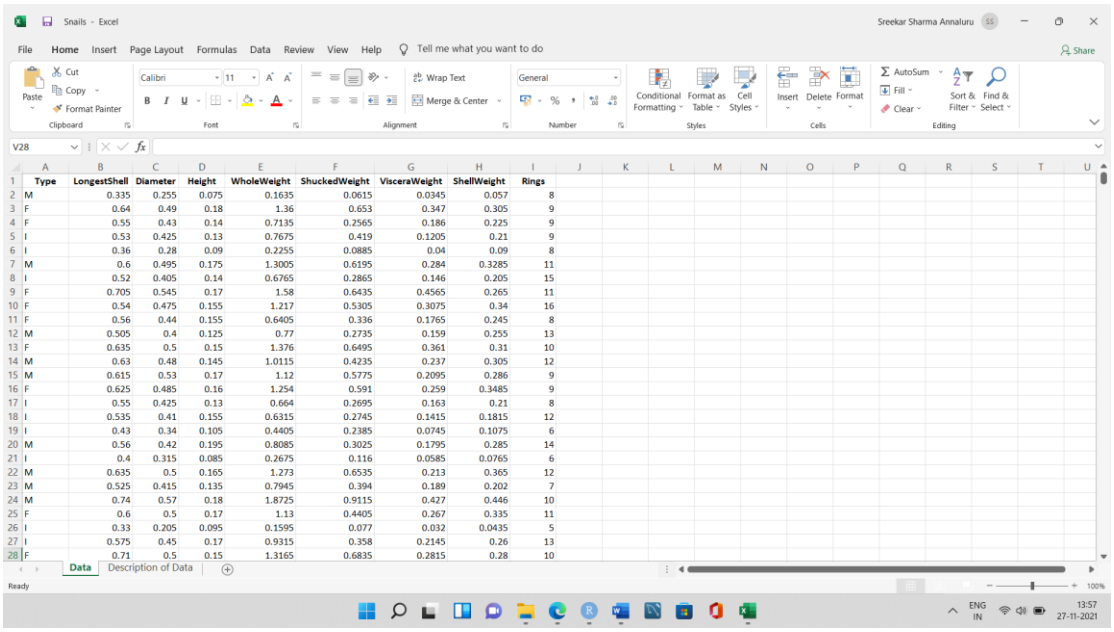
Home Work Assignment-5

Note: First 17 pages of this assignment are detailed analysis of the work I have done. The labelled answers to the questions (in the assignment) will start from page 18. Also, I am attaching the models to the headings of corresponding regression technique. Though I have answered everything in these 17 pages, I want to be clear. Hence, I answered them separately, with labels. Please note this!!!

- Before going into the actual assignment (predicting the age of snails), have a look at the Snails.xlsx data, and summarize the key features or trends in the data set.



Type	LongestShell	Diameter	Height	WholeWeight	ShuckedWeight	VisceraWeight	ShellWeight	Rings
M	0.335	0.255	0.075	0.1635	0.0615	0.0345	0.057	8
F	0.64	0.49	0.18	1.36	0.653	0.347	0.305	9
F	0.55	0.43	0.14	0.7135	0.2565	0.186	0.225	9
I	0.53	0.425	0.13	0.7675	0.419	0.1205	0.21	9
I	0.36	0.28	0.09	0.2255	0.0885	0.04	0.09	8
M	0.6	0.495	0.175	1.3005	0.6195	0.284	0.3285	11
I	0.52	0.405	0.14	0.6765	0.2865	0.146	0.205	15
F	0.705	0.545	0.17	1.58	0.6435	0.4565	0.265	11
F	0.54	0.475	0.155	1.217	0.5305	0.3075	0.34	16
F	0.56	0.44	0.155	0.6405	0.336	0.1765	0.245	8
M	0.505	0.4	0.125	0.77	0.2735	0.159	0.255	13
F	0.635	0.5	0.15	1.376	0.6495	0.361	0.31	10
M	0.63	0.48	0.145	1.0115	0.4235	0.237	0.305	12
M	0.615	0.53	0.17	1.12	0.5775	0.2095	0.286	9
F	0.625	0.485	0.16	1.254	0.591	0.259	0.3485	9
I	0.55	0.425	0.13	0.664	0.2695	0.163	0.21	8
I	0.535	0.41	0.155	0.6315	0.2745	0.1415	0.1815	12
I	0.43	0.34	0.105	0.4405	0.2385	0.0745	0.1075	6
M	0.56	0.42	0.195	0.8085	0.3025	0.1795	0.285	14
I	0.4	0.315	0.085	0.2675	0.116	0.0585	0.0765	6
M	0.635	0.5	0.165	1.273	0.6535	0.213	0.365	12
M	0.525	0.415	0.135	0.7945	0.394	0.189	0.202	7
M	0.74	0.57	0.18	1.8725	0.9115	0.427	0.446	10
F	0.6	0.5	0.17	1.13	0.4405	0.267	0.335	11
I	0.33	0.205	0.095	0.1595	0.077	0.032	0.0435	5
I	0.575	0.45	0.17	0.9315	0.358	0.2145	0.26	13
F	0.71	0.5	0.15	1.3165	0.6835	0.2815	0.28	10

All the units are in MMGS system. It looks like most of the data is less than 1, except for the Rings. The description is given in the next page, which is not that important for prediction (for interpreting-yes).

Target: Predicting the Age (Rings) of Snails.

Applied Techniques: Linear Regression, Random Forest Regression, and KNN Regression.

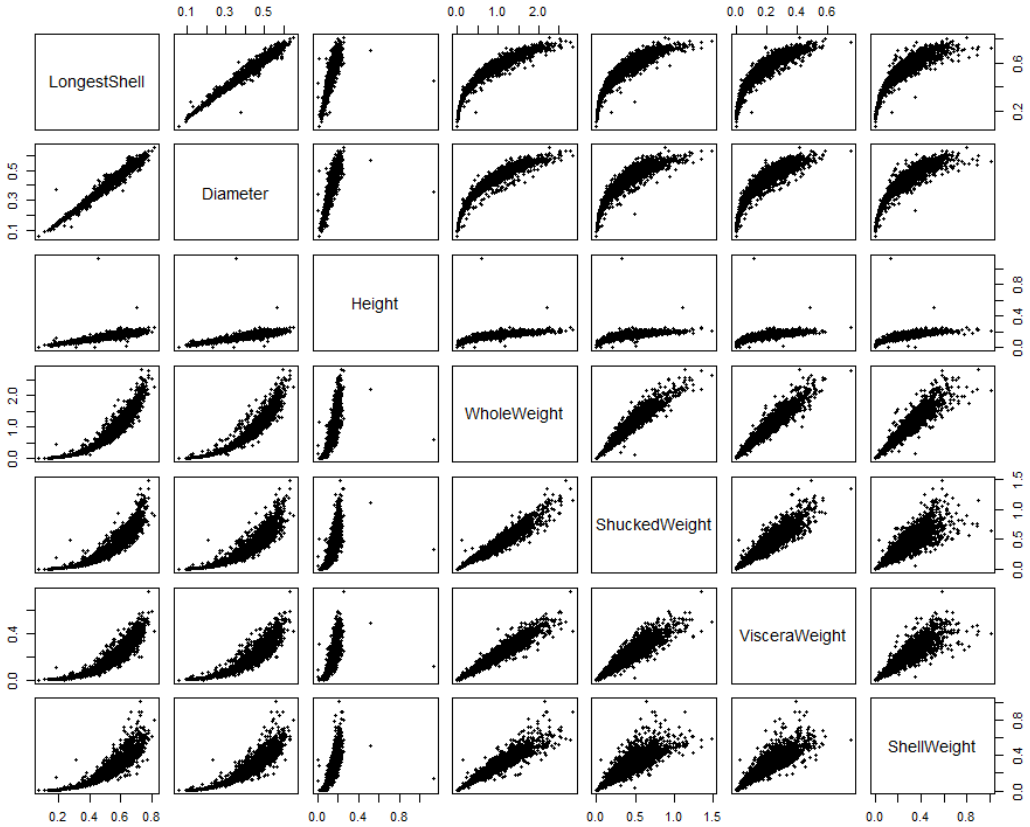
Supporting Methods/Tools: 10-Fold Cross Validation, Standardization of Parameters (For KNN), Validation Set, Subset Selection, and many built-in libraries! (I will explain everything in detail).

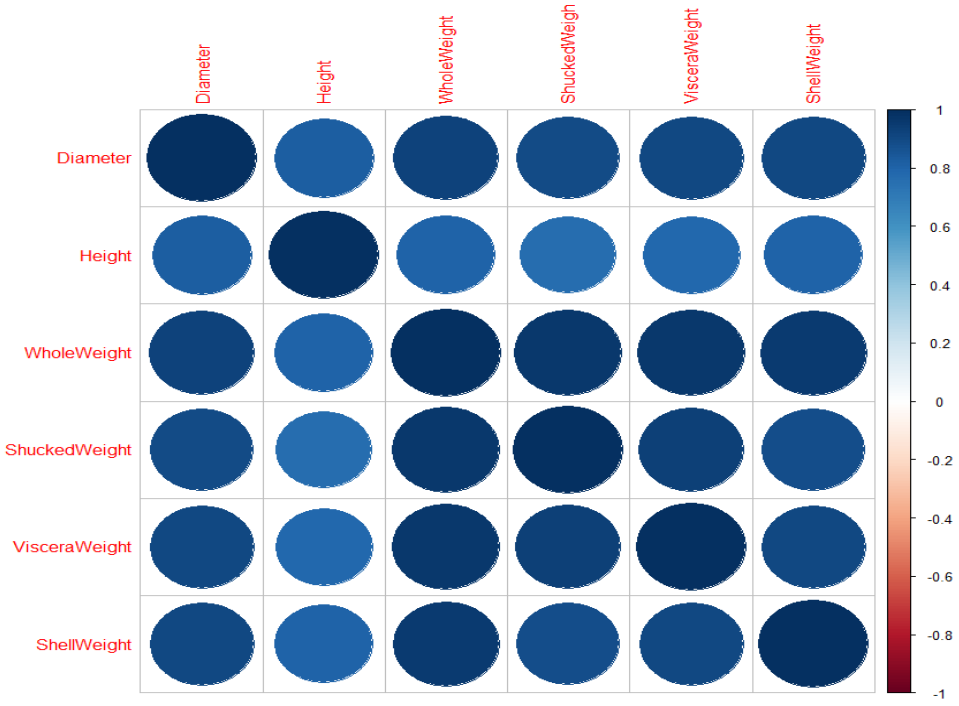
"Predicting the Age of Snails using Regression"

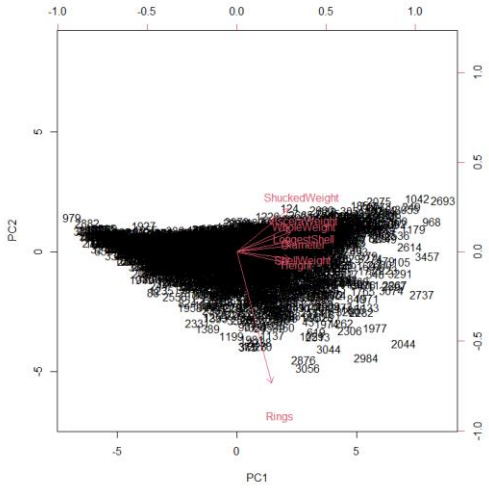
- After importing the Snails data set from my system into R. There are no "NA" values in the data set. Using the methods view (Snails), and summary (Snails), we can see the data in R. The key takeaway from these two methods is that **Whole Weight is a linear function of other weight predictors (Shell, Shucked, and Viscera) with unknown**

mass of water/blood. Also, I have observed that the **minimum value for predictor "Height" is 0**. Practically, this is not possible I will investigate into it!

- Now let's see how the predictors are related using scatter plot using **pairs (Snails)**. I have used the correlation plot as well **corrplot()**, from corrrplot library. I have performed Principal Component Analysis on the same just to cross check the correlation.







1. The most important point is the data is hugely correlated! For instance, Diameter and Whole Weight is 92% correlated! Fun Fact: The least correlated pair is (Shucked Weight, Height). Even these two are 76.5% correlated! This is excluding rings.
   Note: I have included rings in a separate scatter plot (it will be in my R file), according to that with rings all the predictors are 50-60% correlated!
2. As I have mentioned earlier, the Whole Weight seems to be highly correlated with the other weight predictors.
3. Principal Component Analysis is validating the first statement. You can see all the predictors are pretty close on both the principal components. Shucked Weight is slightly away from the rest.
4. Most of the Rings are between 5 and 15.

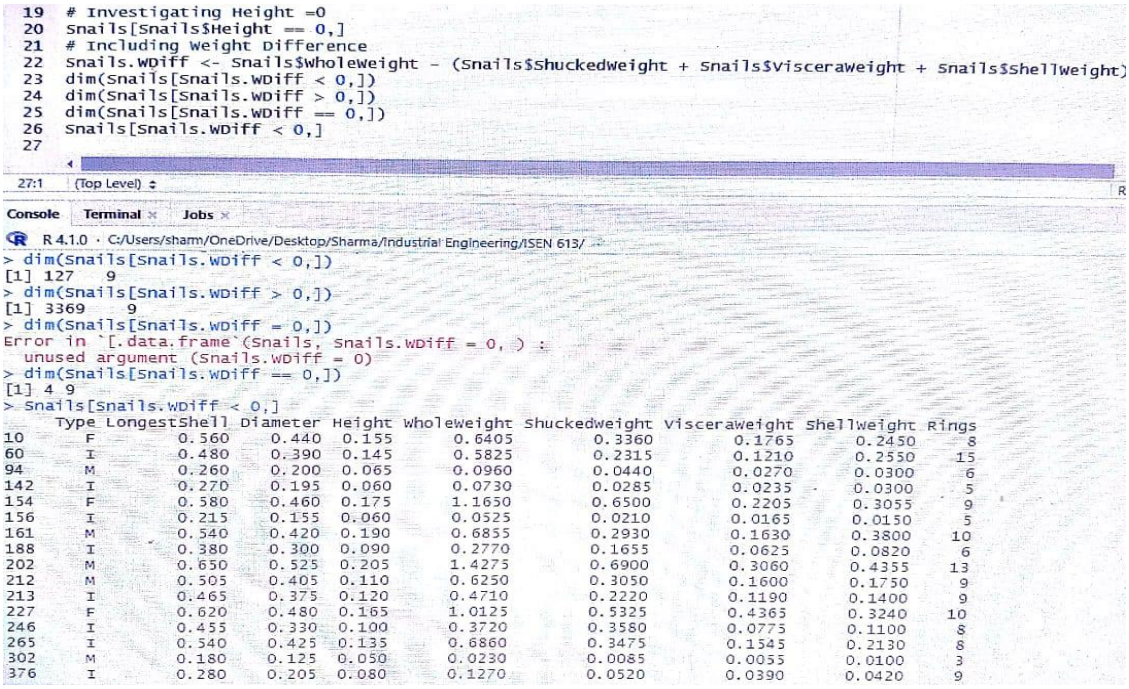
❖ Investigating the Zero Height case. There are two points with Height = 0. Have a look at those 2 points.

```
> # Investigating Height =0
> Snails[Snails$Height == 0,]
     Type LongestShell Diameter Height WholeWeight ShuckedWeight VisceraWeight ShellWeight Rings
1597    I        0.430     0.34      0       0.428         0.2065        0.0860       0.1150     8
2638    I        0.315     0.23      0       0.134         0.0575        0.0285       0.3505     6
> summary(Snails)
     Type         LongestShell      Diameter          Height        WholeWeight      ShuckedWeight
 Length:3500        Min.   :0.075   Min.   :0.0550   Min.   :0.0000   Min.   :0.0020   Min.   :0.0010
 Class :character   1st Qu.:0.450   1st Qu.:0.3500   1st Qu.:0.1150   1st Qu.:0.4404   1st Qu.:0.1850
 Mode  :character   Median :0.545   Median :0.4350   Median :0.1400   Median :0.7985   Median :0.3355
                    Mean   :0.524   Mean   :0.4078   Mean   :0.1396   Mean   :0.8292   Mean   :0.3593
                    3rd Qu.:0.615   3rd Qu.:0.4800   3rd Qu.:0.1650   3rd Qu.:1.1566   3rd Qu.:0.5015
                    Max.   :0.815   Max.   :0.6500   Max.   :1.1300   Max.   :2.8255   Max.   :1.4880
 VisceraWeight      ShellWeight        Rings
 Min.   :0.00050   Min.   :0.0015   Min.   : 1.000
 1st Qu.:0.09287   1st Qu.:0.1300   1st Qu.: 8.000
 Median :0.17125   Median :0.2328   Median : 9.000
 Mean   :0.18061   Mean   :0.2393   Mean   : 9.949
 3rd Qu.:0.25362   3rd Qu.:0.3300   3rd Qu.:11.000
 Max.   :0.76000   Max.   :1.0050   Max.   :29.000
```

The other values seem valid. Also, for the predictor Whole Weight, we see that these values are really small compared to rest of observation and they are below first quantile (Q1). This tells us that this **might not be a data error** and we cannot exclude it from the dataset.

❖ Checking for the Linearity with respect to Whole Weight. By intuition the Whole Weight has to be the summation of the remaining weight, or in worst case less than

the sum of weights (because of losses). However, it cannot be greater than the Whole Weight. In our date we have 127 observations are violating this rule. Let's have a look.



```
19  # Investigating Height =0
20  Snails[Snails$Height == 0,]
21  # Including Weight Difference
22  Snails.WDiff <- Snails$WholeWeight - (Snails$ShuckedWeight + Snails$VisceraWeight + Snails$ShellWeight)
23  dim(Snails[Snails.WDiff < 0,])
24  dim(Snails[Snails.WDiff > 0,])
25  dim(Snails[Snails.WDiff == 0,])
26  Snails[Snails.WDiff < 0,]
27
```

27:1 (Top Level)

Console Terminal Jobs

R 4.1.0 · C:/Users/sharm/OneDrive/Desktop/Sharma/Industrial Engineering/ISEN 613/

```
> dim(Snails[Snails.WDiff < 0,])
[1] 127   9
> dim(Snails[Snails.WDiff > 0,])
[1] 3369    9
> dim(Snails[Snails.WDiff = 0,])
Error in `[.data.frame`(Snails, Snails.WDiff = 0, ) :
  unused argument (Snails.WDiff = 0)
> dim(Snails[Snails.WDiff == 0,])
[1] 4 9
> Snails[Snails.WDiff < 0,]
```

	Type	LongestShell	Diameter	Height	WholeWeight	ShuckedWeight	VisceraWeight	ShellWeight	Rings
10	F	0.560	0.440	0.155	0.6405	0.3360	0.1765	0.2450	8
60	I	0.480	0.390	0.145	0.5825	0.2315	0.1210	0.2550	15
94	M	0.260	0.200	0.065	0.0960	0.0440	0.0270	0.0300	6
142	I	0.270	0.195	0.060	0.0730	0.0285	0.0235	0.0300	5
154	F	0.580	0.460	0.175	1.1650	0.6500	0.2205	0.3055	9
156	I	0.215	0.155	0.060	0.0525	0.0210	0.0165	0.0150	5
161	M	0.540	0.420	0.190	0.6855	0.2930	0.1630	0.3800	10
188	I	0.380	0.300	0.090	0.2770	0.1655	0.0625	0.0820	6
202	M	0.650	0.525	0.205	1.4275	0.6900	0.3060	0.4355	13
212	M	0.505	0.405	0.110	0.6250	0.3050	0.1600	0.1750	9
213	I	0.465	0.375	0.120	0.4710	0.2220	0.1190	0.1400	9
227	F	0.620	0.480	0.165	1.0125	0.5325	0.4365	0.3240	10
246	I	0.455	0.330	0.100	0.3720	0.3580	0.0775	0.1100	8
265	I	0.540	0.425	0.135	0.6860	0.3475	0.1545	0.2130	8
302	M	0.180	0.125	0.050	0.0230	0.0085	0.0055	0.0100	3
376	I	0.280	0.205	0.080	0.1270	0.0520	0.0390	0.0420	9

There are only 4 observations with Whole Weight = Sum (Viscera Weight + Shell Weight + Shucked Weight). Those 127 observations seem to be counterintuitive they might be error while loading

Note that these 127 observations have a combined weight greater than Whole Weight. When we look at 10 such observations, it seems that the other values are correct and there is no similarity and hence we are sure this might be a **data entry error**.

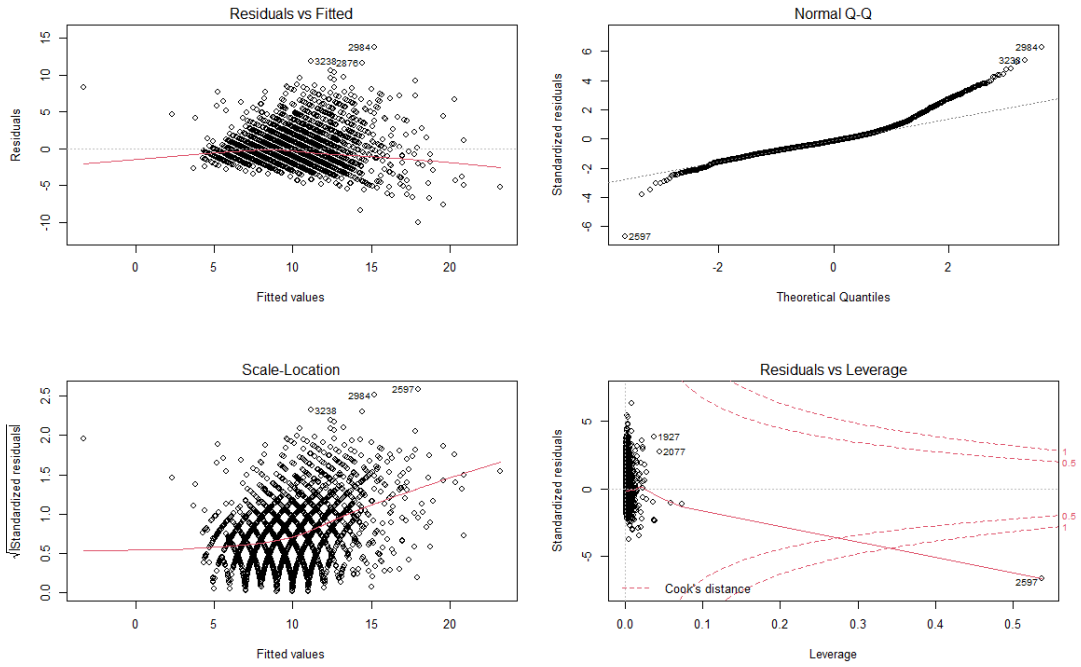
Conclusion: The main reason behind doing this is to check the status of the data. It looks like there are some errors. Somewhat similar to "Exploratory Data Analysis", not entirely though. We can do something about this but for this assignment it is not required. As the data is highly correlated this data entry error is not a huge worry.

Check the attached R-file for more details.

"*Linear Regression Model*"

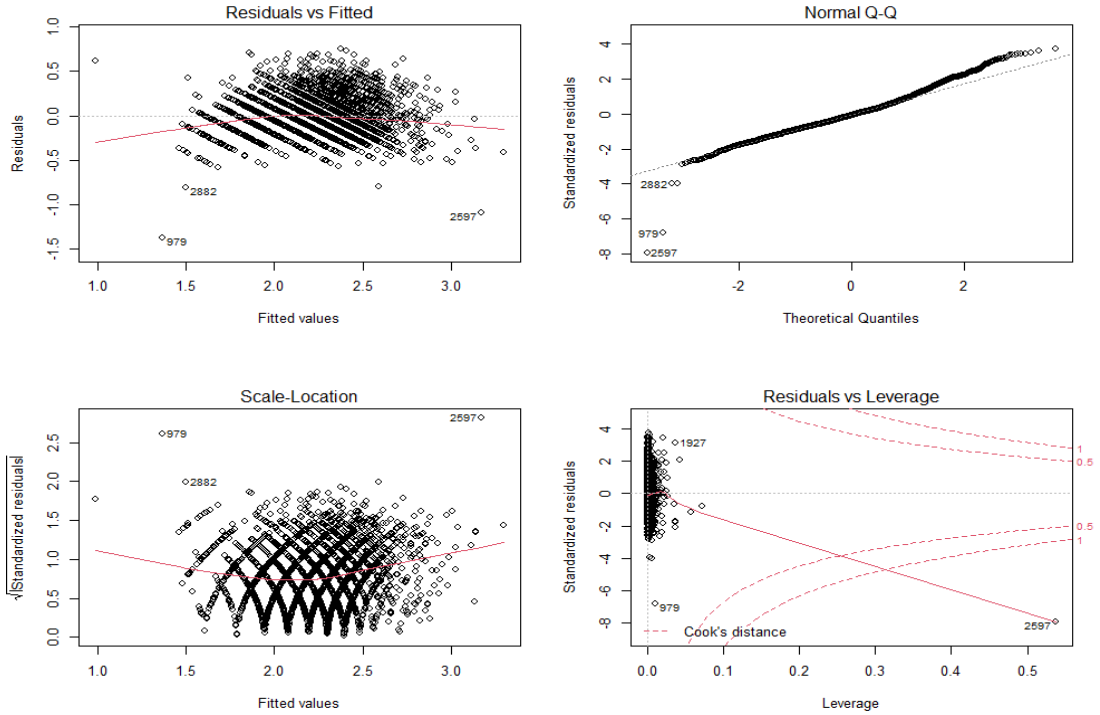
- Before going to the Linear Regression, I would like to comment on Ridge Regression and Lasso, and the reason why I didn't choose them. I have performed both those techniques in my R file. I got a test R-squared of around 54% (with best lambda using 10-fold cross validation), which is less than my FinalModel_LR, which I have submitted as my final model. You can refer to the R code for more details.

