Aprendizaje de Máquina, Clasificación, Selección

Juan F. Pérez

Departamento MACC
Matemáticas Aplicadas y Ciencias de la Computación
Universidad del Rosario

juanferna.perez@urosario.edu.co

Mayo de 2018

Contenidos

Introducción

Introducción

- Búsqueda de patrones
- Estudio de fenómenos físicos
- Reconocimiento de patrones
- Descubrimiento automático de regularidades
- Algoritmos computacionales

Ejemplo

Reconocimiento de dígitos

Ejemplo Reconocimiento de Dígitos

- Cada dígito es una imagen de 28x28 pixeles.
- Vector x de 784 números reales que representan la intensidad en cada pixel (0 blanco, 1 negro)
- Objetivo: construir un mecanismo que permita determinar automáticamente qué dígito corresponde a una nueva imagen dada en el mismo formato
- `http://yann.lecun.com/exdb/mnist/`

Ejemplo Reconocimiento de Dígitos

- Problema de aprendizaje de máquina

Ejemplo Reconocimiento de Dígitos

- Problema de aprendizaje de máquina
- Conjunto de datos de entrenamiento

Ejemplo Reconocimiento de Dígitos

- Problema de aprendizaje de máquina
- Conjunto de datos de entrenamiento
- N imágenes en el formato definido $\{x_1, \ldots, x_N\}$

Ejemplo Reconocimiento de Dígitos

- Problema de aprendizaje de máquina
- Conjunto de datos de entrenamiento
- N imágenes en el formato definido $\{x_1, \ldots, x_N\}$
- x_i: vector de características de la imagen i

Ejemplo Reconocimiento de Dígitos

- Problema de aprendizaje de máquina
- Conjunto de datos de entrenamiento
- N imágenes en el formato definido $\{x_1, \ldots, x_N\}$
- x_i: vector de características de la imagen i
- Para cada imagen conocemos la categoría (dígito que representa)

Ejemplo Reconocimiento de Dígitos

- Problema de aprendizaje de máquina
- Conjunto de datos de entrenamiento
- N imágenes en el formato definido $\{x_1, \ldots, x_N\}$
- x_i: vector de características de la imagen i
- Para cada imagen conocemos la categoría (dígito que representa)
- Categoría de la imagen i: vector t_i

Ejemplo Reconocimiento de Dígitos

- Resultado del algoritmo de aprendizaje de máquina: función $y(x)$

Ejemplo Reconocimiento de Dígitos

- Resultado del algoritmo de aprendizaje de máquina: función $y(x)$
- Para una imagen x dada, $y(x)$ es el dígito asociado

Ejemplo Reconocimiento de Dígitos

- Resultado del algoritmo de aprendizaje de máquina: función $y(x)$
- Para una imagen x dada, $y(x)$ es el dígito asociado
- Fase de Entrenamiento: determinar $y(x)$ a partir de $\{x_1, \ldots, x_N\}$

Ejemplo Reconocimiento de Dígitos

- Resultado del algoritmo de aprendizaje de máquina: función $y(x)$
- Para una imagen x dada, $y(x)$ es el dígito asociado
- Fase de Entrenamiento: determinar $y(x)$ a partir de $\{x_1, \ldots, x_N\}$
- Fase de prueba: para unas imágenes x diferentes a las de entrenamiento, pero con categorías conocidas, probar la precisión de la función obtenida

Ejemplo Reconocimiento de Dígitos

- Resultado del algoritmo de aprendizaje de máquina: función $y(x)$
- Para una imagen x dada, $y(x)$ es el dígito asociado
- Fase de Entrenamiento: determinar $y(x)$ a partir de $\{x_1, \ldots, x_N\}$
- Fase de prueba: para unas imágenes x diferentes a las de entrenamiento, pero con categorías conocidas, probar la precisión de la función obtenida
- Capacidad de generalizar del modelo: responder correctamente a imágenes diferentes a las de entrenamiento

Ejemplo Reconocimiento de Dígitos

Preprocesamiento:

- Transformación inicial de los datos antes de la fase de entrenamiento

Ejemplo Reconocimiento de Dígitos

Preprocesamiento:

- Transformación inicial de los datos antes de la fase de entrenamiento
- Dígitos: imágenes re-escaladas en cajas de tamaño fijo (28x28)

Ejemplo Reconocimiento de Dígitos

Preprocesamiento:

- Transformación inicial de los datos antes de la fase de entrenamiento
- Dígitos: imágenes re-escaladas en cajas de tamaño fijo (28x28)
- Reducción de variabilidad en los datos

Ejemplo Reconocimiento de Dígitos

Preprocesamiento:

- Transformación inicial de los datos antes de la fase de entrenamiento
- Dígitos: imágenes re-escaladas en cajas de tamaño fijo (28x28)
- Reducción de variabilidad en los datos
- Facilita la identificación de categorías

Ejemplo Reconocimiento de Dígitos

Preprocesamiento:

- Transformación inicial de los datos antes de la fase de entrenamiento
- Dígitos: imágenes re-escaladas en cajas de tamaño fijo (28x28)
- Reducción de variabilidad en los datos
- Facilita la identificación de categorías
- Simplifica las características a usar

Ejemplo Reconocimiento de Dígitos

Preprocesamiento:

- Transformación inicial de los datos antes de la fase de entrenamiento
- Dígitos: imágenes re-escaladas en cajas de tamaño fijo (28x28)
- Reducción de variabilidad en los datos
- Facilita la identificación de categorías
- Simplifica las características a usar
- Mejora eficiencia de los algoritmos de aprendizaje

Ejemplo Reconocimiento de Dígitos

Preprocesamiento:

- Transformación inicial de los datos antes de la fase de entrenamiento
- Dígitos: imágenes re-escaladas en cajas de tamaño fijo (28x28)
- Reducción de variabilidad en los datos
- Facilita la identificación de categorías
- Simplifica las características a usar
- Mejora eficiencia de los algoritmos de aprendizaje
- Extracción de características

Ejemplo Reconocimiento de Dígitos

- Datos de entrenamiento contienen tanto las características x_i como las categorías/etiquetas t_i

Ejemplo Reconocimiento de Dígitos

- Datos de entrenamiento contienen tanto las características x_i como las categorías/etiquetas t_i
  - *Aprendizaje supervisado*

Ejemplo Reconocimiento de Dígitos

- Datos de entrenamiento contienen tanto las características x_i como las categorías/etiquetas t_i
  - *Aprendizaje supervisado*
-
- Número de categorías finito

Ejemplo Reconocimiento de Dígitos

- Datos de entrenamiento contienen tanto las características x_i como las categorías/etiquetas t_i
  - *Aprendizaje supervisado*
-
- Número de categorías finito
  - *Clasificación*

Ejemplo Reconocimiento de Dígitos

- Datos de entrenamiento contienen tanto las características x_i como las categorías/etiquetas t_i
  - *Aprendizaje supervisado*
-
- Número de categorías finito
  - *Clasificación*
- Clasificar datos de entrada en una de un número finito de categorías

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i
  - *Aprendizaje supervisado*

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i
    - *Aprendizaje supervisado*
-
- Resultado es una o varias variables continuas (no un número finito de categorías)

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i
  - *Aprendizaje supervisado*
-
- Resultado es una o varias variables continuas (no un número finito de categorías)
  - *Regresión*

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i
  - *Aprendizaje no supervisado*

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i
  - *Aprendizaje no supervisado*
-
- Objetivo es descubrir grupos similares

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i
  - *Aprendizaje no supervisado*
-
- Objetivo es descubrir grupos similares
  - *Clustering (análisis de conglomerados)*

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i
  - *Aprendizaje no supervisado*

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i
    - *Aprendizaje no supervisado*
-
- Objetivo es determinar la distribución de los datos en el espacio de entrada

Otros problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i
  - *Aprendizaje no supervisado*
-
- Objetivo es determinar la distribución de los datos en el espacio de entrada
  - *Estimación de densidades*

Modelos de Clasificación

Modelos de Clasificación

Objetivo:

- Dado un vector de entrada x de dimensión D (características)

Modelos de Clasificación

Objetivo:

