In [1]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6
qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3aietf%3awg%3aoauth%3a2.0%
b&response_type=code&scope=email%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdocs.test%20https%3a%2
www.googleapis.com%2fauth%2fdrive%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly
ttps%3a%2f%2fwww.googleapis.com%2fauth%2fpeopleapi.readonly

Enter your authorization code:
··········
Mounted at /content/drive
```

In [2]:

```
import pandas as pd
df = pd.read_csv('/content/drive/My Drive/Train_data.csv')
print(df)
```

```
        duration protocol_type  ...  dst_host_srv_rerror_rate  xAttack
0              0          icmp  ...                      0.00   normal
1              0           udp  ...                      0.00   normal
2              0          icmp  ...                      0.00      dos
3              0          icmp  ...                      0.01   normal
4              0          icmp  ...                      0.00   normal
...          ...           ...  ...                       ...      ...
125968         0          icmp  ...                      0.00      dos
125969         8           udp  ...                      0.00   normal
125970         0          icmp  ...                      0.00   normal
125971         0          icmp  ...                      0.00      dos
125972         0          icmp  ...                      0.00   normal

[125973 rows x 42 columns]
```

In [3]:

```
X = df.iloc[:, :-1].values
print(X)
```

```
[[0 'icmp' 20 ... 0.0 0.05 0.0]
 [0 'udp' 45 ... 0.0 0.0 0.0]
 [0 'icmp' 50 ... 1.0 0.0 0.0]
 ...
 [0 'icmp' 55 ... 0.0 0.01 0.0]
 [0 'icmp' 31 ... 1.0 0.0 0.0]
 [0 'icmp' 20 ... 0.0 0.0 0.0]]
```

In [5]:

```
y = df.iloc[:, -1].values
print(y)
```

```
['normal' 'normal' 'dos' ... 'normal' 'dos' 'normal']
```

In [6]:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder_X = LabelEncoder()
X[:,0] = labelencoder_X.fit_transform(X[:,0])
X[:,1] = labelencoder_X.fit_transform(X[:,1])
onehotencoder = OneHotEncoder(categorical_features=[0,1])
X = onehotencoder.fit_transform(X).toarray()
print(X)
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/preprocessing/_encoders.py:415: FutureWarning: The
handling of integer data will change in version 0.22. Currently, the categories are determined
based on the range [0, max(values)], while in the future they will be determined based on the
unique values.
If you want the future behaviour and silence this warning, you can specify "categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories to integers, t
hen you can now use the OneHotEncoder directly.
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/preprocessing/_encoders.py:451: DeprecationWarning:
The 'categorical_features' keyword is deprecated in version 0.20 and will be removed in 0.22. You
can use the ColumnTransformer instead.
  "use the ColumnTransformer instead.", DeprecationWarning)
```

```
[[1.   0.   0.   ... 0.   0.05 0.  ]
 [1.   0.   0.   ... 0.   0.   0.  ]
 [1.   0.   0.   ... 1.   0.   0.  ]
 ...
 [1.   0.   0.   ... 0.   0.01 0.  ]
 [1.   0.   0.   ... 1.   0.   0.  ]
 [1.   0.   0.   ... 0.   0.   0.  ]]
```

In [7]:

```
labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)
print(y)
```

```
[1 1 0 ... 1 0 1]
```

In [8]:

```
X.shape
```

Out[8]:

```
(125973, 3023)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA, IncrementalPCA
X_st = StandardScaler().fit_transform(X)
```

In [10]:

```
pca = PCA(n_components=0.9, whiten=True)
X_pca = pca.fit_transform(X_st)
print('Original number of features:', X_st.shape[1])
print('Reduced number of features:', X_pca.shape[1])
```

```
Original number of features: 3023
Reduced number of features: 2693
```

In [0]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_pca, y, test_size=0.2, random_state=0)
```

In [12]:

```
from sklearn.naive_bayes import BernoulliNB
classifier = BernoulliNB(binarize=0.0)
classifier.fit(X_train, y_train)
```

Out[12]:

```
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
```

In [13]:

```
y_pred = classifier.predict(X_test)
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[8238  322  368   26  212]
 [  11 9337 1234  936 1939]
 [ 408  323 1251   90  270]
 [   0   14   11  151   39]
 [   0    1    0    5    9]]
              precision    recall  f1-score   support

           0       0.95      0.90      0.92      9166
           1       0.93      0.69      0.80     13457
           2       0.44      0.53      0.48      2342
           3       0.12      0.70      0.21       215
           4       0.00      0.60      0.01        15

    accuracy                           0.75     25195
   macro avg       0.49      0.69      0.48     25195
weighted avg       0.89      0.75      0.81     25195
```

In [14]:

```
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.7535622147251438
```

In [15]:

```
from imblearn.metrics import sensitivity_score, specificity_score, geometric_mean_score
sensitivity_score(y_test, y_pred, average='macro')
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/externals/six.py:31: DeprecationWarning: The module is deprecated in version 0.21 and will be removed in version 0.23 since we've dropped support for Python 2.7. Please rely on the official version of six (https://pypi.org/project/six/).
  "(https://pypi.org/project/six/).", DeprecationWarning)
```

Out[15]:

```
0.685816066108359
```

In [16]:

```
specificity_score(y_test, y_pred, average='macro')
```

Out[16]:

```
0.9414080527315034
```

In [17]:

```
geometric_mean_score(y_test, y_pred, average='macro')
```

Out[17]:

```
0.8035127673702828
```

In [0]:

```

```