R Project

Zhaosen Guo

5/15/2019

Project: Heart Disease Study

Introducation

The project is inspired by a database that collected heart disease cases from Cleveland, Hungary, Switzerland, and the VA Long Beach. The original database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. This project will be using the Cleveland dataset and the 14 attributes in them. By running logistic regression model, classification tree, and the random forest function, I would like to explore the relationship between the 13 predictor variables and the response, which is a qualitative variable that indicates whether the patient has heart diseases or not. The model-building process is going to constantly check the training set error and the testing set error. In the end, I will try to find the best predictors based on those three methods, then present the "best-avaliable" model to the readers.

Source: Heart Disease Data Set from UCI, https://archive.ics.uci.edu/ml/datasets/Heart+Disease

Data pre-processing, cleaning, and variable renaming are detailed in the Appendix section.

The original datafram is cleaned and splitted into a testing set of 236 entries and a training set of 60 entries.

This dataset includes 303 instance and 14 variables:

1. age (in years)
2. sex (1 = male; 0 = female)
3. cp: chest pain type (1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic)
4. trestbps: resting blood pressure (in mm Hg on admission to the hospital)
5. chol: serum cholesterol in mg/dl
6. fbs: (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
7. restecg: resting electrocardiographic results (0: normal, 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), 2: showing probable or definite left ventricular hypertrophy by Estes' criteria)
8. thalach: maximum heart rate achieved
9. exang: exercise induced angina (1 = yes; 0 = no)
10. oldpeak = ST depression induced by exercise relative to rest
11. slope: the slope of the peak exercise ST segment (1: upsloping, 2: flat, 3: downsloping)
12. ca: number of major vessels (0-3) colored by fluoroscopy, 4 = na
13. thal: (thaldur) duration of exercise test in minutes; 1 = fixed defect; 2 = normal; 3 = reversable defect; 0 = na
14. num: diagnosis of heart disease (angiographic disease status) (0: < 50% diameter narrowing, 1: > 50% diameter narrowing)

Something to keep in mind:

I have looked at all the approximate pricing for the variables that are involved in this data set, and here is a list: (Information gathered from CVS.com and healthcarebluebook.com, estimating for patients in ZIP 13323, NY) High blood pressure evaluation: $100 Cholesterol screenings: $60 Blood sugar test: $20 ECG (electrocardiogram): $45 Cardiac Exercise Stress Test (thalach/exang/thal/ST depression): $170 Fluoroscopy: $120

Logistic Regression

Let's say for this log-regression model we want to only involves variables that are on Q&A basis with doctor and a sum of test that cost around $100. We will have 3 possible combination with testings: blood pressure(trestbps)/ cholesterol(chol) + blood sugar (fbs) / cholesterol(chol) + ECG(restecg).

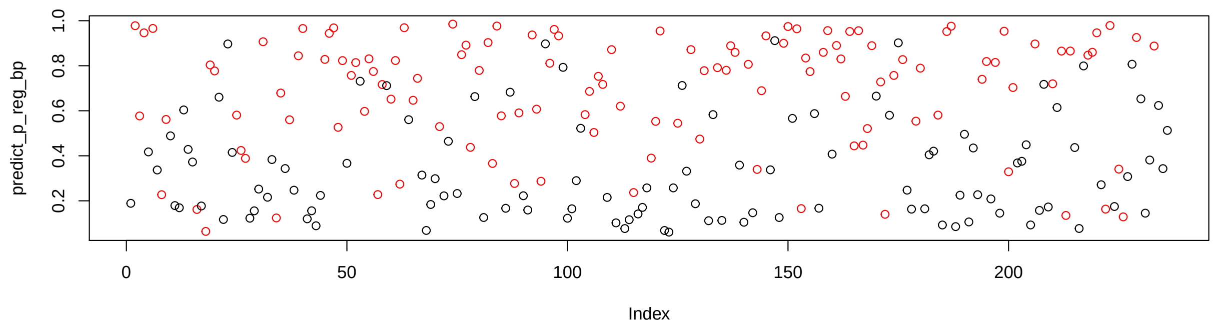
Blood Pressure

```
reg_bp = glm( result ~ age + sex_ + cp_ + trestbps, family = 'binomial', data = heart_train)
summary(reg_bp)
```

```
##
## Call:
## glm(formula = result ~ age + sex_ + cp_ + trestbps, family = "binomial",
##     data = heart_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2019  -0.7535   0.2401   0.7232   2.3434
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       5.806992   1.530490   3.794 0.000148 ***
## age              -0.055048   0.019493  -2.824 0.004744 **
## sex_male         -1.782336   0.384891  -4.631 3.64e-06 ***
## cp_atypical       2.157265   0.478237   4.511 6.46e-06 ***
## cp_non-anginal    2.318072   0.407357   5.691 1.27e-08 ***
## cp_asymptomatic   2.536845   0.673191   3.768 0.000164 ***
## trestbps         -0.019193   0.009386  -2.045 0.040865 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 326.56  on 235  degrees of freedom
## Residual deviance: 231.73  on 229  degrees of freedom
## AIC: 245.73
##
## Number of Fisher Scoring iterations: 5
```