- Dado un vector de entrada x de dimensión D (características)
- Asignarlo a una de K clases $(C_k,\ k = 1, \dots, K)$

Modelos de Clasificación

Objetivo:

- Dado un vector de entrada x de dimensión D (características)
- Asignarlo a una de K clases $(C_k,\ k = 1, \ldots, K)$
- Clases disyuntas: cada observación asignada a una sola de las clases (más usual)

Modelos de Clasificación

Objetivo:

- Dado un vector de entrada x de dimensión D (características)
- Asignarlo a una de K clases ($C_k, \; k = 1, \ldots, K$)
- Clases disyuntas: cada observación asignada a una sola de las clases (más usual)
- Espacio de entrada (donde representamos los datos de entrada) se divide en regiones de decisión

Modelos de Clasificación

Objetivo:

- Dado un vector de entrada x de dimensión D (características)
- Asignarlo a una de K clases $(C_k,\ k = 1, \ldots, K)$
- Clases disyuntas: cada observación asignada a una sola de las clases (más usual)
- Espacio de entrada (donde representamos los datos de entrada) se divide en regiones de decisión
- Fronteras o superficies entre regiones

Modelos de Clasificación

Objetivo:

- Dado un vector de entrada x de dimensión D (características)
- Asignarlo a una de K clases ($C_k,\ k = 1, \ldots, K$)
- Clases disyuntas: cada observación asignada a una sola de las clases (más usual)
- Espacio de entrada (donde representamos los datos de entrada) se divide en regiones de decisión
- Fronteras o superficies entre regiones
- Modelos lineales: superficies son funciones lineales del vector x (hiperplanos en espacio de dimensión D)

Modelos de Clasificación

Objetivo:

- Dado un vector de entrada x de dimensión D (características)
- Asignarlo a una de K clases $(C_k,\ k = 1, \ldots, K)$
- Clases disyuntas: cada observación asignada a una sola de las clases (más usual)
- Espacio de entrada (donde representamos los datos de entrada) se divide en regiones de decisión
- Fronteras o superficies entre regiones
- Modelos lineales: superficies son funciones lineales del vector x (hiperplanos en espacio de dimensión D)
- Datos linealmente separables: clases se pueden separar exactamente por funciones lineales

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?
- Dos clases: $t \in \{0, 1\}$

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?
- Dos clases: $t \in \{0, 1\}$
- $t = 1$: x en C_1
- $t = 0$: x en C_2

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?
- Dos clases: $t \in \{0, 1\}$
- $t = 1$: x en C_1
- $t = 0$: x en C_2
- Interpretar t como probabilidad de que x pertenezca a la clase C_1

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?
- $K > 2$ clases: t vector de longitud K igual a cero, excepto en la posición j donde es igual a 1 si x en C_j

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?
- $K > 2$ clases: t vector de longitud K igual a cero, excepto en la posición j donde es igual a 1 si x en C_j
- Ejemplo $K = 3$

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?
- $K > 2$ clases: t vector de longitud K igual a cero, excepto en la posición j donde es igual a 1 si x en C_j
- Ejemplo $K = 3$
- $t = (1, 0, 0)$ si x en C_1
- $t = (0, 1, 0)$ si x en C_2
- $t = (0, 0, 1)$ si x en C_3

Representación de categorías

- ¿Cómo representar categorías/etiquetas/objetivos t?
- $K > 2$ clases: t vector de longitud K igual a cero, excepto en la posición j donde es igual a 1 si x en C_j
- Ejemplo $K = 3$
- $t = (1, 0, 0)$ si x en C_1
- $t = (0, 1, 0)$ si x en C_2
- $t = (0, 0, 1)$ si x en C_3
- Interpretar t_k como probabilidad de que x pertenezca a la clase C_k

Clasificador Bayesiano Ingenuo

Probabilidades condicionales

- A, B eventos

Probabilidades condicionales

- A, B eventos
- $A|B$: el evento A ocurre dado que el evento B ocurrió

Probabilidades condicionales

- A, B eventos
- $A|B$: el evento A ocurre dado que el evento B ocurrió
- $B|A$: el evento B ocurre dado que el evento A ocurrió

Probabilidades condicionales

- A, B eventos
- $A|B$: el evento A ocurre dado que el evento B ocurrió
- $B|A$: el evento B ocurre dado que el evento A ocurrió
- $P(A)$: probabilidad de que A ocurra

Probabilidades condicionales

- A, B eventos
- $A|B$: el evento A ocurre dado que el evento B ocurrió
- $B|A$: el evento B ocurre dado que el evento A ocurrió
- $P(A)$: probabilidad de que A ocurra
- $P(A|B)$: probabilidad de que A ocurra dado que B ocurrió

Probabilidades condicionales

- A, B eventos
- $A|B$: el evento A ocurre dado que el evento B ocurrió
- $B|A$: el evento B ocurre dado que el evento A ocurrió
- $P(A)$: probabilidad de que A ocurra
- $P(A|B)$: probabilidad de que A ocurra dado que B ocurrió
- $P(B|A)$: probabilidad de que B ocurra dado que A ocurrió

Probabilidades condicionales

- $P(A)$, $P(B)$: probabilidades a priori (prior)

Probabilidades condicionales

- $P(A)$, $P(B)$: probabilidades a priori (prior)
- $P(A|B)$, $P(B|A)$: probabilidades a posteriori (posterior)

Probabilidades condicionales

Ejemplo: lanzamiento de un dado

Probabilidades condicionales

Ejemplo: lanzamiento de un dado

- A el resultado es par

Probabilidades condicionales

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1

Probabilidades condicionales

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1
- $$P(A) = \frac{3}{6} = \frac{1}{2}$$

Probabilidades condicionales

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1
- $$P(A) = \frac{3}{6} = \frac{1}{2}$$
- $$P(B) = \frac{5}{6}$$

Probabilidades condicionales

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1
- $$P(A) = \frac{3}{6} = \frac{1}{2}$$
- $$P(B) = \frac{5}{6}$$
- $$P(A|B) = \frac{3}{5}$$

Probabilidades condicionales

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1
- $$P(A) = \frac{3}{6} = \frac{1}{2}$$
- $$P(B) = \frac{5}{6}$$
- $$P(A|B) = \frac{3}{5}$$
- $$P(B|A) = \frac{3}{3} = 1$$

Teorema de Bayes

Si $P(B) > 0$

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}$$

Teorema de Bayes

Ejemplo: lanzamiento de un dado

Teorema de Bayes

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1

Teorema de Bayes

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1
- $$P(A) = \frac{3}{6} = \frac{1}{2}$$
- $$P(B) = \frac{5}{6}$$

Teorema de Bayes

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1
- $$P(A) = \frac{3}{6} = \frac{1}{2}$$
- $$P(B) = \frac{5}{6}$$
- $$P(B|A) = \frac{3}{3} = 1$$

Teorema de Bayes

Ejemplo: lanzamiento de un dado

- A el resultado es par
- B el resultado es mayor que 1
- $$P(A) = \frac{3}{6} = \frac{1}{2}$$
- $$P(B) = \frac{5}{6}$$
- $$P(B|A) = \frac{3}{3} = 1$$
- $$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{(1)(1/2)}{(5/6)} = \frac{3}{5}$$

Clasificador Bayesiano Ingenuo

- x: características de una observación

Clasificador Bayesiano Ingenuo

- x: características de una observación
- $C_1, \ldots, C_K$: categorías

Clasificador Bayesiano Ingenuo

- x: características de una observación
- $C_1, \ldots, C_K$: categorías
- $P(x)$: prob. de que una observación tenga las características x

Clasificador Bayesiano Ingenuo

- x: características de una observación
- $C_1, \ldots, C_K$: categorías
- $P(x)$: prob. de que una observación tenga las características x
- $P(C_k)$: prob. de que una observación sea de la categoría C_k

