R_Code.R

```r
# NHANES - PAD
# National Health and Nutrition Examination Survey (NHANES) - Peripheral Arte
ry Disease (PAD)
# Data collection:
# Household screener, interview, and physical examination
# Objectives:
# Understand the survey data and create a predictive model to identify the
# main factors that are related to the disease. The model can also be
# useful to prioritize the physical exams and to support the diagnostics.
# Activities:
# - Start the session
# - Prepare the data for Modelling
# - Data Partition (Training and Validation)
# - Feature Engineering (add additional features)
# - Modelling
# - Scoring

####### SET UP THE CONFIGURATION FOR RStudio IN ORDER TO WORK IN CAS FROM R

## Run the following code on R to install following packages before running t
his:
# 1) install.packages("pkgbuild")
# 2) install.packages('jsonlite')
# 3) install.packages("tidyverse")

## Download and install RTools from:
# 4) https://cran.r-project.org/bin/windows/Rtools/

## Install the SWAT vX.X.X package as indicated here:
# 5) https://github.com/sassoftware/R-swat/releases

## Start the session and prepare the environment
library(swat)

## SWAT 1.8.2

library(ggplot2)

# Connect to CAS (this depends on the environment configuration)
your_host <- 'link_to_your_host'
your_port <- XXX
your_username <- 'user'
your_password <- 'password'

conn <- swat::CAS(your_host, your_port, protocol='auto', username=your_userna
me, password=your_password)
```

```
## NOTE: Connecting to CAS and generating CAS action functions for loaded
##       action sets...

## NOTE: To generate the functions with signatures (for tab completion), set
##       options(cas.gen.function.sig=TRUE).

# CAS Server connection details
# out <- cas.builtins.serverStatus(conn)
# print(out)

### Import action sets
cas.builtins.loadActionSet(conn,actionSet="dataStep")

## NOTE: Added action set 'dataStep'.

## NOTE: Information for action set 'dataStep':

## NOTE:    dataStep

## NOTE:       runCodeTable - Runs DATA step code stored in a CAS table

## NOTE:       runCode - Runs DATA step code

## $actionset
## [1] "dataStep"

cas.builtins.loadActionSet(conn,actionSet="dataPreprocess")

## NOTE: Added action set 'dataPreprocess'.

## NOTE: Information for action set 'dataPreprocess':

## NOTE:    dataPreprocess

## NOTE:       rustats - Computes robust univariate statistics, centralized m
oments, quantiles, and frequency distribution statistics

## NOTE:       impute - Performs data matrix (variable) imputation

## NOTE:       outlier - Performs outlier detection and treatment

## NOTE:       binning - Performs unsupervised variable discretization

## NOTE:       discretize - Performs supervised and unsupervised variable dis
cretization

## NOTE:       catTrans - Groups and encodes categorical variables using unsu
pervised and supervised grouping techniques

## NOTE:       histogram - Generates histogram bins and simple bin-based stat
istics for numeric variables
```

```
## NOTE:       transform - Performs pipelined variable imputation, outlier de
tection and treatment, functional transformation, binning, and robust univari
ate statistics to evaluate the quality of the transformation

## NOTE:       kde - Computes kernel density estimation

## NOTE:       highCardinality - Performs randomized cardinality estimation

## $actionset
## [1] "dataPreprocess"

cas.builtins.loadActionSet(conn,actionSet="cardinality")

## NOTE: Added action set 'cardinality'.

## NOTE: Information for action set 'cardinality':

## NOTE:    cardinality

## NOTE:       summarize - Provides actions for evaluating data cardinality

## $actionset
## [1] "cardinality"

cas.builtins.loadActionSet(conn,actionSet="sampling")

## NOTE: Added action set 'sampling'.

## NOTE: Information for action set 'sampling':

## NOTE:    sampling

## NOTE:       srs -  Samples a proportion of data from the input table or pa
rtitions the data into no more than three portions

## NOTE:       stratified - Samples a proportion of data or partitions the da
ta into no more than three portions within each stratum

## NOTE:       oversample - Samples a user-specified proportion of data from
the event level and adjusts the ratio between rare events and non-rare events
to a user-specified ratio

## NOTE:       kfold - K-fold partitioning.

## $actionset
## [1] "sampling"

cas.builtins.loadActionSet(conn,actionSet="decisionTree")

## NOTE: Added action set 'decisionTree'.

## NOTE: Information for action set 'decisionTree':

## NOTE:    decisionTree
```

```
## NOTE:       dtreeTrain - Trains a decision tree

## NOTE:       dtreeScore - Scores a table using a decision tree model

## NOTE:       dtreeSplit - Splits decision tree nodes

## NOTE:       dtreePrune - Prune a decision tree

## NOTE:       dtreeMerge - Merges decision tree nodes

## NOTE:       dtreeCode - Generates DATA step scoring code from a decision t
ree model

## NOTE:       forestTrain - Trains a forest. This action requires a SAS Visu
al Data Mining and Machine Learning license

## NOTE:       forestScore - Scores a table using a forest model

## NOTE:       forestCode - Generates DATA step scoring code from a forest mo
del

## NOTE:       gbtreeTrain - Trains a gradient boosting tree. This action req
uires a SAS Visual Data Mining and Machine Learning license

## NOTE:       gbtreeScore - Scores a table using a gradient boosting tree mo
del

## NOTE:       gbtreeCode - Generates DATA step scoring code from a gradient
boosting tree model

## NOTE:       dtreeExportModel - Export the aStore model for a tree model ta
ble

## $actionset
## [1] "decisionTree"

cas.builtins.loadActionSet(conn,actionSet="astore")

## NOTE: Added action set 'astore'.

## NOTE: Information for action set 'astore':

## NOTE:    astore

## NOTE:       download - Downloads a remote store to a local store

## NOTE:       upload - Uploads a local store to a remote store

## NOTE:       describe - Describes some of the contents of the analytic stor
e

## NOTE:       score - Uses an analytic store to score an input table

## $actionset
## [1] "astore"
```

```
cas.builtins.loadActionSet(conn,actionSet="percentile")

## NOTE: Added action set 'percentile'.

## NOTE: Information for action set 'percentile':

## NOTE:    percentile

## NOTE:       percentile - Calculate quantiles and percentiles

## NOTE:       boxPlot - Calculate quantiles, high and low whiskers, and outl
iers

## NOTE:       assess - Assess and compare models

## $actionset
## [1] "percentile"

## Prepare the data for Modelling and assign variable to the table
path_to_data <- 'path_to_data_folder/nhanes_nof.sas7bdat'
cas.upload(conn,path_to_data,
           casOut=list(name="NHANES_NOF", caslib="CASUSER(alarzo)", replace=T
RUE))

## WARNING: The table NHANES_NOF exists as a global table in caslib CASUSER(a
larzo). By adding a session table with the same name, the session-scope table
takes precedence over the global-scope table.

## NOTE: Cloud Analytic Services made the uploaded file available as table NH
ANES_NOF in caslib CASUSER(alarzo).

## NOTE: The table NHANES_NOF has been created in caslib CASUSER(alarzo) from
binary data uploaded to Cloud Analytic Services.

## $performance
## $performance$cpuUserTime
## [1] 0.02347
##
## $performance$cpuSystemTime
## [1] 0.042296
##
## $performance$systemTotalMemory
## [1] 540873932800
##
## $performance$systemNodes
## [1] 1
##
## $performance$systemCores
## [1] 32
##
## $performance$memory
## [1] 15247392
```

```
##
## $performance$memoryQuota
## [1] 42749952
##
##
## $disposition
## $disposition$severity
## [1] 0
##
## $disposition$reason
## [1] "ok"
##
## $disposition$statusCode
## [1] 0
##
##
## $messages
## $messages[[1]]
## [1] "WARNING: The table NHANES_NOF exists as a global table in caslib CASU
SER(alarzo). By adding a session table with the same name, the session-scope
table takes precedence over the global-scope table."
##
## $messages[[2]]
## [1] "NOTE: Cloud Analytic Services made the uploaded file available as tab
le NHANES_NOF in caslib CASUSER(alarzo)."
##
## $messages[[3]]
## [1] "NOTE: The table NHANES_NOF has been created in caslib CASUSER(alarzo)
from binary data uploaded to Cloud Analytic Services."
##
##
## $results
## $results$caslib
## [1] "CASUSER(alarzo)"
##
## $results$tableName
## [1] "NHANES_NOF"
```

```
## Check the columns
# cas.table.columnInfo(conn,table="NHANES_NOF")

## Create the target variable
cas.dataStep.runCode(conn, code="
    data CASUSER.NHANES_PAD1 promote;
          set CASUSER.NHANES_NOF;
            if LEXRABPI = . then LEXRABPI = LEXLABPI;
            if ((LEXLABPI < 0.9) OR (LEXRABPI< 0.9 )) then PAD_Target = 1;
                else PAD_Target = 0;
    run;"  )
```

```
## $InputCasTables
##            casLib       Name Rows Columns
## 1 CASUSER(alarzo) NHANES_NOF 6929      47
##
## $OutputCasTables
##            casLib        Name Rows Columns Append Promoted
## 1 CASUSER(alarzo) NHANES_PAD1 6929      48    NaN        N
## 2 CASUSER(alarzo)     promote 6929      48    NaN        N
```

```
## Data Partition (Training and Validation)
cas.sampling.srs(conn,
                 table="NHANES_PAD1",
                 samppct=30,
                 partind=TRUE,
                 output=list(casout = list(name="NHANES_PAD_PART", replace=TR
UE),
                             copyvars="ALL")
                 )
```

```
## NOTE: Simple Random Sampling is in effect.

## NOTE: Using SEED=1042972783 for sampling.
```

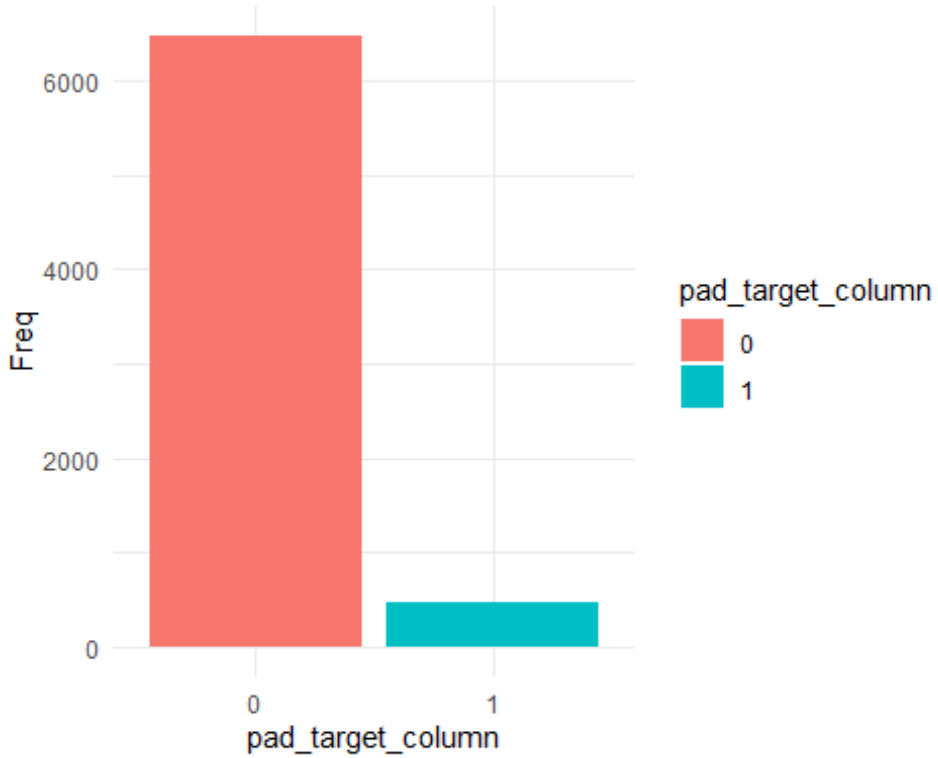
```
## $OutputCasTables
##            casLib            Name Label Rows Columns
## 1 CASUSER(alarzo) NHANES_PAD_PART       6929      49
##
## $SRSFreq
##   NObs NSamp
## 1 6929  2079
```

```
## Present the partitions on a frequancy table and a bar chart
tablesize <- 6929
# With "CAS.TABLE.FETCH"-ACTION we extract a column from a in-memory table in
to
# R-Studio local-memory. Since it is a very long column, SAS pull it as a lis
t
# with many elements. To get it as a unique vector, we use "UNLIST".
# Finally, we use only the numeric elements. For that reason we use "AS.NUMER
IC"
pad_target_column <- as.numeric(unlist(
                        cas.table.fetch(conn,
                                        table="NHANES_PAD_PART",
                                        fetchVars="PAD_TARGET",
                                        index=FALSE,
                                        to=tablesize)
                        ))

# To produce frequency table of partition elements
freq_partition <- data.frame(table(pad_target_column))
freq_partition
```

```
##   pad_target_column Freq
## 1                 0 6470
## 2                 1  459

# To produce bar chart of partition elements
ggplot(data=freq_partition, aes(x=pad_target_column, y=Freq, fill=pad_target_
column)) +
  geom_bar(stat="identity")+theme_minimal()
```




```
## Feature Engineering (add additional features)
cas.dataStep.runCode(conn,
                     code="data CASUSER.NHANES_PAD1(replace=yes);
                           set CASUSER.NHANES_PAD_PART;
                           PulsePreassure = BPXSAR - BPXDAR;
                           TC_HDL = LBXTC / LBDHDL;
                           IF ((DIQ010 In ('Yes','Borderline')) OR (DIQ050
In ('Yes')) OR (LBXGH > 6.5))
                              then Diabetes = 1;
                               else Diabetes = 0;
                           IF ( BPXSAR >= 140 OR BPXDAR >= 90 )
                              then Hypertension = 1;
                               else Hypertension = 0;
                            run; ")

## NOTE: Missing values were generated as a result of performing an operation
on missing values.
```

```
##       Each place is given by: (Number of times) at (Line):(Column).
##       4 at 0:153   2 at 0:208
##       1 at 0:153   1 at 0:208
##       4 at 0:153    19 at 0:208
##       1 at 0:153
##       6 at 0:153    22 at 0:208
##       6 at 0:153   7 at 0:208
##       9 at 0:153    16 at 0:208
##       7 at 0:153    22 at 0:208
##       4 at 0:153    11 at 0:208
##       11 at 0:153   12 at 0:208
##       4 at 0:153    16 at 0:208
##       6 at 0:153    11 at 0:208
##       9 at 0:153    14 at 0:208
##       5 at 0:153    12 at 0:208
##       8 at 0:153   9 at 0:208
##       12 at 0:153   6 at 0:208
##       5 at 0:153    15 at 0:208
##       8 at 0:153   7 at 0:208
##       6 at 0:153    18 at 0:208
##       5 at 0:153    19 at 0:208
##       1 at 0:153    10 at 0:208
##       12 at 0:153   11 at 0:208
##       3 at 0:153    22 at 0:208
##       10 at 0:153   19 at 0:208
## NOTE: Duplicate messages output by DATA step:
## NOTE: Missing values were generated as a result of performing an operation
on missing values.  (occurred 25 times)
```

```
##       Each place is given by: (Number of times) at (Line):(Column).  (occu
rred 25 times)

##       5 at 0:153    12 at 0:208  (occurred 2 times)

## $InputCasTables
##            casLib            Name Rows Columns
## 1 CASUSER(alarzo) NHANES_PAD_PART 6929      49
##
## $OutputCasTables
##            casLib        Name Rows Columns Append Promoted
## 1 CASUSER(alarzo) NHANES_PAD1 6929      53    NaN        N
```

```
## Check the columns
# cas.table.columnInfo(conn,table="NHANES_PAD1")

## Modelling

# Specify the data set inputs and target
interval_inputs <- c('RIDAGEMN_Recode', 'PulsePreassure', 'BMXBMI', 'TC_HDL',
'LBXGH', 'Diabetes', 'Hypertension')
class_inputs <- c('INDHHINC', 'DMDEDUC2', 'RIDRETH1', 'DIQ150', 'DIQ110', 'SM
Q040', 'ALQ100', 'RIAGENDR')
class_vars   <- c('INDHHINC', 'DMDEDUC2', 'RIDRETH1', 'DIQ150', 'DIQ110', 'SM
Q040', 'ALQ100', 'RIAGENDR', 'PAD_Target')
target       <- 'PAD_Target'

# Specify a generic cut-off
Gen_cutoff <- 0.5

# Train the model
cas.decisionTree.gbtreeTrain(conn,
                             table=list(name="NHANES_PAD1", where="strip(put
(_PartInd_, best.))='0'"),
                             target=target,
                             inputs=c(class_inputs,interval_inputs),
                             nominals=class_vars,
                             nTree=150,  m=7,  lasso=0.777,  learningrate=1,
subsamplerate=0.883,  ridge=6.03,  seed=1634211770,
                             leafsize=5,  maxbranch=2,  binorder=TRUE,  enco
dename=TRUE,  mergebin=TRUE,  nBins=20,  maxLevel=6,
                             varImp=TRUE,  missing="USEINSEARCH",
                             casOut=list(name="gb_model", replace=TRUE)
)
```

```
## $DTreeVarImpInfo
##           Variable Importance       Std
## 1         INDHHINC 1.04495306 0.6177220
## 2  RIDAGEMN_Recode 0.84121549 4.4450680
## 3   PulsePreassure 0.55166643 0.5909193
```

```
## 4           BMXBMI 0.46825188 0.3422344
## 5           TC_HDL 0.45360413 0.2687546
## 6            LBXGH 0.44745203 0.3870789
## 7           SMQ040 0.36226398 1.7154911
## 8         RIDRETH1 0.28482808 0.5655375
## 9         DMDEDUC2 0.28295051 0.2753196
## 10          DIQ150 0.13432039 0.7537135
## 11          DIQ110 0.12025807 0.2131510
## 12          ALQ100 0.09798494 0.4731479
## 13    Hypertension 0.05188719 0.3039457
## 14        Diabetes 0.05013120 0.2181774
## 15        RIAGENDR 0.04074772 0.3559001
##
## $ModelInfo
##                                Descr        Value
## 1                   Number of Trees 1.500000e+02
## 2                      Distribution 2.000000e+00
## 3                     Learning Rate 1.000000e+00
## 4                   Subsampling Rate 8.830000e-01
## 5  Number of Selected Variables (M) 7.000000e+00
## 6                     Number of Bins 2.000000e+01
## 7                Number of Variables 1.500000e+01
## 8          Max Number of Tree Nodes 6.300000e+01
## 9          Min Number of Tree Nodes 2.500000e+01
## 10           Max Number of Branches 2.000000e+00
## 11           Min Number of Branches 2.000000e+00
## 12             Max Number of Levels 6.000000e+00
## 13             Min Number of Levels 6.000000e+00
## 14             Max Number of Leaves 3.200000e+01
## 15             Min Number of Leaves 1.300000e+01
## 16           Maximum Size of Leaves 1.870000e+03
## 17           Minimum Size of Leaves 5.000000e+00
## 18               Random Number Seed 1.634212e+09
## 19               Lasso (L1) penalty 7.770000e-01
## 20               Ridge (L2) penalty 6.030000e+00
## 21           Actual Number of Trees 1.500000e+02
## 22         Average number of Leaves 2.444667e+01
##
## $OutputCasTables
##            casLib     Name Rows Columns
## 1 CASUSER(alarzo) gb_model 7184      46
```

```
## Score the model
cas.decisionTree.gbtreeScore(conn,
                             table=list(name="NHANES_PAD1"),
                             modelTable=list(name="gb_model"),
                             casOut=list(name='scored_gb', replace=TRUE),
                             copyVars=list("PAD_Target", "_PartInd_"),
                             encodename = TRUE,
```

```
                              assessonerow = TRUE
)

## $EncodedName
##        LEVNAME LEVINDEX       VARNAME
## 1            1        0 P_PAD_Target1
## 2            0        1 P_PAD_Target0
##
## $EncodedTargetName
##   LEVNAME LEVINDEX      VARNAME
## 1               0 I_PAD_Target
##
## $ErrorMetricInfo
##     TreeID Trees NLeaves        MCR   LogLoss        ASE      RASE     MAX
AE
## 1        0     1      28 0.06624333 0.2194148 0.05988815 0.2447206 0.96938
26
## 2        1     2      56 0.06609900 0.2027616 0.05533159 0.2352267 0.98582
40
## 3        2     3      82 0.06537740 0.1951989 0.05317783 0.2306032 0.99183
82
## 4        3     4     113 0.06407851 0.1891716 0.05184391 0.2276926 0.99480
96
## 5        4     5     144 0.06148073 0.1835200 0.05027885 0.2242295 0.99676
94
## 6        5     6     174 0.06075913 0.1786750 0.04894655 0.2212387 0.99596
96
## 7        6     7     205 0.06090345 0.1752561 0.04785295 0.2187532 0.99723
80
## 8        7     8     237 0.05772839 0.1721860 0.04677588 0.2162773 0.99766
14
## 9        8     9     262 0.05830567 0.1696846 0.04628383 0.2151368 0.99819
35
## 10       9    10     289 0.05715110 0.1664084 0.04541226 0.2131015 0.99838
57
## 11      10    11     321 0.05729543 0.1637129 0.04463766 0.2112763 0.99820
37
## 12      11    12     350 0.05686246 0.1613807 0.04382542 0.2093452 0.99783
78
## 13      12    13     382 0.05599654 0.1591302 0.04307493 0.2075450 0.99859
61
## 14      13    14     414 0.05397604 0.1568138 0.04232646 0.2057340 0.99853
48
## 15      14    15     446 0.05397604 0.1537033 0.04147311 0.2036495 0.99897
15
## 16      15    16     474 0.05094530 0.1515947 0.04048426 0.2012070 0.99885
30
## 17      16    17     506 0.05065666 0.1494457 0.03968736 0.1992169 0.99913
88
## 18      17    18     536 0.05036802 0.1471728 0.03862847 0.1965413 0.99936
```

```
61
## 19      18    19     558 0.04849185 0.1455939 0.03833423 0.1957913 0.99940
68
## 20      19    20     588 0.04719296 0.1444404 0.03765937 0.1940602 0.99928
32
## 21      20    21     617 0.04430654 0.1422244 0.03681740 0.1918786 0.99951
31
## 22      21    22     648 0.04387357 0.1412579 0.03630725 0.1905446 0.99969
96
## 23      22    23     678 0.04358493 0.1393465 0.03557285 0.1886077 0.99966
96
## 24      23    24     709 0.04257469 0.1389331 0.03530642 0.1879000 0.99965
63
## 25      24    25     740 0.04142012 0.1379942 0.03495654 0.1869667 0.99973
28
## 26      25    26     770 0.04026555 0.1358480 0.03414529 0.1847845 0.99978
26
## 27      26    27     796 0.03939962 0.1342035 0.03374582 0.1837004 0.99980
10
## 28      27    28     826 0.03954395 0.1329134 0.03328114 0.1824312 0.99982
43
## 29      28    29     853 0.03810074 0.1317945 0.03278742 0.1810730 0.99986
85
## 30      29    30     884 0.03737913 0.1303998 0.03227251 0.1796455 0.99991
08
## 31      30    31     912 0.03752345 0.1295535 0.03185443 0.1784781 0.99988
02
## 32      31    32     942 0.03622456 0.1289201 0.03146971 0.1773970 0.99992
13
## 33      32    33     970 0.03810074 0.1284048 0.03128586 0.1768781 0.99988
57
## 34      33    34     999 0.03636888 0.1275673 0.03101773 0.1761185 0.99985
79
## 35      34    35    1029 0.03449271 0.1262531 0.03053850 0.1747527 0.99985
79
## 36      35    36    1055 0.03478135 0.1254511 0.02992796 0.1729970 0.99985
96
## 37      36    37    1084 0.03478135 0.1251913 0.02963909 0.1721601 0.99987
66
## 38      37    38    1111 0.03478135 0.1238677 0.02929433 0.1711559 0.99989
97
## 39      38    39    1139 0.03377111 0.1226153 0.02894644 0.1701365 0.99990
09
## 40      39    40    1170 0.03420407 0.1229753 0.02895235 0.1701539 0.99990
80
## 41      40    41    1200 0.03319382 0.1221310 0.02856038 0.1689982 0.99987
97
## 42      41    42    1228 0.03247222 0.1217455 0.02817748 0.1678615 0.99987
79
## 43      42    43    1258 0.03160629 0.1217747 0.02796500 0.1672274 0.99987
```

```
79
## 44      43    44    1287 0.03088469 0.1211079 0.02768060 0.1663749 0.99989
25
## 45      44    45    1316 0.03045172 0.1207887 0.02745929 0.1657084 0.99991
20
## 46      45    46    1345 0.03059605 0.1206068 0.02741983 0.1655893 0.99993
24
## 47      46    47    1375 0.02973012 0.1196409 0.02702699 0.1643989 0.99993
49
## 48      47    48    1400 0.02944148 0.1198514 0.02689806 0.1640063 0.99993
49
## 49      48    49    1428 0.02958580 0.1198354 0.02676944 0.1636137 0.99993
08
## 50      49    50    1455 0.02973012 0.1196954 0.02675469 0.1635686 0.99994
54
## 51      50    51    1484 0.02987444 0.1194331 0.02660481 0.1631098 0.99995
08
## 52      51    52    1508 0.02944148 0.1189167 0.02628120 0.1621148 0.99994
44
## 53      52    53    1538 0.02843123 0.1184713 0.02612137 0.1616211 0.99994
44
## 54      53    54    1563 0.02785395 0.1181798 0.02591347 0.1609766 0.99995
36
## 55      54    55    1594 0.02814259 0.1180376 0.02576269 0.1605076 0.99993
76
## 56      55    56    1624 0.02785395 0.1177876 0.02564978 0.1601555 0.99994
46
## 57      56    57    1653 0.02814259 0.1176966 0.02552830 0.1597758 0.99995
49
## 58      57    58    1677 0.02742098 0.1177341 0.02555607 0.1598627 0.99994
99
## 59      58    59    1704 0.02684370 0.1175049 0.02535244 0.1592245 0.99995
32
## 60      59    60    1732 0.02583345 0.1171037 0.02513927 0.1585537 0.99994
99
## 61      60    61    1756 0.02641074 0.1176106 0.02515658 0.1586083 0.99994
21
## 62      61    62    1786 0.02626642 0.1172461 0.02491112 0.1578326 0.99994
21
## 63      62    63    1815 0.02597777 0.1171786 0.02484646 0.1576276 0.99994
36
## 64      63    64    1840 0.02612210 0.1169402 0.02472020 0.1572266 0.99994
36
## 65      64    65    1863 0.02612210 0.1167672 0.02460291 0.1568531 0.99994
36
## 66      65    66    1890 0.02583345 0.1166485 0.02450196 0.1565310 0.99994
63
## 67      66    67    1919 0.02554481 0.1166191 0.02433105 0.1559841 0.99994
58
## 68      67    68    1945 0.02511185 0.1161239 0.02418079 0.1555017 0.99994
```

```
05
## 69      68    69    1969 0.02554481 0.1163532 0.02422847 0.1556550 0.99994
97
## 70      69    70    1996 0.02540049 0.1163599 0.02422199 0.1556341 0.99995
82
## 71      70    71    2025 0.02525617 0.1162270 0.02417893 0.1554957 0.99995
82
## 72      71    72    2048 0.02467889 0.1160435 0.02403463 0.1550310 0.99995
13
## 73      72    73    2074 0.02511185 0.1157842 0.02392009 0.1546612 0.99995
92
## 74      73    74    2104 0.02453456 0.1155296 0.02367824 0.1538773 0.99996
73
## 75      74    75    2128 0.02467889 0.1156141 0.02370349 0.1539594 0.99996
07
## 76      75    76    2152 0.02467889 0.1153753 0.02358465 0.1535729 0.99996
68
## 77      76    77    2179 0.02453456 0.1149917 0.02342620 0.1530562 0.99996
47
## 78      77    78    2202 0.02439024 0.1150853 0.02340205 0.1529773 0.99996
05
## 79      78    79    2227 0.02395728 0.1147920 0.02326686 0.1525348 0.99996
63
## 80      79    80    2251 0.02424592 0.1148279 0.02322194 0.1523875 0.99996
63
## 81      80    81    2276 0.02410160 0.1148631 0.02319685 0.1523051 0.99996
32
## 82      81    82    2302 0.02453456 0.1145263 0.02314293 0.1521280 0.99996
32
## 83      82    83    2323 0.02424592 0.1144925 0.02310814 0.1520136 0.99996
62
## 84      83    84    2349 0.02424592 0.1143135 0.02303488 0.1517725 0.99997
07
## 85      84    85    2370 0.02424592 0.1140357 0.02300035 0.1516586 0.99996
64
## 86      85    86    2388 0.02424592 0.1142366 0.02293500 0.1514431 0.99997
22
## 87      86    87    2413 0.02381296 0.1139826 0.02288725 0.1512853 0.99996
80
## 88      87    88    2438 0.02467889 0.1143097 0.02286737 0.1512196 0.99997
03
## 89      88    89    2467 0.02395728 0.1142631 0.02276360 0.1508761 0.99997
48
## 90      89    90    2495 0.02395728 0.1141786 0.02275419 0.1508449 0.99997
15
## 91      90    91    2515 0.02395728 0.1140929 0.02273454 0.1507798 0.99996
71
## 92      91    92    2539 0.02395728 0.1137375 0.02265825 0.1505266 0.99996
32
## 93      92    93    2558 0.02395728 0.1135935 0.02257408 0.1502467 0.99996
```

```
01
## 94      93    94    2585 0.02366864 0.1135676 0.02247222 0.1499074 0.99996
70
## 95      94    95    2607 0.02395728 0.1137098 0.02239291 0.1496426 0.99997
07
## 96      95    96    2633 0.02410160 0.1136250 0.02239225 0.1496404 0.99997
07
## 97      96    97    2659 0.02410160 0.1135120 0.02235404 0.1495127 0.99996
97
## 98      97    98    2681 0.02366864 0.1133001 0.02223710 0.1491211 0.99996
97
## 99      98    99    2708 0.02453456 0.1131791 0.02227069 0.1492337 0.99996
82
## 100     99   100    2733 0.02424592 0.1131912 0.02221187 0.1490365 0.99996
77
## 101    100   101    2753 0.02453456 0.1134075 0.02229641 0.1493198 0.99996
94
## 102    101   102    2774 0.02467889 0.1136015 0.02227946 0.1492630 0.99996
94
## 103    102   103    2796 0.02395728 0.1136173 0.02218132 0.1489340 0.99997
33
## 104    103   104    2815 0.02439024 0.1134755 0.02216690 0.1488855 0.99997
33
## 105    104   105    2836 0.02424592 0.1133409 0.02212538 0.1487460 0.99997
33
## 106    105   106    2854 0.02467889 0.1134359 0.02213295 0.1487715 0.99997
33
## 107    106   107    2875 0.02424592 0.1133640 0.02205532 0.1485103 0.99997
48
## 108    107   108    2898 0.02424592 0.1136063 0.02211010 0.1486946 0.99997
48
## 109    108   109    2916 0.02424592 0.1135287 0.02208725 0.1486178 0.99997
24
## 110    109   110    2936 0.02424592 0.1134638 0.02209893 0.1486571 0.99997
07
## 111    110   111    2956 0.02410160 0.1135683 0.02209388 0.1486401 0.99997
51
## 112    111   112    2978 0.02439024 0.1135395 0.02207753 0.1485851 0.99997
52
## 113    112   113    2997 0.02424592 0.1135041 0.02202085 0.1483942 0.99997
52
## 114    113   114    3022 0.02439024 0.1134757 0.02200436 0.1483387 0.99997
32
## 115    114   115    3040 0.02410160 0.1136800 0.02203344 0.1484367 0.99997
63
## 116    115   116    3057 0.02424592 0.1136611 0.02207569 0.1485789 0.99997
62
## 117    116   117    3074 0.02410160 0.1133324 0.02198178 0.1482625 0.99997
62
## 118    117   118    3094 0.02410160 0.1131869 0.02199643 0.1483119 0.99997
```

```
43
## 119    118   119    3113 0.02410160 0.1131612 0.02195574 0.1481747 0.99997
43
## 120    119   120    3131 0.02395728 0.1128361 0.02184702 0.1478074 0.99997
56
## 121    120   121    3149 0.02410160 0.1127946 0.02185928 0.1478489 0.99997
61
## 122    121   122    3172 0.02424592 0.1126032 0.02176640 0.1475344 0.99997
33
## 123    122   123    3193 0.02395728 0.1130660 0.02177833 0.1475748 0.99997
69
## 124    123   124    3213 0.02395728 0.1128943 0.02176344 0.1475244 0.99997
96
## 125    124   125    3229 0.02410160 0.1128512 0.02177883 0.1475765 0.99997
78
## 126    125   126    3242 0.02424592 0.1127557 0.02179317 0.1476251 0.99998
04
## 127    126   127    3257 0.02424592 0.1126666 0.02180428 0.1476627 0.99997
87
## 128    127   128    3274 0.02395728 0.1129574 0.02178514 0.1475979 0.99998
19
## 129    128   129    3294 0.02424592 0.1132681 0.02178892 0.1476107 0.99998
12
## 130    129   130    3309 0.02410160 0.1130785 0.02181046 0.1476837 0.99998
12
## 131    130   131    3324 0.02395728 0.1132568 0.02182529 0.1477339 0.99998
12
## 132    131   132    3344 0.02395728 0.1133443 0.02181584 0.1477019 0.99998
33
## 133    132   133    3365 0.02381296 0.1134164 0.02179765 0.1476403 0.99998
16
## 134    133   134    3382 0.02395728 0.1133906 0.02174700 0.1474686 0.99997
97
## 135    134   135    3404 0.02410160 0.1132889 0.02171679 0.1473662 0.99998
01
## 136    135   136    3426 0.02395728 0.1132904 0.02171331 0.1473544 0.99998
09
## 137    136   137    3441 0.02410160 0.1134743 0.02168419 0.1472555 0.99998
18
## 138    137   138    3459 0.02410160 0.1134270 0.02168696 0.1472649 0.99998
18
## 139    138   139    3477 0.02410160 0.1133888 0.02167106 0.1472109 0.99998
18
## 140    139   140    3499 0.02366864 0.1135619 0.02167653 0.1472295 0.99998
41
## 141    140   141    3520 0.02366864 0.1134631 0.02160580 0.1469891 0.99998
41
## 142    141   142    3541 0.02366864 0.1132069 0.02160808 0.1469969 0.99998
41
## 143    142   143    3557 0.02395728 0.1132941 0.02159744 0.1469607 0.99998
```

```
41
## 144    143   144    3578 0.02424592 0.1131506 0.02156345 0.1468450 0.99998
41
## 145    144   145    3593 0.02410160 0.1133653 0.02155900 0.1468298 0.99998
41
## 146    145   146    3608 0.02410160 0.1130494 0.02144378 0.1464369 0.99998
41
## 147    146   147    3621 0.02410160 0.1131867 0.02146724 0.1465170 0.99998
50
## 148    147   148    3636 0.02410160 0.1133475 0.02146887 0.1465226 0.99998
55
## 149    148   149    3654 0.02410160 0.1136613 0.02148860 0.1465899 0.99998
55
## 150    149   150    3667 0.02410160 0.1137400 0.02151608 0.1466836 0.99998
46
##
## $OutputCasTables
##            casLib      Name Rows Columns
## 1 CASUSER(alarzo) scored_gb 6929       6
##
## $ScoreInfo
##                          Descr                          Value
## 1 Number of Observations Read                           6929
## 2 Number of Observations Used                           6929
## 3 Misclassification Error (%)                     2.4101601963
```

```
## Assess and compare model
assessed <- cas.percentile.assess(conn,
                                  table = 'scored_gb',
                                  inputs = 'P_PAD_Target1',
                                  casout = list(name = 'assessed', replace =
TRUE),
                                  response = target,
                                  event = '1'
)

## We get 2 produced tables from the previous step that we can use to plot RO
C and LIFT
assessed[["OutputCasTables"]]
```

```
##            casLib         Name Rows Columns
## 1 CASUSER(alarzo)     assessed   20      21
## 2 CASUSER(alarzo) assessed_ROC  100      21
```

```
cas.table.columnInfo(conn,table="assessed")
```

```
## $ColumnInfo
##              Column                              Label ID   Type RawLengt
h
## 1          _Column_                  Analysis Variable  1   char        1
3
```

```
## 2           _Event_                                Event  2   char        1
2
## 3           _Depth_                                Depth  3 double
8
## 4           _Value_                                Value  4 double
8
## 5            _NObs_                Sum of Frequencies  5 double
8
## 6         _NEvents_                  Number of Events  6 double
8
## 7     _NEventsBest_              Best Number of Events  7 double
8
## 8            _Resp_                % Captured Response  8 double
8
## 9        _RespBest_           Best % Captured Response  9 double
8
## 10           _Lift_                             Lift 10 double
8
## 11       _LiftBest_                        Best Lift 11 double
8
## 12        _CumResp_      Cumulative % Captured Response 12 double
8
## 13    _CumRespBest_ Cumulative Best % Captured Response 13 double
8
## 14        _CumLift_                   Cumulative Lift 14 double
8
## 15    _CumLiftBest_               Best Cumulative Lift 15 double
8
## 16        _PctResp_                        % Response 16 double
8
## 17    _PctRespBest_                   Best % Response 17 double
8
## 18     _CumPctResp_              Cumulative % Response 18 double
8
## 19 _CumPctRespBest_          Cumulative Best % Response 19 double
8
## 20           _Gain_                               Gain 20 double
8
## 21       _GainBest_                          Best Gain 21 double
8
##    FormattedLength Format NFL NFD
## 1               13      $   0   0
## 2               12      $   0   0
## 3               12   BEST   0   0
## 4               12   BEST   0   0
## 5               12   BEST   0   0
## 6               12   BEST   0   0
## 7               12   BEST   0   0
## 8               12   BEST   0   0
## 9               12   BEST   0   0
```

```
## 10              12   BEST   0   0
## 11              12   BEST   0   0
## 12              12   BEST   0   0
## 13              12   BEST   0   0
## 14              12   BEST   0   0
## 15              12   BEST   0   0
## 16              12   BEST   0   0
## 17              12   BEST   0   0
## 18              12   BEST   0   0
## 19              12   BEST   0   0
## 20              12   BEST   0   0
## 21              12   BEST   0   0
```

```
cas.table.columnInfo(conn,table="assessed_ROC")
```

```
## $ColumnInfo
##           Column                     Label ID   Type RawLength FormattedLe
ngth
## 1       _Column_         Analysis Variable  1   char        13
13
## 2        _Event_                     Event  2   char        12
12
## 3       _Cutoff_                    Cutoff  3 double         8
12
## 4           _TP_              True Positive  4 double         8
12
## 5           _FP_             False Positive  5 double         8
12
## 6           _FN_             False Negative  6 double         8
12
## 7           _TN_              True Negative  7 double         8
12
## 8  _Sensitivity_                Sensitivity  8 double         8
12
## 9  _Specificity_                Specificity  9 double         8
12
## 10          _KS_                  KS Cutoff 10 double         8
12
## 11         _KS2_   Kolmogorov-Smirnov (KS) 11 double         8
12
## 12       _FHALF_                 F0.5 Score 12 double         8
12
## 13         _FPR_       False Positive Rate 13 double         8
12
## 14         _ACC_                   Accuracy 14 double         8
12
## 15         _FDR_      False Discovery Rate 15 double         8
12
## 16          _F1_                   F1 Score 16 double         8
12
```

```
## 17           _C_             Area Under ROC 17 double         8
12
## 18        _GINI_          Gini Coefficient 18 double         8
12
## 19       _GAMMA_                     Gamma 19 double         8
12
## 20         _TAU_                       Tau 20 double         8
12
## 21   _MiscEvent_ Misclassification (Event) 21 double         8
12
##    Format NFL NFD
## 1       $   0   0
## 2       $   0   0
## 3    BEST   0   0
## 4    BEST   0   0
## 5    BEST   0   0
## 6    BEST   0   0
## 7    BEST   0   0
## 8    BEST   0   0
## 9    BEST   0   0
## 10   BEST   0   0
## 11   BEST   0   0
## 12   BEST   0   0
## 13   BEST   0   0
## 14   BEST   0   0
## 15   BEST   0   0
## 16   BEST   0   0
## 17   BEST   0   0
## 18   BEST   0   0
## 19   BEST   0   0
## 20   BEST   0   0
## 21   BEST   0   0
```

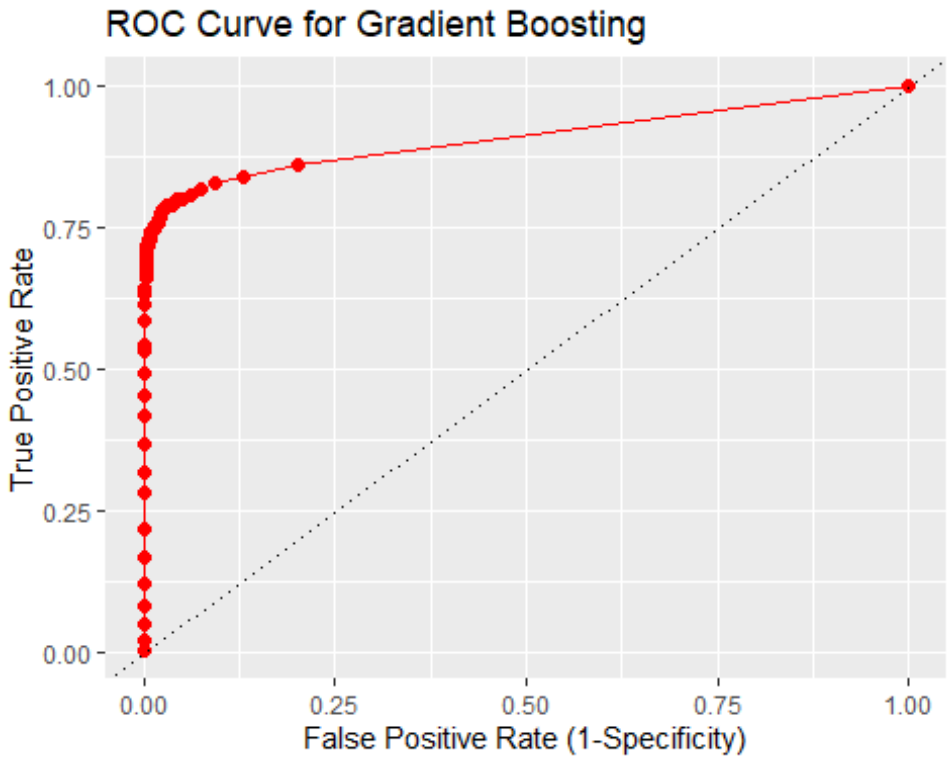
```
## Plot ROC-Curve
specify_ROC <- as.numeric(unlist(
  cas.table.fetch(conn,
                  table="assessed_ROC",
                  fetchVars="_Specificity_",
                  index=FALSE,
                  to=tablesize)
))
False_Positive_Rate <- 1-specify_ROC

sensitivity_ROC <- as.numeric(unlist(
  cas.table.fetch(conn,
                  table="assessed_ROC",
                  fetchVars="_Sensitivity_",
                  index=FALSE,
                  to=tablesize)
))
```

```
True_Positive_Rate <- sensitivity_ROC

df <- data.frame(False_Positive_Rate, True_Positive_Rate)

ROC_Curve <- ggplot(df, aes(x = False_Positive_Rate, y = True_Positive_Rate))
+
              geom_path(color='red') + geom_point(size = 2, color='red') +
              geom_abline(slope = 1,linetype="dotted") +
              labs(title="ROC Curve for Gradient Boosting", x="False Positive
Rate (1-Specificity)", y="True Positive Rate")
ROC_Curve
```



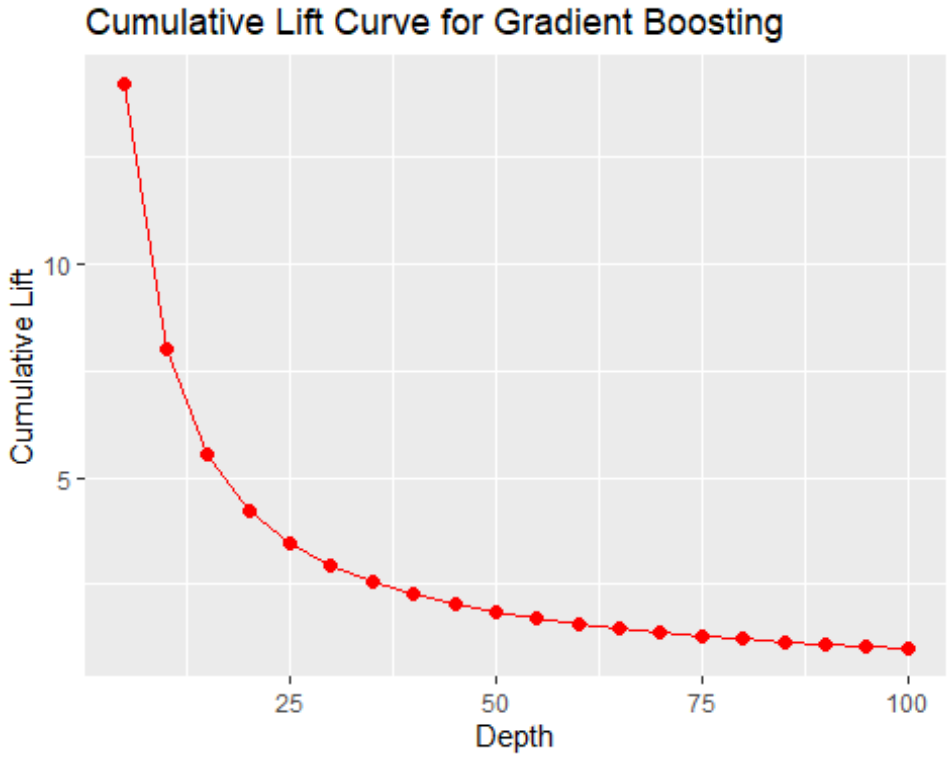

```
## Plot Cumulative Lift Curve
depth_LIFT <- as.numeric(unlist(
  cas.table.fetch(conn,
                  table="assessed",
                  fetchVars="_Depth_",
                  index=FALSE,
                  to=tablesize)
))

cumulative_LIFT <- as.numeric(unlist(
  cas.table.fetch(conn,
                  table="assessed",
                  fetchVars="_CumLift_",
                  index=FALSE,
```

```
                    to=tablesize)
))

df2 <- data.frame(depth_LIFT, cumulative_LIFT)

LIFT_Curve <- ggplot(df2, aes(x = depth_LIFT, y = cumulative_LIFT)) +
  geom_path(color='red') + geom_point(size = 2, color='red') +
  labs(title="Cumulative Lift Curve for Gradient Boosting", x="Depth", y="Cum
ulative Lift")
LIFT_Curve
```




```
cas.terminate(conn)

## [1] 0
```