Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский Политехнический Университет Петра Великого
—
Институт компьютерных наук и технологий
Высшая школа искусственного интеллекта

ЛАБОРАТОРНАЯ РАБОТА №3

«Метод опорных векторов»

по дисциплине «Машинное обучение, часть1»

Выполнил: студент группы
3540201/20302 С.А. Ляхова

<*подпись*>

Проверил: Л.В. Уткин
д.т.н., профессор

<*подпись*>

Санкт-Петербург
2022

Содержание

1. Цель работы

Исследовать метод *svm* пакета *e1071* языка R, разделяющий гиперплоскостью данные в более многомерном пространстве, чем исходное, выполнив поставленные задачи и проанализировав результаты.

2. Формулировка задания

Данные для обучения и тестирования SVM-моделей, которые необходимо построить в приведенных ниже заданиях, хранятся в файлах с именами svmdataI.txt и svmdataItest.txt, где I номер задания.

1. Постройте алгоритм метода опорных векторов типа "C-classification" с параметром C = 1, используя ядро "linear". Визуализируйте разбиение пространства признаков на области с помощью полученной модели. Выведите количество полученных опорных векторов, а также ошибки классификации на обучающей и тестовой выборках.

2. Используя алгоритм метода опорных векторов типа "C-classification" с линейным ядром, добейтесь нулевой ошибки сначала на обучающей выборке, а затем на тестовой, путем изменения параметра C. Выберите оптимальное значение данного параметра и объясните свой выбор. Всегда ли нужно добиваться минимизации ошибки на обучающей выборке?

3. Среди ядер "polynomial", "radial" и "sigmoid" выберите оптимальное в плане количества ошибок на тестовой выборке. Попробуйте различные значения параметра degree для полиномиального ядра.

4. Среди ядер "polynomial", "radial" и "sigmoid" выберите оптимальное в плане количества ошибок на тестовой выборке.

5. Среди ядер "polynomial", "radial" и "sigmoid" выберите оптимальное в плане количества ошибок на тестовой выборке. Изменяя значение параметра gamma, продемонстрируйте эффект переобучения, выполните при этом визуализацию разбиения пространства признаков на области.

6. Постройте алгоритм метода опорных векторов типа "eps-regression" с параметром C = 1, используя ядро "radial". Отобразите на графике зависимость среднеквадратичной ошибки на обучающей выборке от значения параметра ε. Прокомментируйте полученный результат.

3. Ход работы

Задание №1

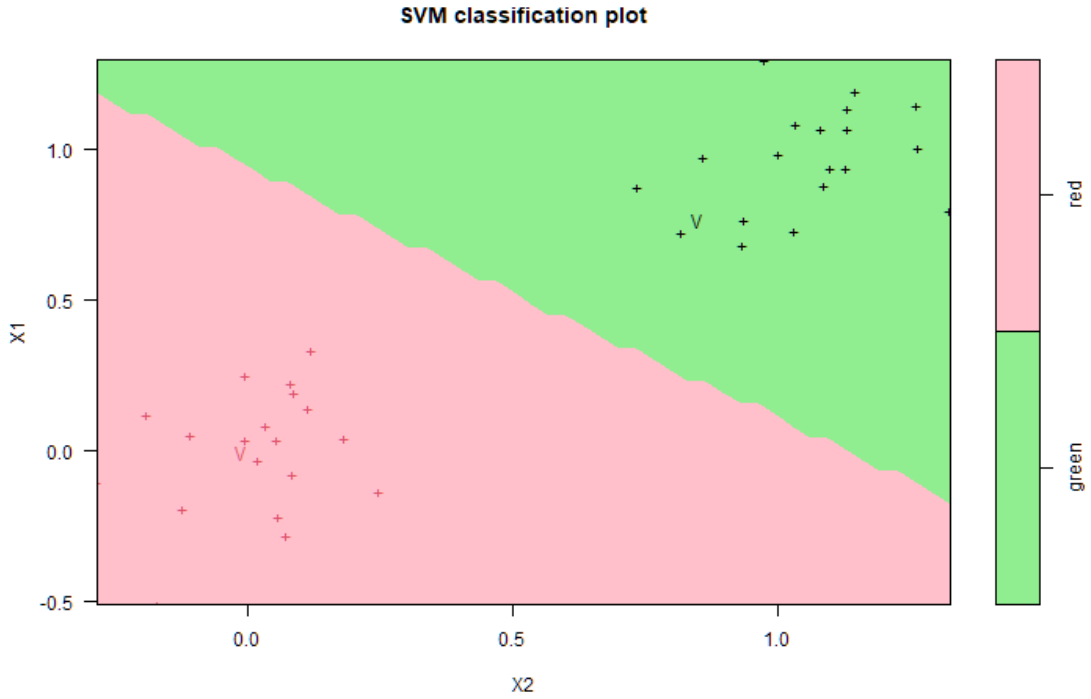

Рисунок 1. Классификация SVM с использованием линейного ядра

Был разработал классификатор SVM с линейным ядром.

model <- svm(Color ~ ., kernel = "linear", data = data_train, type="C-classification", cost=1)

Можем заметить по рисунку 1, что данные сильно кластеризованы, поэтому классификатор сработал с точностью в 100%. Количество полученных опорных векторов – 2.

Задание №2

Аналогичный построенному в задании 1 алгоритм метода опорных векторов был построен для данных svmdata2.txt и svmdata2test.txt.

