# Churn Prediction Using Classification Model

In [1]:
```
# importing libraries
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from scipy.stats import pearsonr, chi2_contingency
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.feature_selection import RFE
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
%matplotlib inline
```

In [2]:
```
df = pd.read_csv('CustomerChurn.csv') # read_data
df.head()
```

Out[2]:

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

In [3]:
```
df.dtypes
```

Out[3]:
```
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
Complain               int64
Satisfaction Score     int64
Card Type             object
Point Earned           int64
dtype: object
```

In [4]:
```
# display any null values
df.isnull().sum()
```

```
RowNumber            0
```

Out[4]:
```
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
Complain              0
Satisfaction Score    0
Card Type             0
Point Earned          0
dtype: int64
```

In [5]:
```python
df.describe(include='all')
```

Out[5]:

| | RowNumber | CustomerId | Surname | CreditScore | Geography | Gender | Age | Tenure | |
|---|---|---|---|---|---|---|---|---|---|
| **count** | 10000.00000 | 1.000000e+04 | 10000 | 10000.000000 | 10000 | 10000 | 10000.000000 | 10000.000000 | 1000 |
| **unique** | NaN | NaN | 2932 | NaN | 3 | 2 | NaN | NaN | |
| **top** | NaN | NaN | Smith | NaN | France | Male | NaN | NaN | |
| **freq** | NaN | NaN | 32 | NaN | 5014 | 5457 | NaN | NaN | |
| **mean** | 5000.50000 | 1.569094e+07 | NaN | 650.528800 | NaN | NaN | 38.921800 | 5.012800 | 7648 |
| **std** | 2886.89568 | 7.193619e+04 | NaN | 96.653299 | NaN | NaN | 10.487806 | 2.892174 | 6239 |
| **min** | 1.00000 | 1.556570e+07 | NaN | 350.000000 | NaN | NaN | 18.000000 | 0.000000 | |
| **25%** | 2500.75000 | 1.562853e+07 | NaN | 584.000000 | NaN | NaN | 32.000000 | 3.000000 | |
| **50%** | 5000.50000 | 1.569074e+07 | NaN | 652.000000 | NaN | NaN | 37.000000 | 5.000000 | 9719 |
| **75%** | 7500.25000 | 1.575323e+07 | NaN | 718.000000 | NaN | NaN | 44.000000 | 7.000000 | 12764 |
| **max** | 10000.00000 | 1.581569e+07 | NaN | 850.000000 | NaN | NaN | 92.000000 | 10.000000 | 25089 |

# Data Cleaning

Change float/int data types to object for categorical variables

In [6]:
```python
df = df.astype({'Satisfaction Score':object,'Exited':object,'IsActiveMember':object,
                'HasCrCard':object,'CustomerId':object})
```

In [7]:
```python
df.columns
```

Out[7]:
```
Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
       'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
       'IsActiveMember', 'EstimatedSalary', 'Exited', 'Complain',
       'Satisfaction Score', 'Card Type', 'Point Earned'],
      dtype='object')
```

In [8]:
```python
# drop row number column
```

```
df.drop('RowNumber',axis=1,inplace=True)
```

In [9]:

```
print('Number of customers that have exited/churned:', df[df['Exited']==1]['Exited'].count
```

```
Number of customers that have exited/churned: 2038
```

# Data exploration

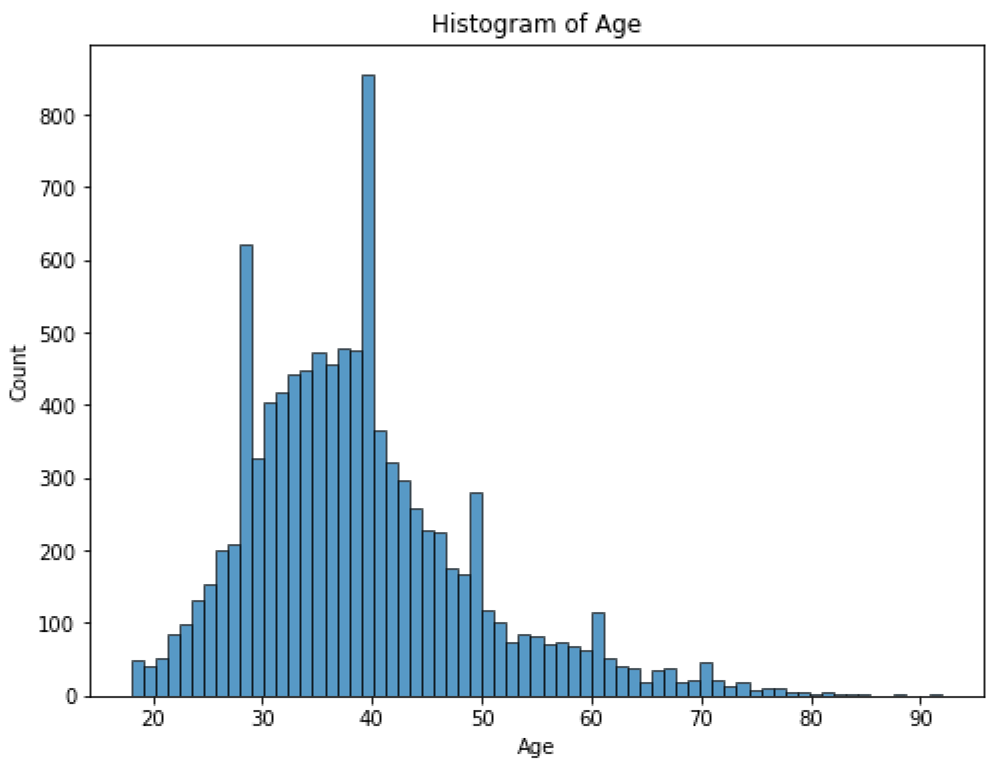
In [10]:

```
# set default graph size
matplotlib.rcParams['figure.figsize'] = (8, 6)
```

## Demographic Information

In [11]:

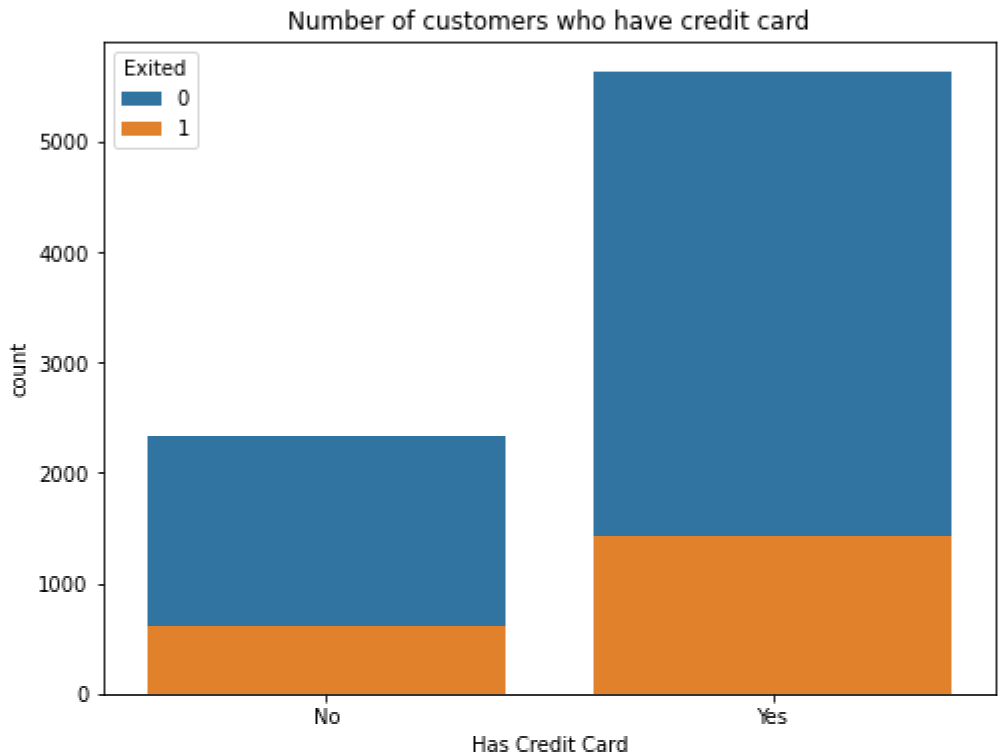
```
ax = sns.histplot(x='Age', data=df)
ax.set(title='Histogram of Age')
plt.show()
```




The distribution of age looks right skewed, indicating that is not normally distributed. We could probably fix this by transforming the value using `log` to follow normal distribution

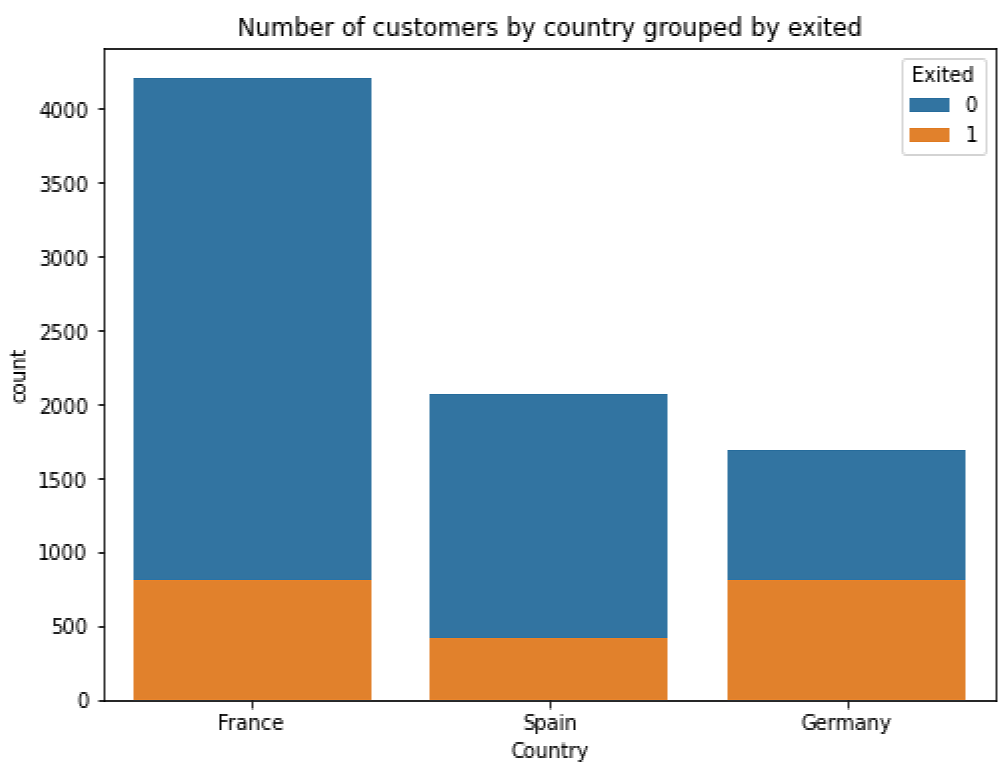
In [12]:

```
ax=sns.countplot(x=df['HasCrCard'], hue=df['Exited'], dodge=False)
ax.set(title='Number of customers who have credit card',xlabel='Has Credit Card',xticklabe
plt.show()
```



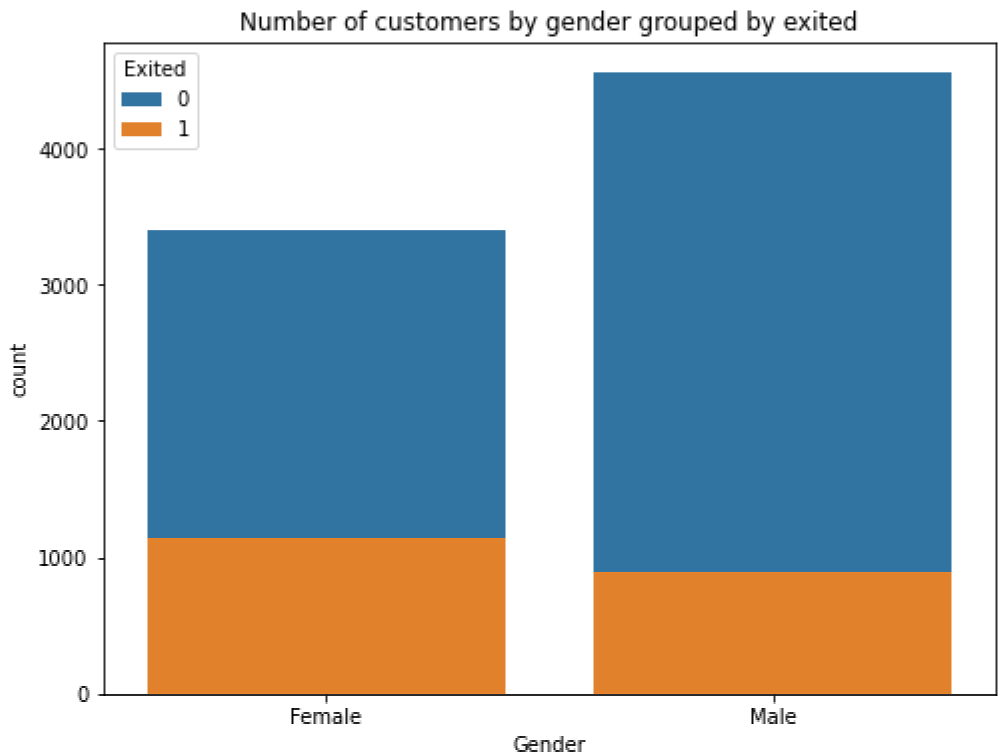

The number of customers who have credit card are significantly lower. However, the number of customers that churn are mostly the one who have credit card.

