WESTERN SYDNEY
UNIVERSITY

Computing, Engineering & Mathematics

ASSIGNMENT / REPORT COVER SHEET

This sheet must be attached to all material being submitted for marking.

Student name: **Student number:**	Manan Bhatia 22035587
Sections completed individually	Section 1: Regression Section 2: Classification
Unit name & number:	COMP2025 Introduction to Data Science
Tutorial day and time:	Thursday 3-5pm
Title of Assignment:	Introduction to Data Science Assignment 1
Student Submitting the Assignment:	Manan Bhatia
Date submitted:	8/09/2022

Student Declaration (must be signed)

Declaration:

- ❑ I hold a copy of this assignment if the original is lost or damaged.
- ❑ I hereby certify that no part of this assignment or product has been copied from any other student's work or from any other source except where due acknowledgement is made in the assignment.
- ❑ No part of the assignment/product has been written / produced for me by any other person except where collaboration has been authorised by the subject lecturer/tutor concerned
- ❑ I am aware that this work may be reproduced and submitted to plagiarism detection software programs for the purpose of detecting possible plagiarism ***(which may retain a copy on its database for future plagiarism checking))***
- ❑ I hereby certify that no part of this assignment or product has been submitted by me in another (previous or current) assessment, except where appropriately referenced, and with prior permission from the Lecturer/Tutor/ Unit Co-ordinator for this unit.
- ❑

Student signature and date:	

Note: An examiner or lecturer/tutor has the right to not mark this assignment if the above declaration has not been signed.

IDS Assignment 1

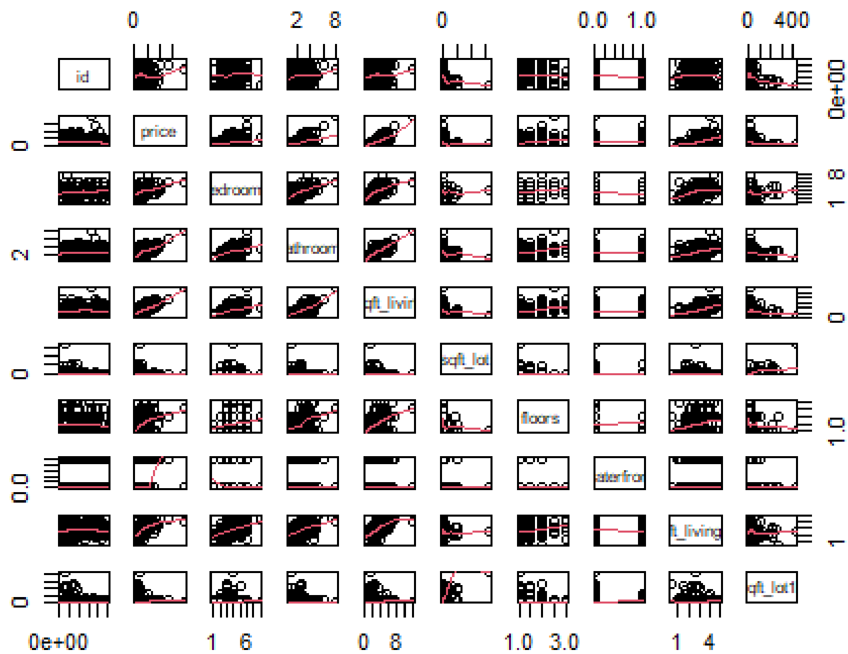
Manan Bhatia

2022-08-29

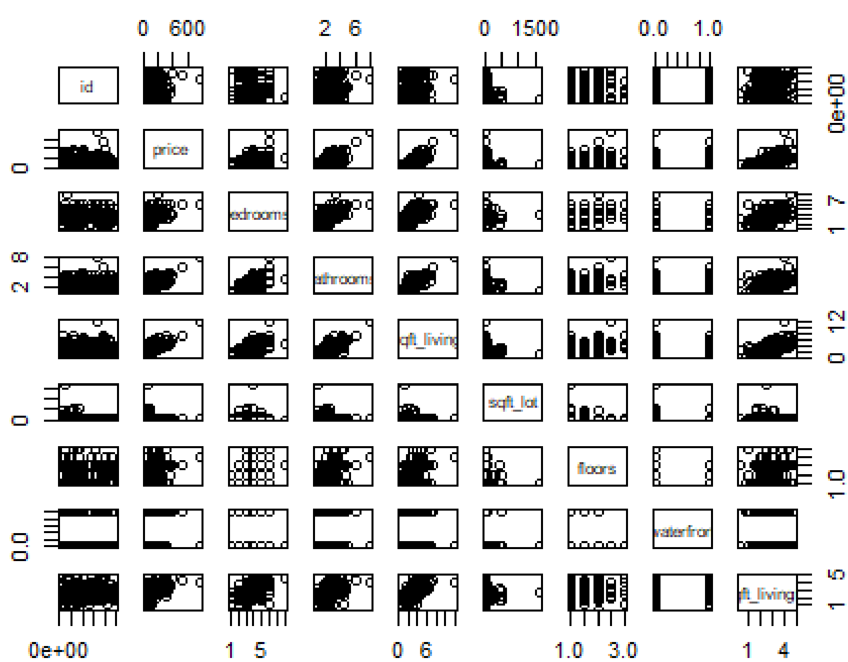
Question 1: Regression

1) Construct the matrix plot and correlation matrix (consider only relevant variables). Comment on the relationship among variables.

```
house=read.csv("kc_house.csv",header = TRUE)
attach(house)
dim(house)

## [1] 341  10

names(house)

##  [1] "id"            "price"         "bedrooms"      "bathrooms"
##  [5] "sqft_living"   "sqft_lot"      "floors"        "waterfront"
##  [9] "sqft_living15" "sqft_lot15"

View(house)
head(house)

##           id  price bedrooms bathrooms sqft_living sqft_lot floors waterfr
ont
## 1 7922800400  95.10        5      3.25        3.25   14.342      2
0
## 2 1516000055  65.00        3      2.25        2.15   21.235      1
0
## 3 2123039032  36.99        1      0.75        0.76   10.079      1
1
## 4 9297300045  55.00        3      2.00        1.97    4.166      2
0
## 5 1860600135 238.40        5      2.50        3.65    9.050      2
0
## 6 1560930070  84.00        4      3.50        2.84   40.139      1
0
##   sqft_living15 sqft_lot15
## 1          2.96     11.044
## 2          2.57     18.900
## 3          1.23     14.267
## 4          2.39      4.166
## 5          2.88      5.400
## 6          3.18     36.852

pairs(house,panel=panel.smooth)
```



```
pairs(house [,1:9])
```



