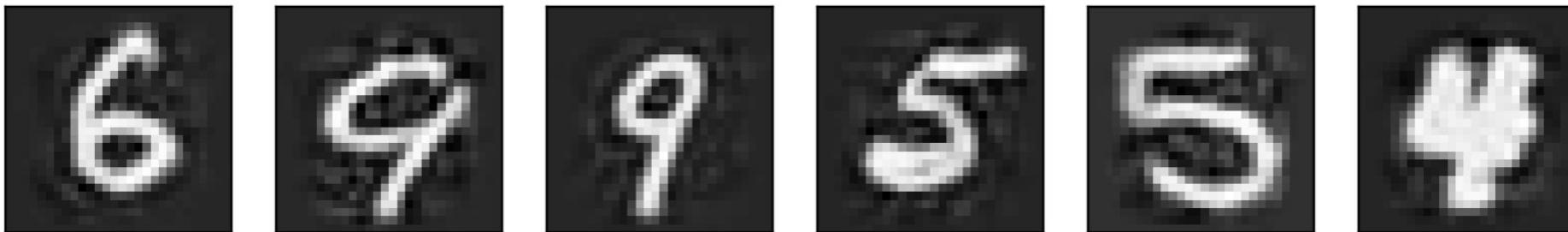


Original

Análisis de Componentes Principales (PCA)

Reconstrucción (100 componentes)

```
1 pca = PCA(n_components=100)
2 Xp = pca.fit_transform(X_mnist)
3 Xr = pca.inverse_transform(Xp)
4
5 show_img_matrix_pca(X_mnist, Xr)
```



Reconstruida



Original

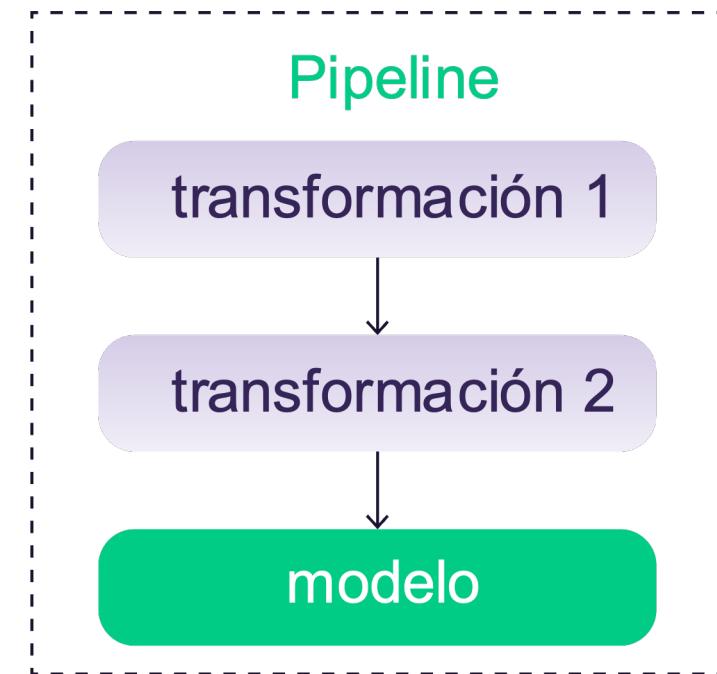
3

Preprocesamiento y Pipelines de ejecución



Preprocesamiento y *Pipelines* de ejecución

Los *pipelines* encadenan múltiples procesos en uno solo, por ejemplo, transformaciones de los datos, reducción de la dimensionalidad y entrenamiento de modelos supervisados y no supervisados.



Esquema de un *Pipeline*

Preprocesamiento y *Pipelines* de ejecución



Sklearn implementa esto en la clase *pipeline*, cuyo uso ofrece varias ventajas (Pedregosa et al., 2011):

Conveniencia y encapsulación

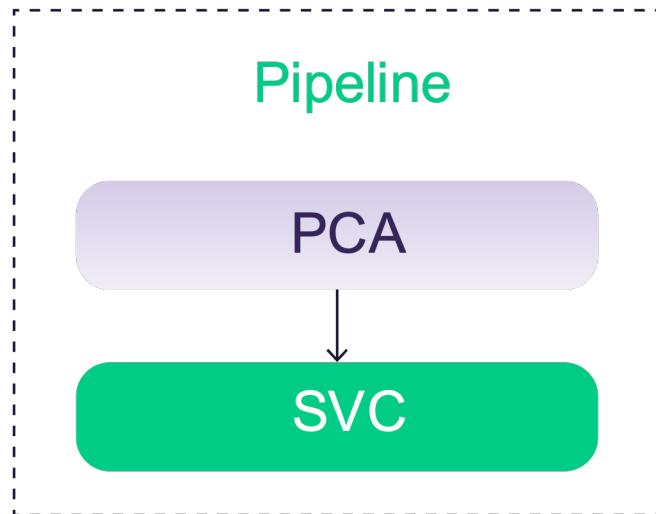
Solo se debe llamar *fit* y *predict* una vez en los datos para ajustar una secuencia completa de estimadores.

Búsqueda de hiperparámetros conjunta

Se podrá hacer búsqueda de los hiperparámetros de las transformaciones y modelo al mismo tiempo.

