Machine Learning Workshop

Mentor: Nicolas Känzig

Email: nkaenzig@gmail.com
Workshop Repository: https://github.com/nkaenzig/ml-workshop

Contenido

Modulo 1
- Introducción ML
- Python crashcourse

Modulo 2
- Análisis de datos
- Preprocesamiento de datos
- Ejemplo ML

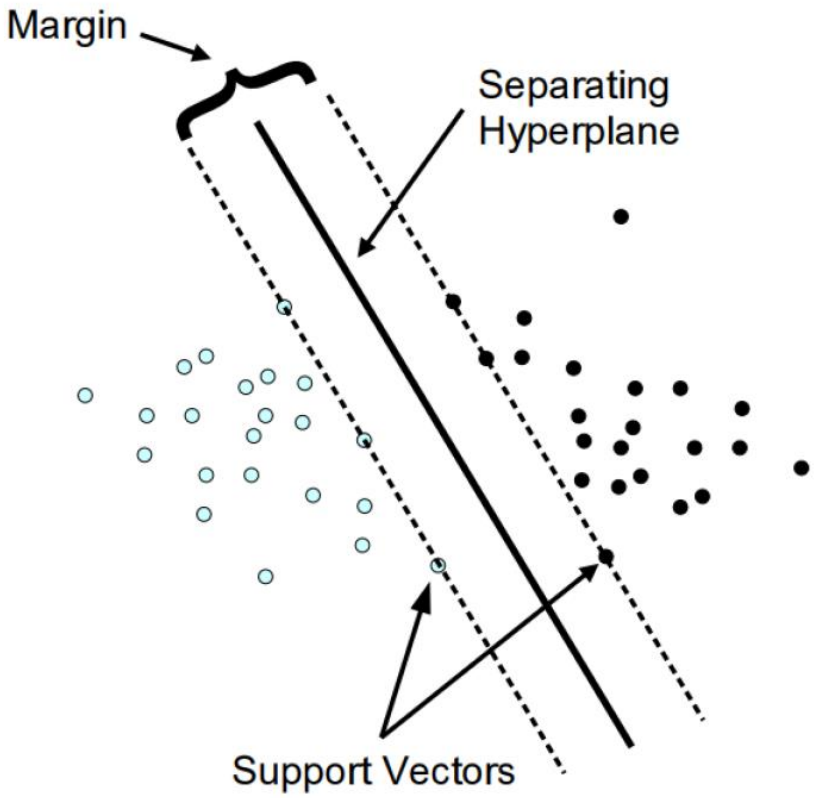
Modulo 3
- Modelos de ML
- Técnicas de evaluación
- Ejemplos ML

Machine Learning – Modelos

SVM (Suport Vector Machine)

- Supervised Classification
- Scikit-Learn:
  - sklearn.svm.SVC()

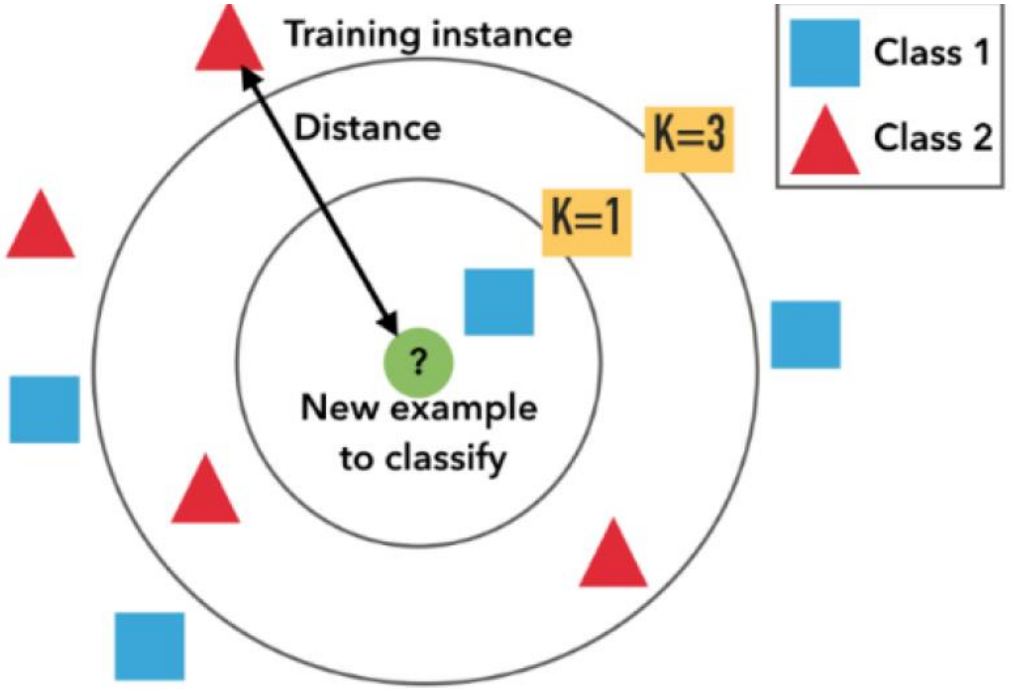
- Time Complexity (Training):
  - Linear SVM: **O(n)**
  - Non-Linear SVM: **$O(n^2)$ - $O(n^3)$**



