Bank Customer Analysis

February 27, 2022

1 Libraries

[2]:

```
import numpy as np
import pandas as pd
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import OneHotEncoder
```

2 Dataset

[3]:

```
dataset = pd.read_csv('Churn_Modelling.csv')
X = dataset.iloc[:, 3:-1].values
y = dataset.iloc[:, -1].values
```

3 Label Encoder

[4]:

```
# gender column
le = LabelEncoder()
X[:, 2] = le.fit_transform(X[:, 2])
```

4 One Hot Encoding

[5]:

```
# location column
# in order to change simply change the [1] to any index number that you want to␣
 ↪do hot encoding
ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [1])],␣
 ↪remainder='passthrough')
X = np.array(ct.fit_transform(X))
```

5 Train and Test

[16]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,␣
 ↪random_state = 0)
```

6 Feature Scaling

[17]:
```
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test  = sc.transform(X_test)
```

7 Artificial Neural Networks (ANN)

[18]:
```
# initialiing ann

ann = tf.keras.models.Sequential()

# adding first hidden layer

ann.add(tf.keras.layers.Dense(units=6, activation='relu'))

# adding scond layer

ann.add(tf.keras.layers.Dense(units=6, activation='relu'))

# adding output layer

ann.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))

# for binany 'sigmoid'
# for non binary 'softmax' (when prediction more than two categories)
```

8 Training a Artificial Neural Network

8.1 Compiling ANN

[19]:
```
ann.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics =␣
 ↪['accuracy'])

# loss -> for binary classification 'binary_crossentropy'
# loss -> for non binary classifation 'categorilcal_crossentropy'
```

8.2 Training a ANN on the Traning set

[20]:

```
ann.fit(X_train, y_train, batch_size=32, epochs=100)
# batch_size can be change default 32
# epochs can be change
```

```
Epoch 1/100
250/250 [==============================] - 0s 763us/step - loss: 0.5946 -
accuracy: 0.7825
Epoch 2/100
250/250 [==============================] - 0s 772us/step - loss: 0.4697 -
accuracy: 0.8083
Epoch 3/100
250/250 [==============================] - 0s 781us/step - loss: 0.4408 -
accuracy: 0.8179
Epoch 4/100
250/250 [==============================] - 0s 767us/step - loss: 0.4297 -
accuracy: 0.8199
Epoch 5/100
250/250 [==============================] - 0s 767us/step - loss: 0.4197 -
accuracy: 0.8214
Epoch 6/100
250/250 [==============================] - 0s 739us/step - loss: 0.4098 -
accuracy: 0.8241
Epoch 7/100
250/250 [==============================] - 0s 739us/step - loss: 0.3987 -
accuracy: 0.8250
Epoch 8/100
250/250 [==============================] - 0s 755us/step - loss: 0.3889 -
accuracy: 0.8314
Epoch 9/100
250/250 [==============================] - 0s 775us/step - loss: 0.3803 -
accuracy: 0.8324
Epoch 10/100
250/250 [==============================] - 0s 767us/step - loss: 0.3738 -
accuracy: 0.8335
Epoch 11/100
250/250 [==============================] - 0s 726us/step - loss: 0.3694 -
accuracy: 0.8334
Epoch 12/100
250/250 [==============================] - 0s 728us/step - loss: 0.3663 -
accuracy: 0.8345
Epoch 13/100
250/250 [==============================] - 0s 739us/step - loss: 0.3630 -
accuracy: 0.8357
Epoch 14/100
250/250 [==============================] - 0s 731us/step - loss: 0.3607 -
accuracy: 0.8349
```

```
Epoch 15/100
250/250 [==============================] - 0s 735us/step - loss: 0.3582 -
accuracy: 0.8357
Epoch 16/100
250/250 [==============================] - 0s 739us/step - loss: 0.3566 -
accuracy: 0.8400
Epoch 17/100
250/250 [==============================] - 0s 763us/step - loss: 0.3548 -
accuracy: 0.8556
Epoch 18/100
250/250 [==============================] - 0s 726us/step - loss: 0.3535 -
accuracy: 0.8581
Epoch 19/100
250/250 [==============================] - 0s 722us/step - loss: 0.3521 -
accuracy: 0.8585
Epoch 20/100
250/250 [==============================] - 0s 775us/step - loss: 0.3505 -
accuracy: 0.8594
Epoch 21/100
250/250 [==============================] - 0s 726us/step - loss: 0.3496 -
accuracy: 0.8585
Epoch 22/100
250/250 [==============================] - 0s 718us/step - loss: 0.3485 -
accuracy: 0.8602
Epoch 23/100
250/250 [==============================] - 0s 723us/step - loss: 0.3476 -
accuracy: 0.8596
Epoch 24/100
250/250 [==============================] - 0s 743us/step - loss: 0.3464 -
accuracy: 0.8621
Epoch 25/100
250/250 [==============================] - 0s 743us/step - loss: 0.3459 -
accuracy: 0.8614
Epoch 26/100
250/250 [==============================] - 0s 770us/step - loss: 0.3451 -
accuracy: 0.8604
Epoch 27/100
250/250 [==============================] - 0s 790us/step - loss: 0.3441 -
accuracy: 0.8611
Epoch 28/100
250/250 [==============================] - 0s 794us/step - loss: 0.3438 -
accuracy: 0.8619
Epoch 29/100
250/250 [==============================] - 0s 761us/step - loss: 0.3427 -
accuracy: 0.8634
Epoch 30/100
250/250 [==============================] - 0s 766us/step - loss: 0.3424 -
accuracy: 0.8619
```

```
Epoch 31/100
250/250 [==============================] - 0s 801us/step - loss: 0.3422 -
accuracy: 0.8600
Epoch 32/100
250/250 [==============================] - 0s 744us/step - loss: 0.3413 -
accuracy: 0.8611
Epoch 33/100
250/250 [==============================] - 0s 771us/step - loss: 0.3409 -
accuracy: 0.8609
Epoch 34/100
250/250 [==============================] - 0s 731us/step - loss: 0.3404 -
accuracy: 0.8619
Epoch 35/100
250/250 [==============================] - 0s 731us/step - loss: 0.3405 -
accuracy: 0.8625
Epoch 36/100
250/250 [==============================] - 0s 735us/step - loss: 0.3401 -
accuracy: 0.8631
Epoch 37/100
250/250 [==============================] - 0s 743us/step - loss: 0.3395 -
accuracy: 0.8634
Epoch 38/100
250/250 [==============================] - 0s 739us/step - loss: 0.3390 -
accuracy: 0.8619
Epoch 39/100
250/250 [==============================] - 0s 735us/step - loss: 0.3383 -
accuracy: 0.8646
Epoch 40/100
250/250 [==============================] - 0s 759us/step - loss: 0.3386 -
accuracy: 0.8622
Epoch 41/100
250/250 [==============================] - 0s 747us/step - loss: 0.3381 -
accuracy: 0.8625
Epoch 42/100
250/250 [==============================] - 0s 739us/step - loss: 0.3377 -
accuracy: 0.8629
Epoch 43/100
250/250 [==============================] - 0s 763us/step - loss: 0.3373 -
accuracy: 0.8635
Epoch 44/100
250/250 [==============================] - 0s 727us/step - loss: 0.3369 -
accuracy: 0.8618
Epoch 45/100
250/250 [==============================] - 0s 739us/step - loss: 0.3371 -
accuracy: 0.8620
Epoch 46/100
250/250 [==============================] - 0s 763us/step - loss: 0.3367 -
accuracy: 0.8624
```