Seguridad

Los pipelines ayudan a evitar que se filtre información de los datos de prueba al modelo entrenado. Los mismos ejemplos se utilizan para entrenar las transformaciones y los modelos.

Preprocesamiento y *Pipelines* de ejecuciónDefinición de un *pipeline* en *sklearn*

```
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC
from sklearn.decomposition import PCA

pipe = Pipeline([('reduce_dim', PCA()),
                 ('clf', SVC())])
```

Preprocesamiento y *Pipelines* de ejecución

Pipeline para clasificación de documentos

```
1 # Pipeline con los tres estimadores a utilizar.  
2  
3 pipeline = Pipeline([  
4     ('vect', CountVectorizer()),  
5     ('tfidf', TfidfTransformer()),  
6     ('clf', RandomForestClassifier())  
7 ])
```

Calcula frecuencias de tokens

Aplicar peso TFIDF

Modelo de clasificación

```
1 parameters = {  
2     'vect__max_features': (1000, 2000),  
3     'vect__ngram_range': ((1, 1), (1, 2)),  
4     'tfidf__use_idf': (True, False),  
5     'clf__n_estimators': [50, 100],  
6     'clf__max_features': [0.1, 0.3]  
7 }
```

Especifica los parámetros a explorar

```
1 grid_search = GridSearchCV(pipeline, parameters, verbose=3, cv=3)
```

Crea el objeto GridSearchCV

```
1 grid_search.fit(text_data.data, text_data.target)
```

Realiza la exploración de parámetros

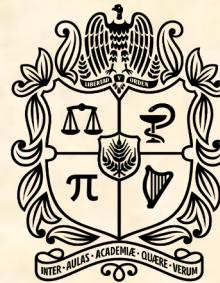


Despedida

Fabio Augusto Gonzalez, PhD.

<https://dis.unal.edu.co/~fgonza/>

fagonzalezo@unal.edu.co



UNIVERSIDAD
NACIONAL
DE COLOMBIA

Departamento de Ingeniería de Sistemas e Industrial
Facultad de Ingeniería
Universidad Nacional de Colombia
Sede Bogotá



Referencias

Deisenroth, M. P., Faisal, A. A., Ong, C. S. (2020). Mathematics for Machine Learning.
Cambridge University Press.

Gleeson, P. (Octubre 12 de 2017). Carga de trabajo computacional [Figura].
<https://www.freecodecamp.org/news/the-curse-of-dimensionality-how-we-can-save-big-data-from-itself-d9fa0f872335/>

Pedregosa, F., Mueller, A., Grisel, O., Niculae, V. Prettenhofer, P., Gramfort , A., Grobler,
J. Layton, R. VanderPlas, J. Joly, A. & Holt, B. (2011). Scikit-learn: Machine Learning in Python.
<https://scikit-learn.org/stable/modules/clustering.html#k-means>

Witten, I., Frank, E., Hall, M. & Pal, C. (2016). Data mining: practical machine learning tools and techniques.
Cambridge, MA: Morgan Kaufmann Publisher.



Recursos adicionales

Brownlee, J. (27 de noviembre del 2019). Cómo elegir un método de selección de funciones para el aprendizaje automático. Machine Learning Mastery. <https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/>

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, 24, 417–441, and 498–520.

Pearson, K. (1901). "On Lines and Planes of Closest Fit to Systems of Points in Space". Philosophical Magazine. 2 (11): 559–572. doi:10.1080/14786440109462720.

Universidad de Stanford (s.f.). CS231n: redes neuronales convolucionales para el reconocimiento visual. Consultado el 29 de julio del 2020. <https://cs231n.github.io/neural-networks-2/#datapre>

Raschka, S. (27 de enero del 2015). Análisis de Componentes Principales en tres pasos. SebastianRaschka. https://sebastianraschka.com/Articles/2015_pca_in_3_steps.html#1---eigendecomposition---computing-eigenvectors-and-eigenvalues



Derechos de imágenes

Gleeson, P. (2017). Muestra de la dimensionalidad. [Gráfica].

"Escaping the Curse of Dimensionality." <https://www.freecodecamp.org/news/the-curse-of-dimensionality-how-we-can-save-big-data-from-itself-d9fa0f872335/>

Deisenroth, M., Faisal A., y On C.S. (2020). Proyección lineal. [Gráfica].

Adaptada del libro "Mathematics for Machine Learning" <https://mml-book.com>.

Gleeson, P. (2017). Muestra de la dimensionalidad. [Gráfica].

"Escaping the Curse of Dimensionality." <https://www.freecodecamp.org/news/the-curse-of-dimensionality-how-we-can-save-big-data-from-itself-d9fa0f872335/>

Alley, G. (2018, 26 noviembre). What is a Data Pipeline? Alooma. <https://www.alooma.com/blog/what-is-a-data-pipeline>



Créditos

Facultad de
INGENIERÍA

Autores

Fabio Augusto González Osorio, PhD

Asistente docente

Miguel Ángel Ortiz Marín

Diseño instruccional

Claudia Patricia Rodríguez Sánchez

Diseño gráfico

Clara Valeria Suárez Caballero

Milton R. Pachón Pinzón

Diagramadora PPT

Daniela Duque

Fecha
2021-I

