Regression

Muhammad Shariq Azeem

09-24-2022

Linear Regression:

In simple terms, the point of linear regression is to find a relationship between two or more values. It helps us figure out how much our target variable, y, would change, if we change the predictor values, x.

Strengths:

- It is easy to understand and implement.
- The results are easy to interpret.

Weaknesses:

- Does not fit well with non-linear relationships.
- Prone to over-fitting.
- Sensitive to outliers.

Data:

For this assignment, I selected a data set that contains information about houses in Sydney. Information like: the number of beds, number of baths, sell price, etc.

Source of the data set: https://www.kaggle.com/datasets/mihirhalai/sydney-house-prices

Cleaning the data:

- Got rid of any rows that contained empty values.
- Only selected rows with property type 'house', to reduce the size of the data set.
- Only selected rows with a sell price of less than 1 million, to further reduce the size of the data set and eliminate any crazy values; for example, a house being sold for more than 2 Billion.
- Only selected rows of houses that have 10 or less bedrooms, to further reduce the size of the data set and eliminate any crazy values; for example, a house having 99 rooms.
- Eliminated the date and property type column because I don't need them for any calculations.

```
SydneyHousePrices <- read.csv("C:/Users/shari/Downloads/SydneyHousePrices.csv", header=T)
df <- SydneyHousePrices
df <- df[complete.cases(df[, 1:9]),]
df <- df[df$propType == "house",]
df <- df[df$sellPrice < 1000000,]
```

```
df <- df[df$bed <= 10,]
df <- df[,-1]
df <- df[,-8]
```

Step A: Divide the data into train and test.

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

Step B: Five Data Exploration functions on the training data.

1- names():

List the names of all the attributes contained in the data set.

```
names(train)
```

```
## [1] "Id"         "suburb"     "postalCode" "sellPrice"  "bed"
## [6] "bath"       "car"
```

2- dim():

Print the number of rows and the number of columns in the data set.

```
dim(train)
```

```
## [1] 56763     7
```

3- head():

Print the first n rows of the data set.

```
head(train, n = 10)
```

```
##            Id        suburb postalCode sellPrice bed bath car
## 122396 122396       Newtown       2042    525000   1    1   1
## 39715   39715 St Johns Park       2176    615000   3    1   1
## 99216   99216  Marsden Park       2765    727900   4    2   2
## 193150 193150   Mount Colah       2079    615000   3    1   1
## 178553 178553  Georges Hall       2198    869988   3    1   1
## 194197 194197   West Hoxton       2171    800000   4    2   2
## 43412   43412       Holroyd       2142    430000   3    1   1
## 97464   97464   West Pymble       2073    720000   3    1   2
## 146706 146706       Padstow       2211    840000   4    2   2
## 42798   42798     Granville       2142    600000   5    3   4
```

4- str():

List the structure of each column of the data set.

```
str(train)
```

```
## 'data.frame':    56763 obs. of  7 variables:
##  $ Id        : int  122396 39715 99216 193150 178553 194197 43412 97464 146706 42798 ...
##  $ suburb    : chr  "Newtown" "St Johns Park" "Marsden Park" "Mount Colah" ...
##  $ postalCode: int  2042 2176 2765 2079 2198 2171 2142 2073 2211 2142 ...
##  $ sellPrice : int  525000 615000 727900 615000 869988 800000 430000 720000 840000 600000 ...
##  $ bed       : num  1 3 4 3 3 4 3 3 4 5 ...
##  $ bath      : int  1 1 2 1 1 2 1 1 2 3 ...
##  $ car       : num  1 1 2 1 1 2 1 2 2 4 ...
```

5- summary():

Print basic statistics calculated from the values contained in the data set.

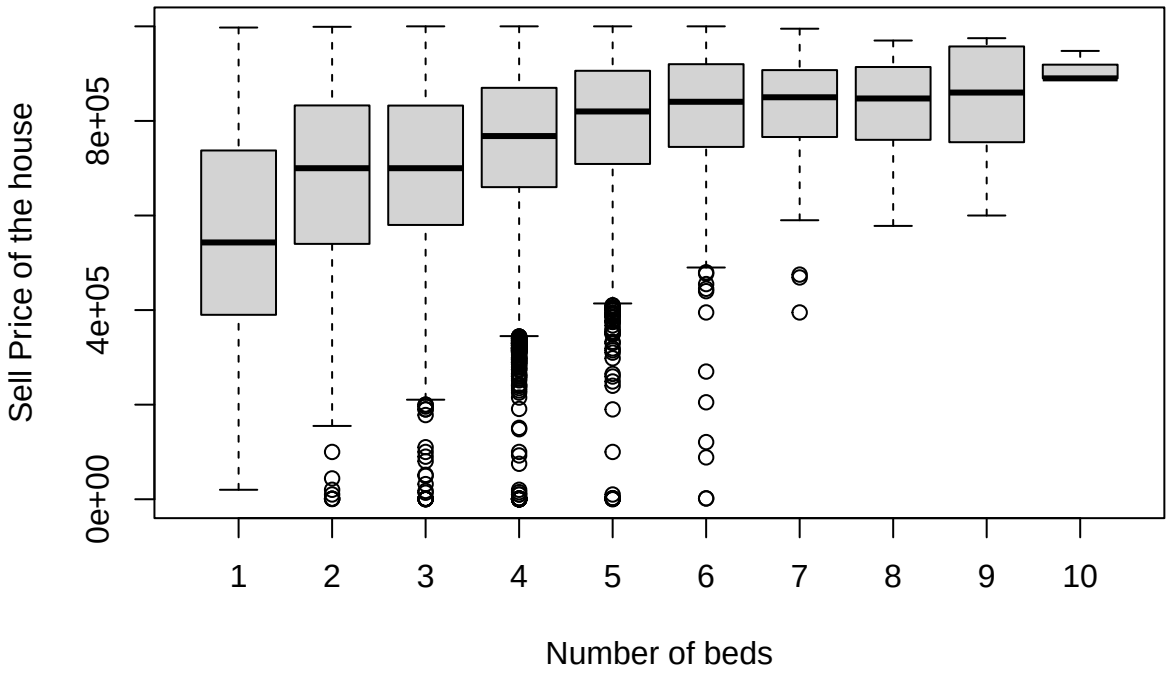
```
summary(train)
```

```
##        Id            suburb            postalCode     sellPrice
##  Min.   :   459   Length:56763       Min.   :2000   Min.   :     1
##  1st Qu.: 45522   Class :character   1st Qu.:2146   1st Qu.:610000
##  Median :103697   Mode  :character   Median :2192   Median :736000
##  Mean   :102720                      Mean   :2302   Mean   :721434
##  3rd Qu.:157919                      3rd Qu.:2560   3rd Qu.:855000
##  Max.   :199502                      Max.   :4878   Max.   :999999
##       bed             bath            car
##  Min.   : 1.00   Min.   :1.000   Min.   : 1.000
##  1st Qu.: 3.00   1st Qu.:1.000   1st Qu.: 1.000
##  Median : 3.00   Median :2.000   Median : 2.000
##  Mean   : 3.41   Mean   :1.612   Mean   : 1.828
##  3rd Qu.: 4.00   3rd Qu.:2.000   3rd Qu.: 2.000
##  Max.   :10.00   Max.   :6.000   Max.   :22.000
```

Step C: 2 Informative Graphs on the training data.

1- boxplot():

```
boxplot(train$sellPrice ~ (as.factor(train$bed)), xlab = "Number of beds",
        ylab = "Sell Price of the house")
```



2- hist():

```
hist(train$bed, breaks = c(0,1,2,3,4,5,6,7,8,9,10), xlab = "Number of bedrooms",
     ylab = "Number of houses")
```

Step D: Build a linear regression model and print its summary.

