

**软件可靠性课程实验报告**

**题 目: G-O**软件可靠性模型

**院 系:** 计算机科学与技术学院

**专 业:** 软件工程

**学生姓名:** 徐宇顺

**学 号:** 031720106

**二零一九 年 十 月 六日**

目录

[1.引言 3](#_Toc529726265)

[1．1编写目的 3](#_Toc529726266)

[2.模型理论 3](#_Toc529726267)

[2.1 模型背景 3](#_Toc529726268)

[2.2 模型假设 3](#_Toc529726269)

[2.3 模型推导 4](#_Toc529726270)

[3.算法实现 5](#_Toc529726271)

[3.1 流程图 5](#_Toc529726272)

[3.2 算法伪代码 6](#_Toc529726273)

[3.3 算法实现 7](#_Toc529726274)

[3.5 数据来源 8](#_Toc529726275)

[3.6 结果展示 8](#_Toc529726276)

[4.总结 9](#_Toc529726277)

[参考文献 9](#_Toc529726278)

# 1.引言

## 1．1编写目的

随着软件规模越来越大，结构日趋复杂，应用日趋广泛。软件危机依然是我们难以逾越的障碍， 加强软件工程管理，势在必行，势在必然！改进和提高软件可靠性，为部队提供可靠顶用的装备是我们的义务和责任！

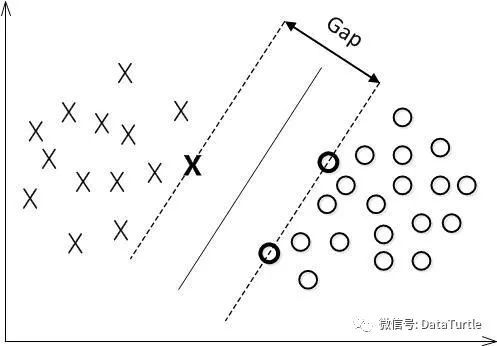
1. 模型理论
   1. 模型背景

SVR全称是support vector regression，是SVM（支持向量机support vector machine）对回归问题的一种运用。

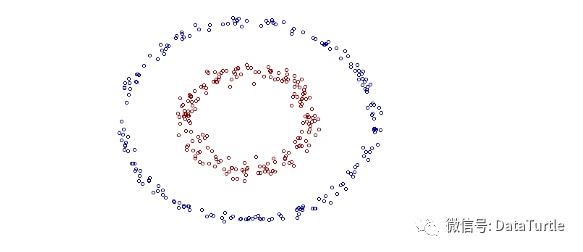
* 1. 模型假设

SVM与logistic分类器类似，也是一种二类分类模型，其基本模型定义为特征空间上的间隔最大的线性分类器，其学习策略便是间隔最大化，最终可转化为一个凸二次规划问题的求解。

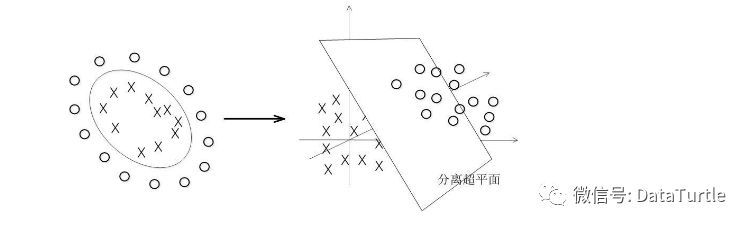
对于下面一个数据集，有两类分别使用×和○来表示。那么我们想要找到一条曲线来区分这两类。可想而知，这样的曲线存在无数条，如何找到那一条最优的曲线呢？那就是要找到间隔（Gap）最大的那一条曲线，然后我们就可以根据这条曲线来分类了。而这条曲线，我们称为超平面。图中加粗的×与○，我们则称这些向量为支持向量。



而实际上，我们很少会遇到上面的情况，能直接使用一条直线将数据分类，更多的时候，数据无法通过直线来分类。比如下面这种情况：



然在二维上该数据无法使用线性分类，那我们就将数据映射到更高维上，在高维上构造超平面，从而完成分类。



这就是为什么将上面的直线称为超平面。

所以对于非线性的模型，我们需要：

（1）使用非线性映射将数据投影至特征空间；

（2）在特征空间使用线性分类器；

看似很简单，但是我们在映射时，就会发现当变量增多时，映射到高维空间的维度是呈指数增长的，计算起来十分困难，这时候就需要核函数（kernel function）了。核函数也是将特征从低维到高维进行转换，但是它是先在低维上进行计算，实际的分类效果表现在高维上。这样，我们就避免了在高维上复杂的计算，仍得到相同的结果。

一些常用的核函数：

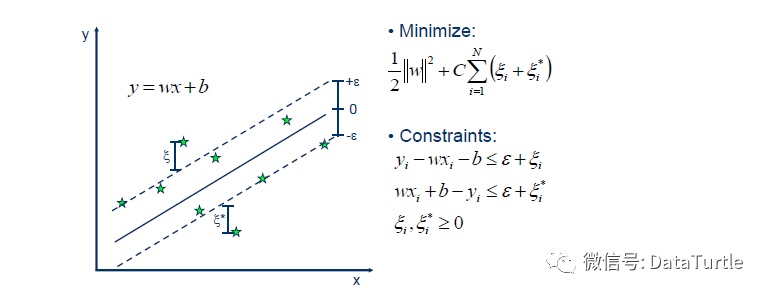
（1）多项式核

（2）高斯核

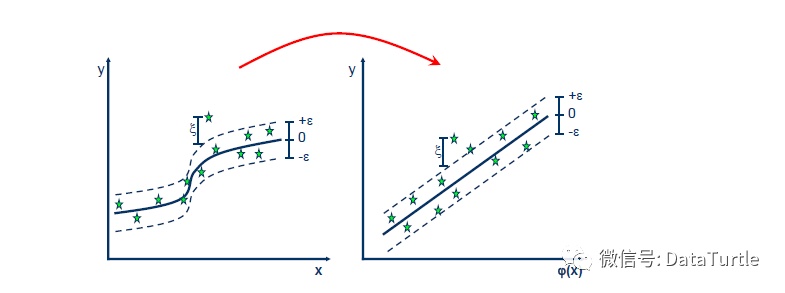
（3）线性核

* 1. 模型推导

最简单的线性回归模型是要找出一条曲线使得残差最小。同样的，SVR也是要找出一个超平面，使得所有数据到这个超平面的距离最小。

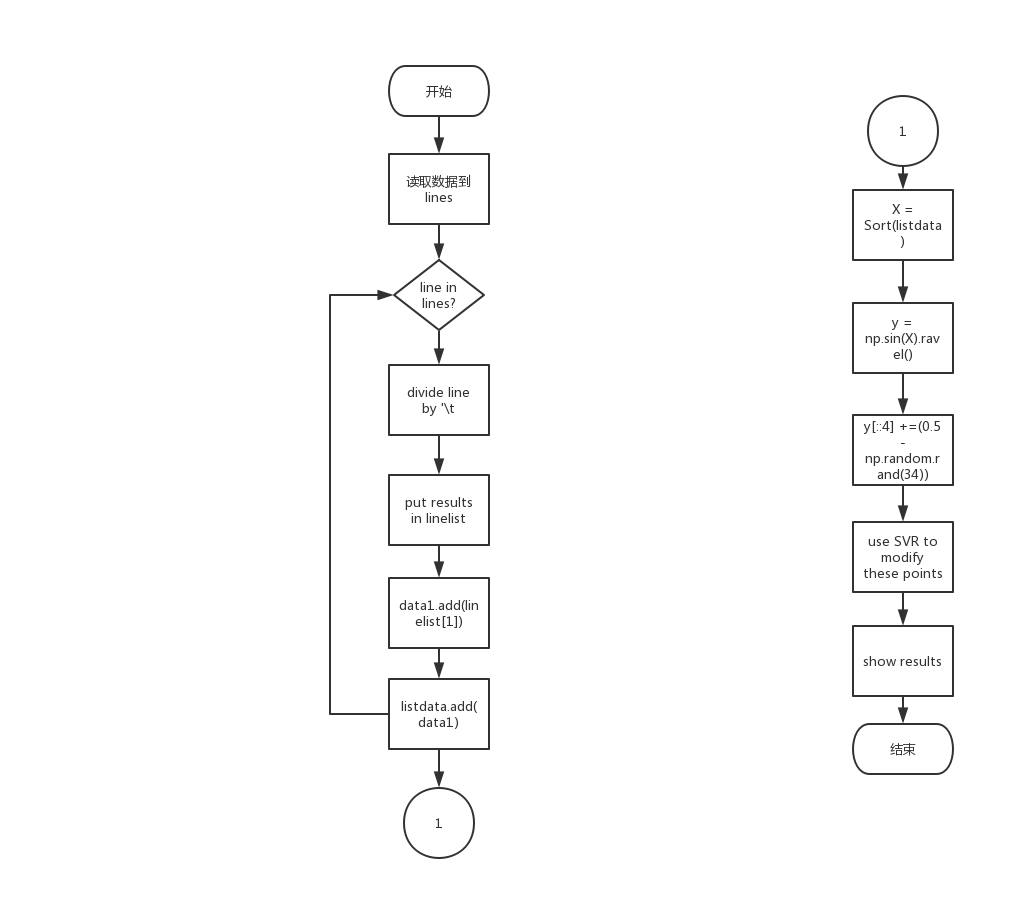


SVR是SVM的一种运用，基本的思路是一致，除了一些细微的区别。使用SVR作回归分析，与SVM一样，我们需要找到一个超平面，不同的是：在SVM中我们要找出一个间隔（gap）最大的超平面，而在SVR，我们定义一个ε，如上图所示，定义虚线内区域的数据点的残差为0，而虚线区域外的数据点（支持向量）到虚线的边界的距离为残差（ζ）。与线性模型类似，我们希望这些残差（ζ）最小。所以大致上来说，SVR就是要找出一个最佳的条状区域（2ε宽度），再对区域外的点进行回归。



# 3.算法实现

## 3.1 流程图

****

## 3.2 算法伪代码

Procedure

Begin

Read data from the file 'SYS1(failue\_count).txt'

put this data in lines

for line in lines

divide line by '\t'

and put these number in linelist

data1.add(linelist[1])

listdata.add(data1) --\*元素为list的list

X = sort(listdata)

y = np.sin(X).ravel() --\*对应为列

--\*add noise to targets

y[::4] +=(0.5 - np.random.rand(34))

use SVR to modify these points

end

## 3.3 算法实现

本次实验采用Python语言编写

#-\*-coding:utf-8-\*-

import numpy as np

from sklearn.svm import SVR

import matplotlib.pyplot as plt

from scipy.interpolate import spline

filename = 'D:\\小黄片\\软件可靠性\\SYS1(failue\_count).txt'

with open(filename) as file\_object:

lines = file\_object.readlines()

listdata=[]

for line in lines:

data1=[]

linestr = line.strip()

linestrlist = linestr.split("\t")

linelist = [int(i) for i in linestrlist]

data1.append(linelist[1])

listdata.append(data1)

del data1

# Generate sample data

X = np.sort(listdata, axis=0)

y = np.sin(X).ravel() #np.sin()输出的是列，和X对应，ravel表示转换成行

# Add noise to targets

y[::4] +=(0.5 - np.random.rand(34))

svr\_rbf10 = SVR(kernel='rbf',C=100, gamma=10.0)

svr\_rbf1 = SVR(kernel='rbf', C=100, gamma=0.1)

svr\_rbf1 = SVR(kernel='rbf', C=100, gamma=0.1)

y\_rbf10 = svr\_rbf10.fit(X, y).predict(X)

y\_rbf1 = svr\_rbf1.fit(X, y).predict(X)

# look at the results

lw = 2 #line width

plt.scatter(X, y, color='darkorange', label='data')

plt.plot(X, y,color='red',linewidth=1,label='Real line')

plt.plot(X, y\_rbf10, color='navy', lw=lw, label='RBF gamma=10.0')

plt.plot(X, y\_rbf1, color='c', lw=lw, label='RBF gamma=1.0')

plt.xlabel('data')

plt.ylabel('target')

plt.title('SVR')

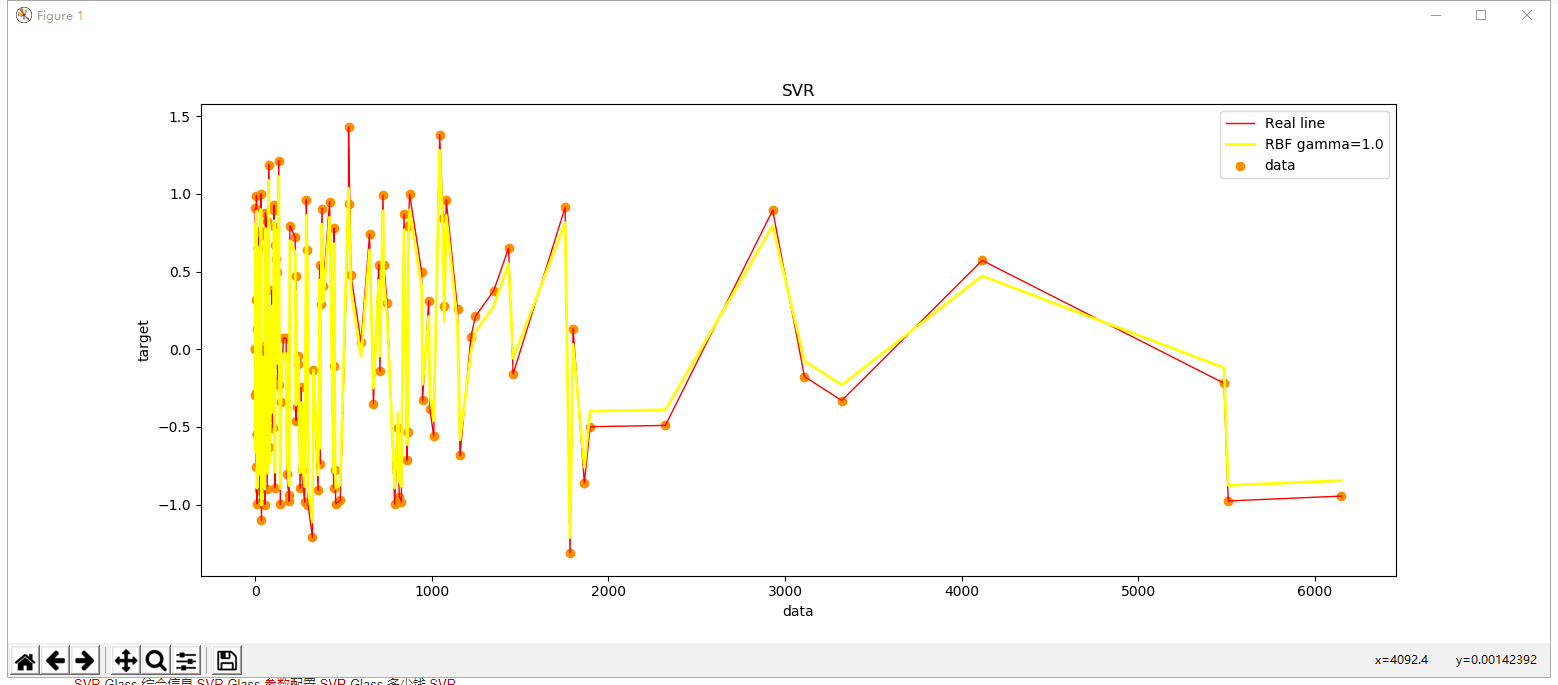
plt.legend()

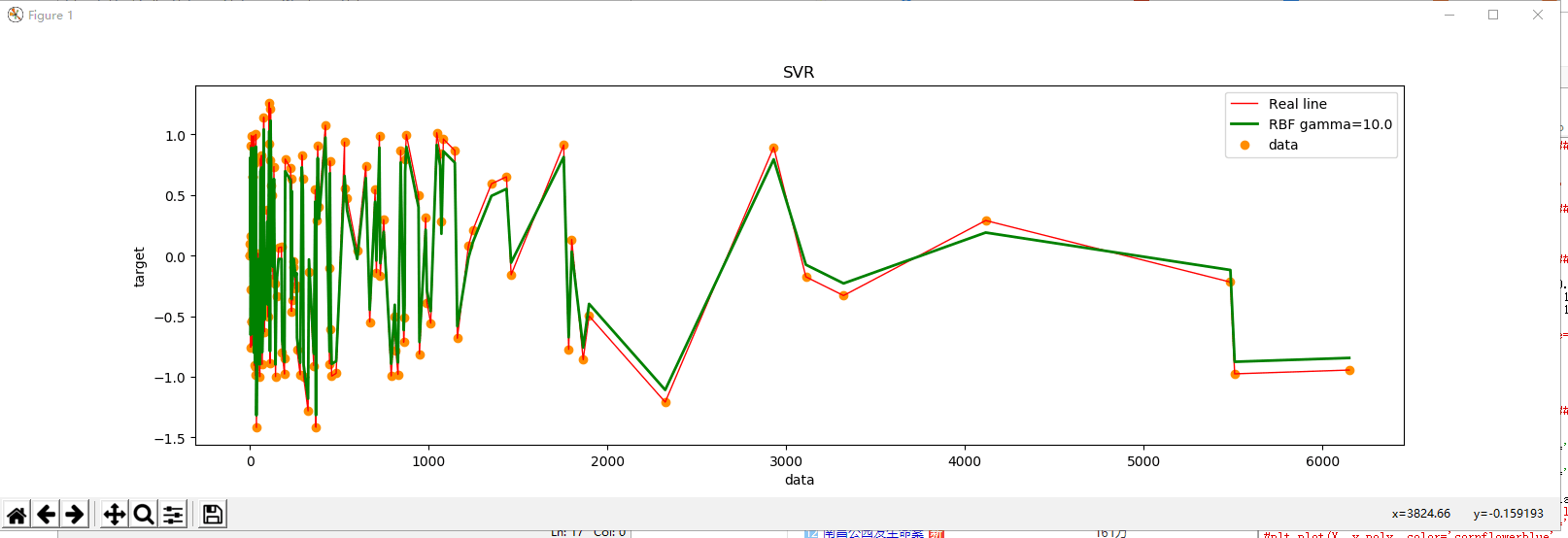
plt.show()

## 3.5 数据来源

数据来源是老师提供的失效错误数据SYS1(failue\_count).txt，我已放入了java工程目录下。

## 3.6 结果展示





以上是结果

Gamma不同是指的是核函数的系数不同