Introduction to Machine Learning

Systems Sciences Foundations

Author: Eng. Carlos Andrés Sierra, M.Sc.
cavirguezs@udistrital.edu.co

Lecturer
Department of Computer Engineering
School of Engineering
Universidad Distrital Francisco José de Caldas

2025-I

Outline

1. Fundamentals of Machine Learning ✓
2. Python Tools for Machine Learning ✓
3. Supervised Machine Learning ✓
4. Machine Learning Models Evaluation ✓

Outline

1. Fundamentals of Machine Learning
2. Python Tools for Machine Learning
3. Supervised Machine Learning
4. Machine Learning Models Evaluation

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Key Concepts in Machine Learning

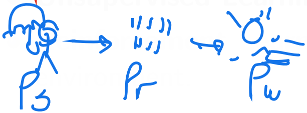
Machine Learning

- **Machine learning** is a method of data analysis that automates analytical model building.
- It is a **branch** of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with `minimal human intervention`.

statistics → Bayes

mo [illegible]

- Supervised Learning: The model is trained on labeled data.
- Unsupervised Learning: The model is trained on unlabeled data.
- Reinforcement Learning: The model learns by interacting with an environment.



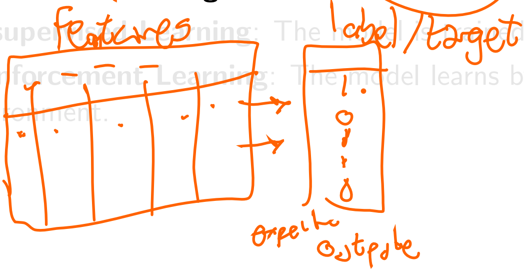
Key Concepts in Machine Learning

Machine Learning

- **Machine learning** is a method of data analysis that automates analytical model building.
- It is a **branch** of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with `minimal human intervention`.

1

- **Supervised Learning**: The model is trained on labeled data.
- **Unsupervised Learning**: The model is trained on unlabeled data.
- **Reinforcement Learning**: The model learns by interacting with an environment.

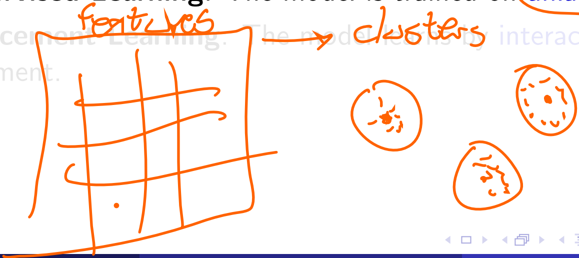


Key Concepts in Machine Learning

Machine Learning

- **Machine learning** is a method of data analysis that automates analytical model building.
- It is a **branch** of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with `minimal human intervention`.

- **Supervised Learning**: The model is trained on labeled data.
- **Unsupervised Learning**: The model is trained on unlabeled data.
- **Reinforcement Learning**: The model learns by interacting with an environment.



Key Concepts in Machine Learning

Machine Learning

- **Machine learning** is a method of data analysis that automates analytical model building.
- It is a **branch** of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with `minimal human intervention`.

- **Supervised Learning**: The model is trained on labeled data.
- **Unsupervised Learning**: The model is trained on unlabeled data.

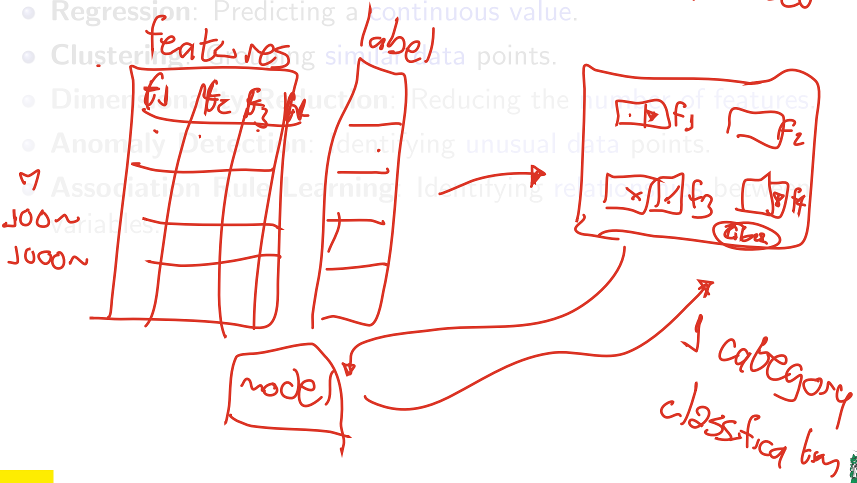
step

- **Reinforcement Learning**: The model learns by interacting with an environment.

multi-step → cybernetics

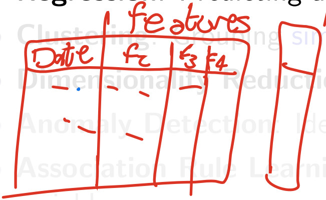
Typical Machine Learning Problems

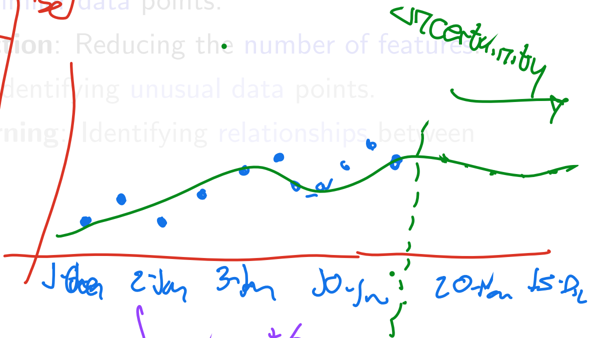
- **Classification**: Predicting a label. target → supervised
- **Regression**: Predicting a continuous value.
- **Clustering**: Grouping similar data points.
- **Dimensionality Reduction**: Reducing the number of features.
- **Anomaly Detection**: Identifying unusual data points.
- **Association Rule Learning**: Identifying relationships between variables.

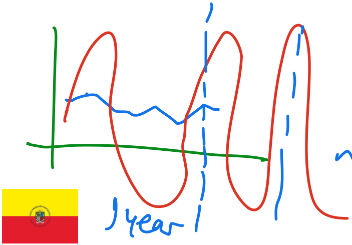


Typical Machine Learning Problems

- **Classification**: Predicting a label.
- **Regression**: Predicting a continuous value.
- **Clustering**: Grouping similar data points.
- **Dimensionality Reduction**: Reducing the number of features.
- **Anomaly Detection**: Identifying unusual data points.
- **Association Rule Learning**: Identifying relationships between variables.







~3 years
synthetic + bias
cross-validate
K-fold

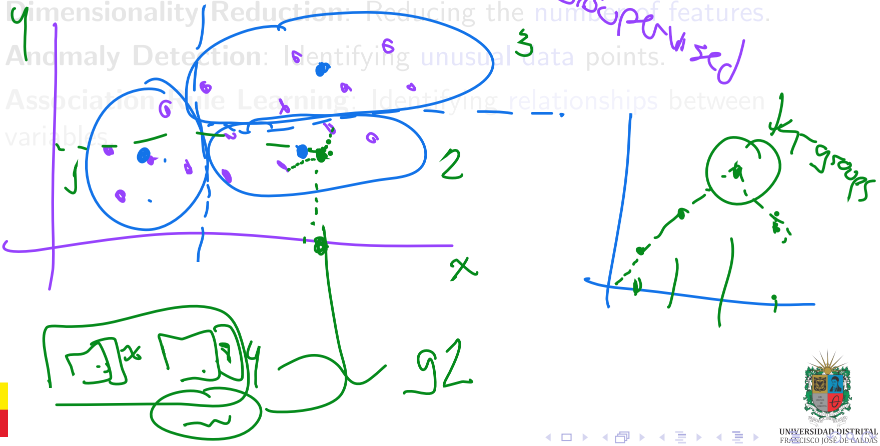
today

Typical Machine Learning Problems

- **Classification**: Predicting a label.
- **Regression**: Predicting a continuous value.
- **Clustering**: Grouping similar data points.
- **Dimensionality Reduction**: Reducing the number of features.
- **Anomaly Detection**: Identifying unusual data points.
- **Association Rule Learning**: Identifying relationships between variables.

