

**模式识别大作业**

题 目 泰坦尼克号

学 院 信息科学与工程

专 业 计算机技术

组 员 孙双运

指导教师 赵海涛

**完成日期： 2018 年 10 月26日**

**模式识别作业报告——**泰坦尼克号

组员：孙双运

经过第二次课程的模式识别学习，在赵海涛老师的辛勤指导下，我对模式识别的SVD算法有了一定的了解，并通过本次针对泰坦尼克号数据集的实验来巩固所学内容。

经过一段时间的程序编写及调试，并尝试了各种主流的推荐系统算法，最终完成了我的实验。

**一、**泰坦尼克号实验简介

**泰坦尼克号**(RMS Titanic)是英国白星航运公司下辖的一艘奥林匹克邮轮，在其处女航行中，因与一座冰山相撞而至沉船。在这次事故中，有约2/3的人丧生。现在给定泰坦尼克号船上的乘客信息，你需要学习一个模型来判断一名乘客在沉船灾难中能否最终存活下来。

* 本题是一个典型的二分类问题
* 先从train.csv中提取每个乘客的多项特征（feature）和存活信息(label)，使用feature和label进行模型训练
* 乘客的特征选取和处理比较重要。类别的特征，如Embarked、Pclass等，可以转换成one-hot encoding的表示形式。数值型的特征，如Age、Parch等可以进行范围标准化。一些很难处理和使用的特征，如Name等，可以考虑直接忽略
* 模型可以使用传统的分类模型，如SVM，Decision Tree等。在训练过程中，要适当引入正则化项，防止模型过拟合，而致其泛化能力变差
* 最后使用训练好的模型预测test.csv中的乘客最终能否存活

SVM(Support Vector Machine)指的是支持向量机，是常见的一种判别方法。在机器学习领域，是一个有监督的学习模型，通常用来进行模式识别、分类以及回归分析。

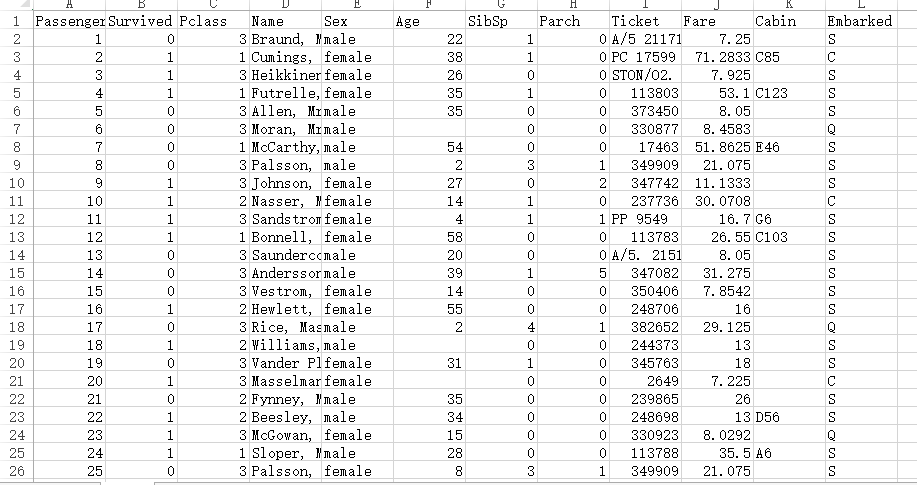
SVM的主要思想可以概括为两点：

1. 它是针对线性可分情况进行分析，对于线性不可分的情况，通过使用非线性映射算法将低维输入空间线性不可分的样本转化高维特征空间使其线性可分，从而使得高维特征空间采用线性算法对样本的非线性特征进行线性分析成为可能。
2. 它基于结构风险最小化理论之上在特征空间中构建最优超平面，使得学习器得到全局最优化，并且在整个样本空间的期望以某个概率满足一定上界。