K-NN (K-Nearest Neighbors)

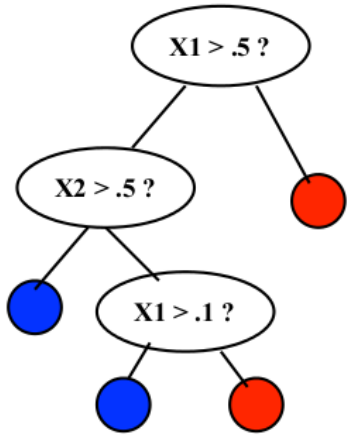
- Supervised Classification
- Scikit-Learn:
  - sklearn.neighbors.KNeighborsClassifier()

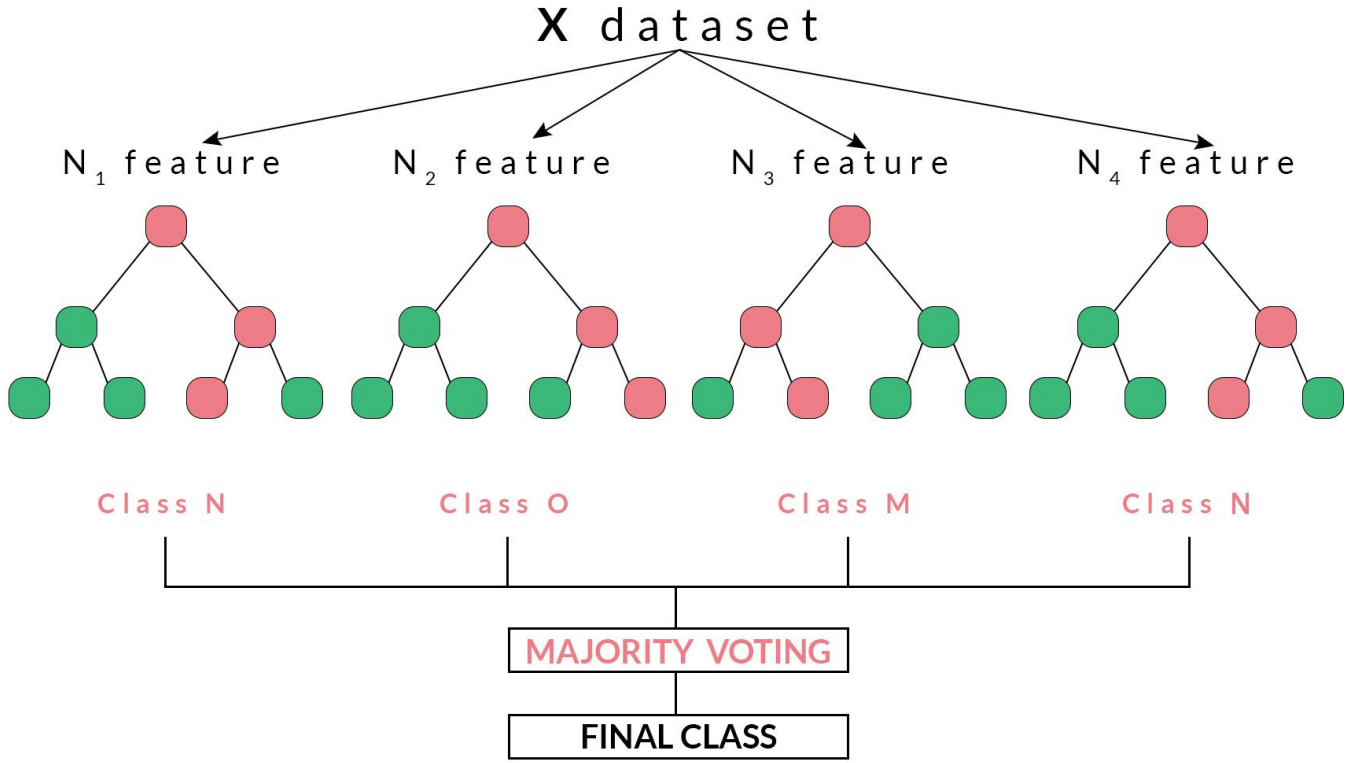
- Time Complexity (Training):
  - **O(n)**



Random Forest

- Supervised Classification
- Scikit-Learn:
  - sklearn.ensemble.RandomForestClassifier
- Time Complexity (Training):
  - **O(nlog(n))**