Explanation:

By looking at the output of the summary() method, I can see that, for this model, I have a slope of 43069.6, and an intercept of 574582.6. I can ignore the intercept because that is just used to fit the data. But, the slope is very important because it tells me how much y would change for each unit of change in x. In this case, if I increase the number of bedrooms in a house by 1, its price would increase by $43,069.60. This value will help me predict the price of a house based on how many rooms it has.

Another thing I can notice in the summary is the three asterisks at the end of the line for bed; that means that the number of beds is a good predictor for predicting the sell price of a house. Next, I see that the R Squared value is not so close to 1, which means that my predictor does not explain all the variance found in my model.

Lastly, I can see that my model has a F-statistic of 2655, which is greater than 1, and a very low p-value; that means that my predictor is a significant predictor, and I can ignore the null hypothesis.

```
lm1 <- lm(sellPrice~bed, data = train)
summary(lm1)
```

```
##
## Call:
## lm(formula = sellPrice ~ bed, data = train)
##
```

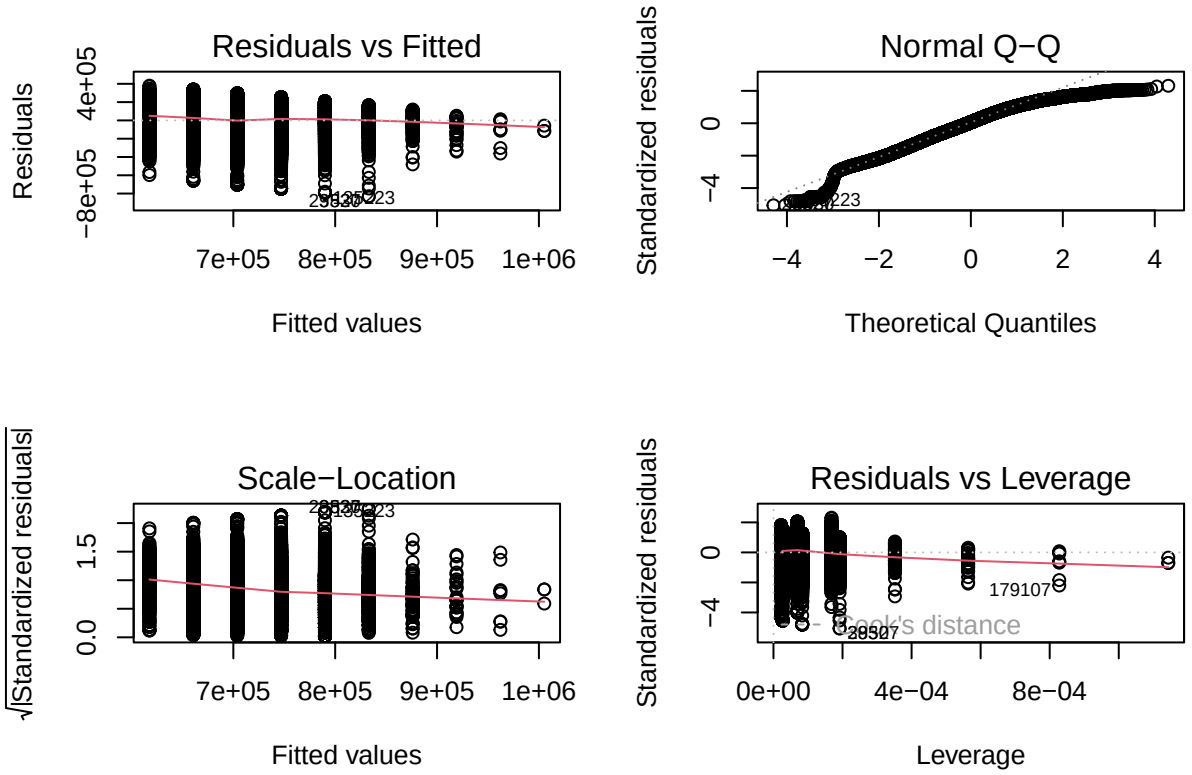
```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -831495 -108791   10139  129278  379848
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 574582.6     2932.4  195.94   <2e-16 ***
## bed          43069.6      835.9   51.52   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 164200 on 56761 degrees of freedom
## Multiple R-squared:  0.04468,    Adjusted R-squared:  0.04466
## F-statistic:  2655 on 1 and 56761 DF,  p-value: < 2.2e-16
```

Step E: Plot the residuals.

