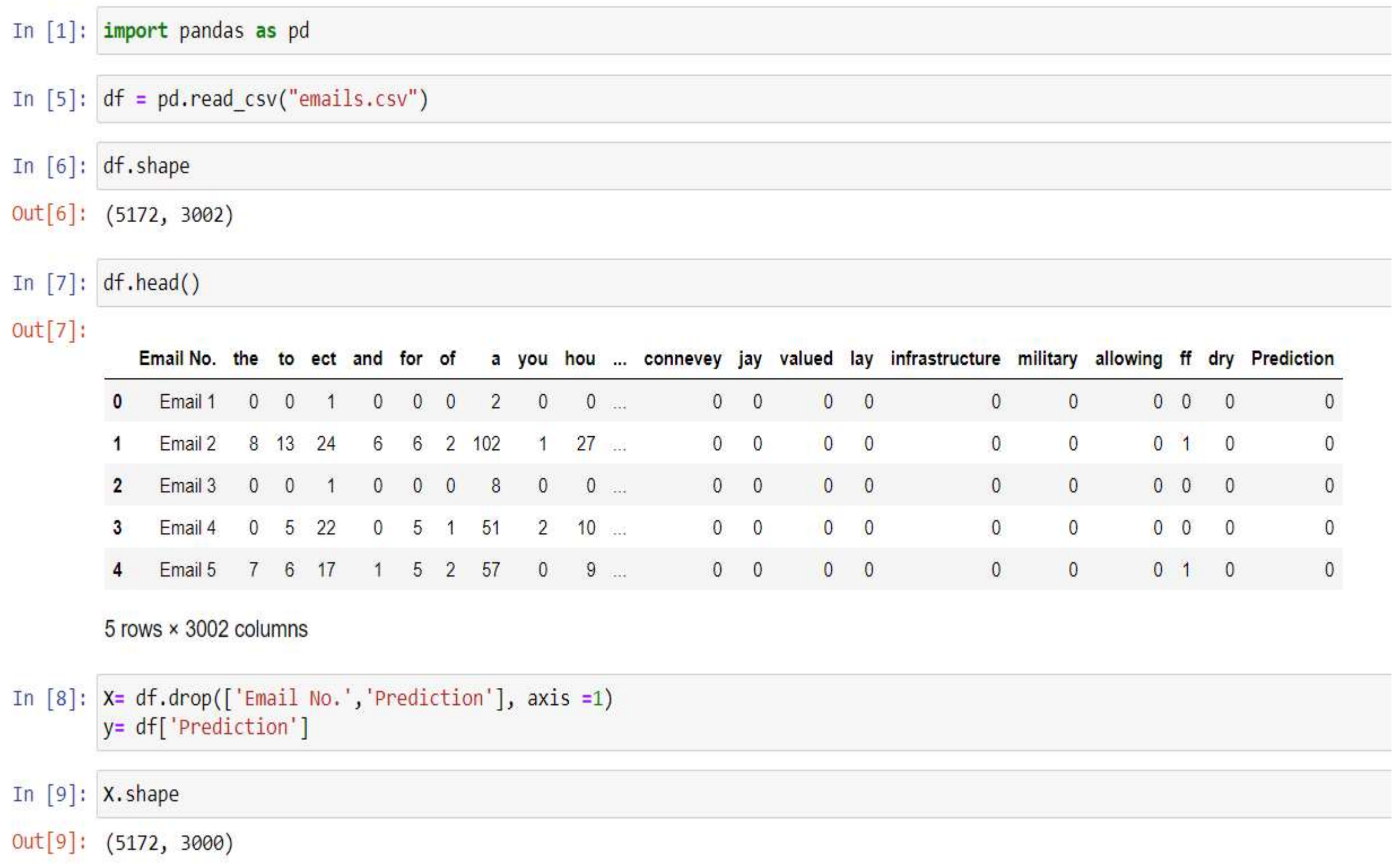
Assignment No. 2

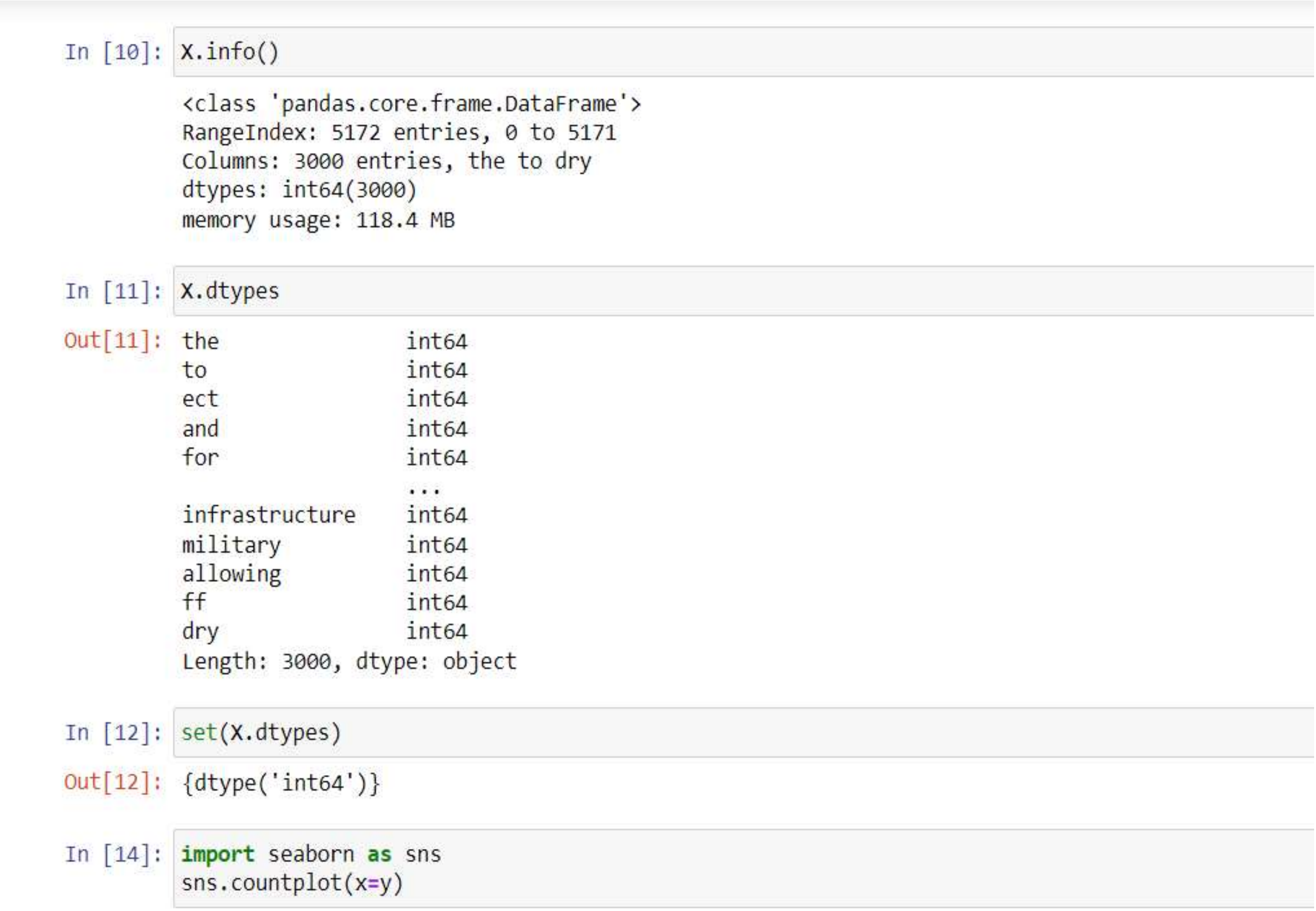


```
In [1]: import pandas as pd

In [5]: df = pd.read_csv("emails.csv")

In [6]: df.shape

Out[6]: (5172, 3002)

In [7]: df.head()

Out[7]:
```

	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
1	Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	0	0	0	1	0	0
2	Email 3	0	0	1	0	0	0	8	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
4	Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	0	0	0	1	0	0

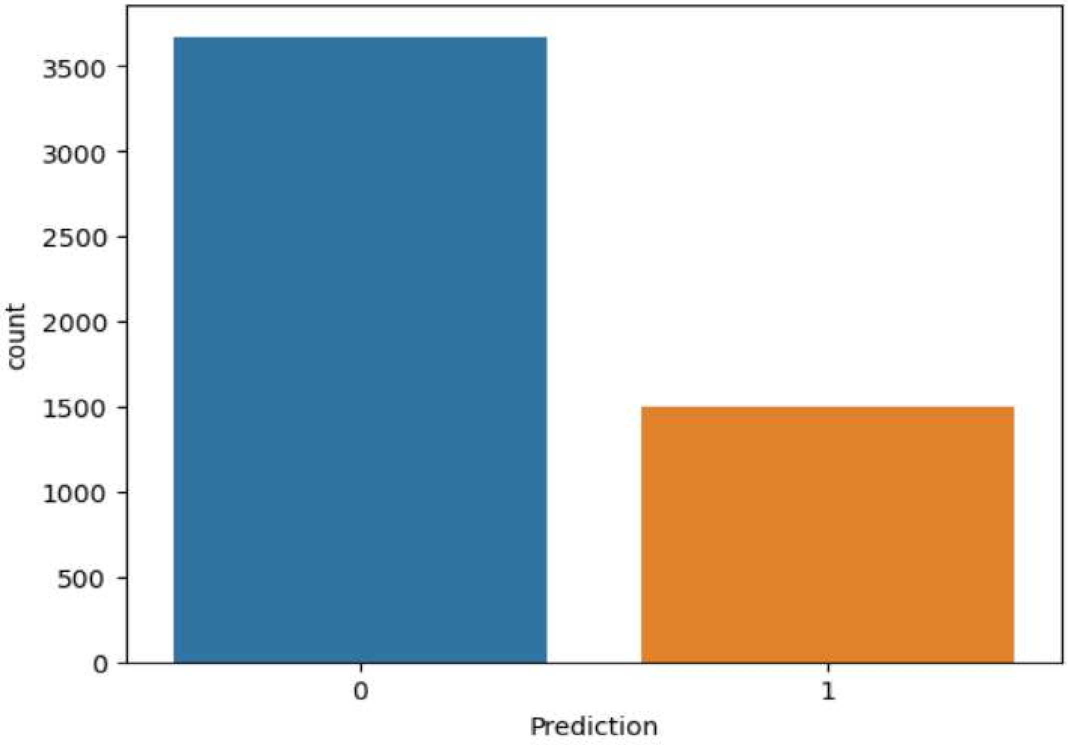
5 rows × 3002 columns

```
In [8]: X= df.drop(['Email No.','Prediction'], axis =1)
        y= df['Prediction']

In [9]: X.shape

Out[9]: (5172, 3000)
```



```
In [10]: X.info()

         <class 'pandas.core.frame.DataFrame'>
         RangeIndex: 5172 entries, 0 to 5171
