Ames, Iowa House Data

Hamish Lyon

01-01-2020

Introduction

The Dataset

The AmesDataset is a sample of the houses which have been sold in Ames, Iowa, I have chosen this data set because it has a significant number of features which can be explored. It is also appealing because I am currently in the process of looking at houses to buy myself, albiet in Western Australia, and as this is a real set of real estate data so I am interested in whether or not there is any interesting features which drive the price of houses which might be counter intuitive.

```
## # A tibble: 1,460 x 1
##    HouseData$Id $MSSubClass $MSZoning $LotFrontage $LotArea $Street $Alley
##           <int>       <int> <fct>            <int>    <int> <fct>   <fct>
##  1            1          60 RL                  65     8450 Pave    <NA>
##  2            2          20 RL                  80     9600 Pave    <NA>
##  3            3          60 RL                  68    11250 Pave    <NA>
##  4            4          70 RL                  60     9550 Pave    <NA>
##  5            5          60 RL                  84    14260 Pave    <NA>
##  6            6          50 RL                  85    14115 Pave    <NA>
##  7            7          20 RL                  75    10084 Pave    <NA>
##  8            8          60 RL                  NA    10382 Pave    <NA>
##  9            9          50 RM                  51     6120 Pave    <NA>
## 10           10         190 RL                  50     7420 Pave    <NA>
## # … with 1,450 more rows, and 74 more variables: $LotShape <fct>,
## #   $LandContour <fct>, $Utilities <fct>, $LotConfig <fct>, $LandSlope <fct>,
## #   $Neighborhood <fct>, $Condition1 <fct>, $Condition2 <fct>, $BldgType <fct>,
## #   $HouseStyle <fct>, $OverallQual <int>, $OverallCond <int>,
## #   $YearBuilt <int>, $YearRemodAdd <int>, $RoofStyle <fct>, $RoofMatl <fct>,
## #   $Exterior1st <fct>, $Exterior2nd <fct>, $MasVnrType <fct>,
## #   $MasVnrArea <int>, $ExterQual <fct>, $ExterCond <fct>, $Foundation <fct>,
## #   $BsmtQual <fct>, $BsmtCond <fct>, $BsmtExposure <fct>, $BsmtFinType1 <fct>,
## #   $BsmtFinSF1 <int>, $BsmtFinType2 <fct>, $BsmtFinSF2 <int>,
## #   $BsmtUnfSF <int>, $TotalBsmtSF <int>, $Heating <fct>, $HeatingQC <fct>,
## #   $CentralAir <fct>, $Electrical <fct>, $X1stFlrSF <int>, $X2ndFlrSF <int>,
## #   $LowQualFinSF <int>, $GrLivArea <int>, $BsmtFullBath <int>,
## #   $BsmtHalfBath <int>, $FullBath <int>, $HalfBath <int>, $BedroomAbvGr <int>,
## #   $KitchenAbvGr <int>, $KitchenQual <fct>, $TotRmsAbvGrd <int>,
## #   $Functional <fct>, $Fireplaces <int>, $FireplaceQu <fct>,
## #   $GarageType <fct>, $GarageYrBlt <int>, $GarageFinish <fct>,
## #   $GarageCars <int>, $GarageArea <int>, $GarageQual <fct>, $GarageCond <fct>,
## #   $PavedDrive <fct>, $WoodDeckSF <int>, $OpenPorchSF <int>,
## #   $EnclosedPorch <int>, $X3SsnPorch <int>, $ScreenPorch <int>,
## #   $PoolArea <int>, $PoolQC <fct>, $Fence <fct>, $MiscFeature <fct>,
## #   $MiscVal <int>, $MoSold <int>, $YrSold <int>, $SaleType <fct>,
## #   $SaleCondition <fct>, $SalePrice <int>
```

Investing the structure of the dataset we can see that the dataset consistents of integers and factors. We can also see that there are quite a few features which measure different aspects of the same feature of the house. For example, the Basement of the house has BsmtQual, BasmtCond, BsmtExposure, BsmtFinType1, BsmtFinSF1, BasmtFinType2, BsmtFinSF2, BsmtUnfSF, and finally - TotalBsmtSF. This is true for quite a few features, some which common senses tell us will demonstrate collinearity, for example there is a measure of how many square feet there are in the garage and also how many cars will fit in the garage. This is essentially two different ways of measuring the size of the garage of a premise.

