САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ им. ПЕТРА ВЕЛИКОГО

Институт прикладной математики и механики

Высшая школа прикладной математики и вычислительной физики

Отчёт

по лабораторной работе №1 «Метод ближайших соседей» по дисциплине «Системы искусственного интеллекта»

Студентка гр. 3630201/70101 _____ О. В. Саксина

Преподаватель _____ Л. В. Уткин

Санкт-Петербург,
2021г.

Содержание

1 Постановка задачи

1.1 Задание 1

Исследовать, как объем обучающей выборки и количество тестовых данных, влияет на точность классификации или на вероятность ошибочной классификации в примере крестики-нолики и примере о спаме e-mail сообщений.

1.2 Задание 2

- Построить классификатор для обучающего множества Glass, данные которого характеризуются 10-ю признаками:

  1. Id number: 1 to 214;

  2. RI: показатель преломления;

  3. Na: сода (процент содержания в соотвествующем оксиде); 4. Mg;

  5. Al;

  6. Si;

  7. K;

  8. Ca;

  9. Ba;

  10. Fe.

  Классы характеризуют тип стекла:

  (1) окна зданий, плавильная обработка

  (2) окна зданий, не плавильная обработка

  (3) автомобильные окна, плавильная обработка

  (4) автомобильные окна, не плавильная обработка (нет в базе)

  (5) контейнеры

  (6) посуда

  (7) фары

- Построить графики зависимости ошибки классификации от значения k и от типа ядра.
- Исследовать, как тип метрики расстояния (параметр distance) влияет на точность классификации.
- Определить, к какому типу стекла относится экземпляр с характеристиками RI =1.516 Na =11.7 Mg =1.01 Al =1.19 Si =72.59 K=0.43 Ca =11.44 Ba =0.02 Fe =0.1
- Определить, какой из признаков оказывает наименьшее влияние на определение класса путем последовательного исключения каждого признака.

1.3 Задание 3

- Для построения классификатора использовать заранее сгенерированные обучающие и тестовые выборки, хранящиеся в файлах svmdata4.txt, svmdata4test.txt.
- Найти оптимальное значение k, обеспечивающее наименьшую ошибку классификации.
- Посмотреть, как выглядят данные на графике.

1.4 Задание 4

Разработать классификатор на основе метода ближайших соседей для данных Титаник (Titanic dataset) - https://www.kaggle.com/c/titanic

2 Реализация

2.1 Задание 1

2.1.1 Исследование влияния объёма обучающей выборки на результат

Для исследования влияния объёма обучающей выборки на точность классификации обучение производилось в несколько итераций, где в каждой итерации размер обучающей выборки менялся с 10% до 80% от размера всего датасета. Для получения усреднённого результата на каждой итерации было произведено 20 экспериментов, заключающихся в

1. формировании обучающей выборки, размер которой задан в начале итерации,
2. формировании тестовой выборки размера 100,
3. запуске процедуры обучения методом k ближайших соседей,
4. расчёте точности классификации, как отношения количества верных предсказаний к сумме всех предсказаний.

На рис. 1 приведён график зависимость точности предсказания от размера обучающей выборки для данных крестики-нолики.

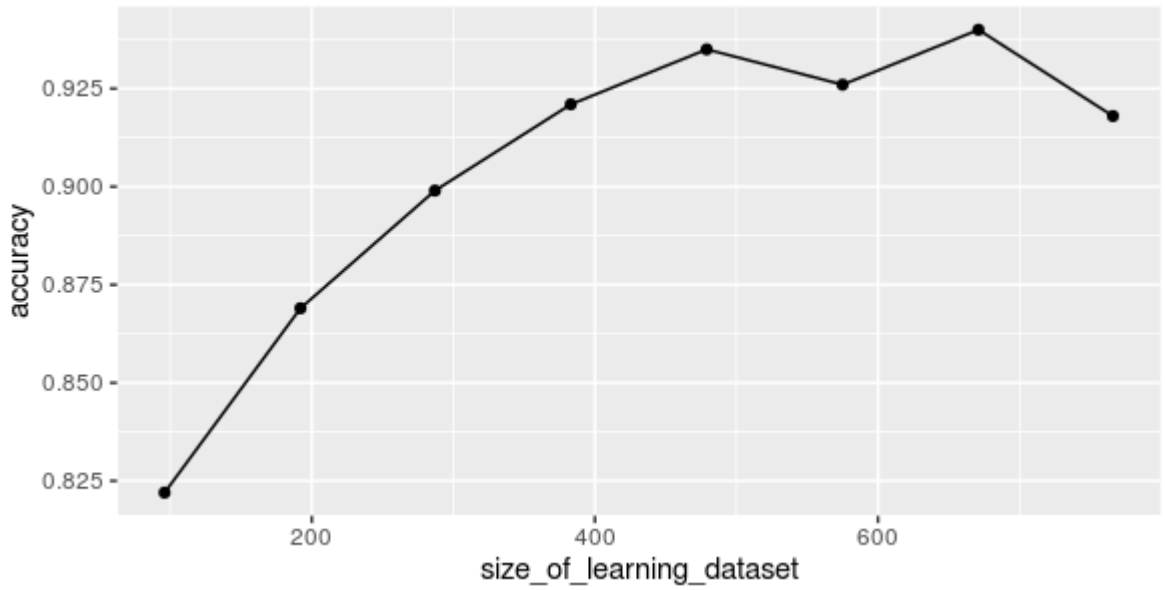

Рис. 1: Зависимость точности предсказания от размера обучающей выборки для данных крестики-нолики

По графику видно, что увеличение размера обучающей выборки до 500 даёт увеличение точности предсказания, дальнейшее увеличение может как улучшить, так и ухудшить результат, то есть достигнув достаточно больших размерах обучающей выборки не имеет смысла увеличивать его ещё больше. Также стоит заметить, что точность достигает практически 100% при больших размерах обучающей выборки, что скорее всего говорит о переобучении.
Рис. 2 представляет график зависимости точности предсказания от размера обучающей выборки для данных спам. Видно, что остались справедливыми выводы, сделанные для предыдущего примера.

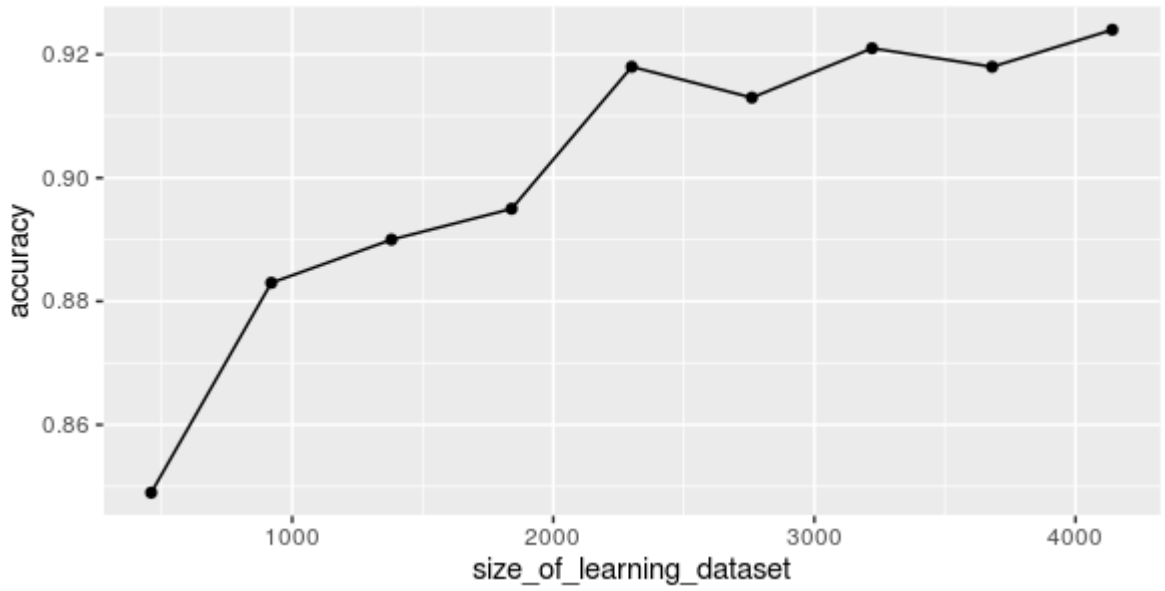

Рис. 2: Зависимость точности предсказания от размера обучающей выборки для данных спам

2.1.2 Исследование влияния объёма тестовой выборки на результат

Процедура проведения исследования влияния объёма тестовой выборки на результат аналогична описанной в предыдущем пункте, но в этом случае размер обучающей выборки равен 300, а размер тестовой выборки варьировался от 100 до 600. Результаты для примера крестики-нолики представлены на рис. 3.

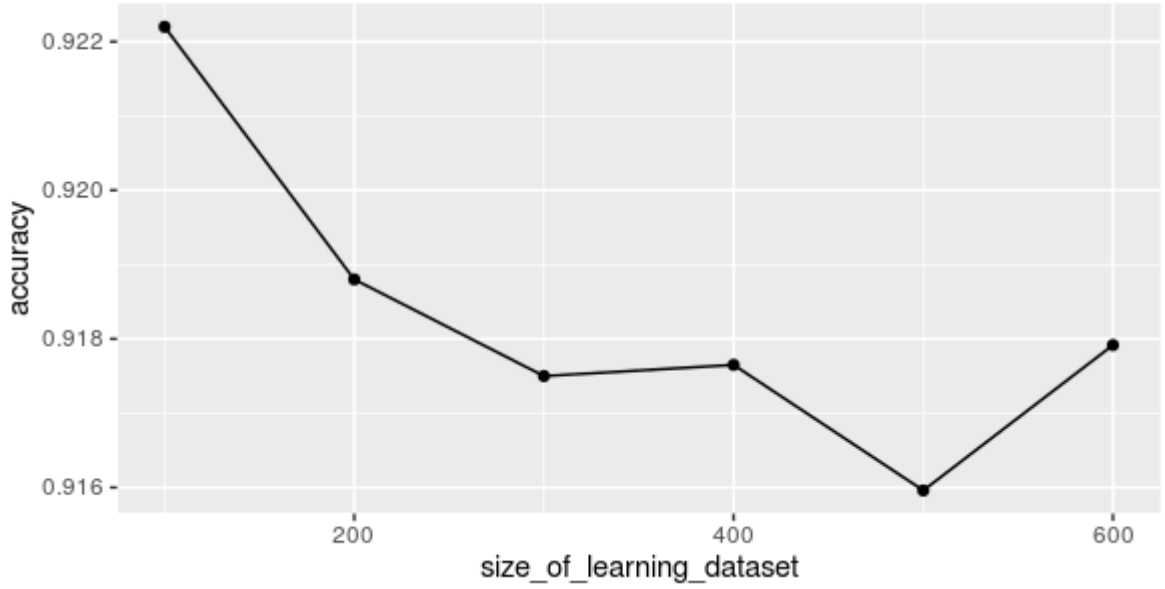

Рис. 3: Зависимость точности предсказания от размера тестовой выборки для данных крестики-нолики

Видно, что с увеличением размера тестовой выборки ухудшается точность результата. Это может быть связано с характеристикой данных, попавших в тестовую выборку: при её малом размере вероятность того, что попадутся примеры с выбросами или какими-то нетипичными параметрами, ниже. Такой случайностью выбора примеров в выборку можно объяснить и рост точности в конце при размере равном 600. Также стоит отметить небольшую величину разброса значений: разница между максимальной и минимальной точностью составила примерно 0.6%.
На рис. 4 представлен график зависимости точности предсказания от размера тестовой выборки для данных спам, где размер обучающей выборки составлял 2500.

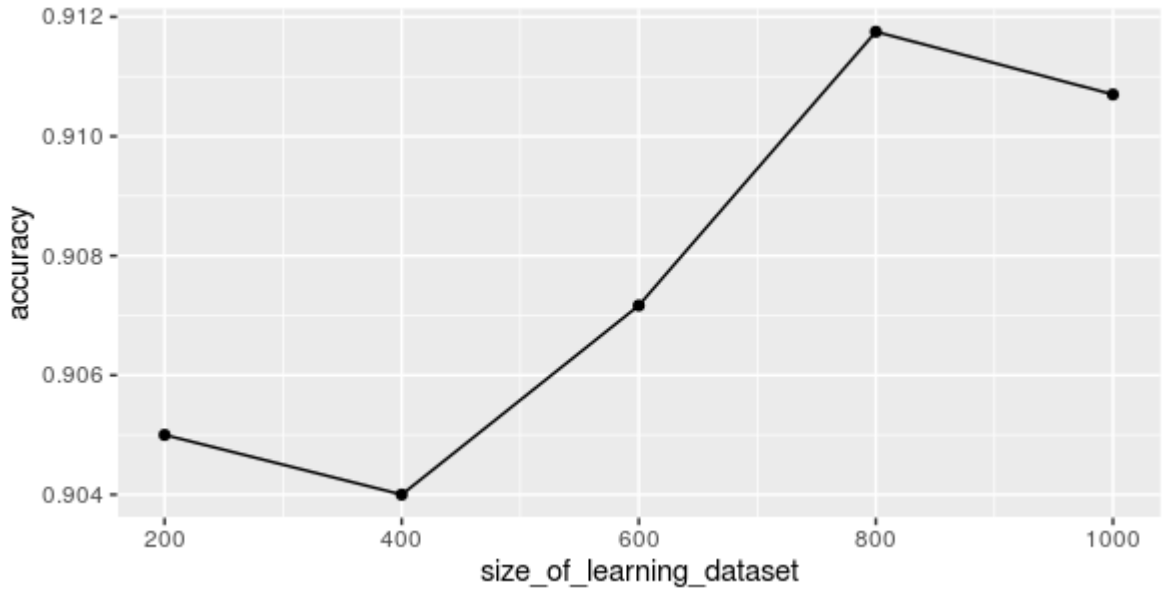

Рис. 4: Зависимость точности предсказания от размера тестовой выборки для данных спам

Здесь прослеживается обратная зависимость: чем больше размер тестовой выборки, тем больше точность. Это может объясняться тем, что в данных меньшего размера оказалось слишком много выбросов, а при увеличении выборки нормальные данные уменьшили влияние этих выбросов.

2.2 Задание 2

2.2.1 Исследование зависимости ошибки классификации от значения k и от типа ядра

На рис. 5 представлен график зависимости ошибки классификации от k и типа ядра. Лучшими параметрами получившегося классификатора, при которых ошибка равна 0.2616822, являются: k = 8, kernel = biweight.

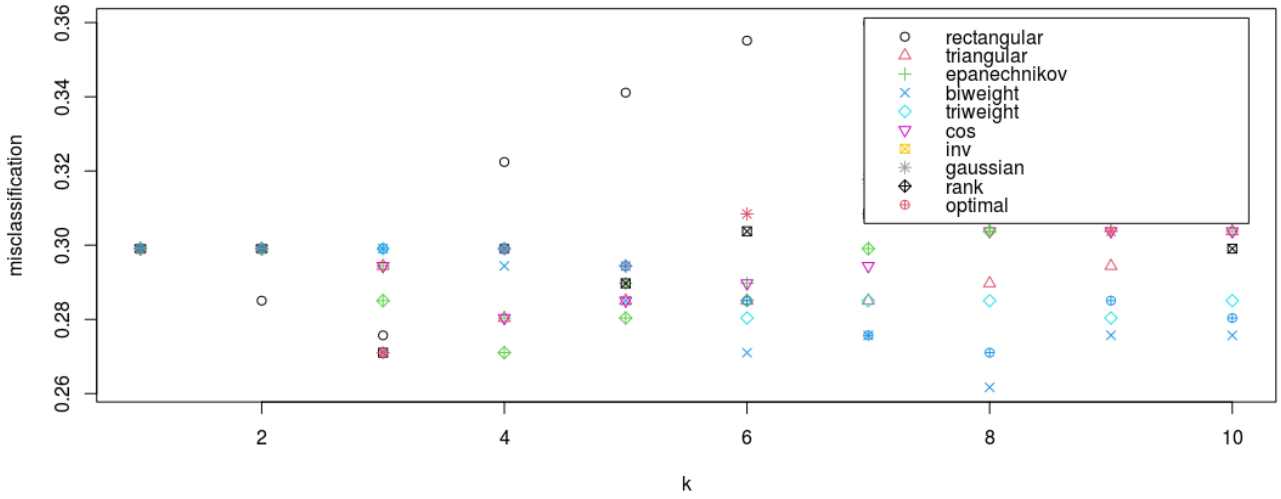

Рис. 5: Зависимость ошибки классификации от k и типа ядра

2.2.2 Исследование зависимости ошибки классификации от метрики расстояния

На рис. 6 представлен график зависимости ошибки классификации от метрики расстояния при параметрах k = 8 и kernel = biweight. Лучшим значением метрики является 1.

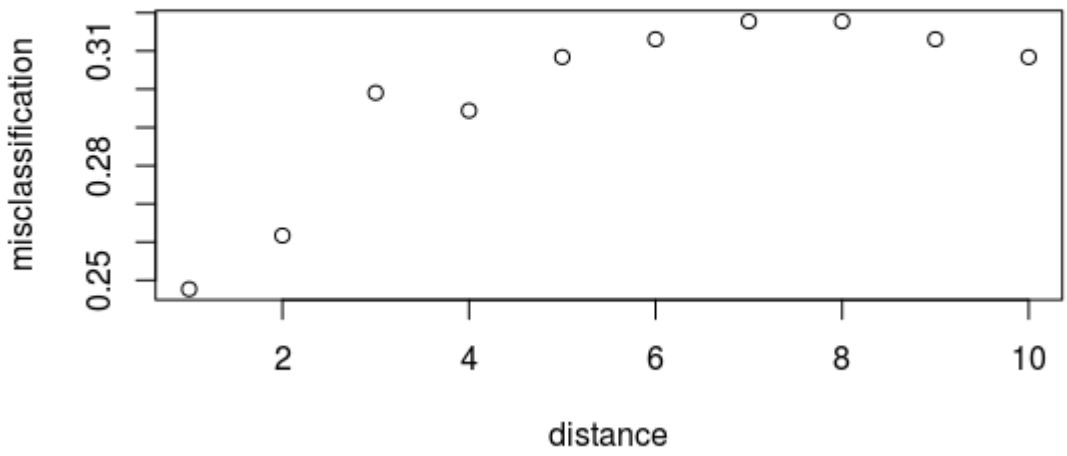

Рис. 6: Зависимость ошибки классификации от метрики расстояния

2.2.3 Определение класса примера

Экземпляр с данными характеристиками принадлежит к типу 5. При исключении одного из признаков получались следующие результаты классификации, выведенные с вероятностями определений данных классов:

- "RI"
  5
  0.9250552
- "Na"
  5
  0.5257786
- "Mg"
  5
  0.4049444
- "Al"
  5
  0.8371996
- "Si"
  5
  0.8727281
- "K"
  5
  0.7279954
- "Ca"
  2
  0.3762198
- "Ba"
  5
  0.7279954

- "Fe"
  5
  0.75672

Видно, что неправильно определяется класс при исключении признака Ca, что означает, что он сильно влияет на результат классификации. Также стоит обратить внимание на относительно низкую вероятность правильной классификации при удалении признака Mg, возможно он тоже влияет на результат, пусть и в меньшей степени.

2.3 Задание 3

Распределение данных на графике представлено на рис. 7.

Рис. 7: Распределение данных

Обучение метода KNN методом LOO дало следующие параметры классификатора:
Minimal misclassification: 0.035
Best kernel: optimal
Best k: 8
На рис. 8 представлен график зависимости величины ошибки классификации от k.

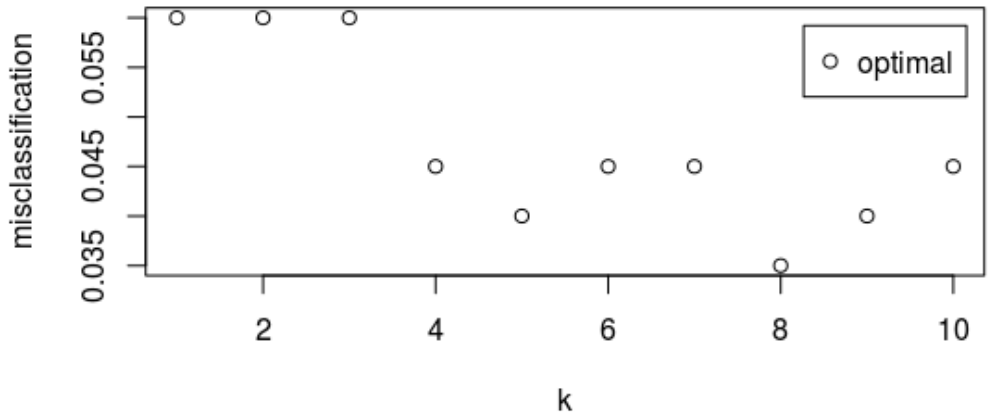

Рис. 8: Ошибка классификации при различных k

2.4 Задание 4

Данные содержат следующую информацию о пассажирах:

- Идентификатор,
- Спасся человек или нет,
- Класс билета,
- Имя,
- Пол,
- Возраст,
- Количество родственников на борту 2-го порядка,
- Количество родственников на борту 1-го порядка,
- Номер билета,
- Цена билета,
- Каюта,
- Порт посадки.

Логично предположить, что на вероятность быть спасённым не влияют такие факторы, как идентификатор, имя, номер билета, каюта и порт посадки. Тогда столбцы PassengerId, Name, Ticket, Cabin и Embarked можно удалить (в ходе дальнейших экспериментов было выявлено, что поле Fare также не улучшает точность предсказания модели). Влияние факторов класс, пол и наличие родственников на выживаемость наглядно продемонстрированы на рис. 9-12.

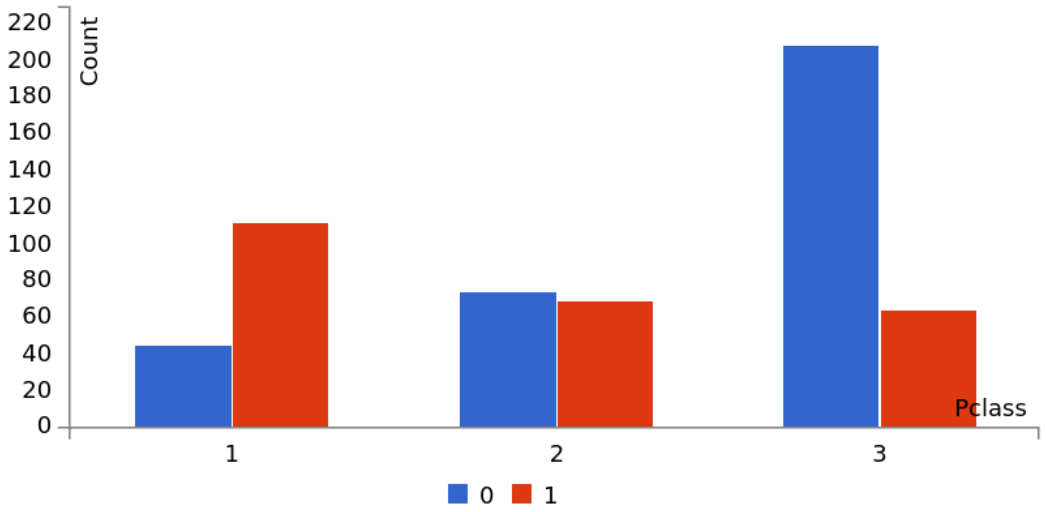

Рис. 9: График зависимости количества выживших от класса

Видно, что доля выживших преобладает только в категории пассажиров первого класса, а большая часть погибших были среди пассажиров третьего класса.

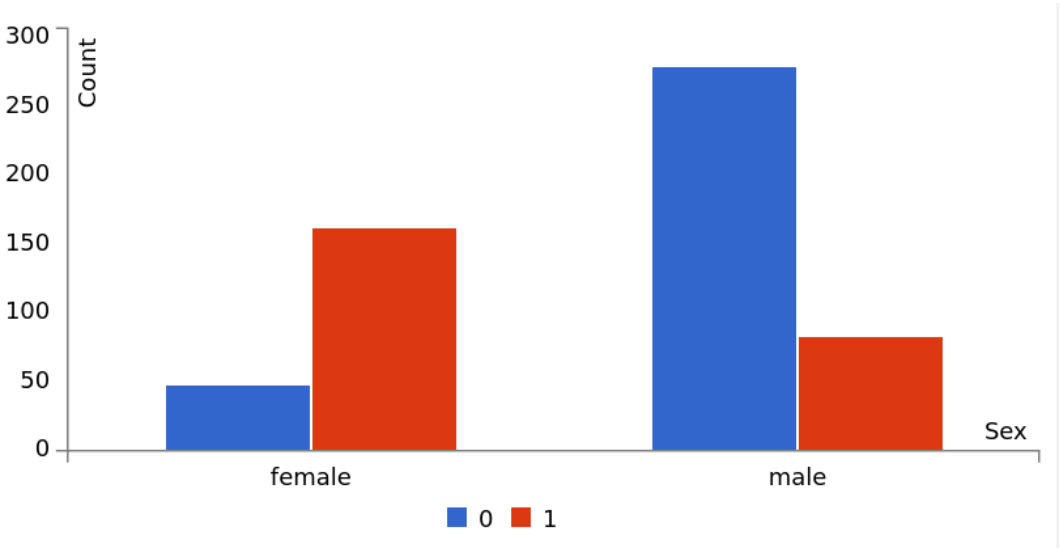

Рис. 10: График зависимости количества выживших от пола

Пол имеет большое значение для определения того, выживет пассажир или нет: большая часть женщин была спасена, а большая часть мужчин – нет.

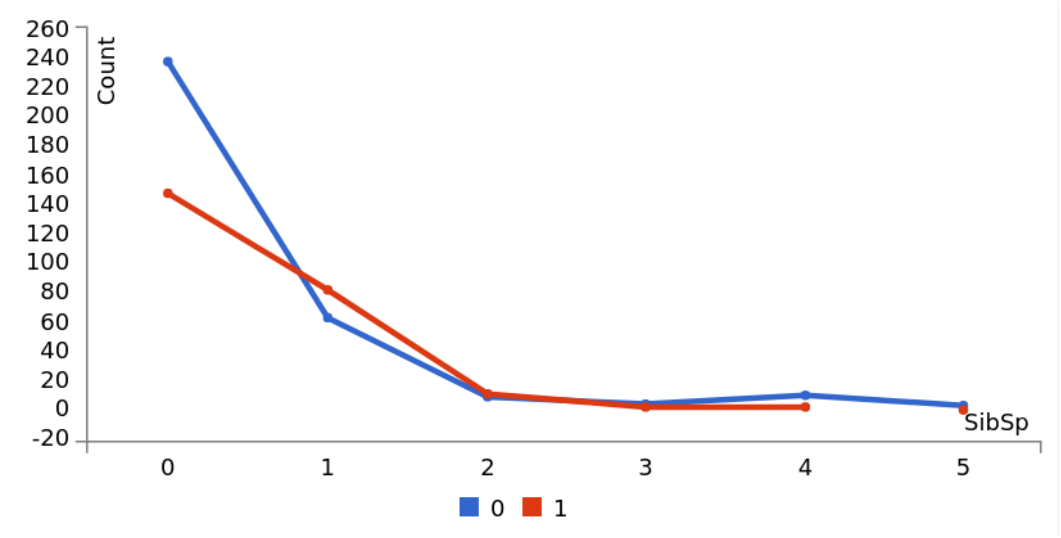

Рис. 11: График зависимости количества выживших от количества родственников 2-го порядка

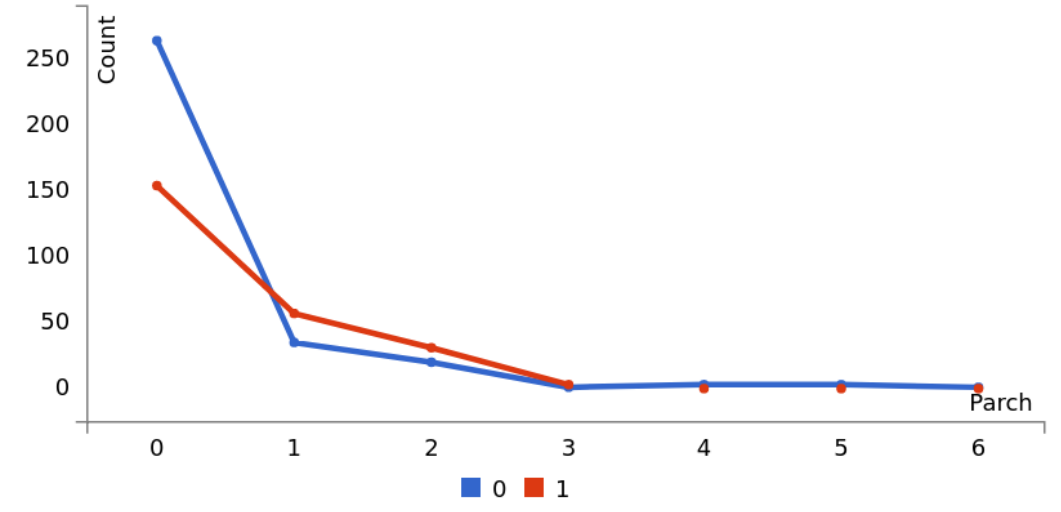

Рис. 12: График зависимости количества выживших от количества родственников 1-го порядка

Среди тех, кто имел большое количество родственников на борту, никто не выжил.

Данные содержат пропущенные значения (NA). В обучающих данных их всего 117, все они содержатся в поле Age. В данных для тестирования такие значения присутствуют в полях Age (86) и Fare (1). Записи с NA были удалены, т.к. попытка присвоить пропущенным значениям медианные значения всей выборки дала в итоге меньшую точность ответа.

В данных для тестирования отсутствовали метки Survived, поэтому была сформирована валидационная выборка для оценки точности предсказаний модели, составляющая 20% от обучающей выборки.

В итоге обучения на тестовых данных модель предсказала 195 погибших и 137 выживших пассажира. Точность на валидационной выборке составила 81%.

3 Приложение

Задание 1

```
library(kknn)
library(ggplot2)
library(dplyr)
setwd("/home/olga/MyProjects/Polikek/ML/KNN/datasets")

data <- read.delim("Tic_tac_toe.txt", sep=",", head=FALSE)
data$V10 <- as.factor(data$V10)
m <- dim(data)[1]
mean_accuracy <- c()
#size_list <- seq(0.1, 0.8, by=0.1) * m
size_list <- c(100, 200, 300, 400, 500)
for (j in 1:length(size_list)){
  sample_size <- size_list[j]
  accuracy <- c()
  for (k in 1:20){
    val <- sample(1:m, size = 400)
    data.learn <- data[val,]
    data2 <- data[-val,]
    val2 <- sample(m-400, size = sample_size)
    data.valid <- data2[val2,]
    data.kknn <- kknn(V10~., data.learn, data.valid)
    accuracy_table <- table(data.kknn$fitted.values, select(data.valid, V10)[,1])
    accuracy[k] <- sum(diag(accuracy_table)) / sum(accuracy_table)
  }
  mean_accuracy[j] <- mean(accuracy)
}


new_data <- data.frame(
  size_of_learning_dataset=size_list,
  accuracy=mean_accuracy
)
ggplot(new_data, aes(x=size_of_learning_dataset, y=accuracy)) +
  geom_line() +
  geom_point()


library(kernlab)
library(kknn)

data(spam)
m <- dim(spam)[1]
mean_accuracy <- c()
#size_list <- seq(0.1, 0.9, by=0.1) * m
size_list <- c(200, 400, 500, 600, 700, 800, 900, 1000)
for (j in 1:length(size_list)){
  sample_size <- size_list[j]
  accuracy <- c()
  for (k in 1:100){
    val <- sample(1:m, size = 2500)
    spam.learn <- spam[val,]
    data2 <- spam[-val,]
    val2 <- sample(m-2500, size = sample_size)
    spam.valid <- data2[val2,]
    spam.kknn <- kknn(type~., spam.learn, spam.valid)
    accuracy_table <- table(spam.kknn$fitted.values, select(spam.valid, type)[,1])
    accuracy[k] <- sum(diag(accuracy_table)) / sum(accuracy_table)
  }
  mean_accuracy[j] <- mean(accuracy)
}#600

library(ggplot2)
```

```
new_data <- data.frame(
  size_of_learning_dataset=size_list,
  accuracy=mean_accuracy)

ggplot(new_data, aes(x=size_of_learning_dataset, y=accuracy)) +
  geom_line() +
  geom_point()
```

Задание 2

```
library(kknn)
library(mlbench)
library(ggplot2)
data(Glass)
kernels <- c("rectangular","triangular", "epanechnikov", "biweight",
             "triweight", "cos", "inv", "gaussian", "rank", "optimal" )
result <- train.kknn(Type~., Glass, kmax=10,
          kernel=c("rectangular","triangular", "epanechnikov", "biweight",
          "triweight", "cos", "inv", "gaussian", "rank", "optimal" ))


plot(result)
misclassification <- c()
for (i in 1:10){
  result <- train.kknn(Type~., Glass, distance=i, ks=c(8), kernel="biweight")
  misclassification[i] <- result$MISCLASS
}
distance <- seq(1,10)
plot(x=distance, y=misclassification)

example <- list(RI =1.516, Na =11.7, Mg =1.01, Al =1.19, Si =72.59,
                K=0.43, Ca =11.44, Ba =0.02, Fe =0.1 )

example_kknn <- kknn(Type~., Glass, example)
print(example_kknn$fitted.values, max.levels = 0)
print(example_kknn$prob[,"5"])
for (i in 1:length(example)){
  print(names(example)[i])
  new_ex <- example[-i]
  new_Glass <- Glass[-i]
  example_kknn <- kknn(Type~., new_Glass, new_ex)
  #sprintf("class: %d", example_kknn$fitted.values, max.levels = 0)
  #sprintf("probability: %f", example_kknn$prob[,"5"])
  print(example_kknn$fitted.values, max.levels = 0)
  print(example_kknn$prob[,example_kknn$fitted.values])
  new_ex <- example
  new_Glass <- Glass
}
```

Задание 3

```
library(kknn)
library(ggplot2)
setwd("/home/olga/MyProjects/Polikek/ML/KNN/datasets")
mydata.learn <- read.delim("svmdata4.txt", sep="\t")
mydata.learn$Colors <- as.factor(mydata.learn$Colors)
mydata.test <- read.delim("svmdata4test.txt", sep="\t")
mydata.test$Colors <- as.factor(mydata.test$Colors)
mydata.train <- train.kknn(Colors~., mydata.learn, kmax=10)
plot(mydata.train)
plot(mydata.learn$X1, mydata.learn$X2, pch=21,
     bg=c("green","red") [unclass(mydata.learn$Colors)],  main="My train data")
```

Задание 4

```r
library(kknn)
library(dplyr)
library(ggplot2)
library(rpivotTable)
library(dummies)
setwd("/home/olga/MyProjects/Polikek/ML/KNN/datasets")

data.train = read.csv("Titanic_train2.csv", sep=",", quote ="\"", stringsAsFactors = TRUE)
data.train = select(data.train, Survived, Pclass, Age, Sex, SibSp, Parch)
#data.train$Fare[data.train$Fare==0] <- NA
#data.train$Age[is.na(data.train$Age)] <- median(data.train$Age, na.rm = TRUE)
data.train = na.omit(data.train)
data.train$Survived = factor(data.train$Survived)
data.train$Pclass = factor(data.train$Pclass, ordered=TRUE, levels = c(3, 2, 1))
#colSums(is.na(data.train))


data.test = read.csv("Titanic_test2.csv", sep=",", quote ="\"", stringsAsFactors = TRUE)
data.test = select(data.test, Pclass, Age, Sex, SibSp, Parch)
data.test$Pclass = factor(data.test$Pclass, ordered=TRUE, levels = c(3, 2, 1))
#data.train$Age[is.na(data.train$Age)] <- median(data.train$Age, na.rm = TRUE)
data.test = na.omit(data.test)
#data.test$Age[is.na(data.test$Age)] <- mean(data.test$Age, na.rm = TRUE)
#data.test$Fare[is.na(data.test$Fare)] <- mean(data.test$Fare, na.rm = TRUE)
s = round(seq(dim(data.train)[1]*0.2))
data.valid = data.train[s,]
data.train = data.train[-s,]

data.knn = kknn(Survived~., data.train, data.test, k=8)
table(data.knn$fitted.values)
data.check = kknn(Survived~., data.train, data.valid, k=8)
t = table(data.check$fitted.values, data.valid$Survived)
accuracy = sum(diag(t)) / sum(t)


#data.train$Sex = factor(data.train$Sex)
#data.train$SibSp = factor(data.train$SibSp)
#data.train$Parch = factor(data.train$Parch)
rpivotTable(data.train,rows="Survived", cols="Sex")
rpivotTable(data.train,rows="Survived", cols="SibSp")
rpivotTable(data.train,rows="Survived", cols="Pclass")
rpivotTable(data.train,rows="Survived", cols="Age")
rpivotTable(data.train,rows="Survived", cols="Parch")
```