         Columns: 3000 entries, the to dry
         dtypes: int64(3000)
         memory usage: 118.4 MB

In [11]: X.dtypes

Out[11]: the               int64
         to                int64
         ect               int64
         and               int64
         for               int64
                           ...
         infrastructure    int64
         military          int64
         allowing          int64
         ff                int64
         dry               int64
         Length: 3000, dtype: object

In [12]: set(X.dtypes)

Out[12]: {dtype('int64')}

In [14]: import seaborn as sns
         sns.countplot(x=y)
```

```
Out[14]: <AxesSubplot:xlabel='Prediction', ylabel='count'>
```



```
In [15]: y.value_counts()
```

```
Out[15]: 0    3672
         1    1500
         Name: Prediction, dtype: int64
```

```
In [17]: from sklearn.preprocessing import MinMaxScaler
         scaler = MinMaxScaler()
         x_scaled = scaler.fit_transform(X)
```

```
In [18]: x_scaled
```

```
Out[18]: array([[0.        , 0.        , 0.        , ..., 0.        , 0.        ,
                 0.        ],
                [0.03809524, 0.09848485, 0.06705539, ..., 0.        , 0.00877193,
                 0.        ],
                [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
                 0.        ],
                ...,
                [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
                 0.        ],
                [0.00952381, 0.0530303 , 0.        , ..., 0.        , 0.00877193,
                 0.        ],
                [0.1047619 , 0.18181818, 0.01166181, ..., 0.        , 0.        ,
                 0.        ]])
