Machine Learning Workshop

Nicolas Känzig

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

Contenido

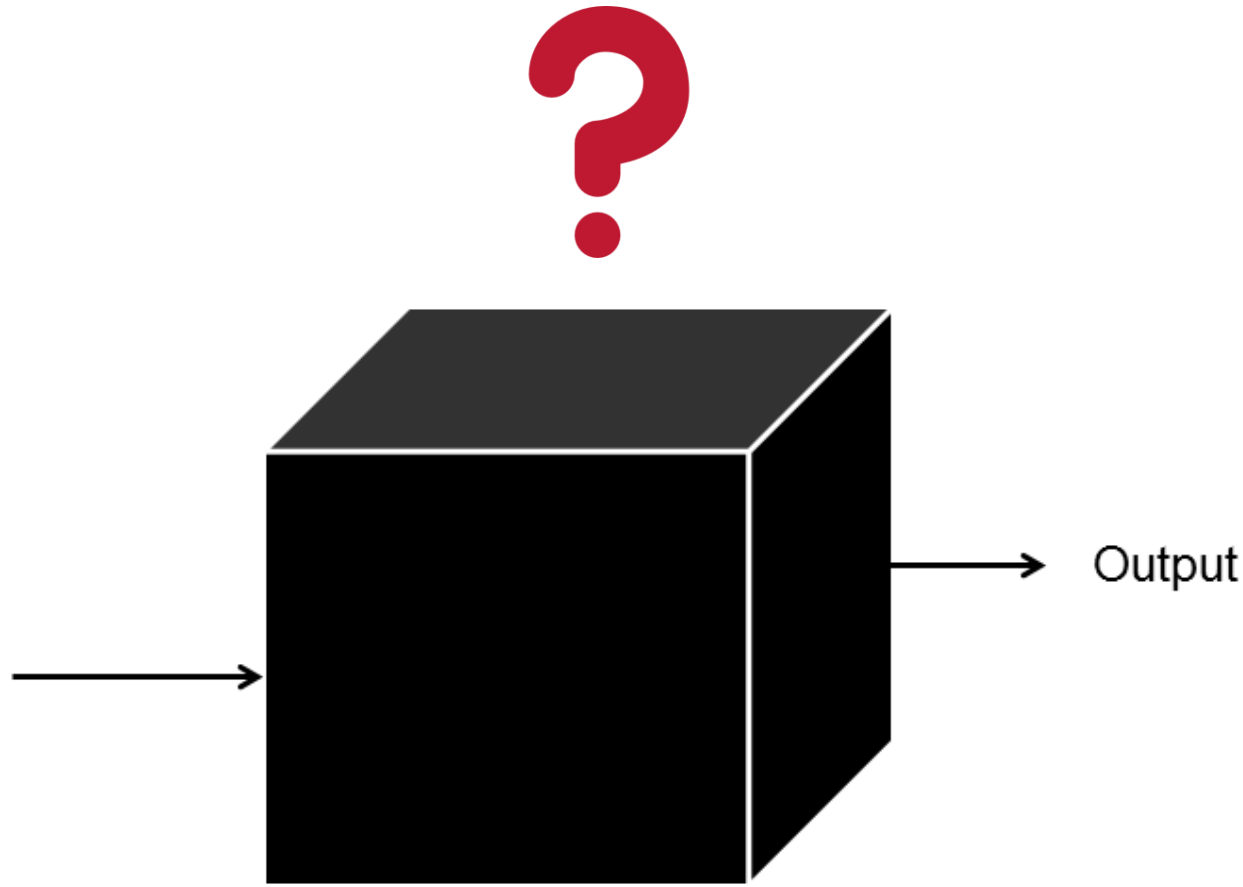
Modulo 1
- Introducción ML
- Python

Modulo 2
- Análisis de datos
- Preprocesamiento de datos

Modulo 3
- Modelos de ML
- Técnicas de evaluación

Machine Learning – Modelos

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

