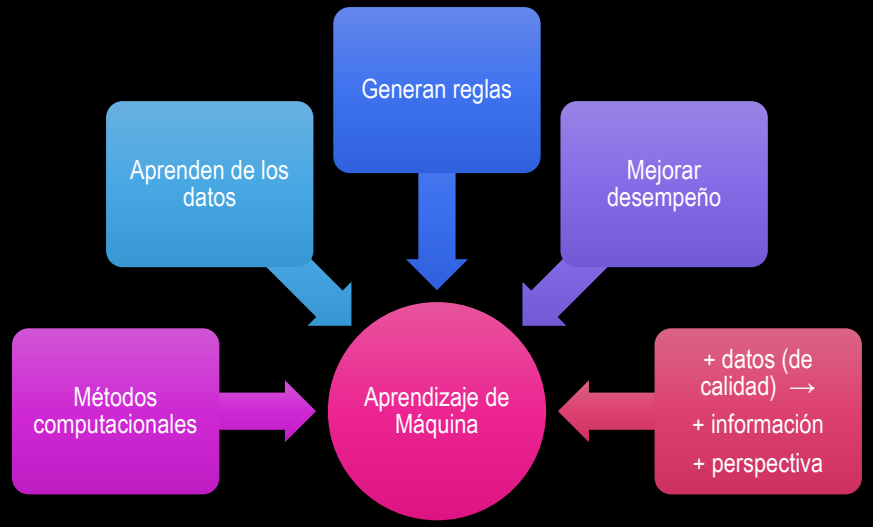
Tema 3:
Conceptos básicos de Aprendizaje de Máquina

Parte 3

Recordemos



Ejemplos (¿Dónde aplicar ML?):

- **Determinar** si dar un préstamo o no (algoritmos, en vez de evaluaciones personales).
- **Reconocer** placas de un coche de una fotografía.
- **Detectar** llamados de ballenas en grabaciones de boyas.*
- **Estimar** la demanda de productos de una compañía.
- **Dividir** a los clientes de Netflix según sus gustos.
- **Recomendar** productos a clientes.

¿Por qué usar ML?

- Respuesta barata, rápida, automatizada, y con suficiente precisión.
- Superar el desempeño actual de los expertos o de reglas simples.
- Entender de manera más completa y sistemática el comportamiento de un fenómeno, identificando variables o patrones importantes.

Algoritmos de Aprendizaje Supervisado

Aprendizaje Supervisado

Construir un modelo para predecir o estimar una *variable de salida* a partir de ciertas variables de entrada.

Objetivo general: Hacer las mejores predicciones bajo ciertos supuestos y restricciones.

Problemas de Regresión: cuando la variable respuesta es numérica.

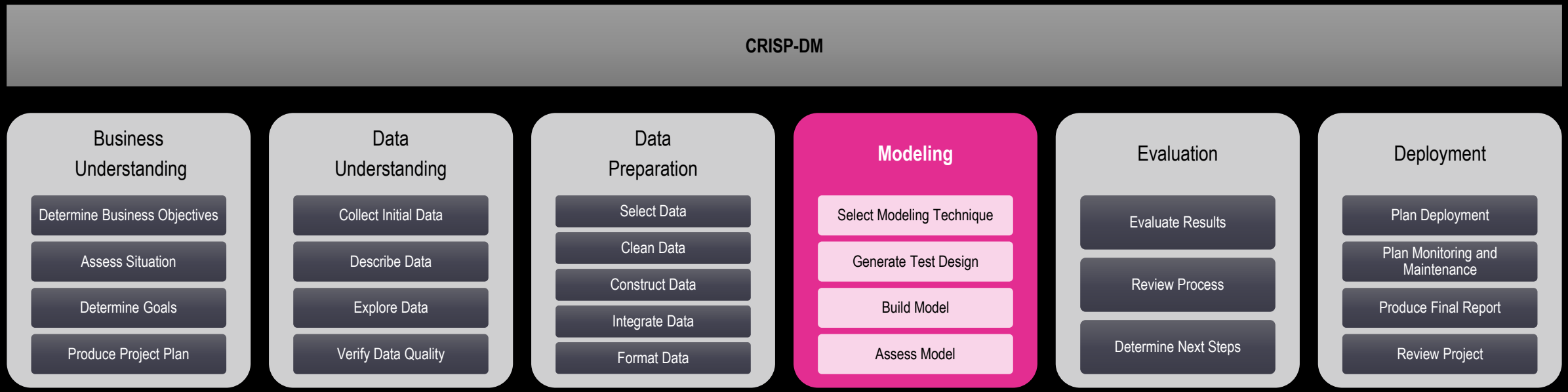
Ejemplo: Proyecto Final de la Materia: Predecir el número de unidades que se van a vender.

Problemas de Clasificación: cuando la variable respuesta es categórica.

