

# Housing data analysis - Women Who Code workshop

*Darya Vanichkina*

*13/03/2018*

Load the data in from csv.

```
trainH <- read.csv("train.csv")
testH <- read.csv("test.csv")
```

What features are there in the data? What are the dimensions of the data? What are the column headers? Use the summary() and str() functions to explore...

```
dim(trainH)
```

```
## [1] 1459    69
```

```
names(trainH)
```

```
## [1] "Id"           "MSSubClass"    "MSZoning"      "LotFrontage"
## [5] "LotArea"      "Street"        "LotShape"      "LandContour"
## [9] "LotConfig"    "LandSlope"     "Neighborhood"  "BldgType"
## [13] "HouseStyle"   "OverallQual"   "OverallCond"   "YearBuilt"
## [17] "YearRemodAdd" "Foundation"    "BsmtQual"      "BsmtCond"
## [21] "BsmtExposure" "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"
## [25] "BsmtFinSF2"   "BsmtUnfSF"     "TotalBsmtSF"   "Heating"
## [29] "HeatingQC"    "CentralAir"    "Electrical"     "X1stFlrSF"
## [33] "X2ndFlrSF"    "LowQualFinSF"  "GrLivArea"      "BsmtFullBath"
## [37] "BsmtHalfBath" "FullBath"      "HalfBath"       "BedroomAbvGr"
## [41] "KitchenAbvGr" "KitchenQual"   "TotRmsAbvGrd"  "Functional"
## [45] "Fireplaces"   "FireplaceQu"   "GarageType"     "GarageYrBlt"
## [49] "GarageFinish" "GarageCars"    "GarageArea"     "GarageQual"
## [53] "GarageCond"   "PavedDrive"    "WoodDeckSF"     "OpenPorchSF"
## [57] "EnclosedPorch" "X3SsnPorch"    "ScreenPorch"    "PoolArea"
## [61] "PoolQC"       "Fence"         "MiscFeature"    "MiscVal"
## [65] "MoSold"       "YrSold"        "SaleType"       "SaleCondition"
## [69] "SalePrice"
```

```
summary(trainH)
```

```
##      Id      MSSubClass      MSZoning      LotFrontage
## Min.   : 1.0    Min.   : 20.00  C (all): 10  Min.   : 21.00
## 1st Qu.: 365.5  1st Qu.: 20.00  FV      : 65  1st Qu.: 59.00
## Median : 730.0  Median : 50.00  RH      : 16  Median : 69.00
## Mean   : 730.1  Mean   : 56.88  RL      :1150  Mean   : 70.05
## 3rd Qu.:1094.5  3rd Qu.: 70.00  RM      : 218  3rd Qu.: 80.00
## Max.   :1460.0  Max.   :190.00              Max.   :313.00
##                                     NA's   :259
##      LotArea      Street      LotShape      LandContour      LotConfig
## Min.   : 1300    Grvl: 6    IR1:484    Bnk: 63    Corner : 263
## 1st Qu.: 7549    Pave:1453  IR2: 41    HLS: 50    CulDSac: 94
## Median : 9477              IR3: 10    Low: 36    FR2      : 47
## Mean   : 10517              Reg:924    Lvl:1310   FR3      : 4
## 3rd Qu.: 11603              Inside :1051
## Max.   :215245
```

```

##
## LandSlope   Neighborhood   BldgType       HouseStyle     OverallQual
## Gtl:1381   NAmes :225   1Fam :1219   1Story :726   Min. : 1.0
## Mod: 65   CollgCr:150   2fmCon: 31   2Story :445   1st Qu.: 5.0
## Sev: 13   OldTown:113   Duplex: 52   1.5Fin :154   Median : 6.0
##           Edwards:100   Twnhs : 43   SLvl : 64   Mean : 6.1
##           Somerst: 86   TwnhsE: 114   SFoyer : 37   3rd Qu.: 7.0
##           Gilbert: 79           1.5Unf : 14   Max. :10.0
##           (Other):706           (Other): 19
## OverallCond   YearBuilt   YearRemodAdd   Foundation   BsmtQual
## Min. :1.000   Min. :1872   Min. :1950   BrkTil:146   Ex :121
## 1st Qu.:5.000   1st Qu.:1954   1st Qu.:1967   CBlock:634   Fa : 35
## Median :5.000   Median :1973   Median :1994   PConc :646   Gd :617
## Mean :5.576   Mean :1971   Mean :1985   Slab : 24   TA :649
## 3rd Qu.:6.000   3rd Qu.:2000   3rd Qu.:2004   Stone : 6   NA's: 37
## Max. :9.000   Max. :2010   Max. :2010   Wood : 3
##
## BsmtCond   BsmtExposure   BsmtFinType1   BsmtFinSF1   BsmtFinType2
## Fa : 45   Av :221   ALQ :220   Min. : 0.0   ALQ : 19
## Gd : 65   Gd :134   BLQ :148   1st Qu.: 0.0   BLQ : 33
## Po : 2   Mn :114   GLQ :418   Median : 384.0   GLQ : 14
## TA :1310   No :952   LwQ : 74   Mean : 443.9   LwQ : 46
## NA's: 37   NA's: 38   Rec :133   3rd Qu.: 712.5   Rec : 54
##           Unf :429   Max. :5644.0   Unf :1255
##           NA's: 37           NA's: 38
## BsmtFinSF2   BsmtUnfSF   TotalBsmtSF   Heating   HeatingQC
## Min. : 0.00   Min. : 0.0   Min. : 0   Floor: 1   Ex:741
## 1st Qu.: 0.00   1st Qu.: 223.0   1st Qu.: 796   GasA :1427   Fa: 49
## Median : 0.00   Median : 479.0   Median : 992   GasW : 18   Gd:240
## Mean : 46.58   Mean : 567.4   Mean :1058   Grav : 7   Po: 1
## 3rd Qu.: 0.00   3rd Qu.: 808.0   3rd Qu.:1298   OthW : 2   TA:428
## Max. :1474.00   Max. :2336.0   Max. :6110   Wall : 4
##
## CentralAir   Electrical   X1stFlrSF   X2ndFlrSF   LowQualFinSF
## N: 95   FuseA: 94   Min. : 334   Min. : 0.0   Min. : 0.000
## Y:1364   FuseF: 27   1st Qu.: 882   1st Qu.: 0.0   1st Qu.: 0.000
##           FuseP: 3   Median :1088   Median : 0.0   Median : 0.000
##           Mix : 1   Mean :1163   Mean : 346.8   Mean : 5.848
##           SBrkr:1334   3rd Qu.:1392   3rd Qu.: 728.0   3rd Qu.: 0.000
##           Max. :4692   Max. :2065.0   Max. :572.000
##
## GrLivArea   BsmtFullBath   BsmtHalfBath   FullBath
## Min. : 334   Min. :0.0000   Min. :0.00000   Min. :0.000
## 1st Qu.:1129   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.000
## Median :1464   Median :0.0000   Median :0.00000   Median :2.000
## Mean :1516   Mean :0.4256   Mean :0.05757   Mean :1.565
## 3rd Qu.:1778   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:2.000
## Max. :5642   Max. :3.0000   Max. :2.00000   Max. :3.000
##
## HalfBath   BedroomAbvGr   KitchenAbvGr   KitchenQual
## Min. :0.0000   Min. :0.000   Min. :0.000   Ex:100
## 1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:1.000   Fa: 39
## Median :0.0000   Median :3.000   Median :1.000   Gd:585
## Mean :0.3825   Mean :2.866   Mean :1.047   TA:735

```

```

## 3rd Qu.:1.0000 3rd Qu.:3.000 3rd Qu.:1.000
## Max. :2.0000 Max. :8.000 Max. :3.000
##
## TotRmsAbvGrd Functional Fireplaces FireplaceQu GarageType
## Min. : 2.000 Maj1: 14 Min. :0.0000 Ex : 24 2Types : 6
## 1st Qu.: 5.000 Maj2: 5 1st Qu.:0.0000 Fa : 33 Attchd :870
## Median : 6.000 Min1: 31 Median :1.0000 Gd :380 Basment: 19
## Mean : 6.517 Min2: 34 Mean :0.6134 Po : 20 BuiltIn: 87
## 3rd Qu.: 7.000 Mod : 15 3rd Qu.:1.0000 TA :313 CarPort: 9
## Max. :14.000 Sev : 1 Max. :3.0000 NA's:689 Detchd :387
## Typ :1359 NA's : 81
## GarageYrBlt GarageFinish GarageCars GarageArea GarageQual
## Min. :1900 Fin :351 Min. :0.000 Min. : 0 Ex : 3
## 1st Qu.:1961 RFn :422 1st Qu.:1.000 1st Qu.: 333 Fa : 48
## Median :1980 Unf :605 Median :2.000 Median : 480 Gd : 14
## Mean :1978 NA's: 81 Mean :1.767 Mean : 473 Po : 3
## 3rd Qu.:2002 3rd Qu.:2.000 3rd Qu.: 576 TA :1310
## Max. :2010 Max. :4.000 Max. :1418 NA's: 81
## NA's :81
## GarageCond PavedDrive WoodDeckSF OpenPorchSF EnclosedPorch
## Ex : 2 N: 90 Min. : 0.00 Min. : 0.00 Min. : 0.00
## Fa : 35 P: 30 1st Qu.: 0.00 1st Qu.: 0.00 1st Qu.: 0.00
## Gd : 9 Y:1339 Median : 0.00 Median : 25.00 Median : 0.00
## Po : 7 Mean : 94.24 Mean : 46.69 Mean : 21.97
## TA :1325 3rd Qu.:168.00 3rd Qu.: 68.00 3rd Qu.: 0.00
## NA's: 81 Max. :857.00 Max. :547.00 Max. :552.00
##
## X3SsnPorch ScreenPorch PoolArea PoolQC
## Min. : 0.000 Min. : 0.00 Min. : 0.000 Ex : 2
## 1st Qu.: 0.000 1st Qu.: 0.00 1st Qu.: 0.000 Fa : 2
## Median : 0.000 Median : 0.00 Median : 0.000 Gd : 3
## Mean : 3.412 Mean : 15.07 Mean : 2.761 NA's:1452
## 3rd Qu.: 0.000 3rd Qu.: 0.00 3rd Qu.: 0.000
## Max. :508.000 Max. :480.00 Max. :738.000
##
## Fence MiscFeature MiscVal MoSold
## GdPrv: 59 Gar2: 2 Min. : 0.00 Min. : 1.000
## GdWo : 54 Othr: 2 1st Qu.: 0.00 1st Qu.: 5.000
## MnPrv: 157 Shed: 49 Median : 0.00 Median : 6.000
## MnWw : 11 TenC: 1 Mean : 43.52 Mean : 6.323
## NA's :1178 NA's:1405 3rd Qu.: 0.00 3rd Qu.: 8.000
## Max. :15500.00 Max. :12.000
##
## YrSold SaleType SaleCondition SalePrice
## Min. :2006 WD :1266 Abnorml: 101 Min. : 34900
## 1st Qu.:2007 New : 122 AdjLand: 4 1st Qu.:129950
## Median :2008 COD : 43 Alloca : 12 Median :163000
## Mean :2008 ConLD : 9 Family : 20 Mean :180930
## 3rd Qu.:2009 ConLI : 5 Normal :1197 3rd Qu.:214000
## Max. :2010 ConLw : 5 Partial: 125 Max. :755000
## (Other): 9

