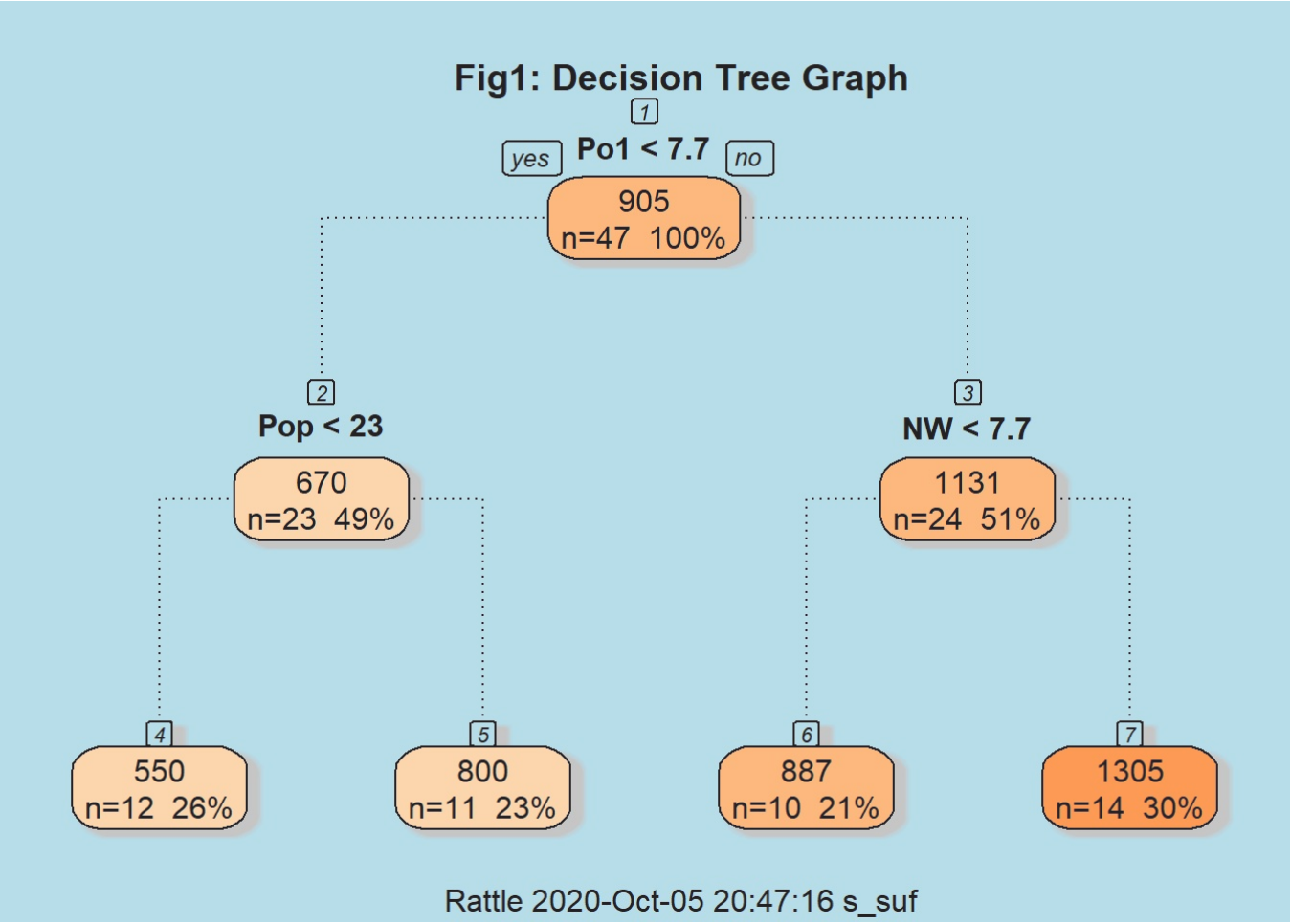
DEMO 7 - CART, RANDOM FORESTS, AND LOGISTIC REGRESSION

QUESTION 1.1 - REGRESSION TREE MODEL ON CRIME DATA

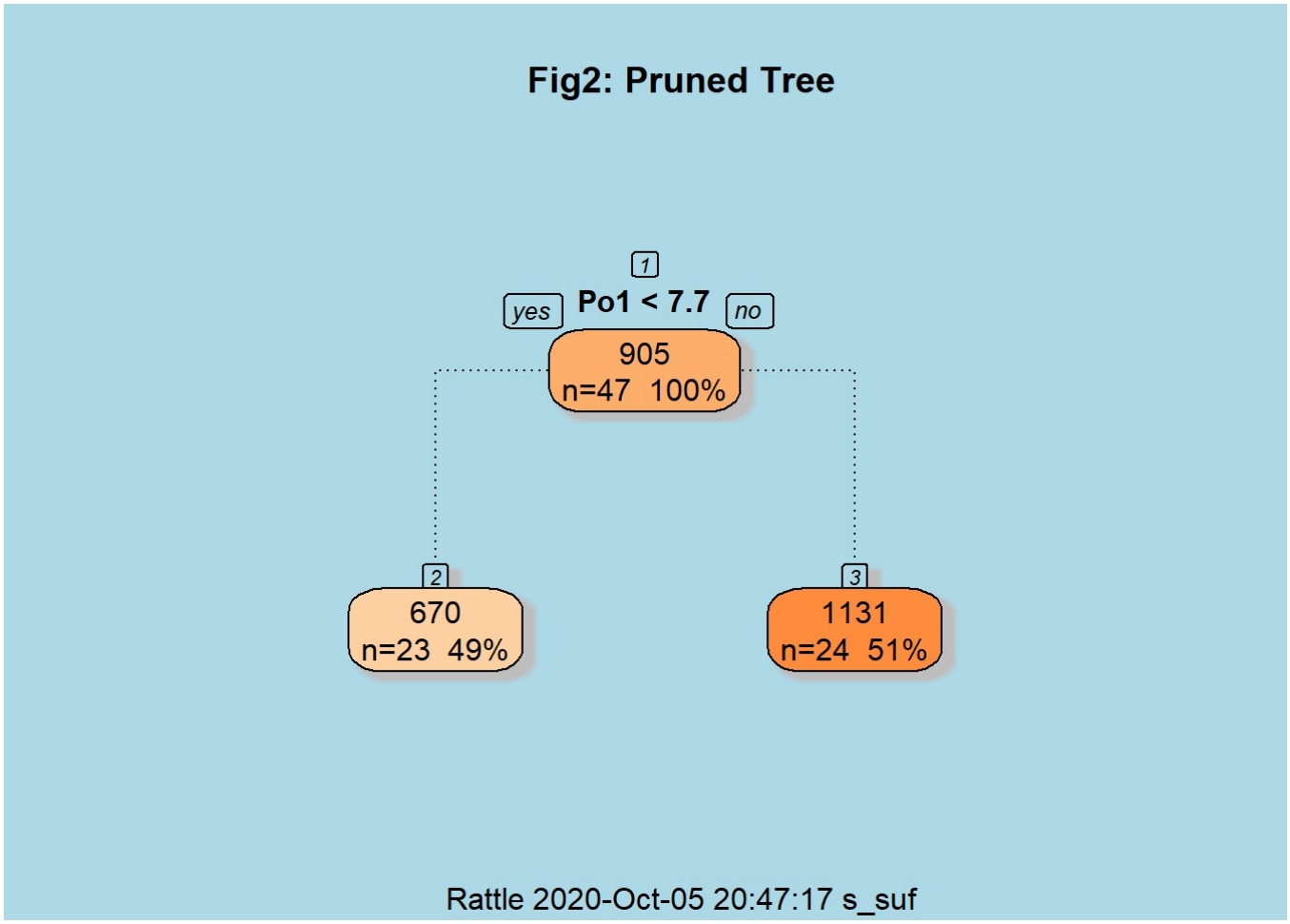
Step-By-Step Analysis ~

1 - The first part involved using the ANOVA model to construct the tree based on predictor set (P01, POP, NW) based on importance. We get the following tree with 4 leaf nodes and two branching points -



