Logistic Regression on Bank churn dataset.

In [1]:
```
#importing the libraries

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

In [2]:
```
# lets load the  dataset

bank_data =pd.read_csv("G:/dataset files/Bank_churn_modelling.csv")
```

In [3]:
```
#viewing the data using head by which we can see the top 5 rows.

bank_data.head()
```

Out[3]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balan
0	1	15634602	Hargrave	619	France	Female	42	2	0.
1	2	15647311	Hill	608	Spain	Female	41	1	83807.
2	3	15619304	Onio	502	France	Female	42	8	159660.
3	4	15701354	Boni	699	France	Female	39	1	0.
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.

In [4]:
```
# to see the columns in dataset

bank_data.columns
```

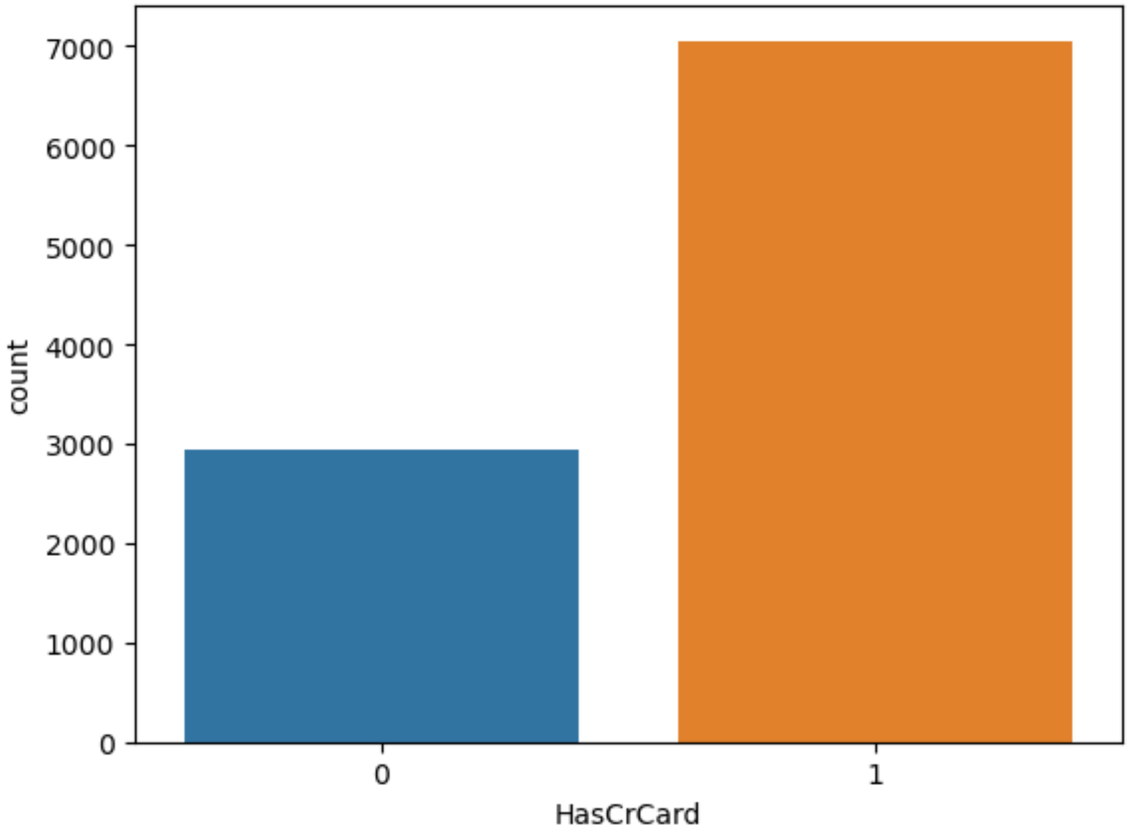
Out[4]:
```
Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
       'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
       'IsActiveMember', 'EstimatedSalary', 'Exited'],
      dtype='object')
```

In [5]:
```
# lets check the datatypes for each columns

