I. Executive Summary

Purpose:
The purpose of this project is to get students familiar with the concepts covered in the class and data analysis methods to select the best model. It aims to encourage students to try out all concepts and cross validation testing in order to achieve the least possible error.

Problem Statement:
Project is divided into two phases: training and testing phase. In the training phase, a data set of 550 observations are provided and three best performing regression models should be developed. In the testing phase, another dataset with 218 observations is provided and best model from testing will be used to predict the error. Any further improvements can be done on the best model in this stage to improve the fit.

Problem Analysis:
It is decided to try out linear regression models on the train data to remove any insignificant predictors in the initial stages. Correlation plots are used to identify if any predictors are correlated or masking each other. Interaction effects are identified using trail and error and the information from the previous stages. Sequentially, all regression techniques are performed on the training dataset and cross validation results are compared to arrive at the best solution. Best model will be used for testing error rate and it is further improved by including an additional predictor.

Results of analysis:
After careful comparison of training and cross validation errors, the models are ranked in the order in which they are given below:
1) Boosting
2) Random Forest
3) Linear Regression

Boosting is the best model for both training and testing error. Adding a predictor which was not included initially further improved the model.

Group contribution:
Assignments were equally distributed among the group. Divya and Sandeep did the initial analysis of correlation, linear regression, interaction, ridge and Lasso methods. Ganesh, Vijay and Soham completed the analysis related to regression trees and cross validation of errors. After aggregating all the results, the final report is compiled.

II. Technical report of Model 1: Boosting

The first model that we considered is the Boosting regression tree on the predictors X_1, X_4, X_5, X_7, X_8 and the interaction term X_1:X_3 with gaussian distribution and 1000 trees.

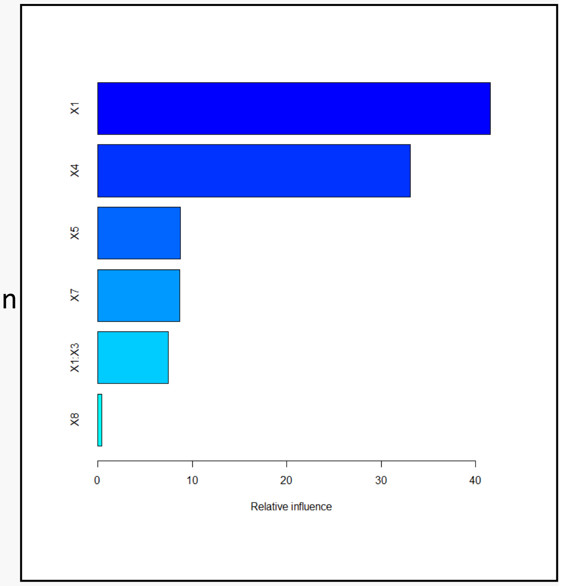
- Started out with a linear model with all eight predictors then started adding 2 pair interaction terms to improve the model.
- Eliminated X_2 since X_1 and X_2 are highly correlated, the model is affected by collinearity between these variables, the variance inflation factor (VIF) of these predictors is also very high, so one of these predictors (X_2 in this case) was dropped.
- Eliminated X_3 and X_6 since they were not significant in the linear model.
- Eliminated the 2 pair interaction terms which were not significant by using trial and run method on all ${}^8C_2$ terms.
- Performed Boosting on the final model to obtain training error and approximate test error using validation set approach.
- We did not choose any three factor and more interactions because of "Effect Hierarchy Principle". The effect hierarchy principle states that lower-order effects are more important than higher- order effects. Using this principle, we can focus on lower-order effects, say, main effects and two-factor interactions, assuming that the higher-order interactions are negligible.

Here we divide predictor space into distinct regions(branch) and use those regions to make predictions of whole data(tree). Boosting grows trees sequentially and each tree is grown using information from previously grown trees. Cross validation is performed for trees ranging from 1000 to 5000 and the after evaluation of MSE we selected 1800 as the optimal value for this model.

The main R code for the final model is shown below.

```
library(ISLR)
library(boot)
library(randomForest)
library(gbm)
library(readxl)
train <- read_excel("C:/Users/ganes/OneDrive/Desktop/MS in TAMU/Spring 2021/S
pring 2021 Courses/613/Project/train.xlsx")
View(train)
#Boosting
set.seed(1)
boost <- gbm(Y1 ~ X1 + X4 + X5 + X7 + X8 + X1:X3, data = train, distribution
```

```
= "gaussian", n.trees = 1800, interaction.depth = 4)
summary(boost)
```




```
##          var    rel.inf
## X1        X1 41.5791625
## X4        X4 33.1121128
## X5        X5  8.7203234
## X7        X7  8.6660249
## X1:X3 X1:X3  7.4657944
## X8        X8  0.4565819
set.seed(1)
yhat.boost <- predict(boost, newdata = train
## Using 1800 trees...
## Using 1800 trees...
mean((yhat.boost - train$Y1)^2)
## [1] 0.1507029
#Validation set for boosting
set.seed(1)
tr=sample(1:nrow(train),275)
te=train[-tr,]
ts= train[tr,]
set.seed(1)
bs <- gbm(Y1 ~ X1 + X4 + X5 + X7 + X8 + X1:X3, data = ts, distribution = "gau
ssian", n.trees = 1800, interaction.depth = 4)
summary(bs)
##          var   rel.inf
## X1        X1 61.590270
## X4        X4 17.003507
## X7        X7  8.707009
## X1:X3 X1:X3  6.287697
## X5        X5  4.701175
## X8        X8  1.710342
set.seed(1)
bs2 <- predict(bs, newdata = te)
## Using 1800 trees...
mean((bs2 - te$Y1)^2)
## [1] 0.2797331
```

Training Error: The training error that we obtained on this model was 0.1507029.

Approximation of test error (Cross Validation): We used the Validation Set approach to obtain an approximation of test error, the acquired error is 0.2797331.

III. Technical report of Model 2: Linear Model

The second model that we considered is the Liner model on the same set of predictors, that is, X1, X4, X5, X7, X8 and the interaction term X1:X3 but with log(Y).

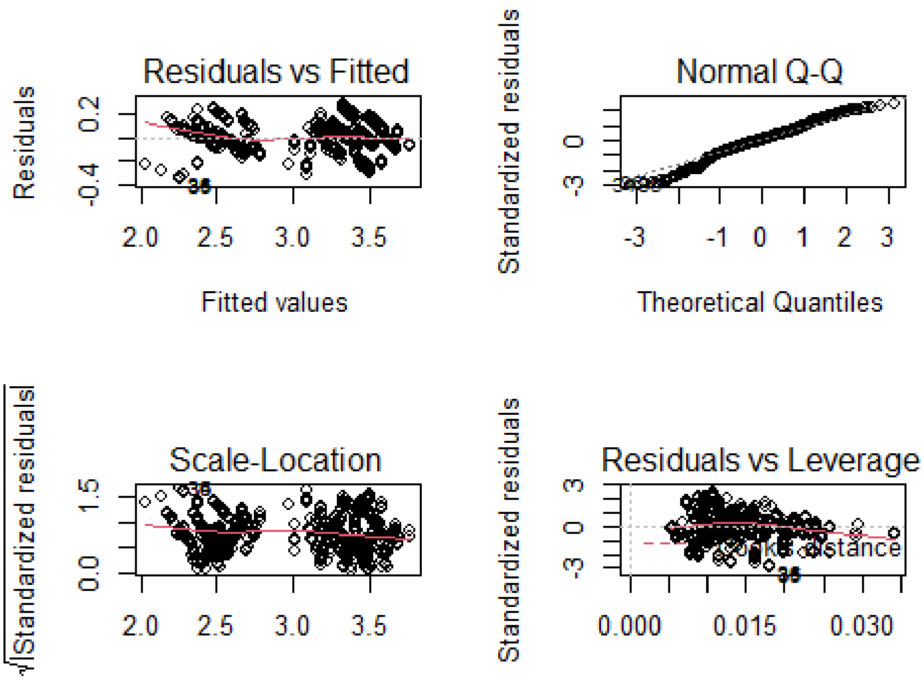
We used log transformation on response to reduce non constant variance in the residuals vs fitted plot.

Used the same process as in Model 1 to come up with the predictors and interaction terms and build the model using the Liner method (number of predictor variables = 6).

The main R code for the final model is shown below.

```
library(ISLR)
library(readxl)
set.seed(1)
tr=sample(1:nrow(train),275)
te=train[-tr,]
ts= train[tr,]

linear_model <- lm(log(Y1) ~ X1 + X4+ X1*X3 + X5  + X7 + X8, data = train)
summary(linear_model)

##
## Call:
## lm(formula = log(Y1) ~ X1 + X4 + X1 * X3 + X5 + X7 + X8, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.34949 -0.06172  0.00566  0.07253  0.28593
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.414098   1.198898  -4.516 7.74e-06 ***
## X1          -2.296109   0.694093  -3.308    0.001 **
## X4           0.022849   0.003595   6.356 4.39e-10 ***
## X3          -0.011022   0.002406  -4.582 5.72e-06 ***
## X5           0.373238   0.022336  16.710  < 2e-16 ***
## X7           1.190043   0.054364  21.890  < 2e-16 ***
## X8           0.023027   0.003301   6.975 8.96e-12 ***
## X1:X3        0.030348   0.004943   6.140 1.60e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1223 on 542 degrees of freedom
## Multiple R-squared:  0.9384, Adjusted R-squared:  0.9376
## F-statistic:  1179 on 7 and 542 DF,  p-value: < 2.2e-16
```

```
lm2 <- predict(linear_model, data = train)
mean((exp(lm2) - train$Y1)^2)

## [1] 7.163106

lm_cv <- lm(log(Y1) ~ X1 + X4+ X1*X3 + X5  + X7 + X8, data = ts)
lm_cv_test <- predict(lm_cv, newdata = te)
mean((exp(lm_cv_test) - te$Y1)^2)

## [1] 7.810343

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



Training Error: The training error that we obtained on this model was 7.163106.

Approximation of test error (Cross Validation): We used the Validation Set approach to obtain an approximation of test error, the acquired error is 7.810343.

IV. Technical report of Model 3: Random Forest

The third model that we considered is the random forest tree on the same set of predictors, that is, X1, X4, X5, X7, X8 and the interaction term X1:X3.

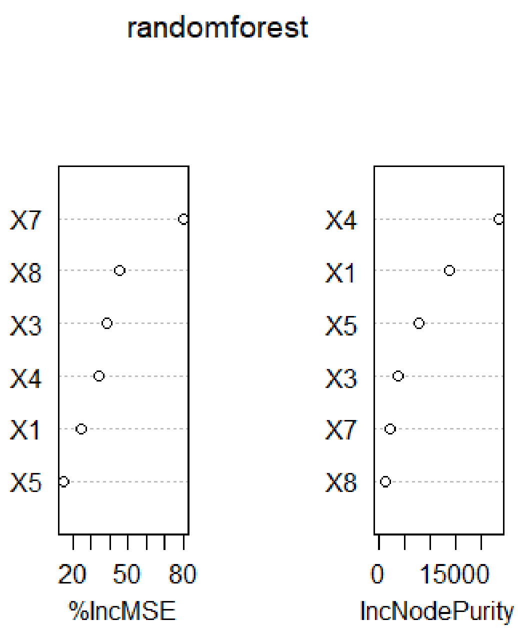
Here we divide predictor space into distinct regions(branch) and use those regions to make predictions of whole data(tree). It builds on the idea of bagging, but it provides an improvement because it de-correlates the trees by taking the random number of predictors.

Used the same process as in Model 1 to come up with the predictors and interaction terms and build the model using the random forest method. We considered 2 predictors at a time to make the random splits. Also, it is recommended to use m/3 predictors for random forest. But we are getting better results with m/2 i.e. 3. So we used mtry=3.

The main R code for the final model is shown below.