Explanation:

- Residuals vs Fitted: In this graph, I see a pretty accurate horizontal line, which is good, but I also see more variation on left than I see on the right, which means that my model did not capture all the variation in the data.
- Normal Q-Q: In this graph, I see that most of the residual points follow the dashed line, especially in the middle; this means that most of the residuals are normally distributed.
- Scale-Location: In this graph, I see that as we go from left to right, the residuals are spreading wider and wider, but as we get to almost two-thirds of the graph, the residuals start to narrow down and almost come together on the right side of the graph. Despite that, I see a fairly horizontal line which means that most of the data is of same variance.
- Residuals vs Leverage: In this graph, I don't see the red dashed line that is supposed to represent the Cook's distance, that means that all of the points are inside the Cook's distance and there is no single instance that influences the entire model.

```
par(mfrow=c(2,2))
plot(lm1)
```



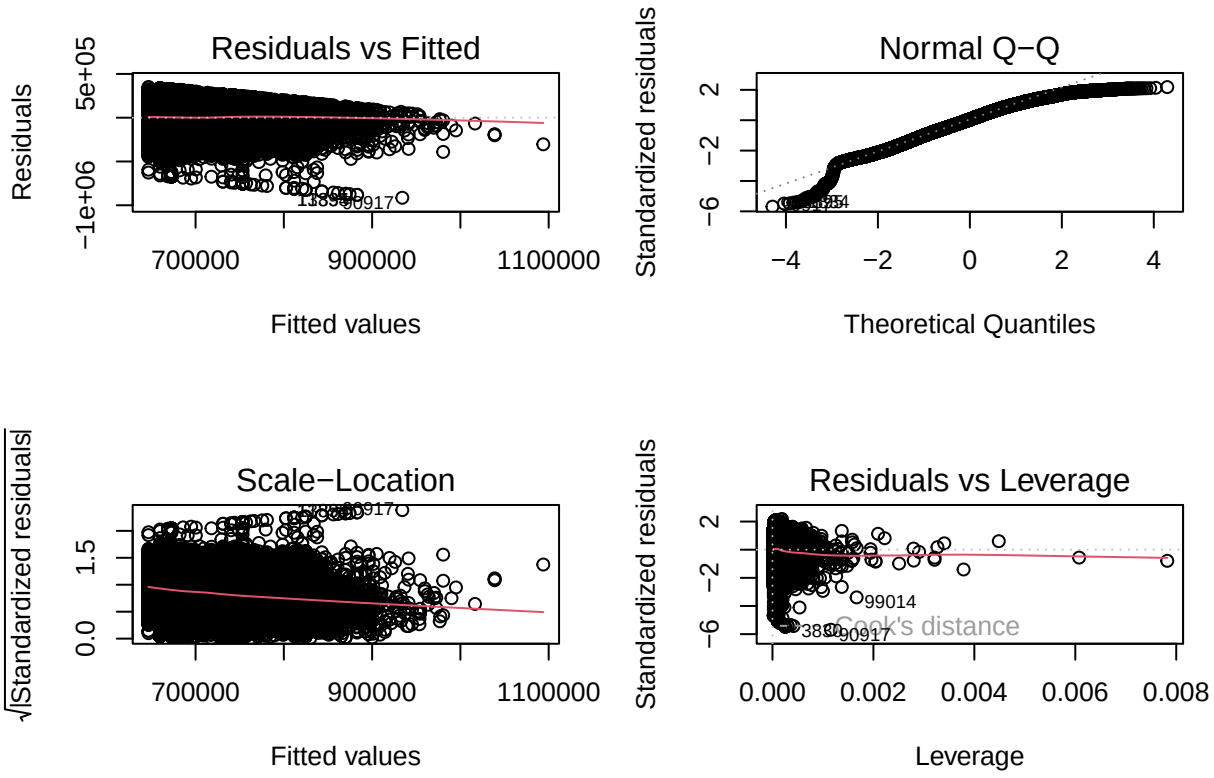
Step F: Build a multiple linear regression model, print its summary and its residual plots.

```
lm2 <- lm(sellPrice~bed+bath+car, data = train)
summary(lm2)
```

```
##
## Call:
## lm(formula = sellPrice ~ bed + bath + car, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -914196 -105498    7227  123621  350856
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   566096       2934  192.97   <2e-16 ***
## bed            13064       1021   12.80   <2e-16 ***
## bath           58304       1262   46.20   <2e-16 ***
## car             9181        705   13.02   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 160800 on 56759 degrees of freedom
## Multiple R-squared:  0.08377,    Adjusted R-squared:  0.08373
## F-statistic:  1730 on 3 and 56759 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lm2)
```