supervised

unsupervised



Typical Machine Learning Problems

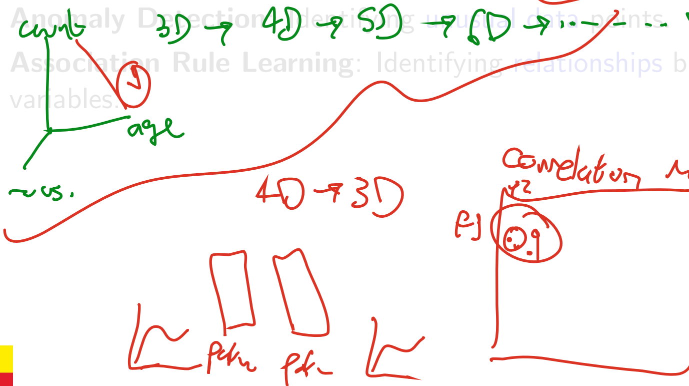
f1 f2 f3

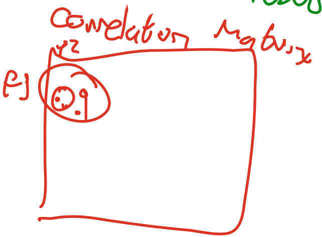
- **Classification**: Predicting a label.
- **Regression**: Predicting a continuous value.
- **Clustering**: Grouping similar data points.
- **Dimensionality Reduction**: Reducing the number of features.
- **Anomaly Detection**: Identifying unusual data points.
- **Association Rule Learning**: Identifying relationships between variables.

3D → 4D → 5D → 6D → ... → ∞ Dimensions

↓

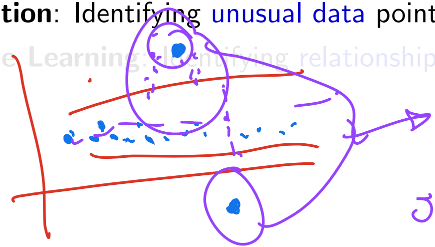
resources





Typical Machine Learning Problems

- **Classification**: Predicting a label.
- **Regression**: Predicting a continuous value.
- **Clustering**: Grouping similar data points.
- **Dimensionality Reduction**: Reducing the number of features.
- **Anomaly Detection**: Identifying unusual data points.
- **Association Rule Learning**: Identifying relationships between variables.



outliers

Support Vector Machines

Typical Machine Learning Problems

- **Classification**: Predicting a label.
- **Regression**: Predicting a continuous value.
- **Clustering**: Grouping similar data points.
- **Dimensionality Reduction**: Reducing the number of features.
- **Anomaly Detection**: Identifying unusual data points.
- **Association Rule Learning**: Identifying relationships between variables.

language

↳ a words sequence

correlation

$$\left.\begin{matrix} f_1 \\ f_2 \end{matrix}\right\} \rightarrow f_3$$

The Machine Learning Workflow

- **Data Collection**: Gathering the data.
- **Data Preprocessing**: Cleaning and preparing the data.
- **Feature Engineering**: Creating new features.
- **Model Selection**: Choosing the best model.
- **Model Training**: Training the model on the data.
- **Model Evaluation**: Assessing the model's performance.
- **Model Deployment**: Putting the model into production.

J

APIs - Documents - DB ----

NoSQL

Datalake (raw)

ETL (schedule)

SQL

DataWareHouse

The Machine Learning Workflow

- **Data Collection**: Gathering the data.
- **Data Preprocessing**: Cleaning and preparing the data.
- **Feature Engineering**: Creating new features.
- **Model Selection**: Choosing the best model.
- **Model Training**: Training the model on the data.
- **Model Evaluation**: Assessing the model's performance.
- **Model Deployment**: Putting the model into production.

High quality

Null?

Outliers?

correlation?

features?

new features?

transform

selection

Format?

The Machine Learning Workflow

- **Data Collection**: Gathering the data.
- **Data Preprocessing**: Cleaning and preparing the data.
- **Feature Engineering**: Creating new features.
- **Model Selection**: Choosing the best model.
- **Model Training**: Training the model on the data.
- **Model Evaluation**: Assessing the model's performance.
- **Model Deployment**: Putting the model into production.

mix →

math ← statistics

category

dist

money

risks

0

1

2

{ unsupervised

i → dist[]

1 →

2 →

⋮

[]

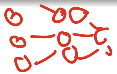
Data Engineer

Data Analyst

Machine Learning

Data Scientist

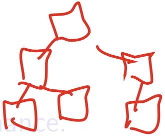
The Machine Learning Workflow



ax+b → 50
100
200

- **Data Collection**: Gathering the data.
- **Data Preprocessing**: Cleaning and preparing the data.
- **Feature Engineering**: Creating new features.
- **Model Selection**: Choosing the best model.
- **Model Training**: Training the model on the data.
- **Model Evaluation**: Assessing the model's performance.
- **Model Deployment**: Putting the model into production.

Depends on data → Trees



cluster → unsupervised

AutoML

experience

The Machine Learning Workflow

- **Data Collection**: Gathering the data.
- **Data Preprocessing**: Cleaning and preparing the data.
- **Feature Engineering**: Creating new features.
- **Model Selection**: Choosing the best model.
- **Model Training**: Training the model on the data.
- **Model Evaluation**: Assessing the model's performance.
- **Model Deployment**: Putting the model into production.

onehot encoding

$$\begin{pmatrix} M \\ R \\ G \end{pmatrix} \rightarrow \begin{array}{ccc} M & R & G \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}$$

8 - train

2 - evaluate

Stewart

8k

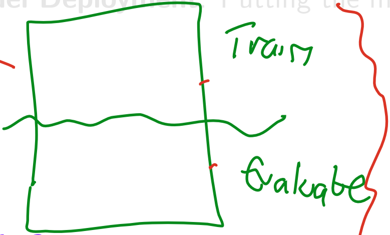
2k

1k

k.

1m

2m.



10 6003 → groups

syntetic

%≈%

numeric

3x3=9

3x4=12

12987x 4532=

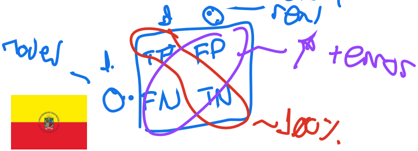
The Machine Learning Workflow

- **Data Collection**: Gathering the data.
- **Data Preprocessing**: Cleaning and preparing the data.
- **Feature Engineering**: Creating new features.
- **Model Selection**: Choosing the best model.
- **Model Training**: Training the model on the data.
- **Model Evaluation**: Assessing the model's performance.
- **Model Deployment**: Putting the model into production.

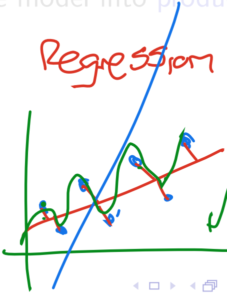
Metrics => Error

Classification

- Matrix Cofusion



Regression



↑ distances

$\sqrt{d_1^2+d_2^2}$ ~

The Machine Learning Workflow

Do SW

- **Data Collection**: Gathering the data.
- **Data Preprocessing**: Cleaning and preparing the data.
- **Feature Engineering**: Creating new features.
- **Model Selection**: Choosing the best model.
- **Model Training**: Training the model on the data.
- **Model Evaluation**: Assessing the model's performance.
- **Model Deployment**: Putting the model into production.

a = " "
b = " "
c = " "

Notebook → classes → OOP → API Rest → Pickle

object

Docker → Cloud (serverless) → MLOps

Parameters → Features

Examining the Data

- **Data Exploration**: Understanding the data.
- **Data Cleaning**: Preparing the data.
- **Feature Engineering**: Creating new features.
- **Feature Selection**: Selecting the most important features.
- **Data Preprocessing**: Preparing the data for modeling.

bias? → noise

categorical? numerical?

impact? → business

text → word cloud

keywords

10-15%

↓

Descriptive Analytics

Examining the Data

- **Data Exploration**: Understanding the data.
- **Data Cleaning**: Preparing the data.
- **Feature Engineering**: Creating new features.
- **Feature Selection**: Selecting the most important features.
- **Data Preprocessing**: Preparing the data for modeling.

null?

