Machine Learning in Python

Introduction to Machine Learning

Cristian A. Marocico, A. Emin Tatar

Center for Information Technology
University of Groningen

Monday, June 30^{th} 2025

Outline

What is Machine Learning?

Machine Learning Definition

ML vs Traditional Programming:

What is Machine Learning?

Machine Learning Definition

Machine Learning is the study of algorithms that learn patterns from data to make predictions or decisions without being explicitly programmed.

ML vs Traditional Programming:

What is Machine Learning?

Machine Learning — Definition

Machine Learning is the study of algorithms that learn patterns from data to make predictions or decisions without being explicitly programmed.

ML vs Traditional Programming:

- Traditional: input + rules → output

What is Machine Learning?

Machine Learning — Definition

Machine Learning is the study of algorithms that learn patterns from data to make predictions or decisions without being explicitly programmed.

ML vs Traditional Programming:

- Traditional: input + rules $\rightarrow$ output
- ML: input + output $\rightarrow$ rules (the model)

The Machine Learning Workflow

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition
- Data Collection

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition
- Data Collection
- Train-Test Split

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition
- Data Collection
- Train-Test Split
- Data Exploration and Preprocessing

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition
- Data Collection
- Train-Test Split
- Data Exploration and Preprocessing
- Model Selection

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition
- Data Collection
- Train-Test Split
- Data Exploration and Preprocessing
- Model Selection
- Training and Validation

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition
- Data Collection
- Train-Test Split
- Data Exploration and Preprocessing
- Model Selection
- Training and Validation
- Evaluation

The Machine Learning Workflow

A typical Machine Learning workflow has the following structure:

- Problem Definition
- Data Collection
- Train-Test Split
- Data Exploration and Preprocessing
- Model Selection
- Training and Validation
- Evaluation
- Deployment and Monitoring

Train and Test Datasets

Splitting the dataset

Definition

Why Split?

Train and Test Datasets

Splitting the dataset

Definition

In machine learning, we split the dataset into two parts:

Why Split?

Train and Test Datasets

Splitting the dataset

Definition

In machine learning, we split the dataset into two parts:

- Training Set: Used to train the model.

Why Split?

Train and Test Datasets

Splitting the dataset
Definition

In machine learning, we split the dataset into two parts:

- Training Set: Used to train the model.
- Testing Set: Used to evaluate the model's performance.

Why Split?

Train and Test Datasets

Splitting the dataset — Definition

In machine learning, we split the dataset into two parts:

- Training Set: Used to train the model.
- Testing Set: Used to evaluate the model's performance.

Why Split?

- To ensure the model generalizes well to unseen data.

Train and Test Datasets

Splitting the dataset — Definition

In machine learning, we split the dataset into two parts:

- Training Set: Used to train the model.
- Testing Set: Used to evaluate the model's performance.

Why Split?

- To ensure the model generalizes well to unseen data.
- To prevent overfitting, where the model learns noise in the training data instead of the underlying pattern.

Train and Test Datasets

Splitting the dataset

Definition

In machine learning, we split the dataset into two parts:

- Training Set: Used to train the model.
- Testing Set: Used to evaluate the model's performance.

Why Split?

- To ensure the model generalizes well to unseen data.
- To prevent overfitting, where the model learns noise in the training data instead of the underlying pattern.
- The test dataset is put aside and not used until the end, when the model is evaluated.

Cross-Validation

The train dataset can be further split into training and validation sets to tune the model's hyperparameters and prevent overfitting.

Cross-Validation

Cross-Validation

Definition

Cross-Validation

Cross-Validation — Definition

Cross-validation is a technique to assess how the results of a statistical analysis will generalize to an independent dataset.

Cross-Validation

Cross-Validation

Definition

Cross-validation is a technique to assess how the results of a statistical analysis will generalize to an independent dataset.

- K-Fold Cross-Validation: The dataset is divided into K subsets. The model is trained on K-1 subsets and tested on the remaining subset. This process is repeated K times, with each subset used as the test set once.