```

```
str(trainH)
```

```
## 'data.frame': 1459 obs. of 69 variables:
```

```

## $ Id      : int  1 2 3 4 5 6 7 8 9 10 ...
## $ MSSubClass : int  60 20 60 70 60 50 20 60 50 190 ...
## $ MSZoning  : Factor w/ 5 levels "C (all)","FV",...: 4 4 4 4 4 4 4 4 5 4 ...
## $ LotFrontage : int  65 80 68 60 84 85 75 NA 51 50 ...
## $ LotArea    : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
## $ Street     : Factor w/ 2 levels "Grvl","Pave": 2 2 2 2 2 2 2 2 2 ...
## $ LotShape   : Factor w/ 4 levels "IR1","IR2","IR3",...: 4 4 1 1 1 1 4 1 4 4 ...
## $ LandContour : Factor w/ 4 levels "Bnk","HLS","Low",...: 4 4 4 4 4 4 4 4 4 ...
## $ LotConfig  : Factor w/ 5 levels "Corner","CulDSac",...: 5 3 5 1 3 5 5 1 5 1 ...
## $ LandSlope  : Factor w/ 3 levels "Gtl","Mod","Sev": 1 1 1 1 1 1 1 1 1 ...
## $ Neighborhood : Factor w/ 25 levels "Blmngtn","Blueste",...: 6 25 6 7 14 12 21 17 18 4 ...
## $ BldgType   : Factor w/ 5 levels "1Fam","2fmCon",...: 1 1 1 1 1 1 1 1 2 ...
## $ HouseStyle : Factor w/ 8 levels "1.5Fin","1.5Unf",...: 6 3 6 6 6 1 3 6 1 2 ...
## $ OverallQual : int  7 6 7 7 8 5 8 7 7 5 ...
## $ OverallCond : int  5 8 5 5 5 5 5 6 5 6 ...
## $ YearBuilt   : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
## $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
## $ Foundation : Factor w/ 6 levels "BrkTil","CBlock",...: 3 2 3 1 3 6 3 2 1 1 ...
## $ BsmtQual    : Factor w/ 4 levels "Ex","Fa","Gd",...: 3 3 3 4 3 3 1 3 4 4 ...
## $ BsmtCond    : Factor w/ 4 levels "Fa","Gd","Po",...: 4 4 4 2 4 4 4 4 4 4 ...
## $ BsmtExposure : Factor w/ 4 levels "Av","Gd","Mn",...: 4 2 3 4 1 4 1 3 4 4 ...
## $ BsmtFinType1 : Factor w/ 6 levels "ALQ","BLQ","GLQ",...: 3 1 3 1 3 3 3 1 6 3 ...
## $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
## $ BsmtFinType2 : Factor w/ 6 levels "ALQ","BLQ","GLQ",...: 6 6 6 6 6 6 6 2 6 6 ...
## $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
## $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
## $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
## $ Heating     : Factor w/ 6 levels "Floor","GasA",...: 2 2 2 2 2 2 2 2 2 ...
## $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",...: 1 1 1 3 1 1 1 1 3 1 ...
## $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 ...
## $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",...: 5 5 5 5 5 5 5 5 2 5 ...
## $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
## $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
## $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
## $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
## $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
## $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
## $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
## $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
## $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
## $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
## $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",...: 3 4 3 3 3 4 3 4 4 4 ...
## $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
## $ Functional   : Factor w/ 7 levels "Maj1","Maj2",...: 7 7 7 7 7 7 7 3 7 ...
## $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
## $ FireplaceQu  : Factor w/ 5 levels "Ex","Fa","Gd",...: NA 5 5 3 5 NA 3 5 5 5 ...
## $ GarageType   : Factor w/ 6 levels "2Types","Attchd",...: 2 2 2 6 2 2 2 2 6 2 ...
## $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
## $ GarageFinish : Factor w/ 3 levels "Fin","Rfn","Unf": 2 2 2 3 2 3 2 2 3 2 ...
## $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
## $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
## $ GarageQual   : Factor w/ 5 levels "Ex","Fa","Gd",...: 5 5 5 5 5 5 5 5 2 3 ...
## $ GarageCond   : Factor w/ 5 levels "Ex","Fa","Gd",...: 5 5 5 5 5 5 5 5 5 5 ...
## $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...

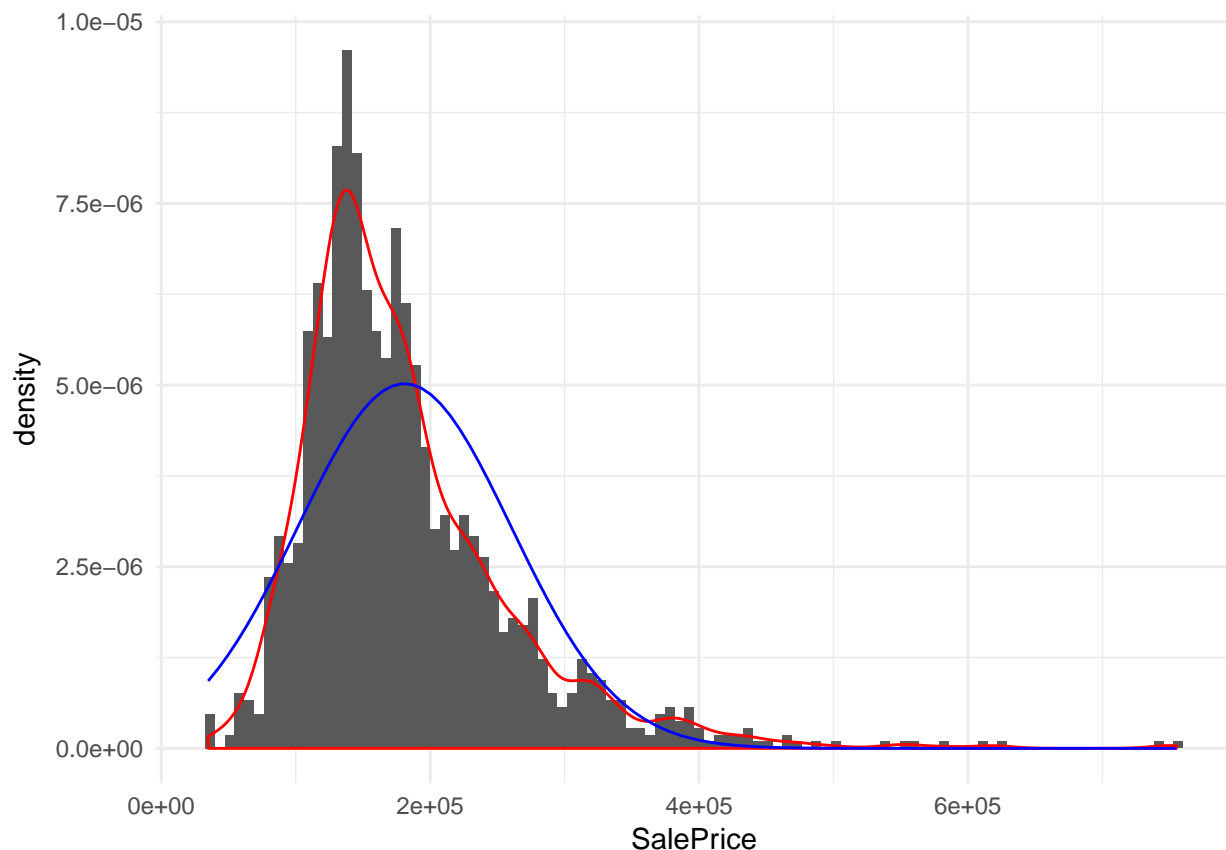
```

```
## $ WoodDeckSF : int  0 298 0 0 192 40 255 235 90 0 ...
## $ OpenPorchSF : int  61 0 42 35 84 30 57 204 0 4 ...
## $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
## $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
## $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
## $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ PoolQC       : Factor w/ 3 levels "Ex","Fa","Gd": NA NA NA NA NA NA NA NA NA NA ...
## $ Fence        : Factor w/ 4 levels "GdPrv","GdWo",...: NA NA NA NA NA 3 NA NA NA NA ...
## $ MiscFeature   : Factor w/ 4 levels "Gar2","Othr",...: NA NA NA NA NA 3 NA 3 NA NA ...
## $ MiscVal       : int  0 0 0 0 0 700 0 350 0 0 ...
## $ MoSold        : int  2 5 9 2 12 10 8 11 4 1 ...
## $ YrSold        : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
## $ SaleType      : Factor w/ 9 levels "COD","Con","ConLD",...: 9 9 9 9 9 9 9 9 9 ...
## $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",...: 5 5 5 1 5 5 5 5 1 5 ...
## $ SalePrice     : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

## What does the distribution of sale price look like?

Is the sale price (the variable we're interested in predicting) normally distributed? Find its mean, standard deviation, and plot a histogram of the distribution using ggplot2.

```
trainH %>% ggplot(., aes(x = SalePrice)) +
  geom_histogram(bins = 100, aes(y = ..density..)) +
  geom_density(col = "red") + theme_minimal() +
  stat_function(fun=dnorm, color="blue", args=list(mean=mean(trainH$SalePrice), sd=sd(trainH$SalePrice)))
```



```
# what is the mean?
mean(trainH$SalePrice)
```

```
## [1] 180930.4
```

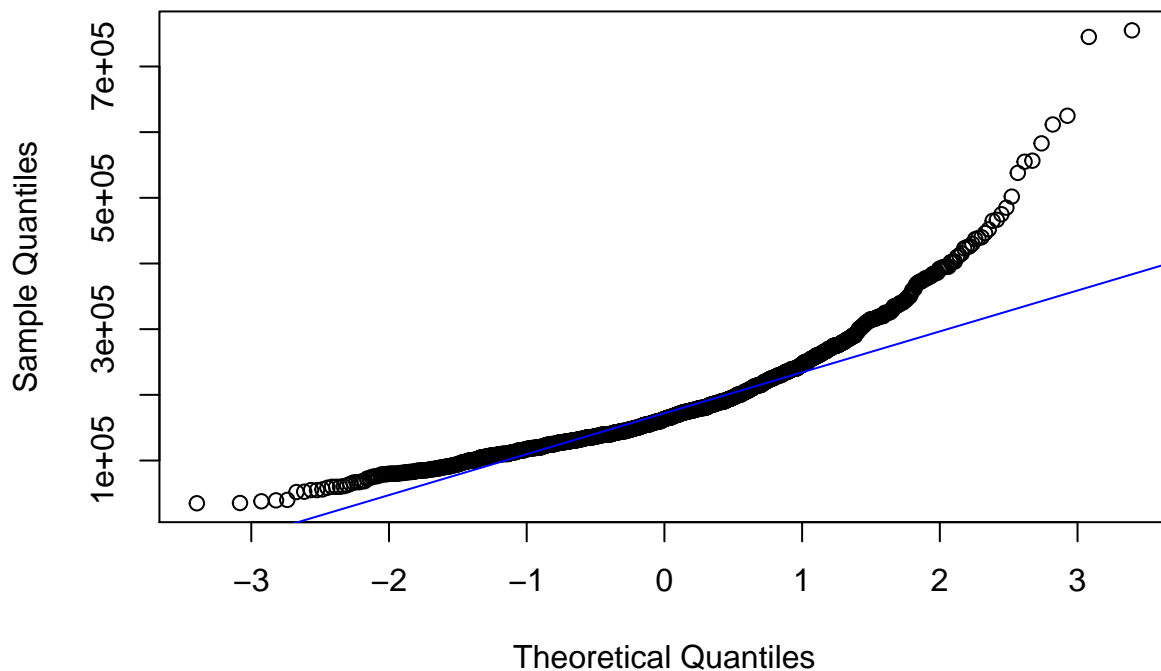
```
# what is the standard deviation?
sd(trainH$SalePrice)
```

```
## [1] 79468.96
```

Plot a quantile-quantile plot (QQ plot) to “assess” normality. This plot compared the data we have (Sample Quantiles) with a theoretical sample coming from a normal distribution. Each point (x, y) corresponds to one of the quantiles of the second distribution (x-coordinate, theoretical) plotted against the same quantile of the first distribution (y-coordinate, our data). Thus the line is a parametric curve with the parameter which is the number of the interval for the quantile.

```
qqnorm(trainH$SalePrice)
qqline(trainH$SalePrice, col = "blue")
```

## Normal Q-Q Plot



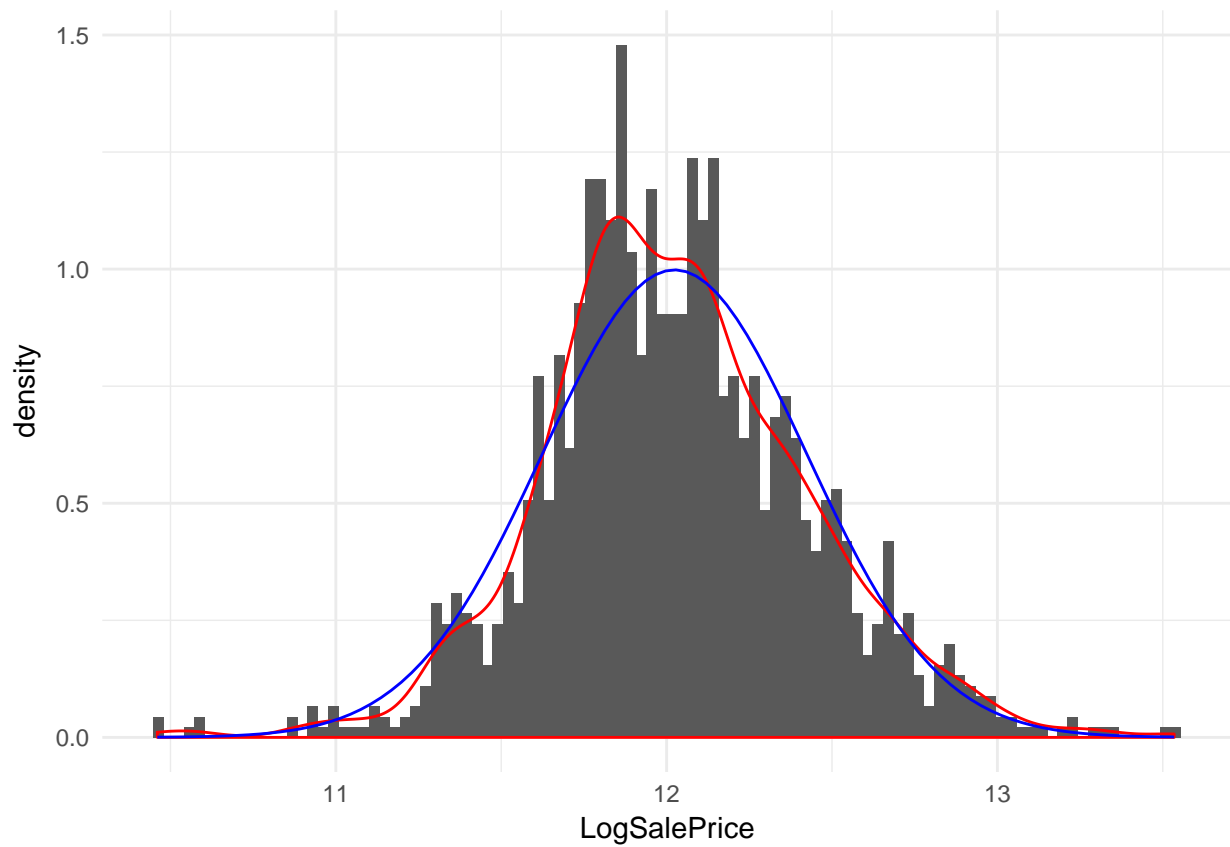
A standard way of transforming the data to be better approximated by a normal distribution is by using the log-transform?

Carry out this transformation and use a histogram and QQ plot to see whether it works...

```
trainH <- trainH %>%
  mutate(LogSalePrice = log(SalePrice + 1)) %>%
  mutate(SalePrice = NULL)
```

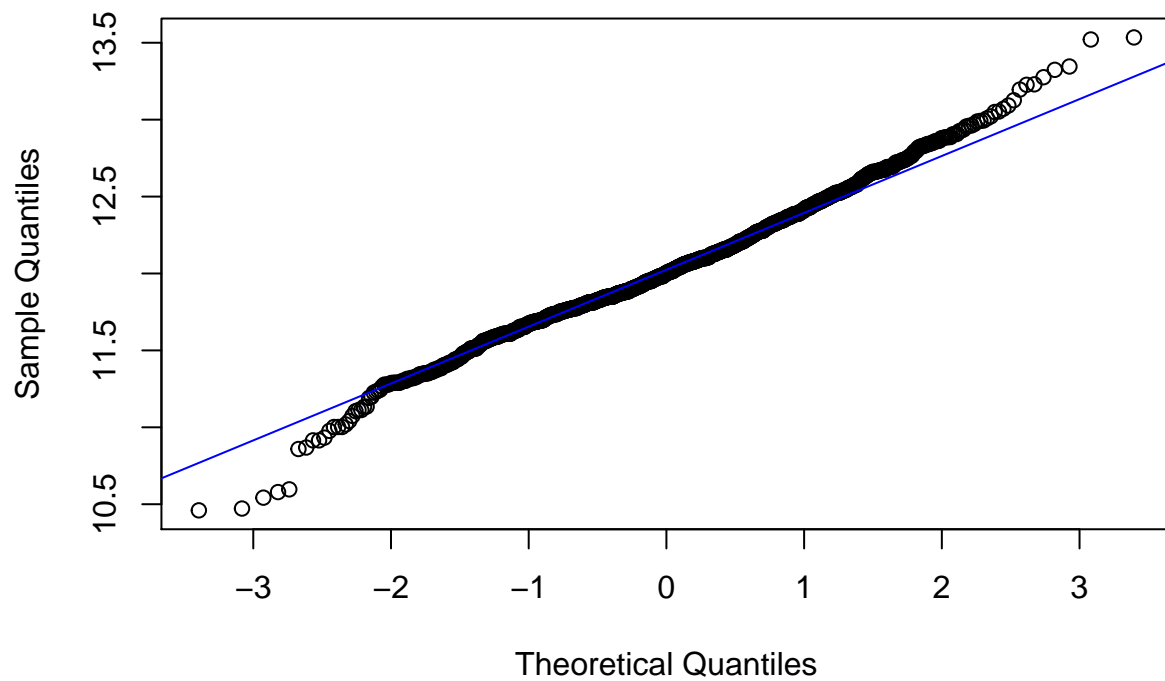
```
# plot
```

```
trainH %>% ggplot(., aes(x = LogSalePrice)) + geom_histogram(bins = 100, aes(y = ..density..)) + geom_d
```



```
qqnorm(trainH$LogSalePrice)
qqline(trainH$LogSalePrice, col = "blue")
```

**Normal Q-Q Plot**



## Missing data

What happens if we only use complete data? How much data is missing?

```
trainHcomplete <- trainH[complete.cases(trainH), ]
colSums(sapply(trainH, is.na)) [colSums(sapply(trainH, is.na)) > 0]
```

```
## LotFrontage      BsmtQual      BsmtCond BsmtExposure BsmtFinType1
##           259           37           37           38           37
## BsmtFinType2 FireplaceQu GarageType  GarageYrBlt GarageFinish
##           38           689           81           81           81
## GarageQual    GarageCond      PoolQC      Fence MiscFeature
##           81           81          1452          1178          1405
```

```
colSums(sapply(testH, is.na)) [colSums(sapply(testH, is.na)) > 0]
```

```
## LotFrontage      BsmtQual      BsmtCond BsmtExposure BsmtFinType1
##           226           39           40           39           37
## BsmtFinType2 FireplaceQu GarageType  GarageYrBlt GarageFinish
##           37           721           76           77           77
## GarageQual    GarageCond      PoolQC      Fence MiscFeature
##           77           77          1445          1160          1397
```

We need to combine the datasets for imputation, so that we don't have NAs in the test data as well!

```
trainH$source <- "train"
testH$source <- "test"
testH$LogSalePrice <- NA
alldata <- rbind(trainH, testH)
colSums(sapply(alldata, is.na)) [colSums(sapply(alldata, is.na)) > 0]
```

```
## LotFrontage      BsmtQual      BsmtCond BsmtExposure BsmtFinType1
##           485           76           77           77           74
## BsmtFinType2 FireplaceQu GarageType  GarageYrBlt GarageFinish
##           75          1410          157          158          158
## GarageQual    GarageCond      PoolQC      Fence MiscFeature
##           158          158          2897          2338          2802
## LogSalePrice
##           1448
```

How do we impute the missing data?

```
table(alldata$PoolQC)
```

```
##
## Ex Fa Gd
##  4  2  4
```

Read the metadata file and see that many of the NAs should be recoded as None since these features are lacking in the house.

```
alldata <- alldata %>%
  mutate(PoolQC = fct_explicit_na(PoolQC, na_level = "None")) %>%
  mutate(MiscFeature = fct_explicit_na(MiscFeature, na_level = "None")) %>%
  mutate(Fence = fct_explicit_na(Fence, na_level = "None")) %>%
  mutate(FireplaceQu = fct_explicit_na(FireplaceQu, na_level = "None")) %>%
  mutate(GarageType = fct_explicit_na(GarageType, na_level = "None")) %>%
  mutate(GarageFinish = fct_explicit_na(GarageFinish, na_level = "None")) %>%
  mutate(GarageQual = fct_explicit_na(GarageQual, na_level = "None")) %>%
```



```
mutate(GarageCond = fct_explicit_na(GarageCond, na_level = "None")) %>%
mutate(BsmtQual = fct_explicit_na(BsmtQual, na_level = "None")) %>%
mutate(BsmtCond = fct_explicit_na(BsmtCond, na_level = "None")) %>%
mutate(BsmtExposure = fct_explicit_na(BsmtExposure, na_level = "None")) %>%
mutate(BsmtFinType1 = fct_explicit_na(BsmtFinType1, na_level = "None")) %>%
mutate(BsmtFinType2 = fct_explicit_na(BsmtFinType2, na_level = "None"))

colSums(sapply(alldata, is.na)) [colSums(sapply(alldata, is.na)) > 0]
```

```
## LotFrontage GarageYrBlt LogSalePrice
##          485          158          1448
```

For the GarageYrBlt set to zero.

```
alldata <- alldata %>% replace_na(list(BsmtFinSF1 = 0, BsmtFinSF2 = 0, BsmtUnfSF = 0, TotalBsmtSF = 0,
colSums(sapply(alldata, is.na)) [colSums(sapply(alldata, is.na)) > 0]
```

```
## LotFrontage LogSalePrice
##          485          1448
```

Lot frontage - set as median for the neighborhood.

```
alldata <- alldata %>%
  group_by(Neighborhood) %>%
  mutate(LotFrontage=ifelse(is.na(LotFrontage), median(LotFrontage, na.rm=TRUE), LotFrontage))
```

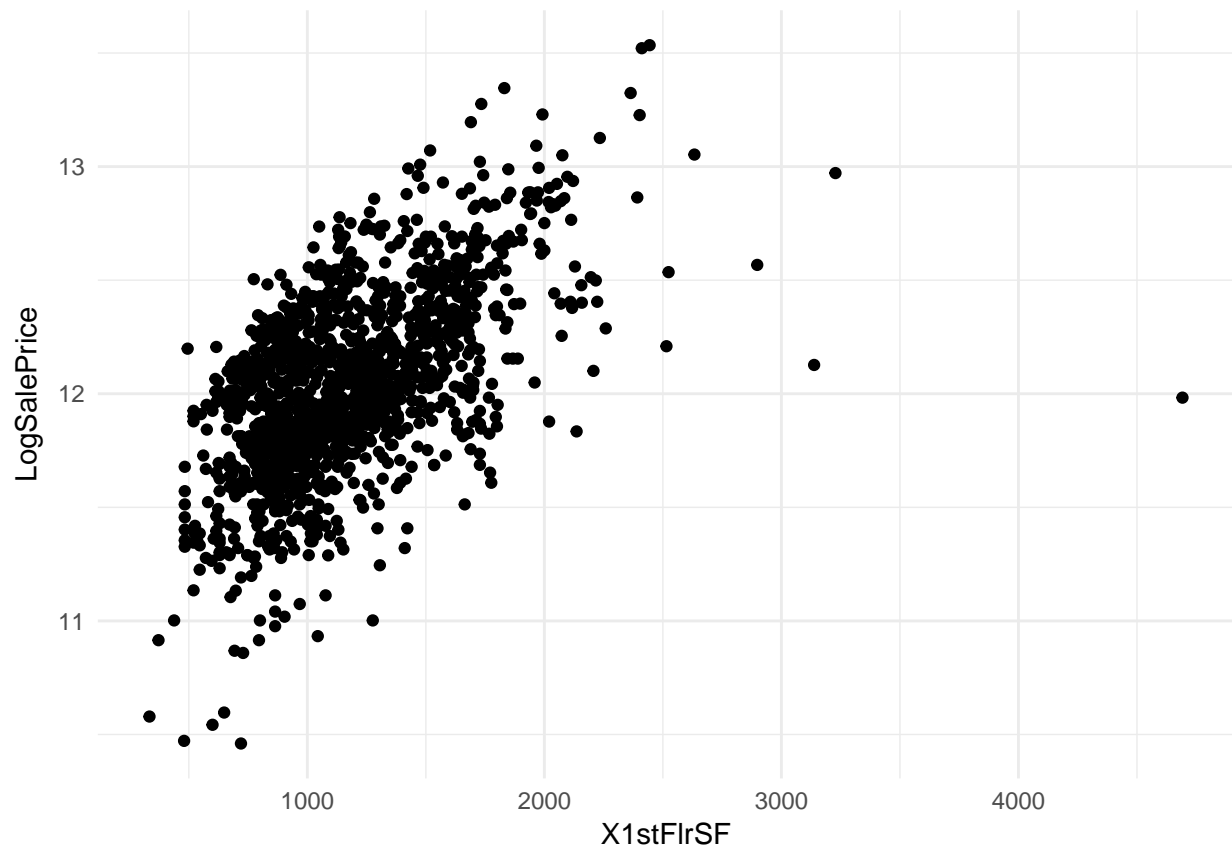
## Now split data again

```
trainHC <-alldata %>% filter(source == "train")
testHC <-alldata %>% filter(source == "test")
```

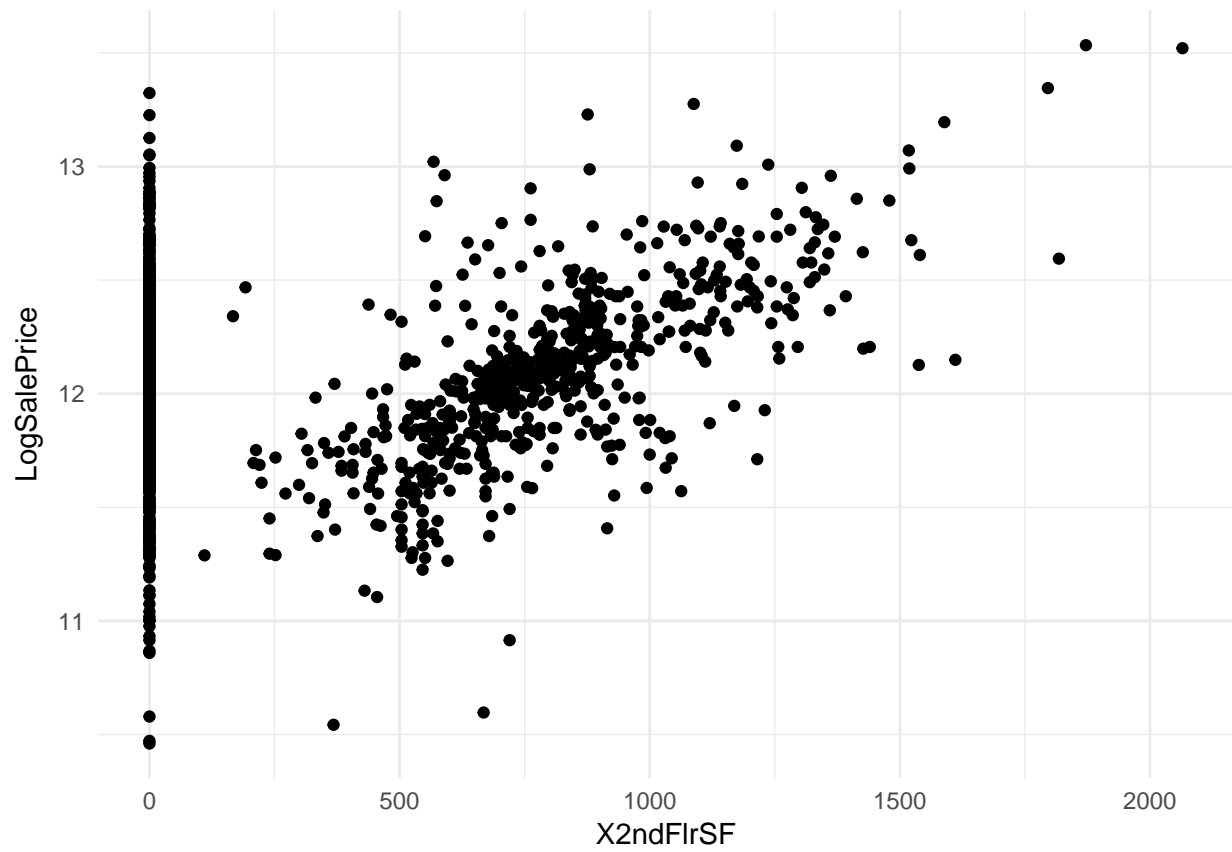
## Basic exploratory data analysis of training data

How does the sale price depend on living area: X1stFlrSF, X2ndFlrSF, TotalBsmtSF? Create a variable TotalSqFt which is a combination of these 3. Does it better predict the house price?

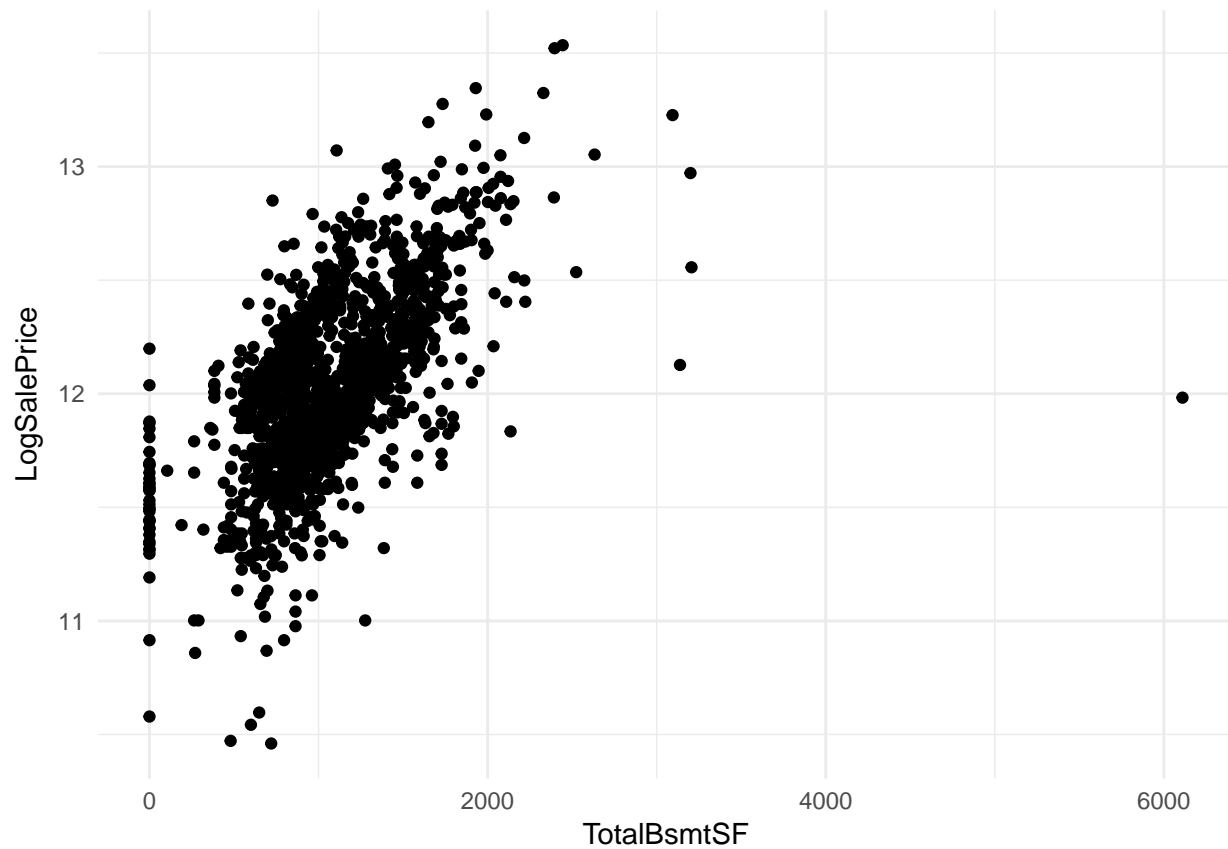
```
trainHC %>% ggplot(aes(x=X1stFlrSF, y = LogSalePrice)) + geom_point() + theme_minimal()
```



```
trainHC %>% ggplot(aes(x=X2ndFlrSF, y = LogSalePrice)) + geom_point() + theme_minimal()
```

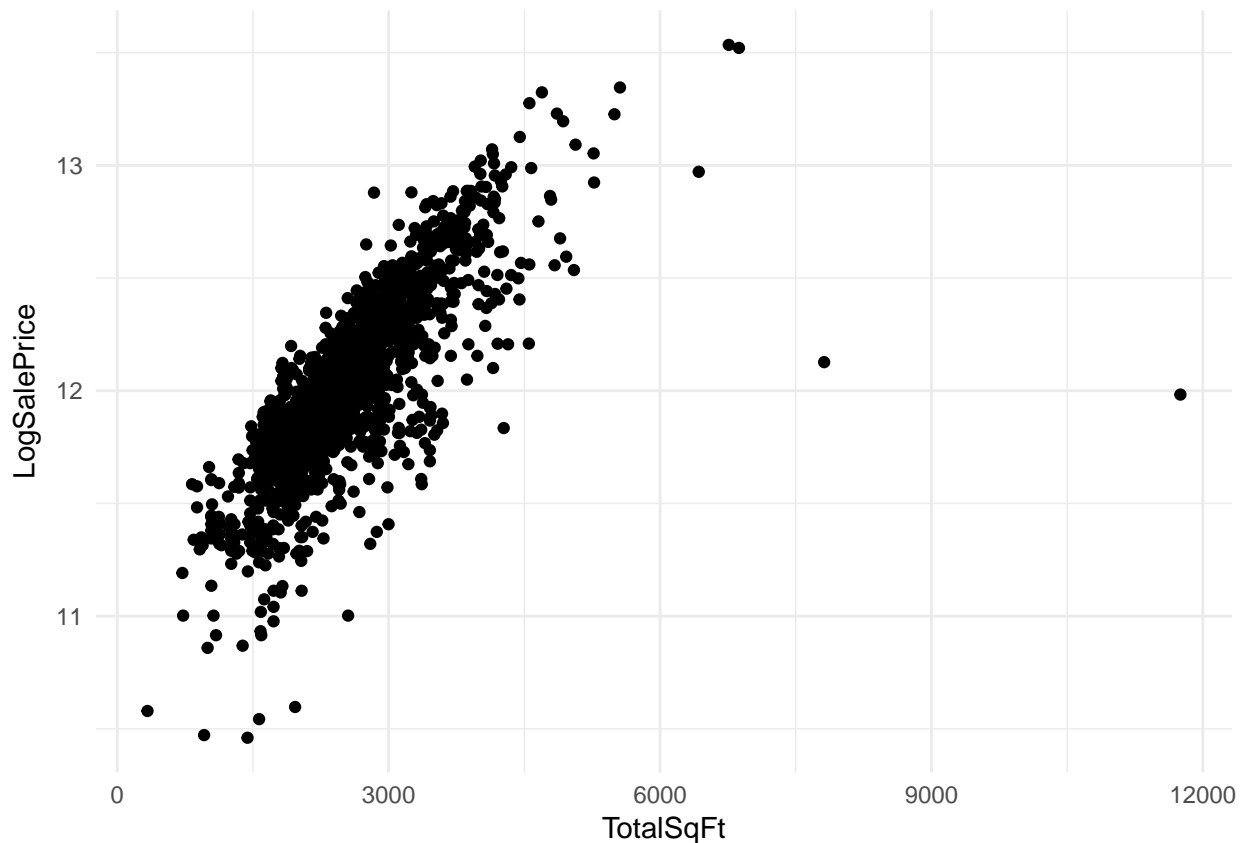


```
trainHC %>% ggplot(aes(x=TotalBsmtSF, y = LogSalePrice)) + geom_point() + theme_minimal()
```



```
# create extra variable
trainHC$TotalSqFt <- trainHC$X1stFlrSF + trainHC$X2ndFlrSF + trainHC$TotalBsmtSF

trainHC %>% ggplot(aes(x=TotalSqFt, y = LogSalePrice)) + geom_point() + theme_minimal()
```



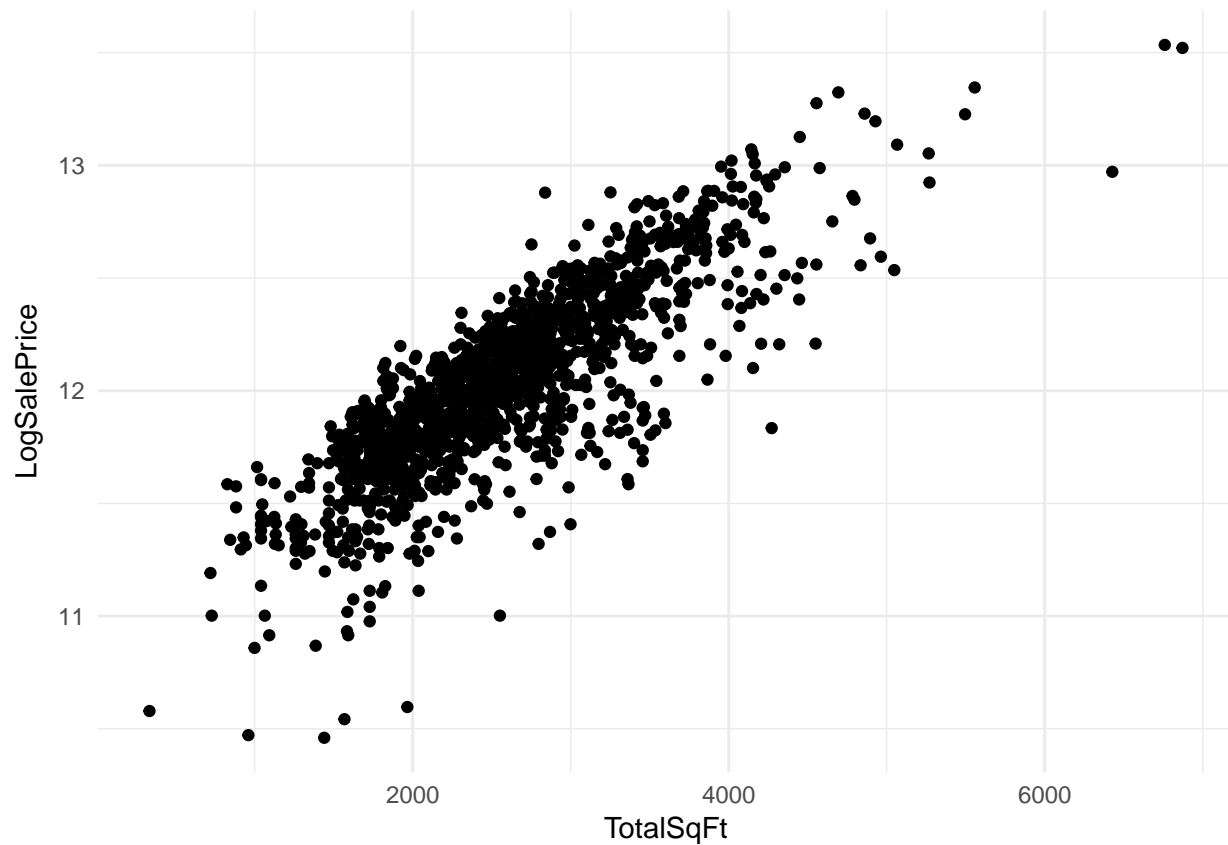
Identify and remove outliers with a high total square foot, but low price

```
# identify largest houses by area
trainHC %>% arrange(desc(TotalSqFt)) %>% select(Id, TotalSqFt)
```

```
## Adding missing grouping variables: `Neighborhood`
```

```
## # A tibble: 1,459 x 3
## # Groups:   Neighborhood [25]
##   Neighborhood    Id TotalSqFt
##   <fct>         <int>    <dbl>
## 1 Edwards       1299    11752.
## 2 Edwards        524     7814.
## 3 NoRidge       1183     6872.
## 4 NoRidge        692     6760.
## 5 NoRidge        497     6428.
## 6 NoRidge       1170     5557.
## 7 NridgHt       441     5496.
## 8 NoRidge       1354     5271.
## 9 NoRidge       1374     5266.
## 10 NridgHt       799     5066.
## # ... with 1,449 more rows
```

```
# filter out based on size of top 2
trainHC <- trainHC %>% filter(TotalSqFt <= 7800)
# check that we've removed them
trainHC %>% ggplot(aes(x=TotalSqFt, y = LogSalePrice)) + geom_point() + theme_minimal()
```

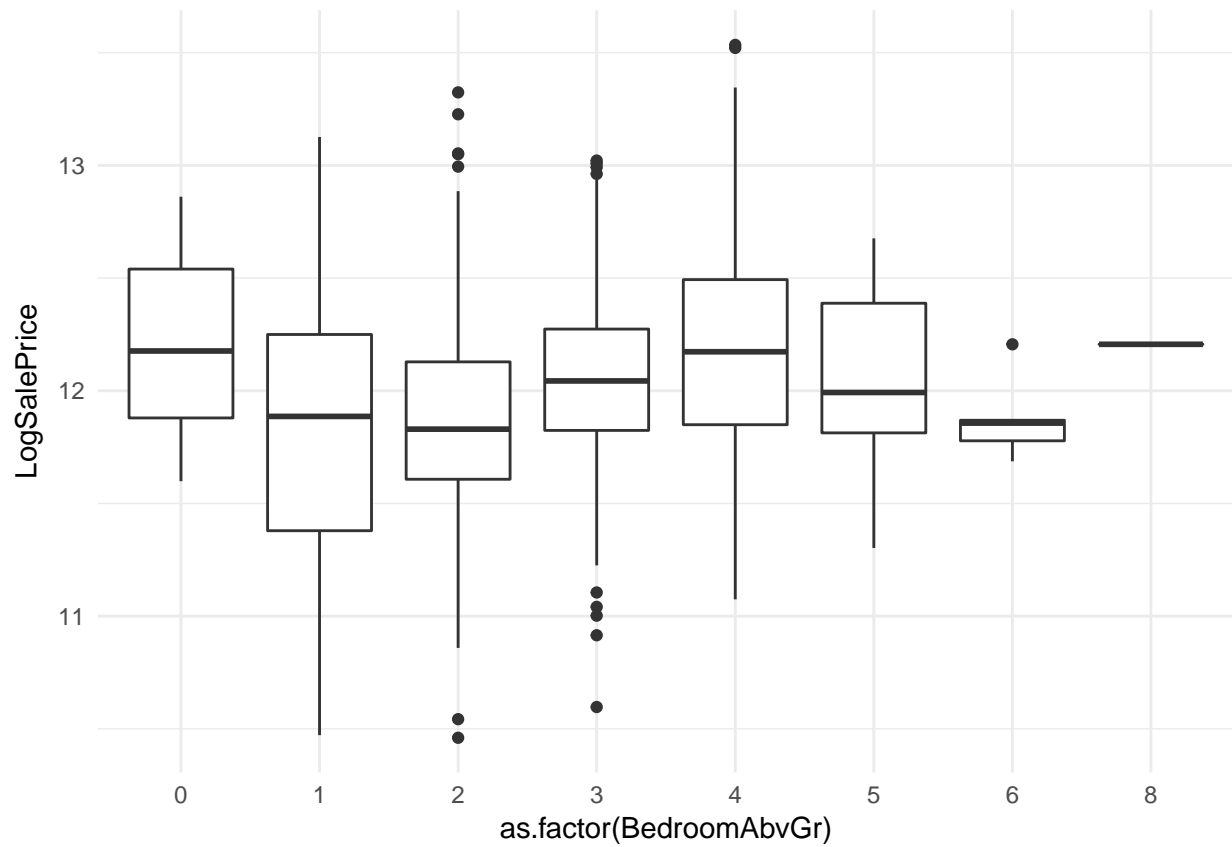


Does having more bedrooms increase sale price?

```
trainHC$BedroomAbvGr %>% summary()
```

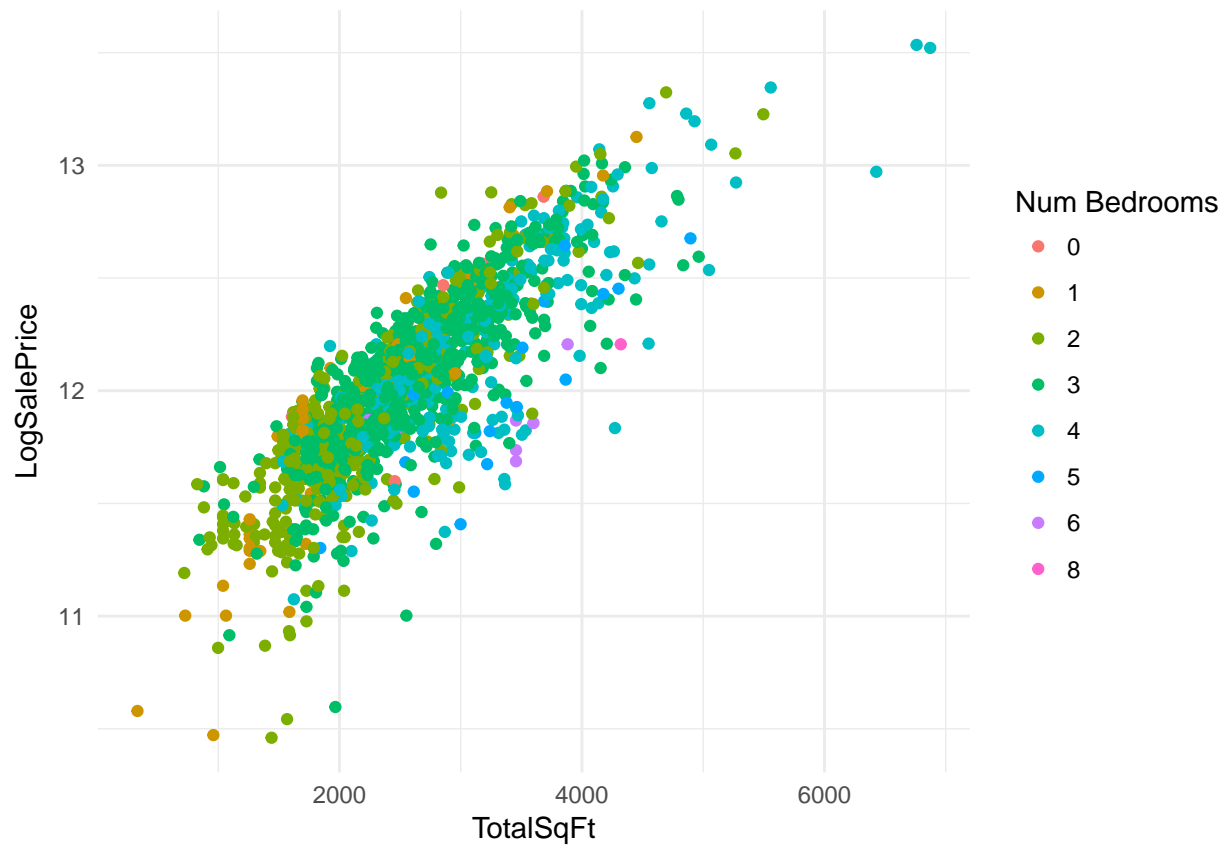
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.000   2.000   3.000   2.866   3.000   8.000
```

```
trainHC %>% ggplot(aes(x=as.factor(BedroomAbvGr), y = LogSalePrice)) +
  geom_boxplot() + theme_minimal()
```



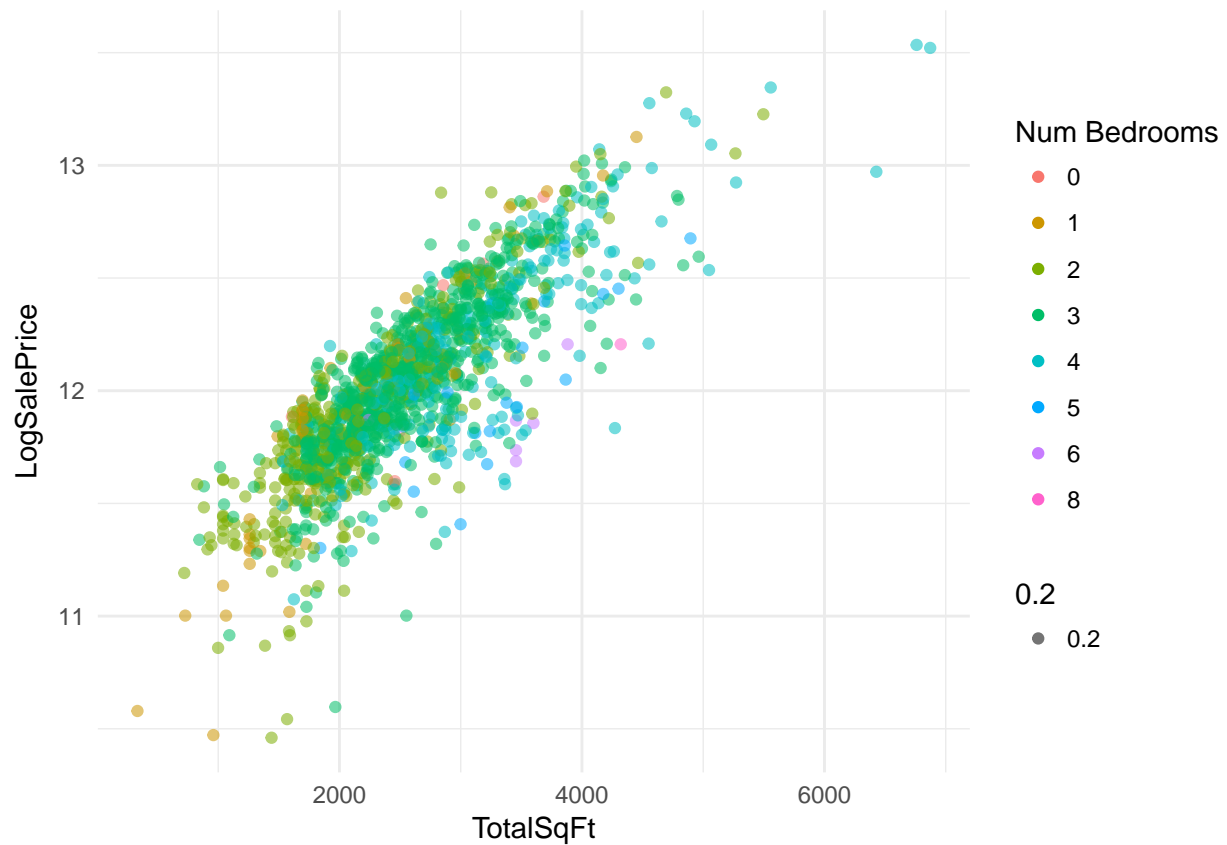
Visualise both number of bedrooms (as a factor) and TotalSqFt as a scatterplot to see if a trend is visible.

```
trainHC %>% ggplot(aes(x=TotalSqFt, y = LogSalePrice, colour = as.factor(BedroomAbvGr))) + geom_point()
```



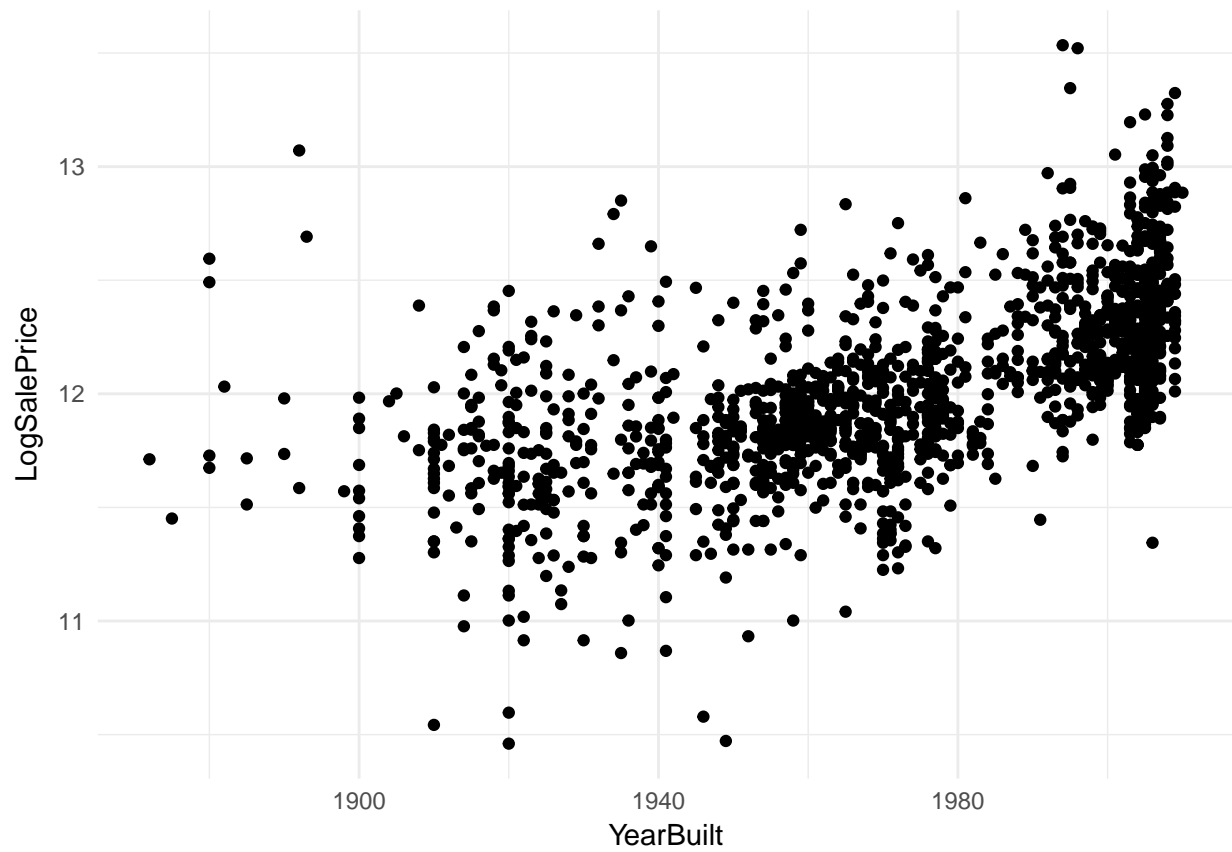
```
trainHC %>% ggplot(aes(x=TotalSqFt, y = LogSalePrice, colour = as.factor(BedroomAbvGr), alpha = 0.2)) +
```





Are newer or more recently renovated properties more expensive? Investigate this generally and then specifically for 2 - 4 bedroom properties.

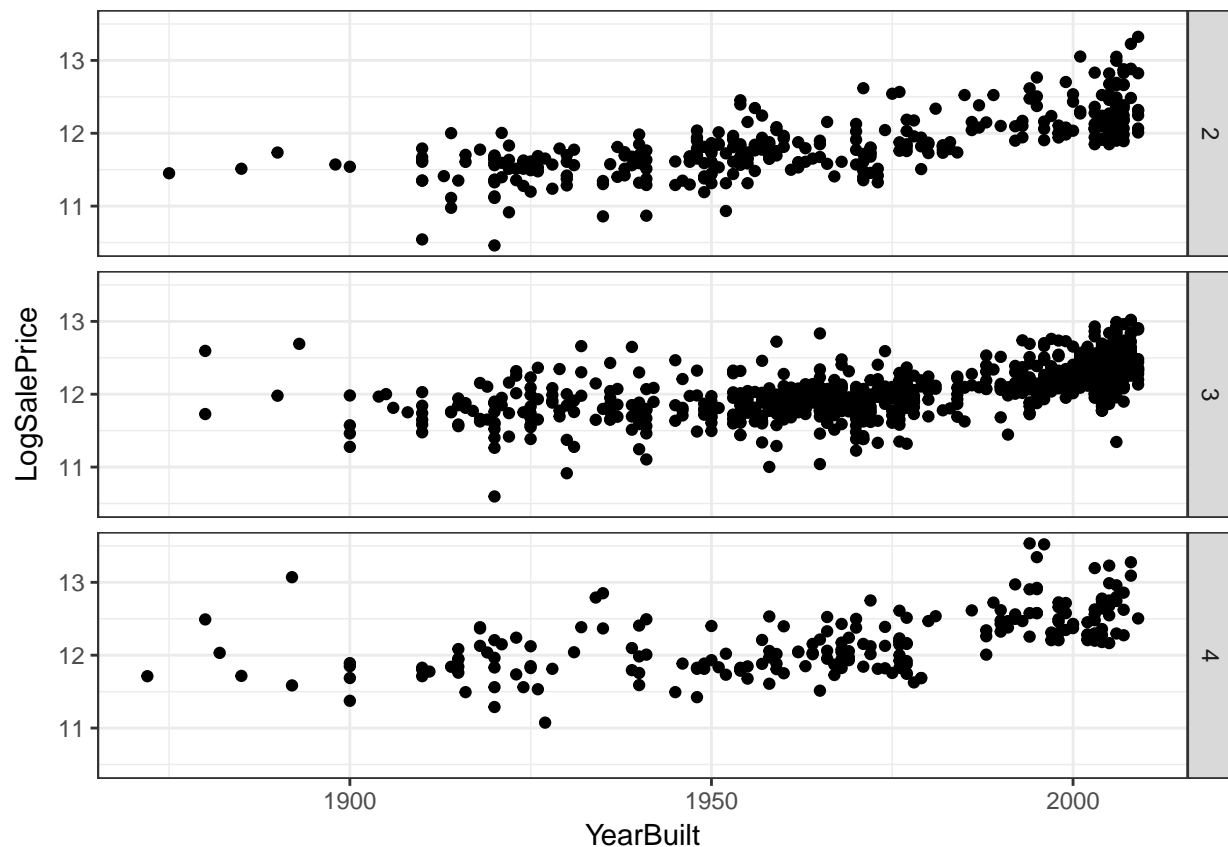
```
trainHC %>% ggplot(aes(x=YearBuilt, y = LogSalePrice)) + geom_point() + theme_minimal()
```



```
trainHC %>% ggplot(aes(x=YearRemodAdd, y = LogSalePrice)) + geom_point() + theme_minimal()
```



```
trainHC %>% filter(BedroomAbvGr >= 2) %>% filter(BedroomAbvGr <= 4) %>% ggplot(aes(x=YearBuilt, y = LogSalePrice))
```



Lets convert kitchen quality to numeric (we'll see why we need this later):

From the metadata we know it can be:

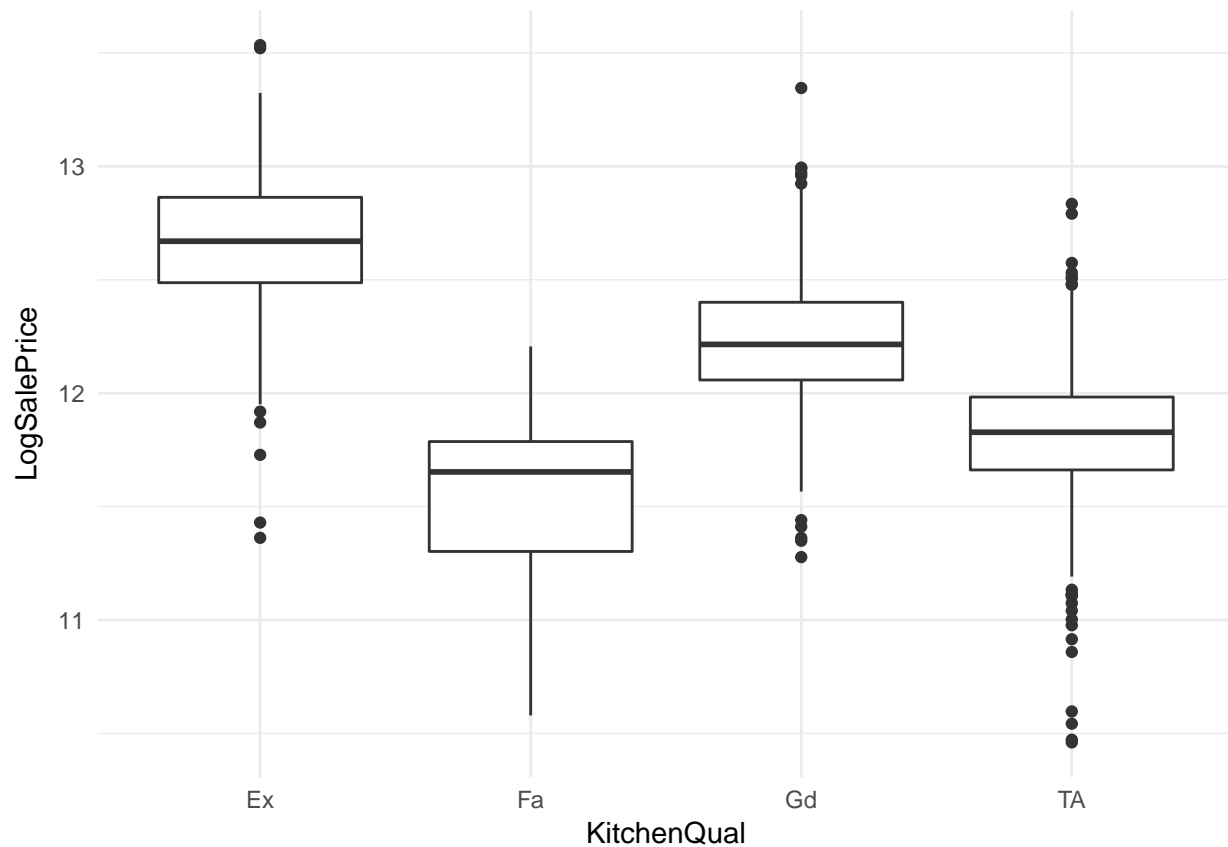
- Ex Excellent
- Gd Good
- TA Typical/Average
- Fa Fair
- Po Poor

Recode this to numeric values using mutate() and recode().

```
class(trainHC$KitchenQual)
```

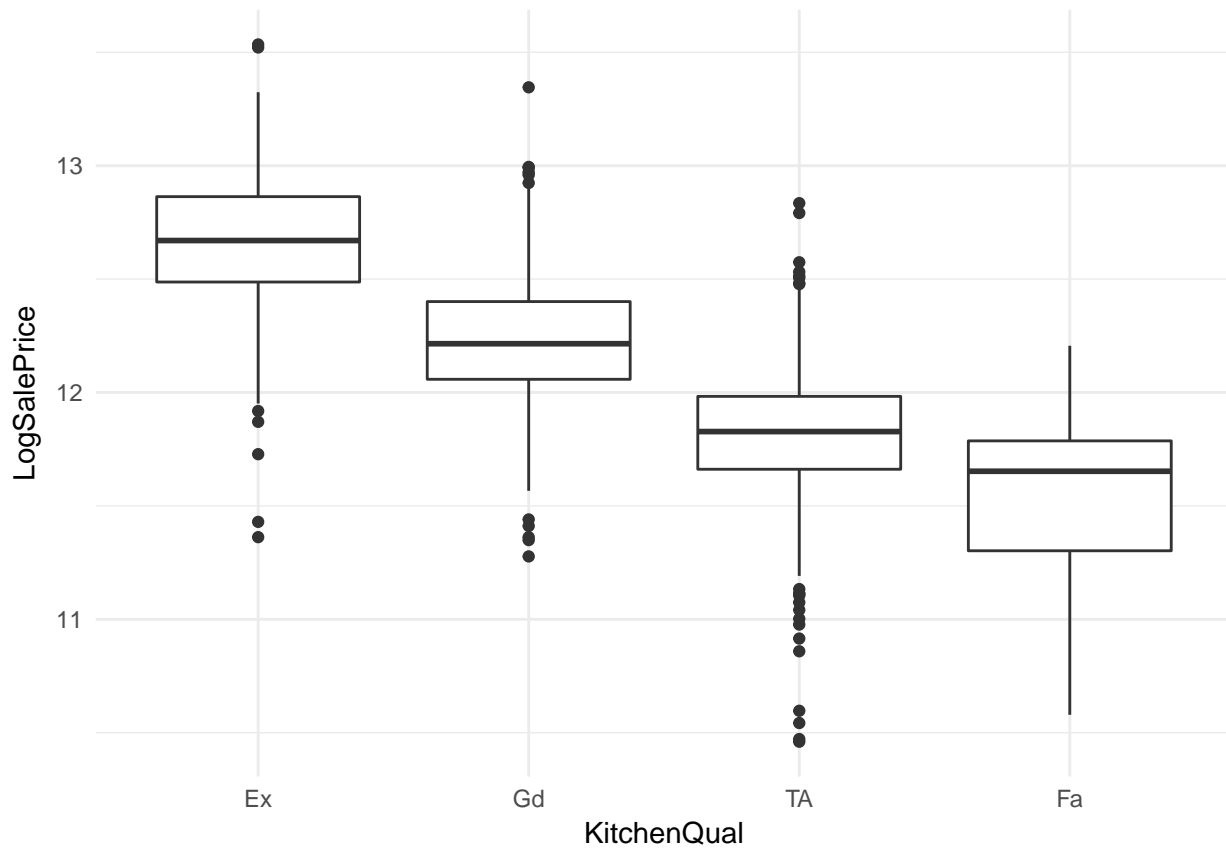
```
## [1] "factor"
```

```
trainHC %>% ggplot(aes(x = KitchenQual, y = LogSalePrice)) + geom_boxplot() + theme_minimal()
```



```
trainHC %>% mutate( KitchenQual = fct_relevel(KitchenQual, "Ex", "Gd", "TA", "Fa", "Po")) %>% ggplot(aes(
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
```

```
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
## Warning: Unknown levels in `f`: Po
```



```
trainHC <- trainHC %>% mutate( KitchenQual = dplyr::recode(KitchenQual, `Ex` = 5L, `Gd` = 4L, `TA` = 3L, `Fa` = 2L))
testHC <- testHC %>% mutate( KitchenQual = dplyr::recode(KitchenQual, `Ex` = 5L, `Gd` = 4L, `TA` = 3L, `Fa` = 2L))
```

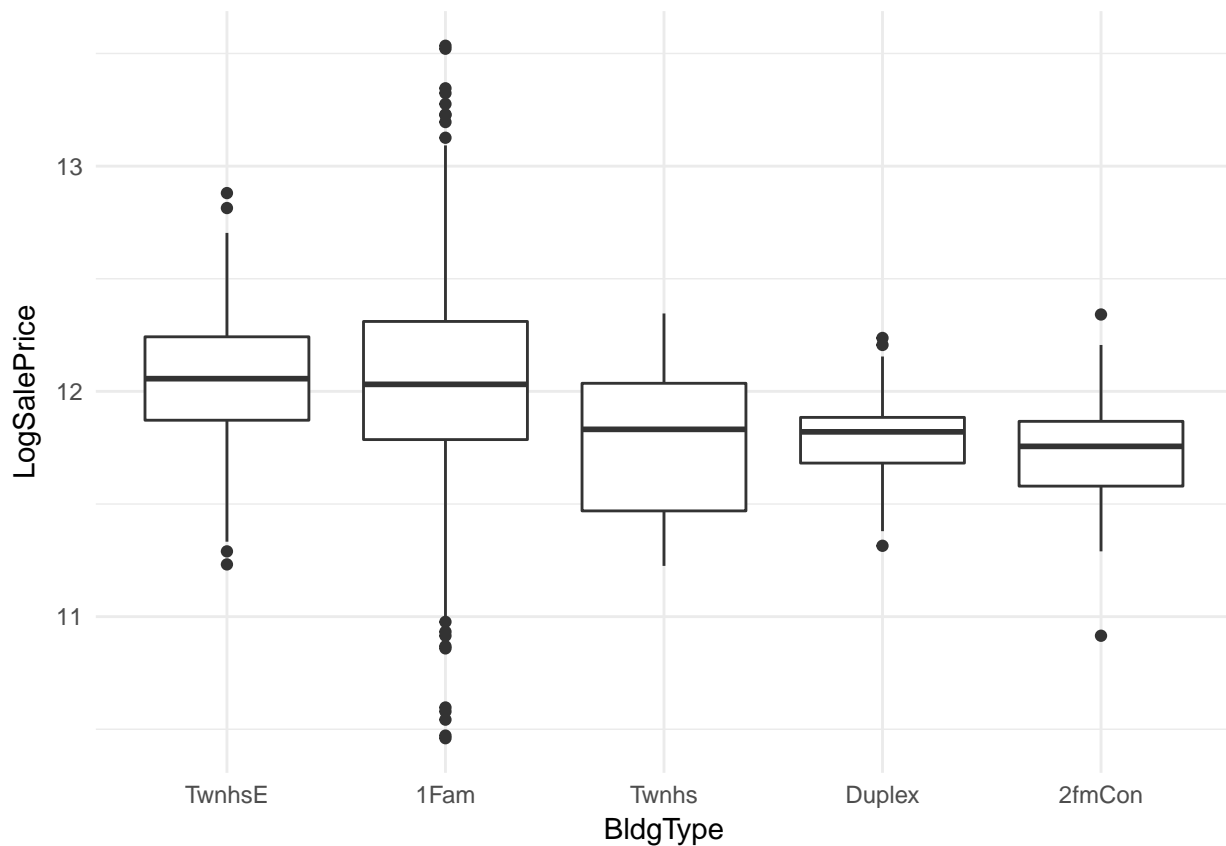
```
# %>% ggplot(aes(x = as.factor(KitchenQual), y = LogSalePrice)) + geom_boxplot() + theme_minimal()
```

Convert BldgType to numeric

```
trainHC %>% group_by(BldgType) %>%
  summarise( med = median(LogSalePrice)) %>%
  arrange(desc(med))
```

```
## # A tibble: 5 x 2
##   BldgType   med
##   <fct>     <dbl>
## 1 TwnhsE    12.1
## 2 1Fam      12.0
## 3 Twnhs     11.8
## 4 Duplex    11.8
## 5 2fmCon    11.8
```

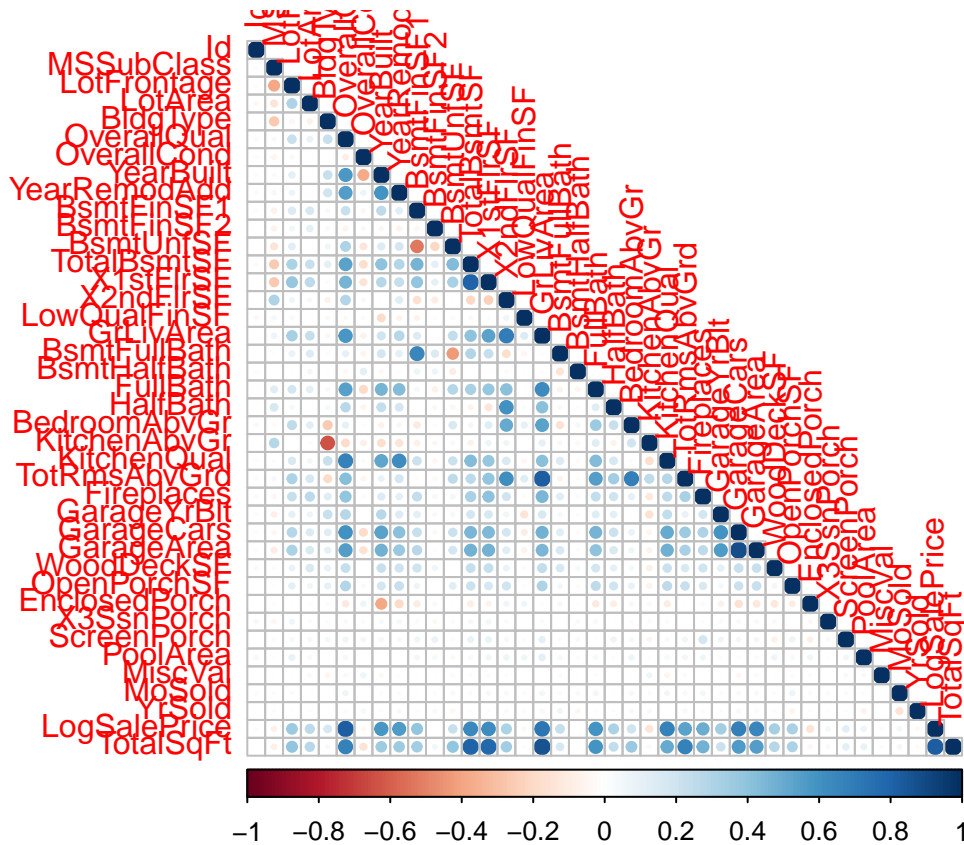
```
trainHC %>% mutate( BldgType = fct_relevel(BldgType, "TwnhsE", "1Fam", "Twnhs", "Duplex", "2fmCon")) %>%
```



```
trainHC <- trainHC %>% mutate( BldgType = dplyr::recode(BldgType, `TwnhsE` = 5L, `1Fam` = 4L, `Twnhs` = 3L, `Duplex` = 2L, `2fmCon` = 1L))
testHC <- testHC %>% mutate( BldgType = dplyr::recode(BldgType, `TwnhsE` = 5L, `1Fam` = 4L, `Twnhs` = 3L, `Duplex` = 2L, `2fmCon` = 1L))
```

What variables are correlated with each other and with price? Plot a correlation plot using corplot() for all numeric variables and those that show the top correlation with LogSalePrice.

```
trainHCnumeric <- trainHC[, sapply(trainHC, is.numeric)]
corplot(cor(trainHCnumeric, use="everything"), method="circle", type="lower", sig.level = 0.01, insig
```



```
correllationmatrix <- as.data.frame(cor(trainHCnumeric, use="everything"))
correllationmatrix$name <- row.names(correllationmatrix)
correllationmatrix %>% select(LogSalePrice, name) %>% arrange(desc(LogSalePrice))
```

##	LogSalePrice	name
## 1	1.000000000	LogSalePrice
## 2	0.825621563	TotalSqFt
## 3	0.821589445	OverallQual
## 4	0.725226325	GrLivArea
## 5	0.681053086	GarageCars
## 6	0.670110020	KitchenQual
## 7	0.656157276	GarageArea
## 8	0.648154025	TotalBsmtSF
## 9	0.620761376	X1stFlrSF
## 10	0.596021367	FullBath
## 11	0.587301350	YearBuilt
## 12	0.566207618	YearRemodAdd
## 13	0.537716260	TotRmsAbvGrd
## 14	0.492158735	Fireplaces
## 15	0.392429541	BsmtFinSF1
## 16	0.367707716	LotFrontage
## 17	0.349021245	GarageYrBlt
## 18	0.334250438	WoodDeckSF
## 19	0.325277237	OpenPorchSF
## 20	0.319997950	X2ndFlrSF
## 21	0.314338716	HalfBath
## 22	0.260545186	LotArea

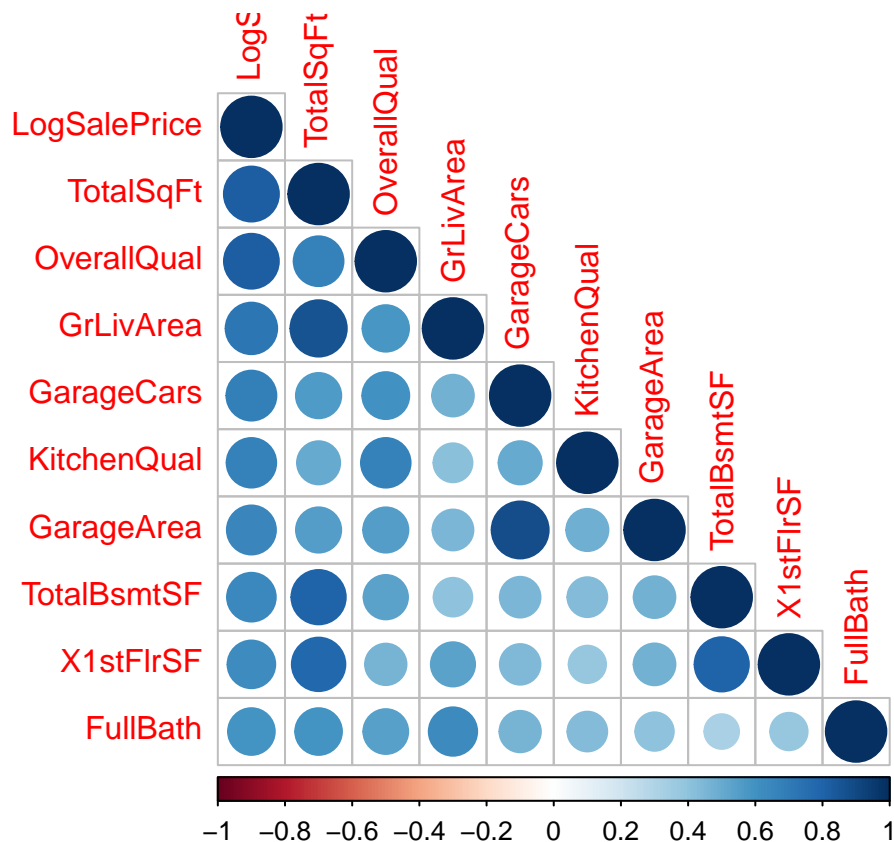


```
## 23 0.237160715 BsmtFullBath
## 24 0.221908923 BsmtUnfSF
## 25 0.209036056 BedroomAbvGr
## 26 0.176740414 BldgType
## 27 0.121250676 ScreenPorch
## 28 0.074338323 PoolArea
## 29 0.057073178 MoSold
## 30 0.054915665 X3SsnPorch
## 31 0.004865712 BsmtFinSF2
## 32 -0.005122225 BsmtHalfBath
## 33 -0.017801106 Id
## 34 -0.020011515 MiscVal
## 35 -0.036820651 OverallCond
## 36 -0.037152238 YrSold
## 37 -0.037950654 LowQualFinSF
## 38 -0.073981156 MSSubClass
## 39 -0.147534749 KitchenAbvGr
## 40 -0.149033033 EnclosedPorch
```

