Fundamentos de aprendizaje de máquina

Diplomado de especialización de
desarrollo de aplicaciones con Inteligencia Artificial

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

PUCP

ia.inf.pucp.edu.pe

Naive Bayes y Reducción de dimensionalidad

Diego Salas Guillén
salas.diego@pucp.pe

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

PUCP

ia.inf.pucp.edu.pe

Agenda

1. SVM: Conceptos generales
2. Árboles de decisión
3. Regresión Lineal y logística
4. SVM y kNN
5. **Naive Bayes y Reducción de dimensionalidad**
6. Métodos ensamblados
7. Aprendizaje no supervisado
8. Redes neuronales e introducción a Deep Learning

Naive Bayes

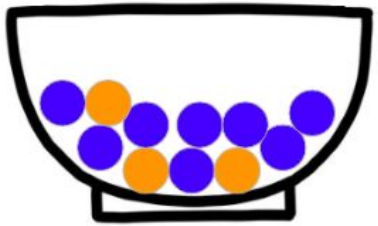
Clasificador Naive Bayes

El modelo de clasificación Naive Bayes utiliza un conjunto de algoritmos basados en aplicar el teorema de bayes con la asunción “ingenua” de que existe independencia condicional entre cada para de características, dada la variable de clase.

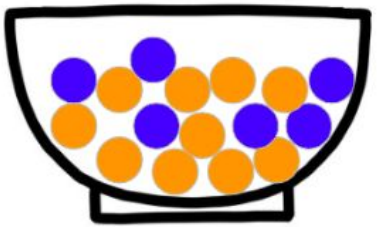
$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots x_n \mid y)}{P(x_1, \ldots, x_n)}$$

$$P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y),$$

Teorema de Bayes



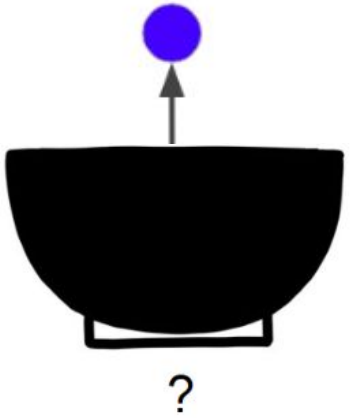
Bowl X



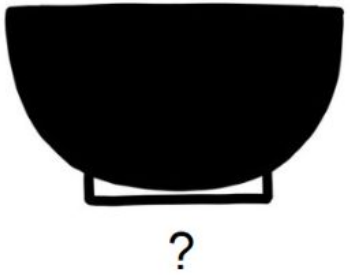
Bowl Y

Se tienen dos recipientes con esferas azules y naranjas. Se puede determinar la probabilidad de extraer una esfera azul o una esfera naranja fácilmente.

El problema es: Dado que se extrajo una esfera azul. ¿Cómo determinar de qué recipiente se extrajo? Para resolver este problema se puede utilizar el teorema de Bayes.



?

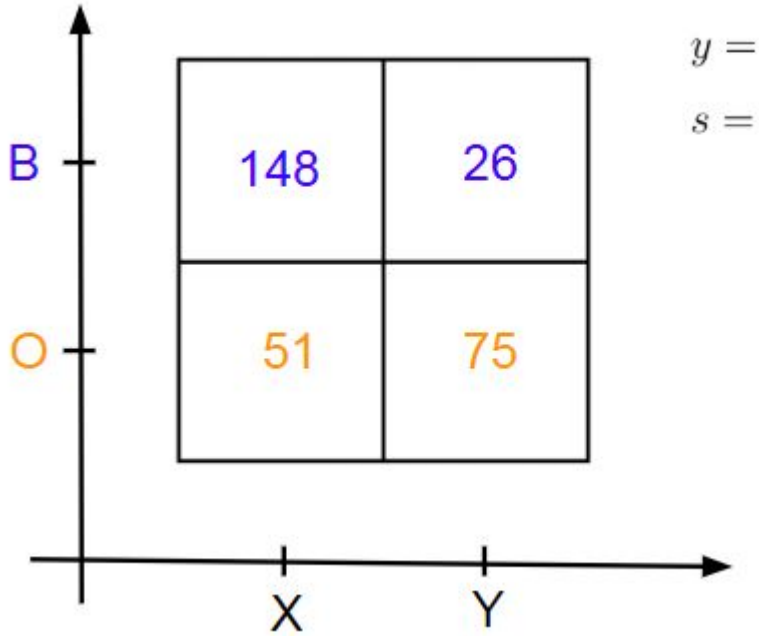


?

Teorema de Bayes

Considerando un conjunto de 300 esferas se plantea el siguiente experimento:

- Si se obtiene 4 o menos en un lanzamiento de dado se extrae una esfera del recipiente Y.
- En caso contrario se extrae del recipiente X.
- Se repite el experimento 300 veces.



$y = \{B, O\}$

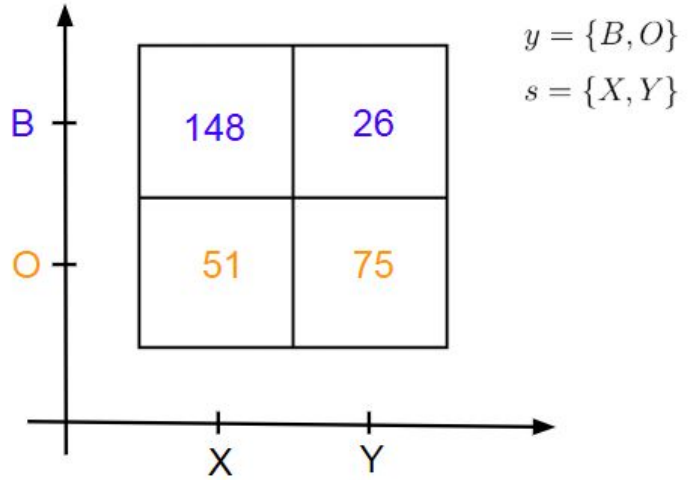
$s = \{X, Y\}$

Teorema de Bayes

¿Cuál es la probabilidad de escoger una esfera del recipiente X? ¿Del recipiente Y?

