FORECASTING AND ANALYSIS OF SALES TRANSACTION DATA

Project Report

Machine Learning

Developed By:

MEHUL GUPTA

ABSTRACT

Sales forecasting is the process of estimating future sales. Accurate sales forecasts enable companies to make informed business decisions and predict short-term and long-term performance. Companies can base their forecasts on past sales data, industry-wide comparisons, and economic trends. An important example of time series data is the sales transaction data. At present time different time series theories and models have been developed like Holt-Winters model, ARMA, PROPHET, SARIMA, SARIMAX, GARCH, etc. A popular and widely used statistical method for time series forecasting is the ARMA model.

Time series forecasting is the use of a model to predict future values based on previously observed values. Time series are widely used for non-stationary data, like economic, weather, stock price, and retail sales. Released by Facebook in 2017, forecasting tool Prophet is designed for analysing time-series that display patterns on different time scales such as yearly, weekly and daily. It also has advanced capabilities for modelling the effects of trends, and seasonality on a time-series and implementing custom changepoints. We will demonstrate different approaches for forecasting retail sales time series.

Our aim in this project is to predict and forecast the future values of sales by using the given values of sales (i.e.) the previous values of sales from some previous time and date. We shall also validate our model to obtain the best model and shall do the comparison for the same using ARMA and fbProphet.

ARCHITECTURE:

The architecture of this project is as follows:

a) Loading the dataset
b) Cleanse the dataset
c) Data pre-processing
d) Remove outliers
e) Processing noisy data
f) Checking whether the data is stationary or not
g) Validating the model using train-test split ratios
h) Visualizing the models
i) Predicting value of total sales of all stores per month for next 5 months
j) Calculate accuracy using accuracy measures
k) Filter the best possible solution
l) Comparing the model using defined metrices.

End Users are: store owners and managers

Objective: To predict the amount of sales that would take place using the given amount of sales from previous years.

Tools used:

- Anaconda IPython cluster, jupyter notebook, libraries like:
- Sklearn
- Pandas
- Numpy
- Iter-tools
- matplotlib
- seaborn
- scipy etc.

Dataset description:

The dataset was taken from UCI machine learning repository, which was further customised according to our needs in the project by splitting the attributes. The attribute description of the dataset is given below:

- ID - an Id that represents a (Shop, Item) tuple within the test set
- 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
- item_category_name - name of item category
- s0-s51-51 different shops
- max-max item sold
- min-min item sold

The dataset contains about 29,35,849 rows which is very huge to process. It represents the sales of each product day-wise on each store.

Hence, there are around 22170 items and 60 stores in our dataset, which calls for about million timeseries.

ALGORITHM DESCRIPTION

1) AR (Autoregression): A model that uses the dependent relationship between an observation and some number of lagged observations. p is a parameter of how many lagged observations to be taken in. AR model can be defined using the formula.

$$x_t = \alpha_1 x_{t-1} + \ldots \alpha_p x_{t-p} + z_t, \text{ or}$$

$$x_t = \sum_{i=1}^{p} \alpha_i x_{i-1} + z_t$$

2) I (Integrated): A model that uses the differencing of raw observations (e.g. subtracting an observation from the previous time step). Differencing in statistics is a transformation applied to time-series data in order to make it stationary. This allows the properties do not depend on the time of observation, eliminating trend and seasonality and stabilizing the mean of the time series.

3) MA (Moving Average): A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations. q is a parameter of how many lagged observations to be taken in. MA model can be defined using the formula below:

$$x_t = \beta_0 z_t + \beta_1 z_{t-1} + \ldots \beta_q z_{t-q}, \text{ or}$$

$$x_t = \sum_{i=0}^{q} \beta_i z_{t-i}$$

4) Autoregressive–moving-average (ARMA) model: provide a parsimonious description of a (weakly) stationary stochastic process in terms of two polynomials, one for the autoregression (AR) and the second for the moving average (MA).

Given a time series of data X_t, the ARMA model is a tool for understanding and, perhaps, predicting future values in this series. The AR part involves regressing the variable on its own lagged (i.e., past) values. The MA part involves modelling the error term as a linear combination of error terms occurring contemporaneously and at various times in the past. The model is usually referred to as the ARMA(p,q) model where p is the order of the AR part and q is the order of the MA part (as defined below).

$$x_t = \alpha_1 x_{t-1} + \ldots \alpha_p x_{t-p} + z_t + \beta_1 z_{t-1} + \ldots . \beta_q z_{t-q}$$

Hence, ARMA (1,1) model is:

x(t) = a*x(t-1) + b*e(t-1) + e(t)

where e(t) is white noise with E[e(t)] = 0

5) FbProphet: 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.

Prophet is open source software released by Facebook's Core Data Science team. It is available for download on CRAN and PyPI.