```

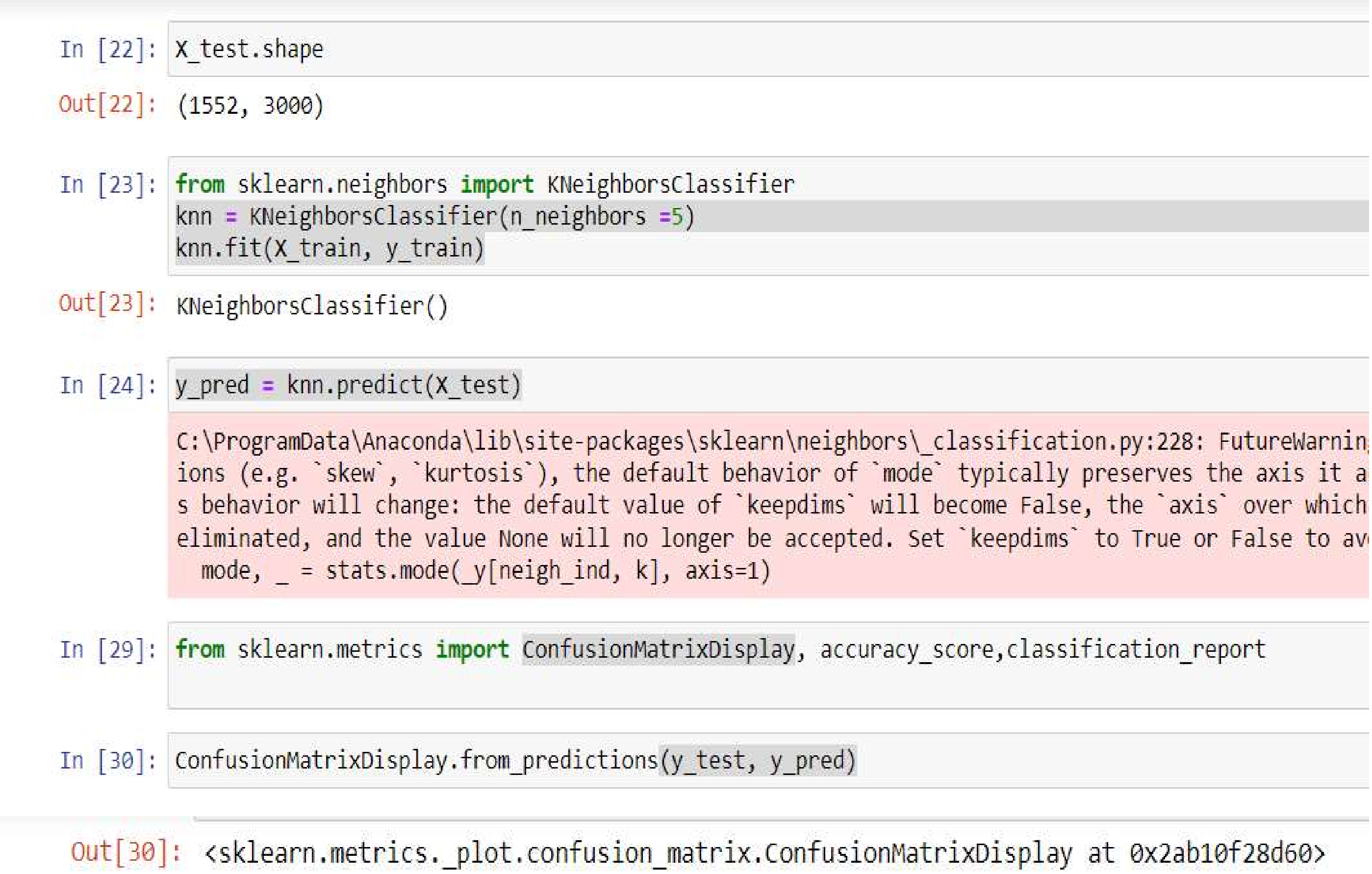
```
In [19]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(x_scaled, y, random_state=0, test_size = 0.30)
```

```
In [20]: x_scaled.shape
```

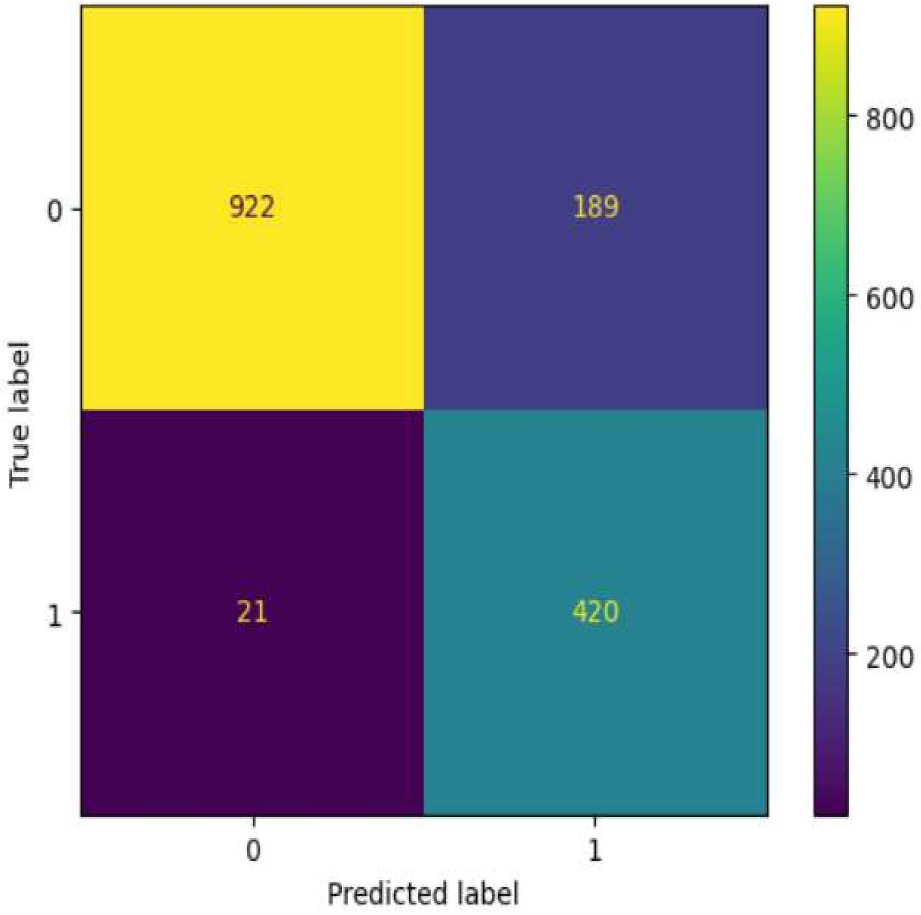
```
Out[20]: (5172, 3000)
```

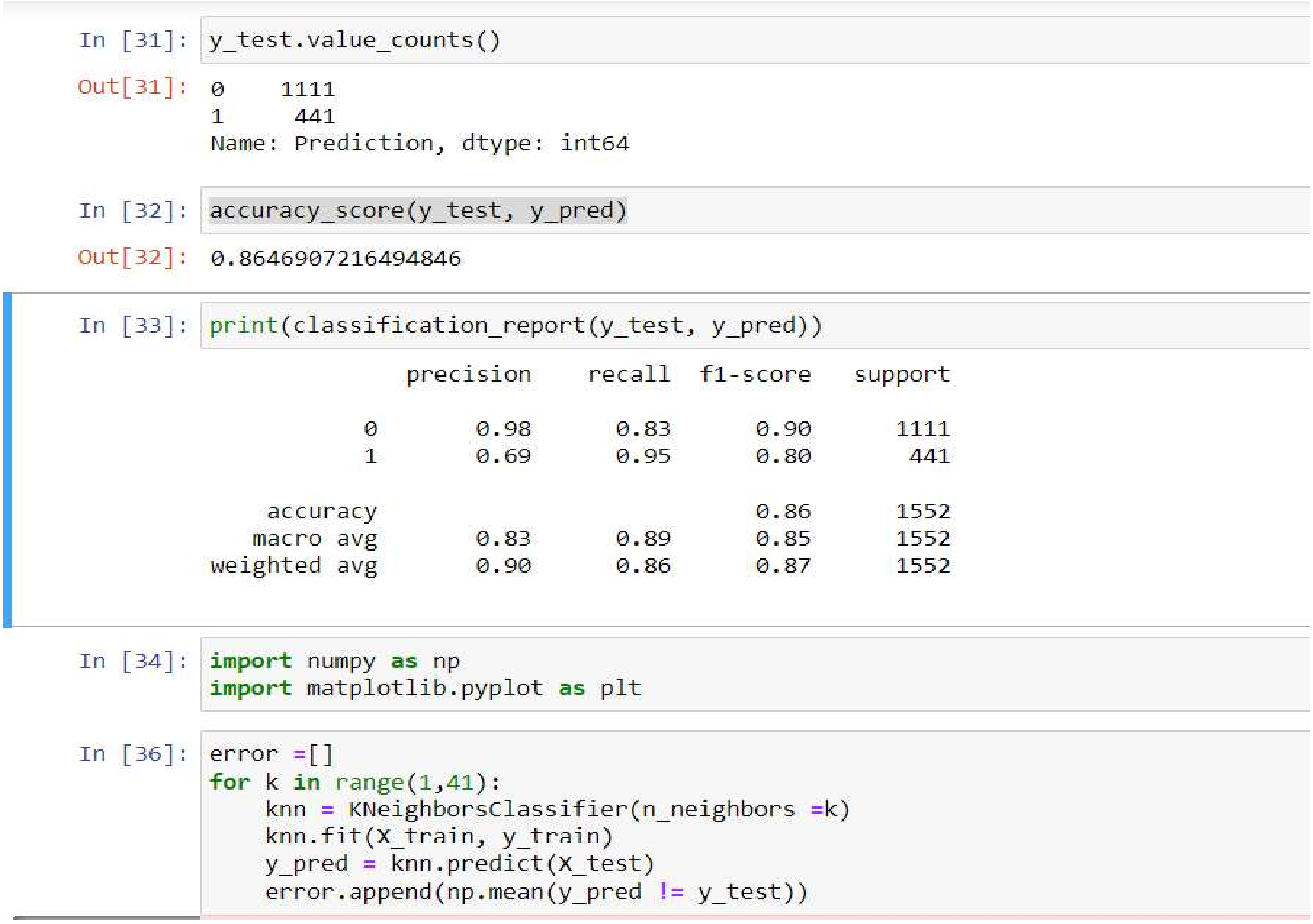
```
In [21]: X_train.shape
```

```
Out[21]: (3620, 3000)
```

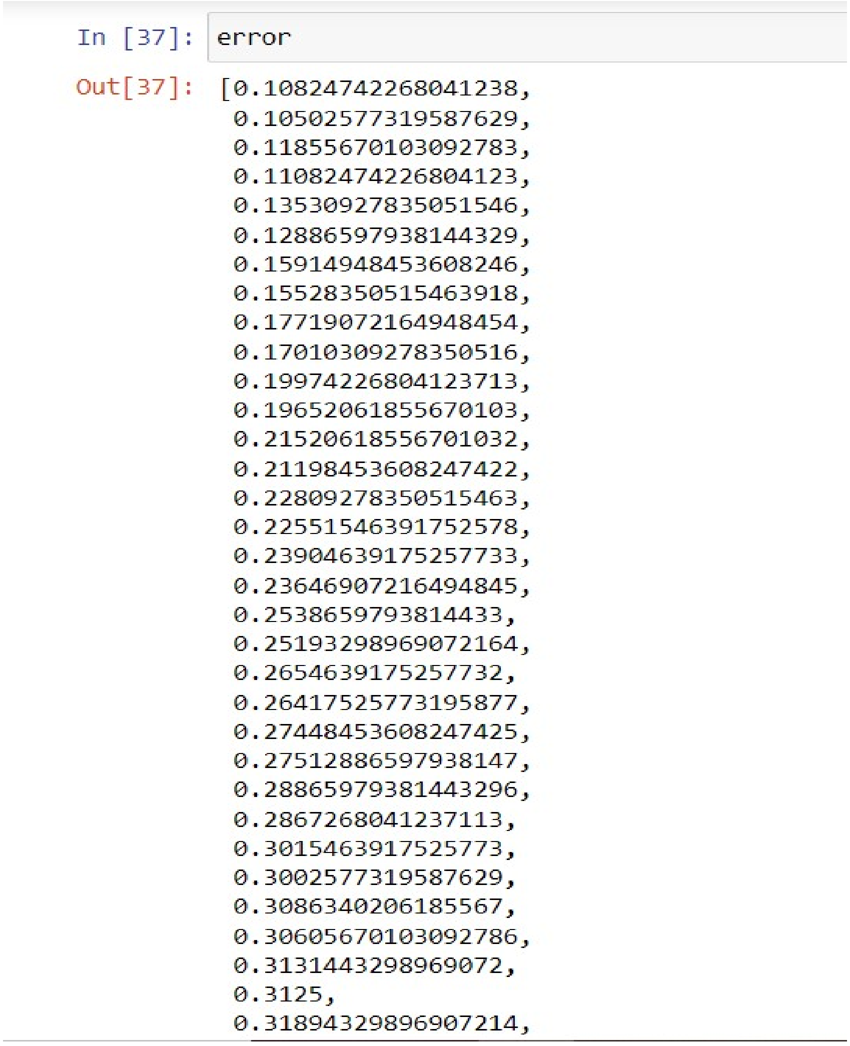


```
In [22]: X_test.shape

Out[22]: (1552, 3000)

In [23]: from sklearn.neighbors import KNeighborsClassifier
         knn = KNeighborsClassifier(n_neighbors =5)
         knn.fit(X_train, y_train)

Out[23]: KNeighborsClassifier()

In [24]: y_pred = knn.predict(X_test)

         C:\ProgramData\Anaconda\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarnin
         ions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it a