¿Cuál es la probabilidad de escoger una esfera azul (B) / naranja (O)?

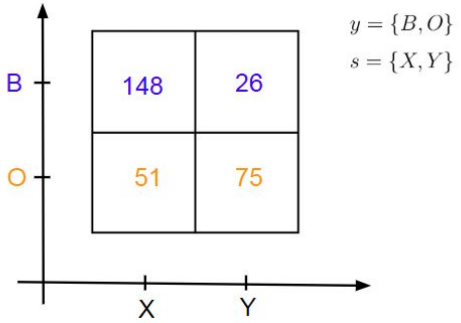
¿Cuál es la probabilidad de escoger una esfera azul del recipiente X?



Teorema de Bayes

¿Cuál es la probabilidad de escoger una esfera del recipiente X? ¿Del recipiente Y?

p(s=X) = n(s=X,y=B) + n(s=X,y=O) / N
p(s=X) = 0.66
p(s=Y) = 1 - 0.66 = 0.34



¿Cuál es la probabilidad de escoger una esfera azul (B) / naranja (O)?

p(y=B) = n(y=B,s=X) + n(y=B,s=Y) / N
p(y=B) = 0.58
p(y=O) = 1 - 0.58 = 0.42

Teorema de Bayes

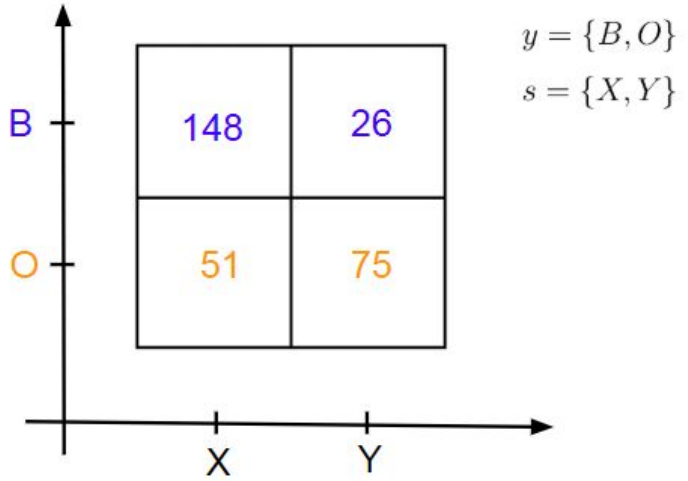
¿Cuál es la probabilidad de escoger una esfera azul del recipiente X?

p(s=X, y=B) = n(s=X,y=B) / N = 0.49

p(s=X,y=O) = 0.17

p(s=Y,y=B) = 0.09

p(s=X,y=O) = 0.25



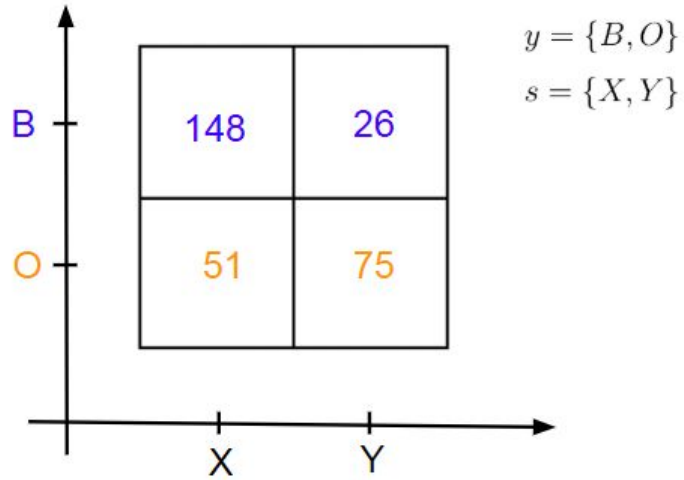
Teorema de Bayes

Dado que se extrajo una esfera del recipiente X. ¿Cuál es la probabilidad de que sea una esfera azul?

En este caso sabemos de qué recipiente se extrajo la esfera. Dado este hecho, se puede calcular la probabilidad de que la esfera sea azul p(y=B| s=X).

$$p(y = B|s = X) = \frac{n(s = X, y = B)}{n(s = X, y = B) + n(s = X, y = O)}$$

p(y=B|s=X) = 0.74



Teorema de Bayes: Regla del producto

Partiendo de la probabilidad calculada p(s=X,y=B) se calcula la siguiente igualdad aprovechando las relaciones previamente establecidas.

$$p(s=X,y=B) = \frac{n(s=X,y=B)}{N} \cdot \frac{n(s=X,y=B)+n(s=X,y=O)}{n(s=X,y=B)+n(s=X,y=O)}$$

$$= p(y=B|s=X) \cdot p(s=X)$$

Teorema de Bayes: Regla de la suma

La regla de la suma permite calcular la probabilidad p(X) sumando la probabilidades conjuntas que contienen la ocurrencia de X y las ocurrencias de las respectivas otras posibilidades.

$$p(s = X) = \frac{n(s = X, y = B) + n(s = X, y = O)}{N}$$

$$= \frac{n(s = X, y = B)}{N} + \frac{n(s = X, y = O)}{N}$$

$$= p(s = X, y = B) + p(s = X, y = O)$$

Teorema de Bayes

Para la regla del producto, el orden de las variables aleatorias no afecta el resultado. Por lo tanto: p(s,y) y p(y,s) tienen el mismo resultado.

$$p(s,y) = p(y|s)p(s)$$

$$p(y,s) = p(s|y)p(y)$$

$$\rightarrow p(s,y) = p(y,s)$$

Si planteamos la igualdad con los valores de p(s, y) y p(y, s) y los reorganizamos despejando p(s|y), obtenemos una expresión para calcular p(s|y).

