Ames Iowa House Prediction Project

Jordan Kassof and Jose Torres

2/11/2018

Contents

Introduction

Buying a house is one of the largest financial decisions many people will make. So many factors go into someone's decision, but it can be hard to really explain why one house "felt right" and another didn't. We want to quantify the factors that add up to someone making the decision to purchase a house.

Data Description

We will be using the Ames, Iowa individual residential property sales data set freely available on Kaggle.com. The data set contains 2,930 observations with 79 explanatory variables. All observations occur between 2006 and 2010. It is important to remember that any conclusions drawn from this data can only be applied in Ames, Iowa. For more information on the data, or to download it yourself, visit https://www.kaggle.com/c/house-prices-advanced-regression-techniques.

See the codebook.txt file in the github repository for complete information about all variables.

Exploratory Analysis

Our exploratory analysis was focused on getting a sense for the (numerous) variables in the dataset. We wanted to understand the marginal distributions of individual variables, the relationship between those individual variables and the sale price, and the correlation between the variables themselves.

In `cleaner_script.R` (Appendix C), the training and test sets are cleaned up and plots of variable histograms and scatter/box-plots are created for Sale Price vs each variable. View the plots here.

There are several features of a home that are not present for all homes in the data set, most notably various area measurements for home that don't have those features. There are 14 indicator variables which were created and added to the data set. The purpose of these variables is to allow flexible fits for features which are missing in some homes.

To see the mathematical intent of these indicator variables, consider the univariate model Log(SalePrice) = For example, the plot below shoes the Log of Sale Price vs Log of Lot Frontage. ***Continue discussion later***

Questions of Interest

We focused on two approaches, one geared towards model performance and one towards model interpretability by one of the parties involved in a home purchase.

Interpretable Models

For the interpretable model approach, rather than just taking a handful of easily understood parameters and building a model, we wanted to take a different approach. There are many adages when it comes to home buying, we wanted to see which are the most true. The three ideas about what drives the price of a house that we looked at are as follows:

- Location, location, location!
  - For this model, we used parameters that are related to the physical location of the property. For example, neighborhood, zoning, frontage, lot size, etc.
- It's all about the curb appeal
  - For this model, we used parameters related to the external appearance of the property. For example, house style, roof style, external veneer materials, etc.
- It's what's on the inside that counts
  - For this model, we used parameters related to the internals of the property, the bones if you will. For example, the foundation, the electrical and heating system, etc.

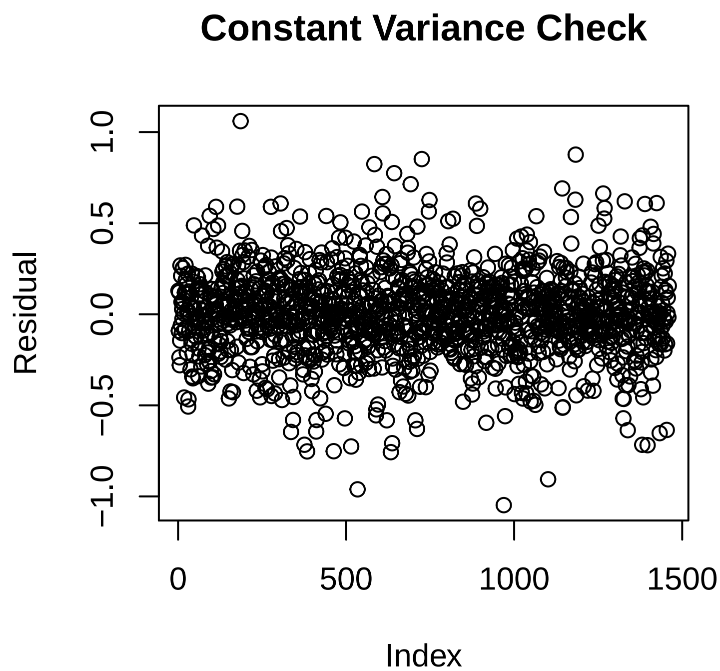
We also included the Sale Condition variable in all three models because the context of the sale seems like way too key of a factor to leave out of any model that is meant to be easily understood

Model Selection

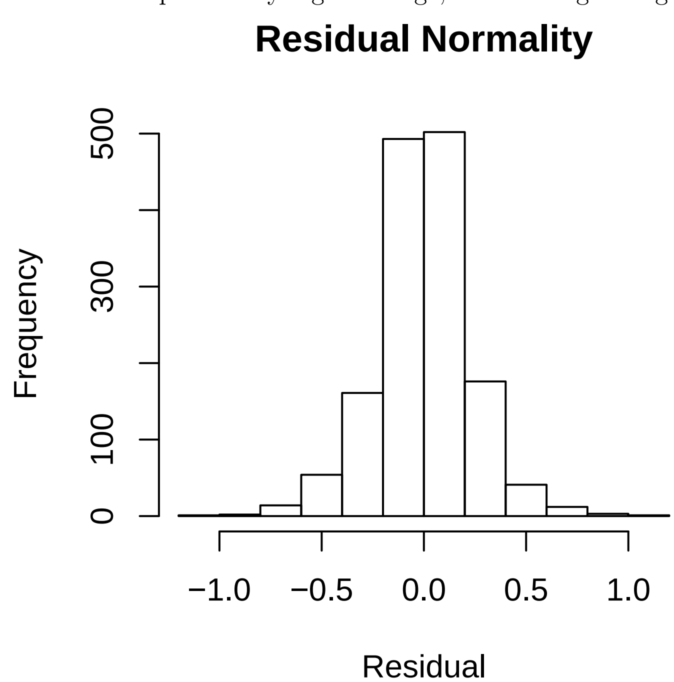
After running our three models, let's take a look at some diagnostics. After picking a model based on diagnostics, we will examine assumptions and parameters.

ModelName	adj.r.squared	AIC	BIC	df
location	0.6567534	-52.13605	185.7426	44
inside	0.6296121	81.06299	440.5240	67
outside	0.5964634	178.29170	384.4532	38

Based on R^2, AIC, and BIC, the best model appears to be the location model. Let's examine some diagnostic plots to make sure the assumptions of linear regression are met.



In the plot above, it appears there is not strong evidence against the assumption of constant variance in our residuals. I see a few points that have potentially high leverage, but nothing too egregious so we can proceed.

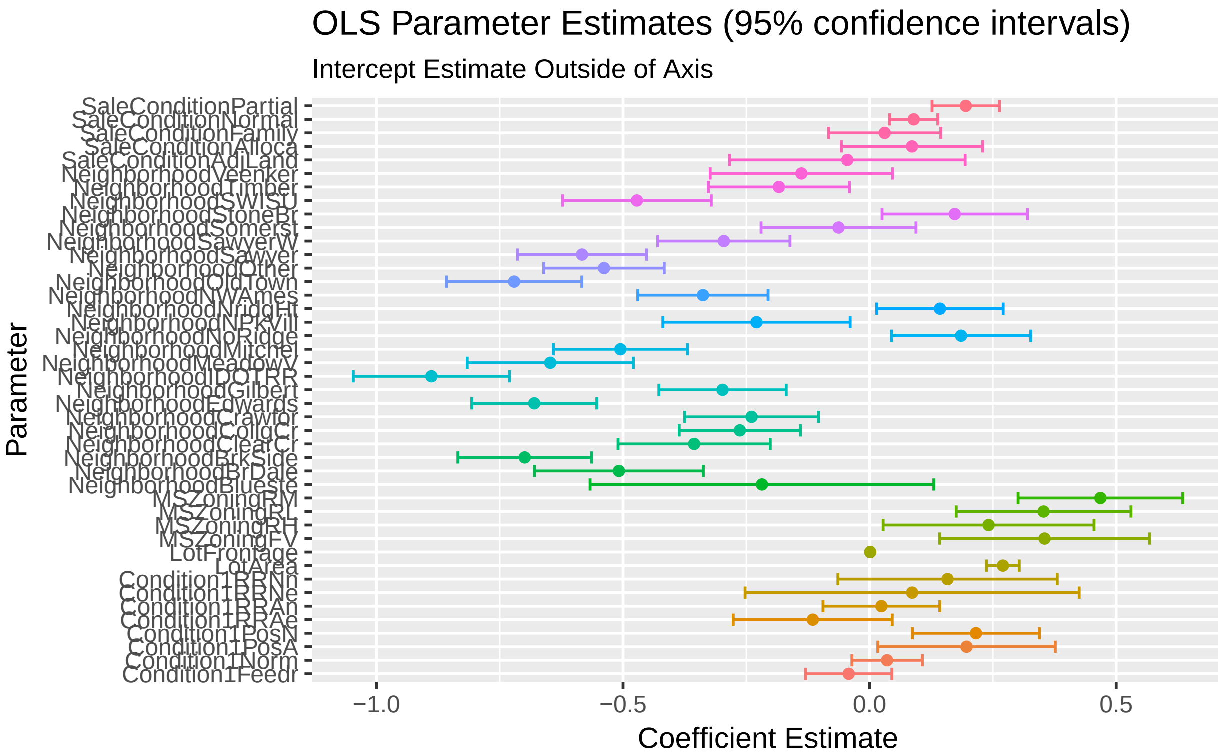


This histogram shows a symmetric distribution, and does not provide strong evidence against normality.

Our final assumption of independent observations, we will assume the data collection was conducted in a way that will provide independent observations. This assumption is probably the weakest as property values within a given city (and more so within a given neighborhood) are fairly dependent on each other. We will proceed with caution.

Parameter Interpretation

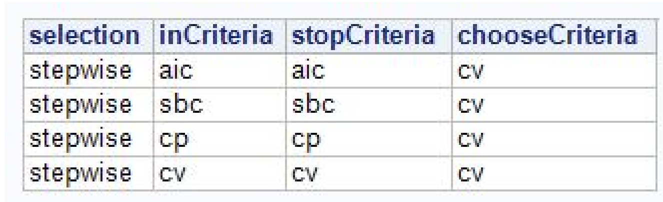
Below we will take a cursory look at all parameter estimates, then discuss a few of the parameters with the strongest influence on sale price.



The neighborhood variable has quite a lot of levels, but we can get an idea here for all the parameters estimates and a 95% confidence interval. Let's take a closer look at the five variables with the largest coefficient estimates.

term	estimate	std.error	statistic	p.value	conf.low	conf.high
MSZoningRM	0.4680163	0.0851141	5.498690	0.0000000	0.3010530	0.6349797
MSZoningFV	0.3547451	0.1085082	3.269294	0.0011041	0.1418910	0.5675992
MSZoningRL	0.3527361	0.0903168	3.905542	0.0000984	0.1755670	0.5299053
LotArea	0.2701473	0.0169594	15.929064	0.0000000	0.2368790	0.3034155
MSZoningRH	0.2411929	0.1089905	2.212972	0.0270585	0.0273928	0.4549931

It appears that the zoning variable has the strongest effect on price, as does the size of the lot. The RM Level of the zoning variable means "Residential Medium Density." Bearing in mind that we did a log transform to the sale price (which makes this a log-linear model), we can estimate that a property in this zoning area increases the median sale price by a multiplicative factor of $e^{.458} = 1.5809$. Similarly, zoning classification FV (floating village residential) indicates an $e^{.354} = 1.4248$ multiplier to the median sale price. Our most influential continuous variable, the area of the lot the proper is built on, gives a $e^{.27} = 1$ multiplier to the median sale price for each unit (acre) increase.



selection	inCriteria	stopCriteria	chooseCriteria
stepwise	aic	aic	cv
stepwise	sbc	sbc	cv
stepwise	cp	cp	cv
stepwise	cv	cv	cv

Figure 1: Figure 1. Assumption Validation Regression

Predictive Models

Introduction and Type of Selection

The goal of this section is to make as performant a model as possible. We are not trying to be interpretable or parsimonious, we are primarily optimizing on Kaggle score or average squared error for test data sets (and ideally also for training data sets).

The process of of model optimization begins with an early analysis and review of basic regression assumptions, followed by comparing individual model performances for various fit statistics: adjusted R^2, Akaike Information Criterion (AIC), Bayesian Information Criterion (SBC), internal and external average squared error (ASE), and internal cross validation partial residual sum of squares (cv press). The final step is to optimize models by including new features and interactions.

Predictive models analyzed here are limited to one of four types of penalty-based regression estimation: stepwise selection (penalty-free least-squares estimation), modified forward selection via least angle regression selection (lar/lars), least absolute shrinkage and selection operator regression (lasso), and elastic net regression. More information can be found on these regression types here.

SAS and R will both be used throughout this process. The latter is used to clean and merge training and test data sets which are exported into SAS for regression estimation. As mentioned above in the Exploratory Analysis section, The R script used for this is `cleaner_script.R`.

Early Analysis and Assumption Review

Because of the high number of possible explanatory variables, an initial regression estimation phase is performed in order to ascertain which regression selections tend to minimize key statistics, which will reveal a tentative model to analyze for the purpose of evaluating assumptions.

The following 4 models are compared to identify most common predictors to use for assumption review. We will use the SPLIT option in SAS to allow for each factor level to enter or exit the model independently.

Since the second model, based on selecting according to lowest BIC, has the fewest predictors, we will proceed with that and check residuals for normality and constant variance. These predictors and diagnostic plots are shown below.

The residuals displayed in the panel below show excellent conformity with the constant variance and normally-distributed residual assumptions. There is some evidence of outlier presence in the studentized residual vs predicted value plot (Row 1, Column 2), which will be addressed in the section below on outlier analysis.

Model Selection: First Analysis

For the first pass model selection phase, the regression permutations in Figure 1 above are expanded to include LARS, LASSO, and Elastic Net. There are two choose options used: internal cv press or BIC. The table below show all of the model selection permutations used in this modeling phase.

model
MSSubClass
MSZoning
MSZoning
LotArea
Neighborhood
Condition1
OverallQual
OverallCond
YearBuilt
YearRemodAdd
Exterior1st
ExterCond
Foundation
BsmtQual
BsmtCond
BsmtExposure
BsmtFinSF1
TotalBsmtSF
Heating
GrLivArea
BsmtFullBath
FullBath
KitchenAbvGr
KitchenQual
Fireplaces
GarageType
GarageCars
GarageQual
WoodDeckSF
ScreenPorchSF
SaleCondition
idxhasB

Figure 2: Figure 2. Predictors used in assumption validation model

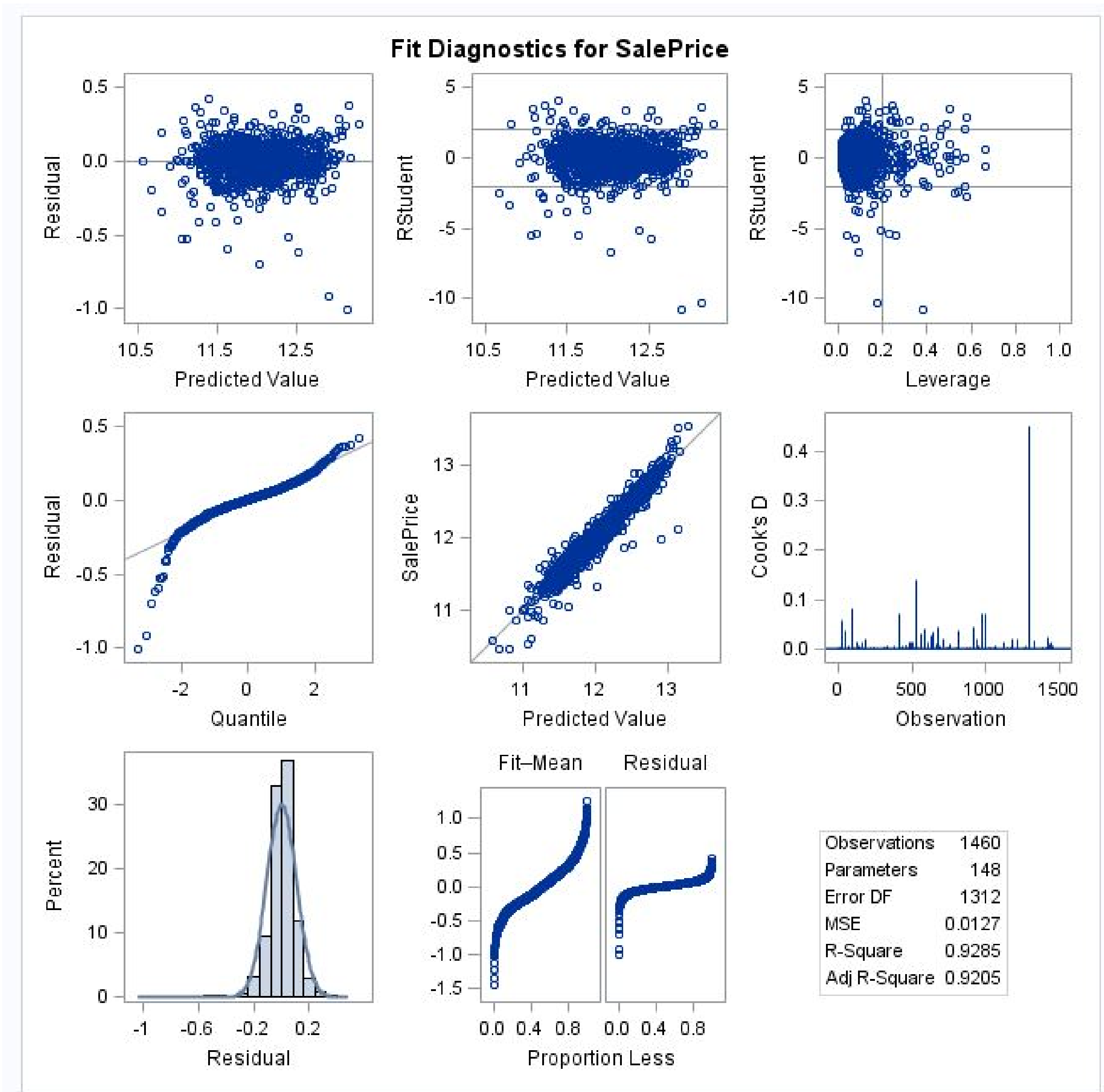


Figure 3: Figure 3. Residual Diagnostic Panel

Model Number	Selection Alg	Criterion to Enter	Criterion to Leave	Min Criterion to Choose
1	elasticnet	aic	aic	cv press
2	elasticnet	cp	cp	cv press
3	elasticnet	cv	cv	cv press
4	elasticnet	sbc	sbc	cv press
5	lar	aic	aic	cv press
6	lar	cp	cp	cv press
7	lar	cv	cv	cv press
8	lar	sbc	sbc	cv press
9	lasso	aic	aic	cv press
10	lasso	cp	cp	cv press
11	lasso	cv	cv	cv press
12	lasso	sbc	sbc	cv press
13	stepwise	aic	aic	cv press
14	stepwise	cp	cp	cv press
15	stepwise	cv	cv	cv press
16	stepwise	sbc	sbc	cv press
17	elasticnet	aic	aic	sbc
18	elasticnet	cp	cp	sbc
19	elasticnet	cv	cv	sbc
20	elasticnet	sbc	sbc	sbc
21	lar	aic	aic	sbc
22	lar	cp	cp	sbc
23	lar	cv	cv	sbc
24	lar	sbc	sbc	sbc
25	lasso	aic	aic	sbc
26	lasso	cp	cp	sbc
27	lasso	cv	cv	sbc
28	lasso	sbc	sbc	sbc
29	stepwise	aic	aic	sbc
30	stepwise	cp	cp	sbc
31	stepwise	cv	cv	sbc
32	stepwise	sbc	sbc	sbc

Figure 4: Figure 4. Model Selection

Model Number	Selection Alg	Criterion to Enter	Criterion to Leave	Min Criterion to Choose	Adj R-Sq
13	stepwise	aic	aic	cv press	0.9492
14	stepwise	cp	cp	cv press	0.9491
15	stepwise	cv	cv	cv press	0.9446
16	stepwise	sbc	sbc	cv press	0.9401
31	stepwise	cv	cv	sbc	0.9383
29	stepwise	aic	aic	sbc	0.9306
30	stepwise	cp	cp	sbc	0.9305
32	stepwise	sbc	sbc	sbc	0.9148
2	elasticnet	cp	cp	cv press	0.9051
3	elasticnet	cv	cv	cv press	0.9027

Model Number	Selection Alg	Criterion to Enter	Criterion to Leave	Min Criterion to Choose	AIC
13	stepwise	aic	aic	cv press	-3298.46028
14	stepwise	cp	cp	cv press	-3298.37451
15	stepwise	cv	cv	cv press	-3221.76738
16	stepwise	sbc	sbc	cv press	-3190.87755
31	stepwise	cv	cv	sbc	-3167.11952
29	stepwise	aic	aic	sbc	-3099.33771
30	stepwise	cp	cp	sbc	-3097.9196
32	stepwise	sbc	sbc	sbc	-2931.31242
2	elasticnet	cp	cp	cv press	-2788.64385
3	elasticnet	cv	cv	cv press	-2770.52618

Model Number	Selection Alg	Criterion to Enter	Criterion to Leave	Min Criterion to Choose	SBC
16	stepwise	sbc	sbc	cv press	-3854.79118
31	stepwise	cv	cv	sbc	-3840.62008
14	stepwise	cp	cp	cv press	-3765.75603
13	stepwise	aic	aic	cv press	-3756.25486
29	stepwise	aic	aic	sbc	-3724.5393
30	stepwise	cp	cp	sbc	-3723.12119
15	stepwise	cv	cv	cv press	-3679.56196
32	stepwise	sbc	sbc	sbc	-3647.90672
19	elasticnet	cv	cv	sbc	-3518.93554
23	lar	cv	cv	sbc	-3518.8005

Figure 5: Figure 5a. Top Adj. Rsq. / AIC / BIC Models

The tables below showcase the top 10 models with respect to each of the six target statistics. Note that 40% of the training data was held out as test data.

Surprisingly, the basic stepwise algorithm that is often heralded as being inferior to penalty-based regression selection methods performed the best overall on all accounts except for the Test ASE. A figure containing all of the predictors for the above 32 model fits is not included because this is considered exploratory. However, the SAS code provided in the appendix can be used to recreate these exact fits using SEED=12345 option.

Because the Test ASE is an important statistic to optimize, we will take an "average predictor" approach in which the frequency that each predictor is selected contributes to its score, with the top scoring predictors across all models being used as the set of predictors to include in the regression model. Furthermore, we will not partition the data set for this part to ensure that we using as much data as possible to establish a set of initial predictors.

Top predictors are defined as having a score of 4 or more. The score is the average number of times the predictor was selected. The levels for any categorical predictor that meets this criteria will be examined to identify potential groupings of levels that offer additional degrees of freedom. This also helps reduce the possibility that our factor estimates represent training data noise and not a real difference in the predicted sale price.

The unique list of factors below contains every predictor selected in at least one of the 32 regression fits. Frequency A represents the number of times each factor was selected when the `CHOOSE=CV` option was used, and Frequency B represents the number of times each factor was selected when the `CHOOSE=SBC` option was

Model Number	Selection Alg	Criterion to Enter	Criterion to Leave	Min Criterion to Choose	ASE (Train)
13	stepwise	aic	aic	cv press	0.00742
14	stepwise	cp	cp	cv press	0.00745
15	stepwise	cv	cv	cv press	0.00808
16	stepwise	sbc	sbc	cv press	0.00921
31	stepwise	cv	cv	sbc	0.0095
29	stepwise	aic	aic	sbc	0.01057
30	stepwise	cp	cp	sbc	0.01059
32	stepwise	sbc	sbc	sbc	0.01327
2	elasticnet	cp	cp	cv press	0.01472
3	elasticnet	cv	cv	cv press	0.0152

Model Number	Selection Alg	Criterion to Enter	Criterion to Leave	Min Criterion to Choose	ASE (Test)
32	stepwise	sbc	sbc	sbc	0.01487
18	elasticnet	cp	cp	sbc	0.01509
22	lar	cp	cp	sbc	0.0216
26	lasso	cp	cp	sbc	0.0216
17	elasticnet	aic	aic	sbc	0.0226
20	elasticnet	sbc	sbc	sbc	0.0226
21	lar	aic	aic	sbc	0.0226
24	lar	sbc	sbc	sbc	0.0226
25	lasso	aic	aic	sbc	0.0226
28	lasso	sbc	sbc	sbc	0.0226

Model Number	Selection Alg	Criterion to Enter	Criterion to Leave	Min Criterion to Choose	CV PRESS
15	stepwise	cv	cv	cv press	8.85661
14	stepwise	cp	cp	cv press	9.33218
13	stepwise	aic	aic	cv press	9.34275
16	stepwise	sbc	sbc	cv press	9.76983
3	elasticnet	cv	cv	cv press	11.87947
7	lar	cv	cv	cv press	11.87947
11	lasso	cv	cv	cv press	11.87947
2	elasticnet	cp	cp	cv press	11.88508
1	elasticnet	aic	aic	cv press	12.25036
5	lar	aic	aic	cv press	13.07142

Figure 6: Figure 5b. Top ASE and Cv Press Models

used. This is Model 1.

Predictor	Frequency A	Frequency B	Average
Intercept	16	16	16
GarageCars	16	16	16
GrLivArea	16	16	16
OverallQual	16	16	16
YearBuilt	16	16	16
YearRemodAdd	16	16	16
Fireplaces_0	13	14	13.5
BsmtFinSF1*idxhasFB1	13	13	13
LotArea	13	13	13
MSZoning_RM	13	13	13
FirstFlrSF	12	12	12
CentralAir_N	10	10	10
GarageArea	9	10	9.5
BsmtExposure_Gd	9	9	9
BsmtFullBath_0	9	9	9
BsmtQual_Ex	9	9	9
Condition1_Norm	9	9	9
HeatingQC_Ex	9	9	9
KitchenQual_Ex	9	9	9
MSZoning_C	9	9	9
Neighborhood_Crawfor	9	9	9
Neighborhood_Edwards	9	9	9
OverallCond_3	9	9	9
SaleCondition_Abnorml	9	9	9
TotalBsmtSF*idxhasB	9	9	9
WoodDeckSF*idxhasWD	9	9	9
KitchenAbvGr_2	8	8	8
OverallCond_4	8	8	8
Fireplaces_2	6	7	6.5
Foundation_PConc	6	6	6
FullBath_3	6	6	6
Neighborhood_NoRidge	6	6	6
Neighborhood_NridgHt	6	6	6
ScreenPorch*idxhasSP	6	6	6
Neighborhood_StoneBr	5	6	5.5
BsmtFullBath_1	5	5	5
ExterQual_TA	5	5	5
KitchenQual_TA	5	5	5
Exterior1st_BrkFace	4	5	4.5
BsmtCond_Po	4	4	4
GarageType_2Types	4	4	4
MSSubClass_160	4	4	4
Neighborhood_Somerst	4	4	4
OverallCond_5	4	4	4
OverallCond_6	4	4	4
OverallCond_9	4	4	4
Neighborhood_BrkSide	3	4	3.5
ExterCond_Po	3	4	3.5
GarageQual_Ex	3	4	3.5
GarageCond_Fa	3	4	3.5
MSSubClass_30	2	5	3.5
Foundation_Stone	3	3	3
Heating_Grav	3	3	3
Fireplaces_3	3	3	3
idxhasB	3	3	3
HalfBath_1	1	5	3
ExterCond_Fa	2	3	2.5
KitchenAbvGr_1	2	2	2
LotConfig_CulDSac	0	4	2
BsmtExposure_No	0	4	2
FullBath_1	0	4	2
BsmtFinType1_GLQ	1	2	1.5
GarageFinish_Unf	1	2	1.5
TotRmsAbvGrd	0	3	1.5
MSSubClass_50	0	3	1.5
MSSubClass_60	0	3	1.5
Condition1_RRAe	0	3	1.5
Exterior2nd_VinylSd	0	3	1.5
MasVnrType_BrkCmn	0	3	1.5

Predictor	Frequency A	Frequency B	Average
Heating_GasW	0	3	1.5
Electrical_SBrkr	0	3	1.5
EnclosedPor*idxhasEP	0	3	1.5
idxhasWD	0	3	1.5
BedroomAbvGr_4	0	2	1
OpenPorchSF*idxhasOP	0	2	1
SaleCondition_Partial	0	2	1
Neighborhood_Mitchel	0	2	1
Exterior1st_MetalSd	0	2	1
CentralAir_Y	0	2	1
BedroomAbvGr_3	0	2	1
GarageQual_Gd	0	2	1
SaleCondition_Family	0	2	1
BsmtQual_None	1	0	0.5
ExterQual_Gd	0	1	0.5
BsmtQual_TA	0	1	0.5
HalfBath_0	0	1	0.5
FireplaceQu_Gd	0	1	0.5
GarageType_Attchd	0	1	0.5
GarageFinish_Fin	0	1	0.5
GarageCond_TA	0	1	0.5
PavedDrive_N	0	1	0.5
PavedDrive_Y	0	1	0.5
idxhasFB1	0	1	0.5
Fireplaces_1	0	1	0.5
BsmtFinType1_None	0	1	0.5
LotShape_Reg	0	1	0.5
LotConfig_FR2	0	1	0.5
LotConfig_FR3	0	1	0.5
Neighborhood_Blueste	0	1	0.5
Neighborhood_BrDale	0	1	0.5
Neighborhood_ClearCr	0	1	0.5
Neighborhood_CollgCr	0	1	0.5
Neighborhood_Gilbert	0	1	0.5
Neighborhood_Meadow'	0	1	0.5
Neighborhood_SawyerW	0	1	0.5
Neighborhood_Timber	0	1	0.5
Neighborhood_Veenker	0	1	0.5
Condition1_RRNn	0	1	0.5
BldgType_Twnhs	0	1	0.5
OverallCond_1	0	1	0.5
RoofStyle_Mansard	0	1	0.5
Exterior2nd_AsphShn	0	1	0.5
Exterior2nd_MetalSd	0	1	0.5
MasVnrArea*idxhasMV	0	1	0.5
BsmtCond_TA	0	1	0.5
BsmtExposure_None	0	1	0.5
BsmtFinType1_LwQ	0	1	0.5
BsmtFinType2_ALQ	0	1	0.5
BsmtFinType2_GLQ	0	1	0.5
BsmtFullBath_3	0	1	0.5
BedroomAbvGr_6	0	1	0.5
GarageQual_Fa	0	1	0.5
MoSold_5	0	1	0.5
MoSold_10	0	1	0.5
YrSold_2006	0	1	0.5
SaleCondition_AdjLand	0	1	0.5
idxhasLF	0	1	0.5

In addition, manual factor level grouping was performed by analyzing group patterns and judgmentally selecting similar groups or combining groups without sufficient information to reliably estimate an average for that group. These manual groupings along with the same predictors in Model 1 form Model 2. Finally, Model 3 expands on Model 2 by including additional variables which improved the overall fit.

The table below shows the test ASE and Kaggle scores for Models 1-3.

Outlier Analysis

Model Selection: Second Analysis and Kaggle

Additional Improvements

Conclusion

In the battle of the property value tropes, it turns out that the old adage of "location, location, location" is right afterall. We compared models that focused on location, curb appeal, and the interior construction of a property, and found that on all measures, the location centric model was most predictive. This is useful for the parties in a real estate transaction because they know that playing up the location of a property can lead to a higher sale price. On the flipside, if you are looking for a home, it is good to know that you can likely get a good deal on a otherwise very nice property if you are williing to live outside of the premier neighborhoods.

Appendix: Code for all analyses

Appendix A: interp_models.R

The below commented code provides the steps taken in order to create, analyze, and select our models focused on interpretability.

```
# Load libraries
# If you don't have one, you will have to run install.packages('library')
library(dplyr)
library(purrr)
library(broom)

train <- read.csv("data/train1_clean.csv")

# Select variables related to the location of a property
location <- train %>%
  select(MSZoning,
         LotFrontage,
         LotArea,
         Neighborhood,
         Condition1,
         SaleCondition,
         SalePrice)
# Select variables related to the external appearance of a property
outside <- train %>%
  select(LotConfig,
```

```
         BldgType,
         HouseStyle,
         RoofStyle,
         Exterior1st,
         Exterior2nd,
         MasVnrType,
         MasVnrArea,
         ExterQual,
         ExterCond,
         PavedDrive,
         SaleCondition,
         SalePrice)

# Select variables related to the internal
inside <- train %>%
  select(Foundation,
         BsmtFinType1,
         BsmtFinType2,
         Heating,
         HeatingQC,
         CentralAir,
         Electrical,
         Fireplaces,
         SaleCondition,
         SalePrice)

# Combine our 3 datasets into a list for easy functional mapping
model_dfs <- list(location, outside, inside)

# Creat a helper function that will generate a model formula
# and run a standard OLS regression of SalePrice against all
# variables for each of our 3 datasets
model_fit <- function(x) {
  model_formula <- formula("SalePrice ~ .")
  lm(model_formula, data = x)
}

# Map our model fitting function above to all 3 datasets
models <- map(model_dfs, model_fit) %>%
  set_names(c("location", "outside", "inside"))

# Add NAmes as a level because it shows up in the test
# data and not the training data
models$location$xlevels$Neighborhood <-
  c(models$location$xlevels$Neighborhood, "NAmes")

# Map the broom::tidy function across our models to get parameter
# estimates in a tidy data frame
model_params <- map(models, tidy)

# Map the broom::glance function across our models to
# get model diagnostics in a tidy data frame
model_diags <- map(models, glance)
```

```
# After looking at the below, we can see that location seems to
# be the best model, will continue analysis with that model
bind_cols(
  data.frame(ModelName = c("location", "inside", "outside")),
  bind_rows(model_diags)
)
# Give selected model its own variable for easy reference
loc_lm <- models$location

# Check Assumptions
# Constant variance looks good, one suspicious point but not too bad
#plot(loc_lm$residuals)

# Symmetric mostly normal distribution, assumption ok
#hist(loc_lm$residuals)

# Read in the test data and get rid of the SalePrice column
# which we added to facilitate prediction in SAS
test <- read.csv("data/test_clean.csv")
test_x <- test[, -which(names(test) == "SalePrice")]

# Map the predict function aross our 3 models to generate
# predictions for submitting to kaggle
preds <- list()
preds <- map(models, predict, newdata = test_x)
```

Appendix B: cleaner_funs.R

The below code provides the functions we used for cleaning up the data set.

```
#Libraries

#Functions
factor.Adjust = function(data,adj,narep=FALSE) {
  for (i in seq(nrow(adj))) {
    x = as.character(unlist(unname(adj[i,])))
    feature = x[1]
    replace = x[2]
    with = x[3]

    if (!narep) {

      if (is.factor(data[,feature])) {
        feature.NewLevels = gsub(replace,with,levels(data[,feature]),fixed=TRUE)
        levels(data[,feature]) = feature.NewLevels
      } else
        data[data[,feature]==as.numeric(replace),feature] = as.numeric(with)
    } else {
      my.Class = class(data[,feature])
      data[is.na(data[,feature]),feature] = with
      class(data[,feature]) = my.Class

    }
  }
```

```
  return(data)
}

indicator.Add = function(data,indices) {
  for (i in seq(length(indices))) {
    new.Col = names(indices)[i]
    data[,new.Col] = 0
    data[indices[[i]],new.Col] = 1
  }
  return(data)
}

impute.bySampling = function(data) {
  data.Complete = data[!is.na(data)]
  na.Indices = which(is.na(data))

  sample.Size = length(na.Indices)
  data[na.Indices] = sample(data.Complete,size=sample.Size,replace=TRUE)
  return(data)
}
```

Appendix C: cleaner_script.R

The below code provides the script that calls the above cleaner functions, and does various other data cleaning.

```
# Import data

train <- read.csv('data/train1.csv')
test <- read.csv('data/test.csv')

#Drop unwanted columns
train_drop = c('LandContour','Utilities','LandSlope','Condition2',
                 'SaleType','Street','RoofMatl','X3SsnPorch', 'Functio.1l')

test_drop = c('LandContour','Utilities','LandSlope','Condition2',
                 'SaleType','Street','RoofMatl','X3SsnPorch','PoolQC', 'Fence',
                 'MiscFeature', 'Alley', 'Functional')

train <- train[, !(names(train) %in% train_drop)]
test  <- test[, !(names(test) %in% test_drop)]

# Fix column names
names(train)[names(train)=='X1stFlrSF'] = 'FirstFlrSF'
names(train)[names(train)=='X2ndFlrSF'] = 'SecFlrSF'
names(train)[names(train)=='Kitche.1bvGr'] = 'KitchenAbvGr'
names(train)[names(train)=='Functio.1l'] = 'Functional'
names(train)[names(train)=='ScreenPorch'] = 'ScreenPorchSF'

names(test)[names(test)=='X1stFlrSF'] = 'FirstFlrSF'
names(test)[names(test)=='X2ndFlrSF'] = 'SecFlrSF'
names(test)[names(test)=='Functio.1l'] = 'Functional'
names(test)[names(test)=='ScreenPorch'] = 'ScreenPorchSF'
```

```
# Create None factor level for FireplaceQu to represent the NA values
levels(test$FireplaceQu) = c(levels(test$FireplaceQu),'None')

#Identify source of missing data and reassign values
#The custom function factor.adjust will handle both categorical and
# continuous variables

level.Fix =  rbind.data.frame(
  c('MSZoning','C (all)','C'),
  c('LotFrontage','-1','0'),
  c('Neighborhood','-1mes','Other'),
  c('MasVnrType','-1','None'),
  c('MasVnrArea','-1','0'),
  c('BsmtQual','-1','None'),
  c('BsmtCond','-1','None'),
  c('BsmtExposure','-1','None'),
  c('BsmtFinType1','-1','None'),
  c('BsmtFinType2','-1','None'),
  c('Electrical','-1','SBrkr'),
  c('FireplaceQu','-1','None'),
  c('GarageType','-1','None'),
  c('GarageYrBlt','-1','0'),
  c('GarageFinish','-1','None'),
  c('GarageQual','-1','None'),
  c('GarageCond','-1','None')
)

level.Fix2 =  rbind.data.frame(
  c('MSZoning','C (all)','C'),
  c('Neighborhood','-1mes','Other')
)

#Assign missing values to existing factor levels

level.Fix3 = rbind.data.frame(
  c('MSZoning','NA','RL'),
  c('Exterior1st','NA','VinylSd'),
  c('Exterior2nd','NA','VinylSd'),
  c('MasVnrType','NA','None'),
  c('MasVnrArea','NA','0'),
  c('BsmtQual','NA','TA'),
  c('BsmtCond','NA','TA'),
  c('BsmtExposure','NA','No'),
  c('BsmtFinSF1','NA','0'),
  c('BsmtFinSF2','NA','0'),
  c('BsmtFinType2','NA','Unf'),
  c('BsmtUnfSF','NA','0'),
  c('TotalBsmtSF','NA','0'),
  c('BsmtHalfBath','NA','0'),
  c('KitchenQual','NA','TA'),
  c('FireplaceQu','NA','None'),
  c('GarageCars','NA','2'),
  c('GarageQual','NA','TA'),
  c('GarageCond','NA','TA')
```

```
)

names(level.Fix) = c('Feature','Replace','With')

train = factor.Adjust(data=train,adj=level.Fix)
test = factor.Adjust(data=test,adj=level.Fix2)
test = factor.Adjust(data=test,adj=level.Fix3,narep=TRUE)



# Create a predicted SalePrice for the train. Use this line if exporting to
# SAS
test$SalePrice = -1

# Indicator variables which are required for correct parameterization of
# continuous variables which have no value for certain homes
# e.g. 81 homes have no garage, so we use hasGarage x GarageYrBlt
# instead of dropping GarageYrBlt because of the homes with value 0

new.Indicators.Indices =
  list(
    'idxhasG' = which(train$GarageYrBlt != 0),
    'idxhasMV' = which(train$MasVnrArea != 0),
    'idxhasFB1' = which(train$BsmtFinSF1 != 0),
    'idxhasFB2' = which(train$BsmtFinSF2 != 0),
    'idxhasB' = which(train$TotalBsmtSF != 0),
    'idxhasSF' = which(train$SecFlrSF != 0),
    'idxhasPool' = which(train$PoolArea != 0),
    'idxhasLF' = which(train$LotFrontage != 0),
    'idxhasLQF' = which(train$LowQualFinSF != 0),
    'idxhasWD' = which(train$WoodDeckSF != 0),
    'idxhasOP' = which(train$OpenPorchSF != 0),
    'idxhasEP' = which(train$EnclosedPorch != 0),
    'idxhasSP' = which(train$ScreenPorchSF != 0),
    'idxhasMV' = which(train$MiscVal != 0)
  )

new.Indicators.Indices2 =
  list(
    'idxhasG' = which(test$GarageYrBlt != 0),
    'idxhasMV' = which(test$MasVnrArea != 0),
    'idxhasFB1' = which(test$BsmtFinSF1 != 0),
    'idxhasFB2' = which(test$BsmtFinSF2 != 0),
    'idxhasB' = which(test$TotalBsmtSF != 0),
    'idxhasSF' = which(test$SecFlrSF != 0),
    'idxhasPool' = which(test$PoolArea != 0),
    'idxhasLF' = which(test$LotFrontage != 0),
    'idxhasLQF' = which(test$LowQualFinSF != 0),
    'idxhasWD' = which(test$WoodDeckSF != 0),
    'idxhasOP' = which(test$OpenPorchSF != 0),
    'idxhasEP' = which(test$EnclosedPorch != 0),
    'idxhasSP' = which(test$ScreenPorchSF != 0),
    'idxhasMV' = which(test$MiscVal != 0)
  )
```

```
train = indicator.Add(data=train,indices = new.Indicators.Indices)
test = indicator.Add(data=test,indices = new.Indicators.Indices2)


#Imputing by sampling
test$LotFrontage = impute.bySampling(data=test$LotFrontage)
test$BsmtFinType1 = impute.bySampling(data=test$BsmtFinType1)
test$BsmtFullBath = impute.bySampling(data=test$BsmtFullBath)
test$GarageType = impute.bySampling(data=test$GarageType)
test$GarageYrBlt = impute.bySampling(data=test$GarageYrBlt)
test$GarageFinish = impute.bySampling(data=test$GarageFinish)
test$GarageArea = impute.bySampling(data=test$GarageArea)

# Variable Transformations

log.Transform = c('LotFrontage', 'LotArea', 'MasVnrArea', 'BsmtFinSF1',
                  'TotalBsmtSF', 'FirstFlrSF', 'SecFlrSF', 'GrLivArea',
                  'WoodDeckSF', 'OpenPorchSF', 'MiscVal', 'SalePrice')

log.Transform2 = c('LotFrontage', 'LotArea', 'MasVnrArea', 'BsmtFinSF1',
                  'TotalBsmtSF', 'FirstFlrSF', 'SecFlrSF', 'GrLivArea',
                  'WoodDeckSF', 'OpenPorchSF', 'MiscVal')

# Log transform skewed variables
train[,log.Transform] = log(train[,log.Transform] + .001)
test[,log.Transform2] = log(test[,log.Transform2] + .001)


# Merge training and test files, including '.' for empty salePrices for SAS to predict
merge = rbind.data.frame(train,test)

write.csv(train,'data/train1_clean.csv',row.names=FALSE)
write.csv(test,'data/test_clean.csv',row.names=FALSE)
write.csv(merge,'data/merge_clean.csv',row.names=FALSE)


# Below code can be uncommented and run if desired. It generates a PDF with
# histograms and scatter/box plots (against sale price) for each variable in the data set.

# pdf('plots.pdf')
# feature.Levels = lapply(X=train,table)
# for (i in seq(feature.Levels)) {
# f.num = i
# barplot(feature.Levels[[f.num]],main=names(feature.Levels[f.num]))
# plot(x=train[,names(feature.Levels[f.num])],y=log(train$SalePrice),
#      xlab=names(feature.Levels[f.num]))
# }
# dev.off()
# fit = lm(log(SalePrice) ~ train[,names(feature.Levels[f.num])],data=train)
#
```

Appendix C: sas_modeling.txt

The below code provides the SAS code used for generating and evaluating our SAS regressions used for finding the most predictive model.