Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Пермский государственный аграрно-технологический университет

имени академика Д.Н. Прянишникова»

Кафедра информационных технологий и программной инженерии

**ЛАБОРАТОРНАЯ РАБОТА № 2**

Линейная классификация.

|  |  |
| --- | --- |
| Выполнил: | группа ПИНб-4  П.С. Плотников |
| Проверил: | доцент каф. ИТиПИ,  Т.А. Казаченко |

Пермь-2024 г.

# Ответы на вопросы

1. Ответьте на вопросы: **Таблица 1**

|  |  |  |
| --- | --- | --- |
| **№** | **Вопрос** | **Ответ** |
|  | Как называется метод, изучаемый в лабораторной 2? | Линейная классификация *Линейная бинарная классификация* |
|  | К какой категории методов ML он относится? | Обучение с учителем |
|  | Что может прогнозировать алгоритм, обученный данным методом? | Соответствие объекта какому-то классу |
|  | Что такое опорные вектора? | Точки, которые находятся ближе всего к гиперплоскости |
|  | Что должен построить алгоритм SVM при обучении? | Гиперплоскость, которая разделяет выборку |
|  | Что определяет знак ***margin*** в SVM? | К какому классу относится объект *+ правильная классификация, - неправильная классификация* |
|  | Запишите функцию потерь в задаче линейной бинарной классификации **в РАЗВЕРНУТОМ ВИДЕ** | Изображение выглядит как текст, рукописный текст, линия, рукописный  Автоматически созданное описание |
|  | Запишите ядро "linear" для парной классификации. | Изображение выглядит как текст, рукописный текст, линия  Автоматически созданное описание |

**Задание №1**

Данные для обучения и тестирования SVM-моделей, которые необходимо построить в приведеном ниже задании, храняться в файлах с именами svmdata1.txt и svmdata1test.txt.

1) Постройте машину опорных векторов для логистической функции потерь.

2) Аналогично рисунку 1 визуализируйте разбиение пространства признаков на области с помощью полученных алгоритмов.

3) Выведите количество полученных опорных векторов

Изображение выглядит как текст, диаграмма, снимок экрана, График

Автоматически созданное описание

Рисунок 1 – Пример визуализации разбиения пространства

**Программный код**

# Подключение библиотеки

library(ggplot2)

# Загрузка данных для обучения

svmdata <- read.table("./svmdata1.txt", header = TRUE)

# Загрузка тестовых данных

svmdata\_test <- read.table("./svmdata1test.txt", header = TRUE)

# Замена класса "red" на -1, а "green" на 1 в данных

svmdata$Color <- ifelse(svmdata$Color == "red", -1, ifelse(svmdata$Color == "green", 1, NA))

svmdata\_test$Color <- ifelse(svmdata\_test$Color == "red", -1, ifelse(svmdata\_test$Color == "green", 1, NA))

# Функция вычисления экспоненты

exponent\_func <- function(w, y, x1, x2) {

exp(-y \* (w[1] + w[2] \* x1 + w[3] \* x2))

}

# Функция ошибки

error\_func <- function(w, y, x1, x2, l) {

(1/l)\*sum(log(1 + exponent\_func(w, y, x1, x2)))

}

# Градиент

gradient <- function(w, y, x1, x2, l) {

exp\_vals <- exponent\_func(w, y, x1, x2)

df\_w0 <- (1/l)\*sum((-y \* exp\_vals) / (1 + exp\_vals))

df\_w1 <- (1/l)\*sum((-y \* x1 \* exp\_vals) / (1 + exp\_vals))

df\_w2 <- (1/l)\*sum((-y \* x2 \* exp\_vals) / (1 + exp\_vals))

c(df\_w0, df\_w1, df\_w2)