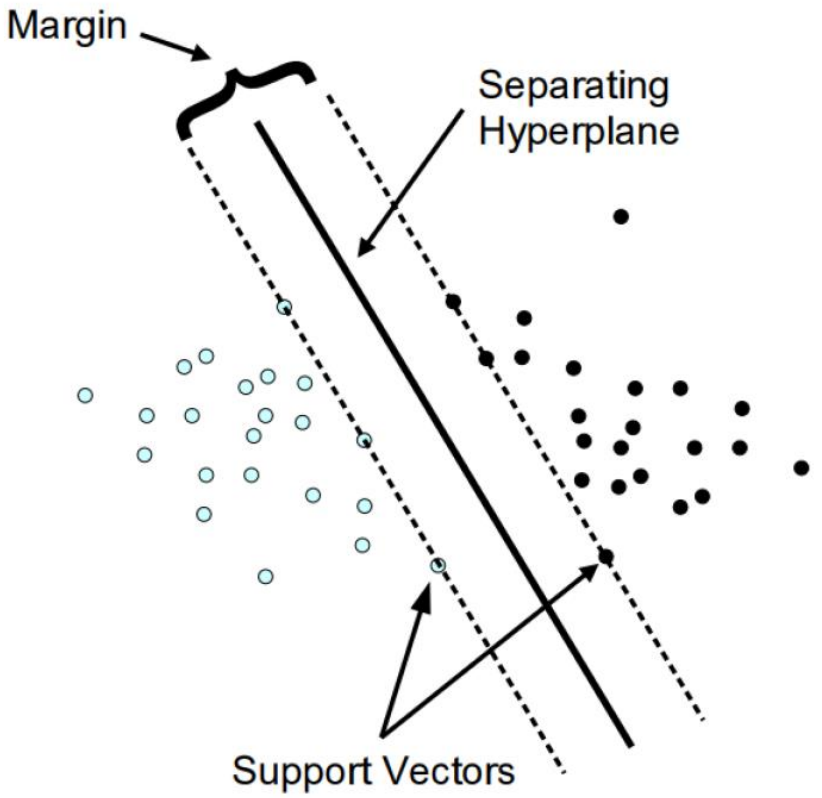


$a_{ij} \in \mathbb{R}$

SVM (Support Vector Machine)

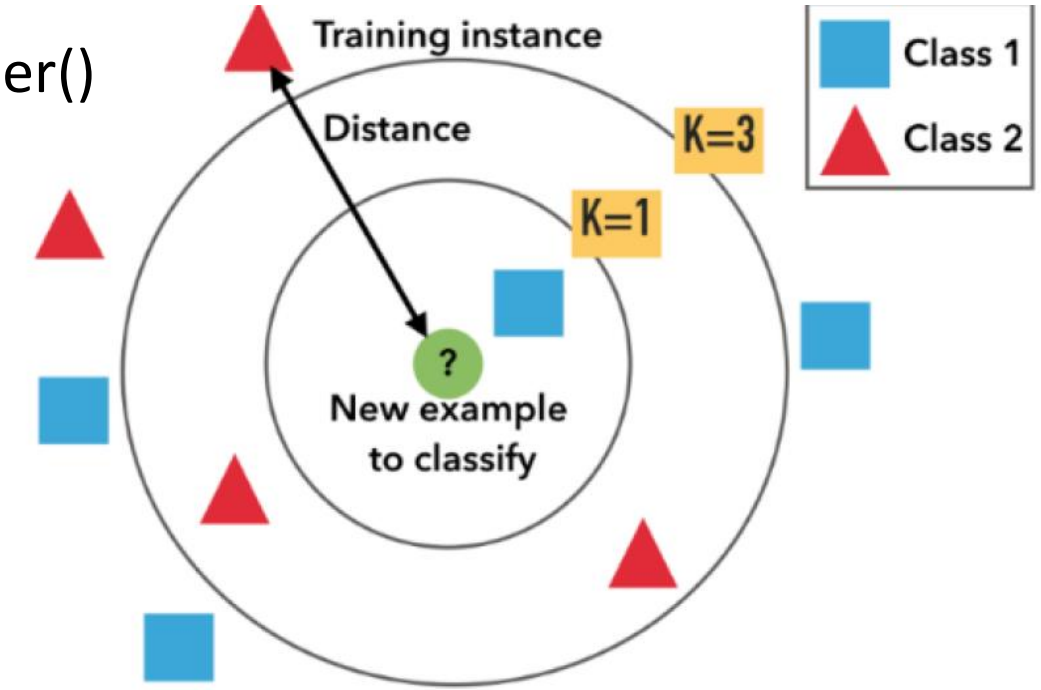
- **Supervised Classification**
- Scikit-Learn:
  - sklearn.svm.SVC()

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

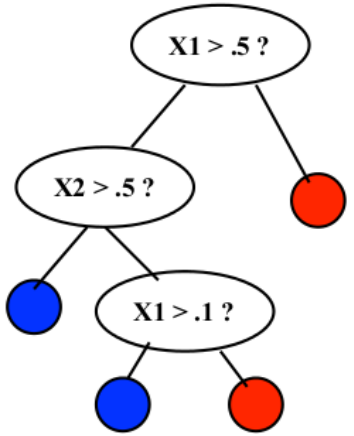


K-NN (K-Nearest Neighbors)