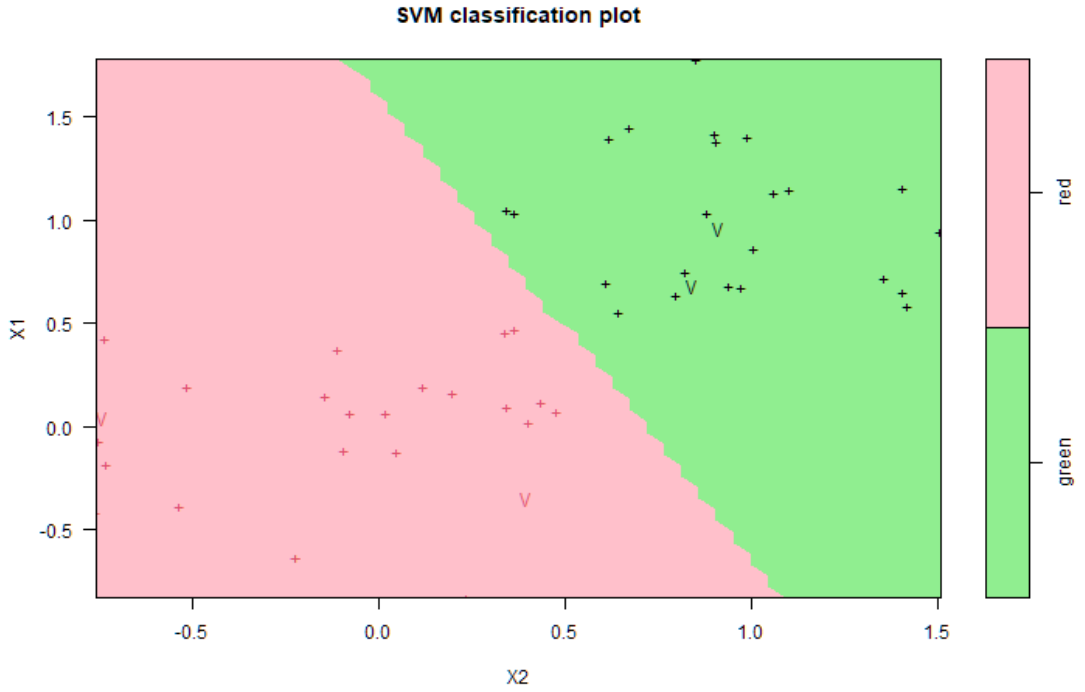

Рисунок 2. Кластеризация SVM с линейным ядром и C = 1

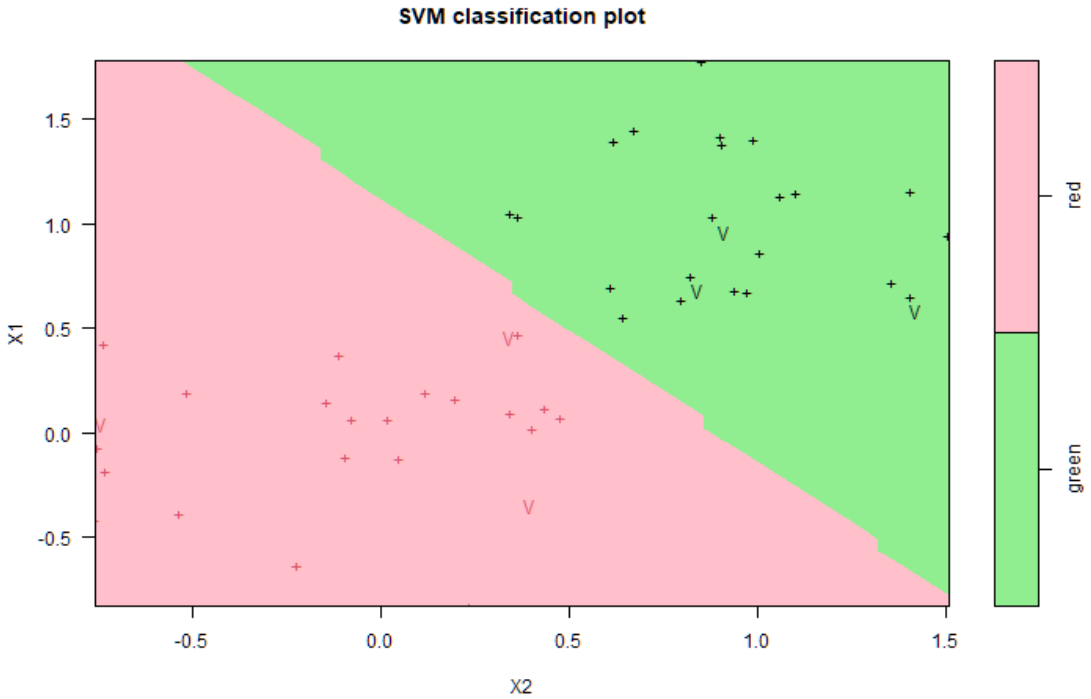

Рисунок 3. Кластеризация SVM с линейным ядром и C = 10

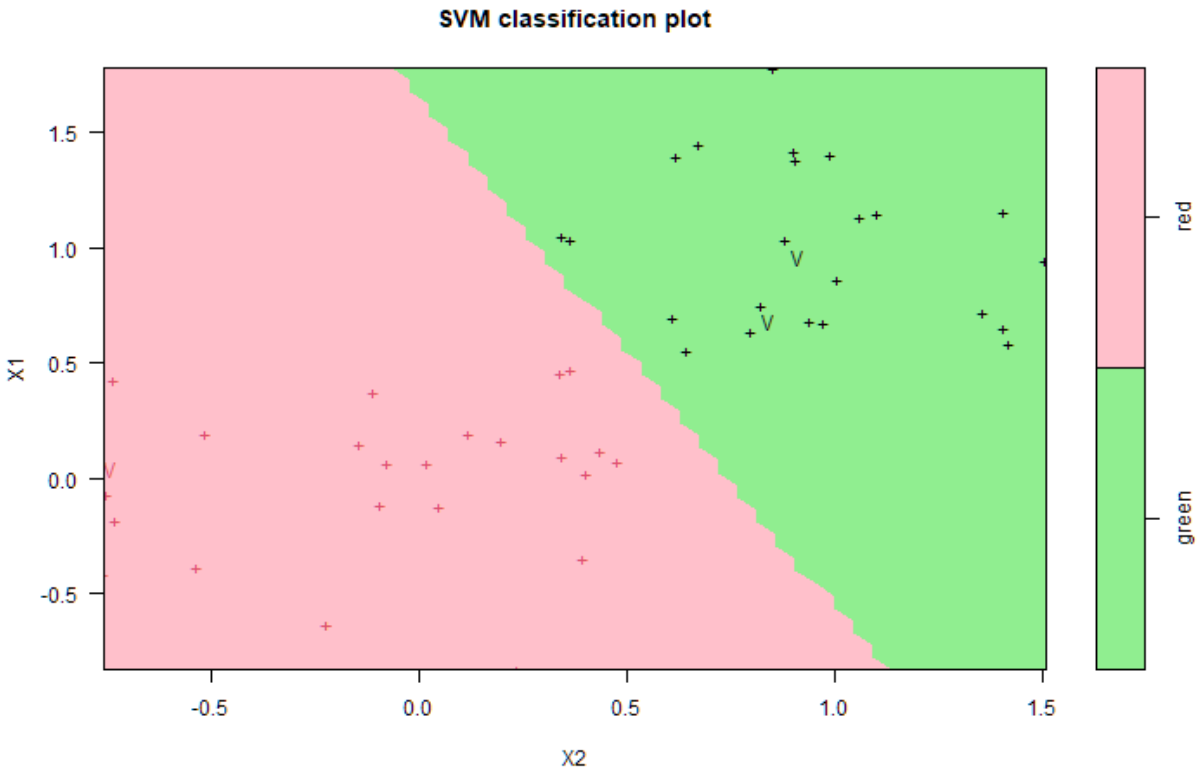

Рисунок 4. Кластеризация SVM с линейным ядром и C = 50

Рисунок 5. Кластеризация SVM с линейным ядром и C = 100

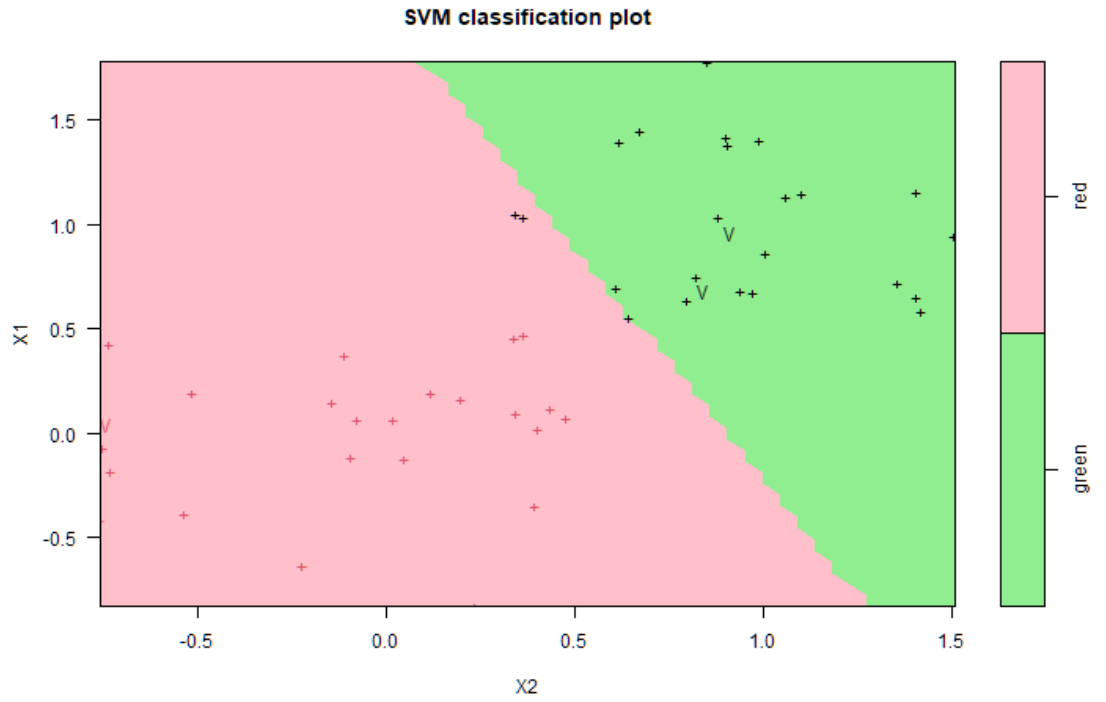

Рисунок 6. Кластеризация SVM с линейным ядром и C = 500

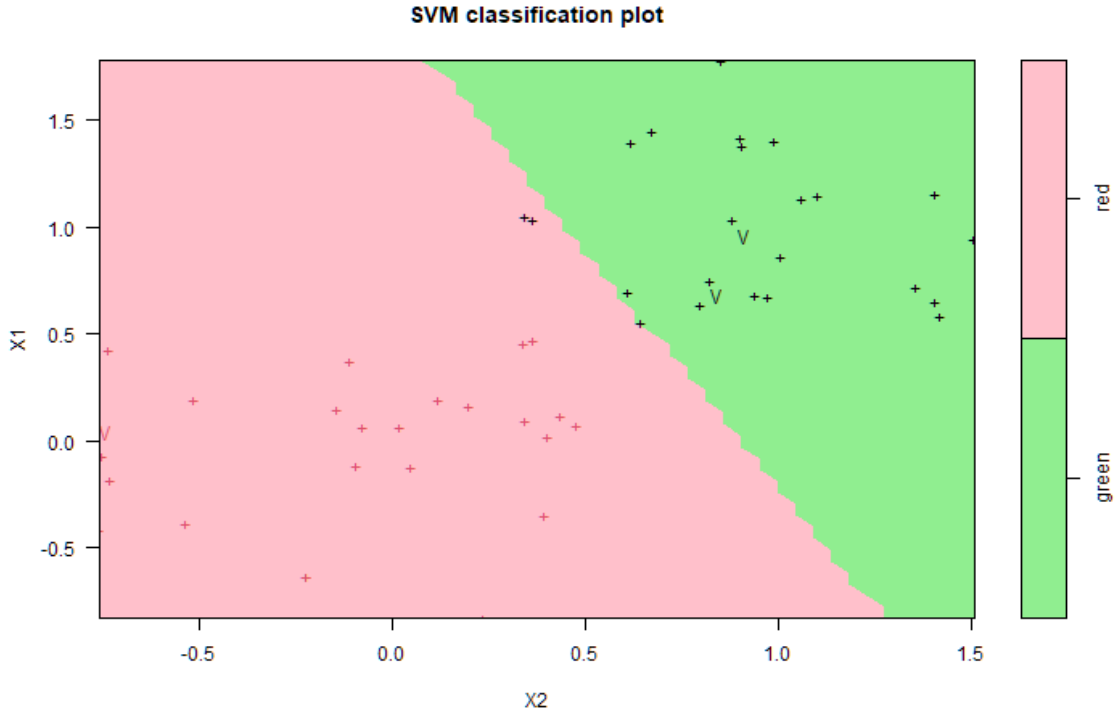

Рисунок 7. Кластеризация SVM с линейным ядром и C = 1000

Можем заметить, что исходные данные чуть менее, но все равно довольно сильно кластеризованы.

Для параметра штрафа, равном 1, 10 и 50 количество опорных векторов получилось 6, 4 и 3. Точность кластеризации на тестовой выборке составила 100%. Эти параметры можно назвать оптимальными для данного датасета.

Для параметра штрафа, равном 100, получилось 3 опорных вектора и точность кластеризации 96%.

Для параметров штрафа, равных 500 и 1000 общая картина кластеризации совсем не меняется (результаты идентичны). Точность кластеризации составила 94%, опорных векторов 3.

При этом при достижении безошибочный классификации тестовых данных (например, параметр C = 10), достоверность классификации тренировочных данных составляет 98%. При изображении тестовых и тренировочных данных на одном рисунке 4.6 видно, что невозможно провести разделяющую плоскость без ошибки классификации. Тогда подбор параметра C заключается в получении наилучших результатов, как по тренировочной, так и по тестовой выборке.