- First things first, I have built a simple multiple regression using all the predictors, without any interactions and transformations. Have a look at the plot.



As you can see the there are "Non-Constant Variance Terms". Hence, I have applied logarithm to the Rings (to remove **Heteroscedasticity**) and there is a huge improvement. Also, the data is hugely deviated from the Normal. High Leverage Points and Outliers are not a huge problem of concern for this data, as they are only a couple.

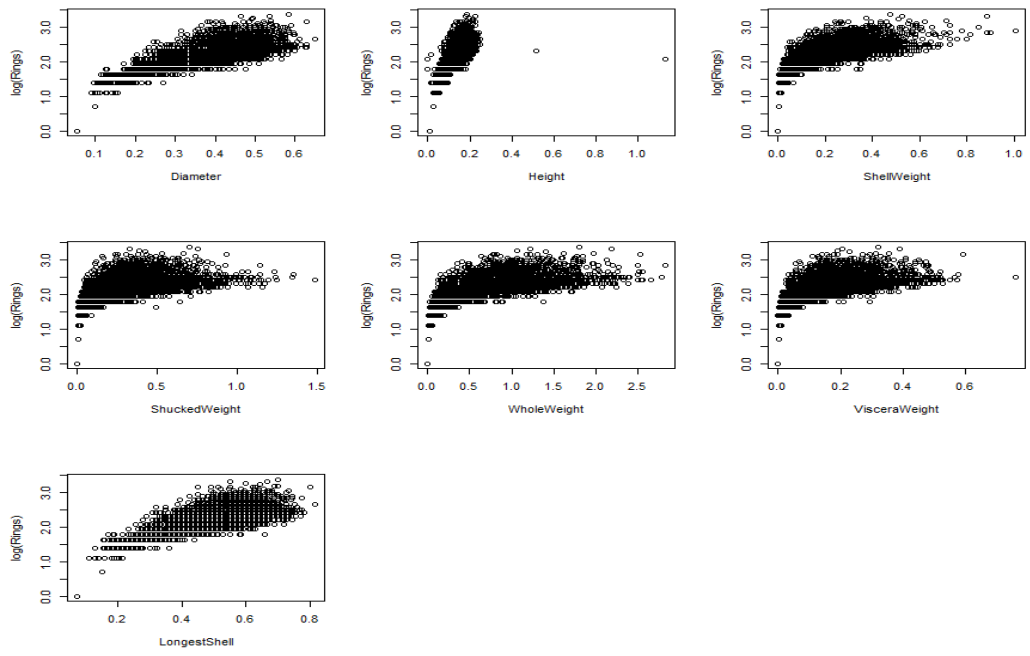
- After removing heteroscedasticity, the training R-Squared has increased to 0.6 from 0.54. Updated model plot {This is the only instance where I have used training-Squared}. Everywhere else I have used test R-Squared. That's because I didn't split the data yet!



Now the residuals look a lot better. Also, you can see the deviation from normality in Q-Q plot has improved. Hence applying logarithm to Rings is very useful, in fact mandatory.

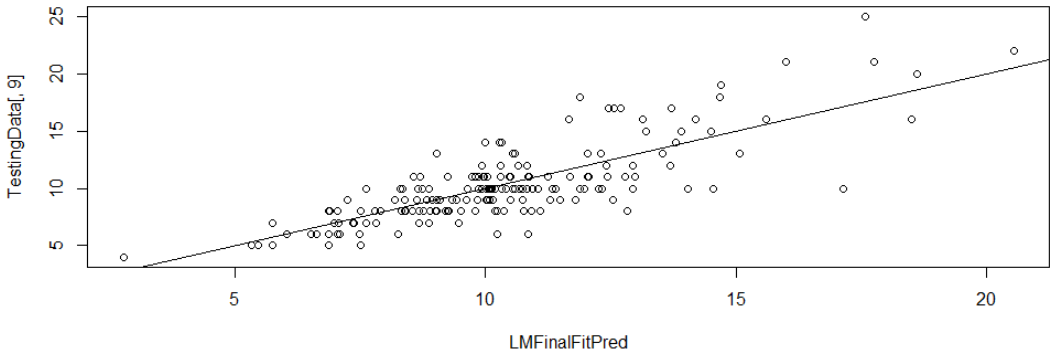
Note that the relation is slightly non-linear which I will address as I go ahead!!

- Now let's split the data into Training and Testing. I have gone with 95:5 split. "**I will explain my logic behind this in a few minutes"**. Before going to the Test R-Squared. Let's see how the plots of predictors vs Rings look like. This will give the reason behind my split.

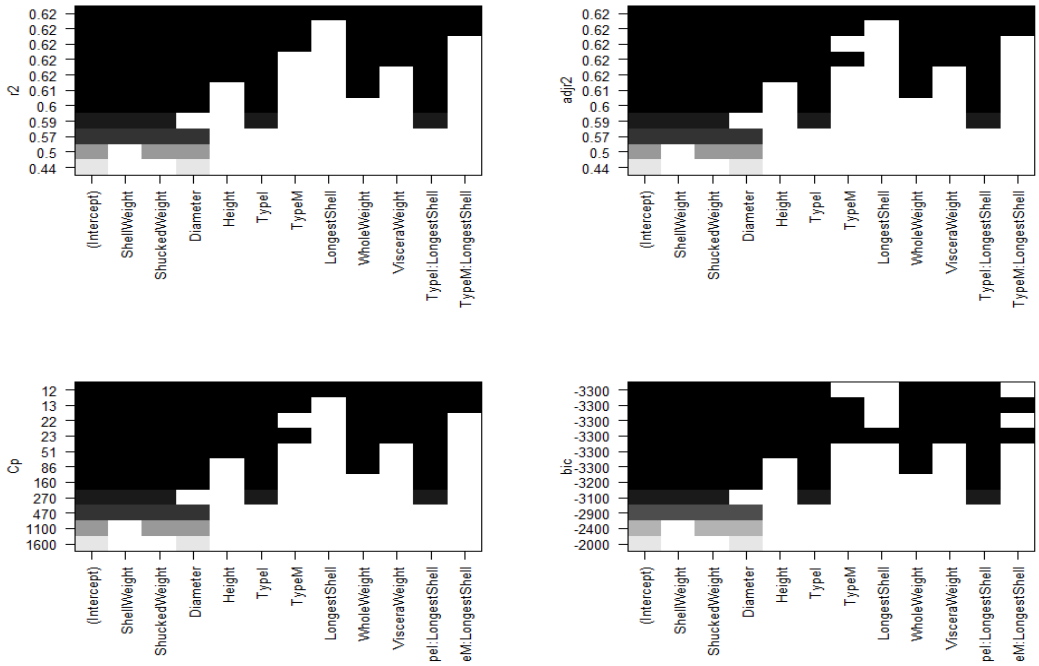


The predictors are almost linearly related to log (Rings). As soon as I see this plot, I am very confidant that Linear Model will perform well on this data. Hence, I have given very few test data (5%). **One important point to note is that predictors like, Whole Weight, Viscera Weight, Shucked Weight, Shell Weight, and Diameter are slightly non-linear**. I have performed 10-fold Cross Validation and solved that problem. By transforming the predictors my Linear Regression test R-Squared is **64% (Don't forget seed 1**.

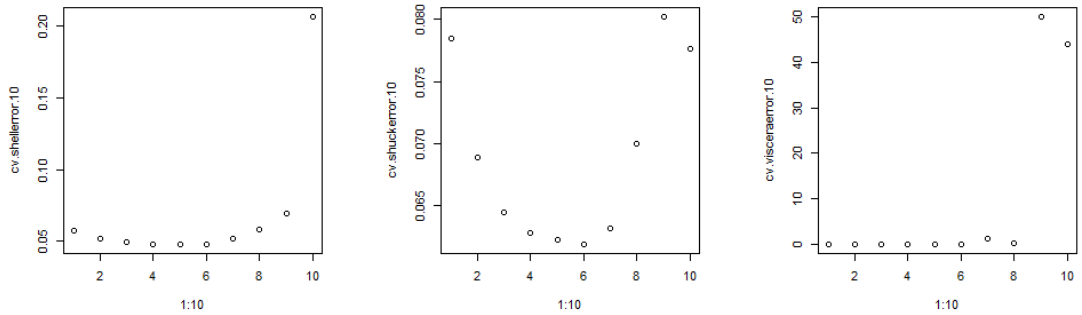
- Here is the plot of fitted values vs actual values.