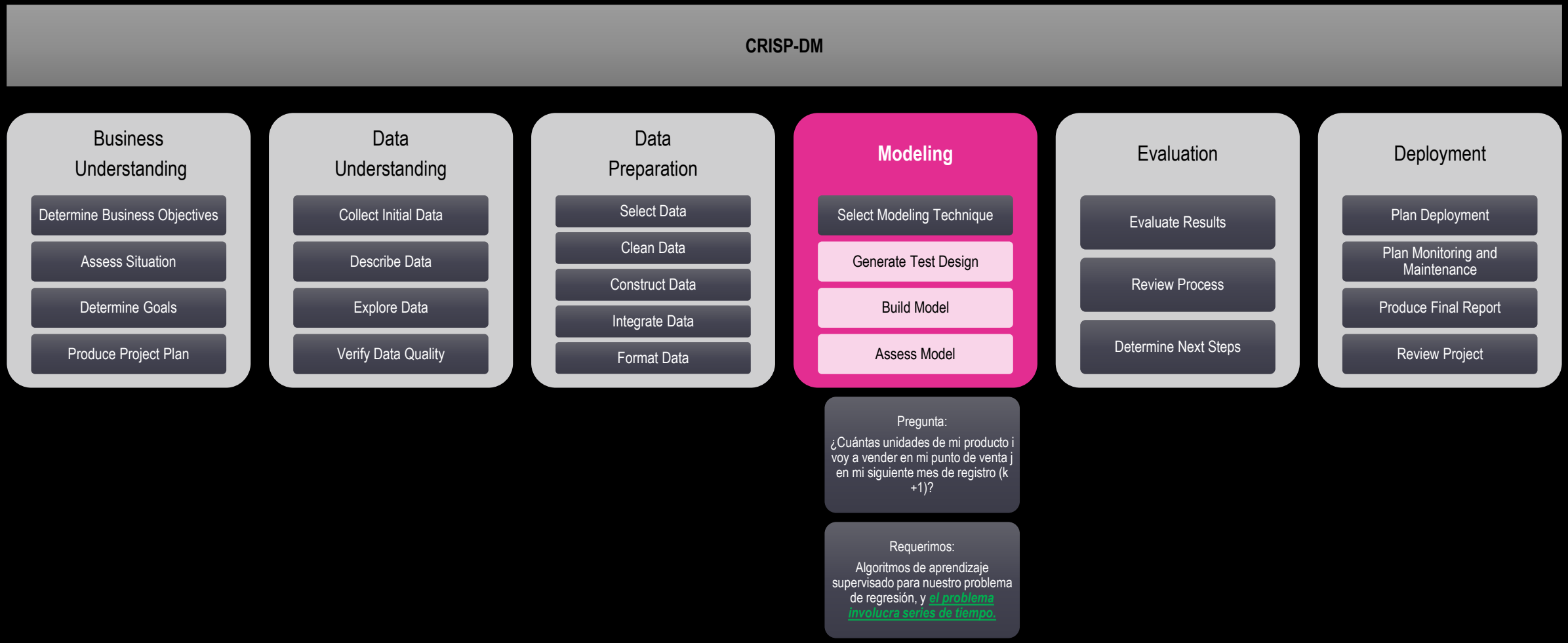
Ejemplo: Detección de números escritos a mano.

Ejemplo: Clasificación de clientes.

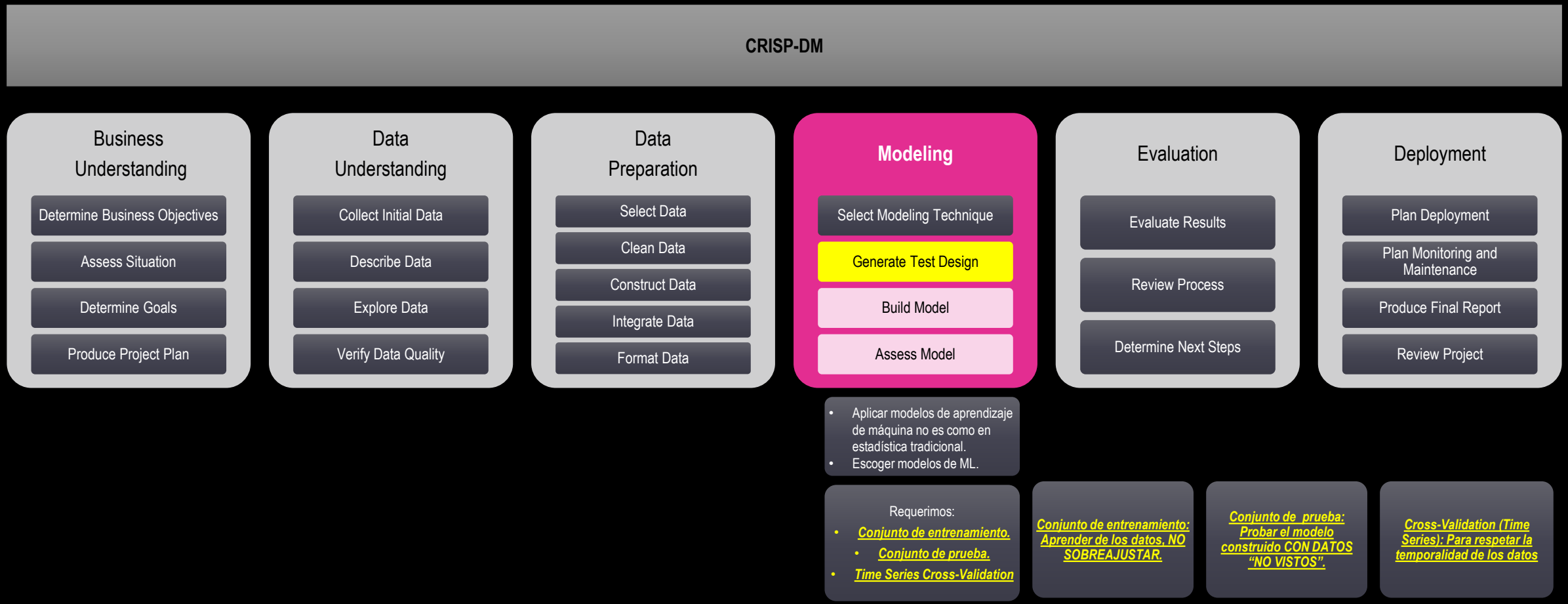
CRISP – DM


CRISP-DM
Business Understanding
Determine Business Objectives
Assess Situation
Determine Goals
Produce Project Plan
Data Understanding
Collect Initial Data
Describe Data
Explore Data
Verify Data Quality
Data Preparation
Select Data
Clean Data
Construct Data
Integrate Data
Format Data
Modeling
Select Modeling Technique
Generate Test Design
Build Model
Assess Model
Evaluation
Evaluate Results
Review Process
Determine Next Steps
Deployment
Plan Deployment
Plan Monitoring and Maintenance
Produce Final Report
Review Project

