

作业 1 数据预处理和线性回归

作者: Zezhong 联系: lightac@mail.ustc.edu.cn

1. 现已使用 Pandas 读取数据集 challenge.csv
 - 请提取该数据集的字段名称, 将结果存为 cols
 - 请获取给数据的字段和样本数量, 将结果分别存为 col_num 和 sam_num
 - 请获取该数据集的前五行记录, 将最后的 DataFrame 存为 five_data

答:

开始答题:

```
import pandas as pd
titanic = pd.read_csv("challenge.csv")
# 获取字段名称
cols = titanic.columns.values
# 获取字段数量
col_num = len(cols)
# 获取样本数量
sam_num = len(titanic)
# 获取样本前 5 行样本
five_data = titanic.head(5)
```

```
[8] import pandas as pd
titanic = pd.read_csv("/content/sample_data/california_housing_test.csv")

cols = titanic.columns
print(cols)

[10] col_num = len(cols)
print(col_num)

9

[13] sam_num = len(titanic)
print(sam_num)

3000

five_data = titanic.head(5)
print(five_data)
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
0	-122.05	37.37	27.0	3885.0	661.0	
1	-118.30	34.26	43.0	1510.0	310.0	
2	-117.81	33.78	27.0	3589.0	507.0	
3	-118.36	33.82	28.0	67.0	15.0	
4	-119.67	36.33	19.0	1241.0	244.0	

	population	households	median_income	median_house_value
0	1537.0	606.0	6.6085	344700.0
1	809.0	277.0	3.5990	176500.0
2	1484.0	495.0	5.7934	270500.0
3	49.0	11.0	6.1359	330000.0
4	850.0	237.0	2.9375	81700.0

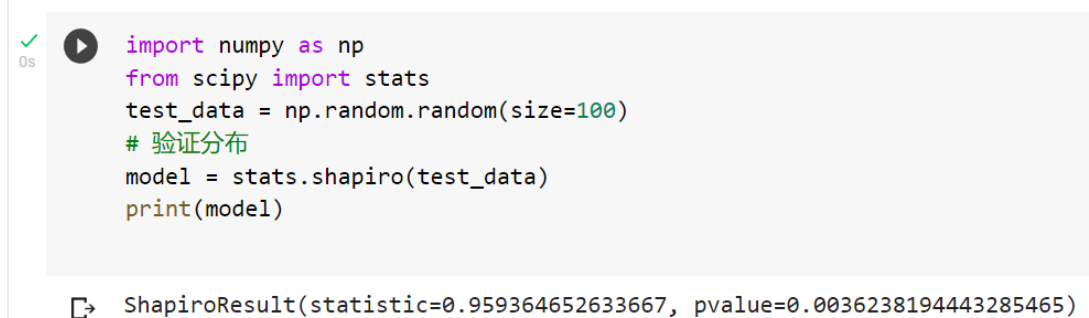
图 1 第一题实验运行结果

2. 现已使用 Numpy 生成服从均匀分布的一维数据集，样本容量为 100;
- 使用 scipy 库中的 stats 模块，对生成的数据进行正态性检验，将检验的结果存为 model

答：

开始答题：

```
import numpy as np
from scipy import stats #题目中此处出现错误，原来为 from scipy.stats import stats
test_data = np.random.random(size=100)
# 验证分布
model = stats.shapiro(test_data)
print(model)
```

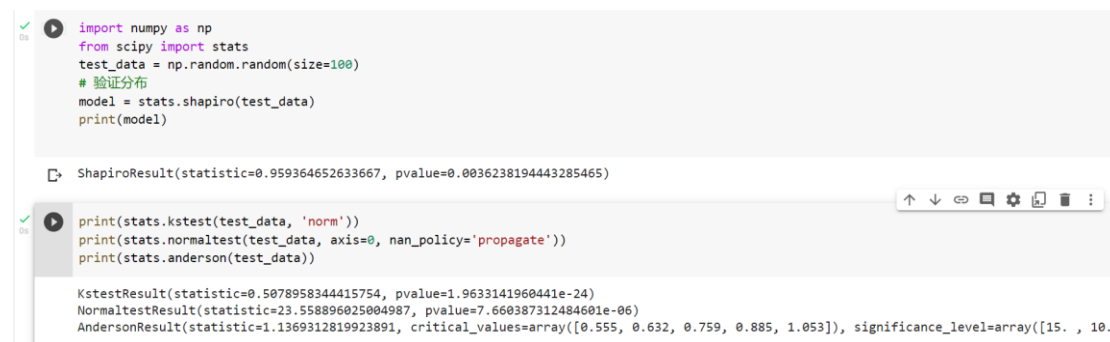


