Machine Learning Engineer Nanodegree
Capstone Proposal
Shamit Patel
May 27, 2017

Domain Background

One of the most intriguing applications of machine learning is home valuation, because a home is often the most significant purchase an individual makes in their lifetime [1] and an accurate valuation can be the difference between making a successful sale and not. Zillow's Zestimate was developed to provide consumers with as much as information about homes and the real estate market as possible [1] so that they can make highly informed decisions about purchasing a home. A Zestimate is an estimated home value based on 7.5 million statistical and machine learning models that analyze hundreds of features of every property [1]. Zillow has cemented itself as one of the most reliable sources for real estate information in the U.S. because it has constantly improved upon its margin of error in valuating homes [1].

Problem Statement

This project is based on Kaggle's Zillow Prize: Zillow's Home Value Prediction competition, whose initial goal is to improve the Zestimate residual error, and ultimately to build a home valuation algorithm from the ground up [1]. In this project, I will focus on the initial goal of predicting the log-error between Zillow's Zestimate and the actual sale price of a home, given all the features of a home. The log-error is defined as

$$logerror = \log(Zestimate) - \log(SalePrice) \text{ [2].}$$

Predicting this error is important because it allows us to precisely gauge how accurate our home valuation predictions (Zestimates) are.

Datasets and Inputs

As described in [2], the data consists of two files:

- properties_2016.csv – list of all properties with their features for 2016
- train_2016.csv – training set with transactions from 1/1/2016-12/31/2016

This data was obtained from [2] and the features from the properties file are:

```
parcelid,airconditioningtypeid,architecturalstyletypeid,basementsqft,bathroomcnt,bedroomcnt,build
ingclasstypeid,buildingqualitytypeid,calculatedbathnbr,decktypeid,finishedfloor1squarefeet,calcul
atedfinishedsquarefeet,finishedsquarefeet12,finishedsquarefeet13,finishedsquarefeet15,finishedsqu
arefeet50,finishedsquarefeet6,fips,fireplacecnt,fullbathcnt,garagecarcnt,garagetotalsqft,hashottu
borspa,heatingorsystemtypeid,latitude,longitude,lotsizesquarefeet,poolcnt,poolsizesum,pooltypeid1
0,pooltypeid2,pooltypeid7,propertycountylandusecode,propertylandusetypeid,propertyzoningdesc,rawc
ensustractandblock,regionidcity,regionidcounty,regionidneighborhood,regionidzip,roomcnt,storytype
id,threequarterbathnbr,typeconstructiontypeid,unitcnt,yardbuildingsqft17,yardbuildingsqft26,yearb
uilt,numberofstories,fireplaceflag,structuretaxvaluedollarcnt,taxvaluedollarcnt,assessmentyear,la
ndtaxvaluedollarcnt,taxamount,taxdelinquencyflag,taxdelinquencyyear,censustractandblock
```

A sample data instance from the properties file looks like:

```
10754147,,,,0.0,0.0,,,,,,,,,,,,06037,,,,,,,34144442,-118654084,85768.0,,,,,,010D,269,,
060378002.0410,37688,3101,,96337,0.0,,,,,,,,,,,9.0,2015,9.0,,,,
```

The training file looks like:

```
parcelid logerror transactiondate
11016594    0.0276        1/1/2016
…and 90,810 more instances through 12/31/2016
```

Clearly, not all the properties of a home are actually described, as can be seen from the sample data instance from the properties file above. So, I will retain only features whose values are actually described *in every data instance*. This will ensure that our learning model performs optimally by having maximum possible information. After this feature cleanup is done, I will convert the `propertycountylandusecode` feature to a numerical value because as given, it is an alphanumeric string like `010D`. This conversion will be done by converting each unique land use code to a unique number in the range [1, # of unique land use codes]. Then, I will normalize all the feature values using the following formula: $x_{new} = \frac{x_{old} - \bar{x}}{\sigma_x}$, where x_{old} is the existing feature value, $\bar{x}$ is the mean feature value across all data instances, and σ_x is the standard deviation of the feature values across all data instances. This will also ensure that our learning algorithm performs properly by constraining all the values within a certain range. Finally, I will use these cleaned up features to train a supervised learning algorithm like linear regression, which will be described next.

Solution Statement

The solution will be in the form of either a Generalized Linear Model (GLM) or a nonlinear regressor that will predict a log-error given a particular list of feature values. A Generalized Linear Model is a flexible generalization of typical linear regression that allows for response variables whose errors are not distributed normally [3]. GLMs that will be explored are linear regression, polynomial regression, ridge regression, Bayesian regression, lasso regression, Elastic Net, Least Angle Regression, Orthogonal Matching Pursuit, Stochastic Gradient Descent Regression, and Perceptron. Nonlinear regressors that will be explored are Support Vector Regression, Kernel ridge regression, Nearest Neighbors Regression, Gaussian Process Regression, Decision Tree Regression, Random Forest Regression, AdaBoost Regression, Gradient Boosting Regression, Isotonic Regression, and Multi-Layer Perceptron Regression. The model that performs best will be chosen as the solution.

Benchmark Model

The benchmark model will simply be a function that always predicts the mean of the log-errors in the training data. This makes sense because the mean is a robust population estimator.

Evaluation Metrics

The regression model and benchmark model will be evaluated using the Mean Absolute Error between the predicted log error and the actual log error, keeping in line with Kaggle's evaluation procedure [1]. The mean absolute error (MAE) is the average of the magnitudes of the differences between the actual and predicted values [4]:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| = \frac{1}{n}\sum_{i=1}^{n} |e_i| .$$

Project Design

I will initially use linear regression to develop a model of the relationship between log-error and home properties because both the feature values and output variables are continuous values. Specifically, I will use sklearn's LinearRegression module to train on the data instances with transactions from January-September 2016, and then test on the instances with transactions from

October-December 2016. If I find that linear regression does not produce reasonable results, then I will try all the other GLMs and nonlinear regressors I mentioned in the Solution Statement until I get satisfactory results. In the end, the solution will be the model that performs best according to the evaluation metric we have defined (mean absolute error).

References

1. https://www.kaggle.com/c/zillow-prize-1
2. https://www.kaggle.com/c/zillow-prize-1/data
3. https://en.wikipedia.org/wiki/Generalized_linear_model
4. https://www.kaggle.com/wiki/MeanAbsoluteError