Ridge Regression

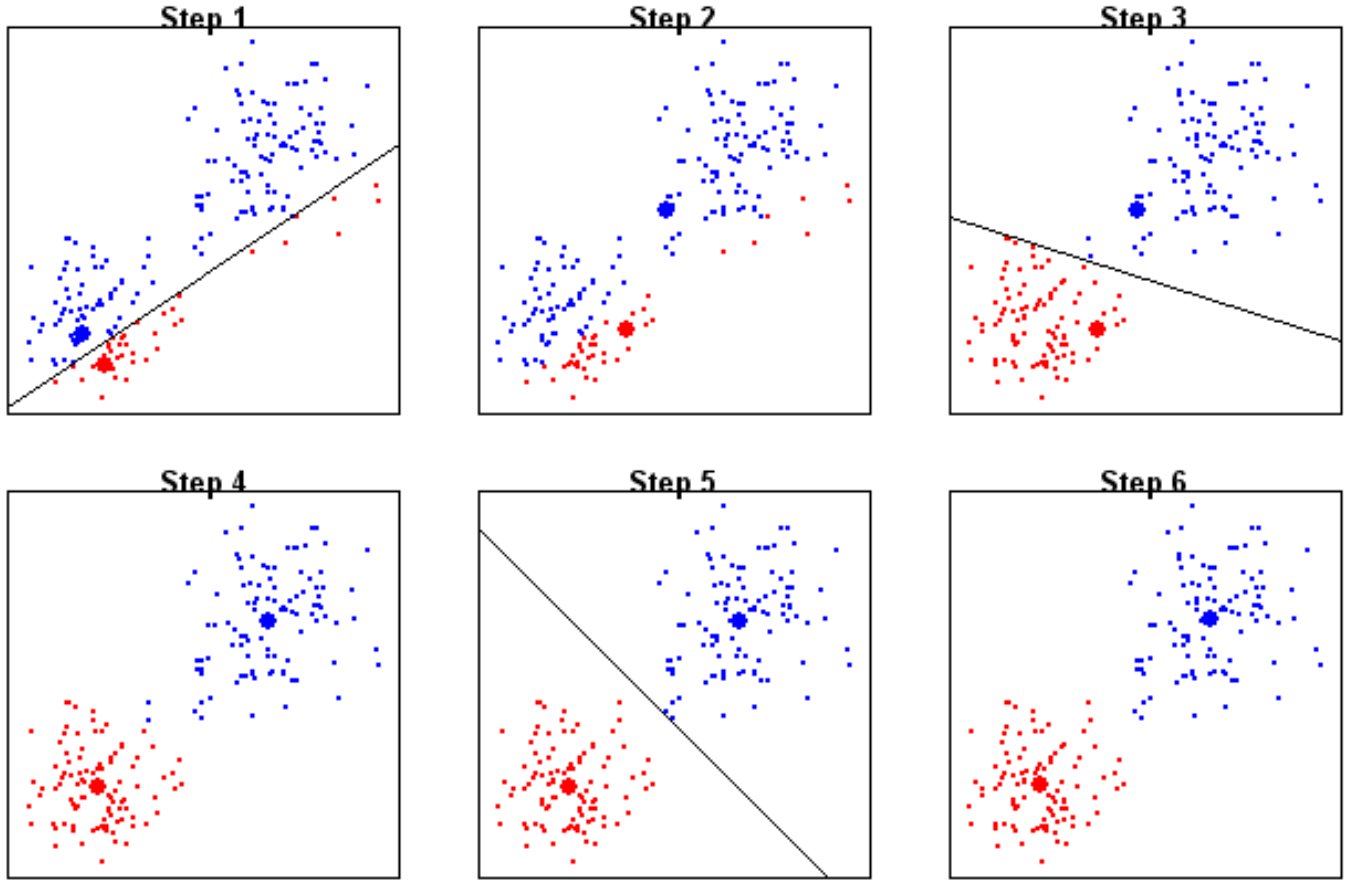
- Supervised Regression
- Scikit-Learn:
  - sklearn.linear_model.Ridge
- Time Complexity (Training):
  - **O(n)**

$$f(x,\theta) = \theta_0 + \theta_1 x$$

$$\theta^* = \underset{\theta}{\operatorname{argmin}} |f(x,\theta) - y|^2 + \alpha \left|\left|\theta\right|\right|_2^2$$

