
# 实 验 报 告

## (2017 / 2018 学年 第 2 学期)

| | |
|---|---|
| 课程名称 | 机器学习导论 |
| 实验名称 | Logistic Regression Classifier |
| 实验时间 | 2018 年 5 月 29 日 |
| 指导教师 | 王邦 |

| | | | |
|---|---|---|---|
| 姓名 | 游浩然 | 学号 | U201515429 |

# 1 问题重述

- 钞票数据集（Banknote Dataset）涉及根据给定钞票的 4 个度量的特征。据此预测是真钞还是假钞。

# 2 Python Code for Logistic regression

## 2.1 logistic

```python
# -*- coding: utf-8 -*-
"""
@author : Haoran You

"""
import numpy as np
import matplotlib.pyplot as plt
# from bigfloat import exp
import random
import csv
import os

class logistic():
    def name(self):
        return 'Logistic Model'

    def sigmoid(self, x):
        return 1.0 / (1 + np.exp(x))

    def stoGradAscent(self, data, label, numIter):
        m, n = np.shape(data)
        weight = [float(1.0) for i in range(n)]
        for j in range(numIter):
            dataIndex = list(range(m))
            for i in range(m):
                # alpha = 4 / (1.0+j+i) + 0.0001
                alpha = 0.00001
                randIndex = int(random.uniform(0, len(dataIndex)))
                h = self.sigmoid(np.sum(np.dot(data[randIndex], weight)))
                error = label[randIndex] - h
                weight += alpha * error * np.array(data[randIndex])
                del(dataIndex[randIndex])
        return weight

    def classifyVector(self, x, weight):
        pred = self.sigmoid(np.sum(np.dot(x, weight)))
        if pred > 0.5: return 1.0
        else: return 0.0

    def train(self, data, label, numIter):
        self.train_data = data
        self.train_label = label
        self.train_weight = self.stoGradAscent(data, label, numIter)

    def val(self, data, label):
        self.val_data = data
        self.val_label = label
        error = 0
        numTest = np.shape(data)[0]
        for i in range(numTest):
            pred = self.classifyVector(data[i], self.train_weight)
            # print(pred, label[i])
            if int(pred) != int(label[i]):
                error += 1
        errorRate = (float(error) / numTest)
        print('Error rate of val data : %f' % errorRate)

    def test(self, data):
        if os.path.exists('results.csv'):
            os.remove('results.csv')
        f = open('results.csv', 'a', newline='')
        csv_write = csv.writer(f, dialect='excel')
        i = 0
        for vector in data:
            result = []
            i += 1
            pred = self.classifyVector(vector, self.train_weight)
            result.append(i)
            for item in vector:
                result.append(item)
            result.append(pred)
            csv_write.writerow(result)
```

## 2.2 data

```python
# -*- coding: utf-8 -*-
"""
@author : Haoran You

"""
import random

def parse_file(filename, has_cls=True):
    f = open(filename, 'r', encoding='gbk')
    data, label = [], []
    for line in f.readlines():
        if has_cls == True:
            data_str = line.strip().split(',')[:-1]
            data_list = []
            for data_item in data_str:
                data_list.append(float(data_item))
            data.append(data_list)
            label.append(float(line.strip().split(',')[-1]))
        else:
            data_str = line.strip().split(',')
            data_list = []
            for data_item in data_str:
                data_list.append(float(data_item))
            data.append(data_list)
    return data, label

def divide(data, label):
    num_train = int(0.8*len(data))
    train_data, train_label = [], []
    val_data, val_label = [], []
    index = random.sample(range(len(data)), num_train)
    for i in range(0, len(data)):
        if i in index:
            train_data.append(data[i])
            train_label.append(label[i])
        else:
            val_data.append(data[i])
            val_label.append(label[i])
    return train_data, train_label, val_data, val_label

def dataset():
    data, label = parse_file('train.txt')
    train_data, train_label, val_data, val_label = divide(data, label)
    test_data, test_label = parse_file('test.txt', has_cls=False)
    print('number of train : ', len(train_data))
    print('number of val   : ', len(val_data))
    print('number of test  : ', len(test_data))
    return train_data, train_label, val_data, val_label, test_data
```

## 2.3 Main

```python
# -*- coding: utf-8 -*-
"""
@author : Haoran You

"""
from data import dataset
from logistic import logistic

# load data
train_data, train_label, val_data, val_label, test_data = dataset()
# train
logistic = logistic()
logistic.train(train_data, train_label, 30)
# val
logistic.val(val_data, val_label)
# test
logistic.test(test_data)
```