Homework 2

模式识别基础

第二次作业

陈翰墨 自65 2016010302

2019.3.22

Problem 1

Fisher 准则的最小二乘法推导

(1)

$$\frac{\partial E}{\partial \omega_0} = \sum_{i=1}^{n}(\omega_0 + \omega^T x_i - t_i) = 0 \tag{1}$$

又 $\sum\limits_{i=1}^{n} t_i = n_1 \times \frac{n}{n_1} - n_2 \times \frac{n}{n_2} = 0$

故 $n\omega_0 + \omega^T \sum\limits_{i=1}^{n} x_i = 0$，亦即

$$\omega_0 = -\omega^T m, \text{where } m = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2}$$

(2)

代入 $\omega_0 = -\omega^T m$

$$E = \frac{1}{2}[\sum_{i \in C_1}(\omega^T(x_i - m) - \frac{n}{n_1})^2 + \sum_{i \in C_2}(\omega^T(x_i - m) + \frac{n}{n_2})^2] \tag{3}$$

令

$$\frac{\partial E}{\partial \omega} = 0 \tag{4}$$

得到

$$\sum_{i \in C_1}(x_i - m)[(x_i - m)^T\omega - \frac{n}{n_1}] + \sum_{i \in C_2}(x_i - m)[(x_i - m)^T\omega + \frac{n}{n_2}] = 0 \tag{5}$$

$$\begin{aligned} \left[\sum_{i=1}^{n}(x_i - m)(x_i - m)^T\right]\omega &= \frac{n}{n_1}\sum_{i \in C_1}(x_i - m) - \frac{n}{n_2}\sum_{i \in C_2}(x_i - m) \\ &= n(m_1 - m) - n(m_2 - m) = n(m_1 - m_2) \end{aligned}$$

其中 $S_T = \sum_{i=1}^{n}(x_i - m)(x_i - m)^T$ 为总方差矩阵

即要证明

$$S_T = S_w + \frac{n_1 n_2}{n} S_B \tag{6}$$

而

$$S_T = \sum_{i=1}^{n}(x_i - m)(x_i - m)^T = \sum_{i=1}^{n}(xx^T - mm^T) \tag{7}$$

$$S_w + \frac{n_1 n_2}{n} S_B = \sum_{i=1}^{n} xx^T - n_1 m_1 m_1^T - n_2 m_2 m_2^T + \frac{n_1 n_2}{n}(m_2 - m_1)(m_2 - m_1)^T$$

转化为证明

$$nmm^T = n_1 m_1 m_1^T + n_2 m_2 m_2^T - \frac{n_1 n_2}{n}(m_2 - m_1)(m_2 - m_1)^T \tag{8}$$

代入$m = \frac{n_1}{n} m_1 + \frac{n_2}{n} m_2$ 即得到上式。

(3)

由

$$(S_\omega + \frac{n_1 n_2}{n} S_B)\omega = n(m_1 - m_2) \tag{9}$$

得到

$$S_\omega \omega = (\frac{n_1 n_2}{n}(m_1 - m_2)^T \omega + n)(m_1 - m_2) \tag{10}$$

其中 $\frac{n_1 n_2}{n}(m_1 - m_2)^T \omega + n$ 是标量不影响ω的方向

从而得到

$$\omega \propto S_w^{-1}(m_1 - m_2) \tag{11}$$

Problem 2

Logistic Regression

Code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

# Data Cleaning
df = pd.read_csv("breast-cancer-wisconsin.txt", header=None, sep="\t")
df = df[df[6] != '?']
df[6]=df[6].astype('int64')
df.to_csv("data.txt", sep="\t", index=False, header=False)
df=df.values;

# Spilt train set and test set
X=df[:,range(1,10)]
y=df[:,10]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)

# Standardization
sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
```

```python
X_test_std = sc.transform(X_test)

# Logistic Regression
lr = LogisticRegression()
lr.fit(X_train_std, y_train)
y_pred = lr.predict(X_test_std)

# Model Checking
print(classification_report(y_test, y_pred))
```

Result

```
              precision    recall  f1-score   support
           0       0.94      0.99      0.96       103
           1       0.98      0.90      0.94        68
   micro avg       0.95      0.95      0.95       171
   macro avg       0.96      0.94      0.95       171
weighted avg       0.95      0.95      0.95       171
```

Fisher's Linear Discriminant

Code

```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report

# Data Cleaning
df = pd.read_csv("breast-cancer-wisconsin.txt", header=None, sep="\t")
df = df[df[6] != '?']
df[6]=df[6].astype('int64')
df.to_csv("data.txt", sep="\t", index=False, header=False)
df=df.values;

```

```
# Spilt train set and test set
X=df[:,range(1,10)]
y=df[:,10]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=42)


# Standardization
sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
X_test_std = sc.transform(X_test)

# Sort
X_train_good = X_train_std[y_train==1]
X_train_bad = X_train_std[y_train==0]

# Calculate the mean vector
mean1 = np.mean(X_train_good, axis=0)
mean0 = np.mean(X_train_bad,axis=0)

# Calculate SS within classes

SS_1=0

for i in range(X_train_good.shape[0]):
    x=X_train_good[i,:]-mean1
    SS_1 += np.dot(x.reshape(9,1),x.reshape(1,9))



SS_2=0

for i in range(X_train_bad.shape[0]):
    x = X_train_bad[i, :] - mean0
    SS_1 += np.dot(x.reshape(9, 1), x.reshape(1, 9))



SS_within=SS_1+SS_2

w= np.linalg.inv(SS_within).dot(mean1-mean0)

w0 = w.dot(mean0+mean1)/2
#w0 =
w.dot(X_train_bad.shape[0]*mean0+X_train_good.shape[0]*mean1)/(X_train_bad.shape[
0]+X_train_good.shape[0])

y_pred=np.zeros(X_test_std.shape[0])

for i in range(X_test_std.shape[0]):
```

```
    x= X_test_std[i,:]
    if ( np.dot(x,w) > w0 ):
        y_pred[i] = 1
    else:y_pred[i] = 0

print(classification_report(y_test, y_pred))
```

Result

```
              precision    recall  f1-score   support

           0       0.92      0.99      0.95       103
           1       0.98      0.87      0.92        68
   micro avg       0.94      0.94      0.94       171
   macro avg       0.95      0.93      0.94       171
weighted avg       0.94      0.94      0.94       171
```

Problem 3

(1)

非线性分类器。

原因：不同分类的边界不是线性的超平面，而是由 Sigmoid 函数定义的空间曲面。

(2)

Code

```
from skimage import io
import numpy as np
import glob
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

```

```python

n=2
imgCnt=0
data=np.zeros([3000,2305])

nums = np.random.choice(10,n)
for i in nums:
    imglist=glob.glob("./Pictures/"+i.astype('str')+"/*.png")
    for imgpath in imglist:
        img = io.imread(imgpath, as_gray=True)
        data[imgCnt,range(2304)] =img.reshape(2304);
        data[imgCnt,2304] = i;
        imgCnt +=1

data=data[range(imgCnt),:]

X=data[:,range(2304)]
y=data[:,2304]

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.25,random_state=42)

# Standardization
sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
X_test_std = sc.transform(X_test)

# Softmax Regression
lr = LogisticRegression(solver='newton-cg',multi_class='multinomial')
lr.fit(X_train_std, y_train)
y_pred = lr.predict(X_test_std)

# Model Checking
print(classification_report(y_test, y_pred))
```

$n = 2$

	precision	recall	f1-score	support
4.0	0.99	0.93	0.96	82
6.0	0.92	0.99	0.95	68
micro avg	0.95	0.95	0.95	150
macro avg	0.95	0.96	0.95	150
weighted avg	0.96	0.95	0.95	150

$n = 5$

	precision	recall	f1-score	support
3.0	0.79	0.80	0.79	65
4.0	0.73	0.84	0.78	73
5.0	0.73	0.73	0.73	77
6.0	0.73	0.73	0.73	90
8.0	0.64	0.54	0.59	70
micro avg	0.73	0.73	0.73	375
macro avg	0.73	0.73	0.73	375
weighted avg	0.73	0.73	0.73	375

$n = 7$

```
              precision    recall  f1-score   support
         0.0       0.76      0.80      0.78        69
         1.0       0.82      0.70      0.76        94
         2.0       0.59      0.61      0.60        66
         4.0       0.70      0.73      0.72        78
         6.0       0.65      0.64      0.64        75
         7.0       0.59      0.66      0.62        76
         8.0       0.63      0.61      0.62        67
   micro avg       0.68      0.68      0.68       525
   macro avg       0.68      0.68      0.68       525
weighted avg       0.68      0.68      0.68       525
```

$n = 10$

总结

可以发现，随着需要分类的类别数量的增加，分类的准确率逐渐下滑。

猜测可能是随着类别数增加，虽然读取的图像数量增加，但是有效信息(即每个人的特征)并没有增加，而分类的难度增加，因此导致准确率下滑

References

1. [sklearn.linear_model.LogisticRegression — scikit-learn 0.20.3 documentation]
2. [sklearn.metrics.classification_report — scikit-learn 0.20.3 documentation]
3. [fisher判别分析原理+python实现 - PJZero - CSDN博客]