WEEK -3

This week I obtained the historical data without downloading the CSV file from Yahoo finance but did it using the API. The reason I used the API to get historical data is because we get reliable data from the time you would want until now and everytime I run the code the data would automatically adjust to the end-point to the present day .So I PIP installed Yfinance and imported the API. I then set the datetime to be a reference to the class and then set it to the module . I then give the start datetime to 2015 because cryptocurrency started gaining attention since 2014 and then give the endpoint to the present day using the datetime.now().For this I am using ‘BTC-USD’ bitcoin as the crypto of choice.I then created a dataframe that contains all the data from the API and I downloaded it .To check the data from the top I gave df.head().I converted the data frame into a csv file called ‘coin.csv’.Now I wanted to check the data frame types whether the data was float ,objects etc. Now I can start with training the dataset but before that I need to give the training attributes and the test attributes . So I gave data_training as data of the date <2020-01-01 as a test case to see if the training can take place from 2014 and similarly for the test data also .I now give the training data by adjusting the columns. Since for training the dataset I would only require the “open,high,low,close,volume” I need to drop the columns that are not required like “date and adj close” with the axis 1 to drop the coloumn . The next thing I did was normalization .We do that to differentiate the values or the numeric values in the columns in the dataset to a common scale .We do that without changing the differences in the range of values . For this I used MinMaxscaler. It basically scales the features here to a given range.Now I give X_train for the attributes and Y_train for the labels.X contains the samples and the features and Y contains the shape for the samples.Now I am setting the tuple shape of the training_data array . Next, for the given range for the training _data shape , I created 60 timestamps for each given value so the first row will have 0-59 and the second row will have 1-60 etc. Now I will append the X_train.Contains the following values after the 60 timestamps .Example row 1 =last value of row 2 and row2=last value of row 3 , this can be used to predict the future values.And then for the Y_train I append because it should be used to convert the numpy array to be accepted into the rnn(lstm).Now I am feeding the training data into the lstm model .My model is made out of the sequential input layer and has 3 lstm layers and a dense layer .Now we would start with the model sequential layer basically adding the first lstm layer and some dropout regularization,Adding a second LSTM layer and some Dropout regularization,Adding a third LSTM layer and some Dropout regularization,Adding a fourth LSTM layer and some Dropout regularization,Adding the output layer,Compiling the RNN,Fitting the rnn model .I used an Adam optimizer , the optimizer is a combination of two other optimizers and they are ADgrad and RMSprop.They help in the learning rate ,that is they use different lr because every parameter uses different lr and the latter one helps with diminishing lr.I then regularize the model because we need to prevent overfitting the data points because of the weights .Dropouts that we use is also used to prevent overfitting but used in nn because of neurons they make them robust and accurate .The last one is the output generation and in this the output generates from the nn is compared to the target output.For this model in the optimization I gave the loss function to be the mse or the mean square error .After that i did model.fit() using the batch size as 50 and the validation split as 0.1. Validation split is basically to predict the how well the model would fit by considering a small amount of training data to the hypothetical test without the testing the data , it will help in fine tuning the model after every epoch , here I used 20 epochs and 10%of the data in the validation set and 90% of the data for the later test set.And then last but no the least plotted the validation loss and the training loss.