Analytics Vidya Job A Thon- September 2021

Supplement Sales Prediction

By
Prasad Vasant Verulkar

Problem Statement

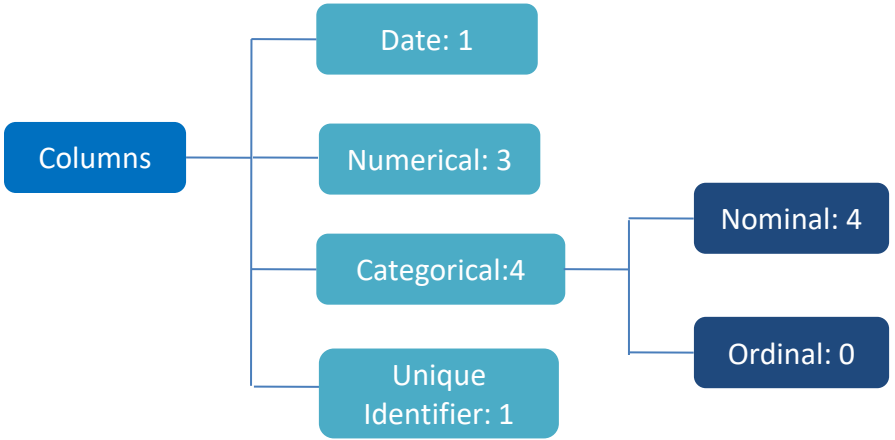
To predict the store sales for each store in the test set for the next two months.

Dataset Description

Train Data

Rows: 188340, Columns: 10
Y Variable: Sales

#	Column	Non-Null Count	Dtype
0	ID	188340 non-null	object
1	Store_id	188340 non-null	int64
2	Store_Type	188340 non-null	object
3	Location_Type	188340 non-null	object
4	Region_Code	188340 non-null	object
5	Date	188340 non-null	datetime64[ns]
6	Holiday	188340 non-null	int64
7	Discount	188340 non-null	object
8	#Order	188340 non-null	int64
9	Sales	188340 non-null	float64



Test Data

Rows: 22265, Columns: 8

#	Column	Non-Null Count	Dtype
0	ID	22265 non-null	object
1	Store_id	22265 non-null	int64
2	Store_Type	22265 non-null	object
3	Location_Type	22265 non-null	object
4	Region_Code	22265 non-null	object
5	Date	22265 non-null	datetime64[ns]
6	Holiday	22265 non-null	int64
7	Discount	22265 non-null	object

Steps in Building ML Model

1. **Reading Data**

2. **Data Preprocessing**
   - Handling Missing values
   - Handling Outliers
   - Check for Duplicates

3. **Exploratory Data Analysis**
   Analysing data to reveal hidden insights.

4. **Feature Engineering**
   - Generating New Features
   - Categorical Encodings

5. **Setting up Validation Strategy**
   - Splitting train and validation set based on date.

6. **Feature Scaling**
   MinMax Scaler

7. **Feature Selection**
   Since features in dataset were already less, hence feature selection is not performed.

8. **Model Building**

9. **Cross Validation**
   - To make model robust and reduce over fitting

Check for Missing Values

- No missing Values in Dataset

Train

```
# 0 missing values
train.isnull().sum()
```

```
ID                0
Store_id          0
Store_Type        0
Location_Type     0
Region_Code       0
Date              0
Holiday           0
Discount          0
#Order            0
Sales             0
dtype: int64
```

Test

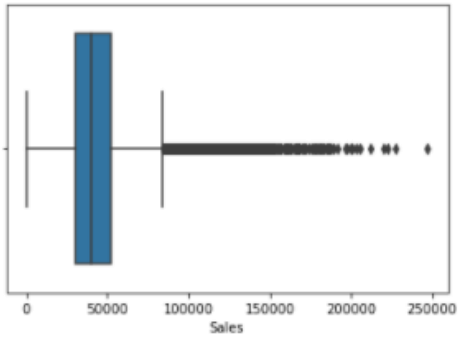
```
# 0 missing values
test.isnull().sum()
```

```
ID                0
Store_id          0
Store_Type        0
Location_Type     0
Region_Code       0
Date              0
Holiday           0
Discount          0
dtype: int64
```

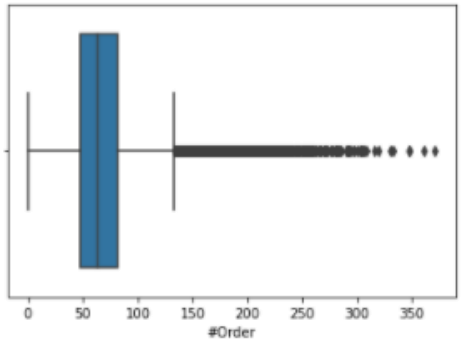
Handling Outliers

- Sales and Orders column contain outliers.
- 3.1% of Sales values are greater than upper whisker.
- 3.76% of Orders values are greater than upper whisker.
- Can't remove them since these data points reveal important information.
- Therefore, Trees and Ensemble techniques can be used for model building.

Sales



Orders



Check for Duplicated Rows

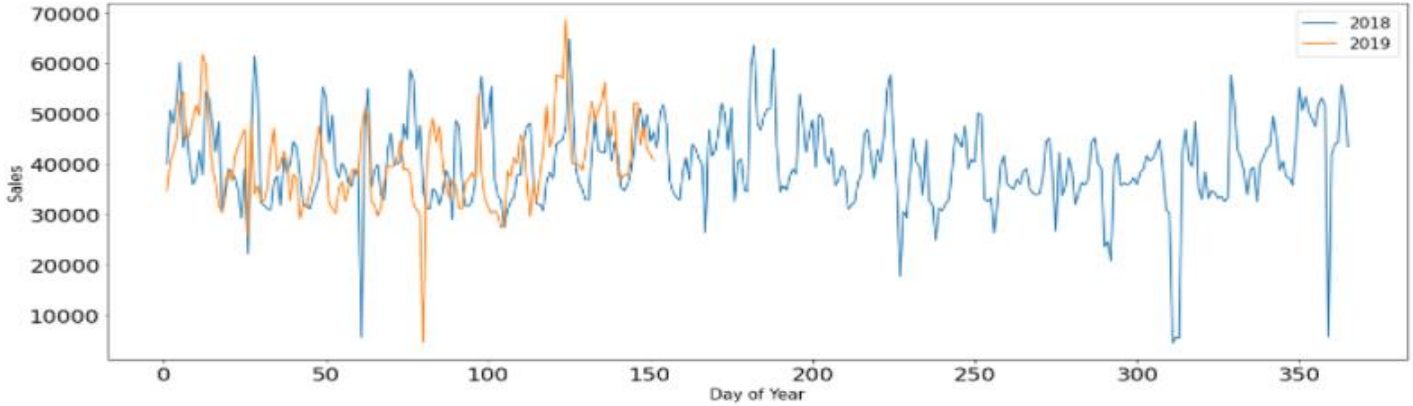
```
# 0 duplicates in train data.
train[train.duplicated()].shape[0]
```

```
0
```

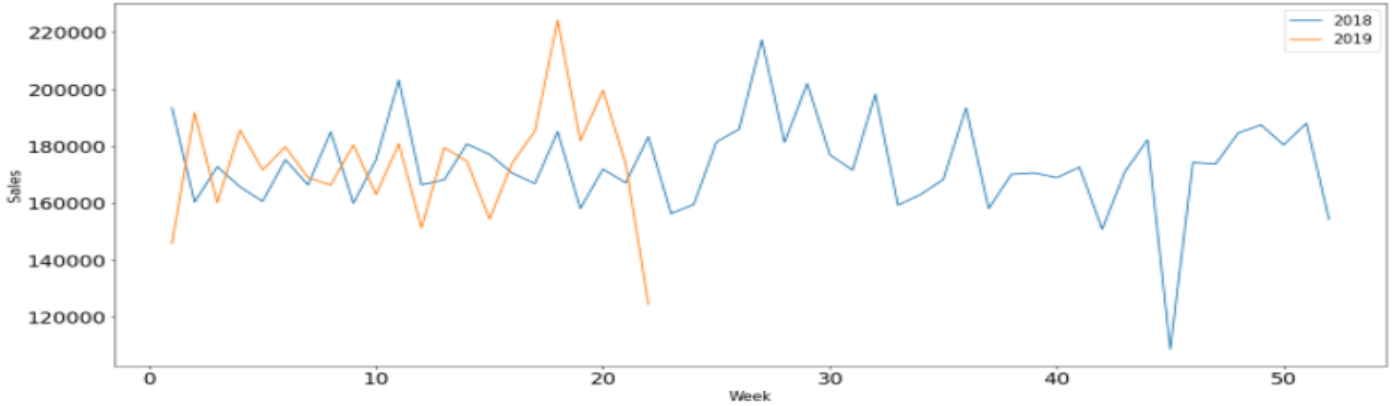
```
# 0 duplicates in test data.
test[test.duplicated()].shape[0]
```

```
0
```

Median Daily Sales



Median Weekly Sales



New Features

- Total Weekly Orders based on Store_type.
- Total weekends orders in every week of a month based on store_type.
- Total Weekly Orders based on Store_type & Location_Type.
- Total Weekend Orders based on Store_type & Location_Type.
- Total Weekly Orders by Store_id
- Total weekends orders in every week of a month based on Store_id.
- Avg Weekly Orders by Store_id
- Avg weekends orders in every week of a month based on Store_id.

store.

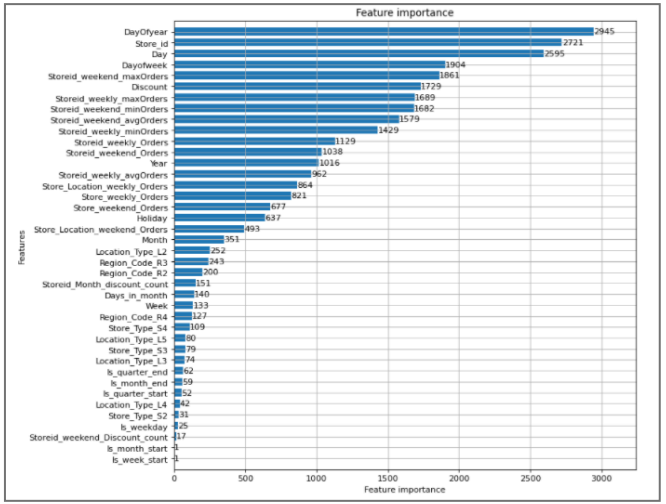
- Minimum Weekly Orders by Store_id
- Minimum weekends orders in every week of a month based on Store_id.
- Maximum Weekly Orders by Store_id
- Maximum weekends orders in every week of a month based on Store_id.
- Count of discounts offered by each store in every month
- Count of discounts on weekends of every week in a month for every store.

Feature Encoding

•One Hot Encoding for nominal categorical variables

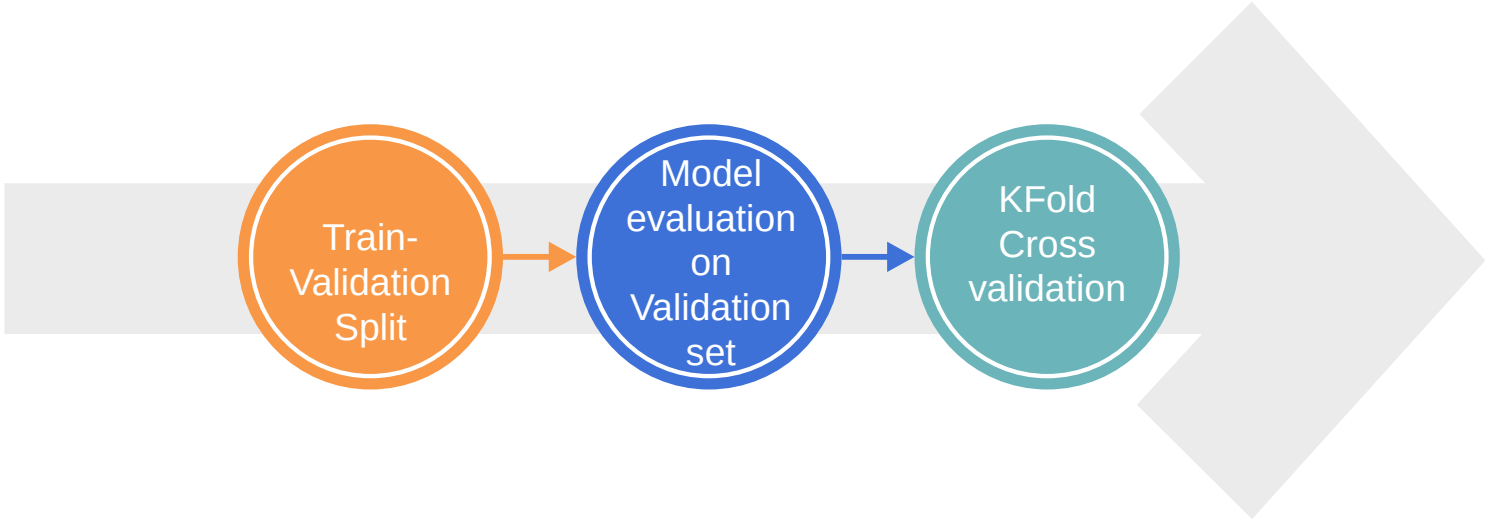
```
# One Hot Encoding
l = ['Store_Type','Location_Type','Region_Code']
tr_x = pd.get_dummies(tr_x, columns = l,drop_first=True)
val_x = pd.get_dummies(val_x, columns = l,drop_first=True)
test = pd.get_dummies(test, columns = l,drop_first=True)
```

Total Features after Feature Encoding & One Hot Encoding: 45



Min-Max Scaler for Numeric Columns

```
1  from sklearn.preprocessing import MinMaxScaler
2
3  scaler = MinMaxScaler()
4  scaler.fit(tr_x[numcols])
5  tr_x[numcols] = scaler.transform(tr_x[numcols])
6  val_x[numcols] = scaler.transform(val_x[numcols])
7  test[numcols] = scaler.transform(test[numcols])
```



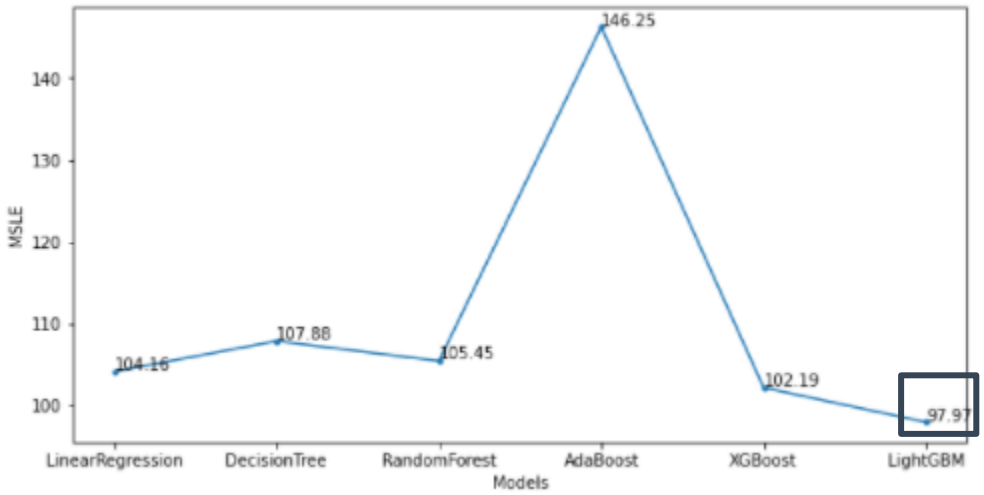
- Data Before 1st April 2019 : Train Set
- Data from 1st April 2019: Validation Set

Building models and tuning hyper parameters on train set and evaluating on validation set.

KFold cross validation on train set for final model evaluation.

Total Models Built: 6

Linear Regression | Decision Tree | Random Forest | AdaBoost | XgBoost | LightGBM



Best Performing model is LightGBM

	Models	MSLE
0	LinearRegression	104.159669
1	DecisionTree	107.875496
2	RandomForest	105.445846
3	AdaBoost	146.251692
4	XGBoost	102.185981
5	LightGBM	97.974466

-Being the lowest MSLE of 0.97, LightGBM Regressor model is selected among all other models.
- These scores are for the validation set and thus final scores are evaluated using Cross Validation.