САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ им. ПЕТРА ВЕЛИКОГО

Институт прикладной математики и механики

Высшая школа прикладной математики и вычислительной физики

Отчёт

по курсовой работе
по дисциплине «Системы искусственного интеллекта»

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

Преподаватель _____ Л. В. Уткин

Санкт-Петербург,
2021г.

Содержание

1 Постановка задачи

Данные для исследования в курсовом проекте взяты из базы данных «Wine recognition data». Эти данные являются результатами химического анализа вин, выращенных в одном регионе Италии, но полученных из трех разных сортов. Данные содержат следующие признаки:

1. Alcohol
2. Malic acid
3. Ash
4. Alcalinity of ash
5. Magnesium
6. Total phenols
7. Flavanoids
8. Nonflavanoid phenols
9. Proanthocyanins
10. Color intensity
11. Hue
12. OD280/OD315 of diluted wines
13. Proline

Курсовой проект заключается в разработке классификаторов для базы данных, визуализации данных, исследовании и настройки классификаторов.
В ходе работы необходимо выполнить следующие задания:

1. Разработать 3 классификатора и осуществить настройку их параметров для минимизации ошибки классификации на тестовых данных. Выполнить визуализацию данных при помощи метода t-SNE.
2. Сравнить классификаторы (по критерию вероятность ошибки классификации для тестовых данных) и обосновать выбор наилучшего из них.
3. Удалить их базы метки классов и осуществить кластеризацию данных. Построить дендограмму. Сравнить полученные результаты с реальными метками данных. Определить долю ошибочно кластеризованных данных.
4. Используя логистическую регрессию в рамках метода Лассо, определить наиболее значимые признаки, влияющие на отнесение объектов к определенному классу.

2 Разработка классификаторов

Визуализация данных, выполненная при помощи алгоритма t-SNE, представлена на рис.1

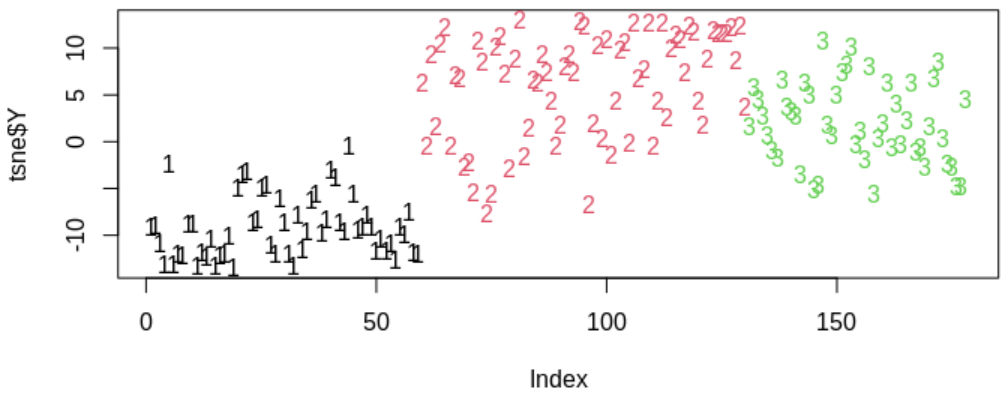

Рис. 1: Визуализация данных

Для классификации данных были обучены модели методом опороных векторов, k ближайших соседей и деревьев решений.

2.1 Метод опорных векторов

Изменение точности модели при различных параметрах, измеренное с применением скользящего контроля, представлено на рис. 2. Для модели с ядром типа «radial» и параметром C, равном 1, на тестовых данных получена точность 98%

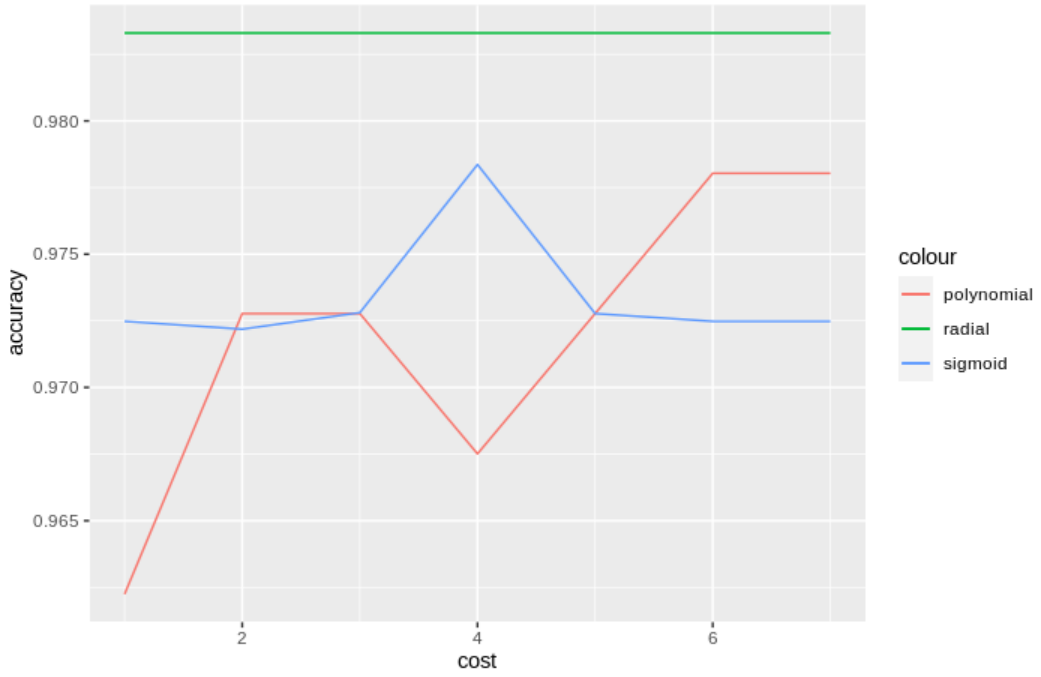

Рис. 2: График зависимости точности модели от типа ядра и параметра C

2.2 Метод k ближайших соседей

На рис. 3 и 4 представлены графики зависимости ошибки классификации от k и типа ядра при параметре distance = 1 и distance = 2 соответственно.

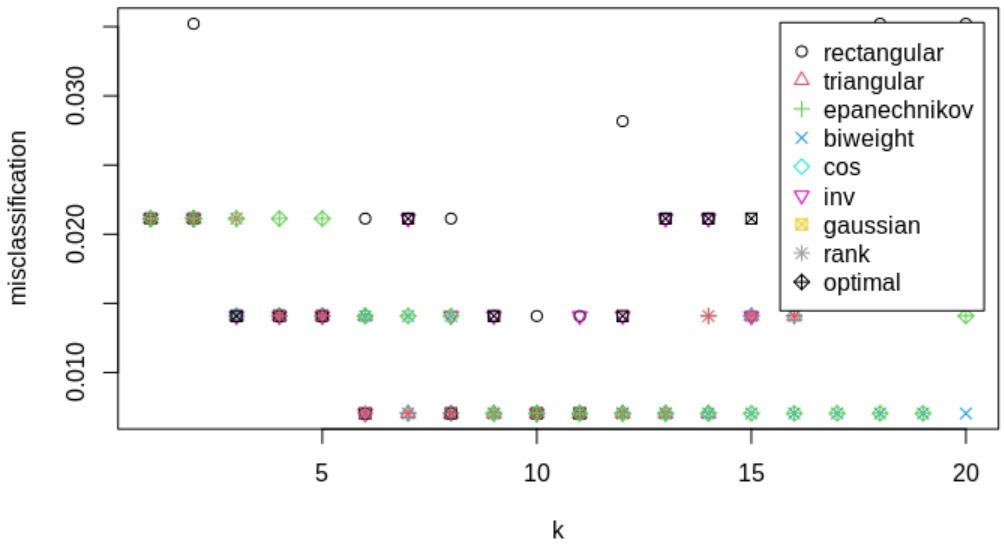

Рис. 3: distance = 1

Минимальная ошибка классификации: 0.007042254, лучшее ядро: triangular, лучшее k: 7.

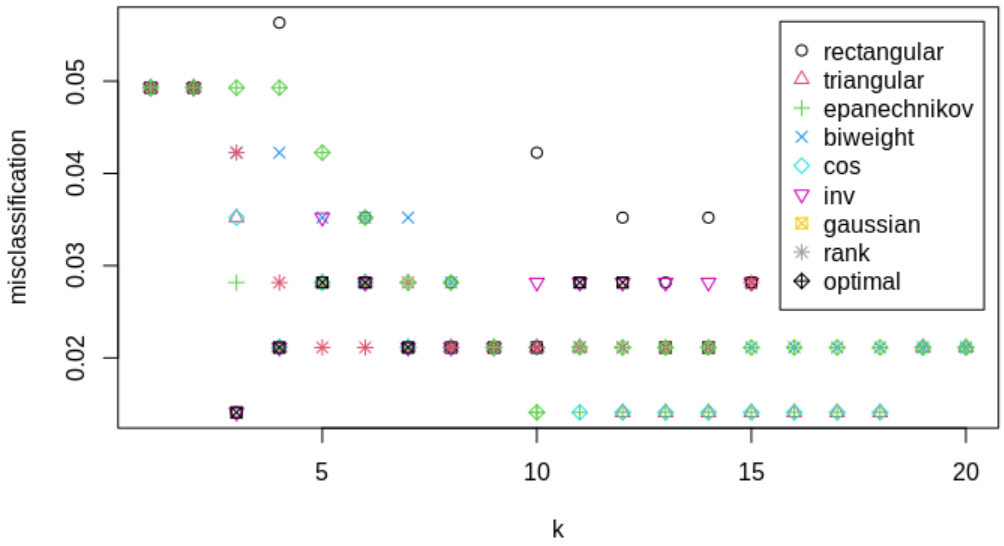

Рис. 4: distance = 2

Минимальная ошибка классификации: 0.01408451, лучшее ядро: rectangular, лучшее k: 3.

Модель с ядром triangular, k=7, distance=1 классифицировала тестовые данные с 97% точностью.

2.3 Деревья решений

На рис. 3 представлено одно из деревьев, полученных при обучении модели на обучающих данных. Средняя точность классификации, полученная при кросс-валидации на 10 разбиениях – 89%

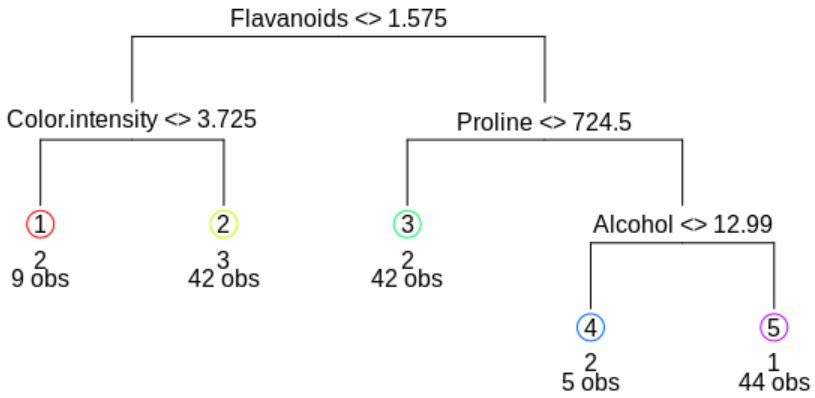

Рис. 5: Дерево решений

2.4 Результат

Были найдены оптимальные параметры для трёх моделей классификации: метод опорных векторов, метод k ближайших соседей и деревья решений. Точность классификации на тестовых данных составила:

- для метода опорных векторов – 98%;
- для k ближайших соседей – 97%
- для деревьев решений – 89%;

Таким образом, метод опорных является наиболее оптимальным.

3 Кластеризация

Кластеризация была проведена иерархическим методом. На рис. 5 представлена построенная дендограмма. Доля ошибочно кластеризованных данных — 0.522.

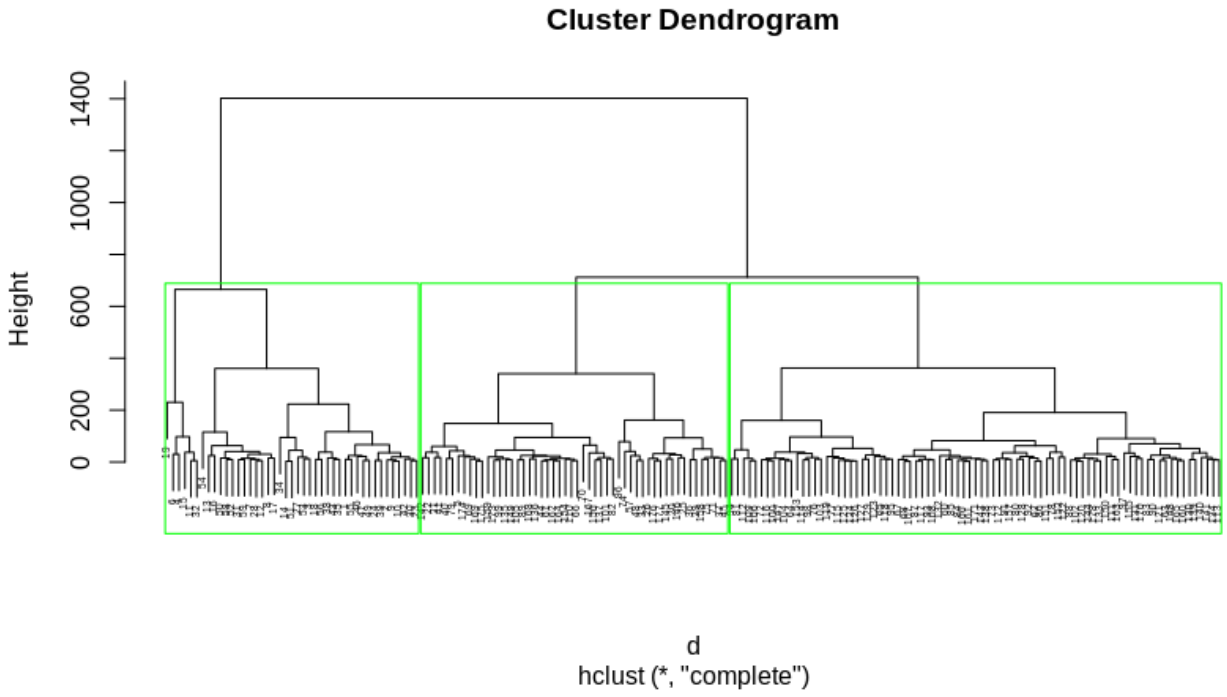

Рис. 6: Дендограмма

4 Определение наиболее значимых признаков

На рис. 6 – значения коэффициентов для всех признаков.
Наименее значимыми являются признаки Magnesium, Proline, Proanthocyanins. Наиболее значимыми – Flavanoids, Wines, Hue.

```
(Intercept)           4.245141633
Alcohol              -0.092500138
Malic.acid            0.018843655
Ash                  -0.109275403
Alcalinity.of.ash     0.035653531
Magnesium             .
Total.phenols         0.018053043
Flavanoids           -0.293313197
Nonflavanoid.phenols -0.062598400
Proanthocyanins       0.008081763
Color.intensity       0.067682052
Hue                  -0.209990663
Wines                -0.247457272
Proline              -0.000691265
```

Рис. 7: Дендограмма

Заключение

В ходе курсовой работы было проделано следующее:

1. Выполнена визуализация данных при помощи метода t-SNE.
2. Разработано 3 классификатора (метод опорных векторов, метод k ближайших соседей и деревья решений), каждый из которых довольно точно классифицировал данные из тестовой выборки. Самым точным оказался метод опорных векторов, дающий результат, равный 98%.
3. Осуществлена кластеризацию данных, с довольно высокой долей ошибочно кластеризованных данных $\approx 52\%$.
4. Методом Лассо определены наиболее и наименее значимые признаки. Наименее значимые признаки: Magnesium, Proline, Proanthocyanins. Наиболее значимые: Flavanoids, Wines, Hue.

Список используемых источников

[1] MachineLearning.ru Профессиональный информационно-аналитический ресурс, посвященный машинному обучению, распознаванию образов и интеллектуальному анализу данных
[Электронный ресурс]: http://www.machinelearning.ru/wiki/index.php (Дата обращения: 01.05.21)
[2] Документация по языку R.

Приложение

```
library(e1071)
library(ggplot2)
library(kknn)
library(Rtsne)
library(tree)
library(maptree)
library(cluster)
library(factoextra)
library(dendextend)
library(glmnet)
library(pvclust)

setwd("/home/olga/MyProjects/Polikek/ML/Kurs")
data = read.delim("wine.data", sep = ",")

d = dim(data)[1]
target = data$class
data$class = as.factor(target)
####################################
#t-sne
####################################
data = subset(data, select = -c(class))
tsne = Rtsne(data, dims=1)
plot(tsne$Y , t='n')
text(tsne$Y,labels=target,col = target)

names(target) = seq(1:d)
data$class = as.factor(target)
####################################
#KNN
####################################
set.seed(1234)
s = sample(d, size = 0.8 * d)
train = data[s,]
test = data[-s,]

kernels <- c("rectangular","triangular", "epanechnikov", "biweight", "cos", "inv", "gaussian", "rank", "optimal" )
result <- train.kknn(class~., train, kmax=20, distance = 2,
                     kernel=kernels)
plot(result)
knn = kknn(class~., train, test, k=3, kernel="optimal", distance = 2)
table_knn = table(test$class, knn$fitted.values)
table_knn
accuracy_knn = sum(diag(table_knn))/sum(table_knn)
accuracy_knn
prediction = predict(result, train[,-1])
table(prediction, as.numeric(train$class))

####################################
#SVM
####################################
s = sample(d)
data = data[s,]
m = round(d/10)
accuracy_svm = 0
kernel = "polynomial"
cost = 100
for (i in 1:10){
  if (i == 1){
    test = data[1:m,]
    train = data[(m+1):d,]
    test = na.omit(test)
    svm = svm(class~., data=train)
```

```
    prediction = predict(svm, test)
    table_svm = table(test$class, prediction)
    accuracy_svm = accuracy_svm + sum(diag(table_svm))/sum(table_svm)
  }
  else{
    if (i == 10){
      test = data[(m*10):d,]
      train = data[1:(m*10-1),]
      test = na.omit(test)
      svm = svm(class~., data=train)
      prediction = predict(svm, test)
      table_svm = table(test$class, prediction)
      accuracy_svm = accuracy_svm + sum(diag(table_svm))/sum(table_svm)
    }
    else{
      test = data[(m*i):(m*i + m),]
      train = rbind(data[1:(m*i - 1),], data[(m*i + m + 1):d,])
      test = na.omit(test)
      svm = svm(class~., data=train)
      prediction = predict(svm, test)
      table_svm = table(test$class, prediction)
      accuracy_svm = accuracy_svm + sum(diag(table_svm))/sum(table_svm)
    }
  }
}
accuracy_svm = accuracy_svm/10
accuracy_svm
####################################
#tree
####################################

accuracy_tree = 0
for (i in 1:10){
  if (i == 1){
    test = data[1:m,]
    train = data[(m+1):d,]
    test = na.omit(test)
    tree = tree(class~., train)
    prediction = predict(tree, test)
    error = 0
    for (i in 1:length(test$class)){
      if (prediction[i, test$class[i]] != 1){
        error = error + 1
      }
    }
    accuracy_tree = accuracy_tree + 1 - error/dim(test)[1]
  }
  else{
    if (i == 10){
      test = data[(m*10):d,]
      train = data[1:(m*10-1),]
      test = na.omit(test)
      tree = tree(class~., train)
      prediction = predict(tree, test)
      error = 0
      for (i in 1:length(test$class)){
        if (prediction[i, test$class[i]] != 1){
          error = error + 1
        }
      }
      accuracy_tree = accuracy_tree + 1 - error/dim(test)[1]
    }
    else{
      test = data[(m*i):(m*i + m),]
      train = rbind(data[1:(m*i - 1),], data[(m*i + m + 1):d,])
      test = na.omit(test)
```

```
      tree = tree(class~., train)
      prediction = predict(tree, test)
      error = 0
      for (i in 1:length(test$class)){
        if (prediction[i, test$class[i]] != 1){
          error = error + 1
        }
      }
      accuracy_tree = accuracy_tree + 1 - error/dim(test)[1]
    }
  }
}
accuracy_tree/10
draw.tree(tree)

tree = tree(class~., train)
prediction = predict(tree, test)
error = 0
for (i in 1:length(test$class)){
  if (prediction[i, test$class[i]] != 1){
    error = error + 1
  }
}
1 - error/dim(test)[1]
####################################
#cluster
####################################
data = read.delim("wine.data", sep = ",")
target = data$class
data = subset(data, select = -c(class))
d = dist(data)
dend = hclust(d)
plot(dend, cex=0.4)
groups = cutree(dend, k=3)
rect.hclust(dend, k = 3, border = "green")
t = table(groups, target)
1 -(sum(diag(t)) / sum(t))

####################################
#lasso
####################################
x = as.matrix(data[,-1])
y = as.matrix(data[,1])
lambda_seq <- 10^seq(2, -2, by = -.1)
cv_output <- cv.glmnet(x, y, alpha = 1,
                       lambda = lambda_seq, nfolds = 5)
plot(cv_output)
best_lam <- cv_output$lambda.min
best_lam
lasso_best <- glmnet(x, y, alpha = 1, lambda = best_lam, label=TRUE)
coef(lasso_best)
```