**Exploratory Data Analysis**

First Step was to examine the data and identify missing values/outliers/ inconsistent values and treat them accordingly

* The **Item Weight** Attribute had missing values. It was also observed that majority of the missing values could be treated by picking up the value corresponding to the same Item Identifier from another record. *(For a particular item ID, some of the records had missing values. Hence the missing values could be imputed from the weight values in the remaining records for the same Item ID).* For some item IDs, there were no weight values in any of the records. Since there were only 4 records of such type left in the data with missing value , we dropped them from the training dataset
* The **Outlet Size** had missing values for some outlets. The values were imputed from other Outlets having similarity in terms of **Total Sales** through the outlet and **Distinct Item types** sold from that outlet
* There were many records (~9% of the data) where **item visibility was 0.**
  + Approach 1 (**refer BigDataEDA\_1.ipynb**)-We first tested by dropping all these records and then fitting the model on the remaining data. Post that , we predicted values for the test submission data. The solution yielded a score of **1149.175**
  + Approach 2(**refer BigDataEDA\_2.ipynb**)-- We then tested by imputing the values for item visibility based on KNN imputer with nearest two neighbours looking at similarity based on Item Attributes (Item Type, Weight, MRP) and Store attributes(Location, Size, type, Item Sales). Post that , we predicted values for the test submission data. The solution yielded a score of **1152.77**
  + Since Approach 1 yielded comparatively better results , we continued with the same
* We transformed the Establishment Year attribute of an outlet to a numerical value highlighting years since established w.r.t to 2013 which could be more useful in the model building process

**Model Fitting and Validation**

* To ensure that the model was robust to outliers and multicollinearity among, we decided to use Tree Based Algorithms ( Decision Tree, Random Forest, XG Boost). Also, the data consisted of many categorical variables which were better suited to tree based algorithms
* All the categorical variables were encoded ( label encoding/one hot encoding for ordinal/nominal variables respectively)
* We then created an OOS(Out of sample) dataset based on ~5% of the original training data. The remaining data would then be split into train and validation sets to fit individual models and for hyperparameter tuning. For each of models, we used grid search to identify the best parameters
* The models were then tested on the OOS test set to compare performances of the models on the same unseen data. While the performance of individual models was consistent across the validation and OOS datasets, XG Boost model was observed to be performing better on the OOS data amongst the 3 models. Hence , XG Boost was selected to predict **Item Outlet Sale** values for the final test submission

**PTO**

**Note**

* Refer file **BigDataEDA\_1.ipynb**  for final EDA approach
* Refer file **BigDataMart\_ModelTraining\_Validation.ipynb** for model development and validation
* **train\_v9rqX0R.csv** is the raw training data downloaded from the hackathon portal
* **test\_AbJTz2l.csv** is the raw testing data downloaded from the hackathon portal to generate predictions for final submission
* **updated\_train.csv** is the updated training data generated from **BigDataEDA\_1.ipynb**  to be used in **BigDataMart\_ModelTraining\_Validation.ipynb**
* **FinalSubmission\_Hackathon,csv**  is the final set of predictions generated for testing on the hackathon portal. This solution yielded a score of **1149.175** and leaderboard rank of **577** The fie is generated through **BigDataMart\_ModelTraining\_Validation.ipynb.**