homework1文档

代码结构

| getData.py 自己写的一些获取数据集的函数

| logistic.py 自己写的LogisticRegression类

| part1.py 作业2代码

| part2.py 补充题代码

作业1

上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

理论题 1.

1) $y = e^{wx+b}$ ：最小二乘法求解 w 与 b

有输入 $X = [\vec{x}_1, \vec{x}_2, \vec{x}_3, \vec{x}_4, \cdots, \vec{x}_n]^T$ $X_{n\times l}$ $\vec{x}_i$ $1\times l$

$Y^* = [y_1, y_2, y_3, \cdots, y_n]^T$ $Y^*_{n\times 1}$ $W_{l\times 1}$ $W^T_{1\times l}$

~~$Y = e^{W^T x+b}$~~ $Y = e^{X\cdot W^T+b}$

$\Rightarrow \ln Y = X\cdot W^T + B$ Target $\ln(Y^*)$ $B = \begin{bmatrix} b \\ b \\ b \\ \vdots \\ b \end{bmatrix}_{n\times 1}$

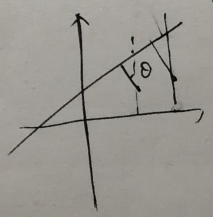
设 $P = \ln Y$ $P^* = \ln Y^*$

$P - XW^T + B = 0$

B为b元拓展矩阵

$$Error = \underset{argmin}{\frac{|P^* - XW^T + B|}{\|W^T\|^2}}$$

又因为



距离与偏差只存在系数关系，可化简为

$Error = \underset{argmin}{\|P^* - XW^T + B\|^2} = \underset{argmin}{(P^* - XW^T + B)^T (P^* - XW^T + B)}$

$\frac{\partial E}{\partial W^T} = 2(-X^T)(P^* - XW^T + B) = 0$ ~~$P^* = X$~~

~~$\frac{\partial E}{\partial W}$~~ $\frac{\partial E}{\partial B} = 2(P^* - XW^T + B)$

把 $\vec{x}_i$ 改为 $\begin{pmatrix}\vec{x}_i \\ 1\end{pmatrix}$ b加入W，改为 $\frac{\partial E}{\partial W'^T} = 2(-X'^T)(P^* - X'W'^T) = 0$

$X'W'^T = P^*$ $\Rightarrow W'^T = (X'^T X')^{-1} X'^T P^*$ 其中 W' 为 $[w_1, w_2, \cdots, w_l, b]$

$\Rightarrow$ ~~$X'^T \cdot X' W'^T = X'^T P^*$~~ $X' = [\vec{x}_1', \vec{x}_2' \cdots \vec{x}_n']^T$

$\Rightarrow W'^T = (X'^T X')^{-1} \cdot X'^T \cdot P^*$ $\vec{x}_i' = [x_{i1}, x_{i2}, x_{i3}, \cdots x_{il}, 1]$

$P^* = \ln(Y^*)$

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

2. A, B, C工厂

	产品占比	次品率
A	0.35 = P(A)	0.015 = P(次|A)
B	0.35 = P(B)	0.010 = P(次|B)
C	0.30 = P(C)	0.020 = P(次|C)

1) 随机抽一个为次品的概率

$$P(次) = P(A)\cdot P(次|A) + P(B)P(次|B) + P(C)P(次|C)$$
$$= 0.35\cdot 0.015 + 0.35\cdot 0.01 + 0.3\times 0.02 = 0.01475$$

