```
import numpy as np
import pandas as pd
```

```
df=pd.read_csv("/content/emails.csv")
```

```
df.head()
```

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff	dry	Prediction
0	Email 1	0	0	1	0	0	0	2	0	0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	Email 2	8	13	24	6	6	2	102	1	27	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
2	Email 3	0	0	1	0	0	0	8	0	0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	Email 4	0	5	22	0	5	1	51	2	10	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	Email 5	7	6	17	1	5	2	57	0	9	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0

5 rows × 3002 columns

```
df.tail()
```

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff	dry	Prediction
2250	Email 2251	0	0	1	0	1	0	6	0	0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2251	Email 2252	0	0	2	0	1	0	6	0	0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2252	Email 2253	0	0	1	0	0	0	2	0	0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

```
df.dtypes
```

```
Email No.      object
the             int64
to              int64
ect             int64
and             int64
                ...
military      float64
allowing      float64
ff            float64
dry           float64
Prediction    float64
Length: 3002, dtype: object
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2255 entries, 0 to 2254
Columns: 3002 entries, Email No. to Prediction
dtypes: float64(1747), int64(1254), object(1)
memory usage: 51.6+ MB
```

```
df.shape
```

```
(2255, 3002)
```

```
df.drop(columns=['Email No.'], inplace=True)
```

```
df.head
```

```
<bound method NDFrame.head of       the  to  ect  and  for  of    a  you  hou  in  ...  connevey  jay  \
```

```
0       0   0    1    0    0   0    2    0    0   0  ...       0.0  0.0
1       8  13   24    6    6   2  102    1   27  18  ...       0.0  0.0
2       0   0    1    0    0   0    8    0    0   4  ...       0.0  0.0
3       0   5   22    0    5   1   51    2   10   1  ...       0.0  0.0
4       7   6   17    1    5   2   57    0    9   3  ...       0.0  0.0
...   ...  ..  ...  ...  ...  ..  ...  ...  ...  ..  ...       ...  ...
2250    0   0    1    0    1   0    6    0    0   0  ...       0.0  0.0
2251    0   0    2    0    1   0    6    0    0   0  ...       0.0  0.0
2252    0   0    1    0    0   0    2    0    0   0  ...       0.0  0.0
2253    4   3    1    0    1  17  163    2    0  32  ...       0.0  0.0
2254    1   2    4    3    0   0   70    2    0  17  ...       NaN  NaN

      valued  lay  infrastructure  military  allowing    ff  dry  Prediction
0        0.0  0.0             0.0       0.0       0.0   0.0  0.0         0.0
1        0.0  0.0             0.0       0.0       0.0   1.0  0.0         0.0
2        0.0  0.0             0.0       0.0       0.0   0.0  0.0         0.0
3        0.0  0.0             0.0       0.0       0.0   0.0  0.0         0.0
4        0.0  0.0             0.0       0.0       0.0   1.0  0.0         0.0
...      ...  ...             ...       ...       ...   ...  ...         ...
2250     0.0  0.0             0.0       0.0       0.0   0.0  0.0         0.0
2251     0.0  0.0             0.0       0.0       0.0   0.0  0.0         0.0
2252     0.0  0.0             0.0       0.0       0.0   0.0  0.0         0.0
2253     0.0  0.0             0.0       0.0       0.0  12.0  0.0         1.0
2254     NaN  NaN             NaN       NaN       NaN   NaN  NaN         NaN

[2255 rows x 3001 columns]>
```

```
df.dropna( axis=0,inplace=True)
```

```
df.isnull().any().value_counts()
```

```
False    3001
dtype: int64
```

```
df.isnull()
```

