

Министерство образования и науки Российской Федерации

Калужский филиал
федерального государственного бюджетного образовательного
учреждения высшего образования
**«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»**
(КФ МГТУ им. Н.Э. Баумана)

И.И. Кручинин
(к.т.н. доцент)

РАЗРАБОТКА НЕЙРОННЫХ СЕТЕЙ С ПОМОЩЬЮ ЯЗЫКА R
Методические указания по выполнению домашней работы
по курсу «Введение в машинное обучение»

Калуга - 2018

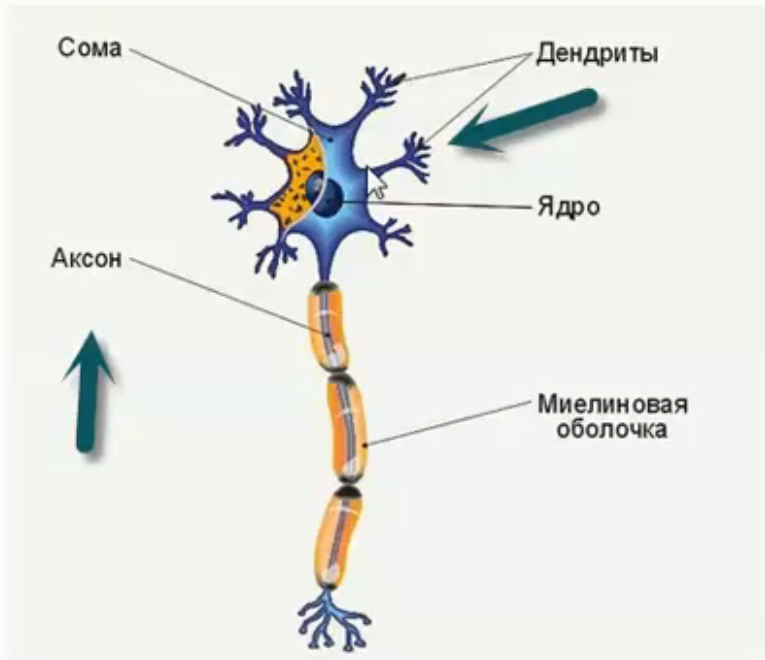
ВВЕДЕНИЕ

Настоящие методические указания составлены в соответствии с программой проведения лабораторных работ по курсу «Введение в машинное обучение» на кафедре «Программное обеспечение ЭВМ, информационные технологии и прикладная математика» факультета фундаментальных наук Калужского филиала МГТУ им. Н.Э. Баумана.

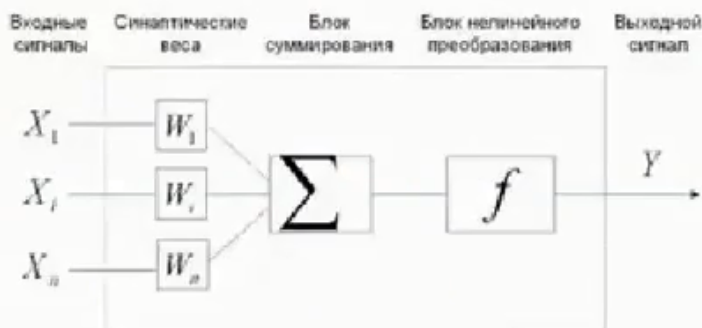
Методические указания, ориентированные на студентов 3-го курса направления подготовки 09.03.04 «Программное обеспечение ЭВМ», содержат краткое описание извлечения и представления знаний при разработке нейронных сетей, а также задание для домашней работы. Для выполнения домашней работы студенту необходимы минимальные знания по программированию на высокоуровневом языке программирования и знание базовых понятий в области нейронных сетей.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ. РАЗРАБОТКА НЕЙРОННЫХ СЕТЕЙ С ПОМОЩЬЮ ЯЗЫКА R

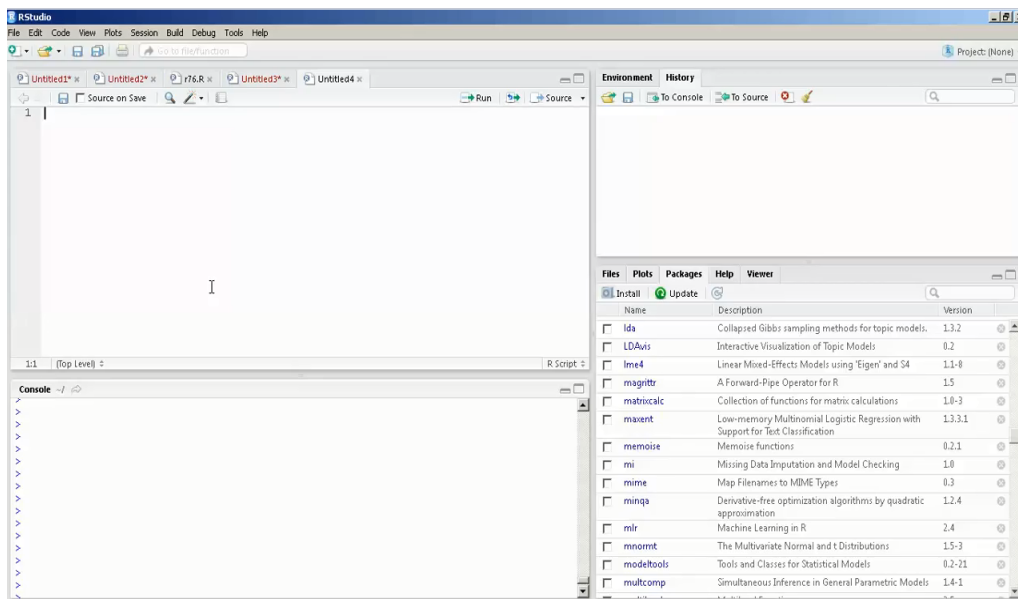
Биологический нейрон:



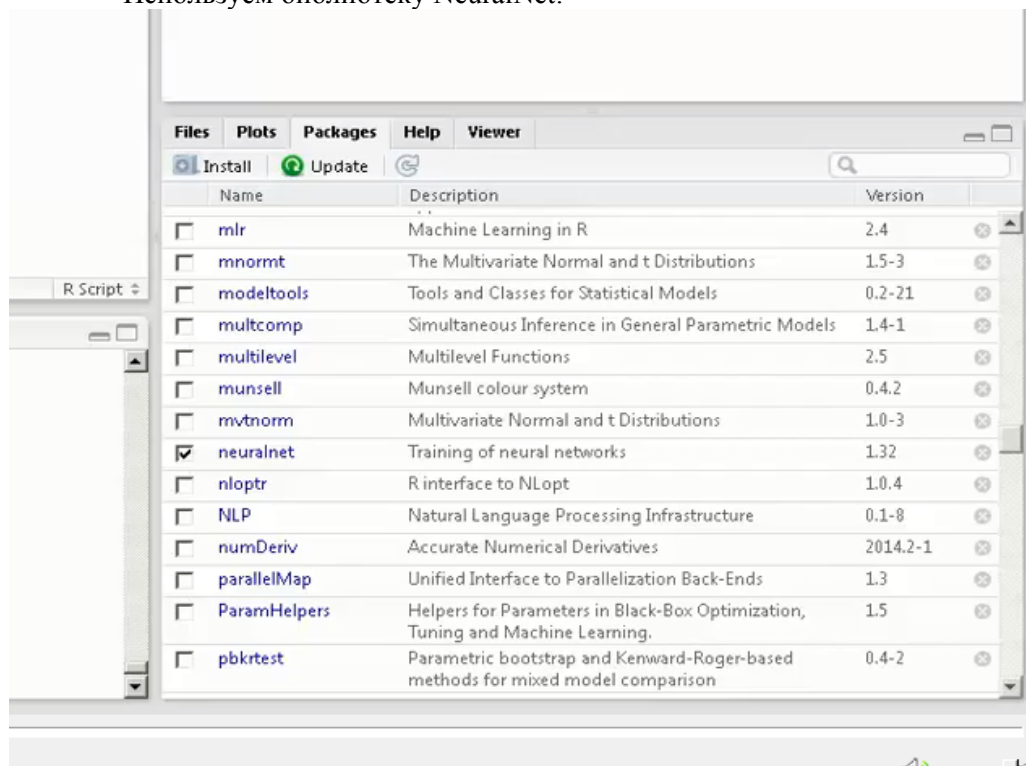
Искусственный нейрон



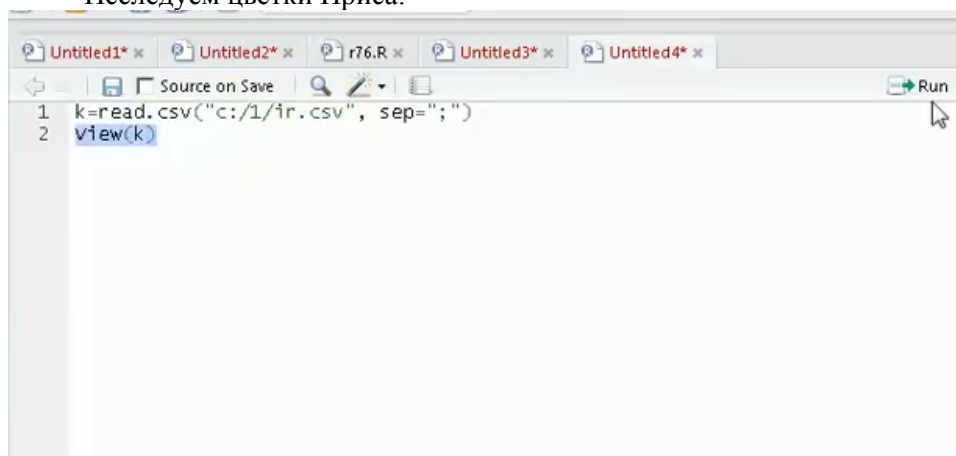
Открываем оболочку языка R:



Используем библиотеку NeuralNet:



Исследуем цветки Ириса:



RStudio

File Edit Code View Plots Session Build Debug Tools Help

Go to file/function

Untitled1* x Untitled2* x r76.R x Untitled3* x Untitled4* x k x

Filter

	SEPALLEN	SEPALWID	PETALLEN	PETALWID	IRISTYPE
1	5.0	3.3	1.4	0.2	SETOSA
2	6.4	2.8	5.6	2.2	VIRGINIC
3	6.5	2.8	4.6	1.5	VERSICOL
4	6.7	3.1	5.6	2.4	VIRGINIC
5	6.3	2.8	5.1	1.5	VIRGINIC
6	4.6	3.4	1.4	0.3	SETOSA
7	6.9	3.1	5.1	2.3	VIRGINIC
8	6.2	2.2	4.5	1.5	VERSICOL
9	5.9	3.2	4.8	1.8	VERSICOL
10	4.6	3.6	1.0	0.2	SETOSA
11	6.1	3.0	4.6	1.4	VERSICOL
12	6.0	2.7	5.1	1.6	VERSICOL

Showing 1 to 12 of 150 entries

Console

```
>
>
>
>
>
```

```

1 k=read.csv("c:/1/ir.csv", sep=";")
2 view(k)
3 v=data.frame(k,s=apply(levels(d$IRISTYPE),
4   function(l){d<-rep(0,nrow(k));
5     d[k$IRISTYPE==l]<-1;d}))

```

Environment

Name	Description
mlr	Machine Learning in R
mnormt	The Multivariate Normal and t Distribution
modeltools	Tools and Classes for Statistical Models
multcomp	Simultaneous Inference in General Param
multilevel	Multilevel Functions
munzell	Munzell colour system
mvtnorm	Multivariate Normal and t Distributions
neuralnet	Training of neural networks
nloptr	R interface to Nloptr
NLP	Natural Language Processing Infrastructure
numDeriv	Accurate Numerical Derivatives
parallelMap	Unified Interface to Parallelization Back-End
ParamHelpers	Helpers for Parameters in Black-Box Optim
ParamHelpers	Tuning and Machine Learning
pkbtest	Parametric bootstrap and Kenward-Roger methods for mixed model comparison

```

1 k=read.csv("c:/1/ir.csv", sep=";")
2 view(k)
3 v=data.frame(k,s=apply(levels(d$IRISTYPE),
4   function(l){d<-rep(0,nrow(k));
5     d[k$IRISTYPE==l]<-1;d}))

```

Так на каждый класс назначается отдельный нейрон с создаваемой сети.

```

> v=data.frame(k,s=apply(levels(d$IRISTYPE),
+   function(l){d<-rep(0,nrow(k));
+   d[k$IRISTYPE==l]<-1;d}))
> v
  SEPALLEN SEPALWID PETALLEN PETALWID IRISTYPE  S.SETOSA  S.VERSICOL  S.VIRGINIC
1      5.0      3.3      1.4      0.2   SETOSA      1      0      0
2      6.4      2.8      5.6      0.2  VIRGINIC      0      0      1
3      6.5      2.8      4.6      1.5  VERSICOL      0      1      0
4      6.7      3.1      5.6      2.4  VIRGINIC      0      0      1
5      6.3      2.8      5.1      1.5  VIRGINIC      0      0      1
6      4.6      3.4      1.4      0.3   SETOSA      1      0      0
7      6.9      3.1      5.1      2.3  VIRGINIC      0      0      1

```

Более прогрессивный метод:


```

6
7 set.seed(23)
8 size.sample=50
9 iristrain=k[sample(1:nrow(k),size.sample),]
10 nnet_iristrain=iristrain
11
12
13 nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='SETOSA')
14 nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='VERSICOL')
15 nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='VIRGINIC')
16
17 names(nnet_iristrain)[6]<-'SETOSA'
18 names(nnet_iristrain)[7]<-'VERSICOL'
19 names(nnet_iristrain)[8]<-'VIRGINIC'
20

```

Результат:

```

20
21 head(nnet_iristrain)

```

21:1 (Top Level) ↕

Console ↗ ↖

```

> nnet_iristrain=iristrain
> nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='SETOSA')
> nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='VERSICOL')
> nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='VIRGINIC')
> names(nnet_iristrain)[6]<-'SETOSA'
> names(nnet_iristrain)[7]<-'VERSICOL'
> names(nnet_iristrain)[8]<-'VIRGINIC'
> head(nnet_iristrain)

```

	SEPALLE	SEPALWID	PETALLE	PETALWID	IRISTYPE	SETOSA	VERSICOL	VIRGINIC
87	5.5	4.2	1.4	0.2	SETOSA	TRUE	FALSE	FALSE
34	6.3	2.5	5.0	1.9	VIRGINIC	FALSE	FALSE	TRUE
50	4.8	3.0	1.4	0.3	SETOSA	TRUE	FALSE	FALSE
105	5.9	3.0	4.2	1.5	VERSICOL	FALSE	TRUE	FALSE
120	5.1	2.5	3.0	1.1	VERSICOL	FALSE	TRUE	FALSE
62	6.2	2.8	4.8	1.8	VIRGINIC	FALSE	FALSE	TRUE

Иллюстрируем работу сети:

```

22
23 nn=neuralnet(SETOSA+VERSICOL+VIRGINIC~SEPALLE+SEPALWID+PETALLE+PETALWID,
24             data=nnet_iristrain,hidden = 2,err.fct="ce",linear.output = FALSE)
25
26 ?neuralnet

```

26:1 (Top Level) ↕

Console ↗ ↖

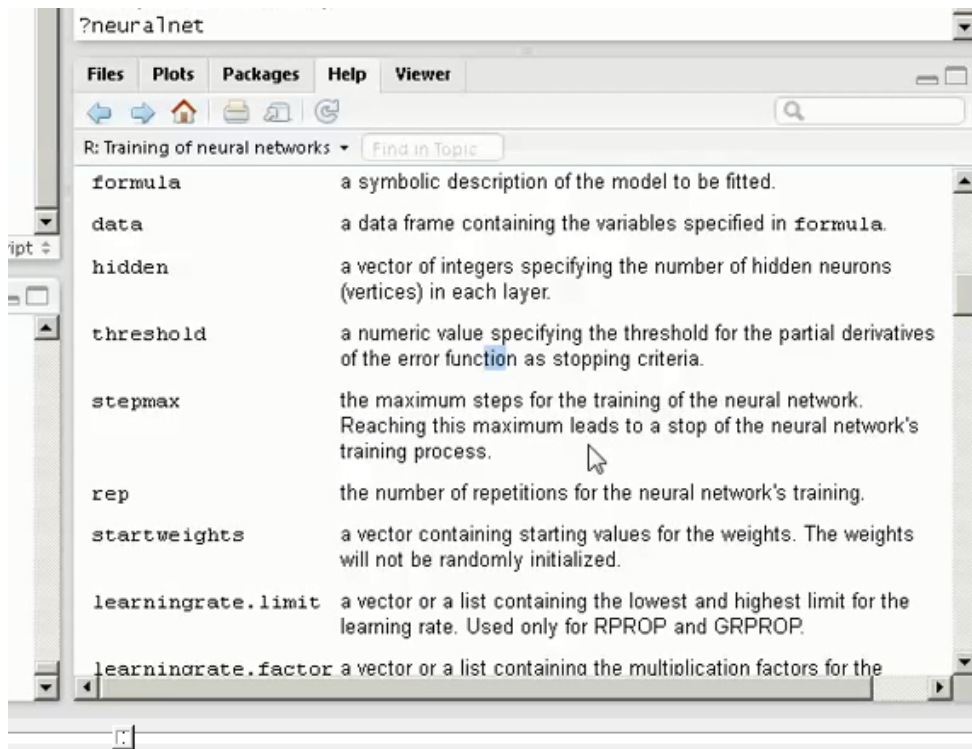
```

> nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='SETOSA')
> nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='VERSICOL')
> nnet_iristrain=cbind(nnet_iristrain,iristrain$IRISTYPE=='VIRGINIC')
> names(nnet_iristrain)[6]<-'SETOSA'
> names(nnet_iristrain)[7]<-'VERSICOL'
> names(nnet_iristrain)[8]<-'VIRGINIC'
> head(nnet_iristrain)

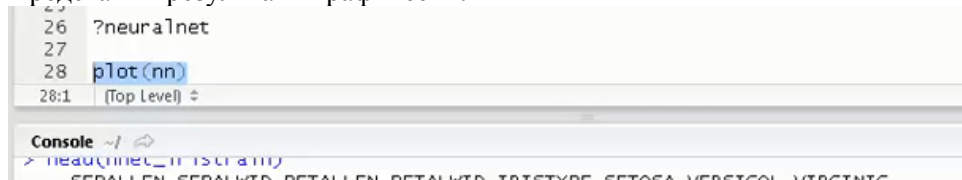
```

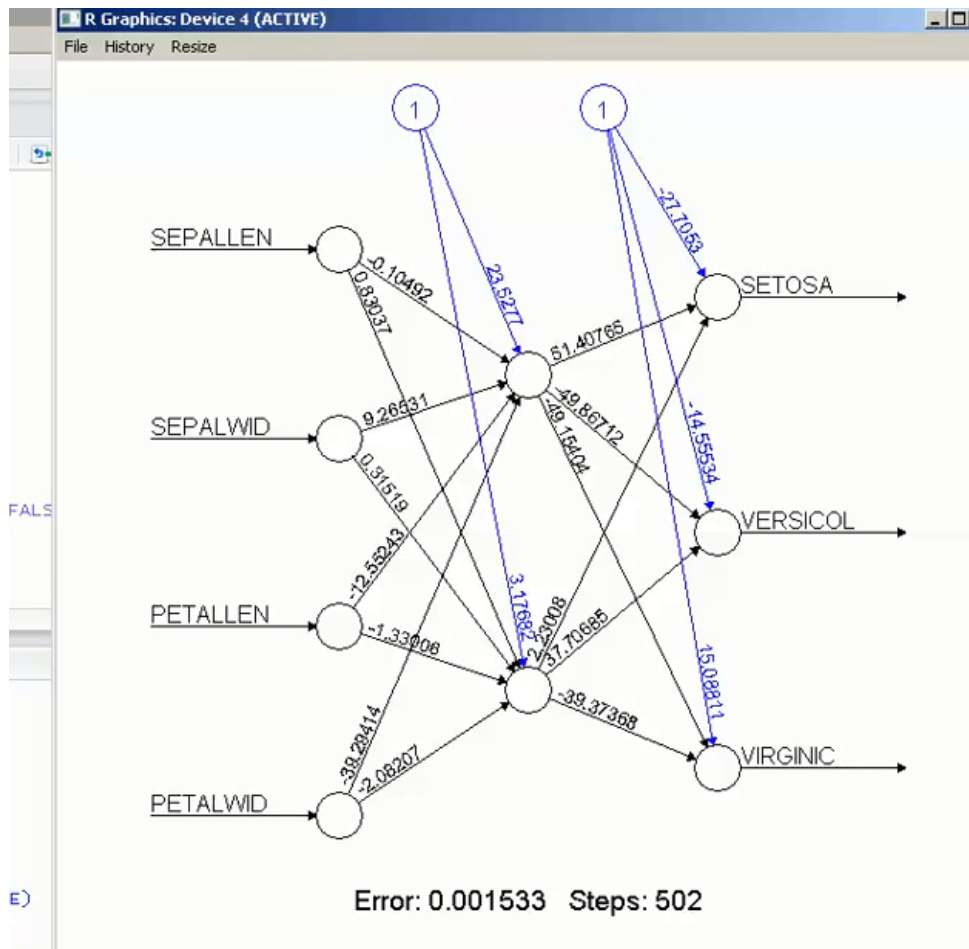
	SEPALLE	SEPALWID	PETALLE	PETALWID	IRISTYPE	SETOSA	VERSICOL	VIRGINIC
87	5.5	4.2	1.4	0.2	SETOSA	TRUE	FALSE	FALSE
34	6.3	2.5	5.0	1.9	VIRGINIC	FALSE	FALSE	TRUE
50	4.8	3.0	1.4	0.3	SETOSA	TRUE	FALSE	FALSE
105	5.9	3.0	4.2	1.5	VERSICOL	FALSE	TRUE	FALSE
120	5.1	2.5	3.0	1.1	VERSICOL	FALSE	TRUE	FALSE

Описание значение конструкции NeuralNet:



Представим результаты графически:





Далее:

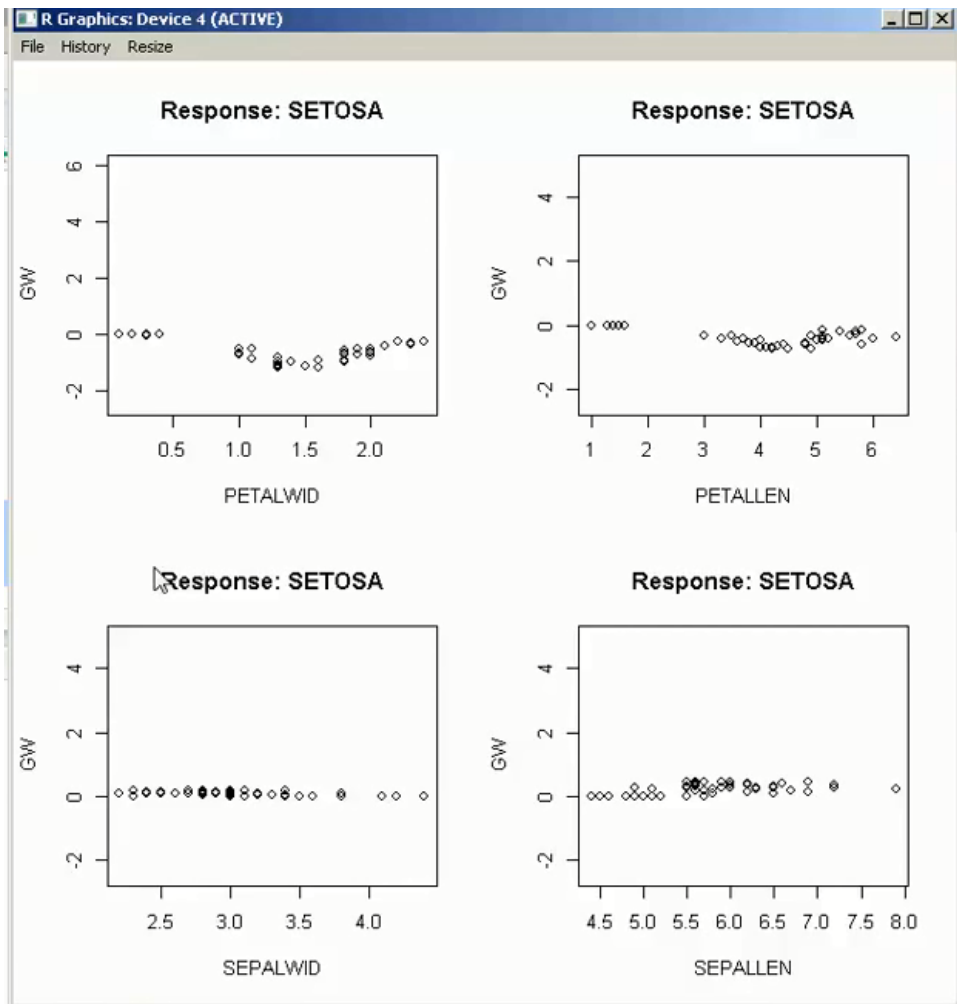
```

29
30 nn$net.result
31 nn$weights
32 nn$result.matrix
33
34
35 mypredict=compute(nn,k[-5])$net.result
36 maxidx=function(arr)
37   {return(which(arr==max(arr)))}
38
39 idx=apply(mypredict,c(1), maxidx)
40 prediction=c('SETOSA','VERSICOL','VIRGINIC')[idx]
41 table(prediction, k$IRISTYPE)
42
43 par(mfrow=c(2,2))
44 gwplot(nn,selected.covariate = "PETALWID",min=-2.5,max=6)
45 gwplot(nn,selected.covariate = "PETALLEN",min=-2.5,max=5)
46 gwplot(nn,selected.covariate = "SEPALWID",min=-2.5,max=5)
47 gwplot(nn,selected.covariate = "SEPALLEN",min=-2.5,max=5)

```

43:1 | (Top Level) ↕

Графики:



Другой пример:

```

File Edit Code View Plots Session Build Debug Tools Help
Go to file/function Addins
neuralnets.R* x
Source on Save Run
1 library(MASS)
2 library(neuralnet)
3
4 #####Setting the seed so the we get same results each time
5 ##we run neural nets
6 set.seed(123)
7
8 ###Storing the data set named "Boston" into DataFrame
9 DataFrame <- Boston
10
11 ###Help on Boston data
12 help("Boston")
13
14 ###Structure of Boston data
15 str(DataFrame)
16
17
8:21 (Top Level)

```

neuralnets.R* x DataFrame x

Filter

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
1	0.00632	18.0	2.31	0	0.5380	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.7
2	0.02731	0.0	7.07	0	0.4690	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
3	0.02729	0.0	7.07	0	0.4690	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
4	0.03237	0.0	2.18	0	0.4580	6.998	45.6	6.0622	3	222	18.7	394.63	2.94	33.7
5	0.06905	0.0	2.18	0	0.4580	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.8
6	0.02985	0.0	2.18	0	0.4580	6.430	58.7	6.0622	3	222	18.7	394.12	5.21	28.8
7	0.08829	12.5	7.87	0	0.5240	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.5
8	0.14455	12.5	7.87	0	0.5240	6.172	96.1	5.9505	5	311	15.2	396.90	19.15	27.3
9	0.21124	12.5	7.87	0	0.5240	5.631	100.0	6.0821	5	311	15.2	386.63	29.93	16.7
10	0.17004	12.5	7.87	0	0.5240	6.004	85.9	6.5921	5	311	15.2	386.71	17.10	18.5
11	0.22489	12.5	7.87	0	0.5240	6.377	94.3	6.3467	5	311	15.2	392.52	20.45	15.1
12	0.11747	12.5	7.87	0	0.5240	6.009	82.9	6.2267	5	311	15.2	396.90	13.27	18.5

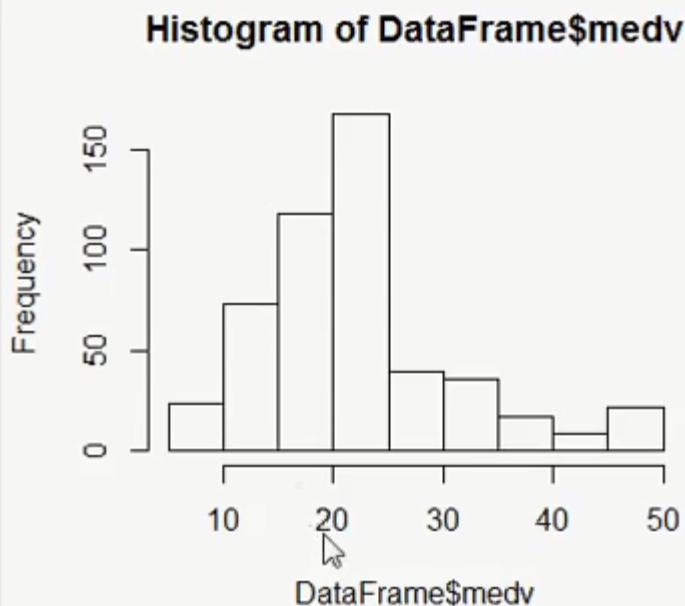
Showing 1 to 13 of 506 entries

Console C:/Users/Arpan/Desktop/ ↗
 'citation()' on how to cite R or R packages in publications.

```

16
17 ###Histogram of the medv
18 hist(DataFrame$medv)
19
20 ####Check the dimention of this data frame
21 dim(DataFrame)
22
23

```



```

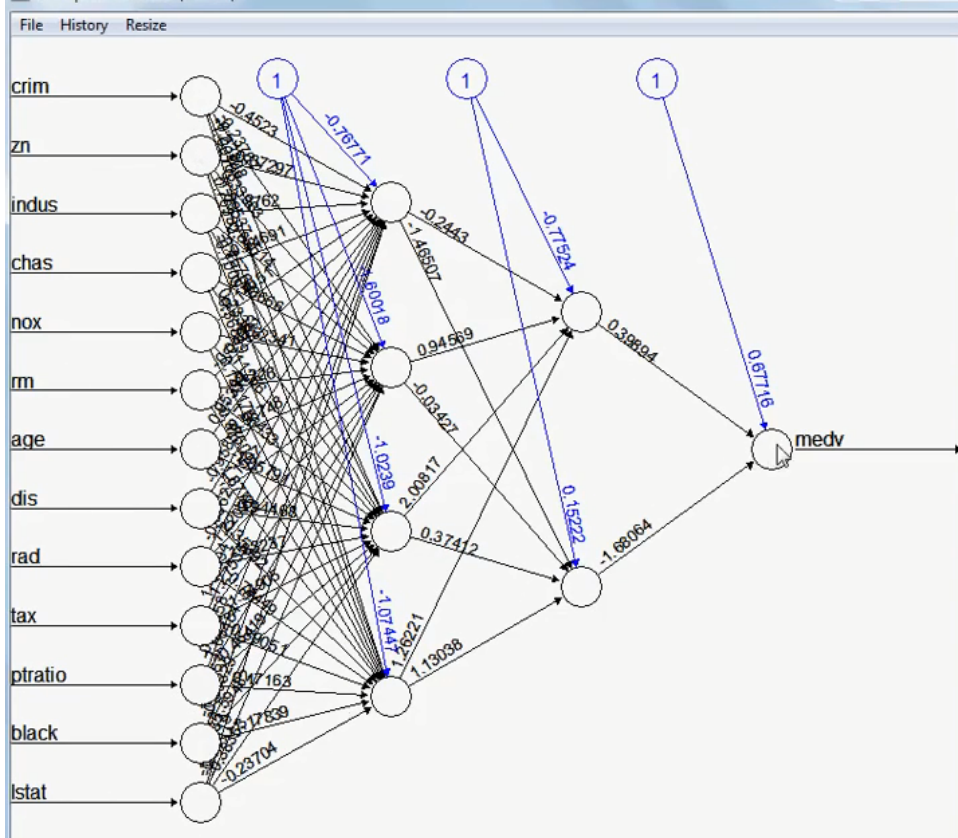
28 ##This will give min and max value for each of the variable
29 apply(DataFrame,2,range)
30
31
32
33
34
35
36
37
38
39 maxValue <- apply(DataFrame, 2, max)
40 minValue <- apply(DataFrame, 2, min)
41
42 DataFrame<-as.data.frame(scale(DataFrame,center = minValue,scale = maxV
43
44
45 ###Lets create the train and test data set
46
47 ind<-sample(1:nrow(DataFrame),400)
48 trainDF<-DataFrame[ind,]
49

```

```

61
62 allVars<-colnames(DataFrame)
63 predictorVars<-allVars[!allVars%in%"medv"]
64 predictorVars<-paste(predictorVars,collapse = "+")
65 form=as.formula(paste("medv~",predictorVars,collapse = "+"))
66
67 neuralModel<-neuralnet(formula =form,hidden = c(4,2),linear.output = T,
68 data =trainDF)
69
70
71 ###Plot the neural net
72
73 plot(neuralModel)

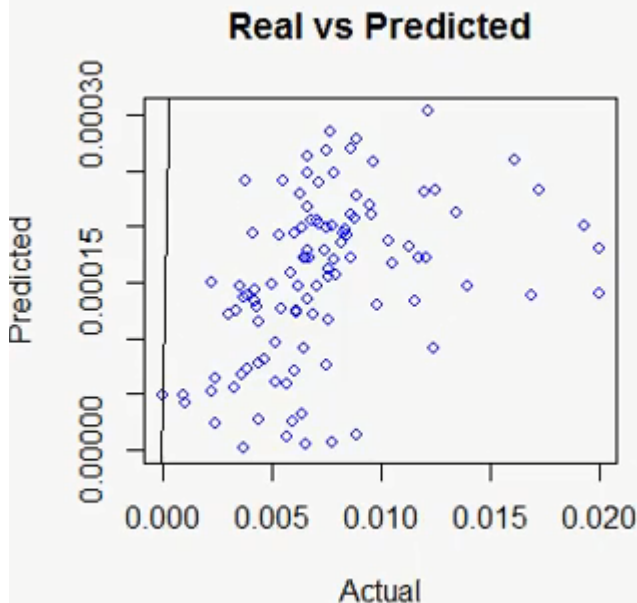
```




```

76 ###Predict for test data set
77 predictions <- compute(neuralModel,testDF[,1:13])
78 str(predictions)
79
80 predictions <- predictions$net.result*(max(testDF$medv)-min(testDF$medv))+min(testDF$medv)
81 actualValues <- (testDF$medv)*(max(testDF$medv)-min(testDF$medv))+min(testDF$medv)
82
83
84 MSE <- sum((predictions - actualValues)^2)/nrow(testDF)
85 MSE
86
87 plot(testDF$medv,predictions,col='blue',main='Real vs Predicted',pch=1,cex=0.9,type = "p",xlab=
88 abline(0,1,col="black")
89

```



В качестве альтернативной возможности классификации исходных множеств в рамках данного домашнего задания предлагается использовать функцию языка статистического моделирования R: MLP (многослойный персептрон). Данный алгоритм также основан на постулатах теории нейронных сетей. Используется обучение с учителем прямого и обратного распространения.

Параметр функции MLP – learnFunc (алгоритм обучения):

- 1) Std_Backpropagation,

- 2) BackpropBatch,
- 3) BackpropChunk,
- 4) BackpropMomentum,
- 5) BackpropWeightDecay,
- 6) Rprop,
- 7) Quickprop,
- 8) SCG (scaled conjugate gradient)

```
mlp(x, ...) # присутствует в пакете rsnns
# S3 method for default
mlp(x, y, size = c(5), maxit = 100,
  initFunc = "Randomize_Weights", initFuncParams = c(-0.3, 0.3),
  learnFunc = "Std_Backpropagation", learnFuncParams = c(0.2, 0),
  updateFunc = "Topological_Order", updateFuncParams = c(0),
  hiddenActFunc = "Act_Logistic", shufflePatterns = TRUE, linOut = FALSE,
  outputActFunc = if (linOut) "Act_Identity" else "Act_Logistic",
  inputsTest = NULL, targetsTest = NULL, pruneFunc = NULL,
  pruneFuncParams = NULL, ...)
```

В качестве примера можно рассмотреть нейросетевую классификацию цветков ириса из стандартных наборов данных языка R.

```
# NOT RUN {
demo(iris)
# }
# NOT RUN {
demo(laser)
# }
# NOT RUN {
demo(encoderSnnsCLib)
# }
# NOT RUN {

data(iris)

#shuffle the vector
iris <- iris[sample(1:nrow(iris),length(1:nrow(iris))),1:ncol(iris)]

irisValues <- iris[,1:4]
irisTargets <- decodeClassLabels(iris[,5])
#irisTargets <- decodeClassLabels(iris[,5], valTrue=0.9, valFalse=0.1)
```

```

iris <- splitForTrainingAndTest(iris$Values, iris$Targets, ratio=0.15)
iris <- normTrainingAndTestSet(iris)

model <- mlp(iris$inputsTrain, iris$targetsTrain, size=5,
  learnFuncParams=c(0.1),
  maxit=50, inputsTest=iris$inputsTest, targetsTest=iris$targetsTest)

summary(model)
model
weightMatrix(model)
extractNetInfo(model)

par(mfrow=c(2,2))
plotIterativeError(model)

predictions <- predict(model,iris$inputsTest)

plotRegressionError(predictions[,2], iris$targetsTest[,2])

confusionMatrix(iris$targetsTrain,fitted.values(model))
confusionMatrix(iris$targetsTest,predictions)

plotROC(fitted.values(model)[,2], iris$targetsTrain[,2])
plotROC(predictions[,2], iris$targetsTest[,2])

#confusion matrix with 402040-method
confusionMatrix(iris$targetsTrain, encodeClassLabels(fitted.values(model),
  method="402040", l=0.4, h=0.6))

# }

```

Параметры нейронных сетей можно оптимизировать средствами пакета **caret**

Функция `train()` из пакета `caret` перекрестной проверкой оценивает оптимальные значения числа скрытых нейронов `size` и параметр “ослабления весов” `decay`, который осуществляет регуляризацию точности подстройки коэффициентов (при `decay = 0` стремление к точности может перерасти в эффект переусложнения модели).

```

library(nnet)
library(caret)

```

```
load(file = "data/abalone.RData")
set.seed(123)
train.aba <- train(Возраст ~ ., data = abalone[, c(3:8, 10)],
  method = "nnet", trace = FALSE, linout = 1,
  tuneGrid = expand.grid(.decay = c(0, 0.05, 0.2), .size = 4:9),
  trControl = trainControl(method = "cv"))
train.aba
```

Была выполнена 10-кратная перекрестная проверка 18 нейросетевых моделей с числом нейронов в скрытом слое от 4 до 9 и разных значениях “ослабления”. При найденных значениях `size = 7` и `decay = 0`, приводящих к максимальной точности Ассигасу, построим далее модель с помощью функции `nnet()`. Для визуализации сети применим функцию из скрипта `nnet_plot_update.r`, которая имеет ряд полезных опций (можно скачать с <https://www.r-bloggers.com>).

```
source("scripts/nnet_plot_update.r")
nn.aba <- nnet(Возраст ~ ., data = abalone[, c(3:8, 10)],
  decay = 0, size = 7, niter = 200, trace = FALSE)
plot.nnet(nn.aba)
summary(nn.aba)
```

Рассмотрим теперь, насколько плезна полученная нейросетевая модель при выполнении предсказаний:

```
pred <- predict(nn.aba, abalone[, 3:8], type = "class")
nn.table <- table(abalone[, 10], pred)
confusionMatrix(nn.table)
```

Точность предсказания возрастных категорий несколько возросла по сравнению с кумулятивным логитом (0.598 против 0.557), даже несмотря на то, что ковариату пол не использовали.

Функция `rsaNNet()` является своеобразной оберткой для совместного выполнения предобработки данных, анализа главных компонент, и запуска функции `nnet()` для обучения сети на полученных главных компонентах.

Выполним обучение сети для предсказания возрастной категории морских ушек с использованием параметра `thresh = 0.975` и числа нейронов в скрытом слое `size = 7`:

```
pcaNNet.Fit <- pcaNNet(abalone[, 3:8], abalone[,10],  
  size = 7, thresh = 0.975,  
  linout = TRUE, trace = FALSE)
```

При значении `thresh = 0.975` на вход сети подается 4 главных компоненты, отвечающих этому условию. Для тестируемых данных такое же преобразование (основанное на факторных нагрузках для обучающего множества) применяется к новым значениям предикторов.

```
pred <- predict(pcaNNet.Fit, abalone[, 3:8], type = "class")  
(table(Факт = abalone$Возраст, Прогноз = pred))
```

```
Acc <- mean(pred == abalone$Возраст)  
paste("Точность=", round(100*Acc, 2), "%", sep = "")
```

Функция `avNNet()` осуществляет обучение заданного множества моделей нейронной сети на одном и том же наборе данных. Для моделей классификации функция `avNNet()` оценивает среднее значение вероятностей классов на основе частных прогнозов каждой из моделей созданного ансамбля и далее производит заключительное предсказание класса.

Для рассматриваемого примера сформируем 10 экземпляров моделей ИНС с использованием бэггинга:

```
avNNet.Fit <- avNNet(Возраст ~ ., data = abalone[, c(3:8, 10)],  
  size = 7, repeats = 10, linout = TRUE,  
  trace = FALSE, bag = TRUE)  
pred <- predict(avNNet.Fit, abalone[, 3:8], type = "class")  
Acc <- mean(pred == abalone$Возраст)  
paste("Точность=", round(100*Acc, 2), "%", sep = "")
```

РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ ДОМАШНЕЙ РАБОТЫ

Необходимо на языке R реализовать нейронную сеть для предсказания размера пенсии в зависимости средней зарплаты. Для реализации использую пакет `neuralnet`. В качестве среды - программа Rgui для windows. Вначале у нас есть два ряда данных - средняя зарплата по городу за последние 10 лет и средняя пенсия за последние 10 лет. Эти данные используются для обучения нейронной сети:

```
#средняя зарплата за каждый год
traininginput <- c(0.225, 690, 2313, 2931, 4061, 4937, 5809, 7096, 8803,
10095, 12229, 13572)
#средняя пенсия за каждый год
trainingoutput <- c(0.118, 274, 949, 1270, 1668, 2001, 2434, 3028, 3393,
4519, 5594, 7610)
данные для обучения:
trainingdata <- cbind(traininginput,trainingoutput)
colnames(trainingdata) <- c("Input","Output")
И обучение сети
net.pension <- neuralnet(Output~Input,trainingdata, hidden=10,
threshold=0.01)
print(net.pension)
```

При вводе средней зарплаты на будущий год нейронная сеть должна выдавать прогноз средней пенсии на следующий год.

```
#Отправляем на вход среднюю зарплату на будущий год
testdata <- c(15851)
net.results <- compute(net.pension, testdata)
ls(net.results)
#Lets see the results
print(net.results)
```

Применим код:

```
library(neuralnet)
```

```

# 1. creating the initial data, plotting
data <- data.frame (
  input = c(0.225, 690, 2313, 2931, 4061, 4937, 5809,
7096, 8803, 10095, 12229, 13572),
  output = c(0.118, 274, 949, 1270, 1668, 2001, 2434,
3028, 3393, 4519, 5594, 7610)
)
plot(data$output ~ data$input, main="Distribution of the
pension relative to the salary", xlab="Salary",
ylab="Pension")

# 2. normalizing the data, plotting
min.input <- min(data$input)
min.output <- min(data$output)
range.input <- diff(range(data$input))
range.output <- diff(range(data$output))
data.norm <- data.frame (
  input = (data$input - min.input) / range.input,
  output = (data$output - min.output) / range.output
)
plot(data.norm$output ~ data.norm$input, main="Distribution
of the pension relative to the salary (normalized)",
xlab="Salary", ylab="Pension")

# 3. neural network
net <- neuralnet(output ~ input, data.norm)

# 4. test the output

```

```
testdata <- seq(0, 25000, by=500)
testdata.norm <- (testdata - min.input) / range.input
result <- round(compute(net, testdata.norm)$net.result *
range.output + min.output)
plot(testdata, result, main="Predicred outcome",
xlab="Salary", ylab="Pension")
```

ЗАДАЧИ И ПОРЯДОК ВЫПОЛНЕНИЯ ДОМАШНЕЙ РАБОТЫ

Разработать нейронную модель предметной области, указанной в варианте задания – проанализировать полученные знания.

Реализовать разработанную нейронную модель на высокоуровневом языке программирования - R. Созданная нейронная сеть должна предоставить ответы согласно выбранному варианту задания.

ВАРИАНТЫ ЗАДАНИЙ ДЛЯ ДОМАШНЕГО ПРОЕКТИРОВАНИЯ:

Задания могут выполняться группой студентов (численностью до 2-ух человек)

1. Необходимо на языке R реализовать нейронную сеть для предсказания размера пенсии военнослужащего в зависимости средней зарплаты в звании капитана, майора и подполковника и мест прохождения службы (Дальний Восток, Таджикистан, Красноярский край). Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать Std_Backpropagation). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

2. Необходимо на языке R реализовать нейронную сеть для предсказания размера квартплаты в регионах: Хабаровский край, Калининградская область, Московская область в зависимости от изменений цен на нефть в течение 2015- 2017 годов. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать BackpropBatch). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

3. Разработать нейросетевой классификатор для распознавания видов водостойких растений (гидатофиты, гидрофиты, гигрофиты). Разработать набор признаков характеризующих каждый из трех заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать BackpropChunk). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

4. Разработать нейросетевой классификатор для распознавания видов засухоустойчивых растений (мезофиты, ксерофиты, склерофиты). Разработать набор признаков характеризующих каждый из трех заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать BackpropMomentum). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

5. Разработать нейросетевой классификатор для распознавания видов растений Ксерофитов (суккуленты и склерофиты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать BackpropWeightDecay). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

6. Разработать нейросетевой классификатор для распознавания видов растений Склерофитов (зуксерофиты и стипаксерофиты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать Rprop). Оптимизировать параметры нейронных сетей с

помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

7. Разработать нейросетевой классификатор для распознавания видов растений по способу регулирования воды внутри организма (пойкилогидридные и гомогидридные). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать Quickprop). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

8. Разработать нейросетевой классификатор для распознавания видов растений по потребности во влажной среде (эвригигробионты и стеногигробионты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать SCG). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

9. Разработать нейросетевой классификатор для распознавания видов растений по потребности поглощать свет (гелиофиты и сциофиты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать Std_Backpropagation). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

10. Разработать нейросетевой классификатор для распознавания видов теневыносливых растений (лиственница, ясень и липа). Разработать набор признаков характеризующих каждый из трех заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать BackpropBatch). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

11. Разработать нейросетевой классификатор для распознавания видов жаростойких растений (мегатермофиты, мезотермофиты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc

(алгоритм обучения) выбрать BackpropChunk). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

12. Разработать нейросетевой классификатор для распознавания видов холодостойких растений (микротермофиты, гекистотермофиты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать BackpropMomentum). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

13. Разработать нейросетевой классификатор для распознавания видов растений по отношению к механическому составу почвы(литифиты, хасмофиты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать BackpropWeightDecay). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

14. Разработать нейросетевой классификатор для распознавания видов растений по отношению к механическому составу песчаных почв (пелитофиты, псаммофиты). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать Rprop). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

15. Разработать нейросетевой классификатор для распознавания видов растений по отношению к содержанию питательных веществ в почве (эутрофные, олиготрофные). Разработать набор признаков характеризующих каждый из двух заданных классов растений. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать Quickprop). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

16. Необходимо на языке R реализовать нейронную сеть для предсказания уровня инфляции (виды инфляции – ползучая, галопирующая, гиперинфляция) на основании выбранных признаков: индекс потребительских цен, индекс цен производителей, дефлятор ВВП, паритет покупательной способности, индекс Пааше. Использовать функции NeuralNet и MLP (параметр learnFunc (алгоритм обучения) выбрать SCG). Оптимизировать параметры нейронных сетей с помощью пакета caret и сравнит полученные результаты. Полный список с исходными данными взять из приложения к домашнему заданию.

ФОРМА ОТЧЕТА ПО ДОМАШНЕЙ РАБОТЕ

На выполнение домашней работы отводится 8 академических часов: 7 часов на выполнение и сдачу домашней работы и 1 час на подготовку отчета.

Номер варианта студенту выдается преподавателем.

Отчет на защиту предоставляется в печатном виде.

Структура отчета (на отдельном листе(-ах)): титульный лист, формулировка задания (вариант), описание формы представления знаний, этапы обработки данных системой, результаты выполнения работы выводы.

ОСНОВНАЯ ЛИТЕРАТУРА

1. Jesse, Russell Искусственная нейронная сеть / Jesse Russell. - М.: VSD, 2012. - 0 с.
2. Jesse, Russell Нейрон / Jesse Russell. - М.: VSD, 2012. - 0 с.
3. Барский, А. Б. Логические нейронные сети / А.Б. Барский. - М.: Интернет-университет информационных технологий, Бином. Лаборатория знаний, 2007. - 352 с.
4. Барский, А.Б. Логические нейронные сети / А.Б. Барский. - М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2013. - 0 с.
5. Бунаков, В. Е. Нейронная физика. Учебное пособие: моногр. / В.Е. Бунаков, Л.В. Краснов. - М.: Издательство Санкт-Петербургского университета, 2015. - 200 с.
6. Головинский, П. А. Математические модели. Теоретическая физика и анализ сложных систем. Книга 2. От нелинейных колебаний до искусственных нейронов и сложных систем / П.А. Головинский. - М.: Либроком, 2012. - 234 с.
7. Денис, Хусаинов Механизмы ритмической активности нейронов виноградной улитки / Хусаинов Денис , Иван Коренюк und Татьяна Гамма. - М.: LAP Lambert Academic Publishing, 2012. - 108 с.
8. Как устроено тело человека. Выпуск 25. Нейроны. - М.: DeAgostini, 2007. - 30 с.
9. Катехоламинергические нейроны. - М.: Наука, 1979. - 296 с.
10. Круглов, В.В. Искусственные нейронные сети. Теория и практика: моногр. / В.В. Круглов, В.В. Борисов. - М.: Горячая линия - Телеком; Издание 2-е, стер., 2002. - 382 с.
11. Мандельштам, Ю. Е. Нейрон и мышца насекомого: моногр. / Ю.Е. Мандельштам. - М.: Наука, 1983. - 168 с.
12. Нейронные сети. Statistica Neural Networks. Методология и технологии современного анализа данных. - М.: Горячая линия - Телеком, 2008. - 392 с.
13. Парвин, Манучер Из серого. Концерт для нейронов и синапсов / Манучер Парвин. - М.: Страта, 2015. - 408 с.
14. Позин, Н. В. Моделирование нейронных структур / Н.В. Позин. - М.: Наука, 1970. - 264 с.
15. Рассел, Джесси Вербализация нейронных сетей / Джесси Рассел. - М.: VSD, 2013. - 0 с.

16. Рассел, Джесси Искусственный нейрон / Джесси Рассел. - М.: VSD, 2013. - 0 с.
17. Татузов, А. Л. Нейронные сети в задачах радиолокации / А.Л. Татузов. - М.: Радиотехника, 2009. - 432 с.
18. Толкачев, С. Нейронное программирование диалоговых систем / С. Толкачев. - Москва: **РГТУ**, 2016. - 192 с.
19. Шибзухов, З. М. Конструктивные методы обучения сигма-пи нейронных сетей / З.М. Шибзухов. - М.: Наука, 2006. - 160 с.
20. Юревич, Артур Нейронные сети в экономике / Артур Юревич. - М.: LAP Lambert Academic Publishing, 2014. - 80 с.

Электронные ресурсы:

- 1) <http://alexanderdyakonov.narod.ru/upR.pdf>
- 2) <http://cran.gis-lab.info/web/packages/nnet/nnet.pdf>
- 3) <http://www.faqs.org/faqs/ai-faq/neural-nets/>
- 4) <http://r-analytics.blogspot.com/>