Home_Prices4

Shailaja_K

June 16, 2017

```r
knitr::opts_chunk$set(error = TRUE)
suppressWarnings(library(dplyr))
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
suppressWarnings(library(ggplot2))
suppressWarnings(library(tidyr))
suppressWarnings(library(rpart))
suppressWarnings(library(rpart.plot))
suppressWarnings(library(poLCA))
```

```
## Loading required package: scatterplot3d

## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
suppressWarnings(library(AER))
```

```
## Loading required package: car

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## Loading required package: lmtest

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich
```

```
## Loading required package: survival
```

```r
suppressWarnings(library(randomForest))
```

```
## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
suppressWarnings(library(caret))
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:survival':
##
##     cluster
```

```r
suppressWarnings(library(ALS))
```

```
## Loading required package: nnls

## Loading required package: Iso

## Iso 0.0-17
```

```r
suppressWarnings(library(Matrix))
```

```
##
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':
##
##     expand
```

```r
suppressWarnings(library(relaimpo))
```

```
## Loading required package: boot

##
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':
##
##     melanoma

## The following object is masked from 'package:survival':
##
##     aml

## The following object is masked from 'package:car':
##
```

```
##     logit

## Loading required package: survey

## Loading required package: grid

##
## Attaching package: 'survey'

## The following object is masked from 'package:graphics':
##
##     dotchart

## Loading required package: mitools

## This is the global version of package relaimpo.

## If you are a non-US user, a version with the interesting additional metric pmvd is available

## from Ulrike Groempings web site at prof.beuth-hochschule.de/groemping.
```

Distribution of the Dependent Variable

The dependent variable is SalePrice. Since parametric models give very good interpretation, this section contains multiple checks to see if the dependent variable fits any known theoritical distribution. log-log transformation of the response variable makes it aproximately gaussian.

First, Plot the histogram of SalePrice to identify the distribution.

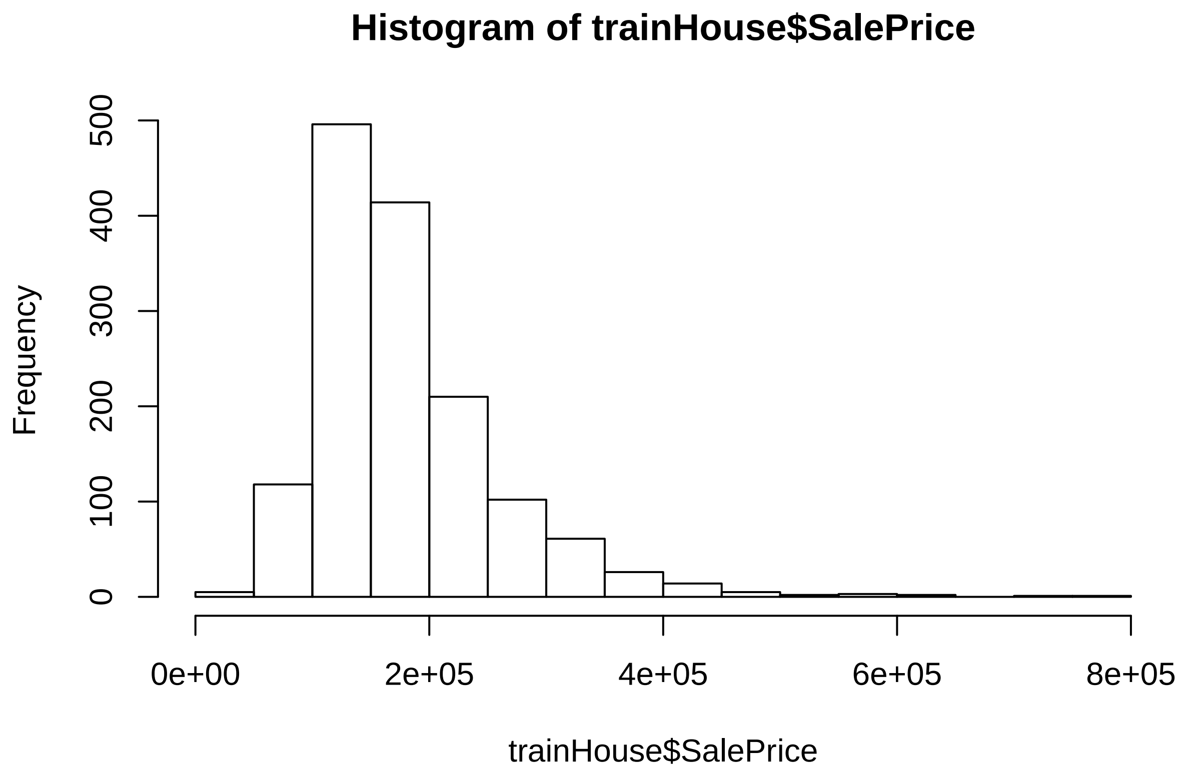
```
trainHouse <- read.csv("./Data/train.csv", header = T)
# testHouse <- read.csv("C:/GitHub_Local/Home_Prices/test.csv", header = T)

# Commenting out. This is line producing a long output.
# glimpse(trainHouse)
# summary(trainHouse)
dim(trainHouse)
```

```
## [1] 1460   81
```

```
# Remove Id variable
trainHouse <- trainHouse[,!colnames(trainHouse) %in% c("Id")]
dim(trainHouse)
```

```
## [1] 1460   80
```

```
hist(trainHouse$SalePrice)
```



The house prices are skewed to the right side. Let us try fitting a set of skewed distributions to SalePrice and determine if the fit is appropriate using kolmogorov-smirnov test.

1. Log-normal: Price is a real valued variable. Log-normal is a skewed distribution, typically applied to prices. The following function performs a 1000 ks.tests for the given data vector with the given distribution. Since the test depends on random number generation, the ks.tests are performed multiple times, instead of performing just once.

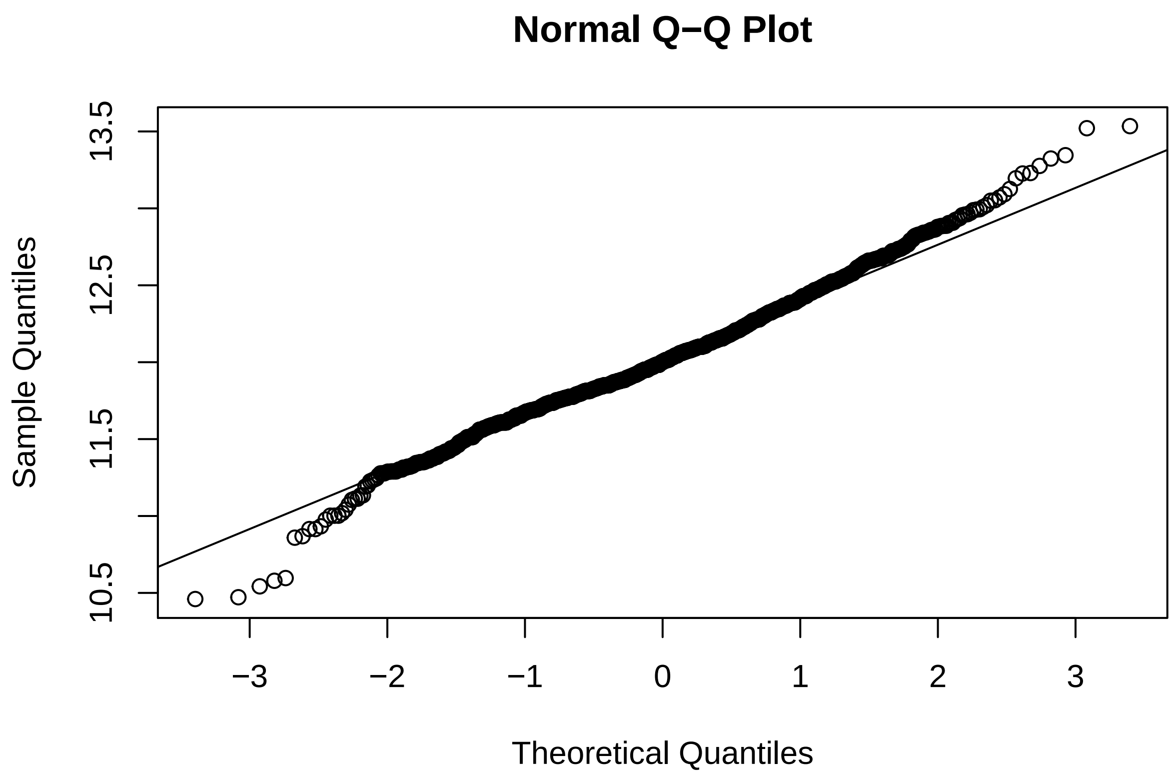
```
set.seed(0)

# 1. Log-normal
# A function to run 1000 ks.tests.
fitDist1000 <- function(vec,fun,params){
  counter = 0
  size = length(vec)
  listofParams <- lapply(c(size,params), function(x){x})
  for(i in c(1:1000)){

  res <- ks.test(vec, do.call(match.fun(fun),listofParams))

  if (res$p.value > 0.05){
    counter = counter+1
    }
  }
  return(counter)
}
```

```r
fit.lognorm.Params <- fitdistr(trainHouse$SalePrice, "lognormal")
(fitDist1000(trainHouse$SalePrice,"rlnorm",fit.lognorm.Params$estimate))
```

