### Исследование временных рядов с помощью среды R. Несбалансированные данные

По определению, данные являются несбалансированными, если количества наблюдений, принадлежащих различным классам, существенно отличаются. Например, в случае данных “Credit” количество отрицательных клиентов составляет около 6,5% (Рис. 1.). Как следствие, алгоритм обучения может полностью игнорировать малочисленный класс, который очень важен.

Рис. 1. Соотношение «плохих» и «хороших» клиентов в тренировочном наборе

Количество «плохих» и «хороших» клиентов в тренировочном наборе можно определить, например, так:

|  |
| --- |
| length(A\_train[A\_train[,2]==1,2]) # плохие  length(A\_train[A\_train[,2]==0,2]) # хорошие |

### Однородное ансамблирование. Метод случайных сбалансированных подмножеств. Паспорта скользящего контроля.

Однородное ансамблирование – усреднение элементарных классификаторов (так называемых base-learners), каждый из которых построен при использовании сбалансированной выборки исходных данных.

Согласно принципам однородного ансамблирования, решающая функция вычисляется как среднее большого числа отдельных учеников (или базовых классификаторов), где каждый отдельный ученик основан на случайной выборке исходных наблюдений.

Паспорта скользящего контроля (cv-passports) напрямую связаны с однородным ансамблированием. Предположим, что мы используем 75% всех доступных нам данных для тренировки. Тогда оставшиеся 25% мы можем использовать для валидационного контроля. По мере вычисления ансамбля, валидационные результаты накапливаются и образуют паспорт скользящего контроля.

Следующая функция реализует метод случайных сбалансированных подмножеств с использованием паспортов скользящего контроля. В качестве классификационной модели мы использовали функцию *gbm*. Необходимо предварительно подключить библиотеки PROC, gbm. Результаты выполнения кода приведены в Таблице 1.

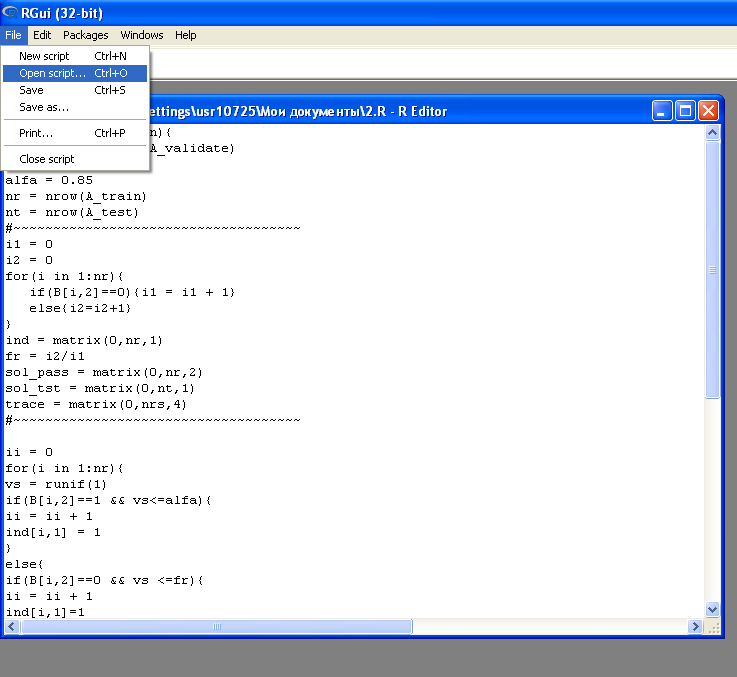
|  |
| --- |
| B = rbind(A\_train,A\_validate)  nrs = 1  alfa = 0.85  nr = nrow(A\_train)  nt = nrow(A\_test)  #~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  i1 = 0  i2 = 0  for(i in 1:nr){  if(B[i,2]==0){i1 = i1 + 1}  else{i2=i2+1}  }  ind = matrix(0,nr,1)  fr = i2/i1  sol\_pass = matrix(0,nr,2)  sol\_tst = matrix(0,nt,1)  trace = matrix(0,nrs,4)  #~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  for(jj in 1:nrs){  ii = 0  for(i in 1:nr){  vs = runif(1)  if(B[i,2]==1 && vs<=alfa){  ii = ii + 1  ind[i,1] = 1  }  else{  if(B[i,2]==0 && vs <=fr){  ii = ii + 1  ind[i,1]=1  }  else{ind[i,1]=0}  }  }  object = gbm(B[ind==1,]$SeriousDlqin2yrs~.,data=B[ind==1,],distribution="adaboost",n.trees=100,shrinkage=0.01,interaction.depth=10,n.minobsinnode=7)  tr <- predict(object,newdata=B,n.trees=100,type="response")  sol\_tst = sol\_tst + predict(object,newdata=A\_test,n.trees=100,type="response")  k = 0  k1 = 0  for(i in 1:nr){  if(ind[i,1] == 1){k = k + 1}  else{  sol\_pass[i,1] = sol\_pass[i,1] + 1  sol\_pass[i,2] = sol\_pass[i,2] + tr[i]  k1 = k1 + 1  }  }  jj  length(ind[ind==1])  length(sol\_pass[sol\_pass[sol\_pass[i,1]!=0],1])  auc(B[,2],tr)  } |

Табл. 1. Метод однородного ансамблирования: результаты выполнения (для параметра nrs=10)

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 6367 | 43633 | 0.8599 |
| 2 | 6292 | 47267 | 0.8520 |
| 3 | 6388 | 47828 | 0.8567 |
| 4 | 6291 | 48187 | 0.8532 |
| 5 | 6299 | 48468 | 0.8543 |
| 6 | 6200 | 48698 | 0.8549 |
| 7 | 6300 | 48896 | 0.8549 |
| 8 | 6362 | 49058 | 0.8531 |
| 9 | 6224 | 49200 | 0.8543 |
| 10 | 6314 | 49307 | 0.8558 |

В Табл. 1. представлены: первый столбец – индекс подмножества, второй столбец – размер подмножества, третий столбец – общее кол-во валидационных данных, четвёртый столбец – AUC, соответствующий подмножеству.

Подобные скрипты предпочтительнее выполнять через редактор скриптов.



Для выполнения необходимо выделить необходимые строки, чаще – все (ctrl+A) и выполнить, нажав  или ctrl+R.

### Нейронные сети и особенности их работы с данными Credit

Искусственная нейронная сеть — математическая модель, а также её программная или аппаратная реализации, построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. Это понятие возникло при изучении процессов, протекающих в мозге, и при попытке смоделировать эти процессы.

С точки зрения машинного обучения, нейронная сеть представляет собой обобщение линейной регрессии.

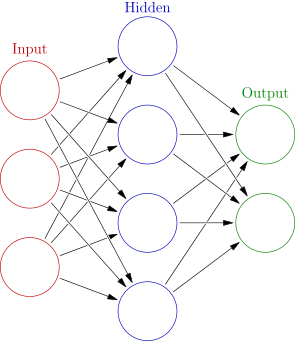


Рис. 6. Схема простой нейронной сети

Пакет *nnet* позволяет настраивать и использовать нейронные сети с одним скрытым слоем и multinomial log-linear модели. При помощи команды *library(nnet)* мы загрузим пакет в среду *R*. Итак, рассмотрим простейшую нейронную сеть:

|  |
| --- |
| B = rbind(A\_train) #объединяем наборы данных  coeff <- lm(B$SeriousDlqin2yrs~., data=B) #обучаем линейную модель  x <- predict(coeff, A\_validate) #строим прогноз на основании неё  object <- nnet(B$SeriousDlqin2yrs~., data=B, size=20) #обучаем нейронную сеть  pr <- predict(object, A\_validate) #строим прогноз на основании неё  plot(x, pr, type="b") #строим график |

В результате применения функции *nnet* к данным “Credit” мы получаем тривиальное нулевое решение. Проблема заключается в несбалансированности данных. Как это было отмечено ранее, количество плохих клиентов в разы меньше, чем хороших. Нейронная сеть просто игнорирует это меньшинство.

Рассмотрим следующий код, используемый для формирования вертикального индекса отбора наблюдений в сбалансированное подмножество:

|  |
| --- |
| B = rbind(A\_train) #объединяем наборы данных  ind=matrix(0,nr,1)  alfa=0.85 #указывает, что используется 85% плохих клиентов  beta=0.05 #указывает, что используется 5% хороших клиентов  for(i in 1:nr){  q = runif(1)  if(B[i,2]==1 && q<=alfa){ind[i,1]=1}  else{  q = runif(1)  if(B[i,2]==0 && q<=beta){ind[i,1]=1}}}  object <- nnet(B[ind==1,]$SeriousDlqin2yrs~., data=B[ind==1,], size=20)  pr <- predict(object, A\_validate)  plot(pr) #строим график |

Согласно значениям регулирующих параметров *alfa* и *beta*, соотношение положительных и отрицательных клиентов в каждом случайном подмножестве будет приблизительно равным, и при обучении модели будем использовать только отобранные данные.

### Отбор признаков при помощи метода Кёрнел-Фишер дискриминант

Кёрнел-Фишер дискриминант:

 (1)

где *μ* и *σ* соответствуют среднему и дисперсии внутри класса. Данный показатель может быть использован для отбора наиболее влиятельных признаков с тем, чтобы уменьшить проблему перетренировки, см. Рис. 7.

Алгоритм KFD реализуется следующим образом:

|  |
| --- |
| A <- read.table("F:/cs-test.csv",header=TRUE,sep=",",NA=0)  A\_train<- A[sample(1:nrow(A), 80000, replace=FALSE),]  A0 <- A\_train[A\_train[,2]==0,]  A1 <- A\_train[A\_train[,2]==1,]  nc=10  rates = matrix(0,1,nc)  for(i in 1:nc){  i1=i+2  m0 = mean(A0[,i1])  m1 = mean(A1[,i1])  s0 = sd(A0[,i1])  s1 = sd(A1[,i1])  rates[i] = abs(m0-m1)/(s0+s1)  }    barplot(rates, main="KFD", xlab="feature", names.arg=c(1:10)) |

В результате получим график, где по горизонтальной оси отмечен номер переменной, а по вертикальной – важность.

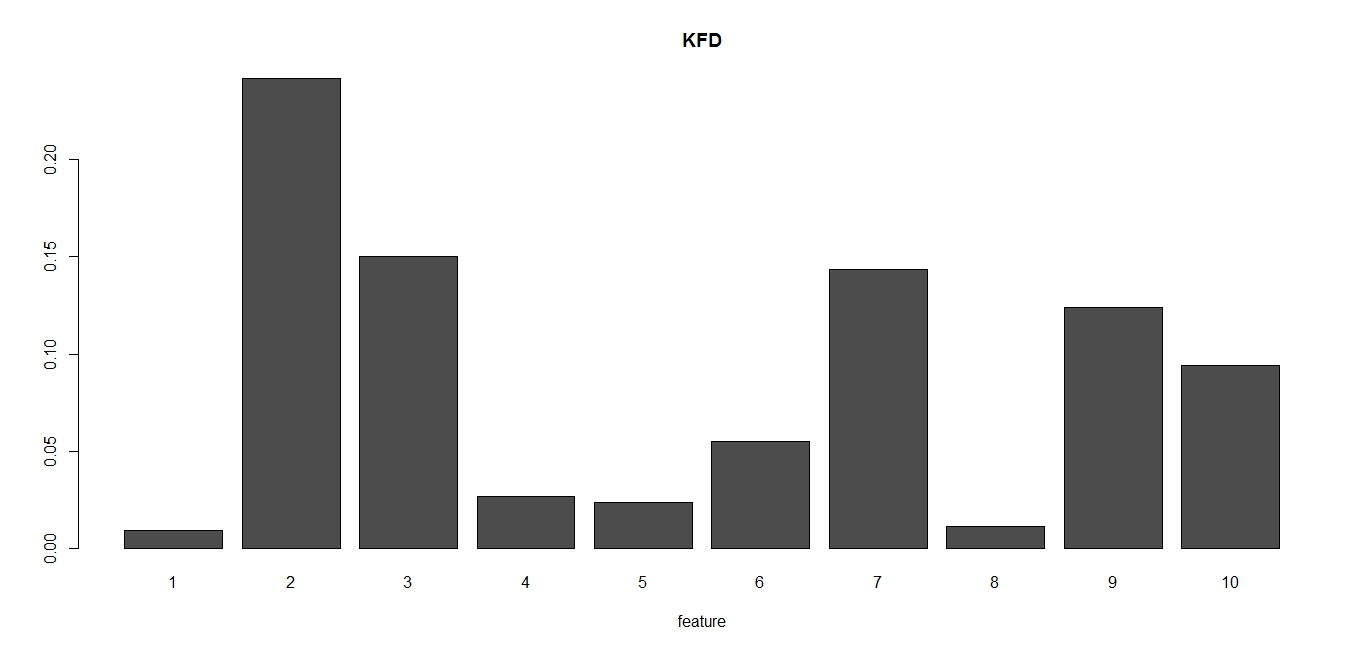


Рис. 8. График значимости переменных.

По графику видно, что переменные *v1, v4, v5, v8* – незначимы. Рассмотрим линейный прогноз при использовании этих признаков.

|  |
| --- |
| object <- lm(A\_train$SeriousDlqin2yrs~A\_train$RevolvingUtilizationOfUnsecuredLines+A\_train$DebtRatio+A\_train$MonthlyIncome+A\_train$NumberRealEstateLoansOrLines,data=A)  sol\_vld <- predict(object,newdata=A\_train)  Z <- auc(A\_train[,2],sol\_vld)  print(Z) |

Получаем значение *AUC*, равное 0.5516. Это говорит о крайне низком качестве классификации.

Проверим тоже самое для значимых переменных – *v2*, *v3*, *v7*, *v9*, *v10*. *AUC* в этом случае примет значение 0.6951, что значительно лучше.

Такое разделение весьма условно. Так, в качестве незначимых переменных можно принять только *v1, v8* и т.д., основным критерием является *AUC*, т.е. от незначимых переменных можно избавляться только, если значение *AUC* не сильно отличается от *AUC* полной модели (использующей все переменные).

### Задание

1) Провести 10 опытов по определению «хороших» и «плохих» клиентов на различных случайных выборках. Использовать данные cs-test.csv из предыдущих работ. Результат представить в таблице:

|  |  |  |
| --- | --- | --- |
| № выборки (опыта) | Количество «плохих» клиентов | % «плохих» клиентов |
| 1 | 344 | 7,043 |
| 2 | 694 | 6,94 |
| 3 | 1285 | 6,425 |
| 4 | 1959 | 6,53 |
| 5 | 2755 | 6,8875 |
| 6 | 3377 | 6,754 |
| 7 | 4044 | 6,74 |
| 8 | 4622 | 6,6 |
| 9 | 5223 | 6,52 |
| 10 | 6024 | 6,69 |

2) Выполнить метод случайных сбалансированных подмножеств с использованием паспортов скользящего контроля.

Провести ряд опытов с различными значениями параметра alpha (0.75,0.8, 0.85,0.9, 0.95).

|  |  |  |  |
| --- | --- | --- | --- |
| alpha | размер подмножества | общее кол-во валидационных данных | AUC, соответствующий подмножеству |
| 0.75 | 9400 | 70600 | 0.8598 |
| 0.8 | 9469 | 70531 | 0.8599 |
| 0.85 | 9845 | 70155 | 0.8595 |
| 0.9 | 10295 | 69705 | 0.8599 |
| 0.95 | 10483 | 69517 | 0.8595 |

3) Построить модели нейронной сети с числом нейронов скрытого слоя 3, 10, 20. Вывести графики.

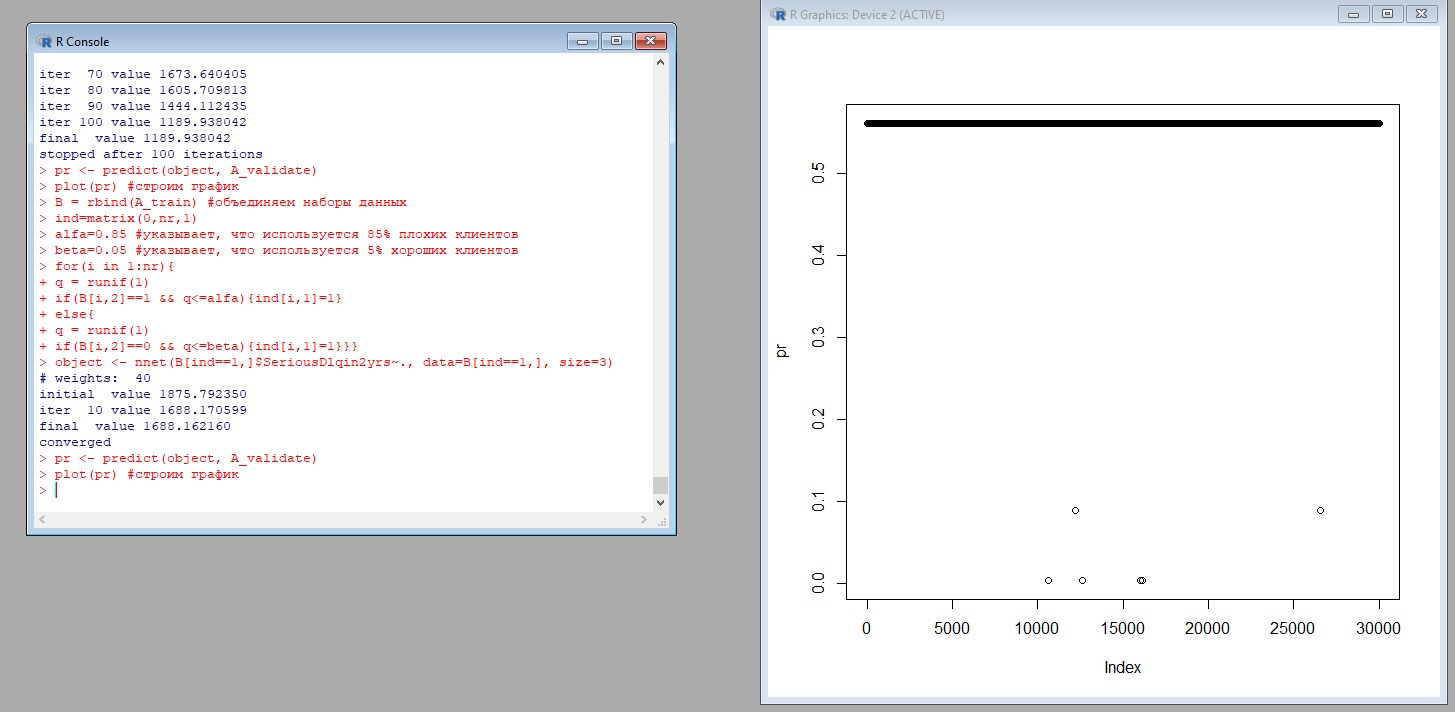


Рисунок 1 – Число нейронов 3.

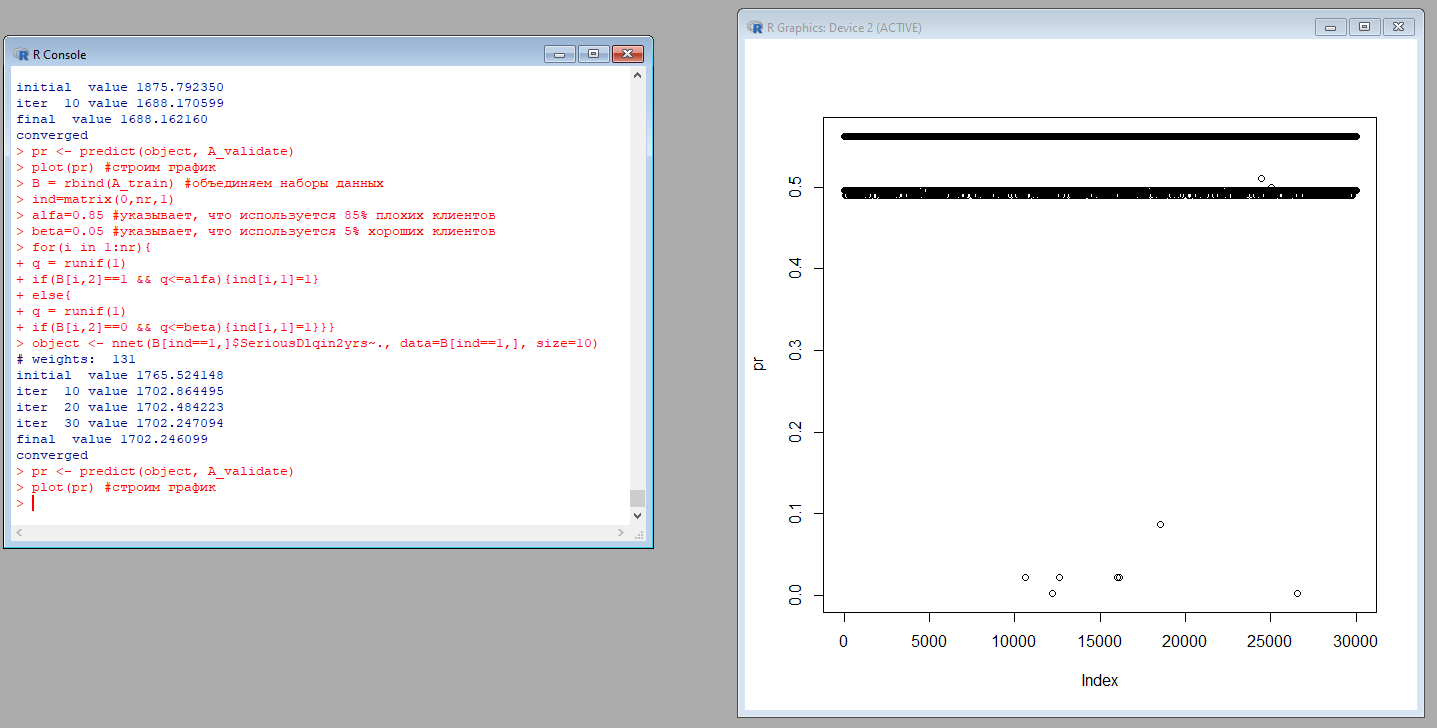


Рисунок 2 – Число нейронов 10.

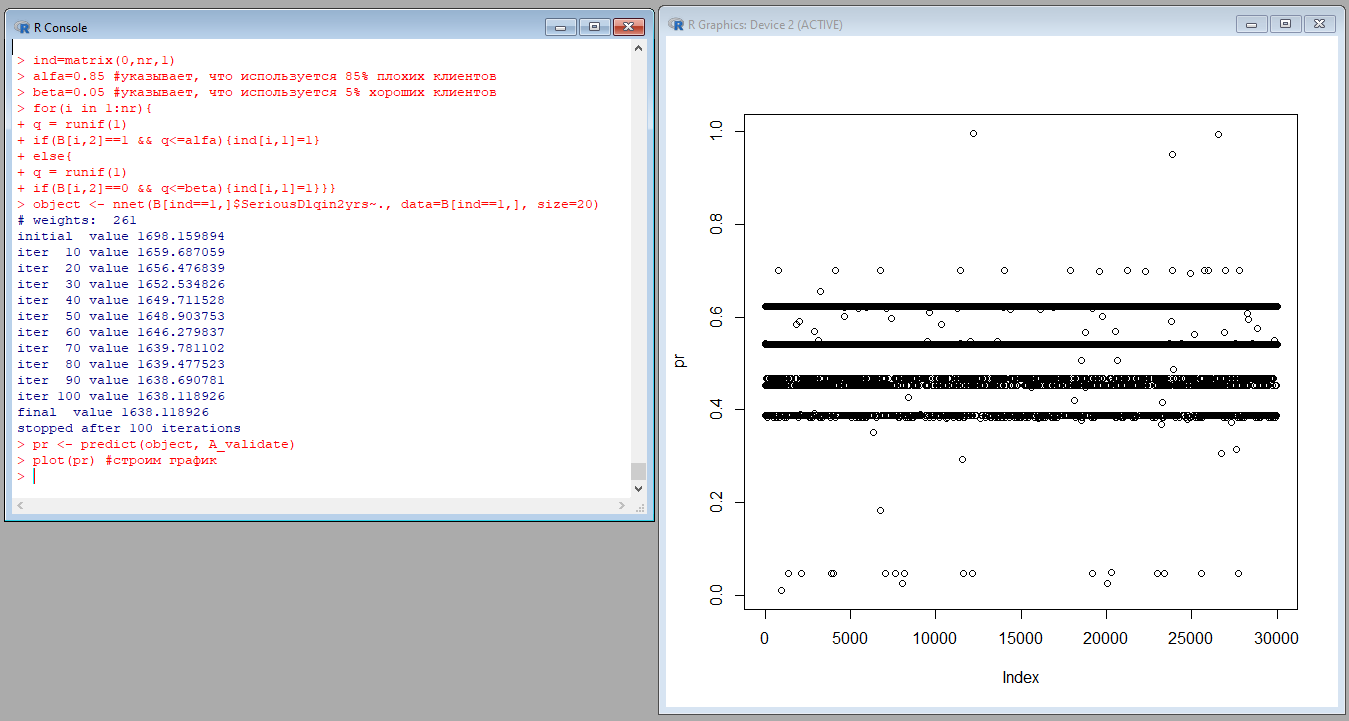


Рисунок 3 – Число нейронов 20.

4) Определить незначимые переменные для данных Credits. Сравнить AUC построенной модели и полной модели.

AUC полной модели = 0.5422

AUC построенной модели = 0.6832

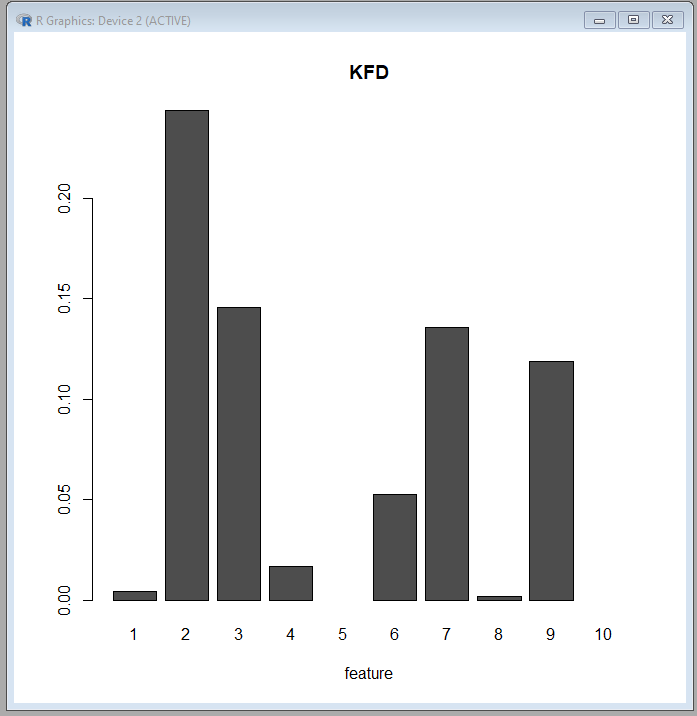


Рисунок 4 – Значимые и незначимые переменные.