DATA605 | FINAL PROJECT

Abdellah, Ait Elmouden

Problem 1

Using R, generate a random variable X that has 10,000 random uniform numbers from 1 to N, where N can be any number of your choosing greater than or equal to 6. Then generate a random variable Y that has 10,000 random normal numbers with a mean of $\mu = \sigma = (N + 1)/2$

Part 1

Solution

Let's generate a random variable X

```
# set seed value
set.seed(123)
n <- 9
X <- runif(10000, min = 1, max = n)
head(X)
```

```
## [1] 3.300620 7.306441 4.271815 8.064139 8.523738 1.364452
```

Generate a random variable Y

```
# mean
mu <- (n+1)/2
std <- mu
Y <- rnorm(10000 , mean = mu, std)
head(Y)
```

```
## [1]  2.5291306 10.6379673 -0.7347477 12.4050930  9.5809561  6.6756551
```

Probability.

Calculate as a minimum the below probabilities a through c. Assume the small letter "x" is estimated as the median of the X variable, and the small letter "y" is estimated as the 1st quartile of the Y variable. Interpret the meaning of all probabilities.

```
x = median(X)
y = quantile(Y,0.25)

paste0('The median of X = ',mean(X[X>y]>x))
```

```
## [1] "The median of X = 0.546567555749891"
```

```
paste0('The estimated 1st quartile of Y = ',y)
```

```
## [1] "The estimated 1st quartile of Y = 1.67320851731387"
```

a. $P(X > x|X > y)$

Probability that X is greater than the median, given that X is greater than the 1st quartile of Y.

$$P(X > x|X > y) = \frac{P(P(X > x) \bigcap P(X > y))}{P(X > y)}$$

```
denominator <- length(X[X>y]) / n
nominator <- length(X[X>max(x,y)]) / n
probability1 <- nominator/denominator
paste0('The probability P(X>x | X>y)= ', probability1)
```

```
## [1] "The probability P(X>x | X>y)= 0.546567555749891"
```

b. $P(X > x, Y > y)$

Probability that X is greater than the median and that Y is greater than the 1st quartile.

```
probability2 <- sum(X>x & Y>y)/length(X)
paste0('The probability P(X>x, Y>y) = ', probability2)
```

```
## [1] "The probability P(X>x, Y>y) = 0.3756"
```

c. $P(X < x|X > y)$

Probability that X is less than the median, given that X is greater than the 1st quartile of Y.

$$P(X < x|X > y) = P(P(X < x)|P(X > y)) = \frac{P(P(X < x) \bigcap P(X > y))}{P(X > y)}$$

```
nominator <- length(X[(X < x) & (X > y)]) / n
denominator <- length(X[X>y]) / n
probability3 <- nominator/denominator
paste0('The probability P(X<x | X>y) = ', probability3)
```

```
## [1] "The probability P(X<x | X>y) = 0.453432444250109"
```

Part 2

Investigate whether $P(X > x \text{ and } Y > y) = P(X > x)P(Y > y)$ by building a table and evaluating the marginal and joint probabilities.

Answer

```
data <- data.frame(X = rep(X, n),Y = rep(Y, n))

#(X>x and Y > y)
a <- nrow(data[data$X>x & data$Y>y,]) / n

#(X>x and Y <= y)
b <- nrow(data[data$X>x & data$Y<=y,]) / n

#(X<=x and Y>y)
c <- nrow(data[data$X<=x & data$Y>y,]) / n

#(X<=x and Y<=y)
d <- nrow(data[data$X<=x & data$Y<=y,]) / n
rm(data)
```

Table of Counts

	$Y > y$	$Y \leq y$	Total
$X > x$	3723	1277	5000
$X \leq x$	3731	1269	5000
Total	7454	2546	10000

Joint Probabilities

	$Y > y$	$Y \leq y$
$X > x$	$\frac{3723}{10000} = 0.3723$	$\frac{1277}{10000} = 0.127$ \$
$X \leq x$	$\frac{3731}{10000} = 0.37$	$\frac{1269}{10000} = 0.127$

Marginal Probabilities

$$P_X = \begin{pmatrix} \frac{5000}{10000} & \frac{5000}{10000} \end{pmatrix}$$
$$P_Y = \begin{pmatrix} \frac{7454}{10000} & \frac{2546}{10000} \end{pmatrix}$$

$$P(X > x \text{ and } Y > y) = \frac{3733}{10000} = 0.3733$$
$$P(X > x)P(Y > y) = \frac{5000}{10000} \cdot \frac{7500}{10000} = 0.375$$

Part 3

Check to see if independence holds by using Fisher's Exact Test and the Chi Square Test. What is the difference between the two? Which is most appropriate?

Hypotheses: H0: (null) the variables are independent H1: the variables are dependent

```
contingency <- matrix(c(sum(X>x & Y>y), sum(X>x & Y<=y), sum(X<=x & Y>y), sum(X<=x & Y<=y)), nrow = 2)
```

Fisher's Exact Test:

```
contingency <- matrix(c(sum(X>x & Y>y), sum(X>x & Y<=y), sum(X<=x & Y>y), sum(X<=x & Y<=y)), nrow = 2)
fisher.result <- fisher.test(contingency)
fisher.result
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  contingency
## p-value = 0.7995
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.9242273 1.1100187
## sample estimates:
## odds ratio
##   1.012883
```

Chi-Squared Test

```
chisquare.result <- chisq.test(contingency)
chisquare.result
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contingency
## X-squared = 0.064533, df = 1, p-value = 0.7995
```

The chi square test for independence compares two variables to see if they are related. A small chi-square p-value demonstrates that there is a significant association between the two variables.

In both cases the p-value is much greater than a reasonable threshold of 0.05. So we do not reject the null hypothesis of independence and conclude that they are indeed independent.

Fisher's exact test tests the null Hypothesis of independence and used when then the sample size is small.

Chi-squared test is used when the sample size is large.

We have a large enough sample size, so chi-square is more appropriate in this case.

Problem 2

Objective

The goal of this problem is to discuss the major factors that affect housing price and make precise predictions for it. to do this we'll use the house prices predicting problem on [kaaggle[(https://www.kaggle.com/c/house-prices-advanced-regression-techniques) . I want you to do the following.

We want to answer following two questions:

1. What are the important features that affect the house price?
2. try to build a model to predict the house price?

Data Description & Pre-Processing

Data was downloaded from the website kaggle and saved in a local folder. The data is available in to csv file a training dataset that has a total of 1460 observations with 81 variables and testing dataset.

```
library(readr)
library(dplyr)
train <- read_csv('train.csv')
test <- read_csv('test.csv')
```

To Clean our data We'll create a function that will help us to clean the train data and later the test data to ensure consistency in pre-processing.

```
clean <- function (df){
    #------------------
    # Numerical Values
    #------------------

    # First, we deal with numerical values.
    # We'll assume that `NAs` in these variables means 0

    df$LotFrontage[is.na(df$LotFrontage)] <- 0
    df$MasVnrArea[is.na(df$MasVnrArea)] <- 0
    df$BsmtFinSF1[is.na(df$BsmtFinSF1)] <- 0
    df$BsmtFinSF2[is.na(df$BsmtFinSF2)] <- 0
    df$BsmtUnfSF[is.na(df$BsmtUnfSF)] <- 0
    df$TotalBsmtSF[is.na(df$TotalBsmtSF)] <- 0
    df$BsmtFullBath[is.na(df$BsmtFullBath)] <- 0
    df$BsmtHalfBath[is.na(df$BsmtHalfBath)] <- 0
    df$GarageCars[is.na(df$GarageCars)] <- 0
    df$GarageArea[is.na(df$GarageArea)] <- 0

    # One special case is the variable "GarageYrBlt".
    # We can assume that the year that the garage was built is the same than
    # when the house itself was built.

    df$GarageYrBlt[is.na(df$GarageYrBlt)] <- df$YearBuilt[is.na(df$GarageYrBlt)]

    # Fix a typo on GarageYrBlt from 2207 to 2007
```

```r
df$GarageYrBlt[df$GarageYrBlt==2207] <- 2007

#--------------------
# Categorical Values
# --------------------

# `NAs` in this dataset might be due to:
# 1) Missing data.  2) Empty values for this feature (for instance, a house does not have Garage).

# Let's first address NAs, that represent missing values impute them with the most common
# value for this   feature.

df$KitchenQual[is.na(df$KitchenQual)] <- names(sort(-table(df$KitchenQual)))[1]
df$MSZoning[is.na(df$MSZoning)] <- names(sort(-table(df$MSZoning)))[1]
df$SaleType[is.na(df$SaleType)] <- names(sort(-table(df$SaleType)))[1]
df$Exterior1st[is.na(df$Exterior1st)] <- names(sort(-table(df$Exterior1st)))[1]
df$Exterior2nd[is.na(df$Exterior2nd)] <- names(sort(-table(df$Exterior2nd)))[1]
df$Functional[is.na(df$Functional)] <- names(sort(-table(df$Functional)))[1]

# For empty values, we will change the `NA` value to a new value - 'No'.

df$Alley = factor(df$Alley, levels=c(levels(df$Alley), "No"))
df$Alley[is.na(df$Alley)] = "No"

# Bsmt : NA for basement features is "no basement"
df$BsmtQual = factor(df$BsmtQual, levels=c(levels(df$BsmtQual), "No"))
df$BsmtQual[is.na(df$BsmtQual)] = "No"

df$BsmtCond = factor(df$BsmtCond, levels=c(levels(df$BsmtCond), "No"))
df$BsmtCond[is.na(df$BsmtCond)] = "No"

df$BsmtExposure[is.na(df$BsmtExposure)] = "No"

df$BsmtFinType1 = factor(df$BsmtFinType1, levels=c(levels(df$BsmtFinType1), "No"))
df$BsmtFinType1[is.na(df$BsmtFinType1)] = "No"

df$BsmtFinType2 = factor(df$BsmtFinType2, levels=c(levels(df$BsmtFinType2), "No"))
df$BsmtFinType2[is.na(df$BsmtFinType2)] = "No"

# Fence : NA means "no fence"
df$Fence = factor(df$Fence, levels=c(levels(df$Fence), "No"))
df$Fence[is.na(df$Fence)] = "No"

# FireplaceQu : NA means "no fireplace"
df$FireplaceQu = factor(df$FireplaceQu, levels=c(levels(df$FireplaceQu), "No"))
df$FireplaceQu[is.na(df$FireplaceQu)] = "No"

# Garage : NA for garage features is "no garage"
df$GarageType = factor(df$GarageType, levels=c(levels(df$GarageType), "No"))
df$GarageType[is.na(df$GarageType)] = "No"

df$GarageFinish = factor(df$GarageFinish, levels=c(levels(df$GarageFinish), "No"))
df$GarageFinish[is.na(df$GarageFinish)] = "No"
```

```
    df$GarageQual = factor(df$GarageQual, levels=c(levels(df$GarageQual), "No"))
    df$GarageQual[is.na(df$GarageQual)] = "No"

    df$GarageCond = factor(df$GarageCond, levels=c(levels(df$GarageCond), "No"))
    df$GarageCond[is.na(df$GarageCond)] = "No"

    # MasVnrType : NA most likely means no veneer
    df$MasVnrType = factor(df$MasVnrType, levels=c(levels(df$MasVnrType), "No"))
    df$MasVnrType[is.na(df$MasVnrType)] = "No"

    # MiscFeature : NA = "no misc feature"
    df$MiscFeature = factor(df$MiscFeature, levels=c(levels(df$MiscFeature), "No"))
    df$MiscFeature[is.na(df$MiscFeature)] = "No"

    # PoolQC : data description says NA means "no pool"
    df$PoolQC = factor(df$PoolQC, levels=c(levels(df$PoolQC), "No"))
    df$PoolQC[is.na(df$PoolQC)] = "No"

    # Electrical : NA means "UNK"
    df$Electrical = factor(df$Electrical, levels=c(levels(df$Electrical), "UNK"))
    df$Electrical[is.na(df$Electrical)] = "UNK"

    # GarageYrBlt: It seems reasonable that most houses would build a garage when
    # the house itself was built.
    idx <- which(is.na(df$GarageYrBlt))
    df[idx, 'GarageYrBlt'] <- df[idx, 'YearBuilt']

    # Remove meaningless features and incomplete cases.
    df$Utilities <- NULL
    df$Id <- NULL

    # There are some categories which are clearly a ranking, we'll change them to Numerical
    df$ExterQual<- recode(df$ExterQual,"None"=0,"Po"=1,"Fa"=2,"TA"=3,"Gd"=4,"Ex"=18)

    return(df)
}
```

Clean the train dataset

```
train_clean <- clean(train)
```

Let's Check if our data is clean:

```
na.cols <- which(colSums(is.na(train_clean)) > 0)
paste('There are now', length(na.cols), 'columns with missing values')
```

```
## [1] "There are now 0 columns with missing values"
```

Descriptive and Inferential Statistics

- SalePrice -
- LotArea - Lot size in square feet
- LotFrontage: Linear feet of street connected to property

```
dim(train_clean)
```

```
## [1] 1460   79
```

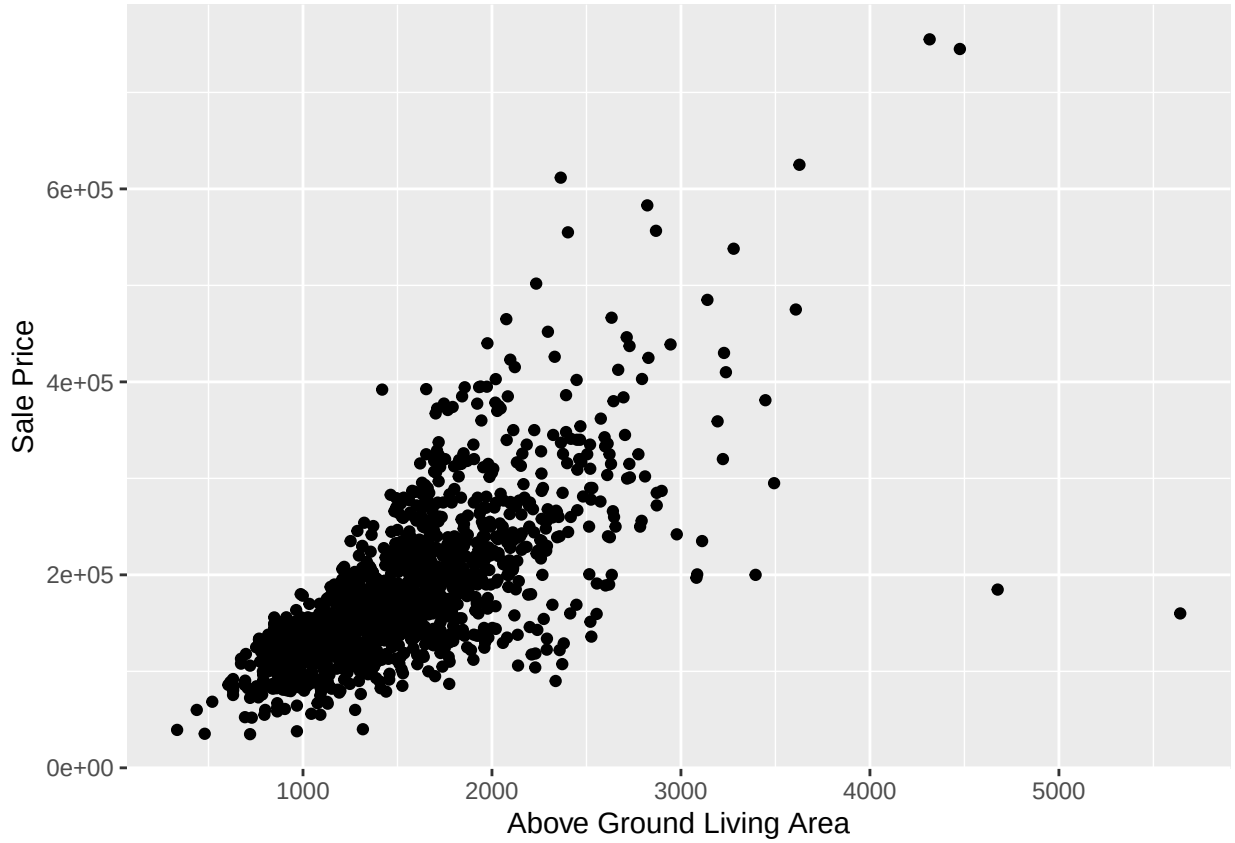
```
cols <- c("SalePrice","LotArea","LotFrontage", "OverallQual", "YearBuilt", "YearRemodAdd")
```

```
summary(train_clean[cols])
```

```
##    SalePrice         LotArea        LotFrontage      OverallQual
##  Min.   : 34900   Min.   :  1300   Min.   :  0.00   Min.   : 1.000
##  1st Qu.:129975   1st Qu.:  7554   1st Qu.: 42.00   1st Qu.: 5.000
##  Median :163000   Median :  9478   Median : 63.00   Median : 6.000
##  Mean   :180921   Mean   : 10517   Mean   : 57.62   Mean   : 6.099
##  3rd Qu.:214000   3rd Qu.: 11602   3rd Qu.: 79.00   3rd Qu.: 7.000
##  Max.   :755000   Max.   :215245   Max.   :313.00   Max.   :10.000
##    YearBuilt     YearRemodAdd
##  Min.   :1872   Min.   :1950
##  1st Qu.:1954   1st Qu.:1967
##  Median :1973   Median :1994
##  Mean   :1971   Mean   :1985
##  3rd Qu.:2000   3rd Qu.:2004
##  Max.   :2010   Max.   :2010
```

Area vs Price

```
library(ggplot2)
library(gridExtra)

# Basic scatter plot
train_clean %>% ggplot(aes(x = GrLivArea, y = SalePrice)) +
  geom_point() +
  xlab("Above Ground Living Area") +
  ylab("Sale Price")
```

