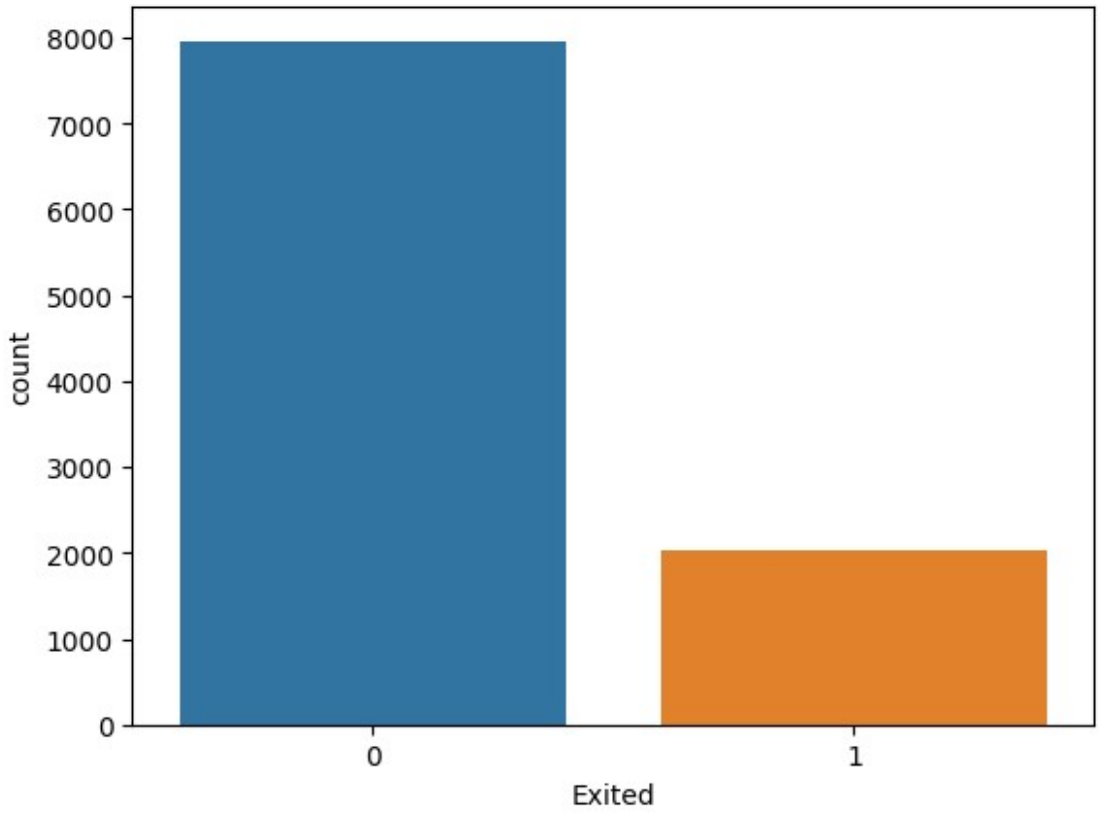
```
import pandas as pd
import seaborn as sns

df = pd.read_csv("Dataset/Churn_Modelling.csv")

df.shape

(10000, 14)

df.columns

Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore',
'Geography',
       'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts',
'HasCrCard',
       'IsActiveMember', 'EstimatedSalary', 'Exited'],
      dtype='object')

df.head()

   RowNumber  CustomerId   Surname  CreditScore Geography  Gender  Age
\
0          1    15634602  Hargrave          619    France  Female   42

1          2    15647311      Hill          608     Spain  Female   41

2          3    15619304      Onio          502    France  Female   42

3          4    15701354      Boni          699    France  Female   39

4          5    15737888  Mitchell          850     Spain  Female   43


   Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0       2       0.00              1          1               1
1       1   83807.86              1          0               1
2       8  159660.80              3          1               0
3       1       0.00              2          0               0
4       2  125510.82              1          1               1

   EstimatedSalary  Exited
0        101348.88       1
1        112542.58       0
2        113931.57       1
3         93826.63       0
4         79084.10       0

# Input Data
x =
df[['CreditScore','Age','Tenure','Balance','NumOfProducts','HasCrCard'
,'IsActiveMember','EstimatedSalary']]
```

```
# Output Data
y = df['Exited']

x
```

```
      CreditScore  Age  Tenure    Balance  NumOfProducts  HasCrCard  \
0             619   42       2       0.00              1          1
1             608   41       1   83807.86              1          0
2             502   42       8  159660.80              3          1
3             699   39       1       0.00              2          0
4             850   43       2  125510.82              1          1
...           ...  ...     ...        ...            ...        ...
9995          771   39       5       0.00              2          1
9996          516   35      10   57369.61              1          1
9997          709   36       7       0.00              1          0
9998          772   42       3   75075.31              2          1
9999          792   28       4  130142.79              1          1

      IsActiveMember  EstimatedSalary
0                  1        101348.88
1                  1        112542.58
2                  0        113931.57
3                  0         93826.63
4                  1         79084.10
...              ...              ...
