MLND Capstone Project: Predicting Ames Housing Prices

Gendith M. Sardane

admin@adalace.org || *Github:gmsardane*

October 5, 2016

Abstract

I extend and update the Boston Housing regression problem, using the more contemporary dataset for homes sold in Ames, IA from 2006-2010. The regression model was built using Gradient Boosted decision trees. The final optimal model was trained using 1000 weak estimators and a learning rate of Ł $= 0.1$. The training data which has 1460 examples was split so that 80% was alloted for training, while the remaining 20% to be used for validation. The optimal model yields an $R^2 = 0.98$ and an RMSE = 7.23×110^7 USD. In comparison, the optimal model for the MLND project on the Boston housing price prediction, which used a decision tree regressor, yielded an R^2 = 0.80 and an RMSE = 6.78×10^9 USD.

1 Introduction

As a first project for the MLND, we were tasked to predict the prices of homes located in various suburbs in Boston, MA. The Boston housing dataset (Harrison and Rubinfeld 1978) was collected in 1978. It contains roughly 500 examples, each associated to 14 features. For the project, we were only given four variables, namely, the median sale price, student-teacher ratio, poverty ratio and the number of rooms in the home. Furthermore, in an attempt to make the data catch up with contemporary home values, each entry was *multiplicatively* scaled to account for the $>$ 30-year gap in the housing data. Due to the 30-year data gap and the feature set available, I was quite dissatisfied with such an analysis. In my opinion, a simple scaling such as with a multiplicative factor fails to capture the intricacies of how the value of a home evolves across time. As I am planning to buy a home very near in the future, I was naturally quite curious about the statistics and how various aspects of a house factor into the final price tag. Needless to say, I am in search for a real, more recent dataset.

For my capstone project, I desire an updated version of such an example of a regression problem. As it turns out, Kaggle currently hosts a competition on predicting the prices that a house sells using the Ames, IA housing dataset (De Cock 2011). The competition is live from Aug 30, 2016 through Mar 1, 2017.

For this analysis, I use the data Kaggle tabulated in "train.csv" to train a **gradient boosted decision tree** to predict home values. In this report, I will present the

results and the process that I took in finding the optimal model that would best predict the prices of homes sold from 2006-2010 in Ames, IA.

The paper is organized as follows: In §2, I will give a brief background on the dataset. In §3, I describe the Gradient Boosting regression algorithm. The statistical properties of some major features in the dataset, and the feature selection process will be presented in §4. I then present the main results in §5. I end with some concluding remarks in §6.

2 The Ames Housing Dataset

The main reference for the Ames housing dataset is due to De Cock (2011). In brief, the dataset is a compilation of attributes pertinent to more than 2900 properties sold between 2006 and 2010 in various locations in Ames, IA. The data originally came as a data dump from the City's Assessor's Office of Ames, IA. De Cock (2011) collated the dataset into a collection of 80 features directly related to property sales, including the sale price. In particular, the data consist of 23 nominal, 23 ordinal, 14 discrete, and 20 continuous explanatory variables. As with the Boston dataset, the target variable is the "Sale Price", which a continuous quantity representing the monetary value of the home in US dollars (USD).

For the purposes of the competition, Kaggle divides the dataset into two sets of roughly equal sizes. The first set (**train.csv**), which contains 1460 entries and 80 features, is intended for training. The second set (**test.csv**), which is meant for validation, contains 1459 entries and 79 features, with the property sale price withheld.

Appendix A summarizes these quantities, grouped according to their respective data classes, the notations used for column nomenclature, as well as whether these features were used in the search for an optimal regression model.

3 The Gradient Boosting Regression

Gradient boosting (GB) regression is one type of ensemble machine learning technique. In constructing regression models, the algorithm consists of "collections" of (weak) regressors, typically decision trees. Here, the learners are learned sequentially, as new models are fit for a more accurate estimate of the response variable. The new-base learners are "constructed to be maximally correlated with the negative gradient of the loss function" for the whole ensemble (Natekin & Knoll 2013). In general, the loss function (LF) can be arbitrarily specified (e.g. squared error, absolute error). At each stage a regression tree is fit on the negative gradient

of the loss function. For the classic case where LF is the squared-error loss, the learning procedure entails consequentially optimizing the residual error.

To illustrate the algorithm, refer to Figure 5. In this example, the data is fit using **five** (weak) decision tree regressors at each step. Each panel in the top row illustrates the cumulative improvement due to the inclusion of the results from the previous steps. The residuals at each stage are plotted in the bottom row. The panels in Figure 5 are numerically labeled to illustrate the succession of steps relevant to fitting a gradient boosted decision tree to the example data.

The first step is to fit a weak decision tree regressor to the data, as shown in the first panel at the top (i.e Panel "1"). Notice that the model is quite too simple to capture the intricate details of the data. The residuals of the resulting fit in Panel 1 is shown in Panel 2 (bottom leftmost). A new decision tree is then fit onto the residuals in Panel 2. The sum of the decision trees in Panel 1 and Panel 2 are then added to create the decision tree regressor in Panel 3. The process is repeated three more times as decision trees are fit to the residuals in Panels 4, 6 and 8. The final model, shown in Panel 9, is then a (weighted) combination of all previous learners.

Among the main advantages of using a GB regressor are as follows: (1) works on data having features on different scales; (2) supports arbitrary loss functions, which it *directly* optimizes; and (3) is robust and often the best possible model. [1,2]

On the other hand, major drawbacks include: (1) its sensitivity to noise and extreme values; (2) slow to train (but fast to predict); (3) and the several hyperparameters one needs to carefully tune to find the optimal model.[1,2]

The main model parameters which need fine-tuning are: (1) the number of estimators, $n_{estimators}$; (2) the tree depth; and (3) the learning rate, Ł. The number of trees, or weak learners, is controlled by $n_{estimators}$. The tree depth is controlled by max_depth. The contribution of each learner to the final model is controlled by the learning rate (also called the shrinkage) parameter, Ł. Empirically, a small Ł (i.e. Ł < 0.1) leads to better generalization, compared to that in the absence of shrinkage (Ł $= 1$). Models with smaller learning rates, however, require larger number of estimators, leading to longer computational times.[3]

In Python, the gradient boosting regressor is implemented under the sklearn's *sklearn.ensemble* package. Commercial web search engines as Yahoo and Yandex have used variants of gradient boosting in learning ranking functions for web

[1] M. Landry (2015)
[2] P. Prettenhofer (2015)
[3] S. Saita 2016

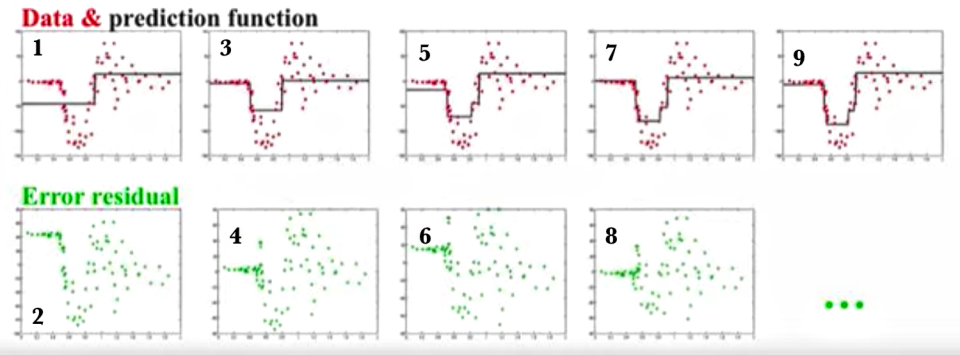

Figure 1: An illustration [4]of gradient boosting fitting. *Top:* Data (red points) and prediction function (black curve). *Bottom:* The error residuals shown as green points. *See text for discussion.*

search (Z. Zheng et al. 2007).

4 Methodology

4.1 Statistical Properties of the Ames Housing Data

Figure 2 shows the distribution of the sale prices of homes sold in Ames, IA from 2006-2010. The box plot representation of the sale prices at the top panel shows a few outliers both at the low and high price range. As depicted by the black curve overlaid on the histogram, the distribution is non-gaussian, with a slight positive skew. From 2006-2010, the cheapest home sold for $34,900, while the fanciest home sold for $755,000. The average home is valued at $181,000. The median price is slightly less at $163,000.

In terms of the size of the properties, Figure 3 shows the distribution of the of the land areas of these properties. An average property had a land area of 11,000 ft^2. The total land area of the properties ranged from 1300 - 215000 ft^2. For comparison, the most expensive property had a total area of 21,535 ft^2.

In addition to continuous home variables, the general behavior of some home features which are essential drivers of home prices, but which can only take discrete and categorical values, is explored. Examples include the number of half and full-size bathrooms, bedrooms, car capacity of the garage, and home quality indicators. Figure 4 shows how the sale prices vary as a function of these features. Here we see that the sale price is independent of the month the house was sold, but varies strongly as the total number of rooms and its overall condition grade. Furthermore, the older the home the lower the price.

Figure 5 shows the general relationship between the sale price with some of the

[4]Annotated figure grabbed from the GBT regression discussion video by A. Ihler.

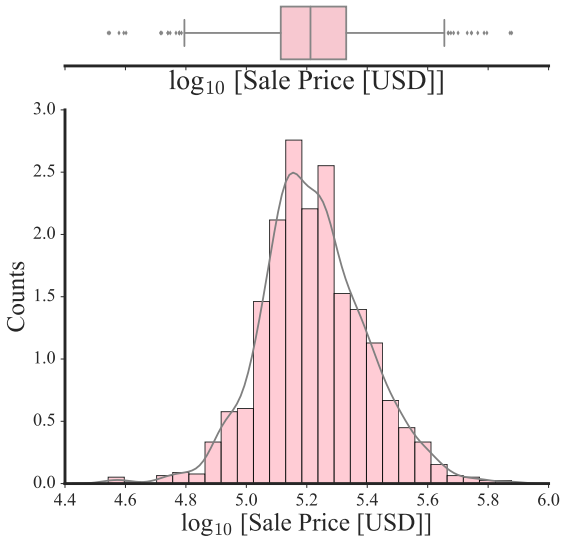

Figure 2: The $\log_{10}$ distribution of the sale prices of properties sold in Ames, IA from 2006 −2010. *Top:* A box plot representation of the prices showing a number of outliers. *Bottom:* The distribution of the of the prices, overlaid with a positively skewed non-normal fit to the distribution.

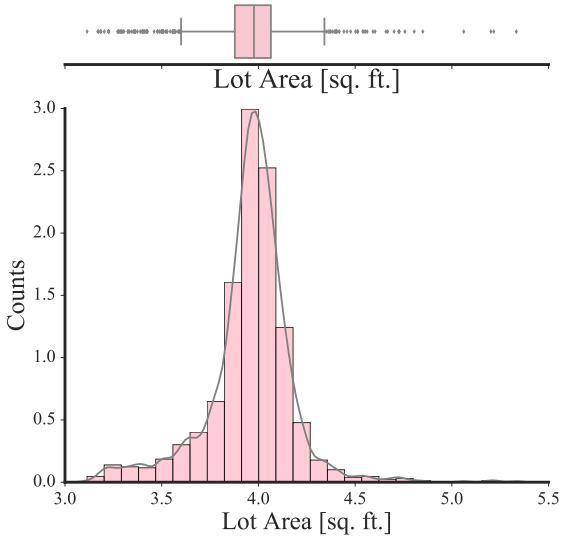

Figure 3: The $\log_{10}$ distribution of the land areas of properties sold in Ames, IA from 2006 −2010. *Top:* A box plot representation of the areas, showing a couple of outliers. *Bottom:* The distribution of the logarithm of the land areas in square feet, overlaid with a negatively skewed non-normal fit to the distribution.

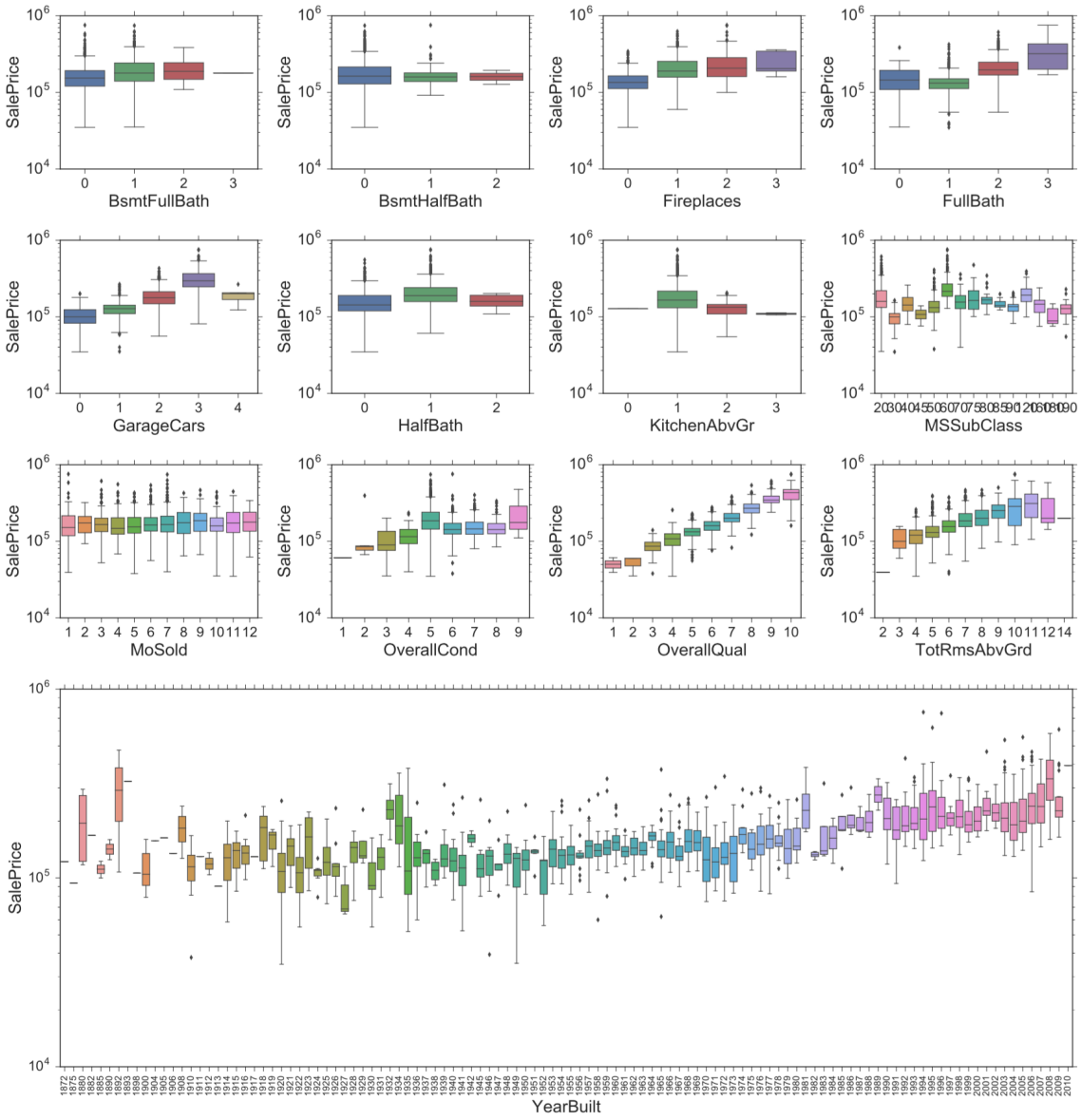

Figure 4: Visualizing correlations of discrete features in the Ames housing data with the sale price. Strong correlations with the sale price of some features such as the overall quality and total rooms of a house are obvious. The month the house was sold shows no apparent influence on the value of a home.

categorical features in the dataset. Here, different neighborhoods result to varying home values. As with intuition, homes with kitchens, garages, pools, fireplaces and heating in excellent condition demand higher prices. Land contour, shape and slope do not generally affect property values.

These exploratory results would be useful in determining which features and more important and more relevant relative to others. As many of these features are categorical, efficiently selecting which features to include in training would greatly help in reducing the dimensionality of the problem. The next section would discuss which features would be included in this analysis and the reasoning behind.

4.2 Feature Selection

4.2.1 On Continuous Features

To be more discerning of the features in a multi-variable regression analysis, it would be useful to examine the correlations between home features and the sale price. This starts by examining how continuous features correlate with the sale price, and with all other continuous features. Figure 6 summarizes the Pearson correlation coefficients between any two continuous features. The red indicates a positive correlation, while blue corresponds to anticorrelation. The darker the hues, the stronger correlations (or anti-correlations).

Most notably, Figure 6 tells us that the year the garage was built (GarageYrBlt) is strongly related to the year the house was built. This demonstrates that whenever a house was built, a garage was also built. The total basement area (TotalBsmtSF) stronly relates to the first floor area (1stFlrSF), implying that larger basements lead to more space for the first floor (and intuitively, vice-versa). Moreover, based on the Pearson coefficients with respect to SalePrice, we could **exclude the following variables: BsmtFinSF2, LowQualFinSF, EnclosedPorch, 3SsnPorch, ScreenPorch, MiscVal and PoolArea**.

4.2.2 Non-Continuous Features

With insights from Figures 4 & 5, it is apparent, at least qualitatively, that some features are less relevant than others. From Figure 4, the cost of the home is unlikely to be sensitive to the number of half-baths and kitchens, and the month the home was sold. From Figure 5, the neighborhood is a likely cost-driver, as well as the quality and condition of various sections of a house.

With insights from Figures 4-6, and after iterating through various combinations of relevant features, I find that the following feature set gives a concise and con-

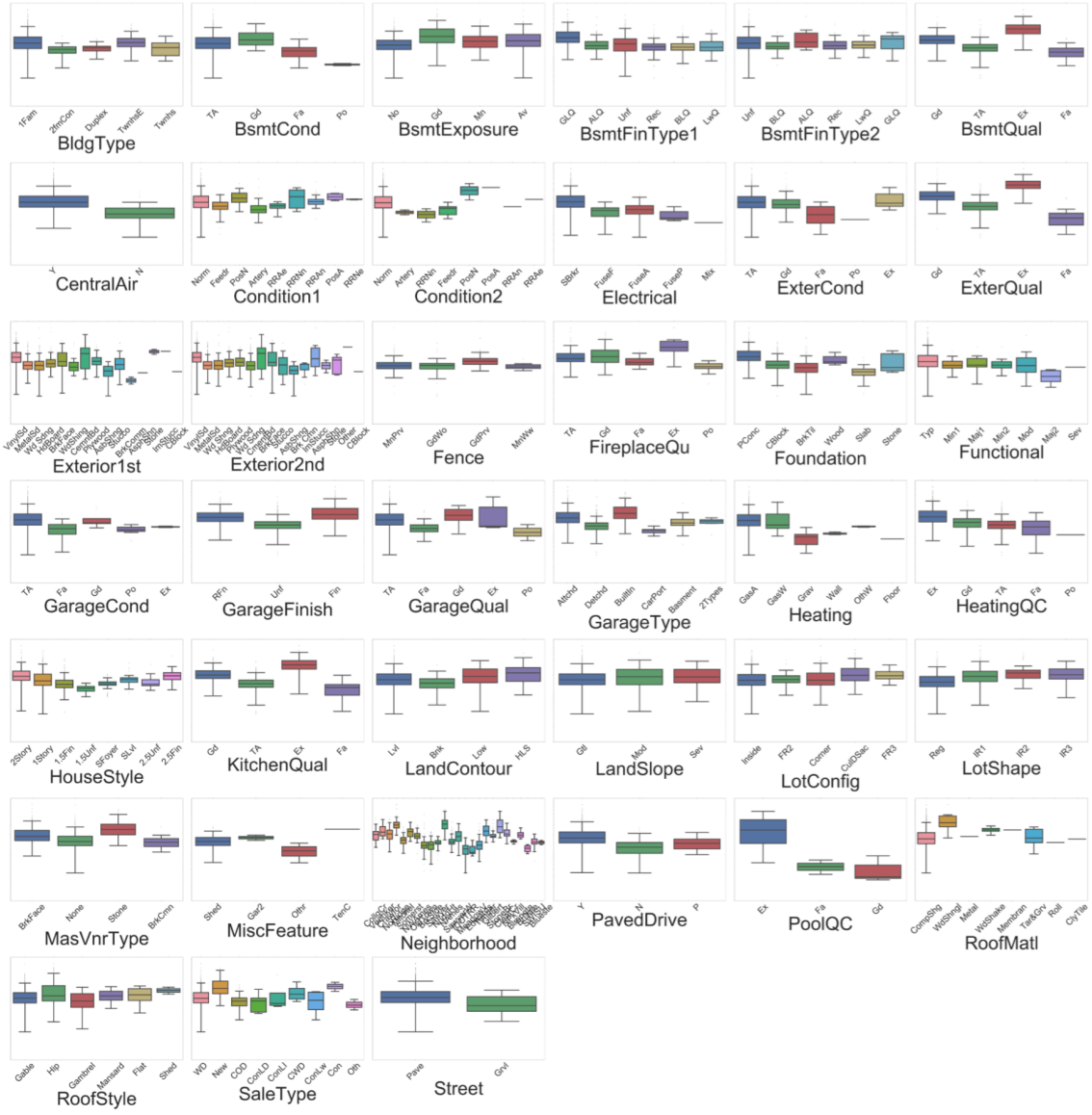

Figure 5: Visualizing correlations of categorical features in the Ames housing data with the sale price. The quality of garage, heating, and exterior of a house show some level of correlation with the sale price. Land attributes such as contour, slope and shape of the property show no apparent influence.

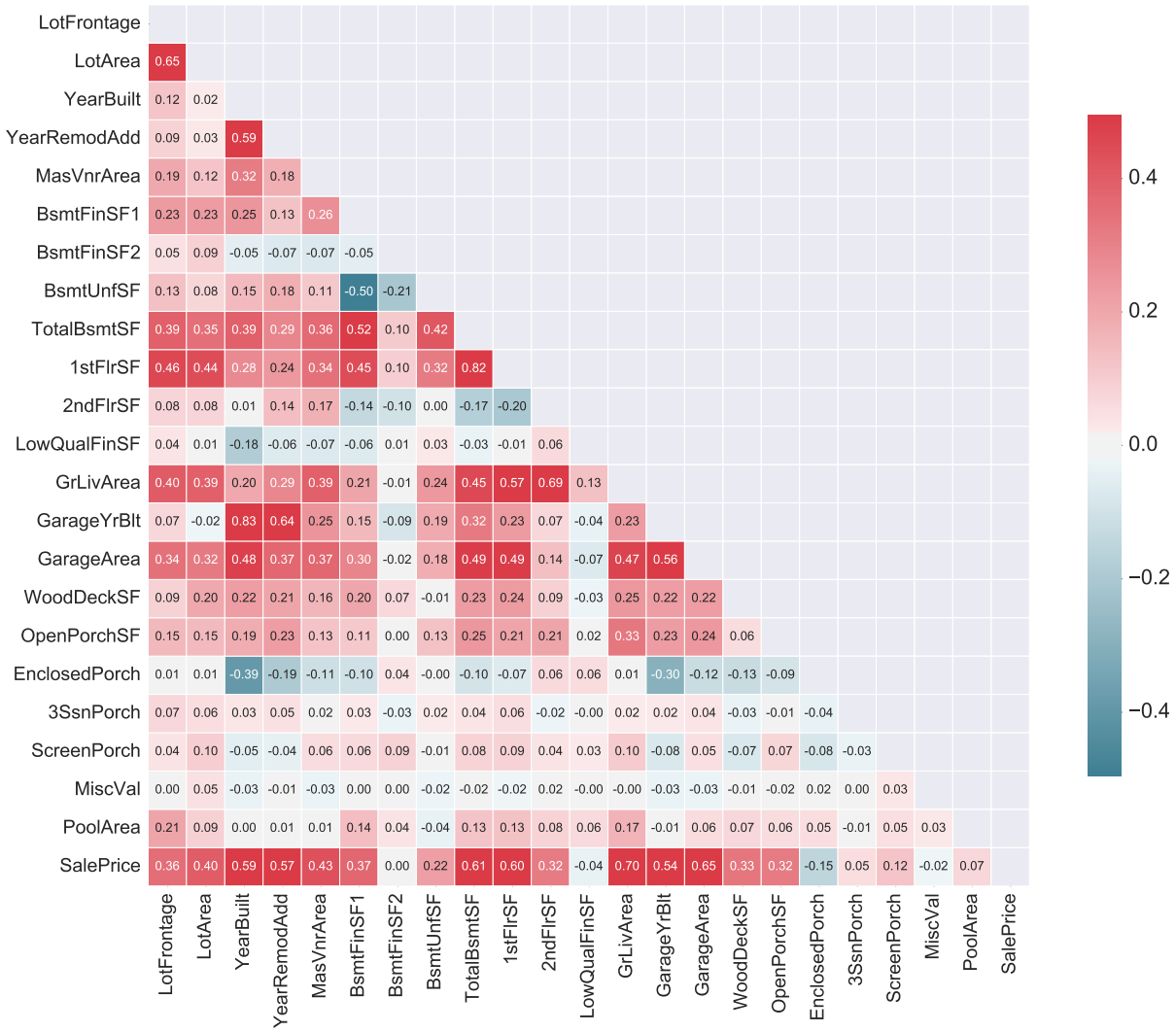

Figure 6: A heatmap of the Pearson correlation coefficients between pairs of features in the dataset. The redder (bluer) the hue, the stronger the correlations (anti-correlations). For this analysis, the SalePrice is the target. If the correlation coefficient between two non-target feature is large (i.e. $\rho \geq 0.6$), then the feature that correlates more with the target variable is selected.

Table 1: Selected features for regression.

Regression Feature Set			
Continuous			
BsmtFinSF1	GarageArea	GrLivArea	LotArea
MasVnrArea	OpenPorchSF	SalePrice	TotalBsmtSF
WoodDeckSF	YearBuilt	YearRemodAdd	···
Discrete			
BedroomAbvGr	Fireplaces	FullBath	GarageCars
GarageCond	GarageQual	Heating	TotRmsAbvGrd
PavedDrive	PoolQC	SaleCondition	···
Categorical			
BsmtQual	Condition1	Foundation	ExterQual
HeatingQC	KitchenQual	Neighborhood	MSZoning
OverallCond	OverallQual	RoofStyle	···

siderably better performance than the Boston housing prediction model:

4.3 Preprocessing

Thanks to Kaggle, little preprocessing of the data is involved for this analysis. As far as this project is concerned, there were only two preprocessing steps involved. First, it is important to carefully understand what the NaN's actually signify for a given feature, before these examples be excluded. Second is the transformation of categorical features into dummy variables using **pandas.get_dummies()** function. For all columns in this dataset, an NaN indicates an *absence* of that component in a home. For instance, the absence of a pool would imply NaNs for both PoolQC and PoolArea. Hence, an NaN for a numerical column is replaced by **zero**. For categorical features, these are replaced by the code **"nothing"**, which now becomes an additional feature in itself. It is worthwhile to consider that the addition of categorical variables into the feature set increases the dimensionality of the data by the number of unique elements of that category. Thus, a careful and thorough selection of these features is indispensable.

4.4 Model Selection and Evaluation

The impressive advantages of using gradient boosted decision trees discussed in §3 led me to choose the GB regressor to solve the Ames Housing prediction problem. Note, however, that prior to the final choice of algorithm, two other ensemble regression methods were tested, namely, the random forests and Adaboosted de-

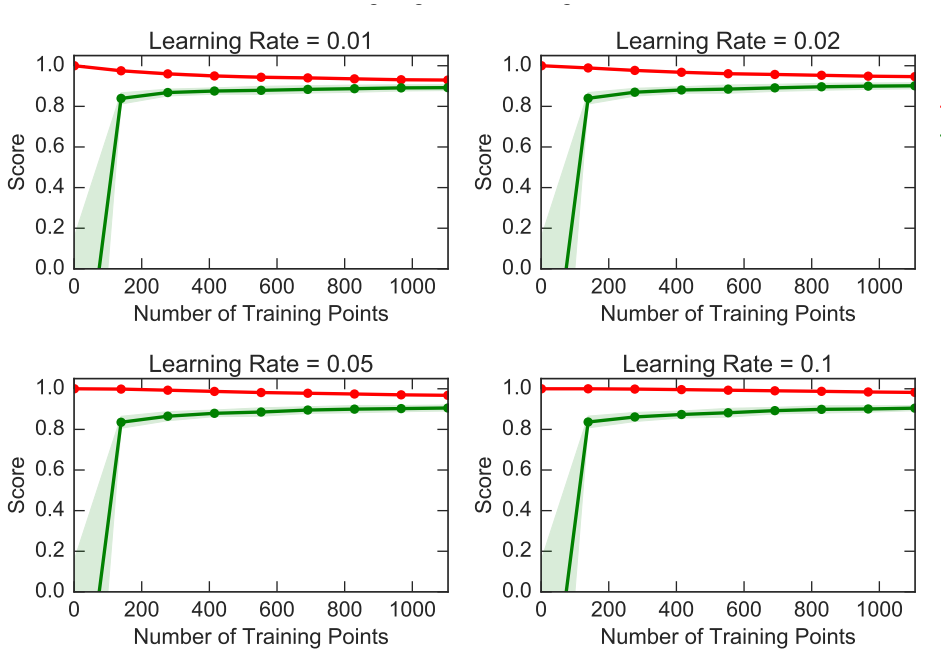

Figure 7: Learning curves as a function of the training size for various values of learning rates. The red curve is for training, while the green curve is for testing. At at least 1000 training points, a learning rate of Ł = 0.02 gives the best performance. As the validation score still rises, albeit slowly, a larger training size could still prove useful for these example values of Ł.

cision trees. These resulted to a much poorer performance in terms of the R^2 and RMSE of the test set. §5 will tackle more of this.

Even with over 1400 examples at our disposal, it is imperative that we explore what the optimal training size would be for various parameters of the GB regressor. The learning curves of Figures 7 & 8 imply that given the feature set of Table 1, excellent performance will be achieved using about 1000 weak learners at a learning rate of 0.02, using roughly 1000 examples. Such set-up would require a train-test split of 80% for training, and the remaining 20% to be used for testing. Note that the scoring metric is via the usual R^2 (coefficient of determination) regression score function.

Figure 9 zooms into the various learning rates using ($\max_{depth}$, $\min_{samples\_leaf}$, $\min_{samples\_split}$, $n_{estimators}$) = (2, 3, 2, 1000). An Ł = 0.02 gives the optimal model. At Ł $\gtrsim$ 0.02, the validation score saturates, and the model begins to overfit the data.

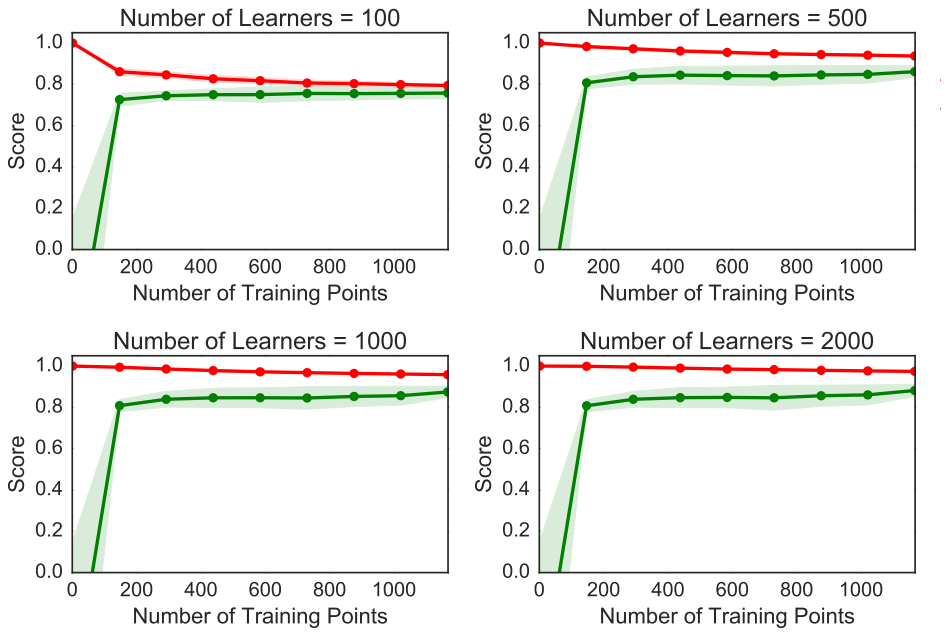

Figure 8: Learning curves for various values of $n_{estimators}$. For the parameter set ($\max_{depth}$, $\min_{samples\_leaf}$, $\min_{samples\_split}$, Ł) = (2, 3, 2, 0.1), the optimal model is given by $n_{estimators} = 1000$. (*Same color code as Figure 7.*)

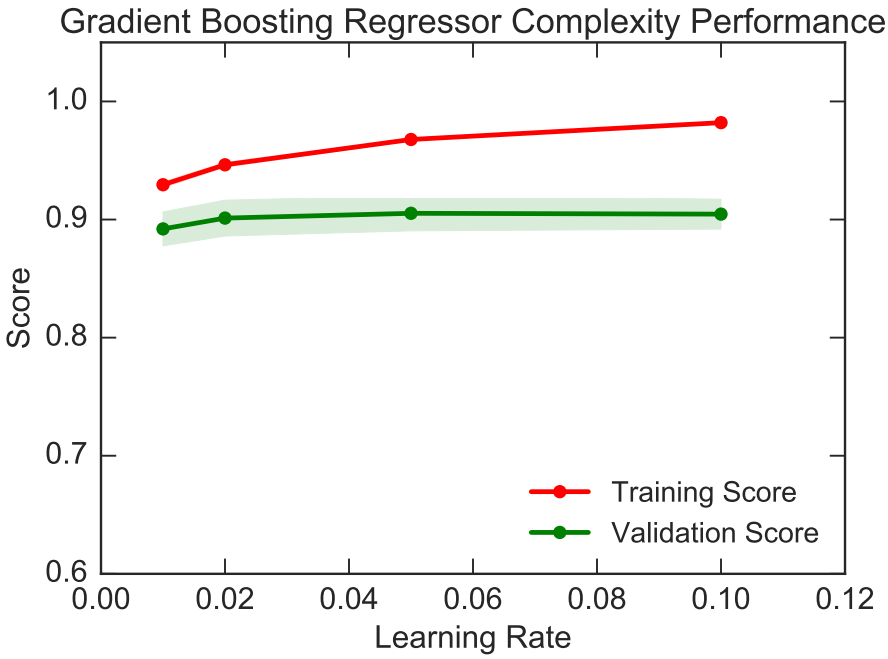

Figure 9: Learning curves for various values of learning rates. For the set of parameters as in Figure 8 case, Ł = [0.02, 0.05] would be optimal.

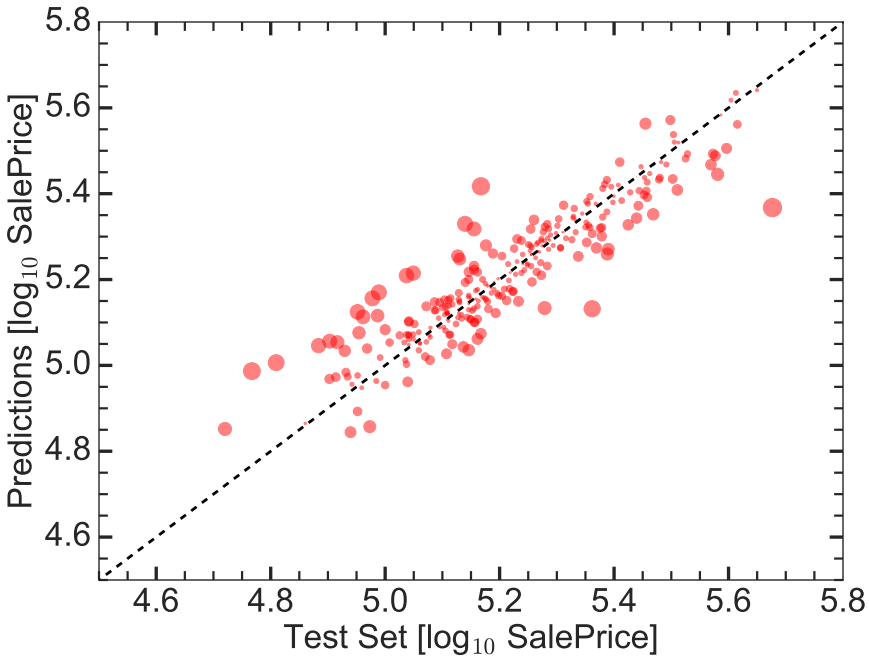

Figure 10: True sale price versus the model's prediction of the sale price, using *only the continuous variables* in Table 1. The perfect fit is depicted by the dashed line for reference. For emphasis, the size of the dots is proportional to the absolute difference between the prediction and the true sale price.

5 Results and Discussion

5.1 A Quick Check

In the pursuit of exploiting as few number of features as possible, one might wonder whether simply using the set on continuous variables listed in Table 1 would suffice. An 80-20 split results in 1168 samples for training and 292 for validation. Performing a grid search using Python's **GridSearchCV**, results to model where ($\max_{depth}$, $\min_{samples\_leaf}$, $\min_{samples\_split}$, $n_{estimators}$, Ł) = (4, 2, 2, 1000, 0.01). In log-space, this model yields an $R^2 = 0.81$, a root-mean-square-error, RMSE $= 4.36 \times 10^{-03}$, and a cross-validation score of 0.85 (out of 1.00). Removing the 74 *repeat* outliers[5] is no better at an $R^2 = 0.80$, and an RMSE $= 4.49 \times 10^{-03}$. Figure 10 illustrates the predictions as a function of the true price for the case which includes outliers. For comparison, a random forest regression and Adaboosted trees on the same set of features yield an $R^2 \sim 0.6 - 0.7$.

[5]Outliers are defined according to Tukey's rule: A data point with a feature that is beyond an outlier step ($\equiv 1.5 \times IQR$) outside of the IQR for that feature is considered abnormal.

5.2 Training on the Feature Set of Table 1

Now, will the inclusion of non-continuous variables improve the performance of the model?

Transforming categorical variables into dummies, the full set of Table 1 has now ballooned into a total of 123 features. Without those entries which are repeat outliers, the optimal model is given by ($\max_{depth}$, $\min_{samples\_leaf}$, $\min_{samples\_split}$, $n_{estimators}$, Ł) = (2, 2, 2, 1000, 0.05). The predictions are shown in Figure 11 against the true values. Based on an $R^2 = 0.88$ and RMSE = 5.14×10^8, the model demonstrates quite a significant improvement. We can further investigate the accuracy of predictions relative to the true value. Figure 12 shows the distribution of the percent difference relative to the true value. This distribution has a mean of $\mu = -2.74\%$ and a standard deviation of $\sigma = \pm 14.18\%$. The negative mean and fat tail towards the left are both indicative of the model being more under-predictive than otherwise. The worst of these cases under predicts the sale price by as much as twice (100% of) the true value.

Figure 13 depicts the distribution of the sale prices on the left axis. The right axis of this plot illustrates the percent difference as a function of the sale price. Here, we can clearly identify four points the model performed the worst: two at $\simeq \$50,000$, and one each at $\simeq \$100,000$, and $\simeq \$250,000$.

5.3 More checks

The relentless pursuit of achieving an even better performance led me to verify that *simply* plugging-in the entire set of 304 features should be no better than what has been achieved upon thorough consideration of the feature set. [6]

Two cases were implemented, one for the whole sample set, and the other which excludes those examples which are outliers in at least two continuous features (repeat outliers). The former resulted to a comparable $R^2 = 0.90$ and an RMSE that is 10% worse.[7] I was pleasantly surprised by the latter, however, as this yields an **even better** performance.

Figures 14-15 depict the predictions versus the true home values, after training the full feature set, and the removal of chronic outliers. The sizes of the circles are scaled in the same manner as Figure 11. The mean percent difference is $\mu = -0.47\%$ and spread of $\pm\sigma = 4.58\%$. Though still leaning on the under-predictive side, it is clear that the predictions are more accurate and much tighter, by as much

[6] The code and results are documented in a separate NB (*Training_Whole_Features_Set_GBT.ipynb*).

[7] For brevity the plots are not shown here. Refer to the notebook instead.

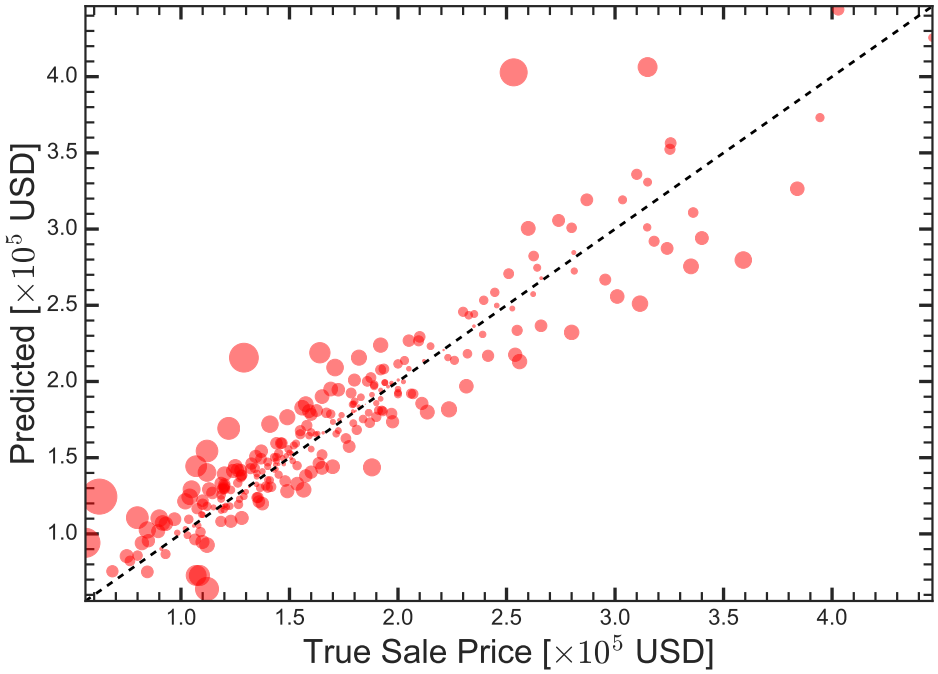

Figure 11: The predicted price vs. **true** sale price in units of $100,000. There were 276 testing examples, derived from an 80−20 training-to-testing split. The black dashed line denotes the $y = x$ line (a perfect fit). For emphasis, the sizes of the circles are 1000 times the absolute difference between the predicted and the true values.

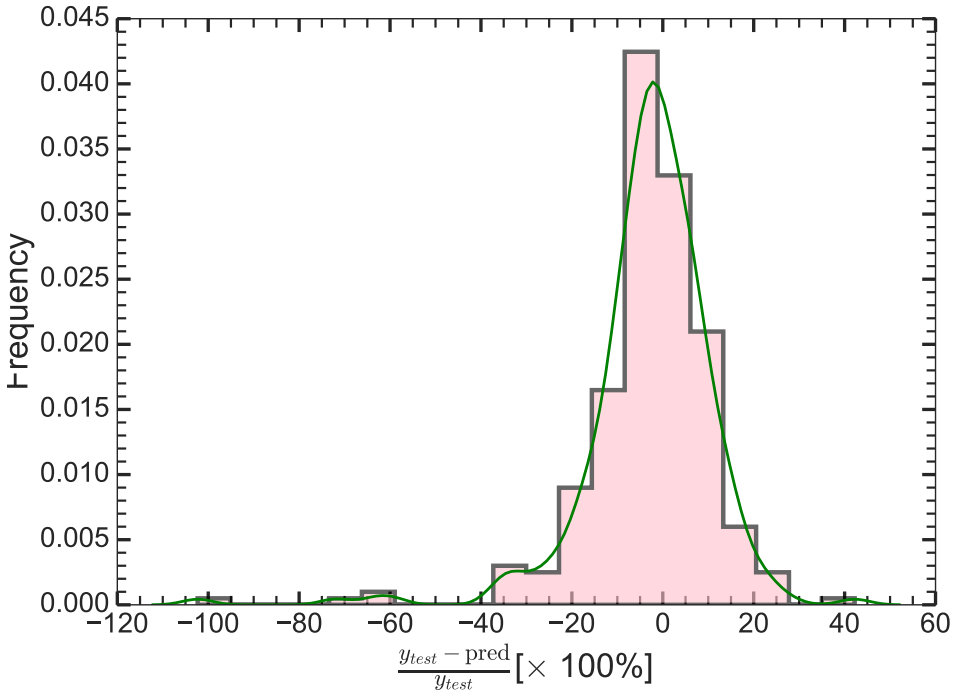

Figure 12: The fractional relative difference of the predicted price to the true sale price, overlain with a non-normal fit in green. The worst cases have a relative difference of 80-100% from the true sale price. The model has an $R^2 \sim 0.90$.

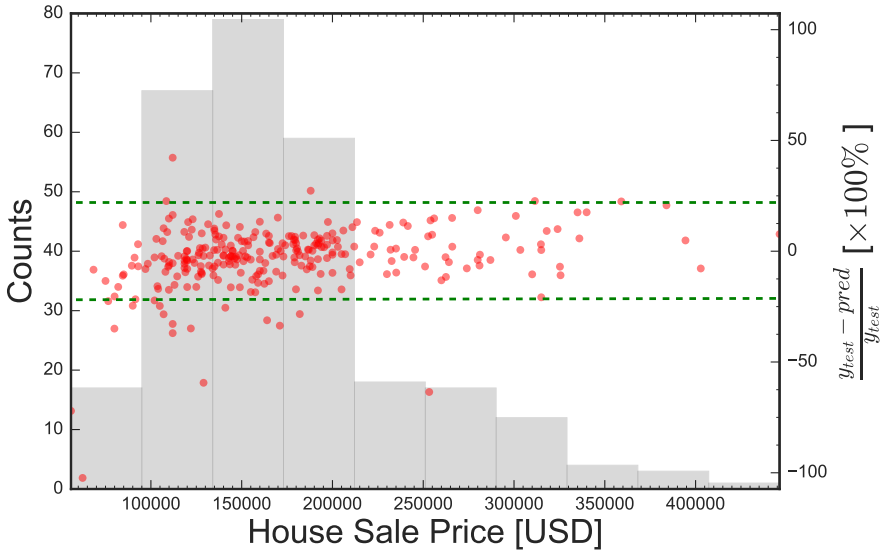

Figure 13: *Left axis:* The distribution of the sale prices in linear scale. *Right axis* The relative difference between the predicted and the true sale price. The green dashed lines denote the location of the $\pm 2\sigma$ range. The examples outside the green lines, which show the worst fit, have been over predicted by the model.

as three times. The fit has **R^2 = 0.98, and an RMSE = 7.23$\times 10^7$ USD**.

I am also curious as to how the model is doing in predicting the outliers. Figure 16 depicts the model predictions on the examples that were removed because of the ill-behaved features. Clearly, even on outliers, the model still performs impressively well, based on the RMSE and the R^2 score.

In comparison with the Boston housing project, which has 506 entries, the optimal decision tree regressor yielded an $R^2 = 0.80$ RMSE = 6.78×10^9 USD. The opt model utilized the three features, representing the number of rooms in a house, the student-teacher ratios and the percentage of the population with below a certain income level.

5.4 On Training Times

Table 2 compares the training times of the two models. Training the entire feautres set took almost twice as long. Here, training time pertains to the entire duration GridSearchCV() took to find the optimal parameters for the GB regressor. We see, however, relatively fast prediction times for test sample sizes of > 200. Although the model took some time to find the optimal regression parameters, its highly

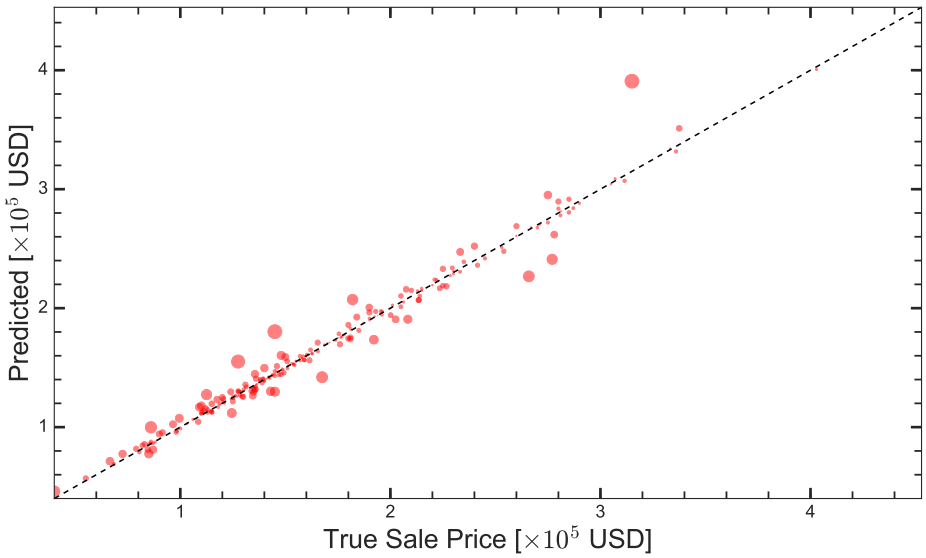

Figure 14: Predictions of the model (using the entire feature set) versus the true price for 247 test examples. The 223 examples with repeat outliers in any of their 20 continuous features were removed. Optimal parameters are ($\max_{depth}$, $\min_{samples\_leaf}$, $\min_{samples\_split}$, Ł, $n_{estimators}$) = (4, 3, 2, 0.01, 1000), with $R^2 = 0.97$, and RMSE is 4× that in Fig. 11.

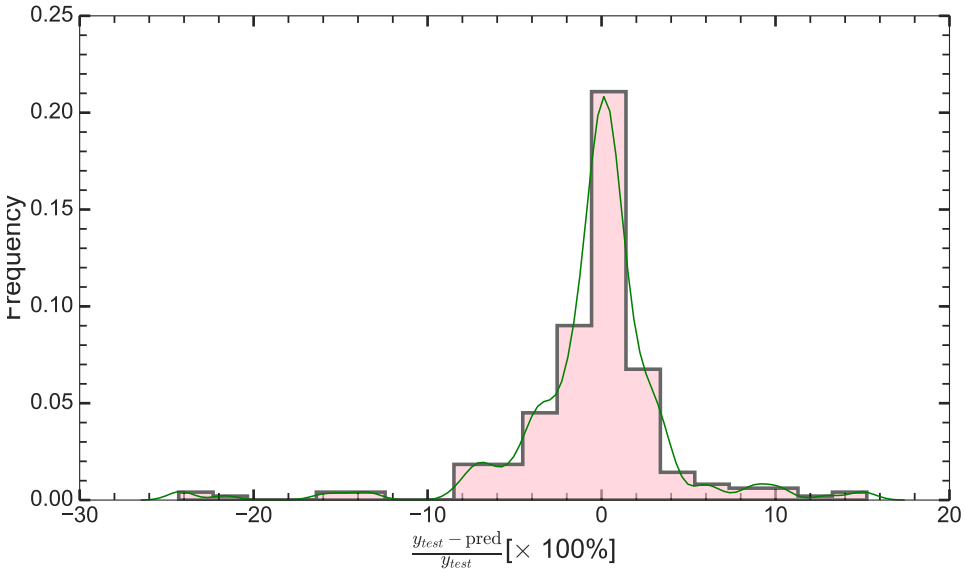

Figure 15: Percent difference relative to the true price for Figure 14, showing better results than Figure 11.

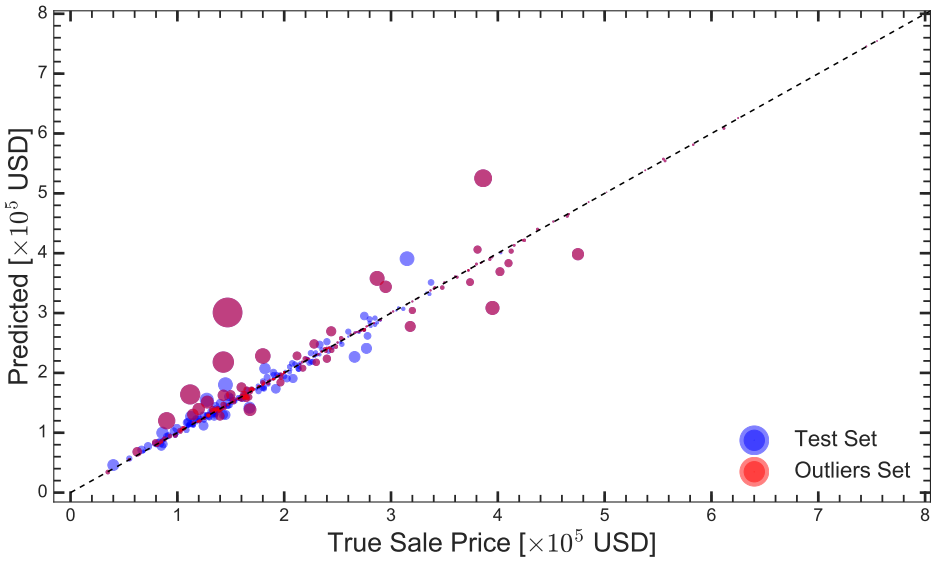

Figure 16: The predictions versus the true price of the test set overplotted with that for the outliers. Both fits have $R^2 = 0.98$. The RMSE = 3.86×10^8 USD for the test+outliers set, and RMSE = 2.21×10^8 USD.