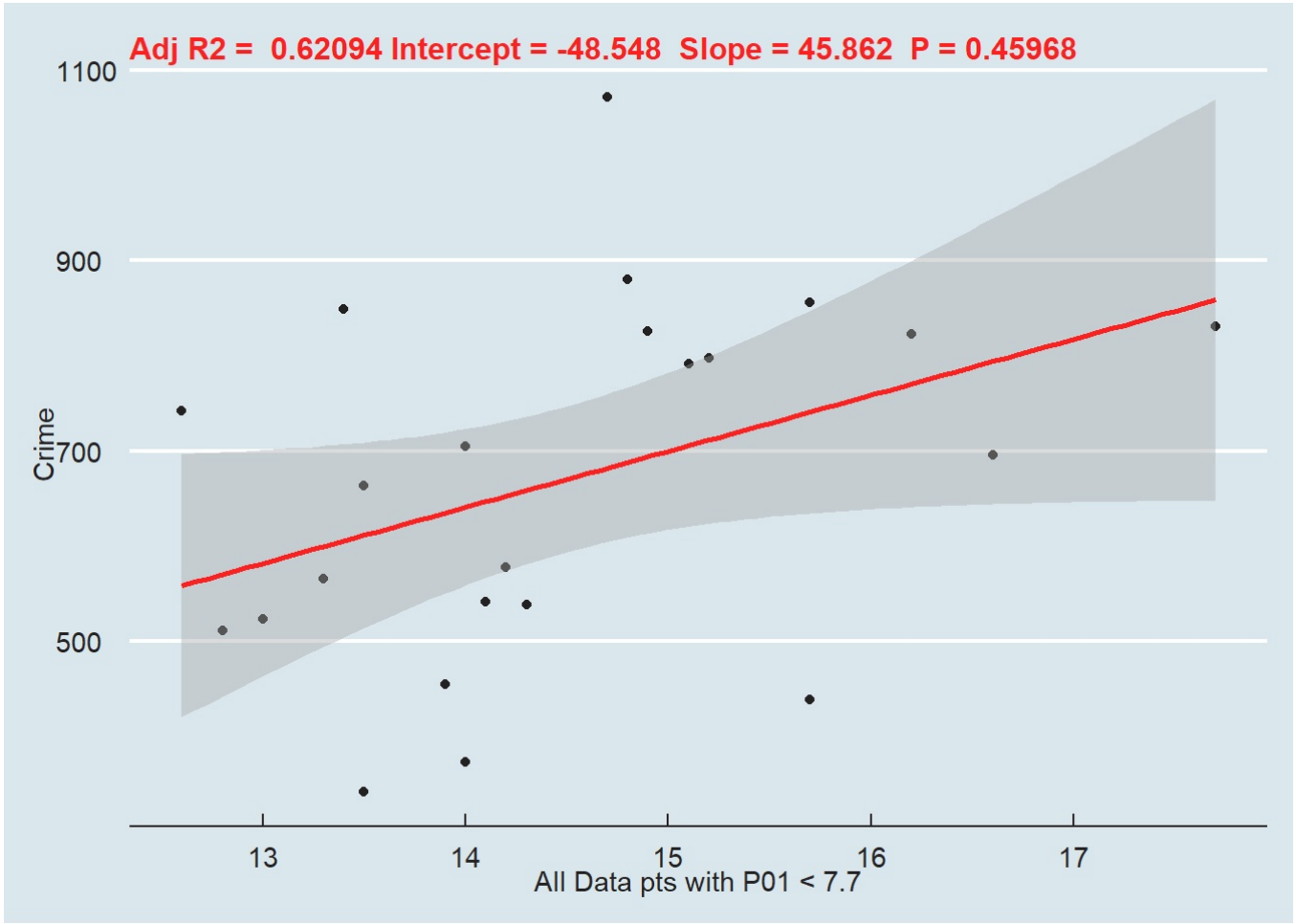
2 - We now use cross-validated training on our data. We find the that we get the best results with a data split of around 0.95% with minimum error.

3 - We then build out pruned tree with the lowest value of "CP" and we get the following tree constrained to single split -

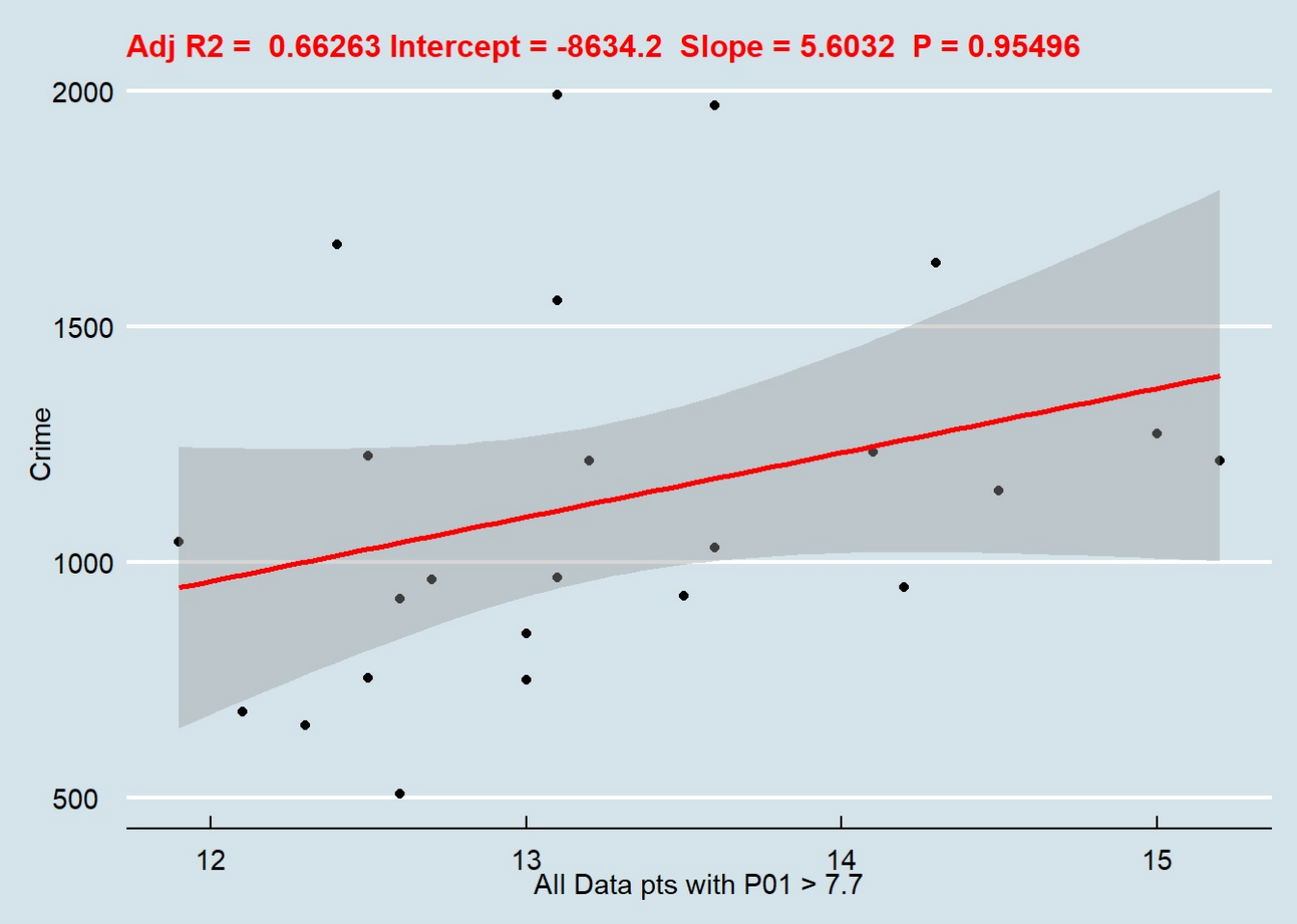


4 - We now run linear regression model on this branching as shown in it.