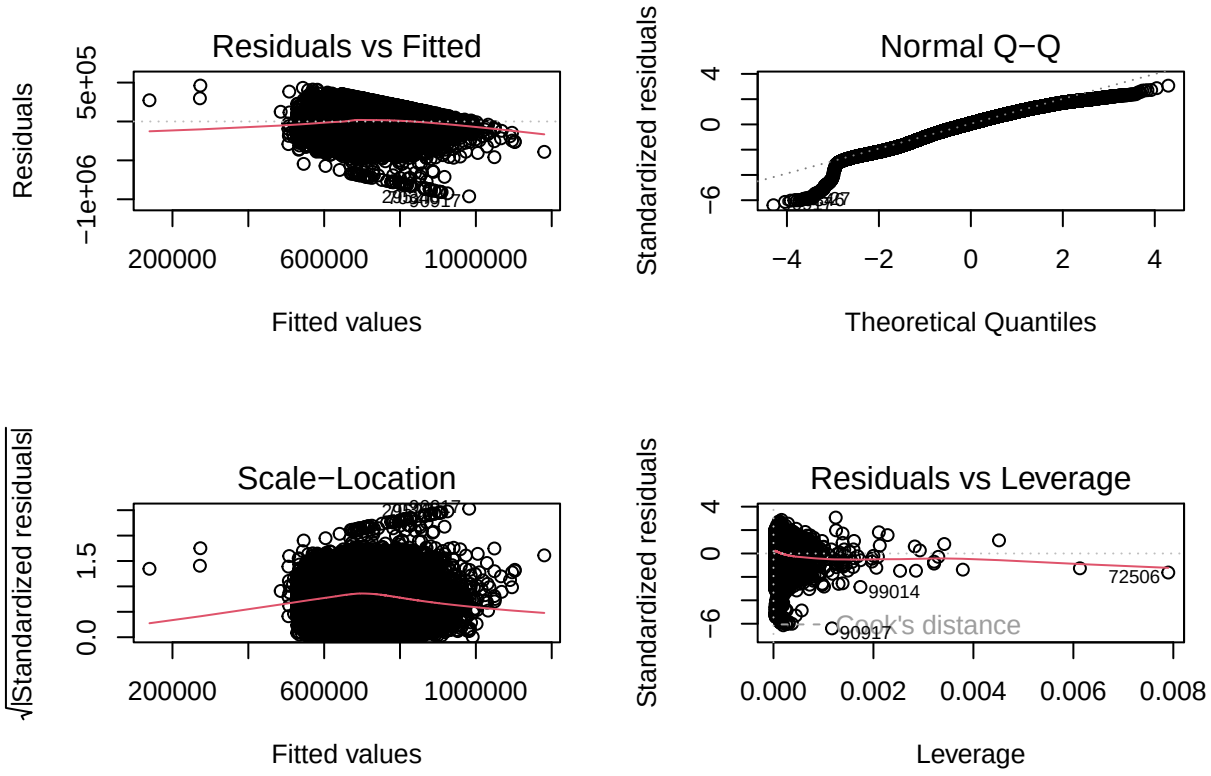
Step G: Build a third linear regression model, print its summary and its residual plots.

```
lm3 <- lm(sellPrice~bed+bath+car+postalCode, data = train)
summary(lm3)
```

```
##
## Call:
## lm(formula = sellPrice ~ bed + bath + car + postalCode, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -962362  -91487   12505  108358  461484
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 1080875.66    6380.04  169.41   <2e-16 ***
## bed           25983.45     966.61   26.88   <2e-16 ***
## bath          51676.22    1183.77   43.65   <2e-16 ***
## car           12150.44     660.87   18.39   <2e-16 ***
## postalCode     -240.50       2.69  -89.39   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 150600 on 56758 degrees of freedom
## Multiple R-squared:  0.1969, Adjusted R-squared:  0.1968
## F-statistic:  3478 on 4 and 56758 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lm3)
```