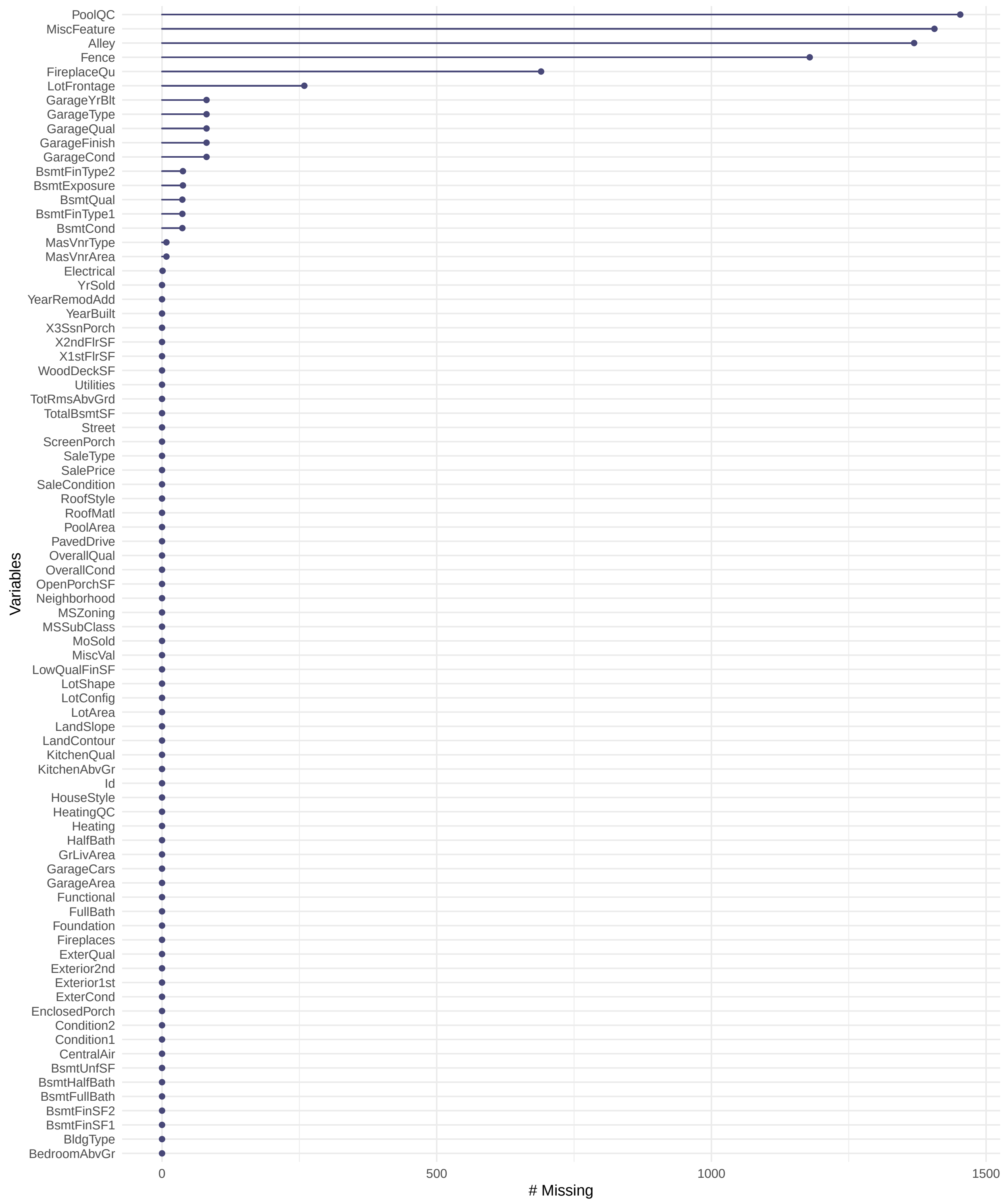
Another interesting feature of the dataset is that ther are a lot of NA values on the PoolQC, Fence, MiscFeature, and Alley - intuitively it seems like those houses don't have a pool, a fence, aren't next to an alley, and don't have any interesting additional features worth mentioning. Interesting there is only one feature which indicates the location, in my experience location is very important to determining the price of a house so it'll be interesting to see how this feature impacts the prediction of the price.

Methods

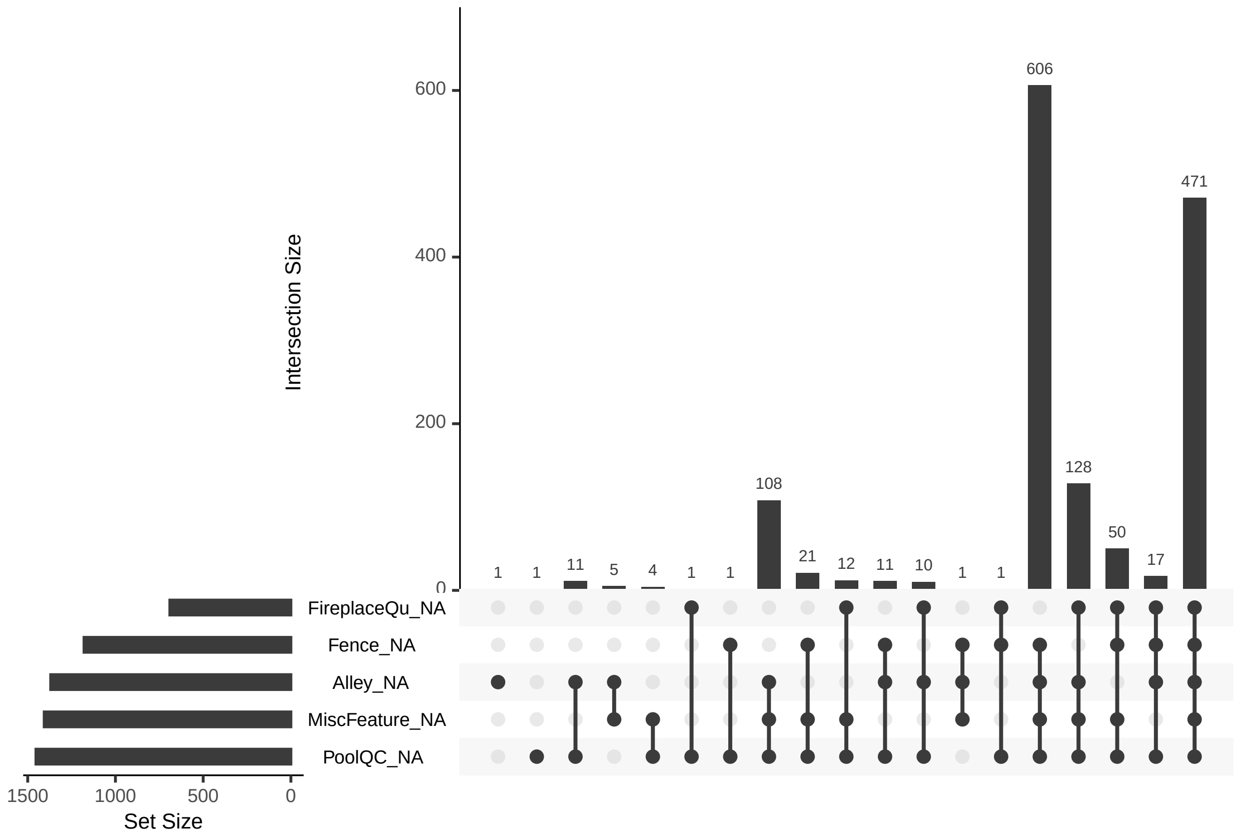
In the method sections we'll look at cleaning the data as previously we identified a lot of NA values. Exploring the dataset by visualising some of the more interesting relationships; discuss any insights which are gleaned from this analysis, how this influenced the modelling approach.