Table 2: Comparison of training and prediction times for the two GB regressors having different sets (and expanse) of features.

Model	Optimal Parameters	Num. of Features	n_{train}	t_{train}	n_{test}	t_{pred}
-	(max_{depth}, Ł, $n_{estimators}$)	-	-	[minutes]		[seconds]
GBT	(2, 0.05, 1000)	123	1110	39.1	277	0.0029
GBT	**(3, 0.1, 1000)**	**304**	**990**	**72.5**	**247**	**0.0058**

accurate predictions makes the "waiting" all worth my while.

6 Summary, Conclusions and Reflection

For this project, I have shown the results of the exploration and regression analysis on the Ames, IA housing dataset for homes sold between 2006-2010. The data is an updated version of the housing price regression problem. With over 2900 examples and 80 home features, including the >40 categorical variables, more advanced regression techniques are necessary. To solve the problem, gradient boosted decision tree regressors were employed. The procedure which led to the optimal model required the removal of a few repeat outlier entries, and the utilization of the entire feature set. The optimal GB regressor is described by the second row of Table 2, highlighted in blue.

All the trouble that I had gone through in selecting the "right" features seemed to have been beaten by the algorithm. To me I was trying to put as much human

intuition and judgment onto this problem. This took the biggest chunk of my alloted time for this project. I could have been done a week. Obviously, there must have been other features that were important by itself, or in combination with others. Being aware of what the outliers are doing was also crucial in this analysis.

Overall, I feel "betrayed" by the algorithm (if this makes sense), because it only took the algorithm 70 minutes and it spit out an accuracy far beyond what I set for myself (i.e $R^2 = 0.9$)! Gradient boosting is incredibly a powerful ML tool, and this has definitely become a favorite. Obviously, not all 304 features are useful and one could always take out the last ten-fifty or so of them, depending on how much accuracy one is shooting for.

References

[1] D. De Cock. Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project. *J. Stat. Ed.*, 19:1–14, Nov. 2011.

[2] D. Harrison and D. Rubinfeld. Hedonic Housing Prices and the Demand for Clean Air. *Journal of Environmental Economics and Management*, 5:81–102, 1978.

[3] A. Natekin and A. Knoll. *Boosting Simplified Fuzzy Neural Networks*, pages 330–339. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[4] P. Prettenhofer. *Gradient Boosted Regression Trees in scikit-learn*, 2014. `https://www.youtube.com/watch?v=IXZKgIsZRm0`.

[5] Wikipedia. *Outlier*, 2014. `https://en.wikipedia.org/wiki/Outlier`.

[6] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1697–1704. Curran Associates, Inc., 2008.

A Appendix

Table 3: Features of the Ames Housing dataset that were selected for this analysis.

Feature	Description	Units	Selected
Continuous Features			
LotArea	Lot size	square feet	Yes
LotFrontage	Linear feet of street connected to property	feet	No
GrLivArea	Above grade (ground) living area	square feet	Yes
TotalBsmtSF	Total basement area	square feet	Yes
BsmtFinSF1	Type 1 finished	square feet	Yes
BsmtFinSF2	Type 2 finished	square feet	No
BsmtUnfSF	Unfinished basement area	square feet	
LowQualFinSF	Low quality finish	square feet	No
1stFlrSF	First Floor Area	square feet	Yes
2ndFlrSF	Second floor area	square feet	Yes
MasVnrArea	Masonry veneer area	square feet	Yes
WoodDeckSF	Wood deck area	square feet	Yes
OpenPorchSF	Open Porch area	square feet	Yes
ScreenPorch	Screen porch area	square feet	Yes
EnclosedPorch	Enclosed porch area	square feet	No
3SsnPorch	Three season porch area	square feet	No
PoolArea	Pool area	square feet	No
MiscVal	Value of miscellaneous feature	USD	No
GarageArea	Size of garage in square feet	square feet	Yes
SalePrice	Price the home was sold	USD	Yes
Discrete Features			
OverallQual	Rates the overall material and finish of the house	Numeric code	Yes
OverallCond	Rates the overall condition of the house	Integer code	Yes
BsmtFullBath	Basement full bathrooms	Integer	No
BsmtHalfBath	Basement half bathrooms	Integer	No
FullBath	Full bathrooms above grade	Integer	No
HalfBath	Half baths above grade	Integer	No
Bedroom	Bedrooms above grade [8]	Integer	Yes
Kitchen	Kitchens above grade	Integer	No
Fireplaces	Number of fireplaces	Integer	No
GarageCars	Size of garage in car capacity	Interger	Yes
MSSubClas	Identifies the type of dwelling involved in the sale	Integer code	No
GarageYrBlt	Year garage was built	Year (YYYY)	Yes
YearRemodAdd	Remodel date	Year (YYYY)	Yes
YrSold	Year Sold	Year (YYYY)	Yes
Categorical Features			
MoSold	Month Sold (MM)	Integer 1−12	No
MSZoning	Identifies the general zoning classification of the sale	Letter code	Yes
Street	Type of road access to property	Gravel or Paved	Yes
Alley	Type of alley access to property	Gravel or Paved access	No
LotShape	General shape of property	Shape code	No
LandContour	Flatness of the property	Letter code	No
Utilities	Type of utilities available	Letter code	No
LotConfig	Lot configuration	Letter code	No
LandSlope	Slope of property	Letter code	No
Neighborhood	Physical locations within Ames city limits	Letter code	Yes
Condition1	Proximity to various conditions	Letter code	Yes
Condition2	Proximity to various conditions [9]	Letter code	No
BldgType	Type of dwelling	Letter code	No
HouseStyle	Style of dwelling	Letter code	No
RoofStyle	Type of roof	Letter code	Yes
RoofMatl	Roof material	Letter code	No
Exterior1st	Exterior covering on house	Letter code	No
Exterior2nd	Exterior covering on house [10]	Letter code	No
MasVnrType	Masonry veneer type	Letter code	No
ExterQual	Evaluates the quality of the material on the exterior	Letter code	Yes
ExterCond	Evaluates the condition of the material on the exterior	Letter code	No
Foundation	Type of foundation	Letter code	Yes
BsmtQual	Evaluates the height of the basement	Letter code	Yes
BsmtCond	Evaluates the general condition of the basement	Letter code	No
BsmtExposure	Refers to walkout or garden level walls	Letter code	No
BsmtFinType1	Rating of basement finished area	Letter code	No
BsmtFinType2	Rating of basement finished area [11]	Letter code	No
Heating	Type of heating	Letter code	No

Continued on next page

[8] Does not include basement bedrooms.
[9] If more than one is present.
[10] If more than one material exists.
[11] If multiple types exist.

Table 3 – *Continued from previous page*

Feature	Description	Units	Selected
HeatingQC	Heating quality and condition	Letter code	Yes
CentralAir	Central air conditioning	Letter code	No
Electrical	Electrical system	Letter code	No
KitchenQual	Kitchen quality	Letter code	Yes
Functional	Home functionality [12]	Letter code	No
GarageType	Garage location	Letter code	No
FireplaceQu	Fireplace quality	Letter code	No
GarageFinish	Interior finish of the garage	Letter code	No
GarageQual	Garage quality	Letter code	No
GarageCond	Garage condition	Letter code	No
PavedDrive	Paved driveway	Letter code	No
PoolQC	Pool quality	Letter code	No
Fence	Fence quality	Letter code	No
MiscFeature	Misc. feature not covered in other categories	Letter code	No
SaleType	Type of sale	Letter code	No
SaleCondition	Condition of sale	Letter code	Yes

[12] Assume typical unless deductions are warranted.