Zillow Zestimate: Home Value Prediction

Business Objective

Buying a house that suits his / her choices is every person's desire and it is thus known as their dream house. There are several aspects that one considers while buying a house starting from the budget, the location, the number of rooms available, and many more. But how to find a house that satisfies one's requirements? This is not a quick and easy task.

But no need to worry, homebuyers can nowadays find their dream home with a click of a button. Zillow is a popular estimator for house evaluation available online. It is considered one of the top real estate marketplaces for buying a house in the United States. Zillow's Zestimate provides the homebuyers to search for a house that satisfies their requirements of location, area, budget, etc.

The Zillow Zestimate provides the homebuyers with information on the real worth of the house based on public data. The accuracy of the Zestimate information depends on the location and availability of the data of a specific area. Hence the more data available the more is the accuracy of the Zestimate.

The purpose of this project is to build a machine learning model that will give the best future sales prediction of homes.

Data Description

The Zillow Zestimate dataset is a dataset from Kaggle, that is used for making future sales predictions and improving the log error.
There are two datasets available –

- train_2016 – This dataset consists of the target variable i.e. the logerror
- properties_2016 – Contents all the features related to the property/home.

There are around 60 attributes in the dataset on basis of which the model can be built.
Some of the important features amongst them are as follows:

- train_2016:

1. parcelid - Unique identification for every parcel
2. transactiondate - The date on which the home was sold
3. logerror - The residual between the actual and the predicted sale price of homes. (Target Array)

- properties_2016:

1. parcelid - Unique identification for every parcel (common in both the datasets)
2. bathroomcnt - Total number of bathrooms in the house
3. bedroomcnt - Total number of bedrooms in the house
4. buildingqualitytypeid - This gives the quality assessment of the building ranging from best to worst.
5. finishedsqaurefeet12 - Finished living area of the house.

6. fips - This stand for Federal Information Processing Standard code (for more detail regarding this check: https://en.wikipedia.org/wiki/FIPS_county_code
7. latitude & longitude - longitude and latitude of the home location
8. propertylandusetypeid - This gives the type of land the property is zoned for.
9. regionidcity - The city in which the property is situated.
10. regionidcounty - The county where the property is situated.
11. regionidzip - This provides the zip code for the location of the property.
12. taxamount - Property tax for each assessment year

Aim

To predict the sale prices of the houses and improve the log error i.e. the error due to the difference between the actual and the predicted home values.

Tech stack

- Language - Python
- Libraries - Scikit-learn, pandas, numpy, matplotlib, seaborn, scipy, xgboost, joblib

Approach

1. Importing the required libraries and reading the dataset.
    - Merging of the two datasets
    - Understanding the dataset
2. Exploratory Data Analysis (EDA) –
    - Data Visualization
3. Feature Engineering
    - Duplicate value removal
    - Missing value imputation
    - Rescaling of incorrectly scaled data
    - Standardization
    - Encoding of categorical variables
    - Generation of new feature wherever required.
    - Dropping of redundant feature columns
    - Checking for multi-collinearity and removal of highly correlated features
    - Check for the outliners and removal of outliers.

4. Model Building
    - Performing train test split
    - Feature Scaling
    - Dropping features if necessary
    - Linear Regression Model
    - Elastic Net
    - Ridge Regression
    - Lasso Regressor
    - XGBoost Regressor

- Adaboost Regressor
- Gradient Boosting Regressor
- Decision Tree Regressor
- Random Forest Regressor

5. Model Validation
    - Mean Absolute Error
6. Hypermeter Tuning (GridSearchCV)
    - For Random Forest Regressor
7. Checking for Feature Importance
8. Creating the final model and making predictions

Modular code overview

```
input
  |_properties_2016.csv
  |_train_2016.csv

src
  |_engine.py
  |_ML_Pipeline
            |_duplicates.py
            |_grid_model.py
            |_label_encoding.py
            |_missing_values.py
            |_model_evaluation.py
            |_outliers.py
            |_rescale_variables.py
            |_scaler.py
            |_train_model.py
            |_utils.py

lib
  |_Zillow_Home_Value_Value_Prediction_EDA.ipynb
  |_Zillow_Home_Value_Feature_Engineering.ipynb
  |_Zillow_Home_Value_Model_Creation.ipynb


output
  |_final_best_fitted_model.pkl
```

Once you unzip the modular_code.zip file you can find the following folders within it.

1. input

2. src

3. output

4. lib

1. Input folder - It contains all the data that we have for analysis. There are two csv files in our case,
    - train_2016_v2
    - properties_2016

2. src folder - This is the most important folder of the project. This folder contains all the modularized code for all the above steps in a modularized manner. This folder consists of:
    - engine.py
    - ML_Pipeline

The ML_pipeline is a folder that contains all the functions put into different python files which are appropriately named. These python functions are then called inside the engine.py file.

3. Output folder – The output folder contains the best fitted model that we trained for this data. This model can be easily loaded and used for future use and the user need not have to train all the models from the beginning.
   Note: This model is built over a chunk of data. One can obtain the model for the entire data by running engine.py by taking the entire data to train the models.

4. lib folder - This is a reference folder. It contains the original ipython notebook that we saw in the videos.

Project Takeaways

1. Understanding the business problem.
2. Importing the dataset and required libraries.
3. Performing basic Exploratory Data Analysis (EDA).
4. Data cleaning and missing data handling if required, using appropriate methods.
5. Checking data distribution using statistical techniques.
6. Checking for outliers and how they need to be treated as per the model selection.
7. Using python libraries such as matplotlib and seaborn for data interpretation and advanced visualizations.
8. Splitting Dataset into Train and Test using various sampling techniques
9. Performing Feature Engineering on sample data for better performance.

10. Training a model using Regression techniques like Linear Regression, Random Forest Regressor, XGBoost Regressor, etc.
11. Training multiple models using different Machine Learning Algorithms suitable for the scenario and checking for best performance.
12. Understanding feature scaling importance and applying them if required.
13. Performing Cross Validation to check if the model is overfitting and whether results are somewhat constant.
14. Tuning hyper-parameters of models to achieve optimal performance.
15. Making predictions using the trained model.
16. Gaining confidence in the model using metrics such as MAE, MSE, RMSE.
17. What features are most helpful for predictive power using Feature Importance.
18. How Target variable is dependent on the values of Input features.
19. Selection of the best model based on performance metrics and Hyper-Parameter Optimization.