9995               0         96270.64
9996               1        101699.77
9997               1         42085.58
9998               0         92888.52
9999               0         38190.78

[10000 rows x 8 columns]
```

```
sns.countplot(x = y);
```




```
y.value_counts()
```

```
0    7963
1    2037
Name: Exited, dtype: int64
```

```
from imblearn.over_sampling import RandomOverSampler

ros = RandomOverSampler(random_state=0)

x_yes, y_res = ros.fit_resample(x, y)

y_res.value_counts()
```

```
1    7963
0    7963
Name: Exited, dtype: int64
```

```
# Normalize
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

x_scaled = scaler.fit_transform(x)

x_scaled
```

```
array([[-0.32622142,  0.29351742, -1.04175968, ...,  0.64609167,
         0.97024255,  0.02188649],
       [-0.44003595,  0.19816383, -1.38753759, ..., -1.54776799,
         0.97024255,  0.21653375],
       [-1.53679418,  0.29351742,  1.03290776, ...,  0.64609167,
        -1.03067011,  0.2406869 ],
       ...,
       [ 0.60498839, -0.27860412,  0.68712986, ..., -1.54776799,
         0.97024255, -1.00864308],
       [ 1.25683526,  0.29351742, -0.69598177, ...,  0.64609167,
        -1.03067011, -0.12523071],
       [ 1.46377078, -1.04143285, -0.35020386, ...,  0.64609167,
        -1.03067011, -1.07636976]])
```

```
# Cross validation
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
random_state=0, test_size=0.25)
```

```
x.shape
```

```
(10000, 8)
```

```
x_test.shape
```

```
(2500, 8)
```

```
x_train.shape
```

```
(7500, 8)
```

```
from sklearn.neural_network import MLPClassifier
```

```
ann = MLPClassifier(hidden_layer_sizes=(100, 100, 100),
random_state=0, max_iter=100, activation='relu')
```

```
ann.fit(x_train, y_train)
```

```
C:\Users\ashre\anaconda3\envs\pandas\lib\site-packages\sklearn\
neural_network\_multilayer_perceptron.py:686: ConvergenceWarning:
Stochastic Optimizer: Maximum iterations (100) reached and the
optimization hasn't converged yet.
  warnings.warn(
```

```
MLPClassifier(hidden_layer_sizes=(100, 100, 100), max_iter=100,
random_state=0)
```

```
y_pred = ann.predict(x_test)
```

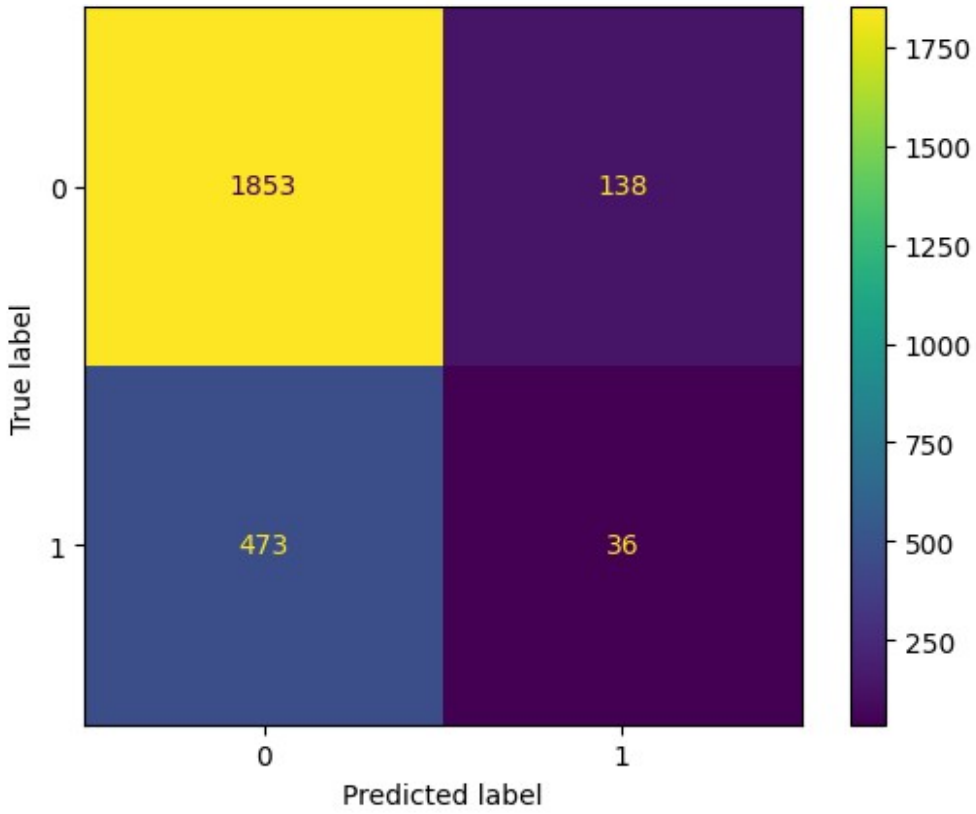
```
from sklearn.metrics import ConfusionMatrixDisplay,
classification_report
from sklearn.metrics import accuracy_score
```

```
y_test.value_counts()
```

```
0    1991
1     509
Name: Exited, dtype: int64
```

```
ConfusionMatrixDisplay.from_predictions(y_test, y_pred)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x1e13fcbde70>
```




```
accuracy_score(y_test,y_pred)
```

```
0.7556
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.80      0.93      0.86      1991
           1       0.21      0.07      0.11       509

    accuracy                           0.76      2500
   macro avg       0.50      0.50      0.48      2500
```

```
weighted avg       0.68      0.76      0.71      2500
```