**Approach and Steps for Big Mart Sales Prediction Hackathon**

1. **Analysis of the data set:**

I began with the analysis of the datasets which was given as part of input.

1. **Exploratory Data Analysis:**

EDA of the numerical and categorical features were done both univariate and bivariate to understand how the distributions of the categorical and numerical variables are. Below were the observations that were made:

* + The Item\_Weight ranges from 5- 20 kg
  + Item\_Visibility has a right skewed distribution
  + Most of the Item's MRP ranges from Rs 100 to 180
  + Most of the outlets have been establised in the year of 1985, 1997 to 1999
  + Item\_Outlet\_Sales is right skewed
  + Item\_Weight, Item\_MRP and OutletEstablishment\_Year has no outliers
  + Item\_Visibility and Item\_Outlet\_Sales has outliers in the data
  + the distribution of data both in test and train are almost similar
  + Item\_Fat\_Content feature needs to be cleaned
  + LF, Low Fat and low fat should be made the same category "Low Fat"
  + reg and Regular should be mapped to same category "Regular"
  + Outlet\_Location\_type can be encoded as per the tier number

1. **Missing Value Treatment:**

Some of the columns like Item\_Weight (numerical) and Outlet\_Size(categorical) had missing values. Mean and mode was used to fill the two columns respectively.

1. **Feature engineering:**

As part of Feature engineering following steps were taken:

* + Outliers were capped in the columns where Outliers were seen during the EDA phase
  + Item\_Fat\_Content column had incorrect categories which was corrected accordingly
  + Outlet\_Age column was created to indicate the for how many years the outlet has been working
  + Broader category of item was created from the Item identifier column to indicate to which category the item belongs to- Food, Drinks or Non consumable
  + Label encoding was done for some of the ordinal categorical columns
  + One-hot encoding was done for the other remaining categorical columns

1. **Model Building:**

As part of model building the following steps were carried out:

* + The train data was split into train and validation sets.
  + Different models- Linear, Ridge, Lasso (with scaled data), GradientBoost, RandomForest, XGBoost and LightGBM were compared to see which model gave the better training accuracy, test accuracy and minimal RMSE score.
  + LightGBM and GradientBoost performed well with similar RMSE score.
  + Tried Hyperparameter tuning with both LightGBM and GradientBoost
  + LightGBM with hyperparameter tuning worked the best with test data and it provided the lowest RMSE score when evaluated on the hackathon platform.