In [13]:
```
ax=sns.countplot(x=df['Geography'], hue=df['Exited'], dodge=False)
ax.set(title="Number of customers by country grouped by exited",xlabel='Country')
plt.show()
```




Most of the customers are based on France with Spain in 2nd and Germany in 3rd. It is interesting that most of the customers that churned are from Germany and the number reaches around 50% of the population.

In [14]:
```
ax=sns.countplot(x=df['Gender'], hue=df['Exited'], dodge=False)
ax.set(title="Number of customers by gender grouped by exited")
plt.show()
```



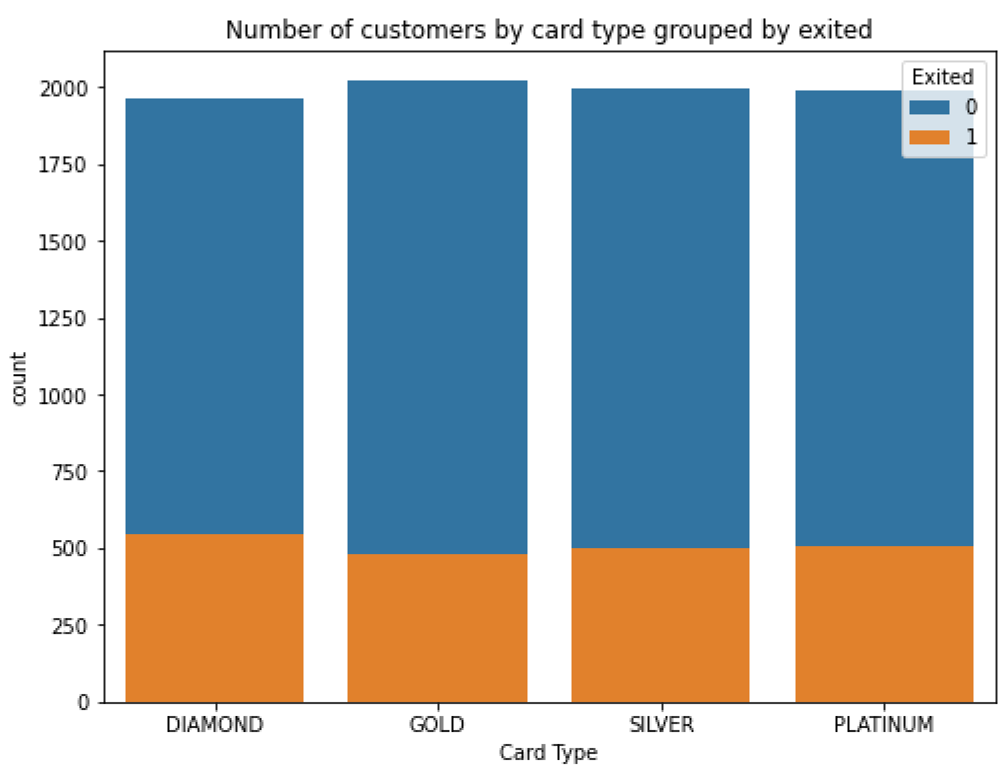

From the plot it can be seen that there are more male customers than female. However, it can be seen that there are slightly more customers who churned.

From the demographics exploration, we can say that the features `Gender` , `Geographic` , and `HasCrCard` are interesting and should be further explored.

# Customer Account Information

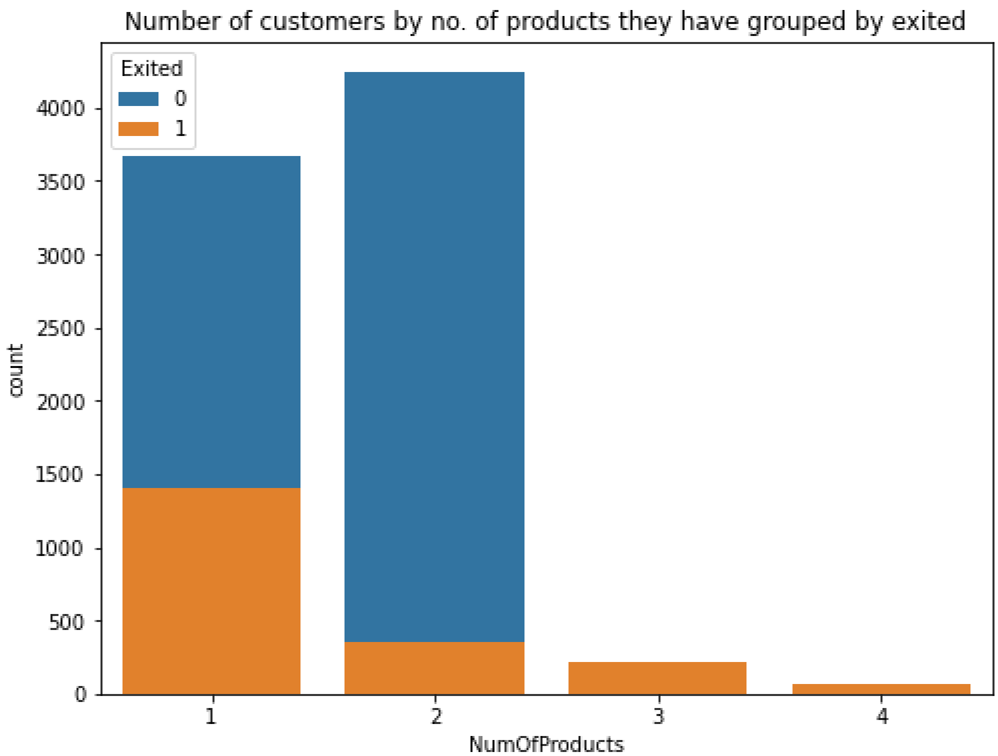
In [15]:

```
ax = sns.countplot(x=df['Card Type'], hue=df['Exited'], dodge=False)
ax.set(title="Number of customers by card type grouped by exited")
plt.show()
```



