小灶商业数据分析Python训练营

week6　线性回归与销售额预测

第一课 宝洁销售额预测数据探索

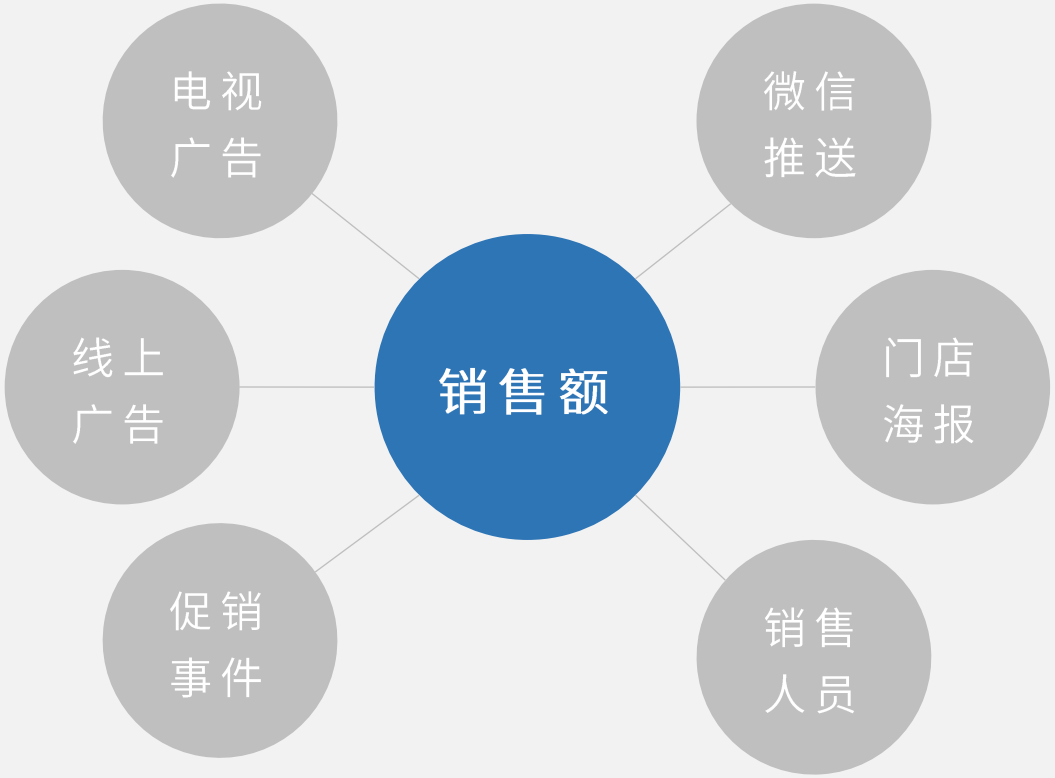
案例背景

对于宝洁这样的快消品企业，重要的数据应用：

- 对商超门店的销售额做出精准预测
- 量化自身所控制的各种促销因素所能产生的效果
- 对营销资源做出合理规划

案例分析

分析宝洁哪个营销渠道对门店销售额的影响最大



P&G

小灶能力派

案例分析

关键数据维度

- 以下数据均以月为观测窗口
  - Revenue门店销售额
  - Reach微信推送次数
  - Local_ tv本地电视广告投入
  - Online线上广告投入
  - Instore门店内海报陈列等投入
  - Person门店销售人员投入
  - Event促销事件:cobranding品牌联合促销，holiday节假日，special门店特别促销，non-event无促销活动

分析流程

数据概况分析

- 数据行/列数量
- 缺失值分布

单变量分析

- 数字型变量的描述指标(平均值,最小值,最大值,标准差等)
- 类别型变量(多少个分类，各自占比)

相关与可视化

- 按类别交叉对比
- 变量之间的相关性分析
- 散点图/热力图

回归模型

- 模型建立
- 模型评估与优化

目 录

小灶能力派

1.1 了解数据的概况

代码讲解

- import pandas as pd #调包
- store=pd.read_csv('w2_store_rev.csv') #数据读取
- store.info()

- 读取数据

小灶能力派

1.1 了解数据的概况

代码结果

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 985 entries, 0 to 984
Data columns (total 8 columns):
Unnamed: 0    985 non-null int64
revenue       985 non-null float64
reach         985 non-null int64
local_tv      929 non-null float64
online        985 non-null int64
instore       985 non-null int64
person        985 non-null int64
event         985 non-null object
dtypes: float64(2), int64(5), object(1)
memory usage: 61.7+ KB
```

- 发现Unnamed

1.2 去除Unnamed=0数据

代码讲解

- store=pd.read_csv('w2_store_rev.csv'，index_col=0)
  - 意思是第一列就是index值，不用新增一列unnamed

1.2 去除Unnamed=0数据

代码结果

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 985 entries, 845 to 26
Data columns (total 7 columns):
revenue     985 non-null float64
reach       985 non-null int64
local_tv    929 non-null float64
online      985 non-null int64
instore     985 non-null int64
person      985 non-null int64
event       985 non-null object
dtypes: float64(2), int64(4), object(1)
memory usage: 61.6+ KB
```

- local_tv有50多个空值
- 发现event不是数字型变量，

  无法在sklearn中处理

小灶能力派

1.3 处理发现的空值

代码讲解

- store.isnull().sum()
  - .isnull()判断数据集store每一列是否是空，是的话标1，不是的话标0
  - .sum()最后加总

小灶能力派

1.3 处理发现的空值

代码结果

```
revenue       0
reach         0
local_tv     56
online        0
instore       0
person        0
event         0
dtype: int64
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 985 entries, 845 to 26
Data columns (total 7 columns):
revenue      985 non-null float64
reach        985 non-null int64
local_tv     929 non-null float64
online       985 non-null int64
instore      985 non-null int64
person       985 non-null int64
event        985 non-null object
dtypes: float64(2), int64(4), object(1)
memory usage: 61.6+ KB
```

1.3 处理发现的空值

代码讲解

- store.describe()
  - 了解数据分布、大小情况
  - 确认数据与真实情况一致

小灶能力派

1.3 处理发现的空值

代码结果

	revenue	reach	local_tv	online	instore	person
count	985.000000	985.000000	929.000000	985.000000	985.000000	985.000000
mean	38357.355025	3.395939	31324.061109	1596.527919	3350.962437	11.053807
std	11675.603883	1.011913	3970.934733	496.131586	976.546381	3.041740
min	5000.000000	0.000000	20000.000000	0.000000	0.000000	0.000000
25%	30223.600000	3.000000	28733.830000	1253.000000	2690.000000	9.000000
50%	38159.110000	3.000000	31104.520000	1607.000000	3351.000000	11.000000
75%	45826.520000	4.000000	33972.410000	1921.000000	4011.000000	13.000000
max	79342.070000	7.000000	43676.900000	3280.000000	6489.000000	24.000000