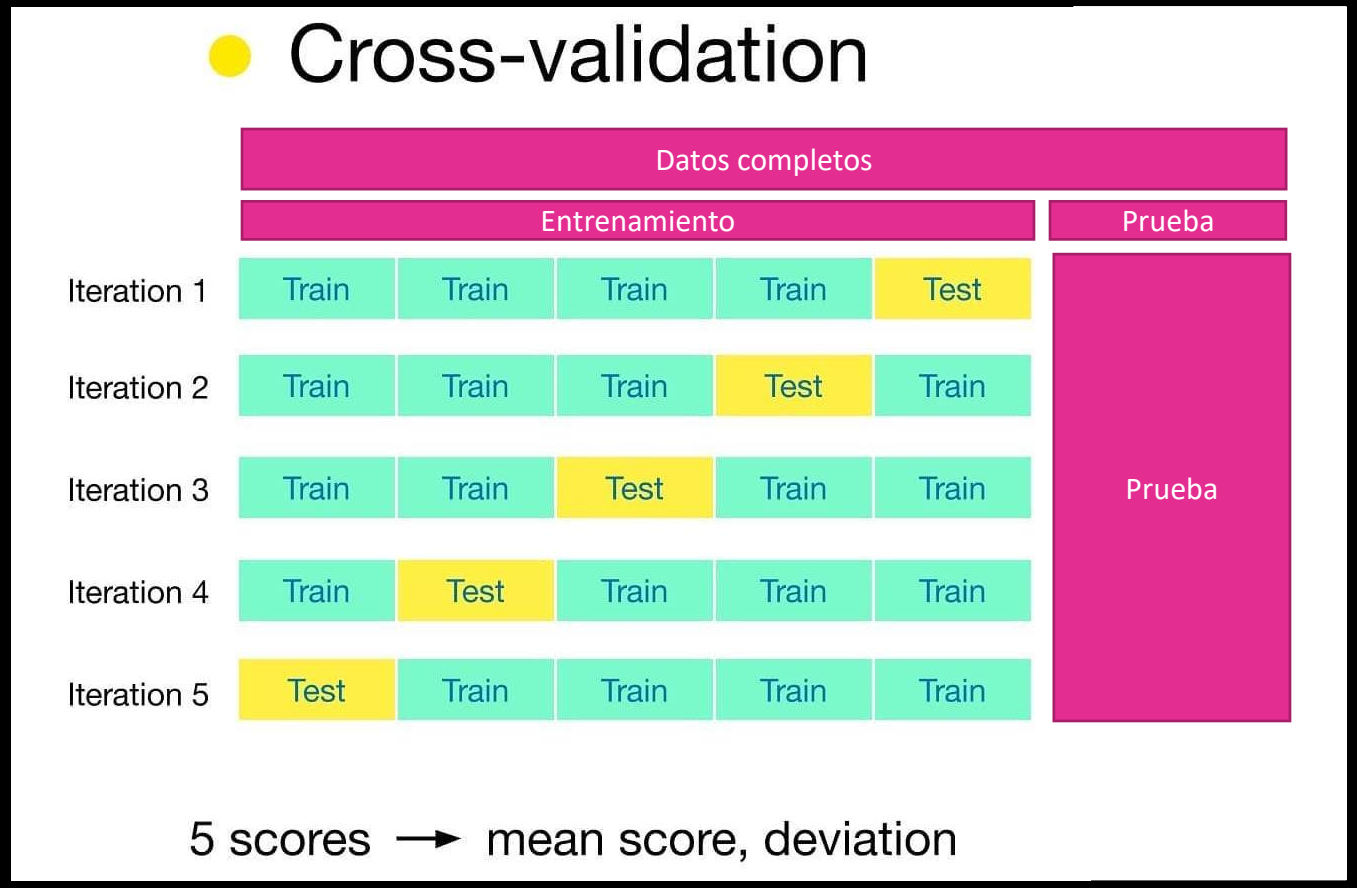
CRISP – DM


CRISP-DM
Business Understanding
Determine Business Objectives
Assess Situation
Determine Goals
Produce Project Plan
Data Understanding
Collect Initial Data
Describe Data
Explore Data
Verify Data Quality
Data Preparation
Select Data
Clean Data
Construct Data
Integrate Data
Format Data
Modeling
Select Modeling Technique
Generate Test Design
Build Model
Assess Model
Evaluation
Evaluate Results
Review Process
Determine Next Steps
Deployment
Plan Deployment
Plan Monitoring and Maintenance
Produce Final Report
Review Project
Pregunta:
¿Cuántas unidades de mi producto i voy a vender en mi punto de venta j en mi siguiente mes de registro (k +1)?
Requerimos:
Algoritmos de aprendizaje supervisado para nuestro problema de regresión, y el problema involucra series de tiempo.

CRISP – DM


CRISP-DM
Business Understanding
Determine Business Objectives
Assess Situation
Determine Goals
Produce Project Plan
Data Understanding
Collect Initial Data
Describe Data
Explore Data
Verify Data Quality
Data Preparation
Select Data
Clean Data
Construct Data
Integrate Data
Format Data
Modeling
Select Modeling Technique
Generate Test Design
Build Model
Assess Model
Evaluation
Evaluate Results
Review Process
Determine Next Steps
Deployment
Plan Deployment
Plan Monitoring and Maintenance
Produce Final Report
Review Project
• Aplicar modelos de aprendizaje de máquina no es como en estadística tradicional.
• Escoger modelos de ML.
Requerimos:
• Conjunto de entrenamiento.
• Conjunto de prueba.
• Time Series Cross-Validation
Conjunto de entrenamiento: Aprender de los datos, NO SOBREAJUSTAR.
Conjunto de prueba: Probar el modelo construido CON DATOS "NO VISTOS".
Cross-Validation (Time Series): Para respetar la temporalidad de los datos

Cross-Validation



Cross-Validation / Validación cruzada:

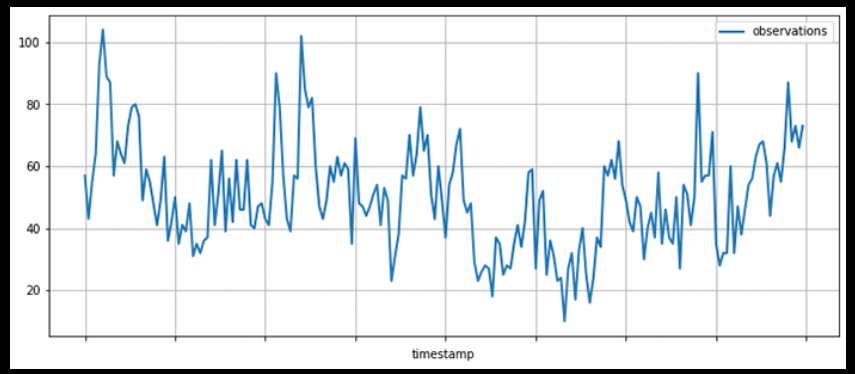
- Queremos evitar sobreajuste.
- Cross-Validation: Método computacional para producir una estimación interna del error de predicción (usando sólo muestra de entrenamiento).
  - K vueltas y se construyen al azar particiones con tamaños similares.
  - K pequeño: modelos construidos con pocos datos (comparado al modelo con todos los datos de entrenamiento).
  - K grande: Costoso (por ejemplo, si k=N, hay que entrenar un modelo para cada dato de entrada).

