# Classification of Univariate Random Time Series Data and Prediction using ARIMA Model

**Presented By :**

Siddhartha Narayana (711CS1029)

**Under the Guidance of:**

Dr. Bidyut Kumar Patra


**Dept. of Computer Science and Engineering**
**National Institute of Technology, Rourkela**
**Rourkela – 769008, Odisha, India**

# Abstract

The last decade has seen a huge interest in classification of time series. Most of this work assumes that the data resides in main memory and is processed offline. However, recent advances in sensor technologies require resource-efficient algorithms that can be implemented directly on the sensors as real-time algorithms. Throughout recent years, 1-nearest neighbor (1-NN) algorithm has remained as a robust similarity measure in time series classification (TSC) serving as a benchmark.

In our work, we have used the Sensor Data from OV-102 Columbia space shuttle of 1989. The IMU and gyroscope sensors provide the time series data upon which we have developed and implemented our algorithms on. We have first done analysis of the data like finding slope and variance with taking data in a timed constraint. Here we have used only the amplitude domain to do the classification and then prediction using ARIMA model. Also in the analysis phase, we have used Savitzky-Golay filter for check the seasonality and recurence of the data as well as to smooth the erratic data points, while all the same preserving the discrete timed information.

For Classification of data we have used the Nearest Neighbor Classification technique. Further, prediction of the next 100 data points has been done using various values of AR, I and MA. Since the data is random and the next stimuli in which the space vehicle will be arbitrary, hence the data points can be erratic. However, little room does exist for curve fitting and prediction and that is what we have tried to achieve in our work here. This paper is explains the method that we have applied and the results and inferences that we have achieved. Also, we have provided an insight on the work that can be done in this regard in the future.

# Contents

# 1 Introduction

The last decade has seen a huge interest in classification of time series (TSC) [1]. Most of this work assumes that the data resides in main memory and is processed offline. However recent advances in sensor technologies require resource-efficient algorithms that can be implemented directly on the sensors as real-time algorithms. substantial amounts of data has been collected and analyzed to assist in optimal deci- sion making. This collection and analysis of large datasets has become a primary contributor toward the innovation and growth of the current and emerging markets. A significant portion of this collected data is in the form of time series. The exponentially increasing volume and complexity of time series data is a result of emerging new sensing technolo- gies (robot sensors, wearable sensors, smart meters, satellites, smart mobile phones, etc.), along with an influx of inexpen- sive storage. The key objective of time series analysis is to extract hidden insights from raw data. Time series classifica- tion (TSC) is one of the most important tasks in time series analysis. There are a variety of TSC techniques in the litera- ture. Distance-based classification techniques such as 1-nearest neighbor (1-NN) require a similarity measure to calculate the distance (similarity) between two time series.

Euclidean distance (ED) [2] distance-measures for time series data is fast and efficient. However since one dimensional data is being considered here, hence Manhattan Distance is seen fitting and fast in our approach. 1-NN ED is often used for fast classification of the time series of equal length. However, its accuracy is highly sensitive to noise and it cannot be used when the lengths of time series are different. implementation of 1-NN using Manhattan Distance reduces the computational complexity by omitting square root calculations, using lower bounding, and applying early aban- doning. However, it is still inefficient when classifying long time series and large datasets. Here, for faster calculation we transformed raw time series data, from the 2-D space of time-values, to a sequence of segments each forming a tuple of (segment begin time, segment average of values, segment duration); with the durations being z- normalized across the training time series. The data reduction results in reduced noise, space, and length of the time series before classification.

# 2 Work Done

A time series T$(t_1 , t_2 , . . . t_n)$, is a sequence set of n real values recorded over time. Defining a similarity/distance measure between two time series is at the

core of most TSC techniques. The Manhattan Distance is given by 1. Generally, raw time series contain various sorts of distor- tion and noise, which makes their comparison difficult. The common distortions and invariances are amplitude and off- set invariance, local scaling (warping) invariance, uniform scaling invariance, phase invariance, occlusion invariance and complexity-invariant distortions.

$$D = x_i - x_j \quad \text{for distinct } i, j \in \{1, \ldots, n\} \tag{1}$$

## 2.1 Smoothing: Savitzky Golay Filter

The data obtained from sensors is quite erratic and is not suitable for classification as there is noise involved and sheldom does real time objects show erratic behaviour in their locomotion. Hence we have employed a Savitzky Golay Filter. Savitzky–Golay filter is a digital filter that can be applied to a set of digital data points for the purpose of smoothing the data, that is, to increase the signal-to-noise ratio without greatly distorting the signal. This is achieved, in a process known as convolution, by fitting successive sub-sets of adjacent data points with a low-degree polynomial by the method of linear least squares. When the data points are

equally spaced, an analytical solution to the least-squares equations can be found, in the form of a single set of "convolution coefficients" that can be applied to all data sub-sets, to give estimates of the smoothed signal, (or derivatives of the smoothed signal) at the central point of each sub-set [3]. The data consists of a set of n xj, yj points (j = 1, ..., n), where x is an independent variable and $x_i$, $y_j$ is an observed value. They are treated with a set of m convolution coefficients, $C_i$, according to the expression given in 2.

$$Y_j = \sum_{i=-(m-1)/2}^{(m-1)/2} C_i y_{j+i} \quad i, j \in \{1, \ldots, n\} \tag{2}$$

A demonstration of this filter is given in Figure.1 and 2. TSBs are aggregated representations of time series. Another aggregation scheme is presented in [4], where data maps are created that represent the sensory data as well as temporal and spatial details associated with a given segment of data. However, these data maps are not analyzed in real-time, but deposited at sink nodes that are more powerful for pattern analysis.

In order to analyse these data, seasonality and trend are to be understood. A time series data has the following two characteristics.

- **1 Trend:** varying mean over time. For eg, in this case we saw that on average, the number of passengers was growing over time.

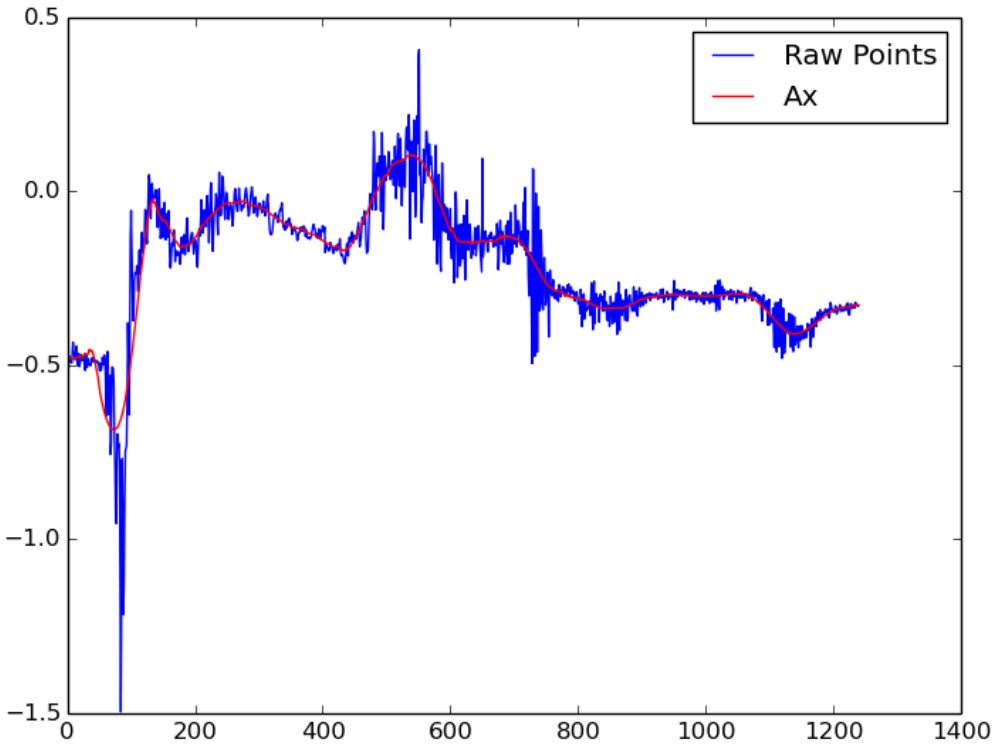


Figure 1: Smooth Data, overlapped over the raw TSD Acceleration X(erratic)

- **2 Seasonality:** variations at specific time-frames. eg people might have a tendency to buy cars in a particular month because of pay increment or festivals.

## 2.2 Classification: Nearest Neighbor

The nearest neighbour algorithm is easy to implement and executes quickly, but it can sometimes miss shorter routes which are easily noticed with human insight, due to its "greedy" nature. As a general guide, if the last few stages of the tour are comparable in length to the first stages, then the tour is reasonable; if they are much greater, then it is likely that there are much better tours. Another check is to use an algorithm such as the lower bound algorithm to estimate if this tour is good enough.

In the worst case, the algorithm results in a tour that is much longer than the optimal tour. To be precise, for every constant r there is an instance of the traveling salesman problem such that the length of the tour computed by the nearest neighbour algorithm is greater than r times the length of the optimal tour.

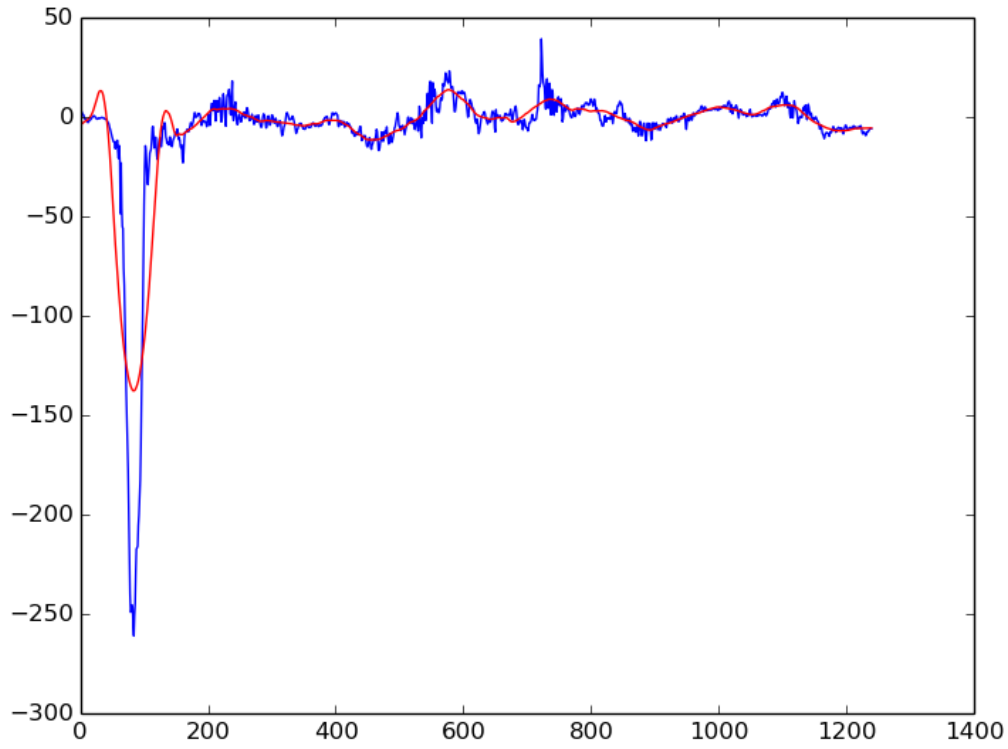


Figure 2: Smooth Data, overlapped over the raw TSD Gyroscope Z(erratic)

Moreover, for each number of cities there is an assignment of distances between the cities for which the nearest neighbor heuristic produces the unique worst possible tour.

## 2.3 Prediction : ARIMA

Autoregressive Integrated Moving Average (ARIMA)[5] model is a generalization of an autoregressive moving average (ARMA) model. Both of these models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting). ARIMA models are applied in some cases where data show evidence of non-stationarity, where an initial differencing step (corresponding to the "integrated" part of the model) can be applied one or more times to eliminate the non-stationarity.

The AR part of ARIMA indicates that the evolving variable of interest is regressed on its own lagged (i.e., prior) values. The MA part indicates that the regression error is actually a linear combination of error terms whose values occurred contemporaneously and at various times in the past. The I (for

"integrated") indicates that the data values have been replaced with the difference between their values and the previous values (and this differencing process may have been performed more than once). The purpose of each of these features is to make the model fit the data as well as possible.

Non-seasonal ARIMA models are generally denoted ARIMA(p,d,q) where parameters p, d, and q are non-negative integers, p is the order (number of time lags) of the autoregressive model, d is the degree of differencing (the number of times the data have had past values subtracted), and q is the order of the moving-average model. Seasonal ARIMA models are usually denoted ARIMA(p,d,q)(P,D,Q)m, where m refers to the number of periods in each season, and the uppercase P,D,Q refer to the autoregressive, differencing, and moving average terms for the seasonal part of the ARIMA model.[2][3]

When two out of the three terms are zeros, the model may be referred to based on the non-zero parameter, dropping "AR", "I" or "MA" from the acronym describing the model. For example, ARIMA (1,0,0) is AR(1), ARIMA(0,1,0) is I(1), and ARIMA(0,0,1) is MA(1). The forecast intervals (confidence intervals for forecasts) for ARIMA models are based on assumptions that the residuals are uncorrelated and normally distributed. If either of these assumptions does not hold, then the forecast intervals may be incorrect. For this reason, researchers plot the ACF and histogram of the residuals to check the assumptions before producing forecast intervals.

# 3 Results and Inferences

The Sensors data recieved is classified into 3 classes. Namely ***Fast***, ***Very Fast*** and ***Vibration***. When Nearest Neighbor Algorithm was applied to the slopes of the TSD so as to retain the properties of the readings thus received as shown in Figure 3 and 4

The Prediction Part we have made use of the ARIMA model. ARIMA model has 3 parameters namely **`AR, I and MA`**. These values depend on the seasonality and trend of the data. Now since we are using a random univariate TSD, finding best values for AR, I, MA is tricky, Hence we are to try with every value possible in the range for AR[0...4], I[0..2], MA[0...6]. And hence then predict the next 100 points and check the validity. The Figure.5. show the predicted values for each model for each dimension of the readinh recieved from the sensor.

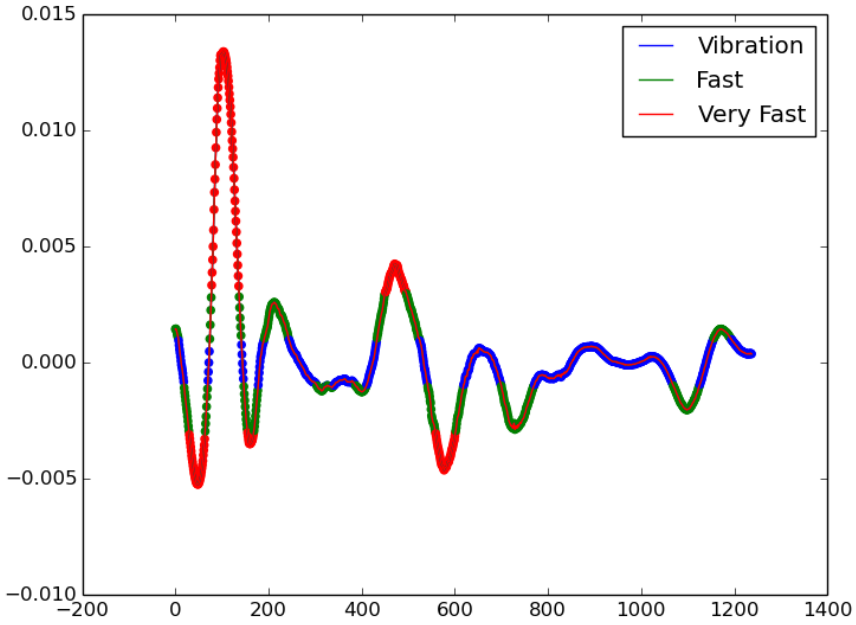


Figure 3: Classified TSD with Linear Acceleration X direction: Classes shown in labels

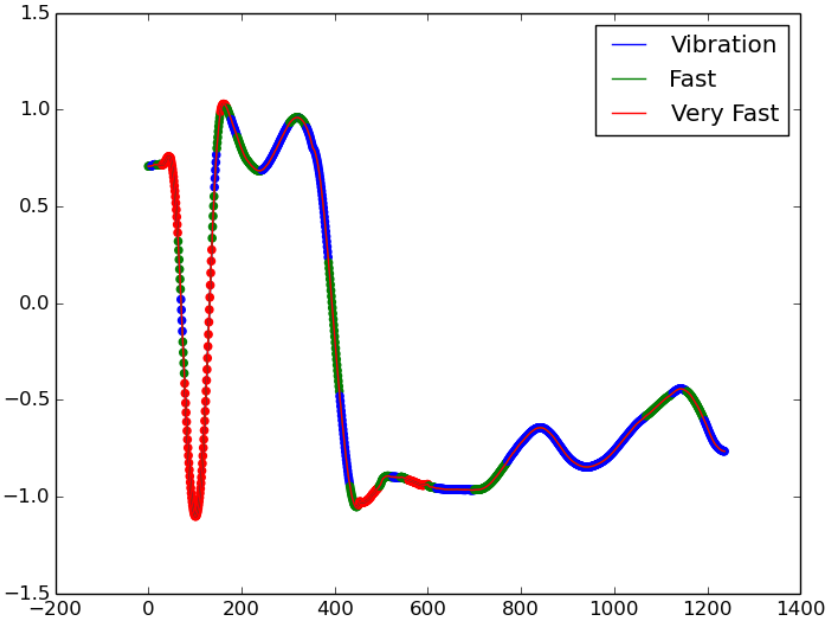


Figure 4: Classified TSD with Linear Acceleration Y direction: Classes shown in labels

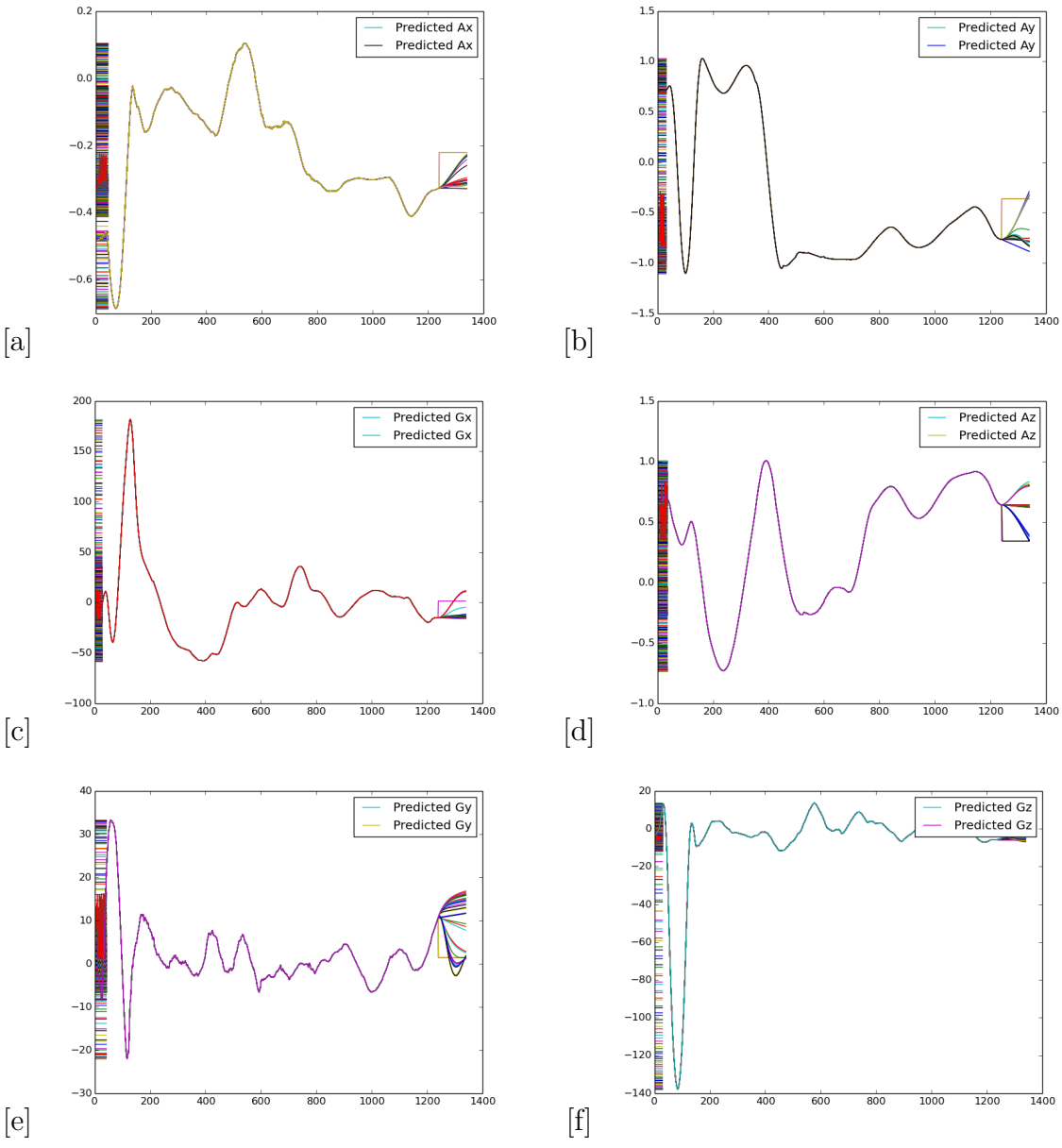


Figure 5: Predicted Sensor Information for the next 100 time instaneces with ARIMA model

# 4 Future Work and Conclusion

Further feature extraction from TSD may provide better overlook at classification. The feature patterns become attributes in the machine learning, and are improved by using the genetic algorithm. The improved feature patterns are useful for improving the classification accuracy. The experimental results demonstrate the effectiveness of the system. In future work, we will try to apply this method to other data such as music, medical data, spam mail, and etc. Approaches like SVM and regression can be applied.Further, a the TSD of various data can be implemented as cros sectional data from different sensors and dimensions at the same time instance. Time Series bitmaps can be done for faster analysis and to get a detailed idea of any robot in time.

# References


[1] E. Keogh, S. Lonardi, and C. Ratanamahatana. *Towards Parameter-Free Data Mining*. In Proc. of the 10th ACM SIGKDD, 2004.

[2] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, *"Fast sub- sequence matching in time-series databases,"* SIGMOD Rec., vol. 23, no. 2, pp. 419–429, Jun. 1994. [Online]

[3] Schafer, Ronald W. *"What is a Savitzky-Golay filter?[lecture notes]."* IEEE Signal processing magazine 28.4 (2011): 111-117.

[4] Li, Mo, Yunhao Liu, and Lei Chen. *"Nonthreshold-based event detection for 3D environment monitoring in sensor networks."* IEEE Transactions on Knowledge and Data Engineering 20.12 (2008).

[5] Box, George EP, and David A. Pierce. *"Distribution of residual autocorrelations in autoregressive-integrated moving average time series models."* Journal of the American statistical Association 65.332 (1970): 1509-1526.