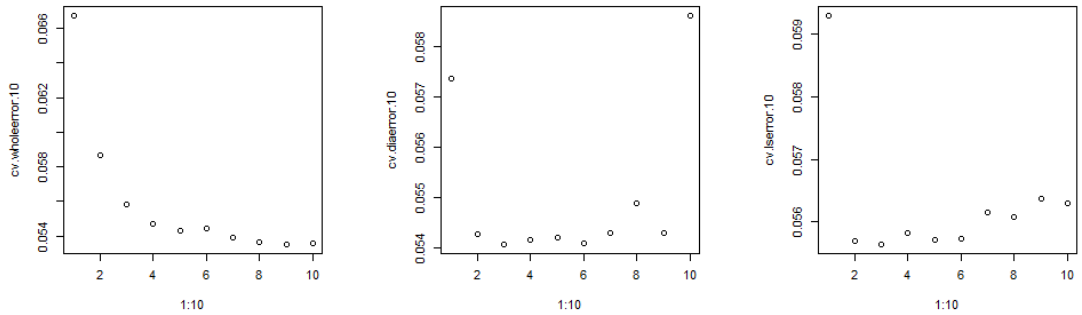


- Now I have performed best subset selection to choose the best subsets. I have used the argument nvmax as 15. If you observe the plot the values of Adjusted R2, Cp, and BIC have reached a **saturation limit after 8 variables**. In fact, the minimum value of BIC is exactly at 8. The minimum values of Adjusted R2 and Cp are 11. The key point is "**Minimum number of predictors in the model should be at least 8!**" If we are interested in interpretability, we can stick to 8/9. But we are worried about predictability hence we can go as far as we can, provided it should increase test R-Squared.

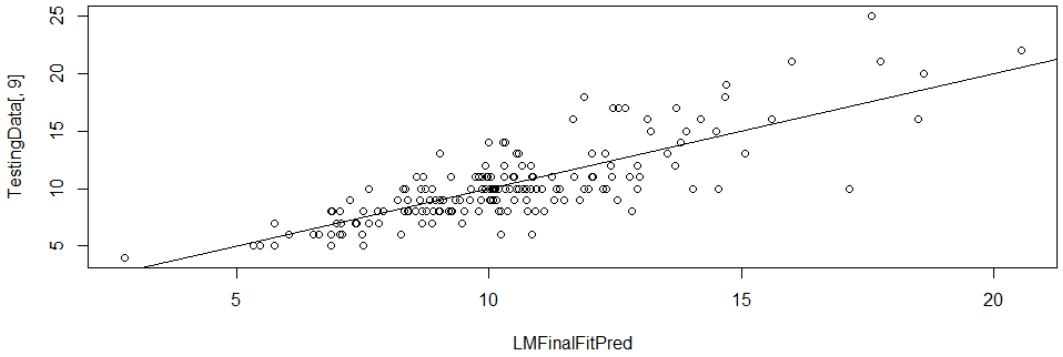


❖ Now I have performed cross validation on the predictors. As we have seen some of them are slightly non-linear. Before going to that, I would like to explain why I have gone with 10-fold CV. The one-word answer is **"Bias-Variance Tradeoff"**. LOOCV gives the least bias, however, but it tends to have higher variance compared to k-fold CV. The reason is LOOCV makes the output highly correlated hence it has higher variance. Remember that out date is already correlated. Keeping this point in mind I didn't perform LOOCV. Why 10 over 5? Well, there is no strong answer. We can go with 5 as well but I have read, in ISLR textbook, that 10 is more preferred as it gives results close to the true test results. There are a lot of 10-fold CV's in text book but very few 5-fold. Let's have a look at the plots. The minimum values of test errors for the predictors are: {**Shell Weight - 6, Shucked Weight - 6, Viscera Weight - 4, Whole Weight - 9, Longest Shell - 3, and Diameter- 3**}. As we are not worried about interpretability, I have transformed all these predictors. **There is a good improvement in test R-Squared**, which is very crucial for predictability.





After all these techniques I got a **training Adjusted R-Squared of 0.6637**. Corresponding **testing R-Squared is 0.64,** which is a good improvement from previous model. As you can see, we cannot improve the R-Squared, irrespective of the number of parameters, recollect the BIC and Adj R2 graphs. They have saturated after a certain limit. Finally, lets have a look at the plot between Final Fitted Values and Testing data.

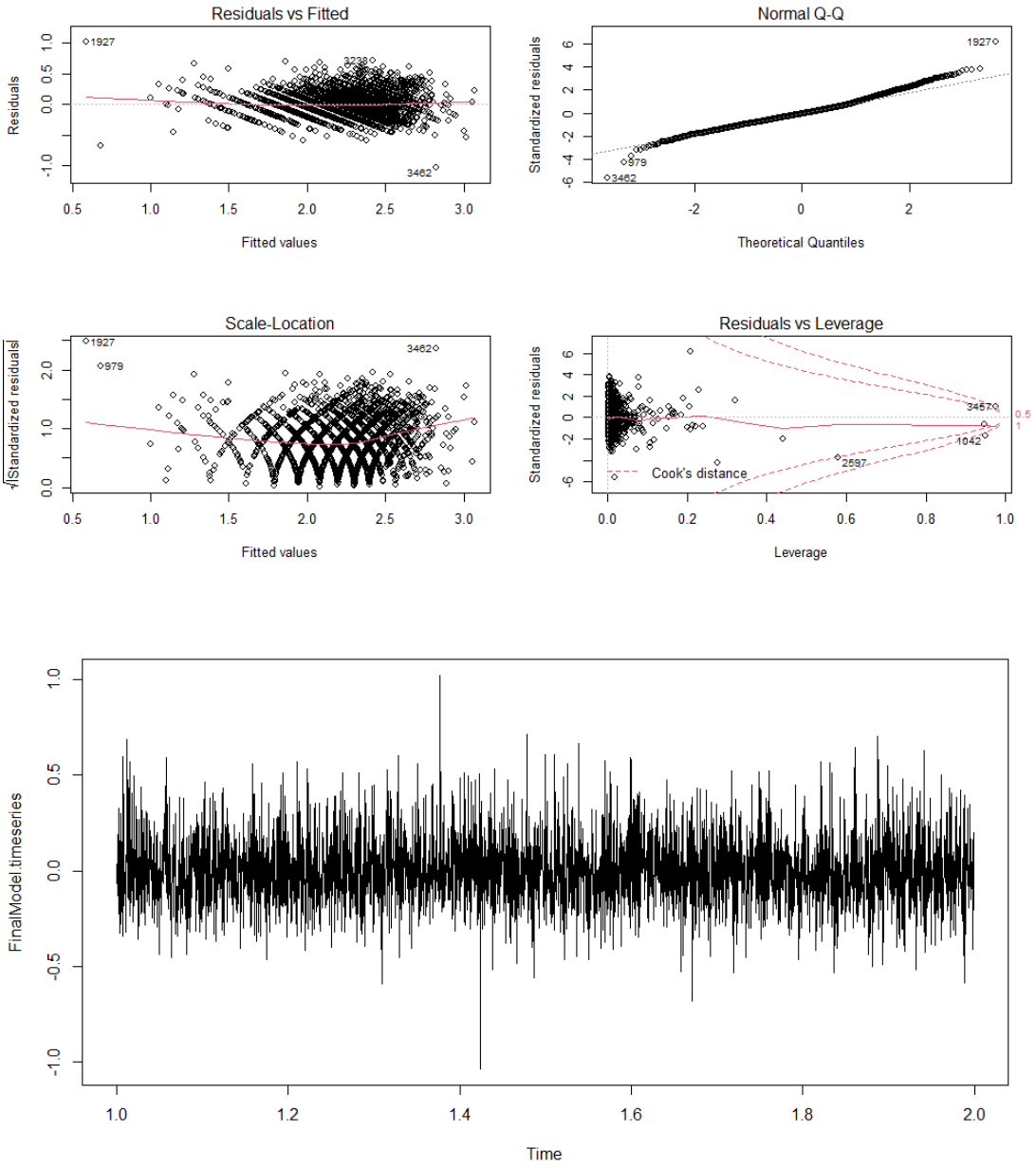


You can see the relation is “Linear”. It is very clear. Hence this is my first model. The pic of the model is:

HW3Solutions.R | Assignment-5.R | Assignment-5 Practice.R* | Model-1 @ Linear Regression.R*