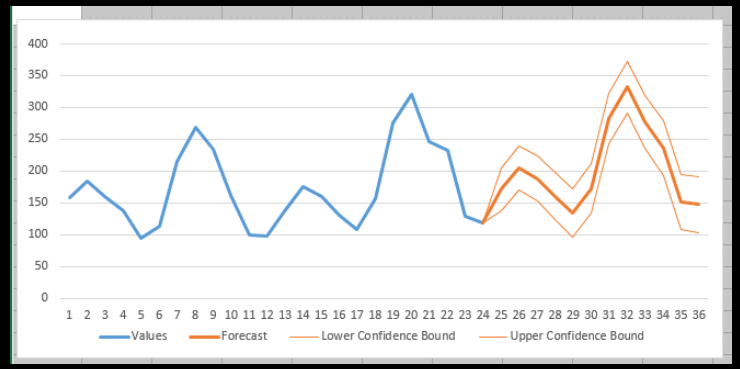
¿Qué pasa cuando tenemos un problema que involucra series de tiempo?
¿Puedes particionar los datos de esta manera (al azar)?

Time Series Cross-Validation

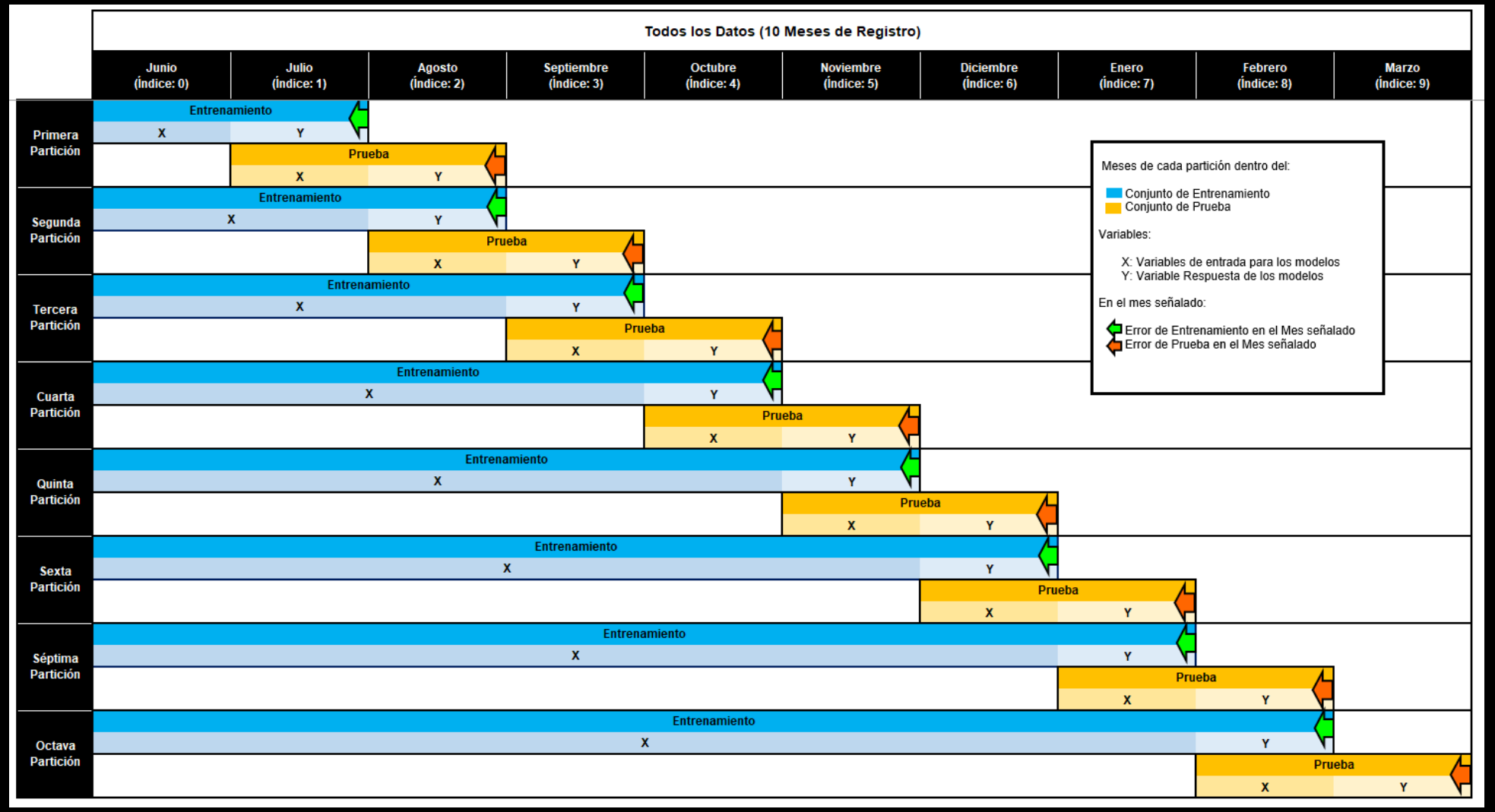
Cross-Validation / Validación cruzada (PARA SERIES DE TIEMPO):

- Ya se determinó el modelo base: Promedios móviles.
- Queremos proponer otros modelos (de aprendizaje de máquina) y comparar desempeños.
- Construcción modelos tradicionales ≠ Construcción de Modelos de Aprendizaje de máquina.
- Queremos evitar sobreajuste.
- **¡Importante!** Cuando los registros tienen una **secuencia periódica** que respetar, se debe de aplicar **Validación Cruzada para Series de Tiempo** con el fin de **evitar el sobreajuste de los modelos** y **comprobar su efectividad en observaciones no vistas con anterioridad**.
- La idea general de esta técnica consiste en **particionar los datos en varios conjuntos de entrenamiento y prueba**, **respetando su secuencia temporal** [10].





