Stat4620_Project

Project Group 1

2024-11-20

```
library(ISLR)
library(pls)
```

```
##
## Attaching package: 'pls'

## The following object is masked from 'package:stats':
##
##     loadings
```

```
library(ggplot2)
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-8
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tidyr::expand() masks Matrix::expand()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x tidyr::pack()   masks Matrix::pack()
## x tidyr::unpack() masks Matrix::unpack()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(broom)
library(dplyr)
library(MASS)
```

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

```
library(corrplot)
```

```
## corrplot 0.95 loaded
##
## Attaching package: 'corrplot'
##
## The following object is masked from 'package:pls':
##
##     corrplot
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
##
## The following object is masked from 'package:pls':
##
##     R2
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
##
```

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

```
train_data = read.csv("train.csv")
test_data = read.csv("test_new.csv")
```

#Part I: Exploratory Data Analysis

The AMES Housing data set contains information regarding to house prices and the characteristics of them. Variables ranges from numerical and categorical types of property locations, rooms and house furnishings.

```
# Check missing values for each column
missing_counts <- colSums(is.na(train_data))
missing_features <- missing_counts[missing_counts > 0]
missing_features
```

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

There is one variable (LotFrontage) that contained a lot of actual missing values and thus we will drop it. We will also drop the ID column in the data set as it's used as an identifier and has no useful information. Upon analyzing the remaining missing features with NAs, we realized those NAs represent an actual category and are not missing data values, so we will keep them in the dataset for now.

```
train_data = train_data[, !(names(train_data) %in% c("Id", "LotFrontage"))]
```

We'll also drop categorical variables that don't provide a good split of the data space. Doing this will further simplify the number of features without losing any important patterns or information. Kaggle provides us a comprehensive view of the percentage break down of the buckets in the categorical variables. We'll drop variables that have buckets that exceed 85% of the observations.

```
train_data = train_data[, !(names(train_data) %in% c("Street", "Alley", "PoolQC", "MiscFeature", "LandC
```

We will then fill in the NAs for the remaining variables with missing values, replacing NAs in categorical variables with "None". There are two remaining continuous variables with missing values: GarageYrBuilt and MasVnrArea. For GarageYrBuilt, we will replace the NAs with the median value in that variable, but for "MasVnrArea", we will replace with the value 0 to correspond with the 8 missing values of categorical variable "MasVnrType".

```
summary(train_data)
```

```
##    MSSubClass      MSZoning            LotArea         LotShape
##  Min.   : 20.0   Length:1460        Min.   :  1300   Length:1460
##  1st Qu.: 20.0   Class :character   1st Qu.:  7554   Class :character
##  Median : 50.0   Mode  :character   Median :  9478   Mode  :character
##  Mean   : 56.9                      Mean   : 10517
##  3rd Qu.: 70.0                      3rd Qu.: 11602
##  Max.   :190.0                      Max.   :215245
##
##   LotConfig         Neighborhood         BldgType          HouseStyle
##  Length:1460        Length:1460        Length:1460        Length:1460
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   OverallQual      OverallCond      YearBuilt     YearRemodAdd
##  Min.   : 1.000   Min.   :1.000   Min.   :1872   Min.   :1950
##  1st Qu.: 5.000   1st Qu.:5.000   1st Qu.:1954   1st Qu.:1967
##  Median : 6.000   Median :5.000   Median :1973   Median :1994
##  Mean   : 6.099   Mean   :5.575   Mean   :1971   Mean   :1985
##  3rd Qu.: 7.000   3rd Qu.:6.000   3rd Qu.:2000   3rd Qu.:2004
##  Max.   :10.000   Max.   :9.000   Max.   :2010   Max.   :2010
##
##   RoofStyle         Exterior1st        Exterior2nd         MasVnrType
##  Length:1460        Length:1460        Length:1460        Length:1460
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    MasVnrArea      ExterQual          Foundation          BsmtQual
##  Min.   :   0.0   Length:1460        Length:1460        Length:1460
##  1st Qu.:   0.0   Class :character   Class :character   Class :character
##  Median :   0.0   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 103.7
##  3rd Qu.: 166.0
##  Max.   :1600.0
##  NA's   :8
##  BsmtExposure       BsmtFinType1         BsmtFinSF1       BsmtFinSF2
##  Length:1460        Length:1460        Min.   :   0.0   Min.   :   0.00
##  Class :character   Class :character   1st Qu.:   0.0   1st Qu.:   0.00
##  Mode  :character   Mode  :character   Median : 383.5   Median :   0.00
##                                        Mean   : 443.6   Mean   :  46.55
##                                        3rd Qu.: 712.2   3rd Qu.:   0.00
##                                        Max.   :5644.0   Max.   :1474.00
##
##    BsmtUnfSF       TotalBsmtSF      HeatingQC           X1stFlrSF
##  Min.   :   0.0   Min.   :   0.0   Length:1460        Min.   : 334
##  1st Qu.: 223.0   1st Qu.: 795.8   Class :character   1st Qu.: 882
```

```
##  Median : 477.5   Median : 991.5   Mode  :character   Median :1087
##  Mean   : 567.2   Mean   :1057.4                      Mean   :1163
##  3rd Qu.: 808.0   3rd Qu.:1298.2                      3rd Qu.:1391
##  Max.   :2336.0   Max.   :6110.0                      Max.   :4692
##
##    X2ndFlrSF     LowQualFinSF       GrLivArea     BsmtFullBath
##  Min.   :   0   Min.   :  0.000   Min.   : 334   Min.   :0.0000
##  1st Qu.:   0   1st Qu.:  0.000   1st Qu.:1130   1st Qu.:0.0000
##  Median :   0   Median :  0.000   Median :1464   Median :0.0000
##  Mean   : 347   Mean   :  5.845   Mean   :1515   Mean   :0.4253
##  3rd Qu.: 728   3rd Qu.:  0.000   3rd Qu.:1777   3rd Qu.:1.0000
##  Max.   :2065   Max.   :572.000   Max.   :5642   Max.   :3.0000
##
##   BsmtHalfBath        FullBath        HalfBath       BedroomAbvGr
##  Min.   :0.00000   Min.   :0.000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:2.000
##  Median :0.00000   Median :2.000   Median :0.0000   Median :3.000
##  Mean   :0.05753   Mean   :1.565   Mean   :0.3829   Mean   :2.866
##  3rd Qu.:0.00000   3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :2.00000   Max.   :3.000   Max.   :2.0000   Max.   :8.000
##
##   KitchenAbvGr   KitchenQual         TotRmsAbvGrd      Fireplaces
##  Min.   :0.000   Length:1460        Min.   : 2.000   Min.   :0.000
##  1st Qu.:1.000   Class :character   1st Qu.: 5.000   1st Qu.:0.000
##  Median :1.000   Mode  :character   Median : 6.000   Median :1.000
##  Mean   :1.047                      Mean   : 6.518   Mean   :0.613
##  3rd Qu.:1.000                      3rd Qu.: 7.000   3rd Qu.:1.000
##  Max.   :3.000                      Max.   :14.000   Max.   :3.000
##
##  FireplaceQu         GarageType         GarageYrBlt   GarageFinish
##  Length:1460        Length:1460        Min.   :1900   Length:1460
##  Class :character   Class :character   1st Qu.:1961   Class :character
##  Mode  :character   Mode  :character   Median :1980   Mode  :character
##                                        Mean   :1979
##                                        3rd Qu.:2002
##                                        Max.   :2010
##                                        NA's   :81
##    GarageCars      GarageArea       WoodDeckSF       OpenPorchSF
##  Min.   :0.000   Min.   :   0.0   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:1.000   1st Qu.: 334.5   1st Qu.:  0.00   1st Qu.:  0.00
##  Median :2.000   Median : 480.0   Median :  0.00   Median : 25.00
##  Mean   :1.767   Mean   : 473.0   Mean   : 94.24   Mean   : 46.66
##  3rd Qu.:2.000   3rd Qu.: 576.0   3rd Qu.:168.00   3rd Qu.: 68.00
##  Max.   :4.000   Max.   :1418.0   Max.   :857.00   Max.   :547.00
##
##  EnclosedPorch      X3SsnPorch      ScreenPorch        PoolArea
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   :  0.000
##  1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.000
##  Median :  0.00   Median :  0.00   Median :  0.00   Median :  0.000
##  Mean   : 21.95   Mean   :  3.41   Mean   : 15.06   Mean   :  2.759
##  3rd Qu.:  0.00   3rd Qu.:  0.00   3rd Qu.:  0.00   3rd Qu.:  0.000
##  Max.   :552.00   Max.   :508.00   Max.   :480.00   Max.   :738.000
##
##     Fence              MiscVal             MoSold           YrSold
```

```
##  Length:1460        Min.   :    0.00   Min.   : 1.000   Min.   :2006
##  Class :character   1st Qu.:    0.00   1st Qu.: 5.000   1st Qu.:2007
##  Mode  :character   Median :    0.00   Median : 6.000   Median :2008
##                     Mean   :   43.49   Mean   : 6.322   Mean   :2008
##                     3rd Qu.:    0.00   3rd Qu.: 8.000   3rd Qu.:2009
##                     Max.   :15500.00   Max.   :12.000   Max.   :2010
##
##  SaleCondition        SalePrice
##  Length:1460        Min.   : 34900
##  Class :character   1st Qu.:129975
##  Mode  :character   Median :163000
##                     Mean   :180921
##                     3rd Qu.:214000
##                     Max.   :755000
##
```

```
missing_counts <- colSums(is.na(train_data))
missing_features <- missing_counts[missing_counts > 0]
missing_features
```

```
##   MasVnrType   MasVnrArea     BsmtQual BsmtExposure BsmtFinType1  FireplaceQu
##            8            8           37           38           37          690
##   GarageType  GarageYrBlt GarageFinish        Fence
##           81           81           81         1179
```

```
median_value <- median(train_data$GarageYrBlt, na.rm = TRUE)
train_data$GarageYrBlt[is.na(train_data$GarageYrBlt)] <- median_value
train_data$MasVnrArea[is.na(train_data$MasVnrArea)] <- 0

train_data[is.na(train_data)] <- "None"

colSums(is.na(train_data))#there are now no NA's
```

```
##    MSSubClass      MSZoning       LotArea      LotShape     LotConfig
##             0             0             0             0             0
##  Neighborhood      BldgType    HouseStyle   OverallQual   OverallCond
##             0             0             0             0             0
##     YearBuilt  YearRemodAdd     RoofStyle   Exterior1st   Exterior2nd
##             0             0             0             0             0
##    MasVnrType    MasVnrArea     ExterQual    Foundation      BsmtQual
##             0             0             0             0             0
##  BsmtExposure  BsmtFinType1    BsmtFinSF1    BsmtFinSF2     BsmtUnfSF
##             0             0             0             0             0
##   TotalBsmtSF     HeatingQC     X1stFlrSF     X2ndFlrSF  LowQualFinSF
##             0             0             0             0             0
##     GrLivArea  BsmtFullBath  BsmtHalfBath      FullBath      HalfBath
##             0             0             0             0             0
##  BedroomAbvGr  KitchenAbvGr   KitchenQual  TotRmsAbvGrd    Fireplaces
##             0             0             0             0             0
##   FireplaceQu    GarageType   GarageYrBlt  GarageFinish    GarageCars
##             0             0             0             0             0
##    GarageArea    WoodDeckSF   OpenPorchSF EnclosedPorch    X3SsnPorch
##             0             0             0             0             0
```

```
##   ScreenPorch      PoolArea         Fence       MiscVal        MoSold
##             0             0             0             0             0
##        YrSold SaleCondition     SalePrice
##             0             0             0
```