50-60;

imputing? → imputes

fill

?

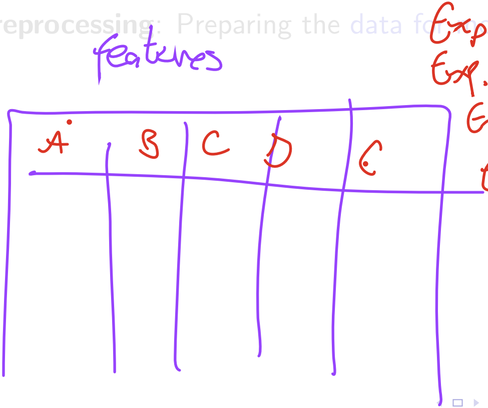
Examining the Data

- **Data Exploration**: Understanding the data.
- **Data Cleaning**: Preparing the data.
- **Feature Engineering**: Creating new features.
- **Feature Selection**: Selecting the most important features.
- **Data Preprocessing**: Preparing the data for modeling.

Birthday	Country	IsDrunk?
2008	USA	0
2005	Colombia	1

Examining the Data

- **Data Exploration**: Understanding the data.
- **Data Cleaning**: Preparing the data.
- **Feature Engineering**: Creating new features.
- **Feature Selection**: Selecting the most important features.
- **Data Preprocessing**: Preparing the data for modeling.



Exp 1.	ABC
Exp. 2.	ABCDE
Exp. 3	ACE
Exp 4	BD

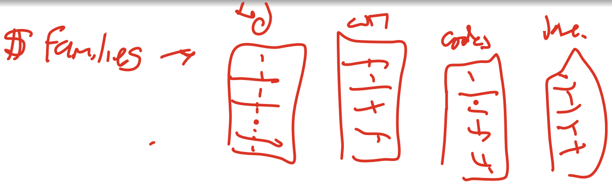
Examining the Data

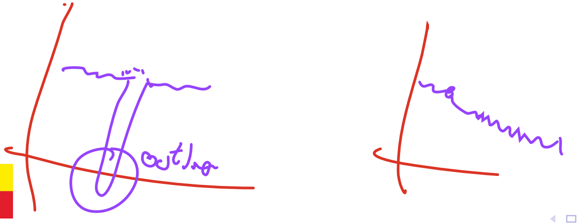
- **Data Exploration**: Understanding the data.
- **Data Cleaning**: Preparing the data.
- **Feature Engineering**: Creating new features.
- **Feature Selection**: Selecting the most important features.
- **Data Preprocessing**: Preparing the data for modeling.

Categoric
Text } → Numeric → Vector

Algorithmic Bias

- **Algorithmic bias** is a systematic error in a model that results in unfair outcomes.
- It can be caused by biased training data, biased algorithms, or biased decision-making.





Outline

1. Fundamentals of Machine Learning
2. Python Tools for Machine Learning
3. Supervised Machine Learning
4. Machine Learning Models Evaluation

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Python Tools for Machine Learning

~97

Guido Von Rossum

Python Tools

- **NumPy**: A library for numerical computing. → Travis Oliphant
- **Pandas**: A library for data manipulation and analysis.
- **Matplotlib**: A library for data visualization.
- **Scikit-learn**: A library for machine learning.

Vectorization → no loops

→ linear algebra

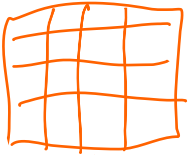
C

arrays

matriz (GPU)

→ operations

Tabular Data



Dataframe (df)

→ Analysis

SQL (ORM)

Time Series

Scatter

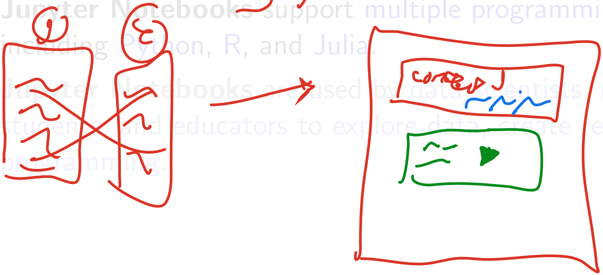
Jupyter Notebooks

COLAB

- **Jupyter Notebooks** are a web-based interactive computing environment that allows you to create and share documenmts that contain live code, equations, visualizations, and narrative text.
- **Jupyter Notebooks** are used for data cleaning, data transformation, statistical modeling, data visualization, machine learning, and more.

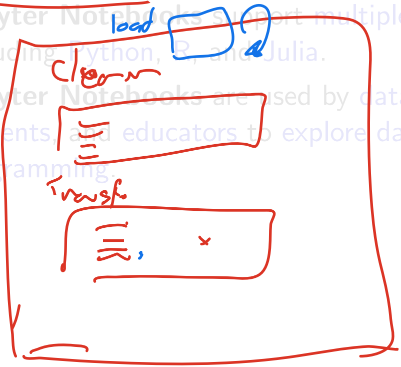
Carol Willing / Fernando Peréz → ipython

- **Jupyter Notebooks** support multiple programming languages, including Python, R, and Julia.
- **Jupyter Notebooks** are used by data scientists, researchers, students, and educators to explore data, create reports, and teach programming.



Jupyter Notebooks

- **Jupyter Notebooks** are a web-based interactive computing environment that allows you to create and share documenmts that contain live code, equations, visualizations, and narrative text.
- **Jupyter Notebooks** are used for data cleaning, data transformation, statistical modeling, data visualization, machine learning, and more.
- **Jupyter Notebooks** support multiple programming languages, including Python, R, and Julia.
- **Jupyter Notebooks** are used by data scientists, researchers, students, and educators to explore data, create reports, and teach programming.



Jupyter Notebooks

- **Jupyter Notebooks** are a web-based interactive computing environment that allows you to create and share documenmts that contain live code, equations, visualizations, and narrative text.
- **Jupyter Notebooks** are used for data cleaning, data transformation, statistical modeling, data visualization, machine learning, and more.
- **Jupyter Notebooks** support multiple programming languages, including Python, R, and Julia.
- **Jupyter Notebooks** are used by data scientists, researchers, students, and educators to explore data, create reports, and teach programming.

Jupyter

Popular — Data Science

Jupyter Notebooks

- **Jupyter Notebooks** are a web-based interactive computing environment that allows you to create and share documenmts that contain live code, equations, visualizations, and narrative text.
- **Jupyter Notebooks** are used for data cleaning, data transformation, statistical modeling, data visualization, machine learning, and more.
- **Jupyter Notebooks** support multiple programming languages, including Python, R, and Julia.
- **Jupyter Notebooks** are used by data scientists, researchers, students, and educators to explore data, create reports, and teach programming.

Machine learning Introduction

Lambda Functions

f x: x+2

Definition

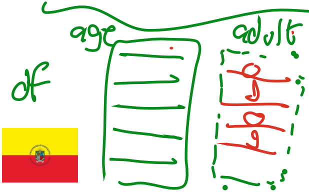
A **lambda function** is a small anonymous function. A **lambda function** can take any number of arguments, but can only have one expression.

lambda arguments : expression

↳ lambda (arguments){
return expression;
}

x = lambda a: a*2

print(x(10)) → 20



df['adult'] = lambda df['age']: 1 if x≥18 else 0

Numerical Python Library — Numpy

- **Numpy** is the core library for scientific computing in Python. It is the fundamental package for scientific computing with Python.
- **Numpy** is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

vectors

- **Numpy** was created by Travis Oliphant in 2005, and it is an open-source project. Coming soon, Numpy version 2.0 will be released.