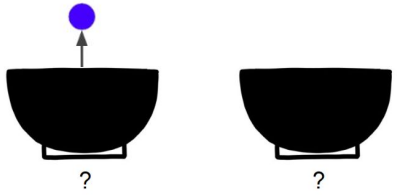
$$p(s,y) = p(y,s)$$

$$p(y|s)p(s) = p(s|y)p(y)$$

$$\rightarrow p(s|y) = \frac{p(y|s)p(s)}{p(y)}$$

Teorema de Bayes

Finalmente. ¿De qué recipiente se extrajo la esfera azul?



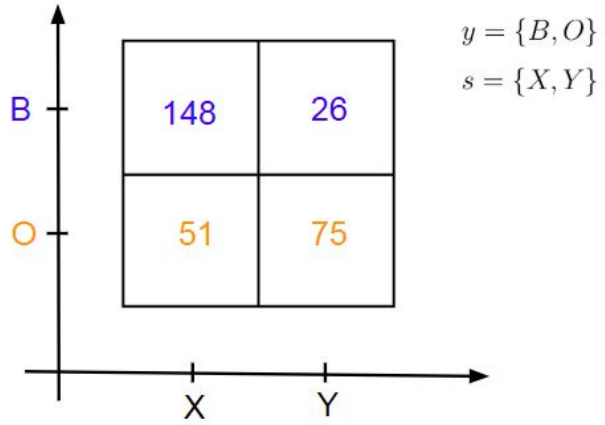
$$p(s = X|y = B) = \frac{p(y = B|s = X)p(s = X)}{p(y = B)}$$

$$= \frac{p(y = B|s = X)p(s = X)}{p(y = B, s = X) + p(y = B, s = Y)}$$

p(y=B|s=X) = 0.74
p(s=X) = 0.66
p(y=B) = 0.58

p(s=X|y=B) = 0.84

p(s=Y|y=B) = 0.16



Clasificador Bayesiano

Considere una colección de N objetos cada uno con un vector de características d-dimensional X. Sea Xk el vector de características del K-ésimo objeto. El vector de características se entiende como la d-tupla que describe al objeto. La tarea es clasificar los objetos dentro de una de C categorías.

$$\mathbf{P}(w_i|X_k) \geq \mathbf{P}(w_j|X_k), \forall j = 1, 2, .., C$$

Clasificador Bayesiano

Aplicando el teorema de Bayes, la regla de decisión se puede reescribir como:

$$\frac{\mathbf{P}(X_k|w_i) \times \mathbf{P}(w_i)}{\sum_{i=1}^{C} \mathbf{P}(X_k|w_i) \times \mathbf{P}(w_i)} \geq \frac{\mathbf{P}(X_k|w_j) \times \mathbf{P}(w_j)}{\sum_{j=1}^{C} \mathbf{P}(X_k|w_j) \times \mathbf{P}(w_j)}$$

El denominador es el mismo para ambos lados de la desigualdad:

$$\mathbf{P}(X_k|w_i) \times \mathbf{P}(w_i) \geq \mathbf{P}(X_k|w_j) \times \mathbf{P}(w_j)$$

Clasificador Bayesiano: Ejemplo - Tennis

Outlook	Temperature	Humidity	Windy	¿Play?
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rain	mild	high	FALSE	yes
rain	cool	normal	FALSE	yes
rain	cool	normal	TRUE	no
overcast	cool	normal	TRUE	yes
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
rain	mild	normal	FALSE	yes
sunny	mild	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rain	mild	high	TRUE	no

Problema

Clasificar el vector:

X = <rain,hot, high, false>

Clasificador Bayesiano: Ejemplo - Tennis

P(Yes) = 9/14
P(No) = 5/14

Outlook	
P(sunny|Yes) = 2/9	P(sunny|No) = 3/5
P(overcast|Yes) = 4/9	P(overcast|No) = 0
P(rain|Yes) = 3/9	P(rain|No) = 2/5

Temperature	
P(hot|Yes) = 2/9	P(hot|No) = 2/5
P(mild|Yes) = 4/9	P(mild|No) =2/5
P(cool|Yes) = 3/9	P(cool|No) = 1/5

Clasificador Bayesiano: Ejemplo - Tennis

Humidity	
P(high|Yes) = 3/9	P(high|No) = 4/5
P(normal|Yes) = 6/9	P(normal|No) = 1/5

Windy	
P(true|Yes) = 3/9	P(true|No) = 3/5
P(false|Yes) = 6/9	P(false|No) = 2/5

Clasificador Bayesiano: Ejemplo - Tennis

X = <rain, hot, high, false>

P(X|Yes) * P(Xes) = P(rain|Yes) * P(hot|Yes) * P(high|Yes) * P(false|Yes) * P(Yes)

P(X|No) * P(No) = P(rain|No) * P(hot|No) * P(high|No) * P(false|No) * P(No)

Clasificador Bayesiano: Ejemplo - Tennis

X = <rain, hot, high, false>

P(X|Yes) * P(Xes) = P(rain|Yes) * P(hot|Yes) * P(high|Yes) * P(false|Yes) * P(Yes)
= 3/9 * 2/9 * 3/9 * 6/9 * 9/14 = 0.010582

P(X|No) * P(No) = P(rain|No) * P(hot|No) * P(high|No) * P(false|No) * P(No)
= 2/5 * 2/5 * 4/5 * 2/5 * 5/14 = 0.018286

P(Yes|X) = (P(X|Yes) * P(Yes)) / P(X|Yes)*P(Yes) + P(X|No)*P(No)
P(Yes|X) = 0.37 = 37%
P(No|X) = 63%

Gaussian Naive Bayes

Se implementa en la clase GaussianNB. Asume que la distribución de probabilidad de las características es Gaussiana:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Bernoulli Naive Bayes

Se implementa en la clase BernoulliNB. Asume que la distribución de probabilidad de las características es Bernoulli multivariada. Pueden haber varias características pero cada una debe ser de valor binario (Bernoulli, boleana). La clase BernoulliNB puede binarizar la data recibida en base al parámetro “binarize”.