**二、整体解决方案**

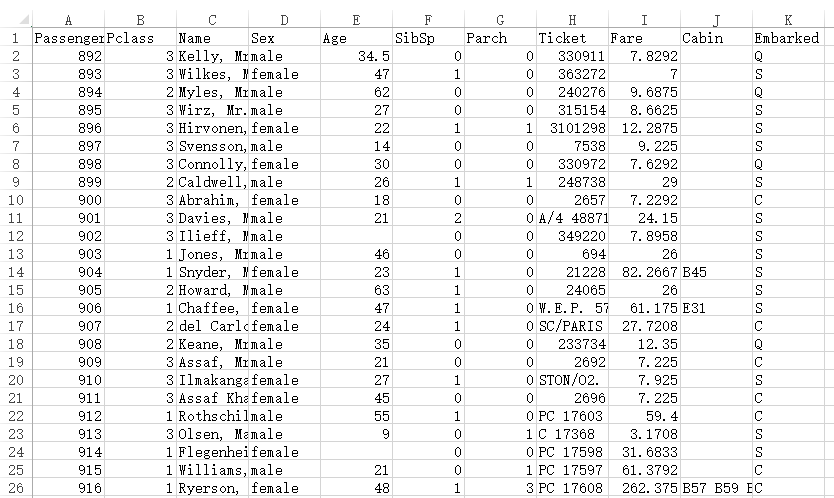
**2.1 数据分析**

首先，数据的结构主要如下图所示：



**图1 训练数据的读入**

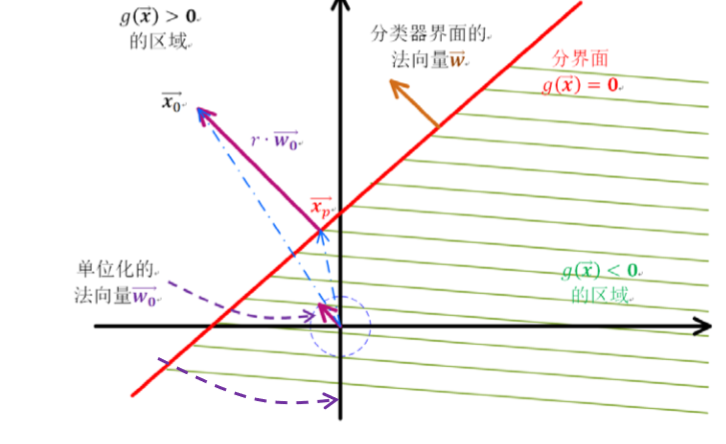
由图可知，数据内容包括PassengerId Survived Pclass Name Sex Age SibSp Parch Ticket Fare Cabin Embarked共计12项内容，这是要训练的数据。

