Rossmann Store Sales Prediction
(Forecast sales using store, promotion, and competitor data)

MT2018002 ABHAY NANDA
MT2018128 UTKARSH AGRAWAL
MT2018104 SATYAM KHARE
TEAM NAME - MichaelTrevorAndFranklin

December 7, 2018

1 Introduction

Rossmann operates over 3,000 drug stores in 7 European countries. Currently, Rossmann store managers are tasked with predicting their daily sales for up to six weeks in advance. Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality. With thousands of individual managers predicting sales based on their unique circumstances, the accuracy of results can be quite varied.Challenge is to predict 6 weeks of daily sales for 1,115 stores located across Germany

2 Feature Engineering and Data Visualization

Following are the features that were present in the training set :

- **Id** - an Id that represents a (Store, Date) duple within the test set
- **Store** - a unique Id for each store
- **Sales** - the turnover for any given day (this is what you are predicting)
- **DayOfWeek** - a number representing which day of the week it is
- **Date** - date of the transaction
- **Customers** - the number of customers on a given day
- **Open** - an indicator for whether the store was open: 0 = closed, 1 = open
- **StateHoliday** - indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None
- **SchoolHoliday** - indicates if the (Store, Date) was affected by the closure of public schools

- **StoreType** - differentiates between 4 different store models: a, b, c, d
- **Assortment** - describes an assortment level: a = basic, b = extra, c = extended
- **CompetitionDistance** - distance in meters to the nearest competitor store
- **CompetitionOpenSince[Month/Year]** - gives the approximate year and month of the time the nearest competitor was opened
- **Promo** - indicates whether a store is running a promo on that day
- **Promo2** - Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating
- **Promo2Since[Year/Week]** - describes the year and calendar week when the store started participating in Promo2
- **PromoInterval** - describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew. E.g. "Feb,May,Aug, Nov" means each round starts in February, May, August, November of any given year for that store

Following features were **extracted** from the above available features :

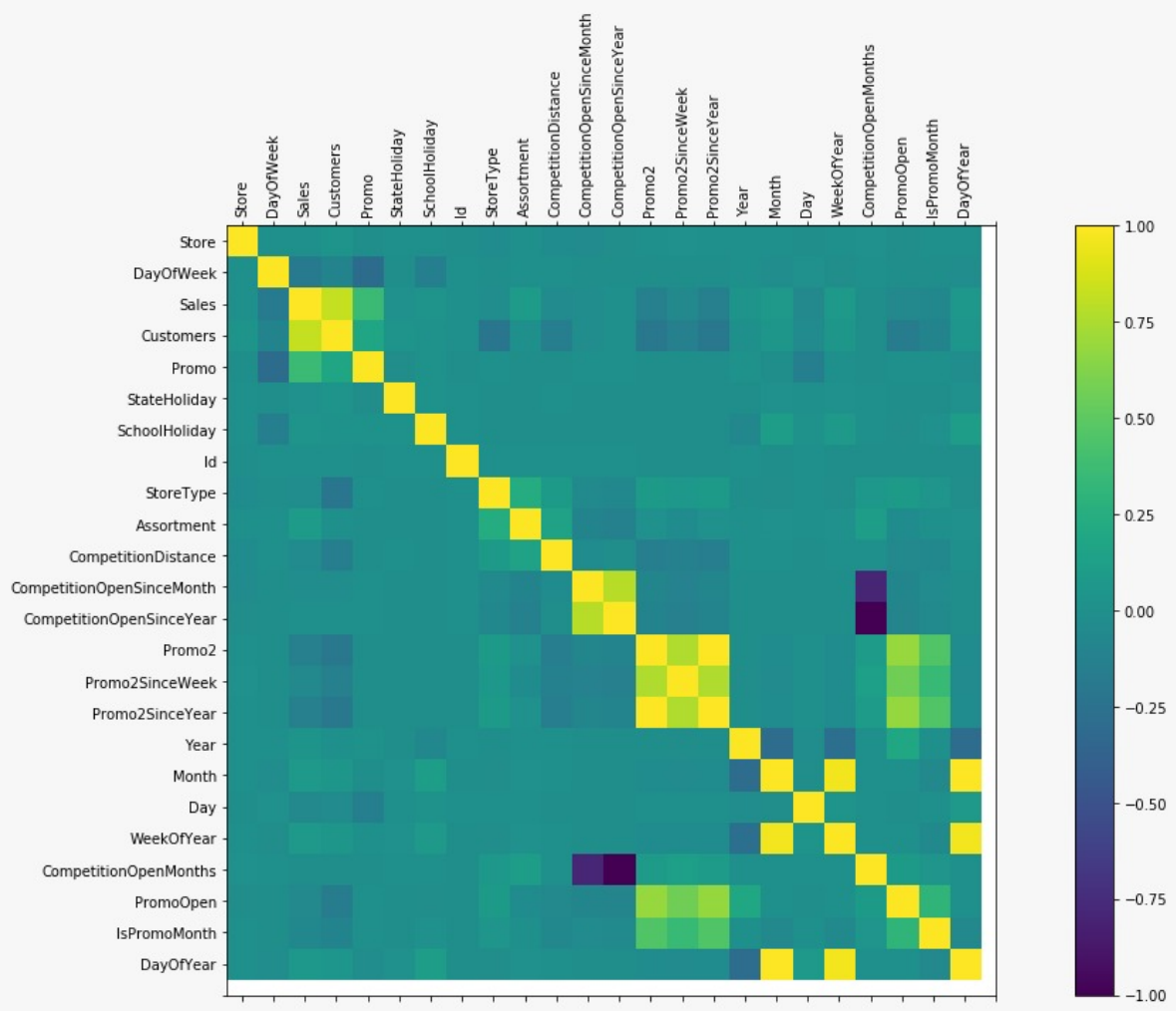
- **Month**, **Day** and **Week** is extracted from the transaction *date*.
- **CompetitionOpen** is engineered from the two features *CompetionOpenSinceMonth* and *CompetitionOpenSinceYear* and represents the number of months passed since the opening of rival store.
- **PromoOpen** is engineered from the two features *Promo2SinceWeek* and *Promo2SinceYear* and represents the number of weeks passed since promo2 is valid.
- **IsPromoMonth** is engineered from *PromoInterval* and represents whether the promo2 was valid in the month in which transaction took place.
- **DayOfYear** represents the number of the day (out of 365) on which transaction occurred.
- **WeekOfYear** represents the number of week (out of 52) in which the transaction took place.