Source on Save | Run | Source

```
# Final Linear Model
set.seed(202)
par(mfrow = c(1,1))
trainingindex<- sample(1:nrow(Snails), 95*nrow(Snails)/100,replace=F)

TrainingData<- Snails[trainingindex,]
TestingData <- Snails[-trainingindex,]

FinalModel_LR <- lm(log(Rings) ~ log(I(ShellWeight)^6) + poly(ShuckedWeight,6) +poly(Diameter,3) +
                    Type*poly(LongestShell,3) + Height + poly(wholeweight,9) + poly(VisceraWeight,4)
                    WholeWeight:VisceraWeight , data = TrainingData)
summary(FinalModel_LR)

# Test R-Squared for final model!
LMFinalFitPred <- predict(FinalModel_LR, TestingData )
TestRSS <- sum((LMFinalFitPred - log(TestingData[,9]))^2)
TestTSS <- sum((mean(log(TestingData[,9]))- log(TestingData[,9]))^2)
1 TestRSS/TestTSS
plot(LMFinalFitPred, log(TestingData[,9]))
```



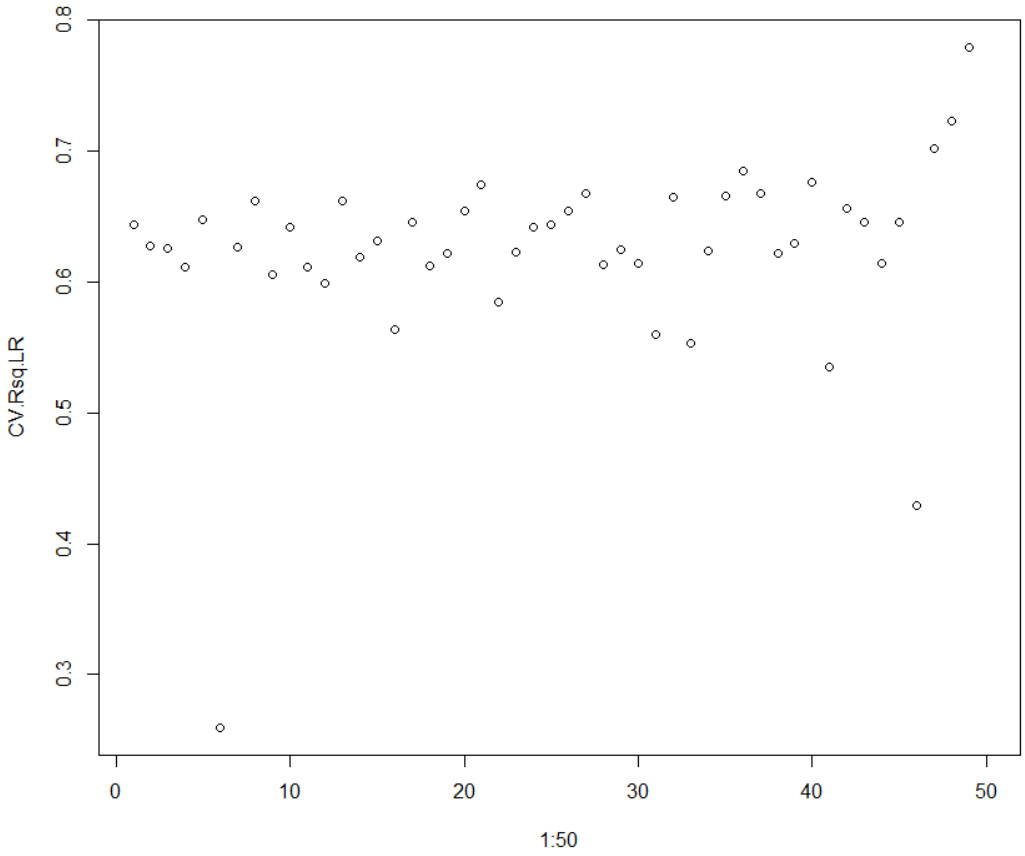
Finally, I have performed 10-fold CV on my final model to find out test error. I got an error of **0.0369** for this model.

```
    Null deviance: 360.48  on 3499  degrees of freedom
Residual deviance: 121.40  on 3463  degrees of freedom
AIC: -1756.5

Number of Fisher Scoring iterations: 2

> library(boot)
> set.seed(1)
> cv.error.10 <- cv.glm(Snails, glm.fit, K=10)$delta[1]
> cv.error.10
[1] 0.03699901
```

glm.fit is FinalModel_LR.

I have performed all the techniques and arrived at this model! I am very confidant that this will perform better on any un seen data! The final model is satisfying all the four primary assumptions of a Linear Model. **Linear Relationship, Homoscedasticity, Independence, Normality**. I am sure "Linear Regression outperform all the techniques" for this dataset.

Fun Fact: I have done a lot of iterations on splitting the data and plotted how train R-Squared varies with respect to splitting ratio, starting from 50:50. Have a look at the plot. Iterating part of the code is Column = **{(50+i)*nrow/100**, **for i in 1:50}**.

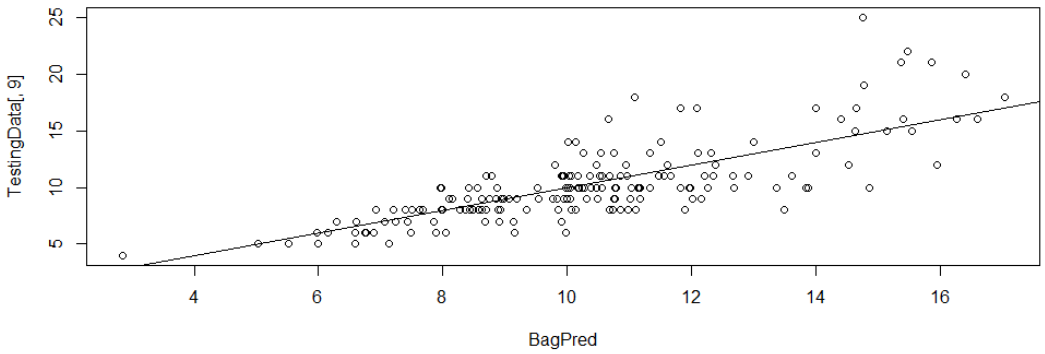


This may not be the technical cross validation but somewhat similar to it. As you can see the trend has a very minute positive slope. After plotting this I have finalized my splitting ratio- 95:5.

"Random Forest Regression"

- I have chosen Random Forest as my second model. We know that decision trees do not have same level of prediction accuracy compared to other regression techniques. Hence, we should aggregate many decision trees to improve the prediction capacity. Note: In my R file I have performed all the three ensemble methods (Bagging, Random Forest, and Boosting). RF is performing slightly better than Bagging. Hence, it is my third model. This seems quite logical. We know that Bagging predicts outputs which are highly correlated. Adding to that, our date is highly correlated. As a result, it might not perform well on new un seen data. On the other hand, **Random Forest decorrelates the data**. Because of this it will tend to perform better on a new unseen data set! Again "*Bias-Variance Tradeoff*". Coming to Boosting, it might be over fitting the data, which is already correlated, hence, it is performing slightly worse compared to Bagging and Random Forest.

- I have installed package "randomForest". I have split the data in the ratio of 95:5. Then I have performed Bagging, Random Forest and Boosting respectively. I have set a seed of 1. I have performed Bagging, using mtry = 8 all predictors. The plot of the Bagging fit is shown below.



As you can see the predictions are very accurate. Let's have a look at the code used.

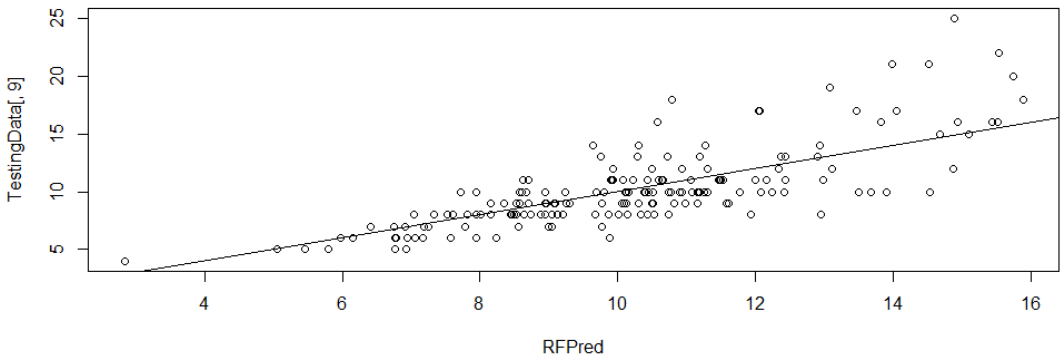
```
# Performing Bagging.
install.packages("randomForest")
library(randomForest)
set.seed(1)
trainingindex<- sample(1:nrow(Snails), 95*nrow(Snails)/100,replace=F)

TrainingData<- Snails[trainingindex,]
TestingData <- Snails[-trainingindex,]
bag.Snails <- randomForest(log(Rings) ~ Shellweight + Shuckedweight +  Diameter +
                             Type + LongestShell + Height + Wholeweight+Visceraweight ,
                           data = Snails,  mtry = 8, importance = TRUE, subset = trainingindex )
bag.Snails
BagPred <- exp(predict(bag.Snails, newdata = Snails[-trainingindex,]))
TestRSS <- sum((BagPred - TestingData[,9])^2)
TestTSS <- sum((mean(TestingData[,9])- TestingData[,9])^2)
1-TestRSS/TestTSS
# 59.7% Variability explained!!!
```

I am not worried about Training R-Squared! In the entire assignment I never mentioned that parameter. **All the percentages are corresponding to testing data, as our target is to predict**.

