Time-series forecasting of retail products sales using Machine Learning

Vartika Tewari

Abstract

Accurate Time series prediction has various real applications. In this project we tackle a similar problem provided by Kaggle, as part of their ongoing "Predict Future sales" competition [4]. We handle challenging time series data of daily sales from the largest Russian software firms- 1C Company. Given past data the goal is to predict monthly sales for every possible shop item pair, and make the model robust to the seasonal changes and trends. We employ and compare three models, Prophet, random forest and xgboost for prediction, and evaluate them on Root Mean Squared Error(RMSE). We obtain the best peformance with xgboost as 0.85 RMSE on the test set where the target range is [0-20].

1 Introduction

Forecasting future sales has been an important problem in all organizations. These help in making decisions on budgeting, prospecting, and other revenue-impacting factors.[5] Having accurate estimation of any organizations growth and changes helps in preparing strategies for short-term demand and supply and identification of areas for investment.

There are a lot of factors which can impact sales forecast like, market changes, product changes as well as seasonality. Robustness of the model to such changes is an essential part of the project.

In this project we work with a challenging time-series dataset, from an ongoing Kaggle challenge "Predicting Future Sales" [4] consisting of daily sales data, provided by one of the largest Russian software firms - 1C Company. The data consists of shop and product changes each month. The problem statement is to accurately predict total sales for every product and store in the next month. That is, to predict monthly sales for every possible shop item pair.

2 Data

2.1 Dataset Description

The data-set consists of daily sales data. There are about 3 million such records in the training set, collected over 60 shops selling 20,000 unique items.

Training data consists of records with information that a particular item had been sold in a particular shop, in a particular day, in the training period. The training period is about one and a half year. Testing period is the month that falls on training period. The data-set also has missing data as not every item is sold in the above time period.

Dataset	Time series
Min Date	01.01.2013
Max Date	31.12.2014
#Items	22170
#Shops	60
#datapoints	2935849

Table 1: Dataset Statistics

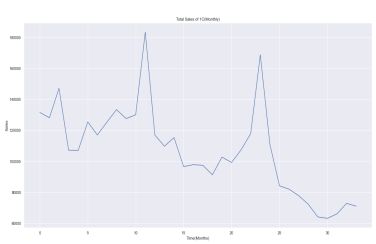
2.2 Data Cleaning and pre-processing

We removed the negatively priced items from the dataset. We also dropped the text features completely as they were in Russian.

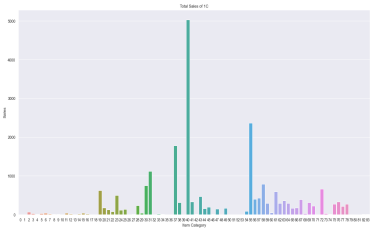
We replaced missing product count with 0 We removed outliers by clipping item count to be less than or equal to 20.

2.3 Exploratory Data Analysis

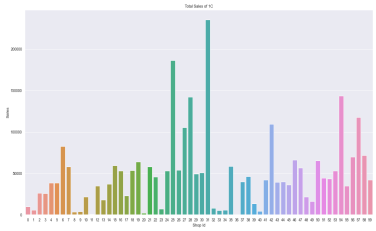
We did some extensive data analysis to understand the data which helped in feature engineering. We did the exploration on two levels: 1) The Whole data 2) Summed Yearly



(a) Total Sales on Whole Data

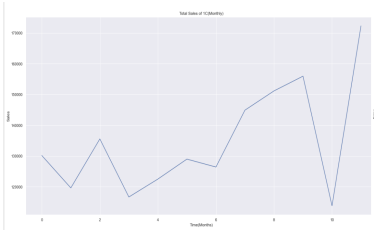


(b) Sum of sales over Categories on Whole Data

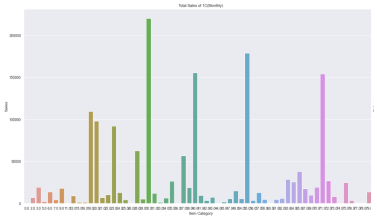


(c) Sum of sales over Shops on Whole Data

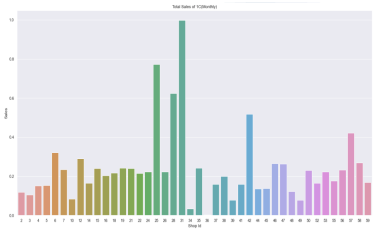
Figure 1: Aggregate Analysis on Whole Data



(a) Total Sales on Monthly Data



(b) Sum of sales over Categories on Monthly Data



(c) Sum of sales over Shops on Monthly Data

Figure 2: Aggregate Analysis on Monthly Level Data

Key observations:

- The sales rise at the end of the year, most probably due to holiday season.
- There are few categories which always sell more.
- There are few shops which sell more, probably because they are bigger shops which carry more items.

