In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarning: A N
umPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected
version 1.26.1
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:

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

In [3]:

```
df.head(10)
```

Out[3]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance
0	1	15634602	Hargrave	619	France	Female	42	2	0.00
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86
2	3	15619304	Onio	502	France	Female	42	8	159660.80
3	4	15701354	Boni	699	France	Female	39	1	0.00
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82
5	6	15574012	Chu	645	Spain	Male	44	8	113755.78
6	7	15592531	Bartlett	822	France	Male	50	7	0.00
7	8	15656148	Obinna	376	Germany	Female	29	4	115046.74
8	9	15792365	He	501	France	Male	44	4	142051.07
9	10	15592389	H?	684	France	Male	27	2	134603.88

In [4]:

```
df.sample(10)
```

Out[4]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Bala
2346	2347	15706163	Enyinnaya	518	Germany	Male	46	4	113625
2623	2624	15653696	Goliwe	515	France	Female	28	9	0
6423	6424	15600720	Moore	652	Spain	Male	41	8	115144
1125	1126	15645316	Han	612	Germany	Female	58	1	149641
7455	7456	15748499	Johnson	550	Germany	Male	33	4	118400
8285	8286	15572631	Ndubuisi	609	France	Male	25	10	0
9326	9327	15601787	Greco	641	Germany	Male	35	2	103711
7224	7225	15609823	Chieloka	751	Spain	Female	34	8	127095
9436	9437	15771000	Powell	684	France	Male	38	4	0
2857	2858	15769829	Cheng	534	Spain	Male	51	3	0

```
In [5]: df.isnull().sum()

Out[5]: 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
```

```
In [6]: df.shape

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

```
In [7]: df.drop(["RowNumber","CustomerId","Surname"],axis=1,inplace=True)
```

```
In [8]: df.shape

Out[8]: (10000, 11)
```

```
In [9]: df.dtypes

Out[9]: CreditScore          int64
        Geography           object
        Gender              object
        Age                  int64
        Tenure               int64
        Balance            float64
        NumOfProducts        int64
        HasCrCard            int64
        IsActiveMember       int64
        EstimatedSalary    float64
        Exited               int64
        dtype: object
```

```
In [10]: df=pd.get_dummies(df,columns=["Geography","Gender"])
```

```
In [11]: df.sample(10)
```

Out[11]:

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estimate
1188	528	30	2	128262.72	2	1	0	5
6301	766	45	6	97652.96	1	1	0	12
2375	815	39	6	0.00	1	1	1	8
5804	625	35	5	86147.46	2	1	0	16
6692	662	39	5	138106.75	1	0	0	1
2483	750	37	3	0.00	2	1	0	1
9493	664	36	0	103502.22	1	1	1	14
3397	820	33	2	132150.26	2	1	0	2
174	512	40	5	0.00	2	1	1	14
5539	614	39	3	151914.93	1	0	0	5

In [12]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [13]:
```
X = df.drop(['Exited'], axis=1)

y = df[['Exited']]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

In [14]:
```
model_dtc = DecisionTreeClassifier(max_depth=6)
```

In [15]:
```
model_dtc.fit(X_train,y_train)
```

Out[15]:
```
DecisionTreeClassifier(max_depth=6)
```

In [16]:
```
model_dtc.score(X_train,y_train)
```

Out[16]:
```
0.8688571428571429
```

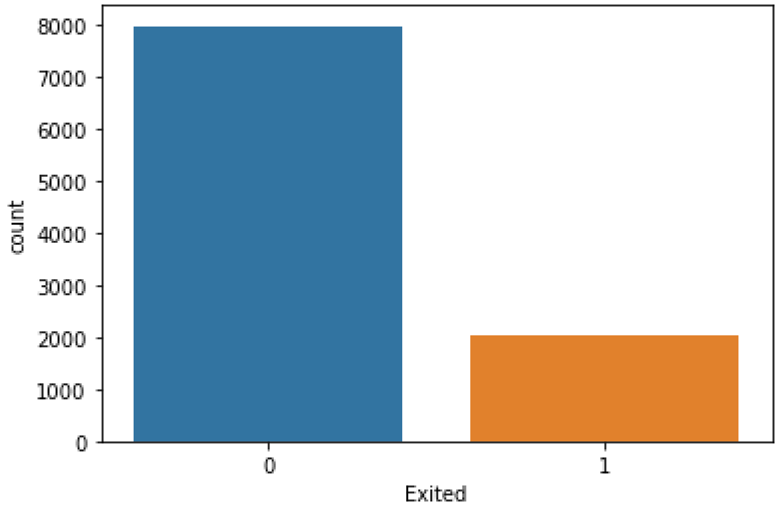
In [17]:
```
model_dtc.score(X_test,y_test)
```

