Solution Approach Document

1. Approach

I used XGBoost regressor to train the model.

For checking, I divided the training data into train and validation sets and evaluation metric as Mean Squared Log Error * 1000.

Training data was divided into train and alidation sets in the following ratio: -

1. Train - All except last 2 months data
2. Validation -- last 2 months data (It is a good practice to maintain validation set at least as large as test data or forecasted results)

I trained XGBoost model with following parameters:

n_estimators = 1000, objective='reg:linear', ,early_stopping_rounds=50,verbose=True

and got MSLE score of 231.2.

2. Data-preprocessing / feature engineering ideas

1. I grouped data by the Store_Type, Location_Type and Region_Code individually and found the mean of sales. The category with highest mean got the highest label-encoding value.

2. I did one hot encoding for Discount column and dropped the first column since that might lead to highly correlated independent columns.

3. Final Model

It was a time-series problem. So I first thought of fitting time forecasting models like ARIMA, SARIMAX, VAR and alike. But they seem to be a poor choice since the problem statement was predicting the sales for a particular store and store location, so individually for store we didn't have enough data.

Next, I decided to take up Boosting approach "LightGBM" by intoducing lags, rotational, Exponentially Weighted Moving Averages columns in the dataset. Upon, submission the result didn't turn out to be that good. I could notice some overfitting happening so I did PCA to redice the dimensions of the dataset. After I performed that scores improved but only slightly. So I decided to change my approach.

I tried taking tree-based approach, since the data was not very large and it is comparatively easy to do hyperparameter tuning on "Random-Forest". moreover, I could notice some outliers in the data which tree based algorithms could have handled easily. I got a good improvement in my scores this time.

My curiosity didn't stop there, I wanted to try XGBoost too. Let's give another shot to Boosting! When I trained that, doing some hyperparameter tuning(the best selection of parameters is in the notebook). I got the best score I got so far!!. XGBoost worked for me this time as well! I trained XGBoost on the given data by doing Label encoding and one hot encoding for categorical columns.

To, improve my score even more I tried introducing lag(shift), rotation and other EWMA columns, introduced weekend element also but didn't get very good score, so I decided to stick to XGBoost with given variables only.