- Now Let's see random forest results!

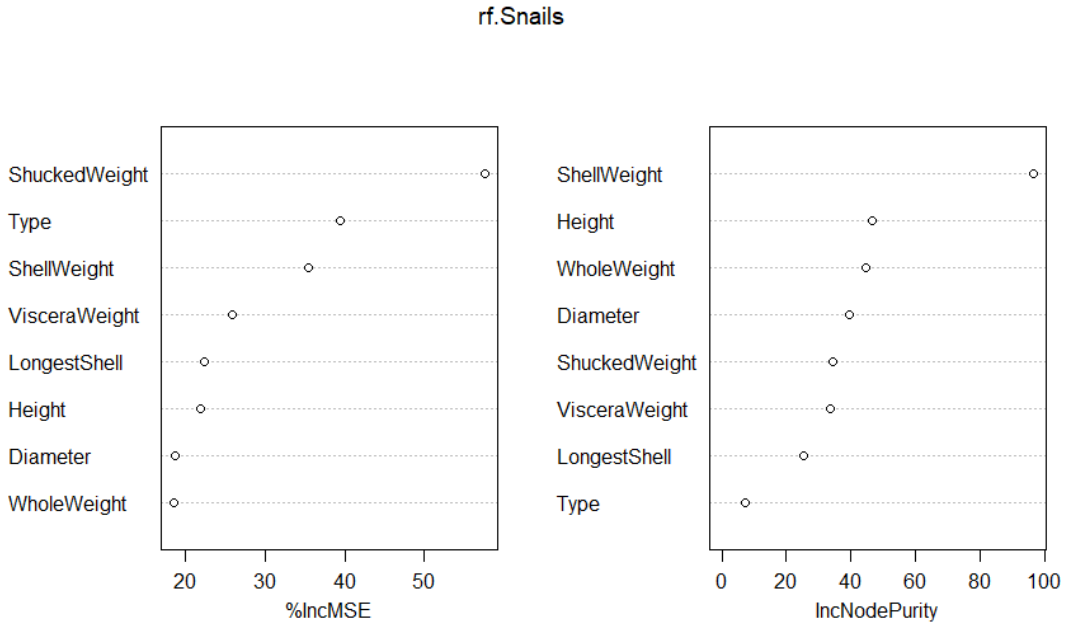
```
> # Let's Perform Random Forest
> set.seed(1)
> rf.Snails <- randomForest(log(Rings) ~ Shellweight + Shuckedweight +  Diameter +
+                             Type + LongestShell + Height + Wholeweight +Viscera weight ,
+                           data = Snails,  mtry = 2, importance = TRUE, subset = trainingindex )
> rf.Snails

Call:
 randomForest(formula = log(Rings) ~ Shellweight + Shuckedweight +      Diameter + Type + LongestShell + Heig
ht + Wholeweight + Viscera weight,      data = Snails, mtry = 2, importance = TRUE, subset = trainingindex)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 2

          Mean of squared residuals: 0.03625972
                    % Var explained: 64.86
> RFPred <- exp(predict(bag.Snails, newdata = Snails[-trainingindex,]))
> TestRSS <- sum((RFPred - TestingData[,9])^2)
> TestTSS <- sum((mean(TestingData[,9])- TestingData[,9])^2)
> 1-TestRSS/TestTSS
[1] 0.5973099
> RFPred <- exp(predict(rf.Snails, newdata = Snails[-trainingindex,]))
> TestRSS <- sum((RFPred - TestingData[,9])^2)
> TestTSS <- sum((mean(TestingData[,9])- TestingData[,9])^2)
> 1-TestRSS/TestTSS
[1] 0.5909595
```

For this I have used mtry = 2. I want to decorrelate the model as much as possible. Hence, I have picked 2 as my mtry argument.



You can see the plots are almost similar, so do the testing R-Squared. As RF is performing better and it tends to decorrelates the output, I feel it will outperform Bagging on the new unseen data. Hence, Random Forest is my second model.

- Let's look at the importance of variables. The lefthand side is corresponds to the MSE, and the righthand side is corresponds to the Node Purity, that means how confidant you can be predicting using that predictor.



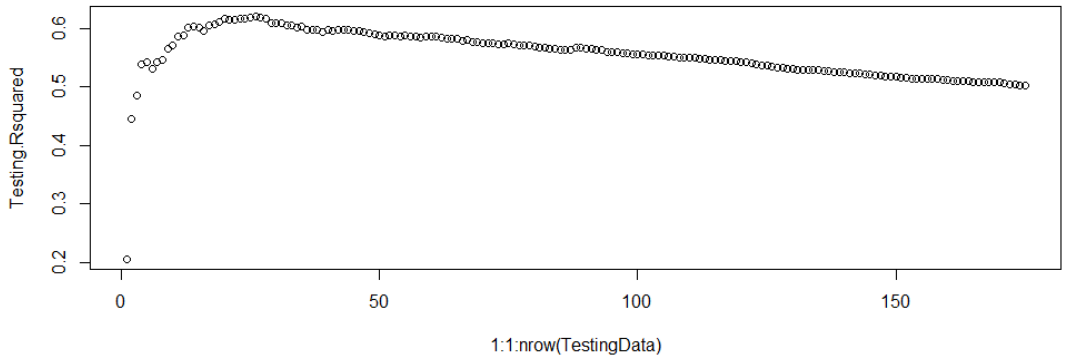
- I am not summarizing boosting, BART here. It is in my R code for the second model. The results are worse than Bagging and Random Forest. Hence, I just dropped those!

"KNN Regression"

- The third model I have chosen is K Nearest Neighbours Regression. Before diving into the detailed analysis from scratch I would like to comment on "Curse of Dimensionality". As we all know KNN Regression suffers from the curse of dimensionality. But we have to keep in mind that this curse of dimensionality is more profound if the sample size is too small compared to the predictors (higher dimension). In our case, the model is higher dimensional (p=7), however, the sample size (n) = 3500, this implies the curse is not that profound in this case. Also, we know that if n is large KNN performs better and it is very useful for predicting, which is exactly what we want.

- 1)- Let's perform the KNN Regression, without Type predictor**.** I didn't worry about training r-squared as I got K = 1 and R-Squared =1. Hence, I have directly calculated

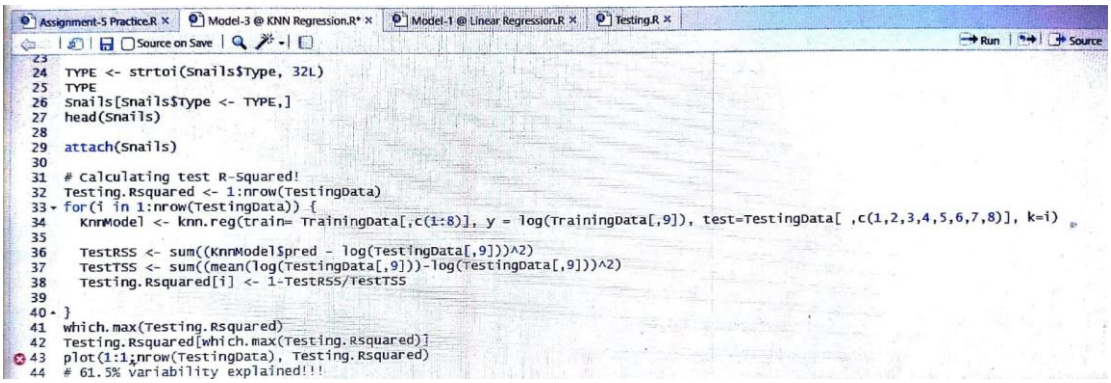
test R-Squared and found the corresponding value of K. Let's have a look at the picture. I have set a seed of 1.




```
> # Calculating test R-Squared!
> Testing.Rsquared <- 1:nrow(TestingData)
> for(i in 1:nrow(TestingData)) {
+   KnnModel <- knn.reg(train= TrainingData[,c(2:8)], y = TrainingData[,9], test=TestingData[ ,c(2,3,4,5,6,7,
8)], k=i)
+
+   TestRSS <- sum((KnnModel$pred - TestingData[,9])^2)
+   TestTSS <- sum((mean(TestingData[,9])-TestingData[,9])^2)
+   Testing.Rsquared[i] <- 1-TestRSS/TestTSS
+
+ }
> which.max(Testing.Rsquared)
[1] 26
> Testing.Rsquared[which.max(Testing.Rsquared)]
[1] 0.6210538
```

As you can see with K=26, I got the maximum testing R-Squared, which is 62.1%.

- Now I have performed the KNN again with the same seed but by including the numeric version of Type variable, using strtoi(). Have a look at the plot and result.

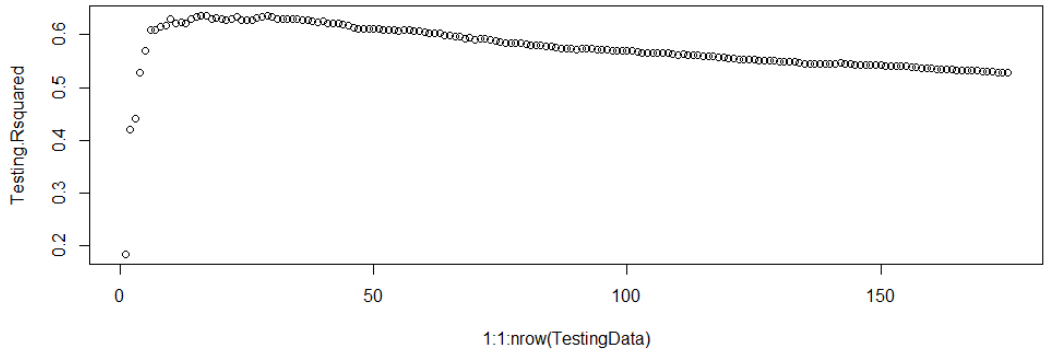


The testing R-Squared is 61.5%, the prediction accuracy has decreased. Hence, I would like to stick to the previous model as my third model. This might be wrong but I just tried converting the Type predictor into a numeric.

- What is the probability that KNN might give more weightage to bigger points, despite the fact that all are in MMGS system. Though normalizing is not required I have done it to see what might change. Indeed, there is a change, the testing R-Squared went down to 52% from 61%. Note that, I have converted the predictor "Type" into numeric

before performing KNN. I have performed KNN without using “TYPE” and the model has performed better.

- I have standardized the predictors and performed the KNN regression again with same seed. Let’s have a look at the results. You can see the inverted U shape.




```
> set.seed(1)
> Standardized.Snails <- scale(Snails[,c(-1)])
> Standardized.Test <- scale(true_test[, c(-1)])
> trainingindex<- sample(1:nrow(Standardized.Snails), 95*nrow(Standardized.Snails)/100,replace=F)
> TrainingData<- Standardized.Snails[trainingindex,]
> TestingData <- Standardized.Snails[-trainingindex,]
> Testing.Rsquared <- 1:nrow(TestingData)
> for(i in 1:nrow(TestingData)) {
+   KnnModel <- knn.reg(train= TrainingData[,c(1,2,3,4,5,6,7)], y = TrainingData[,8], test=TestingData[ ,c(1,2,
3,4,5,6,7)], k=i)
+
+   TestRSS <- sum((KnnModel$pred - TestingData[,8])^2)
+   TestTSS <- sum((mean(TestingData[,8])-TestingData[,8])^2)
+   Testing.Rsquared[i] <- 1-TestRSS/TestTSS
+
+ }
> which.max(Testing.Rsquared)
[1] 17
> Testing.Rsquared[which.max(Testing.Rsquared)]
[1] 0.6360462
> par(mfrow=c(1,1))
> plot(1:1:nrow(TestingData), Testing.Rsquared)
```

KNN is explaining 63% variability. However, Linear Regression should perform better as the true relationship is almost linear and number of predictors are more, higher dimensionality.

Conclusion: I have executed all methods and techniques that have been discussed in the class and finalized the above three models. For more information please do check the corresponding R file. I have used the Testing R-Squared as the metric to compare all the techniques. Also, I used 95:5 splitting ratio to all the models.

After performing all the techniques, Linear Regression seems to outperform all the methods for this Snails dataset. Hence, I am suggesting that as my final model. I have attached all my codes to the titles and in the question 4.

Linear Regression	*Random Forest*	*KNN Regression*
Performed better than techniques like Ridge and Lasso and Ridge	Performed better than Bagging and Boosting	Easy to compute and used for predicting
Applied log to the Rings to remove Heteroscedasticity.	Used m value of 2	Performed with and without standardizing
10-Fold CV has been used on all the predictors	No transformations & validations performed on the predictors	No transformation has been performed on predictors/rings
Seed = 1	Seed = 1	Seed = 1
Training R-Squared = 66%	Training R-Squared = 64.5%	Training R-Squared = 100%
Testing R-Squared = 64%	Testing R-Squared = 59%	Testing R-Squared = 62%
Validation Set Approach with 95:5 ratio.	Validation Set Approach with 95:5 ratio	Validation Set Approach with 95:5 ratio
Cross Validation Error of 0.0369 on the final model.	No Cross Validation required	Cross Validation not performed.
Best Regression for this dataset.	Almost a tie up with Linear Regression. Second best.	Not good compared to LR, and RF. But can be used
True relation is almost linear.	Decorrelating the Output. Hence performing better.	A trustworthy regression for prediction.

"Assignment 5 Answers"

1. The single model which I would be suggesting is Linear Model. My final model is:

   *FinalModel_LR <- lm(log(Rings) ~ log(I(ShellWeight)^6) + poly(ShuckedWeight,6) + poly(Diameter,3) + Type*poly(LongestShell,3) + Height + poly(WholeWeight,9) + poly(VisceraWeight, 4) + WholeWeight:VisceraWeight , data = TrainingData)*

   I have selected this model because the true relation is almost linear. As you can see in my analysis, the graph of FinalModel_LR has a very clear linear trend. Adding to that, I got a very good Testing R-Squared (around 64%). Finally, the final fit model has solved all the deviations of the initial model (Heteroscedasticity, Independence, Normality, and Linearity). I believe in all the methods I have done. Hence, I recommend this transformed "Linear Model" will yield accurate results, even on the un seen testing data.

2. First things first, as prediction is our primary goal, I was not at all worried about the complexity of predictors. I have started with a simple multiple regression with all the predictors. That model is clearly having Heteroscedasticity, and hugely deviating from Normality, also, the residuals are not Linear. Hence, I have addressed these points. My second model solves the first 2 points. Now for solving the Linearity I have performed 10-fold Cross Validation on each predictor to minimize test error. Before that, I have performed best subset selection and found out the number of minimum predictors required to yield accurate results. Had our target been interpreting, I would have dig deep into it and make the more interpretable. Going back to CV, after performing 10-fold Cross Validation, I got this complicated model. It looks complicated but very accurate. I suggest this model for testing on the un seen data!

3. **Linear Regression**: The testing R-Squared for this model is 64%. It explains about 64% variability on the test data. The relation between predicted values and test values is close to linear. The logarithm transformation has been applied to remove heteroscedasticity. The polynomial function has been used to remove the non-linearity of residuals. Time Series plot has been used to check independence.

   Random Forest: The testing R-Squared for this model is 59%. I have used 2 variables (m = 2) for Random Forest Regression. In total 500 trees were created. As our target is prediction, I neglected it. Had it been interpretability, I would have reduced the number of trees. According to RF, the important predictors are Shucked Weight, Type

and Shell Weight. Predictors with higher node purity are Shell Weight, Whole Weight, Diameter, and Height.

KNN Regression: Firstly, I have performed KNN without standardizing the data. I got maximum testing R-Squared as 62.1% at K = 26. The testing R-Squared is not a perfectly inverted 'U' – inverted hockey stick (NIKE logo), nevertheless it is an inverted U! Then I thought what is the probability that KNN is giving more weightage to the points which are higher. Adding to that, we are dealing with higher dimensions. Hence, I have decided to standardize the data and re-iterated the whole process. Now, the maximum testing R-Squared is 64% at K = 17. It is an inverted hockey stick shape.

4. Click the links to open the corresponding R files.

   Model-1 (Suggested Model) – Linear Regression.

   Model-2 – Random Forest Regression.

   Model-3 – KNN Regression.

5. This is the R code containing my suggested model. Set your directly and run this code then you will get testing R-Squared as an output. Testing.R.