BA305

PREDICTING HOUSE PRICES IN AMES, IOWA

TEAM 20: VICTOR FLORIANO, MAHIMA MASETTY, ANERI PATEL, JORDAN TEMAN

Table of Contents

Project and Data Description

Building a Housing Sales Price Prediction Model for Houses in Ames, Iowa

According to Zillow, the nationwide median error rate for Zestimate for on-market homes is 1.9%, while the Zestimate for off-market homes has a median error rate of 6.9%. Taking inspiration from this Zestimate model, we wanted to build a prediction model that takes in input variables describing different features of a house and outputs a price for the house. Additionally, with the market for homes constantly changing, which directly affects pricing, we thought a price prediction model for houses is all the more useful. Our goal is that someone who uses our model can hopefully not only tell the predictive price of the house they're looking at, but also be informed of the features that impact that price.

When looking for datasets to build our model off of, we found a dataset on Kaggle that contained extensive information on house features and prices for houses in Ames, Iowa. While this dataset limits the location of where our model could be used, it is a good starting point to build our model from.

Describing Our Data

The dataset we found on Kaggle about houses sold in Ames, Iowa has 81 variables and 1460 instances and contains data from 1950 to 2010. The variables and their descriptions can be seen in *Appendix 1:Description of Variables in the Original Dataset.* After one-hot-encoding our variables, we ended up having 247 predictors with 37 of them being numerical and 210 of them being categorical. It was important for us to minimize the data to account for the curse of dimensionality of data. As a result we spent a lot of time reducing our data. After preprocessing we had 115 predictors, 27 being numerical and 88 being categorical. To further reduce the data, we also performed forward selection and after the top features were selected, we reduced our dataset to contain only 23 predictors, 11 being numerical and 12 being categorical.

Data Pre-Processing

Dropping Predictors To Reduce Data Dimension

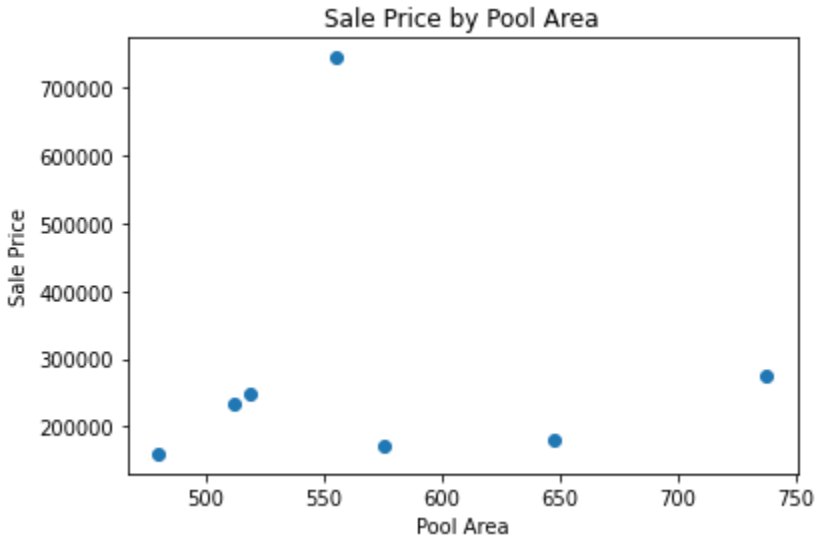
Since we were dealing with a dataset containing a large number of predictors with a lot of them being categorical predictors which are hard to work with, we had to spend a lot of time cleaning our data before we could run any models.

We started our pre-processing work with dropping predictors and we did this in the following two ways:

1. We automatically dropped predictors if more than 20% of their values were NaN.
2. We manually dropped predictors that
    a. Did not intuitively seem to be very relevant to the price of a house. For example: Mason Veneer Area variable which gives us information about the outer coatings of the house.
    b. Did not have enough values to even use as a reference to run imputation models to fill in the missing values. For example: Pool Area has values for only 7 instances as you can see in *Exhibit 1: Relationship Between Pool Area and House Sale Price.*
    c. Explained or captured the same information as some other predictors. For example: No. of Cars in Garage provides information about space of the garage just like Garage Sq. Ft.
    d. Were used in feature engineering to create a new variable that captured more information on our dataset. For example: Years Since Remodel predictor that was created using Year

Remodel Added and Years Sold predictors already present in our dataset.

Exhibit 1: Relationship Between Pool Area and House Sale Price



Creating Predictors to Improve Data Usability

The next step in pre-processing was for us to create some new predictors in place of some of the predictors already present in our dataset to both reduce data and to improve its usability. We saw that there were many predictors that provided rankings in strings and if we were to turn them into a usable form for our models, it would create 5-10 dummy variables for each of the string ranking predictors which is not ideal. To fix this, we turned them into numerical rankings on a scale from either 1-5 or 1-10 based on the predictor's original ranking structure as you can see in *Exhibit 2: Changing String Rankings to Numerical Rankings.*

Exhibit 2: Changing String Rankings to Numerical Rankings.

	BsmtQual	BsmtCond	KitchenQual	HeatingQC	GarageQual	GarageCond
0	Gd	TA	Gd	Ex	TA	TA
1	Gd	TA	TA	Ex	TA	TA
2	Gd	TA	Gd	Ex	TA	TA
3	TA	Gd	Gd	Gd	TA	TA
4	Gd	TA	Gd	Ex	TA	TA

Ranking in String Form

	BsmtQual	BsmtCond	KitchenQual	HeatingQC	GarageQual	GarageCond
0	4	3	4	5	3	3
1	4	3	3	5	3	3
2	4	3	4	5	3	3
3	3	4	4	4	3	3
4	4	3	4	5	3	3

Ranking in Numerical Form

We then noticed that for variables like number of bathrooms and total area of the house, there were pairs of predictors that captured similar information about the basement and above ground level separately. To further reduce our data dimension in this case, we combined these pairs of predictors to capture information about a feature for the entire house in one variable instead of in two variables. For example, we had variables like Basement Full Bath which had information on the number of full bathrooms in the basement, and Above Ground Full Bath which had information on the number of full bathrooms above the basement. We combined these two variables to give us the Total Full Bath variable, which gives us information about the total number

of full bathrooms in the House, as you can see in *Exhibit 3: Combining Information About Features in the Basement and Above Ground Level.*

Exhibit 3: Combining Information About Features in the Basement and Above Ground Level

	BsmtFullBath	FullBath	BsmtHalfBath	HalfBath	TotalBsmtSF	GrLivArea
0	1	2	0	1	856	1710
1	0	2	1	0	1262	1262
2	1	2	0	1	920	1786
3	1	1	0	0	756	1717
4	1	2	0	1	1145	2198

Before Combining the Variables

	Total_Full_Bath	Total_Half_Bath	Total_SF
0	3	1	2566
1	2	1	2524
2	3	1	2706
3	2	0	2473
4	3	1	3343

After Combining the Variables

Additionally, a lot of our predictors captured similar information about different iterations of a feature. For example, if a house has two lots, information about the condition of each of the two lots was captured in two sets of 8 categorical variables. For this reason, we combined them and cut the predictors needed by half as you can see in *Exhibit 4: Combining Variables Based on Features.*

Exhibit 4: Combining Variables Based on Iterations of the Same Features

	Condition1_Feedr	Condition2_Feedr	Condition1_Norm	Condition2_Norm	Condition1_PosA	Condition2_PosA
0	0	0	1	1	0	0
1	1	0	0	1	0	0
2	0	0	1	1	0	0
3	0	0	1	1	0	0
4	0	0	1	1	0	0

Condition of Lot Variables Before Combining

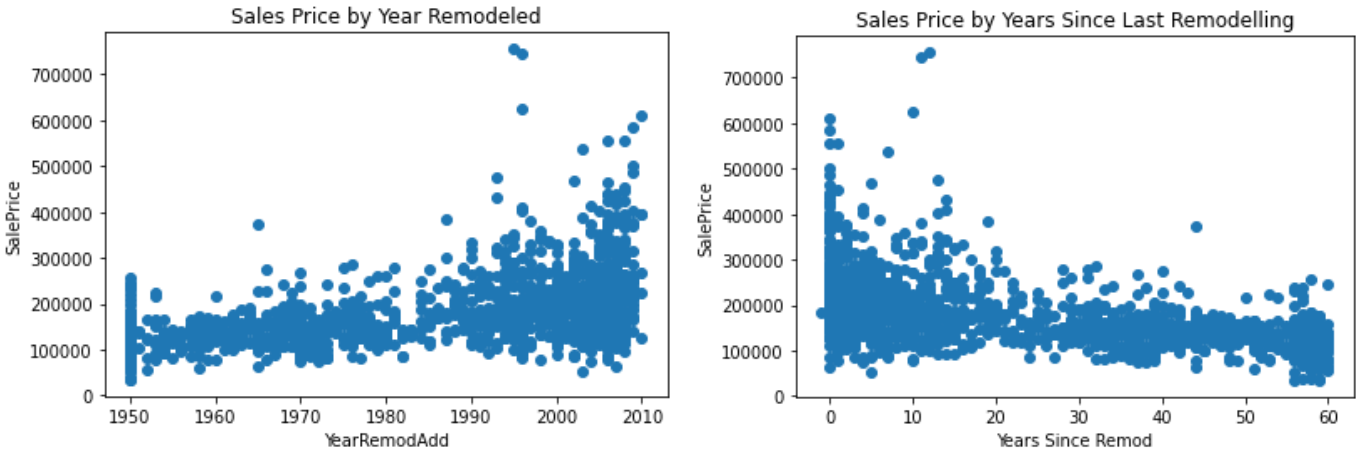
	Condition_Feedr	Condition_Norm	Condition_PosA
0	0	2	0
1	1	1	0
2	0	2	0
3	0	2	0
4	0	2	0

Condition of Lot Variable After Combining

Lastly, we thought that it was important to have a predictor regarding the age of the house to know if it's a fairly modern house or if it's an old house. However, if we used the Year Remodel Added predictor present in our original dataset, we'd end up with 60 categorical variables since the house age ranges from 1950 to 2010. To avoid this, we created a new variable called Years Since Remodel by subtracting an instance's value for Year Remodel Added variable from an instance's value for Year Sold. In *Exhibit 5: Creating Year Since Remodel Variable* you can see that the correlation these variables have to the sale price of a house are opposite in direction. This means that while the relationship between Year Remodel Added and Sale Price is direct, indicating that the later a house was remodeled the high its sale price, the relationship between Years Since Remodel and Sale Price is indirect , indicating the longer it has passed since a remodel the lower

the sale price.

Exhibit 5: Creating Year Since Remodel Variable



Imputing Missing Values

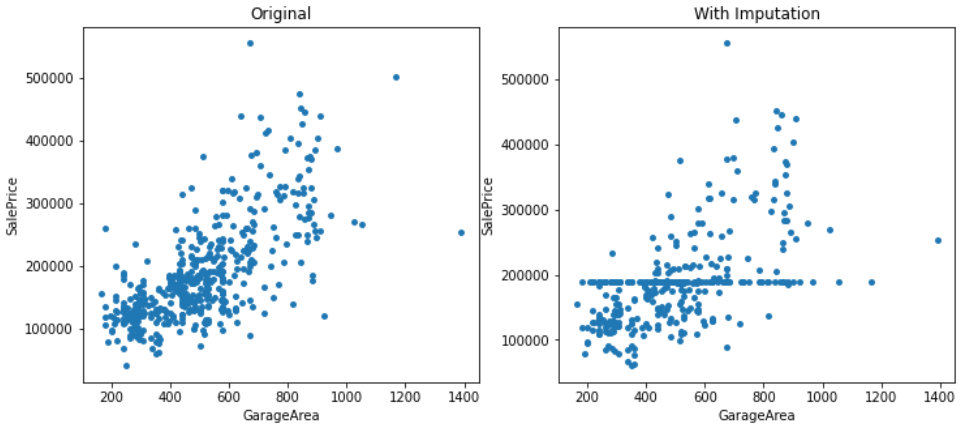
The last step for pre-processing was to impute values for predictors where less than 20% of their values were NaN. Before we imputed values for variables missing observations, we had to test which imputation method is the best. The process we used for comparing the imputation methods is as follows:

1. We took our original dataset and created a new dataset in which we changed some of the values, say 25% of the values, to NaN for a variable that already had values for all the instances like Garage Area.
2. Then we used imputation methods, both univariate imputation using mean and multivariate imputation by chained equations (also known as MICE), to predict the NaN values in this new dataset using the original dataset as a reference.
3. Finally we compared the structure of the imputed dataset to that of the original dataset.

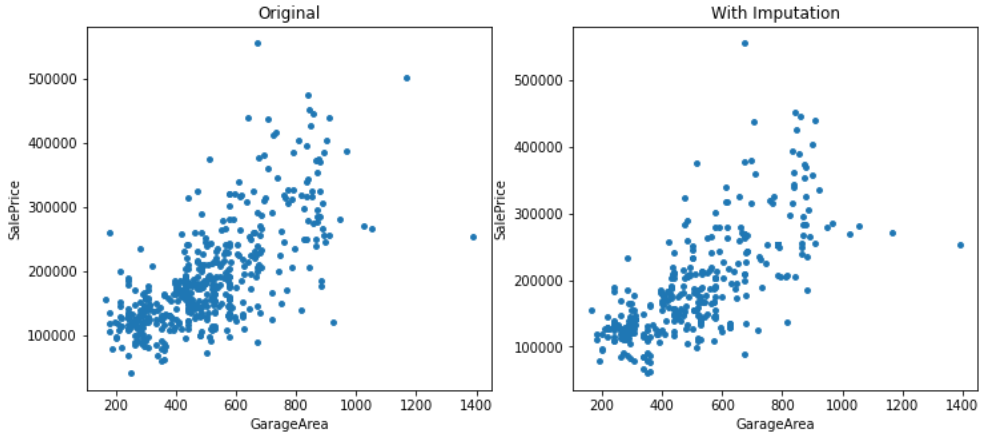
This process can be seen more clearly through the diagram in *Appendix 2: Process for Comparing Imputation Methods.*

From the graphs in *Exhibit 6: Results from Comparing Imputation Methods*, which show the comparison between imputation two methods used for the Garage Area column, we can clearly see that MICE keeps the structure of the original dataset better while univariate imputation using mean fills in the NaN values with a constant mean value which produces a line with instances concentrated around the mean. Therefore, we decided to go with the MICE imputation method to fill in values for variables that were actually missing observations. Going through this entire data cleaning process greatly helped us with data dimension reduction, since we went from 247 predictors to 115 predictors.

Exhibit 6: Results from Comparing Imputation Methods



Using Univariate Imputation Using Mean



Using Multivariate Imputation By Chained Equations

Exploratory Data Visualizations

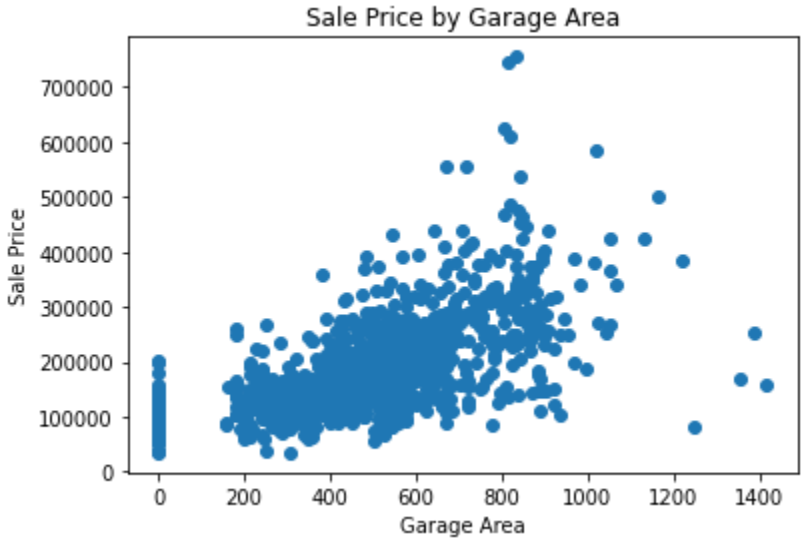
Exploratory Visualizations to Form an Initial Hypothesis

Before we began running our models we decided to make graphs with the variables we thought would most impact Sale Price the most, to see if we wanted to include them in our baseline model. First we made a chart of sale prices by each zone as shown by *Exhibit 7: Sale Price by Zone.* From this, we found that 'Floating Village Residential' and 'Residential Low Density' have the highest priced homes. This correlated with our initial intuition that since the lower density zones are quieter areas with more families and adults, the homes are most likely larger with more land or more expensive in general.

Exhibit 7: Sale Price by Zone

Next we decided to graph Garage Area by Sale Price, because we believed this to be a variable that homeowners would heavily consider when purchasing a home. From the graph in *Exhibit 8: Sale Price by Garage Area,* we gather that the area of the garage generally has a relationship with sales price. As the area increased, and therefore the number of cars a garage increased, the sale price increased. Additionally, the houses with no garage, or instances for which the garage area was 0, were evidently priced lower than those with a garage. This showed us that Garage Area could be impactful in predicting on Sale Price.

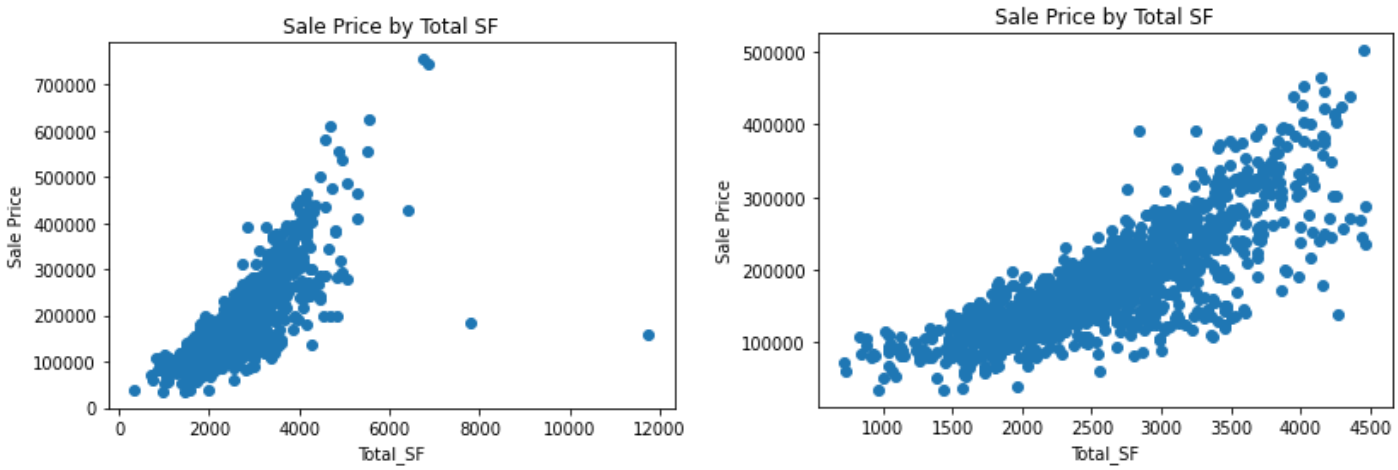
Exhibit 8: Sale Price by Garage Area



We then decided to make a graph examining the Total Square Feet variable, as we believed this would be the most important factor in determining the price of a home. The initial graph was what we expected, a very clear relationship between total square foot and sales price. However, there were a few outliers that made the graph more difficult to view. We removed the outliers by finding the interquartile range, and eliminating above the 75th quartile plus 1.5 times the interquartile range. We did the same process for below the 25th quartile threshold. We then were able to even more clearly see the relationship between the two variables, leaving us to understand

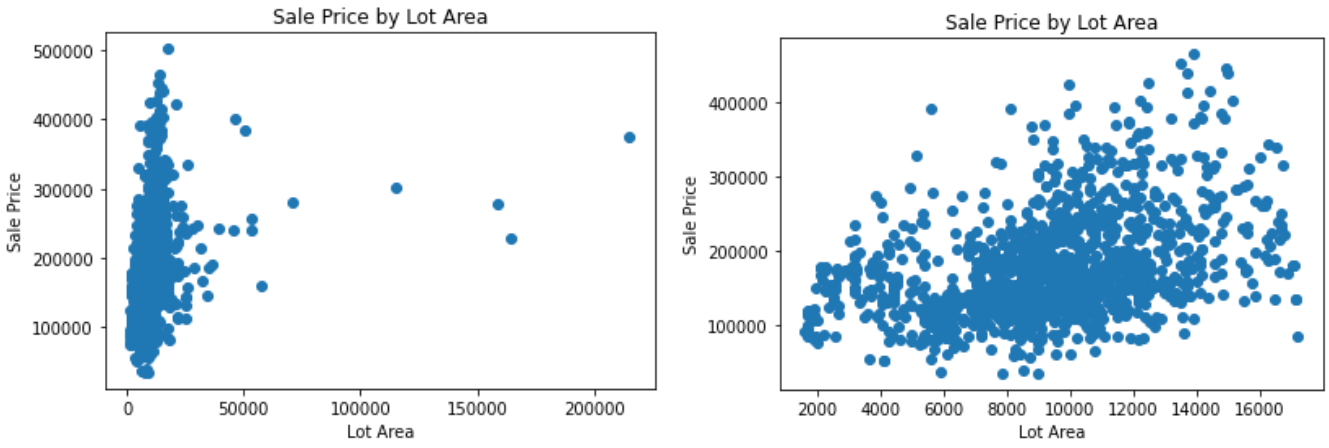
that on average as the total square footage of a house increases, the price of a house also increases.

Exhibit 9: Sale Price by Total Square Foot (Before and After Outlier Removal)



Finally, we decided to investigate the Lot Area because we believed it would behave similarly to Total Square Feet. Initially it was difficult to identify a trend so we followed the same process for removing the outliers as we did for Total Square Feet. We then got our new graph, which did not show as strong of a relationship as we had thought it would. The relationship is much more random than we had hypothesized, showing homes with large lot sizes priced very low. This told us that while lot area may be important, there are other variables that more strongly impact sales price, such as total square footage of a house.

Exhibit 10: *Sale Price by Lot Area (Before and After Outlier Removal)*



Predictive Modeling

Baseline

We then were able to run our baseline model, and use the information we had gathered from our preprocessing and descriptive models. Our baseline model was our human model, where we picked the variables we believed would be most impactful on sales price, and ran a regression on them. We picked 31 variables, 23 of them categorical variables broken into categories, including the variables from the graphs above. We ended up with an RMSE of $30,694, which is not a large error considering the average house price is about $175,009. Overall, we were confident to run our baseline model against other models to see how we could improve.

Exhibit 11: Baseline Model - Selected Predictors

PCA

Once we had our baseline model, we wanted to find a more quantitative method to select our features. Our first attempt to do so was to run a PCA model on all our 115 variables.

Exhibit 12: PCA Results

Component	**% of Variance**	**Cumulative %**
0	0.094	0.094
1	0.037	0.131
2	0.033	0.164
3	0.027	0.191
4	0.023	0.215
5…	0.023	0.237
51	0.008	0.771
52	0.008	0.773
53	0.008	0.787
54	0.008	0.794
55	**0.008**	**0.802**

However, the results were not what we expected. From the image we can see that we would need to include 55 components to reach a cumulative variance of 80%. Our best component (0) also only explained around 9% of the variance. We believe that our PCA did not do well because we had too many categorical predictors in our dataset (88), so we decided to find an alternative

way to shrink our number of features.

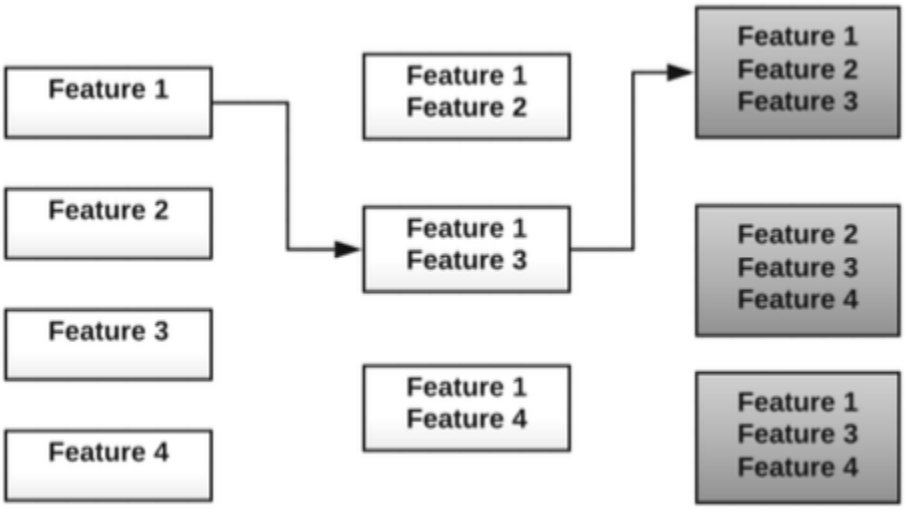
Train/Test Split

The train/test split of our data followed the standard practice, we split our dataset into 80% train and 20% test. We had to do our split before running our feature selection method (Forward Selection), because we wanted to select variables based only on our training data to keep our test data untouched.

Using Forward Selection As Our Feature Selection Method

The method we ended up using for feature selection was the Sequential Feature Selection from Scikit Learn, set to forward selection. The way this method works is: it will first run a linear regression with each feature individually and record their MSE's. Then it will select the feature that led to the lowest MSE and run another set of linear regression with the chosen feature plus an additional feature, one at a time. It will then pick the pair with the lowest MSE and repeat the process until the method selects the number of predictors we specified.

Exhibit 13: Forward Selection Process



In our case, we wanted the Forward Selection Model to keep 20% of our initial features, which turned out to be 23 (11 numerical and 12 categorical). Some of the predictors selected were what we would have guessed and included in our baseline model, such as “Total Sqft”, but there were also variables not included in our base model. The list of all variables chosen by Forward Selection can be found in the *Appendix 3: Variables Chosen From Forward Selection.*

Naive Bayes did not fit our data structure

The first model we tried was a Naive Bayes. However, we soon ran into an issue, the most common Naive Bayes methods are Gaussian NB and Multinomial NB. And while the first works well with numerical variables, the second is used for categorical variables, and neither works with mixed data, which is what we had.

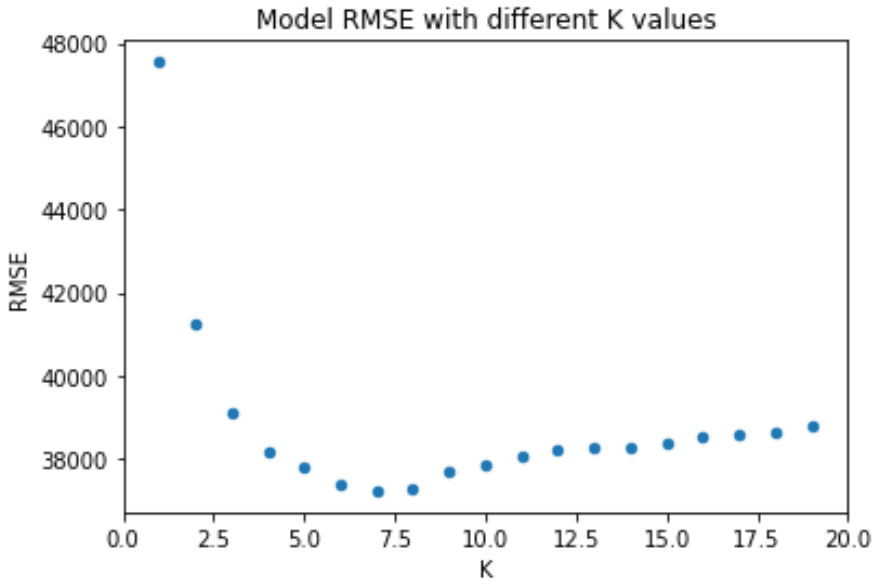
The solution we came up with was to break our continuous/numerical variables into multiple ranges (bins), so that we could treat them as categorical. While this was a valid solution, we decided against it for two reasons. First, we would lose a lot of information from our numerical variables but turning them into ranges. Second, we would end up creating a lot of new predictors, which was something we were trying to avoid. Therefore, we decided not to include Naive Bayes with our other models.

K-Nearest Neighbors And The K Tuning Process

We then decided to test our KNN model. First, we created a pipeline that would first standardize the data, then use it in a KNN model with K=2. We then fit the pipeline to our train data and predicted our yhat_test values. The RMSE of this baseline KNN model was $32,090. From this, we then decided to tune two parameters in our KNN model, which was the number of

neighbors (K), and the weights (uniform or distance). To do so, we used a GridSearch Cross-Validation (cv=5), and that resulted in a best K of 7 and the distance weights metric.

Exhibit 14: KNN Model RMSE by Different K Values

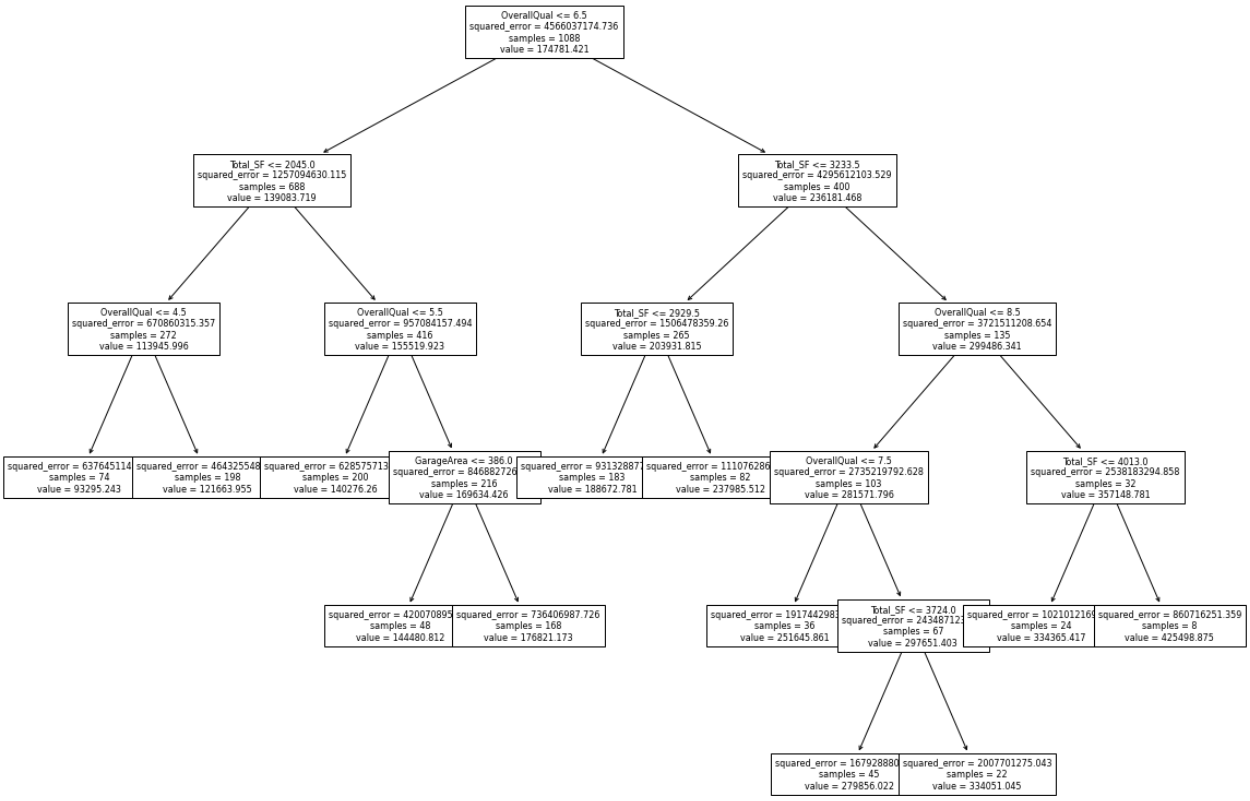


Once we had our tuned parameters, we created a new KNN(Tuned) model, which had a significantly better RMSE of $29,430.

Decision Tree And Pruning

For our Decision Tree Regressor, we decided to tune our maximum leafs parameter using cross-validation (cv=5). After running the model with the best parameter, max_leafs = 12, we had a RMSE score of $30,828. Besides being worse than our KNN model, another issue with our Decision Tree was that DTs have a very high variance, so small changes in our samples would lead to significant changes in our RMSE score.

Exhibit 15: Pruned Decision Tree



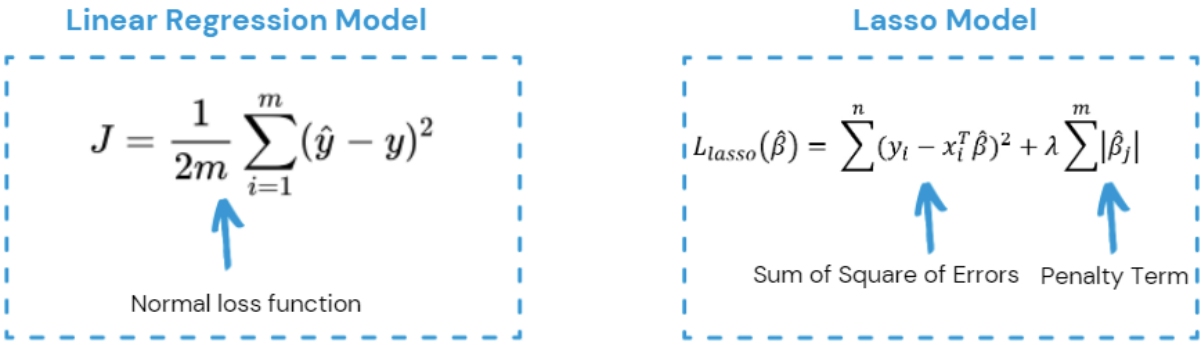
Random Forest Showed An Improvement Over Our Pruned Decision Tree

To try to improve on our DT model, we decided to create a Random Forest model. We were surprised by how well the model performed, it gave us a RMSE of $21,854. Additionally, because the Random Forest gets the average prediction of multiple Decision Trees (with different sampling), we can also expect a lower variance from this model.

Choosing A Regularized Regression - Lasso

Our next model was a Regression. Lasso is a type of linear regression that uses shrinkage in the form of a penalty term. While a normal Linear Regression will choose the coefficients that minimize MSE, Lasso will choose the coefficients that minimize the MSE plus a penalty term, as shown in *Exhibit16*:

Exhibit 16: Different Loss Functions for Linear Regression and Lasso Models



Given that the penalty term in Lasso penalizes large coefficients, we had to standardize our data with a pipeline to avoid having some predictors having their coefficients disproportionally penalized, due to differences in scale. We also tuned the model with cross-validation(cv=5) to find the best value for the “alpha” parameter

The reason why Lasso can sometimes improve on regression models is because by shrinking the coefficients, it lowers the flexibility of the model, increasing its bias but decreasing its variance. When the decrease in variance is more significant than the increase in bias, Lasso will have a better test set performance since test MSE can be put into the form of Variance plus Bias squared:

Exhibit 17: MSE Equation in Terms of Bias and Variance

$$E\left[\left(\hat{\theta}-\theta\right)^2\right] = E\left[\left(\hat{\theta}-E\left[\hat{\theta}\right]\right)^2\right] + \left(E\left[\hat{\theta}\right]-\theta\right)^2$$

$$MSE = Variance + Bias^2$$

Additionally, if shrinkage does not lead to any improvements, when tuning the strength of the regularization (alpha), the model will choose 0, which is essentially the same as running a normal linear regression. In our tuned model, we got our second best RMSE of $23,026.

Key Takeaways

Final Hypothesis and Key Takeaways

Besides our Decision Tree, all of the models showed an improvement over our baseline RMSE ($30,694). Our two best models, as shown in *Exhibit 18: RMSE Values For All Models* were the Random Forest (RMSE $21,854) and the Lasso Regression (RMSE $23,026). Given that our main objective with this project was to be as accurate as possible when predicting house prices and wanted higher interpretability, we selected the Random Forest over Lasso as our final best model.

Exhibit 18: RMSE Values For All Models

Prediction Method/Model Used	**RMSE**
Baseline Model	$30,694
Decision Trees	$30,828
Random Forest	**$21,854**
KNN	$29,430
Lasso Tuned	$23,026

Building this model included a lot of trial and error. Some things that worked for us is that forward selection validated some choices we made for predictors and allowed us to shrink our number of predictors. Additionally, models based on Forward Selection features reached a low RMSE, indicating that the predictors we used had high predictive power. Lastly, tuning our models with the best parameters led to lower RMSEs in every instance.

As for what could be better. It would be better if we had more up to date data. Our data is from 2010, so it is dated and having more up to date data would be beneficial as there might be new features included in the dataset that people in the current time would care about when looking to buy a house. Additionally, we wish we had a bigger dataset, with more instances, so that our model could be more accurate. Many predictors had only a few observations. For example, we thought Pool Area would be an important variable, but it only had a couple observations as discussed previously so we could not conclude anything about its impact on a house's sale price. We also thought the Miscellaneous Feature variable would be helpful because it includes any feature that is not already covered by other variables, but similarly we found there were only a few observations, about 54 out of 1460, and could not conclude any information from it. Furthermore, some of the descriptions of the variables were confusing to understand. It would have been better if the descriptions were clear and detailed, so we would know exactly what it meant. For example, one of the variables was about the different zones and it wasn't clear to us even after researching what a Floating Residential Village is. Finally, if we wanted to predict prices in other regions we would need to get data from other states to scale our model.

Appendix

Appendix 1: Description of Variables in the Original Dataset

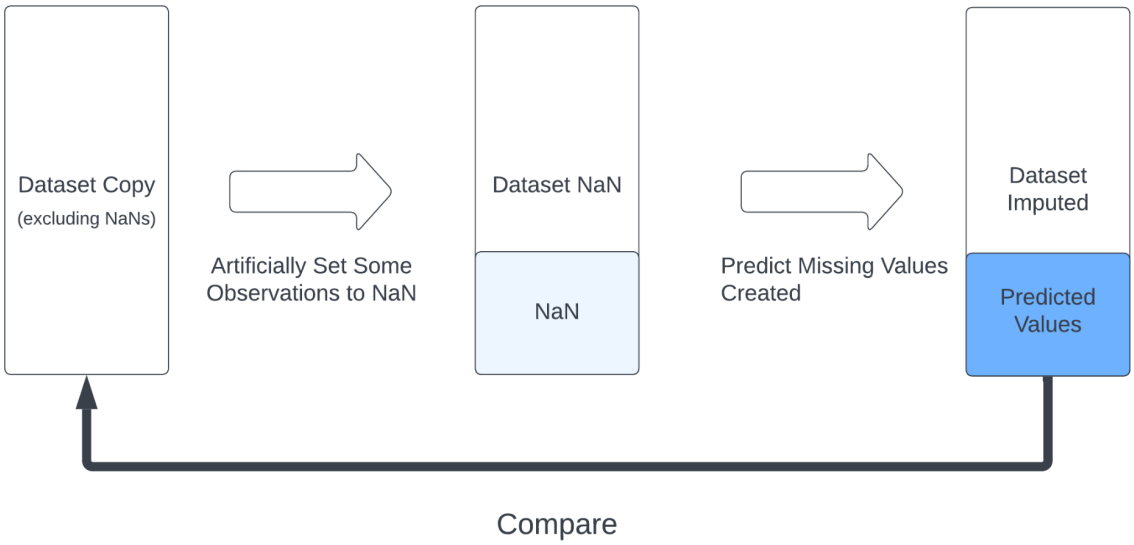
Variable	Description
SalePrice	The property's sale price in dollars
MSSubClass	The building class
MSZoning	The general zoning classification
LotFrontage	Linear feet of street connected to property
LotArea	Lot size in square feet
Street	Type of road access
Alley	Type of alley access
LotShape	General shape of property
LandContour	Flatness of the property
Utilities	Type of utilities available
LotConfig	Lot configuration
LandSlope	Slope of property
Neighborhood	Physical locations within Ames city limits
Condition1	Proximity to main road or railroad
Condition2	Proximity to main road or railroad (if a second is present)

BldgType	Type of dwelling
HouseStyle	Style of dwelling
OverallQual	Overall material and finish quality
OverallCond	Overall condition rating
YearBuilt	Original construction date
YearRemodAdd	Remodel date
RoofStyle	Type of roof
RoofMatl	Roof material
Exterior1st	Exterior covering on house
Exterior2nd	Exterior covering on house (if more than one material)
MasVnrType	Masonry veneer type
MasVnrArea	Masonry veneer area in square feet
ExterQual	Exterior material quality
ExterCond	Present condition of the material on the exterior
Foundation	Type of foundation
BsmtQua	Height of the basement
BsmtCond	General condition of the basement
BsmtExposure	Walkout or garden level basement walls
BsmtFinType1	Quality of basement finished area
BsmtFinSF1	Type 1 finished square feet
BsmtFinType2	Quality of second finished area (if present)
BsmtFinSF2	Type 2 finished square feet
BsmtUnfSF	Unfinished square feet of basement area
TotalBsmtSF	Total square feet of basement area

Heating	Type of heating
HeatingQC	Heating quality and condition
CentralAir	Central air conditioning
Electrical	Electrical system
1stFlrSF	First Floor square feet
2ndFlrSF	Second floor square feet
LowQualFinSF	Low quality finished square feet (all floors)
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	Number of bedrooms above basement level
Kitchen	Number of kitchens
KitchenQual	Kitchen quality
TotRmsAbvGrd	Total rooms above grade (does not include bathrooms)
Functional	Home functionality rating
Fireplaces	Number of fireplaces
FireplaceQu	Fireplace quality
GarageType	Garage location
GarageYrBlt	Year garage was built
GarageFinish	Interior finish of the garage
GarageCars	Size of garage in car capacity
GarageArea	Size of garage in square feet
GarageQual	Garage quality

GarageCond	Garage condition
PavedDrive	Paved driveway
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
PoolQC	Pool quality
Fence	Fence quality
MiscFeature	Miscellaneous feature not covered in other categories
MiscVal	$Value of miscellaneous feature
MoSold	Month Sold
YrSold	Year Sold
SaleType	Type of sale
SaleCondition	Condition of sale

Appendix 2: Process for Comparing Imputation Method



Appendix 3: Variables Chosen From Forward Selection