The relationship among the variables like the Id, price, bedrooms, bathrooms, sqft_living, sqft_lot, floors, waterfront, sqft_living15 and sqft_lot 15. Between ‘id’ and ‘sqft_living, there is a weak linear relationship which goes in a horizontal straight line. Between the ’price’ and ‘bedrooms, there is a weak positive linear relationship. And between, ’bedrooms’ and ‘floors’, there is no significant relation ship. And there are many more relationships between two other variables as well.

```
cor(house)

##                         id       price     bedrooms    bathrooms sqft_livi
ng
## id             1.000000000 -0.02353996  0.108453574  0.096787228  0.042232
20
## price         -0.023539962  1.00000000  0.365855613  0.649982920  0.788079
30
## bedrooms       0.108453574  0.36585561  1.000000000  0.554497306  0.557906
87
## bathrooms      0.096787228  0.64998292  0.554497306  1.000000000  0.780949
05
## sqft_living    0.042232201  0.78807930  0.557906874  0.780949053  1.000000
00
## sqft_lot      -0.148251918 -0.08902246 -0.006739024 -0.121073782 -0.104880
69
## floors        -0.067458098  0.37379631  0.192062371  0.418907572  0.358113
98
## waterfront    -0.090588173  0.26183259 -0.219087311 -0.003185506 -0.013236
08
## sqft_living15 -0.002437858  0.60567854  0.392195987  0.494167386  0.649314
94
## sqft_lot15    -0.204135766 -0.14040054 -0.021601830 -0.127558035 -0.129234
90
##                   sqft_lot      floors   waterfront sqft_living15  sqft_lo
t15
## id            -0.148251918 -0.06745810 -0.090588173  -0.002437858 -0.20413
577
## price         -0.089022456  0.37379631  0.261832586   0.605678536 -0.14040
054
## bedrooms      -0.006739024  0.19206237 -0.219087311   0.392195987 -0.02160
183
## bathrooms     -0.121073782  0.41890757 -0.003185506   0.494167386 -0.12755
804
## sqft_living   -0.104880686  0.35811398 -0.013236081   0.649314935 -0.12923
490
## sqft_lot       1.000000000 -0.07646619 -0.012406719  -0.050714790  0.73550
275
## floors        -0.076466191  1.00000000  0.068261884   0.195168270 -0.06666
012
## waterfront    -0.012406719  0.06826188  1.000000000  -0.029921022  0.00202
911
## sqft_living15 -0.050714790  0.19516827 -0.029921022   1.000000000 -0.12127
202
```

```
## sqft_lot15     0.735502746 -0.06666012  0.002029110  -0.121272017  1.00000
000
```

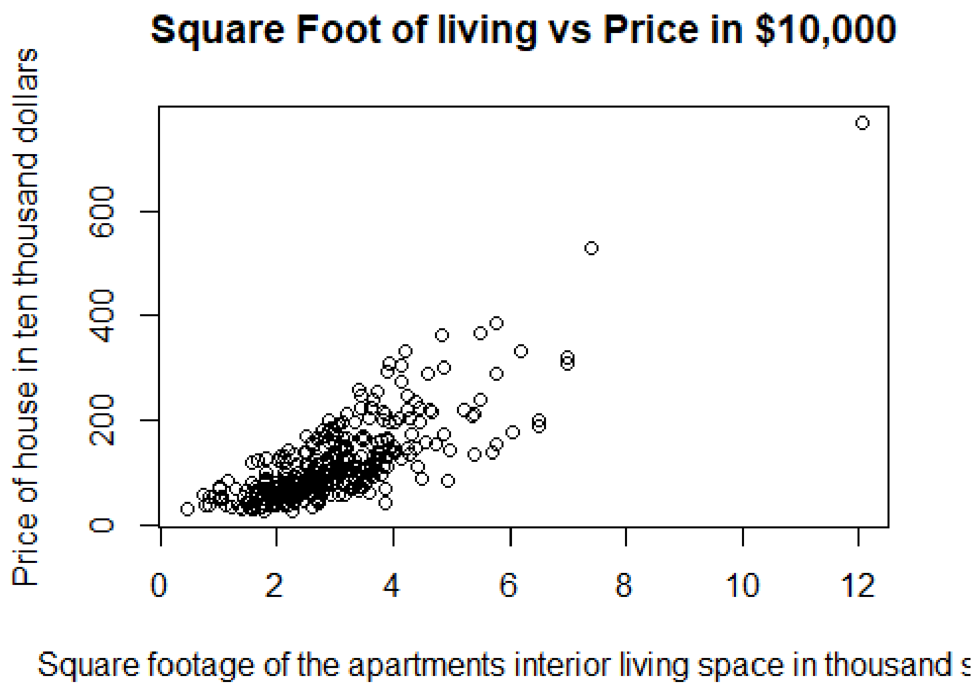
```
cov(house)
```

```
##                            id         price      bedroooms     bathrooms
## id            8.390431e+18 -5.456028e+09  3.280144e+08  2.601298e+08
## price        -5.456028e+09  6.402614e+03  3.056647e+01  4.825702e+01
## bedrooms      3.280144e+08  3.056647e+01  1.090219e+00  5.372003e-01
## bathrooms     2.601298e+08  4.825702e+01  5.372003e-01  8.609141e-01
## sqft_living   1.513977e+08  7.804266e+01  7.209449e-01  8.967804e-01
## sqft_lot     -4.610665e+10 -7.648026e+02 -7.554836e-01 -1.206149e+01
## floors       -1.003287e+08  1.535723e+01  1.029671e-01  1.995709e-01
## waterfront   -1.026070e+08  8.192483e+00 -8.945144e-02 -1.155770e-03
## sqft_living15 -5.124176e+06  3.516773e+01  2.971553e-01  3.327189e-01
## sqft_lot15   -2.493511e+10 -4.737481e+02 -9.511466e-01 -4.991002e+00
##                sqft_living      sqft_lot        floors    waterfront
## id            1.513977e+08 -4.610665e+10 -1.003287e+08 -1.026070e+08
## price         7.804266e+01 -7.648026e+02  1.535723e+01  8.192483e+00
## bedrooms      7.209449e-01 -7.554836e-01  1.029671e-01 -8.945144e-02
## bathrooms     8.967804e-01 -1.206149e+01  1.995709e-01 -1.155770e-03
## sqft_living   1.531676e+00 -1.393639e+01  2.275642e-01 -6.405546e-03
## sqft_lot     -1.393639e+01  1.152769e+04 -4.215409e+00 -5.208843e-01
## floors        2.275642e-01 -4.215409e+00  2.636321e-01  1.370536e-02
## waterfront   -6.405546e-03 -5.208843e-01  1.370536e-02  1.529067e-01
## sqft_living15  5.831260e-01 -3.951202e+00  7.271631e-02 -8.490107e-03
## sqft_lot15   -6.744711e+00  3.330086e+03 -1.443329e+00  3.345946e-02
##              sqft_living15    sqft_lot15
## id           -5.124176e+06 -2.493511e+10
## price         3.516773e+01 -4.737481e+02
## bedrooms      2.971553e-01 -9.511466e-01
## bathrooms     3.327189e-01 -4.991002e+00
## sqft_living   5.831260e-01 -6.744711e+00
## sqft_lot     -3.951202e+00  3.330086e+03
## floors        7.271631e-02 -1.443329e+00
## waterfront   -8.490107e-03  3.345946e-02
## sqft_living15  5.265590e-01 -3.710942e+00
## sqft_lot15   -3.710942e+00  1.778280e+03
```

