ECE-9603A Assignment 1

Joe Abley

Contents

Introduction

This document is a submission in response to Assignment 1 in the course ECE-9603A, Fall 2018, Western University Faculty of Engineering, Department of Electrical and Computer Engineering. It has been written in R Markdown, and the document source can be found in this GitHub repository.

Packages

```r
library(e1071)
library(randomForest)
```

```
## randomForest 4.6-14
```

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

```r
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

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

```r
library(scales)
library(ggrepel)
library(Metrics)
```

Forecasting Problem

Given a dataset that describes various features of individual houses along with details of their sale, identify forecasting models that are able to predict the prices realised by the sale of houses at particular times.

Available Data

We make use of a dataset that describes house sales in Iowa, published on and retrieved from Kaggle as part of a Kaggle competition entitled "House Prices: Advanced Regression Techniques". This data set was suggested in the assignment directions.

Importing Data

```r
train <- read.csv("train.csv")
test <- read.csv("test.csv")
```

Abridged Feature Engineering

```r
dim(train)
```

```
## [1] 1460   81
```

There are 1460 rows in this dataset and 81 columns. Of those columns one is a numeric id and one is the sale price; the other 79 are parameters that describe each house, some of which are numeric variables and some of which are categories.

From the description provided with the source data, the numeric variables are as follows:

Variable Name	Description
`LotFrontage`	Linear feet of street connected to property
`LotArea`	Lot size in square feet
`MasVnrArea`	Masonry veneer area in square feet
`BsmtFinSF2`	Type 2 finished square feet
`BsmtUnfSF`	Unfinished square feet of basement area
`TotalBsmtSF`	Total square feet of basement area
`1stFlrSF`	First Floor square feet
`2ndFlrSF`	Second floor square feet
`LowQualFinSF`	Low quality finished square feet (all floors)

Variable Name	Description
`GrLivArea`	Above grade (ground) living area square feet
`BsmtFullBath`	Basement full bathrooms
`BsmtHalfBath`	Basement half bathrooms
`FullBath`	Full bathrooms above grade
`HalfBath`	Half baths above grade
`Bedroom`	Bedrooms above grade (does NOT include basement bedrooms)
`Kitchen`	Kitchens above grade
`TotRmsAbvGrd`	Total rooms above grade (does not include bathrooms)
`Fireplaces`	Number of fireplaces
`GarageCars`	Size of garage in car capacity
`GarageArea`	Size of garage in square feet
`WoodDeckSF`	Wood deck area in square feet
`OpenPorchSF`	Open porch area in square feet
`EnclosedPorch`	Enclosed porch area in square feet
`3SsnPorch`	Three season porch area in square feet
`ScreenPorch`	Screen porch area in square feet
`PoolArea`	Pool area in square feet
`MiscVal`	$Value of miscellaneous feature

The stated purpose of this assignment is "to experiment with different models"" and its focus is "applying forecasting approaches and not on optimising models". In the spirit of that direction we will not complete a detailed feature analysis and instead will select a set of numeric samples that seem likely to be sufficiently representative to give some kind of correlation, based on general background knowledge gained buying and selling houses in places other than Iowa, because how different can people from Iowa be? A smaller set of features seems helpful.

We can construct a new variable `newBathrooms`, derived from the various other bathroom variables:

- `newBathrooms` (Total number of bathrooms, full and half, all levels) = `BsmtFullBath` + `BsmtHalfBath` + `FullBath` + `HalfBath`

```
train$newBathrooms = train$BsmtFullBath + train$BsmtHalfBath + train$FullBath + train$HalfBath
```

We can eliminate some anticipated redundancy by identifying variables that seem likely to be closely related, and arbitrarily choosing the one that seems most interesting.

- `LotFrontage` and `LotArea` both relate to the size of the lot, which seems pertinent. Retain `LotArea`.
- `BsmtFinSF2`, `BsmtUnfSF` and `TotalBsmtSF` all relate to the size of the basement. Retain `TotalBsmtSF`.
- `1stFlrSF`, `2ndFlrSF`, `LowQualFinSF` and `GrLivArea` all relate to the size of the rest of the house. Retain `GrLivArea`.
- `Bedroom`, `Kitchen` and `TotRmsAbvGrd` all relate to the number of rooms above the basement. Retain `TotRmsAbvGrd`.
- `GarageCars` and `GarageArea` both relate to the size of the garage. Retain `GarageArea`.

We can keep some variables as-is, because they seem harmless and potentially interesting:

- `Fireplaces`

We arbitrarily declare all remaining variables to be uninteresting. We take care to retain `SalePrice` which is our outcome/response variable.

```
interesting <- c("newBathrooms", "LotArea", "TotalBsmtSF", "GrLivArea", "TotRmsAbvGrd",
  "Fireplaces", "GarageArea", "SalePrice")
train <- train[, (names(train) %in% interesting)]
```

To avoid surprises, we check for variables that might have missing data. Fortunately we seem not to have any.

```r
which(colSums(is.na(train)) > 0)
```

```
## named integer(0)
```

Our cauterised training data set now looks like this:

```r
summary(train)
```

```
##     LotArea        TotalBsmtSF       GrLivArea     TotRmsAbvGrd
##  Min.   :  1300   Min.   :   0.0   Min.   : 334   Min.   : 2.000
##  1st Qu.:  7554   1st Qu.: 795.8   1st Qu.:1130   1st Qu.: 5.000
##  Median :  9478   Median : 991.5   Median :1464   Median : 6.000
##  Mean   : 10517   Mean   :1057.4   Mean   :1515   Mean   : 6.518
##  3rd Qu.: 11602   3rd Qu.:1298.2   3rd Qu.:1777   3rd Qu.: 7.000
##  Max.   :215245   Max.   :6110.0   Max.   :5642   Max.   :14.000
##    Fireplaces      GarageArea       SalePrice      newBathrooms
##  Min.   :0.000   Min.   :   0.0   Min.   : 34900   Min.   :1.000
##  1st Qu.:0.000   1st Qu.: 334.5   1st Qu.:129975   1st Qu.:2.000
##  Median :1.000   Median : 480.0   Median :163000   Median :2.000
##  Mean   :0.613   Mean   : 473.0   Mean   :180921   Mean   :2.431
##  3rd Qu.:1.000   3rd Qu.: 576.0   3rd Qu.:214000   3rd Qu.:3.000
##  Max.   :3.000   Max.   :1418.0   Max.   :755000   Max.   :6.000
```

Finally, we transform and reduce our test data set in the same way, since keeping it the same seems less likely to be confusing.

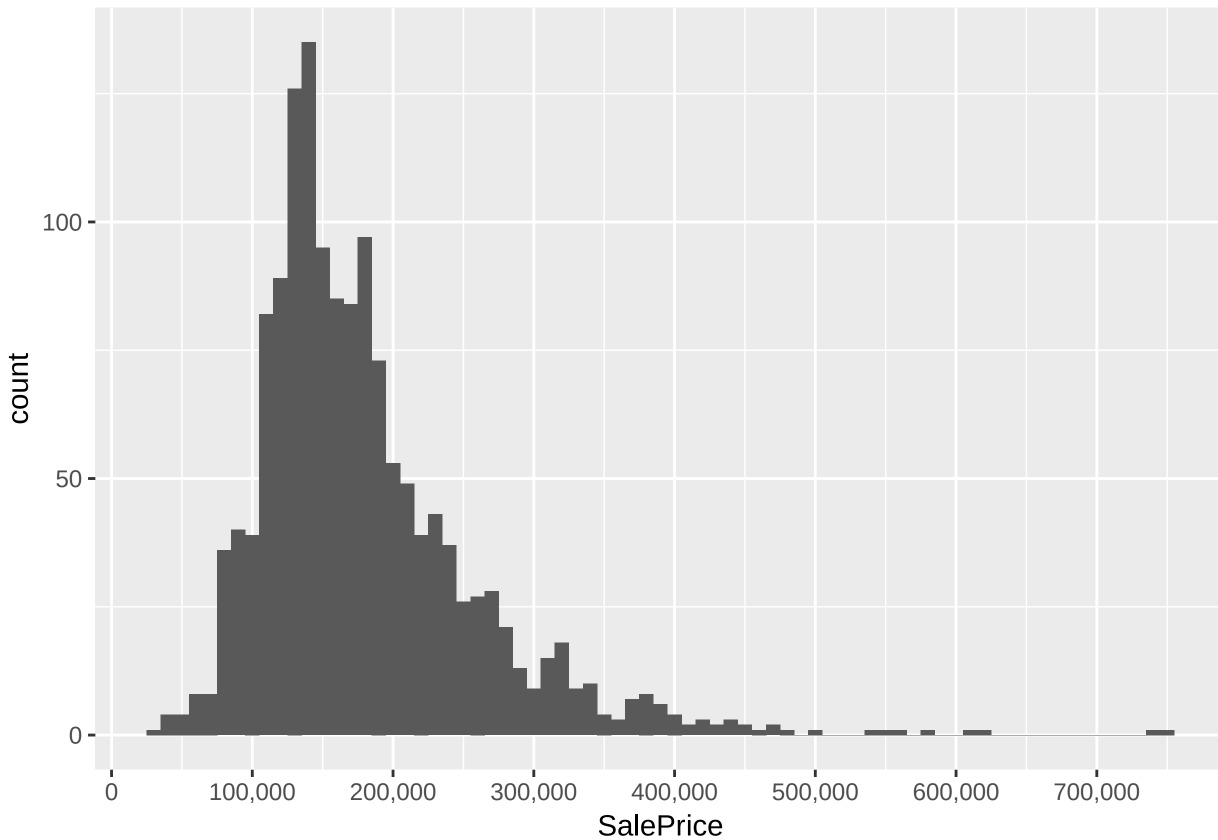
```r
test$newBathrooms = test$BsmtFullBath + test$BsmtHalfBath + test$FullBath + test$HalfBath
test <- test[, (names(test) %in% interesting)]
```

Data Inspection

SalePrice

The distribution of sale prices is not symmetrical; there are more houses sold at lower prices and a long tail of expensive houses as is shown in the following histogram.

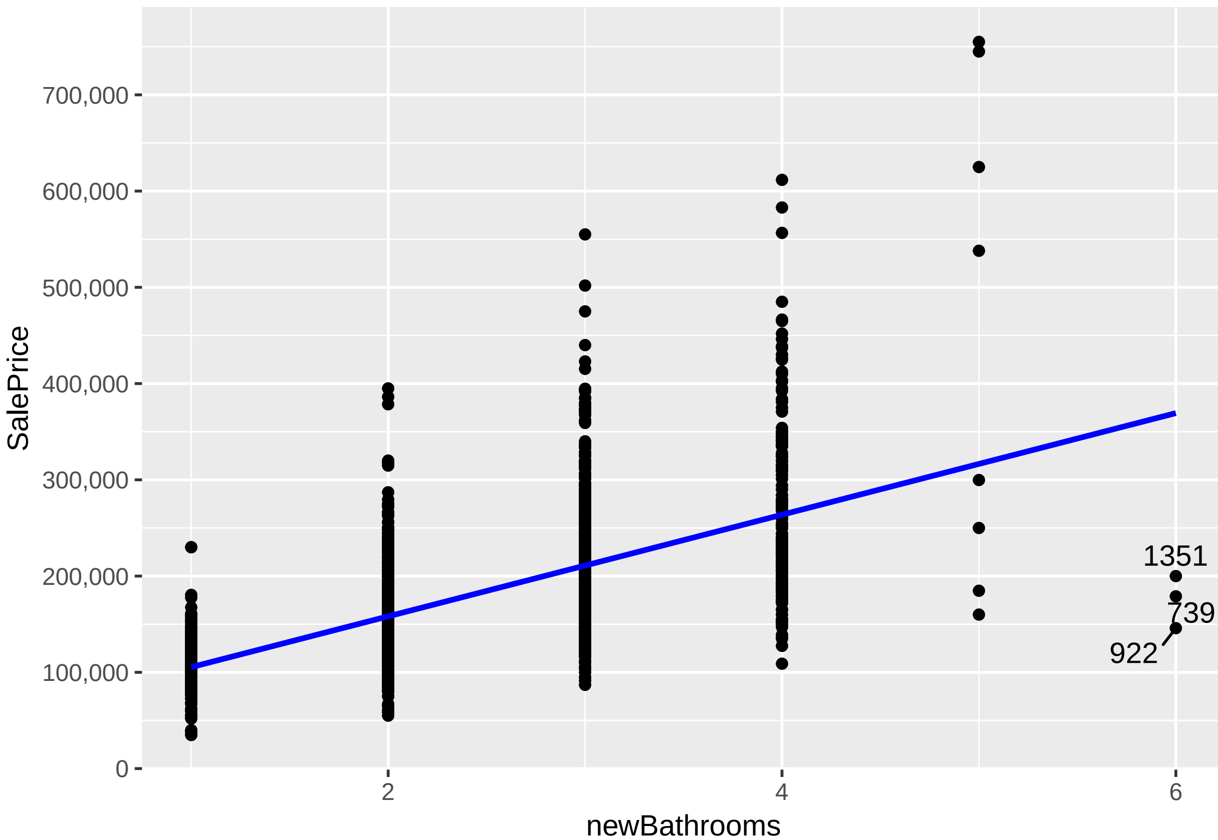
```
ggplot(data=train[!is.na(train$SalePrice),], aes(x = SalePrice)) +
  geom_histogram(binwidth = 10000) +
  scale_x_continuous(breaks = seq(0, 800000, by = 100000), labels = comma)
```



newBathrooms

The total number of bathrooms seems to correlate to `SalePrice`, although there are a small number of outliers that suggest that at some point you really don't get much value from adding more toilets.

```
ggplot(data=train[!is.na(train$SalePrice),], aes(x=newBathrooms, y=SalePrice)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, colour = "blue", aes(group = 1)) +
  scale_y_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
  geom_text_repel(aes(label = ifelse(train$newBathrooms[!is.na(train$SalePrice)] > 5,
    rownames(train), '')))
```

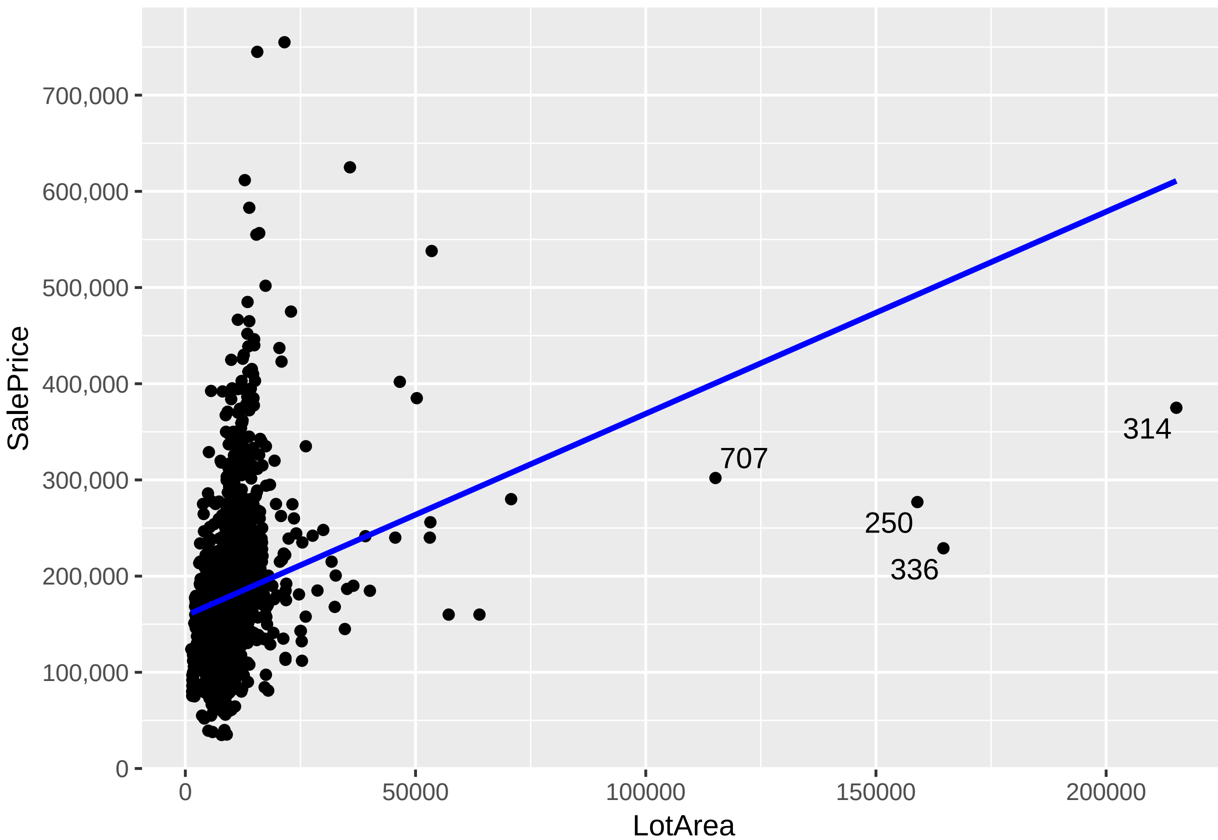


The houses with six bathrooms don't seem to fit a linear relationship very well; they are labelled in the graph above in case we need some persistent troublemakers to eliminate as we build our models later.

LotArea

For many houses there seems to be a strong correlation between `LotArea` and `SalePrice`. As with the tentative toilet hypothesis, however, it seems possible that the size of the lot beyond a certain point just starts to seem more annoying to mow.

```
ggplot(data=train[!is.na(train$SalePrice),], aes(x=LotArea, y=SalePrice)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, colour = "blue", aes(group = 1)) +
  scale_y_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
  geom_text_repel(aes(label = ifelse(train$LotArea[!is.na(train$SalePrice)] > 100000,
    rownames(train), '')))
```

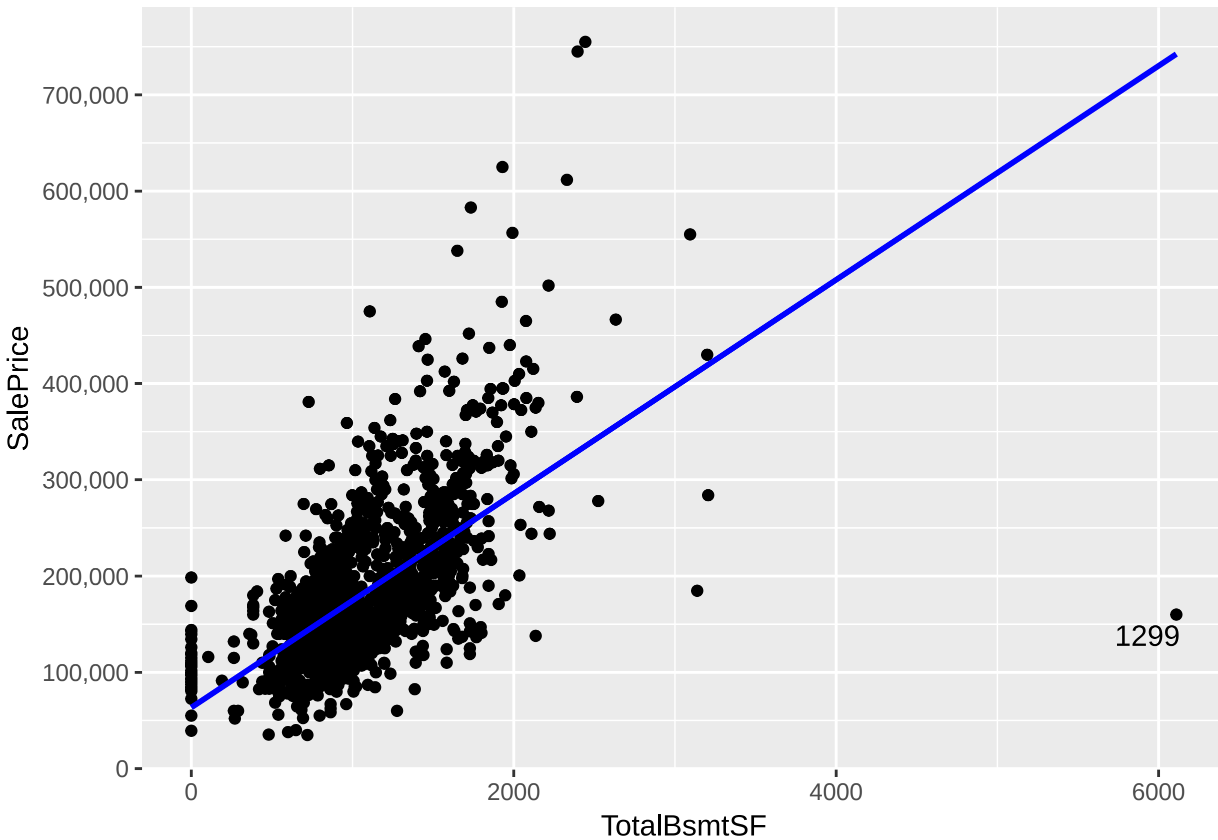


The properties with lot areas larger than 100,000 square feet have bee labelled as candidates for shunning.

TotalBsmtSF

A strong linear correlation is observed between `TotalBsmtSF` and `SalePrice`, with just a single outlier that we might imagine corresponds to a basement that is over-large for an unsavoury reason.

```
ggplot(data=train[!is.na(train$SalePrice),], aes(x=TotalBsmtSF, y=SalePrice)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, colour = "blue", aes(group = 1)) +
  scale_y_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
  geom_text_repel(aes(label = ifelse(train$TotalBsmtSF[!is.na(train$SalePrice)] > 4000,
    rownames(train), '')))
```

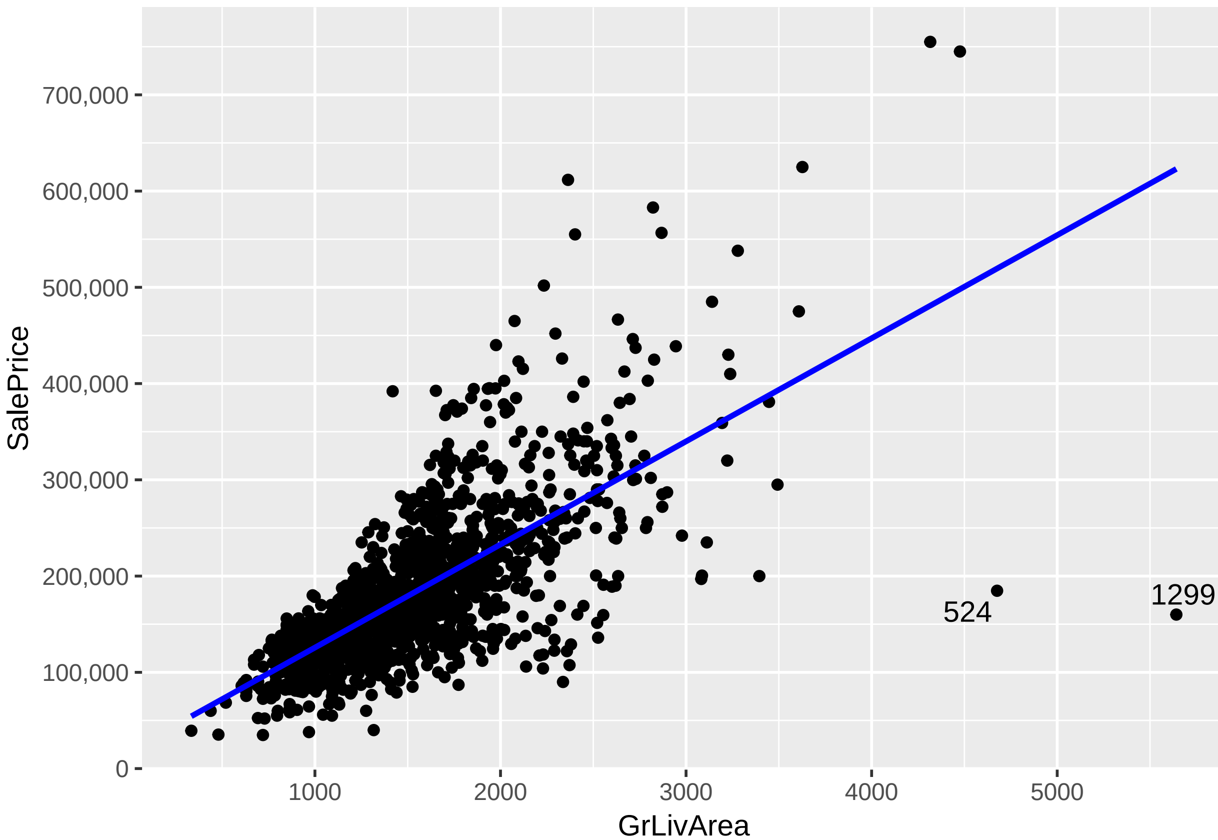


The problem basement has once again been labelled, above.

GrLivArea

There is a strong linear correlation observed between `GrLivArea` and `SalePrice`. The properties with a living area over 4,000 square feet seem to be outliers.

```r
ggplot(data=train[!is.na(train$SalePrice),], aes(x=GrLivArea, y=SalePrice)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, colour = "blue", aes(group = 1)) +
  scale_y_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
  geom_text_repel(aes(label = ifelse(train$GrLivArea[!is.na(train$SalePrice)] > 4500,
    rownames(train), '')))
```

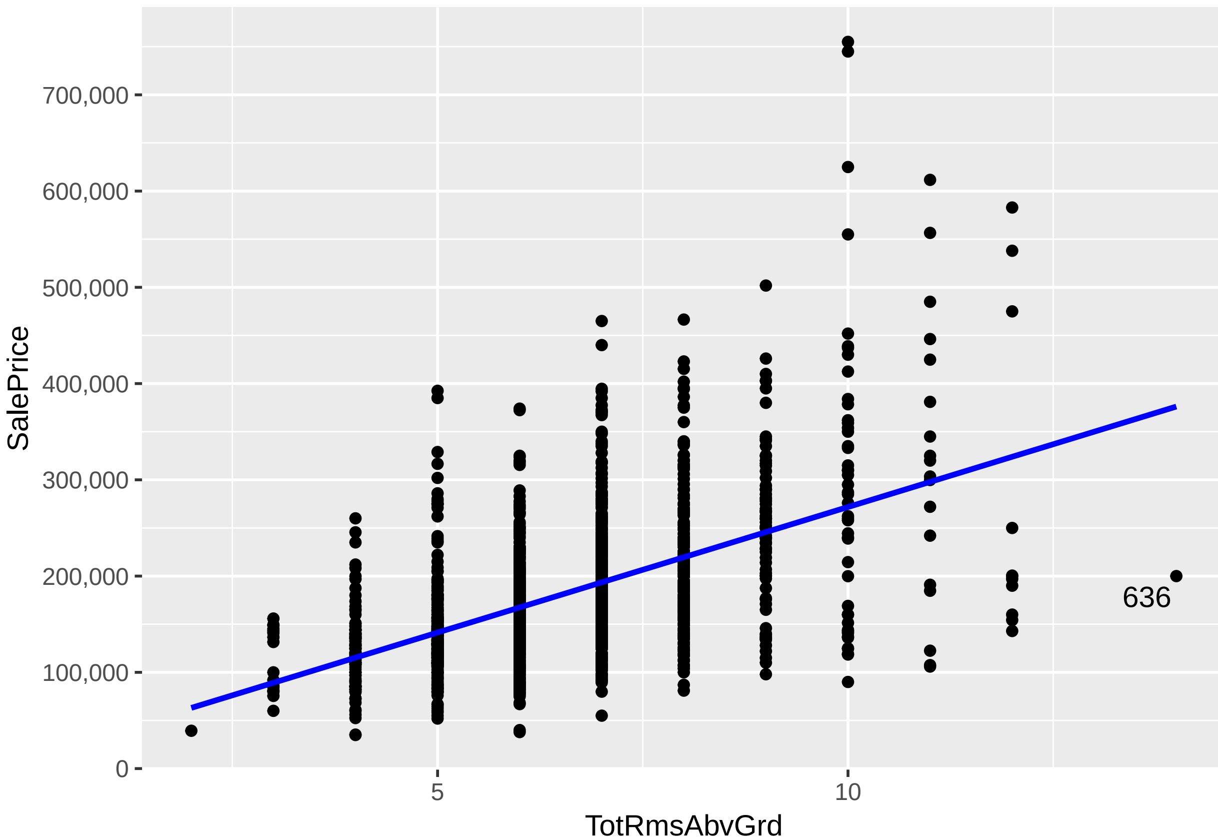


Since the outliers are somewhat evenly balanced above and below the line resulting from simple linear regression they do not appear to affect the legitimacy of the relationship; eliminating them would presumably not affect it either, however, so they are labelled.

TotRmsAbvGrd

There is a strong linear correlation observed between `TotRmsAbvGrd` and `SalePrice`. There is a single property with 14 rooms that seems excessive and strange.

```
ggplot(data=train[!is.na(train$SalePrice),], aes(x=TotRmsAbvGrd, y=SalePrice)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, colour = "blue", aes(group = 1)) +
  scale_y_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
  geom_text_repel(aes(label = ifelse(train$TotRmsAbvGrd[!is.na(train$SalePrice)] > 13,
    rownames(train), '')))
```

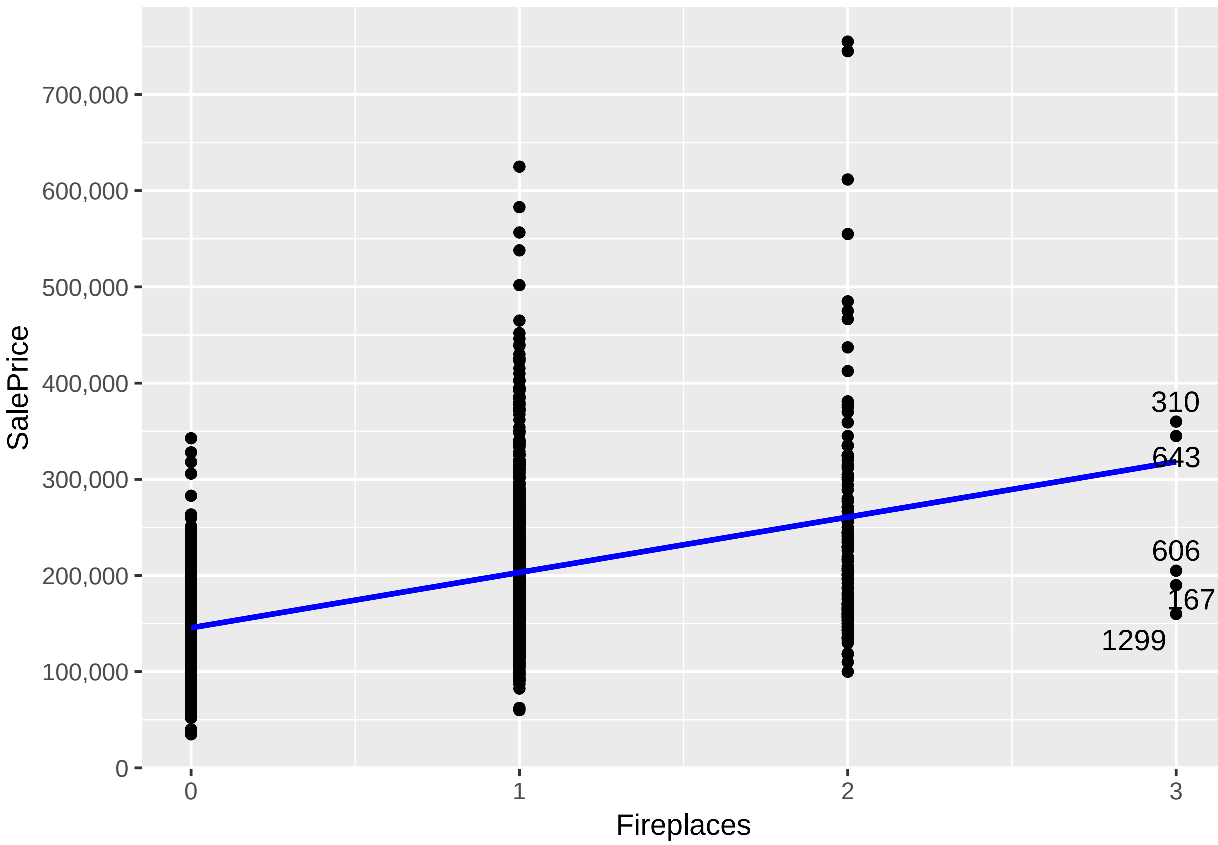


The strange outlier has been labelled.

Fireplaces

The relevance of `Fireplaces` does not seem entirely clear, although a slight upward trend is certainly observed. People in Iowa like to burn things, but not *that* much.

```
ggplot(data=train[!is.na(train$SalePrice),], aes(x=Fireplaces, y=SalePrice)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, colour = "blue", aes(group = 1)) +
  scale_y_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
  geom_text_repel(aes(label = ifelse(train$Fireplaces[!is.na(train$SalePrice)] > 2,
    rownames(train), '')))
```

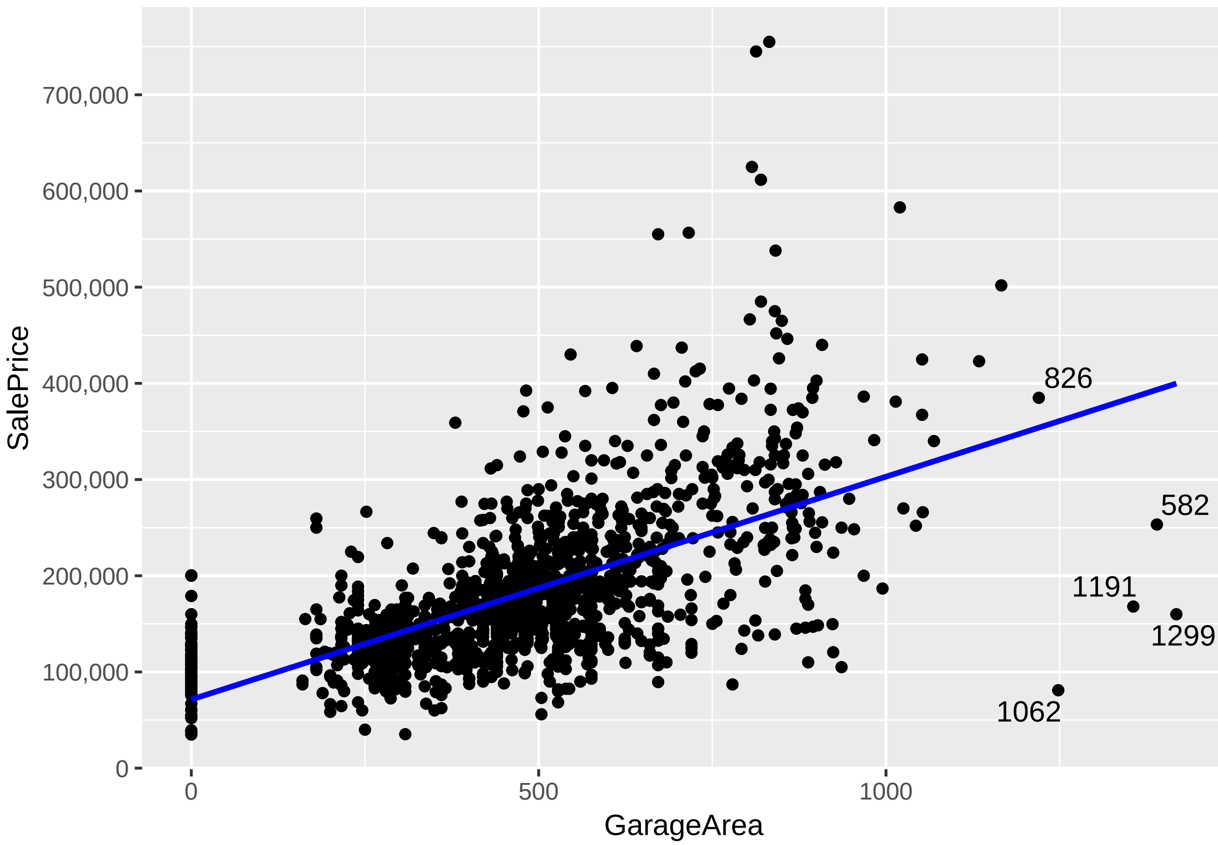


The pyromaniac paradises have been labelled.

GarageArea

Properties with more garage space seem to command higher prices. The extremely large garages seem to have a lower impact on price.

```
ggplot(data=train[!is.na(train$SalePrice),], aes(x=GarageArea, y=SalePrice)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, colour = "blue", aes(group = 1)) +
  scale_y_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
  geom_text_repel(aes(label = ifelse(train$GarageArea[!is.na(train$SalePrice)] > 1200,
    rownames(train), '')))
```



The batcaves have been labelled.

Elimination of Outliers

For the sake of simplicity we remove all the labelled properties in the previous section. There aren't that many of them, and it seems plausible that they all have some odd characteristic that has skewed their sale price and hence will not contribute productively to our modelling.

```
train <- train[-c(167, 250, 310, 314, 336, 363, 524, 582, 606, 643, 692, 707, 739, 826,
  922, 1062, 1183, 1191, 1299, 1351)]
```

Our final training set contains 1460 observations and just seven parameters, all of which appear to plausibly fit a linear relationship with `SalePrice`:

```
dim(train)
```

```
## [1] 1460    8
```

```
summary(train)
```

```
##     LotArea        TotalBsmtSF       GrLivArea     TotRmsAbvGrd
##  Min.   :  1300   Min.   :   0.0   Min.   : 334   Min.   : 2.000
##  1st Qu.:  7554   1st Qu.: 795.8   1st Qu.:1130   1st Qu.: 5.000
##  Median :  9478   Median : 991.5   Median :1464   Median : 6.000
##  Mean   : 10517   Mean   :1057.4   Mean   :1515   Mean   : 6.518
##  3rd Qu.: 11602   3rd Qu.:1298.2   3rd Qu.:1777   3rd Qu.: 7.000
##  Max.   :215245   Max.   :6110.0   Max.   :5642   Max.   :14.000
##    Fireplaces       GarageArea       SalePrice       newBathrooms
##  Min.   :0.000   Min.   :   0.0   Min.   : 34900   Min.   :1.000
##  1st Qu.:0.000   1st Qu.: 334.5   1st Qu.:129975   1st Qu.:2.000
##  Median :1.000   Median : 480.0   Median :163000   Median :2.000
##  Mean   :0.613   Mean   : 473.0   Mean   :180921   Mean   :2.431
##  3rd Qu.:1.000   3rd Qu.: 576.0   3rd Qu.:214000   3rd Qu.:3.000
##  Max.   :3.000   Max.   :1418.0   Max.   :755000   Max.   :6.000
```

Selected Algorithms

Multivariate Regression

Multivariate linear regression is a method of supervised regression, used to predict a numerical outcome from a set of observations. In this exercise we have identified seven features (`LotArea`, `TotalBsmtSF`, `GrLivArea`, `TotRmsAbvGrd`, `Fireplaces`, `GarageArea` and `newBathrooms`) and have eliminated a small number of outliers with the result that each of those features is observed to have a (different) linear relationship with `SalePrice`. Consequently, we will build and test a multivariate regression model with no further transforms and assess its goodness of fit.

```
modelLR <- lm(SalePrice ~ LotArea + TotalBsmtSF + GrLivArea +
  TotRmsAbvGrd + Fireplaces + GarageArea + newBathrooms, train)
summary(modelLR)
```

```
##
## Call:
## lm(formula = SalePrice ~ LotArea + TotalBsmtSF + GrLivArea +
##     TotRmsAbvGrd + Fireplaces + GarageArea + newBathrooms, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -609201  -19719    -710   17683  282847
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.782e+04  5.470e+03  -5.085 4.15e-07 ***
## LotArea       6.984e-02  1.225e-01   0.570    0.569
## TotalBsmtSF   4.500e+01  3.233e+00  13.917  < 2e-16 ***
## GrLivArea     4.913e+01  4.801e+00  10.232  < 2e-16 ***
## TotRmsAbvGrd -8.584e+02  1.280e+03  -0.670    0.503
## Fireplaces    1.276e+04  2.074e+03   6.151 9.93e-10 ***
## GarageArea    9.035e+01  6.605e+00  13.679  < 2e-16 ***
## newBathrooms  1.687e+04  1.621e+03  10.409  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 43930 on 1452 degrees of freedom
## Multiple R-squared:  0.6957, Adjusted R-squared:  0.6943
## F-statistic: 474.3 on 7 and 1452 DF,  p-value: < 2.2e-16
```

```
predictLR <- predict(modelLR, test, type = "response")
outputLR <- cbind(test, predictLR)
rmse(outputLR$SalePrice, outputLR$prediction)
```

```
## [1] NaN
```

Support Vector Regression

Support Vector Regression (SVR) is another method of supervised regression. SVR is an adaptation of Support Vector Machines for function estimation, and is built around analogous hyperparameters, of which we are principally concerned with the soft margin loss setting ϵ, an acceptable error in the resulting regression model.

```
modelSVR <- svm(SalePrice ~ LotArea + TotalBsmtSF + GrLivArea +
  TotRmsAbvGrd + Fireplaces + GarageArea + newBathrooms, train)
summary(modelSVR)
```

```
##
## Call:
## svm(formula = SalePrice ~ LotArea + TotalBsmtSF + GrLivArea +
##     TotRmsAbvGrd + Fireplaces + GarageArea + newBathrooms, data = train)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.1428571
##     epsilon:  0.1
##
##
## Number of Support Vectors:  1030
```

Random Forest

Random Forest is an algorithm that uses many decision trees and makes predictions based on the average predicted values resulting from each component tree. This is intended to result in better accuracy than using a single tree.

Accuracy Comparison

Hold-out or cross-validation