

# Ex4: Predicting Customer Churn

- Cho dữ liệu Churn\_Modelling.csv chứa thông tin của 10000 khách hàng của công ty.
- Là phụ trách bộ phận chăm sóc khách hàng bạn nhận thấy việc phải xây dựng một mô hình Machine Learning để dự đoán việc khách hàng sẽ ra đi hay ở lại. Công việc này vô cùng quan trọng vì giữ chân được khách hàng càng lâu doanh nghiệp của bạn sẽ càng tiết kiệm được chi phí và tăng doanh thu.
- Note: Vì khi áp dụng DecisionTree model bị overfitting nên hãy thử làm với RandomForest xem kết quả như thế nào.

## Gợi ý

```
In [1]: # from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
```

```
In [2]: # %cd '/content/gdrive/My Drive/LDS6_MachineLearning/Practice_2023/Chapter5_Random_Forest/'
```

```
In [3]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score
from sklearn.metrics import confusion_matrix, roc_curve, auc
from sklearn.metrics import classification_report
import seaborn as sns
from sklearn.model_selection import train_test_split
```

## Đọc dữ liệu, tiền xử lý dữ liệu

```
In [4]: data = pd.read_csv("Churn_Modelling.csv")
```

```
In [5]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 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  
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

```
In [6]: data.head()
```

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

```
In [7]: # Dựa trên thông tin trên ta thấy các cột không dùng trong model là:
# RowNumber, CustomerId, Surname
# inputs: các cột còn lại trừ cột Exited
# output: cột Exited
```

```
In [8]: X = data.iloc[:, 3:13]
y = data.iloc[:, 13]
```

```
In [9]: X.head()
```



```
Out[9]:
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
0	619	France	Female	42	2	0.00	1	1	1	101348.88
1	608	Spain	Female	41	1	83807.86	1	0	1	112542.58
2	502	France	Female	42	8	159660.80	3	1	0	113931.57
3	699	France	Female	39	1	0.00	2	0	0	93826.63
4	850	Spain	Female	43	2	125510.82	1	1	1	79084.10

```
In [10]: type(y)
```

```
Out[10]: pandas.core.series.Series
```

```
In [11]: y.value_counts() # 0: ở lại, 1: ra đi; tỷ lệ ~ 4:1
```

```
Out[11]: 0    7963
         1    2037
         Name:Exited, dtype: int64
```

```
In [12]: # Các thuộc tính phân loại
objects = [f for f in X.columns if X.dtypes[f] == 'object']
objects
```

```
Out[12]: ['Geography', 'Gender']
```

```
In [13]: # Xem xét thuộc tính phân loại: Geography
X.groupby(by='Geography')['CreditScore'].count()
```

```
Out[13]: Geography
         France    5014
         Germany    2509
         Spain    2477
         Name:CreditScore, dtype: int64
```

```
In [14]: # Dựa trên kết quả ta thấy có 3 quốc gia
# => cần chuyển sang dữ liệu kiểu số
```

```
In [15]: # Xem xét thuộc tính phân loại: Gender
X.groupby(by='Gender')['CreditScore'].count()
```

```
Out[15]: Gender
         Female    4543
         Male    5457
         Name:CreditScore, dtype: int64
```

```
In [16]: # Dựa trên kết quả ta thấy có 2 giới tính
# => cần chuyển sang dữ liệu kiểu số
```

```
In [17]: X_new = pd.get_dummies(X)
```

```
In [18]: X_new.head()
```

```
Out[18]:
```

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Geography_France	Geography_Germany	Geography_Spain
0	619	42	2	0.00	1	1	1	101348.88	1	0	0
1	608	41	1	83807.86	1	0	1	112542.58	0	0	0
2	502	42	8	159660.80	3	1	0	113931.57	1	0	0
3	699	39	1	0.00	2	0	0	93826.63	1	0	0
4	850	43	2	125510.82	1	1	1	79084.10	0	0	0

```
In [19]: # from sklearn.preprocessing import StandardScaler
# sc = StandardScaler()
```

```
In [20]: # X_new_1 = sc.fit_transform(X_new)
```

