Experiment No. 3

Objective: Automate e-mail classification and response using k-NN classifier.

K-NN Classifier: KNN is a classification technique based on "learning by analogy." This method is based on the similarity between the tuples of the dataset. To calculate the similarity between the tuples of the dataset, similarity matrix or distance matrix like Euclidean, Manhattan method has been applied. KNN has been learning by analogy method so complete dataset can be considered as the training set, and similarity matrix has been applied between the test data set (single data point: Yi) and training dataset (all data points: Xi). After the calculation of similarity, we need to decide how many neighbors we consider as nearest neighbors, K is the number of the nearest neighbor. To determine the K, cross validation has been applied, first we take only one neighbor, K = 1, and perform KNN to the check the accuracy and error rate of the learner. Iterative process has been applied with different values of K to obtain the best value of K, and adopt the value of K which given satisfactory performance of the learner. This method has been completely based on the similarity matrix so we need to normalize all the attributes of a given dataset.

KNN is a lazy learner, in this algorithm learner waits for model construction (training) until we need to test the model with the test dataset. These types of learners do less work for training and more work for classification.

For the implementation of KNN Classifier following steps have been applied.

Let $\mathbf{X = (x_1, x_2, \ldots.., x_n)}$ as *training dataset* and $\mathbf{Y = (y_1, y_2, \ldots.., y_n)}$ as *test set*.

Step 1 *Handle the data*: first check the missing and outlier values and handle it, next normalize the dataset.

Step 2 *Similarity:* to make the prediction calculate the similarity between training set and test set using the distance matrix. Distance matrix has been applied according to the behavior of dataset.

Dist = $[\sum (x_i - y_i)^2]^{1/2}$

Step 3: *K similarities:* choose the number of K and sort the similarities of the training set with testing set up to K values.

Step 4: *Classification response:* take the top K-1 sorted values for similarity (in some cases take the majority) to obtain label (class) of the test set.

Step 5: *Accuracy:* calculate the accuracy of the algorithm using the different number for K using step 2, 3 and 4.

Step 6: When the satisfactory result obtained with a specific value of K, fix the value of K for the classification process.

Step 7: End

Implementation Steps:

```
1. Load the spam and ham emails
2. Remove common punctuation and symbols
3. Lowercase all letters
4. Remove stopwords (very common words like pronouns, articles, etc.)
5. Split emails into training email and testing emails
6. For each test email, calculate the similarity between it and all training emails
    6.1. For each word that exists in either test email or training email, count its frequency in both
emails
    6.2. calculate the euclidean distance between both emails to determine similarity
7. Sort the emails in ascending order of euclidean distance
8. Select the k nearest neighbors (shortest distance)
9. Assign the class which is most frequent in the selected k nearest neighbours to the new email
```

Dataset:

The email data set for spam and ham (normal email) is obtained from “The Enron-Spam datasets”. It can be found at http://nlp.cs.aueb.gr/software_and_datasets/Enron-Spam/index.html under Enron2. The data set we’re using contains 5857 emails. Every email is stored in a text file, and the text files are divided and stored into two folders, ham folder, and spam folder. This means that the emails are already labelled. Every text file will be loaded by the program, and each email will be read and stored as a string variable. Every distinct word inside the string will be counted as a feature.

Code:

Python / Colab File Shared.

https://classroom.google.com/w/NjE4OTg0MjUzMTU2/t/all