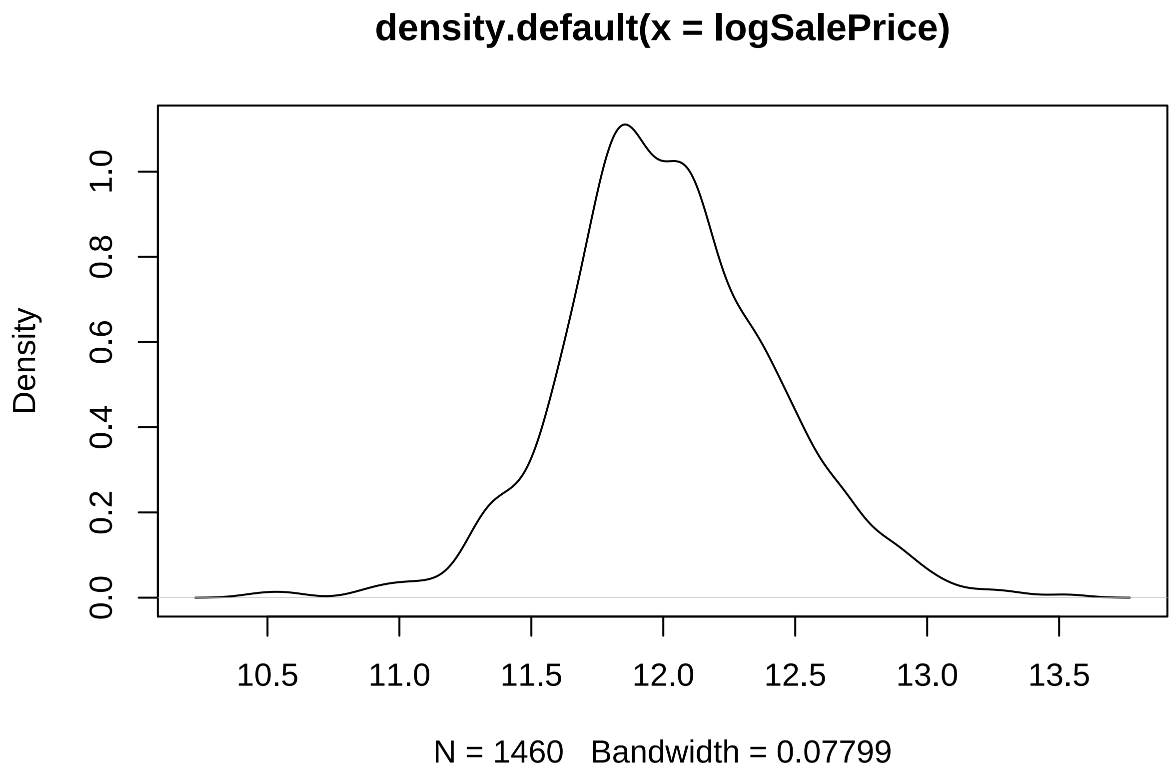
```
## [1] 629
```

Null hypothesis of the ks.test is that the two input vectors have the same distribution. But at 95% confidence level, the null hypothesis is not rejected 63% times. Let us make sure visually that SalePrice distribution looks like log-normal:

```r
logSalePrice <- log(trainHouse$SalePrice)
qqnorm(logSalePrice)
qqline(logSalePrice)
```




```r
plot(density(logSalePrice))
```



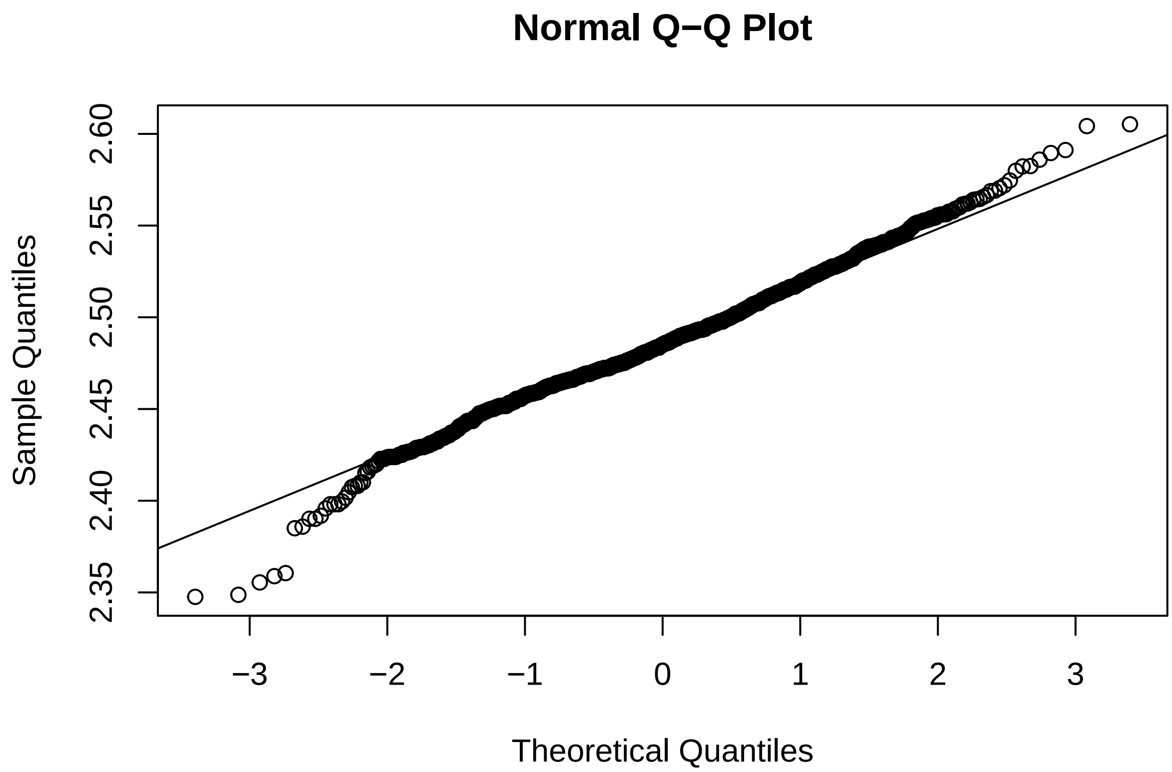
QQ-plot shows that log(SalesPrice) has fatter tails compared to the normal distribution. The distribution is still skewed to the right even after taking log.

2. Loglog transformation: Applying log twice:

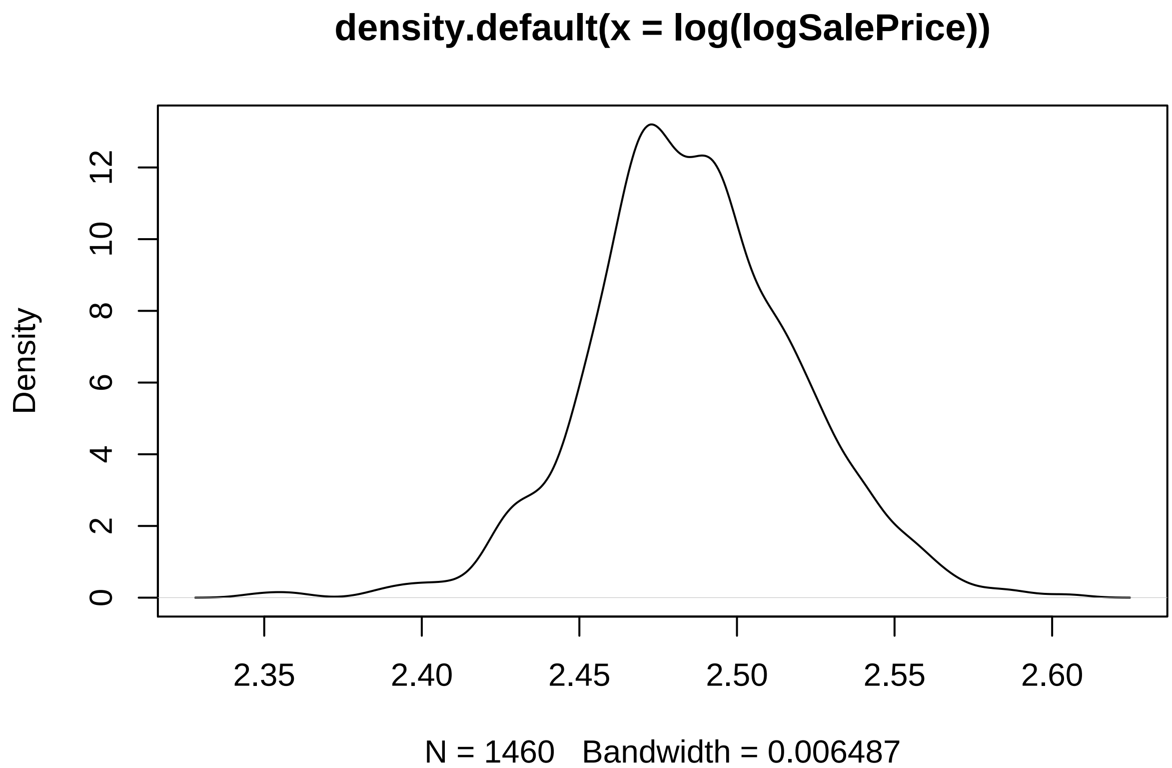
```
# loglog <- log(log(trainHouse$SalePrice))
# hist(loglog)
logSalePrice <- log(trainHouse$SalePrice)

set.seed(0)
fit.loglognorm.Params <- fitdistr(logSalePrice, "lognormal")
(fitDist1000(logSalePrice,"rlnorm",fit.loglognorm.Params$estimate))
```

```
## [1] 745
```

```
qqnorm(log(logSalePrice))
qqline(log(logSalePrice))
```