Áp dụng model, nhận xét kết quả

```
In [21]: X_train, X_test, y_train, y_test = train_test_split(X_new, y,
                                                             test_size = 0.2,
                                                             random_state = 0)
```

```
In [22]: model = RandomForestClassifier()
```

```
In [23]: model.fit(X_train, y_train)
```

```
Out[23]: RandomForestClassifier()
```



```
In [24]: print('Train score: ', model.score(X_train,y_train))
Train score:  1.0

In [25]: print('Test score: ', model.score(X_test,y_test))
Test score:  0.8665

In [26]: # Kết quả trên cho thấy model ít overfitting hơn

In [27]: yhat_test = model.predict(X_test)
yhat_test

Out[27]: array([0, 0, 0, ..., 0, 0, 0], dtype=int64)

In [28]: print("Test Accuracy is ", accuracy_score(y_test,yhat_test)*100,"%")
Test Accuracy is  86.65 %

In [29]: cm = confusion_matrix(y_test, yhat_test)

In [30]: cm

Out[30]: array([[1526,  69],
                [ 198, 207]], dtype=int64)

In [31]: print(classification_report(y_test, yhat_test))

              precision    recall  f1-score   support

    0           0.89       0.96       0.92       1595
    1           0.75       0.51       0.61        405

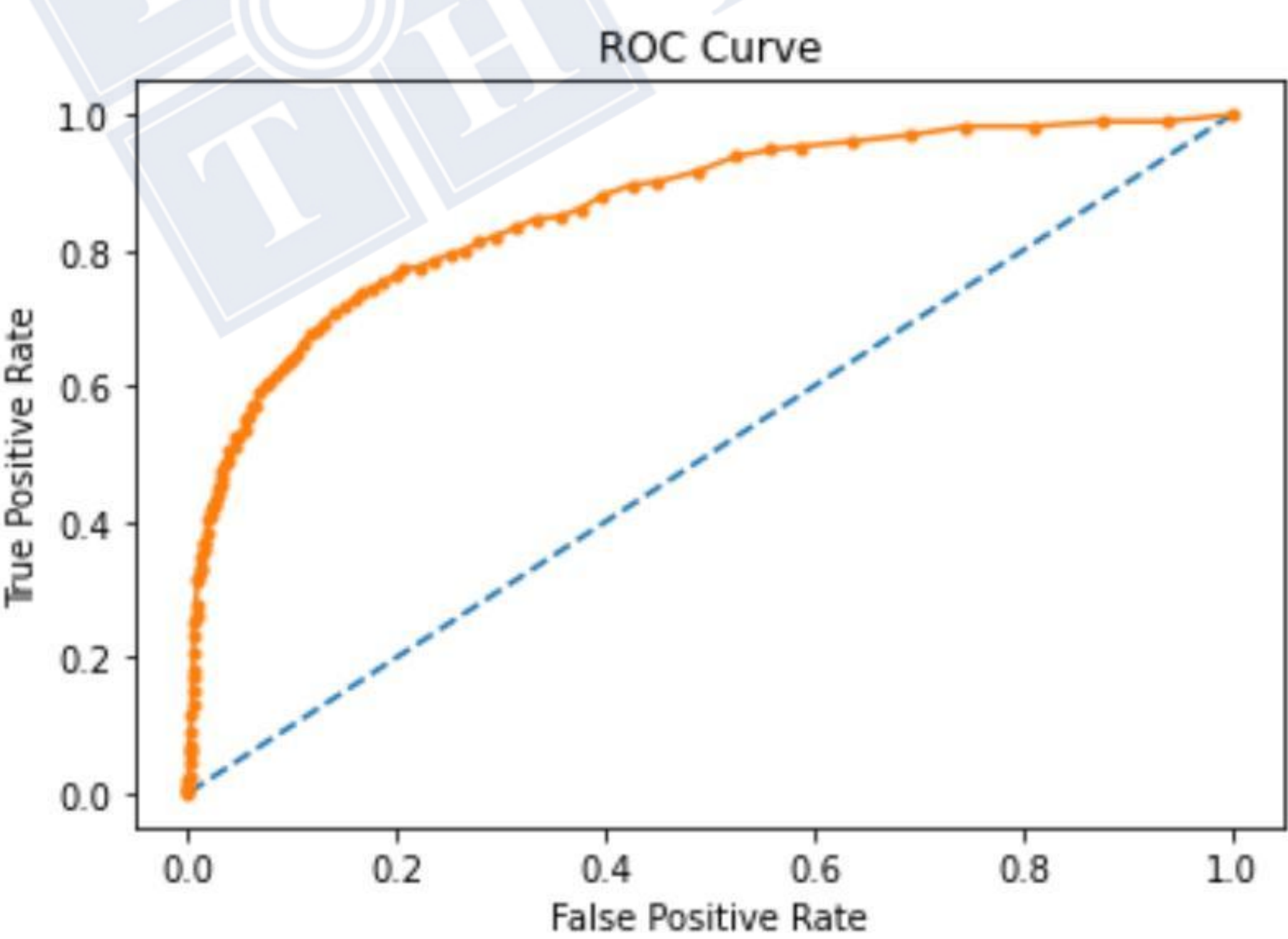
 accuracy          0.87       0.87       0.87       2000
 macro avg         0.82       0.73       0.76       2000
 weighted avg      0.86       0.87       0.86       2000
```