Step H: Compare the models.

First, if we look at the residual plots of each of the three models, we can see that there is not that much difference in them. The residuals are clustered and further apart at almost the exact same areas. The only difference I can notice is in the Scale-Location and Residuals vs Fitted graphs of model 3; the graph shows some points on the left of the cluster, which was not shown in the residual plots of models 1 and 2.

Next, if we look at the summaries, we can notice that every predictor in each model is marked with three asterisks; that means that all of them were good predictors for our target variable. But, if we look at their R^2 values, even though model 2 had an increase of 0.04, it's not that significant. On the other hand, for

model 3, the R^2 value is 0.19, which is almost 5 times more than the R^2 value of model 1. That means that the predictors in model 3 explain the variance in the model much better than the predictors in model 1.

Therefore, I believe that the model 3 is the better one out of all three models.

Step I: Use all three models to predict on the test data. Evaluate the results, using correlation and mse.

Explanation:

After looking at the Correlation and MSE values of all three models, I can see that they confirm my previous thoughts. Even though the predictors in each model were considered good predictors, neither of the three models explained all the variance in our data. That is why we can see that neither of the correlation values are close to 1. But, similar to the R^2 values observed earlier, the correlation value of model 3 is over 2 times more than the correlation value of model 1. This also tells us that model 3 is better than the previous two models. Model 3 also gives us less error values than the other 2 models.

If I have to guess on why these type of results were produced, I would say it is because the price of a house is not solely dependent on where it is located and how many rooms it has, but it also has some external factors that can not be included in a data set.

```
pred1 <- predict(lm1, newdata=test)
cor1 <- cor(pred1, test$sellPrice)
mse1 <- mean((pred1-test$sellPrice)^2)

print("Model 1")
```

```
## [1] "Model 1"
```

```
print(paste('Correlation:', cor1))
```

```
## [1] "Correlation: 0.205334727086427"
```

```
print(paste('MSE:', mse1))
```

```
## [1] "MSE: 26978320964.8711"
```

```
pred2 <- predict(lm2, newdata=test)
cor2 <- cor(pred2, test$sellPrice)
mse2 <- mean((pred2-test$sellPrice)^2)

print("Model 2")
```

```
## [1] "Model 2"
```

```
print(paste('Correlation:', cor2))
```

```
## [1] "Correlation: 0.283004496628883"
```

```
print(paste('MSE:', mse2))
```

```
## [1] "MSE: 25911468170.8727"
```

```
pred3 <- predict(lm3, newdata=test)
cor3 <- cor(pred3, test$sellPrice)
mse3 <- mean((pred3-test$sellPrice)^2)

print("Model 3")
```

```
## [1] "Model 3"
```

```
print(paste('Correlation:', cor3))
```

```
## [1] "Correlation: 0.443846589869125"
```

```
print(paste('MSE:', mse3))
```

```
## [1] "MSE: 22621231429.3049"
```