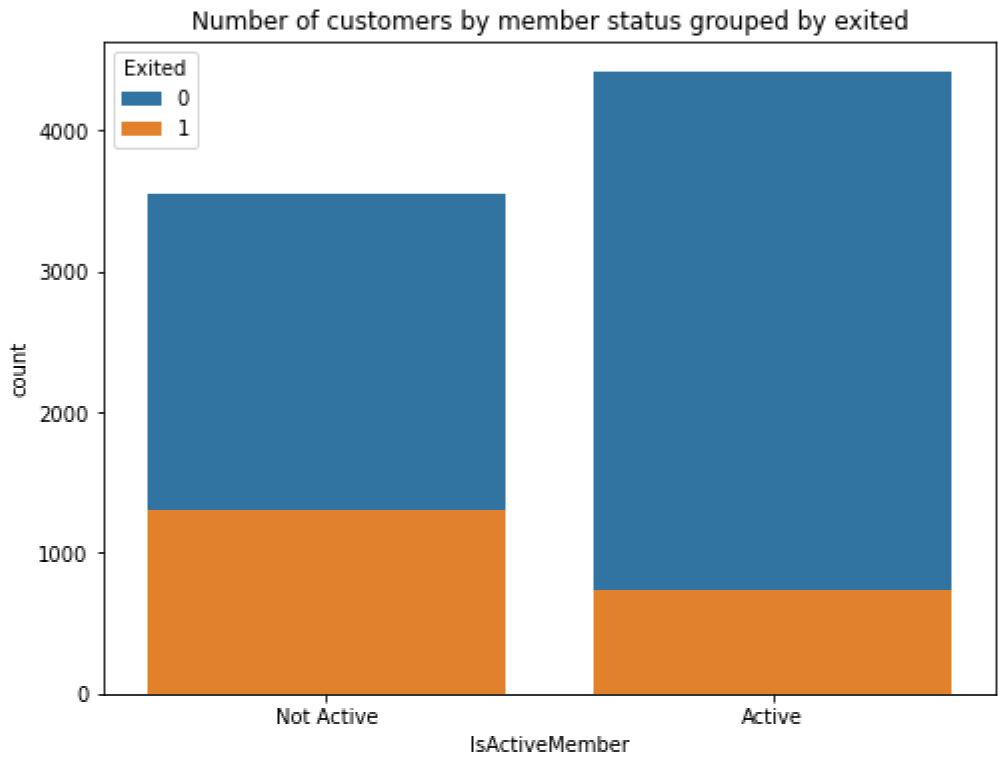
From the plot it can be seen that the number of customers that hold certain type of card are equal even after being grouped by churn status. We can consider that card type does not have influence on the churn status

In [16]:
```
ax = sns.countplot(x=df['NumOfProducts'], hue=df['Exited'], dodge=False)
ax.set(title="Number of customers by no. of products they have grouped by exited")
plt.show()
```



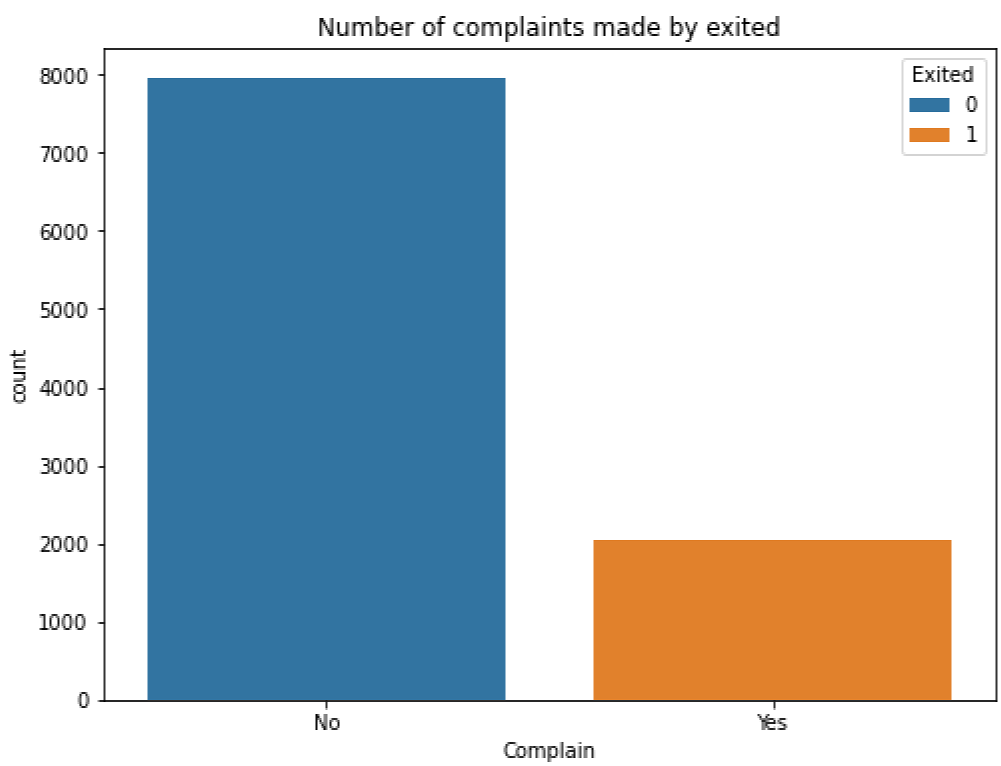

Customers that churned tend to have only 1 product with the bank. There is a small percentage of the population that have 3 or 4 products with the bank and all have churned. Customers that did not churned mostly have 2 products.

In [17]:
```
ax = sns.countplot(x=df['IsActiveMember'], hue=df['Exited'], dodge=False)
ax.set(title="Number of customers by member status grouped by exited", xticklabels=['Not A
plt.show()
```



