PART-1

Methodology

For each measurement location, 4-fold (each fold should be correspond to one week of data) cross-validation was used to examine the predictive power of polynomials of occupancies with orders 2, 4, 6, 8 on volumes and log-transformed volumes.
Models were estimated using conventional least squares (use function lm) and robust regression (use rlm.
Results were converted for log-transformed volumes to correspond to the original scale
Report Mean Absolute Error, Median Absolute Error and Symmetric Median Absolute Percentage Error (sMdAPE) with Yt observed and Ft predicted volumes in 3 matrices.
Each column corresponds to a measurement location, each row to one of the (16) predictive models;
Averages were reported for performance metrics across folds.

Basic Idea

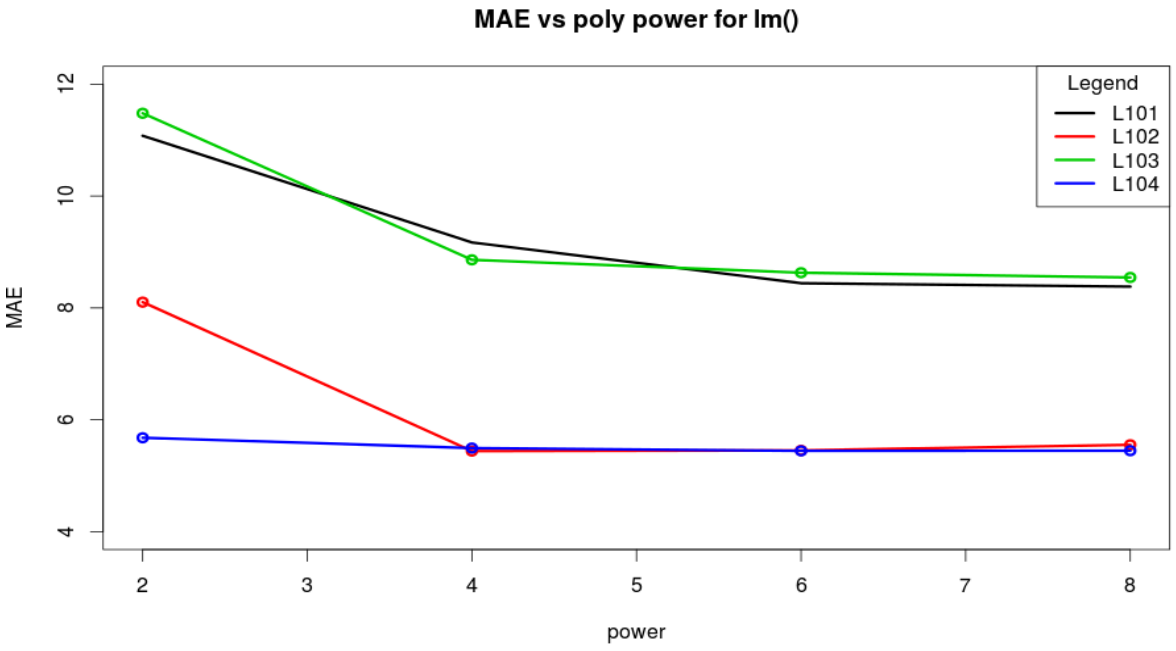
The method that was that employed was two for loops the first one was used for running across the polynomial powers and the second was for dividing the traffic data set into 4 parts such that each partition approximately corresponded to a week of data.
For the Linear model the lm command was used
For the robust model rlm was used.
For estimating the model and the error for log transformed values "1" was added to the Traffic Occupancy to avoid log(0) values for the calculation then transformed back to the original scale.
16 R files were created for this program that had 4 polynomial powers for the 4 kinds of regression.

ERRORS

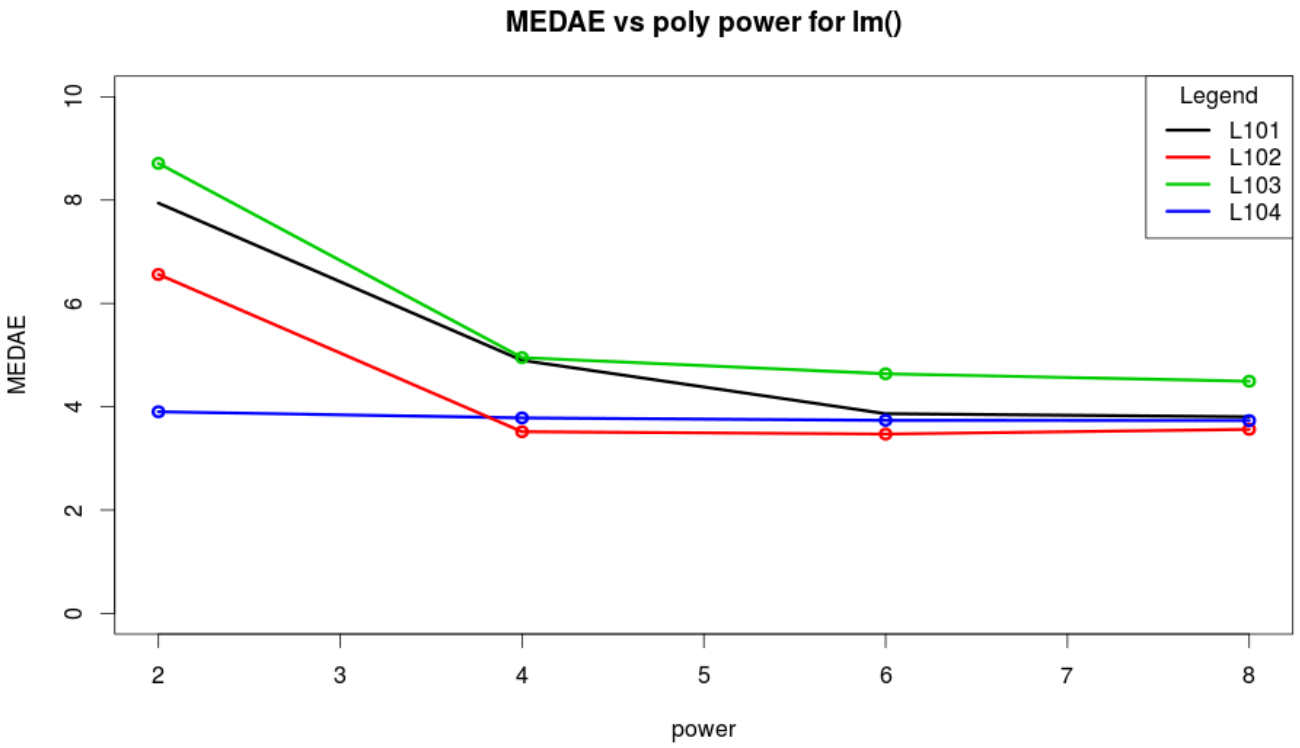
The three errors (MAE, MEDAE and SMDAPE) that were calculated were each column corresponds to different measurement locations and each row to one of the (16) predictive models is attached in a folder called **errors**. They are named MAE.csv MEDAE.csv and SMDAPE.csv.