- Kết quả chưa được tốt lắm, có giải pháp nào để tốt hơn không?

```
In [32]: # Print ROC_AUC score using probabilities
probs = model.predict_proba(X_test)

In [33]: scores = probs[:,1]
fpr, tpr, thresholds = roc_curve(y_test, scores)

In [34]: plt.plot([0, 1], [0, 1], linestyle='--')
plt.plot(fpr, tpr, marker='.')
plt.title("ROC Curve")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.show()
```



```
In [35]: auc(fpr, tpr)

Out[35]: 0.8624226943767174
```

**Predicting new samples**

```
In [36]: # Cần phải chuẩn hóa dữ liệu để mẫu mới có cùng cấu trúc với dữ liệu đang có

In [37]: columns = X_new.columns
columns
```



```
Out[37]: Index(['CreditScore', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
              'IsActiveMember', 'EstimatedSalary', 'Geography_France',
              'Geography_Germany', 'Geography_Spain', 'Gender_Female', 'Gender_Male'],
              dtype='object')
```

```
In [38]: new_samples = X.iloc[[0, 1, 2]]
```

```
In [39]: new_samples = pd.get_dummies(new_samples)
```

```
In [40]: new_samples.columns
```

```
Out[40]: Index(['CreditScore', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
              'IsActiveMember', 'EstimatedSalary', 'Geography_France',
              'Geography_Spain', 'Gender_Female'],
              dtype='object')
```

```
In [41]: missing_cols = set(X_new.columns) - set(new_samples.columns)
missing_cols
```

```
Out[41]: {'Gender_Male', 'Geography_Germany'}
```

```
In [42]: for c in missing_cols:
          new_samples[c] = 0
          # Ensure the order of column in the test set
          # is in the same order than in train set
          new_samples = new_samples[X_new.columns]
```

```
In [43]: new_samples.columns
```

```
Out[43]: Index(['CreditScore', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
              'IsActiveMember', 'EstimatedSalary', 'Geography_France',
              'Geography_Germany', 'Geography_Spain', 'Gender_Female', 'Gender_Male'],
              dtype='object')
```

```
In [44]: #new_samples = sc.transform(new_samples)
```

```
In [45]: new_predictions = model.predict(new_samples)
new_predictions
```

```
Out[45]: array([1, 0, 1], dtype=int64)
```

```
In [46]: # Nhận xét kết quả
          # Có giải pháp nào giúp cho kết quả cải thiện hơn không?
```

## Kết luận:

- Kết quả có phù hợp chưa? Có giải pháp nào giúp cho kết quả cải thiện hơn không? Đề xuất.

```
In [ ]:
```