Para estudiantes de Educación Básica y Media.
UNIVERSIDAD DE CHILE

Clase 12: Modelos Clasicos de Aprendizaje Supervisado

FM849 - Programación Científica para Proyectos de Inteligencia Artificial (IA)

Contenidos de esta clase

Algunos de los modelos mas utilizados en Machine Learning:

- Recuerdo sobre Aprendizaje Supervisado.
- Support Vector Machine (SVM).
- k-Nearest Neighbors (k-NN)
- Naïve Bayes.

Recuerdo de Aprendizaje Supervisado

Idea General: Tenemos pares features y etiquetas $\{\overrightarrow{x}_i, y_i\}$. Queremos encontrar un modelo h, al cual le entregaremos features de un ejemplo $\overrightarrow{x}_i$ y retornara la etiqueta y_i.

Esto es lo mismo que decir

$$h(\overrightarrow{x}_i) = y_i$$

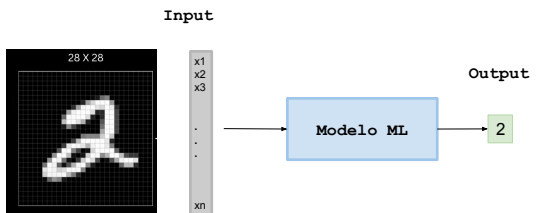


Figura 1: Esquema de un modelo ML en aprendizaje supervisado.

Recuerdo de Aprendizaje Supervisado

Estudiaremos diferentes tipos de modelos utilizados para resolver el problema de clasificación. Empezaremos dando una idea general, luego un ejemplo de como utilizar el modelo para clasificar y finalmente hablaremos de detalles que existen en cada modelo.

- k-Nearest Neighbors (k-NN)
- Naïve Bayes.
- Support Vector Machine (SVM).

Naive Bayes

Idea: Dado un conjunto de features vamos a generar una probabilidad de ocurrencia para cada clase y_i.

Dado un input x de n features $x = [x_1, x_2, ..., x_n]$ queremos obtener $P(y_i|x_1, x_2, ..., x_n)$. Para hacer esto deberiamos buscar los casos dado x ocurre y_i, lo cual es intratable computacionalmente

La idea sera entonces convertir la expresión $P(y_i|x)$ en algo mas simple y calculable. Usamos dos cosas: Teorema de Bayes e Independencia Condicional.

- Teorema de Bayes: $P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$
- Independencia Condicional: Dada una clase y_i, los features $x_1, ..., x_n$ ocurren de manera independiente.

Naive Bayes

Entonces usando el Teorema de Bayes tenemos que:

$$P(y_i|x_1, x_2, ..., x_n) = \frac{P(x_1, x_2, ..., x_n|y_i)P(y_i)}{P(x_1, x_2, ..., x_n)}$$

Usando la independencia condicional entre los features llegamos a:

$$P(y_i|x_1, x_2, ..., x_n) = \frac{P(x_1|y_i) \cdot P(x_2|y_i) \cdots P(x_n|y_i) \cdot P(y_i)}{P(x_1, x_2, ..., x_n)}$$

Podemos ver que el denominador es una constante para todas las clases del problema, entonces:

$$\hat{y} = \arg \max_{y_i \in Y} P(x_1|y_i) \cdot P(x_2|y_i) \cdots P(x_n|y_i) \cdot P(y_i)$$

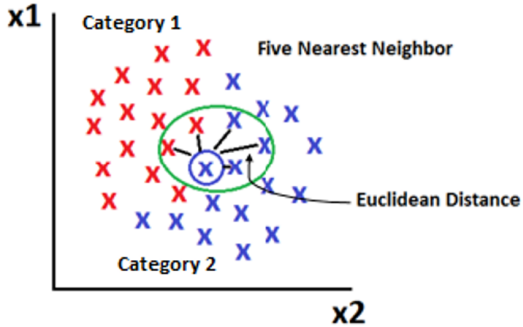
Naive Bayes: Ejemplo

Revisemos un ejemplo para entender la idea de Naive Bayes. https://docs.google.com/presentation/d/1oFK21SaZLpy28ArDHc7CaM0GWLVksyOxqw27pcC7DcM/edit?usp=sharing

k-Nearest Neighbors (kNN)

Idea: Clasificamos un dato según que tan similar es todo nuestro dataset.