```
#random forest
set.seed(1)
randomforest <- randomForest(Y1 ~ X1 + X4 + X5 + X7 + X8 + X1:X3, data = trai
n, ntree=1000, mtry = 3, importance = TRUE)
randomforest

##
## Call:
##  randomForest(formula = Y1 ~ X1 + X4 + X5 + X7 + X8 + X1:X3, data = train,
      ntree = 1000, mtry = 3, importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.3364269
##                     % Var explained: 99.65

set.seed(1)
yhat.rf <- predict(randomforest, newdata = train)
mean((yhat.rf - train$Y1)^2)

## [1] 0.2431385

set.seed(1)
rf <- randomForest(Y1 ~ X1 + X4 + X5 + X7 + X8 + X1:X3, data = ts, mtry = 3,
ntree=1000, importance = TRUE)
rf

##
## Call:
##  randomForest(formula = Y1 ~ X1 + X4 + X5 + X7 + X8 + X1:X3, data = ts,
   mtry = 3, ntree = 1000, importance = TRUE)
```

```
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.5586441
##                     % Var explained: 99.43

set.seed(1)
yhat.rf2 <- predict(rf, newdata = te)
mean((yhat.rf2 - te$Y1)^2)

## [1] 0.5181016

varImpPlot(randomforest)
```



Training Error: The training error that we obtained on this model was 0.2431385.

Approximation of test error (Cross Validation): We used the Validation Set approach to obtain an approximation of test error, the acquired error is 0.5181016.

V. Comparison of the 3 models and the best model

Models with predictors X_1, X_4, X_5, X_7, X_8 and the interaction term X_1:X_3	Training Error using MSE	Prediction of Test Error using Cross Validation using validation set approach
Boosting	0.1507029	0.2797331
Liner Model	7.163106	7.810343
Random Forest	0.2431385	0.5181016

For boosting, we used 1800 trees and we got the training error with given data as 0.151.

To find out prediction of test error we used validation set approach and got the error as 0.2797.

For liner model with selected predictors, we used log Y transformation and received the training error of 7.13 and test error of 7.81.

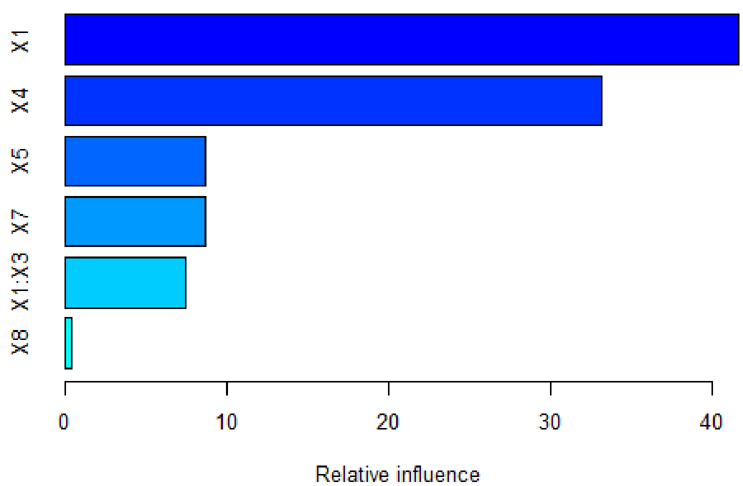
For random forest, we got percentage variance explained as 99.65% with train error of 0.24 and after doing cross validation with validation set approach, we get test error as 0.52.

So, the predicted test error is less for Boosting, so best model we receive after considering predictors X1, X4, X5, X7, X8 and the interaction term X1:X3 and applying Boosting to it.

VI. Evaluation on Test Data

After testing the model with the test data, the test error rate is obtained as 0.3134. Previously, the training error was 0.15 and the CV error rate was 0.27.

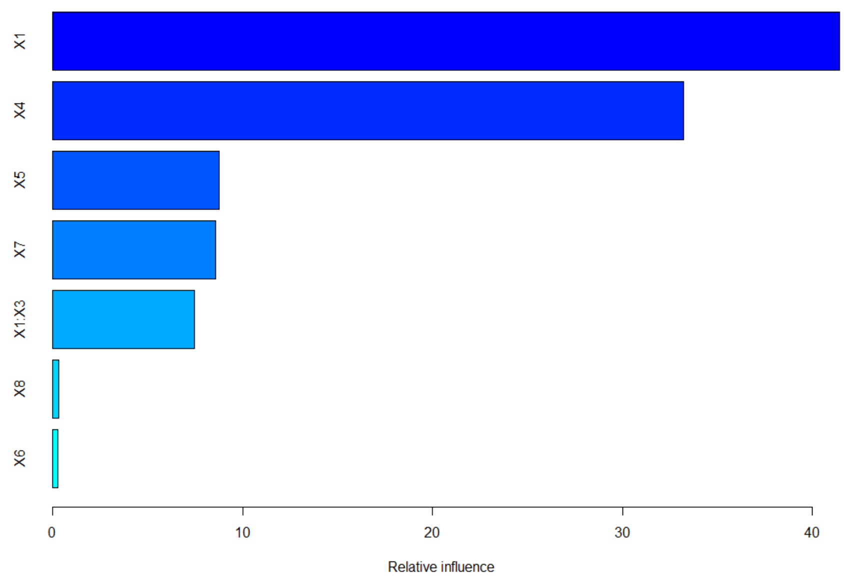
```
set.seed(1)
boost <- gbm(Y1 ~ X1 + X4 + X5 + X7 + X8 + X1:X3, data = train, distributi
on = "gaussian", n.trees = 1800, interaction.depth = 4)
summary(boost)
```




```
yhat.boost <- predict(boost, newdata = test)

## Using 1800 trees...

## Using 1800 trees...
mean((yhat.boost - test$Y1)^2)

## [1] 0.3134626
```

VII. Improvements on Best Model

```
set.seed(1)
boost <- gbm(Y1 ~X1 + X4 + X5 + X6 + X7 + X8 + X1:X3, data = train, distribut
ion = "gaussian", n.trees = 2500, interaction.depth = 4)
summary(boost)
```




```
yhat.boost <- predict(boost, newdata = test)

## Using 2500 trees...

## Using 2500 trees...
mean((yhat.boost - test$Y1)^2)

## [1] 0.2427676
```

1. In Boosting each tree is grown in sequence to fix up the past tree's mistakes. Since many numbers of trees can easily overfit, we must find the optimal number of trees that give the best error rate. After various trials using the test and training data, 2500 number of trees is selected as it gives the best error without overfitting the data.

```
cor(train)

##               X1            X2           X3           X4           X5
## X1  1.0000000000 -0.9918259561 -0.207716906 -0.869749198  0.826216198
## X2 -0.9918259561  1.0000000000  0.197543254  0.882616781 -0.858398841
## X3 -0.2077169062  0.1975432544  1.000000000 -0.286474634  0.275922328
## X4 -0.8697491981  0.8826167810 -0.286474634  1.000000000 -0.971270378
## X5  0.8262161981 -0.8583988410  0.275922328 -0.971270378  1.000000000
## X6  0.0002449909  0.0002562604 -0.007352087  0.003776097 -0.003124881
## X7  0.0853390647 -0.0873738465  0.009735304 -0.090063087  0.094652818
## X8 -0.0373030255  0.0381924601 -0.004255452  0.039367969 -0.041374211
## Y1  0.6171683367 -0.6561695421  0.453993605 -0.859014959  0.891966337
##               X6           X7           X8           Y1
## X1  0.0002449909  0.085339065 -0.037303025  0.617168337
## X2  0.0002562604 -0.087373847  0.038192460 -0.656169542
## X3 -0.0073520868  0.009735304 -0.004255452  0.453993605
## X4  0.0037760971 -0.090063087  0.039367969 -0.859014959
## X5 -0.0031248814  0.094652818 -0.041374211  0.891966337
## X6  1.0000000000 -0.007690250  0.003361527 -0.009660666
## X7 -0.0076902497  1.000000000  0.163600893  0.321708931
## X8  0.0033615271  0.163600893  1.000000000  0.044760609
## Y1 -0.0096606662  0.321708931  0.044760609  1.000000000
```

2. X6 was initially excluded as it was deemed insignificant after running a linear model, X6 was thought to be masking other variables. But after close investigation, X6 is not correlated with any variables. After including X6 on the boosting model, the test error rate was obtained to be **0.242.**
3. The test error is improved from **0.3134** in the original model to **0.242** in the improved model. The test error is improved by **23%.**