2) Simple Linear Regresion

i) Fit a model to predict price in terms of sqft_living.

```
plot(price~sqft_living, xlab=" Square footage of the apartments interior livi
ng space in thousand sq.ft", ylab="Price of house in ten thousand dollars", m
ain="Square Foot of living vs Price in $10,000")
```



ii) Discuss the significance of the slope parameter estimate. (Write down the relevant hypothesis)

H0: β = 0 (There is NO linear relationship between X & Y) HA: β != 0 (There are some linear relationship between X & Y)

```
model = lm(price~sqft_living)
summary(model)

##
## Call:
## lm(formula = price ~ sqft_living)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -131.704  -29.885   -6.956   24.696  190.005
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -33.982      6.956  -4.885 1.59e-06 ***
## sqft_living   50.952      2.162  23.572  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 49.33 on 339 degrees of freedom
## Multiple R-squared:  0.6211, Adjusted R-squared:   0.62
## F-statistic: 555.6 on 1 and 339 DF,  p-value: < 2.2e-16
```

The p-value is 2.2e-16 which is less than 0.05, so there is a strong evidence to reject the null hypothesis at 5% level of significance and support the alternative hypothesis.

iii) Discuss the accuracy of the parameter estimates. (Standard errors/confidence intervals)

```
confint(model)

##                 2.5 %    97.5 %
## (Intercept) -47.66432 -20.30013
## sqft_living  46.70061  55.20429
```

The 95% Confidence intervals for the intercepts are between (-47.66432, -20.30013). Standard errors for the intercepts is 6.956 units.

The 95% Confidence intervals for the slope (Sqft_Living) are between (46.70061, 55.20429). Standard errors for the slope (Sqft_Living) is 2.162 units.

iv)Discuss the model accuracy. (R-squared, residual standard error etc.)

```
anova(model)

## Analysis of Variance Table
##
## Response: price
##              Df  Sum Sq Mean Sq F value    Pr(>F)
## sqft_living   1 1351998 1351998  555.62 < 2.2e-16 ***
## Residuals   339  824891    2433
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

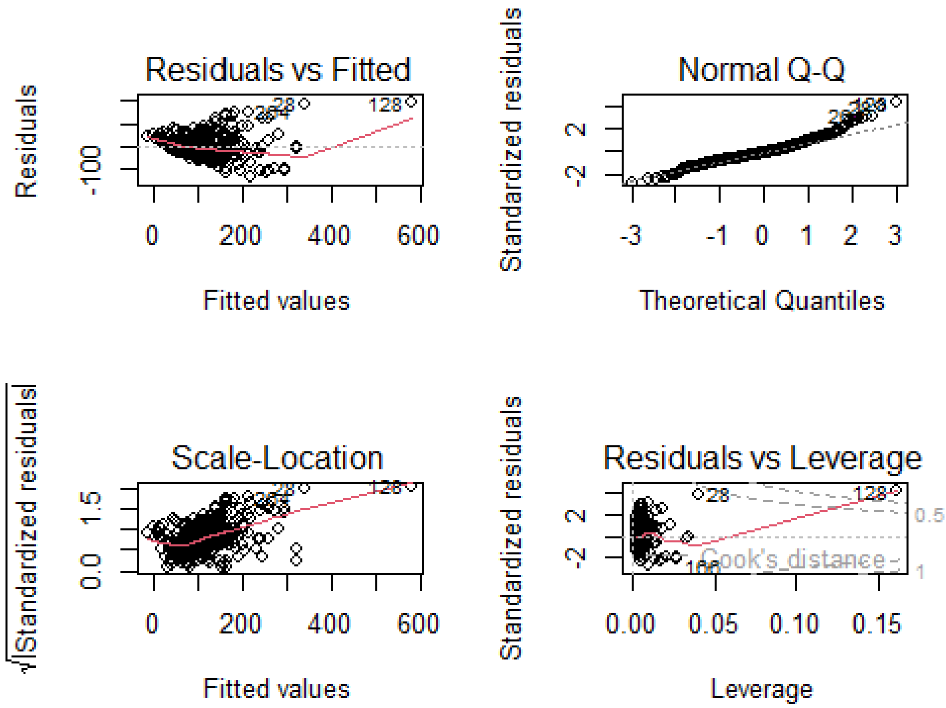
Sum Square of the slope is 1351998 and the residual of the sum square is 824891 R^2 = 1351998/1351998 + 824891 = 1351998/2176889 = 0.6211

So 62.11% variation in the price is explained by regression.

The estimated V(Y) is σ^2 = 2433 and so the σ is the square root of 2433 which is 49.33

v) Check for the model assumptions.

