上机操作实例 16July

一、软件安装

安装python和任意的开发IDE

在 terminal 中输入

pip install sklearn –i https://pypi.douban.com/simple/ pip install pandas –i https://pypi.douban.com/simple/ pip install matplotlib –i https://pypi.douban.com/simple/ pip install gplearn –i https://pypi.douban.com/simple/ pip install sympy –i https://pypi.douban.com/simple/

二、最小二线作业讲解

1、导入所需的算法包

In [10]:

```
import numpy as np # 处理数据
import pandas as pd # 处理表格
import matplotlib.pyplot as plt #绘图工具
from sklearn.linear_model import LinearRegression as LR # 最小二乘回归算法
```

2、转化数据类型/导入数据

In [3]:

```
def xlsx_to_csv_pd():
    data_xls = pd.read_excel('example1.xlsx', index_col=0)
    data_xls.to_csv('example1.csv', encoding='utf-8')


if __name__ == '__main__':
    xlsx_to_csv_pd()
```

In [4]:

```
data = pd.read_csv(r".\example1.csv")
```

In [17]:

```
X=data.iloc[:,:-1]
Y=data.iloc[:,-1]
```

In [18]:

```
X
```

Out[18]:

	x
0	25
1	50
2	70
3	80

In [19]:

```
Y
```

Out[19]:

```
0    4.11
1    4.92
2    6.10
3    6.66
Name: y, dtype: float64
```

3、数据处理

In [20]:

```
Y=np.log(Y)
X=1/(X+273.15)
```

In [22]:

```
X
```

Out[22]:

	x
0	0.003354
1	0.003095
2	0.002914
3	0.002832

In [23]:

```
Y
```

Out[23]:

```
0    1.413423
1    1.593309
2    1.808289
3    1.896119
Name: y, dtype: float64
```

4、最小二乘回归

In [25]:

```
reg = LR().fit(X, Y)
```

In [26]:

```
print(reg.coef_)
print(reg.intercept_)
```

```
[-931.51736703]
4.517606466278733
```

In [27]:

```
reg.score(X,Y)
```

Out[27]:

```
0.982817843764725
```

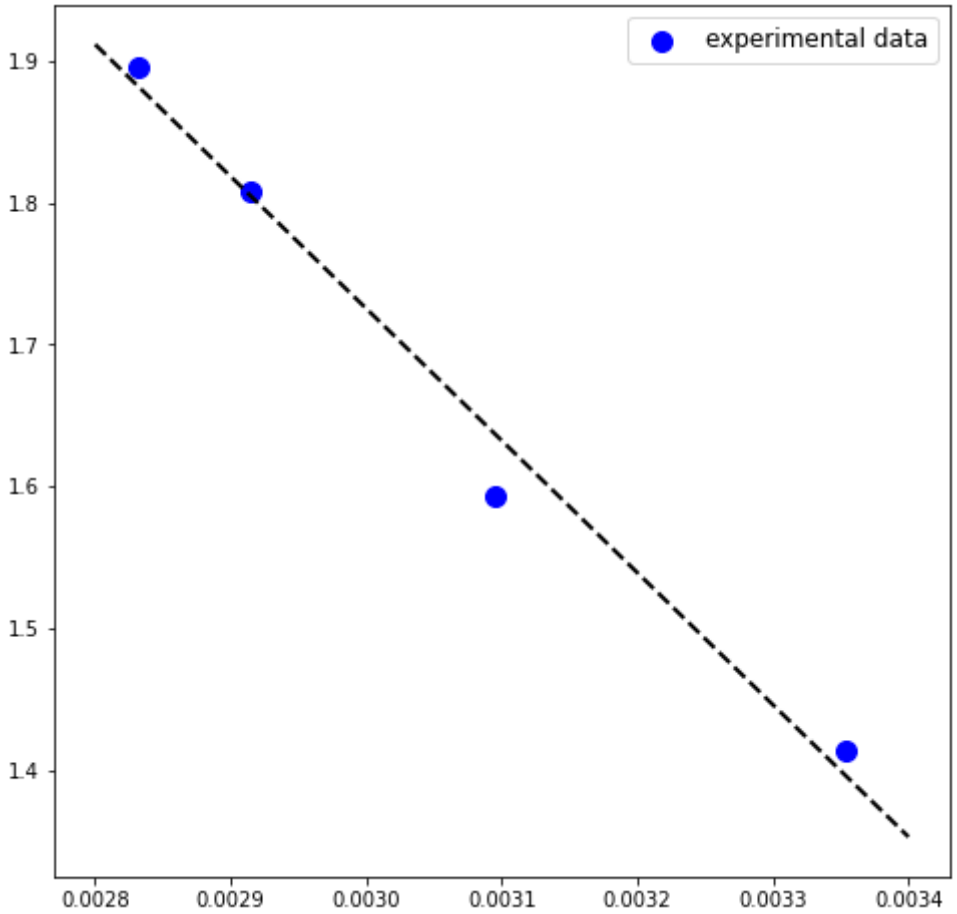
5、利用matplotlib画图

In [36]:

```
fig = plt.figure(figsize=[8,8])# 设置画布大小
plt.scatter(X,Y,c='b',s=100,marker='o',label='experimental data')#画实验数据点
x = np.linspace(0.0028, 0.0034, 50)
y=-931.52*x+4.52   # 最小二乘拟合得到的公式
plt.plot(x, y, color='k', linewidth=2.0, linestyle='--')#画出拟合直线
plt.legend(fontsize=12)
```

Out[36]:

<matplotlib.legend.Legend at 0x1661f232148>



三、logistic regression 实例

1、导入逻辑斯特回归算法

In [38]:

```
from sklearn.linear_model import LogisticRegression as LR
```

2、转化数据类型/导入数据

In [40]:

```
def xlsx_to_csv_pd():
    data_xls = pd.read_excel('example2.xlsx', index_col=0)
    data_xls.to_csv('example2.csv', encoding='utf-8')


if __name__ == '__main__':
    xlsx_to_csv_pd()
```

In [42]:

```
data1 = pd.read_csv(r".\example2.csv")
```

In [64]:

```
data1
```

Out[64]:

	Hmix	Smix	Phase
0	-5.09	5.22	1
1	-1.86	3.80	1
2	-3.49	4.31	1
3	-11.44	7.84	1
4	-5.00	4.31	1
5	-3.37	4.97	1
6	-4.02	5.48	1
7	-2.47	4.86	1
8	2.20	4.30	1
9	-29.15	7.63	1
10	-1.43	5.22	1
11	-2.30	5.48	1
12	-3.15	5.72	1
13	-18.40	7.90	1
14	-25.87	9.29	0
15	-25.18	9.41	0
16	-24.97	10.85	0
17	-40.10	10.52	0
18	-41.16	10.17	0
19	-33.35	13.66	0
20	-27.50	10.12	0
21	-27.15	9.93	0
22	-26.89	9.48	0
23	-32.22	8.39	0

	Hmix	Smix	Phase
24	-28.10	9.99	0
25	-30.46	10.39	0
26	-25.48	9.39	0
27	-25.22	10.49	0
28	-27.31	10.06	0
29	-35.59	10.70	0
30	-25.46	9.43	0
31	-25.47	10.04	0
32	-26.66	9.99	0
33	-28.58	10.70	0

In [43]:

```
X1=data.iloc[:,:-1]
Y1=data.iloc[:,-1]
```

In [44]:

```
X1
```

Out[44]:

	Hmix	Smix
0	-5.09	5.22
1	-1.86	3.80
2	-3.49	4.31
3	-11.44	7.84
4	-5.00	4.31
5	-3.37	4.97
6	-4.02	5.48
7	-2.47	4.86
8	2.20	4.30
9	-29.15	7.63
10	-1.43	5.22
11	-2.30	5.48
12	-3.15	5.72
13	-18.40	7.90
14	-25.87	9.29
15	-25.18	9.41
16	-24.97	10.85
17	-40.10	10.52
18	-41.16	10.17
19	-33.35	13.66
20	-27.50	10.12
21	-27.15	9.93
22	-26.89	9.48
23	-32.22	8.39

	Hmix	Smix
24	-28.10	9.99
25	-30.46	10.39
26	-25.48	9.39
27	-25.22	10.49
28	-27.31	10.06
29	-35.59	10.70
30	-25.46	9.43
31	-25.47	10.04
32	-26.66	9.99
33	-28.58	10.70

In [45]:

```
Y1
```

Out[45]:

```
0     1
1     1
2     1
3     1
4     1
5     1
6     1
7     1
8     1
9     1
10    1
11    1
12    1
13    1
14    0
15    0
16    0
17    0
18    0
19    0
20    0
21    0
22    0
23    0
24    0
25    0
26    0
27    0
28    0
29    0
30    0
31    0
32    0
33    0
Name: Phase, dtype: int64
```

2、模型实例化

In [61]:

```
lr = LR(C=1000,max_iter=1000).fit(X1,Y1)
```

In [62]:

```
lr.score(X1, Y1)
```

Out[62]:

1.0

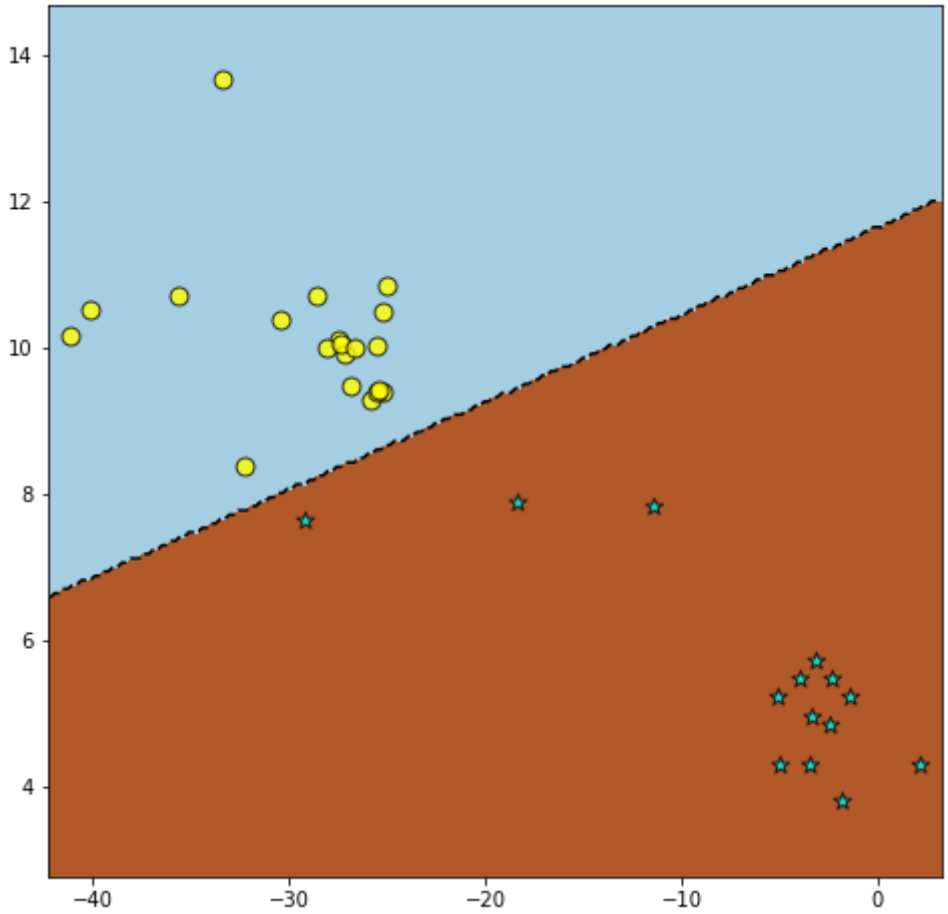
3、可视化

In [69]:

```
fig = plt.figure(figsize=[8,8]) # 设置画布大小



x_min, x_max = X1.iloc[:, 0].min() - 1, X1.iloc[:, 0].max() + 1
y_min, y_max = X1.iloc[:, 1].min() - 1, X1.iloc[:, 1].max() + 1
XX, YY = np.mgrid[x_min:x_max:200j, y_min:y_max:200j]
# 画布内铺设网格点

Z = lr.predict(np.c_[XX.ravel(), YY.ravel()]).reshape(XX.shape)-0.5
plt.pcolormesh(XX, YY, Z > 0, shading='auto',cmap=plt.cm.Paired)
plt.contour(XX, YY, Z, colors=['k'], linestyles=['--'],levels=[0])
# 画分类面



plt.scatter(X1.iloc[0:14, 0], X1.iloc[0:14, 1], marker='*',c='cyan',s=80,edgecolors='k',alpha=0.8,label='BCC')
plt.scatter(X1.iloc[14:36, 0], X1.iloc[14:36, 1], marker='o',c='yellow',s=80,edgecolors='k',alpha=0.8,label='FCC')
#画实验数据点
```

Out[69]:

<matplotlib.collections.PathCollection at 0x16620a45608>



练习要求

1、利用example1中的数据尝试lammda=0.1,1和10三种情况下LASSO的回归结果。并尝试作图

reference:https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html?highlight=lasso#sklearn.linear_model.Lasso (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html?highlight=lasso#sklearn.linear_model.Lasso)

2、利用感知机将example2中的数据完成分类，并画图

reference:https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html?highlight=perceptron#sklearn.linear_model.Perceptron (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html?highlight=perceptron#sklearn.linear_model.Perceptron)

。。

3、完成以上作业的同学尝试适用example3.csv文件中的数据，使用遗传算法找到最合适数据的公式

reference:https://gplearn.readthedocs.io/en/stable/examples.html (https://gplearn.readthedocs.io/en/stable/examples.html)

In [ ]:

```

```