Jemael Nzihou

2025-02-20

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Develop a Model to Predict Used Home Prices Using Multiple Regression

The goal was to develop a Model to Predict Used Home Prices Using Multiple Regression. The assignment will incorporate a project developed in R, a report presenting the results. Introduction The assignment is to explore how to construct, analyze, and refine a multiple regression model using a House Prices, Advanced Regression Techniques (https://shorturl.at/cGigI) dataset. The task was to optimize the model based on degrees of freedom, the F score, and the number of predictors, providing a detailed rationale for the different steps taken. This dataset includes various features related to residential homes in Ames, Iowa, and their sale prices, making it an excellent candidate for multiple regression analysis.

#Load the dataset

houseprice <- read.csv("C:/Users/jemae/Downloads/DDS-8521 v1 Statistical Modeling/House-prices-advanced-regression-techniques .csv", header=TRUE)

#houseprice<- House.prices.advanced.regression.techniques.

Data Exploration and Preprocessing:

1. Start by exploring the dataset to understand the features available.

Identify continuous, categorical, and ordinal variables.

 # Check the dimensions (number of rows and columns)

dim(houseprice)

[1] 2919 84

dim(houseprice) Displays the number of rows (observations) and columns (features).

 # Display the first few rows of the dataset

head(houseprice)

Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape

1 1461 20 RH 80 11622 Pave <NA> Reg

2 1462 20 RL 81 14267 Pave <NA> IR1

```
## 3 1463       60      RL         74   13830   Pave  <NA>     IR1
## 4 1464       60      RL         78    9978   Pave  <NA>     IR1
## 5 1465      120      RL         43    5005   Pave  <NA>     IR1
## 6 1466       60      RL         75   10000   Pave  <NA>     IR1
##   LandContour Utilities LotConfig LandSlope Neighborhood Condition1 Condition2
## 1         Lvl    AllPub    Inside       Gtl       NAmes      Feedr       Norm
## 2         Lvl    AllPub    Corner       Gtl       NAmes       Norm       Norm
## 3         Lvl    AllPub    Inside       Gtl     Gilbert       Norm       Norm
## 4         Lvl    AllPub    Inside       Gtl     Gilbert       Norm       Norm
## 5         HLS    AllPub    Inside       Gtl     StoneBr       Norm       Norm
## 6         Lvl    AllPub    Corner       Gtl     Gilbert       Norm       Norm
##   BldgType HouseStyle OverallQual OverallCond YearBuilt YearRemodAdd RoofStyle
## 1    1Fam     1Story           5           6      1961         1961     Gable
## 2    1Fam     1Story           6           6      1958         1958       Hip
## 3    1Fam     2Story           5           5      1997         1998     Gable
## 4    1Fam     2Story           6           6      1998         1998     Gable
## 5   TwnhsE     1Story           8           5      1992         1992     Gable
## 6    1Fam     2Story           6           5      1993         1994     Gable
##   RoofMatl Exterior1st Exterior2nd MasVnrType MasVnrArea ExterQual ExterCond
## 1  CompShg     VinylSd     VinylSd       None          0        TA        TA
## 2  CompShg     Wd Sdng     Wd Sdng     BrkFace        108        TA        TA
## 3  CompShg     VinylSd     VinylSd       None          0        TA        TA
## 4  CompShg     VinylSd     VinylSd     BrkFace         20        TA        TA
## 5  CompShg     HdBoard     HdBoard       None          0        Gd        TA
## 6  CompShg     HdBoard     HdBoard       None          0        TA        TA
##   Foundation BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinSF1
## 1      CBlock       TA       TA          No          Rec        468
## 2      CBlock       TA       TA          No          ALQ        923
## 3       PConc       Gd       TA          No          GLQ        791
```

```
## 4     PConc      TA      TA         No        GLQ       602
## 5     PConc      Gd      TA         No        ALQ       263
## 6     PConc      Gd      TA         No        Unf         0
##   BsmtFinType2 BsmtFinSF2 BsmtUnfSF TotalBsmtSF Heating HeatingQC CentralAir
## 1        LwQ       144       270        882    GasA       TA          Y
## 2        Unf         0       406       1329    GasA       TA          Y
## 3        Unf         0       137        928    GasA       Gd          Y
## 4        Unf         0       324        926    GasA       Ex          Y
## 5        Unf         0      1017       1280    GasA       Ex          Y
## 6        Unf         0       763        763    GasA       Gd          Y
##   Electrical X1stFlrSF X2ndFlrSF LowQualFinSF GrLivArea BsmtFullBath
## 1     SBrkr       896        0           0       896            0
## 2     SBrkr      1329        0           0      1329            0
## 3     SBrkr       928      701           0      1629            0
## 4     SBrkr       926      678           0      1604            0
## 5     SBrkr      1280        0           0      1280            0
## 6     SBrkr       763      892           0      1655            0
##   BsmtHalfBath FullBath HalfBath BedroomAbvGr KitchenAbvGr KitchenQual
## 1            0        1        0            2            1          TA
## 2            0        1        1            3            1          Gd
## 3            0        2        1            3            1          TA
## 4            0        2        1            3            1          Gd
## 5            0        2        0            2            1          Gd
## 6            0        2        1            3            1          TA
##   TotRmsAbvGrd Functional Fireplaces FireplaceQu GarageType GarageYrBlt
## 1            5        Typ          0       <NA>     Attchd        1961
## 2            6        Typ          0       <NA>     Attchd        1958
## 3            6        Typ          1         TA     Attchd        1997
## 4            7        Typ          1         Gd     Attchd        1998
```

```
## 5          5       Typ        0       <NA>     Attchd       1992
## 6          7       Typ        1         TA     Attchd       1993
##   GarageFinish GarageCars GarageArea GarageQual GarageCond PavedDrive
## 1          Unf         1        730         TA         TA          Y
## 2          Unf         1        312         TA         TA          Y
## 3          Fin         2        482         TA         TA          Y
## 4          Fin         2        470         TA         TA          Y
## 5          RFn         2        506         TA         TA          Y
## 6          Fin         2        440         TA         TA          Y
##   WoodDeckSF OpenPorchSF EnclosedPorch X3SsnPorch ScreenPorch PoolArea PoolQC
## 1        140          0             0          0         120        0   <NA>
## 2        393         36             0          0           0        0   <NA>
## 3        212         34             0          0           0        0   <NA>
## 4        360         36             0          0           0        0   <NA>
## 5          0         82             0          0         144        0   <NA>
## 6        157         84             0          0           0        0   <NA>
##   Fence MiscFeature MiscVal MoSold YrSold SaleType SaleCondition SalePrice  X
## 1 MnPrv        <NA>       0      6   2010       WD        Normal    163000 NA
## 2  <NA>        Gar2   12500      6   2010       WD        Normal    163000 NA
## 3 MnPrv        <NA>       0      3   2010       WD        Normal    163000 NA
## 4  <NA>        <NA>       0      6   2010       WD        Normal    163000 NA
## 5  <NA>        <NA>       0      1   2010       WD        Normal    163000 NA
## 6  <NA>        <NA>       0      4   2010       WD        Normal    163000 NA
##   X.1 X.2
## 1  NA  NA
## 2  NA  NA
## 3  NA  NA
## 4  NA  NA
## 5  NA  NA
```

6 NA NA

head(houseprice) Shows the first few rows to get a preview of the dataset.

Display the last few rows of the dataset

tail(houseprice)

```
##       Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape
## 2914 1455        20       FV         62    7500   Pave  Pave      Reg
## 2915 1456        60       RL         62    7917   Pave  <NA>      Reg
## 2916 1457        20       RL         85   13175   Pave  <NA>      Reg
## 2917 1458        70       RL         66    9042   Pave  <NA>      Reg
## 2918 1459        20       RL         68    9717   Pave  <NA>      Reg
## 2919 1460        20       RL         75    9937   Pave  <NA>      Reg
##      LandContour Utilities LotConfig LandSlope Neighborhood Condition1
## 2914         Lvl    AllPub    Inside       Gtl     Somerst       Norm
## 2915         Lvl    AllPub    Inside       Gtl     Gilbert       Norm
## 2916         Lvl    AllPub    Inside       Gtl      NWAmes       Norm
## 2917         Lvl    AllPub    Inside       Gtl     Crawfor       Norm
## 2918         Lvl    AllPub    Inside       Gtl       NAmes       Norm
## 2919         Lvl    AllPub    Inside       Gtl     Edwards       Norm
##      Condition2 BldgType HouseStyle OverallQual OverallCond YearBuilt
## 2914       Norm     1Fam     1Story           7           5      2004
## 2915       Norm     1Fam     2Story           6           5      1999
## 2916       Norm     1Fam     1Story           6           6      1978
## 2917       Norm     1Fam     2Story           7           9      1941
## 2918       Norm     1Fam     1Story           5           6      1950
## 2919       Norm     1Fam     1Story           5           6      1965
##      YearRemodAdd RoofStyle RoofMatl Exterior1st Exterior2nd MasVnrType
## 2914         2005     Gable  CompShg     VinylSd     VinylSd       None
## 2915         2000     Gable  CompShg     VinylSd     VinylSd       None
## 2916         1988     Gable  CompShg     Plywood     Plywood      Stone
```

```
## 2917       2006    Gable  CompShg    CemntBd    CmentBd      None
## 2918       1996      Hip  CompShg    MetalSd    MetalSd      None
## 2919       1965    Gable  CompShg    HdBoard    HdBoard      None
##      MasVnrArea ExterQual ExterCond Foundation BsmtQual BsmtCond BsmtExposure
## 2914        0       Gd       TA     PConc      Gd       TA         No
## 2915        0       TA       TA     PConc      Gd       TA         No
## 2916      119       TA       TA     CBlock     Gd       TA         No
## 2917        0       Ex       Gd     Stone      TA       Gd         No
## 2918        0       TA       TA     CBlock     TA       TA         Mn
## 2919        0       Gd       TA     CBlock     TA       TA         No
##      BsmtFinType1 BsmtFinSF1 BsmtFinType2 BsmtFinSF2 BsmtUnfSF TotalBsmtSF
## 2914          GLQ        410          Unf          0       811        1221
## 2915          Unf          0          Unf          0       953         953
## 2916          ALQ        790          Rec        163       589        1542
## 2917          GLQ        275          Unf          0       877        1152
## 2918          GLQ         49          Rec       1029         0        1078
## 2919          BLQ        830          LwQ        290       136        1256
##      Heating HeatingQC CentralAir Electrical X1stFlrSF X2ndFlrSF LowQualFinSF
## 2914    GasA       Ex          Y      SBrkr      1221         0           0
## 2915    GasA       Ex          Y      SBrkr       953       694           0
## 2916    GasA       TA          Y      SBrkr      2073         0           0
## 2917    GasA       Ex          Y      SBrkr      1188      1152           0
## 2918    GasA       Gd          Y      FuseA      1078         0           0
## 2919    GasA       Gd          Y      SBrkr      1256         0           0
##      GrLivArea BsmtFullBath BsmtHalfBath FullBath HalfBath BedroomAbvGr
## 2914      1221            1            0        2        0            2
## 2915      1647            0            0        2        1            3
## 2916      2073            1            0        2        0            3
## 2917      2340            0            0        2        0            4
```

```
## 2918     1078         1         0      1      0         2
## 2919     1256         1         0      1      1         3
##      KitchenAbvGr KitchenQual TotRmsAbvGrd Functional Fireplaces FireplaceQu
## 2914            1          Gd            6        Typ          0       <NA>
## 2915            1          TA            7        Typ          1         TA
## 2916            1          TA            7       Min1          2         TA
## 2917            1          Gd            9        Typ          2         Gd
## 2918            1          Gd            5        Typ          0       <NA>
## 2919            1          TA            6        Typ          0       <NA>
##      GarageType GarageYrBlt GarageFinish GarageCars GarageArea GarageQual
## 2914    Attchd        2004          RFn          2        400         TA
## 2915    Attchd        1999          RFn          2        460         TA
## 2916    Attchd        1978          Unf          2        500         TA
## 2917    Attchd        1941          RFn          1        252         TA
## 2918    Attchd        1950          Unf          1        240         TA
## 2919    Attchd        1965          Fin          1        276         TA
##      GarageCond PavedDrive WoodDeckSF OpenPorchSF EnclosedPorch X3SsnPorch
## 2914         TA          Y          0        113             0          0
## 2915         TA          Y          0         40             0          0
## 2916         TA          Y        349          0             0          0
## 2917         TA          Y          0         60             0          0
## 2918         TA          Y        366          0           112          0
## 2919         TA          Y        736         68             0          0
##      ScreenPorch PoolArea PoolQC Fence MiscFeature MiscVal MoSold YrSold
## 2914           0        0  <NA>  <NA>       <NA>       0     10   2009
## 2915           0        0  <NA>  <NA>       <NA>       0      8   2007
## 2916           0        0  <NA> MnPrv       <NA>       0      2   2010
## 2917           0        0  <NA> GdPrv       Shed    2500      5   2010
## 2918           0        0  <NA>  <NA>       <NA>       0      4   2010
```

```
## 2919          0       0   <NA>  <NA>       <NA>      0     6   2008
##      SaleType SaleCondition SalePrice  X X.1 X.2
## 2914       WD       Normal    185000 NA  NA  NA
## 2915       WD       Normal    175000 NA  NA  NA
## 2916       WD       Normal    210000 NA  NA  NA
## 2917       WD       Normal    266500 NA  NA  NA
## 2918       WD       Normal    142125 NA  NA  NA
## 2919       WD       Normal    147500 NA  NA  NA
```

tail(houseprice) shows the last few rows to get a preview of the dataset.

 # Get an overview of the dataset structure (column names, types, etc.)

str(houseprice)

```
## 'data.frame':    2919 obs. of  84 variables:
##  $ Id          : int  1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 ...
##  $ MSSubClass   : int  20 20 60 60 120 60 20 60 20 20 ...
##  $ MSZoning     : chr  "RH" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  80 81 74 78 43 75 NA 63 85 70 ...
##  $ LotArea      : int  11622 14267 13830 9978 5005 10000 7980 8402 10176 8400 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "IR1" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "Corner" "Inside" "Inside" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "NAmes" "NAmes" "Gilbert" "Gilbert" ...
##  $ Condition1   : chr  "Feedr" "Norm" "Norm" "Norm" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "1Story" "1Story" "2Story" "2Story" ...
```

```
##  $ OverallQual  : int  5 6 5 6 8 6 6 6 7 4 ...
##  $ OverallCond  : int  6 6 5 6 5 5 7 5 5 5 ...
##  $ YearBuilt    : int  1961 1958 1997 1998 1992 1993 1992 1998 1990 1970 ...
##  $ YearRemodAdd : int  1961 1958 1998 1998 1992 1994 2007 1998 1990 1970 ...
##  $ RoofStyle    : chr  "Gable" "Hip" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "Wd Sdng" "VinylSd" "VinylSd" ...
##  $ Exterior2nd  : chr  "VinylSd" "Wd Sdng" "VinylSd" "VinylSd" ...
##  $ MasVnrType   : chr  "None" "BrkFace" "None" "BrkFace" ...
##  $ MasVnrArea   : int  0 108 0 20 0 0 0 0 0 0 ...
##  $ ExterQual    : chr  "TA" "TA" "TA" "TA" ...
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "CBlock" "CBlock" "PConc" "PConc" ...
##  $ BsmtQual     : chr  "TA" "TA" "Gd" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "TA" ...
##  $ BsmtExposure : chr  "No" "No" "No" "No" ...
##  $ BsmtFinType1 : chr  "Rec" "ALQ" "GLQ" "GLQ" ...
##  $ BsmtFinSF1   : int  468 923 791 602 263 0 935 0 637 804 ...
##  $ BsmtFinType2 : chr  "LwQ" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  144 0 0 0 0 0 0 0 0 78 ...
##  $ BsmtUnfSF    : int  270 406 137 324 1017 763 233 789 663 0 ...
##  $ TotalBsmtSF  : int  882 1329 928 926 1280 763 1168 789 1300 882 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "TA" "TA" "Gd" "Ex" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##  $ X1stFlrSF    : int  896 1329 928 926 1280 763 1187 789 1341 882 ...
##  $ X2ndFlrSF    : int  0 0 701 678 0 892 0 676 0 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ GrLivArea    : int  896 1329 1629 1604 1280 1655 1187 1465 1341 882 ...
##  $ BsmtFullBath : int  0 0 0 0 0 0 1 0 1 1 ...
##  $ BsmtHalfBath : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  1 1 2 2 2 2 2 2 1 1 ...
##  $ HalfBath     : int  0 1 1 1 0 1 0 1 1 0 ...
##  $ BedroomAbvGr : int  2 3 3 3 2 3 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ KitchenQual  : chr  "TA" "Gd" "TA" "Gd" ...
##  $ TotRmsAbvGrd : int  5 6 6 7 5 7 6 7 5 4 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  0 0 1 1 0 1 0 1 1 0 ...
##  $ FireplaceQu  : chr  NA NA "TA" "Gd" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Attchd" ...
##  $ GarageYrBlt  : int  1961 1958 1997 1998 1992 1993 1992 1998 1990 1970 ...
##  $ GarageFinish : chr  "Unf" "Unf" "Fin" "Fin" ...
##  $ GarageCars   : int  1 1 2 2 2 2 2 2 2 2 ...
##  $ GarageArea   : int  730 312 482 470 506 440 420 393 506 525 ...
##  $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  140 393 212 360 0 157 483 0 192 240 ...
##  $ OpenPorchSF  : int  0 36 34 36 82 84 21 75 0 0 ...
##  $ EnclosedPorch: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ScreenPorch  : int  120 0 0 0 144 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  "MnPrv" NA "MnPrv" NA ...
##  $ MiscFeature  : chr  NA "Gar2" NA NA ...
```

$ MiscVal : int 0 12500 0 0 0 0 500 0 0 0 ...

$ MoSold : int 6 6 3 6 1 4 3 5 2 4 ...

$ YrSold : int 2010 2010 2010 2010 2010 2010 2010 2010 2010 2010 ...

$ SaleType : chr "WD" "WD" "WD" "WD" ...

$ SaleCondition: chr "Normal" "Normal" "Normal" "Normal" ...

$ SalePrice : int 163000 163000 163000 163000 163000 163000 163000 163000 163000 163000 ...

$ X : logi NA NA NA NA NA NA ...

$ X.1 : logi NA NA NA NA NA NA ...

$ X.2 : int NA NA NA NA NA NA NA NA NA NA ...

str(houseprice) Provides information about the structure, including column names, data types. there is character(Chr) type of variable and integer (int) on V81 which contains NA

 # View colums names in the data set

colnames(houseprice)

[1] "Id" "MSSubClass" "MSZoning" "LotFrontage"

[5] "LotArea" "Street" "Alley" "LotShape"

[9] "LandContour" "Utilities" "LotConfig" "LandSlope"

[13] "Neighborhood" "Condition1" "Condition2" "BldgType"

[17] "HouseStyle" "OverallQual" "OverallCond" "YearBuilt"

[21] "YearRemodAdd" "RoofStyle" "RoofMatl" "Exterior1st"

[25] "Exterior2nd" "MasVnrType" "MasVnrArea" "ExterQual"

[29] "ExterCond" "Foundation" "BsmtQual" "BsmtCond"

[33] "BsmtExposure" "BsmtFinType1" "BsmtFinSF1" "BsmtFinType2"

[37] "BsmtFinSF2" "BsmtUnfSF" "TotalBsmtSF" "Heating"

[41] "HeatingQC" "CentralAir" "Electrical" "X1stFlrSF"

[45] "X2ndFlrSF" "LowQualFinSF" "GrLivArea" "BsmtFullBath"

[49] "BsmtHalfBath" "FullBath" "HalfBath" "BedroomAbvGr"

[53] "KitchenAbvGr" "KitchenQual" "TotRmsAbvGrd" "Functional"

[57] "Fireplaces" "FireplaceQu" "GarageType" "GarageYrBlt"

[61] "GarageFinish" "GarageCars" "GarageArea" "GarageQual"

```
## [65] "GarageCond"    "PavedDrive"    "WoodDeckSF"    "OpenPorchSF"
## [69] "EnclosedPorch" "X3SsnPorch"    "ScreenPorch"   "PoolArea"
## [73] "PoolQC"        "Fence"         "MiscFeature"   "MiscVal"
## [77] "MoSold"        "YrSold"        "SaleType"      "SaleCondition"
## [81] "SalePrice"     "X"             "X.1"           "X.2"
```

```
# Summarize each column to understand the distribution
summary(houseprice)
```

```
##        Id          MSSubClass      MSZoning         LotFrontage
##  Min.   :   1.0   Min.   : 20.00   Length:2919        Min.   : 21.00
##  1st Qu.: 730.5   1st Qu.: 20.00   Class :character   1st Qu.: 59.00
##  Median :1460.0   Median : 50.00   Mode  :character   Median : 68.00
##  Mean   :1460.0   Mean   : 57.14                      Mean   : 69.31
##  3rd Qu.:2189.5   3rd Qu.: 70.00                      3rd Qu.: 80.00
##  Max.   :2919.0   Max.   :190.00                      Max.   :313.00
##                                                       NA's   :486
##     LotArea         Street            Alley            LotShape
##  Min.   :  1300   Length:2919        Length:2919        Length:2919
##  1st Qu.:  7478   Class :character   Class :character   Class :character
##  Median :  9453   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 10168
##  3rd Qu.: 11570
##  Max.   :215245
##
##  LandContour        Utilities         LotConfig         LandSlope
##  Length:2919        Length:2919        Length:2919        Length:2919
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
```

```
##
##
##  Neighborhood       Condition1        Condition2         BldgType
##  Length:2919        Length:2919       Length:2919        Length:2919
##  Class :character   Class :character  Class :character   Class :character
##  Mode  :character   Mode  :character  Mode  :character   Mode  :character
##
##
##
##
##   HouseStyle        OverallQual      OverallCond      YearBuilt
##  Length:2919        Min.   : 1.000   Min.   :1.000    Min.   :1872
##  Class :character   1st Qu.: 5.000   1st Qu.:5.000    1st Qu.:1954
##  Mode  :character   Median : 6.000   Median :5.000    Median :1973
##                     Mean   : 6.089   Mean   :5.565    Mean   :1971
##                     3rd Qu.: 7.000   3rd Qu.:6.000    3rd Qu.:2001
##                     Max.   :10.000   Max.   :9.000    Max.   :2010
##
##   YearRemodAdd   RoofStyle          RoofMatl         Exterior1st
##  Min.   :1950   Length:2919        Length:2919        Length:2919
##  1st Qu.:1965   Class :character   Class :character   Class :character
##  Median :1993   Mode  :character   Mode  :character   Mode  :character
##  Mean   :1984
##  3rd Qu.:2004
##  Max.   :2010
##
##  Exterior2nd        MasVnrType         MasVnrArea      ExterQual
##  Length:2919        Length:2919        Min.   :   0.0   Length:2919
##  Class :character   Class :character   1st Qu.:   0.0   Class :character
```

```
##  Mode  :character   Mode  :character   Median :   0.0   Mode  :character
##                               Mean   : 102.2
##                               3rd Qu.: 164.0
##                               Max.   :1600.0
##                               NA's   :23
##   ExterCond         Foundation         BsmtQual          BsmtCond
##  Length:2919       Length:2919       Length:2919       Length:2919
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  BsmtExposure      BsmtFinType1        BsmtFinSF1     BsmtFinType2
##  Length:2919       Length:2919       Min.   :   0.0   Length:2919
##  Class :character   Class :character   1st Qu.:   0.0   Class :character
##  Mode  :character   Mode  :character   Median : 368.5   Mode  :character
##                               Mean   : 441.4
##                               3rd Qu.: 733.0
##                               Max.   :5644.0
##                               NA's   :1
##    BsmtFinSF2       BsmtUnfSF       TotalBsmtSF       Heating
##  Min.   :   0.00   Min.   :   0.0   Min.   :   0.0   Length:2919
##  1st Qu.:   0.00   1st Qu.: 220.0   1st Qu.: 793.0   Class :character
##  Median :   0.00   Median : 467.0   Median : 989.5   Mode  :character
##  Mean   :  49.58   Mean   : 560.8   Mean   :1051.8
##  3rd Qu.:   0.00   3rd Qu.: 805.5   3rd Qu.:1302.0
##  Max.   :1526.00   Max.   :2336.0   Max.   :6110.0
##  NA's   :1         NA's   :1        NA's   :1
```

```
##   HeatingQC         CentralAir         Electrical         X1stFlrSF
##  Length:2919        Length:2919        Length:2919        Min.   : 334
##  Class :character   Class :character   Class :character   1st Qu.: 876
##  Mode  :character   Mode  :character   Mode  :character   Median :1082
##                                                           Mean   :1160
##                                                           3rd Qu.:1388
##                                                           Max.   :5095
##
##    X2ndFlrSF      LowQualFinSF       GrLivArea     BsmtFullBath
##  Min.   :   0.0   Min.   :   0.000   Min.   : 334   Min.   :0.0000
##  1st Qu.:   0.0   1st Qu.:   0.000   1st Qu.:1126   1st Qu.:0.0000
##  Median :   0.0   Median :   0.000   Median :1444   Median :0.0000
##  Mean   : 336.5   Mean   :   4.694   Mean   :1501   Mean   :0.4299
##  3rd Qu.: 704.0   3rd Qu.:   0.000   3rd Qu.:1744   3rd Qu.:1.0000
##  Max.   :2065.0   Max.   :1064.000   Max.   :5642   Max.   :3.0000
##                                                     NA's   :2
##   BsmtHalfBath       FullBath       HalfBath       BedroomAbvGr
##  Min.   :0.00000   Min.   :0.000   Min.   :0.0000   Min.   :0.00
##  1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:2.00
##  Median :0.00000   Median :2.000   Median :0.0000   Median :3.00
##  Mean   :0.06136   Mean   :1.568   Mean   :0.3803   Mean   :2.86
##  3rd Qu.:0.00000   3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:3.00
##  Max.   :2.00000   Max.   :4.000   Max.   :2.0000   Max.   :8.00
##  NA's   :2
##   KitchenAbvGr   KitchenQual        TotRmsAbvGrd     Functional
##  Min.   :0.000   Length:2919        Min.   : 2.000   Length:2919
##  1st Qu.:1.000   Class :character   1st Qu.: 5.000   Class :character
##  Median :1.000   Mode  :character   Median : 6.000   Mode  :character
##  Mean   :1.045                      Mean   : 6.452
```

```
##  3rd Qu.:1.000                  3rd Qu.: 7.000
##  Max.   :3.000                  Max.   :15.000
##
##   Fireplaces     FireplaceQu        GarageType        GarageYrBlt
##  Min.   :0.0000   Length:2919        Length:2919        Min.   :1895
##  1st Qu.:0.0000   Class :character   Class :character   1st Qu.:1960
##  Median :1.0000   Mode  :character   Mode  :character   Median :1979
##  Mean   :0.5971                                        Mean   :1978
##  3rd Qu.:1.0000                                        3rd Qu.:2002
##  Max.   :4.0000                                        Max.   :2207
##                                                        NA's   :159
##  GarageFinish       GarageCars     GarageArea     GarageQual
##  Length:2919        Min.   :0.000   Min.   :   0.0   Length:2919
##  Class :character   1st Qu.:1.000   1st Qu.: 320.0   Class :character
##  Mode  :character   Median :2.000   Median : 480.0   Mode  :character
##                     Mean   :1.767   Mean   : 472.9
##                     3rd Qu.:2.000   3rd Qu.: 576.0
##                     Max.   :5.000   Max.   :1488.0
##                     NA's   :1       NA's   :1
##   GarageCond        PavedDrive         WoodDeckSF       OpenPorchSF
##  Length:2919        Length:2919        Min.   :   0.00   Min.   :  0.00
##  Class :character   Class :character   1st Qu.:   0.00   1st Qu.:  0.00
##  Mode  :character   Mode  :character   Median :   0.00   Median : 26.00
##                                        Mean   :  93.71   Mean   : 47.49
##                                        3rd Qu.: 168.00   3rd Qu.: 70.00
##                                        Max.   :1424.00   Max.   :742.00
##
##  EnclosedPorch      X3SsnPorch       ScreenPorch        PoolArea
##  Min.   :   0.0   Min.   :  0.000   Min.   :  0.00   Min.   :  0.000
```

```
##  1st Qu.:   0.0   1st Qu.:  0.000   1st Qu.:  0.00   1st Qu.:  0.000
##  Median :   0.0   Median :  0.000   Median :  0.00   Median :  0.000
##  Mean   :  23.1   Mean   :  2.602   Mean   : 16.06   Mean   :  2.252
##  3rd Qu.:   0.0   3rd Qu.:  0.000   3rd Qu.:  0.00   3rd Qu.:  0.000
##  Max.   :1012.0   Max.   :508.000   Max.   :576.00   Max.   :800.000
##
##    PoolQC            Fence          MiscFeature          MiscVal
##  Length:2919        Length:2919        Length:2919        Min.   :    0.00
##  Class :character   Class :character   Class :character   1st Qu.:    0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :    0.00
##                                                           Mean   :   50.83
##                                                           3rd Qu.:    0.00
##                                                           Max.   :17000.00
##
##     MoSold           YrSold       SaleType         SaleCondition
##  Min.   : 1.000   Min.   :2006   Length:2919        Length:2919
##  1st Qu.: 4.000   1st Qu.:2007   Class :character   Class :character
##  Median : 6.000   Median :2008   Mode  :character   Mode  :character
##  Mean   : 6.213   Mean   :2008
##  3rd Qu.: 8.000   3rd Qu.:2009
##  Max.   :12.000   Max.   :2010
##
##    SalePrice         X            X.1             X.2
##  Min.   : 34900   Mode:logical   Mode:logical   Min.   :163000
##  1st Qu.:163000   NA's:2919      NA's:2919      1st Qu.:163000
##  Median :163000                                 Median :163000
##  Mean   :171964                                 Mean   :163000
##  3rd Qu.:163000                                 3rd Qu.:163000
##  Max.   :755000                                 Max.   :163000
```

```
##                                   NA's   :2918
```

summary(houseprice) gives summary statistics like min, max, mean, and quartiles for numeric variables and counts for categorical variables.

1b. Identify Data Types: Continuous, Categorical, and Ordinal Variables

We categorize the features into continuous, categorical, and ordinal variables.

```
# Identify categorical variables (non-numeric variables)
categorical_vars <- names(houseprice)[sapply(houseprice, is.character)]
print("Categorical Variables:")
```

```
## [1] "Categorical Variables:"
```

```
print(categorical_vars)
```

```
##  [1] "MSZoning"     "Street"       "Alley"        "LotShape"
##  [5] "LandContour"   "Utilities"     "LotConfig"     "LandSlope"
##  [9] "Neighborhood"  "Condition1"    "Condition2"    "BldgType"
## [13] "HouseStyle"    "RoofStyle"     "RoofMatl"      "Exterior1st"
## [17] "Exterior2nd"   "MasVnrType"    "ExterQual"     "ExterCond"
## [21] "Foundation"    "BsmtQual"      "BsmtCond"      "BsmtExposure"
## [25] "BsmtFinType1"  "BsmtFinType2"  "Heating"       "HeatingQC"
## [29] "CentralAir"    "Electrical"    "KitchenQual"   "Functional"
## [33] "FireplaceQu"   "GarageType"    "GarageFinish"  "GarageQual"
## [37] "GarageCond"    "PavedDrive"    "PoolQC"        "Fence"
## [41] "MiscFeature"   "SaleType"      "SaleCondition"
```

```
# Identify numerical variables
numeric_vars <- names(houseprice)[sapply(houseprice, is.numeric)]
print("Numerical Variables:")
```

```
## [1] "Numerical Variables:"
```

```
print(numeric_vars)
```

```
##  [1] "Id"          "MSSubClass"    "LotFrontage"   "LotArea"
##  [5] "OverallQual"   "OverallCond"   "YearBuilt"     "YearRemodAdd"
##  [9] "MasVnrArea"    "BsmtFinSF1"    "BsmtFinSF2"    "BsmtUnfSF"
```

[13] "TotalBsmtSF" "X1stFlrSF" "X2ndFlrSF" "LowQualFinSF"

[17] "GrLivArea" "BsmtFullBath" "BsmtHalfBath" "FullBath"

[21] "HalfBath" "BedroomAbvGr" "KitchenAbvGr" "TotRmsAbvGrd"

[25] "Fireplaces" "GarageYrBlt" "GarageCars" "GarageArea"

[29] "WoodDeckSF" "OpenPorchSF" "EnclosedPorch" "X3SsnPorch"

[33] "ScreenPorch" "PoolArea" "MiscVal" "MoSold"

[37] "YrSold" "SalePrice" "X.2"

 # Check if there are any ordinal variables (categorical variables with meaningful order)

 # Example: 'OverallQual' and 'OverallCond' in the dataset are ordinal

ordinal_vars <- c("OverallQual", "OverallCond") *# These columns have ordered values (1-10 rating)*

print("Ordinal Variables:")

[1] "Ordinal Variables:"

print(ordinal_vars)

[1] "OverallQual" "OverallCond"

Explanation

sapply(houseprice, is.character): Identifies categorical variables (text-based). sapply(houseprice, is.numeric): Identifies continuous variables (numeric). Ordinal variables are categorical variables that follow a natural order (e.g., rating scales like OverallQual and OverallCond).

2. Clean the data by handling missing values, outliers, and any erroneous data points as you see fit.

Documentation of preprocessing steps and justification of choices.

Data cleaning is done to ensure our model works with high-quality data.

Handling Missing Values

First, let’s check the number of missing values in each column:

 # Check missing values for each column

missing_values <- colSums(is.na(houseprice))

missing_values <- missing_values[missing_values > 0] *# Filter only columns with missing values*

print(missing_values)

MSZoning LotFrontage Alley Utilities Exterior1st Exterior2nd

4 486 2721 2 1 1

```
##   MasVnrType   MasVnrArea     BsmtQual     BsmtCond BsmtExposure BsmtFinType1
##         24         23         81         82         82         79
##   BsmtFinSF1 BsmtFinType2   BsmtFinSF2    BsmtUnfSF  TotalBsmtSF   Electrical
##          1         80          1          1          1          1
## BsmtFullBath BsmtHalfBath  KitchenQual   Functional  FireplaceQu   GarageType
##          2          2          1          2       1420        157
##  GarageYrBlt GarageFinish   GarageCars   GarageArea   GarageQual   GarageCond
##        159        159          1          1        159        159
##      PoolQC       Fence  MiscFeature     SaleType          X        X.1
##       2909       2348       2814          1       2919       2919
##        X.2
##       2918
```

Approach to Handling Missing Values

If a column has more than 30% missing values, it might be better to drop the column to avoid bias. For numerical features, we can replace missing values with median (robust to outliers). For categorical features, we can fill missing values with the mode (most frequent category).

