Assignment 2 – Simple Digits Classifier

Description:

A classification algorithm is a type of supervised machine learning algorithm that learns from labeled training data to categorize new, unlabeled observations into predefined categories or classes. Many different classification algorithms exist. Through this assignment, you are going to explore multiple classification algorithms on digit classification tasks.

Purpose:

- Get familiar with the machine learning pipeline.
- Explore the scikit-learn library.
- Develop a KNN algorithm from scratch.
- Solve a given problem using supervised learning methods.

Directions:

For this assignment, you need to do two tasks: 1) create a KNN classifier from scratch and 2) try different classification methods using the scikit-learn library.

Below is a detailed instruction on what you may need to do.

- Dataset Preparation
  - Load the dataset using *sklearn.datasets.load_digits*.
    - More information about the function can be found at: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html#sklearn.datasets.load_digits
  - After loading the dataset, randomly shuffle it to split the data into training, validation, and testing sets.
    - Use 70% of the data for the train set, 15% for the dev set, and 15% for the test set
    - After shuffling the data, you must ensure the labels and images match.
      - You may want to use the random shuffle function provided by *Numpy*.
- KNN
  - Create a KNN model from scratch
    - Try at least three different distance functions and determine which one yields the best performance.
      - Some possible distance functions are: Euclidean Distance, Manhattan Distance, Minkowski Distance, Cosine Similarity, Mahalanobis Distance, etc.
      - You would need to implement the distance functions by yourself (online tutorials/references are allowed)
  - Training the model
    - Use the train set to train the model and the val set to determine the best K and distance metric.
      - Multiple K values need to be tested to find the optimal ones.
    - Each sample in the dataset is an 8x8 image (stored as an 8x8 matrix). For simplicity, you might reshape the 8x8 matrix to a 1x64 matrix (i.e., a vector) for computing the distances between samples.
      - Feel free to use any tricks to improve the model's performance, e.g., data normalization, different ways to represent features, etc.
  - Test the model

        - After selecting the optimal K value and distance metric, test the model using the test set.
- Other Classification Algorithms
    - Apply at least two other classification algorithms from the scikit-learn library to the same task.
    - Tune the hyperparameters to achieve optimal performance for each algorithm.
        - Use the Train and Val sets for this step
    - Once the optimal hyperparameters are decided, test the model on the test set.
- Submission
    - Submit a table like the one attached below in Blackboard, with the highest performance of each column highlighted in bold.

Model Name	Hyperparameters	Accuracy	F1 Score	Precision Score	Recall Score
KNN	K=4, Distance Function: Mahalanobis	0.82	0.74	**0.92**	0.63
Decision Tree	criterion='gini', splitter='best', max_depth=5, ...	0.81	**0.88**	0.8	0.76
...	...	**0.98**	0.8	0.77	**0.84**
...	...	...	...	...	...

    - Submit your code in Blackboard
        - You may save your Colab code as a PDF file
        - Make sure the code includes all the distance functions of the KNN and all the other models you tested (no need to show the code for all the hyperparameters)

Evaluation Criteria:

This assignment is worth 100 points.

- (50 pts) Implementation of KNN with at least three distance functions
- (40 pts) The other two ML models
- (10 pts) The result table