The following features were ***LabelEncoded*** thereby converting the categories into their numerical equivalent :

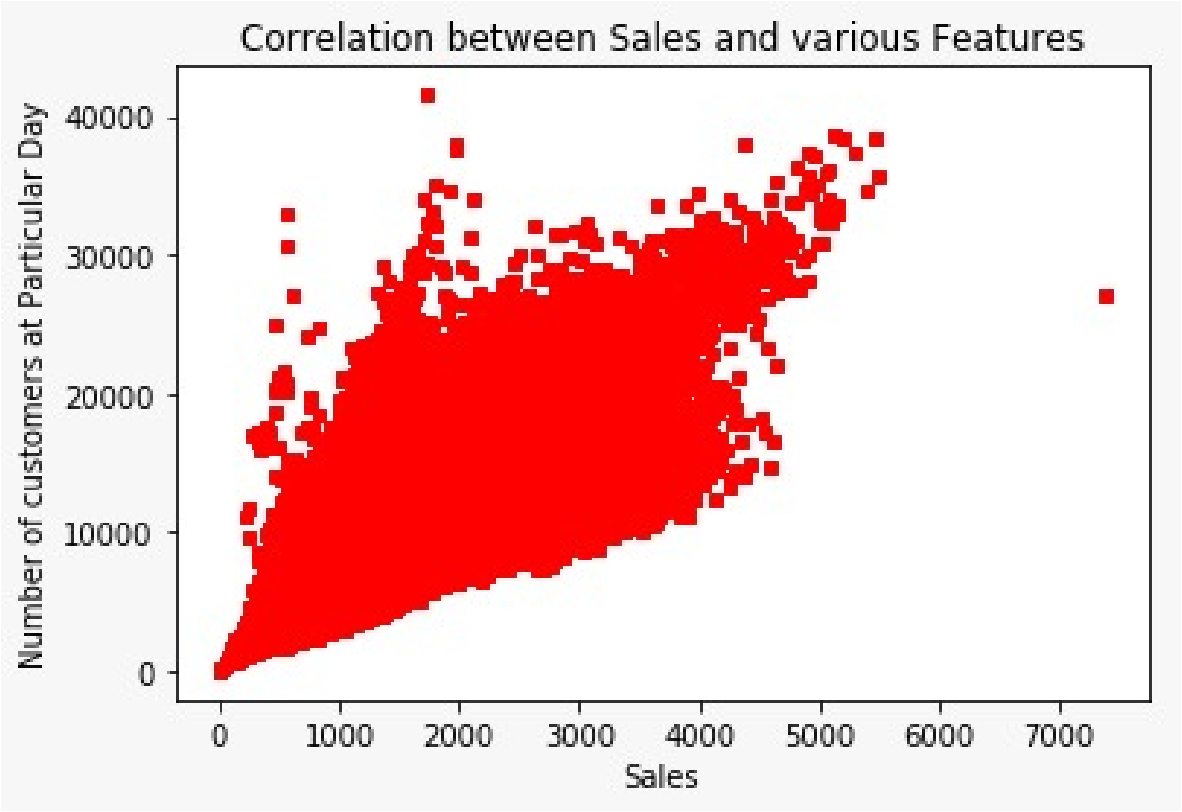
- StoreType
- Assortment

RMSE is used to measure the validation score

Correlation Matrix



Correlation Matrix between all pairs of features



The target variable Sales has the most correlation with feature Customer

3 Different Model Experimentations

3.1 HyperParameter Tuning

Hyperparameter tuning relies more on experimental results than theory, and thus the best method to determine the optimal settings is to try many different combinations evaluate the performance of each model.
However, evaluating each model only on the training set can lead to one of the most fundamental problems in machine learning : overfitting.
The standard procedure for hyperparameter optimization accounts for overfitting through cross validation.
KFold cross-validation techniques has been used over every model trained.Implementation of kFold cross validation is taken from sklearn sklearn.model_selection.KFold.Number of folds given as the parameter is 5.

3.2 Features Used

Features provided in the training set were : Store, DayOfWeek, Date, Sales, Customers, Open, Promo, StateHoliday, SchoolHoliday, Id
Feature related to the store given were : Store, StoreType, Assortment, CompetitionDistance, CompetitionOpenSinceMonth, CompetitionOpenSinceYear, Promo2, Promo2SinceWeek, Promo2SinceYear, PromoInterval
The features added and data cleaning part is described in the feature engineering section
The final features used in the model are : Store, Customers, Promo, StoreType, Assortment, CompetitionOpenMonths, CompetitionDistance, DayOfWeek, Promo2, PromoOpen

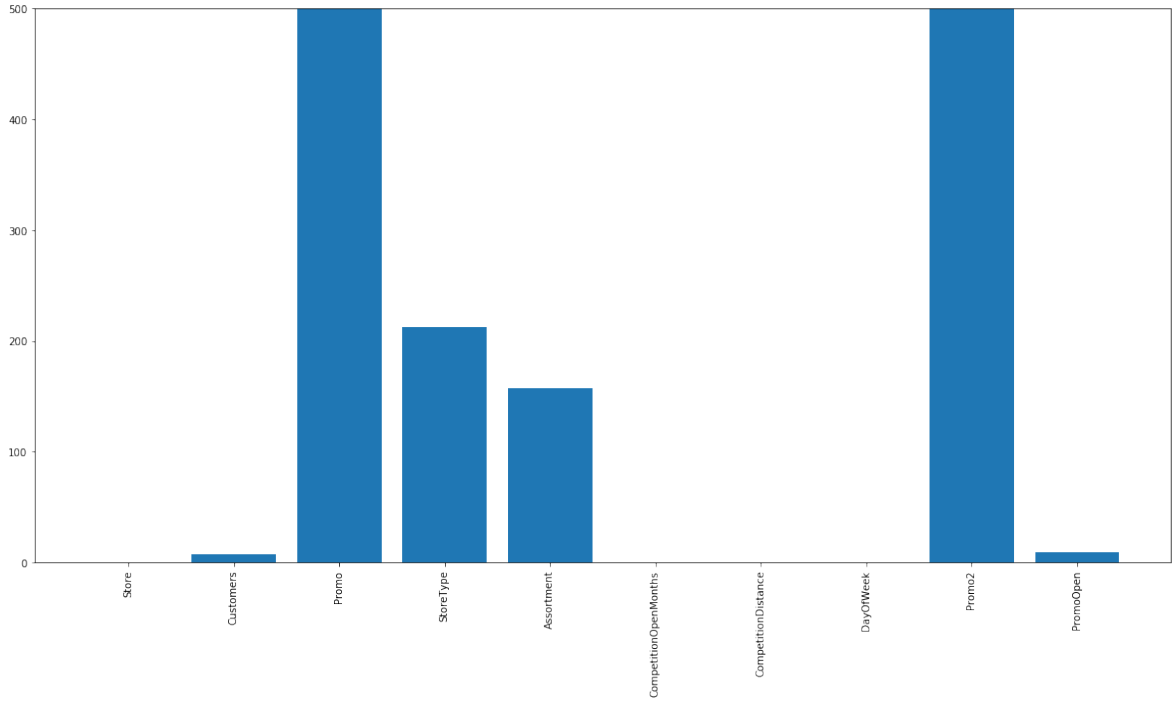
3.3 Random Search Cross Validation in Scikit-Learn

Usually, we only have a vague idea of the best hyperparameters and thus the best approach to narrow our search is to evaluate a wide range of values for each hyperparameter.
Using Scikit-Learns RandomizedSearchCV method, we can define a grid of hyperparameter ranges, and randomly sample from the grid, performing **K-Fold CV** with each combination of values.To determine if random search yielded a better model, we compare the base model with the best random search model.Random search returns the model and hyperparameter values of the best model.

3.4 Models Trained

3.4.1 Linear Regression

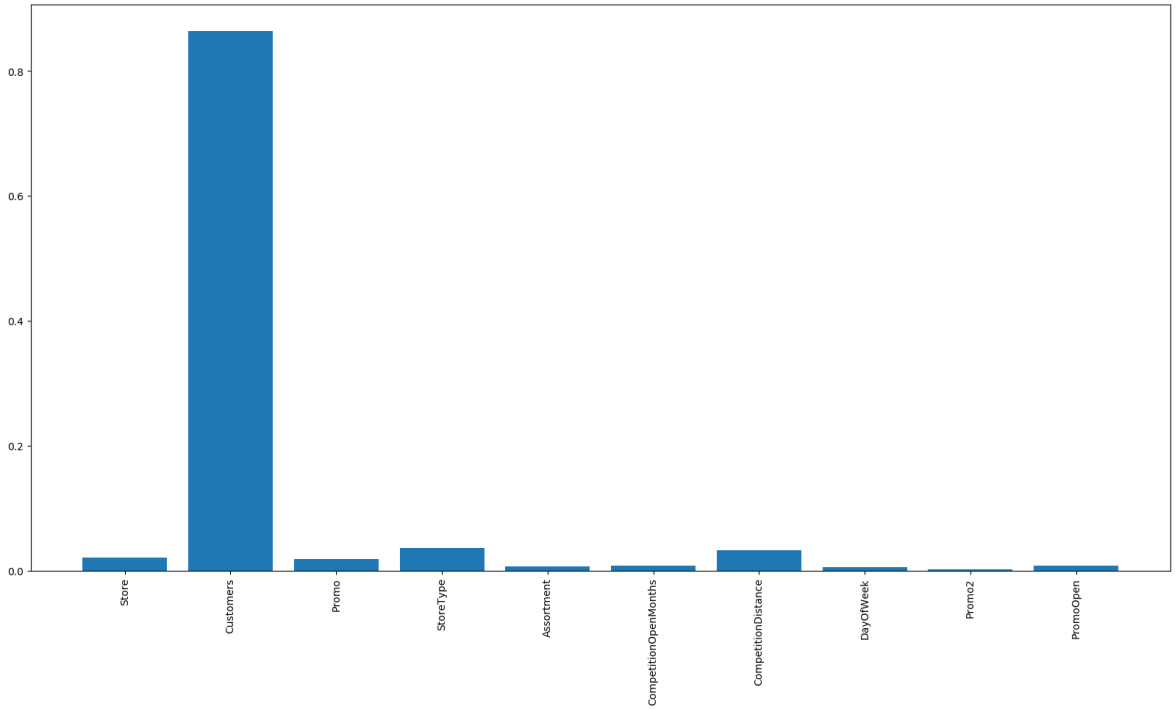
Description
Linear regressionis a model that checks the linear relationship between various variablesthe training variable(s) and the predictions. It tries to fit the model on various features which are used in the dataset and the feature to be predicted. The model trained will now be used to predict the values of the test data.
Score On Kaggle
Private Score : 1.28285
Public Score : 538.49333
Validation Score
1504.076
HyperParameters
copy_X=False,fit_intercept=True,normalize=True