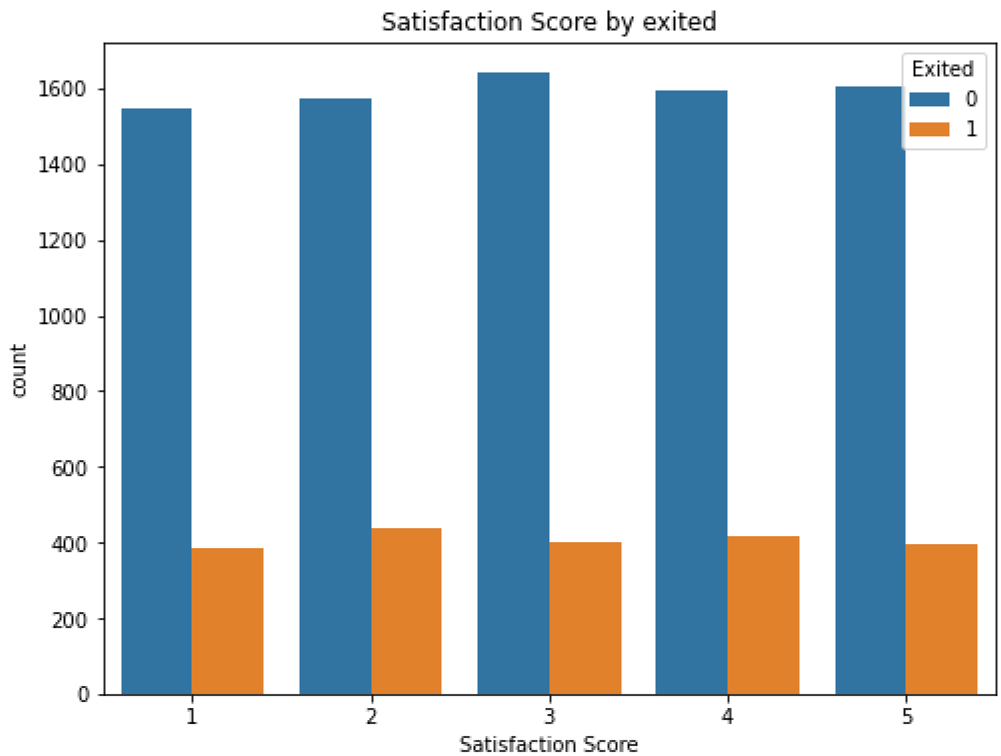

There are slightly more active members with the bank. Churned customers tend to not have a member with the bank.

In [18]:
```
ax = sns.countplot(x=df['Complain'], hue=df['Exited'], dodge=False)
ax.set(title="Number of complaints made by exited", xticklabels=['No','Yes'])
plt.show()
```




From the plot it can be seen that almost all of the customers that have lodged a complain has churned. There is a bias towards customers that do complaint will churn.

In [19]:
```
ax = sns.countplot(data=df,x='Satisfaction Score',hue='Exited')
ax.set(title='Satisfaction Score by exited')
plt.show()
```




The number of score given by customers seem to be equal among all scores from 1 to 5.

In [20]:

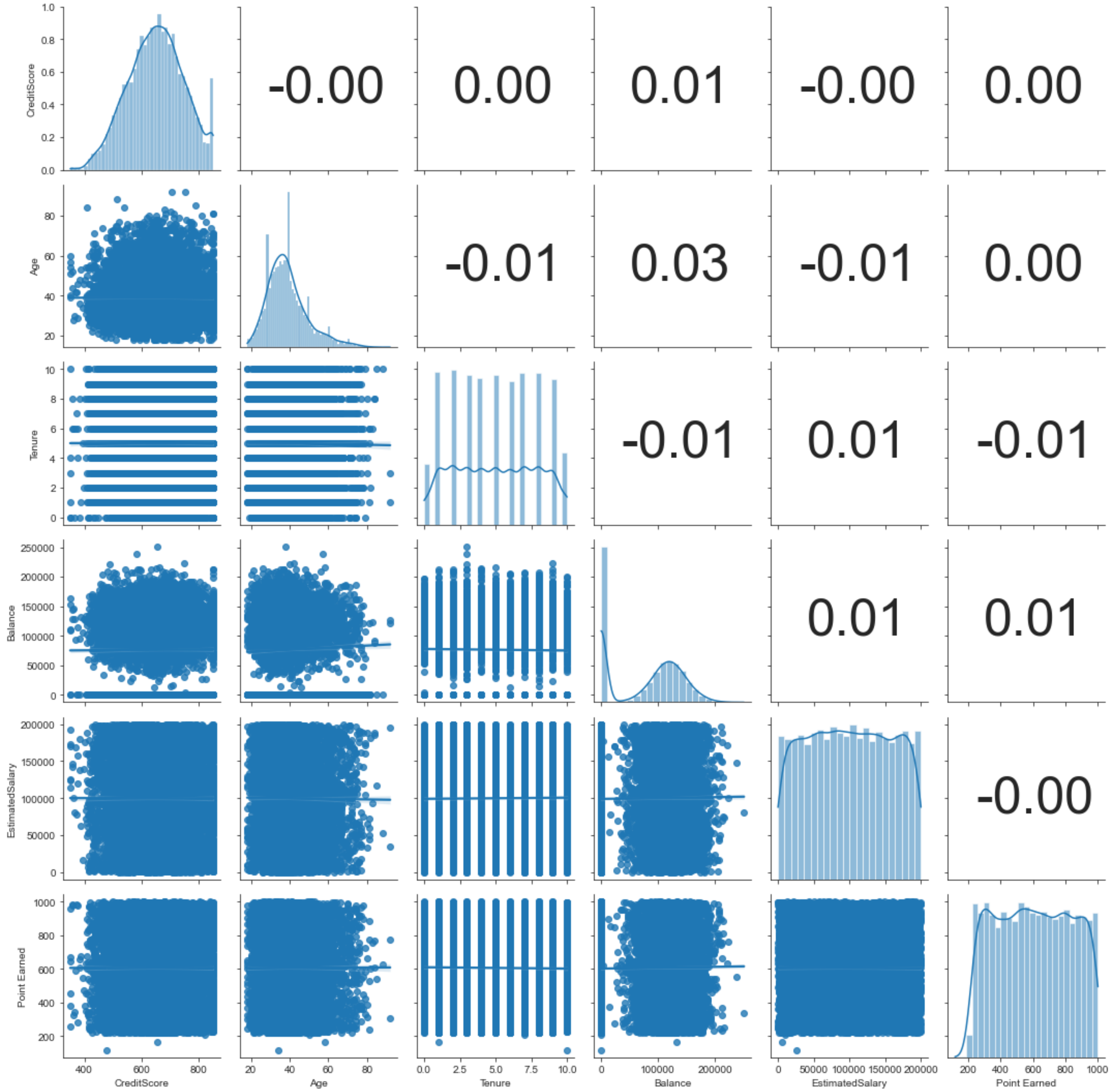
```
sns.set_style('ticks')
variables_for_pair = ['CreditScore','Age','Tenure','Balance','EstimatedSalary',
                      'Point Earned']
pairplot_df = df[variables_for_pair]

def rscore(x, y, **kws):
    r, _ = pearsonr(x, y)
    ax = plt.gca()
    ax.annotate("{:.2f}".format(r), xy = (0.55, 0.5), size=50, xycoords = ax.transAxes, va

pplot = sns.PairGrid(pairplot_df,diag_sharey=False)
pplot.map_lower(sns.regplot)
pplot.map_diag(sns.histplot, kde=True)
pplot.map_upper(rscore)
```

