CSC440 Project
Predicting Of House Price With Its Parameters

Andrii Osipa

December 2016

1 Problem Statement

House price prediction is actual problem as it is always a task to evaluate home before selling in a proper way, so it will be sold for its value and will not stay for very long time "on sale" because its value was set too high. House price is variable that depends on many other parameters in a very unclear way. Therefore we have a problem to find dependence of a price from house parameters, select ones which affect the price and drop those which does not and still maintain the interpretability of the price got in result.

2 Dataset

Dataset provided by Dean De Cock - Ames Housing Dataset. It contains list of different 79 attributes for each home. These variables explain almost every property of home including its type, age, lot size, type of street and district. Data is from real homes located in Ames, Iowa. There are number of continuous, discrete variables and ordinal and nominal categorical variables. Total amount of records in dataset is 1460 in training and 1459 in test part.

2.1 Data analysis

Given dataset contains many different features and they have different types of values. Firstly I would like to take a look at SalesPrice distribution generally in the dataset. As we see on *Figure 1* there may be found three categories of prices: low, medium and high. And limits of those categories are approximately $100k$ and $200k$. For medium category price growth is linear, while for low and high we have something like polynomial. Mostly examples belong to the medium category.

Another point that may be important is: if price distribution is different for different house types? In dataset there are 15 different types of houses, but there is problem with this features: many classes are underrepresented. Top 3 classes cover 67% representatives from the training set, and least popular class has

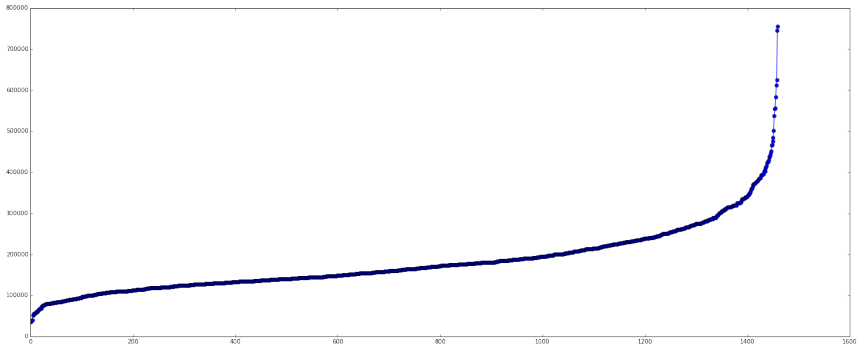

Figure 1: Sales price distribution.

only 4 representatives. For all three top classes $20, 50, 60$ we have almost same price distributions as we had generally. Important fact we can note from this plot that *medium* or normal price for different types of houses have significant difference. In other words we should not drop feature.

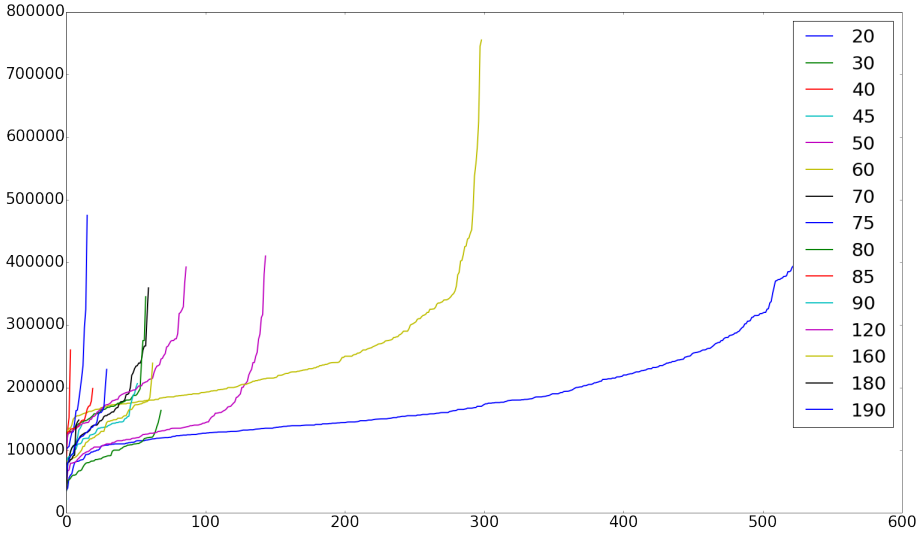

Figure 2: Sales price distribution for different types of houses.