2.1

Lineal Algebra with Numpy

- **Numpy** provides a comprehensive set of linear algebra functions.
- **Numpy** provides the functionality to create and manipulate matrices.
- **Numpy** provides the functionality to solve linear systems of equations.
- **Numpy** provides the functionality to calculate the determinant of a matrix.
- **Numpy** provides the functionality to calculate the inverse of a matrix.

Eigen

Soli Equations

$$\begin{bmatrix} 2 & 1 \\ 3 & 5 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Lineal Algebra with Numpy

- **Numpy** provides a comprehensive set of linear algebra functions.
- **Numpy** provides the functionality to create and manipulate matrices.
- **Numpy** provides the functionality to solve linear systems of equations.
- **Numpy** provides the functionality to calculate the determinant of a matrix.
- **Numpy** provides the functionality to calculate the inverse of a matrix.

Lineal Algebra with Numpy

- **Numpy** provides a comprehensive set of linear algebra functions.
- **Numpy** provides the functionality to create and manipulate matrices.
- **Numpy** provides the functionality to solve linear systems of equations.
- **Numpy** provides the functionality to calculate the determinant of a matrix.
- **Numpy** provides the functionality to calculate the inverse of a matrix.

Lineal Algebra with Numpy

- **Numpy** provides a comprehensive set of linear algebra functions.
- **Numpy** provides the functionality to create and manipulate matrices.
- **Numpy** provides the functionality to solve linear systems of equations.
- **Numpy** provides the functionality to calculate the determinant of a matrix.
- **Numpy** provides the functionality to calculate the inverse of a matrix.

Lineal Algebra with Numpy

- **Numpy** provides a comprehensive set of linear algebra functions.
- **Numpy** provides the functionality to create and manipulate matrices.
- **Numpy** provides the functionality to solve linear systems of equations.
- **Numpy** provides the functionality to calculate the determinant of a matrix.
- **Numpy** provides the functionality to calculate the inverse of a matrix.

Vectorization with Numpy

- **Vectorization** is the process of converting an algorithm from operating on a single value at a time to operating on a set of values at one time.
- **Vectorization** is the process of replacing explicit loops with array expressions or matrix operations.
- The advantages of vectorization are speed and clarity. The disadvantages are memory and complexity.
- **Numpy** provides the functionality to vectorize operations on arrays.

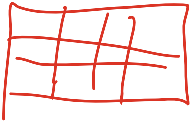
for i - - -
+memory
-compute

Vectorization with Numpy

- **Vectorization** is the process of converting an algorithm from operating on a single value at a time to operating on a set of values at one time.
- **Vectorization** is the process of replacing explicit loops with array expressions or matrix operations.
- The advantages of vectorization are speed and clarity. The disadvantages are memory and complexity.
- **Numpy** provides the functionality to vectorize operations on arrays.

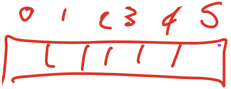
Introduction to Pandas

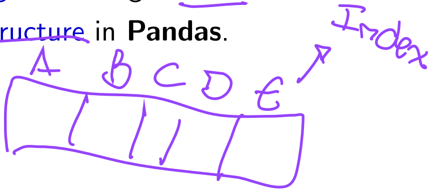
- **Pandas** is a fast, powerful, flexible, and easy-to-use open-source data manipulation and data analysis library built on top of the Python programming language.
- **Pandas** is a high-level data manipulation tool developed by Wes McKinney in 2008. → Polars (Rust)
- **Pandas** is a fast and efficient data manipulation tool that is built on top of NumPy. → numerical
- **Pandas** is one of the most popular and widely-used data manipulation libraries in the world.

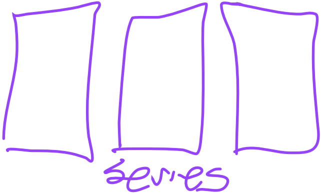


The "Series" Data Structure

- A **Series** is a one-dimensional array-like object that contains a sequence of values and an associated array of data labels, called the index.
- The index of a **Series** is an array of labels that correspond to the values in the **Series**. The index of a **Series** is an optional parameter that defaults to a sequence of integers starting at zero.
- The **Series** object is a core data structure in **Pandas**.



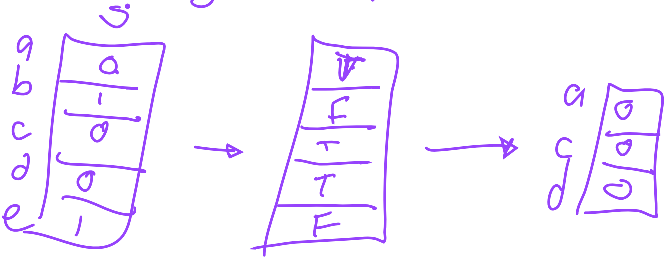




Querying a Series

- You could query a **Series** using indexing (boolean or fancy).
- You could query a **Series** using loc and iloc indexers.

$s = s[cond] \rightarrow x=0$



Querying a Series

- You could query a **Series** using indexing (boolean or fancy).
- You could query a **Series** using loc and iloc indexers.

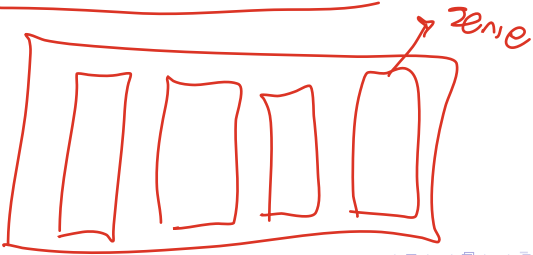
condition

index

0 - [n-1]

The "DataFrame" Data Structure

- A **DataFrame** is a two-dimensional labeled data structure with columns of potentially different types. ≈ matrix
- A **DataFrame** is a tabular data structure that is similar to a spreadsheet or a SQL table. → operations
- A **DataFrame** is a core data structure in **Pandas**. It is a two-dimensional size-mutable data structure with labeled axes (rows and columns).
- A **DataFrame** is a container for Series objects.



DataFrame Indexing and Loading

- You could index a **DataFrame** using column names.
- You could load a **DataFrame** from a CSV file.
- You could load a **DataFrame** from a JSON file.
- You could load a **DataFrame** from a SQL database.

DataFrame Indexing and Loading

- You could index a **DataFrame** using column names.
- You could load a **DataFrame** from a CSV file.
- You could load a **DataFrame** from a JSON file.
- You could load a **DataFrame** from a SQL database.

DataFrame Indexing and Loading

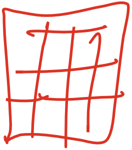
- You could index a **DataFrame** using column names.
- You could load a **DataFrame** from a CSV file.
- You could load a **DataFrame** from a JSON file.
- You could load a **DataFrame** from a SQL database.

different

null

DataFrame Indexing and Loading

- You could index a **DataFrame** using column names.
- You could load a **DataFrame** from a CSV file.
- You could load a **DataFrame** from a JSON file.
- You could load a **DataFrame** from a SQL database.



DateTime Handling in Pandas

- You could convert a string to a datetime object using the to_datetime() method.

Y-M-d H:M:S

- You could convert a datetime object to a string using the strftime() method.
- You could convert a datetime object to a timestamp using the timestamp() method.

DateTime Handling in Pandas

- You could convert a string to a datetime object using the to_datetime() method.
- You could convert a datetime object to a string using the strftime() method.
- You could convert a datetime object to a timestamp using the timestamp() method.

DateTime Handling in Pandas

- You could convert a string to a datetime object using the to_datetime() method.
- You could convert a datetime object to a string using the strftime() method.
- You could convert a datetime object to a timestamp using the timestamp() method.

