TÉCNICAS DE REDUCCIÓN DE DIMENSIONALIDAD

* Análisis de componentes principales (PCA)
* Análisis de discriminante lineal (LDA)
* Análisis de discriminante cuadrático (QDA)

Qué es la DIMENSIONALIDAD?

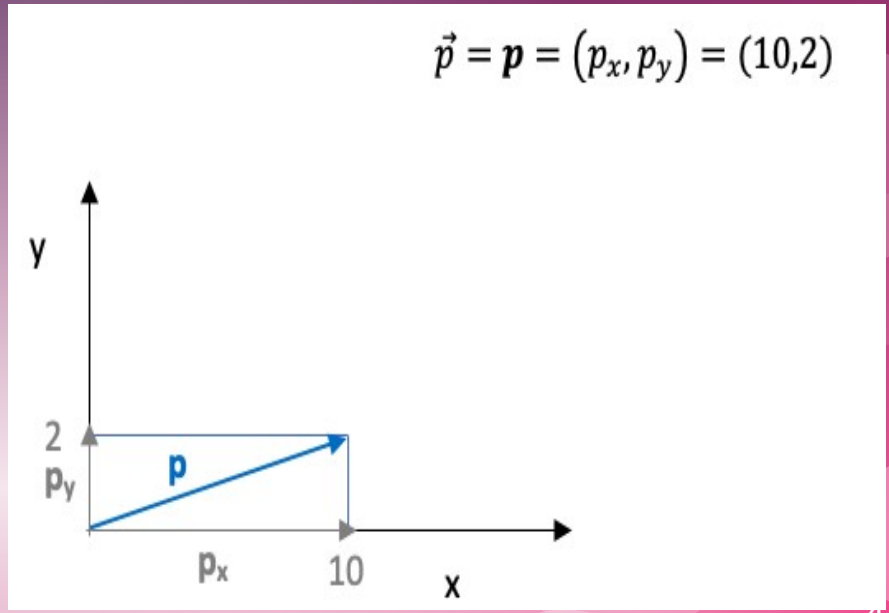
- El dataset consiste en una cierta cantidad de muestras N que son DESCRIPTAS por un conjunto de M características (o variables predictoras).
- La cantidad de variables que describen al dataset es la DIMENSIONALIDAD del dataset, a la que hemos llamado M.
- Si N es mucho mayor que M, la gran mayoría de los modelos de aprendizaje automático pueden resolver los problemas de clasificación sin problemas.
- Si N está cerca de M, empezamos a encontrar menos modelos que funcionan…
- Y si N es mucho menor que M, son todavía menos!
- Además existe 'la MALDICIÓN DE LA DIMENSIONALIDAD' (THE CURSE OF DIMENSIONALITY): A medida que crece la cantidad de dimensiones de un espacio de características, los datos disponibles se van volviendo dispersos. Esto genera un problema de significación estadística y obliga a tratar de encontrar más datos para poder describirlo. https://es.wikipedia.org/wiki/Maldici%C3%B3n_de_la_dimensi%C3%B3n

CÓMO ENCARAR LA SOLUCIÓN?

- Para evitar la ‘maldición’ (o una cantidad de características excesiva que haga el entrenamiento de los modelos de aprendizaje automático una ‘misión imposible’), se usan las técnicas de reducción de dimensionalidad.
- Vamos a ver una técnica no supervisada (PCA) y otra supervisada (LDA, que es un caso particular de QDA).
- Ambas técnicas usan un enfoque algo diferente...
- Y vamos a recordar algo de álgebra lineal

Análisis de Componentes Principales

- Desempolvemos el álgebra lineal: cada punto en un espacio vectorial se ‘identifica’ con una tupla de números. Ejemplo: en un espacio de 3 dimensiones espaciales como el que habitamos, nos alcanza para describir la posición de cualquier objeto con una tupla de 3 números (x, y, z)
- A ésto se llama ‘vector’ y no es más
- que un segmento orientado,
- porque tiene dirección y sentido
- (piensen en autos en una ruta…).
- Lo que está implícito es que cada
- vector se puede escribir como una
- suma de otros vectores...