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

Multinomial Naive Bayes

Se implementa en la clase MultinomialNB. Asume que la distribución de probabilidad de las características es multinomial. Es una de las variantes clásicas de Naive Bayes para clasificación de texto.

La distribución está parametrizada por los vectores $\theta_y = (\theta_{y1}, ..., \theta_{yn})$ para cada clase y, n es el número de características. θ(yi) es la probabilidad P(Xi | y) de la característica y. Esta se estima usando una versión ajustada de la máxima verosimilitud:

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

α es un parámetro de ajuste. Nyi es el número de apariciones de la característica i en la clase y. Ny es el número de muestras de clase y.

Complement Naive Bayes

Se implementa en la clase ComplementNB. CNB es una adaptación del algoritmo Multinomial que se utiliza para datasets imbalanceados.

- Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of naive bayes text classifiers. In ICML (Vol. 3, pp. 616-623).

¿Preguntas?

Reducción de dimensionalidad

Problema

Muchos problemas de aprendizaje automático manejan miles de características.

Esto no sólo **ralentiza el entrenamiento de los modelos**, también puede complicar mucho la tarea de **encontrar una buena solución**. Además **dificulta la visualización** de la información.

Afortunadamente, en problemas reales, suele ser posible **reducir el número de características sin perder mucha información**.

Ej. imagenes MNIST.

Maldición de la dimensionalidad

El comportamiento de los objetos varía mucho en espacios altamente dimensionales.

Ej. en un espacio continuo de tamaño 1x1 la probabilidad de que un punto aleatorio esté a menos de 0.001 de un borde es de 0.4%. En un hipercubo 10000-dimensional unitario, esta probabilidad es de 99.999999%.

Ej. la distancia promedio entre dos puntos en un cuadrado unitario es de 0.52. En un cubo unitario 0.66. En el hipercubo 100000-dimensional es de 408.25.

Estos problemas podría solucionarse aumentando los datos de entrenamiento. Sin embargo, **conforme aumentan las dimensiones la cantidad de datos necesaria crece exponencialmente**.

Principales aproximaciones al problema: Proyección

Las instancias de entrenamiento **no están distribuidas uniformemente**. Algunas características mantienen **valores constantes** y otras están **altamente correlacionadas**. Como resultado, las instancias de entrenamiento **se ubican cerca a una superficie de menor dimensión**.

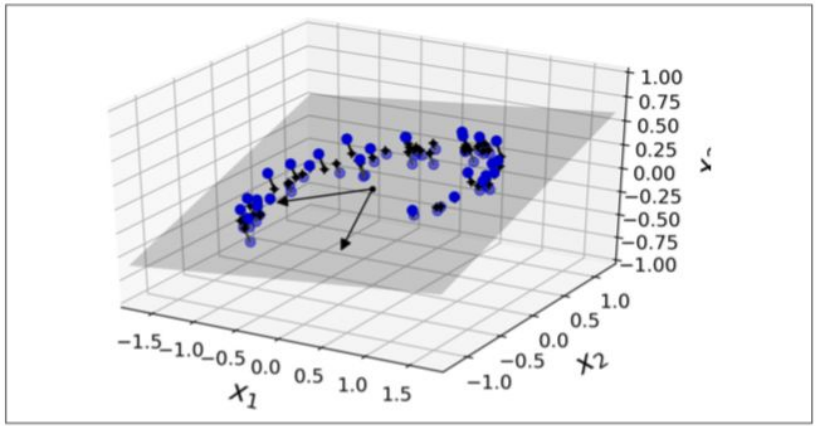


Figure 8-2. A 3D dataset lying close to a 2D subspace

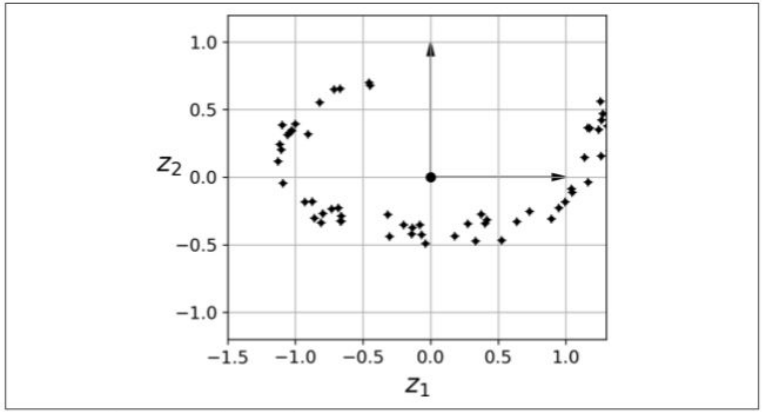


Figure 8-3. The new 2D dataset after projection

Principales aproximaciones al problema: Manifold Learning

La proyección no siempre es la mejor aproximación:

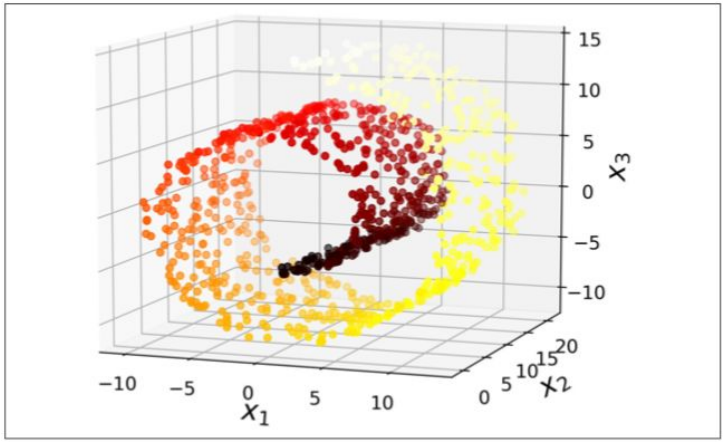


Figure 8-4. Swiss roll dataset

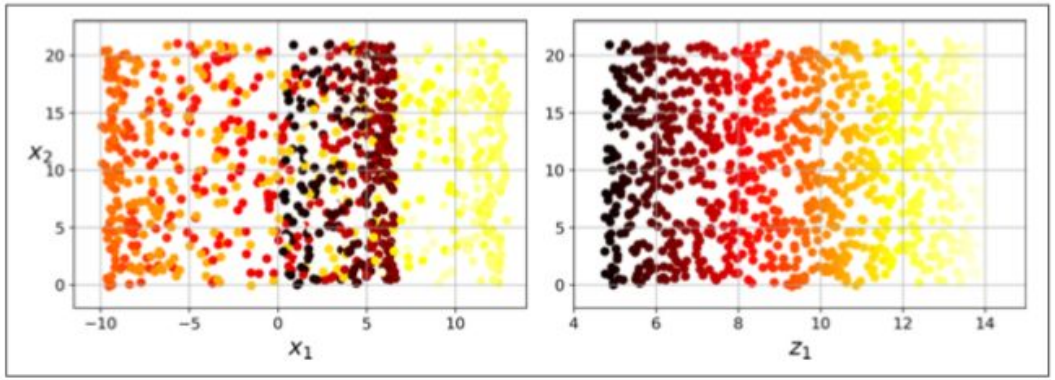


Figure 8-5. Squashing by projecting onto a plane (left) versus unrolling the Swiss roll (right)

Principales aproximaciones al problema: Manifold Learning

Un manifold es una superficie D-dimensional que puede ser acomodada en un espacio N-dimensional (D<N). En el ejemplo del enrrollado suizo, se utiliza un plano 2D enrollado respecto a la distribución de puntos 3D.

Muchos algoritmos de reducción de dimensionalidad tratan de **modelar la superficie en la que se encuentran las instancias de entrenamiento** (confiando en lo que se conoce como **manifold hypothesis**).

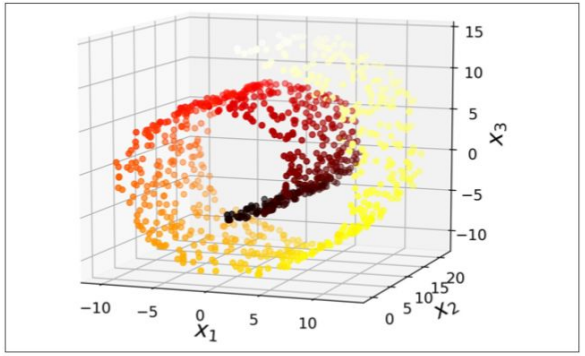


Figure 8-4. Swiss roll dataset

Principales aproximaciones al problema: Manifold Learning

A pesar de la utilidad de trabajar con un menor número de dimensiones, hay ocasiones en que el espacio original permite obtener una separación más adecuada.

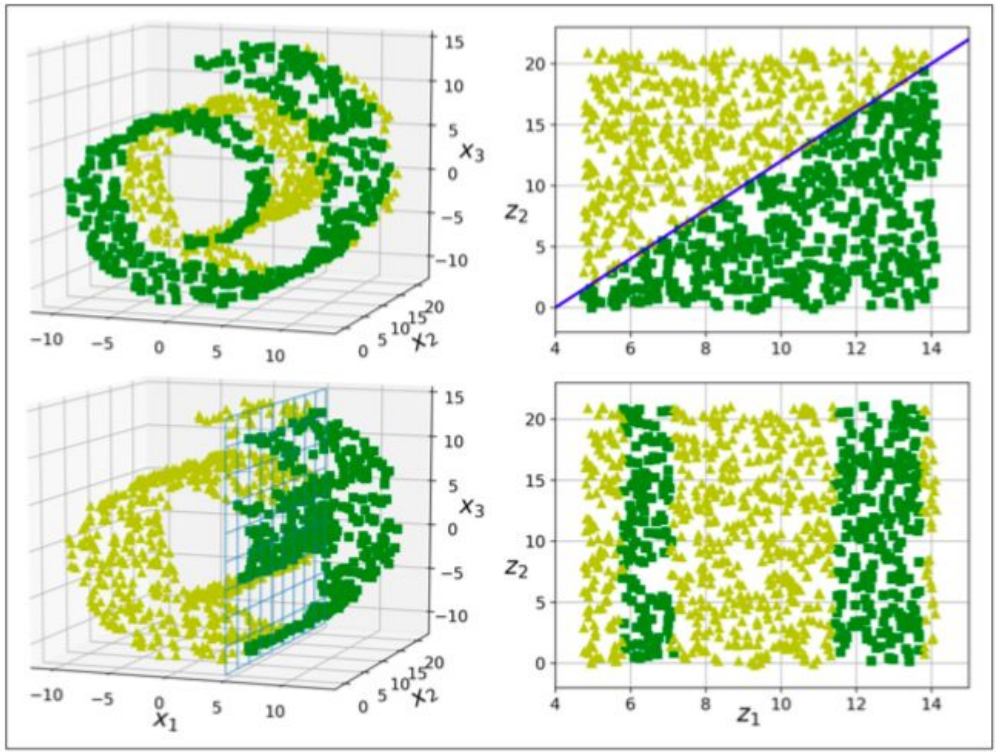


Figure 8-6. The decision boundary may not always be simpler with lower dimensions

Principal Component Analysis (PCA)

Algoritmo más popular, busca proyectar los datos hacia el hiperplano más cercano.

