University of Nebraska Lincoln

Practical Machine Learning
ENGR 491/891

Programming Assignment 1

Spring 2022

Study of the K-Nearest Neighbors Model

Assignment Goals

The goal of this assignment is to study the K-Nearest Neighbors (K-NN) classifier model.

- **Part A**: Model optimization via feature selection & varying threshold
- **Part B**: Understanding the curse of dimensionality & the fundamental limitation of the K-NN model

---

Assignment Instructions

Note: You must use Scikit-Learn to create the K-NN model. You are allowed to use Python libraries such as Pandas, NumPy.

i. The code should be written in a Jupyter notebook. Use the following naming convention.
   <lastname>_ assignment1.ipynb
ii. The Jupyter notebook should be submitted via Canvas.

The programming code will be graded on **both implementation and correctness**.

Score Distribution

ENGR 891: 100 points

Part A: Classification of Structured Data

You will create a K-NN classifier (using Scikit-Learn) to perform binary classification on the following structured dataset.

Dataset:
The *KNNDataset.csv* has 250 samples and 300 features.

- Feature columns: column '0' to column '299'
- Target column: column 'target'

The 'id' column should not be used for classification. It doesn't represent a feature.

Pre-Processing: **[15 pts]**

- Load the CSV file as a Pandas DataFrame object.
- Create a data frame object for the features and another data frame object for the target.
- Convert the above two data frame objects into two NumPy arrays (you may use the NumPy "asarray" function).
- Convert the target array type into "int".
- Partition the dataset into training & test subsets: 80% training & 20% test (you may use Scikit_learn's train_test_split() function)

Experiments:
You will perform binary classification using K-NN models for the following experiments. Do hyperparameter tuning to determine optimal values for the following 3 hyperparameters: *n_neighbors*, *p*, and *weights*.

- Experiment 1) All features & no standardization. Report train accuracy, test accuracy, and test confusion matrix. **[10 pts]**

- Experiment 2) All features & standardization. Report train accuracy, test accuracy, and test confusion matrix). **[10 pts]**

- Experiment 3) A subset of the features & standardization: you may use Pearson's correlation coefficient to select features. However, it's up to you to decide the threshold for feature selection. Your goal would be to increase the test accuracy. You are free to try any combinations of the features. Report train accuracy, test accuracy, test precision, test recall, test F1 score, and test confusion matrix. **[10 pts]**

- Experiment 4) Generate the ROC curve and the Precision-Recall Curve for the model of experiment 3. Find the optimal threshold. Using the optimal threshold,

compute train accuracy, test accuracy, test precision, test recall, test F1 score, and test confusion matrix. [**10 pts**]

Answer the following question.

- Q-1) Which experiment has the highest test accuracy between experiment 1 and 2? **Why**? [**3 pts**]

Part B: Classification of Unstructured Data

You will create a K-NN classifier (using Scikit-Learn) to perform multi-class classification on the following unstructured dataset.

Dataset:
The CIFAR-10 dataset (Canadian Institute For Advanced Research) contains 60,000 32 x 32 color images in 10 different classes. The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class. You may directly load this dataset using the Keras API:
https://keras.io/api/datasets/cifar10/

The train set contains 50,000 images, and the test set contains 10,000 images.

Pre-processing: [**10 pts**]
You will need to perform some pre-processing steps. First step is to reshape the data. The dimension of the training set is 50000 x 32 x 32 x 3 and test set is 10000 x 32 x 32 x 3. Before you use this data for the K-NN model, you need to flatten each sample (i.e., 32 x 32 x 3 = 3072) such the dimension of training and test set becomes:

- 50000 x 3072
- 10000 x 3072

Use the NumPy "reshape" function for this purpose.
https://numpy.org/doc/stable/reference/generated/numpy.reshape.html

Example: X_train = X_train.reshape((X_train.shape[0], 3072))

The train and test labels will be loaded as a 1D column vector. You need to convert the 1D vector into a 1D array (because the sklearn K-NN model requires the label data to be a 1D array).

You may convert the label data into a 1D array by using the NumPy "ravel" function:
https://numpy.org/doc/stable/reference/generated/numpy.ravel.html

Example: y_train = y_train.ravel()

Finally, scale the data using the min-max scaling technique, i.e., by dividing the training & test data matrix by 255.0.

Example: X_train = X_train/255.0

Experiments:

- Experiment 5) Create a K-NN classifier (using Scikit-Learn) and perform multi-class classification. Report train accuracy, test accuracy, and test confusion matrix.

You are free to perform hyperparameter tuning using grid search. But note that due to the high dimension of the data, training a K-NN model for this data may require a **couple of hours**. So, hyperparameter tuning for a range of values might take more than 1 day. If you are curious and have patience, then go ahead and perform hyperparameter tuning. Otherwise, use the given values for the following hyperparameters:

- n_neighbors=5
- p=1

For the remaining hyperparameters use default values.

[20 pts]

Answer the following question.

- Q-2) To answer the questions below you need to compare the poor performance of your K-NN model on the CIFAR-10 dataset with the performance of a K-NN model on the MNIST handwritten digits image dataset. Click on the following link and observe the performance of a K-NN model on the MNIST dataset. The model obtained over 97% test accuracy. https://github.com/rhasanbd/K-Nearest-Neighbors-Learning-Without-Learning/blob/master/K-NN-6-Curse%20of%20Dimensionality.ipynb

  However, your K-NN model on the CIFAR-10 dataset would perform awfully poorly.

  a) Explain why your K-NN model was unable to obtain high test accuracy on the CIFAR-10 image classification problem.
  b) Why does a K-NN model perform accurately on the MNIST handwritten digits image classification problem? Following notebooks might be useful to answer this question: https://github.com/rhasanbd/Study-of-Analogy-based-Learning-Image-Classification
  c) Is it possible to achieve above 90% accuracy on the CIFAR-10 dataset using a K-NN model? Justify your answer.

  [5 + 5 + 2 pts]

Deliverables:

You will submit two artefacts.

- A single Jupyter notebook containing all experiments. For each experiment show the required performance measures. If it's convenient, you may split your experiments in two notebooks and submit. Your code must be clearly annotated. Also add header descriptions for each block.
- A PDF copy of the written report with the cover page (get it from Canvas). The report must include the following items.

a) Experiments 1 and 2 report: training accuracy, test accuracy, test confusion matrix, values of the following hyperparameters: *n_neighbors*, *p*, and *weights*
b) Experiment 3 report: training accuracy, test accuracy, test precision, test recall, test F1 score, test confusion matrix, values of the following hyperparameters: *n_neighbors*, *p*, and *weights*
c) Experiment 4 report: ROC curve, Precision-Recall curve, optimal threshold value, test accuracy, test precision, test recall, test F1 score, and test confusion matrix
d) Experiment 5 report: training accuracy, test accuracy, test confusion matrix, values of the following hyperparameters: *n_neighbors*, *p*, and *weights*
e) Answer to Q-1 & Q-2 (3 parts a, b, and c).