Data Cleaning

Before we take a look at some of the data in any details - lets first take a look at the missing features. While the graph below looks a little congested, I left it like this so it wouldn't take up too much space. So, it turns out that as we observed above PoolQC, Fence, MiscFeature, and Alley are nearly entirely NA. Additionally we see that just under half of the properties don't have a fire place and the remaining NAs relate to the LotFrontage, aspects of the Garage, and aspects of the Basement. There are a few features which are missing data but it it appears to be a bit more incomplete than the previously mentioned features.



There doesn't seem to be a structure to the missingness of the data other than the groups containing PoolQC, Fence, MiscFeature, and Alley seem to be houses without these features and NA is the method of recording this for the house.



Next, the features are divided into those which I am going to explicitly make the feature NA a factor and those for which I am going to impute a value; the imputation method is using a cart model. If I had more time I would explore the `MICE` package and imputation methods further, this was my first introduction to using a method like this and the idea that you can predictively replace data is very powerful.

First I convert MSSubClass into a factor as this is a factor like feature which R has read in as an integer.

```
HouseData <- HouseData %>%
  mutate(MSSubClass = as.factor(HouseData$MSSubClass))
```

Next, I have removed the following columns `MissingData <- c('PoolQC', 'MiscFeature', 'Alley', 'Fence')` and call the mice function on the remaining columns.

```
imputatedHouse <- mice(HouseData, m=1, method='cart', printFlag=FALSE)
HouseData <- complete(imputatedHouse)
```

Then, the features `c('PoolQC', 'MiscFeature', 'Alley', 'Fence')` are converted from a true NA value to adding NA as a factor so that it is available to future models.

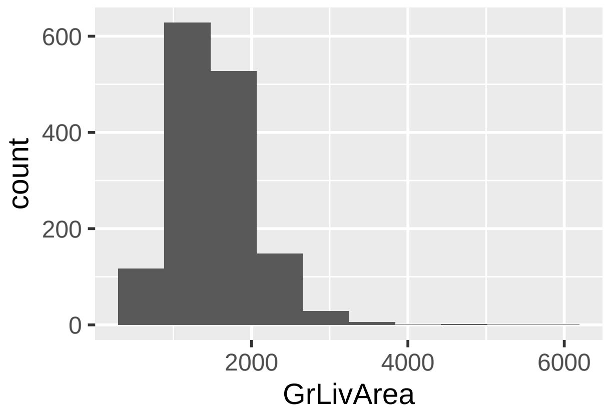
Finally, I bind the two datasets back into the original dataset - HouseData. Hopefully it's clear that for those features which I was resonably sure were intentionally not recorded have been treated appropriately and those which I was not sure about were predicted from the dataset itself.

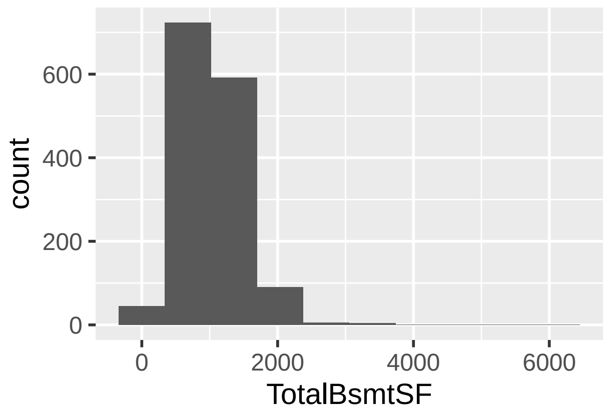
Data Exploration

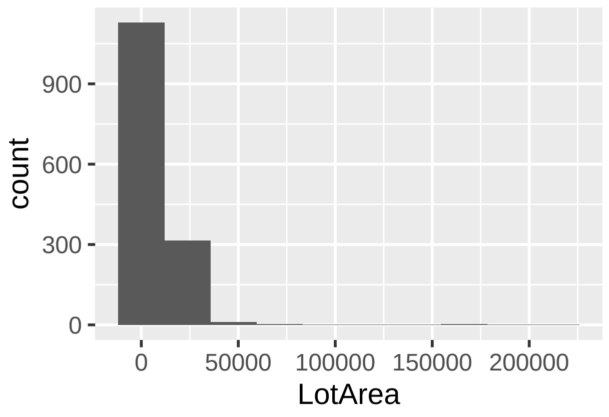
Looking at the summary of the data, `GrLivArea`, `TotalBsmtSF`, `LotArea`, and `LotFrontage` all have maximum values which seem to deviate from their distributions central location. Lets investigate the distributions individually to assess whether or not this is caused by outliers or by kurtosis by creating histograms of each variable.

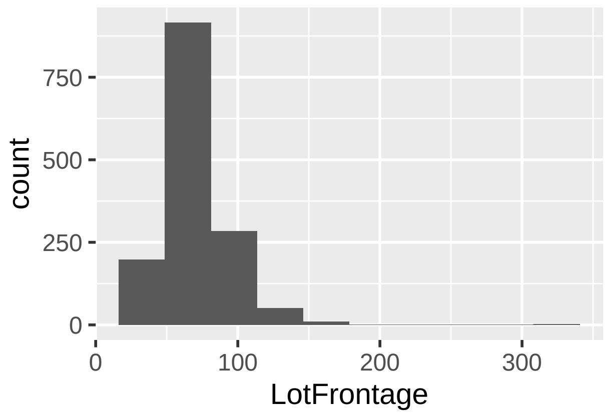
GrLivArea	TotalBsmtSF	LotArea	LotFrontage
Min. : 334	Min. : 0.0	Min. : 1300	Min. : 21.00
1st Qu.:1130	1st Qu.: 795.8	1st Qu.: 7554	1st Qu.: 59.00
Median :1464	Median : 991.5	Median : 9478	Median : 70.00
Mean :1515	Mean :1057.4	Mean : 10517	Mean : 70.28
3rd Qu.:1777	3rd Qu.:1298.2	3rd Qu.: 11602	3rd Qu.: 80.00
Max. :5642	Max. :6110.0	Max. :215245	Max. :313.00

Reviewing the histograms of these features below it is apparent that all these maximum values deviate from the distribution significantly enough for them to be considered outliers.







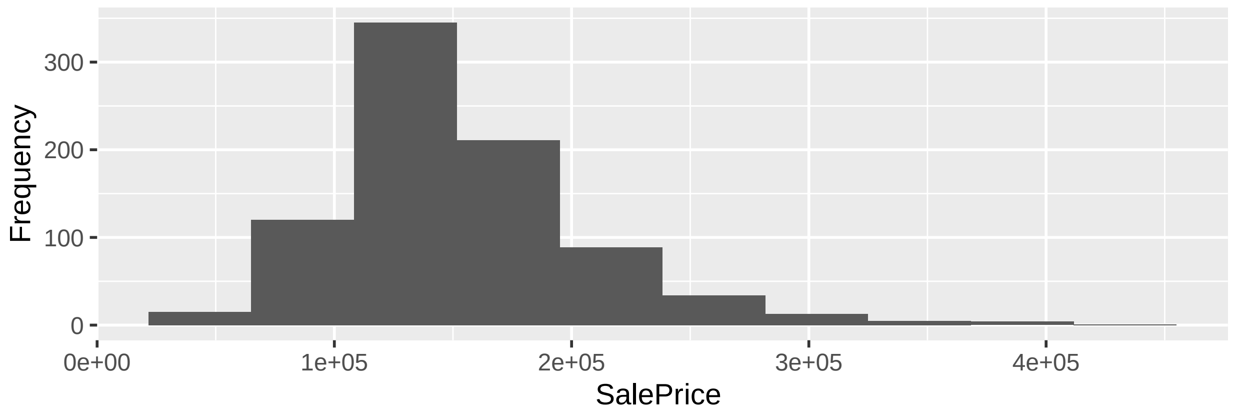


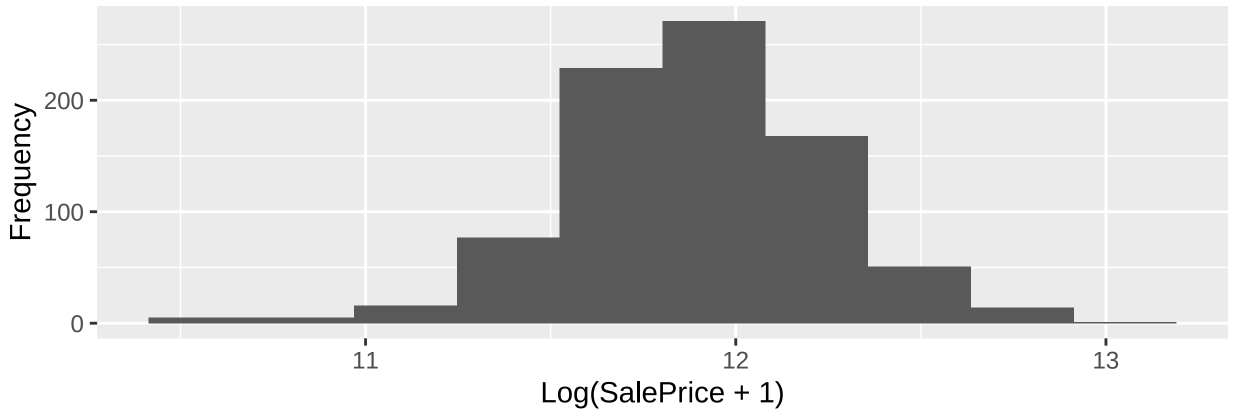
My method of removing outliers is quite basic, I inspected the graphs and filtered the data below to remove the outliers about a visually identified constant. I believe this method could be improved upon and in the future I would like to use a method which 'derives' the limts from the data. The constants can be found in the code below:

```r
HouseData <- HouseData %>%
              filter(GrLivArea < 3999.9,
                     TotalBsmtSF < 2999.9,
                     LotArea < 9999.9,
                     LotFrontage < 199.9)
```

Sale Price Transformation

Below there is a histrogram of SalePrice and following that a graph of a $log(SalePrice + 1)$ transformation graphed to illustrate the skewed nature of SalePrice and the result of the transformation.





Seeing this result it seems reasonable to assume that the $log(y + c)$ transformation produces a sufficiently normal outcome. The SalePrice is transformed below:

```r
HouseData$SalePrice <- log(HouseData$SalePrice + 1)
```

I did this to avoid skewing the errors during the evaluation of the prediction later in the report. I understand that my use of $log(y + 1)$ could be improved by fitting a c which improves the model outcome in the future. I also believe there are opportunities for additional transformations within the data as there are many other features which demonstrate visual skewedness but it falls outside of the scope of what I am doigng now.

Finally, the data is split into a training and testing at this point because there are no more transformations to be done on the data.

```r
set.seed(2)
indexSet <- createDataPartition(y = HouseData$SalePrice, times = 1, p = 0.1, list = FALSE)
trainingSet <- HouseData[-indexSet, ]
testingSet <- HouseData[indexSet, ]
```

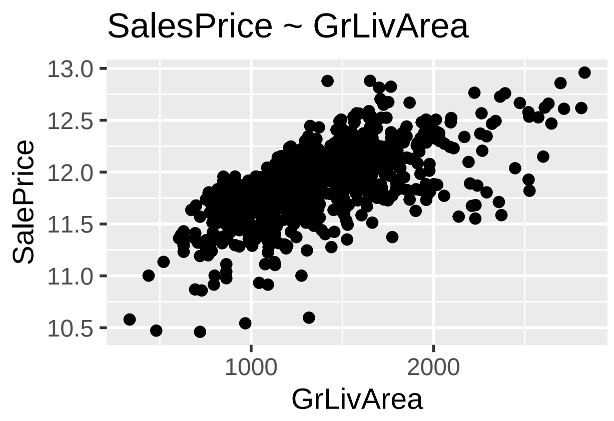
Modelling: Decision Tree

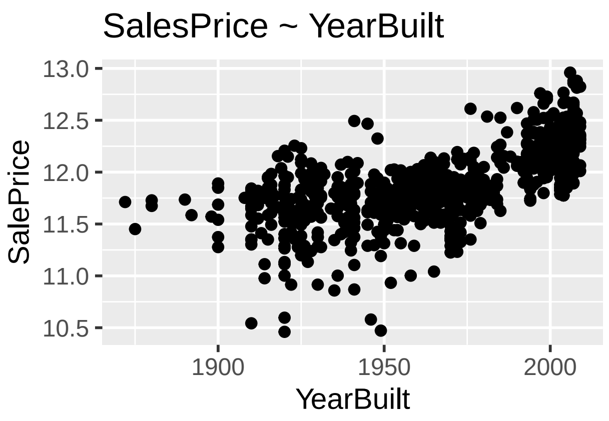
When thinking about the model I researched decision trees because it was the most intuitively similar approach to how I view house prices as a collection of individual decisions around how many rooms it has, whether or not it has a pool, and so forth. Below I create the output a tree model and print its output. When inspecting the various decision nodes I realised that is concerned with a lot variables such could more easily be predicted by linear relationships between variables, like GrLivArea.

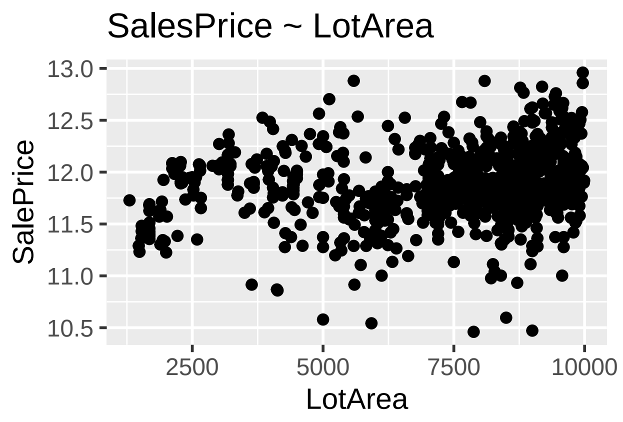
```
## n= 751
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 751 90.1359700 11.88629
##    2) YearBuilt< 1984.5 488 36.6015200 11.71800
##      4) OverallQual< 4.5 89  8.9354580 11.45262
##        8) X1stFlrSF< 735 22  2.9320430 11.09912
##         16) CentralAir=N 10  0.9340470 10.81433 *
##         17) CentralAir=Y 12  0.5111010 11.33644 *
##        9) X1stFlrSF>=735 67  2.3515810 11.56869 *
##      5) OverallQual>=4.5 399 19.9999400 11.77719
##       10) Neighborhood=BrDale,BrkSide,CollgCr,Edwards,IDOTRR,MeadowV,OldTown,Sawyer,SawyerW 223 10.4
##         20) GrLivArea< 1473.5 170  7.1236170 11.64150
##           40) MSSubClass=30,40,45,50,75,160,180,190 86  2.4847940 11.55147 *
##           41) MSSubClass=20,60,70,80,85,90,120 84  3.2282610 11.73367
##             82) YearRemodAdd< 1960 9  0.6215519 11.37434 *
##             83) YearRemodAdd>=1960 75  1.3052520 11.77678 *
##         21) GrLivArea>=1473.5 53  1.6031990 11.84651 *
##       11) Neighborhood=Blueste,ClearCr,Crawfor,Mitchel,NAmes,NPkVill,NWAmes,StoneBr,SWISU,Veenker 17
##         22) GrLivArea< 1104 62  0.8727686 11.75539 *
##         23) GrLivArea>=1104 114  3.2101780 11.95918
##           46) GrLivArea< 2046 107  1.9700130 11.93626 *
##           47) GrLivArea>=2046 7  0.3242599 12.30963 *
##    3) YearBuilt>=1984.5 263 14.0702700 12.19854
##      6) OverallQual< 7.5 207  6.8012710 12.13213
##       12) GrLivArea< 1206 34  0.3512241 11.89324 *
##       13) GrLivArea>=1206 173  4.1282960 12.17908
##         26) GrLivArea< 1725 135  2.0961220 12.13620 *
##         27) GrLivArea>=1725 38  0.9022768 12.33141 *
##      7) OverallQual>=7.5 56  2.9812210 12.44403
##       14) TotalBsmtSF< 1383 25  0.7545913 12.28878 *
##       15) TotalBsmtSF>=1383 31  1.1381020 12.56923 *
```