Feature Importance - LinearRegression

3.4.2 Decision Tree

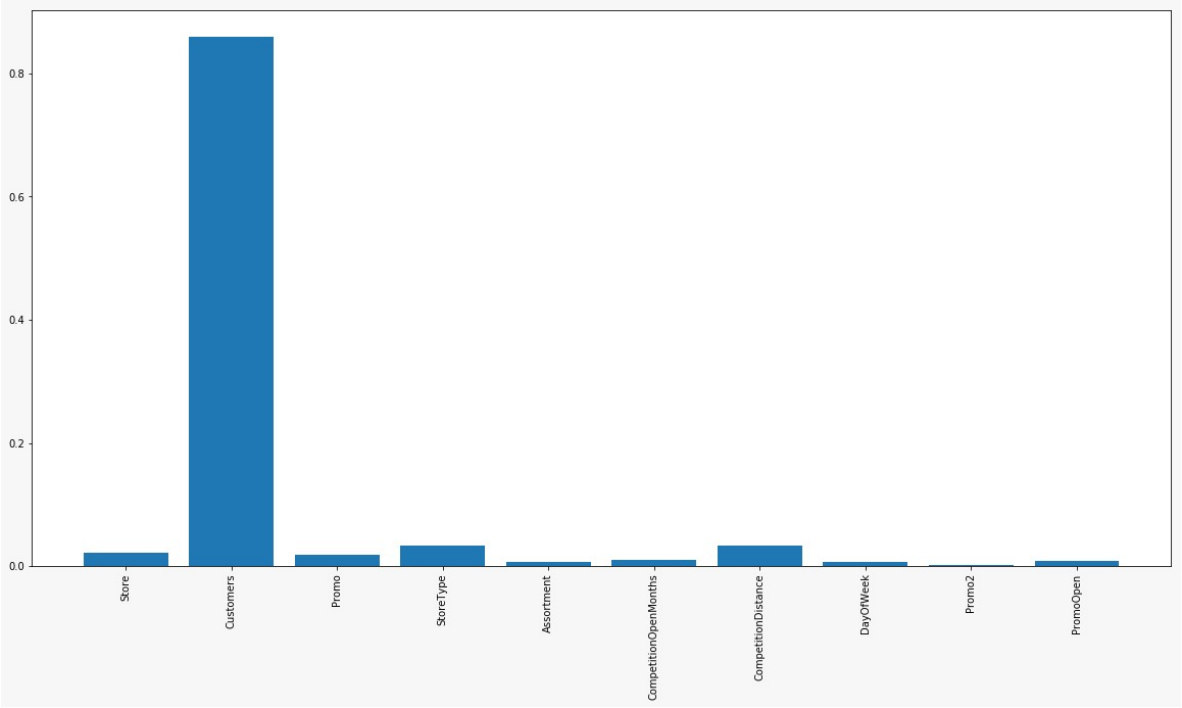
Description
Decision tree builds regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.
Score On Kaggle
Private Score : 0.20231
Public Score : 73.79280
Validation Score
676.469
HyperParameters
max_depth = 20



Feature Importance - DecisionTreeRegressor

3.4.3 Random Forest Regression

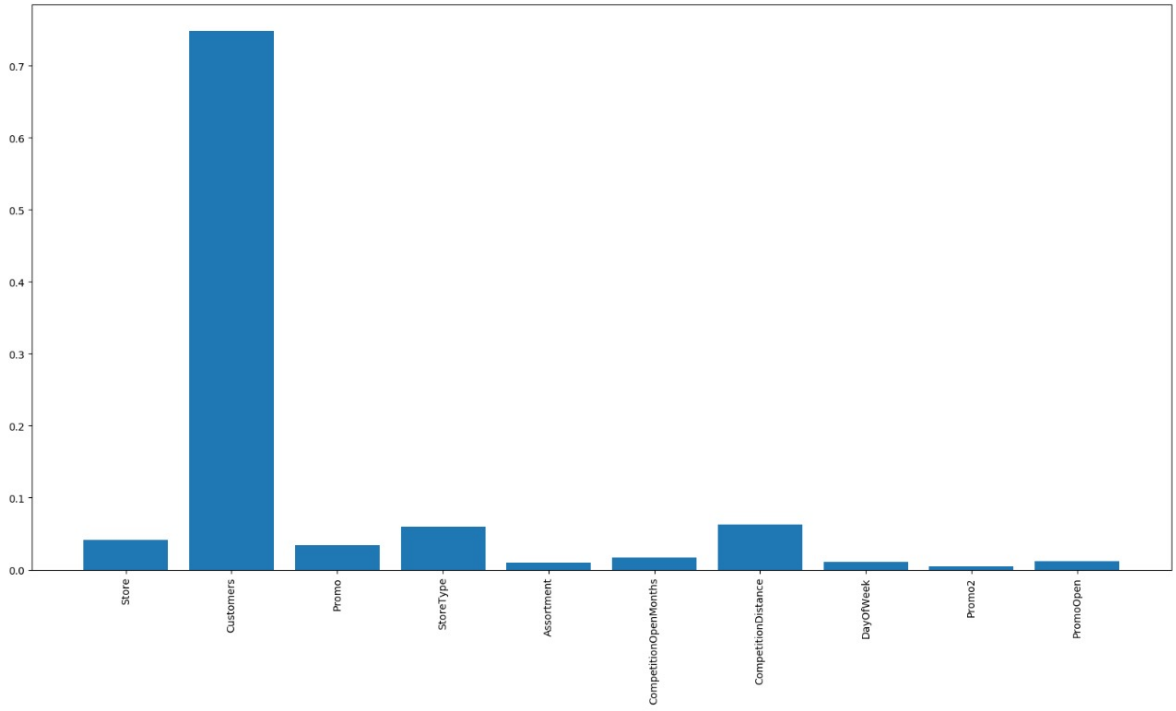
Description
The random forest model is a type of additive model or ensemble that makes predictions by combining decisions from a sequence of base models. Model was trained using Random Forest Regressor,with KFold cross validation with number of splits used as 5.
Score On Kaggle
Private Score:0.25245
Public Score:1.25831
Validation Score
746.47
HyperParameters
n_estimators=warn, criterion=mse, max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=auto, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False



Feature Importance - RandomForestRegressor1

3.4.4 Random Forest Regression HyperParametersTuned

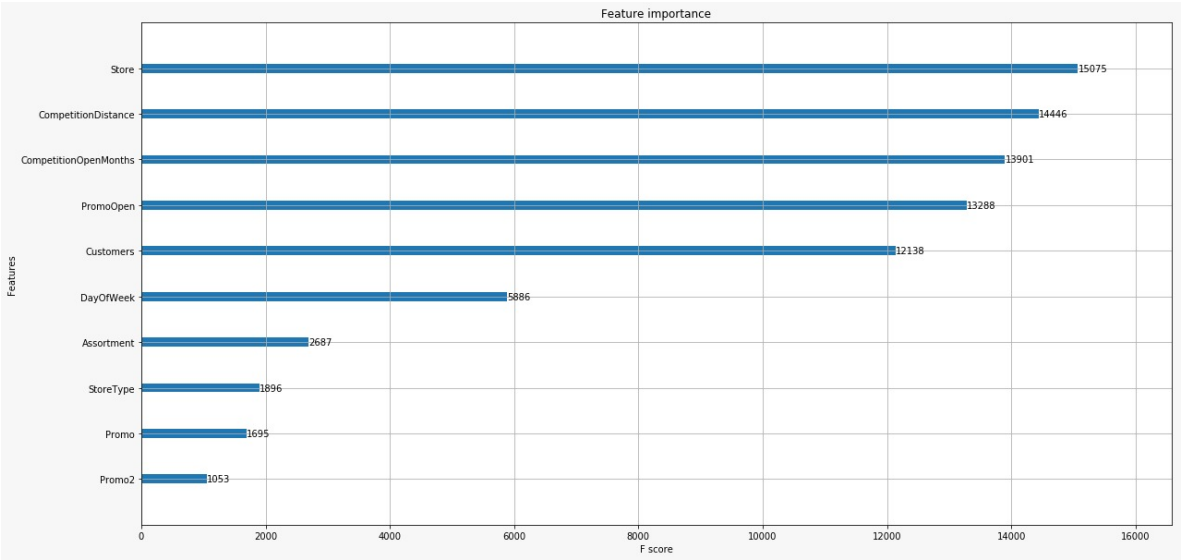
Score On Kaggle
Private Score:0.0825
Public Score:0.275
Validation Score
381.26
HyperParameters
max_depth=50,max_features='auto',min_samples_leaf=4,min_samples_split=10,n_estimators=400



Feature Importance - RandomForestRegressor2

3.4.5 XGBoost Regression

Description
XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.
Score On Kaggle
Private Score : 0.05564
Public Score : 57.84954
Validation Score
1101.826056110147
HyperParameters
objective = reg:linear, booster = gbtree, eta = 0.03, max_depth = 10, subsample = 0.9, colsample_bytree = 0.7, silent = 1, seed = 10, num_boost_round = 1000



Feature Importance - XGBoostRegressor