"Big Mart Sales Prediction"

Submitted By:
Mounisha.S (AP18110010337)
Himanshi Chauhan(AP18110010374)
Asritha.P(AP181110010378)
Veda Gayathri Ravi(AP18110010381)
Aparna.G(AP18110010662)
of
GROUP-24
IN
CSE-F (006)

Abstract

Sales forecasting is a crucial aspect of various companies engaged in retailing, logistics, manufacturing, marketing and wholesaling. It allows companies to efficiently allocate resources, to estimate achievable sales revenue and to plan a far better strategy for future growth of the corporate. In this paper, prediction of sales of a product from a specific outlet is performed via a two-level approach that produces better predictive performance compared to any of the popular single model predictive learning algorithms. The approach is performed on Big Mart Sales data of the year 2013. Data exploration, data transformation and have engineering play an important role in predicting accurate results. The result demonstrated that the two-level statistical approach performed better than one model approach because the former provided more information that results in better prediction.

Introduction:-

This is an in-depth exploratory report on the big Mart Sales challenge. It's a regression practice problem wherein we've to predict sales product-wise and store-wise. Sales forecasting as well as analysis of sale forecasting has been conducted by many authors as summarized. Statistical and computational methods are studied in also this paper elaborates the automated process of data acquisition. Machine learning is that the process where a machine will learn from data within the sort of statistically or computationally method and process knowledge acquisition from experiences

Problem Survey

Pat Langley and Herbert A aimed out most broadly used data mining technique in a Comparative Study of Big Mart Sales Prediction 3 the field of business is the Rule Induction (RI) technique as compared to other data mining techniques whereas sale prediction of a pharmaceutical distribution company has been described in. In this paper focuses on two issues

- Stock state shouldn't experience out of stock, and
- It avoids the client dissatisfaction by predicting the sales that manages the stock level of medicines. Handling of footwear sale fluctuation in a period of time has been addressed in.

This paper focuses on using neural network for predicting of weekly retail sales, which downscale the uncertainty present in the short-term planning of sales. Linear and non-linear a comparative analysis model for sales forecasting is proposed for the retailing sector. Beheshti-Kashi and Samaneh performed sales prediction in fashion market. A two-level statistical method is elaborated for forecasting the big mart sales prediction. Xia and Wong proposed the differences between classical methods (predicated on mathematical and statistical models) and modern heuristic approaches and also named exponential smoothing, regression, auto regressive integrated moving average (ARIMA), generalized auto regressive conditionally heteroskedastic (GARCH) approaches. Most of these models are linear and aren't suitable to deal with the asymmetric behavior in most real- world deals data. Some of the challenging factors like lack of

Dataset description and Preprocessing:-

We use data.table's fread() to speed up reading in the datasets.It's good to quicky check the dimensions of our data.

Train dataset has 8523 rows and 12 features and test has 5681 rows and 11 columns. train has 1 extra column which is the target variable. We will predict this target variable for the test dataset later in this tutorial.

Feature names of train and test datasets

```
[1] "Item_Identifier"            "Item_Weight"
"Item_Fat_Content"           "Item_Visibility"

 [5] "Item_Type"                 "Item_MRP"
"Outlet_Identifier"          "Outlet_Establishment_Year"

 [9] "Outlet_Size"               "Outlet_Location_Type"      "Outlet_Type"               "Item_Outlet_Sales"
```

```
[1] "Item_Identifier"            "Item_Weight"
"Item_Fat_Content"           "Item_Visibility"

 [5] "Item_Type"                 "Item_MRP"
"Outlet_Identifier"          "Outlet_Establishment_Year"

 [9] "Outlet_Size"               "Outlet_Location_Type"      "Outlet_Type"
```

Item_Outlet_Sales is present in train but not in test dataset because this is the target variable that we have to predict.

Structure of train and test datasets

```
Classes 'data.table' and 'data.frame':  8523 obs. of  12 variables:

 $ Item_Identifier          : chr  "FDA15" "DRC01" "FDN15" "FDX07" ...

 $ Item_Weight              : num  9.3 5.92 17.5 19.2 8.93 ...

 $ Item_Fat_Content         : chr  "Low Fat" "Regular" "Low Fat" "Regular" ...

 $ Item_Visibility          : num  0.016 0.0193 0.0168 0 0 ...
```

```
 $ Item_Type                : chr  "Dairy" "Soft Drinks" "Meat" "Fruits and Vegetables" ...

 $ Item_MRP                 : num  249.8 48.3 141.6 182.1 53.9 ...

 $ Outlet_Identifier        : chr  "OUT049" "OUT018" "OUT049" "OUT010" ...

 $ Outlet_Establishment_Year: int  1999 2009 1999 1998 1987 2009 1987 1985 2002 2007 ...

 $ Outlet_Size              : chr  "Medium" "Medium" "Medium" "" ...

 $ Outlet_Location_Type     : chr  "Tier 1" "Tier 3" "Tier 1" "Tier 3" ...

 $ Outlet_Type              : chr  "Supermarket Type1" "Supermarket Type2" "Supermarket Type1" "Grocery Store" ...

 $ Item_Outlet_Sales        : num  3735 443 2097 732 995 ...

 - attr(*, ".internal.selfref")=<externalptr>

Classes ‘data.table’ and 'data.frame':  5681 obs. of  11 variables:

 $ Item_Identifier          : chr  "FDW58" "FDW14" "NCN55" "FDQ58" ...

 $ Item_Weight              : num  20.75 8.3 14.6 7.32 NA ...

 $ Item_Fat_Content         : chr  "Low Fat" "reg" "Low Fat" "Low Fat" ...

 $ Item_Visibility          : num  0.00756 0.03843 0.09957 0.01539 0.1186 ...

 $ Item_Type                : chr  "Snack Foods" "Dairy" "Others" "Snack Foods" ...

 $ Item_MRP                 : num  107.9 87.3 241.8 155 234.2 ...

 $ Outlet_Identifier        : chr  "OUT049" "OUT017" "OUT010" "OUT017" ...

 $ Outlet_Establishment_Year: int  1999 2007 1998 2007 1985 1997 2009 1985 2002 2007 ...

 $ Outlet_Size              : chr  "Medium" "" "" "" ...

 $ Outlet_Location_Type     : chr  "Tier 1" "Tier 2" "Tier 3" "Tier 2" ...

 $ Outlet_Type              : chr  "Supermarket Type1" "Supermarket Type1" "Grocery Store" "Supermarket Type1" ...

 - attr(*, ".internal.selfref")=<externalptr>
```

To explore data in any data science competetion, it is advisable to append test data to the train data. So, we will combine both train and test to carry out data visualization, feature engineering, one-hot encoding, and label encoding. Later we would split this combined data back to train and test datasets.

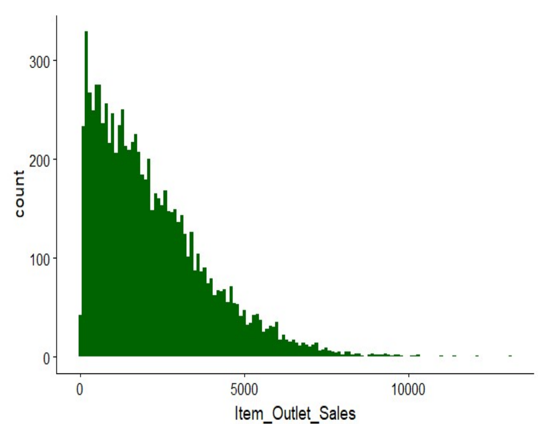
Exploratory Data Analysis

Univariate Analysis

We will start off by plotting and exploring all the individual variables to gain some insights. In this tutorial ggplot2 package has been used to generate all the plots.

Target Variable

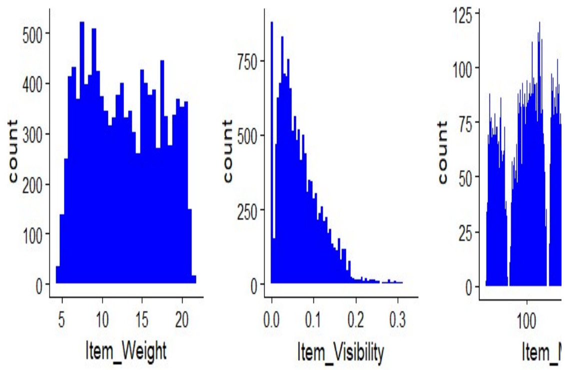
Since our target variable is continuous, we can visualise it by plotting its histogram.



As you can see, it is a right skewed variable and would need some data transformation to treat its skewness. We will come back to it later.

Independent Variables (numeric variables)

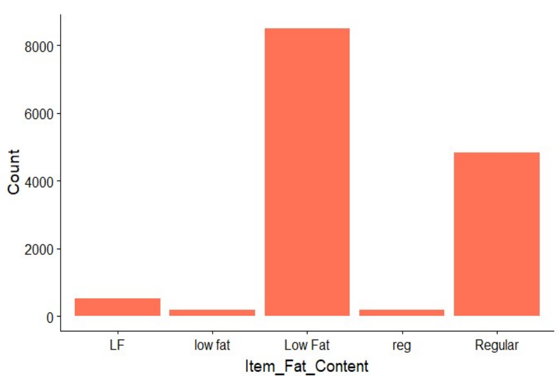
Now let's check the numeric independent variables. We'll again use the histograms for visualizations because that will help us in visualizing the distribution of individual variables.



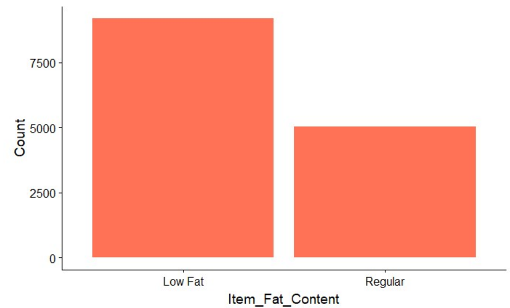
As you can see, there is no clear pattern in Item_Weight and Item_MRP. However, Item_Visibility is right-skewed and should be transformed to curb its skewness.

Independent Variables (categorical variables)

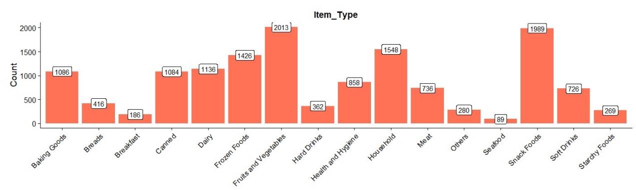
Now we'll try to explore and gain some insights from the categorical variables. A categorical variable or feature can have only a finite set of values. Let's first plot Item_Fat_Content.

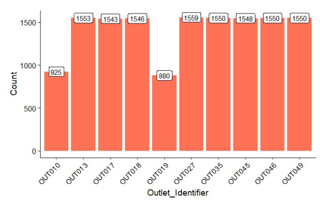


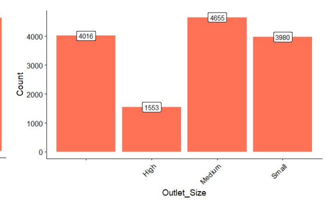
In the figure above, 'LF', 'low fat', and 'Low Fat' are the same category and can be combined into one. Similarly, we can be done for 'reg' and 'Regular' into one. After making these corrections we'll plot the same figure again.



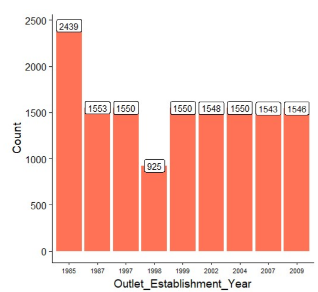
Now let's check the other categorical variables.

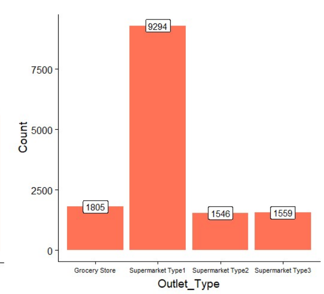




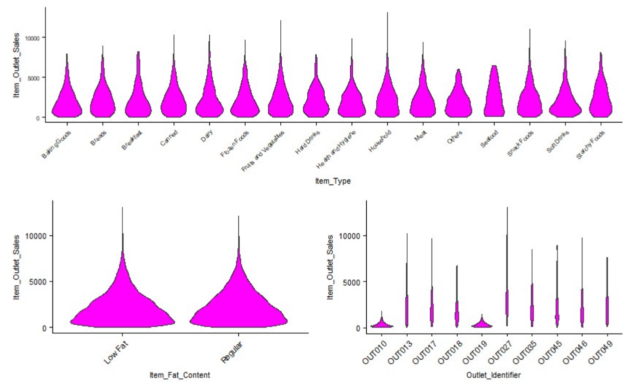


In Outlet Size's plot, for 4016 observations, Outlet_Size is blank or missing. We will check for this in the bivariate analysis to substitute the missing values in the Outlet_Size.





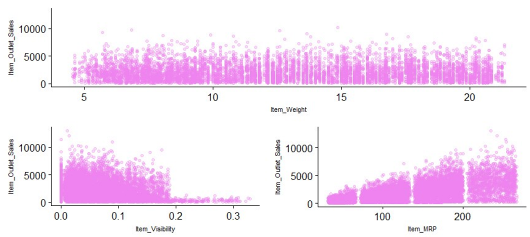
There are lesser number of observations in the data for the outlets established in the year 1998 as compared to the other years. Supermarket Type 1 seems to be the most popular category of Outlet_Type.



Distribution of Item_Outlet_Sales across the categories of Item_Type is not very distinct and same is the case with Item_Fat_Content. However, the distribution for OUT010 and OUT019 categories of Outlet_Identifier are quite similar and very much different from the rest of the categories of Outlet_Identifier.

Bivariate Analysis

After looking at every feature individually, let's now explore them again with respect to the target variable. Here we will make use of scatter plots for continuous or numeric variables and violin plots for the categorical variables.

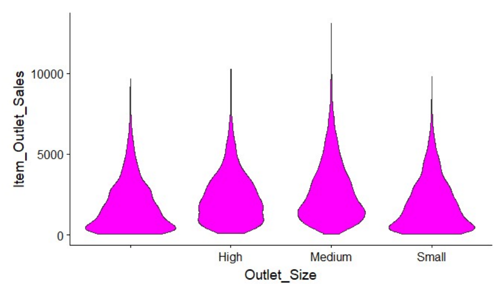


Item_Outlet_Sales is spread well across the entire range of the Item_Weight without any obvious pattern. In the Item_Visibility vs Item_Outlet_Sales, there is a string of points at Item_Visibility = 0.0 which seems strange as item visibility cannot be completely zero. We will take note of this issue and deal with it in the later stages.

In the third plot of Item_MRP vs Item_Outlet_Sales, we can clearly see 4 segments of prices that can be used in feature engineering to create a new variable.

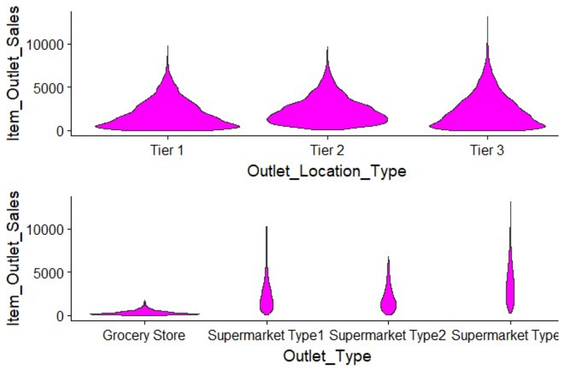
Now we'll visualise the categorical variables with respect to Item_Outlet_Sales. We will try to check the distribution of the target variable across all the categories of each of the categorical variable. We can do this using boxplot but instead we'll use the violin plots as they show the full distribution of the data. The horizontal width of a violin plot at a particular level indicates the concentration of data at that level.

In the univariate analysis, we came to know about the empty values in Outlet_Size variable. Let's check the distribution of the target variable across Outlet_Size.



The distribution of 'Small' Outlet_Size is almost identical to the distribution of the blank category (first vioin) of Outlet_Size. So, we can substitute the blanks in Outlet_Size with 'Small'. Please note that this is not the only way to impute missing values, but for the time being we will go ahead and impute the missing values with 'Small'.

Let's examine the remaining variables.



Tier 1 and Tier 3 locations of Outlet_Location_Type look similar. In the Outlet_Type, Grocery Store has most of its data points around the lower sales values as compared to the other categories. These are the kind of insights that we can extract by visualizing our data. Hence, data visualization should be an important part of any kind data analysis.

Missing Value Treatment

Missing data can have a severe impact on building predictive models because the missing values might be containing some vital information which could help in making better predictions. So, it becomes imperative to carry out missing data imputation. There are different methods to treat missing values based on the problem and the data. Some of the common techniques are as follows:

1. **Deletion of rows**: In train dataset, observations having missing values in any variable are deleted. The downside of this method is the loss of information and drop in prediction power of model.
2. **Mean/Median/Mode Imputation**: In case of continuous variable, missing values can be replaced with mean or median of all known values of that variable. For categorical variables, we can use mode of the given values to replace the missing values.
3. **Building Prediction Model**: We can even make a predictive model to impute missing data in a variable. Here we will treat the variable having missing data as the target variable and the other variables as predictors. We will divide our data into 2 datasets—one without any missing value for that variable and the other with missing values for that variable. The former set would be used as training set to build the predictive model and it would then be applied to the latter set to predict the missing values.

As you can see above, we have missing values in Item_Weight and Item_Outlet_Sales. Missing data in Item_Outlet_Sales can be ignored since they belong to the test dataset. We'll now impute Item_Weight with mean weight based on the Item_Identifier variable. Replacing 0's in Item_Visibility variable

Feature Engineering

Most of the times the given features in a dataset are not enough to give satisfactory predictions. In such cases, we have to create new features which might help in improving the model's performance. Let's try to create some new features for our dataset.

We can have a look at the Item_Type variable and classify the categories into perishable and non-perishable as per our understanding and make it into a new feature.

Let's compare Item_Type with the first 2 characters of Item_Identifier, i.e., 'DR', 'FD', and 'NC'. These identifiers most probably stand drinks, food, and non-consumable.

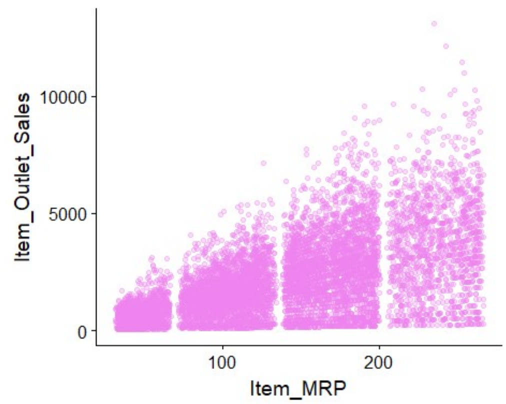
	DR	FD	NC
Baking Goods	0	1086	0
Breads	0	416	0

Breakfast	0	186	0
Canned	0	1084	0
Dairy	229	907	0
Frozen Foods	0	1426	0
Fruits and Vegetables	0	2013	0
Hard Drinks	362	0	0
Health and Hygiene	0	0	858
Household	0	0	1548
Meat	0	736	0
Others	0	0	280
Seafood	0	89	0
Snack Foods	0	1989	0
Soft Drinks	726	0	0
Starchy Foods	0	269	0

Based on the above table we can create a new feature. Let's call it Item_category.

We will also change the values of Item_Fat_Content wherever Item_category is 'NC' because non-consumable items cannot have any fat content. We will also create a couple of more features — Outlet_Years (years of operation) and price_per_unit_wt (price per unit weight).

Earlier in the Item_MRP vs Item_Outlet_Sales plot, we saw Item_MRP was spread across in 4 chunks. We can use k Means clustering to create 4 groups using Item_MRP variable. K Means clustering requires prior knowledge of K i.e., no. of clusters we want to divide our data into. Here we will go ahead with K=4. Let's try it out.



Let's use these clusters as a new independent feature in our data.

Encoding Categorical Variables

In this stage, we will convert our categorical variables into numerical ones. We will use 2 techniques — Label Encoding and One Hot Encoding.

1. **Label encoding** simply means converting each category in a variable to a number. It is more suitable for ordinal variables — categorical variables with some order.
2. In **One hot encoding**, each category of a categorical variable is converted into a new binary column (1/0).

We will use both the encoding techniques.

Label encoding for the categorical variables

We will label encode Outlet_Size and Outlet_Location_Type as these are ordinal variables. One hot encoding for the categorical variable

Preprocessing Data

Before feeding our data into any model, it is a good practice to pre-process the data. We will do pre-processing on both independent variables and target variable

Checking Skewness

Skewness in variables is undesirable for predictive modelling. Some machine learning methods assume normally distributed data and a skewed variable can be transformed by taking its log, square root, or cube root so as to make the distribution of the skewed variable as close to normal distribution as possible.

Scaling numeric predictors

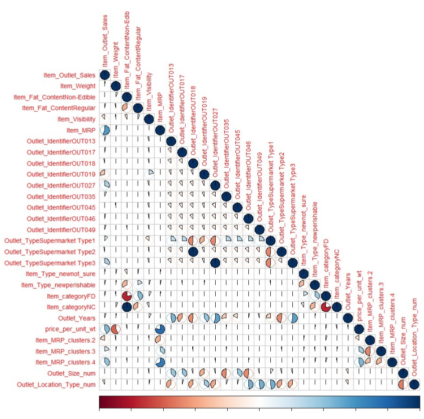
Let's scale and centre the numeric variables to make them have a mean of zero, standard deviation of one and scale of 0 to 1. Scaling and cantering are required for linear regression models.

Correlated Variables

Let's examine the correlated features of train dataset. Correlation varies from -1 to 1.

1. negative correlation: < 0 and >= -1
2. positive correlation: > 0 and <= 1
3. no correlation: 0

It is not desirable to have correlated features if we are using linear regressions.



Variable's price_per_unit_wt and Item_Weight is highly correlated as the former one was created from the latter. Similarly, price_per_unit_wt and Item_MRP are highly correlated for the same reason.

Modeling

Finally, we have arrived at most interesting stage of the whole process — predictive modelling. We will start off with the simpler models and gradually move on to more sophisticated models. We will build the models using…

- Linear Regression
- Lasso Regression
- Ridge Regression
- Random Forest
- XGBoost

We will use **5-fold cross validation** in all the models we are going to build. Basically, cross validation gives an idea how well a model generalizes to unseen data.

Implementation and Resultant graphs:-

Linear Regression

Linear regression is the simplest and most widely used statistical technique for predictive modelling. Given below is the linear regression equation:

$$Y = \theta_1 X_1 + \theta_2 X_2 + \dots \theta_n X_n$$

where X1, X2,…,Xn are the independent variables, Y is the target variable and all thetas are the coefficients. Magnitude of a coefficient wrt to the other coefficients determines the importance of the corresponding independent variable.

For a good linear regression model, the data should satisfy a few assumptions. One of these assumptions is that of absence of multicollinearity, i.e., the independent variables should be correlated. However, as per the correlation plot above, we have a few highly correlated independent variables in our data. This issue of multicollinearity can be dealt with regularization.

For the time being, let's build our linear regression model with all the variables.

Making predictions on the test dataset

Leader board score: 1202.26

Regularised regression models can handle the correlated independent variables well and helps in overcoming overfitting. **Ridge** penalty shrinks the coefficients of correlated predictors towards each other while the **Lasso** tends to pick one of a pair of correlated features and discard the other. The tuning parameter **lambda** controls the strength of the penalty.

Lasso Regression

Leader board score: 1202.26

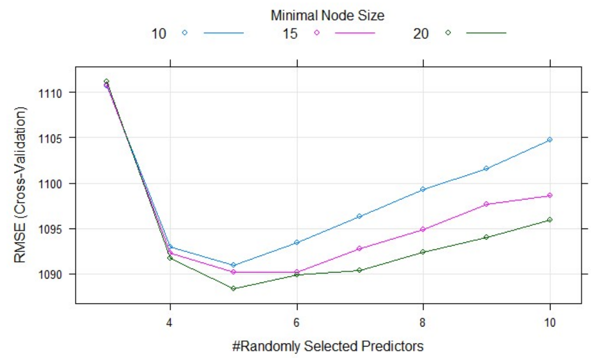
Ridge Regression

Leader board score: 1219.08

RandomForest

RandomForest is a tree based bootstrapping algorithm wherein a certain no. of weak learners (decision trees) is combined to make a powerful prediction model. For every individual learner, a random sample of rows and a few randomly chosen variables are used to build a decision tree model. Final prediction can be a function of all the predictions made by the individual learners. In case of regression problem, the final prediction can be mean of all the predictions

We will now build a RandomForest model with 400 trees. The other tuning parameters used here are mtry — no. of predictor variables randomly sampled at each split, and min.node.size — minimum size of terminal nodes (setting this number large causes smaller trees and reduces overfitting).



As per the plot shown above, the best score is achieved at mtry = 5 and min.node.size = 20.

Leader board score: 1157.25

XGBoost Model

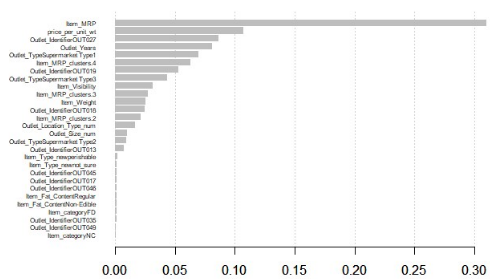
XGBoost is a fast and efficient algorithm and has been used to by the winners of many data science competitions. It's a boosting algorithm and works only with numeric variables and we have already done that. There are many tuning parameters in XGBoost which can be broadly classified into General Parameters, Booster Parameters and Task Parameters.

- General parameters refer to which booster we are using to do boosting. The commonly used are tree or linear model
- Booster parameters depends on which booster you have chosen
- Learning Task parameters that decides on the learning scenario, for example, regression tasks may use different parameters with ranking tasks.

Let's have a look at the parameters that we are going to use in our model.

1. **eta**: It is also known as the learning rate or the shrinkage factor. It actually shrinks the feature weights to make the boosting process more conservative. The range is 0 to 1. Low eta value means model is more robust to overfitting.
2. **gamma**: The range is 0 to ∞. Larger the gamma more conservative the algorithm is.
3. **max_depth**: We can specify maximum depth of a tree using this parameter.
4. **subsample**: It is the proportion of rows that the model will randomly select to grow trees.
5. **colsample_bytree**: It is the ratio of variables randomly chosen for build each tree in the model.

Variable Importance



As expected Item_MRP is the most important variable in predicting the target variable. New features created by us, like price_per_unit_wt, Outlet_Years, Item_MRP_Clusters, are also among the top most important variables. This is why feature engineering plays such a crucial role in predictive modelling.

Conclusion: -

After trying and testing 5 different algorithms, the best score on the public leader board is achieved by XGBoost (1154.70), followed by RandomForest (1157.25). This report helps you in understanding how a machine learning competition is approached and what are the steps one should go through to build a robust model.