Math 218: Project Proposal

Team Haydoja

Payoja Adhikari and Hayden Hunskor

11/13/2022

```
library(tidyverse)
library(dplyr)
library(tidyr)
library(ggcorrplot)
```

Data description

We picked the Ames Housing dataset compiled by Dean De Cock, which is publicly available on Kaggle. The dataset includes extensive information on the sale of residential properties in Ames, Iowa from 2006 to 2010. It contains 1460 observations with 79 variables on various house specifications. The variables are a mix of continuous, categorical and discrete types. Here is a link to the dataset itself which includes a description of all the data fields:

https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data

Define variables you want to explore

Rather than list out all 79 variables from our dataset, we selected a few that we found particularly interesting and listed them below. That said, we are interested in starting out with models that include all available predictors and applying various shrinkage techniques.

Categorical Variable:

- Street: Type of road access to property (Gravel/Paved)
- Utilities: Type of Utilities available
- Neighborhood: Physical locations within Ames city limits
- Heating: Type of heating
- CentralAir: Central air conditioning (Yes/No)
- PavedDrive: Paved driveway
- GarageFinish: Interior finish of the garage
- SaleType: Type of sale
- SaleCondition: Condition of sale

Continuous Variable:

- SalePrice: Price of house
- PoolArea: Pool area in square feet
- Lot Area: Lot size in square feet
- TotalBsmtSF: Total square feet of basement area
- Garage Area :Size of garage in square feet
- GrLivArea: Above ground living area square feet

Discrete Variable

- FullBath: Number of full bathrooms above ground
- HalfBath: Number of half bathrooms above ground
- BedroomAbvGr: Number of bedrooms above ground
- KitchensAbvGr: Number of kitchens above ground
- FirePlaces: Number of fireplaces
- YrBuilt: Original construction date
- YrSold: Year house was sold

EDA

The Kaggle zip file included both a train and a test set. The key difference between the two is that only the train set has the SalePrice data field. The following code chunk loads in the two data sets and then combines them into one data frame with NA values for testing SalePrice data.

```r
#Read in data
train <- read_csv("data/train.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   Id = col_double(),
##   MSSubClass = col_double(),
##   LotFrontage = col_double(),
##   LotArea = col_double(),
##   OverallQual = col_double(),
##   OverallCond = col_double(),
##   YearBuilt = col_double(),
##   YearRemodAdd = col_double(),
##   MasVnrArea = col_double(),
##   BsmtFinSF1 = col_double(),
##   BsmtFinSF2 = col_double(),
##   BsmtUnfSF = col_double(),
##   TotalBsmtSF = col_double(),
##   ‘1stFlrSF‘ = col_double(),
##   ‘2ndFlrSF‘ = col_double(),
##   LowQualFinSF = col_double(),
##   GrLivArea = col_double(),
##   BsmtFullBath = col_double(),
##   BsmtHalfBath = col_double(),
##   FullBath = col_double()
##   # ... with 18 more columns
## )
## i Use ‘spec()‘ for the full column specifications.
```

```
test <- read_csv("data/test.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   Id = col_double(),
##   MSSubClass = col_double(),
##   LotFrontage = col_double(),
##   LotArea = col_double(),
##   OverallQual = col_double(),
##   OverallCond = col_double(),
##   YearBuilt = col_double(),
##   YearRemodAdd = col_double(),
##   MasVnrArea = col_double(),
##   BsmtFinSF1 = col_double(),
##   BsmtFinSF2 = col_double(),
##   BsmtUnfSF = col_double(),
##   TotalBsmtSF = col_double(),
##   ‘1stFlrSF‘ = col_double(),
##   ‘2ndFlrSF‘ = col_double(),
##   LowQualFinSF = col_double(),
##   GrLivArea = col_double(),
##   BsmtFullBath = col_double(),
##   BsmtHalfBath = col_double(),
##   FullBath = col_double()
##   # ... with 17 more columns
## )
## i Use ‘spec()‘ for the full column specifications.
```

```
#Add NA values for SalePrice to test set
test$SalePrice <- rep(NA, nrow(test))

#Join train and test set
all_data <- rbind(test, train)
```

The following code chunk isolates numeric and categorical variables:

```
#Find numeric variables
numeric_var <- names(all_data)[which(sapply(train, is.numeric))]

#Find categorical variables
```