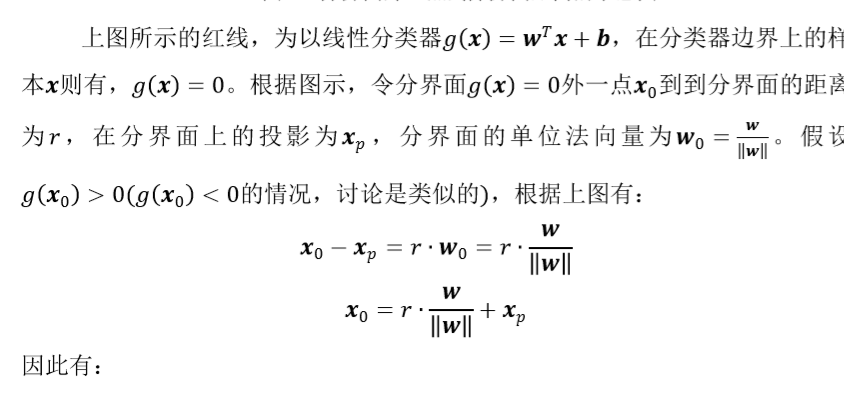


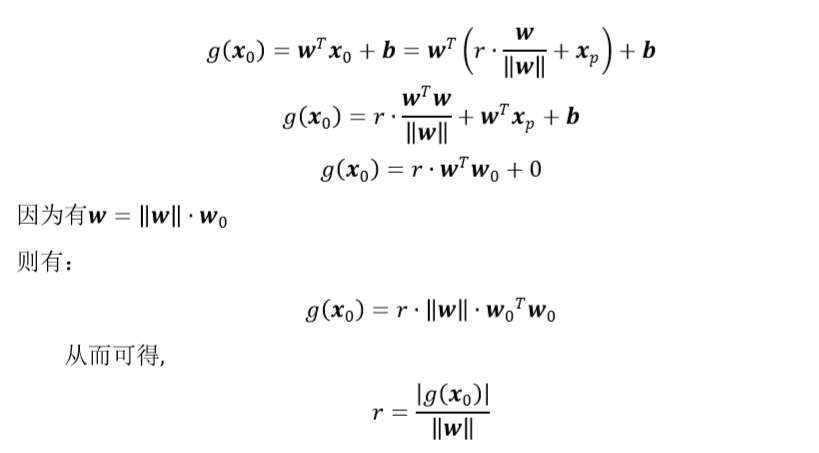
**图2 测试数据的读入**

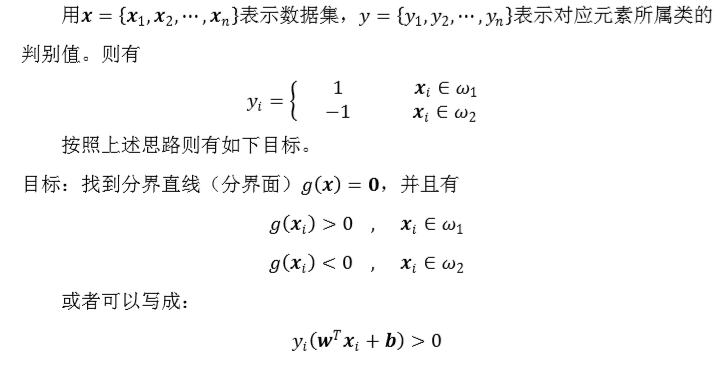
测试集的数据只含有十一项，相对于训练集我们需要给出测试集中每一成员的存活的可能性。本次采用SVM算法进行预测，下面介绍SVM的工作原理。

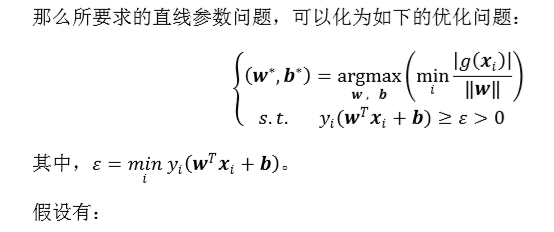
## 2.2 SVM的工作原理

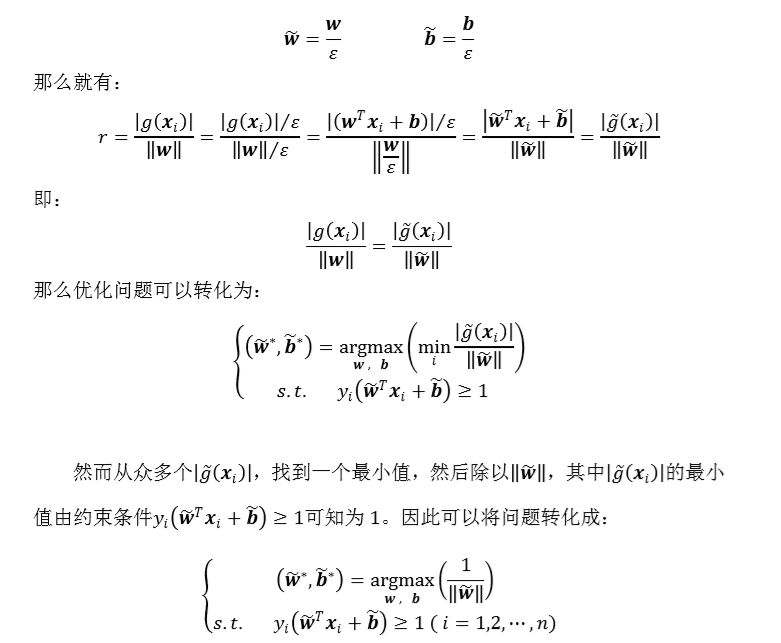


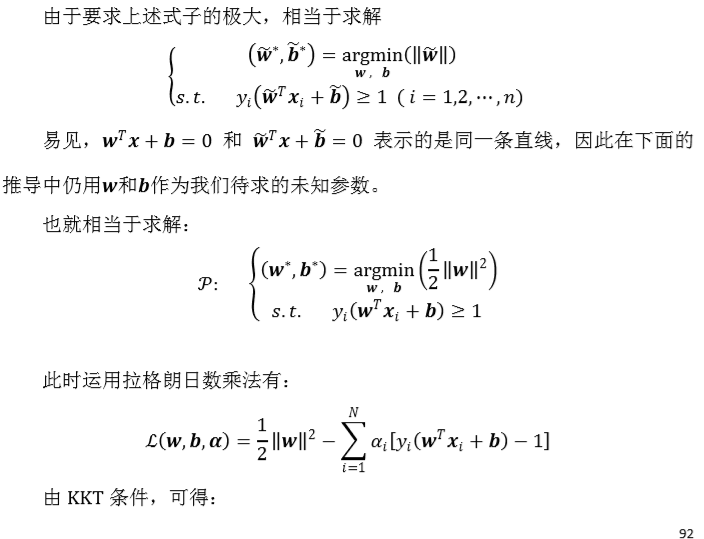


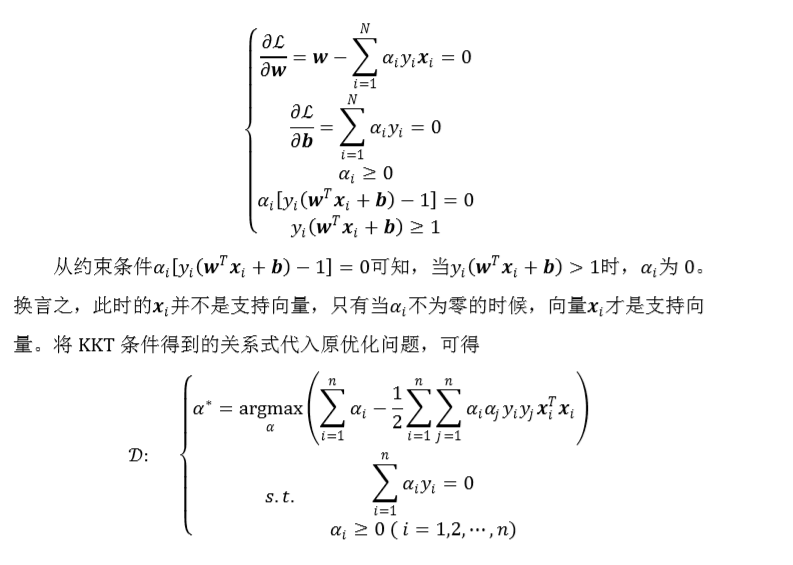


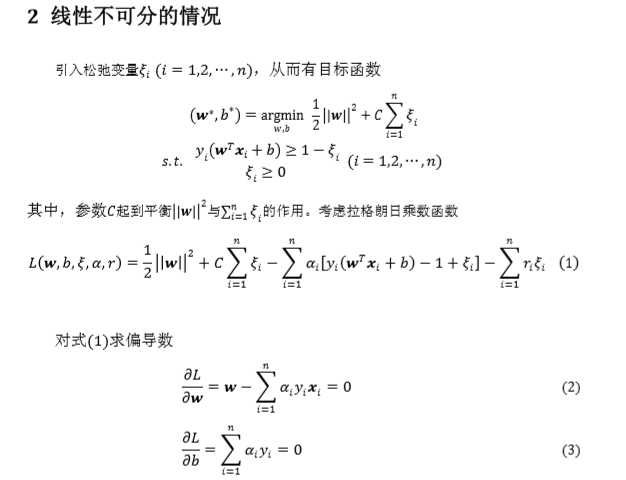


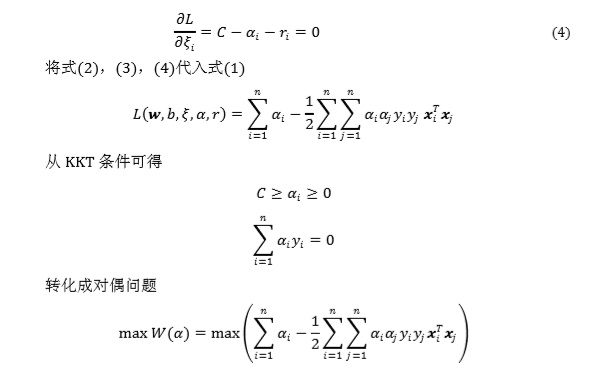












# 2.3编码实现

数据包导入

from \_\_future\_\_ import division

import scipy

from sklearn import svm

import numpy as np

import csv

import sys

from sklearn.svm import SVC

数据声明：

Xtrain=[] # Will contain the characteristic data for each passenger. Xtrain will

# contain info about the passenger's class, sex, age, siblings, parch,

# fare, and cabin, and embark status

Ytrain=[] # WIll contain a binary label for whether each passenger survived

Xtest=[] # Will contain the characteristic data for the passengers in the test set

numTrainExamples = 891

numTestExamples = 418

训练函数编写：

def process(pclass):

pclass = [float(x) for x in pclass]

mean = sum(pclass) / float(len(pclass))

my\_range = max(pclass) - min(pclass)

pclass = [(x-mean)/my\_range for x in pclass]

return pclass

def processGender(gender):

genderToNum=[]

for x in gender:

if (x == "male"):

genderToNum.append(-.5)

else: # x is female

genderToNum.append(.5)

return genderToNum

def processAge(age):

ageWithoutBlanks=[]

for x in age:

if (x != ""):

ageWithoutBlanks.append(x)

ageWithoutBlanks = [float(x) for x in ageWithoutBlanks]

mean = sum(ageWithoutBlanks) / float(len(ageWithoutBlanks))

my\_range = max(ageWithoutBlanks) - min(ageWithoutBlanks)

for x in range(0,len(age)):

if (age[x] == ""): # If there is a blank value, then just set it to mean

age[x] = mean

age = [float(x) for x in age]

age = [(x-mean)/my\_range for x in age]

return age

def processCabin(cabin):

cabinToNum=[]

for x in gender:

if (x == ""):

cabinToNum.append(-.5)

else:

cabinToNum.append(.5)

return cabinToNum

def processFare(fare):

for x in range(0,len(fare)):

if (fare[x] == ""):

fare[x] = 0

fare = [float(x) for x in fare]

mean = sum(fare) / float(len(fare))

my\_range = max(fare) - min(fare)

fare = [(x-mean)/my\_range for x in fare]

return fare

def processEmbarked(embarked):

embarkedToNum=[]

for x in embarked:

if (x == "S"):

embarkedToNum.append(-.5)

elif (x == "C"):

embarkedToNum.append(0)

elif (x == "Q"):

embarkedToNum.append(0.5)

else:

embarkedToNum.append(0)

return embarkedToNum

训练集数据导入：

skip = True

train\_file = open('train.csv')

csv\_file = csv.reader(train\_file)

生成训练模型：

# Creating temporary lists where we store data for each feature/characteristic

gender,Pclass,age,sibSP,parch,fare,cabin,embarked = ([] for i in range(8))

for row in csv\_file:

if (skip == True):

skip = False

continue

# Filling lists with values from train.csv

Ytrain.append(row[1])

Pclass.append(row[2])

gender.append(row[4])

age.append(row[5])

sibSP.append(row[6])

parch.append(row[7])

fare.append(row[9])

cabin.append(row[10])

embarked.append(row[11])

# Processing each feature list

Pclass = process(Pclass)

gender = processGender(gender)

age = processAge(age)

sibSP = process(sibSP)

parch = process(parch)

fare = processFare(fare)

cabin = processCabin(cabin)

embarked = processEmbarked(embarked)

# Adding values from previous feature lists to one large Xtrain list of lists

for x in range(0,numTrainExamples):

Xtrain.append([Pclass[x],gender[x],age[x],sibSP[x],

parch[x],fare[x],cabin[x],embarked[x]])

测试集数据导入：

skip = True

test\_file = open('test.csv')

csv\_file2 = csv.reader(test\_file)

测试集数据导入：

skip = True

test\_file = open('test.csv')

csv\_file2 = csv.reader(test\_file)

测试集数据训练:

gender,Pclass,age,sibSP,parch,fare,cabin,embarked = ([] for i in range(8))

for row in csv\_file2:

if (skip == True):

skip = False

continue

Pclass.append(row[1])

gender.append(row[3])

age.append(row[4])

sibSP.append(row[5])

parch.append(row[6])

fare.append(row[8])

cabin.append(row[9])

embarked.append(row[10])

Pclass = process(Pclass)

gender = processGender(gender)

age = processAge(age)

sibSP = process(sibSP)

parch = process(parch)

fare = processFare(fare)

cabin = processCabin(cabin)

embarked = processEmbarked(embarked)

for x in range(0,numTestExamples):

Xtest.append([Pclass[x],gender[x],age[x],sibSP[x],

parch[x],fare[x],cabin[x],embarked[x]])

# Representing the list of lists as numpy arrays so that we can

# use the different scikit functions.

Xtrain = np.asarray(Xtrain)

Xtest = np.asarray(Xtest)

Ytrain = np.asarray(Ytrain)

clf = svm.SVC()

print ("Fitting SVM" )

clf.fit(Xtrain, Ytrain)

results = np.ones((numTestExamples,2))

counter = numTrainExamples + 1

print ("Predicting outputs for testing dataset")

#for x in range(0,numTestExamples):

# results[x,0] = counter

# counter = counter + 1

# results[x,1] = clf.predict([Xtest[x].tolist()])[0]

测试集结果输出：

for x in range (0,numTestExamples):

results[x,1]=clf.predict([Xtest[x].tolist()])[0]

results[x,0]=counter

counter=counter+1

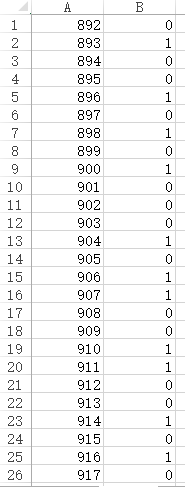
#Saving predictions into a test file that can be uploaded to Kaggle

#NOTE: You have to add a header row before submitting the txt file

np.savetxt('result.csv', results, delimiter=',', fmt = '%i')

print("success")

部分数据结果如下：



**三、小组分工**

程序设计及编写：孙双运

程序调试：孙双运

实验报告：孙双运

**四、作业总结**

这次的大作业让增加了我对模式识别的兴趣，同时原来神秘的算法变得立体清晰。感谢赵老师教学帮助，并且在作业过程中给了我们大家很多指导。

**附：文件说明**

本次附件一共包含有：

1 大作业报告；

2 最终的Python实现程序源码：Titanic SVM1.py

3 导出了预测数据集：result.csv