第二课　宝洁促销事件单变量分析

目录

2.1 了解促销事件

代码讲解

- store.event.unique()
  - 数据来源.变量名.unique()
  - unique()函数的作用：以数组的形式返回该列所有唯一值
  - store，DataFrame名
  - 替代写法：store['event'].unique()

2.1 了解促销事件

代码结果

- store.event.unique()

```
Out[95]: array(['non_event', 'special', 'cobranding', 'holiday'], dtype=object)
```

2.2 了解促销事件带来的收入

代码讲解

- store.groupby(['event'])['revenue'].describe()
  - 数据来源.groupby(['你想分类的维度'])['你想查看的数据维度'].describe
  - 把收入按照促销事件分类

小灶能力派

2.2 了解促销事件带来的收入

代码结果

event	count	mean	std	min	25%	50%	75%	max
cobranding	398.0	38277.664497	11879.097324	7146.99	30472.1525	37864.155	46333.5600	79342.07
holiday	103.0	37791.890583	11942.369136	5000.00	29644.5250	38432.780	46036.1300	73377.15
non_event	192.0	37903.081563	11186.436740	6874.43	29852.3775	37937.175	44611.6375	69429.39
special	292.0	38964.136438	11648.616882	10207.96	30325.8125	39197.870	45897.0400	71757.50

- special的起点和25%分位数据相对较高

2.3 数据处理（增加哑变量）

代码讲解

- store=pd.get_dummies(store)
- store.head(10)

- 量化变量，设定变量的值，方便数据处理。例：男=1，女=0
- 格式：pd.get_dummies(数据)
- store=pd.get_dummies(store['event'])，只处理event变量

2.3 数据处理（增加哑变量）

代码结果

	revenue	reach	local_tv	online	instore	person	event_cobranding	event_holiday	event_non_event	event_special
845	45860.28	2	31694.91	2115	3296	8	0	0	1	0
483	63588.23	2	35040.17	1826	2501	14	0	0	0	1
513	23272.69	4	30992.82	1851	2524	6	0	0	0	1
599	45911.23	2	29417.78	2437	3049	12	0	0	0	1
120	36644.23	2	35611.11	1122	1142	13	1	0	0	0
867	36172.81	4	22372.59	2001	1881	17	1	0	0	0
847	43797.03	3	31443.74	1667	1846	15	1	0	0	0
950	41629.80	4	35775.75	1155	2715	12	0	0	0	1
942	21303.48	2	24888.31	1853	3677	4	0	0	1	0
550	20746.15	4	26623.48	1497	3075	9	0	1	0	0

小灶能力派

2.3 数据处理（增加哑变量）

代码讲解

- store.info()
  - 获取处理后的新的数据信息

2.3 数据处理（增加哑变量）

代码结果

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 985 entries, 845 to 26
Data columns (total 10 columns):
revenue              985 non-null float64
reach                985 non-null int64
local_tv             929 non-null float64
online               985 non-null int64
instore              985 non-null int64
person               985 non-null int64
event_cobranding     985 non-null uint8
event_holiday        985 non-null uint8
event_non_event      985 non-null uint8
event_special        985 non-null uint8
dtypes: float64(2), int64(4), uint8(4)
memory usage: 57.7 KB
```

- 确认类别变量已经转化为数字变量
- 找到关键变量，为下一步建模做准备

第三课　宝洁变量相关与可视化分析

本节课内容

研究各个变量间的相关性

- 找出与销售额相关性最高的变量
- 针对相关性最高的3个变量做可视化分析
- 更加直观地观察变量间的相关性，完成建模前的探索性分析

小灶能力派

目录

小灶能力派

3.1 相关性分析

代码讲解

- store.corr()
  - 数据.corr()会直接列出数据中所有变量彼此对应的相关性

小灶能力派

3.1 相关性分析

代码结果

- 在这里可以初步看到local_tv、instore与person和revenue的相关性

```
1 store.corr()
```

	revenue	reach	local_tv	online	instore	person	event_cobranding	event_holiday	event_non_event	event_special
revenue	1.000000	-0.155314	0.602114	0.171227	0.311739	0.559208	-0.005623	-0.016559	-0.019155	0.033752
reach	-0.155314	1.000000	-0.034039	-0.025141	0.035635	0.061417	0.043809	0.020398	-0.043128	-0.023330
local_tv	0.602114	-0.034039	1.000000	0.006775	-0.046825	0.048664	0.020886	-0.039650	0.011335	-0.005874
online	0.171227	-0.025141	0.006775	1.000000	-0.026399	0.036662	-0.024646	-0.018596	-0.020587	0.056799
instore	0.311739	0.035635	-0.046825	-0.026399	1.000000	-0.007482	-0.057725	0.045963	0.015495	0.017788
person	0.559208	0.061417	0.048664	0.036662	-0.007482	1.000000	0.002439	-0.025692	-0.025568	0.036771
event_cobranding	-0.005623	0.043809	0.020886	-0.024646	-0.057725	0.002439	1.000000	-0.281389	-0.405169	-0.534499
event_holiday	-0.016559	0.020398	-0.039650	-0.018596	0.045963	-0.025692	-0.281389	1.000000	-0.168151	-0.221824
event_non_event	-0.019155	-0.043128	0.011335	-0.020587	0.015495	-0.025568	-0.405169	-0.168151	1.000000	-0.319403
event_special	0.033752	-0.023330	-0.005874	0.056799	0.017788	0.036771	-0.534499	-0.221824	-0.319403	1.000000

3.1 相关性分析

代码讲解

store.corr()[['revenue']].sort_values('revenue',ascending=False)

仅查看所有变量与revenue的相关性，同时根据相关性做降序排列展示

- corr()[['字段']]查看某一字段和所有变量的相关性
  - 在函数后加括号corr()[['revenue']]，可获得与revenue（即分析中的因变量y）有关的变量相关性分析
  - 在这里，选取df中的单列字段要加两个中括号，否则会变成series

3.1 相关性分析

代码讲解

store.corr()[['revenue']].sort_values('revenue',ascending=False)

仅查看所有变量与revenue的相关性，同时根据相关性做降序排列展示

- sort_values()根据某一字段做升降序排列
  - sort_values('字段',ascending=False)
  - 不加ascending则默认升序
  - ascending=False则设置为降序

小灶能力派

3.1 相关性分析

代码结果

