1 Introduction

The basic business model for retail stores is to stock the stores with merchandise for customers to make purchases at the locations. It is an important task for stores to know how much stock they need to acquire to meet the demand of customers. Failing in doing so, a store can lose money in one of two ways. Merchandize running out of stock too soon can lose out on potential transactions and having too much stock can result in failure of turning all goods into revenue. Therefore, forecasting sales is crucial to stocking efficiency.

Conventionally, company can simply use sales record from previous months for forecast stocking. For example, if one item has sold 5 units in the current month, then it may be likely that 5 more units will be sold next month if demand remains constant. However, the conventional method can be improved by using machine learning that incorporate seasonality and trends.

The problem that the capstone project 2 is designed to solve is to forecast future sales of products in a collection of stores. The project is aimed to help companies to improve forecasting performance by using machine learning instead of using sales data of previous months as baseline model. Precise forecasting is particularly beneficial for stores that have products with high price tags and low profit margin. Electronic store, for example, has products with high price tags and low profit margin, which particularly vulnerable to over stocking because having too much merchandize in stock can impact the company's cash flow significantly.

2 Data Acquisition

The dataset used in this project is featured in a Kaggle challenge: Predict Future Sales - Final project for "How to win a data science competition" Coursera course. The data used in this project is a challenging time-series dataset of daily sales data provided by a large Russian software firm - 1C Company. The data is already cleaned and have no missing value.

The Kaggle challenge provides the data in multiple datasets. The training set consists of time-series data of more than 20,000 unique item IDs (SKUs) in 50 stores. And the data span from January 2013 to October 2015. The test set is a collection of store-item labels for predicting the total sales of each store-item pairs in November 2015. There are also a set of files as supplemental information for shop location, item categories, and names of each items.

Since the test set in the kaggle challenge does not provide the target variable, the period between Jan 2013 to Sep 2015 of the training set will be used to train and validate our models. To compare models, the final month of the training data, which is Oct 2015, will be extracted as hold-out set for model comparison.

3 Data Exploration

3.1 Overall data

The entire training dataset contains 2,935,849 entries, which consists of more than 20,000 unique item IDs in total of 50 shops. However, there are roughly 5000 unique item IDs in the month of September 2015 and about 5400 unique item IDs in October 2015, which are only fractions of the item IDs in the entire data set.

Nevertheless, it can be unfeasible to forecast time-series of more than 5000 unique item IDs in multiple stores, which can be upward of 250,000 time-series. Therefore, this project will extract and focus on only the top 50 items with the highest number of units sold between Jan 2013 to Oct 2015.

After extracting the 50 top selling items, the data has now 205,326 record of daily sales. The total sales record of all records combined as time-series in Fig. 1 shows some spikes in sales and higher sale volume between September and January each year. When compared with Fig. 2, it shows that most of the spikes in quantity sold also corresponding to spike in revenue except for one particular occurrence around Nov 2014.

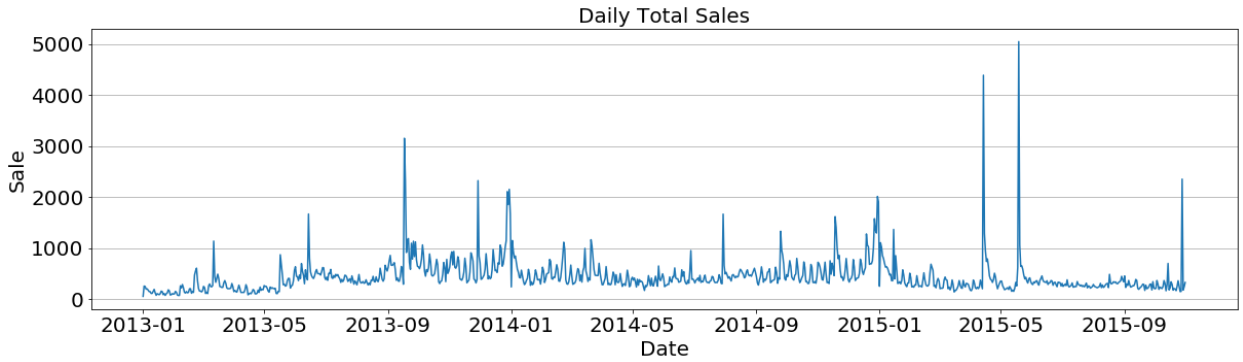


Figure 1: Total daily sales of all shops

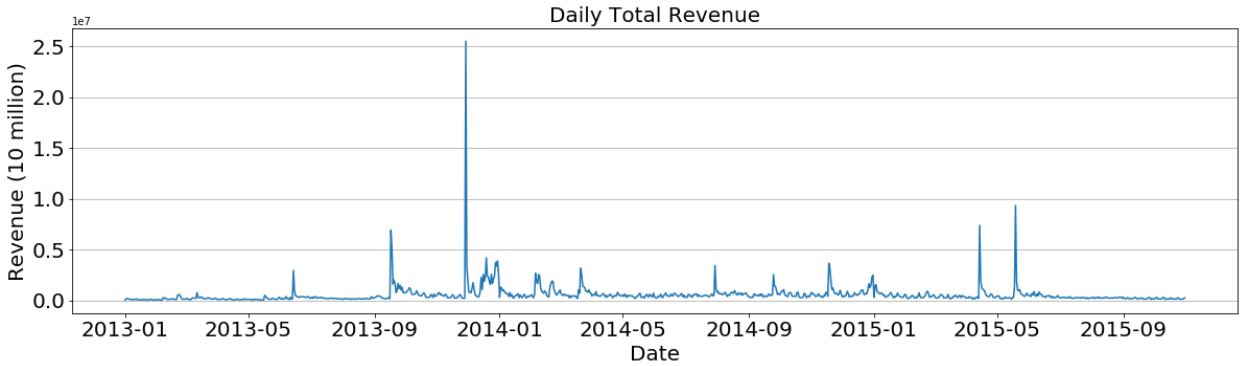


Figure 2: Total daily revenue of all shops

The data contains some rows that have negative values in the item_cnt_day column, they can be assumed as return of items. However, even on monthly level, some items have less quantity sold than items returned, which can be seen as monthly aggregated negative value below.

Number of items have negative sales in a given month: 25

	date_block_num	shop_id	item_id	item_cnt_day
4604	7	2	6457	-1.0
4751	7	12	1830	-1.0
4754	7	12	2753	-1.0
4758	7	12	5822	-1.0
4760	7	12	6740	-2.0
4762	7	12	7894	-1.0
4765	7	12	15044	-1.0
6694	9	12	1830	-1.0
7786	10	12	3329	-1.0
7788	10	12	3732	-4.0
7789	10	12	3734	-2.0
11083	12	56	6675	-1.0
13864	15	6	2753	-1.0
16517	17	12	1905	-1.0
18098	18	24	4870	-1.0
20806	20	21	1830	-1.0
26566	24	12	3733	-1.0
28218	25	19	7018	-1.0
31785	28	4	3733	-1.0
35087	30	41	5672	-1.0
35613	31	4	6675	-1.0
35771	31	12	6497	-1.0
36900	32	5	6738	-1.0
36995	32	12	2808	-1.0
37900	32	57	6675	-1.0

Table 1: Negative monthly sales

3.2 Category

There are total of 84 unique categories in the dataset provided in the Kaggle challenge, but the top 50 selling items in the data for the project only fall into 14 categories. As seen in Fig. 3, nearly of half total items in the data fall into one category.

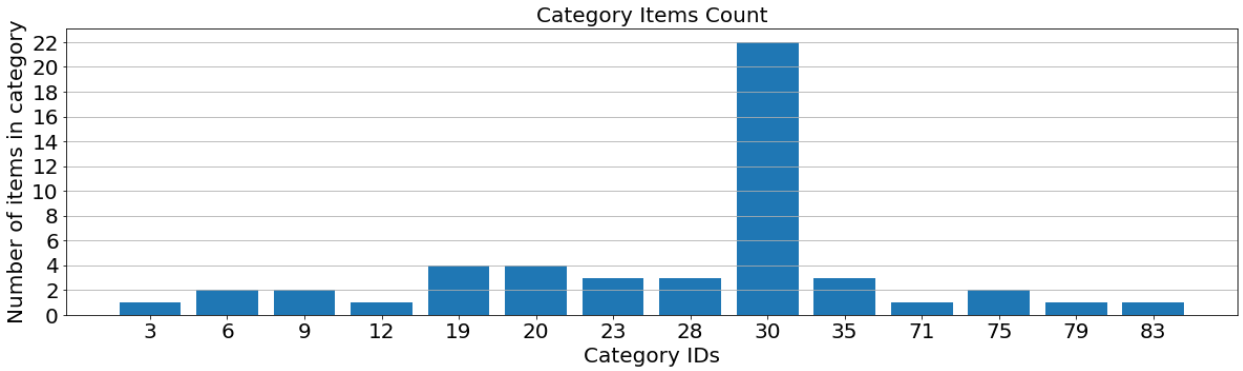


Figure 3: Items count in each category

Interestingly, category 71 has only one item in the category yet it is the highest selling category and also having the lowest revenue according to Fig. 4 and Fig. 5. This indicates that the price for the item in the category must be much lower than any other categories. On the contrary, category 12 has the highest revenue with also single item in the category and moderate sales in comparison of other categories. This means that category 12 has much higher price point than the rest. Category 30 has both second highest number of sales and revenue among the categories.

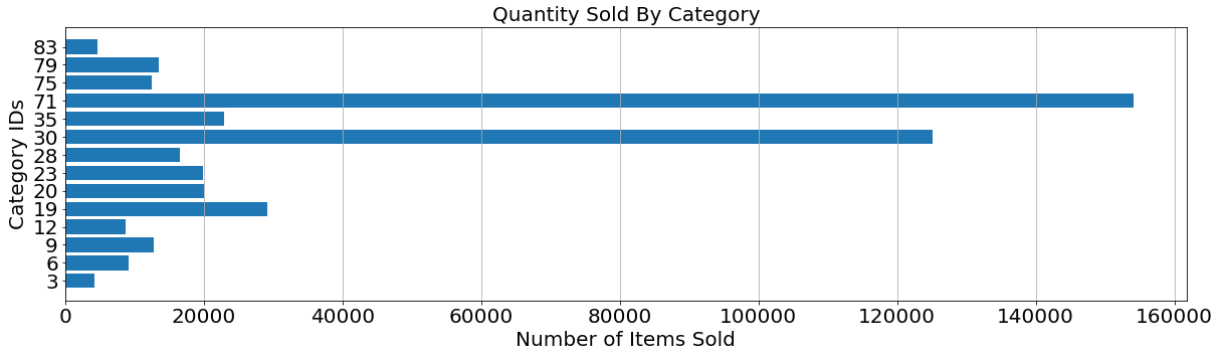


Figure 4: Number of units sold per category

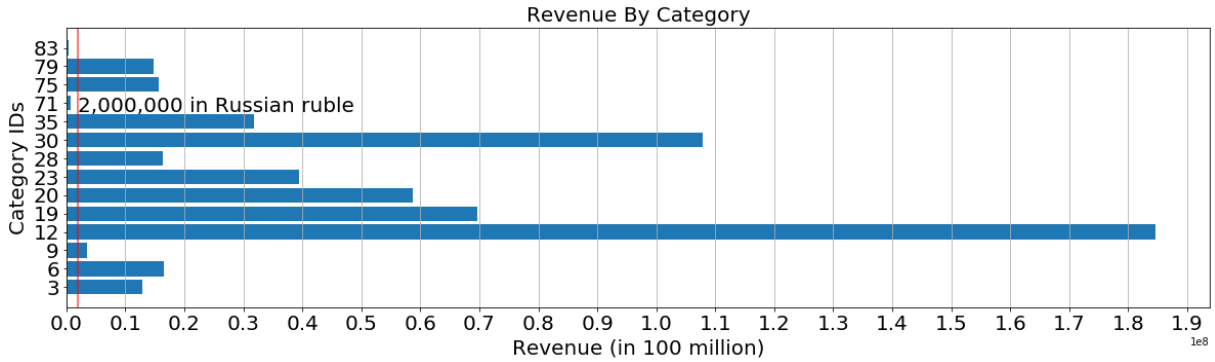


Figure 5: Total revenue per category

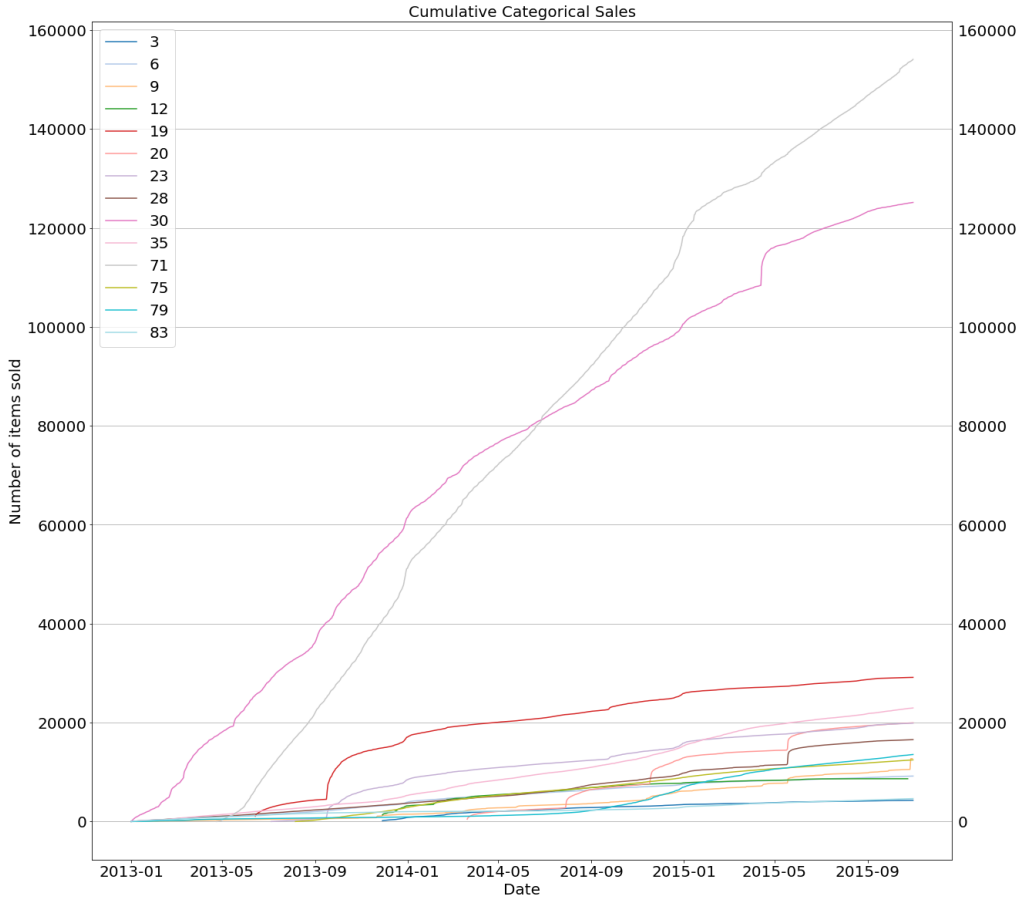


Figure 6: Cumulative sales of all categories

We are able to see growth of sales from Fig. 6 that category 30 and 71 have much higher growth rates than other categories. It also shows that each individual categories have different growth rates and some categories are created in different periods as well. There are some jumps in sales among the categories in different periods, which is likely due to introduction of new items. And there is a jump in sales across multiple categories simultaneously during 2015-05.

3.3 Store

There are total of 60 stores in the entire training set from Kaggle challenge, and it contains sales data of only 43 and 44 stores in September and October of 2015. This means that some stores have ceased operation during the given period. The processed data for the project shows sales record of 40 stores in October 2015, indicating that there are a handful of stores that did not carry the 50 top selling items during the period. Furthermore, Fig. 7 below shows that some shops have only became operational between Jan 2013 and Jan 2015. All 40 stores in the figure show that they have been in operation for at least 12 months, while majority of them are in operation for more than 24 months.

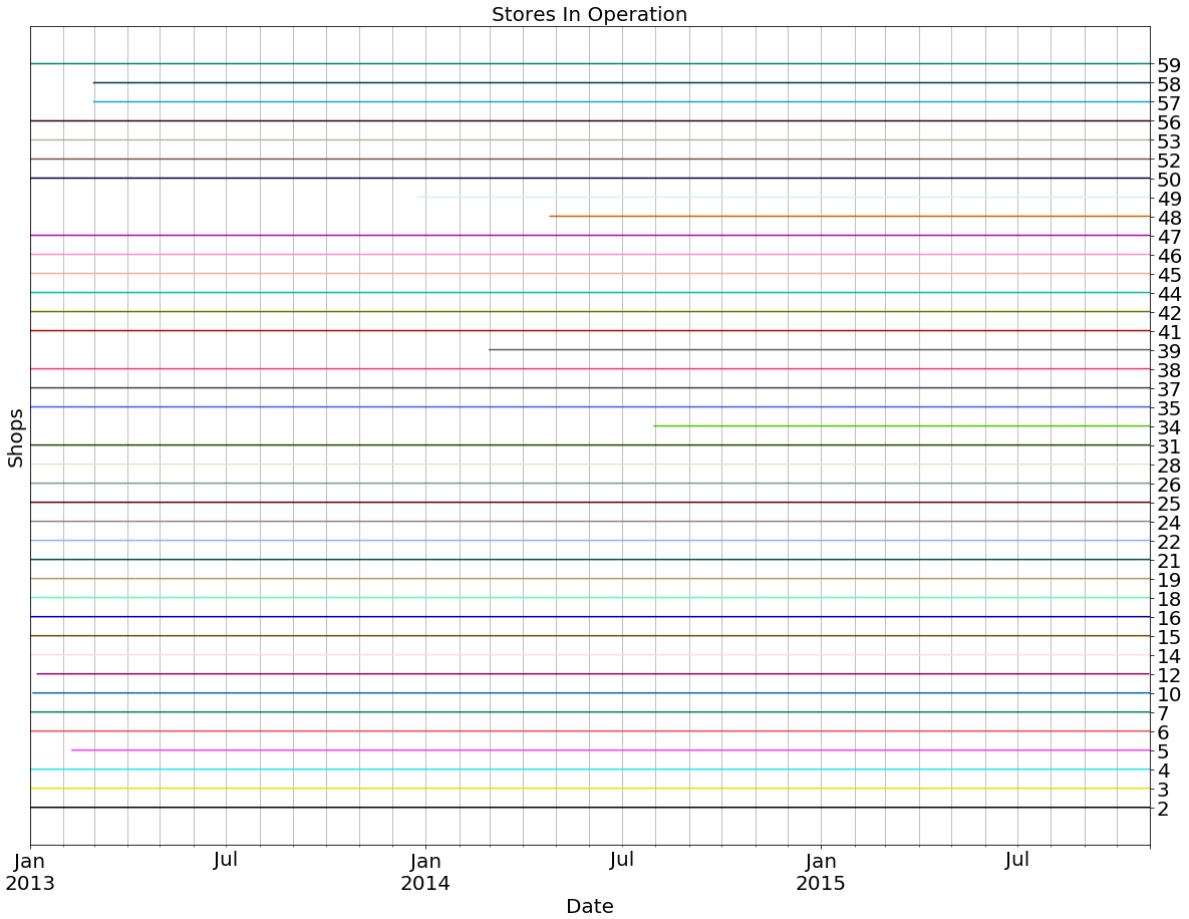


Figure 7: Duration of shops in operation

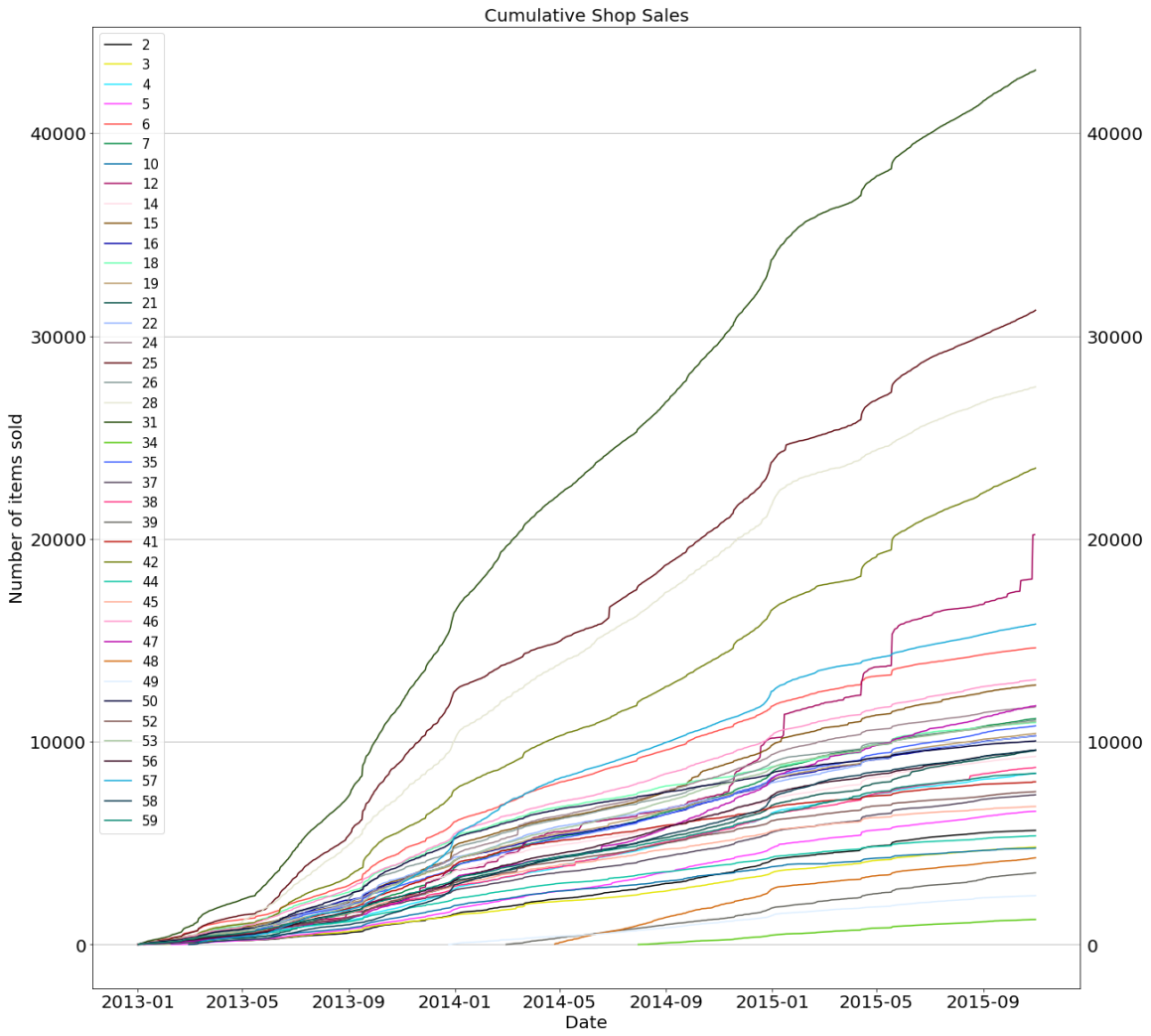


Figure 8: Cumulative sales of each store

Seasonality in sales is visible in Fig. 8 around January of both 2014 and 2015 that many stores have jumps in growth simultaneously. There is also a jump in growth in multiple stores around May 2015 but not in May 2014, this may be explained by other factors such as promotions or release of new items.

The stores with the shortest operation durations also have the least amount of sales per Fig. 9. However, amount of sales and also revenue, Fig. 10, varied greatly among stores that have been operation for longer than 24 months. Furthermore, the relative differences of sales from stores and revenues from stores are similar between Fig. 9 and Fig. 10, which the lengths of the bars of each stores are similar between both figures. This means that proportion of items sold and their pricing are similar among the stores, otherwise the revenue would be disproportionate to the quantity sold by each stores.

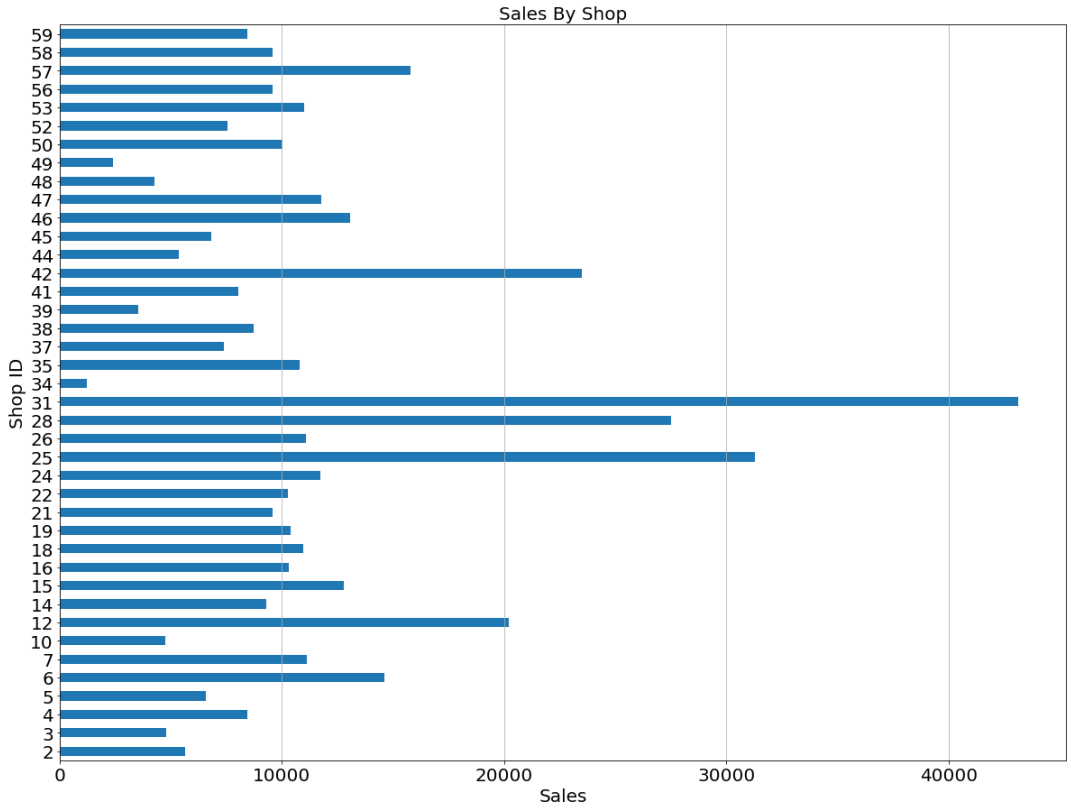


Figure 9: Quantity sold by stores

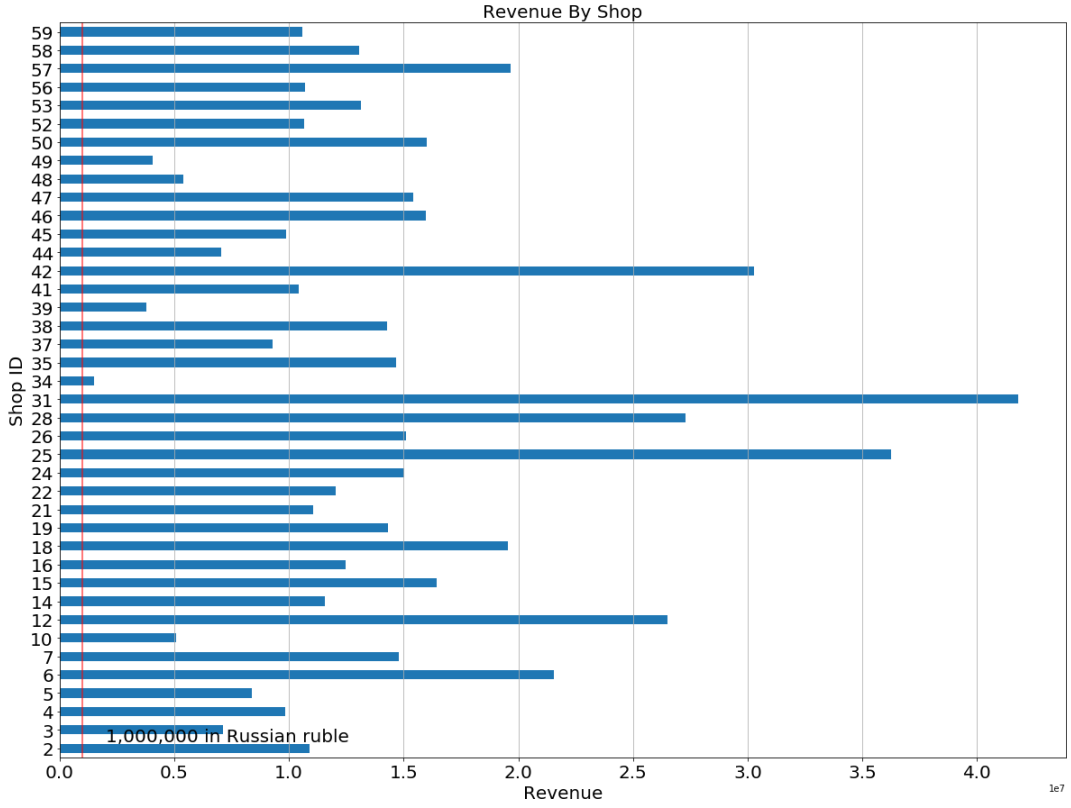


Figure 10: Revenue by stores

3.4 Item

Majority of the items are available in store after January 2013 as shown in Fig.11. There are 13 items that has sales record of less than 24 months, which can be difficult to determine whether they have annual seasonality. The quantity sold of each item is not proportionate to the duration of availability of each items in stores, which indicates that demand varies among the items.

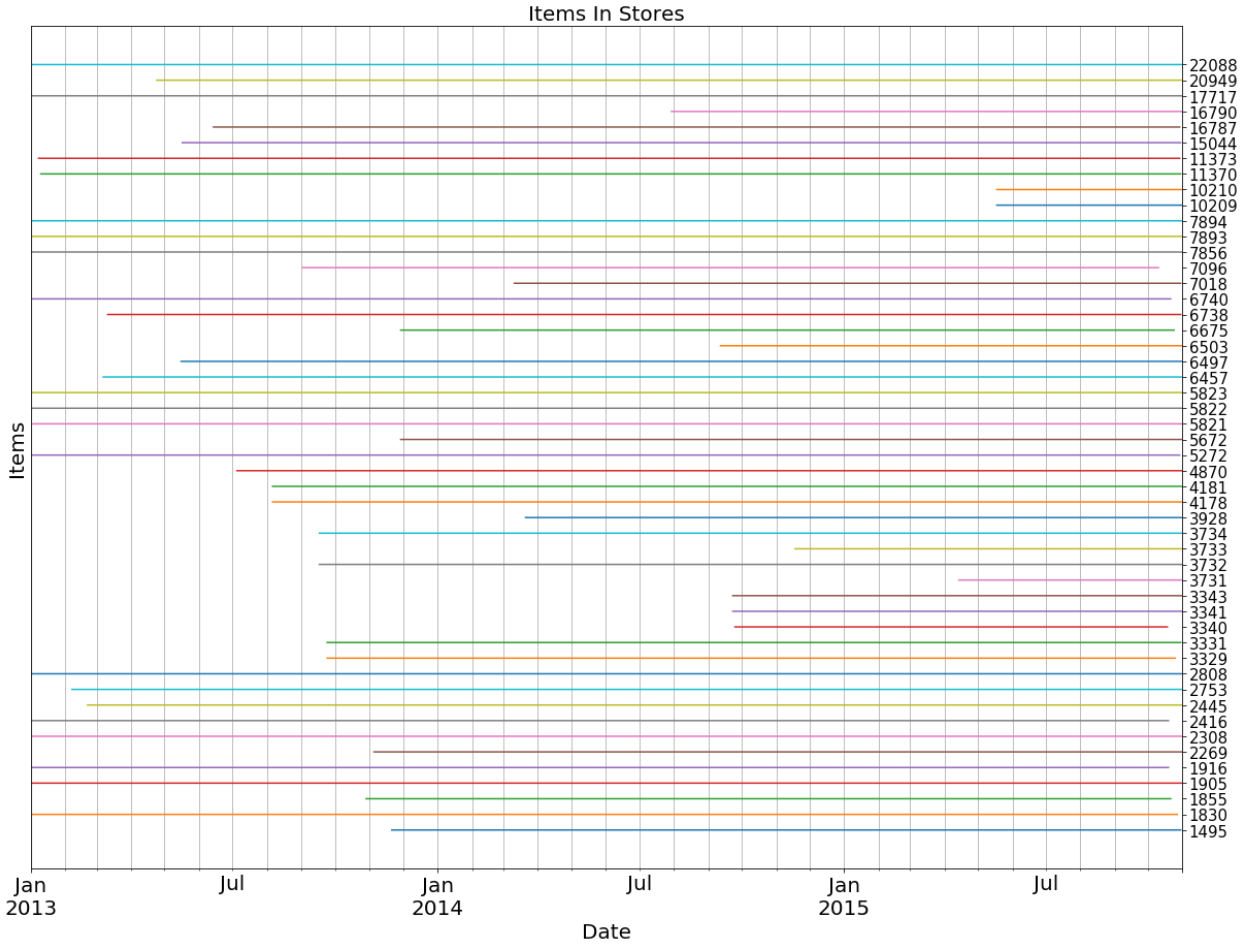


Figure 11: Timeline of item availability

The number of items sold for each item are within 20,000 units except item ID 20949 as shown in Fig. 12, which sold almost 10 times more than the second highest selling item in Fig. 13. However, the revenue of the item is minuscule compared to other items. On the other hand, item ID 6675 has sold around 10,000 units according to the dataset, but has almost 5 times more in revenue than the second top grossing item.

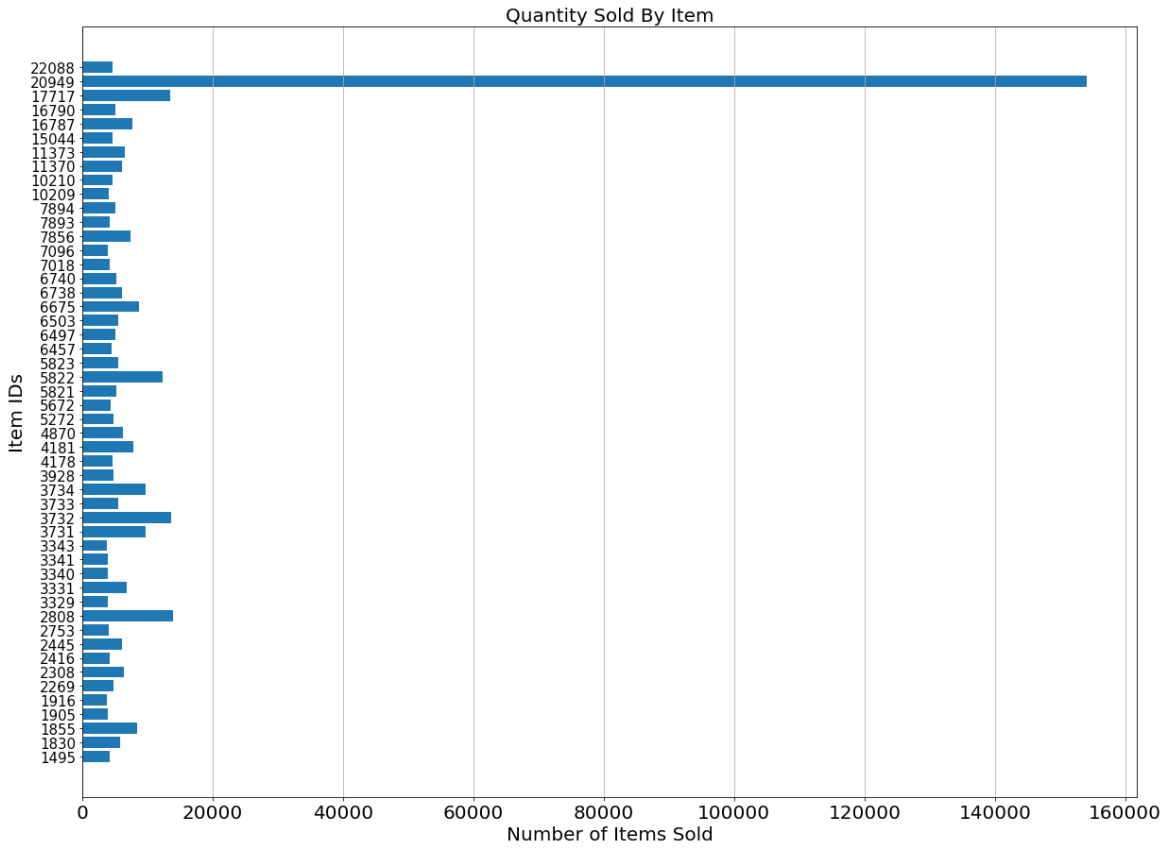


Figure 12: Quantity of items sold for each unique item

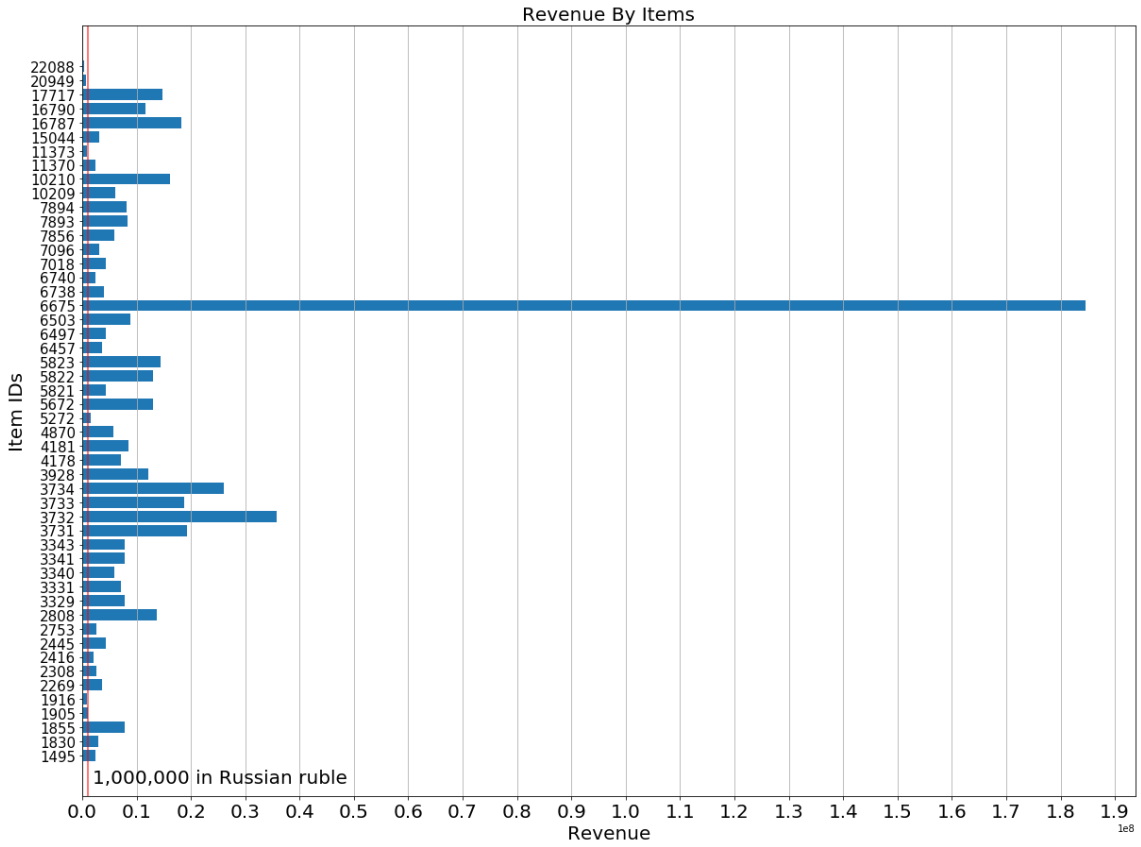


Figure 13: Revenue of each unique item

3.5 Price

The data includes the price of an item when it is sold. With prices and sales record of items as time-series, we should be able to find possible decrease in price that indicates discount or promotion and whether it has correlation to sale increase. Hypothetically, any fluctuation of pricing should affect the number of items sold as long as demand remains constant.

After plotting the price and sales of items as time-series, we can see that the ways that pricing behave can be separated into 3 groups. The first group has the price of items increase over time, shown in Fig. 14. The second group has price of items decrease over time, Fig. 15. Third group has price of items that are not fixed, Fig. 16. Price discounts are visible in most graphs except for the ones that have non-fixed prices.

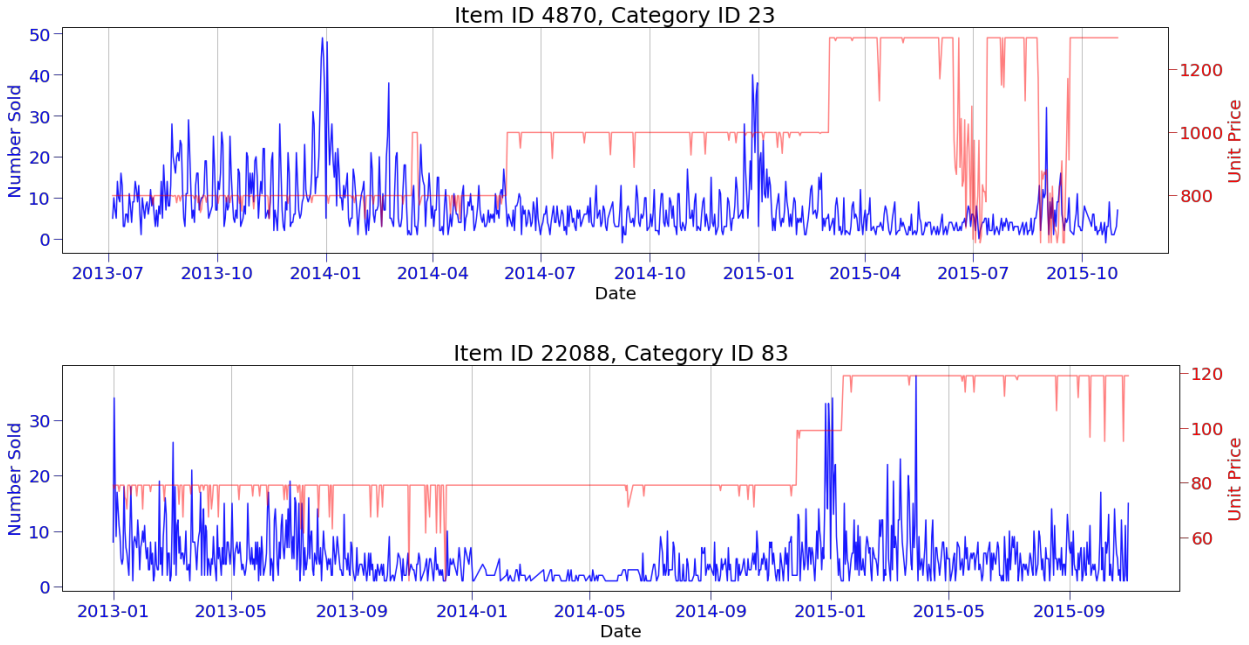


Figure 14: Items with increasing prices

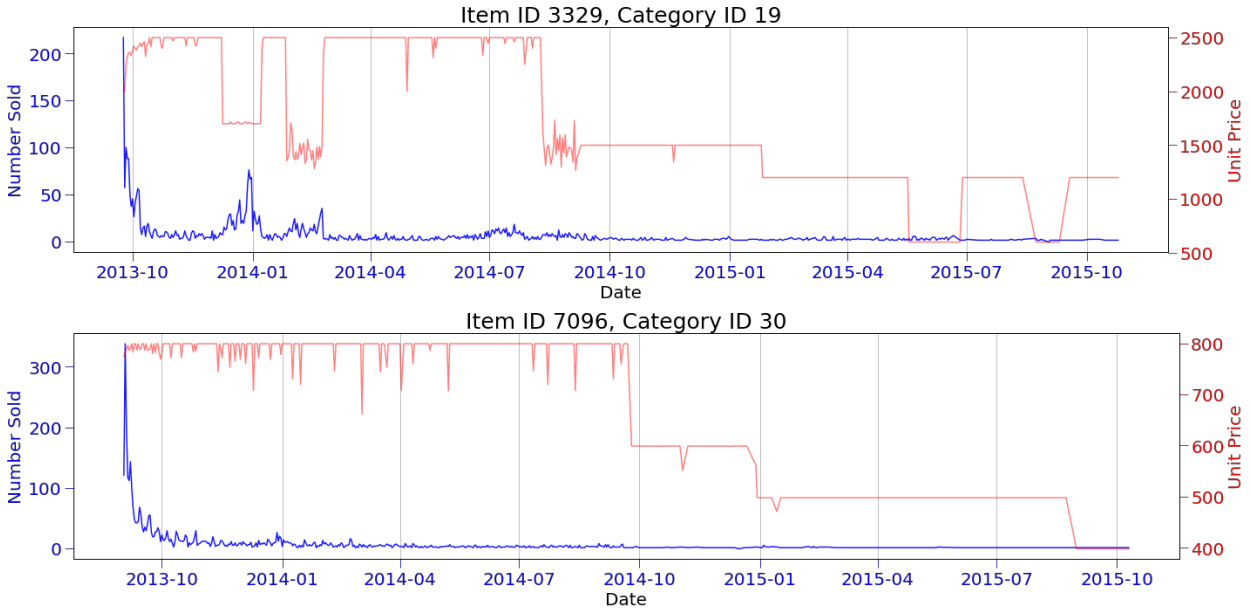


Figure 15: Items with decreasing prices

Items that are most responsive to discounts are the ones with price that decrease over time instead ones that increase over time. This is reasonable because products with increasing pricing usually have inelastic demand compared to items with decreasing pricing. And if demand is inelastic, sales would not change drastically when discount or price hike occurs. On the other hand, products with elastic demand would show larger increase or decrease in sales when responding to price changes.

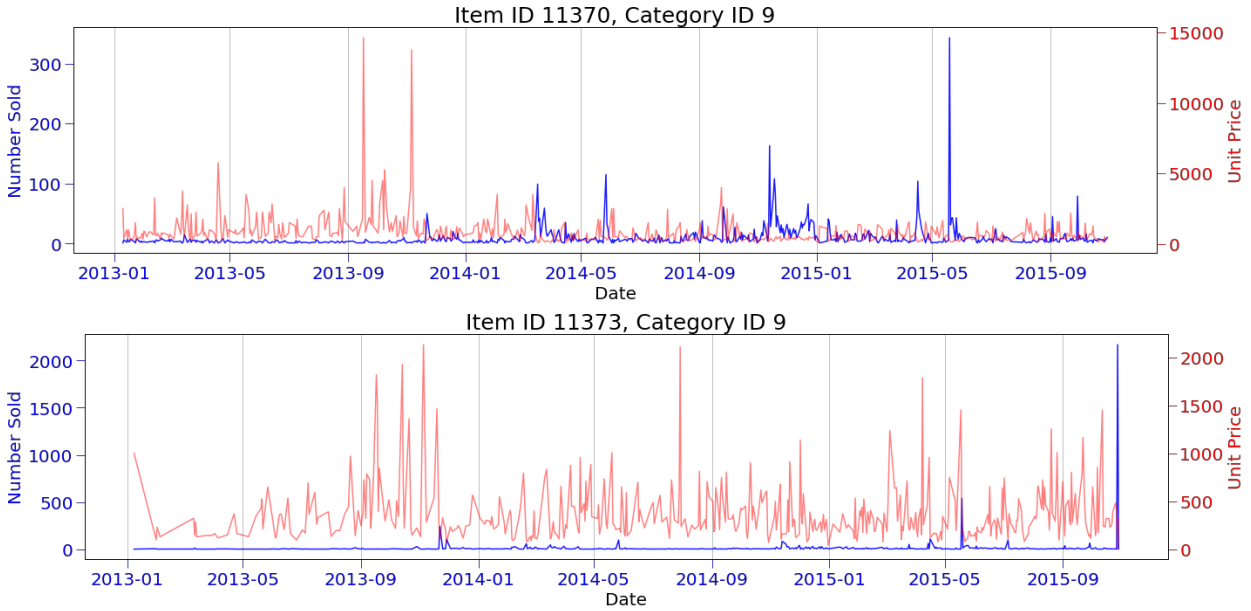


Figure 16: Items with fluctuating prices

We can investigate the items with fluctuating item prices, shown in Fig.16, by looking at their item names. The two items are item 11370 and 11373. By using Google Translate to translate their Russian names provided in the item name file, we can see the product names in English are "Delivery (Moscow)" and "Delivery to the point of delivery (Boxberry)". It makes sense that delivery fees are not fixed fees but flexible pricing according to distance of delivery. Therefore, discount on delivery fee can be difficult to measure so we can ignore them.

Since time-series sales data are not iid, it is difficult to calculate correlation between price discounts and increase in sales besides visualizing the changes. However, we can see its effect on some products which can be useful for engineering additional features for regressors. We can add mean price and price difference for feature engineering.

4 Modeling

4.1 Choosing Methods

Since there are different stores that carry different array of items, the forecast task can be categorized as hierarchical time-series. The total sales from the company can be disaggregated into each stores, and each stores can be disaggregated into each items. There can be three different approaches to make forecast for hierarchical time-series on each item in all of the stores. The approaches are top-down, middle-out, and bottom-up approach.

In this project, the optimal approach would be either bottom up and middle out approach. Bottom up approach predicts all of the time-series at the base level, which is directly calculating monthly sale sum of each unique item ID in each store or their daily sum and aggregate them to monthly sum of sales. Middle out approach takes a middle level and calculate weights that represent the proportion of the total sales at the level would be given to each base level. Afterward, forecast at the middle level will be converted to the base level values. For example, we estimate the proportion of sales each shop would get for an item ID as weight, and divide the total monthly sale of that item ID by the weight to get the monthly sales of the item in the store. Middle out approach is ideal for data that contains hierarchical time series, because the forecast in this project contains large number of unique items in multiple stores and can be very slow to compute.

4.2 Data Pre-Processing

Because the data set has thousands of unique items, it can take millions of iterations even with middle out approach to build models and it is very slow. Therefore, this project will only focus on forecasting the top 50 best selling items within the duration of the data set. Before making predictions, weights of each unique items in each store based on the prior month of prediction will be calculated. The ideology behind that is to use the proportion of an item’s total sale in a given month from each stores would be similar to how the next month would look like. On the

other hand, it could be more noisy to use middle out approach on the store level, because the proportion of how each items are sold in a store may change drastically throughout each month.

4.3 Model Evaluation & Metrics

The project will treat September sales data as test set to evaluate the performances of each models and the final month of the training data, which is Oct 2015, will be extracted as hold-out set for model comparison. However, time-series forecasting method such as SARIMA is most accurate by using the most recent data. Therefore, SARIMA will use all sales data up to September 2015 as training data and use the last month, October 2015, for evaluate performance.

To measure model performance, metrics such as mean absolute error (MAE) and coefficient of determination (R squared) would be appropriate for evaluation. MAE is a popular metric to measure time-series analysis performance and R squared is used to evaluate regressors such as random forest and gradient boosting regressor. On the other hand, another popular metric for time-series analysis which is mean absolute percent error (MAPE) is not used in this project. The reason is that he dataset contains zero in daily and even monthly sales sum for some items and MAPE would yield infinite values that are uninterpretable.

Without machine learning, people use historical data to make forecast on future sales. In this project, the objective is to evaluate the performance of the models. Besides using metric to evaluate model performance, we will also create a model from sale data of prior month as baseline to see improvement of forecasting by using machine learning models.

The baseline model has fairly high accuracy on sales data for Sep 2015. The model is R2 score of 0.84 and MAE of 2.44. When forecasting sales for Oct 2015, the baseline model has R2 score of 0.289 and mean absolute error of 5.273. The MAE means that all of the unique items in each store are off by 5.273 on average. In Fig. 17, there are two points that are way off of the central diagonal line that represents accurate prediction and they may be contributing to the low performance.

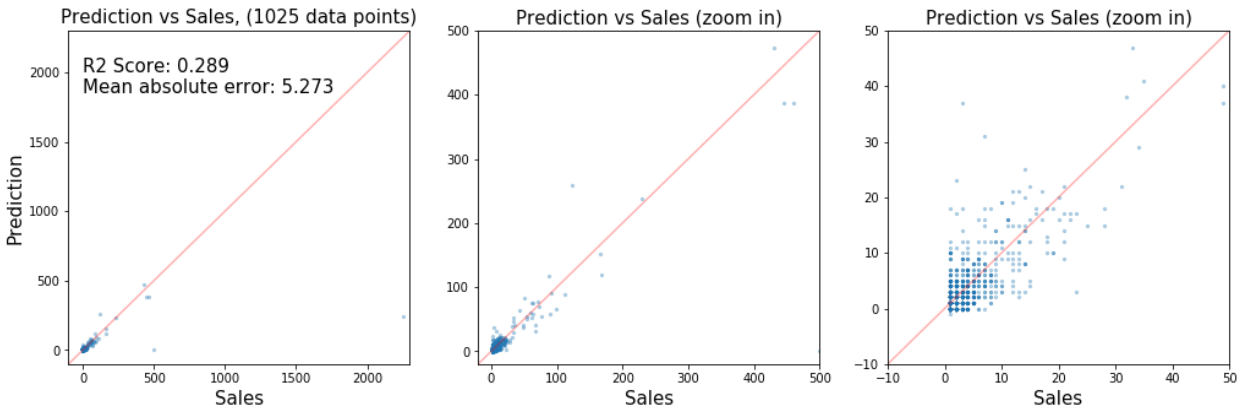


Figure 17: Scatter plot of prediction vs actual sales for baseline model for Oct 2015

If we would remove the abnormal sales data of the two points, we can see that baseline model has high accuracy of R2 score of 0.936. And the mean absolute error is nearly halved from 5.273 to 2.833.

4.4 Seasonal Autoregressive Integrated Moving Average (SARIMA)

Since the project is to predict the sales of items in the last month of dataset, October 2015, typical time-series forecast model such as seasonal autoregressive integrated moving average (SARIMA) would be used for modeling. Because the data set has thousands of unique items, it can take millions of iterations even with middle out approach. However, to reduce time on building the model, the dataset is resampled as monthly sales data. The dataset is then converted into a dictionary of time-series that each key-value represents an item ID.

The data that is used for fitting SARIMA is the date of sales and the number sold, since the model does not take other features in consideration. SARIMA has total of 8 hyperparameters that need to be configured to make proper predictions. In this case, the project uses grid search to determine the hyperparameters for each time-series of individual items. The grid search uses sales data of October 2015 as validation to find the most fitting hyperparameters. The returned hyperparameters associated to lowest mean absolute error on the Oct 2015 are then save in a dictionary then sliced to fit into the SARIMA model that iterates the training data as dictionary.

The SARIMA forecast on sales data of Oct 2015 has R2 score of 0.38, which indicates higher accuracy than the baseline model. However, the mean absolute error is 5.804, which means higher error on average despite higher accuracy.

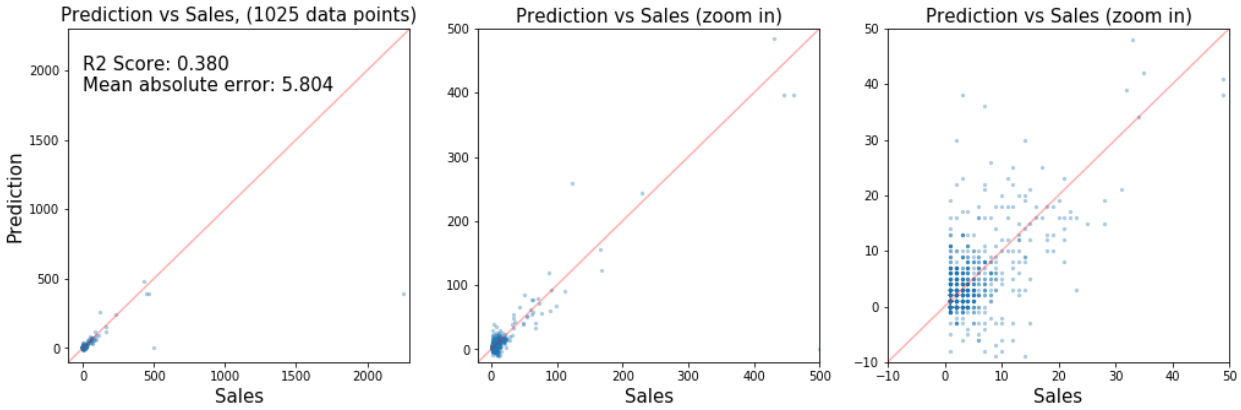


Figure 18: Scatter plot of prediction vs actual sales for SARIMA model

4.5. Facebook Prophet

This project will also implement Prophet, an automatic forecasting procedure developed by Facebook. To fit data into Prophet, the dataset has to be converted into only two columns, which the date column is renamed as ds and number of units sold is renamed as y. As seen previously in

exploratory data analysis, some items sales behave like an exponential function where sales are the greatest in the beginning of release and quickly drop an asymptote that is near one or zero in daily sales. To find fitting model with Prophet for a time-series that behave like an exponential function, we can assume they have logistic growth rate that decrease over time.

To build Prophet model on time-series that have logistic growth rate, it requires definition of both growth cap and growth floor. The daily sales would remain at zero and does not decrease into negative unless some units are returned, this means that we have to set a saturation minimum or growth floor at 0 for modeling to ensure that time-series projections do not below 0. Since growth does not saturate, we pick the maximum sales value in the past three months as growth cap as an arbitrary number.

Besides growth rate, each item also have different characteristics in term of weekly and yearly seasonality. Like SARIMA modeling, we use grid search to determine the hyperparameters to build the Prophet models using root-squared error for validation.

The performance of Prophet on predicting sales for Sep 2015 has R2 of 0.88 and is fairly accurate. The Prophet forecast on sales data of Oct 2015 has R2 score of 0.312 and MAE of 5.577. It means that although the model is slightly less accurate, but it has smaller error on average.

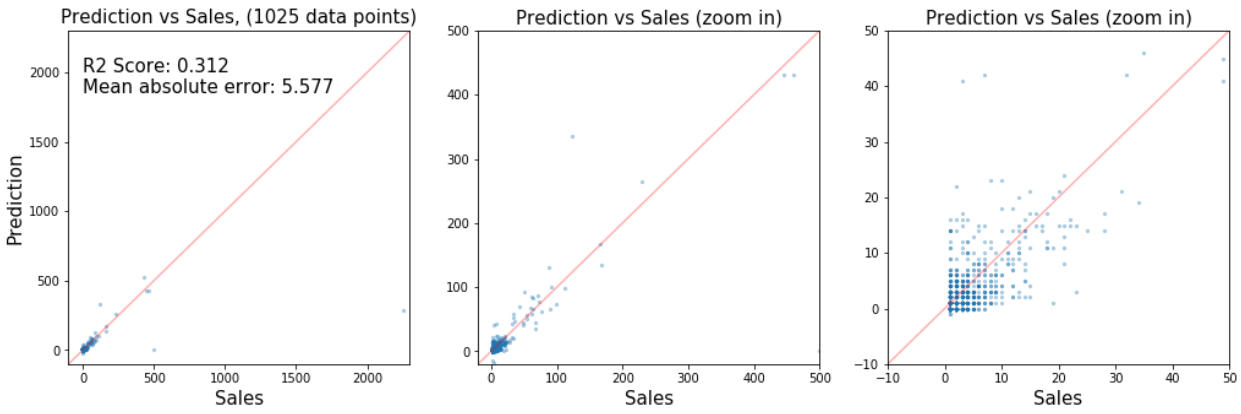


Figure 19: Scatter plot of prediction vs actual sales for Prophet model

4.7 Random Forest

Besides models particularly designed for time-series analysis, the project will also include regressors such as random forest and gradient boosting. To speed up the forecast of random forest, we aggregated the daily sales data into monthly total. Unlike SARIMA and Prophet, the dataset used in regressor would be a single dataframe that contains all unique items that are differentiated by columns of shop ID and item ID. Besides features that identify each items, additional features can be incorporated in model training. We will also include item pricing to engineer lagged and difference data.

The new features that are generated from existing data in the dataset are last month's sales, sales difference from last month, last month's price, price difference from last month, and percent change from last month's price. To test the optimal amount of engineered features, we created two more data frames that contain additional lagged data that goes back two and three months each. Next, the model is trained with walk forward validation on both August and September 2015. The optimal number of features is the one that contains lagged data with only one month prior. The result is lowest mean absolute error averaged from both months, indicating less overfitting. The out-of-box result of the model shows improvement from the baseline model on forecasting sales data of Sep 2015 with R2 score of 0.93 and MAE of 1.973.

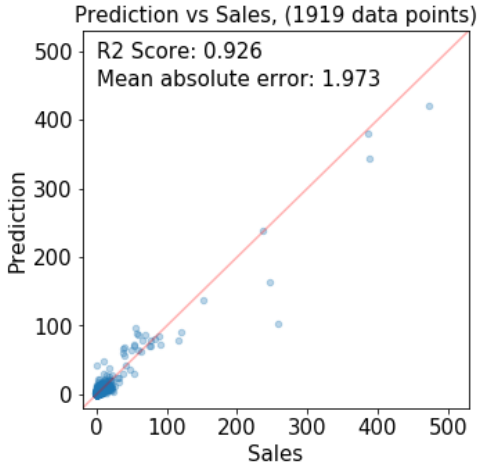


Figure 20: Scatter plot of prediction vs actual sales for random forest model on Sep 2015 data

The random forest model is further optimized by graph the R2 scores of different number of trees. We find that R2 score plateaued at 100 trees, which mean using 100 trees as hyperparameter would achieve similar accuracy as greater number of trees but the computational time is reduced.

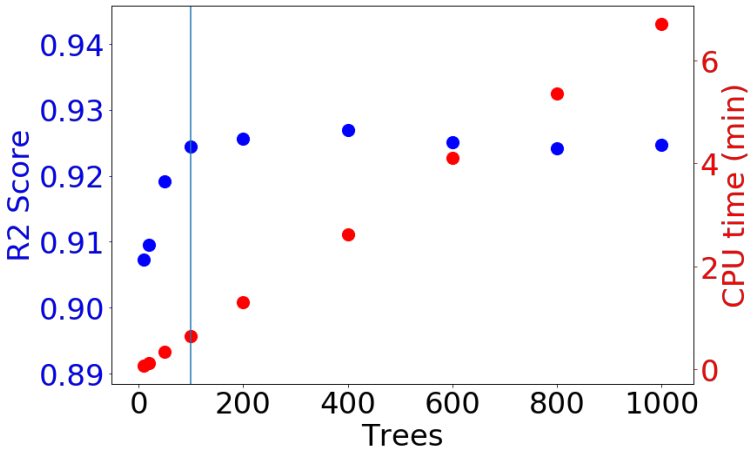


Figure 21: R2 score (blue) and CPU time (min) as a function of number of estimators, the vertical blue line points out R2 score plateaus at 100 trees.

The model uses total of 12 features for forecasting. The feature importance of random forest shows that last month's sales has significantly higher importance compared to the rest at 0.64, and all other features are below 0.1. This makes sense because the baseline model that uses last month's sale record has R2 value of 0.84 on predicting September's sales. The other reason for this observation is that the data is a time-series which each data point may be dependent to prior data. Since the model has only 12 features, there are not much room for improving calculation by removing features.

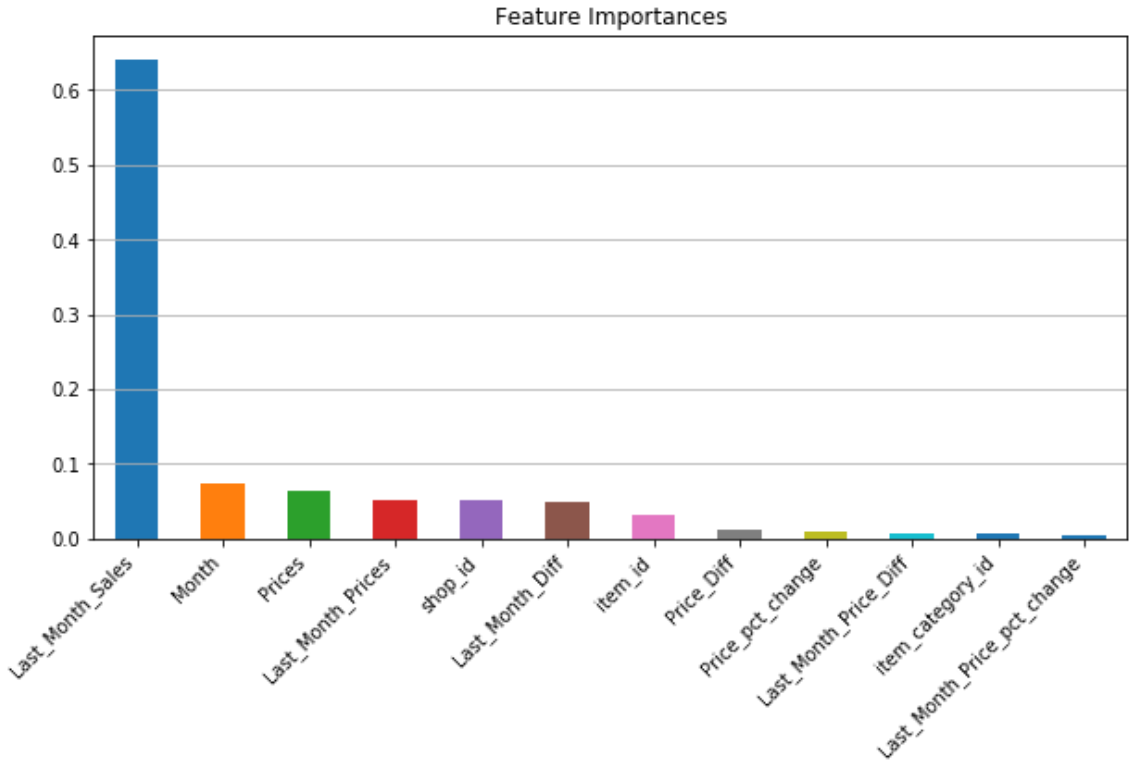


Figure 22: Feature importance of random forest regressor model

After tuning the model, the performance of random forest regressor on the last month's data, Oct 2015, has R2 score of 0.252 and MAE of 5.014.

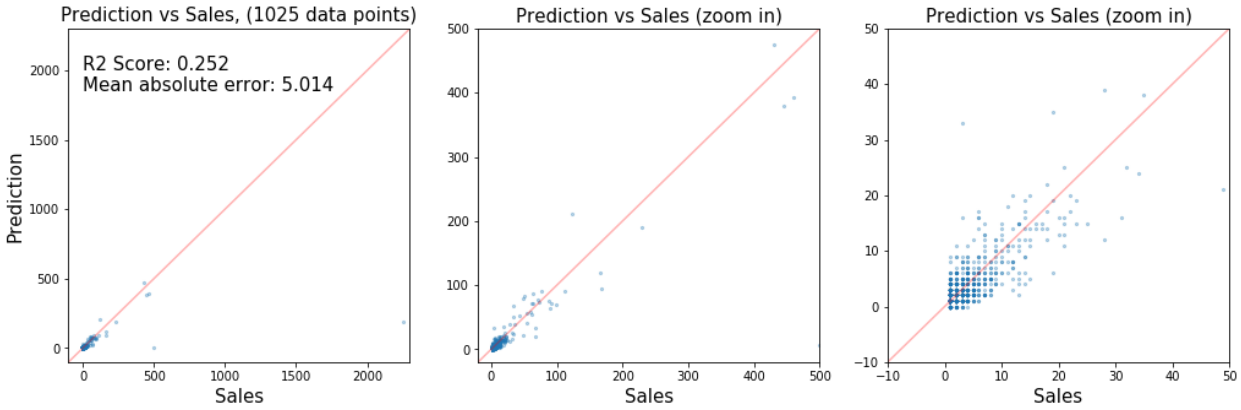


Figure 23: Scatter plot of prediction vs actual sales for random forest model

4.8 Gradient Boosting

Like random forest regressor, gradient boosting also uses the single dataset that contains all items for forecasting. The training time of gradient boosting is much quicker compared to random forest. Gradient boosting model also shows least overfitting at first lag of monthly data for Aug and Sep 2015. The performance of the model shows little or no improvement from baseline model on sales data for Sep 2015 with R2 score 0.90 and MAE of 2.5.

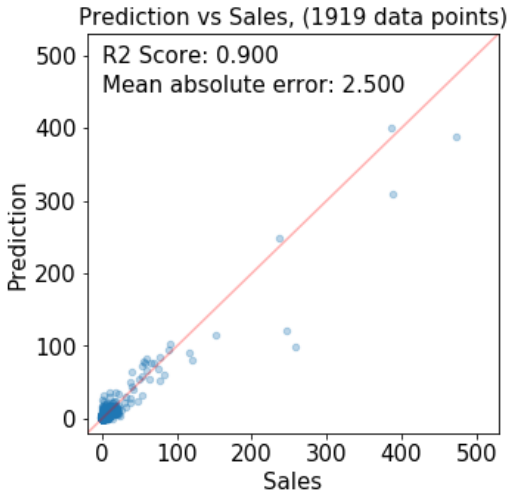


Figure 24: Scatter plot of prediction vs actual sales for gradient boosting model on Sep 2015

Using the same method, we can see the R2 score plateaued at 200 estimators for gradient boosting. We will use 200 as number estimators for model comparison.

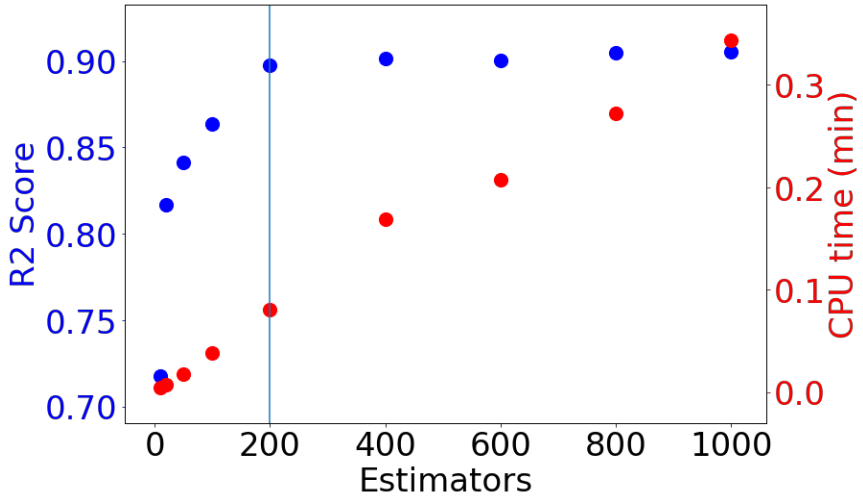


Figure 25: R2 score (blue) and CPU time (min) as a function of number of estimators, the vertical blue line points out R2 score plateaus at 200 estimators.

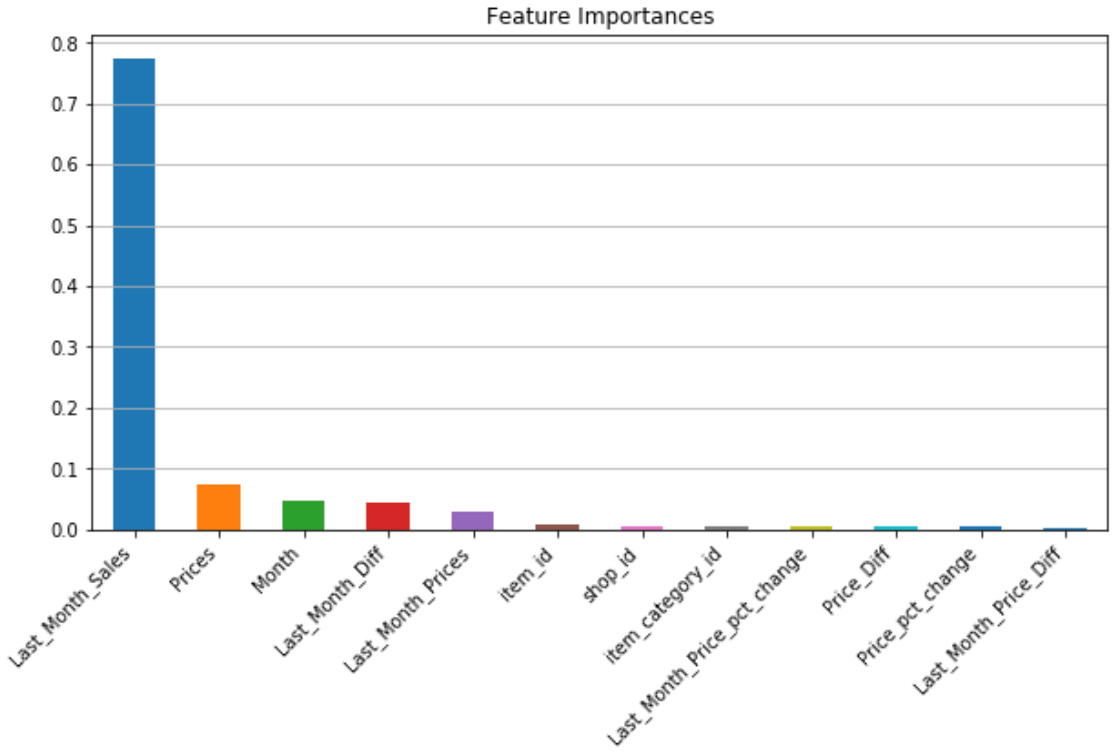


Figure 26: Feature importance of gradient boosting regressor model

Feature importance from gradient boosting regressor also shows sale of last month is weighted much more than the rest of features with even higher value at 0.77. Again, there is not much room for improvement with feature selection since there are total of 12 features.

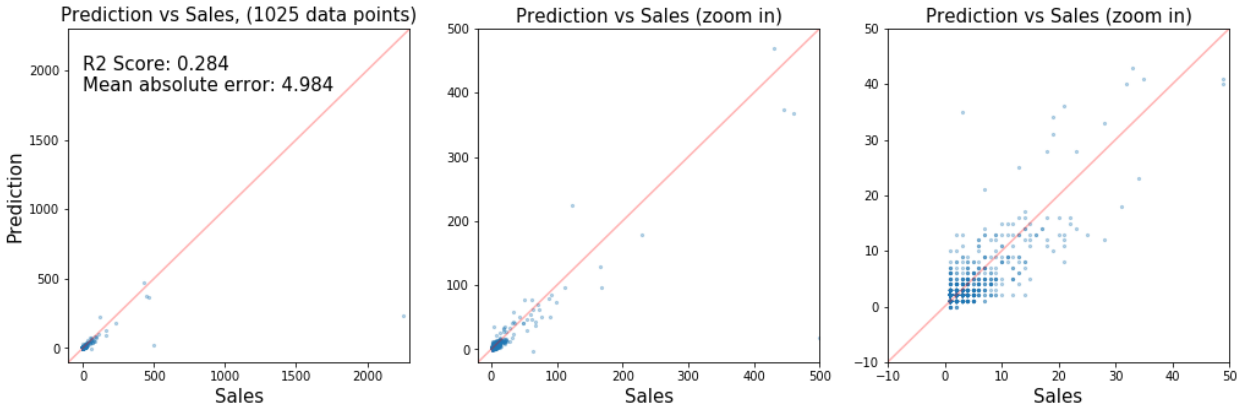


Figure 27: Scatter plot of prediction vs actual sales for gradient boosting model

After tuning the model, the performance of gradient boosting regressor on the last month's data, Oct 2015, has R2 score of 0.284 and MAE of 4.984.

5 Model Comparison/Finding

Overall, all models show little or no improvement from the baseline model with MAE of 5.273 on forecasting the sales of Oct 2015. Gradient boosting model yielded the lowest mean absolute error at 4.984. Both SARIMA and Prophet yielded higher R2 score than baseline, but they also have higher MAE.

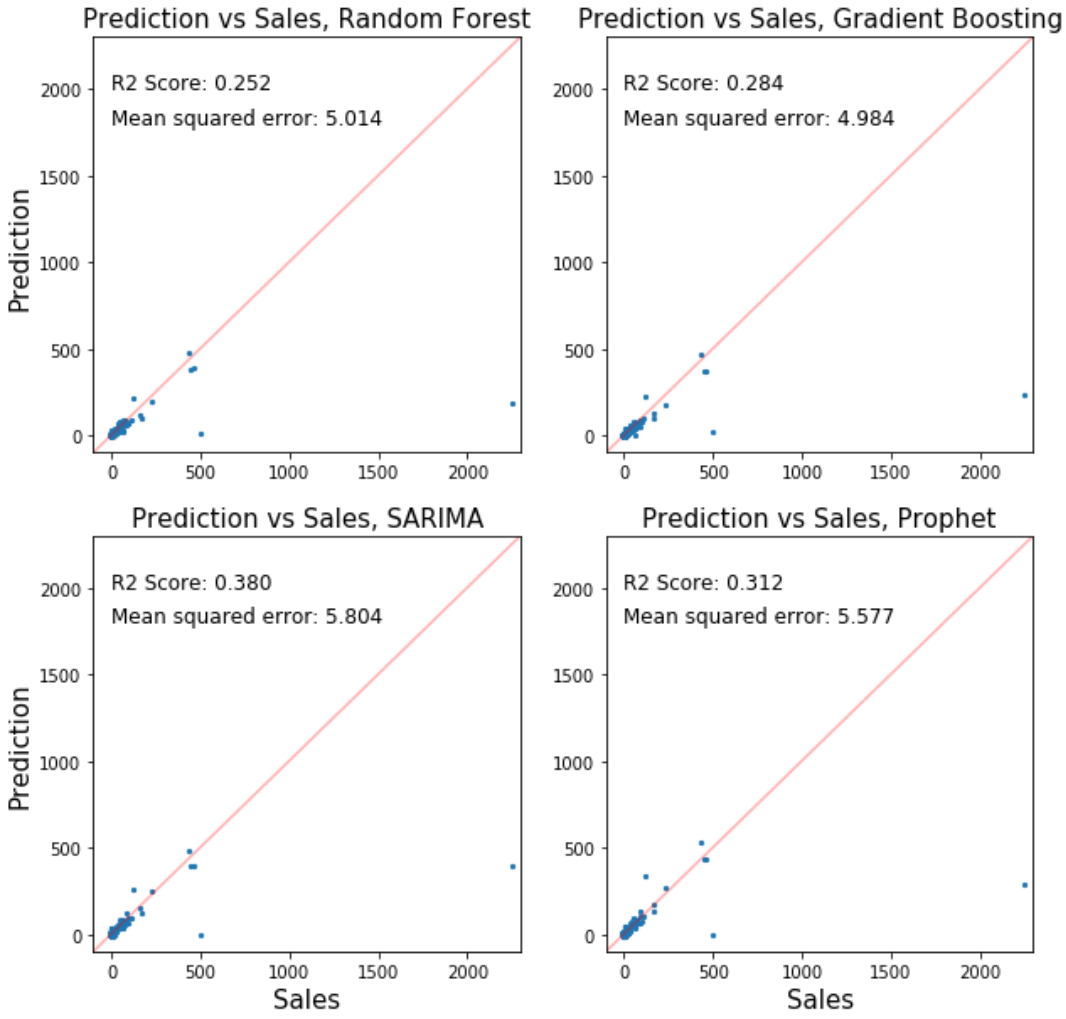
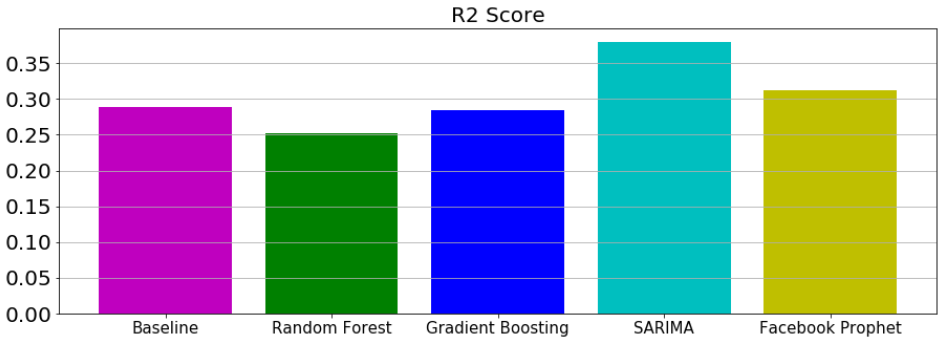


Figure 28: Scatter plots of all four models



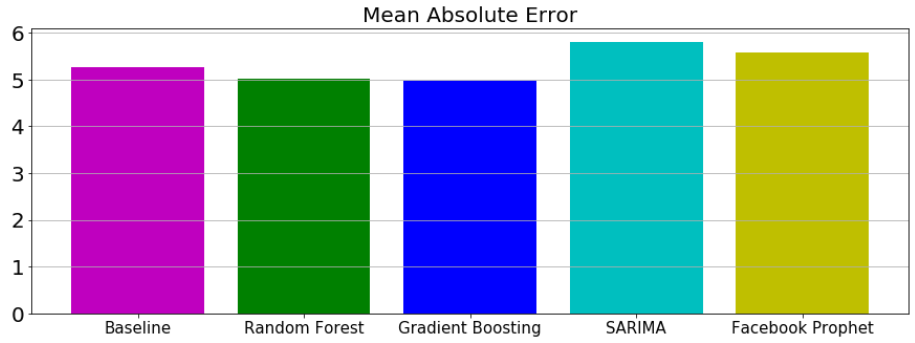


Figure: 29: Bar graphs of R2 score and mean absolute error for all models

All models, including baseline, failed to capture two data points with the largest number of sales. One of the data points have quantity sold at around 2400 and the other is around 500. By removing these two abnormal points, we can see that each model performed similarly to their performance on sales data for Sep 2015. Although R2 score of each model increased nearly three fold, the MAEs only reduced by around half. This indicates that R2 score is more sensitive to larger data points. As a result, both random forest and gradient boosting regressors have smaller MAE than the baseline model.

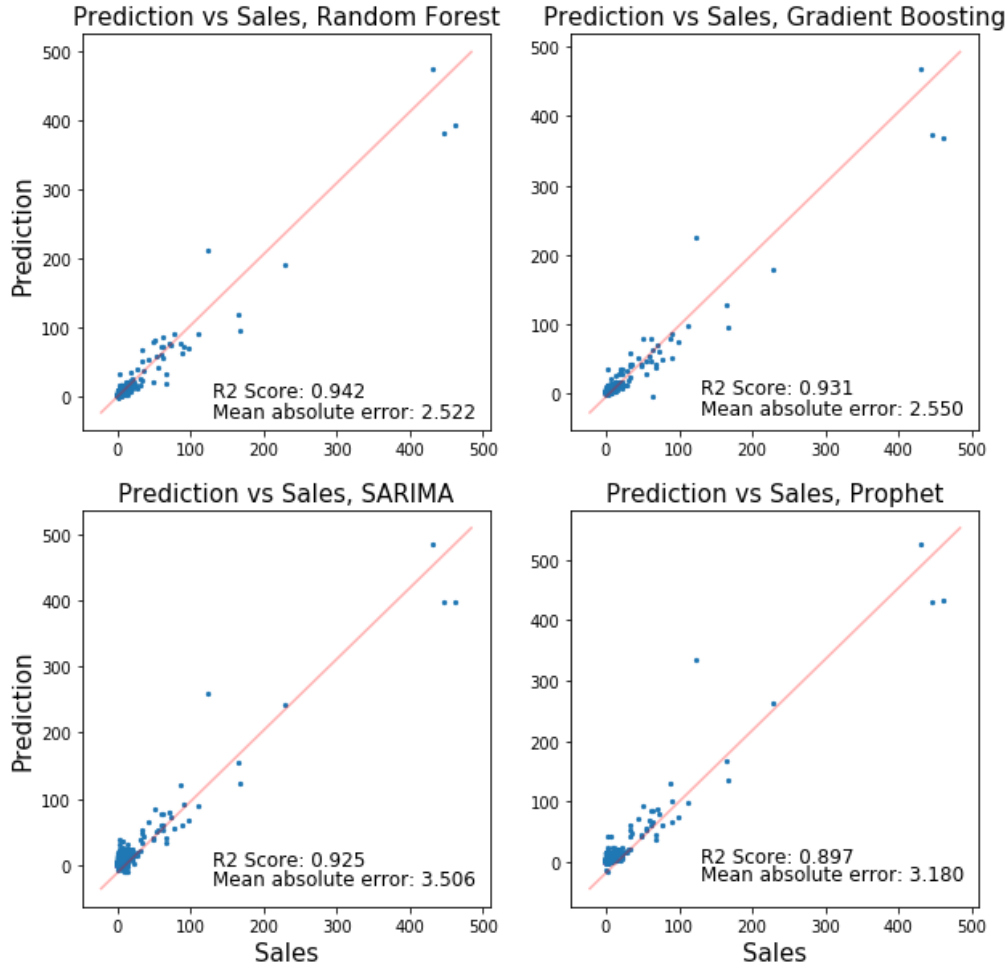


Figure 30: Scatter plots of all four models after removing the two abnormally high sales

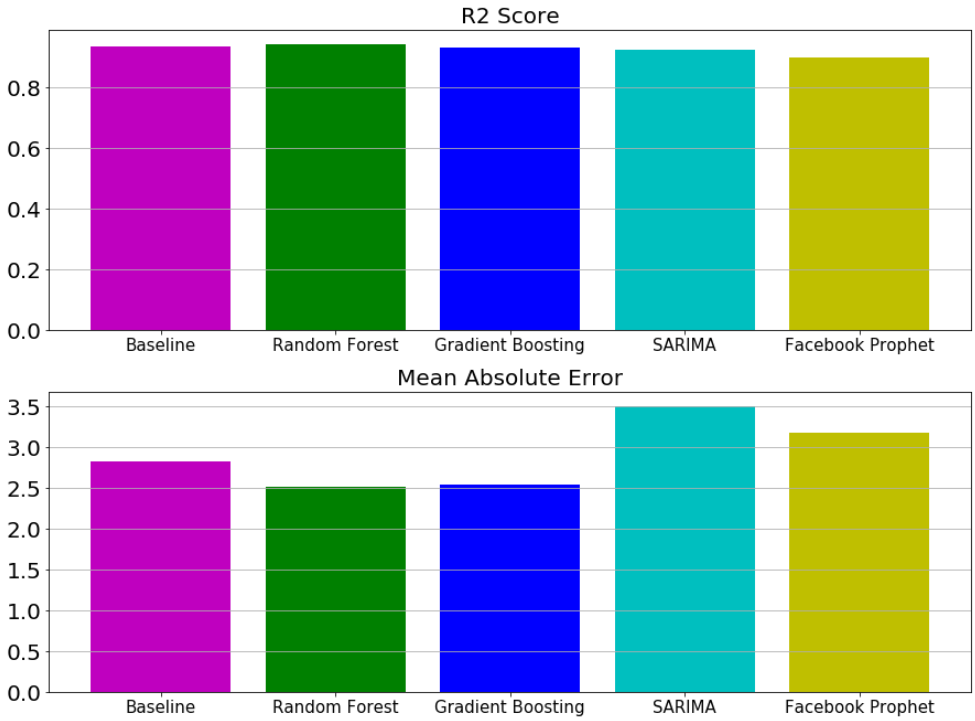


Figure 31: Bar graphs of R2 score and mean absolute error for models after removing the two abnormally high sales

Besides the accuracy of forecasting sales, each model requires different amount of time to train and tune the hyperparameters. Since SARIMA has the most hyperparameters, it takes the longest to be built. In this case, it took nearly 8 hours to perform grid search to obtain the hyperparameters for only 50 items. For Prophet, it takes about 20 minutes to obtain the hyperparameters without walk through validation. On the other hand, random forest and gradient boosting regressors only took minutes to train to forecast. After all, although gradient boosting regressor is not a typical method for making time-series forecasting, it is able to make predictions that is quicker than SARIMA and Prophet. The prediction from gradient boosting regressor also has lower mean absolute error compared to the time-series forecast models.

6 Conclusion

6.1 Model Conclusion

Gradient boosting regressor successfully reduced mean absolute error from baseline model by 0.289. There are total of 1025 store-item pairs in the sales data of Oct 2015, which is total reduction error of nearly 300 units in a single month. This error reduction includes either over prediction or under prediction on future sales. Although this may be a small number in a scale of 40+ stores, the result is yielded from only 50 unique items. This number can be larger when including all unique item IDs.

6.2 Assumption and Limitation

The project used various methods to make time-series prediction. The models assume that all of the sales are not independent. Although regression methods such as random forest and gradient boosting are used, they retained dependency of time-series by using lagged data.

The sales data contains negative units sold. In this case, we assume the negative sales are items returned. We can replace all of the negative values on days that number of units returned exceeds units sold, but we will not be able to exclude any units returned on the days that there are more units sold. By doing so, it can cause distortion of the sales data. Since we do not have accurate record of units returned, the project accepts negative sales value and recognize them in modeling.

6.3 Future Work

Beyond the sales data that was acquired for the project, there are additional data that can be acquired to enhance our forecast. The data used in the project is part of the daily sales data obtained from 1C Company's store in Russia. We can acquire dates of special events in Russia that may affect sales. For example, dates that are similar to Black Friday or Cyber Monday in the US that could boost sales of software and electronics. The dates can feed into built-in feature of Prophet to model holiday effects which may enhance forecast.

Besides special event dates, internal promotion data on price reduction of items and duration can allow models to accurately capture the effects of promotion instead of extrapolate the information passively from sales data.

Since the project only focused on the top selling items in the company, it may neglect any dependency between items in a given store. Building forecast models based on the entire sales data may improve the accuracy.