Project Report

Dataset Used : Melbourne Housing Dataset

Steps involved:

- **Data Tidying**
- **Feature Engineering**
- **Model Selection**

Data Tidying

- First and foremost, after importing necessary libraries, the dataset was Cross checked to see how many missing values were present.

- Missing values were present in:
  Distance, Postcode, Rooms, Bathroom, Car, Landsize, BuildingArea, YearBuilt, Latitude, Longitude and the column of Propertycount.

- For filling the missing values in YearBuilt Column, we went with the assumption that in a particular Suburb, all the houses that are existing there were mostly built in and around the same time or at max within a gap of 1-2 years. So we grouped YearBuilt according to Suburb column and filled the missing year values in YearBuilt with the median of each of the groupings depending on which Suburb it belonged to.

- For Postcode column missing values were filled after grouping them together Suburb wise since for each Suburb the postcode would be the same. Once the grouping was done, the median value from each group was used to fill in the missing values. Just in case a grouping could'nt be found for a particular suburb value, most_frequent value was then used for filling that particular postcode value. For CouncilArea and

RegionName and Bathroom columns, the missing values were replaced with the most frequent ones.

- For the purpose of filling the Latitude and Longitude columns, we took the following approach: Since the Latitude and Longitude were denoted with four places of decimals which can easily differentiate between different councils or suburbs of a region, so we grouped the latitudes and longitudes of the various places according to the Suburbs, Postcodes and CouncilAreas and filled the missing values with the median latitude/longitude values corresponding to the groupings of each particular suburb, postcode and councilarea. The remaining missing values which could'nt be grouped(0.3%of the total values) were filled with most frequent values.

- PropertyCount missing values were filled with the median value of the entire column and the Distance column was filled with the mean value of the entire column.

- For filling missing values in Car column, we first grouped the Car column as per the TotalRooms column, then filled it with the median of each group since any property will always have garage space in proportion to the number of rooms, it won't obviously be higher than that. The risk of inputing more car space than total rooms increases if we directly replace missing values with mean or median.

- Landsize and Building Area columns were filled with their median values. Some values of the Landsize column had zeroes in them which is quite absurd, they were replaced with the median as well.

Feature Engineering

- On closer inspection of the dataset, we will notice that for some columns(even before missing values were filled) LandSize was less than the Total BuildingArea which did’nt make much sense. So the values of such rows where LandSize < BuildingArea were swapped.

- Apart from that, an important parameter which can affect the price of a property is the available Lawn Area. Obviously, a spacious property(having more lawn area) would most probably fetch a higher price than a crowded for space property. So a new column named LawnSpace was created by subtracting BuildingArea from the LandSize column. The LandSize column was then dropped because we have all the info regarding land measurements of the property.

- If we observe the data closely, then we will see that Rooms column is exactly the replica of Bedroom2 column, with some additional missing values. So, Rooms column was dropped and replaced with Bedroom2 column.

- An additional column named TotalRooms was created which denoted the sum of all the rooms of a property(excluding garage space) as generally, a property with more rooms would fetch a higher price than a property which is smaller in size.

- Label Encoding was done for the categorical variables.

- Skewness of the dataset was accounted for and it turns out some columns were highly skewed. In order to get rid of the skewness boxcox transformation was done to the columns where skewness was greater than 0.75.

- The target variable price was also found to right skewed and hence log1p transformation which applies log(1+x) to all values of a column was applied to the Price column.

- Columns like Regionname and CouncilArea were dropped as they were only creating redundancy in the dataset when so many other location indicating paramaters like latitude, longitude, suburb, postcode etc were already present.

Model Selection

- For the purpose of model selection, we first defined the function to calculate the root mean square logarithmic error(rmsle) and also to calculate the r_2 score.

- The dataset was split into training and evaluation sets and a total of 3 models: LightGBM, GradientBoostingRegressor and RandomForestRegressor were tried out on the basis of their performance on the evaluation set(or test set). GridSearch was implemented for finding the appropriate set of parameters for the gradient boosting algorithms, after finding the r2_score and rmsle error of each individual model on the evaluation set, we stacked together a single model comprising of contributions from each individual model based on their performance in the evaluation(test) set. A 50% contribution from GradientBoostingRegressor, and 25% each from LightGBM and RandomForest comprised our final stacked model.

- Test set prediction was then made using the stacked up model and saved in a .csv file.

Submitted by:
Arindam Baruah(ECE, 4thsem)