ML_Prac3

October 18, 2023

0.0.1 Aim: Given a bank customer, build a neural network-based classifier that can determine whether they will leave or not in the next 6 months. Dataset Description: The case study is from an open-source dataset from Kaggle. The dataset contains 10,000 sample points with 14 distinct features such as CustomerId, CreditScore, Geography, Gender, Age, Tenure, Balance, etc.

Name: Chinmay Gokhale

Div: BE-A

Roll No. B211047

0.0.2 Import the Libraries

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[ ]:
```
```

[2]:
```
df=pd.read_csv("Churn_Modelling.csv")
```

[3]:
```
df
```

[3]:
```
      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
…           …         …         …         …         …      …  …
9995       9996    15606229   Obijiaku          771    France    Male   39
9996       9997    15569892  Johnstone          516    France    Male   35
9997       9998    15584532        Liu          709    France  Female   36
9998       9999    15682355  Sabbatini          772   Germany    Male   42
9999      10000    15628319     Walker          792    France  Female   28
```

```
      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
…        …        …             …         …            …
9995       5       0.00              2          1               0
9996      10   57369.61              1          1               1
9997       7       0.00              1          0               1
9998       3   75075.31              2          1               0
9999       4  130142.79              1          1               0

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

[10000 rows x 14 columns]
```

[4]:
```
df.head()
```

[4]:
```
   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
```

[5]:
```
df.tail()
```

[5]:
```
      RowNumber  CustomerId    Surname  CreditScore Geography  Gender  Age  \
9995       9996    15606229   Obijiaku          771    France    Male   39
9996       9997    15569892  Johnstone          516    France    Male   35
9997       9998    15584532        Liu          709    France  Female   36
9998       9999    15682355  Sabbatini          772   Germany    Male   42
9999      10000    15628319     Walker          792    France  Female   28

      Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
9995       5       0.00              2          1               0
9996      10   57369.61              1          1               1
9997       7       0.00              1          0               1
9998       3   75075.31              2          1               0
9999       4  130142.79              1          1               0

      EstimatedSalary  Exited
9995         96270.64       0
9996        101699.77       0
9997         42085.58       1
9998         92888.52       1
9999         38190.78       0
```

[6]:
```
df.shape
```

[6]:
```
(10000, 14)
```

[7]:
```
df.describe()
```

[7]:
```
         RowNumber    CustomerId   CreditScore           Age        Tenure  \
count  10000.00000  1.000000e+04  10000.000000  10000.000000  10000.000000
mean    5000.50000  1.569094e+07    650.528800     38.921800      5.012800
std     2886.89568  7.193619e+04     96.653299     10.487806      2.892174
min        1.00000  1.556570e+07    350.000000     18.000000      0.000000
25%     2500.75000  1.562853e+07    584.000000     32.000000      3.000000
50%     5000.50000  1.569074e+07    652.000000     37.000000      5.000000
75%     7500.25000  1.575323e+07    718.000000     44.000000      7.000000
max    10000.00000  1.581569e+07    850.000000     92.000000     10.000000

             Balance  NumOfProducts    HasCrCard  IsActiveMember  \
count   10000.000000   10000.000000  10000.00000    10000.000000
mean    76485.889288       1.530200      0.70550        0.515100
std     62397.405202       0.581654      0.45584        0.499797
```

```
min          0.000000        1.000000       0.00000         0.000000
25%          0.000000        1.000000       0.00000         0.000000
50%      97198.540000        1.000000       1.00000         1.000000
75%     127644.240000        2.000000       1.00000         1.000000
max     250898.090000        4.000000       1.00000         1.000000

       EstimatedSalary        Exited
count     10000.000000  10000.000000
mean     100090.239881      0.203700
std       57510.492818      0.402769
min          11.580000      0.000000
25%       51002.110000      0.000000
50%      100193.915000      0.000000
75%      149388.247500      0.000000
max      199992.480000      1.000000
```