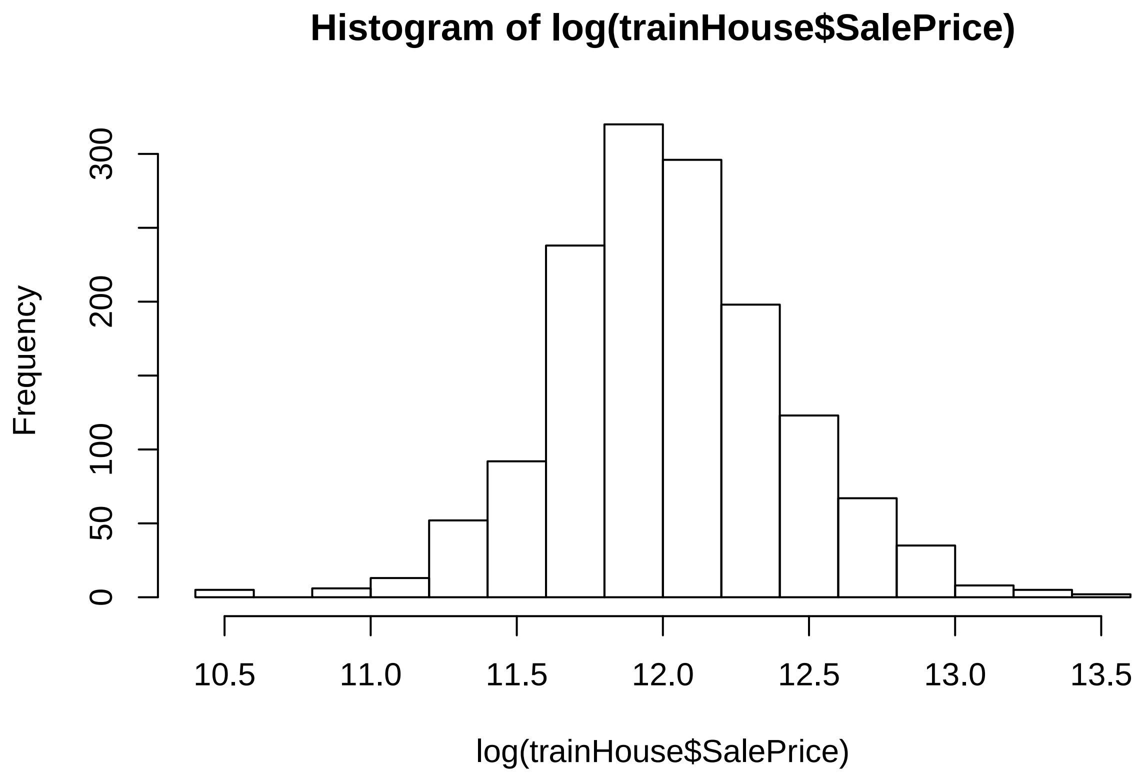



```
plot(density(log(logSalePrice)))
```



Though the density plot and qqplot look similar to those of logSalePrice, loglog transformation seem to fit lognormal distribution better. ks.test could not reject the null hypothesis 74.5% of the time.

```
hist(log(trainHouse$SalePrice))
```




```r
hist(log(log(trainHouse$SalePrice)))
```

Both log and loglog transformed values have almost symmetric distribution with fat tails. So trying t-distribution on log and loglog transformed data might bring the distribution closer to gaussian distribution.

4. t- distribution

```
set.seed(0)
logSalePrice <- log(trainHouse$SalePrice)
fit.pois.Params <- fitdistr(logSalePrice, "t")
(fitDist1000(logSalePrice,"rt",(fit.pois.Params$estimate)[3]))
```

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

```
set.seed(0)
loglogSalePrice <- log(log(trainHouse$SalePrice))
fit.pois.Params <- fitdistr(loglogSalePrice, "t")
(fitDist1000(loglogSalePrice,"rt",(fit.pois.Params$estimate)[3]))
```

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

The log and log-log transformations do not fit t-distribution.

3. Poisson: Even though Poisson distribution is not appropriate for real valued variables, I would like to try fitting Poisson distribution to sales price:

```
set.seed(0)
fit.pois.Params <- fitdistr(trainHouse$SalePrice, "Poisson")
(fitDist1000(trainHouse$SalePrice,"rpois",fit.pois.Params$estimate))
```

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

4. Negative binomial distribution

```
# set.seed(0)
# fit.pois.Params <- fitdistr(trainHouse$SalePrice, "negative binomial")
# (fitDist1000(trainHouse$SalePrice,"rnbinom",fit.pois.Params$estimate))
#
# scaledSalePrice <- scale(trainHouse$SalePrice)
# hist(scaledSalePrice)
```

SalePrice could not fit negative binomial distribution. The system became singular. Scaling the variable resulted in negative values, but negative binomial expects positive values, so can't fit negative binomial to scaled values.

```
set.seed(0)
fit.pois.Params <- fitdistr(trainHouse$SalePrice, "gamma",list(shape = 1, rate = 0.1), lower = 0.01)
(fitDist1000(trainHouse$SalePrice,"rgamma",fit.pois.Params$estimate))
```

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

Gamma distribution did not fit, either.

2. Cleaning the missing data as appropriate.

```
# Data prep
logtrainHouse <- trainHouse
# Add logSalePrice column to the dataset
logtrainHouse$logSalePrice <- log(trainHouse$SalePrice)

# Remove SalePrice column
logtrainHouse <- logtrainHouse[, !colnames(logtrainHouse) %in% c("SalePrice")]

# Make the logSalePrice column to be the first column.
logtrainHouse <- logtrainHouse[,c(80,c(1:79))]
#colnames(logtrainHouse)

# Count the missing values in all columns
missingValuesinColumns <- apply(logtrainHouse,2, function(x){sum(is.na(x))})
missingValuesinColumns[missingValuesinColumns>0]
```

```
##  LotFrontage        Alley   MasVnrType   MasVnrArea     BsmtQual
##          259         1369            8            8           37
##     BsmtCond BsmtExposure BsmtFinType1 BsmtFinType2   Electrical
##           37           38           37           38            1
##  FireplaceQu   GarageType  GarageYrBlt GarageFinish   GarageQual
##          690           81           81           81           81
##   GarageCond       PoolQC        Fence  MiscFeature
##           81         1453         1179         1406
```

```
dim(logtrainHouse)
```

```
## [1] 1460   80
```

```
sum(missingValuesinColumns[missingValuesinColumns>0])
```

```
## [1] 6965
```

```
# sum(missingValuesinColumns>0)
```

Alley, PoolQC, Fence, MiscFeature - These variables have more than 80% values missing. Imputing them from the available values would be unrealistic. Therefore, deleting these columns.

```
logtrainHouse <- logtrainHouse[,c(c(1:6),c(8:57),c(59:72),c(76:80))]
#colnames(logtrainHouse)[4]
```

Garage related fields do not seem to be missing at random. All garage related fields have (almost) equal number of values missing. Let us investigate further:

```
# colnames(logtrainHouse)

# Give 1 to each cell of the df with a missing value
missingDF <- as.data.frame(abs(is.na(logtrainHouse)))

# Extract columns with missing values. sapply applies mean function to each column and returns
# 0 or a positive value indicating no nulls and nulls, respectively.
onlyMissingDF <- missingDF[sapply(missingDF, mean) > 0 ] %>% dplyr::select(contains('Garage'))
head(onlyMissingDF,2)
```

```
##   GarageType GarageYrBlt GarageFinish GarageQual GarageCond
## 1          0           0            0          0          0
## 2          0           0            0          0          0
```

```
# Check the relationship of these variables:
cor(onlyMissingDF)
```

```
##              GarageType GarageYrBlt GarageFinish GarageQual GarageCond
## GarageType            1           1            1          1          1
## GarageYrBlt           1           1            1          1          1
## GarageFinish          1           1            1          1          1
## GarageQual            1           1            1          1          1
## GarageCond            1           1            1          1          1
```

```
head(logtrainHouse[logtrainHouse$GarageArea==0,]%>% dplyr::select(contains('Garage')),2)
```

```
##    GarageType GarageYrBlt GarageFinish GarageCars GarageArea GarageQual
## 40       <NA>          NA         <NA>          0          0       <NA>
## 49       <NA>          NA         <NA>          0          0       <NA>
##    GarageCond
## 40       <NA>
## 49       <NA>
```

```
# a<- c(10,10,10)
# cor(a)
```

The correlation value 1 shows that Garage related fields are not missing at random at all! The Garage related attributes, such as finish and yearbuilt are missing because there is no garage in these houses.

But instead of removing 81 rows with missing garage related attributes, only the variable indicating garage presence, which has no null values - GarageArea - can be included.

```
head(logtrainHouse[is.na(logtrainHouse$GarageType),] %>% dplyr::select(contains("Garage")),2)
```

```
##    GarageType GarageYrBlt GarageFinish GarageCars GarageArea GarageQual
## 40       <NA>          NA         <NA>          0          0       <NA>
## 49       <NA>          NA         <NA>          0          0       <NA>
##    GarageCond
```

```
## 40       <NA>
## 49       <NA>
```

```r
# Exclude all garage related fileds except GarageArea
logtrainHouse <- logtrainHouse %>% dplyr::select(-starts_with('Garage'), GarageArea)

# colnames(logtrainHouse)
```

Imputation: Convert year to age. Put 999 where there is no garage. Cars = 0, remove the finish column.

Check the missingess of Basement related fields:

```r
head(logtrainHouse[is.na(logtrainHouse$BsmtQual),] %>% dplyr::select(contains("Bsmt")),2)
```

```
##    BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinSF1 BsmtFinType2
## 18     <NA>     <NA>         <NA>         <NA>          0         <NA>
## 40     <NA>     <NA>         <NA>         <NA>          0         <NA>
##    BsmtFinSF2 BsmtUnfSF TotalBsmtSF BsmtFullBath BsmtHalfBath
## 18          0         0           0            0            0
## 40          0         0           0            0            0
```

```r
logtrainHouse <- logtrainHouse %>% dplyr::select(-starts_with('Bsmt'),BsmtFinSF1,BsmtFinSF2)

# colnames(logtrainHouse)
```

Basement related fields are also not missing at random. Excluding all additional parameteres related to basement except BasementFinSF.

