week6 小红书营销渠道效果预测分析

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

1.数据查看

In [2]:
```
data=pd.read_csv('C:\\Users\mac\Desktop\数据分析班\week6\\小红书数据.csv')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29452 entries, 0 to 29451
Data columns (total 8 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   revenue                29452 non-null  float64
 1   gender                 17723 non-null  float64
 2   age                    17723 non-null  float64
 3   engaged_last_30        17723 non-null  float64
 4   lifecycle              29452 non-null  object 
 5    days_since_last_order  29452 non-null  float64
 6   previous_order_amount  29452 non-null  float64
 7   3rd_party_stores       29452 non-null  int64  
dtypes: float64(6), int64(1), object(1)
memory usage: 1.8+ MB
```

gender，age和engaged_last_30有缺失值
gender和engaged_last_30的数据类型错误

In [3]:
```
data.isnull().sum()
```

Out[3]:
```
revenue                      0
gender                   11729
age                      11729
engaged_last_30          11729
lifecycle                    0
 days_since_last_order       0
previous_order_amount        0
3rd_party_stores             0
dtype: int64
```

缺失值比例

In [4]:
```
data.isnull().sum()/data.shape[0]
```

Out[4]:
```
revenue                   0.000000
gender                    0.398241
age                       0.398241
engaged_last_30           0.398241
lifecycle                 0.000000
 days_since_last_order    0.000000
previous_order_amount     0.000000
```

```
3rd_party_stores          0.000000
dtype: float64
```

比例接近40%，不可直接删除，必须只能填充

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

Out[5]:

	revenue	gender	age	engaged_last_30	lifecycle	days_since_last_order	previous_order_amount
0	72.98	0.0	43.0	0.0	B	4.26	2343.870
1	200.99	0.0	34.0	0.0	A	0.94	8539.872
2	69.98	0.0	16.0	0.0	C	4.29	1687.646
3	649.99	NaN	NaN	NaN	C	14.90	3498.846
4	83.59	NaN	NaN	NaN	C	21.13	3968.490

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

Out[6]:

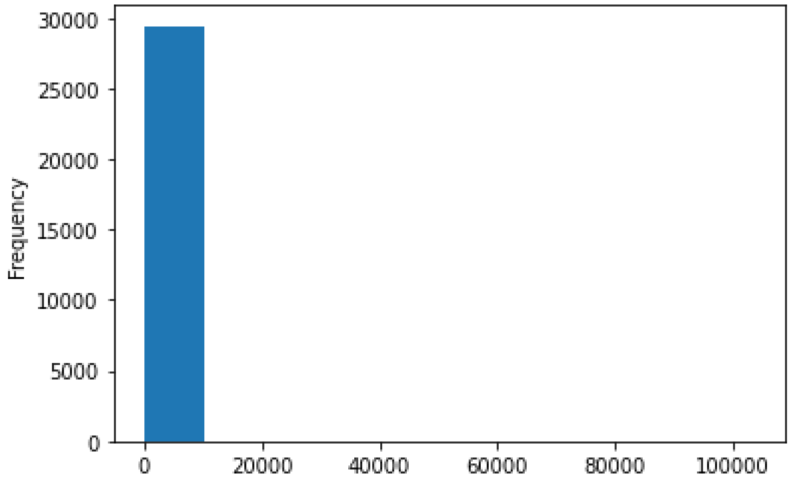
	revenue	gender	age	engaged_last_30	days_since_last_order	previous_c
count	29452.000000	17723.000000	17723.000000	17723.000000	29452.000000	
mean	397.071515	0.298200	29.419286	0.073069	7.711348	
std	959.755615	0.457481	9.213604	0.260257	6.489289	
min	0.020000	0.000000	14.000000	0.000000	0.130000	
25%	74.970000	0.000000	21.000000	0.000000	2.190000	
50%	175.980000	0.000000	29.000000	0.000000	5.970000	
75%	498.772500	1.000000	37.000000	0.000000	11.740000	
max	103466.100000	1.000000	45.000000	1.000000	23.710000	

