Big Data

Algoritmos No Lineales: KNN y SVM*

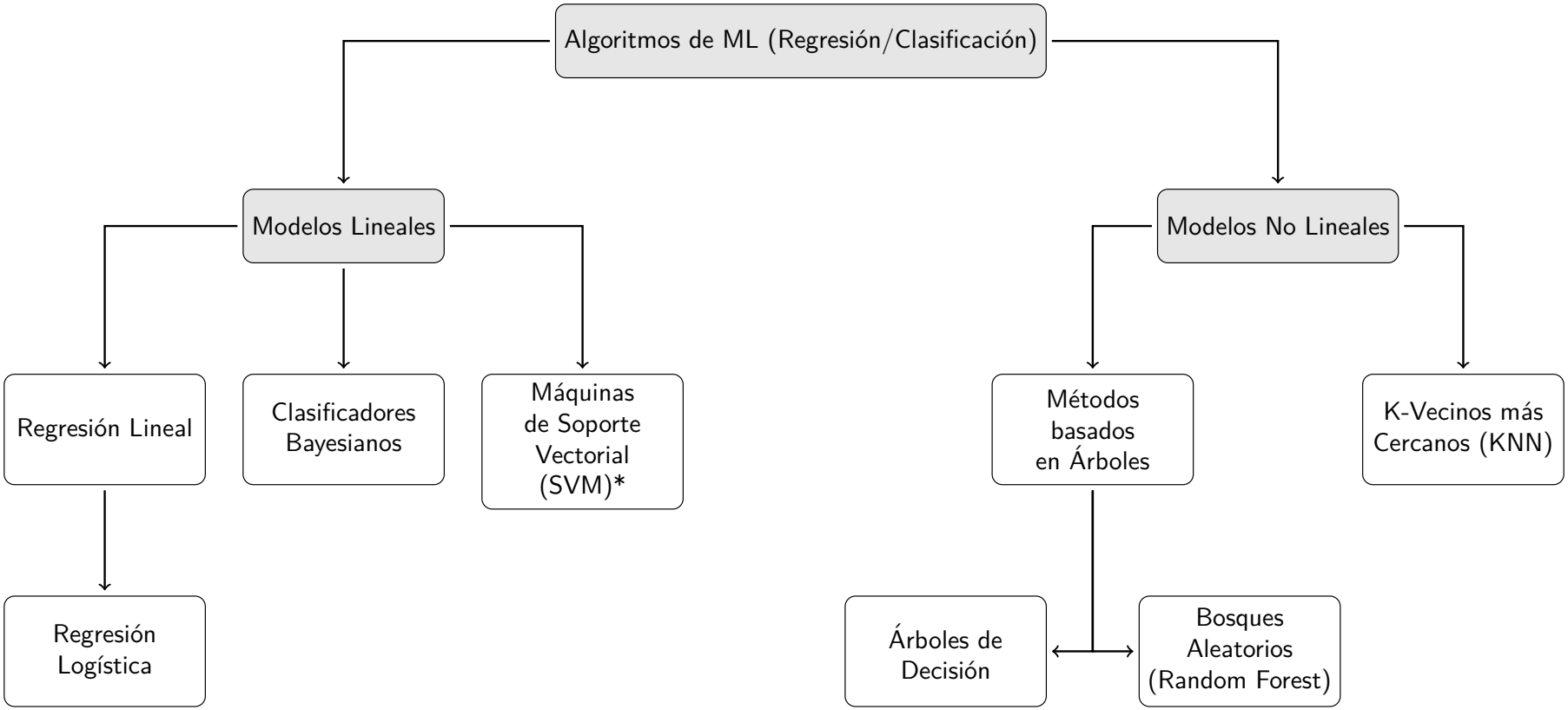
Sergio A. Cantillo
sacantillo@uao.edu.co

Facultad de Ingeniería y Ciencias Básicas
Universidad Autónoma de Occidente

Agenda

1. Contexto
2. K-Nearest Neighbors (KNN)
   - Clasificación
     - Funcionamiento
   - Regresión
     - Funcionamiento
   - Hiperparámetros
3. Support Vector Machines (SVM)
   - Clasificación
     - Funcionamiento
   - Regresión
     - Funcionamiento
   - Hiperparámetros
4. Referencias

Contexto



* Es un algoritmo de regresión/clasificación que puede ser lineal o no lineal dependiendo de su configuración.

Agenda

K-Nearest Neighbors (KNN)

Definición

Algoritmo de aprendizaje automático *no lineal* y *no paramétrico*, basado en el principio de que los datos con características similares tienden a estar cerca. KNN estima valores numéricos para una variable dependiente considerando los valores de las características de los puntos de datos cercanos en un conjunto de datos de entrenamiento.

Consideraciones

- **Similaridad de datos:** El supuesto fundamental de KNN es que puntos de datos similares (distancia) en el espacio de características pertenecen a clases similares (clasificación) o valores de variable dependiente similares (regresión).
- **Valor de k adecuado**: k representa el número de vecinos más cercanos a considerar al hacer una predicción. Un valor incorrecto de k puede llevar tanto a overfitting como a underfitting del modelo.
- **Distribución uniforme de datos**: KNN asume que los datos de entrenamiento están distribuidos uniformemente en el espacio de características.
- **Baja complejidad:** Este modelo reviste de baja complejidad de procesamiento al inferir $O(nm)$. No entrena!!.

