Ruprecht-Karls-Universität Heidelberg

Time Series Forecasting using Machine Learning

Fundamentals of Machine Learning
Heidelberg Collaboratory for Image Processing (HCI)

Project report by
Narasimha Abhinav Koganti

WS 2017 / 2018

Supervisor: Dr. Ullrich Köthe / Lorenzo Cerrone

Abstract

As time progresses different methods are being sorted out by statisticans, data scientists to machine learning experts for forecasting time dependent data . In this project work, we analyse the efficency of different methods viz. LSTM ,Sequence Vector Regression (SVR) , Vector auto regression (VAR), Auto regressive integrated moving average (ARIMA) and Random Walk model and their variations,if any, on uni variate as well as multivariate datasets viz. Airline freight TAC dataset (uni variate) and Beijing pm2.5 Pollution dataset(multi variate) and compare results using stipulated metrics viz. MAE loss and R2 score. We analyse all the models on stipulated train and test sets for comparision. We find the best, both approximation of the models as well as generative capability (forecasting future), of models on test set.

Out of the all the models we explored, we find that SVR model achieves the best R2 score of 0.967 and best MAE loss of 0.0066 on TAC dataset test set.

And, we find that Stacked LSTM model achieve the best R2 score of 0.966 and best MAE loss of 1.92 on Beijing Pollution dataset test set.

Links:
TAC index data set: http://www.aircargonews.net/graphs/tac-index-airfreight-rates.html
Beijing pm2.5 data set : https://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data

Table of contents

Chapter 1. Introduction

Forecasting plays an important role in many fields such as data science, economics, finance, supply chain, marketing, weather and nature conditions etc. Planning and preparing for the future is a most critical and important part in any organization. Long-term success of any business is often depending on how well the management can anticipate the future scenarios and provide strategies based on them [1]. It is not enough to make judgements relying only on intuition or awareness of economic or political situation. Thus, good forecasting, which is providing as much as possible an accurate picture of the future is vital. Hence, the first question arises, what is forecasting. It is the process of predicting future trends, values based on the historical and present data. A time-series forecasting is the prediction based on a collection of observations made sequentially through equally spaced periods of time [2]. Examples may include to predict future sales, prices or index trends of a particular product based on past data in successive years, months, weeks; to predict the weather based on the features such as temperature rate, wind and humidity taken hourly; demand on special goods. As it has been mentioned, good forecasting, where thorough collection and study of the past event to develop a suitable model, which can after inheritance of the structural representation of the time series generate future observations is important. Thus, various methods and techniques are proposed and are in a wide use. A variety of predictive methodologies is presented in a classical approach of statistical based methods. One of the representatives is the linear autoregressive models (AR). These methods are popular because of their flexible application to any stationary time series. However, these well-known approaches is limited to univariate time series. The generalization of AR method includes Vector Autoregressive Models (VAR), proposed by Chris Sims in 1980. It is an extension of the univariate autoregressive model to multivariate time series. According to the Wold theorem, ensuring any vector of time series to have a VAR structure [29], makes VAR as the starting point of the analysis.

Previously mentioned AR models include extensions such as autoregressive moving average models (ARMA) [30], [31]. The ARMA models are suitable for representation a linear relationship between consequent observations; hence, they fail to forecast nonlinear and non-stationary cases, which are mainly typical examples of real-world systems. Methods such as the linear autoregressive integrated moving average (ARIMA) [30] are able to overcome non-stationary cases by removing non-stationary applying differencing. In addition, they are able to extract seasonal and other cyclical components from the time series by means of an iterated finite moving average procedure [32]. ARIMA methods are widely employed in financial markets. However, differencing technique in ARIMA increase frequency of noise in the series, making hard to determine the order of an ARIMA model [34]. Moreover, ARIMA models are limited to take into account the first-order non-stationarity of the time series.

The solution can be found in recent applications of data mining and machine learning techniques such as Artificial Neural Networks (ANNs) [35, 36, 37] and Support Vector Machines (SVM) [7, 43, 44, 45]. The usage of these methods to predict the time series are increasing day by day. Nevertheless, initially ANNs were biologically inspired, later they have been applied to many different spheres, and especially for forecasting and classification targets [34]. The advantage of ANNs in application to the time series prediction is their ability to model non-linear cases and not making assumptions on statistical distribution of the time series. Based on the given data structure the appropriate model is derived adaptively [34]. During the past few years considerable amounts of research have been performed on ANNs modelling and forecasting of the time series. There are different types of ANNs for forecasting presented. The most common and appropriate to the time series modelling is the Recurrent Neural Network (RNN) [38, 39, 40]. However, RNN has its issue of vanishing gradient problem [41]. A network called the Long Short-Term Memory (LSTM) is able to solve this problem, thus it is considered as an improvement over RNN.

Another machine learning technique, which has a major breakthrough in the area of time series forecasting is Support Vector Regression (SVR). SVR is an extension of Support Vector Machines (SVM), which can handle regression case. There is a study paper that SVR can outperform a Random Walk model [42], which is a classical method of ARIMA of order (0, 1, 0), in contrast to the Burton Malkiel statement [46]. Furthermore, there are some research papers on the time series applying ANNs and SVR [47, 48, 49, 50], where SVR has shown better results.

Thus, this work is aimed to apply and compare classical and contemporary methods to model the time series for predicting future values.

1.1 Objective

The objective of this mini research project is to carry out a comparative study of different time series forecasting techniques.

The techniques we chose to model are:

1. LSTM based neural network regression model.
2. Sequence Vector Regression (SVR) model.
3. Autoregressive integrated moving average (ARIMA) model.
4. Vector autoregression (VAR) model.
5. Random Walk model.

The techniques are explored on two data sets viz. TAC index data (Univariate data) and PM2.5 Beijing Pollution Index data set (Multivariate data).

Once these techniques are modeled on the data sets appropriately, the prediction / forecast values will be compared according to chosen loss metrics to assess each model's performance.

1.2 Outline

This report is structured in the following way. The next chapter describes existing different time series types. In addition, it describes two different datasets, which are used in this project for forecasting. Chapter 4 introduces different methods used in this project. It also provides results and evaluation on applied techniques. Chapter 5 discusses and concludes the results among applied techniques and presents several recommendations for future work.

Chapter 2. Dataset description

2.1 Types of the time series

Univariate time series

In univariate time series, we use values of the same variable for forecasting future values. The forecast value is calculated using lagged values, which are usually measured at regular intervals. Time series are modelled with univariate models, when the factors affecting the time series are properly not known [60].

$$y_t = F\left(y_{t-1}, \dots\dots, y_{t-p}\right)$$

Multivariate time series

Multivariate time series extend univariate time series techniques to deal with vectors. They explain how a group of time series variables move and interact over time. Multivariate time series models help in understanding how the moments of one variable effect the values of other variables, which helps in better forecasting [61].

Linear time series:

Linear time series can be modelled using linear equations. They are based on covariances between lagged values in the time series. Some examples of linear time series models are Auto regression (AR) and Moving Average (MA), where we forecast future values using linear combination of past values [62].

Nonlinear time series:

A nonlinear time series have high moments, thresholds, breaks etc, which cannot be modelled by linear equations. Examples of non-linear models are GARCH, TGARCH etc, where we try to model unexpected variance in the time series.

Stationary series:

Stationary series are subset of the linear time series. A series is said to be stationary if its statistical properties do not depend on the time. In general, a stationary time series values fluctuates around the mean and its statistical properties (mean, variance etc) are constant in all periods of time [63].

Non-stationary time series:

Contrarily to a stationary time series, in a non-stationary we see certain patterns like trend, seasonality and cyclic [64].

Trend

A time series is said to have trend, if it shows a decrease or increase in the long term with the possibility that it can change its direction multiple times.

Seasonal

Certain data like temperature, rain etc. shows seasonality in their values, for example, they are high in certain periods and low in other periods of the year. When our data exhibits this kind of behavior, it is said that the time series has a seasonal component. Seasonality can be of any type such as quarterly, monthly, yearly etc.

Cyclic

A time series is said to have a cyclic component, if there is rise or fall in the data for unexpected period and time.

2.2 Tac index dataset

For shipping cargo in air there is no benchmark price its mostly based on tenders. TAC index gets data from major firms, which control most of the air freight and provide information about air cargo pricing for selected lanes. This helps in data drive decisions by finding trends in the prices [65]. TAC index dataset used in this work is univariate and non-stationary. Non-stationarity of the data is checked by visual inspection and Augmented Dickey Fuller test, which helps to clarify the data type (given in the following sections). The data consists of 1058 points in total, representing TAC index over three years. The data does not have missing values. The graph below shows TAC index data:

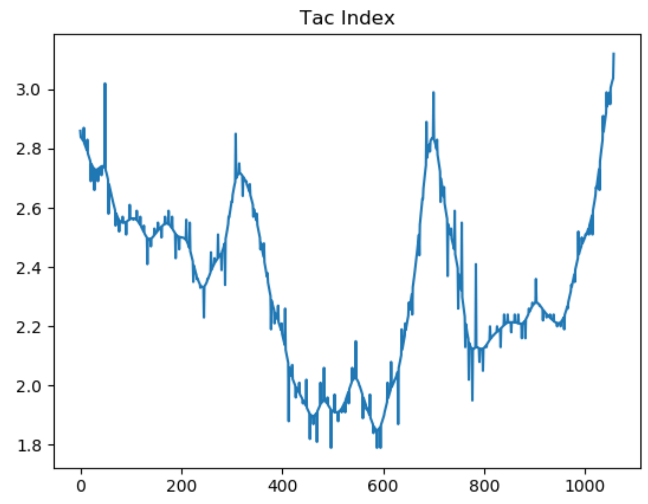


Figure 1. TAC index dataset plot.

2.3 Beijing pollution dataset

Beijing PM 2.5 dataset is taken from UCI Machine Learning Repository [3]. It is hourly data containing 'fine particulate matter PM 2.5, an air pollutant' in Beijing, causing a concern for people's health when levels in air are high [4]. In addition, the dataset consists of meteorological data including dew point, temperature, pressure, wind direction, wind speed and cumulated hours of snow and rain, thus making the data multivariate.

The period of the dataset is between Jan 1st, 2010 to Dec 31st, 2014 [3]. Total number of instances is 43824 with integer, categorical and real values. The complete list of raw data can be represented as:

No.	**Name**	**Description**	**Type of value**
1	No	Row number	Integer
2	Year	Year of data	Integer
3	Month	Month of data	Integer
4	Day	Day of data	Integer
5	Hour	Hour of data	Integer
6	Pm2.5	PM2.5 concentration	Integer
7	DEWP	Dew point	Integer
8	TEMP	Temperature	Integer
9	PRES	Pressure	Integer
10	Cbwd	Combined wind direction	Categorical
11	lws	Cumulated wind speed	Float
12	ls	Cumulated hours of snow	Integer
13	lr	Cumulated hours of rain	Integer

Table 1. The list of attributes of Beijing PM2.5 data

The first 24 rows of the data are without information on PM2.5 concentration, thus as a part of data pre-processing and cleaning it has been removed from the actual dataset. Despite that, the data contains several missing values in Pm2.5 concentration, which has been then filled through interpolation. Python has interp1d class in scipy.interpolate, which allows acquiring unknown values by passing 1-d vector and using linear interpolation function [5].

Figures below represent Beijing PM2.5 dataset over five years except wind direction feature, as it is categorical one. From the plot of the data, the data is non-linear, stationary, without trends, but with seasonality. Stationarity of the data has been also checked by Augmented Dickey Fuller Test (has been provided in the later sections).

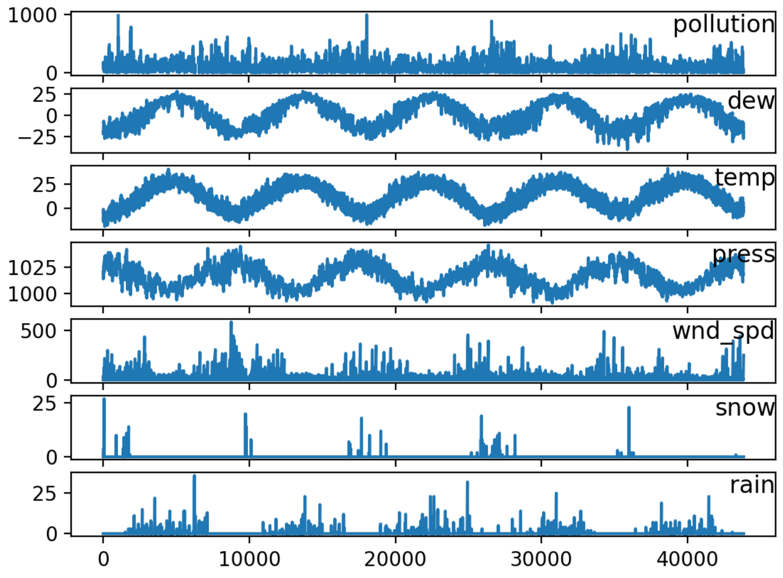


Figure 2. Beijing pollution dataset plots.

Chapter 3. Loss Metrics

As we use a number of methods in the project to explore the concept of time series forecasting, it is imperative to have common ground for comparison in terms of loss. A Loss function is an important part in machine learning, which is used to measure the inconsistency between predicted value (ŷ) and actual label (y). It is a non-negative value, where the robustness of model increases along with the decrease of the value of the loss function.

Generally, the loss function consists of a risk term and a regularization term. The objective of any approach is to minimize the below equation:

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{n}\sum_{i=1}^{n} L\big(y^{(i)}, f(\mathbf{x}^{(i)}, \boldsymbol{\theta})\big) + \lambda \cdot \Phi(\boldsymbol{\theta})$$

Since the objective is regression (where the output variable takes continuous values) than classification (where the output variable takes class labels or categorical variables), accuracy of the prediction can be only expressed in terms of losses incurred in the prediction.

Hence, we choose two loss functions as a metric to evaluate our results.

3.1. MAE / L1 Loss

MAE – Mean Absolute Error Loss is a measure of difference between two continuous variables. It indicates how forecasts or predictions are close to the eventual outcomes. It is determined by taking the average of L1 loss between the predictions and ground truth.

$$\boldsymbol{\mathcal{L}} = \frac{1}{n}\sum_{i=1}^{n} \left| y^{(i)} - \hat{y}^{(i)} \right|$$

Considering a constant interval time series plot in Cartesian coordinates, Mean Absolute Error (MAE) is the average vertical distance between the forecasted and ground truth values. As in [28], MAE is more robust to outliers since it does not make use of square as in the popular MSE loss. Also, MSE loss is not required here because x – axis (time axis) is consistent across the dataset.

A more popular variant is MAPE – Mean Absolute Percentage Error, used as a measure of prediction accuracy of a forecasting method. However, as it has some severe drawbacks such as a risk of division by zero or non-existence of upper limit of error for too high forecasts, which prevents us from using it.

$$\boldsymbol{\mathcal{L}} = \frac{1}{n}\sum_{i=1}^{n} \left| \frac{y^{(i)} - \hat{y}^{(i)}}{y^{(i)}} \right| \cdot 100$$

3.2 R2_squared coefficient

R-squared coefficient is used to determine how close the data points are to the fitted regression curve. It is the proportion of the variance in the dependent variable that is predictable from the independent variables. Or is, simply, the percent of variation expressed by regression out of total variation [83]:

$$r^2 = \frac{SSR}{SSTO}$$

$$r^2 = 1 - \frac{\sum_{i=1}^{n}(\acute{\hat{y}}_i - \acute{y})^2}{\sum_{i=1}^{n}(y_i - \acute{y})^2}$$

High R-squared values means that most of variance is explained by regression. Thus, more points lie around the regression curve. curve. An R-squared value of 1.0 is the best. It means there is no error in your regression. An R-squared value of 0 means the regression is no better than taking the mean value. A Negative R2 means the regression is worse than the mean value.

Limitations:

As R-squared values are biased towards the linear model, in some cases we can also have good R-squared values for badly fitted curves. And in some fields like psychology, where the model is harder to predict, low R-square values does not really mean a bad model. Also, R2 increases as we increase the number of variables in the model. Hence, it is not reliable to depend only on R-square values it Is generally used in combination with other measurements.

Chapter 4.

Methods: theoretical background, implementation and results

4.1 LSTM based Time Series Forecasting

4.1.1 Introduction:

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). An LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events. LSTM was proposed in 1997 by Sepp Hochreiter and Jürgen Schmidhuber[33] and improved in 2000 by Felix Gers[54].

Here in this section, we approach the LSTM model as a means of time series forecasting we try out 2 different configurations on the two datasets and present the results. We formulate the supervised learning problem as predicting the data at the current hour (t) given the input variables (univariate or multi variate depending on the data set) at the prior time steps.

NB: *This section does not discuss about basics of neural networks &/or deep learning. Those topics are out of scope for this report.*

4.1.2 Theoretical Background:

Artificial Neural Networks (ANNs) were designed, in the most general sense, as a tool for predicting a certain variable, given as set of data. This prediction can take the form of a Classification or a Linear Regression problem, with the key difference between the two being the continuity of the data. Classification problems refer to the categorization of the data into different sub categories e.g. being given a data set of dog pictures and being able to classify or label the images as their correct breed.

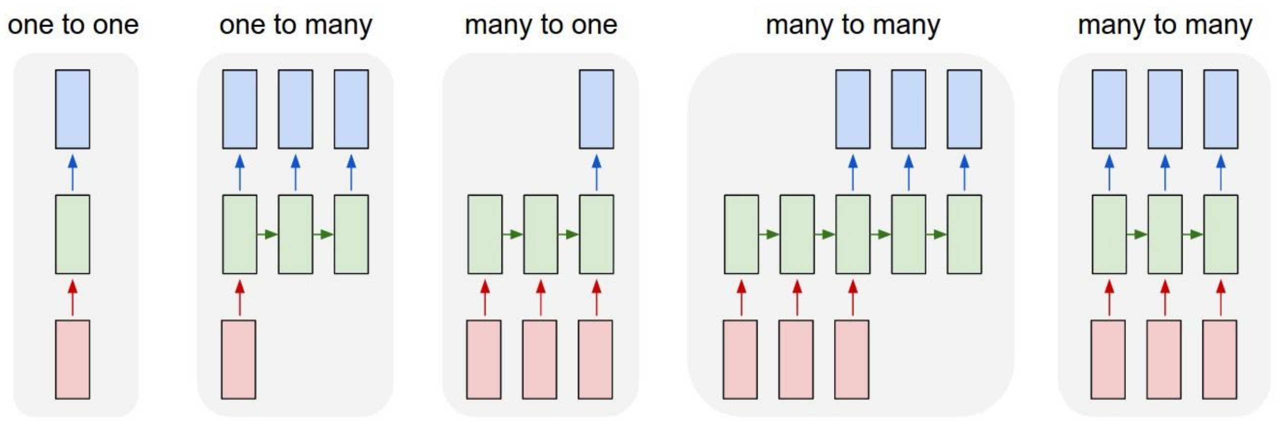
Regression allowed us to find relationships between the input variables and the outputs and then predict with more precision the output value. However, the problem with classical neural networks is that they require the variables to be roughly independent [55] which is not possible for time series input data.

Data, which is represented as a time series normally, represents a single object but the input variable will be the state of the object at any given time.

Traditional neural networks which are used for classification or regression problems are not suitable for time-series forecasting problems, as they are unable to retrain information about the previous inputs in memory; for e.g. in case of predicting the next word in a sentence.

The Recurrent Neural Network is the solution. These are neural networks, which take the weights of the neural network as an input as well as the input variable themselves. So, in effect, it remembers the previous state of the last time step. This essentially causes a loop in the neural network allowing information to persist over time steps. Thus, RNNs operate over a sequence of vectors.

Different types of RNN configurations:



Here, in the above figure [Courtesy: http://karpathy.github.io/],

1. The “one to one” is the vanilla neural network which takes fixed-sized input to fixed-sized output (e.g. image classification)

2. The “one to many” setup is sequence output (e.g. image captioning takes an image and outputs a sentence of words).

3. The “many to one” setup is sequence input (e.g. human sentiment analysis)

4. The “many to many” is sequence input and sequence output (e.g. Machine Translation, Time series analysis)

5. Another “many to many” - synced sequence input and output (eg. Video classification, time series analysis)

The basic working of a RNN can be summarized in the below equation:

$$h_t = \tanh(W_{hh}\, h_{t-1} + W_{xh}\ x_t)$$

$$y_t = W_{hy} h_t$$

where

h_t, h_{t-1} denotes the hidden state at time t and t-1 respectively.

X_t denotes the input at time t.

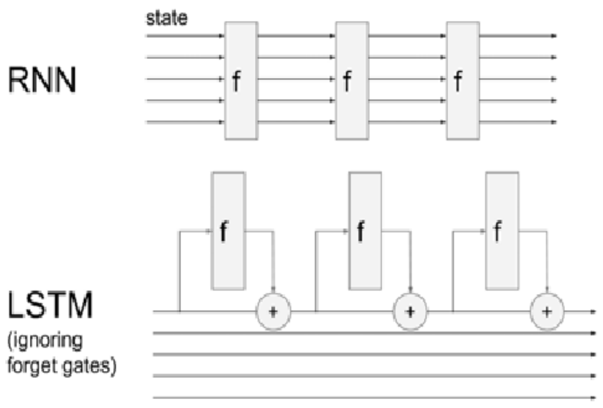
W_{hh} and W_{xh} denotes the network weight matrices to be trained.

Now, the LSTM is an improvement over the RNN with some elaborate changes in the above equation, but the basic configuration being the same. This will be explained in the coming sections of the report.

A RNN can be unfolded in both time and space. However, the important problem with RNN is its vanishing gradients while back propagation. More the unfolding, deeper a RNN gets. In train time back propagation each of the neural network's weights receives an update proportional to the gradient (calculated by chain rule) of the loss function with respect to the current weight in each iteration of training. The problem is that in some cases, the gradient will be vanishingly small, effectively preventing the weight from changing its value. The RNNs being trained by unfolding them into very deep feed forward networks, where a new layer is created for each time step of an input sequence processed by the network, vanishing gradient problem is significant.

LSTM completely avoids the vanishing gradient problem. LSTM is normally augmented by recurrent gates called "forget" gates with which the errors can flow backwards through unlimited numbers of virtual layers unfolded in space.