FOR P0 < 7.7

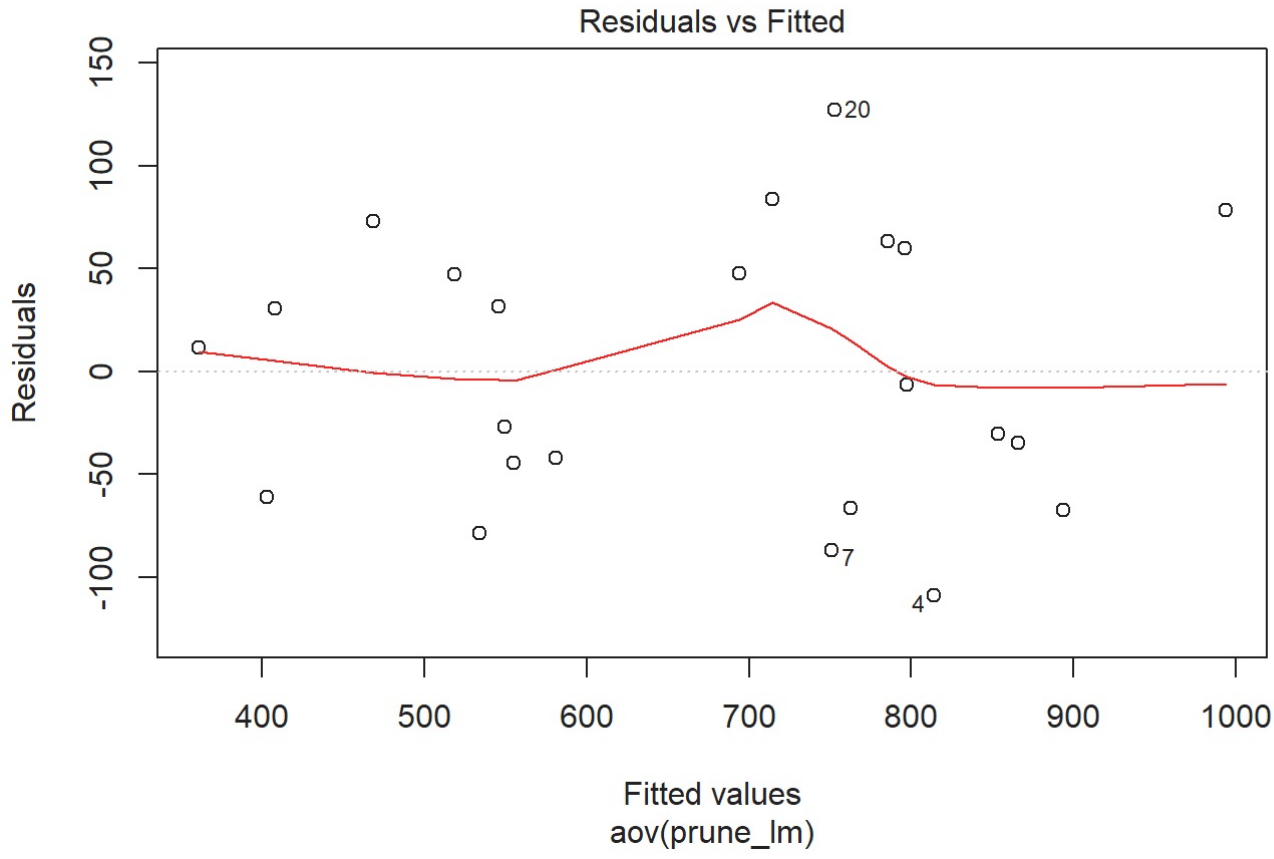


FOR P0 > 7.7

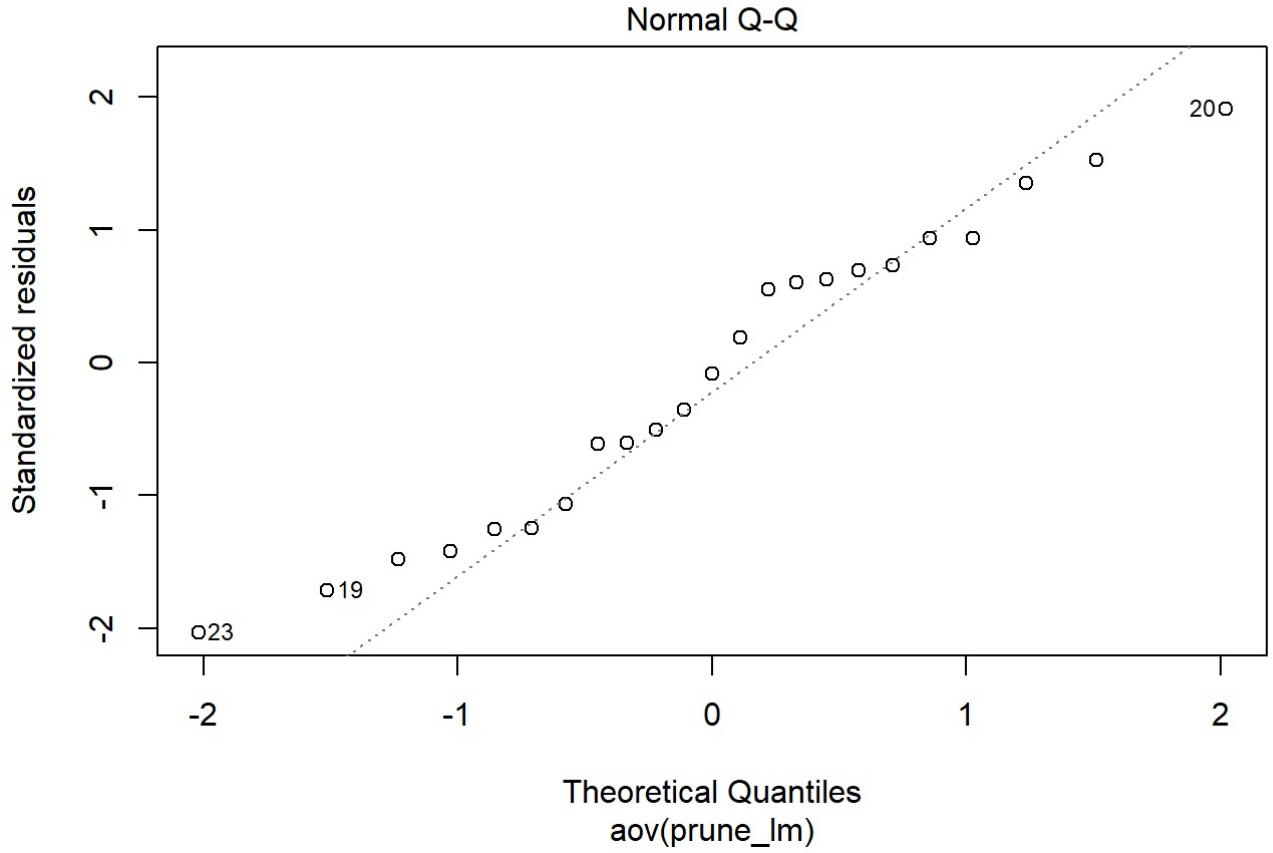


5 - now we do our regular variance and residual checks -

Residuals Vs Fitted -



Normal Q-Q -



AND THE PLOTS FOR P01 > 7.7 ARE SIMILAR TO P01 <7.7.