Comparison of errors for various powers of different regression methods

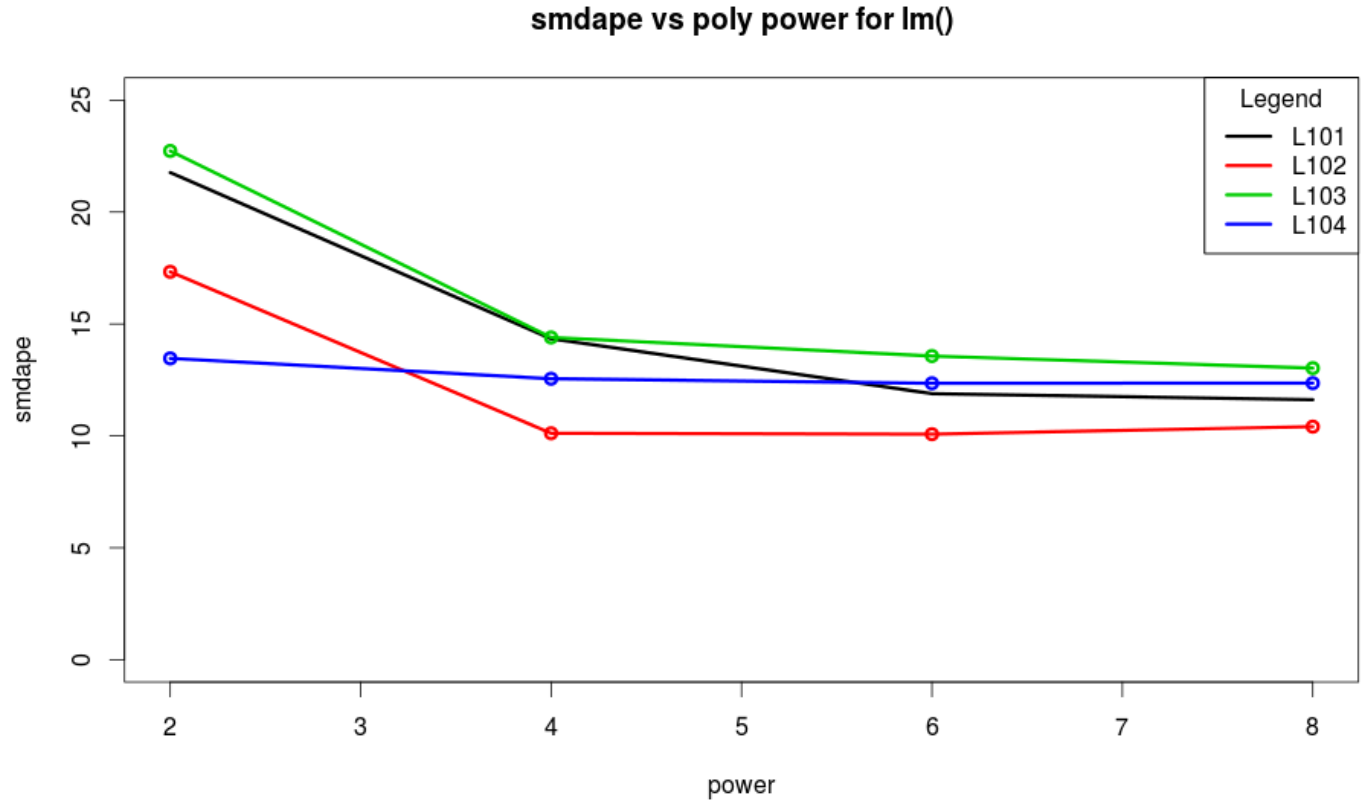
ERRORS for Conventional Least Square Regression



Mean absolute errors for various powers when we use Conventional Least Square Regression.



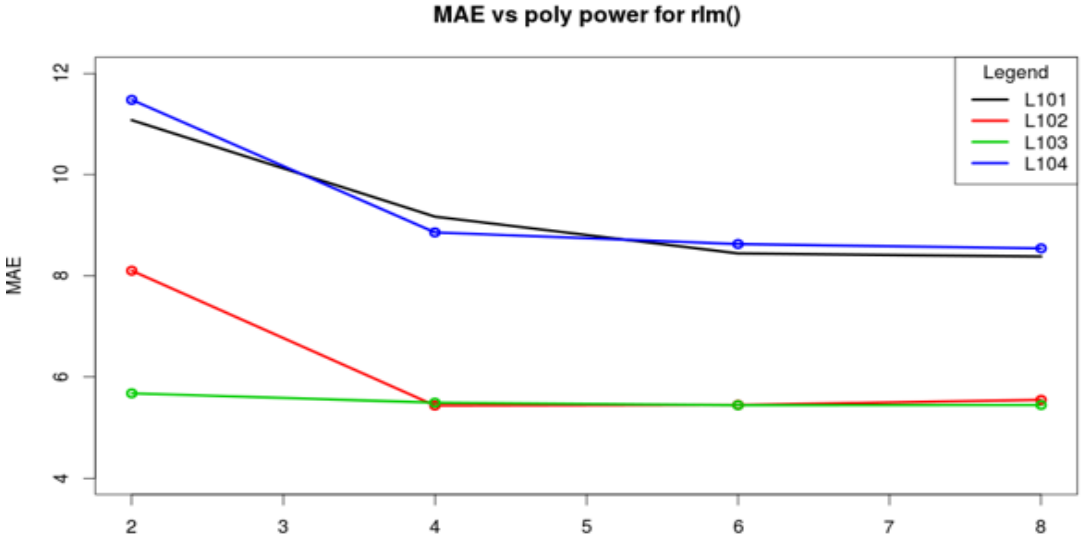
Median absolute errors for various powers when we use Conventional Least Square Regression.