	the	to	ect	and	for	of	a	you	hou	in	...	connev
0	False	False	False	False	False	False	False	False	False	False	...	Fal
1	False	False	False	False	False	False	False	False	False	False	...	Fal
2	False	False	False	False	False	False	False	False	False	False	...	Fal
3	False	False	False	False	False	False	False	False	False	False	...	Fal
4	False	False	False	False	False	False	False	False	False	False	...	Fal
...	...	...	...	...	...	...	...	...	...	...	...	
2249	False	False	False	False	False	False	False	False	False	False	...	Fal
2250	False	False	False	False	False	False	False	False	False	False	...	Fal
2251	False	False	False	False	False	False	False	False	False	False	...	Fal

```
X=df.iloc[:, :df.shape[1]-1]
y=df.iloc[:,-1]
X.shape, y.shape
```

```
((2254, 3000), (2254,))
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.15, random_state=8)
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC, LinearSVC
from sklearn.neural_network import MLPClassifier
```

```
models={"Logistic Regression": LogisticRegression(random_state=8, solver='lbfgs', max_iter=3000),
"Linear SVM":LinearSVC(random_state=8, max_iter=3000), "Polynomical SVM":SVC(kernel="poly",
degree=2, random_state=8), "RBF SVM":SVC(kernel="rbf", random_state=8),
"Sigmoid SVM":SVC(kernel="sigmoid", random_state=8),
"Multi-layer Perceptron Classification": MLPClassifier(hidden_layer_sizes=[20, 20],
learning_rate='adaptive', random_state=8)}
```

```
from sklearn.metrics import accuracy_score
```

```
from sklearn.model_selection import train_test_split

X = df.iloc[:, 1:-1].values
y = df.iloc[:, -1].values


X.shape
```

```
(2254, 2999)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=101)


from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier


mnb_model=MultinomialNB()
mnb_model.fit(X_train,y_train)
```

```
MultinomialNB()
```

```
lr_model = LogisticRegression(solver='liblinear')
lr_model.fit(X_train,y_train)
```

```
LogisticRegression(solver='liblinear')
```

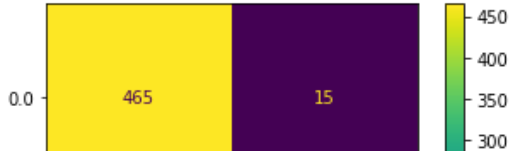
```
rfc_model=RandomForestClassifier()
rfc_model.fit(X_train,y_train)
```

```
RandomForestClassifier()
```

```
from sklearn.metrics import plot_confusion_matrix,classification_report,plot_precision_recall_curve,plot_roc_curve


def report(model):
  preds = model.predict(X_test)
  print(classification_report(preds,y_test))
  plot_confusion_matrix(model,X_test,y_test)
  plot_precision_recall_curve(model,X_test,y_test)
```

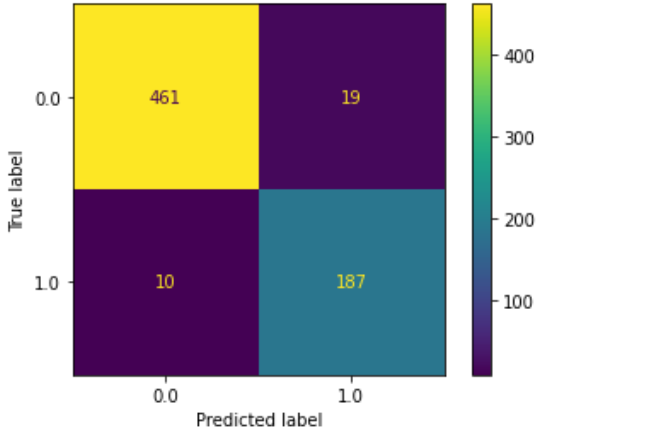
```
    plot_roc_curve(model,X_test,y_test)


print("LOGISTIC REGRESSION MODEL")
report(lr_model)
```

```
LOGISTIC REGRESSION MODEL
              precision    recall  f1-score   support

         0.0       0.97      0.96      0.96       484
         1.0       0.90      0.92      0.91       193

    accuracy                           0.95       677
   macro avg       0.94      0.94      0.94       677
weighted avg       0.95      0.95      0.95       677

/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is deprecated; Function `plot_con
  warnings.warn(msg, category=FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_precision_recall_curve is deprecated; Function `pl
  warnings.warn(msg, category=FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_roc_curve is deprecated; Function :func:`plot_roc_
  warnings.warn(msg, category=FutureWarning)
```



