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

Assignment/Lab 1: Winter 2025

Implementing and Testing the K-Nearest Neighbors (KNN) Model

Objectives

- Set up your Python development environment.
- Get familiar with commonly used Python packages.
- Implement the KNN algorithm from *scratch*.
- Use your implementation to perform regression on the datasets provided in a separate file.
- Evaluate the performance of your first machine learning algorithm.

Instructions

Step 0: Set up your Python development environment

1. Download and install Anaconda distribution platform
2. Create a virtual environment and name it "aimlcourse"
3. Install the following python packages:

- **Data Manipulation and Analysis:**
  - **NumPy:**
    For numerical computing and handling multi-dimensional arrays.
  - **Pandas:**
    For data manipulation and analysis with easy-to-use data structures like DataFrames.
  - **SciPy:**
    A library for scientific computing with modules for optimization, integration, interpolation, and more.
- **Data Visualization:**
  - **Matplotlib:**
    For creating static, interactive, and animated visualizations.
- **Core Machine Learning Libraries:**
  - **scikit-learn:**
    A powerful library for traditional machine learning algorithms, preprocessing, and evaluation metrics. Includes tools for regression, classification, clustering, and dimensionality reduction.

Step 1: Implement the KNN Algorithm (3 pts.)

Your task is to implement the KNN algorithm from *scratch* **without** using any machine learning libraries like `scikit-learn` for the core functionality. Follow these steps:

1. Create a **KNNRegressor** class with the following methods:

- fit (X, y): Train the algorithm for the given X and y. Where X : is the input features, y: is the output.
- predict (X): Predicts the target values for a given set of examples.
- You should add other methods or modify the input arguments for the methods above as needed.

2. In your implementation, you must set the Euclidean distance formula as the default method to compute distances between points.

3. Return the *average target value* of the **k** nearest neighbors for regression.

Step 2: Load the Dataset (0.5 pt.)

You will receive files named ***training_dataset_lab-1.csv*** and ***validation_dataset_lab-1.csv*** containing the datasets. Perform the following:

1. Load the data from the provided csv files.
2. Ensure you understand the dataset before proceeding with your implementation. Use visualizations to gain better insights.
3. Preprocess the data if necessary (e.g., handle missing values).

Step 3: Train the KNN Model (0.5 pt.)

1. Initialize your KNNRegressor model with a value of k = 1.
2. Train the KNNRegressor model using the `fit` method with the provided training dataset.

Step 4: Test and Evaluate the Model (3 pts.)

1. Use the `predict` method to predict target values for the validation dataset.
2. Calculate the Root Mean Squared Error (RMSE) of your model.
3. Experiment with different values of `k` and record the RMSE for each k and select the best k.
4. Compare the nearest neighbour model (k =1) with the model which is corresponding to the best 'k' from step3. Use visualizations to gain better insights into the results.
4. Compare the result of your KNNRegressor with the result of KNeighborsRegressor from scikit-learn. Use visualizations to gain better insights into the results.

Setp-5: Answer the following questions in your Jupyter notebook: (1.5 pts.)

1. How does the choice of **k** affect the model's performance?
2. What challenges did you face while implementing the KNNRegressor algorithm?
3. How does the KNNRegressor algorithm handle noisy data?

The overall organization of your solution (1.5 pts.)

What to hand in?

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