Spam_classification

November 11, 2018

```
In [64]: import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.multiclass import OneVsRestClassifier
         from sklearn.metrics import classification_report

In [65]: # Reading number of Data Points
         with open('PhishingData.arff') as file:
             line_count=0
             for line in file:
                 line_count = line_count+1
         file.close()
         # Defining Data Matrix
         w1 = 9
         w2 = 1
         h = line_count
         Data = np.zeros(shape=(h,w1))
         Label = np.zeros(h)
         length_of_training_set = h
         # Reading the Data Provided
         with open('PhishingData.arff') as file:
             line_count=0
             for line in file:
                 line = line.strip()
                 line = line.split(',')
                 for i in range(10):
                     if i<9:
                         Data[line_count][i] = int(line[i])
                     else:
                         Label[line_count] = int(line[i])

                 line_count = line_count+1
         file.close()

In [66]: # Split into Train - Test Set By Stratified Sampling
         # Label 1 Legitimate Label 0  Suspicious Label -1 Phishing
         X_train, X_test, y_train, y_test = train_test_split(Data, Label, test_size=0.20, random
         #print(X_train.shape)
```

```
        #print(X_test.shape)
        print("Total Instances: {}".format(Label.shape[0]))
        print("Instances in training set: {}".format(y_train.shape[0]))
        print("Instances in in test set: {}".format(y_test.shape[0]))
        print("Instances with legitimate label in dataset: {0:.2f}%".format(100*np.sum(Label==1
        print("Instances with suspicious label in dataset: {0:.2f}%".format(100*np.sum(Label==0
        print("Instances with phishing label in dataset: {0:.2f}%".format(100*np.sum(Label==-1)
        print("Instances with legitimate label in Training set: {0:.2f}%".format(100*np.sum(y_t
        print("Instances with suspicious label in Training set: {0:.2f}%".format(100*np.sum(y_t
        print("Instances with phishing label in Training set: {0:.2f}%".format(100*np.sum(y_tra
        print("Instances with legitimate label in Test set: {0:.2f}%".format(100*np.sum(y_test=
        print("Instances with suspicious label in Test set: {0:.2f}%".format(100*np.sum(y_test=
        print("Instances with phishing label in Test set: {0:.2f}%".format(100*np.sum(y_test==-
```

```
Total Instances: 1353
Instances in training set: 1082
Instances in in test set: 271
Instances with legitimate label in dataset: 40.50%
Instances with suspicious label in dataset: 7.61%
Instances with phishing label in dataset: 51.88%
Instances with legitimate label in Training set: 40.11%
Instances with suspicious label in Training set: 7.21%
Instances with phishing label in Training set: 52.68%
Instances with legitimate label in Test set: 42.07%
Instances with suspicious label in Test set: 9.23%
Instances with phishing label in Test set: 48.71%
```

```
In [67]: # We Try different classifiers and compare their accuracies

         # Classifier 1 Random Forest
         from sklearn.ensemble import RandomForestClassifier

         model = OneVsRestClassifier(RandomForestClassifier(n_estimators=100,max_depth=10,random
         model.fit(X_train,y_train)
         print("Training Accuracy:{0:.2f}%".format(100*model.score(X_train,y_train)))
         print("Training Accuracy:{0:.2f}%".format(100*model.score(X_test,y_test)))
         target_names = ['class -1', 'class 0', 'class 1']
         print(classification_report(y_test, model.predict(X_test), target_names=target_names))
```

```
Training Accuracy:96.77%
Training Accuracy:87.82%
             precision    recall  f1-score   support

   class -1       0.89      0.90      0.90       132
    class 0       0.83      0.76      0.79        25
    class 1       0.87      0.88      0.87       114
```

```
avg / total       0.88      0.88      0.88       271
```

```
In [68]: # Classifier 2 Normal Decision Trees

         from sklearn import tree