Now we will look at the correlation matrix of our continuous predictors and address any irrelevant features.

```
# Correlation matrix for numeric features
train_data_numeric <- train_data[sapply(train_data, is.numeric)]
cor_matrix <- cor(train_data_numeric)

#subset(as.data.frame.table(cor_matrix), abs(Freq) < 1 & abs(Freq) > 0.75)

cor_sal <- cor_matrix[, "SalePrice"]
cor_sal
```

```
##    MSSubClass       LotArea   OverallQual   OverallCond     YearBuilt
##   -0.08428414    0.26384335    0.79098160   -0.07785589    0.52289733
##  YearRemodAdd    MasVnrArea    BsmtFinSF1    BsmtFinSF2     BsmtUnfSF
##    0.50710097    0.47261450    0.38641981   -0.01137812    0.21447911
##   TotalBsmtSF     X1stFlrSF     X2ndFlrSF  LowQualFinSF     GrLivArea
##    0.61358055    0.60585218    0.31933380   -0.02560613    0.70862448
##  BsmtFullBath  BsmtHalfBath      FullBath      HalfBath  BedroomAbvGr
##    0.22712223   -0.01684415    0.56066376    0.28410768    0.16821315
##  KitchenAbvGr  TotRmsAbvGrd    Fireplaces   GarageYrBlt    GarageCars
##   -0.13590737    0.53372316    0.46692884    0.46675365    0.64040920
##    GarageArea    WoodDeckSF   OpenPorchSF EnclosedPorch    X3SsnPorch
##    0.62343144    0.32441344    0.31585623   -0.12857796    0.04458367
##   ScreenPorch      PoolArea       MiscVal        MoSold        YrSold
##    0.11144657    0.09240355   -0.02118958    0.04643225   -0.02892259
##     SalePrice
##    1.00000000
```

```
# All variables not highly correlated with SalePrice
names(cor_sal[abs(cor_sal) < 0.5])
```

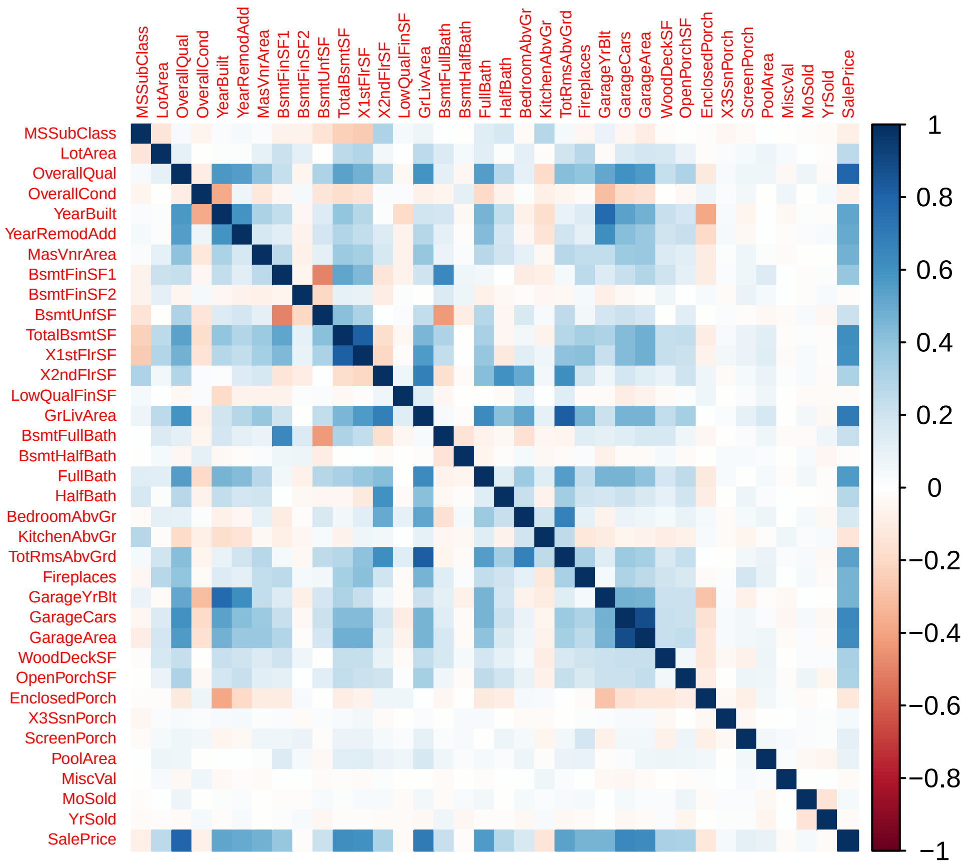
```
##  [1] "MSSubClass"    "LotArea"       "OverallCond"   "MasVnrArea"
##  [5] "BsmtFinSF1"    "BsmtFinSF2"    "BsmtUnfSF"     "X2ndFlrSF"
##  [9] "LowQualFinSF"  "BsmtFullBath"  "BsmtHalfBath"  "HalfBath"
## [13] "BedroomAbvGr"  "KitchenAbvGr"  "Fireplaces"    "GarageYrBlt"
## [17] "WoodDeckSF"    "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"
## [21] "ScreenPorch"   "PoolArea"      "MiscVal"       "MoSold"
## [25] "YrSold"
```

We will remove all the continuous variables that are not highly correlated with our response variable, SalePrice, based on the correlation matrix above. Those continuous variables with a correlation value higher than 0.5 or lower than -0.5 will remain in our dataset.

```
train_data = train_data[, !(names(train_data) %in% c("MSSubClass", "LotArea", "OverallCond", "BsmtFinSF
```

We will now look at the correlation between all predictor variables to see if there are any two that are highly correlated. If two of them are highly correlated then we will remove the one that is least correlated with the response variable.

```r
# Create correlation plot
corrplot(cor_matrix, method = "color", tl.cex = 0.5)
```




```r
# Print all relationships with 0.75 correlation or more
subset(as.data.frame.table(cor_matrix), abs(Freq) < 1 & abs(Freq) > 0.75)
```

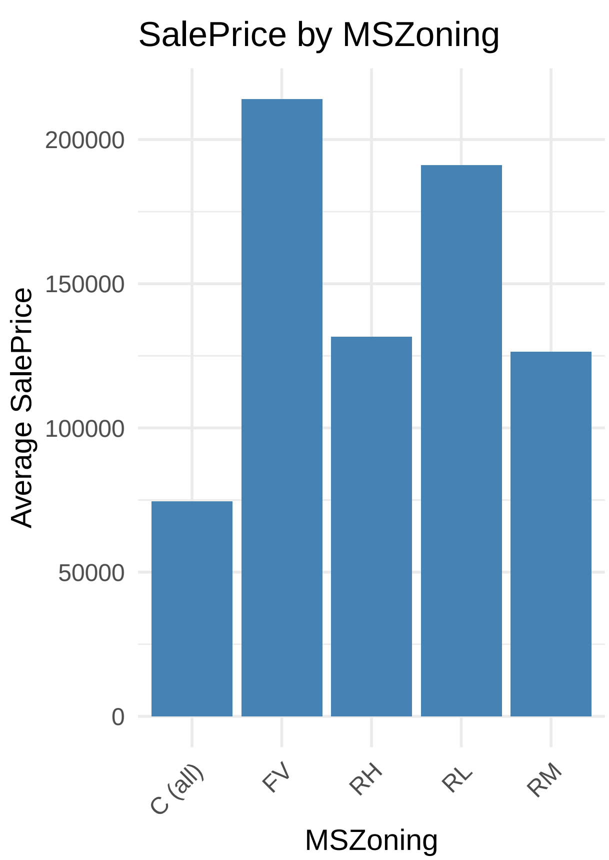
```
##              Var1        Var2      Freq
## 108     SalePrice OverallQual 0.7909816
## 168   GarageYrBlt   YearBuilt 0.7771818
## 372     X1stFlrSF TotalBsmtSF 0.8195300
## 407   TotalBsmtSF   X1stFlrSF 0.8195300
## 526  TotRmsAbvGrd   GrLivArea 0.8254894
## 771     GrLivArea TotRmsAbvGrd 0.8254894
## 833     YearBuilt GarageYrBlt 0.7771818
## 890    GarageArea  GarageCars 0.8824754
## 925    GarageCars  GarageArea 0.8824754
## 1263  OverallQual   SalePrice 0.7909816
```

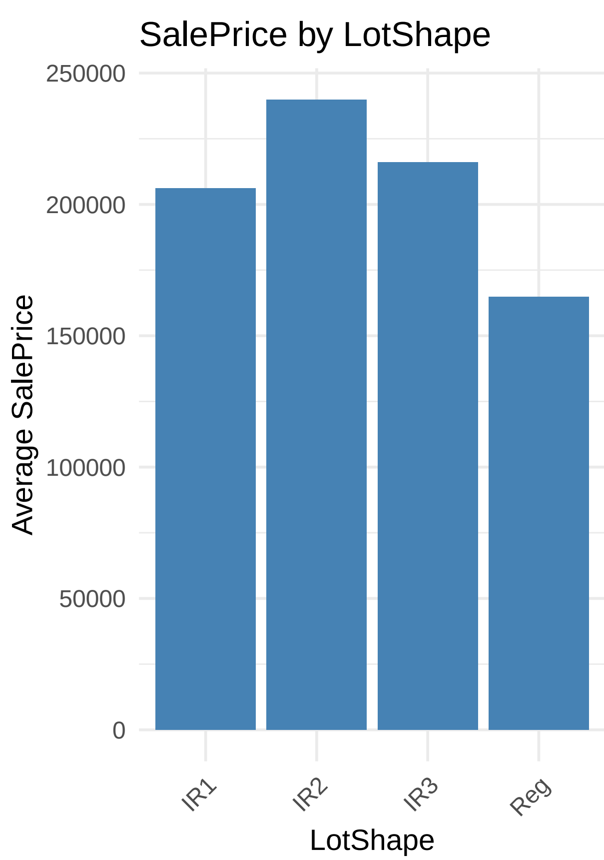
In the table above we can see that 4 of the predictor variables are highly correlated with another 4 variables so we will remove those, keeping the ones with higher correlation to the response.

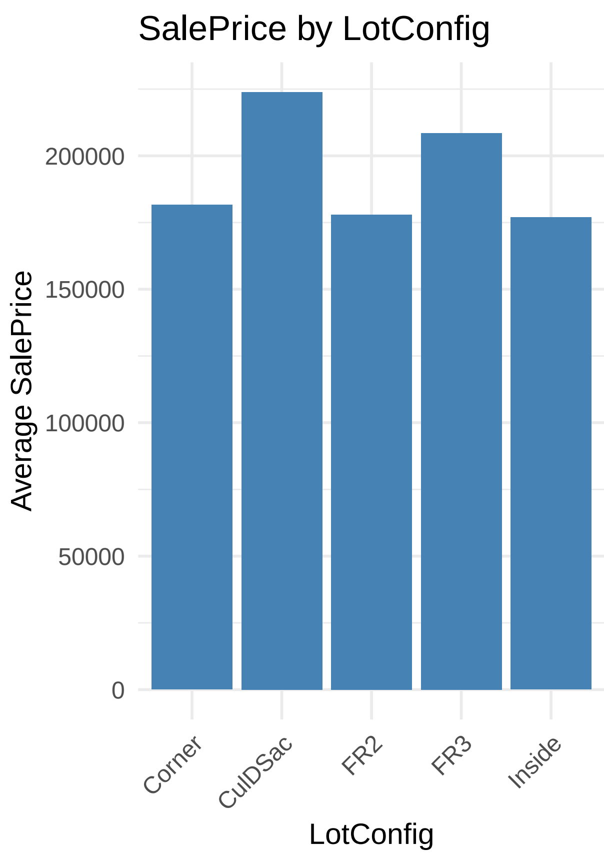
```r
# Remove variables due to multicollinearity
train_data = train_data[, !(names(train_data) %in% c("GarageYrBlt", "X1stFlrSF", "TotRmsAbvGrd", "Garag
```

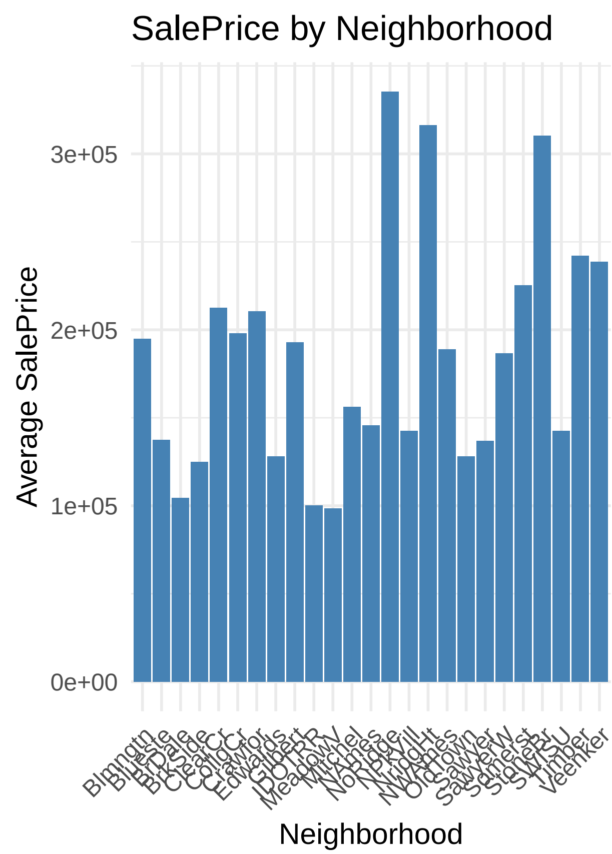
We will now look at all the categorical variables to see if they all have a unique distribution of SalePrice across different categories, deeming them useful.

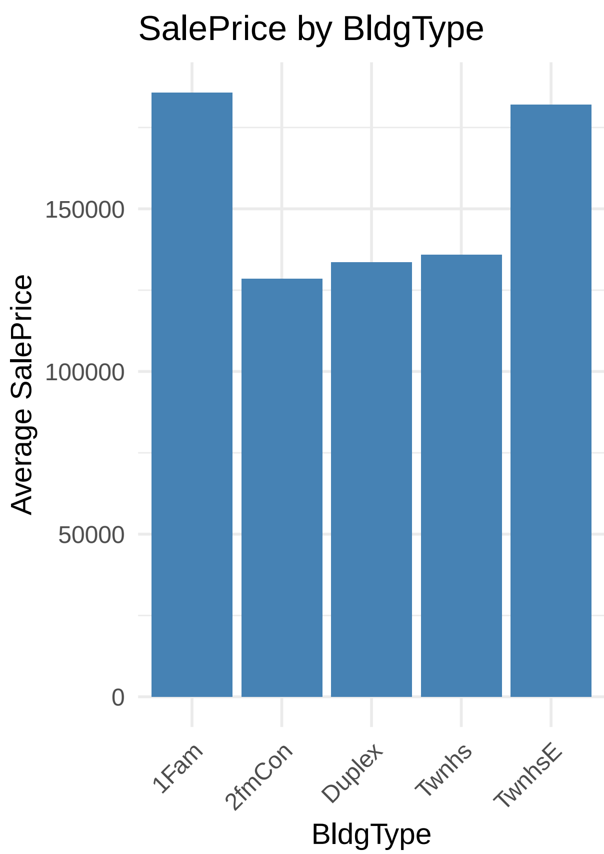
```
# List of all categorical variables
categorical_vars <- c("MSZoning", "LotShape", "LotConfig", "Neighborhood", "BldgType", "HouseStyle", "R

plot_list <- list()

# Loop through categorical variables and store plots in the list
for (var in categorical_vars) {
  x <- ggplot(train_data, aes_string(x = var, y = "SalePrice")) +
    geom_bar(stat = "summary", fun = "mean", fill = "steelblue") +
    labs(title = paste("SalePrice by", var),
         x = var, y = "Average SalePrice") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))

  # Add the plot to the list
  plot_list[[length(plot_list) + 1]] <- x
}

# Arrange and print the plots two at a time
for (i in seq(1, length(plot_list), by = 2)) {
  plots_to_print <- plot_list[i:min(i + 1, length(plot_list))]
  grid.arrange(grobs = plots_to_print, ncol = 2)
}
```

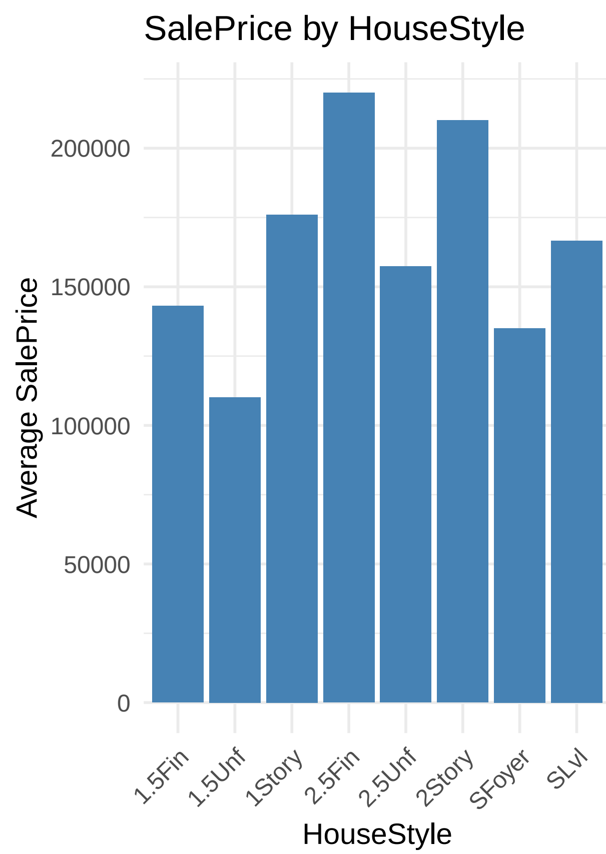


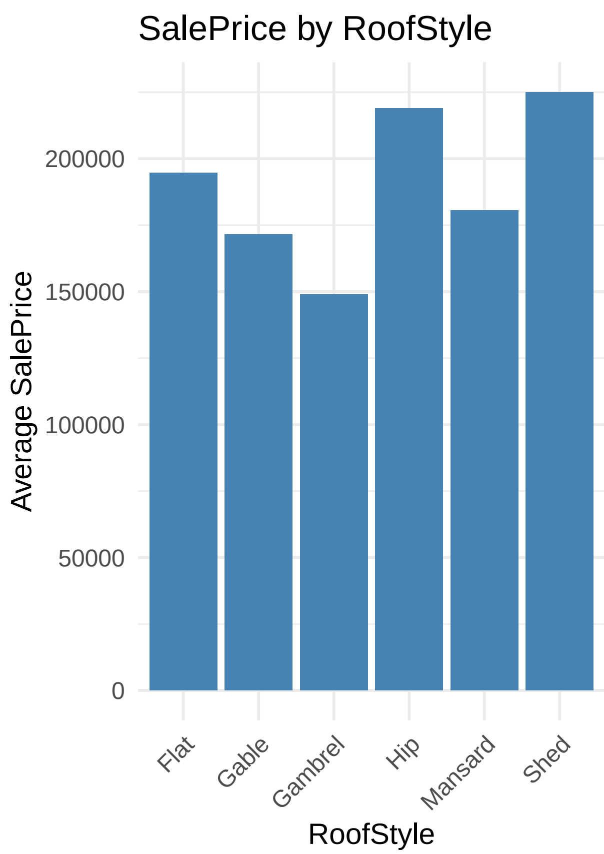


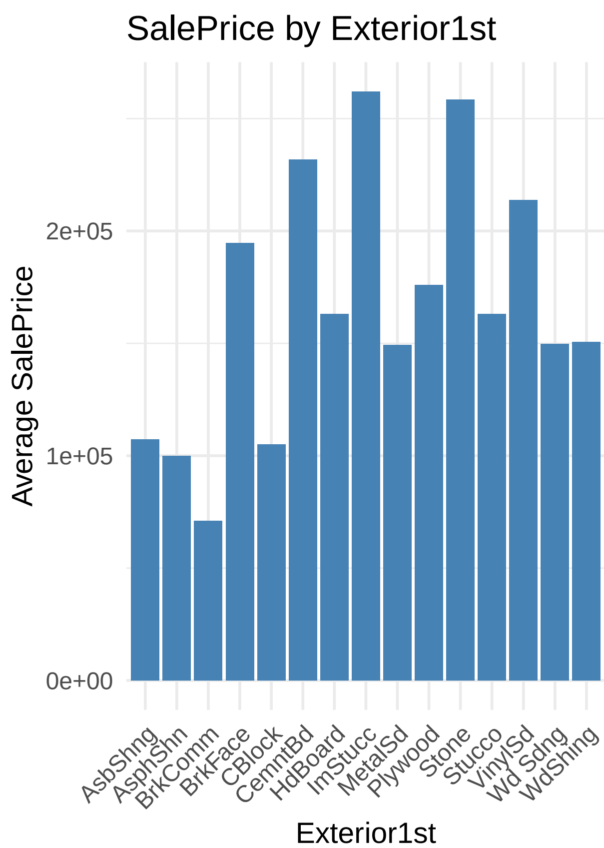

SalePrice by LotConfig
Average SalePrice
200000
150000
100000
50000
0
Corner
CulDSac
FR2
FR3
Inside
LotConfig

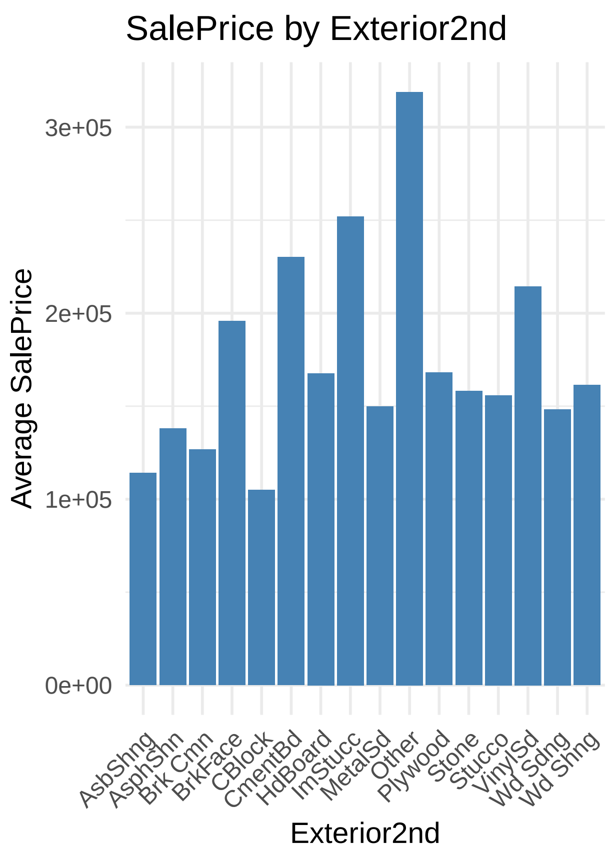

SalePrice by Neighborhood
Average SalePrice
3e+05
2e+05
1e+05
0e+00
Neighborhood

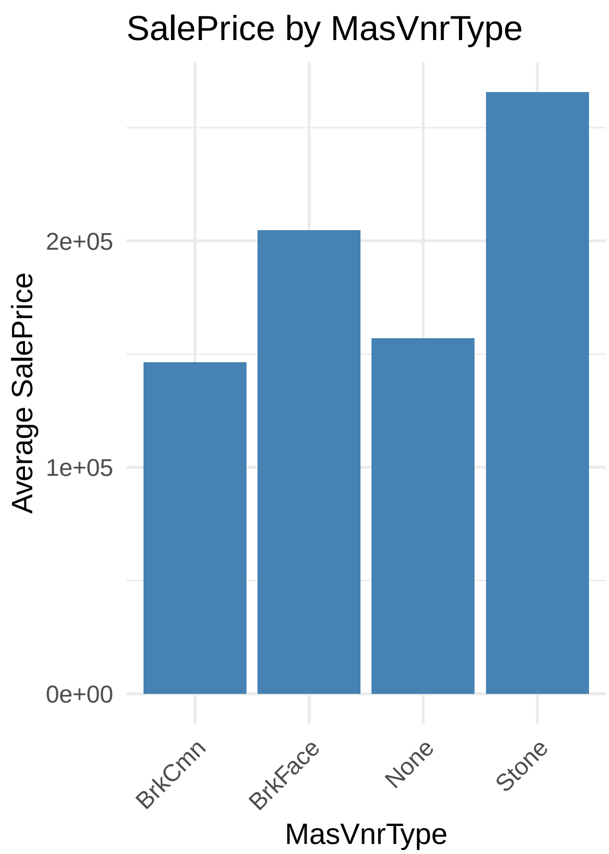

SalePrice by BldgType
Average SalePrice
150000
100000
50000
0
1Fam
2fmCon
Duplex
Twnhs
TwnhsE
BldgType

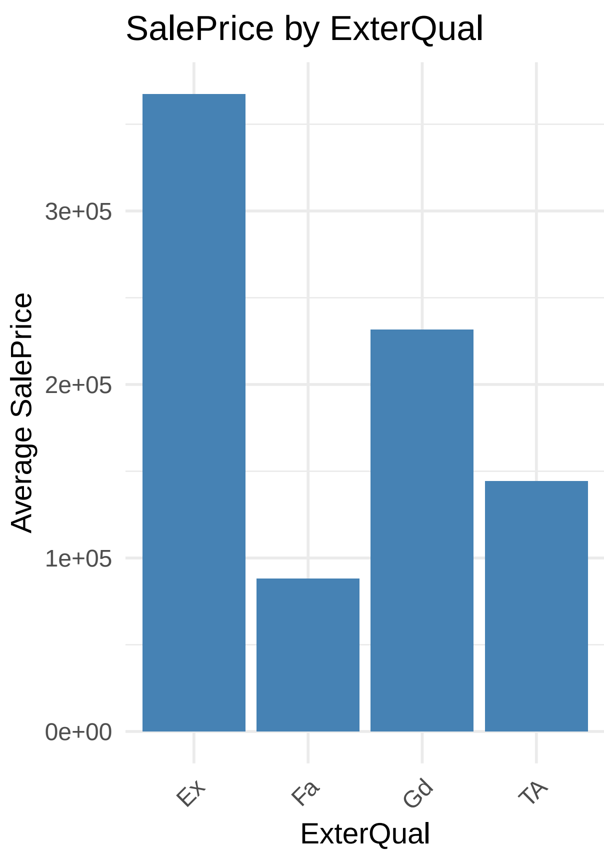

SalePrice by HouseStyle
Average SalePrice
200000
150000
100000
50000
0
1.5Fin
1.5Unf
1Story
2.5Fin
2.5Unf
2Story
SFoyer
SLvl
HouseStyle

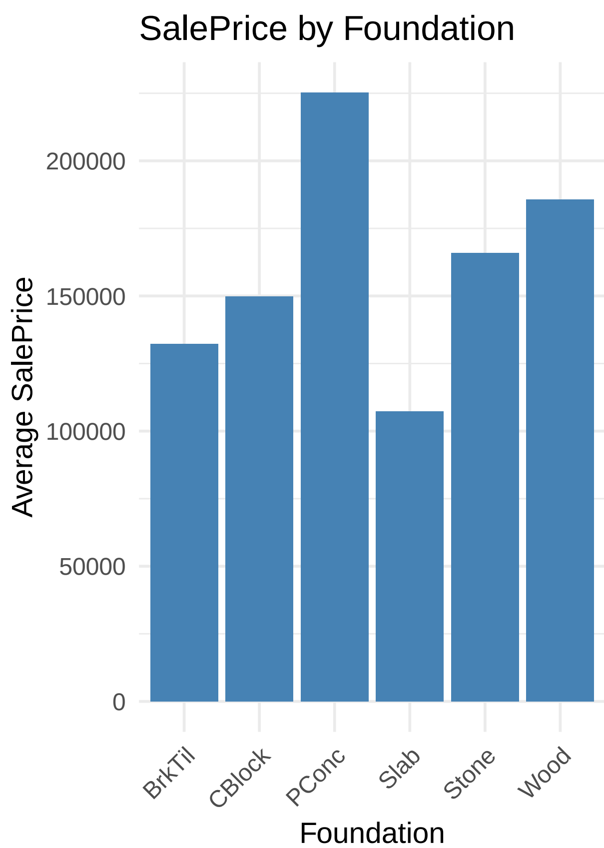

SalePrice by RoofStyle
Average SalePrice
200000
150000
100000
50000
0
Flat
Gable
Gambrel
Hip
Mansard
Shed
RoofStyle

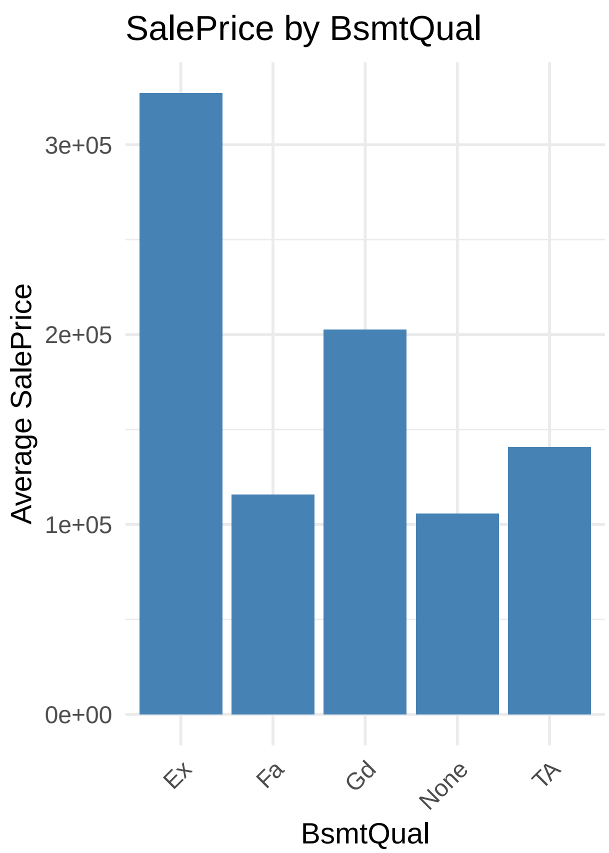

SalePrice by Exterior1st
Average SalePrice
2e+05
1e+05
0e+00
AsbShng
AsphShn
BrkComm
BrkFace
CBlock
CemntBd
HdBoard
ImStucc
MetalSd
Plywood
Stone
Stucco
VinylSd
Wd Sdng
WdShing
Exterior1st


SalePrice by Exterior2nd
Average SalePrice
3e+05
2e+05
1e+05
0e+00
AsbShng
AsphShn
Brk Cmn
BrkFace
CBlock
CmentBd
HdBoard
ImStucc
MetalSd
Other
Plywood
Stone
Stucco
VinylSd
Wd Sdng
Wd Shng
Exterior2nd

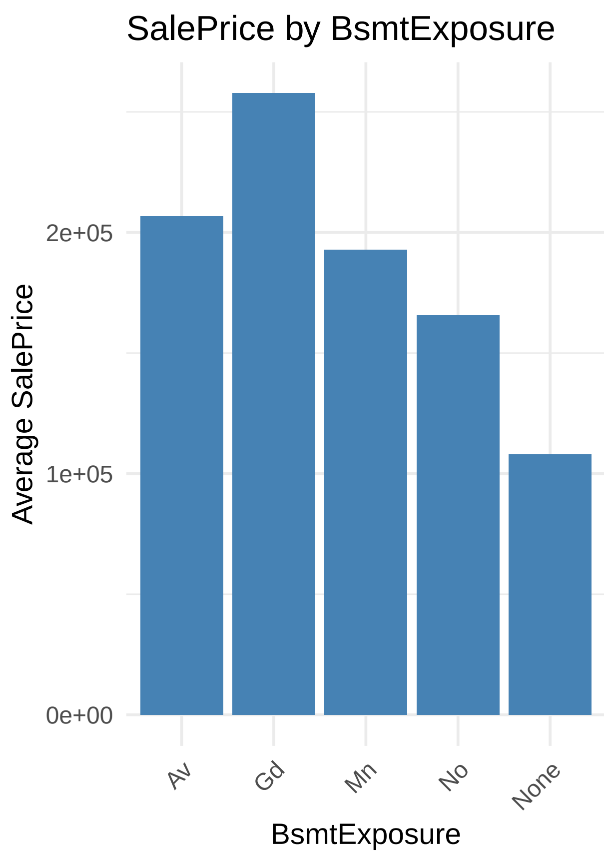

SalePrice by MasVnrType
Average SalePrice
2e+05
1e+05
0e+00
BrkCmn
BrkFace
None
Stone
MasVnrType


SalePrice by ExterQual
Average SalePrice
3e+05
2e+05
1e+05
0e+00
Ex
Fa
Gd
TA
ExterQual

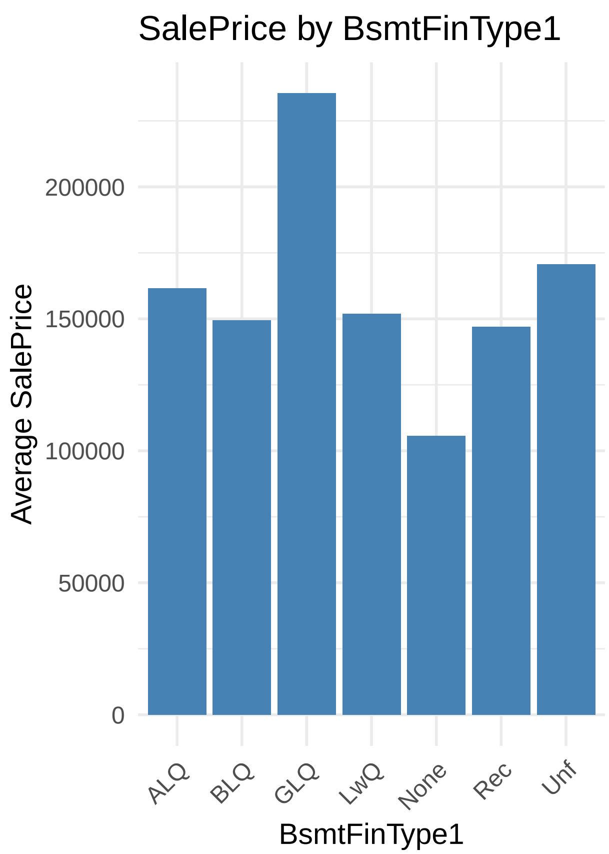

SalePrice by Foundation
Average SalePrice
200000
150000
100000
50000
0
BrkTil
CBlock
PConc
Slab
Stone
Wood
Foundation

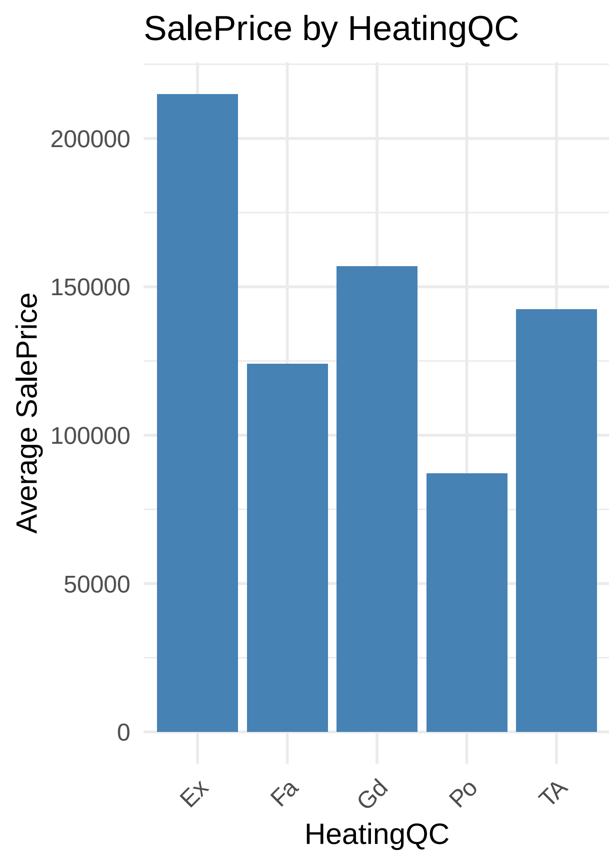


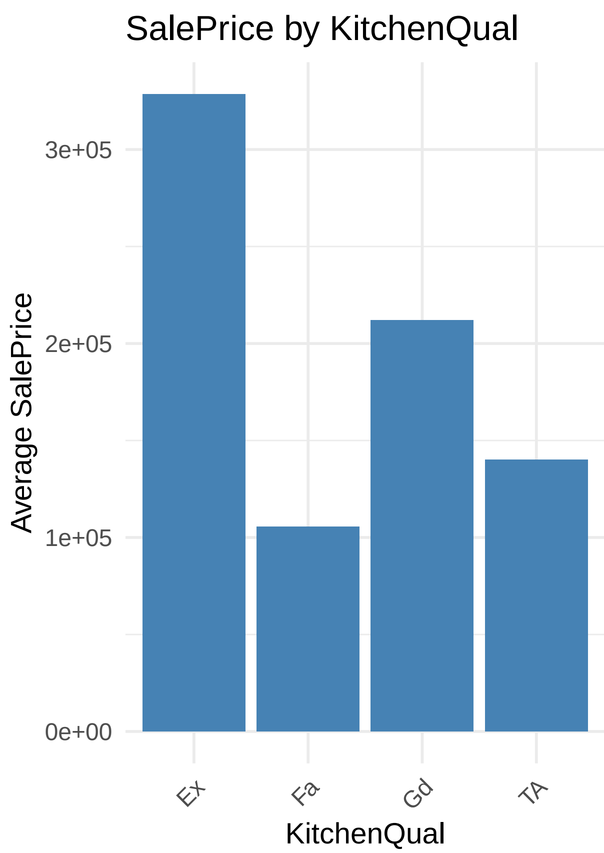
SalePrice by BsmtQual
Average SalePrice
3e+05
2e+05
1e+05
0e+00
Ex
Fa
Gd
None
TA
BsmtQual

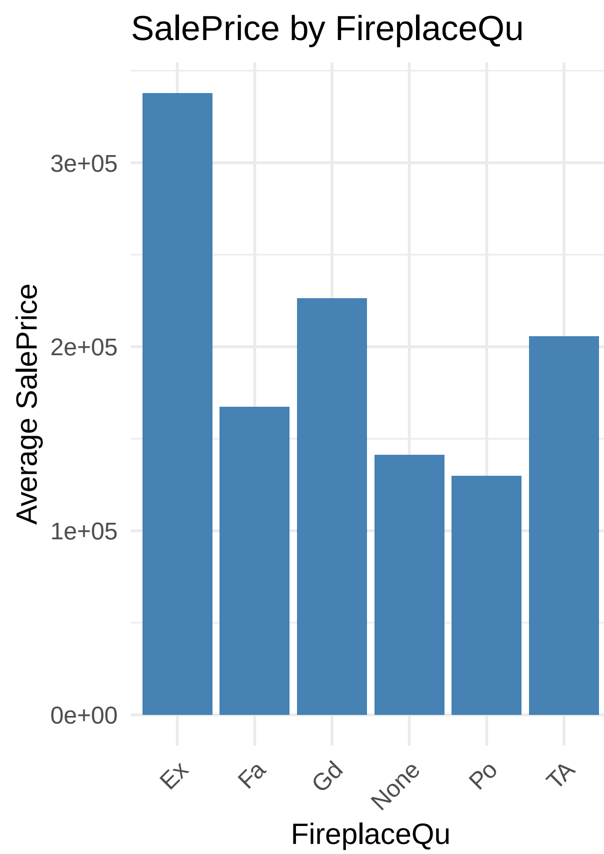


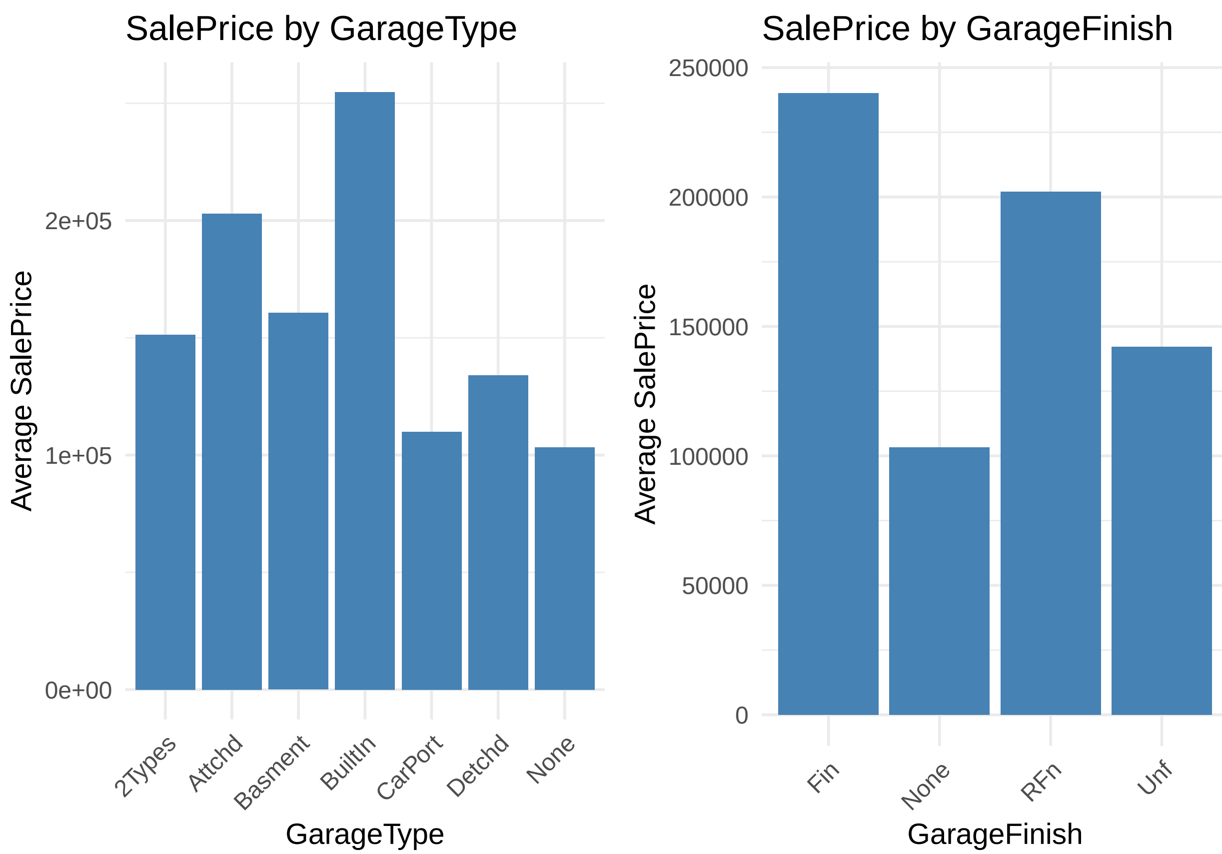
SalePrice by BsmtExposure
Average SalePrice
2e+05
1e+05
0e+00
Av
Gd
Mn
No
None
BsmtExposure

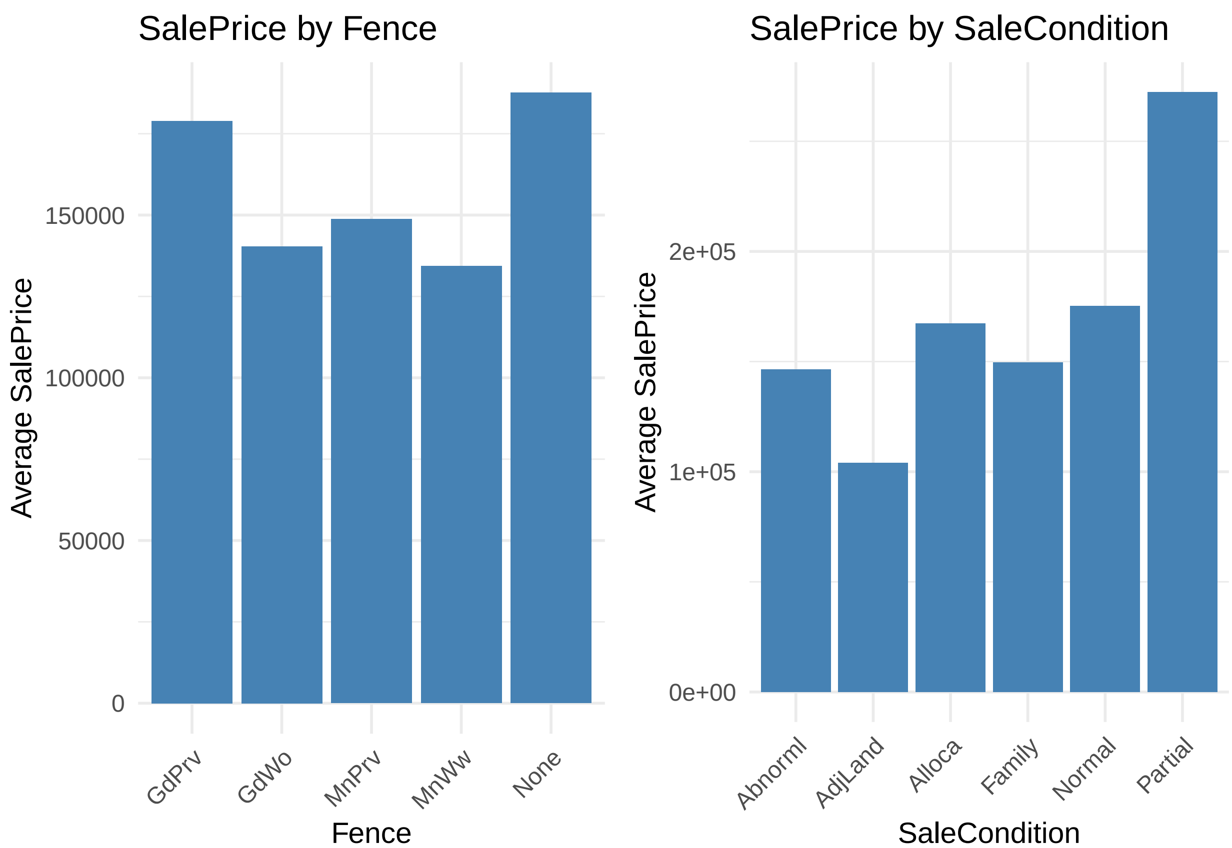

SalePrice by BsmtFinType1
Average SalePrice
0
50000
100000
150000
200000
ALQ
BLQ
GLQ
LwQ
None
Rec
Unf
BsmtFinType1


SalePrice by HeatingQC
Average SalePrice
0
50000
100000
150000
200000
Ex
Fa
Gd
Po
TA
HeatingQC


SalePrice by KitchenQual
Average SalePrice
3e+05
2e+05
1e+05
0e+00
Ex
Fa
Gd
TA
KitchenQual


SalePrice by FireplaceQu
Average SalePrice
3e+05
2e+05
1e+05
0e+00
Ex
Fa
Gd
None
Po
TA
FireplaceQu


SalePrice by GarageType
Average SalePrice
2e+05
1e+05
0e+00
2Types
Attchd
Basment
BuiltIn
CarPort
Detchd
None
GarageType
SalePrice by GarageFinish
Average SalePrice
250000
200000
150000
100000
50000
0
Fin
None
RFn
Unf
GarageFinish



We can see that for each categorical variable that the SalePrice is different across each category in each categorical variable which is good and tells us that they will all be useful.

#Part II: Model Analysis

+After cleaning our data and performing EDA we are going to fit a regression tree model to our data.

+A regression tree model is a decision tree that predicts a continuous variable. It predicts by recursively partitioning the predictor space into smaller and smaller subregions the more we split the tree. The tree defines local regions using a step-function-like approach.

+The regression tree model would give us the best results because they work with complex data sets where there's a mix of categorical and numerical variables.Trees also don't have the typical linear regression interpretation, so we wouldn't need to create indicator variables to represent the categorical features. Lastly, regression trees allow us to easily interpret the results.

+The regression tree model makes minimal assumptions on the relationships in the data set. The assumption generally being that the data can be partitioned into subsets and that each split is independent and interpretable.

```
cv <- trainControl(method = "cv", number = 5)  # 5-fold cross-validation
mtry_grid <- expand.grid(.mtry = c(15, 20, 25, 30)) # Tuning grid for mtry

# Train the model using random forest with cross-validation
set.seed(123)
rf_cv_model <- train(SalePrice ~ .,data = train_data,method = "rf",trControl = cv,tuneGrid = mtry_grid,
print(rf_cv_model)
```

```
## Random Forest
```

```
##
## 1460 samples
##   30 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1169, 1169, 1167, 1168, 1167
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared   MAE
##   15    30690.20  0.8621157  18241.05
##   20    29802.93  0.8685531  17814.46
##   25    29474.77  0.8697370  17700.93
##   30    29266.70  0.8710967  17554.12
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 30.
```

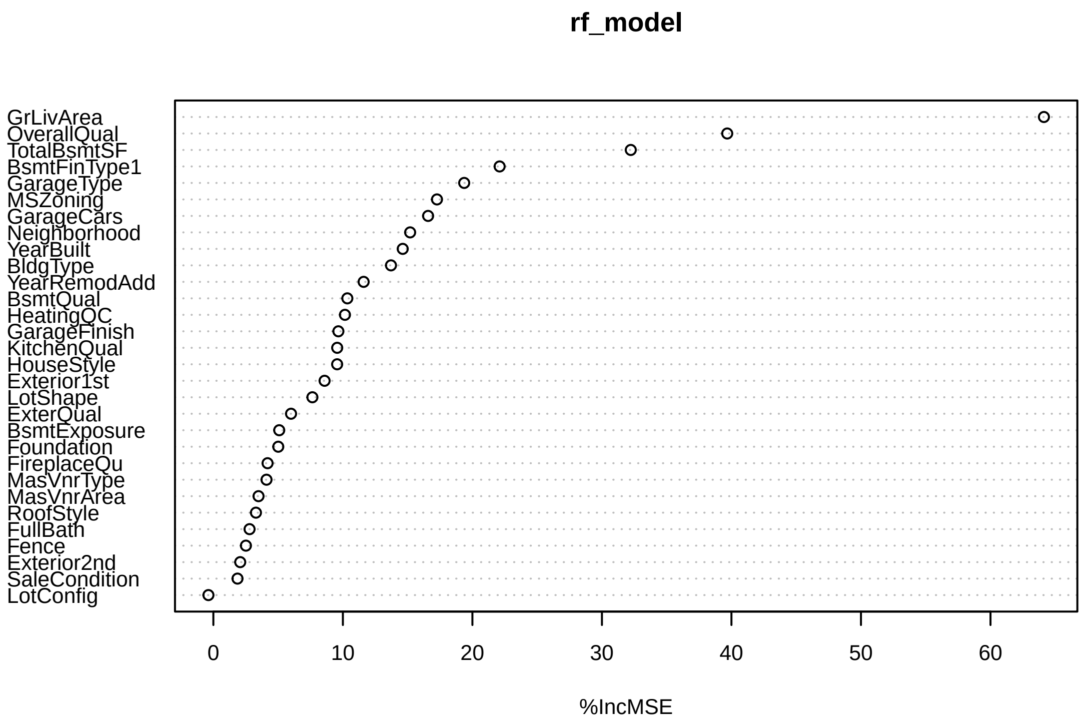
After running cross-validation to select the best value for mtry which represents the number of predictors sampled for splitting at each node. We can see that mtry=30 gives the best results. We will now fit the model with 500 trees and mtry=30

```
set.seed(123)
rf_model <- randomForest(SalePrice ~ .,data = train_data,ntree = 500, mtry = 30, importance = TRUE)
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = SalePrice ~ ., data = train_data, ntree = 500,      mtry = 30, importance = 
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 30
##
##           Mean of squared residuals: 934607442
##                     % Var explained: 85.18
```

After fitting the model we got an R^2 value of 0.8218, indicating that 82% of the variance in the response variable is explained by the selected features. This suggests that the model provided a strong fit for the data.

```
par(cex = 0.7)
varImpPlot(rf_model, type = 1) # Plot variable importance
```



We can see that the most important variable is `GrLiveArea` (Above ground living area square feet) which makes sense because larger houses will cost more. Along with that variable, `OverallQual` (Overall material and finish of the house) and `TotalBsmtSF` (Total square feet of the basement) are also very important to the model and separate themselves from the other variables. `TotalBsmtSF`is very similar to `GrLiveArea` and probably gives a similar value so that explains why it is so important and if the quality of the house is low then the price will also be lower.

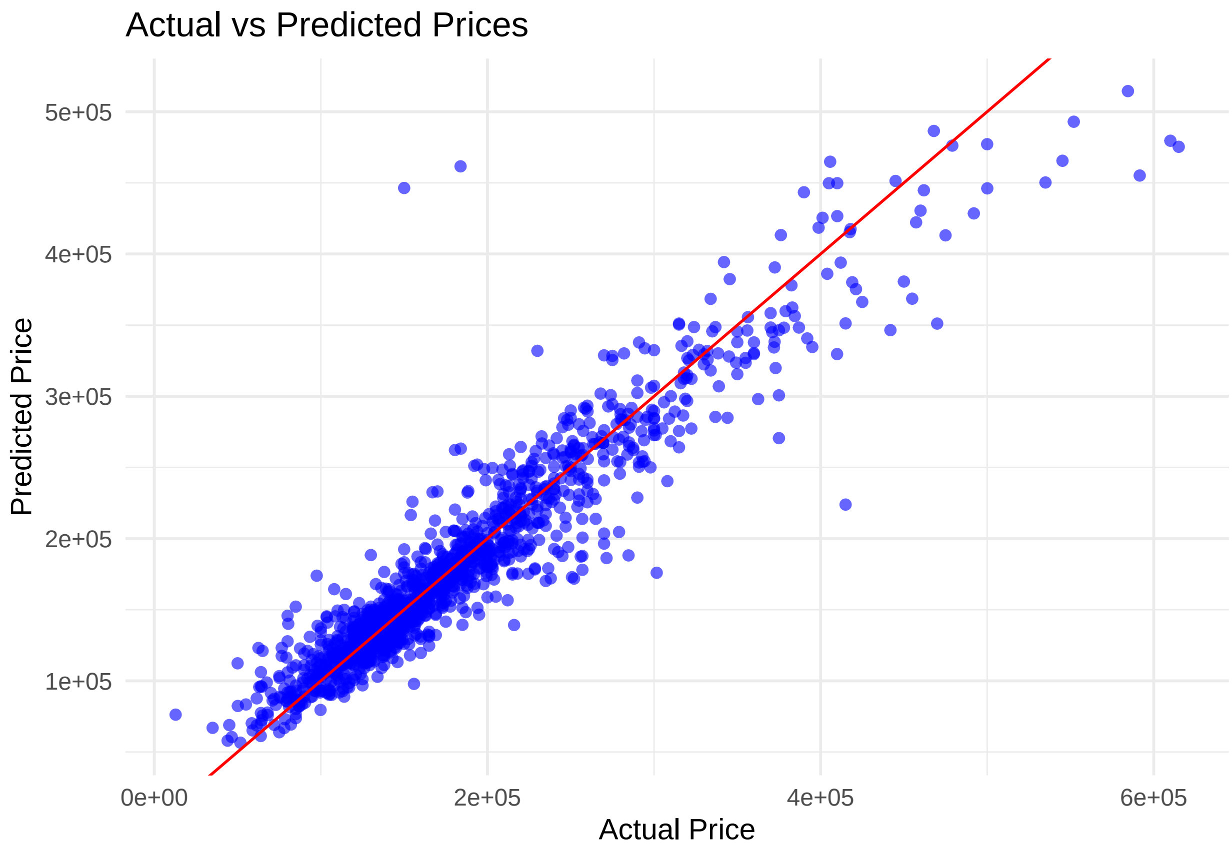
We will now run the test data through the pre-processing and then evaluate it's performance with the model.

```
#Pre-Processing on test_data
test_data = test_data[, !(names(test_data) %in% c("Id", "LotFrontage", "Street", "Alley", "PoolQC", "Mi

test_data$MasVnrArea[is.na(test_data$MasVnrArea)] <- 0
test_data[is.na(test_data)] <- "None"
```

```
test_x = test_data[, !(names(test_data) == "SalePrice")] #predictors of test data
test_y = test_data[, (names(test_data) == "SalePrice")] #response of test data

predictions <- predict(rf_model, newdata = test_x) #predict sale price on test data
```

```
#data drame of results
results = data.frame(Actual = test_y, Predicted = predictions)

# Scatterplot of actual vs predicted values
ggplot(data = results, aes(x = Actual, y = Predicted)) +
  geom_point(color = "blue", alpha = 0.6) +
```

```r
geom_abline(slope = 1, intercept = 0, color = "red", linetype = "solid") +
labs(title = "Actual vs Predicted Prices",
     x = "Actual Price",
     y = "Predicted Price") +
theme_minimal()
```




```r
rmse <- sqrt(mean((predictions - test_y)^2)) #get rmse of predictions
cat("RMSE: ", rmse, "\n")
```

```
## RMSE:  26306.08
```

After running the test data through the model we can see that on average we are $26,306.08 off from the actual sale price.