KNN - Clasificación

Funcionamiento

Su funcionamiento es bastante simple. A diferencia de otros algoritmos de ML, **no usa una función de costo específica** ni tampoco entrena de la misma forma. Podemos resumir su funcionamiento en los siguientes pasos:

- **Decidir** el número de k y la forma de medir $distancia$ a considerar al hacer una predicción. Este valor debe ser seleccionado antes de aplicar el algoritmo (hiperparámetro).
- **Almacenar** todo el conjunto de datos de entrenamiento para los cálculos en tiempo real
- Para un punto de datos nuevo, obtener la **distancia** entre ese punto y todos los puntos en el conjunto de datos de entrenamiento.
- Determinar los valores de variable dependiente de los k vecinos más cercanos. La clase con **mayor votación** será la $\hat{y}$ en este modelo.

Agenda

KNN - Regresión

Funcionamiento

El funcionamiento coincide mayormente con lo que ocurre en clasificación, salvo la forma de determinar el valor de salida. Podemos resumir su funcionamiento en los siguientes pasos:

- **Decidir** el número de k a considerar al hacer una predicción. Este valor debe ser seleccionado antes de aplicar el algoritmo (hiperparámetro).
- **Almacenar** todo el conjunto de datos de entrenamiento para los cálculos en tiempo real
- Para un punto de datos nuevo, obtener la **distancia** entre ese punto y todos los puntos en el conjunto de datos de entrenamiento.
- **Promediar** los valores de variable dependiente de los k vecinos más cercanos. Esta sera la $\hat{y}$ en este modelo.

Agenda

KNN - Hiperparámetros

Consideraciones para Selección

En el caso de KNN, usualmente se presentan dos hiperparámetros a considerar: k y $distancia$.

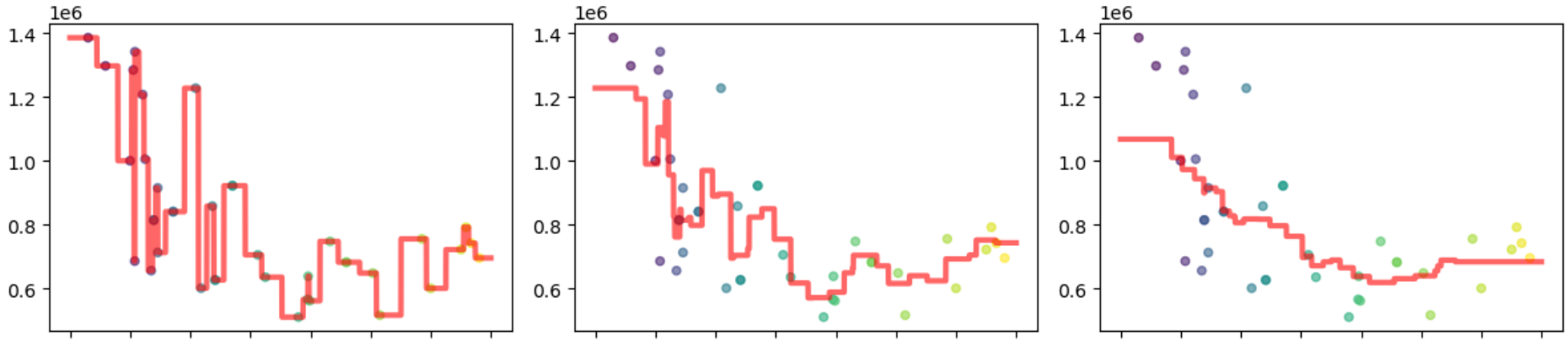
El valor de k, debe ser entero y mayor a uno. Aquí se describen sus efectos:

- Si k es un valor muy pequeño (1 o 2 vecinos) el modelo será muy sensible al ruido en los datos. Puede llevar a una predicción demasiado variable (overfitting).
- Si k es un valor muy grande (hasta tamaño del conjunto de datos), el modelo se volverá más suave y menos sensible al ruido. Sin embargo, puede perder detalles importantes en los datos (underfitting).

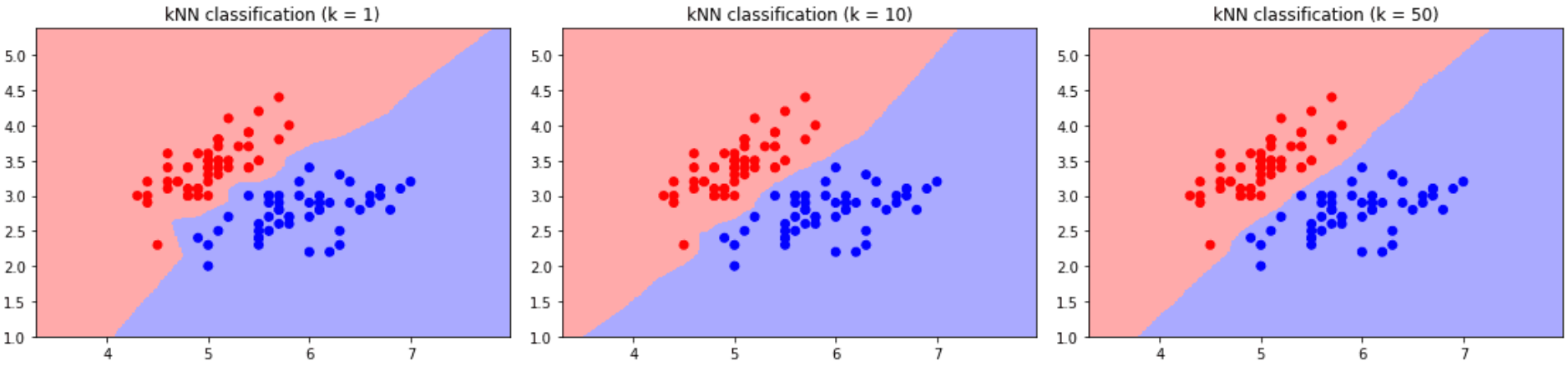
Respecto de la distancia, se debe escoger la forma en la que se medirá con relación a los k-vecinos tenidos en cuenta.

KNN - Hiperparámetros: Efecto de k

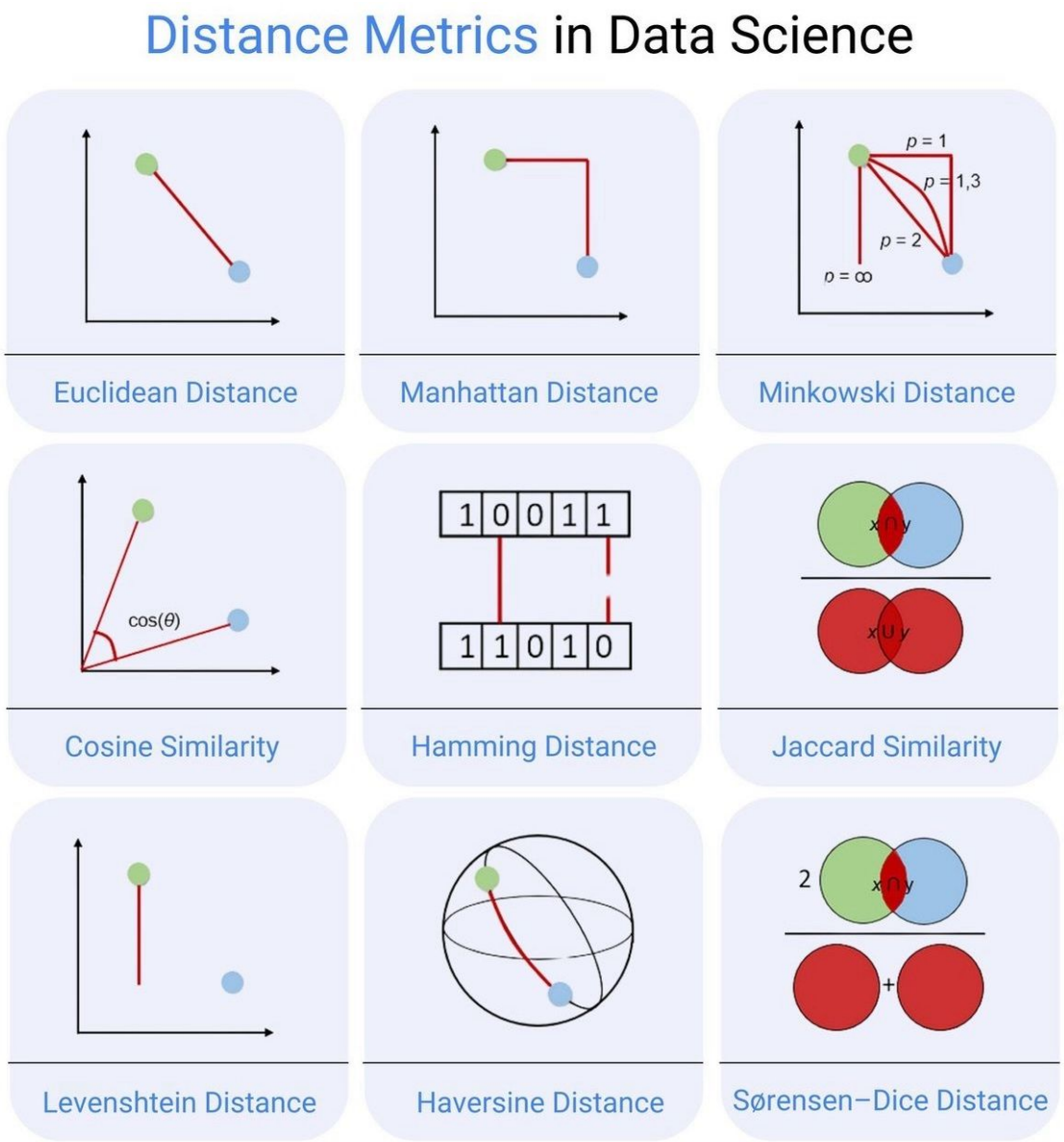
En Regresión:



En Clasificación:



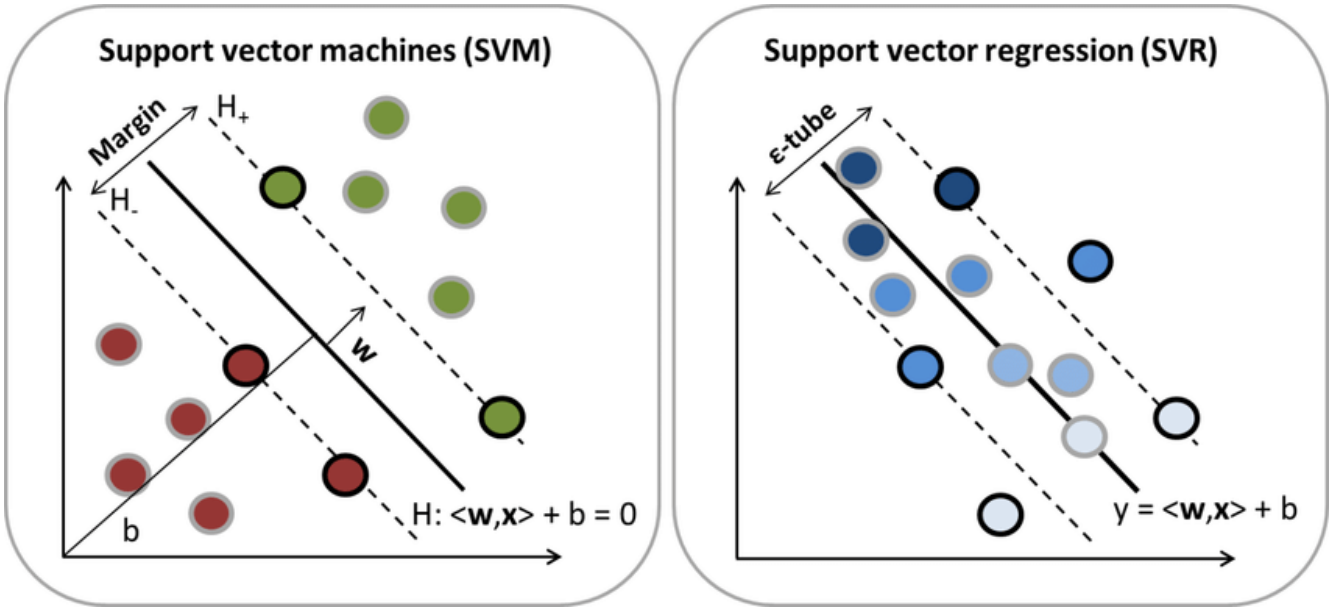
KNN - Hiperparámetros: Distancia



Support Vector Machines (SVM)

Definición

Algoritmos lineales* utilizados tanto para la regresión como para clasificación binaria (adaptable a multiclase) *basados en gradiente*. En esencia, las SVM buscan encontrar el hiperplano óptimo que mejor separa los datos en diferentes clases (clasificación) o que mejor se ajuste a los datos para predecir una variable continua (regresión).



Support Vector Machines (SVM)

Consideraciones

- La *elección del kernel* es fundamental en SVM, ya que determina cómo se mapean (transforman) los datos a un espacio de características de mayor dimensionalidad o no linealidad.
- La elección de características relevantes es crucial en SVM. Las características irrelevantes o redundantes (colineales) afectan negativamente su rendimiento. Sin embargo, no es un algoritmo que se destaque por la *interpretabilidad*.
- Los SVM son *sensibles a la escala de las características*, por lo que es común aplicar la estandarización o normalización.
- No es adecuado para **conjuntos de datos demasiado grandes**, dado que el tiempo y costo computacional para su entrenamiento es muy alto $O(n^2m + n^3)$.

Agenda

SVM - Clasificación

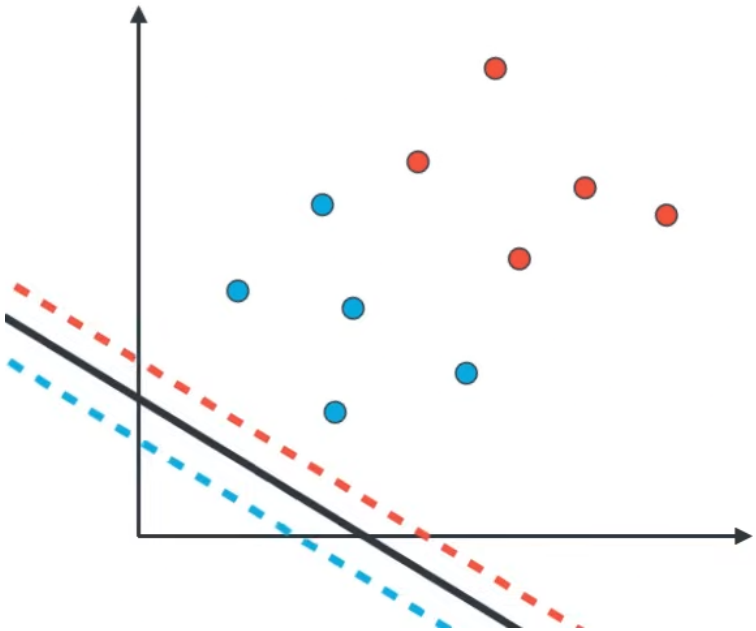
Funcionamiento

Este algoritmo esta basado en gradiente. No obstante presenta diferencias importantes. Este proceso se detalla a continuación:

Etapa 1: *Inicializar los parámetros* del hiperplano (pendientes e intercepto) de forma aleatoria. Junto a ellas, plantear dos lineas paralelas, una encima y otra debajo.

En caso de que el problema a resolver sea **multiclase**, cada hiperplano de separación incluirá **2 hiperplanos paralelos adicionales**.

Al mismo tiempo, se determina la **cantidad de iteraciones**, y las **tasas de aprendizaje y expansión**.



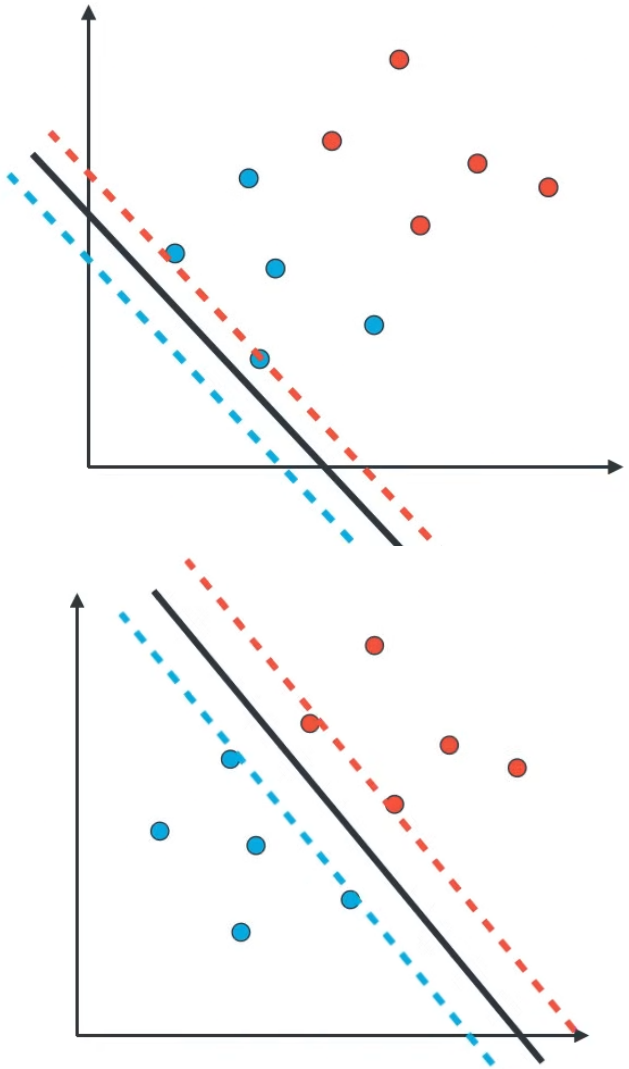
SVM - Clasificación

Etapa 2: Escoger una muestra de forma aleatoria:

- Si esta bien clasificada no hacer nada.
- Si esta mal clasificada **actualizar las pendientes e interceptos** según se requiera.

Posterior a la actualización, se *separan las lineas paralelas* multiplicando cada parámetro con la tasa de expansión.

Esta ultima parte de separación se repetirá hasta que estas lineas pasen por encima de los puntos mas cercanos de cada clase (convergencia).



SVM - Clasificación

Función de Costo

La función de costo típica en SVC es la función de pérdida bisagra o *hinge loss*. Incluye dos componentes: la **penalización por el error de clasificación** y la **regularización del margen**.

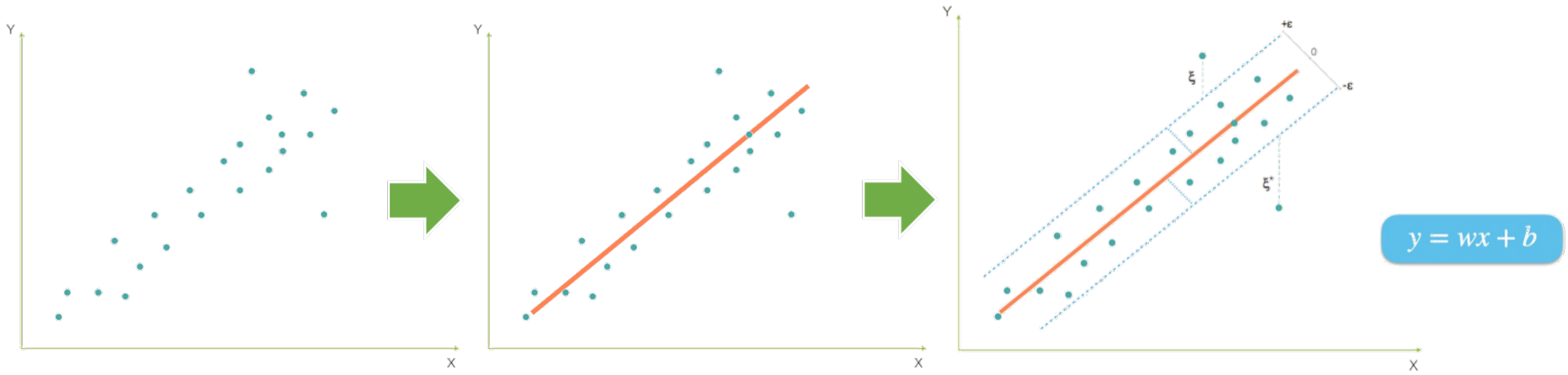
$$L(\mathbf{y}, \mathbf{f}(\mathbf{x})) = \sum_{i=1}^{N} [\text{máx}(0, 1 - y_i \cdot f(\mathbf{x}_i))] + \frac{1}{2C} \|\mathbf{w}\|^2$$

Donde:

- y es el vector de etiquetas reales de los puntos de datos (+1 o -1 en el caso de una clasificación binaria).
- $f(x)$ es el vector de salidas del modelo SVM para los puntos de datos.
- C es el parámetro de regularización.
- $\|\mathbf{w}\|$ es la norma del vector de pesos que define el hiperplano de separación.
- N es el número total de puntos de datos.

Agenda

SVM - Regresión



Funcionamiento

- Obtener un hiperplano que mejor represente el comportamiento de los datos
- Construir bandas paralelas (vectores de apoyo o de soporte) al hiperplano que cubra la mayor cantidad de datos.
- Los datos fuera de estas bandas serán los errores y los que se deben considerar para la fórmula del algoritmo.

SVM - Regresión

Función de Costo

Encontrar los parámetros ideales en este algoritmo, parte desde un problema de optimización con 2 propósitos: maximizar el ancho entre vectores de soporte (margen) y minimizar el error.

$$\underbrace{\min_{w,w_0,\xi,\xi^*} \frac{1}{2}w^2}_{Max.\ Margen} + \underbrace{C\sum_{i=1}^{N}(\xi+\xi^*)}_{Min.\ error}$$

Donde:

- w es el margen (ancho entre vectores de soporte),
- C es el **equilibrio** entre regularización y robustez (hiperparámetro), mayor a cero.
- ξ y ξ^* controlan el error al aproximar el margen.

Agenda

SVM - Hiperparámetros

Consideraciones para Selección

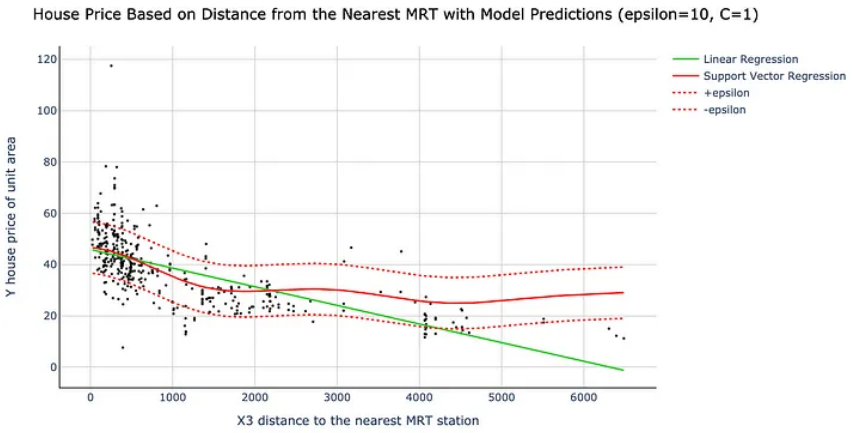
Las maquinas de soporte vectorial en clasificación, así como en regresión presentan hiperparámetros principalmente relacionados con su regularización, la transformación de datos no lineales entrantes y la mitigación de eventos con datos no balanceados. Aquí se describen:

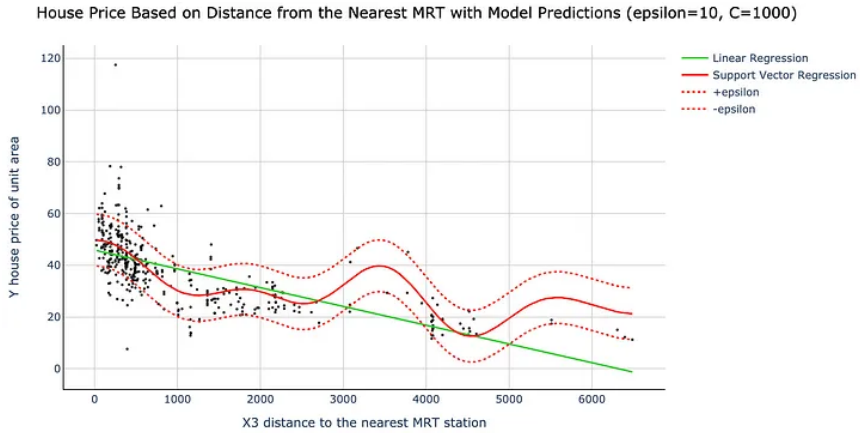
- **Regularización** (C): Parámetro que busca reducir el overfitting asociado a este algoritmo. Es el hiperparámetro mas importante de este método.

  El efecto del hiperparámetro C en este caso es el siguiente:
  - Si C tiende a infinito, se da prioridad a minimizar el error (tolerancia cero) y el margen en este caso será muy delgado.
  - Si C tiende a cero, se da prioridad a maximizar el margen, de modo que la tolerancia al error sera mayor.

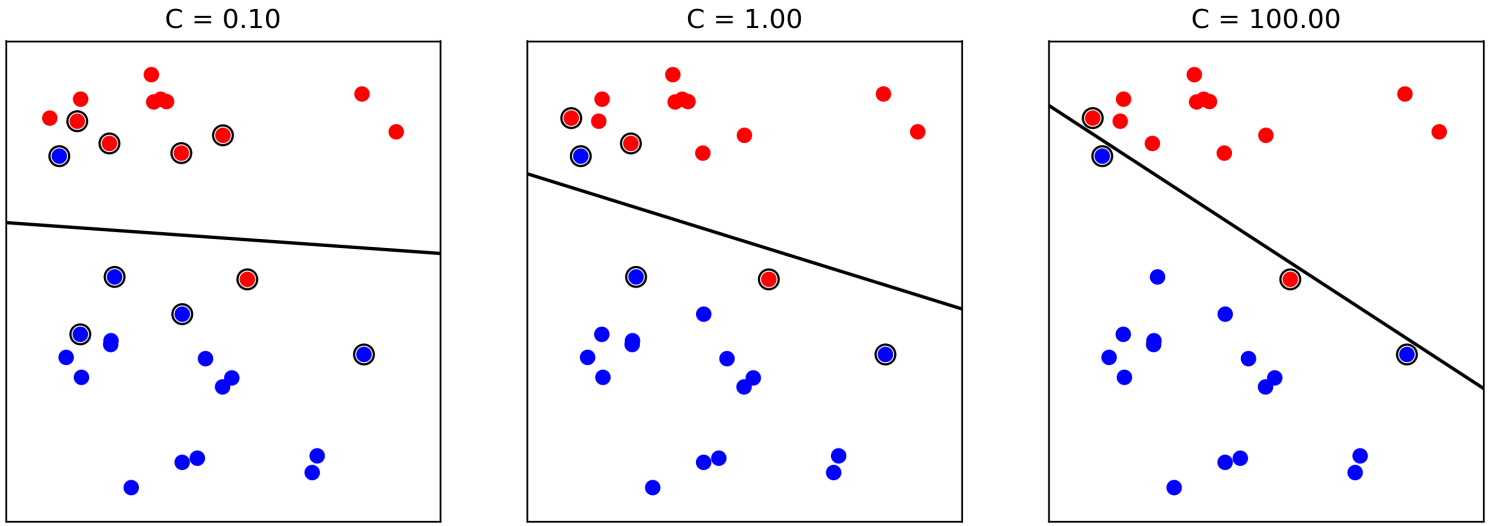
SVM - Hiperparámetros: Efecto de C

En Regresión:



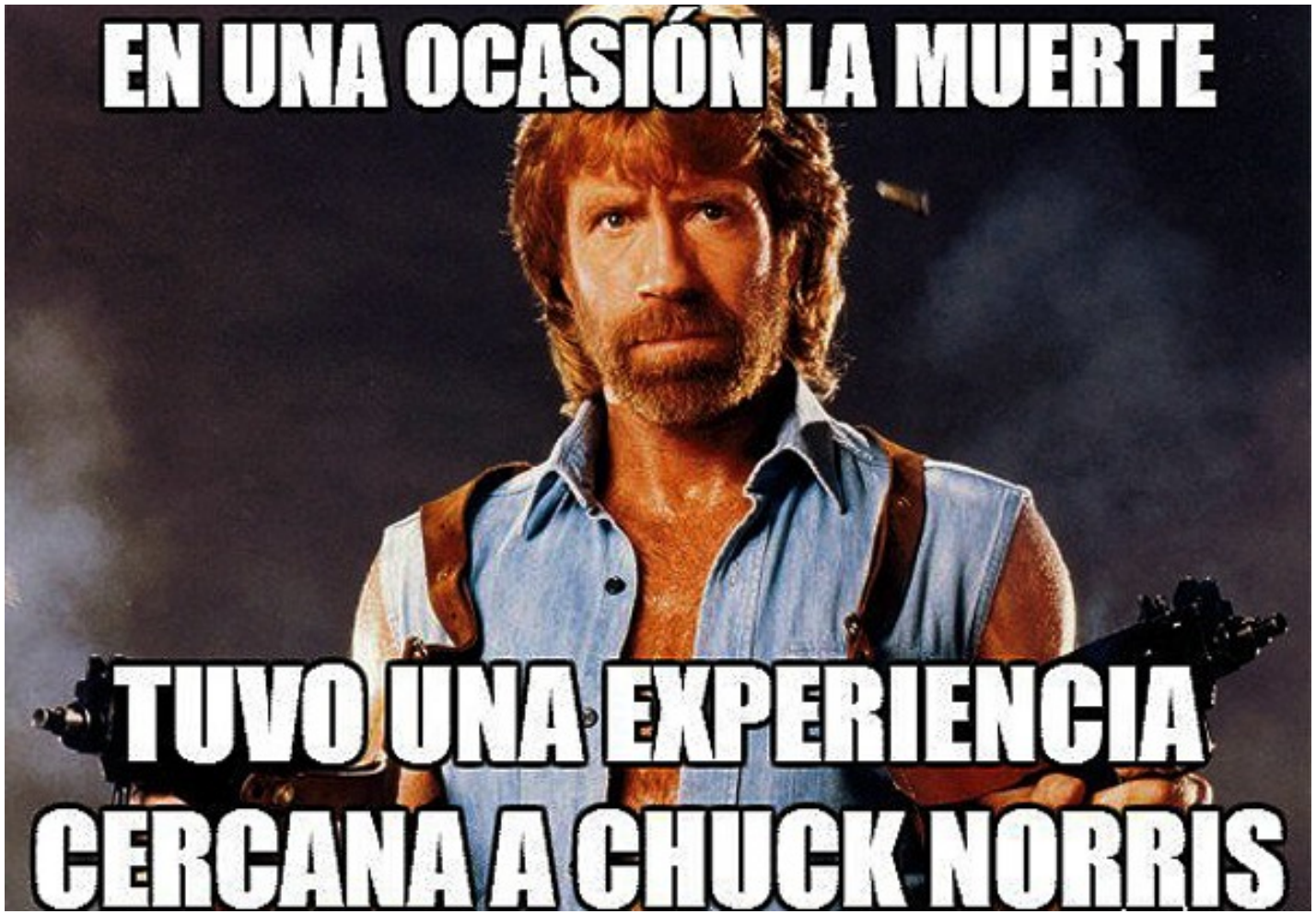


En Clasificación:



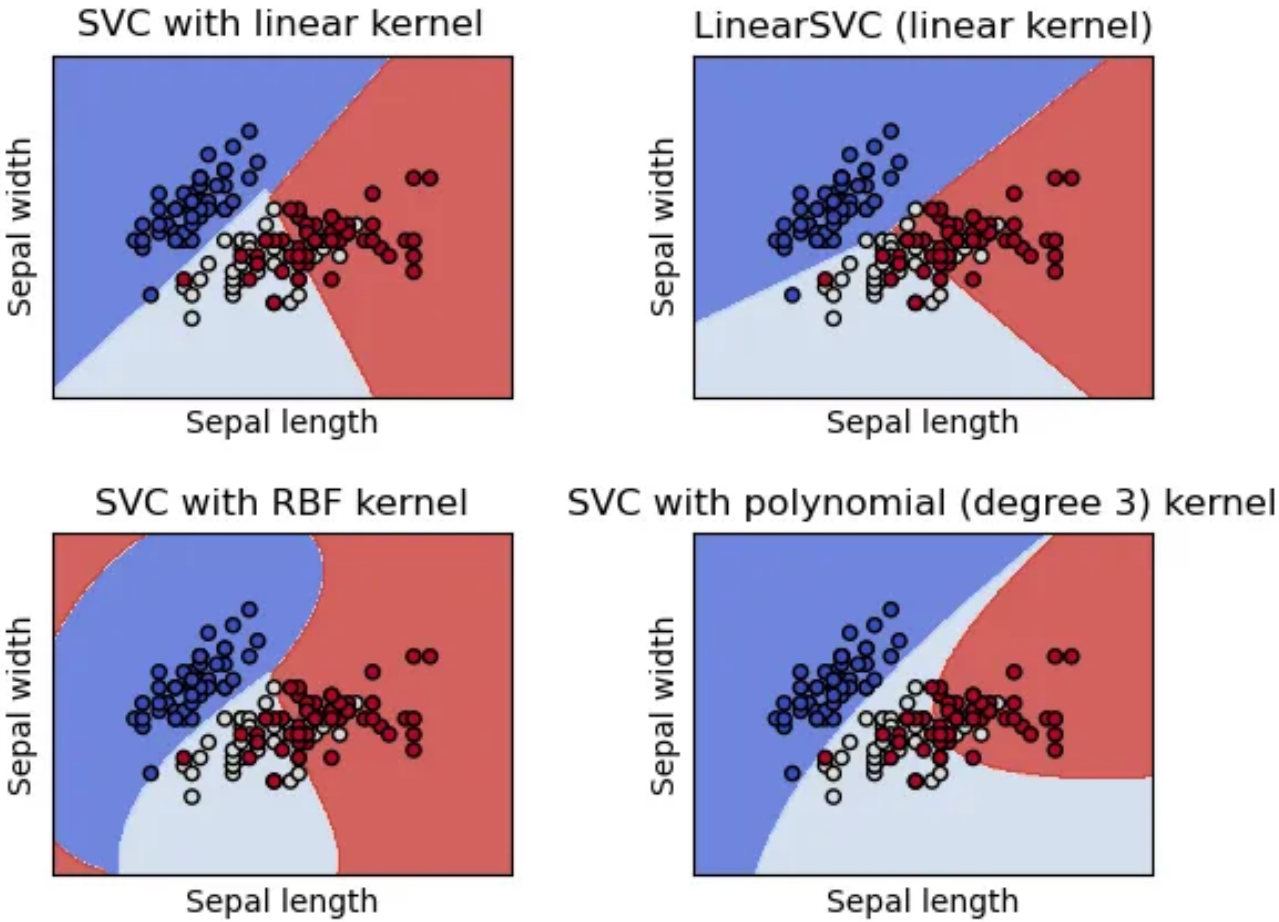
SVM - Hiperparámetros: Efecto del Kernel

¿Y si los datos son no lineales?!!



SVM - Hiperparámetros: Efecto del Kernel

- **Transformación de datos** $(kernel)$: Se encarga de transformar los datos en espacios dimensionales superiores y diferentes para trabajar en entornos lineales.



Referencias

https://scikit-learn.org/stable/index.html

https://towardsdatascience.com/getting-acquainted-with-k-nearest-neighbors-ba0a9ecf354f

https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0

https://medium.com/@csarchiquerodriguez/maquina-de-soporte-vectorial-svm-92e9f1b1b1ac