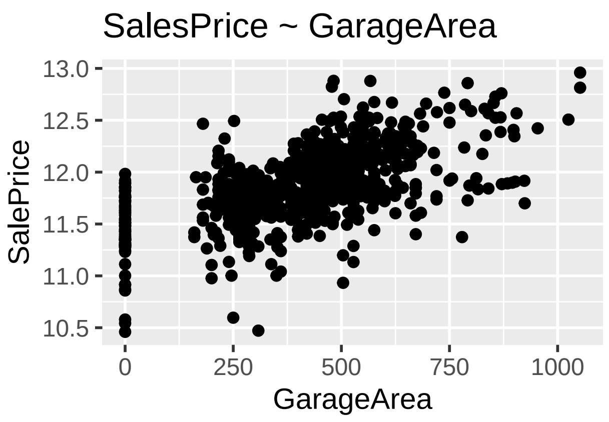
Following this I investigated some of the more important linear relationships with newly transformed SalePrice target variable.

Modelling: Exploring Linear Relationships









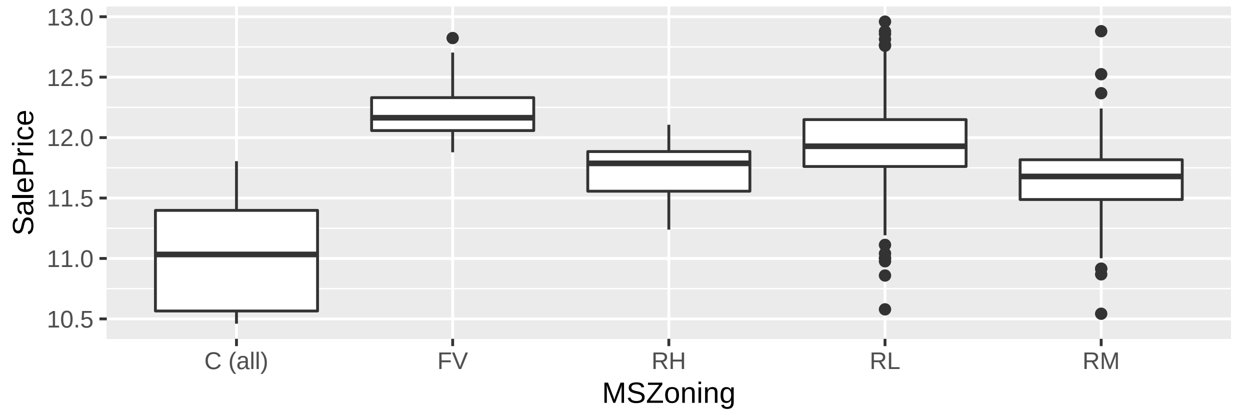
Below is a summary of some of the different independent variables and their relationship to the target variable, Sale Price. There is general linear response between some variables, most noteably `SalePrice ~ GrLivArea`.

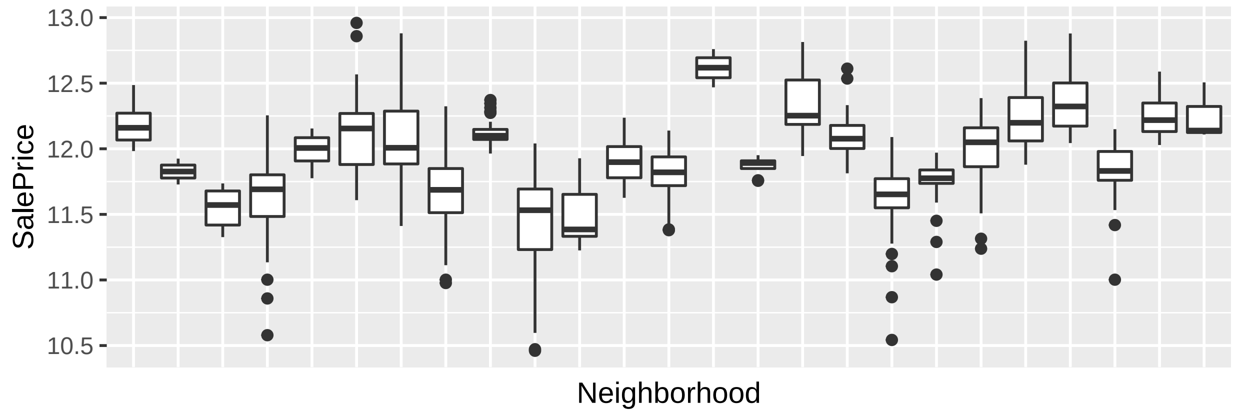
Modelling: Linear Models

The first linear model based on the analysis of linear impacts of the numerical values in the dataset is below:

```
##
## Call:
## lm(formula = SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt +
##     LotArea + GarageArea, data = trainingSet)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.95775 -0.07715  0.01403  0.09226  0.62525
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.971e+00  4.698e-01   4.196 3.04e-05 ***
## GrLivArea   3.579e-04  1.650e-05  21.699  < 2e-16 ***
## TotalBsmtSF 2.345e-04  2.033e-05  11.531  < 2e-16 ***
## YearBuilt   4.583e-03  2.420e-04  18.937  < 2e-16 ***
## LotArea     1.075e-05  2.945e-06   3.649 0.000281 ***
## GarageArea  2.777e-04  3.928e-05   7.070 3.57e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1682 on 745 degrees of freedom
## Multiple R-squared:  0.7661, Adjusted R-squared:  0.7645
## F-statistic: 487.9 on 5 and 745 DF,  p-value: < 2.2e-16
```

Beyond this analysis of numerical features there were several two factors which seems to vary significantly in response to the SalePrice variable.





```
##
## Call:
## lm(formula = SalePrice ~ GrLivArea + TotalBsmtSF + YearBuilt +
##     LotArea + GarageArea + Neighborhood, data = trainingSet)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.84016 -0.06931  0.01566  0.08315  0.46735
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         3.384e+00  8.664e-01   3.906 0.000103 ***
## GrLivArea           3.240e-04  1.740e-05  18.620  < 2e-16 ***
## TotalBsmtSF         1.968e-04  2.063e-05   9.538  < 2e-16 ***
## YearBuilt           3.940e-03  4.342e-04   9.075  < 2e-16 ***
## LotArea              1.281e-05  4.033e-06   3.176 0.001559 **
## GarageArea           2.386e-04  3.803e-05   6.273 6.09e-10 ***
## NeighborhoodBlueste -3.789e-02  1.612e-01  -0.235 0.814199
## NeighborhoodBrDale  -1.811e-01  5.912e-02  -3.063 0.002273 **
## NeighborhoodBrkSide -4.957e-02  5.481e-02  -0.904 0.366076
## NeighborhoodClearCr -1.676e-02  8.814e-02  -0.190 0.849220
## NeighborhoodCollgCr -5.751e-02  4.713e-02  -1.220 0.222715
## NeighborhoodCrawfor  1.564e-01  6.168e-02   2.535 0.011453 *
## NeighborhoodEdwards -1.468e-01  5.091e-02  -2.884 0.004039 **
## NeighborhoodGilbert -3.621e-02  5.377e-02  -0.674 0.500808
## NeighborhoodIDOTRR  -2.348e-01  5.880e-02  -3.993 7.20e-05 ***
## NeighborhoodMeadowV -2.215e-01  5.957e-02  -3.718 0.000216 ***
## NeighborhoodMitchel -6.685e-02  5.411e-02  -1.235 0.217057
## NeighborhoodNAmes   -7.011e-02  4.912e-02  -1.427 0.153914
## NeighborhoodNoRidge -6.372e-03  7.918e-02  -0.080 0.935882
## NeighborhoodNPkVill -4.684e-02  6.862e-02  -0.683 0.495077
## NeighborhoodNridgHt  6.756e-02  5.258e-02   1.285 0.199238
## NeighborhoodNWAmes  -7.886e-02  6.190e-02  -1.274 0.203077
## NeighborhoodOldTown -1.117e-01  5.419e-02  -2.061 0.039677 *
## NeighborhoodSawyer  -1.026e-01  5.287e-02  -1.942 0.052576 .
## NeighborhoodSawyerW -1.079e-01  5.323e-02  -2.026 0.043107 *
## NeighborhoodSomerst  3.910e-02  4.620e-02   0.846 0.397693
## NeighborhoodStoneBr  1.424e-01  5.876e-02   2.423 0.015656 *
## NeighborhoodSWISU   -2.327e-02  6.333e-02  -0.367 0.713440
## NeighborhoodTimber   3.755e-02  7.410e-02   0.507 0.612510
## NeighborhoodVeenker  1.054e-01  1.011e-01   1.043 0.297385
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1555 on 721 degrees of freedom
## Multiple R-squared:  0.8065, Adjusted R-squared:  0.7987
## F-statistic: 103.6 on 29 and 721 DF,  p-value: < 2.2e-16
```

When intepreting the outcome you can see that when `Neighborhood` is converted into a dummy variable only some are useful and this can practically be intepreted as the 'goodness' of each neighbourhood. Rather than trying to engineer this into a better feature I will try to better predict these complex factors below.

Modelling: Extreme Gradient Boosting

Finally, I realised upon studying output of the second linear model that there was significant feature engineering which would need to be done to make the factor however I understand that more sophisticated modelling methods like the `xgboost()` can capture a lot of this complexity without the need for the engineering of new features. For this reason I tried Extreme Gradient Boosting.

When building the model for lack of deep understanding of the xgboost package played and tested individual settings rather than following a rigorious tuning process. If I was progressing this further I would investigate an optimal tuning method for the many paramters required by the `xgboost()` algorithm. One aspect I focused on was avoiding over fitting to maximise the result on my testing set, to do this I restricted the number of runs to `early_stopping_rounds = 50`. The use of `sparse.model.matrix(SalePrice~ .-Id, trainingSet)` is memory saving as a sparse matrix will not save the zero values into memory.

```
set.seed(2)
extremeGradientBoosting <- xgboost(
  params = list(
  max_depth = 5,
  eta = 0.02,
  gamma = 0,
  colsample_bytree = 0.65,
  subsample = 0.6,
  min_child_weight = 3),
  data = sparse.model.matrix(
    SalePrice~ .-Id,
    trainingSet),
  label = trainingSet$SalePrice,
  nrounds = 1000, nfold = 10,
  showsd = F,
  stratified = T,
  print_every_n = 100,
  early_stopping_rounds = 50,
  maximize = F)
```

```
## [1]  train-rmse:11.164755
## Will train until train_rmse hasn't improved in 50 rounds.
##
## [101]    train-rmse:1.510746
## [201]    train-rmse:0.234009
## [301]    train-rmse:0.076535
## [401]    train-rmse:0.055953
## [501]    train-rmse:0.046218
## [601]    train-rmse:0.039086
## [701]    train-rmse:0.033356
## [801]    train-rmse:0.028594
## [901]    train-rmse:0.024881
## [1000]   train-rmse:0.021781
```

Now we inspect the featured with high Gain, which is the improvement in accuracy brought by a feature to the branches it is on. Interestingly GrLivArea, TotalBsmtSF, and many other variables which were identified earlier for the linear model show up as features which are most requently used by the model:

```
## Selecting by Frequency
```

Feature	Gain	Cover	Frequency
GrLivArea	0.1656607	0.0584588	0.0453250
TotalBsmtSF	0.0915029	0.0479665	0.0447286
YearBuilt	0.0822997	0.0304236	0.0330661
GarageArea	0.0711117	0.0506462	0.0509575
BsmtFinSF1	0.0280309	0.0420401	0.0502286
X1stFlrSF	0.0275991	0.0273099	0.0298854
LotArea	0.0220111	0.0424669	0.0542708
BsmtUnfSF	0.0140726	0.0423503	0.0566563
WoodDeckSF	0.0095376	0.0346671	0.0350540
OpenPorchSF	0.0067967	0.0220304	0.0297528

While the following have very low gain, and do intuitively contribute to uninteresting features of a property:

```
## Selecting by Frequency
```

Feature	Gain	Cover	Frequency
HeatingGrav	0.0003463	0.0000204	6.63e-05
HouseStyle2.5Unf	0.0000462	0.0000214	6.63e-05
MSSubClass75	0.0000376	0.0000143	6.63e-05
SaleConditionAdjLand	0.0000110	0.0000188	6.63e-05
NeighborhoodTimber	0.0000070	0.0000239	6.63e-05
FunctionalMin2	0.0000057	0.0000127	6.63e-05
HouseStyle1.5Unf	0.0000050	0.0001961	6.63e-05
LandContourLow	0.0000042	0.0002292	6.63e-05
NeighborhoodBrDale	0.0000038	0.0000097	6.63e-05
MSSubClass85	0.0000036	0.0000255	6.63e-05
HeatingGasW	0.0000032	0.0000153	6.63e-05
HeatingWall	0.0000019	0.0000250	6.63e-05
MSSubClass180	0.0000004	0.0000046	6.63e-05

Results

I have defined my own root mean squared error loss function, as per the definition in the course textbook, and have used this to comparatively analyise the outcome of the four models:

```r
RMSE <- function(predicted, observed){
  sqrt(mean((observed - predicted)^2))
}
```

The result of these models is then compared, using the RMSE Function above, to the testing set. This demonstrates that each modelling iteration was more successful than the last. It does however suggest to me that there is a lot more potential in the linear modelling as adding an additional term was highly significant.

Prediction Method	RSME
Cart Model	0.1809110
Linear Regression	0.1672365
Linear Regression, with Suburb	0.1461783
Extreme Gradient Boosting	0.1309226

Conclusion

The best predictive model produced RSME = 0.13, I know from looking at some high performing kaggle competitions it's posible to acheive a result below $0.10 < RSME < 0.11$, however the users who wrote those scripts seemed to have a much deeper understanding the modelling of data than I did. There seemed to be opportunity to improve in the engineering of new features, the adjusting of skewedness in the features of the dataset, and the use of other packages, most noteably gtlmnet.

I found a lot of limitations, and consequently a lot of opportunity, for future work throughout the process which was mainly due to my inexperience. While I was attempting to perform a modelling step I was also simultaneously learning both conceptually what I was trying to acheive and grappling with the technicalities of implementation. Althought our course was quite comprehensive I realised that there are opportunities for learning about the treatment of missing data in sophisticated ways, opportunities to investigate features and derive new and meaingful ones. Finally, there is an opportunity to explore both how to fit models to data and also how to use the modelling process to investigate the data. While I was able to extract the features which had a high 'Gain' and I could relate that to successful additions to the linear models, I believe I could use more advanced modelling to better unstand which features are import and why.