Zillow House Price Prediction Report

- Swarupa Vaishampayan

I primarily followed following roadmap when building the model.

Data Visualization and Data exploration:

I first explored the training data by plotting scatter plots and correlation heatmap to see the relation of other variables with the target 'SaleDollarCnt'. I also calculated the percentage missing values in all the variables in training data.

Scaling of Target Variable:

I used log transformation on the target variable.

Dropping columns:

After calculating the missing values percentage, I found 77% values missing in the training data for variable 'ViewType'. Hence, I dropped the 'ViewType' column. I dropped 'TransDate' column as I found it meaningless for prediction purpose after plotting it against target. I also ended up dropping 'Longitude' column later as I found that was increasing my prediction accuracies.

Removing Outliers:

By plotting the scatterplots of all the variables against the target variable I looked for any prominent outliers. I ended up removing outliers in 'StoryCnt',' GarageSquareFeet', 'FinishedSquareFeet', 'BathroomCnt', 'LotSizeSquareFeet', 'BGPctOwn', 'BGPctVacant'.

Handling missing values:

I filled in missing values in train and test datasets separately. I filled 'GarageSquareFeet', 'BGMedRent', 'BGMedHomeValue' by 0 and 'BGMedYearBuilt' by mode.

Changing column Datatype:

I changed 'BuiltYear', 'BGMedYearBuilt', 'censusblockgroup', 'Usecode' into Categorical variables. As it made more sense for them to be Categorical than numerical.

Removing Skewness:

I used box cox transformation to remove skewness of variables with skewness > 0.75. I used 0.15 as value of lambda. I used scipy library for this.

Using dummies:

I only used dummies to convert the categorical variables in numeric as categorical variables looked nominal.

Building model:

- **Trying different algorithms:**
  Since I had participated in a Kaggle competition before for predicting house prices, I already had an idea for what algorithms would work best. I tried Gradient Boosting, XGboost, LightGBM, Kernel Ridge, Elastic Net, Lasso on the training data. I used Sklearn for all the above algorithms.

- **Calculating RMSE and AAPE:**
  I used 5-fold cross validation to calculate RMSE (Root Mean Square Error) score and AAPE(Average Absolute Percent Error). I wrote a function to calculate AAPE using the formula given in the pdf. After tuning in the parameters, I found following accuracies for the above algorithms on my training dataset.
  RMSE (Root Mean Square Error) Scores:
  1. Gradient Boosting: 0.1807
  2. LightGBM: 0.1822
  3. XGBoost: 0.1866
  4. Kernel Ridge: 0.1992
  5. Lasso: 0.2131
  6. Elastic Net: 0.2134

  AAPE (Average Absolute Percent Error):
  1. Gradient Boosting: 0.1193563245973828 (~ 11.93%)
  2. LightGBM: 0.12422754022042845 (~ 12.42%)
  3. XGBoost: 0.13237407667586443 (~ 13.23%)
  4. Kernel Ridge: 0.13737026450451859 (~ 13.73%)
  5. Elastic Net: 0.14992587493363166 (~ 14.99%)
  6. Lasso: 0.1502296842444727 (~ 15.02%)

- **Stacking:**
  I used stacking of averaged models. A technique I stumbled upon while on the Kaggle competition in past. I used Gradient Boosting and LightGBM as my base models and Elastic Net as the meta model. It gave me best accuracies in terms of RMSE and AAPE.
  RMSE: 0.1778
  AAPE: 0.1189998552730294 (~ 11.899%)