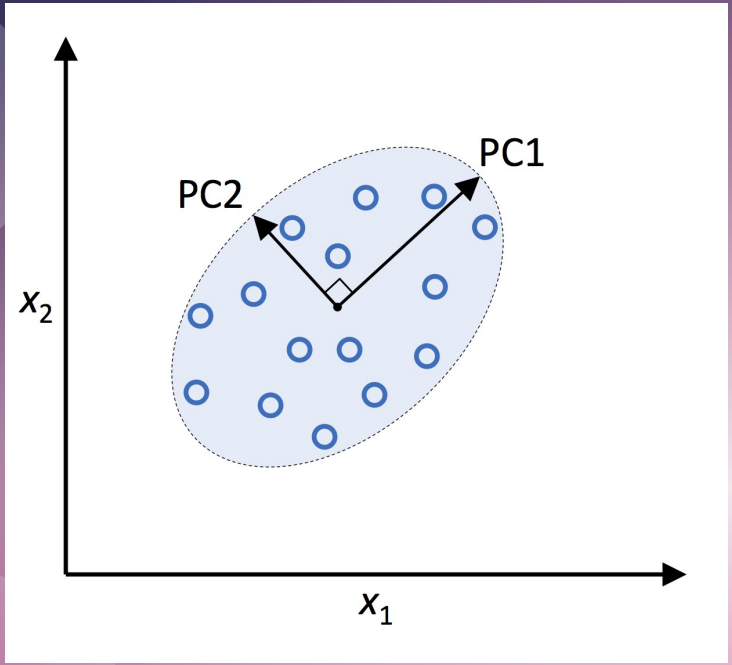
Y QUÉ TIENE QUE VER CON PCA?

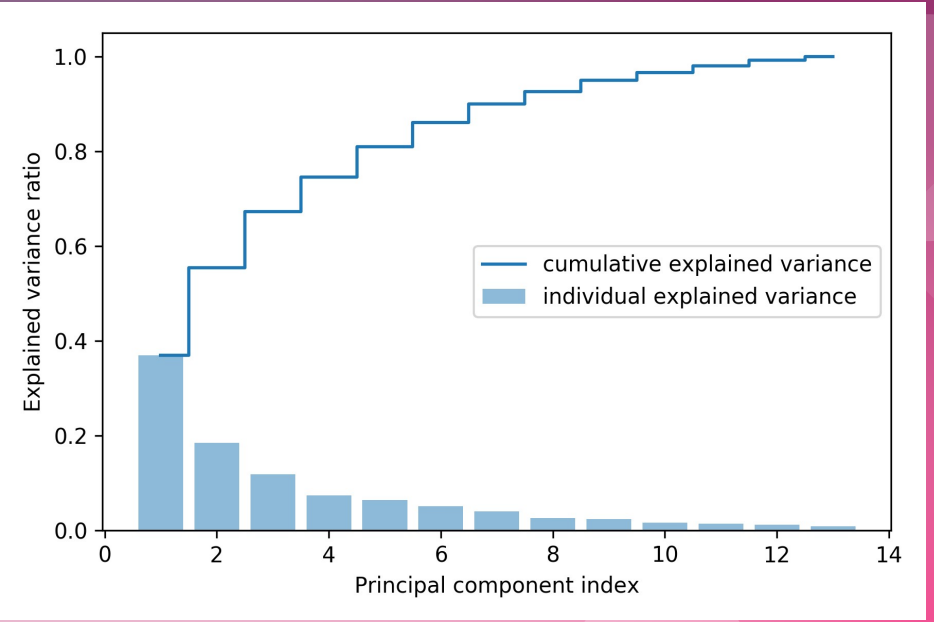
- El dataset es un conjunto de muestras que está descripta por los valores de sus características
- Cada muestra se puede representar como un vector en su espacio multidimensional de características
- Ese espacio se puede pensar como originado por sus propios autovectores y cada muestra es una suma de esos vectores generadores
- Entonces podemos pensar en recuperar esos vectores (los autovectores del espacio de características) y obtener los autovalores asociados a ellos.
- Los autovectores que más ‘importantes’ son, tienen asociado el autovalor mayor.

Cada autovector se puede interpretar como las direcciones de mayor varianza del espacio, entonces la matriz de varianza va a ‘representar’ a los autovectores

PCA permite seleccionar un subconjunto de esos autovectores y ‘achicar’ el espacio de características

Pagamos un precio en alguna pérdida de información, pero podemos elegir esa pérdida...