Another simple analysis to perform is to see plots Feature vs SalesPrice for some of the features given. Next two figures 3 and 4 demonstrate features that have quite low variance or one of the feature‘s values is dominating and almost every record has exactly this value. These features are candidates for unneeded features.

Another problem with dataset is that some features do not reach all values stated in the documentation. Therefore big variety of values becomes useless

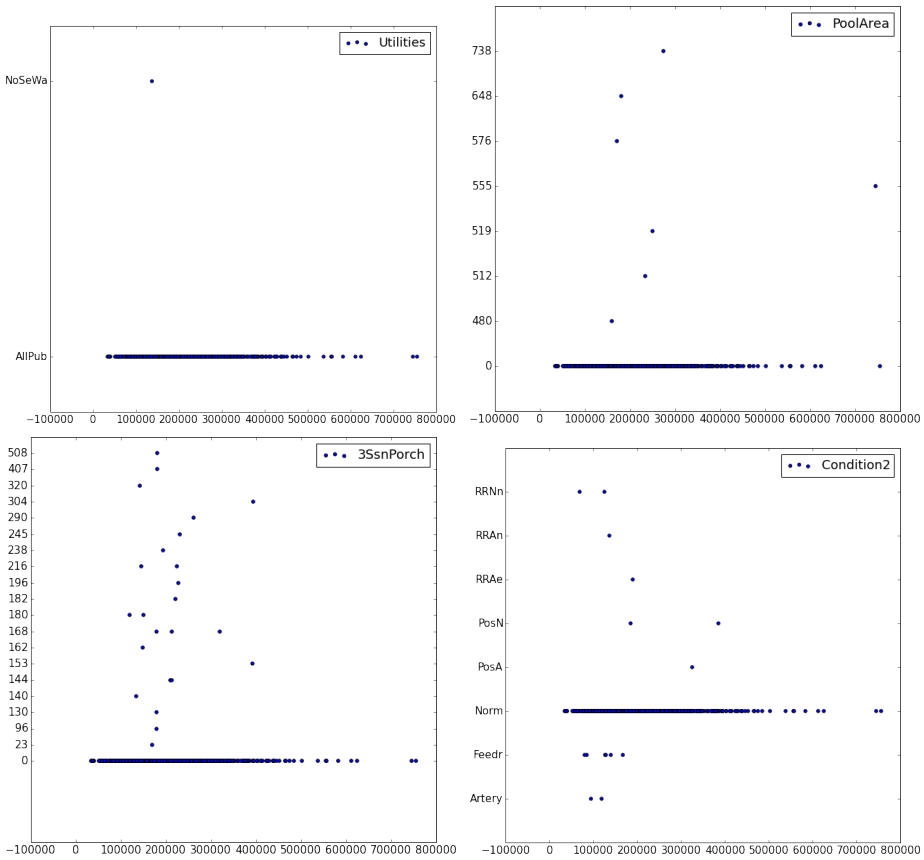

Figure 3: Features with low variance.

sometimes but still causing complexity as we have many dimensions.

3 Performance evaluation

Result will be evaluated on Root-Mean-Squared-Error (RMSE) between the logarithm of the predicted value and the logarithm of the observed sales price. (Taking logs means that errors in predicting expensive houses and cheap houses will affect the result equally.)

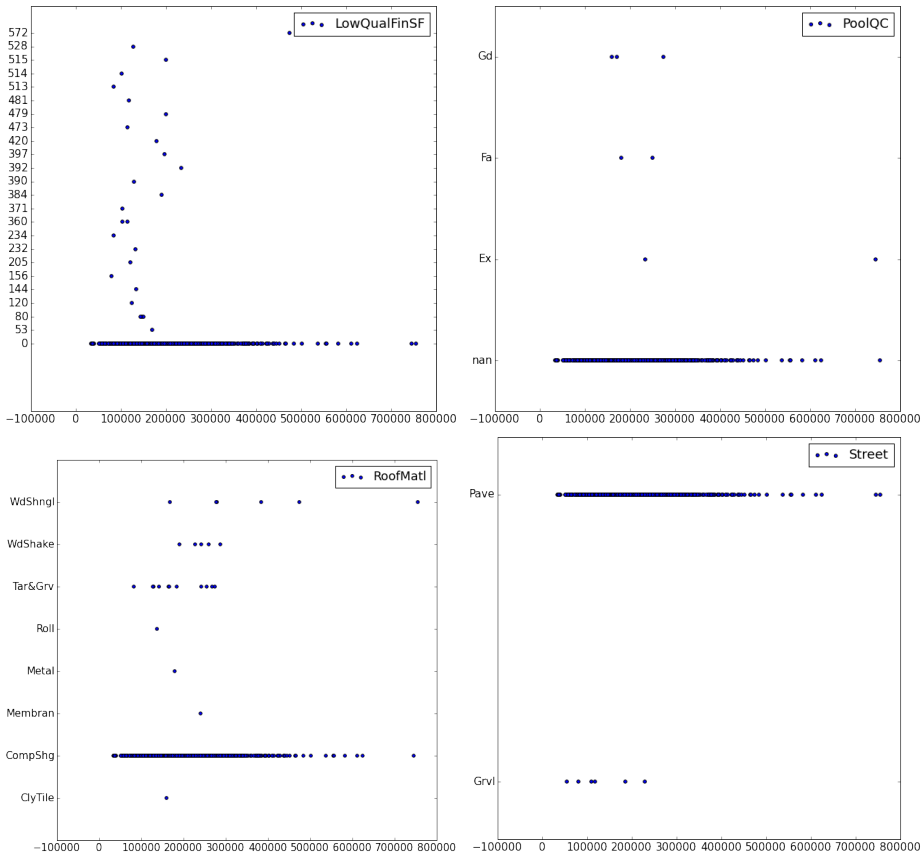

Figure 4: Features with low variance(cont'd).

4 Data Preprocessing

4.1 Test and Train Split

Data was split 80% for train and 20% for test. Split is preformed by random selection of indexes. Final testing set contains 1459 more entries, which is almost same size as train and test sets together.

4.2 Encoding

As it was mentioned before there are big amount of features and they are not only continuous and discrete but also ordinal and nominal. To make data processable by Python Sklearn everything must be numerical.

On the first stage all nominal and ordinal features were encoded just by LabelEncoder. This encoding achieves goal that all data is numerical but it

ignores nominal data. Features values were encoded to some numbers with some unwilled order, obviously. In most of the trainings this data was used.

Next stage of encoding is encoding that preserves no order in nominal features. This was done by OneHotEncoder from Sklearn. Important note that this step was performed after feature importance analysis and feature number reduction. Reason for this is that OneHot encoding increases dimensionality by replacing each feature value by binary vector of length equal to number of values of the feature. As mentioned before, this encoding was performed on reduced feature set, there were 69 of them After encoding each data entry size increased from 69 to 241.

4.3 Feature Importance

As we have seen before, we already had candidates for "bad" features. But when we look at distribution of one feature and price, it may not be very effective, as we still ignore all other features.

I used two methods for finding least important features. One of them is low variance analysis and other is selection of k best features by regression analysis. Regression analysis was performed on the not onehot encoded data. After given scores for each feature I dropped worst 9 features as they had big enough difference even with 10^{th} lowest feature. Some of these dropped features were mentioned before in Data part, so they already were marked as candidates for bad features: Utilities, Condition2, BsmtFinSF2, LowQualFinSF, BsmtHalfBath, FireplaceQu, MiscVal. And other feature that was dropped is YearSold. Logically it makes sense to drop date of sale, but MonthSold got big score in importance and it probably makes sense, that part of a year has affect on a homes price.

Variance analysis was performed on the same data, but results gained from it were not worth dropping, as they seem to be important in terms of natural logic.

5 Prediction with Random Forest

First model to test performance was Random Forest. One of the test results gained with this model had RMSE equal 0.22 on test set and 0.25 on final test. Parameter fitting was done by small grid search. Therefore this is just sample score, achieved with this model, and it may be better if model parameters fitted better.

Big parameter fitting was not performed as it takes time and main goal for this step was to get sample scores for different models.

Now lets take a look at prediction results with ground truth to understand what does exactly error 0.22 looks like. On the *Figure 5* we see that really difference with predicted price and real one is quite big. Especially there are problems with very expensive houses.