Time Series Cross-Validation



En el caso de este proyecto:

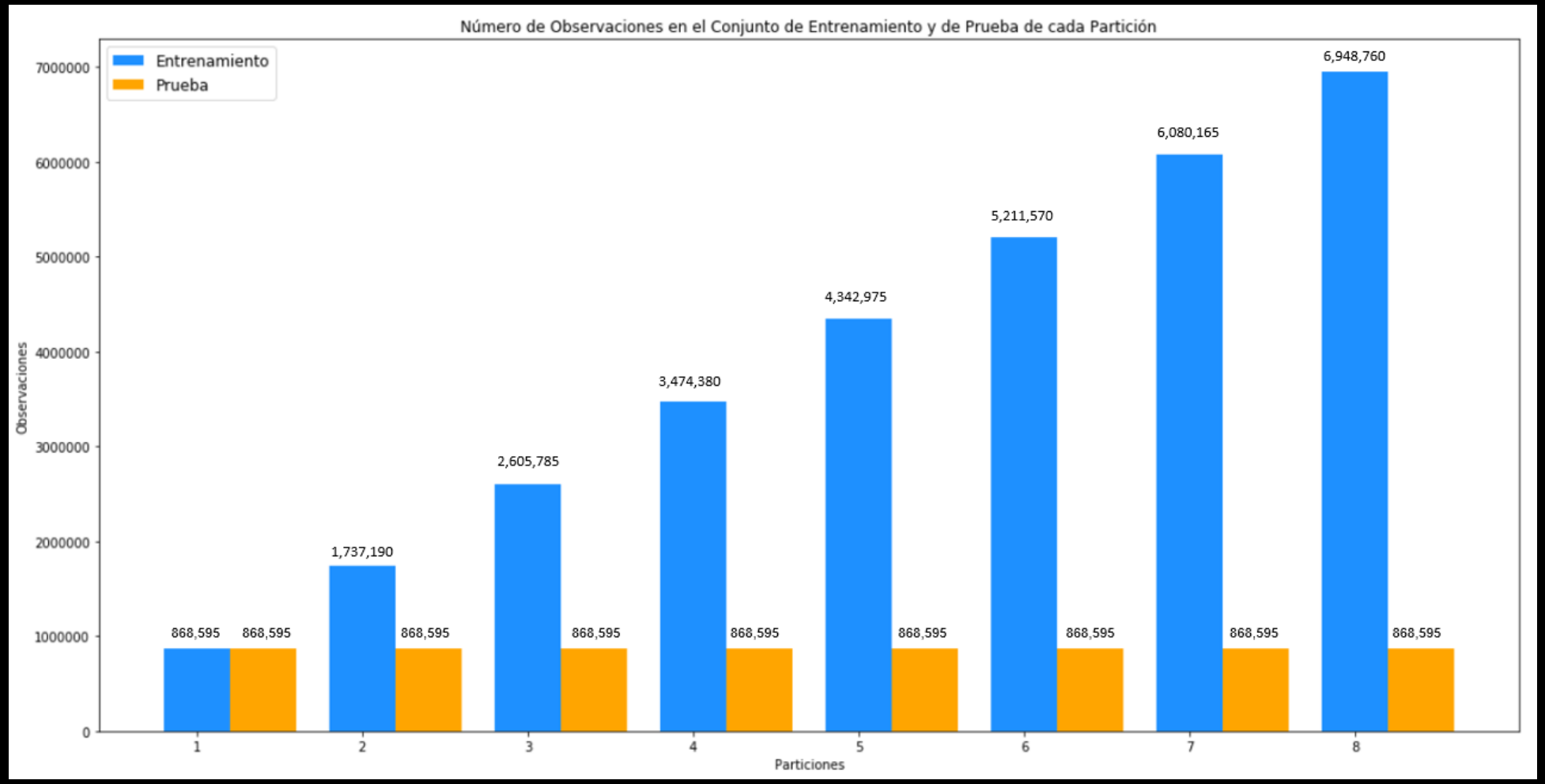
- 8 particiones. Cada una con su respectivos conjuntos de entrenamiento y prueba.

Con cada mes que pasa:

- Entrenamiento: Crece. Conforme pasen los meses, los algoritmos propuestos van a tener una mayor ingesta de datos buscando aprender cada vez más de éstos.
- Prueba: en todas las particiones corresponde a un mes únicamente y éste se va recorriendo conforme pasa el tiempo.

En cada partición se van a calcular dos errores, el error de entrenamiento y el error de prueba, **sin embargo, el más importante a tomar en cuenta va a ser el de prueba.**

Time Series Cross-Validation



Time Series Cross-Validation

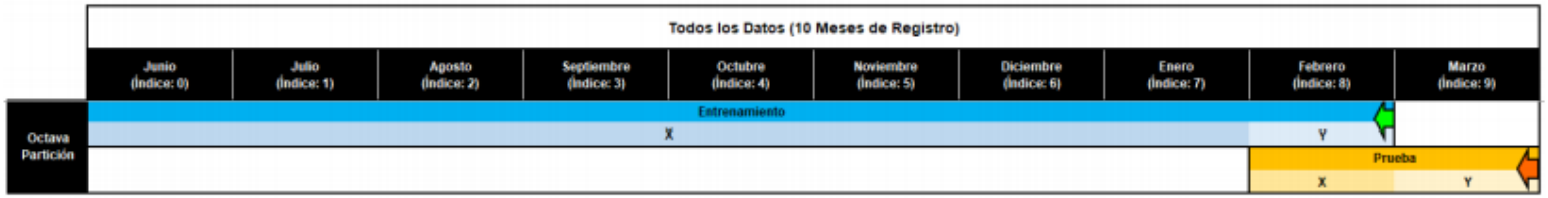


Figura 12: Validación Cruzada para Series de Tiempo de la última Partición de los Datos.