Out[20]:

```
<seaborn.axisgrid.PairGrid at 0x1fb9840b490>
```



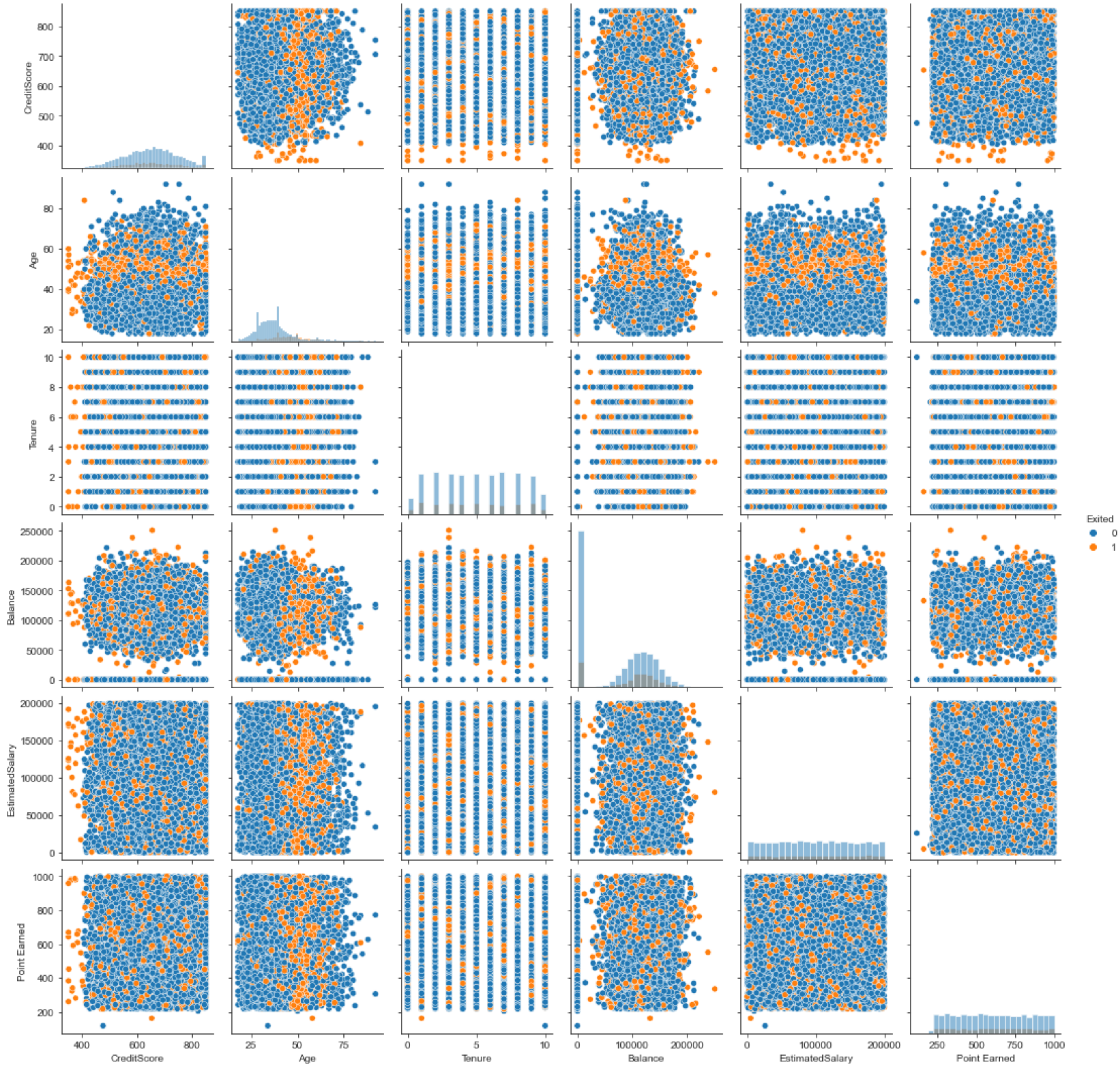

None of the variables `'CreditScore','Age','Tenure','Balance','EstimatedSalary','Point Earned'` are related to each other indicating there is no relationship exist in the independent variables.

In [21]:

```
variables_for_pair = ['CreditScore','Age','Tenure','Balance','EstimatedSalary',
                      'Point Earned','Exited']
pairplot_df=df[variables_for_pair]
sns.pairplot(pairplot_df, hue='Exited',diag_kind='hist')
```

Out[21]:

```
<seaborn.axisgrid.PairGrid at 0x1fb981dc0d0>
```

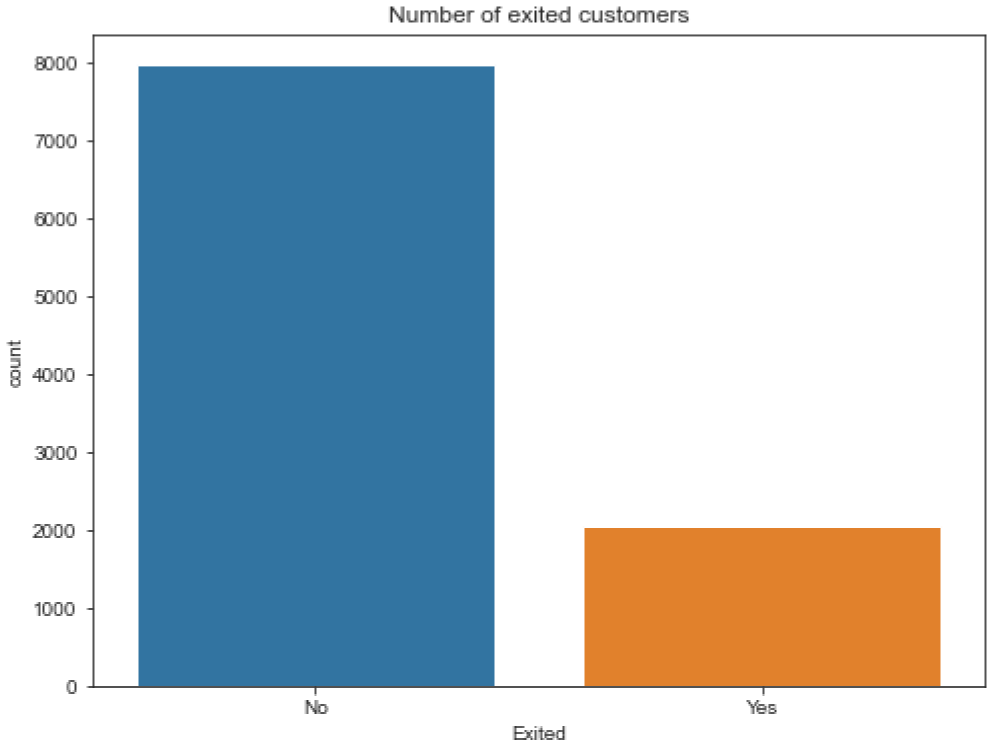



From the pairplot, it seems that the data is unbalanced i.e. there is more customers that did not churned (0) than did (1).

