

ANÁLISIS Y APRENDIZAJE AUTOMÁTICO



IBERIA EXPRESS

Yvonne Gala García y Jesús Prada Alonso



Yvonne Gala García
**Responsable Advanced
Analytics en Iberia Express**

Estudios:

- Doctorado en Aprendizaje Automático en la Universidad Autónoma de Madrid (UAM).
- Máster universitario en Bioinformática y bioestadística, UOC.
- Máster Universitario en Investigación e Innovación en TIC, especialidad inteligencia computacional, UAM.
- Máster Educación Secundaria Obligatoria, especialidad matemáticas, UAM.
- Licenciatura en Matemáticas, UAM.

Experiencia Laboral:

- Gerente Marketing Digital en Iberia Express.
- Responsable Advanced Analytics en Iberia Express.
- Freelance Data Scientist: Iberia Express, Piperlab.
- Profesora en EDEM y DevAcademy.
- Investigadora en el Grupo de Aprendizaje Automático de la UAM.



Jesús Prada Alonso
Responsable Area Machine Learning en Sigesa y Data Scientist en Iberia Express

Estudios:

- Doctorado en Aprendizaje Automático en la Universidad Autónoma de Madrid (UAM).
- Máster universitario en Bioinformática y bioestadística, UOC.
- Doble Máster Universitario en Investigación e Innovación en TIC y en Matemáticas y Aplicaciones, UAM.
- Doble Titulación en Ingeniería Informática y Licenciatura en Matemática, UAM.

Experiencia Laboral:

- Responsable Area Machine Learning en Sigesa.
- Senior Data Scientist en Iberia Express
- Freelance Data Scientist: M+ Vision Consortium, Hospital Clínico San Carlos, UMOT, etc.
- Profesor en EDEM e IE School of Human Sciences and Technology, HST.
- Data Scientist en Kernel Analytics.
- Investigador en el Grupo de Aprendizaje Automático de la UAM.

ÍNDICE

EDEM

Escuela de Empresarios



BLOQUE I (01/04/2022): Introducción ML. ML en el sector turístico

TEORÍA

1. Introducción ML.

- ¿Qué es Machine Learning?
- Estructura proyecto ML.
- Casos prácticos sector turístico.

2. Aprendizaje Supervisado Vs No supervisado.

- Supervisado: Clasificación vs Regresión.
- No supervisado: Clustering, Reducción de dimensionalidad

3. ML Supervisado. Conceptos básicos ML.

- Conjuntos Train/Test/Validación. Cross validation.
- Métricas.
- Metaparametrización.
- Trade off bias/variance. Overfitting/underfitting.

4. Sistema de Recomendación

BLOQUE I (01/04/2022): Introducción ML. ML en el sector turístico

PRÁCTICA

Datasets:

- Dataset scikit-learn.
- Ejemplo sistema de recomendación pequeño en sector turístico.
- Ejemplo sistema de recomendación películas.

Ejercicios:

- Introducción a scikit-learn.
- Split data set, evaluación, overfitting...
- Sistema de recomendación de vuelos.
- Sistema de recomendación de películas.

BLOQUE II (02/04/2022): Modelos ML I, regresión lineal, logística y SVM. Clasificación y Regresión.

TEORÍA

1. Regresión lineal
 - Conceptos.
 - Regularización. Lasso / Ridge.
2. Regresión logística
 - Conceptos.
 - Regularización. Lasso / Ridge.
3. SVM
 - Conceptos.
 - Clasificación.
 - Regresión.

BLOQUE II (02/04/2022): Modelos ML I, regresión lineal, logística y SVM. Clasificación y Regresión

PRÁCTICA:

Datasets:

- Dataset scikit-learn.
- Datos médicos.
- Datos aerolínea (Iberia Express).

Ejercicios:

- Script de regresión lineal, logística, SVM.
- Problema de clasificación en medicina.
- Problema de regresión en una aerolínea. Predicción de demanda.

BLOQUE III (09/04/2022): Modelos ML II, árboles de decisión, random forest, xgboost. Clasificación y Regresión.

TEORÍA

1. Árboles de decisión.
 - Conceptos.
 - Representación gráfica.
 - Hiperparámetros.
2. Random Forest.
 - Conceptos.
 - Ensembles de árboles.
 - Hiperparámetros.
3. XGBoost.
 - Conceptos.
 - Intuición Random Forest vs XGBoost.
 - Hiperparámetros.
 - Comparación. Interpretabilidad vs Accuracy

BLOQUE III (09/04/2021): Modelos ML II, árboles de decisión, random forest, xgboost. Clasificación y Regresión.

PRÁCTICA:

Datasets:

- Dataset scikit-learn.
- Datos médicos.
- Datos aerolínea (Iberia Express).

Ejercicios:

- Script de árboles de decisión, random forest, xgboost.
- Problema de clasificación en medicina. Comparar los 3 modelos y optimización de parámetros.
- Problema de regresión en una aerolínea. Predicción de demanda. Comparar los 3 modelos y optimización de parámetros.
- Comparación final de métricas de todos los modelos vistos.

[illegible]

INTRODUCCIÓN ML





Machine Learning

Introduction

What is machine learning



Machine Learning

Introduction

What is machine learning



Machine Learning

Introduction

What is machine learning



Machine Learning

Introduction

What is machine learning

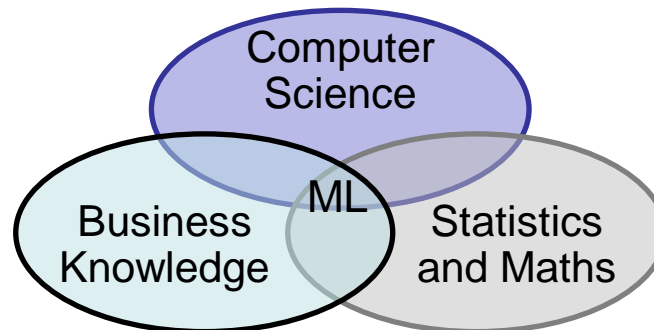
¿QUÉ ES DATA SCIENCE?

¿Qué es Data Science y Machine Learning?

Data Science es un conjunto de herramientas, técnicas y disciplinas que se enfocan en convertir grandes cantidades de datos en información útil para explicar la relación entre variables y generar modelos predictivos.

Machine Learning, ML, o **Aprendizaje Automático** es una rama de la Inteligencia Artificial cuyo objetivo es construir sistemas que aprendan automáticamente de los datos.

A machine learns with respect to a particular task T , performance metric P , and type of experience E , if the system reliably improves its performance P at task T , following experience E .



Paso previo - Plantear un problema o hipótesis a demostrar.

Paso 1 - Recolección de datos.

Paso 2 - Preproceso.

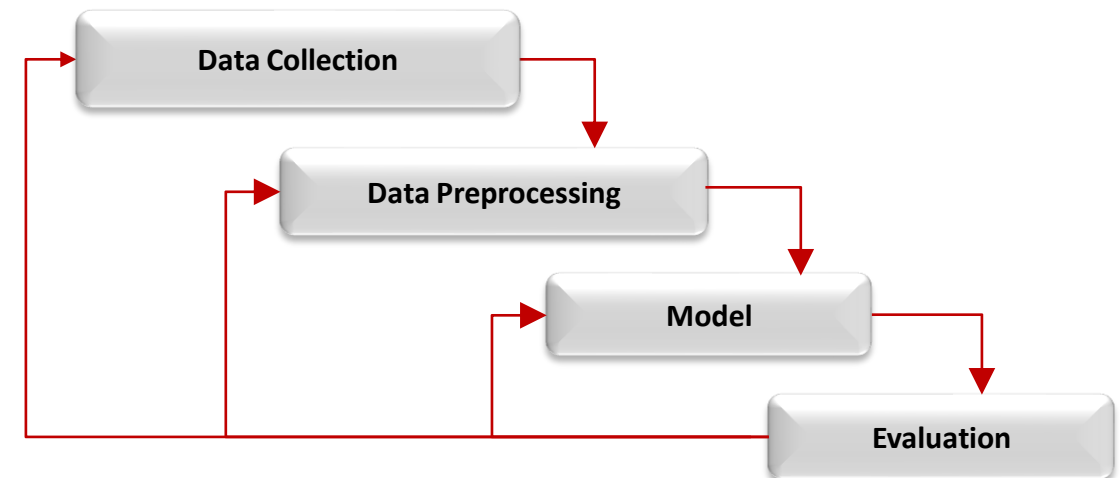
Paso 3 - Modelización.

Paso 4 - Validación.

Paso 5 - Conclusiones/informes/visualización.

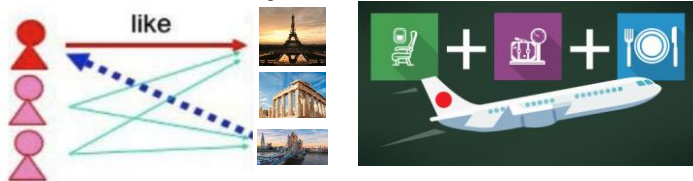
Paso 6 - Puesta en producción.

Paso posterior - Mantenimiento.



Proyectos

Recomendación Rutas y Ancillaries



Detección de fraude web



Chatbot



Optimización de precios Billetes, Maletas y Asientos



Estimaciones Pasajeros, Maletas y No show



Sistema de recomendación

Elección de la ruta a recomendar a cada cliente en función de sus interacciones con Iberia Express.

Sirve para personalizar:

- Campañas de emails
- Página web:
 - Home
 - Gestión de Reserva
 - Check in



Optimización de precios

Elección del precio óptimo en billetes vendidos por la web.

Basado en:

- Características del vuelo
- Histórico de llenado del vuelo
- Momento de la compra
- Disponibilidad de asientos
- Precio de la competencia
- Búsquedas web Iberia express
- Búsquedas competencia



Overbooking

Predicción de overbooking en los aviones. Es decir, billetes vendidos por encima de la capacidad del vuelo.

Con este proyecto se optimiza el número de asientos vendidos, maximizando el revenue por vuelo y minimizando el impacto negativo sobre el cliente.



Fraude (I). Descripción

Detección de compras fraudulentas en la web.

Basado en:

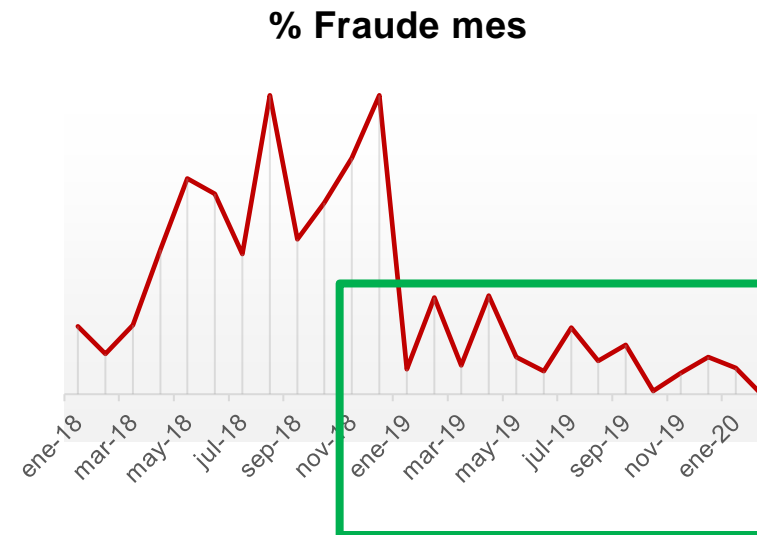
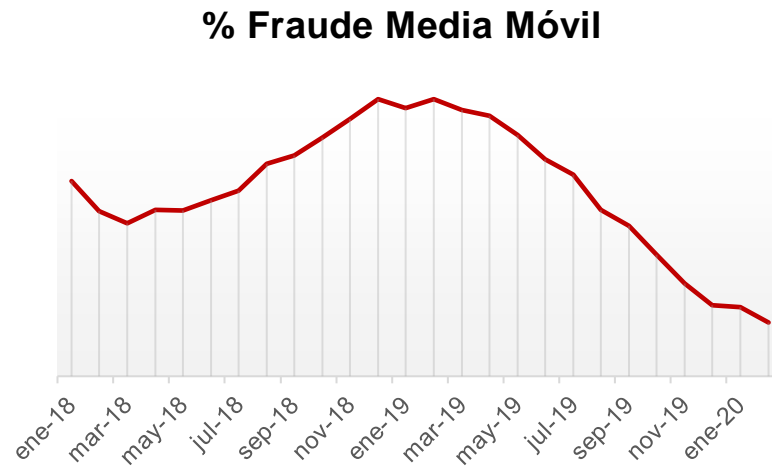
- Características del vuelo
- Características de la compra
- Características del comprador
- Histórico de fraude anterior
- Momento de la compra

Este proyecto ha perdido importancia en los dos últimos años tras la entrada en vigor de la psd2.



Fraude (II). Resultados

El modelo se puso en producción el 2019-01-01



El fraude se redujo 10 veces de noviembre de 2018 a enero de 2019

Bajada de maletas

Predicción de bajada de maletas en el embarque del avión en función del llenado y características del vuelo.

Con este proyecto se avisa a los usuarios para facturar su maleta en el mostrador de facturación evitando retrasos y haciendo mejor la experiencia del usuario.



NLP: Procesamiento de Lenguaje Natural

Crear mediante NLP, un asistente virtual que permita responder a las preguntas de los usuarios de la web de Iberia Express automáticamente.

NLP se basa en algoritmos con mecanismos de atención que simulan el comportamiento del cerebro humano y entendimiento del lenguaje.

Muchas mas precision que modelos que buscan palabras claves.

Mejora la experiencia de usuario de los clientes con un servicio “inteligente” 24x7 y automático.



Analítica pre y post COVID

Las aerolíneas es uno de los sectores económicos más afectados por esta crisis sanitaria.

Esto tiene una afección directa en los datos.

- Han **cambiado los patrones de comportamiento** de los usuarios. Tanto de compra (ej: recomendaciones), como de vuelo (precios dinámicos).
- Han **cambiado las necesidades de negocio** → muchos productos no se pueden ofertar y la empresa tiene otros objetivos.
- **No tenemos un histórico** de los nuevos comportamientos de los clientes **en el entorno actual**.

Soluciones

- Adaptar los proyectos según necesidades.
- Adaptar los datos, quitando información pre covid o dándole menos importancia.

Estimación de reservas COVID

- **Objetivo:** abrir rutas que puedan tener volumen de pasajeros en los próximos meses según la evolución del COVID.
- **Datos:**
 - Incluimos solo información del periodo covid.
 - Información de la evolución de la pandemia, número de casos y vacunas.
- **Resultados:** Estimación de reservas a dos o tres meses.



Estimación no show de pasajeros COVID

- **Objetivo:** optimizar el llenado del avión cumpliendo medidas sanitarias.
- **Datos:**
 - Incluimos solo información del period covid y variables que puedan aumentar la probabilidad de no show.
- **Resultados:** predicciones de reservas y no show.

SUPERVISADO VS NO SUPERVISADO

EDEM

Escuela de Empresarios



- Conceptos.
- Aprendizaje supervisado: Clasificación vs Regresión.
- Aprendizaje no supervisado: Clustering, reducción de dimensionalidad.

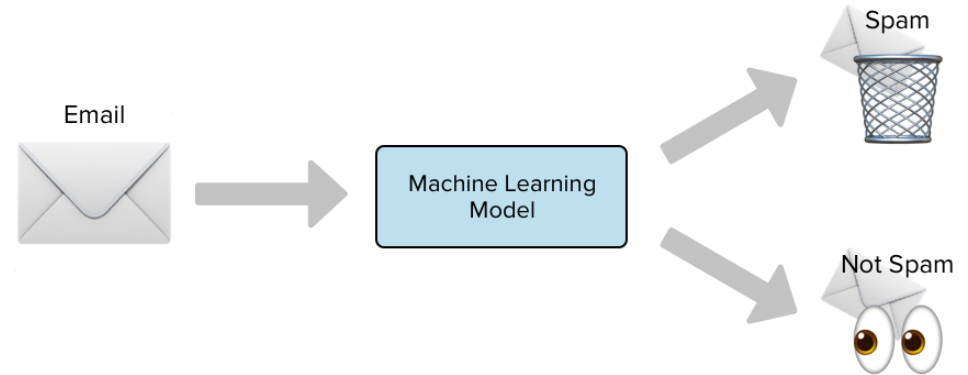
Aprendizaje supervisado VS no supervisado (I)

Aprendizaje supervisado:

Para entrenar el modelo se utiliza un dataset o conjunto de **muestras etiquetado (train)**. El objetivo es **predecir** la etiqueta que tendrán futuras muestras (test) que el modelo no ha visto en su entrenamiento.

Ejemplo:

- Clasificar si un correo es spam o no.
- Predecir la producción de energía solar producida en una planta.



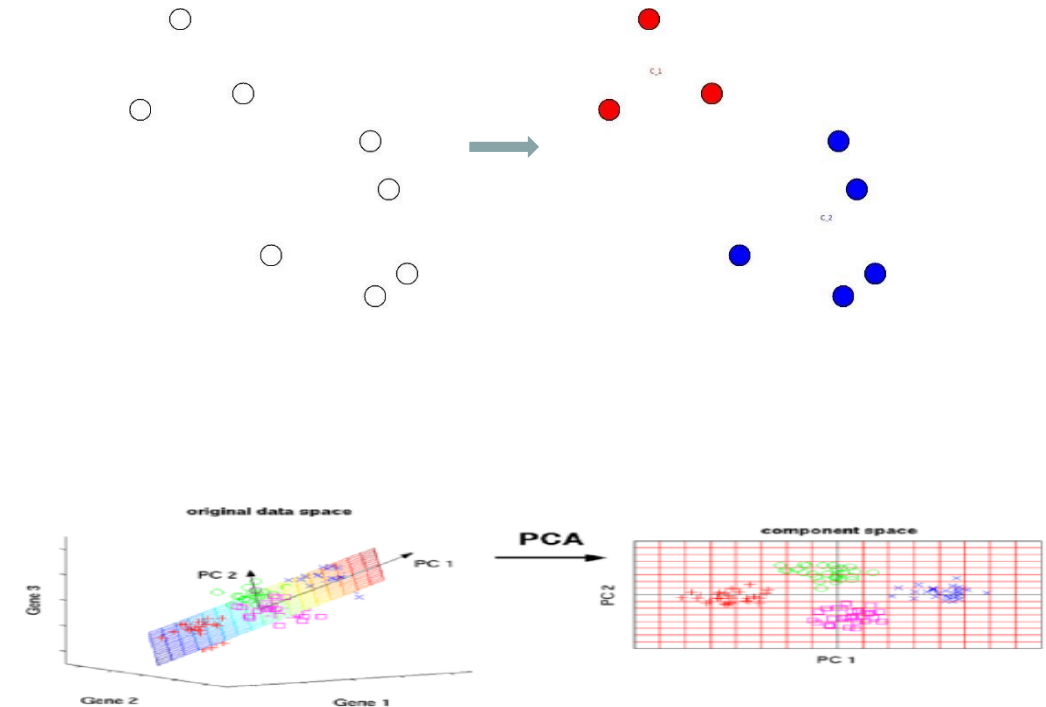
Aprendizaje supervisado VS no supervisado (II)

Aprendizaje no supervisado:

Para entrenar el modelo se utiliza un dataset o conjunto de **muestras sin etiquetar**. El objetivo es encontrar **patrones** en los datos para extraer conocimiento útil.