LotForntage has 259 values missing. Investigate the nature of missingness.

```r
# colnames(onlyMissingDF)

# Give 1 to each cell of the df with a missing value
missingDF <- as.data.frame(abs(is.na(logtrainHouse)))

# Extract columns with missing values. sapply applies mean function to each column
# and returns 0 or a positive value indicating no nulls and nulls, respectively.
onlyMissingDF <- missingDF[sapply(missingDF, mean) > 0 ]
head(onlyMissingDF,2)
```

```
##   LotFrontage MasVnrType MasVnrArea Electrical
## 1           0          0          0          0
## 2           0          0          0          0
```

```r
# Missingmess in LotFrontage is not coinciding with missingness in any other column.
# This may be missing at random. Remove 259 rows from the dataset.
cor(onlyMissingDF)
```

```
##              LotFrontage   MasVnrType   MasVnrArea   Electrical
## LotFrontage   1.00000000  0.014107374  0.014107374 -0.012157681
## MasVnrType    0.01410737  1.000000000  1.000000000 -0.001943274
## MasVnrArea    0.01410737  1.000000000  1.000000000 -0.001943274
## Electrical   -0.01215768 -0.001943274 -0.001943274  1.000000000
```

```r
# dim(logtrainHouse)
logtrainHouse <- logtrainHouse[!is.na(logtrainHouse$LotFrontage),]
```

Missingness of LotFrontage does not seem to coincide with others. This could be missing at random. Therefore, removing the rows with missing values in LotFrontage column.

Eventhough missingness in MasVnrArea and MasVnrType are coinciding, the number of rows with missing values are small. Removing those rows may not impact the solution much.

Check the missingness of the rest of the data frame. Since the number of rows with missing data is small, remove the rows.

```
missingValuesinColumns <- apply(logtrainHouse,2, function(x){sum(is.na(x))})
missingValuesinColumns
```

```
##   logSalePrice    MSSubClass      MSZoning   LotFrontage       LotArea
##              0             0             0             0             0
##         Street      LotShape   LandContour     Utilities     LotConfig
##              0             0             0             0             0
##      LandSlope  Neighborhood    Condition1    Condition2      BldgType
##              0             0             0             0             0
##     HouseStyle   OverallQual   OverallCond     YearBuilt  YearRemodAdd
##              0             0             0             0             0
##      RoofStyle      RoofMatl   Exterior1st   Exterior2nd    MasVnrType
##              0             0             0             0             6
##     MasVnrArea     ExterQual     ExterCond    Foundation   TotalBsmtSF
##              6             0             0             0             0
##        Heating     HeatingQC    CentralAir    Electrical      X1stFlrSF
##              0             0             0             1             0
##      X2ndFlrSF  LowQualFinSF     GrLivArea      FullBath      HalfBath
##              0             0             0             0             0
##   BedroomAbvGr  KitchenAbvGr   KitchenQual  TotRmsAbvGrd    Functional
##              0             0             0             0             0
##     Fireplaces    PavedDrive    WoodDeckSF   OpenPorchSF EnclosedPorch
##              0             0             0             0             0
##     X3SsnPorch   ScreenPorch      PoolArea       MiscVal        MoSold
##              0             0             0             0             0
##         YrSold      SaleType SaleCondition    GarageArea    BsmtFinSF1
##              0             0             0             0             0
##     BsmtFinSF2
##              0
```

```
logtrainHouse <- logtrainHouse[apply(logtrainHouse,1, function(x){sum(is.na(x))==0}),]
# dim(logtrainHouse)

# sum(is.na(logtrainHouse))
```

All missing values have been eliminated.

Some of the variables are 'Year' vaules, converting them to duration would be appropriate.

```
# Replace year column with age, w.r.t 2017.

yearToAge <- (logtrainHouse %>% dplyr::select(contains('Year'),YrSold) %>%
                mutate(BuiltBefore = (2017 - YearBuilt),
                       RemodBefore = (2017 - YearBuilt),
                       SoldBefore = (2017 - YrSold)) %>%
                dplyr::select(BuiltBefore, RemodBefore,SoldBefore))
```

```
## Warning: package 'bindrcpp' was built under R version 3.3.3
```

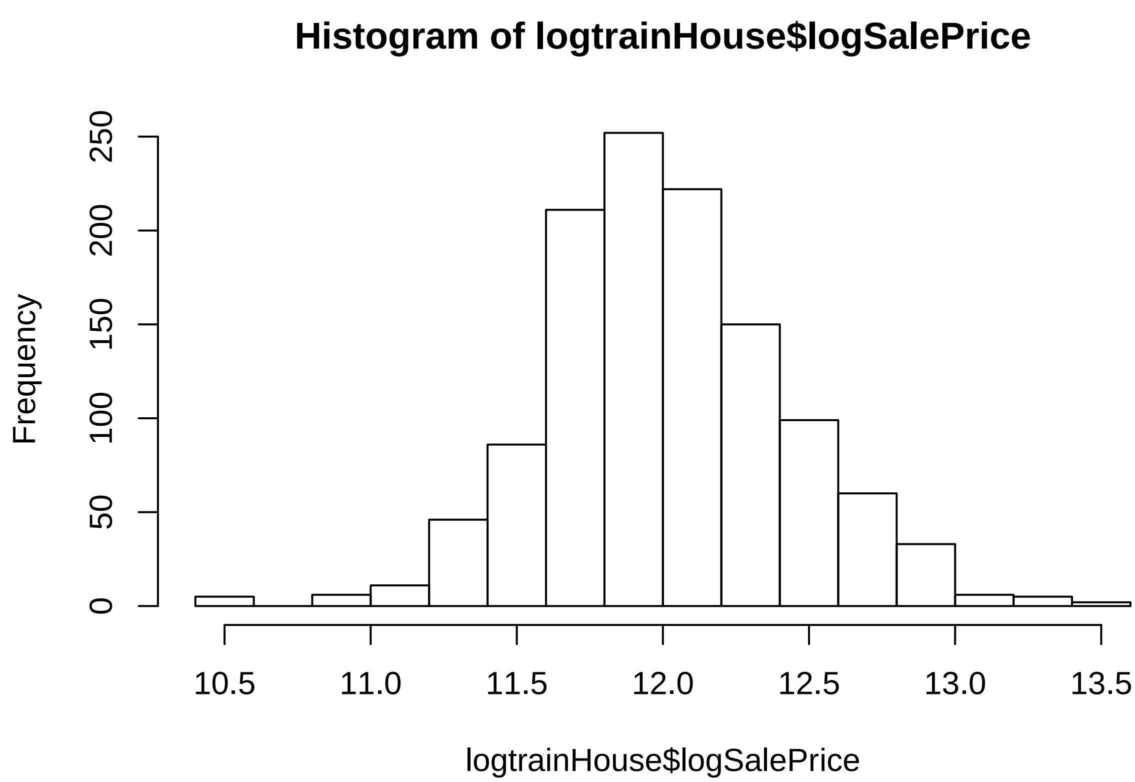
```
logtrainHouse <- logtrainHouse %>% dplyr::select(-contains('Year'))
logtrainHouse <- logtrainHouse %>% dplyr::select(-YrSold)
```

```
logtrainHouse$BuiltBefore <- yearToAge$BuiltBefore
logtrainHouse$RemodBefore <-  yearToAge$RemodBefore
logtrainHouse$SoldBefore <- yearToAge$SoldBefore
logtrainHouse <- logtrainHouse[,!colnames(logtrainHouse) %in% c("MoSold")]
```

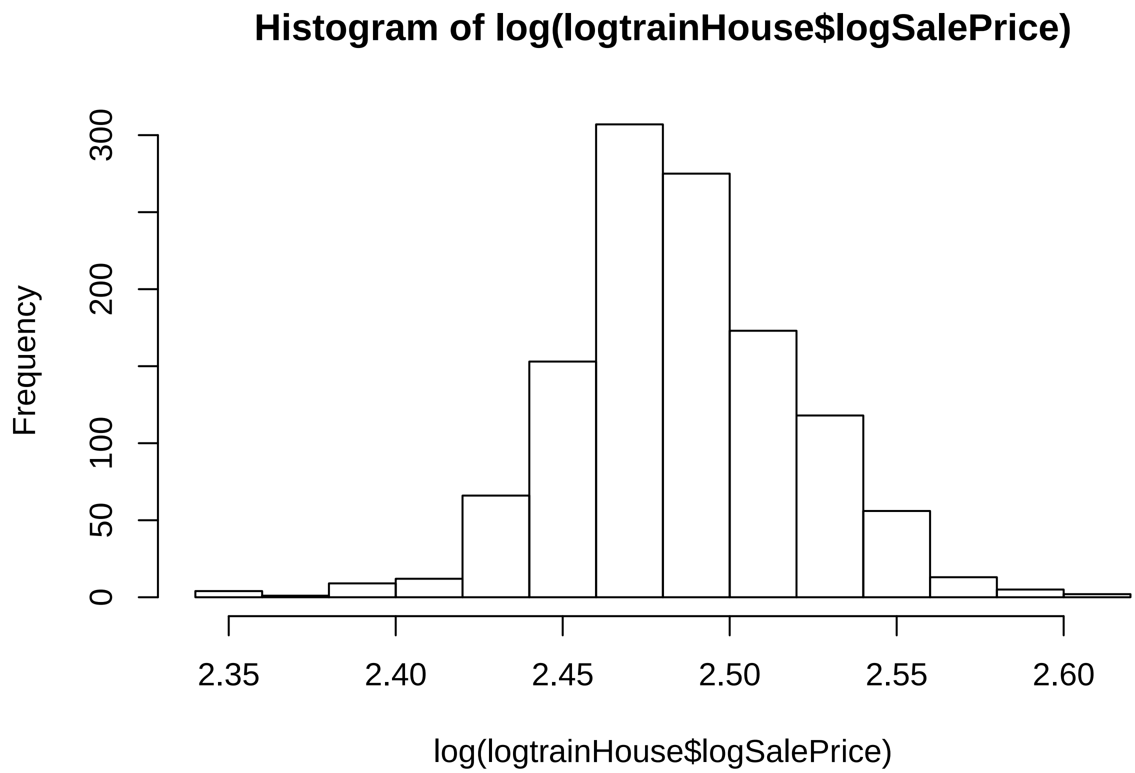
The above code also removes 'Month sold' column, since 'Year Sold' variable is already present in the dataset, this variable do not add much value.

Let us check the distribution of the dependent variable again.

```
hist(logtrainHouse$logSalePrice)
```




```
hist(log(logtrainHouse$logSalePrice))
```



Check if the cleansed data fits t-distribution.

```
set.seed(0)
logSalePrice <- logtrainHouse$logSalePrice
fit.pois.Params <- fitdistr(logSalePrice, "t")
(fitDist1000(logSalePrice,"rt",(fit.pois.Params$estimate)[3]))
```

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

```
set.seed(0)
loglogSalePrice <- log(logSalePrice)
fit.pois.Params <- fitdistr(loglogSalePrice, "t")
(fitDist1000(loglogSalePrice,"rt",(fit.pois.Params$estimate)[3]))
```

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

```
# colnames(logtrainHouse)
```

t-distribution does not fit the data.

Check again what transformation fits the response variable better with lognormal distribution.

```
set.seed(0)
logSalePrice <- logtrainHouse$logSalePrice
fit.pois.Params <- fitdistr(logSalePrice, "normal")
(fitDist1000(logSalePrice,"rnorm",fit.pois.Params$estimate))
```

```
## [1] 590
```

```r
set.seed(0)
loglogSalePrice <- log(logSalePrice)
fit.pois.Params <- fitdistr(loglogSalePrice, "normal")
(fitDist1000(loglogSalePrice,"rnorm",fit.pois.Params$estimate))
```

```
## [1] 740
```

Log-log transformation seems to work well even with the truncated data.

3. Model fitting

1. GLM (with gaussian family)

```r
model2 <- glm(log(logSalePrice)~. , data = logtrainHouse)
```

```
## Error in `contrasts<-`(`*tmp*`, value = contr.funs[1 + isOF[nn]]): contrasts can be applied only to f
```

```r
summary(model2)
```

```
## Error in summary(model2): object 'model2' not found
```

But glm is failing because 'Utilities' column is categorical and has a single level. Since all the values are the same, it doesn't explain any variance in logSalePrice column. Therefore, removing the column.

```r
factorsWith1Level <- function(x){
  if(is.factor(x)){
    return(length(unique(x)) == 1)
  }
  else{
    return(FALSE)
  }
}

names(which(sapply(logtrainHouse, factorsWith1Level)))
```

```
## [1] "Utilities"
```

```r
logtrainHouse <- logtrainHouse[,-which(sapply(logtrainHouse, factorsWith1Level))]
```

glm with log and loglog SalePrice as response variable:

```r
# Test if glm fails again by trying log model
loglinearModel <- glm(logSalePrice~.,data = logtrainHouse)

# log-log model
logloglinearModel <- glm(log(logSalePrice)~.,data = logtrainHouse)


(c(logModelAIC=loglinearModel$aic, loglogModelAIC=logloglinearModel$aic))
```

```
##    logModelAIC loglogModelAIC
##       -1727.37       -7614.23
```

loglog transformation of the dependent variable results in lower AIC for the given dataset. Therefore, I am pursuing loglog model further.

```r
# ?train
# summary(logloglinearModel)
trCtrl <- trainControl(method = "cv", number = 10)
```

```r
# summary(lm(log(logSalePrice)~.,data = logtrainHouse))
lm.cv <- train(log(logSalePrice)~.,
               data = logtrainHouse,
               method = "lm",
               trControl = trCtrl)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```r
lm.cv$results
```

```
##   intercept       RMSE  Rsquared      RMSESD RsquaredSD
## 1      TRUE 0.01772482 0.7675574 0.007152877  0.1443489
```

```r
summary(lm.cv$finalModel)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.061828 -0.003914  0.000133  0.004698  0.056312
##
## Coefficients: (9 not defined because of singularities)
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           2.087e+00  2.410e-02  86.585  < 2e-16 ***
## MSSubClass           -2.285e-05  4.183e-05  -0.546 0.585053
```

```
## MSZoningFV           4.072e-02  4.939e-03   8.243 5.17e-16 ***
## MSZoningRH           3.721e-02  5.022e-03   7.408 2.69e-13 ***
## MSZoningRL           3.694e-02  4.213e-03   8.769  < 2e-16 ***
## MSZoningRM           3.500e-02  3.936e-03   8.893  < 2e-16 ***
## LotFrontage          4.021e-05  2.008e-05   2.003 0.045497 *
## LotArea              2.513e-07  5.653e-08   4.446 9.73e-06 ***
## StreetPave           7.244e-03  5.332e-03   1.359 0.174582
## LotShapeIR2          2.445e-03  2.121e-03   1.153 0.249385
## LotShapeIR3          4.727e-03  4.603e-03   1.027 0.304691
## LotShapeReg          7.432e-04  7.718e-04   0.963 0.335818
## LandContourHLS       4.849e-03  2.237e-03   2.168 0.030398 *
## LandContourLow      -2.117e-03  3.281e-03  -0.645 0.518837
## LandContourLvl       3.578e-03  1.641e-03   2.181 0.029450 *
## LotConfigCulDSac     2.926e-03  1.914e-03   1.529 0.126602
## LotConfigFR2        -2.576e-03  1.956e-03  -1.317 0.188147
## LotConfigFR3        -8.581e-03  5.149e-03  -1.666 0.095926 .
## LotConfigInside     -1.054e-03  8.307e-04  -1.269 0.204890
## LandSlopeMod         3.217e-03  1.782e-03   1.805 0.071368 .
## LandSlopeSev        -1.758e-02  6.016e-03  -2.923 0.003548 **
## NeighborhoodBlueste -1.707e-03  7.740e-03  -0.221 0.825518
## NeighborhoodBrDale  -5.186e-03  4.707e-03  -1.102 0.270777
## NeighborhoodBrkSide  9.667e-04  4.145e-03   0.233 0.815626
## NeighborhoodClearCr  3.386e-03  4.422e-03   0.766 0.444022
## NeighborhoodCollgCr  4.550e-05  3.178e-03   0.014 0.988578
## NeighborhoodCrawfor  1.091e-02  3.770e-03   2.894 0.003887 **
## NeighborhoodEdwards -5.475e-03  3.472e-03  -1.577 0.115142
## NeighborhoodGilbert  3.686e-04  3.479e-03   0.106 0.915645
## NeighborhoodIDOTRR  -1.500e-03  4.639e-03  -0.323 0.746421
## NeighborhoodMeadowV -1.418e-02  4.879e-03  -2.906 0.003739 **
## NeighborhoodMitchel -2.995e-03  3.634e-03  -0.824 0.410058
## NeighborhoodNAmes   -1.089e-03  3.406e-03  -0.320 0.749254
## NeighborhoodNoRidge  3.505e-03  3.723e-03   0.941 0.346722
## NeighborhoodNPkVill -1.162e-03  7.519e-03  -0.155 0.877173
## NeighborhoodNridgHt  6.975e-03  3.196e-03   2.182 0.029336 *
## NeighborhoodNWAmes  -2.421e-03  3.616e-03  -0.670 0.503268
## NeighborhoodOldTown -3.358e-03  4.167e-03  -0.806 0.420518
## NeighborhoodSawyer  -3.202e-04  3.622e-03  -0.088 0.929569
## NeighborhoodSawyerW  3.166e-04  3.410e-03   0.093 0.926036
## NeighborhoodSomerst  1.836e-03  3.853e-03   0.477 0.633716
## NeighborhoodStoneBr  1.206e-02  3.697e-03   3.263 0.001140 **
## NeighborhoodSWISU    1.675e-03  4.166e-03   0.402 0.687762
## NeighborhoodTimber   1.121e-03  3.608e-03   0.311 0.756183
## NeighborhoodVeenker  7.412e-03  5.001e-03   1.482 0.138629
## Condition1Feedr      1.099e-03  2.110e-03   0.521 0.602596
## Condition1Norm       6.041e-03  1.694e-03   3.566 0.000380 ***
## Condition1PosA       2.532e-03  6.213e-03   0.408 0.683715
## Condition1PosN       4.801e-03  4.220e-03   1.138 0.255524
## Condition1RRAe      -1.937e-03  4.067e-03  -0.476 0.634007
## Condition1RRAn       4.617e-03  2.877e-03   1.605 0.108808
## Condition1RRNe       7.744e-03  9.711e-03   0.797 0.425400
## Condition1RRNn       8.270e-03  5.909e-03   1.400 0.161932
## Condition2Feedr      8.064e-03  9.123e-03   0.884 0.376977
## Condition2Norm       1.952e-03  7.747e-03   0.252 0.801121
## Condition2PosA       1.292e-02  1.489e-02   0.867 0.386009
```

```
## Condition2PosN       -6.991e-02  1.124e-02  -6.219 7.31e-10 ***
## Condition2RRAe               NA         NA      NA       NA
## Condition2RRAn               NA         NA      NA       NA
## Condition2RRNn       -6.021e-04  1.067e-02  -0.056 0.955003
## BldgType2fmCon        4.865e-03  6.009e-03   0.810 0.418310
## BldgTypeDuplex       -1.213e-05  3.283e-03  -0.004 0.997053
## BldgTypeTwnhs        -4.020e-03  4.898e-03  -0.821 0.411969
## BldgTypeTwnhsE        2.169e-04  4.493e-03   0.048 0.961503
## HouseStyle1.5Unf     -2.785e-03  3.199e-03  -0.870 0.384274
## HouseStyle1Story     -3.654e-03  1.921e-03  -1.902 0.057498 .
## HouseStyle2.5Fin     -4.130e-03  5.043e-03  -0.819 0.413060
## HouseStyle2.5Unf      2.305e-03  3.934e-03   0.586 0.557998
## HouseStyle2Story     -1.705e-03  1.574e-03  -1.083 0.278885
## HouseStyleSFoyer      2.439e-04  2.716e-03   0.090 0.928443
## HouseStyleSLvl       -6.553e-04  2.570e-03  -0.255 0.798817
## OverallQual           3.762e-03  4.505e-04   8.350 2.22e-16 ***
## OverallCond           3.568e-03  3.695e-04   9.657  < 2e-16 ***
## RoofStyleGable        1.292e-02  1.073e-02   1.203 0.229092
## RoofStyleGambrel      1.393e-02  1.122e-02   1.242 0.214668
## RoofStyleHip          1.287e-02  1.077e-02   1.196 0.232158
## RoofStyleMansard      1.725e-02  1.175e-02   1.468 0.142507
## RoofStyleShed                NA         NA      NA       NA
## RoofMatlCompShg       2.359e-01  1.341e-02  17.591  < 2e-16 ***
## RoofMatlMembran       2.931e-01  2.125e-02  13.794  < 2e-16 ***
## RoofMatlMetal                NA         NA      NA       NA
## RoofMatlRoll          2.402e-01  1.664e-02  14.435  < 2e-16 ***
## `RoofMatlTar&Grv`     2.456e-01  1.641e-02  14.966  < 2e-16 ***
## RoofMatlWdShake       2.263e-01  1.657e-02  13.657  < 2e-16 ***
## RoofMatlWdShngl       2.445e-01  1.392e-02  17.559  < 2e-16 ***
## Exterior1stAsphShn    9.552e-04  1.351e-02   0.071 0.943634
## Exterior1stBrkComm   -1.690e-02  1.246e-02  -1.356 0.175346
## Exterior1stBrkFace    6.545e-03  5.220e-03   1.254 0.210146
## Exterior1stCBlock    -6.818e-03  1.086e-02  -0.628 0.530148
## Exterior1stCemntBd   -1.182e-02  8.983e-03  -1.315 0.188707
## Exterior1stHdBoard    2.090e-03  5.303e-03   0.394 0.693590
## Exterior1stImStucc   -3.463e-03  1.127e-02  -0.307 0.758721
## Exterior1stMetalSd    8.236e-03  5.993e-03   1.374 0.169657
## Exterior1stPlywood    4.427e-05  5.258e-03   0.008 0.993285
## Exterior1stStone      4.998e-03  1.258e-02   0.397 0.691227
## Exterior1stStucco     3.206e-03  5.843e-03   0.549 0.583357
## Exterior1stVinylSd    2.881e-03  5.433e-03   0.530 0.596038
## `Exterior1stWd Sdng` -3.664e-04  4.992e-03  -0.073 0.941508
## Exterior1stWdShing    2.294e-03  5.362e-03   0.428 0.668865
## Exterior2ndAsphShn    7.808e-04  9.021e-03   0.087 0.931041
## `Exterior2ndBrk Cmn`  7.147e-03  9.489e-03   0.753 0.451524
## Exterior2ndBrkFace   -1.906e-03  5.422e-03  -0.352 0.725235
## Exterior2ndCBlock            NA         NA      NA       NA
## Exterior2ndCmentBd    1.701e-02  8.849e-03   1.922 0.054886 .
## Exterior2ndHdBoard    5.976e-04  5.124e-03   0.117 0.907172
## Exterior2ndImStucc    4.035e-03  5.850e-03   0.690 0.490478
## Exterior2ndMetalSd   -2.829e-03  5.836e-03  -0.485 0.628010
## Exterior2ndOther     -6.343e-03  1.111e-02  -0.571 0.568007
## Exterior2ndPlywood    2.817e-03  4.909e-03   0.574 0.566254
## Exterior2ndStone      1.645e-03  7.745e-03   0.212 0.831883
```

```
## Exterior2ndStucco     1.761e-03  5.621e-03   0.313 0.754137
## Exterior2ndVinylSd    1.819e-03  5.218e-03   0.349 0.727522
## `Exterior2ndWd Sdng`  4.079e-03  4.800e-03   0.850 0.395624
## `Exterior2ndWd Shng`  1.446e-04  4.975e-03   0.029 0.976818
## MasVnrTypeBrkFace     3.144e-03  3.417e-03   0.920 0.357780
## MasVnrTypeNone        2.697e-03  3.417e-03   0.789 0.430167
## MasVnrTypeStone       4.694e-03  3.553e-03   1.321 0.186767
## MasVnrArea           -7.580e-07  2.533e-06  -0.299 0.764782
## ExterQualFa           1.484e-03  4.620e-03   0.321 0.748182
## ExterQualGd          -1.070e-04  2.084e-03  -0.051 0.959050
## ExterQualTA          -1.469e-04  2.353e-03  -0.062 0.950211
## ExterCondFa          -7.571e-03  7.378e-03  -1.026 0.305031
## ExterCondGd          -5.251e-03  7.000e-03  -0.750 0.453312
## ExterCondPo          -9.583e-03  1.297e-02  -0.739 0.460017
## ExterCondTA          -4.343e-03  6.980e-03  -0.622 0.533941
## FoundationCBlock      1.120e-03  1.334e-03   0.839 0.401403
## FoundationPConc       2.954e-03  1.472e-03   2.007 0.044985 *
## FoundationSlab       -3.424e-03  3.449e-03  -0.993 0.321101
## FoundationStone       1.276e-02  4.441e-03   2.873 0.004147 **
## FoundationWood       -1.495e-02  7.338e-03  -2.037 0.041876 *
## TotalBsmtSF           5.736e-06  1.735e-06   3.305 0.000983 ***
## HeatingGasA          -9.925e-03  6.793e-03  -1.461 0.144293
## HeatingGasW          -3.348e-03  7.193e-03  -0.465 0.641739
## HeatingGrav          -2.900e-02  7.948e-03  -3.649 0.000277 ***
## HeatingOthW          -8.096e-03  9.949e-03  -0.814 0.415978
## HeatingWall                  NA         NA      NA       NA
## HeatingQCFa          -2.888e-03  2.143e-03  -1.347 0.178175
## HeatingQCGd          -2.792e-03  9.335e-04  -2.990 0.002853 **
## HeatingQCPo          -7.425e-03  1.090e-02  -0.681 0.496012
## HeatingQCTA          -3.844e-03  9.384e-04  -4.097 4.53e-05 ***
## CentralAirY           6.221e-03  1.644e-03   3.784 0.000164 ***
## ElectricalFuseF      -1.030e-03  2.505e-03  -0.411 0.680978
## ElectricalFuseP      -6.699e-03  7.011e-03  -0.956 0.339513
## ElectricalMix         8.138e-03  1.139e-02   0.715 0.474980
## ElectricalSBrkr      -4.151e-04  1.282e-03  -0.324 0.746157
## X1stFlrSF             2.082e-05  2.332e-06   8.927  < 2e-16 ***
## X2ndFlrSF             1.531e-05  2.328e-06   6.576 7.75e-11 ***
## LowQualFinSF          1.267e-05  7.985e-06   1.586 0.112979
## GrLivArea                    NA         NA      NA       NA
## FullBath              2.261e-03  9.705e-04   2.330 0.020028 *
## HalfBath              2.788e-03  9.302e-04   2.997 0.002790 **
## BedroomAbvGr          2.036e-04  5.961e-04   0.342 0.732733
## KitchenAbvGr         -3.353e-03  2.401e-03  -1.397 0.162770
## KitchenQualFa        -4.869e-03  2.658e-03  -1.831 0.067326 .
## KitchenQualGd        -4.469e-03  1.462e-03  -3.057 0.002292 **
## KitchenQualTA        -5.495e-03  1.675e-03  -3.281 0.001072 **
## TotRmsAbvGrd          5.697e-04  4.223e-04   1.349 0.177568
## FunctionalMaj2       -2.093e-02  5.907e-03  -3.542 0.000415 ***
## FunctionalMin1        2.357e-03  3.687e-03   0.639 0.522848
## FunctionalMin2       -5.215e-04  3.627e-03  -0.144 0.885705
## FunctionalMod        -4.665e-03  4.532e-03  -1.029 0.303536
## FunctionalSev                NA         NA      NA       NA
## FunctionalTyp         4.878e-03  3.107e-03   1.570 0.116674
## Fireplaces            2.074e-03  6.203e-04   3.344 0.000858 ***
```

```
## PavedDriveP          -1.780e-03  2.384e-03  -0.747 0.455410
## PavedDriveY           8.634e-04  1.433e-03   0.602 0.547034
## WoodDeckSF            8.915e-06  2.687e-06   3.318 0.000938 ***
## OpenPorchSF           4.857e-06  5.228e-06   0.929 0.353051
## EnclosedPorch         1.382e-05  5.602e-06   2.467 0.013774 *
## X3SsnPorch            2.031e-05  9.956e-06   2.040 0.041575 *
## ScreenPorch           2.414e-05  5.554e-06   4.346 1.53e-05 ***
## PoolArea              4.290e-06  8.292e-06   0.517 0.604959
## MiscVal              -3.148e-06  1.684e-06  -1.869 0.061918 .
## SaleTypeCon           6.515e-03  7.309e-03   0.891 0.372984
## SaleTypeConLD         1.219e-02  4.282e-03   2.847 0.004507 **
## SaleTypeConLI        -4.281e-03  5.261e-03  -0.814 0.416034
## SaleTypeConLw         1.920e-03  5.007e-03   0.384 0.701406
## SaleTypeCWD           4.877e-03  5.336e-03   0.914 0.360875
## SaleTypeNew           3.955e-03  6.557e-03   0.603 0.546498
## SaleTypeOth           7.723e-03  5.925e-03   1.303 0.192718
## SaleTypeWD           -1.803e-03  1.923e-03  -0.938 0.348637
## SaleConditionAdjLand  1.317e-02  5.996e-03   2.197 0.028268 *
## SaleConditionAlloca   9.189e-03  3.802e-03   2.417 0.015840 *
## SaleConditionFamily   4.243e-04  2.645e-03   0.160 0.872579
## SaleConditionNormal   7.056e-03  1.277e-03   5.523 4.23e-08 ***
## SaleConditionPartial  5.028e-03  6.315e-03   0.796 0.426088
## GarageArea            1.561e-05  1.936e-06   8.064 2.08e-15 ***
## BsmtFinSF1            8.220e-06  8.497e-07   9.674  < 2e-16 ***
## BsmtFinSF2            2.900e-06  2.047e-06   1.417 0.156890
## BuiltBefore          -1.813e-04  3.075e-05  -5.897 5.04e-09 ***
## RemodBefore                  NA         NA      NA       NA
## SoldBefore            7.364e-05  2.281e-04   0.323 0.746919
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.009294 on 1011 degrees of freedom
## Multiple R-squared:  0.9388, Adjusted R-squared:  0.9278
## F-statistic: 85.24 on 182 and 1011 DF,  p-value: < 2.2e-16
```

