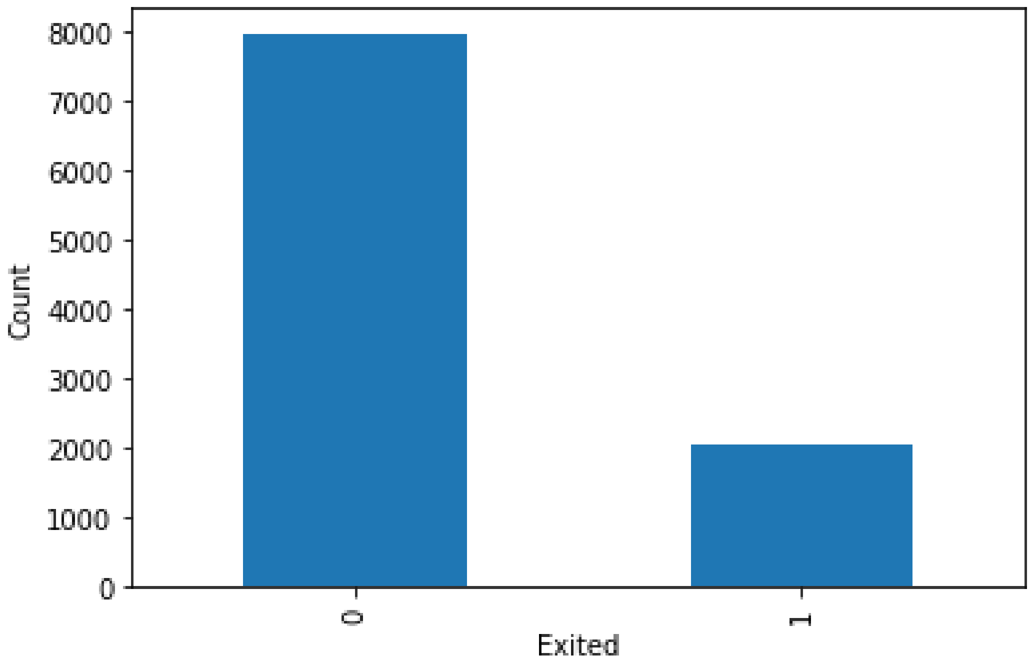
```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import tensorflow as tf

df = pd.read_csv('Churn_Modelling.csv')
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  
```

```python
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

plt.xlabel('Exited')
plt.ylabel('Count')
df['Exited'].value_counts().plot.bar()
plt.show()
```



```
df['Geography'].value_counts()

France     5014
Germany    2509
Spain      2477
Name: Geography, dtype: int64

df =
pd.concat([df,pd.get_dummies(df['Geography'],prefix='Geo')],axis=1)

df = pd.concat([df,pd.get_dummies(df['Gender'])],axis=1)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 19 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  
 14  Geo_France       10000 non-null  uint8  
 15  Geo_Germany      10000 non-null  uint8  
 16  Geo_Spain        10000 non-null  uint8  
 17  Female           10000 non-null  uint8  
 18  Male             10000 non-null  uint8  
dtypes: float64(2), int64(9), object(3), uint8(5)
memory usage: 1.1+ MB
```

```
df.drop(columns=['RowNumber','CustomerId','Surname','Geography','Gende
r'],inplace=True)
```

```
df.head()
```

```
   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   

   IsActiveMember  EstimatedSalary  Exited  Geo_France  Geo_Germany  \
0               1        101348.88       1           1            0   
1               1        112542.58       0           0            0   
2               0        113931.57       1           1            0   
3               0         93826.63       0           1            0   
4               1         79084.10       0           0            0   

   Geo_Spain  Female  Male  
0          0       1     0  
1          1       1     0  
2          0       1     0  
```

```
3          0       1     0
4          1       1     0
```

## Splitting Data

```
y = df['Exited'].values
x = df.loc[:,df.columns != 'Exited'].values

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test =
train_test_split(x,y,random_state=20,test_size=0.25)
```

## Scaling Data

```
from sklearn.preprocessing import StandardScaler
std_x = StandardScaler()
x_train = std_x.fit_transform(x_train)
x_test = std_x.transform(x_test)

x_train.shape

(7500, 13)
```

## Tensorflow Model - Neural Network Classifier

```
import tensorflow as tf
from tensorflow.keras.layers import Dense,Conv1D,Flatten
from tensorflow.keras.models import Sequential, Model

model=Sequential()
model.add(Flatten(input_shape=(13,)))
model.add(Dense(100,activation='relu'))
model.add(Dense(1,activation='sigmoid'))

model.compile(optimizer='adam',metrics=['accuracy'],loss='BinaryCrosse
ntropy')

model.fit(x_train,y_train,batch_size=64,validation_split=0.1,epochs=10
0)

Epoch 1/100
106/106 [==============================] - 1s 2ms/step - loss: 0.5125
- accuracy: 0.7587 - val_loss: 0.4188 - val_accuracy: 0.8240
Epoch 2/100
106/106 [==============================] - 0s 1ms/step - loss: 0.4232