- Dado un ejemplo desconocido a clasificar, busca los k ejemplos conocidos más cercanos. Por defecto, cada clase suma 1 voto si un ejemplo suyo está entre los k más cercanos.
- El nuevo ejemplo es clasificado con la clase de mayor votación.



k-Nearest Neighbors (kNN)

Tenemos que **buscar** los ejemplos más cercanos. **Como hacemos esto ?** Necesitamos una métrica de distancia para buscarlos ejemplos mas cercanos.

- Distancia euclidiana.

$$d_{eucli}(x, z) = \sqrt{\sum_{i=1}^{d}(x_i - z_i)^2}$$

Problema: Es importante escalar los datos para esta métrica, pues ciertas dimensiones en los datos pueden tener mayores magnitudes.

Ejemplo de KNN

Vamos a ver un ejemplo sobre prediccion de la especie de pinguinos. https://docs.google.com/presentation/d/1pbGDrzTU_X1yop1pkZ3rE3dJkxAQL5mLzGHv42PNERo/edit?usp=sharing

Supongamos que tenemos un dataset con dos especies de pingüinos. Cada pingüino se describe mediante dos características numéricas.

Especie	**Peso (kg)**	**Altura (cm)**	**Clase**
Emperador	35	115	Emperador
Emperador	32	110	Emperador
⋮	⋮	⋮	⋮
Humboldt	5	65	Humboldt
Humboldt	4.8	62	Humboldt

Tabla 1: Ejemplo de dataset de pingüinos

KNN: problema de escalamiento

En KNN, la clasificación se basa en **distancias** entre ejemplos.

Ejemplo: cada pingüino se describe por

- Peso (kg)
- Altura (cm)

Punto a clasificar:

$$x = (30,\ 100)$$

Dos ejemplos del dataset:

$$z_1 = (32,\ 110) \quad \text{(Emperador)}$$

$$z_2 = (5,\ 65) \quad \text{(Humboldt)}$$

Problema: La altura (cm) tiene valores mucho mayores que el peso (kg), por lo que **domina la distancia euclidiana** y el peso tiene poco impacto.

KNN: ¿por qué escalar?

Para que ambas características influyan de manera comparable, escalamos los datos (por ejemplo, con normalización min–max):

$$x_i^{\text{norm}} = \frac{x_i - \text{mín}(x_i)}{\text{máx}(x_i) - \text{mín}(x_i)}$$

Tras escalar:

- Peso y altura quedan en el mismo rango
- Ninguna variable domina a la otra
- KNN decide usando información real, no las unidades de medida

Support Vector Machine

Idea: Generar un espacio de separación (hiperplano) perpendicular a los datos que separe las clases.

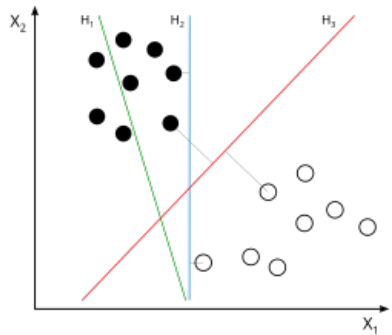


Figura 2: Ejemplo de hiperplanos. Fuente: https://en.wikipedia.org/wiki/Support_vector_machine.

Support Vector Machine

- Corresponde a un método que busca separar las clases con un hiperplano (en 2D, una recta).
- El hiperplano elegido es el que maximiza el margen que se genera con los vectores de soporte.
- Cada vez que evaluamos un nuevo punto en la ecuación del hiperplano $w \cdot x + b = 0$, clasificamos según el signo de $w \cdot x + b$.