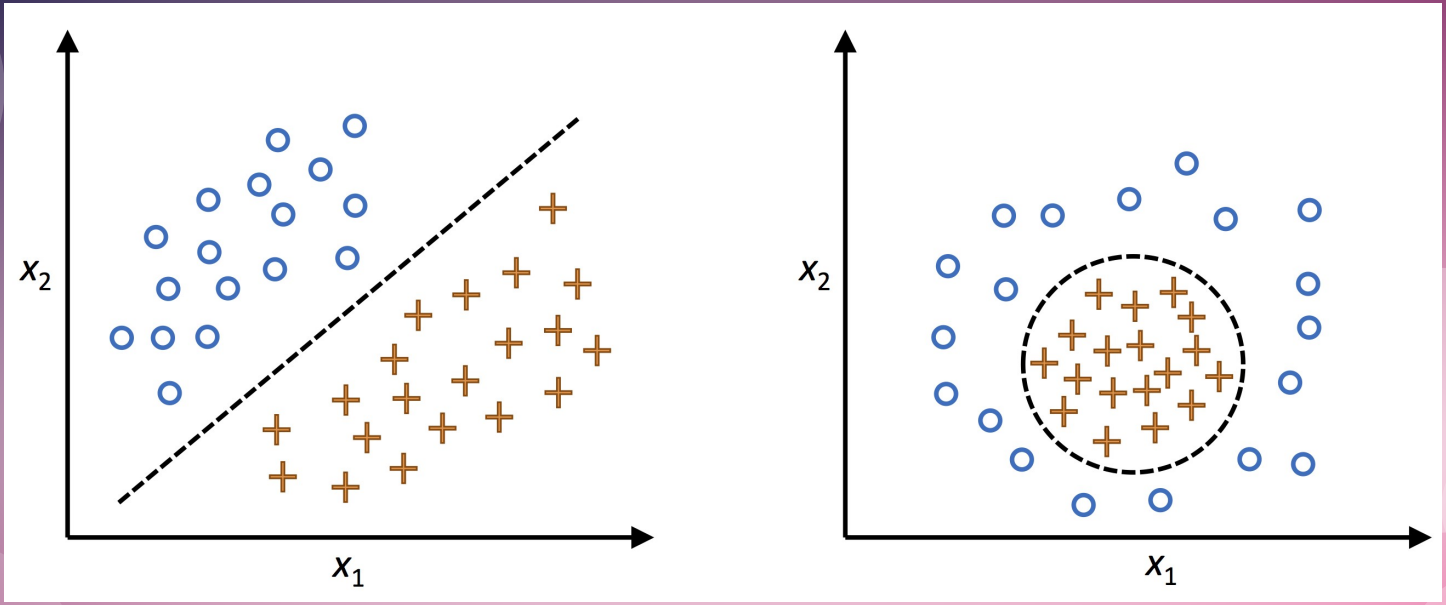


PCA EN SKLEARN

- Por supuesto no tenemos que hacer estas cuentas, para eso está Sklearn :)
- class sklearn.decomposition.PCA(n_components=None, *, copy=True, whiten=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversamples=10, power_iteration_normalizer='auto', random_state=None)

LDA

- Como dijimos, PCA y LDA tienen un enfoque diferente para reducir la dimensionalidad
- LDA sirve para problemas linealmente separables



LDA y QDA

- LDA es un clasificador que combina linealmente las características sin pérdida de información, reduciendo la dimensión del espacio de muestras
- Se puede aplicar si N es mayor que M (si es mucho mayor, mejor) porque necesita resolver ecuaciones que de otra manera, serían irresolubles
- Si el problema no se puede resolver con una frontera lineal, se puede intentar con QDA, que generaliza la idea de LDA para superficies de separación cuadráticas
- No siempre es mejor QDA que LDA pues necesita muchas muestras para funcionar bien, ya que calcula muchos parámetros más que LDA.
- Sirve si N es muy grande y M es mucho menor que N

- Class sklearn.discriminant_analysis.LinearDiscriminantAnalysis(solver='svd', shrinkage=None, priors=None, n_components=None, store_covariance=False, tol=0.0001, covariance_estimator=None)
- class sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis(*, priors=None, reg_param=0.0, store_covariance=False, tol=0.0001)

DUDAS O CONSULTAS?

MUCHAS GRACIAS POR SU PRESENCIA!!!

FUENTES:

* Python Machine Learning, Sebastian Raschka - Vahir Mirjalili, Marcombo Editorial- 2da Edición
* Documentación oficial de Scikit Learn
* Wikipedia
* y un poco de acá y de allá