Clasificador Bayesiano Ingenuo

- x: características de una observación
- $C_1, \ldots, C_K$: categorías
- $P(x)$: prob. de que una observación tenga las características x
- $P(C_k)$: prob. de que una observación sea de la categoría C_k
- Clasificación de x:

$$P(C_k|x)$$

Clasificador Bayesiano Ingenuo

- x: características de una observación
- $C_1, \ldots, C_K$: categorías
- $P(x)$: prob. de que una observación tenga las características x
- $P(C_k)$: prob. de que una observación sea de la categoría C_k
- Clasificación de x:

$$P(C_k|x)$$

- Bayes:

$$P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}$$

Clasificador Bayesiano Ingenuo

Seleccionando entre dos clases C_k y C_j

- $$P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}$$

Clasificador Bayesiano Ingenuo

Seleccionando entre dos clases C_k y C_j

- $$P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}$$
- $$P(C_j|x) = \frac{P(x|C_j)P(C_j)}{P(x)}$$

Clasificador Bayesiano Ingenuo

Seleccionando entre dos clases C_k y C_j

- $$P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}$$
- $$P(C_j|x) = \frac{P(x|C_j)P(C_j)}{P(x)}$$
- Cociente: $$\frac{P(C_k|x)}{P(C_j|x)} = \frac{P(x|C_k)P(C_k)}{P(x|C_j)P(C_j)}$$

Clasificador Bayesiano Ingenuo

Seleccionando entre dos clases C_k y C_j

- $$P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}$$
- $$P(C_j|x) = \frac{P(x|C_j)P(C_j)}{P(x)}$$
- Cociente:
$$\frac{P(C_k|x)}{P(C_j|x)} = \frac{P(x|C_k)P(C_k)}{P(x|C_j)P(C_j)}$$
- Si es mayor a 1 escogemos C_k, de lo contrario C_j

Clasificador Bayesiano Ingenuo

Seleccionando entre dos clases C_k y C_j

- $$P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}$$
- $$P(C_j|x) = \frac{P(x|C_j)P(C_j)}{P(x)}$$
- Cociente:
$$\frac{P(C_k|x)}{P(C_j|x)} = \frac{P(x|C_k)P(C_k)}{P(x|C_j)P(C_j)}$$
- Si es mayor a 1 escogemos C_k, de lo contrario C_j
- Entre K categorías: escogemos la categoría k para la que el valor de $P(x|C_k)P(C_k)$ es mayor

Clasificador Bayesiano Ingenuo

- Cantidad clave:

$$P(x|C_k)P(C_k)$$

Clasificador Bayesiano Ingenuo

- Cantidad clave:

$$P(x|C_k)P(C_k)$$

- $P(C_k)$: estimar como fracción de observaciones en categoría C_k

Clasificador Bayesiano Ingenuo

- Cantidad clave:

$$P(x|C_k)P(C_k)$$

- $P(C_k)$: estimar como fracción de observaciones en categoría C_k
- $P(x|C_k)$: más complejo

Clasificador Bayesiano Ingenuo

- Cantidad clave:

$$P(x|C_k)P(C_k)$$

- $P(C_k)$: estimar como fracción de observaciones en categoría C_k
- $P(x|C_k)$: más complejo
- *Ingenuidad*: suponer un modelo *paramétrico* sencillo para $P(x|C_k)$

Clasificador Bayesiano Ingenuo

- Cantidad clave:
$$P(x|C_k)P(C_k)$$
- $P(C_k)$: estimar como fracción de observaciones en categoría C_k
- $P(x|C_k)$: más complejo
- *Ingenuidad*: suponer un modelo *paramétrico* sencillo para $P(x|C_k)$
- Modelo **generativo**

Clasificador Bayesiano Ingenuo Gaussiano

- Los datos en cada categoría tienen una distribución normal/Gaussiana

Clasificador Bayesiano Ingenuo Gaussiano

- Los datos en cada categoría tienen una distribución normal/Gaussiana
- A partir de los datos de entrenamiento en C_k estimar la media μ_k y la varianza σ_k^2 (parámetros) de los datos en C_k

Clasificador Bayesiano Ingenuo Gaussiano

- Los datos en cada categoría tienen una distribución normal/Gaussiana
- A partir de los datos de entrenamiento en C_k estimar la media μ_k y la varianza σ_k^2 (parámetros) de los datos en C_k
- Calcular $P(x|C_k)$ como la probabilidad de obtener x como resultado de una distribución normal con media y varianza calculadas

Clasificador Bayesiano Ingenuo Gaussiano

- Los datos en cada categoría tienen una distribución normal/Gaussiana
- A partir de los datos de entrenamiento en C_k estimar la media μ_k y la varianza σ_k^2 (parámetros) de los datos en C_k
- Calcular $P(x|C_k)$ como la probabilidad de obtener x como resultado de una distribución normal con media y varianza calculadas
- Para características continuas usar densidad de probabilidad $f(x|C_k)$

Clasificador Bayesiano Ingenuo Gaussiano

- Ejemplo con una sola característica (o cada característica independiente):

$$f(x|C_k) = \frac{1}{(2\pi\sigma)^{1/2}} \exp\{-\frac{1}{2}\frac{(x-\mu_k)}{\sigma_k}\}$$

Clasificador Bayesiano Ingenuo Gaussiano

- Ejemplo con una sola característica (o cada característica independiente):

$$f(x|C_k) = \frac{1}{(2\pi\sigma)^{1/2}} \exp\{-\frac{1}{2}\frac{(x-\mu_k)}{\sigma_k}\}$$

- Ejemplo con D características, vector promedio μ_k y matriz de covarianzas Σ:

$$f(x|C_k) = \frac{1}{2\pi^{D/2}\Sigma} \exp\{-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)\}$$

Scikit-Learn

Scikit-Learn

- Librería para Python con múltiples algoritmos de aprendizaje de máquina

Scikit-Learn

- Librería para Python con múltiples algoritmos de aprendizaje de máquina
- Uso uniforme de los métodos disponibles (API)

Scikit-Learn

- Librería para Python con múltiples algoritmos de aprendizaje de máquina
- Uso uniforme de los métodos disponibles (API)
- Datos: tablas/dataframes

Scikit-Learn

- Librería para Python con múltiples algoritmos de aprendizaje de máquina
- Uso uniforme de los métodos disponibles (API)
- Datos: tablas/dataframes
- Tabla de características (features - X)

Scikit-Learn

- Librería para Python con múltiples algoritmos de aprendizaje de máquina
- Uso uniforme de los métodos disponibles (API)
- Datos: tablas/dataframes
- Tabla de características (features - X)
- Filas: observaciones (`n_samples`)
- Columnas: características (`n_features`)

Scikit-Learn

- Librería para Python con múltiples algoritmos de aprendizaje de máquina
- Uso uniforme de los métodos disponibles (API)
- Datos: tablas/dataframes
- Tabla de características (features - X)
- Filas: observaciones (`n_samples`)
- Columnas: características (`n_features`)
- Tabla de etiquetas (targets - y)

Scikit-Learn

- Librería para Python con múltiples algoritmos de aprendizaje de máquina
- Uso uniforme de los métodos disponibles (API)
- Datos: tablas/dataframes
- Tabla de características (features - X)
- Filas: observaciones (`n_samples`)
- Columnas: características (`n_features`)
- Tabla de etiquetas (targets - y)
- Filas: observaciones (`n_samples`)
- Columnas: etiquetas (`n_targets`)

Scikit-Learn

- Número de métodos limitado y similares

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas
- Incluye valores de parámetros por defecto

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas
- Incluye valores de parámetros por defecto
- API:

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas
- Incluye valores de parámetros por defecto
- API:
  - Importar la clase de modelo buscado

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas
- Incluye valores de parámetros por defecto
- API:
  - Importar la clase de modelo buscado
  - Escoger parámetros

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas
- Incluye valores de parámetros por defecto
- API:
  - Importar la clase de modelo buscado
  - Escoger parámetros
  - Determinar matrices de características y objetivos

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas
- Incluye valores de parámetros por defecto
- API:
  - Importar la clase de modelo buscado
  - Escoger parámetros
  - Determinar matrices de características y objetivos
  - Ajustar el modelo a los datos con el método `fit()`

Scikit-Learn

- Número de métodos limitado y similares
- Datos representados con arreglos de Numpy y dataframes de pandas
- Datos representados con arreglos de Numpy y dataframes de pandas
- Incluye valores de parámetros por defecto
- API:
  - Importar la clase de modelo buscado
  - Escoger parámetros
  - Determinar matrices de características y objetivos
  - Ajustar el modelo a los datos con el método `fit()`
  - Aplicar el modelo a nuevos datos con el método `predict()` (aprendizaje supervisado)