bank_data.dtypes
```

Out[5]:
```
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
```

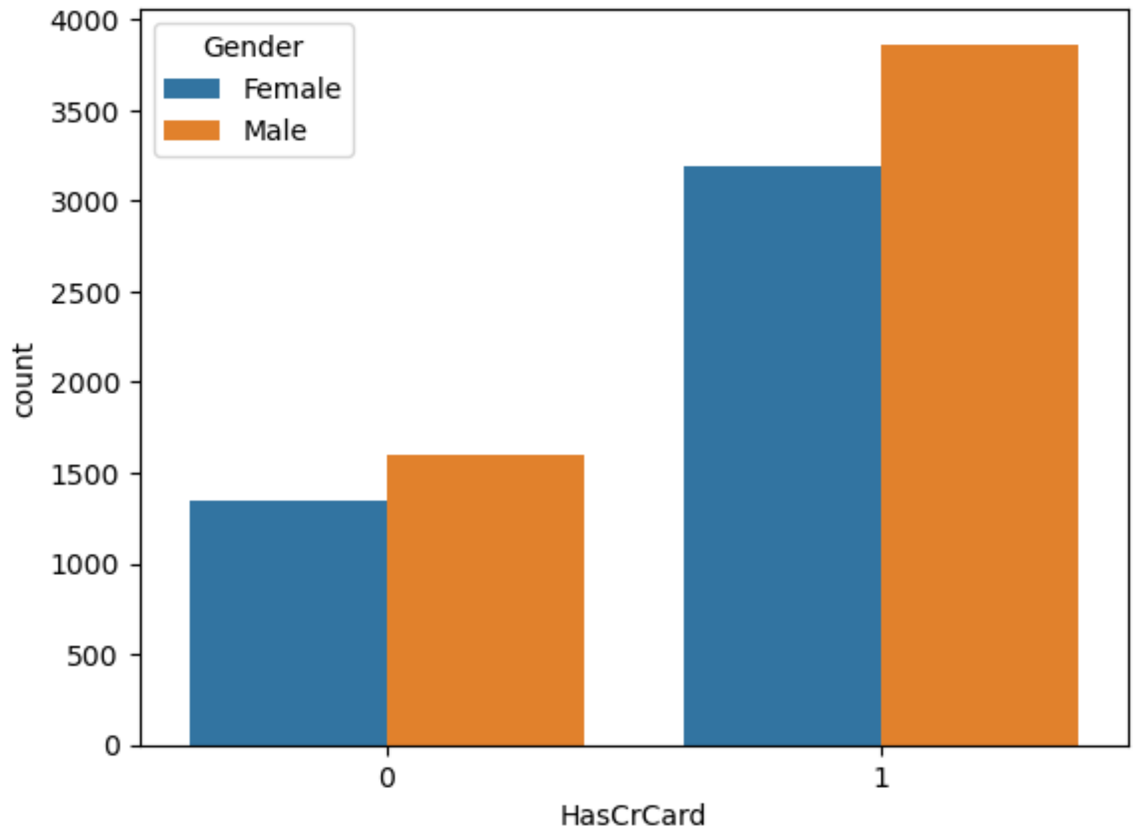
DATA ANALYSIS

In [6]:
```
# plotting between having Creditcard Vs not having credit card Customers

sns.countplot(x='HasCrCard', data = bank_data )
plt.show()
```



In [7]:
```
# Male Vs Female customers
```

In [8]:
```
sns.countplot(x='HasCrCard', data = bank_data, hue='Gender')
plt.show()
```

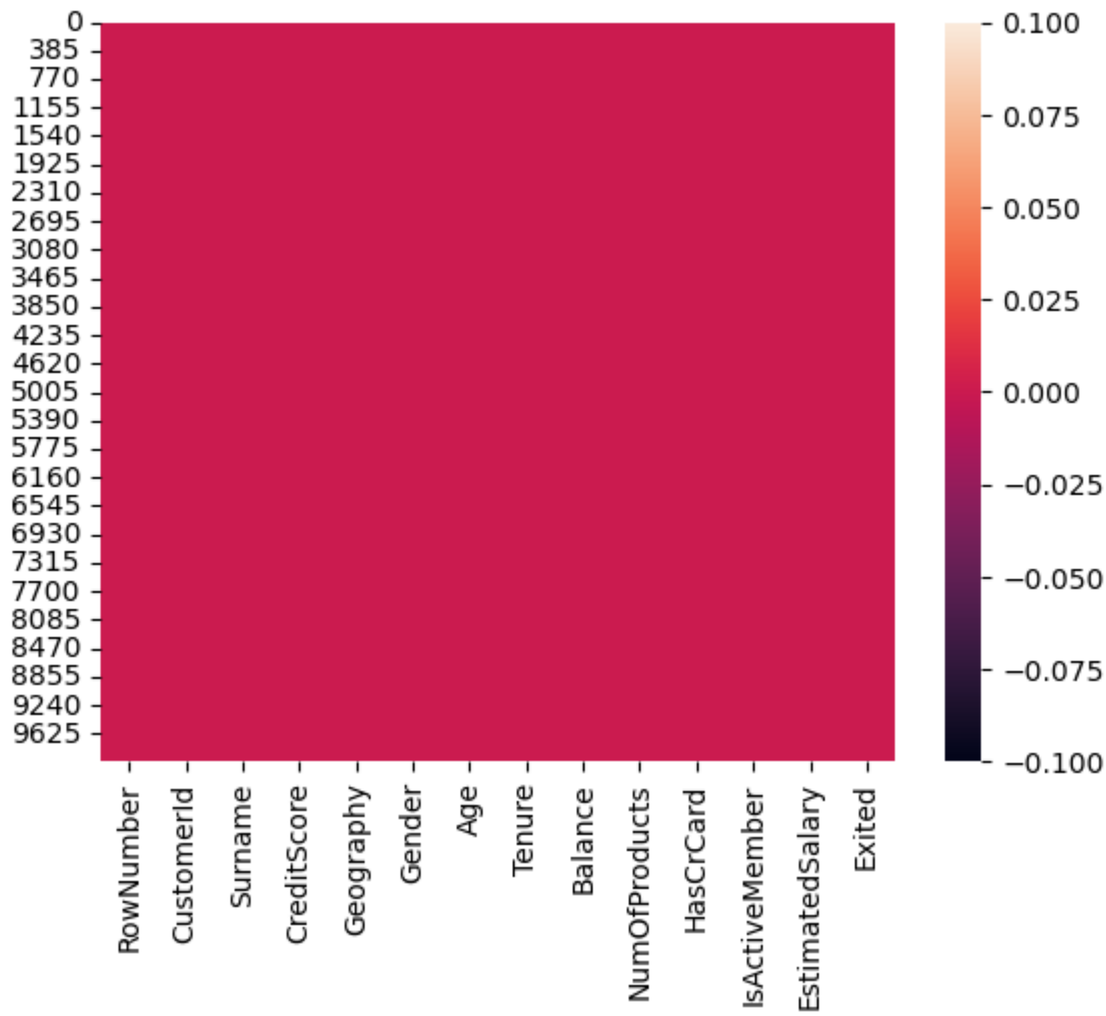


In [9]:
```
# checking the null values