```
1 store.corr()[['revenue']].sort_values('revenue',ascending=False)
```

	revenue
revenue	1.000000
local_tv	0.602114
person	0.559208
instore	0.311739
online	0.171227
event_special	0.033752
event_cobranding	-0.005623
event_holiday	-0.016559
event_non_event	-0.019155
reach	-0.155314

- 在这一步我们看到local_tv/person/instore是三个与revenue相关性最高的变量
- 对于相关性较低的变量虽然不能优先选为模型中的因变量，但也不代表它们与业务的相关性真的很弱
- 回归、预测等分析，数据的颗粒度越细越好，当前的数据不全，不适合过度解读

3.2 相关性可视化

代码讲解

- sns.regplot('local_tv','revenue',store)# 对local_tv变量进行线性关系可视化分析
- sns.regplot('person','revenue',store)# 对person变量进行线性关系可视化分析
- sns.regplot('instore','revenue',store)# 对instore变量进行线性关系可视化分析

3.2 相关性可视化

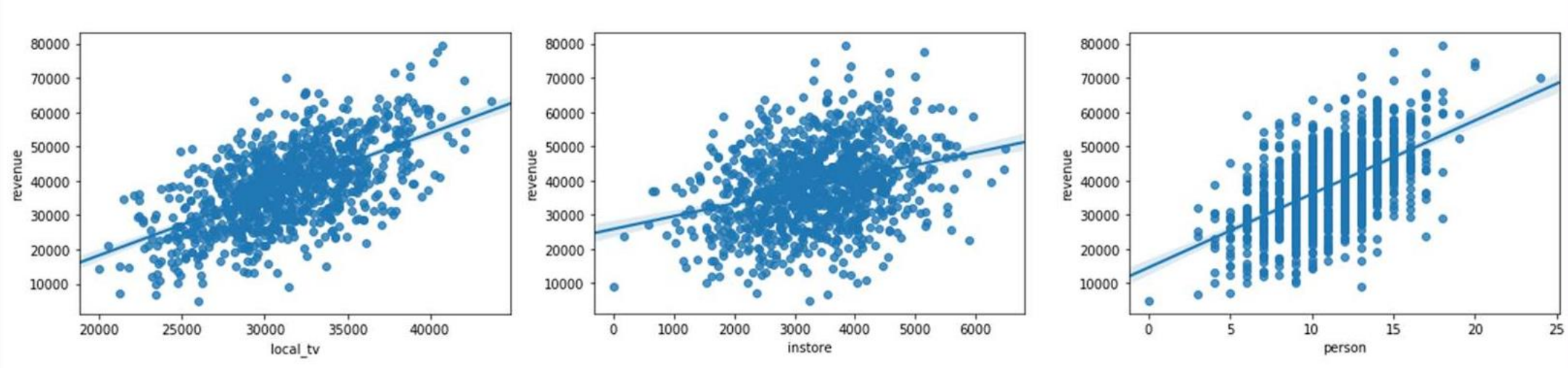
代码讲解

- regplot()进行相关性可视化
  - sns.regplot(字段1，字段二，数据)会针对数据中的字段1和字段2生成一副散点图并添加一条线性回归的拟合参考线
  - 以sns.regplot('local_tv', 'revenue', store)为例
    I. 自变量x在前，为local_tv
    II. 因变量y在后，为revenue

小灶能力派

3.2 相关性可视化

代码结果



local_tv与revenue

本地电视广告与门店销售额

instore与revenue

门店海报陈列与门店销售额

person与revenue

门店销售与门店销售额

小灶能力派

第四课　线性回归模型的建立与评估

目录

小灶能力派

4.1 线性回归模型的建立

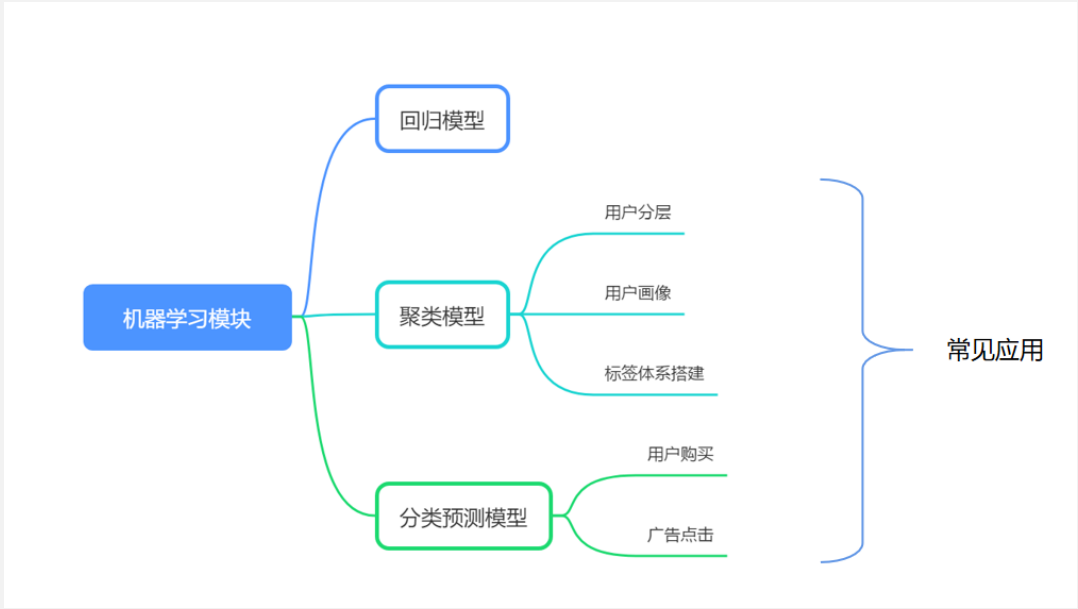
什么是sklearn?

- sklearn（是Scikit-learn的缩写）是Python中常用于机器学习的第三方模块。在实战中使用Scikit-learn可以极大节省编写代码的时间，减少代码量，让我们有精力分析数据分布，调整模型和修改参数。
- sklearn中包含很多种机器学习的方式，学习时不要直接用，先了解一下都有什么模型方法，然后选择适当的方法，来达到你的目标。

4.1 线性回归模型的建立

sklearn

- 常见的机器学习模块：回归模型、聚类模型、分类预测模型
- 常见应用：用户分层/用户画像/标签体系搭建，用户购买/广告点击



小灶能力派

4.1 线性回归模型的建立

降维

- 贷款需求——在P2P平台借钱、申请信用卡、信用卡额度的使用情况……
- 建模时，只选择有代表性的变量，简化建模的计算

4.1 线性回归模型的建立

线性回归模型

- 作用：确定两种或两种以上变量之间，线性的变量关系

4.1 线性回归模型的建立

代码演示——8行代码完成回归模型