Cross-Validation

Cross-Validation — Definition

Cross-validation is a technique to assess how the results of a statistical analysis will generalize to an independent dataset.

- K-Fold Cross-Validation: The dataset is divided into K subsets. The model is trained on K-1 subsets and tested on the remaining subset. This process is repeated K times, with each subset used as the test set once.
- Stratified K-Fold: Ensures that each fold has the same proportion of classes as the entire dataset, which is particularly useful for imbalanced datasets.

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) Definition

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) — Definition

Exploratory Data Analysis (EDA) is the process of analyzing datasets to summarize their main characteristics, often using visual methods.

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

EDA Techniques

Example

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

- To understand the data distribution and relationships between variables.

EDA Techniques

Example

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.

EDA Techniques

Example

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.
- To generate hypotheses and inform feature engineering.

EDA Techniques

Example

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.
- To generate hypotheses and inform feature engineering.

EDA Techniques

Example

Exploratory Data Analysis (EDA)

Purpose of EDA
Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.
- To generate hypotheses and inform feature engineering.

EDA Techniques
Example

- Descriptive Statistics: Mean, median, mode, standard deviation, etc.

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.
- To generate hypotheses and inform feature engineering.

EDA Techniques

Example

- Descriptive Statistics: Mean, median, mode, standard deviation, etc.
- Data Visualization: Histograms, scatter plots, box plots, etc.

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.
- To generate hypotheses and inform feature engineering.

EDA Techniques

Example

- Descriptive Statistics: Mean, median, mode, standard deviation, etc.
- Data Visualization: Histograms, scatter plots, box plots, etc.
- Correlation Analysis: Identifying relationships between variables.

Exploratory Data Analysis (EDA)

Purpose of EDA
Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.
- To generate hypotheses and inform feature engineering.

EDA Techniques
Example

- Descriptive Statistics: Mean, median, mode, standard deviation, etc.
- Data Visualization: Histograms, scatter plots, box plots, etc.
- Correlation Analysis: Identifying relationships between variables.
- Missing Value Analysis: Identifying and handling missing data.

Exploratory Data Analysis (EDA)

Purpose of EDA

Example

- To understand the data distribution and relationships between variables.
- To identify patterns, trends, and anomalies.
- To generate hypotheses and inform feature engineering.

EDA Techniques

Example

- Descriptive Statistics: Mean, median, mode, standard deviation, etc.
- Data Visualization: Histograms, scatter plots, box plots, etc.
- Correlation Analysis: Identifying relationships between variables.
- Missing Value Analysis: Identifying and handling missing data.
- Outlier Detection: Identifying and handling outliers in the data.

Data Preprocessing

Data Preprocessing — Definition

Data Preprocessing

Data Preprocessing — Definition

Data Preprocessing is the process of transforming raw data into a format suitable for analysis and modeling.

Data Preprocessing

Purpose of Data Preprocessing

Example

Common Data Preprocessing Techniques

Example

Data Preprocessing

Purpose of Data Preprocessing Example

- To clean the data by handling missing values, outliers, and inconsistencies

Common Data Preprocessing Techniques Example

Data Preprocessing

Purpose of Data Preprocessing
Example

- To clean the data by handling missing values, outliers, and inconsistencies
- To normalize or scale numerical features

Common Data Preprocessing Techniques
Example

Data Preprocessing

Purpose of Data Preprocessing Example

- To clean the data by handling missing values, outliers, and inconsistencies
- To normalize or scale numerical features
- To encode categorical variables into numerical formats

Common Data Preprocessing Techniques Example

Data Preprocessing

Purpose of Data Preprocessing

Example

- To clean the data by handling missing values, outliers, and inconsistencies
- To normalize or scale numerical features
- To encode categorical variables into numerical formats

Common Data Preprocessing Techniques

Example

- Handling Missing Values: Techniques include imputation (mean, median, mode), deletion, or using algorithms that can handle missing data.

