Tree Based Modeling

Süleyman Buğra Gülsoy

06 01 2022

Introduction

The project aims to enlighten the factors that has an impact over diabetes on Pima Indians Classifying the people whether they have diabetes or not depending on the given parameters Will help us suspect the diabetes on future diseased people prior to diagnosis. Knowing which semptoms to look for in diabetes is going to be a major breakthorugh on Calculating the potential risk of a person's suffering from diabetes in the future Moreover, the project will help people the track their variable threshold levels that creates a potential risk

Methodology

For the research modeling method classification tree is going to be used The target variable is factor with 2 levels which makes tree based model suitable for design Pruning and tuning techniques is going to be used to modify the model to have better classification

Data Set

```
library("mlbench")

data(PimaIndiansDiabetes2)

head(PimaIndiansDiabetes2,10)

##    pregnant glucose pressure triceps insulin mass pedigree age diabetes
## 1         6     148       72      35      NA 33.6    0.627  50      pos
## 2         1      85       66      29      NA 26.6    0.351  31      neg
## 3         8     183       64      NA      NA 23.3    0.672  32      pos
## 4         1      89       66      23      94 28.1    0.167  21      neg
## 5         0     137       40      35     168 43.1    2.288  33      pos
## 6         5     116       74      NA      NA 25.6    0.201  30      neg
## 7         3      78       50      32      88 31.0    0.248  26      pos
## 8        10     115       NA      NA      NA 35.3    0.134  29      neg
## 9         2     197       70      45     543 30.5    0.158  53      pos
## 10        8     125       96      NA      NA   NA    0.232  54      pos
```

The data is about observed parameters of Pima Indian people that tested for diabetes and their diabet results The aim of the data set is explaining the diabetes diagnosis results with given parameters

The data is avaliable in “mlbench” package in CRAN

The data set “PimaIndiansDiabetes2” consists 768 observations with 9 total variables

8 of which is numeric regressor variables and 1 categorical target variable

pregnant : Number of times pregnant glucose : Plasma glucose concentration (glucose tolerance test) pressure : Diastolic blood pressure (mm Hg) triceps : Triceps skin fold thickness (mm) insulin : 2-Hour serum insulin (mu U/ml) mass : Body mass index (weight in kg/(height in m)^2) pedigree : Diabetes pedigree function age : Age (years) diabetes : Class variable (test for diabetes)

Model Fitting

```
library("car")

## Zorunlu paket yükleniyor: carData

library("ggplot2")
library("leaps")
library("gbm")

## Loaded gbm 2.1.8

library("tree")
library("caret")

## Zorunlu paket yükleniyor: lattice

set.seed (58270)

nacount = c()

for(i in 1:ncol(PimaIndiansDiabetes2)){

  nacount = c(nacount,sum(is.na(PimaIndiansDiabetes2[,i])))
}

nacount

## [1]   0   5  35 227 374  11   0   0   0
```

The variable “insulin” has half of its observarions as NA Hence, it is better to Remove the variable from data Also, the variable “triceps” has %30 of its observations as NA so removing the rows that has NA Would damage the data instead NA values of “triceps” will be replaced with its mean value Than the NA rows can be deducted from data

```
PimaIndiansDiabetes2[,"insulin"] = NULL


mean(na.omit(PimaIndiansDiabetes2$triceps))

## [1] 29.15342

PimaIndiansDiabetes2$triceps[is.na(PimaIndiansDiabetes2$triceps)] = 29.15342


PimaIndiansDiabetes2 = na.omit(PimaIndiansDiabetes2)

sum(duplicated(PimaIndiansDiabetes2))

## [1] 0
```

There does not have any duplicated values

```
cor(PimaIndiansDiabetes2[,1:7])

##              pregnant   glucose      pressure   triceps       mass
pedigree
## pregnant  1.00000000 0.1349149  2.096681e-01 0.0800123 0.01234162 -
2.599607e-02
## glucose   0.13491495 1.0000000  2.233312e-01 0.1956949 0.22327644
1.366297e-01
## pressure  0.20966808 0.2233312  1.000000e+00 0.1924827 0.28740346 -
7.527687e-05
## triceps   0.08001230 0.1956949  1.924827e-01 1.0000000 0.55435104
1.056122e-01
## mass      0.01234162 0.2232764  2.874035e-01 0.5543510 1.00000000
1.548582e-01
## pedigree -0.02599607 0.1366297 -7.527687e-05 0.1056122 0.15485819
1.000000e+00
## age       0.55706615 0.2635602  3.248975e-01 0.1261195 0.02083534
2.309810e-02
##                 age
## pregnant 0.55706615
## glucose  0.26356023
## pressure 0.32489747
## triceps  0.12611948
## mass     0.02083534
## pedigree 0.02309810
## age      1.00000000
```

Since all the values of corrolatin matirx is below 0.8 it can be concluded that data does not have

Multicollinearity problem

```
str(PimaIndiansDiabetes2)
```

```
## 'data.frame':    724 obs. of  8 variables:
##  $ pregnant: num  6 1 8 1 0 5 3 2 4 10 ...
##  $ glucose : num  148 85 183 89 137 116 78 197 110 168 ...
##  $ pressure: num  72 66 64 66 40 74 50 70 92 74 ...
##  $ triceps : num  35 29 29.2 23 35 ...
##  $ mass    : num  33.6 26.6 23.3 28.1 43.1 25.6 31 30.5 37.6 38 ...
##  $ pedigree: num  0.627 0.351 0.672 0.167 2.288 ...
##  $ age     : num  50 31 32 21 33 30 26 53 30 34 ...
##  $ diabetes: Factor w/ 2 levels "neg","pos": 2 1 2 1 2 1 2 2 1 2 ...
##  - attr(*, "na.action")= 'omit' Named int [1:44] 8 10 16 50 61 76 79 82
146 173 ...
##   ..- attr(*, "names")= chr [1:44] "8" "10" "16" "50" ...
```

All the variables are in the correct form for anlysis

```
levels(PimaIndiansDiabetes2$diabetes) = c(0,1)
split_data = sample(1:nrow(PimaIndiansDiabetes2),
0.8*nrow(PimaIndiansDiabetes2))

train_set = (PimaIndiansDiabetes2)[split_data,]

test_set = (PimaIndiansDiabetes2)[-split_data,]
```

The data has spllited into test and train sets with respect to %80, %20 rule

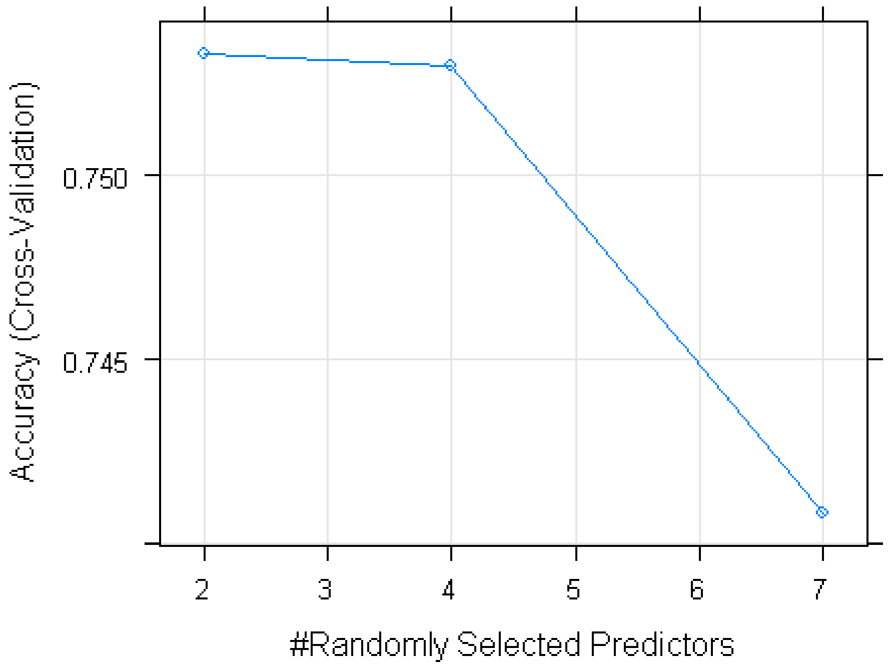
```
tree_model = train(diabetes~., data = train_set, method = "rf",
                   trControl = trainControl("cv", number = 10), ntree = 1000)


summary(tree_model)

##                 Length Class      Mode
## call               5   -none-     call
## type               1   -none-     character
## predicted        579   factor     numeric
## err.rate        3000   -none-     numeric
## confusion          6   -none-     numeric
## votes           1158   matrix     numeric
## oob.times        579   -none-     numeric
## classes            2   -none-     character
## importance         7   -none-     numeric
## importanceSD       0   -none-     NULL
## localImportance    0   -none-     NULL
## proximity          0   -none-     NULL
## ntree              1   -none-     numeric
## mtry               1   -none-     numeric
## forest            14   -none-     list
## y                579   factor     numeric
## test               0   -none-     NULL
## inbag              0   -none-     NULL
## xNames             7   -none-     character
## problemType         1    -none-     character
## tuneValue           1    data.frame list
## obsLevels           2    -none-     character
## param               1    -none-     list
```

Training error rate is obtained as 3000

```
plot(tree_model)
```



As it seen 4 number of predictors has the highest accracy thus model with 4 variables should be used

```
predict_tree = predict(tree_model,test_set)

table(Predicted = predict_tree, Actual = test_set$diabetes)

##          Actual
## Predicted  0  1
##         0 88 13
##         1 13 31
```

The model predicted correctly 119 of 145 observation

And it miss predicted the 26 of 145 observations

0.8068966 is accuracy for the model

0.1931034 is the error rate

```
tree_model2 = train(diabetes~., data = train_set, method = "rf",
                   trControl = trainControl("cv", number = 10), ntree =
1000,maxdepth = 4)


predict_tree2 = predict(tree_model2,test_set)


table(Predicted = predict_tree2, Actual = test_set$diabetes)
```

```
##           Actual
## Predicted  0  1
##         0 87 13
##         1 14 31
```

The pruned model failed to classify 27 of 145 observations The error rate is 0.1862069

Conclusion

The pruned the with the optimal number of depth Did a better job at classifying the diabetes With respect to decrease at error rate of 0.0068965 points Both models seems to work and classify at decent level Thus pruned tree with depth 4 having slight edge