[8]:
```
df.isnull()
```

[8]:
```
      RowNumber  CustomerId  Surname  CreditScore  Geography  Gender    Age  \
0         False       False    False        False      False   False  False
1         False       False    False        False      False   False  False
2         False       False    False        False      False   False  False
3         False       False    False        False      False   False  False
4         False       False    False        False      False   False  False
…           …          …       …           …          …       …      …
9995      False       False    False        False      False   False  False
9996      False       False    False        False      False   False  False
9997      False       False    False        False      False   False  False
9998      False       False    False        False      False   False  False
9999      False       False    False        False      False   False  False

      Tenure  Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0      False    False          False      False           False
1      False    False          False      False           False
2      False    False          False      False           False
3      False    False          False      False           False
4      False    False          False      False           False
…        …       …             …          …              …
9995   False    False          False      False           False
9996   False    False          False      False           False
9997   False    False          False      False           False
9998   False    False          False      False           False
9999   False    False          False      False           False

      EstimatedSalary  Exited
0               False   False
1               False   False
```

```
2               False   False
3               False   False
4               False   False
…               …     …
9995            False   False
9996            False   False
9997            False   False
9998            False   False
9999            False   False

[10000 rows x 14 columns]
```

[9]:
```
df.isnull().sum()
```

[9]:
```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

[10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   RowNumber        10000 non-null  int64
 1   CustomerId       10000 non-null  int64
 2   Surname          10000 non-null  object
 3   CreditScore      10000 non-null  int64
 4   Geography        10000 non-null  object
 5   Gender           10000 non-null  object
 6   Age              10000 non-null  int64
 7   Tenure           10000 non-null  int64
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64
```

```
 10  HasCrCard        10000 non-null  int64
 11  IsActiveMember   10000 non-null  int64
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

[11]:
```
df.info
```

[11]:
```
<bound method DataFrame.info of       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
…           …         …          …          …        …     …   …
9995       9996    15606229   Obijiaku          771    France    Male   39
9996       9997    15569892  Johnstone          516    France    Male   35
9997       9998    15584532        Liu          709    France  Female   36
9998       9999    15682355  Sabbatini          772   Germany    Male   42
9999      10000    15628319     Walker          792    France  Female   28

      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
…        …         …            …         …              …
9995       5       0.00              2          1               0
9996      10   57369.61              1          1               1
9997       7       0.00              1          0               1
9998       3   75075.31              2          1               0
9999       4  130142.79              1          1               0

      EstimatedSalary  Exited
0           101348.88       1
1           112542.58       0
2           113931.57       1
3            93826.63       0
4            79084.10       0
…                 …     …
9995         96270.64       0
9996        101699.77       0
9997         42085.58       1
9998         92888.52       1
```

```
9999         38190.78       0

[10000 rows x 14 columns]>
```

[12]:
```
df.columns
```

[12]:
```
Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
       'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
       'IsActiveMember', 'EstimatedSalary', 'Exited'],
      dtype='object')
```

[13]:
```
df.dtypes
```

[13]:
```
RowNumber            int64
CustomerId           int64
Surname             object
CreditScore          int64
Geography           object
Gender              object
Age                  int64
Tenure               int64
Balance            float64
NumOfProducts        int64
HasCrCard            int64
IsActiveMember       int64
EstimatedSalary    float64
Exited               int64
dtype: object
```

0.0.3 Spliting the data

[14]:
```
x = df.
↪drop(["RowNumber","CustomerId","Surname","Geography","Gender","Exited"],axis=1)
```

[15]:
```
x
```

[15]:
```
      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]
```

[16]:
```
y=df["Exited"]
```

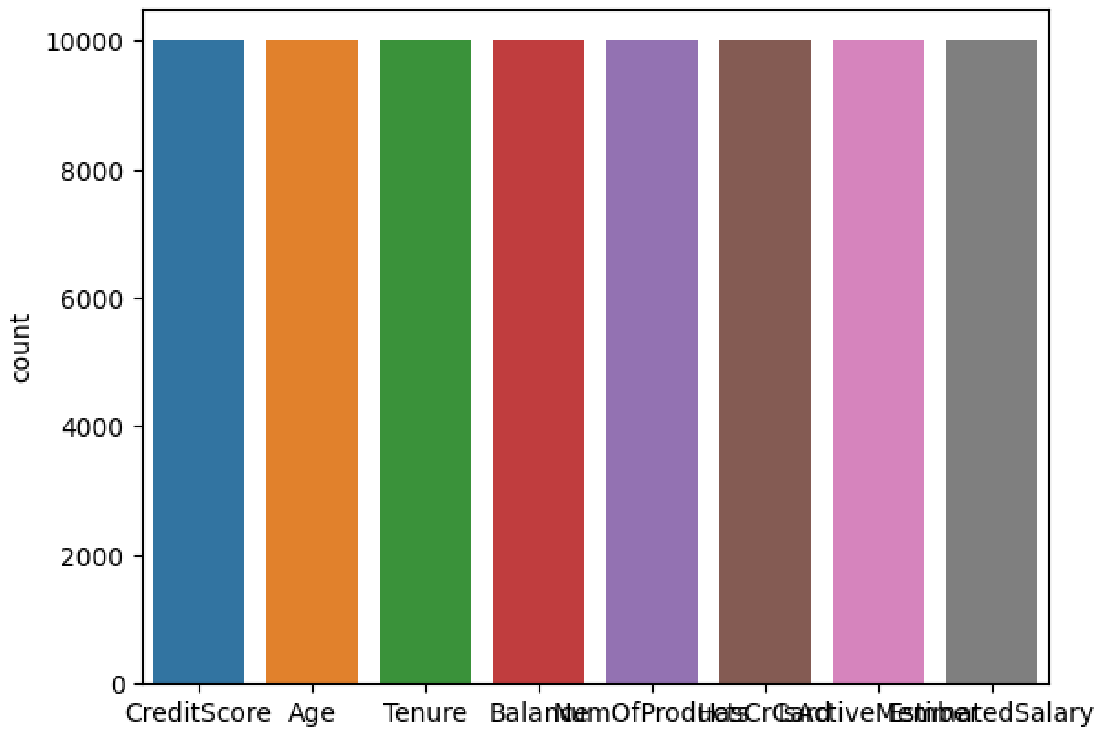
[17]:
```
y
```

[17]:
```
0       1
1       0
2       1
3       0
4       0
