**Ограничения линейной регрессии**

* Линейность: зависимая переменная может линейно аппроксимировать независимые переменные
* Нормальность распределения Y и ε
* Отсутствие избытка влиятельных наблюдений
* Гомоскедастичность распределения остатков
* Отсутствие мультиколлинеарности

Загрузим встроенные данные из библиотеки Skalern:

from sklearn.datasets import load\_boston  
boston = load\_boston()

Boston

|  |  |  |
| --- | --- | --- |
| CRIM | per capita crime rate by town | Уровень преступности на душу населения по городам |
| ZN | proportion of residential land zoned for lots over 25,000 sq.ft. | Доля земли под жилую застройку зонирована под участки площадью более 25000 кв. футов. |
| INDUS | proportion of non-retail business acres per town | Доля акров на город, не связанных с розничной торговлей. |
| CHAS | Charles River dummy variable (= 1 if tract bounds river; 0 otherwise) | Фиктивная переменная Charles River (принимает значение 1, если участок ограничивает реку; 0 в противном случае) |
| NOX | nitric oxides concentration (parts per 10 million) | Концентрация оксидов азота (частей на 107) |
| RM | average number of rooms per dwelling | Среднее количество комнат в доме |
| AGE | proportion of owner-occupied units built prior to 1940 | Доля домов, построенных до 1940 года, занимаемых владельцами. |
| DIS | weighted distances to five Boston employment centres | Расстояние до пяти бостонских центров занятости |
| RAD | RAD index of accessibility to radial highways | Индекс доступности радиальных автомобильных дорог. |
| TAX | full-value property-tax rate per $10,000 | Полная ставка налога на имущество. |
| PTRATIO | pupil-teacher ratio by town | Соотношение учеников и учителей. |
| B | 1000(Bk - 0.63)^2 where Bk is the proportion of black people by town | 1000 (Bk - 0,63) ^ 2, где Bk - доля чернокожих по городам. |
| LSTAT | % lower status of the population | Процент населения с более низким статусом. |
| MEDV | Median value of owner-occupied homes in $1000's | Средняя стоимость домов, занимаемых владельцами, оценивается в 1000 долларов США. |

**Посмотрим на структуру данных:**

print(boston.DESCR)

Number of Attributes: 13 numeric/categorical predictive. Median Value (attribute 14) is usually the target.

Количество атрибутов: 13 числовых / категориальных прогнозов. Среднее значение (атрибут 14) обычно является целевым.

**Разведывательный анализ**

Преобразуем наши данные в датафрейм:

boston\_df = pd.DataFrame(boston.data, columns=boston.feature\_names)

Присоединим столбец с целевой переменной:

boston\_df['MEDV'] = boston.target

Для оптимальной визуализации определим размер датафрейма:

print(boston\_df.shape)

По результатам выберем оптимальное разрешение и выведем датафрейм для просмотра:

pd.set\_option('display.max\_columns',25)  
pd.set\_option('display.width',550)  
print(boston\_df)

Визуализируем матрицу корреляций в виде тепловой карты:

cols = ['LSTAT', 'INDUS', 'NOX', 'RM', 'MEDV']  
hm = sns.heatmap(boston\_df[cols].corr(), cbar=True, annot=True)  
print(hm)  
plt.show()

**Определим зависимую и независимые переменные**

X = boston\_df[['LSTAT']].values

y = boston\_df['MEDV'].values

from sklearn.linear\_model import LinearRegression  
slr = LinearRegression()  
slr.fit(X, y)  
y\_pred = slr.predict(X)  
  
print('Slope: {:.2f}'.format(slr.coef\_[0]))  
print('Intercept: {:.2f}'.format(slr.intercept\_))

**Визуализация**

plt.scatter(X, y)  
res=plt.plot(X, slr.predict(X), color='red', linewidth=2);  
print(res)  
plt.show()

а можно и так:

res=sns.regplot(x="LSTAT", y="MEDV", data=boston\_df)

print(res)  
plt.show()

Работаем с данными StudentsPerformance

Загрузим и посмотрим на них:

st\_per=pd.read\_csv('C:/Users/Ольга/Documents/МИСиС/4 курс/ПСА(new)/Данные для исследования/StudentsPerformance.csv')

pd.set\_option('display.max\_columns',10) # эта строка необходима для того, чтобы выводились все столбики

print(st\_per.head()) # первые 6 строк

print(st\_per.tail()) # последние 6 строк

print(st\_per.dtypes) # типы данных

print(st\_per.describe()) # описательные статистики для числовых переменных  
  
print(st\_per.groupby('gender').aggregate({'writing\_score':'mean'}))# среднее значение по группе

plt.show()

Как отобрать нужные наблюдения.

В Pandas необходимо указывать способ, по которому мы хотим обратиться к колонкам и строкам.

1. Можно обращаться по порядковым номерам.

print(st\_per.iloc[0:5,0:3]) # первые 5 строк и первые 3 колонки

print(st\_per.iloc[[0,3,10],[-1,-2,-3]]) # выбираем не подряд идущие строки (и три последних сторлбца)

То есть, если мы хотим получить не подряд идущие строки (колонки), мы оборачиваем их в список!

1. Можно использовать имена колонок (название переменных).

Чтобы обращаться к строкам по имени

st\_per\_with\_names = st\_per.iloc**[[0,3,4,7,8]]** # для вывода строк мы ставим индексы на место строк

Можно было сделать и так:

st\_per\_with\_names = st\_per.iloc**[[0,3,4,7,8],: ]**

Изменим индексы строк на более содержательные:

st\_per\_with\_names.index=['Cersei','Tywin','Gregor','Joffrey','Ilyn Payne']

Теперь мы можем отбирать интересующие нас данные по лейблам, то есть по именам строк и столбцов:

print(st\_per\_with\_names.loc[['Cersei','Gregor']])# выводим наблюдения, относящиеся к Cersei и Gregor.

print(st\_per\_with\_names.loc[['Cersei','Gregor'],['gender','writing\_score']])

Особенности работы с участками датафрейма

Важно понимать, что датафрейм есть объединение столбцов, каждый из которых является пандовской серией. То есть, можно собрать датафрейм из столбиков как словарь , где ключ – название колонки.

print(st\_per.iloc[[0,3,10],[-1,-2,-3]]) # датафрейм

print(st\_per.iloc[0:5,0:3])# серия

print(type(st\_per**['writing\_score']**))



print(type(st\_per**[['writing\_score']]**))



Фильтрация данных

Как отбирать данные по конкретному условию:

print(st\_per[[**'gender',****'writing\_score'**]]) ])# отобрали все строки с данными только для 'writing\_score' и для девушек, и для юношей

print(st\_per.loc[**st\_per.gender=='female'**])# отобрали все строки с данными только для девушек

print(st\_per.loc[st\_per.gender=='female',[**'gender','writing\_score'**]]) # отобрали столбец с оценкой за чтение и только для девушек

Как вычислять статистики:

print(st\_per.loc[st\_per.gender=='female',['gender','writing\_score']].**mean()**)

Можно использовать библиотеку statsmodels.