Below the figure shows the conceptual difference between a RNN & LSTM:



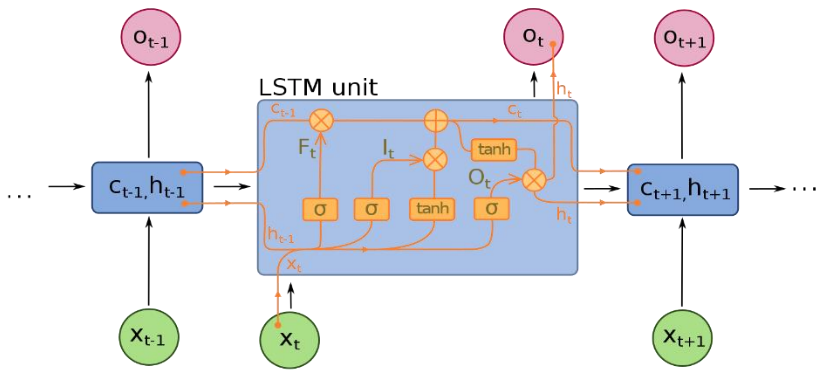
Thus, an LSTM models the following function:

$$
\begin{aligned}
i_t &= \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{(t-1)} + b_{hi}) \\
f_t &= \sigma(W_{if}x_t + b_{if} + W_{hf}h_{(t-1)} + b_{hf}) \\
g_t &= \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{(t-1)} + b_{hg}) \\
o_t &= \sigma(W_{io}x_t + b_{io} + W_{ho}h_{(t-1)} + b_{ho}) \\
c_t &= f_t c_{(t-1)} + i_t g_t \\
h_t &= o_t \tanh(c_t)
\end{aligned}
$$

where h_t is the hidden state at time *t*, c_t is the cell state at time *t*, x_t is the hidden state of the previous layer at time *t* or input for the first layer, and i_t, f_t, g_t, o_t are the input, forget, cell, and out gates, respectively. σ is the sigmoid function.

W_{xx} denotes the network weight matrices to be trained.

Below is the diagram of a LSTM: *[Picture Courtesy: Wikipedia]*



LSTM network can be assumed as a function that maps:

$\mathbf{R^n \times R^d \times R^d \rightarrow R^d \times R^d}$

Matrix wise, this is written as:

$$\underset{Dim:4d}{\begin{bmatrix} i \\ f \\ o \\ g \end{bmatrix}} = \underset{Dim:4dx4d}{\begin{bmatrix} sig \\ sig \\ sig \\ tanh \end{bmatrix}} \underset{Dim:4dx(n+d)}{[Wt]} \underset{Dim:(n+d)x1}{\begin{bmatrix} h_t \\ h_{t-1} \end{bmatrix}}$$

where n is the size of input and d is the hidden state size.

4.1.3 Data Preprocessing

We explore the scope of LSTM network performance on prediction on TAC data set and Beijing pollution data set as explained in the previous sections.

We first process the data set by normalizing it. As in [55] Normalization (scaling) of data within a uniform range (e.g., 0–1) is essential (i) to prevent larger numbers from overriding smaller ones, and (ii) to prevent premature saturation of hidden nodes, which impedes the learning process. Hence, we scale the data set in the range 0-1 with the formula:

$$X\_std = \frac{X - X.min(axis=0)}{X.max(axis=0) - X.min(axis=0)}$$

$$X_{normalized} = X\_std \times (max - min) + min$$

where min and max denote the range 0-1 respectively.

This type of normalizing is called Feature Scaling. This type of normalization is popular for regression problems, where the gradient is found using L1/L2 distance. The range of all features should be normalized so that each feature contributes approximately proportionately to the final distance.

Next stage is phrasing time series data as supervised learning data. Given a sequence of numbers for a time series data set, we can restructure the data to look like a supervised learning problem. This is done by using previous time steps as input variables and use the next time step as the output variable. This is called the Sliding Window technique.

Below Table shows the first 4 rows (hourly data) of Beijing Pollution data set.

Pm2.5 X_1	**DEWP** X_2	**TEMP** X_3	**PRES** X_4	**cbwd** X_5	**Iws** X_6	**Is** X_7	**Ir** X_8	**Label** **Y**
0.12977	0.35294	0.2459	0.52727	0.666	0.0229	0.00	0.0	0.14889
0.14889	0.36764	0.2459	0.52727	0.666	0.0038	0.00	0.0	0.15996
0.15996	0.42647	0.2295	0.5454	0.666	0.00533	0.00	0.0	0.18209
0.18209	0.48529	0.2295	0.56363	0.666	0.00839	0.03703	0.0	0.1388

Table 2. The first four rows of Beijing PM2.5 data

Columns X_1 to X_8 shows the 8 dimensions of the input data and the Label Y is the ground truth value we expect the LSTM to predict.

One can note that the Label Y reappears in the X1 column for the next sample of the data set. This is because we are modeling the LSTM to predict pm2.5 pollution value for the next hour, given current hour.

Similar Sliding window is applied to the univariate time series data set of TAC.

4.1.4 Methodology

We attempt 2 configurations of LSTMs for these data sets.

1. Stacked LSTM configuration for univariate data.

2. seq2seq Encoder Decoder configuration for multivariate data.

4.1.4.1 Stacked LSTM network

Stacking LSTM hidden layers makes the model deeper, more accurately earning the description as a deep learning technique. It is the depth of neural networks that is generally attributed to the success of the approach on a wide range of challenging prediction problems. Stacked LSTMs or Deep LSTMs were introduced by Graves, et al. [56] in their application of LSTMs to speech recognition, beating a benchmark on a challenging standard problem. A Stacked LSTM architecture can be defined as an LSTM model comprised of multiple LSTM layers. An LSTM layer above provides a sequence output rather than a single value output to the LSTM layer next.

Architecture: **LSTM** (Input shape =1, Output shape = 51) →**LSTM** (Input shape =51, Output shape = 51) → **FC layer** (Output shape = 1) → Optimize Module

Optimize Module:

To predict the results with more accuracy, trend information can be exploited manually. We propose an optimization based on the core idea that LSTM output can be modified to perfection by referring to previous available values i.e. current trend. With an arbitrary number of previous points available we do a linear regression and find the slope of the approximated line.

$$Y = \text{slope} \times x + \text{intercept}.$$

Once we have slope from the line fitting, we assume intercept =0 and next point can be calculated. This estimated next point is checked with the LSTM output and the error is added to the LSTM prediction.

LSTM is trained on TAC data set on 1000 data points and tested on 58 points. Also,

LSTM is trained on Pollution data set on 12000 data points taking only pm2.5 values (Univariate) and tested on 3000 points.

4.1.4.1.1 Results

The graph (fig. 3) shows the LSTM performance on TAC data test set.

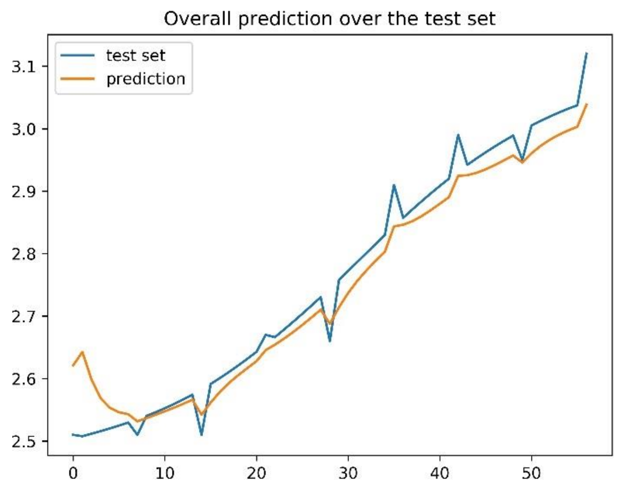


Figure 3. LSTM's prediction over the TAC test set.

L1 loss over test set is 0.0305 and R2 loss is 0.955 which is very close to 1, can be considered as well fitted model.

The graph (fig. 4) shows the LSTM performance on Beijing pm2.5 Pollution data test set:

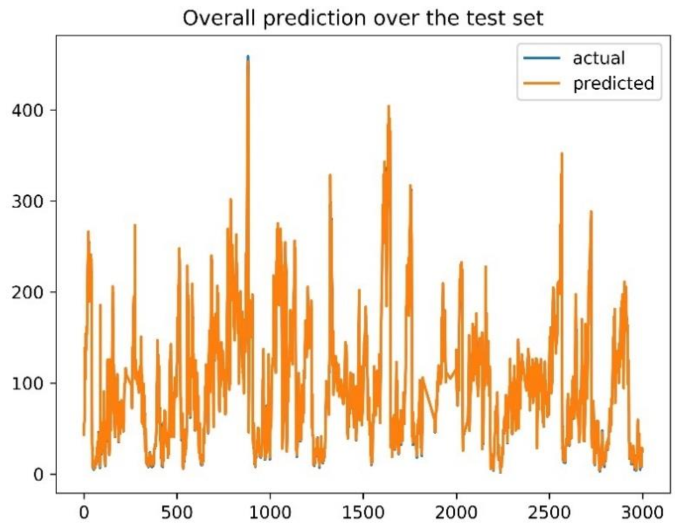


Figure 4. LSTM's prediction over the Beijing pm2.5pollution test set.

L1 over test set is 1.92 and R2 loss is 0.996 which is also very close to 1, can be considered as well fitted model.

This plot shows that LSTM can make fairly approximate predictions of time (t+1)th instance given instance t. However this is not the case in forecasting (t+2)th or further instances as the error can get compounded with the feedback input of the predicted outcome. This is a challenging case in forecasting arena. We see what happens when LSTM is let to generate 3 future instances at different starting points of the test set.

The plot (fig. 5) shows attempts made by LSTM on TAC Dataset to forecast 3 future points in a gap of 5 points on test set:

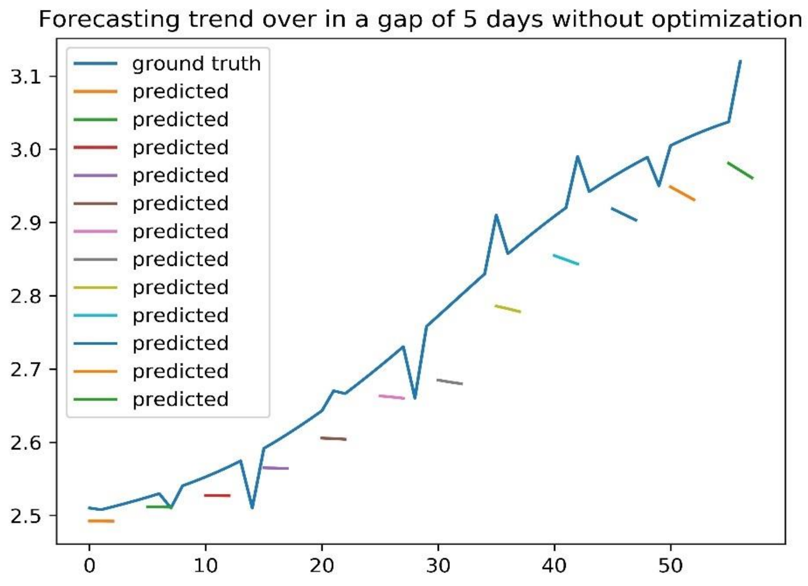


Figure 5. LSTM's forecasting over the TAC test set without optimization.

We can see LSTM almost fails to predict the trend especially towards the end (<30 days). Here is the situation we apply some optimization as explained in the architecture. The average MAE loss over all forecasting is 0.04.

The plot (fig. 6) shows attempts made by LSTM (with optimization) to forecast 3 future points considering an arbitrary window of 3 points of the past (window), in a gap of 5 points on test set.

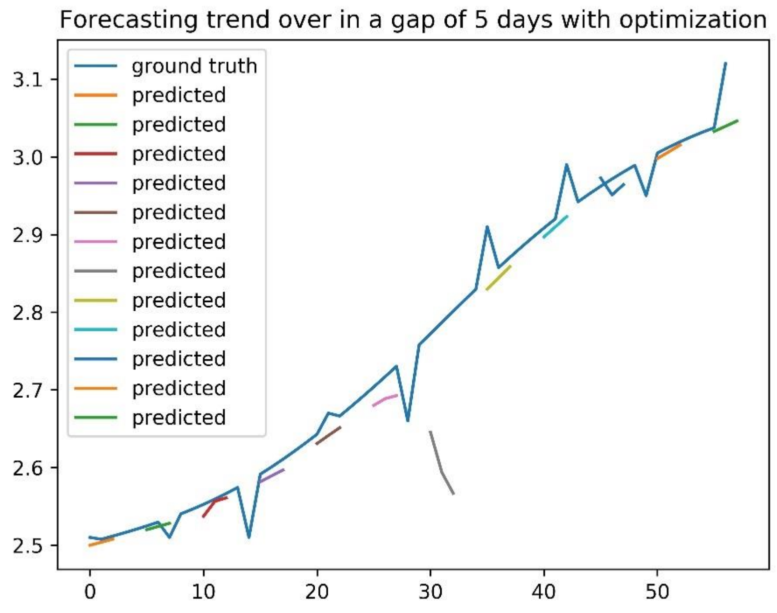


Figure 6. LSTM's forecasting over the TAC test set with optimization.

The average MAE loss over all forecasting is 0.025.

Thus, we see huge improvement in the quality of trend prediction of TAC values. Expect at 30th day where the trend is terribly wrong, all other points are fairly accurate. Another issue with the newly proposed optimization module is that it is insensitive to suddenly shock variations to the data set.

These can be speculated to happen because optimization is done using linear regression. Linear regression is a simple technique, however having many pitfalls. For example, it is sensitive to outliers. The reason for this is that since the least squares regression is concerned with minimizing the sum of the squared error, any training point that has a dependent value that differs a lot from the rest of the data will have a disproportionately large effect on the resulting constants that are being solved for. Thus, for the sudden shock variations in TAC variation near the 30th day of the test set, regression gets largely mistaken for the trend.

Also, forecasts insensitive to shock variations can be seen at many instances in plot is because these variations contribute a lot to non-linearity of the points in the window of concern.

Similarly, the plot (fig. 7) shows attempts made by LSTM on Pollution univariate data set (On an arbitrary 100 points of test set) to forecast 3 future points (without optimization) in a gap of 15 points on test set:

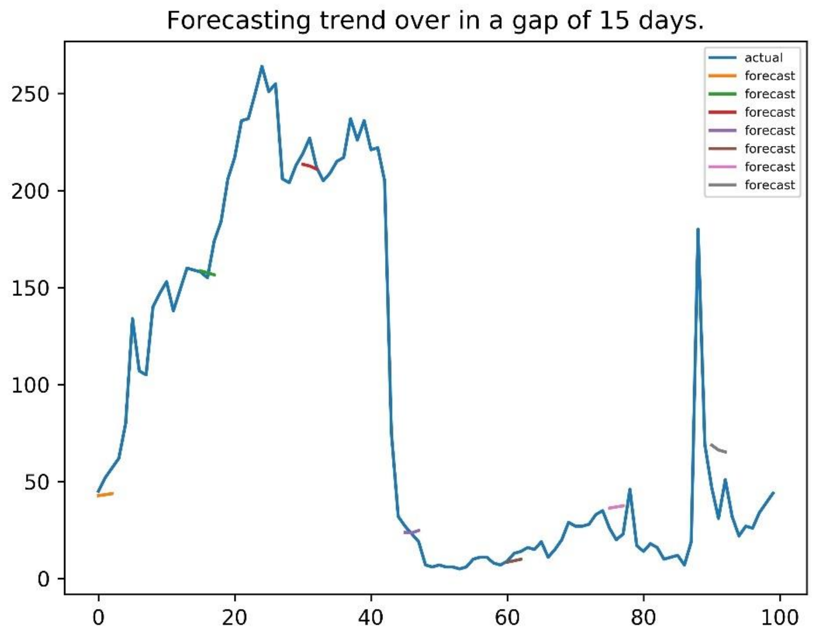


Figure 7. LSTM's forecasting over the Beijing pollution test set without optimization.

The average of all MAE losses for the forecasts are 9.436

Now with optimization, the plot looks like:

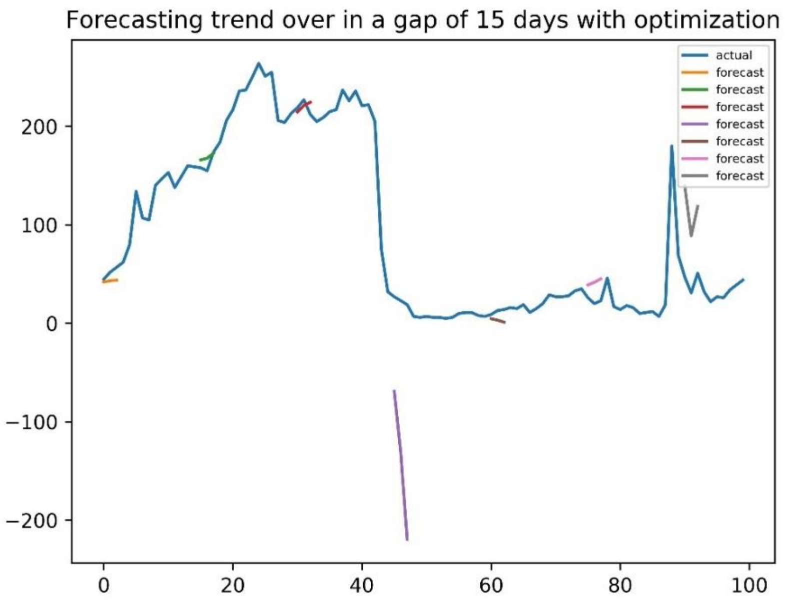


Figure 8. LSTM's forecasting over the Beijing pollution test set with optimization.

With optimization the trend predicted is more legible (except for a big mistake, the purple line)

Especially the forecast after 80th instance takes the exact 'V' shape, however shifted higher than the ground truth values.

The average of all MAE losses for the forecasts are 20.58. This is higher than the previous plot because of the up/down shift of the trend forecasts.

<u>4.1.4.1.2 Discussion</u>

Thus, we propose that stacked LSTMs along with suitable optimization using regression techniques can give promising results, especially when there is limited training data. Further comparison of losses from every other model will be discussed elaborately in the later sections of the report.

<u>4.1.4.2 seq2seq Encoder Decoder network</u>

The encoder-decoder model provides a pattern for using recurrent neural networks to address challenging sequence-to-sequence prediction problems, such as machine translation. Sequence to Sequence Learning with Neural Networks [57] was introduced in 2014.

Below diagram (fig. 9) summarizes the seq2seq architecture in context of machine translation.

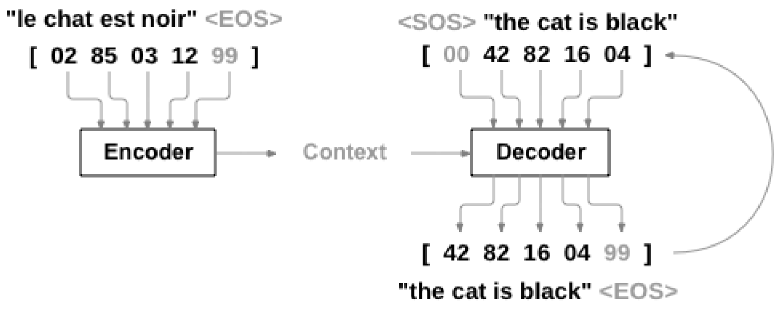


Figure 9. The seq2seq architecture.

Here we attempt to use the concept for time series forecasting for multivariate Beijing PM 2.5 data set. The motivation here is to train a model to encode all the features of the input and then train another model (decoder) to forecast.

Architecture:

Encoder:

LSTM (Input shape =8, Output shape = 51) →**LSTM** (Input shape =51, Output shape = 100) →**LSTM** (Input shape =100, Output shape = 100)→ **FC layer** (Output shape = 1)

Decoder:

LSTM (Input shape =1, Output shape = 51) →**LSTM** (Input shape =51, Output shape = 51) → **FC layer** (Output shape = 1)

Thus, an Encoder Decoder network, is a model consisting of two RNNs called the encoder and decoder. The encoder reads an input sequence and outputs a single vector, and the decoder reads that vector to produce an output sequence. Unlike sequence prediction with a single RNN/LSTM, where every input corresponds to an output, the seq2seq model frees us from sequence length and order.

With a seq2seq model the encoder creates a single vector which, in the ideal case, encodes the "meaning" of the input sequence into a single vector — a single point in some N dimensional space of features. Importantly, the decoder uses the initial state vectors from the encoder, which is how the decoder obtains information about what it is supposed to generate.

Here we approach the encoder encoding all the dimensions of the Beijing pollution data set at time t into one-dimension pm2.5 value (at time t+1) and then using decoder greedily, to produce again one dimension pm2.5 value at (t+2). Thus, effectively we are attempting to forecast 2 time steps ahead, end to end.

Network was trained with L1Loss and Adam optimizer dynamically varying learning rate from 1e-1 to 1e-6.

4.1.4.2.1 Results

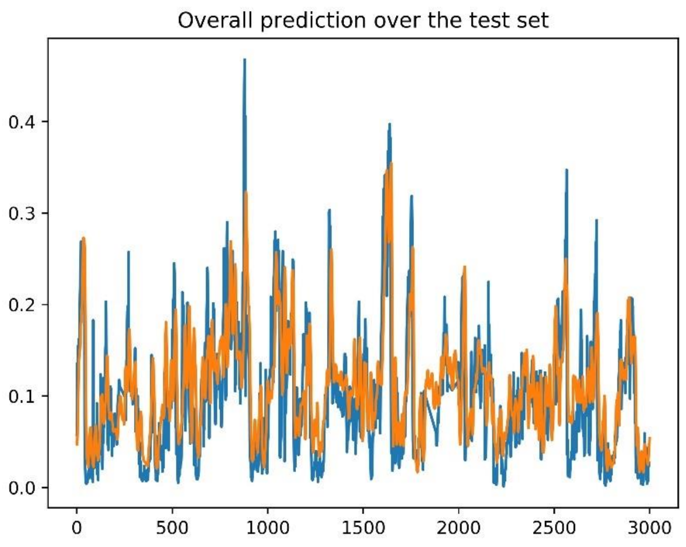


Figure 10. LSTM's prediction over the Beijing pollution test set.

The above plot (fig. 10) shows the LSTM's performance over the test set where the blue line is the ground truth and orange line shows the LSTM model's predictions.

This is plot spanning 3000 data points, hence, let us have a closer look.

The below plot (fig. 11) shows LSTM's performance over an arbitary 100 consecutive points of the test set.

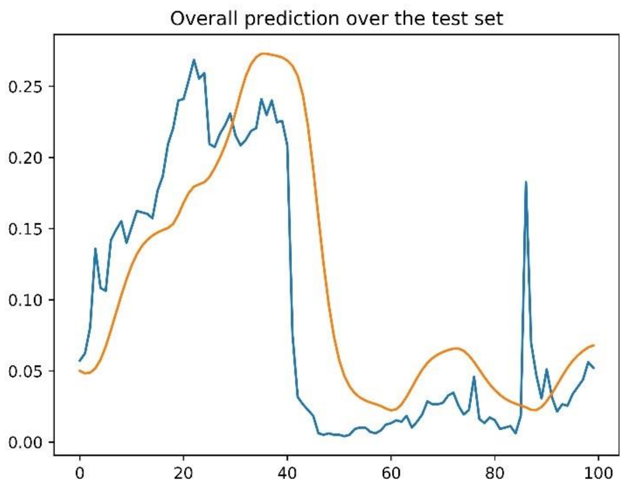


Figure 11. LSTM's prediction over 100 points of the Beijing pollution test set.

The L1 loss was calculated on the reverse normalized test data points and is found to be 37.86 and R2 score is 0.541.

Forecasting for 2 time instances is plotted as seen below:

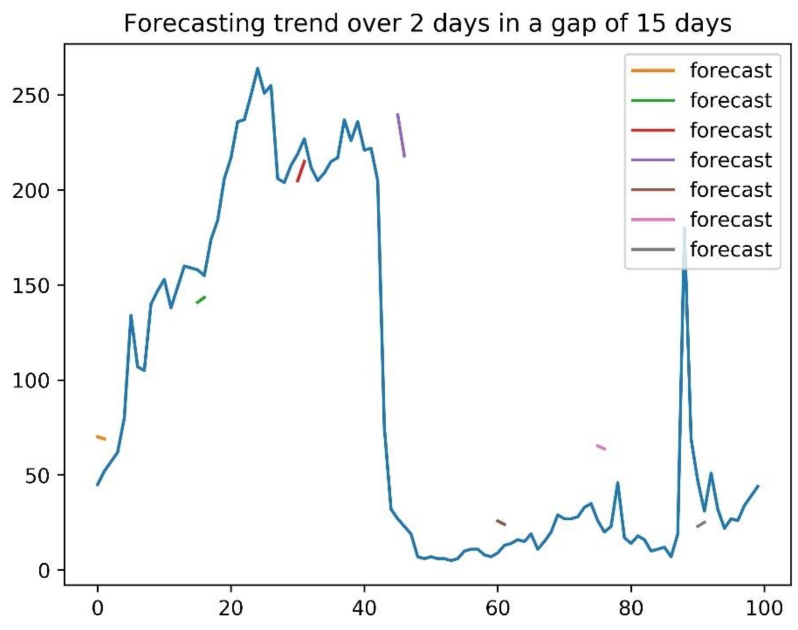


Figure 12. LSTM's forecasting over 100 points of the Beijing pollution test set.

Forecasting 2 points ahead is default configuration output for this encoder-decoder setup as encoder was trained to predict (t+1)th instance and decoder for (t+2)th.

Now we attempt to feedback the output of decoder back to decoder to produce next output. The plot shows what happens then.

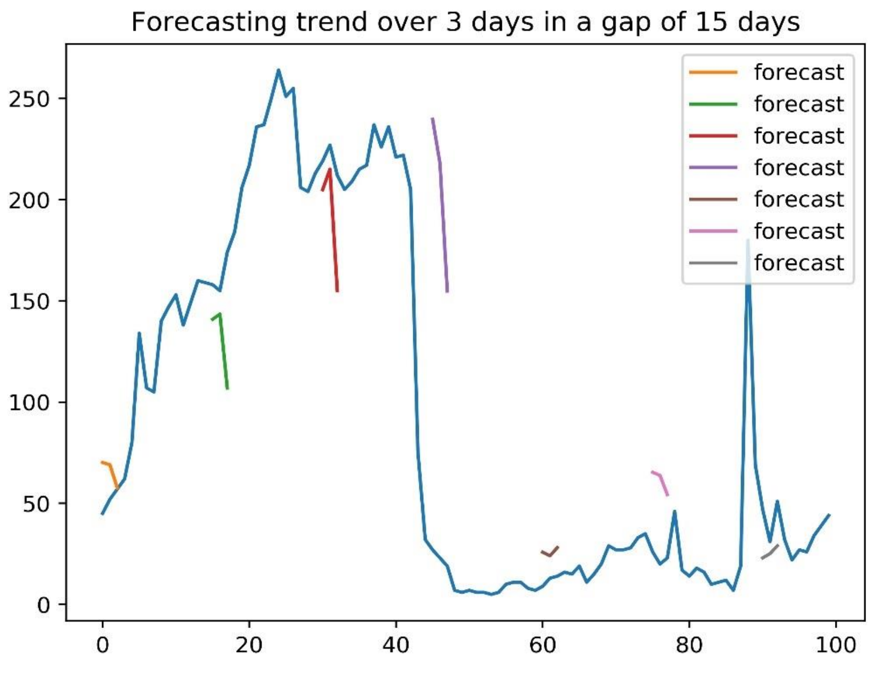


Figure 13. LSTM's forecasting over 100 points of the Beijing pollution test set.

The average of all MAE losses for the forecasts are 46.33.

Now using optimization,

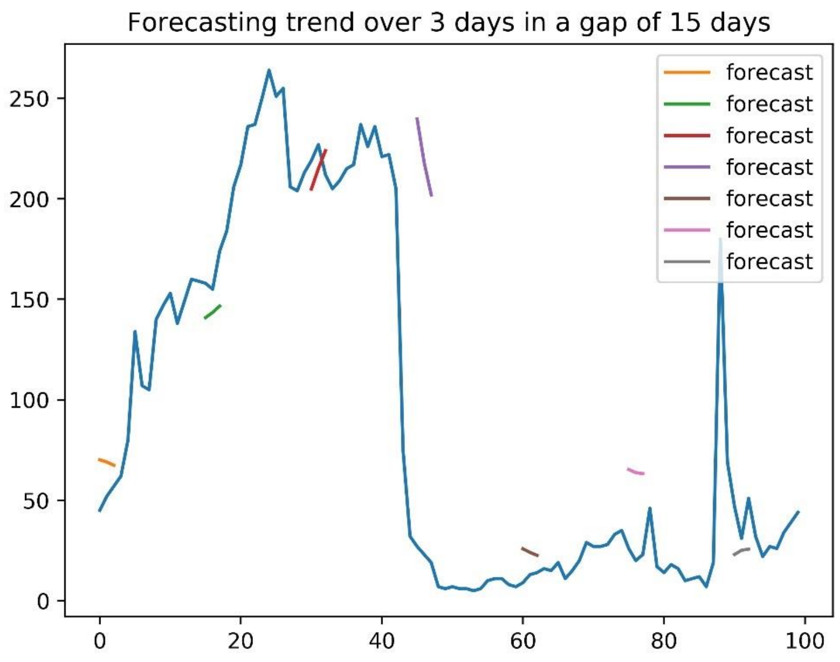


Figure 14. LSTM's forecasting over 100 points of the Beijing pollution test set.

As expected the forecasts has become more legible than in the previous plot. However, overall this architecture doesn't seem much promising.

The average of all MAE losses for the forecasts are 45.33

<u>4.1.4.2.2 Discussion</u>

Compared to the stacked LSTM univariate approach, the encoder-decoder approach doesn't show much promise in prediction / forecasting. This is counter intuitive to the fact that a natural phenomenon like Pollution is contributed by a lot of factors. The encoder-decoder setup, capable of handling multi variate data, couldn't approximate the data set well.

One would think why a greedy approach was taken to predict (t+2)th instance from t. Apparently this approach have caused this degraded performance by the LSTMs. However, from the perspective of forecasting, we are interested only in pm2.5 column, i.e. the pollution levels, given multi variate data. Encoder was fed with 8 dimensions at time t and at train time, output loss (for back propagation) was found using (t+1)th instance of the pm2.5 column. Decoder was fed with univariate data output from encoder and at train time, loss was found using (t+2) th instance.

To forecast future pm2.5 values, we need to feed current instance back in. This would be impossible for a multivariate input network. This is made possible with the decoder network. Hence this encoder-decoder configuration was designed.

Univariate approach using stacked LSTMs performs better than multivariate LSTM model.

4.2 SVR based Time Series Forecasting

4.2.1 Introduction

One of the competing methods, which takes a major place in time-series prediction, is Support Vector Regression. Originally, Support Vector Machines (SVM) algorithm was proposed by Vapnik and his team in 1995 [51]. Initially SVM was used to solve pattern classification problems, but later this technique has been applied to problems of function estimation, signal processing, regression, and time series prediction [34]. SVM is called as SVR when applied to regression type of tasks. In recent years SVR has been implementing more and more in time series prediction.

4.2.2 Theoretical Background

Given a training set of data points {(x_1, y_1), (x_2, y_2), …, (x_m, y_m)}, where input $x_i \in X \subset R^n$, and target $y_i \in Y \subset R$, the form of nonlinear function for modeling is:

$$f(x) = w^T \varphi(x_i) + b,$$

where the input data x is mapped into a high dimensional feature space by non-linear mapping $\varphi(x)$, b is a bias term and w is a weight vector with the same dimension as the feature space [18].

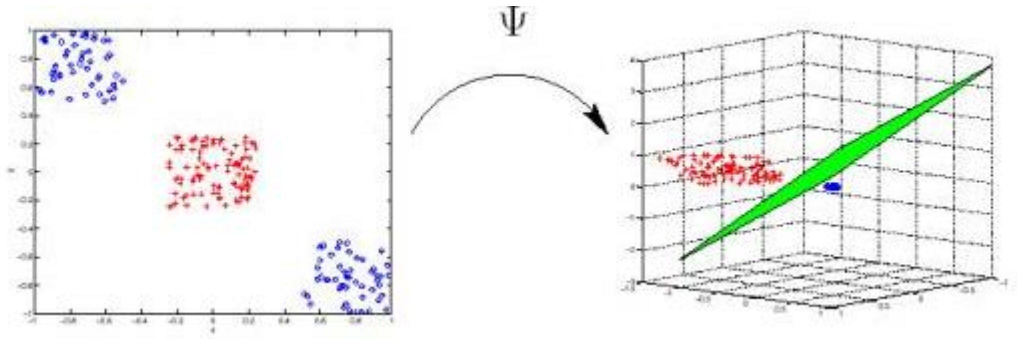


Figure 15. φ(x) mapping

The goal is to find a function $f(x)$ with the biggest deviation ε from the actual targets, but being as small as possible [15]. In addition, in order to have the function as flat as possible, it is required to minimize$w^T w$. Hence, the equation can be represented as a constrained convex optimization problem:

$$min \frac{1}{2} w^T w$$

$$s.t.: \begin{cases} y_i - (w^T \varphi(x_i) + b) \leq \varepsilon \\ y_i - (w^T \varphi(x_i) + b) \geq \varepsilon, \end{cases}$$

where ε (≥ 0) is defined by user [15]. The above equation can be solved by introducing slack variables ξ_i ξ_i^*, i = 1, …, n, and assuming that it is feasible. For this purpose, a constant term C > 0 is presented, which is a tradeoff between the flatness and the value of deviation greater than ε. Therefore, the equation is:

$$min \frac{1}{2} w^T w + C \sum_{i=1}^{n} (\xi_i + \xi_i \ )$$

$$s.t.: \begin{cases} y_i - w^T\varphi(x_i) - b \leq \varepsilon + \xi_i \\ w^T\varphi(x_i) + b - y_i \geq \varepsilon + \xi_i \ , \\ \xi_i, \xi_i \ \ \geq 0, i = 1, \dots, n \end{cases}$$

Thus, the term $C\sum_{i=1}^{n}(\xi_i + \xi_i \ \ )$ measures the ε -insensitive loss function$|\xi|_\varepsilon$, where $|\xi|_\varepsilon \begin{cases} 0 if |\xi| \leq \varepsilon \\ |\xi| - \varepsilon else \end{cases}$ [19]. The slack variables ξ_i ξ_i^*represents the size of upper and lower deviations from ε (Fig 9). The black lines outside the tube are support vectors, the slope if defined by C, and the decision function is determined regarding data points outside the tube [15].

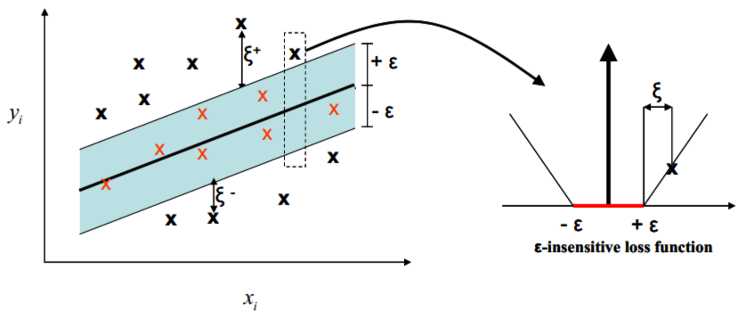


Figure 16. ε - intensive loss function.

The idea is to build a Lagrange function with the help of introducing Lagrangian multipliers (αi+, αi-, ηi+, ηi-). After making some reformulation, a dual problem to the eq can be achieved:

$$max \frac{1}{2}\sum_{i,j=1}^{n} K(x_i, x_j)$$

$$s.t.$$

The solution of this problem is given by:

$$f(x) = \sum_{i=1}^{n}$$

which is the support vector regression expansion [19], where *n* is a number of support vectors, and $K(x_i, x_j)$ is a kernel function.

<u>Kernel functions used in SVR</u>

There exist several kernel functions in SVR. Most popular and widely used ones are a polynomial kernel function, a radial basis function kernel and a sigmoid kernel.

1) A polynomial kernel function can be described as:

$$K(x_i, x_j) = (\gamma(x_i \cdot x_j) + r|d), d = 1, \dots$$

2) A radial basis kernel (RBF) can be:

$$K(x_i, x_j) = exp\left(-\gamma\|x_i - x_j\|^2\right), \gamma \geq 0$$

3) A sigmoid kernel is constructed:

$$K(x_i, x_j) = tanh(\gamma(x_i \cdot x_j) + r),$$

where r is a bias andγ is a slope. Among these kernel functions, according to the researches [15], [20], [21], [22], [23], [24] and many others, the most promising to proper suit the time series and achieve high results is the RBF kernel. Thus, this kernel function is considered as the first choice in implementation step. Also choosing suitable parameters defines the structure of the high dimensional feature space [21] involving in better performance.

SVR parameters

Accurate choosing of kernel function, kernel parameters and meta-parameters C, ε lead to better estimation accuracy. An optimal selection of parameters is the topic of research, and there have been developed several different methods as [7], [25], [19], [26], [27] to select the best parameters. In this work grid search based on cross validation is implemented, as it is considered as a common approach, and is good for batch processing.

4.2.3 Data Preprocessing

Data preprocessing

Data preprocessing step is a crucial part before application any model. This allows accomplishing data prediction and forecasting more accurate and efficient. Several forecasting methods require the format of the input data [14], thus making data preprocessing as a necessary step. Depending on the raw data and forecasting methods applied later, it might be needed to perform several data transformations as a preprocessing. Data preprocessing used in this work can be divided into two main parts as data cleaning and transformation, these methods are described below.

Data cleaning

The first assessment of the data includes checking its quality, identifying missing values, outliers and errors. It might be essential to modify outliers, correct errors and infer missing values. It is relevant to implement data cleaning prior to fitting the data into SVR model [15], because on the other hand the model might predict non-valid results [16]. Visual inspection of the data itself and its graphical representation is one of the pertinent approaches for determining inconsistencies.

Missing value detection

If the data contains missing values, they can be filled either manually, which is not effective, or using interpolation.

Outliers detection

An outlier is a point, which is considerably divergent from other points in the dataset. It might be an incorrect observation, and it is better to remove from the dataset or replace by the mean of its neighboring measurements [14].

Transformation

Data transformation implies converting the data into appropriate form. Different data transformations are presented below.

Differencing

One of the common method to make the non-stationary data more stable is differencing. It can be presented as:

$$y_t = x_t - x_{t-1},$$

where original time series x_t is converted into a new y_t. Differencing can be applied several times, until stationary data is obtained. Due to this procedure the data points become dependent sharing two consecutive points in common from the original set [15].

Z-transformation

Z- transformation is one of the useful transformation for normalizing time series. It is appropriate to use when time series data have outliers [53] and is stationary. Z- transformation mainly focuses on structural similarities and dissimilarities [6]. This method requires that output's mean is zero and the standard deviation close to one [6]. Z-score transformation was performed as a pre-processing step in several works on the time series analysis [7], [8].

The formula of z-score transformation is:

$$z = \frac{x - \mu}{\sigma},$$

where x – data point at time t, μ – mean and σ – standard deviation of time series.

However, this method is not suitable for non-stationary time series, because of variation of the mean and standard deviation over time [9].

Logarithmic transformation

This transformation technique is more useful when the data is non-stationary, as it is deemed to stabilize the variance of a time series [9]. Thus, by applying natural logarithms to input data series a more stationary series is acquired.

$$\dot{x} = ln(x)$$

Usage of logarithmic transformation as a pre-processing step can be seen in works of Abecasis and Lapenta [10], Nogales et al [11], Nelson and Granger [12]. Moreover, Nelson and Granger demonstrated that using this transformation shows better results in forecasting than without this pre-processing [12].

Scaling

Min-max scaling

This method has been described in the previous section of the work. In comparison to standardization, this technique allows to obtain smaller standard deviations, which can overcome the impact of outliers [13].

4.2.4 Methodology and implementation

Grid search

Grid search method is widely used technique for exhaustively searching over specified parameters given in a grid for an estimator [17]. The output is the best combination of given parameters in the grid for the data.

Cross-validation

While performing grid search on evaluating parameters a 10-fold cross-validation is used on a training set.

Implementation

The following section describes the implementation of SVR method on TAC and Beijing pollution dataset. Firstly, modelling on TAC dataset with results and summary is presented

4.2.4.1 Tac index dataset

4.2.4.1.1 Pre-processing

From the visual inspection and analysis, the data does not have any missing values or errors. The data is non-stationary, thus before fitting the data into SVR, data transformation such as logarithmic has been implemented. Logarithmic transformation is taken, as it is considered to be more appropriate to this nonlinear dataset. Thus, subsequently a more stable data has been obtained to fit the model. However, for the comparison SVR technique has been applied for the data with and without logarithmic transformation, and the results is given in the following part.

Next stage is to transform regression time series data into supervised learning data, which has been obtained using Sliding window technique (described in the previous part).

Train and test set is split as 1000 and 58 points accordingly.

SVR is performed using sklearn library in Python.

4.2.4.1.2 Implementation

Grid-search with 10-fold cross-validation is implemented on the train set, using sklearn.model_selection.GridSearchCV library in Python. Parameters set includes values of C [1, 5, 10, 100, 1000], ε [0.1, 0.01, 0.001], γ [0.1, 0.01, 0.001] for RBF kernel. For comparison linear, polynomial and sigmoid kernels are fitted. Certainly, RBF kernel demonstrates the best results, thus it is performed not only on t+1 step, but also on further prediction of t+2, t+3 steps.

In [58] Cauchy kernel has performed successful results for time series prediction. It can be derived as:

$$K(x_i, x_j) = \frac{1}{1 + \frac{\left\|x_i - x_j\right\|^2}{\sigma^2}}$$

This kernel can compete with well suited RBF kernel, because of its ability to capture extreme values of financial time series data [58, 59]. This type of kernel is better suited for the financial time series [58, 59]. Thus, in addition, Cauchy kernel has been performed. Parameters set includes values of C [1, 5, 10, 100, 1000], ε [0.1, 0.01, 0.001], σ [0.01, 0.1, 1.0, 1.5, 2].

The best performance on RBF kernel and Cauchy kernel with found optimal parameters are given below. As it is stated previously, SVR is executed on original and transformed dataset.

4.2.4.1.3 Results and discussion

Note: here for comparison only, SVR on the original dataset is included too, however, as a preprocessing step it is usually required to transform the data before applying SVR model.

Data	RBF Kernel				
	C	ε	γ	MAE	R2_score
Original dataset	5	0.001	0.01	0.024043	0.9671
Transformed dataset	100	0.001	0.001	0.006560	0.9667
	Cauchy Kernel				
	C	ε	σ	MAE	R2_score
Original dataset	1	0.0001	0.5	0.053637	0.8186
Transformed dataset	10	0.001	1	0.006673	0.9653

Table 3. Performance of SVR on original and transformed dataset.

From the table 3, transformed data provides better results, than the original one with smaller MAE value and bigger R2_score, which makes it essential to pre-process – transform the data before using it directly to the model. In addition, Cauchy kernel could not outperform RBF kernel model, nevertheless, as stated in [58, 59] that Cauchy kernel best fits the financial data.

The figure below (fig. 16) shows a prediction on test set of TAC dataset. This plot is drawn according to the best obtained parameters. The graph shows t+1 prediction. The plot indicates the perfect follow of the trend and increase / decrease in directions of the predicted values t+1 between original test values. Thus, the model successfully fits the given time series.

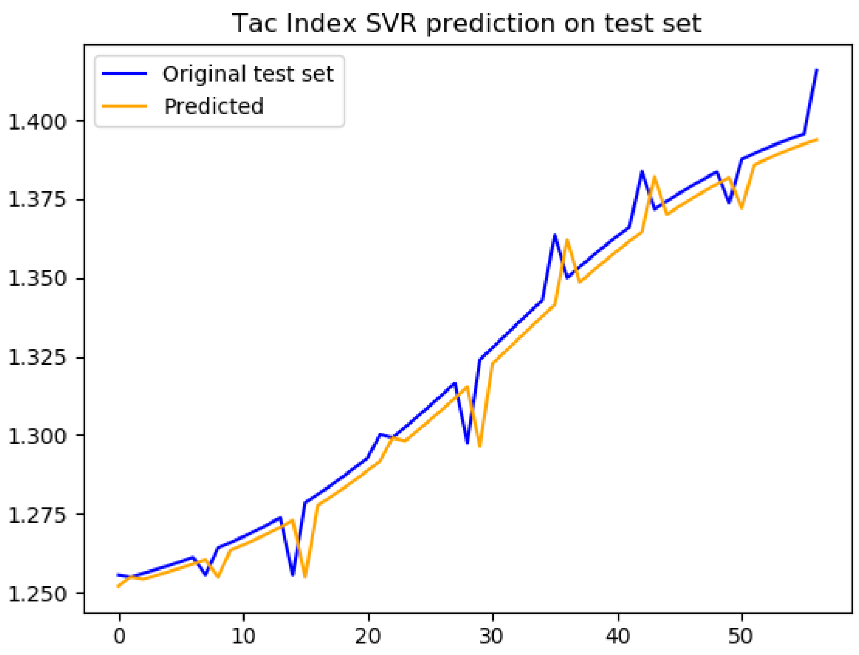


Figure 16. SVR prediction on TAC test set.

Forecasting

The figure below (fig. 17) shows prediction on test dataset, forecasting 3 future point in a gap of 10 days. The best result is obtained on t+1 prediction, as period of future prediction increases, t+3, it is getting more inaccurate.

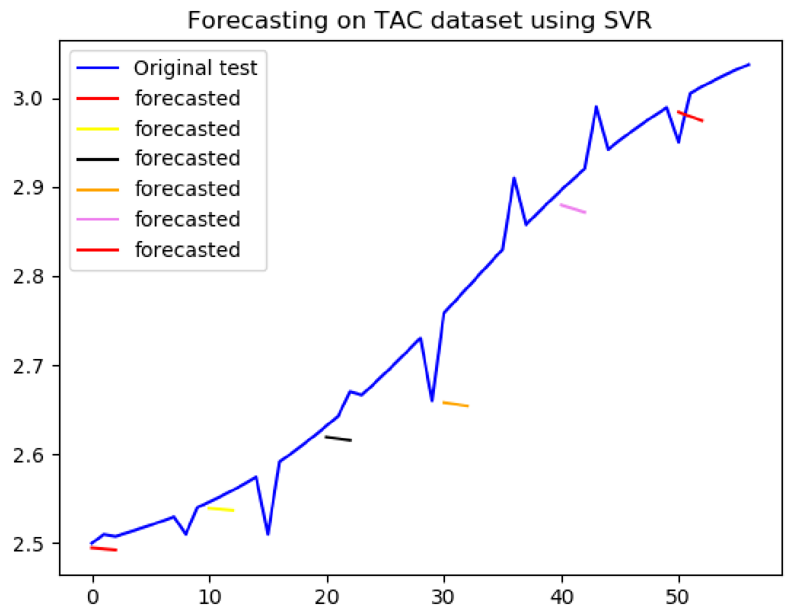


Figure 17. SVR Multistep ahead prediction on TAC test set.

Forecasted period	MAE on original data	R2_score on original data	MAE on transformed data	R2_score on transformed data
t = 0, t+3	0.016	-97.203	0.005	-129.563
t = 10, t+3	0.0215	-14.570	0.007	-19.359
t = 20, t+3	0.042	-12.704	0.012	-14.804
t = 30, t+3	0.131	-124.180	0.036	-134.933
t = 40, t+3	0.064	-3.327	0.017	-3.555
t = 50, t+3	0.033	-33.555	0.007	-25.568

Table 4. Performance of SVR multistep ahead prediction on TAC test set.

From the table 4 it is implied that the best performance is obtained on transformed data. Furthermore, r2_score provides negative results, meaning that the predicted shape of a line and its direction is different from the original shape of the data. Average MAE value on original data is 0.051, and on the transformed data is 0.014.

4.2.4.2 Beijing dataset

4.2.4.2.1 Pre-processing

Prior to implementing model prediction technique data preprocessing was required. According to the first visual inspection, the data contains several missing values, which has been interpolated. As stated in the data description part, data itself is stationary. Thus, z-score transformation has been applied, which is well suited for the stationary data. (For the comparison, the results are given for scaled dataset and transformed dataset).

Transform of the regression time series data into the supervised learning data, has been obtained using Sliding window technique (described in the previous section).

Train and test set is split as 12000 and 3000 points accordingly. (Previously train and test has been split as 35039 and 87600 accordingly, however, because of tremendous computational resources and time requirements, it has been decided to reduce number of samples).

SVR is performed using sklearn library in Python.

4.2.4.2.2 Implementation

Here also a Grid-search with a 10-fold cross-validation is implemented on the train set, using sklearn.model_selection.GridSearchCV library in Python. As it is stated previously, SVR is executed on the original and transformed dataset.

Prediction t+1

Parameters set here also includes the same values as for the previous data of C [1, 5, 10, 100, 1000], ε [0.1, 0.01, 0.001], γ [0.1, 0.01, 0.001] for RBF kernel. (RBF has provided better results than polynomial kernel of d [0, 1, 2, 3]).

4.2.4.2.3 Results and discussion

The table below (5) shows the result of SVR with the best parameters.

Data	C	ε	γ	MAE	R2_score
Scaled dataset	1000	0.001	0.01	11.443	0.922
Transformed dataset	10	0.01	0.01	0.127	0.922

Table 5. Performance of SVR on original and transformed dataset.

There is a significant difference in results between scaled data and transformed data set. As it can be seen from the table, it is better to transform the data before using SVR model to have smaller MAE value. Also, R2_score is 0.922, is close to 1, which is considered as well fitted model. Thus, for further forecasting the results given on transformed data (and as it has been defined in the data preprocessing step, it is appropriate to use transformation (zscore) for this data).

The graph of the prediction on the test set is given below:

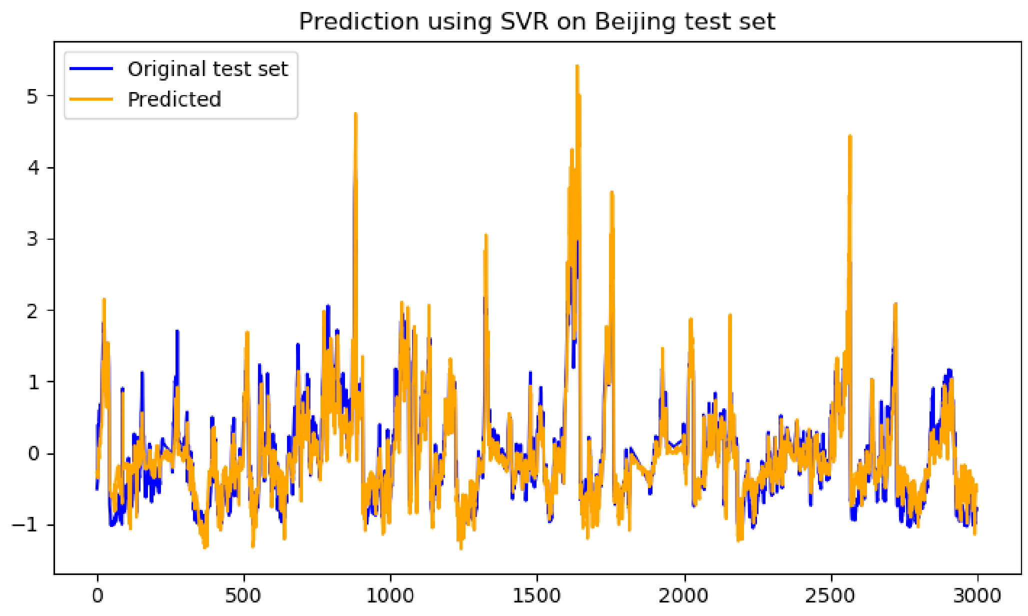


Figure 18. SVR prediction on PM2.5 test set.

From the graph, t+1 prediction on test set gives an appropriate shape and direction, following all the spikes. Although, it is better to see the performance on smaller number of test points (100) on the graph, which is given below:

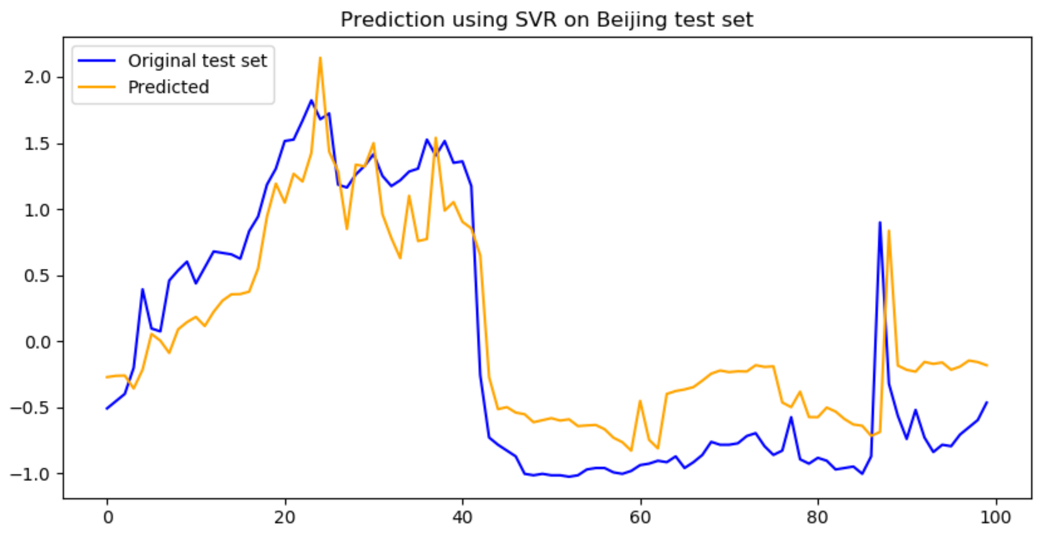


Figure 19. SVR prediction on PM2.5 test set with 100 point.

After reducing the size for better visual representation, it can be seen, that the model still follows the direction, peaks and falls of the original trend, however, the shape is not perfect. Although, relatively small MAE and big R2_score implies for a proper model fit for t+1 prediction.

Forecasting

As the transformed dataset gives better results, multistep forecasting is given on the transformed data with the best parameters obtained.

The table below (6) shows a forecasted value of 3 future points for the test set with a gap of 15 days. MAE values and R2_score on 3 consecutive points is acquired by comparing 3 forecasted points with the ground truth. Small MAE values indicate relatively small difference between compared values, while in all cases there is a negative R2_score. Negative values of R2_score is defined of different shape (direction) of the model fitted and original values. Average MAE value is 0.184.

Forecasted period	MAE on transformed data	R2_score on transformed data
t = 0, t+3	0.082	-2.444
t = 15, t+3	0.155	-0.791
t = 30, t+3	0.099	-0.177
t = 45, t+3	0.181	-5.972
t = 60, t+3	0.091	-44.730
t = 75, t+3	0.139	-0.203
t = 90, t+3	0.355	-12.307

Table 6. Performance of SVR multistep ahead prediction on PM2.5 data.

As it can be indicated from the figure below, the shape of the forecasted values is different from the shape of original values, giving negative R2_score.

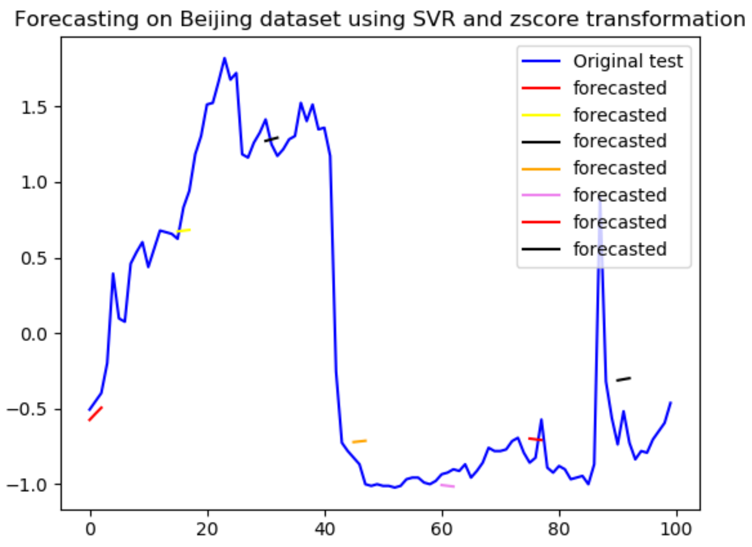


Figure 20. SVR Multistep ahead prediction on PM2.5 data.

<u>4.2.5 Summary on SVR:</u> To sum up, it is better firstly to transform the data properly regarding its characteristics before fitting to the model. It is also useful to find the best tuned parameters such as a Kernel type, global C and ε values, as well as local depended on the Kernel type. For this

purposes Grid-search and cross-validation are useful. The best kernel type is RBF, which allows fitting the model precisely. T+1 prediction gives relatively successful results, based on two completely different datasets, while multistep forecasting fails to have proper R2_score in both cases. Thus, the time series prediction is better performed on short-term prediction of t+1 values.

4.3 Autoregressive Integrated Moving Average (ARIMA) based Time Series Forecasting

4.3.1 Introduction:

Autoregressive Integrated Moving Average (ARIMA) also known as Box-Jenkins model is a popular univariate time series model in which time series data is fitted to forecast future values. It is based on the combination of both AR (Auto Regression) and MA (Moving Average) by exploiting the information about correlation between lagged values in stationary time series data.

4.3.2 Theoretical Background:

Auto Regression (AR):

ARIMA models are used on stationary series. When the values of time series fluctuate around the mean, the naive prediction for the next forecast will be its mean with some error.

$$y_t = \mu + error$$

However, in the most cases we observe that the value of the error will be correlated with previously observed values of time series [66]. Thus, some parts of the error can be corrected by adding terms, which are multiple of previously observed values. We can find the parameter values by using regression:

$$y_t = \mu + \varphi_1 y_{t-1} \ldots\ldots \varphi_k y_{t-k} + e_t,$$

Where y_t is the value of stationary time series at time t, μ is constant, e_t is the error term also known as shock, coefficients $\emptyset_{1\ldots.k}$ are calculated by regression. This is also known as AR (k) model, where k is the number of lags on which a future value is dependent.

Moving Average models (MA):

Moving average models in contrast to AR models use past forecast errors instead of past values [67]:

$$y_t = c + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} \ldots\ldots + \theta_q e_{t-q}$$

Where c is a constant and $e_t$are errors at respective time periods.

Autoregressive Integrated Moving Average (ARIMA):

We can use one of the above to forecast t. Although, in practice, it is observed that the combination of AR and MA give better results. This combination is called ARIMA model. ARIMA model is generally represented as ARIMA (p,d,q), where p denotes the number of auto-regression terms, d is the differentiation required to make the time series stationary and q is the number of MA terms [66].

General forecasting equation for ARIMA is given:

$$y_t = c + \emptyset_1 y_{t-1} + \cdots + \emptyset_p y_{t-p} + \theta_1 e_{t-1} + \cdots \theta_q e_{t-q} + e_t$$

4.2.3 Methodology and implementation

Beijing dataset

Plotting:

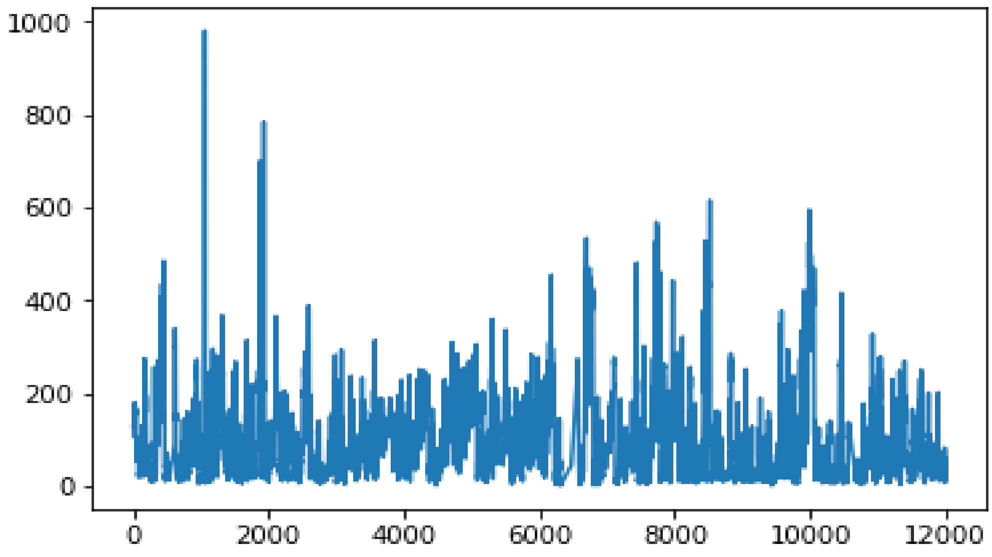


Figure 21. Plot of pm2.5 training data

It is not clear whether the data is stationary form the plot, hence we use other methods to determine if the data is stationary [68].

Auto correlation Function (ACF):

Auto correlation is the correlation between the values in the time series with its delayed values. It helps in determining how well the values in the time series are correlated with its lagged values. The number of MA terms required is determined by ACF [69].

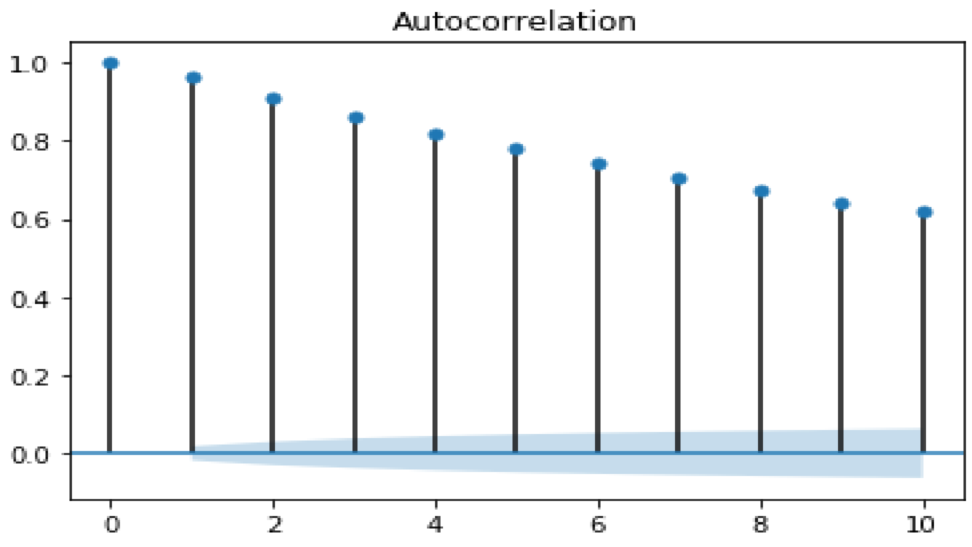


Figure 22. Auto correlation for various lags on pm2.5 data

When the data has significant auto correlation for more than 10 lags the data needs to be differentiated by a higher order [70].

The plot of differenced data:

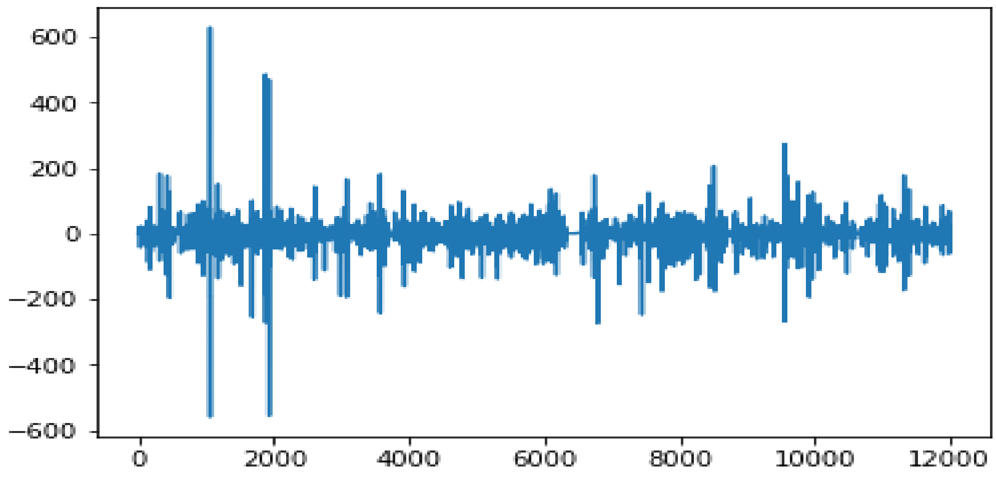
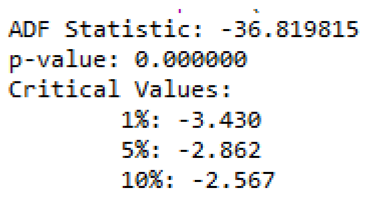


Figure 23. pm2.5 after 1st order differencing

The differenced data is now fluctuating around the mean, behaving as stationary. It can also be confirmed by using Augmented Dickey fuller test:

<u>Augmented Dickey fuller test:</u>

Augmented dickey fuller test helps in conforming the stationarity of a time series. The null hypothesis states that the series contains unit root and it is non-stationary, while the alternative hypothesis states that the series is stationary. The more negative the test value is the strong we reject null hypothesis



```
ADF Statistic: -36.819815
p-value: 0.000000
Critical Values:
        1%: -3.430
        5%: -2.862
        10%: -2.567
```

<u>Number of AR and MA terms</u>

Now the important goal is to find number of AR and MA terms. This can be done by observing the Auto correlation (ACF) and Partial Auto correlation (PACF) of stationary series [71].

ACF plot of differenced data

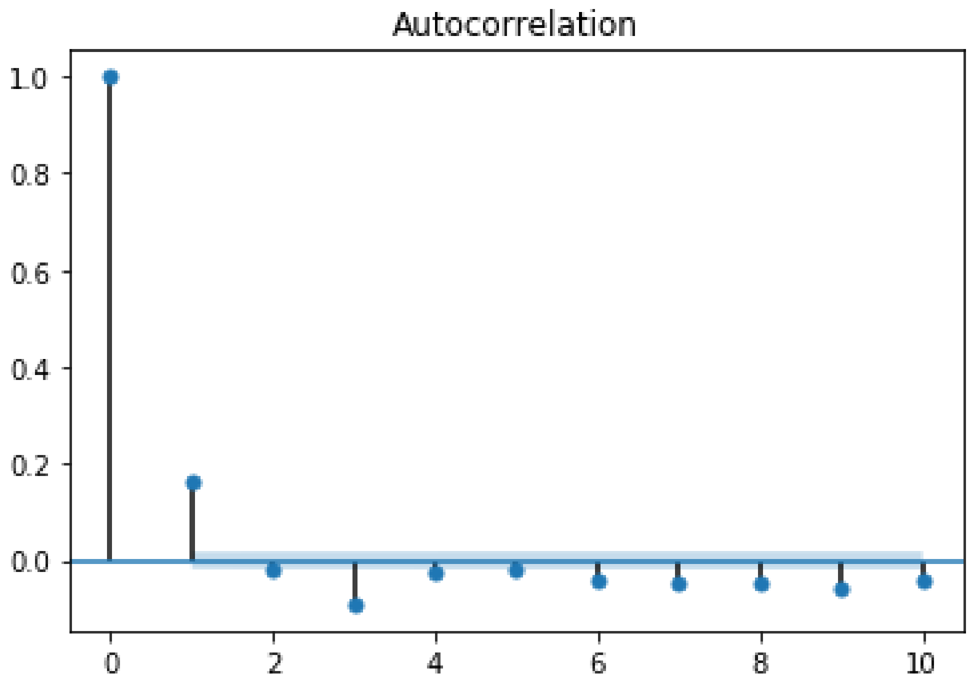


Figure 24. Auto correlation plot of differenced pm2.5 data

Partial Autocorrelation Function (PACF):

While auto correlation helps in finding if the values in time series depend on the previous values, it does not explain to what extent only the values at particular lag affect, because in auto correlation there is also effect of intermediate lags. PACF eliminates the effect of intermediate lags. Thus, the number of AR terms is necessary to be determined by PACF plot [72].

PACF plot for pm2.5 dataset:

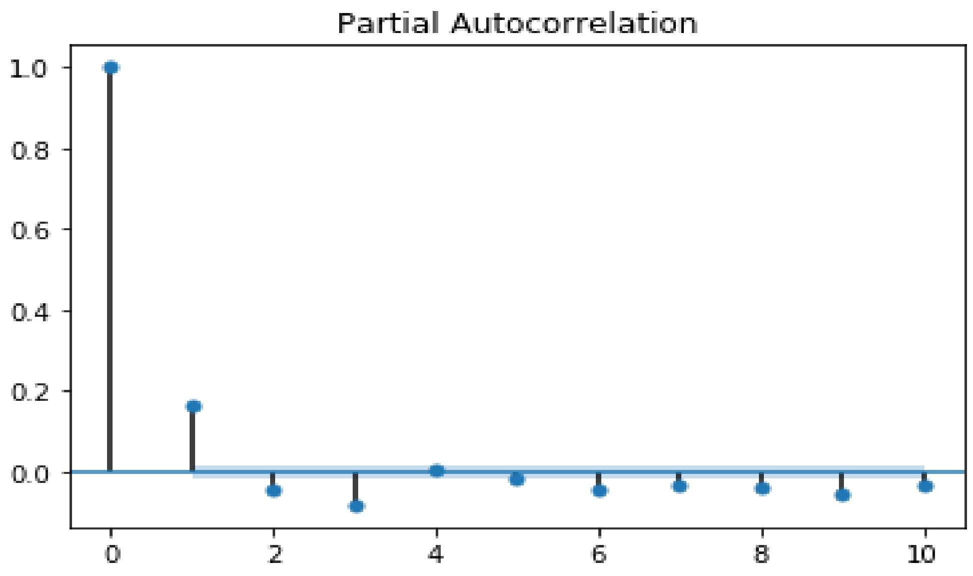


Figure 25. Partial Auto Correlation (PACF) plot for pm2.5 data

From ACF and PACF plots we can see that pm2.5 time series should be modelled with ARIMA (1,1,1).

Model fit of ARIMA

Figure 26 shows the prediction (fit) of ARIMA on pm2.5 training data.

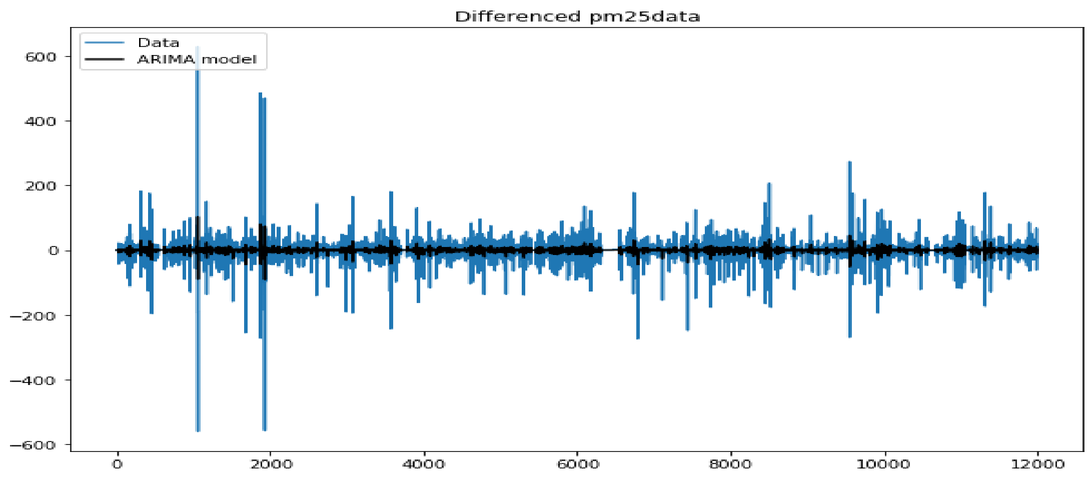


Figure 26. Model fit of ARIMA on differenced pm2.5 data

4.2.3.1 Results on Beijing PM2.5 pollution data

As ARIMA model is now using differenced data we need to indifference the values forecasted to get required forecasted values.

Short term forecasts with ARIMA model (2 values forecasted)

The below figure 27 shows 2 values forecasted using ARIMA at intervals of 10 over last 100 points of test data

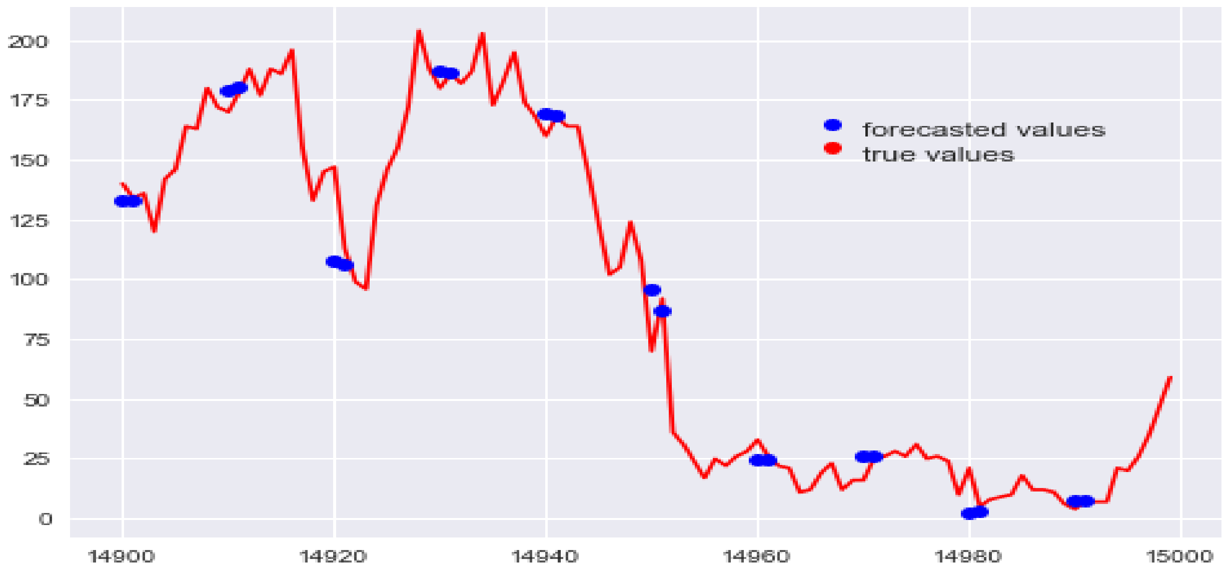


Figure 27. 2 values forecasted at regular intervals over last 100 points of pm2.5 test data

From figure 27 we see that ARIMA models are generally good for the short-term forecasts.

Medium term forecasts with ARIMA model (5 values forecasted)

We now try to forecast for an increased range using ARIMA model. Fig 28 shows 5 values forecasted at intervals of 10 (fitting all the previous data to ARIMA model) over last 100 values in pm2.5 test data

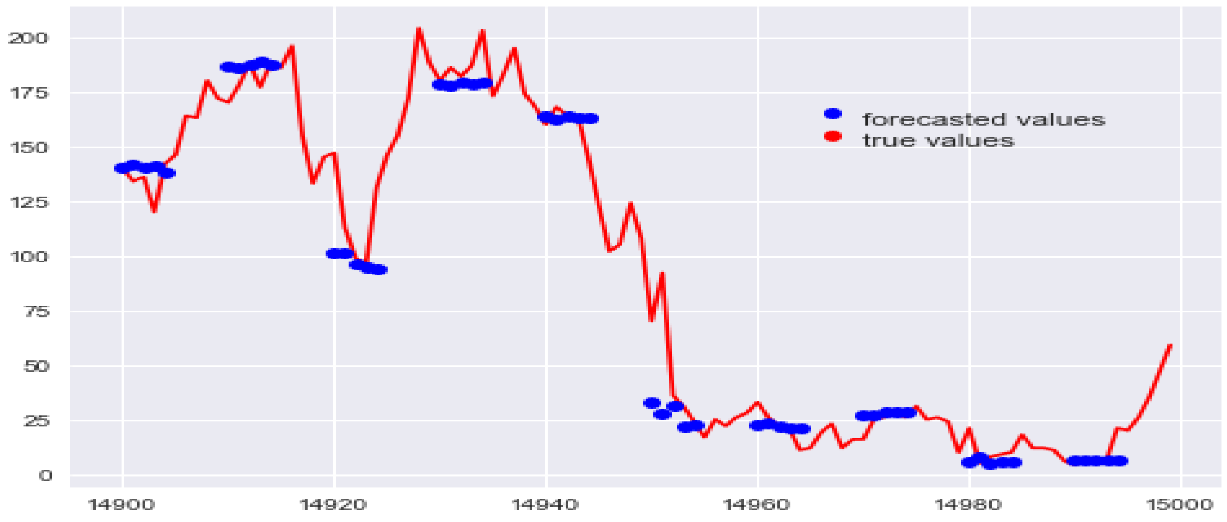


Figure 28. 5 values forecasted at regular intervals over last 100 points of pm2.5 test data

As we can see, although the forecasts done by ARIMA model is still satisfactory. ARIMA model tends to forecast badly as the forecast range increases

Long term forecast

Figure 29 shows 100 values forecasted by ARIMA model over last 100 points of test data set

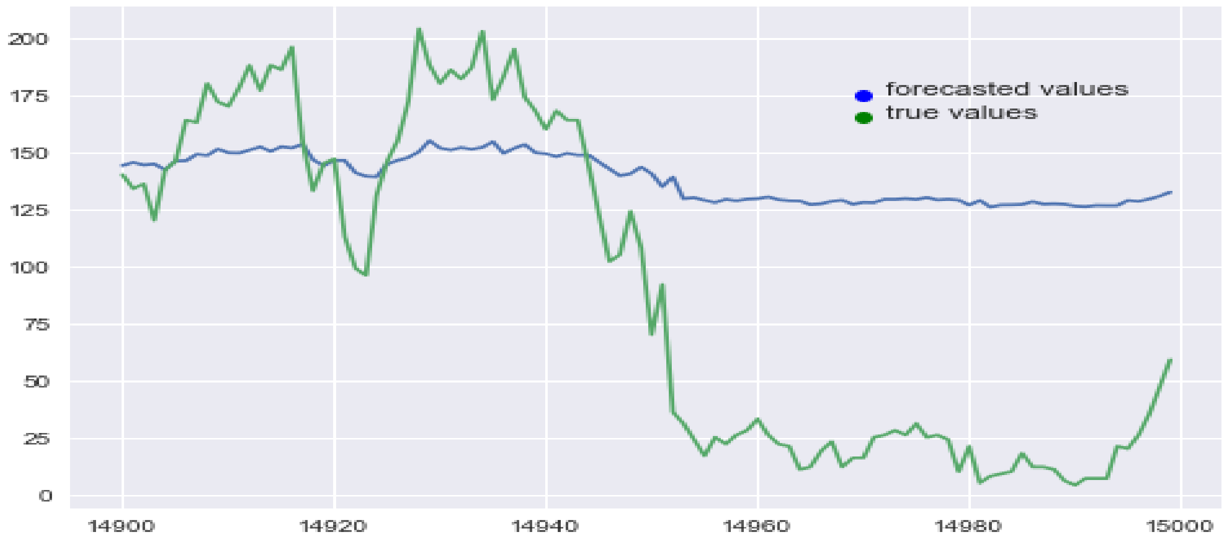


Figure 29. 100 values forecasted over last 100 points of pm2.5 test data

According to the properties, long range forecasts done by ARIMA model converges to the mean. Therefore, ARIMA models usually are used only for short term forecasting.

Forecast	MAE-score	R2-score
Short(next 2 values)	12.89	0.91
Medium(next 5 values)	25.71	0.68
Long(next 100 values)	64.5	-0.204

Table 7. Performance of ARIMA model on Beijing Pollution data.

ARIMA on TAC Data:

Plot of TAC data (training):

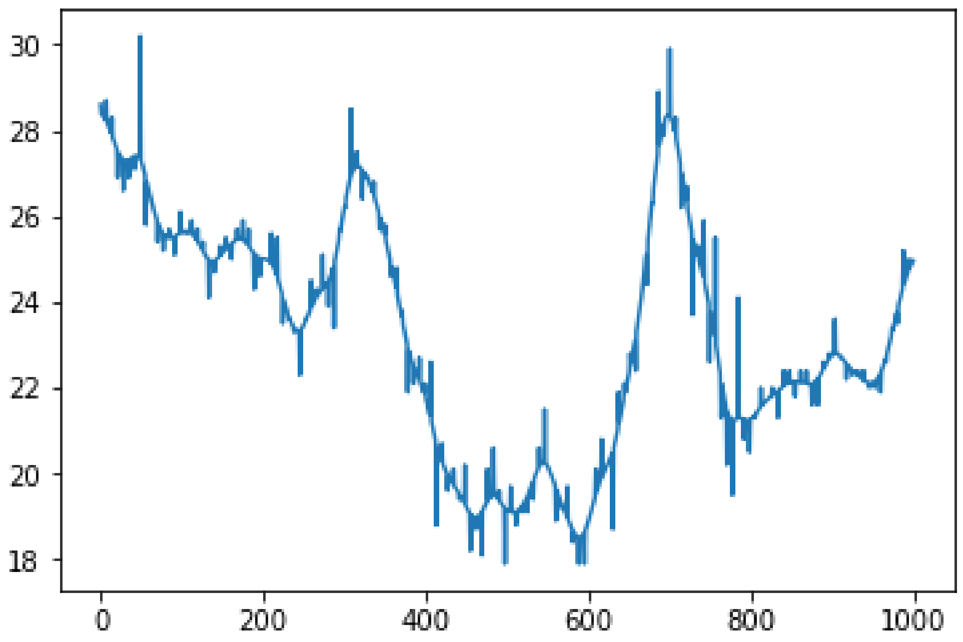


Figure 30. Plot of TAC data set

Clearly, the data looks non-stationary. We can differentiate it to get stationary series

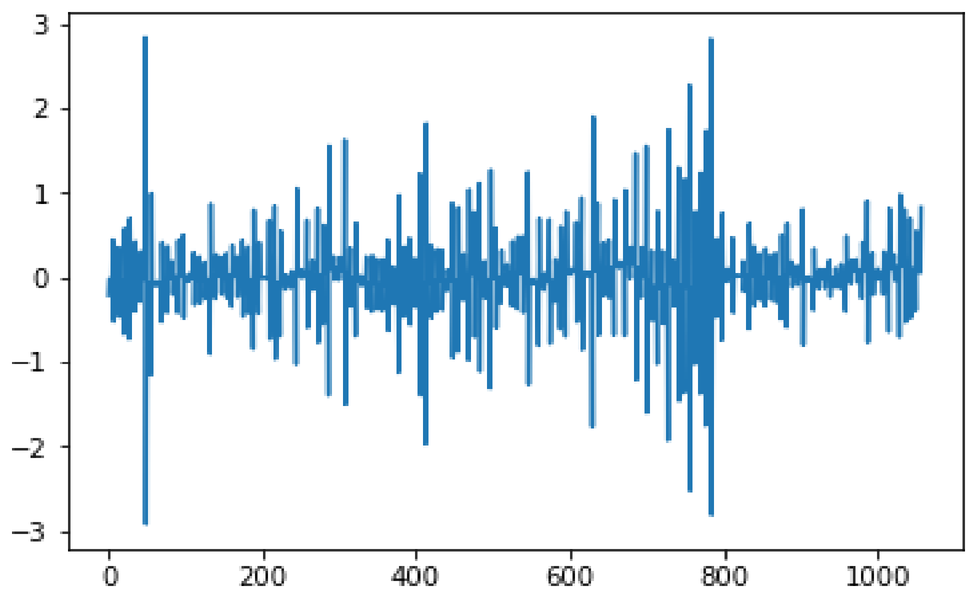


Figure 31. Differenced TAC data

Augmented Dicky fuller test:

```
ADF Statistic: -4.123391
p-value: 0.000886
Critical Values:
        1%: -3.437
        5%: -2.864
        10%: -2.568
```

The value of ADF test is more negative than the critical region so we confirm that the series is stationary.

Auto correlation Plot:

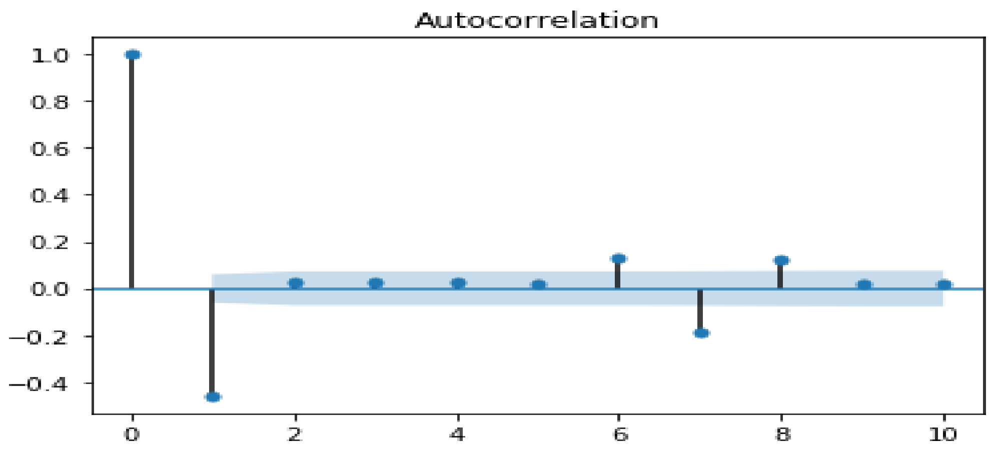


Figure 32. Auto correlation plot of differenced TAC data

The auto correlation plot shows the number of MA terms as 1.

Partial Auto correlation plot:

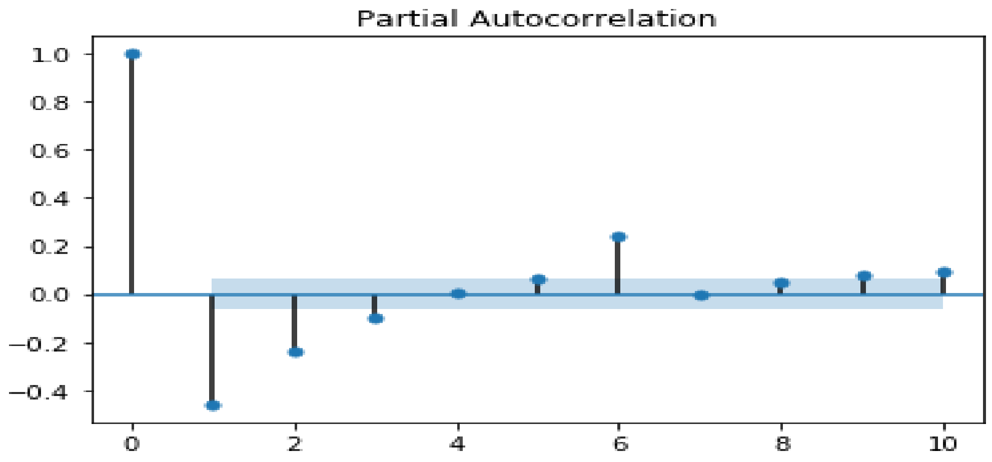


Figure 33. Partial Auto correlation plot of differenced TAC data

From PACF plot shows the number of AR terms as 2.

Fitting ARIMA (2,1,1) model

Model fit of ARIMA

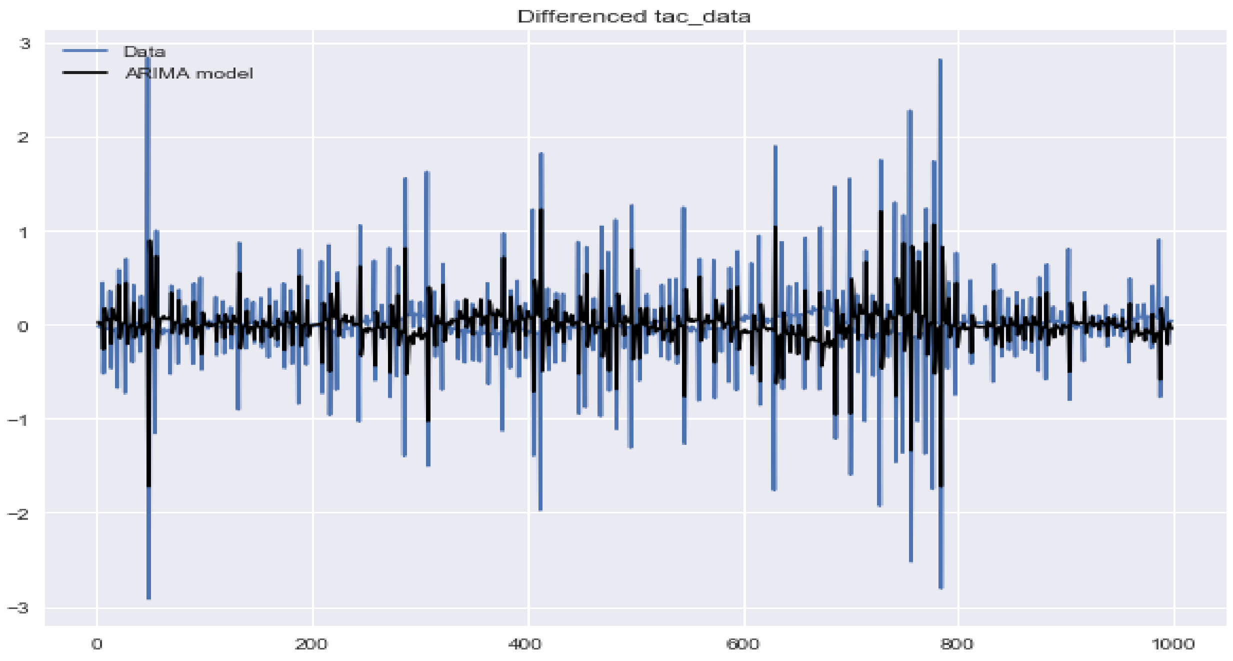


Figure 34. Model Fit of ARIMA over TAC data

4.2.3.2 Results on TAC data

ARIMA short term forecast

As figure 35 shows short term forecasts done by ARIMA for TAC data is satisfactory

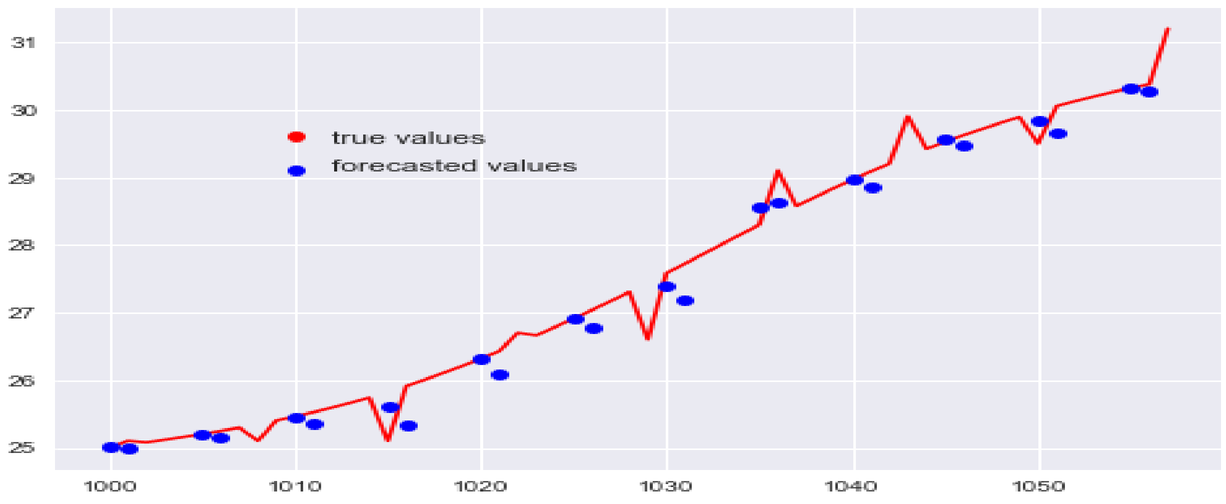


Figure 35. 2 forecasted values at regular intervals over last 100 test data

ARIMA mid-range forecast

From figure 37 we can observe that ARIMA forecasts on TAC data starts getting bad as forecast range increases especially at points where fluctuations in data are high

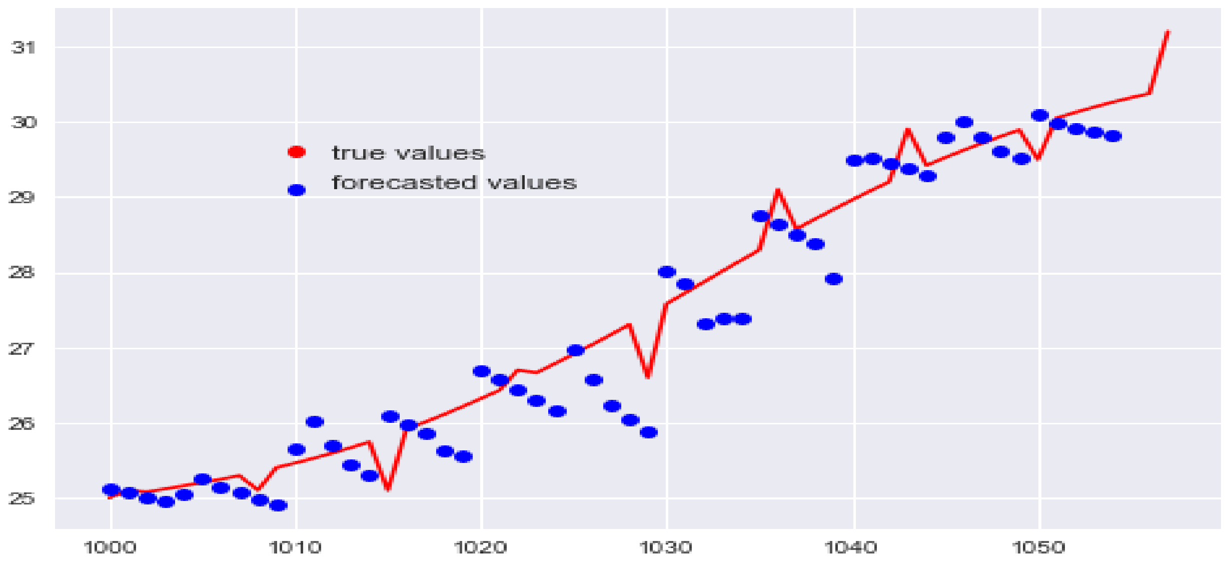


Figure 37. 5 forecasted values at regular intervals over last 100 test data

Long range ARIMA forecast:

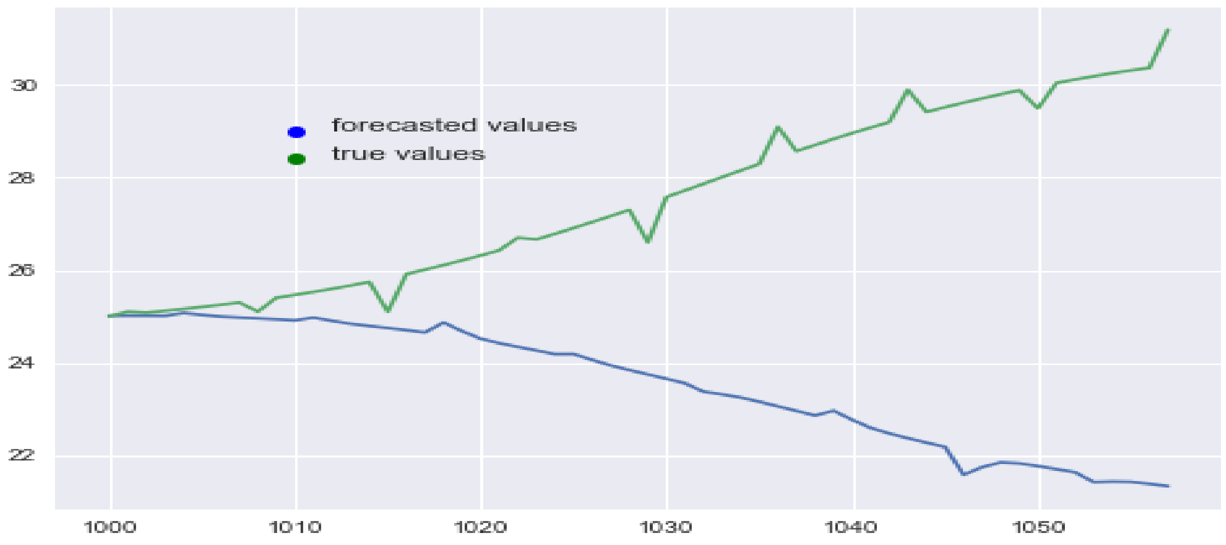


Figure 38 100 values forecasted over last 100 points of TAC test data

As we can see form Figure 38, in most cases ARIMA model completely fails on very long forecasting.

Errors

Forecast	MAE-score	R2-score
Short (next 2 values)	0.334	0.94
Medium (next 5 values)	0.64	0.81
Long (next 100 values)	3.99	-6.35

Table 8. Performance of ARIMA model on TAC dataset.

Summary of ARIMA models:

Although ARIMA models are good in short term forecasting they fail significantly for long term. ARIMA models cannot capture seasonality, trend etc. by default, they need special treatment to convert the data into stationary

In most real cases time series data does not only depend on its past values, various factors influence time series data. Certain events effect the time series data significantly. Effect of these events can be modeled by using a exogenous variable which is also known as ARIMAX. In the next section we try to include various factors influencing pm2.5 data in the model to obtain a better forecast

4.4 Vector Autoregression (VAR) based Time Series Forecasting

4.4.1 Introduction

A VAR model is an extension of AR, where we use Vectors instead of univariate variables. The advantage of VAR model over AR model is that it allows multiple factors, which affect the time series, to be involved while modeling [73].

4.4.2 Theoretical Background:

There are two types of VAR models

Structural VAR:

A structural VAR is of the form:

$$B_0 y_t = c_0 + B_1 y_{t-1} + B_2 y_{t-2} + \cdots .. + B_l y_{t-l} + \epsilon_t$$

Here c_0 and ϵ_t are vectors of size k and $B_{1....p}$ are square matrices with shape (k, k), where k is the number of elements in each vector.

Sample equation with vector size 2 and a lag 1:

$$\begin{bmatrix} 1 & B_{0;1,2} \\ B_{0;2,1} & 1 \end{bmatrix} \begin{bmatrix} y_{1,t} \\ y_{2,t} \end{bmatrix} = \begin{bmatrix} c_{0;1} \\ c_{0;2} \end{bmatrix} + \begin{bmatrix} B_{1;1,1} & B_{1;1,2} \\ B_{1;2,1} & B_{1;2,2} \end{bmatrix} \begin{bmatrix} y_{1,t-1} \\ y_{2,t-1} \end{bmatrix} + \begin{bmatrix} e_{1,t} \\ e_{2,t} \end{bmatrix}$$

$$y_{1,t} = c_{0,1} - B_{0;1,2} y_{2,t} + B_{1;1,1} y_{1,t-1} + B_{1;1,2} y_{2,t-1} + \epsilon_{1,t}$$

In a structural VAR the value of an element in vector y_t at a time t is also affected by other values at the same time [74]. However, this type of setup is not suitable for the time series forecasting as in many cases we do not know any values of the elements in a vector that we need to forecast. To avoid this problem, we use Reduced-form of VAR.

Reduced VAR:

$$y_t = B_0^{-1} c_0 + B_0^{-1} B_1 y_{t-1} + B_0^{-1} B_2 y_{t-2} + \cdots .. + B_0^{-1} B_p y_{t-l} + \epsilon_t$$

This can be rewritten as:

$$y_t = c + A_1 y_{t-1} + A_2 y_{t-2} + \cdots .. + A_l y_{t-l} + \epsilon_t$$

The advantage of the Reduced VAR form over the structural form is that in reduced form all the variables are dependent only on the previous values [75].

Properties of the error [76]:

1.) The errors are not correlated across the time.

2.) The mean of the error terms is zero.

4.4.3 Methodology and implementation

Application of Vector Auto Regression model on pm2.5 dataset.

As the number of factors are increased the complexity of VAR model increases. So only the factors which have high correlation with pm2.5 data are used

Here we are going to use variables pm2.5, Dew point, temperature and precipitation as vectors for VAR. unless there is co integration all time series in Vector Auto Regression model should be stationary.

Plots of pm2.5, dew point, temperature and precipitation.

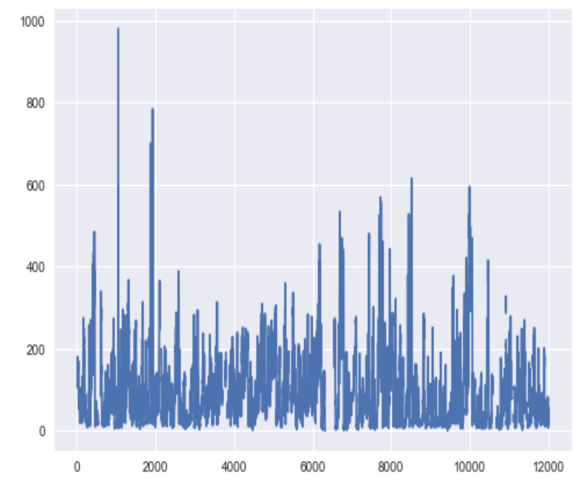


Figure 37. plot of pm2.5 training data

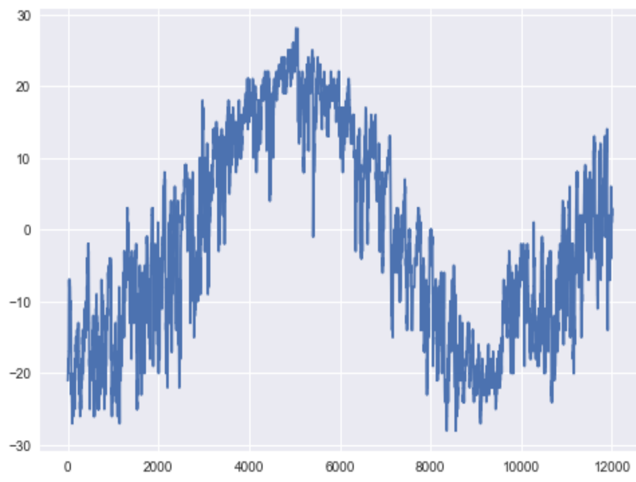


Figure 38. plot of dew point training data

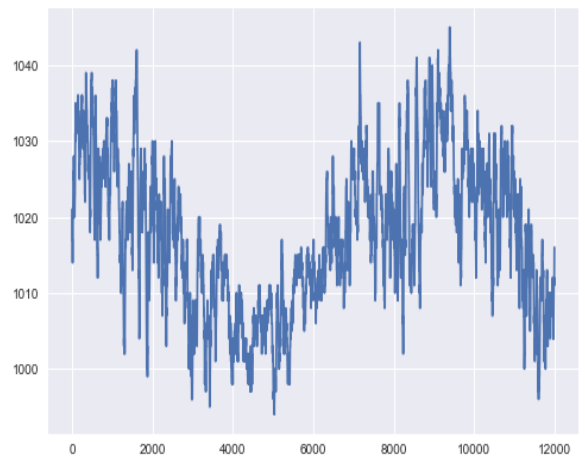


Figure 39. plot of temperature training data

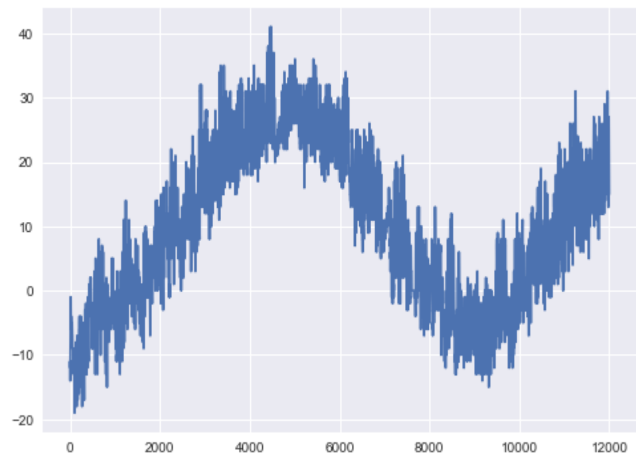


Figure 40. plot of precipitation training data

As all the series are non-stationary, we need to stationarize them by differencing.

Plots of Differentiated series (pm2.5, dew point, temperature and precipitation.)

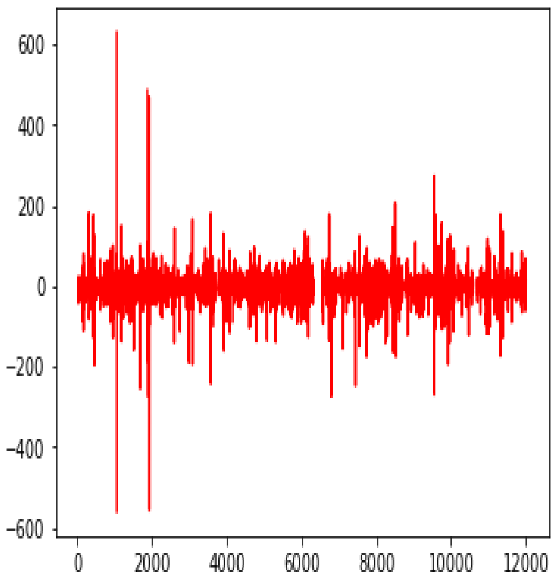


Figure 41. plot of pm2.5 differenced data

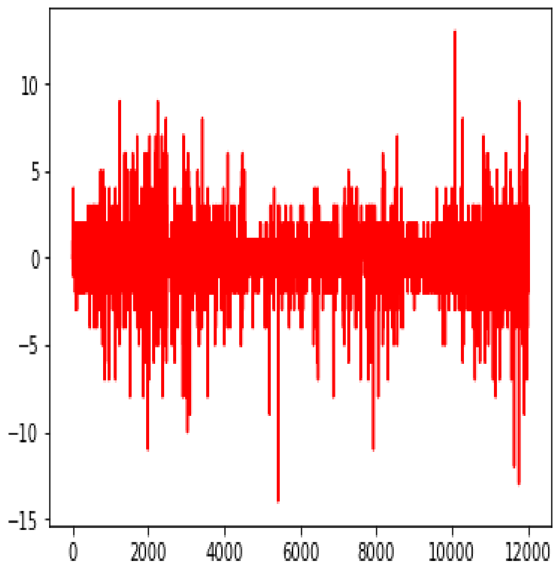


Figure 42. plot of dew point differenced data

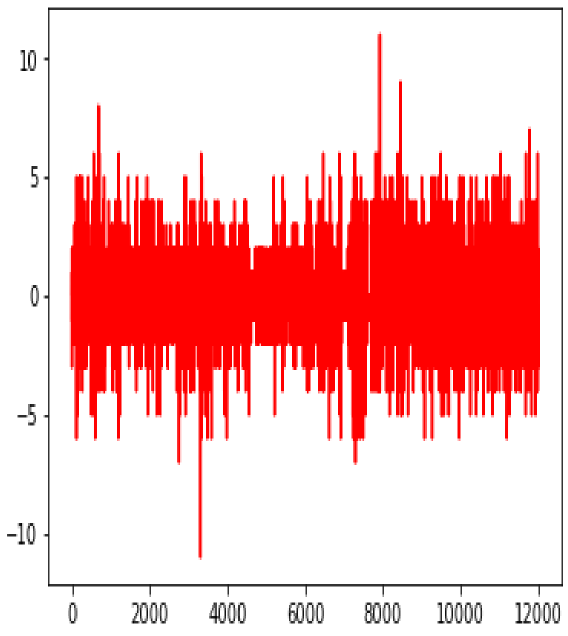


Figure 43. plot of temperature differenced data

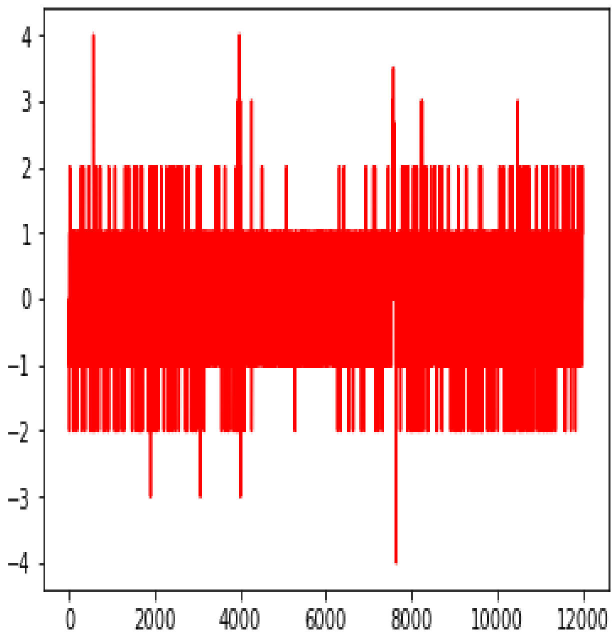


Figure 44. plot of precipitation differenced data

Selecting number of lags in Vector Auto Regression (VAR):

Firstly, we fit the model with large number of lags, which results in a good fit and then we perform a likelihood ratio test with multiple small lags, and use the shortest lag, which satisfies the likelihood ratio test [77]. Secondly, other information such as Akaike information criterion (AIC) and Bayesian information criterion (BIC) are also used in selecting the number of lags.

Likelihood ratio:

$$\Lambda(x) = \frac{L(\theta_1 \vee x)}{\{L(\theta|x): \theta \epsilon \{\theta_1, \theta_2\}\}}$$

Wilks theorem:

Wilks theorem is an extension of likelihood ratio test, which helps using chi-square distribution to accept or reject null hypothesis. According to Wilks theorem [78], the test statistic -2log (Λ follows chi-square distribution with degrees of freedom equal to $\theta_1 - \theta_2 \vee$

According to the above procedure, lag of 4 fits best for pm2.5 dataset.

Model Fit

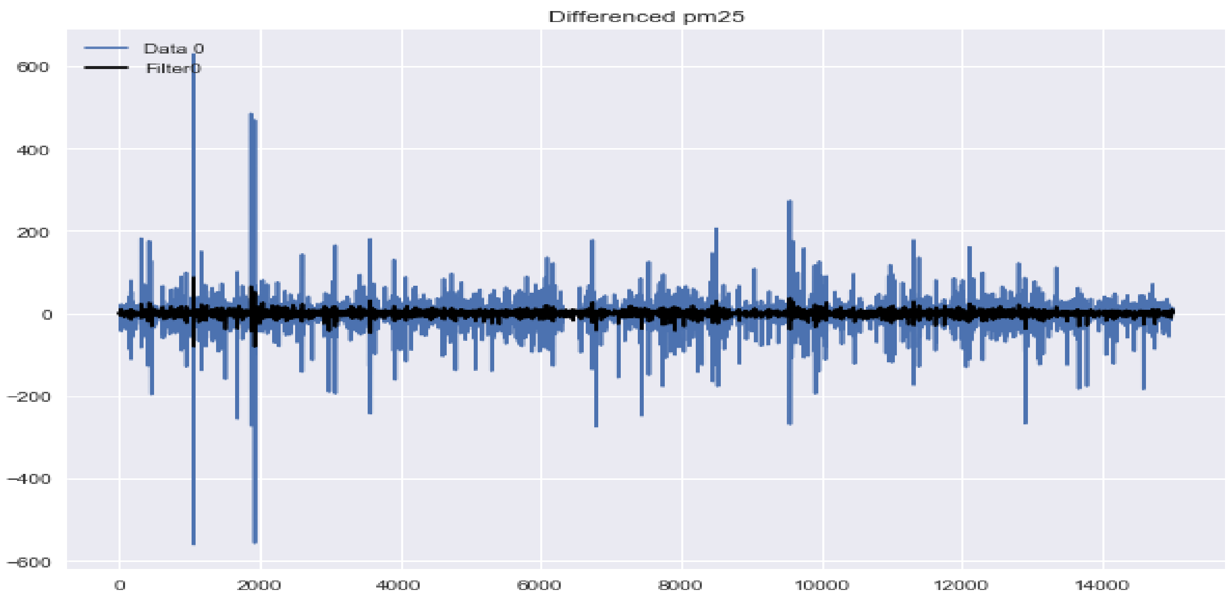


Figure 45. VAR model fit on differenced pm2.5 data

4.4.4 Results

Short term forecast:

Fig 46 shows that the forecasted values are more accurate using VAR model than ARIMA so we can conclude that the factors have significant effect on pm2.5

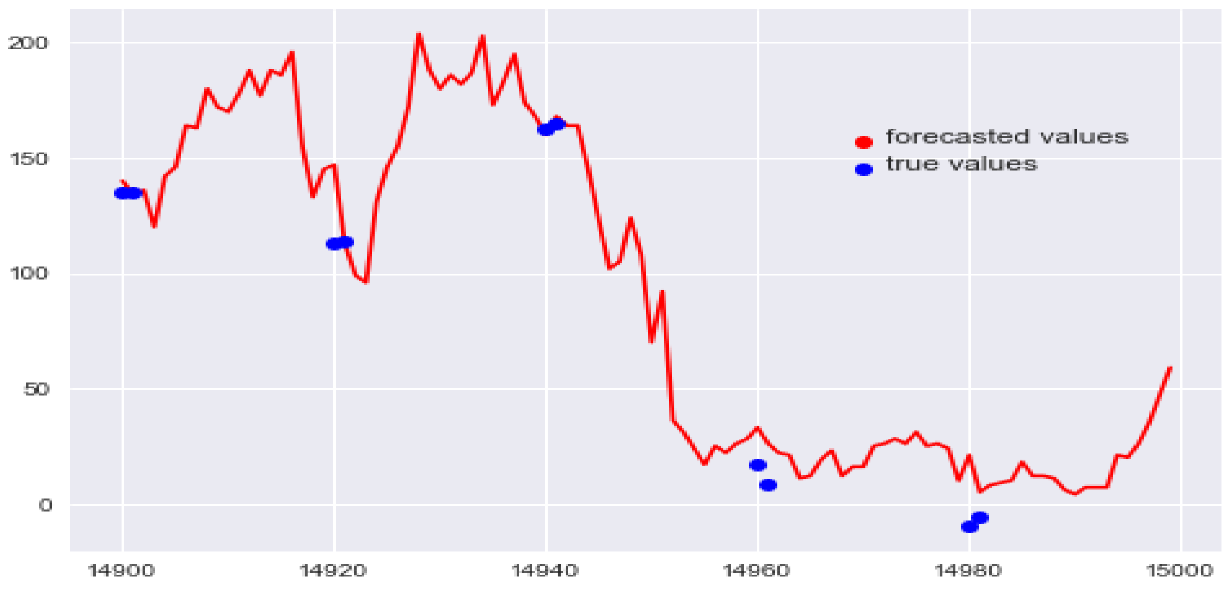


Figure 46. 2 pm2.5 values forecasted at regular interval on last 100 points in test data using VAR model

Midterm forecast:

VAR model tend to give better forecast when compared with ARIMA as the forecast range increases. Because in VAR model the forecasting is also monitored by other variable instead of depending only on univariate variable

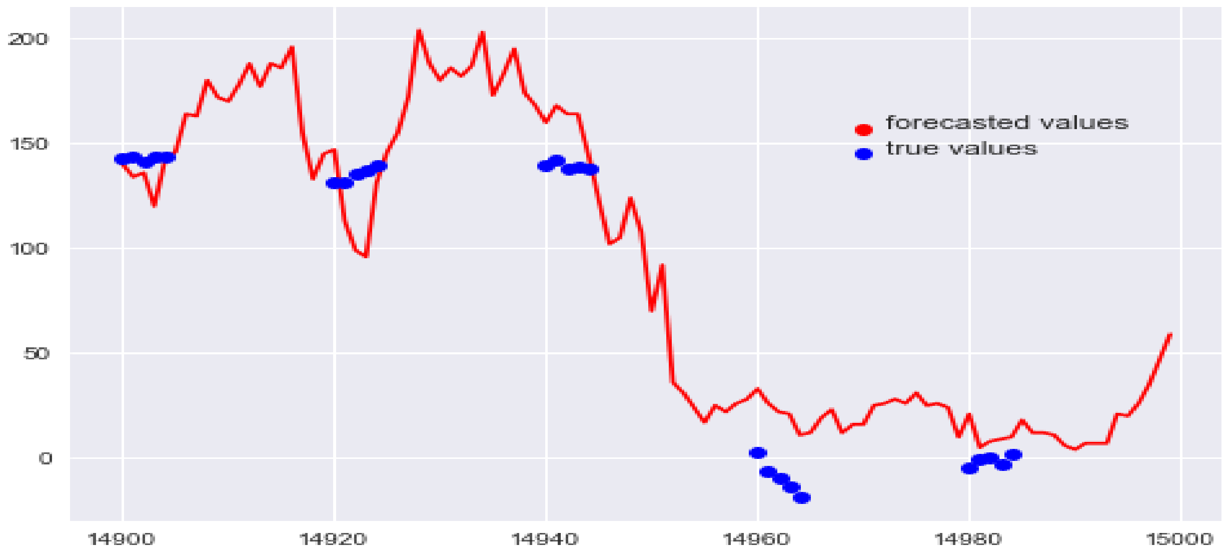


Figure 47. 5 pm2.5 values forecasted at regular intervals on last 100 points in test data using VAR model

Long term forecast

Fig 48 shows that for very long range VAR models also fail significantly.

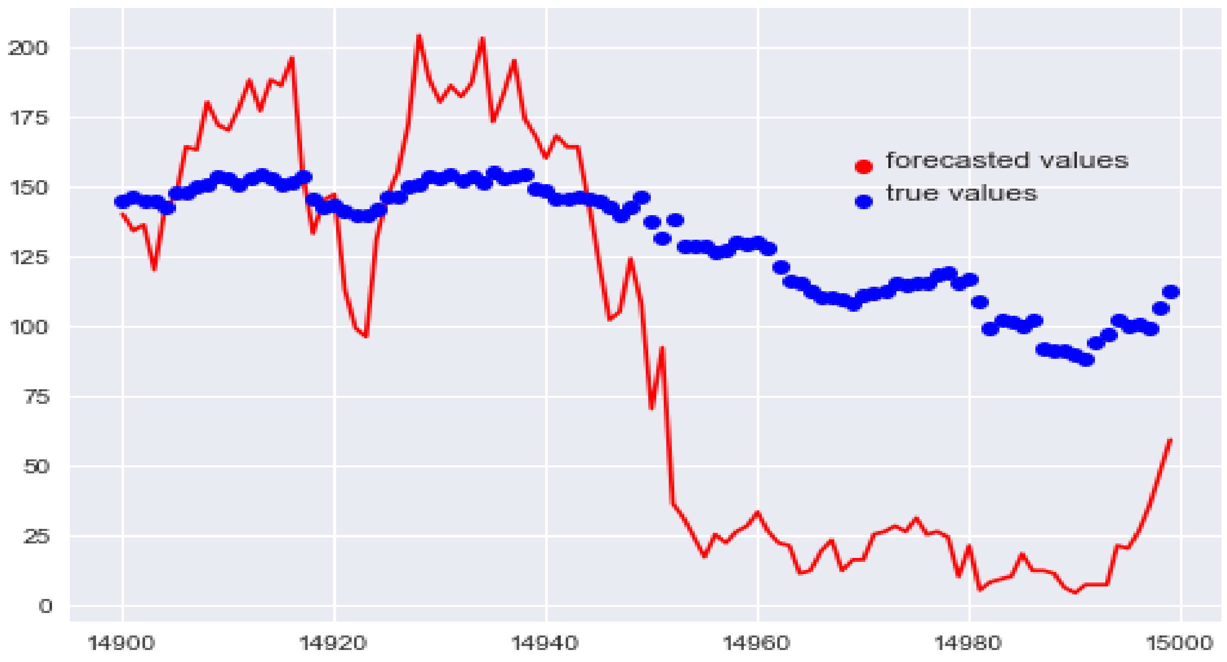


Figure 48. 100 pm2.5 values forecasted on last 100 data points using VAR model.

Forecast	MAE-score	R2-score
Short (next 2 values)	2.83	0.97
Medium (next 5 values)	3.117	0.945
Long (next 100 values)	56.1	0.11

Table 9. Performance of VAR model on Beijing dataset.

Summary of VAR models

Although VAR models are better than ARIMA in most cases. In real world it is difficult to find all the factors which effect the time series so we use other techniques like kalman filter to model the unknown effects.

Like traditional ARIMA, VAR models also cannot model multiple type of variances in time series data and they are poor in handling unexpected variances. For the pm2.5 data certain festive events like New Year etc. have significant effect on its value and they have to be treated separately .So generally VAR models are used in combination with more advanced techniques

4.5 Random walk based Time Series Forecasting

4.5.1 Introduction

Random walk model is the simplest, yet the most important models. This model is in a wide usage in a finance area, as the stock prices or exchange rates are assumed to follow a random walk [51]. A random walk, in general, means that today's stock price can be derived from yesterday's adding a random shock value. A random walk is usually applied to a non-stationary time series.

4.5.2 Theoretical Background

The idea behind a random walk model is that in each period the variable takes a random step from its preceding value, where steps should follow independent and identical distribution.

A random walk model for the time series can be described as:

$$X_t = X_{t-1} + w_t,$$

where X_t – value at time *t*, X_{t-1} – value at time *t-1*, w_t – a random shock value at time *t* (an error term). Here the term w_t can be considered as a Gaussian white noise term, which is distributed from standard normal distribution of N (0, σ^2). The time series itself is not random, however in the model is used the first difference of the time series, which is random [51]. Thus, the model can be seen as:

$$X_t - X_{t-1} = w_t,$$

where the original time series is transformed to the first order difference time series.

There are two different types of random walk [51]:

1) Random walk without a drift term

$$X_t = X_{t-1} + w_t$$

2) Random walk with a drift

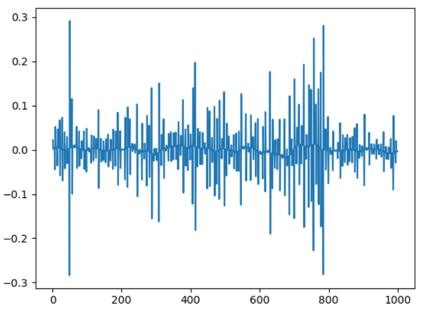
$$X_t = X_{t-1} + \mu_t + w_t$$

A drift is a constant, which is an average of period-to-period change observed in the past [52].

4.5.3 Methodology and implementation

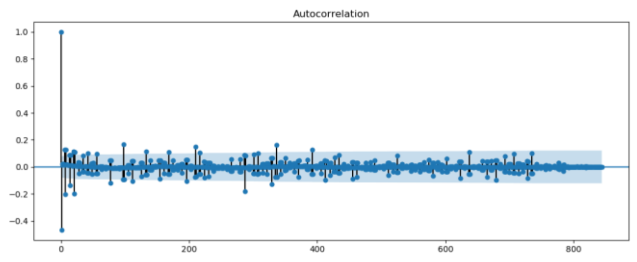
A Random walk model is this project has been applied to a TAC index dataset only, as it is a non-stationary and univariate data. At first, a lag operator (backward shift) is needed to represent X_{t-1} values, if the time series is assumed to be X_t. A lag operator can return values as one-time unit preceding. Thus, the simple model can be shown as: $X_t = X_{t-1}$.

To obtain $\boldsymbol{w_t}$, firstly we need to take the first order difference between values $\boldsymbol{X_t}$ and $\boldsymbol{X_{t-1}}$. The plot of the first order difference is shown:



The mean of the differenced values is: 0.000364, and the standard deviation is: 0.0418. The mean is not significantly different from zero (described below by t-test statistics), and by assumption, the normal distribution, giving values, for $\boldsymbol{w_t}$ is obtained. Thus, random shock values are generated for each step.

To answer the question is there a drift term or not, it is useful to look at the mean of the daily changes. For this purpose, it is better to perform t-test statistics. The standard error of the mean is a division of the standard deviation by the square root of the sample size, which is 0.0418 / (999^2) = 0.00132. Then the ratio of the mean to its own standard error, which is 0.000364 / 0.00132 = 0.275, is obtained. Thus, this value 0.275 is not greater than the t-table value = 1.65 (5% level), hence a null hypothesis that $\mu_1 = \mu_2$, where $\mu_1 = 0$ is not rejected, meaning that there is no drift. This also can be checked by looking at the autocorrelation plot, which is achieved by importing the package tsaplots (plot_acf) in python from statsmodels.graphics:



The plot clearly shows that there are not significant autocorrelations, and the data is random. Moreover, except lag 0, which is always 1 by default, all the autocorrelations are within the confidence interval of 95%. In addition, there is no clear pattern, which lead to randomness. Thus, having summarized, the TAC index dataset can almost perfectly fit a random walk model without drift.

4.5.4 Results and discussions

After fitting the Random walk to the train dataset, it can be seen, that the model can fit the data and follows its shape, except some spikes:

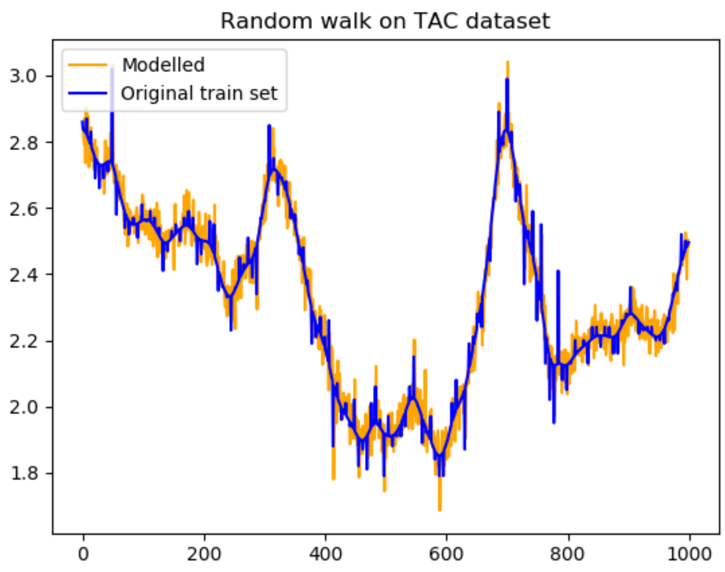


Figure 49. Model fitting of Random walk on TAC dataset.

In order to obtain prediction for t+1, the model is given as: $\boldsymbol{X_{t+1} = X_t + w_{t+1}}$

To sample the term w_{t+1} from Gaussian Normal distribution, giving the standard deviation from the train set (1000 points), (np.random.seed is set to 1), then prediction scores for the time t+1 for the test set of (58 points) is derived as:

Note: here just for the comparison the model has been fitted and later predicted on the original non-stationary data and on transformed data, using logarithmic transformation. However, the Random walk model is usually performed on non-stationary original data. In following discussions, the value of the original data will be used as referring to the results of the Random walk model.

Data	MAE	R2_score
Original dataset	0.042	0.927
Transformed dataset	0.012	0.915

Table 10. Performance of Random walk model on TAC dataset.

According to the table, the scores show MAE of 0.042 and R_2 score of 0.927, compared to the ground truth values.

Below the graph shows an original test set and generated predicted values for t+1 time. Despite that the predicted curve is not following perfectly an original graph, it catches the trend and might still closely predict the direction at t+1. Moreover, R_2 score of 0.927 is close to 1, and MAE is relatively small. However, the predicted curve has more oscillations, probably because of the random shock values, generated randomly from a Gaussian distribution.

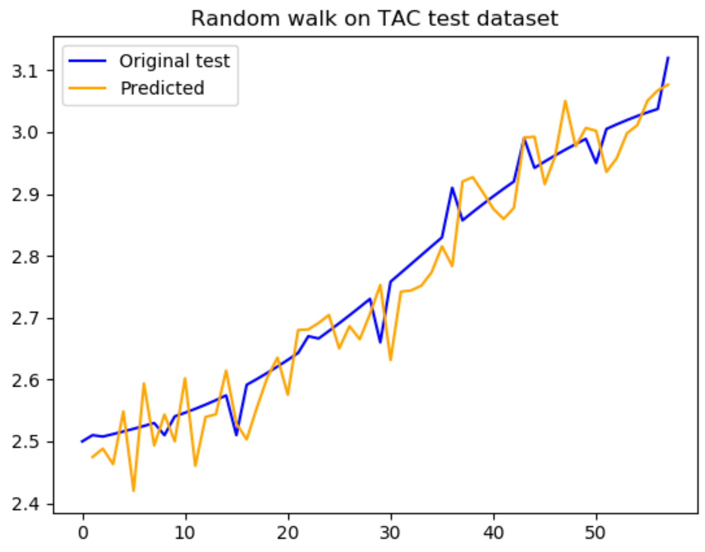


Figure 50. Performance of Random walk on TAC test set.

Forecasting part

The plot below shows a derivation of forecasting of Random walk model for 3 future points in a gap of 10 days. For this purpose, the model has been trained iteratively for the next future points. As it can be seen, forecasted values do not actually follow the shape of original values, which makes a sense to predict only for the near future of t+1. However, it can still provide some spikes at point 40+3 and 50+3, where the direction is at some degree looks closely. To evaluate it more precisely, a table of MAE and R2_score values is given below.

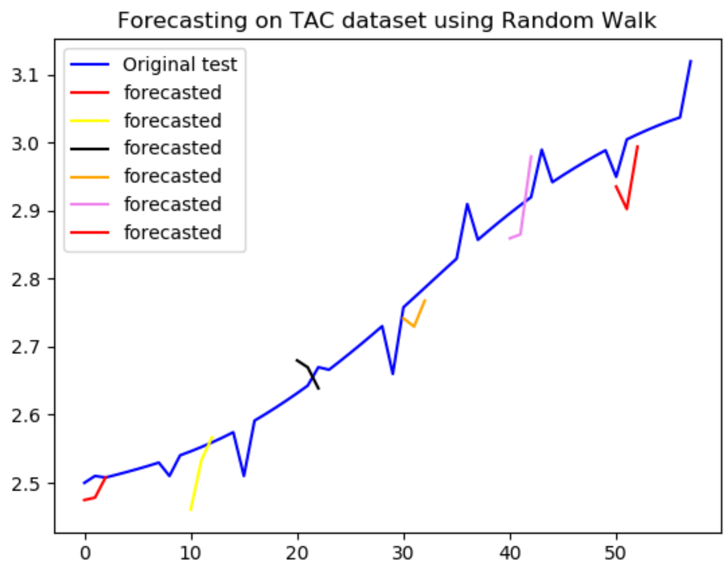


Figure 51. Performance of Random walk multistep forecasting on TAC test set.

The table shows evaluation of ground truth values with forecasted values of t+3 future points. All the data has a negative R2_score, meaning that the shape of line does not properly fit the model. More negative value means more different in shape from the actual. At the point t = 10, t+3, the value of R2_score is the most negative, and according to the graph, it can be seen that the shape is totally far away from the original. The smallest negative r2_score value is on the point t = 50, t+3, meaning that at some degree the shape of the model is following, however, it is still negative. The first point t = 0, t+3 has the smallest MAE of 0.019, meaning that 3 consecutive points of forecasted and truth values are relatively significantly not different. However, the rest points have bigger MAE value. Average MAE is 0.0348.

Forecasted period	MAE on TAC data	R2_score on TAC data
t = 0, t+3	0.019	-28.251
t = 10, t+3	0.038	-89.982
t = 20, t+3	0.035	-4.185
t = 30, t+3	0.026	-4.942
t = 40, t+3	0.046	-22.671
t = 50, t+3	0.045	-3.741

Table 11. Performance of multistep forecasting of Random walk model on TAC dataset.

Chapter 5. Conclusion

5.1 Model comparison

TAC index data

Note: here for comparison only, SVR on the original dataset is included too, however, as a preprocessing step it is usually required to transform the data before applying SVR model.

Prediction (t+1)		
Model	MAE	R2_score
Stacked LSTM	0.0305	0.955
SVR (original data)	0.0240	0.967
SVR (data preprocessing-transform)	0.0066	0.967
ARIMA	0.334	0.94
Random walk	0.042	0.927

Table 12. Performance of prediction between different models on TAC dataset.

The table above (12) presents prediction on t+1 between different models used in this research. As it can be seen, results on SVR model has the smallest MAE value (either using original data or transformed as a preprocessing and required step) and the largest R2_score, being closer to 1. This means that SVR model best suits this data and, thus, gives better performance. Stacked LSTM has performed better than ARIMA, but worse than a Random walk model. It might be the case that the given data follows a Random walk (as assumed for the financial time series [51]). However, R2_score of stacked LSTM is better the Random walk's model (and also better than ARIMA). Probably, it might be the case of ability of LSTM better fitting a non-linear model.

Nevertheless, that ARIMA is considered as a classical model for prediction time series data, it has the largest MAE value. Moreover, Random walk model, which is a considerably simpler model than ARIMA, has performed better considering MAE value.

Mutistep forward forecast (t+3)	
Model	MAE
Stacked LSTM without optimization	0.04
Stacked LSTM with optimization	0.025
SVR (original data)	0.052
SVR (data preprocessing-transform)	0.014
ARIMA	0.64
Random walk	0.0348

Table 13. Performance of multistep forecasting between different models on TAC dataset.

The table above (13) shows performance on multistep ahead forecasting (t+3) on TAC test set considering different models. The smallest MAE value has SVR model (on transformed data), meaning that SVR can forecast for 3 future points with smaller error compared to other models. To mention, SVR on original data (without preprocessing-transformation, which is not advised to do, as the data must be preprocessed-transformed) performs worse, however even better than a classical for the time series prediction ARIMA model. The next good performance shows stacked LSTM model, which has smaller MAE values compared to ARIMA and Random walk models. It is recommended to use stacked LSTM with optimization compared to without optimization model for the future consecutive points forecasting for a better performance.

Beijing PM2.5 pollution data

Prediction (t+1)		
Model	MAE	R2_score
Stacked LSTM	1.92	0.996
Seq2seq LSTM	37.86	0.541
SVR	0.127	0.922
ARIMA	12.89	0.91
VAR	2.83	0.97

Table 14. Performance of prediction between different models on Beijing pm2.5 pollution dataset.

The table above (14) indicates the performance of the time series prediction of different models studied in this research and fitted on Beijing PM2.5 pollution dataset. The best performance shows SVR model with the smallest MAE value of 0.127. However, the largest R2_score close to 1 has stacked LSTM model among all models, better fitting nonlinear case. It is recommended to use stacked LSTM model compared to seq2seq LSTM. VAR model outperforms ARIMA, having significantly smaller MAE value. The reason for it is that VAR uses features (factors) of the data.

Mutistep forward forecast (t+3)	
Model	MAE
Stacked LSTM without optimization	9.436
Stacked LSTM with optimization	20.58
Seq2seq LSTM without optimization	46.33
Seq2seq LSTM with optimization	45.33
SVR	0.184
ARIMA	25.71
VAR	3.117

Table 15. Performance of multistep forecasting between different models on Beijing pm2.5 pollution dataset.

The table above (15) represents the performance of multistep forecasting between different models applied to Beijing pm2.5 pollution dataset. It can be seen, that the best performance show SVR model with the lowest MAE value. The next successful performance shows VAR model. Stacked LSTM model performs better than seq2seq LSTM. ARIMA model, which use only one feature performs better than seq2seq LSTM.

Thus, to sum up, SVR is the best performed on the two different time series data (univariate and multivariate) for t+1 prediction and t+3 forecasting.

5.2 Recommendation and future work

This section below provides several possible recommendations and future work according to the different models used in this research.

5.2.1 LSTM model

Theoretically, the least squares solution line does a terrible job of modeling non-linearity. Other methods such as Kernel Ridge Regression can be used to eliminate these flaws to model the non-linearity much better and overall better results.

Choosing different dimension for the hidden state in the LSTM can also change the fate of results. However this is a hyper parameter which needs tuning.

5.2.2 SVR model

The prediction results for SVR model for both datasets have been successful with a relatively small MAE, and R2_score close to 1. However, this prediction is at point t+1, for multistep forecasting values are getting worse. To overcome this the possible future work is suggested to have a hybrid models using SVR with other different models such as EMD-SVR, which might obtain better results. Another suggestion might be to use a hybrid SVR as linear SVR as a pre-processor and non-linear SVR as a following step. However, these models are out of the scope of this project work. One limitation of this model is that it requires a considerable amount of computational resources and time (almost more than 15 hours on train set of 35K for 10-fold cross validation), when running on the Beijing Pollution dataset. Thus, this dataset has been reduced to 15000 points x 8 features (12000 train and 3000 test), which only represents 34% of the data. However, this also required a running time of approximately 2 hours with 10 cross-validation, thus a 3-fold cross-validation is used, when checking some parameters. Possibly applying another technique of dimensionality reduction before using SVR might be worth.

5.2.3 ARIMA model

<u>ARIMA extension with GARCH (ARIMA-GARCH)</u>

In the basic ARIMA model we assume that the random innovations are independent and they all come from the same distribution (homoscedastic). They are called white noise, error term, and innovation etc, based on the use case.

However, some real-world problems we observe is that some periods change rapidly, and some periods change very slowly. Thus, if we know the periods, we use different distributions according to the time to model the time series, which is known as a conditional Heteroscedasticity (conditional on time). Also, in some cases we might not even know the periods when the volatility happens. Therefore, we try to model this unexpected variance using Auto regression model [79].

ARCH (p)

We model different time dependent standard deviations of known periods using errors [80]:

$$\sigma_t^2 = \hat{\alpha}_0 + \alpha_1 \epsilon_{t-1}^2 + .. + \alpha_q \epsilon_{t-p}^2$$

A GARCH model is an extension of ARCH model, where we also model unexpected volatility along with volatility for which the periods are known, by using variances of previous lags. GARCH model is represented by GARCH (p,q), where p is the number of error terms used to predict the standard deviation for a time period of the time series, and q is the lag with which we try to model unexpected volatility using Auto regression:

$$\sigma_t^2 = w + \alpha_1 \epsilon_{t-1}^2 + .. + \alpha_q \epsilon_{t-p}^2 + \beta_1 \sigma_{t-1}^2 + .. + \beta_q \sigma_{t-p}^2$$

As we can see in the pm2.5 data set, there are periods with unexpected rise and some known periods like festive events, which also have their effect on pm2.5. Therefore, we can try to model these errors using GARCH model after fitting ARIMA.

5.2.4 VAR model

It might be possible to use VAR model with a combination of Kalman filter.

Kalman filter with Vector Auto Regression model (VAR).

Kalman filters are used in cases of continuous changes with uncertain information about some changes in the system. It uses correlation between variables to give us a better forecast [81].

In general, some mathematical equations for predicting is used:

$$\hat{x}_k = F_k \hat{x}_{k-1}$$

$$P_k = F_k P_{k-1} F^T{}_k$$

Where $\hat{x}_k$ is vector of all changing variables, F_k are the coefficients and P_k is the correlation matrix between variables at time t.

However, due to the various external unknown factors the predicted values ($\hat{x}_k$ differ from real ones. Kalman filters try to use the real values of the data (training data) to modify the model and give a better approximation. According to the Kalman filter, the variables, which are varying, come from multivariate Gaussian distribution, i.e. some combinations of them more likely than the others. As the system moves, the covariance’s between variables change and we will have a new Gaussian distribution [82].

Using information from real values

But in reality, we observe different values (also multivariate Gaussian), thus we assume that these values can be modelled with matrix H (the observed values are linear combination of predicted values)

$$\hat{\mu}_{expected} = H_k \hat{x}_k$$

$$\in_{expected} = H_k P_k H^T{}_k$$

Where $\hat{x}_k$ is vector predicted by naive model and we expect that the mean of observed values ($\hat{\mu}_{expected}$ is a combination of values predicted by model (vector of internal state).

Let z_k represents the real values observed (which is also a multivariate Gaussian distribution), for which the covariance matrix is represented byR_k.

Now we have two Gaussians, which we multiply to obtain a clever prediction from the both information available.

After some mathematical operations and taking advantage of covariances from real observed values and predicted values, the forecasting equations become:

$$\hat{x}'_k = \hat{x}_k + K'(z_k - H_k\hat{x}_k)$$
$$P'_k = P_k - K'H_kP_k$$
$$K' = P_kH_k^t\left(H_kP_kH_k^T + R_k\right)^{-1}$$

In the time series case we say that x are the values to be forecasted, but as we do not know all the factors, which influence vectors, we consider it as an internal state and try to model the effect of unknown factors by using a Kalman filter to get a better forecast.

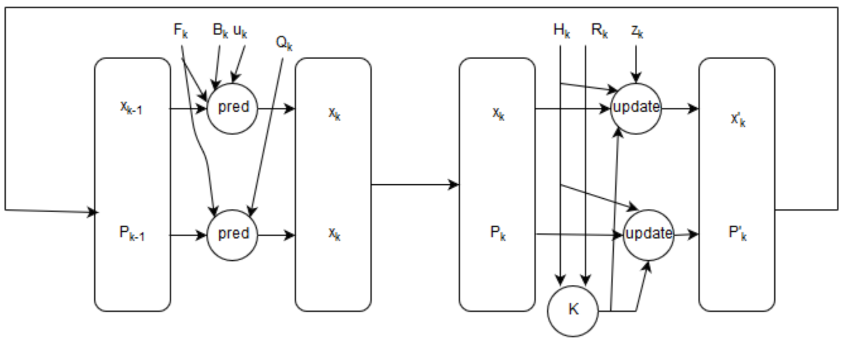


Figure 52. Kalman filter.

5.2.4 Random walk model

The random walk model is considered as a benchmark of the financial time series prediction, and it is assumed that financial trends follow a random walk. However, in this research study a machine learning technique SVR has performed better on the dataset. Thus, it is better to consider also other techniques such as SVR or more complicated than the Random walk such as ARIMA models. However, in this work the Random walk model has been applied only to one dataset (because of its appropriate features). Probably, using the Random walk model on various datasets there would be different results. Hence, a recommendation for the future work would be applying the Random walk to different time series data. In addition, a Random walk can be considered as a good starting point for a model, but always it is better to compare also with other more complicated models.

References

[1] Inc.com (n.d.) Forecasting. <https://www.inc.com/encyclopedia/forecasting.html>

[2] Chatfield (2000) *Time Series Forecasting.* Boca Raton London New York Washington, D.C: Chapman & Hall/CRC

[3] UCI Machine Learning Repository (2017) Beijing PM2.5 Data. <https://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data>

[4] Department of Health. (n.d.) Fine Particles (PM 2.5) Questions and Answers. <https://www.health.ny.gov/environmental/indoors/air/pmq_a.htm>

[5] ScipyOrg (n.d.) Interpolation <https://docs.scipy.org/doc/scipy/reference/tutorial/interpolate.html>

[6] Senin (2016) Z-normalisation of time series SAX-VSM https://jmotif.github.io/sax-vsm_site/morea/algorithm/znorm.html

[7] Cao, L.J., Tay, F.E.H., (2003). Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on Neural Networks,* Vol. 14, No. 6: 1506 – 1518

[8] Ogasawara E., Martinez L., Oliveira D., Zimbrão G., Pappa, G., Mattoso, M. () Adaptive Normalization: A Novel Data Normalization Approach for Non-Stationary Time Series. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.718.9985&rep=rep1&type=pdf>

[9] Lütkepohl, H., Fang, X. (2010) The role of the log transformation in forecasting economic variables. Springer-Verlag, 619-638. DOI 10.1007/s00181-010-0440-1 <https://link.springer.com/article/10.1007/s00181-010-0440-1>

[10] Abecasis, S.M., Lapenta, E.S. and Pedreira, C.E. (1999). Performance metrics for financial time series forecasting. *Journal of computational intelligence in finance,* 7 (4), 5 - 23

[11] Nogales F.J., Contreras J., Conejo A.J., and Espínola R., (2002). Forecasting next-day electricity prices by time series models. *IEEE Transactions on Power Systems,* 17 (2) 342 - 348

[12] Nelson, H.L. Jr., and Granger, C.W.J. (1979) Experience with Using the BoxCox Transformation When Forecasting Economic Time Series. *Journal of Econometrics,* 10, 57 - 69

[13] Raschka, S. (n.d.) MinMax Scaling. MLxtend

<https://rasbt.github.io/mlxtend/user_guide/preprocessing/minmax_scaling/>

[14] Ostashchuk, O. (2017) Time Series Data Prediction and Analysis. Master's Thesis, Czech Technical University Digital Library <https://dspace.cvut.cz/bitstream/handle/10467/70524/F3-DP-2017-Ostashchuk-Oleg-Prediction%20Time%20Series%20Data%20Analysis.pdf?sequence=-1&isAllowed=y>

[15] Bukola, O. (2006) Support Vector Regression for Non-Stationary Time Series. Master's Thesis, University of Tennessee <http://trace.tennessee.edu/cgi/viewcontent.cgi?article=3107&context=utk_gradthes>

[16] Thomason, M., (1999). The practitioner methods, *Journal of Computational Intelligence in Finance,* 7 (3) 36 – 45

[17] Scikit-learn (n.d.) sklearn.model_selection.GridSearchCV <http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>

[18] Guo, Y., Zheng, Y., Wang, X. (2014) Improved multi-kernel LS-SVR for time series online prediction with incremental learning. *IEEE: Prognostics and Health Management (PHM)* DOI: 10.1109/ICPHM.2014.7036376

[19] Smola, A., Scholkopf B. (2003) A tutorial on support vector regression. RSISE, Australian National University, Canberra 0200, Australia <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=97B56DF4E17609D55AF2443BA9DEE9FA?doi=10.1.1.114.4288&rep=rep1&type=pdf>

[20] Ruping, S. (2001) SVM Kernels for Time Series Analysis <http://www-ai.cs.uni-dortmund.de/EVENTS/FGML2001/FGML2001-Paper-Rueping.pdf>

[21] Samsudin, R. ,Shabri A. and Saad, P. (2010) A Comparison of Time Series Forecasting using Support Vector Machine and Artificial Neural Network Model. *Journal of Applied Sciences, 10: 950-958* <https://scialert.net/fulltext/?doi=jas.2010.950.958>

[22] Ding, Y., Song X. and Zen, Y. (2008) Forecasting financial condition of Chinese listed companies based on support vector machine. *Expert Syst. Appl.*, 34: 3081-3089

[23] Eslamian, S.S., Gohari, S.A., Biabanaki M. and Malekian, R. (2008) Estimation of monthly pan evaporation using artificial neural networks and support vector machines. *Applied Science*, 8: 3497-3502

[24] Wang, W.C., Chau, K.W., Chen C.T., Qiu, L. (2009) A comparison of performance of several artificial intelligence methods for forecasting monthly discharge time series. *Hydrol*, 374: 294-306.

[25] Cherkassky, V., and Ma, Y., (2004). Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks,* 17 (1) 113 – 126

[26] Scholkopf, B., Burges, C., and Smola, A., (1999). Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, Massachusetts

[27] Kwok, J.T., and Tsang I.W., (2003) Linear dependency between ε and the input noise in ε -support vector regression. *IEEE Transactions on Neural Networks,* 1 – 8

[28] Willmott, C., Matsuura, K. (2005) Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, 30, 79–82. DOI:10.3354/cr030079

[29] Canova, F. (2007) VAR Models. Methods for Applied Macroeconomic Research Princeton University Press, <http://apps.eui.eu/Personal/Canova/Articles/ch4.pdf>

[30] Fan, J. and Yao, Q. (2003) Nonlinear Time Series: Nonparametric and Parametric Methods, Springer-Verlag, New York, NY.

[31] Weron, R. and Misiorek, A. (2008) Forecasting spot electricity prices: a comparison of parametric and semiparametric time series models. International Journal of Forecasting, 24 (4), 744–763

[32] Findley, D.F., Monsell, B.C., Otto, M.C., Bell, W.R., Pugh M. (1992) Towards X-12 ARIMA. Technical report, Bureau of the Census

[33] Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. PMID 9377276.

[34] Adhikari, R., Agrawal R. K. (n.d.) An Introductory Study on Time Series Modeling and Forecasting https://arxiv.org/ftp/arxiv/papers/1302/1302.6613.pdf

[35] Zhang, G.P. (2003) Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing,* 50, 159–175

[36] Kihoro, J.M., Otieno, R.O., Wafula, C. (n.d.) Seasonal Time Series Forecasting: A Comparative Study of ARIMA and ANN Models. *African Journal of Science and Technology (AJST) Science and Engineering,* 5 (2), 41-49

[37] Kamruzzaman, J. Begg, R., Sarker, R. (2006) *Artificial Neural Networks in Finance and Manufacturing.* Idea Group Publishing, USA.

[38] Graves, A. (2013) Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850.

[39] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems.* 3104–3112

[40] Zaremba, W., Sutskever, I., and Vinyals, O. (2014) Recurrent neural network regularization. <arXiv preprint arXiv:1409.2329>

[41] Bengio, Y., Simard, P., Frasconi, P. (1994) Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions* 5(2), 157–166

[42] Bissoondeeal, R. K., Binner, J. M., Bhuruth, M., Gazely, A., Mootanah, V. P. (2008) Forecasting exchange rates with linear and nonlinear models. *Global Business and Economics Review* 10 (4) 414–429

[43] Ravisankar, P., Ravi, V., Raghava G., Bose, R. (2011) Detection of financial statement fraud and feature selection using data mining techniques. *Decision Support Systems* 50 (2) 491–500. DOI:10.1016/j.dss.2010.11.006

[44] Sun, J., Li, H. (2012) Financial distress prediction using support vector machines: Ensemble vs. individual. *Applied Soft Computing* 12 (8) 2254–2265. DOI:10.1016/j.asoc.2012.03.028.

[45] Gong, X., Si, Y.-W., Fong, S., Biuk-Aghai, R. P. (2016) Financial time series pattern matching with extended UCR Suite and Support Vector Machine. *Expert Systems with Applications,* 55 284–296. DOI:10.1016/j.eswa.2016.02.017.

[46] Malkiel, B. (1973) A random walk down Wall Street

[47] Kim, K. (2003) Financial time series forecasting using support vector machines, Neurocomputing 55 (1-2) 307–319. DOI: 10.1016/s0925-2312(03)00372-2.

[48] Kamruzzaman, J., Sarker, R. A., Ahmad, I. (2003) SVM based models for predicting foreign currency exchange rates. *Third IEEE International Conference on Data Mining,* ICDM, IEEE, 557–560

[49] D.-Z. Cao, S.-L. Pang, Y.-H. Bai, Forecasting exchange rate using support vector machines, in: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, Vol. 6, IEEE, 2005, pp. 3448–3452.

[50] Jacovides, A. (2008) Forecasting Interest Rates from the Term Structure: Support Vector Machines Vs Neural Networks. Master Thesis, University of Nottingham <http://citeseerx.ist.psu.edu/viewdoc/download> DOI=10.1.1.454.9948&rep=rep1&type=pdf

[51] Vapnik, V. (1998) Statistical Learning Theory, New York: Wiley

[52] Imdadullah, M. (2016) Random Walk Model. Time Series Analysis and Forecasting. *Basic Statistics and Data Analysis, Lecture notes.*

<http://itfeature.com/time-series-analysis-and-forecasting/random-walk-model>

[53] Thomason, M., Caldwell, R., (1998). The practitioner methods. *Journal of Computational Intelligence in Finance*, Vol. 6, No.1, 42 – 47

[54] Gers, F., Schmidhuber J., Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. *Neural Computation,* 12 (10): 2451–2471.

[55] Basheer, I.A. and Najmeer, M. (2000) Artificial neural networks: fundamentals, computing, design, and application, *Journal for Microbiolgical Methods,* 43 (1) DOI.org/10.1016/S0167-7012(00)00201-3

[56] Graves, et al. (2013) Speech Recognition with Deep Recurrent Neural Networks,Neural and Evolutionary Computing <https://arxiv.org/pdf/1303.5778.pdf>

[57] Sutskever, I., Vinyals, O., Le, Q. (2014) Sequence to Sequence Learning with Neural Networks, <https://arxiv.org/pdf/1409.3215.pdf>

[58] EditorialExpress.com (2016) A new look upon the Meese-Rogoff puzzle based on Support Vector Regression. *Journal of International Money and Finance* <https://editorialexpress.com/cgi-bin/conference/download.cgi?db_name=SBF2017&paper_id=10>

[59] W. Tian, T. Wang, B. Li, Risk measures with wang transforms under flexible skew-generalized settings, *International Journal of Intelligent Technologies and Applied Statistics* 7 (3) (2014) 185–205. doi:10.6148/IJITAS.2014.0703.01

[60] University of Pennsylvania. *Applied Time Series Analysis*. https://onlinecourses.science.psu.edu/stat510/node/47

[61] Gary Koop, D. K. *Bayesian Multivariate Time Series Methods for Empirical Macroeconomics*

[62] D.Hamilton, J. *Time Series Analysis.* Princeton University Press

[63] Otexts. *Stationarity and differencing*. https://www.otexts.org/fpp/8/1

[64] Iordanova, T.*Introduction To Stationary And Non-Stationary* https://www.investopedia.com/articles/trading/07/stationary.asp

[65] The Air Freight Index Company. *The TAC Index*.https://www.tacindex.com/

[66] Duke. (n.d.). *Introduction to ARIMA.* Retrieved from https://people.duke.edu/~rnau/411arim.htm

[67] Hyndman, R. J., & Athanasopoulos, G. *Forecasting: principles and practice*

[68] Dickey, D. A. Stationarity Issues in Time Series Models.

[69] Autocorrelation Plot. https://www.itl.nist.gov/div898/handbook/eda/section3/autocopl.htm

[70] Duke. Identifying *the order of differencing*. Retrieved from https://people.duke.edu/~rnau/411arim2.htm

[71] Duke University. (n.d.). *Rules for identifying ARIMA models*.https://people.duke.edu/~rnau/arimrule.htm

[72] Penn State University. (n.d.). *Partial Autocorrelation Function (PACF)*. Retrieved from https://onlinecourses.science.psu.edu/stat510/node/62

[73] Penn State Ebert college of science. *Vector Autoregressive models VAR(p) models*.https://onlinecourses.science.psu.edu/stat510/node/79

[74] Sims, C. *Structural VAR.*

[75] Viegi, N. *Introduction to VAR model.*

[76] Hauser, M. . *Vector Autoregression*

[77] A.M. Mood, . F. (n.d.). *Introduction to Theory of Statistics.*

[78] Wilks, S. S. (n.d.). The Large-Sample Distribution of the Likelihood Ratio for Testing Composite Hypotheses

[79] Mosayeb Pahlavani, R. R. . The Comparison among ARIMA and hybrid ARIMA.

[80] Bollerslev, T. Gloassary to GARCH.

[81] Paul Zarchan, H. M. (n.d.). *Fundamentals of Kalman Filtering.*

[82] G.W.Morrison, D. (n.d.). Kalman Filter Applied To Statistical Forecasting

[83] Frost, J.*Regression Analysis*. Retrieved from http://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit