1 INTRODUCTION

K-Nearest-Neighbors (KNN) is a kind of method of supervised learning, used for classification and regression. In this report, KNN will be applied to classify the daily e-mails for detecting whether they are spams or hams. As the overflowing spams in mailboxes each day, the KNN algorithm can be adopted as a filter to roughly classify the spams from the mailbox to promote the working efficiency for the people. In order to demonstrate the whole process, the report has been divided into five parts. It will initially introduce the method to find an appropriate data set and the following section will illuminate the preprocessing of the data set before the KNN is applied to it. After that, it will estimate the whole model. Finally, it will give some suggestions for promoting the model.

2 ACCESS TO THE DATA SET

2.1 UCI Repository

UC Irvine machine learning repository is an archive maintaining 474 data sets including the data set used in this report named "Spambase Data Set" which is a classical data set mainly associated with the classification. The data set refers to
https://archive.ics.uci.edu/ml/datasets/spambase

2.2 Introduction to the Data Set

The samples of the spams are collected from postmaster and individuals having filed spam and samples of non-spam e-mails come from filed work and personal e-mails where should be noticed that the word "George" and number "650" are indicators of non-spam. These features are useful when constructing the filter. One would either must blind such non-spam indicators or get a very wide collection of non-spam to generate a general-purpose spam filter.

Information Values	Corresponding Features
48 continuous real [0,100]	percentage of words in the e-mail that match WORD
6 continuous real [0,100]	percentage of characters in the e-mail that match CHAR
1 continuous real [1,...]	average length of uninterrupted sequences of capital letters
1 continuous integer [1,...]	length of longest uninterrupted sequence of capital letters
1 continuous integer [1,...]	total number of capital letters in the e-mail
1 nominal {0,1}	whether the e-mail was considered spam (1) or not (0)

3 THE OPERATIONS OF THE DATA SET

3.1 Data Set Labeling

The data set should be preprocessed before applying KNN to it. The first step is to label each list according to the corresponding features using *pd.read_csv*.

word_freq_WORD	char_freq_CHAR	capital_run_length_average	capital_run_length_longest	capital_run_length_tota	spam
0.000	0.000	3.756	61	278	1
0.180	0.048	5.114	101	1028	1
0.184	0.010	9.821	485	2259	1
0.000	0.000	3.537	40	191	1
0.000	0.000	3.537	40	191	1

3.2 Data Set Separation

In order to use KNN, the data set should be split into four groups for training and testing separately, two with five features and two with labeling 'spam' or 'non-spam', respectively.

Firstly, it can use *scaler.fit(df.drop('spam', axis=1))* to split the names of group/class and names of features. The next step is to use *X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=101)* to split them for varied use.

4 APPLICATION OF KNN

4.1 Train the Data Set

The library 'sklearn.neighbors' has the function '*KNeighborsClassifier(K).fit(X_train, y_train)'* that can automatically train the data set using training data set with the value of K.

4.2 Prediction

From the same library, importing '*KNeighborsClassifier().predict(X_test)*' can estimate the training data set using the preparing testing data set. Following that is to use '*print(classification_report(y_test, pred))*' to observe the result.

	precision	recall	f1-score	support
0	0.84	0.89	0.87	561
1	0.82	0.74	0.77	360
micro avg	0.83	0.83	0.83	921
macro avg	0.83	0.82	0.82	921
weighted avg	0.83	0.83	0.83	921

Precision: How many selected items are relevant?
Recall: How many relevant items are selected?
F1: The Harmonic Mean of precision and recall.

```
example1 = np.array([0.000,0.000,3.756,61,278])
example1 = example1.reshape(1,-1)
prediction = knn.predict(example1)
print(prediction)

[1]
```

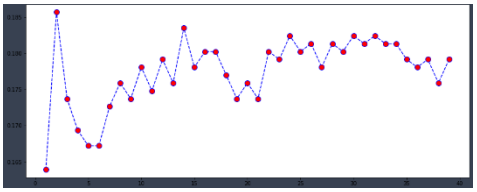
It can also test the data set by intersecting a new sample, and the return value is 'spam' in the shown example.

5 K VALUE DETERMINATION

The best way to find K is to repeatedly calculate (iteration) the error rate to find a minimum one. It also should be noticed that K should not less than two because spam and ham are two groups. The following code can help to find the best K value:

```
error_rate = []
for i in range(1,40):
    knn = KNeighborsClassifier(i)
    knn.fit(X_train,y_train)
    pred_i = knn.predict(X_test)
    error_rate.append(np.mean(pred_i != y_test))
```

The result is shown:



It can obviously observe that when k=5, it gains the best fitting accuracy (lowest error rate).

6 CONCLUSIONS

According to this report, K-Nearest-Neighbors can work for spam classifying efficiently. The accuracy is over 80% which means more than 80% spams from receiving e-mails can be filtered. This model can be used for the further test in the future using more data sets.

access to all the codes from: https://github.com/Timo9Madrid7/My-Spam-Classification