         s behavior will change: the default value of `keepdims` will become False, the `axis` over which
         eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to av
           mode, _ = stats.mode(_y[neigh_ind, k], axis=1)

In [29]: from sklearn.metrics import ConfusionMatrixDisplay, accuracy_score,classification_report

In [30]: ConfusionMatrixDisplay.from_predictions(y_test, y_pred)

Out[30]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x2ab10f28d60>
```

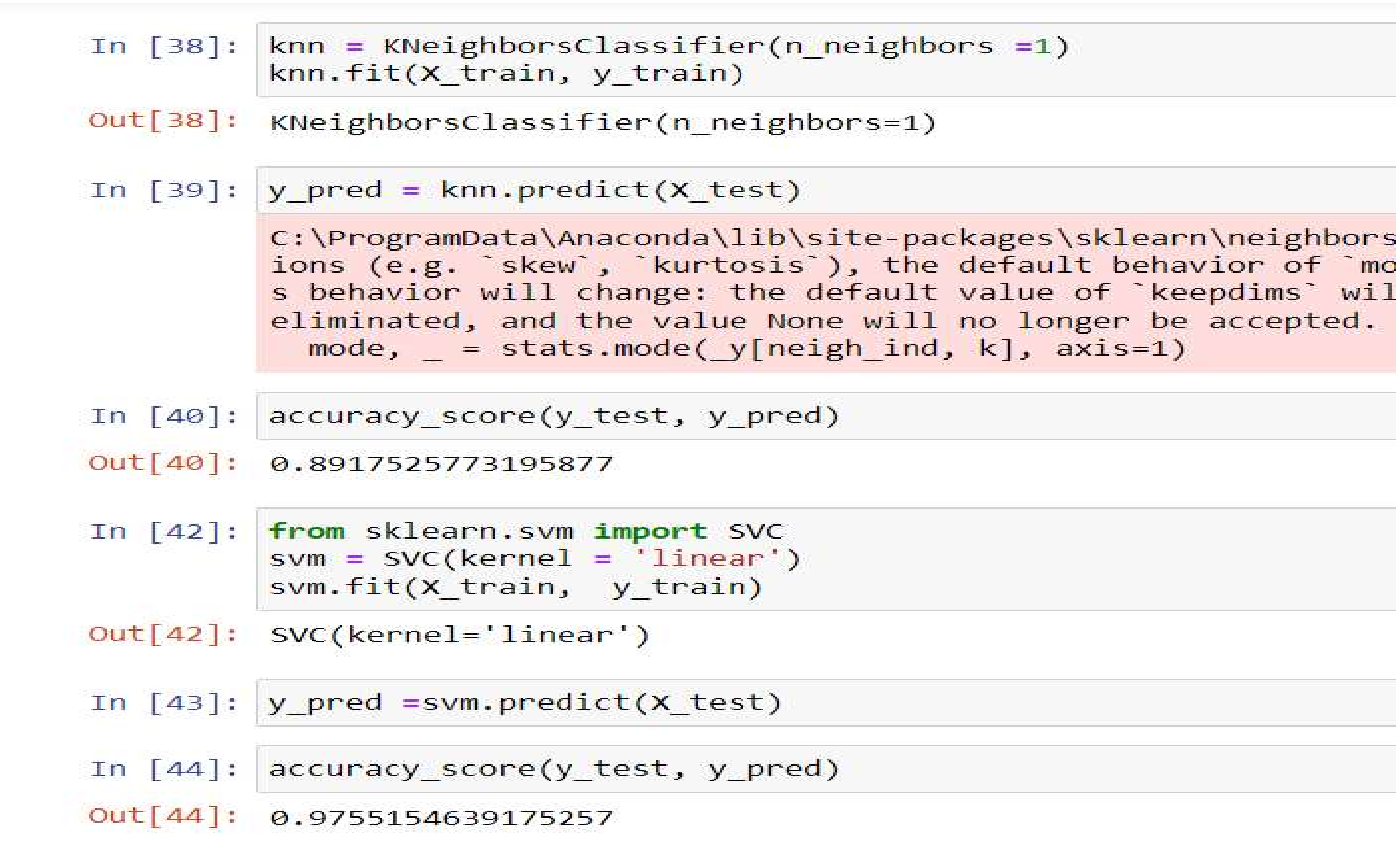




```
In [31]: y_test.value_counts()

Out[31]: 0    1111
         1     441
         Name: Prediction, dtype: int64

In [32]: accuracy_score(y_test, y_pred)

Out[32]: 0.8646907216494846

In [33]: print(classification_report(y_test, y_pred))

                       precision    recall  f1-score   support

                    0       0.98      0.83      0.90      1111
                    1       0.69      0.95      0.80       441

             accuracy                           0.86      1552
            macro avg       0.83      0.89      0.85      1552
         weighted avg       0.90      0.86      0.87      1552


In [34]: import numpy as np
         import matplotlib.pyplot as plt

In [36]: error =[]
         for k in range(1,41):
             knn = KNeighborsClassifier(n_neighbors =k)
             knn.fit(X_train, y_train)
             y_pred = knn.predict(X_test)
             error.append(np.mean(y_pred != y_test))
```



```
In [37]: error

Out[37]: [0.10824742268041238,
          0.10502577319587629,
          0.11855670103092783,
          0.11082474226804123,
          0.13530927835051546,
          0.12886597938144329,
          0.15914948453608246,
          0.15528350515463918,
          0.17719072164948454,
          0.17010309278350516,
          0.19974226804123713,
          0.19652061855670103,
          0.21520618556701032,
          0.21198453608247422,
          0.22809278350515463,
          0.22551546391752578,
          0.23904639175257733,
          0.23646907216494845,
          0.2538659793814433,
          0.25193298969072164,
          0.2654639175257732,
          0.26417525773195877,
          0.27448453608247425,
          0.27512886597938147,
          0.28865979381443296,
          0.2867268041237113,
          0.3015463917525773,
          0.3002577319587629,
          0.3086340206185567,
          0.30605670103092786,
          0.3131443298969072,
          0.3125,
          0.31894329896907214,
```



```
In [38]: knn = KNeighborsClassifier(n_neighbors =1)
         knn.fit(X_train, y_train)

Out[38]: KNeighborsClassifier(n_neighbors=1)

In [39]: y_pred = knn.predict(X_test)

         C:\ProgramData\Anaconda\lib\site-packages\sklearn\neighbors
         ions (e.g. `skew`, `kurtosis`), the default behavior of `mo
         s behavior will change: the default value of `keepdims` wil
         eliminated, and the value None will no longer be accepted.
           mode, _ = stats.mode(_y[neigh_ind, k], axis=1)

In [40]: accuracy_score(y_test, y_pred)

Out[40]: 0.8917525773195877

In [42]: from sklearn.svm import SVC
         svm = SVC(kernel = 'linear')
         svm.fit(X_train,  y_train)

Out[42]: SVC(kernel='linear')

In [43]: y_pred =svm.predict(X_test)

In [44]: accuracy_score(y_test, y_pred)

Out[44]: 0.9755154639175257
```