发现：revenue和previous_order_amount标准差太大，回归分析时需要偏离较大的值

2收入EDA

In [7]:
```
data.revenue.plot(kind="hist")
```

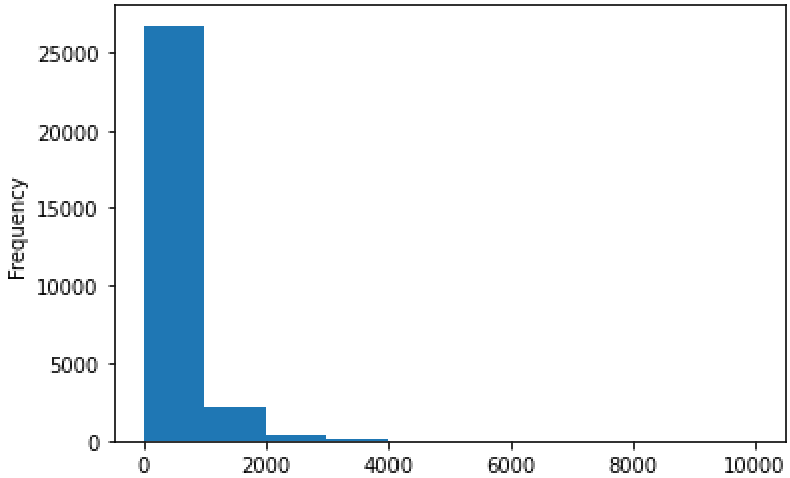
Out[7]: <AxesSubplot:ylabel='Frequency'>



In [8]:

```
data[data.revenue<10000]['revenue'].plot(kind="hist")
```

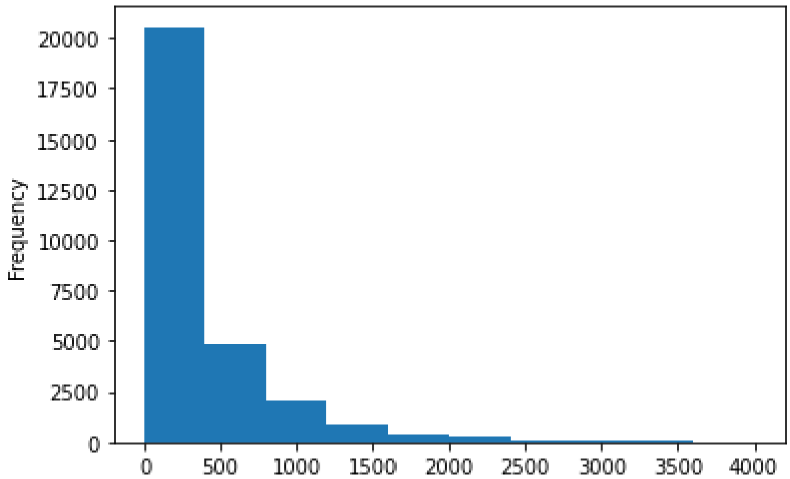
Out[8]: <AxesSubplot:ylabel='Frequency'>