Se busca escoger el hiperplano que preserve mejor la varianza de los datos. En el ejemplo, escoger la línea sólida permitirá mantener mayor información sobre los datos.

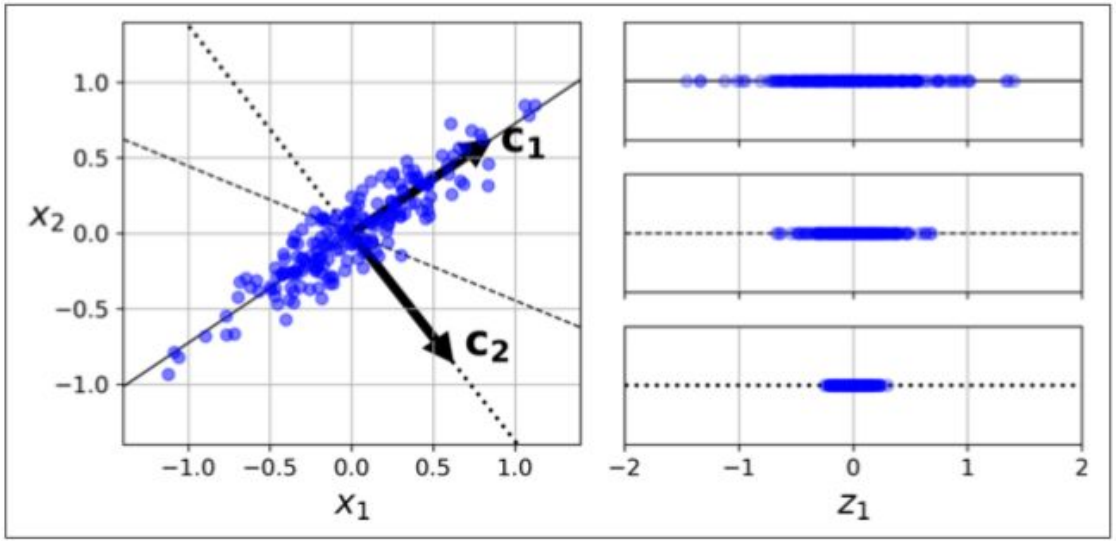


Figure 8-7. Selecting the subspace onto which to project

Componentes principales

PCA determina que la línea sólida carga la mayor varianza. Adicionalmente, encuentra un segundo eje ortogonal al primero (línea punteada) que explica la mayor parte del resto de la varianza. En un espacio de mayor dimensionalidad, habría un tercer eje (ortogonal a los dos anteriores) un cuarto, quinto y n-ésimo eje.

Se define como **componente principal** al **i-ésimo vector unitario que define al i-ésimo eje**. En la figura, C1 y C2 son los dos componentes principales.

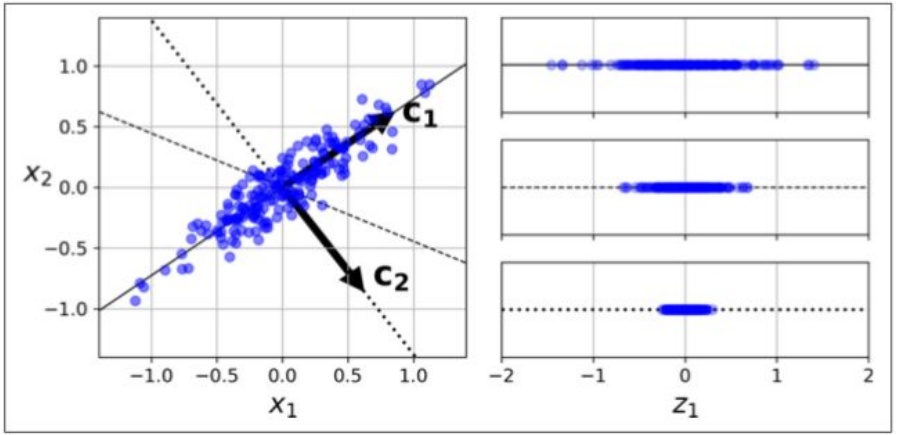


Figure 8-7. Selecting the subspace onto which to project

Single Value Decomposition (SVD)

Técnica estándar de factorización de matrices que permite descomponer una matriz de entrenamiento X en la multiplicación de tres matrices:

$$X = U\Sigma V^T$$

Cada columna de UΣ corresponde al vector de puntaje de los componentes principales de X.

Cada fila de Vt contiene los vectores de carga de los componentes principales de X.

```
X_centered = X - X.mean(axis=0)
U, s, Vt = np.linalg.svd(X_centered)
c1 = Vt.T[:, 0]
c2 = Vt.T[:, 1]
```

Single Value Decomposition (SVD)

Una vez que se ha obtenido los componentes principales se puede **reducir la dimensionalidad de un conjunto proyectándolo sobre los primeros D componentes principales**.

Sea Wd la matriz con lo d primeros componentes principales, se puede calcular la proyección de X mediante la siguiente multiplicación:

$$\mathbf{X}_{d\text{-proj}} = \mathbf{X}\mathbf{W}_d$$

Proporción de varianza explicada

Se puede extraer información importante sobre la varianza de los vectores UΣ. Esta información se encuentra en la propiedad explained_variance_ratio_ de la clase PCA de Scikit-Learn:

```
from sklearn.decomposition import PCA

pca = PCA(n_components = 2)
X2D = pca.fit_transform(X)
```

```
>>> pca.explained_variance_ratio_
array([0.84248607, 0.14631839])
```

La información obtenida se puede interpretar de la siguiente manera:

El componente principal 1 explica el 84.2% de la varianza y el componente principal 2 explica el 14.6% de la varianza. Se ha podido conservar el 98.8% de la varianza original.

Escoger el número correcto de dimensiones

El criterio principal para escoger el número de dimensiones es acumular la mayor parte de la varianza. Se puede establecer un valor deseado (por ejemplo, 95%).