Ejemplo:

- Segmentar tus usuarios en 2 grupos.
- Reducción a 2 dimensiones.



Aprendizaje Supervisado vs No Supervisado

| | Supervisado | No supervisado |
|------------------|-----------------------------------------------------|-------------------------------------------------------|
| Etiquetas | SI | NO |
| Objetivo | Dar predicciones a futuro sobre el conjunto de test | Encontrar patrones en los datos o reducir dimensiones |
| Modelos | Regresión lineal, árboles, SVM, Redes Neuronales | Clustering, PCA |
| Ejemplo | Predecir si una transacción es fraudulenta | Encontrar clientes con perfiles similares |

Clasificación

Las etiquetas son categóricas, indicando la pertenencia de una determinada muestra a una clase en particular.

Ejemplo:

TRAIN



----> 1



----> 0



----> 1



----> 1

TEST



----> 1 ✓



----> 0 ✓



----> 1 ✗ ¿Overfitting?

Regresión

Las etiquetas son numéricas, indicando un valor asociado a cada muestra.

Ejemplo:

TRAIN

target

| Superficie | Antigüedad | Ciudad | Precio |
|------------|------------|---------|--------|
| 50 | 1 | Madrid | 1000€ |
| 50 | 1 | Algete | 600€ |
| ... | ... | ... | ... |
| 100 | 10 | Sevilla | 650 |

TEST

target

| Superficie | Antigüedad | Ciudad | Precio |
|------------|------------|----------|--------|
| 30 | 15 | Madrid | ? |
| 230 | 5 | Galicia | ? |
| ... | ... | ... | ... |
| 80 | 1 | Canarias | ? |

Clasificación. Matriz de confusion

Nos dará un conteo de los aciertos y errores de cada una de las clases por las que estemos clasificando.

| | | Clasificador | |
|------------|---|--------------|----|
| | | + | - |
| Valor real | + | TP | FN |
| | - | FP | TN |

- **TP – True Positives:** Es el número verdaderos positivos, es decir, de predicciones correctas para la clase +.
- **FN – False Negatives:** Es el número de falsos negativos, es decir, la predicción es negativa cuando realmente el valor tendría que ser positivo.
- **FP – False Positives:** Es el número de falsos positivos, es decir, la predicción es positiva cuando realmente el valor tendría que ser negativo
- **TN – True Negatives:** Es el número de verdaderos negativos, es decir, de predicciones correctas para la clase -.

Clasificación

- **Accuracy:** Porcentaje de aciertos del modelo.
- **Sensibilidad, o recall, VPR:** ratio de verdaderos positivos.
- **Especificidad, VNR:** ratio de verdaderos negativos.
- **Precisión:** probabilidad de que, dada una predicción positiva, la realidad sea positiva también.
- **F1-score:** f1-score es una medida que mezcla la precision y el recall. Mide si nuestro modelo tiene falsos positivos y falsos negativos a la vez.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

$$Sensibilidad = \frac{TP}{TP+FN}$$

$$Especificidad = \frac{TN}{TN+FP}$$

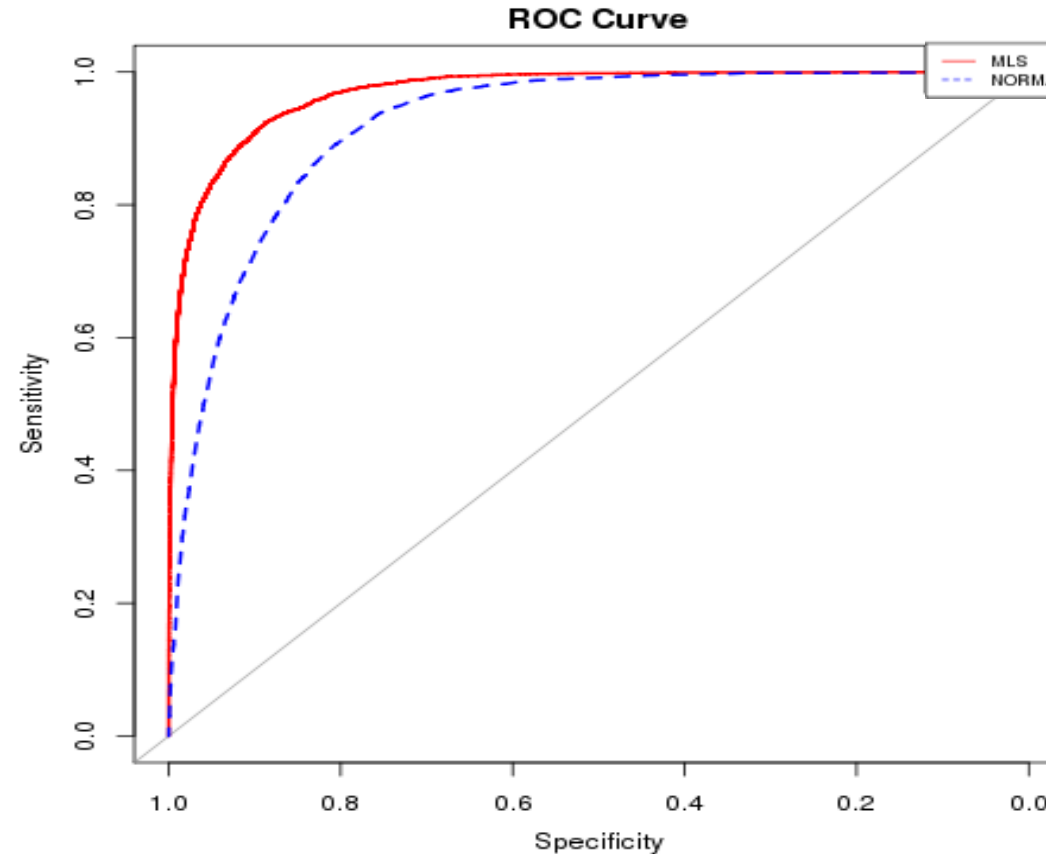
$$Precision = \frac{TP}{TP+FP}$$

$$F1 = 2 * \frac{precision*sensibilidad}{precision+sensibilidad}$$

Clasificación. AUC

$$\text{Sensibilidad} = \frac{TP}{TP + FN}$$

Sensibilidad, recall, VPR:
ratio de verdaderos positivos.

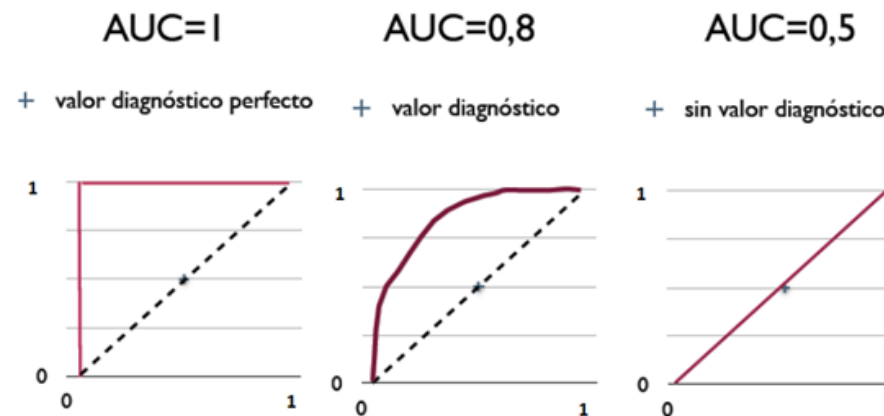


Especificidad, VNR: ratio de verdaderos negativos.

$$\text{Especificidad} = \frac{TN}{TN + FP}$$

Clasificación. AUC

- La curva ROC se define por FPR (Ratio Falsos Positivos) y VPR (Ratio True Positive) como ejes x e y respectivamente.
- Representa los intercambios entre verdaderos positivos (beneficios) y falsos positivos (costes).
- Cada valor umbral usado como punto de corte para distinguir entre qué es una predicción positiva y qué una negativa representa un punto en el espacio ROC.



Regresión

- **MAE o Error absoluto medio:** es la media de la diferencia absoluta entre los puntos de datos reales y el valor de predicción.

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

- **MSE o Error cuadrático medio:** es la media de la diferencia entre los puntos reales de datos y el valor de predicción al cuadrado. Penaliza más las diferencias mayores o extremas.

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- **RMSE:** Raíz cuadrada del MSE. Proporciona mayor intuición que el MSE.

$$\text{RMSE} = \sqrt{\text{MSE}}$$

- **MAPE o Error absoluto porcentual medio:** Permite medir error relativos a la magnitud del valor real.

$$\text{MAPE} = \frac{1}{N} \sum_{i=1}^N \frac{|y_i - \hat{y}_i|}{|y_i|}$$

Aprendizaje Supervisado. Clasificación vs Regresión

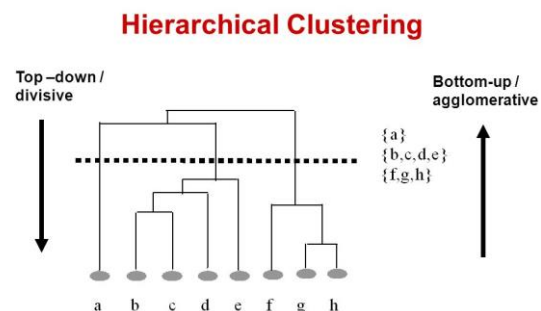
| | Clasificación | Regresión |
|------------------|--------------------------------------------|---------------------------------------------------------------|
| Etiquetas | Categóricas. | Numéricas. |
| Ejemplo | Predecir si un email es spam (1) o no (0). | Predecir el precio de alquiler de una casa (550,632,1057...). |
| Métrica | AUC | MSE |

Características

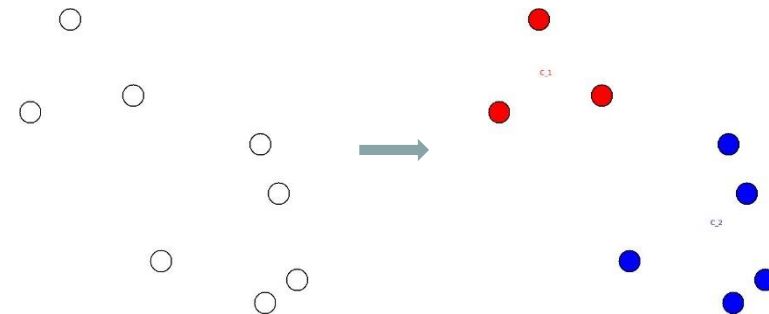
- **Objetivo:** Para entrenar el modelo se utiliza un dataset o conjunto de muestras sin etiquetar. El objetivo aquí no es predecir un target sino encontrar patrones en los datos para extraer conocimiento útil.
- **Tipos:** 2 clases principales de problemas:
 1. Clustering.
 2. Reducción de dimensionalidad.

Segmentación o clustering

- **Objetivo:** Clustering es la tarea de agrupar un conjunto de objetos tales que los objetos en el mismo grupo (clúster) son más similares entre sí que a los de otros grupos.
- Métodos de agrupación:
 - **Métodos jerárquicos:** se descomponen en forma de árbol el conjunto de datos.
 - **Métodos de partición:** se crean divisiones sucesivas del conjunto de datos.



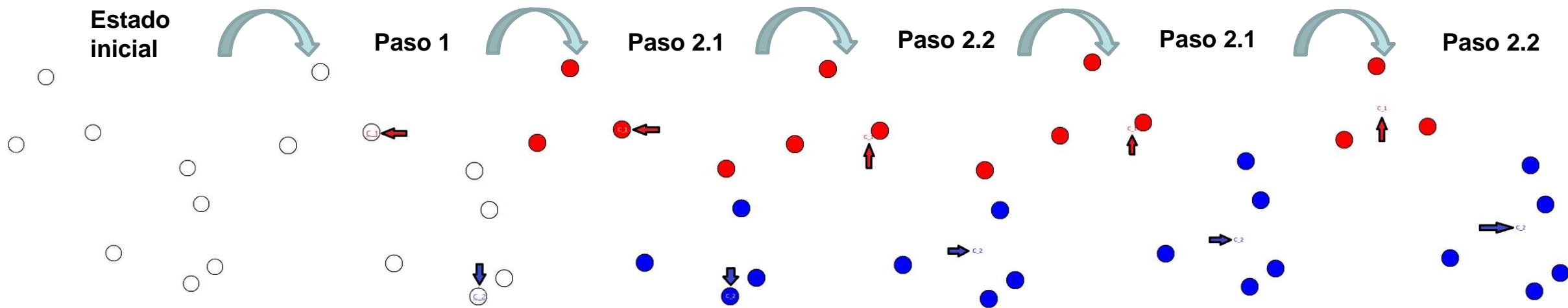
© 2007 Cios / Pedrycz / Swiniarski / Kurgan 21



Segmentación o clustering. K- means

Ejemplo: K-means.

1. Seleccionar aleatoriamente k instancias como centros iniciales de las particiones. También puede utilizarse k puntos aleatorios del espacio de búsqueda.
 2. Repetir
 1. Re/asignar instancias a la partición con el centro más próximo
 2. Recalcular el centro de cada partición (media) en función de los nuevas instancias asignadas.
- Mientras haya cambios en las particiones



Segmentación o clustering. K- prototypes

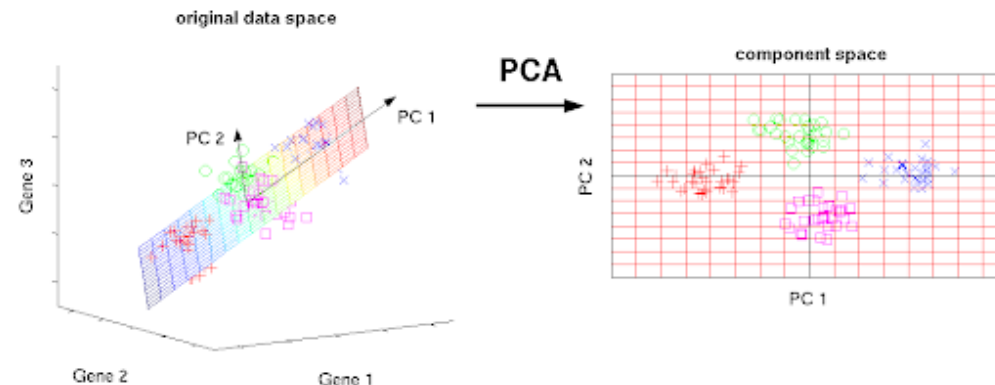
- K-means solo acepta datos numéricos ya que usa la distancia euclídea. Sin embargo, hay algoritmos como k-prototypes que extienden k-means para admitir una **combinación de variables numéricas y variables categóricas**.

$$E = E_{num} + \lambda E_{cat}$$

- El parámetro λ sirve para **equilibrar la importancia** dada a las variables numéricas vs las categóricas.

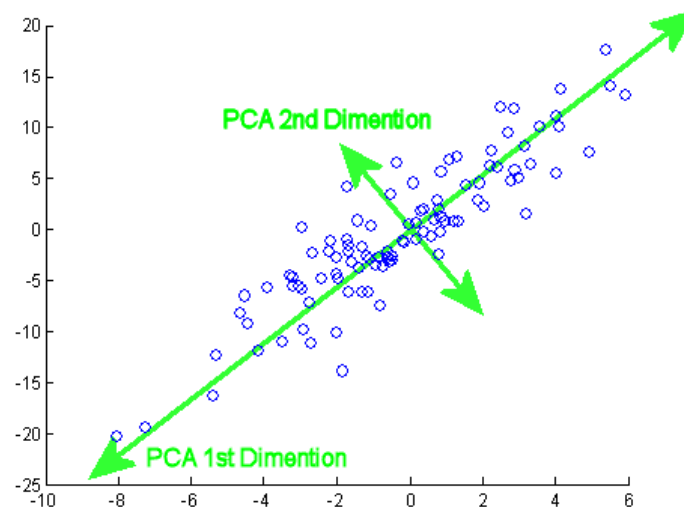
Reducción de dimensionalidad

- **Objetivo:** Reducir el número de variables o columnas en un dataset. Con esto se busca:
 - Disminuir coste computacional
 - Conseguir un nuevo dataset con menos variables irrelevantes o ruido.
 - Realizar visualizaciones
 - ...
- **Ejemplo: PCA**



Reducción de dimensionalidad. PCA

- Transforma las variables originales en un conjunto de nuevas variables, combinación de las anteriores, linealmente no correlacionadas.
- Estas variables reciben el nombre de **componentes principales**.
- Estas componentes son las que contienen la mayor información del dataset original.

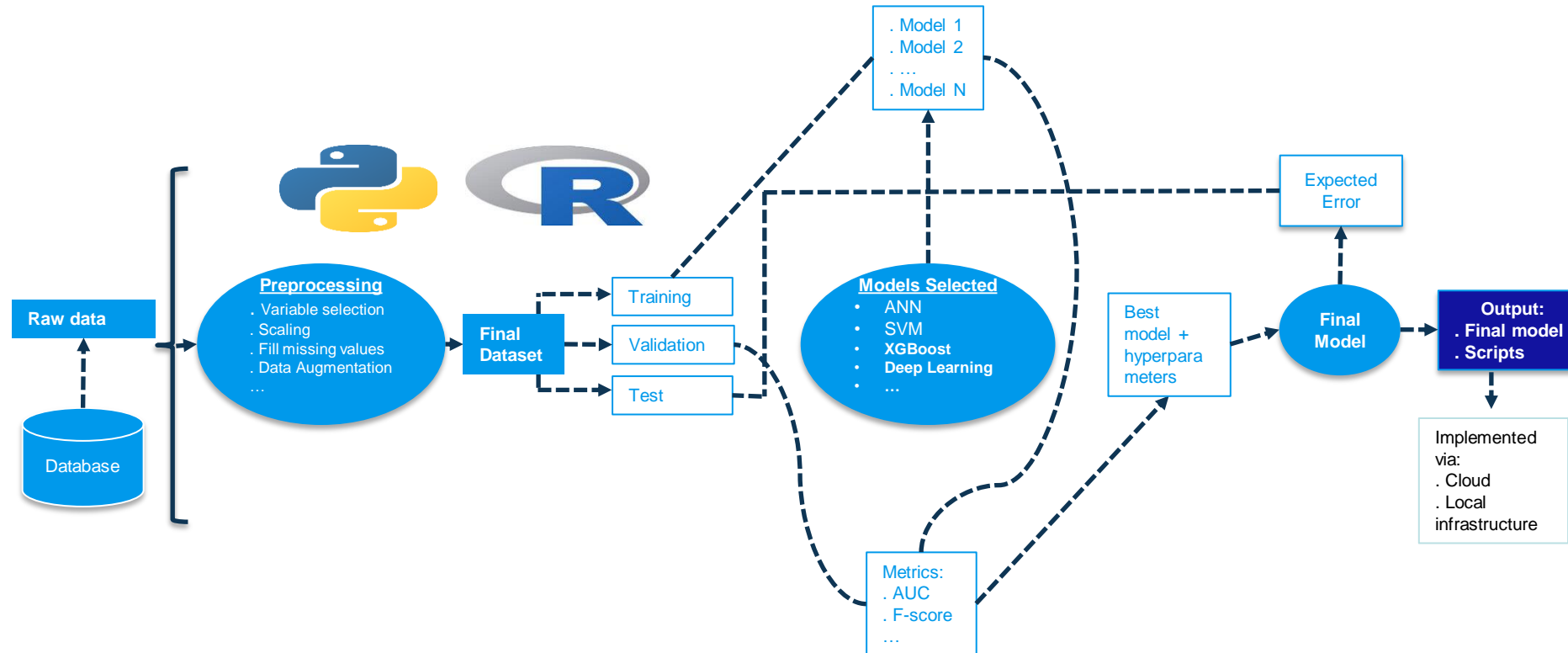


ML SUPERVISADO



- Conjuntos Train/Validación/Test. Cross-validation.
- Métricas.
- Metaparametrización.
- Trade off bias/variance. Overfitting/Underfitting.

ESQUEMA ML



Conjuntos Train/Validación/Test

Tipos de conjuntos

- **Muestra de Entrenamiento (TRAINING):** Datos de los que los modelos extraen patrones. Son los únicos para los que el modelo “ve” el target o etiqueta a predecir.
- **Muestra de Validación (VALIDATION):** Se emplea para seleccionar el mejor de los modelos entrenados cuando realizamos el ajuste de parámetros o metamodelización.
- **Muestra de Prueba (TEST):** Proporciona el error real esperado con el modelo seleccionado.

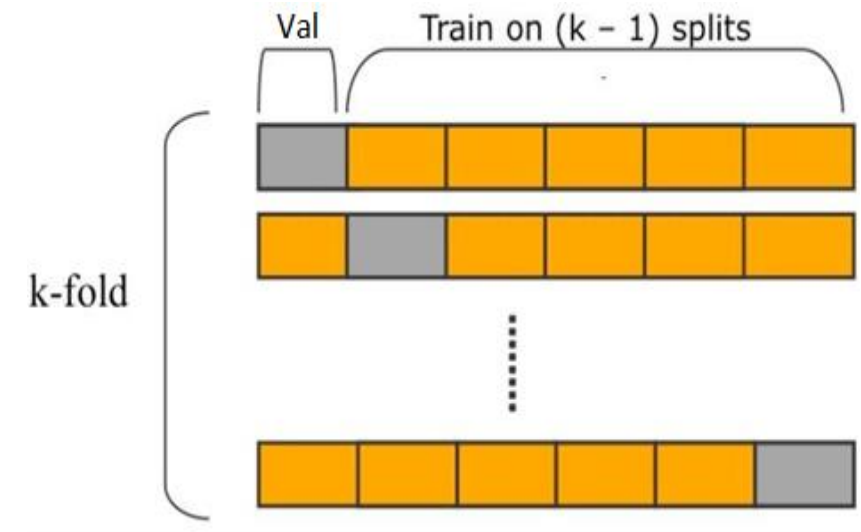
Consideraciones de los conjuntos de train, validación y test

- Que sean lo **suficientemente grandes** como para generar resultados significativos desde el **punto de vista estadístico**.
- Que sean **representativos** de todo el conjunto de datos. Es decir, no elegir un conjunto de prueba con características diferentes (**sesgo**) al del conjunto de entrenamiento.
- **No existe** una solución óptima (**golden rule**) para elegir el porcentaje del total de datos asignado a cada conjunto, ya que depende del problema. Pero un estándar típico es 70/15/15.

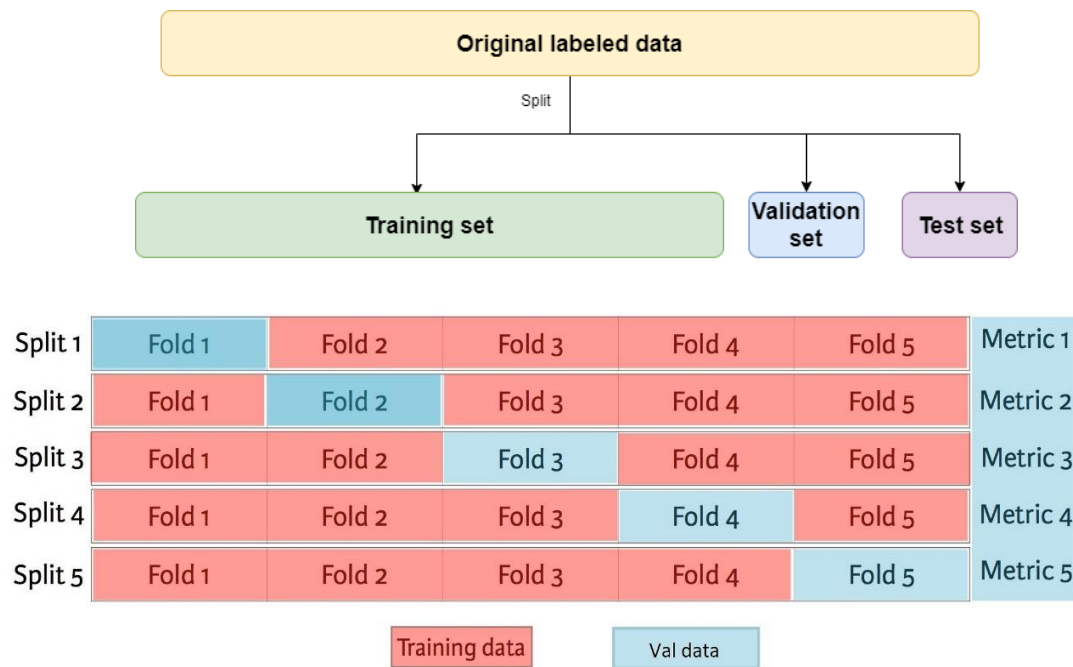
Cross-validation

Es un método alternativo a la división en train/val/test para realizar la optimización de hiperparámetros. Permite no tener que crear un conjunto de validación, sustituyendo su funcionalidad por la siguiente metodología:

- Se hace una separación del datatest en k subconjuntos del mismo tamaño.
- Se realiza el $k - 1$ conjuntos para entrenar y 1 para validación.
- Se repite el procedimiento k veces rotando el conjunto de validación.
- Se evalúa con la métrica seleccionada.



Validación fija VS. Cross-validation



Permite aprovechar más volumen del dataset para su uso como train en el entrenamiento de los modelos

Implica tener que estimar un porcentaje óptimo para 2 conjuntos en lugar de 3



Puede tener efectos negativos cuando existe una dimensión temporal en el problema

Es más costoso computacionalmente

Métricas

- Para comparar el rendimiento obtenido por cada combinación de tipo de modelos y conjunto de hiperparámetros necesitaremos de un valor numérico que nos informe de su bondad predictiva.
- Este valor numérico vendrá dado por la métrica elegida.
- La elección de esta métrica dependerá del tipo de problema, de los datos y del objetivo a solucionar.
- Ejemplo: MAE

$$MAE = \frac{1}{n} \sum_{i=1}^n |e_i|, \text{ where } e_i = \text{original}_i - \text{predict}_i$$

Metaparametrización (I)

- Los modelos ML suelen incluir un conjunto de **hiperparámetros** que nos permiten controlar su comportamiento.
- De su correcta elección dependerá la bondad del modelo entrenado.
- Los hiperparámetros dependen del perfil de los datos que estamos analizando (**problem-dependent**), por lo que no es sencillo establecer un procedimiento estándar para su obtención.

Metaparametrización (II). Grid search

1. Elegimos una familia de modelos.
2. Elegimos unos hiperparámetros a optimizar, les llamaremos par1 y par2.
3. Para cada hiperparámetro, elegimos una serie de valores a probar.

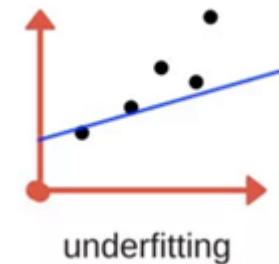
| par1/par2 | 10 | 100 | 1000 |
|-----------|------|------|------|
| 0.1 | 0.3 | 0.22 | 0.25 |
| 0.01 | 0.15 | 0.14 | 0.14 |
| 0.001 | 0.35 | 0.05 | 0.11 |

4. Entrenamos nuestro modelo sobre el conjunto de train con los diferentes hiperparámetros haciendo todas las combinaciones posibles.
5. Hacemos la predicción de los diferentes modelos sobre el conjunto de validación y calculamos el error con la métrica seleccionada.
6. Escogemos el que mejor métrica obtenga y lo aplicamos sobre el conjunto de test para ver el error final esperado de nuestro modelo.

Trade off bias/variance

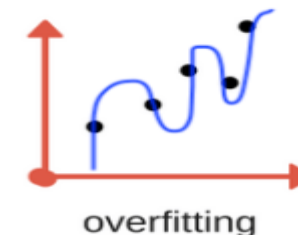
Bias:

- El sesgo es la diferencia entre la predicción promedio de nuestro modelo y el valor correcto que estamos tratando de predecir.
- El modelo con alto sesgo presta muy poca atención a los datos de entrenamiento y simplifica en exceso el modelo.
- Un modelo muy sesgado siempre da un error alto en los datos de train. **Underfitting.**



Variance:

- Es la variabilidad de las predicciones cuando se introducen datos que difieren entre sí.
- El modelo con alta variación se ajusta mucho a los datos de entrenamiento y no generaliza bien con datos que no ha visto antes.
- Dichos modelos funcionan muy bien con los datos de entrenamiento pero tienen altos índices de error en los datos de prueba. **Overfitting.**

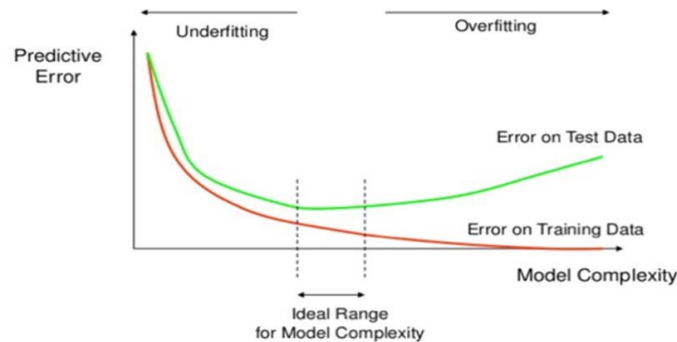


Overfitting/underfitting (I)

Las principales causas al obtener malos resultados en ML son el overfitting o el underfitting de los datos.

- Si nuestros datos de entrenamiento son muy pocos o nuestro modelo es demasiado sencillo no será capaz de aprender a resolver el problema → **underfitting** (High bias - Low variance).
- Cuando el algoritmo sólo se ajusta a aprender los casos particulares que le enseñamos y es incapaz de reconocer nuevos datos de entrada → **overfitting** (Low bias - High variance)

Overfitting/underfitting (II)



¿Cómo detectar el overfitting?

Si el modelo entrenado tiene en el conjunto de validación un error mucho mayor que en el conjunto de train, esto sugiere la posibilidad de un problema de overfitting.

¿Cómo detectar el underfitting?

Cuándo el error de train parece demasiado elevado, podemos tener sospechas de overfitting.

También si en el conjunto de validación sólo se acierta un tipo de clase o el único resultado que se obtiene es siempre el mismo valor, o valores similares.

Overfitting/underfitting (III)

Prevenir el overfitting

- **Cantidad mínima de muestras** tanto para entrenar el modelo como para validarlo.
- **Clases variadas y equilibradas** en cantidad: es importante que los datos de entrenamiento estén balanceados.
- **Conjunto de validación.** Subdividir el conjunto de datos y mantener una porción del mismo «oculto» al modelo.
- Parameter Tunning o **Ajuste de Parámetros:** deberemos experimentar con distintas configuraciones hasta encontrar el equilibrio.
- A veces conviene eliminar o reducir la cantidad de características que utilizaremos para entrenar el modelo, por ejemplo cuando se tiene una cantidad excesiva de dimensiones (features), con muchas variantes distintas, sin suficientes muestras. Una herramienta útil para hacerlo es PCA.

Prevenir el underfitting

- Entrenar un **modelo más complejo.**
- **Aumentar el número de variables** en el dataset.

SISTEMAS DE RECOMENDACIÓN

EDEM

Escuela de Empresarios



Definición

- Un **sistema de recomendación** es un algoritmo que nos permite **dar predicciones de cuál es el producto o ítem más adecuado para un usuario**.
- Los sistemas de recomendación pueden ser de **varias clases** según el algoritmo utilizado: basados en contenido, filtrado colaborativo, etc.
- Los sistemas de recomendación basados en algoritmos de filtrado colaborativo utilizan las valoraciones o interacciones de los usuarios sobre ciertos elementos del conjunto total para predecir interés en el resto de los elementos y recomendar los de mayor valoración predicha.
- Tipos de filtrado colaborativo:
 1. User-based Collaborative Filtering
 2. Item-based Collaborative Filtering

User Based

- **Personas con intereses similares** en el pasado es probable que tengan intereses parecidos en el futuro.
- Para predecir **el interés de un usuario sobre un producto (ítem) usamos la opinión o interacciones de usuarios similares**. Cuánto más similar sea el usuario, más tendremos en cuenta su opinión o historial de interacciones.
- Se calculan las **similitudes entre usuarios** en base a las opiniones o interacciones sobre los productos usando una determinada distancia.

$$sim(user1, user2) = \frac{\sum_j dist(puntuacion(user_1, item_j), puntuacion(user_2, item_j))}{\text{Número de items}}$$

- **Recomendaciones basadas en la actividad de usuarios similares a mí.**
- **Ejemplo:** Netflix.
 - Jesús ha visto True detective y Breaking Bad.
 - Miguel ha visto True detective, Breaking Bad y Fargo.
 - El algoritmo infiere que Jesús y Miguel son usuarios similares.
 - El sistema de recomendación **recomienda Fargo a Jesús**.

Item Based

- Si a un **usuario le ha interesado en el pasado** un determinado producto, es probable que en el futuro le interesen **productos similares**.
- Para predecir el interés de un usuario sobre un producto (ítem) usamos su opinión o interacciones sobre productos similares. Cuánto más similar sea el ítem, más tendremos el historial de puntuaciones o interacciones.
- En este caso se **calculan las similitudes entre ítems** o productos en función de las opiniones o interacciones de los usuarios.

$$sim(item_1, item_2) = \frac{\sum_j dist(puntuacion(user_j, item_1), puntuacion(user_j, item_2))}{\text{Número de users}}$$

- **Recomendaciones basadas en la actividad relacionada con productos similares a los que yo he comprado.**
- **Ejemplo: Netflix.**
 - True detective ha sido vista por Jesús, Sonia y Miguel.
 - Fargo ha sido vista por Sonia y Miguel.
 - El algoritmo infiere que True detective y Fargo son ítems similares.
 - El sistema de recomendación **recomienda Fargo a Jesús.**

Item VS User Based

Ventajas del Enfoque ítem-based

- En la mayoría de los casos **disponemos de más usuarios que ítems**, por lo que la matriz de similitudes final **es más escalable** al tener dimensiones más bajas. Si es una matriz de similitudes ítem-ítem, la dimensión de la matriz será $N \times N$, donde N es el número de ítems, en vez de $M \times M$, donde M es número de usuarios y $M \gg N$.
- Disponemos de **más interacciones por elemento** en el enfoque ítem-ítem, por lo que los **resultados** son más estables y **estadísticamente significativos**.
- En muchos casos, el **inventario de productos se mantiene más estable en el tiempo** que el inventario o bolsa de usuarios.

Algoritmo Item Based

- **Paso 1:** Se calcula una **matriz** de puntuaciones o **interacción usuario-ítem** ($LM \times N$).
- **Paso 2:** Se calcula una **matriz de similitudes entre los ítems** o rutas ($SN \times N$) usando alguna distancia, por ejemplo, distancia del coseno.
- **Paso 3:** **multiplicar matriz de interacciones por matriz de similitudes** para obtener el ítem que tiene más probabilidad de interesarle a cada usuario.
- **Paso 4:** generar las **recomendaciones (lista con n primeros elementos) mediante la ordenación** de las puntuaciones para cada usuario.

Distancias

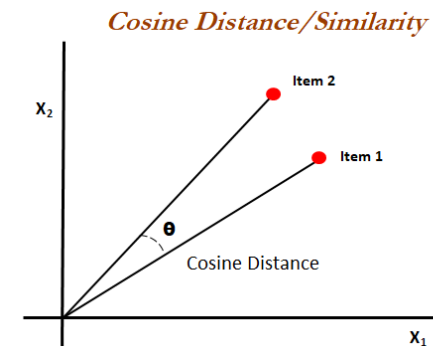
- Basamos el CF en similitudes ítem-ítem; hay que **calcular una matriz de similitudes** entre ítems.
- Para ello, hay multitud de medidas de similitud posibles, una de las más populares es la distancia del coseno:

$$sim_{\cos(item1,item2)} = \frac{item1 \cdot item2}{||item1|| ||item2||}$$

- También hay otras métricas de similitud como:
 - Correlación de Pearson
 - Distancia de Jaccard

$$\rho_{xy} = \frac{Cov_{xy}}{\sigma_x \sigma_y}$$

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$



EJEMPLO (I)

Supongamos que **tenemos 5 rutas** con los siguientes destinos (origen Madrid):

Ítem 1: Tenerife (TFN).

Ítem 2: Sevilla (SVQ).

Ítem 3: Santiago (SCQ).

Ítem 4: Valencia (VAL).

Ítem 5: Vigo (VGO).

7 clientes con sus interacciones:

Usuario 1:

Ha comprado a TFN y VAL.

Usuario 2:

Ha buscado en la web VAL.

Usuario 3:

Ha comprado a SCQ.

Ha buscado SCQ y VGO (2 veces).

Ha clicado en la campaña de SCQ.

Usuario 4:

Ha clicado en la campaña de TFN, SVQ y VAL.

Usuario 5:

Ha comprado a VAL, VGO, SCQ (2 veces).

Ha buscado SCQ (3 veces).

Usuario 6:

Ha buscado VAL (3 veces) y SVQ.

Ha clicado en la campaña de VAL y TFN.

Usuario 7:

Ha comprado TFN.

Ha buscado TFN.

Ha clicado TFN (2 veces) y VAL.

EJEMPLO (II)

Interacciones clientes:

Usuario 1:

Ha comprado a TFN y VAL.

Usuario 2:

Ha buscado en la web VAL.

Usuario 3:

Ha comprado a SCQ.

Ha buscado SCQ y VGO (2 veces).

Ha clicado en la campaña de SCQ.

Usuario 4:

Ha clicado en la campaña de TFN, SVQ y VAL.

Usuario 5:

Ha comprado a VAL, VGO, SCQ (2 veces).

Ha buscado SCQ (3 veces).

Usuario 6:

Ha buscado VAL (3 veces) y SVQ.

Ha clicado en la campaña de VAL y TFN.

Usuario 7:

Ha comprado TFN.

Ha buscado TFN.

Ha clicado TFN (2 veces) y VAL.

MATRIZ COMPRAS

| | SCQ | SVQ | TFN | VAL | VGO |
|----------|-----|-----|-----|-----|-----|
| Usuario1 | 0 | 0 | 1 | 1 | 0 |
| Usuario2 | 0 | 0 | 0 | 0 | 0 |
| Usuario3 | 1 | 0 | 0 | 0 | 0 |
| Usuario4 | 0 | 0 | 0 | 0 | 0 |
| Usuario5 | 2 | 0 | 0 | 1 | 1 |
| Usuario6 | 0 | 0 | 0 | 0 | 0 |
| Usuario7 | 0 | 0 | 1 | 0 | 0 |

MATRIZ BÚSQUEDAS

| | SCQ | SVQ | TFN | VAL | VGO |
|----------|-----|-----|-----|-----|-----|
| Usuario1 | 0 | 0 | 0 | 0 | 0 |
| Usuario2 | 0 | 0 | 0 | 1 | 0 |
| Usuario3 | 1 | 0 | 0 | 0 | 2 |
| Usuario4 | 0 | 0 | 0 | 0 | 0 |
| Usuario5 | 3 | 0 | 0 | 0 | 0 |
| Usuario6 | 0 | 1 | 0 | 3 | 0 |
| Usuario7 | 0 | 0 | 1 | 0 | 0 |

MATRIZ CLICS

| | SCQ | SVQ | TFN | VAL | VGO |
|----------|-----|-----|-----|-----|-----|
| Usuario1 | 0 | 0 | 0 | 0 | 0 |
| Usuario2 | 0 | 0 | 0 | 0 | 0 |
| Usuario3 | 1 | 0 | 0 | 0 | 0 |
| Usuario4 | 0 | 1 | 1 | 1 | 0 |
| Usuario5 | 0 | 0 | 0 | 0 | 0 |
| Usuario6 | 0 | 0 | 1 | 1 | 0 |
| Usuario7 | 0 | 0 | 2 | 1 | 0 |

EJEMPLO (III)

| MATRIZ COMPRAS | | | | | |
|----------------|-----|-----|-----|-----|-----|
| | SCQ | SVQ | TFN | VAL | VGO |
| Usuario1 | 0 | 0 | 1 | 1 | 0 |
| Usuario2 | 0 | 0 | 0 | 0 | 0 |
| Usuario3 | 1 | 0 | 0 | 0 | 0 |
| Usuario4 | 0 | 0 | 0 | 0 | 0 |
| Usuario5 | 2 | 0 | 0 | 1 | 1 |
| Usuario6 | 0 | 0 | 0 | 0 | 0 |
| Usuario7 | 0 | 0 | 1 | 0 | 0 |

| MATRIZ BÚSQUEDAS | | | | | |
|------------------|-----|-----|-----|-----|-----|
| | SCQ | SVQ | TFN | VAL | VGO |
| Usuario1 | 0 | 0 | 0 | 0 | 0 |
| Usuario2 | 0 | 0 | 0 | 1 | 0 |
| Usuario3 | 1 | 0 | 0 | 0 | 2 |
| Usuario4 | 0 | 0 | 0 | 0 | 0 |
| Usuario5 | 3 | 0 | 0 | 0 | 0 |
| Usuario6 | 0 | 1 | 0 | 3 | 0 |
| Usuario7 | 0 | 0 | 1 | 0 | 0 |

| MATRIZ CLICS | | | | | |
|--------------|-----|-----|-----|-----|-----|
| | SCQ | SVQ | TFN | VAL | VGO |
| Usuario1 | 0 | 0 | 0 | 0 | 0 |
| Usuario2 | 0 | 0 | 0 | 0 | 0 |
| Usuario3 | 1 | 0 | 0 | 0 | 0 |
| Usuario4 | 0 | 1 | 1 | 1 | 0 |
| Usuario5 | 0 | 0 | 0 | 0 | 0 |
| Usuario6 | 0 | 0 | 1 | 1 | 0 |
| Usuario7 | 0 | 0 | 2 | 1 | 0 |

*

| MATRIZ INTERACCIÓN FINAL ($I_{7 \times 5}$) | | | | | |
|-----------------------------------------------|-----|-----|-----|-----|-----|
| | SCQ | SVQ | TFN | VAL | VGO |
| Usuario1 | 0 | 0 | 1 | 1 | 0 |
| Usuario2 | 0 | 0 | 0 | 1 | 0 |
| Usuario3 | 3 | 0 | 0 | 0 | 2 |
| Usuario4 | 0 | 1 | 1 | 1 | 0 |
| Usuario5 | 5 | 0 | 0 | 1 | 1 |
| Usuario6 | 0 | 1 | 1 | 4 | 0 |
| Usuario7 | 0 | 0 | 4 | 1 | 0 |

*Nota: Ejemplo ilustrativo simplificado, en la práctica es una combinación ponderada.

EJEMPLO (IV)

Una vez tenemos la matriz de interacción user-ítem

| MATRIZ INTERACCIÓN FINAL ($L_{7 \times 5}$) | | | | | |
|-----------------------------------------------|-----|-----|-----|-----|-----|
| | SCQ | SVQ | TFN | VAL | VGO |
| Usuario1 | 0 | 0 | 1 | 1 | 0 |
| Usuario2 | 0 | 0 | 0 | 1 | 0 |
| Usuario3 | 3 | 0 | 0 | 0 | 2 |
| Usuario4 | 0 | 1 | 1 | 1 | 0 |
| Usuario5 | 5 | 0 | 0 | 1 | 1 |
| Usuario6 | 0 | 1 | 1 | 4 | 0 |
| Usuario7 | 0 | 0 | 4 | 1 | 0 |

Calculamos ahora la matriz de similitudes. Enfoque ítem-ítem con la distancia del coseno.

Si queremos obtener, por ejemplo, la **similitud del ítem 1 (SCQ) con el ítem 5 (VGO)** el cálculo sería.

$$sim(item_1, item_5) = \frac{(0,0,3,0,5,0,0) \cdot (0,0,2,0,1,0,0)}{||(0,0,3,0,5,0,0)|| \cdot ||(0,0,2,0,1,0,0)||} = \frac{6 + 5}{\sqrt{34}\sqrt{5}} = 0.84$$

| $S_{5 \times 5}$ | SCQ | SVQ | TFN | VAL | VGO |
|------------------|------|------|------|------|------|
| SCQ | 1 | 0 | 0 | 0.19 | 0.84 |
| SVQ | 0 | 1 | 0.32 | 0.77 | 0 |
| TFN | 0 | 0.32 | 1 | 0.50 | 0 |
| VAL | 0.19 | 0.77 | 0.50 | 1 | 0.10 |
| VGO | 0.84 | 0 | 0 | 0.10 | 1 |

EJEMPLO (V)

Calculamos la ruta que le puede resultar más interesante **al cliente 4**:

- Ha clicado TFN, SVQ y VAL.

Multiplicamos el vector de interés del cliente 4 de la matriz de interacciones user-ítem es decir, $I_{(4,)} = (0,1,1,1,0)$, por cada una de las columnas de la matriz de similitudes S calculada.

$$\text{SCQ: } (0, 1, 1, 1, 0) * (1, 0, 0, 0.19, 0.84) = 0.19$$

$$\text{SVQ: } (0, 1, 1, 1, 0) * (0, 1, 0.32, 0.77, 0) = 2.09$$

$$\text{TFN: } (0, 1, 1, 1, 0) * (0, 0.32, 1, 0.50, 0) = 1.82$$

$$\text{VAL: } (0, 1, 1, 1, 0) * (0.19, 0.77, 0.5, 1, 0.1) = 2.27$$

$$\text{VGO: } (0, 1, 1, 1, 0) * (0.84, 0, 0, 0.10, 1) = 0.10$$

| $S_{5 \times 5}$ | SCQ | SVQ | TFN | VAL | VGO |
|------------------|------|------|------|------|------|
| SCQ | 1 | 0 | 0 | 0.19 | 0.84 |
| SVQ | 0 | 1 | 0.32 | 0.77 | 0 |
| TFN | 0 | 0.32 | 1 | 0.50 | 0 |
| VAL | 0.19 | 0.77 | 0.50 | 1 | 0.10 |
| VGO | 0.84 | 0 | 0 | 0.10 | 1 |

Le recomendaríamos Valencia por haber obtenido el score de interés mayor, siendo SVQ la segunda ruta a recomendar.

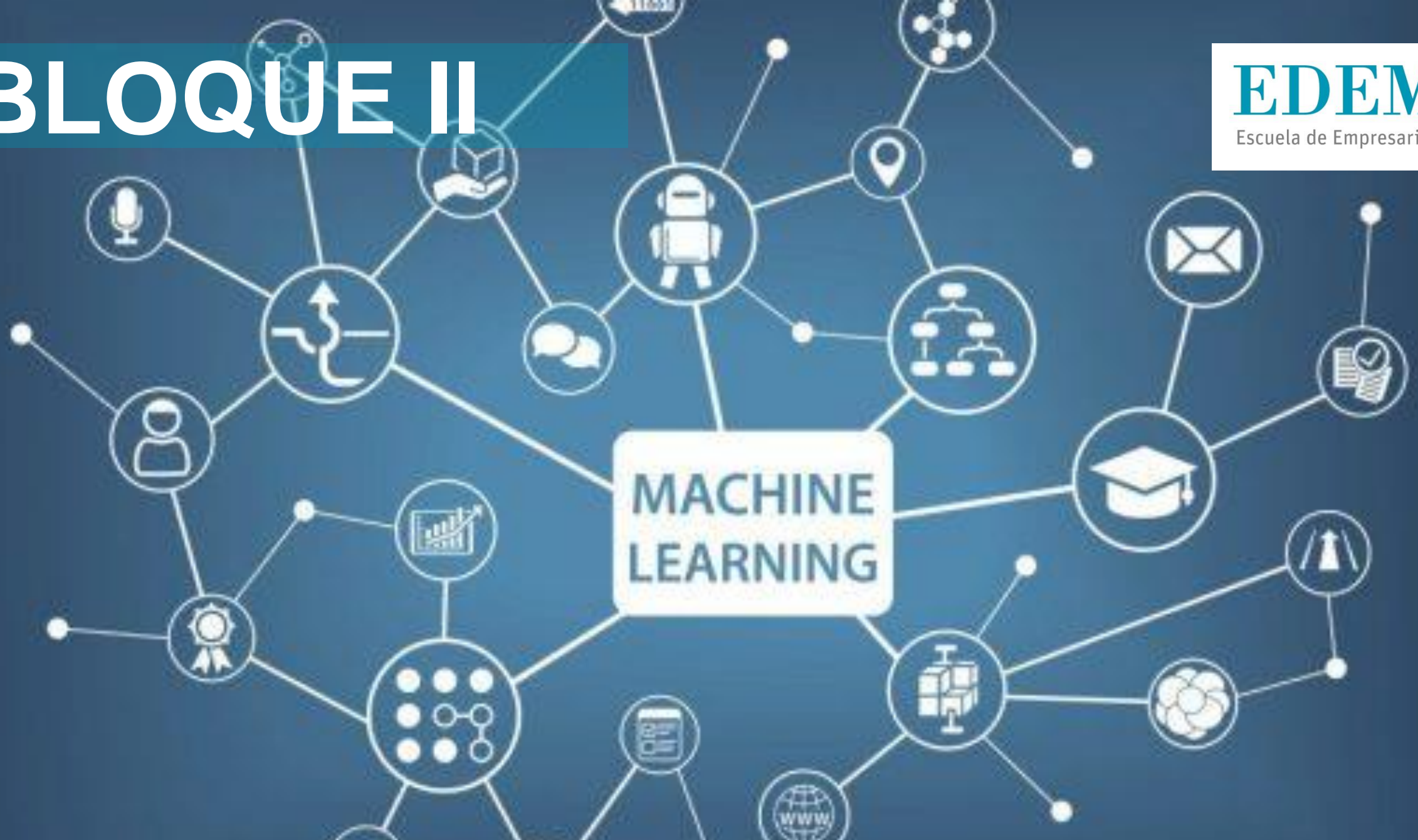
Siguiendo un proceso análogo **para el resto de usuarios**, lo cual es equivalente a **multiplicar $I * S$ (matriz de interacciones user-ítem por matriz de similitudes ítem-ítem)** obtendríamos la siguiente matriz de scores de interés:

| | SCQ | SVQ | TFN | VAL | VGO |
|----------|------|------|------|------|------|
| Usuario1 | 0,19 | 1,09 | 1,5 | 1,5 | 0,1 |
| Usuario2 | 0,19 | 0,77 | 0,5 | 1 | 0,1 |
| Usuario3 | 4,68 | 0 | 0 | 0,77 | 4,52 |
| Usuario4 | 0,19 | 2,09 | 1,82 | 2,27 | 0,1 |
| Usuario5 | 6,03 | 0,77 | 0,5 | 2,05 | 5,3 |
| Usuario6 | 0,76 | 4,4 | 3,32 | 5,27 | 0,4 |
| Usuario7 | 0,19 | 2,05 | 4,5 | 3 | 0,1 |

RESUMEN MODELOS



BLOQUE II



BLOQUE II (02/04/2022): Modelos ML I, regresión lineal, logística y SVM. Clasificación y Regresión.

TEORÍA

1. Regresión lineal
 - Conceptos.
 - Regularización. Lasso / Ridge.
2. Regresión logística
 - Conceptos.
 - Regularización. Lasso / Ridge.
3. SVM
 - Conceptos.
 - Clasificación.
 - Regresión.

BLOQUE II (02/04/2022): Modelos ML I, regresión lineal, logística y SVM. Clasificación y Regresión

PRÁCTICA:

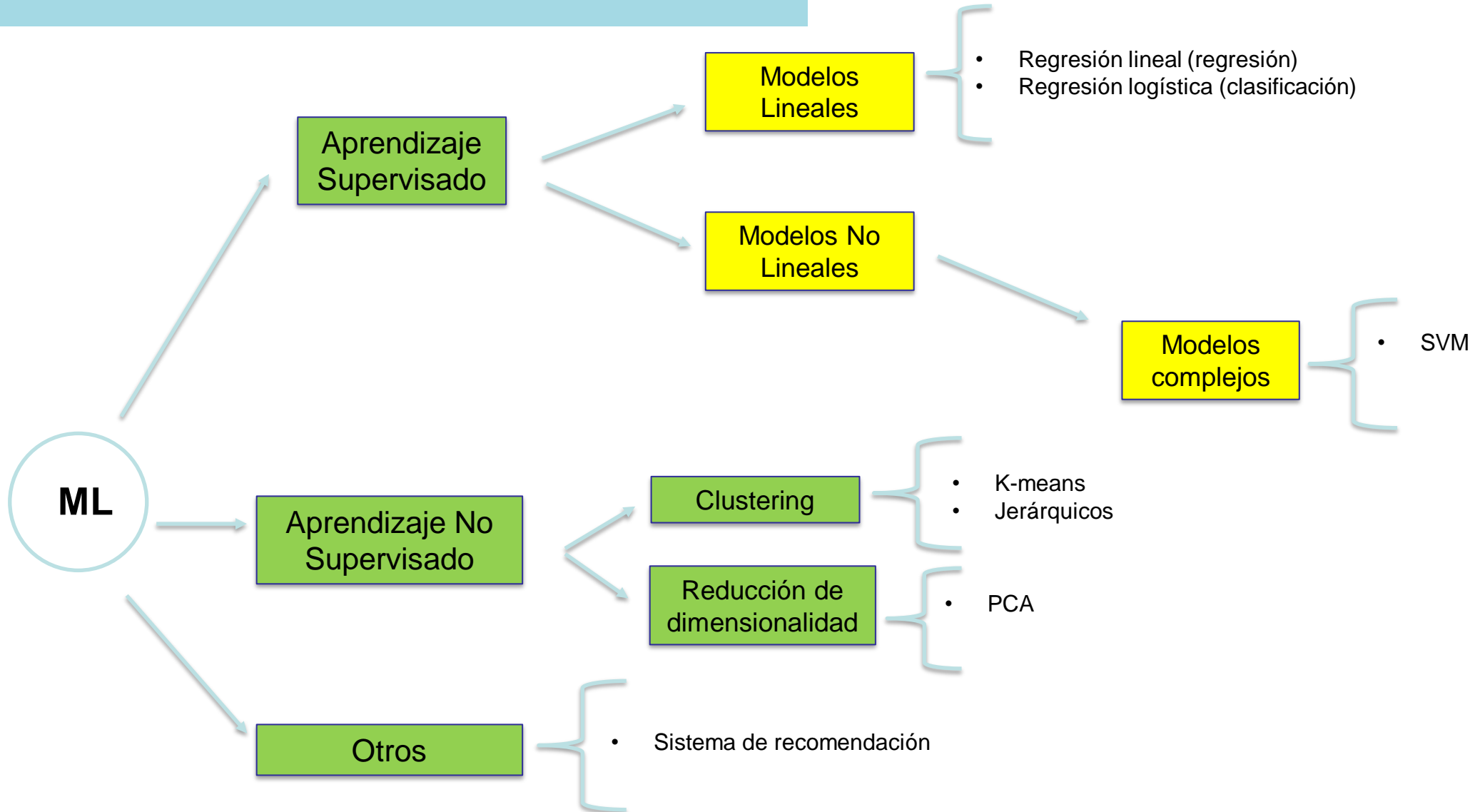
Datasets:

- Dataset scikit-learn.
- Datos médicos.
- Datos aerolínea (Iberia Express).







Ejercicios:

- Script de regresión lineal.
- Script SVM.
- Problema de regresión en una aerolínea. Predicción de demanda.
- Problema de clasificación en medicina.

RESUMEN MODELOS



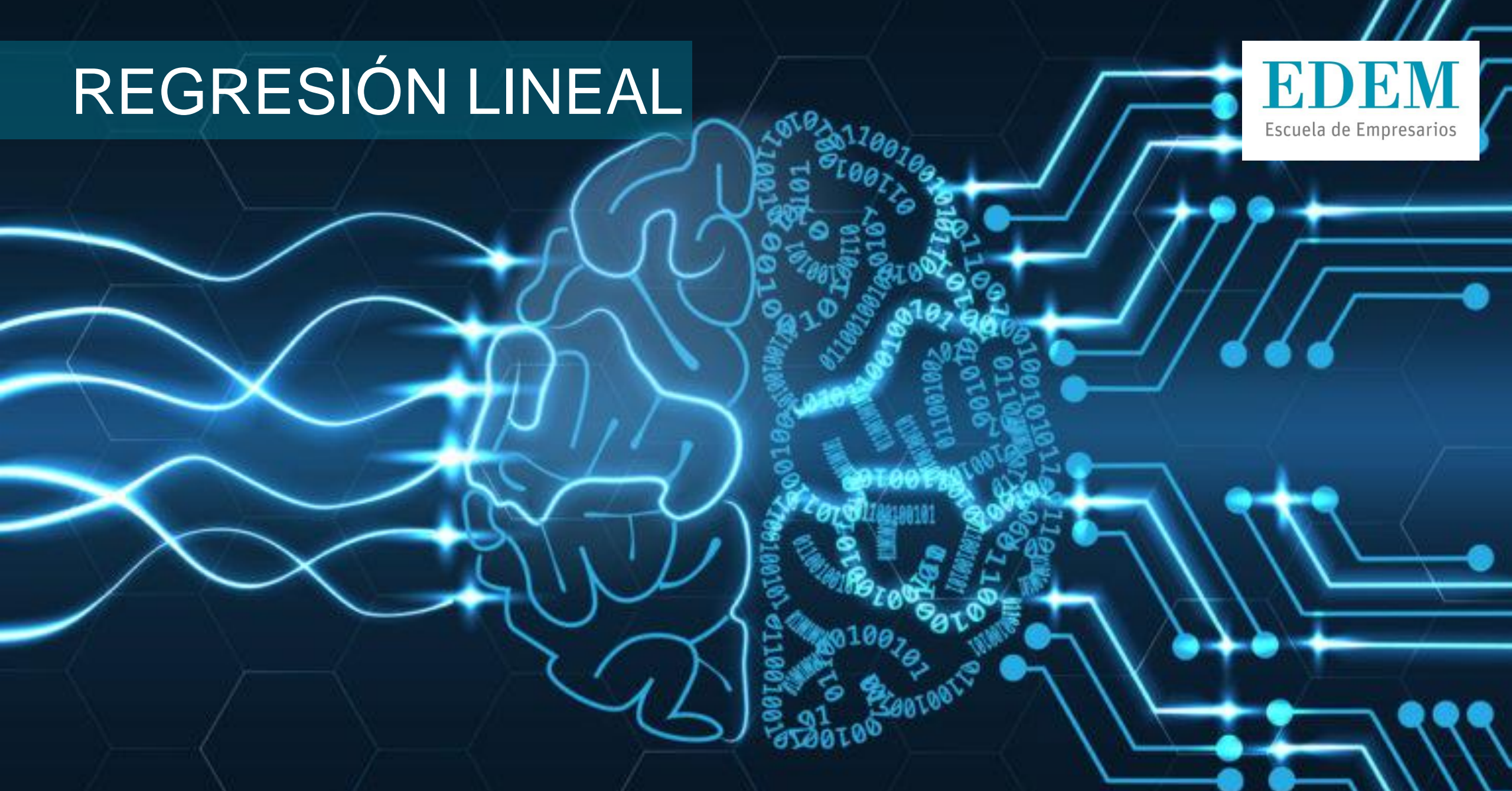
COMPARACIÓN ALGORITMOS

| | TYPE | NAME | DESCRIPTION | ADVANTAGES | DISADVANTAGES |
|----------------|-------------------------------------------------------------------------------------|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Linear |  | Linear regression | The "best fit" line through all data points. Predictions are numerical. | Easy to understand -- you clearly see what the biggest drivers of the model are. | <ul style="list-style-type: none"> ✗ Sometimes too simple to capture complex relationships between variables. ✗ Tendency for the model to "overfit". |
| |  | Logistic regression | The adaptation of linear regression to problems of classification (e.g., yes/no questions, groups, etc.) | Also easy to understand. | <ul style="list-style-type: none"> ✗ Sometimes too simple to capture complex relationships between variables. ✗ Tendency for the model to "overfit". |
| Tree-based |  | Decision tree | A graph that uses a branching method to match all possible outcomes of a decision. | Easy to understand and implement. | <ul style="list-style-type: none"> ✗ Not often used on its own for prediction because it's also often too simple and not powerful enough for complex data. |
| |  | Random Forest | Takes the average of many decision trees, each of which is made with a sample of the data. Each tree is weaker than a full decision tree, but by combining them we get better overall performance. | A sort of "wisdom of the crowd". Tends to result in very high quality models. Fast to train. | <ul style="list-style-type: none"> ✗ Can be slow to output predictions relative to other algorithms. ✗ Not easy to understand predictions. |
| |  | Gradient Boosting | Uses even weaker decision trees, that are increasingly focused on "hard" examples. | High-performing. | <ul style="list-style-type: none"> ✗ A small change in the feature set or training set can create radical changes in the model. ✗ Not easy to understand predictions. |
| Support Vector |  | SVM | Creates a hyperplane with maximum margin. | Can handle extremely complex tasks | <ul style="list-style-type: none"> ✗ Slow to train ✗ Difficult to understand |

REGRESIÓN LINEAL

EDEM

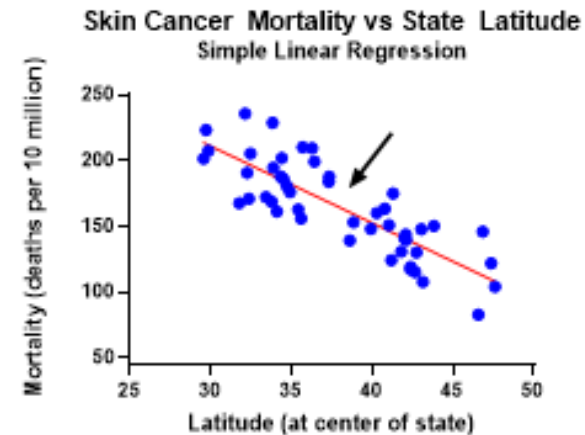
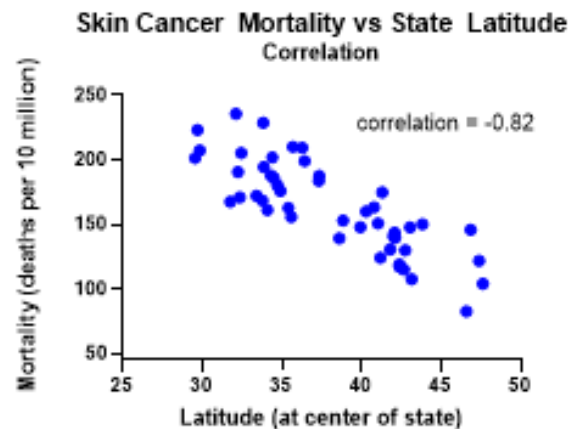
Escuela de Empresarios



- Conceptos.
 - Objetivo.
 - Definición.
 - Optimización de coeficientes. Función de coste.
 - Ejemplo.
- Regularización. Lasso/Ridge.

Objetivo

- Se aplica a problemas de aprendizaje **supervisado de regresión**.
- Es la familia de modelos ML más **sencilla** para este tipo de problemas.
- Trata de **ajustar una recta** lo mejor posible a un grupo de elementos, en este caso, una serie de targets numéricos a predecir.



Definición

- Trata de modelar la relación entre las variables de entrada o patrones y la variable de salida o target mediante la siguiente ecuación:

$$\hat{y}_i = w_0 + w_1x_{i1} + w_2x_{i2} + \dots + w_dx_{id} ,$$

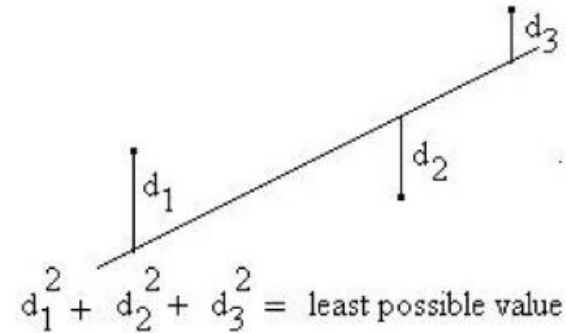
donde:

- \hat{y}_i es la predicción dada por el modelo para la instancia o caso i.
 - $x_i = (x_1 + x_2 + \dots + x_d)$ son las variables predictoras o input del modelo para la fila i.
 - $w = (w_1 + w_2 + \dots + w_d)$ son los coeficientes o pesos de la regresión lineal para cada una de las variables predictoras x.
 - d es el número de dimensiones o variables del input.
 - w_0 es el llamado bias de la regresión lineal.
- Podemos expresar la predicción del modelo para todos lo registros como: $\hat{y} = Xw + w_0$

Cálculos de los pesos, w. Función de coste

- ¿Cómo se calculan los pesos w de la regresión lineal?
- Son aquellos que minimizan una métrica, normalmente llamada **función de coste**.
- En la regresión lineal, la función de coste es la distancia al cuadrado entre las predicciones del modelo, \hat{y} , y el valor real del target, y . Es decir, el MSE:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$



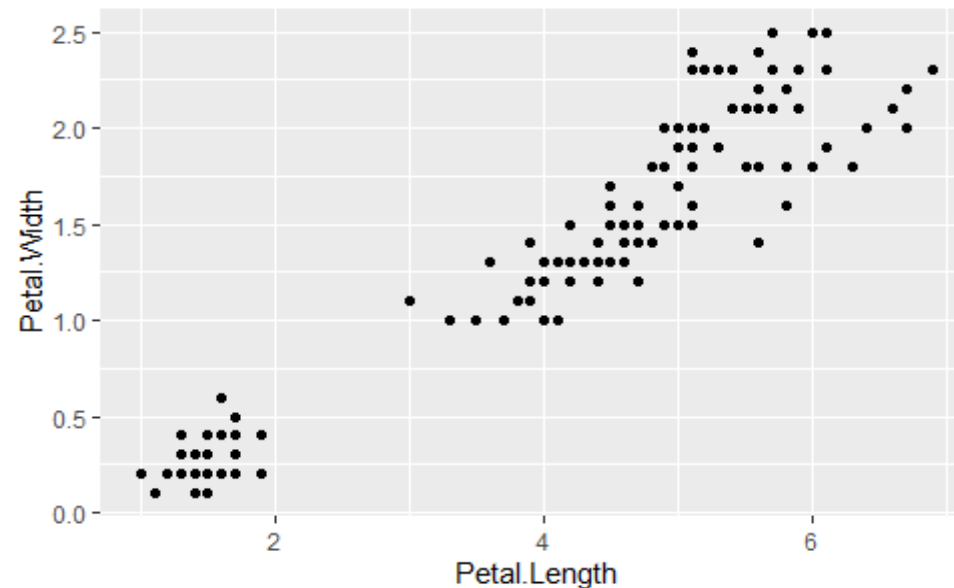
- Por eso este modelo también es conocido como **Least Squares Regression**.
- Se puede demostrar que $w = (X^T X)^{-1} X^T y$.

Ejemplo (I)

- Vamos a entrenar una regresión lineal sobre el dataset iris para predecir Petal.Width en base a los valores de Petal.Length. Es decir:

$y = \text{Petal.Width}$

$X = \text{Petal.Length}$



Ejemplo (II)

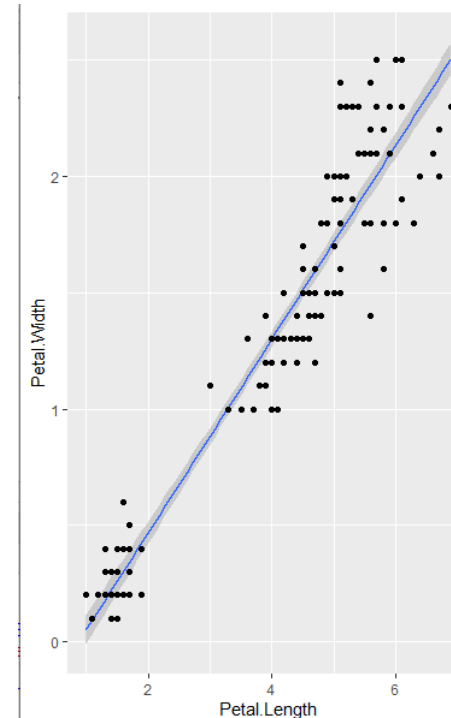
- Los pesos óptimos, \mathbf{w} , obtenidos, es decir aquellos que minimizan el MSE, son:

$$w_0 = -0,3631$$

$$w_1 = 0,4158$$

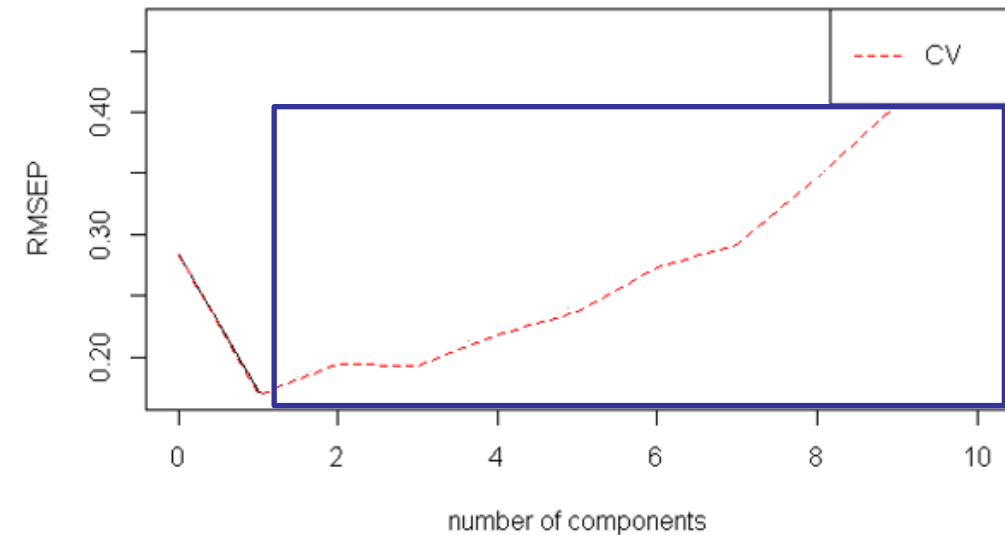
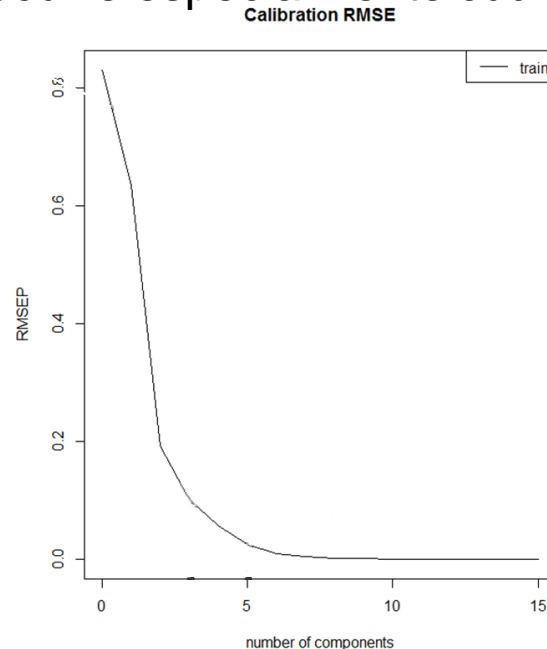
- Por tanto:

$$\widehat{Petal.Width} = \theta_0 + \theta_1 \cdot Petal.Length$$
$$\widehat{Petal.Width} = -0.3631 + 0.4158 \cdot Petal.Length$$



Objetivo

- Si aplicamos la formulación estándar de la regresión lineal podemos tener problemas de Overfitting.
- Esto ocurre especialmente cuándo se utiliza un alto número de variables predictoras.



¡Overfitting!

- Para evitar este fenómeno se utilizan técnicas de regularización.

Definición

- Consiste en obtener los pesos, w , del modelo minimizando el siguiente valor (**función objetivo**)

$$l(\widehat{y}_i, y_i) + \lambda \text{reg}(w) ,$$

donde:

- l es la función de coste. En el caso de la regresión lineal $l = \text{MSE}$.
- reg es la función de regularización o penalty term. Penaliza w con valores grandes. Hay distintas opciones.
- λ es el hiperparámetro que controla la magnitud de la regularización.
 - $\lambda = 0 \rightarrow$ Modelo estándar, sin regularización.
 - λ pequeño \rightarrow Modelo con poca regularización, es decir, más varianza o complejidad.
 - λ grande \rightarrow Modelo con mucha regularización, $w \approx 0$, es decir, menos varianza y más bias.

Tienden al
overfitting

Tiende al
underfitting

Lasso o regularización L1

- En este caso la función de regularización es la norma L1

$$reg(w) = ||w||_1 = \sum_{i=1}^d |w_i| ,$$

- Ejemplo:

$$w_i = (1, -2, 3) \rightarrow ||w||_1 = 1 + 2 + 3 = 6$$

- Por tanto, la función objetivo a minimizar en Lasso regression es:

$$l(\hat{y}_i - y) + \lambda reg(w) = \frac{1}{N} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^d |w_i|$$

Ridge o regularización L2

- En este caso la función de regularización es la norma L2 al cuadrado

$$reg(w) = (\|w\|_2)^2 = \sum_{i=1}^d w_i^2,$$

- Ejemplo:

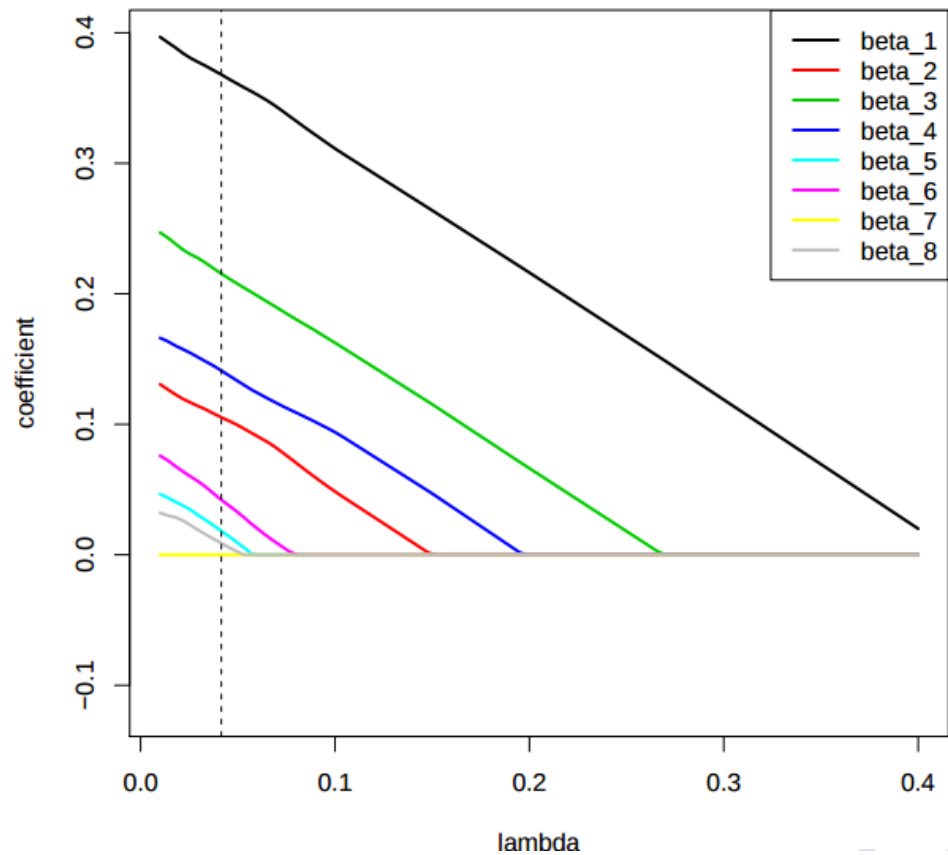
$$w_i = (1, -2, 3) \rightarrow (\|w\|_2)^2 = 1^2 + 2^2 + 3^2 = 14$$

- Por tanto, la función objetivo a minimizar en Ridge regression es:

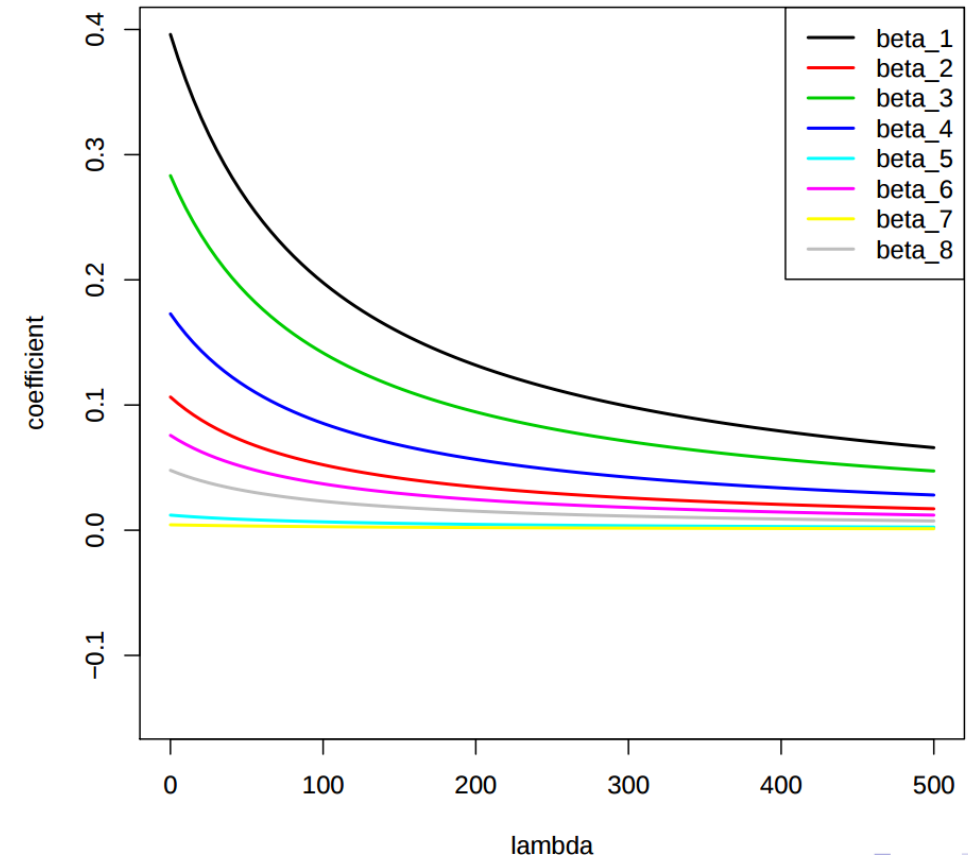
$$l(\hat{y}_i - y) + \lambda reg(w) = \frac{1}{N} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^d w_i^2$$

Lasso vs Ridge

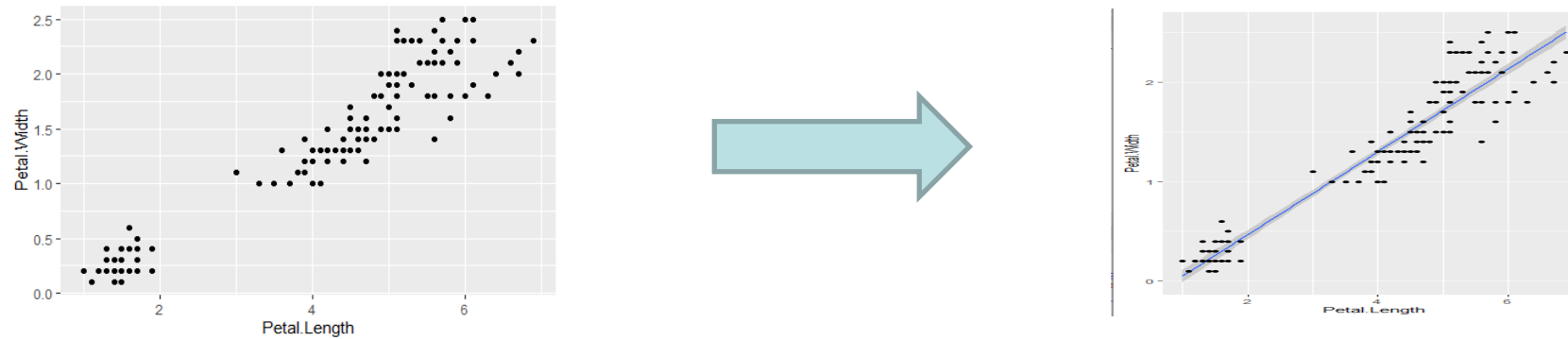
Lasso



Ridge



- **Tipo de problemas:** ML supervisado de regresión.
- **Intuición:** Trata de ajustar una recta lo mejor posible a un grupo de elementos.



- **Fórmula:** $\hat{y}_i = w_0 + w_1x_{i1} + w_2x_{i2} + \dots + w_dx_{id}$
- **Regularización:** Busca evitar overfitting.

Lasso: $\frac{1}{N} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^d |w_i|$

Ridge: $\frac{1}{N} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^d w_i^2$

- **PROS/CONS:** Es la familia de modelos ML más **sencilla** para este tipo de problemas.

REGRESIÓN LOGÍSTICA

EDEM

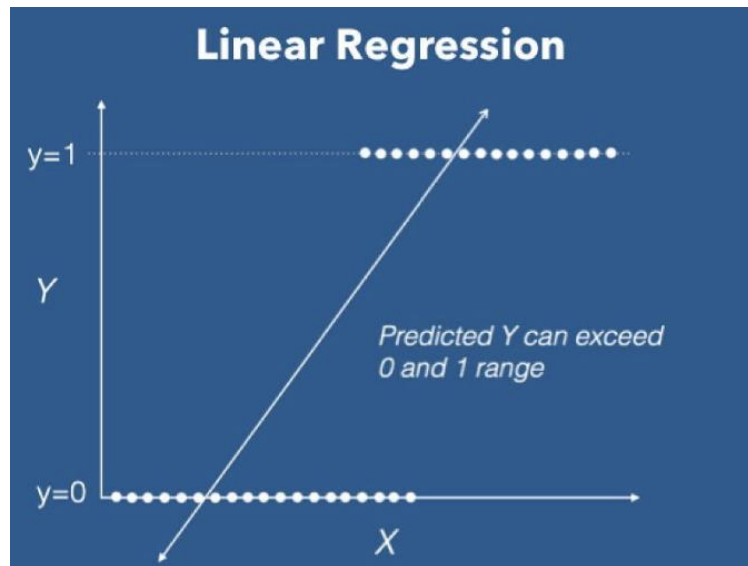
Escuela de Empresarios




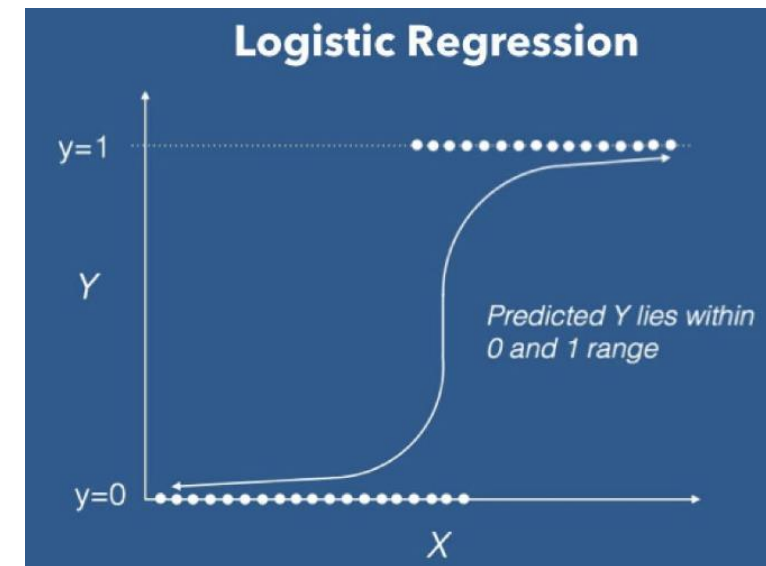
- Conceptos.
 - Objetivo.
 - Definición. Problema binario.
 - Definición. Problema multiclase.
 - Optimización de coeficientes. Función de coste.
 - Ejemplo.
- Regularización. Lasso/Ridge.

Objetivo

- Se aplica a problemas de aprendizaje **supervisado de clasificación**.
- Es la familia de modelos ML más **sencilla** para este tipo de problemas.
- Mide la relación entre la variable dependiente categórica y variables independientes numéricas estimando probabilidades utilizando una **función logística al resultado de una regresión lineal**.



$$\phi(z) = \frac{1}{1 + e^{-z}}$$




Definición. Problema binario (2 clases)

- Es una combinación de dos pasos:
 - Primero se obtiene el resultado de una regresión lineal

$$z_i = w_0 + w_1x_{i1} + w_2x_{i2} + \dots + w_dx_{id}$$

- A este resultado se le aplica la función logística para obtener probabilidades

$$\hat{y}_i = P(y_i = 1) = \text{logistic}(z_i) = \frac{1}{1 + e^{-z_i}}$$

- Por lo que la fórmula final es:

$$\hat{y}_i = \frac{1}{1 + e^{-w_0 + w_1x_{i1} + \dots + w_dx_{id}}}$$

Definición. Problema multiclase ($k > 2$ clases). Opción 1

- Una opción sería entrenar k regresiones logísticas binarias, donde el resultado de cada una de ellas nos daría la probabilidad de pertenencia a la correspondiente clase.
- Por tanto, tendríamos k regresiones lineales, una para cada clase.

$$z_{ic} = w_{0c} + w_{1c}x_{i1} + w_{2c}x_{i2} + \dots + w_{dc}x_{id}$$

- Por ejemplo, el resultado de la regresión logística para la clase c

$$\widehat{y}_{ic} = P(y_i = c) = \text{logistic}(z_{ic}) = \frac{1}{1 + e^{-z_{ic}}}$$

Nos daría la probabilidad de que el registro i pertenezca a la clase c .

- Para asignar una clase final para el registro i bastaría con buscar el valor máximo de estas k probabilidades, es decir, la c que maximiza \widehat{y}_{ic} .

Definición. Problema multiclase ($k > 2$ clases). Opción 2

- Sin embargo, normalmente se opta por otra solución.
- Para ello se reemplaza la función logística por la función softmax:

$$\widehat{y}_{ic} = P(y = c) = \text{softmax}(z_i) = \frac{e^{z_{ic}}}{\sum_{j=1}^k e^{-z_{ij}}}$$

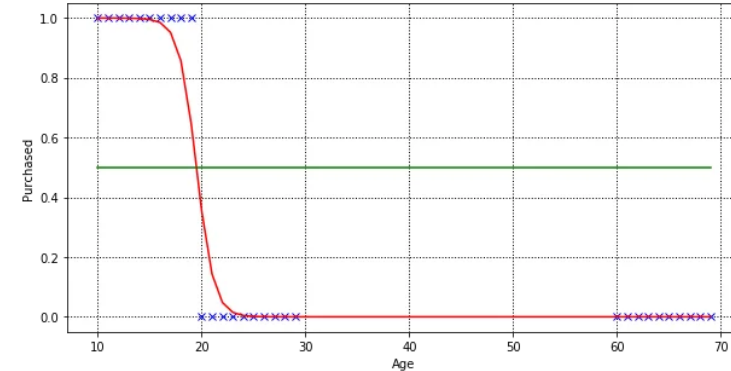
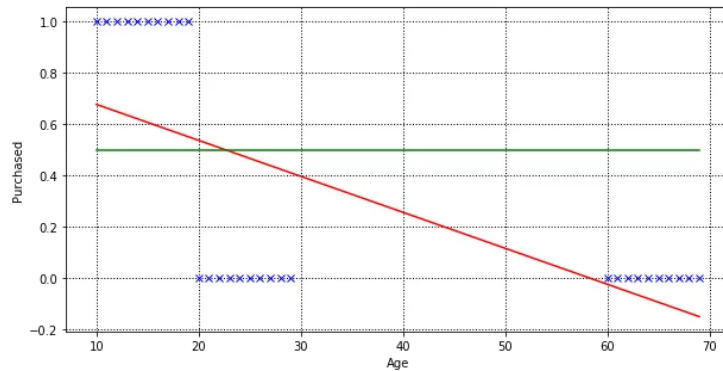
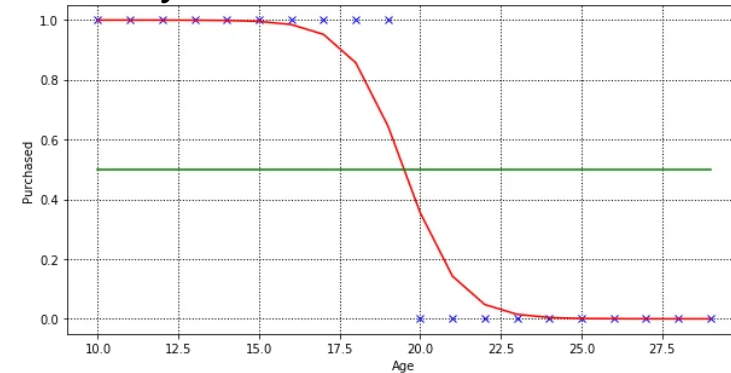
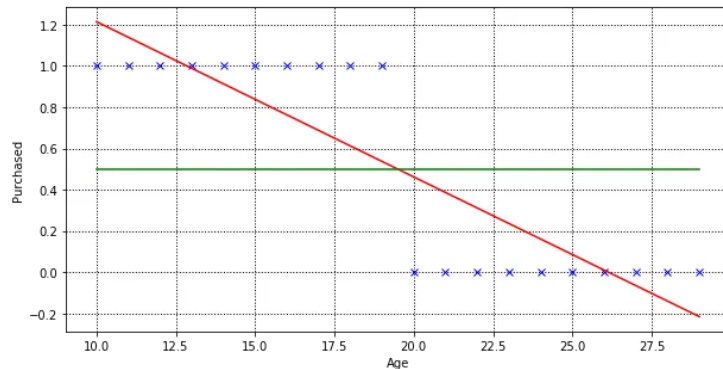
- A diferencia de la opción anterior, ahora se cumple:

$$\sum_{k=0}^n \widehat{y}_i = 1$$

- De nuevo la clase asignada será la c que maximiza \widehat{y}_{ic} .

Regresión lineal vs regresión logística en clasificación

- La regresión lineal no da probabilidades, solo un score. Además, este no está en (0,1).
- La regresión lineal es muy sensible a outliers o datos muy diferentes entre sí.



Cálculos de los pesos, w. Función de coste

- En este caso la función de coste utilizada en lugar del MAE es la **cross-entropy**.
- Para 2 clases:

$$l(\hat{y}_i, y_i) = -y_i \log(\hat{y}_i) - (1 - y_i) \log(1 - \hat{y}_i)$$

| y_i | \hat{y}_i | $l(\hat{y}_i, y_i)$ |
|-------|-------------|-----------------------------------------|
| 0 | 0 | $-0 - 1\log(1) = -\log(1) = \mathbf{0}$ |
| 0 | 1 | $-0 - 1\log(0) = -\log(0) = \infty$ |
| 1 | 0 | $-1\log(0) - 0 = -\log(0) = \infty$ |
| 1 | 1 | $-1\log(1) - 0 = -\log(1) = \mathbf{0}$ |

- Para $k > 2$ clases:

$$l(\hat{y}_i, y_i) = -\sum_{c=1}^k y_{ic} \log(\hat{y}_{ic})$$

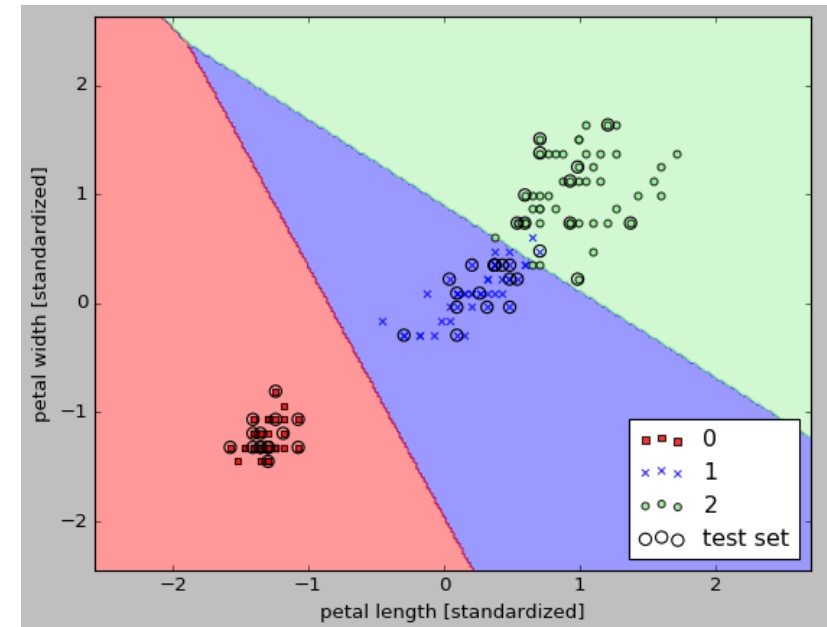
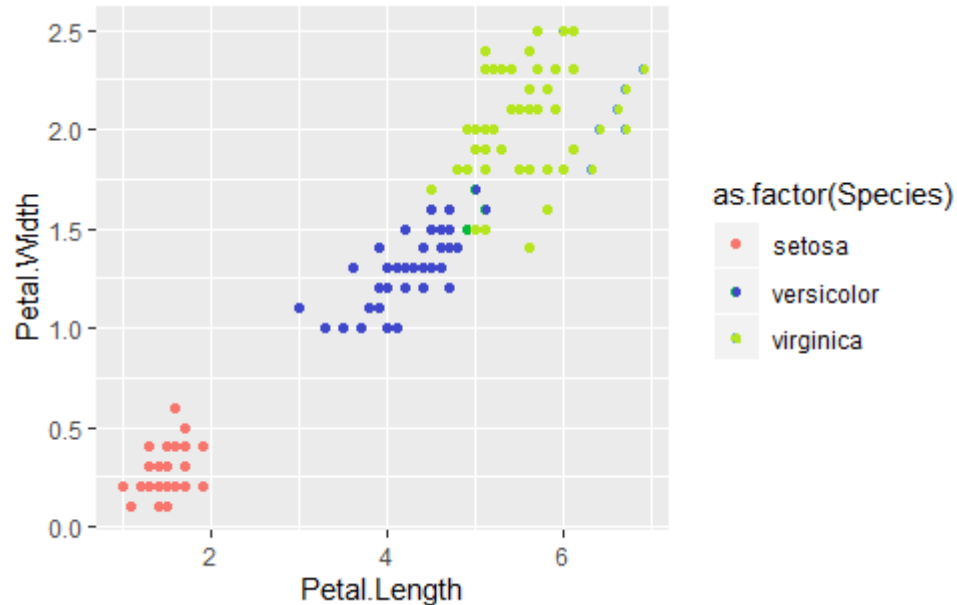
- Para $k = 2$ clases ambas fórmulas son equivalentes. **Demostración.**

Ejemplo

- Vamos a entrenar una regresión logística sobre el dataset iris para predecir Species en base a los valores de Petal.Width y Petal.Length. Es decir:

$y = \text{Species}$

$\mathbf{X} = (\text{Petal.Width}, \text{Petal.Length})$



Regresión Logística Lasso/Ridge

- Los mismos mecanismos de regularización aplicados a regresión lineal se pueden utilizar para modelos de regresión logística. Lo único que cambia es la función de coste, $l(\hat{y}_i, y_i)$.

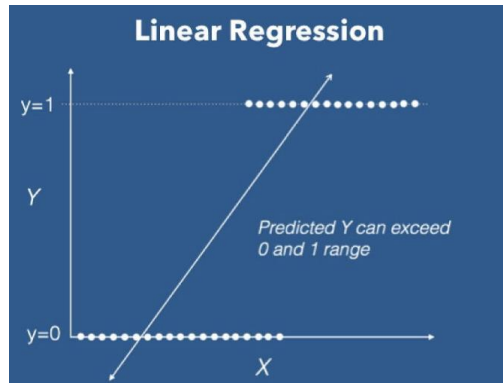
- Lasso:**

$$l(\hat{y}_i - y_i) + \lambda \sum_{i=1}^d |w_i| \implies -\sum_{c=1}^k y_{ic} \log(\hat{y}_{ic}) + \lambda \sum_{i=1}^d |w_i|$$

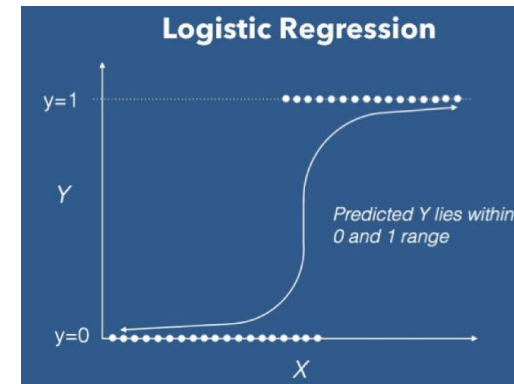
- Ridge:**

$$l(\hat{y}_i - y_i) + \lambda \sum_{i=1}^d w_i^2 \implies -\sum_{c=1}^k y_{ic} \log(\hat{y}_{ic}) + \lambda \sum_{i=1}^d w_i^2$$

- **Tipo de problemas:** ML supervisado de clasificación.
- **Intuición:** Estima probabilidades utilizando una función logística al resultado de una regresión lineal.



$$\phi(z) = \frac{1}{1 + e^{-z}}$$



- **Fórmula:** $\hat{y}_i = \frac{1}{1 + e^{-w_0 + w_1 x_{i1} + \dots + w_d x_{id}}}$
- **Regularización:** Busca evitar overfitting.

Lasso: $-\sum_{c=1}^k y_{ic} \log(\hat{y}_{ic}) + \lambda \sum_{i=1}^d |w_i|$

Ridge: $-\sum_{c=1}^k y_{ic} \log(\hat{y}_{ic}) + \lambda \sum_{i=1}^d w_i^2$

- **PROS/CONS:** Es la familia de modelos ML más **sencilla** para este tipo de problemas.







SVM

EDEM

Escuela de Empresarios



COMPARACIÓN ALGORITMOS

| | TYPE | NAME | DESCRIPTION | ADVANTAGES | DISADVANTAGES |
|----------------|-------------------------------------------------------------------------------------|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Linear |  | Linear regression | The "best fit" line through all data points. Predictions are numerical. | Easy to understand -- you clearly see what the biggest drivers of the model are. | <ul style="list-style-type: none"> ✗ Sometimes too simple to capture complex relationships between variables. ✗ Tendency for the model to "overfit". |
| |  | Logistic regression | The adaptation of linear regression to problems of classification (e.g., yes/no questions, groups, etc.) | Also easy to understand. | <ul style="list-style-type: none"> ✗ Sometimes too simple to capture complex relationships between variables. ✗ Tendency for the model to "overfit". |
| Tree-based |  | Decision tree | A graph that uses a branching method to match all possible outcomes of a decision. | Easy to understand and implement. | <ul style="list-style-type: none"> ✗ Not often used on its own for prediction because it's also often too simple and not powerful enough for complex data. |
| |  | Random Forest | Takes the average of many decision trees, each of which is made with a sample of the data. Each tree is weaker than a full decision tree, but by combining them we get better overall performance. | A sort of "wisdom of the crowd". Tends to result in very high quality models. Fast to train. | <ul style="list-style-type: none"> ✗ Can be slow to output predictions relative to other algorithms. ✗ Not easy to understand predictions. |
| |  | Gradient Boosting | Uses even weaker decision trees, that are increasingly focused on "hard" examples. | High-performing. | <ul style="list-style-type: none"> ✗ A small change in the feature set or training set can create radical changes in the model. ✗ Not easy to understand predictions. |
| Support Vector |  | SVM | Creates a hyperplane with maximum margin. | Can handle extremely complex tasks | <ul style="list-style-type: none"> ✗ Slow to train ✗ Difficult to understand |

- Introducción
- SVM: Support Vector Machines para Clasificación.
 - Definición.
 - Hiperplano de máximo margen y distancia vectores soporte.
 - Soft margin \rightarrow C parameter.
 - Kernel Trick \rightarrow gamma parameter.
- SVR: Support Vector Machines para Regresión.
 - Definición.
 - Función de coste ϵ -insensitive \rightarrow ϵ parameter.
- Hiperpárametros de las SVM.

SVM CLASIFICACION, SVC

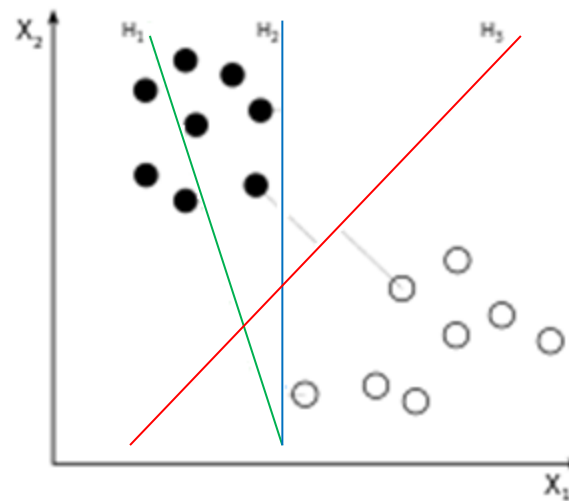


- Es un algoritmo de aprendizaje **supervisado** que sirve tanto para **clasificación como regresión**.
- Desarrollados por Vladimir Vapnik en los **años 90**.
- Originariamente diseñado para problemas de clasificación, su versión para regresión es posterior.
- Es uno de los **algoritmos más potentes** en ML.
- Se basa en la intuición geométrica de **construir el hiperplano que separe las clases** con **mayor margen** o distancia posible.

Definición

Es un clasificador en el que el objetivo es construir un **hiperplano óptimo que separe las clases**.

El hiperplano óptimo es aquel que maximiza las distancias del punto más cercano de cada clase.



Esto reduce la posibilidad de clasificar mal un nuevo patrón.

Hiperplano de máximo margen (MMH)

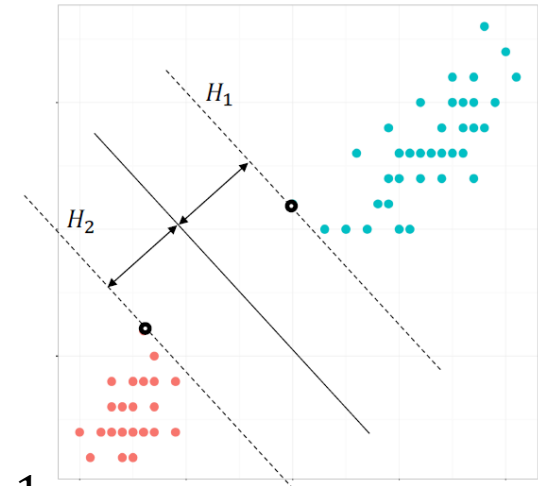
La ecuación del hiperplano que separa las clases es:

$$w^T x + b = 0$$

Los pesos, w , de la ecuación del MMH se pueden ajustar para obtener las ecuaciones de los márgenes H_1 y H_2 (maximiza la distancia a los puntos más cercanos)

$$H_1: w^T x + b \geq 1 \text{ para } y_i = +1$$
$$H_2: w^T x + b \leq -1 \text{ para } y_i = -1$$

$$y_i(w^T x + b) \geq 1 \quad \forall i$$



Los puntos que caigan **por encima de H1** serán la **clase positiva**, $y_i = +1$

Los puntos que caigan **por encima de H2** serán la **clase negativa**, $y_i = -1$

Los puntos que caigan **sobre H1 o H2** serán los **vectores soporte**, y dan nombre a este modelo.

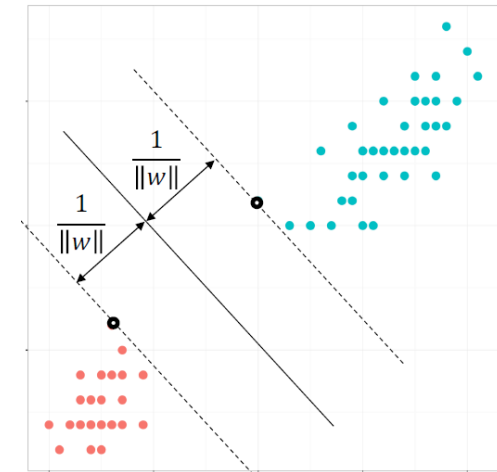
Cálculo de distancia de los vectores soporte al MMH

Por definición, la distancia de cualquier vector al hiperplano es mayor o igual a $M = \frac{1}{\|w\|}$

Donde $\|w\| = \sqrt{w_1^2 + w_2^2 + \dots + w_n^2}$ es la norma euclídea.

La distancia entre H1 y H2 será: $\frac{2}{\|w\|}$

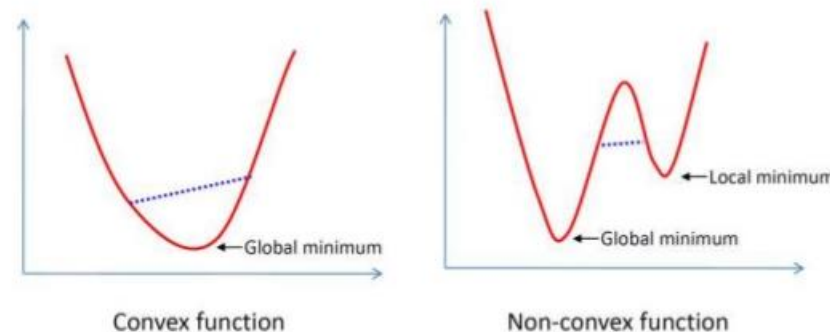
Maximizar el margen M equivale a minimizar $\|w\|$.



Por motivos de practicidad, en realidad se minimiza $\|w\|^2$ y se transforma a una **función convexa**.

Solución única

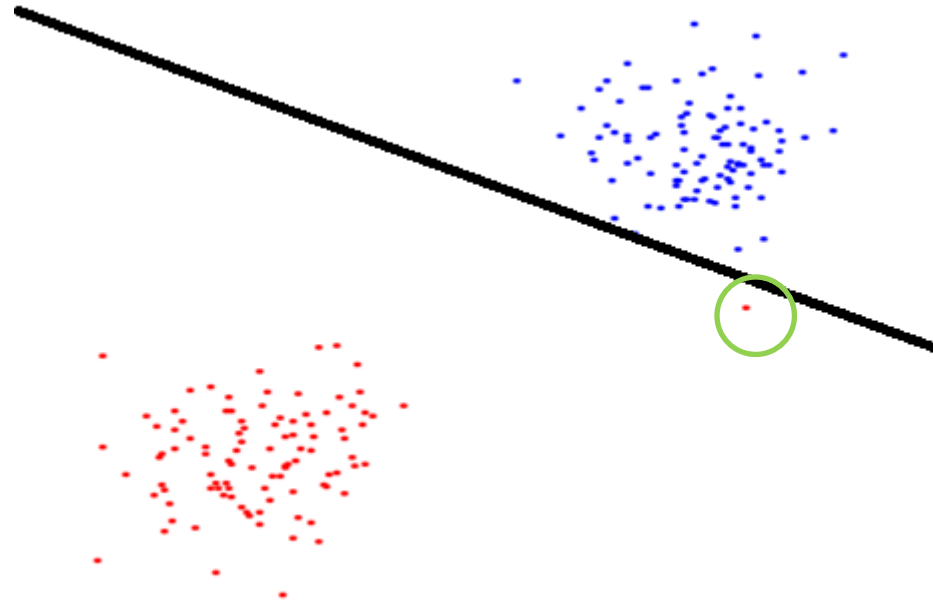
- La **solución** al problema de optimización de las SVMs tiene **garantizada su existencia y unicidad**.
- La existencia está garantizada por ser la función a minimizar, $||w||^2$, **cuadrática**.
- Además, la unicidad de la solución también es segura por tratarse de un problema de optimización **convexo**.



- Esto supone una de las principales ventajas de las SVMs frente a modelos como las ANN.

Soft Margin (I)

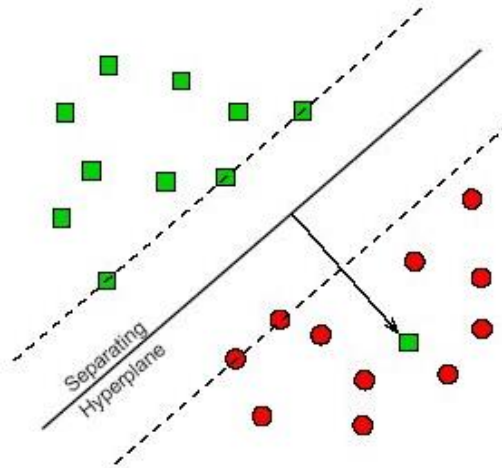
- Cuando los datos no son perfectos y tienen ruido o el problema es complejo, no podemos encontrar un separador linear perfecto.
- Incluso aunque si podamos, puede no ser recomendable debido al riesgo de overfitting.



Soft Margin (II)

- En este caso podemos crear un modelo con una buena generalización a datos nuevos aunque no todos los datos del conjunto de train se clasifiquen perfectos.

Use linear separation, but admit training errors.



- Los puntos que están en H_1 y H_2 , y **ahora también los puntos** a los que se **permite no cumplir con el margen M** , son los denominados **vectores soporte**.

Soft Margin (III)

- La capacidad de generalización del modelo se puede controlar con el **hiperparámetro C**, que nos da la **complejidad** del modelo.
- Su funcionalidad es **la inversa de λ** en la regularización Lasso o Ridge

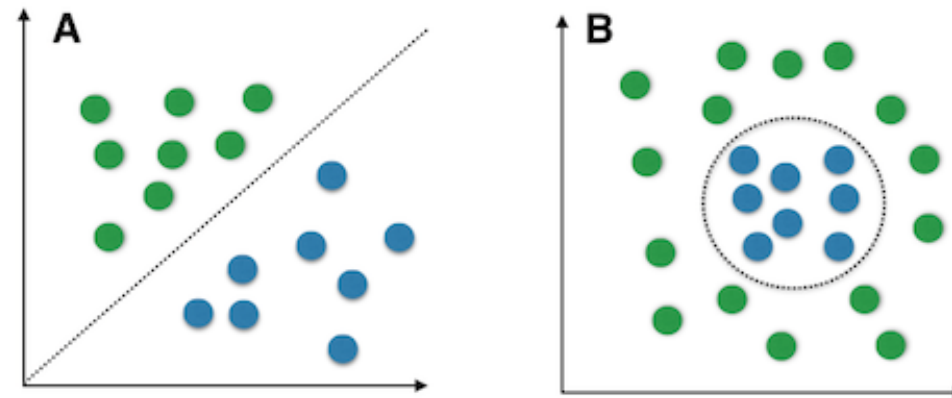
$$l(\widehat{y}_i, y_i) + \lambda \text{reg}(w) \rightarrow l(\widehat{y}_i, y_i) + \frac{1}{C} \text{reg}(w)$$

- Valores de C:
 - Un **C pequeño** (mucha regularización) puede dar un modelo con underfitting o demasiado sencillo → **High bias – Low variance**.
 - Un **C grande** (poca regularización) puede ser un modelo que no generalice bien y se ajuste demasiado al train, provocando overfitting → **Low bias – High variance**.

SVM no lineal. Kernel trick

- La SVM más sencilla solo permite separar datos linealmente separables.
- Por este motivo existen también versiones no lineales de SVM.

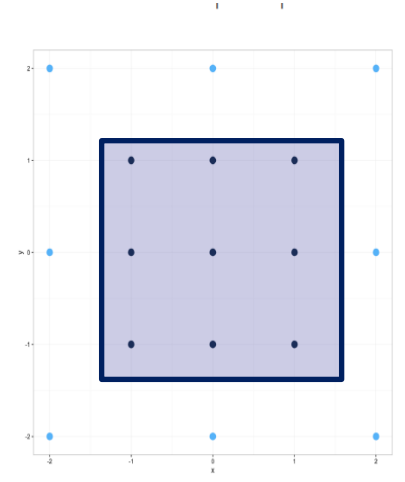
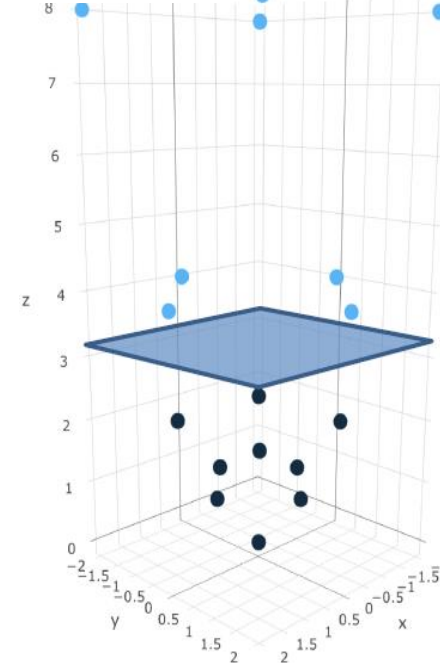
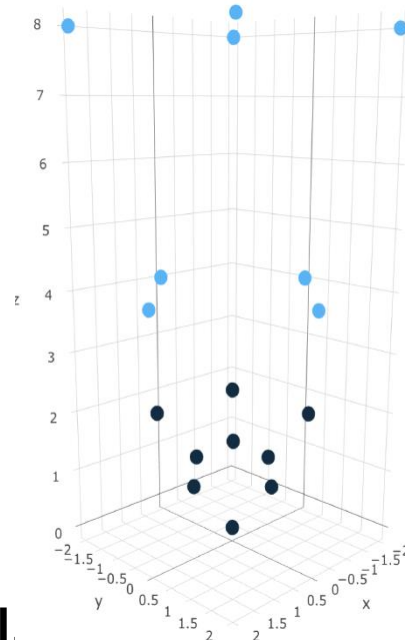
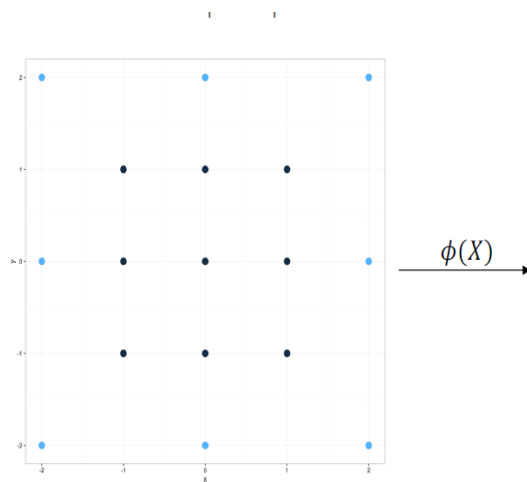
Linear vs. nonlinear problems



SVM no lineal. Kernel trick (II)

- Para conseguir esta linealidad se utiliza un recurso llamado kernel trick.
- Consiste en pasar los datos a un espacio de dimensión superior en el que se construye un hiperplano que los separe.

$$\phi_1(x) = x_1, \phi_2(x) = x_2 \text{ y } \phi_3(x) = x_1^2 + x_2^2$$



- La función Φ se denomina **kernel**.

SVM no lineal. Tipos de kernel

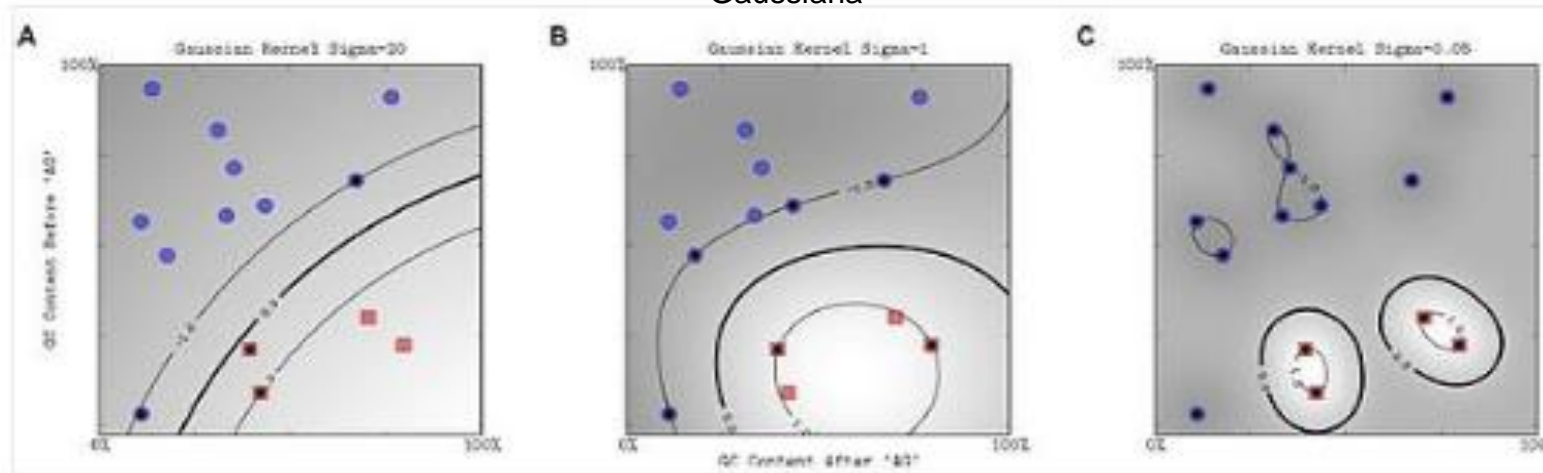
Funciones kernel utilizadas:

- Lineal: $K(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$
- Sigmoide: $K(\vec{x}_i, \vec{x}_j) = \tanh(\gamma \vec{x}_i \cdot \vec{x}_j + coef)$
- Polinómica: $K(\vec{x}_i, \vec{x}_j) = (\gamma \vec{x}_i \cdot \vec{x}_j + coef)^{grado}$
- Gaussiana (RBF): $K(\vec{x}_i, \vec{x}_j) = \exp(-\gamma |\vec{x}_i - \vec{x}_j|^2)$ *



Hiperparámetro γ

Gaussiana

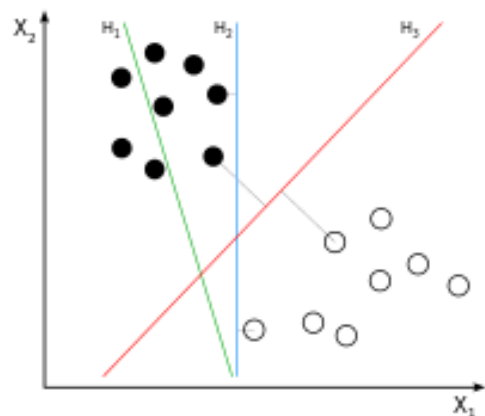


* T. Van Gestel, J. A. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, and J. Vandewalle, "Benchmarking least squares support vector machine classifiers," Machine learning, vol. 54, no. 1, pp. 5–32, 2004.

Definiciones

Es un clasificador en el que construye un hiperplano óptimo que separe las clases (H3).

Vectores soporte son los puntos mas cercanos al hiperplano y determinan el modelo.

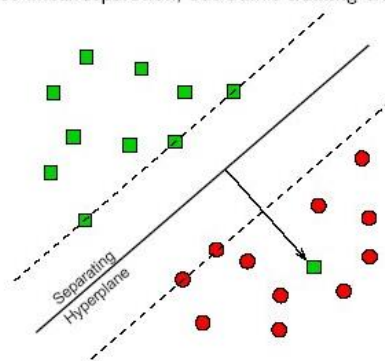


Evitar overfitting - complejidad

Permitimos clasificar “mal” algunos puntos.

La cantidad de puntos “mal” clasificados se mide con el **hiperparámetro C** y determina la complejidad del modelo

Use linear separation, but admit training errors.

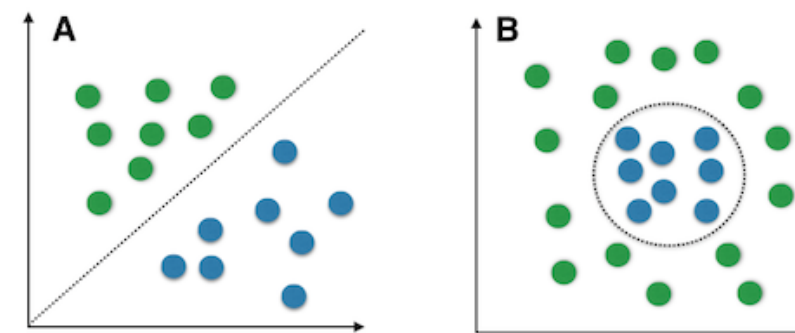


Datos no linealmente separables

Se construyen los datos en una dimensión superior en el que se construye un hiperplano que los separe.

Hiperparámetro gamma

Linear vs. nonlinear problems



La **solución** las SVMs tiene **garantizada existencia y unicidad** por ser una **función cuadrática y convexa**.

SVM REGRESION, SVR



Definición

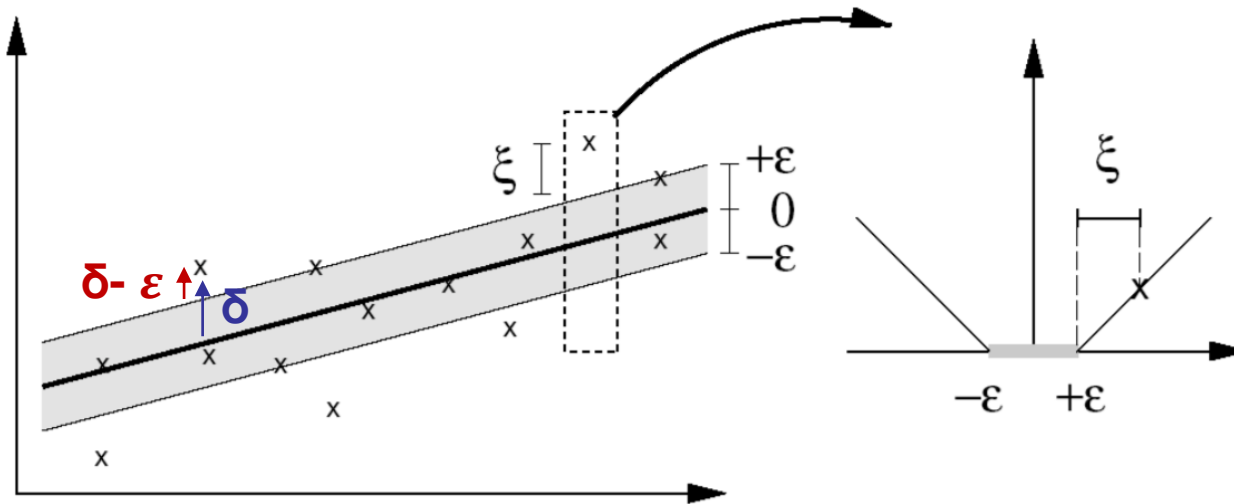
- Los modelos SVM también pueden ser aplicados a **problemas de regresión, SVR**.
- La idea intuitiva geométrica, margen máximo, no es tan evidente.
- El problema de optimización a resolver SVR es análogo al de Ridge Regression pero cambiando la función de coste, $l(\hat{y}_i, y_i)$.

$$l(\hat{y}_i, y_i) + \lambda \sum_{i=1}^d w_i^2$$

- λ se convierte en $\frac{1}{C}$
- La función de coste, $l(\hat{y}_i, y_i)$, utilizada en SVR se llama **ϵ -insensitive**.

Función de coste ε -insensitive

- El objetivo es encontrar una función con una desviación menos ε del target



$$l_\varepsilon(\delta) = \begin{cases} -\delta - \varepsilon, & \delta < -\varepsilon \\ 0, & \delta \in [-\varepsilon, \varepsilon] \\ \delta - \varepsilon, & \delta > \varepsilon \end{cases}$$

$$\delta = ERROR = pred - real$$

$$\xi = |\delta| - \varepsilon$$

- ε es un nuevo Hiperparámetro a optimizar, solo necesario en problemas de regresión.

Optimización de los hiperparámetros

- Una de las principales ventajas de las SVM es que tenemos pocos parámetros a optimizar.
 - 1 en el caso de la **SVM lineal para clasificación** $\rightarrow C$
 - 2 en el caso de **SVM no lineal con Kernel Trick en clasificación** $\rightarrow C, \gamma$
 - 3 en el caso de **SVR no lineal con Kernel Trick en regresión** $\rightarrow C, \gamma, \epsilon$
- Esto lo convierte en un algoritmo fácil de optimizar por tener pocos hiperparámetros, con respecto a otros algoritmos complejos, como ANN.
- La existencia de solución y unicidad, ya que el mínimo es global, es una de las mayores ventajas al optimizar los parámetros.

Optimización de los hiperparámetros (II)

- Al tratarse de pocos hiperparámetros es muy fácil programar una rejilla para optimizarlos.
- Aunque no existan reglas que los definan, **Cherkassky** propuso una solución estadística que tiene en cuenta la distribución y volumen de los datos del problema.

- **Parámetro C**

$$\max(|\bar{y} + 3\sigma_y|, |\bar{y} - 3\sigma_y|)$$

Donde \bar{y} es la media del target y σ_y es la desviación típica del target.

- **Parámetro ε**

$$\varepsilon = \frac{\sigma}{\sqrt{n}} \quad \text{y para un conjunto de datos con muchos patrones } \varepsilon = 3\sigma \sqrt{\frac{\log n}{n}}$$

- **Parámetro γ**

para valores escalados de 0 a 1 $\sigma^d = (0.2, 0.5)$

Ventajas y desventajas

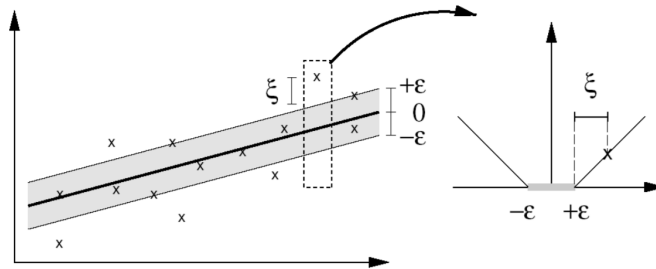
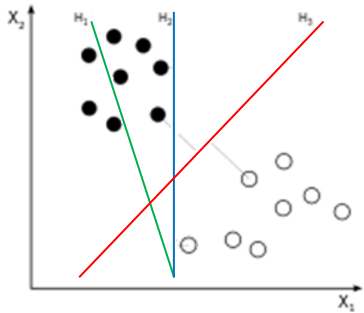
- **Ventajas:**

- Las SVM son un buen algoritmo en problemas con **datos muy complejos**.
- El kernel trick permite hacer modelos muy complejos y resolver **problemas no lineales**.
- A diferencia de las redes neuronales, el problema de optimización no tiene mínimos locales, **el óptimo siempre es global**.
- Tiene **pocos hiperparámetros** a optimizar por lo que metaparametrización es más sencilla y rápida.
- Escala bien en memoria para datos con muchas características comparado con otros modelos complejos, gracias a que utiliza un subconjunto de puntos de entrenamiento en la función de decisión (vectores de soporte).

Ventajas y desventajas (II)

- **Desventajas:**
 - La intuición para entender los hiperparámetros, especialmente el relativo al kernel, es más compleja que en modelos como la regresión lineal o logística.
 - El entrenamiento es costoso computacionalmente a pesar de que solo use los vectores soporte comparado con modelos sencillos.
 - Es una “caja negra” y tiene una **interpretación analítica más compleja** que otros modelos más sencillos como la regresión lineal o los árboles de decisión.

SVM VS SVR



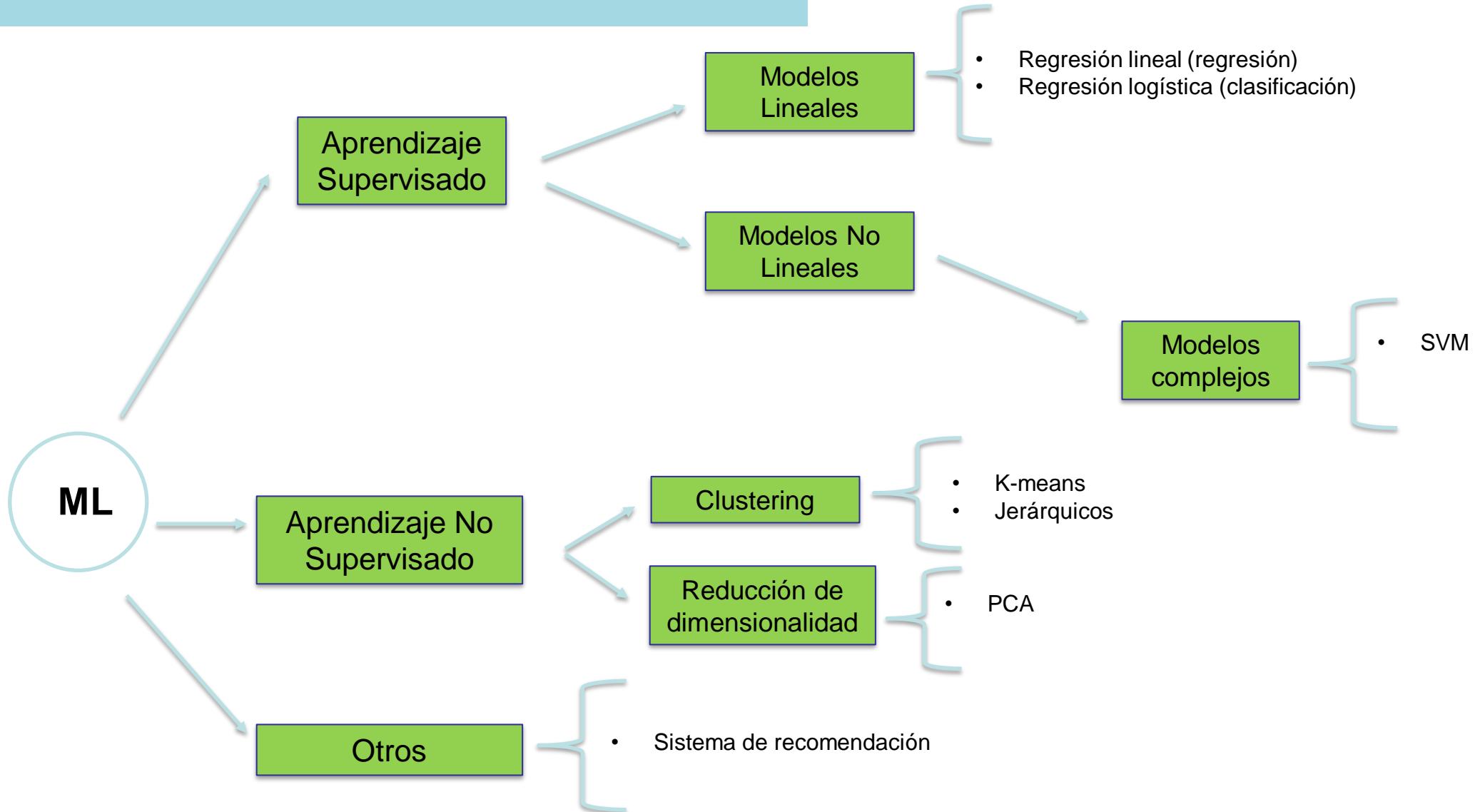
Hiperpámetros Cherskarsky

| Parámetro | Fórmula | Función |
|-----------|---------------------------------------------------------------------------------------|----------------------------|
| C | $\max(\bar{y} + 3\sigma_y , \bar{y} - 3\sigma_y)$ | Complejidad |
| Gamma | $\sigma^d = (0.2, 0.5)$ | Kernel (datos no lineales) |
| Epsilon | $\varepsilon = \frac{\sigma}{\sqrt{n}}, \varepsilon = 3\sigma\sqrt{\frac{\log n}{n}}$ | Regresión |

Ventajas VS desventajas

- **Algoritmo complejo:** Funciona bien en datos no lineales.
- **No hay mínimos locales:** solución existe y es única.
- **Fácil optimización:** Hay pocos parámetros y tenemos la definición de Cherskarsky.
- **Escala bien en memoria:** por lo vectores soporte.
- Aunque es una **caja negra y difícil de tener una intuición gráfica y explicar.**
- **Es muy sensible a los hiperparámetros utilizados.**
- **Costoso computacionalmente** para un alto volumen de datos.

RESUMEN MODELOS



EDEM

Escuela de Empresarios

EDEM

Escuela de Empresarios

Yvonne Gala García:
yvonne.gala@iberiaexpress.com

Jesús Prada Alonso:
ext.jprada@iberiaexpress.com