6 - We derive that using P01 as a predictor for branching gives an unsatisfactory Q-Q plot despite the good R-squared values, hence we move to other sets of predictors based on two-leaf regression analysis, and we derive these set of predictors - (Ed, Pop, Prob, Time) with a 90% confidence interval.

For the first leaf ~

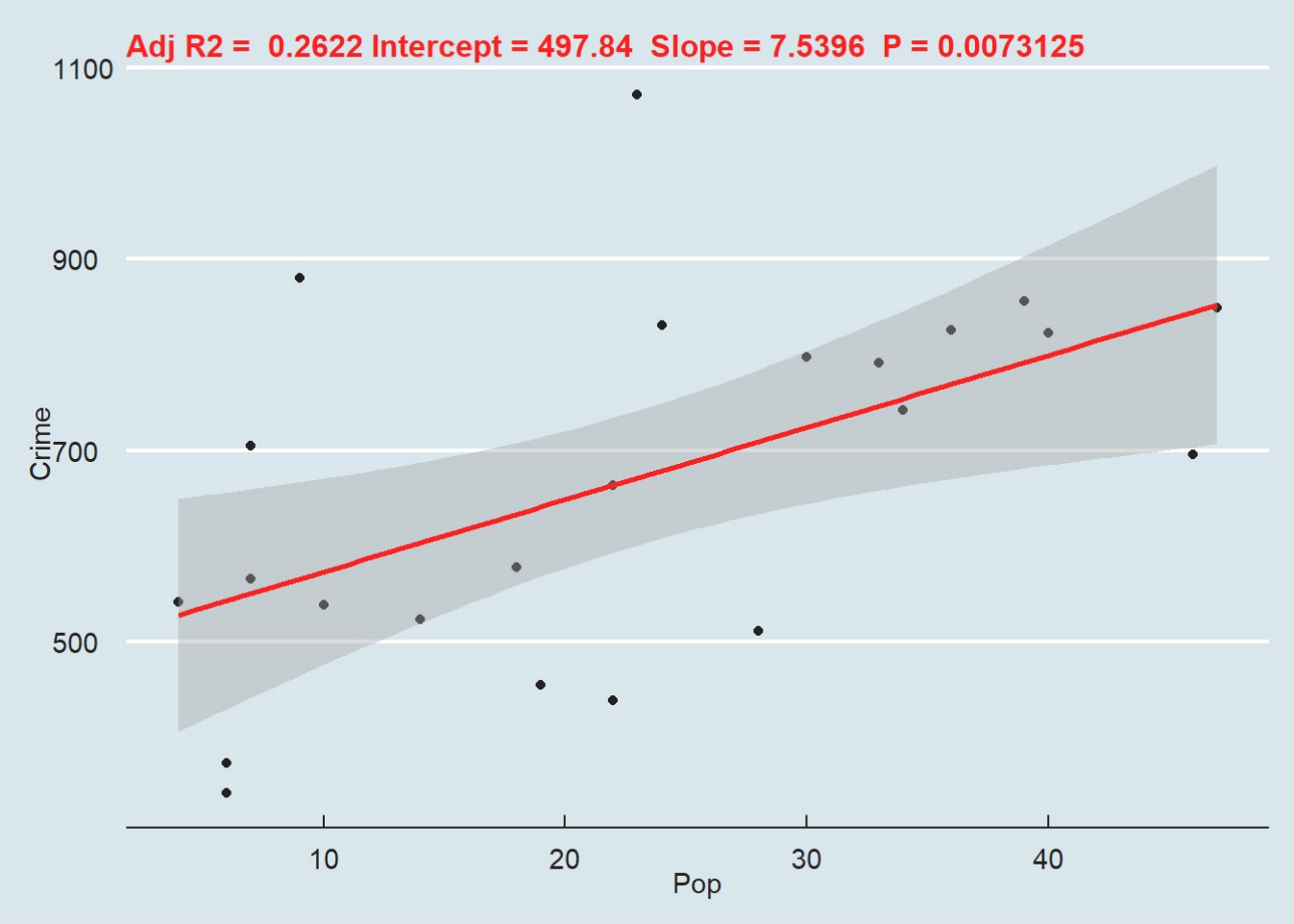
```
Call:
lm(formula = Crime ~ Ed + Pop + Prob + Time, data = leaf.2)

Residuals:
    Min      1Q  Median      3Q     Max
-206.35  -90.22   -7.59   59.64  357.11

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   819.960    515.112   1.592   0.1288
Ed              9.499     34.869   0.272   0.7884
Pop            11.395      3.229   3.529   0.0024
Prob        -3164.075   2095.755  -1.510   0.1485
Time          -12.130      6.830  -1.776   0.0927

Residual standard error: 154.5 on 18 degrees of freedom
Multiple R-squared:  0.4485, Adjusted R-squared:  0.3259
F-statistic: 3.659 on 4 and 18 DF,  p-value: 0.02379
```