- accuracy: 0.8166 - val_loss: 0.3954 - val_accuracy: 0.8400
Epoch 3/100
106/106 [==============================] - 0s 971us/step - loss:
0.4057 - accuracy: 0.8267 - val_loss: 0.3766 - val_accuracy: 0.8507
Epoch 4/100
106/106 [==============================] - 0s 945us/step - loss:
```

```
0.3900 - accuracy: 0.8342 - val_loss: 0.3611 - val_accuracy: 0.8653
Epoch 5/100
106/106 [==============================] - 0s 940us/step - loss:
0.3775 - accuracy: 0.8415 - val_loss: 0.3479 - val_accuracy: 0.8680
Epoch 6/100
106/106 [==============================] - 0s 965us/step - loss:
0.3673 - accuracy: 0.8470 - val_loss: 0.3356 - val_accuracy: 0.8667
Epoch 7/100
106/106 [==============================] - 0s 958us/step - loss:
0.3607 - accuracy: 0.8511 - val_loss: 0.3314 - val_accuracy: 0.8747
Epoch 8/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3560
- accuracy: 0.8510 - val_loss: 0.3263 - val_accuracy: 0.8693
Epoch 9/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3521
- accuracy: 0.8547 - val_loss: 0.3255 - val_accuracy: 0.8787
Epoch 10/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3495
- accuracy: 0.8567 - val_loss: 0.3246 - val_accuracy: 0.8773
Epoch 11/100
106/106 [==============================] - 0s 975us/step - loss:
0.3486 - accuracy: 0.8545 - val_loss: 0.3229 - val_accuracy: 0.8720
Epoch 12/100
106/106 [==============================] - 0s 934us/step - loss:
0.3463 - accuracy: 0.8563 - val_loss: 0.3203 - val_accuracy: 0.8760
Epoch 13/100
106/106 [==============================] - 0s 959us/step - loss:
0.3443 - accuracy: 0.8573 - val_loss: 0.3188 - val_accuracy: 0.8800
Epoch 14/100
106/106 [==============================] - 0s 939us/step - loss:
0.3436 - accuracy: 0.8588 - val_loss: 0.3214 - val_accuracy: 0.8760
Epoch 15/100
106/106 [==============================] - 0s 997us/step - loss:
0.3429 - accuracy: 0.8573 - val_loss: 0.3170 - val_accuracy: 0.8773
Epoch 16/100
106/106 [==============================] - 0s 974us/step - loss:
0.3420 - accuracy: 0.8573 - val_loss: 0.3179 - val_accuracy: 0.8773
Epoch 17/100
106/106 [==============================] - 0s 978us/step - loss:
0.3414 - accuracy: 0.8556 - val_loss: 0.3147 - val_accuracy: 0.8827
Epoch 18/100
106/106 [==============================] - 0s 984us/step - loss:
0.3403 - accuracy: 0.8588 - val_loss: 0.3166 - val_accuracy: 0.8760
Epoch 19/100
106/106 [==============================] - 0s 955us/step - loss:
0.3391 - accuracy: 0.8597 - val_loss: 0.3143 - val_accuracy: 0.8773
Epoch 20/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3390
- accuracy: 0.8588 - val_loss: 0.3175 - val_accuracy: 0.8760
```

```
Epoch 21/100
106/106 [==============================] - 0s 963us/step - loss:
0.3380 - accuracy: 0.8590 - val_loss: 0.3182 - val_accuracy: 0.8733
Epoch 22/100
106/106 [==============================] - 0s 976us/step - loss:
0.3376 - accuracy: 0.8590 - val_loss: 0.3163 - val_accuracy: 0.8813
Epoch 23/100