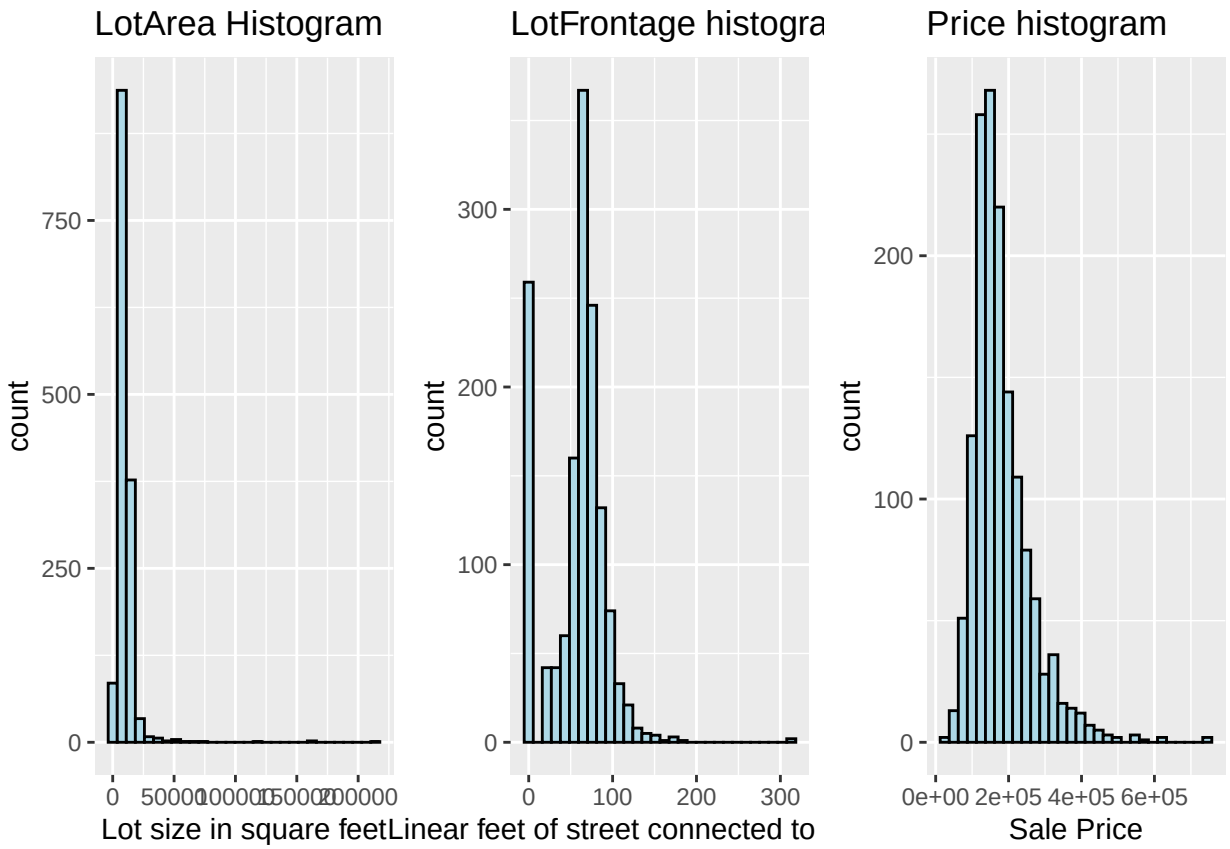


There are two outliers which has high areas but low sale price. When fitting the models, we delete these two outliers in the training data.

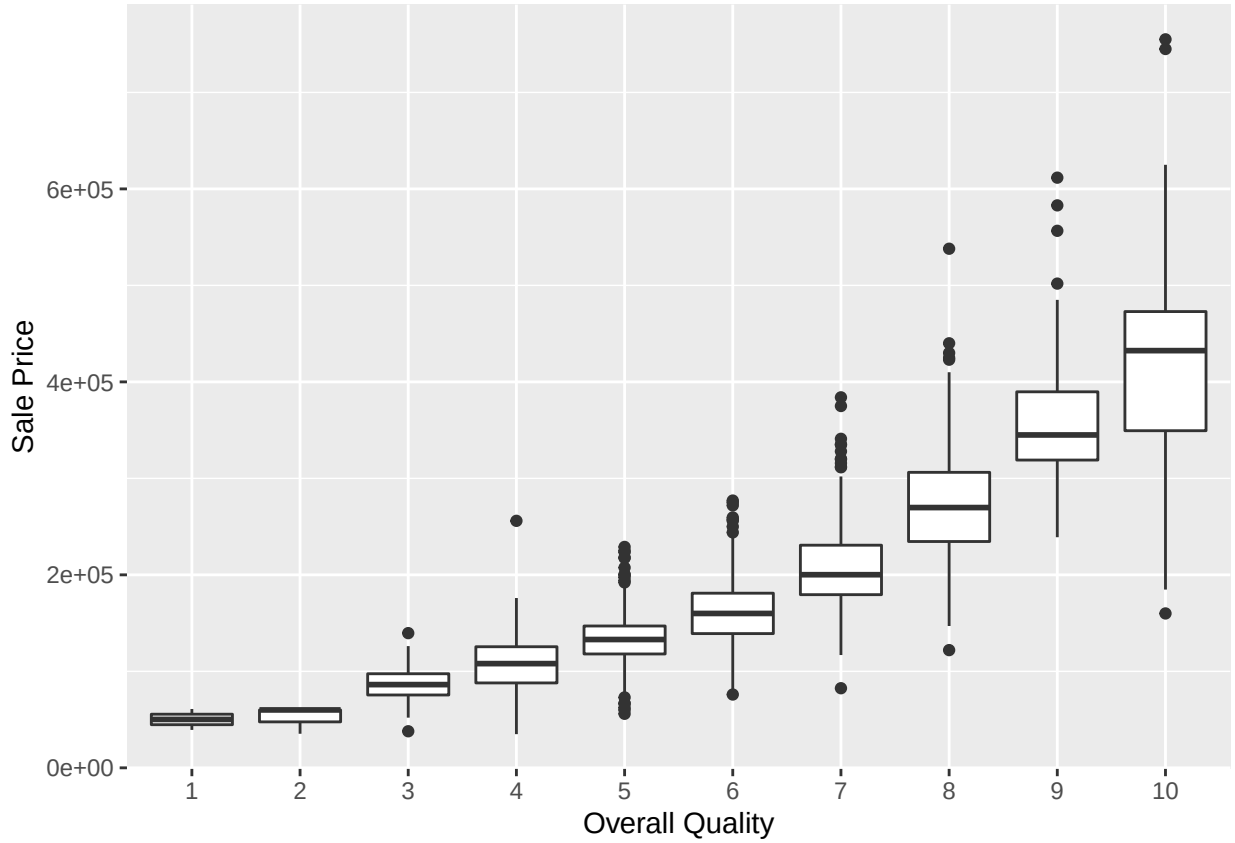
Overall Quality vs Price

Histogram

```
# Basic scatter plot
p1 <- ggplot(train_clean,
             aes(x=train_clean$LotArea)) +
                geom_histogram(color="black",
                fill="lightblue") +

      ggtitle("LotArea Histogram") +

      xlab("Lot size in square feet")

p2 <- ggplot(train_clean, aes(x=train_clean$LotFrontage))+
            geom_histogram(color="black", fill="lightblue")+
            ggtitle("LotFrontage histogram") +
            xlab("Linear feet of street connected to property")

p3 <- ggplot(train_clean, aes(x=train_clean$SalePrice))+
              geom_histogram(color="black", fill="lightblue")+
      ggtitle("Price histogram") +
      xlab("Sale Price")

grid.arrange(p1, p2,p3, nrow = 1)
```



Overall Quality vs Price



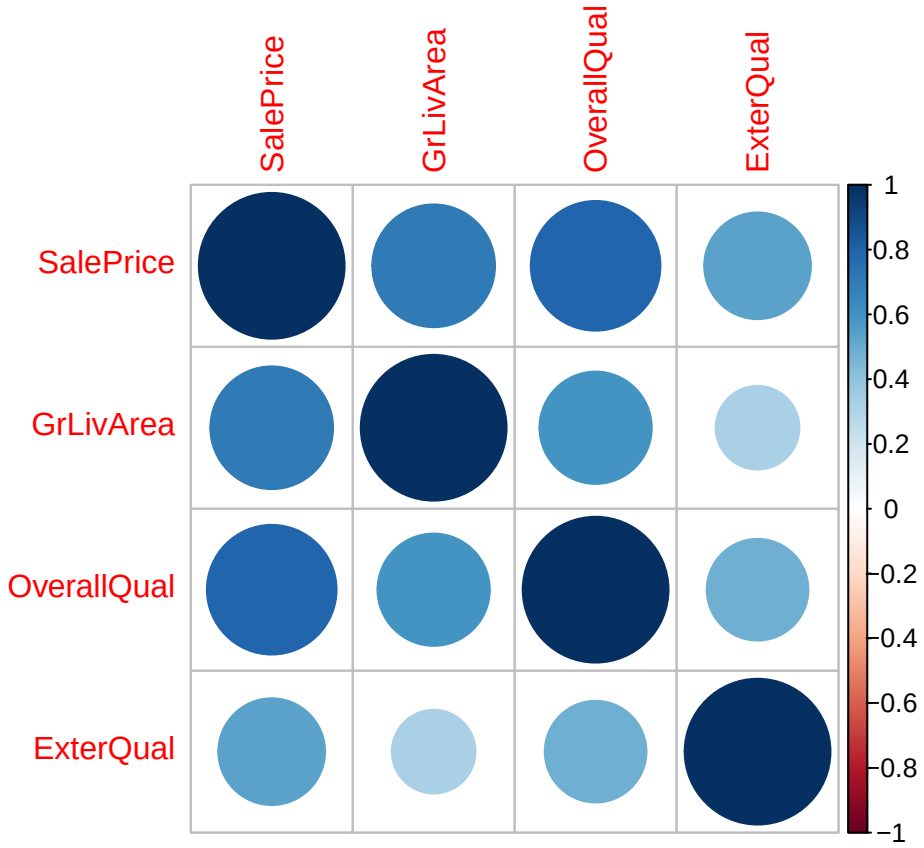
Overall quality of the house is better indicate that the house price is higher

Correlation Matrix

```
corr_matrix <- cor(train_clean[,c("SalePrice", "GrLivArea", "OverallQual", "ExterQual")])
corr_matrix
```

```
##             SalePrice GrLivArea OverallQual ExterQual
## SalePrice   1.0000000 0.7086245   0.7909816 0.5357513
## GrLivArea   0.7086245 1.0000000   0.5930074 0.3296945
## OverallQual 0.7909816 0.5930074   1.0000000 0.4866375
## ExterQual   0.5357513 0.3296945   0.4866375 1.0000000
```

Performing Pairwise correlation Testing indicated that correlation between each pairwise set of variables is not 0, but the correlation varies between variables. From the above matrice, we can easily learn that saleprice have strong positive correlation with some factors such as overallquality(0.79), GrLiveArea(Above grade (ground) living area square feet)(0.70), ExterQual(the quality of the material on the exterior)(0.53). This correlogram can give some basic information about the relationship between variables and saleprice, which we need to predict in the future.



Hypothesis Testing

Let's test the hypothesis that the correlations between each pairwise set of variables is 0 and provide an 80% confidence interval.

SalePrice vs GrLivArea

- H_0: The correlation between GrLivArea and SalePrice is 0
- H_a: The correlation between GrLivArea and SalePrice is other than 0

```
SalePricevsGrLivArea <- cor.test(train_clean$SalePrice, train_clean$GrLivArea,
                                 method="pearson", conf.level=0.8)
SalePricevsGrLivArea
```

```
##
##  Pearson's product-moment correlation
##
## data:  train_clean$SalePrice and train_clean$GrLivArea
## t = 38.348, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.6915087 0.7249450
## sample estimates:
##       cor
## 0.7086245
```

SalePrice vs OverallQual

- H_0: The correlation between OverallQual and SalePrice is 0
- H_a: The correlation between OverallQual and SalePrice is other than 0

```
SalePricevsOverallQual <- cor.test(train_clean$SalePrice, train_clean$OverallQual,
                                   method="pearson", conf.level=0.8)
SalePricevsOverallQual
```

```
##
##  Pearson's product-moment correlation
##
## data:  train_clean$SalePrice and train_clean$OverallQual
## t = 49.364, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.7780752 0.8032204
## sample estimates:
##       cor
## 0.7909816
```

SalePrice vs ExterQual

- H_0: The correlation between ExterQual and SalePrice is 0
- H_a: The correlation between ExterQual and SalePrice is other than 0

```
SalePricevsExterQual <- cor.test(train_clean$SalePrice, train_clean$ExterQual,
                                 method="pearson", conf.level=0.8)
SalePricevsExterQual
```

```
##
##  Pearson's product-moment correlation
##
## data:  train_clean$SalePrice and train_clean$ExterQual
## t = 24.227, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.5113847 0.5592571
## sample estimates:
##       cor
## 0.5357513
```

Test summary

	p-value	Correlation Coefficient	t Statistic	80% Confidence Interval
GrLivArea	< 2.2e-16	0.7086245	38.348	[0.6915087, 0.7249450]
OverallQual	< 2.2e-16	0.7909816	49.364	[0.7780752, 0.8032204]
ExterQual	< 2.2e-16	0.5357513	24.227	[0.5113847, 0.5592571]

the p values is so smaller than 0.05 at 5% level of significance, we reject the null hypothesis and conclude that the correlation between the other variables and SalePrice is other than 0.

Family wise error

I don't think I should worry about familywise error. The data set is large with 1460 observations. Moreover, each of the 3 pair sets have demonstrated a high t-statistic and very low p-value indicating that we have enough evidence to reject the null hypothesis.

Linear Algebra and Correlation.

Invert your correlation matrix from above. (This is known as the precision matrix and contains variance inflation factors on the diagonal.) Multiply the correlation matrix by the precision matrix, and then multiply the precision matrix by the correlation matrix. Conduct LU decomposition on the matrix.

Find inverse

```
precision_matrix <- solve(corr_matrix)
precision_matrix
```

```
##              SalePrice  GrLivArea OverallQual  ExterQual
## SalePrice    3.8220786 -1.3690574  -1.8796345 -0.6816123
## GrLivArea   -1.3690574  2.0380752  -0.2039330  0.1607736
## OverallQual -1.8796345 -0.2039330   2.7318691 -0.2551777
## ExterQual   -0.6816123  0.1607736  -0.2551777  1.4363476
```

Multiply the correlation matrix by the precision matrix

```
correlation_precision <- corr_matrix %*% precision_matrix
correlation_precision
```

```
##                 SalePrice     GrLivArea  OverallQual     ExterQual
## SalePrice    1.000000e+00  2.775558e-17 2.220446e-16  0.000000e+00
## GrLivArea   -8.326673e-17  1.000000e+00 3.191891e-16 -5.551115e-17
## OverallQual -5.551115e-17 -4.163336e-17 1.000000e+00  0.000000e+00
## ExterQual   -2.220446e-16 -2.775558e-17 3.885781e-16  1.000000e+00
```

Multiply the precision matrix by the correlation matrix

```
prec_cor <- precision_matrix %*% corr_matrix
prec_cor
```

```
##                 SalePrice     GrLivArea   OverallQual     ExterQual
## SalePrice    1.000000e+00  4.440892e-16  4.996004e-16  4.440892e-16
## GrLivArea   -1.804112e-16  1.000000e+00 -2.498002e-16 -1.665335e-16
## OverallQual -2.220446e-16 -1.249001e-16  1.000000e+00  1.665335e-16
## ExterQual   -2.220446e-16 -1.665335e-16 -1.110223e-16  1.000000e+00
```

LU Decomposistion

```
## $L
##              SalePrice   GrLivArea OverallQual ExterQual
## SalePrice    1.0000000  0.00000000   0.0000000         0
## GrLivArea   -0.3581971  1.00000000   0.0000000         0
## OverallQual -0.4917833 -0.56679092   1.0000000         0
```

```
## ExterQual   -0.1783355 -0.05387276  -0.4866375         1
##
## $U
##              SalePrice GrLivArea OverallQual   ExterQual
## SalePrice    3.822079 -1.369057  -1.8796345 -0.68161234
## GrLivArea    0.000000  1.547683  -0.8772126 -0.08337795
## OverallQual  0.000000  0.000000   1.3103001 -0.63764113
## ExterQual    0.000000  0.000000   0.0000000  1.00000000
```

Calculus-Based Probability & Statistics

Many times, it makes sense to fit a closed form distribution to data. Select a variable in the Kaggle.com training dataset that is skewed to the right, shift it so that the minimum value is absolutely above zero if necessary. Then load the MASS package and run fitdistr to fit an exponential probability density function. See https://stat.ethz.ch/R-manual/R-devel/library/MASS/html/fitdistr.html. Find the optimal value of for this distribution, and then take 1000 samples from this exponential distribution using this value (e.g., rexp(1000,)). Plot a histogram and compare it with a histogram of your original variable. Using the exponential pdf, find the 5th and 95th percentiles using the cumulative distribution function (CDF). Also generate a 95% confidence interval from the empirical data, assuming normality. Finally, provide the empirical 5th percentile and 95th percentile of the data. Discuss.

We have selected the variable LotArea, because it is skewed to the right.

```
summary(train_clean$LotArea)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1300    7554    9478   10517   11602  215245
```

```
hist(train_clean$LotArea)
```

Optimal value of exponential for this distribution

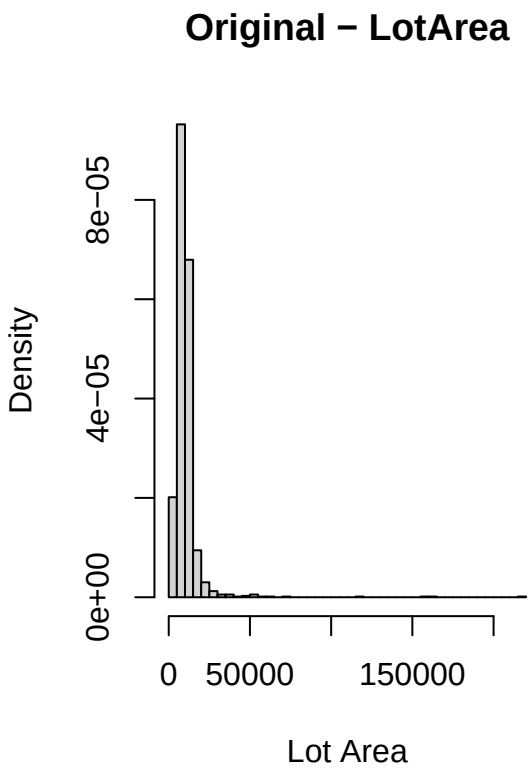
```
##        rate
##   9.508570e-05
##  (2.488507e-06)
```

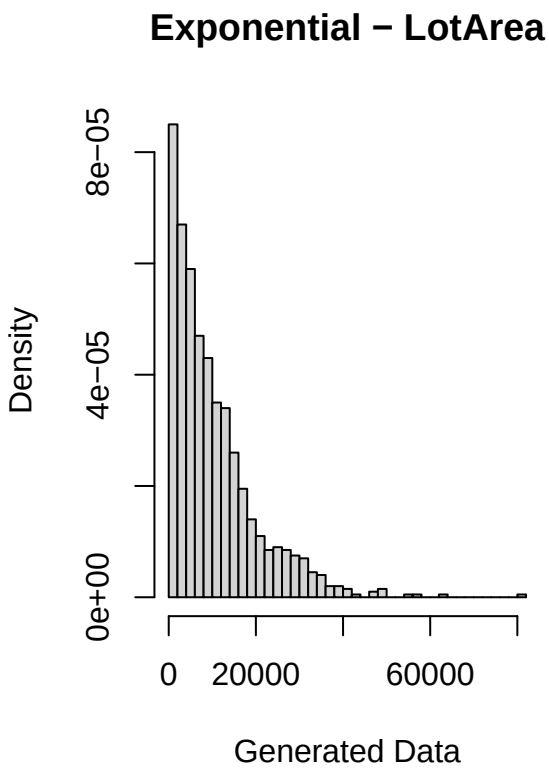
```
# optimal value of lambda
fit_dist$estimate
```

```
##        rate
## 9.50857e-05
```

1000 samples from this exponential distribution using this value

```
values <- rexp(1000, rate = fit_dist$estimate)
par(mfrow=c(1,2))
# Actual vs simulated distribution
hist(train_clean$LotArea, breaks=40, prob=TRUE, xlab="Lot Area",
     main="Original - LotArea")
hist(values, breaks=40, prob=TRUE, xlab="Generated Data",
     main="Exponential - LotArea")
```





From the two plots we can see that our Lot Area approximately fits a exponential distribution.

5th and 95th percentiles using the cumulative distribution function (CDF)

```
five_95 <- ecdf(values)
values[five_95(values)==0.05]
```

```
## [1] 580.0635
```

```
values[five_95(values)==0.95]
```

```
## [1] 30457.87
```

So: 5% is 451.8907 and 95% is 34918.55

95% confidence interval from the empirical data

```
t.test(values)$conf.int
```

```
## [1]  9783.818 11002.469
## attr(,"conf.level")
## [1] 0.95
```

Empirical 5th percentile and 95th percentile of the data

```
t.test(train$LotArea)$conf.int
```

```
## [1] 10004.42 11029.24
## attr(,"conf.level")
## [1] 0.95
```

Modeling

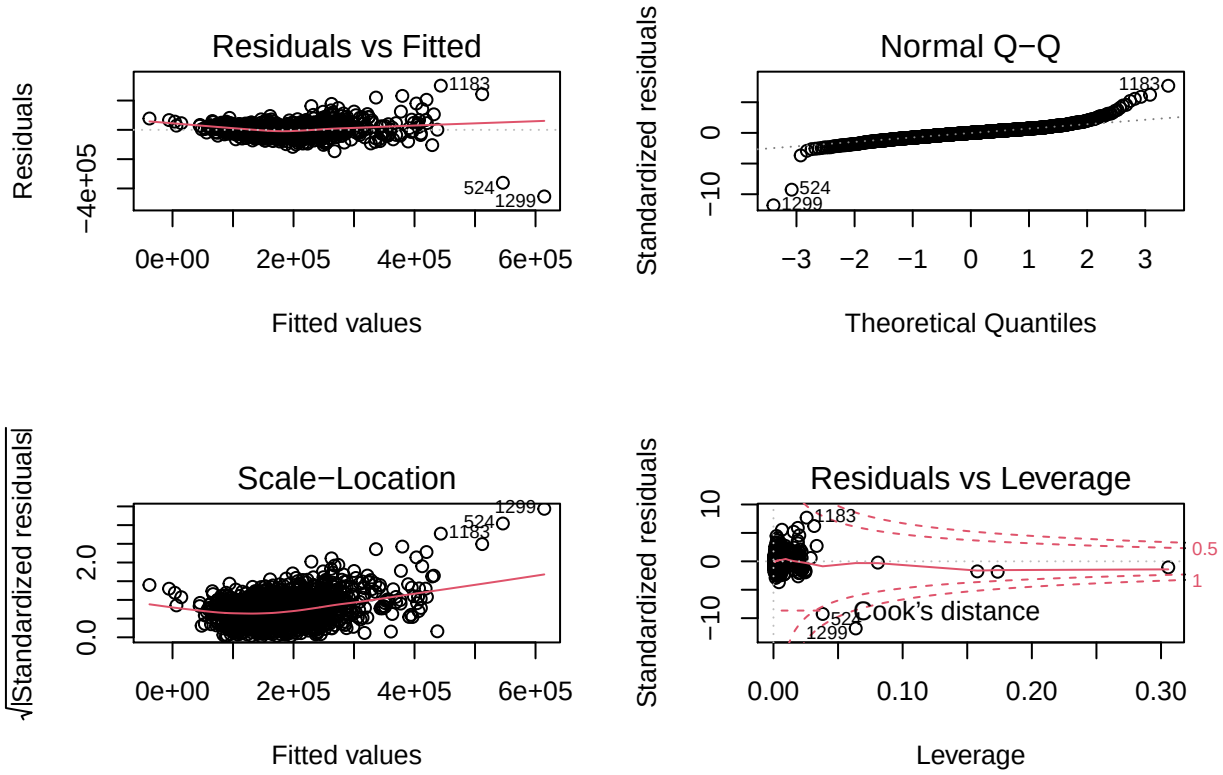
Build some type of multiple regression model and submit your model to the competition board. Provide your complete model summary and results with analysis. Report your Kaggle.com user name and score.

We'll consider the following variable to build the model: LotArea, GrLivArea, OverallQual, ExterQual

```
lm <- lm(SalePrice ~ OverallQual + ExterQual + LotArea + GrLivArea, data = train_clean)
summary(lm)
```

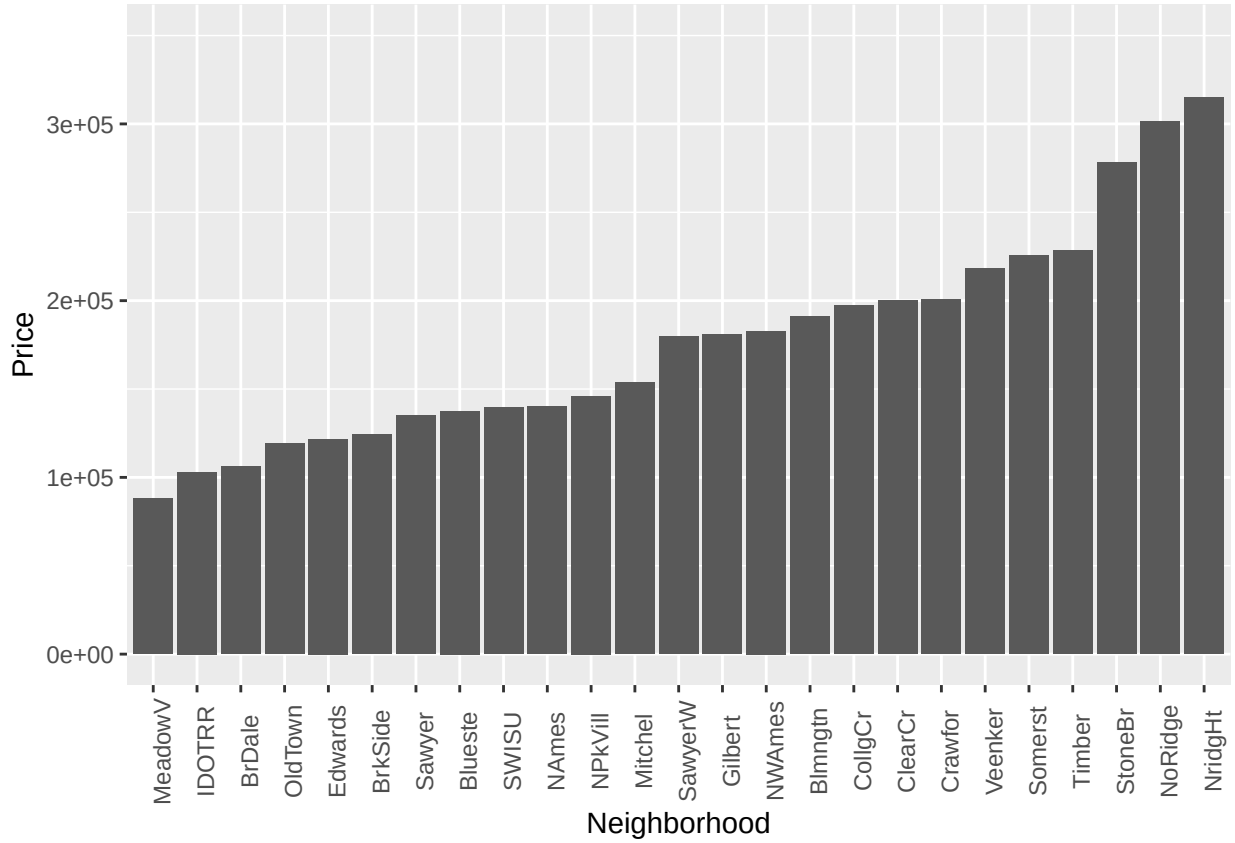
```
##
## Call:
## lm(formula = SalePrice ~ OverallQual + ExterQual + LotArea +
##     GrLivArea, data = train_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -454458  -20241      37   18424  301568
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -9.785e+04  4.843e+03 -20.205  < 2e-16 ***
## OverallQual  2.880e+04  1.015e+03  28.385  < 2e-16 ***
## ExterQual    5.071e+03  4.325e+02  11.725  < 2e-16 ***
## LotArea      8.900e-01  1.084e-01   8.211  4.8e-16 ***
## GrLivArea    4.895e+01  2.546e+00  19.227  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39780 on 1455 degrees of freedom
## Multiple R-squared:  0.7499, Adjusted R-squared:  0.7493
## F-statistic:  1091 on 4 and 1455 DF,  p-value: < 2.2e-16
```