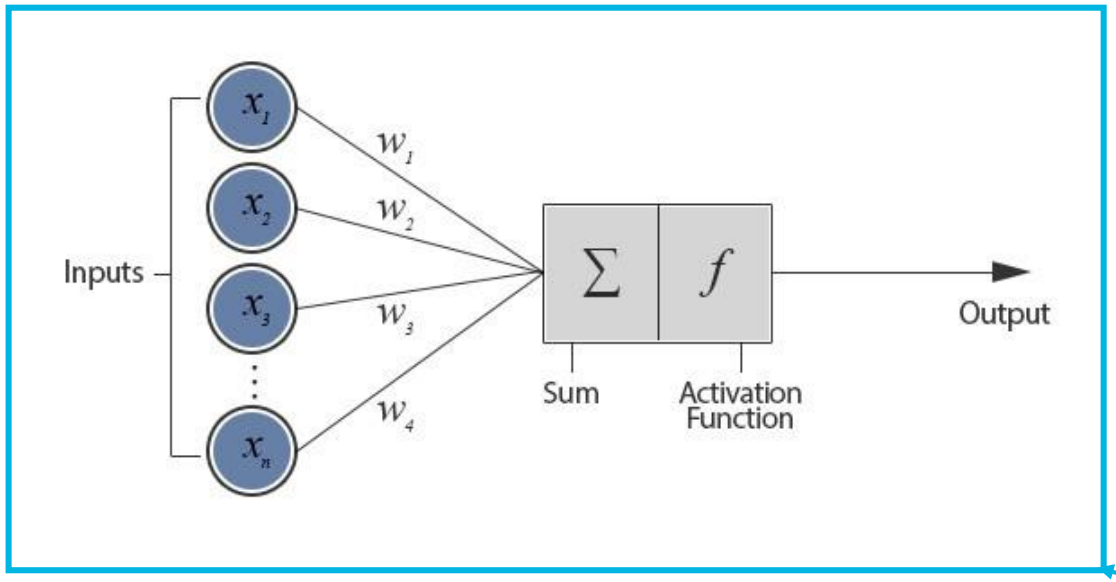
K-Means

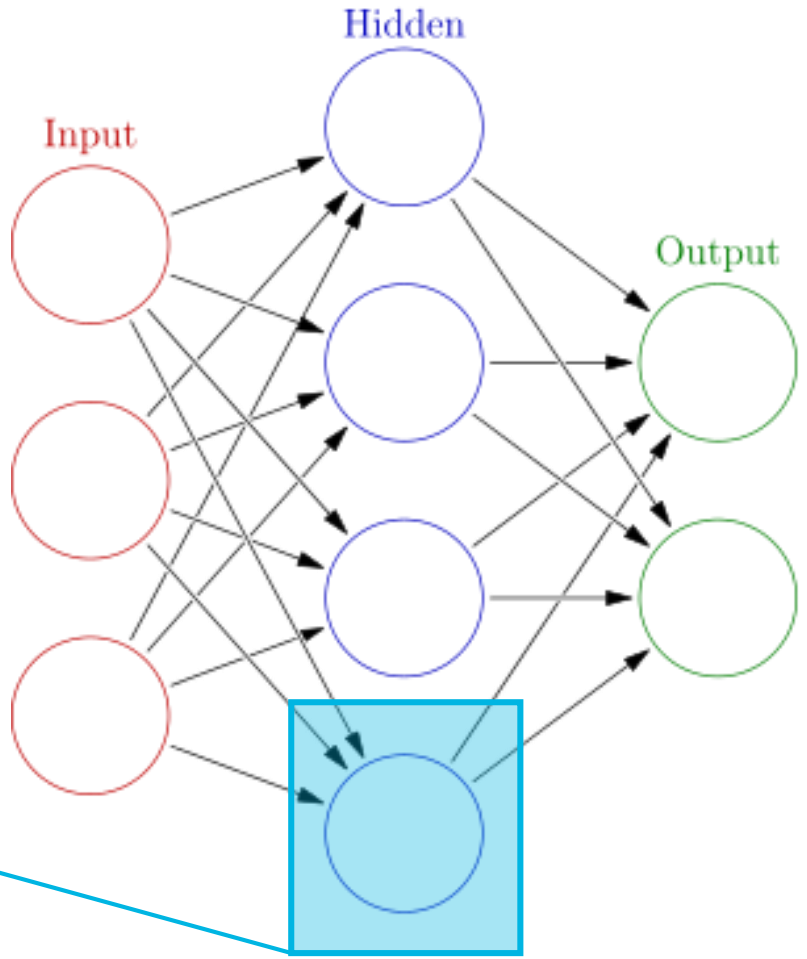
- Unsupervised Clustering
- Scikit-Learn:
  - sklearn.cluster.KMeans()
- Time Complexity (Training):
  - **O(n)**



Artificial Neural Networks

Neurona artificial

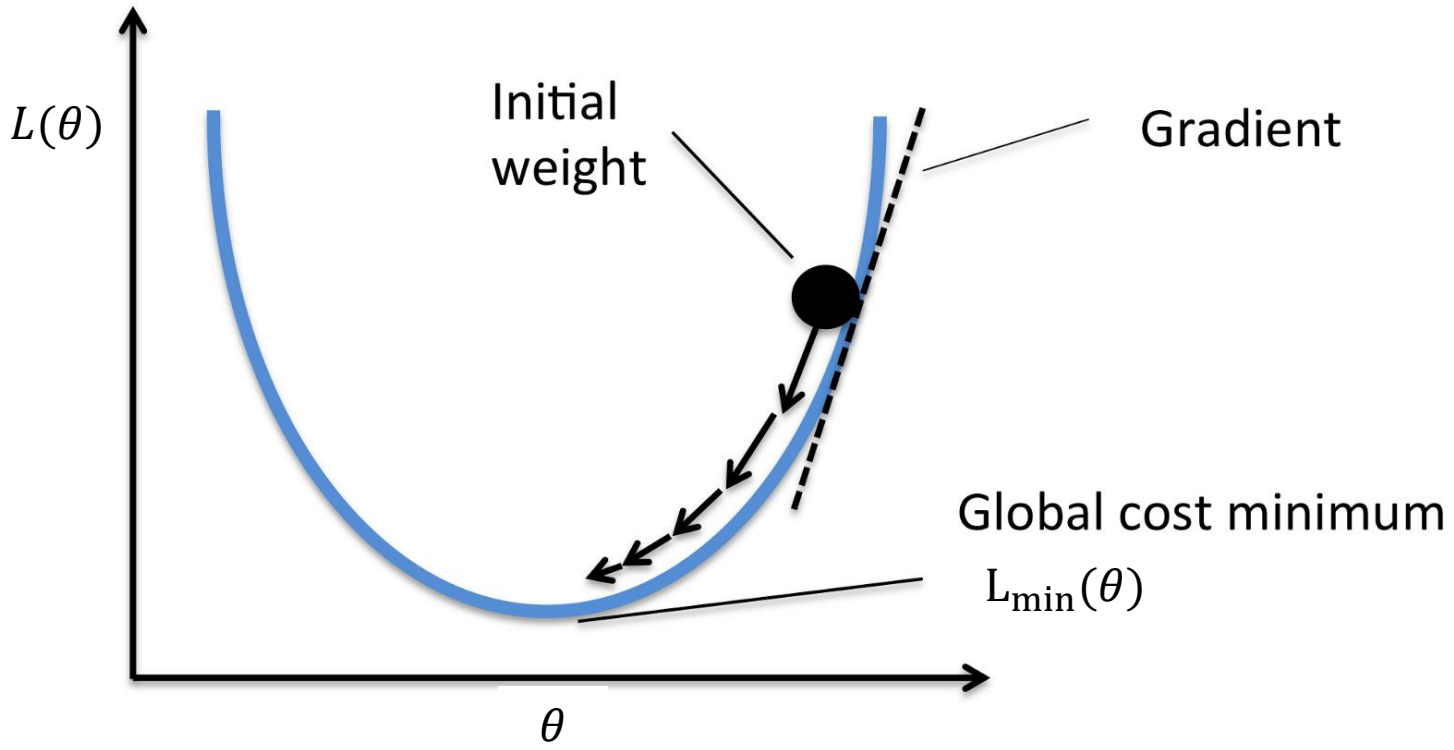




Como funciona el Training?

- **Objetivo:** $\theta^* = \underset{\theta}{\operatorname{argmin}}\, L(x, y, \theta)$