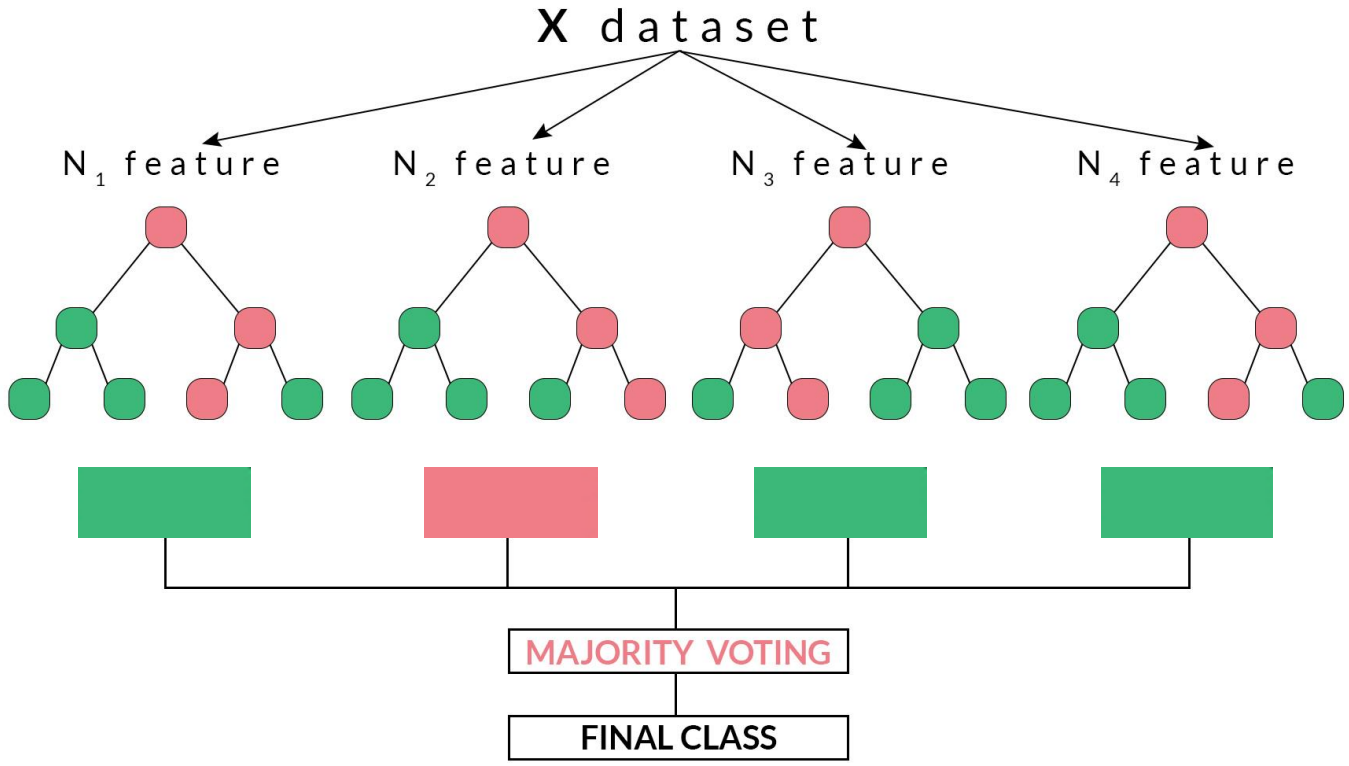
- **Supervised Classification**
- Scikit-Learn:
  - sklearn.neighbors.KNeighborsClassifier()
- Time Complexity (Training):
  - **O(n)**



Random Forest



- **Supervised Classification & Regression**
- 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)**

