Ensemble Prediction: Computing Accuracy Using Bootstrapping And Bagging Model

Aridaman Singh Nandan, Sanjiu Raja Brahma, and Dr. Nonita Sharma

Department of Computer Science and Engineering, Dr. B R Ambedkar National
Institute of Technology, Jalandhar, Punjab
aridamansn.cs.18@nitj.ac.in
sanjiurb.cs.18@nitj.ac.in
nsnonita@gmail.com

Abstract. Forecasting models are of many types. Difference lies in the way of generation of forecast, amount of data they use and their complexity. In this paper we have analysed some forecasting methods that could effectively fit on time series to capture the data. Our dataset consist of FCI wheat stock of Punjab region. We applied forecasting techniques to this data to get some predictions and then improved the accuracy of models using ensemble methods: bootstrapping and bagging. To evaluate forecast accuracy and to compare among different models fitted to time series, we have used mean percentage error as a performance measure.

Keywords: Time series · Forecasting · Ensemble Methods · Bootstrapping · Bagging.

1 Introduction

The aim of time series modelling is to collect the past observations and develop a model which describes the structure. The model should be properly and carefully chosen to fit the underlying time series. Efforts have been done for developing efficient models to improve accuracy of forecasting. Most popular time series modelling technique is Autoregressive Integrated Moving Average (ARIMA); considers time series to be linear. For seasonal time series SARIMA, a variation of ARIMA model is used. Also Artificial Neural Networks (ANN) proved successful for classification and forecasting purposes. ANNs have capability to model non-linear data which is not following any statistical distribution. Two models are generally used for time series:

– Multiplicative model:

$$Q(t) = Tr(t) \times Ss(t) \times Cy(t) \times Irr(t) \tag{1}$$

– Additive model:

$$Q(t) = Tr(t) + Ss(t) + Cy(t) + Irr(t) \tag{2}$$

Here Q(t) is observation, Tr(t) is trend, Ss(t) is seasonality, Cy(t) and Irr(t) is cyclic and irregular variation. Multiplicative model is based on assumption that time series can affect one another i.e. not necessarily independent. Additive model assumes components are independent of each other.

2   Time-Series Forecasting

Time series can be defined as an ordered sequence of values of a variable at equally spaced time intervals [27]. Time series models obtains details of forces and structure exhibited by observed data. Classical approaches decompose the time series into trend, seasonal and residual components. Before applying models data is examined and if necessary, some transformations are applied to be able to interpret the series better. This can stabilize the variance. For example, if there is trend in the series then logarithmic transformation is suggested. If there are outliers   they   need   to   be   adjusted. Time series is classified into two types:

- Univariate- means you have the time-stamp, and you have one variable connected to the time-stamp. For univariate datasets we have common statistics like mean and variance.
- Multivariate- is like a metric with several variables attached to time-stamp.

The fitting of a model is an ambitious undertaking. Many kind of models are there. Data stationarity is a common assumption in many time series techniques. Stationarity is a key trait when it comes to working with time data. Basically the question behind it is- has the data the same statistical properties throughout the time series? Stats properties which determine stationarity contain variance, mean and autocorrelation. Discussion on these models is done in [9,19,20,24].

3   Preparation Of Data

To make some predictions about the future the data has to be time series. For this project we used the FCI wheat stock data of Punjab region which had daily wheat stock record for the year 2017 [2] The dataset contains the daily observations of commodity stocks.

Converting to univariate The acquired dataset was multivariate as it contained data of multiple sub-regions of Punjab whereas our analysis required data from only one sub-region. Thus, we first removed the data related to other sub-regions and made it a univariate dataset.

Removing attributes We removed the attributes which were of no use in the forecasting process like the sub-region identifier ID and many more. Only contributing attributes in the forecasting were kept.

Converting to ts We used the ts() function of R[3] to create time series object of vector dataset. The data was sampled at equispaced points in time.

Decomposing The data had trend in it, and we found change in mean as a result of trend. Predictions tend to be underestimated as a result of this. One solution to the problem is to de-trend the data[4]. Thus, data was decomposed into single components[9], removed the trend component, and checked for trend stationarity. This was not enough and differencing was used, which gives difference stationarity.

Data cleaning Outliers induce biasing in forecasting models. They are unusual or may be errors. tsclean() function identifies and replaces outliers[19]. Other way could be logarithmic transformation[20].

Testing Stationarity Unit root tests can be used to check for trend stationarity or difference stationarity. The most prominent of these test is the augmented Dickey-Fuller test[24]. The adf.test() function in R[18] performs this test for null hypothesis of a unit root on a univariate ts. It removes autocorrelation before it tests for stationarity.

Now that the data is converted into a form that is suitable for accurate predictions, we move to forecasting methods. In next section we discuss about different forecasting techniques.
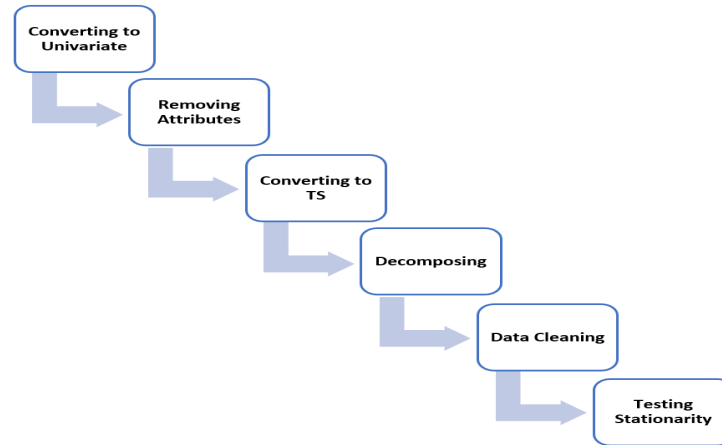
**Fig. 1.** Steps in preparing the data

4 Time Series Models And Forecasting

Interestingly, simple and primitive forecasting methods can outperform advanced models in certain circumstances. If dataset is random, then its hard with models

like ARIMA and exponential smoothing. The beauty of advanced models is that they can exploit patterns really well. The idea of forecasting model is to exploit one obvious fact. The fact can be the last observation, it can be the mean, or it can be the general trend. Forecasting methods involves several approaches to predicting uncertain events. Such system requires identifying forecasting problems, implementing a range of forecasting methods, making an appropriate selection for each problem and refining forecasting methods by evaluation over time[25].

4.1   Basic Forecasting Models: Mean, Naive And Drift

These models carry the observations forward. Last observations are projected into the future. The naive method can be tweaked so that it can work with seasonal dataset. Last observations of same seasonal stage are projected. Another simple method is average or mean which simply calculates the mean of data. Forecast of future values is equal to mean of historical data. Drift method calculate the increase between the first and last observation and carry the increase to the future. Figure 2 shows the predictions based on these models.
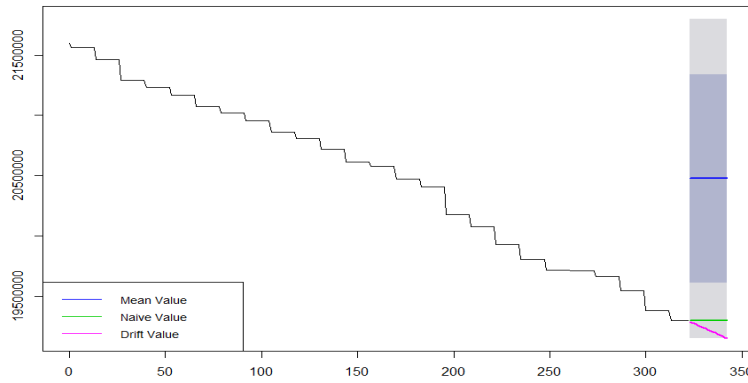
Fig. 2. Basic Models: Mean, Naive and Drift model forecast for next 20 days

4.2   ARIMA

ARIMA describe autocorrelation present in data. Seasonality and trend affect the time series at different intervals. ARIMA requires time series to be stationary whose properties do not depend on time at which series was observed. Differencing makes non stationary time series stationary. It computes differences in observations. Differencing stabilize the mean and therefore, trend and seasonality are either removed or reduced. Mathematically it is described as-

$$y(t) = y(t) - y(t-1)$$

The new series will have T - 1 values. Second order differencing can also be used if first order differencing is not satisfactory. ARIMA has three parameters:

- p : number of lags used in the model. It refers to the previous values used in the regression equation.
- d : used to stabilize the series when not stationary.
- q : moving average component represents the error of the model with combination of previous error.

The whole model is summation of lags. R has auto.arima() function that does the differencing automatically and calculate the parameters whereas the arima() function requires the parameters to be passed. Figure 3 and 4 shows the model with different parameters.
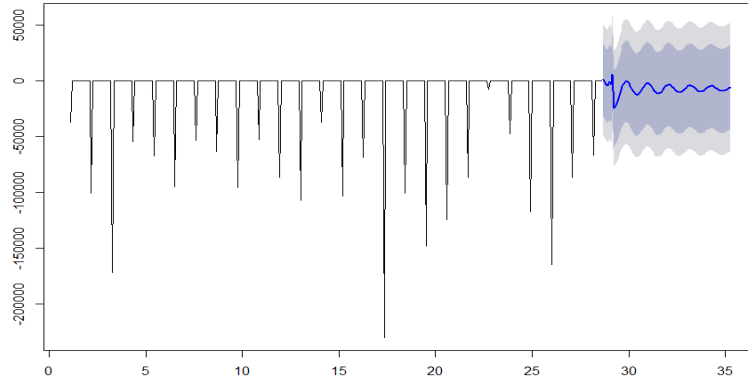
Fig. 3. Forecast from auto.arima() model(3,0,3) fitted to stabilized commodity stock
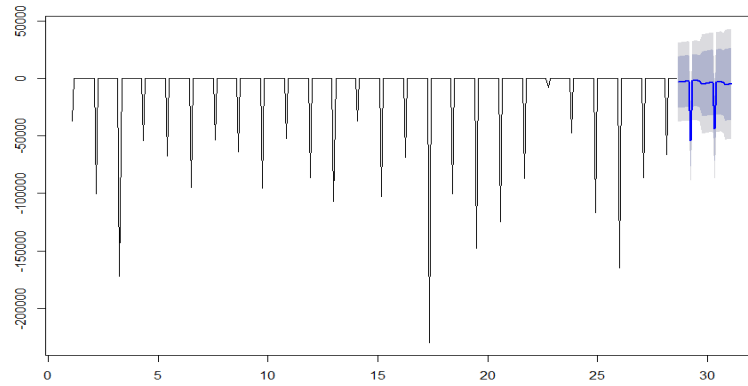
Fig. 4. Arima provided with parameters p=13, d=0, q=4

4.3   Exponential Smoothing

Exponential smoothing weight the previous observations by decreasing weights as the observations get older. Old data is considered as less relevant and progressively more weight is assigned to newer data. It ignores the deviations irrelevant to the purpose and hence can be used as a forecasting method to forecast data with no clear trend or seasonality. Moving average assigns equal weight of 1/N of all observations. In exponential smoothing to assign weights to the observations there are some smoothing parameters to be estimated.

Simple exponential smoothing decreases weight using weighted moving average. The basic formula is-

$$S_t = \alpha z_{t-1} + (1 - \alpha)S_{t-1} \tag{3}$$

where a is a value from 0 to 1 called smoothing constant and t is time period. Smoothing happens to be slow if a is close to zero. For data showing trend double exponential smoothing is more reliable:

$$l_t = \beta(S_t - S_{t-1}) + (1 - \beta)l_{t-1} \tag{4}$$

where beta is a constant which is chosen in reference to alpha. Model consist of three characters for identifying the method used. Error type is denoted by first letter, second denotes trend type and third denotes the season type. The type could be "N"=none, "M"=Multiplicative, "A"=Additive and "Z"=automatically. Figure 5 shows the prediction by additive error exponential smoothing model using ets() function in R.
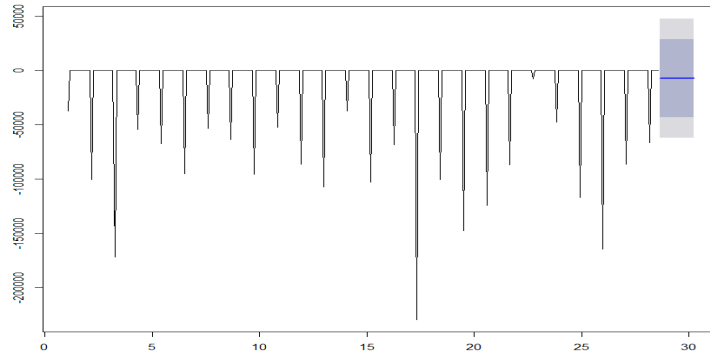

Fig. 5. Forecast from ETS(A,N,N)

4.4   Neural Network Model

It is a statistical model popularly used in machine learning. ANN model works as a brain in a mathematical way. It is a network of neurons organized in layers.

Linear combination of the inputs provides the forecast. The output of the hidden layer is modified using a non linear function called sigmoid. This is the associated activation function used to transform the input to an output which in turn acts as input to the subsequent layers.

For NN implementation to be successful proper training on training set is required. This process of training makes the NN system learn to adjust the weights connected with a suitable learning algorithm.

$$s(z) = \frac{1}{1 + e^{-z}} \qquad (5)$$
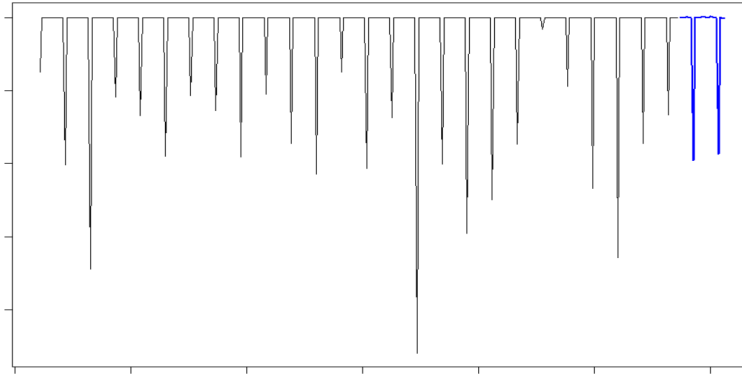
**Fig. 6.** Forecasts from neural network model using nnetar() function in R

We use the notation NNAR(p,q) to indicate p lagged inputs and k hidden layer nodes[11][Fig.6]. For example NNAR(7,3) means model is a neural network with last 7 observations.

4.5   SARIMA

Seasonal data can also be modelled using ARIMA. It explicitly supports univariate time-series with seasonal component. This is achieved by introducing additional terms in ARIMA. The seasonal part of the model consists of terms that are very similar to the non-seasonal components of the model. The four seasonal elements are:

- P: seasonal autoregressive order
- D: seasonal difference order
- Q: seasonal moving average order
- m: number of time steps for a single seasonal period

SARIMA chooses the lag equal to the number of seasons to remove the additive seasonal effects. Using the sarima() function we inspect the model fit diagnostics[14].
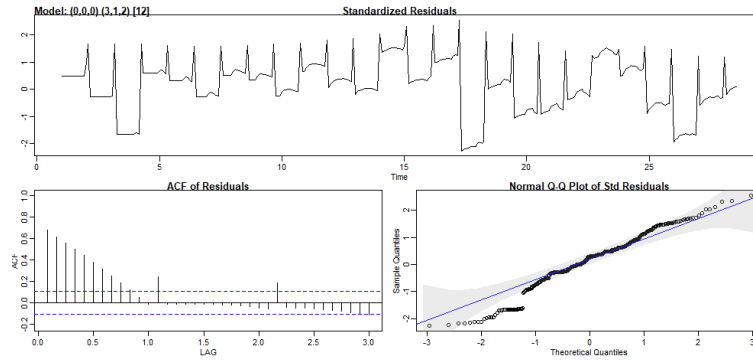
Fig. 7. SARIMA model diagnostics

Looking at the model diagnostics, we can see that the model does fit fine, although there might still be some outliers in the data with unexplained variance as shown in the Normal QQ plot[Fig.7]. Using the sarima.for() function, we can provide a forecast of the next few time intervals based on our model[Fig.6]. There is a prediction bounds[Fig.8], along with the predicted data. The uncertainty increases in amplitude as time progresses.
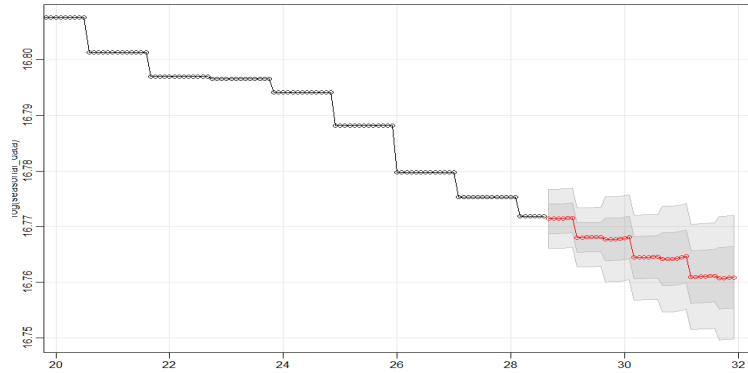
Fig. 8. SARIMA forecast prediction for 40 future points

5   Comparing Models Accuracy

Working on a problem we don't settle on one model, rather on two or more well-performing models. We compare the models by tuning the parameters of each and discover the worst and the best performer. It gives an idea of spread of model or perhaps one can be improved. The models constructed and tuned here

are Arima, exponential smoothing, Neural Network and Sarima. The models are tuned automatically(except Arima) and evaluated using MPE. Once the models are tuned with optimal configurations found for each the accuracy results of best models are collected. Data was partitioned into training dataset used to prepare the model and an unseen test set used to evaluate the model. The assessment of all methods shown in table 1.

Table 1. Model accuracy in terms of Mean Percentage Error.

| Forecasting Model	MPE
Mean	-5
Naive	-1.01
Drift	1.5
Auto Arima	0.96
Arima(13,0,4)	0.7
ETS	1.96
NNAR	0.64

6   Ensemble Method: Bootstrapping and Bagging

Time series can be simulated to predict future values by using a model. A similar time series to the observed series can be generated using bootstrapping. We present a method for bootstrap aggregation (bagging) of exponential smoothing methods. Time series is first Box-Cox transformed[16], and decomposed into trend, seasonal and residual components. A bootstrapped remainder series is obtained by shuffling versions of residual components. Contiguous sections of time series are selected at random and joined together. Trend and seasonality components are added and Box-Cox transformation is reversed to original time series.

Bootstrapping is useful in two ways:

- better measure of forecasting uncertainty.
- provides a way for improving forecast accuracy by using "bagging"

Our focus is on improving the accuracy of forecasting methods. Producing forecasts from each of the additional time series, and average the forecasting result, we get better forecasts than by applying on the original time series directly. This is called bagging. We will use ets() to forecast each of these series. Fig 9 shows ten forecasts obtained in this way.

We use the bootstrap() function in forecast package of R to create bootstrapped series. Code snippet shown below creates 10 different bootstrapped series on original data by performing bootstrapping and converting the series into time series objects. Fig. 9 shows 10 different bootstrap series fitted on the original time series. Plot is obtained using autoplot() function in R.
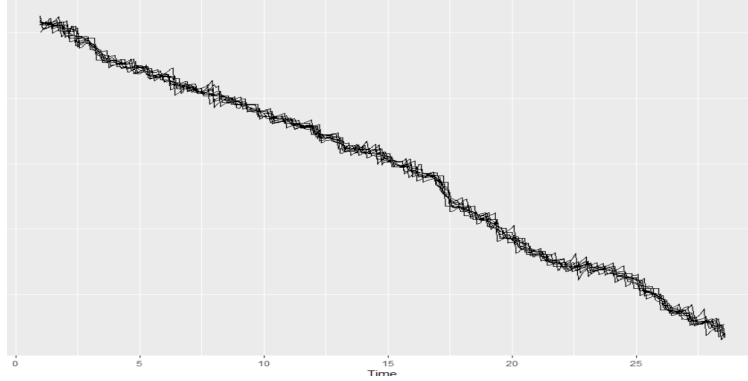
Fig. 9. Bootstrapped version of commodity stock

```
bs = bld.mbb.bootstrap(data, 10) %>%
  as.data.frame() %>% ts(frequency=12)
f = purrr::map(as.list(bs),
 function(x){forecast(ets(x))[["mean"]]})
  %>% as.data.frame() %>%
  ts(frequency=12)
```

The average of these forecast models gives the bagged forecast of original data. The entire process is handled by baggedETS() function. By default, 100 bootstrapped series are used.

```
ets_output = data %>% ets() %>%
               forecast(h=36)
bagged_output = data %>% baggedETS()
               %>% forecast(h=36)
```

Resulting forecasts are shown in Fig. 10. The difference seems to be very little. But bagging gives better forecast by applying on huge datasets. The method gave MPE of 0.01 on commodity stock dataset and outperformed all other models.

7   Conclusion

In this analysis we compared the results from multiple models, with their parameters optimized. Capability of model is evaluated on unseen data. Basic forecasting models are outperformed by advanced methods. Neural network model performed good and exponential smoothing did worst. Improving the accuracy by using ensemble methods on exponential smoothing the error was reduced largely. This is because bagging considers the average of forecast uncertainty to improve forecasting accuracy. It simulates original series by creating dummy series and fit the newly created series on original one hence averaging the result for prediction. It reduce the variance in prediction and improve the stability and accuracy. Of course, it is slower as more computations are required.
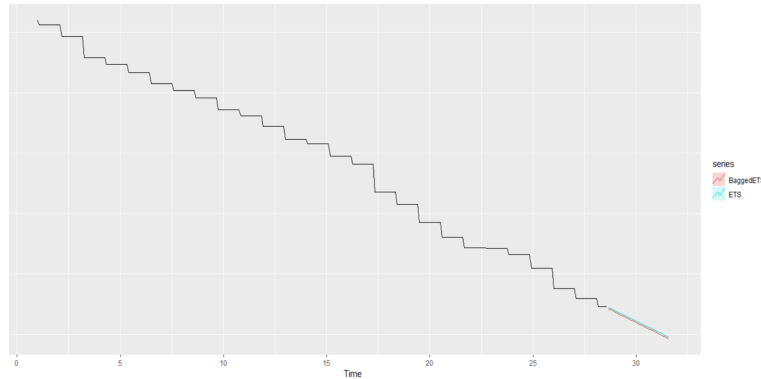
Fig. 10. Bagged ETS forecasts and ETS applied directly to the data.

References

1. Lajbcygier, Paul R., and J. T. Conner. "Improved option pricing using bootstrap methods." Proceedings of International Conference on Neural Networks (ICNN'97). Vol. 4. IEEE, 1997.
2. Open Government Data(OGD) Platform India, https://data.gov.in/keywords/fci-stock
3. ts: Time-Series Objects, https://www.rdocumentation.org/packages/stats/versions/3.5.3/topics/ts
4. J. Brownlee, How To Decompose Time Series Data Into Trend And Seasonality, https://machinelearningmastery.com/decompose-time-series-data-trend-seasonality/
5. He, Haibo, and Xiaoping Shen. "Bootstrap methods for foreign currency exchange rates prediction." 2007 International Joint Conference on Neural Networks. IEEE, 2007.
6. Moraes, L. A., et al. "Development of a methodology to forecast time series using few input variables." 2013 IEEE PES Conference on Innovative Smart Grid Technologies (ISGT Latin America). IEEE, 2013.
7. Rahman, Md Mustafizur, et al. "Forecasting time series—A layered ensemble architecture." 2014 International Joint Conference on Neural Networks (IJCNN). IEEE, 2014.
8. Mahalakshmi, Ganapathy, S. Sridevi, and S. Rajaram. "A survey on forecasting of time series data." 2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16). IEEE, 2016.
9. Seasonal Decomposition Of Time Series By Loess- Decompose a time series into seasonal, trend and irregular components using loess, https://www.rdocumentation.org/packages/stats/versions/3.6.0/topics/stl
10. Xinxiang, Zhang, Zhou Bo, and Fu Huijuan. "A comparison study of outpatient visits forecasting effect between ARIMA with seasonal index and SARIMA." 2017 International Conference on Progress in Informatics and Computing (PIC). IEEE, 2017.
11. Kedrowski, Reilly, et al. "Short-Term Seasonal Energy Forecasting." 2018 IEEE International Conference on Electro/Information Technology (EIT). IEEE, 2018.

12. Ratakonda, Sai Swaroop, and Sreela Sasi. "Seasonal Trend Analysis on Multi-Variate Time Series Data." 2018 International Conference on Data Science and Engineering (ICDSE). IEEE, 2018.
13. Debasish Sena1 and Naresh K. Nagwani, A NEURAL NETWORK AUTORE-GRESSION MODEL TO FORECAST PER CAPITA DISPOSABLE INCOME
14. Kenneth Foo, Seasonal lags: SARIMA modelling and forecasting, Jan 3, 2018, https://medium.com/@kfoofw/seasonal-lags-sarima-model-fa671a858729
15. Kandananond, K. (2012). A Comparison of Various Forecasting Methods for Auto-correlated Time Series. International Journal of Engineering Business Management. https://doi.org/10.5772/51088
16. Box, G. E. P. and Cox, D. R. (1964). An analysis of transformations, Journal of the Royal Statistical Society, Series B, 26, 211-252.
17. Vaishnav, Vaibhav, and Jayashri Vajpai. "Seasonal Time Series Forecasting by Group Method of Data Handling." 2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS). IEEE, 2018.
18. Augmented Dickey-Fuller Test, https://www.rdocumentation.org/packages/aTSA/versions/3.1.2/topics/adf.test
19. Dealing with missing values and outliers, https://otexts.com/fpp2/missing-outliers.html
20. Transforming Data, http://rcompanion.org/handbook/I-12.html
21. Dong, Xibin, et al. "A survey on ensemble learning." Frontiers of Computer Science: 1-18.
22. Maqsood, Imran, Muhammad Riaz Khan, and Ajith Abraham. "An ensemble of neural networks for weather forecasting." Neural Computing & Applications 13.2 (2004): 112-122.
23. Rao, J. Sunil. "Bootstrapping to assess and improve atmospheric prediction models." Data Mining and Knowledge Discovery 4.1 (2000): 29-41.
24. Rob J Hyndman and George Athansopoulos, Forecasting, planning and goals https://otexts.com/fpp2/planning.html
25. Rob J Hyndman and George Athansopoulos, Advanced Forecasting Methods, https://otexts.com/fpp2/nnetar.html
26. Zaman, M. Faisal, and Hideo Hirose. "Classification performance of bagging and boosting type ensemble methods with small training sets." New Generation Computing 29.3 (2011): 277.
27. NIST/SEMATECH e-Handbook of Statistical Methods, http://www.itl.nist.gov/div898/handbook/
28. Rob J Hyndman, Unit Root Tests, https://www.rdocumentation.org/packages/aTSA/versions/3.1.2/topics/adf.test