En esta partición se puede observar:

- **Número de registros:** Esta partición cuenta con 6,948,760 registros en el conjunto de entrenamiento, y 868,565 registros en el conjunto prueba.
- **X Entrenamiento:** Las variables de entrada (33) para entrenar el modelo que abarcan los registros de junio a enero.
- **Y Entrenamiento:** La variable de salida con la que se va a evaluar el conjunto de entrenamiento, corresponde a las ventas del mes de febrero (se calcula error de entrenamiento en el mes de febrero).
- **X Prueba:** Las variables de entrada (33) del conjunto de prueba que abarcan los registros del mes de febrero.
- **Y Prueba:** La variable de salida con la que se va a evaluar el conjunto de prueba corresponde a las ventas del mes de marzo (se calcula error de prueba en el mes de marzo).

Modelos (Regresión)

Proceso generador de datos:

Para construir propuestas de modelos que permitan lidiar mejor con la problemática del proyecto, es necesario desglosarlo para entenderlo mejor y así su planteamiento sea más sencillo.

1. Problema que requiere de modelos de aprendizaje supervisado.
2. Problema de regresión.
3. Problema de series de tiempo (datos contienen observaciones correspondientes a ventas a lo largo de 10 meses, valores que están secuencialmente ordenados y no pueden ser permutados).

Ya que se definió el problema de esta manera, se puede tomar como base el ***proceso generador de datos (modelo teórico)*** [3]. De este proceso se puede rescatar que:

- Y es la variable que se busca predecir;
- X es una variable, o conjunto de variables, que se espera que pueda mejorar la predicción de Y ;
- Y y X están relacionadas de la siguiente manera: $Y = f(X) + \varepsilon$

Donde: f expresa la relación sistemática que hay entre X y Y , y representa el efecto de variables que no se han medido o procesos aleatorios que determinan la respuesta.

En problemas de Aprendizaje de Máquina, f no se conoce, por ende, hay que estimarla $\hat{f}$ con ayuda de los datos (aprender de los datos).

1. Dividir los datos en conjuntos de entrenamiento y de prueba.
2. Revisar el desempeño de los modelos considerando métricas que permita comparar el valor real con el de las predicciones [4].

La idea general de un algoritmo de aprendizaje de máquina es aprender de una muestra de entrenamiento y así generar una $\hat{f}$ que permita hacer predicciones $\hat{Y} = \hat{f}(X)$.

Con respecto al proyecto:

$Yi_{j,k+1}$: Número de unidades del producto con índice i en el punto de venta con índice j que se van a vender en el mes siguiente del mes de registro k. (Definir i,k y k dependiendo de cada marca)

Lo que busca este proyecto es estimar el valor de la variable Yi,j,k+1 :, por lo tanto, se quiere calcular:

$\hat{Y}$ i,j,k+1 : Predicción del número de unidades del producto con índice i en el punto de venta con índice j que se van a vender en el mes siguiente del mes de registro k.

Modelos (Regresión)

¿Cómo estimar $\hat{f}$ para que nos permita estimar estimar $\hat{Y}$ y así contentas nuestra pregunta?

– Con modelos de aprendizaje de máquina

- Modelos de Regresión

Modelos de aprendizaje para problemas de regresión:

- Regresión Lineal
- K vecinos más cercanos
- Redes Neuronales
- Árboles de Decisión
- Bosques Aleatorios
- XGBoost

Modelos (Regresión)

- Modelos de Regresión
  - Regresión Lineal (Ejemplo)
    - X: Variables de Entrada (X1, X2, … Xp)
      Y: Variable Respuesta.
    - Forma sencilla de intentar predecir Y en función de X's.
    - $f_\beta(x)$: $\beta_0 + \beta_1 X_1 + \ldots + \beta_P X_P$
    - Dada una muestra de entrenamiento, hay que encontrar valores β_s apropiados de para construir nuestro predictor: $\widehat{f_\beta(x)}$ *para hacer predicciónes de* $\hat{Y}$
  - K vecinos más cercanos
  - Árboles de Decisión
  - Bosques Aleatorios

Razones para tener malas predicciones: Sesgo y Varianza.

Sesgo (subajuste) : Qué tan lejos en promedio están las estimaciones de nuestro modelo del verdadero valor

Varianza (sobreajuste): qué tanto varían las estimaciones del modelo.

Penalización: **Ridge** y Lasso

Para que coeficientes caigan en rangos más aceptables.

La idea de regresión penalizada consiste en estabilizar la estimación de los coeficientes, especialmente en casos donde tenemos muchas variables en relación a los casos de entrenamiento. La penalización no permite que varíen tan fuertemente los coeficientes.

Parámetros (Alpha)

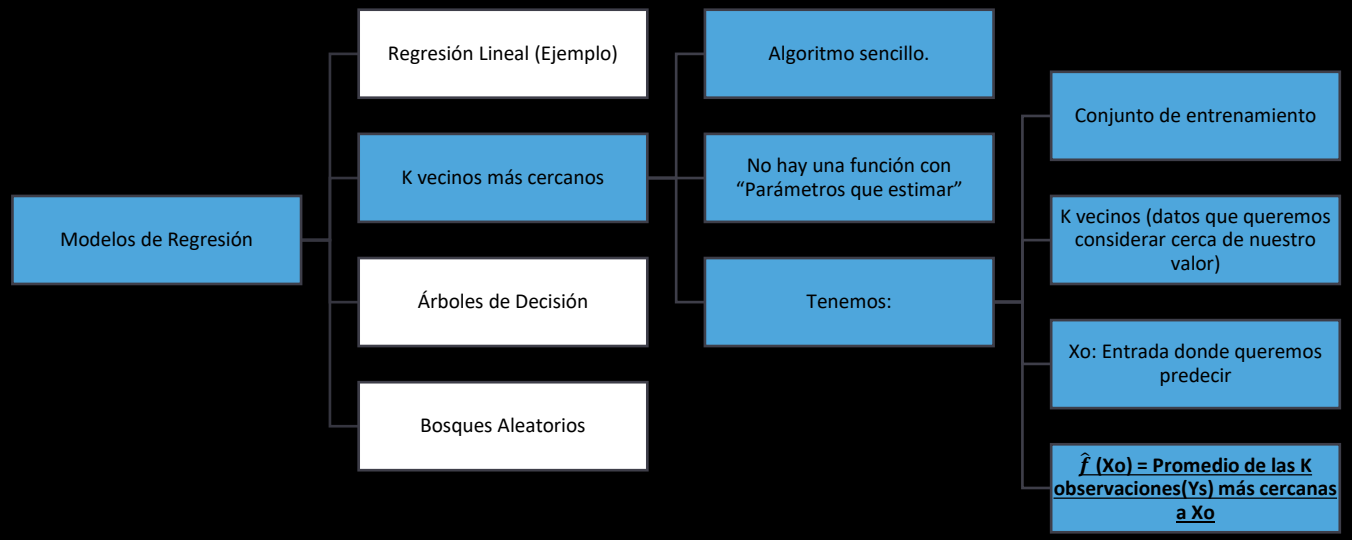
En R:

```
library(glmnet)
mod_restringido <- glmnet(x =, y =,
  alpha = 0, …)
```

En Jupyter:

https://towardsdatascience.com/how-to-perform-lasso-and-ridge-regression-in-python-3b3b75541ad8

Modelos (Regresión)



Parámetros (k)

En R:

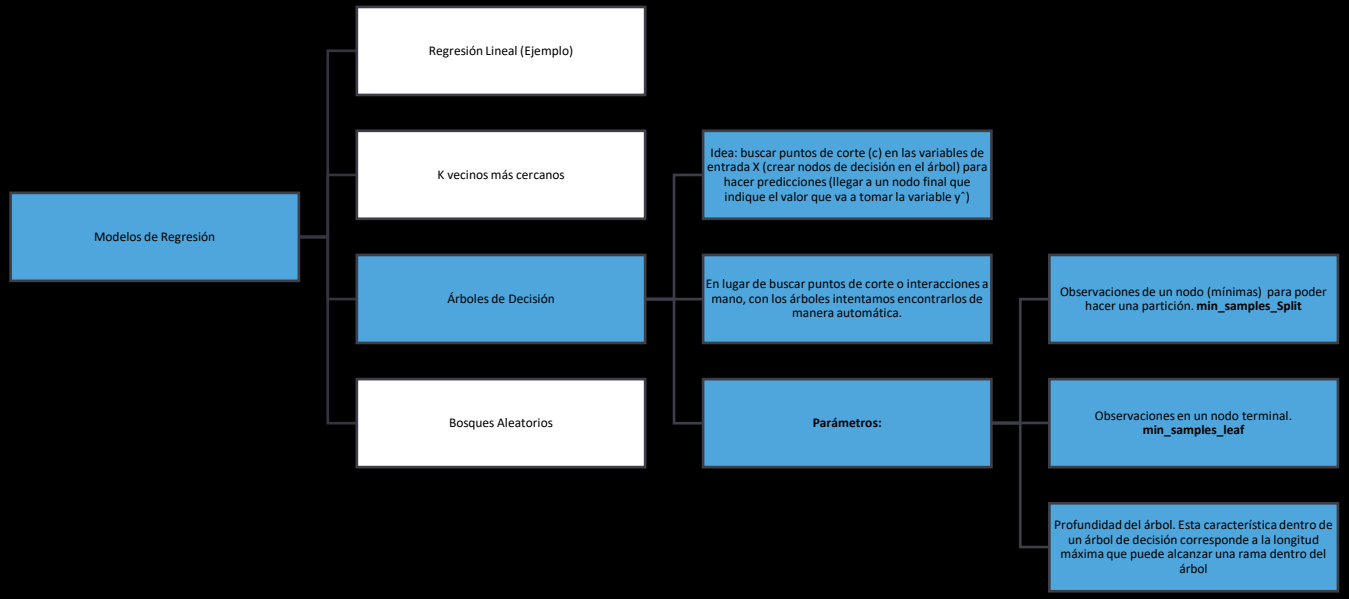
```
library(kknn)
mod_15vmc <- kknn(rendimiento_kpl ~ peso_kg, train = datos_entrena,
                  test = data_frame(peso_kg=seq(700,2200, by = 10)),
                  k=15)
```

¡Importante! Normalizar - las variables con escalas más grandes dominan el cálculo

En Jupyter:

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html

Modelos (Regresión)



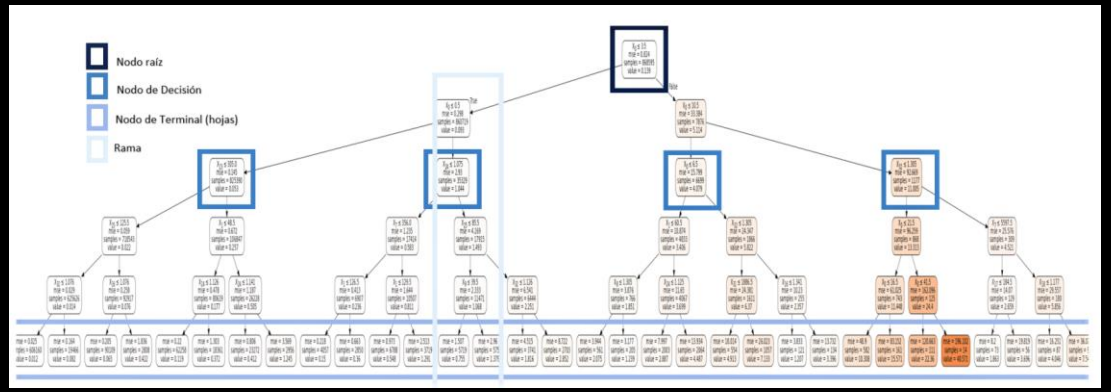
Parámetros (varios: profundidad de árbol, min_samples_Split, min_samples_leaf)

En R:
https://www.statmethods.net/advstats/cart.html

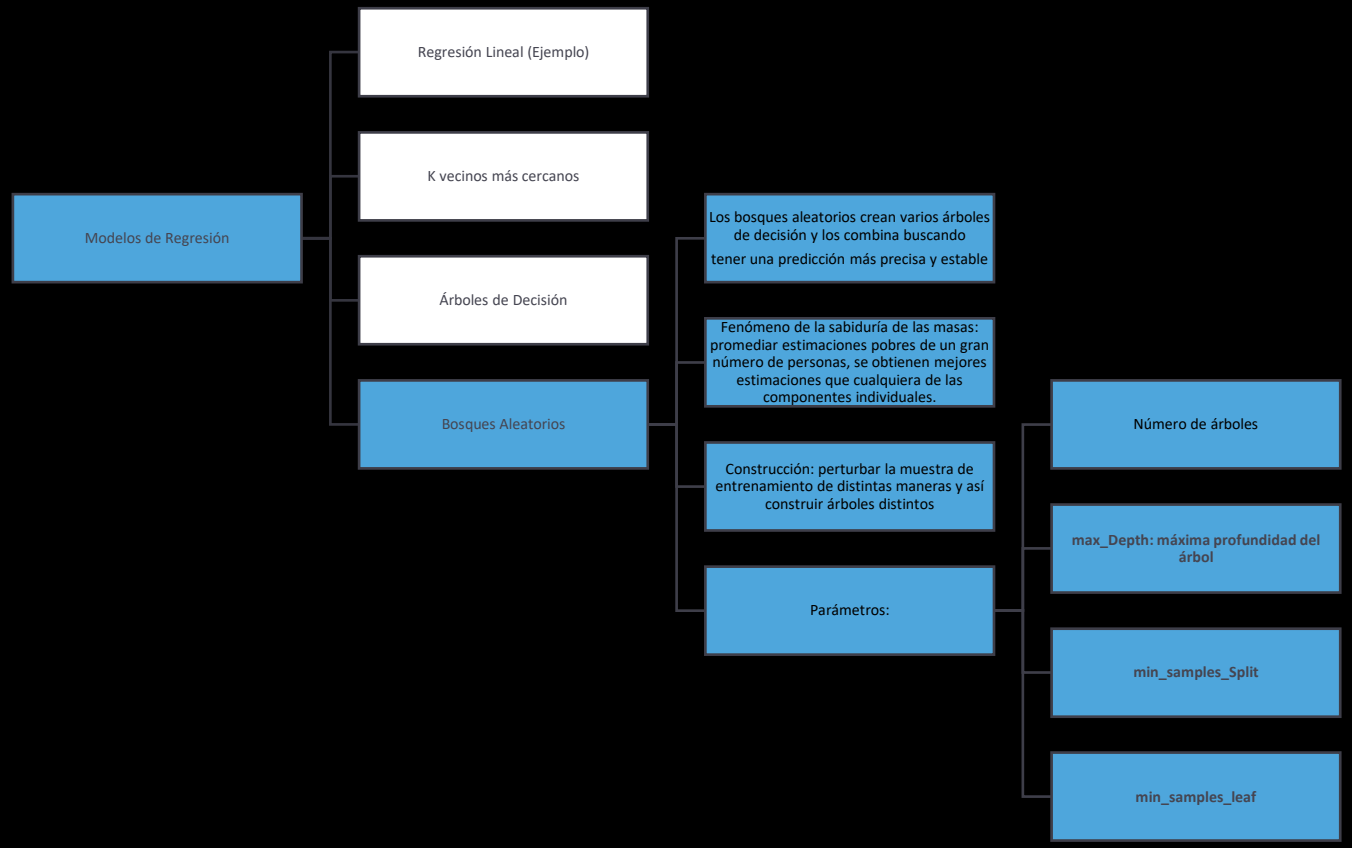
En Jupyter:
https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

```
from sklearn import tree
#Entrenamiento
regr_1 = tree.DecisionTreeRegressor(max_depth=1)
regr_1.fit(x_1, y_1)
pred_1_entrenamiento = regr_1.predict(x_1) #calcula error de entrenamiento

#Prueba: probar modelo creado (regr_1) con datos de prueba
pred_1 = regr_1.predict(prueba_x_1) #falta calcular el error de prueba
```



Modelos (Regresión)



Parámetros (varios: número de árboles, profundidad de árbol, min_samples_Split, min_samples_leaf)

En R:
https://www.geeksforgeeks.org/random-forest-approach-for-regression-in-r-programming/

En Jupyter:
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

```
from sklearn.ensemble import RandomForestRegressor
#Entrenamiento
regr_1 = RandomForestRegressor(
    n_estimators=100,
   # n_jobs=-1
)
regr_1.fit(x_1, y_1)
pred_1_entrenamiento = regr_1.predict(x_1) #Calcular errores de entrenamiento

#Prueba
pred_1 = regr_1.predict(prueba_x_1) #Calcular errores de prueba
```

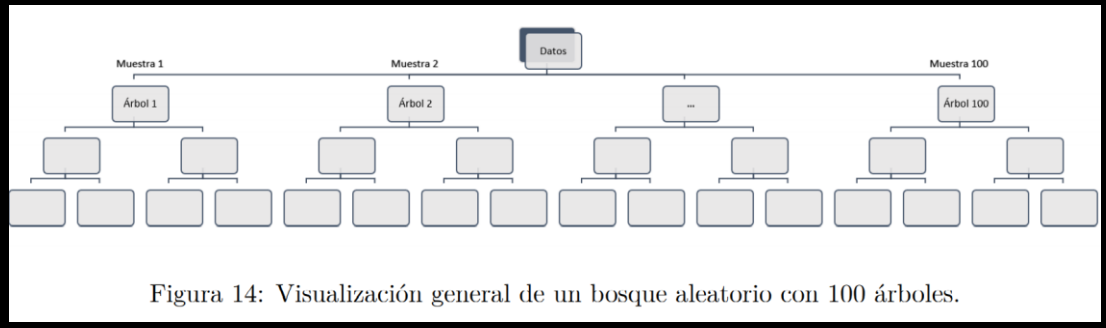


Figura 14: Visualización general de un bosque aleatorio con 100 árboles.

Bibliografía

- .html?gclid=CjwKCAjwnef6BRAgEiwAgv8mQZI1qtq3ZaUfJ_C2b1YTzQ_dYiEhUpxITdCEA5kiNZ7KTTZrRThfYBoCb5QQAvD_BwE
- González, L. (2018). ¿Qué es aprendizaje de máquina (machine learning)?. 26 de septiembre de 2020, de Github Sitio web: https://github.com/felipegonzalez/aprendizaje-maquina-mcd-2018
- Mármol, J. (2018). Introducción. 15 de agosto de 2020, de Github Sitio web: https://github.com/AnaLuisaMasetto/intro-to-data-science-2018