Process_data_From_Blockchain

June 1, 2020

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        from sklearn.impute import SimpleImputer
        from sklearn.naive_bayes import GaussianNB, MultinomialNB
        from sklearn.ensemble import RandomForestClassifier
        from sklearn import svm
        from sklearn.preprocessing import StandardScaler
        from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, f1
            precision_score, roc_curve, auc, matthews_corrcoef
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LogisticRegression
```

```
In [2]: file_read = open("data_retrieved_from_blockchain.txt","r")
        input_data = file_read.readlines()
        # input_data[0]
```

```
In [3]: for i in range(len(input_data)):
            input_data[i] = (input_data[i].split(":")[1].split('"')[1]).split(',')
```

```
In [4]: input_data = np.array(input_data)
```

```
In [5]: # input_data[0].split('"')
```

```
In [6]: total_labels = input_data[:,-1].astype(np.float).astype(np.int)
        total_data  = input_data[:,:-1].astype(np.float)
```

```
In [7]: # total_labels
```

```
In [8]: # total_labels = total_labels.tolist()
        for i in range(len(total_labels)):
            if(total_labels[i] <= 3):
                total_labels[i] = 1
        # total_labels = np.array(total_labels)
```

```
In [9]: # total_labels
```

```
In [10]: # total_data
In [11]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(total_data, total_labels, test_si
```

```
In [12]: # for i in range(len(y_train)):
         #     if(y_train[i] == 1):
         #         y_train[i] = 4
         #     elif(y_train[i] == 4):
         #         y_train[i] = 1
         #     elif(y_train[i] == 5):
         #         y_train[i] = 4
```

```
In [13]: nbclf = GaussianNB()
         rfclf = RandomForestClassifier(n_estimators=100)
         svmclf = svm.SVC(kernel='linear') #Linear Kernel
```

```
In [14]: def classify_and_report(classifier,X_train, X_test, y_train, y_test):
             classifier.fit(X_train,y_train)
             y_predicted = classifier.predict(X_test)
             #print(confusion_matrix(Y_test, Y_predicted))
         #     print("Naive Bayes Classifier: \n")
             print("Multiclass classification: ")
             print('Accuracy:', accuracy_score(y_test, y_predicted))
             #print('F1 score:', f1_score(Y_test, Y_predicted,average='macro'))
             #print('F1 score:', f1_score(Y_test, Y_predicted,average='weighted'))
             print('F1 score:', f1_score(y_test, y_predicted,average='macro'))
             print('Recall:', recall_score(y_test, y_predicted,average='macro'))
             print('Precision:', precision_score(y_test, y_predicted,average='macro'))
             print("Matthews Correlation Coefficient: ",matthews_corrcoef(y_test, y_predicted)
             print('Classification report:', classification_report(y_test, y_predicted))
```

```
In [15]: print ("Naive Bayes")
         classify_and_report(nbclf,X_train, X_test, y_train, y_test)
```

```
Naive Bayes
Multiclass classification:
Accuracy: 0.8
F1 score: 0.5416666666666666
Recall: 0.625
Precision: 0.5833333333333333
Matthews Correlation Coefficient:  0.6968731476445666
Classification report:               precision    recall  f1-score   support

           1       1.00      1.00      1.00         6
           4       0.33      1.00      0.50         1
           5       0.00      0.00      0.00         1
           6       1.00      0.50      0.67         2

    accuracy                           0.80        10
   macro avg       0.58      0.62      0.54        10
```

```
weighted avg       0.83      0.80      0.78        10
```

```
/home/ashish/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: Unde
  'precision', 'predicted', average, warn_for)
/home/ashish/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: Unde
  'precision', 'predicted', average, warn_for)
/home/ashish/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: Unde
  'precision', 'predicted', average, warn_for)
```

```
In [16]: print ("Random Forest")
         classify_and_report(rfclf,X_train, X_test, y_train, y_test)
```

```
Random Forest
Multiclass classification:
Accuracy: 0.9
F1 score: 0.8333333333333333
Recall: 0.875
Precision: 0.875
Matthews Correlation Coefficient:  0.8448275862068966
Classification report:               precision    recall  f1-score   support

           1       1.00      1.00      1.00         6
           4       0.50      1.00      0.67         1
           5       1.00      1.00      1.00         1
           6       1.00      0.50      0.67         2

    accuracy                           0.90        10
   macro avg       0.88      0.88      0.83        10
weighted avg       0.95      0.90      0.90        10
```

```
In [17]: print ("SVM")
         classify_and_report(svmclf,X_train, X_test, y_train, y_test)
```

```
SVM
Multiclass classification:
Accuracy: 0.7
F1 score: 0.375
Recall: 0.5
Precision: 0.3333333333333333
Matthews Correlation Coefficient:  0.5360562674188974
Classification report:               precision    recall  f1-score   support

           1       1.00      1.00      1.00         6
           4       0.00      0.00      0.00         1
           5       0.33      1.00      0.50         1
           6       0.00      0.00      0.00         2

    accuracy                           0.70        10
   macro avg       0.33      0.50      0.38        10
weighted avg       0.63      0.70      0.65        10
```

```
/home/ashish/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: Unde
  'precision', 'predicted', average, warn_for)
/home/ashish/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: Unde
  'precision', 'predicted', average, warn_for)
/home/ashish/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: Unde
  'precision', 'predicted', average, warn_for)
```