We have seen the relationship/correlation between variables earlier. In the next step, we need to look at association between variables.

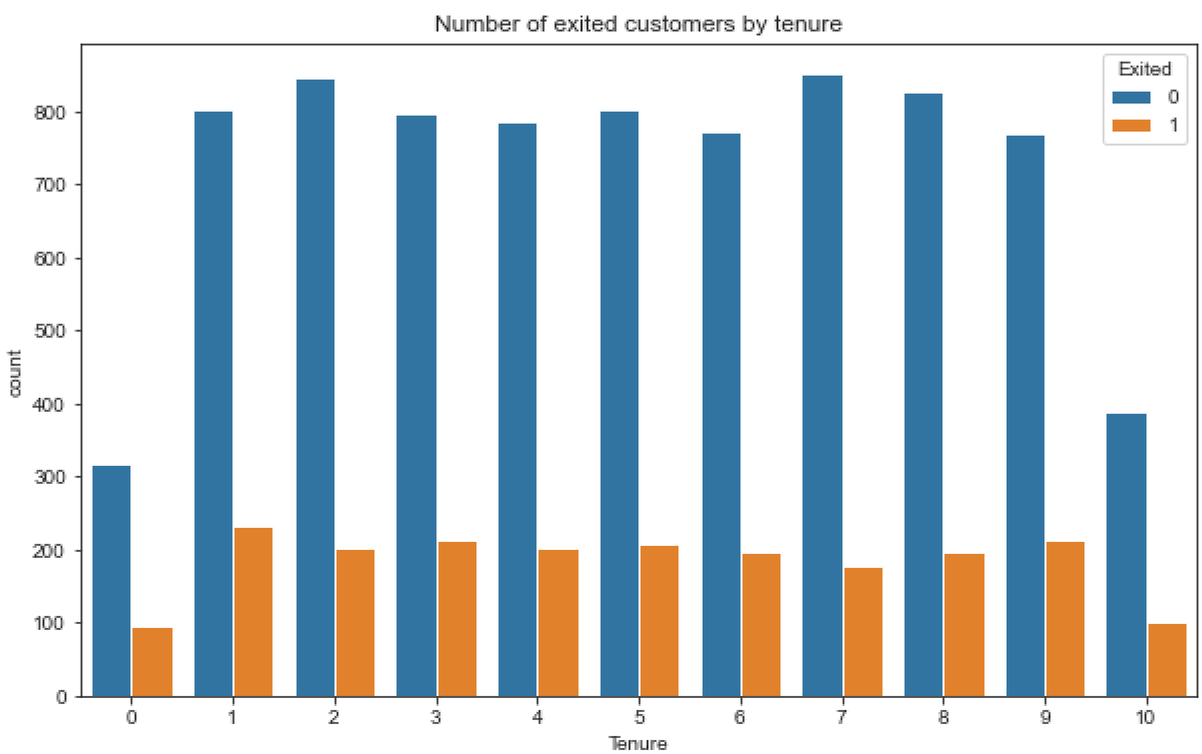
In [22]:

```
ax = sns.countplot(x=df["Exited"])
ax.set(title='Number of exited customers')
ax.set_xticklabels(['No','Yes'])
plt.show()
```




The data is indeed unbalanced with more customers that did **not** exited/churned.

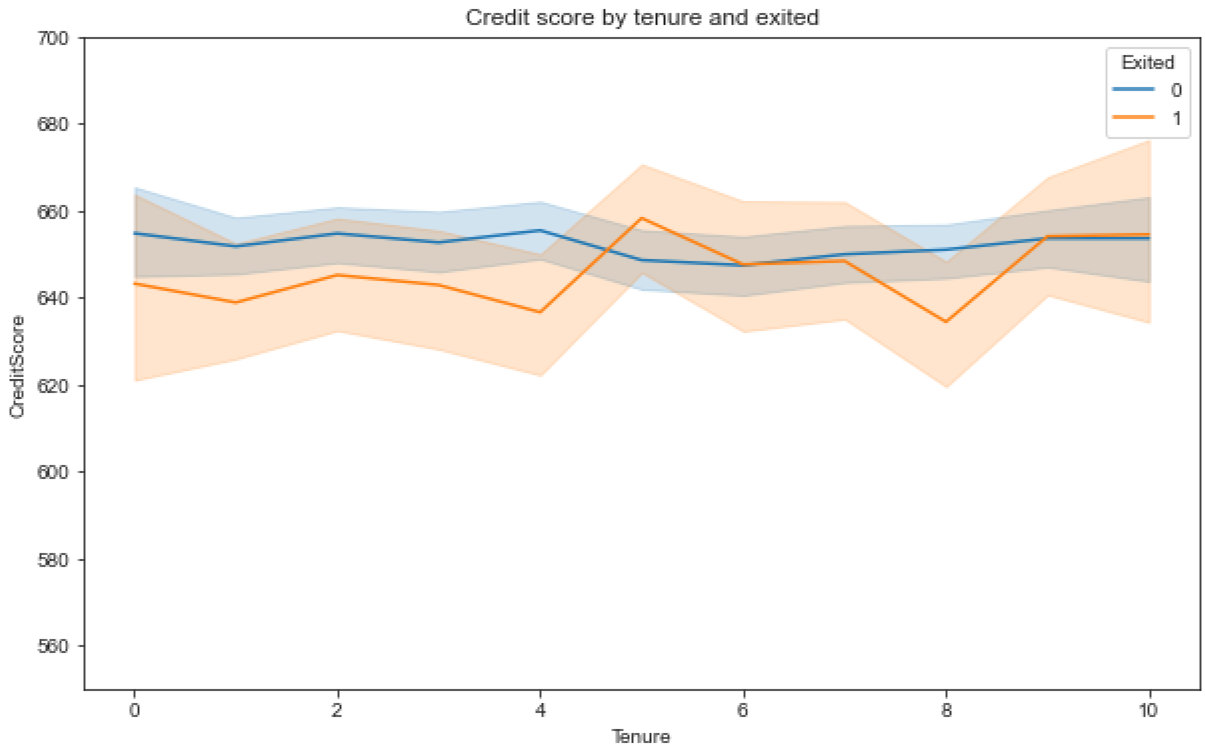
In [23]:
```
plt.figure(figsize=(10,6))
ax = sns.countplot(x=df['Tenure'], hue=df['Exited'])
ax.set(title='Number of exited customers by tenure')
plt.show()
```




We could not see any significant evidence that tenure influenced churned customers. We could probably see that most customers churned after 1 year with the bank

In [24]:
```
plt.figure(figsize=(10,6))
ax = sns.lineplot(x=df['Tenure'],y=df['CreditScore'],hue=df['Exited'])
ax.set(title='Credit score by tenure and exited')
```

```
ax.set_ylim(550,700)
plt.show()
```




Although the customers that did not churned tend to have slighlty higher credit score by tenure, in comparison between churn and not churn, they are in similar range.

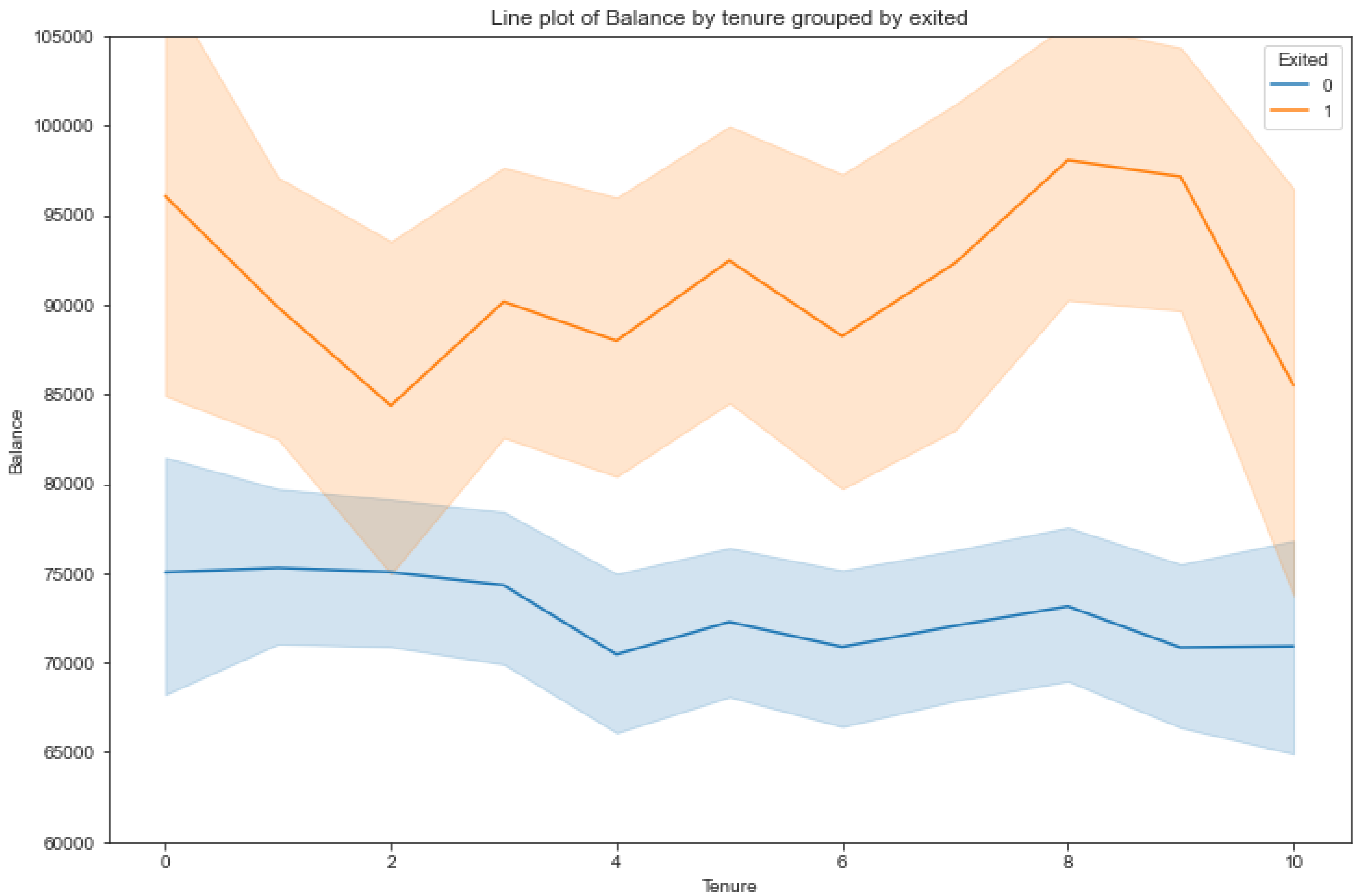
In [25]:

```
ax=sns.violinplot(y=df['CreditScore'], x=df['Exited'])
ax.set(title='Violin plot for Credit Score by Exited')
plt.show()
```


From the plot, we can see that both class have a similar distribution, customers that have exited do have slightly lower credit score distribution.

In [26]:

```
plt.figure(figsize=(12,8))
ax=sns.lineplot(x=df['Tenure'],y=df['Balance'],hue=df['Exited'])
ax.set(title='Line plot of Balance by tenure grouped by exited')
ax.set_ylim(60000,105000)
plt.show()
```




The plot indicates that balance isolates the number of customers that churned and did not churned by tenure. Customers that have higher balance tend to churn more than who have lower balance.

# Data Preparation

## One-hot encoding

The features `'Gender'`, `'Geographic'`, `'HasCrCard'` needs to be encoded as they currently contain 2 or more values.

In [27]:

```
def encode_concat(data, column):
    new_df = pd.get_dummies(data[column])
    df = pd.concat([data,new_df],axis=1)
    df.drop(column, axis=1,inplace=True)
    return df

prepDf = encode_concat(df,'Gender')
prepDf = encode_concat(prepDf, 'Geography')
prepDf.head()
```

Out[27]:

| | CustomerId | Surname | CreditScore | Age | Tenure | Balance | NumOfProducts | HasCrCard | IsActiveMember | Estima |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 15634602 | Hargrave | 619 | 42 | 2 | 0.00 | 1 | 1 | 1 | |

| | CustomerId | Surname | CreditScore | Age | Tenure | Balance | NumOfProducts | HasCrCard | IsActiveMember | Estima |
|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 15647311 | Hill | 608 | 41 | 1 | 83807.86 | 1 | 0 | 1 | |
| **2** | 15619304 | Onio | 502 | 42 | 8 | 159660.80 | 3 | 1 | 0 | |
| **3** | 15701354 | Boni | 699 | 39 | 1 | 0.00 | 2 | 0 | 0 | |
| **4** | 15737888 | Mitchell | 850 | 43 | 2 | 125510.82 | 1 | 1 | 1 | |

In [30]:
```
prepDf.to_csv('PrepChurnData.csv', index=False) # Export data into csv file
```

In [ ]:
```

```