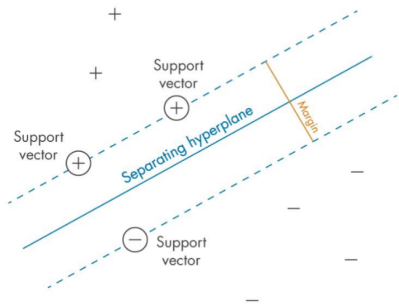


Figura 3: Ilustración de SVM.

Problema de Outliers en Support Vector Machine

Los **outliers** no permiten generar una separación perfecta cuando usamos SVM.

- En esos casos conviene generar un modelo mas flexible, es decir, le permitimos equivocarse en algunos ejemplos.
- La idea sera hacer la çalle"tan larga como sea posible con la menor cantidad de ejemplos errados. Esto se controla con un hiperparametro C del modelo.
- Esta técnica es comúnmente llamada *soft Margin*.

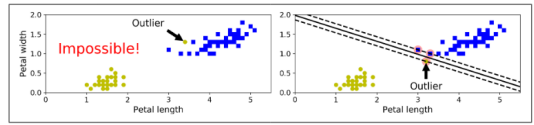


Figura 4: Ejemplos de datos con outliers.

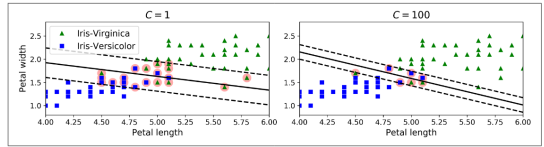


Figura 5: Modelo con *soft Margin*.

Ejemplo 1: Support Vector Machine

Supongamos que tenemos una tabla de datos con información del peso de dos especies de pingüinos. Cada fila representa un pingüino y su especie corresponde a la clase a predecir.

Peso (kg)	**Especie**
32	Emperador
35	Emperador
38	Emperador
12	Humbolt
14	Humbolt
16	Humbolt

Ejemplo 1: Support Vector Machine

Supongamos que tenemos una tabla de datos con información del peso de dos especies de pingüinos. Ahora agregamos un nuevo pingüino cuya especie es desconocida.

Peso (kg)	Especie
32	Emperador
35	Emperador
38	Emperador
12	Humbolt
14	Humbolt
16	Humbolt
25	?

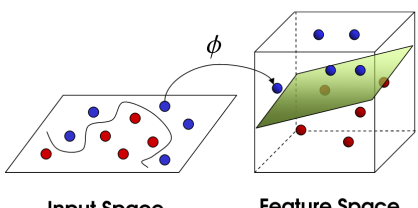
Qué especie es el pinguino en la ultima fila ?

Animated version: Google Slides with animations

Support Vector Machine: Kernel

Existen casos en los cuales los datos no son linearmente separables (por ejemplo Figura 17). En estos casos, SVM utiliza una herramienta matematica llamada **kernel**.

El kernel transforma el espacio original de input features en un nuevo espacio vectorial, de tal forma de poder separar los datos.



Ejemplo 2: Support Vector Machine

Supongamos que tenemos una tabla de datos con información del peso de dos especies de pingüinos. Ahora agregamos un nuevo pingüino cuya especie es desconocida.

Peso (kg)	Especie
32	Emperador
35	Emperador
38	Emperador
12	Humbolt
14	Humbolt
16	Humbolt
25	?

Qué especie es el pinguino en la ultima fila ?

Animated version: Google Slides with animations

Resumen de Ventajas y Desventajas de cada algoritmo

Modelo	**Ventaja**	**Desventaja**
kNN	No requiere entrenamiento explícito.	Sensible a la dimensionalidad.
NB	Eficiente incluso en alta dimensión.	Independencia condicional poco realista.
SVM	Data efficient ($\sim$ 1000).	Costoso y sensible a hiperparámetros.

Referencias:

- https://medium.com/data-science/the-math-behind-the-curse-of-dimensionality-cf8780307