Corporación Favorita Grocery Sales Forecasting

Business Content

The stock in the grocery is highly influenced by the predicted sales. If predict a little over, the perishable goods will be overstocked in the grocers; oppositely, if estimated a little bit under, the popular items sell out quickly, which induce money loss and customer unsatisfied. The forecasting can become more complicated when particular item on promotion and new stores are added or facing new economic environment.

Corporación Favorita, a large Ecuadorian-based grocery retailer, which has hundreds of supermarkets, with over 200,000 different products on their shelves, used to rely on subjective forecasting methods, tries to find a solution in machine learning models to accurately predict the unit sales for thousands of items sold at different Favorita stores located in Ecuador.

Data Description

The data comes from Kaggle competition.
https://www.kaggle.com/c/favorita-grocery-sales-forecasting/data

It includes training data, test data, stores data, items data, oil data, transaction data and holiday events data.
In training data, it gives the unit_sales and the information of whether or not on promotion by date, store number and item number. The store meta data includes city, state, type and cluster (a grouping of similar stores) for particular store numbers. The items meta data includes family, class and perishable) for particular item numbers. Other additional data like oil data, transaction data show daily oil price and daily transactions. The holiday events give information of holiday type and date. More detailed description please refers to the website.

The date of training data provided are from 2013-01-01 to 2017-08-15 while that of test data are from 2017-08-16

EDA

First of all, we are going to look at the distribution of category variables in our data.

1. Store information

The count plot for store numbers in the training data are given in figure1. The majority of stores have items sale records more than 1,500,000 since January 2013. An interesting store is number 52, only has less than 500,000 sale records.

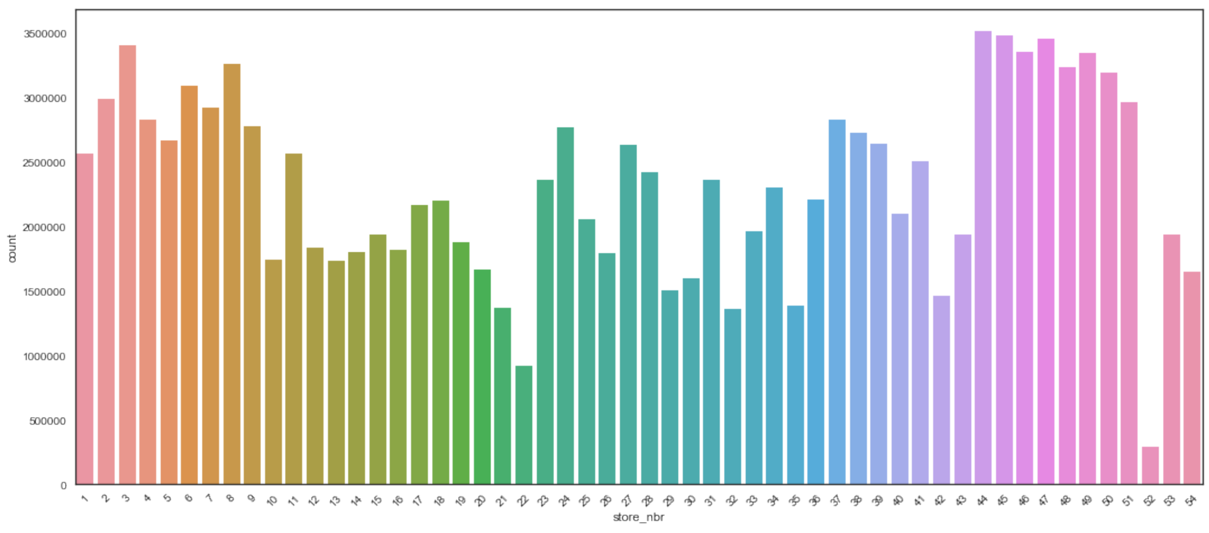


figure 1.1 count plot for store numbers in the training data

The geolocation distributions of stores are given in figure 1.2. From the figure, we see that the city of Quito and Guayaquil take the leading board, which have significant numbers of stores. A similar pattern can be seen in the state distribution, where the state Rchincha and Guayas outnumber other states. It can be inferred those cities and states have more product demands or large population density.

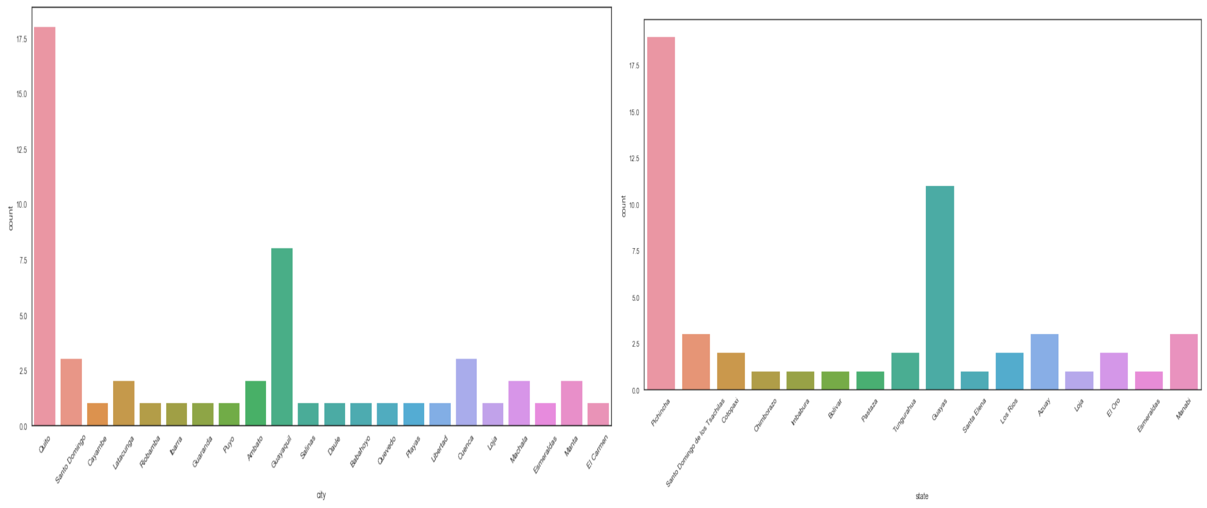


figure 1.2 geolocation distributions of stores

The types and clusters of store information are shown in figure 1.3. The difference of store numbers for different clusters is not big. It is similar for different store types.

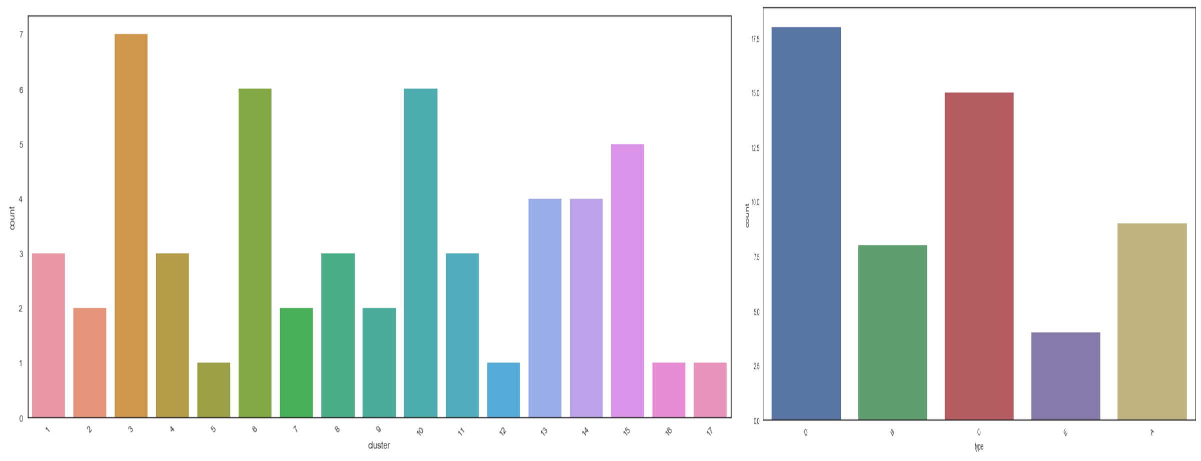


figure 1.3 types and clusters of store information

2. Item information

The coutplot of item family and perishable information are given in figure 1.4. The majority of items fall into the grocery and beverage family. And items in non-perishable categories are roughly 3 times compared with items in perishable ones.

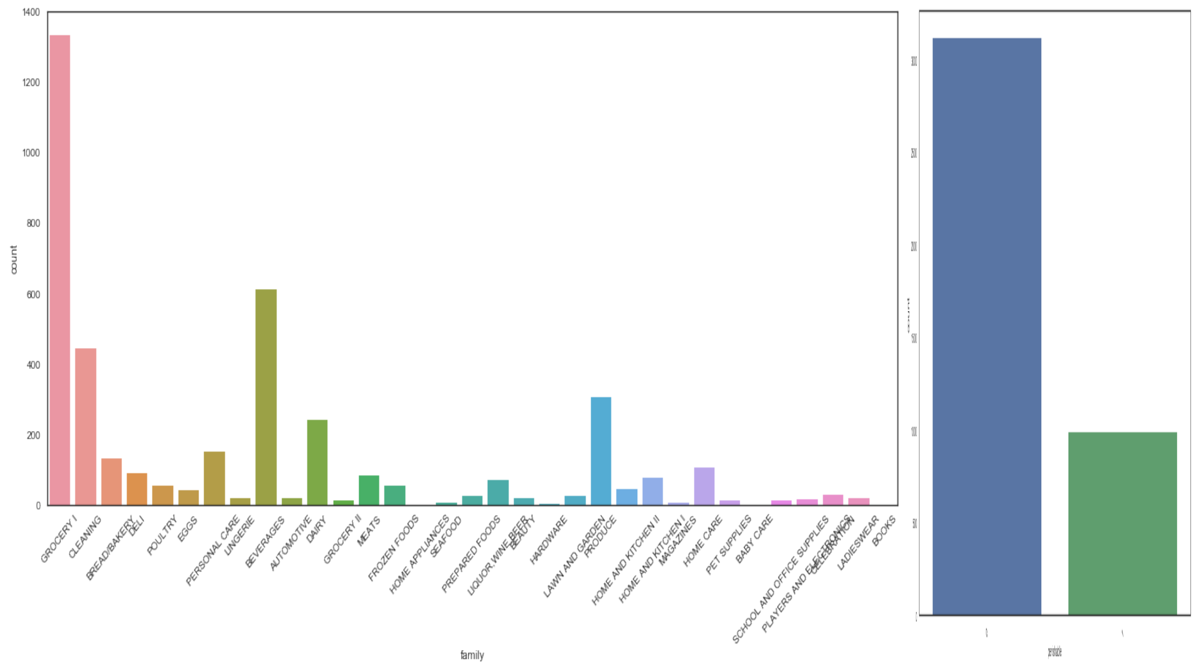


figure 1.4 count plot of item family and perishable information

3. Oil price vs total unit sales

The daily oil price and daily total unit sales are displayed in figure 1.5 and 1.6 respectively.

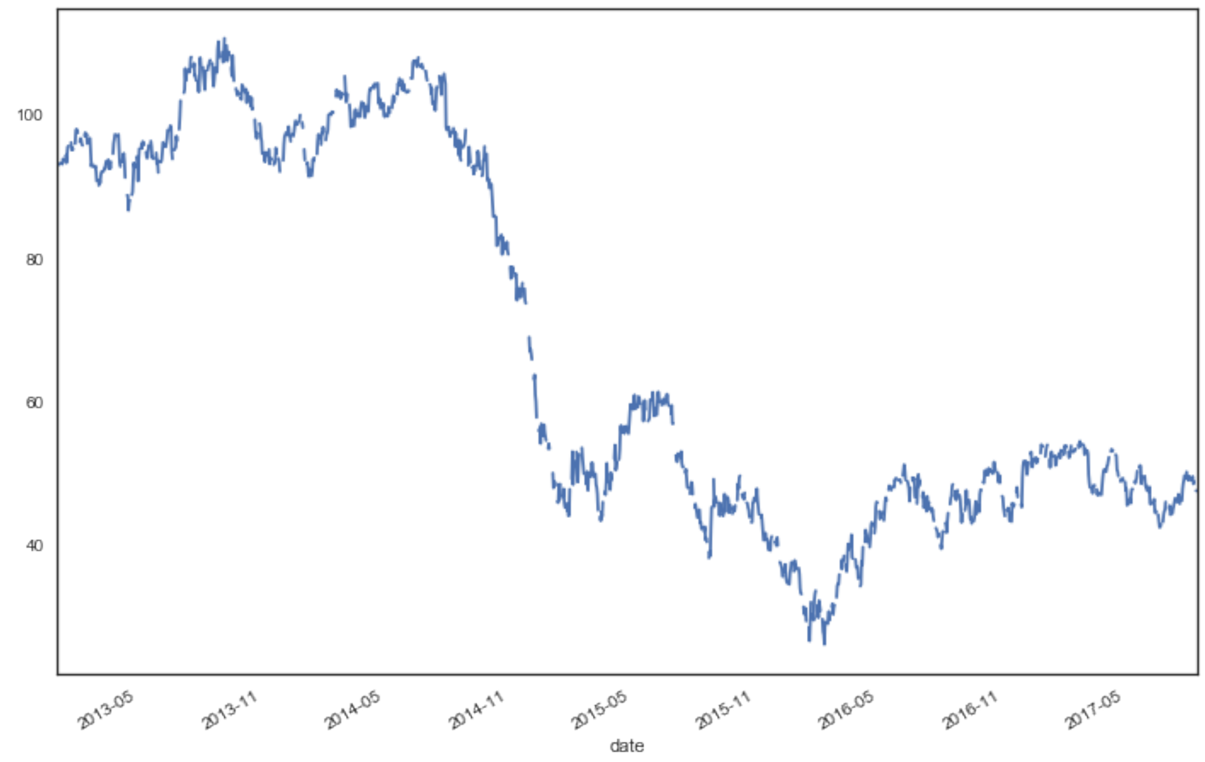


figure 1.5 daily oil price

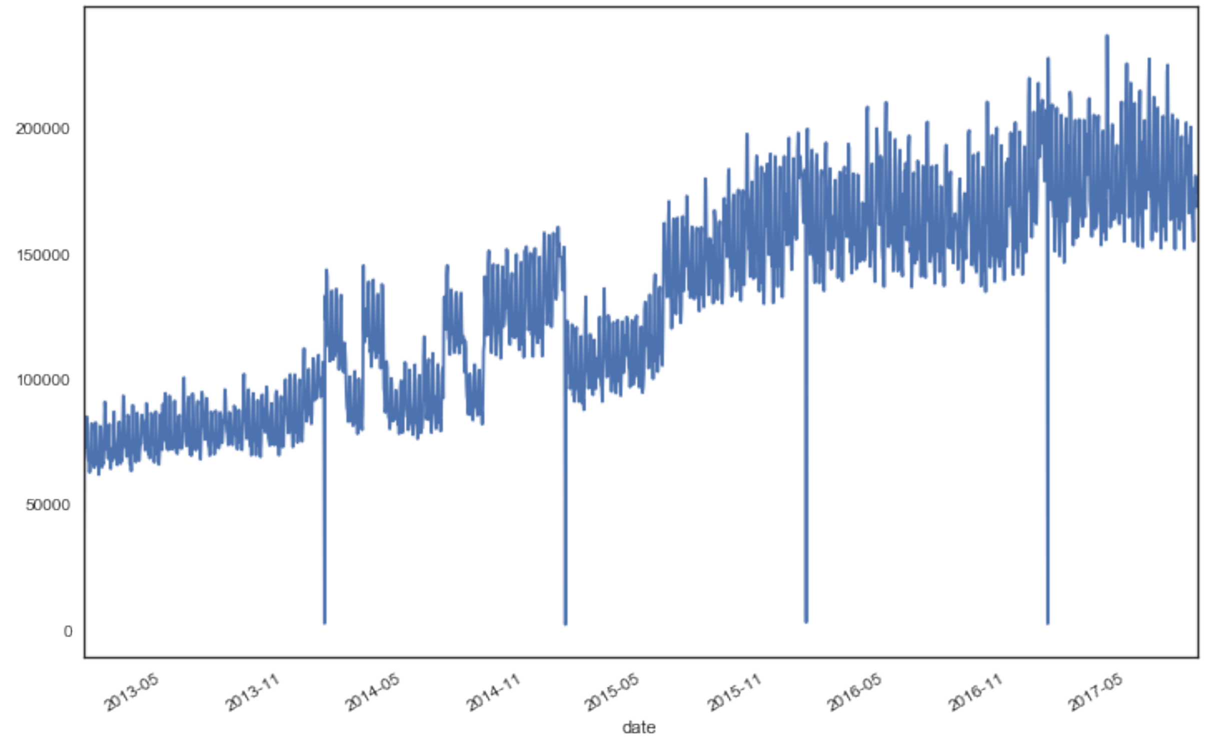


figure 1.6 daily total unit sales

The total unit sales show an increasing trend in the long term and a fluctuation in the short term. Several significant drops happened around Christmas, which was closed of stores on those days. Oppositely, the oil price demonstrates a decline trend in the long terms and a huge drop is seen at the beginning of 2015. If we look at the recent data since 2016, a growing total unit sales and oil price are both observed.

Inferential statistics

1. Is there any statistically relationship between oil price and total unit sales?

We conduct the liner regression and the summary report shows that p-value is 0, a strong evidence that we can reject the null hypothesis (oil price and total unit sales are independent each other). There is a statistical significant correlation between oil price and total unit sales.

2. Is promotion and unit sales independent each other?

We perform student t test on unit sale for the two groups which are with promotion and without promotion. The t test results (p-value = 0) can reject the hull hypothesis (promotion and unit sales are independent with each other). There is a statistical significant correlation between promotion and unit sales.

3. Is store type and cluster independent with each other and is store city and cluster in dependent with each other

We use a chi-square test of independence of variables in a contingency table. The p-value of chi-square test for store type and cluster is 0 while for store city and cluster is 0.73. As a result, we can statistically infer that there is a dependence on store type and cluster while no dependence on store city and cluster.

Data Wrangling and Feature engineering

Stage1. Feature transformation and selection

1.1Loading data with unit_sale values transformed

As the unit_sale has negative values (-1) for return items, we transform the unit_sale by log(1+x) when loading the training data.

1.2 Filling the missing values

After a missing values check, we find that it happens on the data of oil price and promotion. We just fill the oil price missing values with forward fill method and promotion missing values with 0.

1.3 National holiday date selection

We select the non- transferred national holidays dates, that is, the actually celebration dates and compare that with test data date. We find no national holiday happens in the prediction range.

1.4 Category label encoding

We transform the string-based category variables into numerical values by LabelEncoder method provided by sklearn. The transformed variables include city, state, type in store meta data; family in item meta data and locale_name in selected holiday data.

1.5 Mapping perishable and promotion features

Then, we map the boolean ‘onpromotion’ variable into 0-1 variable. The perishable feature is mapped as {0->1; 1->1.25} as required for weight in prediction.

1.6 Feature selection

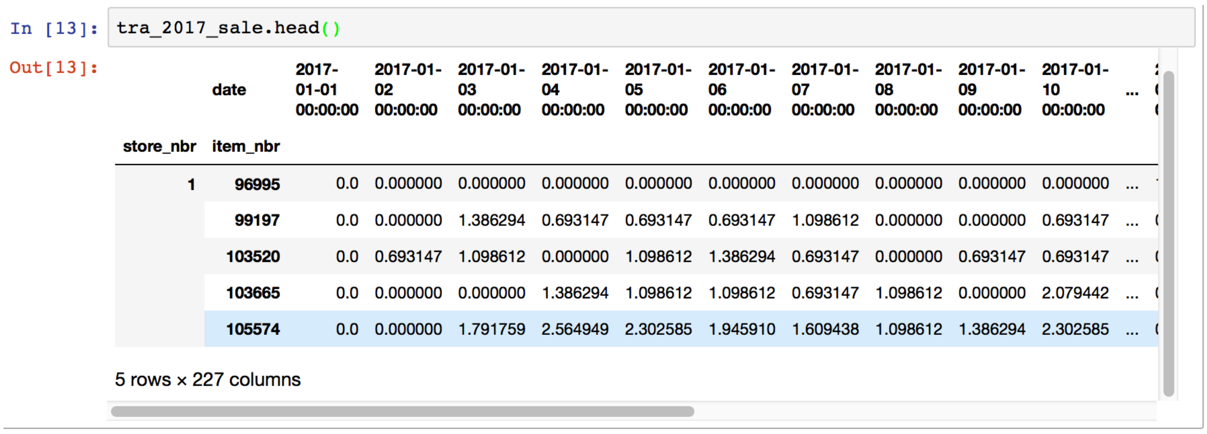
Our task in essence is time series forecasting, the data provided are since 2013 in the past. We prefer to learn from the recent past information, so choose the date from 2017 as a start point. And training data set will be the major data source for building up the model. The oil data does not provide strong evidence on influencing the short term forecasting. The holiday data indicates no big events happened in the prediction range. The store and item meta data give some information and we can use these data to create cross features to improve the predict ability of model. But for simplification, we will not consider them in the beginning.

Stage2. Create Time series feature

2.1 Data Frame reshape

We are interested in the time series data of unit_sale and promotion state for each item in each of stores. In order to create time series based features in future, we need to transform the timestamp (date) from rows to columns by set_index and unstack function in pandas.

The reshape operation is done as follows: firstly, we set the first, second and third level of index as store_nbr and item_nbr and date and choose the unit_sale or promotion as values for new dataframe. Then we unstack(transform/reshape) the last level (date) as columns and fill missing values with zero. The created new dataframes for training data are shown as follows



```
In [13]: tra_2017_sale.head()
```

Out[13]:

	date	2017-01-01 00:00:00	2017-01-02 00:00:00	2017-01-03 00:00:00	2017-01-04 00:00:00	2017-01-05 00:00:00	2017-01-06 00:00:00	2017-01-07 00:00:00	2017-01-08 00:00:00	2017-01-09 00:00:00	2017-01-10 00:00:00	...
store_nbr	item_nbr											
1	96995	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
	99197	0.0	0.000000	1.386294	0.693147	0.693147	0.693147	1.098612	0.000000	0.000000	0.693147	...
	103520	0.0	0.693147	1.098612	0.000000	1.098612	1.386294	0.693147	0.000000	0.693147	0.693147	...
	103665	0.0	0.000000	0.000000	1.386294	1.098612	1.098612	0.693147	1.098612	0.000000	2.079442	...
	105574	0.0	0.000000	1.791759	2.564949	2.302585	1.945910	1.609438	1.098612	1.386294	2.302585	...

5 rows × 227 columns

At the same time, we set the index of item meta data and store meta data the same as the reshaped training dataset by getting its first and second level of index.

We do the reshape operations on the promotion variables in the test data as well. Then we join the promotion time series data in the training and test data as a new dataframe.

2.2 Define time window functions

The time window is a period between start date and end date. We define our time window by giving the end date and period. As a result, we can get information in the past by the given time.

2.2.1 Moving statistics for days of interval
We want to create statistics features based on days of interval(period). So we define the moving statistics functions which calculate the mean, mean of difference for adjacent, decay mean, median, min, max, standard deviation within defined days of intervals.

2.2.2 Moving aggregation for days of interval
We also want to get the aggregation features based on days of interval. So we define the moving aggregation function which calculates the sum of values within defined days of intervals.

2.2.3 Select values of consecutive days
We want to know the values in a few consecutive days as well, so the function which selects the values of consecutive days is also given.

2.3 Create time series features based on the defined time window functions.

We conduct following procedures for unti_sale time series data and promotion time series data to create input features (X)

In the unti_sale time series data, we generate features from defined date as follows:

1. Moving statistics (mean, mean of difference for adjacent, decay mean, median, min, max, standard deviation) of unit_sale in 3, 7, 14, 30, 60, 140 past days of interval
2. Moving aggregation (sum) of unit_sale in 3, 7, 14, 30, 60, 140 past days of interval
3. Unit_sale in 16 past consecutive days (where 16 is the prediction date range)

In the promotion time series data, we generate features from defined date as follow:
1. Moving aggregation (sum) of promotion state in 14, 60, 140 past days of interval
2. Moving aggregation (sum) of promotion state in 3, 7, 14 future days of interval
3. Promotion state in 16 past consecutive days (where 16 is the prediction date range)
4. Promotion state in 16 future consecutive days (where 16 is the prediction date range)

Then, we create labels (y) by selecting unit_sale in 16 future consecutive days (where 16 is the prediction date range)

Machine Learning models

1.1 Prepare training, validation and test dataset

After define the time series input and label features, we can prepare our training, validation and test dataset by giving specific date.

We prepare four folders of training date set with dates on 2017-06-14, 2017-06-21, 2017-06-28, 2017-07-05(a week interval for chosen dates) separately and then concatenate them together.
For validate date set, we choose the date as 2017-07-26.

Finally, we prepare the test features (X_test) by specifying date as 2017-08-16.

The data shape in our training, validation and test dataset is 670060*144, 167515*144, 167515*144 respectively

1.2 Different algorithms for prediction

We train 16 models and each of model select one day's unit sales as label for training, then predict correspondent unit sales in the 16 future consecutives days.
The performance is measured by the Normalized Weighted Root Mean Squared Logarithmic Error, where the weight is perishability of items.

The model performance is given in the following tables

Model	Performance/Error
Liner regression models_Lasso	0.3370012403947538
Liner regression models_Ridge	0.32294528746799755
Support Vector Machine	0.32922112263898434
Extra Trees	0.33125629069949236
Random Forest	0.33167082556360583

1.3 Light GBM algorithm for prediction

We select the powerful light GBM algorithms, which is fast and more accurate. We train 16 light GBM models and each of model select one day's unit sales as label for training, then predict correspondent unit sales in the 16 future consecutives days.

The parameters for an optimal model is given as

parameters = {

```
   'num_leaves': 100,
   'objective': 'regression',
   'max_depth': 6,
   'min_data_in_leaf': 200,
   'learning_rate': 0.05,
   'feature_fraction': 0.2,
   'bagging_fraction': 0.4,
   'bagging_freq': 1,
   'metric': 'l2',
   'num_threads': 16,
    'max_bin': 127
}
```

The model performance is `0.3142811400188674`

We also submit our light GBM model and the error in the test dataset is 0.515. It ranks within 10% in the Kaggle leader board.