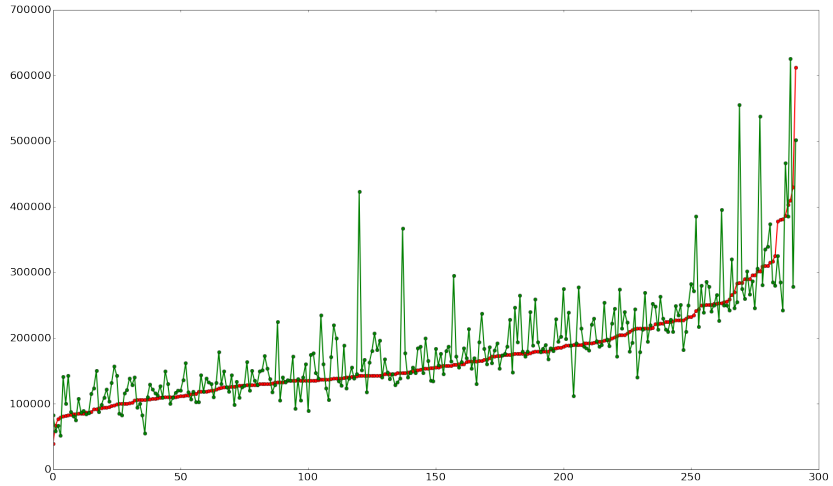

Figure 5: Prediction from RF and ground truth.

Disadvantages of RF: price is a continuous feature and random forest have some fitted values in trees' leaves and therefore prediction is limited buy those values. Therefore this model is not the best one to predict price.

6 Prediction with Elastic Net Regression

Second model to train is Elastic network regression. Regression seems to be reasonable model to predict price as price is a continuous variable. Now as we have numerical data regression can be applied to this data. Model was trained for same reason as RF, and so parameter fitting was done in the same way: small grid search. Result achieved here was 0.1639 on test. Therefore we see great increase of performance with previous model.

Note, that this model was trained on whole data without onehot encoding.

Next step was testing same model on data with best features only. This approach gave very little positive difference with previous score.

7 Prediction with XGBoost Regression

Last model I used on this data was gradient boost regression. I decided to try this regression model as it generally gives better performance compared to ENet Regression. This model gave best achieved score. In training process I also used grid search for global estimator parameters and also cross-validation with 5 folds. Other fact that can be noticed is that we may estimate base score for regression different from 0. From training set I selected base score 30000 and this improved performance for models with other parameters unchanged.

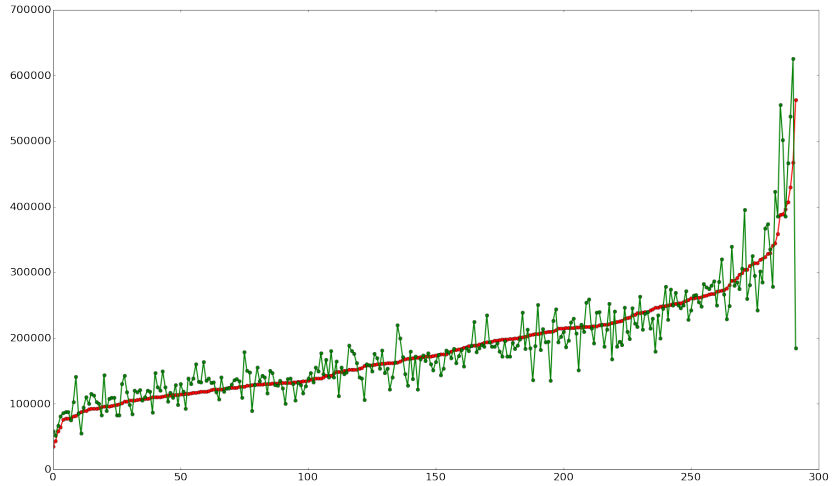

Figure 6: Prediction from Enet and ground truth.

To look at results gained we will also look at RMSE on training set to see if the model is potentially overfitted or not. Achieved results were the following:

- In this case model was trained without onehot encoding but with best features only. Model fitting took 3.3 hours. RMSE achieved on test set was 0.13. RMSE achieved on train set is 0.043.
- This model was trained on best features with onehot encoding, which removes order from features, where there is no order originally, but has dimensionality 241. Therefore, this training data is supposed to be better. RMSE on given test: 0.126725776313. RMSE on train: 0.0230632976871. Training took 7.7 hours. Parameters were searched from same grid as in previous case.