Out[17]:
```
0.8606666666666667
```

VISUALIZATION

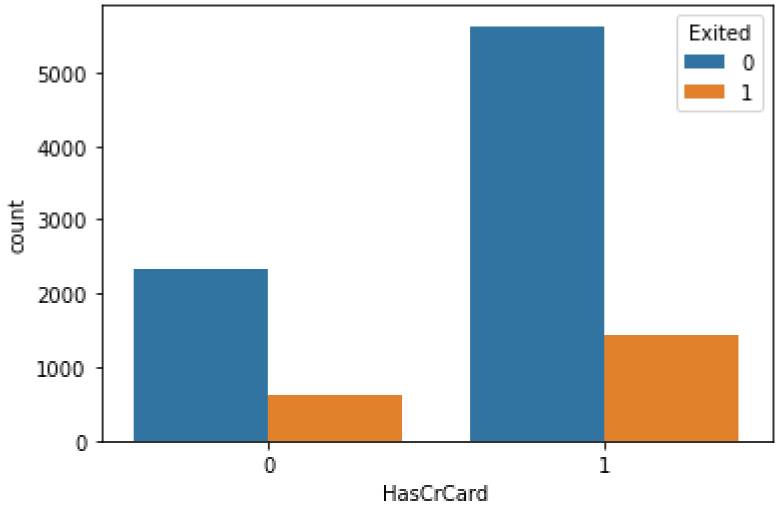
In [18]:
```
sns.countplot(x="Exited",data=df)
```

Out[18]:
```
<AxesSubplot:xlabel='Exited', ylabel='count'>
```



```
In [19]: sns.countplot(x="HasCrCard",hue="Exited",data=df)

Out[19]: <AxesSubplot:xlabel='HasCrCard', ylabel='count'>
```



```
In [20]: from sklearn.ensemble import BaggingClassifier
```

```
In [21]: model_bc = BaggingClassifier(n_estimators=250,estimator=model_dtc,warm_start=True)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [21], in <cell line: 1>()
----> 1 model_bc = BaggingClassifier(n_estimators=250,estimator=model_dtc,warm_sta
rt=True)

TypeError: __init__() got an unexpected keyword argument 'estimator'
```

```
In [ ]: model_bc.fit(X_train,y_train)
```

```
In [ ]: model_bc.score(X_train,y_train)
```

```
In [ ]: model_bc.score(X_test,y_test)
```

```
In [25]: from sklearn.ensemble import AdaBoostClassifier
         from sklearn.ensemble import GradientBoostingClassifier
         from sklearn.ensemble import RandomForestClassifier
```

```
In [26]: model_abc = AdaBoostClassifier(learning_rate=5,n_estimators=500)
```

In [27]:

```
model_abc.fit(X_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:993: DataCo
nversionWarning: A column-vector y was passed when a 1d array was expected. Please
change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[27]:

```
AdaBoostClassifier(learning_rate=5, n_estimators=500)
```

In [28]:

```
model_abc.score(X_train,y_train)
```

Out[28]:

```
0.20357142857142857
```

In [29]:

```
model_abc.score(X_test,y_test)
```

Out[29]:

```
0.204
```

In [30]:

```
model_gb = GradientBoostingClassifier()
```

In [31]:

```
model_gb.fit(X_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494: DataConver
sionWarning: A column-vector y was passed when a 1d array was expected. Please cha
nge the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[31]:

```
GradientBoostingClassifier()
```

In [32]:

```
model_gb.score(X_train,y_train)
```

Out[32]:

```
0.8757142857142857
```

In [33]:

```
8
```

Out[33]:

```
8
```

In [34]:

```
model_rf = RandomForestClassifier()
```

In [35]:

```
model_rf.fit(X_train,y_train)
model_rf.score(X_train,y_train)
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_4452\771388674.py:1: DataConversionWarn
ing: A column-vector y was passed when a 1d array was expected. Please change the
shape of y to (n_samples,), for example using ravel().
  model_rf.fit(X_train,y_train)
```

