Future Sales Prediction

ATAKAN FILGOZ ENES KARANFIL BURAK BARAN OMER SEKER

Abstract

Nowadays, one of the largest goals of craftsmen is to predict the sales for the future that they get from the products they sell. There is a considerable number of factors which affect the revenue. These factors can be guessed by reviewing the characteristics of daily historical data. Prediction of sales of the craftsmen can be seen as a machine learning problem. Therefore, in this study we try to predict the future sales of products based on their properties.

1. Introduction

Owing to the advancing technology, methods such as machine learning and deep learning are used in solving big problems in our present time. The most important criteria in efficiency of these methods and obtaining results is having enough data on the problem. Recently, data collection and processing the collected data in all fields used for large problems. One of these fields is the commerce industry. This article consists of the problem in commerce data. In order for a shop to generate high revenue, a shop has to meet a number of criterion. The data on the subject and grip of a shops content, the current price of items,the number of products sold in the past are some of the criterion determinative of future sales. The data provided in this project will help to estimate the future sales of the shops. Carrying out works and studies on commerce data and carrying out analysis to estimate future sales and other content are particularly important information for craftsmen. The most important reasoning is that such analysis sets forth financial risks in advance and provides the opportunity to operate dependently. With a credible estimation, necessary measures may be taken by the craftsmen in the early stages of receiving items and financial risks may be reduced with improvements. Especially, it is important for craftsmen to solve this problem to make provision against fluctuating incomes. Thus, tradesmen can determine their living standards by having this knowledge.

Studies on this subject using different data sets are carried out. CNN algorithms are used in future sales predictions.

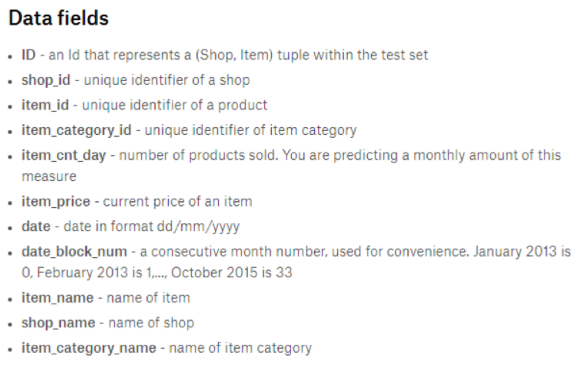
2. Methods

2.1. Dataset

The data set used in the project, is taken from a competition from Kaggle. In this section, details about the data set will be shared. The training data set consists of 2.935.849 samples and 10 columns which consist of data item id which represents unique identifier of a product, shop id for unique identifier of a shop, ID that related with shop-item tuple within the test set, item category id,item count day is number of product that they sold, item price current price of an item, date, date block num is used for a consecutive month number that used for convenience, item name, shop name, item category name. The test data has 214.201 samples. After examining the data set, we have noticed that data set consists of time series. Sales over time of each store-item is a time-series itself. Thus, it is foreseen that it may cause difficulties in the future.

2.2. Data Preprocessing

We need to preprocess the data before we start developing models to predict future sales. At first, observed the number of missing values in the data. There was not any missing values in the data. All of the data held in a single CSV. Data has attributes which can be seen in Figure-1 and given without any order.

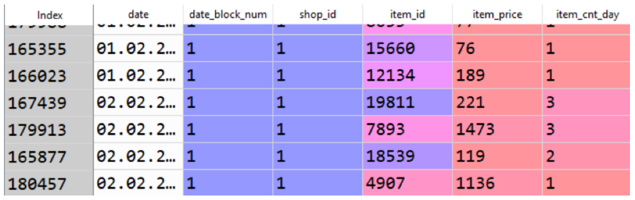


Data fields

- **ID** - an Id that represents a (Shop, Item) tuple within the test set
- **shop_id** - unique identifier of a shop
- **item_id** - unique identifier of a product
- **item_category_id** - unique identifier of item category
- **item_cnt_day** - number of products sold. You are predicting a monthly amount of this measure
- **item_price** - current price of an item
- **date** - date in format dd/mm/yyyy
- **date_block_num** - a consecutive month number, used for convenience. January 2013 is 0, February 2013 is 1,..., October 2015 is 33
- **item_name** - name of item
- **shop_name** - name of shop
- **item_category_name** - name of item category