```
# 调用sklearn中的线性回归工具包
from sklearn.linear_model import LinearRegression
# 导入数据
loaded_data=datasets.load_boston()
# 设定X和Y变量
data_X=loaded_data.data
data_Y=loaded_data.target
```

4.1 线性回归模型的建立

代码演示——8行代码完成回归模型

```
# 设置模型为线性回归
model=LinearRegression()
# 训练数据，得出参数
model.fit(data_X,data_Y)
# 利用模型预测新数据，和原标签比较
print(model.predict(data_X[:4,:]))
print(data_Y[:4])
```

4.1 线性回归模型的建立

代码结果

```
print(model.predict(data_X[:4,:]))
print(data_Y[:4])
```

```
[30.00384338 25.02556238 30.56759672 28.60703649]
[24.  21.6 34.7 33.4]
```

4.1 线性回归模型的建立

代码讲解

- # 调用sklearn中的线性回归工具包

```
from sklearn.linear_model import LinearRegression
```

- # LinearRegression()设置模型为线性回归

```
model=LinearRegression()
```

- # 设定自变量和因变量

```
y=store['revenue']
x=store[['local_tv','person','instore']]
```

- model. fit(x,y)

4.1 线性回归模型的建立

代码讲解

- from sklearn.linear_model import LinearRegression，

  sklearn-库，linear_model-线性模型库，LinearRegression-

  线性模型的一个分支

- model=LinearRegression()，命名，避免重复写函数
- model.fit(data_X,data_Y)，通过fit()函数建模

4.1 线性回归模型的建立

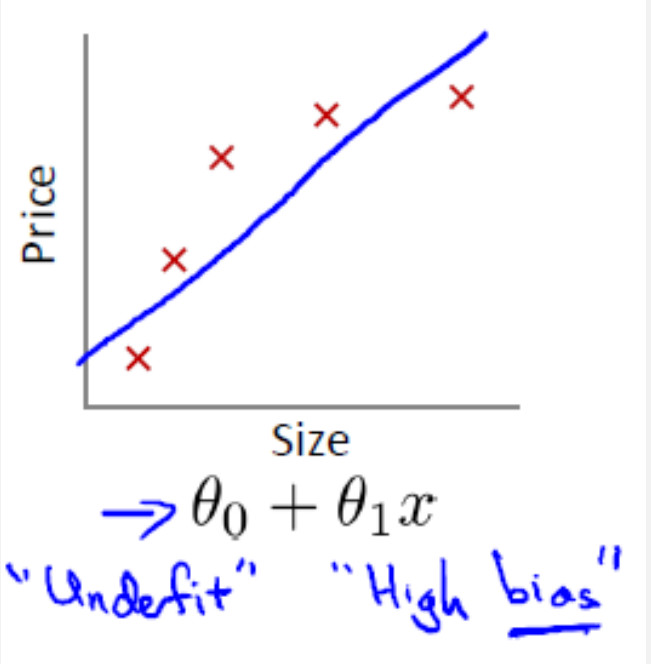
建模结果的拟合与过拟合

- 拟合：找到描述X和Y之间线性关系的线的过程
- 过拟合：拟合的函数完美匹配训练集数据，导致模型的解释率变差

4.1 线性回归模型的建立

建模结果的拟合与过拟合

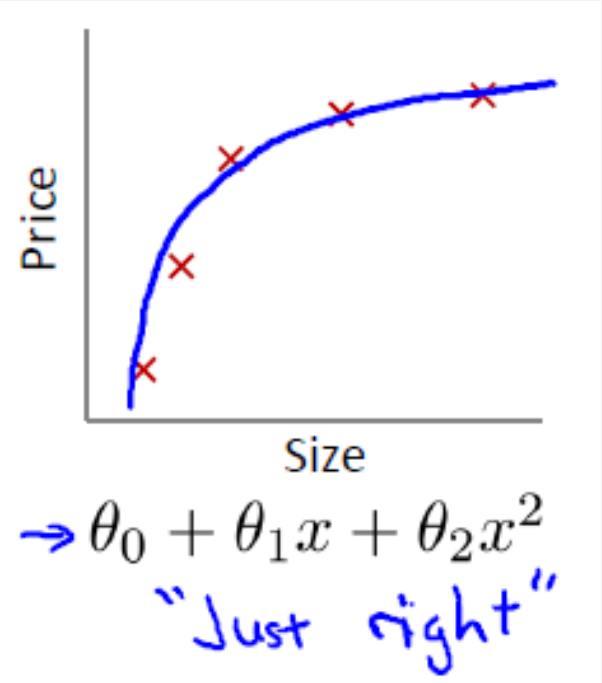
- 拟合的函数和训练集误差较大——欠拟合



4.1 线性回归模型的建立

建模结果的拟合与过拟合

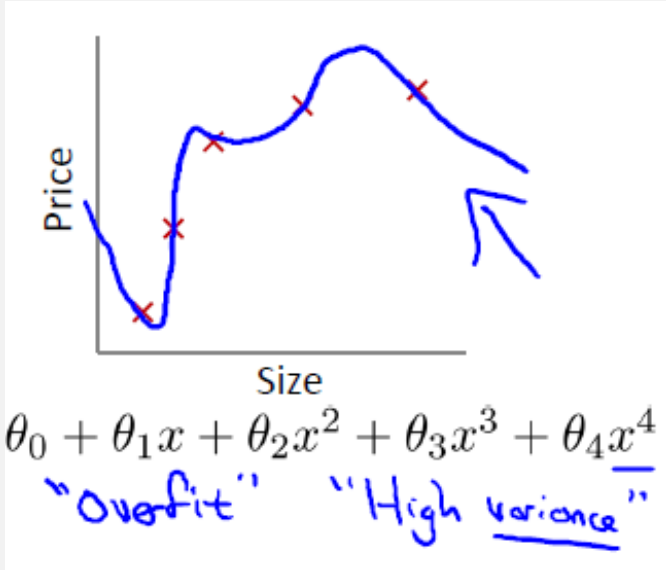
- 拟合的函数和训练集误差较小——合适拟合



4.1 线性回归模型的建立

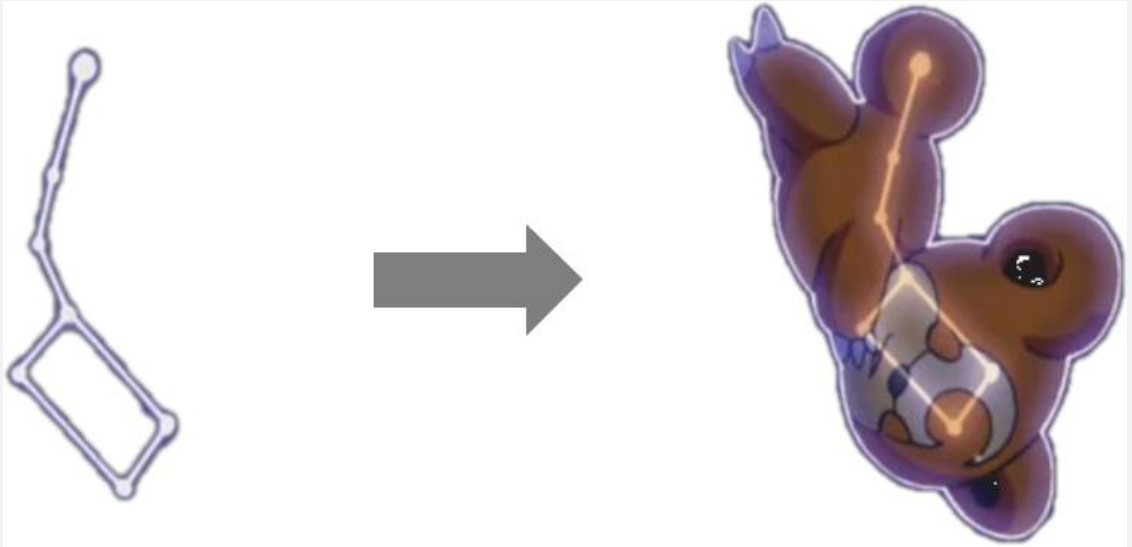
建模结果的拟合与过拟合

- 拟合的函数完美匹配训练集数据——过拟合



4.1 线性回归模型的建立

建模结果的拟合与过拟合



- 拟合
- 过度拟合

4.1 线性回归模型的建立

代码结果

