House Price Prediction

2024-04-09

About the Data

The data was collected from the website 'Kaggle', which contains thousands of datasets used for training predictive algorithms. The specific dataset we are using contains training and testing data fortunately named "train" and "test", which contains multiple different variables such as area, year built, number of rooms, and more used to predict the price of a house.

Link to dataset on Kaggle: https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data

Loading Packages

```
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 4.2.3

## Warning: package 'ggplot2' was built under R version 4.2.3

## Warning: package 'tibble' was built under R version 4.2.3

## Warning: package 'tidyr' was built under R version 4.2.3

## Warning: package 'readr' was built under R version 4.2.3

## Warning: package 'purrr' was built under R version 4.2.3

## Warning: package 'dplyr' was built under R version 4.2.3

## Warning: package 'stringr' was built under R version 4.2.3

## Warning: package 'forcats' was built under R version 4.2.3

## Warning: package 'lubridate' was built under R version 4.2.3

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ──────────────────────────────────────────
```

```
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors

library(corrplot)

## Warning: package 'corrplot' was built under R version 4.2.3

## corrplot 0.92 loaded

library(lubridate)
library(ggplot2)
library(readr)
library(caTools)

## Warning: package 'caTools' was built under R version 4.2.3

library(GGally)

## Warning: package 'GGally' was built under R version 4.2.3

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(caret)

## Warning: package 'caret' was built under R version 4.2.3

## Loading required package: lattice

## Warning: package 'lattice' was built under R version 4.2.3

##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift

library(leaps)

## Warning: package 'leaps' was built under R version 4.2.3

library(gridExtra)

## Warning: package 'gridExtra' was built under R version 4.2.3

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

```
##
##     combine
```

Reading the data and understanding it

```
train_data = read.csv("train.csv")
test_dataa = read.csv("test.csv")
```

Converting all character columns to factor:

```
train_data <- as.data.frame(unclass(train_data), stringsAsFactors = TRUE)
test_dataa <- as.data.frame(unclass(test_dataa), stringsAsFactors = TRUE)
```

Now, lets view the first row of the training set:

```
head(train_data,1)

##   Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape
LandContour
## 1  1         60       RL          65    8450   Pave  <NA>      Reg
Lvl
##   Utilities LotConfig LandSlope Neighborhood Condition1 Condition2
BldgType
## 1    AllPub    Inside       Gtl      CollgCr       Norm       Norm
1Fam
##   HouseStyle OverallQual OverallCond YearBuilt YearRemodAdd RoofStyle
RoofMatl
## 1     2Story           7           5      2003         2003     Gable
CompShg
##   Exterior1st Exterior2nd MasVnrType MasVnrArea ExterQual ExterCond
Foundation
## 1     VinylSd     VinylSd    BrkFace        196        Gd        TA
PConc
##   BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinSF1 BsmtFinType2
## 1       Gd       TA           No          GLQ        706          Unf
##   BsmtFinSF2 BsmtUnfSF TotalBsmtSF Heating HeatingQC CentralAir Electrical
## 1          0       150         856    GasA        Ex          Y      SBrkr
##   X1stFlrSF X2ndFlrSF LowQualFinSF GrLivArea BsmtFullBath BsmtHalfBath
FullBath
## 1       856       854            0      1710            1            0
2
##   HalfBath BedroomAbvGr KitchenAbvGr KitchenQual TotRmsAbvGrd Functional
## 1        1            3            1          Gd            8        Typ
##   Fireplaces FireplaceQu GarageType GarageYrBlt GarageFinish GarageCars
## 1          0        <NA>     Attchd        2003          RFn          2
##   GarageArea GarageQual GarageCond PavedDrive WoodDeckSF OpenPorchSF
## 1        548         TA         TA          Y          0          61
##   EnclosedPorch X3SsnPorch ScreenPorch PoolArea PoolQC Fence MiscFeature
## 1             0          0           0        0   <NA>  <NA>        <NA>
##   MiscVal MoSold YrSold SaleType SaleCondition SalePrice
## 1       0      2   2008       WD        Normal    208500
```

From our first row of data, we can already see some columns such as "Alley" have missing values classified as NA. We also notice the column "ID" is essentially useless because it does not provide any descriptions to what the house may be like, it is essentially an identity column.

```
str(train_data)

## 'data.frame':    1460 obs. of  81 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : Factor w/ 5 levels "C (all)","FV",..: 4 4 4 4 4 4 4 4 5
4 ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120
7420 ...
##  $ Street       : Factor w/ 2 levels "Grvl","Pave": 2 2 2 2 2 2 2 2 2 2
...
##  $ Alley        : Factor w/ 2 levels "Grvl","Pave": NA NA NA NA NA NA NA
NA NA NA ...
##  $ LotShape     : Factor w/ 4 levels "IR1","IR2","IR3",..: 4 4 1 1 1 1 4 1
4 4 ...
##  $ LandContour  : Factor w/ 4 levels "Bnk","HLS","Low",..: 4 4 4 4 4 4 4 4
4 4 ...
##  $ Utilities    : Factor w/ 2 levels "AllPub","NoSeWa": 1 1 1 1 1 1 1 1 1
1 ...
##  $ LotConfig    : Factor w/ 5 levels "Corner","CulDSac",..: 5 3 5 1 3 5 5
1 5 1 ...
##  $ LandSlope    : Factor w/ 3 levels "Gtl","Mod","Sev": 1 1 1 1 1 1 1 1 1
1 ...
##  $ Neighborhood : Factor w/ 25 levels "Blmngtn","Blueste",..: 6 25 6 7 14
12 21 17 18 4 ...
##  $ Condition1   : Factor w/ 9 levels "Artery","Feedr",..: 3 2 3 3 3 3 3 5
1 1 ...
##  $ Condition2   : Factor w/ 8 levels "Artery","Feedr",..: 3 3 3 3 3 3 3 3
3 1 ...
##  $ BldgType     : Factor w/ 5 levels "1Fam","2fmCon",..: 1 1 1 1 1 1 1 1 1
2 ...
##  $ HouseStyle   : Factor w/ 8 levels "1.5Fin","1.5Unf",..: 6 3 6 6 6 1 3 6
1 2 ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939
...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950
...
##  $ RoofStyle    : Factor w/ 6 levels "Flat","Gable",..: 2 2 2 2 2 2 2 2 2
2 ...
##  $ RoofMatl     : Factor w/ 8 levels "ClyTile","CompShg",..: 2 2 2 2 2 2 2
2 2 2 ...
##  $ Exterior1st  : Factor w/ 15 levels "AsbShng","AsphShn",..: 13 9 13 14
```

```
13 13 13 7 4 9 ...
##  $ Exterior2nd  : Factor w/ 16 levels "AsbShng","AsphShn",..: 14 9 14 16
14 14 14 7 16 9 ...
##  $ MasVnrType   : Factor w/ 4 levels "BrkCmn","BrkFace",..: 2 3 2 3 2 3 4
4 3 3 ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 4 3 4 3 4 4
4 ...
##  $ ExterCond    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5
5 ...
##  $ Foundation   : Factor w/ 6 levels "BrkTil","CBlock",..: 3 2 3 1 3 6 3 2
1 1 ...
##  $ BsmtQual     : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 3 3 4 3 3 1 3 4
4 ...
##  $ BsmtCond     : Factor w/ 4 levels "Fa","Gd","Po",..: 4 4 4 2 4 4 4 4 4
4 ...
##  $ BsmtExposure : Factor w/ 4 levels "Av","Gd","Mn",..: 4 2 3 4 1 4 1 3 4
4 ...
##  $ BsmtFinType1 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 3 1 3 1 3 3 3 1
6 3 ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 6 6 6 6 6 6 6 2
6 6 ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : Factor w/ 6 levels "Floor","GasA",..: 2 2 2 2 2 2 2 2 2
2 ...
##  $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",..: 1 1 1 3 1 1 1 1 3
1 ...
##  $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",..: 5 5 5 5 5 5 5 5 2
5 ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077
...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 3 3 4 3 4 4
4 ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : Factor w/ 7 levels "Maj1","Maj2",..: 7 7 7 7 7 7 7 7 3 7
...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
```

```
## $ FireplaceQu  : Factor w/ 5 levels "Ex","Fa","Gd",..: NA 5 5 3 5 NA 3 5
5 5 ...
## $ GarageType   : Factor w/ 6 levels "2Types","Attchd",..: 2 2 2 6 2 2 2 2
6 2 ...
## $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939
...
## $ GarageFinish : Factor w/ 3 levels "Fin","RFn","Unf": 2 2 2 3 2 3 2 2 3
2 ...
## $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
## $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
## $ GarageQual   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 2
3 ...
## $ GarageCond   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5
5 ...
## $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...
## $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
## $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
## $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
## $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
## $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
## $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ PoolQC       : Factor w/ 3 levels "Ex","Fa","Gd": NA NA NA NA NA NA NA
NA NA NA ...
## $ Fence        : Factor w/ 4 levels "GdPrv","GdWo",..: NA NA NA NA NA 3
NA NA NA NA ...
## $ MiscFeature  : Factor w/ 4 levels "Gar2","Othr",..: NA NA NA NA NA 3 NA
3 NA NA ...
## $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
## $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
## $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008
...
## $ SaleType     : Factor w/ 9 levels "COD","Con","ConLD",..: 9 9 9 9 9 9 9
9 9 9 ...
## $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",..: 5 5 5 1 5 5 5
5 1 5 ...
## $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000
200000 129900 118000 ...
```

We can see that our data consists of only integer and Factor columns, meaning that they are either whole numbers or some text description of the house.

```
summary(train_data)
```

```
##        Id           MSSubClass       MSZoning     LotFrontage
##  Min.   :   1.0   Min.   : 20.0   C (all):  10   Min.   : 21.00
##  1st Qu.: 365.8   1st Qu.: 20.0   FV     :  65   1st Qu.: 59.00
##  Median : 730.5   Median : 50.0   RH     :  16   Median : 69.00
##  Mean   : 730.5   Mean   : 56.9   RL     :1151   Mean   : 70.05
##  3rd Qu.:1095.2   3rd Qu.: 70.0   RM     : 218   3rd Qu.: 80.00
##  Max.   :1460.0   Max.   :190.0                  Max.   :313.00
```

```
##                                                   NA's   :259
##     LotArea        Street      Alley      LotShape  LandContour  Utilities
##  Min.   :  1300   Grvl:   6   Grvl:  50   IR1:484   Bnk:  63
AllPub:1459
##  1st Qu.:  7554   Pave:1454   Pave:  41   IR2: 41   HLS:  50    NoSeWa:
1
##  Median :  9478               NA's:1369   IR3: 10   Low:  36
##  Mean   : 10517                           Reg:925   Lvl:1311
##  3rd Qu.: 11602
##  Max.   :215245
##
##    LotConfig    LandSlope   Neighborhood   Condition1     Condition2
##  Corner : 263   Gtl:1382   NAmes  :225   Norm   :1260   Norm   :1445
##  CulDSac:  94   Mod:  65   CollgCr:150   Feedr  :  81   Feedr  :   6
##  FR2    :  47   Sev:  13   OldTown:113   Artery :  48   Artery :   2
##  FR3    :   4              Edwards:100   RRAn   :  26   PosN   :   2
##  Inside :1052              Somerst: 86   PosN   :  19   RRNn   :   2
##                            Gilbert: 79   RRAe   :  11   PosA   :   1
##                            (Other):707   (Other):  15   (Other):   2
##    BldgType      HouseStyle   OverallQual      OverallCond      YearBuilt
##  1Fam  :1220   1Story :726   Min.   : 1.000   Min.   :1.000   Min.   :1872
##  2fmCon:  31   2Story :445   1st Qu.: 5.000   1st Qu.:5.000   1st Qu.:1954
##  Duplex:  52   1.5Fin :154   Median : 6.000   Median :5.000   Median :1973
##  Twnhs :  43   SLvl   : 65   Mean   : 6.099   Mean   :5.575   Mean   :1971
##  TwnhsE: 114   SFoyer : 37   3rd Qu.: 7.000   3rd Qu.:6.000   3rd Qu.:2000
##                1.5Unf : 14   Max.   :10.000   Max.   :9.000   Max.   :2010
##                (Other): 19
##   YearRemodAdd    RoofStyle       RoofMatl     Exterior1st   Exterior2nd
##  Min.   :1950   Flat   :  13   CompShg:1434   VinylSd:515   VinylSd:504
##  1st Qu.:1967   Gable  :1141   Tar&Grv:  11   HdBoard:222   MetalSd:214
##  Median :1994   Gambrel:  11   WdShngl:   6   MetalSd:220   HdBoard:207
##  Mean   :1985   Hip    : 286   WdShake:   5   Wd Sdng:206   Wd Sdng:197
##  3rd Qu.:2004   Mansard:   7   ClyTile:   1   Plywood:108   Plywood:142
##  Max.   :2010   Shed   :   2   Membran:   1   CemntBd: 61   CmentBd: 60
##                                (Other):   2   (Other):128   (Other):136
##    MasVnrType    MasVnrArea     ExterQual ExterCond  Foundation  BsmtQual
##  BrkCmn : 15   Min.   :   0.0   Ex: 52     Ex:   3   BrkTil:146   Ex  :121
##  BrkFace:445   1st Qu.:   0.0   Fa: 14     Fa:  28   CBlock:634   Fa  : 35
##  None   :864   Median :   0.0   Gd:488     Gd: 146   PConc :647   Gd  :618
##  Stone  :128   Mean   : 103.7   TA:906     Po:   1   Slab  : 24   TA  :649
##  NA's   :  8   3rd Qu.: 166.0              TA:1282   Stone :  6   NA's: 37
##                Max.   :1600.0                        Wood  :  3
##                NA's   :8
##  BsmtCond    BsmtExposure BsmtFinType1   BsmtFinSF1     BsmtFinType2
##  Fa  :  45   Av  :221     ALQ :220     Min.   :   0.0   ALQ :  19
##  Gd  :  65   Gd  :134     BLQ :148     1st Qu.:   0.0   BLQ :  33
##  Po  :   2   Mn  :114     GLQ :418     Median : 383.5   GLQ :  14
##  TA  :1311   No  :953     LwQ : 74     Mean   : 443.6   LwQ :  46
##  NA's:  37   NA's: 38     Rec :133     3rd Qu.: 712.2   Rec :  54
##                           Unf :430     Max.   :5644.0   Unf :1256
```

```
##                            NA's: 37                       NA's:  38    
##    BsmtFinSF2        BsmtUnfSF       TotalBsmtSF      Heating      
HeatingQC
##  Min.   :   0.00   Min.   :   0.0   Min.   :   0.0   Floor:   1   Ex:741   
##  1st Qu.:   0.00   1st Qu.: 223.0   1st Qu.: 795.8   GasA :1428   Fa: 49   
##  Median :   0.00   Median : 477.5   Median : 991.5   GasW :  18   Gd:241   
##  Mean   :  46.55   Mean   : 567.2   Mean   :1057.4   Grav :   7   Po:  1   
##  3rd Qu.:   0.00   3rd Qu.: 808.0   3rd Qu.:1298.2   OthW :   2   TA:428   
##  Max.   :1474.00   Max.   :2336.0   Max.   :6110.0   Wall :   4            
##                                                                            
##  CentralAir Electrical     X1stFlrSF      X2ndFlrSF     LowQualFinSF    
##  N:  95     FuseA:  94   Min.   : 334   Min.   :   0   Min.   :  0.000  
##  Y:1365     FuseF:  27   1st Qu.: 882   1st Qu.:   0   1st Qu.:  0.000  
##             FuseP:   3   Median :1087   Median :   0   Median :  0.000  
##             Mix  :   1   Mean   :1163   Mean   : 347   Mean   :  5.845  
##             SBrkr:1334   3rd Qu.:1391   3rd Qu.: 728   3rd Qu.:  0.000  
##             NA's :   1   Max.   :4692   Max.   :2065   Max.   :572.000  
##                                                                         
##    GrLivArea     BsmtFullBath     BsmtHalfBath       FullBath    
##  Min.   : 334   Min.   :0.0000   Min.   :0.00000   Min.   :0.000  
##  1st Qu.:1130   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.000  
##  Median :1464   Median :0.0000   Median :0.00000   Median :2.000  
##  Mean   :1515   Mean   :0.4253   Mean   :0.05753   Mean   :1.565  
##  3rd Qu.:1777   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:2.000  
##  Max.   :5642   Max.   :3.0000   Max.   :2.00000   Max.   :3.000  
##                                                                   
##     HalfBath       BedroomAbvGr    KitchenAbvGr   KitchenQual 
TotRmsAbvGrd
##  Min.   :0.0000   Min.   :0.000   Min.   :0.000   Ex:100      Min.   : 
2.000
##  1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:1.000   Fa: 39      1st Qu.: 
5.000
##  Median :0.0000   Median :3.000   Median :1.000   Gd:586      Median : 
6.000
##  Mean   :0.3829   Mean   :2.866   Mean   :1.047   TA:735      Mean   : 
6.518
##  3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1.000               3rd Qu.: 
7.000
##  Max.   :2.0000   Max.   :8.000   Max.   :3.000               Max.   
:14.000
##                                                                          
##  Functional    Fireplaces    FireplaceQu   GarageType   GarageYrBlt  
##  Maj1:  14   Min.   :0.000   Ex  : 24    2Types :  6   Min.   :1900  
##  Maj2:   5   1st Qu.:0.000   Fa  : 33    Attchd :870   1st Qu.:1961  
##  Min1:  31   Median :1.000   Gd  :380    Basment: 19   Median :1980  
##  Min2:  34   Mean   :0.613   Po  : 20    BuiltIn: 88   Mean   :1979  
##  Mod :  15   3rd Qu.:1.000   TA  :313    CarPort:  9   3rd Qu.:2002  
##  Sev :   1   Max.   :3.000   NA's:690    Detchd :387   Max.   :2010  
##  Typ :1360                               NA's   : 81   NA's   :81    
##  GarageFinish   GarageCars      GarageArea     GarageQual  GarageCond
```

```
##  Fin :352     Min.   :0.000   Min.   :   0.0   Ex  :   3   Ex  :   2
##  RFn :422     1st Qu.:1.000   1st Qu.: 334.5   Fa  :  48   Fa  :  35
##  Unf :605     Median :2.000   Median : 480.0   Gd  :  14   Gd  :   9
##  NA's: 81     Mean   :1.767   Mean   : 473.0   Po  :   3   Po  :   7
##               3rd Qu.:2.000   3rd Qu.: 576.0   TA  :1311   TA  :1326
##               Max.   :4.000   Max.   :1418.0   NA's:  81   NA's:  81
##
##  PavedDrive   WoodDeckSF      OpenPorchSF     EnclosedPorch
X3SsnPorch
##  N:  90     Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   :
0.00
##  P:  30     1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:
0.00
##  Y:1340     Median :  0.00   Median : 25.00   Median :  0.00   Median :
0.00
##             Mean   : 94.24   Mean   : 46.66   Mean   : 21.95   Mean   :
3.41
##             3rd Qu.:168.00   3rd Qu.: 68.00   3rd Qu.:  0.00   3rd Qu.:
0.00
##             Max.   :857.00   Max.   :547.00   Max.   :552.00   Max.
:508.00
##
##   ScreenPorch        PoolArea        PoolQC       Fence      MiscFeature
##  Min.   :  0.00   Min.   :  0.000   Ex   :   2   GdPrv:  59   Gar2:   2
##  1st Qu.:  0.00   1st Qu.:  0.000   Fa   :   2   GdWo :  54   Othr:   2
##  Median :  0.00   Median :  0.000   Gd   :   3   MnPrv: 157   Shed:  49
##  Mean   : 15.06   Mean   :  2.759   NA's:1453   MnWw :  11   TenC:   1
##  3rd Qu.:  0.00   3rd Qu.:  0.000                NA's :1179   NA's:1406
##  Max.   :480.00   Max.   :738.000
##
##     MiscVal            MoSold           YrSold       SaleType
##  Min.   :    0.00   Min.   : 1.000   Min.   :2006   WD     :1267
##  1st Qu.:    0.00   1st Qu.: 5.000   1st Qu.:2007   New    : 122
##  Median :    0.00   Median : 6.000   Median :2008   COD    :  43
##  Mean   :   43.49   Mean   : 6.322   Mean   :2008   ConLD  :   9
##  3rd Qu.:    0.00   3rd Qu.: 8.000   3rd Qu.:2009   ConLI  :   5
##  Max.   :15500.00   Max.   :12.000   Max.   :2010   ConLw  :   5
##                                                     (Other):   9
##  SaleCondition    SalePrice
##  Abnorml: 101   Min.   : 34900
##  AdjLand:   4   1st Qu.:129975
##  Alloca :  12   Median :163000
##  Family :  20   Mean   :180921
##  Normal :1198   3rd Qu.:214000
##  Partial: 125   Max.   :755000
##
```

From this command, we can see the summary from every column. More importantly, the summary statistics of the numerical columns. We can see that the average price of a home is 180,921. We can also see that an average home was built in 1971 with around 2.8 rooms above ground.

Data Cleaning

As seen above, this dataset has rows with missing values, let's check how many missing values there actually are.

```
NA_values = data.frame(NA_value=colSums(is.na(train_data)))
NA_values
```

```
##                NA_value
## Id                    0
## MSSubClass            0
## MSZoning              0
## LotFrontage         259
## LotArea               0
## Street                0
## Alley              1369
## LotShape              0
## LandContour           0
## Utilities             0
## LotConfig             0
## LandSlope             0
## Neighborhood          0
## Condition1            0
## Condition2            0
## BldgType              0
## HouseStyle            0
## OverallQual           0
## OverallCond           0
## YearBuilt             0
## YearRemodAdd          0
## RoofStyle             0
## RoofMatl              0
## Exterior1st           0
## Exterior2nd           0
## MasVnrType            8
## MasVnrArea            8
## ExterQual             0
## ExterCond             0
## Foundation            0
## BsmtQual             37
## BsmtCond             37
## BsmtExposure         38
## BsmtFinType1         37
## BsmtFinSF1            0
## BsmtFinType2         38
## BsmtFinSF2            0
```

```
## BsmtUnfSF          0
## TotalBsmtSF        0
## Heating            0
## HeatingQC          0
## CentralAir         0
## Electrical         1
## X1stFlrSF          0
## X2ndFlrSF          0
## LowQualFinSF       0
## GrLivArea          0
## BsmtFullBath       0
## BsmtHalfBath       0
## FullBath           0
## HalfBath           0
## BedroomAbvGr       0
## KitchenAbvGr       0
## KitchenQual        0
## TotRmsAbvGrd       0
## Functional         0
## Fireplaces         0
## FireplaceQu      690
## GarageType        81
## GarageYrBlt       81
## GarageFinish      81
## GarageCars         0
## GarageArea         0
## GarageQual        81
## GarageCond        81
## PavedDrive         0
## WoodDeckSF         0
## OpenPorchSF        0
## EnclosedPorch      0
## X3SsnPorch         0
## ScreenPorch        0
## PoolArea           0
## PoolQC          1453
## Fence           1179
## MiscFeature     1406
## MiscVal            0
## MoSold             0
## YrSold             0
## SaleType           0
## SaleCondition      0
## SalePrice          0
```

Let's drop any columns with missing values in both the training and test set, as well as the ID column as it is not useful.

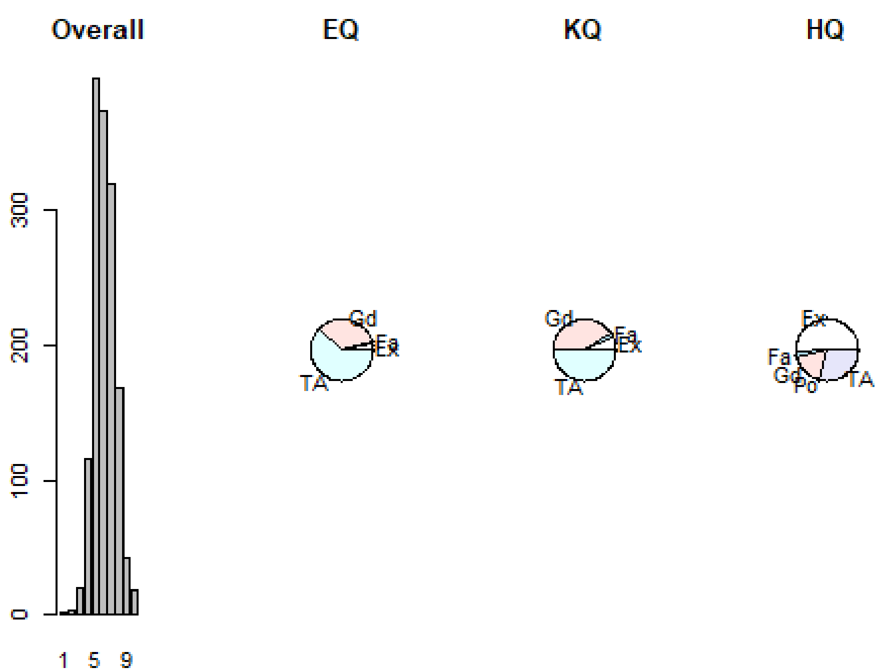
```
# Getting rid of training and testing columns with missing values
train_data = subset(train_data, select = -
```

```
c(Id,LotFrontage,Alley,MasVnrType,MasVnrArea,BsmtQual,BsmtCond,BsmtExposure,B
smtFinType1,BsmtFinType2,Electrical,FireplaceQu,GarageType,GarageYrBlt,Garage
Finish,GarageQual,GarageCond,PoolQC,Fence,MiscFeature))

test_data = subset(test_dataa, select = -
c(Id,LotFrontage,Alley,MasVnrType,MasVnrArea,BsmtQual,BsmtCond,BsmtExposure,B
smtFinType1,BsmtFinType2,Electrical,FireplaceQu,GarageType,GarageYrBlt,Garage
Finish,GarageQual,GarageCond,PoolQC,Fence,MiscFeature))
```

Exploratory Data Analysis on Training Set

Pie Plot

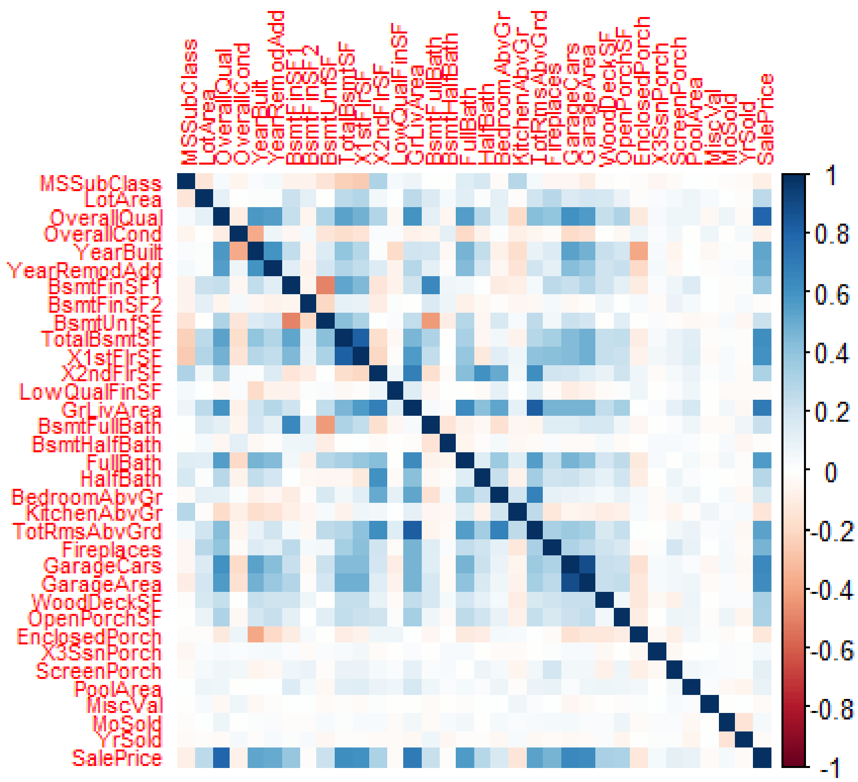
```
par(mfrow=c(1,4))
barplot(table(train_data$OverallQual),main="Overall")

pie(table(train_data$ExterQual), labels =
names(table(train_data$ExterQual)),main="EQ")

pie(table(train_data$KitchenQual), labels =
names(table(train_data$KitchenQual)),main="KQ")

pie(table(train_data$HeatingQC), labels =
names(table(train_data$HeatingQC)),main="HQ")
```



As we can see, most houses have quality of 5-6 which are average. This coincides with our kitchen and heating quality as most are “TA” or average. Surprisingly, most houses have excellent exterior quality. This may be due to how much the kitchen and heating get used over time so it wears down.

Correlation Matrix Plot

```
numerical_data <- train_data %>% dplyr::select(where(is.numeric))
cor_data=data.frame(numerical_data)
correlation = cor(cor_data)
par(mfrow=c(1,1))
corrplot(correlation,method="color", tl.cex = 0.7)
```

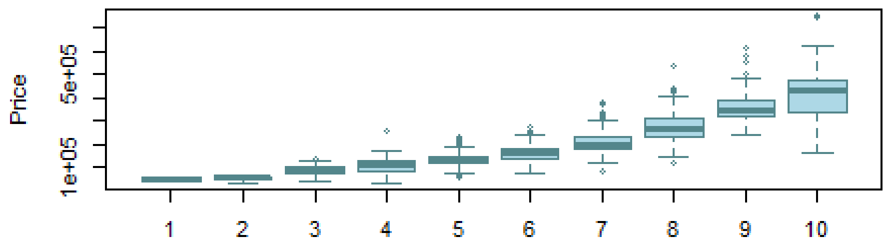


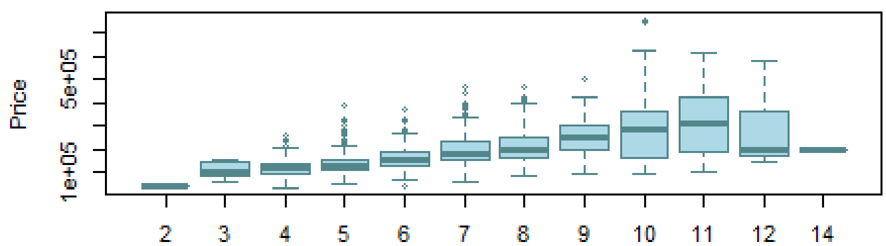
In our correlation plot of only our numerical columns, we can see some interesting findings. Importantly, the SalePrice column has some very strong positive correlations with the overall quality of the house (OverallQual) and the above ground living area (GrLivArea), with also some strong correlations with most columns. There are columns with not much correlation with SalePrice as we can see, the building class (MSSubClass) has almost no correlation since the color is white. Most columns after the Enclosed Porch column have no correlation to SalePrice. There are not many negative correlations to SalePrice aswell.

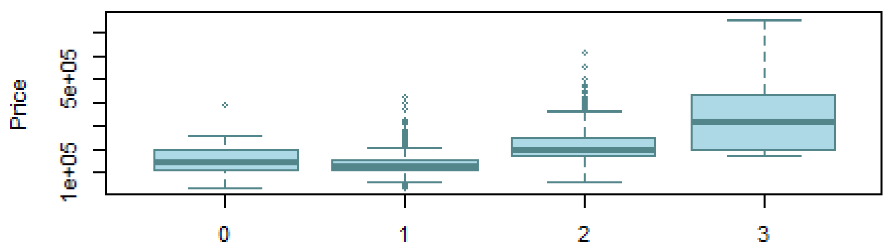
Strongly Correlated Columns Boxplot

```
par(mfrow=c(3,1))
par(plt = c(0.1, 0.9, 0.3, 1))

boxplot(SalePrice~OverallQual, data=train_data, col="lightblue",
border="cadetblue4",
        main="SalePrice and Overall Quality",
```

```
        xlab="Overall Quality", ylab="Price")

boxplot(SalePrice~TotRmsAbvGrd, data=train_data, col="lightblue", border="cadetblue4",
        main="SalePrice and #Above Ground Rooms",
        xlab="Total Rooms Above Ground", ylab="Price")

boxplot(SalePrice~FullBath, data=train_data, col="lightblue", border="cadetblue4",
        main="SalePrice and #Full Bathrooms",
        xlab="Full Bathrooms", ylab="Price")
```







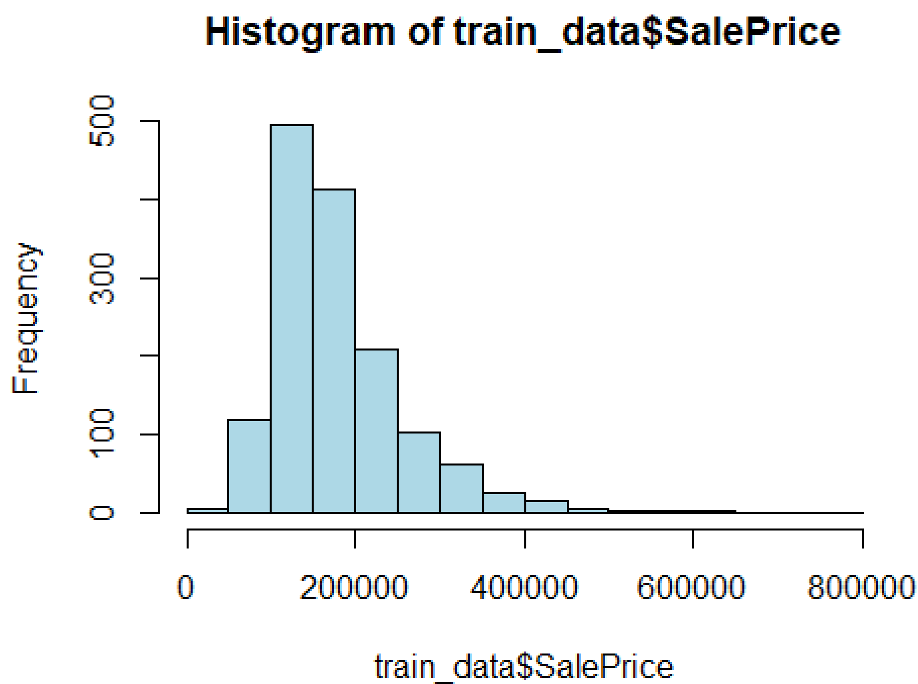
We can see the general trend in these boxplots. When the overall quality increases on average, so does the price of a house as expected. The total above ground rooms, there seems to be a positive correlation but the price goes down when there are 12-14 rooms for some reason. This could be due to different factors of each house individually which cause a decrease in the overall median price. When there is 1 full bathroom, it seems to have the same median price as a house with 0 full baths, which is quite unexpected.

```
print(sum(train_data$FullBath == 0))

## [1] 9

print(sum(train_data$FullBath == 1))

## [1] 650
```

As shown above, the it is hard to compare 0 Full bathroom houses with 1 Full Bathroom houses since there are not many houses with 0 full bathrooms in the dataset, hence why they have similar median prices.

Histogram of SalePrice to see Price distribution

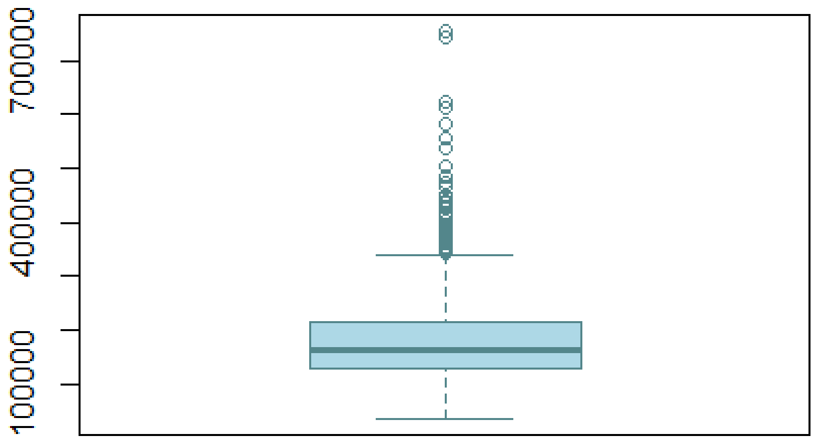
```
options(scipen = 999)
hist(train_data$SalePrice, col="lightblue")
```



As we can see, most of the houses in this dataset are $100,000 - $200,000.

SalePrice Boxplot to see outliers

```
boxplot(train_data$SalePrice, col="lightblue", border = "cadetblue4")
```



As we can see, the outliers are above around $350,000, while most houses range between around $150,000 to $220,000. We notice that the number of outliers may not be significant enough in comparison to the entire dataset to remove.

Data Altering

There are a lot of columns that are not correlated to the price of a house. Since we are using a linear classifier, we should see the effects of a raw dataset with all of our columns, and another dataset with reduced columns that have no correlation ones removed. We should also check if the number of outliers are significant to our entire dataset.

```
message("Total Number of Training Data: ", nrow(train_data))

## Total Number of Training Data: 1460

message("Number of outliers: ", sum(train_data$SalePrice > 350000))

## Number of outliers: 54
```

As we can see, there are clearly not enough significant outliers that can heavily alter our dataset, so it is not worth removing them.

Now, Let's create two linear models, one which has every column in the dataset, while the other excludes columns that have low correlation with SalePrice

```
numerical_data <- train_data %>% dplyr::select(where(is.numeric))
cor_data=data.frame(numerical_data)
print(cor(cor_data$SalePrice, cor_data))
```

```
##       MSSubClass   LotArea OverallQual OverallCond YearBuilt YearRemodAdd
## [1,] -0.08428414 0.2638434   0.7909816 -0.07785589 0.5228973     0.507101
##      BsmtFinSF1  BsmtFinSF2 BsmtUnfSF TotalBsmtSF X1stFlrSF X2ndFlrSF
## [1,]  0.3864198 -0.01137812 0.2144791   0.6135806 0.6058522 0.3193338
##      LowQualFinSF GrLivArea BsmtFullBath BsmtHalfBath  FullBath  HalfBath
## [1,]  -0.02560613 0.7086245    0.2271222  -0.01684415 0.5606638 0.2841077
##      BedroomAbvGr KitchenAbvGr TotRmsAbvGrd Fireplaces GarageCars
GarageArea
## [1,]    0.1682132   -0.1359074    0.5337232  0.4669288  0.6404092
0.6234314
##      WoodDeckSF OpenPorchSF EnclosedPorch X3SsnPorch ScreenPorch
PoolArea
## [1,]  0.3244134   0.3158562     -0.128578 0.04458367   0.1114466
0.09240355
##          MiscVal     MoSold      YrSold SalePrice
## [1,] -0.02118958 0.04643225 -0.02892259         1
```

We should try a model with columns of correlation above absolute value of 0.3. This way, we do not lose too many features while also retaining the most important factors of house price.

Validation Set

Let's also create a validation set since the test set does not have labels in which we can check our metrics

```
set.seed(42)
sample = sample.split(cor_data, SplitRatio = 0.9)
train.data = subset(cor_data, sample==TRUE)
val.data = subset(cor_data, sample==FALSE)

numerical_t <- test_data %>% dplyr::select(where(is.numeric))
cor_dat=data.frame(numerical_t)

cor_dat[is.na(cor_dat)] <- 0
test.data <- model.matrix(~.,cor_dat)[,-1]
```

Data Modelling

Linear Regression 1

Firstly, Lets create a general Linear model. One with every column and another with only columns with high correlation.

```r
model_orig = lm(SalePrice ~ MSSubClass+LotArea+OverallQual+OverallCond
+YearBuilt+BsmtFinSF1+YearRemodAdd+BsmtFinSF2+BsmtUnfSF+TotalBsmtSF+X1stFlrSF
+X2ndFlrSF+LowQualFinSF +GrLivArea +
BsmtFullBath+BsmtHalfBath+FullBath+HalfBath+BedroomAbvGr+KitchenAbvGr
+TotRmsAbvGrd +Fireplaces +GarageCars +GarageArea+WoodDeckSF +OpenPorchSF
+EnclosedPorch + X3SsnPorch +ScreenPorch+PoolArea+MiscVal + MoSold+
YrSold,data = train.data)

model_reduced = lm(SalePrice ~ OverallQual
+YearBuilt+BsmtFinSF1+YearRemodAdd+TotalBsmtSF+X1stFlrSF +X2ndFlrSF
+GrLivArea +FullBath+TotRmsAbvGrd +Fireplaces +GarageCars
+GarageArea+WoodDeckSF +OpenPorchSF,data = train.data)
```

Lasso Regression

Let's create a Lasso regression model. We will compare all these models in the next part.

```r
library(glmnet)

## Warning: package 'glmnet' was built under R version 4.2.3

## Loading required package: Matrix

## Warning: package 'Matrix' was built under R version 4.2.3

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-8

set.seed(42)
x <- model.matrix(SalePrice~., train.data)[,-1]
y <- train.data$SalePrice

x_val <- model.matrix(SalePrice~., val.data)[,-1]
y_val <- val.data$SalePrice

cv <- cv.glmnet(x,y,alpha=1)

# Fit the model on training data using lowest lambda
modelLass <- glmnet(x,y,alpha=1, lambda=cv$lambda.min)
coef(modelLass)

## 34 x 1 sparse Matrix of class "dgCMatrix"
##                           s0
## (Intercept)   -847338.2958875
## MSSubClass        -88.0074685
## LotArea             0.1948199
```

```
## OverallQual     21075.3152618
## OverallCond         .
## YearBuilt         161.7608713
## YearRemodAdd      237.6645869
## BsmtFinSF1         14.1611180
## BsmtFinSF2          .
## BsmtUnfSF           .
## TotalBsmtSF        10.1828146
## X1stFlrSF           5.0687187
## X2ndFlrSF           .
## LowQualFinSF        .
## GrLivArea          42.9509141
## BsmtFullBath     2936.9278669
## BsmtHalfBath        .
## FullBath            .
## HalfBath            .
## BedroomAbvGr        .
## KitchenAbvGr    -2264.6577129
## TotRmsAbvGrd        .
## Fireplaces       3936.6657142
## GarageCars      11023.7857163
## GarageArea          2.6579724
## WoodDeckSF         19.6422321
## OpenPorchSF         .
## EnclosedPorch       .
## X3SsnPorch          .
## ScreenPorch         .
## PoolArea            .
## MiscVal             .
## MoSold              .
## YrSold              .
```

Elastic Net Regression

```
set.seed(42)
modelNet <- train(SalePrice ~., data=train.data, method="glmnet",
trControl=trainControl("cv",number=10), tuneLength = 10)

modelNet
```

```
## glmnet
##
## 1288 samples
##   33 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1159, 1159, 1160, 1159, 1159, 1159, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      RMSE      Rsquared   MAE
```

```
##    0.1       67.1298  36112.99  0.8006632  21927.71
##    0.1      155.0785  36112.99  0.8006632  21927.71
##    0.1      358.2515  36112.99  0.8006632  21927.71
##    0.1      827.6075  36060.86  0.8011534  21880.06
##    0.1     1911.8807  35985.81  0.8019241  21783.73
##    0.1     4416.6923  35891.08  0.8030804  21615.30
##    0.1    10203.1320  35999.80  0.8032724  21518.41
##    0.1    23570.5583  37078.83  0.7976901  22009.34
##    0.1    54451.0468  40018.02  0.7892309  23774.93
##    0.2       67.1298  36116.25  0.8006069  21932.01
##    0.2      155.0785  36116.25  0.8006069  21932.01
##    0.2      358.2515  36096.36  0.8007989  21913.76
##    0.2      827.6075  36036.85  0.8013884  21844.75
##    0.2     1911.8807  35968.36  0.8021008  21733.75
##    0.2     4416.6923  35941.70  0.8027885  21617.55
##    0.2    10203.1320  36578.11  0.7987037  21872.95
##    0.2    23570.5583  38298.98  0.7916543  22774.16
##    0.2    54451.0468  43271.31  0.7819889  26435.92
##    0.3       67.1298  36117.51  0.8005950  21933.63
##    0.3      155.0785  36117.51  0.8005950  21933.63
##    0.3      358.2515  36080.94  0.8009492  21895.77
##    0.3      827.6075  36025.15  0.8014950  21817.96
##    0.3     1911.8807  35956.46  0.8022339  21717.65
##    0.3     4416.6923  36127.03  0.8012456  21726.58
##    0.3    10203.1320  37044.19  0.7955403  22205.69
##    0.3    23570.5583  39468.24  0.7870941  23588.58
##    0.3    54451.0468  46989.09  0.7725753  29905.05
##    0.4       67.1298  36119.96  0.8005900  21931.95
##    0.4      155.0785  36116.32  0.8006217  21928.87
##    0.4      358.2515  36067.88  0.8010770  21877.39
##    0.4      827.6075  36014.17  0.8015980  21795.33
##    0.4     1911.8807  35956.20  0.8022663  21712.36
##    0.4     4416.6923  36379.86  0.7990023  21898.63
##    0.4    10203.1320  37549.01  0.7921979  22493.84
##    0.4    23570.5583  40712.30  0.7820659  24553.03
##    0.4    54451.0468  50695.25  0.7643866  33332.98
##    0.5       67.1298  36119.02  0.8006149  21930.55
##    0.5      155.0785  36105.37  0.8007176  21919.55
##    0.5      358.2515  36055.52  0.8011944  21859.73
##    0.5      827.6075  36000.83  0.8017118  21778.17
##    0.5     1911.8807  35998.79  0.8018873  21731.66
##    0.5     4416.6923  36686.91  0.7961497  22126.99
##    0.5    10203.1320  38057.10  0.7887529  22852.08
##    0.5    23570.5583  42025.36  0.7759593  25639.02
##    0.5    54451.0468  54404.90  0.7532590  36548.39
##    0.6       67.1298  36116.20  0.8006290  21929.53
##    0.6      155.0785  36095.63  0.8008036  21909.99
##    0.6      358.2515  36049.00  0.8012481  21845.63
##    0.6      827.6075  35985.35  0.8018504  21760.32
##    0.6     1911.8807  36078.97  0.8011955  21766.76
```

```
##   0.6     4416.6923  36899.12  0.7944217  22267.97
##   0.6    10203.1320  38577.09  0.7850389  23250.00
##   0.6    23570.5583  43503.89  0.7665794  26922.41
##   0.6    54451.0468  58460.50  0.7259307  39979.16
##   0.7       67.1298  36117.31  0.8006142  21930.26
##   0.7      155.0785  36089.05  0.8008679  21901.09
##   0.7      358.2515  36044.81  0.8012809  21834.13
##   0.7      827.6075  35974.40  0.8019493  21749.27
##   0.7     1911.8807  36179.50  0.8002798  21814.85
##   0.7     4416.6923  37078.60  0.7930379  22364.24
##   0.7    10203.1320  39072.37  0.7815631  23655.41
##   0.7    23570.5583  44963.10  0.7569792  28240.16
##   0.7    54451.0468  62166.51  0.6991492  43108.72
##   0.8       67.1298  36114.78  0.8006268  21929.86
##   0.8      155.0785  36082.93  0.8009233  21893.55
##   0.8      358.2515  36037.20  0.8013466  21822.43
##   0.8      827.6075  35965.21  0.8020353  21743.13
##   0.8     1911.8807  36290.07  0.7992346  21884.28
##   0.8     4416.6923  37253.84  0.7916940  22466.77
##   0.8    10203.1320  39588.76  0.7777147  24083.00
##   0.8    23570.5583  46287.64  0.7500972  29418.33
##   0.8    54451.0468  65720.45  0.6715741  46098.13
##   0.9       67.1298  36115.40  0.8006190  21930.51
##   0.9      155.0785  36077.55  0.8009745  21885.56
##   0.9      358.2515  36031.74  0.8013915  21813.24
##   0.9      827.6075  35969.58  0.8019961  21742.46
##   0.9     1911.8807  36419.57  0.7979926  21965.17
##   0.9     4416.6923  37442.26  0.7902141  22597.98
##   0.9    10203.1320  40114.96  0.7735845  24517.84
##   0.9    23570.5583  47699.38  0.7409547  30694.05
##   0.9    54451.0468  69218.41  0.6334724  49019.91
##   1.0       67.1298  36112.90  0.8006400  21927.45
##   1.0      155.0785  36072.70  0.8010177  21878.49
##   1.0      358.2515  36025.62  0.8014413  21806.18
##   1.0      827.6075  35985.43  0.8018439  21749.46
##   1.0     1911.8807  36560.02  0.7966230  22065.35
##   1.0     4416.6923  37643.93  0.7886044  22749.16
##   1.0    10203.1320  40652.79  0.7691381  24971.72
##   1.0    23570.5583  49252.63  0.7272645  32088.07
##   1.0    54451.0468  72575.49  0.6327540  51807.66
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0.1 and lambda =
4416.692.
```

```
modelNet$bestTune
```

```
##   alpha   lambda
## 6   0.1 4416.692
```

```r
coef(modelNet$finalModel, modelNet$bestTune$lambda)
```

```
## 34 x 1 sparse Matrix of class "dgCMatrix"
##                              s1
## (Intercept)   -446044.6606322
## MSSubClass       -124.1910671
## LotArea             0.3069953
## OverallQual     17454.8949706
## OverallCond      2975.4227571
## YearBuilt         259.5813826
## YearRemodAdd      210.0733854
## BsmtFinSF1         12.3885024
## BsmtFinSF2         -1.3857927
## BsmtUnfSF           .
## TotalBsmtSF        12.5431143
## X1stFlrSF          16.5328450
## X2ndFlrSF          13.8555191
## LowQualFinSF      -27.6945759
## GrLivArea          27.8625508
## BsmtFullBath     6980.1065437
## BsmtHalfBath      219.3510046
## FullBath         4777.7752290
## HalfBath            .
## BedroomAbvGr    -6868.6949722
## KitchenAbvGr   -14607.7915368
## TotRmsAbvGrd     4558.5791748
## Fireplaces       4817.1568442
## GarageCars       9853.6589058
## GarageArea          8.1078208
## WoodDeckSF         27.5033252
## OpenPorchSF         .
## EnclosedPorch       .
## X3SsnPorch          6.6175783
## ScreenPorch        34.7193789
## PoolArea          -19.8937409
## MiscVal             .
## MoSold             67.9057240
## YrSold           -268.3969512
```

Single Layer Neural Network

```r
library(keras)
```

```
## Warning: package 'keras' was built under R version 4.2.3
```

```r
library(tensorflow)
```

```
## Warning: package 'tensorflow' was built under R version 4.2.3

##
## Attaching package: 'tensorflow'
```

```
## The following object is masked from 'package:caret':
##
##     train
```

```
NNmodel <- keras_model_sequential () %>%
  layer_dense(units=50,activation="relu",input_shape=ncol(x)) %>%
  layer_dropout(rate=0.2) %>%
  layer_dense(units=1)

NNmodel %>% compile(loss="mse", optimizer = optimizer_adam(), metrics =
list("mean_absolute_error"))

#Fitting out NN model
history <- NNmodel %>% fit(x, y, epochs=50, batch_size=4,
validation_data=list(x_val,y_val))
```

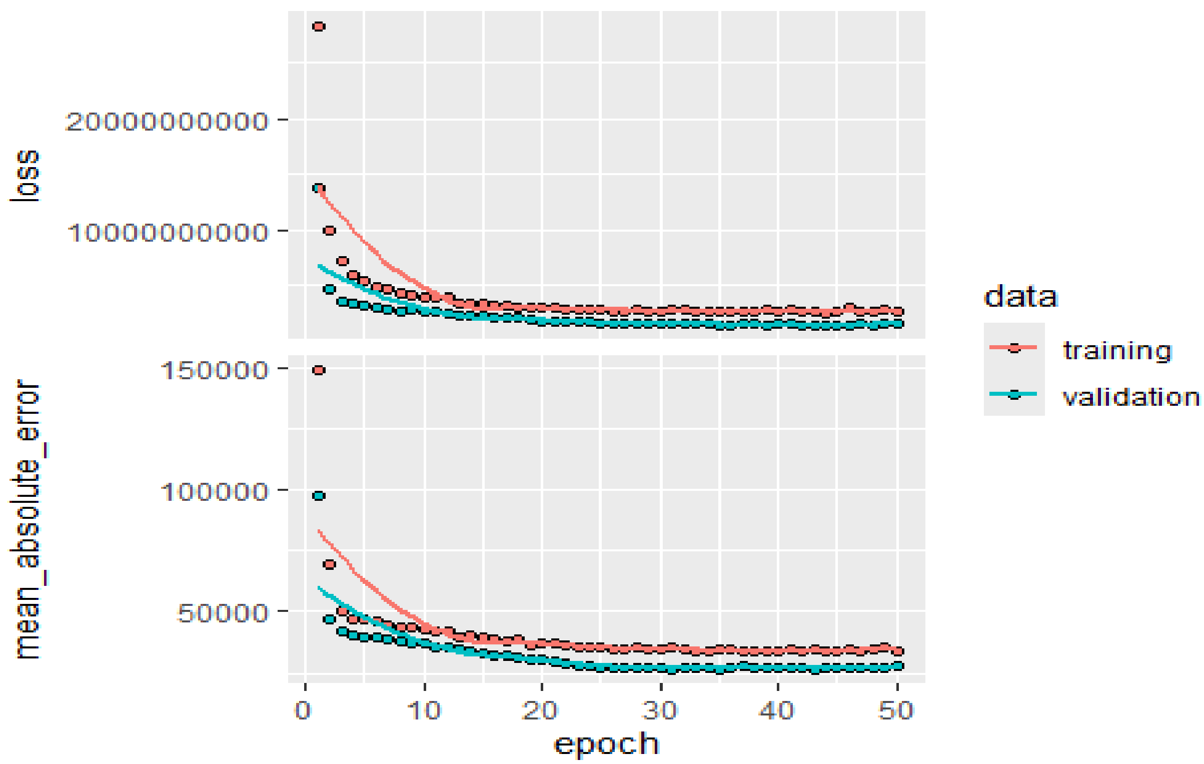
```
## Epoch 1/50
## 322/322 - 1s - loss: 28281761792.0000 - mean_absolute_error: 149165.3281 -
val_loss: 13853976576.0000 - val_mean_absolute_error: 97784.9453 - 1s/epoch -
4ms/step
## Epoch 2/50
## 322/322 - 1s - loss: 9932403712.0000 - mean_absolute_error: 69713.3125 -
val_loss: 4693713920.0000 - val_mean_absolute_error: 46865.5625 - 713ms/epoch
- 2ms/step
## Epoch 3/50
## 322/322 - 1s - loss: 7184622080.0000 - mean_absolute_error: 50536.0117 -
val_loss: 3626127104.0000 - val_mean_absolute_error: 41690.0039 - 706ms/epoch
- 2ms/step
## Epoch 4/50
## 322/322 - 1s - loss: 6019992576.0000 - mean_absolute_error: 46959.5547 -
val_loss: 3340926720.0000 - val_mean_absolute_error: 40630.5078 - 722ms/epoch
- 2ms/step
## Epoch 5/50
## 322/322 - 1s - loss: 5437470720.0000 - mean_absolute_error: 46798.5625 -
val_loss: 3128451584.0000 - val_mean_absolute_error: 39899.2539 - 612ms/epoch
- 2ms/step
## Epoch 6/50
## 322/322 - 1s - loss: 4894033920.0000 - mean_absolute_error: 45847.8516 -
val_loss: 2949830656.0000 - val_mean_absolute_error: 39406.3086 - 583ms/epoch
- 2ms/step
## Epoch 7/50
## 322/322 - 1s - loss: 4596299776.0000 - mean_absolute_error: 44612.0820 -
val_loss: 2894654720.0000 - val_mean_absolute_error: 38460.5156 - 584ms/epoch
- 2ms/step
## Epoch 8/50
## 322/322 - 1s - loss: 4252326912.0000 - mean_absolute_error: 43478.0000 -
val_loss: 2752000768.0000 - val_mean_absolute_error: 38202.8125 - 610ms/epoch
- 2ms/step
## Epoch 9/50
## 322/322 - 1s - loss: 4059081728.0000 - mean_absolute_error: 43592.2656 -
```

```
val_loss: 2776863232.0000 - val_mean_absolute_error: 36964.5273 - 585ms/epoch
- 2ms/step
## Epoch 10/50
## 322/322 - 1s - loss: 4024349184.0000 - mean_absolute_error: 42565.0156 -
val_loss: 2595917824.0000 - val_mean_absolute_error: 37010.2578 - 660ms/epoch
- 2ms/step
## Epoch 11/50
## 322/322 - 1s - loss: 3889306880.0000 - mean_absolute_error: 42232.5234 -
val_loss: 2581872128.0000 - val_mean_absolute_error: 35645.6523 - 594ms/epoch
- 2ms/step
## Epoch 12/50
## 322/322 - 1s - loss: 3951964928.0000 - mean_absolute_error: 42003.0195 -
val_loss: 2536089344.0000 - val_mean_absolute_error: 35070.3906 - 591ms/epoch
- 2ms/step
## Epoch 13/50
## 322/322 - 1s - loss: 3417050624.0000 - mean_absolute_error: 39489.0078 -
val_loss: 2378597376.0000 - val_mean_absolute_error: 34986.9727 - 578ms/epoch
- 2ms/step
## Epoch 14/50
## 322/322 - 1s - loss: 3436976128.0000 - mean_absolute_error: 40303.4336 -
val_loss: 2353676032.0000 - val_mean_absolute_error: 33867.2969 - 611ms/epoch
- 2ms/step
## Epoch 15/50
## 322/322 - 1s - loss: 3390011136.0000 - mean_absolute_error: 39942.0664 -
val_loss: 2271399680.0000 - val_mean_absolute_error: 33037.0156 - 599ms/epoch
- 2ms/step
## Epoch 16/50
## 322/322 - 1s - loss: 3297310720.0000 - mean_absolute_error: 38721.6875 -
val_loss: 2178615040.0000 - val_mean_absolute_error: 32249.4512 - 591ms/epoch
- 2ms/step
## Epoch 17/50
## 322/322 - 1s - loss: 3293226240.0000 - mean_absolute_error: 37852.7539 -
val_loss: 2051465728.0000 - val_mean_absolute_error: 32370.3164 - 580ms/epoch
- 2ms/step
## Epoch 18/50
## 322/322 - 1s - loss: 3059139840.0000 - mean_absolute_error: 38469.7305 -
val_loss: 2085962880.0000 - val_mean_absolute_error: 30898.6191 - 609ms/epoch
- 2ms/step
## Epoch 19/50
## 322/322 - 1s - loss: 3068960000.0000 - mean_absolute_error: 36358.8359 -
val_loss: 1950469120.0000 - val_mean_absolute_error: 30504.9336 - 593ms/epoch
- 2ms/step
## Epoch 20/50
## 322/322 - 1s - loss: 2978004480.0000 - mean_absolute_error: 37281.6367 -
val_loss: 1836294912.0000 - val_mean_absolute_error: 30186.5469 - 590ms/epoch
- 2ms/step
## Epoch 21/50
## 322/322 - 1s - loss: 3092871680.0000 - mean_absolute_error: 37347.8125 -
val_loss: 1793910656.0000 - val_mean_absolute_error: 29326.5371 - 587ms/epoch
- 2ms/step
```

```
## Epoch 22/50
## 322/322 - 1s - loss: 2836470016.0000 - mean_absolute_error: 35907.5977 -
val_loss: 1772911744.0000 - val_mean_absolute_error: 28845.6504 - 601ms/epoch
- 2ms/step
## Epoch 23/50
## 322/322 - 1s - loss: 2844355328.0000 - mean_absolute_error: 35422.7539 -
val_loss: 1720975872.0000 - val_mean_absolute_error: 28355.4590 - 584ms/epoch
- 2ms/step
## Epoch 24/50
## 322/322 - 1s - loss: 2775760384.0000 - mean_absolute_error: 35349.6289 -
val_loss: 1677808256.0000 - val_mean_absolute_error: 27806.7031 - 585ms/epoch
- 2ms/step
## Epoch 25/50
## 322/322 - 1s - loss: 2840767232.0000 - mean_absolute_error: 35431.0664 -
val_loss: 1637430912.0000 - val_mean_absolute_error: 27498.0781 - 587ms/epoch
- 2ms/step
## Epoch 26/50
## 322/322 - 1s - loss: 2734164736.0000 - mean_absolute_error: 35024.2305 -
val_loss: 1644825216.0000 - val_mean_absolute_error: 27321.3770 - 585ms/epoch
- 2ms/step
## Epoch 27/50
## 322/322 - 1s - loss: 2629021696.0000 - mean_absolute_error: 34225.1328 -
val_loss: 1583942912.0000 - val_mean_absolute_error: 27305.3770 - 580ms/epoch
- 2ms/step
## Epoch 28/50
## 322/322 - 1s - loss: 2889126912.0000 - mean_absolute_error: 35627.6641 -
val_loss: 1587999744.0000 - val_mean_absolute_error: 27230.9863 - 588ms/epoch
- 2ms/step
## Epoch 29/50
## 322/322 - 1s - loss: 2684827648.0000 - mean_absolute_error: 34526.8789 -
val_loss: 1520063872.0000 - val_mean_absolute_error: 27310.4062 - 604ms/epoch
- 2ms/step
## Epoch 30/50
## 322/322 - 1s - loss: 2755955712.0000 - mean_absolute_error: 34549.2695 -
val_loss: 1554321280.0000 - val_mean_absolute_error: 26938.0059 - 584ms/epoch
- 2ms/step
## Epoch 31/50
## 322/322 - 1s - loss: 2882132992.0000 - mean_absolute_error: 35165.8555 -
val_loss: 1504360448.0000 - val_mean_absolute_error: 26680.8984 - 594ms/epoch
- 2ms/step
## Epoch 32/50
## 322/322 - 1s - loss: 2802217984.0000 - mean_absolute_error: 34216.1719 -
val_loss: 1544844288.0000 - val_mean_absolute_error: 27031.9395 - 587ms/epoch
- 2ms/step
## Epoch 33/50
## 322/322 - 1s - loss: 2714020352.0000 - mean_absolute_error: 33967.0156 -
val_loss: 1513739520.0000 - val_mean_absolute_error: 26831.5020 - 598ms/epoch
- 2ms/step
## Epoch 34/50
## 322/322 - 1s - loss: 2704364288.0000 - mean_absolute_error: 34183.5508 -
```

```
val_loss: 1517229440.0000 - val_mean_absolute_error: 26922.4121 - 584ms/epoch
- 2ms/step
## Epoch 35/50
## 322/322 - 1s - loss: 2674246400.0000 - mean_absolute_error: 34617.4258 -
val_loss: 1474669952.0000 - val_mean_absolute_error: 26749.0840 - 582ms/epoch
- 2ms/step
## Epoch 36/50
## 322/322 - 1s - loss: 2755875584.0000 - mean_absolute_error: 34366.6367 -
val_loss: 1469974016.0000 - val_mean_absolute_error: 26770.6367 - 645ms/epoch
- 2ms/step
## Epoch 37/50
## 322/322 - 1s - loss: 2588229376.0000 - mean_absolute_error: 33527.0625 -
val_loss: 1576558464.0000 - val_mean_absolute_error: 27586.8730 - 584ms/epoch
- 2ms/step
## Epoch 38/50
## 322/322 - 1s - loss: 2660746496.0000 - mean_absolute_error: 33934.0820 -
val_loss: 1527195136.0000 - val_mean_absolute_error: 27324.3867 - 581ms/epoch
- 2ms/step
## Epoch 39/50
## 322/322 - 1s - loss: 2812005376.0000 - mean_absolute_error: 34100.0312 -
val_loss: 1451259520.0000 - val_mean_absolute_error: 26785.5078 - 584ms/epoch
- 2ms/step
## Epoch 40/50
## 322/322 - 1s - loss: 2734551808.0000 - mean_absolute_error: 34008.7188 -
val_loss: 1505810304.0000 - val_mean_absolute_error: 27251.5254 - 590ms/epoch
- 2ms/step
## Epoch 41/50
## 322/322 - 1s - loss: 2766029568.0000 - mean_absolute_error: 34267.1797 -
val_loss: 1515094912.0000 - val_mean_absolute_error: 27357.5527 - 584ms/epoch
- 2ms/step
## Epoch 42/50
## 322/322 - 1s - loss: 2648256512.0000 - mean_absolute_error: 33872.7969 -
val_loss: 1477571968.0000 - val_mean_absolute_error: 27098.9980 - 587ms/epoch
- 2ms/step
## Epoch 43/50
## 322/322 - 1s - loss: 2590374400.0000 - mean_absolute_error: 34580.2227 -
val_loss: 1425483776.0000 - val_mean_absolute_error: 26700.3633 - 610ms/epoch
- 2ms/step
## Epoch 44/50
## 322/322 - 1s - loss: 2564006144.0000 - mean_absolute_error: 33786.2188 -
val_loss: 1434891392.0000 - val_mean_absolute_error: 26883.0059 - 585ms/epoch
- 2ms/step
## Epoch 45/50
## 322/322 - 1s - loss: 2592933376.0000 - mean_absolute_error: 34074.8281 -
val_loss: 1457150464.0000 - val_mean_absolute_error: 27077.7676 - 604ms/epoch
- 2ms/step
## Epoch 46/50
## 322/322 - 1s - loss: 2945060352.0000 - mean_absolute_error: 34758.0156 -
val_loss: 1463529984.0000 - val_mean_absolute_error: 27087.9121 - 687ms/epoch
- 2ms/step
```

```
## Epoch 47/50
## 322/322 - 1s - loss: 2694374144.0000 - mean_absolute_error: 33500.2734 -
val_loss: 1503827840.0000 - val_mean_absolute_error: 27375.2129 - 623ms/epoch
- 2ms/step
## Epoch 48/50
## 322/322 - 1s - loss: 2683096320.0000 - mean_absolute_error: 34446.2930 -
val_loss: 1435055744.0000 - val_mean_absolute_error: 26869.6484 - 579ms/epoch
- 2ms/step
## Epoch 49/50
## 322/322 - 1s - loss: 2891406080.0000 - mean_absolute_error: 35096.5781 -
val_loss: 1510734464.0000 - val_mean_absolute_error: 27511.3809 - 581ms/epoch
- 2ms/step
## Epoch 50/50
## 322/322 - 1s - loss: 2672335104.0000 - mean_absolute_error: 33863.3828 -
val_loss: 1529366912.0000 - val_mean_absolute_error: 27670.3730 - 597ms/epoch
- 2ms/step

plot(history)
```



Model Comparisons

So we have created 5 models: A regular linear regression with all our columns, another regular linear regression with strongly correlated columns, a Lasso regression model, An ElasticNet regression model, and a simple Neural Network model.

One thing we notice in our neural network model is that the validation loss and validation absolute error is less than the training set equivalent. This could have been due to our dropout rate which reduced the overfitting in our training set.

Let's compare R squared values in our regression models.

Regular regression R^2

```
summary(model_orig)

##
## Call:
## lm(formula = SalePrice ~ MSSubClass + LotArea + OverallQual +
##     OverallCond + YearBuilt + BsmtFinSF1 + YearRemodAdd + BsmtFinSF2 +
##     BsmtUnfSF + TotalBsmtSF + X1stFlrSF + X2ndFlrSF + LowQualFinSF +
##     GrLivArea + BsmtFullBath + BsmtHalfBath + FullBath + HalfBath +
##     BedroomAbvGr + KitchenAbvGr + TotRmsAbvGrd + Fireplaces +
##     GarageCars + GarageArea + WoodDeckSF + OpenPorchSF + EnclosedPorch +
##     X3SsnPorch + ScreenPorch + PoolArea + MiscVal + MoSold +
##     YrSold, data = train.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -475511  -16893   -2383   13380  289170
##
## Coefficients: (2 not defined because of singularities)
##                  Estimate   Std. Error t value             Pr(>|t|)
## (Intercept)    345266.9063 1557437.0846   0.222             0.824592
## MSSubClass       -155.9361      28.9110  -5.394   0.0000000824221369 ***
## LotArea             0.3463       0.1060   3.266             0.001119 **
## OverallQual     18308.4077    1302.3680  14.058 < 0.0000000000000002 ***
## OverallCond      4094.3065    1109.2275   3.691             0.000233 ***
## YearBuilt         326.0210      66.7722   4.883   0.0000011816567526 ***
## BsmtFinSF1         22.8103       5.0835   4.487   0.0000078803809301 ***
## YearRemodAdd      149.5174      71.4315   2.093             0.036535 *
## BsmtFinSF2          5.9618       7.8855   0.756             0.449763
## BsmtUnfSF          11.6895       4.5968   2.543             0.011111 *
## TotalBsmtSF             NA           NA      NA                   NA
## X1stFlrSF          48.4809       6.2356   7.775   0.0000000000000156 ***
## X2ndFlrSF          48.2298       5.4493   8.851 < 0.0000000000000002 ***
## LowQualFinSF       -6.1005      24.5491  -0.249             0.803788
## GrLivArea               NA           NA      NA                   NA
## BsmtFullBath     9255.4046    2862.1693   3.234             0.001254 **
## BsmtHalfBath     3378.1330    4442.1203   0.760             0.447112
## FullBath         3830.4355    3105.9303   1.233             0.217708
## HalfBath        -1375.9813    2926.9289  -0.470             0.638357
## BedroomAbvGr   -10194.7654    1857.2890  -5.489   0.0000000488419628 ***
## KitchenAbvGr   -14752.3253    5692.7753  -2.591             0.009669 **
## TotRmsAbvGrd     5378.7065    1350.4815   3.983   0.0000720153832422 ***
## Fireplaces       3587.7710    1921.2523   1.867             0.062077 .
## GarageCars      10962.9651    3120.5299   3.513             0.000459 ***
```

```
## GarageArea          -0.8172      10.6759  -0.077             0.938998
## WoodDeckSF          28.5755       8.6994   3.285             0.001049 **
## OpenPorchSF        -14.3054      16.2529  -0.880             0.378932
## EnclosedPorch        9.8982      18.3120   0.541             0.588926
## X3SsnPorch          17.7876      33.2294   0.535             0.592539
## ScreenPorch         47.8275      18.8194   2.541             0.011160 *
## PoolArea           -36.0041      24.4594  -1.472             0.141272
## MiscVal             -0.4850       1.9282  -0.252             0.801429
## MoSold             186.9684     376.4723   0.497             0.619535
## YrSold            -671.7207     774.8394  -0.867             0.386154
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35920 on 1256 degrees of freedom
## Multiple R-squared:  0.8013, Adjusted R-squared:  0.7964
## F-statistic: 163.4 on 31 and 1256 DF,  p-value: < 0.00000000000000022
```

```
summary(model_reduced)
```

```
##
## Call:
## lm(formula = SalePrice ~ OverallQual + YearBuilt + BsmtFinSF1 +
##     YearRemodAdd + TotalBsmtSF + X1stFlrSF + X2ndFlrSF + GrLivArea +
##     FullBath + TotRmsAbvGrd + Fireplaces + GarageCars + GarageArea +
##     WoodDeckSF + OpenPorchSF, data = train.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -511461  -17382   -2175   14405  290530
##
## Coefficients:
##                  Estimate   Std. Error t value             Pr(>|t|)
## (Intercept)  -1081490.742   137374.324  -7.873  0.00000000000000739 ***
## OverallQual     20019.147     1271.679  15.742 < 0.0000000000000002 ***
## YearBuilt         175.488       54.301   3.232             0.001262 **
## BsmtFinSF1         19.117        2.819   6.781  0.00000000001821744 ***
## YearRemodAdd      333.681       66.845   4.992  0.00000068114781674 ***
## TotalBsmtSF        13.441        4.665   2.881             0.004029 **
## X1stFlrSF          57.612       25.526   2.257             0.024177 *
## X2ndFlrSF          50.108       25.173   1.991             0.046748 *
## GrLivArea         -11.206       25.050  -0.447             0.654709
## FullBath        -1674.446     2848.891  -0.588             0.556802
## TotRmsAbvGrd     2081.566     1176.934   1.769             0.077195 .
## Fireplaces       7268.603     1918.762   3.788             0.000159 ***
## GarageCars      10164.873     3209.093   3.168             0.001574 **
## GarageArea         10.807       10.966   0.986             0.324541
## WoodDeckSF         35.645        8.800   4.051  0.00005414055959256 ***
## OpenPorchSF        -3.722       16.658  -0.223             0.823216
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 37340 on 1272 degrees of freedom
## Multiple R-squared:  0.7826, Adjusted R-squared:   0.78
## F-statistic: 305.2 on 15 and 1272 DF,  p-value: < 0.00000000000000022
```

Surprisingly, our linear model that has every column has a better R^2 than the select columns with high correlation. This may be due to too many features lost which underfit the training data.

Lasso and ElasticNet R^2 and RMSE

Firstly using lasso and ElasticNet, we make predictions on our validation set and compare the RMSE as well

```
# LASSO
predictionLasso <- modelLass %>% predict(x_val)

# RMSE and R^2
data.frame(
RMSE = caret::RMSE(predictionLasso, y_val),
Rsquare = caret::R2(predictionLasso, y_val))

##       RMSE        s0
## 1 30717.86 0.8497609

# ELASTIC NET
predictionNet <- modelNet %>% predict(x_val)

# RMSE and R^2
data.frame(
RMSE = caret::RMSE(predictionNet, y_val),
Rsquare = caret::R2(predictionNet, y_val))

##       RMSE   Rsquare
## 1 29225.56 0.8625493
```

As we can see, out of all of our regression models, ElasticNet has the largest R squared value meaning it better fits our training data. We also notice the RMSE of ElasticNet is lower than Lasso which concludes that it is more accurate.

When viewing the coefficients of ElasticNet above, we see that the intercept $\hat{\beta}_0$ is -446044. This model also gives us predictors for multiple other columns but also does not include some such as BsmtUnfSF.

Therefore out of our regression models, ElasticNet seems to be the best performing.

ElasticNet vs Neural Network

We will now see the final predictions of our ElasticNet regression model, and compare to the neural network model we made. We will submit to kaggle and see our score on the testing data.

```
elastic <- modelNet %>% predict(test.data)
elastic <- cbind(Id = test_dataa$Id, elastic)
colnames(elastic)[colnames(elastic) == 'elastic'] <- 'SalePrice'

NN <- predict(NNmodel, test.data)

## 46/46 - 0s - 125ms/epoch - 3ms/step

NN <- cbind(Id = test_dataa$Id, NN)
colnames(NN)[colnames(NN) == ''] <- 'SalePrice'

write.csv(elastic, "elastic.csv", row.names = FALSE)

write.csv(NN, "nn.csv", row.names = FALSE)
```

Results

After Submitting, it turns out our Neural Network model is the best performing model, beating out all the linear regression variants. The score for the NN model on Kaggle was 0.22206 while the ElasticNet score was 0.34880. Both are very solid scores but the NN model is the best. This may be due to the number of epochs that I have ran for the NN model, 50 is a lot.

Therefore, overall the best regression model was our ElasticNet model, but it has been beaten by a single layer neural network.

For future model improvements, some heavy feature selection can be made while also testing different parameters like the alpha values in regression.