In [9]:

```
data[data.revenue<4000]['revenue'].plot(kind="hist")
```

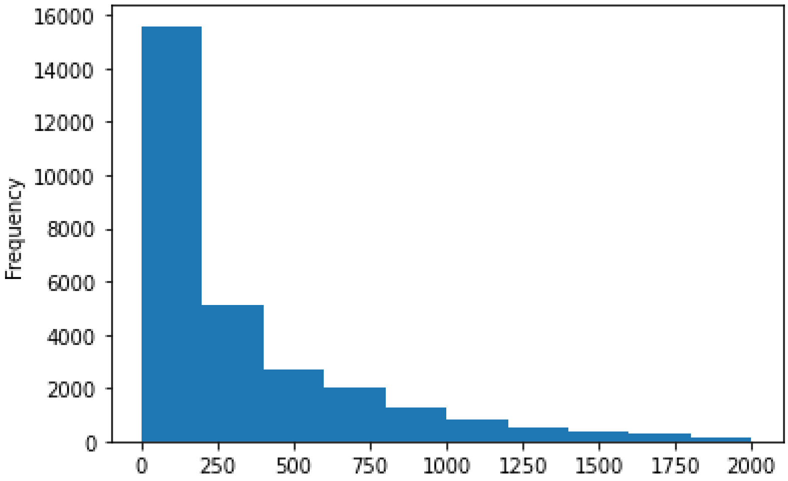
Out[9]: <AxesSubplot:ylabel='Frequency'>



In [10]:

```
data[data.revenue<2000]['revenue'].plot(kind="hist")
```

Out[10]: <AxesSubplot:ylabel='Frequency'>



分布在0~1000的人数占绝大多数

In [11]:
```
data.revenue.describe()
```

Out[11]:
```
count      29452.000000
mean         397.071515
std          959.755615
min            0.020000
25%           74.970000
50%          175.980000
75%          498.772500
max       103466.100000
Name: revenue, dtype: float64
```

In [12]:
```
diff=data.revenue.describe()["75%"]-data.revenue.describe()["25%"]
net_max=data.revenue.describe()["75%"]+1.5*diff
net_max
```

Out[12]:
```
1134.47625
```

故可以认为大于1134.47625的均为revenue的离群值

In [13]:
```
1-data[data.revenue<=1134.47625].shape[0]/data.shape[0]
```

Out[13]:
```
0.07357734619041156
```

离群值占比7%左右，删去是可以接受的

3.相关可视化分析

In [14]:
```
data.corr()['revenue']
```

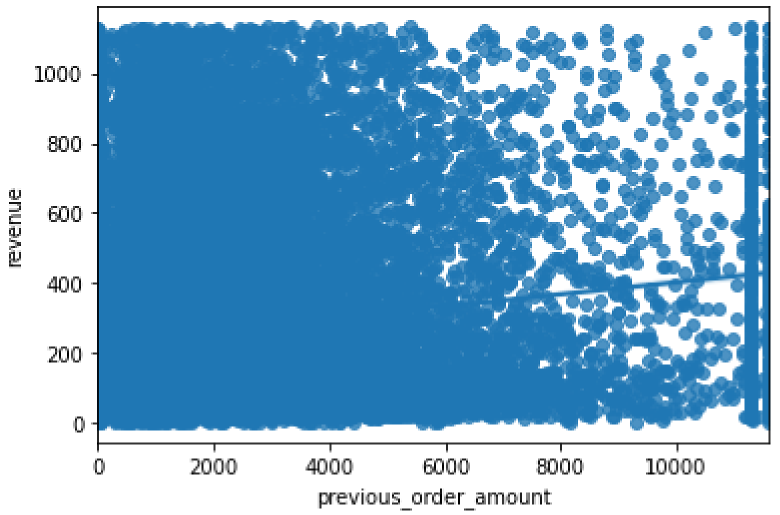
Out[14]:
```
revenue                   1.000000
gender                    0.014944
age                       0.006263
engaged_last_30           0.080031
 days_since_last_order    0.036754
previous_order_amount     0.168186
3rd_party_stores         -0.026102
Name: revenue, dtype: float64
```

由此可得，previous_order_amount 与revenue的相关性最强，其次是engaged_last_30和 days_since_last_order，但都很低

In [15]:
```
data=data[data.revenue<=1134.47625]
```

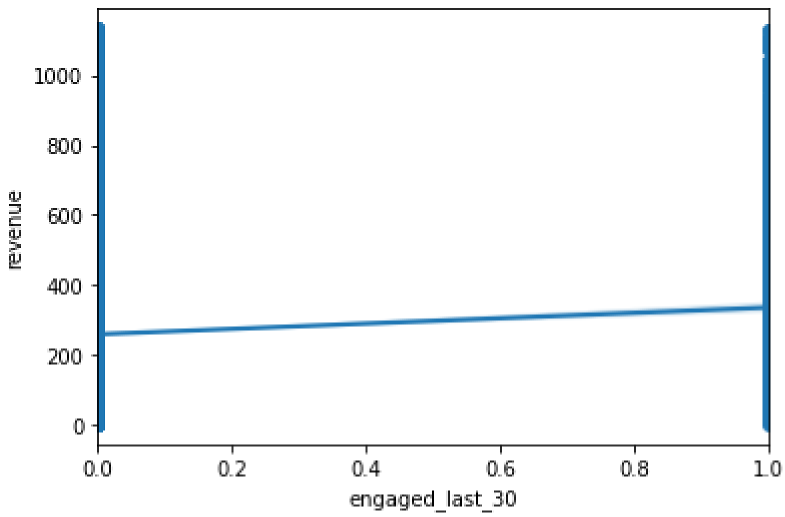
In [16]:
```
sns.regplot(x='previous_order_amount',y='revenue',data=data)
```

Out[16]: <AxesSubplot:xlabel='previous_order_amount', ylabel='revenue'>



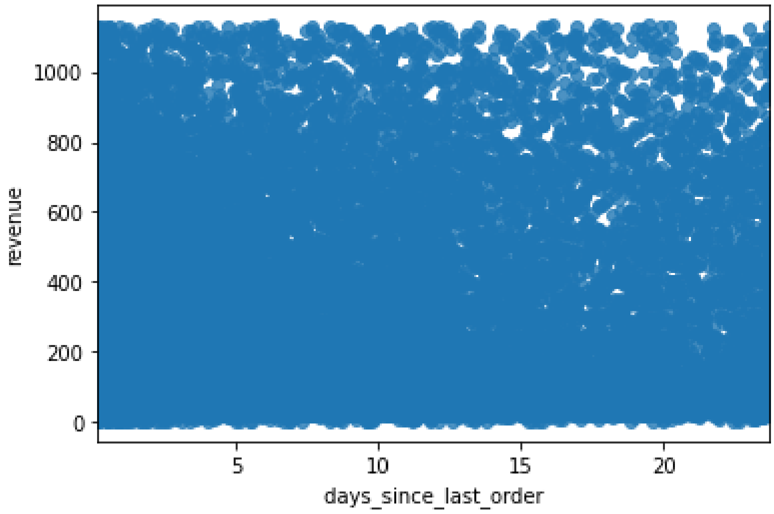
In [17]:
```
sns.regplot(x='engaged_last_30',y='revenue',data=data)
```

Out[17]: <AxesSubplot:xlabel='engaged_last_30', ylabel='revenue'>



In [18]:
```
sns.regplot(x=" days_since_last_order ",y='revenue',data=data)
```

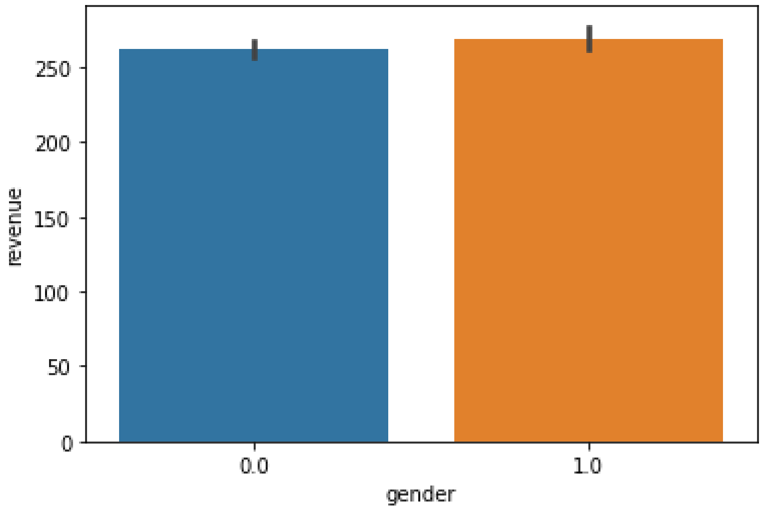
Out[18]: <AxesSubplot:xlabel=' days_since_last_order ', ylabel='revenue'>



In [19]:

```
sns.barplot(x="gender", y="revenue", data=data)
```

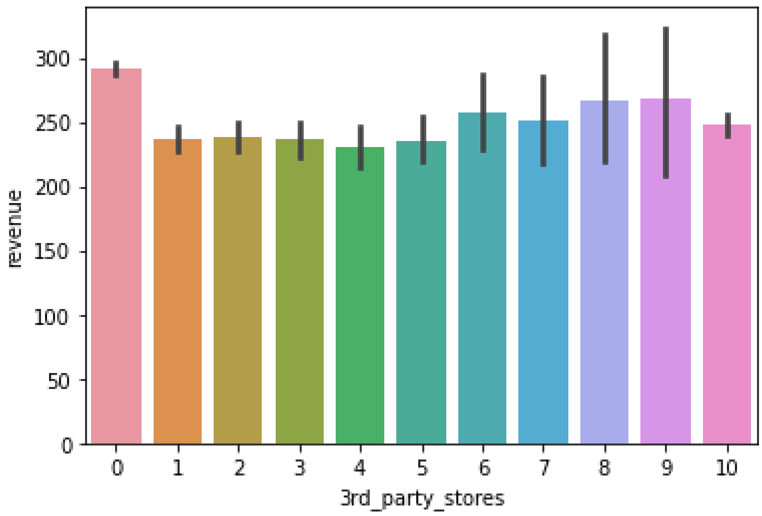
Out[19]: <AxesSubplot:xlabel='gender', ylabel='revenue'>



In [20]:

```
sns.barplot(x="3rd_party_stores", y="revenue", data=data)
```

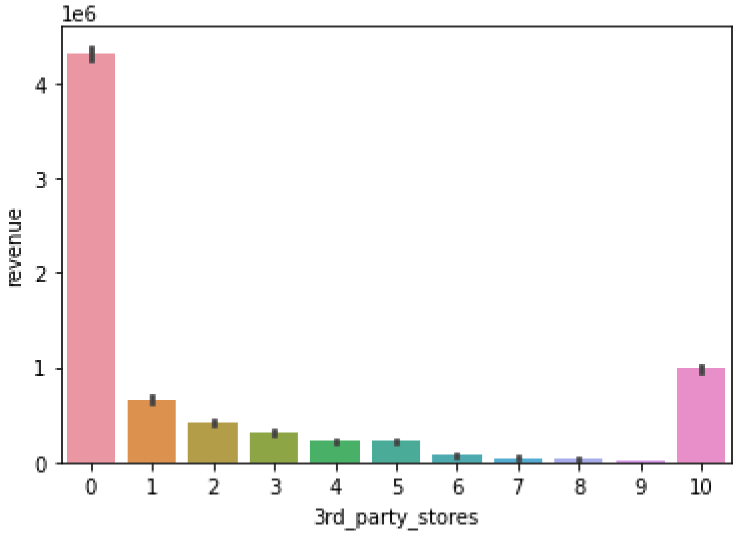
Out[20]: <AxesSubplot:xlabel='3rd_party_stores', ylabel='revenue'>



In [21]:

```
sns.barplot(x="3rd_party_stores", y="revenue", estimator=sum, data=data)
```

Out[21]: <AxesSubplot:xlabel='3rd_party_stores', ylabel='revenue'>



4.数据处理

4.1 revenue数据清洗

In [22]:
```
df=data.copy()
```

In [23]:
```
df=df[df.revenue<=1134.47625]
```

4.2缺失值填充

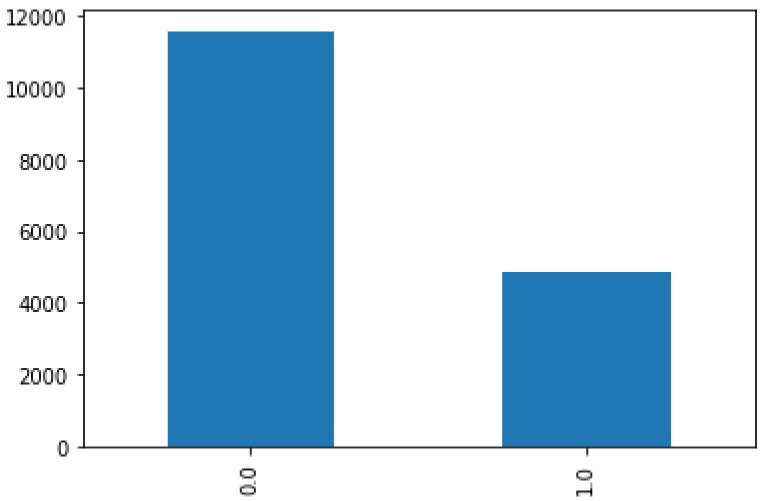
4.2.1性别

In [24]:
```
df.gender.unique()
```

Out[24]: array([0., nan, 1.])

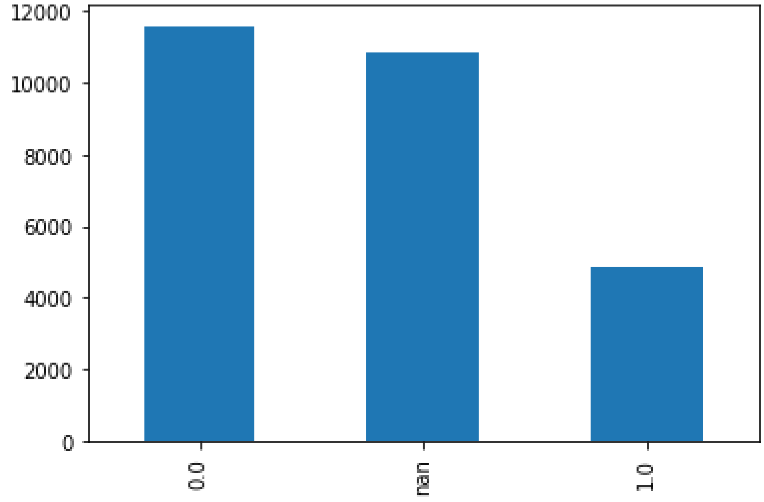
In [25]:
```
df['gender'].value_counts().plot(kind="bar")
```

Out[25]: <AxesSubplot:>



In [26]:
```
df['gender'].value_counts(dropna=False).plot(kind="bar")
```

Out[26]: <AxesSubplot:>

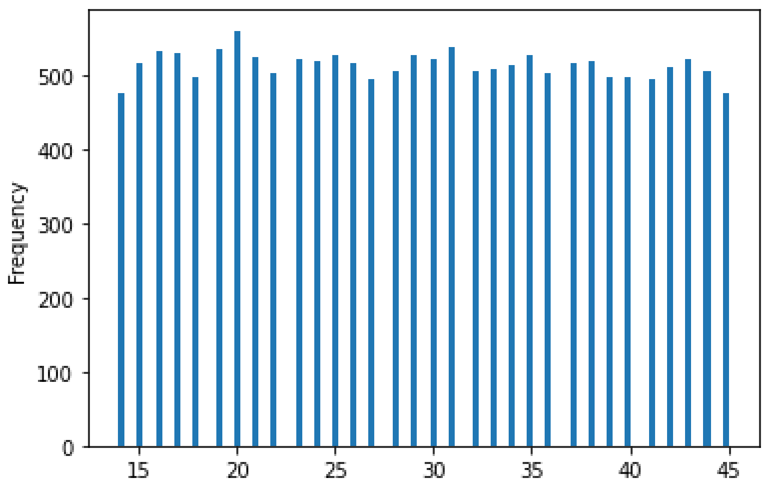


In [27]:
```
df['gender']=df['gender'].fillna('unknow')
```

4.2.2年龄

In [28]:
```
df['age'].plot(kind="hist",bins=100)
```

Out[28]: <AxesSubplot:ylabel='Frequency'>



In [29]:
```
df['age']=df['age'].fillna(df['age'].mean())
```

4.2.3过去三十天参与的活动

In [30]:
```
df['engaged_last_30']=df['engaged_last_30'].fillna(0.0)
```

4.2.4更改数据类型

In [31]:
```
df['gender']=df['gender'].astype(str)
```

In [32]:
```
df['engaged_last_30']=df['engaged_last_30'].astype(str)
```

4.2.5添加哑变量

In [33]:
```
df=pd.get_dummies(df,drop_first=True)
```

In [34]:
```
df.head()
```

Out[34]:

	revenue	age	days_since_last_order	previous_order_amount	3rd_party_stores	gender_1.0
0	72.98	43.000000	4.26	2343.870	0	0
1	200.99	34.000000	0.94	8539.872	0	0
2	69.98	16.000000	4.29	1687.646	1	0
3	649.99	29.402114	14.90	3498.846	0	0
4	83.59	29.402114	21.13	3968.490	4	0

5.使用Python建立线性回归模型

In [35]:
```
from sklearn.linear_model import LinearRegression as lrg
```

In [60]:
```
x=df[['previous_order_amount'," days_since_last_order "]]
y=df['revenue']
```

In [61]:
```
model=lrg()
model.fit(x,y)
```

Out[61]: LinearRegression()

In [62]:
```
model.coef_
```

Out[62]: array([0.01842654, 5.00525826])

In [63]:
```
model.intercept_
```

Out[63]: 189.21781583073073

5.1 模型评估

In [65]:
```
socre=model.score(x,y)
socre
```

Out[65]: 0.03505897168369887

5.1预测用户的消费金额变化

In [66]:
```
y_pred=model.predict(x)
```

In [68]:
```
error=y_pred-y
error
```

Out[68]:
```
0         180.749624
1         150.293029
2         171.807846
3        -321.722220
4         284.514452
             ...
29447     160.778538
29448     212.411158
29449     177.689914
29450     266.004564
29451    -478.274606
Name: revenue, Length: 27285, dtype: float64
```

In [69]:
```
rmse=(error**2).mean()**0.5
rmse
```

Out[69]:
```
263.2192954187874
```

In [70]:
```
mae=abs(error).mean()
mae
```

Out[70]:
```
208.21575378676076
```

In [71]:
```
from statsmodels.formula.api import ols
model_ols=ols('y~x',df).fit()
print(model_ols.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.035
Model:                            OLS   Adj. R-squared:                  0.035
Method:                 Least Squares   F-statistic:                     495.6
Date:                Tue, 15 Dec 2020   Prob (F-statistic):          3.76e-212
Time:                        20:53:22   Log-Likelihood:            -1.9077e+05
No. Observations:               27285   AIC:                         3.816e+05
Df Residuals:                   27282   BIC:                         3.816e+05
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    189.2178      3.030     62.454      0.000     183.279     195.156
x[0]           0.0184      0.001     25.806      0.000       0.017       0.020
x[1]           5.0053      0.249     20.132      0.000       4.518       5.493
==============================================================================
Omnibus:                     5183.316   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             8675.546
Skew:                           1.302   Prob(JB):                         0.00
Kurtosis:                       3.920   Cond. No.                     6.02e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.02e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```