```
--> 542                                    allow_nan=force_all_finite == 'allow-nan')
    543
    544     if ensure_min_samples > 0:

/usr/local/lib64/python3.6/site-packages/sklearn/utils/validation.py in _assert_all_finite(X, allow_n
     54                 not allow_nan and not np.isfinite(X).all()):
     55             type_err = 'infinity' if allow_nan else 'NaN, infinity'
---> 56             raise ValueError(msg_err.format(type_err, X.dtype))
     57     # for object dtype data, we only check for NaNs (GH-13254)
     58     elif X.dtype == np.dtype('object') and not allow_nan:

ValueError: Input contains NaN, infinity or a value too large for dtype('float64').
```

- NaN，空值，之前local_tv中存在空值

小灶能力派

4.1 线性回归模型的建立

代码结果

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 985 entries, 845 to 26
Data columns (total 7 columns):
revenue     985 non-null float64
reach       985 non-null int64
local_tv    929 non-null float64
online      985 non-null int64
instore     985 non-null int64
person      985 non-null int64
event       985 non-null object
dtypes: float64(2), int64(4), object(1)
memory usage: 61.6+ KB
```

- NaN，空值，之前local_tv中存在空值

小灶能力派

4.1 线性回归模型的建立

发现缺失值的处理

- 根据缺失的比例处理
  - 缺失不严重：例如5%左右，可以直接填充
  - 缺失严重：缺失值过多时，例如达到80%~90%，在大数据时代，缺失也是一种信息。把缺失/不缺失，编成编码，看能不能发现相关性

4.1 线性回归模型的建立

发现缺失值的处理

- 填充方式
  - 以0填充
  - 均值填充
  - 中位数填充
  - 数据模型填充——local_tv和其他变量的线性模型填充

4.1 线性回归模型的建立

代码讲解

- store=store.fillna(0)
  - 以0填充：用0来填充not available的值（缺失值）
  - 格式：store=store.fillna(0)

4.1 线性回归模型的建立

代码结果

	revenue	reach	local_tv	online	instore	person
count	985.000000	985.000000	929.000000	985.000000	985.000000	985.000000
mean	38357.355025	3.395939	31324.061109	1596.527919	3350.962437	11.053807
std	11675.603883	1.011913	3970.934733	496.131586	976.546381	3.041740
min	5000.000000	0.000000	20000.000000	0.000000	0.000000	0.000000
25%	30223.600000	3.000000	28733.830000	1253.000000	2690.000000	9.000000
50%	38159.110000	3.000000	31104.520000	1607.000000	3351.000000	11.000000
75%	45826.520000	4.000000	33972.410000	1921.000000	4011.000000	13.000000
max	79342.070000	7.000000	43676.900000	3280.000000	6489.000000	24.000000