Out[35]:

```
1.0
```

In [36]:

```
model_rf.score(X_test,y_test)
```

Out[36]:

```
0.8633333333333333
```

In [37]:

```
from tensorflow.keras.models import Sequential
```

In [38]:

```
from tensorflow.keras.layers import Dense
```

In [39]:

```
classifier = Sequential()
classifier.add(Dense(units=10,kernel_initializer='uniform',activation='relu',input_

classifier.add(Dense(units=10,kernel_initializer='uniform',activation='relu'))
classifier.add(Dense(units=1,kernel_initializer='uniform',activation='sigmoid'))

classifier.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy']
classifier.fit(X_train,y_train,batch_size=10,epochs=10,validation_split=0.1)
```

```
Epoch 1/10
630/630 [==============================] - 3s 4ms/step - loss: 0.5963 - accuracy:
0.7816 - val_loss: 0.5368 - val_accuracy: 0.7957
Epoch 2/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5362 - accuracy:
0.7960 - val_loss: 0.5310 - val_accuracy: 0.7957
Epoch 3/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5298 - accuracy:
0.7963 - val_loss: 0.5261 - val_accuracy: 0.7957
Epoch 4/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5189 - accuracy:
0.7965 - val_loss: 0.5216 - val_accuracy: 0.7957
Epoch 5/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5172 - accuracy:
0.7965 - val_loss: 0.5162 - val_accuracy: 0.7957
Epoch 6/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5107 - accuracy:
0.7963 - val_loss: 0.5184 - val_accuracy: 0.7957
Epoch 7/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5084 - accuracy:
0.7965 - val_loss: 0.5072 - val_accuracy: 0.7957
Epoch 8/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5035 - accuracy:
0.7965 - val_loss: 0.5176 - val_accuracy: 0.7957
Epoch 9/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5017 - accuracy:
0.7960 - val_loss: 0.5058 - val_accuracy: 0.7957
Epoch 10/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5005 - accuracy:
0.7965 - val_loss: 0.5032 - val_accuracy: 0.7957
```

Out[39]:

```
<keras.src.callbacks.History at 0x209744abe80>
```

In [40]:

```
classifier = Sequential()
classifier.add(Dense(units=6,kernel_initializer='uniform',activation='relu',input_

classifier.add(Dense(units=6,kernel_initializer='uniform',activation='relu'))

classifier.add(Dense(units=1,kernel_initializer='uniform',activation='sigmoid'))

classifier.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy']
classifier.fit(X_train,y_train,batch_size=10,epochs=10,validation_split=0.1)
```

```
Epoch 1/10
630/630 [==============================] - 3s 3ms/step - loss: 0.5678 - accuracy:
0.7894 - val_loss: 0.5452 - val_accuracy: 0.7957
Epoch 2/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5280 - accuracy:
0.7965 - val_loss: 0.5259 - val_accuracy: 0.7957
Epoch 3/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5210 - accuracy:
0.7960 - val_loss: 0.5202 - val_accuracy: 0.7957
Epoch 4/10
630/630 [==============================] - 2s 2ms/step - loss: 0.5195 - accuracy:
0.7965 - val_loss: 0.5180 - val_accuracy: 0.7957
Epoch 5/10
630/630 [==============================] - 2s 3ms/step - loss: 0.5110 - accuracy:
0.7965 - val_loss: 0.5084 - val_accuracy: 0.7957
Epoch 6/10
630/630 [==============================] - 2s 4ms/step - loss: 0.5030 - accuracy:
0.7965 - val_loss: 0.5034 - val_accuracy: 0.7957
Epoch 7/10
630/630 [==============================] - 3s 5ms/step - loss: 0.5013 - accuracy:
0.7965 - val_loss: 0.5061 - val_accuracy: 0.7957
Epoch 8/10
630/630 [==============================] - 4s 6ms/step - loss: 0.4991 - accuracy:
0.7965 - val_loss: 0.5042 - val_accuracy: 0.7957
Epoch 9/10
630/630 [==============================] - 4s 6ms/step - loss: 0.4996 - accuracy:
0.7965 - val_loss: 0.5012 - val_accuracy: 0.7957
Epoch 10/10
630/630 [==============================] - 3s 5ms/step - loss: 0.4994 - accuracy:
0.7965 - val_loss: 0.5026 - val_accuracy: 0.7957
```

```
Out[40]: <keras.src.callbacks.History at 0x20976982760>
```

```
In [ ]:
```