Una opción más adecuada es graficar la varianza por componente principal (número de dimensiones) y determinar el punto en el crecimiento de la curva comienza a decaer.

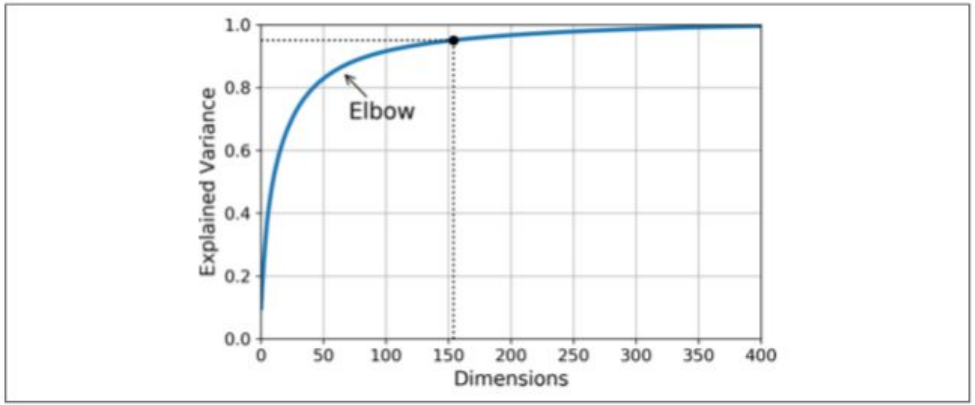


Figure 8-8. Explained variance as a function of the number of dimensions

Randomized PCA

Versión estocástica del algoritmo PCA. Permite e**ncontrar una aproximación de los componentes principales con menor complejidad computacional** cuando la cantidad de dimensiones es menor que la cantidad de muestras.

Se puede utilizar el parámtetro *svd_solver* con el valor *“randomized”*, su valor por defecto es *“auto”*.

Por defecto, se usa RPCA cuando las dimensiones son menores al 80% de la muestra y esta es mayor a 500. Si se desea forzar el algoritmo PCA clásico se puede usar el parámetro *“full”*.

Incremental PCA

Se implementa en la clase IncrementalPCA. Ataca el problema de la implementación original que requiere cargar todo el conjunto de entrenamiento en memoria.

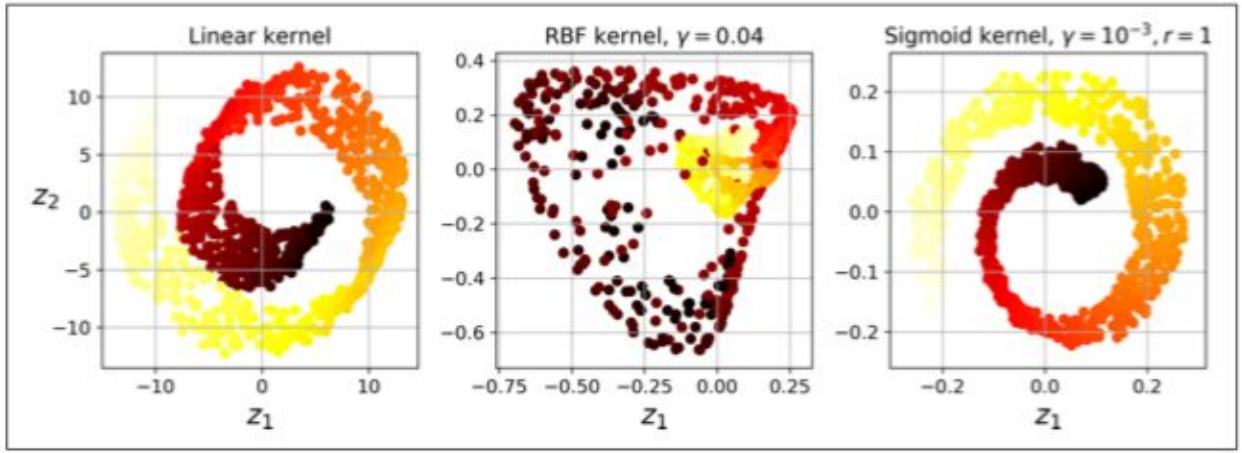
Permite separar el conjunto de entrenamiento en partes y alimentar al algoritmo IPCA una parte a la vez.

```
from sklearn.decomposition import IncrementalPCA

n_batches = 100
inc_pca = IncrementalPCA(n_components=154)
for X_batch in np.array_split(X_train, n_batches):
    inc_pca.partial_fit(X_batch)

X_reduced = inc_pca.transform(X_train)
```

Kernel PCA

Al estudiar las máquinas de soporte vectorial revisamos una técnica matemática para mapear instancias a un espacio dimensional mayor.

Una frontera de decisión en este espacio dimensional correspondía a una frontera compleja no lineal en el espacio original. Esta técnica también puede ser utilizada con PCA.



Locally Linear Embedding (LLE)

Es una técnica no-lineal de reducción de dimensionalidad. No se basa en proyecciones. Calcula la distancia lineal entre instancias vecinas y construye la superficie (manifold) de menor dimensión que mejor preserve esta distancia.

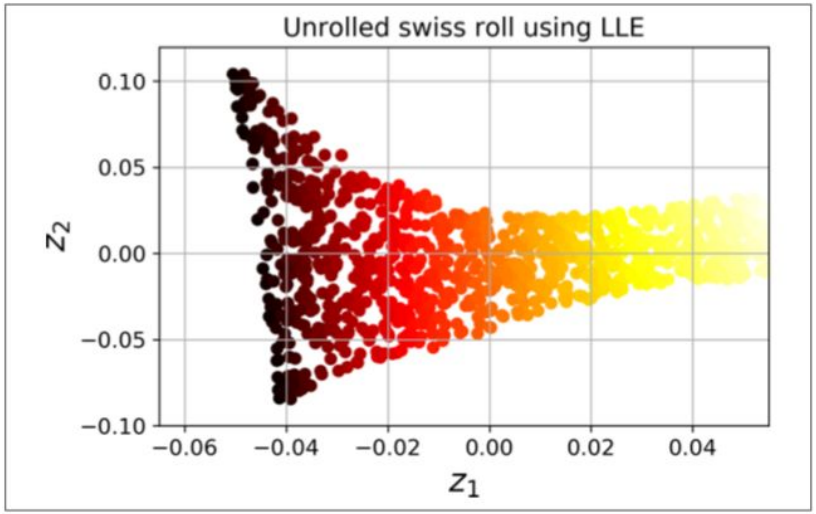


Figure 8-12. Unrolled Swiss roll using LLE

Locally Linear Embedding (LLE)

Para cada instancia de entrenamiento x(i), el algoritmo identifica los k vecinos más cercanos.

Luego trata de reconstruir x(i) como una función lineal de sus vecinos aplicando los pesos w(i,j) minimizando la distancia cuadrada entre x(i) y la sumatoria de w(i,j)*x(j) para j = 1,2, … , m.

Para los elementos que no pertenecen a los k vecinos más cercanos w(i,j) = 0.

Locally Linear Embedding (LLE)

El resultado es un problema de optimización con restricciones:

Equation 8-4. LLE step 1: linearly modeling local relationships

$$\widehat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmin}} \sum_{i=1}^{m} \left(\mathbf{x}^{(i)} - \sum_{j=1}^{m} w_{i,j} \mathbf{x}^{(j)} \right)^2$$

$$\text{subject to} \begin{cases} w_{i,j} = 0 & \text{if } \mathbf{x}^{(j)} \text{ is not one of the } k \text{ c.n. of } \mathbf{x}^{(i)} \\ \sum_{j=1}^{m} w_{i,j} = 1 & \text{for } i = 1, 2, \cdots, m \end{cases}$$

La siguiente etapa es mapear las instancias a un espacio d-dimensional preservando las relaciones locales calculadas en W.

Locally Linear Embedding (LLE)

Sea z(i) la imagen de x(i) en este espacio d-dimensional, entonces se requiere que la distancia entre z(i) y la suma de w(i,j)*z(j) para j = 1,2, … , m sea la menor posible manteniendo w(i,j) constante.

Equation 8-5. LLE step 2: reducing dimensionality while preserving relationships

$$\hat{\mathbf{Z}} = \underset{\mathbf{Z}}{\operatorname{argmin}} \sum_{i=1}^{m} \left(\mathbf{z}^{(i)} - \sum_{j=1}^{m} \hat{w}_{i,j} \mathbf{z}^{(j)} \right)^2$$

Este algoritmo resulta costoso para conjuntos de datos muy grandes.

Otras técnicas

Multidimensional Scaling (MDS): Reduce dimensionalidad preservando distancias entre las instancias.

Isomap: Conecta las instancias con sus vecinos más cercanos en un grafo. Preserva la distancia geodésica, camino más corto entre nodos del grafo.

t-Distributed Stochastic Neighbor Embedding (t-SNE): Reduce la dimensionalidad tratando de mantener distancias similares cercanas y distancias diferentes alejadas. Sirve para visualizar clusters.

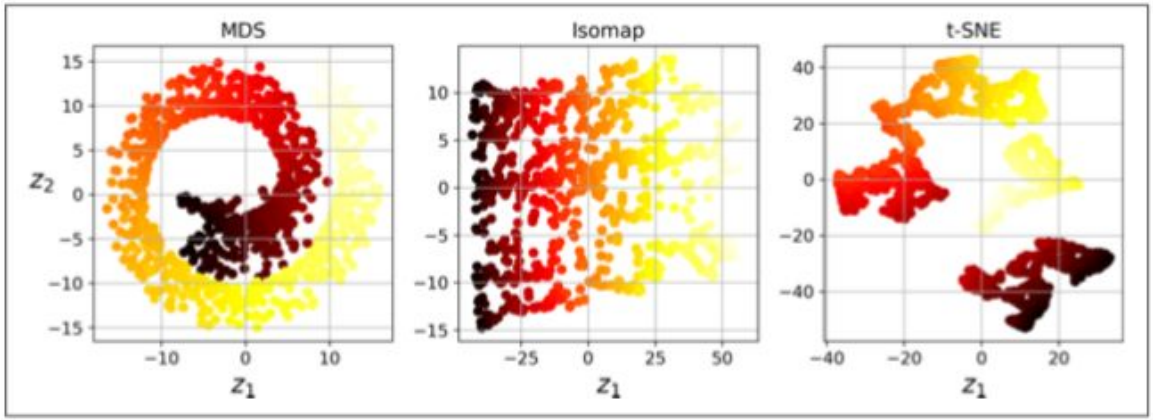


Figure 8-13. Reducing the Swiss roll to 2D using various techniques

¿Preguntas?

Let’s code

Referencias

- Hands on Machine Learning with Sckit-Learn, Teras & Tensorflow - Aurélien Géron (O'Really)
- Naive Bayes - Documentación Scikit-Learn - https://scikit-learn.org/stable/modules/naive_bayes.html
- Bayes theorem - The holy grail of data science - https://towardsdatascience.com/bayes-theorem-the-holy-grail-of-data-science-55d93315defb
- Naive Bayes Classifier - https://www.youtube.com/watch?v=CPqOCI0ahss
- Singular Value Decomposition - https://www.youtube.com/watch?v=Ls2TgGFfZnU
- StatQuest: Principal Component Analysis (PCA), Step-by-Step - https://www.youtube.com/watch?v=FgakZw6K1QQ
- Shlens, J. (2014). A tutorial on principal component analysis. arXiv preprint arXiv:1404.1100.

GRACIAS!