Zillow Prize

Mike Amodeo, Tom Seddon, Robert Foster, Felipe Campos

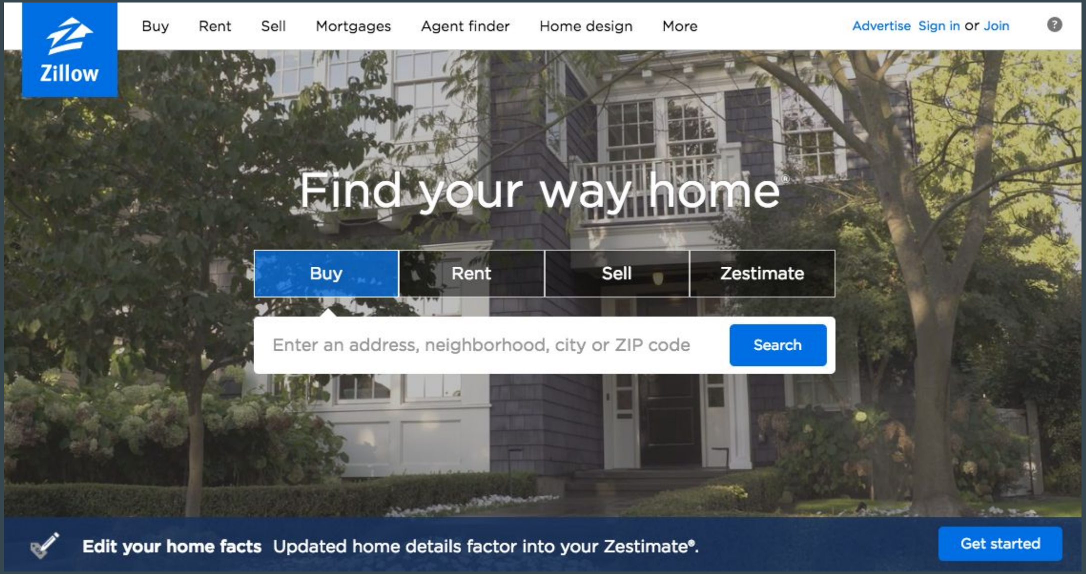
W207 Introduction to Machine Learning

Final Project

August 21, 2017

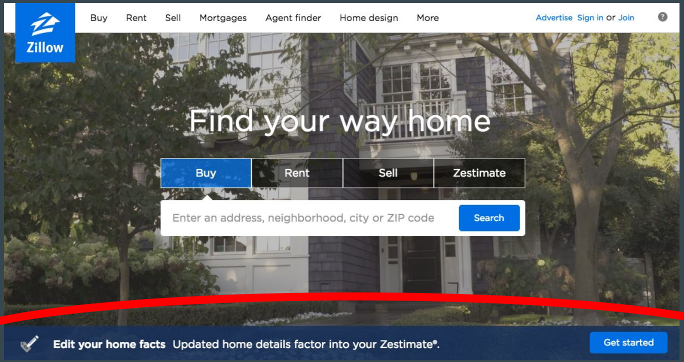
Project Description

Competition is run by Zillow, a large US real estate site

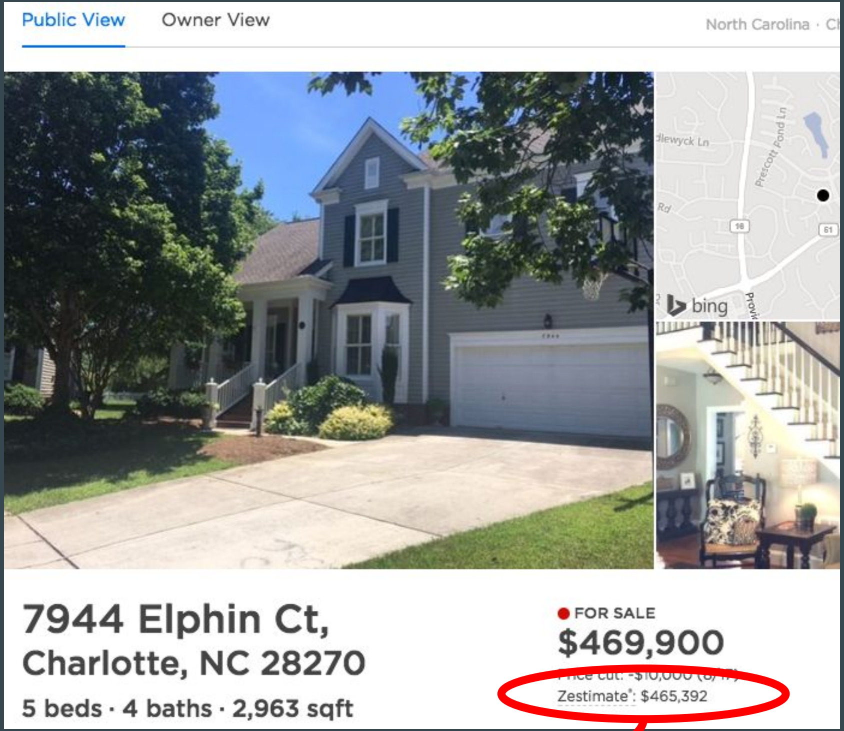


Project Description

Zillow has a proprietary 'Zestimate' model to predict what price a house will sell for, based on 3rd party and owner-reported data



Edit your home facts Updated home details factor into your Zestimate®.



7944 Elphin Ct,
Charlotte, NC 28270
5 beds · 4 baths · 2,963 sqft

● FOR SALE
$469,900
rice cut: -$10,000
Zestimate®: $465,392

Project Description

Competition goal

- Predict how far Zillow's Zestimate is from the actual sold price
- We do not have the actual Zestimate and sales prices, just the log error
- Evaluation is on Mean Absolute Error (MAE)
- Training data is for properties in 3 southern California counties
- Actual transaction data labelled for the months of October, November, and December 2016 (with some held back to Kaggle to base evaluation on)
- Full competition involves also predicting October, November, and December 2017

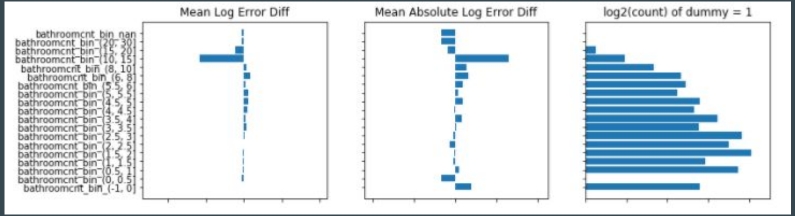
Current Leader: 0.0640172 MAE

There are ~60 fields, so we split the EDA by type of data

Building Data

e.g . # bedrooms, total sq feet

- found some patterns that looked promising



Amenities and Land Use Data

e.g . pool, zoning coding

- Interesting inconsistencies between self-entered and 3rd party fields
- Very high number of land categories

Financial Data

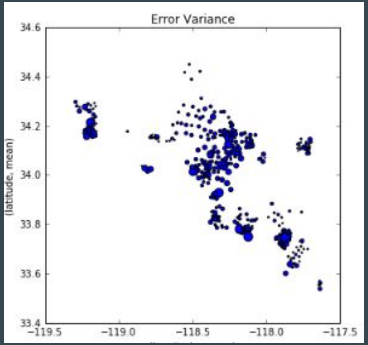
e.g. tax value, delinquency status

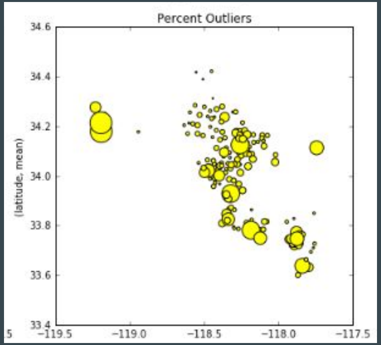
- mostly unusable since full test set does not contain the same information

Location Data

e.g. lat/long, census tract

- possible neighborhood effects on errors





EDA conclusions

Many fields, but a lot were sparsely populated. Mix of continuous data, discrete data, and numerical codes

- Because we are predicting Error in Zillow's models, the absence of data could actually be an important feature

Some of the features only seemed to matter at the extremes

- So need to try both continuous and binned features

A LOT of things that are very useful for predicting actual sales price may not be very helpful in predicting errors in Zillow's model -- so what else could be?

What else might make a house price hard to predict?

A certain amount of variation is going to depend on who the buyer is, how many other people are looking at that time, and other very hard to assess factors

But in talking to realtors, they also pointed out that buyers, sellers and appraisers place a lot of emphasis on comparable sales in establishing what a reasonable price is

- Relies on finding other houses nearby and seeing what they sold for

So we hypothesized that it would be harder for Zillow to have an accurate prediction for houses that are very different from their neighbors

- Created our own features for neighborhood/census tract effects as well as properties that are outliers in their neighborhoods/census tracts

Initial Approach

Needed to predict a continuous value for the logerror

Started with linear approaches

- Linear Regression
- Ridge Regression
- Lasso Regression

Expanded to tools that could cope with complex decision boundaries

- Decision Tree Regression
- Gradient Boost Regressor

Moved to tools that could better cope with overfitting issues

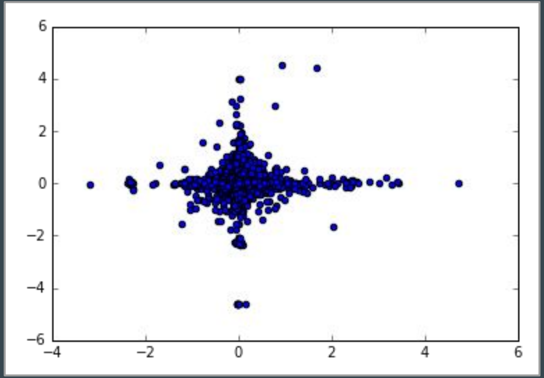
- Random Forest
- XGBoost Regression

Initial Approach -- results

Results were mostly **worse than predicting the mean value** from training data for all records !

Did we just not have our tools set up right...?

Issue: We could not reliably predict outliers:



The more we tried to fit the outliers, the worse the performance on the bulk of the data.

The more we tried to fit the bulk of the data, the worse the performance on outliers.

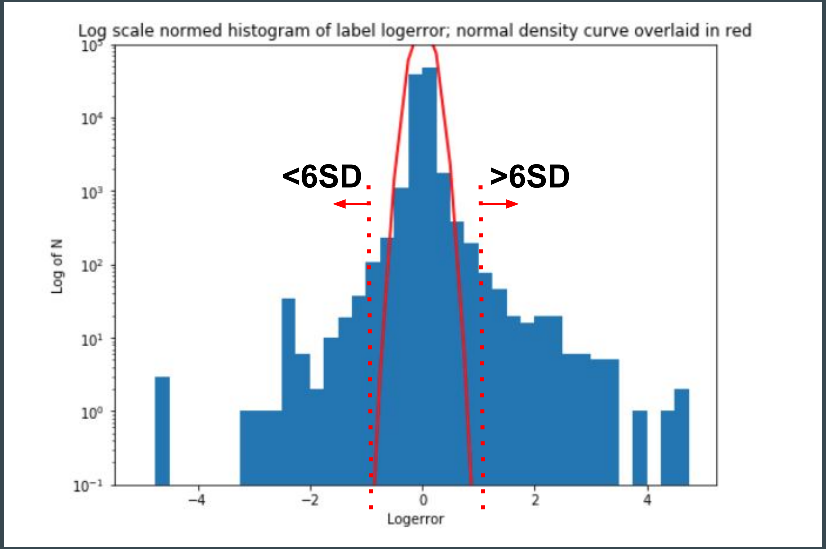
Problem: How to handle outliers?

Distribution of logerrors has very heavy, and imbalanced, 'tails'

- 0.270% above 6SD
- 0.140% below 6SD
- 0.003% expected for a normal distribution

So we decided to try to classify the extreme high and low observations

- Experimented with various thresholds for high and low, to tradeoff number of observations with extremity of difference



Classifier approach

Classes are not balanced:

- Total training set: 90275
- Positive outliers (> 0.4): 1164 (1.3%)
- Negative outliers (< -0.4): 680 (0.7%)
- In-between (-0.4& 0.4): 88431 (98%)

Classifiers get to 98% accuracy by just disregarding all the outliers!

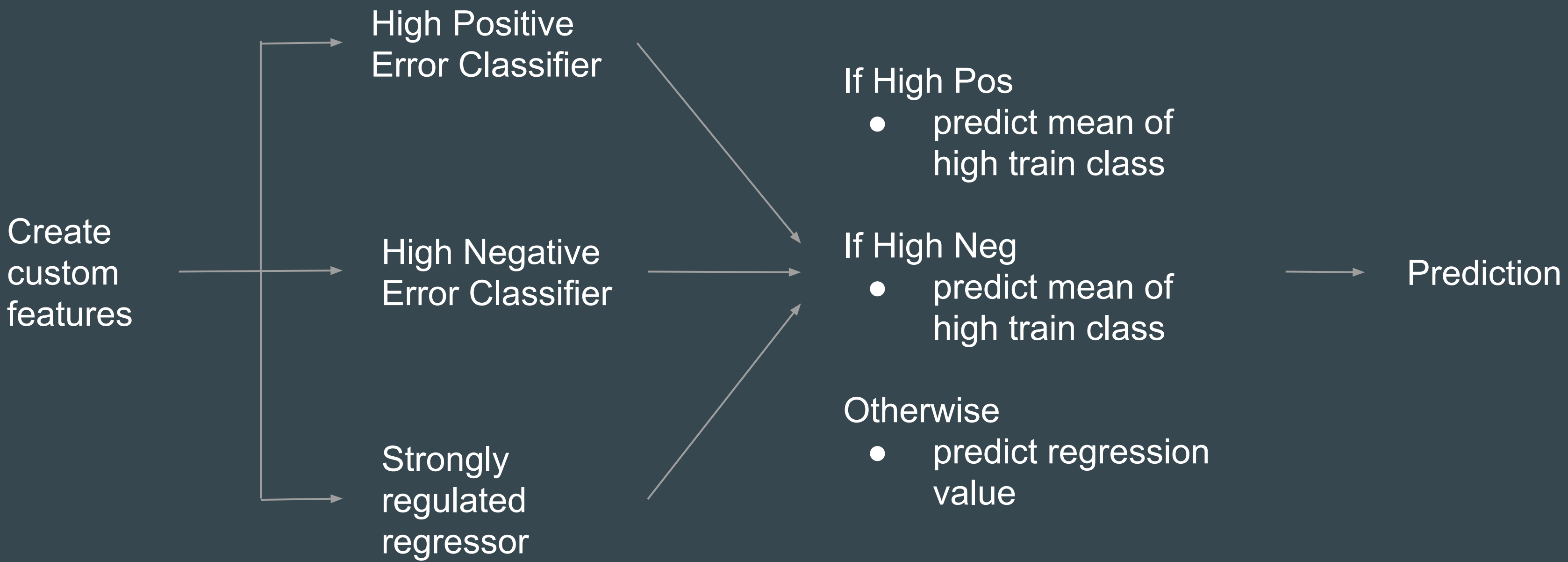
Had to find ways to get the classifier to pay more attention to the low incidence class (the outliers)

- Over-sampling / Under-sampling
- Altering probability threshold for prediction
- Weighting parameters

Solutions tried:

- 2-component PCA and K-Means Clustering
- K-nearest neighbors
- PCA to 2 components & then KNN
- Feature analysis / manually classify
  - Based on features more likely in the outlier groups
- NA analysis
  - Maybe NAs = higher chance for outlier
- Super-feature analysis
  - Maybe location = higher chance for outlier

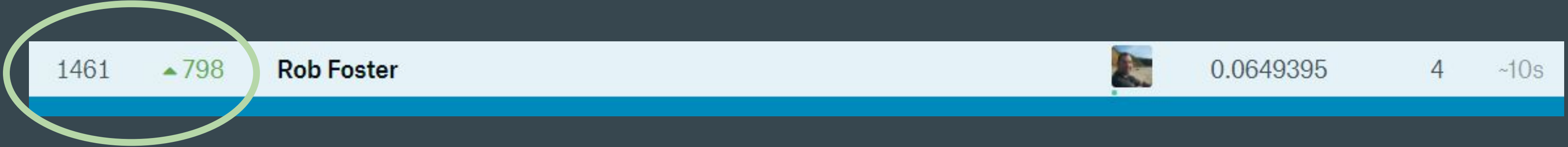
Final Ensemble Approach


Create custom features
High Positive Error Classifier
High Negative Error Classifier
Strongly regulated regressor
If High Pos
● predict mean of high train class
If High Neg
● predict mean of high train class
Otherwise
● predict regression value
Prediction

Results - this is a challenge of thousandths...

Simple linear regression baseline…...	0.0651549
Final Ensemble………………………...	0.0649395
Improvement…………………………...	0.0002154

#1461 on leaderboard



1461 ▲798 Rob Foster 0.0649395 4 ~10s

Current Kaggle leader (8/21 6pm) …. 0.0640172

Summary -- what we'd try next

- More time fine tuning hyperparameters on the most promising approaches
- Engineer more features that might help better identify high/low outliers
- Explore more dimensionality reduction to reduce overfitting to irrelevant patterns
- Explore other methods to help with the strong class imbalance in the classifiers and improve their accuracy
- More fine tuned approach to prediction of value for those classified as high/low