Course: Applications of Artificial Intelligence and Machine Learning
Code: PROG74000
Professor/Instructor(s): Akrem El-ghazal

Assignment/Lab 3: Winter 2025

Hands-On with Real Data: Model Evaluation and Hyperparameter Tuning

Objectives

- Preprocess a real dataset to handle missing values and categorical variables.
- Explore feature scaling techniques.
- Train and evaluate machine learning models.
- Tune hyperparameters for a machine learning model.
- Use cross-validation techniques.
- Compare different machine learning models and choose the best one based on their performance metrics.

Dataset Description

This dataset is inspired by the Titanic dataset, where the goal is to predict whether a passenger survived or not based on various features. Understanding this dataset requires analyzing factors that could influence survival rates. The provided dataset (dataset.csv) includes the following features:

- **PassengerId**: Unique ID for each passenger.
- **Pclass**: Ticket class (1st, 2nd, 3rd), indicating which class of ticket a passenger held on the Titanic. The values are:
  - 1 = First Class
  - 2 = Second Class
  - 3 = Third Class
- **Gender**: Gender of the passenger
- **Age**: Age of the passenger.
- **Fare**: Ticket fare.
- **Embarked**: The Port of Embarkation in the Titanic dataset refers to the location where passengers boarded the ship. The dataset uses the following codes to represent these ports:
  - C = Cherbourg
  - Q = Queenstown
  - S = Southampton
- **FamilySize**: The total number of family members a passenger had on board.
- **Survived (Target)**: Whether the passenger survived (1) or not (0).

Your task is to explore and preprocess the data, train two classifiers (Decision Tree and KNN), tune their hyperparameters, and select the best-performing model based on evaluation metrics.

Instructions

Step 1: Load, Explore and Preprocess the Dataset (2 pts.)

In this step, your task is to explore and preprocess the dataset using your machine learning and visualization skills. This includes, but is not limited to:

- Handle missing values. Since the data is small, **do not remove any data**—you must impute any missing values.
- Scale the features if needed.
- Convert categorical variables.
- Use visualization to explore the dataset

Step 2: Evaluate a Decision Tree Classifier (2 pts.)

In this step, your goal is to use the Decision Tree classifier and tune its parameters so that your model will *perform well* on unseen data (the unseen data is not included in this lab). The parameters you are allowed to tune using `GridSearchCV`* from scikit-learn are:

- `max_depth`: The maximum depth of the tree.
- `min_samples_split`: The minimum number of samples required to split a node.
- `min_samples_leaf`: The minimum number of samples required to be in a leaf node.
- `criterion`: you have only *two options*: "`gini`" or "`entropy`".

For other parameters, use the default values given by scikit-learn.

The result of Step 2 is a Decision Tree model with its parameters and performance measures based on the chosen parameters.

You must implement a function named `test_Decision_tree(file_name, model)`, which follows these requirements:

- `file_name`: The name of the file containing the dataset (similar to the one provided).
- `model`: The best Decision Tree model you have designed and trained.

This function should:

- Load the dataset from `file_name`.
- Compute all relevant performance metrics based on the model (e.g., accuracy, recall, precision, etc.).
- Print the results.

I will use this function to evaluate your Decision Tree model on unseen data that I have.

* `GridSearchCV` is a technique in scikit-learn for tuning the hyperparameters of a machine learning model by performing an exhaustive search over a specified parameter grid.

Step 3: Evaluate a KNN Classifier (2 pts.)

In this step, your goal is to use the KNN classifier and tune its parameter (**k**) using `GridSearchCV` from scikit-learn, so that your model will *perform well* on unseen data (the unseen data is not included in this lab). For other parameters, use the default values given by scikit-learn.

The result of Step 3 is a KNN model with its parameter **k** and performance measures based on the chosen **k**.

You must implement a function named `test_knn(file_name, model)`, which follows these requirements:

- `file_name`: The name of the file containing the dataset (similar to the one provided).
- `model`: The best KNN model you have designed and trained.

This function should:

- Load the dataset from `file_name`.
- Compute all relevant performance metrics based on the model (e.g., accuracy, recall, precision, etc.).
- Print the results.

I will use this function to evaluate your KNN model on unseen data that I have.

Step 4: Reflection Questions (2 pts.)

1. Which model performed better, Decision Tree or KNN? Justify your answer.
2. How did feature scaling affect the Decision Tree and KNN performances?

The overall organization and Clarity of your solution (2 pts.)

- Ensure your solution is well-organized, clearly commented, and easy to follow
- In this lab the only limit is your imagination. Be creative \o/ (+2 pts.)

What to hand in?

Read the Assignment/Lab instructions section on the course shell.