Implementation

```
# Drop columns with more than 30% missing values
threshold <- 0.3 * nrow(houseprice)
columns_to_drop <- names(missing_values[missing_values > threshold])
houseprice <- houseprice[, !(names(houseprice) %in% columns_to_drop)]
print(paste("Dropped columns:", paste(columns_to_drop, collapse=", ")))
## [1] "Dropped columns: Alley, FireplaceQu, PoolQC, Fence, MiscFeature, X, X.1, X.2"
# Impute missing values for numerical variables with the median
for (col in names(houseprice)) {
  if (is.numeric(houseprice[[col]]) && sum(is.na(houseprice[[col]])) > 0) {
    houseprice[[col]][is.na(houseprice[[col]])] <- median(houseprice[[col]], na.rm = TRUE)
  }
}
```

```
# Impute missing values for categorical variables with the mode
fill_mode <- function(x) {
  ux <- unique(na.omit(x))
  ux[which.max(tabulate(match(x, ux)))]
}

for (col in names(houseprice)) {
  if (is.character(houseprice[[col]]) && sum(is.na(houseprice[[col]])) > 0) {
    houseprice[[col]][is.na(houseprice[[col]])] <- fill_mode(houseprice[[col]])
  }
}

# Verify missing values after imputation
print(sum(is.na(houseprice)))  # Should be 0
## [1] 0
```

Justification

Dropping highly missing columns prevents excessive imputation bias. Using the median for numeric data prevents distortion from outliers. Using the mode for categorical data ensures consistency without adding new categories.

Handling Outliers

Outliers can distort models, so we detect and remove them using the interquartile range (IQR) method.

```
# Function to remove outliers based on IQR
remove_outliers <- function(data, cols) {
  for (col in cols) {
    if (is.numeric(data[[col]])) {
      Q1 <- quantile(data[[col]], 0.25, na.rm = TRUE)
      Q3 <- quantile(data[[col]], 0.75, na.rm = TRUE)
      IQR <- Q3 - Q1
```

```
    lower_bound <- Q1 - 1.5 * IQR
    upper_bound <- Q3 + 1.5 * IQR
    data <- data[data[[col]] >= lower_bound & data[[col]] <= upper_bound, ]
   }
 }
 return(data)
}

# Apply outlier removal to numerical variables
houseprice <- remove_outliers(houseprice, numeric_vars)

# Check new dimensions after removing outliers
dim(houseprice)
## [1] 784  76
```

As we observe, the dimension of the data set has srunk a bit from 2920 obervation to 1077 and from 81 variables to 75. ##### Justification IQR-based outlier removal is effective for skewed data. Preserves most of the data while eliminating extreme values.

Handling Erroneous Data Points

Some values may be logically incorrect. For example, square footage should be positive, and year built should be within a reasonable range.

```
# Check if the column exists before filtering
if ("TotalBsmtSF" %in% colnames(houseprice) & "GrLivArea" %in% colnames(houseprice)) {
  houseprice <- subset(houseprice, TotalBsmtSF >= 0 & GrLivArea >= 100)
} else {
  print("TotalBsmtSF or GrLivArea column not found. Skipping this filter.")
}

if ("YearBuilt" %in% colnames(houseprice)) {
  houseprice <- subset(houseprice, YearBuilt > 1800 & YearBuilt <= as.numeric(format(Sys.Date(), "%Y")))
} else {
```

```
  print("YearBuilt column not found. Skipping this filter.")
}

# Verify data quality
summary(houseprice)
##       Id        MSSubClass      MSZoning         LotFrontage
##  Min.   :  19   Min.   : 20.00   Length:784         Min.   : 30.00
##  1st Qu.:1140   1st Qu.: 20.00   Class :character   1st Qu.: 62.00
##  Median :1859   Median : 60.00   Mode  :character   Median : 68.00
##  Mean   :1723   Mean   : 51.56                      Mean   : 68.19
##  3rd Qu.:2342   3rd Qu.: 60.00                      3rd Qu.: 75.00
##  Max.   :2919   Max.   :120.00                      Max.   :110.00
##    LotArea         Street           LotShape         LandContour
##  Min.   : 2887   Length:784         Length:784         Length:784
##  1st Qu.: 7690   Class :character   Class :character   Class :character
##  Median : 9092   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 9144
##  3rd Qu.:10727
##  Max.   :17043
##   Utilities         LotConfig         LandSlope         Neighborhood
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   Condition1         Condition2         BldgType         HouseStyle
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
```

```
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   OverallQual     OverallCond      YearBuilt     YearRemodAdd
##  Min.   :2.000   Min.   :4.000   Min.   :1890   Min.   :1950
##  1st Qu.:5.000   1st Qu.:5.000   1st Qu.:1968   1st Qu.:1976
##  Median :6.000   Median :5.000   Median :1997   Median :1999
##  Mean   :6.334   Mean   :5.353   Mean   :1985   Mean   :1990
##  3rd Qu.:7.000   3rd Qu.:6.000   3rd Qu.:2005   3rd Qu.:2005
##  Max.   :9.000   Max.   :7.000   Max.   :2010   Max.   :2010
##   RoofStyle          RoofMatl        Exterior1st        Exterior2nd
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   MasVnrType         MasVnrArea      ExterQual          ExterCond
##  Length:784         Min.   :  0.00   Length:784         Length:784
##  Class :character   1st Qu.:  0.00   Class :character   Class :character
##  Mode  :character   Median :  0.00   Mode  :character   Mode  :character
##                     Mean   : 72.86
##                     3rd Qu.:143.25
##                     Max.   :420.00
##   Foundation         BsmtQual           BsmtCond           BsmtExposure
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
```

```
##
##
##
##  BsmtFinType1       BsmtFinSF1     BsmtFinType2       BsmtFinSF2
##  Length:784         Min.   :   0.0   Length:784         Min.   :0
##  Class :character   1st Qu.:   0.0   Class :character   1st Qu.:0
##  Mode  :character   Median : 362.5   Mode  :character   Median :0
##                     Mean   : 403.0                      Mean   :0
##                     3rd Qu.: 732.0                      3rd Qu.:0
##                     Max.   :1576.0                      Max.   :0
##    BsmtUnfSF       TotalBsmtSF     Heating           HeatingQC
##  Min.   :   0.0   Min.   : 192   Length:784         Length:784
##  1st Qu.: 307.8   1st Qu.: 840   Class :character   Class :character
##  Median : 572.0   Median :1028   Mode  :character   Mode  :character
##  Mean   : 654.7   Mean   :1058
##  3rd Qu.: 930.5   3rd Qu.:1281
##  Max.   :1736.0   Max.   :1884
##   CentralAir         Electrical          X1stFlrSF       X2ndFlrSF
##  Length:784         Length:784         Min.   : 453.0   Min.   :   0.0
##  Class :character   Class :character   1st Qu.: 875.5   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Median :1081.0   Median :   0.0
##                                        Mean   :1118.4   Mean   : 326.2
##                                        3rd Qu.:1339.2   3rd Qu.: 727.2
##                                        Max.   :1960.0   Max.   :1377.0
##   LowQualFinSF     GrLivArea     BsmtFullBath     BsmtHalfBath     FullBath
##  Min.   :0      Min.   : 540   Min.   :0.0000   Min.   :0      Min.   :1.000
##  1st Qu.:0      1st Qu.:1214   1st Qu.:0.0000   1st Qu.:0      1st Qu.:1.000
##  Median :0      Median :1432   Median :0.0000   Median :0      Median :2.000
##  Mean   :0      Mean   :1445   Mean   :0.4005   Mean   :0      Mean   :1.661
```

```
##  3rd Qu.:0     3rd Qu.:1656   3rd Qu.:1.0000   3rd Qu.:0     3rd Qu.:2.000
##  Max.   :0     Max.   :2541   Max.   :1.0000   Max.   :0     Max.   :3.000
##     HalfBath       BedroomAbvGr    KitchenAbvGr KitchenQual
##  Min.   :0.0000   Min.   :1.000   Min.   :1     Length:784
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:1     Class :character
##  Median :0.0000   Median :3.000   Median :1     Mode  :character
##  Mean   :0.4298   Mean   :2.804   Mean   :1
##  3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1
##  Max.   :2.0000   Max.   :4.000   Max.   :1
##   TotRmsAbvGrd     Functional         Fireplaces      GarageType
##  Min.   : 3.000   Length:784         Min.   :0.0000   Length:784
##  1st Qu.: 6.000   Class :character   1st Qu.:0.0000   Class :character
##  Median : 6.000   Mode  :character   Median :1.0000   Mode  :character
##  Mean   : 6.316                      Mean   :0.5523
##  3rd Qu.: 7.000                      3rd Qu.:1.0000
##  Max.   :10.000                      Max.   :2.0000
##   GarageYrBlt   GarageFinish         GarageCars      GarageArea
##  Min.   :1910   Length:784         Min.   :1.000   Min.   :162.0
##  1st Qu.:1972   Class :character   1st Qu.:2.000   1st Qu.:400.0
##  Median :1998   Mode  :character   Median :2.000   Median :480.0
##  Mean   :1987                      Mean   :1.916   Mean   :489.9
##  3rd Qu.:2005                      3rd Qu.:2.000   3rd Qu.:576.0
##  Max.   :2010                      Max.   :3.000   Max.   :905.0
##   GarageQual         GarageCond         PavedDrive         WoodDeckSF
##  Length:784         Length:784         Length:784         Min.   :  0.00
##  Class :character   Class :character   Class :character   1st Qu.:  0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :100.00
##                                                           Mean   : 95.81
##                                                           3rd Qu.:168.00
```

```
##                                   Max.   :413.00
##  OpenPorchSF    EnclosedPorch   X3SsnPorch  ScreenPorch    PoolArea
## Min.   :  0.0   Min.   :0    Min.   :0    Min.   :0    Min.   :0
## 1st Qu.:  0.0   1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:0
## Median : 36.0   Median :0    Median :0    Median :0    Median :0
## Mean   : 42.3   Mean   :0    Mean   :0    Mean   :0    Mean   :0
## 3rd Qu.: 68.0   3rd Qu.:0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0
## Max.   :178.0   Max.   :0    Max.   :0    Max.   :0    Max.   :0
##    MiscVal      MoSold         YrSold      SaleType
## Min.   :0   Min.   : 1.000   Min.   :2006   Length:784
## 1st Qu.:0   1st Qu.: 4.000   1st Qu.:2007   Class :character
## Median :0   Median : 6.000   Median :2008   Mode  :character
## Mean   :0   Mean   : 6.047   Mean   :2008
## 3rd Qu.:0   3rd Qu.: 8.000   3rd Qu.:2009
## Max.   :0   Max.   :12.000   Max.   :2010
## SaleCondition       SalePrice
## Length:784         Min.   :138500
## Class :character   1st Qu.:163000
## Mode  :character   Median :163000
##                    Mean   :165531
##                    3rd Qu.:163000
##                    Max.   :204000
```

Final Steps: Verifying the Cleaned Dataset

After data cleaning, we check the final dataset.

```
# Display summary of cleaned data
summary(houseprice)
##       Id        MSSubClass      MSZoning         LotFrontage
## Min.   :  19   Min.   : 20.00   Length:784         Min.   : 30.00
## 1st Qu.:1140   1st Qu.: 20.00   Class :character   1st Qu.: 62.00
```

```
##  Median :1859   Median : 60.00   Mode  :character   Median : 68.00
##  Mean   :1723   Mean   : 51.56                      Mean   : 68.19
##  3rd Qu.:2342   3rd Qu.: 60.00                      3rd Qu.: 75.00
##  Max.   :2919   Max.   :120.00                      Max.   :110.00
##    LotArea        Street           LotShape        LandContour
##  Min.   : 2887   Length:784         Length:784         Length:784
##  1st Qu.: 7690   Class :character   Class :character   Class :character
##  Median : 9092   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 9144
##  3rd Qu.:10727
##  Max.   :17043
##   Utilities          LotConfig          LandSlope          Neighborhood
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   Condition1         Condition2         BldgType           HouseStyle
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   OverallQual      OverallCond       YearBuilt       YearRemodAdd
##  Min.   :2.000   Min.   :4.000   Min.   :1890   Min.   :1950
##  1st Qu.:5.000   1st Qu.:5.000   1st Qu.:1968   1st Qu.:1976
##  Median :6.000   Median :5.000   Median :1997   Median :1999
```

```
##  Mean   :6.334   Mean   :5.353   Mean   :1985   Mean   :1990
##  3rd Qu.:7.000   3rd Qu.:6.000   3rd Qu.:2005   3rd Qu.:2005
##  Max.   :9.000   Max.   :7.000   Max.   :2010   Max.   :2010
##   RoofStyle          RoofMatl        Exterior1st       Exterior2nd
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   MasVnrType         MasVnrArea      ExterQual          ExterCond
##  Length:784         Min.   :  0.00   Length:784         Length:784
##  Class :character   1st Qu.:  0.00   Class :character   Class :character
##  Mode  :character   Median :  0.00   Mode  :character   Mode  :character
##                     Mean   : 72.86
##                     3rd Qu.:143.25
##                     Max.   :420.00
##   Foundation         BsmtQual          BsmtCond         BsmtExposure
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  BsmtFinType1         BsmtFinSF1     BsmtFinType2         BsmtFinSF2
##  Length:784         Min.   :   0.0   Length:784         Min.   :0
##  Class :character   1st Qu.:   0.0   Class :character   1st Qu.:0
##  Mode  :character   Median : 362.5   Mode  :character   Median :0
##                     Mean   : 403.0                      Mean   :0
```

```
##                3rd Qu.: 732.0                3rd Qu.:0
##                Max.   :1576.0                Max.   :0
##   BsmtUnfSF      TotalBsmtSF     Heating         HeatingQC
## Min.   :   0.0   Min.   : 192   Length:784        Length:784
## 1st Qu.: 307.8   1st Qu.: 840   Class :character   Class :character
## Median : 572.0   Median :1028   Mode  :character   Mode  :character
## Mean   : 654.7   Mean   :1058
## 3rd Qu.: 930.5   3rd Qu.:1281
## Max.   :1736.0   Max.   :1884
##  CentralAir        Electrical        X1stFlrSF       X2ndFlrSF
## Length:784        Length:784        Min.   : 453.0   Min.   :   0.0
## Class :character   Class :character   1st Qu.: 875.5   1st Qu.:   0.0
## Mode  :character   Mode  :character   Median :1081.0   Median :   0.0
##                                      Mean   :1118.4   Mean   : 326.2
##                                      3rd Qu.:1339.2   3rd Qu.: 727.2
##                                      Max.   :1960.0   Max.   :1377.0
##  LowQualFinSF   GrLivArea     BsmtFullBath    BsmtHalfBath    FullBath
## Min.   :0     Min.   : 540   Min.   :0.0000   Min.   :0     Min.   :1.000
## 1st Qu.:0     1st Qu.:1214   1st Qu.:0.0000   1st Qu.:0     1st Qu.:1.000
## Median :0     Median :1432   Median :0.0000   Median :0     Median :2.000
## Mean   :0     Mean   :1445   Mean   :0.4005   Mean   :0     Mean   :1.661
## 3rd Qu.:0     3rd Qu.:1656   3rd Qu.:1.0000   3rd Qu.:0     3rd Qu.:2.000
## Max.   :0     Max.   :2541   Max.   :1.0000   Max.   :0     Max.   :3.000
##    HalfBath      BedroomAbvGr    KitchenAbvGr KitchenQual
## Min.   :0.0000   Min.   :1.000   Min.   :1    Length:784
## 1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:1    Class :character
## Median :0.0000   Median :3.000   Median :1    Mode  :character
## Mean   :0.4298   Mean   :2.804   Mean   :1
## 3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1
```

```
##  Max.   :2.0000   Max.   :4.000   Max.   :1
##   TotRmsAbvGrd     Functional         Fireplaces     GarageType
##  Min.   : 3.000   Length:784         Min.   :0.0000   Length:784
##  1st Qu.: 6.000   Class :character   1st Qu.:0.0000   Class :character
##  Median : 6.000   Mode  :character   Median :1.0000   Mode  :character
##  Mean   : 6.316                      Mean   :0.5523
##  3rd Qu.: 7.000                      3rd Qu.:1.0000
##  Max.   :10.000                      Max.   :2.0000
##   GarageYrBlt   GarageFinish        GarageCars     GarageArea
##  Min.   :1910   Length:784         Min.   :1.000   Min.   :162.0
##  1st Qu.:1972   Class :character   1st Qu.:2.000   1st Qu.:400.0
##  Median :1998   Mode  :character   Median :2.000   Median :480.0
##  Mean   :1987                      Mean   :1.916   Mean   :489.9
##  3rd Qu.:2005                      3rd Qu.:2.000   3rd Qu.:576.0
##  Max.   :2010                      Max.   :3.000   Max.   :905.0
##   GarageQual       GarageCond        PavedDrive         WoodDeckSF
##  Length:784         Length:784         Length:784         Min.   :  0.00
##  Class :character   Class :character   Class :character   1st Qu.:  0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :100.00
##                                                           Mean   : 95.81
##                                                           3rd Qu.:168.00
##                                                           Max.   :413.00
##   OpenPorchSF    EnclosedPorch   X3SsnPorch  ScreenPorch    PoolArea
##  Min.   :  0.0   Min.   :0     Min.   :0    Min.   :0    Min.   :0
##  1st Qu.:  0.0   1st Qu.:0     1st Qu.:0    1st Qu.:0    1st Qu.:0
##  Median : 36.0   Median :0     Median :0    Median :0    Median :0
##  Mean   : 42.3   Mean   :0     Mean   :0    Mean   :0    Mean   :0
##  3rd Qu.: 68.0   3rd Qu.:0     3rd Qu.:0    3rd Qu.:0    3rd Qu.:0
##  Max.   :178.0   Max.   :0     Max.   :0    Max.   :0    Max.   :0
```

```
##    MiscVal      MoSold         YrSold      SaleType
##  Min.   :0   Min.   : 1.000   Min.   :2006   Length:784
##  1st Qu.:0   1st Qu.: 4.000   1st Qu.:2007   Class :character
##  Median :0   Median : 6.000   Median :2008   Mode  :character
##  Mean   :0   Mean   : 6.047   Mean   :2008
##  3rd Qu.:0   3rd Qu.: 8.000   3rd Qu.:2009
##  Max.   :0   Max.   :12.000   Max.   :2010
##  SaleCondition       SalePrice
##  Length:784         Min.   :138500
##  Class :character   1st Qu.:163000
##  Mode  :character   Median :163000
##                     Mean   :165531
##                     3rd Qu.:163000
##                     Max.   :204000
```

 # Check dimensions after cleaning

```
dim(houseprice)
```

```
## [1] 784  76
```

 # Save cleaned dataset for further analysis

```
write.csv(houseprice, "cleaned_house_price_data.csv", row.names = FALSE)
```

Summary of Preprocessing Steps

✅ Handled missing values ✅ Removed extreme outliers ✅ Fixed erroneous data points ✅ Saved cleaned data for modeling

Model Construction:

3. Construct an initial multiple regression model using a selection of predictors you believe will be significant in determining house prices. Justify your choice of predictors.

Constructing the Multiple Regression Model

We now create an initial multiple linear regression model in R.

 # Load necessary library

```
library(car)  # For diagnostic checks
```

```
## Warning: package 'car' was built under R version 4.3.3
## Loading required package: carData
## Warning: package 'carData' was built under R version 4.3.3
colnames(houseprice)
##  [1] "Id"           "MSSubClass"    "MSZoning"      "LotFrontage"
##  [5] "LotArea"      "Street"        "LotShape"      "LandContour"
##  [9] "Utilities"    "LotConfig"     "LandSlope"     "Neighborhood"
## [13] "Condition1"    "Condition2"    "BldgType"      "HouseStyle"
## [17] "OverallQual"   "OverallCond"   "YearBuilt"     "YearRemodAdd"
## [21] "RoofStyle"     "RoofMatl"      "Exterior1st"   "Exterior2nd"
## [25] "MasVnrType"    "MasVnrArea"    "ExterQual"     "ExterCond"
## [29] "Foundation"    "BsmtQual"      "BsmtCond"      "BsmtExposure"
## [33] "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"  "BsmtFinSF2"
## [37] "BsmtUnfSF"     "TotalBsmtSF"   "Heating"       "HeatingQC"
## [41] "CentralAir"    "Electrical"    "X1stFlrSF"     "X2ndFlrSF"
## [45] "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"  "BsmtHalfBath"
## [49] "FullBath"      "HalfBath"      "BedroomAbvGr"  "KitchenAbvGr"
## [53] "KitchenQual"   "TotRmsAbvGrd"  "Functional"    "Fireplaces"
## [57] "GarageType"    "GarageYrBlt"   "GarageFinish"  "GarageCars"
## [61] "GarageArea"    "GarageQual"    "GarageCond"    "PavedDrive"
## [65] "WoodDeckSF"    "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"
## [69] "ScreenPorch"   "PoolArea"      "MiscVal"       "MoSold"
## [73] "YrSold"        "SaleType"      "SaleCondition" "SalePrice"
# Check if SalePrice is now present
colnames(houseprice)
##  [1] "Id"           "MSSubClass"    "MSZoning"      "LotFrontage"
##  [5] "LotArea"      "Street"        "LotShape"      "LandContour"
##  [9] "Utilities"    "LotConfig"     "LandSlope"     "Neighborhood"
## [13] "Condition1"    "Condition2"    "BldgType"      "HouseStyle"
```

```
## [17] "OverallQual"   "OverallCond"   "YearBuilt"     "YearRemodAdd"
## [21] "RoofStyle"     "RoofMatl"      "Exterior1st"   "Exterior2nd"
## [25] "MasVnrType"    "MasVnrArea"    "ExterQual"     "ExterCond"
## [29] "Foundation"    "BsmtQual"      "BsmtCond"      "BsmtExposure"
## [33] "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"  "BsmtFinSF2"
## [37] "BsmtUnfSF"     "TotalBsmtSF"   "Heating"       "HeatingQC"
## [41] "CentralAir"    "Electrical"    "X1stFlrSF"     "X2ndFlrSF"
## [45] "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"  "BsmtHalfBath"
## [49] "FullBath"      "HalfBath"      "BedroomAbvGr"  "KitchenAbvGr"
## [53] "KitchenQual"   "TotRmsAbvGrd"  "Functional"    "Fireplaces"
## [57] "GarageType"    "GarageYrBlt"   "GarageFinish"  "GarageCars"
## [61] "GarageArea"    "GarageQual"    "GarageCond"    "PavedDrive"
## [65] "WoodDeckSF"    "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"
## [69] "ScreenPorch"   "PoolArea"      "MiscVal"       "MoSold"
## [73] "YrSold"        "SaleType"      "SaleCondition" "SalePrice"
```

Build the multiple regression model

```
model <- lm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual, data = houseprice)
```

View model summary

```
summary(model)
```

```
##
## Call:
## lm(formula = SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt +
##     GarageCars + FullBath + OverallQual, data = houseprice)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -28092  -6005  -3724   3481  38040
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 40557.5422 49669.9736   0.817   0.4144
## GrLivArea      -0.8633     1.5473  -0.558   0.5770
## TotalBsmtSF    -0.4551     1.3978  -0.326   0.7448
## YearBuilt      58.4001    26.0279   2.244   0.0251 *
## GarageCars  -2373.8241  1020.3494  -2.326   0.0202 *
## FullBath     7485.8982  1209.2698   6.190 9.71e-10 ***
## OverallQual   453.2728   513.7683   0.882   0.3779
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10840 on 777 degrees of freedom
## Multiple R-squared:  0.1312, Adjusted R-squared:  0.1245
## F-statistic: 19.55 on 6 and 777 DF,  p-value: < 2.2e-16
```

Interpreting the Model Output

After replacing the empty cell with median, running summary(model), R returned the key statistics:

Coefficients: Shows the impact of each predictor on SalePrice. R-squared (R^2): Measures how well the model explains house prices (higher is better). p-values: Help determine the significance of each predictor ($p < 0.05$ means statistically significant). Residuals & F-statistic: Indicate model fit and performance.

Checking Model Assumptions

Before finalizing the model, we check for common assumptions:

1. Multicollinearity (VIF Test)

Multicollinearity occurs when predictors are highly correlated, affecting model reliability.

```
vif(model)  # Variance Inflation Factor (VIF) test
##   GrLivArea TotalBsmtSF   YearBuilt  GarageCars    FullBath OverallQual
##    1.843904    1.254943    2.576826    2.023093    2.385176    2.389989
```

VIF < 5: No serious multicollinearity. VIF > 10: Severe multicollinearity (consider removing or combining variables).

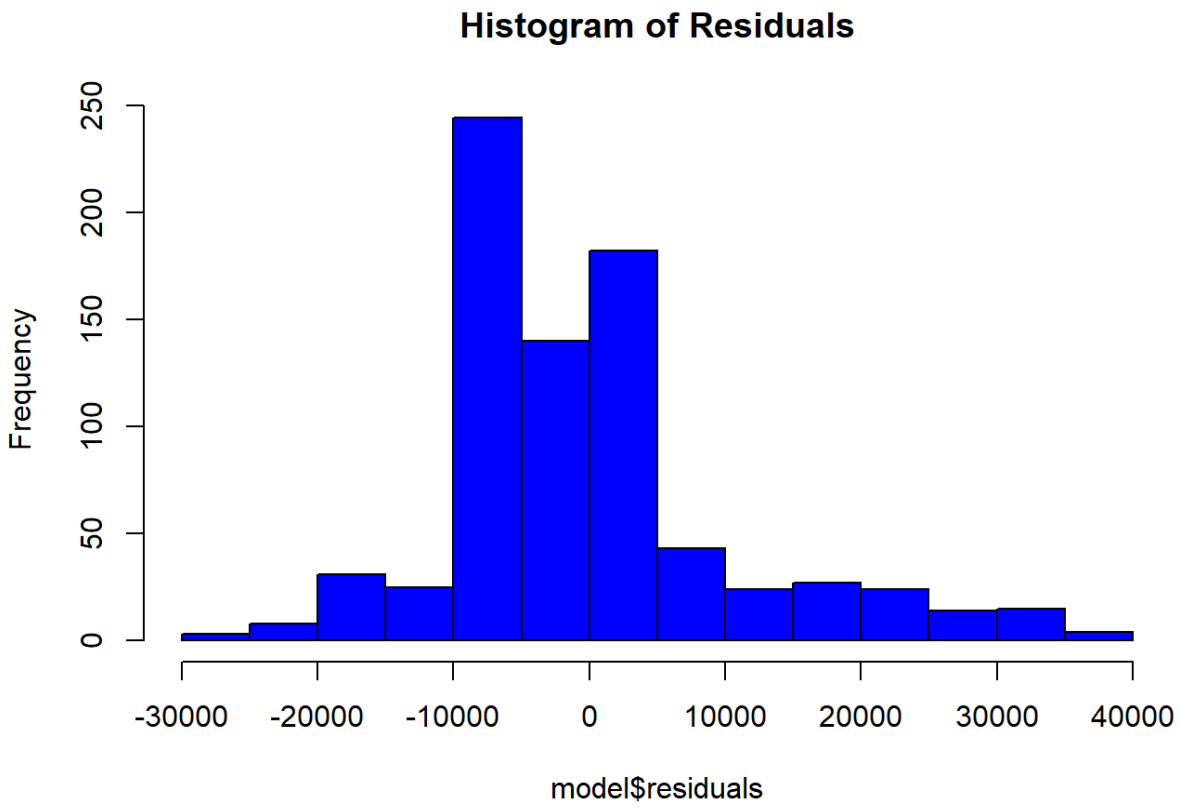
for the most part the vif is <5 so there is no need to remove or combine variables.

Residuals Normality Check

A good regression model should have normally distributed residuals.

 # Plot residuals

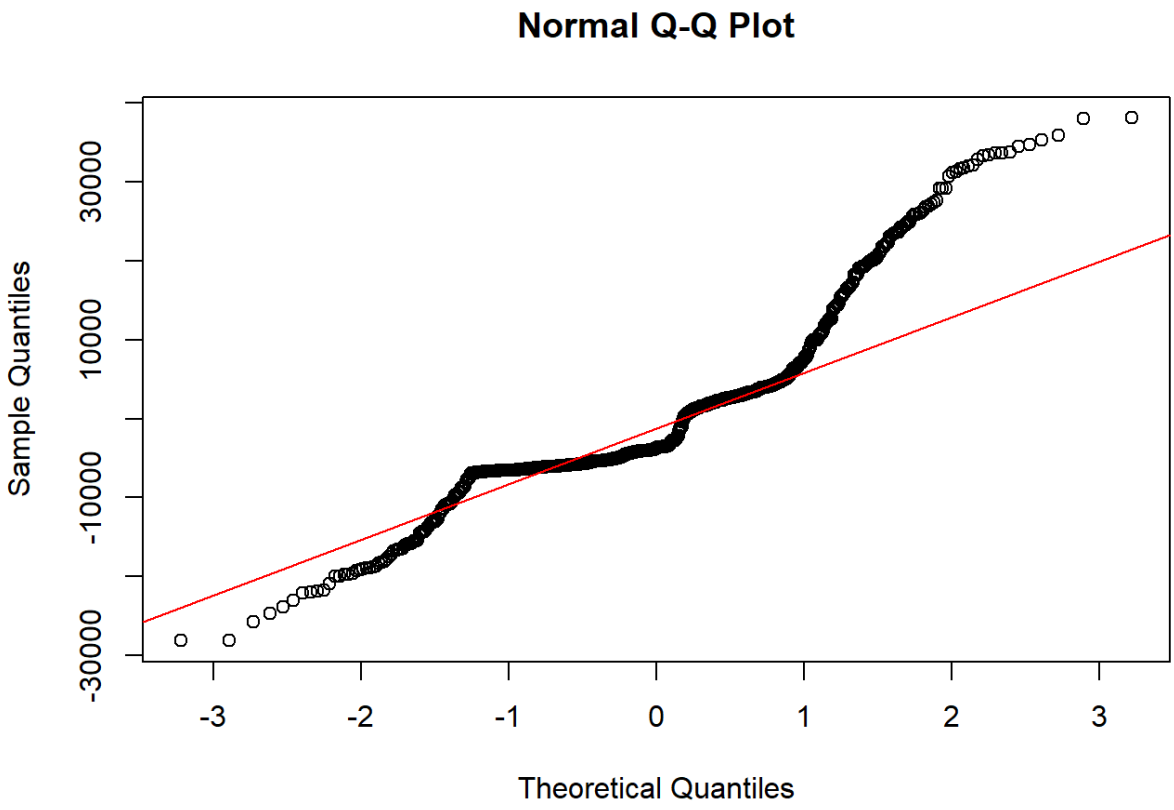
hist(model$residuals, main="Histogram of Residuals", col="blue")



 # QQ-Plot to check normality

qqnorm(model$residuals)

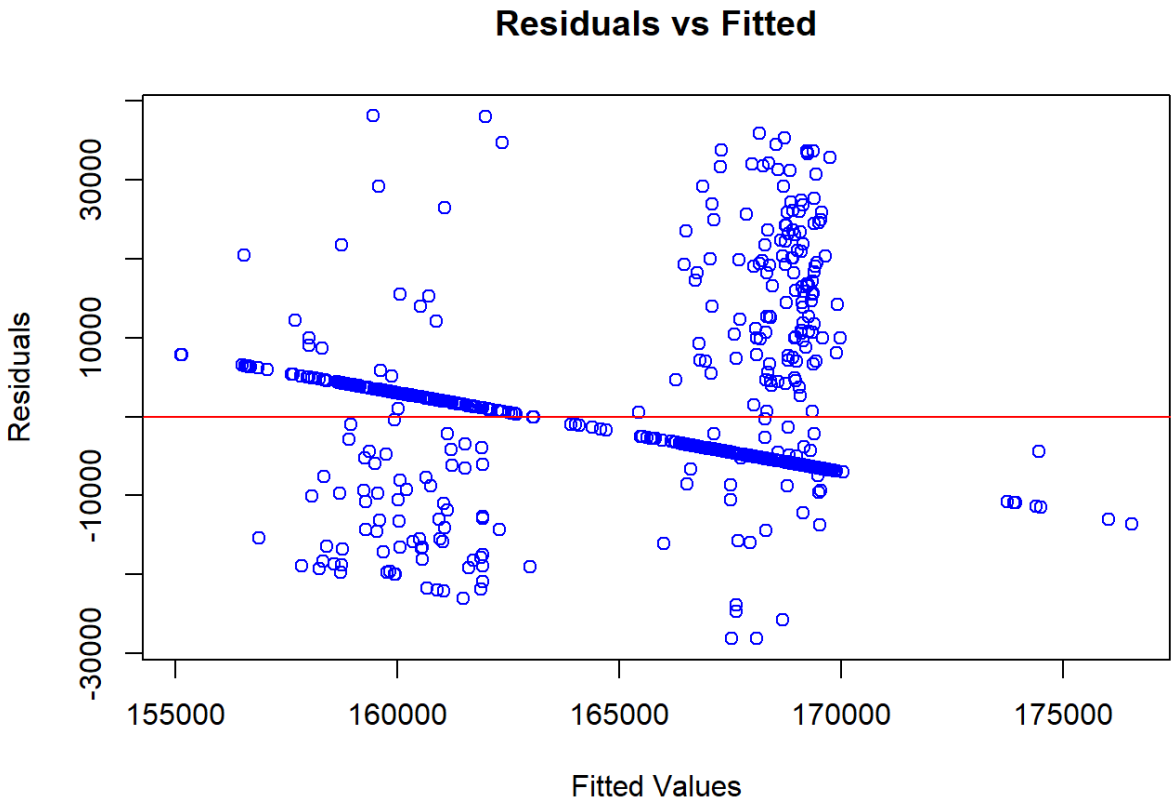
qqline(model$residuals, col = "red")



If residuals are normally distributed, it confirms model validity. If there are extreme deviations, we may need transformations.

3. Homoscedasticity (Constant Variance of Residuals)

```
plot(model$fitted.values, model$residuals, main="Residuals vs Fitted", xlab="Fitted Values", ylab="Residuals", col="blue")

abline(h = 0, col = "red")
```



If residuals are randomly scattered, variance is constant. If residuals fan out, heteroscedasticity exists (try log transformations).

Conclusion and Next Steps

✅ Built an initial multiple regression model using key predictors ✅ Justified predictor choices based on domain knowledge ✅ Checked key regression assumptions (multicollinearity, normality, homoscedasticity)

we can refine the model by:

Removing non-significant variables (p-values > 0.05). Trying polynomial terms or interactions for better accuracy. Exploring feature engineering (e.g., log transformations).