- **Gradient Descent**

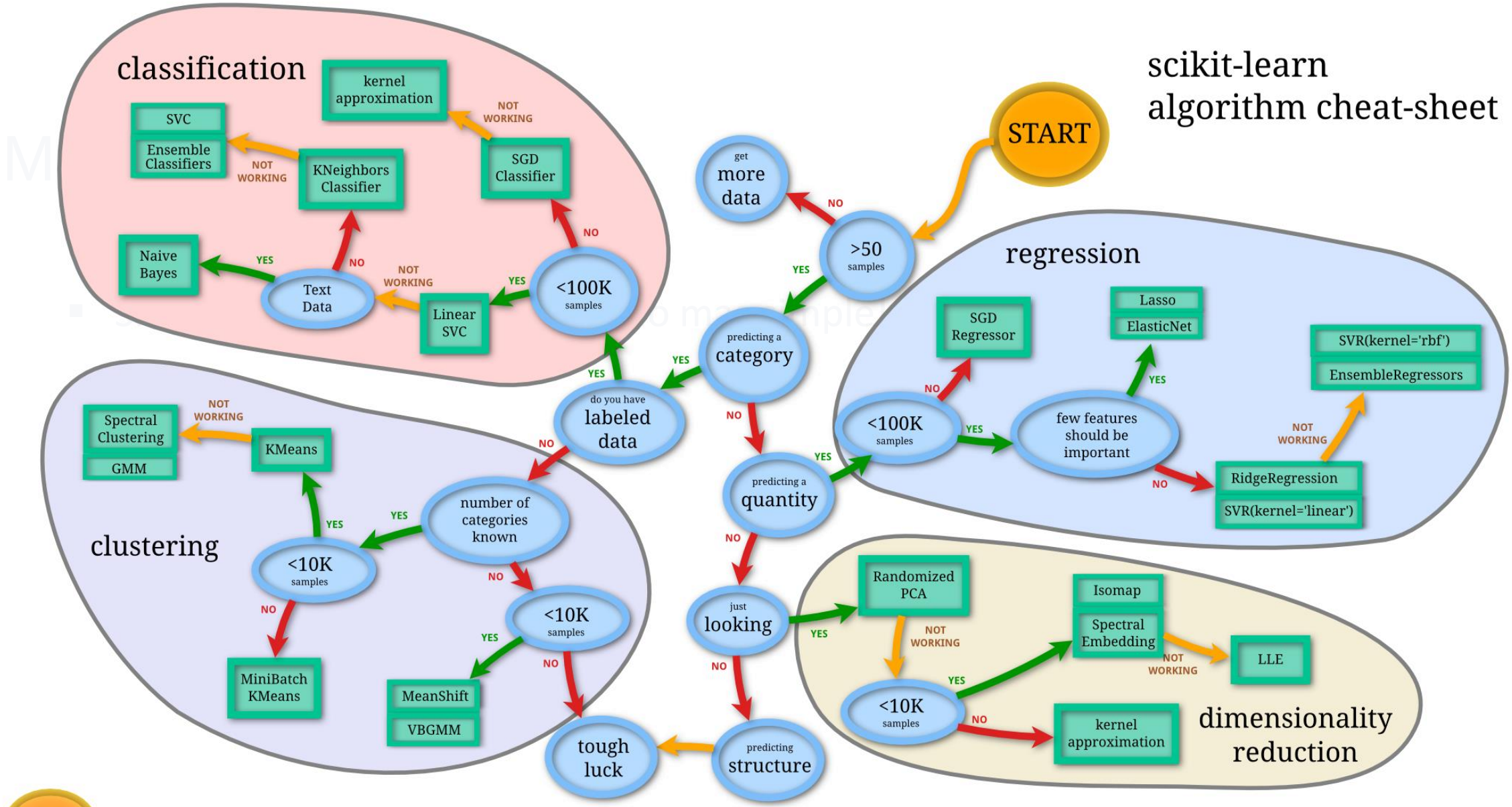


Algoritmos / Modelos

- Los modelos y algoritmos de optimización ya están implementados!
- Machine Learning:

- Deep Learning

scikit-learn algorithm cheat-sheet



Criterios para seleccionar un modelo

- Consejo: Siempre empieza con el modelo mas simple / fácil para usar

- **Preguntas**:
  - El modelo requiere standardization?
  - El modelo requiere one-hot encoding para los features categóricos?
  - El modelo requiere quitar columnas redundantes?
  - Que es la complejidad del modelo?
  - El modelo es lineal o no lineal?
  - Es difícil encontrar buenos parámetros para el modelo?
  - El modelo funciona con “imbalanced” datasets?

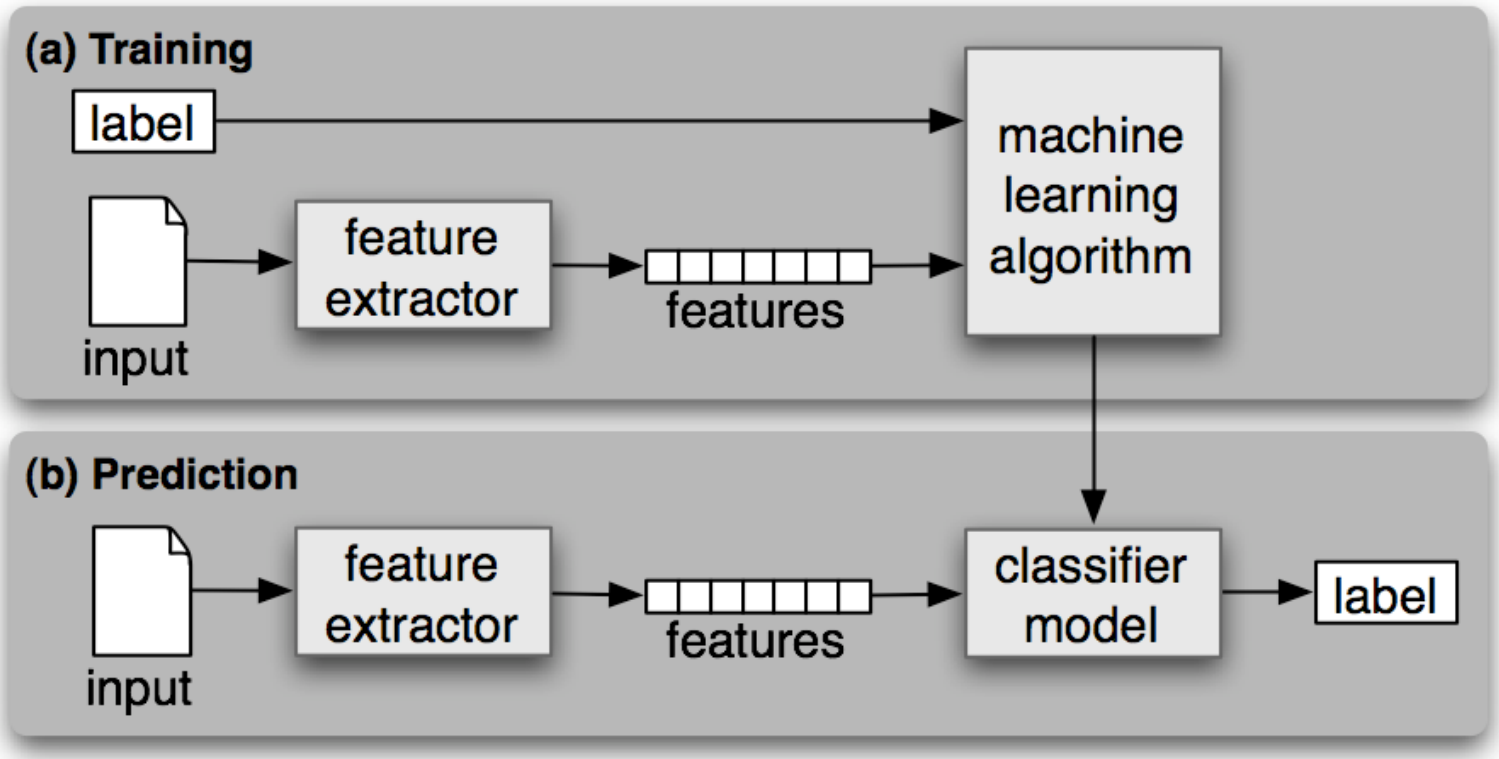
Criterios para seleccionar un modelo

Model	Standardization	Class Balancing	One-Hot Encoding	Non-Linear	Complexity
LinearSVM	yes	yes	yes	no	O(n)
SVM('rbf')	yes	yes	yes	yes	O(n^2) - O(n^3)
K-NN	yes	yes	yes	yes	O(n)
Random Forest	no	no	no	yes	O(nlog(n))

- Siempre empieza con el modelo mas simple / fácil para usar

  → Empezar con Random Forest siempre es una buena idea!

Machine Learning – Evaluación


(a) Training
label
input
feature extractor
features
machine learning algorithm
(b) Prediction
input
feature extractor
features
classifier model
label

Como medir la calidad de las predicciones?

- **Clasificación**:

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{\#\ Correct\ predictions}}{\boldsymbol{\#\ Predictions}}$$

$$\boldsymbol{Precision} = \frac{\boldsymbol{\#\ True\ positives}}{\boldsymbol{\#\ True\ positives\ +\ \#\ False\ Positives}} \qquad \boldsymbol{Recall} = \frac{\boldsymbol{\#\ True\ positives}}{\boldsymbol{\#\ True\ positives\ +\ \#\ False\ Negatives}}$$

Problemas con accuracy

- Dataset:
  - 1000 samples de pacientes **sin** cancer (N)
  - 5 samples de pacientes **con** cancer (P)

- Modelo solamente diagnostica 1 de los 5 pacientes con cancer

$$Accuracy = \frac{\#\ Correct\ predictions}{\#\ Predictions} = ?$$

$$Precision = \frac{\#\ True\ positives}{\#\ True\ positives\ +\ \#\ False\ Positives} = ?$$

$$Recall = \frac{\#\ True\ positives}{\#\ True\ positives\ +\ \#\ False\ Negatives} = ?$$

Problemas con accuracy

- Dataset:
  - 1000 samples de pacientes **sin** cancer (N)
  - 5 samples de pacientes **con** cancer (P)

- Modelo solamente diagnostica 1 de los 5 pacientes con cancer

$$Accuracy = \frac{\#\ Correct\ predictions}{\#\ Predictions} = \frac{1001}{1005}\% = 99.6\%$$

$$Precision = \frac{\#\ True\ positives}{\#\ True\ positives + \#\ False\ Positives} = \frac{1}{1}\% = 100\%$$

$$Recall = \frac{\#\ True\ positives}{\#\ True\ positives + \#\ False\ Negatives} = \frac{1}{5}\% = 20\%$$

Como medir la calidad de las predicciones?

- **Regresión**:
  - Mean Absolute Error

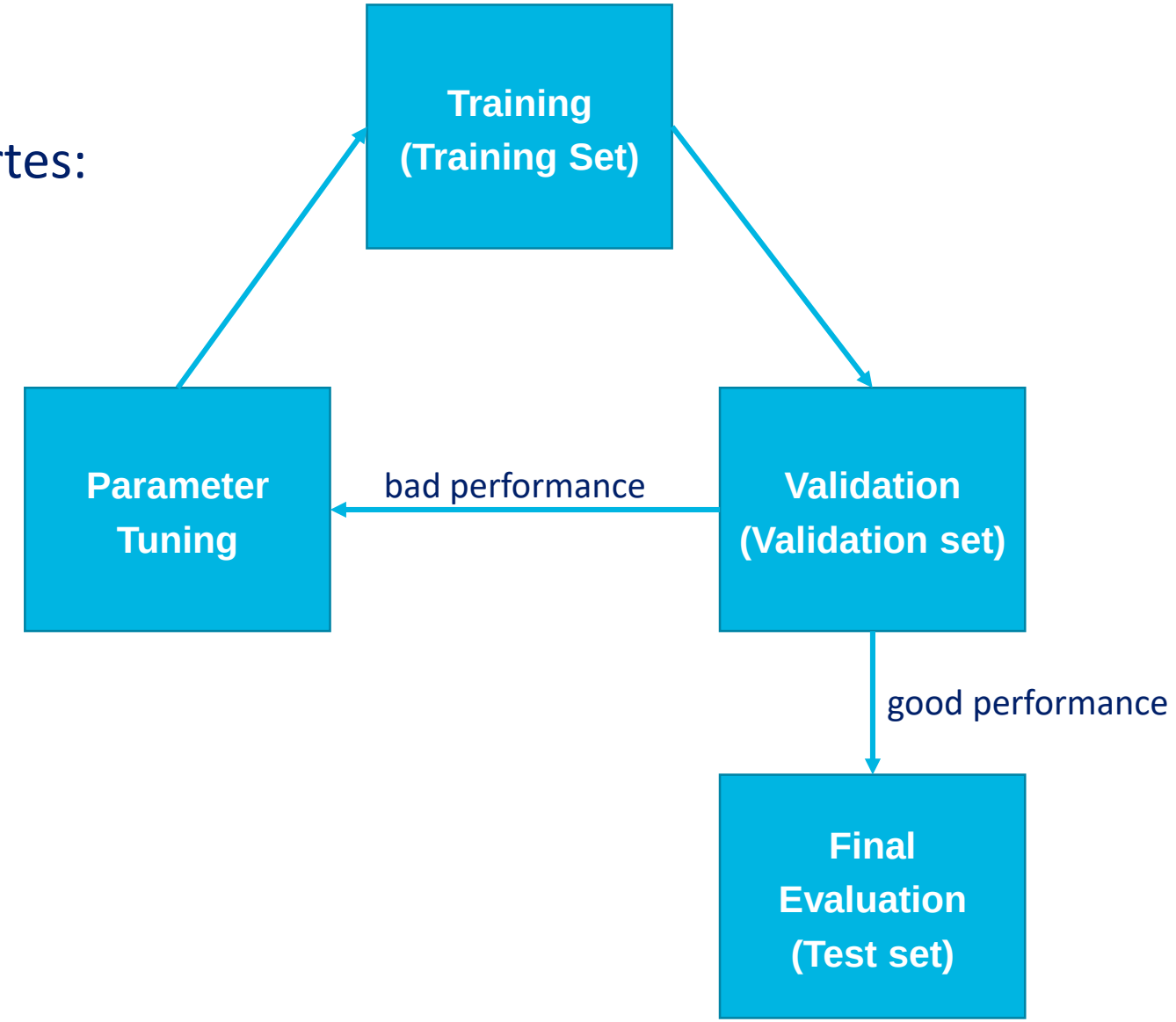
$$MAE = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i|$$

  - Root Mean Squared Error

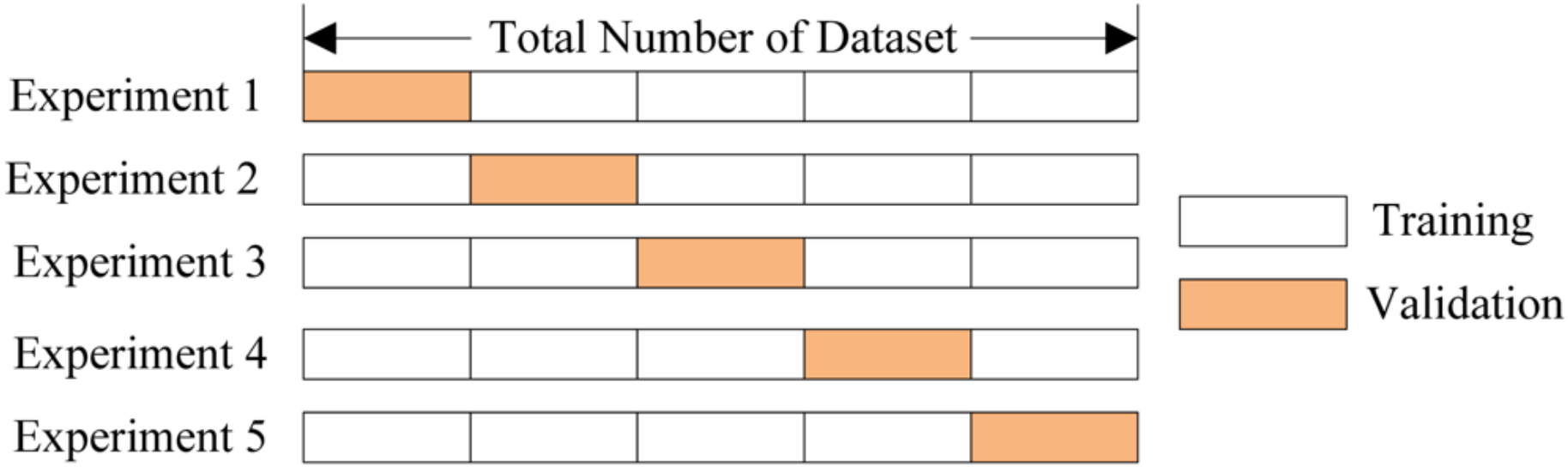
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}$$

Datasets

- División de los datos en 3 partes:
  - Train set (70%)
  - Validation set (20%)
  - Test set (10%)



Cross-Validation



Sklearn

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

# Data loading & preprocessing
# ...
# ...

# Define the model
model = SVC(kernel='linear');

# Train the model
model.fit(x_train, y_train)

# Make predictions
y_predicted = model.predict(x_test, y_test)

# Evaluate
accuracy_score(y_test, y_predicted)
```