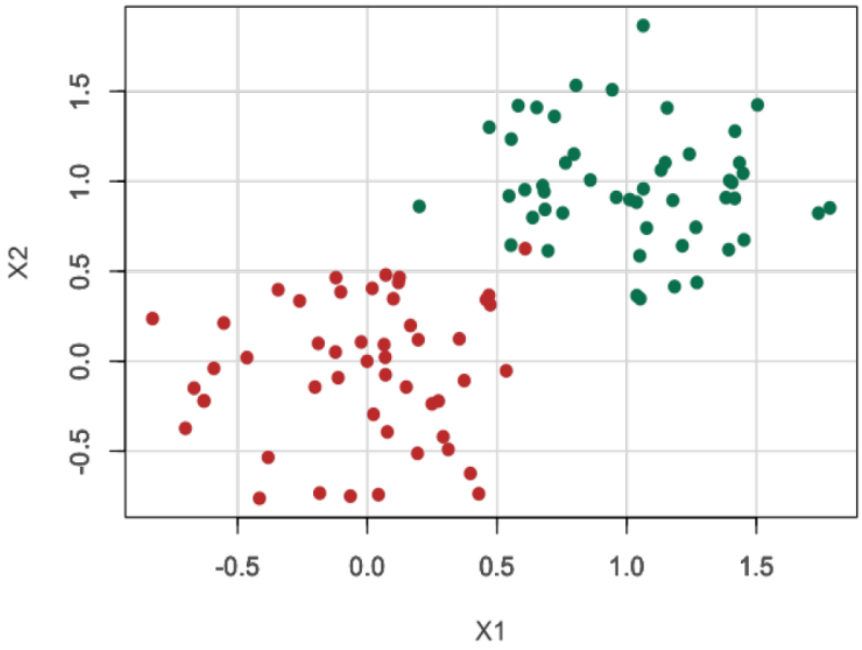

Рисунок 8. Тестовые и тренировочные данные

Задание №3

100 экземпляров набора данных поделены на обучающую (80) и тестовую (20) выборки.

model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "radial")

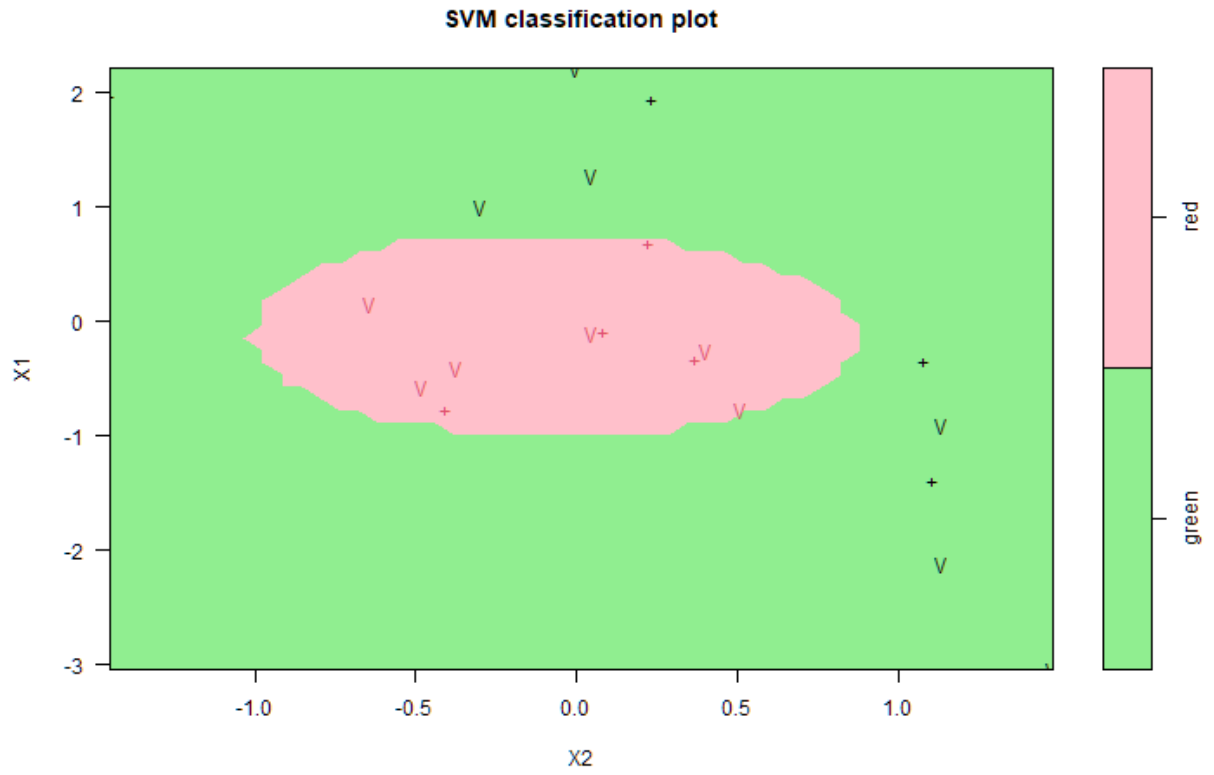

Рисунок 9. Классификация SVM с радиальным ядром

model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "sigmoid")

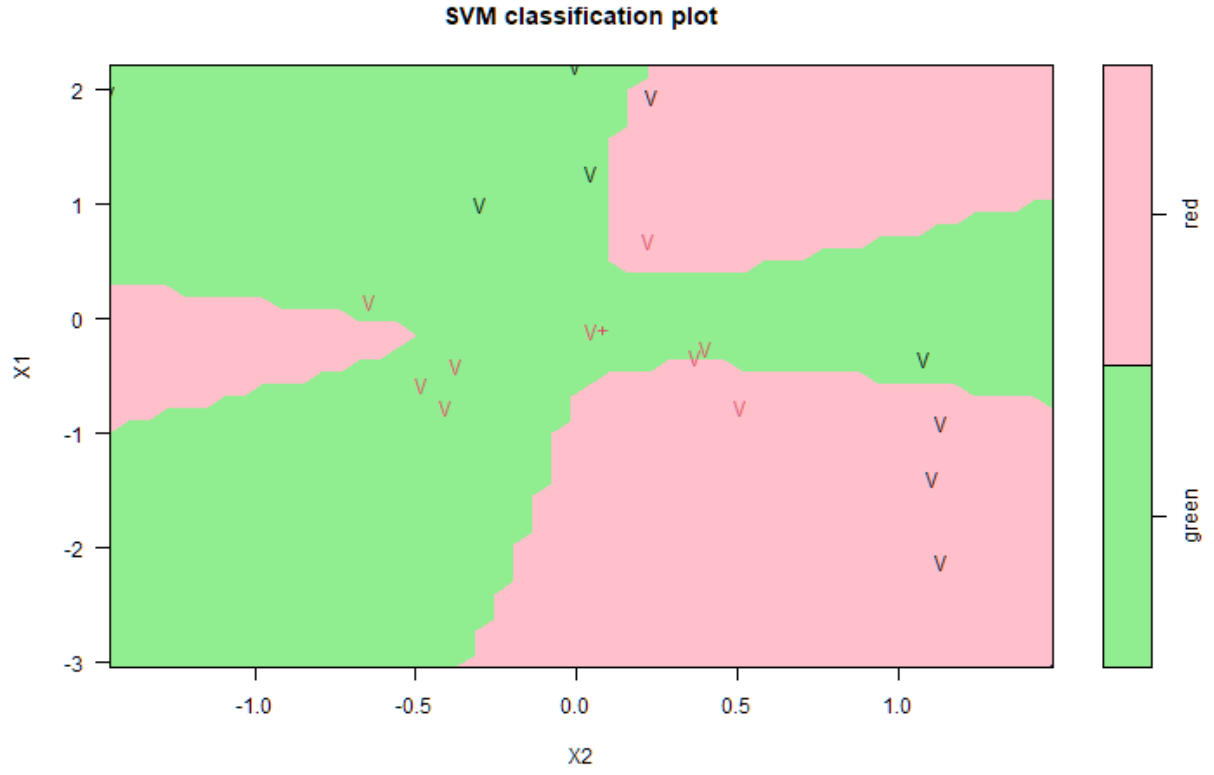

Рисунок 10. Классификация SVM с сигмоидным ядром

model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "poly", degree = d)

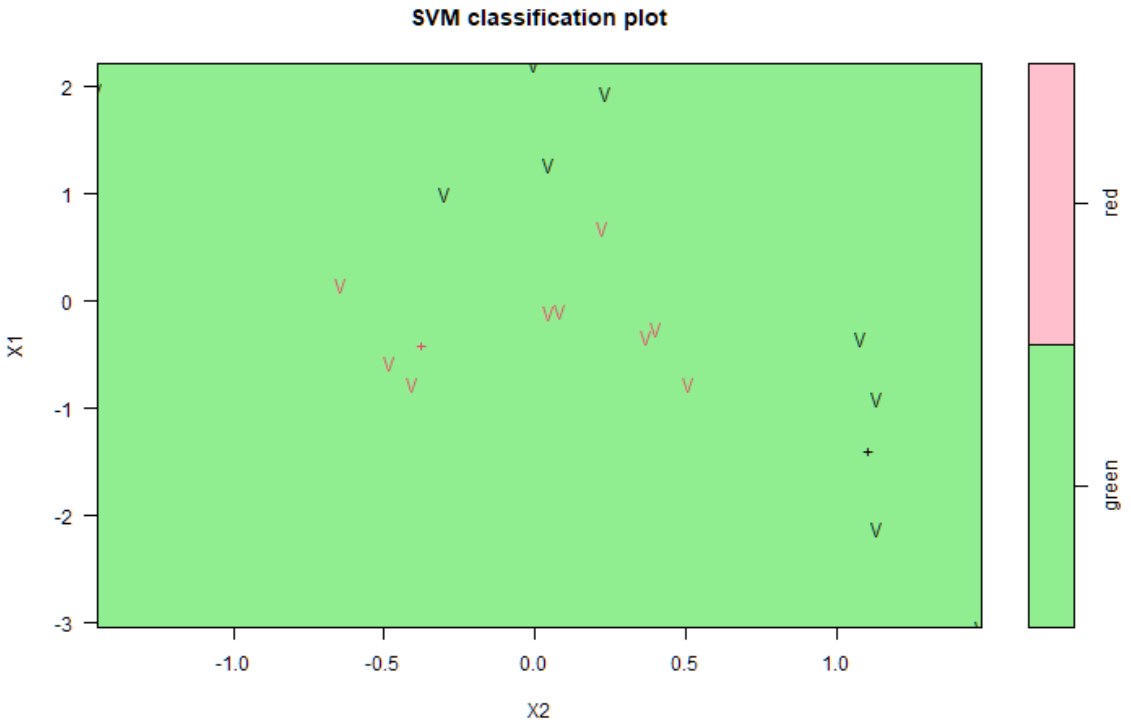

Рисунок 11. Классификация SVM с полиномиальным ядром, d = 1

Рисунок 12. Классификация SVM с полиномиальным ядром, d = 2

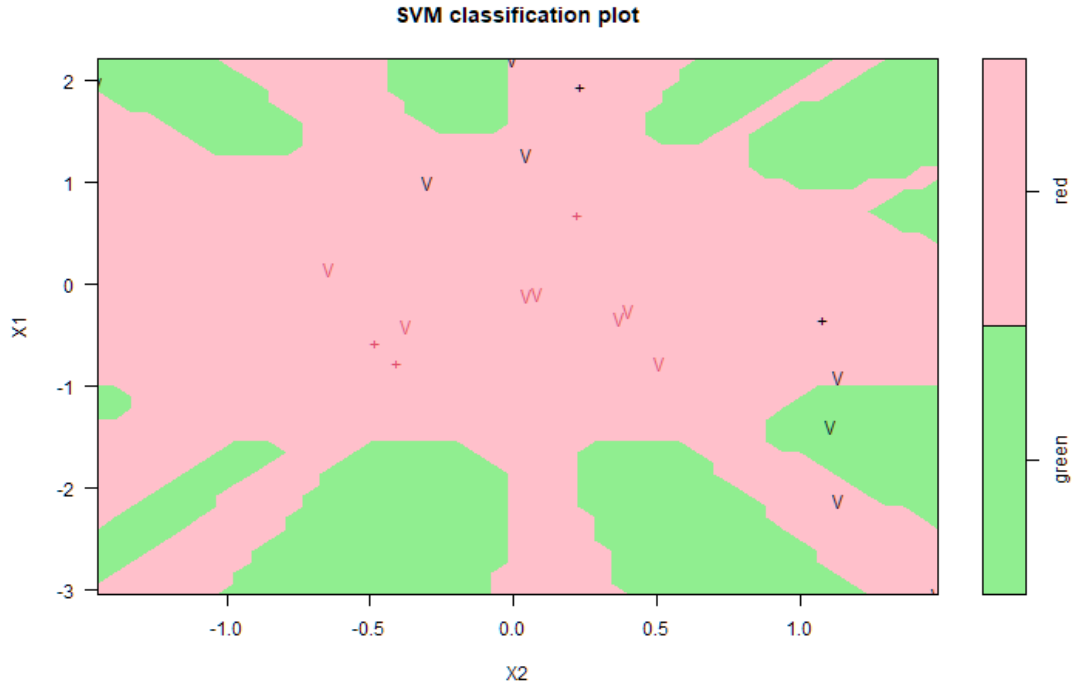

Рисунок 13. Классификация SVM с полиномиальным ядром, d = 25

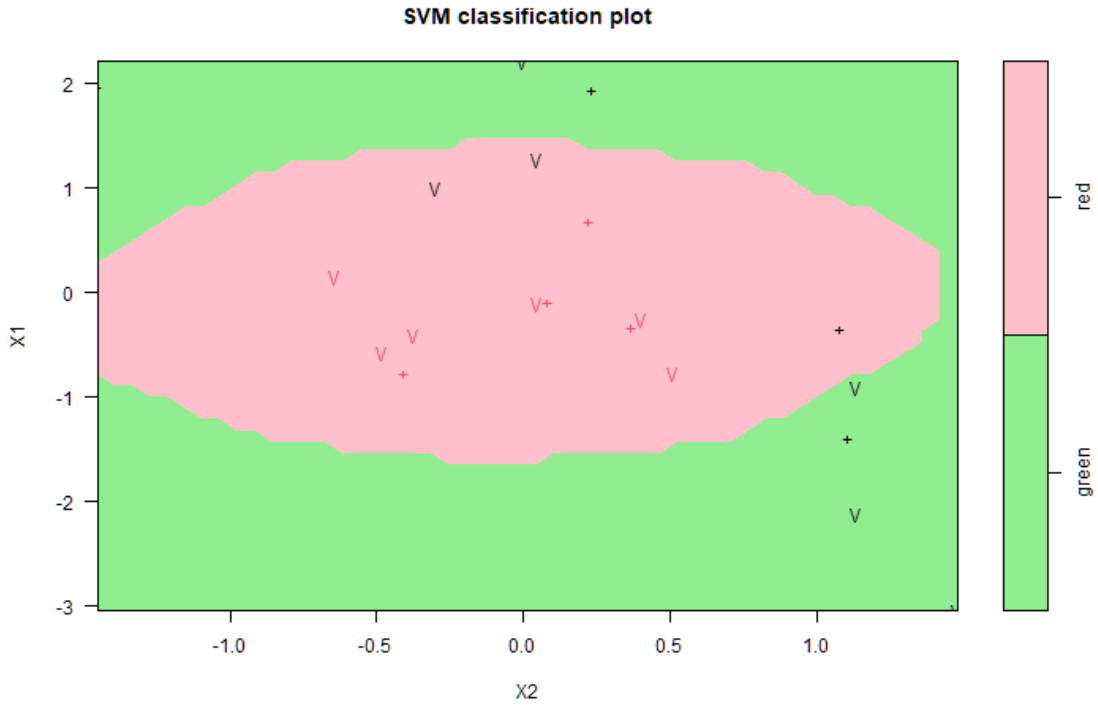

Рисунок 14. Классификация SVM с полиномиальным ядром, d = 50

Наибольшая достоверность классификации тестовых данных (94%) достигается при ядре типа «radial», для ядра типа «polynomial» наибольшая достоверность (86%) достигается при значении параметра degree равном 2, увеличение значения степени снижает точность, наихудшие результаты классификации оказались при ядре типа «sigmoid» - 58%.

Задание №4

В данном задании использовались аналогичные модели задания 3 для датасета svmdata4.txt и svmdata4test.txt.

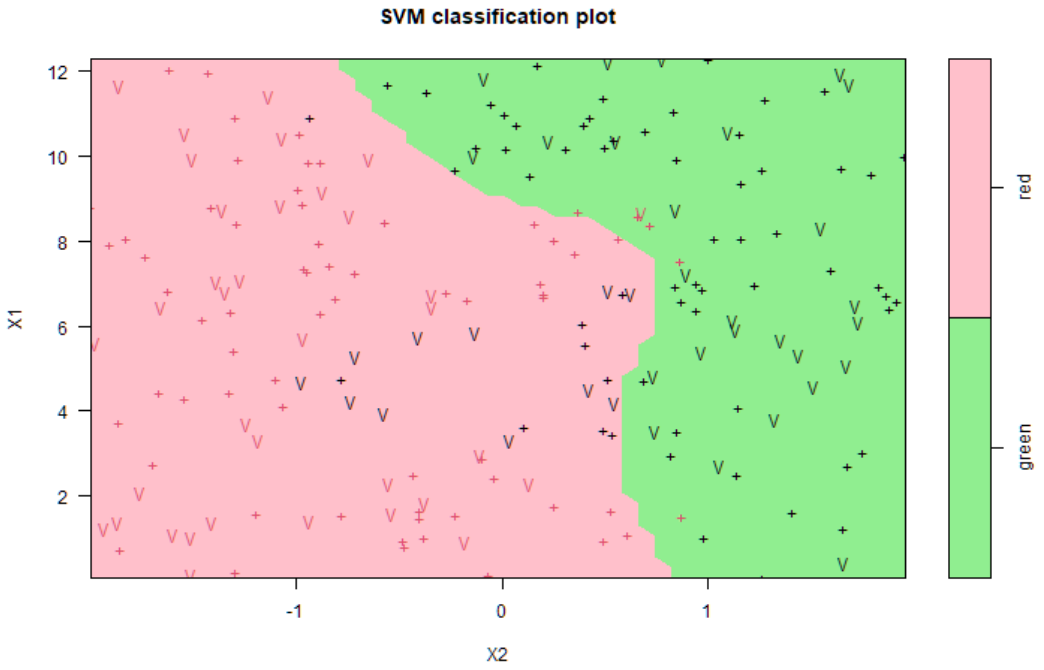

Рисунок 15. Классификация SVM с полиномиальным ядром

Точность классификации – 87%.

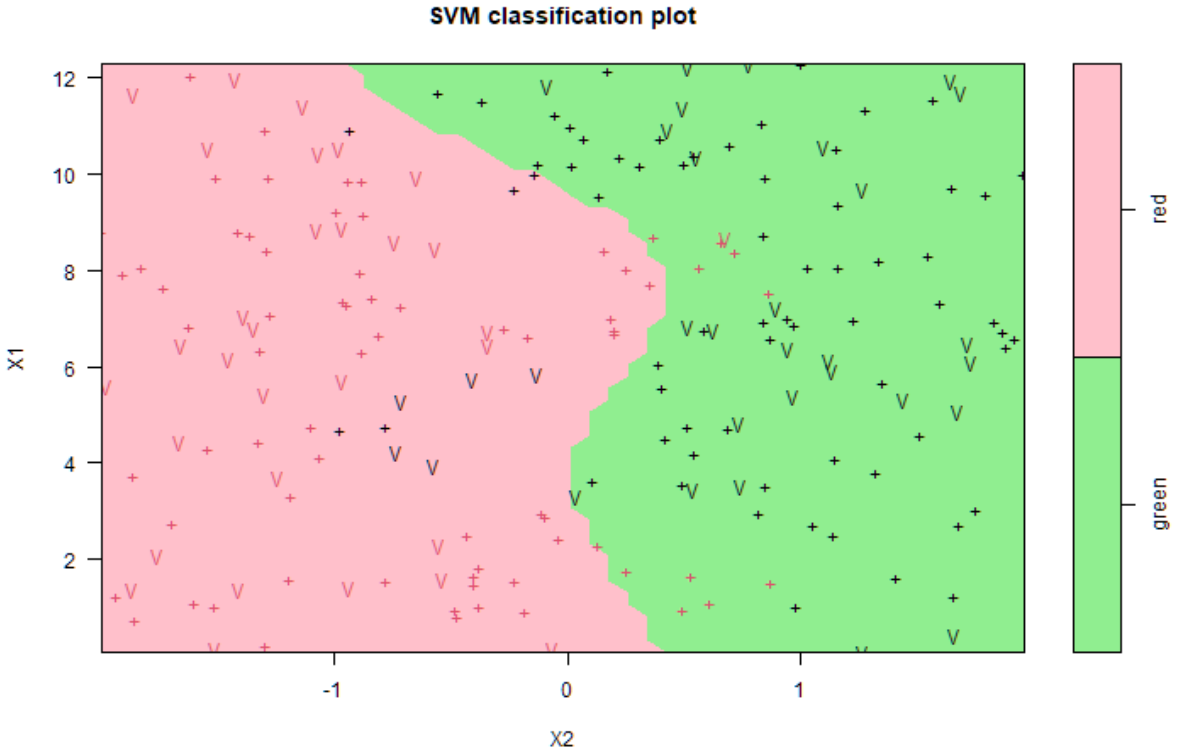

Рисунок 16. Классификация SVM с радиальным ядром

Точность классификации – 89%.

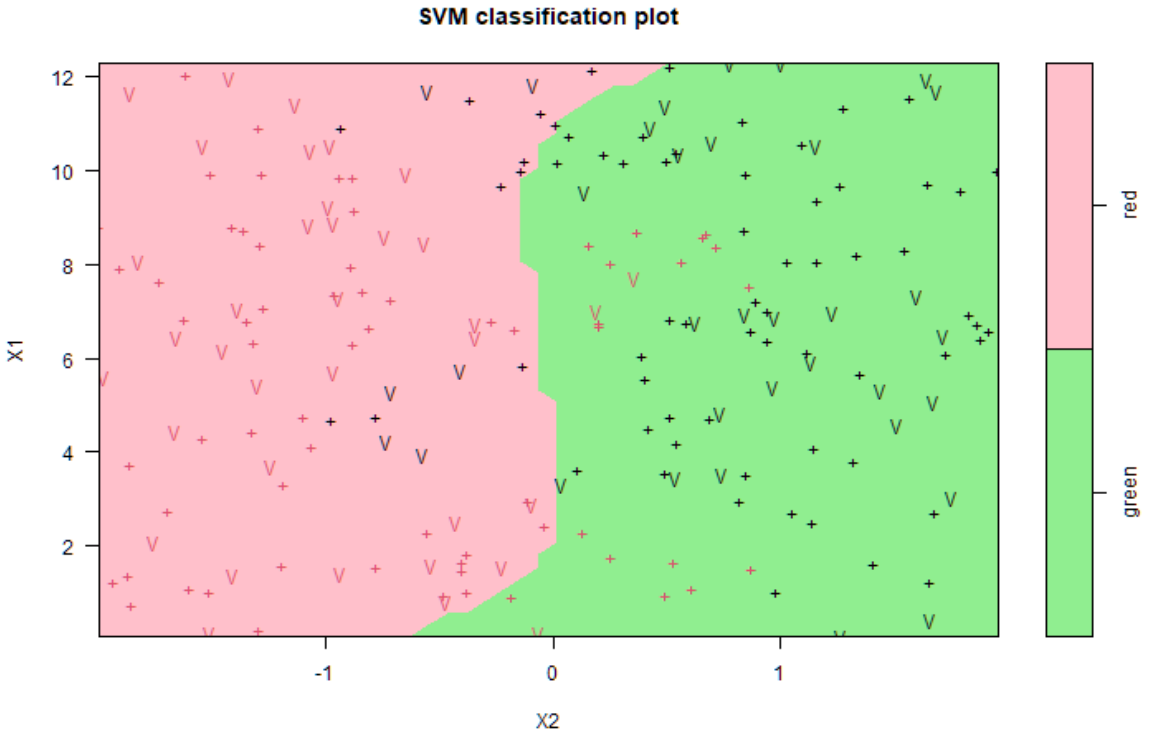

Рисунок 17. Классификация SVM с сигмоидным ядром

Точность классификации – 80.5%.

Наилучшую точность классификации показало радиальное ядро. Все алгоритмы запускались с параметром штрафа, равным 1.

Задание №5

Полиномиальное ядро

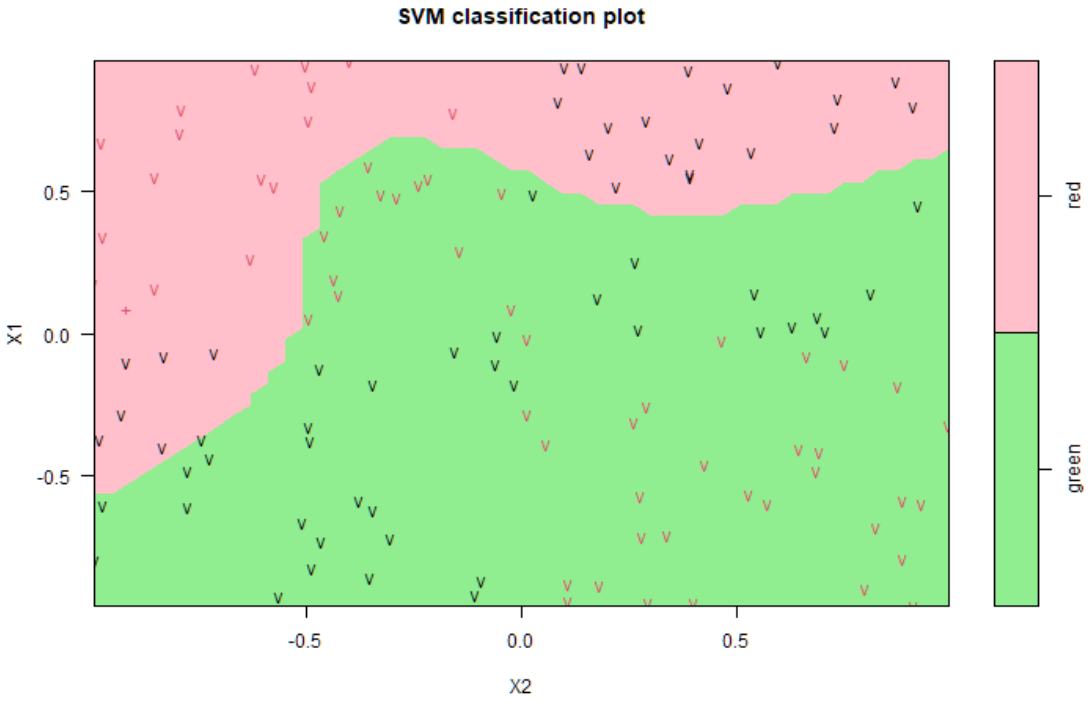

Рисунок 18. Полиномиальное ядро, гамма = 1

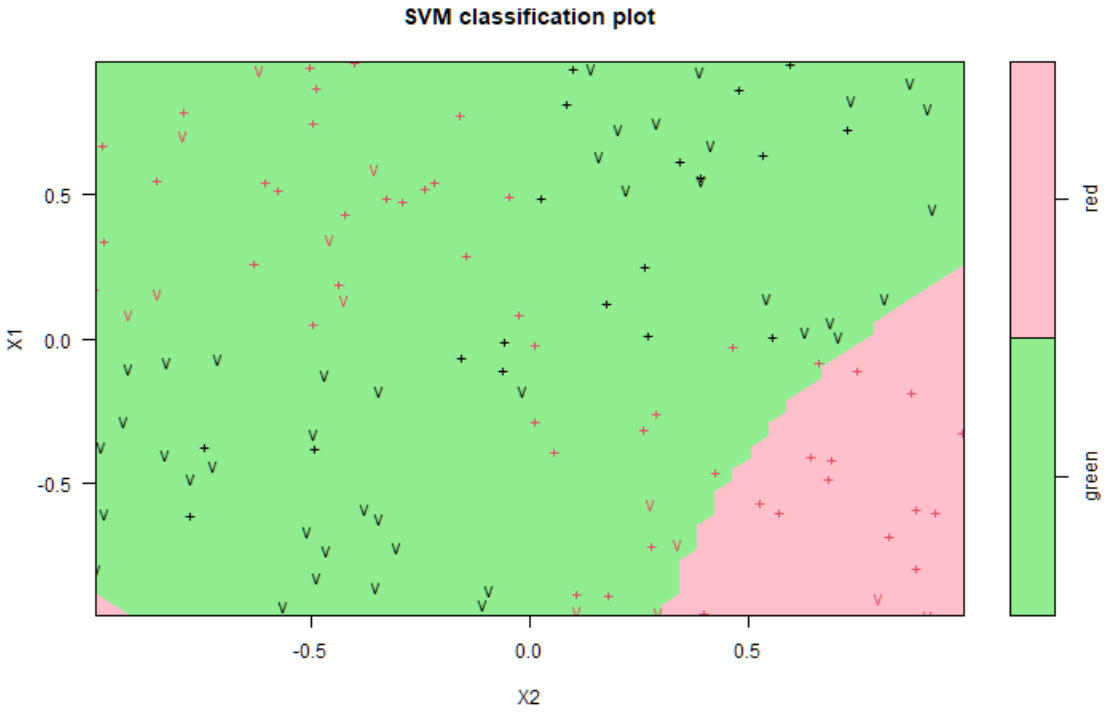

Рисунок 19. Полиномиальное ядро, гамма = 50

Точность классификации – 42.5% и 63.3%.

Радиальное ядро

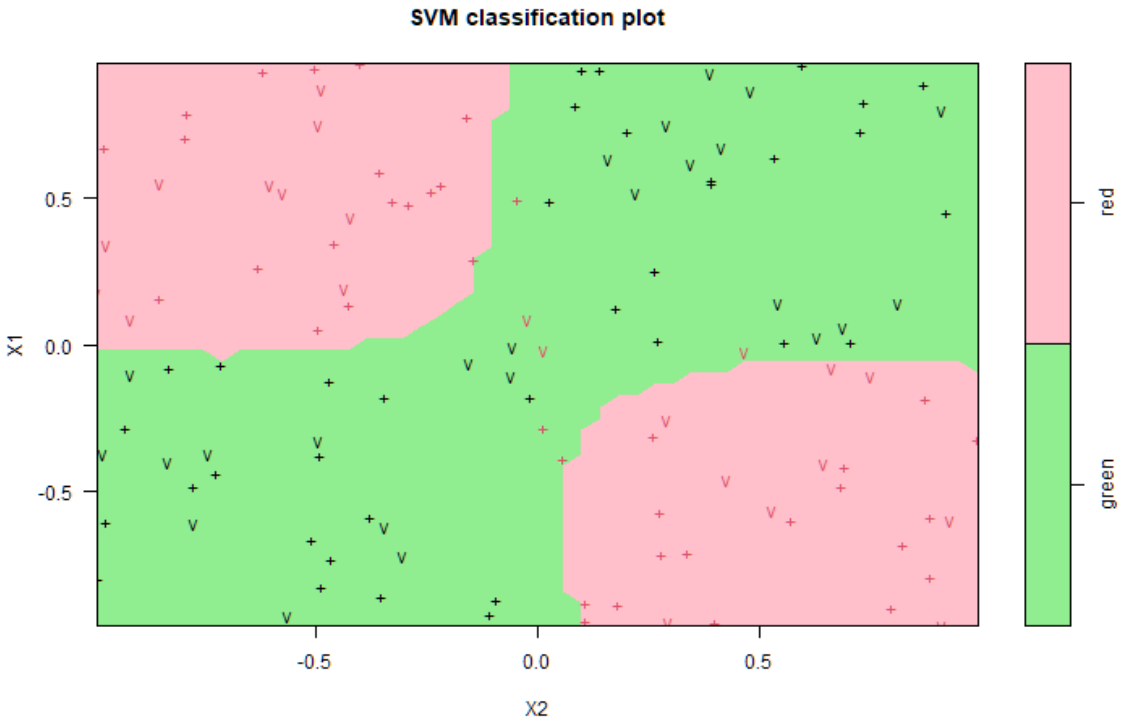

Рисунок 20. Радиальное ядро, гамма = 1

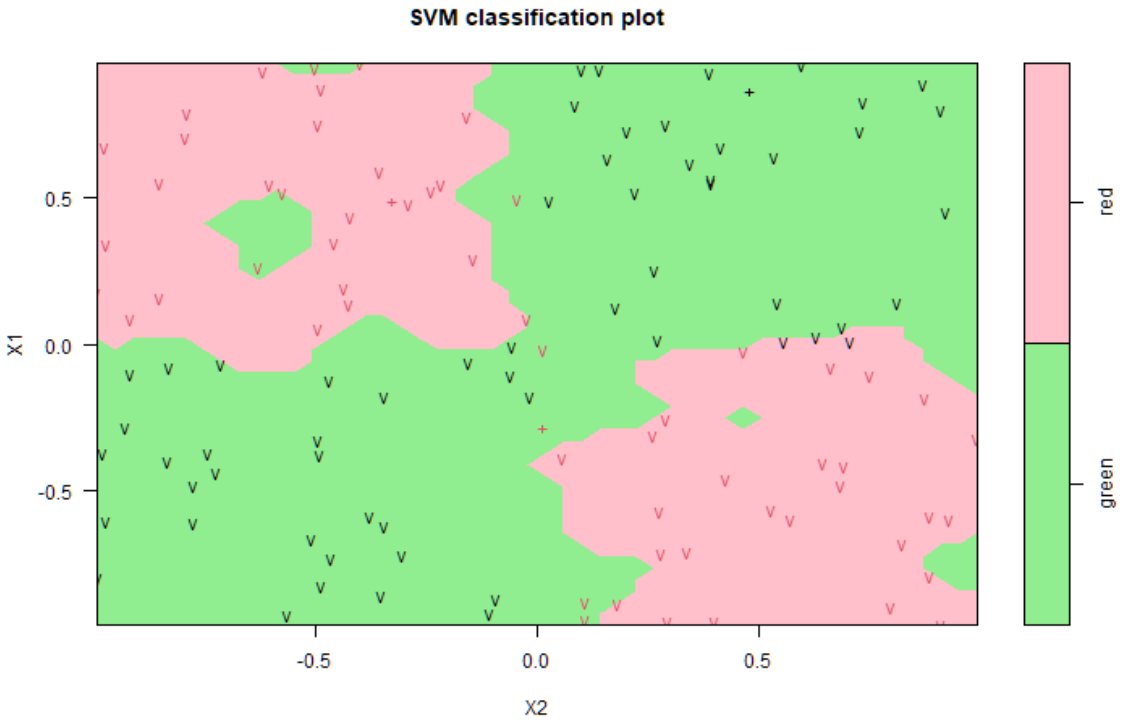

Рисунок 21. Радиальное ядро, гамма = 50

Точность классификации – 95.83% и 90.8%.

Сигмоидное ядро

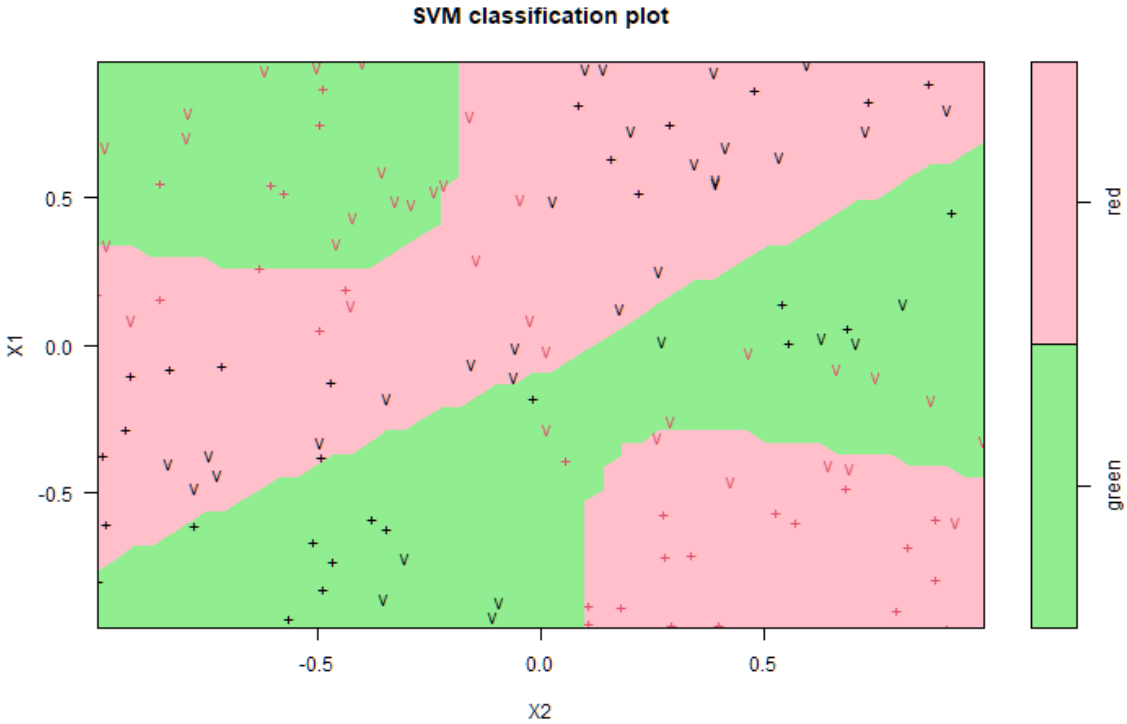

Рисунок 22. Сигмоидное ядро, гамма = 1

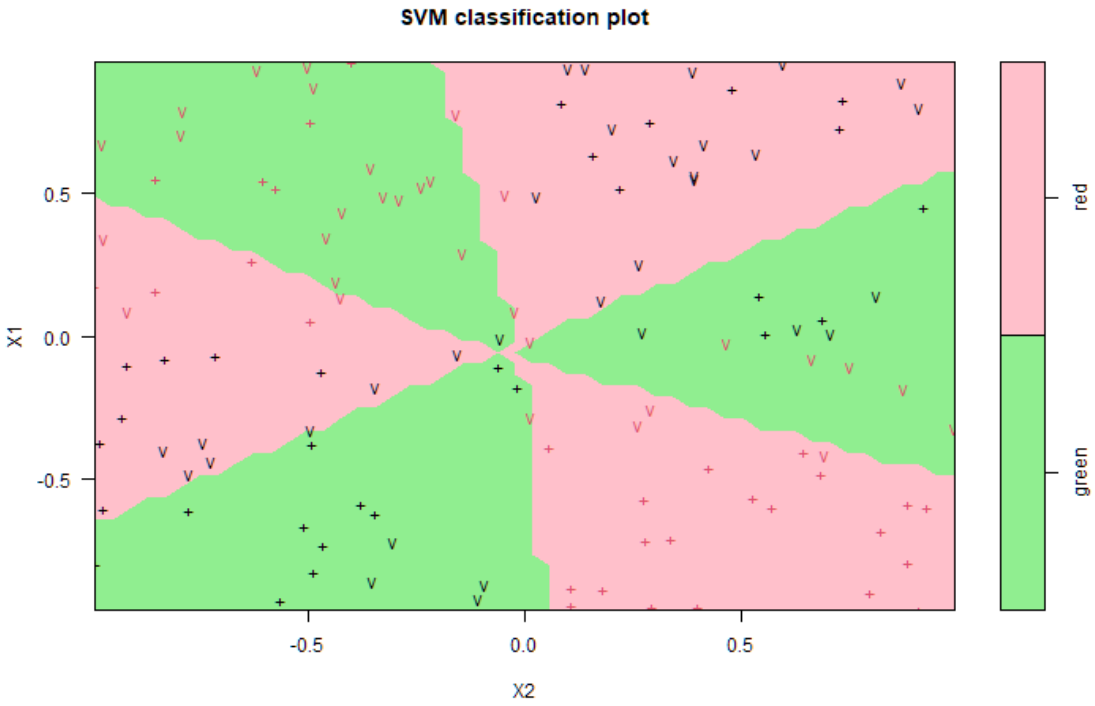

Рисунок 23. Сигмоидное ядро, гамма = 50

Точность классификации – 45.83% и 48.3%.

Можем заметить, что наилучший результат показывает радиальное (гауссово) ядро при параметре гамма равном 1. Эффект переобучения для радиального ядра заметен при параметре гамма, равном 50.