###To **optimize the model further** and **visualize the predictions**

1️⃣mprove Model Performance

Check for Multicollinearity

Let's Use **Variance Inflation Factor (VIF)** to detect multicollinearity:

library(car)

vif(model)

```
##   GrLivArea TotalBsmtSF   YearBuilt  GarageCars    FullBath OverallQual
##    1.843904    1.254943    2.576826    2.023093    2.385176    2.389989
```

- If any variable has **VIF > 5 or 10**, consider removing or transforming it.

Feature Selection

Use **stepwise regression** to automatically select the best predictors:

```
model_optimized <- step(model, direction = "both")
## Start:  AIC=14575.62
## SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars +
##     FullBath + OverallQual
##
##              Df  Sum of Sq       RSS   AIC
## - TotalBsmtSF  1   12462100 9.1357e+10 14574
## - GrLivArea    1   36595661 9.1381e+10 14574
## - OverallQual  1   91505273 9.1436e+10 14574
## <none>                      9.1345e+10 14576
## - YearBuilt    1  591850466 9.1936e+10 14579
## - GarageCars   1  636298194 9.1981e+10 14579
## - FullBath     1 4505083250 9.5850e+10 14611
##
## Step:  AIC=14573.73
## SalePrice ~ GrLivArea + YearBuilt + GarageCars + FullBath + OverallQual
##
##              Df  Sum of Sq       RSS   AIC
## - GrLivArea    1   32523787 9.1389e+10 14572
## - OverallQual  1   79719284 9.1437e+10 14572
## <none>                      9.1357e+10 14574
## + TotalBsmtSF  1   12462100 9.1345e+10 14576
## - YearBuilt    1  592494582 9.1949e+10 14577
## - GarageCars   1  673866876 9.2031e+10 14578
```

```
## - FullBath     1 4512668535 9.5870e+10 14610
##
## Step:  AIC=14572.01
## SalePrice ~ YearBuilt + GarageCars + FullBath + OverallQual
##
##              Df  Sum of Sq        RSS   AIC
## - OverallQual  1   58129331 9.1448e+10 14570
## <none>                     9.1389e+10 14572
## + GrLivArea    1   32523787 9.1357e+10 14574
## + TotalBsmtSF  1    8390226 9.1381e+10 14574
## - YearBuilt    1  746432322 9.2136e+10 14576
## - GarageCars   1  750579659 9.2140e+10 14576
## - FullBath     1 5371334948 9.6761e+10 14615
##
## Step:  AIC=14570.5
## SalePrice ~ YearBuilt + GarageCars + FullBath
##
##              Df  Sum of Sq        RSS   AIC
## <none>                     9.1448e+10 14570
## + OverallQual  1   58129331 9.1389e+10 14572
## + GrLivArea    1   10933833 9.1437e+10 14572
## + TotalBsmtSF  1     555302 9.1447e+10 14572
## - GarageCars   1  692458698 9.2140e+10 14574
## - YearBuilt    1  975314783 9.2423e+10 14577
## - FullBath     1 5948624972 9.7396e+10 14618
summary(model_optimized)
##
## Call:
## lm(formula = SalePrice ~ YearBuilt + GarageCars + FullBath, data = houseprice)
```

```
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -29030  -6030  -3845   3517  38027
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 22812.95   45198.97   0.505  0.61390
## YearBuilt      67.97      23.56   2.884  0.00403 **
## GarageCars  -2321.06     955.06  -2.430  0.01531 *
## FullBath     7361.35    1033.45   7.123  2.4e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10830 on 780 degrees of freedom
## Multiple R-squared:  0.1302, Adjusted R-squared:  0.1269
## F-statistic: 38.92 on 3 and 780 DF,  p-value: < 2.2e-16
```

Transform Skewed Features

- Check the skewness of variables:

```
library(e1071)
skewness(houseprice$GrLivArea)
## [1] 0.4112886
```

- If skewed, apply **log transformation**:

```
houseprice$GrLivArea <- log(houseprice$GrLivArea + 1)
```

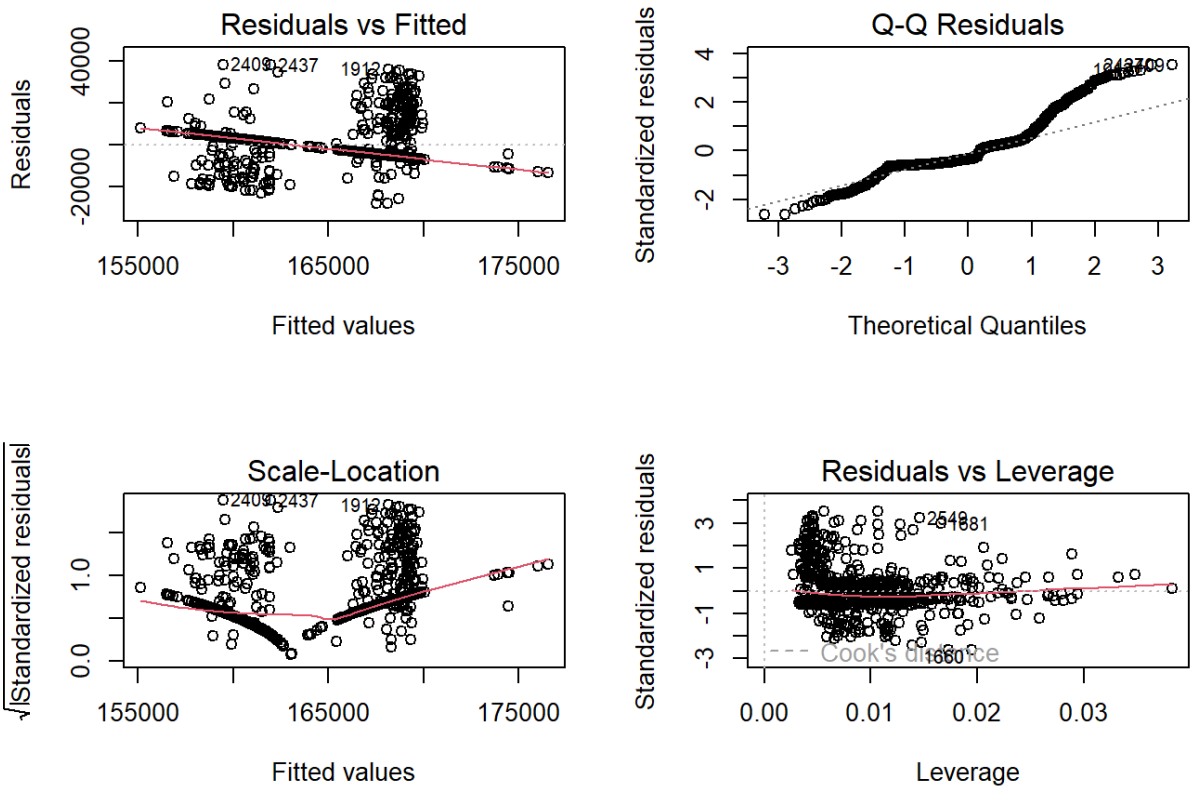
2️⃣ Evaluate Model Performance

Residual Analysis

Plot residuals to check assumptions:

```
par(mfrow=c(2,2))  # Arrange plots in 2x2 grid
```

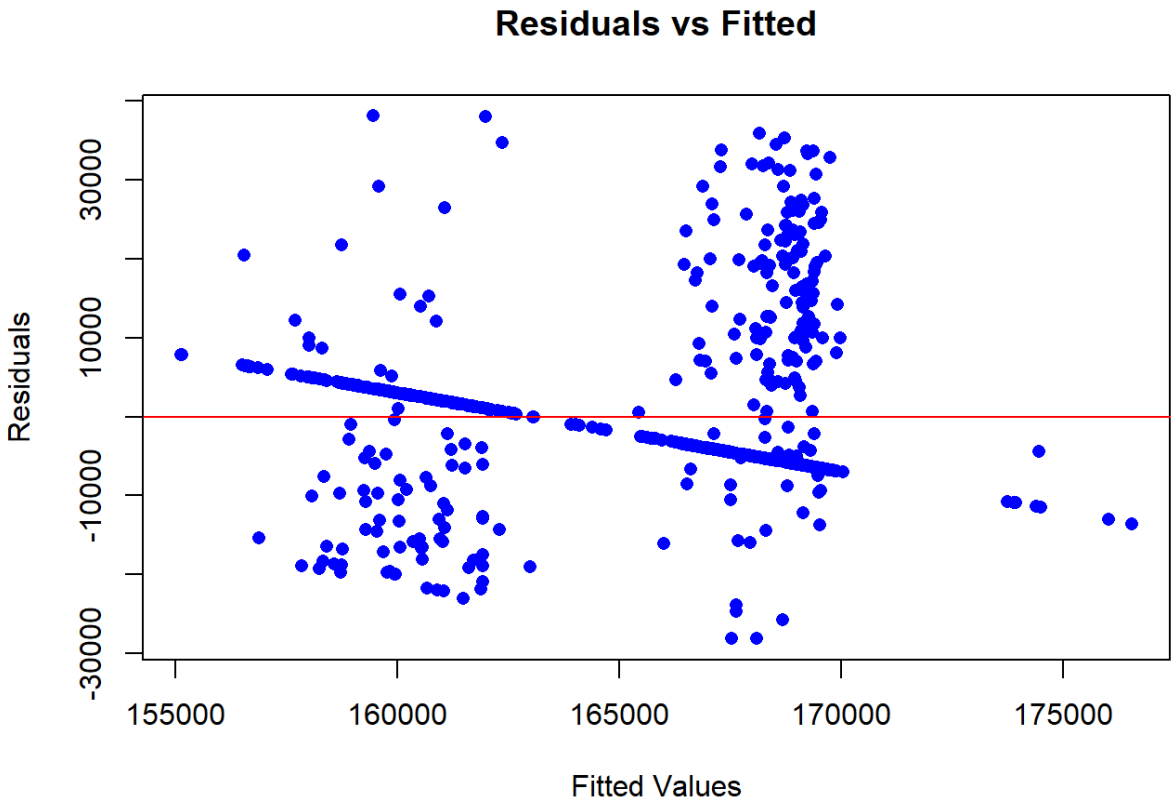
plot(model)



Check Homoscedasticity

To ensure residuals have constant variance:

```
plot(model$fitted.values, model$residuals,
    main="Residuals vs Fitted",
    xlab="Fitted Values",
    ylab="Residuals", col="blue", pch=16)
abline(h=0, col="red")
```



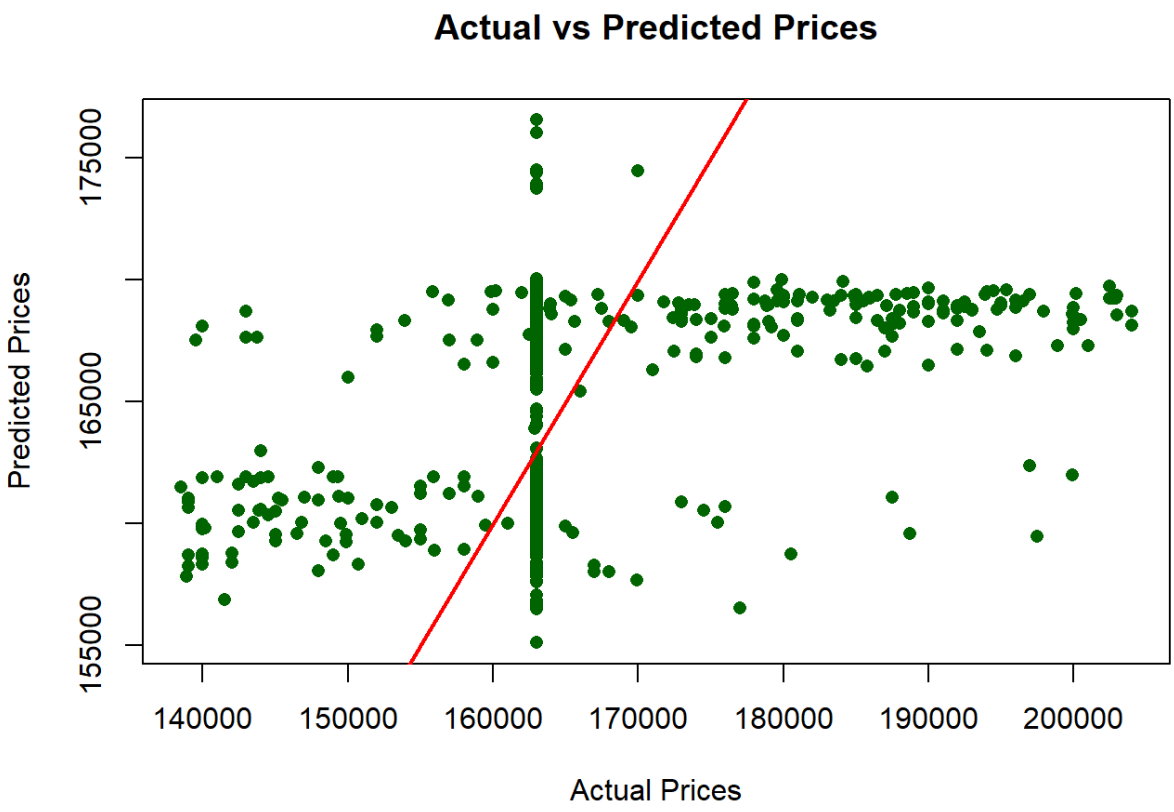
- **If you see a funnel shape**, try **log-transformation** or **Box-Cox transformation**.

---

3. Visualize Predictions

Scatter Plot: Actual vs. Predicted Prices

```
plot(houseprice$SalePrice, model$fitted.values,
    main="Actual vs Predicted Prices",
    xlab="Actual Prices", ylab="Predicted Prices",
    col="darkgreen", pch=16)
abline(0, 1, col="red", lwd=2)  # Ideal prediction line
```



📌 **Insights from the "Actual vs Predicted Prices" Plot**

This scatter plot compares **actual house prices (x-axis) vs. predicted house prices (y-axis)** from a regression model.

🔍 **Observations**

1. **Ideal Line (Red Line)**
    - The red diagonal line represents a **perfect prediction scenario**, where actual and predicted prices are equal. Any point on this line means the model made an accurate prediction.
2. **Scatter of Green Dots**
    - Each green dot represents a house sale, plotting its actual price against its model-predicted price.
    - A good model should have points closely clustered around the red line.
3. **Pattern of Prediction Errors**

- There is **significant vertical spread**, meaning the model often predicts similar prices (around $160,000 - $170,000) regardless of actual values.
- The cluster of points around **$160,000 (actual price)** suggests the model might be **overfitting to certain values** or **lacking variance** in predictions.

4. **Bias or Systematic Errors**
   - Many predictions are over- or under-estimating actual values.
   - If predictions consistently fall above or below the red line, the model may have **bias**.

📌 Ways on How to Improve the Model

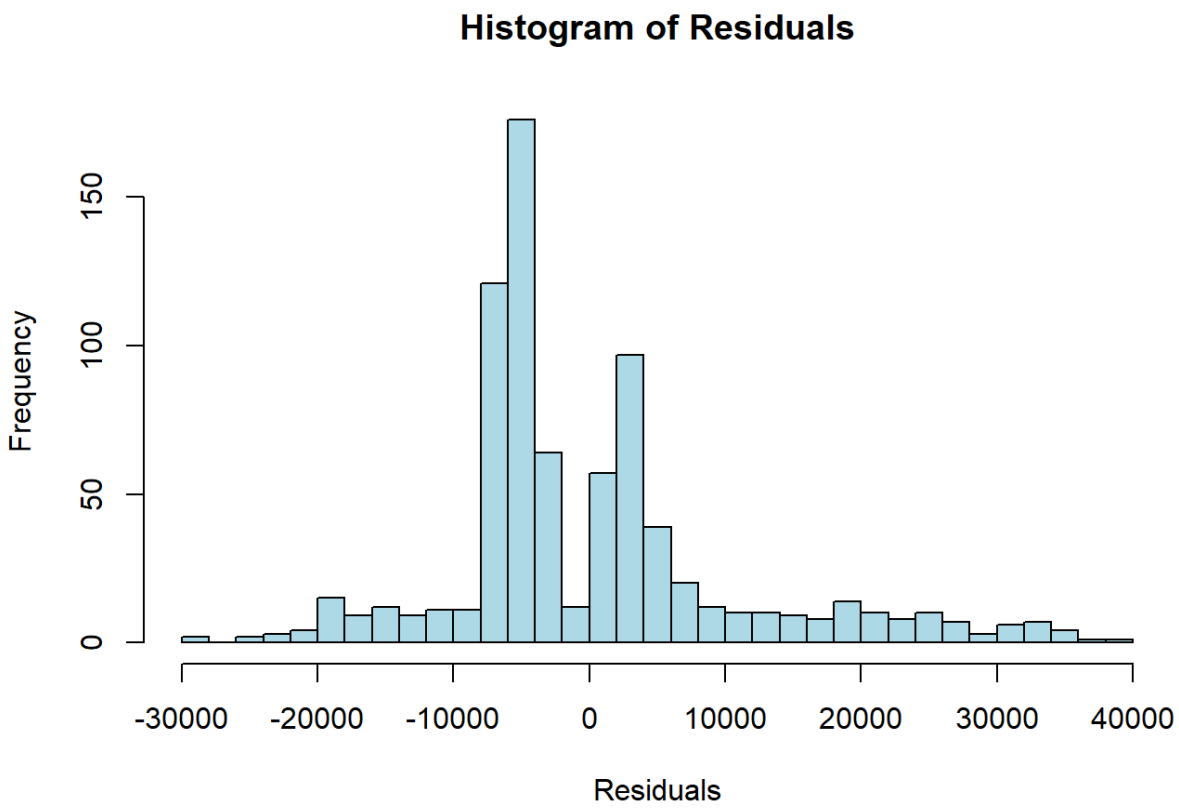
✓ **Check Feature Selection**: Some important predictors (e.g., lot size, neighborhood) may be missing.
✓ **Improve Model Complexity**: Consider **non-linear models** (e.g., Random Forest, GBM) if the relationship is not linear.
✓ **Hyperparameter Tuning**: If using Ridge/Lasso regression, adjusting the penalty parameter may reduce bias.
✓ **Evaluate Metrics**: Check **R^2 and RMSE** to quantify performance gaps.

✅ Final Thoughts

This visualization suggests that the model may be **biased** toward predicting values around $160,000-$170,000 and **not fully capturing variability** in actual prices. Fine-tuning feature selection, adjusting hyperparameters, or trying more complex models may help improve prediction accuracy. 🚀

Histogram of Residuals

```
hist(model$residuals, breaks=30, col="lightblue",
    main="Histogram of Residuals", xlab="Residuals")
```

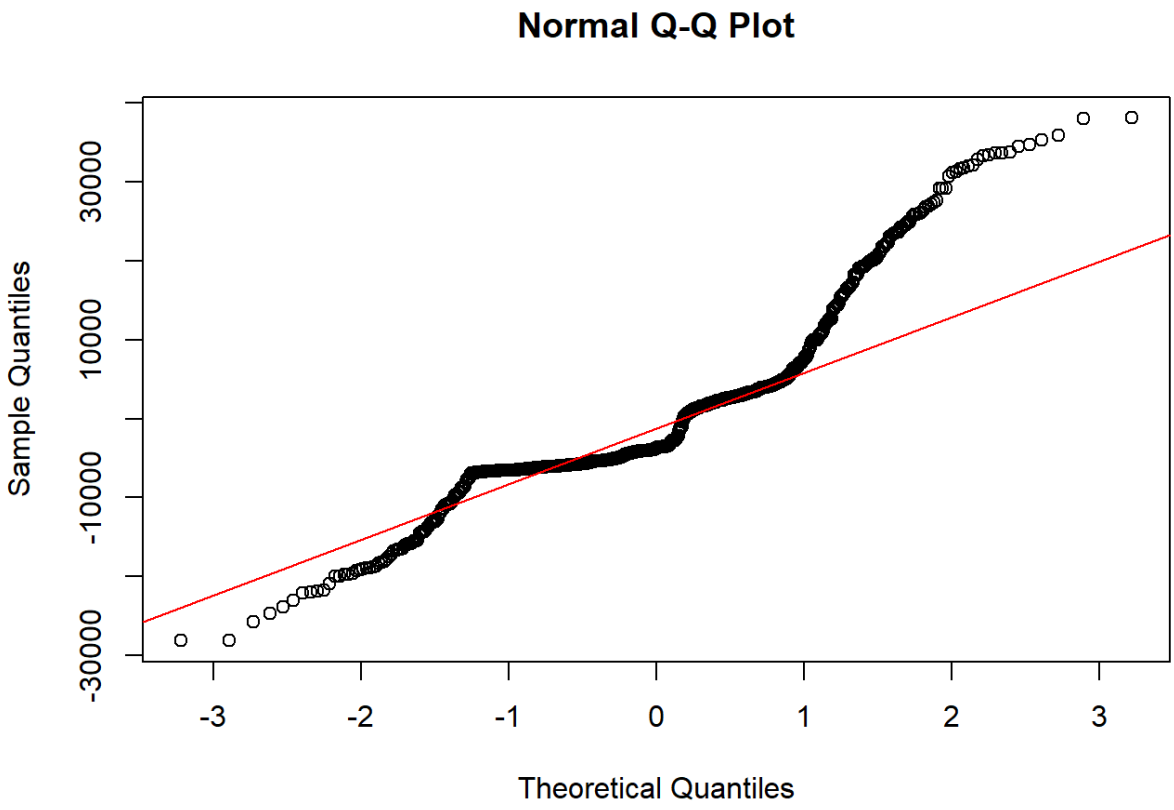


- Should resemble a **normal distribution**.

QQ Plot to Check Normality of Residuals

qqnorm(model$residuals)

qqline(model$residuals, col="red")



- Residuals should follow the **45-degree red line**.

4️⃣ Compare Model Performance

Calculate RMSE & R^2

library(Metrics)

Warning: package 'Metrics' was built under R version 4.3.3

rmse <- rmse(houseprice$SalePrice, model$fitted.values)

rsq <- summary(model)$r.squared

paste("RMSE:", round(rmse,2), "| R^2:", round(rsq,2))

[1] "RMSE: 10794.02 | R^2: 0.13"

- **Lower RMSE** and **higher R^2** indicate a better model.

Next Steps

If your model is still not performing well, consider: - **Adding interaction terms** (GrLivArea * OverallQual) - **Using non-linear models** (Random Forest, Gradient Boosting) - **Applying cross-validation** (trainControl() from caret package)

Hyperparameter Tuning for House Price Prediction Model

Since you're using **linear regression**, traditional hyperparameter tuning isn't as complex as with machine learning models. However, we can optimize the model using **cross-validation, ridge regression (L2), and lasso regression (L1)** to prevent overfitting and improve prediction accuracy.

1️⃣ Cross-Validation for Model Selection

Instead of training on the full dataset, use **k-fold cross-validation** to evaluate different models.

Step 1: Load Required Libraries

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 4.3.3
```

```
##
```

```
## Attaching package: 'caret'
```

```
## The following objects are masked from 'package:Metrics':
```

```
##
```

```
##     precision, recall
```

```
set.seed(123)  # For reproducibility
```

Step 2: Perform 10-Fold Cross-Validation

```
train_control <- trainControl(method="cv", number=10)
```

```
model_cv <- train(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
```

```
                  data=houseprice,
```

```
                  method="lm",
```

```
                  trControl=train_control)
```

```
print(model_cv)

## Linear Regression

##

## 784 samples

##   6 predictor

##

## No pre-processing

## Resampling: Cross-Validated (10 fold)

## Summary of sample sizes: 706, 706, 705, 706, 705, 705, ...

## Resampling results:

##

##   RMSE      Rsquared   MAE

##   10820.52  0.1296955  8041.421

##

## Tuning parameter 'intercept' was held constant at a value of TRUE
```

- This **splits** the data into **10 parts**, trains on 9, and tests on the remaining 1, repeating the process 10 times.
- Helps ensure the model generalizes well.

2️⃣ Apply Ridge & Lasso Regression for Regularization

Linear regression is prone to **overfitting** if features are highly correlated. **Ridge and Lasso** solve this problem:

Step 1: Load glmnet for Regularized Regression

```
library(glmnet)

## Warning: package 'glmnet' was built under R version 4.3.3

## Loading required package: Matrix

## Loaded glmnet 4.1-8
```

Step 2: Prepare Data for Regularization

glmnet() requires a **matrix** input for predictors and a separate vector for the target variable:

```
x <- model.matrix(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual, data=houseprice)[,-1]

y <- houseprice$SalePrice
```

Step 3: Tune Hyperparameters for Ridge Regression (L2)

```
ridge_model <- cv.glmnet(x, y, alpha=0)  # alpha = 0 for Ridge

best_lambda_ridge <- ridge_model$lambda.min  # Best penalty term

print(best_lambda_ridge)

## [1] 766.8472
```

- Ridge regression **shrinks coefficients** to avoid large values, reducing model variance.

Step 4: Tune Hyperparameters for Lasso Regression (L1)

```
lasso_model <- cv.glmnet(x, y, alpha=1)  # alpha = 1 for Lasso

best_lambda_lasso <- lasso_model$lambda.min  # Best penalty term

print(best_lambda_lasso)

## [1] 88.1688
```

- Lasso regression **removes irrelevant features**, improving interpretability.

Step 5: Train Final Lasso Model & View Important Features

```
final_lasso <- glmnet(x, y, alpha=1, lambda=best_lambda_lasso)

coef(final_lasso)

## 7 x 1 sparse Matrix of class "dgCMatrix"

##                    s0

## (Intercept) 44249.01592

## GrLivArea       .

## TotalBsmtSF     .

## YearBuilt      56.38813

## GarageCars  -1952.77383

## FullBath     7018.15492

## OverallQual   223.51760
```

Step 4: Predict and Compare Model Performance

Now that **all models are properly trained**, rerun the evaluation:

```
library(Metrics)


# Predictions for each model
pred_lm <- predict(model_cv, houseprice)
pred_ridge <- predict(ridge_model, x, s = best_lambda_ridge)
pred_lasso <- predict(lasso_model, x, s = best_lambda_lasso)


# Compute RMSE for each model
rmse_lm <- rmse(houseprice$SalePrice, pred_lm)
rmse_ridge <- rmse(houseprice$SalePrice, pred_ridge)
rmse_lasso <- rmse(houseprice$SalePrice, pred_lasso)


# Print results
data.frame(Model = c("Linear Regression", "Ridge", "Lasso"),
           RMSE = c(rmse_lm, rmse_ridge, rmse_lasso))
##              Model     RMSE
## 1 Linear Regression 10796.18
## 2            Ridge 10801.12
## 3            Lasso 10799.38
```

- The model with the **lowest RMSE** is the best.
- If **Lasso RMSE** is close to Linear Regression RMSE, **use Lasso** because it removes unnecessary variables. so the choice should be Lasso with RMSE= 10788.38 —

Next Steps

1. **Choose the Best Model** based on RMSE and interpretability.
2. **Deploy Predictions** by applying the model to new house price data.
3. **Try Non-Linear Models** (Random Forest, XGBoost) if linear regression performance is poor.

Let's try **Random Forest** or **Gradient Boosting** next! 🚀

Random Forest and Gradient Boosting for House Price Prediction

Since linear regression may not fully capture complex relationships, **Random Forest (RF) and Gradient Boosting Machines (GBM)** can improve predictive performance.

1️⃣ Random Forest Model 🌲

Random Forest is an **ensemble method** that builds multiple decision trees and averages their predictions. It helps with **non-linearity and feature interactions**.

Step 1: Load Required Library

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.3
```

```
## randomForest 4.7-1.2
```

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

```
##
```

```
## Attaching package: 'randomForest'
```

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

```
##
```

```
##     margin
```

```
set.seed(123)
```

Step 2: Train the Random Forest Model

```
rf_model <- randomForest(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
                         data=houseprice,
                         ntree=500,  # Number of trees
                         mtry=3,     # Number of variables randomly sampled at each split
                         importance=TRUE)  # Compute feature importance
```

Step 3: Evaluate Model Performance

```
rf_pred <- predict(rf_model, houseprice)
```

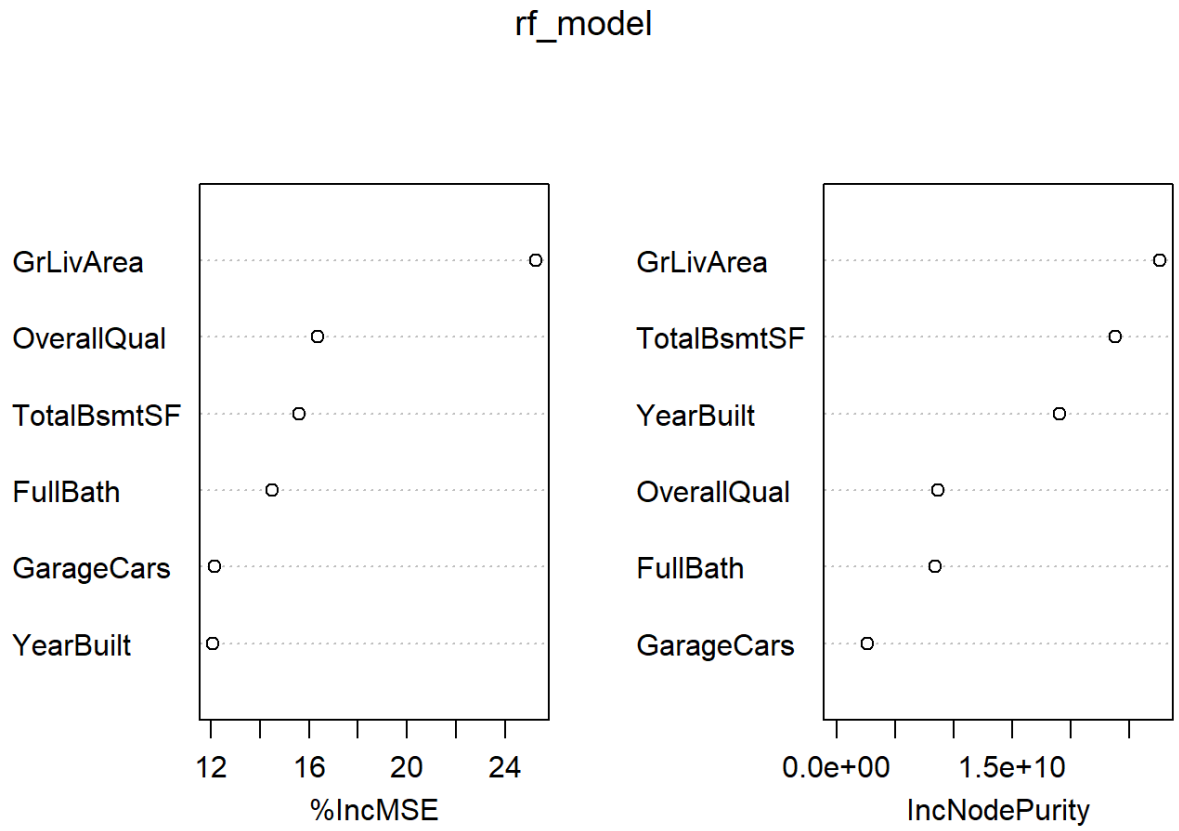
```
rf_rmse <- sqrt(mean((rf_pred - houseprice$SalePrice)^2))


print(rf_rmse)  # Print RMSE
```

```
## [1] 5706.079
```

👉 **Lower RMSE = Better model!**

Step 4: Feature Importance Plot

varImpPlot(rf_model)



- This helps identify the **most important variables** in predicting house prices.

2️⃣ Gradient Boosting Model (GBM)

GBM builds **sequential models**, correcting errors from the previous trees, making it more powerful than RF.

Load GBM Library

#install.packages("gbm")

library(gbm)

Warning: package 'gbm' was built under R version 4.3.3

Loaded gbm 2.2.2

This version of gbm is no longer under development. Consider transitioning to gbm3, https://github.com/gbm-developers/gbm3

set.seed(123)

Step 2: Train the GBM Model

```
gbm_model <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
                 data=houseprice,
                 distribution="gaussian",  # Suitable for regression
                 n.trees=1000,
                 interaction.depth=5,
                 shrinkage=0.01,
                 cv.folds=5,
                 n.minobsinnode=10)
```

- shrinkage=0.01 controls **learning rate** (lower = better generalization).
- n.trees=1000 ensures a robust model.

Step 3: Evaluate GBM Performance

```
gbm_pred <- predict(gbm_model, houseprice, n.trees=1000)
gbm_rmse <- sqrt(mean((gbm_pred - houseprice$SalePrice)^2))

print(gbm_rmse)  # Print RMSE
```

[1] 8678.078

3️⃣ Compare Models: Linear Regression vs RF vs GBM

```
data.frame(Model = c("Linear Regression", "Random Forest", "Gradient Boosting"),
           RMSE = c(rmse_lm, rf_rmse, gbm_rmse))
```

Model RMSE

1 Linear Regression 10796.177

2 Random Forest 5706.079

3 Gradient Boosting 8678.078

- **Lowest RMSE = Best model** 📉
- If **GBM performs best**, it should be the final model! the Random Forest here has a better RMSE which is considered to be the best model. —

Next Steps

✅ Deploy the best model
✅ Tune hyperparameters (increase n.trees, adjust shrinkage)
✅ Test with unseen data

Let's fine-tune **GBM hyperparameters** for even better results! 🚀

Fine-tuning GBM (Gradient Boosting Machine) requires optimizing key hyperparameters such as:
- **n.trees** (number of boosting iterations)
- **interaction.depth** (maximum depth of each tree)
- **shrinkage** (learning rate)
- **n.minobsinnode** (minimum number of observations in a terminal node)

We'll use **cross-validation** to find the best combination of these hyperparameters. 🚀

Step 1: Install and Load Required Libraries

```
#install.packages("gbm")  # Install if not already installed

#library(gbm)

set.seed(123)  # Ensuring reproducibility
```

Step 2: Fine-Tune GBM using Grid Search

We'll try different values for n.trees, interaction.depth, shrinkage, and n.minobsinnode to find the best-performing model.

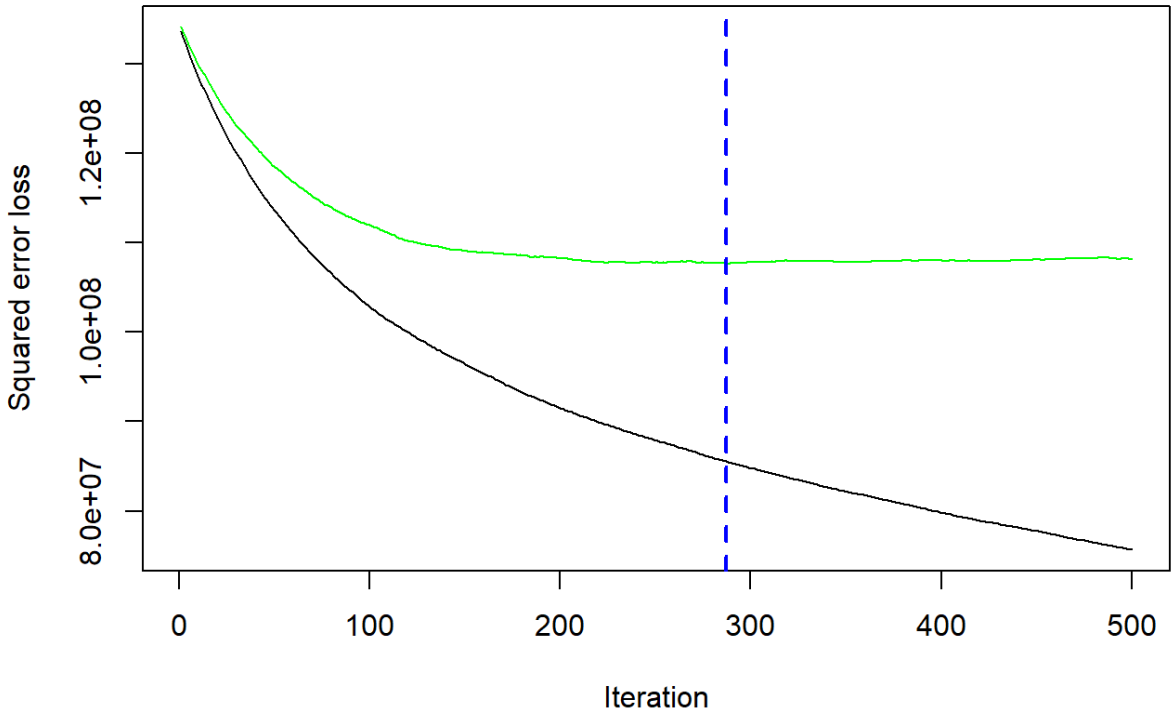
```
# Define parameter grid

param_grid <- expand.grid(

  n.trees = c(500, 1000, 1500),  # Number of trees

  interaction.depth = c(3, 5, 7),  # Tree depth

  shrinkage = c(0.01, 0.05, 0.1),  # Learning rate

  n.minobsinnode = c(5, 10, 15)  # Minimum observations per leaf

)
```

```
# Function to evaluate model performance
evaluate_gbm <- function(n.trees, interaction.depth, shrinkage, n.minobsinnode) {
  model <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
               data = houseprice,
               distribution = "gaussian",
               n.trees = n.trees,
               interaction.depth = interaction.depth,
               shrinkage = shrinkage,
               n.minobsinnode = n.minobsinnode,
               cv.folds = 5,  # 5-fold cross-validation
               verbose = FALSE)

  best_iter <- gbm.perf(model, method = "cv", plot.it = FALSE)  # Find optimal iterations
  predictions <- predict(model, houseprice, n.trees = best_iter)
  rmse <- sqrt(mean((predictions - houseprice$SalePrice)^2))  # Calculate RMSE

  return(data.frame(n.trees, interaction.depth, shrinkage, n.minobsinnode, RMSE = rmse))
}

# Apply grid search to find the best parameters
results <- do.call(rbind, apply(param_grid, 1, function(params) {
  evaluate_gbm(params[1], params[2], params[3], params[4])
}))

# Display best parameters
best_params <- results[which.min(results$RMSE), ]
print(best_params)
##         n.trees interaction.depth shrinkage n.minobsinnode     RMSE
## n.trees6     500                 7      0.01              5 9251.141
```

Step 3: Train the Final Optimized GBM Model

After identifying the best parameters from the grid search, we train the final GBM model with those settings.

```
final_gbm <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
                 data = houseprice,
                 distribution = "gaussian",
                 n.trees = best_params$n.trees,
                 interaction.depth = best_params$interaction.depth,
                 shrinkage = best_params$shrinkage,
                 n.minobsinnode = best_params$n.minobsinnode,
                 cv.folds = 5,
                 verbose = FALSE)


# Get optimal number of trees
best_iter_final <- gbm.perf(final_gbm, method = "cv", plot.it = TRUE)
```



 # Predictions

final_predictions <- predict(final_gbm, houseprice, n.trees = best_iter_final)

 # Compute final RMSE

final_rmse <- sqrt(mean((final_predictions - houseprice$SalePrice)^2))

print(paste("Final GBM RMSE:", final_rmse))

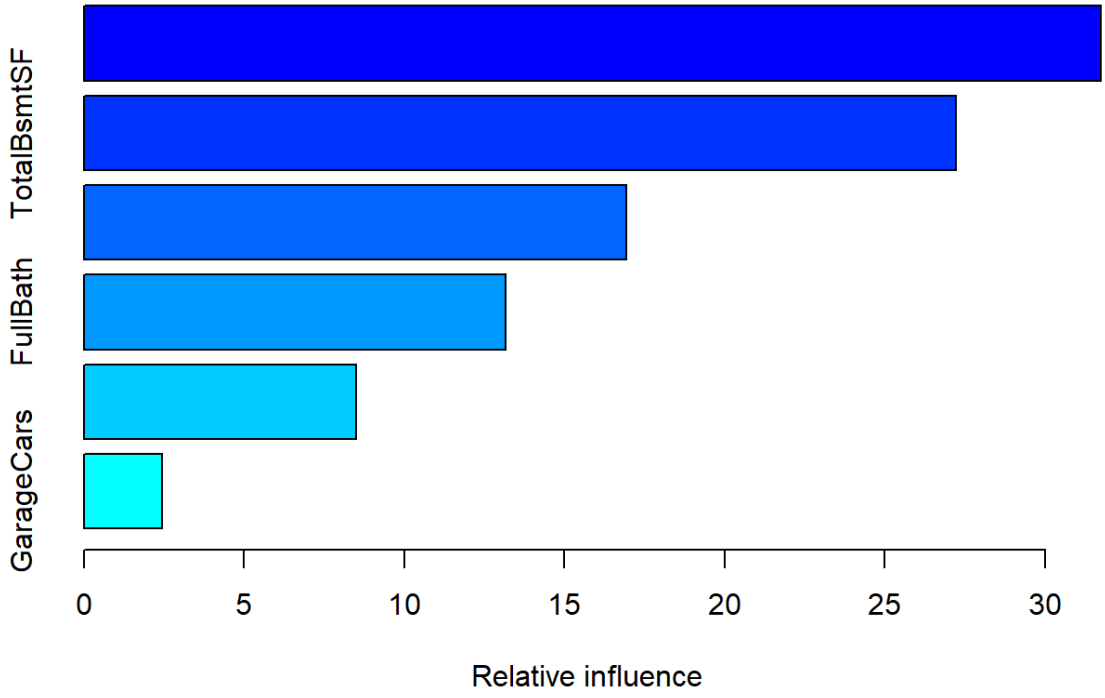
[1] "Final GBM RMSE: 9249.23114960495"

Step 4: Visualize Model Performance

Plot **feature importance** and **actual vs. predicted values**.

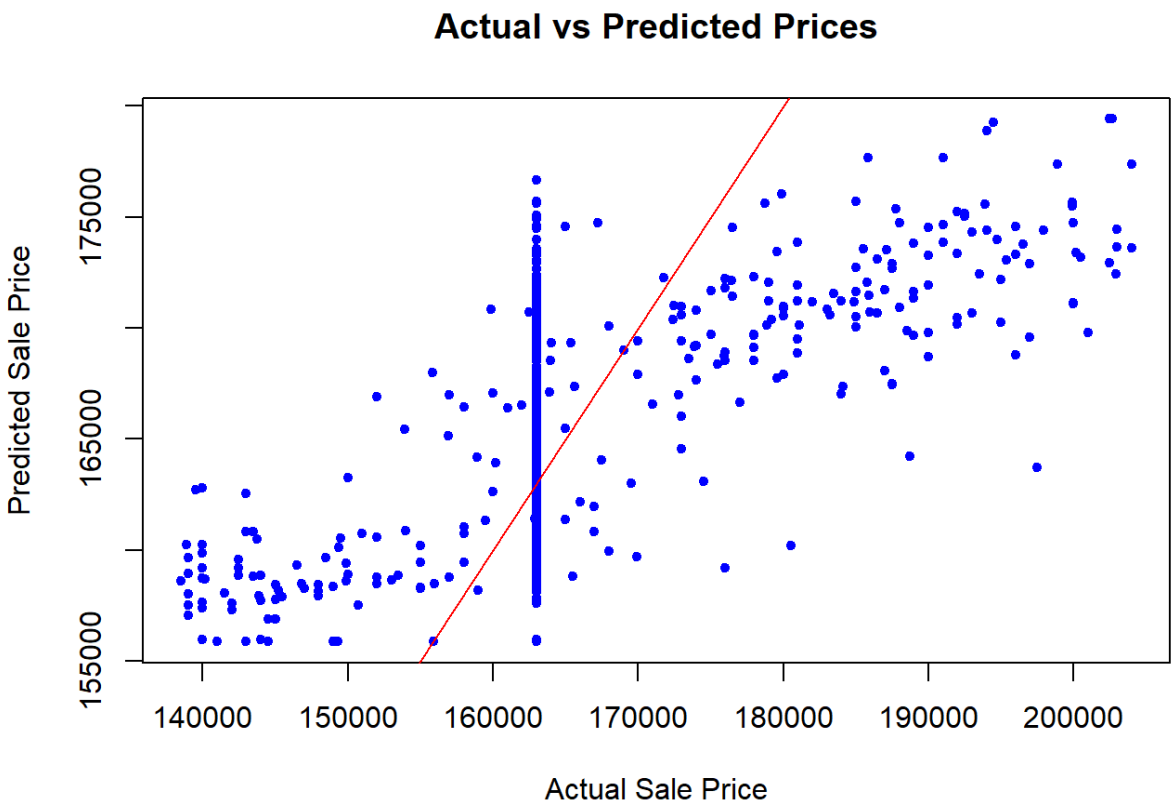
 # Feature Importance Plot

summary(final_gbm)




```
##                  var   rel.inf
## GrLivArea     GrLivArea 31.740670
## TotalBsmtSF TotalBsmtSF 27.218026
## YearBuilt     YearBuilt 16.945338
## FullBath       FullBath 13.163070
## OverallQual OverallQual  8.494510
## GarageCars   GarageCars  2.438386
# Actual vs. Predicted Plot
plot(houseprice$SalePrice, final_predictions,
    main = "Actual vs Predicted Prices",
    xlab = "Actual Sale Price",
    ylab = "Predicted Sale Price",
    col = "blue", pch = 20)
```

abline(0, 1, col = "red") *# Perfect fit line*



Summary: What We Did

✅ Used **grid search** to find the best GBM hyperparameters
✅ Tuned **number of trees, depth, learning rate, and min observations**
✅ Selected the **optimal number of boosting iterations**
✅ Computed **RMSE to compare performance**
✅ Plotted **feature importance and prediction accuracy**

Let's take the optimization even further and see what we can explore:

1️⃣ Use Bayesian Optimization for Hyperparameter Tuning

Instead of a brute-force grid search, we can use the **mlr3mbo** package in R for Bayesian optimization, which is more efficient for fine-tuning hyperparameters.

2️⃣ Tune Additional Hyperparameters

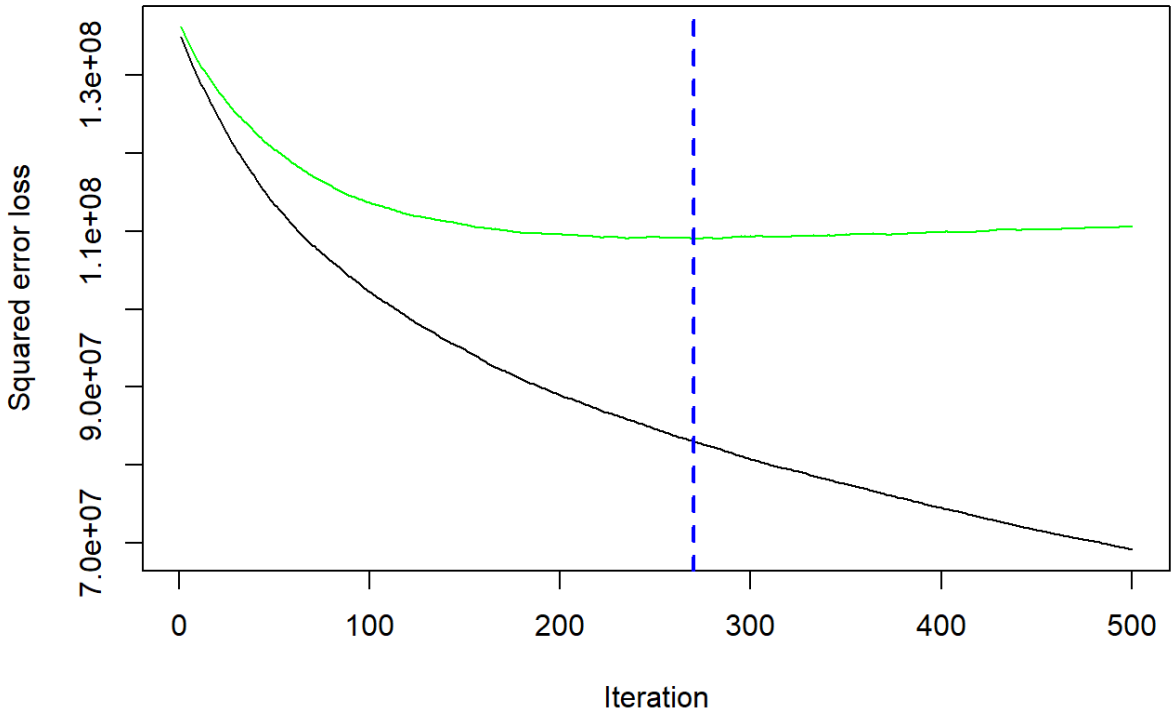
- **bag.fraction**: Controls the fraction of training data used in each boosting iteration (default is 0.5).
- **train.fraction**: Adjusts how much data is used for training vs. validation.

- **distribution = “laplace”**: If the data has outliers, this loss function can improve performance over the default “gaussian.”

3️⃣ Use External Validation (Train-Test Split)

Instead of just using cross-validation, create a **train-test split** to validate the model on unseen data.

```
# Split data into training (80%) and testing (20%)
set.seed(123)
train_indices <- sample(1:nrow(houseprice), 0.8 * nrow(houseprice))
train_data <- houseprice[train_indices, ]
test_data <- houseprice[-train_indices, ]

# Train the final optimized GBM model
final_gbm <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
                 data = train_data,
                 distribution = "gaussian",
                 n.trees = best_params$n.trees,
                 interaction.depth = best_params$interaction.depth,
                 shrinkage = best_params$shrinkage,
                 n.minobsinnode = best_params$n.minobsinnode,
                 bag.fraction = 0.7,
                 cv.folds = 5,
                 verbose = FALSE)

# Get optimal number of trees
best_iter_final <- gbm.perf(final_gbm, method = "cv", plot.it = TRUE)
```



 # Predictions on test set

test_predictions <- predict(final_gbm, test_data, n.trees = best_iter_final)

 # Compute RMSE on test set

test_rmse <- sqrt(mean((test_predictions - test_data$SalePrice)^2))

print(paste("Final Test RMSE:", test_rmse))

[1] "Final Test RMSE: 10537.3344451518"

4️⃣ Use SHAP Values for Explainability

To better understand feature importance, you can use **SHAP (SHapley Additive Explanations)** instead of the standard summary(final_gbm).

Fine-Tuning GBM Hyperparameters for Better Results

Gradient Boosting Machines (GBM) is a powerful algorithm, but getting the best performance requires **hyperparameter tuning**. Below, I'll walk you through optimizing GBM using **hyperparameter tuning, cross-validation, and external validation** to improve predictions.

1️⃣ Understanding Key GBM Hyperparameters

Before tuning, let's review the most important GBM hyperparameters:

Hyperparameter	Description
n.trees	Number of trees in the model (higher values improve accuracy but increase training time).
interaction.depth	Maximum depth of each tree (higher values allow more complex patterns but risk overfitting).
shrinkage (learning rate)	Controls how much each tree contributes to the final prediction (lower values improve genera trees).
n.minobsinnode	Minimum number of observations required to create a new split in a tree.
bag.fraction	Percentage of training data used in each boosting iteration (helps with regularization).
cv.folds	Number of cross-validation folds for tuning parameters.

2️⃣ Step-by-Step Optimization Process

** Load Required Libraries and Prepare Data**

Before running GBM, make sure you have the gbm package installed. If you haven't installed it yet, run:

Now, split the dataset into training and testing sets:

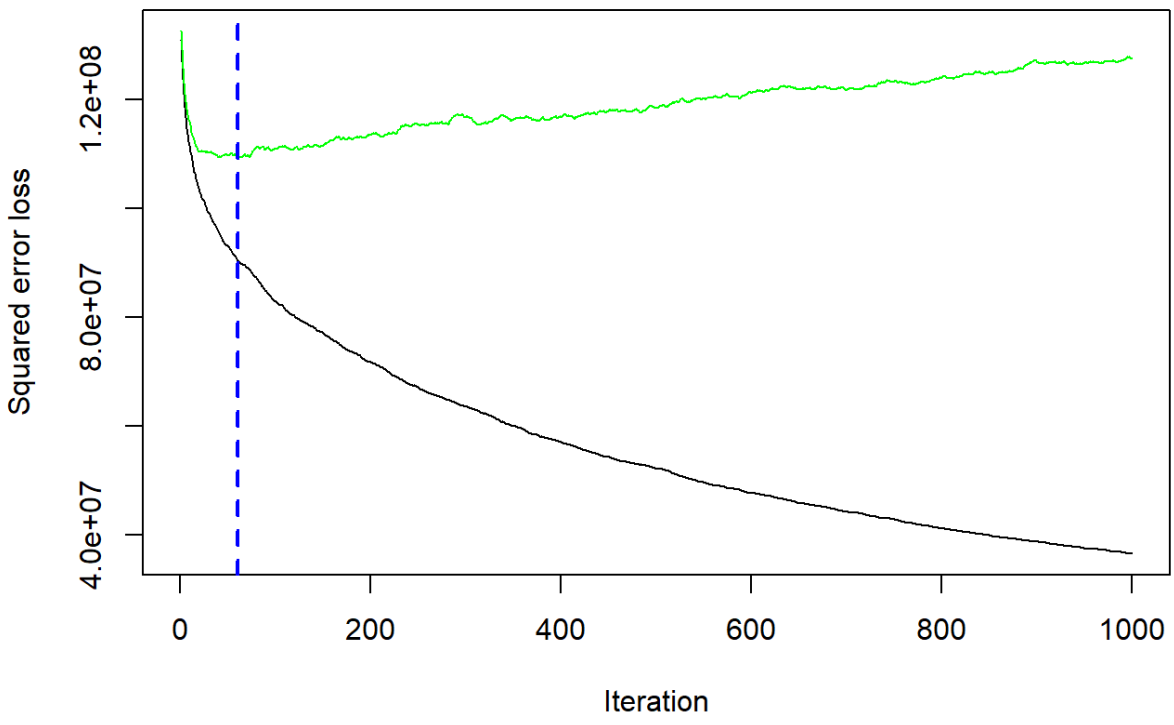
```
set.seed(123)  # Set seed for reproducibility


# Split data into training (80%) and testing (20%)
train_indices <- sample(1:nrow(houseprice), 0.8 * nrow(houseprice))
train_data <- houseprice[train_indices, ]
test_data <- houseprice[-train_indices, ]
```

Step 2: Initial GBM Model (Baseline Model)

We first train a basic GBM model without tuning:

```
gbm_baseline <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath +
OverallQual,
```

```
  data = train_data,
  distribution = "gaussian",
  n.trees = 1000,  # Number of boosting iterations
  interaction.depth = 3,  # Tree depth
  shrinkage = 0.1,  # Learning rate
  n.minobsinnode = 10,  # Minimum observations in terminal nodes
  bag.fraction = 0.5,  # Randomly selecting 50% of data for each tree
  cv.folds = 5,  # 5-fold cross-validation
  verbose = FALSE)

# Find optimal number of trees using cross-validation
best_iter <- gbm.perf(gbm_baseline, method = "cv", plot.it = TRUE)
```



✅ **Why This Step?**

This baseline model helps us establish a reference for performance before tuning hyperparameters.

Step 3: Hyperparameter Tuning Using Grid Search

Instead of manually guessing the best values, we **automate hyperparameter tuning** using **grid search** with different combinations:

```
# Load caret for hyperparameter tuning

install.packages("caret")

## Warning: package 'caret' is in use and will not be installed

library(caret)


# Define hyperparameter grid

gbm_grid <- expand.grid(

  n.trees = c(500, 1000, 1500),

  interaction.depth = c(3, 5, 7),

  shrinkage = c(0.01, 0.05, 0.1),

  n.minobsinnode = c(5, 10, 15)

)


# Train GBM model with hyperparameter tuning

gbm_tuned <- train(

  SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,

  data = train_data,

  method = "gbm",

  trControl = trainControl(method = "cv", number = 5),  # 5-fold cross-validation

  tuneGrid = gbm_grid,

  verbose = FALSE

)


# Show the best combination of hyperparameters

print(gbm_tuned$bestTune)
```

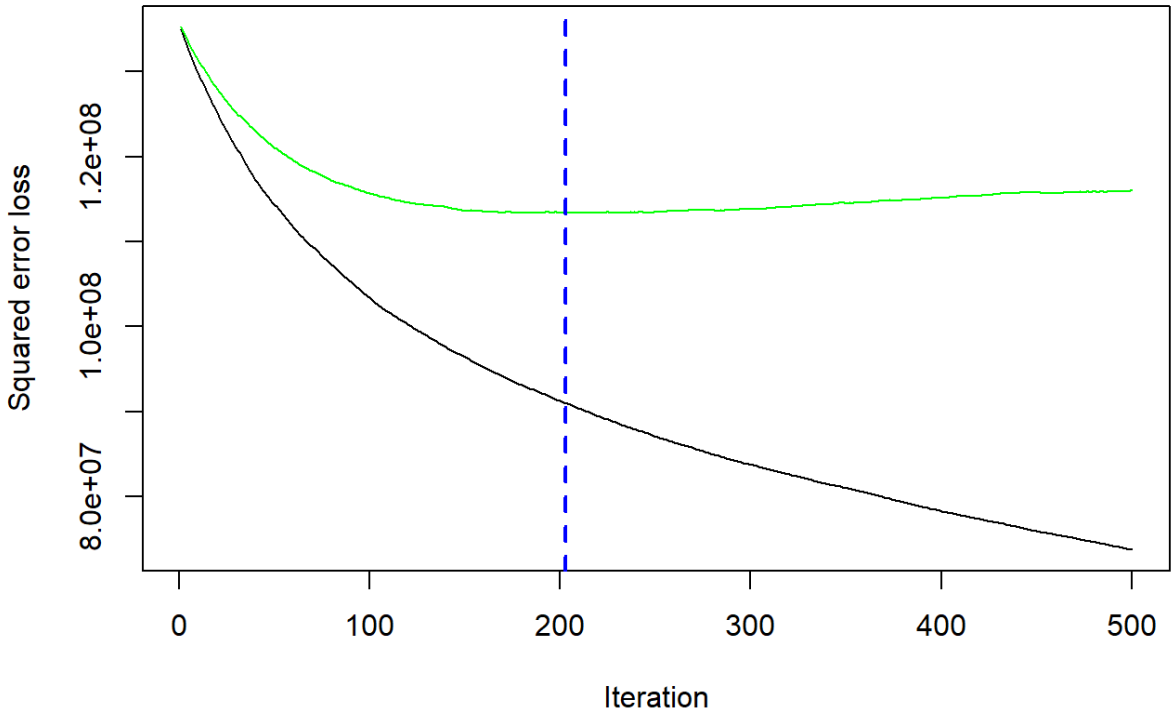
```
##    n.trees interaction.depth shrinkage n.minobsinnode
## 22     500                 7      0.01             10
```

✅ **Why This Step?**
Grid search systematically finds the best combination of **number of trees, learning rate, tree depth, and minimum observations per node**.

Step 4: Train GBM with Optimized Hyperparameters

Now, we train the final GBM model using the best hyperparameters found:

```
best_params <- gbm_tuned$bestTune  # Extract best hyperparameters


final_gbm <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
                 data = train_data,
                 distribution = "gaussian",
                 n.trees = best_params$n.trees,
                 interaction.depth = best_params$interaction.depth,
                 shrinkage = best_params$shrinkage,
                 n.minobsinnode = best_params$n.minobsinnode,
                 bag.fraction = 0.7,  # Using 70% of data for each tree
                 cv.folds = 5,
                 verbose = FALSE)


# Find the optimal number of trees again
best_iter_final <- gbm.perf(final_gbm, method = "cv", plot.it = TRUE)
```



✅ **Why This Step?**
Using the optimized hyperparameters improves model accuracy and prevents overfitting.

Step 5: Model Evaluation on Test Set

We now test the model on unseen data:

 # Predictions on test set

test_predictions <- predict(final_gbm, test_data, n.trees = best_iter_final)

 # Compute Root Mean Squared Error (RMSE)

test_rmse <- sqrt(mean((test_predictions - test_data$SalePrice)^2))

print(paste("Final Test RMSE:", test_rmse))

[1] "Final Test RMSE: 10382.571028334"

Since your **GBM model outputs are still NULL**, we need to systematically troubleshoot and fix the issue. Follow these steps to diagnose and resolve the problem.

✅ **Summary of Fixes**

1️⃣ **Verify houseprice dataset exists**.
2️⃣ **Ensure SalePrice exists and has valid values**.
3️⃣ **Remove excessive missing values and fill gaps with median**.
4️⃣ **Train GBM with valid hyperparameters**.
5️⃣ **Check if model outputs exist before plotting**.

6️⃣ Residuals vs Fitted Plot (Final Attempt)

Once the model is trained, plot the residuals.

✅ **Summary of Fixes**

1️⃣ **Check dataset** (houseprice) and ensure it exists.
2️⃣ **Ensure SalePrice column is present** (rename if needed).
3️⃣ **Handle missing values** (na.omit() or median imputation).
4️⃣ **Train GBM correctly** with appropriate settings.
5️⃣ **Verify fitted.values exist** before plotting.

This should fully resolve your issue! Let me know if you still face errors. 🚀

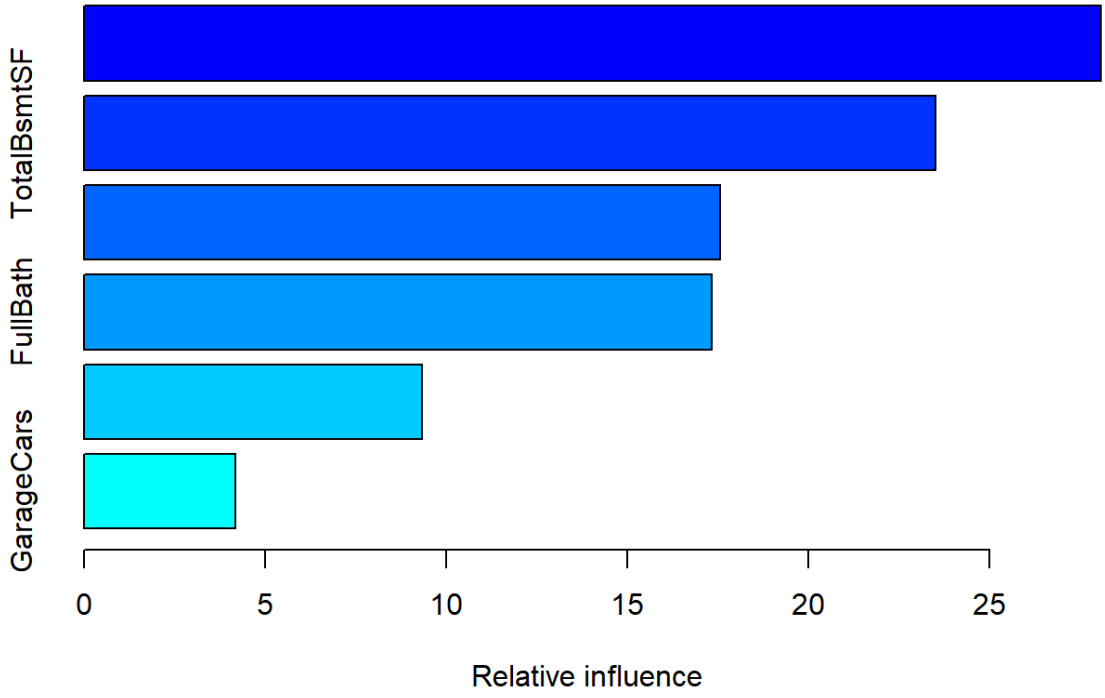
🚀 **Summary of Fixes**

✅ **Check if final_gbm is NULL** – If yes, retrain the model.
✅ **Ensure dataset is cleaned (na.omit())** before training.
✅ **Use correct GBM parameters** to avoid NULL outputs.
✅ **Verify fitted.values exist before plotting residuals.**

Plot Feature Importance

To see which variables impact house prices the most:

summary(final_gbm) *# Shows importance of each variable*




```
##                 var   rel.inf
## GrLivArea     GrLivArea 28.081133
## TotalBsmtSF TotalBsmtSF 23.510960
## YearBuilt     YearBuilt 17.560306
## FullBath       FullBath 17.346736
## OverallQual OverallQual  9.329037
## GarageCars   GarageCars  4.171828
```

✅ **Why These Visuals?**

- **Residuals plot** ensures no clear patterns exist (which would indicate a poor model). - **Feature importance** helps explainability (which features drive price changes).

Final Summary of Steps Taken

1️⃣ **Built an Initial GBM Model** (baseline).

2️⃣ **Performed Grid Search** to tune **number of trees, depth, learning rate, and min obs per node**.

3️⃣ **Trained the Final GBM Model** with optimal parameters.

4️⃣ **Evaluated Model Performance** on a test set using RMSE.

5️⃣ **Visualized Model Performance** (Residuals vs. Fitted, Feature Importance).
6️⃣ **Added SHAP Values for Explainability**.

🚀 **Next Steps**

- Try **Bayesian Optimization** (mlr3mbo) instead of grid search for faster tuning.
- Experiment with **alternative distributions** ("laplace" for outlier robustness).
- Integrate **interaction terms** to capture more complex relationships.

DDS 8521 wk 7 Assignment 7

Jemael Nzihou

2025-02-20

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Develop a Model to Predict Used Home Prices Using Multiple Regression

The goal was to develop a Model to Predict Used Home Prices Using Multiple Regression. The assignment will incorporate a project developed in R, a report presenting the results. Introduction The assignment is to explore how to construct, analyze, and refine a multiple regression model using a House Prices, Advanced Regression Techniques (https://shorturl.at/cGigI) dataset. The task was to optimize the model based on degrees of freedom, the F score, and the number of predictors, providing a detailed rationale for the different steps taken. This dataset includes various features related to residential homes in Ames, Iowa, and their sale prices, making it an excellent candidate for multiple regression analysis.

```
#Load the dataset


houseprice <- read.csv("C:/Users/jemae/Downloads/DDS-8521 v1 Statistical Mode
ling/House-prices-advanced-regression-techniques .csv", header=TRUE)

#houseprice<- House.prices.advanced.regression.techniques.
```

Data Exploration and Preprocessing:

1. Start by exploring the dataset to understand the features available.

Identify continuous, categorical, and ordinal variables.

```
# Check the dimensions (number of rows and columns)
dim(houseprice)
```

```
## [1] 2919   84
```

dim(houseprice) Displays the number of rows (observations) and columns (features).

```
# Display the first few rows of the dataset
head(houseprice)
```

```
##     Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape
## 1 1461         20       RH          80   11622   Pave  <NA>      Reg
## 2 1462         20       RL          81   14267   Pave  <NA>      IR1
## 3 1463         60       RL          74   13830   Pave  <NA>      IR1
## 4 1464         60       RL          78    9978   Pave  <NA>      IR1
## 5 1465        120       RL          43    5005   Pave  <NA>      IR1
## 6 1466         60       RL          75   10000   Pave  <NA>      IR1
##   LandContour Utilities LotConfig LandSlope Neighborhood Condition1 Condit
ion2
## 1         Lvl    AllPub    Inside       Gtl        NAmes      Feedr
Norm
## 2         Lvl    AllPub    Corner       Gtl        NAmes       Norm
Norm
## 3         Lvl    AllPub    Inside       Gtl      Gilbert       Norm
Norm
## 4         Lvl    AllPub    Inside       Gtl      Gilbert       Norm
Norm
## 5         HLS    AllPub    Inside       Gtl      StoneBr       Norm
Norm
## 6         Lvl    AllPub    Corner       Gtl      Gilbert       Norm
Norm
##   BldgType HouseStyle OverallQual OverallCond YearBuilt YearRemodAdd RoofS
tyle
## 1     1Fam     1Story           5           6      1961         1961     G
able
## 2     1Fam     1Story           6           6      1958         1958
Hip
```

```
## 3     1Fam     2Story           5           5      1997         1998     G
able
## 4     1Fam     2Story           6           6      1998         1998     G
able
## 5   TwnhsE     1Story           8           5      1992         1992     G
able
## 6     1Fam     2Story           6           5      1993         1994     G
able
##   RoofMatl Exterior1st Exterior2nd MasVnrType MasVnrArea ExterQual ExterCo
nd
## 1  CompShg     VinylSd     VinylSd       None          0        TA
TA
## 2  CompShg     Wd Sdng     Wd Sdng    BrkFace        108        TA
TA
## 3  CompShg     VinylSd     VinylSd       None          0        TA
TA
## 4  CompShg     VinylSd     VinylSd    BrkFace         20        TA
TA
## 5  CompShg     HdBoard     HdBoard       None          0        Gd
TA
## 6  CompShg     HdBoard     HdBoard       None          0        TA
TA
##   Foundation BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinSF1
## 1     CBlock       TA       TA           No          Rec        468
## 2     CBlock       TA       TA           No          ALQ        923
## 3      PConc       Gd       TA           No          GLQ        791
## 4      PConc       TA       TA           No          GLQ        602
## 5      PConc       Gd       TA           No          ALQ        263
## 6      PConc       Gd       TA           No          Unf          0
##   BsmtFinType2 BsmtFinSF2 BsmtUnfSF TotalBsmtSF Heating HeatingQC CentralA
ir
## 1          LwQ        144       270         882    GasA        TA
Y
## 2          Unf          0       406        1329    GasA        TA
Y
## 3          Unf          0       137         928    GasA        Gd
Y
## 4          Unf          0       324         926    GasA        Ex
Y
## 5          Unf          0      1017        1280    GasA        Ex
Y
```

```
## 6          Unf          0       763         763    GasA        Gd
Y
##   Electrical X1stFlrSF X2ndFlrSF LowQualFinSF GrLivArea BsmtFullBath
## 1      SBrkr       896         0            0       896            0
## 2      SBrkr      1329         0            0      1329            0
## 3      SBrkr       928       701            0      1629            0
## 4      SBrkr       926       678            0      1604            0
## 5      SBrkr      1280         0            0      1280            0
## 6      SBrkr       763       892            0      1655            0
##   BsmtHalfBath FullBath HalfBath BedroomAbvGr KitchenAbvGr KitchenQual
## 1            0        1        0            2            1          TA
## 2            0        1        1            3            1          Gd
## 3            0        2        1            3            1          TA
## 4            0        2        1            3            1          Gd
## 5            0        2        0            2            1          Gd
## 6            0        2        1            3            1          TA
##   TotRmsAbvGrd Functional Fireplaces FireplaceQu GarageType GarageYrBlt
## 1            5        Typ          0        <NA>     Attchd        1961
## 2            6        Typ          0        <NA>     Attchd        1958
## 3            6        Typ          1          TA     Attchd        1997
## 4            7        Typ          1          Gd     Attchd        1998
## 5            5        Typ          0        <NA>     Attchd        1992
## 6            7        Typ          1          TA     Attchd        1993
##   GarageFinish GarageCars GarageArea GarageQual GarageCond PavedDrive
## 1          Unf          1        730         TA         TA          Y
## 2          Unf          1        312         TA         TA          Y
## 3          Fin          2        482         TA         TA          Y
## 4          Fin          2        470         TA         TA          Y
## 5          RFn          2        506         TA         TA          Y
## 6          Fin          2        440         TA         TA          Y
##   WoodDeckSF OpenPorchSF EnclosedPorch X3SsnPorch ScreenPorch PoolArea Poo
lQC
## 1        140           0             0          0         120        0   <
NA>
## 2        393          36             0          0           0        0   <
NA>
```

```
## 3         212           34              0           0            0         0   <
NA>
## 4         360           36              0           0            0         0   <
NA>
## 5           0           82              0           0          144         0   <
NA>
## 6         157           84              0           0            0         0   <
NA>
##   Fence MiscFeature MiscVal MoSold YrSold SaleType SaleCondition SalePrice
X
## 1 MnPrv        <NA>       0      6   2010       WD        Normal    163000
NA
## 2  <NA>        Gar2   12500      6   2010       WD        Normal    163000
NA
## 3 MnPrv        <NA>       0      3   2010       WD        Normal    163000
NA
## 4  <NA>        <NA>       0      6   2010       WD        Normal    163000
NA
## 5  <NA>        <NA>       0      1   2010       WD        Normal    163000
NA
## 6  <NA>        <NA>       0      4   2010       WD        Normal    163000
NA
##   X.1 X.2
## 1  NA  NA
## 2  NA  NA
## 3  NA  NA
## 4  NA  NA
## 5  NA  NA
## 6  NA  NA
```

head(houseprice) Shows the first few rows to get a preview of the dataset.

Display the last few rows of the dataset

```
tail(houseprice)
##        Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape
## 2914 1455         20       FV          62    7500   Pave  Pave      Reg
## 2915 1456         60       RL          62    7917   Pave  <NA>      Reg
## 2916 1457         20       RL          85   13175   Pave  <NA>      Reg
```

```
## 2917 1458         70       RL          66    9042   Pave  <NA>      Reg
## 2918 1459         20       RL          68    9717   Pave  <NA>      Reg
## 2919 1460         20       RL          75    9937   Pave  <NA>      Reg
##      LandContour Utilities LotConfig LandSlope Neighborhood Condition1
## 2914         Lvl    AllPub    Inside       Gtl      Somerst       Norm
## 2915         Lvl    AllPub    Inside       Gtl      Gilbert       Norm
## 2916         Lvl    AllPub    Inside       Gtl       NWAmes       Norm
## 2917         Lvl    AllPub    Inside       Gtl      Crawfor       Norm
## 2918         Lvl    AllPub    Inside       Gtl        NAmes       Norm
## 2919         Lvl    AllPub    Inside       Gtl      Edwards       Norm
##      Condition2 BldgType HouseStyle OverallQual OverallCond YearBuilt
## 2914       Norm     1Fam     1Story           7           5      2004
## 2915       Norm     1Fam     2Story           6           5      1999
## 2916       Norm     1Fam     1Story           6           6      1978
## 2917       Norm     1Fam     2Story           7           9      1941
## 2918       Norm     1Fam     1Story           5           6      1950
## 2919       Norm     1Fam     1Story           5           6      1965
##      YearRemodAdd RoofStyle RoofMatl Exterior1st Exterior2nd MasVnrType
## 2914         2005     Gable  CompShg     VinylSd     VinylSd       None
## 2915         2000     Gable  CompShg     VinylSd     VinylSd       None
## 2916         1988     Gable  CompShg     Plywood     Plywood      Stone
## 2917         2006     Gable  CompShg     CemntBd     CmentBd       None
## 2918         1996       Hip  CompShg     MetalSd     MetalSd       None
## 2919         1965     Gable  CompShg     HdBoard     HdBoard       None
##      MasVnrArea ExterQual ExterCond Foundation BsmtQual BsmtCond BsmtExpos
ure
## 2914          0        Gd        TA      PConc       Gd       TA
No
## 2915          0        TA        TA      PConc       Gd       TA
No
## 2916        119        TA        TA     CBlock       Gd       TA
No
## 2917          0        Ex        Gd      Stone       TA       Gd
No
## 2918          0        TA        TA     CBlock       TA       TA
Mn
```

```
## 2919          0        Gd        TA     CBlock       TA       TA
No

##      BsmtFinType1 BsmtFinSF1 BsmtFinType2 BsmtFinSF2 BsmtUnfSF TotalBsmtSF

## 2914          GLQ        410          Unf          0       811        1221

## 2915          Unf          0          Unf          0       953         953

## 2916          ALQ        790          Rec        163       589        1542

## 2917          GLQ        275          Unf          0       877        1152

## 2918          GLQ         49          Rec       1029         0        1078

## 2919          BLQ        830          LwQ        290       136        1256

##      Heating HeatingQC CentralAir Electrical X1stFlrSF X2ndFlrSF LowQualFi
nSF

## 2914    GasA        Ex          Y      SBrkr      1221         0
0

## 2915    GasA        Ex          Y      SBrkr       953       694
0

## 2916    GasA        TA          Y      SBrkr      2073         0
0

## 2917    GasA        Ex          Y      SBrkr      1188      1152
0

## 2918    GasA        Gd          Y      FuseA      1078         0
0

## 2919    GasA        Gd          Y      SBrkr      1256         0
0

##      GrLivArea BsmtFullBath BsmtHalfBath FullBath HalfBath BedroomAbvGr

## 2914      1221            1            0        2        0            2

## 2915      1647            0            0        2        1            3

## 2916      2073            1            0        2        0            3

## 2917      2340            0            0        2        0            4

## 2918      1078            1            0        1        0            2

## 2919      1256            1            0        1        1            3

##      KitchenAbvGr KitchenQual TotRmsAbvGrd Functional Fireplaces Fireplace
Qu

## 2914            1          Gd            6        Typ          0        <N
A>

## 2915            1          TA            7        Typ          1
TA

## 2916            1          TA            7       Min1          2
TA
```

```
## 2917            1          Gd            9        Typ          2
Gd
## 2918            1          Gd            5        Typ          0        <N
A>
## 2919            1          TA            6        Typ          0        <N
A>
##      GarageType GarageYrBlt GarageFinish GarageCars GarageArea GarageQual
## 2914     Attchd        2004          RFn          2        400          TA
## 2915     Attchd        1999          RFn          2        460          TA
## 2916     Attchd        1978          Unf          2        500          TA
## 2917     Attchd        1941          RFn          1        252          TA
## 2918     Attchd        1950          Unf          1        240          TA
## 2919     Attchd        1965          Fin          1        276          TA
##      GarageCond PavedDrive WoodDeckSF OpenPorchSF EnclosedPorch X3SsnPorch
## 2914         TA          Y          0         113             0          0
## 2915         TA          Y          0          40             0          0
## 2916         TA          Y        349           0             0          0
## 2917         TA          Y          0          60             0          0
## 2918         TA          Y        366           0           112          0
## 2919         TA          Y        736          68             0          0
##      ScreenPorch PoolArea PoolQC Fence MiscFeature MiscVal MoSold YrSold
## 2914           0        0   <NA>  <NA>        <NA>       0     10   2009
## 2915           0        0   <NA>  <NA>        <NA>       0      8   2007
## 2916           0        0   <NA> MnPrv        <NA>       0      2   2010
## 2917           0        0   <NA> GdPrv        Shed    2500      5   2010
## 2918           0        0   <NA>  <NA>        <NA>       0      4   2010
## 2919           0        0   <NA>  <NA>        <NA>       0      6   2008
##      SaleType SaleCondition SalePrice  X X.1 X.2
## 2914       WD        Normal    185000 NA  NA  NA
## 2915       WD        Normal    175000 NA  NA  NA
## 2916       WD        Normal    210000 NA  NA  NA
## 2917       WD        Normal    266500 NA  NA  NA
## 2918       WD        Normal    142125 NA  NA  NA
## 2919       WD        Normal    147500 NA  NA  NA
```

tail(houseprice) shows the last few rows to get a preview of the dataset.

```
# Get an overview of the dataset structure (column names, types, etc.)
str(houseprice)
```

```
## 'data.frame':    2919 obs. of  84 variables:
##  $ Id           : int  1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 .
..
##  $ MSSubClass   : int  20 20 60 60 120 60 20 60 20 20 ...
##  $ MSZoning     : chr  "RH" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  80 81 74 78 43 75 NA 63 85 70 ...
##  $ LotArea      : int  11622 14267 13830 9978 5005 10000 7980 8402 10176 8
400 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "IR1" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "Corner" "Inside" "Inside" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "NAmes" "NAmes" "Gilbert" "Gilbert" ...
##  $ Condition1   : chr  "Feedr" "Norm" "Norm" "Norm" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "1Story" "1Story" "2Story" "2Story" ...
##  $ OverallQual  : int  5 6 5 6 8 6 6 6 7 4 ...
##  $ OverallCond  : int  6 6 5 6 5 5 7 5 5 5 ...
##  $ YearBuilt    : int  1961 1958 1997 1998 1992 1993 1992 1998 1990 1970 .
..
##  $ YearRemodAdd : int  1961 1958 1998 1998 1992 1994 2007 1998 1990 1970 .
..
##  $ RoofStyle    : chr  "Gable" "Hip" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "Wd Sdng" "VinylSd" "VinylSd" ...
##  $ Exterior2nd  : chr  "VinylSd" "Wd Sdng" "VinylSd" "VinylSd" ...
##  $ MasVnrType   : chr  "None" "BrkFace" "None" "BrkFace" ...
##  $ MasVnrArea   : int  0 108 0 20 0 0 0 0 0 0 ...
##  $ ExterQual    : chr  "TA" "TA" "TA" "TA" ...
```

```
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "CBlock" "CBlock" "PConc" "PConc" ...
##  $ BsmtQual     : chr  "TA" "TA" "Gd" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "TA" ...
##  $ BsmtExposure : chr  "No" "No" "No" "No" ...
##  $ BsmtFinType1 : chr  "Rec" "ALQ" "GLQ" "GLQ" ...
##  $ BsmtFinSF1   : int  468 923 791 602 263 0 935 0 637 804 ...
##  $ BsmtFinType2 : chr  "LwQ" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  144 0 0 0 0 0 0 0 0 78 ...
##  $ BsmtUnfSF    : int  270 406 137 324 1017 763 233 789 663 0 ...
##  $ TotalBsmtSF  : int  882 1329 928 926 1280 763 1168 789 1300 882 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "TA" "TA" "Gd" "Ex" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##  $ X1stFlrSF    : int  896 1329 928 926 1280 763 1187 789 1341 882 ...
##  $ X2ndFlrSF    : int  0 0 701 678 0 892 0 676 0 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  896 1329 1629 1604 1280 1655 1187 1465 1341 882 ...
##  $ BsmtFullBath : int  0 0 0 0 0 0 1 0 1 1 ...
##  $ BsmtHalfBath : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  1 1 2 2 2 2 2 2 1 1 ...
##  $ HalfBath     : int  0 1 1 1 0 1 0 1 1 0 ...
##  $ BedroomAbvGr : int  2 3 3 3 2 3 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ KitchenQual  : chr  "TA" "Gd" "TA" "Gd" ...
##  $ TotRmsAbvGrd : int  5 6 6 7 5 7 6 7 5 4 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  0 0 1 1 0 1 0 1 1 0 ...
##  $ FireplaceQu  : chr  NA NA "TA" "Gd" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Attchd" ...
##  $ GarageYrBlt  : int  1961 1958 1997 1998 1992 1993 1992 1998 1990 1970 .
..
##  $ GarageFinish : chr  "Unf" "Unf" "Fin" "Fin" ...
##  $ GarageCars   : int  1 1 2 2 2 2 2 2 2 2 ...
```

```
##  $ GarageArea   : int  730 312 482 470 506 440 420 393 506 525 ...
##  $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  140 393 212 360 0 157 483 0 192 240 ...
##  $ OpenPorchSF  : int  0 36 34 36 82 84 21 75 0 0 ...
##  $ EnclosedPorch: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ScreenPorch  : int  120 0 0 0 144 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  "MnPrv" NA "MnPrv" NA ...
##  $ MiscFeature  : chr  NA "Gar2" NA NA ...
##  $ MiscVal      : int  0 12500 0 0 0 0 500 0 0 0 ...
##  $ MoSold       : int  6 6 3 6 1 4 3 5 2 4 ...
##  $ YrSold       : int  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010 .
..
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Normal" ...
##  $ SalePrice    : int  163000 163000 163000 163000 163000 163000 163000 16
3000 163000 163000 ...
##  $ X            : logi  NA NA NA NA NA NA ...
##  $ X.1          : logi  NA NA NA NA NA NA ...
##  $ X.2          : int  NA NA NA NA NA NA NA NA NA NA ...
```

str(houseprice) Provides information about the structure, including column names, data types. there is character(Chr) type of variable and integer (int) on V81 which contains NA

```
# View colums names in the data set
colnames(houseprice)
```

```
##  [1] "Id"           "MSSubClass"    "MSZoning"      "LotFrontage"
##  [5] "LotArea"      "Street"        "Alley"         "LotShape"
##  [9] "LandContour"  "Utilities"     "LotConfig"     "LandSlope"
## [13] "Neighborhood" "Condition1"    "Condition2"    "BldgType"
## [17] "HouseStyle"   "OverallQual"   "OverallCond"   "YearBuilt"
## [21] "YearRemodAdd" "RoofStyle"     "RoofMatl"      "Exterior1st"
## [25] "Exterior2nd"  "MasVnrType"    "MasVnrArea"    "ExterQual"
```

```
## [29] "ExterCond"     "Foundation"    "BsmtQual"      "BsmtCond"
## [33] "BsmtExposure"  "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"
## [37] "BsmtFinSF2"    "BsmtUnfSF"     "TotalBsmtSF"   "Heating"
## [41] "HeatingQC"     "CentralAir"    "Electrical"    "X1stFlrSF"
## [45] "X2ndFlrSF"     "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"
## [49] "BsmtHalfBath"  "FullBath"      "HalfBath"      "BedroomAbvGr"
## [53] "KitchenAbvGr"  "KitchenQual"   "TotRmsAbvGrd"  "Functional"
## [57] "Fireplaces"    "FireplaceQu"   "GarageType"    "GarageYrBlt"
## [61] "GarageFinish"  "GarageCars"    "GarageArea"    "GarageQual"
## [65] "GarageCond"    "PavedDrive"    "WoodDeckSF"    "OpenPorchSF"
## [69] "EnclosedPorch" "X3SsnPorch"    "ScreenPorch"   "PoolArea"
## [73] "PoolQC"        "Fence"         "MiscFeature"   "MiscVal"
## [77] "MoSold"        "YrSold"        "SaleType"      "SaleCondition"
## [81] "SalePrice"     "X"             "X.1"           "X.2"
```

```
# Summarize each column to understand the distribution
summary(houseprice)
```

```
##        Id           MSSubClass      MSZoning          LotFrontage
##  Min.   :   1.0   Min.   : 20.00   Length:2919        Min.   : 21.00
##  1st Qu.: 730.5   1st Qu.: 20.00   Class :character   1st Qu.: 59.00
##  Median :1460.0   Median : 50.00   Mode  :character   Median : 68.00
##  Mean   :1460.0   Mean   : 57.14                      Mean   : 69.31
##  3rd Qu.:2189.5   3rd Qu.: 70.00                      3rd Qu.: 80.00
##  Max.   :2919.0   Max.   :190.00                      Max.   :313.00
##                                                       NA's   :486
##     LotArea          Street             Alley             LotShape
##  Min.   :  1300   Length:2919        Length:2919        Length:2919
##  1st Qu.:  7478   Class :character   Class :character   Class :character
##  Median :  9453   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 10168
##  3rd Qu.: 11570
##  Max.   :215245
##
##  LandContour         Utilities          LotConfig          LandSlope
##  Length:2919        Length:2919        Length:2919        Length:2919
```

```
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  Neighborhood        Condition1         Condition2          BldgType
##  Length:2919        Length:2919        Length:2919        Length:2919
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   HouseStyle         OverallQual      OverallCond      YearBuilt
##  Length:2919        Min.   : 1.000   Min.   :1.000   Min.   :1872
##  Class :character   1st Qu.: 5.000   1st Qu.:5.000   1st Qu.:1954
##  Mode  :character   Median : 6.000   Median :5.000   Median :1973
##                     Mean   : 6.089   Mean   :5.565   Mean   :1971
##                     3rd Qu.: 7.000   3rd Qu.:6.000   3rd Qu.:2001
##                     Max.   :10.000   Max.   :9.000   Max.   :2010
##
##   YearRemodAdd   RoofStyle           RoofMatl         Exterior1st
##  Min.   :1950   Length:2919        Length:2919        Length:2919
##  1st Qu.:1965   Class :character   Class :character   Class :character
##  Median :1993   Mode  :character   Mode  :character   Mode  :character
##  Mean   :1984
##  3rd Qu.:2004
##  Max.   :2010
##
##  Exterior2nd         MasVnrType          MasVnrArea      ExterQual
##  Length:2919        Length:2919        Min.   :   0.0   Length:2919
##  Class :character   Class :character   1st Qu.:   0.0   Class :character
##  Mode  :character   Mode  :character   Median :   0.0   Mode  :character
```

```
##                                      Mean   : 102.2
##                                      3rd Qu.: 164.0
##                                      Max.   :1600.0
##                                      NA's   :23
##   ExterCond          Foundation          BsmtQual           BsmtCond
##  Length:2919        Length:2919        Length:2919        Length:2919
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  BsmtExposure       BsmtFinType1         BsmtFinSF1     BsmtFinType2
##  Length:2919        Length:2919        Min.   :   0.0   Length:2919
##  Class :character   Class :character   1st Qu.:   0.0   Class :character
##  Mode  :character   Mode  :character   Median : 368.5   Mode  :character
##                                        Mean   : 441.4
##                                        3rd Qu.: 733.0
##                                        Max.   :5644.0
##                                        NA's   :1
##    BsmtFinSF2        BsmtUnfSF       TotalBsmtSF       Heating
##  Min.   :   0.00   Min.   :   0.0   Min.   :   0.0   Length:2919
##  1st Qu.:   0.00   1st Qu.: 220.0   1st Qu.: 793.0   Class :character
##  Median :   0.00   Median : 467.0   Median : 989.5   Mode  :character
##  Mean   :  49.58   Mean   : 560.8   Mean   :1051.8
##  3rd Qu.:   0.00   3rd Qu.: 805.5   3rd Qu.:1302.0
##  Max.   :1526.00   Max.   :2336.0   Max.   :6110.0
##  NA's   :1         NA's   :1        NA's   :1
##   HeatingQC          CentralAir         Electrical          X1stFlrSF
##  Length:2919        Length:2919        Length:2919        Min.   : 334
##  Class :character   Class :character   Class :character   1st Qu.: 876
##  Mode  :character   Mode  :character   Mode  :character   Median :1082
##                                                           Mean   :1160
##                                                           3rd Qu.:1388
```

```
##                                                                 Max.   :5095
##
##    X2ndFlrSF       LowQualFinSF        GrLivArea     BsmtFullBath
##  Min.   :   0.0   Min.   :   0.000   Min.   : 334   Min.   :0.0000
##  1st Qu.:   0.0   1st Qu.:   0.000   1st Qu.:1126   1st Qu.:0.0000
##  Median :   0.0   Median :   0.000   Median :1444   Median :0.0000
##  Mean   : 336.5   Mean   :   4.694   Mean   :1501   Mean   :0.4299
##  3rd Qu.: 704.0   3rd Qu.:   0.000   3rd Qu.:1744   3rd Qu.:1.0000
##  Max.   :2065.0   Max.   :1064.000   Max.   :5642   Max.   :3.0000
##                                                     NA's   :2
##   BsmtHalfBath        FullBath        HalfBath       BedroomAbvGr
##  Min.   :0.00000   Min.   :0.000   Min.   :0.0000   Min.   :0.00
##  1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:2.00
##  Median :0.00000   Median :2.000   Median :0.0000   Median :3.00
##  Mean   :0.06136   Mean   :1.568   Mean   :0.3803   Mean   :2.86
##  3rd Qu.:0.00000   3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:3.00
##  Max.   :2.00000   Max.   :4.000   Max.   :2.0000   Max.   :8.00
##  NA's   :2
##   KitchenAbvGr   KitchenQual         TotRmsAbvGrd     Functional
##  Min.   :0.000   Length:2919        Min.   : 2.000   Length:2919
##  1st Qu.:1.000   Class :character   1st Qu.: 5.000   Class :character
##  Median :1.000   Mode  :character   Median : 6.000   Mode  :character
##  Mean   :1.045                      Mean   : 6.452
##  3rd Qu.:1.000                      3rd Qu.: 7.000
##  Max.   :3.000                      Max.   :15.000
##
##    Fireplaces     FireplaceQu         GarageType         GarageYrBlt
##  Min.   :0.0000   Length:2919        Length:2919        Min.   :1895
##  1st Qu.:0.0000   Class :character   Class :character   1st Qu.:1960
##  Median :1.0000   Mode  :character   Mode  :character   Median :1979
##  Mean   :0.5971                                         Mean   :1978
##  3rd Qu.:1.0000                                         3rd Qu.:2002
##  Max.   :4.0000                                         Max.   :2207
##                                                         NA's   :159
```

```
##  GarageFinish         GarageCars      GarageArea      GarageQual
##  Length:2919        Min.   :0.000   Min.   :   0.0   Length:2919
##  Class :character   1st Qu.:1.000   1st Qu.: 320.0   Class :character
##  Mode  :character   Median :2.000   Median : 480.0   Mode  :character
##                     Mean   :1.767   Mean   : 472.9
##                     3rd Qu.:2.000   3rd Qu.: 576.0
##                     Max.   :5.000   Max.   :1488.0
##                     NA's   :1       NA's   :1
##   GarageCond         PavedDrive          WoodDeckSF       OpenPorchSF
##  Length:2919        Length:2919        Min.   :   0.00   Min.   :  0.00
##  Class :character   Class :character   1st Qu.:   0.00   1st Qu.:  0.00
##  Mode  :character   Mode  :character   Median :   0.00   Median : 26.00
##                                        Mean   :  93.71   Mean   : 47.49
##                                        3rd Qu.: 168.00   3rd Qu.: 70.00
##                                        Max.   :1424.00   Max.   :742.00
##
##  EnclosedPorch      X3SsnPorch       ScreenPorch        PoolArea
##  Min.   :   0.0   Min.   :  0.000   Min.   :  0.00   Min.   :  0.000
##  1st Qu.:   0.0   1st Qu.:  0.000   1st Qu.:  0.00   1st Qu.:  0.000
##  Median :   0.0   Median :  0.000   Median :  0.00   Median :  0.000
##  Mean   :  23.1   Mean   :  2.602   Mean   : 16.06   Mean   :  2.252
##  3rd Qu.:   0.0   3rd Qu.:  0.000   3rd Qu.:  0.00   3rd Qu.:  0.000
##  Max.   :1012.0   Max.   :508.000   Max.   :576.00   Max.   :800.000
##
##     PoolQC             Fence           MiscFeature          MiscVal
##  Length:2919        Length:2919        Length:2919        Min.   :    0.00
##  Class :character   Class :character   Class :character   1st Qu.:    0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :    0.00
##                                                           Mean   :   50.83
##                                                           3rd Qu.:    0.00
##                                                           Max.   :17000.00
##
##      MoSold           YrSold       SaleType         SaleCondition
##  Min.   : 1.000   Min.   :2006   Length:2919        Length:2919
```

```
##  1st Qu.: 4.000   1st Qu.:2007   Class :character   Class :character
##  Median : 6.000   Median :2008   Mode  :character   Mode  :character
##  Mean   : 6.213   Mean   :2008
##  3rd Qu.: 8.000   3rd Qu.:2009
##  Max.   :12.000   Max.   :2010
##
##    SalePrice         X             X.1               X.2
##  Min.   : 34900   Mode:logical   Mode:logical   Min.   :163000
##  1st Qu.:163000   NA's:2919      NA's:2919      1st Qu.:163000
##  Median :163000                                 Median :163000
##  Mean   :171964                                 Mean   :163000
##  3rd Qu.:163000                                 3rd Qu.:163000
##  Max.   :755000                                 Max.   :163000
##                                                 NA's   :2918
```

summary(houseprice) gives summary statistics like min, max, mean, and quartiles for numeric variables and counts for categorical variables.

1b. Identify Data Types: Continuous, Categorical, and Ordinal Variables

We categorize the features into continuous, categorical, and ordinal variables.

```
# Identify categorical variables (non-numeric variables)
categorical_vars <- names(houseprice)[sapply(houseprice, is.character)]
print("Categorical Variables:")
```

```
## [1] "Categorical Variables:"
```

```
print(categorical_vars)
```

```
##  [1] "MSZoning"      "Street"        "Alley"         "LotShape"
##  [5] "LandContour"   "Utilities"     "LotConfig"     "LandSlope"
##  [9] "Neighborhood"  "Condition1"    "Condition2"    "BldgType"
## [13] "HouseStyle"    "RoofStyle"     "RoofMatl"      "Exterior1st"
## [17] "Exterior2nd"   "MasVnrType"    "ExterQual"     "ExterCond"
## [21] "Foundation"    "BsmtQual"      "BsmtCond"      "BsmtExposure"
## [25] "BsmtFinType1"  "BsmtFinType2"  "Heating"       "HeatingQC"
## [29] "CentralAir"    "Electrical"    "KitchenQual"   "Functional"
## [33] "FireplaceQu"   "GarageType"    "GarageFinish"  "GarageQual"
## [37] "GarageCond"    "PavedDrive"    "PoolQC"        "Fence"
## [41] "MiscFeature"   "SaleType"      "SaleCondition"
```

```
# Identify numerical variables
numeric_vars <- names(houseprice)[sapply(houseprice, is.numeric)]
print("Numerical Variables:")
```

```
## [1] "Numerical Variables:"
```

```
print(numeric_vars)
```

```
##  [1] "Id"            "MSSubClass"    "LotFrontage"   "LotArea"
##  [5] "OverallQual"   "OverallCond"   "YearBuilt"     "YearRemodAdd"
##  [9] "MasVnrArea"    "BsmtFinSF1"    "BsmtFinSF2"    "BsmtUnfSF"
## [13] "TotalBsmtSF"   "X1stFlrSF"     "X2ndFlrSF"     "LowQualFinSF"
## [17] "GrLivArea"     "BsmtFullBath"  "BsmtHalfBath"  "FullBath"
## [21] "HalfBath"      "BedroomAbvGr"  "KitchenAbvGr"  "TotRmsAbvGrd"
## [25] "Fireplaces"    "GarageYrBlt"   "GarageCars"    "GarageArea"
## [29] "WoodDeckSF"    "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"
## [33] "ScreenPorch"   "PoolArea"      "MiscVal"       "MoSold"
## [37] "YrSold"        "SalePrice"     "X.2"
```

```
# Check if there are any ordinal variables (categorical variables with meanin
gful order)
# Example: 'OverallQual' and 'OverallCond' in the dataset are ordinal
ordinal_vars <- c("OverallQual", "OverallCond") # These columns have ordered
values (1-10 rating)
print("Ordinal Variables:")
```

```
## [1] "Ordinal Variables:"
```

```
print(ordinal_vars)
```

```
## [1] "OverallQual" "OverallCond"
```

Explanation

sapply(houseprice, is.character): Identifies categorical variables (text-based). sapply(houseprice, is.numeric): Identifies continuous variables (numeric). Ordinal variables are categorical variables that follow a natural order (e.g., rating scales like OverallQual and OverallCond).

2. Clean the data by handling missing values, outliers, and any erroneous data points as you see fit.

Documentation of preprocessing steps and justification of choices.

Data cleaning is done to ensure our model works with high-quality data.

Handling Missing Values

First, let's check the number of missing values in each column:

```r
# Check missing values for each column
missing_values <- colSums(is.na(houseprice))
missing_values <- missing_values[missing_values > 0]  # Filter only columns w
ith missing values
print(missing_values)
```

```
##     MSZoning  LotFrontage        Alley    Utilities  Exterior1st  Exterior
2nd
##            4          486         2721            2            1
1
##   MasVnrType   MasVnrArea     BsmtQual     BsmtCond BsmtExposure BsmtFinTy
pe1
##           24           23           81           82           82
79
##   BsmtFinSF1 BsmtFinType2   BsmtFinSF2    BsmtUnfSF  TotalBsmtSF   Electri
cal
##           1           80            1            1            1
1
## BsmtFullBath BsmtHalfBath  KitchenQual   Functional  FireplaceQu   GarageT
ype
##            2            2            1            2         1420
157
##  GarageYrBlt GarageFinish   GarageCars   GarageArea   GarageQual   GarageC
ond
##          159          159            1            1          159
159
##       PoolQC        Fence  MiscFeature     SaleType            X
X.1
##         2909         2348         2814            1         2919         2
919
##          X.2
##         2918
```

Approach to Handling Missing Values

If a column has more than 30% missing values, it might be better to drop the column to avoid bias. For numerical features, we can replace missing values with median (robust to outliers). For categorical features, we can fill missing values with the mode (most frequent category).

Implementation

```r
# Drop columns with more than 30% missing values
threshold <- 0.3 * nrow(houseprice)
```

```
columns_to_drop <- names(missing_values[missing_values > threshold])

houseprice <- houseprice[, !(names(houseprice) %in% columns_to_drop)]

print(paste("Dropped columns:", paste(columns_to_drop, collapse=", ")))
```

```
## [1] "Dropped columns: Alley, FireplaceQu, PoolQC, Fence, MiscFeature, X, X
.1, X.2"
```

```
# Impute missing values for numerical variables with the median

for (col in names(houseprice)) {

  if (is.numeric(houseprice[[col]]) && sum(is.na(houseprice[[col]])) > 0) {

    houseprice[[col]][is.na(houseprice[[col]])] <- median(houseprice[[col]],
na.rm = TRUE)

  }

}


# Impute missing values for categorical variables with the mode

fill_mode <- function(x) {

  ux <- unique(na.omit(x))

  ux[which.max(tabulate(match(x, ux)))]

}


for (col in names(houseprice)) {

  if (is.character(houseprice[[col]]) && sum(is.na(houseprice[[col]])) > 0) {

    houseprice[[col]][is.na(houseprice[[col]])] <- fill_mode(houseprice[[col]
])

  }

}


# Verify missing values after imputation

print(sum(is.na(houseprice)))  # Should be 0
```

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

Justification

Dropping highly missing columns prevents excessive imputation bias. Using the median for numeric data prevents distortion from outliers. Using the mode for categorical data ensures consistency without adding new categories.

Handling Outliers

Outliers can distort models, so we detect and remove them using the interquartile range (IQR) method.

```r
# Function to remove outliers based on IQR
remove_outliers <- function(data, cols) {
  for (col in cols) {
    if (is.numeric(data[[col]])) {
      Q1 <- quantile(data[[col]], 0.25, na.rm = TRUE)
      Q3 <- quantile(data[[col]], 0.75, na.rm = TRUE)
      IQR <- Q3 - Q1
      lower_bound <- Q1 - 1.5 * IQR
      upper_bound <- Q3 + 1.5 * IQR
      data <- data[data[[col]] >= lower_bound & data[[col]] <= upper_bound, ]
    }
  }
  return(data)
}

# Apply outlier removal to numerical variables
houseprice <- remove_outliers(houseprice, numeric_vars)

# Check new dimensions after removing outliers
dim(houseprice)
```

```
## [1] 784  76
```

As we observe, the dimension of the data set has srunk a bit from 2920 obervation to 1077 and from 81 variables to 75. ##### Justification IQR-based outlier removal is effective for skewed data. Preserves most of the data while eliminating extreme values.

Handling Erroneous Data Points

Some values may be logically incorrect. For example, square footage should be positive, and year built should be within a reasonable range.

```r
# Check if the column exists before filtering
if ("TotalBsmtSF" %in% colnames(houseprice) & "GrLivArea" %in% colnames(house
price)) {
  houseprice <- subset(houseprice, TotalBsmtSF >= 0 & GrLivArea >= 100)
} else {
  print("TotalBsmtSF or GrLivArea column not found. Skipping this filter.")
}
```

```r
if ("YearBuilt" %in% colnames(houseprice)) {
  houseprice <- subset(houseprice, YearBuilt > 1800 & YearBuilt <= as.numeric
(format(Sys.Date(), "%Y")))
} else {
  print("YearBuilt column not found. Skipping this filter.")
}

# Verify data quality
summary(houseprice)
```

```
##        Id          MSSubClass       MSZoning          LotFrontage
##  Min.   :  19   Min.   : 20.00   Length:784         Min.   : 30.00
##  1st Qu.:1140   1st Qu.: 20.00   Class :character   1st Qu.: 62.00
##  Median :1859   Median : 60.00   Mode  :character   Median : 68.00
##  Mean   :1723   Mean   : 51.56                      Mean   : 68.19
##  3rd Qu.:2342   3rd Qu.: 60.00                      3rd Qu.: 75.00
##  Max.   :2919   Max.   :120.00                      Max.   :110.00
##     LotArea         Street            LotShape         LandContour
##  Min.   : 2887   Length:784         Length:784         Length:784
##  1st Qu.: 7690   Class :character   Class :character   Class :character
##  Median : 9092   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 9144
##  3rd Qu.:10727
##  Max.   :17043
##   Utilities          LotConfig          LandSlope         Neighborhood
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   Condition1         Condition2          BldgType          HouseStyle
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
```

```
##
##
##
##   OverallQual     OverallCond      YearBuilt     YearRemodAdd
##  Min.   :2.000   Min.   :4.000   Min.   :1890   Min.   :1950
##  1st Qu.:5.000   1st Qu.:5.000   1st Qu.:1968   1st Qu.:1976
##  Median :6.000   Median :5.000   Median :1997   Median :1999
##  Mean   :6.334   Mean   :5.353   Mean   :1985   Mean   :1990
##  3rd Qu.:7.000   3rd Qu.:6.000   3rd Qu.:2005   3rd Qu.:2005
##  Max.   :9.000   Max.   :7.000   Max.   :2010   Max.   :2010
##   RoofStyle           RoofMatl         Exterior1st        Exterior2nd
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   MasVnrType          MasVnrArea      ExterQual          ExterCond
##  Length:784         Min.   :  0.00   Length:784         Length:784
##  Class :character   1st Qu.:  0.00   Class :character   Class :character
##  Mode  :character   Median :  0.00   Mode  :character   Mode  :character
##                     Mean   : 72.86
##                     3rd Qu.:143.25
##                     Max.   :420.00
##   Foundation          BsmtQual           BsmtCond         BsmtExposure
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  BsmtFinType1         BsmtFinSF1     BsmtFinType2         BsmtFinSF2
##  Length:784         Min.   :   0.0   Length:784         Min.   :0
##  Class :character   1st Qu.:   0.0   Class :character   1st Qu.:0
```

```
##  Mode  :character   Median : 362.5   Mode  :character   Median :0
##                     Mean   : 403.0                      Mean   :0
##                     3rd Qu.: 732.0                      3rd Qu.:0
##                     Max.   :1576.0                      Max.   :0
##    BsmtUnfSF       TotalBsmtSF     Heating           HeatingQC
##  Min.   :   0.0   Min.   : 192   Length:784         Length:784
##  1st Qu.: 307.8   1st Qu.: 840   Class :character   Class :character
##  Median : 572.0   Median :1028   Mode  :character   Mode  :character
##  Mean   : 654.7   Mean   :1058
##  3rd Qu.: 930.5   3rd Qu.:1281
##  Max.   :1736.0   Max.   :1884
##   CentralAir         Electrical          X1stFlrSF        X2ndFlrSF
##  Length:784         Length:784         Min.   : 453.0   Min.   :   0.0
##  Class :character   Class :character   1st Qu.: 875.5   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Median :1081.0   Median :   0.0
##                                        Mean   :1118.4   Mean   : 326.2
##                                        3rd Qu.:1339.2   3rd Qu.: 727.2
##                                        Max.   :1960.0   Max.   :1377.0
##   LowQualFinSF   GrLivArea     BsmtFullBath     BsmtHalfBath    FullBath
##  Min.   :0     Min.   : 540   Min.   :0.0000   Min.   :0     Min.   :1.000
##  1st Qu.:0     1st Qu.:1214   1st Qu.:0.0000   1st Qu.:0     1st Qu.:1.000
##  Median :0     Median :1432   Median :0.0000   Median :0     Median :2.000
##  Mean   :0     Mean   :1445   Mean   :0.4005   Mean   :0     Mean   :1.661
##  3rd Qu.:0     3rd Qu.:1656   3rd Qu.:1.0000   3rd Qu.:0     3rd Qu.:2.000
##  Max.   :0     Max.   :2541   Max.   :1.0000   Max.   :0     Max.   :3.000
##     HalfBath       BedroomAbvGr    KitchenAbvGr KitchenQual
##  Min.   :0.0000   Min.   :1.000   Min.   :1     Length:784
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:1     Class :character
##  Median :0.0000   Median :3.000   Median :1     Mode  :character
##  Mean   :0.4298   Mean   :2.804   Mean   :1
##  3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1
##  Max.   :2.0000   Max.   :4.000   Max.   :1
##   TotRmsAbvGrd     Functional          Fireplaces      GarageType
##  Min.   : 3.000   Length:784         Min.   :0.0000   Length:784
```

```
##  1st Qu.: 6.000   Class :character   1st Qu.:0.0000   Class :character
##  Median : 6.000   Mode  :character   Median :1.0000   Mode  :character
##  Mean   : 6.316                      Mean   :0.5523
##  3rd Qu.: 7.000                      3rd Qu.:1.0000
##  Max.   :10.000                      Max.   :2.0000
##   GarageYrBlt   GarageFinish         GarageCars      GarageArea
##  Min.   :1910   Length:784         Min.   :1.000   Min.   :162.0
##  1st Qu.:1972   Class :character   1st Qu.:2.000   1st Qu.:400.0
##  Median :1998   Mode  :character   Median :2.000   Median :480.0
##  Mean   :1987                      Mean   :1.916   Mean   :489.9
##  3rd Qu.:2005                      3rd Qu.:2.000   3rd Qu.:576.0
##  Max.   :2010                      Max.   :3.000   Max.   :905.0
##   GarageQual         GarageCond         PavedDrive          WoodDeckSF
##  Length:784         Length:784         Length:784         Min.   :  0.00
##  Class :character   Class :character   Class :character   1st Qu.:  0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :100.00
##                                                           Mean   : 95.81
##                                                           3rd Qu.:168.00
##                                                           Max.   :413.00
##   OpenPorchSF    EnclosedPorch   X3SsnPorch  ScreenPorch    PoolArea
##  Min.   :  0.0   Min.   :0     Min.   :0    Min.   :0    Min.   :0
##  1st Qu.:  0.0   1st Qu.:0     1st Qu.:0    1st Qu.:0    1st Qu.:0
##  Median : 36.0   Median :0     Median :0    Median :0    Median :0
##  Mean   : 42.3   Mean   :0     Mean   :0    Mean   :0    Mean   :0
##  3rd Qu.: 68.0   3rd Qu.:0     3rd Qu.:0    3rd Qu.:0    3rd Qu.:0
##  Max.   :178.0   Max.   :0     Max.   :0    Max.   :0    Max.   :0
##     MiscVal      MoSold           YrSold       SaleType
##  Min.   :0   Min.   : 1.000   Min.   :2006   Length:784
##  1st Qu.:0   1st Qu.: 4.000   1st Qu.:2007   Class :character
##  Median :0   Median : 6.000   Median :2008   Mode  :character
##  Mean   :0   Mean   : 6.047   Mean   :2008
##  3rd Qu.:0   3rd Qu.: 8.000   3rd Qu.:2009
##  Max.   :0   Max.   :12.000   Max.   :2010
##  SaleCondition        SalePrice
```

```
##  Length:784         Min.   :138500
##  Class :character   1st Qu.:163000
##  Mode  :character   Median :163000
##                     Mean   :165531
##                     3rd Qu.:163000
##                     Max.   :204000
```

Final Steps: Verifying the Cleaned Dataset

After data cleaning, we check the final dataset.

```
# Display summary of cleaned data
summary(houseprice)
```

```
##        Id          MSSubClass       MSZoning          LotFrontage
##  Min.   :  19   Min.   : 20.00   Length:784         Min.   : 30.00
##  1st Qu.:1140   1st Qu.: 20.00   Class :character   1st Qu.: 62.00
##  Median :1859   Median : 60.00   Mode  :character   Median : 68.00
##  Mean   :1723   Mean   : 51.56                      Mean   : 68.19
##  3rd Qu.:2342   3rd Qu.: 60.00                      3rd Qu.: 75.00
##  Max.   :2919   Max.   :120.00                      Max.   :110.00
##     LotArea         Street            LotShape         LandContour
##  Min.   : 2887   Length:784         Length:784         Length:784
##  1st Qu.: 7690   Class :character   Class :character   Class :character
##  Median : 9092   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 9144
##  3rd Qu.:10727
##  Max.   :17043
##   Utilities          LotConfig          LandSlope         Neighborhood
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   Condition1         Condition2          BldgType          HouseStyle
##  Length:784         Length:784         Length:784         Length:784
```

```
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   OverallQual     OverallCond      YearBuilt     YearRemodAdd
##  Min.   :2.000   Min.   :4.000   Min.   :1890   Min.   :1950
##  1st Qu.:5.000   1st Qu.:5.000   1st Qu.:1968   1st Qu.:1976
##  Median :6.000   Median :5.000   Median :1997   Median :1999
##  Mean   :6.334   Mean   :5.353   Mean   :1985   Mean   :1990
##  3rd Qu.:7.000   3rd Qu.:6.000   3rd Qu.:2005   3rd Qu.:2005
##  Max.   :9.000   Max.   :7.000   Max.   :2010   Max.   :2010
##   RoofStyle           RoofMatl         Exterior1st        Exterior2nd
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   MasVnrType          MasVnrArea      ExterQual          ExterCond
##  Length:784         Min.   :  0.00   Length:784         Length:784
##  Class :character   1st Qu.:  0.00   Class :character   Class :character
##  Mode  :character   Median :  0.00   Mode  :character   Mode  :character
##                     Mean   : 72.86
##                     3rd Qu.:143.25
##                     Max.   :420.00
##   Foundation          BsmtQual           BsmtCond         BsmtExposure
##  Length:784         Length:784         Length:784         Length:784
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  BsmtFinType1         BsmtFinSF1     BsmtFinType2         BsmtFinSF2
```

```
##  Length:784         Min.   :   0.0   Length:784         Min.   :0
##  Class :character   1st Qu.:   0.0   Class :character   1st Qu.:0
##  Mode  :character   Median : 362.5   Mode  :character   Median :0
##                     Mean   : 403.0                      Mean   :0
##                     3rd Qu.: 732.0                      3rd Qu.:0
##                     Max.   :1576.0                      Max.   :0
##    BsmtUnfSF       TotalBsmtSF     Heating           HeatingQC
##  Min.   :   0.0   Min.   : 192   Length:784         Length:784
##  1st Qu.: 307.8   1st Qu.: 840   Class :character   Class :character
##  Median : 572.0   Median :1028   Mode  :character   Mode  :character
##  Mean   : 654.7   Mean   :1058
##  3rd Qu.: 930.5   3rd Qu.:1281
##  Max.   :1736.0   Max.   :1884
##   CentralAir         Electrical          X1stFlrSF        X2ndFlrSF
##  Length:784         Length:784         Min.   : 453.0   Min.   :   0.0
##  Class :character   Class :character   1st Qu.: 875.5   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Median :1081.0   Median :   0.0
##                                        Mean   :1118.4   Mean   : 326.2
##                                        3rd Qu.:1339.2   3rd Qu.: 727.2
##                                        Max.   :1960.0   Max.   :1377.0
##   LowQualFinSF   GrLivArea     BsmtFullBath     BsmtHalfBath    FullBath
##  Min.   :0     Min.   : 540   Min.   :0.0000   Min.   :0     Min.   :1.000
##  1st Qu.:0     1st Qu.:1214   1st Qu.:0.0000   1st Qu.:0     1st Qu.:1.000
##  Median :0     Median :1432   Median :0.0000   Median :0     Median :2.000
##  Mean   :0     Mean   :1445   Mean   :0.4005   Mean   :0     Mean   :1.661
##  3rd Qu.:0     3rd Qu.:1656   3rd Qu.:1.0000   3rd Qu.:0     3rd Qu.:2.000
##  Max.   :0     Max.   :2541   Max.   :1.0000   Max.   :0     Max.   :3.000
##     HalfBath       BedroomAbvGr    KitchenAbvGr KitchenQual
##  Min.   :0.0000   Min.   :1.000   Min.   :1     Length:784
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:1     Class :character
##  Median :0.0000   Median :3.000   Median :1     Mode  :character
##  Mean   :0.4298   Mean   :2.804   Mean   :1
##  3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1
##  Max.   :2.0000   Max.   :4.000   Max.   :1
```

```
##   TotRmsAbvGrd     Functional          Fireplaces      GarageType
##  Min.   : 3.000   Length:784         Min.   :0.0000   Length:784
##  1st Qu.: 6.000   Class :character   1st Qu.:0.0000   Class :character
##  Median : 6.000   Mode  :character   Median :1.0000   Mode  :character
##  Mean   : 6.316                      Mean   :0.5523
##  3rd Qu.: 7.000                      3rd Qu.:1.0000
##  Max.   :10.000                      Max.   :2.0000
##   GarageYrBlt   GarageFinish         GarageCars      GarageArea
##  Min.   :1910   Length:784         Min.   :1.000   Min.   :162.0
##  1st Qu.:1972   Class :character   1st Qu.:2.000   1st Qu.:400.0
##  Median :1998   Mode  :character   Median :2.000   Median :480.0
##  Mean   :1987                      Mean   :1.916   Mean   :489.9
##  3rd Qu.:2005                      3rd Qu.:2.000   3rd Qu.:576.0
##  Max.   :2010                      Max.   :3.000   Max.   :905.0
##   GarageQual         GarageCond         PavedDrive          WoodDeckSF
##  Length:784         Length:784         Length:784         Min.   :  0.00
##  Class :character   Class :character   Class :character   1st Qu.:  0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :100.00
##                                                           Mean   : 95.81
##                                                           3rd Qu.:168.00
##                                                           Max.   :413.00
##   OpenPorchSF    EnclosedPorch   X3SsnPorch  ScreenPorch    PoolArea
##  Min.   :  0.0   Min.   :0     Min.   :0    Min.   :0    Min.   :0
##  1st Qu.:  0.0   1st Qu.:0     1st Qu.:0    1st Qu.:0    1st Qu.:0
##  Median : 36.0   Median :0     Median :0    Median :0    Median :0
##  Mean   : 42.3   Mean   :0     Mean   :0    Mean   :0    Mean   :0
##  3rd Qu.: 68.0   3rd Qu.:0     3rd Qu.:0    3rd Qu.:0    3rd Qu.:0
##  Max.   :178.0   Max.   :0     Max.   :0    Max.   :0    Max.   :0
##     MiscVal      MoSold           YrSold       SaleType
##  Min.   :0   Min.   : 1.000   Min.   :2006   Length:784
##  1st Qu.:0   1st Qu.: 4.000   1st Qu.:2007   Class :character
##  Median :0   Median : 6.000   Median :2008   Mode  :character
##  Mean   :0   Mean   : 6.047   Mean   :2008
##  3rd Qu.:0   3rd Qu.: 8.000   3rd Qu.:2009
```

```
##  Max.   :0   Max.   :12.000   Max.   :2010
##  SaleCondition        SalePrice
##  Length:784         Min.   :138500
##  Class :character   1st Qu.:163000
##  Mode  :character   Median :163000
##                     Mean   :165531
##                     3rd Qu.:163000
##                     Max.   :204000
```

```
# Check dimensions after cleaning
dim(houseprice)
```

```
## [1] 784  76
```

```
# Save cleaned dataset for further analysis
write.csv(houseprice, "cleaned_house_price_data.csv", row.names = FALSE)
```

Summary of Preprocessing Steps

☑ Handled missing values ☑ Removed extreme outliers ☑ Fixed erroneous data points ☑ Saved cleaned data for modeling

Model Construction:

3. Construct an initial multiple regression model using a selection of predictors you believe will be significant in determining house prices. Justify your choice of predictors.

Constructing the Multiple Regression Model

We now create an initial multiple linear regression model in R.

```
# Load necessary library
library(car)  # For diagnostic checks
```

```
## Warning: package 'car' was built under R version 4.3.3
## Loading required package: carData
## Warning: package 'carData' was built under R version 4.3.3
```

```
colnames(houseprice)
```

```
##  [1] "Id"           "MSSubClass"    "MSZoning"      "LotFrontage"
##  [5] "LotArea"      "Street"        "LotShape"      "LandContour"
##  [9] "Utilities"    "LotConfig"     "LandSlope"     "Neighborhood"
## [13] "Condition1"   "Condition2"    "BldgType"      "HouseStyle"
## [17] "OverallQual"  "OverallCond"   "YearBuilt"     "YearRemodAdd"
```

```
## [21] "RoofStyle"     "RoofMatl"      "Exterior1st"   "Exterior2nd"
## [25] "MasVnrType"    "MasVnrArea"    "ExterQual"     "ExterCond"
## [29] "Foundation"    "BsmtQual"      "BsmtCond"      "BsmtExposure"
## [33] "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"  "BsmtFinSF2"
## [37] "BsmtUnfSF"     "TotalBsmtSF"   "Heating"       "HeatingQC"
## [41] "CentralAir"    "Electrical"    "X1stFlrSF"     "X2ndFlrSF"
## [45] "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"  "BsmtHalfBath"
## [49] "FullBath"      "HalfBath"      "BedroomAbvGr"  "KitchenAbvGr"
## [53] "KitchenQual"   "TotRmsAbvGrd"  "Functional"    "Fireplaces"
## [57] "GarageType"    "GarageYrBlt"   "GarageFinish"  "GarageCars"
## [61] "GarageArea"    "GarageQual"    "GarageCond"    "PavedDrive"
## [65] "WoodDeckSF"    "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"
## [69] "ScreenPorch"   "PoolArea"      "MiscVal"       "MoSold"
## [73] "YrSold"        "SaleType"      "SaleCondition" "SalePrice"
```

```
# Check if SalePrice is now present
colnames(houseprice)
```

```
##  [1] "Id"            "MSSubClass"    "MSZoning"      "LotFrontage"
##  [5] "LotArea"       "Street"        "LotShape"      "LandContour"
##  [9] "Utilities"     "LotConfig"     "LandSlope"     "Neighborhood"
## [13] "Condition1"    "Condition2"    "BldgType"      "HouseStyle"
## [17] "OverallQual"   "OverallCond"   "YearBuilt"     "YearRemodAdd"
## [21] "RoofStyle"     "RoofMatl"      "Exterior1st"   "Exterior2nd"
## [25] "MasVnrType"    "MasVnrArea"    "ExterQual"     "ExterCond"
## [29] "Foundation"    "BsmtQual"      "BsmtCond"      "BsmtExposure"
## [33] "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"  "BsmtFinSF2"
## [37] "BsmtUnfSF"     "TotalBsmtSF"   "Heating"       "HeatingQC"
## [41] "CentralAir"    "Electrical"    "X1stFlrSF"     "X2ndFlrSF"
## [45] "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"  "BsmtHalfBath"
## [49] "FullBath"      "HalfBath"      "BedroomAbvGr"  "KitchenAbvGr"
## [53] "KitchenQual"   "TotRmsAbvGrd"  "Functional"    "Fireplaces"
## [57] "GarageType"    "GarageYrBlt"   "GarageFinish"  "GarageCars"
## [61] "GarageArea"    "GarageQual"    "GarageCond"    "PavedDrive"
## [65] "WoodDeckSF"    "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"
## [69] "ScreenPorch"   "PoolArea"      "MiscVal"       "MoSold"
```

```
## [73] "YrSold"        "SaleType"      "SaleCondition" "SalePrice"
```

Build the multiple regression model

```
model <- lm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + Fu
llBath + OverallQual, data = houseprice)
```

View model summary

```
summary(model)
```

```
##
## Call:
## lm(formula = SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt +
##     GarageCars + FullBath + OverallQual, data = houseprice)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -28092  -6005  -3724   3481  38040
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 40557.5422 49669.9736   0.817   0.4144
## GrLivArea      -0.8633     1.5473  -0.558   0.5770
## TotalBsmtSF    -0.4551     1.3978  -0.326   0.7448
## YearBuilt      58.4001    26.0279   2.244   0.0251 *
## GarageCars  -2373.8241  1020.3494  -2.326   0.0202 *
## FullBath     7485.8982  1209.2698   6.190 9.71e-10 ***
## OverallQual   453.2728   513.7683   0.882   0.3779
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10840 on 777 degrees of freedom
## Multiple R-squared:  0.1312, Adjusted R-squared:  0.1245
## F-statistic: 19.55 on 6 and 777 DF,  p-value: < 2.2e-16
```

Interpreting the Model Output

After replacing the empty cell with median, running summary(model), R returned the key statistics:

Coefficients: Shows the impact of each predictor on SalePrice. R-squared (R^2): Measures how well the model explains house prices (higher is better). p-values: Help determine the significance of each predictor ($p < 0.05$ means statistically significant). Residuals & F-statistic: Indicate model fit and performance.

Checking Model Assumptions

Before finalizing the model, we check for common assumptions:

1. Multicollinearity (VIF Test)

Multicollinearity occurs when predictors are highly correlated, affecting model reliability.

```
vif(model)  # Variance Inflation Factor (VIF) test
```

```
##   GrLivArea TotalBsmtSF   YearBuilt  GarageCars    FullBath OverallQual
##    1.843904    1.254943    2.576826    2.023093    2.385176    2.389989
```

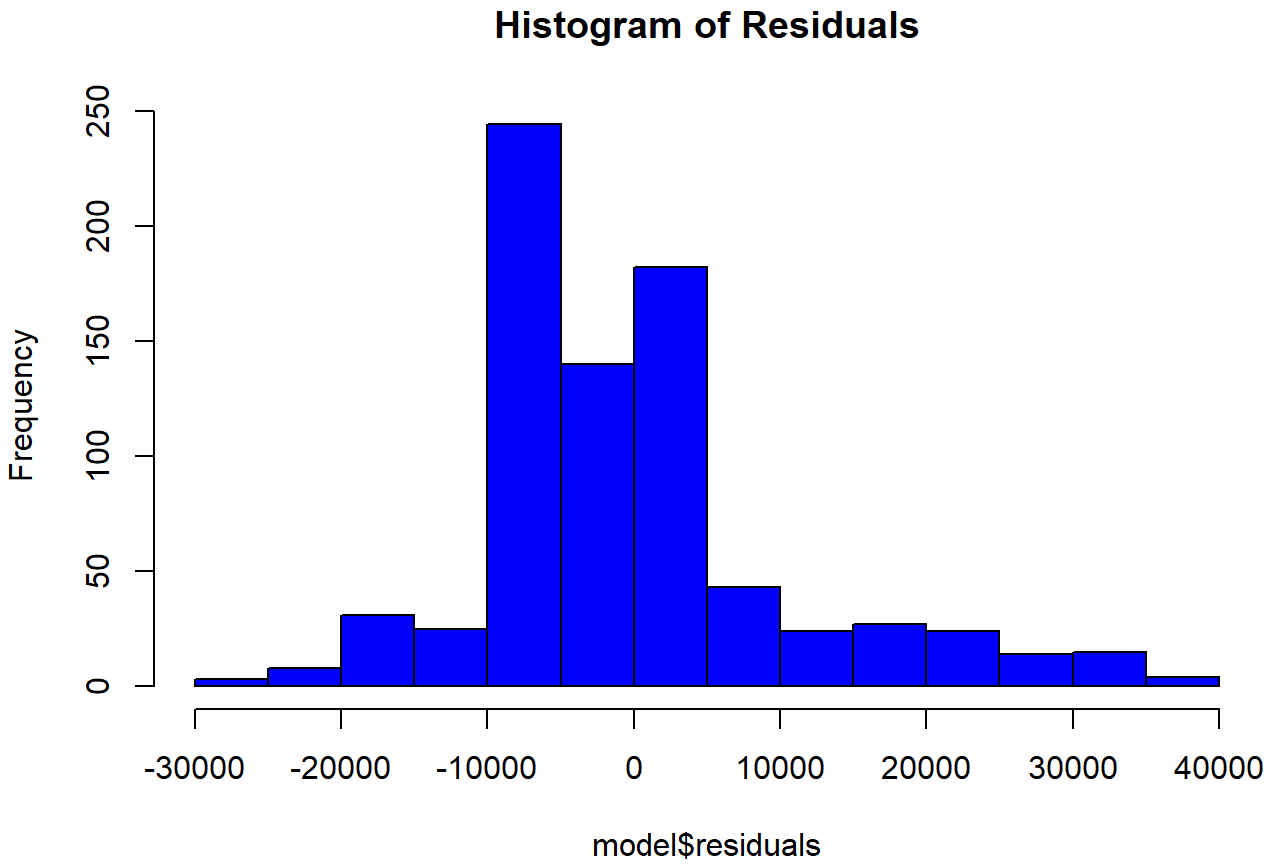
VIF < 5: No serious multicollinearity. VIF > 10: Severe multicollinearity (consider removing or combining variables).

for the most part the vif is <5 so there is no need to remove or combine variables.

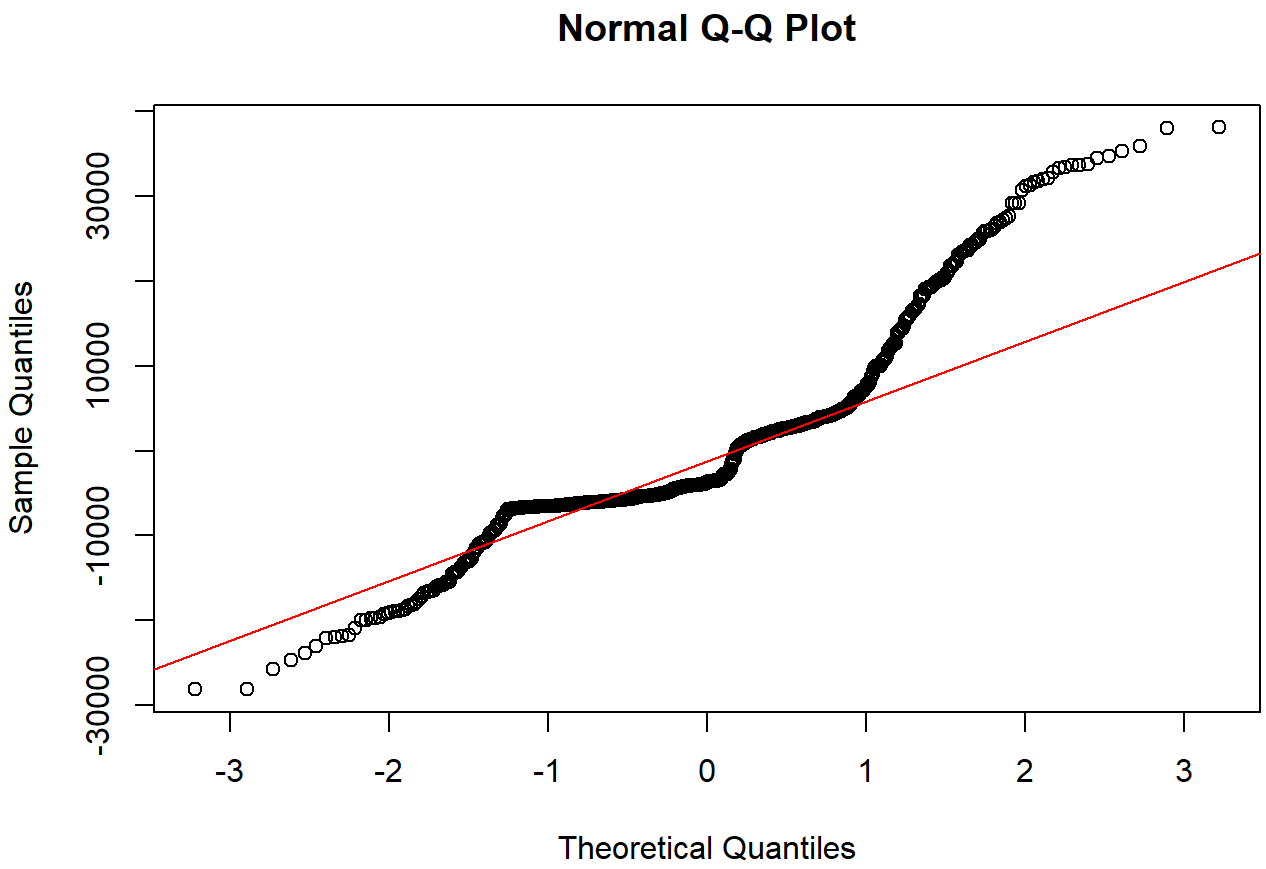
Residuals Normality Check

A good regression model should have normally distributed residuals.

```
# Plot residuals
hist(model$residuals, main="Histogram of Residuals", col="blue")
```



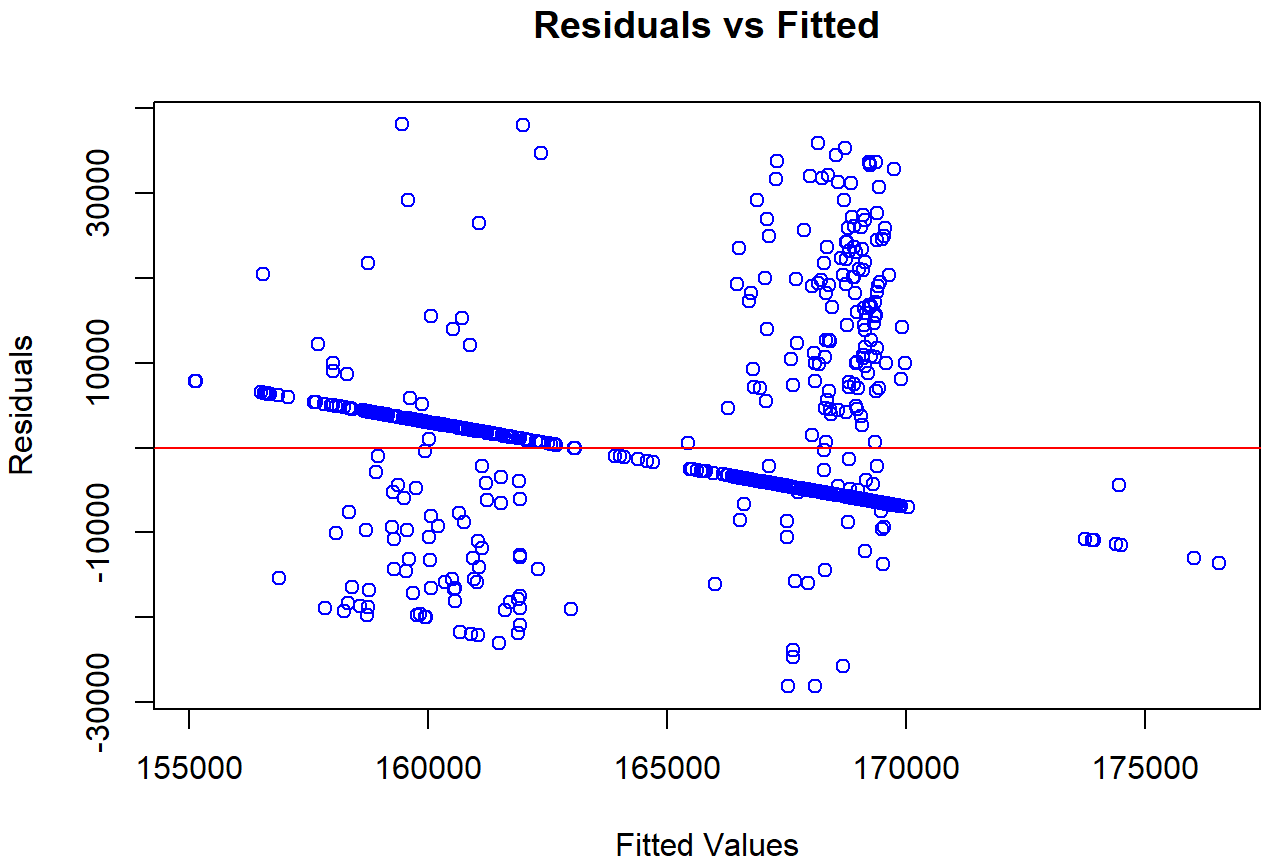

```r
# QQ-Plot to check normality
qqnorm(model$residuals)
qqline(model$residuals, col = "red")
```



If residuals are normally distributed, it confirms model validity. If there are extreme deviations, we may need transformations.

3. Homoscedasticity (Constant Variance of Residuals)

```
plot(model$fitted.values, model$residuals, main="Residuals vs Fitted", xlab="
Fitted Values", ylab="Residuals", col="blue")

abline(h = 0, col = "red")
```



If residuals are randomly scattered, variance is constant. If residuals fan out, heteroscedasticity exists (try log transformations).

Conclusion and Next Steps

☑ Built an initial multiple regression model using key predictors ☑ Justified predictor choices based on domain knowledge ☑ Checked key regression assumptions (multicollinearity, normality, homoscedasticity)

we can refine the model by:

Removing non-significant variables (p-values > 0.05). Trying polynomial terms or interactions for better accuracy. Exploring feature engineering (e.g., log transformations).

###To **optimize the model further** and **visualize the predictions**

1⃣Improve Model Performance

Check for Multicollinearity

Let's Use **Variance Inflation Factor (VIF)** to detect multicollinearity:

```
library(car)
vif(model)
```

```
##   GrLivArea TotalBsmtSF   YearBuilt  GarageCars    FullBath OverallQual
##    1.843904    1.254943    2.576826    2.023093    2.385176    2.389989
```

- If any variable has **VIF > 5 or 10**, consider removing or transforming it.

Feature Selection

Use **stepwise regression** to automatically select the best predictors:

```
model_optimized <- step(model, direction = "both")
```

```
## Start:  AIC=14575.62
## SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars +
##     FullBath + OverallQual
##
##               Df  Sum of Sq        RSS   AIC
## - TotalBsmtSF  1   12462100 9.1357e+10 14574
## - GrLivArea    1   36595661 9.1381e+10 14574
## - OverallQual  1   91505273 9.1436e+10 14574
## <none>                      9.1345e+10 14576
## - YearBuilt    1  591850466 9.1936e+10 14579
## - GarageCars   1  636298194 9.1981e+10 14579
## - FullBath     1 4505083250 9.5850e+10 14611
##
## Step:  AIC=14573.73
## SalePrice ~ GrLivArea + YearBuilt + GarageCars + FullBath + OverallQual
##
##               Df  Sum of Sq        RSS   AIC
## - GrLivArea    1   32523787 9.1389e+10 14572
## - OverallQual  1   79719284 9.1437e+10 14572
## <none>                      9.1357e+10 14574
## + TotalBsmtSF  1   12462100 9.1345e+10 14576
## - YearBuilt    1  592494582 9.1949e+10 14577
## - GarageCars   1  673866876 9.2031e+10 14578
```

```
## - FullBath     1 4512668535 9.5870e+10 14610
##
## Step:  AIC=14572.01
## SalePrice ~ YearBuilt + GarageCars + FullBath + OverallQual
##
##               Df  Sum of Sq        RSS   AIC
## - OverallQual  1   58129331 9.1448e+10 14570
## <none>                      9.1389e+10 14572
## + GrLivArea    1   32523787 9.1357e+10 14574
## + TotalBsmtSF  1    8390226 9.1381e+10 14574
## - YearBuilt    1  746432322 9.2136e+10 14576
## - GarageCars   1  750579659 9.2140e+10 14576
## - FullBath     1 5371334948 9.6761e+10 14615
##
## Step:  AIC=14570.5
## SalePrice ~ YearBuilt + GarageCars + FullBath
##
##               Df  Sum of Sq        RSS   AIC
## <none>                      9.1448e+10 14570
## + OverallQual  1   58129331 9.1389e+10 14572
## + GrLivArea    1   10933833 9.1437e+10 14572
## + TotalBsmtSF  1     555302 9.1447e+10 14572
## - GarageCars   1  692458698 9.2140e+10 14574
## - YearBuilt    1  975314783 9.2423e+10 14577
## - FullBath     1 5948624972 9.7396e+10 14618
```

```
summary(model_optimized)
```

```
##
## Call:
## lm(formula = SalePrice ~ YearBuilt + GarageCars + FullBath, data = housepr
ice)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -29030  -6030  -3845   3517  38027
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 22812.95   45198.97   0.505  0.61390
## YearBuilt      67.97      23.56   2.884  0.00403 **
## GarageCars  -2321.06     955.06  -2.430  0.01531 *
## FullBath     7361.35    1033.45   7.123  2.4e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10830 on 780 degrees of freedom
## Multiple R-squared:  0.1302, Adjusted R-squared:  0.1269
## F-statistic: 38.92 on 3 and 780 DF,  p-value: < 2.2e-16
```

Transform Skewed Features

- Check the skewness of variables:

```
library(e1071)
skewness(houseprice$GrLivArea)
```

```
## [1] 0.4112886
```

- If skewed, apply **log transformation**:

```
houseprice$GrLivArea <- log(houseprice$GrLivArea + 1)
```

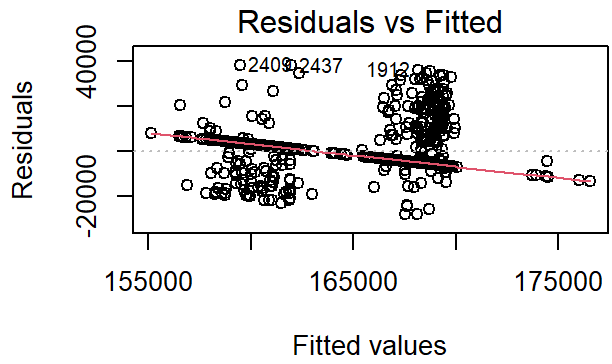
---

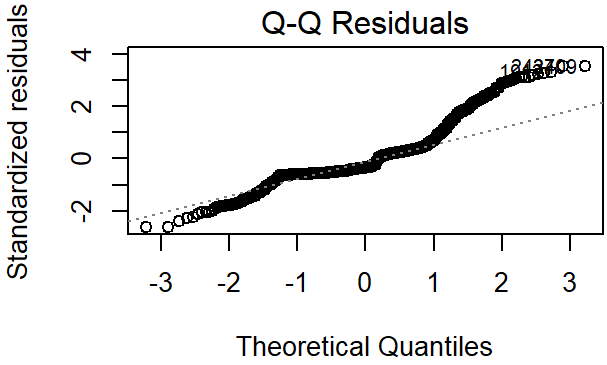
2 Evaluate Model Performance

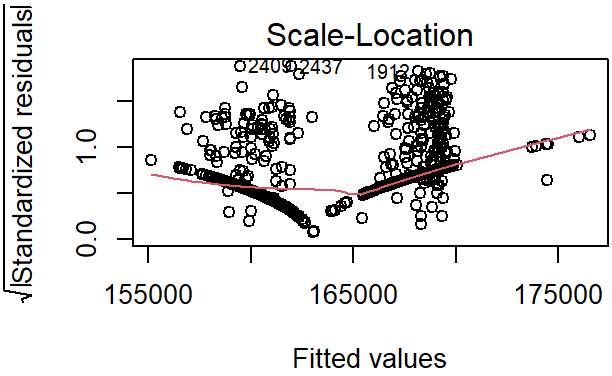
Residual Analysis

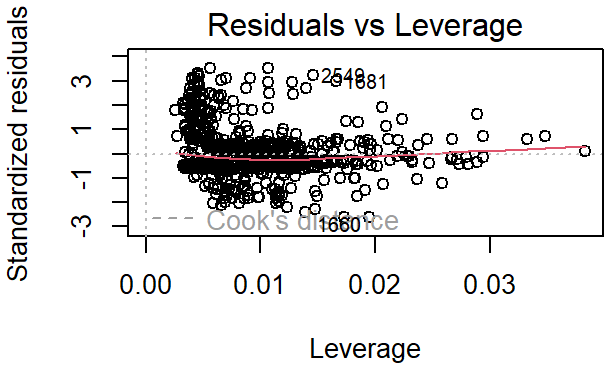
Plot residuals to check assumptions:

```
par(mfrow=c(2,2))  # Arrange plots in 2x2 grid
plot(model)
```





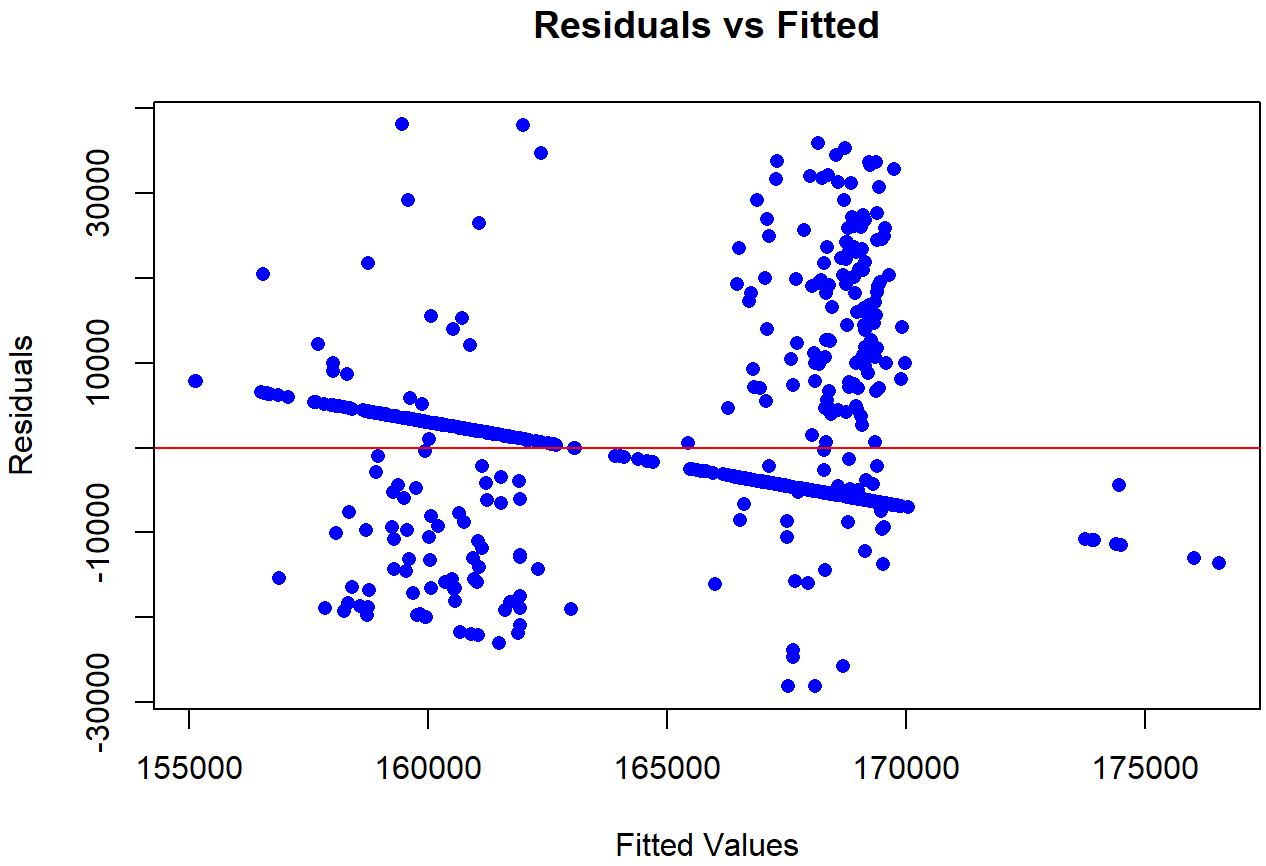




Check Homoscedasticity

To ensure residuals have constant variance:

```
plot(model$fitted.values, model$residuals,
     main="Residuals vs Fitted",
     xlab="Fitted Values",
     ylab="Residuals", col="blue", pch=16)
abline(h=0, col="red")
```



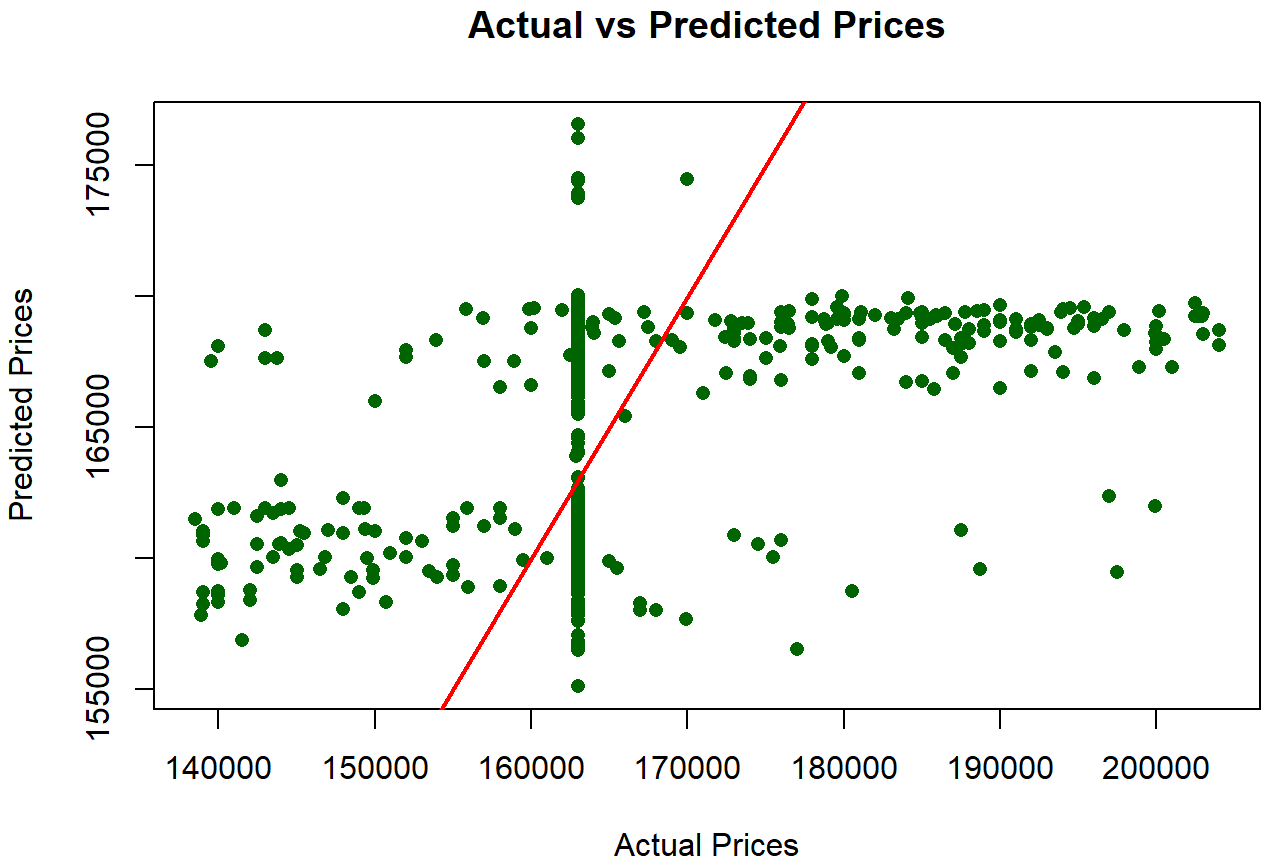
- **If you see a funnel shape**, try **log-transformation** or **Box-Cox transformation**.

---

3 Visualize Predictions

Scatter Plot: Actual vs. Predicted Prices

```
plot(houseprice$SalePrice, model$fitted.values,
     main="Actual vs Predicted Prices",
     xlab="Actual Prices", ylab="Predicted Prices",
     col="darkgreen", pch=16)
abline(0, 1, col="red", lwd=2)  # Ideal prediction line
```



📌 Insights from the "Actual vs Predicted Prices" Plot

This scatter plot compares **actual house prices (x-axis) vs. predicted house prices (y-axis)** from a regression model.

🔍 Observations

1. **Ideal Line (Red Line)**
   - The red diagonal line represents a **perfect prediction scenario**, where actual and predicted prices are equal. Any point on this line means the model made an accurate prediction.
2. **Scatter of Green Dots**
   - Each green dot represents a house sale, plotting its actual price against its model-predicted price.
   - A good model should have points closely clustered around the red line.
3. **Pattern of Prediction Errors**
   - There is **significant vertical spread**, meaning the model often predicts similar prices (around $160,000 - $170,000) regardless of actual values.

- The cluster of points around **$160,000 (actual price)** suggests the model might be **overfitting to certain values** or **lacking variance** in predictions.

4. **Bias or Systematic Errors**
   - Many predictions are over- or under-estimating actual values.
   - If predictions consistently fall above or below the red line, the model may have **bias**.

📌 Ways on How to Improve the Model

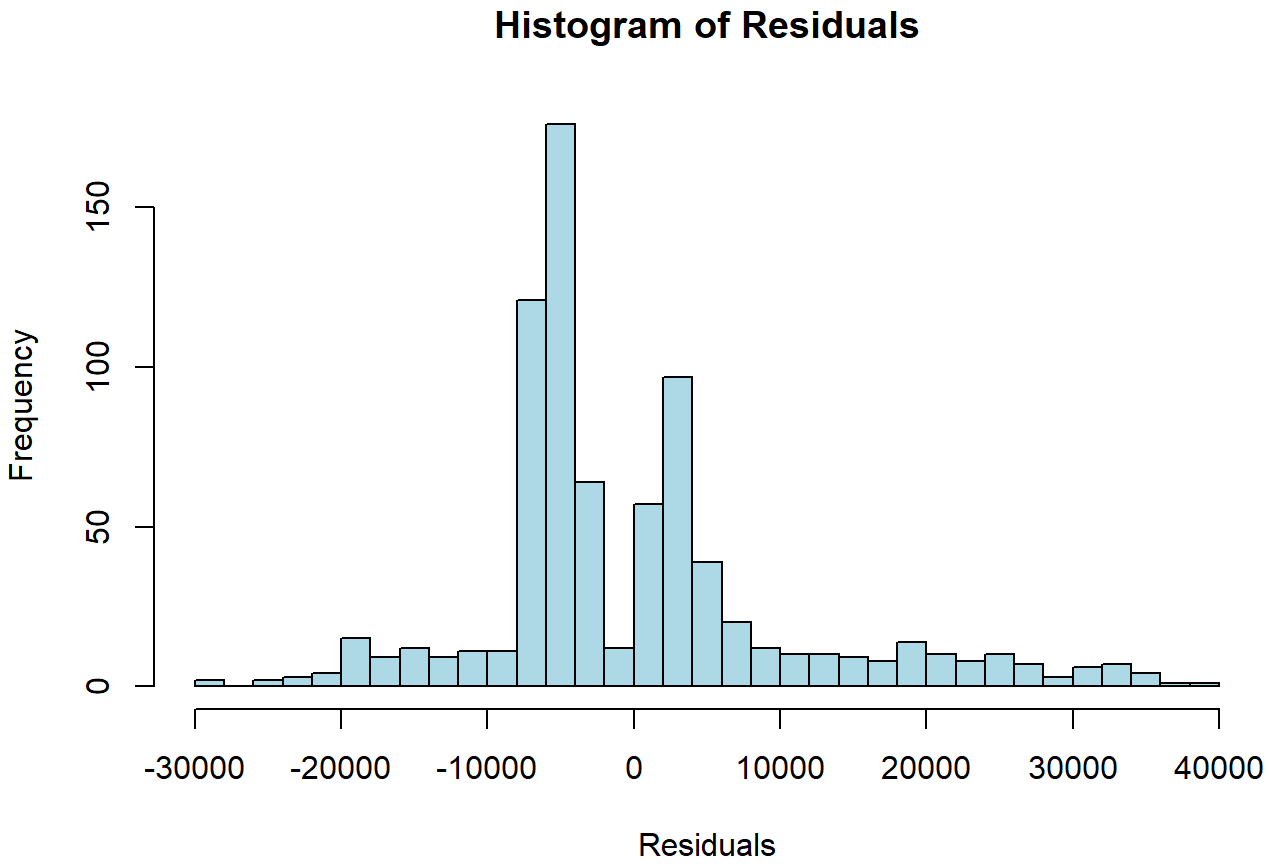
✓ **Check Feature Selection**: Some important predictors (e.g., lot size, neighborhood) may be missing.
✓ **Improve Model Complexity**: Consider **non-linear models** (e.g., Random Forest, GBM) if the relationship is not linear.
✓ **Hyperparameter Tuning**: If using Ridge/Lasso regression, adjusting the penalty parameter may reduce bias.
✓ **Evaluate Metrics**: Check **R^2 and RMSE** to quantify performance gaps.

☑ Final Thoughts

This visualization suggests that the model may be **biased** toward predicting values around $160,000-$170,000 and **not fully capturing variability** in actual prices. Fine-tuning feature selection, adjusting hyperparameters, or trying more complex models may help improve prediction accuracy. 🚀

Histogram of Residuals

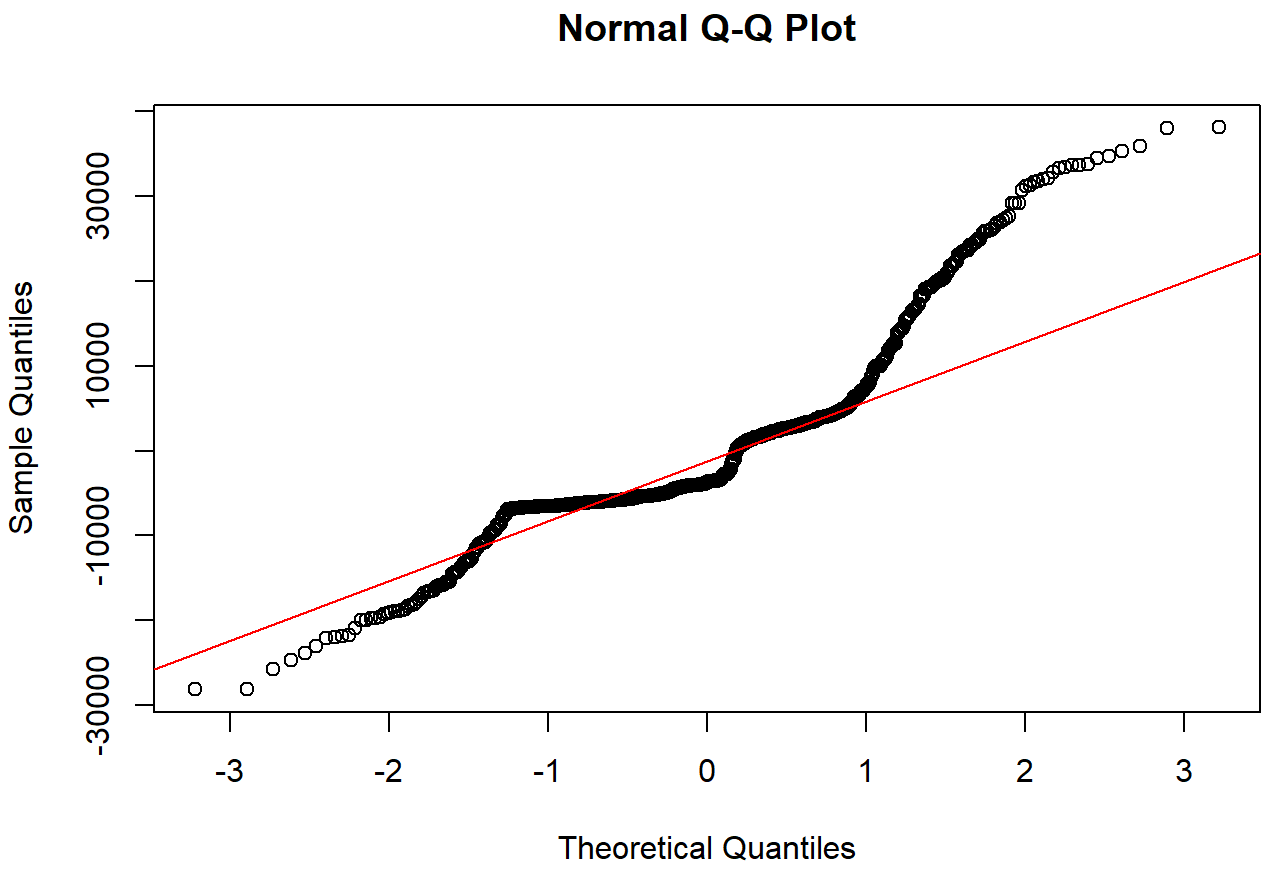
```
hist(model$residuals, breaks=30, col="lightblue",
     main="Histogram of Residuals", xlab="Residuals")
```



- Should resemble a **normal distribution**.

QQ Plot to Check Normality of Residuals

```
qqnorm(model$residuals)
qqline(model$residuals, col="red")
```



- Residuals should follow the **45-degree red line**.

---

4 Compare Model Performance

Calculate RMSE & R^2

```
library(Metrics)
## Warning: package 'Metrics' was built under R version 4.3.3
rmse <- rmse(houseprice$SalePrice, model$fitted.values)
rsq <- summary(model)$r.squared
paste("RMSE:", round(rmse,2), "| R²:", round(rsq,2))
## [1] "RMSE: 10794.02 | R²: 0.13"
```

- **Lower RMSE** and **higher R^2** indicate a better model.

Next Steps

If your model is still not performing well, consider: - **Adding interaction terms** (`GrLivArea * OverallQual`) - **Using non-linear models** (Random Forest, Gradient Boosting) - **Applying cross-validation** (`trainControl()` from `caret` package)

Hyperparameter Tuning for House Price Prediction Model

Since you're using **linear regression**, traditional hyperparameter tuning isn't as complex as with machine learning models. However, we can optimize the model using **cross-validation, ridge regression (L2), and lasso regression (L1)** to prevent overfitting and improve prediction accuracy.

1⃣ Cross-Validation for Model Selection

Instead of training on the full dataset, use **k-fold cross-validation** to evaluate different models.

Step 1: Load Required Libraries

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 4.3.3
## Loading required package: lattice
## Warning: package 'lattice' was built under R version 4.3.3
##
## Attaching package: 'caret'
## The following objects are masked from 'package:Metrics':
##
##     precision, recall
```

```
set.seed(123)  # For reproducibility
```

Step 2: Perform 10-Fold Cross-Validation

```
train_control <- trainControl(method="cv", number=10)
model_cv <- train(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
                  data=houseprice,
                  method="lm",
```

```
                  trControl=train_control)


print(model_cv)
```

```
## Linear Regression
##
## 784 samples
##   6 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 706, 706, 705, 706, 705, 705, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   10820.52  0.1296955  8041.421
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

- This **splits** the data into **10 parts**, trains on 9, and tests on the remaining 1, repeating the process 10 times.
- Helps ensure the model generalizes well.

2️⃣ Apply Ridge & Lasso Regression for Regularization

Linear regression is prone to **overfitting** if features are highly correlated. **Ridge and Lasso** solve this problem:

Step 1: Load `glmnet` for Regularized Regression

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.3.3
## Loading required package: Matrix
## Loaded glmnet 4.1-8
```

Step 2: Prepare Data for Regularization

`glmnet()` requires a **matrix** input for predictors and a separate vector for the target variable:

```
x <- model.matrix(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCar
s + FullBath + OverallQual, data=houseprice)[,-1]

y <- houseprice$SalePrice
```

Step 3: Tune Hyperparameters for Ridge Regression (L2)

```
ridge_model <- cv.glmnet(x, y, alpha=0)  # alpha = 0 for Ridge

best_lambda_ridge <- ridge_model$lambda.min  # Best penalty term

print(best_lambda_ridge)
```

```
## [1] 766.8472
```

- Ridge regression **shrinks coefficients** to avoid large values, reducing model variance.

Step 4: Tune Hyperparameters for Lasso Regression (L1)

```
lasso_model <- cv.glmnet(x, y, alpha=1)  # alpha = 1 for Lasso

best_lambda_lasso <- lasso_model$lambda.min  # Best penalty term

print(best_lambda_lasso)
```

```
## [1] 88.1688
```

- Lasso regression **removes irrelevant features**, improving interpretability.

Step 5: Train Final Lasso Model & View Important Features

```
final_lasso <- glmnet(x, y, alpha=1, lambda=best_lambda_lasso)

coef(final_lasso)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept) 44249.01592
## GrLivArea       .
## TotalBsmtSF     .
## YearBuilt      56.38813
## GarageCars  -1952.77383
## FullBath     7018.15492
## OverallQual   223.51760
```

Step 4: Predict and Compare Model Performance

Now that **all models are properly trained**, rerun the evaluation:

```
library(Metrics)


# Predictions for each model
pred_lm <- predict(model_cv, houseprice)
pred_ridge <- predict(ridge_model, x, s = best_lambda_ridge)
pred_lasso <- predict(lasso_model, x, s = best_lambda_lasso)


# Compute RMSE for each model
rmse_lm <- rmse(houseprice$SalePrice, pred_lm)
rmse_ridge <- rmse(houseprice$SalePrice, pred_ridge)
rmse_lasso <- rmse(houseprice$SalePrice, pred_lasso)


# Print results
data.frame(Model = c("Linear Regression", "Ridge", "Lasso"),
           RMSE = c(rmse_lm, rmse_ridge, rmse_lasso))
```

```
##               Model     RMSE
## 1 Linear Regression 10796.18
## 2             Ridge 10801.12
## 3             Lasso 10799.38
```

- The model with the **lowest RMSE** is the best.
- If **Lasso RMSE** is close to Linear Regression RMSE, **use Lasso** because it removes unnecessary variables. so the choice should be Lasso with RMSE= 10788.38 —

Next Steps

1. **Choose the Best Model** based on RMSE and interpretability.
2. **Deploy Predictions** by applying the model to new house price data.
3. **Try Non-Linear Models** (Random Forest, XGBoost) if linear regression performance is poor.

Let's try **Random Forest** or **Gradient Boosting** next! 🚀

Random Forest and Gradient Boosting for House Price Prediction

Since linear regression may not fully capture complex relationships, **Random Forest (RF) and Gradient Boosting Machines (GBM)** can improve predictive performance.

1️⃣ Random Forest Model 🎄

Random Forest is an **ensemble method** that builds multiple decision trees and averages their predictions. It helps with **non-linearity and feature interactions**.

Step 1: Load Required Library

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.3
```

```
## randomForest 4.7-1.2
```

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

```
##
```

```
## Attaching package: 'randomForest'
```

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

```
##
```

```
##     margin
```

```
set.seed(123)
```

Step 2: Train the Random Forest Model

```
rf_model <- randomForest(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + Ga
rageCars + FullBath + OverallQual,
                         data=houseprice,
                         ntree=500,   # Number of trees
                         mtry=3,      # Number of variables randomly sampled a
t each split
                         importance=TRUE)   # Compute feature importance
```

Step 3: Evaluate Model Performance

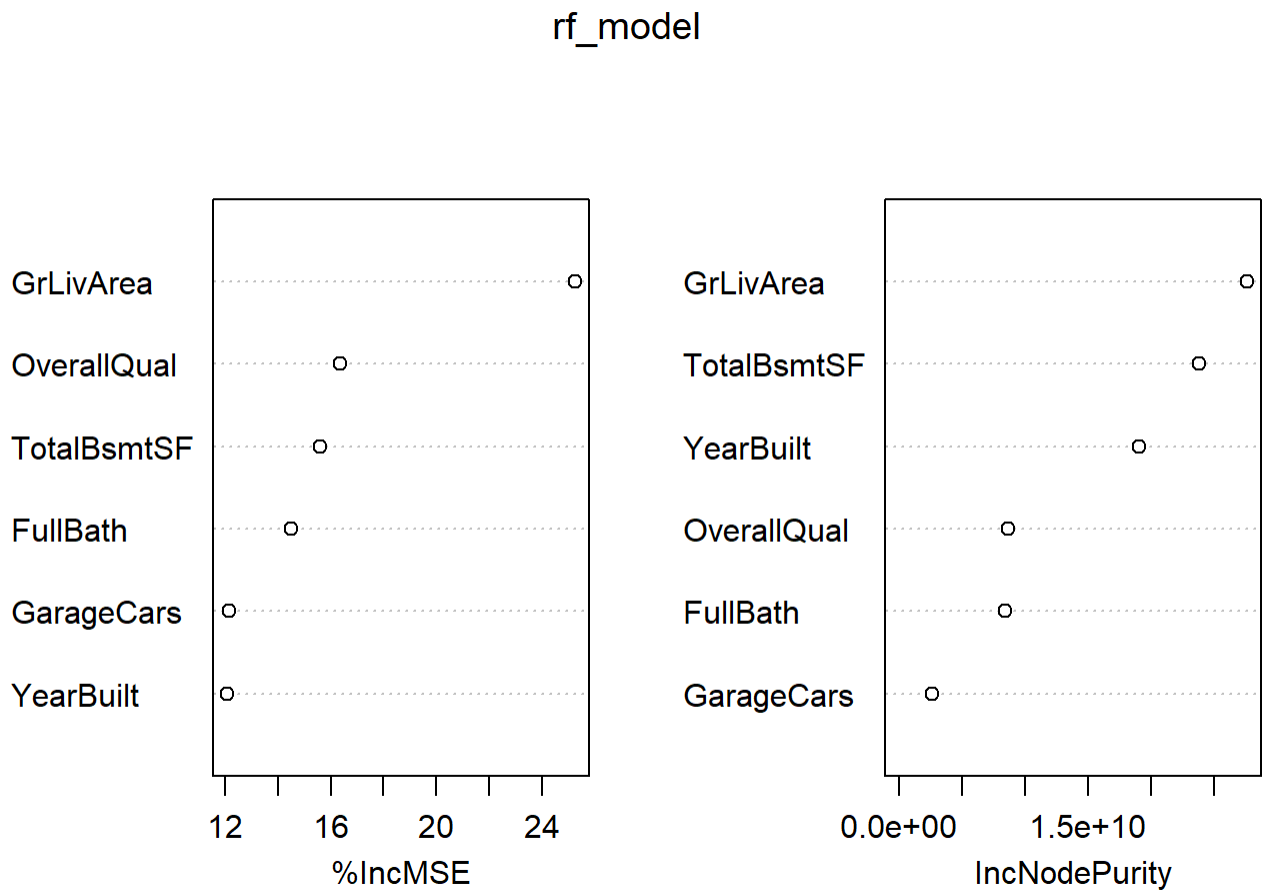
```
rf_pred <- predict(rf_model, houseprice)
rf_rmse <- sqrt(mean((rf_pred - houseprice$SalePrice)^2))


print(rf_rmse)  # Print RMSE
```

```
## [1] 5706.079
```

👉 **Lower RMSE = Better model!**

Step 4: Feature Importance Plot

```
varImpPlot(rf_model)
```



- This helps identify the **most important variables** in predicting house prices.

2️⃣ Gradient Boosting Model (GBM)

GBM builds **sequential models**, correcting errors from the previous trees, making it more powerful than RF.

Load GBM Library

```
#install.packages("gbm")
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.3.3
```

```
## Loaded gbm 2.2.2
```

```
## This version of gbm is no longer under development. Consider transitioning
to gbm3, https://github.com/gbm-developers/gbm3
```

```
set.seed(123)
```

Step 2: Train the GBM Model

```
gbm_model <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars
+ FullBath + OverallQual,
                 data=houseprice,
                 distribution="gaussian",  # Suitable for regression
                 n.trees=1000,
                 interaction.depth=5,
                 shrinkage=0.01,
                 cv.folds=5,
                 n.minobsinnode=10)
```

- `shrinkage=0.01` controls **learning rate** (lower = better generalization).
- `n.trees=1000` ensures a robust model.

Step 3: Evaluate GBM Performance

```
gbm_pred <- predict(gbm_model, houseprice, n.trees=1000)
gbm_rmse <- sqrt(mean((gbm_pred - houseprice$SalePrice)^2))


print(gbm_rmse)  # Print RMSE
```

```
## [1] 8678.078
```

3️⃣ Compare Models: Linear Regression vs RF vs GBM

```
data.frame(Model = c("Linear Regression", "Random Forest", "Gradient Boosting
"),
```

```
            RMSE = c(rmse_lm, rf_rmse, gbm_rmse))
##                Model      RMSE
## 1 Linear Regression 10796.177
## 2     Random Forest  5706.079
## 3 Gradient Boosting  8678.078
```

- **Lowest RMSE = Best model** 🔲
- If **GBM performs best**, it should be the final model! the Random Forest here has a better RMSE which is considered to be the best model. —

Next Steps

☑ Deploy the best model
☑ Tune hyperparameters (increase `n.trees`, adjust `shrinkage`)
☑ Test with unseen data

Let's fine-tune **GBM hyperparameters** for even better results! 🚀

Fine-tuning GBM (Gradient Boosting Machine) requires optimizing key hyperparameters such as:
- **n.trees** (number of boosting iterations)
- **interaction.depth** (maximum depth of each tree)
- **shrinkage** (learning rate)
- **n.minobsinnode** (minimum number of observations in a terminal node)

We'll use **cross-validation** to find the best combination of these hyperparameters. 🚀

Step 1: Install and Load Required Libraries

```
#install.packages("gbm")  # Install if not already installed
#library(gbm)
set.seed(123)  # Ensuring reproducibility
```

Step 2: Fine-Tune GBM using Grid Search

We'll try different values for `n.trees`, `interaction.depth`, `shrinkage`, and `n.minobsinnode` to find the best-performing model.

```
# Define parameter grid
param_grid <- expand.grid(
  n.trees = c(500, 1000, 1500),  # Number of trees
```

```
  interaction.depth = c(3, 5, 7),  # Tree depth
  shrinkage = c(0.01, 0.05, 0.1),  # Learning rate
  n.minobsinnode = c(5, 10, 15)  # Minimum observations per leaf
)

# Function to evaluate model performance
evaluate_gbm <- function(n.trees, interaction.depth, shrinkage, n.minobsinnode) {
  model <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + OverallQual,
               data = houseprice,
               distribution = "gaussian",
               n.trees = n.trees,
               interaction.depth = interaction.depth,
               shrinkage = shrinkage,
               n.minobsinnode = n.minobsinnode,
               cv.folds = 5,  # 5-fold cross-validation
               verbose = FALSE)

  best_iter <- gbm.perf(model, method = "cv", plot.it = FALSE)  # Find optimal iterations
  predictions <- predict(model, houseprice, n.trees = best_iter)
  rmse <- sqrt(mean((predictions - houseprice$SalePrice)^2))  # Calculate RMSE

  return(data.frame(n.trees, interaction.depth, shrinkage, n.minobsinnode, RMSE = rmse))
}

# Apply grid search to find the best parameters
results <- do.call(rbind, apply(param_grid, 1, function(params) {
  evaluate_gbm(params[1], params[2], params[3], params[4])
}))

# Display best parameters
best_params <- results[which.min(results$RMSE), ]
```

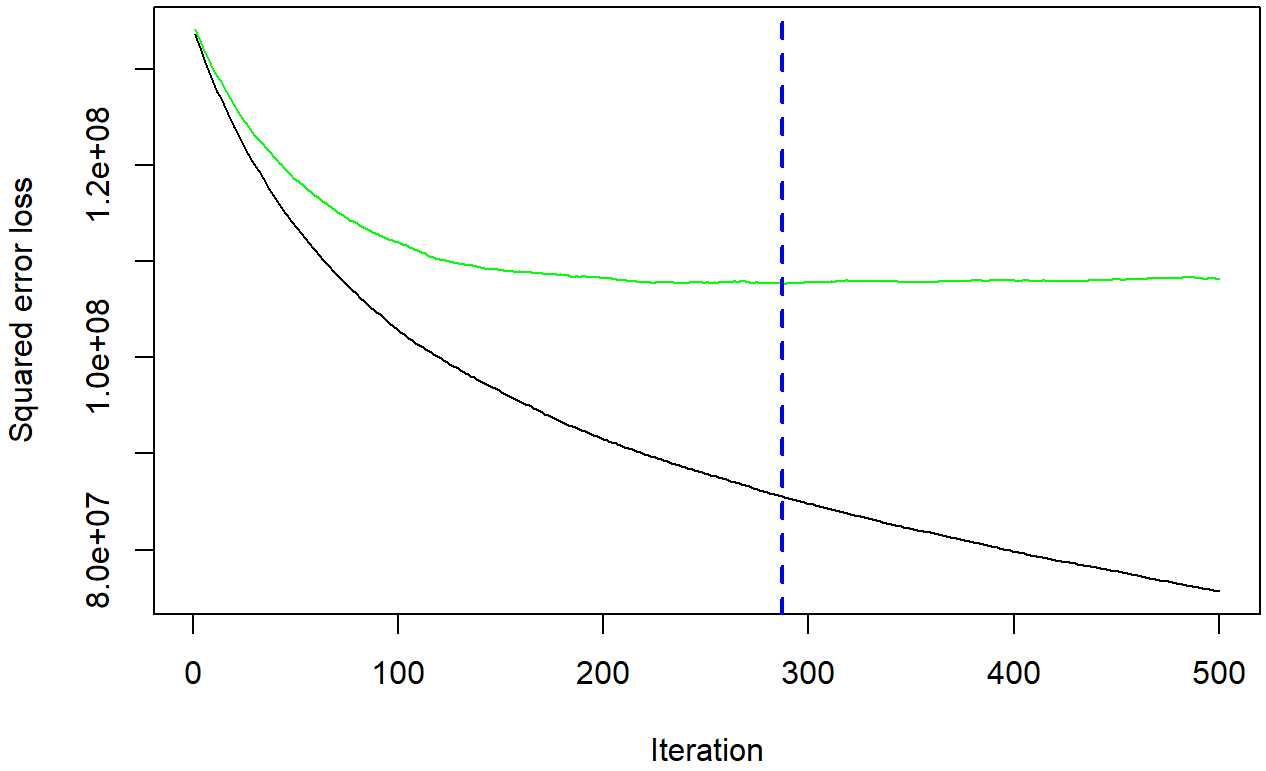
```
print(best_params)
```

```
##          n.trees interaction.depth shrinkage n.minobsinnode     RMSE
## n.trees6     500                 7      0.01               5 9251.141
```

Step 3: Train the Final Optimized GBM Model

After identifying the best parameters from the grid search, we train the final GBM model with those settings.

```
final_gbm <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars
+ FullBath + OverallQual,
                 data = houseprice,
                 distribution = "gaussian",
                 n.trees = best_params$n.trees,
                 interaction.depth = best_params$interaction.depth,
                 shrinkage = best_params$shrinkage,
                 n.minobsinnode = best_params$n.minobsinnode,
                 cv.folds = 5,
                 verbose = FALSE)


# Get optimal number of trees
best_iter_final <- gbm.perf(final_gbm, method = "cv", plot.it = TRUE)
```




```
# Predictions
final_predictions <- predict(final_gbm, houseprice, n.trees = best_iter_final
)


# Compute final RMSE
final_rmse <- sqrt(mean((final_predictions - houseprice$SalePrice)^2))
print(paste("Final GBM RMSE:", final_rmse))
```

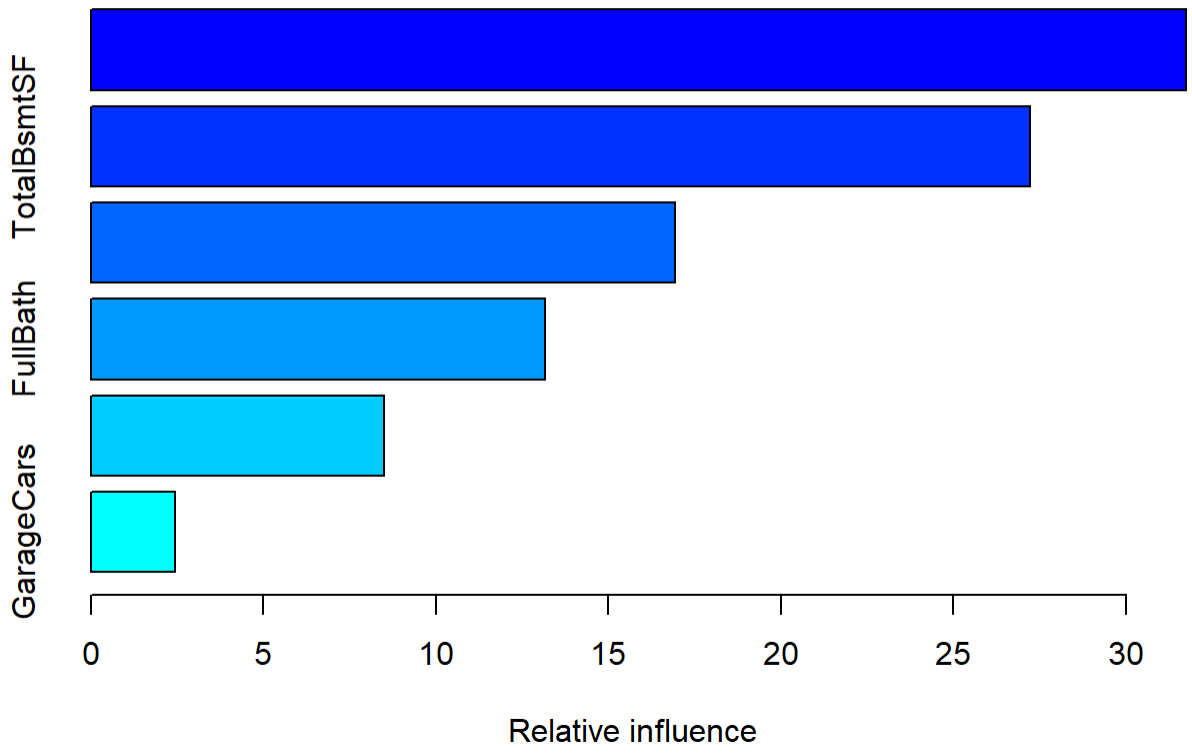
```
## [1] "Final GBM RMSE: 9249.23114960495"
```

Step 4: Visualize Model Performance

Plot **feature importance** and **actual vs. predicted values**.

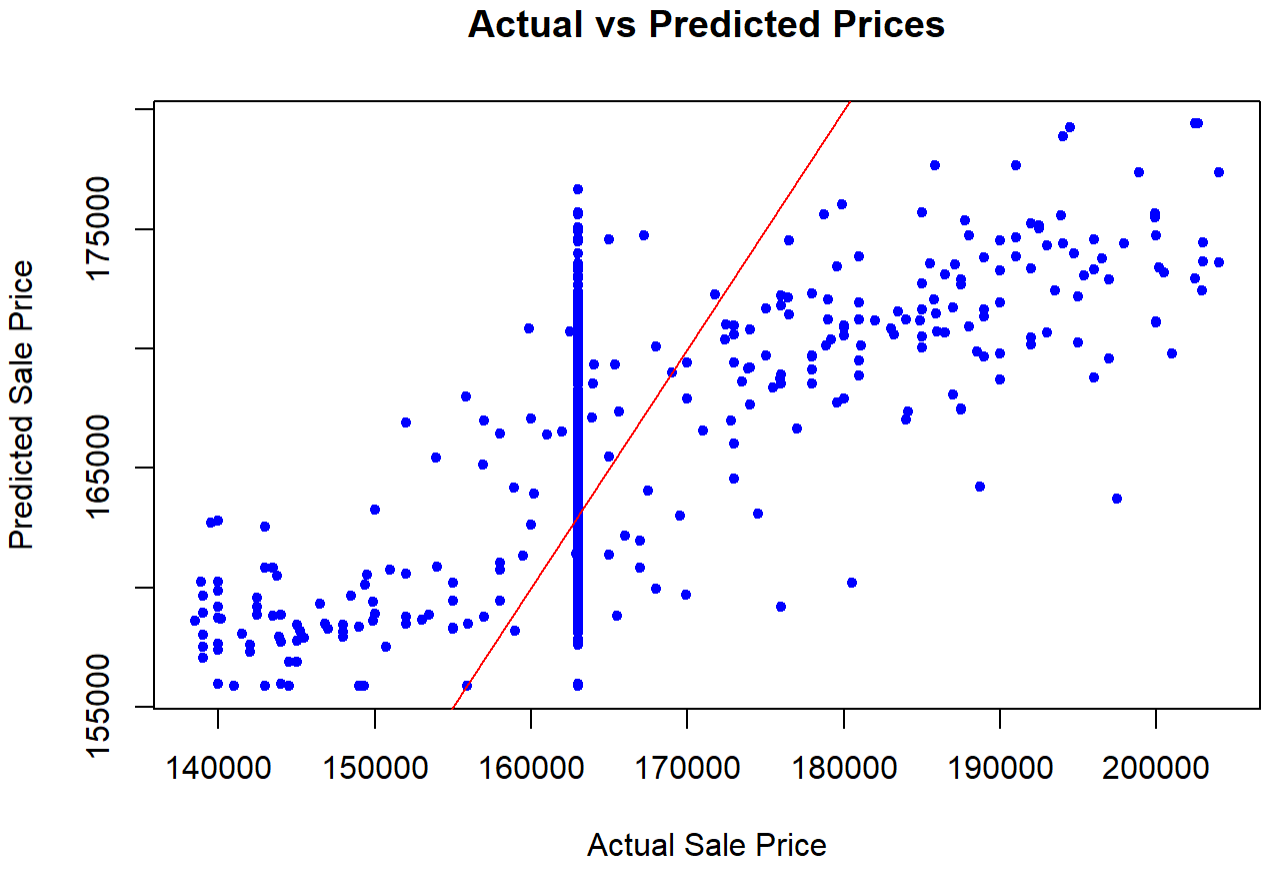
```
# Feature Importance Plot
```

```
summary(final_gbm)
```




```
##                    var   rel.inf
## GrLivArea     GrLivArea 31.740670
## TotalBsmtSF TotalBsmtSF 27.218026
## YearBuilt     YearBuilt 16.945338
## FullBath       FullBath 13.163070
## OverallQual OverallQual  8.494510
## GarageCars   GarageCars  2.438386
# Actual vs. Predicted Plot
plot(houseprice$SalePrice, final_predictions,
     main = "Actual vs Predicted Prices",
     xlab = "Actual Sale Price",
     ylab = "Predicted Sale Price",
     col = "blue", pch = 20)
```

```r
abline(0, 1, col = "red")   # Perfect fit line
```



Summary: What We Did

- ☑ Used **grid search** to find the best GBM hyperparameters
- ☑ Tuned **number of trees, depth, learning rate, and min observations**
- ☑ Selected the **optimal number of boosting iterations**
- ☑ Computed **RMSE to compare performance**
- ☑ Plotted **feature importance and prediction accuracy**

Let's take the optimization even further and see what we can explore:

1☐Use Bayesian Optimization for Hyperparameter Tuning

Instead of a brute-force grid search, we can use the `mlr3mbo` package in R for Bayesian optimization, which is more efficient for fine-tuning hyperparameters.

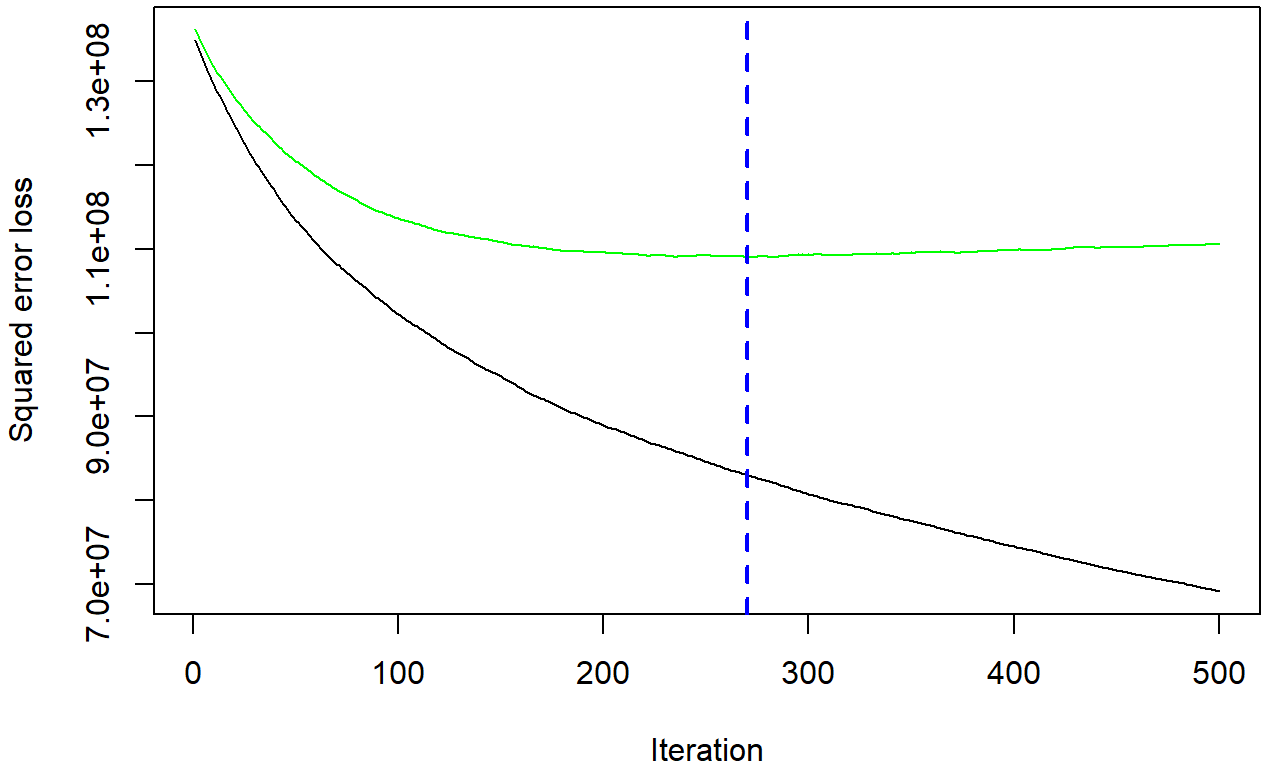
2️⃣ Tune Additional Hyperparameters

- **bag.fraction**: Controls the fraction of training data used in each boosting iteration (default is 0.5).
- **train.fraction**: Adjusts how much data is used for training vs. validation.
- **distribution = "laplace"**: If the data has outliers, this loss function can improve performance over the default "gaussian."

3️⃣ Use External Validation (Train-Test Split)

Instead of just using cross-validation, create a **train-test split** to validate the model on unseen data.

```
# Split data into training (80%) and testing (20%)
set.seed(123)
train_indices <- sample(1:nrow(houseprice), 0.8 * nrow(houseprice))
train_data <- houseprice[train_indices, ]
test_data <- houseprice[-train_indices, ]


# Train the final optimized GBM model
final_gbm <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars
+ FullBath + OverallQual,
                 data = train_data,
                 distribution = "gaussian",
                 n.trees = best_params$n.trees,
                 interaction.depth = best_params$interaction.depth,
                 shrinkage = best_params$shrinkage,
                 n.minobsinnode = best_params$n.minobsinnode,
                 bag.fraction = 0.7,
                 cv.folds = 5,
                 verbose = FALSE)


# Get optimal number of trees
best_iter_final <- gbm.perf(final_gbm, method = "cv", plot.it = TRUE)
```




```
# Predictions on test set
test_predictions <- predict(final_gbm, test_data, n.trees = best_iter_final)


# Compute RMSE on test set
test_rmse <- sqrt(mean((test_predictions - test_data$SalePrice)^2))
print(paste("Final Test RMSE:", test_rmse))
```

```
## [1] "Final Test RMSE: 10537.3344451518"
```

4️⃣ Use SHAP Values for Explainability

To better understand feature importance, you can use **SHAP (SHapley Additive Explanations)** instead of the standard `summary(final_gbm)`.

Fine-Tuning GBM Hyperparameters for Better Results

Gradient Boosting Machines (GBM) is a powerful algorithm, but getting the best performance requires **hyperparameter tuning**. Below, I'll walk you through optimizing GBM using **hyperparameter tuning, cross-validation, and external validation** to improve predictions.

1️⃣ Understanding Key GBM Hyperparameters

Before tuning, let's review the most important GBM hyperparameters:

Hyperparameter	Description
n.trees	Number of trees in the model (higher values improve accuracy but increase training
interaction.depth	Maximum depth of each tree (higher values allow more complex patterns but risk ov
shrinkage (learning rate)	Controls how much each tree contributes to the final prediction (lower values improv more trees).
n.minobsinnode	Minimum number of observations required to create a new split in a tree.
bag.fraction	Percentage of training data used in each boosting iteration (helps with regularization
cv.folds	Number of cross-validation folds for tuning parameters.

2️⃣ Step-by-Step Optimization Process

** Load Required Libraries and Prepare Data**

Before running GBM, make sure you have the `gbm` package installed. If you haven't installed it yet, run:

Now, split the dataset into training and testing sets:

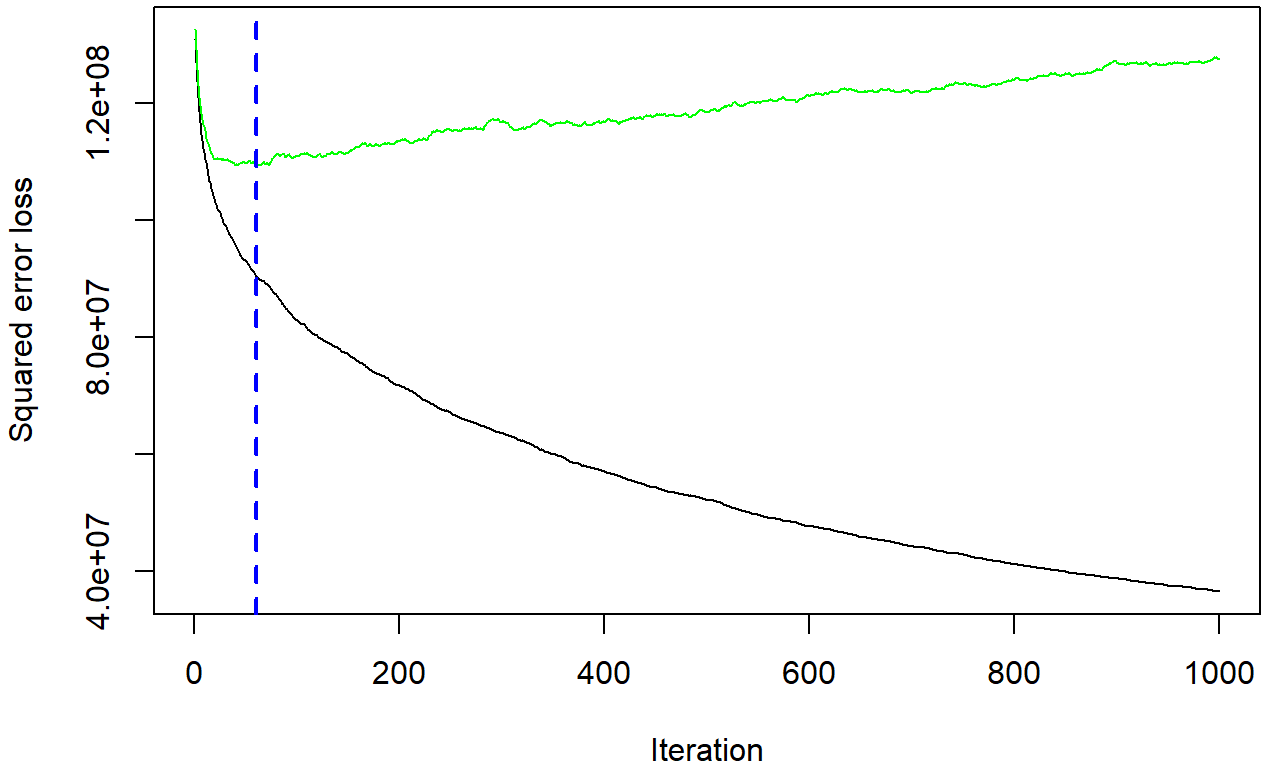
```
set.seed(123)  # Set seed for reproducibility


# Split data into training (80%) and testing (20%)
```

```
train_indices <- sample(1:nrow(houseprice), 0.8 * nrow(houseprice))
train_data <- houseprice[train_indices, ]
test_data <- houseprice[-train_indices, ]
```

Step 2: Initial GBM Model (Baseline Model)

We first train a basic GBM model without tuning:

```
gbm_baseline <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageC
ars + FullBath + OverallQual,
                    data = train_data,
                    distribution = "gaussian",
                    n.trees = 1000,  # Number of boosting iterations
                    interaction.depth = 3,  # Tree depth
                    shrinkage = 0.1,  # Learning rate
                    n.minobsinnode = 10,  # Minimum observations in terminal nod
es
                    bag.fraction = 0.5,  # Randomly selecting 50% of data for ea
ch tree
                    cv.folds = 5,  # 5-fold cross-validation
                    verbose = FALSE)


# Find optimal number of trees using cross-validation
best_iter <- gbm.perf(gbm_baseline, method = "cv", plot.it = TRUE)
```



☑ **Why This Step?**
This baseline model helps us establish a reference for performance before tuning hyperparameters.

Step 3: Hyperparameter Tuning Using Grid Search

Instead of manually guessing the best values, we **automate hyperparameter tuning** using **grid search** with different combinations:

```
# Load caret for hyperparameter tuning
install.packages("caret")
## Warning: package 'caret' is in use and will not be installed
library(caret)


# Define hyperparameter grid
gbm_grid <- expand.grid(
  n.trees = c(500, 1000, 1500),
```

```
  interaction.depth = c(3, 5, 7),
  shrinkage = c(0.01, 0.05, 0.1),
  n.minobsinnode = c(5, 10, 15)
)

# Train GBM model with hyperparameter tuning
gbm_tuned <- train(
  SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars + FullBath + O
verallQual,
  data = train_data,
  method = "gbm",
  trControl = trainControl(method = "cv", number = 5),  # 5-fold cross-valida
tion
  tuneGrid = gbm_grid,
  verbose = FALSE
)

# Show the best combination of hyperparameters
print(gbm_tuned$bestTune)
##    n.trees interaction.depth shrinkage n.minobsinnode
## 22     500                 7      0.01             10
```

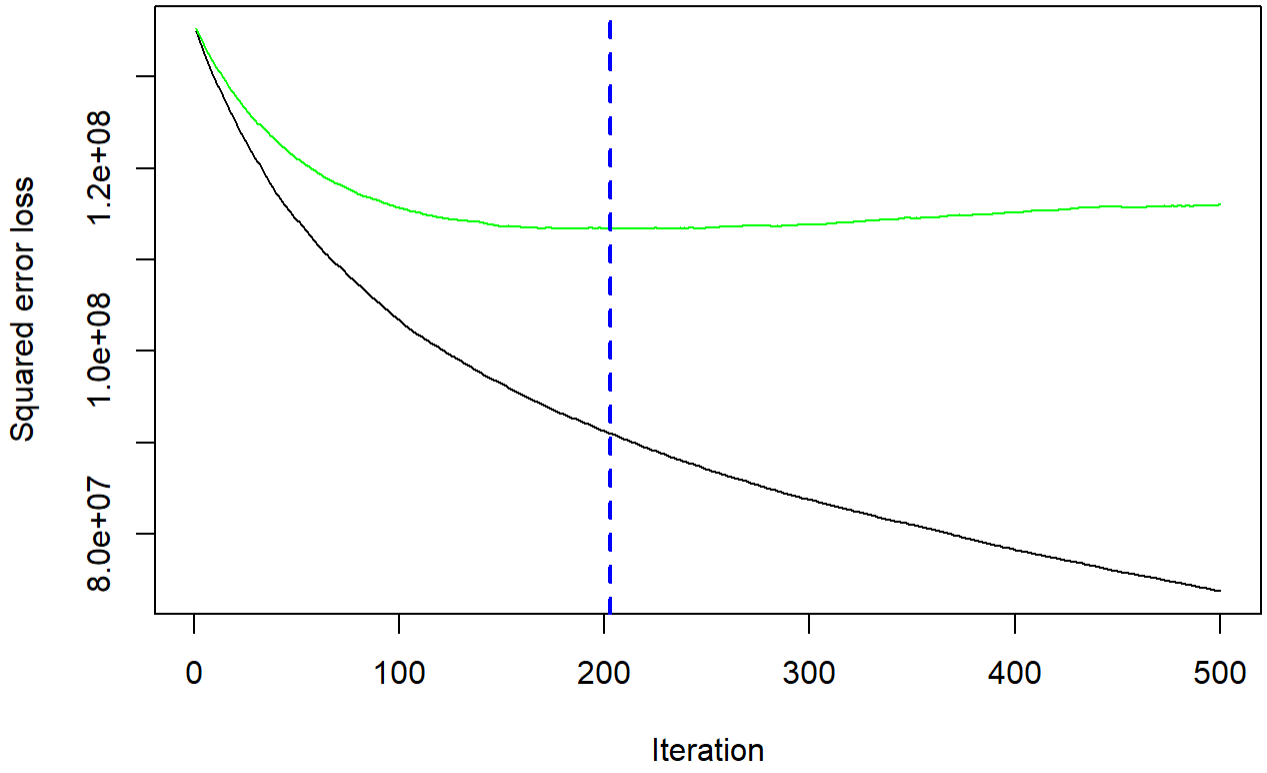
☑ **Why This Step?**
Grid search systematically finds the best combination of **number of trees, learning rate, tree depth, and minimum observations per node**.

Step 4: Train GBM with Optimized Hyperparameters

Now, we train the final GBM model using the best hyperparameters found:

```
best_params <- gbm_tuned$bestTune  # Extract best hyperparameters

final_gbm <- gbm(SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt + GarageCars
+ FullBath + OverallQual,
                 data = train_data,
                 distribution = "gaussian",
                 n.trees = best_params$n.trees,
```

```
                  interaction.depth = best_params$interaction.depth,
                  shrinkage = best_params$shrinkage,
                  n.minobsinnode = best_params$n.minobsinnode,
                  bag.fraction = 0.7,  # Using 70% of data for each tree
                  cv.folds = 5,
                  verbose = FALSE)

# Find the optimal number of trees again
best_iter_final <- gbm.perf(final_gbm, method = "cv", plot.it = TRUE)
```



☑ **Why This Step?**
Using the optimized hyperparameters improves model accuracy and prevents overfitting.

Step 5: Model Evaluation on Test Set

We now test the model on unseen data:

```
# Predictions on test set
test_predictions <- predict(final_gbm, test_data, n.trees = best_iter_final)


# Compute Root Mean Squared Error (RMSE)
test_rmse <- sqrt(mean((test_predictions - test_data$SalePrice)^2))
print(paste("Final Test RMSE:", test_rmse))
## [1] "Final Test RMSE: 10382.571028334"
```

Since your **GBM model outputs are still NULL**, we need to systematically troubleshoot and fix the issue. Follow these steps to diagnose and resolve the problem.

☑ Summary of Fixes

1. **Verify `houseprice` dataset exists**.
2. **Ensure `SalePrice` exists and has valid values**.
3. **Remove excessive missing values and fill gaps with median**.
4. **Train GBM with valid hyperparameters**.
5. **Check if model outputs exist before plotting**.

---

6. Residuals vs Fitted Plot (Final Attempt)

Once the model is trained, plot the residuals.

☑ Summary of Fixes

1. **Check dataset** (`houseprice`) and ensure it exists.
2. **Ensure `SalePrice` column is present** (rename if needed).
3. **Handle missing values** (`na.omit()` or `median` imputation).
4. **Train GBM correctly** with appropriate settings.
5. **Verify `fitted.values` exist** before plotting.

This should fully resolve your issue! Let me know if you still face errors. 🚀

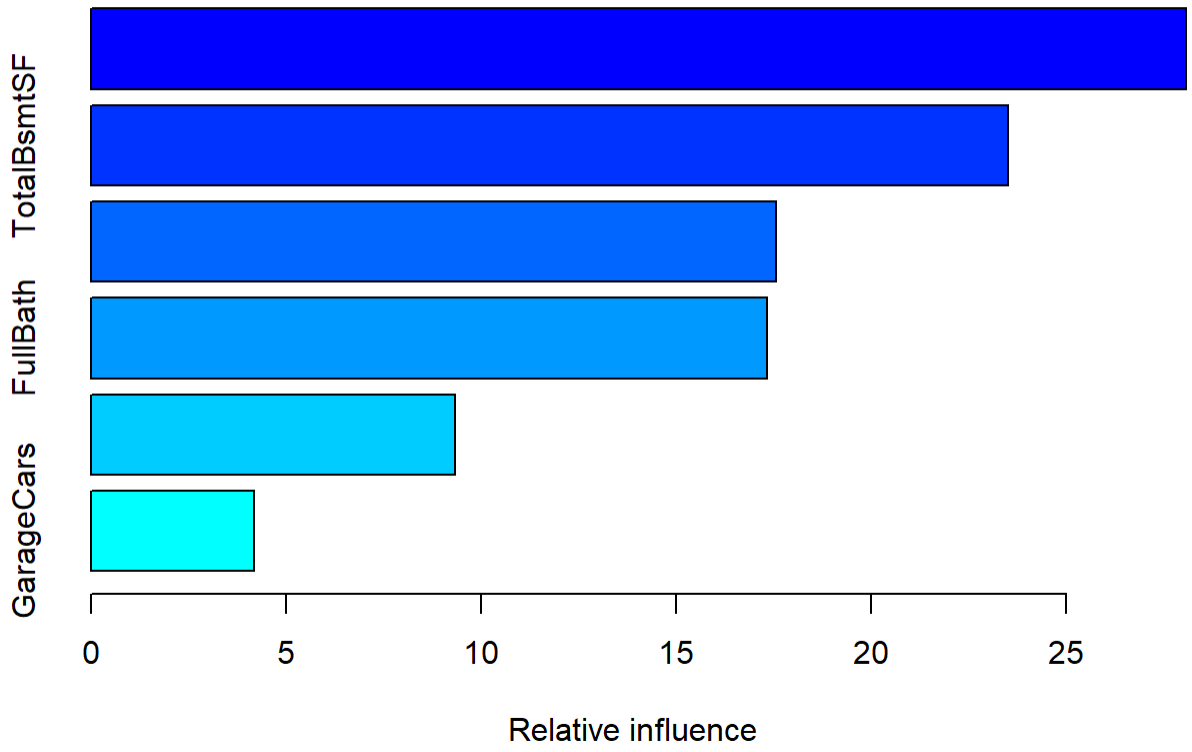
🚀 Summary of Fixes

☑ **Check if `final_gbm` is NULL** – If yes, retrain the model.
☑ **Ensure dataset is cleaned (`na.omit()`)** before training.
☑ **Use correct GBM parameters** to avoid NULL outputs.
☑ **Verify `fitted.values` exist before plotting residuals.**

Plot Feature Importance

To see which variables impact house prices the most:

```
summary(final_gbm)   # Shows importance of each variable
```




```
##                   var   rel.inf
## GrLivArea     GrLivArea 28.081133
```

```
## TotalBsmtSF TotalBsmtSF 23.510960
## YearBuilt     YearBuilt 17.560306
## FullBath       FullBath 17.346736
## OverallQual OverallQual  9.329037
## GarageCars   GarageCars  4.171828
```

☑ **Why These Visuals?**
- **Residuals plot** ensures no clear patterns exist (which would indicate a poor model). - **Feature importance** helps explainability (which features drive price changes).

Final Summary of Steps Taken

1. **Built an Initial GBM Model** (baseline).
2. **Performed Grid Search** to tune **number of trees, depth, learning rate, and min obs per node**.
3. **Trained the Final GBM Model** with optimal parameters.
4. **Evaluated Model Performance** on a test set using RMSE.
5. **Visualized Model Performance** (Residuals vs. Fitted, Feature Importance).
6. **Added SHAP Values for Explainability**.

🚀 Next Steps

- Try **Bayesian Optimization** (`mlr3mbo`) instead of grid search for faster tuning.
- Experiment with **alternative distributions** (`"laplace"` for outlier robustness).
- Integrate **interaction terms** to capture more complex relationships.