Himanshu Mittal

student, Manipal university Jaipur

Traffic forecasting

using Deep learning

**Introduction**

**Transportation forecasting**

**Transportation forecasting** is the attempt of estimating the number of vehicles or people that will use a specific transportation facility in the future. For instance, a forecast may estimate the number of vehicles on a planned road or bridge, the ridership on a railway line, the number of passengers visiting an airport, or the number of ships calling on a seaport.

Traffic forecasting begins with the collection of data on current traffic. This traffic data is combined with other known data, such as population, employment, trip rates, travel costs, etc., to develop a traffic [demand](https://en.wikipedia.org/wiki/Demand) [model](https://en.wikipedia.org/wiki/Mathematical_model) for the current situation.

The current technologies facilitate the access to dynamic data, big data, etc., providing the opportunity to develop new algorithms to improve greatly the predictability and accuracy of the current estimations

Traffic forecasts are used for several key purposes in transportation policy, [planning](https://en.wikipedia.org/wiki/Transportation_planning), and [engineering](https://en.wikipedia.org/wiki/Transportation_engineering): to calculate the capacity of infrastructure, e.g., how many lanes a bridge should have; to estimate the financial and social [viability](https://en.wikipedia.org/wiki/Viability_study) of projects, e.g., using [cost–benefit analysis](https://en.wikipedia.org/wiki/Cost%E2%80%93benefit_analysis) and [social impact assessment](https://en.wikipedia.org/wiki/Social_impact_assessment); and to calculate [environmental impacts](https://en.wikipedia.org/wiki/Environmental_degradation), e.g., air pollution and noise.

**Traffic forecasting using Deep learning**

Traffic forecasting predictions can be achieved using Deep learning algorithms like Long Short-term memory models

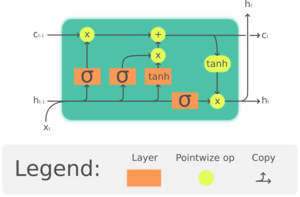
**Long short-term memory** (**LSTM**) is an artificial [recurrent neural network](https://en.wikipedia.org/wiki/Recurrent_neural_network) (RNN) architectureused in the field of [deep learning](https://en.wikipedia.org/wiki/Deep_learning). Unlike standard [feedforward neural networks](https://en.wikipedia.org/wiki/Feedforward_neural_network), LSTM has feedback connections.

It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected [handwriting recognition](https://en.wikipedia.org/wiki/Handwriting_recognition), [speech recognition](https://en.wikipedia.org/wiki/Speech_recognition) and anomaly detection in network traffic or IDSs (intrusion detection systems).

LSTM maintains a cell state as well as carry for ensuring that the signal is not lost as the sequence is processed

LSTM has 3 gates & weight vectors:

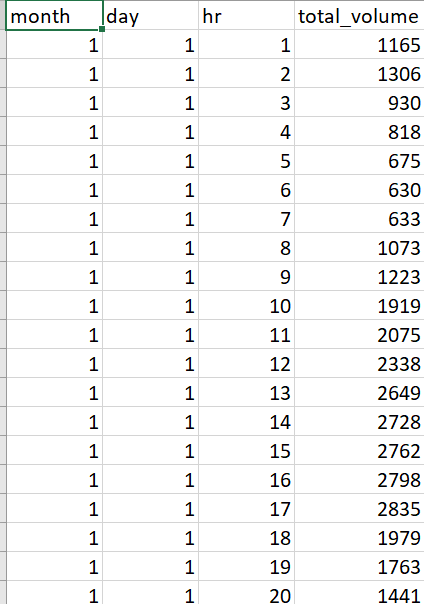
* Forget gate: discards irrelevant information
* Input gate: handles the current input
* Output gate: produces predictions at each time step



**Objective**

The objective of this project is to build a traffic forecaster that can predict the traffic flow in a smart city accurately and in real-time using deep learning model- LSTM and Simple RNN

**Dataset**



**Methodology and techniques used**

* First, we will import all the functions and classes we have to use- sklearn, pandas, NumPy and matplotlib and Keras deep learning library
* We will use LSTM model to make the time-series predictions
* Since, LSTM model is very sensitive to the scale of the input data, it is a standard practice to standardize the data to the range of 0 to 1. sklearn. pre-processing module has MinMaxScaler class that helps us standardize the data easily.
* Now, we will divide the dataset into training and testing data
* LSTM network is designed and trained to give predictions
* The network has a visible layer with 1 input, a hidden layer with 1 LSTM blocks or neurons, and an output layer that makes a single value prediction. The default sigmoid activation function is used for the LSTM blocks. The network is trained for 100 epochs and a batch size of 1 is used. To optimize, we minimize mean of squared error loss function. The error is nothing but the difference between actual traffic volume and predicted traffic volume.
* Now we have our model is fit and trained with the training data. We will now predict traffic value on the test data. If the actual value in the test data is same as the predicted value, we can say, our model is a good predictor. If the model is tested ok, it is ready to be used in the real-life scenarios. Since we scaled the time series, we need to revert to original series. The inverse transform function would do this task

**Python modules used:**

* Scikit-learn (sklearn)
  + Mean\_squared\_error
  + MinMaxScaler
* Keras library:
  + Sequential
  + Dense
  + LSTM, Simple RNN
  + SGD
* Matplotlib
* Pandas
* NumPy

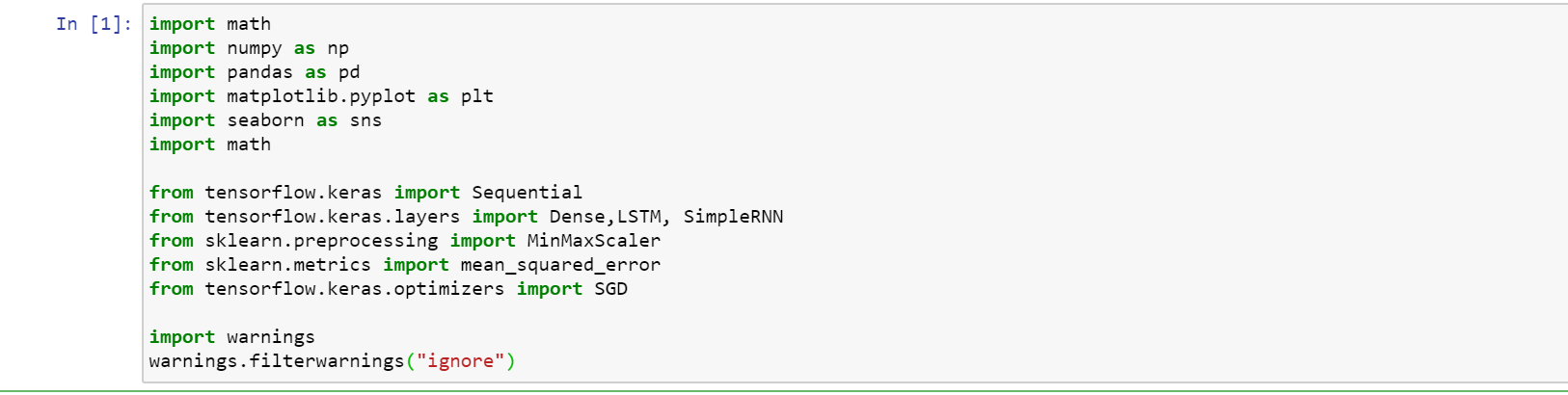
**LSTM model**

**Long short-term memory** (**LSTM**) is an artificial [recurrent neural network](https://en.wikipedia.org/wiki/Recurrent_neural_network) (RNN) architectureused in the field of [deep learning](https://en.wikipedia.org/wiki/Deep_learning). Unlike standard [feedforward neural networks](https://en.wikipedia.org/wiki/Feedforward_neural_network), LSTM has feedback connections.

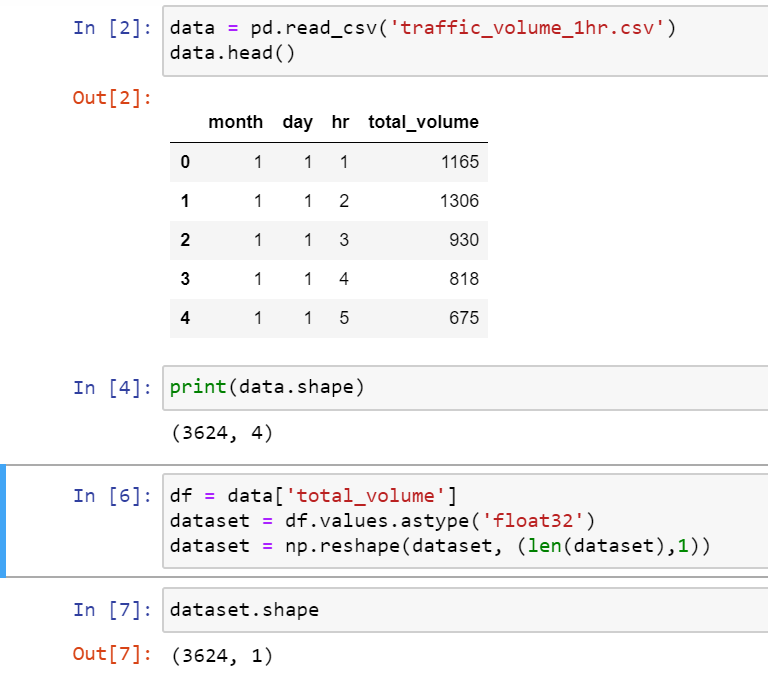
It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected [handwriting recognition](https://en.wikipedia.org/wiki/Handwriting_recognition), [speech recognition](https://en.wikipedia.org/wiki/Speech_recognition) and anomaly detection in network traffic or IDSs (intrusion detection systems).

**Python Code explanation**

* Import the following libraries: **‘math’** for mathematical operations, **‘NumPy’** for numeric operations, **‘pandas’** to load the dataset, **‘matplotlib’** and **‘seaborn’** for visualizations. For building a Keras model, import **‘LSTM’** and keep layers as **‘Dense’**. The sci-kit learn library ‘**MinMaxScaler**’ is for pre-processing of data.

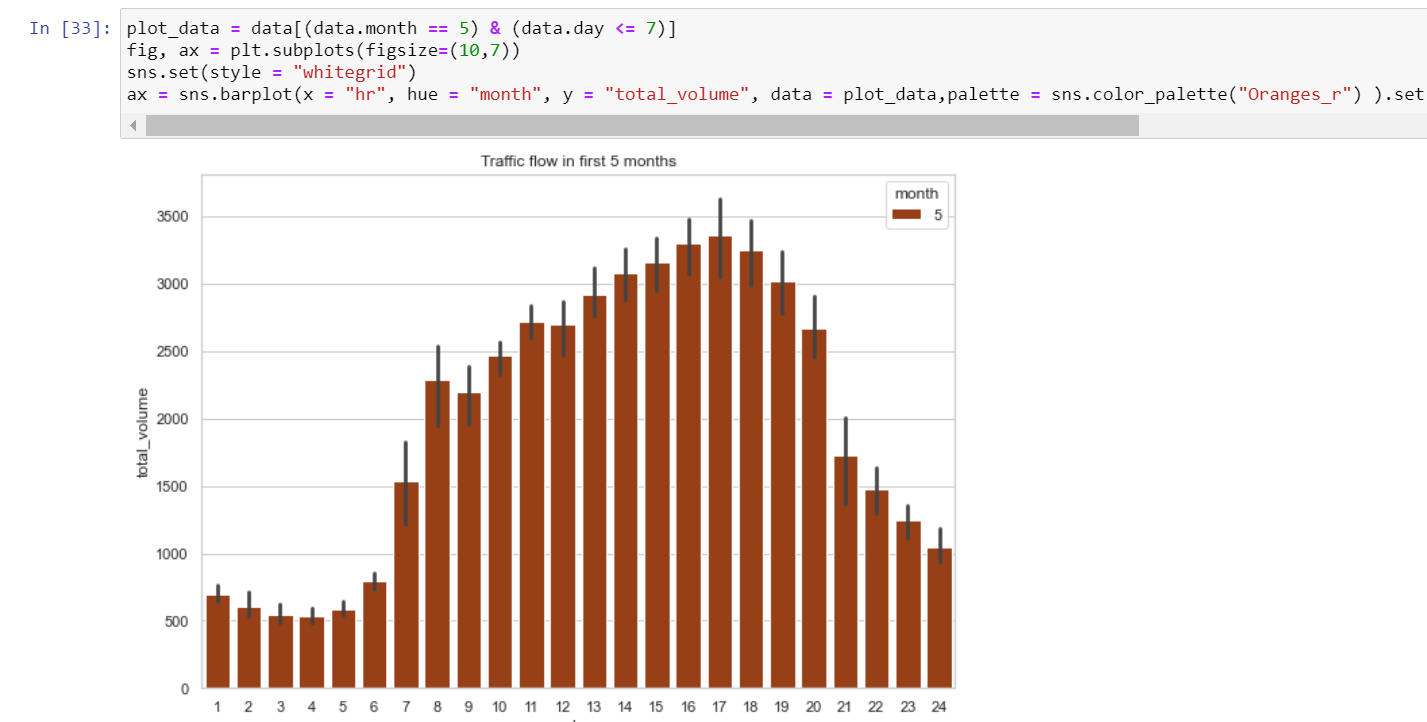


* Next, we will load dataset and change its type to float and reshape it

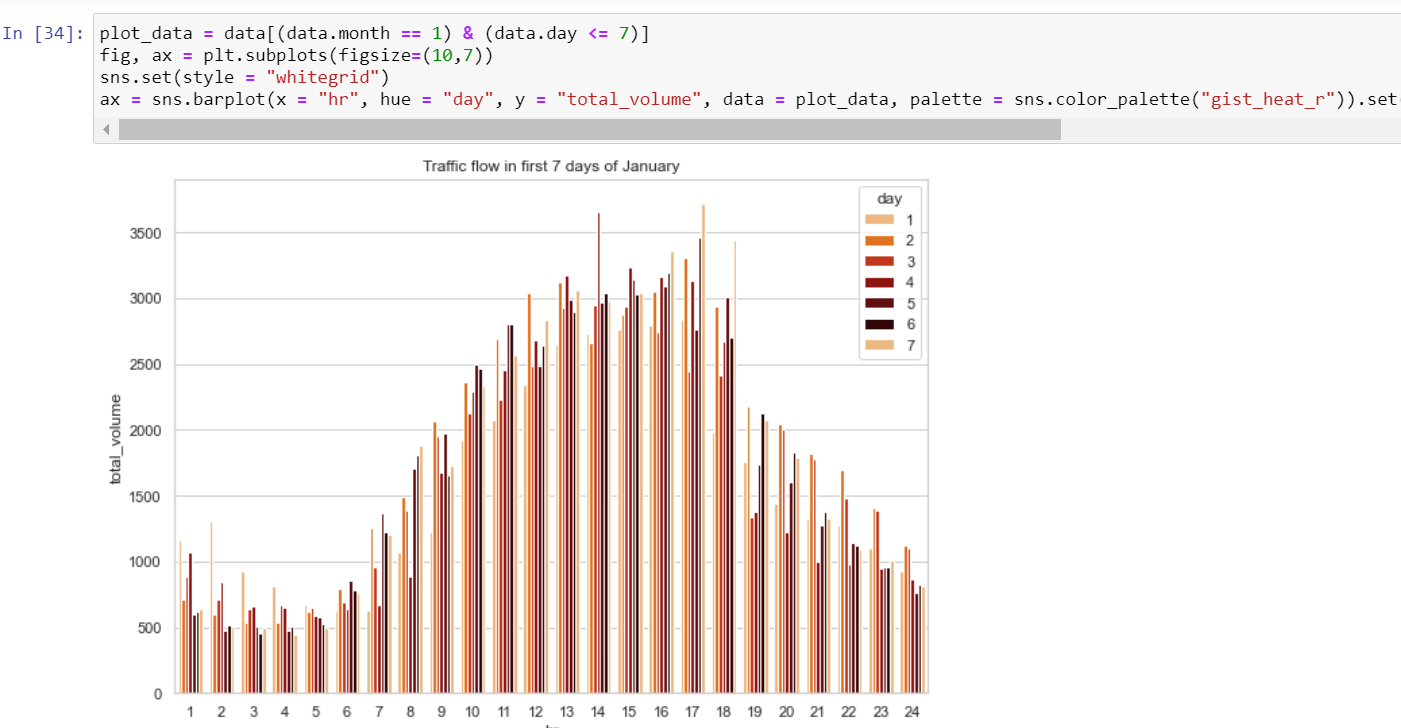


* Now, we will visualize our data with help of bar plot, line graph and pie chart

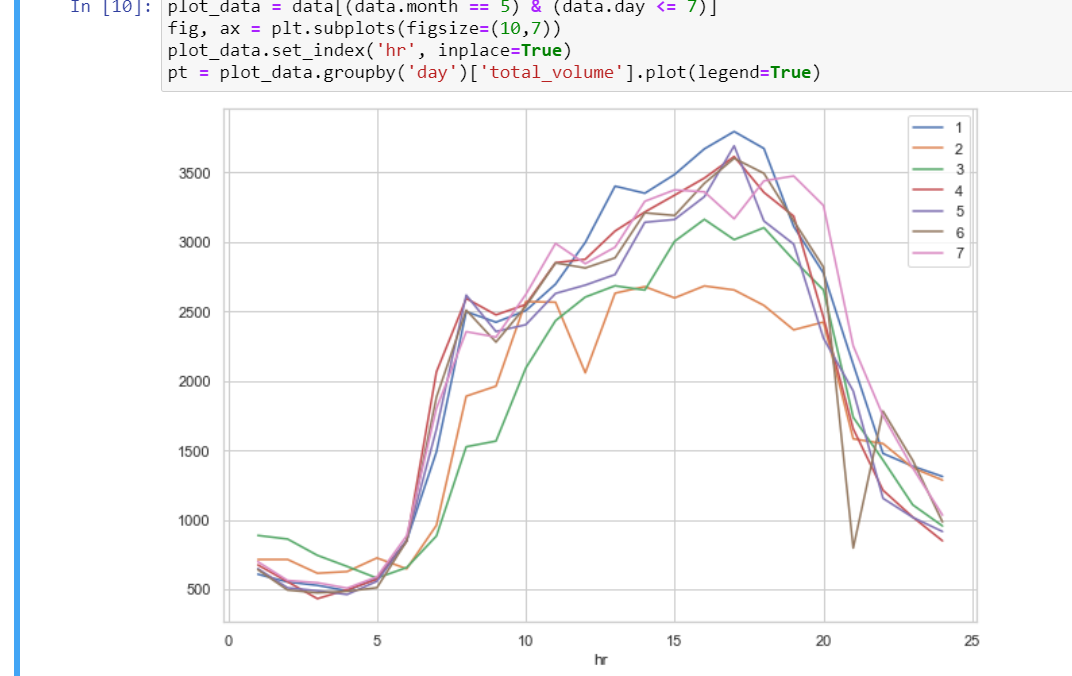
Bar plot to visualize the traffic flow in first 5 months



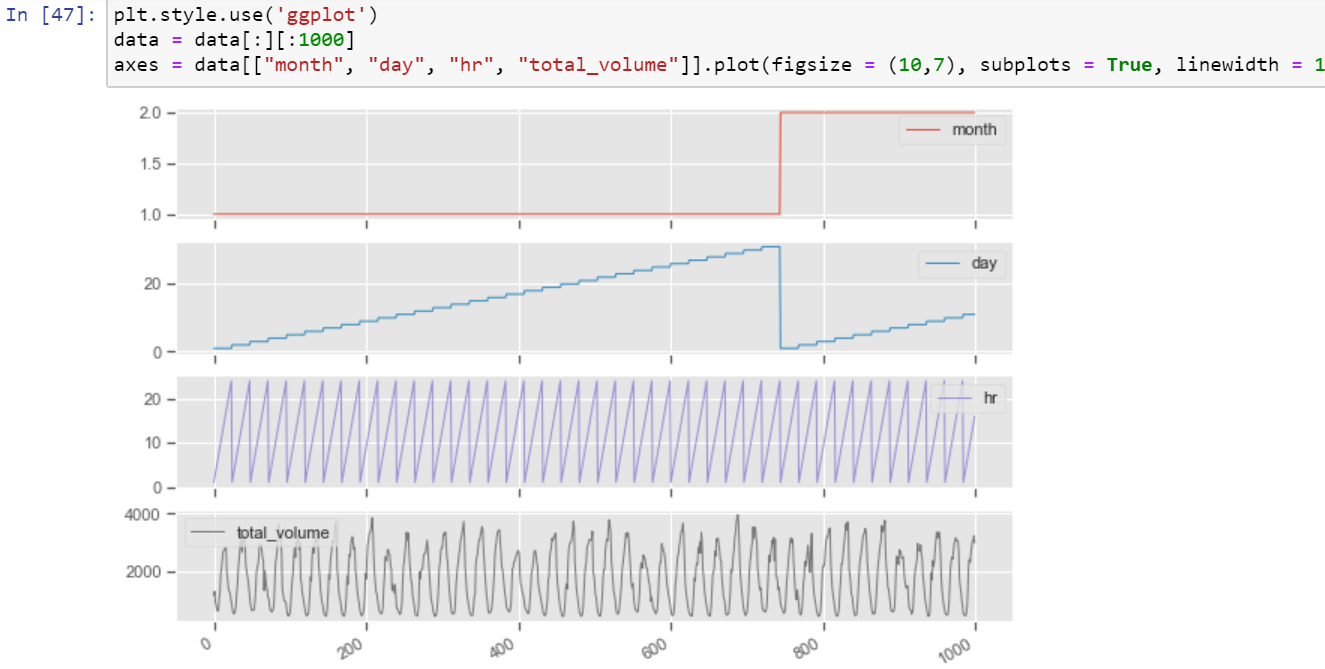
Bar plot to visualize the traffic flow in first 7 days of month of January



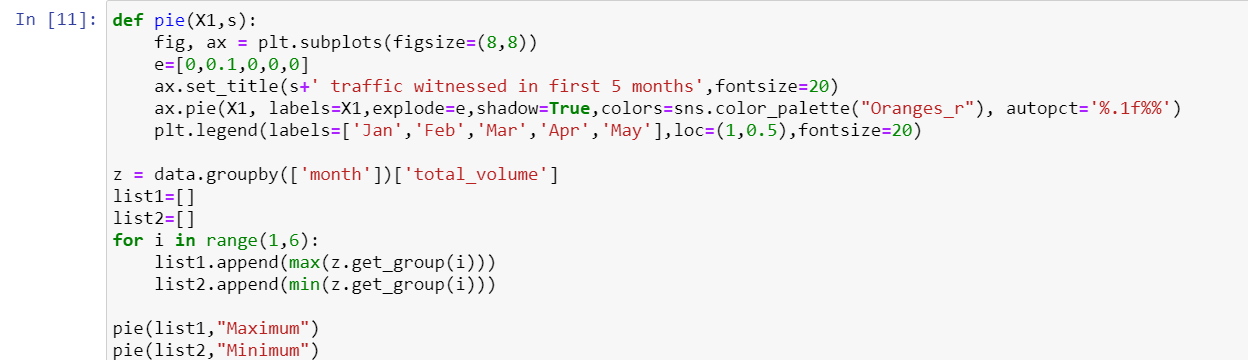
Line graph to visualize the hourly traffic volumes for a particular week

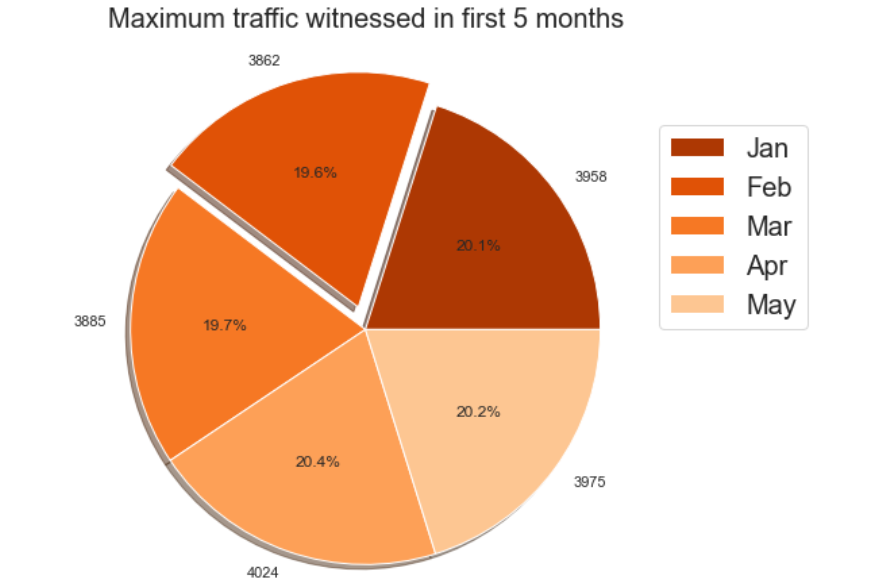
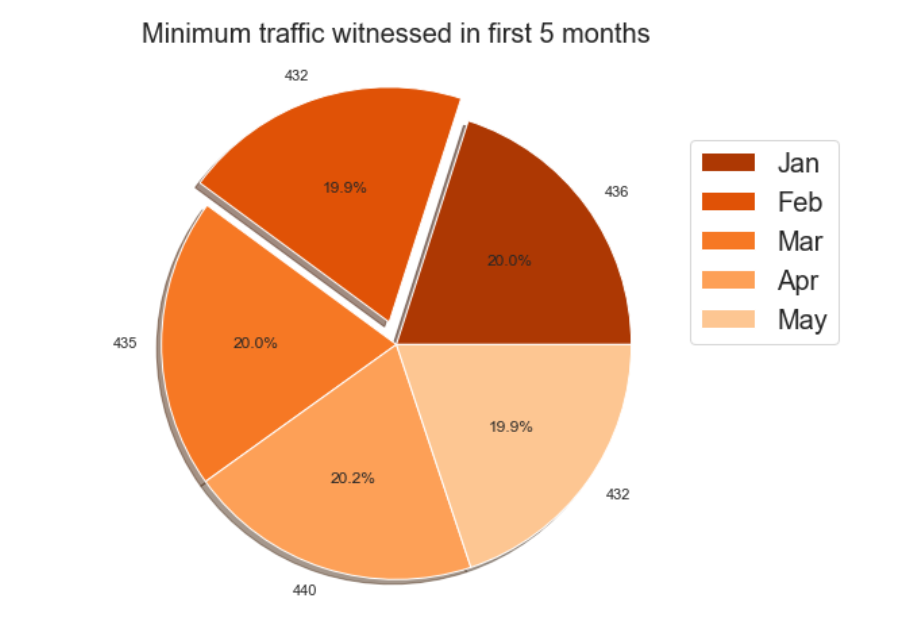


Line graph of month, day, hr and total volume



Pie chart to visualize maximum and minimum traffic witnessed in first 5 months



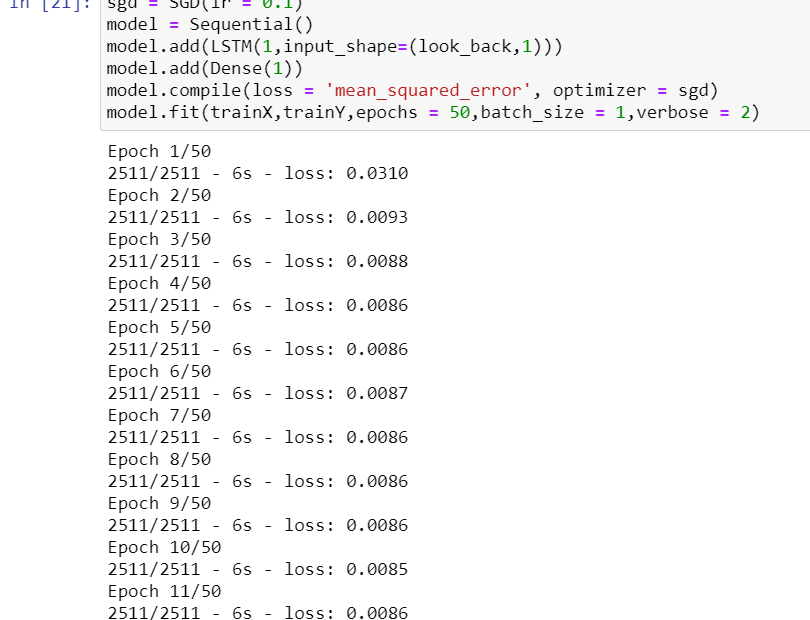
 

* From the above fig we can see that maximum traffic was seen in the month of April & January and minimum traffic was seen in the month of February & May

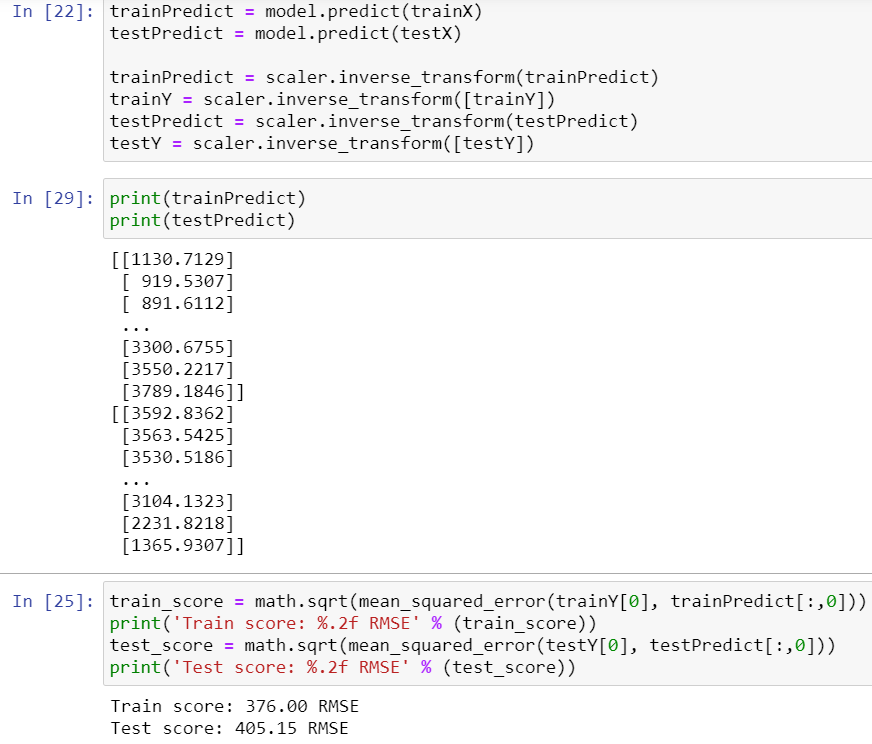
Normalization of dataset & splitting dataset into training and testing set



Next, we design our deep learning model and train it and fit trainX, trainY to it



Prediction of traffic value on test data and calculation Mean\_squared\_error



Plot actual values vs Predicted values