       ..
9995    0
9996    0
9997    1
9998    1
9999    0
Name: Exited, Length: 10000, dtype: int64
```

1 Checking Balancing of data

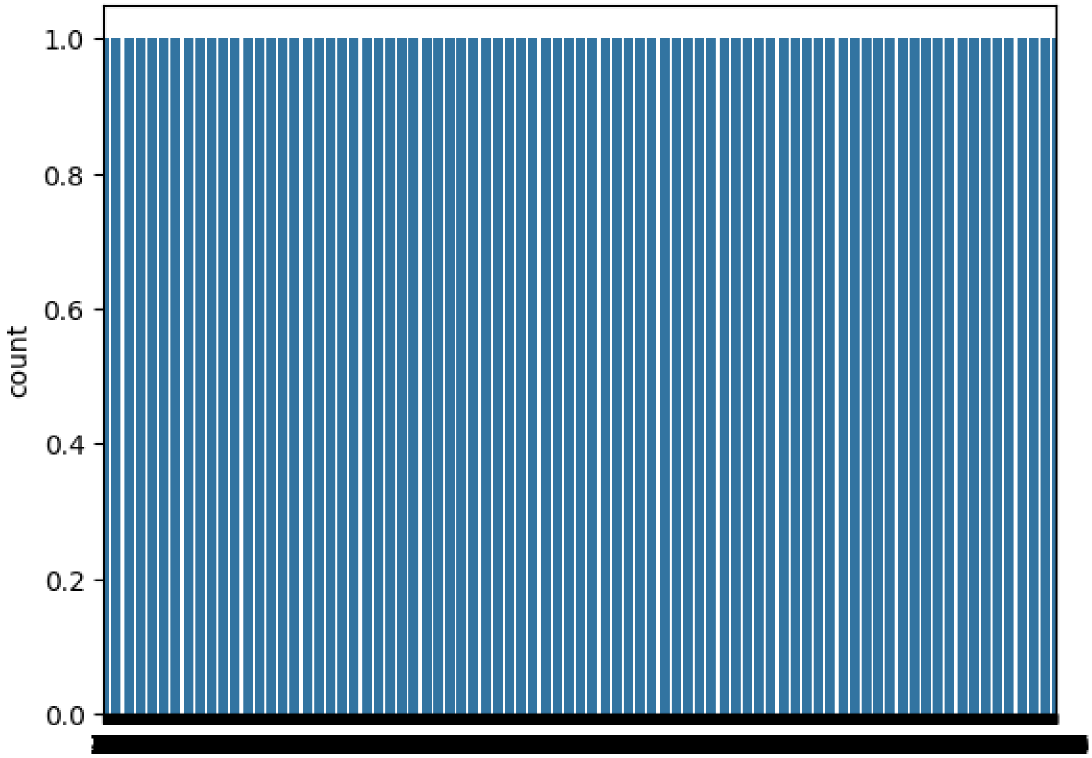
[18]:
```
sns.countplot(x)
```

[18]:
```
<Axes: ylabel='count'>
```



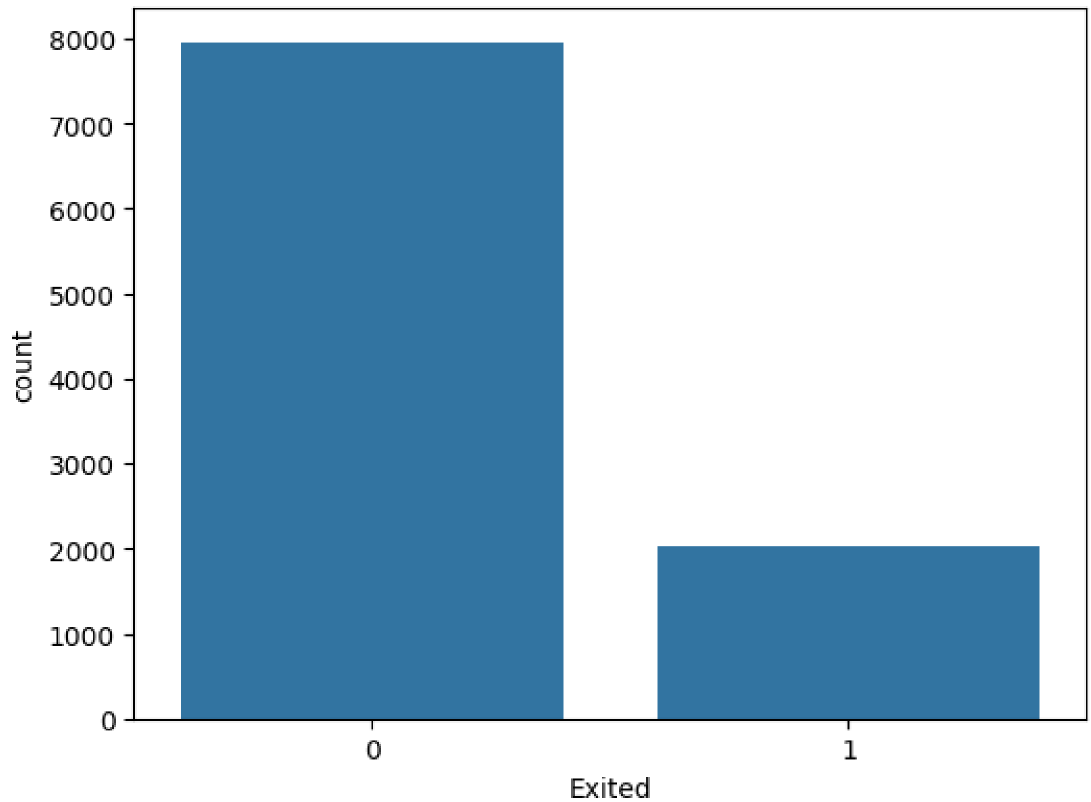
[19]:
```
sns.countplot(y)
```

[19]: <Axes: ylabel='count'>



[20]:
```
sns.countplot(x=y)
```

[20]: <Axes: xlabel='Exited', ylabel='count'>



[21]:
```
y.value_counts()
```

[21]:
```
Exited
0    7963
1    2037
Name: count, dtype: int64
```

[22]:
```
x.value_counts()
```

[22]:
```
CreditScore  Age  Tenure  Balance    NumOfProducts  HasCrCard  IsActiveMember
EstimatedSalary
350          39   0       109733.20  2              0          0
123602.11          1
695          34   9       0.00       2              1          1
67502.12           1
             28   5       171069.39  2              1          1
88689.40           1
             29   5       0.00       2              1          1
6770.44            1
                  9       0.00       2              1          0
111565.45          1
```