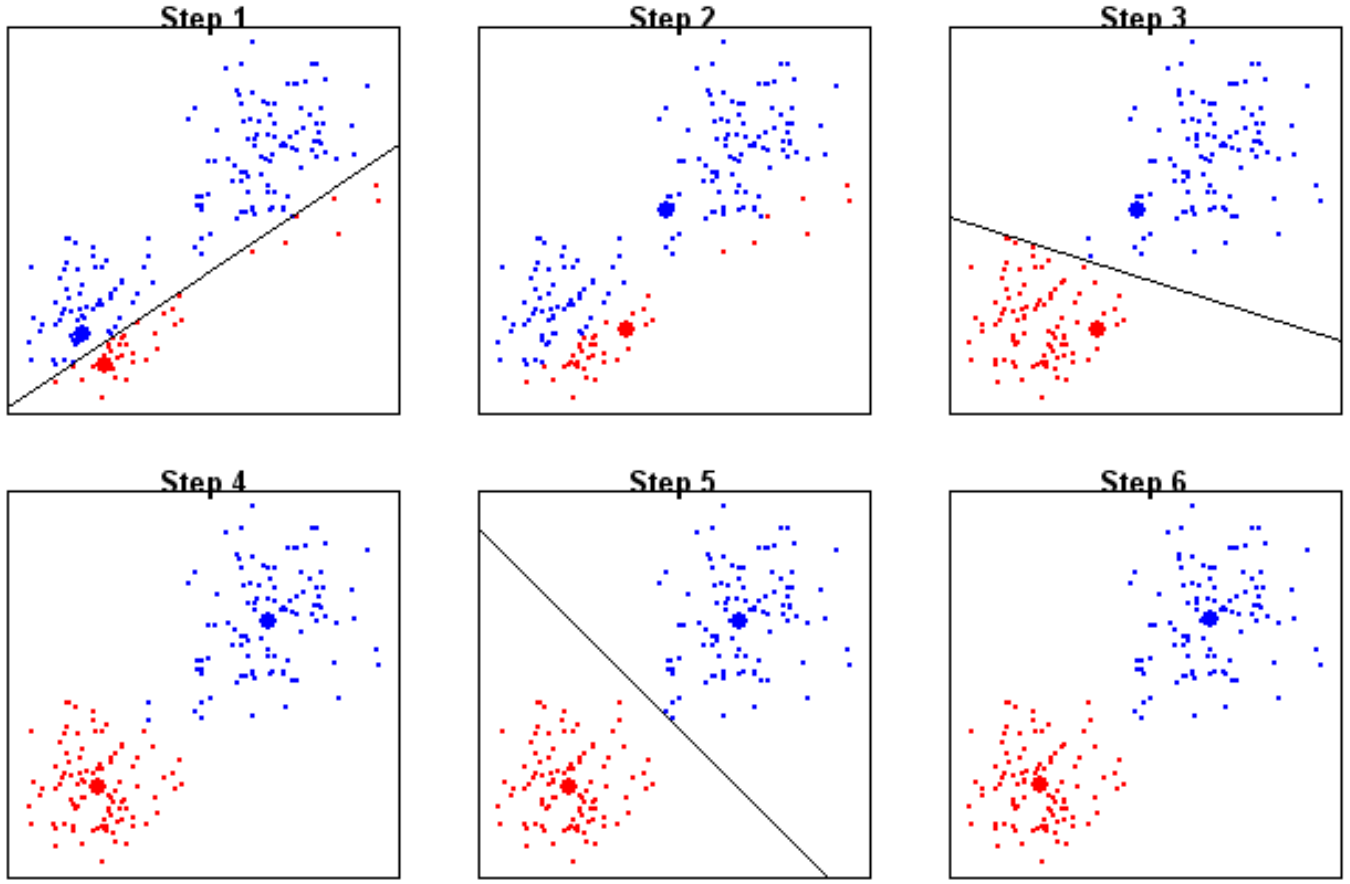
$$f(x,\theta) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_p x_p$$

$$\theta^* = \underset{\theta}{\operatorname{argmin}} |f(x,\theta) - y|^2 + \alpha \sum_{i=0}^{p} \theta_i^2$$

