Regression Analysis of Boston Housing Data

Vinay kumar Sharma
Rajat Agarwal

April 2020

Abstract

Here , we are working on BOSTON HOUSING data which represent the housing values in various suburbs of Boston. Starting with Data pre processing , we checked missing observation in the data and checked whether the variable are qualitative or quantitative in nature .We found that all the variable are quantitaive in nature .Our target variable is 'MEDV'(Median house prices of owner owned house).Here we are performing parametric method of estimation of model. we assumed the linear model setup and standardized the regressors (not the response).Here , we are using validation set approach to estimate test set error standard deviation .We split the data into test set and train set. for further analysis , we checked whether the model we fitted using the training dataset holds regression assumption of hetroscedascity,normality ,linearity e.t.c . We detected Leverage and influential points with help of Cook's Distance , DFFITS , DFBETAS , COVRATIO. Then move to deal with problems like Curvatures (correct them with suitable methods), homoscedasticity(checking using plots of regressor vs residuals and also by Breusch Pagon test if found transformed our model). Deal with non-normality(checking using Q-Q plot if found then corrected using Box-cox transformation on response). Finding RMSE in each step of model diagnosis allows us to compare each model and to get the model with best predicting power.

Key words: RMSE (Root Mean Square Error) , Breusch Pagon test , Influential ,Cook's Distance ,homoscedasticity normality , APR(Augmented Partial Residual) , CPR(Component Plus Residual).

Contents

1 Acknowledgement

We would like to express our heartfelt gratitude to Dr. Minerva Mukhopadhyay for helping us in every difficulty which we face during this project. It has been a great learning experience and has also provided us with a practical insight of the theoretical knowledge gathered during the course Regression Analysis.

We also take this opportunity to thank the authors and publishers of the various books and journals we have consulted. Without those this work would not have been completed.

We would also like to thank our parents for their extensive support throughout the session. Their constant encouragement has enabled us to complete the project within the stipulated time-period.

2 Introduction

The Boston housing market is highly competitive, and we want to be the best real estate agent in the area. To compete with our peers, we decide to leverage a basic machine learning concepts to assist us and our client with finding the best price for home of there choices . Luckily, we've come across the Boston Housing dataset which contains aggregated data on various features for houses in Greater Boston communities, including the median value of homes for each of those areas. Our task is to build an optimal model based on a statistical analysis with the tools available. This model will then be used to estimate the best selling price for our clients' homes as per the feature they want for there house.

3 Objective

Here our interest lies in following point :

(1.)To Build a model that satisfy all usual assumption of General linear model.

(2.)Verify Prediction accuracy of model that we constructed is good enough ?

(2.)Which feature influence less and more on the predicted house prices?

4 RMSE

We will use Root Mean square Error (RMSE) to check how good our model accuracy for predictions about our response. RMSE is a good estimator for the standard deviation of the distribution of our errors. Formula for RMSE is as :-

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_{test} - \hat{y}_{test})^2$$

$$RMSE = \sqrt{MSE}$$

Where y_{test} is our response for test data set and $\hat{y}_{test}$ is the predicted value of response on test data, Calculated based on the fitted model obtained using train data. If the error is small, as estimated by RMSE, this generally means our model is good at predicting 'MEDV'.

5 Data Description

Title of the Data Set :- Boston housing dataset.

Source :- Harrison, D. and Rubinfeld, D.L. (1978) Hedonic prices and the demand for clean air. J. Environ. Economics and Management 5, 81–102.

Belsley D.A., Kuh, E. and Welsch, R.E. (1980) Regression Diagnostics. Identifying Influential Data and Sources of Collinearity. New York: Wiley.

Data Set Information :- The Boston data set has 506 rows and 14 columns.
This data set contains the following columns:

CRIM :- per capita crime rate by town.

ZN :- proportion of residential land zoned for lots over 25,000 sq.ft.

INDUS :- proportion of non-retail business acres per town.

CHAS :- Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).

NOX :- nitrogen oxides concentration (parts per 10 million).

RM :- average number of rooms per dwelling.

AGE :- proportion of owner-occupied units built prior to 1940.

DIS :- weighted mean of distances to five Boston employment centres.

RAD :- index of accessibility to radial highways.

TAX :- full-value property-tax rate per $10,000$.

PTRATIO :- pupil-teacher ratio by town.

BLACK :- $1000(Bk - 0.63)^2$ where Bk is the proportion of blacks by town.

LSTAT :- lower status of the population (percent).

We have these 13 Attributes as regressors(x_j). Which also have 12 Attributes as of continous formed and one is of binary valued Attribute(CHAS)

MEDV :- median value of owner-occupied homes(in $1000's$ dollors).This variable is taken as **response variable (Y)**.

6 Data Pre-processing

6.1 Importing Data

We will use library(MASS) for importing the library containing the Boston Housing Data setting name "data" to our boston housing data, setting Y for our response vector and X as our Design matrix and estimate our coefficients.

```
> library(MASS)  # library including the Boston datasets
> data<-as.matrix(Boston)
> #Structure of dataset
> str(data)
 num [1:506, 1:14] 0.00632 0.02731 0.02729 0.03237 0.06905 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:506] "1" "2" "3" "4" ...
  ..$ : chr [1:14] "crim" "zn" "indus" "chas" ...
> #Dividing the columns of data in Y(as response) and X(as predictor)
> Y=data[,ncol(data)]    #Response
> #Number of obserbvation's in respose
> length(Y)
[1] 506
> X=data[,-ncol(data)]   #Predictor Matrix
> #Predictor matrix including intercept term
> X=cbind(rep(1,nrow(X)),X)
> #dimension of predictor matrix
> dim(X)
[1] 506  14
> colnames(X)[1]="intercept"
> #Names of each predictor variable
> colnames(X)
 [1] "intercept" "crim"      "zn"        "indus"     "chas"      "nox"       "rm"
 [8] "age"       "dis"       "rad"       "tax"       "ptratio"   "black"     "lstat"
> #So, we are removing regressor 'zn' , 'chas' , 'rad'
> j=which(colnames(X)=="zn")
> k=which(colnames(X)=="chas")
> l=which(colnames(X)=="rad")
> #removing columns of ZN,CHAS,RAD from X(Predictor matrix)
> X=X[,-c(j,k,l)]
```

Figure 1: Importing data and removing 'zn','chas', 'rad'

6.2 Missing Value

Fortunately in given dataset it is clearly mention that there is no missing observations which make our work little bit less.

6.3 Categorical Variable

Again fortunately there is no categorical variable,actually our job is already done because in or dataset CHAS has already in indicator or binary typed so no need of doing anything for this variable.

7 Primary Model

Consider General linear regression setup with 'MEDV' as the response, and the remaining variables, except 'ZN', 'CHAS' and 'RAD', as predictors.Our model is as :

$$\mathbf{y} = \mathbf{X}\beta + \epsilon \quad .....(1)$$

Where $\mathbf{X}_{n\times p+1} = (\mathbf{1}, x_1, x_2, \cdots, x_{10})$ is the non-stochastic matrix of predictors (here $\mathbf{1}$ is the column whose all elements are 1) and $\beta = (\beta_0, \beta_1, \cdots, \beta_{10})$ is the vector of regression coefficients and $\epsilon = (\epsilon_1, \epsilon_2, \cdots, \epsilon_n)$ is the error vector.Here , we are not considering three regressor as our interest doesnot include that feature for house prices.

7.1 Assumptions

(i) $\epsilon_i \overset{iid}{\sim} \mathbf{N}(\mathbf{0}, \sigma^{\mathbf{2}})\ \forall i$, where σ^2 is an unknown constant.
(ii) $\mathbf{X}$ is of full column rank.
(iii) There is no curvature.
(iv) There is no **Hetroscedasticity**.

7.2 Standardise our regressors

here , we are standardizing our predictor matrix so as to convert all feature in same scale .

```
> #standardising the regressor of datasets
> for(i in 2:ncol(X))
+ {
+   X[,i]=(X[,i]-mean(X[,i]))/sd(X[,i])
+ }
```

Figure 2: Standardizing Predictor matrix

7.3 Splitting Data into Train and Test

Here use the concept of Validation set approach to check our predictive power of model on the test set . Let split the data randomly into nearly 80% (406 out of 506) as train set and nearly 20% (100 out of 506) as test set and then move to outlier detection and check other assumptions.

```
> # set.seed() will help to select the same random sample
> #whenever  we select a random #sample from data
> set.seed(52)
> #Selecting 100 random sample from total observation
> indices=sample(1:nrow(data),replace=FALSE,100)
> # Setting up  training set(406 observation) and
> #test set(100 observation) data Selected from whole data
> Y_train=(Y[-indices])
> X_train=(X[-indices,])
> #observations in train datasets
> dim(X_train);length(Y_train)
[1] 406  11
[1] 406
> Y_test=Y[indices]
> X_test=(X[indices,])
> #observation in test dataset
> dim(X_test);length(Y_test)
[1] 100  11
[1] 100
```

Figure 3: Splitting into test and train

Now we fitted our model from train data and then find RMSE from test data to check our model accuracy. But , here we have to check whether the predictor matrix is full column rank or not .If the predictor matrix is full column rank then we can further estimates our coefficient estimate using least square solution

7.4 Estimated coefficient, model fitting and RMSE.

```
> det(t(X_train)%*%X_train)
[1] 1.036169e+26
> #determinant of (t(X_train)%*%X_train) is not zero . Hence we conclude the matrix is full colum
n rank matrix
> #Least square solution or co-efficients efficent of the linear model fit
> beta_hat=solve(t(X_train)%*%X_train)%*%t(X_train)%*%Y_train
> beta_hat
                [,1]
intercept 22.3175204
crim      -0.3813723
indus     -0.7324612
nox       -1.6718780
rm         2.4131685
age       -0.2408388
dis       -2.4910960
tax        0.3467440
ptratio   -2.1871680
black      0.6728200
lstat     -3.7625047
> # estimated value of the  mean housing price of testing data
> Y_test_cap=X_test%*%beta_hat
> #RMSE Value of the testing the datasets
> RMSE=sqrt(sum((Y_test-Y_test_cap)^2))/sqrt(length(Y_test))
> RMSE
[1] 5.9042
```

Figure 4: Least Square Solution of coefficients

Here , we can see that RMSE value is 5.9042 which represent the standard deviation of residual error .Here we further check for the assumption of Linear model and Diagnosis each assumption and further check whether there is decrease in RMSE or not .If RMSE is decreasing this indicate that model accuracy has been improving each step.

8 Dealing with the Leverage and Influential Points .

8.1 Leverage Points

Leverage points are those which have unusual x values but does not affect the model estimates much. Leverage points can be find out with the help of Hat matrix(H), where Hat matrix is given as:

$$\mathbf{H} = \mathbf{X}(\mathbf{X^T X})^{-1}\mathbf{X^T}$$

if $\mathbf{h_{ii}}$ is ith diagonal element of Hat matrix, then traditionally ith point is considered as leverage if

$$\mathbf{h}_{ii} > \frac{\mathbf{2 * p}}{\mathbf{n}} \quad .....(a)$$

```
> ##(1)---Diagnosis of leverage and influential observations----
> #1(a).To find leverage point and drop those using suitable threshold
> n=nrow(X_train);p=ncol(X_train)
> #thred is thethreshold above which the points are cosidered as leverage point
> thred=2*p/n
> thred
[1] 0.05418719
> Htr=as.matrix(X_train%*%solve(t(X_train)%*%X_train)%*%t(X_train))
> h_ii=diag(Htr)
> indices=which(h_ii>thred) # 31 leverage points
> length(indices)
[1] 31
> #Removing the leverage point from data
> Y_train=Y_train[-indices]
> X_train=X_train[-indices,]
> #Observations left after removal of leverage point from training set
> dim(X_train);length(Y_train)
[1] 375  11
[1] 375
>
```

Figure 5: Codes for Leverage points

```
#leverage plotting

plot(h_ii,xlim=c(0,410),xlab="sample points",ylab="leverage points",
     col=(h_ii>thred)+1,pch=20)
abline(h=thred,col='blue',lwd=2,lty='dotted')
legend("topright",c("Leverage Point","Non a leveragePoint")
       ,col=c("red","black"),lwd=0.05)
```

Figure 6: Codes for plotting Leverage points

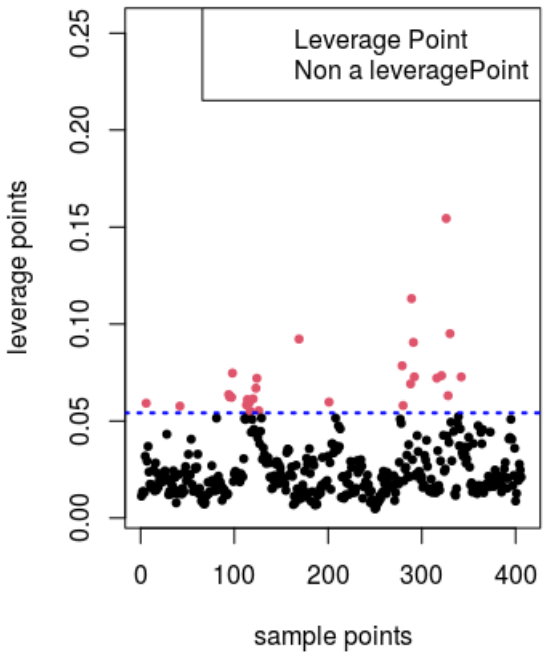

Figure 7: Leverage points graphically

Clearly from the given threshold of $h_{ii} = 0.05418719$ total 31 points are identified as leverage, we remove those point . Now ,the total number of observations in train set is become 375. Further move to detection of influencial points by using $Cook's\ distance$,$DFFITs_i$, $DFBETAs_{ji}$, $CovRatio$

8.2 Diagnostics for Influential points

Influential points are those which have both X and Y unusual.These points have considerable influence on estimates of coefficients and these pull the direction of regression line towards itself.
Our aim is to compute $Cook'sdistance$,$DFFITs_i$, $DFBETAs_{ji}$, $CovRatio$ and identify the influential point .

8.2.1 Cook's Distance

Cook's distance is one of the measure used for identification of Influential Points. Cook's distance is calculated as :

$$\mathbf{C}_i = \frac{(\hat{\beta} - \hat{\beta}_{(\mathbf{i})})^T (\mathbf{X^T X})(\hat{\beta} - \hat{\beta}_{(\mathbf{i})})}{\mathbf{pMSres}}$$

$$\mathbf{C}_i = \frac{r_i^2 * h_{ii}}{1 - h_{ii}}$$

where p is number of regressors, n is total number of observations . $r_i = \frac{e_i}{\sqrt{MSres(1-h_{ii})}}$is the internal studentized residual and e_i is the ith residuals.

It is clear from formula that it contain both part r_i(studentized residual) and h_{ii}(ith diagonal element of hat matrix) which measure how far observation is from data and how well the model fits the ith observation y_i Traditionally, the points for which $\mathbf{C_i}$ is greater than 1 is consider influential. But there is no influential points as per Cooks Distance.

```
> ##COOK'S DISTANCE
> n=nrow(X_train);p=ncol(X_train)
> #'beta_hat1' is Least square solution after removal of leverage point
> beta_hat1=solve(t(X_train)%*%X_train)%*%t(X_train)%*%Y_train
> #'Htr' is hat matrix
> Htr=X_train%*%solve(t(X_train)%*%X_train)%*%t(X_train)
> h=diag(Htr)
> Y_train_cap=X_train%*%beta_hat1
> res=Y_train-Y_train_cap
> MSres=sum(res^2)/(n-p)
> C=((res^2)*h)/(MSres*p*((1-h)^2))
>
> which(C>1) #indicates influential point
integer(0)
> #thus we do not get influential point as per Cook's Distance for this data #set
```

Figure 8: Code for detection of influential points using cook's distance

```
> #Cook's distance plotting
> plot(C,xlim=c(0,380),xlab="sample points",ylab="COOK'S DISTANCE",
+      main="fig:1.1 Cook's Distance Plot",col=(C>1)+1,pch=20)
>
```

Figure 9: codes for plotting Cook's Distance

Here down the graphical reprentation which shows there is no points outside the given traditional threshold C=1.

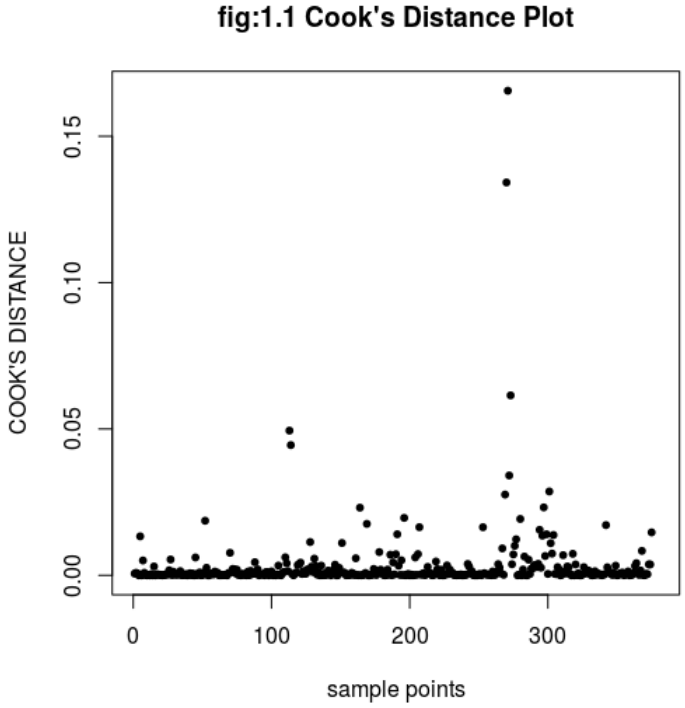

Figure 10: Cook's distance plot when threshold=1

8.2.2 DFFITS

DFFITS is another method for influential point detection and it also contain both part (r_i^{*2} and h_{ii}) which measure how far observation is from data and how well the model fits the ith observation y_i. $DFFIts_i$ is calculated as :

$$\mathbf{DFFITs}_i = \sqrt{\frac{r_i^{*2} * h_{ii}}{1 - h_{ii}}}$$

$r_i^* = \frac{e_i}{\sqrt{MSres_i(1-h_{ii})}}$is the external studentized residual. e_i is the ith residuals. $MSres_i$=

$$\frac{\sum_{j\neg=i}^{n} \mathbf{e}_j^2}{(\mathbf{n-1-p})}$$

$\mathbf{h}_{ii}$ is the ith diagonal element of matrix H=$\mathbf{X}(\mathrm{X}^T X)^{-1} X^T$

Traditionally cut-off is given by :

$$|\mathbf{DFFITs}|_i > \frac{\mathbf{2} * \sqrt{p}}{\sqrt{n}}$$

and if $DFFITS_i$ is greater than this cut-off that indicates that ith sample point is influential point. In this case 0.3425395 is consider as threshold.we found there are 28 influential point based on the DFFits.

```
> #DFFIT
> #'RSSi' vector of residual after removing ith obseravation from training data
> RSSi=sum(res^2)-(res^2/1-h)
> #'MSresi' Estimate of error variance after removing ith observation
> MSresi=RSSi/(n-1-p)
> #'DFFits' is change the predicted response after removing ith observation
> DFFits=(res*sqrt(h))/(sqrt(MSresi)*(1-h))
> # 'a' is Influencial observation on the basis of DFFits
> thresnew=2*sqrt(p/n)
> thresnew
[1] 0.3425395
> a=which(abs(DFFits)>thresnew)
> length(a)
[1] 28
>
```

Figure 11: Code for calculation of influential points using DFFITS

```
> ##DFFits plotting
> plot(DFFits,xlab='Observations',col=(DFFits>(2*sqrt(p/n)))+1,
+      main='fig:1.2 DFFITs PLOT',pch=20)
> abline(h=2*sqrt(p/n),col='blue',lwd=2,lty='dotted')
> legend("topright",c("Influential Point","Non Influential Point")
+        ,col=c("red","black"),lwd=0.2)
>
```

Figure 12: Codes for graphical representation of DFFITS using threshold=0.3425395

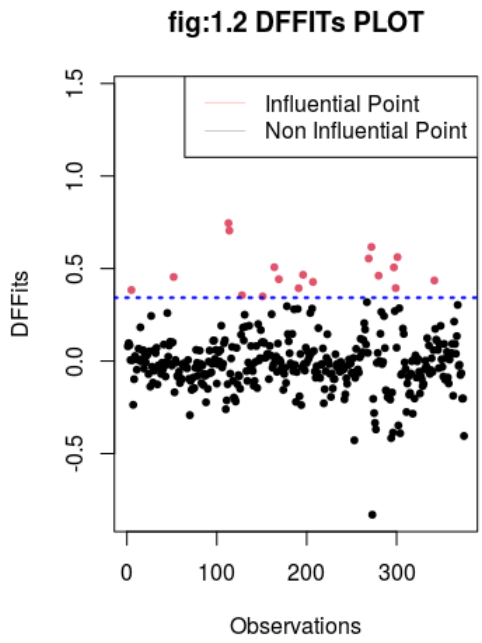

Figure 13: DFFITS graphical representation of DFFITS using threshold=0.3425395

8.2.3 DFBETAS

It is also used for influential point detection and it also contain both part (r_i^{*2} and h_{ii}) which measure how far observation is from data and how well the model fits the ith observation y_i, which is clear from formula of $DFBETAs_{ji}$ $DFBETAs_{ji}$ is calculated as:

$$\mathbf{DFBETAs}_{ji} = \frac{r_i^* * C_j}{\sqrt{h_{jj}(1-h_{ii})}}$$

r_i^* is the external studentized residual. $\mathbf{h}_{ii}$, $\mathbf{h}_{jj}$is the ith,jth diagonal element of matrix H=$\mathbf{X}(X^TX)^{-1}X^T$ respectively and C_{ji} is the is j^{th} row of the matrix C= $((\mathbf{X^TX})^{-1}\mathbf{X^T}$

```
> #DFBETA
> DFBETAs_ji=matrix(data=NA,nrow=n,ncol=p-1)
> Cd=solve(t(X_train)%*%X_train)%*%t(X_train)
> for(j in 1:p-1)
+ {
+   DFBETAs_ji[,j]=(Cd[j+1,]*res)/(sqrt(sum(Cd[j+1,]^2)*MSresi)*(1-h))
+ }
> influence_DFBETA=NA
> for(j in 2:p)
+ {
+   influence_DFBETA=c(influence_DFBETA ,which(abs(DFBETAs_ji[,j-1])>2/sqrt(n)) )
+ }
> b = as.numeric (names (which( table(na.omit(influence_DFBETA))>=7 ) ) )
> length(b)
[1] 5
> # b is index of influential point based on DFBETAs_ji
```

Figure 14: Code for calculation of influential points using $DFBETAS_{ji}$

Generally threshold is given as for this case

$$\mathbf{DFBETAs}_{ji} > \frac{\mathbf{2}}{\sqrt{n}}$$

And if $DFBETAS_{ji}$ exceeds this cut off than we say that ith observation is influential.From Fig.14 it is clear that only those points are considered to be influential points based on DFBETAs which are considered to be influential by 7 or more regressors coefficient out of 10 regressors coeffficient tested for DFBETA. We found 5 influential point based on DFBETAs.

8.2.4 COVRATIO

The $Cook's distance$,$DFFITS_i$,$DFBETAS_{j,i}$ provides insight about the effect of observation on estimates of coefficients and fitted value but they do not provide precision of estimation. COVRATIO is used for this purpose.We express the role of ith observation on precision of estimates in terms of $COVRATIO_i$.
We define $COVRATIO_i$ as

$$\mathbf{covratio_i} = \frac{(MSres_i)^p}{(MSres)^p * (1-h_{ii})}$$

$MSres_i$ is the esimate of the variance of response variable when ith data point is removed from the data.$MSres$ is the esimate of the variance of response variable using all the data.

$\mathbf{h}_{ii}$ is the ith diagonal element of matrix H=$\mathbf{X}(X^TX)^{-1}X^T$.

```
> #COVRATIO
> #cov_ratio helps to compare model accuracy when ith observation is removed
> #whether accuracy improved or not after removing the ith observation
> cov_ratio=(MSresi/MSres)^p/(1-h)
> #'c' is influential observation on the basis of cov_ratio
> c=which(abs(cov_ratio-1)>3*p/n)
> length(c)
[1] 28
```

Figure 15: Code for computation of influential points using COVRATIO

If $COVRATIO_i > 1$ then ith observation improves the precision of estimates and if $COVRATIO_i$ <1 then ith observation degrades precision. Traditionally, Cut-off for COVRATIOS is defined as

$$|\mathbf{covratio_i} - \mathbf{1}| > \frac{\mathbf{3 * p}}{n}$$

There is 28 influential point using this approach

```
> ##Covariance ratio plotting
> plot(cov_ratio,main="fig:1.3 COVRATIO PLOT",col=(abs(cov_ratio-1)>3*p/n)+1, pch=20)
> abline(h=c( 3*p/n +1, -(3*p/n)+1) , col="red", lwd=2, lty="dotted")
> legend("bottomleft",c("Influential Point","Non Influential Point"),col=c("red","black"),pch=20)
>
>
>
```

Figure 16: COVRATIO codes for graph representation

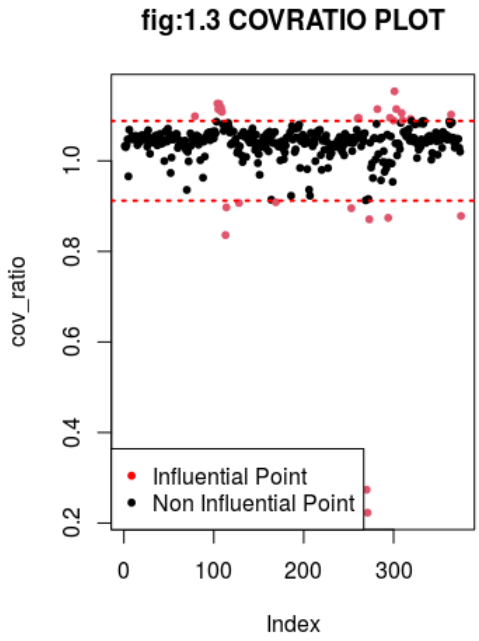

Figure 17: COVRATIO graph representation

Now taking **all of the unique influential observaion's** we get on the basis of DFFITs , DFBETA , COV RATIO.

```
>
> # Now taking all of the unique influential observaion's we get on the basis #of DFFITs,DFBETA , COV RATIO
> a;b;c
 [1]   5  52 113 114 128 151 164 169 191 196 207 253 269 270 271 272 273 277 280 294 296 297 299 301 302 304 342
[28] 375
[1] 113 114 270 271 272
 [1]  79 104 105 106 107 108 109 113 114 128 169 253 260 261 270 271 273 282 294 296 300 301 303 309 310 319 364
[28] 375
> z=unique(c(a,b,c))
> length(z)
[1] 44
> z
 [1]   5  52 113 114 128 151 164 169 191 196 207 253 269 270 271 272 273 277 280 294 296 297 299 301 302 304 342
[28] 375  79 104 105 106 107 108 109 260 261 282 300 303 309 310 319 364
> #Removing influential observation from the trainin data
> Y_train=Y_train[-z]
> X_train=X_train[-z,]
>
> # observation left after removing leverage and influntial point from data
> dim(X_train);length(Y_train)
[1] 331  11
[1] 331
>
```

Figure 18: Codes for overall influencial points

Overall there are 44 influential points. After deleting leverage and influential points we have now 331 observations are left.

```
> #Least square solution after removing of leverage and influential point
> beta_hat2=solve(t(X_train)%*%X_train)%*%t(X_train)%*%Y_train
> # estimated value of the  mean housing price of testing data
> Y_test_cap=X_test%*%beta_hat1
> #RMSE Value of the testing the datasets
> RMSE_new=sqrt(sum((Y_test-Y_test_cap)^2))/sqrt(length(Y_test))
> RMSE;RMSE_new
[1] 5.9042
[1] 5.698279
> if(RMSE_new<RMSE){RMSE=RMSE_new}
> #RMSE_new is less than RMSE .so , the effect of leverage and Outliers is #diagnoised successfully.
>
```

Figure 19: codes for new RMSE after deleting all influential points.

After deleting leverage and influential points RMSE is 5.698279 which is less than the previous RMSE(5.9042) which implies it is better model than previous ones. That means There is significant effect of Outliers in the previous model trained using train dataset.

9 Dealing with Curvature

Here we check that if any curvature or nonlinear pattern are present in residual with repect to predicted response and regressor variable , there are some task here

1.) First we check whether the residuals show any pattern when plotted against the individual regressors. Identify at least one regressor which does not seem linearly related with E(y).

2.) Check the difference between Augmented Partial Residual (APR) and Component Plus Residual (CPR) plots to justify our claim.

3.) Choose a suitable transformation of the regressor(s) (that we tested in (2.) which is linear in E(y). We can get an idea of the appropriate transformation from the plot of partial residuals of y vs the regressor(s), or simply from the plot of errors vs the regressor(s).

4.) Verify (using correlation coefficient) if linearity has improved by using the transformed regressor(s).

5.) Change the regressor(s) chosen in part (3.) by the transformed one. Compute $\hat{y}_{test}$ and RMSE again. Keep the change if we get a non-increasing RMSE after transformation, otherwise do not keep the transformation.

9.1 Plot of residuals vs individual regressors

```
n=nrow(X_train) ; p=ncol(X_train)
res=(Y_train-X_train%*%beta_hat2)
par(mfrow=c(2,(p-1)/2))
for(i in 2:ncol(X_train))
{
  plot(X_train[,i],res,xlab=colnames(X_train)[i],ylab="res",col='royalblue',pch=20)
}
mtext("fig:2.1 Residual versus regressor plot",line=-2 ,outer=TRUE)
```

Figure 20: Codes for plot

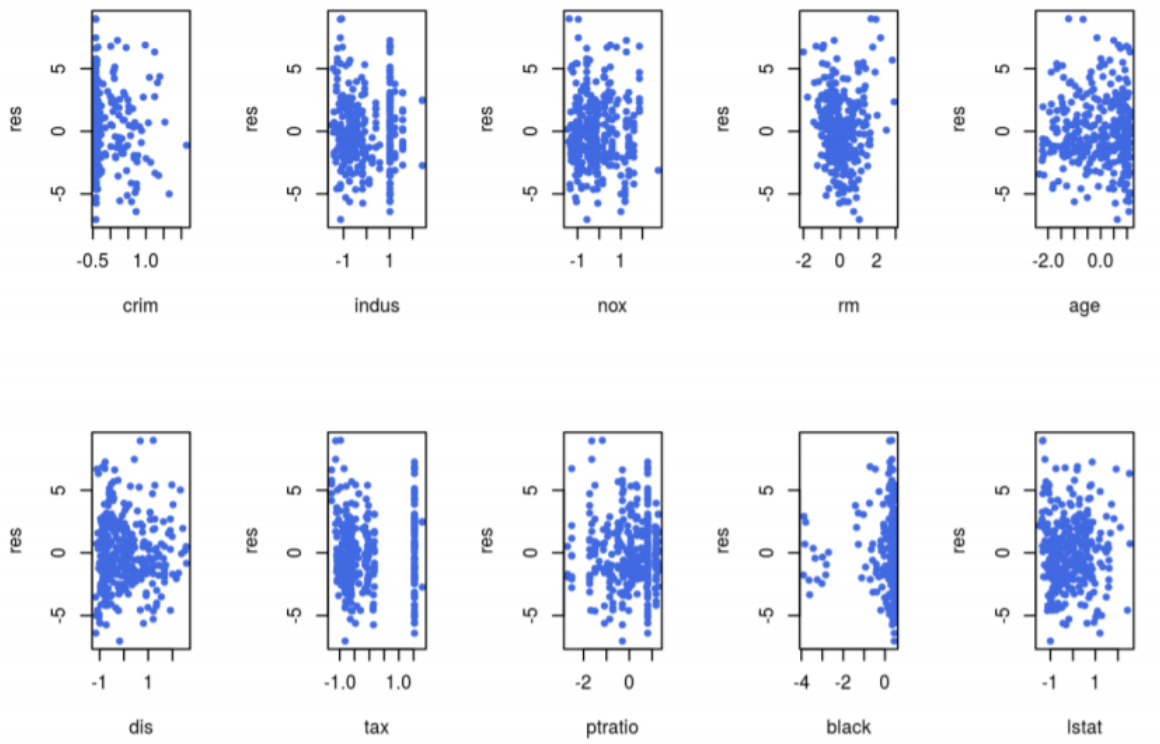

Figure 21: residual vs individual plots

Clearly from the plot of residual vs regressor plot we seem **"rm"** not to be **linearly**

related with E(y).

9.2 Checking APR and CPR Plot

Here we plot APR and CPR PLot with respet to regressor 'rm'.Further , after examining the APR and CPR plot we have to verify our claim of non-linearity of regressor 'rm' with respect to residual.

Checking CPR Plot.

```
>
> #plotting 'rm' with APR(Augmented Partial Residual )
> X_train_apr=cbind(X_train,X_train[,l]^2)
> Y_train_apr=Y_train
> beta_hat_apr=solve(t(X_train_apr)%*%X_train_apr)%*%t(X_train_apr)%*%Y_train_apr
> res_apr=Y_train_apr-X_train_apr%*%beta_hat_apr
> apr=res_apr+beta_hat_apr[l]*X_train[,l]+beta_hat_apr[length(beta_hat_apr)]*(X_train[,l]^2)
> plot(X_train[,l],apr,cex.main=0.75,
+      main="fig:2.3 Augmented Partial Residual plot",xlab="rm",
+      ylab="augmented residual", col="blue")
> #Setting for drawing slope in 'rm' V/S APR(Augmented Partial Residual) Plot
> Y_train_apr=apr
> X_train_apr=cbind(rep(1,n),X_train[,l])
> coef=solve(t(X_train_apr)%*%X_train_apr)%*%t(X_train_apr)%*%Y_train_apr
> abline(a=coef[1],b=coef[2],col="red")
>
```

Figure 22: Codes for CPR Plot

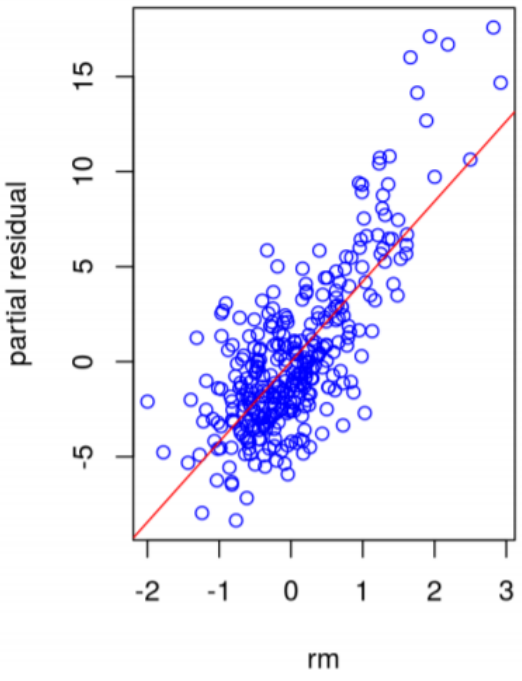

Figure 23: CPR Plot

Checking APR plot

```
> #2(b)Plotting 'rm' regressor with CPR(component plus residual)
> par(mfrow=c(1,2))
> l=which(colnames(X_train)=="rm")
> #'cpr' is the component plus residual
> cpr=res+beta_hat2[l]*X_train[,l]
> #plotting the 'rm' V\S CPR
> plot(X_train[,l],cpr,xlab="rm",ylab="partial residual",cex.main=0.85,main="fig:2.2 Component plus resi
lue")
> #Setting for drawing slope in 'rm' V/s cpr plot
> Y_train_new=cpr
> X_train_new=cbind(rep(1,n),X_train[,l])
> coef=solve(t(X_train_new)%*%X_train_new)%*%t(X_train_new)%*%Y_train_new
> abline(a=coef[1],b=coef[2],col="red")
>
```

Figure 24: codes for APR plot

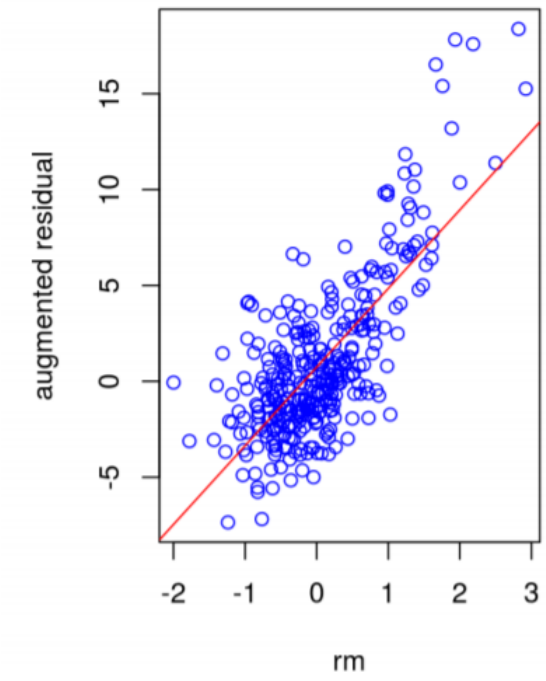

Figure 25: plot for APR

Clearly from Observing CPR and APR plot. We suggest that the **exponential transformation** is needed for the regressor **'rm'**.Thus , our claim of non linearity of regeressor 'rm' is evident.

9.3 Transformation of regressors

After critically examining APR and CPR plot .we suggest that exponential transformation is needed for 'rm' regressor .Our transformed model is :

$$g(x) = \beta_0 * \exp(\beta_1 * X_{trainrm})$$

Fitting the transformed model of variable "rm"

```
>
> #2(c)Observing CPR and APR plot. We suggest that the exponential #transformation is needed for the regressor 'rm'
> #Our transformed model for 'rm' is  g(x)=beta0*exp(beta1*Xtr_rm)
> #Fitting the transformed model
> X_train_rm=cbind(rep(1,n),X_train[,l])
> Y_train_rm=Y_train
> beta_hat_rm=solve(t(X_train_rm)%*%X_train_rm)%*%t(X_train_rm)%*% log(Y_train_rm)
> beta_hat_rm
          [,1]
[1,] 3.0371211
[2,] 0.2978245
> trans_rm= exp(beta_hat_rm[1])*exp(beta_hat_rm[2]*X_train[,l])
> trans_rm
```

Figure 26: codes for transformation

Now we find partial correlation coefficient between residual and "rm" and transformed 'rm' of Xtrain.

```
> ##2(d). to find partial coefficient between Ytrain and "rm" of Xtrain
> X_train_dd=X_train[,-l]
> uu=lm(Y_train~X_train_dd)
> res1=uu$residuals
> vv=lm(X_train[,l]~X_train_dd)
> res2=vv$residuals
> #Correlation coefficient without any transformation
> par_Cor=cor(res1,res2)
> par_Cor
[1] 0.6289756
> ## to find partial coefficient between Ytrain and g(x)
> ww=lm( trans_rm ~X_train_dd)
> res2=ww$residuals
> #Correlation coefficient after transformation
> par_Cor_trans= cor(res1,res2)
> par_Cor_trans
[1] 0.6853392
> ##there is increase in  correlation coefficient ,
> #thus we keep the  transformation of 'rm' and conclude that linearity has #improved
> #Replacing the transformed 'rm' with 'rm' in train dataset
```

Figure 27: codes for Correlation coefficient without any transformation and with exponential transformation.

Clearly we see that without any transformation the partial correlation coefficient between residual and 'rm' is less than our transformed one so that indicate improvement in the linearity of regressor 'rm' with residual .therefore our transformation seems good, now lets change our design matrix with replace X_{rm} with g(x) and then find fitted our OLS model to find RMSE again and verify our result.

New model and finding RMSE again.

```
 #Replacing the transformed 'rm' with 'rm' in train dataset
 X_train[,l]=trans_rm
 colnames(X_train)[l]='transformed_rm'
 colnames(X_train)
[1] "intercept"      "crim"           "indus"          "nox"            "transformed_rm" "age"
[7] "dis"            "tax"            "ptratio"        "black"          "lstat"
 #Estimated coefficient after transformation of model fit
 beta_hat=solve(t(X_train)%*%X_train)%*%t(X_train)%*% Y_train
 #Replacing the transformed 'rm' with 'rm' in train dataset
 Xts1=X_test
 Xts1[,l]= exp(beta_hat_rm[1])* exp(beta_hat_rm[2]*X_test[,l])
 X_test=Xts1
 #2(e)
 RSS= sum((Y_test-X_test%*%beta_hat)^2)
 #Calculation of RMSE after transformation of 'rm'
 RMSE_trans=sqrt(RSS/length(Y_test))
 RMSE_trans ; RMSE
1] 5.128485
1] 5.698279
 if(RMSE_trans<RMSE){RMSE=RMSE_trans}
 RMSE
1] 5.128485

 #from the result we get that RMSE_trans is less than RMSE . thus transformation has  improved linearity of the res
dual
```

Figure 28: codes for new transformed design matrix and again for new RMSE.

Clearly our new RMSE after transformation is less than the pevious RMSE (5.698279) . RMSE after transformation is 5.128485. This indicate a significant effect of non-linearity of 'rm' regressor with respect to residual on our previous model .Thus after transformation our model accuracy has been approved . Hence our further analysis will be done using our transformed model.

10 Dealing with Heteroscedasticity

One of the assumption of our Linear regression model is that error variance is homoscedastic. Consider the regression equation:-

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_{10} x_{i10} + \epsilon_i . \, i = 1(1)n$$

In this model, if ϵ_i has constant variance say σ^2 than data is said to be homoscedastic and if variance of ϵ_i depends on i then data is said to be heteroscedastic and it is said that heteroscedasticity is present in data.

Here we have some task;

1.) check whether the residuals show any pattern when plotted against the fitted values, and then by plotting w.r.t. the individual regressors. Find the regressors with respect to which the variance (We can approximate by e_i^2) is not constant.

2.) Considering $d_i^{'} = \frac{ne_i^2}{\sum_{i=1}^{n} e_i^2}$, then find appropriate variables $z_1, ..., z_k$, so that the following relation is approximately true:

$$d_i^{'} = \alpha_0 + \alpha_1 z_{1,i} + \alpha_2 z_{2,i} + ...\alpha_k z_{k,i} + \epsilon_i^{'}$$

where $\epsilon_i^{'} \overset{iid}{\sim} \mathbf{N(0, 1)}$. Hence, perform a Breusch-Pagan test and verify if heteroscedasticity is present in this data.

3.) If the Breush-Pagan test rejects the homoscedasticity hypothesis, then we consider an appropriate function $\sigma_i^2 = h(Z, \alpha, \beta)$, and estimate σ_i^2 and simultaneously. [Also we may consider a linear function, i.e., $h(Z, \alpha, \beta) = \alpha_0 + \sum_{j=1}^{k} \alpha_j z_{j,i}$ However,We note that this choice of h may not result in positive values. We may estimate the $\hat{\beta}$ and $\hat{\alpha}$ by using this the following iterative algorithm:

Algorithm 1:

Start with $\hat{\underset{\sim}{\beta}}_{(0)} = \left(X^T X\right)^{-1} X^T \underset{\sim}{y}$;

while $\|\hat{\underset{\sim}{\beta}}_{(I)} - \hat{\underset{\sim}{\beta}}_{(I-1)}\|^2 \leq 0.1$ *and* $\|\hat{\underset{\sim}{\alpha}}_{(I)} - \hat{\underset{\sim}{\alpha}}_{(I-1)}\|^2 \leq 0.1$ **do**

In the $I-$th iteration,

(1) update $\hat{\underset{\sim}{\alpha}}_{(I)}$ by minimizing

$$S = \sum_i \left(e_i^2 - h(Z, \underset{\sim}{\alpha}_{(I-1)}, \hat{\underset{\sim}{\beta}}_{(I-1)})\right)^2 .$$

(2) Using the updated $\hat{\underset{\sim}{\alpha}}_{(I)}$ and $\hat{\underset{\sim}{\beta}}_{(I-1)}$, update $\hat{\Sigma}_{(I)} = Diag\,(h_1, \ldots, h_n)$.

(3) Update $\hat{\underset{\sim}{\beta}}_{(I-1)} = \left(X^T \hat{\Sigma}_{(I)}^{-1} X\right)^{-1} X^T \hat{\Sigma}_{(I)}^{-1} \underset{\sim}{y}$.

(4) Set $I = I + 1$.

end

4.) Once the algorithm converges, We again find the RMSE with the new estimate of β.

10.1 Plots of residual vs fiited response.

```
res=(Y_train-X_train%*%beta_hat)
par(mfrow=c(2,(p-1)/2))
for(i in 2:p)
{
  plot(X_train[,i],res,xlab=colnames(X_train)[i],ylab="residual", pch=10, col="blue")
}
mtext("fig:3.1 Residual V/S regressor plot",line=-2 ,outer=TRUE)
par(mfrow=c(1,1))
Y_train_cap=X_train%*%beta_hat
plot(Y_train_cap,res,xlab="fitted values",ylab="residual",main="fig:3.2 fitted values V/S residuals",col="red",pch=20)

# Examining the plot of residuals V/S regressor and residual V/S fitted #values,we claim that crim, indus, age, dis, tax,
#black produce heteroscedasticity
```

Figure 29: codes for plot of residual vs individual regressor and then with fitted response.

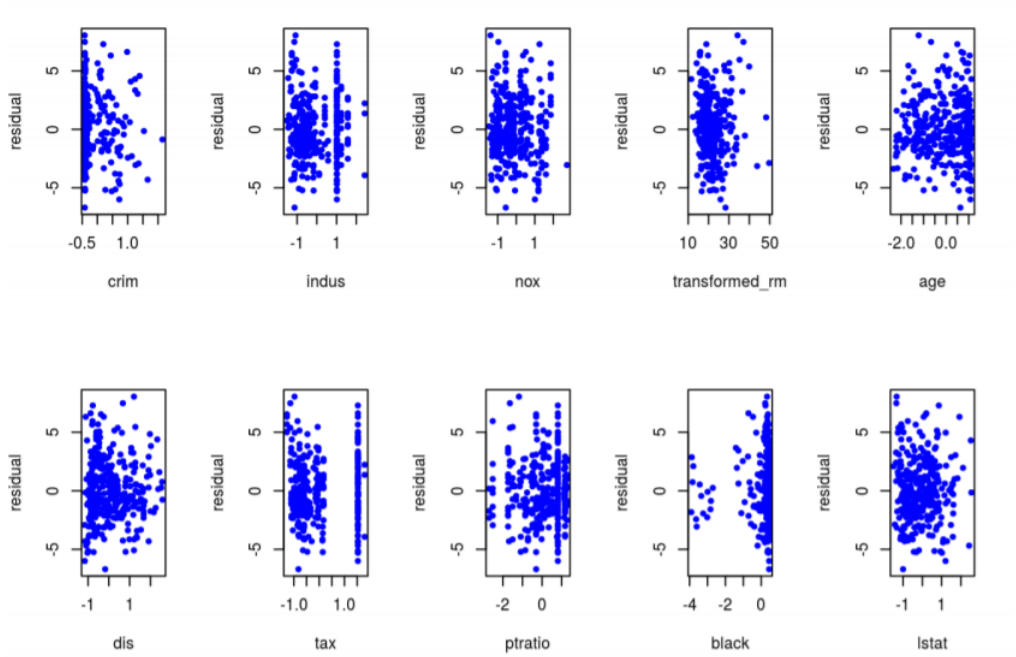

Figure 30: plot of residual vs individual regressors.

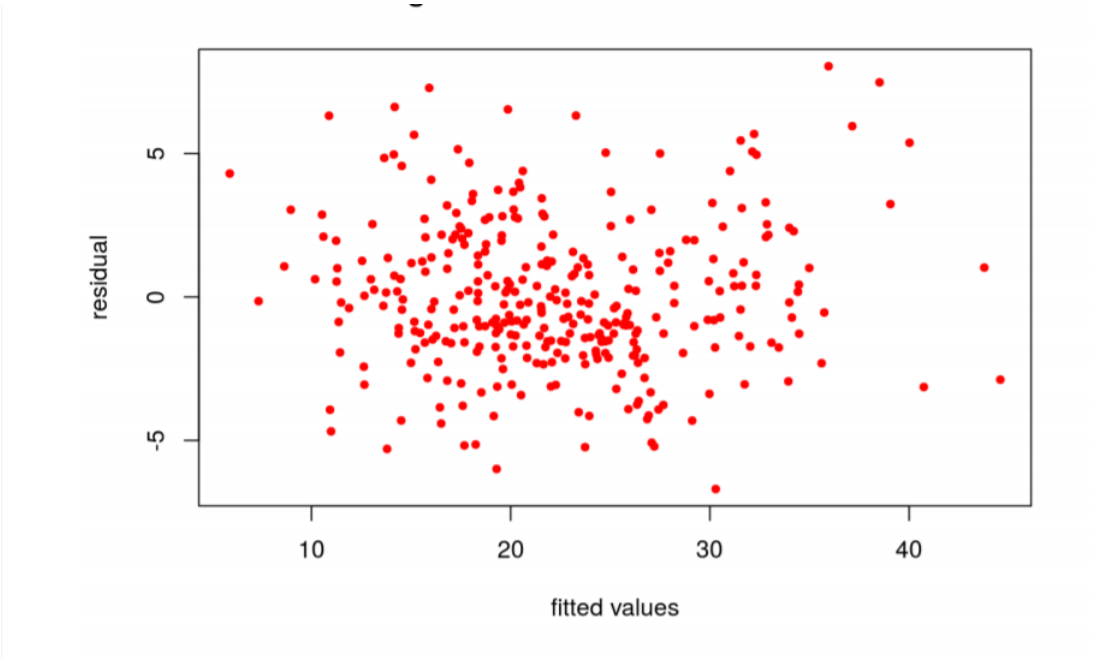

Figure 31: plot of rersidual vs fitted response.

Clearly from the plot of residual vs individual regressors and of fitted response ,we claim that **CRIM, INDUS, AGE, DIS, TAX, BLACK** produce **Heteroscedasticity.**

10.2 Testing of presence of Heteroscedasticity with Breusch Pagan Test

.

We test the hypothesis,

H_0: Error variance is Homoscedastic.

H_1: Error variance is Non- Homoscedastic.

```
>
> #3(b) Performing Breusch Pagan Test to test the heteroscedasticity claim    #Ho: Error variance are homoscedastic
in nature
> #H1: Error variance are Heteroscedastic in nature
> di=(n*res^2)/sum(res^2)
> Ztr=X_train[,which(colnames(X_train)=='intercept'|colnames(X_train)=='crim'|colnames(X_train)=='indus'|colnames(X_
train)=='age'|
+                      colnames(X_train)=='dis'| colnames(X_train)=='tax'|colnames(X_train)=='black')]
> di_hat=Ztr%*% solve(t(Ztr)%*%Ztr) %*% t(Ztr)%*%di
> R_2=(cov(di,di_hat))^2/(var(di)*var(di_hat))
> #Test Stastic under the assumption of null hypothesis
> Q=n*R_2
> print(Q)
         [,1]
[1,] 16.44206
> #Critical Value
> c= qchisq(0.95,6)
> print(c)
[1] 12.59159
> #Decision Rule
> if(Q>c) print("reject H0:(homoscedasticity assumption)") else
+   print("accept H0:(homoscedasticity assumption)")
[1] "reject H0:(homoscedasticity assumption)"
>
```

Figure 32: codes for testing the hypothesis.

Clearly from the results Chi square test statistic Q= 16.44206 which is greater than the critical value (12.59159) ,Hence we reject the null hypothesis that means the errors are heteroscedastic. So now lets perform itteration for converging the algorithm to find appropriate β.

```
> #Performing Iterative method to estimate the coefficients
>
> I=0
> beta_hat=solve(t(X_train)%*%X_train)%*%(t(X_train)%*%Y_train)
> res=Y_train- X_train%*%beta_hat
> alpha= solve(t(Ztr)%*% Ztr) %*%t(Ztr)%*% res^2
> alpha_1=NA
> repeat{
+   beta_hat_aux=beta_hat
+   alpha_1= alpha
+   sig= Ztr %*% alpha_1
+   sigma=diag(as.vector(sig))
+   beta_hat=solve(t(X_train)%*% solve(sigma) %*%X_train)%*% t(X_train)%*% solve(sigma)%*% Y_train
+   res= Y_train- X_train%*%beta_hat
+   alpha= solve(t(Ztr)%*% Ztr) %*%t(Ztr)%*% res^2
+   I=I+1
+   if((sum((beta_hat_aux-beta_hat)^2)<=0.1 & sum((alpha_1-alpha)^2)<=0.1)| I>200)
+   {      break   }
+ }
>
```

Figure 33: codes for itteration.

10.3 Now fit our model again and find RMSE.

```
>
> #3(d)Calculating fitted model coefficient after removing the effect of #heteroscedasticity with transformation
> RSS= sum((Y_test-X_test%*%beta_hat)^2)
> RMSE_new=sqrt(RSS/length(Y_test))
> RMSE; RMSE_new
[1] 5.128485
[1] 5.106228
> # decrease in RMSE
```

Figure 34: codes for new transformed design matrix and again for new RMSE.

Clearly we see that new RMSE(after removing heteroscedasicity with proper transformation) is less than the previous one .Thus there is significant effect of heteroscedasticity in the previous model .So, now we have successfully corrected the effect of heteroscedasticity.Hence , our model accuracy has been improved.

Now this is our new model after removal of outliers(leverage and influencial) ,diagnosing curvature and now removing hetroscedasticity everything is good now.Lets move fo normality assumption.

11 Dealing with the Non-Normality

In section 7.1 we assumed $\epsilon_i \overset{iid}{\sim} \mathbf{N}(\mathbf{0}, \sigma^{\mathbf{2}})$ which is equivalent to assume $\mathbf{y} \overset{iid}{\sim} \mathbf{N}_n(X\beta, \sigma^2 I_n)$. Now as y is observed quantity, we will check only its Normality. For this we have to do some task here;

1.) Firtly we Draw the QQ-plot of the R-student residuals w.r.t. it's population distribution and Comment our results on the normality assumption based on the QQ-plot.

2.) If the plot shows departure from normality, let consider an appropriate Box-Cox transformation to normalize the data. Find the parameter λ using profile likelihood. (We may choose the appropriate λ graphically.)

3.) After the optimal λ is chosen, we recompute the R-students residual with transformed response, and draw the QQ-plot again. And from the plot observe the normality assumption on the transformed y to be correct or not.

11.1 Checking Through Plot

We will check Normality by Normal Q-Q Plot. In Normal Q-Q Plot order statistics of vector of interest are drawn in the y-axis corresponding to the theoretical order statistics from t-distribution with (n-p-1) degree of freedom in the x-axis. Normality assumption can be regarded as true if the points almost in a straight line making an 45degree angle with x-axis .

```
> ##(4)---Diagnosing the normality assumption--------------------
> #4(a)
> #qq.plot() function can plot quantile quantile plot
> #input variable is predictor matrix and response vector
> qq.plot=function(Z,y,text)
+ {
+   n=nrow(Z)
+   p=ncol(Z)-1
+   H=Z%*%solve(t(Z)%*%Z)%*%t(Z)
+   #predicted value of y
+   y_cap=H%*%y
+   #vector of residuals
+   res=y-y_cap
+   RSS=sum(res^2)
+   RSS_i=rep(0,n)
+   for(i in 1:n)
+   {
+     RSS_i[i]=RSS-((res[i]^2)/(1-H[i,i]))
+   }
+   #'r_i' is R-Studentized residuals
+   r_i=rep(0,n)
+   for(i in 1:n)
+   {
+     r_i[i]=res[i]/(sqrt(RSS_i[i]*(1-H[i,i])/(n-p-1)))
+   }
+   #ordered R-Student residuals
+   #ordered R-Student residuals
+   r_i.ord=sort(r_i)
+   #'p_i' is vector of population quantiles of t(n-p-1) distn
+   p_i=rep(0,n)
+   for(i in 1:n)
+   {
+     p_i[i]=qt(i/n,n-p-1)
+   }
+   plot(p_i,r_i.ord,pch=20,main=text,xlab="Population quantiles",ylab="Sample quantiles")
+   #straight line with intercept 0, slope 1 drawn to make inferences
+   abline(a=0,b=1)
+ }
> par(mfrow=c(1,3))
> qq.plot(X_train,Y_train,"fig:4.1 Q-Q Plot ")
> #Looking at the qq plot we find that all the points are not in straight line
```

Figure 35: codes for QQ-plot of y.

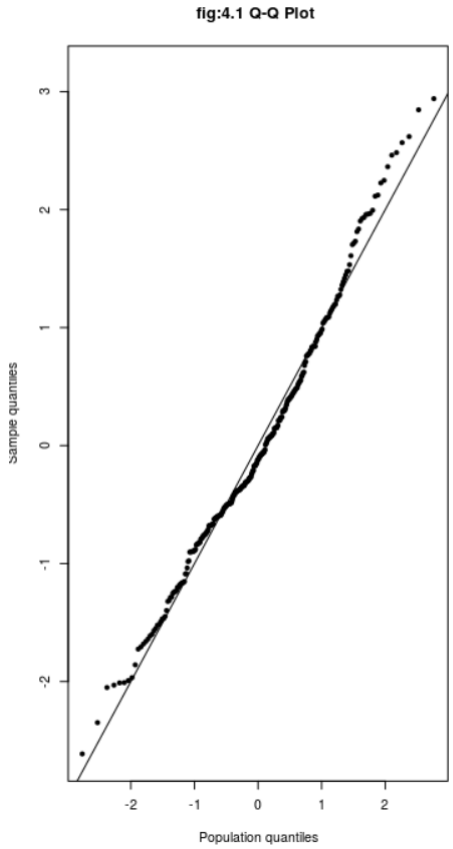

Figure 36: QQ-plot without any transformation.

Looking at the QQ-plot, We find that all the points are not in straight line.Hence the response is not normal.Here , we use Box Cox transformation to transform our response .So, the general linear model assumption of normality of response is satisfied.

11.2 Box-Cox transformation

Performing Box-cox transformation:-

```
> #4(b)Box-Cox Transformation
> Y_train[which(Y_train<0)]
named numeric(0)
> ## all values are positive
> lamda=seq(from=-5, to=5, by=0.1)
> Y_trans= NA
> lik_fun=NA
> for( i in 1:length(lamda))
+ {
+   lam=lamda[i]
+   if(lam!=0)
+   {  Y_trans= (Y_train^lam - rep(1,times=n))/lam  }else
+   {  Y_trans= log(Y_train)  }
+   beta_hat_trans= solve(t(X_train)%*%X_train) %*% t(X_train)%*%Y_trans
+   lik_fun[i]=-(n/2)*( log(2*pi) +log(sum((Y_trans-X_train%*%beta_hat_trans)^2)/n)+1) +
+     (lam-1)* sum(log(Y_train))
+ }
> plot(x=lamda, y=lik_fun, xlab = "lambda", ylab="L(lambda)", pch=20,main="fig:4.2 likelihood as funcn of lambd
a")
>
> lam= lamda[which(lik_fun==max(lik_fun))]
> lam
[1] 1
> abline(v= lam, col="red", lty="dotted", lwd=2)
>
```

Figure 37: codes for maximised λ.

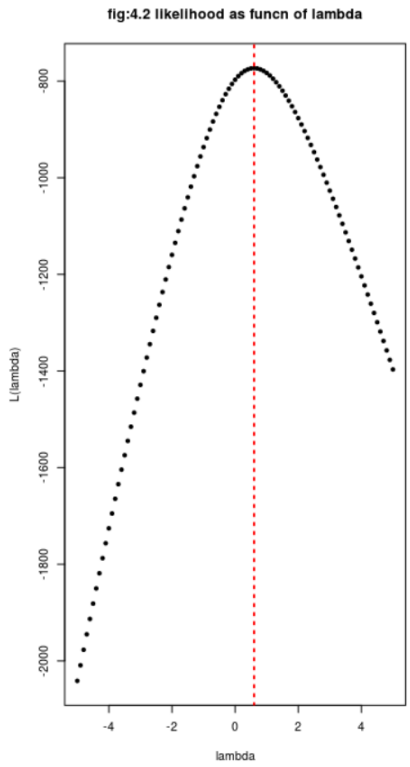

Figure 38: plot of MLE of λ.

Clearly from the graph and R output the value of λ which maximize the profile likelihood is 0.6. Now lets transform our response.

11.3 Transformed response and Checking Q-Q plot again

Now , with the use of transformed response we are again checking nomal Q-Q plot

```
> ytrain_aux= (Y_train^lam - rep(1,times=n))/lam
> qq.plot(X_train, ytrain_aux,'fig:4.3 Q-Q plot after Box Cox transformation')
>
> ## normality assumption seems to have improved as thus obtained plot is closer to ideal situation.
> ## thus we keep the transformation
> Y_train= ytrain_aux
```

Figure 39: Codes of transformation of y.

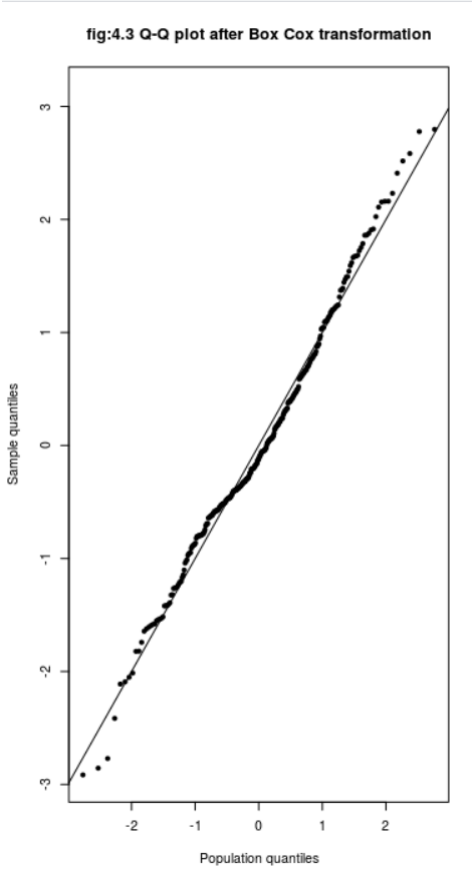

Figure 40: QQ-plot after transformation.

From the trasformed QQ-plot it is clearly seems normality assumption seems to have improved as thus obtained plot is closer to ideal situation.

Now finally calculate our RMSE with respect to this model.

```
> ## thus we keep the transformation
> Y_train= ytrain_aux
>
> beta_hat_final=solve(t(X_train)%*%X_train)%*%t(X_train)%*% Y
_train
> beta_hat_final
                    [,1]
intercept       5.32655724
crim           -0.46115639
indus          -0.15854193
nox            -0.17175952
transformed_rm  0.15569411
age            -0.30314938
dis            -0.51781305
tax            -0.02613209
ptratio        -0.39179899
black           0.30797729
lstat          -0.83449504
> RSS_final= sum((Y_test-X_test%*%beta_hat)^2)
> RMSE_new_final=sqrt(RSS_final/length(Y_test))
> RMSE_new_final
[1] 5.106228
> RMSE
[1] 5.106228
>
> #------------------------ THE END---------------------------
```

Figure 41: Coefficient estimates and RMSE for test data

Finally after solving the normality issue we found our final models RMSE is 5.106228 which is same as previous one.AS, we get a non-deccreaing RMSE which mean there is no significant effect of normality assumption on the model that we have considered previously. RMSE is minimum. Hence this model can be taken final for predicting the boston houses median price (MEDV).

12 Conclusion

Our conclusion are as follows:

(1.) Firstly we fitted data on train set and estimated our RMSE on test set .found RMSE to be 5.9042.

(2.)Secondly we remove the leverage and influential point from the training data and observed that our RMSE value is decreased after removal of these points and Hence conclude that there is significant effect of ouliers in our model that is previously obtained from training data .Thus , after these diagnosis we found that our model trained has RMSE equal to 5.698279.

(3.)Now , we check whether there is non-linearity of any regressor with respect to residual .We found that 'rm' is non-linear with respect to residual and corrected it with the exponential transformation and again find the RMSE value for test data with the help of model trained using training data with transformed 'rm' regressor .Finally we found that our RMSE has decreased to 5.128485 which suggest that thier is significant effect of non-linearity of 'rm' regressor in the previous model that is trained without transformation of 'rm' regressor and Thus after the transformation we have improved the model accuracy .

(4.)After the outlier detection and dealing with non-linearity ,Now we checked whether homoscedastic assumption of linear model is satisfied by our model .We performed Breusch pagan test to test for heteroscedasticity and we found that test statistics Q=16.44206 which is greater than the critical value (12.59159).thus we reject homoscedastic assumption for our model and support the claim of hetroscedasticity is present in our model .Now we transformed our model and with the help of iterative method we estimated our coefficient estimates .Now again we calculated RMSE value on test data set found that there is decrease in RMSE value obtained from previous model and now RMSE is 5.106228 which suggest that there is significant effect of heteroscedasticity in our previous model .

(5.) lastly we have validated the normality assumption of response variable with the help of Q-Q plot and we saw that plot does not give us significant evidence of normality of response .Here we use Box-Cox transformation to transform regressor such that normality assumption is satisfied.Again we calculated the RMSE value with transformed regressor and found that RMSE is 5.106228 which is same as previous model and thus we conclude that there is no significant effect of normality of response on the model .

finally after all the diagnosis like oulier detection, dealing with curvature , heteroscedasticity , normality .we get the model as :

$$MEDV = 5.32655724+(-0.46115639)*crim+(-0.15854193)*indus+(-0.17175952)*nox + 0.15669411 * transformed_{rm} + (-0.30314938) * age + (-0.517181305) * dis + (-0.02613209)*tax+(-0.39179899)*ptrratio+0.30797729*black+(-0.83449504)*lstat$$

$$transformed_{rm} = exp(3.0371211) * exp(0.2978245 * rm)$$

RMSE for test data = 5.106228. Thus our aim of building a model with good model accuracy and model interpretability is almost achieved .As in predicting 'MEDV' RMSE of 5.106228 is considerred as small. We have also validated the assumption of linearity , homoscedasticity , normality assumption of General linear model . Now we can predict MEDV(median house price of owner owned house in boston area) with given feature .

13 Bibliography

1. An Introduction to Statistical Learning,with Applications in R by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani
2. Introduction to Linear Regression Analysis by Douglas C Montgomery, Elizabeth A Peck, G. Geoffrey Vining
3. The Elements of Statistical Learning:Data Mining, Inference, and Prediction by Trevor Hastie,Robert Tibshirani,Jerome Friedman
4. https://stackoverflow.com
5. www.analyticsvidhya.com