(2) 抽一个次品来自

A: $P(A|次) = \dfrac{P(A)P(次|A)}{P(次)} = 0.3559$

B: $P(B|次) = 0.2373$

C: $P(C|次) = 0.4068$

3. ε 的含义

① $\varepsilon = 0$ 样本可在最大边界上及外分类

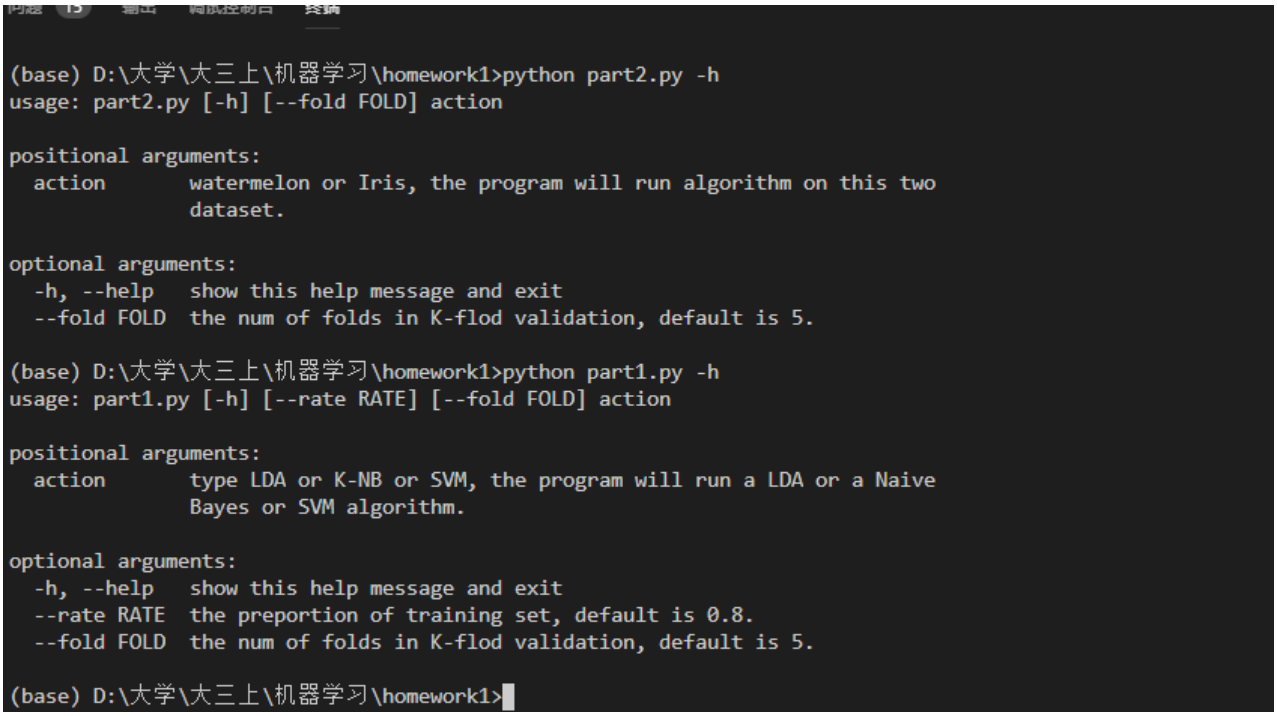
② $0 < \varepsilon \leq 1$ 样本可在最大间隔内分类

③ $\varepsilon > 1$ 样本会被错误分类

作业2

代码使用方法

1. python part1.py -h 可以获得使用帮助



```
(base) D:\大学\大三上\机器学习\homework1>python part2.py -h
usage: part2.py [-h] [--fold FOLD] action

positional arguments:
  action       watermelon or Iris, the program will run algorithm on this two
               dataset.

optional arguments:
  -h, --help   show this help message and exit
  --fold FOLD  the num of folds in K-flod validation, default is 5.

(base) D:\大学\大三上\机器学习\homework1>python part1.py -h
usage: part1.py [-h] [--rate RATE] [--fold FOLD] action

positional arguments:
  action       type LDA or K-NB or SVM, the program will run a LDA or a Naive
               Bayes or SVM algorithm.

optional arguments:
  -h, --help   show this help message and exit
  --rate RATE  the preportion of training set, default is 0.8.
  --fold FOLD  the num of folds in K-flod validation, default is 5.

(base) D:\大学\大三上\机器学习\homework1>
```

2. python part1.py LDA (--rate 0.8) 括号内容可选且值为默认
3. python part1.py K-NB (--fold 5) 括号内容可选且值为默认
4. python part1.py SVM(--rate 0.8) 括号内容可选且值为默认