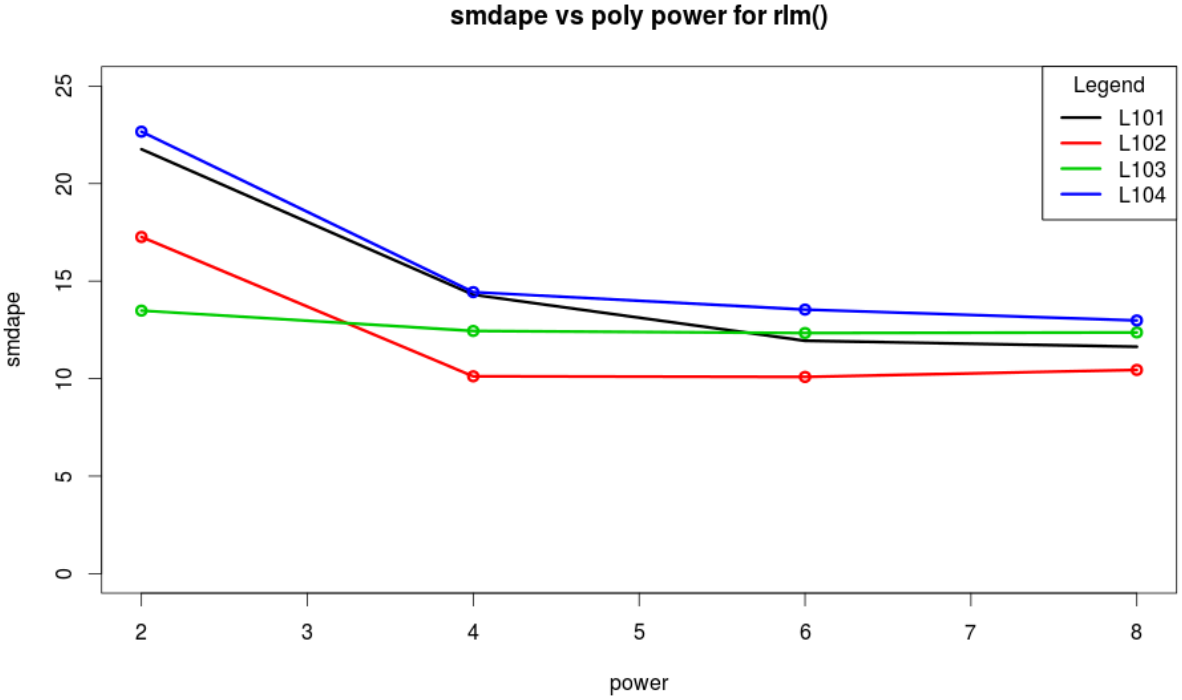


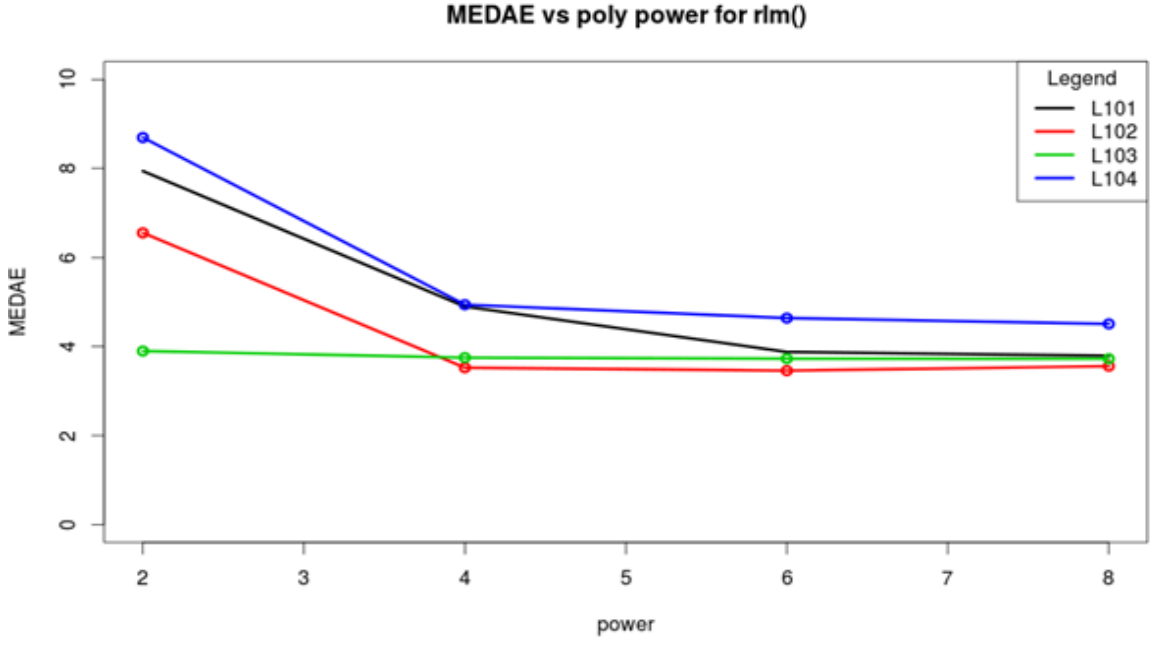
SMDAPE for various powers when we use Conventional Least Square Regression.

As expected lower powers perform not very well for all the data that we consider. The error difference becomes less significant as we approach higher powers.

