

Project Report

Zillow Prize Challenge

Kiranmayi Kasarapu

Department of Computer Science, Stony Brook University

kkasarapu@cs.stonybrook.edu

Challenge

Zillow predicts the prices of real estate houses. The difference in the actual price, the house is sold for and the estimated price by Zillow is calculated in the form of logerror. My challenge here is to predict the logerror for these real estate houses.

Real estate prices vary extensively and there are various factors that influence the price. In most of the data science prediction problems, sometimes the factors affecting may be interdependent and sometimes they may vary with time which is actually the case with the estimation of prices for real estate, therefore since logerror is dependent on estimated prices, it is also indirectly depends on the time.

In this project we will see the important factors that are affecting the logerror, various models that are built varying from simple to advanced models, and see how the mean square error and r^2 varies with each models.

Factors affecting the logerror

At the beginning of the project, I was unable to decide which factors were important, and which were not by just looking at the data. I decided to find a correlation between most of the properties that are provided. I have considered only numeric properties to keep it simple, used only those rows for which data was available and get an idea about the features. I have generated a heatmap for all the correlations between the features.

This map gave me some idea about which features are interdependent and, which features are independent and which features are negatively interdependent. For example, look at the assessment year column/row, It is almost white, which mean a zero correlation with the given properties.

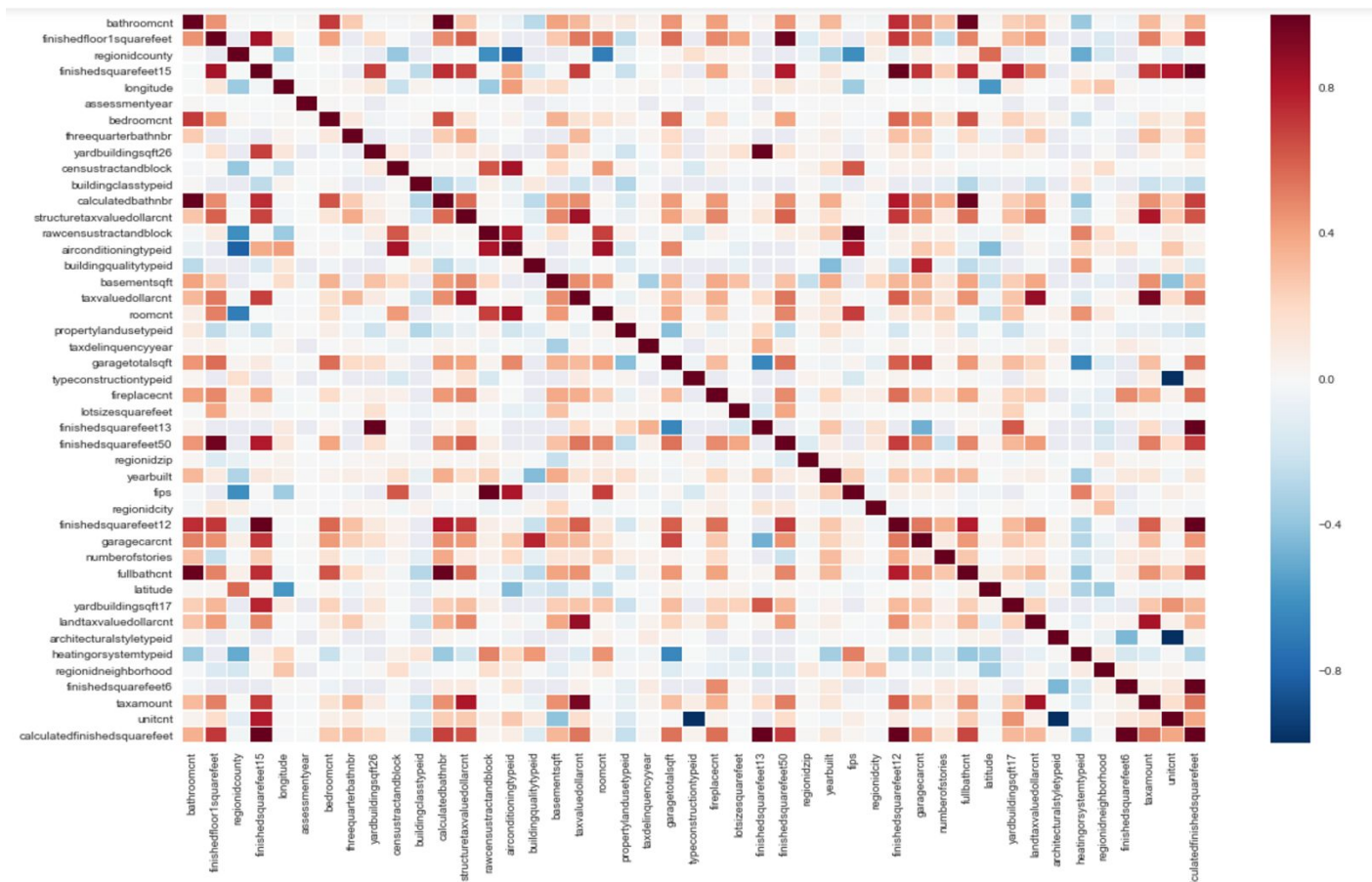
Similarly from the map we can see the most positive correlation is between (finishedsquarefeet13, finishedfloor1squarefeet), (fullbathcnt and bathroomcnt), (calculatedfinishedsquarefeet, finishedsquarefeet15) and many other pairs.

The most negative correlation pairs appears to be (unitcnt, typecontructiontypeid), (unitcnt, architecturalstyletypeid) and few more.

