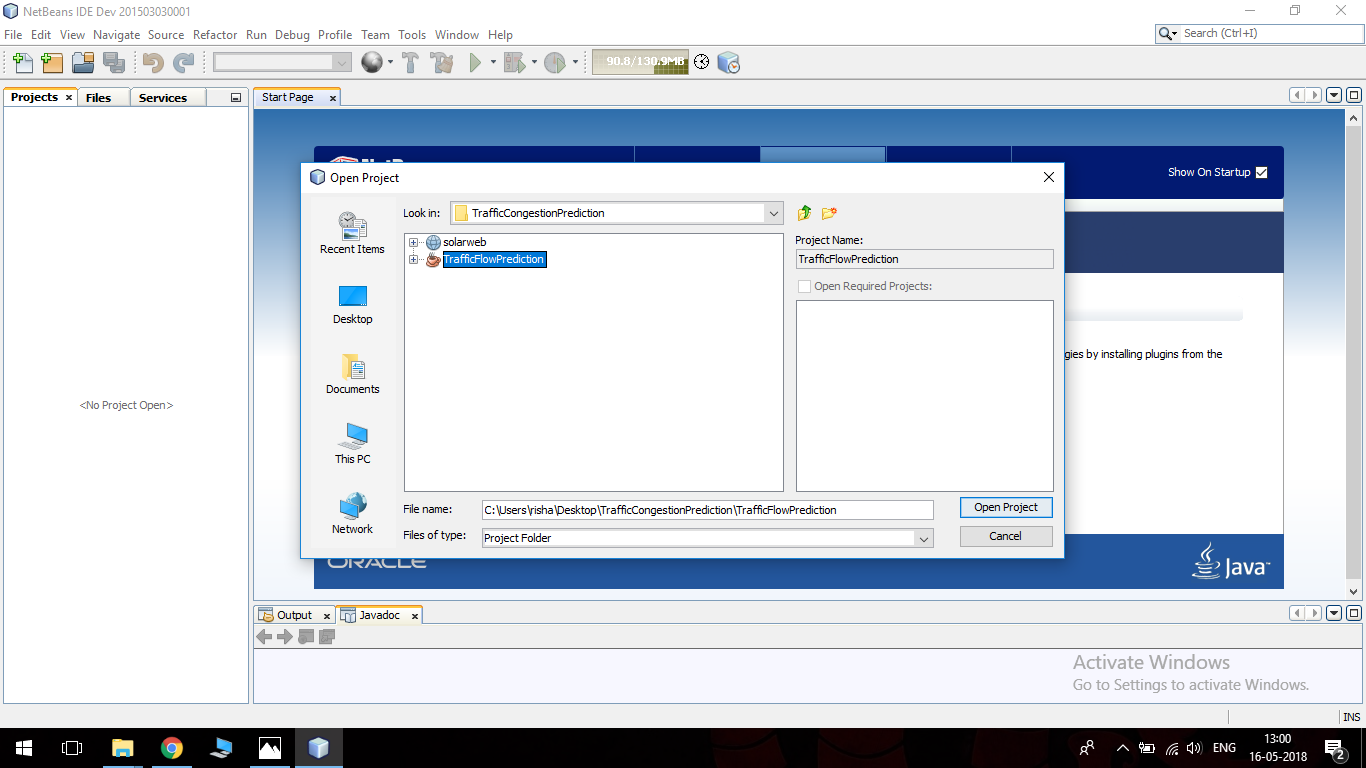
**INTERPRETATION OF RESULT**

The project is implemented in java and the snapshots of the project are given below with explanation for each snapshot.

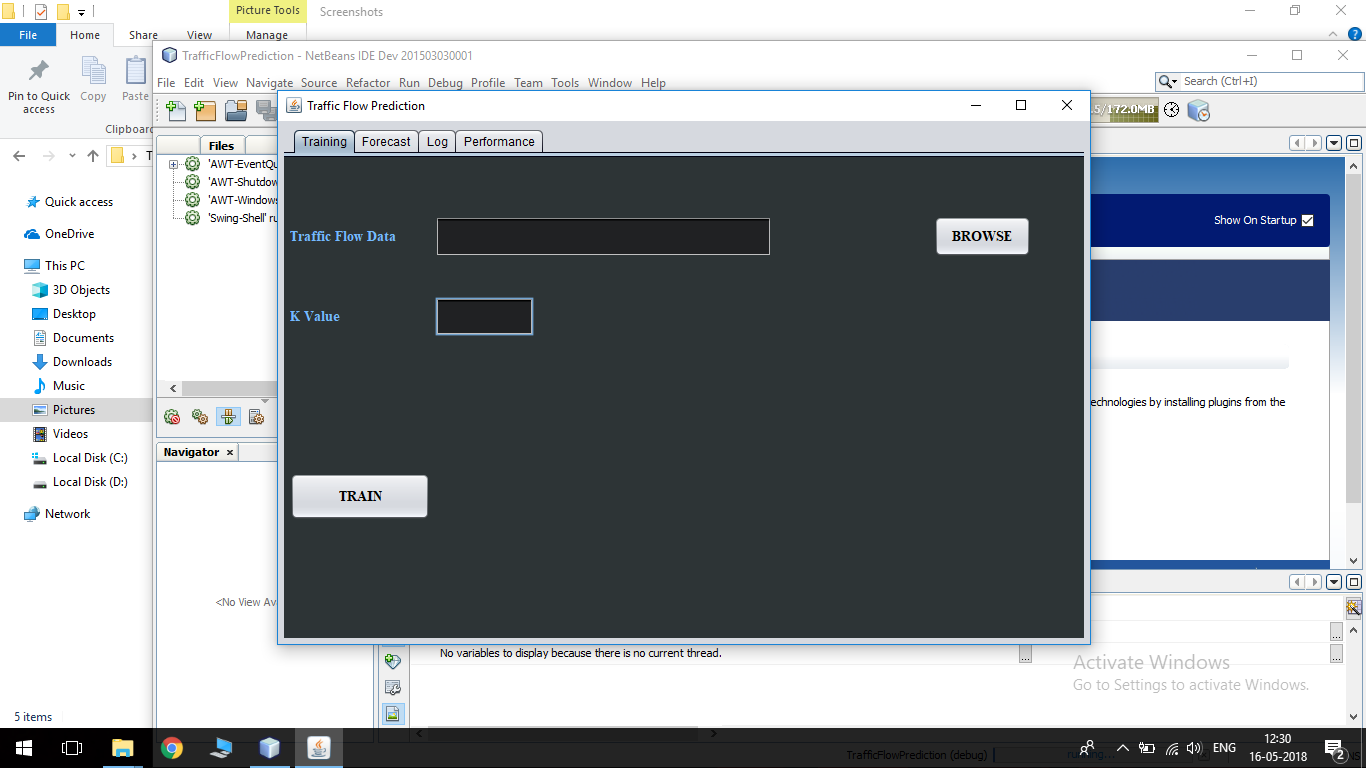
**1.1 Open NetBeans IDE**

**Fig. 1.1 Open Project in NetBeans IDE**

**Description:**

The project coded in Java language is opened in NetBeans Integrated Development Environment.

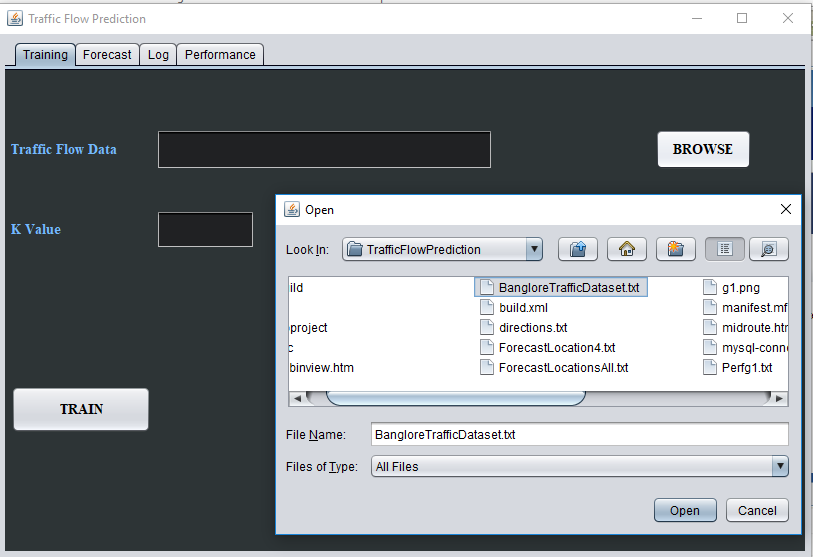
**1.2 Run the project**

**Fig. 1.2 Run Project**

**Description:**

The project opened in NetBeans IDE is run by clicking the green play button on the menu bar.

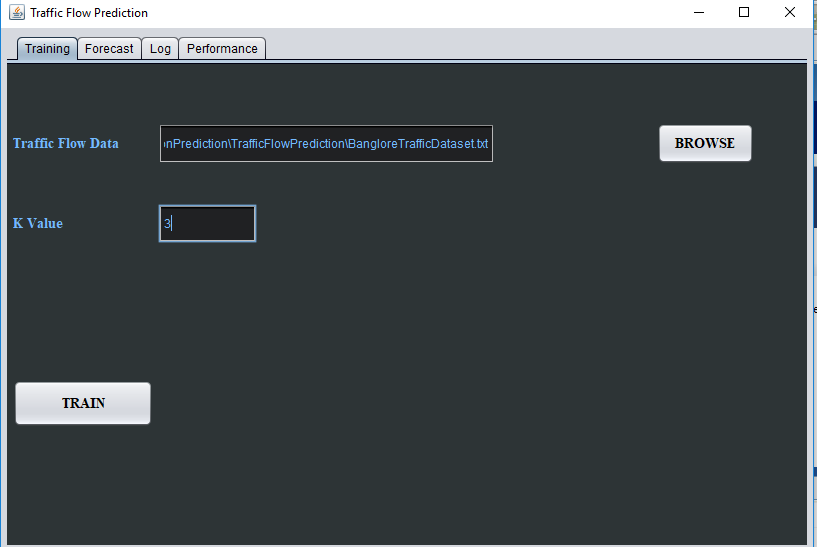
**1.3 Load Traffic Dataset**

**Fig. 1.3 Load Traffic Dataset**

**Description:**

Once the project starts execution, the GUI opens. Browse and load the text file that contains the required traffic dataset to train the model.

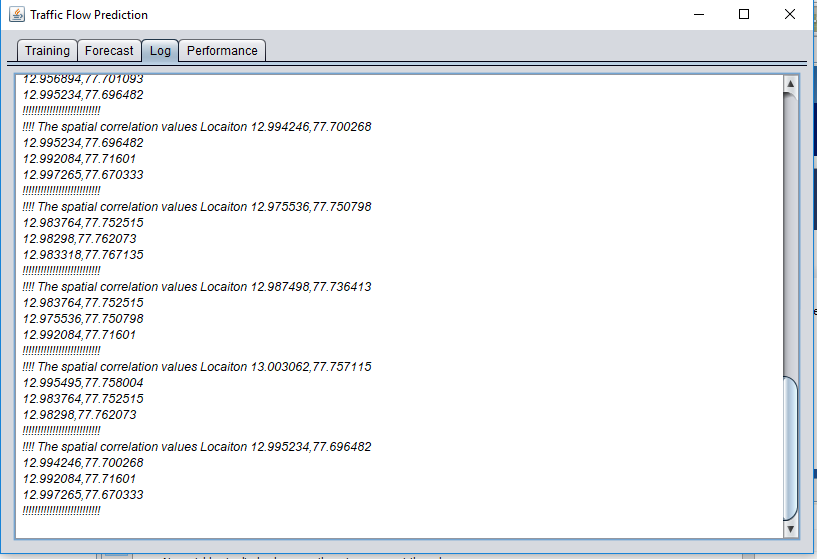
**1.4 Train the model**

**Fig.1.4 Train the model**

**Description:**

After browsing and uploading the traffic dataset, enter the required K-value and click the train button to start training.

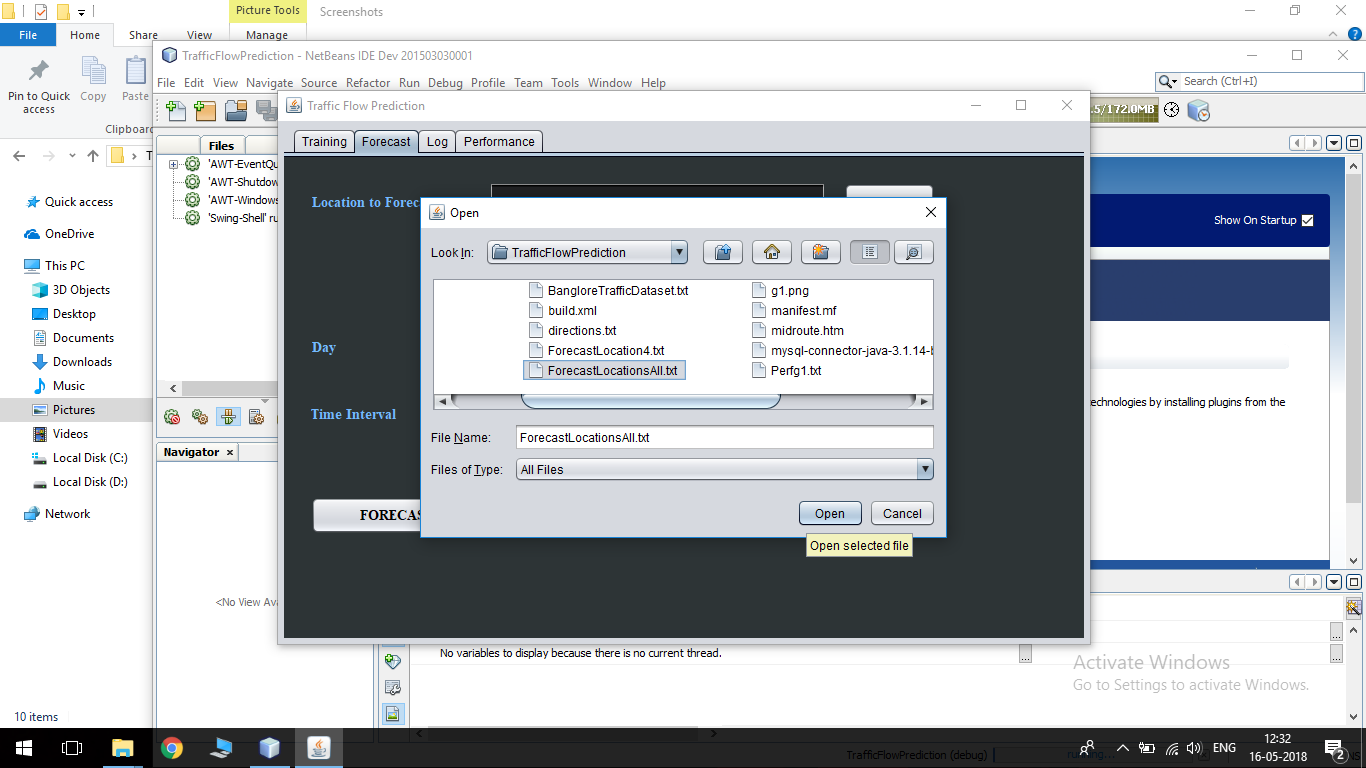
**1.5 The spatial correlation values are displayed in Log.**

**Fig. 1.5 Spatial Correlation Values**

**Description:**

Once the training is complete, the spatial correlation values for each location is displayed on the Log.

**1.6 Load Forecast Locations**

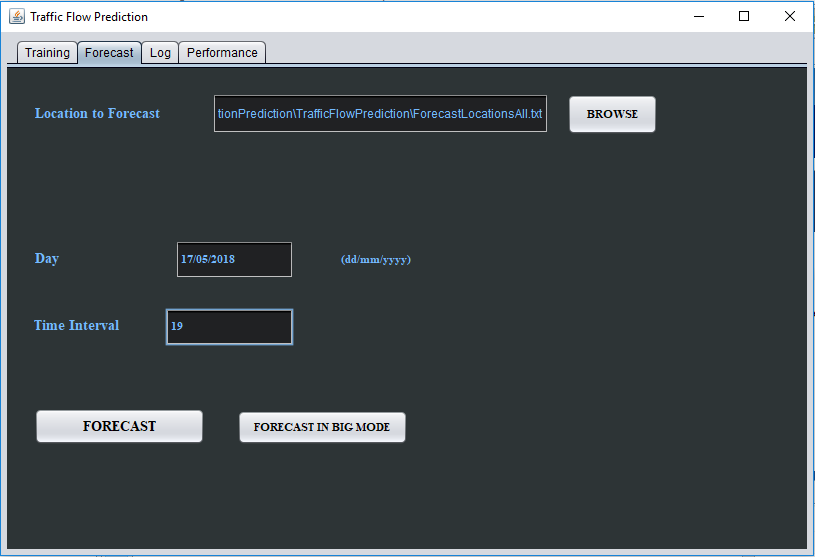


**Fig. 1.6 Load Forecast Location**

**Description:**

Browse and upload the text file containing the locations for which the traffic congestion should be forecasted.

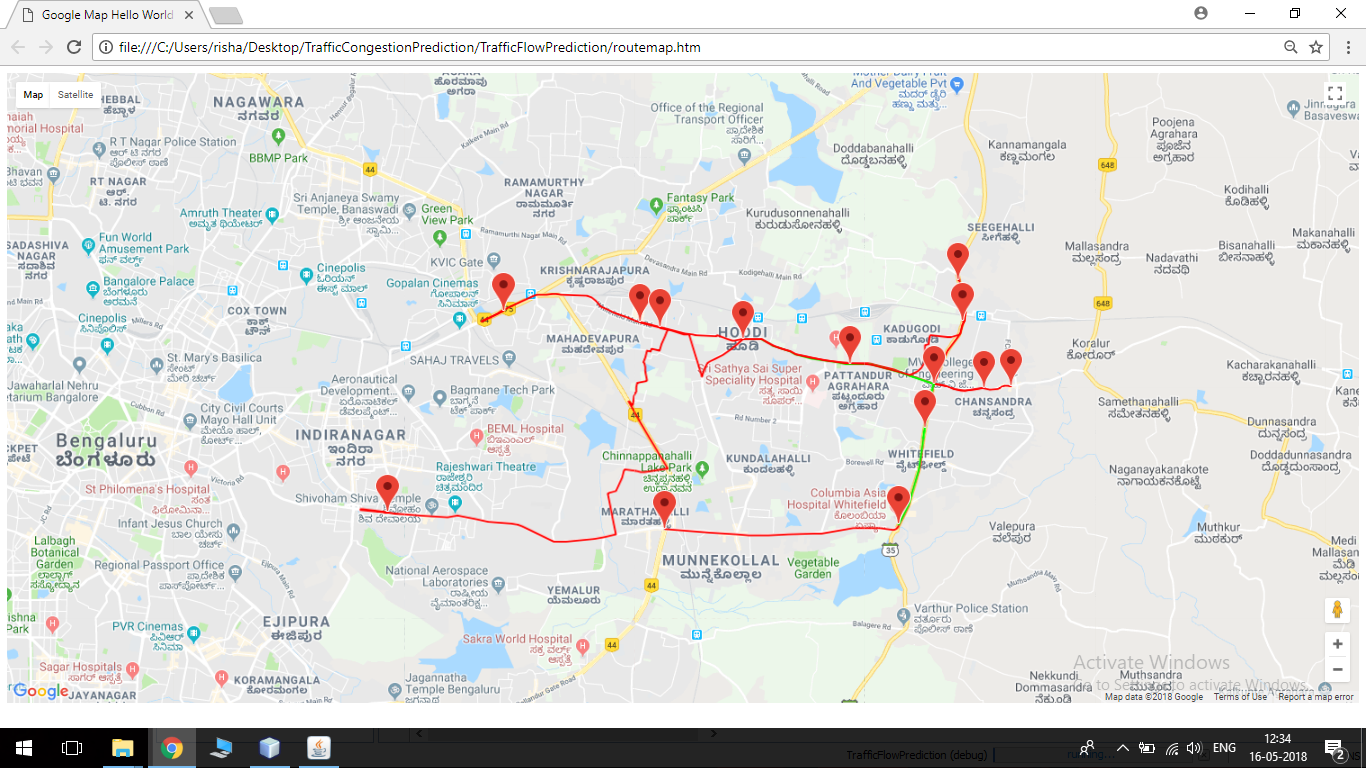
**1.7 Forecast**

**Fig. 1.7 Forecast**

**Description:**

Enter the Date and Time interval and click the Forecast button to begin forecasting.

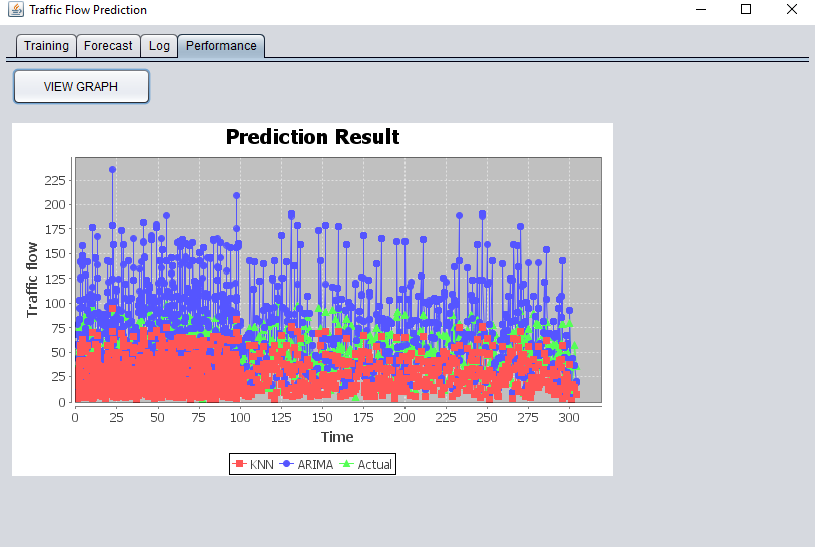
**1.8 Output on Google Maps API**

**Fig. 1.8 Predicted Values on Google Maps API**

**Description:**

The predicted traffic congestion values are displayed on the Google Maps API. Roads marked in red indicate high traffic congestion. Roads marked in green indicate low traffic congestion.

**1.9 Performance Graph**

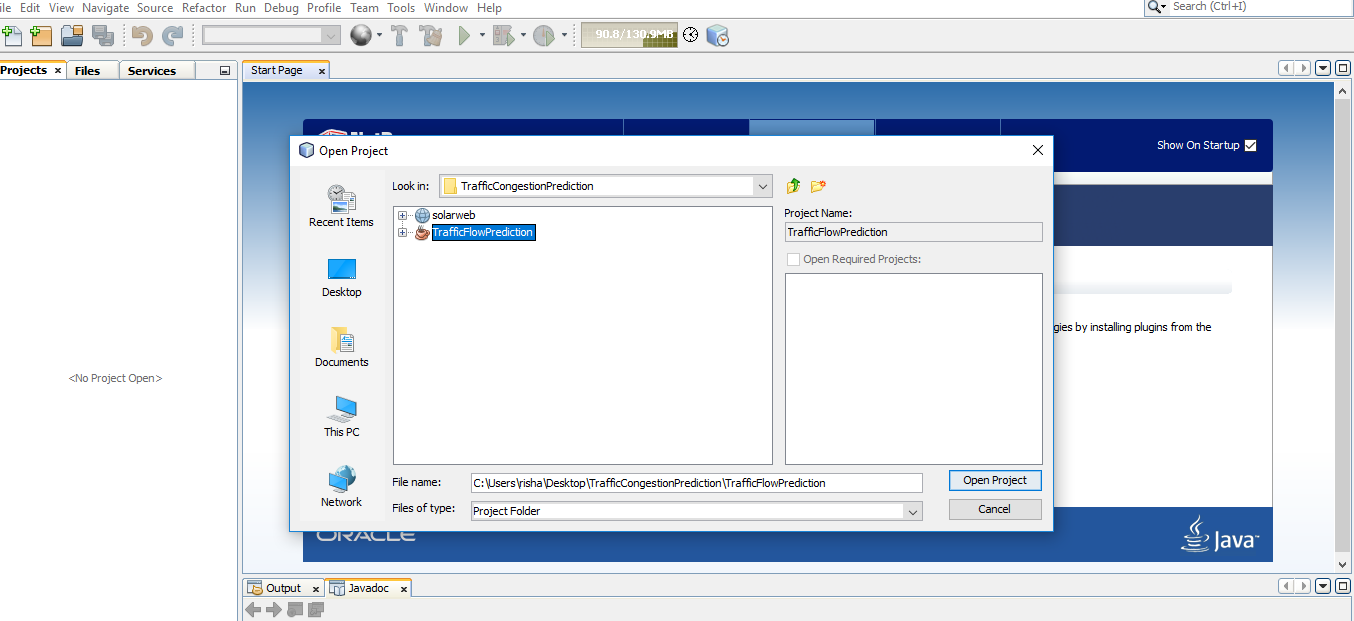
**Fig. 1.9 Performance Graph**

**Description:**

Click on performance tab to view the graph. The forecasted values using KNN and ARIMA are compared with real values. From this, we can see that the values predicted by KNN are very close to the actual values.

To show the project is scalable, we are running it in Hadoop using Linux OS in VMware.

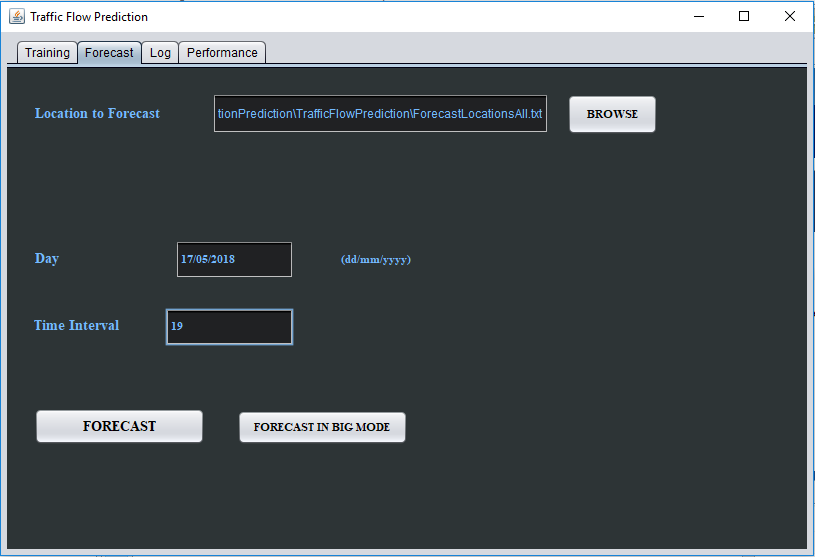
**1.10 Run NetBeans in Linux**

**Fig. 1.10 NetBeans in Linux**

**Description:**

The project coded in Java language is opened in NetBeans IDE in Linux.

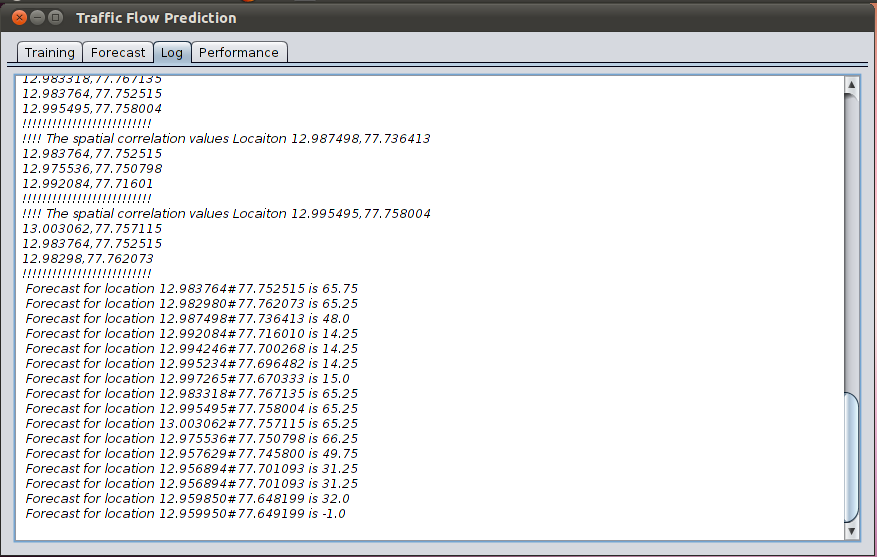
**1.11 Running in Big Data mode**

**Fig. 1.11 Forecast in Big Data Mode**

**Description:**

After training the model with the required traffic dataset, upload the necessary forecast location, enter the day and time interval. Click “Forecast in Big Mode” to run the project in Hadoop.

**1.12 Output in Log**

**Fig. 1.12 Result in Big Data Mode**

**Description:**

The predicted traffic congestion values in the given forecast locations are processed in Hadoop and displayed in the Log.