CWRU DSCI-453: 453SemProj-Final Report

Buxian Li

April 5, 2019

Contents

1 Analysis of rate of Heart Disease in binary classification

1.1 Background

With the development of method and tools of the data analysis and statistic analysis, medical researchers get the great chance of extracting more useful and specific information from the large dataset collected by clinical system. However, converting a medical problem into a statistical hypothesis with appropriate methodological and logical design and then back-translating the statistical results into relevant medical knowledge is a real challenge. On the other hand, Cardiovascular sickness is becoming a major reason of mortality in the present life.Distinguishing proof of cardiovascular ailment is an imperative yet an intricate errand that should be performed minutely and proficiently and the right robotization would be exceptionally attractive. In thie scenario, data analysis will use the historical data which collected tens of factors which vary from demographic factors to syndrom on patients to make a prediction whether the patient have the high probability of getting heart disease. The model is not a replacement of doctors in the hospital and clinic, but a warning for patients and a reference for doctors. In this project, the major goal is to find the key factors that influence the occurence rate and predict the final result, whether the patient get the heart disease.

- The Datasets

-This dataset comes from the Cleveland University Hospital database. It used to include 76 attributes, but 14 of them are used based on the experiments before. University Hospital of Cleveland is a major not-for-profit medical complex. This dataset is upload to Kaggle to be made use by Machine Learning researchers. It contains 303 rows and 14 coloumns.

1.2 Data Book

This is the dataset we are going to use in the project.

Index	Title	Units	Description
1	age	Years	age in years
2	sex	Dummy	(1 = male; 0 = female)
3	cp	NA	chest pain type
4	trestbps	mm HG	resting blood pressure?
5	chol	?mg/dl	serum cholestoral
6	fbs	Dummy	fasting blood sugar > 120 mg/dl 1 = true; 0 = false
7	restecg	NA	resting electrocardiographic results
8	thalach	time/second	maximum heart rate achieved

Index	Title	Units	Description
9	exang	Dummy	exercise induced angina (1 = yes; 0 = no)
10	oldpeak	NA	ST depression induced by exercise relative to rest
11	slope	NA	the slope of the peak exercise ST segment
12	ca	NA	number of major vessels (0-3) colored by flourosopy
13	thal	NA	3 = normal; 6 = fixed defect; 7 = reversable defect
14	target	NA	1 or 0
## Explo	ratory Data	Analysis and	Data Visualization

*Initial EDA

Exploratory Data Analysis is helping us to better understand the data. After downloading the data from kaggle, I used the read.csv to get the dataset into R.

```
Heart_Disease <- read.csv("H:/SemProject/Heart_Disease.csv")
summary(Heart_Disease)
```

```
##       age             sex               cp           trestbps
##  Min.   :29.00   Min.   :0.0000   Min.   :0.000   Min.   : 94.0
##  1st Qu.:47.50   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:120.0
##  Median :55.00   Median :1.0000   Median :1.000   Median :130.0
##  Mean   :54.37   Mean   :0.6832   Mean   :0.967   Mean   :131.6
##  3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:140.0
##  Max.   :77.00   Max.   :1.0000   Max.   :3.000   Max.   :200.0
##       chol            fbs            restecg          thalach
##  Min.   :126.0   Min.   :0.0000   Min.   :0.0000   Min.   : 71.0
##  1st Qu.:211.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:133.5
##  Median :240.0   Median :0.0000   Median :1.0000   Median :153.0
##  Mean   :246.3   Mean   :0.1485   Mean   :0.5281   Mean   :149.6
##  3rd Qu.:274.5   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:166.0
##  Max.   :564.0   Max.   :1.0000   Max.   :2.0000   Max.   :202.0
##      exang           oldpeak         slope             ca
##  Min.   :0.0000   Min.   :0.00   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.00   1st Qu.:1.000   1st Qu.:0.0000
##  Median :0.0000   Median :0.80   Median :1.000   Median :0.0000
##  Mean   :0.3267   Mean   :1.04   Mean   :1.399   Mean   :0.7294
##  3rd Qu.:1.0000   3rd Qu.:1.60   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :6.20   Max.   :2.000   Max.   :4.0000
##       thal           target
##  Min.   :0.000   Min.   :0.0000
##  1st Qu.:2.000   1st Qu.:0.0000
##  Median :2.000   Median :1.0000
##  Mean   :2.314   Mean   :0.5446
##  3rd Qu.:3.000   3rd Qu.:1.0000
##  Max.   :3.000   Max.   :1.0000
```

```
head(Heart_Disease)
```

```
##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal
## 1  63   1  3      145  233   1       0     150     0     2.3     0  0    1
## 2  37   1  2      130  250   0       1     187     0     3.5     0  0    2
## 3  41   0  1      130  204   0       0     172     0     1.4     2  0    2
## 4  56   1  1      120  236   0       1     178     0     0.8     2  0    2
## 5  57   0  0      120  354   0       1     163     1     0.6     2  0    2
## 6  57   1  0      140  192   0       1     148     0     0.4     1  0    1
```

```
##   target
## 1      1
## 2      1
## 3      1
## 4      1
## 5      1
## 6      1
```

```
library(magrittr)

library('psych')
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

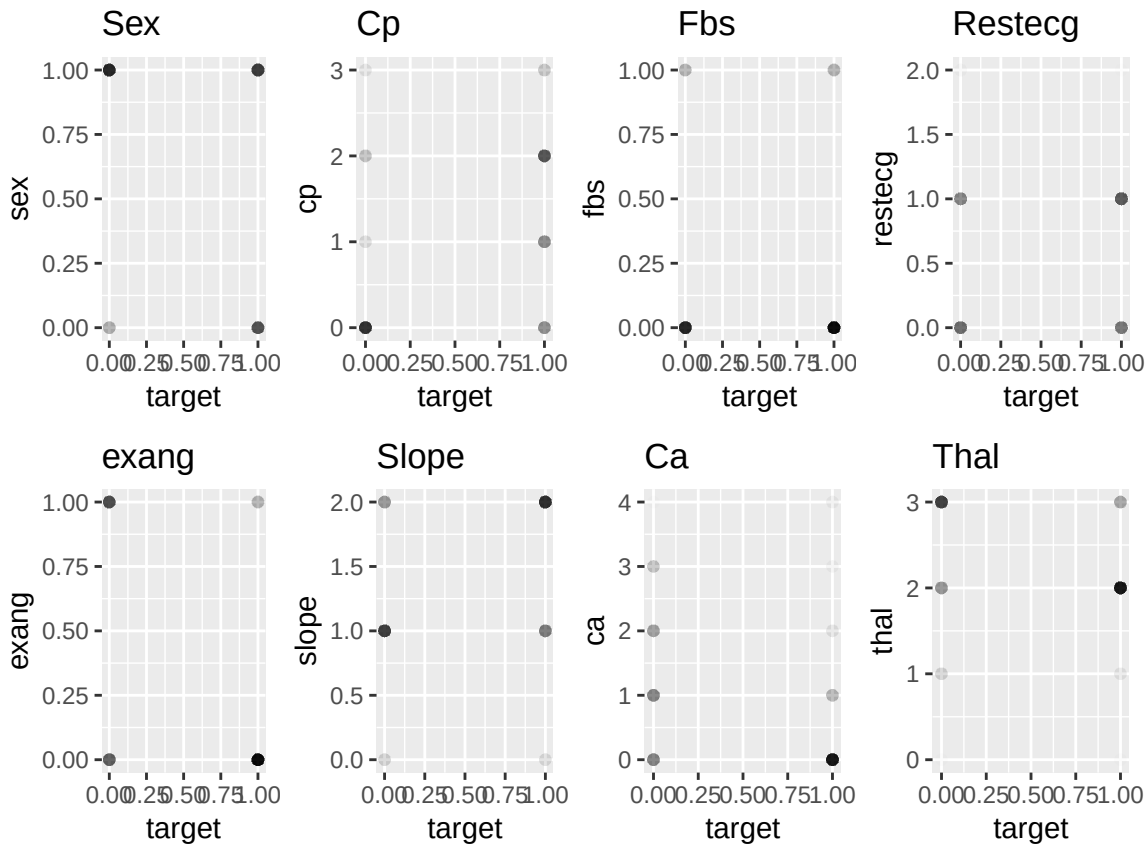
```
##
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

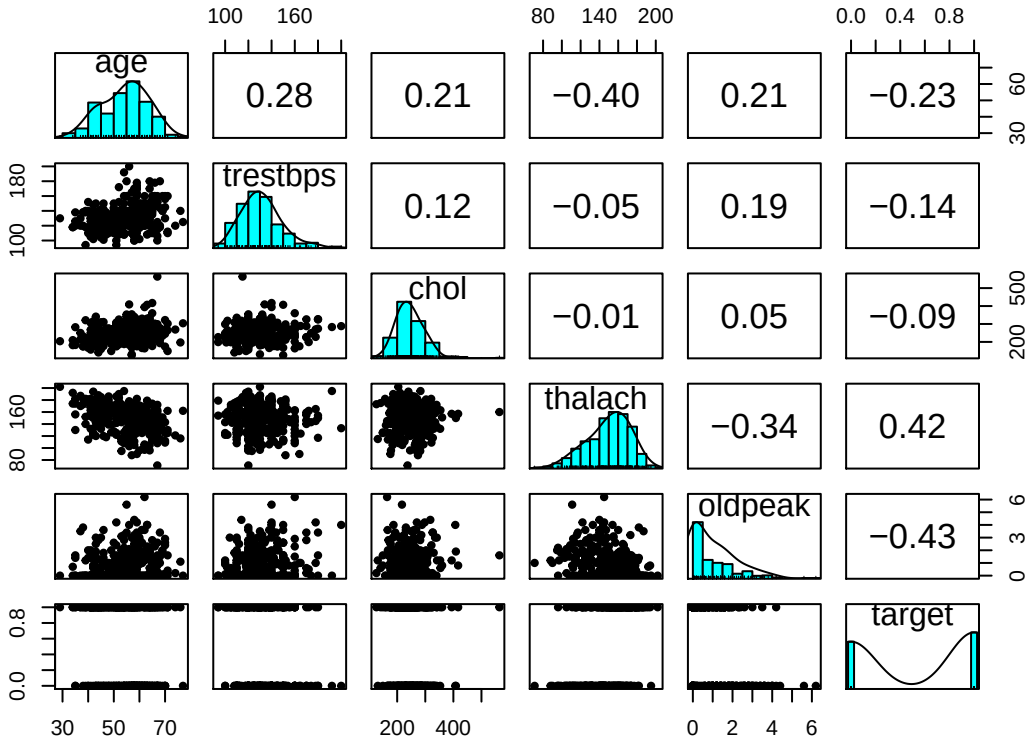
```
library(ggpubr)
attach(Heart_Disease)
theme_set(theme_grey())

sex.plot <- ggplot(Heart_Disease,aes(target,sex)) + geom_point(alpha = 0.01) + ggtitle("Sex")
cp.plot <- ggplot(Heart_Disease,aes(target,cp)) + geom_point(alpha = 0.01) + ggtitle("Cp")
fbs.plot <- ggplot(Heart_Disease,aes(target,fbs)) + geom_point(alpha = 0.01) + ggtitle("Fbs")
restecg.plot <- ggplot(Heart_Disease,aes(target,restecg)) + geom_point(alpha = 0.007) + ggtitle("Restecg
exang.plot <- ggplot(Heart_Disease,aes(target,exang)) + geom_point(alpha = 0.01) + ggtitle("exang")
slope.plot <- ggplot(Heart_Disease,aes(target,slope)) + geom_point(alpha = 0.01) + ggtitle("Slope")
ca.plot <- ggplot(Heart_Disease,aes(target,ca)) + geom_point(alpha = 0.01) + ggtitle("Ca")
thal.plot <- ggplot(Heart_Disease,aes(target,thal)) + geom_point(alpha = 0.01) + ggtitle("Thal")

ggarrange(sex.plot,cp.plot,fbs.plot,restecg.plot,exang.plot,slope.plot,ca.plot,thal.plot,ncol = 4,nrow =
```




```
library('psych')
library(magrittr)
Heart_Disease[c("age","trestbps","chol","thalach","oldpeak","target")] %>%
pairs.panels(smooth = FALSE,lm = FALSE, ellipses = FALSE)
```




```
heart <- Heart_Disease[,-c(7,8)]
dim(heart)
```

```
## [1] 303  12
```

After doing the EDA, I decided to delete variable thalach, because it has obvious relationship with age. In addtition, from the scatterplot, I found little relationship between restecg and target, so I also delete this variable from my dataset.

1.3 Demonstration of method

Make the baseline of the dataset. With the baseline, we can have clear vision of how our model fit the data.

```
table(heart$target)
```

```
##
##   0   1
## 138 165
```

```
165/303
```

```
## [1] 0.5445545
```

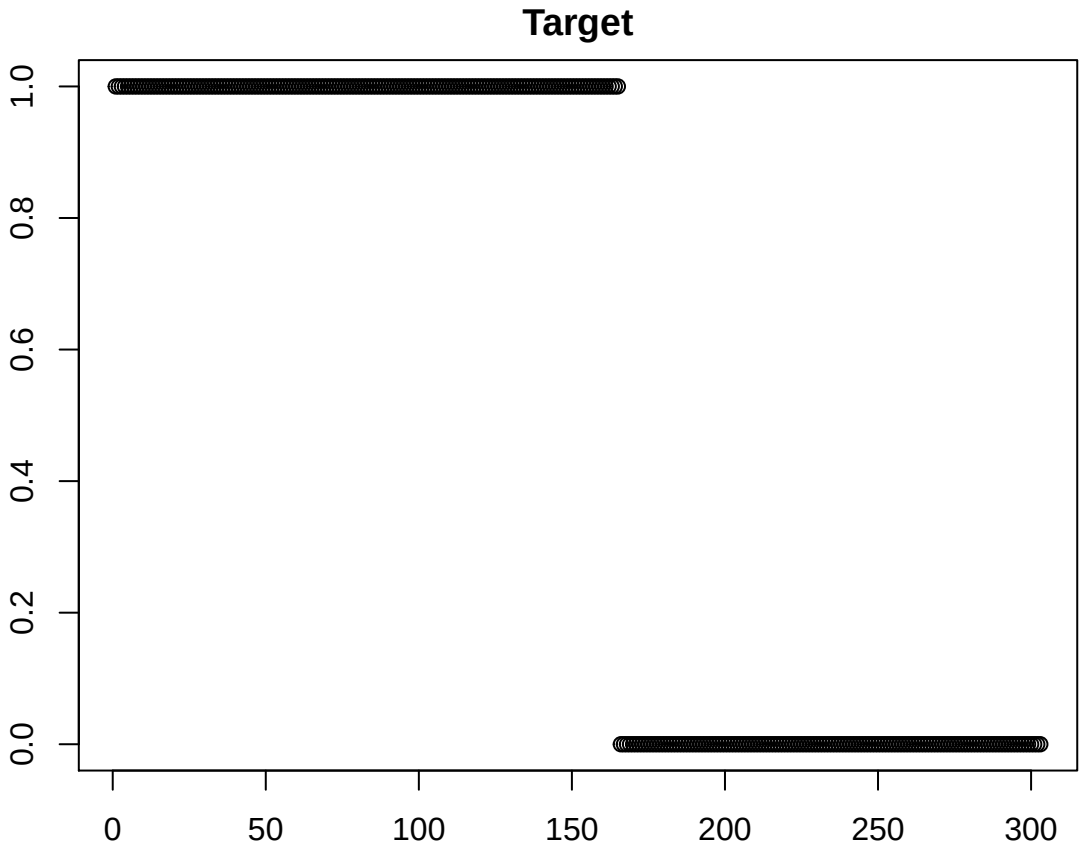
If we predict all the people in the data as heart-disease patient, we can get a 54.46% accuracy rate. So we want to have a result at least better than 54.46%. Logistic regression is a easy and good model for binary model prediction. Firstly we divide the data into training group and testing group. Each accounts of 50% of the whole dataset.

```
attach(heart)
```

```
## The following objects are masked from Heart_Disease:
##
##     age, ca, chol, cp, exang, fbs, oldpeak, sex, slope, target,
##     thal, trestbps
```

```
train.data <- heart[1:150,]
test.data <- heart[151:303,]

 glm.fit <- glm(target ~ .,data = heart,family = binomial)
 par(mar = rep(2, 4))
 plot(heart$target)
 title("Target")
```



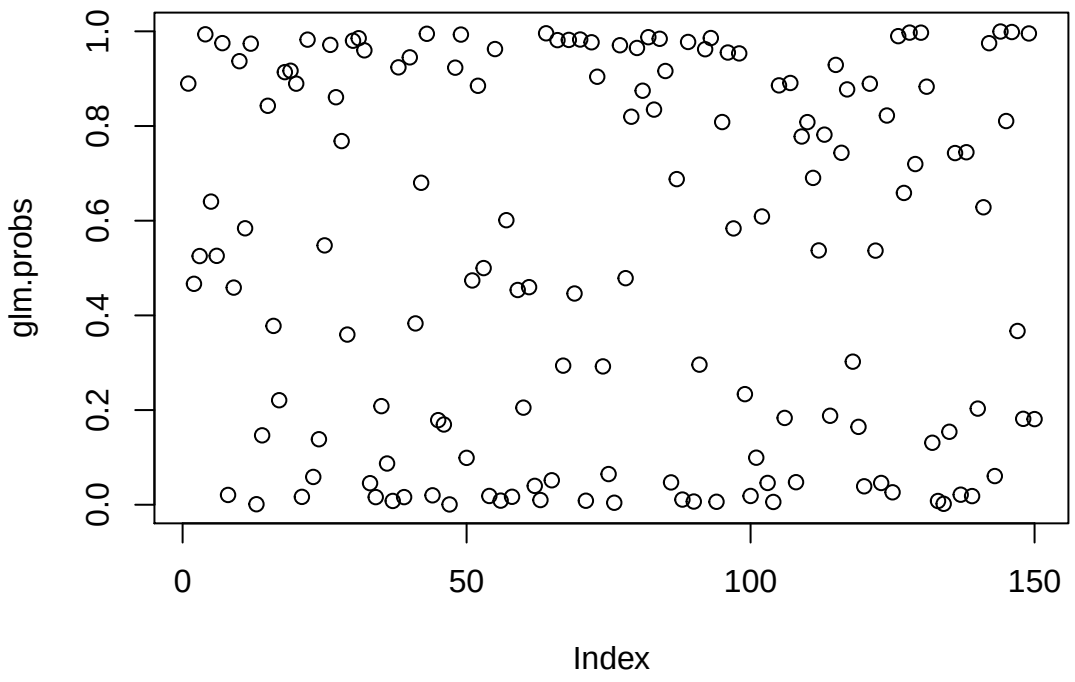
Result of this regression model is not good at all, because the I didn't randomly divide the data into two groups and all the patients who are sick are divided into training group. To have the same random number set to get the same result every time I run the model, I set the it as number 2.

```
library(magrittr)
set.seed(2)
list <- sample(303,150,replace = FALSE)
train.data <- heart[list,]
test.data <- heart[-list,]
```

After getting the new training and test group, I tried logistic regression again.

```
glm.fit <- glm(target~.,data = train.data,family = binomial(link = "logit"))

glm.probs <- predict(glm.fit,newdata = train.data,type = "response" )
plot(glm.probs)
```




```
glm.pred <- rep("Health",150)
glm.pred[glm.probs > .5] = "Sick"
glm.probs2 <- predict(glm.fit,newdata = test.data,type = "response" )
glm.pred2 <- rep("Health",153)
glm.pred2[glm.probs > .5] = "Sick"
table(test.data$target,glm.pred2)
```

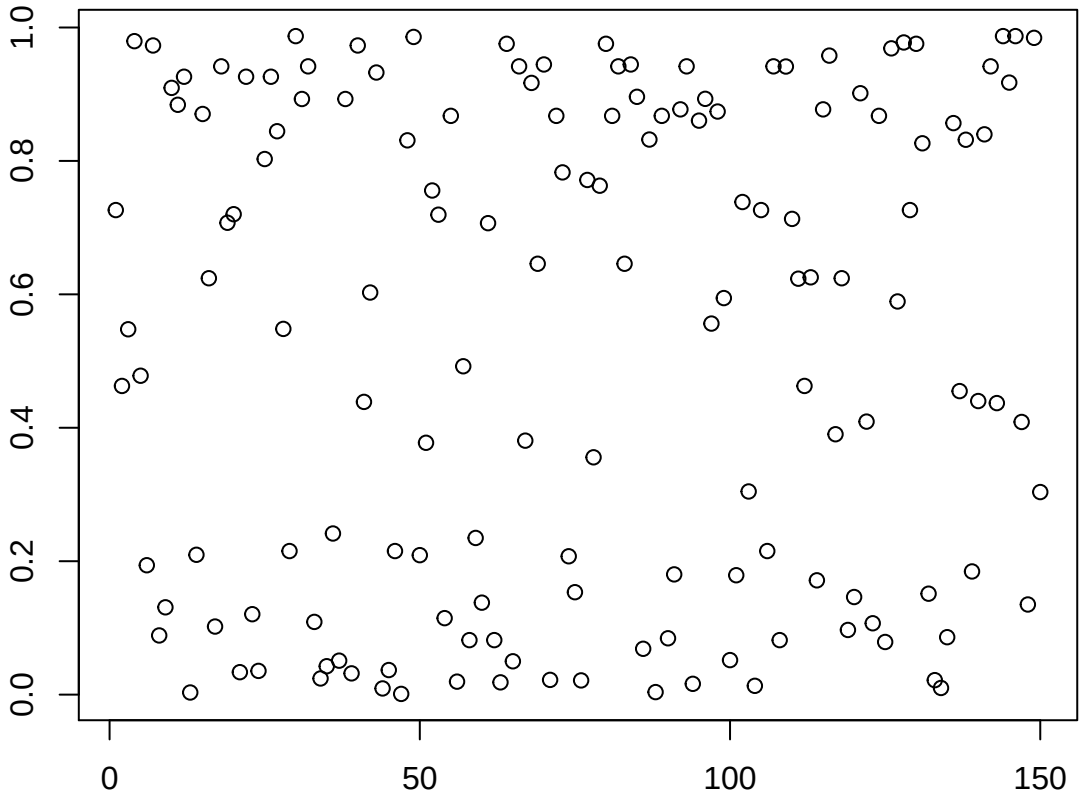
```
##    glm.pred2
##     Health Sick
##   0     29   38
##   1     42   44
```

```
(21+37)/153
```

```
## [1] 0.379085
```

By having the first logistics model, I found that the targets of first half are all the sick patients, so I generate random numbers to have the new model, but the model only got 37.90% correct prediction. As a result, I chose to run the model with the significant variables from last model.

```
glm.fit2 <- glm(target ~ sex + cp + oldpeak + exang + ca,data = train.data,family = binomial(link = "lo
glm.probs3 <- predict(glm.fit2,newdata = train.data,type = "response" )
par(mar = rep(2, 4))
plot(glm.probs3)
```




```
glm.pred3 <- rep("Health",150)
glm.pred3[glm.probs > .5] = "Sick"
table(train.data$target,glm.pred3)
```

```
##    glm.pred3
##     Health Sick
##   0     59   12
##   1     11   68
```

```
glm.probs4 <- predict(glm.fit,newdata = test.data,type = "response" )
glm.pred4 <- rep("Health",153)
glm.pred4[glm.probs4 > .5] = "Sick"
table(test.data$target,glm.pred4)
```

```
##    glm.pred4
##     Health Sick
##   0     46   21
##   1      8   78
```

```
(43+58)/153
```

```
## [1] 0.6601307
```

The prediction results improved significantly and the accuracy rate is larger than our basline. However, we are not satisfied with the current rate. We want to try some models that contains non-linear relationship among independent variables and dependent variables. First, we use anova to decide the level of polynominial we want to apply on these variables

1.4 GAM

```
require(ISLR)
```

```
## Loading required package: ISLR
```

```
require(boot)
```

```
## Loading required package: boot
```

```
##
## Attaching package: 'boot'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

```
fit.age.1 <- lm(target ~ age,data = heart)
fit.age.2 <- lm(target ~ poly(age,2),data = heart)
fit.age.3 <- lm(target ~ poly(age,3),data = heart)
fit.age.4 <- lm(target ~ poly(age,4),data = heart)
fit.age.5 <- lm(target ~ poly(age,5),data = heart)
anova(fit.age.1,fit.age.2,fit.age.3,fit.age.4,fit.age.5)
```

```
## Analysis of Variance Table
##
## Model 1: target ~ age
## Model 2: target ~ poly(age, 2)
## Model 3: target ~ poly(age, 3)
## Model 4: target ~ poly(age, 4)
## Model 5: target ~ poly(age, 5)
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1    301 71.329
## 2    300 70.642  1   0.68770 2.9790 0.08539 .
## 3    299 69.625  1   1.01665 4.4040 0.03670 *
## 4    298 69.623  1   0.00151 0.0065 0.93555
## 5    297 68.562  1   1.06129 4.5973 0.03283 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Analysis of Variance Table
##
## Model 1: target ~ trestbps
## Model 2: target ~ poly(trestbps, 2)
## Model 3: target ~ poly(trestbps, 3)
## Model 4: target ~ poly(trestbps, 4)
## Model 5: target ~ poly(trestbps, 5)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    301 73.570
## 2    300 73.508  1  0.062052 0.2526 0.6156
## 3    299 73.260  1  0.247569 1.0076 0.3163
## 4    298 73.105  1  0.155606 0.6333 0.4268
## 5    297 72.971  1  0.134276 0.5465 0.4603
```

```
## Analysis of Variance Table
##
## Model 1: target ~ chol
## Model 2: target ~ poly(chol, 2)
```

```
## Model 3: target ~ poly(chol, 3)
## Model 4: target ~ poly(chol, 4)
## Model 5: target ~ poly(chol, 5)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    301 74.603
## 2    300 74.157  1   0.44595 1.8016 0.1805
## 3    299 73.781  1   0.37554 1.5172 0.2190
## 4    298 73.531  1   0.24970 1.0088 0.3160
## 5    297 73.516  1   0.01565 0.0632 0.8016
```

```
## Analysis of Variance Table
##
## Model 1: target ~ oldpeak
## Model 2: target ~ poly(oldpeak, 2)
## Model 3: target ~ poly(oldpeak, 3)
## Model 4: target ~ poly(oldpeak, 4)
## Model 5: target ~ poly(oldpeak, 5)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    301 61.209
## 2    300 60.710  1   0.49895 2.4509 0.1185
## 3    299 60.634  1   0.07519 0.3694 0.5438
## 4    298 60.471  1   0.16342 0.8028 0.3710
## 5    297 60.461  1   0.00956 0.0469 0.8286
```

We found out that I want to use only polynominal of age in the model and keep all other variables the same. I also apply the natural spline.

```
library (gam)
```

```
## Loading required package: splines
```

```
## Loading required package: foreach
```

```
## Loaded gam 1.16
```

```
library(mgcv)
```

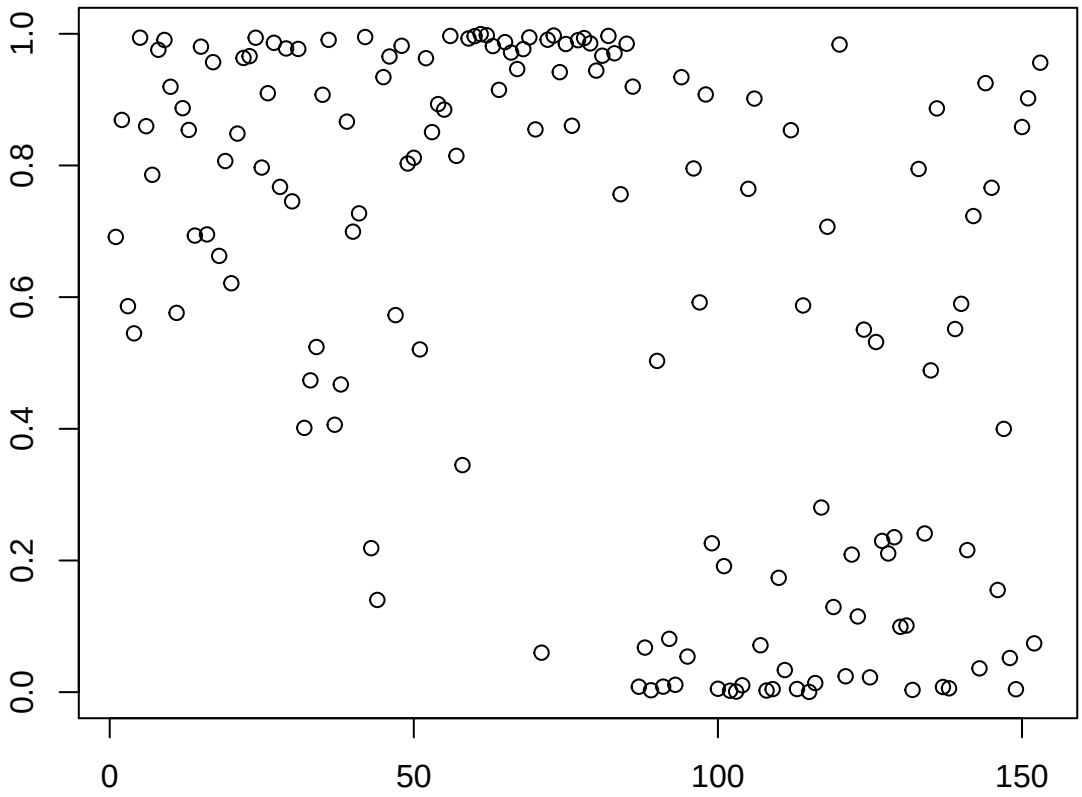
```
## Loading required package: nlme
```

```
## This is mgcv 1.8-27. For overview type 'help("mgcv-package")'.
```

```
##
## Attaching package: 'mgcv'
```

```
## The following objects are masked from 'package:gam':
##
##     gam, gam.control, gam.fit, s
```

```
full.gam<-gam(data=train.data,target ~ ns(age,3)+ns(trestbps,1)+ns(chol,1)+ns(oldpeak,1)+age+sex+cp+fbs
gam.probs <- predict.gam(full.gam,newdata = test.data,type = "response")
par(mar = rep(2, 4))
plot(gam.probs)
```




```
gam.pred <- rep("Health",153)
gam.pred[gam.probs > .5] = "Sick"
table(test.data$target,gam.pred)
```

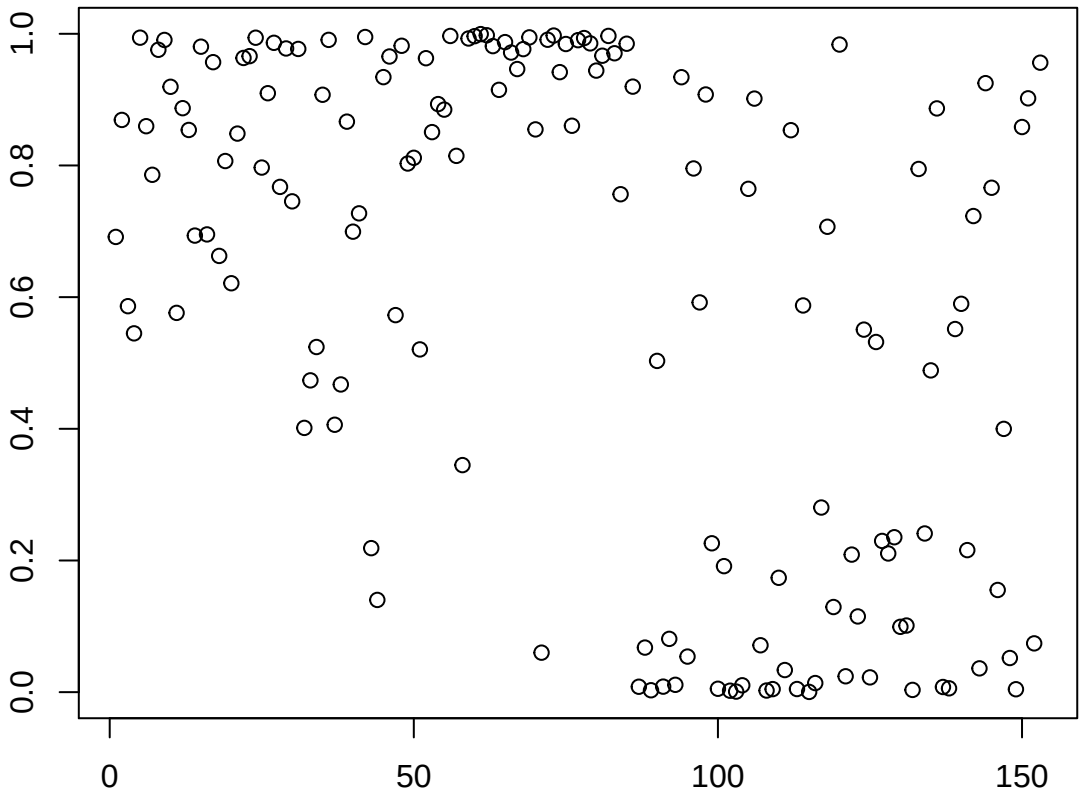
```
##    gam.pred
##     Health Sick
##   0     44   23
##   1      8   78
```

```
(53+70)/153
```

```
## [1] 0.8039216
```

Results have improved significantly and I think I am going to use the trick I used on logistic regression. I am going to use those variables that have obvious relationship with target.

```
full.gam2<-gam(data=train.data,target ~ ns(oldpeak,1)+sex+cp+exang+ca,family = 'binomial')
gam.probs2 <- predict.gam(full.gam,newdata = test.data,type = "response")
par(mar = rep(2, 4))
plot(gam.probs2)
```




```
gam.pred2 <- rep("Health",153)
gam.pred2[gam.probs2 > .5] = "Sick"
table(test.data$target,gam.pred2)
```

```
##    gam.pred2
##     Health Sick
##   0     44   23
##   1      8   78
```

```
(54+67)/153
```

```
## [1] 0.7908497
```

It seems the result didnot improved. I tried knn method.Because the decision tree use distance to measure the data, large scale and mean variables will inevitably have larger influence on model, which is not a good news for me. I will scale them into standard range with same standard devision.

```
standage <- scale(age)
standtrestbps <- scale(trestbps)
standchol <- scale(chol)
standoldpeak <- scale(oldpeak)
newdata <- data.frame(age2 = standage,sex,cp,trestbps2 = standtrestbps,chol2 = standchol,fbs,exang,oldp
```

Because this is not a large dataset, I control the K no larger than 10.

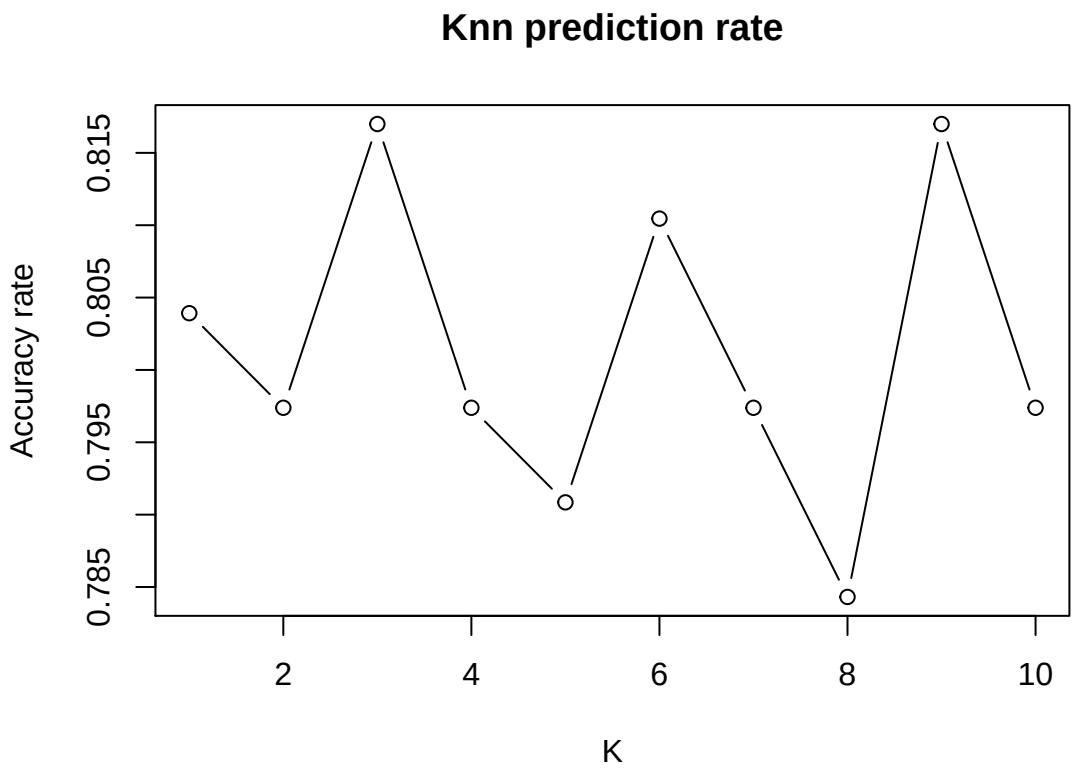
```
set.seed(1)
library(plyr)
```

```
##
```

```
## Attaching package: 'plyr'
```

```
## The following object is masked from 'package:ggpubr':
##
##     mutate
```

```
library(class)
train.x <- newdata[list,]
test.x <- newdata[-list,]
train.y <- target[list]
test.y <- target[-list]
rate <- rep(0,10)
for (i in 1:10){
  knn.pred2 <- knn(train.x, test.x, train.y, k = i)
  Righ <- count(knn.pred2 != test.y)
  rate[i] <- Righ[1,2]/153
}
plot(cbind(1:10),rate,main = "Knn prediction rate",ylab = "Accuracy rate",xlab = "K",type = "b")
```




```
rate[9]
```

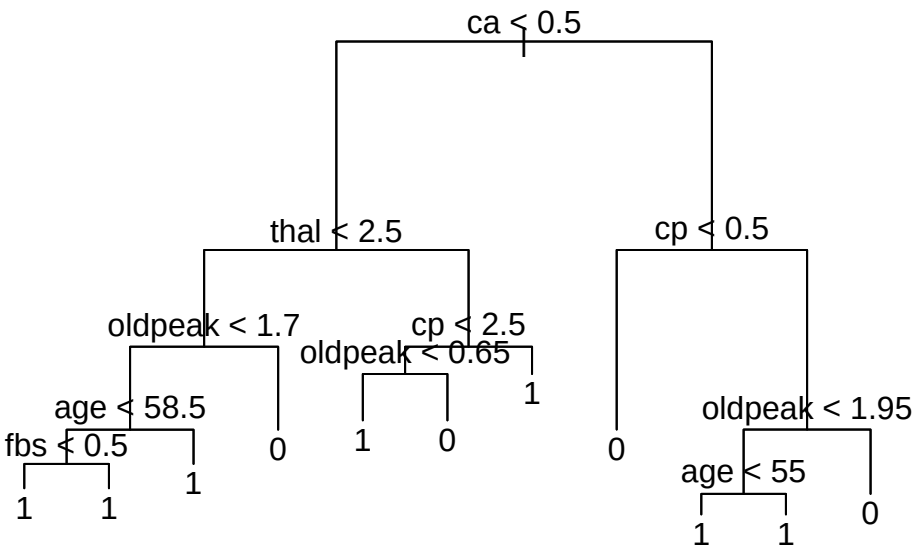
```
## [1] 0.8169935
```

When K is equal to 8, I get the best result. Dicision tree is a decision support tool that use a tree-like model of decisions. Because my dataset has several categorical variables and binary variables, I think decision tree will get a better result.

```
require(tree)
```

```
## Loading required package: tree
```

```
tree.target2 <- tree(as.factor(target)~., data = train.data)
plot(tree.target2)
text(tree.target2, pretty=0)
```




```
tree.pred <- predict(tree.target2,test.data,type="class")
table(tree.pred,test.data$target)
```

```
##
## tree.pred  0  1
##         0 47 13
##         1 20 73
```

```
(47+73)/153
```

```
## [1] 0.7843137
```

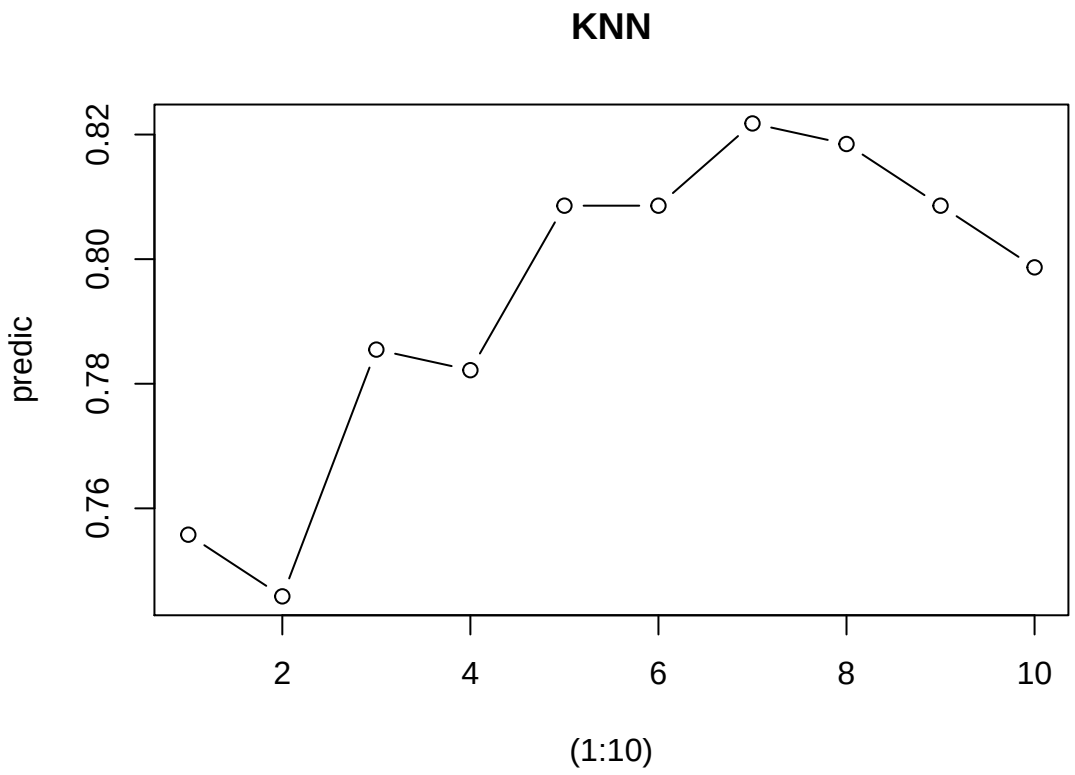
After trying all these models, I decide to use cross validation to apply on decision tree and knn method to determine my final model. Cross validation is a effective way to calculate the actual accuracy rate.

```
tree.predic <- rep(0,303)
for (i in 1:nrow(heart)){
  training <- heart[-i,]
  testing <- heart[i,]
  tree.target3 <- tree(as.factor(target)~., data = training)
  tree.predict <- predict(tree.target3,testing,type="class")
  tree.predic[i] <- tree.predict
```

```
}
(128+16)/303
```

```
## [1] 0.4752475
```

```
knn.predic <- rep(0,303)
predic <- rep(0,10)
for (j in 1:10){
  for (i in 1:nrow(heart)){
    training.x <- newdata[-i,]
    testing.x <- newdata[i,]
    training.y <- target[-i]
    testing.y <- target[i]
    knn.pred3 <- knn(training.x, testing.x, training.y, k = j)
    knn.predic[i] <- knn.pred3
  }
a <- table(knn.predic,target)
predic[j] <- (a[1,1]+a[2,2])/303
}
plot((1:10),predic,main = "KNN",type = "b")
```



It is obious that KNN get far much better result and get the best result at K =7.

1.5 Challenge Results

With all the models I fitted, the best result just reached no more than 85%, which means there are at least 150 wrong prediction in 1000 people. This is not a good result, especially when the dataset includes 11

independent variables. In addition, this data set contains several categorical variables. Although converting them into dummy variables maybe a good idea, but it won't help on knn method.

1.6 Conclusion

This project do has some result on binary classification on heart disease, but 83% prediction rate is still good enough. Collecting new variables may help to better predict the result. In conclusion, this project could provide useful information to health provider.