Использование нейронных сетей в R для прогнозирования цен акций ГАЗПРОМА

Нейросетевые алгоритмы являются одними из самых эффективных инструментов машинного обучения (Machine learning). Они реализованы в рамках экосистемы R. Воспользуемся интегрированными в R алгоритмами нейросетевого моделирования для решения задачи прогнозирования цен акций Газпрома. В качестве исходных данных будем использовать временной ряд (ВР) ежедневных цен закрытия акций Газпрома в период с 2006 года по 2 декабря 2020 года. Данный ВР мы не будем приводить к "хорошему" виду,т.е мы не заполняем пропушенные выходные и праздники (при желании эта проблема легко решается программными средствами R)

Программная реализация алгоритмов nnet в R

Нейросетевые методы реализованы нами в среде R version 3.6.3 64-bit с использованием пакетов $gWidgets$ и $RGtk2$, которые в более поздних версиях не поддерживаются. Были созданы два модуля (скрипта): nnet.R, Prognoz_net.R

Сript $nnet.R$

Данный скрипт формирует окно для ввода необходимых исходных данных и параметров нейросетевых алгоритмов (см. рисунок 1) и запускает скрипт обучения сети и формирования прогноза.

```
library(gWidgets)
library(RGtk2)
library(lubridate)
options(guiToolkit="RGtk2")

g_win4 <-gwindow(title = "Прогнозирование ВР nnet()",
visible = FALSE,do.buttons=FALSE)
vhod_nnet <- ggroup(container = g_win4,horizontal = F)
Vhod_nnet01 <- gframe("Входные параметры ВР",
container = vhod_nnet,horizontal = F)
Lnnet01 <-glabel(" Номер вкладки Excel с данными",
container =Vhod_nnet01)
par_nnet01 <- gedit("2", container =Vhod_nnet01)
Lnnet02 <-glabel("Начальная дата для обучения nnet ггггмм  ",
container = Vhod_nnet01)
```

```
par_nnet02 <-gcalendar(text = "2017-01-01",
 format = "%Y%m%d", handler=NULL,
action=NULL, container =Vhod_nnet01 )
Lnnet03 <-glabel("Конечная дата для обучения nnet ггггммдд  ",
container = Vhod_nnet01)

par_nnet03 <-gcalendar(text = "2020-11-25",
 format = "%Y%m%d", handler=NULL,
action=NULL, container =Vhod_nnet01 )
Lnnet04 <-glabel(" Интервал прогнозирования раб. дн.",
container = Vhod_nnet01)
par_nnet04 <- gedit("30", container =Vhod_nnet01)
gbutton("Ввод параметров", border=TRUE,
container = Vhod_nnet01,
handler = function(h,...){Val <- svalue(par_nnet01)
Val <- as.numeric(Val)
print(Val)
N1 <<-Val
Val2 <- svalue(par_nnet02)
Val2_god=substr(Val2,1,4)
Val2_mon=substr(Val2,6,7)
Val2=paste(Val2_god,Val2_mon,sep="")
print(Val2)
date_0 <<-Val2
Val3 <- svalue(par_nnet03)
Val3_god=substr(Val3,1,4)
Val3_mon=substr(Val3,6,7)
Val3_d=substr(Val3,9,10)
Val3=paste(Val3_god,Val3_mon,Val3_d,sep="")
print(Val3)
date_3 <<-Val3
Val4 <- svalue(par_nnet04)
Val4 <- as.numeric(Val4)
print(Val4)
l_prognoz1 <<-Val4
gmessage("Параметры введены")})
gimage("nnet3.gif",dirname="D:/Machine learning/Nnet",
container = Vhod_nnet01)
```

```
vhod2_nnet <- ggroup(container = g_win4,horizontal = T)
nnet_model <- gframe("Задание параметров нейронной сети"
,container = vhod2_nnet,horizontal = F)
glabel("Число нейронов на скрытых уровнях",
container = nnet_model)
glabel("Первый уровень",container = nnet_model)
par_nnet05 <- gedit("5", container =nnet_model)
glabel("Второй уровень",container = nnet_model)
par_nnet06 <- gedit("3", container =nnet_model)
glabel("Третий уровень",container = nnet_model)
par_nnet07 <- gedit("2", container =nnet_model)
glabel("Предельное значение частных производных",
container = nnet_model)
par_nnet08 <- gedit("0.01", container =nnet_model)
glabel("Максимальное значение итераций",
container = nnet_model)
par_nnet09 <- gedit("600000", container =nnet_model)
glabel("Число повторений",container = nnet_model)
par_nnet10 <- gedit("1", container =nnet_model)
glabel("Алгоритм",container = nnet_model)
nnet_alg <-gcombobox(c('rprop+','backprop','rprop-','sag','slr'),
editable=TRUE, container=nnet_model)
glabel("Функция активации",container = nnet_model)
nnet_fakt <-gcombobox(c('logistic','tanh'),
editable=TRUE, container=nnet_model)
gbutton("Ввод параметров нейронной сети",
 border=TRUE, container =nnet_model,
handler = function(h,...){yn1 <- svalue(par_nnet05)
yn1 <- as.numeric(yn1)
print(yn1)
YN1 <<-yn1
yn2 <- svalue(par_nnet06)
yn2 <- as.numeric(yn2)
print(yn2)
YN2 <<-yn2

yn3 <- svalue(par_nnet07)
```

```
yn3 <- as.numeric(yn3)
print(yn3)
YN3 <<-yn3

pr_df <- svalue(par_nnet08)
pr_df <- as.numeric(pr_df)
print(pr_df)
PR_DF <<-pr_df

max_iter <- svalue(par_nnet09)
max_iter <- as.numeric(max_iter)
print(max_iter)
MAX_ITER <<-max_iter

val_rep <- svalue(par_nnet10)
val_rep <- as.numeric(val_rep)
print(val_rep)
VAL_REP <<-val_rep

alg <- svalue(nnet_alg)
print(alg)
ALG <<-alg

aktivf <- svalue(nnet_fakt)
print(aktivf)
AKTIVF <<-aktivf
gmessage("Параметры введены")})

gbutton("Запуск нейронной сети", border=TRUE,
 container =nnet_model,
handler = function(h,...){source("Prognoz_net.R")
adf = gdf(Dat_prognoz_nnet, container =gwindow())
})
visible(g_win4) <- TRUE
```

В окне приведенном на рисунке 1 набраны параметры, которые использовались для обучения нейронной сети и получения прогноза. В модуле не предусмотрена защита от неправильного ввода. При первом об-

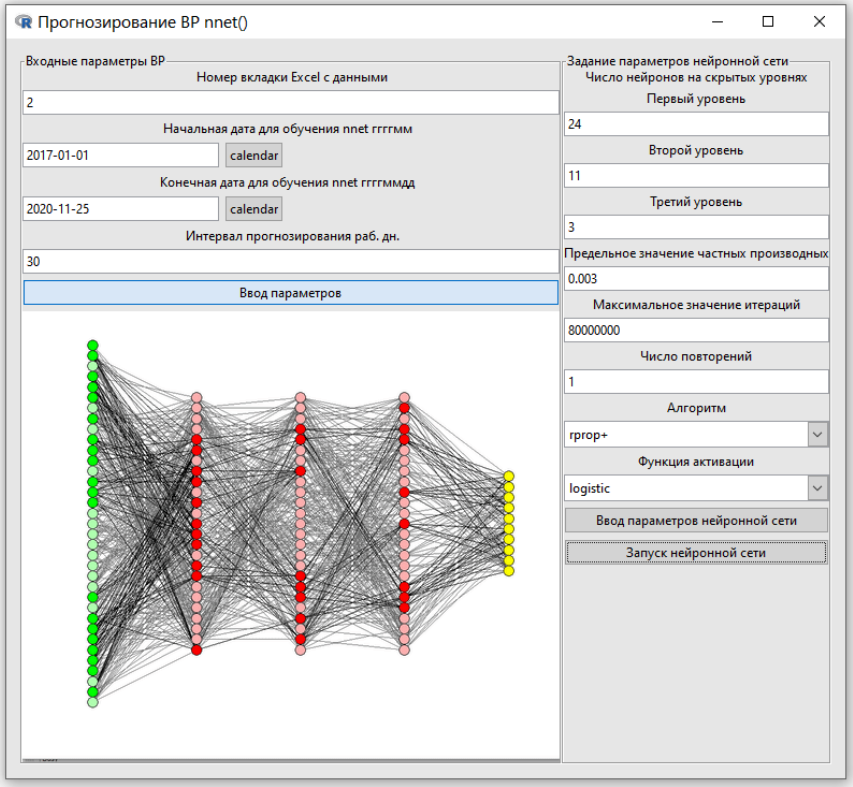

Рис. 1: Окно задания параметров nnet

ращении к окну ввода необходимо сначало нажать кнопку **Ввод параметров**, чтобы ввести параметр (номер вкладки книги Excel) для ввода BP и задать периоды обучения сети и прогноза и только после этого можно пользоваться остальными кнопками. Вдальнейшем можно изменять параметры в любой последовательности перед нажитем кнопки **Запуск нейронной сети**

Срипт *Prognoz_net.R*

Это основной модуль,кторые выполняет следующие функции: вводит из Excel исходный временной ряд, формирует в соотвествии с выбранными параметрами обучающую выборку, готовит нормированные значения входных и выходных данных для сети, реализует процесс обучения сети, обрабатывает полученные в результате обучения данные, формирует на основе обученной сети прогноз и создает соотвествующие графики.

Скрипт имеет следующий вид:

```
library(ggplot2)
library(lubridate)
library(scales)
library(ggthemes)
library(nnet)
library(neuralnet)

# номер листа с данными
# ЗАЗРУЗКА ВРЕМЕННЫХ РЯДОВ GAZP С ЛИСТА N1  EXCEL======
z <-read.xlsx("AkzGASP.xlsx",sheet = N1)

z <<-z
m1=length(z$n)
z0=z[1:m1,1]
z1=z[1:m1,2]
z2=z[1:m1,3]

# начальная дата обучения модели
dat0=as.character(z$DATE)
dat0=substr(dat0,1,6)
ind_dat0 = which(dat0==date_0)
ttn0=ind_dat0[1]


# конечная точка обучения модели
dat3=as.character(z$DATE)
ind_dat3 = which(dat3==date_3)
ttn3=ind_dat3

# ЗАДАНИЕ ГРАНИЦ ОБУЧЕНИЯ, ТЕСТИРОВАНИЯ И ПРОГНОЗИРОВАНИЯ
n0=ttn0 # начало обучения
n3=ttn3 # конец обучения
n4=n3+1 # начало теста
n5=m1    # конец ряда и теста
n1=n5+1 # начало прогноза
```

```
n2=n1+l_prognoz1-1 # конец прогноза

print(c(n0,n3,n4,n5,n1,n2))
# ------------------------------------------------------

# ФОРМИРОВАНИЕ ИСХОДНОЙ ТАБЛИЦЫ ДАННЫХ

t_prog=c(n1:n2)
input_prog =c(z0,t_prog)
n12=n2-n1+1
output_prog=c(z2,rep(1.0,each = n12))

zz =data.frame(input=input_prog,output=log(output_prog))
attach(zz)
# ------------------------------------------------------------
min.input <- min(input)
range.input <- diff(range(input))
input_norm =(input - min.input) / range.input
zz_norm <-data.frame(input=input_norm,output)
zz_norm_train <-zz_norm[n0:n3,]
zz_norm_test  <-zz_norm[n4:n5,]
zz_norm_prognoz <-zz_norm[n1:n2,]

net <- neuralnet(output~input,data=zz_norm_train,
hidden=c(YN1,YN2,YN3), startweights=NULL,threshold=PR_DF,
stepmax =MAX_ITER,rep =VAL_REP,
algorithm=ALG,linear.output = TRUE,act.fct=AKTIVF)
str(net)
plot(net)
result0=net$net.result[1]

resul_test <-compute(net,zz_norm_test)
result1 <- resul_test$net.result
cor(result1,zz_norm_test$output)
result1=as.vector(result1)

resul_prognoz <-compute(net,zz_norm_prognoz)
result2 <- resul_prognoz$net.result
```

```
result2=as.vector(result2)

train.vec= unlist(result0, recursive=TRUE, use.names = TRUE)
inp_ob=c(n0:n3)
zz0=data.frame(inp_ob,result00=exp(train.vec))
inp=zz$input[1:n5]
out=zz$output[1:n5]
zz1=data.frame(inp,out=exp(out))
inp_tr=c(n4:n5)
zz2=data.frame(inp_tr,result1=exp(result1))
inp_pr=c(n1:n2)
zz3=data.frame(inp_pr,result2=exp(result2))
prognoz=exp(result2)
prognoz
detach(zz)

# ====================================
t1=z[length(z$DATE),2]
t1=as.character(t1)
# формирование дат прогноза
t1 <-as.Date(t1,"%Y% m%d")
t_prog=c(t1,t1,t1,t1,t1)
for(i in 1:l_ prognoz1){ t_ prog[i]=t1 + ddays(i)}
t_prog

tt1 <- as.character(z$DATE)
tt1 <-as.Date(tt1,"%Y%m%d")
zzz1 <-data.frame(tt1,out=exp(out))
tt3 <-t_prog
zzz3 <-data.frame(tt3,result2=exp(result2))

tt0 <-z1[inp_ob]
tt0 <- as.character(tt0)
tt0 <-as.Date(tt0,"%Y% m% d")
zzz0 <-data.frame(tt0,result00=exp(train.vec))

g2 <-ggplot(data=zzz1, aes(x=tt1,y=out))+
geom_line(colour ="gray55",size=0.6)
```

```
g2 <-g2 + geom_line(data=zzz3,aes(x=tt3,y=result2),
colour="red",linetype=1,size=1.5)
g2 <-g2 + geom_line(data=zzz0,aes(x=tt0,y=result00),
colour="green",linetype=1,size=1.0)
g2 <-g2 +scale_x_date(breaks = date_breaks("1 years"),
labels = date_format("%Y"))
g2 <-g2 +labs(title ="Уровни цен акций GASP",
x="Временные периоды", y="цена акций, руб")
#g2 <-g2 +theme_economist() + scale_colour_economist()
x11()
print(g2)

ttt0 ="2020-01-01"
zzzz1 <-subset (zzz1,tt1 >=ttt0)
zzzz0 <-subset (zzz0,tt0 >=ttt0)

g3 <-ggplot(data=zzzz1, aes(x=tt1,y=out))+
geom_line(colour ="gray55",size=0.6)
g3 <-g3 + geom_line(data=zzz3,aes(x=tt3,y=result2),
colour="red",linetype=1,size=1.5)
g3 <-g3 + geom_line(data=zzzz0,aes(x=tt0,y=result00),
colour="green",linetype=1,size=1.0)
g3 <-g3 +scale_x_date(breaks = date_breaks("1 years"),
labels = date_format("%Y"))
g3 <-g3 +labs(title ="Уровни цен акций GASP",
x="Временные периоды", y="цена акций, руб")
x11()
print(g3)

prognoz=prognoz[1:l_prognoz1]
# ФОРМИРОВАНИЕ ФРЕЙМА ПРОГНОЗНЫХ ДАННЫХ
prognoz_nnet <-round(prognoz,digits=2)
Dat_prognoz_nnet <-data.frame(t=t_prog,ynnet=prognoz_nnet)
print(Dat_prognoz_nnet)
Dat_prognoz_nnet <<-Dat_prognoz_nnet

# ВЫВОД ДАННЫХ ПРОГНОЗА
# write.xlsx(Dat_prognoz_nnet, "Dat_prognoz_nnet.xlsx")
```

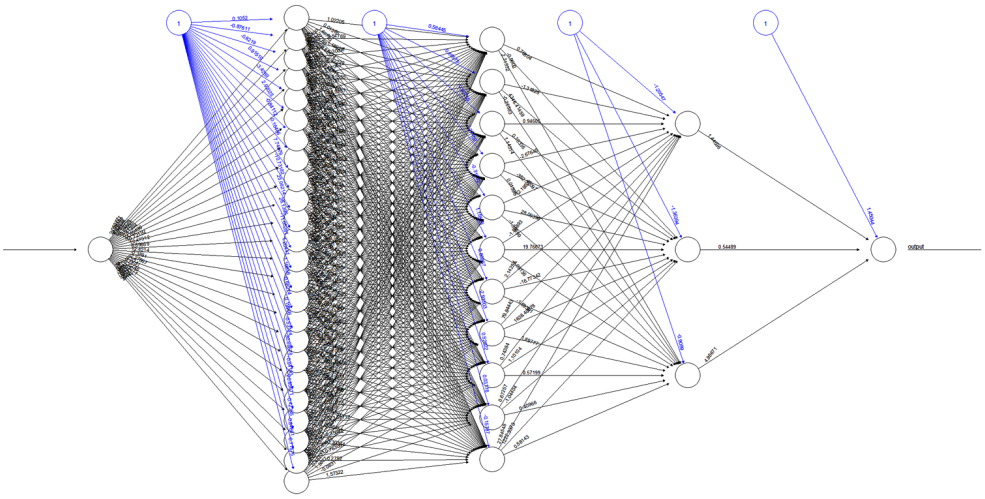

Рис. 2: Структура и веса обученной сети

Результаты моделирования ВР

Мы зафиксировали трехуровневую структуру нейронной сети , число нейроннов на каждом уровне, вид алгоритма и ряд ругих праметров можно подбирать. Процесс настройки сети под фактические данные это творческий и довольно трудоемкий процесс, который конечно частично можно автоматизировать, но все же здесь много неформализуемого. В результате проведенной работы мы получили следующий вид сети (рисунок 2).

Результаты обучения и прогноз представлены на рисунке 3, а укрупненно результы в пределах только 2020 года показаны на рисунке 4. Каждый полученный вариант обучения можно запомнить, сохраняя рабочее пространство R, и затем использовать для воспроиведения полученных результатов.

Кроме обучающей выборки строится тестовая выборка, в нашем случаю это всего несколько точек, суммарный квадрат отклонений в этих точках может использоваться, как формальный критерий качества обучения, иногда используют для этого коэффициент корреляции между полученными на тестовой последовательности значениями и фактическими данными.

Рис. 3: Результаты обучения сети и прогноз 2017-2020 гг.

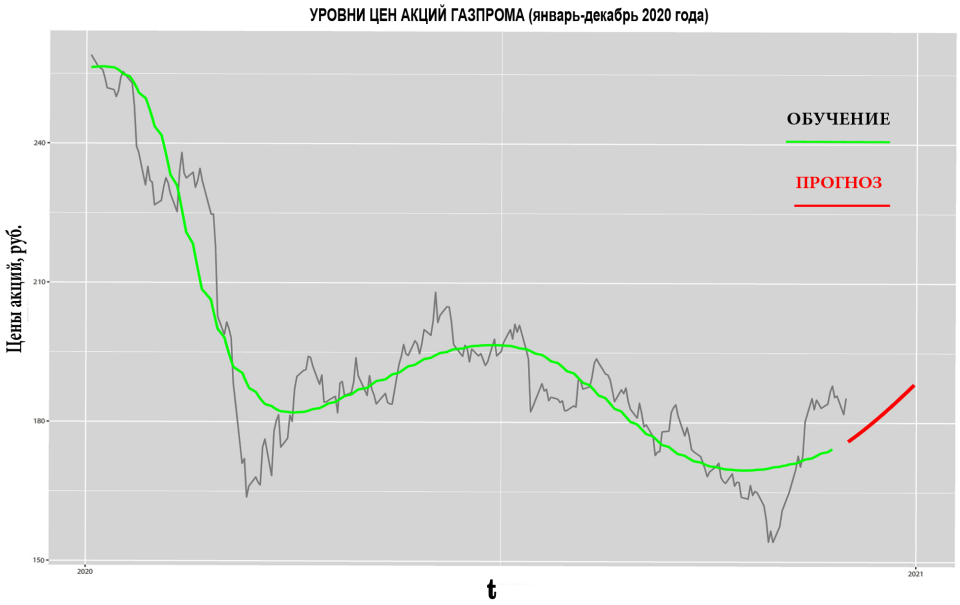

Рис. 4: Результаты обучения сети и прогноз 2000 г.

ВЕРСТКА LATEX 05.12.2020