NEED OF A NEW MODEL

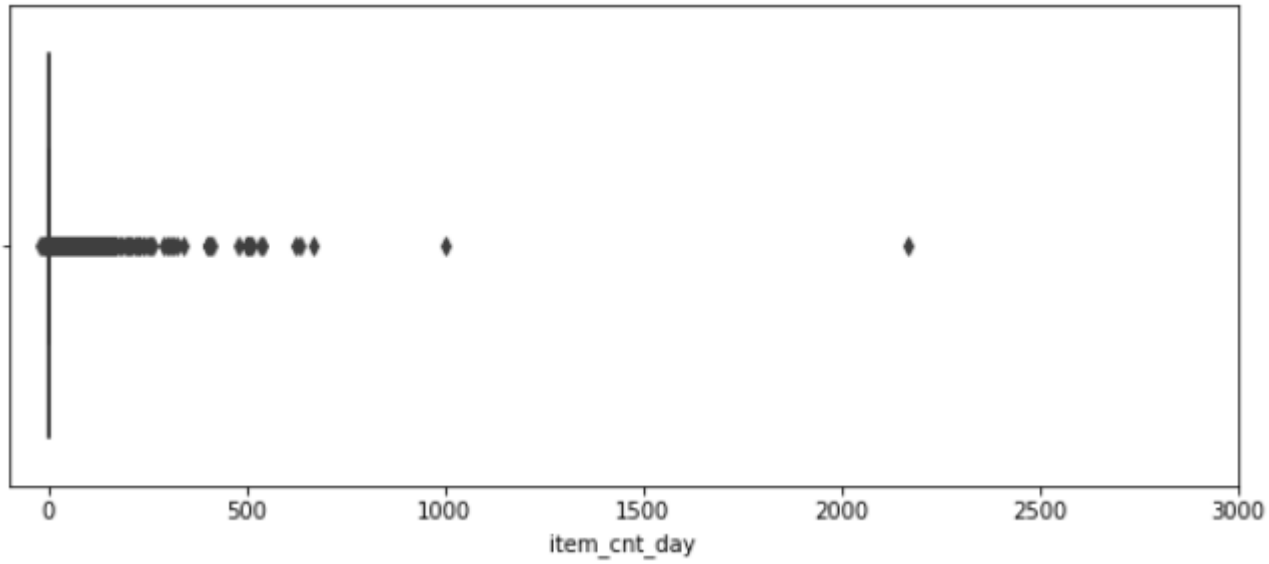
EXISTING SOLUTION

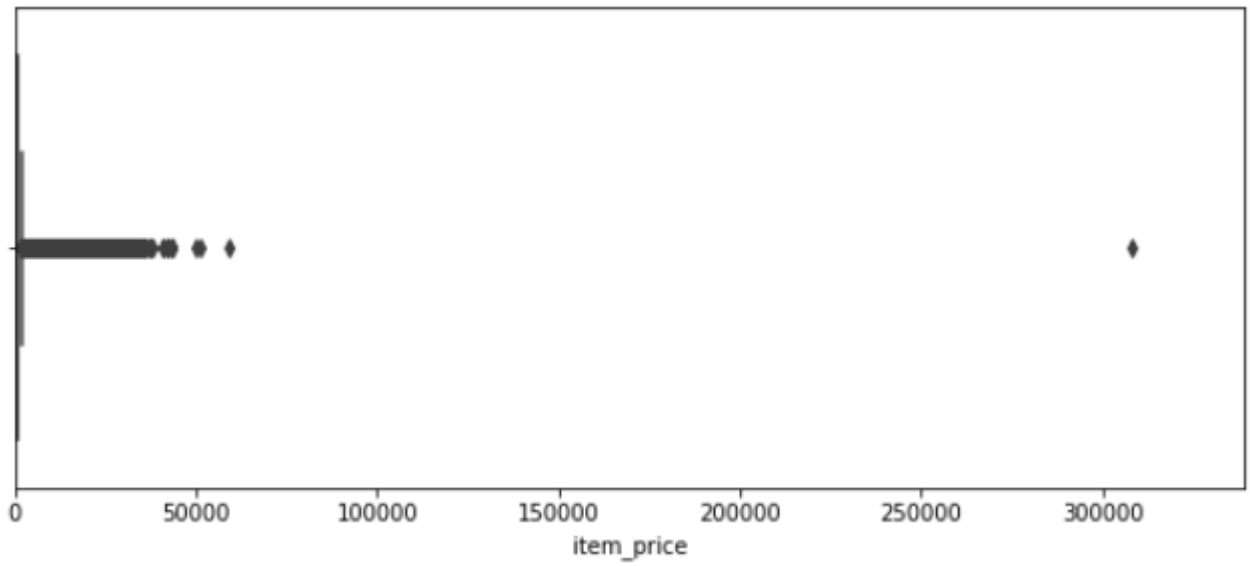
Existing solution uses simple linear regression and ARIMA based models or deep learning models to predict sales with **doesn't take into account the trends and seasonality of the sales**. But considering the temporal nature of data I proposed my own solution and hampered with the data to produce features that will improve the accuracy of the proposed models. After analysing my models and comparing the Mean absolute percentage error, I found that my proposed model is better because the proposed model takes into account the trends and seasonality in the sales while the linear regression performs less accurately and neural networks results in overfitting and also computational cost is too high.

MATHEMATICAL SOLUTION

Outlier Removal

We can see from the graph below, there are items with strange prices and sales. After detailed exploration I decided to remove items with price > 100000 and sales > 1001 with the given values taken as threshold for convenience.






```
In [5]: train = train[train.item_price<100000]
        train = train[train.item_cnt_day<1001]
```

There is one item with price below zero. Filling it with median

```
median = sales[(sales.shop_id==32)&(sales.item_id==2973)&(sales.date_block_num==4)&(sales.item_price>0)].item_price.median()
sales.loc[sales.item_price<0, 'item_price'] = median
```

Removing rows with item count<0

```
In [10]: sales=sales[sales['item_cnt_day']>0]
```

Converting raw string to date-time format as accepted as input by Prophet:

```
#formatting the date column correctly
sales.date=sales.date.apply(lambda x:datetime.datetime.strptime(x, '%d.%m.%Y'))
# check
print(sales.info())
```

MATHEMATICAL FORMULATION:

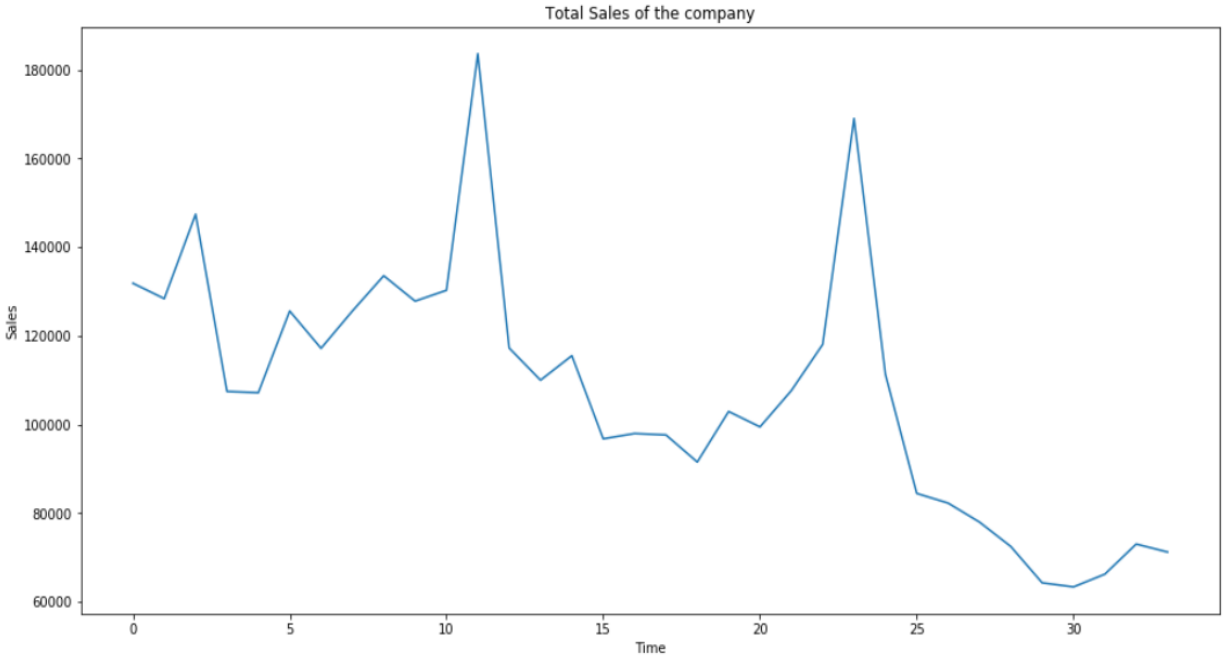
Single series:

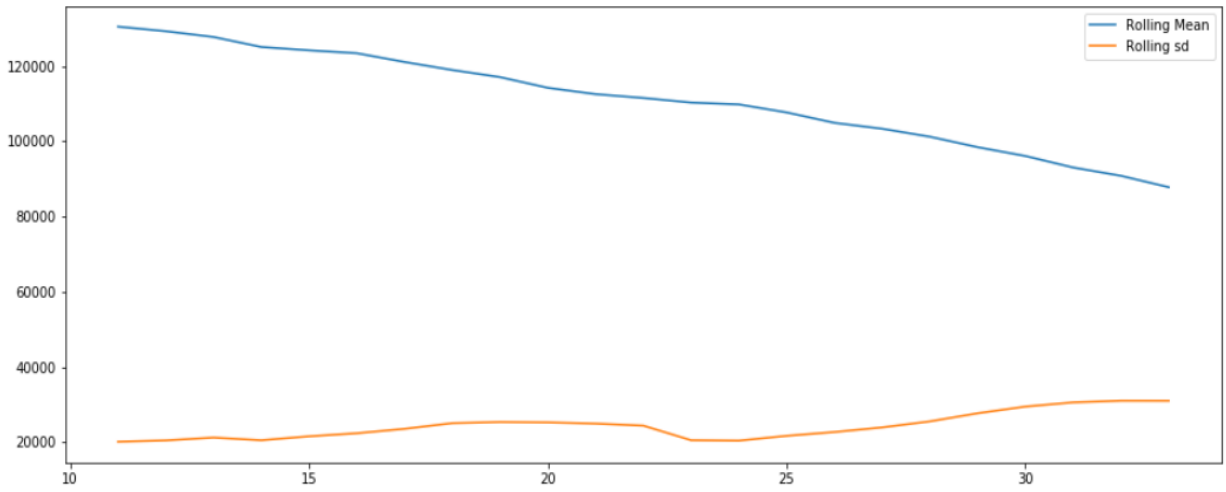
The objective requires us to predict sales for the next month at a store-item combination.

Sales over time of each store-item is a time-series in itself. Before we dive into all the combinations, first let's understand how to forecast for a single series.

I've chosen to predict for the total sales per month for the entire company.

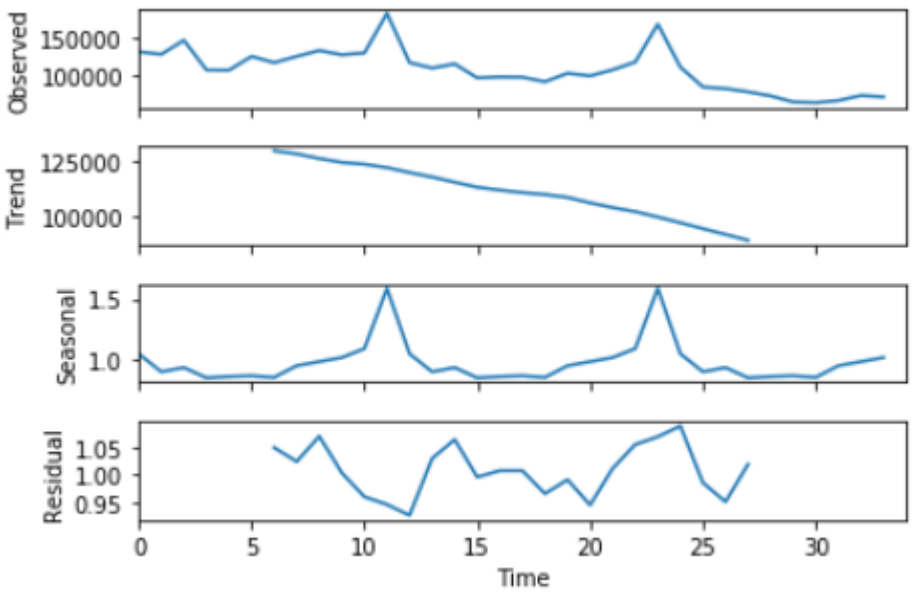
First let's compute the total sales per month and plot that data.





Quick observations: There is an obvious "seasonality" (Eg: peak sales around a time of year) and a decreasing "Trend".

Let's check that with a quick decomposition into Trend, seasonality and residuals.



Stationarity:

There are multiple tests that can be used to check stationarity.

1. ADF(Augmented Dicky Fuller Test)
2. KPSS
3. PP (Phillips-Perron test)

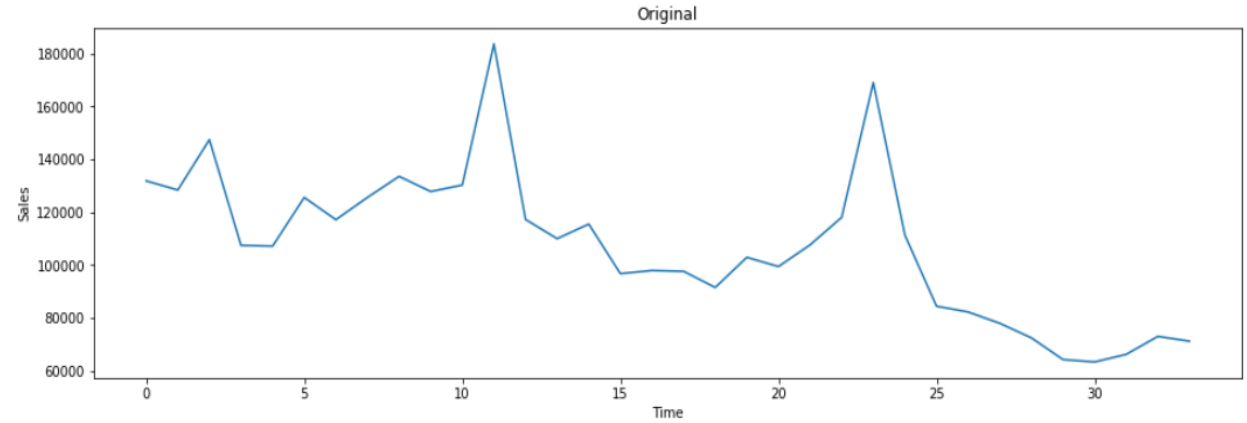
We have performed the ADF which is the most commonly used one.

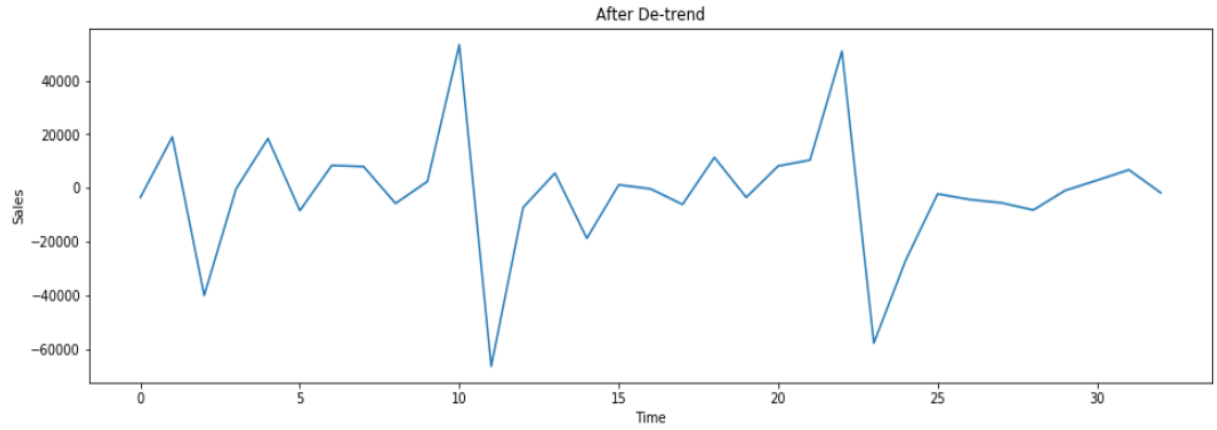
ADF test:

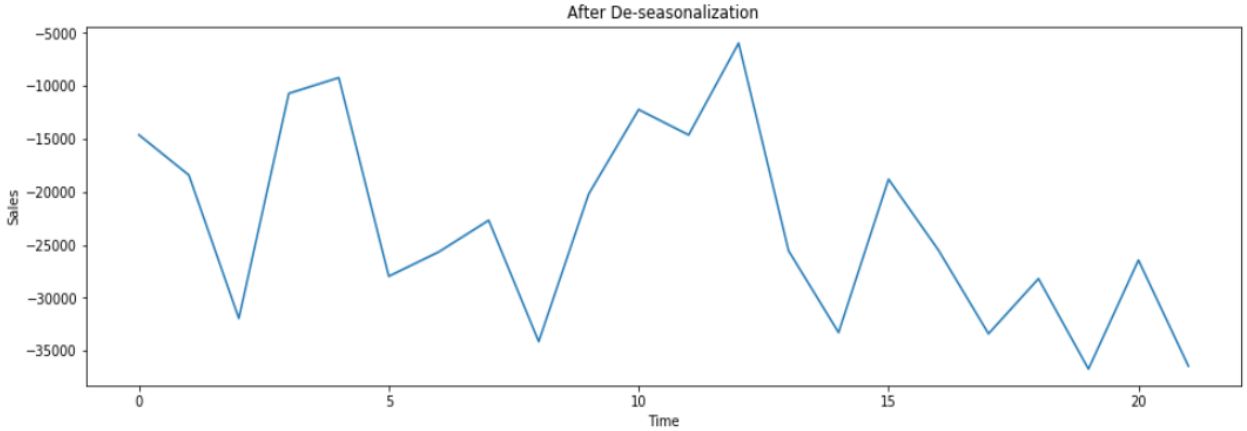
Augmented Dickey–Fuller test (ADF) tests the null hypothesis that a unit root is present in a time series sample. The alternative hypothesis is different depending on which version of the test is used, but is usually stationarity or trend-stationarity. It is an augmented version of the Dickey–Fuller test for a larger and more complicated set of time series models.

The augmented Dickey–Fuller (ADF) statistic, used in the test, is a negative number. The more negative it is, the stronger the rejection of the hypothesis that there is a unit root at some level of confidence.

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t$$







Now after the transformations, we can assume Stationarity of the series.

Validating the model using train-test split:

1) Setting the split ratio to 66%:

```
train_size = int(len(ts)*0.66)
train, test = ts[0:train_size], ts[train_size:len(ts)]
test.reset_index(drop=True,inplace=True)
train_newts=difference(train,12)       # assuming the seasonality is 12 months long
```

Calculating the Mean absolute percentage error:

```
import numpy as np

def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100
```

```
mean_absolute_percentage_error(predictions,test[:10])
```

```
10.155058581947111
```

2) Setting the split ratio to 75%:

```
train_size = int(len(ts)*0.75)
train, test = ts[0:train_size], ts[train_size:len(ts)]
test.reset_index(drop=True,inplace=True)
train_newts=difference(train,12)       # assuming the seasonality is 12 months long
```

Calculating the Mean absolute percentage error:

```
mean_absolute_percentage_error(predictions[:9],test)
```

```
29.251927827355196
```

Methods like k-fold cross validation cannot be used here to validate the data. This is because they assume that there is no relationship between the observations, that each observation is independent. This is not true of time series data, where the time dimension of observations means that we cannot randomly split them into groups. Instead, we must split data up and respect the temporal order in which values were observed.

In time series forecasting, this evaluation of models on historical data is called backtesting. In some time-series domains, such as meteorology, this is called hindcasting, as opposed to forecasting.

Analysis of above results of train-test splits:

As evident from the mean-absolute percentage error metrix above, if we change the split ratio, the mean-absolute percentage error will increase, hence we can assume the best split ratio as 66%.

Mean-absolute percentage error: The MAPE (Mean Absolute Percent Error) measures the size of the error in percentage terms. It is calculated as the average of the unsigned percentage error:

$$\left(\frac{1}{n}\sum\frac{|Actual - Forecast|}{|Actual|}\right)*100$$

MODEL COMPARISON

AR Model v/s MR Model

AR (p) models try to explain the momentum and mean reversion effects often observed in trading markets (market participant effects).

MA(q) models try to capture the shock effects observed in the white noise terms. These shock effects could be thought of as unexpected events affecting the observation process e.g. Surprise earnings, wars, attacks, etc.

The notation AR(p) refers to the autoregressive model of order p. The AR(p) model is written

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t.$$

The notation MA(q) refers to the moving average model of order q:

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{} \theta_i \varepsilon_{t-i}$$

The notation ARMA (p, q) refers to the model with p autoregressive terms and q moving-average terms. This model contains the AR(p) and MA(q) models,

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

Visualizing the ARMA model that we get from the test series:

Results: ARMA

Model:	ARMA	BIC:	476.2702
Dependent Variable:	y	Log-Likelihood:	-233.50
Date:	2019-04-08 02:05	Scale:	1.0000
No. Observations:	22	Method:	mle
Df Model:	2	Sample:	0
Df Residuals:	20		2
Converged:	1.0000	S.D. of innovations:	9237.088
No. Iterations:	9.0000	HQIC:	473.768
AIC:	472.9970		

	Coef.	Std.Err.	t	P>|t|	[0.025	0.975]
ar.L1.y	0.9951	0.0095	104.4153	0.0000	0.9764	1.0138
ma.L1.y	-0.7296	0.1917	-3.8064	0.0011	-1.1053	-0.3539

	Real	Imaginary	Modulus	Frequency
AR.1	1.0049	0.0000	1.0049	0.0000
MA.1	1.3706	0.0000	1.3706	0.0000

PREDICTING FUTURE VALUES

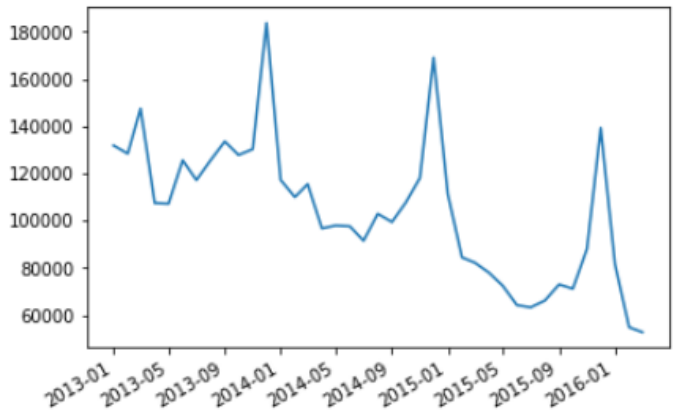
Next part of this project will be to some-how estimate the future values based on current values.

Approach I used for this is, timestamp based prediction. I partitioned the data based on temporal ordering and converted the dates into time stamps values. Based on time stamps, I calculated number of seconds in a year i.e. 60*60*24*365 (number of seconds in a year).

Then I added it into my time stamps in order to generate values for next year. I kept on calculating for subsequent years and finally got a graph based series as an estimator.

Predicted value of total sales of all stores per month for **next 5 months** starting from **01-11-2015** (**November, 2015**)

```
predicted=88089.245521
predicted=139239.818524
predicted=81613.972735
predicted=54852.516843
predicted=52775.920033
```



Forecasting for long periods causes the time series to lose its stationarity and leads to convergence.

Handling problem with multiple times series:

The problem at hand here, has 22170 items and 60 stores. This indicates that there can be around a million-individual time-series (item-store combinations) that we need to predict!

Configuring each of them would be nearly impossible. Let's use Prophet which does it for us.

Starting off with the bottoms up approach. 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.

First, of all we validate our model to find out if it is the best fit to use according to the above metrics followed of Mean absolute percentage error:

Validating the fbprophet model using the same train-test split ratio i.e. 66%

```
train_size = int(len(ts)*0.66)
train, test = ts[0:train_size], ts[train_size:len(ts)]
i=len(train_newts)
train.index=pd.date_range(start = '2013-01-01',periods=len(train), freq = 'MS')
train=train.reset_index()
train.head()
```

The mean absolute percentage error that we get is: 10.184493783, which is slightly greater than the mean absolute percentage error of ARMA model i.e., 10.15505858, at the same train-test split ratio of 66%.

```
# ((pred-test[:10])**2).mean()**.5
mean_absolute_percentage_error(pred,test[:10])
```

```
10.184493783024932
```

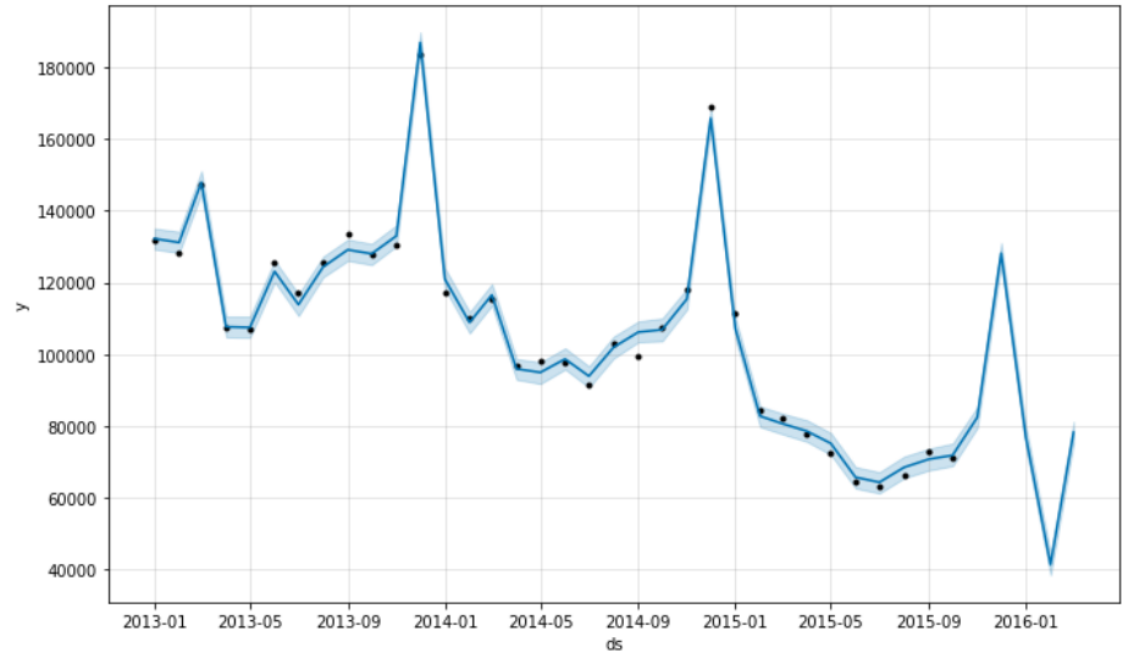
Hence, we can infer that the ARMA model of order (3,0) performs slightly better than the fbProphet model.

Using fbProphet model to forecast sales:

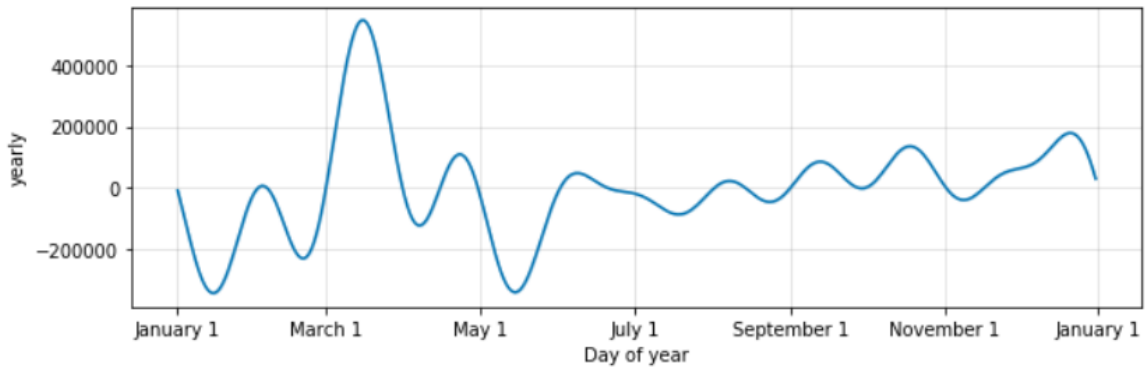
From the graph below, we can infer that the fbProphet model takes care of the trends and seasonality in our time series, which is evident from the sharp peaks we obtain here.

The existing sales are plotted with black dots, and the predicted sales for the next 5 months is plotted starting from November, 2015 (2015-11).

```
1 model.plot(forecast);
```



Plotting the yearly seasonality of the dataset using fbProphet, we get the yearly trend in the data as follows:



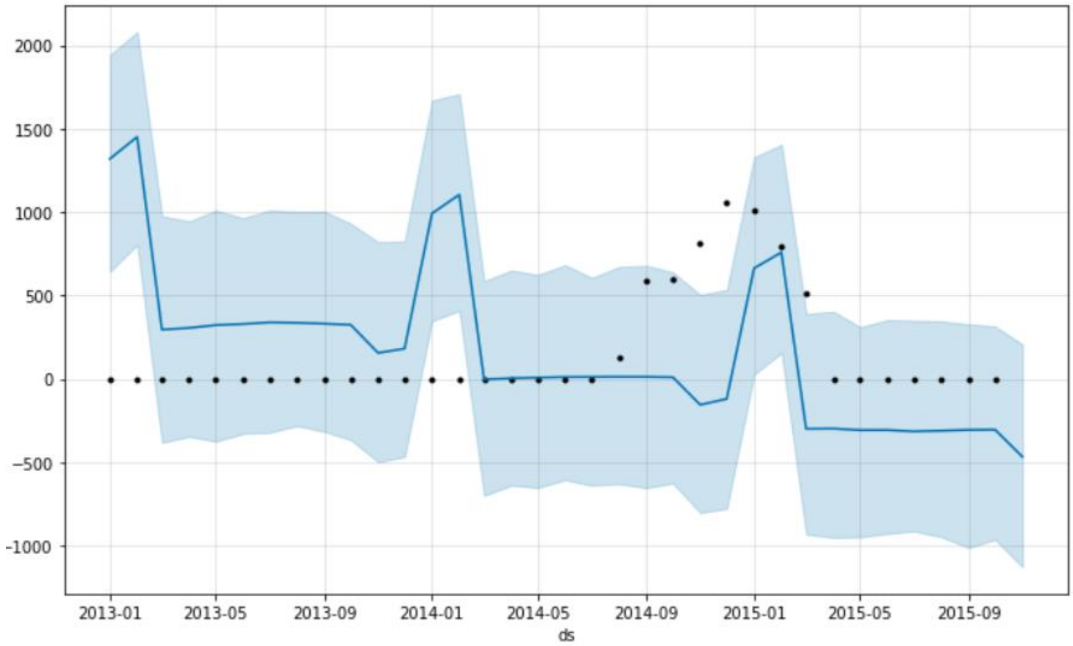
Now, using fbProphet to calculate individual time-series of each product:

Calculating individual time-series of first few products:

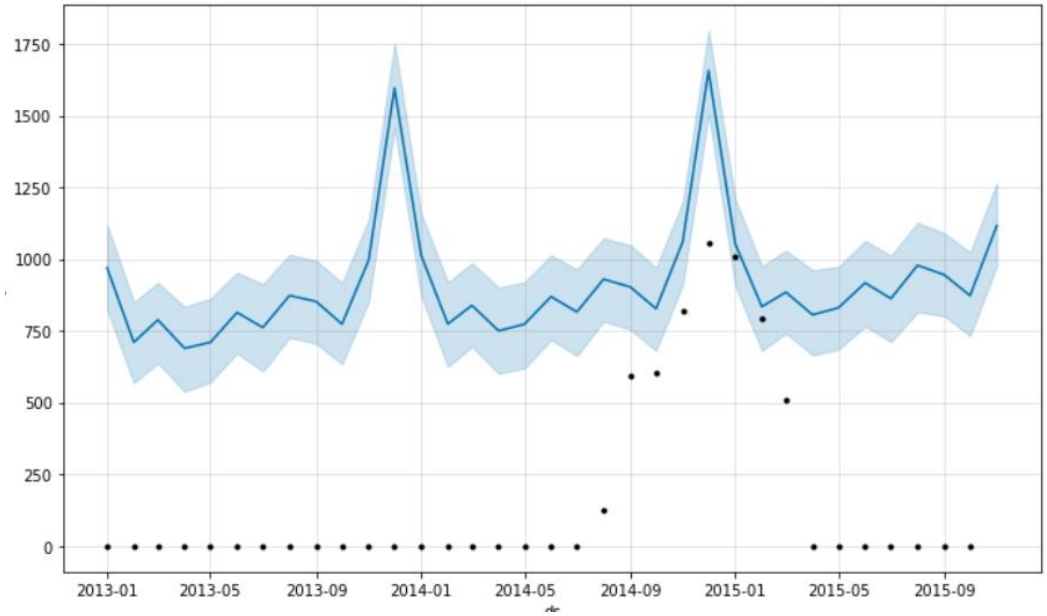
The calculation of forecasting sales of individual products is done by first grouping the items according to item_id and month and then fitting the model by altering the weekly and monthly seasonality, whichever fits best.

```
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
```

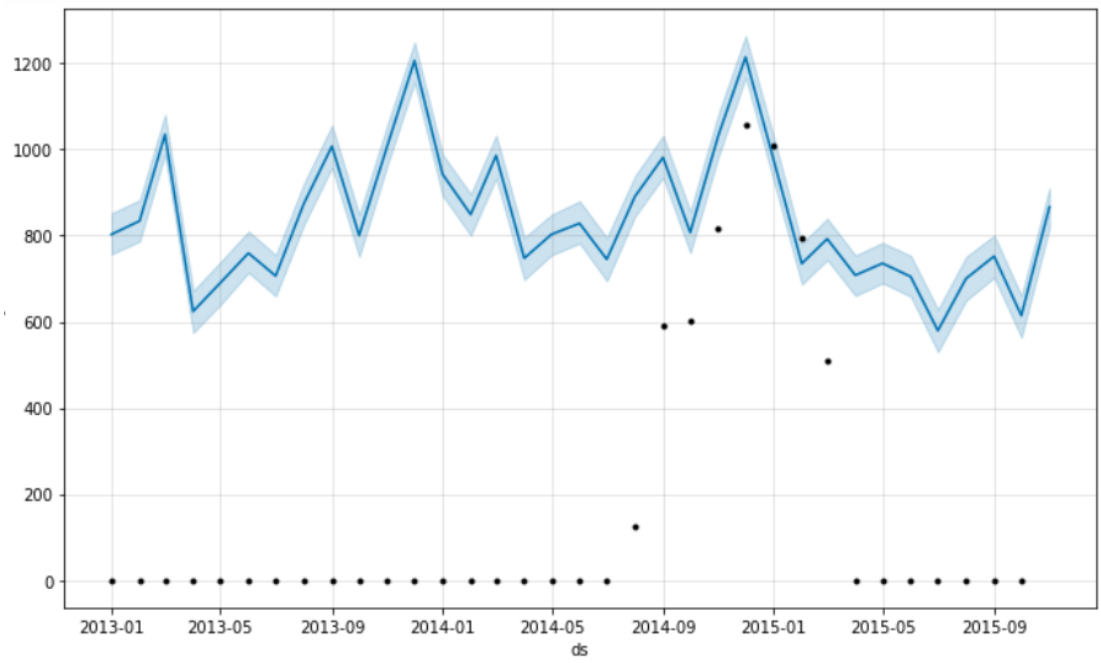
Plotting the predicted sales of first few products, we get the following plots using fbProphet:



Item_id:1



Item_id:2



Item_id:3

Item_id:4

The important idea in Prophet is that by doing a better job of fitting the trend component very flexibly, we more accurately model seasonality and the result is a more accurate forecast. We prefer to use a very flexible regression model (somewhat like curve-fitting) instead of a traditional time series model for this task because it gives us more modelling flexibility, makes it easier to fit the model, and handles missing data or outliers more gracefully.

By default, Prophet will provide uncertainty intervals for the trend component by simulating future trend changes to your time series. If we wish to model uncertainty about future seasonality or holiday effects, we can run a few hundred HMC iterations (which takes a few minutes) and our forecasts will include seasonal uncertainty estimates.

RESULTS

In this project, we predicted and forecasted the future sales by using the given values of sales (i.e.) the previous values of sales from some previous time and date using ARMA model and FbProphet model. We also validated our model to obtain the best trained model. Compared to ARMA, other methods (e.g. ES, Holt's, Regression) can be easily implemented. In this consideration, ARMA may be attractive if it shows significantly advantage in improving forecasting accuracy compared to other methods. If ARMA only shows slightly better result, companies may still prefer simple methods (e.g. ES, Holt's, Regression) because they are easy to implement.

The complete analysis of the accuracy metrices used to compare both the ARMA and FbProphet model is summarized in the table below:

Model	Mean-absolute percentage error
1) ARMA (split ratio=66%)	**10.15505858**
2) ARMA (split ratio=75%)	29.25192782
3) ARMA (split ratio=80%)	11.73101579
4) FbProphet (split ratio=66%)	**10.184493783**

CONCLUSION

The function of time series combination model of sales data in this thesis can have a good reflection in the development of the rules of monthly sales in marketing. This kind of model can be used for forecast that the company concerned and have the number of sales in order to make appropriate and timely decision-making. Hence, we successfully used ARMA Algorithm to predict the number of sales which will happen in future.

FUTURE WORK:

Though FbProphet model and ARMA model works well with only a marginal error of 10%, there is still a scope of reducing this error and forecasting the results with a better accuracy. This may be done by combining the two models and present a hybrid model of the same. We shall also extend the scope of this project to predict time-series even if some products are removed from a store, such that its time-series does not affect the compete time-series of the product.

REFERENCES:

[1] Sales Forecasting Using Neural Networks Frank M. Thiesing and Oliver Vornberger Department of Mathematics and Computer Science University of Osnabriick D-49069 Osnabruck, Germany frank@informat,ik.uni-osnabrueck.de

[2] Discovering Similar Time-Series Patterns with Fuzzy Clustering and DTW Methods' Guoqing Chen, Qiang Wei School of Economics and Management, Tsinghua University University of Pennsylvania Beijing, 100084, China U.S.A. {chengq, weiq} @em.tsinghua.edu.cn

[3] A New Algorithm for Segmenting Data from Time Series Stephen R. Duncan Control Systems Centre UMIST Greyham F. Bryant IRC for Process Systems Engineering Imperial College of Science, Technology and Medicine P.O. Box 88 Manchester M60 1QD U.K. ernail stephen.duncan@umist .ac.uk

[4] New Diffusion Model to Forecast New Products for Realizing Early Decision on Production, Sales, and Inventory Satoshi Munakata and Masaru Tezuka Hitachi East Japan Solutions, Ltd.{munakata, tezuka}@hitachi-to.co.jp

[5] Mining influence correlation of commodity category based on sale data time series Xueyu Geng School of Civil Engineering Qingdao Technological University Qingdao, 266033, China gengxy@gmail.com Jinlong Wang∗School of Computer Engineering Qingdao Technological University Qingdao, 266033, China wangjinlong@gmail.com

[6] Panel Cointegration Modeling of Electricity Consumption and Sales Price Xiusong Gong College of Economy and Trade, Hunan University, Changsha 410082, China E-mail: laishaoyuan@hotmail.com Enfeng He, Hongming Yang College of Mathematics and Computing Science, Changsha University of Science and Technology, Changsha 410076, China E-mail: heenfeng@163.com

[7] The Study of Cluster Predication Method on Sales Forecast Based on Residual Error Modified GM (1, 1)

[8] An Analysis of Shopping Basket in a Supermarket Jiangping Chen, Xiaoxian Yang, Lihua Chen, Li Dong, Yuanyan Fu

[9] Double Trends Time Series Forecasting Using a Combined ARIMA and GMDH Model Aiyun Zheng 1, Weimin Liu 1, Fanggeng Zhao2

[10] Intelligent time series fast forecasting for fashion sales: a research agenda tsan-ming choi, chi-leung hui, yong yu

[11] Time Series Analysis of Beijing's Total Retail Sales of Social Consumer Goods BoCao

[12] Development of Sales Management Support System for Agricultural Produce Using Sales Forecasting Model Toshifumi Uetake Department of Software

[13] Evolving Fuzzy Linear Regression Tree Approach for Forecasting Sales Volume of Petroleum Products Andre Lemos, Daniel Leite, Leandro Maciel, Rosangela Ballini, Walmir Caminhas, and Fernando Gomide

[14] A Prediction Study on E-commerce Sales Based on Structure Time Series Model and Web Search Data Dai Wei1, Peng Geng1, Liu Ying1, Li Shuaipeng1

[15] Detection of Irregularities and Rips by Finding Critical Points of Morse Theory– Preliminary Results on Analysis on Sales Data–SATO Hiroyuki∗, SEINO Yoshihiro∗, OGATA Takanori†, SHIRAKAWA Tatsuya†,

[16] Forecasting Monthly Sales Retail Time Series: A Case Study Giuseppe Nunnari Dipartimento di Ingegneria Elettrica Elettronica e Informatica

[17] Mining the past to determine the future market: sales forecasting using tsdm framework Angelique D. Lacasandile Jasmin D. Niguidula Jonathan M. Caballero National University T.I.P. T.I.P. adlacasandile@national-u.edu.ph

[18] Sales forecast in an IT company using time series Pedro Sobreiro ESDRM, Instituto Politécnico de Santarém; ISLA Santarém; CEPESE

[19] Forecasting of sales by using fusion of Machine Learning techniques Mohit Gurnani , Yogesh Korkey, Prachi Shahz, Sandeep Udmalex, Vijay Sambhe{, and Sunil Bhirudk

[20] Product Popularity Modeling via Time Series Embedding Santosh K C University of Houston skc@uh.edu Sohan De Sarkar

[21] Kalman Filter Based Time Series Prediction of Cake Factory Daily Sale Jiaxuan Wu Qing Fang Yangying Xu Jionglong Su Fei Ma

[22] Do Sales Promotions Affect Dynamic Changes in Sales Outcomes: Estimation of Dynamic State of Product Sales Yuta Kaneko RISS, Data Science Laboratory