**Introduction to the Problem :**

**Data Source** : It was a sample dataset from ERP & CRM. Then we merged and transformed the data using business data integration logic. The idea was to come at daily level dataset for the booking which was captured in Sales Amount Column.

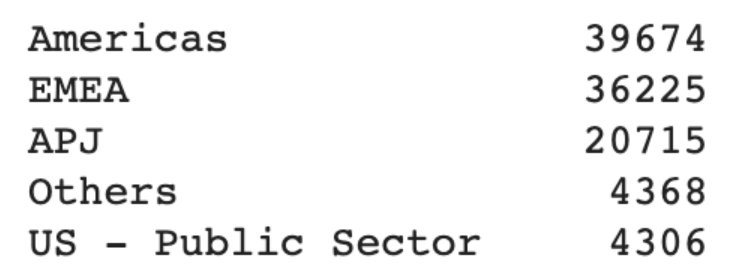
Once we have developed the dataset, it was ranging from 2016 to 2018. It in the Beginning there were 23 columns and 387032 rows. Below is summary/list of the important columns

**Table from other docx**

We have many categorical variables as well for e.g. geography which is continent based, and nation based. Forecasting at the aggregated level( for all the regions) would not make sense. Because, forecasting at aggregated would give less context to the sales amount column, we wanted to be for specific to our forecasting problem

So, Idea is to use LSTM deep Learning model to forecast at a region-based level. There we total 5 regions

Table of value counts region wise:



As we can notice from the table above data points are significantly higher in America and EMEA. So, we would be building LSTM deep Learning model for these two regions.

**Data Exploration and preprocessing**

**Validating data :** Once we have loaded the dataset, We did some quality checks once, Checking the nulls, Data types, preprocessing the data at the level we would be forecasting, changing the datatypes accordingly, subsetting the data into multiple data sets with respect to region  as well as dropping the unnecessary columns.  We are forecasting the sales amount for each region.

**Transforming data** : Where are using Robust Scaler as a transformer to preprocess the data. This Scaler removes the median and scales the data according to the quantile range (defaults to IQR: Interquartile Range). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile).

We are also performing EDA to make sure; we understand the story/ distribution behind each column. Below are the plots with explanations.

**Figure 1** : It show the distribution of sales amount and gives us hint of the fact that it is a right skewed distribution. Upon checking further, we got to know the mean is & standard deviation is

**Figure 2**: It show the sum of sales amount for each region. As we can notice is the highest for the America and second highest for EMEA region. And based upon this graph only we went to forecast for these two regions.

**Figure 3** : Since we noticed there are 3 unique values of Year column, we looked at the distribution Year wise. And it seems like data is pretty smotthly, uniformly distributed.

**Figure 4** : It explains what percentage of sales sum corresponds to each region, as noted 77 percent of the data is captured by America and EMEA.

**Figure 5** : Similarly, we wanted to see how data is distributed quarter wise, and as expected it is uniformly distributed.

**Figure 6** and **Figure 7** are for America Region and **Figure 8 & 9** for EMEA Region.

The distribution monthly wise is looking uniform for both the regions and in the distribution of sales amount by date we are seeing some outliers, but the distribution is same for both the regions.

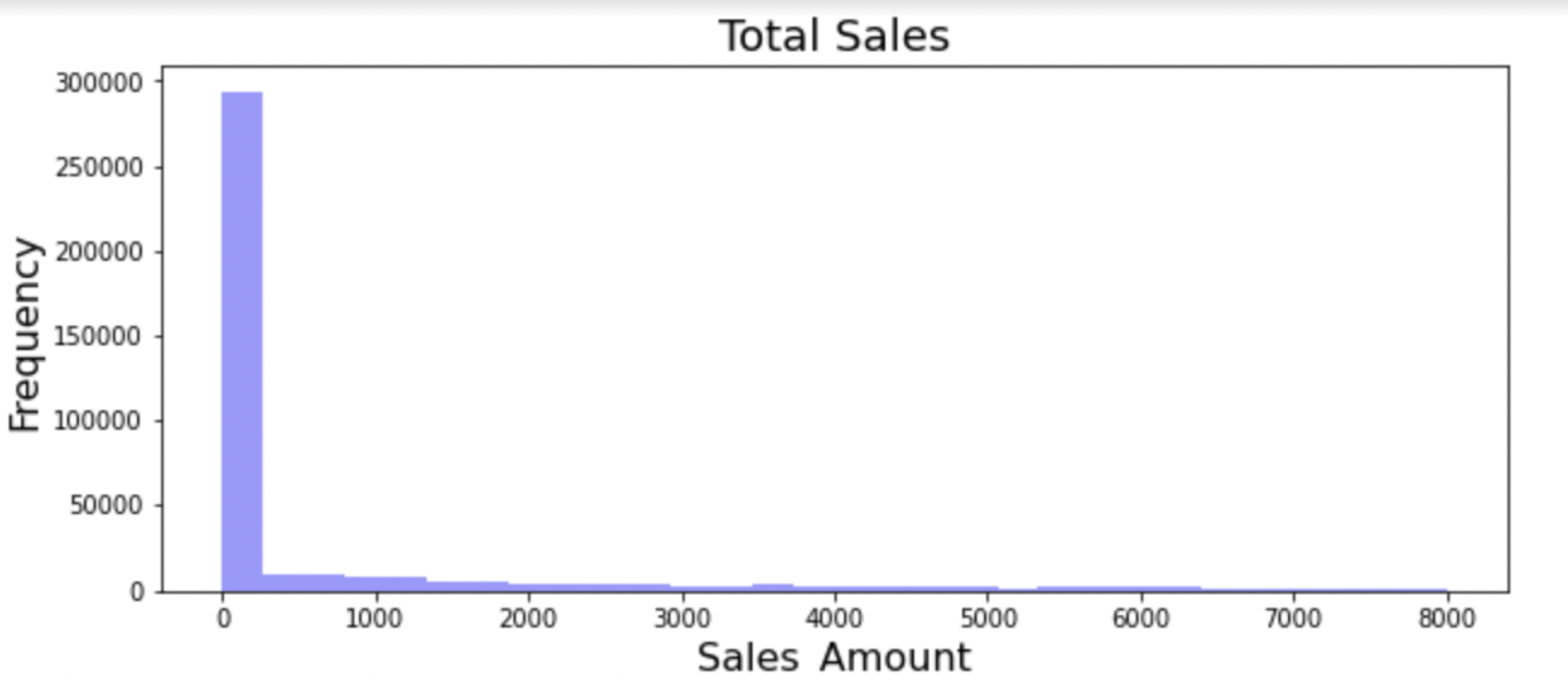
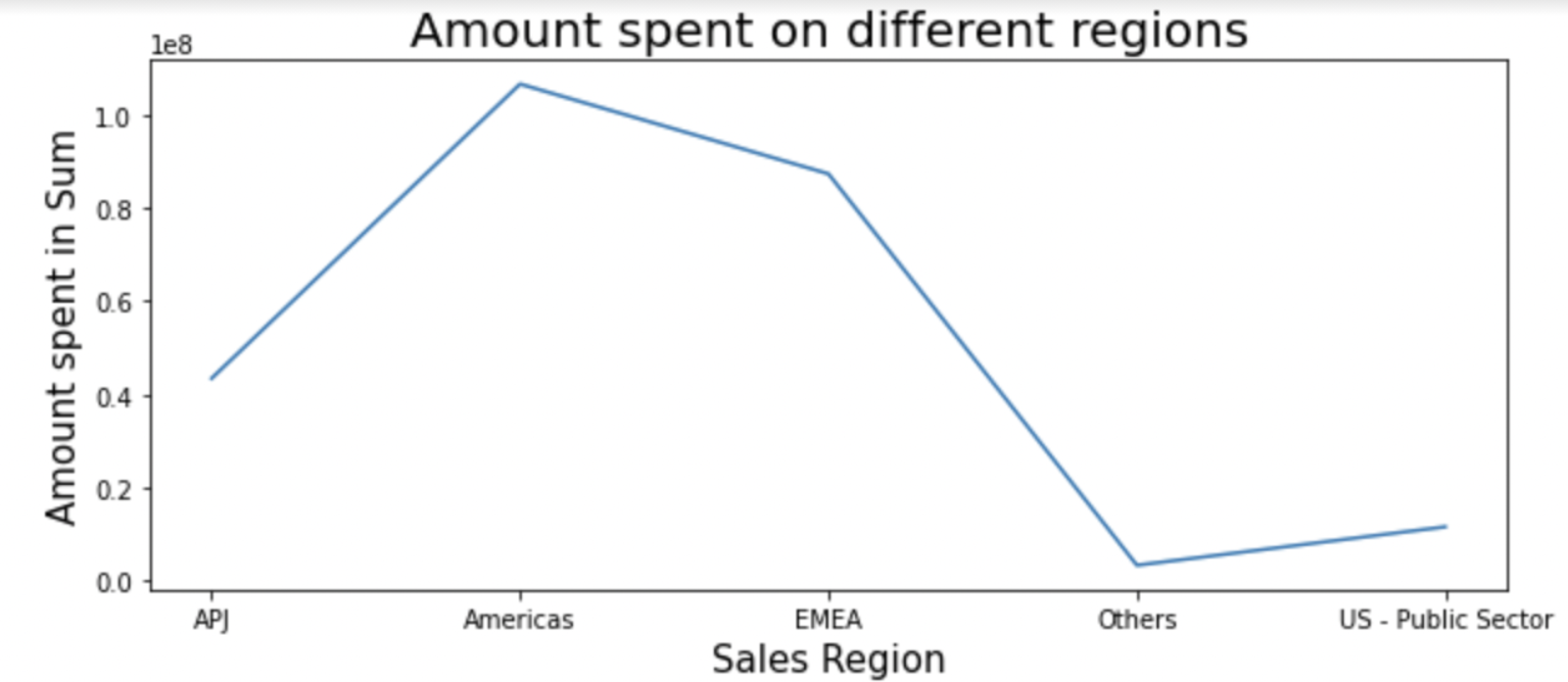
 

FIGURE 2.

FIGURE 1.

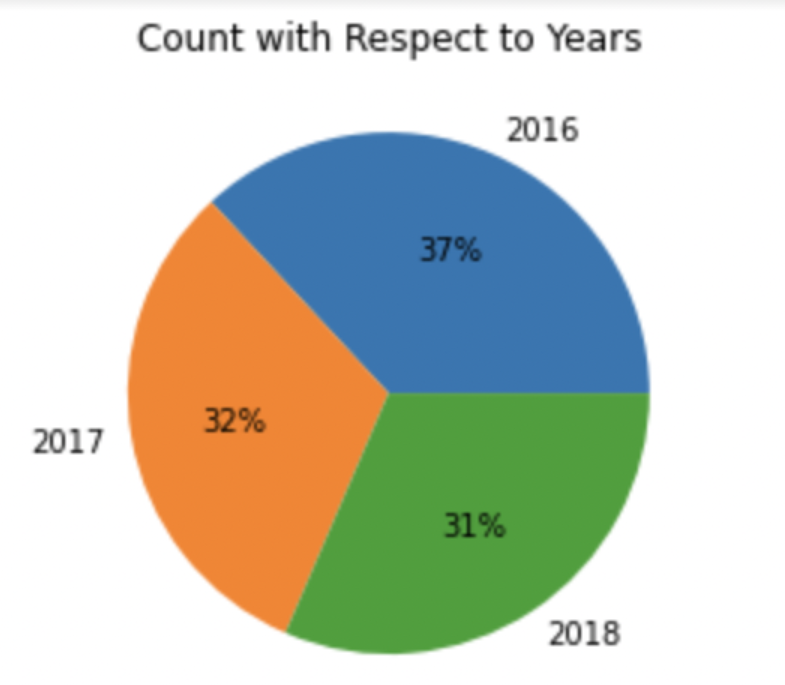
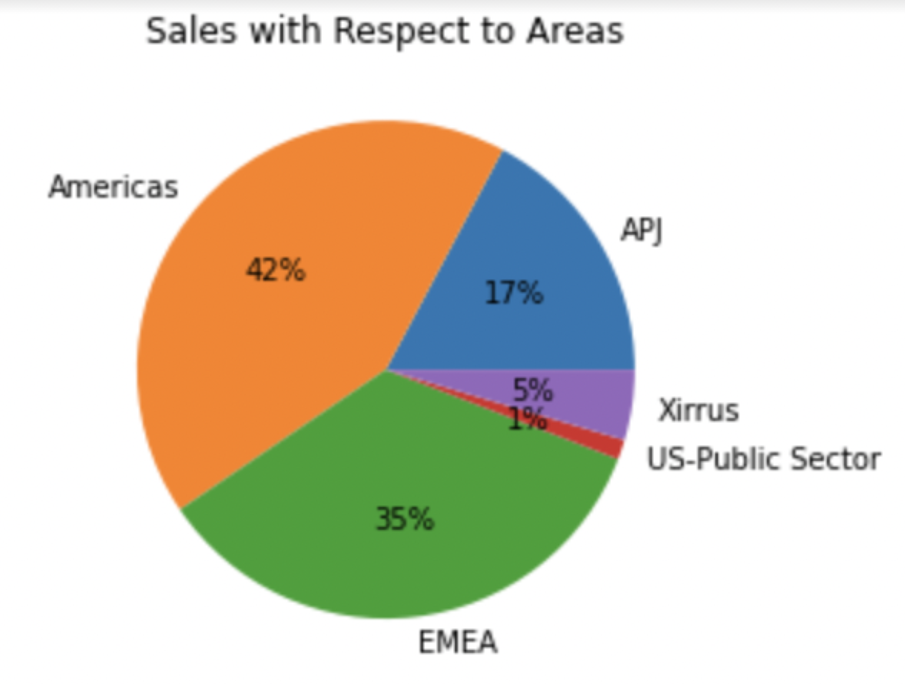
 

FIGURE 4.

FIGURE 3.

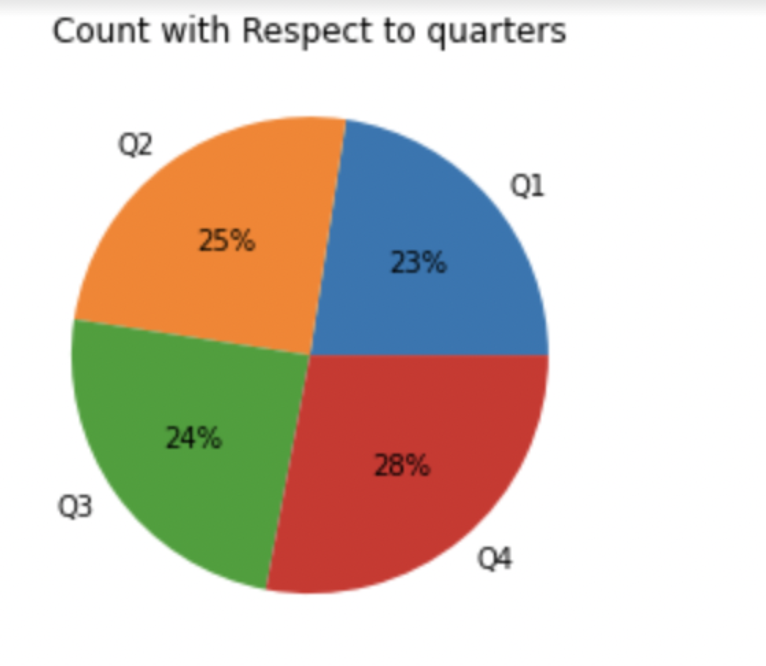


FIGURE 5

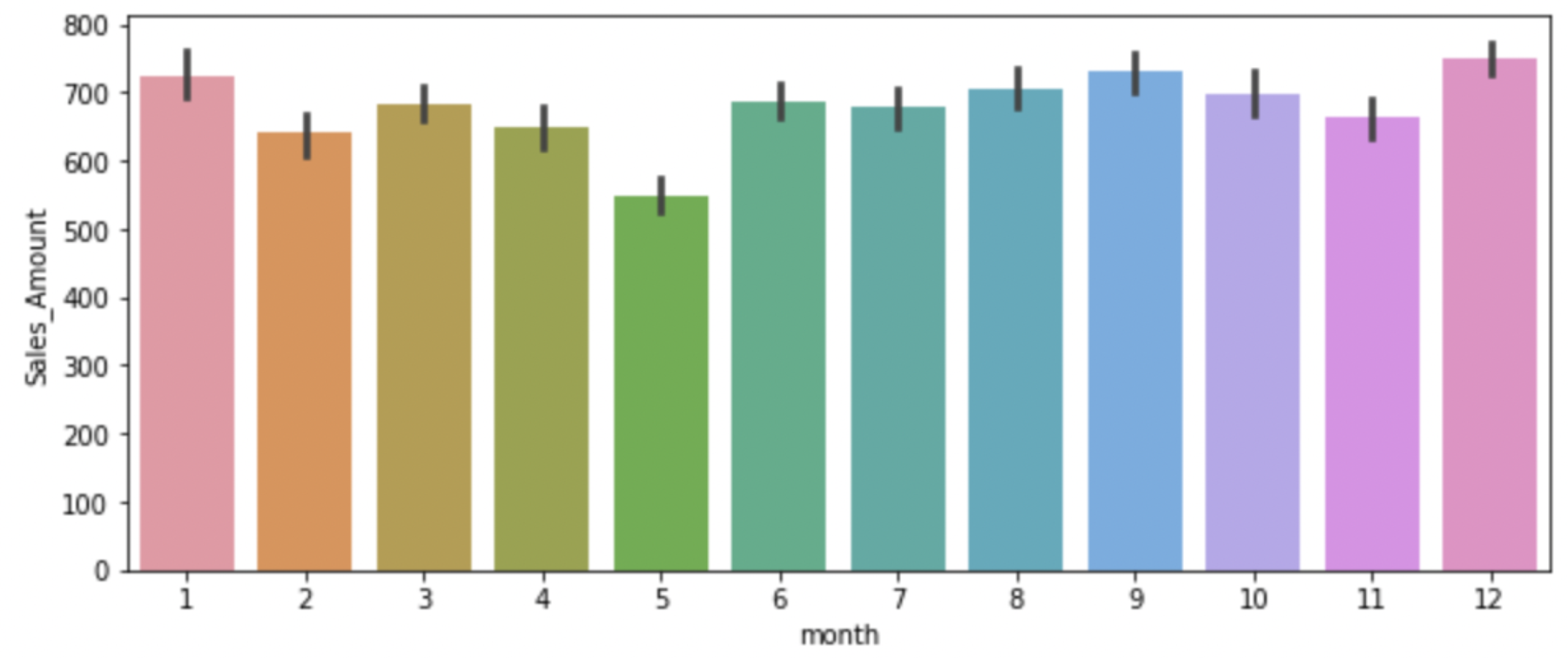
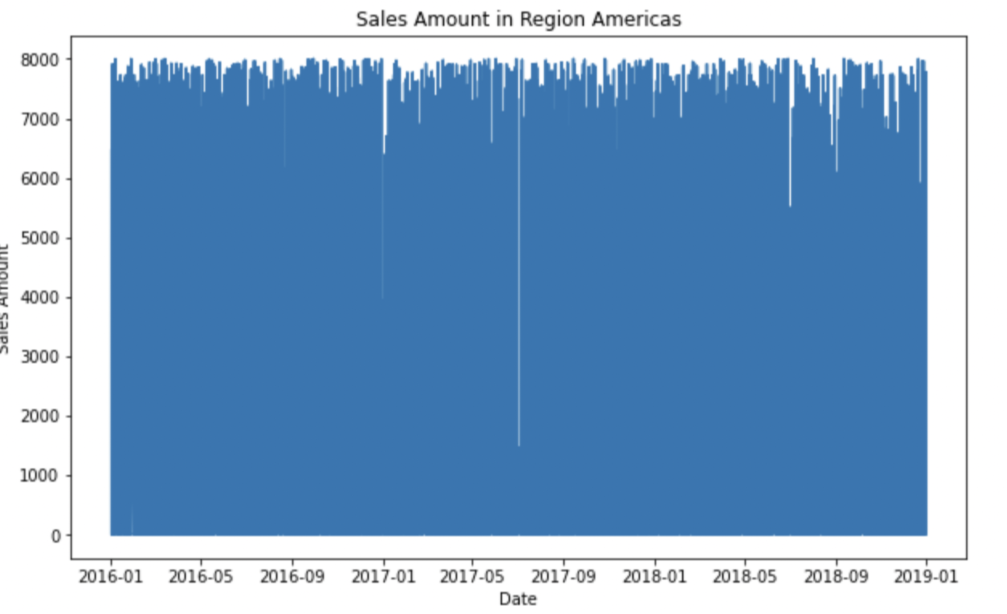
 

FIGURE 7

FIGURE 6

For

EMEA Below :

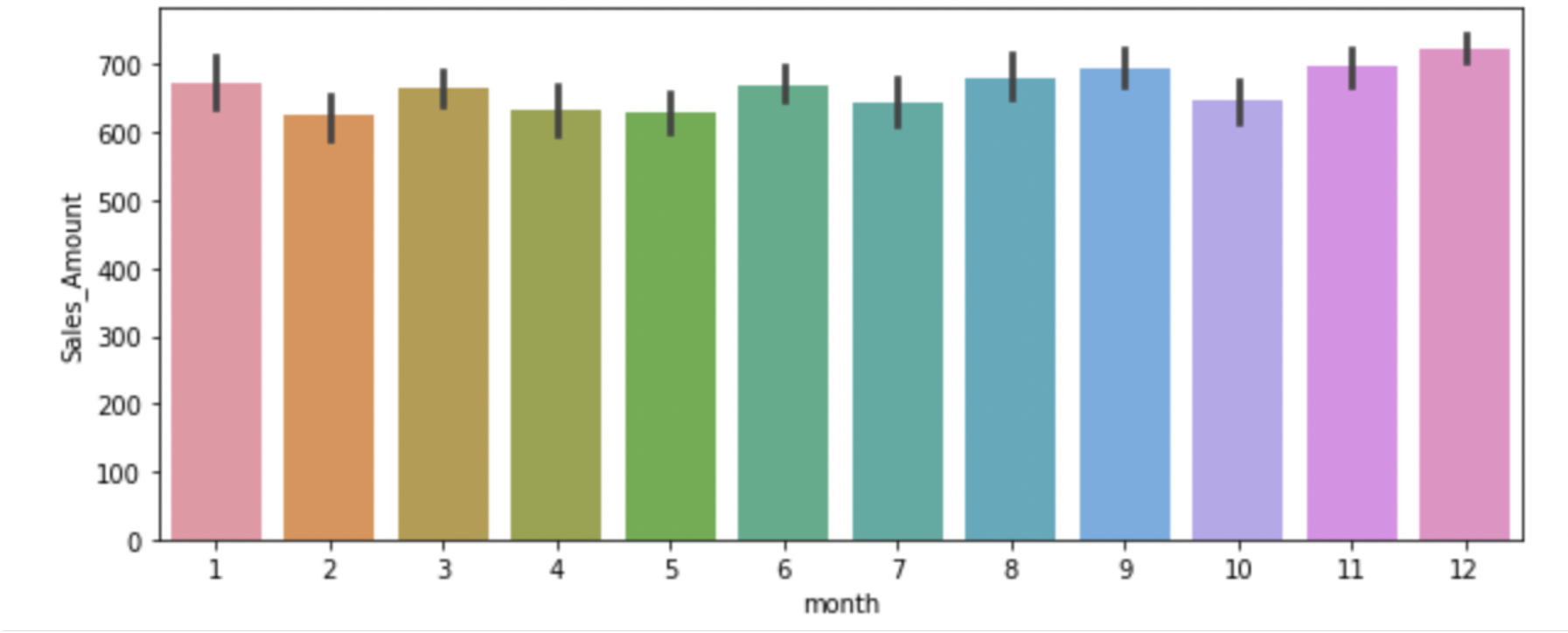
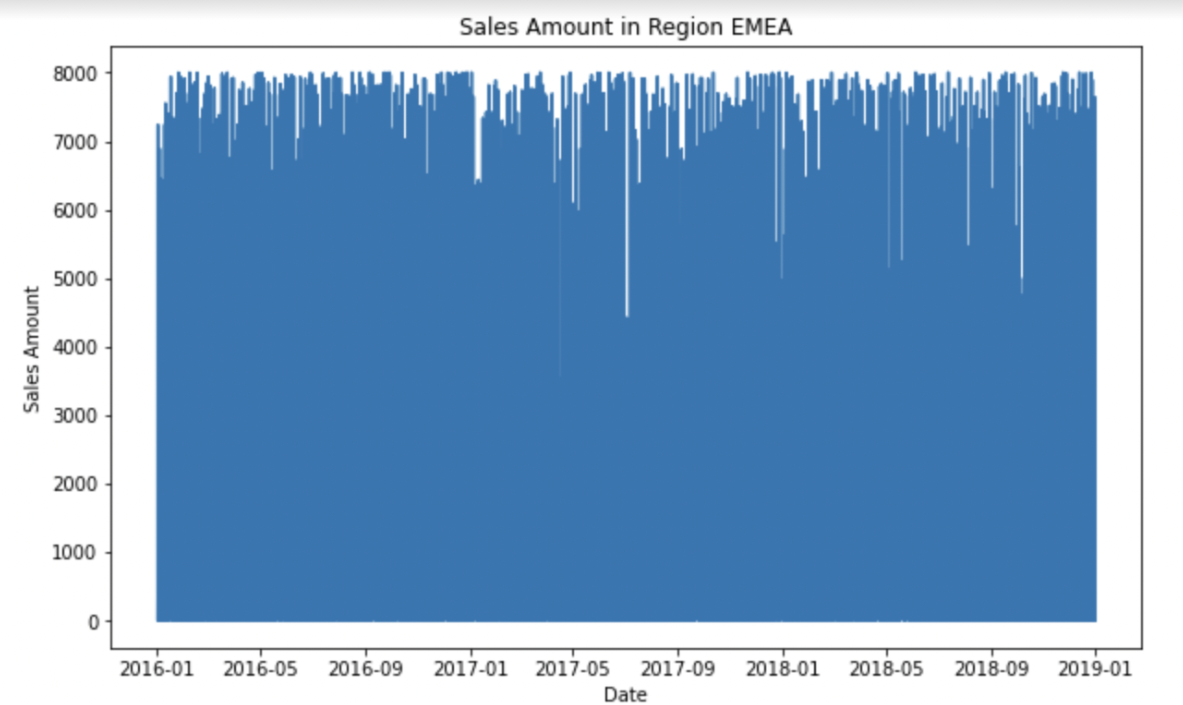
 

FIGURE 9

FIGURE 8

**Data Analysis & Model Architecture**

We are using LSTM as neural network architecture for this time series problem, as we know LSTM stands for Long Short-Term Memory. Generally, in time series all relevant information about the next event is conveyed by a few recent events contained within a small-time window.

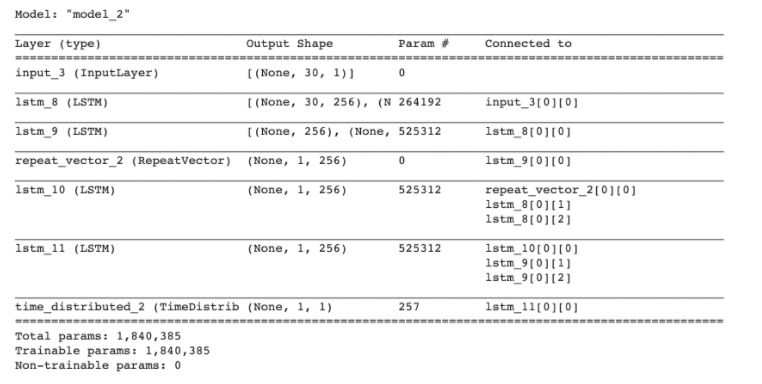
There are various advantages of LSTM for time series problems, I am listing few below :

-          LSTMs have the promise of being able to learn the context required to make predictions in time series forecasting problems, rather than having this context pre-specified and fixed.

-

In our specific problem we are using 2-layer LSTM, with 256 neurons in each layer, with a dropout layer as well. Below is the screenshot of the final architecture.

**Screenshot of the model architecture** :



Given the quality of our data & based upon the reading Blogs and literature, I learnt that 2 layers LSTM model with enough epochs we can learn the underlying the complexity of the data.

**Screenshot of the complier** :



**Evaluating, Tuning and Improving Model**

We Divided our data into three Parts as mentioned/ recommended Train, validation , testing.

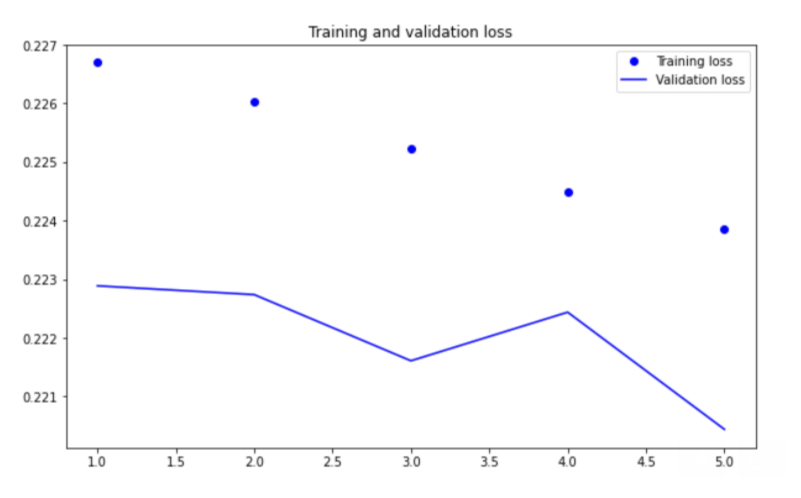
We took 20 percent of the data for testing, and from the training data we took 30 percent as a validation dataset.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cases** | **Epochs** | **Learning Rate** | **Early-Stopping True/False** | **Mape** |
| **Case 1** | 5 | 0.001 | False | 182 |
| **Case 2** | 50 | 0.01 | True(Stopped at 8) | 199.9 |
| **Case 3** | 250 | 0.1 | False | 304.7 |
| **Case 4** | 250 | 0.01 | False | 302 |

**Case 1 : Performance**

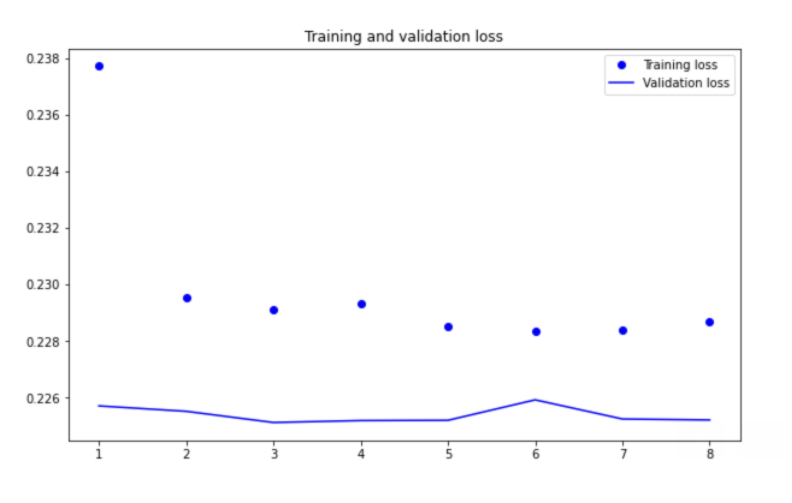
Once we have done enough Literature Review and decided on the basic Architecture as explained above, we are playing majorly with Epochs and Learning rate in our experiments.

In the first case we started with 5 Epochs, just to see the maximum loss is we are having given our model. As we can see in the Graph Below it is 0.227 for train data and 0.223 for validation data



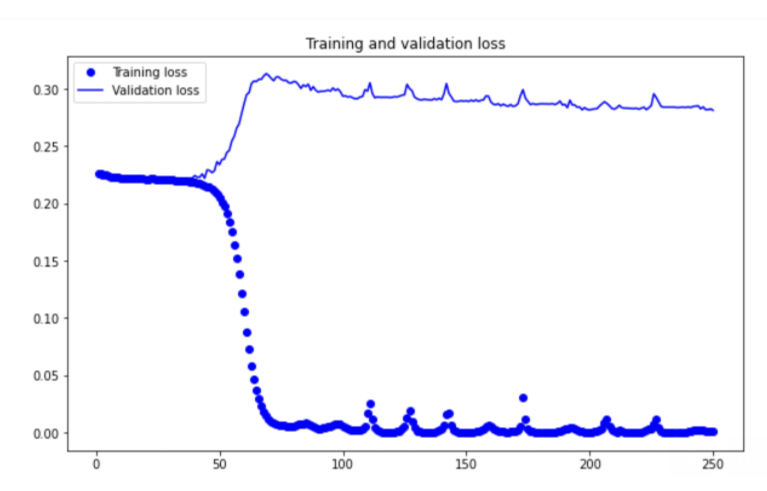
**Case 2 :**

In here, we took 50 epochs and early stopping as true, and we kept patience parameter in early stopping as 5. And we learnt that model stopped training at 8 epochs. And the training loss came down to 0.230 and validation loss remained almost same. To make sure model is not stuck in local minima, we would run many epochs to check we are not underfitting.

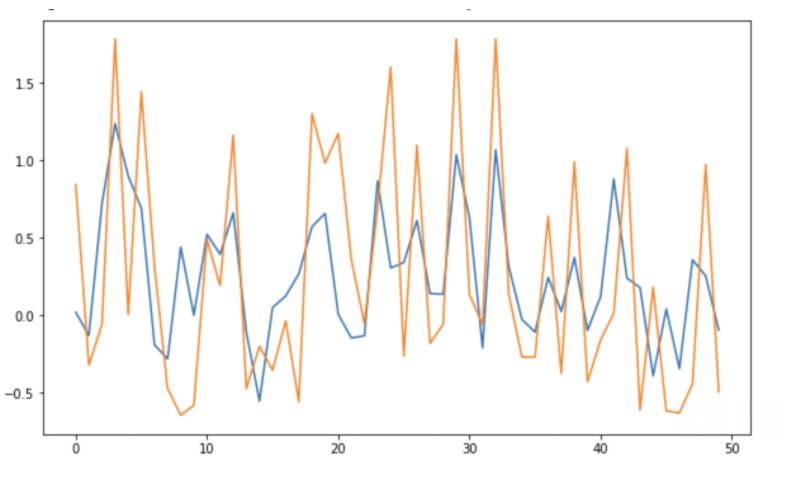


**Case 3 : Prediction + Prediction**

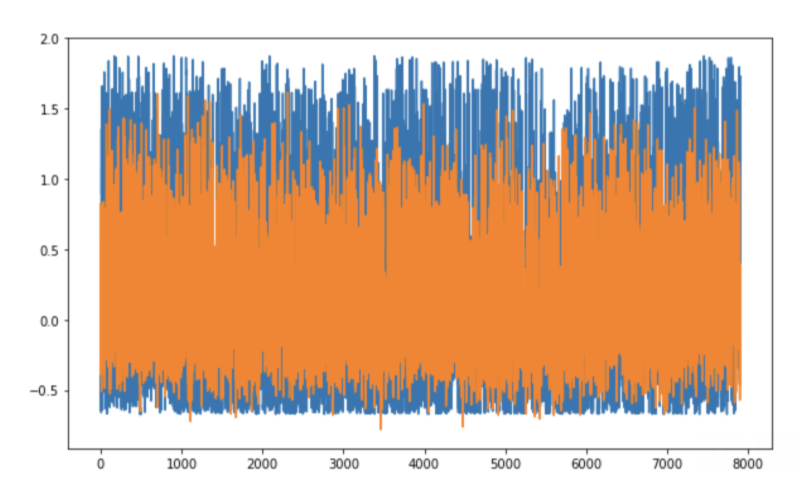
Following the case from above we ran for 250 epochs and learnt that our model was stuck in local minima. To get out of local minima we increased the epochs, and from the graph below we can see that training loss was decreasing slowly but validation loss decreased significantly. We are taking this model as our final model.



This graph below is showing for some sample how the predicted values and actual values are looking on the test dataset.



This graph below is showing for the entire dataset how the predicted values and actual values are looking on the test dataset. As we can see variations are captured pretty well by predicted values.



**Case 4 :**

In this case we changed to learning rate to 0.01 with 250 epochs, and we observed that validation loss didn’t deep that significantly. This experiment proves the point with a different learning rate we can get out of local minima case.



**Conclusion**

Since, this problem was to be solved by Deep Learning and went ahead with LSTM. Otherwise I was very curious to use XGboost as well.

With through reading and defining architecture we tuned our model to remove underfitting and giving plausible predictions and can be seen from the graph in case three. I can think of some improvement areas as well for e.g. having more data, Deeper models  etc.