Timestamp

ms

Y-M-d H:M:S → Y-M-d H:M:S.000 TZ

2025-06-18 10:00:00.000 Z → 0

Zulu

Querying a DataFrame

- You could query a **DataFrame** using indexing (boolean or fancy).
- You could query a **DataFrame** using loc and iloc indexers.
- You could query a **DataFrame** using query method.

Querying a DataFrame

- You could query a **DataFrame** using indexing (boolean or fancy).
- You could query a **DataFrame** using loc and iloc indexers.
- You could query a **DataFrame** using query method.

Querying a DataFrame

- You could query a **DataFrame** using indexing (boolean or fancy).
- You could query a **DataFrame** using loc and iloc indexers.
- You could query a **DataFrame** using query method.

Missing Values in a DataFrame

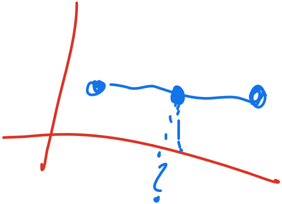
- You could detect missing values in a **DataFrame**. The isnull() method returns a Boolean **DataFrame** indicating the presence of missing values.

Clear

- You could fill missing values in a **DataFrame**. The fillna() method returns a **DataFrame** with missing values filled.

- You could drop missing values in a **DataFrame**. The dropna() method returns a **DataFrame** with missing values dropped.

Imputer → fill



Missing Values in a DataFrame

- You could detect missing values in a **DataFrame**. The isnull() method returns a Boolean **DataFrame** indicating the presence of missing values.
- You could fill missing values in a **DataFrame**. The fillna() method returns a **DataFrame** with missing values filled.
- You could drop missing values in a **DataFrame**. The dropna() method returns a **DataFrame** with missing values dropped.

Null → NaN

Missing Values in a DataFrame

- You could detect missing values in a **DataFrame**. The isnull() method returns a Boolean **DataFrame** indicating the presence of missing values.
- You could fill missing values in a **DataFrame**. The fillna() method returns a **DataFrame** with missing values filled.
- You could drop missing values in a **DataFrame**. The dropna() method returns a **DataFrame** with missing values dropped.

null → NaN → row at least 1

nan → gone

Merging DataFrames

- You could merge two **DataFrames** using the merge() method.
- You could concatenate two **DataFrames** using the concat() method.
- You could join two **DataFrames** using the join() method.

Join

concatenations

Merging DataFrames

- You could merge two **DataFrames** using the merge() method.
- You could concatenate two **DataFrames** using the concat() method.
- You could ~~join two~~ **DataFrames** using the join() method.

Merging DataFrames

- You could merge two **DataFrames** using the merge() method.
- You could concatenate two **DataFrames** using the concat() method.
- You could join two **DataFrames** using the join() method.

GroupBy in Pandas

- You could group a **DataFrame** using the groupby() method.
- You could aggregate a **DataFrame** using the agg() method.
- You could transform a **DataFrame** using the transform() method.
- You could filter a **DataFrame** using the filter() method.

GroupBy in Pandas

- You could group a **DataFrame** using the groupby() method.
- You could aggregate a **DataFrame** using the agg() method.
- You could transform a **DataFrame** using the transform() method.
- You could filter a **DataFrame** using the filter() method.

GroupBy in Pandas

- You could group a **DataFrame** using the groupby() method.
- You could aggregate a **DataFrame** using the agg() method.
- You could transform a **DataFrame** using the transform() method.
- You could filter a **DataFrame** using the filter() method.

GroupBy in Pandas

- You could group a **DataFrame** using the groupby() method.
- You could aggregate a **DataFrame** using the agg() method.
- You could transform a **DataFrame** using the transform() method.
- You could filter a **DataFrame** using the filter() method.

Outline

1. Fundamentals of Machine Learning

2. Python Tools for Machine Learning

3. Supervised Machine Learning

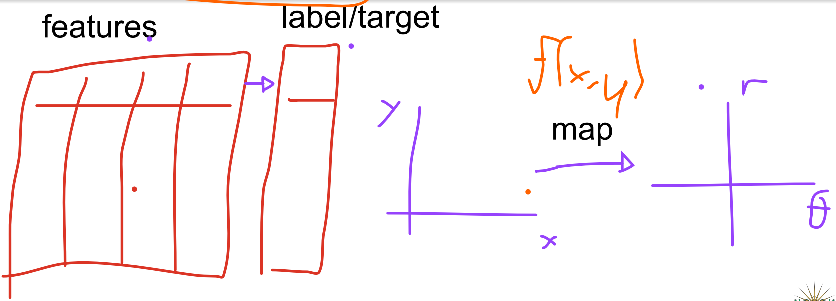
4. Machine Learning Models Evaluation

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Introduction to Supervised Machine Learning

Definition

- **Supervised learning** is a type of machine learning where the model is trained on labeled data.
- It involves training a model to map input data to output data based on example input-output pairs.



Overfitting and Underfitting

Overfitting

Overfitting occurs when a model learns the training data too well and performs poorly on new data.

Underfitting

Underfitting occurs when a model is too simple to capture the underlying structure of the data.

Overfitting and Underfitting

Overfitting

Overfitting occurs when a model learns the training data too well and performs poorly on new data.

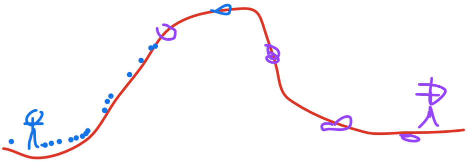
Underfitting

Underfitting occurs when a model is too simple to capture the underlying structure of the data.

Iterations

Learning Rate

Hill Climbing



Supervised Learning Datasets

- **Training Dataset**: The data used to train the model.
- **Validation Dataset**: The data used to tune the model hyperparameters.
- **Test Dataset**: The data used to evaluate the model performance.

60-70%

Learning a pattern to map
features -> label

Supervised Learning Datasets

- **Training Dataset**: The data used to train the model.
- **Validation Dataset**: The data used to tune the model hyperparameters.

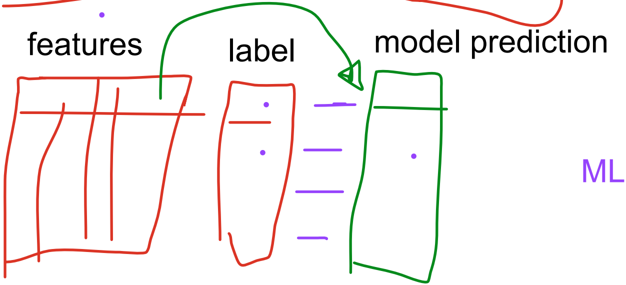
~10%

- **Test Dataset**: The data used to evaluate the model performance.

model setup
* iterations
* learning rate
* optimization metric (error)

Supervised Learning Datasets

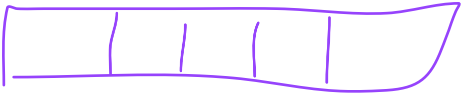
- **Training Dataset**: The data used to train the model.
- **Validation Dataset**: The data used to tune the model hyperparameters.
- **Test Dataset**: The data used to evaluate the model performance.



MLOps

Cross-Validation

- **Cross-validation** is a technique for assessing the performance of a model.
- It involves splitting the data into multiple subsets, training the model on some subsets, and evaluating it on others.
- Common cross-validation **techniques** include k-fold cross-validation and leave-one-out cross-validation. 123 4
- Cross-validation helps to reduce overfitting and provides a more accurate estimate of the model's performance.



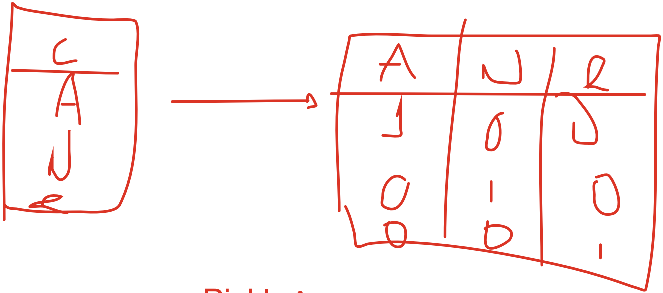
Training 1: 1234 Testing: 5

Training 2: 1235 Testing: 4

One-Hot Encoding

One-Hot Encoding

- **One-hot encoding** is a technique for converting categorical variables into numerical variables.
- It creates a binary vector for each `category`, with a 1 for the *category* and 0s for all other categories.