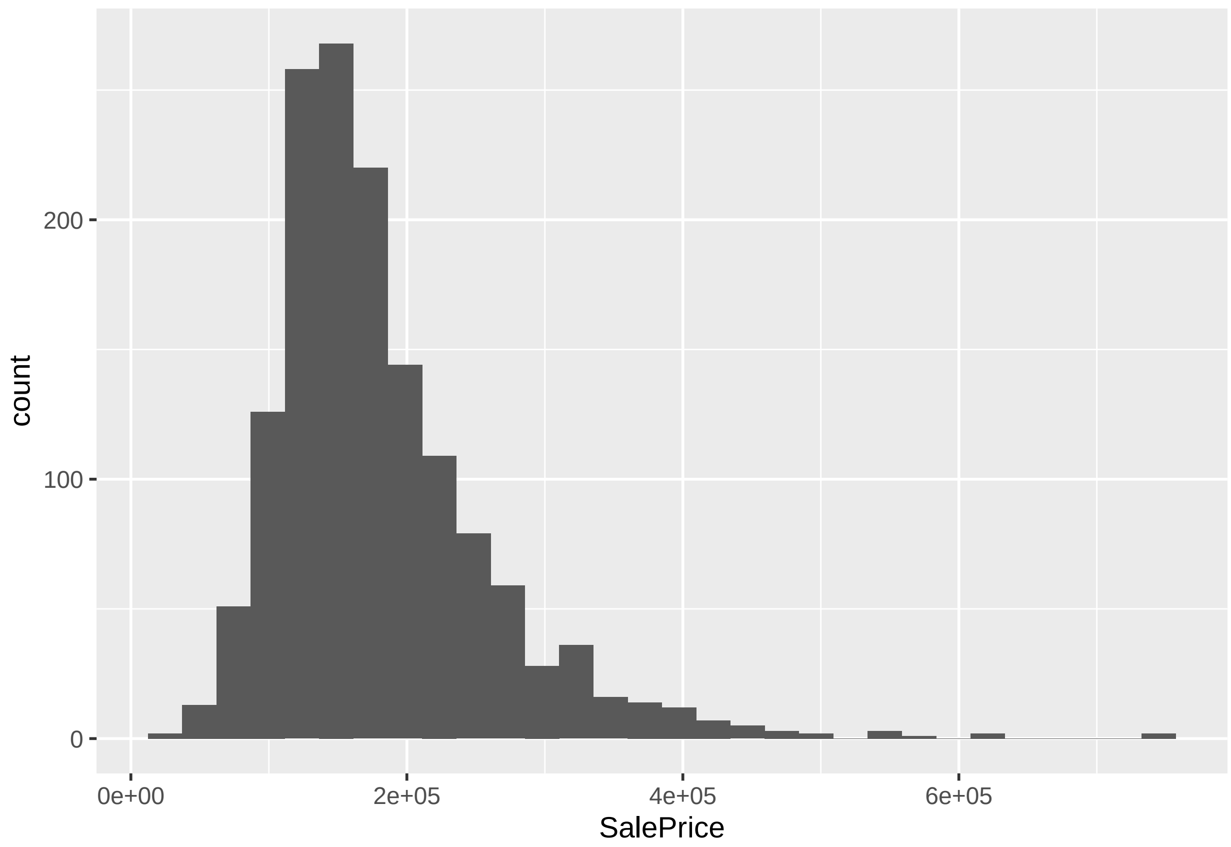
```r
categorical_var <- names(all_data)[-which(sapply(train, is.numeric))]

#Numeric variables that we want to work with
var <- numeric_var[-c(1:3,5:6,8,10:12,16,18:19,27,35:36)]

#Summary table for numeric variables
all_data[var] %>%
  summarise_all(list(mean = mean, std = sd), na.rm = TRUE) %>%
  pivot_longer(cols = everything(),
               names_to = c('col', '.value'),
               names_sep = '_')
```

```
## # A tibble: 23 x 3
##    col                mean      std
##    <chr>             <dbl>    <dbl>
##  1 LotArea      10168.    7887.
##  2 YearBuilt     1971.      30.3
##  3 MasVnrArea     102.     179.
##  4 TotalBsmtSF   1052.     441.
##  5 1stFlrSF      1160.     392.
##  6 2ndFlrSF       336.     429.
##  7 GrLivArea     1501.     506.
##  8 FullBath         1.57     0.553
##  9 HalfBath         0.380    0.503
## 10 BedroomAbvGr     2.86     0.823
## # ... with 13 more rows
```

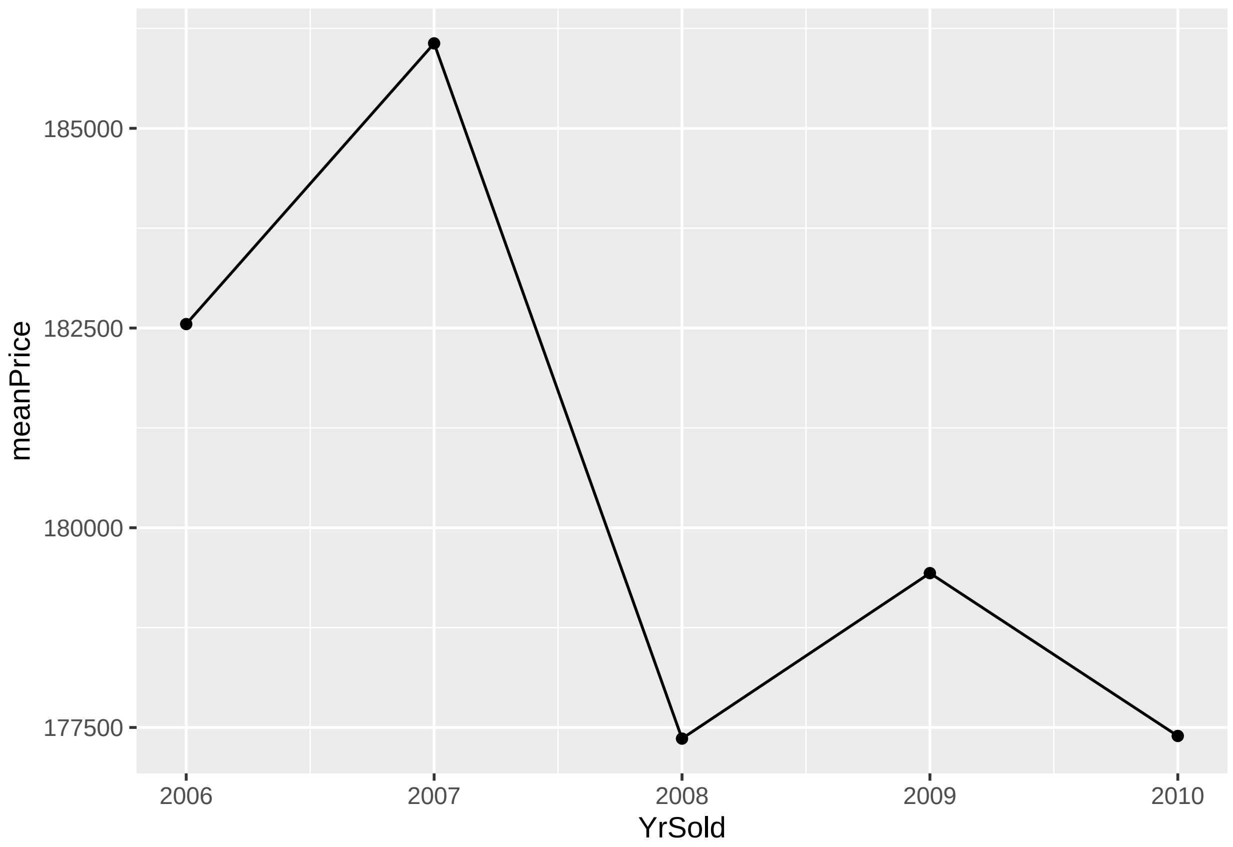
The following code chunk takes a deeper look at the SalePrice feature:

```r
#Histogram of house prices
train %>%
  ggplot(., aes(x = SalePrice)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
#Average house price over time
train %>%
  group_by(YrSold) %>%
  summarise(meanPrice = mean(SalePrice)) %>%
  ggplot(., aes(x = YrSold, y = meanPrice)) +
  geom_point() + geom_line()
```

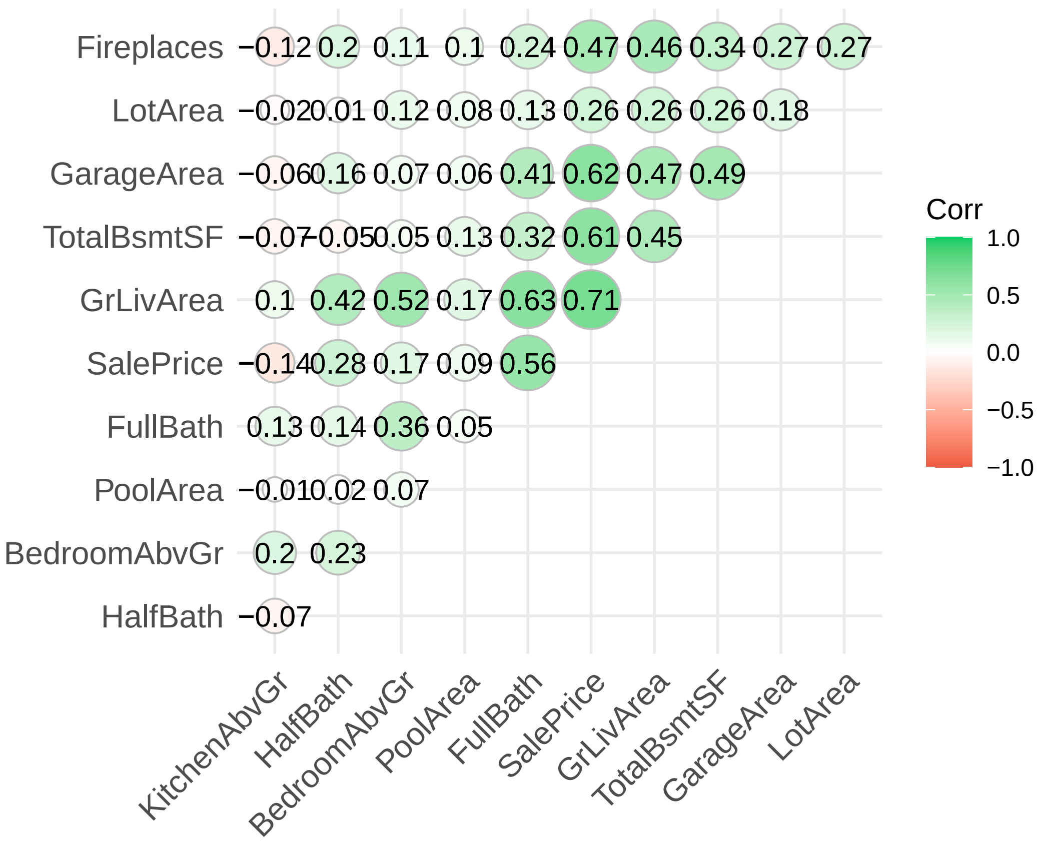


Judging from these two plots, the distribution of SalePrice in our training set is slightly right skewed. This suggests that there are some outlier properties with significantly higher sale prices.

Also, the huge drop in average sale price of houses in our data coincides with the 2008 housing market crash. When predicting sale price it may be helpful to try to account for some of the macroeconomic factors at play, such as the general state of the housing market at a national level. This may be beyond the scope of our work, but definitely something worth addressing in our project limitations.

Correlation between some of the numeric and discrete data fields (most importantly SalePrice):

```
#Note that we are only using the training data here because it has actual values for SalePrice

correlation <- train %>%
  select(SalePrice, PoolArea, LotArea, TotalBsmtSF, GarageArea, GrLivArea, FullBath, HalfBath, BedroomAb

corr_mtx <- cor(correlation, use = "pairwise.complete.obs")

ggcorrplot(corr_mtx, # input the correlation matrix
           hc.order = TRUE,
           type = "upper",
           lab = TRUE,
           method = "circle",
           colors = c("tomato2", "white", "springgreen3"))
```



As seen from above, housing prices seem to be the most correlated (positively) with above ground living area (in sq. ft), total basement area(in sq. ft) and garage area (in sq. ft). The price is also moderately correlated with number of full bathrooms.

Potential Research Questions

- How have housing prices changed over time?
- How do housing prices differ in Ames, Iowa based on the different housing features?
- Which property data fields are the most accurate predictors of a house's sale price?

The main focus of this project will be applying all the various modeling techniques we've learned thus far to our data, and trying to find one that accurately predicts housing prices.

One additional modeling method we are considering using, which we have not learned in class, is XGBoost - this would obviously require some self-teaching.