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




```
train_clean[,c('Neighborhood','SalePrice')] %>%
  group_by(Neighborhood) %>%
  summarise(avg = median(SalePrice, na.rm = TRUE)) %>%
  arrange(avg) %>%
  mutate(sorted = factor(Neighborhood, levels=Neighborhood)) %>%
  ggplot(aes(x=sorted, y=avg)) +
  geom_bar(stat = "identity") +
  labs(x='Neighborhood', y='Price') +
  ylim(NA, 350000) +
  theme(axis.text.x = element_text(angle=90))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
richNeighborhood <- c('Crawfor', 'ClearCr', 'Veenker', 'Somerst', 'Timber', 'StoneBr', 'NridgeHt', 'NoR
train_clean['IsNeighborhoodRich'] <- (train_clean$Neighborhood %in% richNeighborhood) *1
train_clean$NeighborhoodScored <- recode(train_clean$Neighborhood, 'MeadowV' = 0, 'IDOTRR' = 0, 'Sawyer

head(train_clean$IsNeighborhoodRich)
```

```
## [1] 0 1 0 1 1 0
```

Prediction using the test data

lets first clean clean the test data using the function we created before

```
test_clean <- clean(test)
```

```
Predicted <- predict(lm, test_clean)
summary(Predicted)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -10046  129186  168021  178416  215049  565792
```

Export for scoring

Account (User ID 6332918) | username : abutaha

```
kaggle_results <- data.frame(Id = test$Id, SalePrice=Predicted)
write.csv(kaggle_results,"kaggle_results.csv",row.names = FALSE)
```

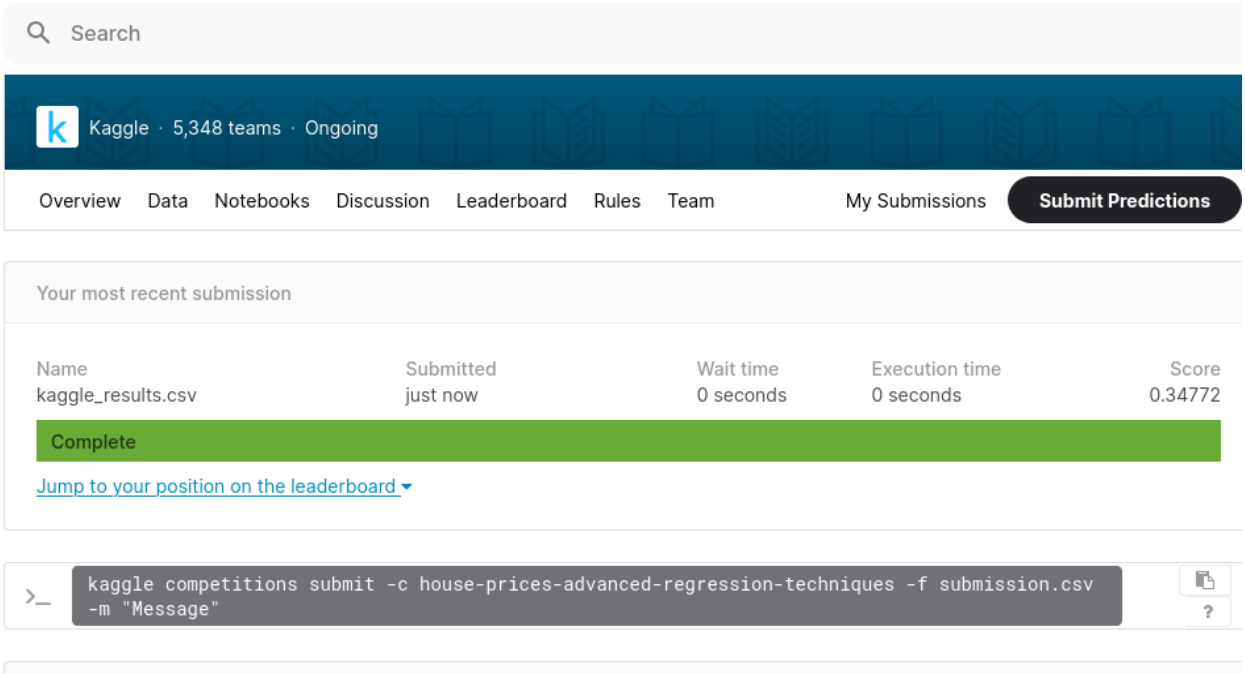


Figure 1: Kaggle Submission