Regularización

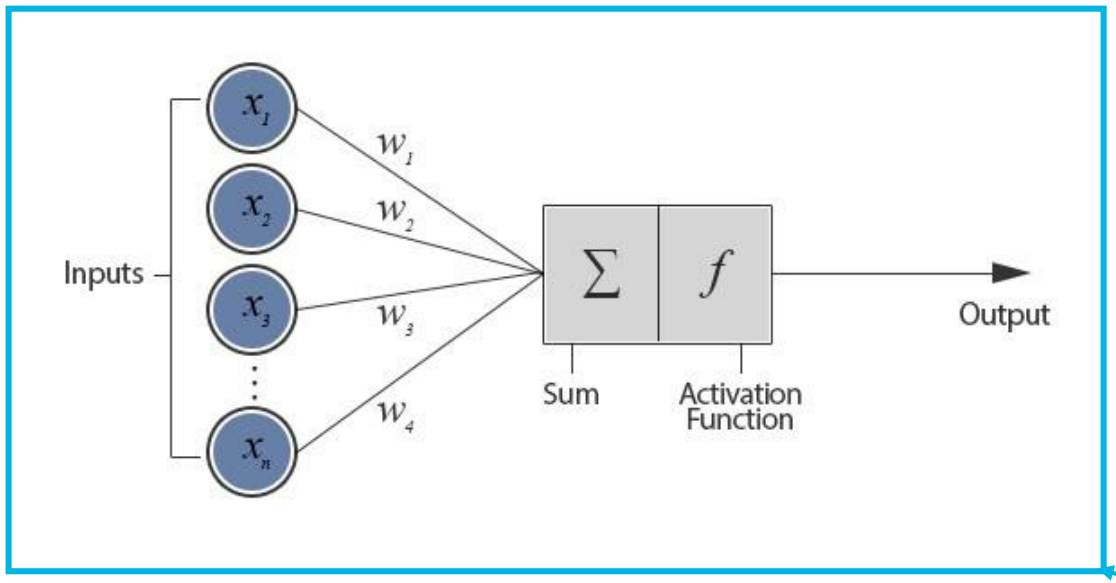
K-Means

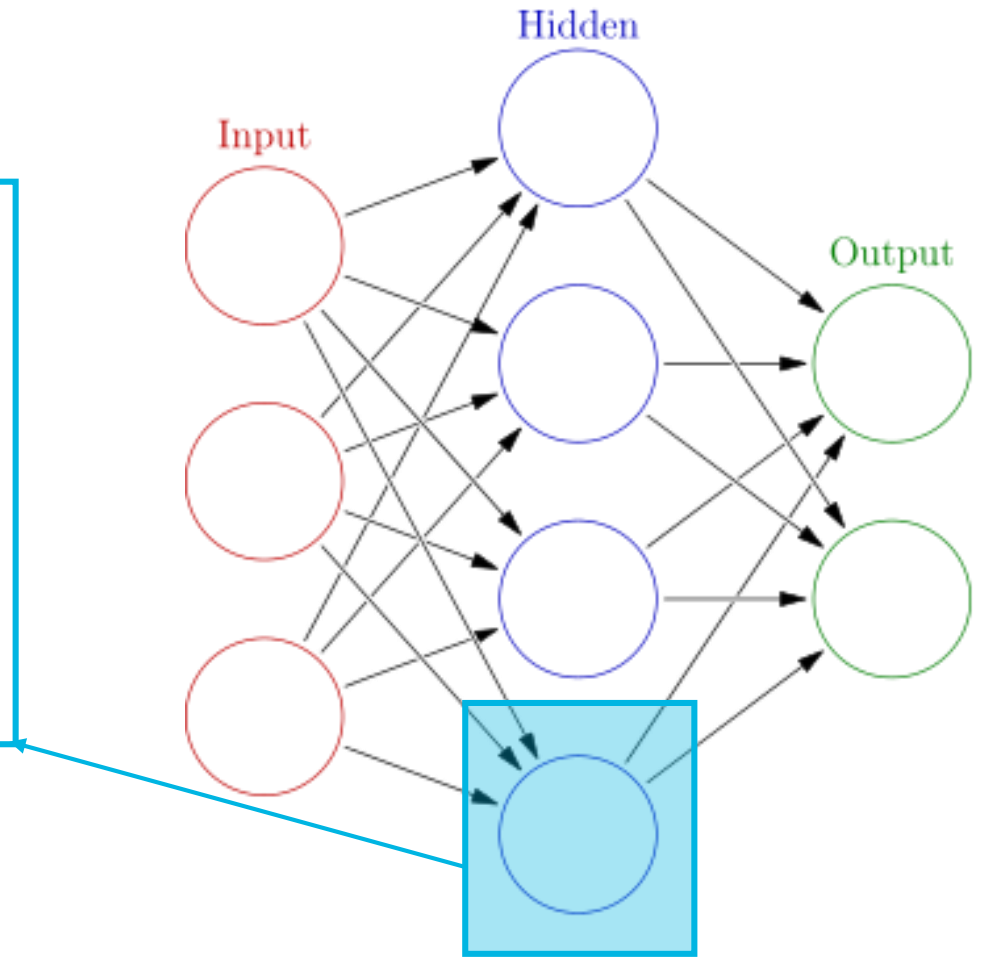
- **Unsupervised Clustering**
- Scikit-Learn:
  - sklearn.cluster.KMeans()
- Time Complexity (Training):
  - **O(n)**



Artificial Neural Networks (Deep Learning)