```
                                   ..
608             33    9       89968.69    1              1          0
68777.26                1
                34    3       106288.54   1              1          1
36639.25                1
                      4       88772.87    1              1          1
168822.01               1
                      7       86656.13    1              0          1
59890.29                1
850             81    5       0.00        2              1          1
44827.47                1
Name: count, Length: 10000, dtype: int64
```

1.1 Feature scaling

[23]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

[24]:
```
X_scale= sc.fit_transform(x)
```

[25]:
```
X_scale
```

[25]:
```
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]])
```

2 Cross Validation

[26]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=␣
 ↪train_test_split(X_scale,y,random_state=47,test_size=0.47)
```

[27]:
```
X_train.shape
```

[27]:
```
(5300, 8)
```

[28]:
```
X_test.shape
```

[28]: (4700, 8)

2.1 Initialize and build the model.

[29]:
```
from sklearn.neural_network import MLPClassifier
ann=MLPClassifier(hidden_layer_sizes=(100,100,100),random_state=2,activation="relu")
```

[30]:
```
ann.fit(X_train, Y_train)
```

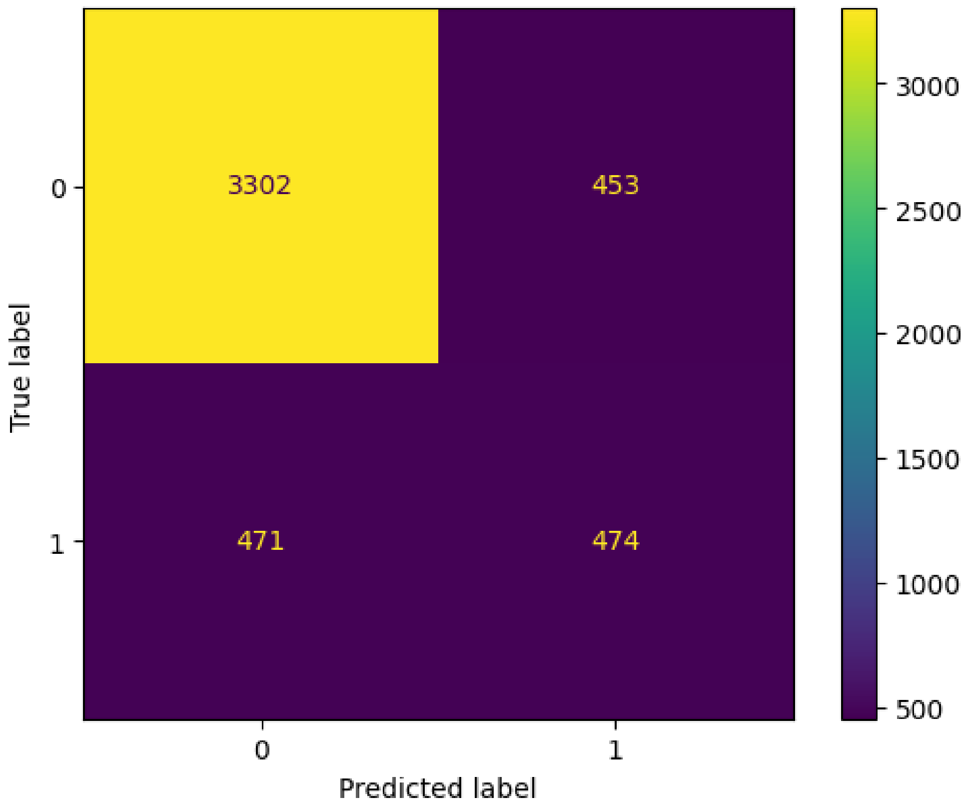
[30]: MLPClassifier(hidden_layer_sizes=(100, 100, 100), random_state=2)

[31]:
```
y_pred = ann.predict(X_test)
y_pred
```

[31]: array([0, 0, 1, …, 0, 0, 0])

[32]:
```
from sklearn.metrics import␣
↪ConfusionMatrixDisplay,accuracy_score,classification_report
ConfusionMatrixDisplay.from_predictions(Y_test,y_pred)
```

[32]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f10943391b0>



[33]:
```
accuracy_score(Y_test,y_pred)
```

[33]: 0.8034042553191489

[34]:
```
print(classification_report(Y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.88      0.88      0.88      3755
           1       0.51      0.50      0.51       945

    accuracy                           0.80      4700
   macro avg       0.69      0.69      0.69      4700
weighted avg       0.80      0.80      0.80      4700
```

[ ]: