Python初級數據分析員證書

（六）數據分析 及可視化專案

13. 數據分析專案 Demo 15 – Bank Customer Churn

Review

- Statistics
- Hypothesis testing
- Algebra
- Linear regression
- Propositional logic
- Python
- R
- SQL

- Pandas, NumPy, SciPy
- Data Visualization, Matplotlib, Seaborn, Plotly
- Dashboard Visualization, Business Intelligence
- Storytelling

Chapter Summary

- Scenario
- Data Import
- EDA
- Label encorder
- Df.corrwith()
- Randomforestclassifier

Scenario

In this **Bank Customer Churn**(流失) dataset, we are going to analyse customer churn and derive insights to help the bank improve customer retention strategies.

Customer churn refers to the phenomenon where customers discontinue their relationship with a company or stop using its products or services.

Understanding the factors that contribute to churn is crucial for businesses to retain customers and maintain long-term profitability.

Data import

```
1  import pandas as pd
2  import numpy as np
3  import matplotlib.pyplot as plt
4  import seaborn as sns
5
6  from sklearn.ensemble import RandomForestClassifier
7  from sklearn.model_selection import train_test_split
8  from sklearn.preprocessing import OrdinalEncoder
9  from sklearn.metrics import accuracy_score
10 from sklearn.inspection import permutation_importance
```

```
1  data = pd.read_csv('CustomerRecords.csv')
2  data.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	H
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	

Overview all columns

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 18 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   RowNumber           10000 non-null  int64  
 1   CustomerId          10000 non-null  int64  
 2   Surname             10000 non-null  object 
 3   CreditScore         10000 non-null  int64  
 4   Geography           10000 non-null  object 
 5   Gender              10000 non-null  object 
 6   Age                 10000 non-null  int64  
 7   Tenure              10000 non-null  int64  
 8   Balance             10000 non-null  float64
 9   NumOfProducts       10000 non-null  int64  
 10  HasCrCard           10000 non-null  int64  
 11  IsActiveMember      10000 non-null  int64  
 12  EstimatedSalary     10000 non-null  float64
 13  Exited              10000 non-null  int64  
 14  Complain            10000 non-null  int64  
 15  Satisfaction Score  10000 non-null  int64  
 16  Card Type           10000 non-null  object 
 17  Point Earned        10000 non-null  int64  
dtypes: float64(2), int64(12), object(4)
memory usage: 1.4+ MB
```

Categoric Column

```
1 data.describe(include='O')
```

	Surname	Geography	Gender	Card Type
count	10000	10000	10000	10000
unique	2932	3	2	4
top	Smith	France	Male	DIAMOND
freq	32	5014	5457	2507

```
1 data['Geography'].unique()
```

```
array(['France', 'Spain', 'Germany'], dtype=object)
```

```
1 data['Card Type'].unique()
```

```
array(['DIAMOND', 'GOLD', 'SILVER', 'PLATINUM'], dtype=object)
```

Transpose first 3 rows of data

```
1 data.head(3).T
```

	0	1	2
RowNumber	1	2	3
CustomerId	15634602	15647311	15619304
Surname	Hargrave	Hill	Onio
CreditScore	619	608	502
Geography	France	Spain	France
Gender	Female	Female	Female
Age	42	41	42
Tenure	2	1	8
Balance	0.0	83807.86	159660.8
NumOfProducts	1	1	3
HasCrCard	1	0	1
IsActiveMember	1	1	0
EstimatedSalary	101348.88	112542.58	113931.57
Exited	1	0	1
Complain	1	1	1
Satisfaction Score	2	3	3
Card Type	DIAMOND	DIAMOND	DIAMOND
Point Earned	464	456	377

Check NaN and Null

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

```
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
Complain             0
Satisfaction Score   0
Card Type            0
Point Earned         0
dtype: int64
```

Column name definition

CreditScore—can have an effect on customer churn, since a customer with a higher credit score is less likely to leave the bank.

Geography—a customer's location can affect their decision to leave the bank.

Gender—it's interesting to explore whether gender plays a role in a customer leaving the bank.

Age—this is certainly relevant, since older customers are less likely to leave their bank than younger ones.

Tenure—refers to the number of years that the customer has been a client of the bank. Normally, older clients are more loyal and less likely to leave a bank.

Balance—also a very good indicator of customer churn, as people with a higher balance in their accounts are less likely to leave the bank compared to those with lower balances.

Column name definition

NumOfProducts—refers to the number of products that a customer has purchased through the bank.

HasCrCard—denotes whether or not a customer has a credit card. This column is also relevant, since people with a credit card are less likely to leave the bank.

IsActiveMember—active customers are less likely to leave the bank.

EstimatedSalary—as with balance, people with lower salaries are more likely to leave the bank compared to those with higher salaries.

Exited—**whether or not the customer left the bank**. (1: Exited , 0: Not exited)

Complain—customer has complaint or not.

Satisfaction Score—Score provided by the customer for their complaint resolution.

Card Type—type of card hold by the customer.

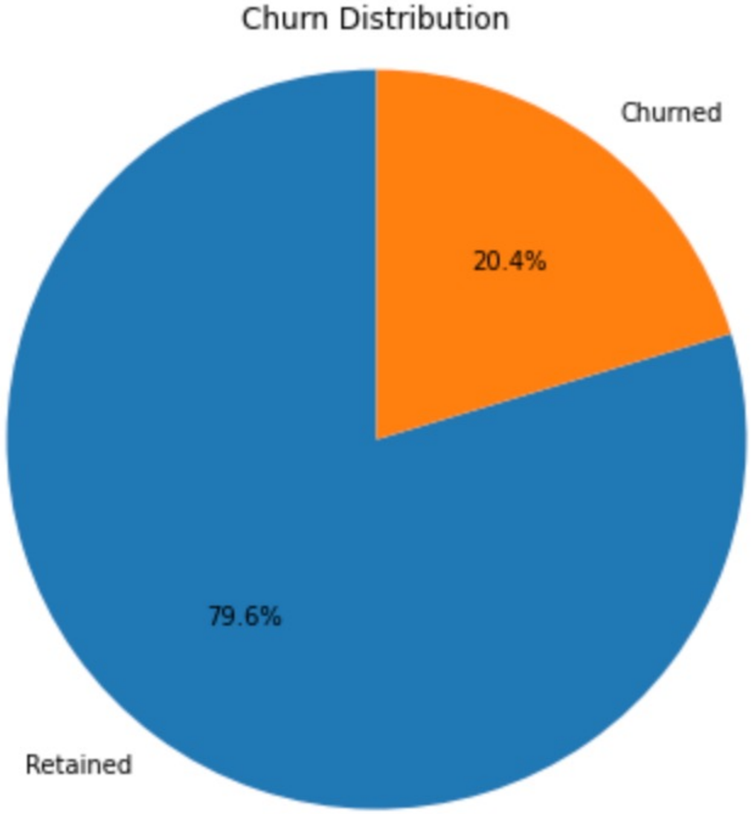
Points Earned—the points earned by the customer for using credit card.

Setting Target for modelling

The target variable is "Exited" which indicates whether a customer has churned (1) or not (0). By exploring the data and performing data analysis, feature engineering, and predictive modelling, we aim to identify patterns and factors associated with customer churn.

Exploratory Data Analysis (EDA)

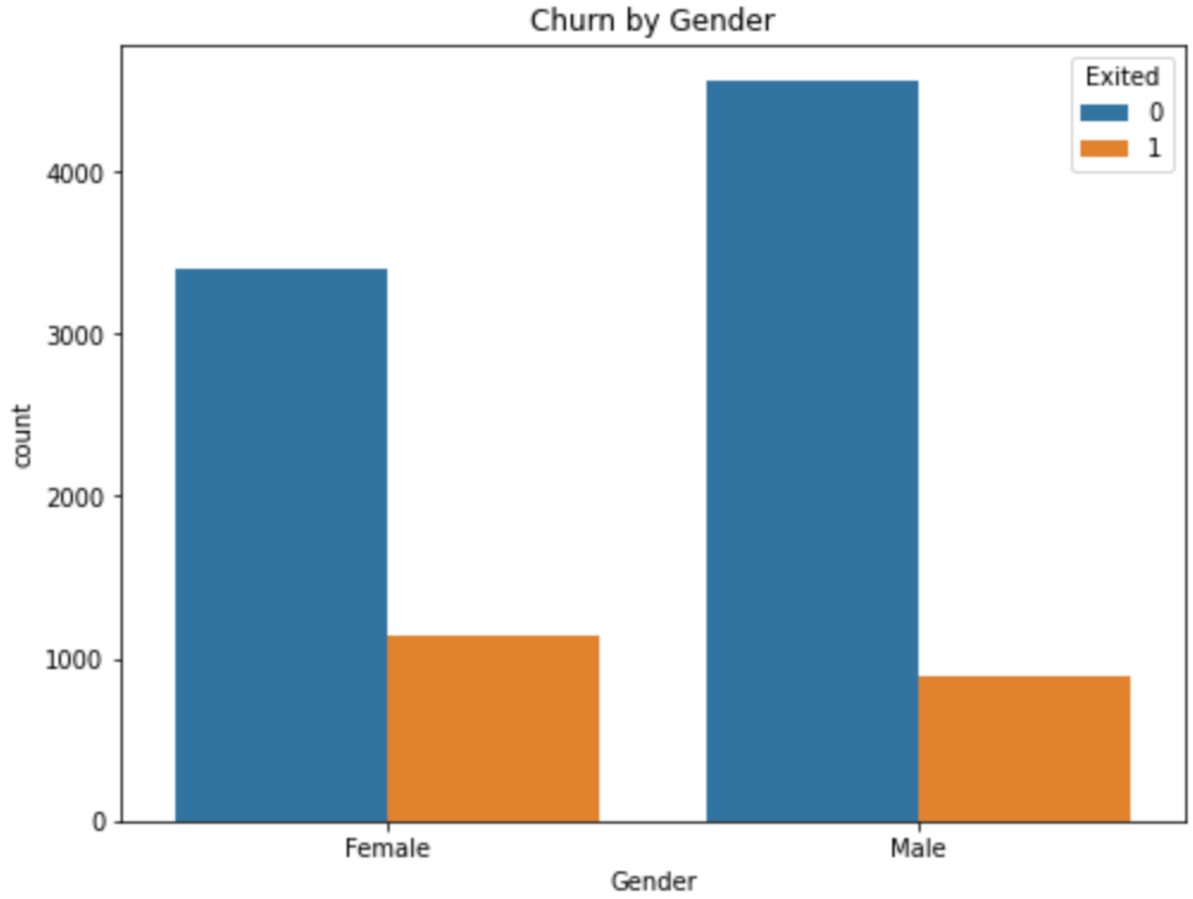
```
1 plt.figure(figsize=(6, 6))
2 churn_counts = data['Exited'].value_counts()
3 plt.pie(churn_counts, labels=['Retained', 'Churned'], autopct='%1.1f%%', startangle=90)
4 plt.axis('equal')
5 plt.title('Churn Distribution')
6 plt.show()
```



Churn distribution

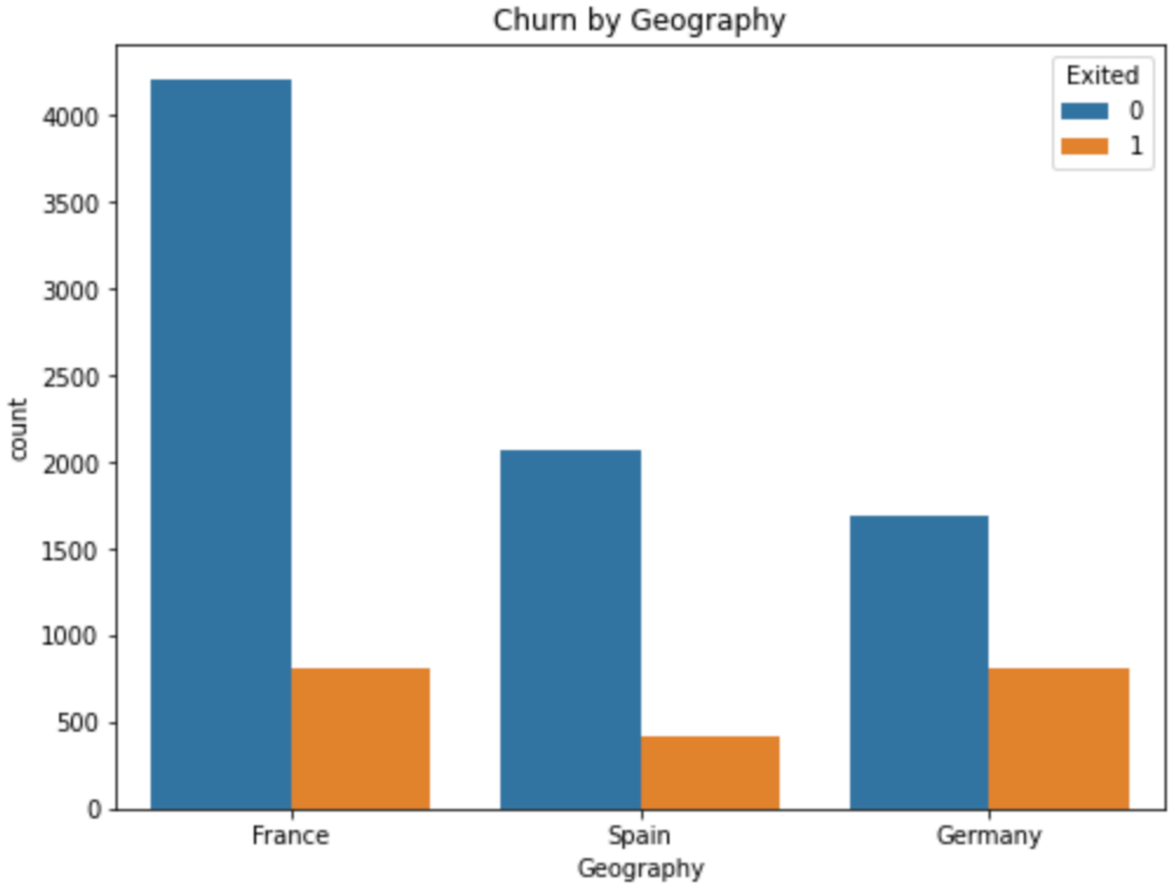
Churn by gender

```
1  plt.figure(figsize=(8, 6))
2  sns.countplot(x='Gender', hue='Exited', data=data)
3  plt.title('Churn by Gender')
4  plt.show()
```



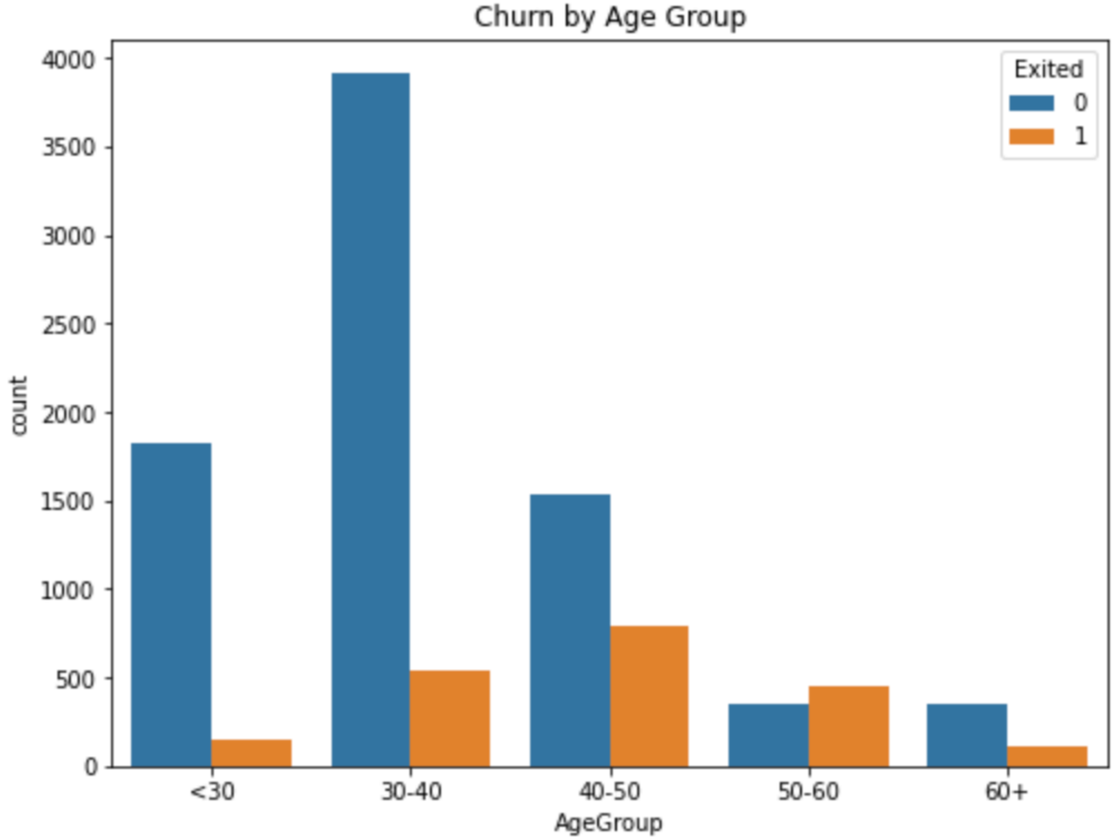
Churn by geography

```
1 plt.figure(figsize=(8, 6))
2 sns.countplot(x='Geography', hue='Exited', data=data)
3 plt.title('Churn by Geography')
4 plt.show()
```



Churn by age group

```
data['AgeGroup'] = pd.cut(data['Age'], bins=[0, 30, 40, 50, 60, np.inf],
                          labels=['<30', '30-40', '40-50', '50-60', '60+'])
plt.figure(figsize=(8, 6))
sns.countplot(x='AgeGroup', hue='Exited', data=data)
plt.title('Churn by Age Group')
plt.show()
```



Churn rate by gender

Churn rate by gender

```
1  churn_rate_gender = data.groupby('Gender')['Exited'].mean()
2  print(churn_rate_gender)
```

```
Gender
Female    0.250715
Male      0.164743
Name: Exited, dtype: float64
```

Churn rate by gender is as follows:

- **Female: 25.1%**
- **Male: 16.5%**

Female’s ratio is significantly higher.

Churn rate by geography

Churn rate by geography

```
1 churn_rate_geography = data.groupby('Geography')['Exited'].mean()
2 print(churn_rate_geography)
```

```
Geography
France      0.161747
Germany     0.324432
Spain       0.166734
Name: Exited, dtype: float64
```

Churn rate by geography is as follows:

- **France: 16.2%**
- **Germany: 32.4%**
- **Spain: 16.7%**

Germany’s churn rate is almost doubled the other two.

Churn rate by age group

Churn rate by age group

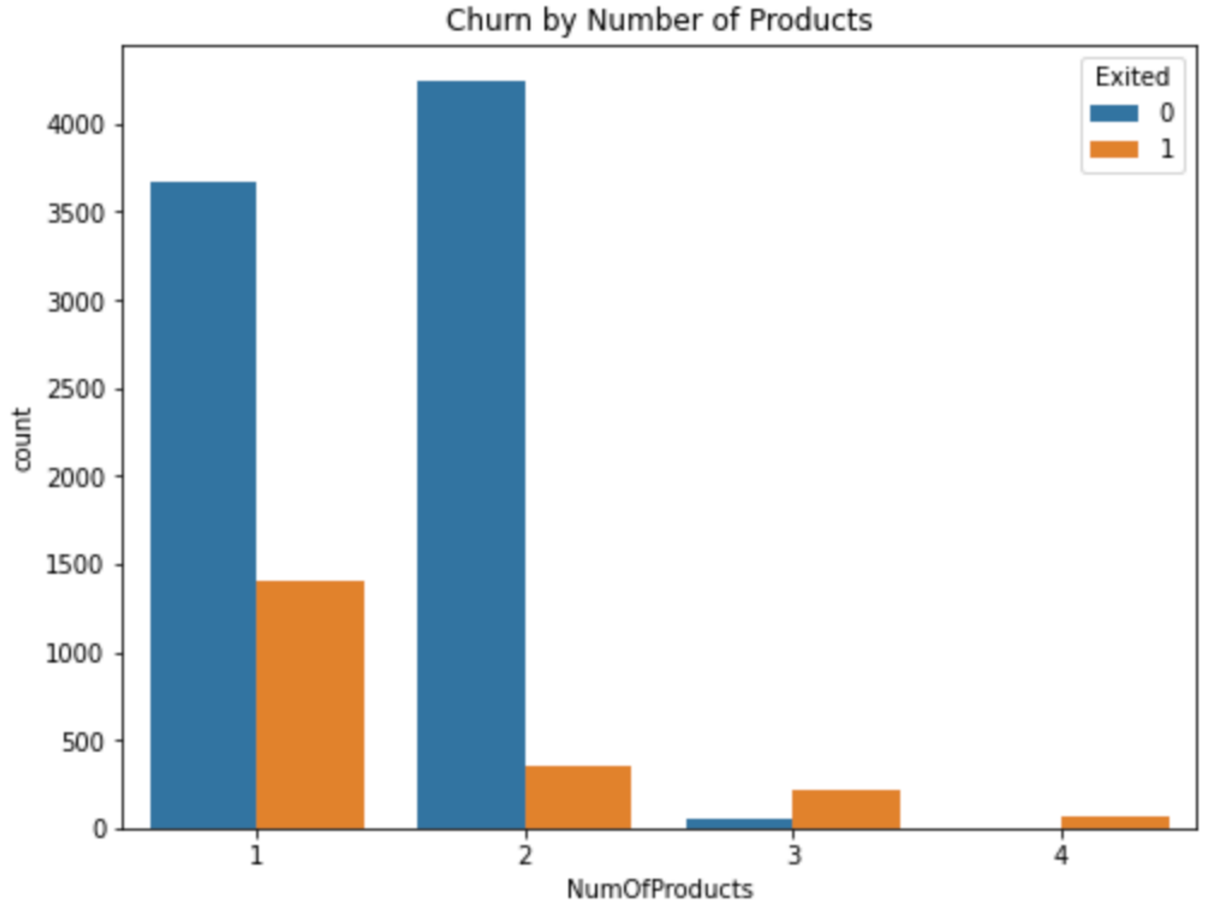
```
1  churn_rate_age = data.groupby('AgeGroup')['Exited'].mean()
2  print(churn_rate_age)
```

```
AgeGroup
<30       0.075203
30-40     0.121096
40-50     0.339655
50-60     0.562108
60+       0.247845
Name: Exited, dtype: float64
```

These churn rates indicate that the highest churn rate is observed in the 50-60 age group (56.2%), followed by the 40-50 age group (33.9%). The lowest churn rate is observed in the <30 age group (7.5%).

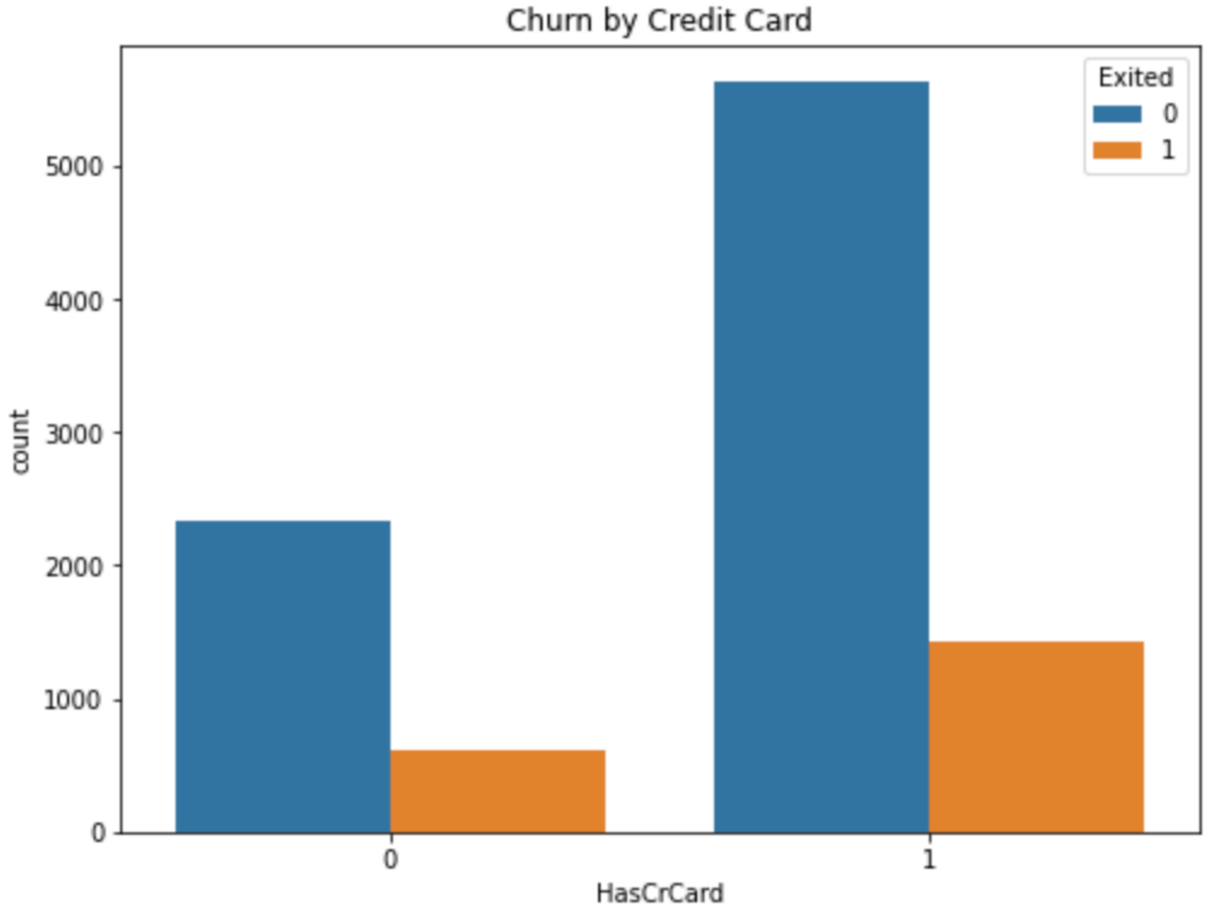
Churn by Number of Products

```
1  plt.figure(figsize=(8, 6))
2  sns.countplot(x='NumOfProducts', hue='Exited', data=data)
3  plt.title('Churn by Number of Products')
4  plt.show()
```



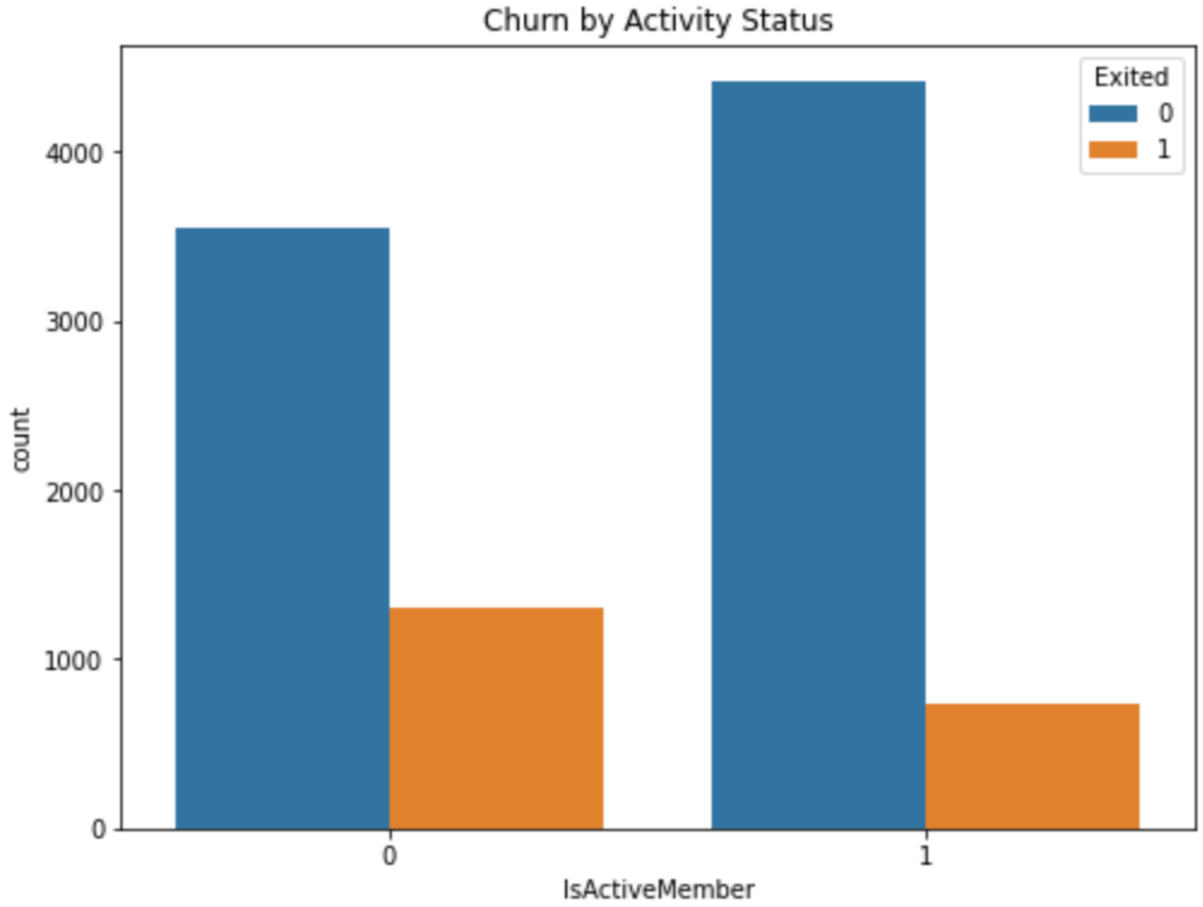
Churn by Credit Card

```
1 plt.figure(figsize=(8, 6))
2 sns.countplot(x='HasCrCard', hue='Exited', data=data)
3 plt.title('Churn by Credit Card')
4 plt.show()
```



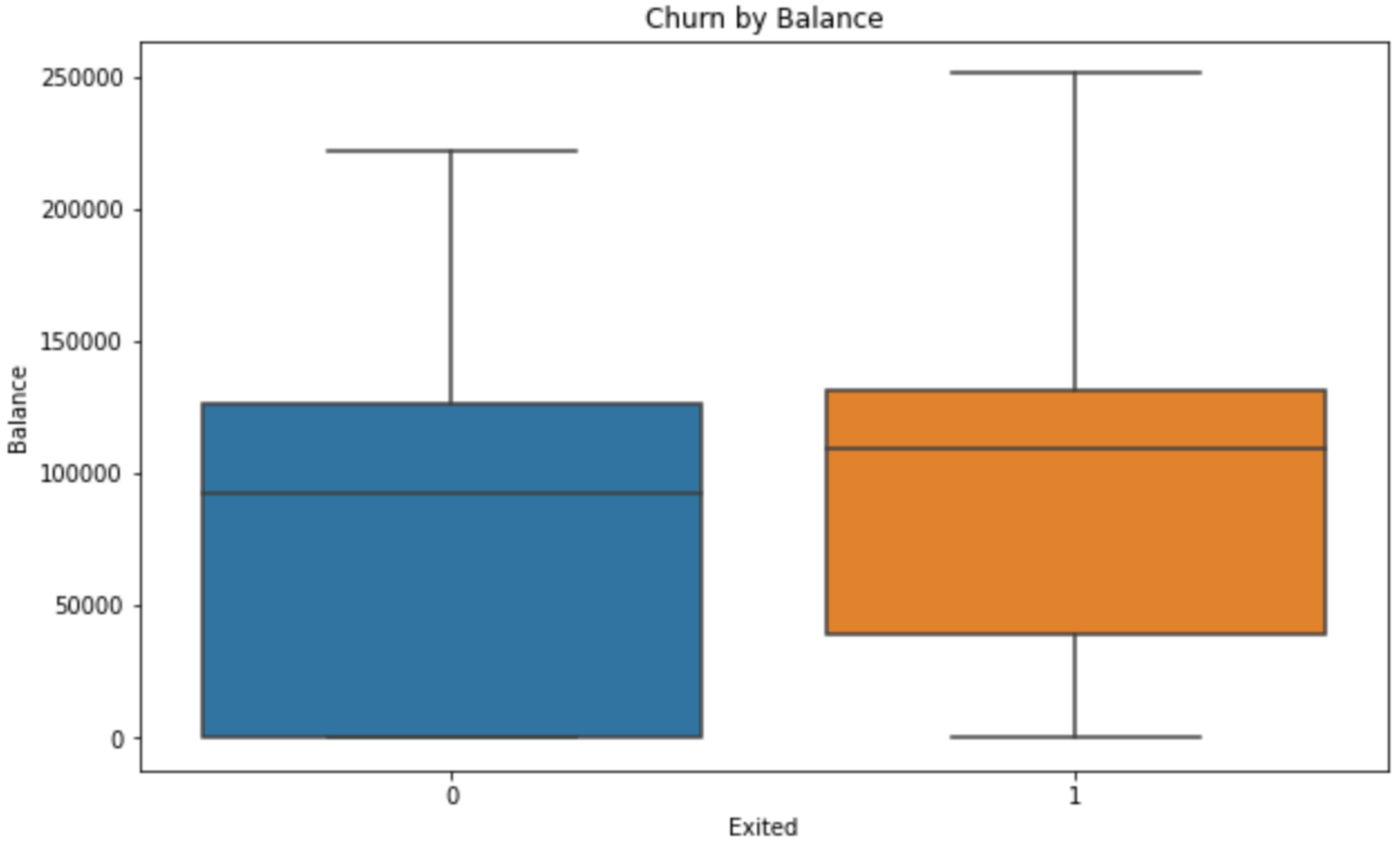
Churn by Activity Status

```
1 plt.figure(figsize=(8, 6))
2 sns.countplot(x='IsActiveMember', hue='Exited', data=data)
3 plt.title('Churn by Activity Status')
4 plt.show()
```



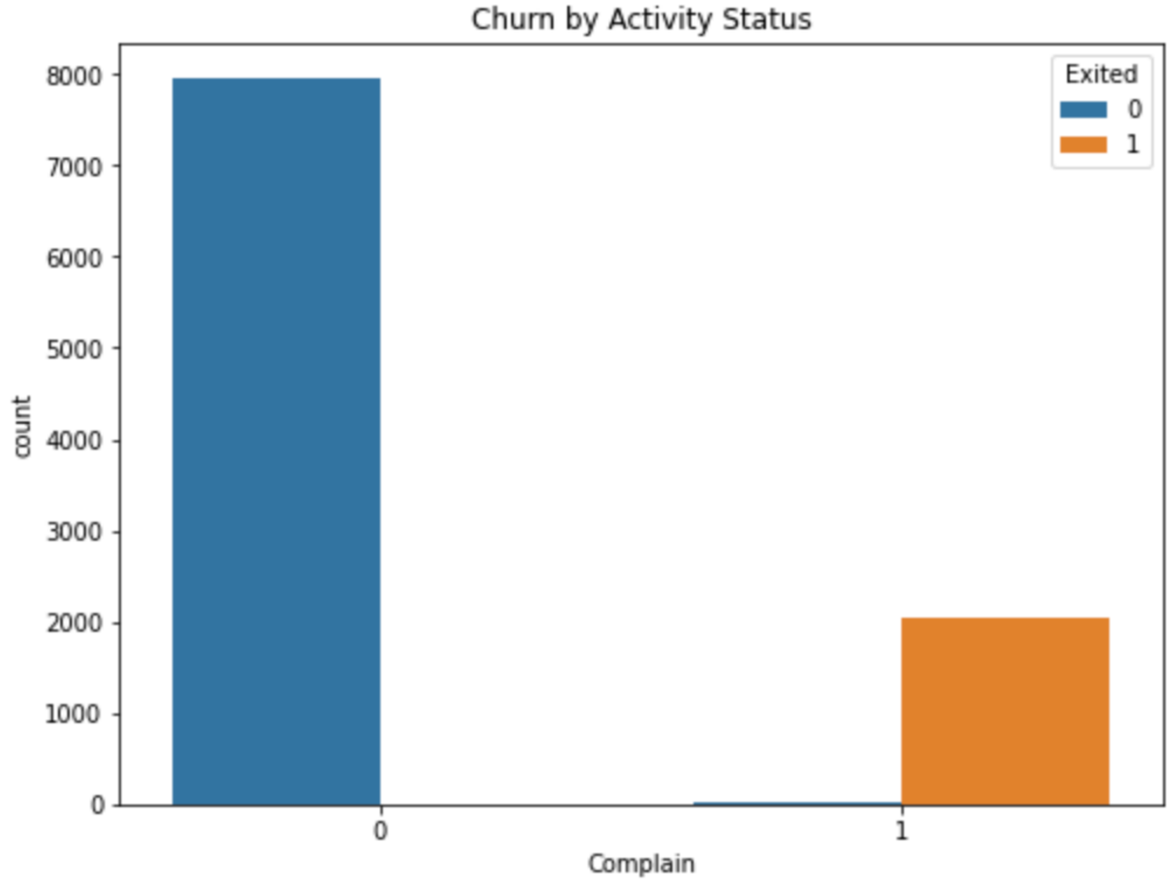
Churn by Balance

```
1  plt.figure(figsize=(10, 6))
2  sns.boxplot(x='Exited', y='Balance', data=data)
3  plt.title('Churn by Balance')
4  plt.show()
```



Churn by Complain

```
plt.figure(figsize=(8, 6))
sns.countplot(x='Complain', hue='Exited', data=data)
plt.title('Churn by Complain')
plt.show()
```



This is an important finding. In the exit portion, the complain number is significantly high.

Create Model for Prediction

Prepare column to model

```
1  # Drop unnecessarry column
2  data.drop(columns=["RowNumber", "CustomerId", "Surname"], inplace=True)
```

```
1  data.head(3)
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMer
0	619	France	Female	42	2	0.00	1	1	
1	608	Spain	Female	41	1	83807.86	1	0	
2	502	France	Female	42	8	159660.80	3	1	

Categorical Encoding

```
1 cat_cols = ["Geography", "Gender", "Card Type","AgeGroup"]
2 enc = OrdinalEncoder()
3 data[cat_cols] = enc.fit_transform(data[cat_cols])
```

```
1 data[cat_cols].nunique()
```

```
Geography     3
Gender        2
Card Type     4
AgeGroup      5
dtype: int64
```

```
1 data[cat_cols].sample(3)
```

	Geography	Gender	Card Type	AgeGroup
8761	2.0	1.0	0.0	3.0
7605	2.0	1.0	3.0	0.0
7134	0.0	1.0	1.0	1.0

Categorise the object columns into number.

Categorical Encoding

```
1 data[cat_cols].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Geography   10000 non-null  float64
 1   Gender      10000 non-null  float64
 2   Card Type   10000 non-null  float64
 3   AgeGroup    10000 non-null  float64
dtypes: float64(4)
memory usage: 312.6 KB
```

Split Train & Test

```
X_train, X_test, y_train, y_test = train_test_split(data.drop("Exited", axis=1),
                                                    data["Exited"], test_size=0.20, random_state=42)
```

Train and test model

```
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)
print("Accuracy on test data: {:.2f}".format(clf.score(X_test, y_test)))
```

```
Accuracy on test data: 1.00
```

Correlation of features and target

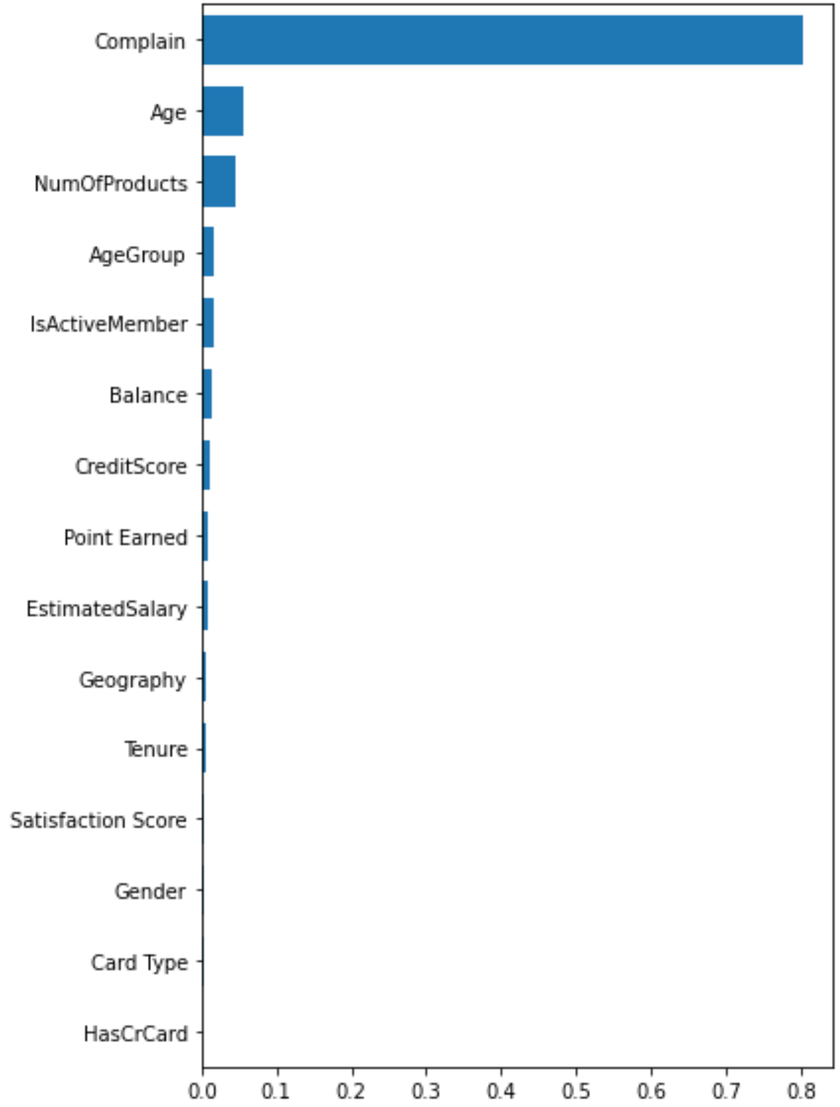
```
1 data.corrwith(data['Exited']).sort_values(ascending=False)
```

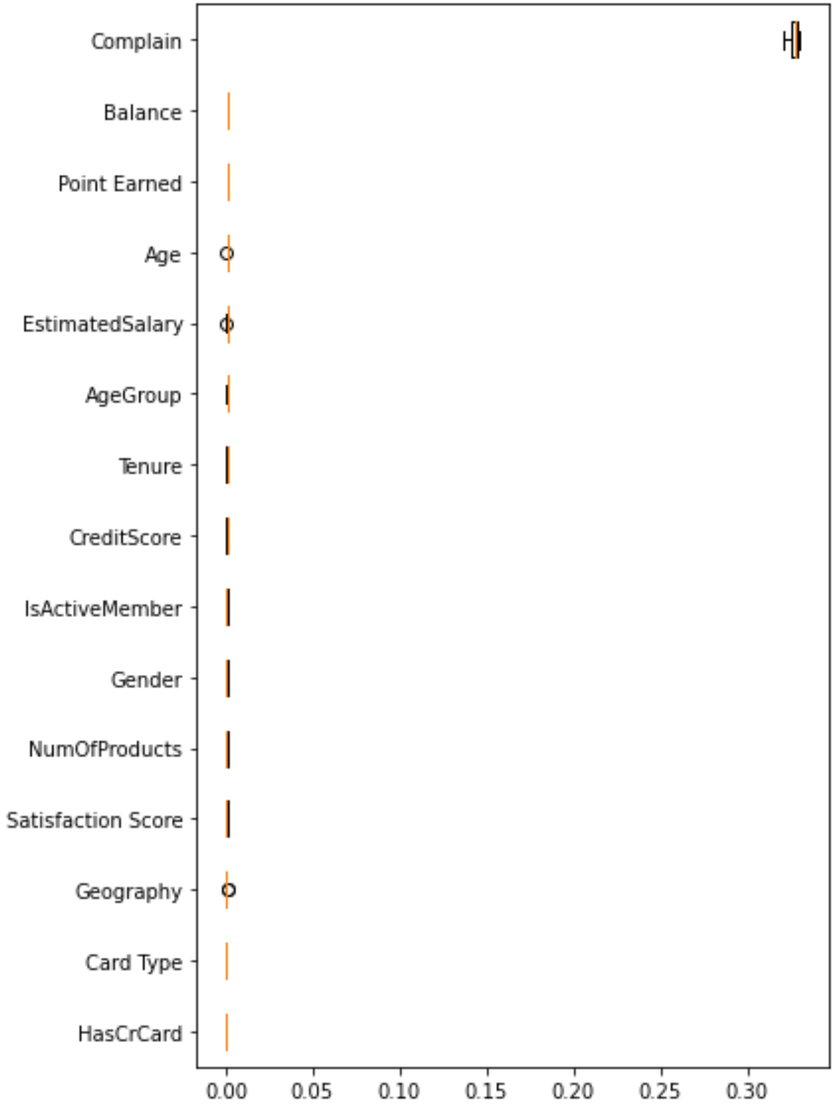
```
Exited                1.000000
Complain              0.995693
Age                   0.285296
Balance               0.118577
Geography             0.035712
EstimatedSalary       0.012490
Point Earned         -0.004628
Satisfaction Score   -0.005849
HasCrCard            -0.006976
AgeGroup             -0.010406
Card Type            -0.010861
Tenure               -0.013656
CreditScore          -0.026771
NumOfProducts        -0.047611
Gender               -0.106267
IsActiveMember       -0.156356
dtype: float64
```

Justify visual with model result

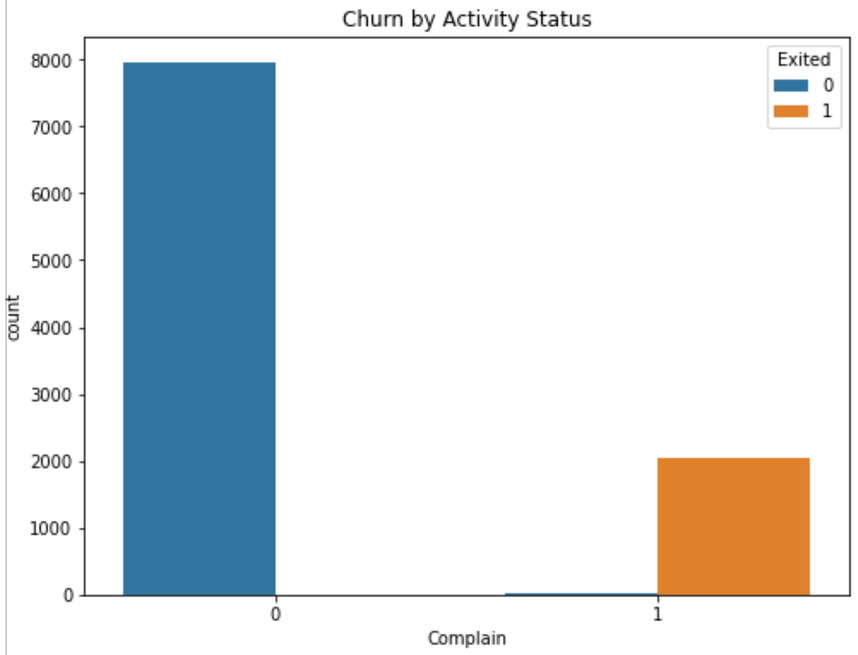
```
result = permutation_importance(clf, X_train, y_train, n_repeats=10, random_state=42)
perm_sorted_idx = result.importances_mean.argsort()

tree_importance_sorted_idx = np.argsort(clf.feature_importances_)
tree_indices = np.arange(0, len(clf.feature_importances_)) + 0.5

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 8))
ax1.barh(tree_indices, clf.feature_importances_[tree_importance_sorted_idx], height=0.7)
ax1.set_yticks(tree_indices)
ax1.set_yticklabels(X_train.columns[tree_importance_sorted_idx])
ax1.set_ylim((0, len(clf.feature_importances_)))
ax2.boxplot(
    result.importances[perm_sorted_idx].T,
    vert=False,
    labels=X_train.columns[perm_sorted_idx],
)
fig.tight_layout()
plt.show()
```

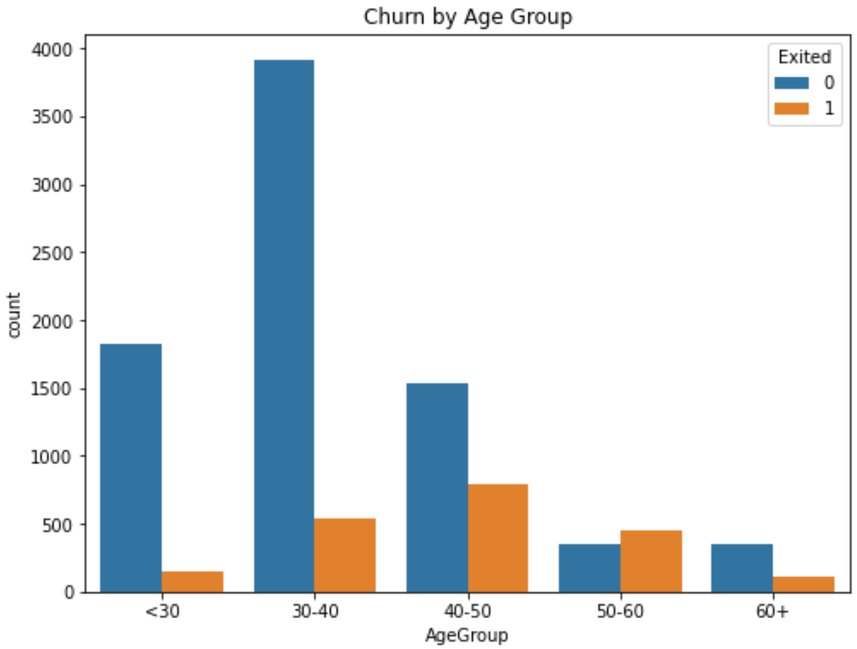
Justify visual with model result

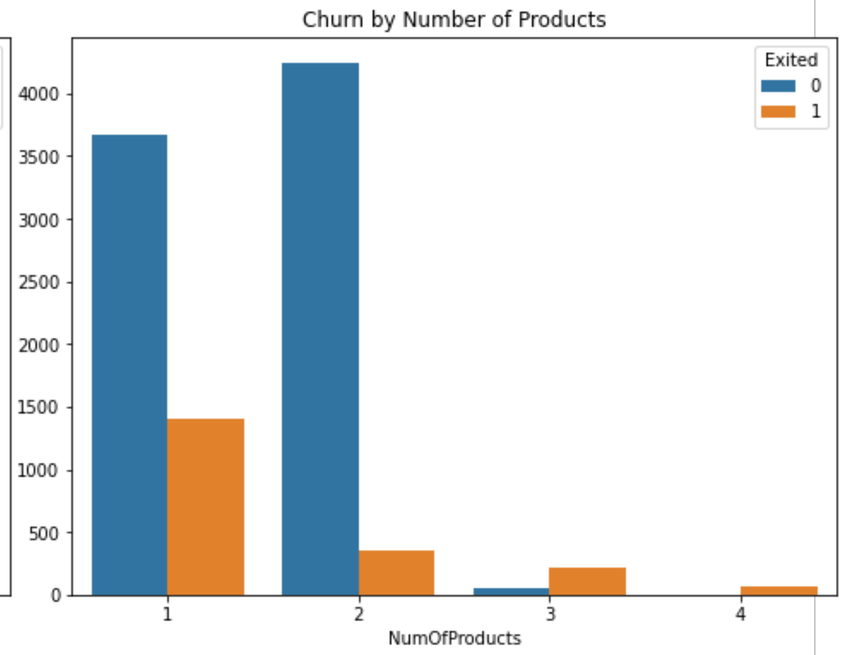




13.數據分析專案 Data Analysis Project – Demo 15







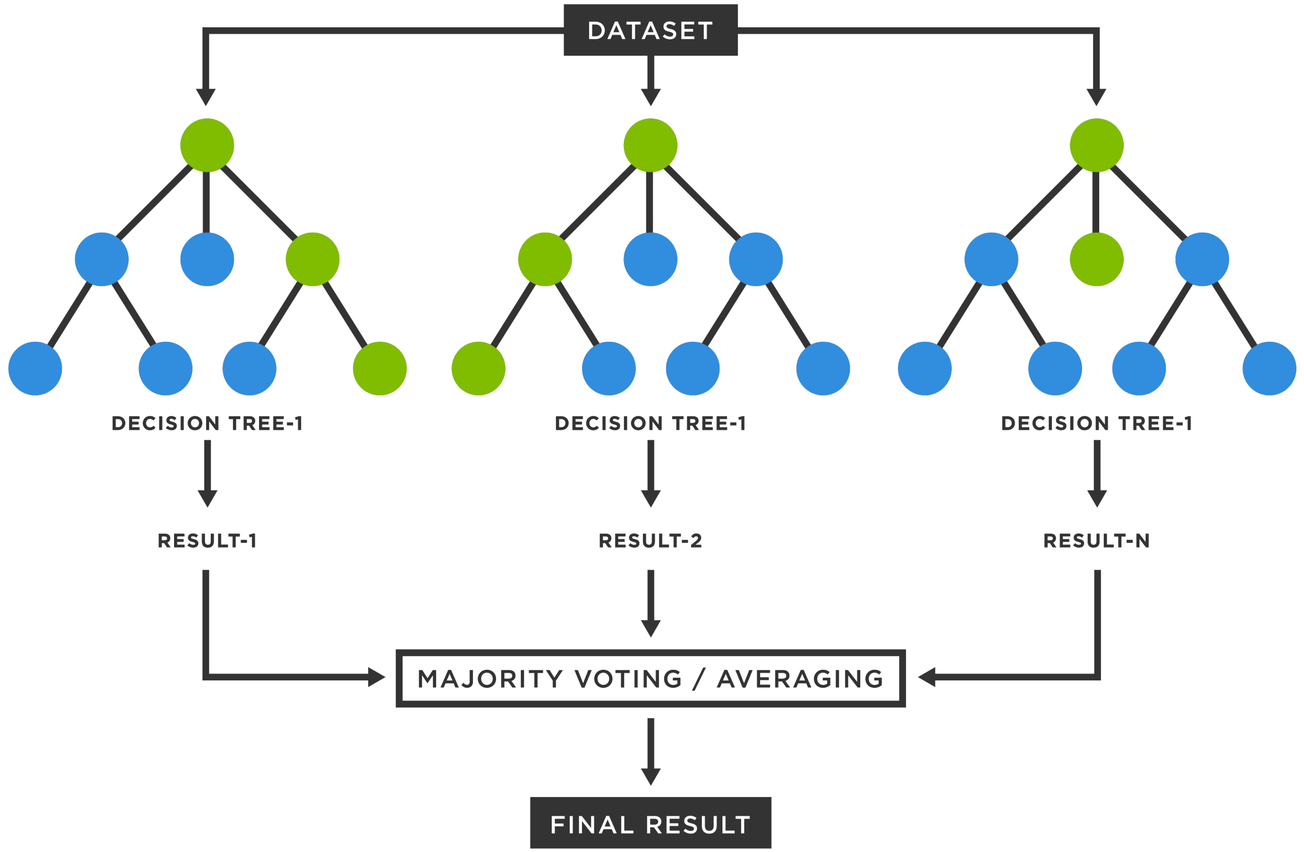
Result explanation

Base on the visualization, correlation and model testing result, the issue of complain is the most important feature that the bank should pay attention to. Secondly, the age group over 40 and female customer exit rate is high. The marketing demography strategy should be adjusted. Thirdly, customer had only one number of product, their exit rate is high. That means the bank should offer more product or motivate customer to use more product.

Chapter Wrap Up

The ramdomforest classifier model is not random at all. It works by the theory of probability and decision tree. So you may imagine it as decision tree forest.

It is very easy to measure the relative importance of each feature on the prediction.



Reference & Resources

Sklearn Website:

https://scikit-learn.org/

Plotly Graph Objects:

https://plotly.com/python/graph-objects/

Seaborn:

https://seaborn.pydata.org/examples/index.html

Matplotlib:

https://matplotlib.org/