Librería Seaborn

- Librería para graficar en Python
- Seaborn y scikit-learn disponibles en **Anaconda**
- Ambiente de desarrollo: **Spyder**

Usando Python para visualizar datos Gaussianos

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import make_blobs

X,y = make_blobs(100, 2, centers=2, random_state=1, \
   cluster_std=1.5)
plt.scatter(X[:,0], X[:,1], c=y, s=50, cmap='RdBu')
```

make_blobs(número de muestras, número de características, número d centroides, semillas, desviación estándar)

El Clasificador Bayesiano Ingenuo en Scikit Learn

```
from sklearn.naive_bayes import GaussianNB
modelo = GaussianNB()
modelo.fit(X,y)
```

El Clasificador Bayesiano Ingenuo en Scikit Learn

```
rng = np.random.RandomState(0)
Xnew = [-15, -8] + [17, 16] * rng.rand(1000, 2)
ynew = modelo.predict(Xnew)
```

El Clasificador Bayesiano Ingenuo en Scikit Learn

```
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu')
lim = plt.axis()
plt.scatter(Xnew[:, 0], Xnew[:, 1], c=ynew, s=20, \
  cmap='RdBu', alpha=0.1)
plt.axis(lim)
```

El Clasificador Bayesiano Ingenuo en Scikit Learn

```
yprob = modelo.predict_proba(Xnew)
print(yprob[0:9])
```

Datos Iris

```
import seaborn as sns
iris = sns.load_dataset('iris')
print(iris.head())
sns.pairplot(iris, hue = 'species', size = 1.5)
```

Clasificador Bayesiano Ingenuo para los datos Iris

```
import seaborn as sns
iris = sns.load_dataset('iris')
print(iris.head())
```

Seleccionar datos de entrenamiento y prueba

```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, ytrain, ytest = \
  train_test_split(X_iris, y_iris, random_state=1)
print(Xtrain.shape)
print(Xtest.shape)
```

Importar y Usar el Clasificador Bayesiano Ingenuo

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(Xtrain, ytrain)
y_model = model.predict(Xtest)
```

Evaluar la Precisión del método

```
from sklearn.metrics import accuracy_score
import numpy
numpy.set_printoptions(threshold=numpy.nan)
acc_score = accuracy_score(ytest, y_model)
print("Precisión: ", acc_score)
print("Test")
print( ytest.values)
print( "Modelo")
print( y_model)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
from sklearn.datasets import load_digits
digits = load_digits()
print(digits.images.shape)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
import matplotlib.pyplot as plt
fig, axes = plt.subplots(
  10, 10, figsize=(8, 8),
  subplot_kw={'xticks':[], 'yticks':[]},
  gridspec_kw=dict(hspace=0.1, wspace=0.1)
  )
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
for i, ax in enumerate(axes.flat):
    ax.imshow(digits.images[i])
    ax.text(0.05, 0.05, str(digits.target[i]),
            color='green'
            )
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
X = digits.data
print("Tamaño X: ", X.shape)
y = digits.target
print("Tamaño y: ", y.shape)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
import pandas as pd
X = pd.DataFrame(X)
y = pd.Series(y)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, ytrain, ytest = \
  train_test_split(X, y, random_state=0)
print("Tamaño Xtrain: ", Xtrain.shape)
print("Tamaño Xtest: ", Xtest.shape)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
from sklearn.naive_bayes import \
    GaussianNB
modelo = GaussianNB()
modelo.fit(Xtrain, ytrain)
y_modelo = modelo.predict(Xtest)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
from sklearn.metrics import accuracy_score
acc_score = accuracy_score(ytest, y_modelo)
print("Precisión: ", acc_score)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
from sklearn.metrics import confusion_matrix
con_mat = confusion_matrix(ytest, y_modelo)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
import seaborn as sns
plt.clf()
sns.heatmap(con_mat, square=True, \
  annot=True, cbar=False)
plt.xlabel('valor predicho')
plt.ylabel('valor real')
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
fig, axes = plt.subplots( \
  10, 10, figsize=(8, 8), \
        subplot_kw={'xticks':[], 'yticks':[]}, \
        gridspec_kw=dict(hspace=0.1, wspace=0.1) \
)
```

Clasificador Bayesiano Ingenuo para Reconocer Dígitos

```
for i, ax in enumerate(axes.flat):
    ax.imshow(digits.images[ytest.index[i]], \
      cmap='binary', interpolation='nearest')
    ax.text(0.05, 0.05, str(y_modelo[i]), \
      color='green' \
      if (ytest.values[i] == y_modelo[i]) \
                              else 'red'
      )
```

Support Vector Machines

Support Vector Machines

Máquinas de Vectores de Soporte

- Considere el problema de encontrar un clasificador lineal

Support Vector Machines

Máquinas de Vectores de Soporte

- Considere el problema de encontrar un clasificador lineal
- Vector de características: x

Support Vector Machines

Máquinas de Vectores de Soporte

- Considere el problema de encontrar un clasificador lineal
- Vector de características: x
- Encontrar una función $y(x)$

$$y(x) = w^T x + w_0$$

Support Vector Machines

Máquinas de Vectores de Soporte

- Considere el problema de encontrar un clasificador lineal
- Vector de características: x
- Encontrar una función $y(x)$

$$y(x) = w^T x + w_0$$

- Determinar los parámetros w y w_0

Support Vector Machines

Máquinas de Vectores de Soporte

- Datos de entrenamiento: características x_i, $i = 1, \ldots, N$

Support Vector Machines

Máquinas de Vectores de Soporte

- Datos de entrenamiento: características x_i, $i = 1, \ldots, N$
- Etiquetas o targets $t_i \in \{-1, 1\}$

Support Vector Machines

Máquinas de Vectores de Soporte

- Datos de entrenamiento: características x_i, $i = 1, \dots, N$
- Etiquetas o targets $t_i \in \{-1, 1\}$
- y clasifica correctamente a x_i si $y(x_i) > 0$ para $t_i = 1$ y $y(x_i) < 0$ para $t_i = -1$

Support Vector Machines

Máquinas de Vectores de Soporte

- Datos de entrenamiento: características x_i, $i = 1, \ldots, N$
- Etiquetas o targets $t_i \in \{-1, 1\}$
- y clasifica correctamente a x_i si $y(x_i) > 0$ para $t_i = 1$ y $y(x_i) < 0$ para $t_i = -1$
- Para todos los puntos bien clasificados (e.g., datos linealmente separables)

$$t_i y(x_i) > 0$$

Support Vector Machines

Máquinas de Vectores de Soporte

- Distancia de un punto x al plano $y(x) = w^T x + w_0 = 0$

$$\frac{|y(x)|}{||w||}$$

Support Vector Machines

Máquinas de Vectores de Soporte

- Distancia de un punto x al plano $y(x) = w^T x + w_0 = 0$

$$\frac{|y(x)|}{||w||}$$

- Buscamos los parámetros w y w_0 que maximizen la distancia al plano de clasificación con puntos correctamente clasificados

Support Vector Machines

Máquinas de Vectores de Soporte

- Distancia de un punto x al plano $y(x) = w^T x + w_0 = 0$

$$\frac{|y(x)|}{||w||}$$

- Buscamos los parámetros w y w_0 que maximizen la distancia al plano de clasificación con puntos correctamente clasificados
- Distancia de punto x_n a superficie de decisión es

$$\frac{t_i y(x_i)}{||w||}$$

Support Vector Machines

Máquinas de Vectores de Soporte

- Distancia de un punto x al plano $y(x) = w^T x + w_0 = 0$

$$\frac{|y(x)|}{||w||}$$

- Buscamos los parámetros w y w_0 que maximizen la distancia al plano de clasificación con puntos correctamente clasificados
- Distancia de punto x_n a superficie de decisión es

$$\frac{t_i y(x_i)}{||w||}$$

- Problema de optimización:

$$\arg\max \left\{ \frac{1}{||w||} \min_i (t_i(w^T x_i + w_0)) \right\}$$

Support Vector Machines en Scikit Learn

```
import numpy as np
import matplotlib.pyplot as plt

from sklearn.datasets.samples_generator import \
    make_blobs
X, y = make_blobs(n_samples=50, centers=2,
    random_state=0, cluster_std=0.60)
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, \
    cmap='autumn')
```