7 - We then narrow it down to a 99% confidence interval and only take Pop -

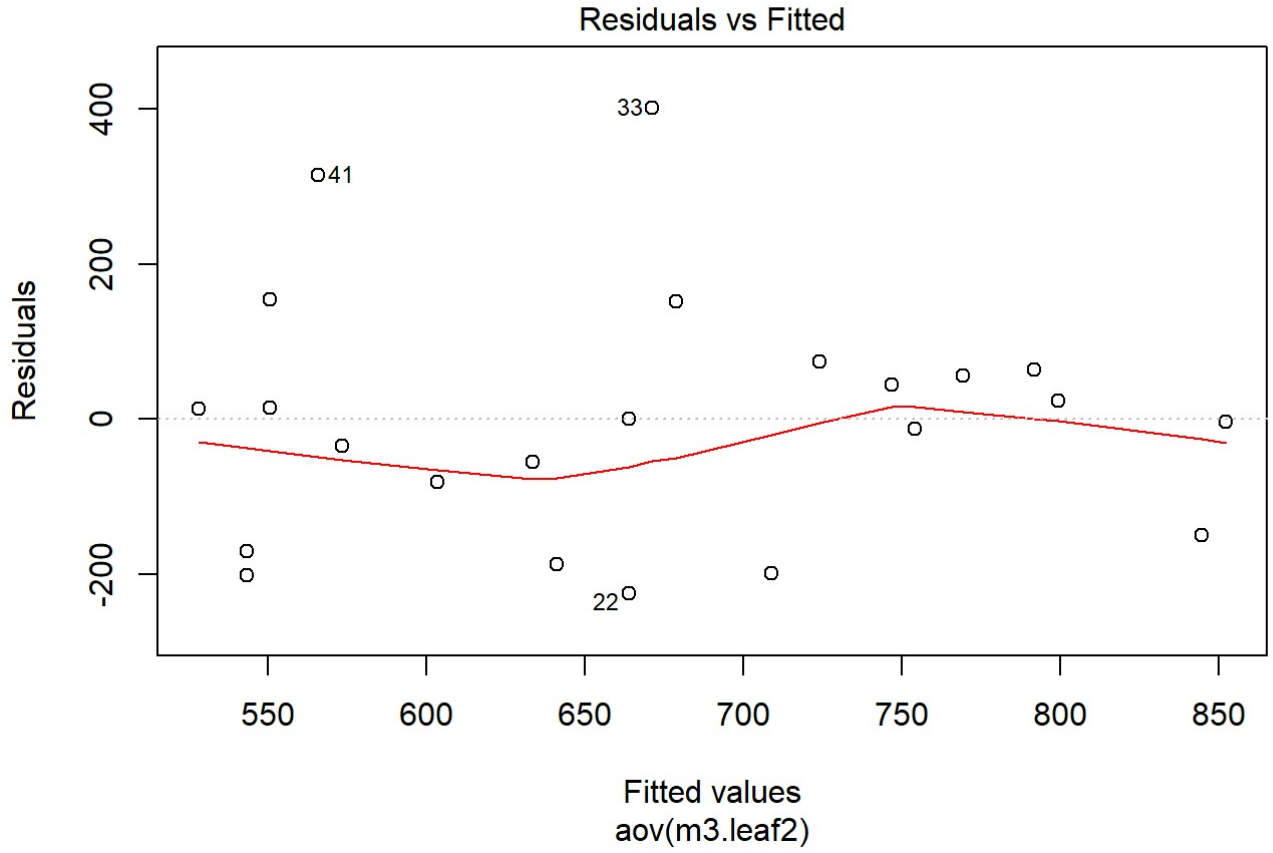
```
Call:
lm(formula = Crime ~ Pop, data = leaf.2)

Residuals:
    Min      1Q  Median      3Q     Max
-224.71 -114.53    0.29   60.43  400.75

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  497.836     66.949   7.436  2.6e-07 ***
Pop            7.540      2.539   2.970  0.00731 **
```

```
Residual standard error: 161.7 on 21 degrees of freedom
Multiple R-squared:  0.2957, Adjusted R-squared:  0.2622
F-statistic: 8.818 on 1 and 21 DF,  p-value: 0.007313
```

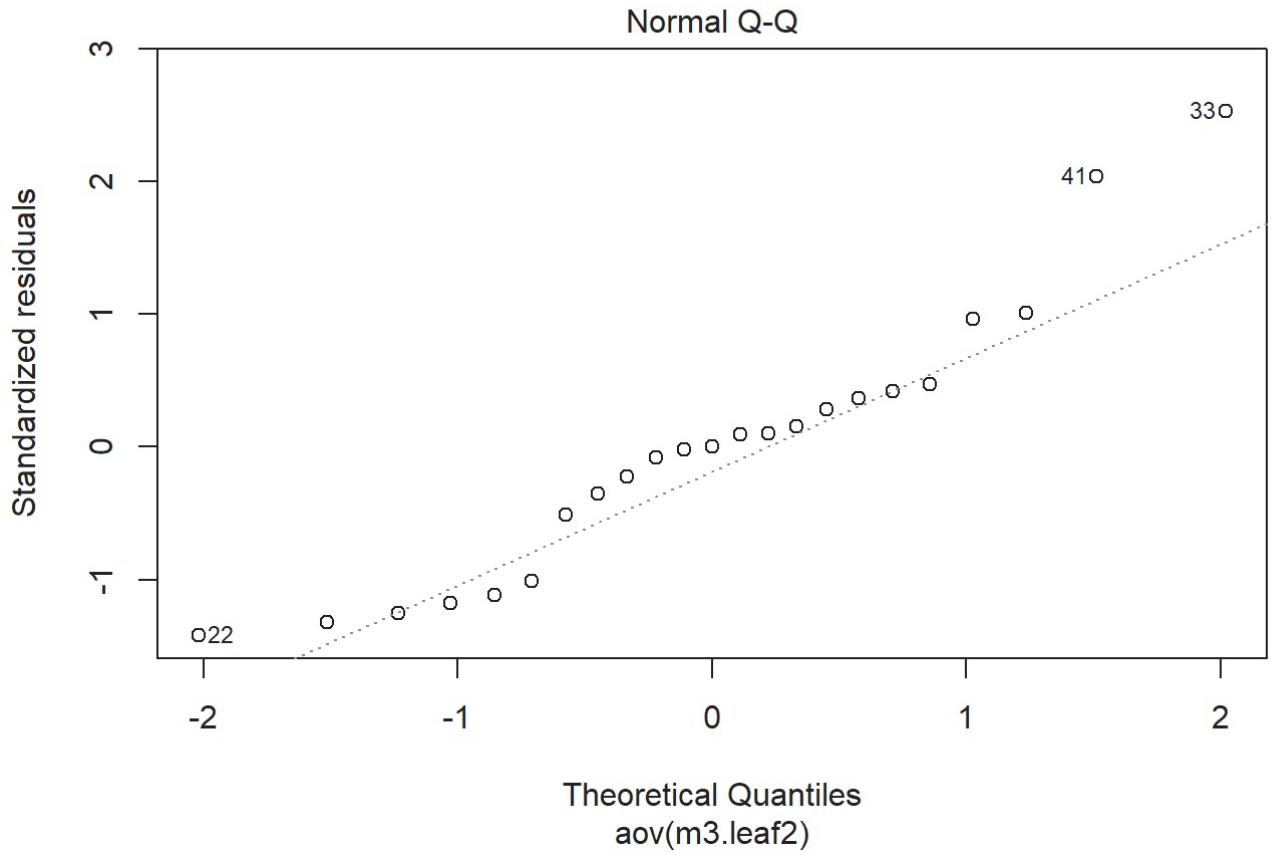
And we get this linear regression model -



Residual VS Fitted -



Normal Q-Q -



For the second leaf ~

We take a 95% confidence interval at use the predictor Ineq -

```
Call:
```

```
lm(formula = Crime ~ Ineq, data = leaf.3)

Residuals:
    Min      1Q  Median      3Q     Max
-612.67 -254.25  -88.83  136.25  918.77

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   702.70     495.52   1.418    0.170
Ineq           24.44      27.91   0.876    0.391

Residual standard error: 397.9 on 22 degrees of freedom
Multiple R-squared:  0.03368,    Adjusted R-squared:  -0.01024
F-statistic: 0.7668 on 1 and 22 DF,  p-value: 0.3907
```

We don't get good enough results for P0 > 7.7 branch and hence we move to assess the random forest model for this dataset.

QUESTION 1.2 - RANDOM FOREST MODEL ON CRIME DATA

Step-By-Step Analysis ~

1 - Using random forest on the unpruned tree and pruned tree that we had -

Unpruned ~

```
[1] "R-Squared of Random Forest Model (All factors & Data Pts) = 0.402"
```

Which is good value for R-squared!

Pruned ~

FOR P01 < 7.7 -

```
23 samples
15 predictors

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 21, 21, 21, 21, 20, 20, ...
Resampling results across tuning parameters:

  mtry  RMSE      Rsquared   MAE
   2    148.6439  0.7932877  132.1649
   8    146.8366  0.8270646  127.4077
  15    148.8677  0.8379726  127.7360

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 8.
```

FOR P01 < 7.7 -

```
24 samples
15 predictors

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 22, 22, 21, 22, 21, 21, ...
Resampling results across tuning parameters:

  mtry  RMSE      Rsquared   MAE
   2    352.3433  0.8550967  307.8084
   8    368.5349  0.8468138  321.6500
  15    390.8036  0.8471077  339.9812

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 2.
```

2 - Check the RMSE values for both branches. The R-squared values indicate a better model than the regression tree anova model based on the sets of predictors we took.