bank_data.isnull().sum()
```

Out[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
```

In [10]:
```
#visualization for Null Values

sns.heatmap(bank_data.isnull())
```

Out[10]:
```
<Axes: >
```



In [11]:
```
# conclusion: there is no null values in our dataset
```

DATA CLEANING

In [12]:
```
# lets check for the non numerical columns
bank_data.dtypes
```

Out[12]:
```
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
```

In [13]:
```
# we can see Geography and Gender are non numerical columns.
# It seems that there is no need for Geography column for further predictions.
```

In [14]:
```
# Now the column Gender we are going to make non-numerical to numerical category
```

In [15]:
```
# lets convert the Gender column into numerical values.

gender = pd.get_dummies(bank_data['Gender'], drop_first=True)
bank_data['Gender'] = gender
```

In [16]:
```
bank_data.head()
```

Out[16]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balan
0	1	15634602	Hargrave	619	France	0	42	2	0.
1	2	15647311	Hill	608	Spain	0	41	1	83807.
2	3	15619304	Onio	502	France	0	42	8	159660.
3	4	15701354	Boni	699	France	0	39	1	0.
4	5	15737888	Mitchell	850	Spain	0	43	2	125510.

In [17]:
```
# lets drop some unwanted columns from our dataset.

bank_data=bank_data.drop(['Surname'],axis=1)
bank_data.head()
```

Out[17]:

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

In [18]:
```
# separate the dependent variable and non dependent variable
```

In [19]:
```
x=bank_data[['RowNumber','CustomerId','CreditScore','Gender','Age','Tenure','Bal
             'NumOfProducts','HasCrCard','IsActiveMember','EstimatedSalary']]
y=bank_data[['Exited']]
```

DATA MODELLING

building model using logistic regression

In [20]:
```
# importing Train test function

from sklearn.model_selection import train_test_split
```

In [30]:
```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, train_size=0.8)
```

In [31]:
```
# importing Logistic regression

from sklearn.linear_model import LogisticRegression
```

In [32]:
```
model=LogisticRegression()
```

In [33]:
```
model.fit(xtrain, ytrain)
```

```
C:\Users\LENOVO\anaconda3\lib\site-packages\sklearn\utils\validation.py:1143: Da
taConversionWarning: 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[33]:
```
▾ LogisticRegression

LogisticRegression()
```

In [34]:
```
# lets predict

predict = model.predict(xtest)
```

In [35]:
```
predict
```

Out[35]:
```
array([0, 0, 0, ..., 0, 0, 0], dtype=int64)
```

In [ ]:
```
# TESTING
# To see how our model is performing
```

In [51]:
```
from sklearn.metrics import classification_report
```

In [52]:
```
print(classification_report(ytest, predict))
```

```
              precision    recall  f1-score   support

           0       0.78      1.00      0.88      1562
           1       0.00      0.00      0.00       438

    accuracy                           0.78      2000
   macro avg       0.39      0.50      0.44      2000
weighted avg       0.61      0.78      0.68      2000

C:\Users\LENOVO\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1
344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to con
trol this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\LENOVO\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1
344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to con
trol this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\LENOVO\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1
344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to con
trol this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [ ]:
```
# thanks....
```