```
import numpy as np
from scipy import stats
test_data = np.random.random(size=100)
# 验证分布
model = stats.shapiro(test_data)
print(model)
```

ShapiroResult(statistic=0.959364652633667, pvalue=0.0036238194443285465)

图 2 作业 2 实验结果 A

除了 shapiro 方法，本作业比较了其它方法包括：anderson、normaltest 和 kstest。实验结果如图 3 所示。



```
import numpy as np
from scipy import stats
test_data = np.random.random(size=100)
# 验证分布
model = stats.shapiro(test_data)
print(model)
```

ShapiroResult(statistic=0.959364652633667, pvalue=0.0036238194443285465)

```
print(stats.kstest(test_data, 'norm'))
print(stats.normaltest(test_data, axis=0, nan_policy='propagate'))
print(stats.anderson(test_data))
```

KstestResult(statistic=0.5078958344415754, pvalue=1.9633141960441e-24)
NormaltestResult(statistic=23.558896025004987, pvalue=7.660387312484601e-06)
AndersonResult(statistic=1.1369312819923891, critical_values=array([0.555, 0.632, 0.759, 0.885, 1.053]), significance_level=array([15., 10., 5., 1., 0.5]))

图 3 作业 2 实验结果 B

3. 下列属于衡量数据整体散度的是（可多选）： a. 欧式距离 b. 标准差 c. 分位数 d. 众数
- 答：b c


4. 现已使用 Pandas 生成 Series 对象 example_data

- 请使用 isnull()函数确定 example_data 是否含有缺失值，将最后的结果存为 boolean_array
- 请使用 fillna()函数使用字符串 missing 替换缺失值，将替换后的 Series 对象存为 new_data

答：

开始答题：

```
import pandas as pd
import numpy as np
example_data = pd.Series([1,2,3,np.nan,4])
# 判断是否含有缺失值
boolean_array = pd.isnull(example_data)
print(boolean_array)
# 缺失值替换
new_data = example_data.fillna('null')
print(new_data)
```

```
✓ 0s  import pandas as pd
import numpy as np
example_data = pd.Series([1,2,3,np.nan,4])
# 判断是否含有缺失值
boolean_array = pd.isnull(example_data)
print(boolean_array)
# 缺失值替换
new_data = example_data.fillna('null')
print(new_data)
```

```
0    False
1    False
2    False
3     True
4    False
dtype: bool
0     1.0
1     2.0
2     3.0
3    null
4     4.0
dtype: object
```

图 4 第 4 题实验结果

5. 现已使用 Pandas 读取数据集 birthrate.csv

- 请对该数据集的 birth_rates 特征使用四分位数作为切分点，通过 qcut()函数完成等频离散化；将最后的结果存为 data_qcut

该数据集详情为：

	country	birth_rates	per_capita_income	proportion_of_population_farming	infant_mortality
0	Venezuela	46.4	392	0.40	68.5
1	Mexico	45.7	110	0.61	87.8
2	Ecuador	45.3	44	0.53	115.8
3	Colombia	38.6	158	0.53	106.8
4	Ceylon	37.2	81	0.53	71.6

开始答题:

```
import pandas as pd
```

```
data = pd.read_csv('birthrate.csv')
```

```
data_qcut = pd.qcut(data['birth_rates'], 4)
```

```
print(data_qcut)
```

```
import pandas as pd
data = pd.read_csv('/content/sample_data/birthrate.csv')