Support Vector Machines en Scikit Learn

```
from sklearn.svm import SVC
model = SVC(kernel='linear', C=1E10)
model.fit(X, y)
```

Support Vector Machines en Scikit Learn

```
def plot_svc(model):
    ax = plt.gca()
    xlim = ax.get_xlim()
    ylim = ax.get_ylim()
    x = np.linspace(xlim[0], xlim[1], 30)
    y = np.linspace(ylim[0], ylim[1], 30)
    X, Y = np.meshgrid(x, y)
    xy = np.vstack([X.ravel(), Y.ravel()]).T
    P = model.decision_function(xy).reshape(X.shape)

    ax.contour(X, Y, P, colors='k', \
      levels=[-1, 0, 1], alpha=0.5,
      linestyles=['--', '-', '--'])
```

Support Vector Machines en Scikit Learn

```
plot_svm(model)

print("Vectores de soporte:\n", \
  model.support_vectors_)
```

Support Vector Machines en Scikit Learn

```
plot_svm(model)

print("Vectores de soporte:\n", \
  model.support_vectors_)
```

Cambie el número de muestras y repita

Datos menos separables

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets.samples_generator import \
    make_blobs

X, y = make_blobs(n_samples=100, centers=2, \
    random_state=0, cluster_std=1)
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, \
    cmap='autumn');
```

Datos menos separables

```
from sklearn.svm import SVC
model = SVC(kernel='linear', C=1E10)
model.fit(X, y)
```

Datos menos separables

```
from sklearn.svm import SVC
model = SVC(kernel='linear', C=1E10)
model.fit(X, y)
```

Cambie C por 1000, 10 y 0.1

C: parámetro de penalización

- Si datos no son linealmente separables, el problema no tiene solución

C: parámetro de penalización

- Si datos no son linealmente separables, el problema no tiene solución
- Permitir un error para poder analizar datos no linealmente separables

C: parámetro de penalización

- Si datos no son linealmente separables, el problema no tiene solución
- Permitir un error para poder analizar datos no linealmente separables
- El error se penaliza con el parámetro C

C: parámetro de penalización

- Si datos no son linealmente separables, el problema no tiene solución
- Permitir un error para poder analizar datos no linealmente separables
- El error se penaliza con el parámetro C
- A mayor C menos tolerancia con el error

C: parámetro de penalización

- Si datos no son linealmente separables, el problema no tiene solución
- Permitir un error para poder analizar datos no linealmente separables
- El error se penaliza con el parámetro C
- A mayor C menos tolerancia con el error
- Suaviza el margen

Datos no Linealmente Separables

```
import matplotlib.pyplot as plt
from sklearn.datasets.samples_generator \
  import make_circles
X, y = make_circles(100, factor=.1, noise=.1)
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, \
  cmap='autumn')
```

Proyección en un Espacio de más Dimensiones

- Datos no separables en el espacio de las características

Proyección en un Espacio de más Dimensiones

- Datos no separables en el espacio de las características
- Proyección en un espacio de más dimensiones

Proyección en un Espacio de más Dimensiones

- Datos no separables en el espacio de las características
- Proyección en un espacio de más dimensiones
- Ejemplo:

$$z = \exp(-(x_1^2 + x_2^2))$$

Proyección en un Espacio de más Dimensiones

- Datos no separables en el espacio de las características
- Proyección en un espacio de más dimensiones
- Ejemplo:

$$z = \exp(-(x_1^2 + x_2^2))$$

- Intentar separar los datos linealmente en el espacio de mayor dimensión

Proyección en un Espacio de más Dimensiones

- Datos no separables en el espacio de las características
- Proyección en un espacio de más dimensiones
- Ejemplo:

$$z = \exp(-(x_1^2 + x_2^2))$$

- Intentar separar los datos linealmente en el espacio de mayor dimensión
- Transformación de las características $\phi(x)$

Proyección en un Espacio de más Dimensiones

- Datos no separables en el espacio de las características
- Proyección en un espacio de más dimensiones
- Ejemplo:

$$z = \exp(-(x_1^2 + x_2^2))$$

- Intentar separar los datos linealmente en el espacio de mayor dimensión
- Transformación de las características $\phi(x)$
- Nuevo problema: encontrar lo parámetros de la función y

$$y(x) = w^T \phi(x) + w_0$$

Proyección en un Espacio de más Dimensiones

```
from mpl_toolkits import mplot3d

z = np.exp(-(X ** 2).sum(1))
ax = plt.subplot(projection='3d')
ax.scatter3D(X[:, 0], X[:, 1], z, c=y, s=50, \
    cmap='autumn')
ax.view_init(elev=30, azim=30)
ax.set_xlabel('x1')
ax.set_ylabel('x2')
ax.set_zlabel('z')
```

Proyección en un Espacio de más Dimensiones

- ¿Cuál proyección $\phi(\cdot)$ usar?

Proyección en un Espacio de más Dimensiones

- ¿Cuál proyección $\phi(\cdot)$ usar?
- Espacios de características de muchas dimensiones

Proyección en un Espacio de más Dimensiones

- ¿Cuál proyección $\phi(\cdot)$ usar?
- Espacios de características de muchas dimensiones
- Kernel: construir proyecciones a partir de relaciones entre pares de puntos

Proyección en un Espacio de más Dimensiones

- ¿Cuál proyección $\phi(\cdot)$ usar?
- Espacios de características de muchas dimensiones
- Kernel: construir proyecciones a partir de relaciones entre pares de puntos
- Ejemplo: permitir que cada punto sea el centro de la función de base radial

Proyección en un Espacio de más Dimensiones

- ¿Cuál proyección $\phi(\cdot)$ usar?
- Espacios de características de muchas dimensiones
- Kernel: construir proyecciones a partir de relaciones entre pares de puntos
- Ejemplo: permitir que cada punto sea el centro de la función de base radial
- Escoger la mejor transformación de una familia

Proyección en un Espacio de más Dimensiones

- ¿Cuál proyección $\phi(\cdot)$ usar?
- Espacios de características de muchas dimensiones
- Kernel: construir proyecciones a partir de relaciones entre pares de puntos
- Ejemplo: permitir que cada punto sea el centro de la función de base radial
- Escoger la mejor transformación de una familia
- Scikit Learn: linear, poly, rbf, sigmoid

Kernel SVM en Scikit Learn

```
from sklearn.svm import SVC
model_kernel = SVC(kernel='rbf', C=1E6)
model_kernel.fit(X, y)
plot_svc(model_kernel)
```

Reconocimiento Facial con SVM

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.datasets import fetch_lfw_people
faces = fetch_lfw_people(min_faces_per_person=60)
print(faces.target_names)
print(faces.images.shape)
```

Reconocimiento Facial con SVM

```
fig, ax = plt.subplots(3, 5)
for i, axi in enumerate(ax.flat):
    axi.imshow(faces.images[i], cmap='bone')
    axi.set(xticks=[], yticks=[],
      xlabel=faces.target_names[faces.target[i]])
```

Reconocimiento Facial con SVM

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

pca = PCA(svd_solver='randomized', n_components=150, \
  whiten=True, random_state=42)
svc = SVC(kernel='rbf', class_weight='balanced')
model = make_pipeline(pca, svc)
```

Reconocimiento Facial con SVM

```
from sklearn.cross_validation import train_test_split
Xtrain, Xtest, ytrain, ytest = \
    train_test_split(faces.data, faces.target, \
        random_state=42)
```

Reconocimiento Facial con SVM

```
from sklearn.model_selection import GridSearchCV
param_grid = {'svc__C': [1, 5, 10, 50], \
  'svc__gamma': [0.0001, 0.0005, 0.001, 0.005]}
grid = GridSearchCV(model, param_grid)
grid.fit(Xtrain, ytrain)
```

Reconocimiento Facial con SVM

```
model = grid.best_estimator_
yfit = model.predict(Xtest)
```

Reconocimiento Facial con SVM