QUESTION 1 - COMBINED ANALYSIS

1. While the regression tree is a good root-creating tool, the R-squared value tells us exactly which branch was performing badly (P01 > 7.7) and what predictors were good ones, with such little data, the model might have overfit.

2. On the other hand, random forest auto-divided the data AND the branching which may have helped with the overfitting, but the problem was that now we don't know exactly which predictors lead to the better result.

This is often the problem with selcting an explainable versus a multiplexed model, and to get the best of both, often times novel algorithms need to be developed that track and explain every combination, but that requires high computation power, as well as proper domain expertise for intermediate results. Random forest function alone cannot give much intel on what to do next in action based on these findings.

CODE FOR REGRESSION TREE

In [ ]:
```
#anova model - unpruned
df <- read.csv(file="uscrime.csv",stringsAsFactors = F, header=T)
set.seed(18)
m1<- rpart(Crime~ ., data = df,method="anova" )
summary(m1)
#picture of tree
par(bg = 'lightblue')
fancyRpartPlot(m1, main="Fig1: Decision Tree Graph", palettes=c("Oranges"), type=1)

#Using cross-validation
min_cp <- m1$cptable[which.min(m1$cptable[,"xerror"]),"CP"]
# pruned tree using best cp
m1_prune <- prune(m1, cp = min_cp)
# picture of pruned tree
par(bg = 'lightblue')
fancyRpartPlot(m1_prune, main="Fig2: Pruned Tree", palettes=c("Oranges"), type=1)

#linear regression model for pruned tree
df.new<-dplyr::filter((df),Po1<7.7)
df.new.greater.7.7<-dplyr::filter((df),Po1>7.7)
prune_lm = lm(Crime ~., data = df.new)
prune_lm.greater.7.7 = lm(Crime ~., data = df.new.greater.7.7)
ggplotRegression <- function (fit) {
require(ggplot2)
ggplot(fit$model, aes_string(x = names(fit$model)[2], y = names(fit$model)[1])) +
  geom_point() +
  stat_smooth(method = "lm", col = "red") +
  labs(title = paste("Adj R2 = ",signif(summary(fit)$adj.r.squared, 5),
                     "Intercept =",signif(fit$coef[[1]],5 ),
                     " Slope =",signif(fit$coef[[2]], 5),
                     " P =",signif(summary(fit)$coef[2,4], 5)))
}
ggplotRegression(prune_lm )+theme_economist()+theme(
  plot.title = element_text(color = "red", size = 12, face = "bold"))+ labs(x = "All Data pts with P01 < 7.7")

#Quality check - variance and normalized errors
res.aov.train <- aov(prune_lm , data = df.new)
summary(res.aov.train)
plot(res.aov.train, 1)
plot(res.aov.train, 2)

#Second iteration with new predictors
leaf.2 <-df[which(m1_prune$where==2),]
leaf.3<-df[which(m1_prune$where==3),]

#leaf 2
m.leaf2<-lm(Crime~., data=leaf.2)
summary(m.leaf2)
#90% confidence interval
m2.leaf2 <- lm(Crime~Ed+Pop+Prob+Time,data=leaf.2)
summary(m2.leaf2 )
#99% confidence interval
m3.leaf2  <- lm(Crime~Pop,data=leaf.2)
summary(m3.leaf2)

#regression model
ggplotRegression(m3.leaf2 )+theme_economist()+theme(
    plot.title = element_text(color = "red", size = 12, face = "bold"))
#quality check - variance and error distribution as before


#leaf 3
m.leaf3 <- lm(Crime~.,data=leaf.3)
summary(m.leaf3 )
#95% confidence interval
m2.leaf3<- lm(Crime~Ineq,data=leaf.3)
summary(m2.leaf3)
```

CODE FOR RANDOM FOREST

In [ ]:
```
#preliminary iteration without pruning
```

```
set.seed(71)
random.forest.model <- randomForest(Crime ~. , data=df,keep.forest=T, importance=TRUE,class=)
print(random.forest.model)
#importance
randomForest::importance(random.forest.model)
#plot
par(bg = 'lightblue2')
varImpPlot(random.forest.model)

#prediction and performance
ypred.RF <- predict(random.forest.model )
RSS <- sum((ypred.RF-df$Crime)^2) # the residual sum of squares
TSS <- sum((mean(df$Crime)-df$Crime)^2)#the total sum of squares
print(sprintf("R-Squared of Random Forest Model (All factors & Data Pts) = %0.3f", R.squared.RF<-1-(RSS/TSS) ))

#plot and analysis
par(bg = 'lightblue')
plot(df$Crime, scale(ypred.RF-df$Crime), pch = 19, col = "maroon1",main="Random Forest: Residual vs. Fitted")
abline(0,0)

#analysis for pruned tree p01<7.7
set.seed(71)
fit.control <- caret::trainControl(method = "repeatedcv", number = 10, repeats = 3)
rf.fit.leaf2 <- caret::train(Crime~.,
                      data = df.new,
                      method = "rf",
                      trControl = fit.control)
print(rf.fit.leaf2)

#analysis for pruned tree p01>7.7
set.seed(71)
fit.control <- caret::trainControl(method = "repeatedcv", number = 10, repeats = 3)
rf.fit.leaf3 <- caret::train(Crime~.,
                      data = df.new.greater.7.7,
                      method = "rf",
                      trControl = fit.control)
print(rf.fit.leaf3)
```

QUESTION 2 - REALISTIC LOGISTIC REGRESSION USE-CASE

In the Ed-Tech field, schools often use computational products to determine which students would need maximum focused help before their exams, based on past history of students. Here is a case, where there is an upcoming periodic test and a teacher needs to know if a student is likely to do well on that test or not, based on the following factors and a dataset of student information -

1. Average number of hours spent studying per day
2. Past assessment grades
3. Activity score in class (Weighted by recency and frequency BOTH)
4. Associated homework validation (1, if done correct, 0 if not)
5. Number of times the past records predicted failure for the student, but the student still did well!