C
A
J
R

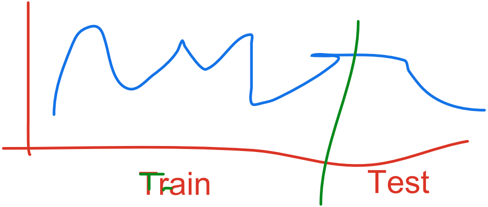
→

A	J	R
1	0	0
0	1	0
0	0	1

Pickle

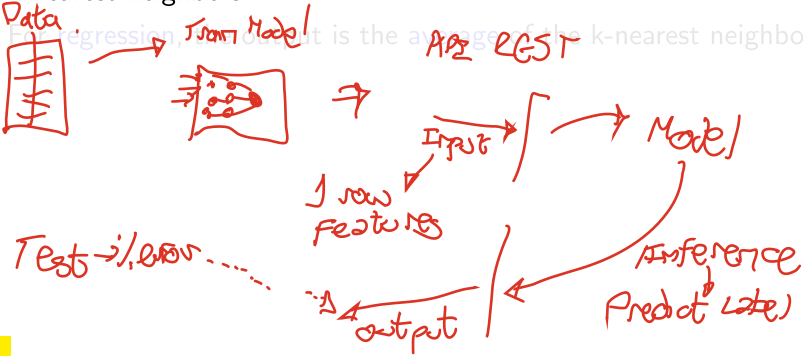
Data Leakage

- **Data leakage** occurs when information from the test set is inadvertently used to train the model.
- It can lead to overfitting and inflated performance metrics.
- Common sources of **data leakage** include target leakage, train-test contamination, and information leakage.
- To prevent **data leakage**, it is important to carefully separate the training and test data and avoid using information from the test set during training.



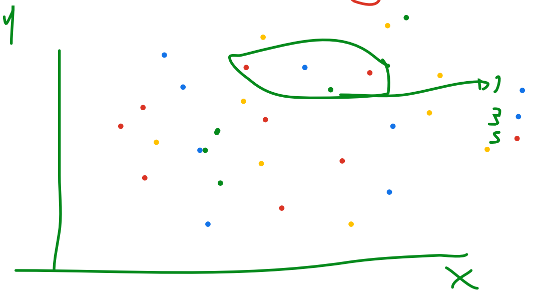
K-Nearest Neighbors: Classification and Regression

- **K-Nearest Neighbors (KNN)** is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure.
- It can be used for both classification and regression tasks.
- For classification, the output is the class label of the majority of the k-nearest neighbors.
- For regression, the output is the average of the k-nearest neighbors.



K-Nearest Neighbors: Classification and Regression

- **K-Nearest Neighbors (KNN)** is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure.
- It can be used for both classification and regression tasks.
- For classification, the output is the class label of the majority of the k-nearest neighbors.
- For regression, the output is the average of the k-nearest neighbors.



Linear Regression with Least Squares

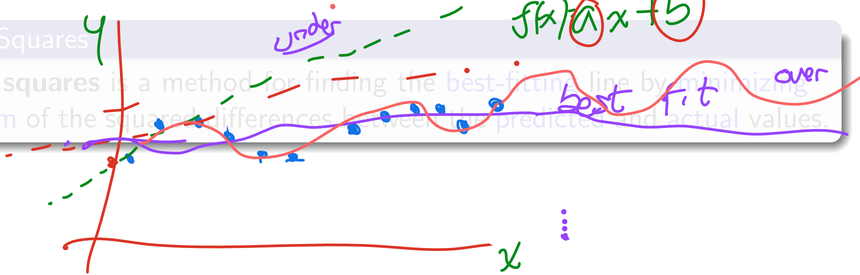
Linear Regression

- **Linear regression** is a type of regression analysis used for predicting the value of a continuous dependent variable.
- It works by finding the line that best fits the data.

$f(x)=\boxed{a}x+\boxed{b}$

Least Squares

Least squares is a method for finding the best-fitting line by minimizing the sum of the squared differences between the predicted and actual values.



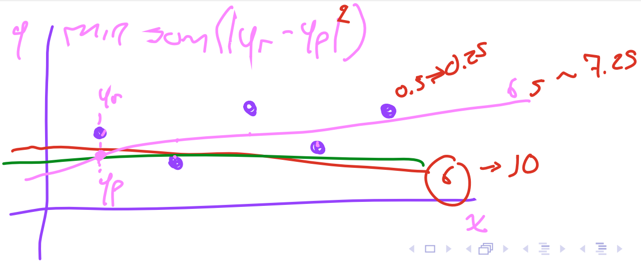
Linear Regression with Least Squares

Linear Regression

- **Linear regression** is a type of regression analysis used for predicting the value of a continuous dependent variable.
- It works by finding the line that best fits the data.

Least Squares

Least squares is a method for finding the best-fitting line by minimizing the sum of the squared differences between the predicted and actual values.



Ridge & Lasso

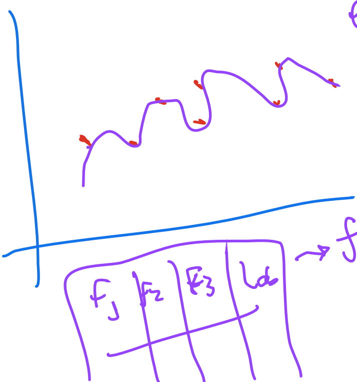
L1 → overfitting

L2 → feature importance

Ridge regression & **Lasso regression** are a type of linear regression that includes a penalty term to prevent overfitting. It works by adding a regularization term to the least squares objective function.

error = 0

penalty



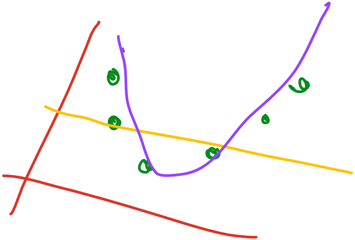
$$\rightarrow f(f_1,f_2,f_3) = \overset{\boxed{5}\uparrow}{a_1} f_1 + \underset{\boxed{2}}{a_2} f_2 + \overset{\boxed{0.5}\downarrow}{a_3} f_3 + b$$

Polynomial Regression

Polynomial Regression

- **Polynomial regression** is a type of regression analysis that models the relationship between the independent and dependent variables as an nth-degree polynomial.
- It can capture non-linear relationships between the variables.

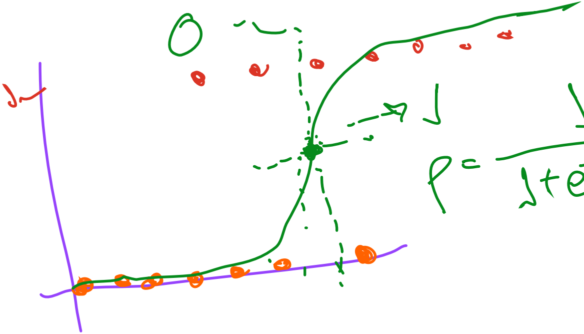
$f(x) = ax^2 + bx + c$



Logistic Regression

Logistic Regression

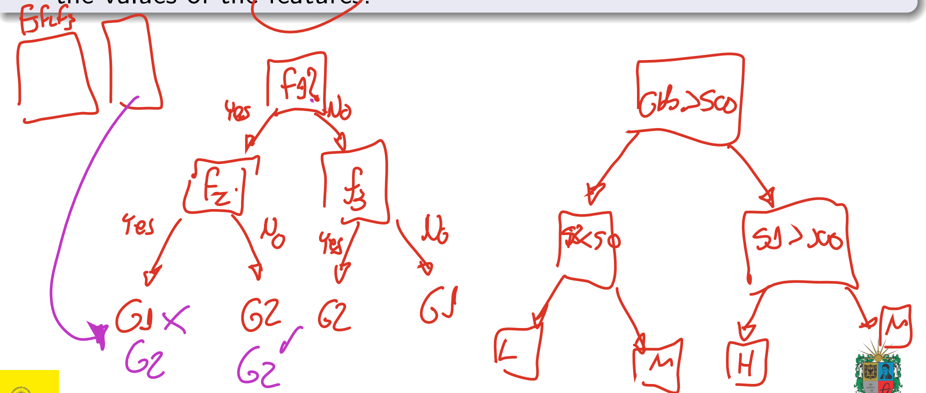
- **Logistic regression** is a type of regression analysis used for predicting the outcome of a categorical dependent variable.
- It is used for binary classification tasks, where the output is a probability between 0 and 1.



$$p = \frac{1}{1+e^{-(ax+b)}}$$

Decision Trees

- **Decision trees** are a type of machine learning model that can be used for both classification and regression tasks.
- They work by recursively partitioning the data into subsets based on the values of the features.



Naive Bayes Classifier

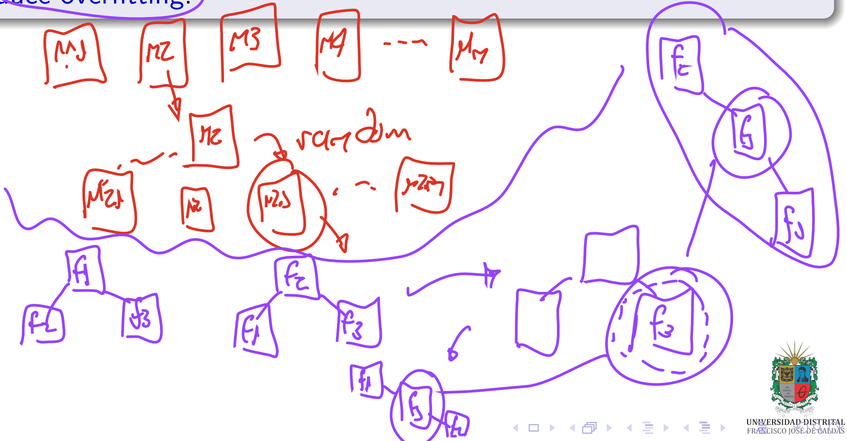
- The **naive Bayes classifier** is a simple probabilistic classifier based on Bayes' theorem.
- It assumes that the features are conditionally independent given the class label.

$P_{rain} \rightarrow P_{wind} \rightarrow P_{sun} \rightarrow P_{Abenpent}.$

Baseline / performance

Random Forest

- **Random forest** is an ensemble learning method that combines multiple decision trees to create a strong predictive model.
- It works by building multiple trees and averaging their predictions to reduce overfitting.



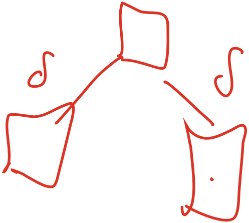
Gradient Boosted Decision Trees

- **Gradient boosted decision trees** are an ensemble learning method that combines multiple decision trees and gradient descendent optimization to create a strong predictive model.

random Forest

- They work by building trees sequentially, with each tree correcting the errors of the previous trees.

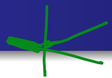
Bagging



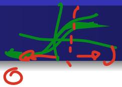
⇒ Min error

Neural Networks

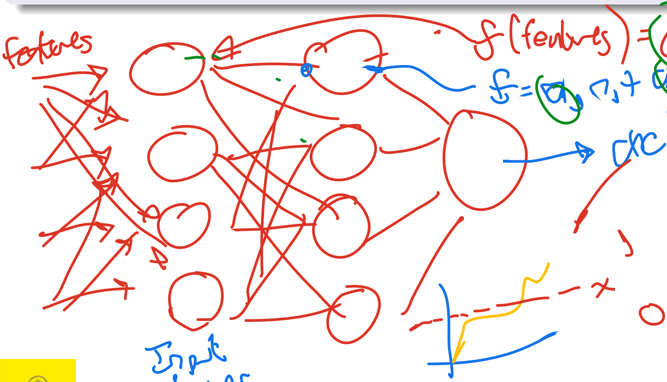
ReLu



Softmax



- **Neural networks** are a type of machine learning model inspired by the ~~human brain~~.
- They consist of layers ~~of interconnected nodes~~ that process input data and produce output data.



$f(\text{features}) = a_1 f_1 + a_2 f_2 + a_n f_n + b$

$f = a_1 n_1 + a_2 n_2 + a_n n_n + b$

activation function

parameters

Outline

1. Fundamentals of Machine Learning
2. Python Tools for Machine Learning
3. Supervised Machine Learning
4. Machine Learning Models Evaluation

Model Evaluation & Selection

- **Model Evaluation**: Assessing the performance of a model.
- **Model Selection**: Choosing the best model for the task.

	M1		M 2	
	error 1	error 2	error 1	error 2
Dataset 1	x_1	y_1	x_2	y_2
Dat. 2	x_3	y_3	x_4	y_4
Dat. 3	x_5	y_5	x_6	y_6

S-10 ⇒ average

error

predict vs expected

Confusion Matrices

↳ Classification

Definition

- A **confusion matrix** is a table that summarizes the performance of a classification model.
- It shows the number of true positives, true negatives, false positives, and false negatives.

		real (expected)	
		1	0
Predicted	1	True Positives → 1.0	False Positive → 0.0
	0	False Negative → 0.0	True Negatives → 1.0

Basic Evaluation Metrics

- **Accuracy**: The proportion of correct predictions.
- **Precision**: The proportion of true positives among all positive predictions.
- **Recall**: The proportion of true positives among all actual positives.
- **F1 Score**: The harmonic mean of precision and recall.

TP	FP
FN	TN

→ 100%

$$\to \text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \approx 1.0$$

→ Total samples test

Basic Evaluation Metrics

- **Accuracy**: The proportion of correct predictions.
- **Precision**: The proportion of true positives among all positive predictions.
- **Recall**: The proportion of true positives among all actual positives.
- **F1 Score**: The harmonic mean of precision and recall.

1

TP	FP
FN	TN

→ Precision = $\frac{TP}{TP + FP}$

FP → great

↳ TP less

↳ Pre ⇒ 0

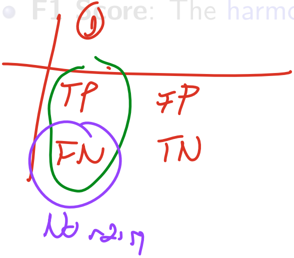
small

$\frac{N_1}{N_2}$ smaller

Pre ⇒ 1

Basic Evaluation Metrics

- **Accuracy**: The proportion of correct predictions.
- **Precision**: The proportion of true positives among all positive predictions.
- **Recall**: The proportion of true positives among all actual positives.
- **F1 Score**: The harmonic mean of precision and recall.



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

$$\downarrow$$

1.0

Basic Evaluation Metrics

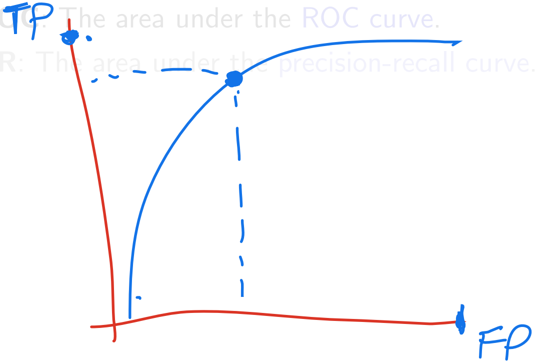
- **Accuracy**: The proportion of correct predictions.
- **Precision**: The proportion of true positives among all positive predictions.
- **Recall**: The proportion of true positives among all actual positives.
- **F1 Score**: The harmonic mean of precision and recall.

$\Rightarrow 2_x = 1$

$$\text{F1 Score} = \frac{\text{Precision} + \text{Recall}}{2 \times \text{Precision} \times \text{Recall}}$$

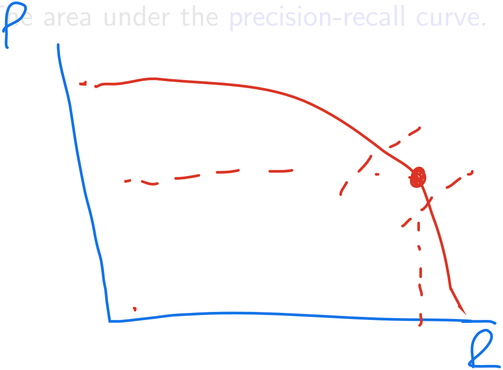
Classifier Metrics

- **ROC Curve**: A plot of the true positive rate against the false positive rate.
- **Precision-Recall Curve**: A plot of precision against recall.
- **AUC-ROC**: The area under the ROC curve.
- **AUC-PR**: The area under the precision-recall curve.



Classifier Metrics

- **ROC Curve**: A plot of the true positive rate against the false positive rate.
- **Precision-Recall Curve**: A plot of precision against recall.
- **AUC-ROC**: The area under the ROC curve.
- **AUC-PR**: The area under the precision-recall curve.



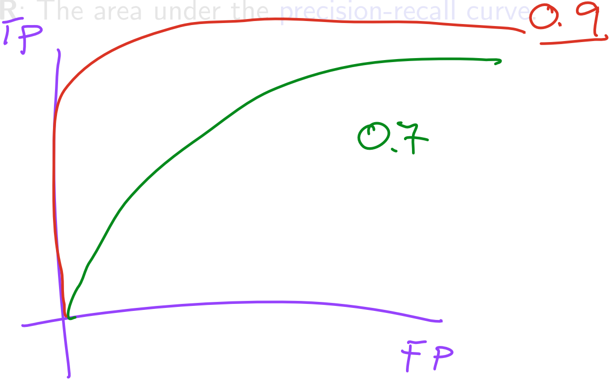
Classifier Metrics

- **ROC Curve**: A plot of the true positive rate against the false positive rate.

parameter → auc-roc

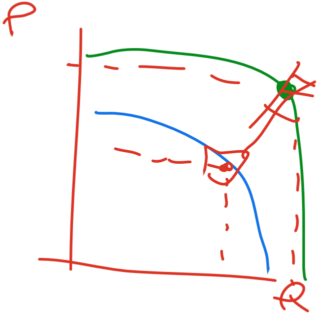
- **Precision-Recall Curve**: A plot of precision against recall.
- **AUC-ROC**: The area under the ROC curve.
- **AUC-PR**: The area under the precision-recall curve.

~ >0.75



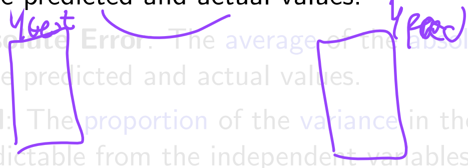
Classifier Metrics

- **ROC Curve**: A plot of the true positive rate against the false positive rate.
- **Precision-Recall Curve**: A plot of precision against recall.
- **AUC-ROC**: The area under the ROC curve.
- **AUC-PR**: The area under the precision-recall curve.



Regression Metrics

- **Mean Squared Error**: The average of the squared differences between the predicted and actual values.



- **Mean Absolute Error**: The average of the absolute differences between the predicted and actual values.
- **R-Squared**: The proportion of the variance in the dependent variable that is predictable from the independent variables.
- **Adjusted R-Squared**: A modified version of R-squared that adjusts for the number of predictors in the model.
- **Root Mean Squared Error**: The square root of the mean squared error.

$$MSE = \frac{(y_i^{pred} - y_i^{test})^2}{len(y)}$$

Regression Metrics

- **Mean Squared Error**: The average of the squared differences between the predicted and actual values.
- **Mean Absolute Error**: The average of the absolute differences between the predicted and actual values.
- **R-Squared**: The proportion of the variance in the dependent variable that is predictable from the independent variables.
- **Adjusted R-Squared**: A modified version of R-squared that adjusts for the number of predictors in the model.
- **Root Mean Squared Error**: The square root of the `mean squared error`.

$$mae = \frac{|y_{pred} - y_{test}|}{len(y)}$$

Regression Metrics

- **Mean Squared Error**: The average of the squared differences between the predicted and actual values.
- **Mean Absolute Error**: The average of the absolute differences between the predicted and actual values.
- **R-Squared**: The proportion of the variance in the dependent variable that is predictable from the independent variables.
- **Adjusted R-Squared**: A modified version of R-squared that adjusts for the number of predictors in the model.
- **Root Mean Squared Error**: The square root of the mean squared error.

features

y pred

Var (features)

(num range)

→ var (y pred)

→ corr

Regression Metrics

- **Mean Squared Error**: The average of the squared differences between the predicted and actual values.
- **Mean Absolute Error**: The average of the absolute differences between the predicted and actual values.
- **R-Squared**: The proportion of the variance in the dependent variable that is predictable from the independent variables.
- **Adjusted R-Squared**: A modified version of `R-squared` that adjusts for the number of predictors in the model.
- **Root Mean Squared Error**: The square root of the `mean squared error`.

features most significance

Regression Metrics

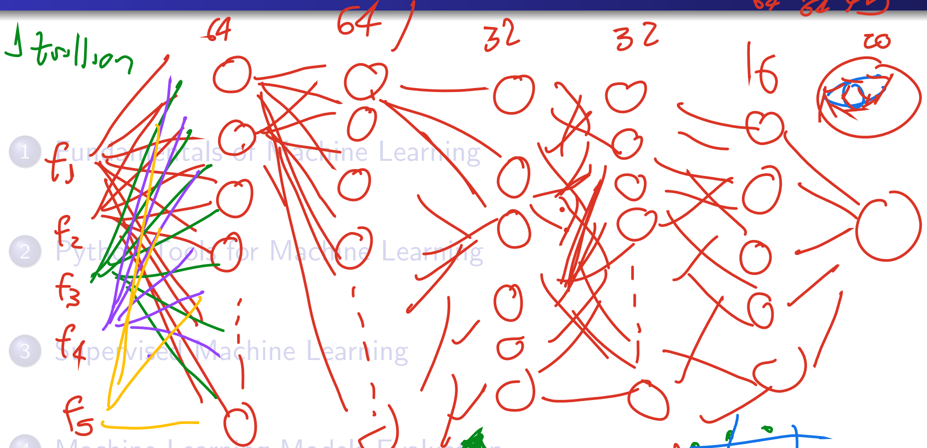
- **Mean Squared Error**: The average of the squared differences between the predicted and actual values.
- **Mean Absolute Error**: The average of the absolute differences between the predicted and actual values.
- **R-Squared**: The proportion of the variance in the dependent variable that is predictable from the independent variables.
- **Adjusted R-Squared**: A modified version of R-squared that adjusts for the number of predictors in the model.
- **Root Mean Squared Error**: The square root of the mean squared error.

$$rmse = \sqrt{\frac{\sum_{i=0}^{m} \left(y_i^{pred} - y_i^{test}\right)^2}{m}}$$

$\downarrow$ 0

Outline

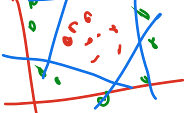
$f() = a_1 x_1 + a_2 x_2 + \dots + a_{64} x_{64} + b$



1. Fundamentals of Machine Learning
2. Python Tools for Machine Learning
3. Supervised Machine Learning
4. Machine Learning Models Evaluation

$f = a_1 f_1 + a_2 f_2 + a_3 f_3 + a_4 f_4 + a_5 f_5 + C$

LLM 7B



7000000000

Thanks!

Questions?

Repo: *https://github.com/EngAndres/ud-public/tree/main/courses/systems-sciences-foundations*