Chapter 2 - Ex3: Predicting Customer Spend

Part2

- Hãy áp dụng thuật toán Linear Regression để xây dựng model dự đoán customer spend dựa vào dữ liệu wrangled_transactions.csv vừa có ở Part1. Đánh giá model. Trực quan hóa kết quả.
- Với '2010 revenue': [1000], 'days_since_last_purchase': [20], 'number_of_purchases': [2], 'avg_order_cost': [500] thì '2011 revenue' là bao nhiêu?

```
!pip install pandas-profiling==2.7.1
```

```
from google.colab import drive
drive.mount("/content/gdrive", force_remount=True)

%cd '/content/gdrive/My Drive/LDS6_MachineLearning/practice_2020/Chapter2_Linear_Regression/'
```

```
Mounted at /content/gdrive
/content/gdrive/My Drive/LDS6_MachineLearning/practice_2020/Chapter2_Linear_Regression
```

Đọc dữ liệu

```
import pandas as pd
import pandas_profiling as pp
```

```
df = pd.read_csv('wrangled_transactions.csv', index_col='CustomerID')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Float64Index: 738 entries, 12347.0 to 18260.0
Data columns (total 6 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   2010 revenue               738 non-null    float64
 1   days_since_first_purchase  738 non-null    float64
 2   days_since_last_purchase   738 non-null    float64
 3   number_of_purchases        738 non-null    float64
 4   avg_order_cost             738 non-null    float64
 5   2011 revenue               738 non-null    float64
dtypes: float64(6)
memory usage: 40.4 KB
```

```
profile = pp.ProfileReport(df)
profile
```

Summarize dataset: 21/? [00:13<00:00, 1.05it/s, Completed]

Generate report structure: 100% 1/1 [00:03<00:00, 3.55s/it]

Render HTML: 100% 1/1 [00:02<00:00, 2.93s/it]

Overview

Overview Reproduction Warnings **2**

Dataset statistics

Number of variables	7
Number of observations	738
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	40.5 KiB
Average record size in memory	56.2 B

Variable types

NUM	7

Variables

```
df.head()
```

CustomerID	2010 revenue	days_since_first_purchase	days_since_last_purchase	number_of_pur
12347.0	711.79	23.0	23.0	
12348.0	892.80	14.0	14.0	
12370.0	1868.02	16.0	13.0	
12377.0	1001.52	10.0	10.0	
12383.0	600.72	8.0	8.0	

```
df.describe()
```

	2010 revenue	days_since_first_purchase	days_since_last_purchase	number_of_purcha
count	738.00000	738.000000	738.000000	738.000
mean	499.80122	21.415989	18.964770	1.342
std	486.51546	5.551448	5.825957	0.708
min	12.45000	7.000000	7.000000	1.000
25%	210.63000	17.000000	14.000000	1.000
50%	337.48000	22.000000	18.000000	1.000
75%	601.47750	25.000000	23.000000	1.000
max	3281.31000	29.000000	29.000000	7.000

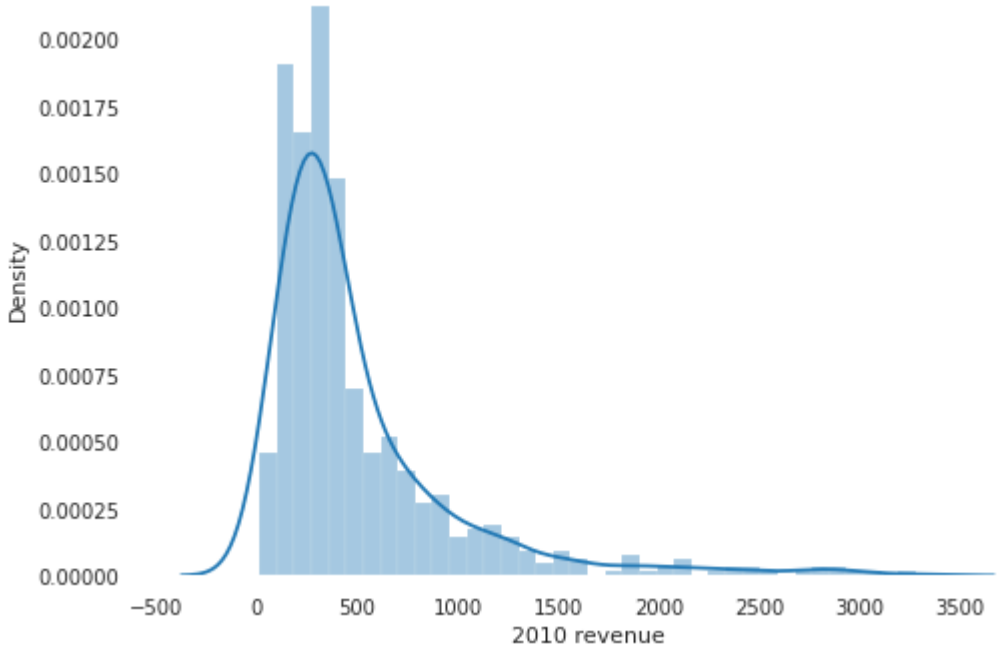
```
df.corr()
```

	2010 revenue	days_since_first_purchase	days_since_last_purchase
2010 revenue	1.000000	0.109692	-0.254964
days_since_first_purchase	0.109692	1.000000	0.641574
days_since_last_purchase	-0.254964	0.641574	1.000000
number_of_purchases	0.504438	0.327502	-0.398268
avg_order_cost	0.779401	-0.074321	-0.054051
2011 revenue	0.548234	0.061743	-0.171294

```
import seaborn as sns
```

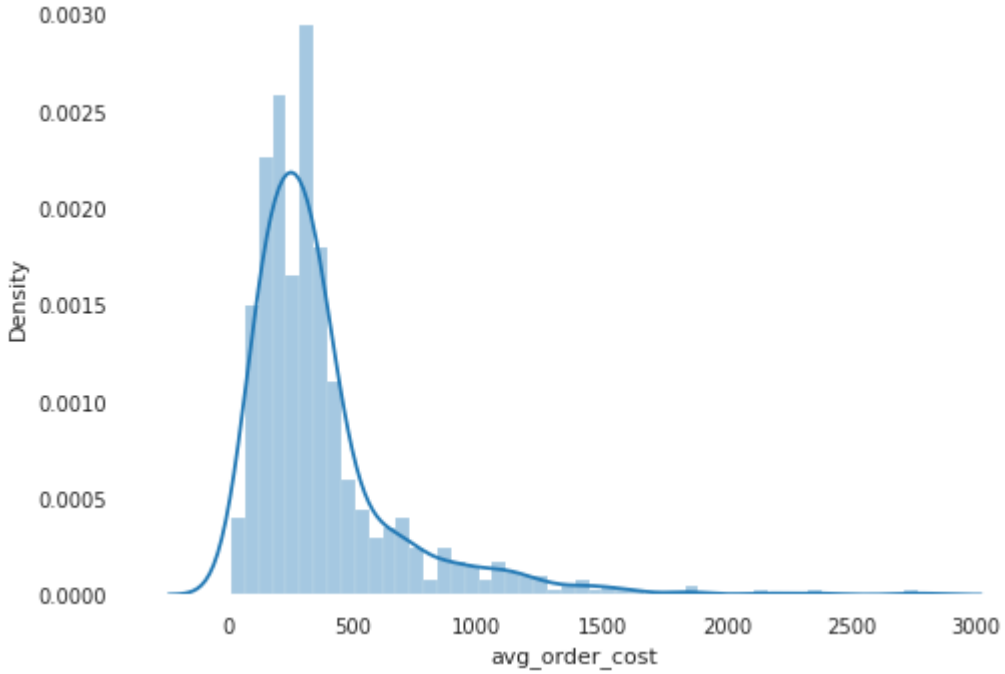
```
sns.distplot(df['2010 revenue'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2557: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f508536ea90>
```




```
sns.distplot(df['avg_order_cost'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2557: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f50852c5d10>
```



```
# Nhận xét: dữ liệu df['2010 revenue'], df['avg_order_cost'] lệch phải


import numpy as np


df['2010 revenue_log'] = np.log(df['2010 revenue'])
df['avg_order_cost_log'] = np.log(df['avg_order_cost'])


df.head()
```

CustomerID	2010 revenue	days_since_first_purchase	days_since_last_purchase	number_of_pur
12347.0	711.79	23.0	23.0	
12348.0	892.80	14.0	14.0	
12370.0	1868.02	16.0	13.0	
12377.0	1001.52	10.0	10.0	
12383.0	600.72	8.0	8.0	

```
X = df[['2010 revenue',
        'days_since_last_purchase',
        'number_of_purchases',
        'avg_order_cost'
        ]]
y = df['2011 revenue']


from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 100)
```

Xây dựng và đánh giá model

[ ] ↳ *11 ô bị ẩn*

Trực quan hóa kết quả

[ ] ↳ *5 ô bị ẩn*

Dự đoán giá trị mới

[ ] ↳ *2 ô bị ẩn*

Áp dụng model sau khi dùng log_scaler

```
df.corr()
```

	2010 revenue	days_since_first_purchase	days_since_last_purchase
2010 revenue	1.000000	0.109692	-0.254964
days_since_first_purchase	0.109692	1.000000	0.641574
days_since_last_purchase	-0.254964	0.641574	1.000000
number_of_purchases	0.504438	0.327502	-0.398268
avg_order_cost	0.779401	-0.074321	-0.054051
2011 revenue	0.548234	0.061743	-0.171294
2010 revenue_log	0.860552	0.117938	-0.248480
avg_order_cost_log	0.716100	-0.051219	-0.064728

```
X1 = df[['2010 revenue_log',
       'days_since_last_purchase',
       'number_of_purchases',
       'avg_order_cost_log'
       ]]
y1 = df['2011 revenue']


X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, random_state = 100)


model1 = LinearRegression()
model1.fit(X1_train,y1_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
model1.score(X1,y1)
```

```
0.24795072793177333
```

```
model1.score(X1_train, y1_train)
```

```
0.24114047911602265
```

```
model1.score(X1_test, y1_test)
```

```
0.2753484766018033
```

```
# Kết quả không tốt
```

Áp dụng model với MinMaxScaler

```
from sklearn.preprocessing import MinMaxScaler
```

```
min_max_scaler = MinMaxScaler()
```

```
X_after_min_max_scaler = min_max_scaler.fit_transform(X)
```

```
X2_train, X2_test, y2_train, y2_test = train_test_split(X_after_min_max_scaler, y, random_sta
```

```
model2 = LinearRegression()
model2.fit(X2_train, y2_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
model2.score(X_after_min_max_scaler, y)
```

```
0.3125705963044775
```

```
model2.score(X2_train, y2_train)
```

```
0.2975351342721504
```

```
model2.score(X2_test, y2_test)
```

```
0.3744858638700586
```

```
# Kết quả không tốt
```

Áp dụng Polinormial Regression

```
from sklearn.preprocessing import PolynomialFeatures
```

```
pr1=PolynomialFeatures(degree=3)
pr1
```

```
PolynomialFeatures(degree=3, include_bias=True, interaction_only=False,
```

```
                            order='C')
```

```
X1_pr=pr1.fit_transform(X)
```

```
X.shape, X1_pr.shape
```

```
((738, 4), (738, 35))
```

```
X1_pr_train, X1_pr_test, y1_pr_train, y1_pr_test = train_test_split(X1_pr, y,
                                                                    random_state = 100)
```

```
model3 = LinearRegression()
model3.fit(X1_pr_train, y1_pr_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
model3.score(X1_pr, y)
```

```
0.4008042974913263
```

```
model3.score(X1_pr_train, y1_pr_train)
```

```
0.43517277743343946
```

```
model3.score(X1_pr_test, y1_pr_test)
```

```
0.2560923127895648
```

Nhận xét:

- Kết quả không tốt. Nguyên nhân:
  - Dữ liệu chưa đủ, chỉ có trong 2 năm 2010 và 2011
  - Model tuyến tính không phù hợp