106/106 [==============================] - 0s 977us/step - loss:
0.3378 - accuracy: 0.8612 - val_loss: 0.3170 - val_accuracy: 0.8800
Epoch 24/100
106/106 [==============================] - 0s 982us/step - loss:
0.3375 - accuracy: 0.8610 - val_loss: 0.3211 - val_accuracy: 0.8787
Epoch 25/100
106/106 [==============================] - 0s 998us/step - loss:
0.3359 - accuracy: 0.8593 - val_loss: 0.3182 - val_accuracy: 0.8733
Epoch 26/100
106/106 [==============================] - 0s 979us/step - loss:
0.3354 - accuracy: 0.8599 - val_loss: 0.3160 - val_accuracy: 0.8800
Epoch 27/100
106/106 [==============================] - 0s 964us/step - loss:
0.3346 - accuracy: 0.8587 - val_loss: 0.3141 - val_accuracy: 0.8773
Epoch 28/100
106/106 [==============================] - 0s 966us/step - loss:
0.3348 - accuracy: 0.8612 - val_loss: 0.3197 - val_accuracy: 0.8787
Epoch 29/100
106/106 [==============================] - 0s 971us/step - loss:
0.3341 - accuracy: 0.8607 - val_loss: 0.3169 - val_accuracy: 0.8773
Epoch 30/100
106/106 [==============================] - 0s 975us/step - loss:
0.3336 - accuracy: 0.8600 - val_loss: 0.3134 - val_accuracy: 0.8840
Epoch 31/100
106/106 [==============================] - 0s 946us/step - loss:
0.3333 - accuracy: 0.8631 - val_loss: 0.3184 - val_accuracy: 0.8720
Epoch 32/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3326
- accuracy: 0.8628 - val_loss: 0.3148 - val_accuracy: 0.8760
Epoch 33/100
106/106 [==============================] - 0s 949us/step - loss:
0.3323 - accuracy: 0.8628 - val_loss: 0.3142 - val_accuracy: 0.8760
Epoch 34/100
106/106 [==============================] - 0s 992us/step - loss:
0.3316 - accuracy: 0.8652 - val_loss: 0.3187 - val_accuracy: 0.8760
Epoch 35/100
106/106 [==============================] - 0s 970us/step - loss:
0.3319 - accuracy: 0.8630 - val_loss: 0.3133 - val_accuracy: 0.8827
Epoch 36/100
106/106 [==============================] - 0s 988us/step - loss:
0.3305 - accuracy: 0.8637 - val_loss: 0.3191 - val_accuracy: 0.8747
Epoch 37/100
```

```
106/106 [==============================] - 0s 952us/step - loss: 
0.3307 - accuracy: 0.8636 - val_loss: 0.3199 - val_accuracy: 0.8760
Epoch 38/100
106/106 [==============================] - 0s 972us/step - loss: 
0.3304 - accuracy: 0.8619 - val_loss: 0.3212 - val_accuracy: 0.8720
Epoch 39/100
106/106 [==============================] - 0s 981us/step - loss: 
0.3299 - accuracy: 0.8634 - val_loss: 0.3176 - val_accuracy: 0.8720
Epoch 40/100
106/106 [==============================] - 0s 950us/step - loss: 
0.3290 - accuracy: 0.8655 - val_loss: 0.3192 - val_accuracy: 0.8720
Epoch 41/100
106/106 [==============================] - 0s 963us/step - loss: 
0.3293 - accuracy: 0.8643 - val_loss: 0.3142 - val_accuracy: 0.8800
Epoch 42/100
106/106 [==============================] - 0s 990us/step - loss: 
0.3281 - accuracy: 0.8637 - val_loss: 0.3185 - val_accuracy: 0.8747
Epoch 43/100
106/106 [==============================] - 0s 950us/step - loss: 
0.3282 - accuracy: 0.8639 - val_loss: 0.3153 - val_accuracy: 0.8827
Epoch 44/100
106/106 [==============================] - 0s 953us/step - loss: 
0.3278 - accuracy: 0.8636 - val_loss: 0.3116 - val_accuracy: 0.8853
Epoch 45/100
106/106 [==============================] - 0s 992us/step - loss: 
0.3272 - accuracy: 0.8640 - val_loss: 0.3134 - val_accuracy: 0.8787
Epoch 46/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3263
- accuracy: 0.8652 - val_loss: 0.3152 - val_accuracy: 0.8760
Epoch 47/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3260
- accuracy: 0.8631 - val_loss: 0.3190 - val_accuracy: 0.8707
Epoch 48/100
106/106 [==============================] - 0s 997us/step - loss: 
0.3253 - accuracy: 0.8643 - val_loss: 0.3149 - val_accuracy: 0.8747
Epoch 49/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3248
- accuracy: 0.8662 - val_loss: 0.3188 - val_accuracy: 0.8747
Epoch 50/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3245
- accuracy: 0.8649 - val_loss: 0.3195 - val_accuracy: 0.8773
Epoch 51/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3242
- accuracy: 0.8649 - val_loss: 0.3140 - val_accuracy: 0.8800
Epoch 52/100
106/106 [==============================] - 0s 980us/step - loss: 
0.3242 - accuracy: 0.8658 - val_loss: 0.3136 - val_accuracy: 0.8800
Epoch 53/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3235
```

```
- accuracy: 0.8652 - val_loss: 0.3155 - val_accuracy: 0.8720
Epoch 54/100
106/106 [==============================] - 0s 982us/step - loss:
0.3228 - accuracy: 0.8655 - val_loss: 0.3121 - val_accuracy: 0.8813
Epoch 55/100
106/106 [==============================] - 0s 984us/step - loss:
0.3224 - accuracy: 0.8686 - val_loss: 0.3130 - val_accuracy: 0.8760
Epoch 56/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3217
- accuracy: 0.8668 - val_loss: 0.3206 - val_accuracy: 0.8733
Epoch 57/100

106/106 [==============================] - 0s 1ms/step - loss: 0.3214
- accuracy: 0.8668 - val_loss: 0.3104 - val_accuracy: 0.8747
Epoch 58/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3206
- accuracy: 0.8640 - val_loss: 0.3102 - val_accuracy: 0.8867
Epoch 59/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3197
- accuracy: 0.8673 - val_loss: 0.3141 - val_accuracy: 0.8787
Epoch 60/100
106/106 [==============================] - 0s 987us/step - loss:
0.3195 - accuracy: 0.8683 - val_loss: 0.3142 - val_accuracy: 0.8760
Epoch 61/100
106/106 [==============================] - 0s 947us/step - loss:
0.3192 - accuracy: 0.8664 - val_loss: 0.3242 - val_accuracy: 0.8773
Epoch 62/100
106/106 [==============================] - 0s 964us/step - loss:
0.3201 - accuracy: 0.8692 - val_loss: 0.3144 - val_accuracy: 0.8840
Epoch 63/100
106/106 [==============================] - 0s 944us/step - loss:
0.3184 - accuracy: 0.8701 - val_loss: 0.3161 - val_accuracy: 0.8760
Epoch 64/100
106/106 [==============================] - 0s 961us/step - loss:
0.3181 - accuracy: 0.8671 - val_loss: 0.3188 - val_accuracy: 0.8787
Epoch 65/100
106/106 [==============================] - 0s 946us/step - loss:
0.3178 - accuracy: 0.8686 - val_loss: 0.3177 - val_accuracy: 0.8760
Epoch 66/100
106/106 [==============================] - 0s 947us/step - loss:
0.3166 - accuracy: 0.8696 - val_loss: 0.3126 - val_accuracy: 0.8773
Epoch 67/100
106/106 [==============================] - 0s 940us/step - loss:
0.3167 - accuracy: 0.8689 - val_loss: 0.3151 - val_accuracy: 0.8773
Epoch 68/100
106/106 [==============================] - 0s 973us/step - loss:
0.3165 - accuracy: 0.8673 - val_loss: 0.3154 - val_accuracy: 0.8760
Epoch 69/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3159
- accuracy: 0.8690 - val_loss: 0.3110 - val_accuracy: 0.8733
```

```
Epoch 70/100
106/106 [==============================] - 0s 995us/step - loss:
0.3160 - accuracy: 0.8701 - val_loss: 0.3158 - val_accuracy: 0.8773
Epoch 71/100
106/106 [==============================] - 0s 977us/step - loss:
0.3159 - accuracy: 0.8701 - val_loss: 0.3150 - val_accuracy: 0.8773
Epoch 72/100
106/106 [==============================] - 0s 983us/step - loss:
0.3146 - accuracy: 0.8681 - val_loss: 0.3154 - val_accuracy: 0.8747
Epoch 73/100
106/106 [==============================] - 0s 942us/step - loss:
0.3147 - accuracy: 0.8689 - val_loss: 0.3153 - val_accuracy: 0.8760
Epoch 74/100
106/106 [==============================] - 0s 949us/step - loss:
0.3140 - accuracy: 0.8711 - val_loss: 0.3142 - val_accuracy: 0.8733
Epoch 75/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3136
- accuracy: 0.8708 - val_loss: 0.3204 - val_accuracy: 0.8680
Epoch 76/100
106/106 [==============================] - 0s 950us/step - loss:
0.3132 - accuracy: 0.8707 - val_loss: 0.3125 - val_accuracy: 0.8787
Epoch 77/100
106/106 [==============================] - 0s 955us/step - loss:
0.3127 - accuracy: 0.8707 - val_loss: 0.3204 - val_accuracy: 0.8720
Epoch 78/100
106/106 [==============================] - 0s 960us/step - loss:
0.3132 - accuracy: 0.8707 - val_loss: 0.3180 - val_accuracy: 0.8733
Epoch 79/100
106/106 [==============================] - 0s 958us/step - loss:
0.3123 - accuracy: 0.8701 - val_loss: 0.3148 - val_accuracy: 0.8733
Epoch 80/100
106/106 [==============================] - 0s 945us/step - loss:
0.3123 - accuracy: 0.8724 - val_loss: 0.3115 - val_accuracy: 0.8787
Epoch 81/100
106/106 [==============================] - 0s 955us/step - loss:
0.3115 - accuracy: 0.8730 - val_loss: 0.3137 - val_accuracy: 0.8773
Epoch 82/100
106/106 [==============================] - 0s 969us/step - loss:
0.3118 - accuracy: 0.8707 - val_loss: 0.3161 - val_accuracy: 0.8720
Epoch 83/100
106/106 [==============================] - 0s 964us/step - loss:
0.3112 - accuracy: 0.8683 - val_loss: 0.3089 - val_accuracy: 0.8773
Epoch 84/100
106/106 [==============================] - 0s 936us/step - loss:
0.3104 - accuracy: 0.8711 - val_loss: 0.3088 - val_accuracy: 0.8813
Epoch 85/100
106/106 [==============================] - 0s 939us/step - loss:
0.3106 - accuracy: 0.8698 - val_loss: 0.3159 - val_accuracy: 0.8707
Epoch 86/100
```

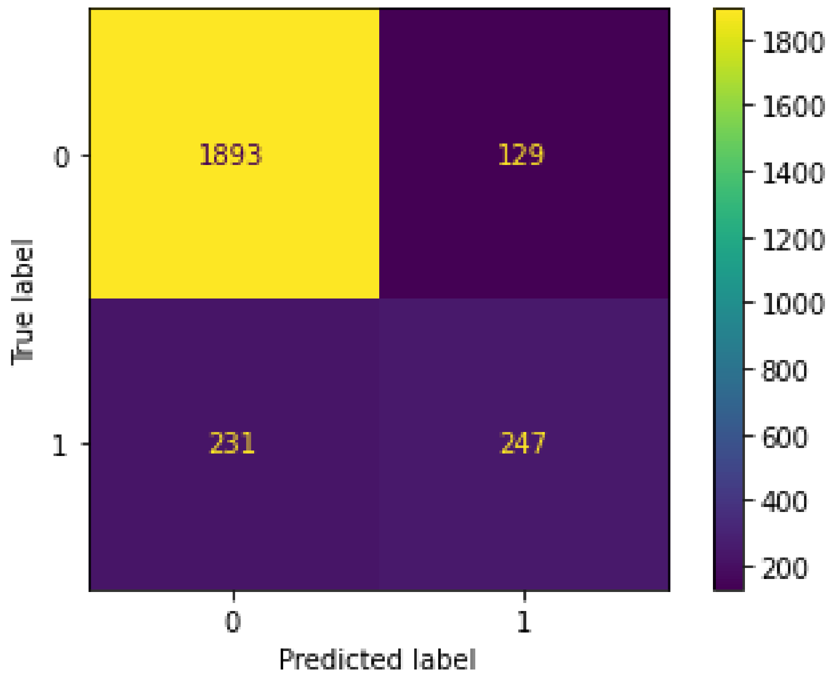
```
106/106 [==============================] - 0s 946us/step - loss:
0.3100 - accuracy: 0.8736 - val_loss: 0.3140 - val_accuracy: 0.8747
Epoch 87/100
106/106 [==============================] - 0s 937us/step - loss:
0.3092 - accuracy: 0.8738 - val_loss: 0.3138 - val_accuracy: 0.8733
Epoch 88/100
106/106 [==============================] - 0s 945us/step - loss:
0.3088 - accuracy: 0.8719 - val_loss: 0.3118 - val_accuracy: 0.8760
Epoch 89/100
106/106 [==============================] - 0s 957us/step - loss:
0.3084 - accuracy: 0.8720 - val_loss: 0.3065 - val_accuracy: 0.8827
Epoch 90/100
106/106 [==============================] - 0s 943us/step - loss:
0.3091 - accuracy: 0.8735 - val_loss: 0.3104 - val_accuracy: 0.8760
Epoch 91/100
106/106 [==============================] - 0s 972us/step - loss:
0.3088 - accuracy: 0.8729 - val_loss: 0.3142 - val_accuracy: 0.8747
Epoch 92/100
106/106 [==============================] - 0s 942us/step - loss:
0.3071 - accuracy: 0.8730 - val_loss: 0.3196 - val_accuracy: 0.8653
Epoch 93/100
106/106 [==============================] - 0s 945us/step - loss:
0.3079 - accuracy: 0.8729 - val_loss: 0.3176 - val_accuracy: 0.8720
Epoch 94/100
106/106 [==============================] - 0s 946us/step - loss:
0.3077 - accuracy: 0.8723 - val_loss: 0.3145 - val_accuracy: 0.8827
Epoch 95/100
106/106 [==============================] - 0s 978us/step - loss:
0.3070 - accuracy: 0.8733 - val_loss: 0.3126 - val_accuracy: 0.8787
Epoch 96/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3073
- accuracy: 0.8748 - val_loss: 0.3122 - val_accuracy: 0.8747
Epoch 97/100
106/106 [==============================] - 0s 1ms/step - loss: 0.3068
- accuracy: 0.8716 - val_loss: 0.3169 - val_accuracy: 0.8773
Epoch 98/100
106/106 [==============================] - 0s 953us/step - loss:
0.3067 - accuracy: 0.8738 - val_loss: 0.3092 - val_accuracy: 0.8747
Epoch 99/100
106/106 [==============================] - 0s 947us/step - loss:
0.3053 - accuracy: 0.8742 - val_loss: 0.3149 - val_accuracy: 0.8813
Epoch 100/100
106/106 [==============================] - 0s 985us/step - loss:
0.3060 - accuracy: 0.8741 - val_loss: 0.3257 - val_accuracy: 0.8680
```

```
<keras.callbacks.History at 0x7f42ce4a5e20>
```

```
pred = model.predict(x_test)
```

```
79/79 [==============================] - 0s 541us/step
```

```
y_pred = []
for val in pred:
    if val > 0.5:
        y_pred.append(1)
    else:
        y_pred.append(0)

from sklearn.metrics import
accuracy_score,confusion_matrix,ConfusionMatrixDisplay

accuracy_score(y_test,y_pred)

0.856

cm = confusion_matrix(y_test,y_pred)
display = ConfusionMatrixDisplay(cm)
display.plott()

<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x7f425ca63520>
```



```
```