实验结果

LDA结果

```python
if __name__ == "__main__":
    parse = argparse.ArgumentParser()
    parse.add_argument("action", type=str, help="type LDA or K-NB or SVM, the program will run a LDA o
    parse.add_argument("--rate", type=float, default=0.8, help="the preportion of training set, defaul
    parse.add_argument("--fold", type=int, default=5, help="the num of folds in K-flod validation, def

    watermalon = pd.read_csv("watermalon.csv",engine="python")
    iris = pd.read_csv("iris.data",header=None)

    args = parse.parse_args()

    if args.action=="LDA":
        LDA(watermalon)

    if args.action=="K-NB":
        KNB(watermalon)

    if args.action=="SVM":
        SVM_func(watermalon,'rbf')
        SVM_func(watermalon,'linear')
        SVM_func(watermalon,'poly')
        SVM_func(watermalon,'sigmod')
        SVM_func(watermalon,'precomputed')
```

问题 8　输出　调试控制台　终端

```
the accuracy of SVM using kernal sigmod  scores          0.25
the accuracy of SVM using kernal precomputed  scores     0.25

D:\大学\大三上\机器学习\homework1>python part1.py LDA
using LDA on watermelon3 dataset,training set rate =  0.8   test accuracy  : 0.5

D:\大学\大三上\机器学习\homework1>python part1.py LDA --rate 0.7
using LDA on watermelon3 dataset,training set rate =  0.7   test accuracy  : 0.75

D:\大学\大三上\机器学习\homework1>python part1.py LDA --rate 0.5
using LDA on watermelon3 dataset,training set rate =  0.5   test accuracy  : 1.0

D:\大学\大三上\机器学习\homework1>python part1.py LDA --rate 0.9
using LDA on watermelon3 dataset,training set rate =  0.9   test accuracy  : 1.0

D:\大学\大三上\机器学习\homework1>python part1.py LDA --rate 0.4
using LDA on watermelon3 dataset,training set rate =  0.4   test accuracy  : 0.75

D:\大学\大三上\机器学习\homework1>
```

K折朴素贝叶斯结果

```python
if __name__ == "__main__":
    parse = argparse.ArgumentParser()
    parse.add_argument("action", type=str, help="type LDA or K-NB or SVM, the program will run a LDA or a Naive Bayes or SVM algorithm.")
    parse.add_argument("--rate", type=float, default=0.8, help="the preportion of training set, default is 0.8.")
    parse.add_argument("--fold", type=int, default=5, help="the num of folds in K-flod validation, default is 5.")

    watermalon = pd.read_csv("watermalon.csv",engine="python")
    iris = pd.read_csv("iris.data",header=None)

    args = parse.parse_args()

    if args.action=="LDA":
        LDA(watermalon)

    if args.action=="K-NB":
        KNB(watermalon)

    if args.action=="SVM":
        SVM_func(watermalon,'rbf')
        SVM_func(watermalon,'linear')
        SVM_func(watermalon,'poly')
        SVM_func(watermalon,'sigmod')
        SVM_func(watermalon,'precomputed')
```

问题 8　输出　调试控制台　终端　1: cmd

```
fold  2  accuracy :  0.5
fold  3  accuracy :  0.5
fold  4  accuracy :  0.5

D:\大学\大三上\机器学习\homework1>python part1.py K-NB
fold  0  accuracy :  0.75
fold  1  accuracy :  0.75
fold  2  accuracy :  0.75
fold  3  accuracy :  0.75
fold  4  accuracy :  0.75

D:\大学\大三上\机器学习\homework1>python part1.py K-NB
fold  0  accuracy :  0.75
fold  1  accuracy :  0.75
fold  2  accuracy :  0.75
fold  3  accuracy :  0.75
fold  4  accuracy :  0.75

D:\大学\大三上\机器学习\homework1>
```

行 194，列 27　空格

SVM结果

```python
if __name__ == "__main__":
    parse = argparse.ArgumentParser()
    parse.add_argument("action", type=str, help="type LDA or K-NB or SVM, the program will run a LDA or a Naive Bayes or SVM algorithm.")
    parse.add_argument("--rate", type=float, default=0.8, help="the preportion of training set, default is 0.8.")
    parse.add_argument("--fold", type=int, default=5, help="the num of folds in K-flod validation, default is 5.")

    watermalon = pd.read_csv("watermalon.csv",engine="python")
    iris = pd.read_csv("iris.data",header=None)

    args = parse.parse_args()

    if args.action=="LDA":
        LDA(watermalon)

    if args.action=="K-NB":
        KNB(watermalon)

    if args.action=="SVM":
        SVM_func(watermalon,'rbf')
        SVM_func(watermalon,'linear')
        SVM_func(watermalon,'poly')
        SVM_func(watermalon,'sigmod')
        SVM_func(watermalon,'precomputed')
```

问题 8 输出 调试控制台 终端 1: cmd

```
fold  2  accuracy :  0.75
fold  3  accuracy :  0.75
fold  4  accuracy :  0.75

D:\大学\大三上\机器学习\homework1>python part1.py SVM
the accuracy of SVM using kernal rbf  scores     0.5
the accuracy of SVM using kernal linear  scores          0.25
the accuracy of SVM using kernal poly  scores    0.5
the accuracy of SVM using kernal sigmod  scores          0.25
the accuracy of SVM using kernal precomputed  scores     0.25

D:\大学\大三上\机器学习\homework1>python part1.py SVM
the accuracy of SVM using kernal rbf  scores     0.25
the accuracy of SVM using kernal linear  scores          0.25
the accuracy of SVM using kernal poly  scores    0.25
the accuracy of SVM using kernal sigmod  scores          0.5
the accuracy of SVM using kernal precomputed  scores     0.5

D:\大学\大三上\机器学习\homework1>
```

补充题

代码使用方法

1. python part2.py -h 调用使用帮助

```
D:\大学\大三上\机器学习\homework1>python part2.py -h
usage: part2.py [-h] [--fold FOLD] action

positional arguments:
  action       watermelon or Iris, the program will run algorithm on this two
               dataset.

optional arguments:
  -h, --help   show this help message and exit
  --fold FOLD  the num of folds in K-flod validation, default is 5.
```

2. python part2.py Iris(--fold 5) 括号内容可选且值为默认
3. python part2.py watermelon(--fold 5) 括号内容可选且值为默认

   实例

```
D:\大学\大三上\机器学习\homework1>python part2.py Iris --fold 10
D:\大学\大三上\机器学习\homework1\logistic.py:37: RuntimeWarning: overflow encountered in exp
  return 1/(1+np.exp(X.dot(self.weight)))
using Naive Beyas on Iris with  10 flods accuracy : [1.0, 1.0, 1.0, 0.9333333333333333, 0.9333333333333333, 0.8, 0.9333333333333333, 0.8666666666666667, 1.0, 0.8666666666666667]
using LDA on Iris with  10 flods accuracy : [1.0, 0.9333333333333333, 1.0, 1.0, 0.8666666666666667, 1.0, 1.0, 1.0, 1.0, 1.0]
using SVM on Iris with  10 flods accuracy : [0.9333333333333333, 0.9333333333333333, 1.0, 1.0, 0.8666666666666667, 0.9333333333333333, 1.0, 1.0, 1.0, 0.9333333333333333]
using LogisticRegression on Iris with  10 flods accuracy : [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]

D:\大学\大三上\机器学习\homework1>
```

实验结果

多种方法iris

part1.py　part2.py ×　getData.py　logistic.py

```python
                probablity0 *= 1/(np.sqrt(2*np.pi)*sigma0)*np.exp(-np.square(float(item[num])-mu0)/(2*np.square(sigma0)))
                probablity1 *= 1/(np.sqrt(2*np.pi)*sigma1)*np.exp(-np.square(float(item[num])-mu1)/(2*np.square(sigma1)))
                probablity2 *= 1/(np.sqrt(2*np.pi)*sigma2)*np.exp(-np.square(float(item[num])-mu2)/(2*np.square(sigma2)))
            porbalis = [probablity0,probablity1,probablity2]
            Y_hat.append((porbalis.index(max(porbalis))))

        Y_hat = np.array(Y_hat)
        Y = np.array(Y)

        return np.sum(Y_hat==Y)/len(X)

if __name__ == "__main__":
    parse = argparse.ArgumentParser()
    parse.add_argument("action", type=str, help="watermelon or Iris, the program will run algorithm on this two dataset.")
    # parse.add_argument("--rate", type=float, default=0.8, help="the preportion of training set, default is 0.8.")
    parse.add_argument("--fold", type=int, default=5, help="the num of folds in K-flod validation, default is 5.")

    args = parse.parse_args()

    if args.action=="watermelon":
        watermalon = pd.read_csv("watermalon.csv",engine="python")
```