```
par(mfrow=c(2,2))
plot(model)
```



Graph 1: The plot seems to be scattered at first, so the constant variance assumption is met.

Graph 2: The plot seems to be in a straight line, so the normality assumption is met.

Graph 3: Same as Graph 1

Graph 4: There are number of influential observations such as 168, 28 and 128.

vi) Write down the model equation. The estimated linear model Y = α + βX

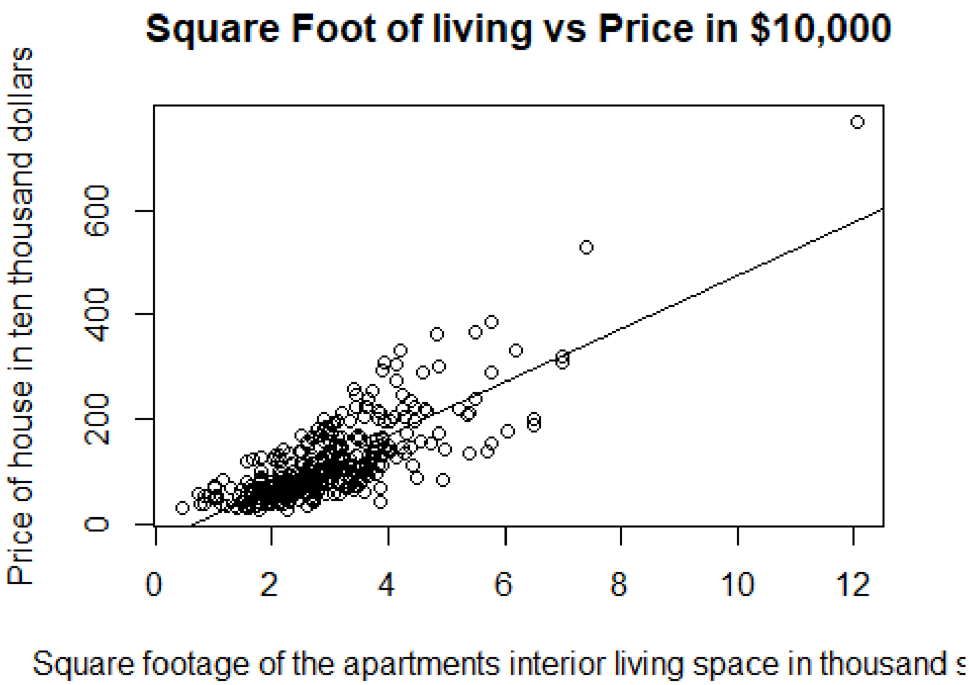
So,

```
lm(price~sqft_living)

##
## Call:
## lm(formula = price ~ sqft_living)
##
## Coefficients:
## (Intercept)  sqft_living
##      -33.98        50.95
```

Therefore, the equation is price = -33.98 + 50.95sqft_living Where α (intercept) is -33.98 and β (slope) is 60.95 and X is sqft_living and Y is price.

```
plot(price~sqft_living, xlab=" Square footage of the apartments interior livi
ng space in thousand sq.ft", ylab="Price of house in ten thousand dollars", m
```

```
ain="Square Foot of living vs Price in $10,000")
abline(a=-33.98, b=50.95)
```



vii) Predict the price of a house with 10, 000 sq.ft of the apartments interior living space (sqft_living).

```
predict(model, list(sqft_living = 10000))

##        1
## 509490.5
```

3) Multiple Linear Regression

i) Fit a model to predict price in terms of all the other quantitative predictors (numerical predictors)

```
model2=lm(price~sqft_living+sqft_lot+floors+bedrooms+bathrooms)
summary(model2)

##
## Call:
## lm(formula = price ~ sqft_living + sqft_lot + floors + bedrooms +
##     bathrooms)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -108.853  -30.652   -5.219   24.802  184.334
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) -33.809222  11.135016  -3.036  0.00258 **
## sqft_living  48.585607   3.498628  13.887  < 2e-16 ***
## sqft_lot      0.005419   0.024646   0.220  0.82612
## floors       13.887855   5.637019   2.464  0.01425 *
## bedrooms     -9.433705   3.119898  -3.024  0.00269 **
## bathrooms     8.186569   4.801476   1.705  0.08912 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 48.26 on 335 degrees of freedom
## Multiple R-squared:  0.6415, Adjusted R-squared:  0.6362
## F-statistic: 119.9 on 5 and 335 DF,  p-value: < 2.2e-16
```

ii)Remove the insignificant variables and fit a model including the rest of the variables.

```
model3=lm(price~floors+bedrooms+bathrooms)
summary(model3)

##
## Call:
## lm(formula = price ~ floors + bedrooms + bathrooms)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -129.57  -38.82   -6.25   29.35  351.61
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -43.662     13.817  -3.160  0.00172 **
## floors        19.301      7.036   2.743  0.00641 **
## bedrooms       1.153      3.776   0.305  0.76017
## bathrooms     50.859      4.592  11.076  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.41 on 337 degrees of freedom
## Multiple R-squared:  0.4351, Adjusted R-squared:  0.4301
## F-statistic: 86.53 on 3 and 337 DF,  p-value: < 2.2e-16
```

I removed square foot of interior living and square foot of land space.

iii) Add the interaction term bedrooms*floors to the model above.

```
multimodel=lm(price~floors+bedrooms+bathrooms+floors*bedrooms)
summary(multimodel)

##
## Call:
## lm(formula = price ~ floors + bedrooms + bathrooms + floors *
##     bedrooms)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -181.26  -35.94   -6.70   29.34  311.85
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       59.707     34.412   1.735  0.08364 .
## floors           -53.177     23.217  -2.290  0.02262 *
## bedrooms         -26.122      9.131  -2.861  0.00449 **
## bathrooms         50.572      4.528  11.169  < 2e-16 ***
## floors:bedrooms   18.903      5.779   3.271  0.00118 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.55 on 336 degrees of freedom
## Multiple R-squared:  0.4526, Adjusted R-squared:  0.446
## F-statistic: 69.44 on 4 and 336 DF,  p-value: < 2.2e-16
```

R^2 = 45.26% The interaction is slightly significant.

iv) Comment on the significance of the parameter estimates of the model above.

H0: βi = 0 (There is NO linear relationship between X & Y) HA: βi != 0 (There are some linear relationship between X & Y)

```
summary(multimodel)
```

```
##
## Call:
## lm(formula = price ~ floors + bedrooms + bathrooms + floors *
##     bedrooms)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -181.26  -35.94   -6.70   29.34  311.85
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       59.707     34.412   1.735  0.08364 .
## floors           -53.177     23.217  -2.290  0.02262 *
## bedrooms         -26.122      9.131  -2.861  0.00449 **
## bathrooms         50.572      4.528  11.169  < 2e-16 ***
## floors:bedrooms   18.903      5.779   3.271  0.00118 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.55 on 336 degrees of freedom
## Multiple R-squared:  0.4526, Adjusted R-squared:  0.446
## F-statistic: 69.44 on 4 and 336 DF,  p-value: < 2.2e-16
```

Since the p-value is 2e-16, which less than 0.05, so there is enough evidence to reject the null hypothesis at 5% level.

So the parameter is slightly significant

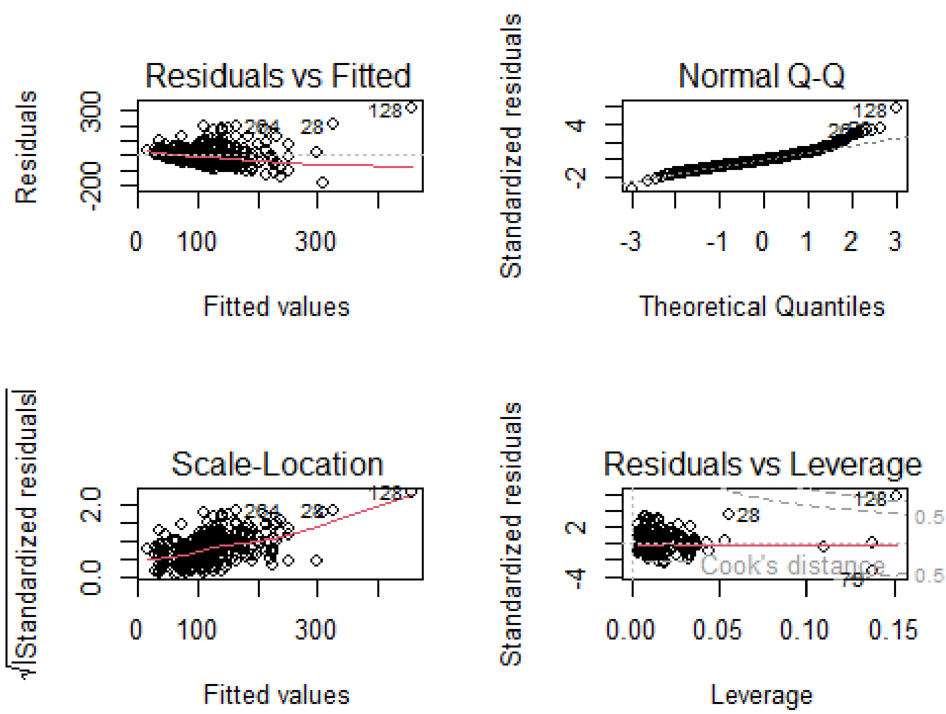
And: α (intercept) = 59.707 β (slope1) = -53.177 (slope2) = -26.122 (slope3) = 50.572 (slope4) = 18.903

The equation is:

price = 59.707 - 53.177floors - 26.122bedrooms + 50.572bathrooms + 18.903floors*bedrooms

v) Check for the model assumptions.

```
par(mfrow=c(2,2))
plot(multimodel)
```



Graph 1: This plot shows a negative linear regression which shows pattern in the resdiual impling that the pattern in the dataset is captured by the model.

Graph 2: This plot shows whether or not the standardized residuals follow a normal distribution. It can be seen clealy from the graph that the standardized residuals deviate from the normal distribution since the data points do not lie on the straight line.

Graph 3: This plot shows that the residual variance is constant.

Graph 4: This plot depicts some outliers.

vi) Compare and comment on the accuracy of the models in part ii and part iii. Suggest the best model.

Part ii accuracy:

```
anova(model3)

## Analysis of Variance Table
##
## Response: price
##            Df  Sum Sq Mean Sq F value    Pr(>F)
## floors      1  304163  304163  83.359 < 2.2e-16 ***
## bedrooms    1  195453  195453  53.566 1.848e-12 ***
## bathrooms   1  447619  447619 122.675 < 2.2e-16 ***
## Residuals 337 1229654    3649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Part iii accuracy:

```
anova(multimodel)

## Analysis of Variance Table
##
## Response: price
##                  Df  Sum Sq Mean Sq F value    Pr(>F)
## floors            1  304163  304163  85.759 < 2.2e-16 ***
## bedrooms          1  195453  195453  55.108 9.455e-13 ***
## bathrooms         1  447619  447619 126.206 < 2.2e-16 ***
## floors:bedrooms   1   37951   37951  10.700  0.001182 **
## Residuals       336 1191703    3547
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The sum square of all slopes for part ii are 304163, 195453 and 447619 respectively and its residuals is 1229654. Whereas, the sum square of all slopes for part iii are 304163, 195453, 447619 and 37951 respectively and its residuals is 1191703.

So the best accuracy of the model is part iii because, it has an extra interaction and that there is a perefect linear regression.

vii) Fit a polynomial regression model to predict price using sqft_living of order 2 and test the model significance.

```
polymodel = lm(price~sqft_living+I(sqft_living*sqft_living))
summary(polymodel)

##
## Call:
## lm(formula = price ~ sqft_living + I(sqft_living * sqft_living))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -125.447  -27.185   -7.469   21.488  162.383
##
```

```
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   8.4278    10.9442   0.770    0.442
## sqft_living                  26.0342     5.4852   4.746 3.06e-06 ***
## I(sqft_living * sqft_living)   3.0542     0.6215   4.914 1.39e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 47.73 on 338 degrees of freedom
## Multiple R-squared:  0.6463, Adjusted R-squared:  0.6442
## F-statistic: 308.9 on 2 and 338 DF,  p-value: < 2.2e-16
```

Using the hypothesis test used in part iv, it is shown that the variable with order 2 are not significant. Hence, they can be removed from the model. This nonlinear regression model is not adequate.

Question 2: Classification

1) Logistic Regression

i) Construct Logistic regression model for “price_cat” in terms of all the other variables (Use training dataset).

Starting from here, I set up a seed to 100 and the set both the train set and test set.

```
price_cat = ifelse(price > median(price), "High","Low")
str(price_cat)

##  chr [1:341] "Low" "Low" "Low" "Low" "High" "Low" "Low" "High" "High" "Low
" ...
```

Here, I added a new variable called price_cat and assigned many values as Low or High.

```
price_cat=as.factor(price_cat)
str(price_cat)

##  Factor w/ 2 levels "High","Low": 2 2 2 2 1 2 2 1 1 2 ...
```

I added the new variable in the house data.

```
Newhouse=data.frame(house,price_cat)
head(Newhouse)

##           id  price bedrooms bathrooms sqft_living sqft_lot floors waterfr
ont
## 1 7922800400  95.10        5      3.25        3.25   14.342      2
0
## 2 1516000055  65.00        3      2.25        2.15   21.235      1
0
## 3 2123039032  36.99        1      0.75        0.76   10.079      1
1
## 4 9297300045  55.00        3      2.00        1.97    4.166      2
0
```

```
## 5 1860600135 238.40        5      2.50        3.65    9.050      2
0
## 6 1560930070  84.00        4      3.50        2.84   40.139      1
0
##   sqft_living15 sqft_lot15 price_cat
## 1          2.96     11.044       Low
## 2          2.57     18.900       Low
## 3          1.23     14.267       Low
## 4          2.39      4.166       Low
## 5          2.88      5.400      High
## 6          3.18     36.852       Low

newhouse=Newhouse[,-2]
names(newhouse)

##  [1] "id"            "bedrooms"      "bathrooms"     "sqft_living"
##  [5] "sqft_lot"      "floors"        "waterfront"    "sqft_living15"
##  [9] "sqft_lot15"    "price_cat"

dim(newhouse)

## [1] 341  10

set.seed(100)
tr = sample(1:nrow(newhouse), nrow(newhouse)*0.75) #to divide the dataset int
o training and test set (50/50)\
price_train= price_cat[tr ]
price_test=price_cat[-tr]
train = newhouse[tr, ] #defining training dataset
dim(train)

## [1] 255  10

test = newhouse[-tr, ] #defining testing dataset
dim(test)

## [1] 86 10
```

Then, I removed the original price variable from this dataset.

I tried finding the logistic regression by doing:

```
model4<-glm(price_cat~.,data=newhouse,family = binomial)

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

summary(model4)

##
## Call:
## glm(formula = price_cat ~ ., family = binomial, data = newhouse)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -2.3919  -0.4941   0.0000   0.4837   2.8958
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.043e+01  1.371e+00   7.605 2.84e-14 ***
## id             4.819e-11  5.601e-11   0.860   0.3895
## bedrooms       8.116e-02  2.071e-01   0.392   0.6951
## bathrooms     -4.208e-01  3.083e-01  -1.365   0.1723
## sqft_living   -1.461e+00  3.391e-01  -4.308 1.65e-05 ***
## sqft_lot      -1.968e-03  4.027e-03  -0.489   0.6251
## floors        -6.586e-01  3.643e-01  -1.808   0.0706 .
## waterfront    -2.592e+00  5.583e-01  -4.642 3.45e-06 ***
## sqft_living15 -2.025e+00  3.932e-01  -5.151 2.59e-07 ***
## sqft_lot15     8.167e-02  2.018e-02   4.048 5.17e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 472.72  on 340  degrees of freedom
## Residual deviance: 234.71  on 331  degrees of freedom
## AIC: 254.71
##
## Number of Fisher Scoring iterations: 7
```

```
model4.5<-glm(price_cat~sqft_living+sqft_living15+waterfront+sqft_lot15,data=
train,family = binomial)
summary(model4.5)
```

```
##
## Call:
## glm(formula = price_cat ~ sqft_living + sqft_living15 + waterfront +
##     sqft_lot15, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2126  -0.5482  -0.0191   0.4978   3.1704
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   10.08524    1.31018   7.698 1.39e-14 ***
## sqft_living   -1.57087    0.32586  -4.821 1.43e-06 ***
## sqft_living15 -2.32260    0.46076  -5.041 4.64e-07 ***
## waterfront    -3.22983    0.66654  -4.846 1.26e-06 ***
## sqft_lot15     0.08122    0.02227   3.648 0.000264 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 353.19  on 254  degrees of freedom
## Residual deviance: 177.25  on 250  degrees of freedom
## AIC: 187.25
##
## Number of Fisher Scoring iterations: 7
```

```
model5<-glm(price_cat~sqft_living+sqft_living15+waterfront+sqft_lot15,data=test,family = binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(model5)
```

```
##
## Call:
## glm(formula = price_cat ~ sqft_living + sqft_living15 + waterfront +
##     sqft_lot15, family = binomial, data = test)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.32563  -0.49560   0.03046   0.54875   1.90329
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    8.38399    2.03198   4.126 3.69e-05 ***
## sqft_living   -2.26706    0.60547  -3.744 0.000181 ***
## sqft_living15 -1.00303    0.66883  -1.500 0.133698
## waterfront    -1.30371    0.89300  -1.460 0.144312
## sqft_lot15     0.07962    0.04615   1.725 0.084500 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 118.056  on 85  degrees of freedom
## Residual deviance:  59.447  on 81  degrees of freedom
## AIC: 69.447
##
## Number of Fisher Scoring iterations: 8
```

ii) Comment on the significance of the parameter estimates.

For Model4.5:

H0: $\beta = 0$ HA: $\beta != 0$

Since the p-values of sqft_living, waterfront, sqft_living15 and sqft_lot15 are less than 0.05, there is enough evidence to reject the null hypothesis at 5% level of significance.

$P(X) = e^\alpha+\beta X/ 1+e^\alpha+\beta X$

α = -10.43 β = -48.19 + -81.16 + 42.08 + 14.61 + 19.68 +65.86 + 25.92 + 20.25 + -81.67/9 = -2.51

So P(X) = e^-10.43 -2.51x/1+e^-10.43-2.51x

This shows that the estimated parameters is just 0 when x= 87. The numbers decreasing if x is less than 87.

iii) Improve the model based on the output in part i. (Hint: Consider the significance of the parameter estimates).

By improving the model, is to remove the unwanted variables which have the p-value above 0.05. Between model4 and model4.5, model4.5 set is a better data than model 4 is because the null deviance is 353.19 compared to the bigger 472.72 in model4. The deviance residual for model4.5 is 177.25 and model4 is 234.71. The AIC for model4.5 is 187.25 whereas for model4, it is 254.71. And the Fisher Scoring for both models is 7. Overall, the best model is model4.5 because the AIC, the null and residual deviances have a lower numbers than model4 which makes the logistic regression better and more improved. Between Model4 and Model4.5, the variables I removed were id, bedrooms

iv) Predict the outputs for the test dataset using the model in part iii and construct misclassification table.

The prediction probability and class of the outputs for the newhouse data set using model4.5:

```
pred_prob <- predict(model4.5, type="response")
head(pred_prob)

##        202        112        206          4        311        326
## 1.00000000 0.03851940 0.25981732 0.85545129 0.84706538 0.02390202
```

Train DATASET:

```
pred_class <- rep("Low", 341)
pred_class[pred_prob>0.5]="High"
head(pred_class)

## [1] "High" "Low"  "Low"  "High" "High" "Low"
```

Missclassification Table or Missclassification Matrix:

```
table(pred_class,price_cat)

##           price_cat
## pred_class High Low
##       High   79  79
##       Low    91  92
```

v) Calculate the misclassification rate. (Use test dataset).

```
Misclassification <- (79+92)/(79+79+91+92)
Misclassification
```

```
## [1] 0.5014663
```

The missclassification rate is 50.01% (train dataset)

```
FPM <- (79)/(91+79)
FPM

## [1] 0.4647059
```

The false positive rate is 46.47% (train dataset)

```
FNM <- (92)/(92+79)
FNM

## [1] 0.5380117
```

The false negative rate is 53.80% (train dataset)

Using the test DATASET:

```
pred_prob2 <- predict(model5, type="response")
head(pred_prob2)

##          6          9         10         21         22         23
## 0.84424844 0.01757486 0.85941545 0.48722575 0.51202278 0.95117506

pred_class2 <- rep("Low", 341)
pred_class2[pred_prob2>0.5]="High"
head(pred_class2)

## [1] "High" "Low"  "High" "Low"  "High" "High"
```

Missclassification table:

```
table(pred_class2,price_cat)

##            price_cat
## pred_class2 High Low
##        High  101  86
##        Low    69  85

Misclassification2 <- (101+85)/(69+85+101+86)
Misclassification2

## [1] 0.5454545
```

The Missclassification rate is 54.55% (test dataset)

```
FPM2 <- (101)/(69+101)
FPM2

## [1] 0.5941176
```

The false positive rate is 59.41% (test dataset)

```r
FNM2 <- (85)/(85+86)
FNM2

## [1] 0.497076
```

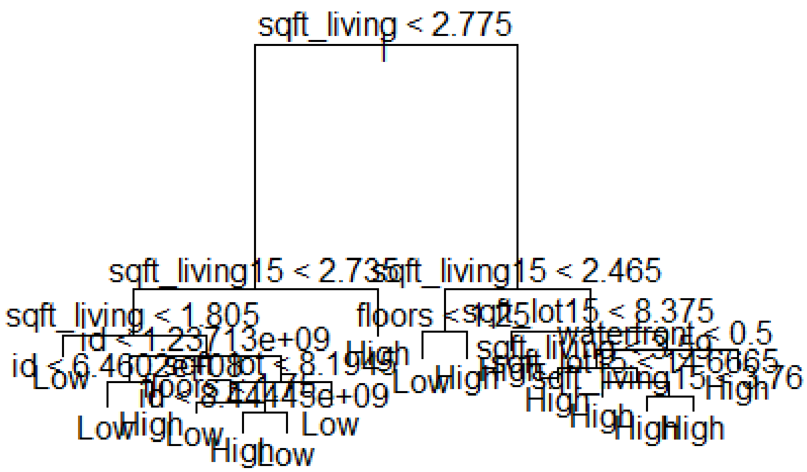
The false negative rate is 49.71% (test dataset)

2) Decision Tree
i) Build a classification tree model for the training dataset

```r
library(ISLR)
library(tree)

house_tree <- tree(price_cat~., train)
house_tree

## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##   1) root 255 353.200 High ( 0.51765 0.48235 )
##     2) sqft_living < 2.775 117 110.100 Low ( 0.17949 0.82051 )
##       4) sqft_living15 < 2.735 104  74.390 Low ( 0.11538 0.88462 )
##         8) sqft_living < 1.805 32   0.000 Low ( 0.00000 1.00000 ) *
##         9) sqft_living > 1.805 72  64.880 Low ( 0.16667 0.83333 )
##          18) id < 1.23713e+09 14  19.410 Low ( 0.50000 0.50000 )
##            36) id < 6.4602e+08 9   9.535 Low ( 0.22222 0.77778 ) *
##            37) id > 6.4602e+08 5   0.000 High ( 1.00000 0.00000 ) *
##          19) id > 1.23713e+09 58  34.070 Low ( 0.08621 0.91379 )
##            38) sqft_lot < 8.1945 26  25.460 Low ( 0.19231 0.80769 )
##              76) floors < 1.75 16   7.481 Low ( 0.06250 0.93750 ) *
##              77) floors > 1.75 10  13.460 Low ( 0.40000 0.60000 )
##               154) id < 8.44445e+09 5   5.004 High ( 0.80000 0.20000 ) *
##               155) id > 8.44445e+09 5   0.000 Low ( 0.00000 1.00000 ) *
##            39) sqft_lot > 8.1945 32   0.000 Low ( 0.00000 1.00000 ) *
##       5) sqft_living15 > 2.735 13  16.050 High ( 0.69231 0.30769 ) *
##     3) sqft_living > 2.775 138 136.400 High ( 0.80435 0.19565 )
##       6) sqft_living15 < 2.465 21  28.680 Low ( 0.42857 0.57143 )
##        12) floors < 1.25 11   6.702 Low ( 0.09091 0.90909 ) *
##        13) floors > 1.25 10  10.010 High ( 0.80000 0.20000 ) *
##       7) sqft_living15 > 2.465 117  89.610 High ( 0.87179 0.12821 )
##        14) sqft_lot15 < 8.375 27   0.000 High ( 1.00000 0.00000 ) *
##        15) sqft_lot15 > 8.375 90  81.100 High ( 0.83333 0.16667 )
##          30) waterfront < 0.5 72  73.690 High ( 0.79167 0.20833 )
##            60) sqft_living < 3.59 33  42.010 High ( 0.66667 0.33333 ) *
##            61) sqft_living > 3.59 39  25.790 High ( 0.89744 0.10256 )
##             122) sqft_lot15 < 14.6665 21   0.000 High ( 1.00000 0.00000 )
*
##             123) sqft_lot15 > 14.6665 18  19.070 High ( 0.77778 0.22222 )
##               246) sqft_living15 < 3.76 10  13.460 High ( 0.60000 0.40000
) *
##               247) sqft_living15 > 3.76 8   0.000 High ( 1.00000 0.00000 )
```

```
*
##           31) waterfront > 0.5 18   0.000 High ( 1.00000 0.00000 ) *
```

```
plot(house_tree)
text(house_tree, pretty = 0)
```



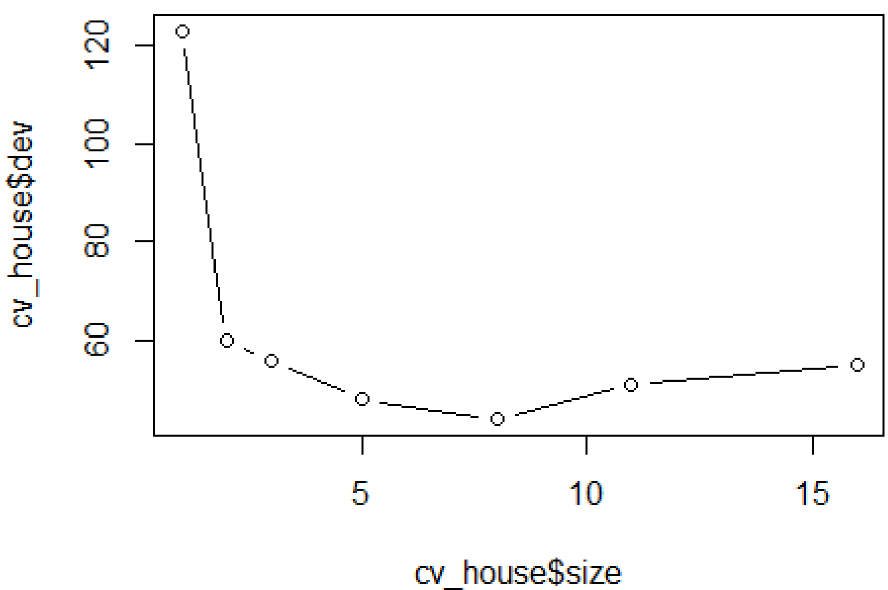