```
Epoch 47/100
250/250 [==============================] - 0s 731us/step - loss: 0.3361 -
accuracy: 0.8630
Epoch 48/100
250/250 [==============================] - 0s 722us/step - loss: 0.3361 -
accuracy: 0.8637
Epoch 49/100
250/250 [==============================] - 0s 723us/step - loss: 0.3358 -
accuracy: 0.8648
Epoch 50/100
250/250 [==============================] - 0s 767us/step - loss: 0.3359 -
accuracy: 0.8621
Epoch 51/100
250/250 [==============================] - 0s 732us/step - loss: 0.3355 -
accuracy: 0.8634
Epoch 52/100
250/250 [==============================] - 0s 752us/step - loss: 0.3354 -
accuracy: 0.8634
Epoch 53/100
250/250 [==============================] - 0s 765us/step - loss: 0.3348 -
accuracy: 0.8635
Epoch 54/100
250/250 [==============================] - 0s 750us/step - loss: 0.3349 -
accuracy: 0.8634
Epoch 55/100
250/250 [==============================] - 0s 739us/step - loss: 0.3345 -
accuracy: 0.8622
Epoch 56/100
250/250 [==============================] - 0s 810us/step - loss: 0.3345 -
accuracy: 0.8634
Epoch 57/100
250/250 [==============================] - 0s 743us/step - loss: 0.3343 -
accuracy: 0.8646
Epoch 58/100
250/250 [==============================] - 0s 733us/step - loss: 0.3339 -
accuracy: 0.8631
Epoch 59/100
250/250 [==============================] - 0s 718us/step - loss: 0.3339 -
accuracy: 0.8635
Epoch 60/100
250/250 [==============================] - 0s 726us/step - loss: 0.3337 -
accuracy: 0.8627
Epoch 61/100
250/250 [==============================] - 0s 735us/step - loss: 0.3336 -
accuracy: 0.8631
Epoch 62/100
250/250 [==============================] - 0s 738us/step - loss: 0.3331 -
accuracy: 0.8634
```

```
Epoch 63/100
250/250 [==============================] - 0s 742us/step - loss: 0.3332 -
accuracy: 0.8652
Epoch 64/100
250/250 [==============================] - 0s 743us/step - loss: 0.3328 -
accuracy: 0.8639
Epoch 65/100
250/250 [==============================] - 0s 735us/step - loss: 0.3326 -
accuracy: 0.8651
Epoch 66/100
250/250 [==============================] - 0s 771us/step - loss: 0.3327 -
accuracy: 0.8636
Epoch 67/100
250/250 [==============================] - 0s 747us/step - loss: 0.3325 -
accuracy: 0.8637
Epoch 68/100
250/250 [==============================] - 0s 726us/step - loss: 0.3323 -
accuracy: 0.8640
Epoch 69/100
250/250 [==============================] - 0s 751us/step - loss: 0.3320 -
accuracy: 0.8634
Epoch 70/100
250/250 [==============================] - 0s 743us/step - loss: 0.3319 -
accuracy: 0.8636
Epoch 71/100
250/250 [==============================] - 0s 722us/step - loss: 0.3316 -
accuracy: 0.8635
Epoch 72/100
250/250 [==============================] - 0s 718us/step - loss: 0.3315 -
accuracy: 0.8648
Epoch 73/100
250/250 [==============================] - 0s 717us/step - loss: 0.3320 -
accuracy: 0.8631
Epoch 74/100
250/250 [==============================] - 0s 726us/step - loss: 0.3311 -
accuracy: 0.8635
Epoch 75/100
250/250 [==============================] - 0s 714us/step - loss: 0.3308 -
accuracy: 0.8644
Epoch 76/100
250/250 [==============================] - 0s 726us/step - loss: 0.3311 -
accuracy: 0.8631
Epoch 77/100
250/250 [==============================] - 0s 722us/step - loss: 0.3308 -
accuracy: 0.8656
Epoch 78/100
250/250 [==============================] - 0s 723us/step - loss: 0.3305 -
accuracy: 0.8645
```

```
Epoch 79/100
250/250 [==============================] - 0s 718us/step - loss: 0.3310 -
accuracy: 0.8631
Epoch 80/100
250/250 [==============================] - 0s 714us/step - loss: 0.3306 -
accuracy: 0.8650
Epoch 81/100
250/250 [==============================] - 0s 714us/step - loss: 0.3302 -
accuracy: 0.8644
Epoch 82/100
250/250 [==============================] - 0s 711us/step - loss: 0.3307 -
accuracy: 0.8650
Epoch 83/100
250/250 [==============================] - 0s 731us/step - loss: 0.3304 -
accuracy: 0.8654
Epoch 84/100
250/250 [==============================] - 0s 726us/step - loss: 0.3306 -
accuracy: 0.8661
Epoch 85/100
250/250 [==============================] - 0s 731us/step - loss: 0.3303 -
accuracy: 0.8649
Epoch 86/100
250/250 [==============================] - 0s 726us/step - loss: 0.3301 -
accuracy: 0.8637
Epoch 87/100
250/250 [==============================] - 0s 744us/step - loss: 0.3299 -
accuracy: 0.8652
Epoch 88/100
250/250 [==============================] - 0s 746us/step - loss: 0.3302 -
accuracy: 0.8639
Epoch 89/100
250/250 [==============================] - 0s 744us/step - loss: 0.3300 -
accuracy: 0.8649
Epoch 90/100
250/250 [==============================] - 0s 741us/step - loss: 0.3298 -
accuracy: 0.8650
Epoch 91/100
250/250 [==============================] - 0s 749us/step - loss: 0.3295 -
accuracy: 0.8645
Epoch 92/100
250/250 [==============================] - 0s 745us/step - loss: 0.3298 -
accuracy: 0.8645
Epoch 93/100
250/250 [==============================] - 0s 747us/step - loss: 0.3293 -
accuracy: 0.8637
Epoch 94/100
250/250 [==============================] - 0s 718us/step - loss: 0.3294 -
accuracy: 0.8649
```

```
Epoch 95/100
250/250 [==============================] - 0s 726us/step - loss: 0.3292 -
accuracy: 0.8643
Epoch 96/100
250/250 [==============================] - 0s 735us/step - loss: 0.3294 -
accuracy: 0.8648
Epoch 97/100
250/250 [==============================] - 0s 735us/step - loss: 0.3289 -
accuracy: 0.8656
Epoch 98/100
250/250 [==============================] - 0s 738us/step - loss: 0.3290 -
accuracy: 0.8635
Epoch 99/100
250/250 [==============================] - 0s 750us/step - loss: 0.3290 -
accuracy: 0.8654
Epoch 100/100
250/250 [==============================] - 0s 745us/step - loss: 0.3290 -
accuracy: 0.8649
```

[20]:

```
<keras.callbacks.History at 0x249f0a47f70>
```

9 Making Prediction and Evaluating the Model

[ ]:

```
# Location France
# Credit Score 600
# Gender Male
# Age 40
# Tenure 3
# Balance $60000
# Number of products 2
# Does this cuntomer have e credit card ? Yes
# is this customer an active member ? Yes
# Estimated Salary $50000
# Should we say goodbye to that customer ?
```

[21]:

```
# 1,0,0 -> france
# 1 -> Male
# credit card 1 -> Yes
# active member 1 -> Yes
# 0.5 can be change

# Accuracy
#print(ann.predict(sc.transform([[1,0,0, 600, 1, 40, 3, 60000, 2, 1, 1,␣
 ↪50000]])))

# Should we say good bye to that customer ?
```

```
print(ann.predict(sc.transform([[1,0,0, 600, 1, 40, 3, 60000, 2, 1, 1,␣
 ↪50000]])) > 0.5)
```

```
[[False]]
```

10 Predicting the Test Result

[22]:

```
y_pred = ann.predict(X_test)
y_pred = (y_pred > 0.5)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.
 ↪reshape(len(y_test),1)),1))

# first 0 predicted stay & actual stay [0 0]
# scond 0 predicted stay & actual left [0 1]
```

```
[[0 0]
 [0 1]
 [0 0]
 …
 [0 0]
 [0 0]
 [0 0]]
```

11 Confusion Matrix

[23]:

```
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)

# 1510 corrected prediction for staying in bank
#  209 corrected prediction for leaving the bank
#  196 incorrect prediction for staying in bank
#   85 incorrect prediction for leaving the bank
```

```
[[1510   85]
 [ 196  209]]
```

[23]:

```
0.8595
```

[ ]: