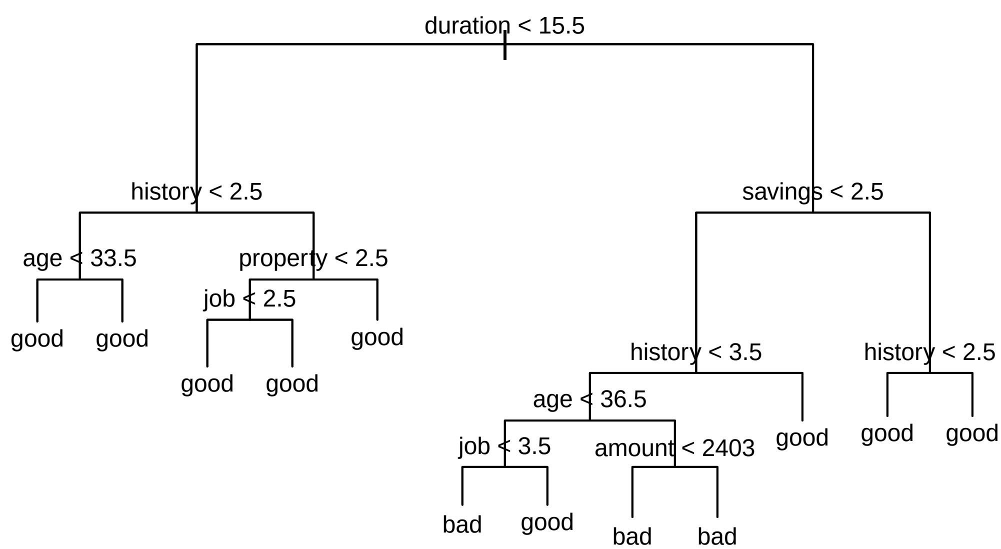
Machine Learning 2

Karthikeyan Devarajan - Karde799

Question 1:Decision Tree with Train/Test/Validate as 50/25/25.

Decison Tree can be split into nodes using gini index and deviation method.
1) Deviance

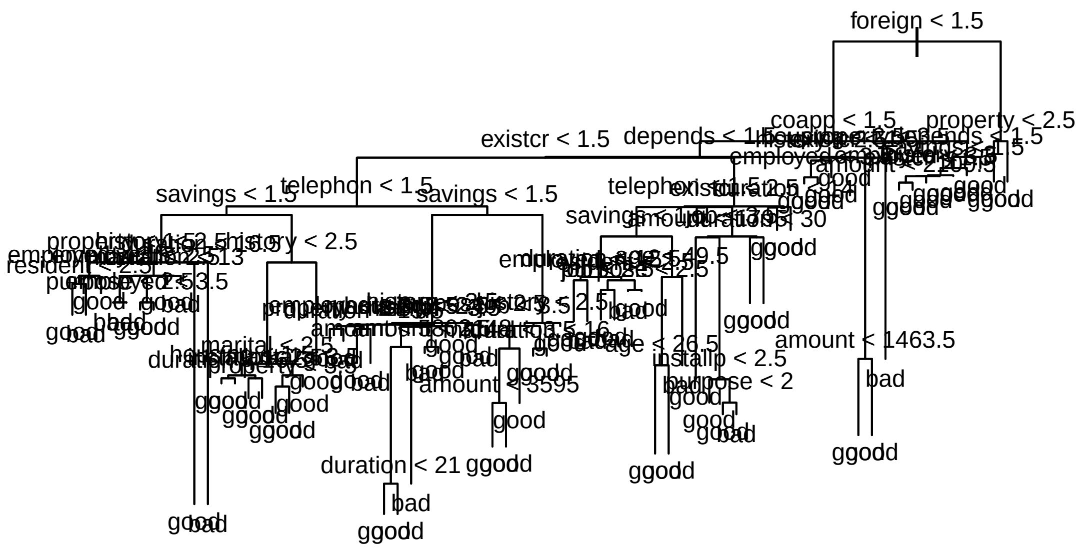



```
## The confusion matrix for train data using deviance method is:

##
## train_model_pred_d bad good
##              FALSE  71  303
##              TRUE   77   49

## The miscalculation rate for train data using deviance method is: 0.24

## The confusion matrix for test data using deviance method is:
```

```
##
## test_model_pred_d bad good
##             FALSE  31  133
##             TRUE   49   37
```

```
## The miscalculation rate for test data using deviance method is: 0.136
```

2)Gini index




```
## The confusion matrix for train data using gini index is:
```

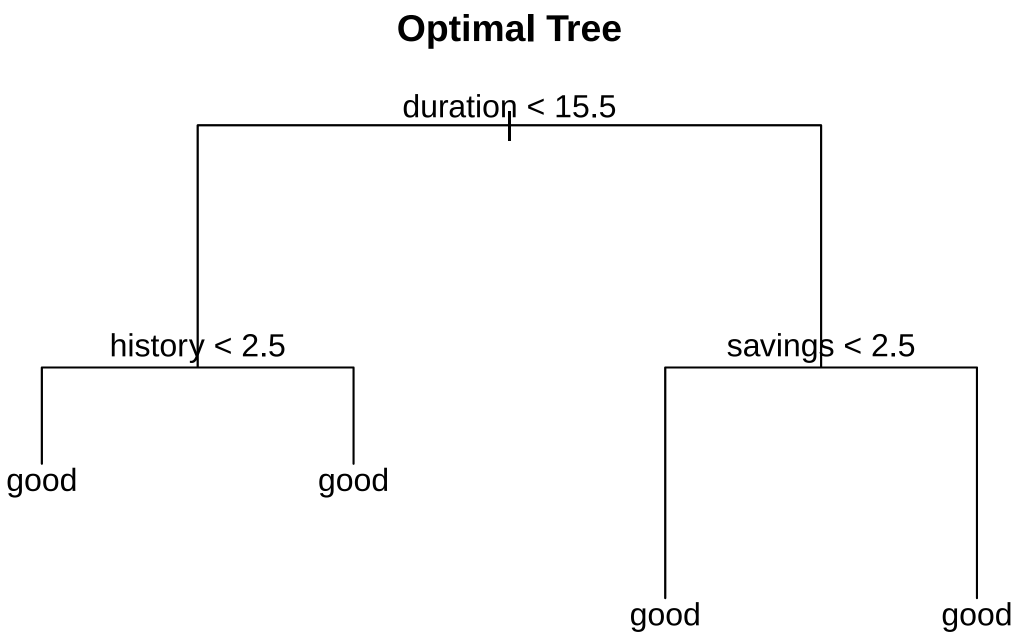
```
##
## train_model_pred_g bad good
##              FALSE  57  315
##              TRUE   91   37
```

```
## The miscalculation rate for train data using gini index is: 0.188
```

```
## The confusion matrix for test data using gini index is:
```

```
##
## test_model_pred_g bad good
##             FALSE  25  134
##             TRUE   55   36
```

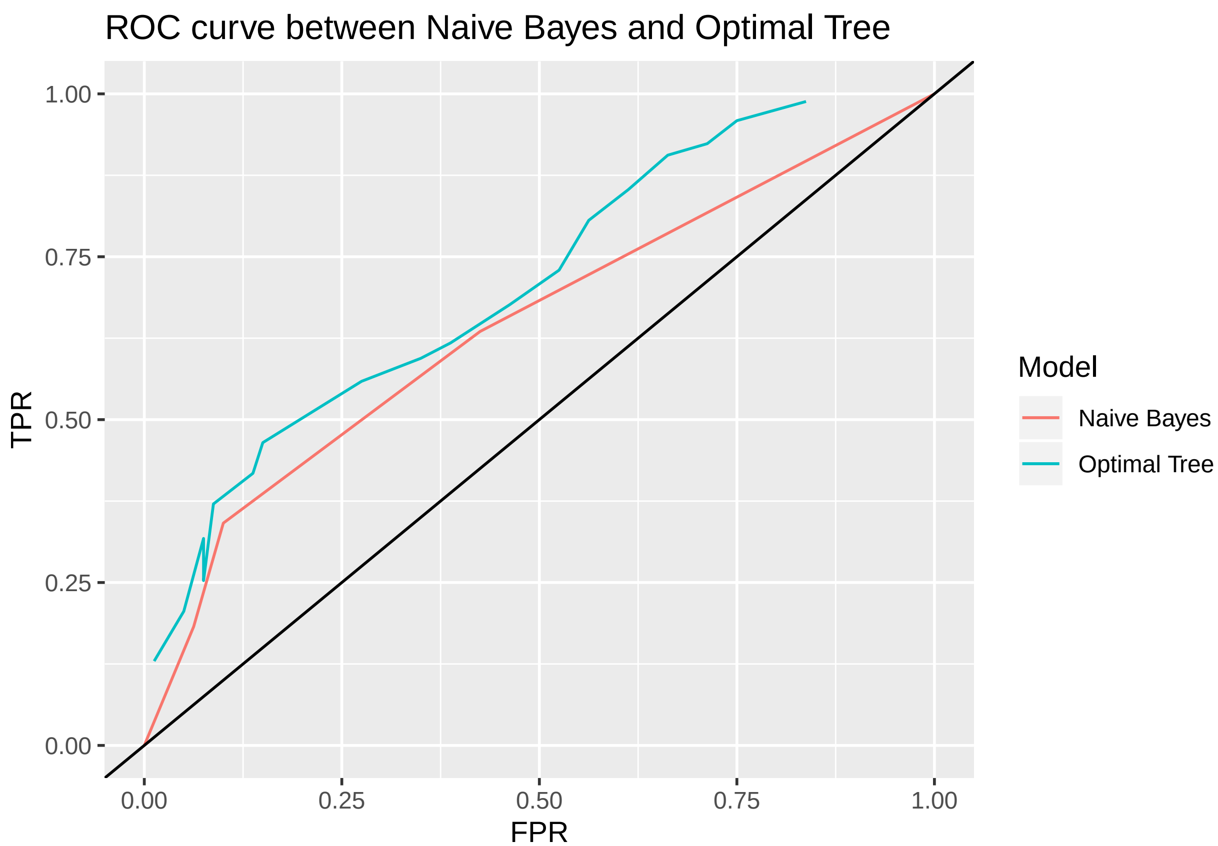
```
## The miscalculation rate for test data using gini index is: 0.122

## The optimal leaf value is: 4

## The optimum number of nodes: 1 2 4 5 3 6 7
```



Naive Bayes

```
## Warning in data.matrix(newdata): NAs introduced by coercion

## The confusion matrix for Naives Bayes is:

##            Actual
## Prediction bad good
##      FALSE  52  101
##      TRUE   28   69

## The miscalculation rate for naives bayes is: 0.484
```

ROC Curve



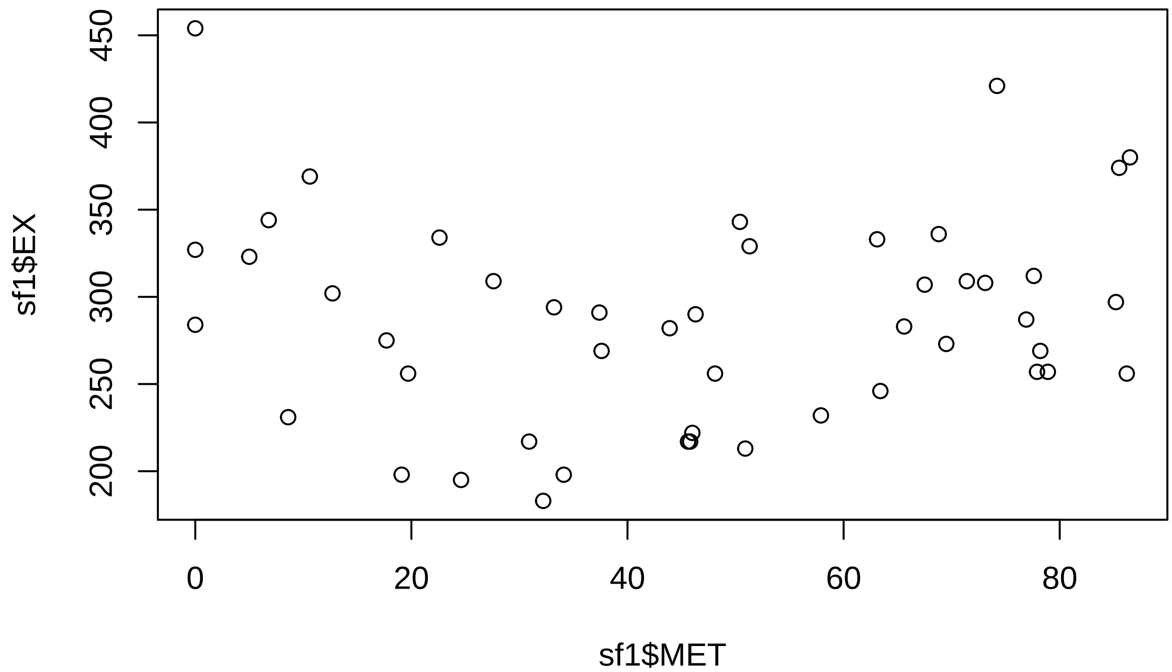
Naives Bayes is similar to optimal tree since it has covered more area in ROC curve.

Question 2:Uncertainity Estimation

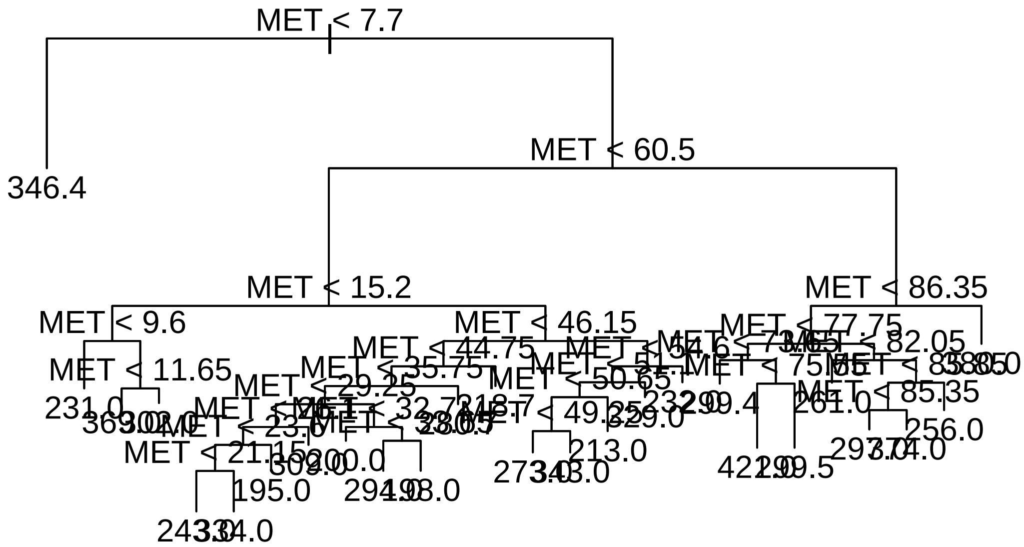
a)Plot EX on MET afte rearranging

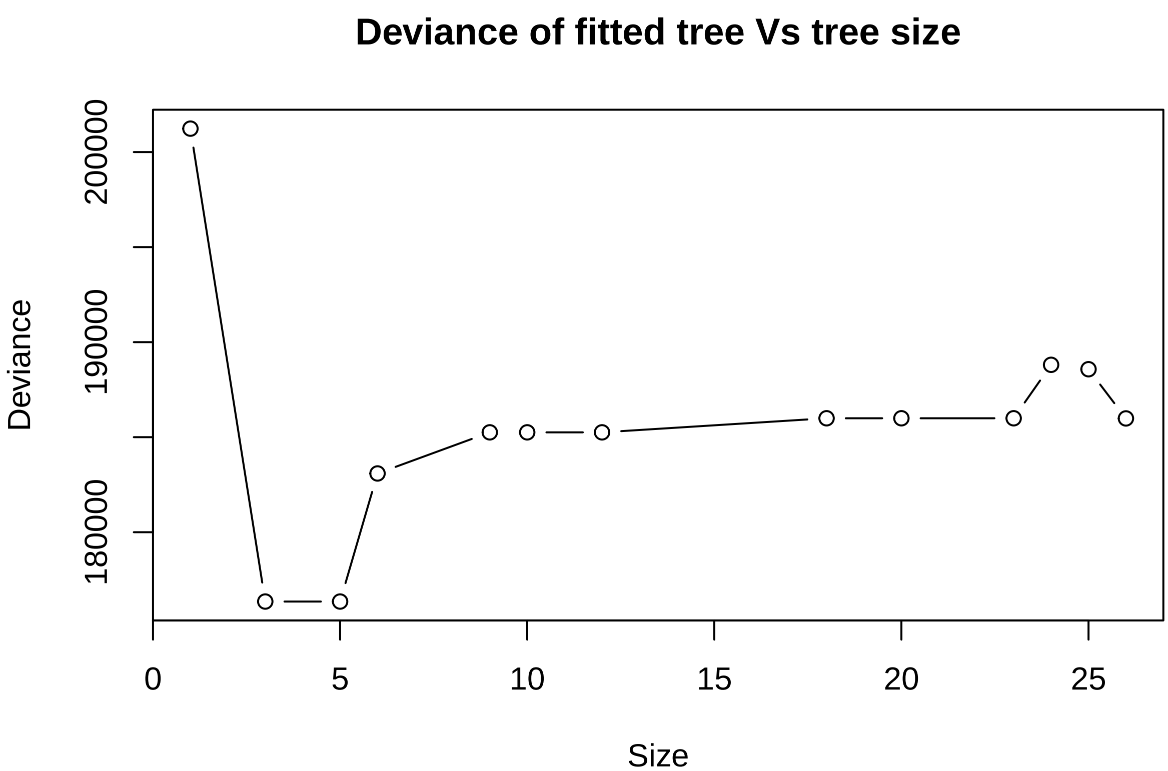
```
##     EX  ECAB  MET GROW YOUNG  OLD WEST STATE
## 1  327  87.0  0.0  3.7  28.5 11.2    0    VT
## 2  284  93.9  0.0 13.3  30.7  8.7    1    ID
## 3  454 125.8  0.0 13.7  29.1  7.8    1    WY
## 4  323  86.0  5.0 21.9  30.3  7.4    1    LA
## 5  344  98.0  6.8 31.5  28.0  9.0    1    CO
## 6  231  57.4  8.6  0.5  32.1  8.7    1    MS
## 7  369  93.4 10.6  2.9  30.2  9.3    1    ND
## 8  302  88.2 12.7  4.6  28.9 10.5    1    SD
## 9  275  94.3 17.7 14.7  26.4 11.2    0    NH
## 10 198  68.6 19.1 -6.2  29.4 10.9    1    AR
## 11 256  85.5 19.7  6.9  29.6 11.0    0    ME
## 12 334  97.6 22.6 13.4  28.9  9.7    1    MT
## 13 195  78.7 24.6 12.4  30.8  6.9    0    NC
## 14 309  86.2 27.6 39.4  31.5  5.4    1    NM
## 15 217  85.1 30.9 -7.4  30.0  9.3    0    WV
## 16 183  65.2 32.2 12.9  32.9  6.3    0    SC
```

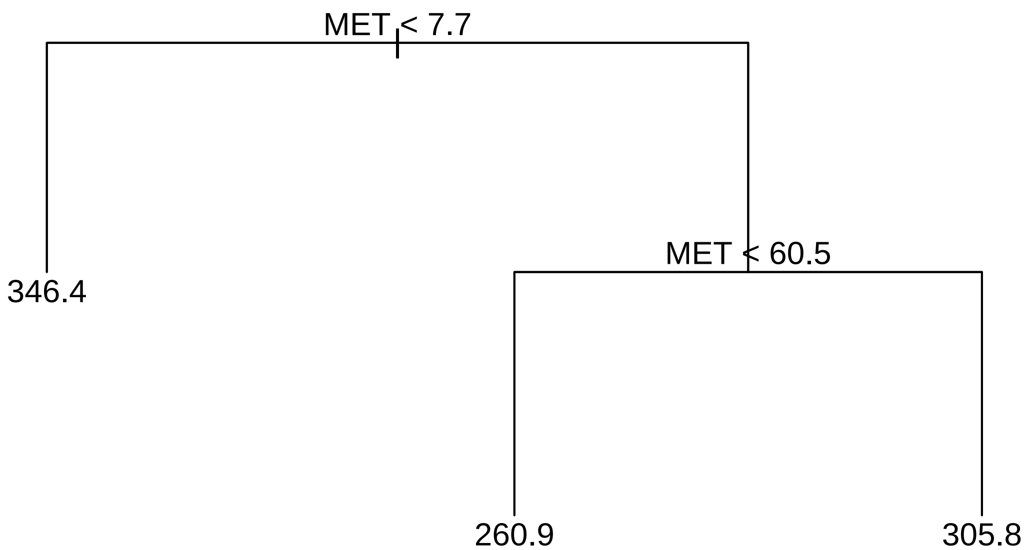
```
## 17 294 100.2 33.2  5.3  27.3 11.9    1    IA
## 18 198  76.8 34.1  0.3  29.4  9.6    0    KY
## 19 291 102.2 37.4 13.7  26.8 11.0    1    KS
## 20 269  99.1 37.6  6.8  26.6 11.6    1    NB
## 21 282  84.9 43.9  6.4  27.4 10.7    1    OK
## 22 217  69.4 45.6  7.0  30.5  8.0    1    AL
## 23 217  75.1 45.8  8.1  28.9  8.7    0    TE
## 24 222  73.0 46.0 14.4  30.0  7.4    0    GA
## 25 290 104.3 46.3 14.9  27.4 10.2    0    WI
## 26 256 110.8 48.1 18.3  27.5  9.6    0    IN
## 27 343  98.0 50.4 15.7  27.7 10.4    1    OR
## 28 213  77.2 50.9 21.9  28.8  7.3    0    VA
## 29 329  95.7 51.3 14.4  28.8 10.4    1    MN
## 30 232  99.1 57.9  9.8  25.6 11.7    1    MO
## 31 333 100.4 63.1 19.9  27.5  9.8    1    WA
## 32 246  98.8 63.4 24.1  28.8  7.8    1    TX
## 33 283  80.9 65.6 77.2  25.5 11.2    0    FL
## 34 307  92.5 67.5 28.7  31.9  6.7    1    UT
## 35 336 116.1 68.8 39.9  26.4  8.0    0    DE
## 36 273 111.8 69.5 21.8  26.9  9.2    0    OH
## 37 309  90.2 71.4 74.3  29.7  6.9    1    AZ
## 38 308 108.4 73.1 22.2  28.0  8.2    0    MI
## 39 421 205.0 74.2 77.8  25.6  6.4    1    NV
## 40 287 120.9 76.9 15.5  25.4  9.7    0    IL
## 41 312 121.6 77.6 25.4  25.2  9.6    0    CT
## 42 257 103.1 77.9  7.8  25.7 10.0    0    PA
## 43 269  93.4 78.2 31.1  27.5  7.3    0    MD
## 44 257 117.9 78.9 25.5  24.8  9.2    0    NJ
## 45 297 107.5 85.2 10.2  25.1 11.1    0    MA
## 46 374 111.5 85.5 12.9  24.0 10.1    0    NY
## 47 256  94.9 86.2  1.0  25.3 10.4    0    RI
## 48 380 112.6 86.5 48.5  26.2  8.8    1    CA
```

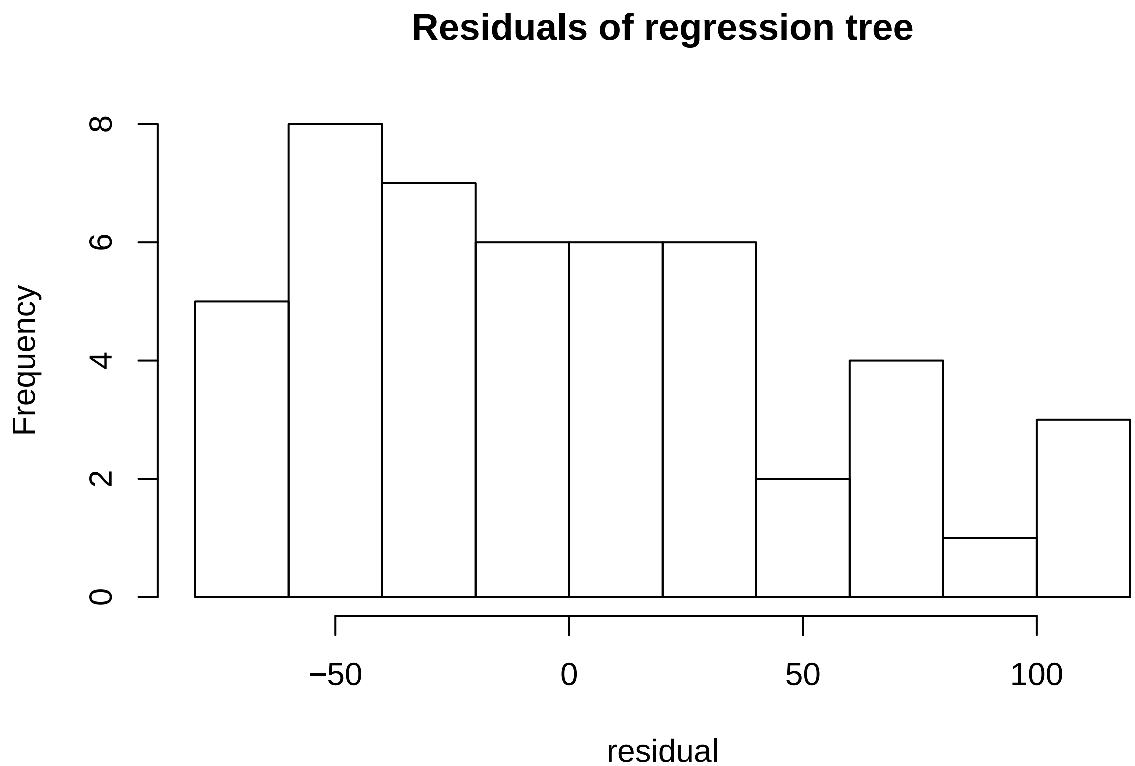

450
400
350
300
250
200
sf1$EX
0
20
40
60
80
sf1$MET

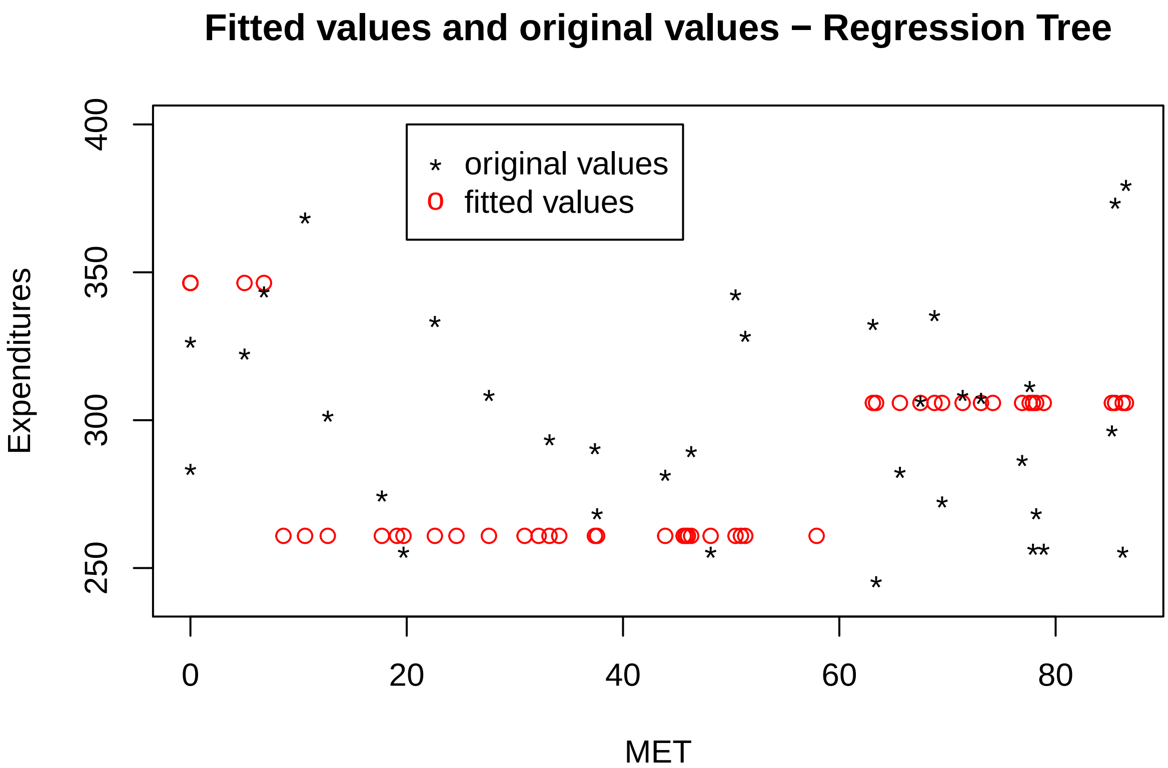
b) Original and Fitted Value,Histogram of Residuals





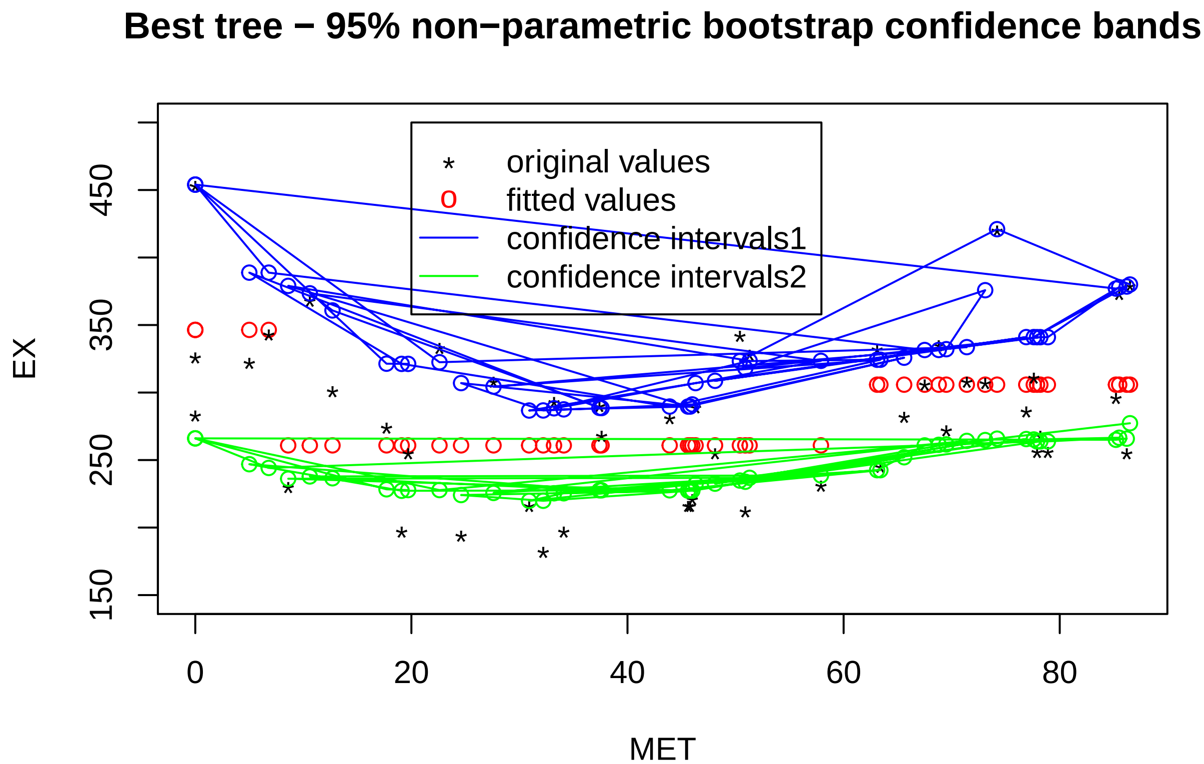

MET < 7.7
346.4
MET < 60.5
260.9
305.8


Residuals of regression tree
Frequency
0
2
4
6
8
−50
0
50
100
residual



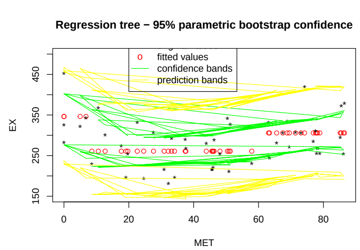
The histogram is skewed towards right. The fit is said to be not proper, when it is skewed towards any one direction. So, this fit need to be improved.

c)Non-Parametric Bootstrap with 95% Confidence Band



The confidence interval is very narrow. This is due to the distribution is skewed to right. This model is similar to the previous graph. The range for EX intervals has increased from 150 to 400. The extreme value is wider than the previuos graph.

d)Parametric Bootstrap with 95% Confidence Band



This is for parametric bootstrap. The graph in question 2 lies between this confidence interval. The paramteric bootstrap interval better than non parametric bootstrap. This can be considered as a good fit but it can be improved.

Question 3: Principal Component Analysis

```
## The lambda values are:
```

```
##                lambda
## comp 1   1.201921e+02
## comp 2   5.335616e+00
## comp 3   1.032371e+00
## comp 4   2.309922e-01
## comp 5   9.381398e-02
## comp 6   7.709080e-02
## comp 7   1.232559e-02
## comp 8   6.079497e-03
## comp 9   3.721974e-03
## comp 10  3.170234e-03
## comp 11  1.977398e-03
## comp 12  1.627416e-03
## comp 13  1.068155e-03
## comp 14  9.350318e-04
## comp 15  7.482932e-04
## comp 16  5.131173e-04
## comp 17  4.540643e-04
## comp 18  3.847087e-04
## comp 19  3.109677e-04
## comp 20  2.973119e-04
## comp 21  2.715252e-04
## comp 22  2.425310e-04
## comp 23  2.085016e-04
## comp 24  1.941418e-04
## comp 25  1.608568e-04
## comp 26  1.572987e-04
## comp 27  1.482543e-04
## comp 28  1.253020e-04
## comp 29  1.196473e-04
## comp 30  1.118573e-04
## comp 31  1.096772e-04
## comp 32  9.943560e-05
## comp 33  9.350305e-05
## comp 34  8.848653e-05
## comp 35  8.521759e-05
## comp 36  8.040405e-05
## comp 37  7.619300e-05
## comp 38  7.139990e-05
## comp 39  6.854340e-05
## comp 40  6.704991e-05
## comp 41  6.428768e-05
## comp 42  6.070819e-05
## comp 43  5.776255e-05
## comp 44  5.673486e-05
## comp 45  5.420189e-05
## comp 46  5.107449e-05
## comp 47  4.884221e-05
## comp 48  4.828848e-05
## comp 49  4.766477e-05
## comp 50  4.518079e-05
## comp 51  4.256623e-05
## comp 52  4.183435e-05
## comp 53  4.066684e-05
```

```
## comp 54  3.974796e-05
## comp 55  3.711319e-05
## comp 56  3.628218e-05
## comp 57  3.555323e-05
## comp 58  3.355271e-05
## comp 59  3.318506e-05
## comp 60  3.073139e-05
## comp 61  3.045438e-05
## comp 62  2.932486e-05
## comp 63  2.903514e-05
## comp 64  2.805553e-05
## comp 65  2.753248e-05
## comp 66  2.707009e-05
## comp 67  2.569241e-05
## comp 68  2.492170e-05
## comp 69  2.465567e-05
## comp 70  2.378068e-05
## comp 71  2.315388e-05
## comp 72  2.155125e-05
## comp 73  2.145416e-05
## comp 74  2.031857e-05
## comp 75  2.010007e-05
## comp 76  1.990437e-05
## comp 77  1.888317e-05
## comp 78  1.863733e-05
## comp 79  1.793241e-05
## comp 80  1.738507e-05
## comp 81  1.680371e-05
## comp 82  1.643144e-05
## comp 83  1.607243e-05
## comp 84  1.556153e-05
## comp 85  1.546692e-05
## comp 86  1.478426e-05
## comp 87  1.403356e-05
## comp 88  1.385650e-05
## comp 89  1.341951e-05
## comp 90  1.312225e-05
## comp 91  1.266819e-05
## comp 92  1.244808e-05
## comp 93  1.188441e-05
## comp 94  1.154811e-05
## comp 95  1.091053e-05
## comp 96  1.076865e-05
## comp 97  1.052484e-05
## comp 98  1.020816e-05
## comp 99  1.002131e-05
## comp 100 9.289475e-06
## comp 101 9.058276e-06
## comp 102 8.808216e-06
## comp 103 8.392901e-06
## comp 104 8.168888e-06
## comp 105 7.481678e-06
## comp 106 7.367374e-06
## comp 107 6.987918e-06
```

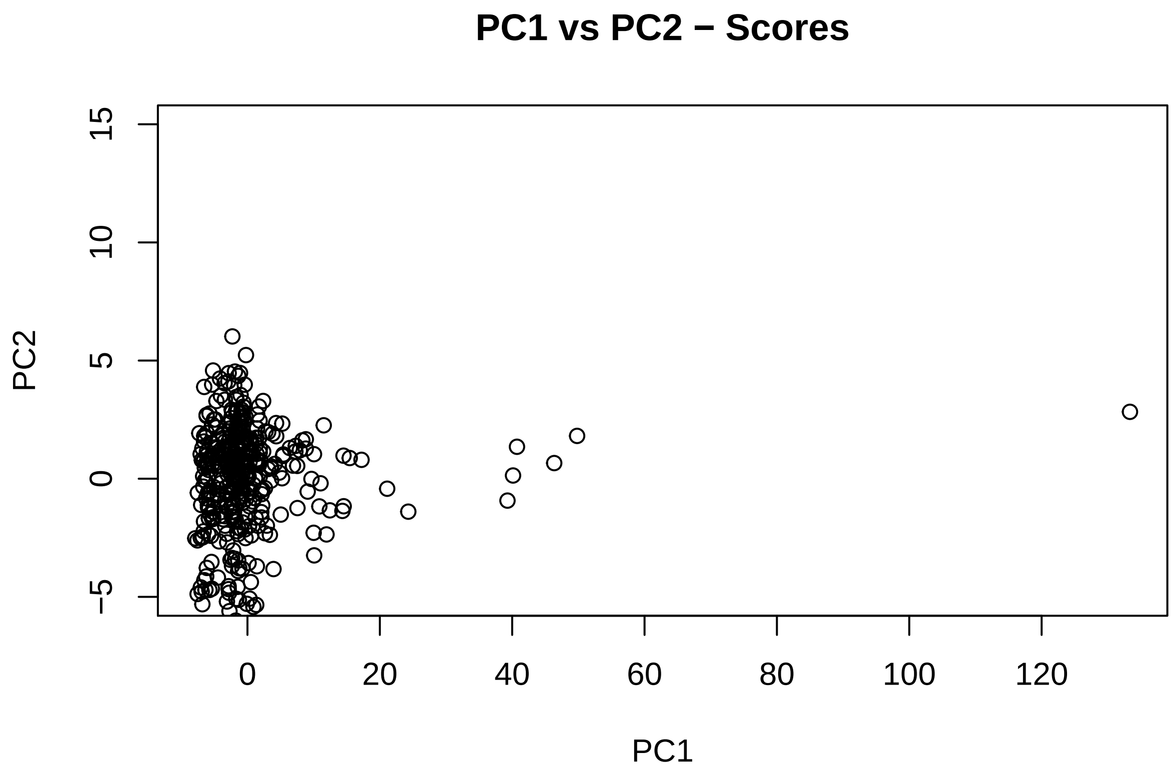
```
## comp 108 6.758358e-06
## comp 109 6.283704e-06
## comp 110 6.225106e-06
## comp 111 5.928797e-06
## comp 112 5.893652e-06
## comp 113 5.487915e-06
## comp 114 5.219927e-06
## comp 115 5.079877e-06
## comp 116 4.979149e-06
## comp 117 4.823115e-06
## comp 118 4.434055e-06
## comp 119 4.275508e-06
## comp 120 4.214351e-06
## comp 121 3.946550e-06
## comp 122 3.647069e-06
## comp 123 3.526302e-06
## comp 124 3.421603e-06
## comp 125 3.057103e-06
## comp 126 2.679962e-06
## comp 127 2.439888e-06
```

```
## The Percentage of variation:
```

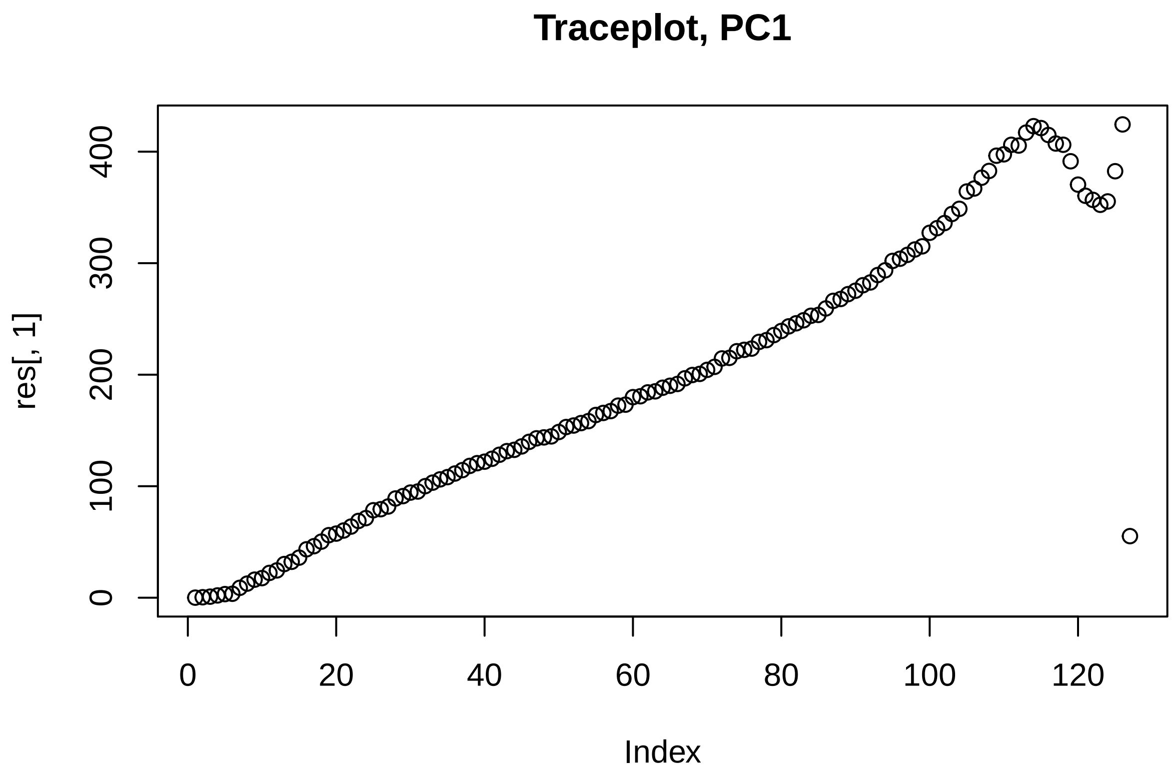
```
##                   pov
## comp 1   9.463948e+01
## comp 2   4.201272e+00
## comp 3   8.128904e-01
## comp 4   1.818837e-01
## comp 5   7.386927e-02
## comp 6   6.070142e-02
## comp 7   9.705188e-03
## comp 8   4.787005e-03
## comp 9   2.930688e-03
## comp 10  2.496247e-03
## comp 11  1.557006e-03
## comp 12  1.281430e-03
## comp 13  8.410668e-04
## comp 14  7.362455e-04
## comp 15  5.892073e-04
## comp 16  4.040294e-04
## comp 17  3.575309e-04
## comp 18  3.029203e-04
## comp 19  2.448564e-04
## comp 20  2.341039e-04
## comp 21  2.137993e-04
## comp 22  1.909693e-04
## comp 23  1.641745e-04
## comp 24  1.528675e-04
## comp 25  1.266589e-04
## comp 26  1.238573e-04
## comp 27  1.167357e-04
## comp 28  9.866302e-05
## comp 29  9.421047e-05
## comp 30  8.807665e-05
```

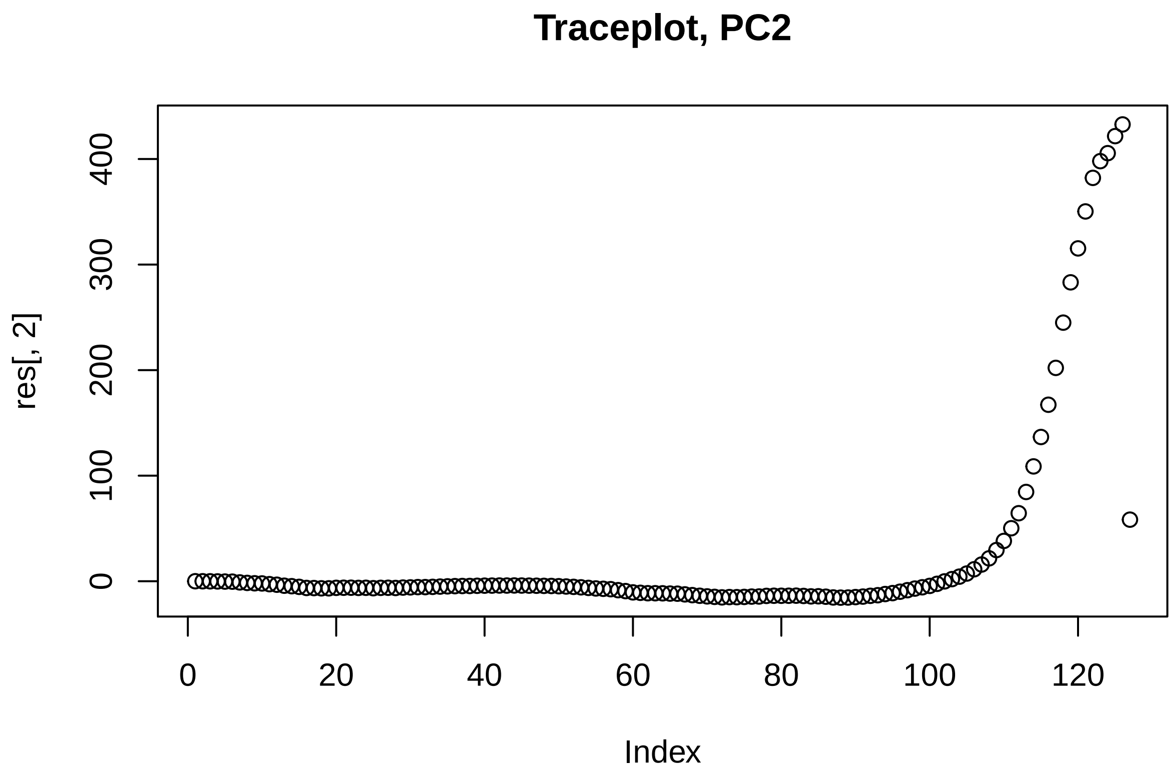
```
## comp 31  8.635998e-05
## comp 32  7.829575e-05
## comp 33  7.362445e-05
## comp 34  6.967443e-05
## comp 35  6.710046e-05
## comp 36  6.331028e-05
## comp 37  5.999448e-05
## comp 38  5.622040e-05
## comp 39  5.397118e-05
## comp 40  5.279521e-05
## comp 41  5.062022e-05
## comp 42  4.780172e-05
## comp 43  4.548233e-05
## comp 44  4.467312e-05
## comp 45  4.267865e-05
## comp 46  4.021613e-05
## comp 47  3.845844e-05
## comp 48  3.802243e-05
## comp 49  3.753131e-05
## comp 50  3.557542e-05
## comp 51  3.351672e-05
## comp 52  3.294043e-05
## comp 53  3.202114e-05
## comp 54  3.129760e-05
## comp 55  2.922299e-05
## comp 56  2.856865e-05
## comp 57  2.799467e-05
## comp 58  2.641946e-05
## comp 59  2.612997e-05
## comp 60  2.419795e-05
## comp 61  2.397982e-05
## comp 62  2.309044e-05
## comp 63  2.286231e-05
## comp 64  2.209097e-05
## comp 65  2.167912e-05
## comp 66  2.131503e-05
## comp 67  2.023024e-05
## comp 68  1.962339e-05
## comp 69  1.941391e-05
## comp 70  1.872494e-05
## comp 71  1.823140e-05
## comp 72  1.696948e-05
## comp 73  1.689304e-05
## comp 74  1.599887e-05
## comp 75  1.582683e-05
## comp 76  1.567273e-05
## comp 77  1.486864e-05
## comp 78  1.467507e-05
## comp 79  1.412001e-05
## comp 80  1.368903e-05
## comp 81  1.323127e-05
## comp 82  1.293814e-05
## comp 83  1.265546e-05
## comp 84  1.225317e-05
```

```
## comp 85  1.217868e-05
## comp 86  1.164115e-05
## comp 87  1.105005e-05
## comp 88  1.091063e-05
## comp 89  1.056654e-05
## comp 90  1.033248e-05
## comp 91  9.974949e-06
## comp 92  9.801638e-06
## comp 93  9.357807e-06
## comp 94  9.092998e-06
## comp 95  8.590965e-06
## comp 96  8.479255e-06
## comp 97  8.287278e-06
## comp 98  8.037921e-06
## comp 99  7.890797e-06
## comp 100 7.314547e-06
## comp 101 7.132501e-06
## comp 102 6.935603e-06
## comp 103 6.608584e-06
## comp 104 6.432195e-06
## comp 105 5.891085e-06
## comp 106 5.801082e-06
## comp 107 5.502298e-06
## comp 108 5.321542e-06
## comp 109 4.947798e-06
## comp 110 4.901658e-06
## comp 111 4.668344e-06
## comp 112 4.640671e-06
## comp 113 4.321193e-06
## comp 114 4.110179e-06
## comp 115 3.999903e-06
## comp 116 3.920590e-06
## comp 117 3.797728e-06
## comp 118 3.491382e-06
## comp 119 3.366542e-06
## comp 120 3.318386e-06
## comp 121 3.107520e-06
## comp 122 2.871708e-06
## comp 123 2.776616e-06
## comp 124 2.694176e-06
## comp 125 2.407168e-06
## comp 126 2.110207e-06
## comp 127 1.921172e-06
```

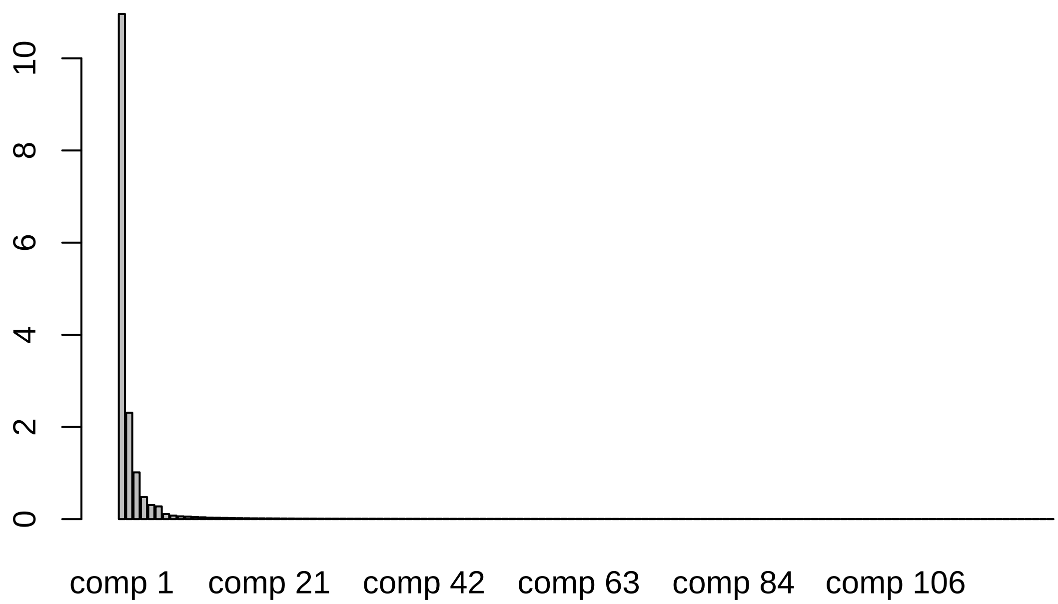


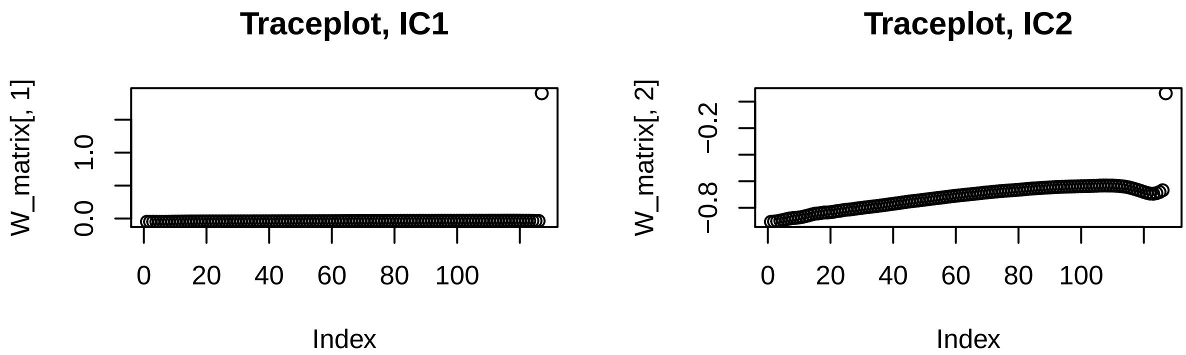
The First two PC function contributes 99%(PC1=94.6,PC2=4.33) of the total variance. Some Most initial points accumuated in the left and there is one point which looks like a outlier.


Traceplot, PC1
res[, 1]
0
100
200
300
400
0
20
40
60
80
100
120
Index

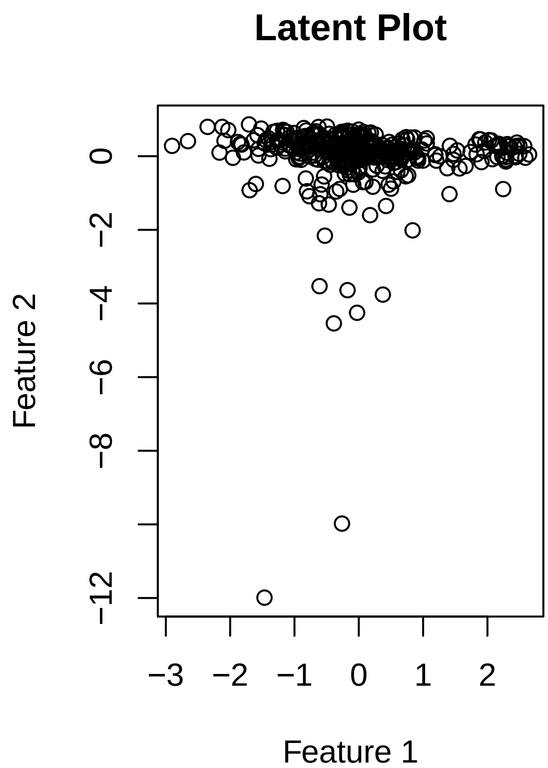


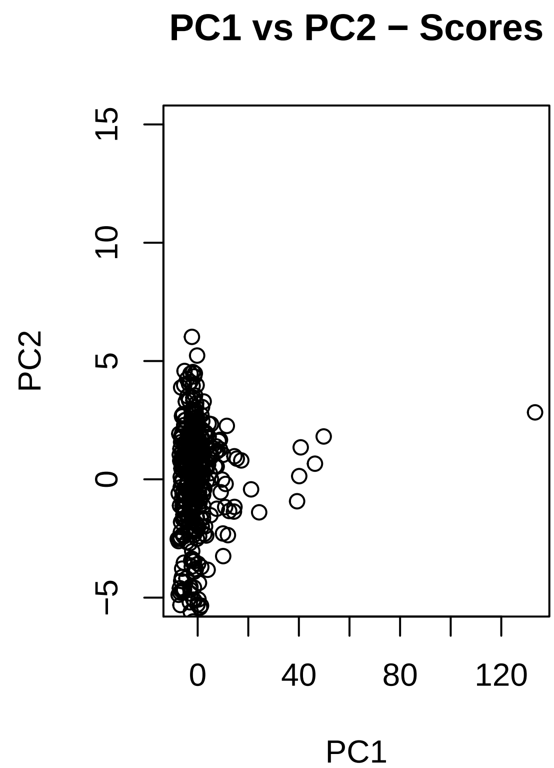
The first principle has a influence from the variable but the second principle component has influence only with parameters over 100 to 126.


10
8
6
4
2
0
comp 1
comp 21
comp 42
comp 63
comp 84
comp 106



Traceplot, IC1
W_matrix[, 1]
0.0
1.0
0
20
40
60
80
100
Index
Traceplot, IC2
W_matrix[, 2]
−0.8
−0.2
0
20
40
60
80
100
Index





The latent plot looks same as the score plot but with the opposite image. The latent feature is (-1) of pca feature. That means they are high negatively correlated.
Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library(readxl)
library(tree)
library(rpart)
library(rpart.plot)
library(e1071)
library(FactoMineR)
library(boot)
library(dplyr)
library(ggplot2)
library(fastICA)
sf2 <- read_excel(file.choose())
sf1 <- read.csv2(file.choose(),header = T,sep = ";",quote = "\"",fill=T)
sf3 <- read.csv2(file.choose(),header = T)
n=dim(sf2)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=sf2[id,]
id1=setdiff(1:n,id)
set.seed(12345)
id2=sample(id1, floor(n*0.25))
valid=sf2[id2,]
```

```
id3=setdiff(id1,id2)
test=sf2[id3,]
sf2.model_d <- tree(as.factor(good_bad)~.,data = train,split = c("deviance"))
plot(sf2.model_d)
text(sf2.model_d,cex=0.75)
sf2.train_d <- predict(sf2.model_d, train,type = "class")
train_model_pred_d <- sf2.train_d != as.factor(train$good_bad)
train_confusion_matrix_d <- table(train_model_pred_d,train$good_bad)
cat("The confusion matrix for train data using deviance method is:\n")
train_confusion_matrix_d
miscalculation_rate_train_d <- ((sum(diag(train_confusion_matrix_d))/sum(train_confusion_matrix_d)))
cat("The miscalculation rate for train data using deviance method is:",miscalculation_rate_train_d,"\n"
sf2.test_d <- predict(sf2.model_d, test,type = "class")
test_model_pred_d <- sf2.test_d != as.factor(test$good_bad)
test_confusion_matrix_d <- table(test_model_pred_d,test$good_bad)
cat("The confusion matrix for test data using deviance method is:\n")
test_confusion_matrix_d
miscalculation_rate_test_d <- ((sum(diag(test_confusion_matrix_d))/sum(train_confusion_matrix_d)))
cat("The miscalculation rate for test data using deviance method is:",miscalculation_rate_test_d,"\n")
sf2.model_g <- tree(as.factor(good_bad)~.,data = train,split = c("gini"))
plot(sf2.model_g)
text(sf2.model_g,cex=0.75)
sf2.train_g <- predict(sf2.model_g, train,type = "class")
train_model_pred_g <- sf2.train_g != as.factor(train$good_bad)
train_confusion_matrix_g <- table(train_model_pred_g,train$good_bad)
cat("The confusion matrix for train data using gini index is:\n")
train_confusion_matrix_g
miscalculation_rate_train_g <- ((sum(diag(train_confusion_matrix_g))/sum(train_confusion_matrix_g)))
cat("The miscalculation rate for train data using gini index is:",miscalculation_rate_train_g,"\n")
sf2.test_g <- predict(sf2.model_g, test,type = "class")
test_model_pred_g <- sf2.test_g != as.factor(test$good_bad)
test_confusion_matrix_g <- table(test_model_pred_g,test$good_bad)
cat("The confusion matrix for test data using gini index is:\n")
test_confusion_matrix_g
miscalculation_rate_test_g <- ((sum(diag(test_confusion_matrix_g))/sum(train_confusion_matrix_g)))
cat("The miscalculation rate for test data using gini index is:",miscalculation_rate_test_g,"\n")
optimal_train <- prune.tree(sf2.model_d)
optimal_valid <- prune.tree(sf2.model_d,newdata = valid)
optimal_leaf <-  optimal_valid$size[which.min(optimal_valid$dev)]
cat("The optimal leaf value is:",optimal_leaf,"\n")
optimal_tree  <- prune.tree(sf2.model_d,best=optimal_leaf)
nodes_optimum <- as.numeric(rownames(optimal_tree$frame))
cat("The optimum number of nodes:",nodes_optimum,"\n")
plot(optimal_tree )
text(optimal_tree )
title(main="Optimal Tree")
Naives_Bayes <- naiveBayes(as.factor(good_bad)~., data=train,laplace = 1,na.omit(NA))
Y_pred <- predict(Naives_Bayes, test, type="class")
prob <- Y_pred != test$good_bad
confusion_matrix_naive <- table(prob, test$good_bad,dnn=c("Prediction","Actual"))
cat("The confusion matrix for Naives Bayes is:\n")
confusion_matrix_naive
miscalculation_rate_naive <- ((sum(diag(confusion_matrix_naive))/sum(confusion_matrix_naive)))
```

```
cat("The miscalculation rate for naives bayes is:",miscalculation_rate_naive,"\n")
pie_seq = seq(0.05, 0.95, 0.05)
nayes_res = matrix(nrow = 0, ncol = 3)
optim_res = matrix(nrow = 0, ncol = 3)
for(i in pie_seq){
  prediction = as.data.frame(predict(Naives_Bayes, test, type = "raw"))
  prediction$res = ifelse(prediction$good > i, "good", "bad")
  miscalsification = sum(prediction$res == test$good_bad)/(nrow(prediction))
  m = (test$good_bad == "good")*1
  n = (prediction$res == "good")*1
  tp = sum(m*n)
  false_positive = abs(sum(n)-tp)/sum(abs(m-1))
  true_positive = tp/(sum(m))
  nayes_res = rbind(nayes_res, c(miscalsification, true_positive, false_positive))

  prediction = as.data.frame(predict(optimal_tree,test))
  prediction$res = ifelse(prediction$good>i, "good", "bad")
  miscalsification = sum(prediction$res == test$good_bad)/(nrow(prediction))
  m= (test$good_bad == "good")*1
  n = (prediction$res == "good")*1
  tp = sum(m*n)
  false_positive = (abs(sum(n)-tp))/sum(abs(m-1))
  true_positive = tp/(sum(m))
  optim_res = rbind(optim_res, c(miscalsification, true_positive, false_positive))
}

nayes_res = as.data.frame(nayes_res)
colnames(nayes_res) = c("MiscRate", "TP", "FP")
optim_res = as.data.frame(optim_res)
colnames(optim_res) = c("MiscRate", "TP", "FP")

ggplot() + geom_line(data=nayes_res,aes(x=FP,y=TP,color="red")) +
  geom_line(data=optim_res,aes(x=FP,y=TP,color="blue"))+ scale_color_discrete(name="Model",labels=c("Na
  geom_abline(intercept=0,slope=1)+
  xlab("FPR")+ylab("TPR")+ggtitle("ROC curve between Naive Bayes and Optimal Tree")
sf1 %>% arrange(MET)
plot(sf1$MET,sf1$EX)
sf1.lte <- tree(EX~MET,data = sf1,control = tree.control(48,minsize=2))
set.seed(12345)
plot(sf1.lte)
text(sf1.lte)
sf1.cv <- cv.tree(sf1.lte)
plot(sf1.cv$size,sf1.cv$dev, type="b", main="Deviance of fitted tree Vs tree size",
     xlab="Size",ylab="Deviance")
reg_tree <- prune.tree(sf1.lte,best=3)
plot(reg_tree)
text(reg_tree,pretty=1)
Y_pred <- predict(reg_tree,sf1)
residual <- sf1$EX - Y_pred

hist(residual,main=c("Residuals of regression tree"),
     xlab="residual")
plot(sf1$MET,Y_pred,col="red",ylim=c(240,400),main="Fitted values and original values - Regression Tree
```

```
     ylab="Expenditures",
     xlab="MET")
points(sf1$MET,sf1$EX,pch="*")
legend(x=20,y=400,c("original values","fitted values"),
       pch=c("*","o"),
       col=c("black","red"))
set.seed(12345)
f1 <- function(data,ind){
  sf_tr <- data[ind,]
  res <- tree(EX ~ MET,sf_tr, control = tree.control(dim(sf_tr)[1],minsize=2))
  best_tree <- prune.tree(res,best=3)
  Y_predictions <- predict(best_tree, newdata=sf1)
  return(Y_predictions)
}
non_para_boot_obj <- boot(sf1,f1, R=1000)
confidence_envel <- envelope(non_para_boot_obj)

plot(sf1$MET,Y_pred,col="red",ylim=c(150,500),main="Best tree - 95% non-parametric bootstrap confidence
     ylab="EX",
     xlab="MET")
points(sf1$MET,sf1$EX,pch="*")
points(sf1$MET,confidence_envel$point[2,], type="o", col="green")
points(sf1$MET,confidence_envel$point[1,], type="o", col="blue")

legend(x=20,y=500,c("original values","fitted values","confidence intervals1","confidence intervals2"),
       pch=c("*","o",NA,NA),lwd=1,lty=c(NA,NA,1,1),
       col=c("black","red","blue","green"))
ran_arg <- function(sf_tr,mle){
  data = data.frame(MET = sf_tr$MET, EX = sf_tr$EX)
  n = length(data$EX)
  data$EX = rnorm(n,predict(mle, newdata=data),
                  sd(sf1$EX-predict(mle, newdata=data)))
  return(data)
}

f2 = function(sf_tr){
  res <- tree(EX ~ MET,sf_tr, control = tree.control(dim(sf_tr)[1],minsize=2))
  best_tree <- prune.tree(res,best=3)
  Y_predictions <- predict(best_tree, newdata=sf1)
  return(Y_predictions)
}

f3 = function(sf_tr){
  res <- tree(EX ~ MET,sf_tr, control = tree.control(dim(sf_tr)[1],minsize=2))
  best_tree <- prune.tree(res,best=3)
  Y_predictions <- rnorm(dim(sf1)[1],predict(best_tree, newdata=sf1),sd(residual))
  return(Y_predictions)
}

set.seed(12345)
para_boot_obj1 <- boot(sf1,statistic = f2, R=1000,mle=reg_tree,ran.gen=ran_arg,sim="parametric")
confidence_envel1 <- envelope(para_boot_obj1)
set.seed(12345)
```

```
para_boot_obj2 <- boot(sf1,statistic =  f3, R=1000,mle=reg_tree,ran.gen=ran_arg,sim="parametric")
confidence_envel2 <- envelope(para_boot_obj2)

plot(sf1$MET,Y_pred,col="red",ylim=c(150,500),main=c("Regression tree - 95% parametric bootstrap confide
     ylab="EX",
     xlab="MET")
points(sf1$MET,sf1$EX,pch="*")
points(sf1$MET,confidence_envel1$point[2,], type="l", col="green")
points(sf1$MET,confidence_envel1$point[1,], type="l", col="green")
points(sf1$MET,confidence_envel2$point[2,], type="l", col="yellow")
points(sf1$MET,confidence_envel2$point[1,], type="l", col="yellow")
legend(x=20,y=550,c("original values","fitted values","confidence bands","prediction bands"),
       pch=c("*","o",NA,NA),lwd=1,lty=c(NA,NA,1,1),
       col=c("black","red","green","yellow"))
set.seed(12345)
pca_matrix <-  PCA(sf3,graph = FALSE)
lambda = pca_matrix$eig[,1]
#eigenvalues
cat("The lambda values are:\n")
as.data.frame(lambda)
#proportion of variation
pov <- pca_matrix$eig[,2]
cat("The Percentage of variation:\n")
as.data.frame(pov)
plot(pca_matrix$ind$coord[,1], pca_matrix$ind$coord[,2], ylim=c(-5,15), xlab = "PC1", ylab = "PC2", mai
res <- (pca_matrix$var$coord/sqrt(pca_matrix$eig[,1]))
pca_1 <- plot(res[,1], main="Traceplot, PC1")
pca_2 <- plot(res[,2],main="Traceplot, PC2")
barplot(sqrt(pca_matrix$eig[,1]))
set.seed(12345)
Ica <- fastICA(sf3,2)
W_matrix <- Ica$K %*% Ica$W
par(mfrow = c(2,2))
ica_1 <- plot(W_matrix[,1], main="Traceplot, IC1")
ica_2 <- plot(W_matrix[,2], main="Traceplot, IC2")
maximum_Likelihood <- solve(Ica$W)
X_tra <- Ica$X %*% W_matrix
D <- X_tra %*% maximum_Likelihood
par(mfrow = c(1,2))
plot(D, main = "Latent Plot", xlab = "Feature 1", ylab = "Feature 2")
plot(pca_matrix$ind$coord[,1], pca_matrix$ind$coord[,2], ylim=c(-5,15), main = "PC1 vs PC2 - Scores", x
```