2.4 Feature Engineering

We created various temporal as well as non-temporal features.

- Item count features: min, max count, item counts monthly etc.
- Price Features: historical price increase/decrease, Max, Min price, current price etc.
- Lag Features: Previous 3 months features to get trend
- Encoded features: Mean encoding of item id, shop id
- Seasonality features: Number of days in the month, Month, Year

3 Technical Approach

We tried out three models for our regression problem: We selected these models so as to compare the traditional time-series prediction approach with Machine learning.

3.1 Auto Regressive Model

As AR models are the go to models for time series prediction, we used Prophet [6] to form the baseline.

As out task required prediction at shop,item level . This led to having too many series to predict. (22k+ items, 60 shops)

So, we used the Middle out approach [1] predicted at shop level and weighted them based on item counts of previous month.

3.1.1 Prophet[6]

Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

Model used can be defined as : $y(t) = g(t) + s(t) + h(t) + e(t)$
where:
g(t): trend models non-periodic changes
s(t): seasonality presents periodic changes
h(t): irregular changes like holidays
e(t): covers other changes not accommodated by the model

3.2 Random Forest[2]

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.

3.3 Extreme Gradient Boosting

We use extreme gradient boosting technique like XGBoost [3], because it has advantages over general gradient boosting in terms of providing regularization through a combination of both the ridge and lasso regressions. It also handles different type of sparsity patterns in the data efficiently. We ran Grid Search to train Hyperparameters.

Best Parameters Obtained:
(base_score=0.5, booster='gbtree', colsample_bylevel=1, colsample_bynode=1, colsample_bytree=0.7, eta=0.3, gamma=0, gpu_id=-1, importance_type='gain', learning_rate=0.300000012, max_delta$_s$$tep = 0, max_depth = 8, min_c hild_weight = 1000, n_estimators = 500, num_parallel_t ree = 1, random_state = 0, reg_alpha = 0, reg_lambda = 1, scale_pos_w eight = 1, seed = 0, subsample = 0.7, tree_method =' exact', validate_parameters = 1$)

4 Results

We get the best RMSE by xgboost model, followed closely by Random Forest and then Prophet. All the results are listed n Table 2. We also calulate the feature importance as listed in Table3. It can be seen that the mean encoded features as well as the features defining the time affect the results to a large extent.

5 Conclusion and Future Work

We tried various different models, and xgboost performed the best in our case. This can be attributed to extensive hyperparameter tuning. Our models did not overfit much due to regularisation parameters set in xgboost. We

Model	Train(RMSE)	Validation(RMSE)	Test(RMSE)
Xgboost	0.6712	0.7971	0.8591
RandomForest	0.6544	0.8152	0.8854
Prophet	-	-	1.855

Table 2: RMSE values for different models

Features	Fscore
meanitem	201
shopmeanitem	189
year	119
pricedecrease	108
monthm	96
monthmean	69
trenditem	66

Table 3: Best features by feature importance in xgboost

saw that feature engineering plays the most crucial role when it comes to tackling such problems. Also optimal hyperparameter selection also plays a key role.

As xgboost takes a long time to run in future, we would like to try running the same model with only the important features and see how much does that affect performance. Also we would like to compare the performance with a deep learning model like LSTM.

References

[1] George Athanasopoulos, Roman A. Ahmed, and Rob J. Hyndman. "Hierarchical forecasts for Australian domestic tourism". In: *International Journal of Forecasting* 25.1 (2009), pp. 146–166. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2008.07.004. URL: http://www.sciencedirect.com/science/article/pii/S0169207008000691.

[2] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[3] Tianqi Chen and Carlos Guestrin. "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794.

[4] *Kaggle-Challenge:Predict Future Sales*. URL: https://www.kaggle.com/c/competitive-data-science-predict-future-sales/overview.

[5] *Sales Forecasting Blog*. published May 14, 2020, updated May 14 2020. URL: https://blog.hubspot.com/sales/sales-forecasting.

[6] Sean J. Taylor and Benjamin Letham. "Forecasting at Scale". In: *The American Statistician* 72.1 (2018), pp. 37–45. DOI: 10.1080/00031305.2017.1380080. eprint: https://doi.org/10.1080/00031305.2017.1380080. URL: https://doi.org/10.1080/00031305.2017.1380080.