data_qcut = pd.qcut(data['birth_rates'], 4)
print(data_qcut)
```

0 (45.7, 46.4]
1 (45.3, 45.7]
2 (38.6, 45.3]
3 (37.199000000000005, 38.6]
4 (37.199000000000005, 38.6]
Name: birth_rates, dtype: category
Categories (4, interval[float64, right]): [(37.199000000000005, 38.6] < (38.6, 45.3] < (45.3, 45.7] < (45.7, 46.4]]

图 5 第五题实验结果

6. [线性回归] 给定数据:

X: 0, 0, 1, 1, 2, 2; Y: 0, 1, 0, 1, 0, 1.

(a) 拟合模型 $Y = a + bX + \varepsilon$ (手算)

(b) 拟合模型 $Y = bX + \varepsilon$ (手算)

答: 计算过程如图 6 所示。由图 7 和图 8 知道实验结果正确。

(a) $Y = a + bx + \varepsilon$
 $L = \sum_{i=1}^6 (y_i - \hat{y}_i)^2 = \sum_{i=1}^6 (y_i - (a + bx_i))^2$
 $\begin{cases} \frac{\partial L}{\partial a} = 0 \\ \frac{\partial L}{\partial b} = 0 \end{cases} \Rightarrow \begin{cases} 2 \sum_{i=1}^6 (y_i - a - bx_i) = 0 \\ 2 \sum_{i=1}^6 (y_i - a - bx_i)x_i = 0 \end{cases}$
解得: $b=0$ $a=0.5$

(b) $Y = bx + \varepsilon$
 $L = \sum_{i=1}^6 (y_i - bx_i)^2$
 $\frac{\partial L}{\partial b} = 0 \Rightarrow 2 \sum_{i=1}^6 (y_i - bx_i)x_i = 0$
解得: $b=0.3$

图 6 第 6 题计算结果



图 7 第 6 题实验结果 A



图 8 第 6 题实验结果 B

7. [线性回归] 给定数据:

X: 0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10,

10, 11, 11, 12, 12

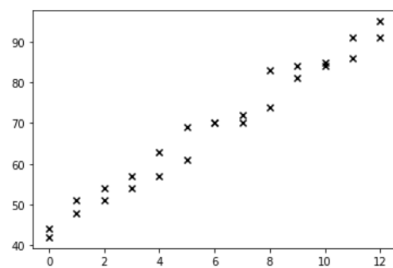
Y: 42, 44, 51, 48, 51, 54, 57, 54, 57, 63, 61, 69, 70, 70, 70, 72,

74, 83, 84, 81, 84, 85, 91, 86, 91, 95

写程序拟合模型 $Y = a + bX + \varepsilon$, 并画图显示数据点和拟合曲线。

答:

```
import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
x = np.array([0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10, 10, 11, 11, 12, 12])
x = np.reshape(x, newshape=(len(x), 1))
y = np.array([42, 44, 51, 48, 51, 54, 57, 54, 57, 63, 61, 69, 70, 70, 70, 72, 74, 83, 84, 81, 84, 85, 91, 86, 91, 95])
y = np.reshape(y, newshape=(len(y), 1))
plt.scatter(x, y, color="black", marker="x")
plt.show()
```



图表 9 第 7 题实验结果 A

✓
1s

```
model = LinearRegression()
model.fit(x, y)
plt.scatter(x, y, color="black", marker="x")
plt.plot(x_data, model.predict(x), "red")
plt.show()
```

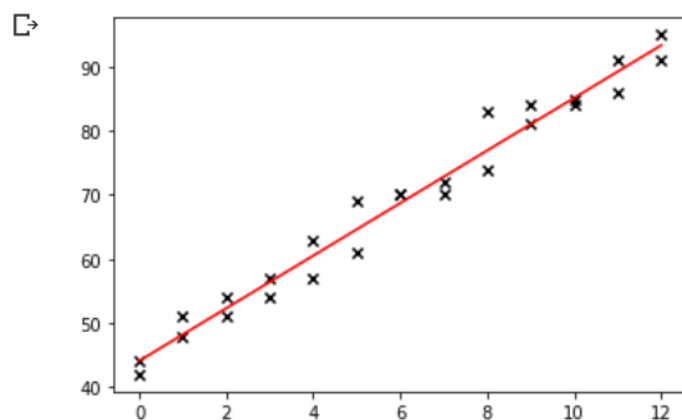


图 10 第 7 题实验结果 B

```
0s print("b: " + str(model.coef_[0]))
print("a: " + str(model.intercept_))

b: [4.1043956]
a: [44.1043956]
```

图 11 第 7 题实验结果 C

8. 给定 $f(x) = x^3 - 6x^2 + 11x - 6$, 编程实现梯度下降法计算出使 $f(x)=0$ 的解, 绘图展示梯度下降法的迭代过程。

答:

迭代轮数为: 106 轮

求得的解为: 1.002

```
import numpy as np
import matplotlib.pyplot as plt
def f(x):
    return x**3 - 6*x**2 + 11*x - 6
def lossf(x):
    return (f(x) - 0)**2
def dlossf(x):
    delta = 0.0000001
    return (lossf(x+delta) - lossf(x-delta))/(2.0*delta)
alpha = 0.01
new_x = []
new_y = []
x = 0.0
iteration = 0
f1 = f2 = f(x)
X = []
while iteration <= 1000 and lossf(x) >= 1e-5:
    iteration += 1
    x = x - alpha * dlossf(x)
    f2 = abs(f1 - f(x))
    f1 = f(x)
    X.append(x)
    if(iteration % 20 == 0):
        new_x.append(x)
        new_y.append(f1)
print(new_x[-1])
print(f(new_x[-1]))

print(iteration)

1.0025581341769498
0.005096652943056945
106
```

图表 12 第 8 题代码及其运行结果

```

X = np.arange(0.95,1.35,0.00001)
Y = f(X)
Y = np.array(Y)
plt.plot(X,Y)
plt.scatter(new_x,new_y,color='red')
plt.title("$y = x^3 - 6x^2 + 11x - 6$")
for i,txt in enumerate(range(1, len(new_x) + 1)):
    plt.annotate(txt,(new_x[i],new_y[i]))
plt.show()

```

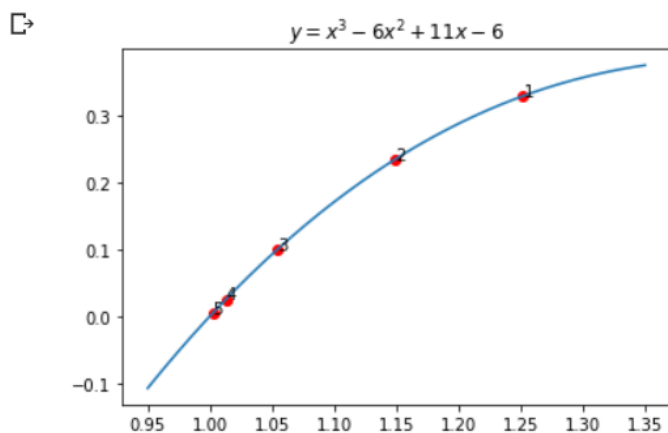
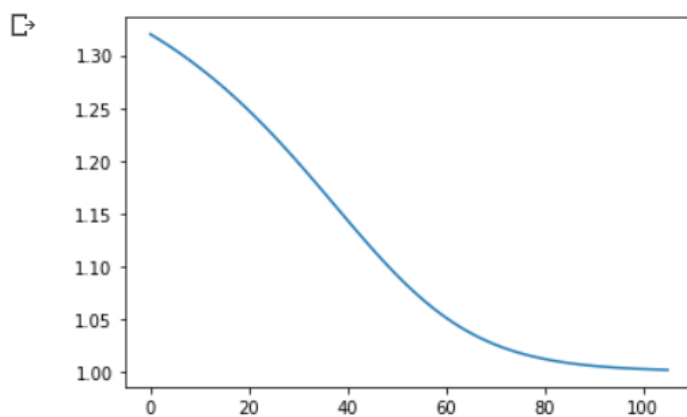


图 13 第八题迭代过程示意图 A

```

plt.plot(X)
plt.show()

```



图表 14 第 8 题迭代过程示意图 B

9. [自学牛顿方法] 牛顿方法和梯度下降法有什么异同点？请写出牛顿方法的推导过程，编程实现牛顿方法求解上一题，并编程绘图展示迭代计算过程。

答：


```

import numpy as np
import matplotlib.pyplot as plt

def f(x):
    return x**3 - 6*x**2 + 11*x - 6
def df(x):
    return 3*x**2 - 12*x + 11
# alpha = 0.01
new_x = []
new_y = []
x = 0.5
x0 = x
x = x0 - f(x0) / df(x0)
iteration = 0
# f1 = f2 = f(x)
while abs(x - x0) >= 1e-10:
    iteration += 1
    x0 = x
    x = x0 - f(x0) / df(x0)
    new_x.append(x)
    new_y.append(f(x))
print(x)
print(f(x))
print(iteration)

```

1.0
0.0
5

图 15 第 9 题代码及其实验结果

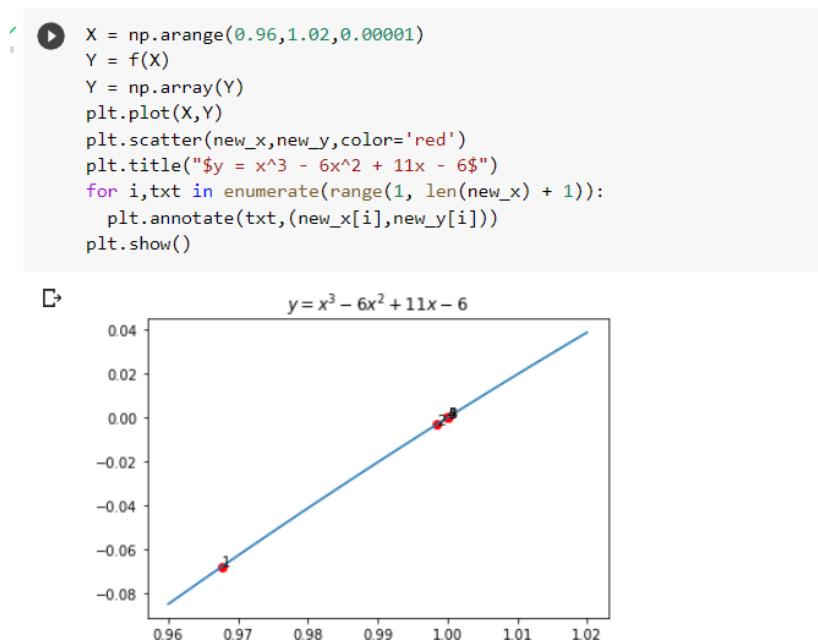


图 16 第九题迭代过程示意图 A

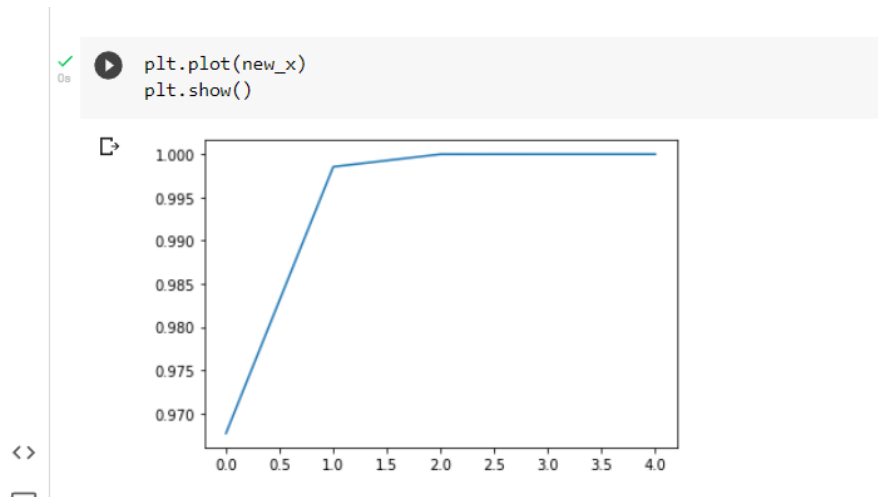


图 17 第九题迭代过程示意图 B

10.数据标准化是将数据按比例缩放到一个特定区间，其主要包括数据同趋化处理和无量纲化处理两个方面。数据标准化的方法有很多种，常用的有最小最大标准化和 z -score 标准化。

请用户对本题中的变量(不包括变量 ID)进行 z -score 标准化

数据说明：本题数据来自 KEEL，数据集一共包含 1 列 ID，4 列特征变量，共 100 个样本点。

列名	类型	说明	示例
ID	Int	ID 样本号	1
CT	Float	Cement 黏固粉	295.7
FA	Float	FlyAsh 粉煤灰	98.8
WT	Float	Water 水	185.6
SP	Float	SuperPlasticizer 超增塑剂	14.2

预设变量：本题使用的数据变量名、含义及其类型如下：

变量名	含义	类型
data	数据集	DataFrame

开始答题：

```
import pandas as pd
data = pd.read_csv('xxx.csv')
continuous_columns = ['CT','FA','WT','SP']
data[continuous_columns] = data[continuous_columns].apply(lambda x : (x-x.mean())/x.std())
print(data.head())
```