Code

Linear Regression of Housign(Notebook 1)

Authors:

Jack Asaad
Andrew Sen
Atmin Sheth
Neo Zhao

Date:

10/10/2022

Introduction

The notebook 1 uses the House Price dataset , acquired from Kaggle, the dataset was taged as a linear regession model usage. Because of this you will see best model being Linearn Regreassion. In the Notebook we are comparing 3 models linear regression, KNN and desicion tree Target is Prices of the hour and rest are set as predictors

Hide

```
library(ROCR)
```

Hide

```
library(mccr)
library(caret)
library(tree)
```

Read the data

Hide

```
hp <- read.csv("HousePrices_HalfMil.csv")
summary(hp)
```

```
      Area          Garage        FirePlace         Baths        White.Marble    Black.Marble   
Indian.Marble        Floors           City          Solar       
 Min.   :  1.0   Min.   :1.000   Min.   :0.000   Min.   :1.000   Min.   :0.000   Min.   :0.0000  
Min.   :0.0000   Min.   :0.0000   Min.   :1.000   Min.   :0.0000  
 1st Qu.: 63.0   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:0.000   1st Qu.:0.0000  
1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:0.0000  
 Median :125.0   Median :2.000   Median :2.000   Median :3.000   Median :0.000   Median :0.0000  
Median :0.0000   Median :0.0000   Median :2.000   Median :0.0000  
 Mean   :124.9   Mean   :2.001   Mean   :2.003   Mean   :2.998   Mean   :0.333   Mean   :0.3327  
Mean   :0.3343   Mean   :0.4994   Mean   :2.001   Mean   :0.4987  
 3rd Qu.:187.0   3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.:4.000   3rd Qu.:1.000   3rd Qu.:1.0000  
3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1.0000  
 Max.   :249.0   Max.   :3.000   Max.   :4.000   Max.   :5.000   Max.   :1.000   Max.   :1.0000  
Max.   :1.0000   Max.   :1.0000   Max.   :3.000   Max.   :1.0000  
    Electric          Fiber          Glass.Doors      Swiming.Pool        Garden            Prices     
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   : 7
725  
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:33
500  
 Median :1.0000   Median :1.0000   Median :0.0000   Median :1.0000   Median :1.0000   Median :41
850  
 Mean   :0.5007   Mean   :0.5005   Mean   :0.4999   Mean   :0.5004   Mean   :0.5016   Mean   :42
050  
 3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:50
750  
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :77
975  
```

#splitting to test and train

Looking at the summary of the data set, the target being prices and predictors beign Area,Bath and floors after doing a relation between area and price I saw there may need a inclusion of city which is also coming into play even after thatt ,using all predictor gives the best result

Hide

```
set.seed(1234)
i<- sample(1:nrow(hp),nrow(hp)*0.8,replace=FALSE)
train <- hp[i,]
test <- hp[-1,]
summary(train)
```

```
      Area         Garage        FirePlace          Baths        White.Marble     Black.Marble   
Indian.Marble        Floors            City          Solar       
 Min.   :  1   Min.   :1.000   Min.   :0.000   Min.   :1.000   Min.   :0.0000   Min.   :0.0000  
Min.   :0.0000   Min.   :0.0000   Min.   :1.000   Min.   :0.0000  
 1st Qu.: 63   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:0.0000  
1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:0.0000  
 Median :125   Median :2.000   Median :2.000   Median :3.000   Median :0.0000   Median :0.0000  
Median :0.0000   Median :0.0000   Median :2.000   Median :0.0000  
 Mean   :125   Mean   :2.001   Mean   :2.005   Mean   :2.998   Mean   :0.3331   Mean   :0.3324  
Mean   :0.3345   Mean   :0.4997   Mean   :2.001   Mean   :0.4984  
 3rd Qu.:187   3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.:4.000   3rd Qu.:1.0000   3rd Qu.:1.0000  
3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1.0000  
 Max.   :249   Max.   :3.000   Max.   :4.000   Max.   :5.000   Max.   :1.0000   Max.   :1.0000  
Max.   :1.0000   Max.   :1.0000   Max.   :3.000   Max.   :1.0000  
    Electric          Fiber         Glass.Doors       Swiming.Pool        Garden           Prices     
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   : 7
725  
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:33
500  
 Median :1.0000   Median :1.0000   Median :1.0000   Median :1.0000   Median :1.0000   Median :41
850  
 Mean   :0.5009   Mean   :0.5003   Mean   :0.5002   Mean   :0.5003   Mean   :0.5016   Mean   :42
056  
 3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:50
775  
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :77
975  
```

Data Exploration

Hide

```
dim(train)
```

```
[1] 400000     16
```

Hide

```
head(train)
```

	A... <int>	Gar... <int>	FirePlace <int>	Ba... <int>	White.Marble <int>	Black.Marble <int>	Indian.Marble <int>	Floors <int>	C... <int>
237392	71	2	2	4	0	0	1	0	3
106390	160	2	0	4	1	0	0	1	2
304108	12	1	3	3	0	0	1	1	3
408457	90	3	3	1	1	0	0	0	2

	A... <int>	Gar... <int>	FirePlace <int>	Ba... <int>	White.Marble <int>	Black.Marble <int>	Indian.Marble <int>	Floors <int>	C... <int>
295846	61	3	3	2	0	0	1	0	3
494468	40	3	4	1	1	0	0	1	3

6 rows | 1-10 of 16 columns

Hide

```
#getting the first 500 attribute
Tsample <- train[1:500,]
```

Hide

```
tail(train)
```

	A... <int>	Gar... <int>	FirePlace <int>	Ba... <int>	White.Marble <int>	Black.Marble <int>	Indian.Marble <int>	Floors <int>	C... <int>
174987	191	3	3	5	0	0	1	0	2
295631	212	2	2	3	1	0	0	0	1
328271	86	1	0	1	0	0	1	1	1
492876	102	3	2	3	0	0	1	1	2
25769	61	3	2	1	0	1	0	1	3
495097	67	1	0	1	0	1	0	1	1

6 rows | 1-10 of 16 columns

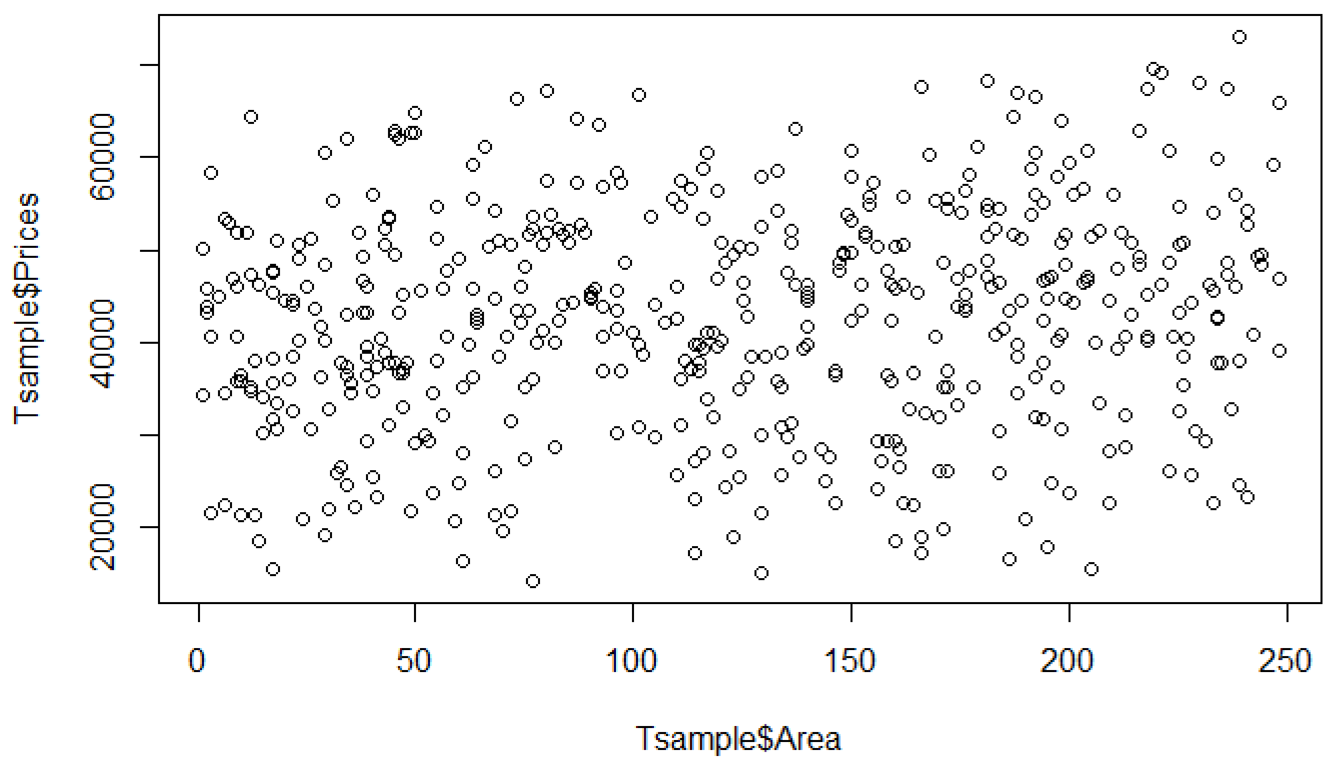
Hide

```
str(train)
```

```
'data.frame':	400000 obs. of  16 variables:
 $ Area        : int  71 160 12 90 61 40 209 126 39 169 ...
 $ Garage      : int  2 2 1 3 3 3 2 2 1 3 ...
 $ FirePlace   : int  2 0 3 3 3 4 0 2 1 2 ...
 $ Baths       : int  4 4 3 1 2 1 1 4 1 2 ...
 $ White.Marble : int  0 1 0 1 0 1 1 1 0 0 ...
 $ Black.Marble : int  0 0 0 0 0 0 0 0 1 0 ...
 $ Indian.Marble: int  1 0 1 0 1 0 0 0 0 1 ...
 $ Floors      : int  0 1 1 0 0 1 0 0 0 1 ...
 $ City        : int  3 2 3 2 3 3 2 2 3 3 ...
 $ Solar       : int  1 0 0 0 0 0 0 1 0 0 ...
 $ Electric    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Fiber       : int  1 0 1 1 0 0 1 0 1 0 ...
 $ Glass.Doors : int  1 0 0 0 1 1 0 0 1 0 ...
 $ Swiming.Pool : int  0 0 0 1 0 0 0 0 0 0 ...
 $ Garden      : int  0 1 1 0 0 1 0 1 1 0 ...
 $ Prices      : int  40475 50250 47300 45250 27975 55950 44475 36150 38425 40475 ...
```

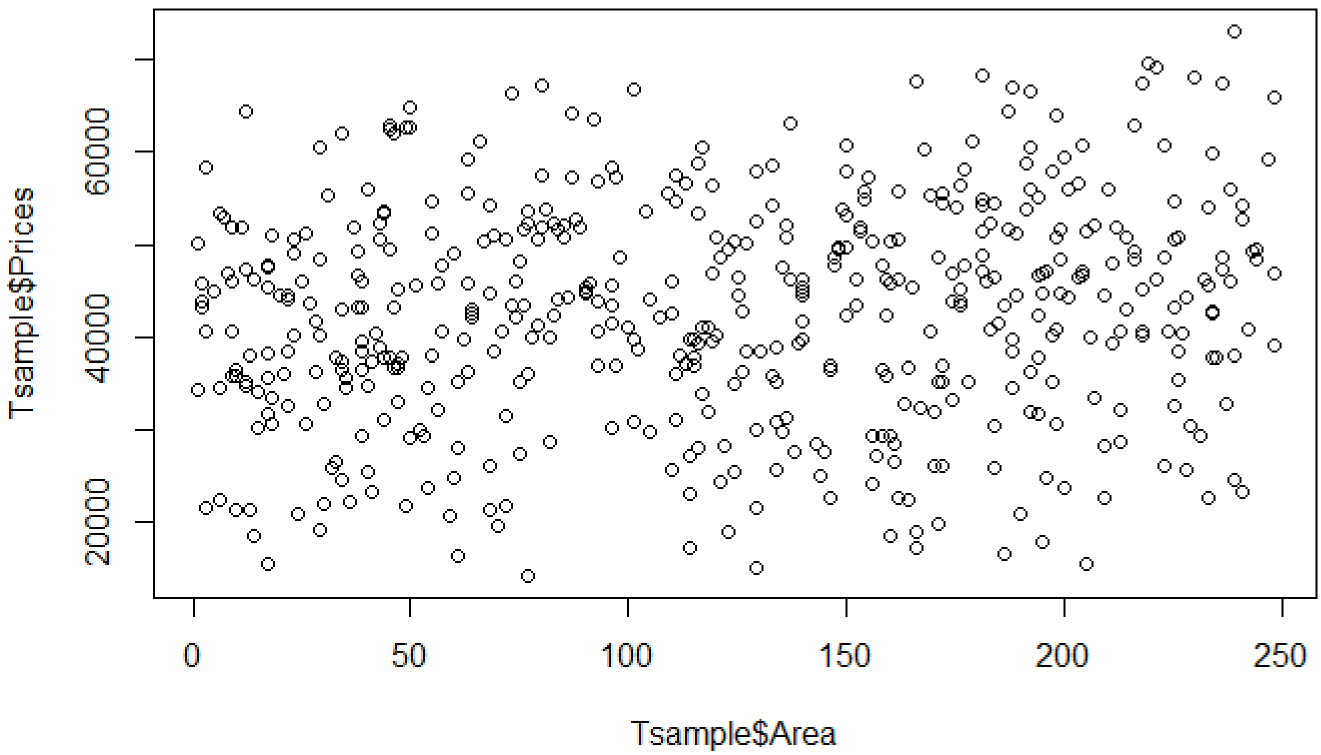
Hide

```
plot(Tsample$Area,Tsample$Prices)
```



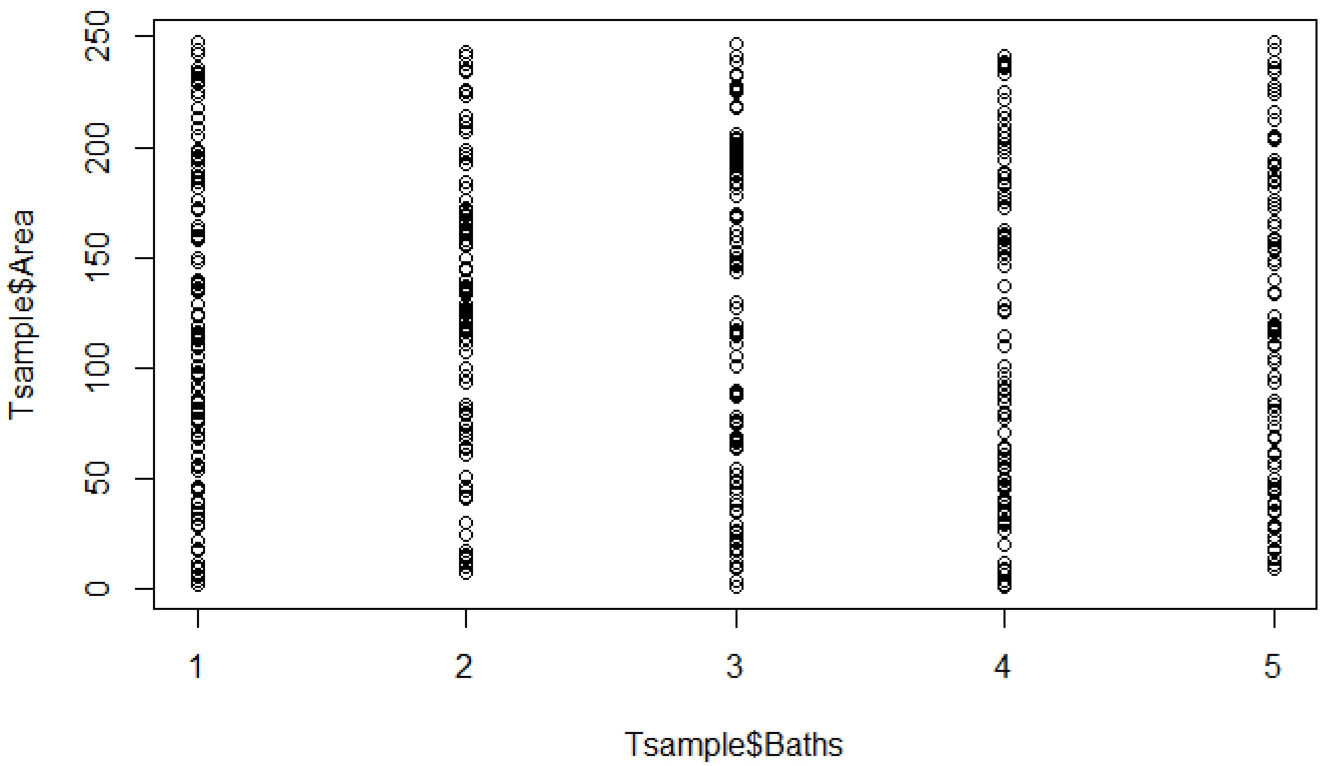
Hide

```
plot(Tsample$Prices~Tsample$Area)
```



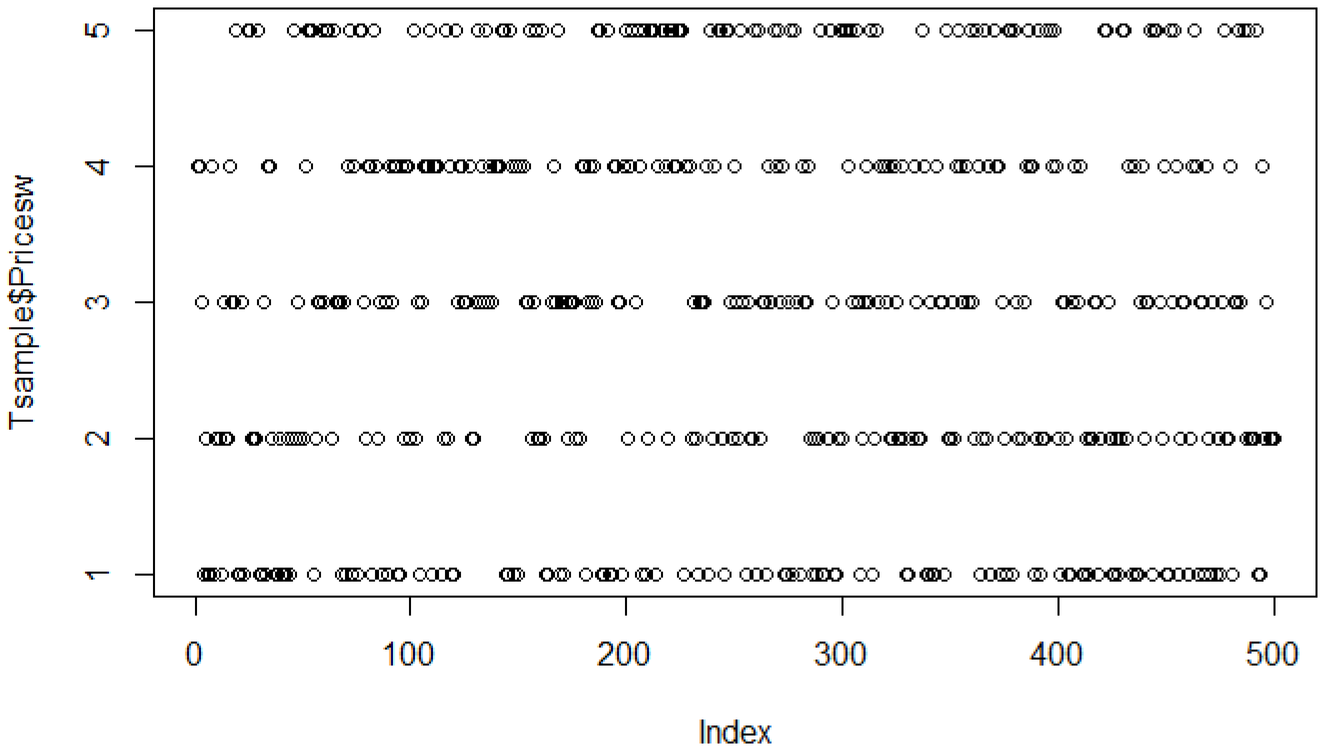
Hide

```
plot(Tsample$Baths,Tsample$Area)
```



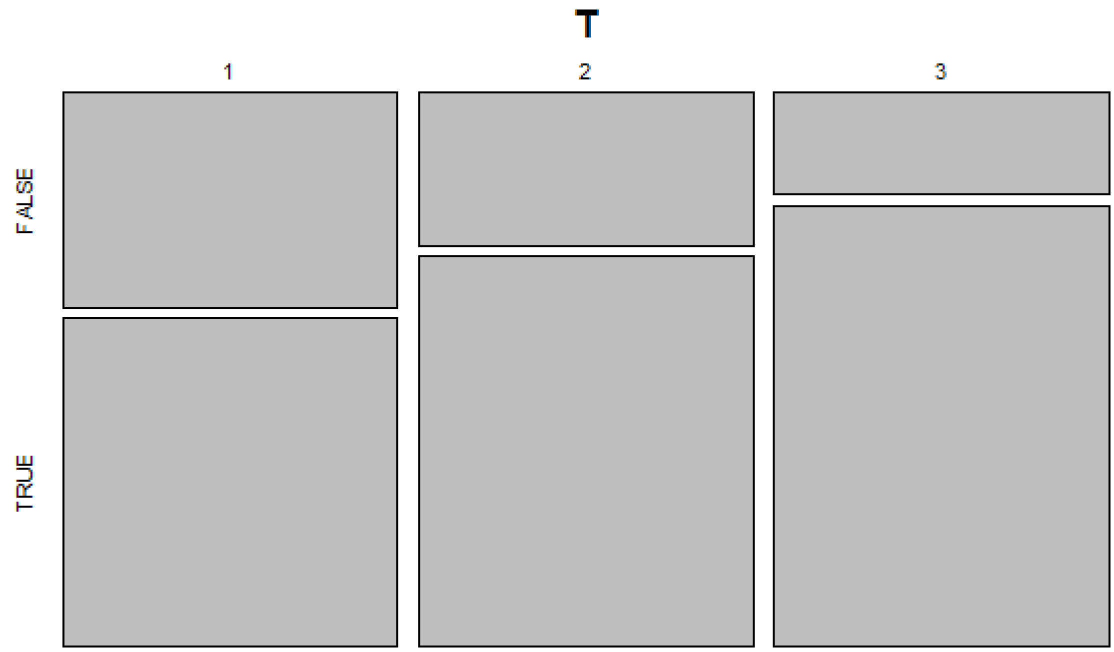
Hide

```
plot(Tsample$Baths,Tsample$Pricesw)
```



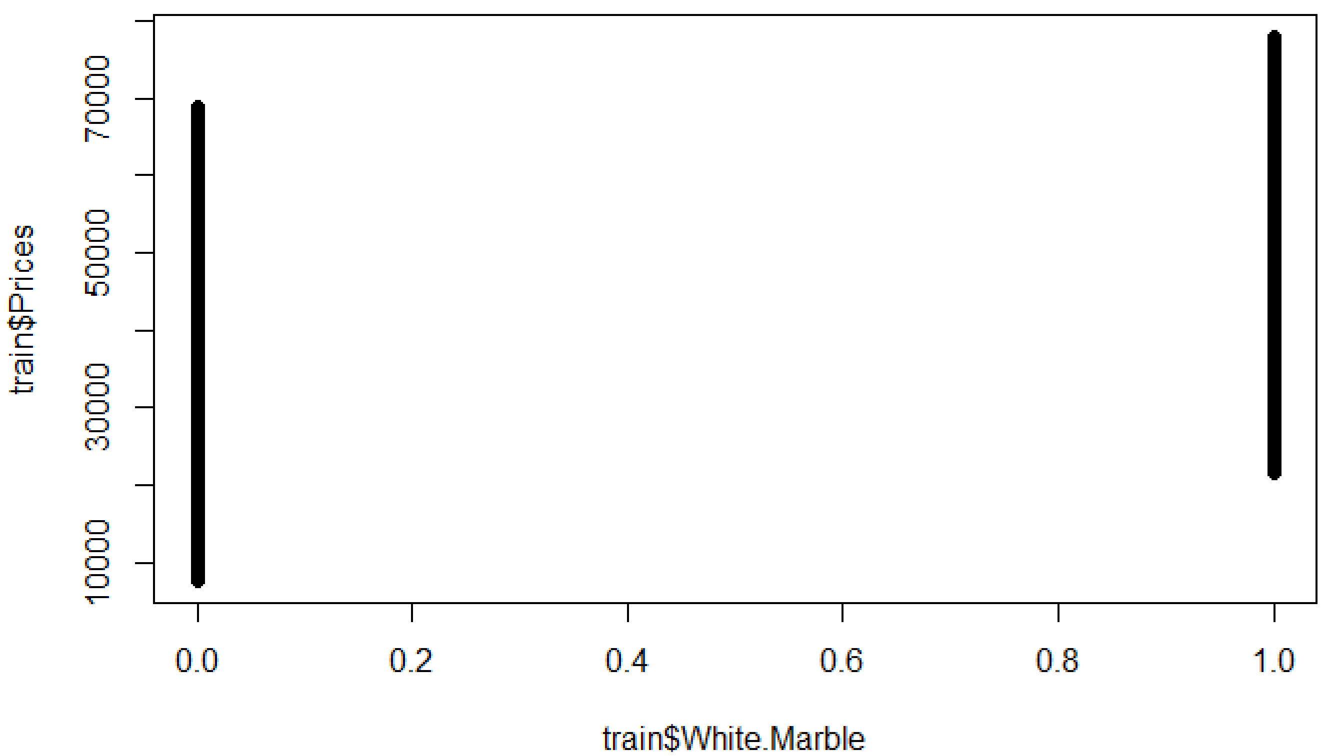
Hide

```
T<- table(train$City,(train$Prices>=35000))
plot(T)
```



Hide

```
plot(train$White.Marble,train$Prices)
```



Linear Regreasstion

Hide

```
lm1 <-lm(Prices~Area, data=train)
summary(lm1)
```

```
Call:
lm(formula = Prices ~ Area, data = train)

Residuals:
   Min     1Q Median     3Q    Max 
-31686  -8434   -229   8573  33023 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 38926.508     38.040 1023.30   <2e-16 ***
Area           25.042      0.264   94.86   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11980 on 399998 degrees of freedom
Multiple R-squared:  0.022, Adjusted R-squared:  0.022 
F-statistic:  8999 on 1 and 399998 DF,  p-value: < 2.2e-16
```

Looking at the krelation of prices upon the area is very week. Though we do have a good p-value. the R-square of .022 is a low value indicating it is not a good predictor, to really knoiw the price of this i think have a all atribute will give the

Hide

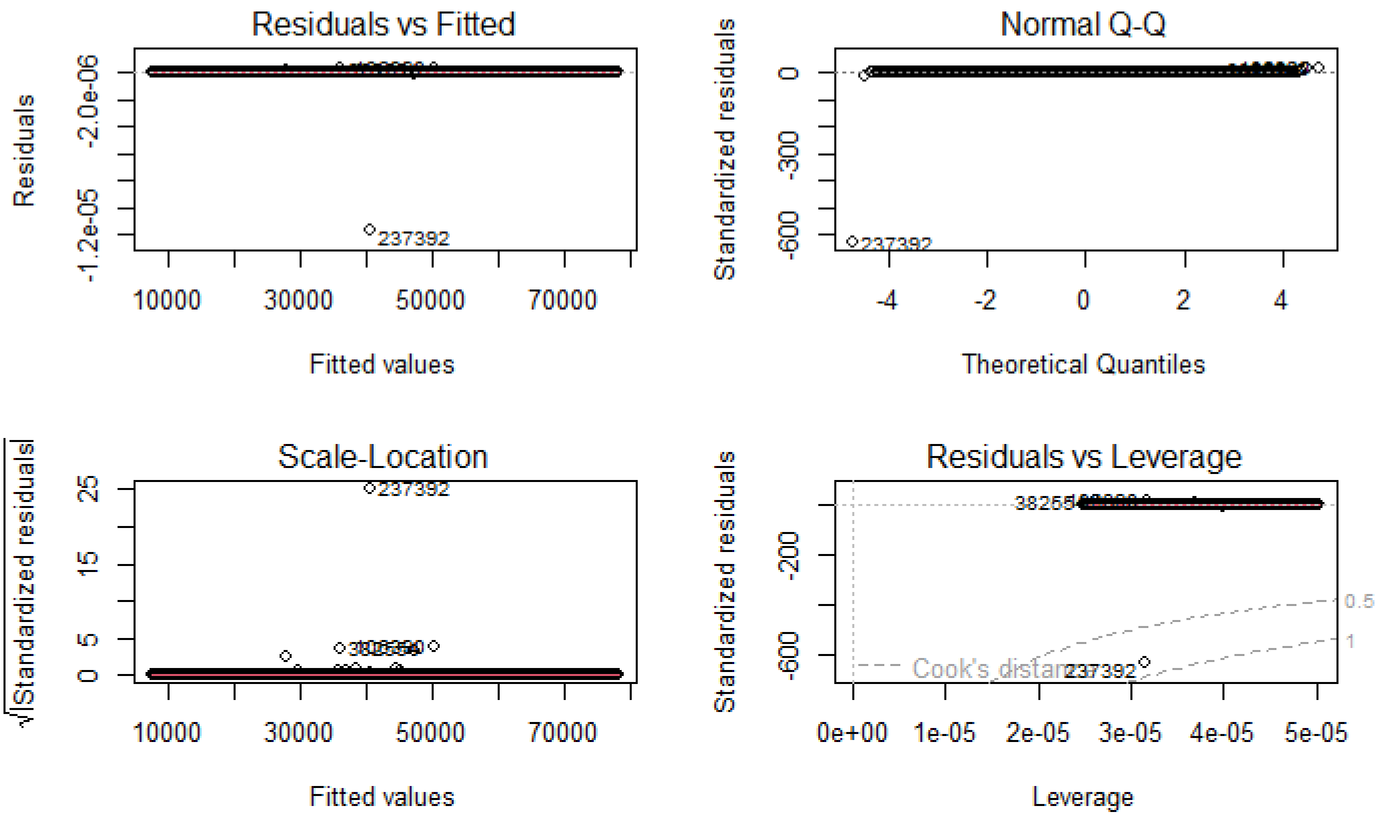
```
lm2 <-lm(Prices~. , data=train)
summary(lm2)
```

```
Call:
lm(formula = Prices ~ ., data = train)

Residuals:
       Min         1Q     Median         3Q        Max 
-1.166e-05  0.000e+00  0.000e+00  1.000e-10  2.712e-07 

Coefficients: (1 not defined because of singularities)
               Estimate Std. Error   t value Pr(>|t|)    
(Intercept)   1.000e+03  1.639e-10 6.102e+12   <2e-16 ***
Area          2.500e+01  4.064e-13 6.151e+13   <2e-16 ***
Garage        1.500e+03  3.570e-11 4.202e+13   <2e-16 ***
FirePlace     7.500e+02  2.062e-11 3.637e+13   <2e-16 ***
Baths         1.250e+03  2.063e-11 6.060e+13   <2e-16 ***
White.Marble  1.400e+04  7.137e-11 1.961e+14   <2e-16 ***
Black.Marble  5.000e+03  7.141e-11 7.002e+13   <2e-16 ***
Indian.Marble        NA         NA        NA       NA    
Floors        1.500e+04  5.832e-11 2.572e+14   <2e-16 ***
City          3.500e+03  3.571e-11 9.801e+13   <2e-16 ***
Solar         2.500e+02  5.832e-11 4.287e+12   <2e-16 ***
Electric      1.250e+03  5.832e-11 2.143e+13   <2e-16 ***
Fiber         1.175e+04  5.832e-11 2.015e+14   <2e-16 ***
Glass.Doors   4.450e+03  5.832e-11 7.631e+13   <2e-16 ***
Swiming.Pool  5.682e-11  5.832e-11 9.740e-01    0.330    
Garden        5.952e-11  5.832e-11 1.021e+00    0.307    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.844e-08 on 399985 degrees of freedom
Multiple R-squared:      1, Adjusted R-squared:      1 
F-statistic: 1.233e+28 on 14 and 399985 DF,  p-value: < 2.2e-16
```

it seems the accurate relation for price for all factor besides Indian Marble, this accuracy is seen from Multiple regression

Hide

```
#ploting residuals
par(mfrow=c(2,2))
plot(lm2)
```



Evaluate

The correlation is 1 which is really good and we missed by 3.31e-16

Hide

```
pred1 <- predict(lm2,newdata = test)
```

```
Warning: prediction from a rank-deficient fit may be misleading
```

Hide

```
cor_lm2 <-cor(pred1,test$Prices)
mme1 <- mean((pred1-test$Prices)^2)
print(paste("cor= ", cor_lm2))
```

```
[1] "cor=  1"
```

Hide

```
print(paste("mse = ", mme1))
```

```
[1] "mse =  3.41282775161943e-16"
```

KNN Regression

we get a cor of .11 and mse of 2149405815.5969

Hide

```
train_cut <- train[,c(1,3:16)]
test_cut <- test[,c(1,3:16)]
unique(train_cut)
```

	A… <int>	FirePlace <int>	Ba… <int>	White.Marble <int>	Black.Marble <int>	Indian.Marble <int>	Floors <int>	C… <int>	Solar <int>
237392	71	2	4	0	0	1	0	3	1
106390	160	0	4	1	0	0	1	2	0
304108	12	3	3	0	0	1	1	3	0
408457	90	3	1	1	0	0	0	2	0
295846	61	3	2	0	0	1	0	3	0
494468	40	4	1	1	0	0	1	3	0
126055	209	0	1	1	0	0	0	2	0
382554	126	2	4	1	0	0	0	2	1
345167	39	1	1	0	1	0	0	3	0
342900	169	2	2	0	0	1	1	3	0

1-10 of 396,318 rows | 1-10 of 15 columns Previous **1** 2 3 4 5 6 ... 100 Next

Hide

```
unique(test_cut)
```

	A… <int>	FirePlace <int>	Ba… <int>	White.Marble <int>	Black.Marble <int>	Indian.Marble <int>	Floors <int>	City <int>	Solar <int>
2	84	0	4	0	0	1	1	2	0
3	190	4	4	1	0	0	0	2	0
4	75	4	4	0	0	1	1	1	1
5	148	4	2	1	0	0	1	2	1
6	124	3	3	0	1	0	1	1	0
7	58	0	2	0	0	1	0	3	0
8	249	1	1	1	0	0	1	1	0
9	243	0	2	0	0	1	1	1	0

	A…	FirePlace	Ba…	White.Marble	Black.Marble	Indian.Marble	Floors	City	Solar
	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>
10	242	2	4	0	0	1	0	2	1
11	61	4	5	0	0	1	1	1	1

1-10 of 494,306 rows | 1-10 of 15 columns Previous **1** 2 3 4 5 6 ... 100 Next

Hide

```
train_cut <-train_cut[1:100,]
test_cut <- test_cut[1:100,]
fit <- knnreg(train_cut[,2:8],train_cut[,1], k=1)
predK <- predict(fit,test_cut[,2:8])
cor_knn1 <-cor(predK,test_cut$Prices)
mse_knn1<-mean((predK-test_cut$Prices)^2)
print(paste("cor=",cor_knn1))
```

```
[1] "cor= 0.115335585683436"
```

Hide

```
print(paste("mse=",mse_knn1))
```

```
[1] "mse= 2149405815.5969"
```

#scale the data In scale data the mse is still high cor is .79 the mse is 72856712.8454861

Hide

```
train_scaled <-train_cut[,2:8]
means <- sapply(train_scaled, mean)
stdvs <- sapply(train_scaled,sd)
train_scaled <-scale(train_scaled,center=means,scale=stdvs)
test_scaled <- scale(test_cut[,2:8],center=means,scale=stdvs)

fit<- knnreg(train_scaled,train_cut$Prices,k=3)
pred_scale <- predict(fit,test_scaled)
cor_knn2 <- cor(pred_scale,test_cut$Prices)
mse_knn2 <- mean((pred_scale-test_cut$Prices)^2)
print(paste("cor=",cor_knn2))
```

```
[1] "cor= 0.796380132352886"
```

Hide

```
print(paste("mse=",mse_knn2))
```

```
[1] "mse= 72856712.8454861"
```

#find the k

Hide

```
cor_k <- rep(0, 20)
mse_k <- rep(0, 20)
i <- 1
for (k in seq(1, 39, 2)){
  fit_k <- knnreg(train_scaled,train_cut$Prices, k=k)
  pred_k <- predict(fit_k, test_scaled)
  cor_k[i] <- cor(pred_k, test_cut$Prices)
  mse_k[i] <- mean((pred_k - test_cut$Prices)^2)
  print(paste("k=", k, cor_k[i], mse_k[i]))
  i <- i + 1
}
```

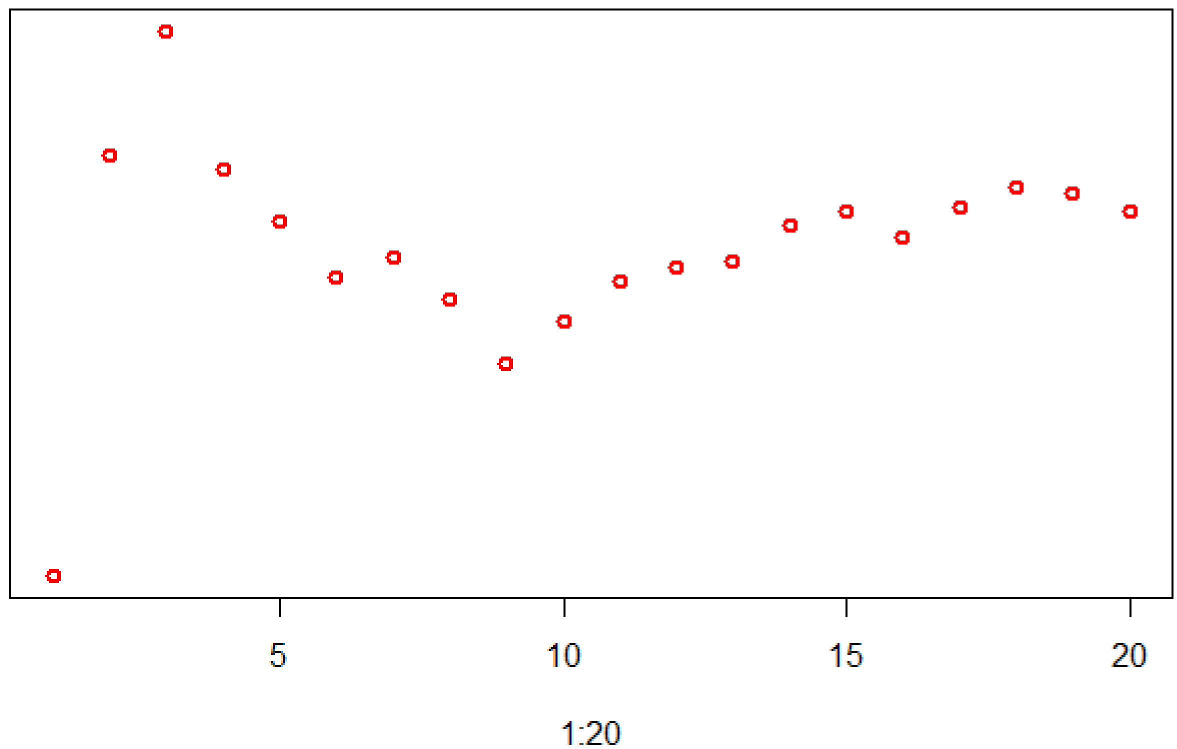
```
[1] "k= 1 0.697705954239255 105260306.076389"
[1] "k= 3 0.796380132352886 72856712.8454861"
[1] "k= 5 0.825611801068904 68438590.0875949"
[1] "k= 7 0.793195626189057 83681324.3540023"
[1] "k= 9 0.780760608314599 91721994.0509902"
[1] "k= 11 0.767439118457355 99191700.8833078"
[1] "k= 13 0.7723757870101 103344738.03055"
[1] "k= 15 0.762245592209111 105324225.523393"
[1] "k= 17 0.747529480607726 109520680.976867"
[1] "k= 19 0.757408849824059 111750797.639822"
[1] "k= 21 0.766495144969614 112892979.919811"
[1] "k= 23 0.770211001336784 116023353.710509"
[1] "k= 25 0.77141133039712 118568941.579382"
[1] "k= 27 0.78009061097317 122020651.182688"
[1] "k= 29 0.782940183693609 124655066.724075"
[1] "k= 31 0.777033251999397 128277486.785901"
[1] "k= 33 0.784321188173674 129876570.171806"
[1] "k= 35 0.788611399479734 131718241.995246"
[1] "k= 37 0.787219503360941 134713413.194961"
[1] "k= 39 0.783035048627945 137270504.669788"
```

Hide

```
plot(1:20, cor_k, lwd=2, col='red', ylab="", yaxt='n')
par(new=TRUE)
```

Hide

```
plot(1:20, mse_k, lwd=2, col='blue', labels=FALSE, ylab="", yaxt='n')
```



find the best k

Hide

```
which.min(mse_k)
```

```
[1] 3
```

Hide

```
which.max(cor_k)
```

```
[1] 3
```

let's compare with k being 20 a slight worst result then k =3 cor = .77 and mse = 111351666.0285

Hide

```
fit_20<- knnreg(train_scaled,train_cut$Prices,k=20)
pred_20<- predict(fit_20,test_scaled)
cor_k20 <- cor(pred_20,test_cut$Prices)
mse_k20 <- mean((pred_20-test_cut$Prices)^2)
print(paste("cor=",cor_k20))
```

```
[1] "cor= 0.765147038491144"
```

Hide

```
print(paste("mse=",mse_k20))
```

```
[1] "mse= 111351666.0285"
```

Using Tree

Hide

```
tree1<- tree(Prices~. , data=train )
summary(tree1)
```

```


Regression tree:
tree(formula = Prices ~ ., data = train)
Variables actually used in tree construction:
[1] "Floors"       "Fiber"        "White.Marble"
Number of terminal nodes:  8
Residual mean deviance:  26600000 = 1.064e+13 / 4e+05
Distribution of residuals:
      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-17550.000  -3594.000      5.653      0.000   3596.000  17500.000
```

Correlation is .9 rsme of 51514

Hide

```
pred<-predict(tree1,newdata = test)
corr_tree <- cor(pred,test$Prices)
print(paste("corr=",corr_tree ))
```

```
[1] "corr= 0.904880125112306"
```

Hide

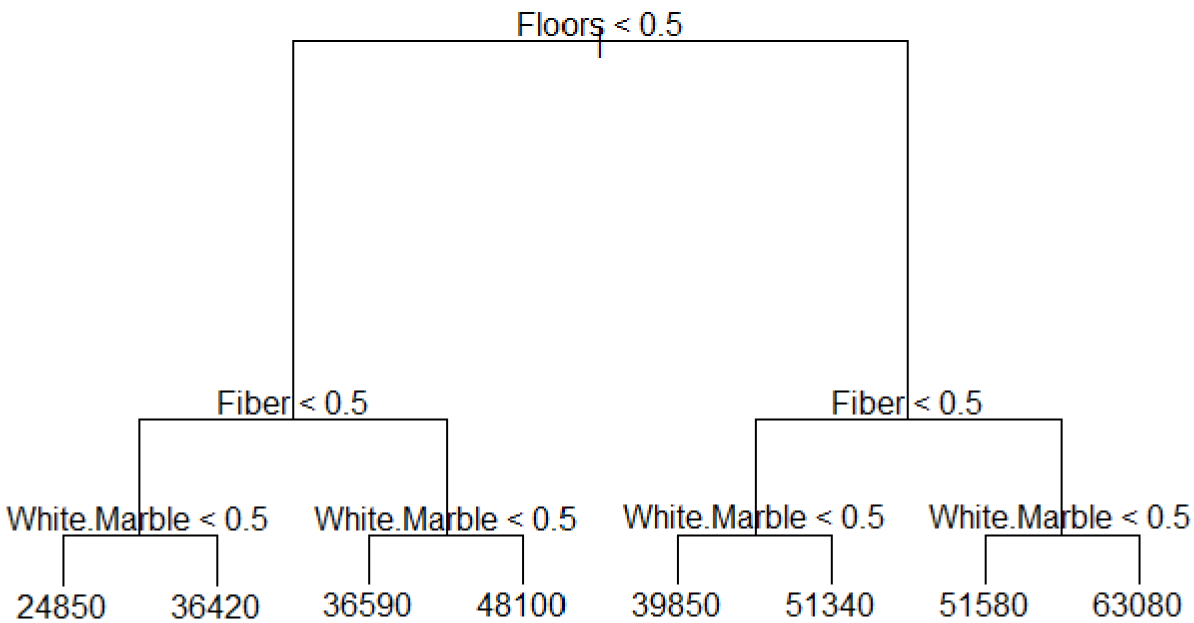
```
rsmeT <- sqrt(mean((pred-test$Prices)^2))
print(paste("RSME=", rsmeT))
```

```
[1] "RSME= 5154.9231931879"
```

The plot is quite neat than expected

Hide

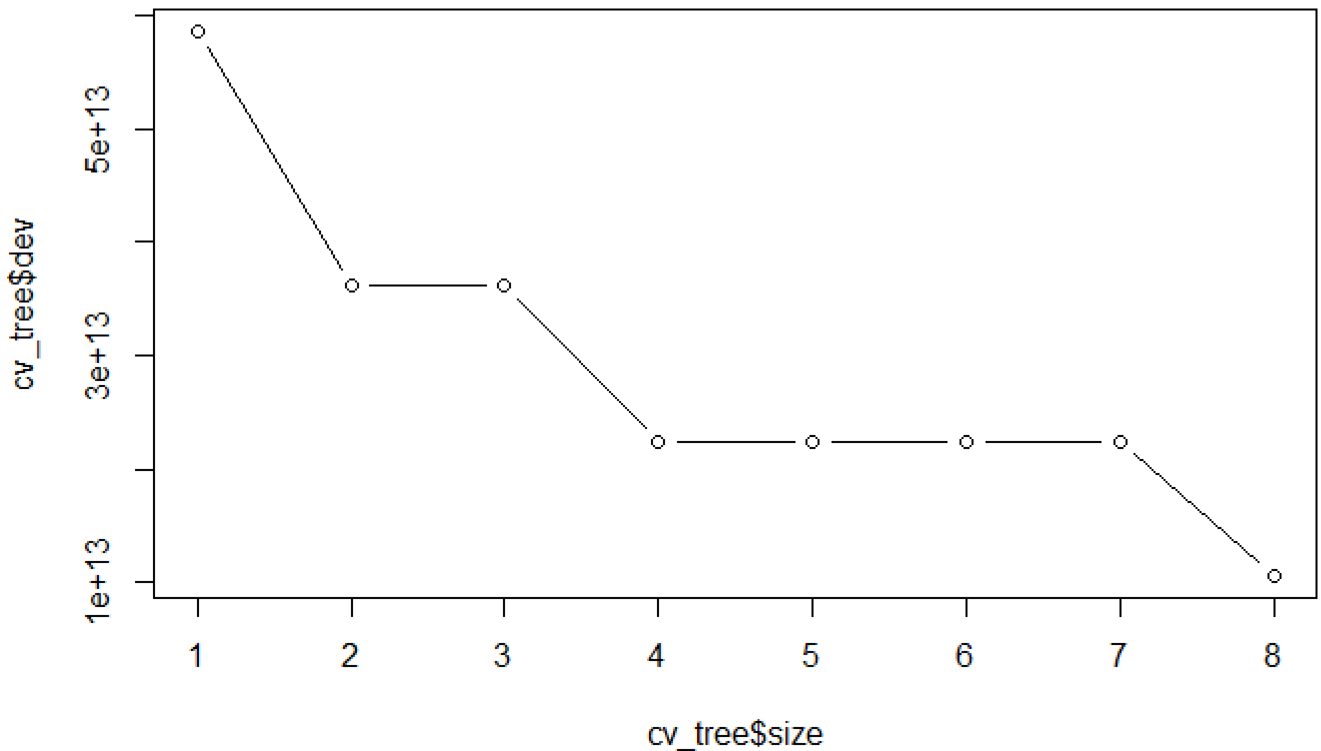
```
plot(tree1)
text(tree1,cex=1,pretty=0)
```



#cross validation The plot shows 8 terminals for the full tree.it seems there are two “dips” happening in the plot, I am taking the bend at 3 as I think that will give me the best tree and better understanding

Hide

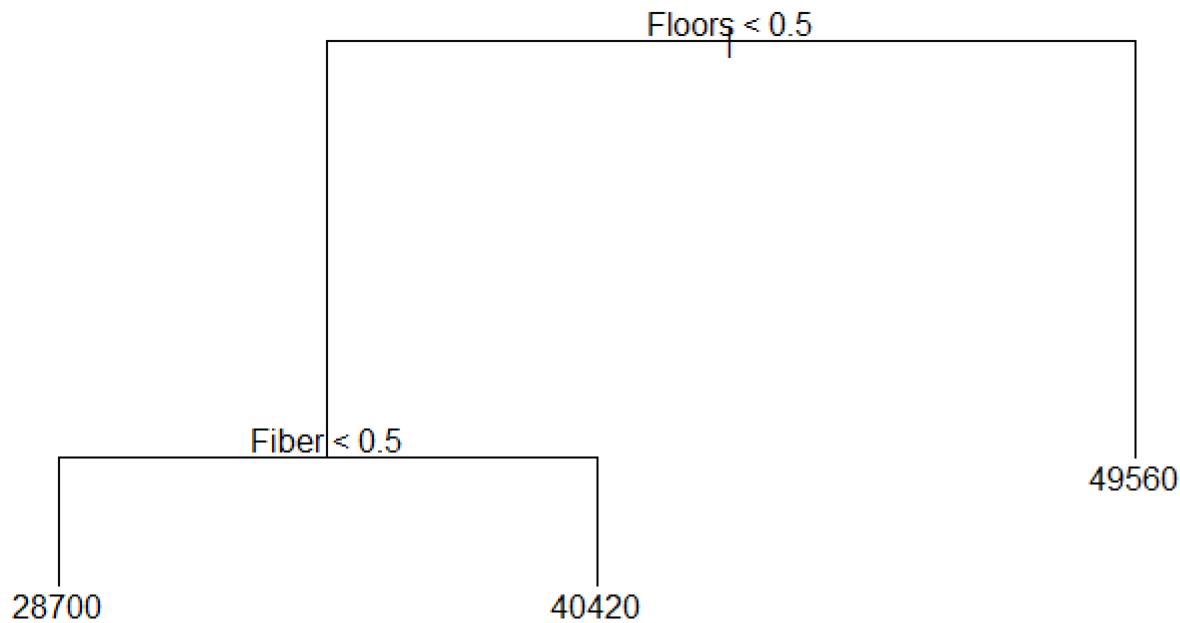
```
cv_tree <- cv.tree(tree1)
plot(cv_tree$size,cv_tree$dev, type='b')
```



prune the tree

Hide

```
tree_prune <- prune.tree(tree1,best=3)
plot(tree_prune)
text(tree_prune,pretty=0)
```



#test the pruned correlation is .71 and the rsme came out to be 8554.561

Hide

```
pred_prunned<-predict(tree_prune,newdata = test)
cor_prunned <- cor(pred_prunned,test$Prices)
rsme_prunned <- sqrt(mean((pred_prunned-test$Prices)^2))
print(paste("cor=",cor_prunned))
```

```
[1] "cor= 0.707821973000141"
```

Hide

```
print(paste("rmse=",rsme_prunned))
```

```
[1] "rmse= 8554.56157221625"
```

Conclusion

We see the best model to be Linear Regression, as we are getting the R-squared being 1 . The worst I believe to be the KNN as I was not able to run the model on the full data set, I had to reduce the records to get the proper model, even then, I got a high mse even when it was scaled. KNN is not good for a large dataset. Decision tree was quite decent, it used three predictor rather than using all of them, it used Fiber, Floors and White marbel for predictore with prices being the target. This means this were the deciding factor for prices at the house. The DEcision tree shows the important predictor for the target set, it gives a good decising facot such as for pricing in this dataset.