Based on the observations from this map I took few interesting features

interestingFeatures = ['yearbuilt', 'structuretaxvaluedollarcnt', 'taxvaluedollarcnt', 'bedroomcnt', 'taxamount', 'bathroomcnt', 'propertylandusetypeid', 'calculatedfinishedsquarefeet', 'unitcnt']

And calculated the correlation among them, which is displayed as the final heatmap in the code.



Other Informative Plots

1. Linechart - I plotted a linechart show the correlation between logerror and couple of features
2. Bar graph - Plotted a bar graph showing the distribution of houses with number of rooms each house has. From the graph, we can see that '0' number of rooms are there in most of the houses, which is clearly wrong. It is possible that data is missing for such entries. I have plotted the same graph again, ignoring those rows.
3. Box chart - Based on the Linechart from (1) we can see that finishedsquarefeet12 and heatingorsystemtypeid has the most positive and negative correlation with logerror. So i decided to plot a graph between these two features. The line inside each box, tells us the mean value of finishedsquarefeet12 for each of the heatingorsystemtypeid.
4. KDE plot - Plotted distribution of logerror using kdeplot. The graph shows that the distribution is almost a normal distribution around value "0"
5. Showing distribution of number of houses built in various years.

Models

Linear regression Model

Linear regression is a statistical model, that examines how the dependent variable changes as a linear function of independent variables. In this model I kept the logerror as the dependent variable and use some properties as independent variables.

We try to fit the logerror into a linear function, usually a straight line and see how it varies as a function of other properties, basically we try to find the best fit line for logerror.

To keep it simple for the initial analysis, I started with raw data, filled the missing values with the mean of the properties for each column and used it for analysis

Raw Data

I selected couple of properties which I think is important for predicting the logerror and built a linear model using these properties only. Initially I did not use the transaction date in my properties list to predict the error. This resulted in predicting the same value for all the time points given in the submission file. Later I included transactiondate as one of the property for this model.

I used all the training data for which logerror was given and tried to fit into a model.

I used the same data to predict the logerror values and calculated the mean square error between the actual and predicted values.

Sample Predictions: [-0.01672309 0.03697879 0.0278185 0.01180325 0.01310295]

Actual Error : [-0.0243 0.0862 0.6637 0.3457 0.0431]

Mean square error = 0.0245899522044

$R^2 = 0.00421466507674$

Normalized data

For the next step, I decided to normalize the data, by subtracting each column with mean, and dividing by standard deviation and fit it into the linear model. There was a little improvement in the Mean square error.

Sample Predictions: [0.00556459 0.02006111 0.01178714 0.01943322 0.02870431]

Actual Error: [-0.0010 0.0139 0.0149 0.0040 -0.0080]

Mean square error = 0.02083390147

$R^2 = 0.0098450535587$

There wasn't much significant improvements, but we got a better R^2 value

K-Nearest Neighbours

The next model, I used is KNN classifier. The idea behind this model to predict unknown values by matching with the most closet or similar known values. In this model, we decide on how much closest neighbours, it needs to match before making a prediction. The number of neighbours can be any number less the total number of rows in the data set. In general, matching with few neighbours makes the model work better, as if the neighbours are less similar to the data, the prediction will be worse. A simple algorithm of how KNN works

```
K - NumNearestNeighbours
Initialize K
Calculate distance between test data and each training data
Sort the distance values
Get K top most values
Find the majority value in the K top most values
```

Raw Data

Here again, I used the raw data with most of the properties, ignoring the non-numeric values. Some observations below

Predictions: [0.0117108 0.02826907 0.01182333 0.015908 0.0075536]
Actual Errors: [-0.0243 0.0862 0.6637 0.3457 0.0431]
Mean square error = 0.024674633953
 $R^2 = 0.000785425249128$

we can clearly see we R^2 is very less than that of linear model. For this data set, it looks like Linear model has worked out better

Normalized Data

Data is normalized and passed into the KNN classifier model:

Observation below:

Predictions: [-0.04091371 0.03431114 0.01842257 0.00483971 0.00655771]
Actual Values: [-0.0243 0.0862 0.6637 0.3457 0.0431]
Mean square error = 0.0245699924054
 $R^2 = 0.00502294949062$

We can see that KNN has done better with normalized data. There is significant increase in the R^2 value as compared to raw data. But again, it is still less than that of linear models R^2 which is 0.0098450535587.

Random Forest Regressor

In this model, data is split into two at every step of decision (usually depth) based on a certain criteria. The predicted value at any node is the average of all values at that node. For regression we use residual sum of the squares as the splitting criteria.

For this model, I have used normalized data. Split the data into train and test sets and fitted into the model.

Observations below:

Predictions: [0.076326 0.054042 0.004077 0.0127475 0.0475405]

Actual values: [-0.001 0.0139 0.0149 0.004 -0.008]

Mean square error = 0.0238195605211

$R^2 = 0.0151608766201$

We can see that R^2 of Random Forest model is better than any of the above models.

I have uploaded the prediction data to the kaggle site for each of the above models. Surprisingly linear model has the lowest mean square value compared to other models.

The scores are listed below

Linear Model Raw Data 0.0651983

Linear Model Normalized Data 0.0651259

KNN Raw Data 0.0730430

KNN Normalized Data 0.0732570

Random Forest Normalized Data 0.0737395

Therefore though R^2 value was better for Random Forest regressor, Linear model has done well in predicting the log errors.

My favourite models in this challenge is Linear model itself, but improving the RF models might yield better results in my opinion.