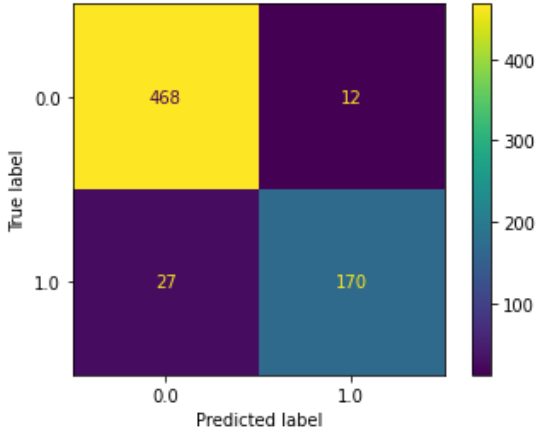

```
print("NAIVE BAYES MODEL")
report(mnb_model)
```

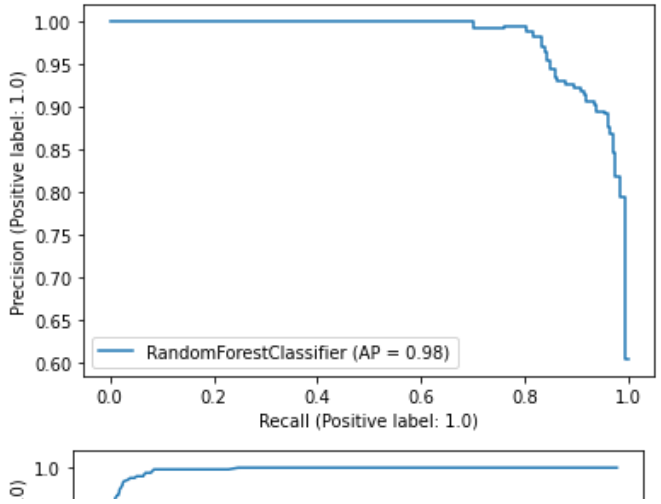
```
NAIVE BAYES MODEL
              precision    recall  f1-score   support

         0.0       0.96      0.98      0.97       471
         1.0       0.95      0.91      0.93       206

    accuracy                           0.96       677
   macro avg       0.95      0.94      0.95       677
weighted avg       0.96      0.96      0.96       677

/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is deprecated; Function `plot_con
  warnings.warn(msg, category=FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_precision_recall_curve is deprecated; Function `pl
  warnings.warn(msg, category=FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_roc_curve is deprecated; Function :func:`plot_roc_
  warnings.warn(msg, category=FutureWarning)
```

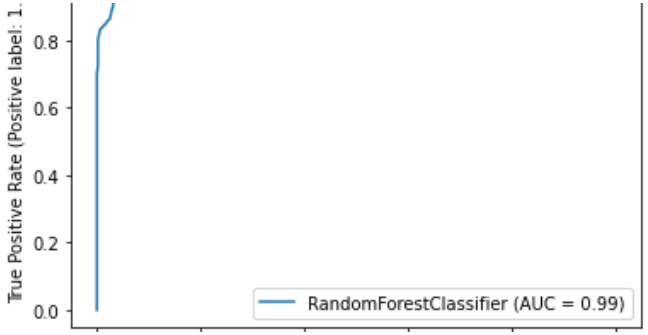



```
print("RANDOM FOREST MODEL")
report(rfc_model)
```

```
RANDOM FOREST MODEL
              precision    recall  f1-score   support

         0.0       0.97      0.95      0.96       495
         1.0       0.86      0.93      0.90       182

    accuracy                           0.94       677
   macro avg       0.92      0.94      0.93       677
weighted avg       0.94      0.94      0.94       677

/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is deprecated; Function `plot_con
  warnings.warn(msg, category=FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_precision_recall_curve is deprecated; Function `pl
  warnings.warn(msg, category=FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_roc_curve is deprecated; Function :func:`plot_roc_
  warnings.warn(msg, category=FutureWarning)
```







✓ 0s completed at 11:19 AM