问题 15　输出　调试控制台　终端　2: Py

```
(base) D:\大学\大三上\机器学习\homework1>python part2.py Iris
D:\大学\大三上\机器学习\homework1\logistic.py:37: RuntimeWarning: overflow encountered in exp
  return 1/(1+np.exp(X.dot(self.weight)))
using Naive Beyas on Iris with  5 flods accuracy : [1.0, 0.9666666666666667, 0.8666666666666667, 0.9333333333333333, 0.9333333333333333]
using LDA on Iris with  5 flods accuracy : [1.0, 0.9666666666666667, 0.9333333333333333, 0.9333333333333333, 1.0]
using SVM on Iris with  5 flods accuracy : [0.9333333333333333, 1.0, 0.9, 1.0, 0.9666666666666667]
using LogisticRegression on Iris with  5 flods accuracy : [1.0, 1.0, 1.0, 1.0, 1.0]

(base) D:\大学\大三上\机器学习\homework1>python part2.py Iris
D:\大学\大三上\机器学习\homework1\logistic.py:37: RuntimeWarning: overflow encountered in exp
  return 1/(1+np.exp(X.dot(self.weight)))
using Naive Beyas on Iris with  5 flods accuracy : [1.0, 0.9666666666666667, 0.8666666666666667, 0.9333333333333333, 0.9333333333333333]
using LDA on Iris with  5 flods accuracy : [1.0, 0.9666666666666667, 0.9333333333333333, 0.9333333333333333, 1.0]
using SVM on Iris with  5 flods accuracy : [0.9333333333333333, 1.0, 0.9, 1.0, 0.9666666666666667]
using LogisticRegression on Iris with  5 flods accuracy : [1.0, 1.0, 1.0, 1.0, 1.0]

(base) D:\大学\大三上\机器学习\homework1>python part2.py Iris
D:\大学\大三上\机器学习\homework1\logistic.py:37: RuntimeWarning: overflow encountered in exp
  return 1/(1+np.exp(X.dot(self.weight)))
using Naive Beyas on Iris with  5 flods accuracy : [1.0, 0.9666666666666667, 0.8666666666666667, 0.9333333333333333, 0.9333333333333333]
using LDA on Iris with  5 flods accuracy : [1.0, 0.9666666666666667, 0.9333333333333333, 0.9333333333333333, 1.0]
using SVM on Iris with  5 flods accuracy : [0.9333333333333333, 1.0, 0.9, 1.0, 0.9666666666666667]
using LogisticRegression on Iris with  5 flods accuracy : [1.0, 1.0, 1.0, 1.0, 1.0]

(base) D:\大学\大三上\机器学习\homework1>
```

多种方法西瓜

问题 15　输出　调试控制台　终端

```
(base) D:\大学\大三上\机器学习\homework1>python part2.py watermelon
using Naive Beyas on watermelon with  5 flods accuracy : [0.75, 0.75, 0.75, 0.75, 0.75]
using LDA on watermelon with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]
using SVM on watermelon with  5 flods accuracy : [0.25, 0.25, 0.25, 0.25, 0.25]
using LogisticRegression on Iris with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]

(base) D:\大学\大三上\机器学习\homework1>python part2.py watermelon
using Naive Beyas on watermelon with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]
using LDA on watermelon with  5 flods accuracy : [0.25, 0.25, 0.25, 0.25, 0.25]
using SVM on watermelon with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]
using LogisticRegression on Iris with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]

(base) D:\大学\大三上\机器学习\homework1>python part2.py watermelon
using Naive Beyas on watermelon with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]
using LDA on watermelon with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]
using SVM on watermelon with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]
using LogisticRegression on Iris with  5 flods accuracy : [0.5, 0.5, 0.5, 0.5, 0.5]

(base) D:\大学\大三上\机器学习\homework1>
```

附注

实验环境

sklearn.**version** = '0.19.1'