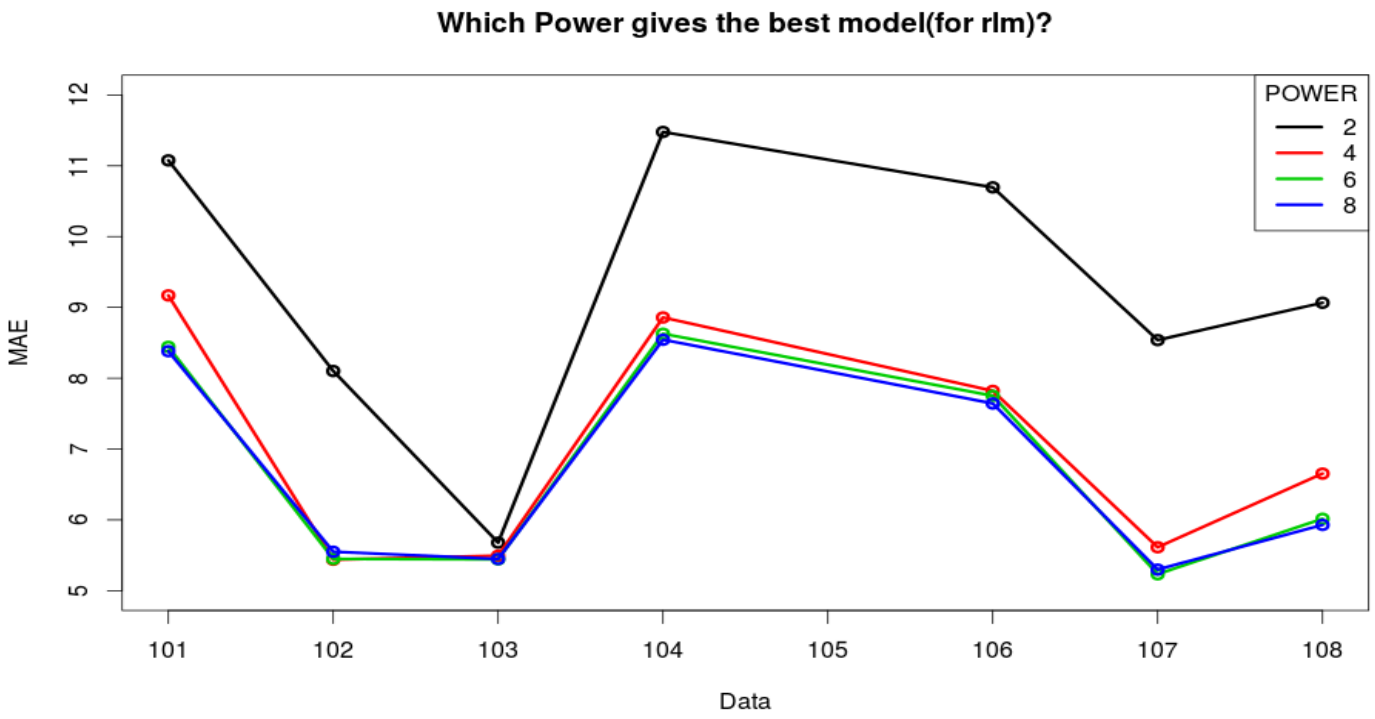
ERRORS for Robust Regression

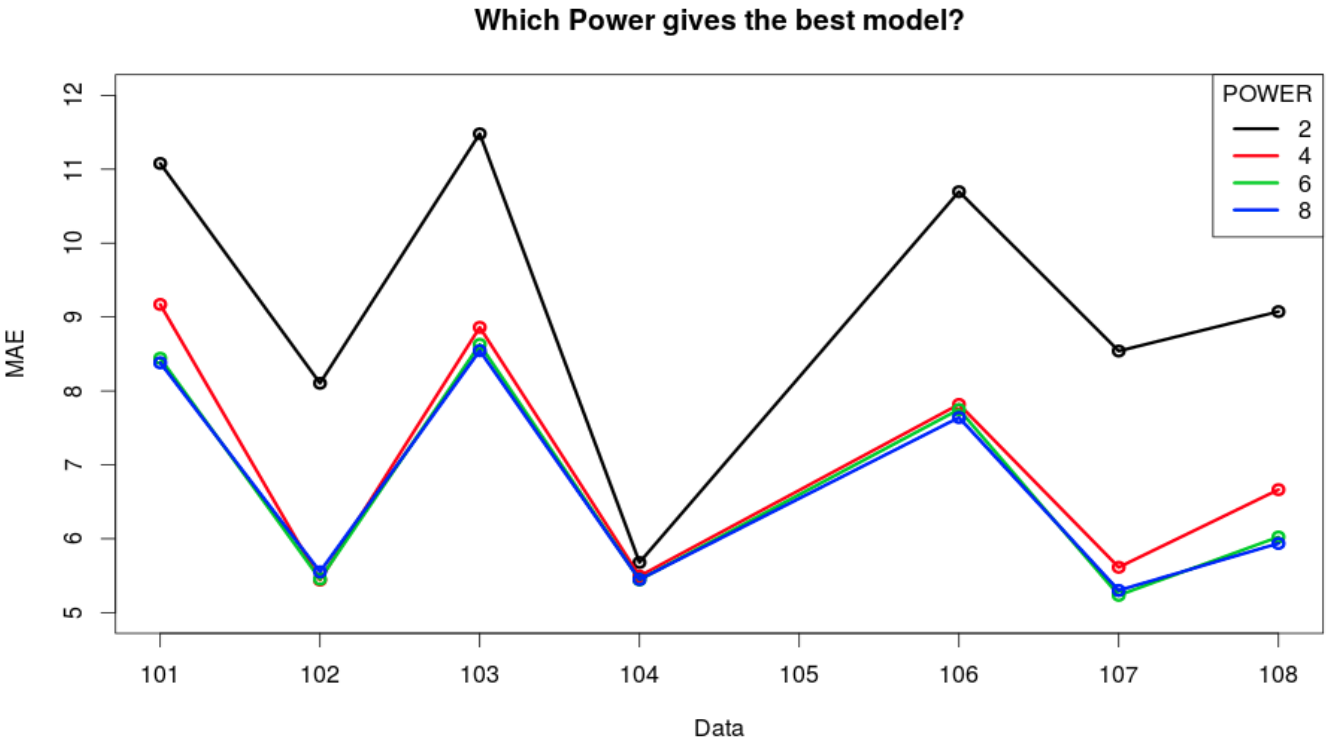






Comparison of error in powers across various data.



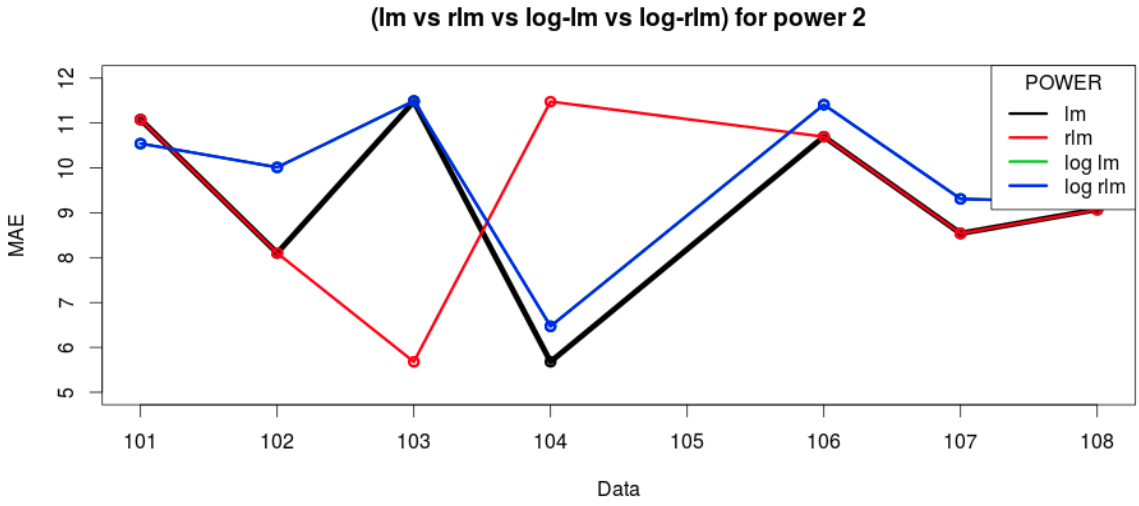


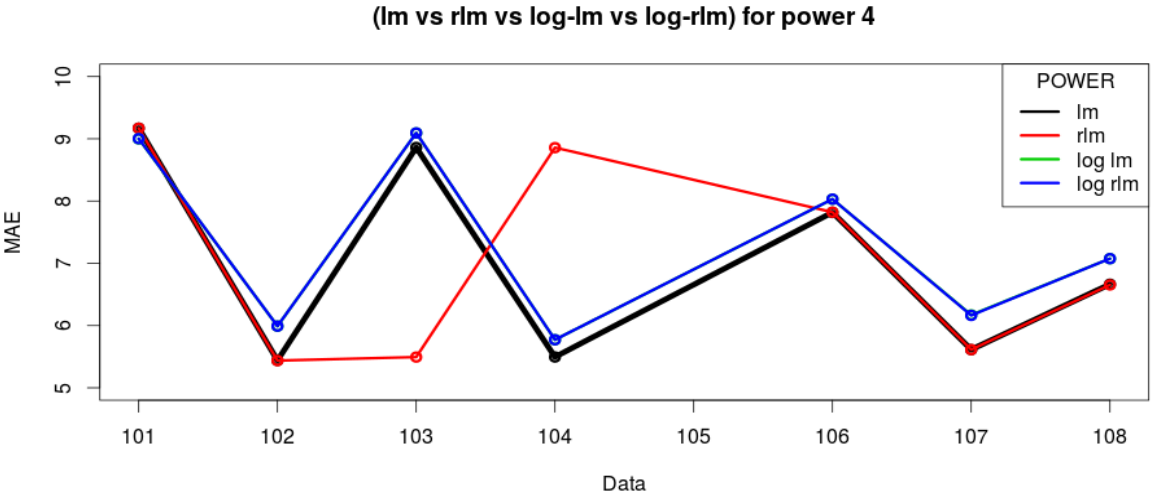
This plot clearly shows that 6 and 8 powers perform better than 2 and 4. Between 6 and 8 depending on the specific case taken into consideration either of them can perform well.

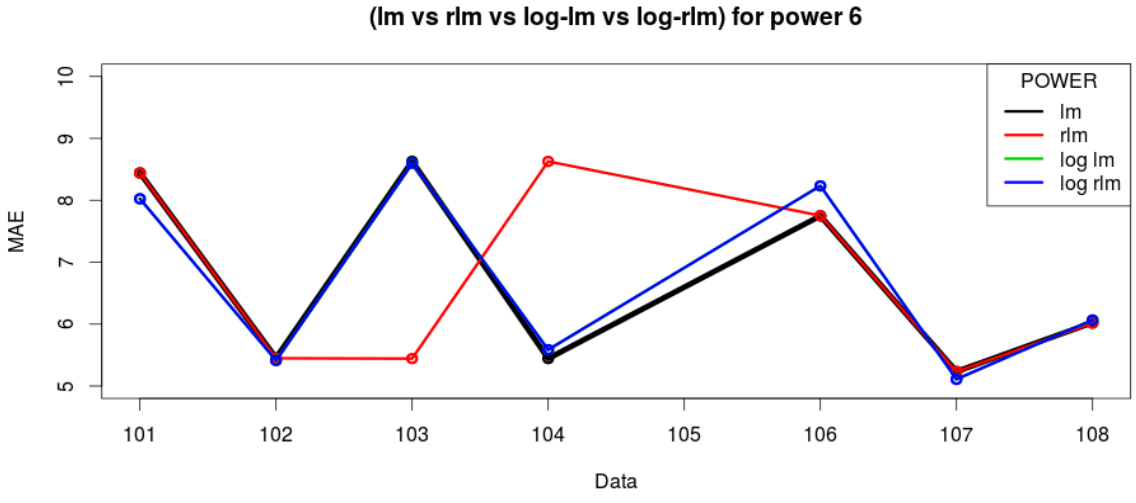
What type of regression gave the lowest error?

We compare the MAE, SMDAPE, MEDAE for the 4 types of regression used.

Comparing various models







With respect to MAE Robust Regression proved to be best.
Here is the proof that supports the claim:
Finding the smallest error in each location:

```
> which(MAE == min(mae1$X101), arr.ind=TRUE)
                          row col
Robust_Linear-rlm-log(8)  16   1


> which(MAE == min(mae1$X102), arr.ind=TRUE)
                          row col
Robust_Linear-rlm-log(6)  15   2


> which(MAE == min(mae1$X103), arr.ind=TRUE)
                     row colRobust_Linear-rlm(6)   7   3
> which(MAE == min(mae1$X104), arr.ind=TRUE)
                       row col
          Linear-lm(6)   3   4


> which(MAE == min(mae1$X107), arr.ind=TRUE)
                          row col
Robust_Linear-rlm-log(8)  16   6


> which(MAE == min(mae1$X108), arr.ind=TRUE)
                          row col
Robust_Linear-rlm-log(8)  16   7
```

Best Performers

Linear- 1/7

Robust- 6/7

Power 6- 3/7

Power 8- 4/7

Log transformed Regression

We use the conventional least squares (function lm) and robust regression (rlm) on log transformed values of the Traffic Volume. This gives us a new problem. We have missing data or 'zero' volumes in certain cases. We deal with this by adding '1' to all the volumes

That is **log(x+1)** where **x= Traffic Volumes** and then convert back to the original scale by **exp(x')-1** where **x'=Predicted values** of Traffic Volumes. From here we use the conventional method to calculate the errors.

Findings

- Higher order polynomials tend to perform better in terms of providing a lower error (better fit). Since higher polynomials tend to explain the influence of outliers and random noises in the data we should choose the lowest polynomial that provides an optimum performance. Having these considerations in mind while choosing the best fit we observe that the best fitting polynomial varied from case to case but in general we observe that 6^{th} order polynomial gives an efficient fit.
- Robust regression proves to be the best predictor in majority of the cases.
  It showed minimum value of error parameters in almost all the cases though it showed some abnormally high values at rare occasions.
- Second order polynomial should be completely avoided for prediction.
- system.time() -utilized to minimize the total running time.

PART 2:

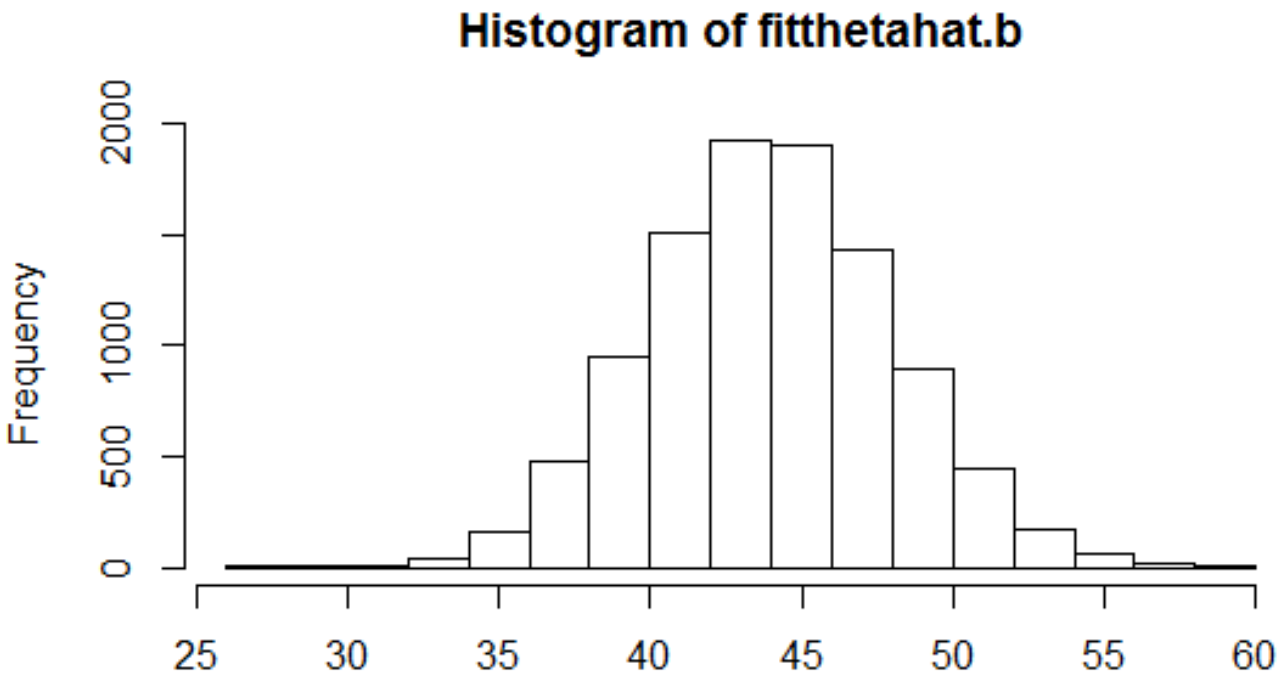
For each measurement location, compute bootstrap confidence intervals for the least squares coefficients of the best performing polynomial (according to MAE). 6^{th} polynomial of the linear regression was naturally chosen as the best performer keeping in mind that its MAE values were similar to that of 8^{th} power and it was a lower order polynomial.

Conventional methods of confidence interval for bootstrap samples takes extremely long time. If decisions need to be taken in a short span of time such as in the case of trading of stocks, this may not end very well. Several methods were tried for finding the confidence interval.

Boot.ci() is one of the ways of finding the interval function as it generates 5 different types of two-sided nonparametric confidence intervals -first order normal approximation, the basic bootstrap interval, the studentized bootstrap interval, the bootstrap percentile interval, and the adjusted bootstrap percentile (BCa) interval.

We could choose-`"norm"`, to assume normality and if not we may pick `"basic", "stud", "perc", "bca"`.

The attached code calculates the confidence interval of all the 7 measurement locations and plots a histogram of the distribution.`



PART3:

Apply logistic regression to predict the probability of default using income and balance on the Default data set (this dataset is included in the ISLR library). Set a random seed before beginning your analysis.

Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps: Split the sample set into a training set and a validation set. Fit a multiple logistic regression model using only the training observations. Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

Repeat the process in above three times, using three different splits of the observations into a training set and a validation set. Comment on the results obtained.

Now consider a logistic regression model that predicts the probability of default using income, balance, and a dummy variable for student. Estimate the test error for this model using the validation set approach. Comment on whether or not including a dummy variable for student leads to a reduction in the test error rate.

Obtain bootstrap estimates of the standard error of the `Pearson' and `Spearman' correlation between income and balance.

Answers:

Here we create a Logistic regression model using dataset of credit card information to predict if a person will default on their credit card payment.

Given data:

Dataset-Default.

Credit Card Default Data.

Description.

A simulated data set containing information on ten thousand customers. The aim here is to predict which

customers will default on their credit card debt.

Usage.
A data frame with 10000 observations on the following 4 variables.

default.
A factor with levels No and Yes indicating whether the customer defaulted on their debt
student.
A factor with levels No and Yes indicating whether the customer is a student.
balance.
The average balance that the customer has remaining on their credit card after making their monthly payment.
income
Income of customer

We will try to predict whether someone will default on their credit card payment, based on the variables student, balance and income

Question: logistic regression to predict the probability of default using income and balance

```
library(ISLR)

str(Default)
data.frame':10000 obs. of  4 variables:
$ default: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
$ student: Factor w/ 2 levels "No","Yes": 1 2 1 1 1 2 1 2 1 1 ...
$ balance: num  730 817 1074 529 786 ...
$ income : num  44362 12106 31767 35704 38463 ...

str(default)
 Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
attach(Default)
table1<-table(default)
(table[[2]]/table[[1]])*100
[1] 3.444709
Percentage of people who default:3.44709

#
logit.fit <-  glm(default ~ income + balance, data = Default, family = binomial)
logit.fit
Call:  glm(formula = default ~ income + balance, family = binomial,
   data = Default)

Coefficients:
(Intercept)      income     balance
 -1.154e+01    2.081e-05    5.647e-03

Degrees of Freedom: 9999 Total (i.e. Null);  9997 Residual
Null Deviance:    2921
Residual Deviance: 1579 AIC: 1585
```

Positive coefficient : This says that relation is directly proportional between default

#Using the validation set approach, we estimate the test error of this model
 # We randomly split the sample set into two equal halves- training set and a validation set.

```
train <- sample(dim(Default)[1], 5000)
```

#Then we fit a multiple logistic regression model using only the training observations.

```
logit.fit <- glm(default ~ income + balance + student, data = Default, family = binomial,subset = train)
```

#We compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

```
logit.pred <- rep("No", 5000)
logit.prob <- predict(glm.fit, Default[-train, ], type = "response")
logit.pred[logit.prob > 0.5] <- "Yes"
mean(logit.pred != Default[-train, ]$default)*100
```

 # Considering a 50% threshold for probability of default.

[1] 2.52
[1] 2.76
[1] 2.98

The observations included in the training set will affect the validation estimate of the test error rate. Hence the validation set approach is not right method to calculate the standard error.

If we check for extreme thresholds

 # Considering a 10% threshold for probability of default.

```
logit.pred[logit.prob > 0.1] <- "Yes"
```

[1] 6.34
[1] 6.5
[1] 7.02

 # Considering a 90% threshold for probability of default.

```
logit.pred[logit.prob > 0.9] <- "Yes"
```

[1] 2.8
[1] 3.2
[1] 3.28

Identifying the effect of a student dummy variable consider a logistic regression model that predicts the probability of default using income, balance, and a dummy variable for student.

```
logit.fit <- glm(default ~ income + balance + student, data = Default, family = binomial)
train <- sample(dim(Default)[1], 5000)
logit.fit <- glm(default ~ income + balance, data = Default, family = binomial,
          subset = train)
logit.pred <- rep("No", 5000)
logit.prob <- predict(glm.fit, Default[-train, ], type = "response")
logit.pred[logit.prob > 0.5] <- "Yes"
c=mean(logit.pred != Default[-train, ]$default)*100
```

Standard Error:

[1] 2.98

Test error rate: No effect on adding student dummy variable.

Summary of the fit

- Summary(logit.fit)

Coefficients:

Estimate Std. Error z value Pr(>|z|)

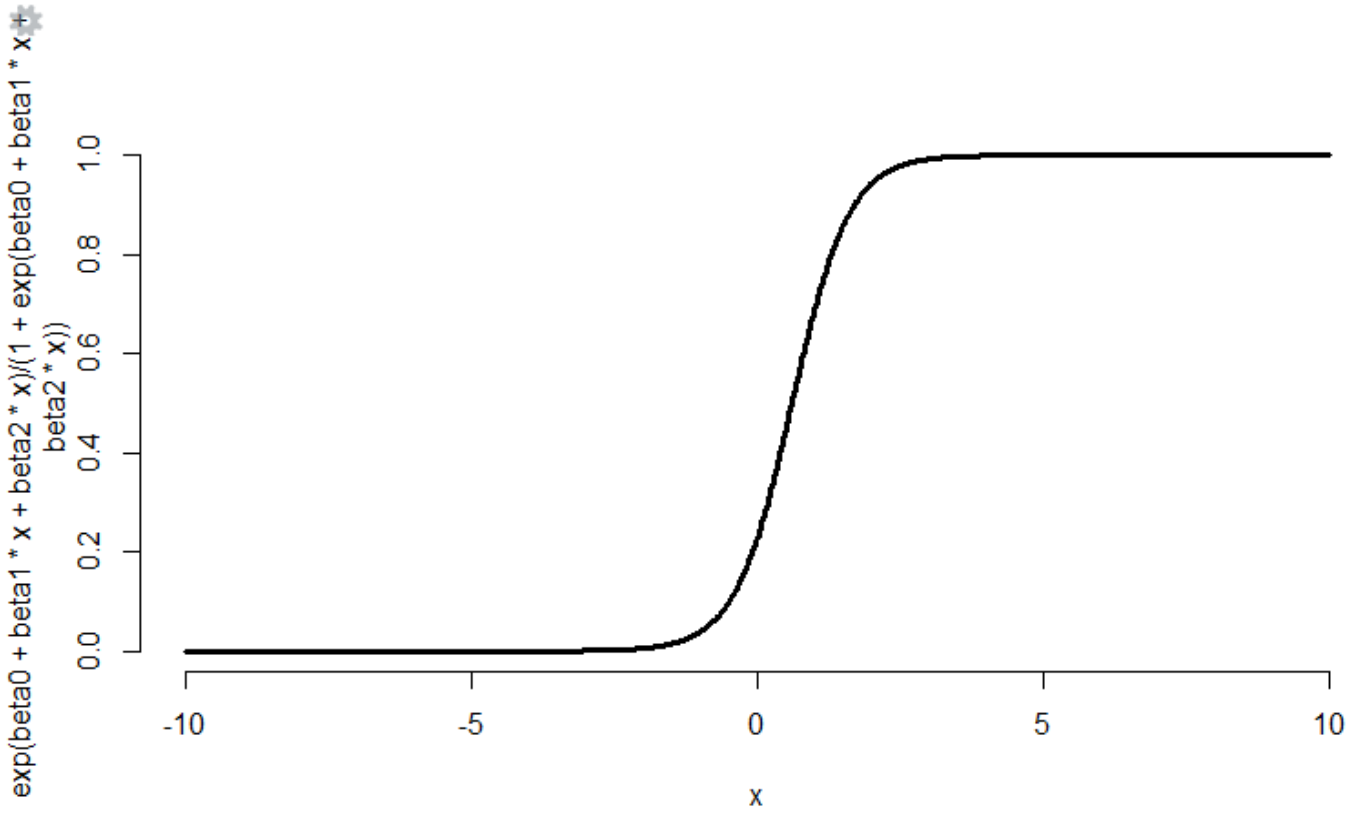
(Intercept) -1.176e+01 6.283e-01 -18.711   <2e-16 *
income   1.424e-05 6.970e-06   2.043   0.0411 *
balance   5.894e-03 3.323e-04 17.734   <2e-16 *****

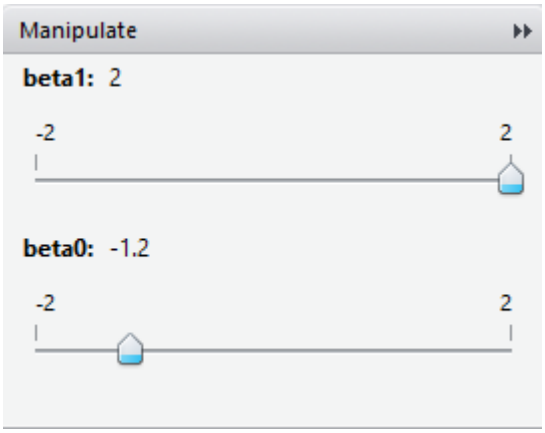
- **Standard errors for beta0=6.283e-01, beta1=6.970e-06, beta2=3.323e-04**

Using manipulate()

```
library(manipulate)

x <- seq(-10, 10, length = 1000)
manipulate(
  plot(x, exp(beta0 + beta1 * x+beta2*x) / (1 + exp(beta0 + beta1 * x+beta2*x)),
       type = "l", lwd = 3, frame = FALSE),
  beta1 = slider(-2, 2, step = .1, initial = 2),
  beta0 = slider(-2, 2, step = .1, initial = 0)
```






```
exp(logit.fit$coefficients)
```

(Intercept) income balance
5.132518e-06 1.000023e+00 1.005936e+00

log(p(X) / 1-p(X)) = β0 + β1X+ β2X
β1=balance
β2=income
β0- intercept from the glm fit.

Confidence Intervals for the Coefficients

```
exp(confint(logit.fit))
                2.5 %      97.5 %
(Intercept) 1.302832e-06 1.794039e-05
income      1.000009e+00 1.000038e+00
balance     1.005283e+00 1.006648e+00
```

ANOVA for the logistic regression

```
anova(logit.fit,test = "Chisq")
Analysis of Deviance Table
Model: binomial, link: logit
Response: default
Terms added sequentially (first to last)

        Df Deviance Resid. Df Resid. Dev Pr(>Chi)
NULL                   4999    1375.12
income   1    1.07     4998    1374.05   0.3016
balance  1  665.49     4997     708.57   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

`Pearson' and `Spearman' correlation between income and balance.

Reference: Discovering Statistics Using R By Andy Field, Jeremy Miles, Zoë Field

```
bs=function(Default,i)cor(Default$income[i],Default$balance[i],use="complete.obs",method="spearman")
b_spearman=boot(Default,bs,2000)
b_spearman
ORDINARY NONPARAMETRIC BOOTSTRAP
Call:
boot(data = Default, statistic = bs, R = 2000)
Bootstrap Statistics :
      original       bias    std. error
t1* -0.1492335 3.067582e-06 0.009794576
ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
bs=function(Default,i)cor(Default$income[i],Default$balance[i],use="complete.obs",method="pearson")
b_pearson=boot(Default,bs,2000)
Call:
boot(data = Default, statistic = bs, R = 2000)
Bootstrap Statistics :
      original       bias    std. error
```

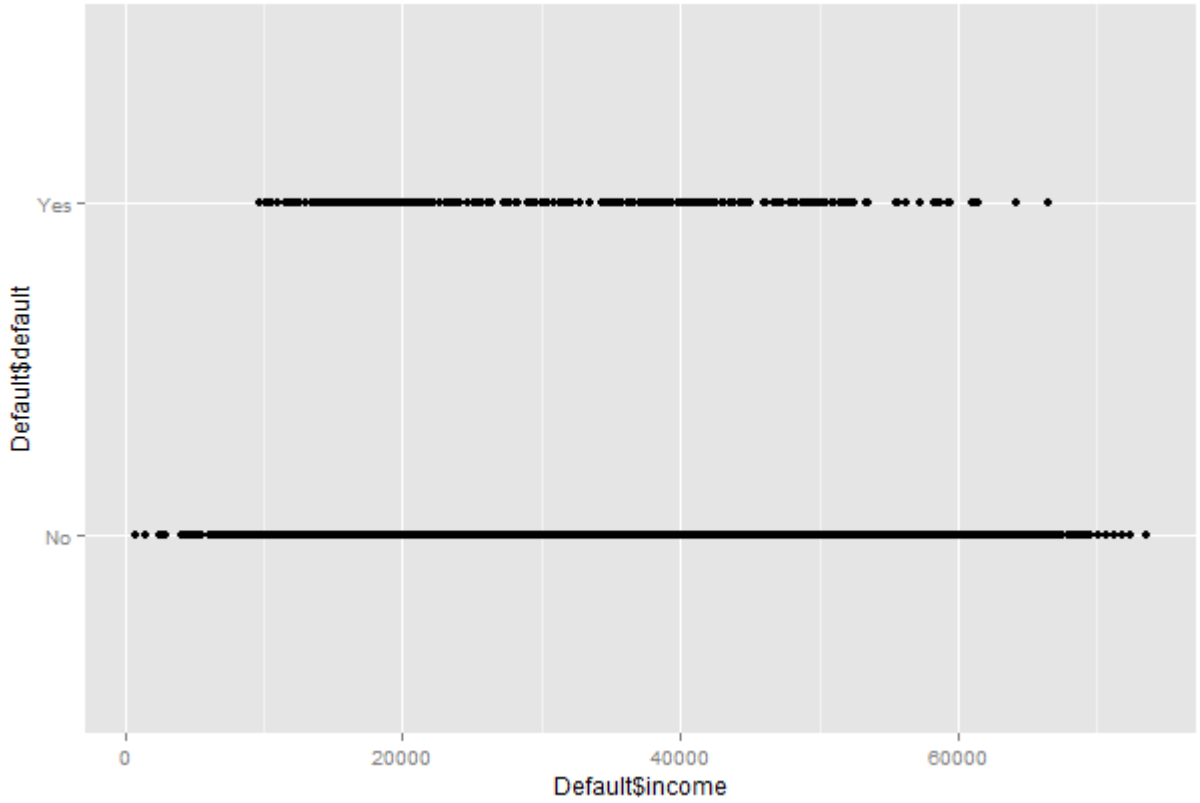
```
t1* -0.1522434 0.0001897402 0.009737723
```

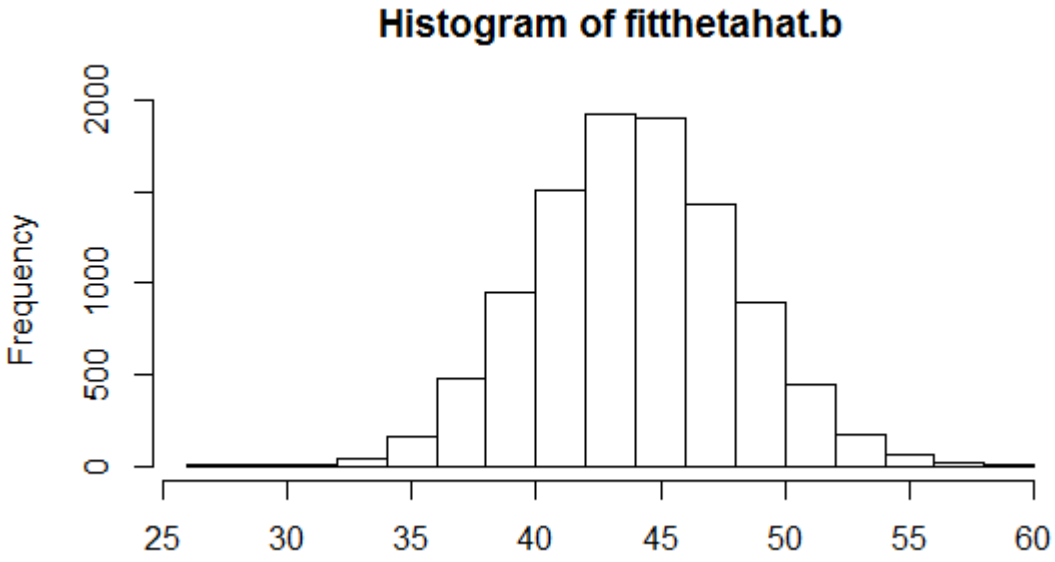
Pearson standard error=0.009737723
Spearman standard error=0.009794576

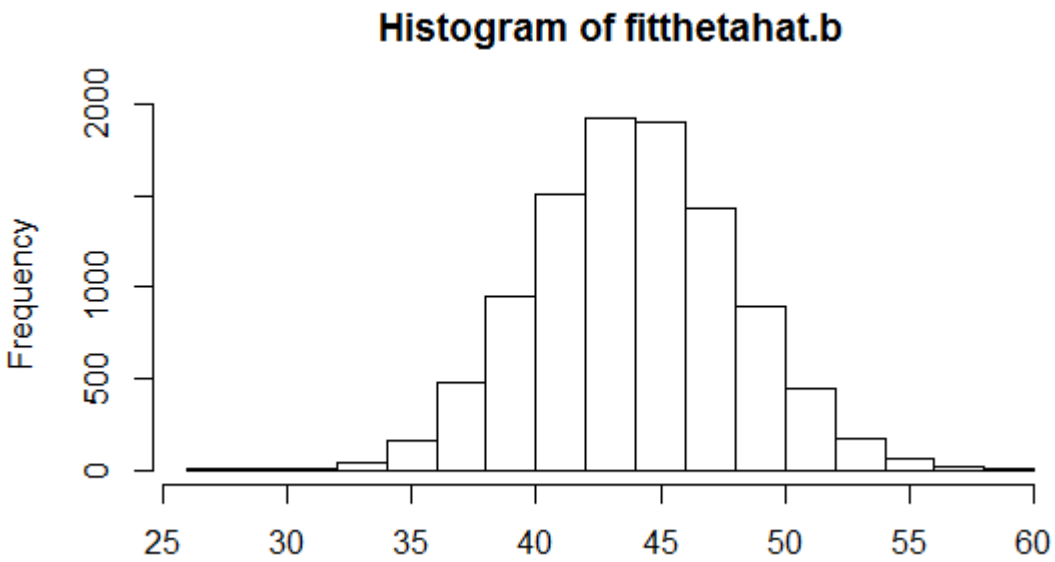
Findings

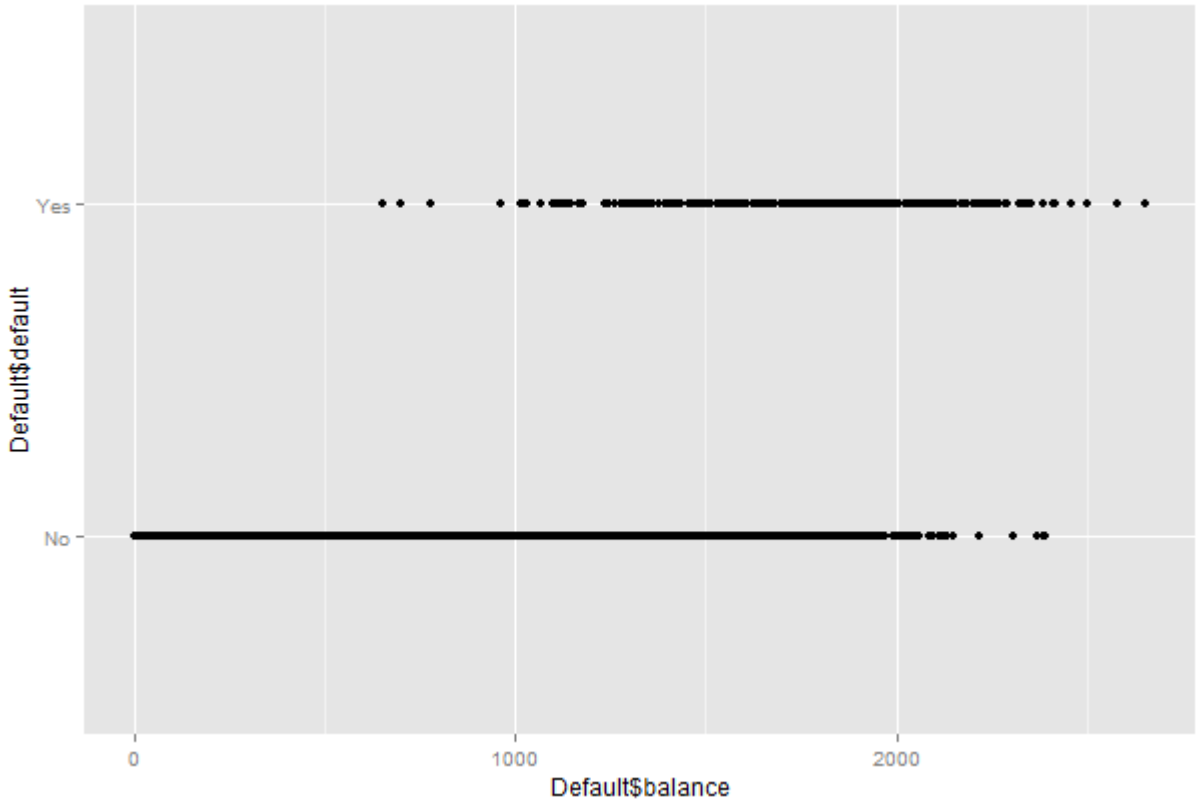
- Income does not play any role in the probability of default. People of all income tend to default on their credit card payments as seen in the ggplot figure.

```
qplot(Default$income,Default$default)
qplot(Default$student,Default$default)
```


Yes
No
Default$default
0
20000
40000
60000
Default$income


Histogram of fitthetahat.b
Frequency
0
500
1000
2000
25
30
35
40
45
50
55
60





- People having higher balance tend to default more.
- Students have higher balances hence students are expected to default more.
-

```
> mean(as.numeric(default1$default[default1$student=="No"])-1)
[1] 0.02919501
> mean(as.numeric(default1$default[default1$student=="Yes"])-1)
[1] 0.04313859)
```

- The above calculation says that students have a probability of 0.4313859 and the rest 0.2919501 of defaulting their payment.
- This may be attributed to student loans.
- Dummy student variable also does not affect the probability of Default.

summary(logit.fit)$coeff[,1]

```
(Intercept)        income       balance    student
-1.122975e+01  7.106967e-07  5.949091e-03 -4.839486e-01
```

- A one-unit increase in balance increases the log odds of default by 5.949091e-03 units.
- Student here has a negative coefficient this says that Students default less in contradiction with the earlier conclusion that they default more.