Training Errors:

```r
predict_p_reg_bp = predict(reg_bp, type ='response')
plot(predict_p_reg_bp , col = heart_train$result)
```

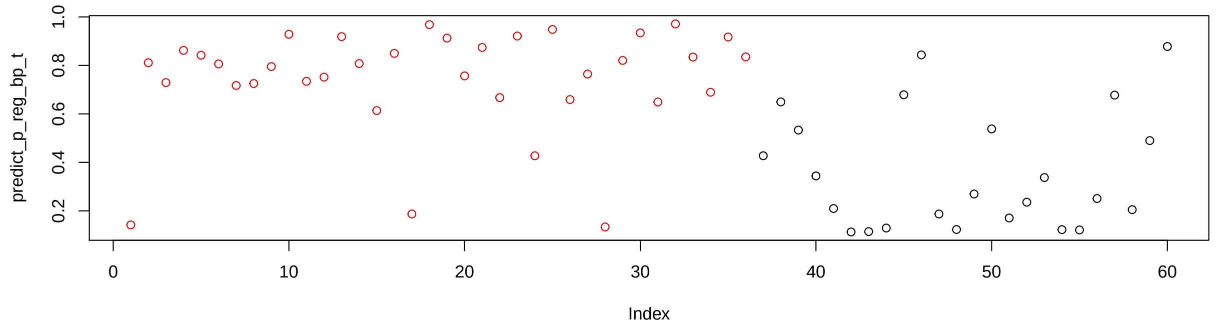


```r
predict_stat_reg_bp = ifelse(predict_p_reg_bp > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_train$result , predicted = predict_stat_reg_bp)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             86            26
##   diagnosis_Yes            25            99
```

And for Testing Errors:

```r
predict_p_reg_bp_t = predict(reg_bp , type ='response', newdata = heart_test)
plot(predict_p_reg_bp_t , col = heart_test$result)
```



```r
predict_stat_reg_bp_t = ifelse(predict_p_reg_bp_t > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_test$result , predicted = predict_stat_reg_bp_t)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             17             7
##   diagnosis_Yes             4            32
```

From all above, we can draw inferences about trestbps (resting blood pressure) variable when adjusted for age, sex, and cp (chest pain type) that it has a p-value of 0.040865, which makes it not that reliable comparing to the other listed variables, yet it is still meaningful addition to my model because changes in this predictor's (trestbps) value are fairly related to changes in the response (result). In this case our model on test variables generated 7 Type I Error and 4 Type II Error.

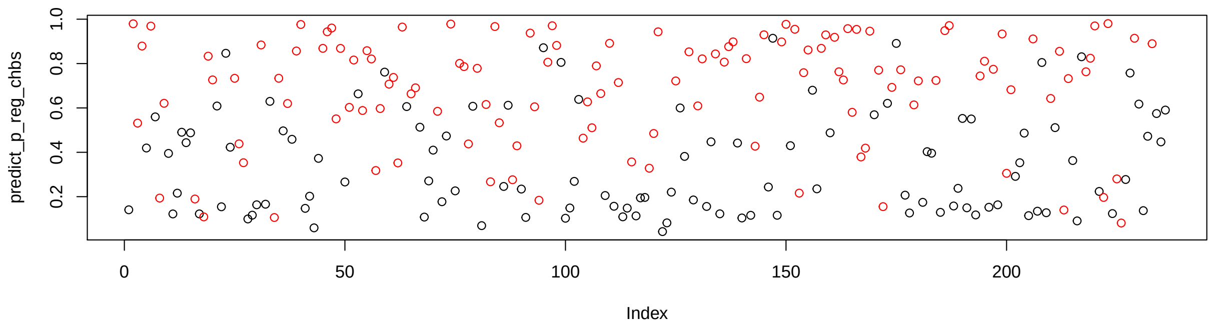
Cholesterol Screening and Blood Sugar Test

```
reg_chbs = glm( result ~ age + sex_ + cp_ + chol + fbs_, family = 'binomial', data = heart_train)
summary(reg_chbs)
```

```
##
## Call:
## glm(formula = result ~ age + sex_ + cp_ + chol + fbs_, family = "binomial",
##     data = heart_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2157  -0.7335   0.2431   0.7870   2.2410
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       4.791264   1.383330   3.464 0.000533 ***
## age              -0.061802   0.019181  -3.222 0.001273 **
## sex_male         -1.813651   0.389578  -4.655 3.23e-06 ***
## cp_atypical       2.131719   0.471962   4.517 6.28e-06 ***
## cp_non-anginal    2.294932   0.407238   5.635 1.75e-08 ***
## cp_asymptomatic   2.297608   0.651374   3.527 0.000420 ***
## chol             -0.004388   0.003174  -1.382 0.166839
## fbs_true         -0.232768   0.429024  -0.543 0.587438
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 326.56  on 235  degrees of freedom
## Residual deviance: 233.97  on 228  degrees of freedom
## AIC: 249.97
##
## Number of Fisher Scoring iterations: 5
```

The p-values are not looking so good… but let's check out the training errors for this one:

```
predict_p_reg_chbs = predict(reg_chbs, type ='response')
plot(predict_p_reg_chbs , col = heart_train$result)
```

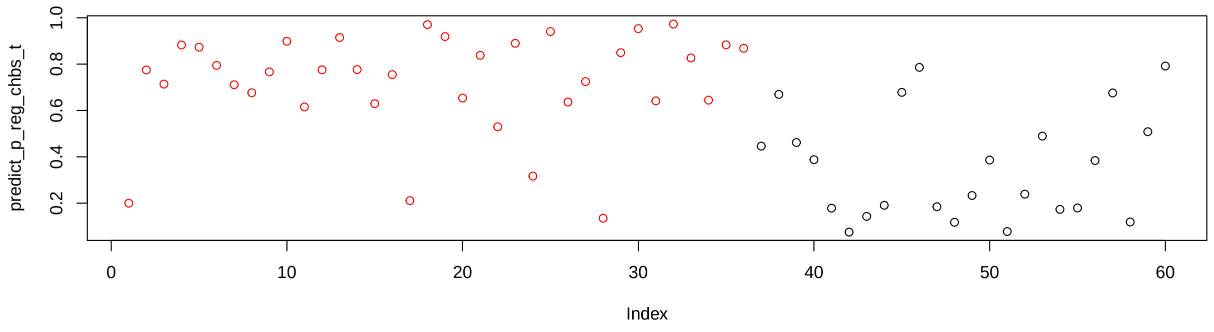


```r
predict_stat_reg_chbs = ifelse(predict_p_reg_chbs > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_train$result , predicted = predict_stat_reg_chbs)
```

```
##                  predicted
## actual            diagnosis_No diagnosis_Yes
##   diagnosis_No              84            28
##   diagnosis_Yes             27            97
```

And for Testing Errors:

```r
predict_p_reg_chbs_t = predict(reg_chbs , type ='response', newdata = heart_test)
plot(predict_p_reg_chbs_t , col = heart_test$result)
```



```r
predict_stat_reg_chbs_t = ifelse(predict_p_reg_chbs_t > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_test$result , predicted = predict_stat_reg_chbs_t)
```

```
##                  predicted
## actual            diagnosis_No diagnosis_Yes
##   diagnosis_No              18             6
##   diagnosis_Yes              4            32
```

This is interesting because though we have two variables that two very high p-value that indicates it has little correlation to the result of diagnosis, yet the testing sample produced 1 less Type I error and the same Type

II error? One explanation I could think of is due to the fact that two predictor variables with one qualitative and one categorical, somehow made the model more complicated and contributed to the overall accuracy. Or, it could be an unfortunate result due to the testing samples I generated are not that representative. Let's continue to another model with another combination of two testing.