Data Preprocessing

Purpose of Data Preprocessing

Example

- To clean the data by handling missing values, outliers, and inconsistencies
- To normalize or scale numerical features
- To encode categorical variables into numerical formats

Common Data Preprocessing Techniques

Example

- Handling Missing Values: Techniques include imputation (mean, median, mode), deletion, or using algorithms that can handle missing data.
- Outlier Detection and Treatment: Identifying outliers using statistical methods (e.g., Z-score, IQR) and deciding whether to remove or transform them.

Data Preprocessing

Purpose of Data Preprocessing

Example

- To clean the data by handling missing values, outliers, and inconsistencies
- To normalize or scale numerical features
- To encode categorical variables into numerical formats

Common Data Preprocessing Techniques

Example

- Handling Missing Values: Techniques include imputation (mean, median, mode), deletion, or using algorithms that can handle missing data.
- Outlier Detection and Treatment: Identifying outliers using statistical methods (e.g., Z-score, IQR) and deciding whether to remove or transform them.
- Feature Scaling: Normalization or standardization.

Data Preprocessing

Purpose of Data Preprocessing — Example

- To clean the data by handling missing values, outliers, and inconsistencies
- To normalize or scale numerical features
- To encode categorical variables into numerical formats

Common Data Preprocessing Techniques — Example

- Handling Missing Values: Techniques include imputation (mean, median, mode), deletion, or using algorithms that can handle missing data.
- Outlier Detection and Treatment: Identifying outliers using statistical methods (e.g., Z-score, IQR) and deciding whether to remove or transform them.
- Feature Scaling: Normalization or standardization.
- Encoding Categorical Variables: Techniques include one-hot encoding, label encoding, or using embeddings for high-cardinality categorical variables.

Feature Engineering

Feature Engineering Definition

Feature Engineering

Feature Engineering — Definition

Feature Engineering is the process of using domain knowledge to select, modify, or create features (variables) that improve the performance of machine learning models.

Feature Engineering

Purpose of Feature Engineering

Example

Common Feature Engineering Techniques

Example

Feature Engineering

Purpose of Feature Engineering — Example

- To improve model performance by providing more relevant information.

Common Feature Engineering Techniques — Example

Feature Engineering

Purpose of Feature Engineering

Example

- To improve model performance by providing more relevant information.
- To reduce dimensionality by selecting the most important features.

Common Feature Engineering Techniques

Example

Feature Engineering

Purpose of Feature Engineering

Example

- To improve model performance by providing more relevant information.
- To reduce dimensionality by selecting the most important features.
- To create new features that capture important patterns in the data.

Common Feature Engineering Techniques

Example

Feature Engineering

Purpose of Feature Engineering — Example

- To improve model performance by providing more relevant information.
- To reduce dimensionality by selecting the most important features.
- To create new features that capture important patterns in the data.

Common Feature Engineering Techniques — Example

- Feature Selection: Techniques include filter methods (e.g., correlation), wrapper methods (e.g., recursive feature elimination), and embedded methods (e.g., LASSO).

Feature Engineering

Purpose of Feature Engineering
Example

- To improve model performance by providing more relevant information.
- To reduce dimensionality by selecting the most important features.
- To create new features that capture important patterns in the data.

Common Feature Engineering Techniques
Example

- Feature Selection: Techniques include filter methods (e.g., correlation), wrapper methods (e.g., recursive feature elimination), and embedded methods (e.g., LASSO).
- Feature Transformation: Techniques include logarithmic transformation, polynomial features, and interaction terms.

Feature Engineering

Purpose of Feature Engineering

Example

- To improve model performance by providing more relevant information.
- To reduce dimensionality by selecting the most important features.
- To create new features that capture important patterns in the data.

Common Feature Engineering Techniques

Example

- Feature Selection: Techniques include filter methods (e.g., correlation), wrapper methods (e.g., recursive feature elimination), and embedded methods (e.g., LASSO).
- Feature Transformation: Techniques include logarithmic transformation, polynomial features, and interaction terms.
- Feature Creation: Creating new features based on existing ones, such as aggregating time series data, extracting date/time components, or creating domain-specific features.