```
fig, ax = plt.subplots(8, 5)
for i, axi in enumerate(ax.flat):
    axi.imshow(Xtest[i].reshape(62, 47), cmap='bone')
    axi.set(xticks=[], yticks=[])
    axi.set_ylabel( \
                   faces.target_names[yfit[i]].split()[
      color='black' if yfit[i] == ytest[i] \
                         else 'red')
```

Reconocimiento Facial con SVM

```
from sklearn.metrics import confusion_matrix
mat = confusion_matrix(ytest, yfit)
sns.heatmap(mat.T, square=True, annot=True, \
  fmt='d', cbar=False,
  xticklabels=faces.target_names,
  yticklabels=faces.target_names)
plt.xlabel('true label')
plt.ylabel('predicted label')
```

Selección de Variables/Características

Selección de Variables/Características

- Decision Trees
- Gran número de características

Selección de Variables/Características

- Decision Trees
- Gran número de características
- Ejemplo (dígitos): si cada dígito se representa en un cuadro de 8x8 pixeles, tenemos 64 características

Selección de Variables/Características

- Decision Trees
- Gran número de características
- Ejemplo (dígitos): si cada dígito se representa en un cuadro de 8x8 pixeles, tenemos 64 características
- Muchas características: problemas de desempeño de los algoritmos, costo computacional, sobre-ajuste (overfitting)

Selección de Variables/Características

- Decision Trees
- Gran número de características
- Ejemplo (dígitos): si cada dígito se representa en un cuadro de 8x8 pixeles, tenemos 64 características
- Muchas características: problemas de desempeño de los algoritmos, costo computacional, sobre-ajuste (overfitting)
- ¿Cuáles características seleccionar?

Selección de Variables/Características

- Reducción de dimensionalidad:

Selección de Variables/Características

- Reducción de dimensionalidad:
  - Selección de características (feature selection)

Selección de Variables/Características

- Reducción de dimensionalidad:
  - Selección de características (feature selection)
  - Extracción de características (feature extraction)

Selección de Variables/Características

- Reducción de dimensionalidad:
  - Selección de características (feature selection)
  - Extracción de características (feature extraction)
    - Proyectar el espacio original de alta dimensionalidad en un espacio de dimensión reducida

Selección de Variables/Características

- Reducción de dimensionalidad:
  - Selección de características (feature selection)
  - Extracción de características (feature extraction)
    - Proyectar el espacio original de alta dimensionalidad en un espacio de dimensión reducida
    - Combinación (lineal o no lineal) de las características originales

Selección de Variables/Características

- Reducción de dimensionalidad:
  - Selección de características (feature selection)
  - Extracción de características (feature extraction)
    - Proyectar el espacio original de alta dimensionalidad en un espacio de dimensión reducida
    - Combinación (lineal o no lineal) de las características originales
    - Ejemplo: análisis de componentes principales

Selección de Variables/Características

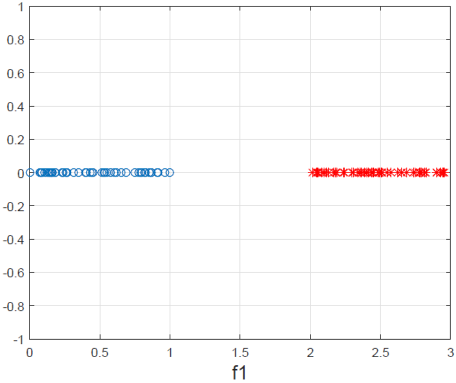
- Reducción de dimensionalidad:
  - Selección de características (feature selection)
  - Extracción de características (feature extraction)
    - Proyectar el espacio original de alta dimensionalidad en un espacio de dimensión reducida
    - Combinación (lineal o no lineal) de las características originales
    - Ejemplo: análisis de componentes principales
    - Limitación: interpretabilidad y análisis de resultados

Selección de Variables/Características

- Reducción de dimensionalidad:
  - Selección de características (feature selection)
  - Extracción de características (feature extraction)
    - Proyectar el espacio original de alta dimensionalidad en un espacio de dimensión reducida
    - Combinación (lineal o no lineal) de las características originales
    - Ejemplo: análisis de componentes principales
    - Ventaja: mecanismos eficientes para reducir la dimensionalidad
    - Limitación: interpretabilidad y análisis de resultados

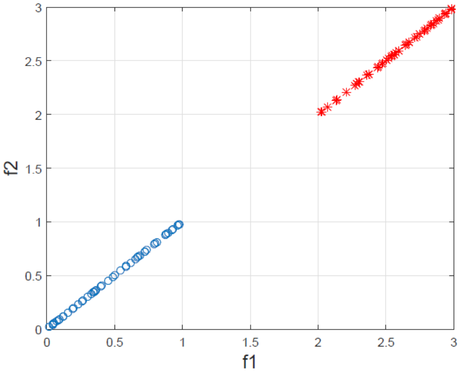
¿Con qué criterio seleccionamos características?

Característica relevante:



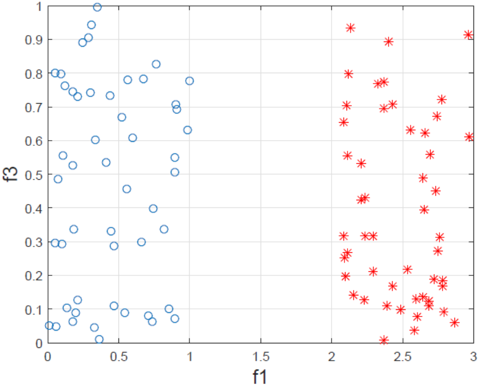
¿Con qué criterio seleccionamos características?

Característica relevante pero redundante en presencia de la primera:



¿Con qué criterio seleccionamos características?

Característica no relevante:



Tipos de métodos de selección

De acuerdo con uso de etiquetas:

Tipos de métodos de selección

De acuerdo con uso de etiquetas:

- Supervisados: usan etiquetas para evaluar qué variables seleccionar (clasificación, regresión)

Tipos de métodos de selección

De acuerdo con uso de etiquetas:

- Supervisados: usan etiquetas para evaluar qué variables seleccionar (clasificación, regresión)
- No supervisados: no usan etiquetas, buscan otros criterios de evaluación (clustering)

Tipos de métodos de selección

De acuerdo con la estrategia de selección:

Tipos de métodos de selección

De acuerdo con la estrategia de selección:

- Filtros: independientes del método de aprendizaje, se aplican previamente a la ejecución del método

Tipos de métodos de selección

De acuerdo con la estrategia de selección:

- Filtros: independientes del método de aprendizaje, se aplican previamente a la ejecución del método
- Wrappers (envolventes): utilizan el resultado de un algoritmo de aprendizaje para evaluar qué variables seleccionar/descartar (problema de búsqueda del mejor conjunto de variables)

Tipos de métodos de selección

De acuerdo con la estrategia de selección:

- Filtros: independientes del método de aprendizaje, se aplican previamente a la ejecución del método
- Wrappers (envolventes): utilizan el resultado de un algoritmo de aprendizaje para evaluar qué variables seleccionar/descartar (problema de búsqueda del mejor conjunto de variables)
- Embebidos: incorporan la selección de características dentro del método de aprendizaje

Tipos de métodos de selección

De acuerdo con la disponibilidad de datos y características:

Tipos de métodos de selección

De acuerdo con la disponibilidad de datos y características:

- Estáticos: todos los datos y sus características están disponibles

Tipos de métodos de selección

De acuerdo con la disponibilidad de datos y características:

- Estáticos: todos los datos y sus características están disponibles
- Flujos (streams): los datos llegan continuamente y las características pueden aparecer y desaparecer (e.g., palabras en redes sociales)

Tipos de métodos de selección

De acuerdo con la estructura de las características:

Tipos de métodos de selección

De acuerdo con la estructura de las características:

- Genéricos: características planas, sin estructura

Tipos de métodos de selección

De acuerdo con la estructura de las características:

- Genéricos: características planas, sin estructura
- Estructurados: características con estructuras (e.g., árboles, grafos, grupos)

Datos genéricos y estáticos

Tipos de Métodos:

Datos genéricos y estáticos

Tipos de Métodos:

- Estadísticas

Datos genéricos y estáticos

Tipos de Métodos:

- Estadísticas
- Teoría de la información

Datos genéricos y estáticos

Tipos de Métodos:

- Estadísticas
- Teoría de la información
- Dispersión (regularización)

Datos genéricos y estáticos

Tipos de Métodos:

- Estadísticas
- Teoría de la información
- Dispersión (regularización)
- Similaridad

Datos genéricos y estáticos

Tipos de Métodos:

- Estadísticas
- Teoría de la información
- Dispersión (regularización)
- Similaridad
- Otros

Baja Varianza

Baja Varianza

- Eliminar características que tienen una varianza menor que un umbral

Baja Varianza

- Eliminar características que tienen una varianza menor que un umbral
- Si la varianza es baja, la característica no es de gran ayuda para separar observaciones en categorías (clasificación)

Baja Varianza

- Eliminar características que tienen una varianza menor que un umbral
- Si la varianza es baja, la característica no es de gran ayuda para separar observaciones en categorías (clasificación)
- Definir un umbral

Baja Varianza en Scikit Learn