}

# Градиентный спуск

gradient\_descent <- function(start, dataset, rate = 0.001, tol = 1e-6, max\_steps = 4000000) {

w <- start

l <- nrow(dataset)

iter <- 0

repeat {

iter <- iter + 1

grad <- gradient(w, dataset$Color, dataset$X1, dataset$X2, l)

w\_new <- w - rate \* grad

if (sqrt(sum((w\_new - w)^2)) < tol || iter >= max\_steps) {

w <- w\_new

break

}

w <- w\_new

}

list(

w = w,

steps = iter,

error = round(error\_func(w, dataset$Color, dataset$X1, dataset$X2, l), 4)

)

}

# Запуск градиентного спуска для обучения на svmdata

start\_point <- c(0, 0, 0)

result <- gradient\_descent(start\_point, svmdata)

# Подсчет маржи для каждой точки в обучающих данных

svmdata$margin <- svmdata$Color \* (result$w[1] + result$w[2] \* svmdata$X1 + result$w[3] \* svmdata$X2)

# Определение опорных векторов

support\_vectors <- svmdata[abs(svmdata$margin - 1) < 3.745, ]

# Создаение сетки значений для X1 и X2 для визуализации

x1\_range <- seq(min(svmdata$X1) - 1, max(svmdata$X1) + 1, length.out = 200)

x2\_range <- seq(min(svmdata$X2) - 1, max(svmdata$X2) + 1, length.out = 200)

grid <- expand.grid(X1 = x1\_range, X2 = x2\_range)

# Вычисление предсказанных классов для сетки

grid$Prediction <- ifelse(result$w[1] + result$w[2] \* grid$X1 + result$w[3] \* grid$X2 > 0, 1, -1)

# Построение графика

ggplot() +

geom\_tile(data = grid, aes(x = X1, y = X2, fill = factor(Prediction)), alpha = 0.3) +

scale\_fill\_manual(values = c("-1" = "red", "1" = "green"), name = "Класс") +

geom\_point(data = svmdata\_test, aes(x = X1, y = X2, color = factor(Color)), size = 3) +

scale\_color\_manual(values = c("-1" = "red", "1" = "green"), name = "Истинный класс (тест)") +

geom\_abline(intercept = -result$w[1] / result$w[3], slope = -result$w[2] / result$w[3], color = "blue", size = 1, linetype = "dashed") +

labs(title = "SVM с линейной функцией ошибки", x = "X1", y = "X2") +

theme\_minimal()

cat("w: ", result$w, "\nКол-во шагов: ", result$steps, "\nФункция ошибки", result$error, "\n")

cat("Количество опорных векторов:", nrow(support\_vectors), "\n")

**Результат вычислений**

Изображение выглядит как текст, диаграмма, линия, График

Автоматически созданное описание

Рисунок 2 – Визуализация разбиения с линейной функцией ошибки

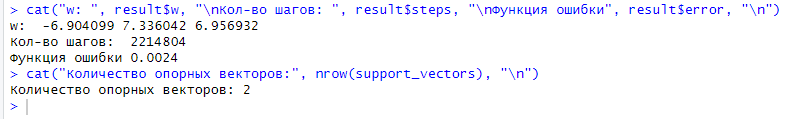


Рисунок 3 – Вывод результата в консоли

**Задание №2**

Данные для обучения и тестирования SVM-моделей, которые необходимо построить в приведеном ниже задании, храняться в файлах с именами svmdata1.txt и svmdata1test.txt.

1) Постройте машину опорных векторов для экспоненциальной функции потерь.

2) Аналогично рисунку 1 визуализируйте разбиение пространства признаков на области с помощью полученных алгоритмов.

3) Выведите количество полученных опорных векторов.