Задание №6

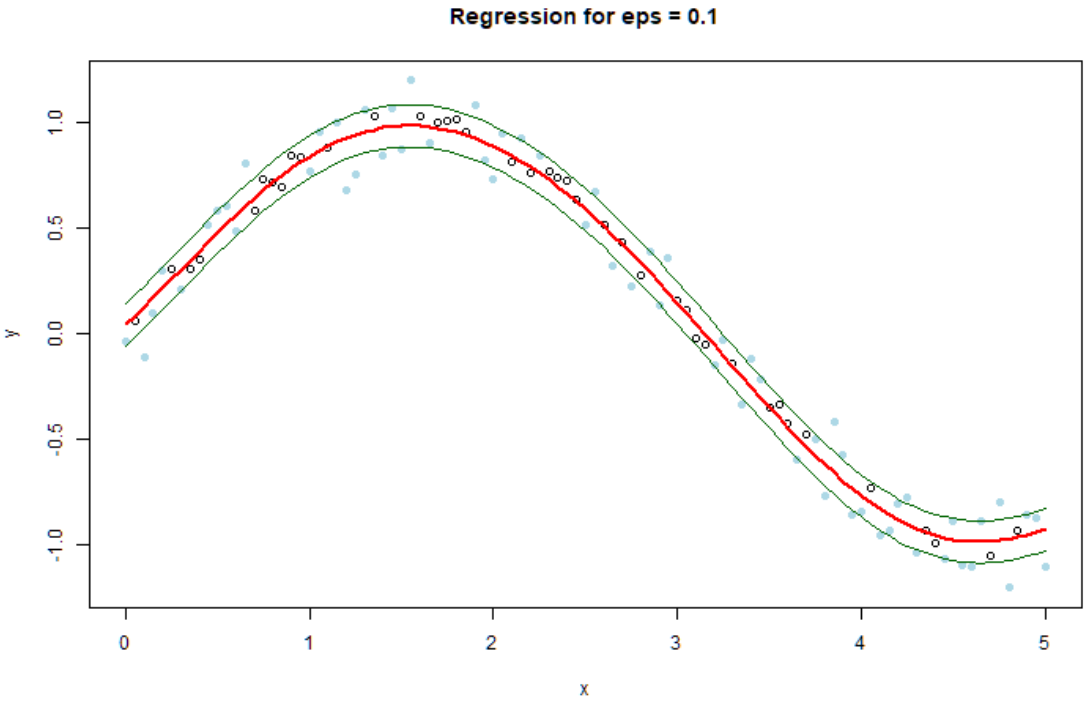

Рисунок 24. Построенная регрессия для eps = 0.1

MSE = 0.01064825

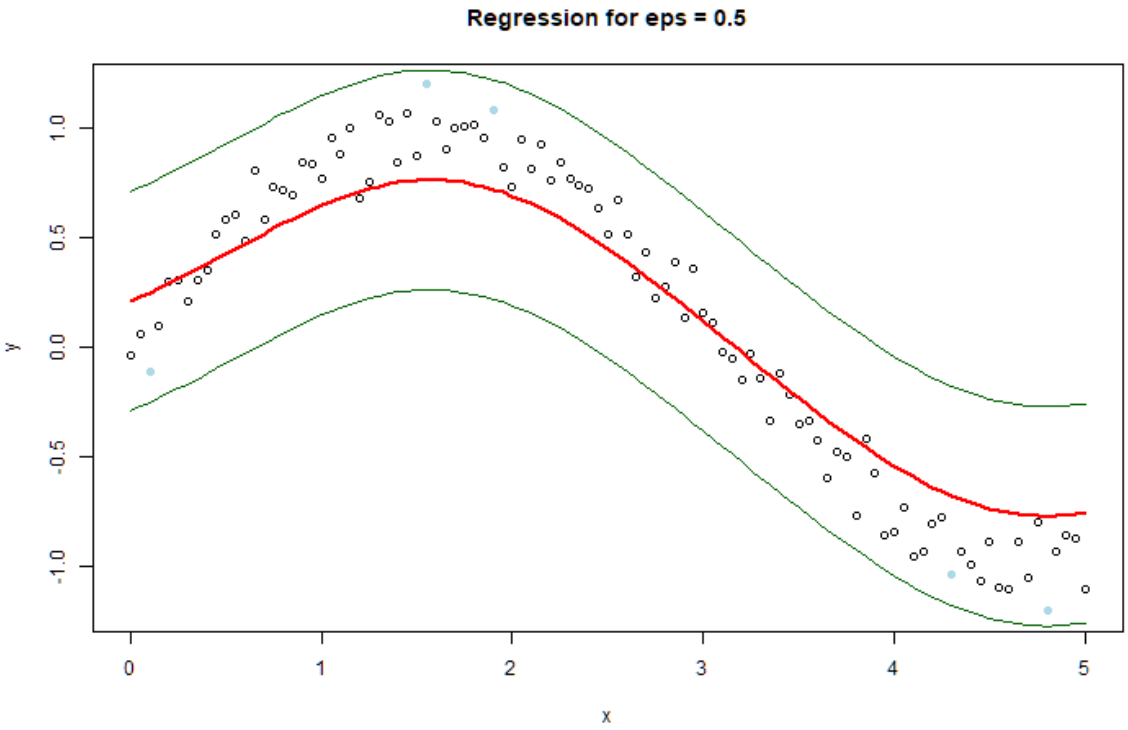

Рисунок 25. Построенная регрессия для eps = 0.5

MSE = 0.04302801

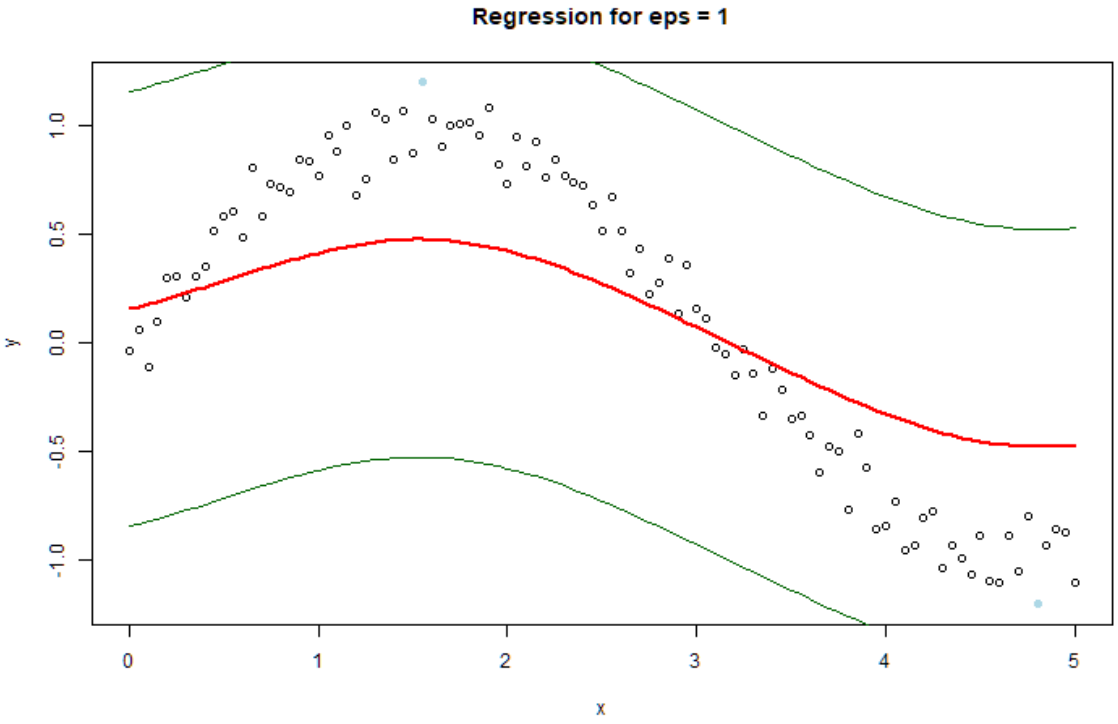

Рисунок 26. Построенная регрессия для eps = 1

MSE = 0.15815918

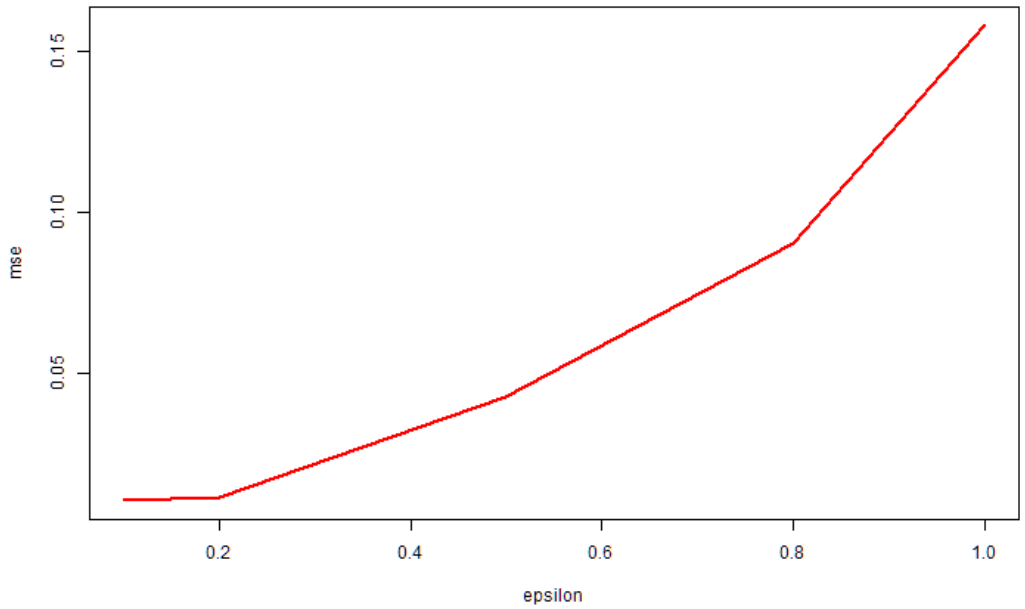

Рисунок 27. Зависимость MSE от eps

По рисунку 27 можем заметить, что среднеквадратичное отклонение экспоненциально возрастает с ростом ε, так как с увеличением ε растёт и доверительная полоса для линии регрессии

Наименьшую ошибку показало значение $\varepsilon = 0.1$.

4. Вывод

В ходе выполнения лабораторной работы был освоен метод опорных векторов, реализованный в пакете e1071 языка R. SVM хорошо работает как с задачами классификации, так и с восстановлением регрессии. Метод позволяет получить высокую точность классификации, однако требуется подбор параметров: ядра, степени полинома, и штрафного параметра.

Приложение 1

```
# Задание 1----------------------------------------------------------------------
library(e1071)

data_train <- read.table(paste(path, "svmdata1.txt", sep =""), stringsAsFactors = TRUE)
data_test <- read.table(paste(path, "svmdata1test.txt", sep =""), stringsAsFactors = TRUE)

X <- data.frame(X1 = data_test$X1, X2 = data_test$X2)
Y <- data.frame(color = data_test$Color)

model <- svm(Color ~ ., kernel = "linear", data = data_train, type="C-classification", cost=1)

predicted <- predict(model, X)

table(data_test$Color, predicted)

png(paste(path, "svmdata1.png"), width = 720, height = 480)
plot(model, data = data_test,
    col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
dev.off()
```

Приложение 2

```
# Задание 2------------------------------------------------------------------
library(e1071)

data_train <- read.table(paste(path, "svmdata2.txt", sep =""), stringsAsFactors = TRUE)
data_test <- read.table(paste(path, "svmdata2test.txt", sep =""), stringsAsFactors = TRUE)

X <- data.frame(X1 = data_test$X1, X2 = data_test$X2)
Y <- data.frame(colors = data_test$Colors)

C_param <- c(1, 10, 50, 100, 500, 1000)

tbl <- list()

for (c in C_param)
{
  model <- svm(Colors ~ ., kernel = "linear", data=data_train, type="C-classification", cost=c)
  print(summary(model))
  predicted <- predict(model, X)
  tbl <- append(tbl, list(table(data_test$Colors, predicted)))
  png(paste(path, "Svmdata2 C=", c, ".png", sep = ""), width = 720, height = 480)
  plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
  dev.off()
}
print(tbl)
```

Приложение 3

```
# Задание 3----------------------------------------------------------------------
library(e1071)

data <- read.table(paste(path, "svmdata3.txt", sep =""), stringsAsFactors = T)

ratio <- 0.8
n <- nrow(data)
nt <- as.integer(n * ratio)

data_rand <- data[order(runif(n)), ]
data_train <- data_rand[1: nt, ]
data_test <- data_rand[(nt + 1): n, ]

X <- data.frame(X1 = data_test$X1, X2 = data_test$X2)
Y <- data.frame(Colors = data_test$Colors)

Degree = c(1, 5, 10, 25, 50)

tbl <- list()

for (d in Degree){

 model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "poly",
degree = d)
 print(summary(model))
 predicted <- predict(model, X)
 tbl <- append(tbl, list(table(data_test$Colors, predicted)))
 png(paste(path, "Svmdata3 Poly D=", d, ".png", sep = ""), width = 720, height = 480)
 plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
 dev.off()

}

model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "radial")
predicted <- predict(model, X)
table(data_test$Colors, predicted)
png(paste(path, "Svmdata3 radial.png", sep = ""), width = 720, height = 480)
plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
dev.off()


model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "sigmoid")
predicted <- predict(model, X)
table(data_test$Colors, predicted)
png(paste(path, "Svmdata3 sigmoid.png", sep = ""), width = 720, height = 480)
plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
dev.off()
```

Приложение 4

```
# Задание 4----------------------------------------------------------------------
library(e1071)

data_train <- read.table(paste(path, "svmdata4.txt", sep =""),  stringsAsFactors = T)
data_test <- read.table(paste(path, "svmdata4test.txt", sep =""), stringsAsFactors = T)

X <- data.frame(X1 = data_test$X1, X2 = data_test$X2)
Y <- data.frame(Colors = data_test$Colors)

table(data_test$Colors)

model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "radial")
predicted <- predict(model, X)
table(data_test$Colors, predicted)
png(paste(path, "Svmdata4 radial.png"), width = 720)
plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
dev.off()


model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "sigmoid")
predicted = predict(model, X)
table(data_test$Colors, predicted)
png(paste(path, "Svmdata4 sigmoid.png"), width = 720)
plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
dev.off()


model <- svm(Colors ~ ., data=data_train, type="C-classification", cost=1, kernel = "poly")
predicted <- predict(model, X)
table(data_test$Colors, predicted)
png(paste(path, "Svmdata4 poly.png"), width = 720)
plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "V")
dev.off()
```

Приложение 5

```
# Задание 5----------------------------------------------------------------
library(e1071)

data_train <- read.table(paste(path, "svmdata5.txt", sep =""), stringsAsFactors = T)
data_test <- read.table(paste(path, "svmdata5test.txt", sep=""), stringsAsFactors = T)
X <- data.frame(X1 = data_test$X1, X2 = data_test$X2)
Y <- data.frame(Colors = data_test$Colors)

table(data_test$Colors)

gamma = c(1, 50)
tbl = list()
for (g in gamma){
  model <- svm(Colors ~ ., data=data_train, type="C-classification",
               cost=1, kernel = "poly", gamma = g)
  predicted <- predict(model, X)
  tbl <- append(tbl, list(table(data_test$Colors, predicted)))
  png(paste(path, "Svmdata5 poly, gamma=", g, ".png", sep = ""), width = 720, height = 480)
  plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "v")
  dev.off()
}
print(tbl)

tbl <- list()

for (g in gamma){
  model <- svm(Colors ~ ., data=data_train, type="C-classification",
               cost=1, kernel = "radial", gamma = g)
  predicted <- predict(model, X)
  tbl <- append(tbl, list(table(data_test$Colors, predicted)))
  png(paste(path, "Svmdata5 radial, gamma=", g, ".png", sep = ""), width = 720, height = 480)
  plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "v")
  dev.off()
}
print(tbl)

tbl <- list()

for (g in gamma){
  model <- svm(Colors ~ ., data=data_train, type="C-classification",
               cost=1, kernel = "sigmoid", gamma = g)
  predicted <- predict(model, X)
  tbl <- append(tbl, list(table(data_test$Colors, predicted)))
  png(paste(path, "Svmdata5 sigmoid, gamma=", g, ".png", sep = ""), width = 720, height = 480)
  plot(model, data_test, col = c("lightgreen", "pink"), dataSymbol = "+", svSymbol = "v")
  dev.off()
}
print(tbl)
```

Приложение 6

```
# Задание 6-----------------------------------------------------------------
library(e1071)
library(Metrics)

data_train <- read.table(paste(path, "svmdata6.txt", sep =""), stringsAsFactors = TRUE)

x = c(X = data_train$X)
y = c(Y = data_train$Y)

eps <- c(0.1, 0.2, 0.5, 0.8, 1)

tbl <- vector()

for(e in eps)
{
  model = svm(x, y, type="eps-regression", eps=e, kernel = "radial", cost = 1)

  predcted = predict(model, x)

  png(paste(path, "Regression for eps =", e, ".png", sep = ""), width = 720, height = 480)

  plot(x, y, main = paste("Regression for eps =", e))

  points(x[model$index], y[model$index], col = "lightblue", pch = 19)

  lines(x, predcted, col = "red", lwd = 2)
  lines(x, predcted + model$epsilon, col = "darkgreen")
  lines(x, predcted - model$epsilon, col = "darkgreen")

  dev.off()

  tbl <- append(tbl, mse(predcted, y))
}

print(tbl)

png(paste(path, "MSE and Eps.png"), width = 720, height = 480)
plot(x = eps, y = tbl, type = "l", xlab = "epsilon", ylab = "mse", col = "red", lwd = 2)
dev.off()
```