```
summary(house_tree)

##
## Classification tree:
## tree(formula = price_cat ~ ., data = train)
## Variables actually used in tree construction:
## [1] "sqft_living"   "sqft_living15" "id"            "sqft_lot"
## [5] "floors"        "sqft_lot15"    "waterfront"
## Number of terminal nodes:  16
## Residual mean deviance:  0.4613 = 110.2 / 239
## Misclassification error rate: 0.102 = 26 / 255
```

ii) Use cross-validation and choose the best size for the tree in part i.

```
set.seed(3)
cv_house = cv.tree(house_tree, FUN=prune.misclass)
cv_house

## $size
## [1] 16 11  8  5  3  2  1
##
## $dev
```

```
## [1]  55  51  44  48  56  60 123
##
## $k
## [1]      -Inf  0.000000  1.000000  1.666667  4.500000  5.000000 75.000000
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
plot(cv_house$size, cv_house$dev, type="b")
```



Pruning will not improve the model. The best size is 8 which is the current size of the tree fitted for the training dataset.

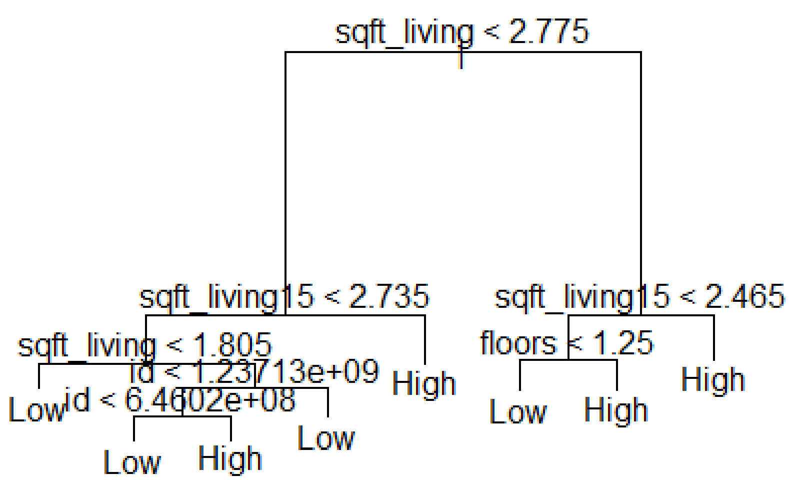
iii) Build the tree with the best size (pruning) obtained in part ii.

```
prune <-prune.misclass(house_tree, best=8)
prune
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 255 353.200 High ( 0.51765 0.48235 )
##    2) sqft_living < 2.775 117 110.100 Low ( 0.17949 0.82051 )
##      4) sqft_living15 < 2.735 104  74.390 Low ( 0.11538 0.88462 )
##        8) sqft_living < 1.805 32   0.000 Low ( 0.00000 1.00000 ) *
```

```
##        9) sqft_living > 1.805 72  64.880 Low ( 0.16667 0.83333 )
##         18) id < 1.23713e+09 14  19.410 Low ( 0.50000 0.50000 )
##           36) id < 6.4602e+08 9   9.535 Low ( 0.22222 0.77778 ) *
##           37) id > 6.4602e+08 5   0.000 High ( 1.00000 0.00000 ) *
##         19) id > 1.23713e+09 58  34.070 Low ( 0.08621 0.91379 ) *
##      5) sqft_living15 > 2.735 13  16.050 High ( 0.69231 0.30769 ) *
##    3) sqft_living > 2.775 138 136.400 High ( 0.80435 0.19565 )
##      6) sqft_living15 < 2.465 21  28.680 Low ( 0.42857 0.57143 )
##       12) floors < 1.25 11   6.702 Low ( 0.09091 0.90909 ) *
##       13) floors > 1.25 10  10.010 High ( 0.80000 0.20000 ) *
##      7) sqft_living15 > 2.465 117  89.610 High ( 0.87179 0.12821 ) *
```

```
plot(prune)
text(prune)
```



iv) Predict the outputs for the test dataset using the model in part iii and construct misclassification table.

```
tree_pred = predict(prune, test, type='class')
length(tree_pred)

## [1] 86

length(price_test)

## [1] 86

length(price_train)
```

```
## [1] 255
```

```
tree_pred
```

```
##  [1] High High Low  High High Low  High High High Low  Low  High High High
Low
## [16] Low  High Low  High Low  High Low  High High High High Low  Low  Low
Low
## [31] High High High High High Low  High High High High Low  Low  High High
Low
## [46] Low  Low  High High High Low  High High Low  Low  Low  Low  Low  Low
Low
## [61] High High High Low  High High High High High Low  High Low  High Low
High
## [76] High Low  Low  Low  Low  Low  Low  High High High High
## Levels: High Low
```

```
table2 = table(tree_pred, price_test)
table2
```

```
##          price_test
## tree_pred High Low
##      High   34  15
##      Low     4  33
```

` v) Calculate the misclassification rate. (Use test dataset).

TEST SET:

```
Misclassification3 <- (table2[1,2]+table2[2,1])/sum(table2)
Misclassification3
```

```
## [1] 0.2209302
```

The Missclassification rate is 22.09% (test set)

False Positive Rate (test set):

```
FPM3 <- (34)/(4+34)
FPM3
```

```
## [1] 0.8947368
```

The false positive rate is 89.47% (test set).

False Negative Rate (test set):

```
FNM3 <- (33)/(33+15)
FNM3
```

```
## [1] 0.6875
```

The false negative rate is 68.75%

3) Compare the models in part 1) and part 2) and suggest the best model. (Give reasons).

ANSWER: The model in Part 1 is the logistic regression model and Part 2 is the Decision Tree model. The Residual Deviance for the logistic regression model (model4.5) is 177.25 whereas for the decision tree model is 0.4613 (house_tree). So therefore, in terms of which model is the best, we have to see which one has a lower residual deviance. And the model that has a lower residual deviance is the decision tree model which is “house_tree”.