

Assignment No. : 3

Link to the Kaggle project: <https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling>
Perform following steps:

- 1. Read the dataset.
- 2. Distinguish the feature and target set and divide the data set into training and test sets.
- 3. Normalize the train and test data.
- 4. Initialize and build the model. Identify the points of improvement and implement the same.
- 5. Print the accuracy score and confusion matrix (5 points).


```
# 1.Read the dataset
import pandas as pd
data = pd.read_csv("/content/Churn_Modelling.csv")
data.head()
```

 Show hidden output

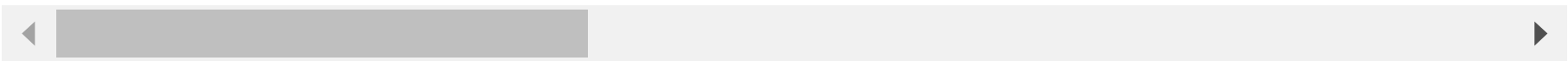
```
data.isnull().sum()
```

 Show hidden output

```
data=data.replace({'Female': 0, 'Male': 1})
data.head()
```



	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	Nu
0	1	15634602	Hargrave	619	France	0	42	2	0.00	
1	2	15647311	Hill	608	Spain	0	41	1	83807.86	
2	3	15619304	Onio	502	France	0	42	8	159660.80	
3	4	15701354	Boni	699	France	0	39	1	0.00	
4	5	15737888	Mitchell	850	Spain	0	43	2	125510.82	



Next steps: [Generate code with data](#) [View recommended plots](#) [New interactive sheet](#)

```
# 2.Distinguish the feature and target set and divide the data set into training and
from sklearn.model_selection import train_test_split
```

```
# Define feature set (X) and target variable (y)
X = data.drop(columns=['RowNumber', 'CustomerId', 'Surname', 'Geography', 'Exited'])
y = data['Exited']
```

```
# Split the dataset into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_stat
```

```
# 3.Normalize the train and test data.
from sklearn.preprocessing import StandardScaler
```

```
# Normalize the features
```

```

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# 4.Initialize and build the model. Identify the points of improvement and implement it
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.callbacks import EarlyStopping

# Build the model
model = Sequential()
model.add(Dense(32, activation='relu', input_shape=(X_train_scaled.shape[1],)))
model.add(Dense(16, activation='relu'))
model.add(Dense(1, activation='sigmoid')) # Binary classification

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Early stopping to prevent overfitting
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Train the model
history = model.fit(X_train_scaled, y_train, epochs=100, validation_split=0.2, callbacks=[early_stopping])

```

 [Show hidden output](#)

```



# 5. Evaluate the Model
from sklearn.metrics import accuracy_score, confusion_matrix

# Make predictions
y_pred = (model.predict(X_test_scaled) > 0.5).astype("int32")

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)

```

 **63/63**  **0s** 2ms/step
 Accuracy: 0.8580
 Confusion Matrix:
 [[1538 69]
 [215 178]]