# Modeling of visco elastic deformable objects for virtual environments

Shashi Prakash: B19EE076
Gautam Kumar: B19EE031
Supervisor: Dr. Amit Bhardwaj
Course Number: EEN2020
Date: 28/11/2021
Semester: 1

# Introduction

1. In this project, we propose a new data-driven approach for haptic modeling of normal interactions on homogeneous viscoelastic deformable objects.
2. The approach is based on some well-known machine learning techniques: Support Vector Regressor and Deep Learning.
3. In this project we are basically focusing in reducing the root mean squared error between actual signal and predicted signal. We have considered high and low amplitude points for training and middle amplitude points for testing the performance of our model.

# Motivation

- The existing methods of Modeling visco elastic deformable objects is very time consuming and less accurate.
- The existing method uses Random Forest with 100 decision trees to estimate the Force variation.
- Hence we thought to come up with two techniques which takes less training time and predicts more accurate results.

# Objectives and targeted deliverables

- Our objective was to come up with a technique or model which can learn the input output relationship of Visco Elastic Deformable objects and give a comparable performance as Random Forest.
- We tried different models with different hyperparameter tuning techniques and analyzed their behaviour under different input conditions.

# Data Description

1. The outlined automatic data acquisition is performed for all combinations of trajectory parameters, i.e., 3 deformation amplitudes $\{\gamma 1,\gamma 2,\gamma 3\}$ , 3 angular frequencies $\{\omega 1,\omega 2,\omega 3\}$ , and 13 phase shifts $\{\phi j\}13j=1$, per contact location pair (i.e., one per opposite surface). Thus, there are $3\times3\times13\times2=117\times2$ discrete-time force curves, obtained from the two force sensors, per contact pair.
2. In addition to the force measurements, matching discrete-time displacements is also stored in vectors d1k, d2k, again separately per contact surface. Displacement readings are provided by the device encoders.
3. In total our 9 datasets consist of $174,672\times72\times9$ interaction data points and force values.

# Implementation Step

1. Firstly, we have separate out desired data that is going to be used in developing an algorithm using Matlab.
2. Then, we have selected a python language for implementation. Python is used in this project for developing as it has Scikit-Learn Library which has many tools that can be used for data training.
3. Then, we have selected an optimal algorithm that is deep learning for training and testing the data.
4. After that we have pre-processed the data.
5. Then, we have done data training with Random Forest, Deep learning and support vector machine using the various input vector and output data.

   1. Taking positions (X1, X2) points as input data and output vector as

   force(F1).

   2.Taking also various stages of past input into consideration and the same

   output vector as force(F1).

   3. Taking fractional derivatives of input vector and output vector as force(F1).

6. We have also done tuning of parameters for better result.

# Implementation

1. We have built a Deep Learning Model which can give comparable performance as Random Forest model.
2. We have also tested the performance of Support Vector Machines and found it to perform significantly well on the given data.
3. We have also noticed the training time of each model and ensured that our models take lower time than pre-existing algorithms to get trained.
4. We have considered several input parameters including present as well as various stages of past inputs and an analytical comparison of models is performed in each case.
5. The codes and report files are added and updated in the following github repository.

   https://github.com/gautamHCSCV/Modelling_Viscoelastic_Objects

# *Data Analysis*

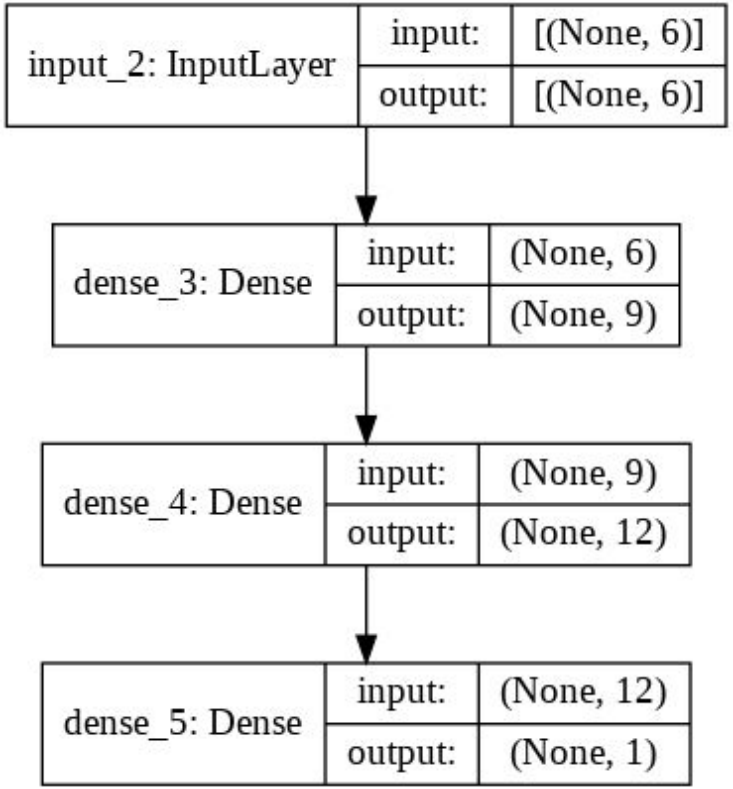
- Number of train data points = 120435
- Number of Test data points = 54235 (predicting middle amplitude points)

PAST 20 INPUTS COMPARISON:

- x1(n), x1(n-$\boldsymbol{\tau}$), x1(n-2*$\boldsymbol{\tau}$) … x1(n-m*$\boldsymbol{\tau}$) and

  x2(n), x2(n-$\boldsymbol{\tau}$), x2(n-2*$\boldsymbol{\tau}$) ... x2(n-m*$\boldsymbol{\tau}$)
- For $\boldsymbol{\tau} = 4$ and m = 5
- For $\boldsymbol{\tau} = 5$ and m = 4

# *DEEP LEARNING MODEL STRUCTURE*

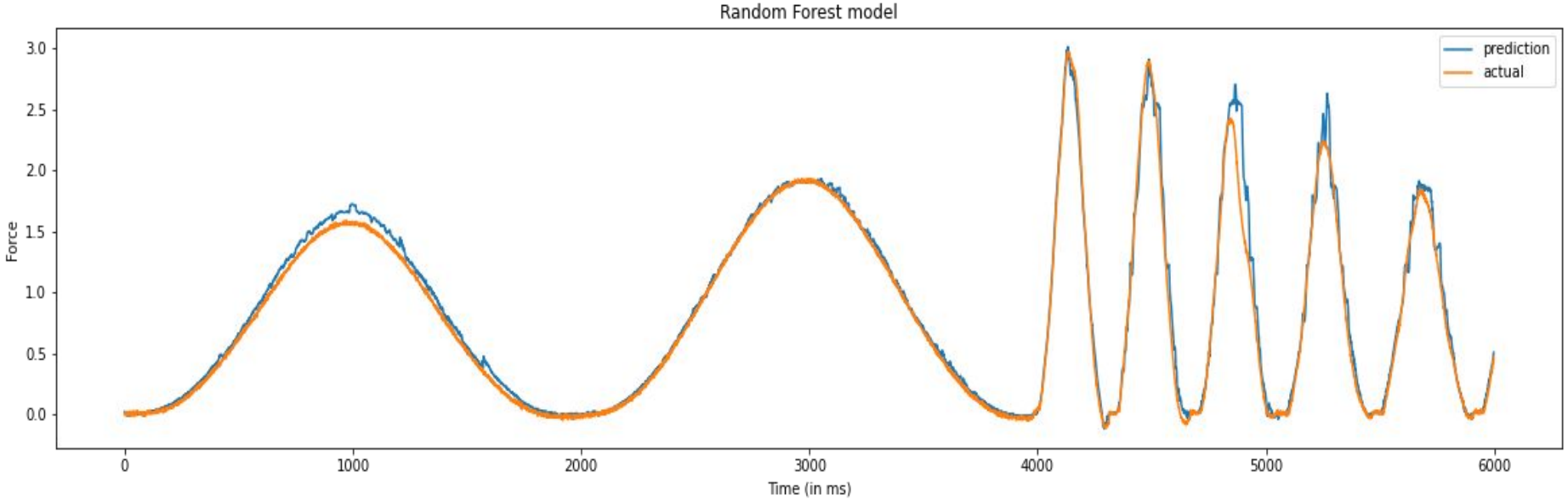


| Input feature vs output feature | $\tau$ = 4 and m = 5<br><br>f1 | | | $\tau$ = 5 and m = 4<br><br>f1 | | |
|---|---|---|---|---|---|---|
| Model | DL model | SVR | RF | DL model | SVR | RF |
| Max error | 0.8490 | 0.690 | 0.8900 | 0.8183 | 0.6861 | 0.9079 |
| Min error | 1.251 * 10^-6 | 3.915 * 10^-7 | 1.387 * 10^-17 | 4.479 * 10^-7 | 8.253 * 10^-8 | 9.999 * 10^-7 |
| Medin error | 0.0215 | 0.0306 | 0.0188 | 0.0319 | 0.0311 | 0.01901 |
| RMSE | 0.0506 | 0.0509 | 0.060 | 0.061 | 0.0511 | 0.0598 |

# Predictions

Random Forest

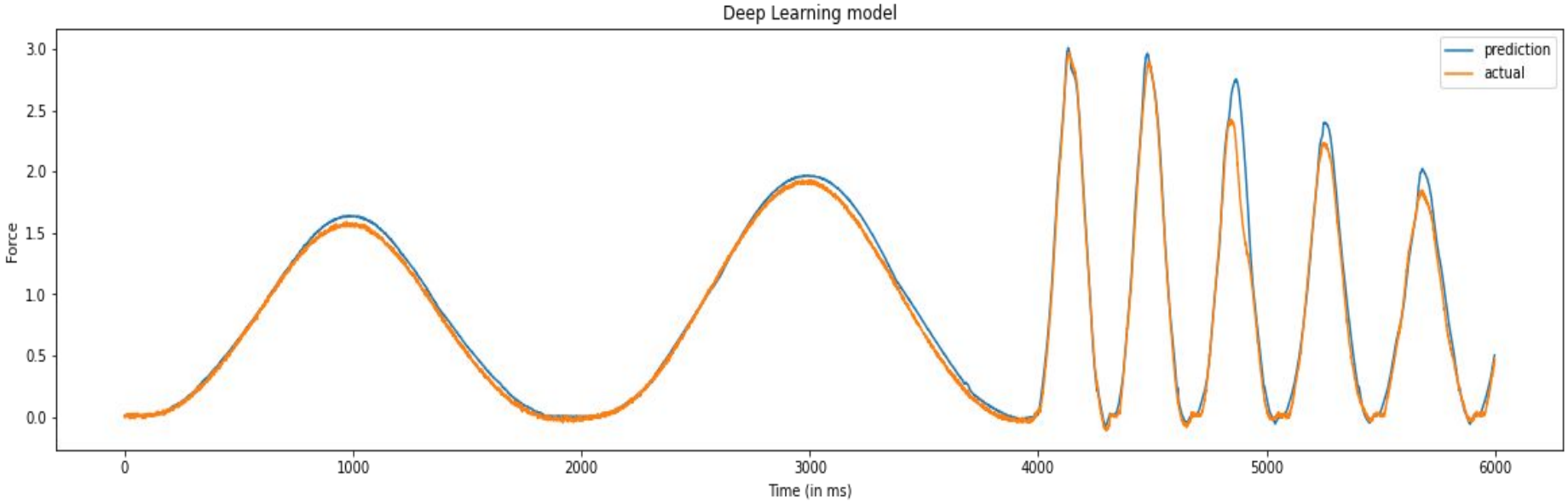
Root Mean squared error =  0.060872



# Predictions

*Deep Learning Model*

*Root Mean squared error = 0.047670*



# Random Forest Vs Deep Learning

**Models' performances on Middle amplitude points**

| Input feature vs output feature | p1,p2<br><br>f1 | | p1[i+1] - p1[i], p2[i+1] - p2[i]<br><br>f1 | | p1[i],p1[i-$\tau$],p1[i-2*$\tau$]<br>p2[i],p2[i-$\tau$],p2[i-2*$\tau$]<br><br>f1 | |
|---|---|---|---|---|---|---|
| Model | DL model | RF | DL model | RF | DL model | RF |
| Time taken by the model(in sec) | ***0:00:27.6*** | **0:01:22.6** | ***0:01:15.9*** | **0:01:12.7** | ***0:01:15.0*** | **0:01:22.3** |

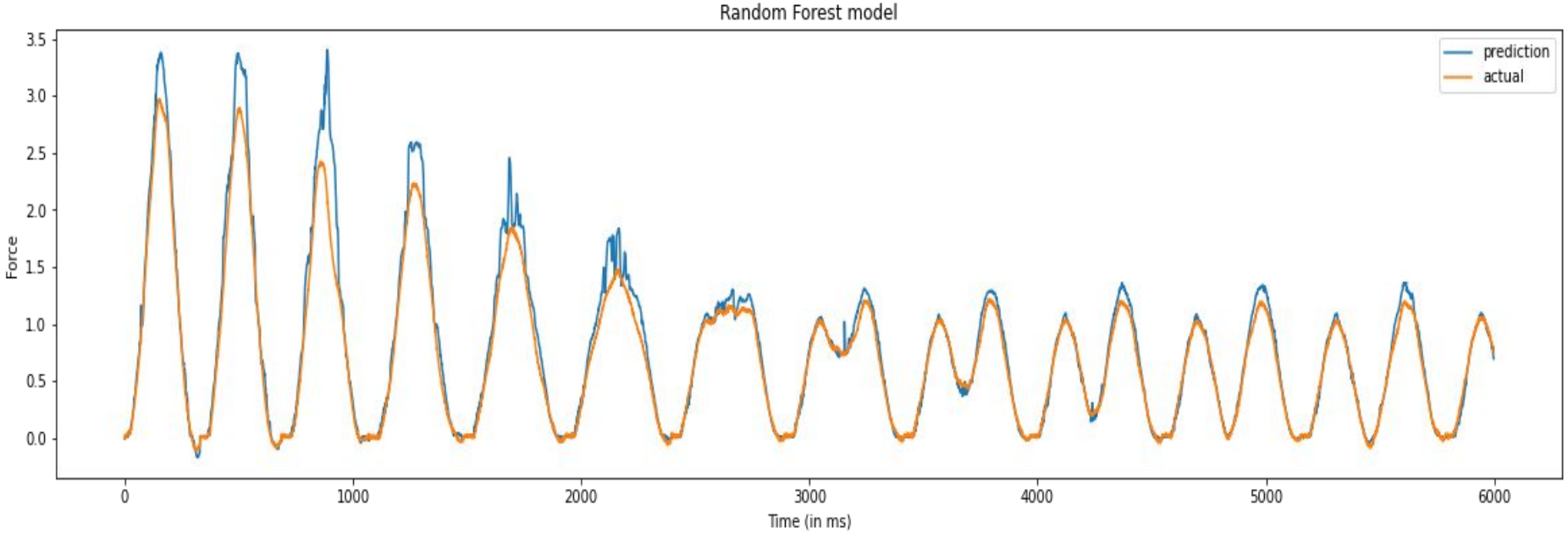
# *ANALYSIS OF DATA AND ENVIRONMENT*

- Number of train data points = `116447`
- Number of Test data points =58223 (predicting High data points and low data points)
- input features: **fractional derivative of input data points X1 and X2 at order = 0.1 to 1**
- Output feature = **Force (F1)**
- Optimizer = Adam
- Loss function = Root Mean Squared Error
- Version of TensorFlow used = 2.6.0

# Predicting Middle Amplitude points(39-78 Interaction points) for a lower time period using fractional derivative

Random Forest (CLOSER VIEW OF INITIAL POINTS)

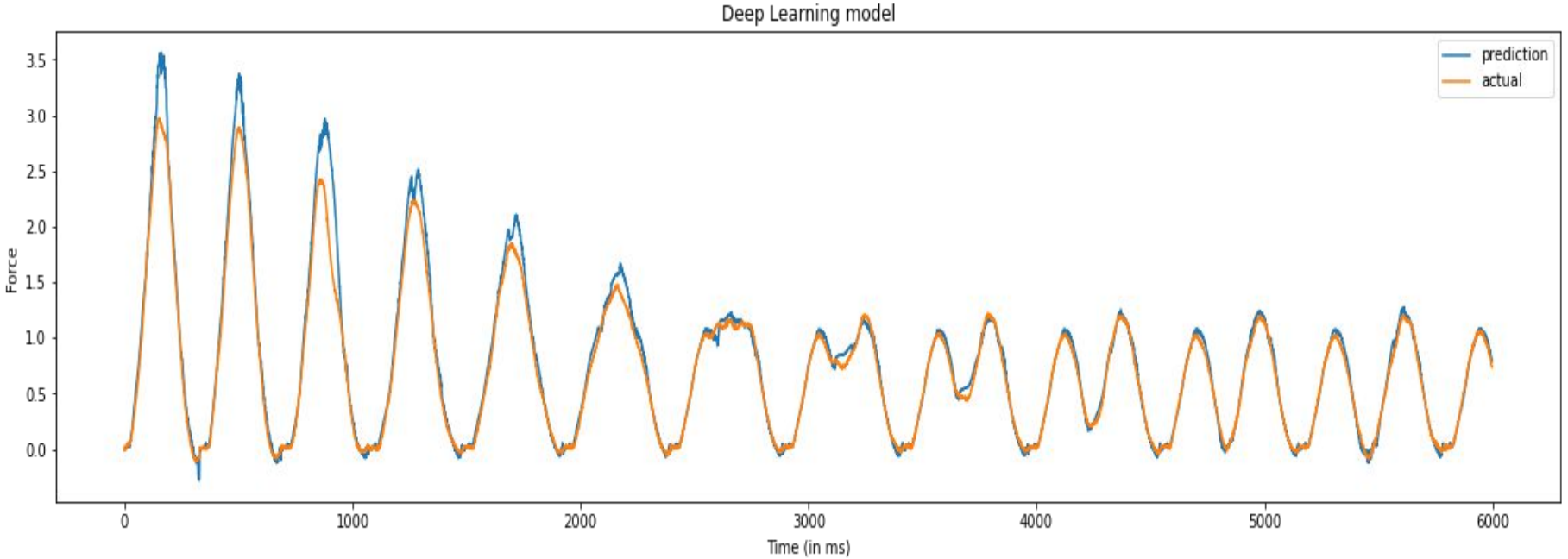
Root Mean squared error = 0.070862



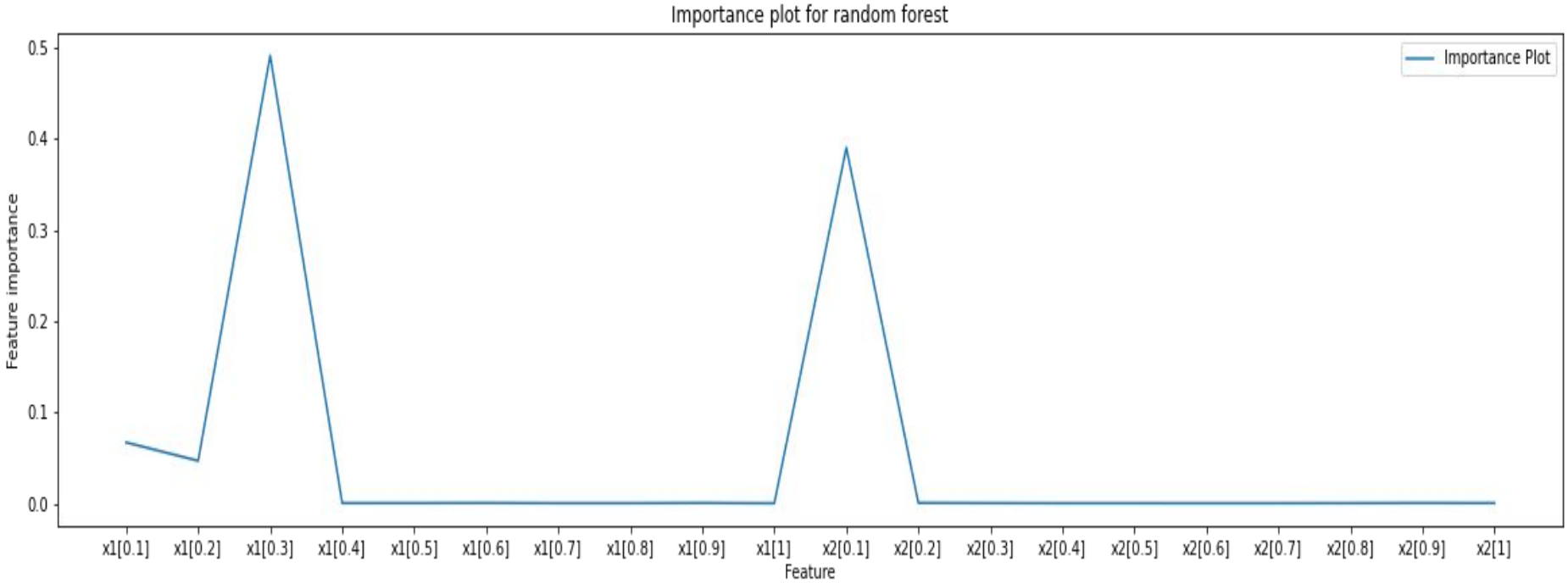
# Predicting Middle Amplitude points(39-78 Interaction points) for a lower time period using fractional derivative

*Deep Learning Model* (CLOSER VIEW OF INITIAL POINTS)

Root Mean squared error = 0.052984



# *Feature Importance plot for random forest using fractional derivative*



# Conclusion

1. We had got different results for models with different hyperparameter tuning techniques.
2. We got best results in the case of the deep learning model and support vector machine compared to the existing model Random Forest.
3. We had also got better results in the case of fractional derivatives.

# Thank You