LM resulted in many insignificant parameters, though it has 76% R-squared. Some levels of categorical factors are insignificant. We can work around this by combining the levels step by step. That requires many iterations give the large number of factors in some categorical variables.

Instead, we can try a non-parametric regression that deals with this issue and handles interactions between variables as well.

2. Random Forest Since the trees in random forest minimize the sum of squares, it is better to use the transformed variable whose distribution is closer to normal distribution.

```r
# dim(logtrainHouse)
trCtrl <- trainControl(method = "cv",number=5)

# Try multiple mtry
rf1Grid <-  expand.grid(mtry = c(25,30,50))

# append the cv results to the below data frame
modelResults <- data.frame(mtry=0,RMSE=0,Rsquared=0,RMSESD=0,RsquaredSD=0)

# Random forest iterations with different number of trees and mtry values
for (i in seq(100,300,by = 50)){
```

```r
  rf1 <- train(log(logSalePrice)~.,
      data = logtrainHouse,
      method = "rf",
      trControl = trCtrl,
      tuneGrid = rf1Grid,
      ntrees = i,
      verbose = F)
  modelResults <- rbind(modelResults,rf1$results)
}

# First row contains all zeros. Removing it.
modelResults <- modelResults[-1,]

# Find the combination of variables with lowest RMSE.
modelResults[which.min(modelResults$RMSE),]

modelResults
```

Though the best test RMSE has occured with mtry = 50 (all the variables) and number of trees = 300, a smaller model with mtry =50 and 100 trees is almost as good as the best model. The smaller model has slightly high RMSE, but it achieves this results with fewer trees.

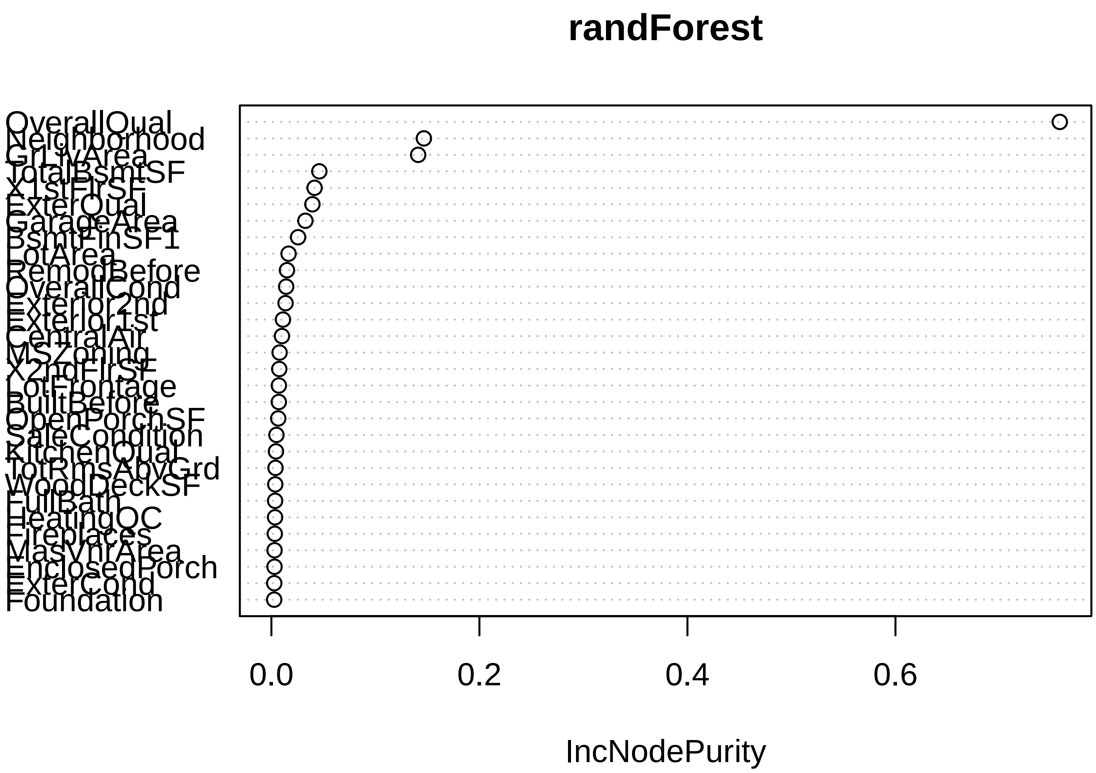
In comparision with linear model, random forest performed better. RMSE of lm = 0.01798476, whereas RMSE of random forest = 0.01217267.

Fit the final random forest with the best parameters.

```r
set.seed(0)
randForest <- randomForest(log(logSalePrice)~.,data = logtrainHouse, mtry = 50, ntree= 100)

# randForest$mse
# randForest$rsq
# randForest$forest


(ForestRMSE = (sum((randForest$y - randForest$predicted)^2)/nrow(logtrainHouse))^0.5)
```

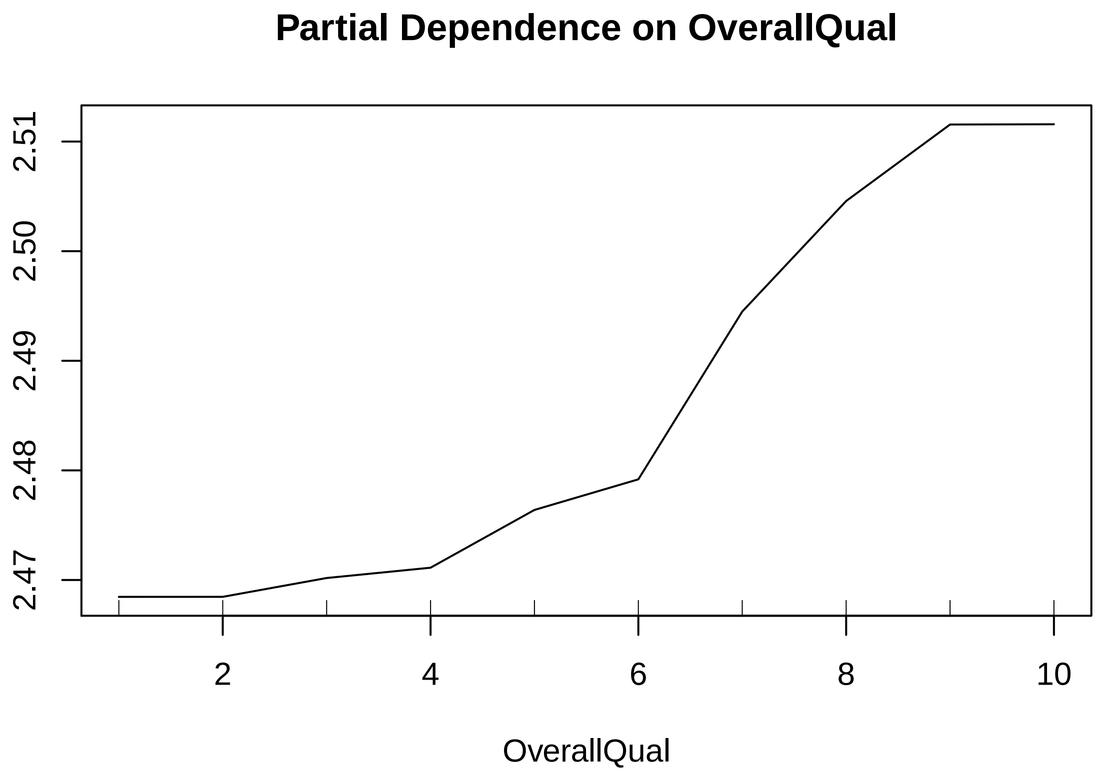
```
## [1] 0.01252426
```

```r
varImpPlot(randForest)
```

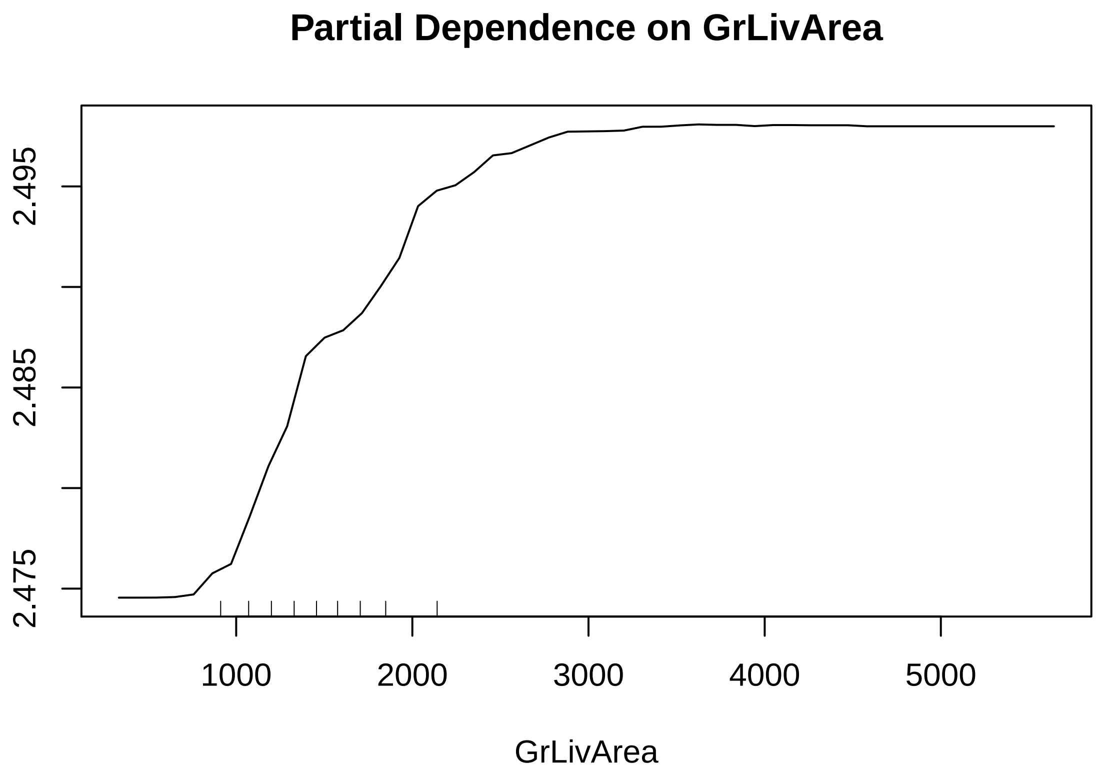


The following partial dependece plots show the marginal impact of a predictor on the dependent variable.

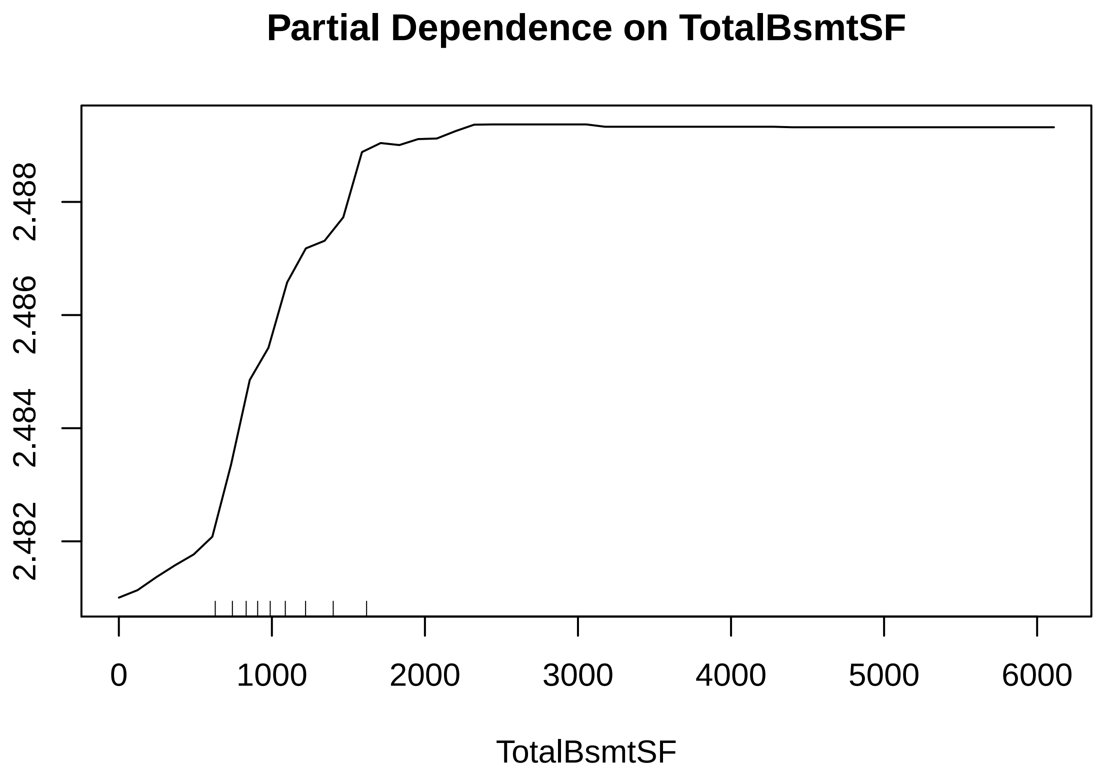
```
# variableImportance <- randForest$importance[order(randForest$importance,decreasing = T),]
# sapply(names(variableImportance[1:5]), function(x) {partialPlot(randForest, logtrainHouse,x)})
# sapply(variableImportance[1:5], function(x) {partialPlot(randForest, logtrainHouse,x)})


partialPlot(randForest, logtrainHouse,OverallQual)
```



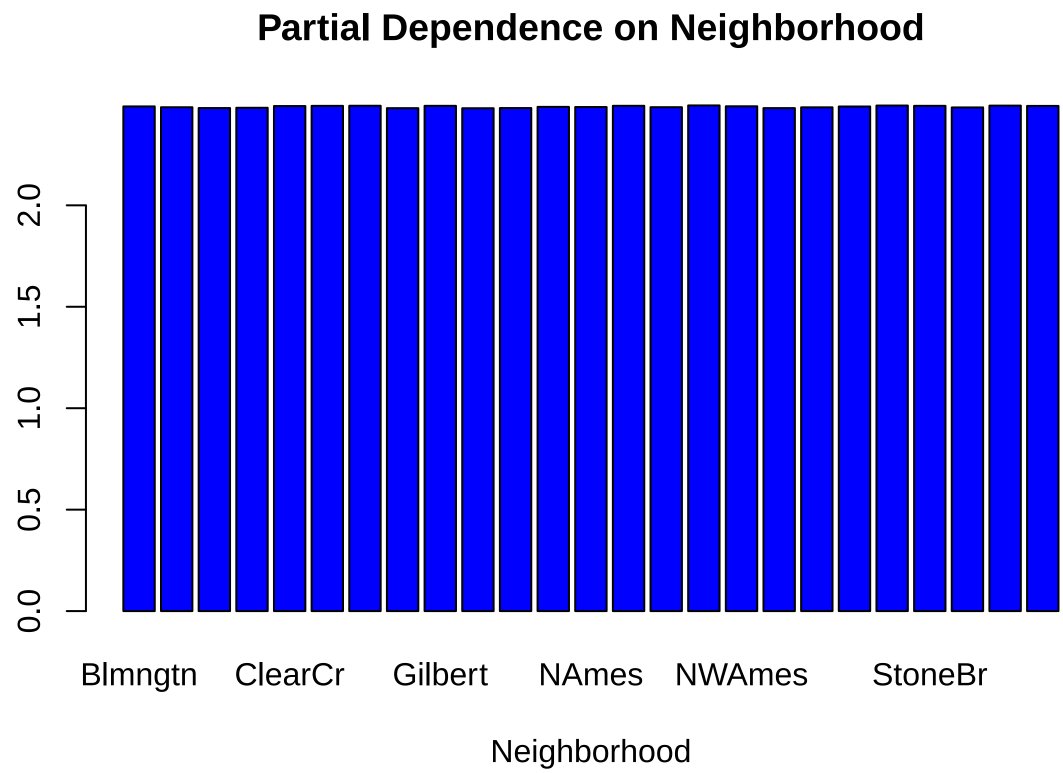

```
partialPlot(randForest, logtrainHouse,GrLivArea)
```




```
partialPlot(randForest, logtrainHouse,TotalBsmtSF)
```




```r
partialPlot(randForest, logtrainHouse,Neighborhood)
```



These partial dependence plots show some interesting insights. 1. As the overall quality of the increases, price goes high. The effect is more dramatic for the higher quality houses (>6). 2. Prices increase as the living area of the house inceases. 3. Total basement square footage matters only upto a point. After 2000 sqft, it doesn't seem to affect the price much. 4. Neighborhood shows a counter intuitive effect, though. According to the plot, neighborhood does impact the price. But in reality, house prices heavily depend on locality.