Figure 1

For an easier understanding, we divided the data in two different ways and sorted by date. First, the data is divided according to shop id. We get 59 different data set for 59 different shop. And the second one is according to item categories. We split data in item categories since there were lots of unique items, more than 20000, so using item categories for creating new data sets was more sensible. A part of sorted data set for first shop can be seen in Figure-2.



Index	date	date_block_num	shop_id	item_id	item_price	item_cnt_day
165355	01.02.2…	1	1	15660	76	1
166023	01.02.2…	1	1	12134	189	1
167439	02.02.2…	1	1	19811	221	3
179913	02.02.2…	1	1	7893	1473	3
165877	02.02.2…	1	1	18539	119	2
180457	02.02.2…	1	1	4907	1136	1

Figure 2

2.3. Feature Engineering

First, we will refer to the preprocess steps we have applied to the attributes which are held in csv format.

date block number: Month of transaction is kept. A consecutive month number, used for convenience. January 2013 is 0, February 2013 is 1...

shop id: Unique identifier of a shop.

item id: Unique identifier of a product.

item category id: Unique identifier of product category.

item count day: Number of products sold in a day. We predict a monthly amount of this feature.

item count month: Number of products sold in a month. We predict this feature. To get this feature, we used item count day, and date block number as an index.

item price: Current price of an item.

date: Since the attribute is unique for all same shop id and item id pairs. Thus, the attribute is dropped.

shop name: In the attribute the name of shop is kept. After extracting new features, the attribute is dropped.

item name: In the attribute the name of item is kept. After extracting new features, the attribute is dropped.

city code: The feature is extracted from shop name and it contains unique city id where the shop locates.

shop type: The feature is extracted from shop name and it contains unique shop id which shows the type of shops. For instance, 2 is given as shop type id to shopping center.

item type code: The attribute is extracted from item name and it contains type of the item like accessories, game consoles and others. Performed Label Encoding on this attribute.

item subtype code: The attribute is extracted from item name and it contains secondary type of the item. For instance, PlayStation2 is a sub type of the Games which is an item type.

shop mean: The attribute contains the average monthly sales of a shop.

shop item mean: The attribute contains the average monthly sales of an item by a specific shop.

date/item average item count: The attribute contains the previous month's average item count by item quantity. It is normalized version of total item sales in a month in all shops.

item count month lag: The attribute contains the previous months' average sales of a specific item.

item price lag: The feature contains the previous months' average item prices of a shop.

item price by shop: The feature contains the average price of items of a shop.

2.4. Feature Importance

As a result of the experiments done with these features, some additional features which are thought to be effective on the sales were added.

It was decided to use feature importance techniques to reach more concrete results about the attributes and their relationship with the target values. After the preprocessing and feature engineering steps in the data set, feature importance techniques were applied.

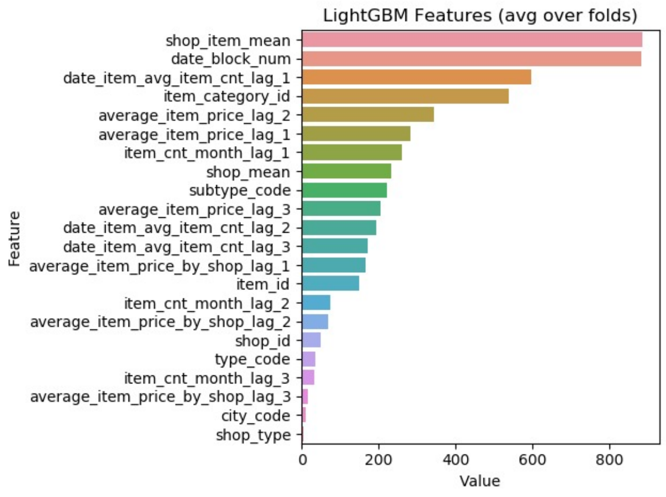


Figure 3

2.5. Data Analysis

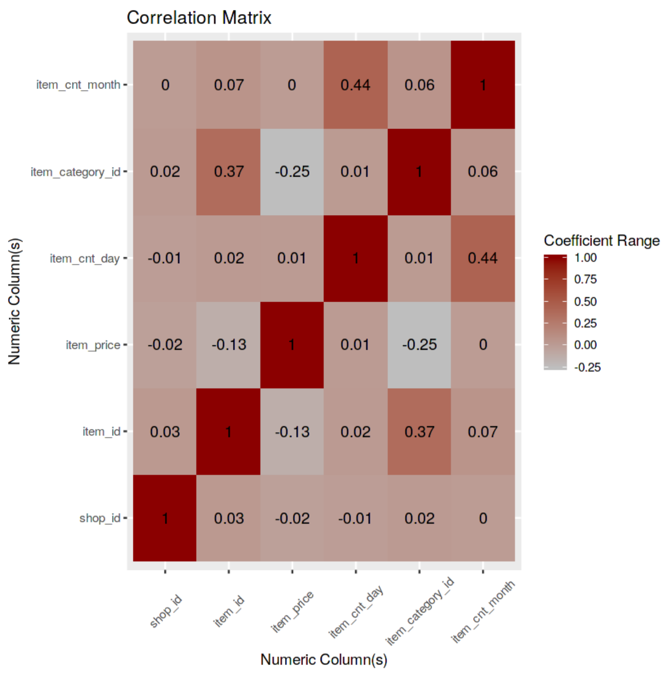


Figure 4

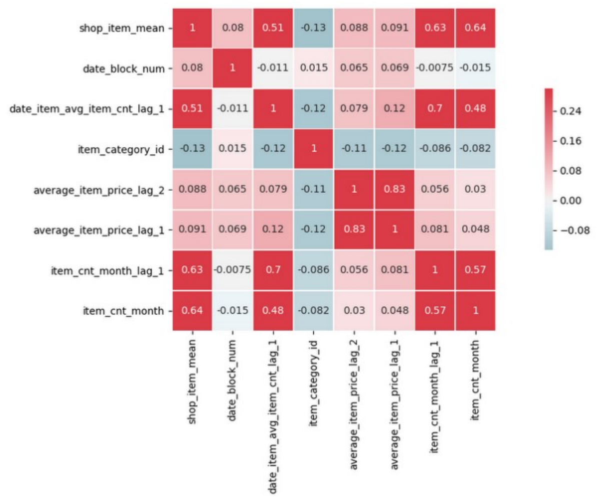


Figure 5

As a result of the experiments done with these features, some additional features which are thought to be effective on the sales were added.

It was decided to use feature importance techniques to reach more concrete results about the attributes and their relationship with the target values. After the preprocessing and feature engineering steps in the data set, feature importance techniques were applied.

In figure 4, we produced correlation matrix with six features and we analyze the relation between them. Thus, we can extract some information for revising our dataset. We can conclude some results with basing correlation matrix. Firstly, there is no relation between shop id and item count month features in our dataset so we can briefly say that shop id is not a key feature for item count mounth.

We want features which we can correlate with other feature that we estimate because we cannot make a substantial assumptions by using uncorrelated features.

In figure 5, we have represented correletion matrix that after we have done pre-processing steps. As seen in correlation matrix in figure 5, we have correlated features which we are using them finding fulfilling assumption.

3. Models

We compare 3 different methods which are Random Forest Regressor, XGBBoost, Light GBM, Decision Tree Regressor. In the Kaggle error rate chosen as root mean square error therefore we use root mean square error for our loss function to evaluate our system. To obtain for better results we try powerful ensemble models. Methods that we have tried so far we get the highest score with XGBoost with 0.93 error rate. Respectively, it follows Random Forest with 0.96 error rate. In this section of paper, we are going to introduce several models that we use in this data science problem.

XGBoost: Xgboost is a scalable machine learning system for tree boosting. The most important factor behind the success of XGBoost is its scalability in all scenarios. Xgboost is short for eXtreme Gradient Boosting package. It is an efficient and scalable implementation of gradient boosting framework. It supports various objective functions, including regression, classification and ranking.

Random Forest Regressor: Random forests is an ensemble machine learning method, which is operated by constructing a multitude of decision trees at training time and outputting the class that is averaged or voted by every individual tree. The random forest model is a type of additive model that makes predictions by combining decisions from a sequence of base models. Random forest model is very good at handling tabular data with numerical features, or categorical features with fewer than hundreds of categories.

Light GBM: LightGBM is a gradient boosting framework that uses tree based learning algorithms. It has several advantages; for instance, faster training speed and higher efficiency and capable of handling large-scale data. LightGBM is a highly efficient gradient boosting decision tree which is consisted from Gradient-based One-Side Sampling and Exclusive Feature Bundling algorithms.

Decision Tree Regressor: Decision tree builds regression or classification 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. A decision node has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data. We used the Decision Tree Regressor, because of numerical data.

4. Experimental Results

4.1. Settings

In this study Python 3 is used as main programming language and developed in Jupyter Notebook. Pandas and Numpy libraries are used for reading and analyzing the data. Matplotlib and Seaborn libraries were used for visualizing the analysis results. Sklearn library is used for training the models.

For calculating the quality of predictions, used root mean squared error, which is the standard deviation of the residuals. Residuals are the measure of the distance between the regression line and data points.

$$RMSE = \sqrt{(\hat{y} - y)^2}$$

where $\hat{y}$ represents the prediction of model and y is the ground truth value.

At first we experimented our modeling with given dataset and get a rmse score more than 2 for all models. After taking these results, some changes were made to the data set to achieve more accurate results. To make more accurate prediction adding more detailed attributes caused overfitting so we tried to keep our features as simple as they can be.

Firstly new features created from existing features and categorical values such as shop name, item name were vectorized by using label encoding. The models which were mentioned in Models section were tried again on the new data. Wtih the new data, we obtained better rmse values on all models. We obtained rmse values of the LightGBM, XGBoost, Random Forest, Decision Tree respectively 1.05, 0.93, 0.96, 1.22.

5. Conclusion

In this study, different models are trained with 3 million transaction samples and predicted the total sales for every product and store. First of all, features in the data set are observed. With the complex data collected on the transactions, the estimation of the store and product sales was done by machine learning algorithms. Before machine learning algorithms were run on the data set, it seems to be data set has a number of invalid values, and data set has been rendered completely meaningful with preprocessing. With the preprocessed data set, machine learning algorithms were run and the results were examined carefully. After these observations the data set was updated with new features. Machine learning algorithms (XGB, Random Forest Regressor, LGB, Decision Tree Regressor) results made it possible to predict the sales of the stores and products. These results are not sufficient for real life. In the future, the data sets will be larger and more complex and machine learning algorithms may improve significantly. These improvements will make it possible to obtain more accurate predictions.

Table 1. Regressing error rate for our four ensemble models that we present for -1C Company dataset provided by Kaggle.

DATA SET	RMSE
DECISION TREE	1.22
XGBOOST	0.93
LIGHT GBM	1.05
RANDOM FOREST	0.96

6. References

[1] https://www.coursera.org/learn/competitive-data-science

[2] https://www.kaggle.com/jagangupta/time-series-basics-exploring-traditional-ts

[3] https://www.kaggle.com/kyakovlev/1st-place-solution-part-1-hands-on-data

[4] https://www.kaggle.com/dimitreoliveira/deep-learning-for-time-series-forecasting