```
from pandas import DataFrame
from sklearn.feature_selection import VarianceThreshol
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data
Xdf= DataFrame(X)
print(Xdf.describe())
print(Xdf.var(ddof=0))

selector = VarianceThreshold(threshold=(0.8*0.8))
selector.fit(X)
print(selector.get_support())

Xbar = selector.transform(X)
print(Xbar)
```

T score / Estadístico T

T score / Estadístico T

- Medida que refleje si el valor promedio de una característica difiere significativamente en dos clases diferentes

T score / Estadístico T

- Medida que refleje si el valor promedio de una característica difiere significativamente en dos clases diferentes
- μ_1: valor medio de la característica en la clase 1

T score / Estadístico T

- Medida que refleje si el valor promedio de una característica difiere significativamente en dos clases diferentes
- μ_1: valor medio de la característica en la clase 1
- μ_2: valor medio de la característica en la clase 2

T score / Estadístico T

- Medida que refleje si el valor promedio de una característica difiere significativamente en dos clases diferentes
- μ_1: valor medio de la característica en la clase 1
- μ_2: valor medio de la característica en la clase 2
- σ_1: desviación estándar del valor de la característica en la clase 1

T score / Estadístico T

- Medida que refleje si el valor promedio de una característica difiere significativamente en dos clases diferentes
- μ_1: valor medio de la característica en la clase 1
- μ_2: valor medio de la característica en la clase 2
- σ_1: desviación estándar del valor de la característica en la clase 1
- σ_2: desviación estándar del valor de la característica en la clase 2

T score / Estadístico T

- Medida que refleje si el valor promedio de una característica difiere significativamente en dos clases diferentes
- μ_1: valor medio de la característica en la clase 1
- μ_2: valor medio de la característica en la clase 2
- σ_1: desviación estándar del valor de la característica en la clase 1
- σ_2: desviación estándar del valor de la característica en la clase 2
- n_1 y n_2: número de observaciones en cada clase

T score / Estadístico T

$$\text{T-score(caract)} = \frac{|\mu_1 - \mu_2|}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$$

T score / Estadístico T

$$\text{T-score(caract)} = \frac{|\mu_1 - \mu_2|}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$$

- Grande: valores de la característica diferentes para cada clase

T score / Estadístico T

$$\text{T-score(caract)} = \frac{|\mu_1 - \mu_2|}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$$

- Grande: valores de la característica diferentes para cada clase
- Pequeño: valores de la característica similares en ambas

F score / Estadístico F

F score / Estadístico F

- T-score: 2 clases
- F-score: generalización a C clases

F score / Estadístico F

- T-score: 2 clases
- F-score: generalización a C clases
- μ: valor medio de la característica

F score / Estadístico F

- T-score: 2 clases
- F-score: generalización a C clases
- μ: valor medio de la característica
- μ_j: valor medio de la característica en la clase j

F score / Estadístico F

- T-score: 2 clases
- F-score: generalización a C clases
- μ: valor medio de la característica
- μ_j: valor medio de la característica en la clase j
- σ_j: desviación estándar del valor de la característica en la clase j

F score / Estadístico F

- T-score: 2 clases
- F-score: generalización a C clases
- μ: valor medio de la característica
- μ_j: valor medio de la característica en la clase j
- σ_j: desviación estándar del valor de la característica en la clase j
- n_j: número de observaciones en la clase j

T score / Estadístico T

$$\text{F-score(caract)} = \frac{\sum_j \frac{n_j}{C-1}(\mu_j - \mu)^2}{\frac{1}{n-C}\sum_j (n_j - 1)\sigma_j^2}$$

T score / Estadístico T

$$\text{F-score(caract)} = \frac{\sum_j \frac{n_j}{C-1}(\mu_j - \mu)^2}{\frac{1}{n-C}\sum_j (n_j - 1)\sigma_j^2}$$

- Grande: valores de la característica diferentes en diferentes clases

T score / Estadístico T

$$\text{F-score(caract)} = \frac{\sum_j \frac{n_j}{C-1}(\mu_j - \mu)^2}{\frac{1}{n-C}\sum_j (n_j - 1)\sigma_j^2}$$

- Grande: valores de la característica diferentes en diferentes clases
- Pequeño: valores de la característica similares en diferentes clases

F-score en Scikit-Learn

```
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
iris = load_iris()
X, y = iris.data, iris.target
print(X.shape)

scores = f_classif(X,y)
print(scores)

selector = SelectKBest(f_classif, k=2)
X_new = selector.fit_transform(X, y)
print(X_new.shape)
print(selector.get_support())
```

Chi^2 score / Estadístico Chi^2

Chi^2 score / Estadístico Chi^2

- n_{jk}: número de observaciones en la clase j con valor de la característica en categoría k (discreta)

Chi^2 score / Estadístico Chi^2

- n_{jk}: número de observaciones en la clase j con valor de la característica en categoría k (discreta)
- n_{j*}: número de observaciones en la clase j

Chi^2 score / Estadístico Chi^2

- n_{jk}: número de observaciones en la clase j con valor de la característica en categoría k (discreta)
- n_{j*}: número de observaciones en la clase j
- n_{*k}: número de observaciones con valor de la característica en categoría k (discreta)

Chi^2 score / Estadístico Chi^2

- n_{jk}: número de observaciones en la clase j con valor de la característica en categoría k (discreta)
- n_{j*}: número de observaciones en la clase j
- n_{*k}: número de observaciones con valor de la característica en categoría k (discreta)
- $$o_{jk} = \frac{n_{*k} n_{j*}}{n} = \frac{n_{*k}}{n} n_{j*} = n_{*k} \frac{n_{j*}}{n}$$

Chi^2 score / Estadístico Chi^2

- n_{jk}: número de observaciones en la clase j con valor de la característica en categoría k (discreta)
- n_{j*}: número de observaciones en la clase j
- n_{*k}: número de observaciones con valor de la característica en categoría k (discreta)
- $$o_{jk} = \frac{n_{*k} n_{j*}}{n} = \frac{n_{*k}}{n} n_{j*} = n_{*k} \frac{n_{j*}}{n}$$
- o_{jk}: número esperado de observaciones en la clase j con valor de la característica en categoría k (discreta) si las clases y la característica son independientes

Chi^2 score / Estadístico Chi^2

$$\text{Chi}^2\text{-score}(\text{caract}) = \sum_{j=1}^{C} \sum_{k=1}^{r} \frac{(n_{jk} - o_{jk})^2}{o_{jk}}$$

Chi^2 score / Estadístico Chi^2

$$\text{Chi}^2\text{-score(caract)} = \sum_{j=1}^{C} \sum_{k=1}^{r} \frac{(n_{jk} - o_{jk})^2}{o_{jk}}$$

- Grande: valores esperados y observados diferentes (clases y característa no independientes)

Chi^2 score / Estadístico Chi^2

$$\text{Chi}^2\text{-score}(\text{caract}) = \sum_{j=1}^{C} \sum_{k=1}^{r} \frac{(n_{jk} - o_{jk})^2}{o_{jk}}$$

- Grande: valores esperados y observados diferentes (clases y característa no independientes)
- Pequeño: valores esperados y observados similares (clases y característa independientes)

Chi^2-score en Scikit-Learn

Con respecto al último ejemplo solo tenemos que cambiar