- 数据比较大，直接填充为0稍草率，之后用其他方法

4.1 线性回归模型的建立

代码讲解

- store.info()
  - 查看是否填充完毕

4.1 线性回归模型的建立

代码结果

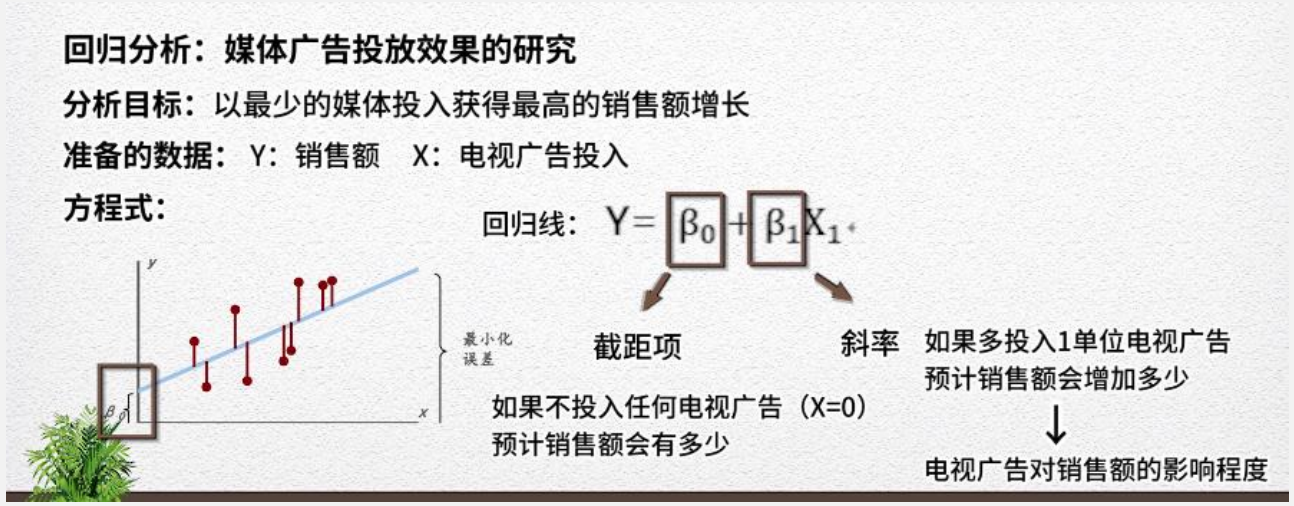
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 985 entries, 845 to 26
Data columns (total 10 columns):
revenue              985 non-null float64
reach                985 non-null int64
local_tv             985 non-null float64
online               985 non-null int64
instore              985 non-null int64
person               985 non-null int64
event_cobranding     985 non-null uint8
event_holiday        985 non-null uint8
event_non_event      985 non-null uint8
event_special        985 non-null uint8
dtypes: float64(2), int64(4), uint8(4)
memory usage: 57.7 KB
```

- 空值填充成功

小灶能力派

4.1 线性回归模型的建立

线性回归模型公式



- 线性回归模型公式：y预测值=截距+系数*x变量

小灶能力派

4.1 线性回归模型的建立

代码讲解

- model.coef_
  - 查看自变量系数

4.1 线性回归模型的建立

代码结果

```
array([4.00189943e-01, 2.13224898e+03, 3.56098623e+00])
```

- 对应'local_tv','person','instore'三个变量
- 分别得到自变量系数

4.1 线性回归模型的建立

代码讲解

- model.intercept_
  - 查看截距

4.1 线性回归模型的建立

代码结果

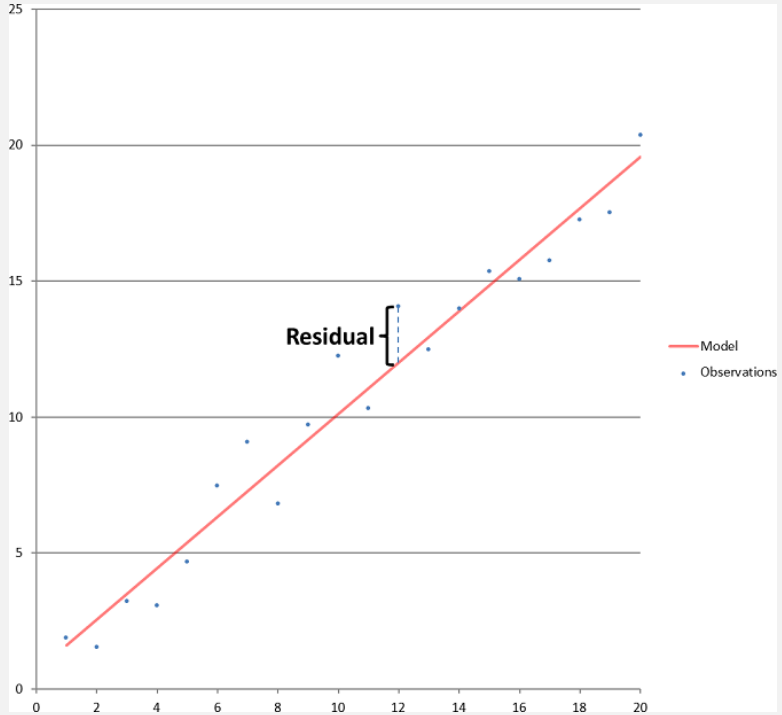
```
-8967.736870300963
```

- 得到截距值

4.2 线性回归模型的评估和优化

线性回归模型的评估

- 核心就是看residual（残差）
- 预测值和真实值之间的差距



4.2 线性回归模型的评估和优化

评估的常用指标

- MAE（平均绝对误差）

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}\left|y_j - \hat{y}_j\right|$$

  - 计算方法：取预测值-真实值的绝对值，加和后再除以样本数n
  - 反映预测值误差的实际情况

4.2 线性回归模型的评估和优化

评估的常用指标

- RMSE（均方根误差）

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2}$$

  - 计算方法：将每个数据点的误差取平方后开方
  - 常用来作为机器学习模型预测结果衡量的标准，比起MAE放大了误差，对误差的惩罚更重

4.2 线性回归模型的评估和优化

评估的常用指标

- RMSE（均方根误差）

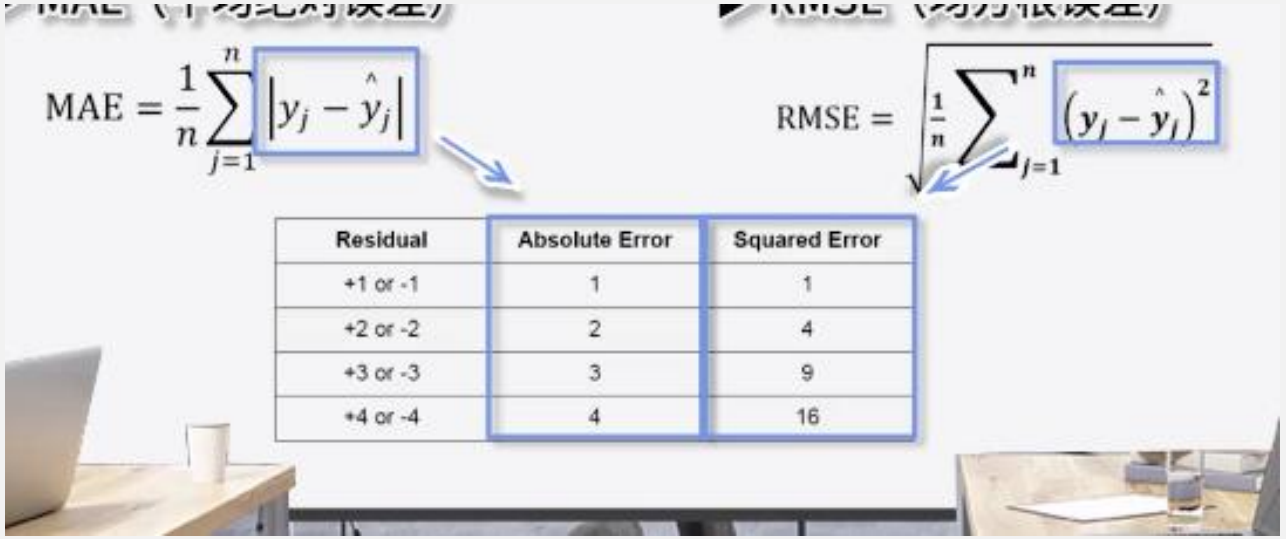
Residual	Absolute	Squared Error
+1 or -1	1	1
+2 or -2	2	4
+3 or -3	3	9
+4 or -4	4	16

小灶能力派

4.2 线性回归模型的评估和优化

评估的常用指标

- RMSE（均方根误差）



小灶能力派

4.2 线性回归模型的评估和优化

代码讲解

```
# 计算y预测值
predictions=model.predict(x)
# 计算误差
error=predictions-y
# 计算rmse
rmse=(error**2).mean()**.5
# 计算mae
mae=abs(error).mean()
print(rmse)
print(mae)
```

4.2 线性回归模型的评估和优化

代码讲解

- score=model.score(x,y)，把生成的模型变成y和x的关系
- predictions=model.predict(x)，生成prediction，存放输入x后y的预测值
- error=predictions-y，残差=预测值-y
- rmse=(error**2).mean()**.5，**为平方的过程

4.2 线性回归模型的评估和优化

代码结果

```
8321.491623472051
6556.036999600778
```

- 分别得到MAE和RSEM的值

4.2 线性回归模型的评估和优化

回顾

- 回归模型的建模过程
  - 调包——建模——设定X和Y——模型训练——模型评估和优化

4.2 线性回归模型的评估和优化

优化阶段1：增加online变量

- 在原有的三个变量中'local_tv','person','instore'，加入第四个'online'，看是否能提高模型的预测效果
- 注意：加入新的变量后，整体模型指标会发生什么变化

  新的变量对系数y有什么影响

4.2 线性回归模型的评估和优化

优化阶段1：增加online变量

```
]: store.corr()
```

]:

	revenue	reach	local_tv	online	inst
revenue	1.000000	-0.155314	0.602114	0.171227	0.311
reach	-0.155314	1.000000	-0.034039	-0.025141	0.035
local_tv	0.602114	-0.034039	1.000000	0.006775	-0.046
online	0.171227	-0.025141	0.006775	1.000000	-0.026
instore	0.311739	0.035635	-0.046825	-0.026399	1.000
person	0.559208	0.061417	0.048664	0.036662	-0.007

4.2 线性回归模型的评估和优化

优化阶段1：增加online变量

代码讲解

- # 调用sklearn中的线性回归工具包
- from sklearn.linear_model import LinearRegression
- # LinearRegression()设置模型为线性回归
- model=LinearRegression()
- # 设定自变量和因变量
- y=store['revenue']
- x=store[['local_tv','person','instore','online']]

4.2 线性回归模型的评估和优化

优化阶段1：增加online变量

代码结果

```
rmse=(error**2).mean()**.5
# 计算mae
mae=abs(error).mean()
print(rmse)
print(mae)
```

```
8321.491623472051
6556.036999600778
```

之前

```
rmse=(error**2).mean()**.5
# 计算mae
mae=abs(error).mean()
print(rmse)
print(mae)
```

```
8106.512169325369
6402.202883441894
```

之后

小灶能力派

4.2 线性回归模型的评估和优化

优化阶段1：增加online变量

代码结果

```
store.corr()
```

	revenue	
revenue	1.000000	-0
reach	-0.155314	1
local_tv	0.602114	-0
online	0.171227	-0
instore	0.311739	0
person	0.559208	0
event_cobranding	-0.005623	0
event_holiday	-0.016559	0
event_non_event	-0.019155	-0
event_special	0.033752	-0

- 结果是预测值和真实值的差，越小越好，可见误差变小了

4.2 线性回归模型的评估和优化

优化阶段2：均值填充local_tv

	revenue	reach	local_tv	online	instore	person
count	985.000000	985.000000	929.000000	985.000000	985.000000	985.000000
mean	38357.355025	3.395939	31324.061109	1596.527919	3350.962437	11.053807
std	11675.603883	1.011913	3970.934733	496.131586	976.546381	3.041740
min	5000.000000	0.000000	20000.000000	0.000000	0.000000	0.000000
25%	30223.600000	3.000000	28733.830000	1253.000000	2690.000000	9.000000
50%	38159.110000	3.000000	31104.520000	1607.000000	3351.000000	11.000000
75%	45826.520000	4.000000	33972.410000	1921.000000	4011.000000	13.000000
max	79342.070000	7.000000	43676.900000	3280.000000	6489.000000	24.000000

- 之前填充local_tv的方法是填充0，但local_tv的值通常较大，填充0会造成一定误差
- 若使用均值填充，可能会有不一样的结果

4.2 线性回归模型的评估和优化

优化阶段2：均值填充local_tv

代码讲解

- store=pd.read_csv('w2_store_rev.csv',index_col=0)
- store=pd.get_dummies(store)
- store=store.fillna(store.local_tv.mean())
- y=store['revenue']
- x=store[['local_tv','person','instore','online']]
- model.fit(x,y)
- #x和y打分
- score=model.score(x,y)

- store=store.fillna(store.local_tv.mean())，用均值填充到趋势值上

4.2 线性回归模型的评估和优化

优化阶段2：均值填充local_tv

代码讲解

- #计算y预测值
- predictions=model.predict(x)
- #计算误差
- error=predictions-y
- #计算rmse
- rmse=(error**2).mean()**.5
- #计算mae
- mae=abs(error).mean()
- print(rmse)
- print(mae)

4.2 线性回归模型的评估和优化

优化阶段2：均值填充local_tv

代码结果

- 使用均值填充后，发现RMSE和MAE的值都大幅降低，模型的预测效果有很大提升

```
rmse=(error**2).mean()**.5
#  计算mae
mae=abs(error).mean()
print(rmse)
print(mae)
```

```
8106.512169325369
6402.202883441894
```

之前

```
model.fit(x,y)
score=model.score(x,y)#x和y打分
predictions=model.predict(x)#计算y预测值
error=predictions-y#计算误差
rmse=(error**2).mean()**.5#计算rmse
mae=abs(error).mean()#计算mae
print(rmse)
print(mae)
```

```
5591.764749669001
4485.506383110867
```

之后

小灶能力派

4.2 线性回归模型的评估和优化

回顾

- 模型优化的两种方式
  - 增加新变量
  - 清洗和整理原数据

小灶能力派

4.3 线性回归模型建立的另一种方式

代码讲解

- # 查看标准的模型输出表
- from statsmodels.formula.api import ols
- x=store[['local_tv','person','instore']]
- y=store['revenue']
- model=ols('y~x',store).fit()
- print(model.summary())

4.3 线性回归模型建立的另一种方式

代码讲解

- ols（ordinary least squares） 普通最小二乘法，它假设残差最小的时候模型是最优的
- x，自变量
- y，应变量（预测目标）
- model=ols('y~x',store).fit()，'y~x'，用x预测y，数据来源是store, 拟合这个模型

4.3 线性回归模型建立的另一种方式

代码结果

```
                            OLS Regression Results
==============================================================================
=======
Dep. Variable:                      y   R-squared:
0.746
Model:                            OLS   Adj. R-squared:
0.745
Method:                 Least Squares   F-statistic:
959.2
Date:                Wed, 24 Jun 2020   Prob (F-statistic):          4.
09e-291
Time:                        10:45:58   Log-Likelihood:
-9947.5
No. Observations:                 985   AIC:                          1.
990e+04
Df Residuals:                     981   BIC:                          1.
992e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
=======
                 coef    std err          t      P>|t|      [0.025
```

小灶能力派

第五课　宝洁回归模型结果与业务解读

目录

小灶能力派

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
#调包
import pandas as pd
# 读取数据
store=pd.read_csv('w2_store_rev.csv',index_col=0)
# 查看数据信息
store.info()
# 去除Unnamed:0的影响
store=pd.read_csv('w2_store_rev.csv',index_col=0)
# 统计数据空值
store.isnull().sum()
# 查看数据分布
store.describe()
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
#了解event的具体值
store.event.unique()
#这些类别对应的revenue(销售额)是怎样的
store.groupby(['event'])['revenue'].describe()
#这几个类别对应的local_tv（本地电视广告投入）是怎样的
store.groupby(['event'])['local_tv'].describe()
#将类别变量转化为哑变量
store=pd.get_dummies(store)
# 查看生成event的4个标签，每个标签取值0/1
store.head(10)
#确认类别变量已经转换成数字变量
store.info()
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
# 调用sklearn中的线性回归工具包
from sklearn.linear_model import LinearRegression
# LinearRegression()设置模型为线性回归
model=LinearRegression()
# 设定自变量和因变量
y=store['revenue']
x=store[['local_tv','person','instore']]
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
# 缺失值填充
store=store.fillna(0)
# 查看是否填充完毕
store.info()
# 重新加载填充完的X变量
x=store[['local_tv','person','instore']]
# 重新训练模型
model.fit(x,y)
# 查看自变量系数
model.coef_
# 查看截距
model.intercept_
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
#模型的评估,x为'local_tv','person','instore'
score=model.score(x,y)#x和y打分
predictions=model.predict(x)#计算y预测值
error=predictions-y#计算误差
rmse=(error**2).mean()**.5#计算rmse
mae=abs(error).mean()#计算mae
print(rmse)
print(mae)
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
#模型的评估,x为'local_tv','person','instore'，'online'
x=store[['local_tv','person','instore','online']]
model.fit(x,y)
#发现模型误差略有调整
score=model.score(x,y)#x和y打分
predictions=model.predict(x)#计算y预测值
error=predictions-y#计算误差
rmse=(error**2).mean()**.5#计算rmse
mae=abs(error).mean()#计算mae
print(rmse)
print(mae)
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
#对local_tv进行均值填充
store=pd.read_csv('w2_store_rev.csv',index_col=0)
store=pd.get_dummies(store)
store=store.fillna(store.local_tv.mean())
y=store['revenue']
x=store[['local_tv','person','instore','online']]
model.fit(x,y)
score=model.score(x,y)#x和y打分
predictions=model.predict(x)#计算y预测值
error=predictions-y#计算误差
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
#计算rmse
rmse=(error**2).mean()**.5
#计算mae
mae=abs(error).mean()
print(rmse)
print(mae)
```

5.1 线性回归模型建立的操作过程

建立模型的代码回顾

```
# 查看标准的模型输出表
from statsmodels.formula.api import ols
x=store[['local_tv','person','instore']]
y=store['revenue']
model=ols('y~x',store).fit()
print(model.summary())
```

5.1 线性回归模型建立的操作过程

建立模型的结果

OLS Regression Results

Dep. Variable:	y	R-squared:	0.746
Model:	OLS	Adj. R-squared:	0.745
Method:	Least Squares	F-statistic:	959.2
Date:	Sun, 08 Dec 2019	Prob (F-statistic):	4.09e-291
Time:	19:39:13	Log-Likelihood:	-9947.5
No. Observations:	985	AIC:	1.990e+04
Df Residuals:	981	BIC:	1.992e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-5.288e+04	1804.489	-29.305	0.000	-5.64e+04	-4.93e+04
x[0]	1.7515	0.049	35.857	0.000	1.656	1.847
x[1]	2050.5749	61.866	33.146	0.000	1929.171	2171.979
x[2]	4.0903	0.193	21.229	0.000	3.712	4.468

Omnibus:	0.352	Durbin-Watson:	2.056
Prob(Omnibus):	0.839	Jarque-Bera (JB):	0.402
Skew:	0.043	Prob(JB):	0.818
Kurtosis:	2.951	Cond. No.	3.05e+05

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.05e+05. This might indicate that there are
strong multicollinearity or other numerical problems.

5.2 宝洁销售额回归模型解读

模型解读

- 销售额=-52880+1.75*local_tv+2050*person+4.09*instore
  - local_tv代表本地电视广告投入
  - person代表门店销售人员投入
  - instore代表门店内海报陈列等投入

5.2 宝洁销售额回归模型解读

业务解读

- 每提升1元的电视广告投入，可以得到1.75元的销售额回报
- 而每提升1元的店内海报投入，则可以实现4.09元的销售回报
- 不断收集数据和添加新变量能提升对整体营销资源投入的把握