For the second model we see that it is probably overfitted as RMSE on train is very low. The *Figure 7* shows how good first regression works. We can see that biggest errors occur for the homes with high price. Also there is one example of outlier, but other errors are mostly lower then in previous ENet model. Other difference to notice: this model gives less underestimations then previous one. Each regression model trained with XGBoost has this one outlier and I still do not have an explanation why this happens.

8 Final Results

Best score achieved on final test set is 0.13323, which was achieved with XGBoost model trained on best features without onehot encoding and with cross-validation. On the general leaderboard on Kaggle this score had 1195

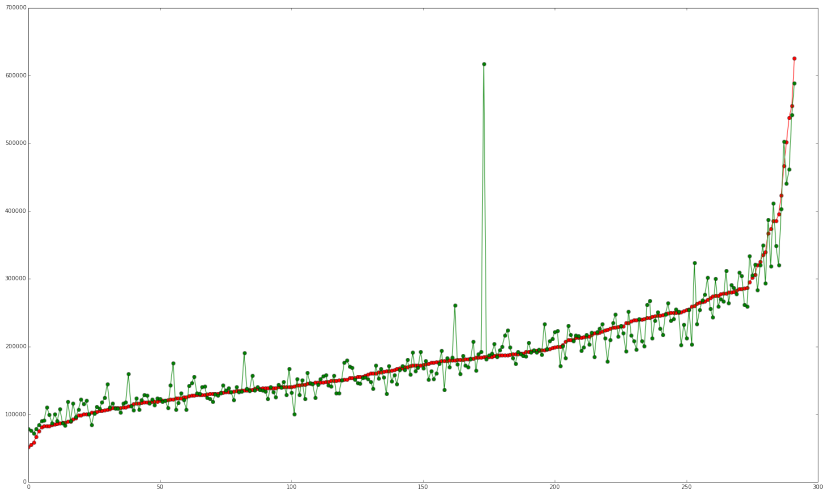

Figure 7: Prediction from XGBoost Regression and ground truth.

position out of 2596 as for Dec 15, 2016. This is far away from top 3 entries, but from 7^{th} position scores start from 0.11. Therefore my model performs not so much worse as the 7th one.

Conclusions: it is very hard to predict price with one model. There is no easy dependence from variables Error of predicted price is quite big if it is taken in terms of absolute error, but not logarithm. This makes the model not very usable in real life as such difference is significant. Regression is good to see what parameters are really affecting price somehow and which ones can be dropped. Also I am wondered that currently performance on a data with onehot encoding is worse. This encoding seems to be proper for this task as it preserves properties of an initial features – no order.

9 Future work

There are few possibilities what to do. One of the problems mentioned before is one outlier with average price. Way to improve regression s to dig into this example to understand reasons.

Other good thing to do is use few different models to calculate final price. As we have seen model still performs quite bad with high prices and oscillates a lot.

And there is more ways to improve current regression: other algorithms for picking hyperparameters of an estimators; define other scoring function for regression training to better control overfitting; perform more dimensionality reduction by getting importance from already got best regression model and use this new training data for new regression. Problem with last approach that there is no natural threshold in importance of the features got from regression

as it was in the given before analysis. Therefore the way to check what is least important is to try all sets of lowest importance scored features, which also creates complex computations. Currently it takes pretty a long time to pick parameters from even not big grid.

10 References

This problem was observed in different works with different datasets. Previous works contained less variables that explain homes and used different regression techniques to find the regression.

All those mentioned works worked with quite different datasets and had much less features. Important issue is that with small amount of parameters and therefore approaches were quite different.

1. **Kaggle. House Prices: Advanced Regression Techniques**
   `https://www.kaggle.com/c/house-prices-advanced-regression-techniques`
2. **Multivariate Regression Modeling for Home Value Estimates with Evaluation using Maximum Information Coefficient**
   `http://www.acisinternational.org/Springer/SamplePaper.pdf`
3. **Machine Learning for a London Housing Price Prediction Mobile Application**
   `http://www.doc.ic.ac.uk/~mpd37/theses/2015_beng_aaron-ng.pdf`
4. **Predicting Home Prices from Various Factors**
   `http://sites.stat.psu.edu/~sesa/stat460/Project/greg.pdf`
5. **House Price Prediction**
   `http://terpconnect.umd.edu/~lzhong/INST737/milestone2_presentation.pdf`