         model = OneVsRestClassifier(tree.DecisionTreeClassifier(random_state=41))
         model = model.fit(X_train, y_train)
         print("Training Accuracy:{0:.2f}%".format(100*model.score(X_train,y_train)))
         print("Test Accuracy:{0:.2f}%".format(100*model.score(X_test,y_test)))
         target_names = ['class -1', 'class 0', 'class 1']
         print(classification_report(y_test, model.predict(X_test), target_names=target_names))
```

```
Training Accuracy:96.77%
Test Accuracy:87.82%
             precision    recall  f1-score   support

   class -1       0.90      0.89      0.89       132
    class 0       0.88      0.84      0.86        25
    class 1       0.85      0.88      0.87       114

avg / total       0.88      0.88      0.88       271
```

```
In [69]: # Classifier 3 KNN
         from sklearn.neighbors import KNeighborsClassifier


         model = OneVsRestClassifier(KNeighborsClassifier(n_neighbors=3))
         model.fit(X_train, y_train)
         print("Training Accuracy:{0:.2f}%".format(100*model.score(X_train,y_train)))
         print("Test Accuracy:{0:.2f}%".format(100*model.score(X_test,y_test)))
         target_names = ['class -1', 'class 0', 'class 1']
         print(classification_report(y_test, model.predict(X_test), target_names=target_names))
```

```
Training Accuracy:93.44%
Test Accuracy:85.61%
             precision    recall  f1-score   support

   class -1       0.89      0.89      0.89       132
    class 0       0.71      0.60      0.65        25
    class 1       0.85      0.88      0.86       114

avg / total       0.85      0.86      0.85       271
```

```
In [70]: # Classifier 4 Gaussian Naive Bayes

         from sklearn.naive_bayes import GaussianNB
         model = OneVsRestClassifier(GaussianNB())
         model.fit(X_train, y_train)
         print("Training Accuracy:{0:.2f}%".format(100*model.score(X_train,y_train)))
         print("Training Accuracy:{0:.2f}%".format(100*model.score(X_test,y_test)))
         target_names = ['class -1', 'class 0', 'class 1']
         print(classification_report(y_test, model.predict(X_test), target_names=target_names))
```

```
Training Accuracy:83.64%
Training Accuracy:80.44%
             precision    recall  f1-score   support

   class -1       0.84      0.89      0.86       132
    class 0       0.00      0.00      0.00        25
    class 1       0.77      0.89      0.82       114

avg / total       0.73      0.80      0.77       271
```

```
In [71]: # Classifier 5 ANN
         from sklearn.neural_network import MLPClassifier

         model = OneVsRestClassifier(MLPClassifier(solver='lbfgs', alpha=1e-5,hidden_layer_sizes
         model.fit(X_train, y_train)
         print("Training Accuracy:{0:.2f}%".format(100*model.score(X_train,y_train)))
         print("Test Accuracy:{0:.2f}%".format(100*model.score(X_test,y_test)))
         target_names = ['class -1', 'class 0', 'class 1']
         print(classification_report(y_test, model.predict(X_test), target_names=target_names))
```

```
Training Accuracy:94.92%
Test Accuracy:87.08%
             precision    recall  f1-score   support

   class -1       0.89      0.92      0.90       132
    class 0       0.77      0.68      0.72        25
    class 1       0.87      0.86      0.86       114

avg / total       0.87      0.87      0.87       271
```

```
In [72]: # Classifier 6 Linear SVM
         from sklearn.svm import SVC


         model = OneVsRestClassifier(SVC(kernel='rbf',gamma=0.4))
```

```
model.fit(X_train, y_train)
print("Training Accuracy:{0:.2f}%".format(100*model.score(X_train,y_train)))
print("Test Accuracy:{0:.2f}%".format(100*model.score(X_test,y_test)))
target_names = ['class -1', 'class 0', 'class 1']
print(classification_report(y_test, model.predict(X_test), target_names=target_names))
```

```
Training Accuracy:94.27%
Test Accuracy:86.72%
             precision    recall  f1-score   support

   class -1       0.90      0.91      0.90       132
    class 0       0.78      0.56      0.65        25
    class 1       0.85      0.89      0.87       114

avg / total       0.86      0.87      0.86       271
```