Neurona artificial

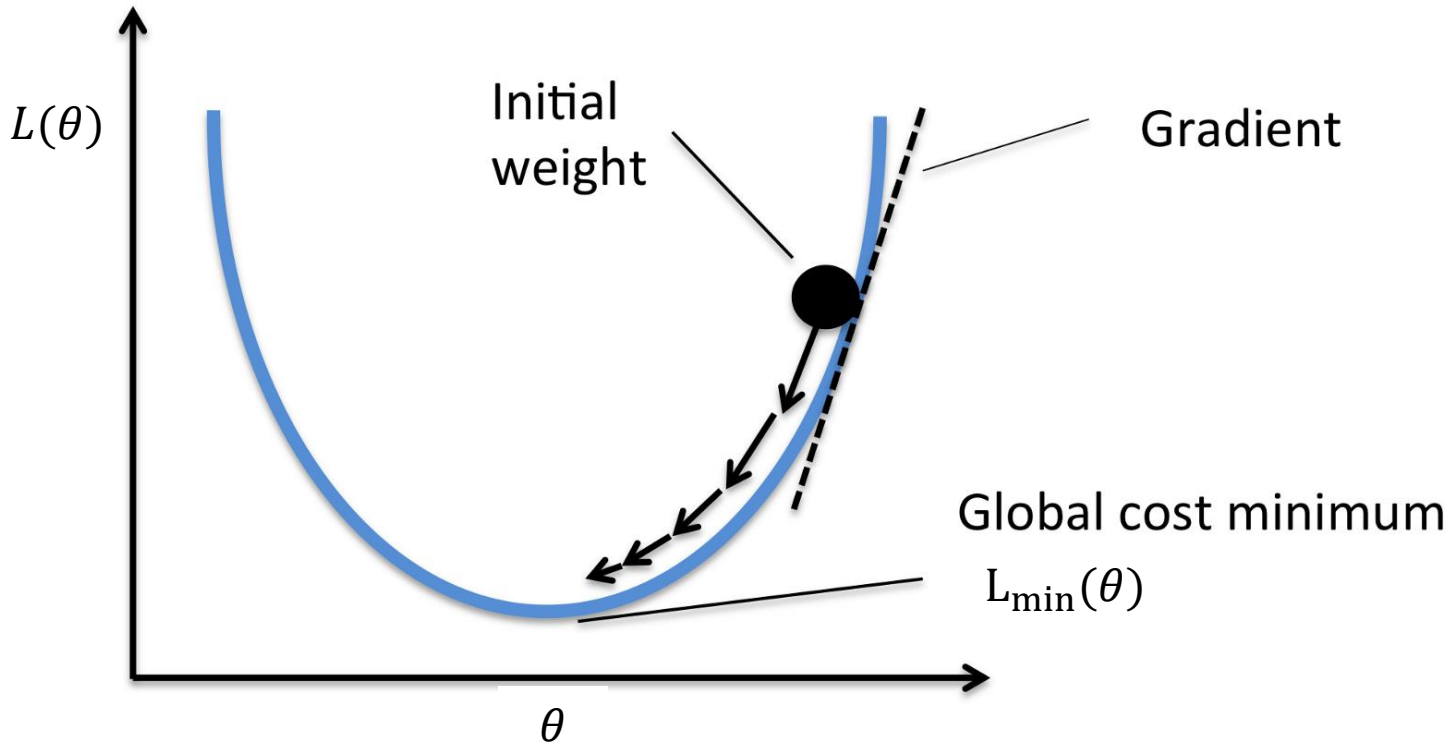




Como funciona el Entrenamiento?

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

- **Gradient Descent**



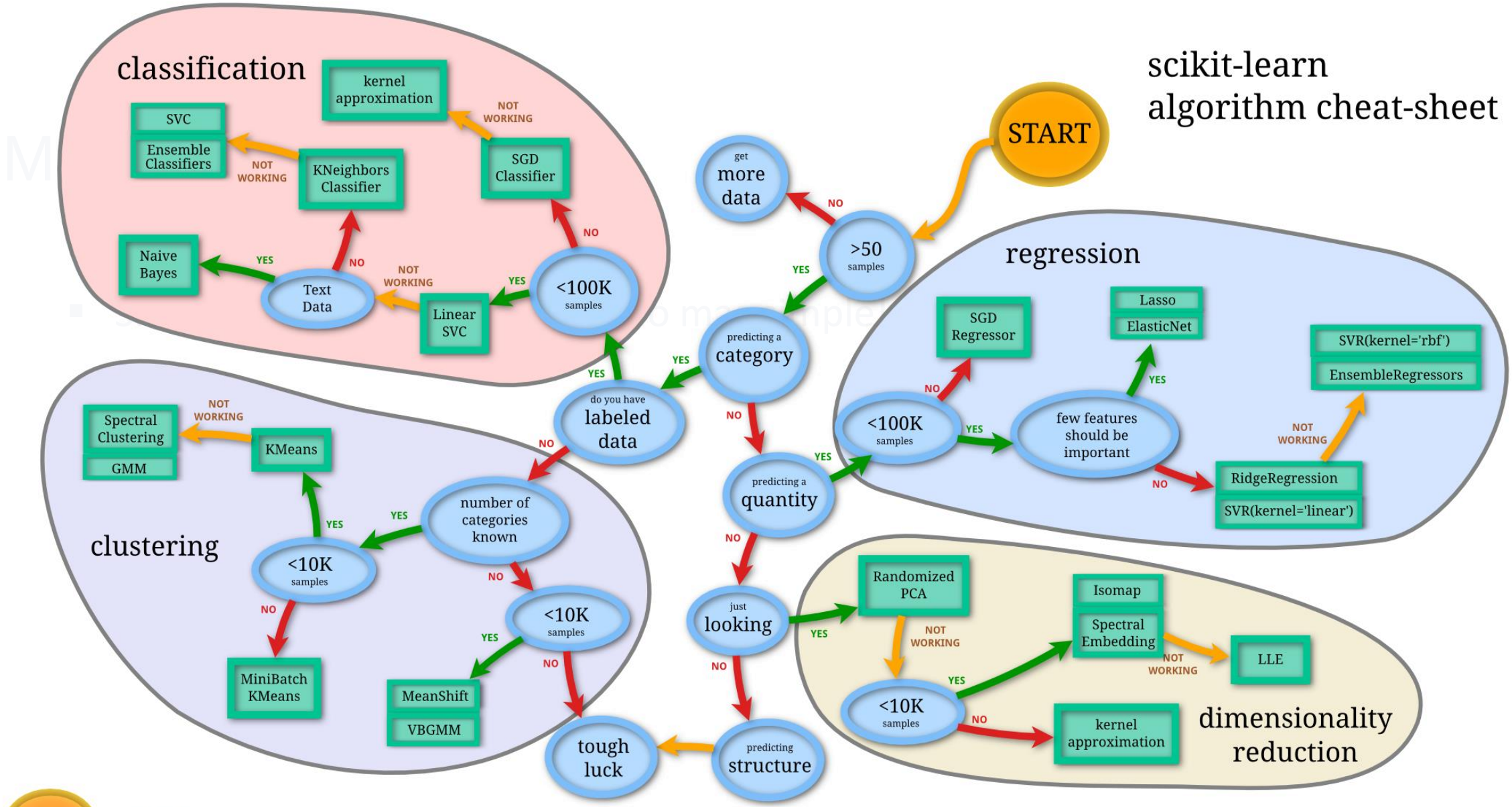
Machine Learning Frameworks

- Los modelos y algoritmos de optimización ya están implementados!

Machine Learning:

Deep Learning

scikit-learn algorithm cheat-sheet



Criterios para seleccionar un modelo

- **Preguntas**:
  - ¿Es difícil configurar el modelo?
  - ¿El modelo hace una suposición sobre la distribución de los datos?
  - ¿El modelo funciona con features categoricos?
  - ¿El modelo funciona con columnas correlacionadas?
  - ¿El entrenamiento funciona con “imbalanced” datasets?
  - ¿Que es la complejidad del modelo?
  - ¿Qué rápido son las predicciones?

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

Criterios para seleccionar un modelo

Model	Feature Selection	Class Balancing	One-Hot Encoding	Non-Linear	Complexity
LinearSVM	no	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))

- Empezar con **Random Forest** siempre es una buena idea!
  - Muy fácil para configurar
  - Muy poco preprocesamiento necesario
  - Rápido (Entrenamiento & Prediccion)
  - No hace ninguna suposición sobre la distribución de los datos
  - ≈50% de los modelos ganadoros en Kaggle usan Random Forest

Machine Learning – Evaluación

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

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

$$Precision = \frac{\#\ True\ positives}{\#\ True\ positives + \#\ False\ Positives} = \frac{1}{1+0}\% = 100\%$$

$$Recall = \frac{\#\ True\ positives}{\#\ True\ positives + \#\ False\ Negatives} = \frac{1}{1+4}\% = 20\%$$

Como medir la calidad de las predicciones?

- **Regresión**:
  - Mean Absolute Error

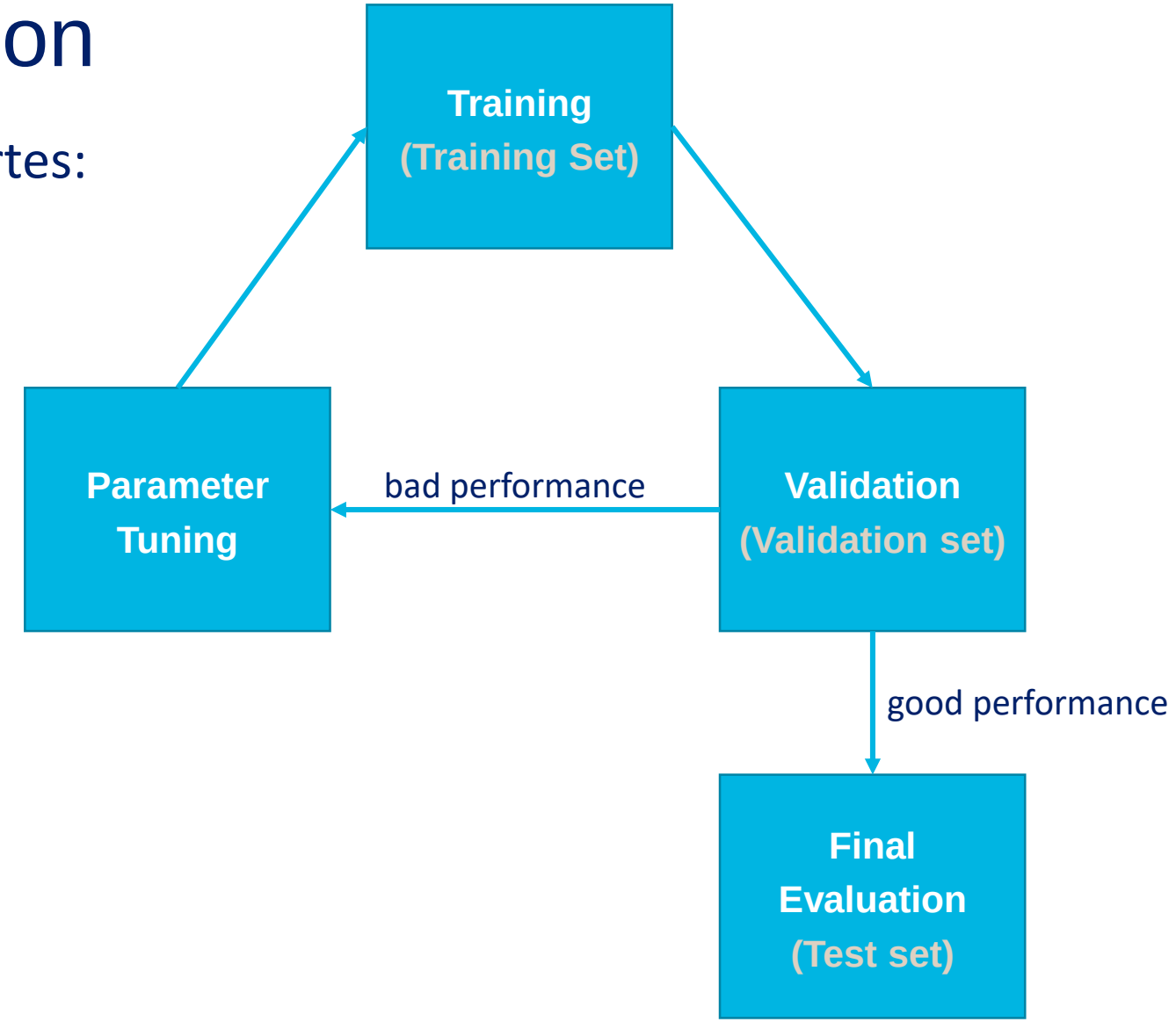
$$MAE = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i|$$

  - Root Mean Squared Error

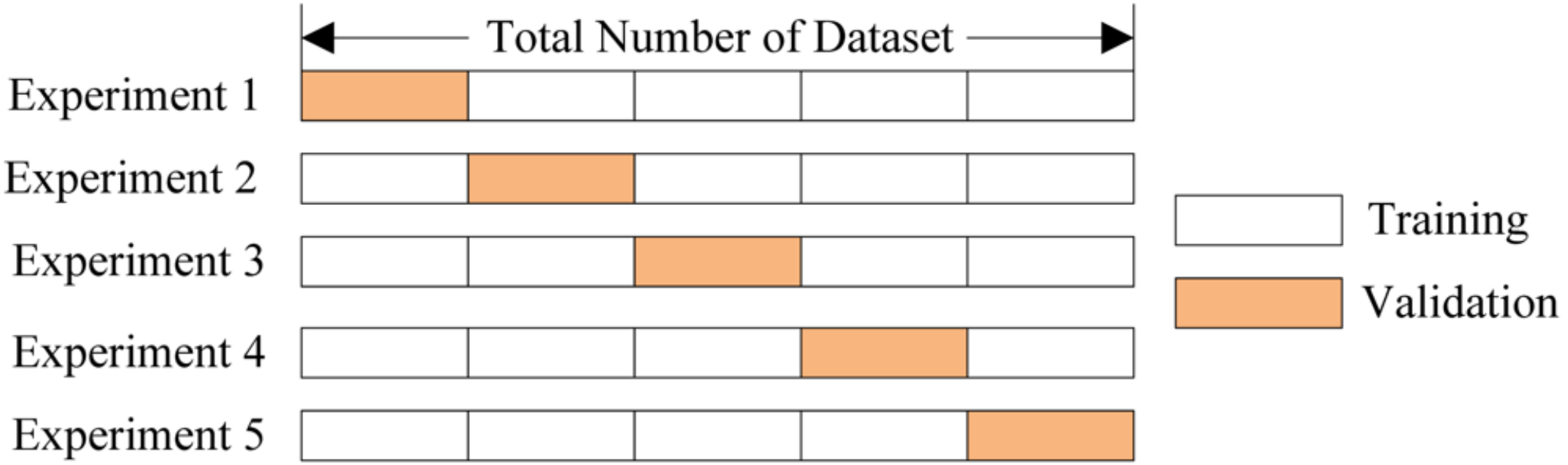
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}$$

Training vs. Validation

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



Cross-Validation




```
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)
```