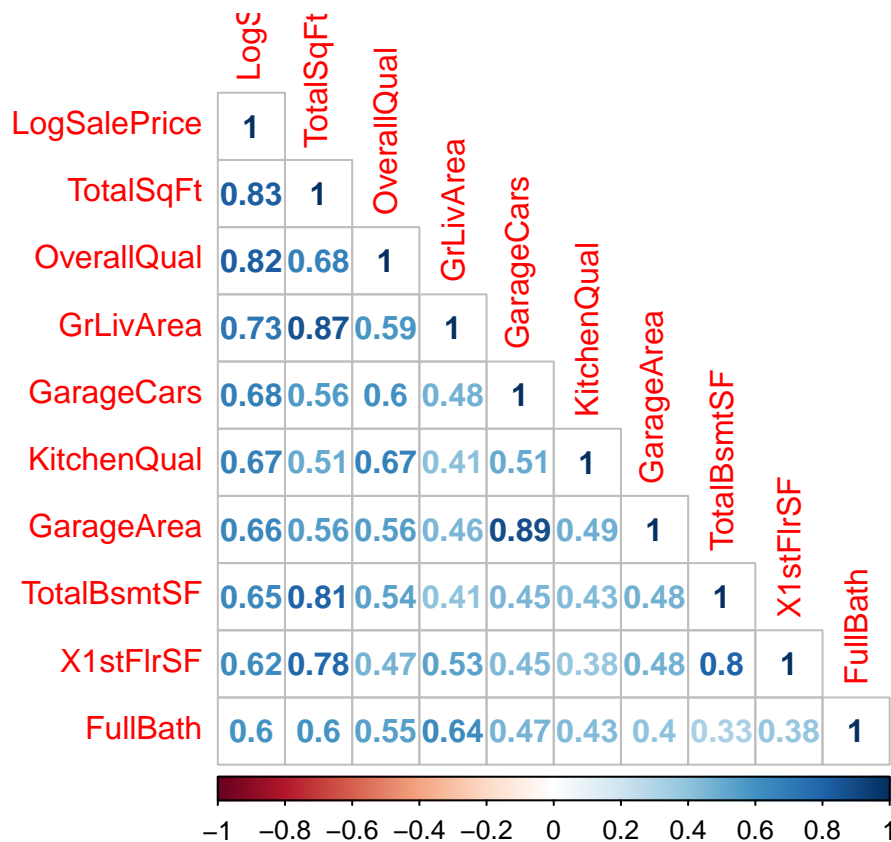
```
# take out the top 10 names
```

```
varscare <- correllationmatrix %>%
  select(LogSalePrice, name) %>%
  arrange(desc(LogSalePrice)) %>%
  head(n = 10L) %>%
  select(name)
```

```
corrplot(cor(trainHC[,varscare$name ], use="everything"), method="circle", type="lower", sig.level = 0
```



```
corrplot(cor(trainHC[,varscare$name ], use="everything"), method="number", type="lower", sig.level = 0
```



Use the createDataPartition() function to separate the training data into a training and testing subset. Allocate 50% of the data to each class. Run set.seed(12) before this.

```
set.seed(12)
partition <- createDataPartition(y = trainHC$LogSalePrice, p = 0.5, list=FALSE)
trainHC$source <- NULL
trainHCtrain <- trainHC[partition,]
trainHCtest <- trainHC[-partition,]
```

Fit a linear model considering the “top 10” correlated (top 9, ignore LogSalePrice for obvious reasons).

```
lm_model_top10 <- lm(LogSalePrice ~ TotalSqFt + OverallQual + GrLivArea + GarageCars + KitchenQual + GarageArea + TotalBsmtSF + X1stFlrSF + FullBath, data = trainHCtrain)
summary(lm_model_top10)
```

```
##
## Call:
## lm(formula = LogSalePrice ~ TotalSqFt + OverallQual + GrLivArea +
##      GarageCars + KitchenQual + GarageArea + TotalBsmtSF + X1stFlrSF +
##      FullBath, data = trainHCtrain)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.81658 -0.07012  0.01089  0.09591  0.48633
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.044e+01  3.425e-02 304.673  < 2e-16 ***
```

```
## TotalSqFt      2.049e-04  1.265e-04   1.620   0.1056
## OverallQual    1.030e-01  7.003e-03  14.704  < 2e-16 ***
## GrLivArea      2.571e-05  1.243e-04   0.207   0.8362
## GarageCars     4.530e-02  1.765e-02   2.566   0.0105 *
## KitchenQual    7.292e-02  1.214e-02   6.009  2.97e-09 ***
## GarageArea     1.115e-04  5.909e-05   1.887   0.0595 .
## TotalBsmntSF   -1.091e-05  1.302e-04  -0.084   0.9333
## X1stFlrSF      1.643e-05  3.096e-05   0.531   0.5958
## FullBath       1.508e-03  1.493e-02   0.101   0.9196
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1557 on 719 degrees of freedom
## Multiple R-squared:  0.8415, Adjusted R-squared:  0.8396
## F-statistic: 424.3 on 9 and 719 DF,  p-value: < 2.2e-16
```

Use `predict()` to predict house prices using our top10 model on the “test” portion of the training dataset. Use `rmse` to assess the root mean square error (our metric of accuracy).

```
prediction_lm10 <- predict(lm_model_top10, trainHCtest, type="response")
trainHCtest$lm10 <- prediction_lm10
rm(prediction_lm10)
# rmse?
rmse(trainHCtest$LogSalePrice, trainHCtest$lm10)
```

```
## [1] 0.1642187
```

Use `randomForest` to train a random forest model on all of the variables. Use `predict` and `rmse` to make the prediction and assess the accuracy respectively. Was a linear (on 9 features) or random forest model more accurate?

```
randFor <- randomForest(LogSalePrice ~ ., data=trainHCtrain)
# Predict using the test set
prediction_rf <- predict(randFor, trainHCtest)
trainHCtest$randFor <- prediction_rf
# rmse?
rmse(trainHCtest$LogSalePrice, trainHCtest$randFor)
```

```
## [1] 0.1426522
```

Use `xgboost` to predict house prices from numeric features of training dataset.

Use `xgb.plot.importance()` to assess which variables are most important for predicting house prices.

```
trainHCtrainNum <- as(as.matrix(trainHCtrain[, sapply(trainHCtrain, is.numeric)]), "sparseMatrix")
trainHCtestNum <- as(as.matrix(trainHCtest[, sapply(trainHCtest, is.numeric)]), "sparseMatrix")

trainD <- xgb.DMatrix(data = trainHCtrainNum, label = trainHCtrainNum[, "LogSalePrice"])

#Cross validate the model
cv.sparse <- xgb.cv(data = trainD,
                    nrounds = 600,
                    min_child_weight = 0,
                    max_depth = 10,
                    eta = 0.02,
                    subsample = .7,
                    colsample_bytree = .7,
                    booster = "gbtree",
```

```

        eval_metric = "rmse",
        verbose = TRUE,
        print_every_n = 50,
        nfold = 4,
        nthread = 2,
        objective="reg:linear")

## [1] train-rmse:11.298711+0.006494 test-rmse:11.298717+0.020349
## [51] train-rmse:4.141027+0.002037 test-rmse:4.140857+0.020967
## [101] train-rmse:1.532924+0.000511 test-rmse:1.533403+0.010594
## [151] train-rmse:0.582279+0.000587 test-rmse:0.590278+0.006681
## [201] train-rmse:0.236154+0.001185 test-rmse:0.262242+0.001627
## [251] train-rmse:0.108163+0.001541 test-rmse:0.165055+0.007037
## [301] train-rmse:0.057875+0.001270 test-rmse:0.141474+0.011346
## [351] train-rmse:0.035724+0.001017 test-rmse:0.135852+0.013019
## [401] train-rmse:0.024493+0.000872 test-rmse:0.134516+0.013661
## [451] train-rmse:0.017517+0.000861 test-rmse:0.134138+0.014040
## [501] train-rmse:0.012909+0.000822 test-rmse:0.133998+0.014113
## [551] train-rmse:0.009612+0.000696 test-rmse:0.134057+0.014240
## [600] train-rmse:0.007242+0.000589 test-rmse:0.134118+0.014283

#Train the model
#Choose the parameters for the model
param <- list(colsample_bytree = .7,
              subsample = .7,
              booster = "gbtree",
              max_depth = 10,
              eta = 0.02,
              eval_metric = "rmse",
              objective="reg:linear")

#Train the model using those parameters
bstSparse <-
  xgb.train(params = param,
            data = trainD,
            nrounds = 600,
            watchlist = list(train = trainD),
            verbose = TRUE,
            print_every_n = 50,
            nthread = 2)

## [1] train-rmse:11.298201
## [51] train-rmse:4.136751
## [101] train-rmse:1.527979
## [151] train-rmse:0.577598
## [201] train-rmse:0.233840
## [251] train-rmse:0.107654
## [301] train-rmse:0.059118
## [351] train-rmse:0.038267
## [401] train-rmse:0.027442
## [451] train-rmse:0.020410
## [501] train-rmse:0.015795
## [551] train-rmse:0.012201
## [600] train-rmse:0.009644

```

```

testD <- xgb.DMatrix(data = trainHCtestNum)

prediction <- predict(bstSparse, testD) #Make the prediction based on the half of the training data set

#Put testing prediction and test dataset all together

prediction <- as.data.frame(as.matrix(prediction))
colnames(prediction) <- "xgboost"
trainHCtest$xgboost <- prediction$xgboost

#Test with RMSE
rmse(trainHCtest$LogSalePrice, trainHCtest$xgboost)

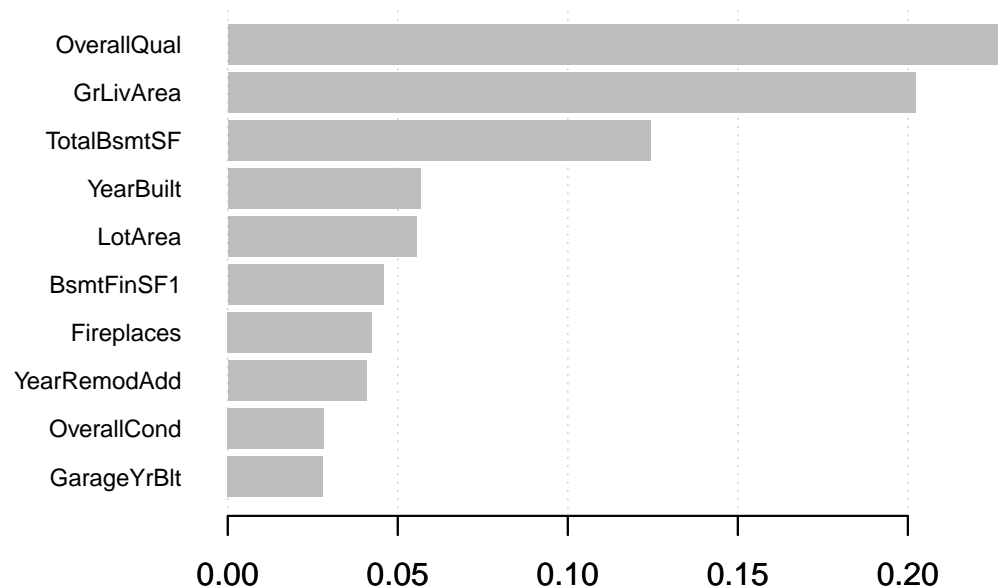
```

```
## [1] 0.1382971
```

```

# Feature importance
importance_matrix <- xgb.importance(dimnames(trainD)[[2]], model = bstSparse)
xgb.plot.importance(importance_matrix[1:10])

```



Use the glmnet library to train a ridge regression model. Is it more or less accurate than XGBoost?

```

trainHCtrainNumMatrix <- as.matrix(trainHCtrain[, sapply(trainHCtrain, is.numeric)])
trainHCtestNumMatrix <- as.matrix(trainHCtest[, sapply(trainHCtest, is.numeric)])
# cross validation for glmnet
glm.cv.ridge <- cv.glmnet(trainHCtrainNum[,c(1:38,40)], trainHCtrainNum[, "LogSalePrice"], alpha = 0)
penalty.ridge <- glm.cv.ridge$lambda.min
glm.ridge <- glmnet(x = trainHCtrainNum[,c(1:38,40)], y = trainHCtrainNum[, "LogSalePrice"], alpha = 0,
y_pred.ridge <- as.numeric(predict(glm.ridge, trainHCtestNum[,c(1:38,40)] ))
rmse(trainHCtest$LogSalePrice, y_pred.ridge)

```

```
## [1] 0.1373092
```