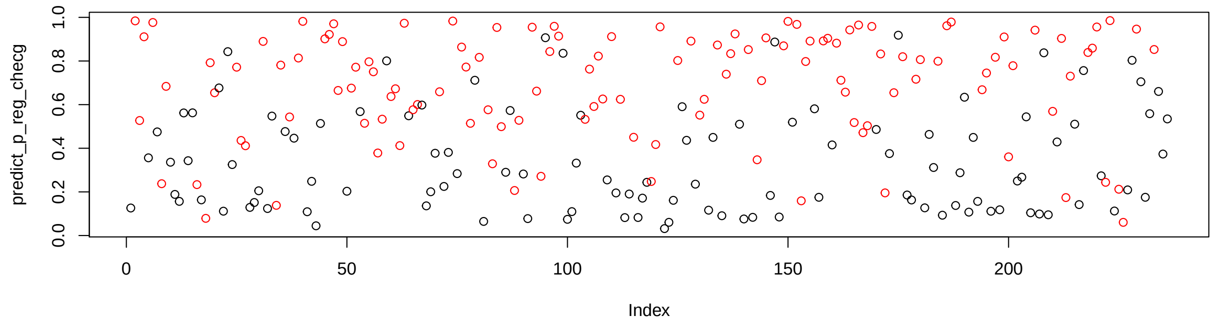
Cholesterol Screening and ECG:

```r
reg_checg = glm( result ~ age + sex_ + cp_ + chol + restecg_, family = 'binomial', data = heart_train)
summary(reg_checg)
```

```
##
## Call:
## glm(formula = result ~ age + sex_ + cp_ + chol + restecg_, family = "binomial",
##     data = heart_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2376  -0.7563   0.2133   0.7600   2.3698
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)          4.287041   1.431947   2.994 0.002755 **
## age                 -0.060537   0.019327  -3.132 0.001735 **
## sex_male            -1.870194   0.399479  -4.682 2.85e-06 ***
## cp_atypical          2.136216   0.479867   4.452 8.52e-06 ***
## cp_non-anginal       2.256655   0.405190   5.569 2.56e-08 ***
## cp_asymptomatic      2.406260   0.667052   3.607 0.000309 ***
## chol                -0.003865   0.003244  -1.191 0.233529
## restecg_stt          0.644031   0.335885   1.917 0.055185 .
## restecg_hypertrophy -0.673372   1.530597  -0.440 0.659980
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 326.56  on 235  degrees of freedom
## Residual deviance: 230.08  on 227  degrees of freedom
## AIC: 248.08
##
## Number of Fisher Scoring iterations: 5
```

This time the p-value of chol level increase very slightly, and we see that the rest ECG that has a ST-T wave change are having a relevant p-value of 0.55. Training Errors:

```r
predict_p_reg_checg = predict(reg_checg, type ='response')
plot(predict_p_reg_checg , col = heart_train$result)
```

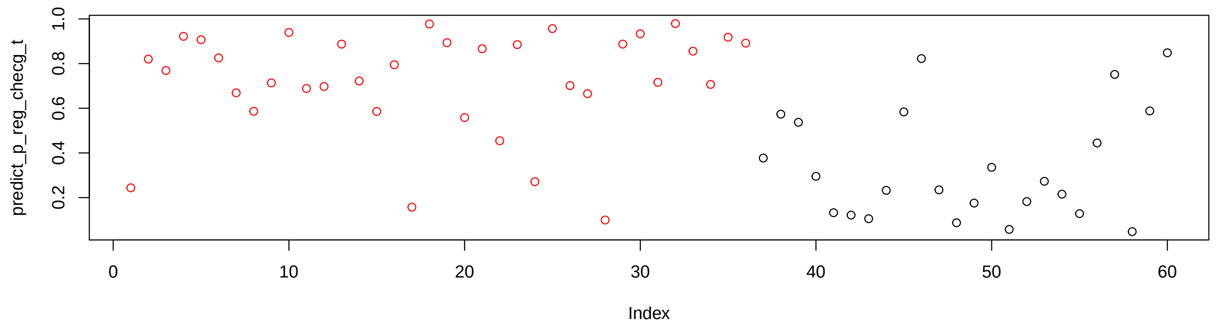


```r
predict_stat_reg_checg = ifelse(predict_p_reg_checg > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_train$result , predicted = predict_stat_reg_checg)
```

```
##                  predicted
## actual             diagnosis_No diagnosis_Yes
##    diagnosis_No              81            31
##    diagnosis_Yes             24           100
```

Testing Errors:

```r
predict_p_reg_checg_t = predict(reg_checg , type ='response', newdata = heart_test)
plot(predict_p_reg_checg_t , col = heart_test$result)
```



```r
predict_stat_reg_checg_t = ifelse(predict_p_reg_checg_t > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_test$result , predicted = predict_stat_reg_checg_t)
```

```
##                  predicted
## actual             diagnosis_No diagnosis_Yes
##    diagnosis_No              17             7
##    diagnosis_Yes              5            31
```

So this model actually has the least accuracy comparing to the other under-100-dollar combinations! Wow! So we can tell from our "messing-around" with predictors that this is not a "the more the merrier" situation, sometimes introducing more predictor variables can actually make the model worse.

The Expensive Model

Assume this time we are one of the privileged few who can afford some facy testing. We will add all the variables relating to Cardiac Exercise Stress Test (EST) and Fluoroscopy (variable: ca) into our model, and just look at the misclassification table because we are rich and plots takes too long to read and that's a waste of time aka $ LOL.

```
reg_rich = glm( result ~ age + sex_ + thalach +
                  exang_ + oldpeak + slope_ + ca + thal_, family = 'binomial', data = heart_train)
summary(reg_rich)
```

```
##
## Call:
## glm(formula = result ~ age + sex_ + thalach + exang_ + oldpeak +
##     slope_ + ca + thal_, family = "binomial", data = heart_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0488  -0.3976   0.1577   0.5085   3.0126
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.26642    2.69902  -0.469  0.63892
## age                0.02042    0.02397   0.852  0.39424
## sex_male          -1.10374    0.50134  -2.202  0.02769 *
## thalach            0.02056    0.01144   1.798  0.07225 .
## exang_yes         -1.44421    0.47436  -3.045  0.00233 **
## oldpeak           -0.32109    0.24267  -1.323  0.18579
## slope_flat        -0.43665    0.87860  -0.497  0.61920
## slope_downsloping  0.71330    1.00467   0.710  0.47772
## ca1               -2.41467    0.54390  -4.440 9.02e-06 ***
## ca2               -2.95891    0.75108  -3.940 8.16e-05 ***
## ca3               -2.36943    0.83997  -2.821  0.00479 **
## thal_normal        0.47038    0.87632   0.537  0.59143
## thal_reversable   -1.26328    0.86278  -1.464  0.14314
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 326.56  on 235  degrees of freedom
## Residual deviance: 158.83  on 223  degrees of freedom
## AIC: 184.83
##
## Number of Fisher Scoring iterations: 6
```

In this case, we only 1 variable from the EST are influential, along with fluoroscopy results, which has some good-looking p-values. Let's check out the training errors:

```
predict_p_reg_rich = predict(reg_rich, type ='response')
predict_stat_reg_rich = ifelse(predict_p_reg_rich > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_train$result , predicted = predict_stat_reg_rich)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             92            20
##   diagnosis_Yes            17           107
```

And for Testing Errors:

```r
predict_p_reg_rich_t = predict(reg_rich , type ='response', newdata = heart_test)
predict_stat_reg_rich_t = ifelse(predict_p_reg_rich_t > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_test$result , predicted = predict_stat_reg_rich_t)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             18             6
##   diagnosis_Yes             6            30
```

So it seems like that the overall misclassifications counts remained the same, comparing to the best performing under-100-Dollar model, however, the Type II misclassifications (the patient who has heart diseases but predicted as healthy) actually increased. And that is contrary to our primary purpose of this modeling, thus, from this data set, I can say that even the costly models cannot take care of the business.
In addition, that's probably why doctors took many years to train to be able to be in their positions, and it boils down to combining all the testing and make a judgement, which a logistic model cannot even handle.

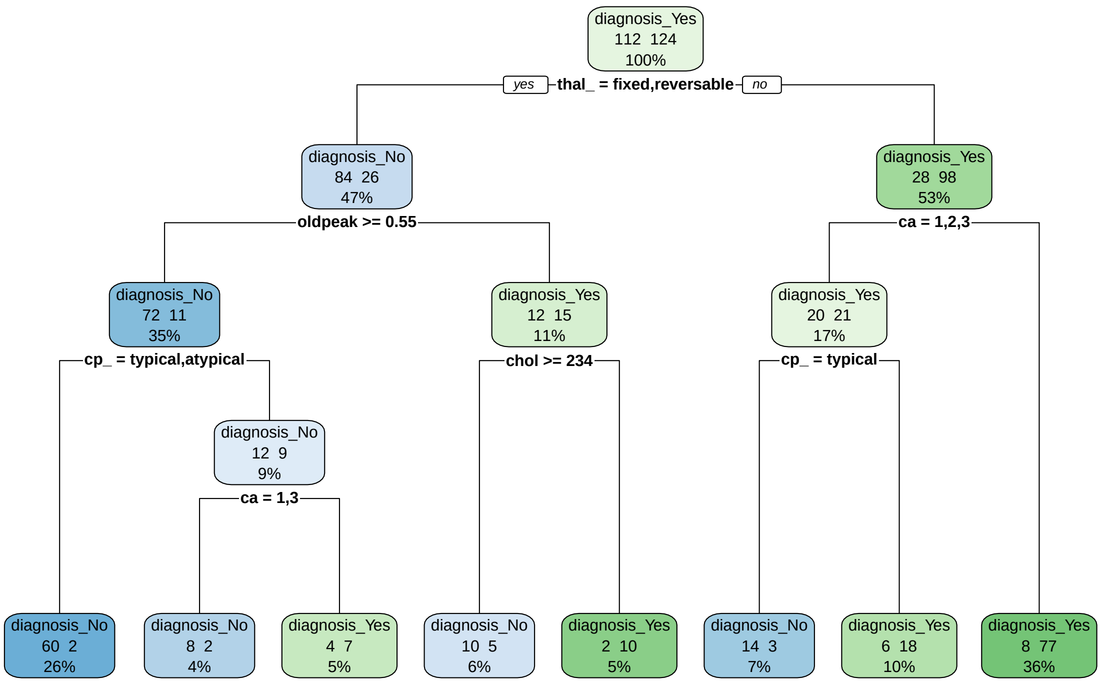
Classification Tree

Based on "Machine Learning"

With a classification tree, we will be able to check every individual variable's ability to best predict the response variable, and see the final ones picked by classification tree algorithm.

```r
library(rpart)
library(rpart.plot)
```

```r
tree_allv = rpart(result ~., method = 'class', data = heart_train)
rpart.plot(tree_allv, extra = 101)
```



Now with this tree model that picked out of all available variables, here I check the training error:

```
predicted_status_tree_allv = predict(tree_allv, type ='class')
table(actual = heart_train$result , predicted = predicted_status_tree_allv)
```

```
##                  predicted
## actual            diagnosis_No diagnosis_Yes
##   diagnosis_No              92            20
##   diagnosis_Yes             12           112
```

Not looking bad at all, now check the the testing result:

```
predicted_status_tree_allv_t = predict(tree_allv, type='class', newdata = heart_test)
table(actual = heart_test$result , predicted = predicted_status_tree_allv_t)
```

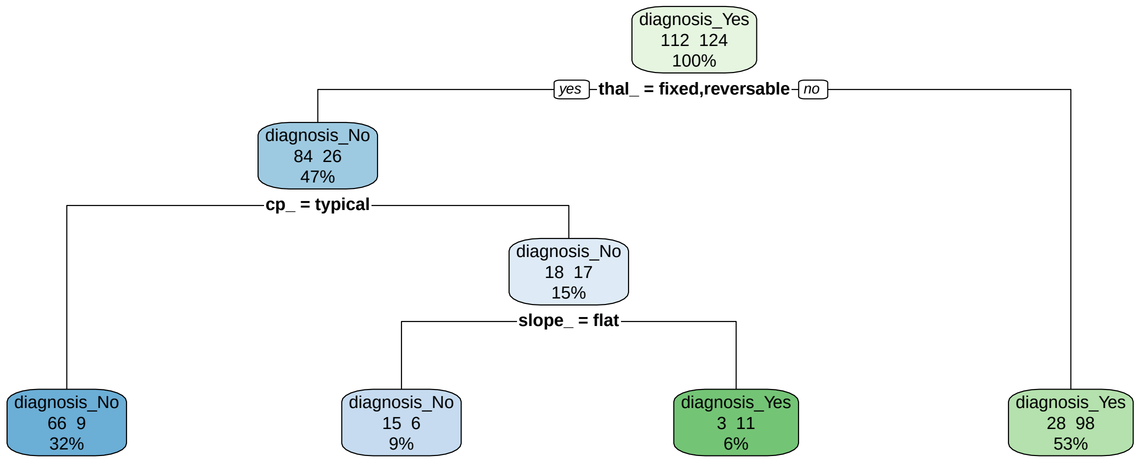
```
##                  predicted
## actual            diagnosis_No diagnosis_Yes
##   diagnosis_No              15             9
##   diagnosis_Yes              1            35
```

Good lord! With the computer exposed to all available variables, it picked the closest one to perfection (at least for this data set), and we only have 1 Type II Error here, which is close to complete the goal of identifying all heart-disease patients! It is also better than any given logistic model above as well, so let me try if human and beat the machine by testing and make my best manual tree!

Based “Human Learning”

After a lof of manual testing, this is the best model that I have:

```
tree_manual = rpart(result ~ cp_ + slope_ + thal_, cp = 0.009,  method = 'class', data = heart_train)
rpart.plot(tree_manual, extra = 101)
```



Training Error:

```
predicted_status_tree_manual = predict(tree_manual, type ='class')
table(actual = heart_train$result , predicted = predicted_status_tree_manual)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             81            31
##   diagnosis_Yes            15           109
```

Testing Error:

```
predicted_status_tree_manual_t = predict(tree_manual, type='class', newdata = heart_test)
table(actual = heart_test$result , predicted = predicted_status_tree_manual_t)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             15             9
##   diagnosis_Yes             4            32
```

In the end, this is the best that I could get down to 4 Type II errors and 9 Type I errors, and to be honest, the model I have is not that much worse from the machine “learned” model; meanwhile, interestingly enough, the three variables used were from 1 EST test and a simple question to figure out what type of chest pain a potential patient has. This finding further consolidates my thought about being expensive and use out all the testing result might not be the most helpful idea. Also, age was a deciding factor before, but with each variable being picked individually, it’s importance (aka correlation with response variable) seems to fall short in these tree models, showcasing the difference underlying two algorithms between a log. regression and a class. tree.