QUESTION 3 - GERMAN CREDIT PROBLEM USING LOGISTIC REGRESSION

Step-By-Step Analysis ~

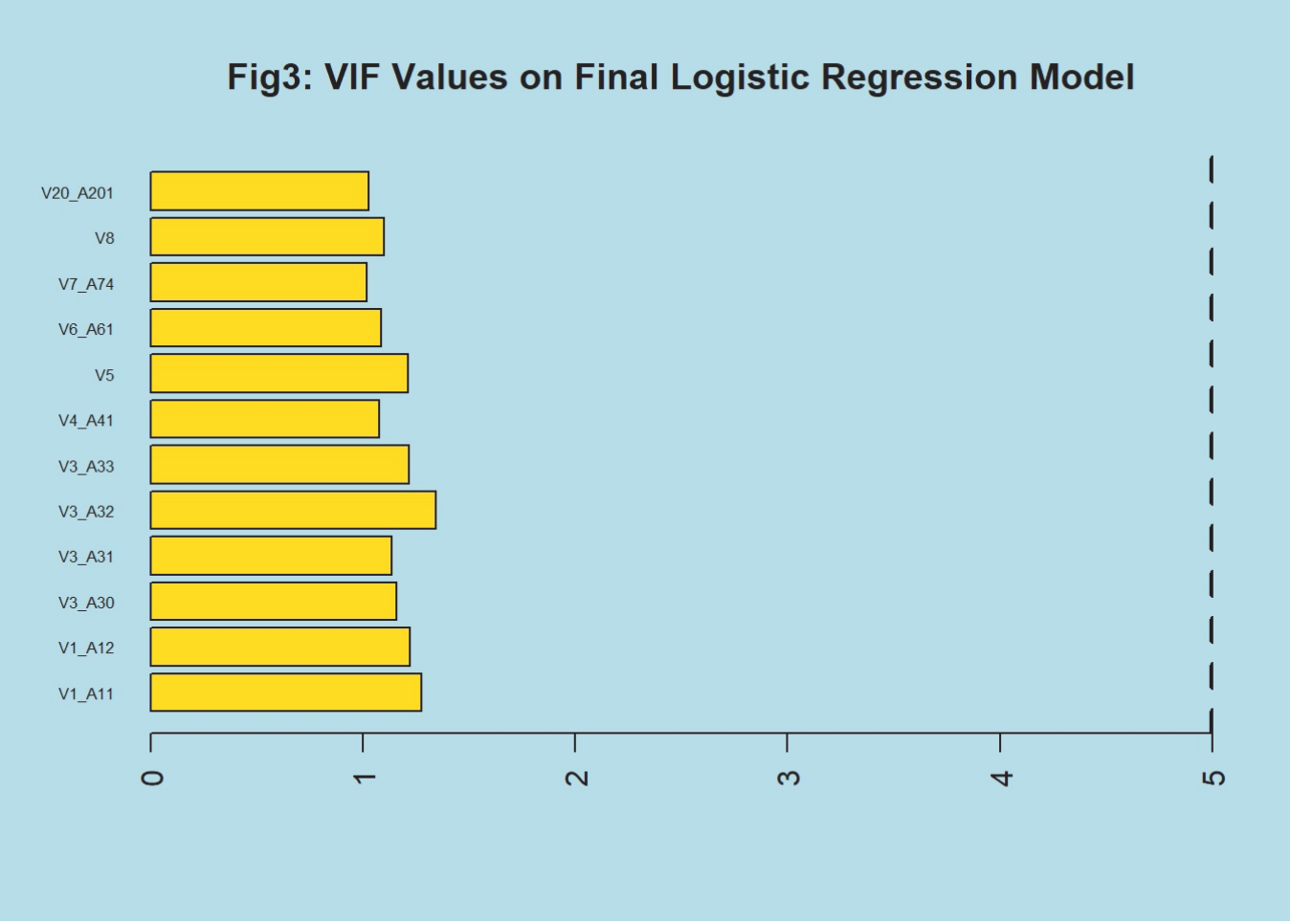
1 - First, we do one-hot encoding for categorical variables and then perform logistic regression on a training set of data (70%) and check the predictors that need to be discarded based on null hypothesis, similar to how we did it in linear regression.

The factors we select are - V1A11+V1A12+V3A30+V3A31+V3A32+V3A33+V4A41+V5+V6A61+V7A74+V8+V20A201 And the new model with pruned factors we get -

```
    Call:
    glm(formula = V21 ~ V1_A11 + V1_A12 + V3_A30 + V3_A31 + V3_A32 +
    V3_A33 + V4_A41 + V5 + V6_A61 + V7_A74 + V8 + V20_A201, family = binomial(link = "logit"), data
   = traindata)

    Deviance Residuals:
        Min       1Q   Median       3Q      Max
    -2.2163  -0.7574  -0.4430   0.8829   2.6810

    Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
    (Intercept) -6.307e+00  9.892e-01  -6.376 1.81e-10 ***
    V1_A11       1.629e+00  2.404e-01   6.775 1.24e-11 ***
    V1_A12       1.243e+00  2.352e-01   5.286 1.25e-07 ***
    V3_A30       1.792e+00  4.552e-01   3.936 8.29e-05 ***
    V3_A31       2.085e+00  4.465e-01   4.670 3.01e-06 ***
    V3_A32       1.027e+00  2.447e-01   4.195 2.73e-05 ***
    V3_A33       1.154e+00  3.599e-01   3.207  0.00134 **
```

```
V4_A41      -1.109e+00  3.878e-01  -2.860  0.00423 **
V5           1.746e-04  3.608e-05   4.839 1.31e-06 ***
V6_A61       6.363e-01  2.091e-01   3.043  0.00234 **
V7_A74      -7.138e-01  2.751e-01  -2.595  0.00947 **
V8           2.814e-01  9.051e-02   3.109  0.00188 **
V20_A201     2.095e+00  8.725e-01   2.401  0.01634 *

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 856.90  on 699  degrees of freedom
Residual deviance: 680.19  on 687  degrees of freedom
AIC: 706.19

Number of Fisher Scoring iterations: 5
```

2 - We check within these selected set of predictors for multicollinarity. Why do we do this after the pruning? It's because the combinations for the VIF function to try would be conserved thanks to our reduced set of predictors. Besides, if a predictor does not have a high probability of affecting our response, we needn't worry about it being correlated to another predictor. The results show that all VIF values are below five and there are no observable multicollinearities.



3 - Now we simply apply our new logistic regression model to our test data (30%) and evaluate the performance of it using;

1. Confusion Matrix

```
    0  1
0 188 23
1  48 41
```

2. Misclassification Error Rate

```
Error = 0.2367 - Which is low and good!
```

3. Specificity and Sensitivity

```
Sensitivity = 0.4606742
Specificity = 0.8909953
```

Observations for step 3 -

Total Accuracy = Total True Cases (positive and negative) / Total Cases = 73%

Misclassifcation error rate: 24%

Sensitivity: 46% Specificity: 89%

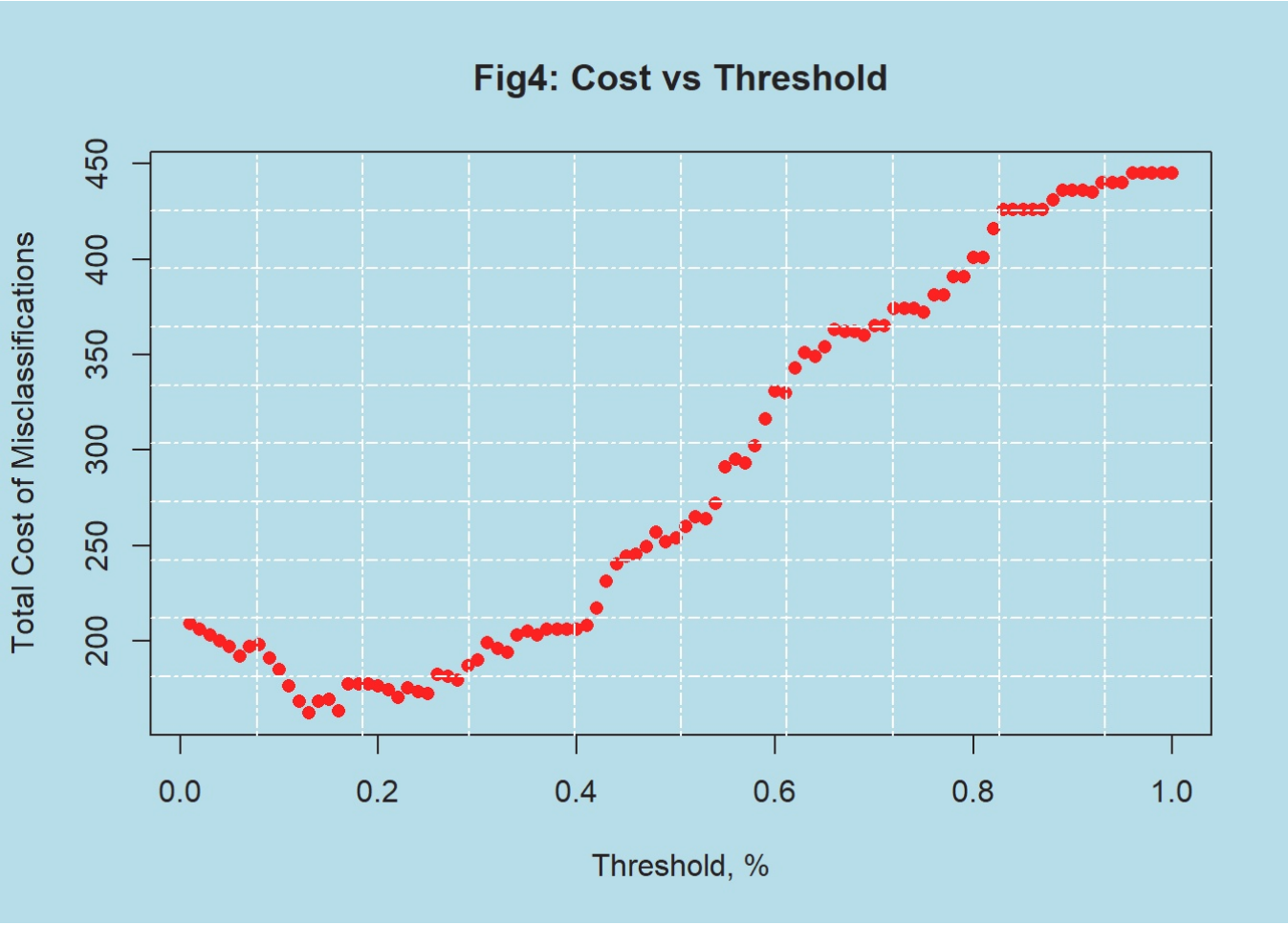
Quality of fit ~ Well! It is possible that accuracy could be improved but it may need the following -

1. A different curve-fitting technique.
2. Augmentation of data based on real-world cases that were left out.
3. Deeper study of causation associations and predictor selection not entirely based on p-value selections.

4 - Now we take into account, the next part of the question. "In this data set, they estimate that incorrectly identifying a bad customer as good, is 5 times worse than incorrectly classifying a good customer as bad." Let's create a matrix we can store this relation in. The reason for this is because we need to apply this transformation later to our dataset to get our thresholds.

```
        Predicted
Actual good bad
  good    0   1
  bad     5   0
```

5 - Now we apply this multiplicative transformation and plot our cost of misclassification against classification threshold;



And then we take the minimum value across as our threshold -

```
"Minimum Threshold due to Uneven Error Cost= 0.130"
```

6 - Conclusion

Final Model

```
             Estimate
(Intercept) -6.307e+00
V1A11        1.629e+00
V1A12        1.243e+00
V3A30        1.792e+00
V3A31        2.085e+00
V3A32        1.027e+00
V3A33        1.154e+00
V4A41       -1.109e+00
V5           1.746e-04
```

```
V6A61       6.363e-01
V7A74      -7.138e-01
V8          2.814e-01
V20A201     2.095e+00
```

Based on extra information given about the ratio of importance of false positives to false negatives we updated our classification threshold to 13%. In other words, that is a VERY conservative model to be sure that false positives do not happen except maybe very rarely. This is understandable, since banks run on credit. And if people default their loans, the bank could fail to function. At the same time, had the factor been 7-10 for false poistives, instead of 5, the bank may not have enough customers that would help them function in the long run, again resulting in failure. Such models are usually updated in real-time assessing -

1. Internal state of affairs in terms of monetary reserves and cash flows
2. External macroeconomic and sociodemographic conditions that may affect the customer base

CODE FOR LOGISTIC REGRESSION

In [ ]:

```
# Loading data
df<-read.table("germancredit.txt",sep = " ",header = FALSE)
head(df,2)

#onehot encoding
set.seed(713)
newdata <- one_hot(as.data.table(df))#one hot encoding the categorical variables
newdata$V21[newdata$V21==1]<-0
newdata$V21[newdata$V21==2]<-1

#Generate a random sample of 70% of the rows
random_row<- sample(1:nrow(newdata ),as.integer(0.7*nrow(newdata ),replace=F))
traindata = newdata [random_row,]
#Assign the test data set to the remaining 30% of the original set
testdata = newdata [-random_row,]
table(newdata$V21)
head(traindata)

#pruned predictor model
set.seed(713)
lognew <- glm(V21~ V1_A11+V1_A12+V3_A30+V3_A31+V3_A32+V3_A33+V4_A41+V5+V6_A61+V7_A74+V8+V20_A201 ,family=binomial
summary(lognew)

#VIF check for multicollinearity
 VIF <- function(linear.model, no.intercept=FALSE, all.diagnostics=FALSE, plot=FALSE) {
    require(mctest)
    if(no.intercept==FALSE) design.matrix <- model.matrix(linear.model)[,-1]
    if(no.intercept==TRUE) design.matrix <- model.matrix(linear.model)
    if(plot==TRUE) mc.plot(design.matrix,linear.model$model[1])
    if(all.diagnostics==FALSE) output <- imcdiag(design.matrix,linear.model$model[1], method='VIF')$idiags[,1]
    if(all.diagnostics==TRUE) output <- imcdiag(design.matrix,linear.model$model[1])
    output
 }

#Vector of VIF values
values <- VIF(lognew)
par(bg = 'lightblue')
barplot(values, main = "Fig3: VIF Values on Final Logistic Regression Model", horiz = TRUE, col = "gold",las=2,
        cex.names=.53,xlim=c(0,5))
abline(v = 5, lwd = 3, lty = 2)

#final model validation on test data
set.seed(713)
predicted <- predict(lognew, testdata, type="response")
cutoff <- optimalCutoff(testdata$V21, predicted)[1]
roundup <- as.integer(predicted > cutoff )

#Confustion matrix
confusionMatrix(roundup, testdata$V21, threshold = cutoff)

#Misclassification error
misClassError(testdata$V21, roundup, threshold = cutoff)

#sensitivity and specificity
sensitivity(testdata$V21, roundup, threshold = cutoff)
specificity(testdata$V21, roundup, threshold = cutoff)

#FP:FN ratio implementation
costs = matrix(c(0, 5, 1, 0), nrow = 2)
dimnames(costs) = list(Actual = c("good", "bad"), Predicted= c("good", "bad"))
print(costs)

#thresholding
cost <- vector(mode = "list")
for (i in 1:100){
    predicted_roundup <- as.integer(predicted > i/100 )
```

```
    cm_matrix <- as.matrix(table(testdata$V21 ,roundup))

    #out of bounds check
    if(nrow(cm_matrix)==2) {fp<-cm_matrix[2,1]} else {fp=0}
    if(ncol(cm_matrix)==2){fn<-cm_matrix[1,2]} else {fn=0}

    cost<-c(cost, fn*1+fp*5)
}

#Plots ov Total cost vs % thresholds
par(bg = 'lightblue')
plot(x=seq(0.01,1,by=0.01),y=cost,xlab = "Threshold, %",ylab = "Total Cost of Misclassifications",main = "Fig4: C
grid (10,10, lty = 6, col = "white")

#minimum threshold
num<-which.min(cost)
t <-num/100
print(sprintf("Minimum Threshold due to Uneven Error Cost= %0.3f", t))
```

THE END----------
