PROPERTY VALUATION AUTOMATION

Building an accurate and resilient model

MGT 6203 – DATA ANALYTICS IN BUSINESS

FINAL PROJECT REPORT – TEAM 22

FALL 2022

AUTHORS:

Calvin Jouard, Othman Ascehybani, Majdi El Tajoury, Armin Paul Allado, Hatem Alzahrani

Contents

PROJECT BACKGROUND

Buying a home is one of the most expensive and important decisions a person will make in their lifetime. It follows that consumers should therefore have accurate insights into housing pricing. This is what the company Zillow sought to do when they rolled out the Zestimate in 2006, a free, publicly available housing price prediction, powered by a model trained on millions of housing valuations.[1] The Zestimate gave rise to additional business opportunities for Zillow.

In 2018, the company established its iBuyer unit, which buys and flips properties, leveraging their Zestimate algorithm to forecast potential payoff.[1] Growth of this business unit continued, and Q3 2021 saw Zillow purchase twice as many houses as it did the previous quarter.[2] Yet just a month later, Zillow CEO Rich Barton announced they would be shuttering the iBuyer program, because the unpredictability in housing price forecasting "would result in too much earnings and balance sheet volatility."[1]

But the problem touches many more players than iBuyer companies, which exist outside of Zillow. Consumers looking to buy a home benefit from accurate price predictions, like the Zestimate. A prediction model could support the purchasing decision by providing a reliable frame of reference. From a city or government perspective, understanding the real causes of increasing housing prices could be directly used to react, or even prevent, housing prices from getting out of control.

In summary, accurate housing price prediction is a pervasive problem that consumers, governments, banks and companies are still wrestling with. And as a result, consumers don't have the most accurate information possible, and thus are unable to have supreme confidence in house-buying decisions. In leveraging the Ames Housing Dataset, we plan to confront this problem. In addition to achieving accurate price predictions, we also hope to sniff out which of the multitude of variables more so explain variations in housing sale price.

DATA PREPARATION

DATASET

For this project our team chose the ***Ames Housing Dataset*** which can be found on the Kaggle website at this URL → House Prices - Advanced Regression Techniques | Kaggle

The data represents the sales homes sold in the city of Ames, Iowa between 2006 and 2010. It contains 1,460 observations and 79 explanatory variables, comprised of 36 numerical variables and 43 categorical variables.

Each row/observation represents the sale of a house, and each column explains a feature of that house. Some examples of explanatory variables include (but are not limited to):

- Lot Area
- Neighborhood
- Building Type
- Roof Material
- Number of rooms
- Age

A screenshot of the first few rows is below:

Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgType	HouseStyle
1	60	RL	65	8450	Pave	NA	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	2Story
2	20	RL	80	9600	Pave	NA	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Feedr	Norm	1Fam	1Story
3	60	RL	68	11250	Pave	NA	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	2Story
4	70	RL	60	9550	Pave	NA	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	Norm	Norm	1Fam	2Story
5	60	RL	84	14260	Pave	NA	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	Norm	Norm	1Fam	2Story
6	50	RL	85	14115	Pave	NA	IR1	Lvl	AllPub	Inside	Gtl	Mitchel	Norm	Norm	1Fam	1.5Fin

Figure 1 – A snapshot of the dataset used for this project

Additionally, the data set contains a **Sale Price** column, indicating the price at which a given house was sold, which is the response variable in our prediction problem.

EXPLORATORY DATA ANALYSIS

Introduction

Our team conducted an inspection of the dataset in order to determine the type of transformations required before an analysis takes place. We aimed at looking at three things:

- The missing values
- Plotting data fields against the response variable
- Multi-collinearity

Missing Values

For the missing values, our team inspected each column in the dataset to determine the appropriate course of action during the data preparation.

In general, when a column has missing values, our team determined to create an indicator variable to flag rows with missing values. The goal was to detect whether there was a pattern in the missing values.

Columns which had a large percentage of missing values were inspected further by creating boxplots to determine whether the presence or absence of a value had an impact on the response variable.

For columns which had less than 5% of missing values, our team investigated whether an imputation was possible. For several columns, imputation was possible.

Boxplots for Categorical variables

Our team created box plots for each categorical variable (total of 43). To be more specific, a boxplot was created for each category value against the response variable.

By comparing the boxplots of different category values, we could visually verify whether the range of response variables was the same or distinctively different. This analysis was useful to determine whether the categorical variable should be included as a predictor for this problem.

After plotting box plots, we calculated the median range for each category (largest group median minus lowest group median) to get a sense of the most revealing categorical variables when it comes to price. The *OverallQual* variable had the greatest median range of all categorical

variables, with a difference of $382,240 (the difference between the median house in the 10 category, ~$400,000, and one in the 1 category at around $50,000). This aligns with an initial hypothesis that this would be a key variable. This variable's box plot can be seen below:

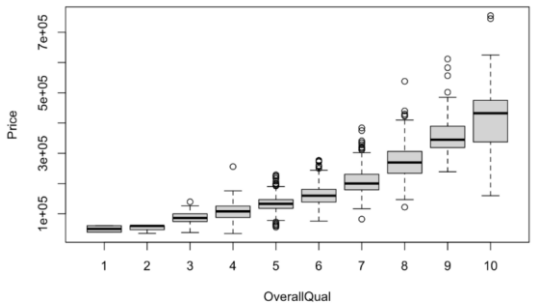


Figure 2 – Boxplots for each value of the *OverallQual* categorical variable. The Y axis represents the values of the response variable. In this example, we can clearly see that the category value has an impact on the range of the response variable.

The next four variables (in descending order) with large median ranges include:

1. *PoolQC: Pool quality*
2. *ExterQual: Evaluates the quality of the material on the exterior*
3. *Condition2: Proximity to various conditions (ie. Arterial street, railroad, park).*
4. *Neighborhood: Physical locations in Ames city limits.*

Our team did not expect *PoolQC* to be a particularly revealing variable and will keep this in mind during the variable selection process.

The five variables with the least amount of median variation between groups include:

1. *Street: Type of road access to property*
2. *BldgType: Type of dwelling (single family, duplex, etc.)*
3. *LotConfig: Lot configuration (corner lot, cul-de-sac, etc.)*
4. *Fence: Fence quality*
5. *Utilities: Type of utilities available*

This list of bottom dwellers isn't totally unexpected, and we plan to reference these when selecting variables.

Scatter plots for numerical variables

Similarly, to the previous section, our team also created two-dimensional scatter plots for all 36 numerical variables. The objective was to visually check whether a linear dependency with the response variable can be inferred. As an example, in the below diagram, we are inspecting the variable named ***MasVnrArea*** against the response variable (Sale Price), and we can see a linear dependency.

This activity enabled us to create a list of variables we believe must be included in the regression analysis.

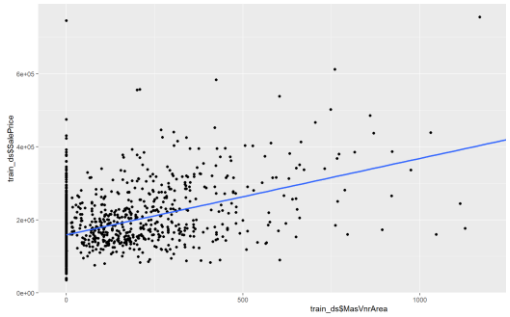


Figure 3 – Scatter plot of the explanatory variable *MasVnrArea* (X axis) against the response variable *SalePrice* (Y axis). In this case, the plot indicates a likely linear dependency between the two variables.

Checking for Multicollinearity

Based on the data dictionary, our team strongly suspected the existence of linear dependencies between the explanatory variables. Also, based on our knowledge of regression analysis, our team determined that it was necessary to perform an analysis on multicollinearity to avoid undesired results.

For that analysis we resorted to below three analytical tools:

- Correlation Matrix.
- Graphing correlations.
- Variance Inflation Ratios

First, our team calculated a large correlation matrix between the numerical variables in the dataset (a total of 36). The matrix had a very large number of correlation values which made it a tedious effort. To make our work easier, our team decided to create a graph that would visually highlight these strong correlations. The graph is depicted in figure 4.

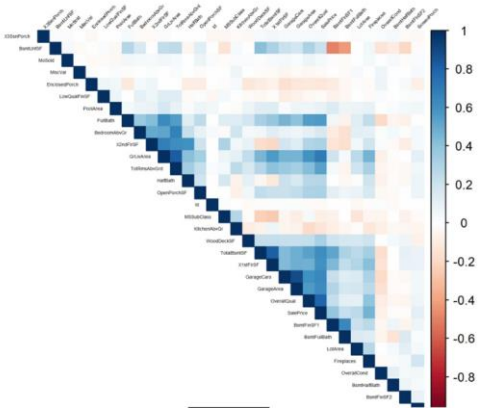


Figure 4 – Visual representation of a correlation matrix which makes it quicker to locate the dependencies. The numerical values are color coded. A darker blue color indicates a strong correlation.

The darkest blue squares indicate highest correlation. The worst offending pairs include:

- *GrLivArea vs. TotRmsAbvGrd* (above ground living area vs. Total rooms above ground)
- *TotalBsmtSF vs. 1stFlrSF* (total basement sq. footage vs. 1st floor sq.)
- *GarageCars vs. GarageArea* (car capacity of garage vs. sq. ft.)
- *OverallQual vs. SalePrice (house quality vs. price)*

These high correlations make intuitive sense. The number of rooms above ground is likely to increase with an increase in above-ground square footage. The same can be said for the car capacity of a garage and the garage area. And since the basement square footage is contained usually by the first-floor square footage, it makes sense that these are highly related. We will want to keep an eye out for these pairings when building our models, and we'll likely just want one variable from each pair in the model, not both. The *OverallQual vs. SalePrice* pairing is not as threatening since sale price is our dependent variable. The high correlation supports the inclusion of *OverallQual* in the model.

This analysis enabled our team to identify the sets of explanatory variables which are strongly correlated which guided our variable selection steps.

To confirm these results our team attempted to run a *Variance Inflation Factor (VIF)* calculation on the dataset. In our statistical package (R), the calculation returned a message indicating the presence of multicollinearity ("there are aliased coefficients in the model").

DATA PROCESSING

The dataset is split into two files, a training dataset, and a testing dataset. Each contains 1460 rows.

As a first step, our team combined the two datasets into one to perform the transformations in one shot. This combination also ensures that the two datasets are processed in the same way. After the processing was completed, the two datasets were separated again to serve their intended purpose.

In the processing pipeline, the data is first inspected for missing values, and if one is detected either imputation or deletion of the data point is performed. As a result, only 7 data points were deleted and nearly 14,000 missing values were inputted.

As the second step in the processing pipeline, indicator variables were created for the categorical variables. One indicator variable was created for nearly each category value. The appendix table lists all the variables created. Each category's base case is clearly documented in the appendix. As a result of that second step, the number of columns increased from 80 to 265.

Commented [JC1]:

Commented [AH2]:

One interesting finding is regarding houses with missing garage build years despite the data indicating those houses do in fact have garages. Initially, we attempted to impute these values by taking the mean as well as linear regression. However, in some cases, this resulted in garage build years predating their associated houses' build years. Therefore, we found no better alternative than to set those values to the maximum between the mean of the garage build year, the house build year and the linear regression model's prediction.

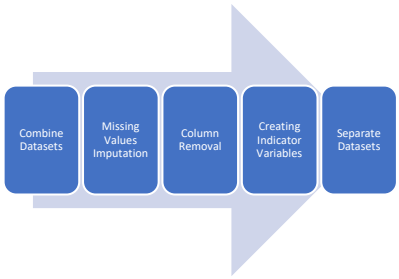
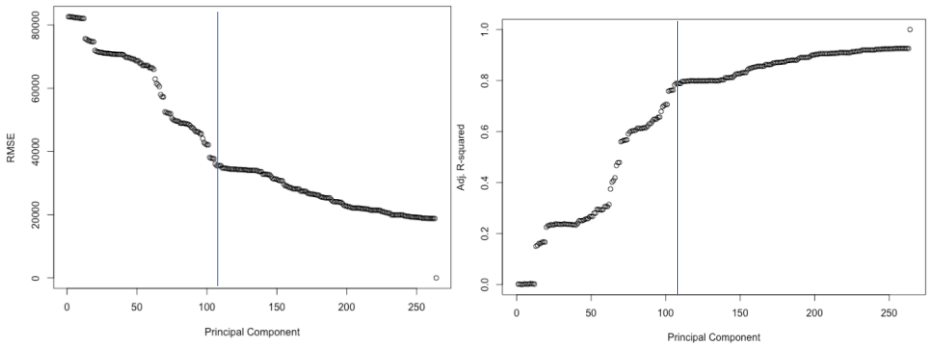


Figure 5– Data Preparation Pipeline

MODELING

Principal Component Analysis and regression

The first modeling approach to be discussed is Principal Component Analysis (PCA) with regression. Given the multitude of explanatory variables (265 after creating dummy variables for categorical features), PCA provides a computationally efficient alternative because it reduces dimensionality. However, the existence of categorical variables further complicated the PCA setup. We cannot apply PCA directly to the dummy variable columns, as this would not give similar weight to all variables over the calculated components.[4] Instead, we divided each value in the dummy column by the square root of the probability of the modality (the number of ones in the column divided by N rows in the dataset). This effectively scales dummy columns, which then allowed us to pursue a more statistically sound PCA transformation. We then iteratively built regression models using a different number of components each run and recorded adjusted R-squared and root mean squared errors. The plots can be viewed below:



The left chart shows the root mean square error (RMSE) by number of components used in the regression model. The right is similar, but substitutes R-squared for RMSE.

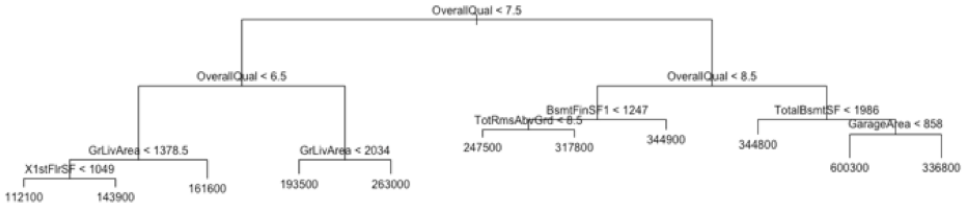
Surprisingly enough, the variation explained (indicated by R-squared) is quite low for the initial few components. Typically, the first few components contain much more explanatory power. This suggests some potential complications with the treatment of dummy variables.

In any case, we found the ideal number of components to be 102, which is indicated by the blue line on both charts. We selected this number in order to balance the benefits of dimensionality reduction and explanatory power. Regression models built with more than 102 principal components see nominal improvements in RMSE and adjusted R-squared. Results for those metrics are 38,106 and 0.7585, respectively. We still need 102 components to get a reasonable

R-squared value, and even then we lose insight into impactful variables. Given these facts, it's unlikely we'll select PCA and regression as the top analytical technique.

Decision Tree

The next technique explored was a regression decision tree using all variables. To recap, this technique recursively splits the original dataset, and then regression models are built on the leaf nodes to offer predictions based on the paired down datasets. The tree was then pruned and deviance scores for varying numbers of terminating nodes were cross-validated. Lower overall deviance was desired, as this signifies that predictions in each node are more alike. It was determined that 11 terminating nodes yielded the lowest amount of deviance in the training dataset ($1.9603 * 10^{12}$). The resulting tree is pictured below:



It was determined that splitting on the following variables reduced deviance the most:

- Overall house quality
- 1st floor sq. ft.
- Above ground living area
- Total rooms above ground
- Sq. ft. of basement if one of the type 1's finish
- Garage area
- Total basement sq. ft.

The tree and its splits give us some insight into important variables. Splitting is done to reduce impurity, which is to say that observations on one side of a split are more alike than compared to the other side. So if a variable proves decisive in reducing deviance, it's likely worth noting. This list mostly tracks with our hypothesized important variables, looking specifically at "Overall Quality" and "Garage area." The decision tree yielded an R-squared of 0.791 and RMSE of 45,186.52, an improvement on PCA regression, and also gave us insight into important variables.

Random Forest

The random forest we used employed 500 total decision trees. Leaf outputs were averaged across trees to give us final predictions. Because predictions are generated from out-of-bag samples, cross-validation is not necessary. Random Forest yielded an RMSE of 34,593 and R-squared of 0.795.

MLR with all features discussion

As a first step towards building linear regression models, we decided to build a model with all 264 predictor variables included in order to have a benchmark and be able to identify whether any adjustments lead to improvements in performance. This model resulted in a low adjusted R^2 value of 0.127 and an RMSE value of 112,061.

Stepwise Regression – Backward Selection

After building the MLR model above, we decided to optimize the number of features while improving the model's performance. In order to achieve this objective, we decided to utilize stepwise regression. Stepwise regression is an iterative method that adds and/or removes features based on their contribution to the model's performance. There are three common methods for applying Stepwise regression: Backward Selection, Forward Selection, and Bidirectional Selection.

We started with Backward Selection, which begins with all variables as predictors and removes least significant variables based on the highest p-value, the least impact on R-squared and the lowest residual sum of squares value. This iterative process continues until a minimum AIC value is reached, indicating the current feature set is optimal. This procedure resulted in a multiple linear regression model with 109 predictors, where we obtained an adjusted R^2 value of 0.263 and an RMSE of 106,328, which is a slight improvement over the previous model.

Stepwise Regression – Forward Selection

Then, we moved on to Forward Selection, which is a procedure that starts with no predictors and iteratively adds new features based on the smallest p-value, highest impact on r-squared and the most drop in residual sum of squares, with a stopping criterion similar to that of backward stepwise regression.

As a result of applying forward feature selection, we obtained an adjusted R^2 value of 0.812 and an RMSE of 35,680, which is a considerable improvement over the previous model's 0.263 adjusted R^2 value.

Stepwise Regression – Bidirectional Selection

After that, we tested Bidirectional Selection, which is a procedure that starts with no predictors and iteratively adds or removes new features based on the p-value as well as the impact on R^2 and the residual sum of squares, with a stopping criterion similar to that of Backward and Forward Selection.

As a result of applying backward feature selection, we obtained an adjusted r-squared value of 0.811 and an RMSE of 35,752, which is a slight drop when compared to forward selection

Lasso Regression

After successfully applying Stepwise Regression, we proceeded to Least Absolute Shrinkage and Selection Operator (LASSO). This technique aims to minimize the prediction error by shrinking certain coefficients to minimize their impact on the model, where some coefficients could get as low as zero.

As a result of applying LASSO, we obtained an adjusted R^2 value of 0.821 and an RMSE of 34,969, which is a slight improvement over the best result we have seen so far, via forward stepwise regression.

Lasso Regression – Log-Linear Transformation

Now that we have applied several feature selection techniques and found LASSO to result in the best model, we decided to use the same features recommended by LASSO, but by building a log-linear model. However, this attempt resulted in a sharp drop in the model's performance, with r-squared value of 0.323 and an RMSE of 66,948

Lasso Regression – Log-Log Transformation

Also, we tried performing a log-log transformation on the data set using the same features selected by LASSO. However, this still did not result in a better model when compared to the previous linear-linear model, with an adjusted R^2 value of 0.733 and an RMSE of 44,078.

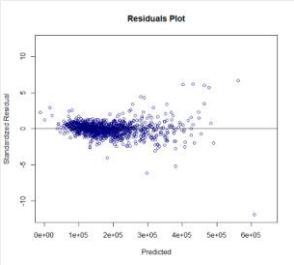
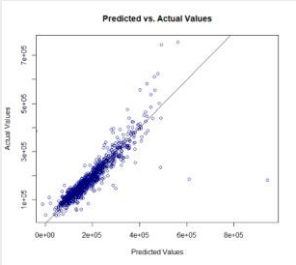
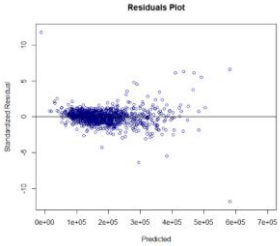
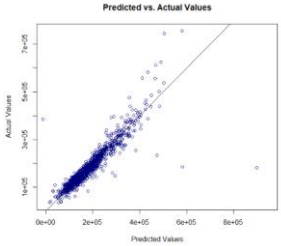
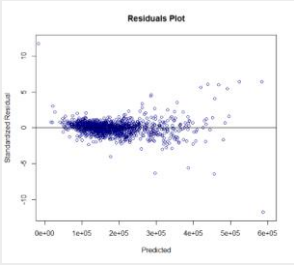
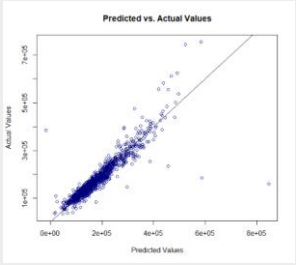
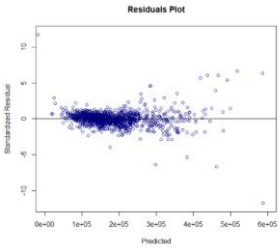
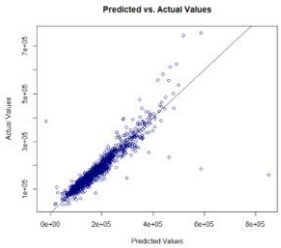
Lasso Regression – Linear-Log Transformation

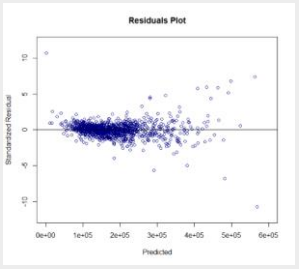
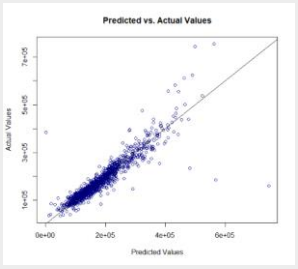
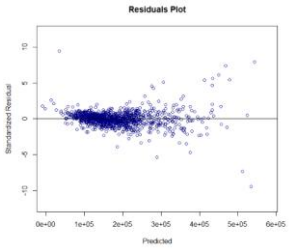
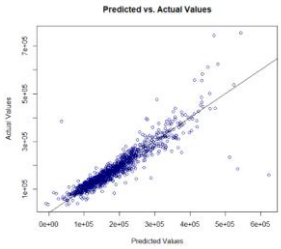
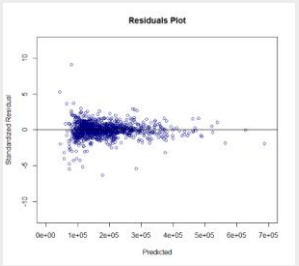
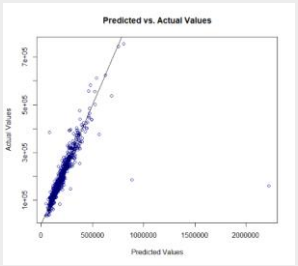
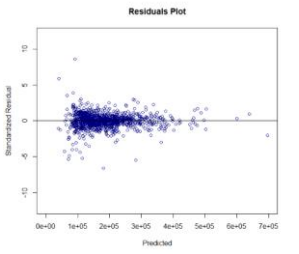
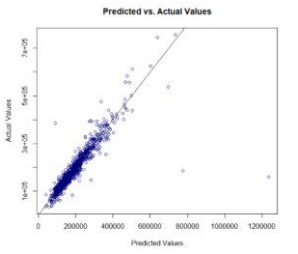
Last but most importantly, and as a final attempt to enhance the model's performance, we performed a linear-log transformation on the data set. This type of transformation resulted in the best model we have been able to achieve, with an adjusted R^2 value of 0.828 and an RMSE of 34,021.

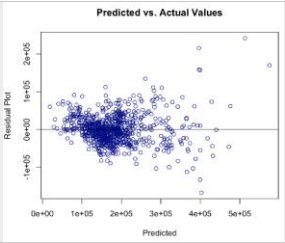
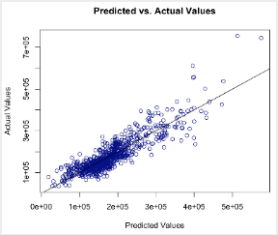
Comparison of models

The below table summarizes the results of the different models used in our analysis. We are comparing the models with their R-Squared and RMSE values.

Model	R-squared	RMSE
MLR with All Features	0.127 (adjusted)	112,061
MLR with Backward Feature Selection	0.263 (adjusted)	106,328
MLR with Forward Feature Selection	0.812 (adjusted)	35,680
MLR with Bidirectional Feature Selection	0.811 (adjusted)	35,752
Linear-Linear MLR with LASSO Feature Selection	0.821 (adjusted)	34,969
Linear-Log MLR with LASSO Feature Selection	0.828 (adjusted)	34,021
Log-Linear MLR with LASSO Feature Selection	0.323 (adjusted)	66,948
Log-Log MLR with LASSO Feature Selection	0.733 (adjusted)	42,078
MLR with PCA	0.758 (adjusted)	38,106
Decision Tree	0.791 (adjusted)	45,186
Random Forest	0.795 (pseudo , non-adjusted)	34,593

	Residuals Plot	Predicted vs Observed Plot
MLR with All Features		
MLR with Backward Feature Selection		
MLR with Forward Feature Selection		
MLR with Bidirectional Feature Selection		

Linear-Linear MLR with LASSO Feature Selection		
Linear-Log MLR with LASSO Feature Selection		
Log-Linear MLR with LASSO Feature Selection		
Log-Log MLR with LASSO Feature Selection		

MLR with PCA		
Decision Tree	N/A	N/A
Random Forest	N/A	N/A

CONCLUSION

After our selection and testing of different prediction models, our team concluded that the price of a house can be predicted with relatively good accuracy (RMSE $34K). Our analysis also gave us insights into the most significant predictors.

If we take a step back, we can now also conclude that the creation of an *Automated Valuation Model* is not beyond the reach of simple analytical models and techniques. However, our research has also indic, many questions remain unanswered because we discovered that such models are not infallible.

Indeed, as per the article from *Mike DelPrete* (ref citations), accuracy is critical to the survival of companies relying on these automated predictions. In the case of **Zillow**, the model didn't adjust rapidly enough to the economic context, which was necessary to avoid financial losses.

That event demonstrated that these models are far from perfect and must improve.

When reading the article of *Chris Stokel-Walker* (ref citations), and the thesis of Ms. *Michaela Hronová* (ref citations), it became clear that the below factors have a strong impact on the price of a house:

- Inflation
- Demand for Housing
- Federal Reserve Interest Rates
- Property Taxation
- Population Growth

When reflecting on this problem, our team raised a new question: if we could "feed" economic indicators into these prediction models, would it be possible to automatically adjust the predictions and avoid the challenge faced by Zillow?

As a next step to this project, our team is proposing to answer that question.

CITATIONS

1. Chris Stokel-Walker, “Why Zillow Couldn't Make Algorithmic House Pricing Work,” Wired (Conde Nast, November 11, 2021), https://www.wired.com/story/zillow-ibuyer-real-estate/.
2. Mike DelPrete, “IBuying Is Hard: Zillow Pauses New Purchases,” Mike DelPrete - Real Estate Tech Strategist (Mike DelPrete - Real Estate Tech Strategist, October 21, 2021), https://www.mikedp.com/articles/2021/10/19/ibuying-is-hard-zillow-pauses-new-purchases
3. Hronová, Michaela. "Determinants of Real Estate Prices in the United States." (2022), https://dspace.cuni.cz/bitstream/handle/20.500.11956/175612/130344362.pdf?sequence=1
4. Blaufuks, William. “FAMD: How to Generalize PCA to Categorical and Numerical Data.” *Towards Data Science*, Medium, 25 May 2021, https://towardsdatascience.com/famd-how-to-generalize-pca-to-categorical-and-numerical-data-2ddbeb2b9210.
5. Amy Fontinelle, Rachel Witkowski. “iBuyer: What Is It & Is It Worth It?”, FORBES Advisor, February 15th 2022, https://www.forbes.com/advisor/mortgages/what-is-ibuyer/

APPENDIX

Column	Numerical/ Categorical ?	NA Values?	NA Handling	Base Case (If Categorical)	Number data points deleted	Done ?
MSSubClass	Numerical	No	No NA Values	Numerical value	0	T
MSZoning	Categorical	Yes	Treated as new category (Other)	Other	0	T
LotFrontage	Numerical	Yes	Predict using LR	Numerical value	0	
LotArea	Numerical	No	No NA Values	Numerical value	0	T
Street	Categorical	No	No NA Values	grvl	0	T
Alley	Categorical	Yes	Treated as new category	No_Alley	0	T
LotShape	Categorical	No	No NA Values	Reg	0	T
LandContour	Categorical	No	No NA Values	Lvl	0	T
Utilities	Categorical	Yes	Data points deleted	AllPub	2	T
LotConfig	Categorical	No	No NA Values	Inside	0	T
LandSlope	Categorical	No	No NA Values	Gtl	0	T
Neighborhood	Categorical	No	No NA Values	CollgCr	0	T
Condition1	Categorical	No	No NA Values	Norm	0	T
Condition2	Categorical	No	No NA Values	Norm	0	T
BldgType	Categorical	No	No NA Values	1Fam	0	T
HouseStyle	Categorical	No	No NA Values	1Story	0	T
OverallQual	Numerical	No	No NA Values	Numerical value	0	T
OverallCond	Numerical	No	No NA Values	Numerical value	0	T
YearBuilt	Numerical	No	No NA Values	Numerical value	0	T
YearRemodAdd	Numerical	No	No NA Values	Numerical value	0	T
RoofStyle	Categorical	No	No NA Values	Flat	0	T
RoofMatl	Categorical	No	No NA Values	WdShngl	0	T
Exterior1st	Categorical	Yes	Data points deleted	WdShing	1	T

Exterior2nd	Categorical	Yes	Automatically deleted	Wd Shng	0	T
MasVnrType	Categorical	Yes	One deleted, other None	None	1	T
MasVnrArea	Numerical	Yes	Set to zero	Numerical value	0	T
ExterQual	Categorical	No	No NA Values	TA	0	T
ExterCond	Categorical	No	No NA Values	TA	0	T
Foundation	Categorical	No	No NA Values	Wood	0	T
BsmtQual	Categorical	Yes	Treated as new category	No_Bsmt	0	T
BsmtCond	Categorical	Yes	Treated as new category	No_Bsmt	0	T
BsmtExposure	Categorical	Yes	Treated as new category	No_Bsmt	0	T
BsmtFinType1	Categorical	Yes	Treated as new category	No_Bsmt	0	T
BsmtFinSF1	Numerical	Yes	Data point deleted	Numerical value	1	T
BsmtFinType2	Categorical	Yes	Treated as new category	No_Bsmt	0	T
BsmtFinSF2	Numerical	Yes	Automatically deleted	Numerical value	0	T
BsmtUnfSF	Numerical	Yes	Automatically deleted	Numerical value	0	T
TotalBsmtSF	Numerical	Yes	Automatically deleted	Numerical value	0	T
Heating	Categorical	No	No NA Values	OthW	0	T
HeatingQC	Categorical	No	No NA Values	TA	0	T
CentralAir	Categorical	No	No NA Values	N	0	T
Electrical	Categorical	Yes	Data point deleted	SBrkr	1	T
1stFlrSF	Numerical	No	No NA Values	Numerical value	0	T
2ndFlrSF	Numerical	No	No NA Values	Numerical value	0	T
LowQualFinSF	Numerical	No	No NA Values	Numerical value	0	T
GrLivArea	Numerical	No	No NA Values	Numerical value	0	T
BsmtFullBath	Numerical	Yes	One data point automatically deleted, one set to zero	Numerical value	0	T
BsmtHalfBath	Numerical	Yes	One data point automatically deleted, one set to zero	Numerical value	0	T

FullBath	Numerical	No	No NA Values	Numerical value	0	T
HalfBath	Numerical	No	No NA Values	Numerical value	0	T
BedroomAbvGr	Numerical	No	No NA Values	Numerical value	0	T
KitchenAbvGr	Numerical	No	No NA Values	Numerical value	0	T
KitchenQual	Categorical	Yes	Data point deleted (only one)	TA	1	T
TotRmsAbvGrd	Numerical	No	No NA Values	Numerical value	0	T
Functional	Categorical	Yes	Converted to "Sal" as per description	Typ	0	T
Fireplaces	Numerical	No	No NA Values	Numerical value	0	T
FireplaceQu	Categorical	Yes	Treated as new category	No_Fireplace	0	T
GarageType	Categorical	Yes	Treated as new category	No_Garage	0	T
GarageYrBlt	Numerical	Yes	max(mean yr, LR, build year)	Numerical value	0	T
GarageFinish	Categorical	Yes	Treated as new category	No_Garage	0	T
GarageCars	Numerical	Yes	Set to zero (only one data point & w/o garage)	Numerical value	0	T
GarageArea	Numerical	Yes	Set to zero (only one data point & w/o garage)	Numerical value	0	T
GarageQual	Categorical	Yes	Treated as new category	No_Garage	0	T
GarageCond	Categorical	Yes	Treated as new category	No_Garage	0	T
PavedDrive	Categorical	No	No NA Values	N	0	T
WoodDeckSF	Numerical	No	No NA Values	Numerical value	0	T
OpenPorchSF	Numerical	No	No NA Values	Numerical value	0	T
EnclosedPorch	Numerical	No	No NA Values	Numerical value	0	T
3SsnPorch	Numerical	No	No NA Values	Numerical value	0	T
ScreenPorch	Numerical	No	No NA Values	Numerical value	0	T
PoolArea	Numerical	No	No NA Values	Numerical value	0	T
PoolQC	Categorical	Yes	Treated as new category	No_Pool	0	T
Fence	Categorical	Yes	Treated as new category	No_Fence	0	T

MiscFeature	Categorical	Yes	Converted to "None" as per description	None	0	T
MiscVal	Numerical	No	No NA Values	Numerical value	0	T
MoSold	Numerical	No	No NA Values	Numerical value	0	T
YrSold	Numerical	No	No NA Values	Numerical value	0	T
SaleType	Categorical	Yes	Imputed using "mode"	WD	0	T
SaleCondition	Categorical	No	No NA Values	Normal	0	T
SalePrice	Numerical	Yes	Dependent variable	Numerical value	0	No need