```
from sklearn.feature_selection import chi2

selector = SelectKBest(chi2, k=2)
```

Chi^2-score en Scikit-Learn

Mantener solamente un % de las características con el valor más alto del score

```
from sklearn.feature_selection import SelectPercentile
iris = load_iris()
X, y = iris.data, iris.target
X.shape

selector = SelectPercentile(chi2, percentile = 50)
X_new = selector.fit_transform(X, y)
X_new.shape
print(selector.get_support())

scores = chi2(X,y)
print(scores)
```

Entropía

- Medida de la incertidumbre de una variable aleatoria X

Entropía

- Medida de la incertidumbre de una variable aleatoria X
- $X \in S = \{x_1, x_2, \ldots\}$ (discreta)

Entropía

- Medida de la incertidumbre de una variable aleatoria X
- $X \in S = \{x_1, x_2, \dots\}$ (discreta)
- Función de masa de probabilidad:

$$P(x_i) = P(X = x_i)$$

Entropía

- Medida de la incertidumbre de una variable aleatoria X
- $X \in S = \{x_1, x_2, \dots\}$ (discreta)
- Función de masa de probabilidad:

$$P(x_i) = P(X = x_i)$$

- Entropía:

$$H(X) = -\sum_{x_i \in S} P(x_i) log(P(x_i))$$

Entropía

- Medida de la incertidumbre de una variable aleatoria X
- $X \in S = \{x_1, x_2, \dots\}$ (discreta)
- Función de masa de probabilidad:

$$P(x_i) = P(X = x_i)$$

- Entropía:

$$H(X) = -\sum_{x_i \in S} P(x_i) log(P(x_i))$$

- Depende solamente de $P(x_i)$, no de x_i

Entropía condicional

- Dos variables aleatorias X y Y

Entropía condicional

- Dos variables aleatorias X y Y
- $X \in S_X = \{x_1, x_2, \dots\}$
- $Y \in S_Y = \{y_1, y_2, \dots\}$

Entropía condicional

- Dos variables aleatorias X y Y
- $X \in S_X = \{x_1, x_2, \dots\}$
- $Y \in S_Y = \{y_1, y_2, \dots\}$
- Probabilidades a priori:

$$P(x_i) = P(X = x_i), \quad P(y_j) = P(Y = y_j)$$

Entropía condicional

- Dos variables aleatorias X y Y
- $X \in S_X = \{x_1, x_2, \dots\}$
- $Y \in S_Y = \{y_1, y_2, \dots\}$
- Probabilidades a priori:

$$P(x_i) = P(X = x_i), \quad P(y_j) = P(Y = y_j)$$

- Probabilidades condicionales:

$$P(x_i|y_j) = P(X = x_i|Y = y_j), \quad P(y_j|x_i) = P(Y = y_j|X = x_i)$$

Entropía condicional

- Entropía condicional de X dado Y:

$$
\begin{aligned}
H(X|Y) &\equiv H(X,Y) - H(Y) \\
&= -\sum_{y_j \in S_Y} P(y_j) \sum_{x_i \in S_X} P(x_i|y_j) log(P(x_i))
\end{aligned}
$$

Entropía condicional

- Entropía condicional de X dado Y:

$$
\begin{aligned}
H(X|Y) &\equiv H(X,Y) - H(Y) \\
&= -\sum_{y_j \in S_Y} P(y_j) \sum_{x_i \in S_X} P(x_i|y_j) log(P(x_i))
\end{aligned}
$$

- Medida de la incertidumbre en X dado Y

Entropía condicional

- Entropía condicional de X dado Y:

$$H(X|Y) \equiv H(X,Y) - H(Y)$$
$$= -\sum_{y_j \in S_Y} P(y_j) \sum_{x_i \in S_X} P(x_i|y_j) log(P(x_i))$$

- Medida de la incertidumbre en X dado Y
- $H(X,Y)$ medida de incertidumbre de X y Y (distribución conjunta)

Entropía condicional

- Entropía condicional de X dado Y:

$$H(X|Y) \equiv H(X,Y) - H(Y)$$
$$= -\sum_{y_j \in S_Y} P(y_j) \sum_{x_i \in S_X} P(x_i|y_j) log(P(x_i))$$

- Medida de la incertidumbre en X dado Y
- $H(X,Y)$ medida de incertidumbre de X y Y (distribución conjunta)
- $$0 \leq H(X|Y) \leq H(X)$$

Ganancia en información / Información mutua

- Ganancia en información / Información mutua:

$$
\begin{aligned}
I(X;Y) &\equiv H(X) + H(Y) - H(X,Y) \\
&= H(X) - H(X|Y) \\
&= H(Y) - H(Y|X) \\
&= \sum_{x_i \in S_X} \sum_{y_j \in S_Y} P(x_i, y_j) \log \frac{P(x_i, y_j)}{P(x_i)P(y_j)}
\end{aligned}
$$

Ganancia en información / Información mutua

- Ganancia en información / Información mutua:

$$
\begin{aligned}
I(X;Y) &\equiv H(X) + H(Y) - H(X,Y) \\
&= H(X) - H(X|Y) \\
&= H(Y) - H(Y|X) \\
&= \sum_{x_i \in S_X} \sum_{y_j \in S_Y} P(x_i, y_j) \log \frac{P(x_i, y_j)}{P(x_i)P(y_j)}
\end{aligned}
$$

- Simétrico:

$$I(X;Y) = I(Y;X)$$

Ganancia en información / Información mutua

- Ganancia en información / Información mutua:

$$\begin{aligned} I(X;Y) &\equiv H(X) + H(Y) - H(X,Y) \\ &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= \sum_{x_i \in S_X} \sum_{y_j \in S_Y} P(x_i, y_j) \log \frac{P(x_i, y_j)}{P(x_i)P(y_j)} \end{aligned}$$

- Simétrico:

$$I(X;Y) = I(Y;X)$$

- Medida de la cantidad de información compartida por X y Y

Ganancia en información condicional/ Información mutua condicional

- Ganancia en información condicional:

$$
\begin{aligned}
I(X;Y|Z) &\equiv H(X|Z) + H(Y|Z) - H(X,Y|Z) \\
&= H(X|Z) - H(X|Y,Z) \\
&= H(Y|Z) - H(Y|X,Z) \\
&= \sum_{z_k \in S_Z} P(z_k) \sum_{x_i \in S_X} \sum_{y_j \in S_Y} P(x_i, y_j|z_k) \log \frac{P(x_i, y_j|z_k)}{P(x_i|z_k)P(y_j|z_k)}
\end{aligned}
$$

Ganancia en información condicional/ Información mutua condicional

- Ganancia en información condicional:

$$
\begin{aligned}
I(X;Y|Z) &\equiv H(X|Z) + H(Y|Z) - H(X,Y|Z) \\
&= H(X|Z) - H(X|Y,Z) \\
&= H(Y|Z) - H(Y|X,Z) \\
&= \sum_{z_k \in S_Z} P(z_k) \sum_{x_i \in S_X} \sum_{y_j \in S_Y} P(x_i, y_j|z_k) \log \frac{P(x_i, y_j|z_k)}{P(x_i|z_k) P(y_j|z_k)}
\end{aligned}
$$

- Simétrico:

$$I(X;Y) = I(Y;X)$$

Ganancia en información condicional/ Información mutua condicional

- Ganancia en información condicional:

$$
\begin{aligned}
I(X;Y|Z) &\equiv H(X|Z) + H(Y|Z) - H(X,Y|Z) \\
&= H(X|Z) - H(X|Y,Z) \\
&= H(Y|Z) - H(Y|X,Z) \\
&= \sum_{z_k \in S_Z} P(z_k) \sum_{x_i \in S_X} \sum_{y_j \in S_Y} P(x_i, y_j|z_k) \log \frac{P(x_i, y_j|z_k)}{P(x_i|z_k)P(y_j|z_k)}
\end{aligned}
$$

- Simétrico:

$$I(X;Y) = I(Y;X)$$

- Medida de la cantidad de información compartida por X y Y