**Программный код**

# Подключение библиотеки

library(ggplot2)

# Загрузка данных

svmdata <- read.table("./svmdata1.txt", header = TRUE)

svmdata\_test <- read.table("./svmdata1test.txt", header = TRUE)

# Замена класса "red" на -1, а "green" на 1 в данных

svmdata$Color <- ifelse(svmdata$Color == "red", -1, ifelse(svmdata$Color == "green", 1, NA))

svmdata\_test$Color <- ifelse(svmdata\_test$Color == "red", -1, ifelse(svmdata\_test$Color == "green", 1, NA))

# Функция вычисления экспоненты

exponent\_func <- function(w, y, x1, x2) {

exp(-y \* (w[1] + w[2] \* x1 + w[3] \* x2))

}

# Функция ошибки

error\_func <- function(w, y, x1, x2, l) {

(1/l)\*sum(exponent\_func(w, y, x1, x2))

}

# Градиент

gradient <- function(w, y, x1, x2, l) {

exp\_vals <- exponent\_func(w, y, x1, x2)

df\_w0 <- (1/l)\*sum((-y \* exp\_vals))

df\_w1 <- (1/l)\*sum((-y \* x1 \* exp\_vals))

df\_w2 <- (1/l)\*sum((-y \* x2 \* exp\_vals))

c(df\_w0, df\_w1, df\_w2)

}

# Градиентный спуск

gradient\_descent <- function(start, dataset, rate = 0.001, tol = 1e-6, max\_steps = 4000000) {

w <- start

l <- nrow(dataset)

iter <- 0

repeat {

iter <- iter + 1

grad <- gradient(w, dataset$Color, dataset$X1, dataset$X2, l)

w\_new <- w - rate \* grad

if (sqrt(sum((w\_new - w)^2)) < tol || iter >= max\_steps) {

w <- w\_new

break

}

w <- w\_new

}

list(

w = w,

steps = iter,

error = round(error\_func(w, dataset$Color, dataset$X1, dataset$X2, l), 4)

)

}

# Запуск градиентного спуска для обучения на svmdata

start\_point <- c(0, 0, 0)

result <- gradient\_descent(start\_point, svmdata)

# Подсчет маржи для каждой точки в обучающих данных

svmdata$margin <- svmdata$Color \* (result$w[1] + result$w[2] \* svmdata$X1 + result$w[3] \* svmdata$X2)

# Определение опорных векторов

support\_vectors <- svmdata[abs(svmdata$margin - 1) < 3.75, ]

# Создание сетки значений для X1 и X2 для визуализации

x1\_range <- seq(min(svmdata$X1) - 1, max(svmdata$X1) + 1, length.out = 200)

x2\_range <- seq(min(svmdata$X2) - 1, max(svmdata$X2) + 1, length.out = 200)

grid <- expand.grid(X1 = x1\_range, X2 = x2\_range)

# Вычисление предсказанных класс для сетки

grid$Prediction <- ifelse(result$w[1] + result$w[2] \* grid$X1 + result$w[3] \* grid$X2 > 0, 1, -1)

# Отображение результата

ggplot() +

geom\_tile(data = grid, aes(x = X1, y = X2, fill = factor(Prediction)), alpha = 0.3) +

scale\_fill\_manual(values = c("-1" = "red", "1" = "green"), name = "Класс") +

geom\_point(data = svmdata\_test, aes(x = X1, y = X2, color = factor(Color)), size = 3) +

scale\_color\_manual(values = c("-1" = "red", "1" = "green"), name = "Истинный класс (тест)") +

geom\_abline(intercept = -result$w[1] / result$w[3], slope = -result$w[2] / result$w[3], color = "blue", size = 1, linetype = "dashed") +

labs(title = "SVM с экспоненциальной функцией ошибки", x = "X1", y = "X2") +

theme\_minimal()

cat("w: ", result$w, "\nКол-во шагов: ", result$steps, "\nФункция ошибки", result$error, "\n")

cat("Количество опорных векторов:", nrow(support\_vectors), "\n")

**Результат вычислений**

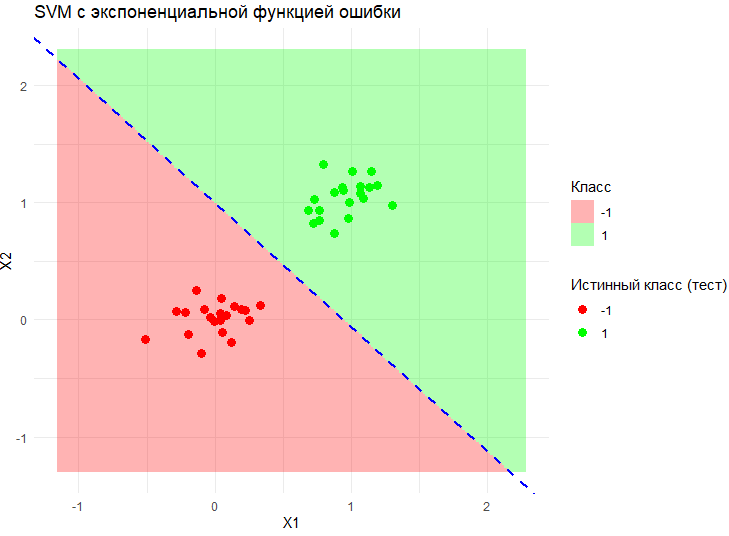


Рисунок 4 – Визуализация разбиения с экспоненциальной функцией

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 5 – Вывод результата в консоли

**Задание №3**

Данные для обучения и тестирования SVM-моделей, которые необходимо построить в приведеном ниже задании, храняться в файлах с именами svmdata1.txt и svmdata1test.txt.

1) Постройте машину опорных векторов "C-classification" с параметром C = 1, используя ядро "linear" с помощью стандартного пакета.

2) Аналогично рисунку 1 визуализируйте разбиение пространства признаков на области с помощью полученных алгоритмов.

3) Выведите количество полученных опорных векторов.

**Программный код**

# Загрузка необходимых пакетов

library(e1071)

library(ggplot2)

# Загрузка данных из файлов

train\_data <- read.table("./svmdata1.txt", header = TRUE)

test\_data <- read.table("./svmdata1test.txt", header = TRUE)

# Преобразуем метки классов в фактор

train\_data$Color <- as.factor(train\_data$Color)

test\_data$Color <- as.factor(test\_data$Color)

# Построение модели SVM с ядром "linear" и C=1

svm\_model <- svm(Color ~ X1 + X2, data = train\_data, kernel = "linear", cost = 1)

# Создание сетки для предсказания

grid <- expand.grid(X1 = seq(min(train\_data$X1), max(train\_data$X1), length = 200),

X2 = seq(min(train\_data$X2), max(train\_data$X2), length = 200))

# Прогнозирование на сетке

predictions <- predict(svm\_model, newdata = grid)

# Создание графика с использованием ggplot2

plot <- ggplot(train\_data, aes(x = X1, y = X2)) +

geom\_point(aes(color = Color), size = 3) +

geom\_raster(data = grid, aes(x = X1, y = X2, fill = predictions), interpolate = TRUE, alpha = 0.3) +

scale\_fill\_manual(values = c("red" = "red", "green" = "green"), name = "Класс по решению SVM") +

scale\_color\_manual(values = c("red" = "grey", "green" = "black"), name = "Цвет точек") + # Цвета точек отличны от классов

theme\_minimal() +

labs(title = "SVM с линейным ядром",

x = "X1", y = "X2")

print(plot)

cat("Количество опорных векторов:", nrow(svm\_model$SV), "\n")

**Результат вычислений**

Изображение выглядит как текст, снимок экрана, диаграмма, График

Автоматически созданное описание

Рисунок 6 – Визуализация со встроенными пакетами

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 7 – Результат в косноли

**Вывод**

В ходе лабораторной работы была проведена линейная классификация тремя способами. Использование готовых библиотек дало более точный и быстрый результат. В свою очередь, методы, которые программировались без готовых решений, помогли закрепить теоретические знания на практике.