However, there should also be consideration for two issues about implementing classification tree models, though they looks good: first, due to the limited sizes of the data set (less than 300), there might not be enough training/testing splits; and second, we have 13 variables to deal with and thus there are lots of combinations of variables that could easily be overlooked or overrated. Thankfully, there's random forest function that can be used

---

Random Forest

Model

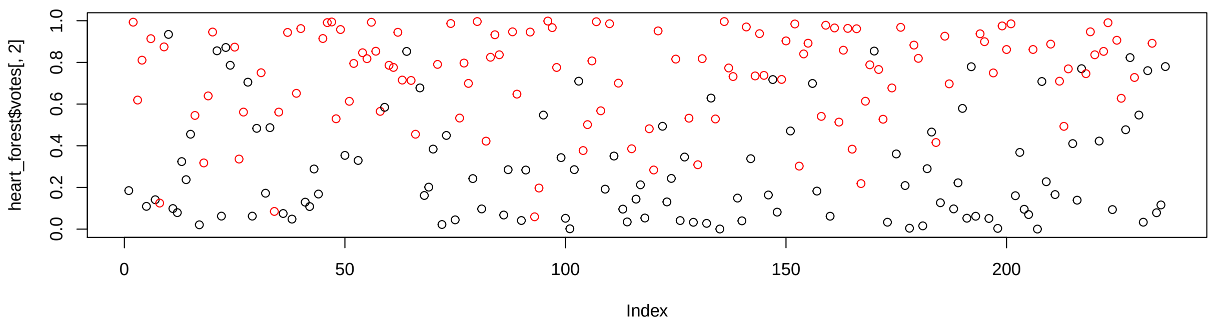
```
library(randomForest)
```

```
heart_forest = randomForest(result ~ ., ntree = 10000, mtry = 6, importance=TRUE, proximity=TRUE, data =
heart_forest$confusion
```

```
##               diagnosis_No diagnosis_Yes class.error
## diagnosis_No            90            22   0.1964286
## diagnosis_Yes           18           106   0.1451613
```

Above is the training error. The class error for Type II error (should be Yes but predicted as no disease) are around 15%. I did not choose a too large "mtry" value (less than half of the total predictors) because we do not want to run into an influential/dominant variable that overshadows other possibilities for branches. Additional graph:

```
#Previous line: "heart_forest$votes""  was hiden in efforts to eliminate the output
plot(heart_forest$votes[,2], col =  heart_train$result)
```



Testing error:

```
predicted_forest_t = predict(heart_forest , type='class' , newdata = heart_test)
table(actual = heart_test$result , predicted = predicted_forest_t)
```

```
##                  predicted
## actual            diagnosis_No diagnosis_Yes
##   diagnosis_No              18             6
##   diagnosis_Yes              1            35
```

Indeed, the lack of data entries and dominant variables were overshadowing the other potential predictions! This random forest model, comparing to the previously classification-tree-build model, achieve the same number (only 1) misclassification for missing out on an actual patient; meantime, also reduced the misdiagnosis for a healthy person by 3 within the testing set! Let's check it out and see what are some of the "star variables" in the forest:

```
importance(heart_forest)
```

```
##          diagnosis_No diagnosis_Yes MeanDecreaseAccuracy MeanDecreaseGini
## age          7.825098    19.3627300            19.788750        8.3810291
## sex_        23.504789    35.0556511            41.466796        2.9852591
## cp_         58.722959    40.1962647            67.809115       14.1790416
## trestbps     6.538803     0.9053817             4.872490        6.7064198
## chol       -14.727917    -2.7475136           -11.511569        8.1265898
## fbs_        -7.348430     3.1989463            -2.441474        0.8409917
## restecg_     7.106569     5.1391743             8.557324        2.0727941
## thalach     25.430435    35.8379368            43.596968       12.5919951
## exang_      20.473370    27.0806621            32.952747        5.8005517
## oldpeak     65.285009    43.5541705            77.683151       13.0256519
## slope_      48.211053    10.4422462            43.492951        5.4540281
## ca          76.941212    91.1049020           110.967448       16.8899610
## thal_       73.608345    79.5043171           101.844879       20.1041499
```

From here, we can see the fluoroscopy (ca), EST (thal/oldpeak), and Chest-pain (cp) are the overall best predictors for the result. These variables are most consistent with the variables that were selected during the classification tree process by "machine". In conclusion, if I would have to choose three most efficient predictor variables for this heart disease, they would be "ca", "thal", and "cp" based on all the model that I explored.

Rebuild a logisitic model based on aforementioned variables

```
reg_machine = glm(result ~ ca + thal_ + cp_ + oldpeak, family = 'binomial', data = heart_train)
summary(reg_machine)
```

```
##
## Call:
## glm(formula = result ~ ca + thal_ + cp_ + oldpeak, family = "binomial",
##     data = heart_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7833  -0.4100   0.1498   0.5116   2.3754
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)       0.8092     0.8034   1.007 0.313825
## ca1              -2.5418     0.5057  -5.026 5.01e-07 ***
## ca2              -1.7491     0.6710  -2.607 0.009147 **
## ca3              -2.6468     0.8963  -2.953 0.003147 **
## thal_normal       1.2593     0.7425   1.696 0.089903 .
## thal_reversable  -0.8294     0.7723  -1.074 0.282862
## cp_atypical       0.8893     0.5520   1.611 0.107126
## cp_non-anginal    2.4160     0.5379   4.492 7.06e-06 ***
## cp_asymptomatic   1.7839     0.7488   2.382 0.017204 *
## oldpeak          -0.9283     0.2393  -3.880 0.000105 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 326.56  on 235  degrees of freedom
## Residual deviance: 163.98  on 226  degrees of freedom
## AIC: 183.98
##
## Number of Fisher Scoring iterations: 6
```

This time the training errors look like this:

```
predict_p_reg_machine = predict(reg_machine, type ='response')
predict_stat_reg_machine = ifelse(predict_p_reg_machine > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_train$result , predicted = predict_stat_reg_machine)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             93            19
##   diagnosis_Yes            17           107
```

And the testing errors are:

```
predict_p_reg_machine_t = predict(reg_machine , type ='response', newdata = heart_test)
predict_stat_reg_machine_t = ifelse(predict_p_reg_machine_t > 0.5, 'diagnosis_Yes', 'diagnosis_No')
table(actual = heart_test$result , predicted = predict_stat_reg_machine_t)
```

```
##                  predicted
## actual           diagnosis_No diagnosis_Yes
##   diagnosis_No             19             5
##   diagnosis_Yes             2            34
```

And it seems like this logistic regression model has been the best-performing log-model in the end. Its total error count ties with the RandomForest at 7, but it has one more missed-diagnosis case and one less misdiagnosed case in the testing set. This result does confirm again for the previous claim of the high correlations between selected predictors and the repose variable.

Appendix

```
library(mosaic)
# Importing the data:
heartdata = read.csv('~/Desktop/heart.csv')
```

```r
# First of all, to check the integrity of the data:
table(!is.na(heartdata))
# The dataframe has 303 * 14, meaning we do not have appearent NA values.
# However, when looking at the description for variables, we see that some columns has NA entries store
# for column "thal"
heartdata = heartdata[-c(282),]
# for column "ca"
heartdata = heartdata[-c(252),]
heartdata = heartdata[-c(165),]
heartdata = heartdata[-c(164),]
heartdata = heartdata[-c(159),]
heartdata = heartdata[-c(93),]
# for column "thal" again
heartdata = heartdata[-c(49),]
#Changing variables accordingly.
heartdata$sex = factor(heartdata$sex)
levels(heartdata$sex) = c("female", "male")

heartdata$cp = factor(heartdata$cp)
levels(heartdata$cp) = c("typical", "atypical", "non-anginal", "asymptomatic")

heartdata$fbs = factor(heartdata$fbs)
levels(heartdata$fbs) = c("false", "true")

heartdata$restecg = factor(heartdata$restecg)
levels(heartdata$restecg) = c("normal", "stt", "hypertrophy")

heartdata$exang = factor(heartdata$exang)
levels(heartdata$exang) = c("no","yes")

heartdata$slope = factor(heartdata$slope)
levels(heartdata$slope) = c("upsloping", "flat", "downsloping")

heartdata$ca = factor(heartdata$ca)

heartdata$thal = factor(heartdata$thal)
levels(heartdata$thal) = c("fixed", "normal", "reversable")

heartdata$target = factor(heartdata$target)
levels(heartdata$target) = c("diagnosis_No", "diagnosis_Yes")
# Rename some columns so later on the summary of regression is more readable:
names(heartdata) = c("age", "sex_", "cp_", "trestbps", "chol", "fbs_", "restecg_",
                     "thalach", "exang_", "oldpeak", "slope_", "ca", "thal_", "result")
# Randomly split the main dataframe to training and testing sets:
# Set the seed so this step can reappear
set.seed(10086)
sample = sample.int(n = nrow(heartdata), size = floor(.8*nrow(heartdata)), replace = F)
heart_train = heartdata[sample, ]
heart_test  = heartdata[-sample, ]
# I have put 80% (236 out of 296 entries) as the training set,
# and 20%  (60 out of 296 entries) as the testing set.
```