

# ROE, ROA, REVENUE, BASIC AND DILUTED EPS FORECASTING USING LSTM

(Time Series Forecasting)

## Description:

I have a data for past 10 years of 244 companies, data consist of ROE, ROA, REVENUE and w.r.t per year and same as BASIC and DILUTED EPS but both EPS are from 2015 to 2019. As we can see the nature of data is similar to time series forecasting for this purpose, I will use LSTM models for forecasting ROE, ROA, REVENUE and BASIC, DILUTED EPS. After forecasting I am classifying the companies in 3 categories Good, Worst and Neutral for example if a company ROE increase by 5% then its current ROE then it will be considered as Good if it drops then 5% then it will consider as Worst otherwise it lies in the neutral category. For each company I made two models in one model I use univariate approach and in other multivariate. I train the model using 2010 to 2019 data and the predict 2019 to 2025.

## Steps:

1. Separate data for individual company.
2. Split data into train and test.
3. Transform data into required format, which can be given as input to our model for training and testing.
4. Build and save model.
5. Load and Predict model.
6. Classify model.
7. Save results in excel file.
8. Visualize Results.

## Separate for individual company:

First let's look how was the original data.

C	D	E	F	G	H	I	J	K	L	M	N	
Company	ROA 2019	ROA 2018	ROA 2017	ROA 2016	ROA 2015	ROA 2014	ROA 2013	ROA 2012	ROA 2011	ROA 2010	REVENUE 2019	REVENUE 2010
Alco inc	9.06	3.08	-2.26	1.54	2.87	3.69	9.88	9.99	3.94	-0.33	122,251	81
Cal Maine foods inc	4.69	10.95	-7.19	28.43	17.36	13.46	6.76	12.36	9.49	10.74	1,361,188	1,50
S and W Seed Co	-6.36	-3.43	-10.10	0.29	-2.59	0.42	-2.94	1.70	-5.35	2.47	109,723	64
America's car mart inc.	9.66	8.01	4.75	2.84	7.36	5.81	8.97	10.60	10.19	10.67	669,122	61
Autozone	16.34	14.31	13.83	14.43	14.32	14.23	14.75	14.85	14.46	13.25	11,863,743	11,2
Marinemax inc	4.59	6.14	3.68	4.13	10.33	2.80	3.93	0.30	-3.17	0.74	1,237,153	1,1
Advaxis inc	-36.71	n.s.	-99.78	-43.51	-39.32	-70.69	-87.09	n.s.	n.s.	n.s.	20,884	6
Alexion Pharmaceuticals Inc.	12.42	1.74	4.03	4.35	3.80	20.76	15.86	15.21	16.47	14.72	4,991,100	4,1
Applied DNA Sciences inc	n.s.	n.s.	n.s.	-78.22	-76.39	-74.55	-72.71	-70.87	-69.04	-67.20	5,389	3
Avid Bioservices inc	-6.81	-21.47	1.18	3.30	-51.67	-39.06	-66.09	n.s.	-98.23	-49.41	53,603	53
Biogen Inc	21.62	17.52	10.74	16.19	18.19	20.50	15.70	13.62	13.64	12.40	14,377,900	13,4
Bio-technie corporation	5.94	7.91	7.19	13.06	14.50	18.71	20.65	22.55	26.71	30.15	714,006	64
Cabot corp	5.13	-3.51	7.43	4.84	-10.86	4.87	3.61	8.82	7.51	5.34	3,337,000	3,2
Cabot microelectronics corp	1.73	14.09	10.43	8.23	8.50	8.44	9.53	7.74	8.22	8.65	1,037,696	59
Cardinal ethanol LLe	-4.80	5.28	8.43	8.78	24.59	52.45	15.62	1.20	14.50	12.36	260,669	26
Clorox co	16.03	16.27	15.33	14.37	13.93	13.11	13.27	12.42	13.38	13.26	6,214,000	6,1
Estee lauder companies Inc.	13.57	8.82	10.80	12.09	13.24	15.30	14.27	13.00	11.17	8.96	14,863,000	13,6
Grante falls energy inc	-7.91	2.45	9.04	7.18	11.14	34.92	8.90	0.26	19.38	13.99	208,777	21
H.B. Fuller company	3.28	4.10	1.36	5.92	4.24	2.66	5.17	7.03	7.26	6.15	2,897,000	3,0
Innovation pharmaceuticals inc.	n.s.	n.s.	n.s.	n.s.	-91.80	-76.00	-60.19	-44.38	-28.58	-12.77	0	
Landec corp	0.08	6.14	2.95	-3.40	3.91	6.10	7.76	4.57	1.90	1.99	557,559	52
Lannett company inc.	-22.92	1.82	-0.04	2.54	29.47	16.66	7.94	2.77	-0.19	5.59	655,407	68

We need to convert this data into individual companies' data, like each company have its own file so we can use it to train our model because I am building a separate model for each company because we cannot predict a revenue of one company based on any other company.

So, we converted data into this shape.

	A	B	C	D	E
1	datetime	ROE	Company	Year	
2	2010	-12.697	113	2010	
3	2011	-12.722	113	2011	
4	2012	-57.835	113	2012	
5	2013	-10.988	113	2013	
6	2014	-0.342	113	2014	
7	2015	-33	113	2015	
8	2016	-1.899	113	2016	
9	2017	-2.534	113	2017	
10	2018	0.597	113	2018	
11	2019	6.438	113	2019	
12					
13					

code for converting data is given bellow, also provided in the source code.

```

1 df = pd.read_excel('Data_sets/orignal_data_files/ROA')
2 companies_names = df.Company
3 correct_name = [x.strip() for x in companies_names]
4 final_name = []
5 for name in correct_name:
6     if name[-1] == '.' or name[-1] == ',':
7         final_name.append(name[:-1])
8     else:
9         final_name.append(name)
10 df['Company'] = final_name
11 df.to_csv('Data_sets/orignal_data_files/ROE.csv', index=False)
12
13 df = pd.read_csv('Data_sets/orignal_data_files/ROA.csv')
14 comany_names = df['Company']
15 unique_comp_names = set(comany_names)
16 df_temp = ""
17 dir_name = 'Data_sets/company_data_ROA/'
18 for com_name in unique_comp_names:
19     df_temp = df[df['Company'] == com_name]
20     df_temp = df_temp.iloc[:, 1:]
21     path_file = dir_name + com_name + '.csv'
22     df_temp.to_csv(path_file)
23 transform_dir = 'Data_sets/company_transform_data_ROA/'
24 com_label = 1
25 Year_list = [2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019]
26 for com in unique_comp_names:
27     Roa_list = []
28     com_label_list = []
29     path_file = dir_name + com + '.csv'
30     df = pd.read_csv(path_file)
31     df = df.loc[0]
32     index_list = list(df.index)
33     R_i_list = index_list[3:13]
34     Roa_list = df[R_i_list]
35     Roa_list = Roa_list.values.tolist()
36     Roa_list.reverse()
37     for i in range(len(Year_list)):
38         com_label_list.append(com_label)
39         com_label += 1
40     my_dic = {'datetime': Year_list, 'ROA': Roa_list, 'Company': com_label_list, 'Year': Year_list}
41     df_temp = pd.DataFrame(my_dic)
42     path_file = transform_dir + com + '.csv'
43     df_temp.to_csv(path_file, index=False)

```

## Split data into train and test:

Now our data is converted in the required format, so let's split data into train and test to build and train our models. I used 80% data for training and 20% data for testing.

After splitting data, we need to transform data into required format which our model needs for training and testing

Data transformation is one of the major tasks because in time series data we need to prepare data using walk-forward validation

Input -> predict

Year1 Year2

Year1+Year2 Year3

Year1+Year2+Year3 Year4

For this purpose, I use to functions `split_dataset ()` and `to_supervised ()`

Code for this data transformation is given bellow as well as in the source code.

```
45 def split_dataset(data):
46     _80_percent = int((80/100)*len(data))
47     train, test = data[0:_80_percent], data[_80_percent:]
48     train = array(split(train, len(train)/2))
49     test = array(split(test, len(test)/2))
50     return train, test
51
52 def to_supervised(train, n_input, n_out=1):
53     data = train.reshape((train.shape[0]*train.shape[1], train.shape[2]))
54     X, y = list(), list()
55     in_start = 0
56     for _ in range(len(data)):
57         in_end = in_start + n_input
58         out_end = in_end + n_out
59         if out_end <= len(data):
60             x_input = data[in_start:in_end, 0]
61             x_input = x_input.reshape((len(x_input), 1))
62             X.append(x_input)
63             y.append(data[in_end:out_end, 0])
64             in_start += 1
65     return array(X), array(y)
```

## Build and save model.

Our data preprocessing is done and our data is ready for training a model so let's not train and save our models into hard disk, why we are saving models in hard disk? We are saving models in

hard disk after training them because we cannot train them again and again because it is very time and computational expensive task. Code for training and saving model is given bellow.

```
# Extracting the companies names from directory
companies_names_ROA_ = os.listdir('Data_sets/company_transform_data_ROA/')
companies_names_ROE = os.listdir('Data_sets/company_transform_data_ROE/')
companies_names_EPS = os.listdir('Data_sets/company_transform_data_Basic_EPS/')

# Removing the extension we can only have name for that company
companies_names_ROA_REVENUE = [fn[:-4] for fn in companies_names_ROA_]
companies_names_for_ROE = [fn[:-4] for fn in companies_names_ROE]
companies_names_for_EPS = [fn[:-4] for fn in companies_names_EPS]
```

First, I am extracting all the available companies' names and then remove the extension from the end so we can have only company names, now in bellow code I am training and saving our models

```
# Build and train models and save them into hard disk so we don't need to train them again
# which is very time consuming and computational expensive task

# Train and save model for ROA and REVENUE
for file_name in companies_names_ROA_REVENUE:
    Train_and_save_model(file_name, "Data_sets/company_transform_data_ROA/", 'Models/ROA_model/')
    Train_and_save_model(file_name, "Data_sets/company_transform_data_Revenue/", 'Models/Revenue_model/' )

# Train and save model for ROE
for file_name in companies_names_for_ROE:
    Train_and_save_model(file_name, "Data_sets/company_transform_data_ROE/", 'Models/ROE_model/')

# Train and save model for Both EPS
for file_name in companies_names_for_EPS:
    Train_and_save_EPS(file_name, 'Data_sets/company_transform_data_Basic_EPS/', 'Models/B_EPS_model/')
    Train_and_save_EPS(file_name, 'Data_sets/company_transform_data_Diluted_EPS/', 'Models/D_EPS_model/')
```

Ok now we can see above code for all files (companies) we are training and saving models but wait where model is training and saving all models? Actually, for ROA, ROE, REVENUE and both EPS I have two sperate functions one for ROE, ROA and Revenue and one for both EPS. Why I am using sperate function for EPS from ROA, ROE, REVENUE, because for EPS we have data set from 2015 to 2019 and for ROA, ROE, REVENUE I have data set from 2010 to 2019, yes I can also make a single function instead of two by applying some changes but I am going this way because it makes more sense and read able. Now let's see those function in bellow code.

```

# This function will train models for ROA, ROE and Revenue, then store them in it hard disk for futher use.
def Train_and_save_model(file_name,file_path,save_path):
    Year_list = [2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019,2020,2021,2022,2023,2024,2025]
    n_input , n_out = 2 , 2
    verbose, epochs, batch_size = 0, 699, 1
    file_path += file_name+'.csv'
    dataset = pd.read_csv(file_path, infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
    train, test = split_dataset(dataset.values)
    train_x, train_y = to_supervised(train, n_input, n_out)
    n_timesteps, n_features, n_outputs = train_x.shape[1], train_x.shape[2], train_y.shape[1]
    model = Sequential()
    model.add(LSTM(200, activation='relu', input_shape=(n_timesteps, n_features)))
    model.add(Dense(100, activation='relu'))
    model.add(Dense(n_outputs))
    model.compile(loss='mse', optimizer='adam')
    history = model.fit(train_x, train_y, validation_split=0.33, epochs=epochs, batch_size=batch_size, verbose=verbose)
    model.save(save_path+file_name+'.h5')

# This function will train models for Basic and Diluted EPS, then store them in it hard disk for futher use.
def Train_and_save_EPS(file_name,file_path,save_path):
    Year_list = [2015, 2016, 2017, 2018, 2019,2020,2021,2022,2023,2024,2025]
    n_input , n_out = 1 , 1
    verbose, epochs, batch_size = 0, 700, 2
    file_path += file_name+'.csv'
    dataset = pd.read_csv(file_path, infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
    train, test = split_dataset_EPS(dataset.values)
    train_x, train_y = to_supervised(train, n_input, n_out)
    n_timesteps, n_features, n_outputs = train_x.shape[1], train_x.shape[2], train_y.shape[1]
    model = Sequential()
    model.add(LSTM(200, activation='relu', input_shape=(n_timesteps, n_features)))
    model.add(Dense(100, activation='relu'))
    model.add(Dense(n_outputs))
    model.compile(loss='mse', optimizer='adam')
    history = model.fit(train_x, train_y, validation_split=0.33, epochs=epochs, batch_size=batch_size, verbose=verbose)
    model.save(save_path+file_name+'.h5')

```

We can see the above code for Training and save model, both functions first loads dataset then split into train and test then pass it through to\_supervised function which transform the data and then we build and train our model after that we are saving our model using save function which stores the train model with .h5 extension

## Load and Predict model.

Now our models are trained and all we need to do now is to load model from hard disk into ram and use it for predictions. I am sharing the code in which I am loading a model and use it for Forecasting but first I am taking all those companies which are available in all categories by category I mean ROA, ROE, Revenue, Both EPS, let's see how I am getting similar companies and then forecasting.

```

# Extracting all those companies which are available for every category
same_companies = set(companies_names_ROA_REVENUE).intersection(set(companies_names_for_ROE))
same_companies = set(same_companies).intersection(set(companies_names_for_EPS))

```

We have similar companies now we will load models and forecast results. Here I am taking a ROE as sample and I will show you code of forecasting ROE, for others also work same so I am not showing here but if you want to see it please refer to source code file.

```
# This function is used for Forecasting ROA, ROE and Revenue
def forecast_ROE_ROA_REVENUE(file_name,file_path, model_path):
    Year_list = [2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019,2020,2021,2022,2023,2024,2025]
    n_input , n_out = 2 , 2
    file_path += file_name+'.csv'
    dataset = pd.read_csv(file_path, infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])

    train, test = split_dataset(dataset.values)
    model = load_model(model_path+file_name+'.hs')

    history = [x for x in train]

    predictions = []
    for i in range(len(test)):
        yhat_sequence = forecast(model, history, n_input)
        predictions.append(yhat_sequence)
        history.append(test[i, :])

    yhat_sequence = forecast(model, history, n_input)
    predictions.append(yhat_sequence)
    year_var = Year_list[10]
    for i in range(int(len(Year_list[10:])/2)):
        yhat_sequence = forecast(model, history, n_input)
        history.append(array((array([yhat_sequence[0], dataset.Company[0], year_var]),
                               array([yhat_sequence[1], dataset.Company[0], year_var+1]))))
        year_var += 2

    history = array(history)
    history = history.reshape((history.shape[0]*history.shape[1], history.shape[2]))
    return history

# file path for ROE data sets
filepath = "Data_sets/company_transform_data_ROE/"
# path where ROE model is stored
modelpath = 'Models/ROE_model/'

performance_of_companies = [forecast_ROE_ROA_REVENUE(x,filepath, modelpath) for x in same_companies]
```

In above code, we can see I am setting up a file path where my ROE files are stored and also path where ROE model has been stored then I am using list comprehension where I am using all same companies one by one and pass it to function names as forecast\_ROE\_ROA\_REVENUE Which loads the data and model and the forecast and return the results of forecast using list variable named as history which will store all companies results in list named as performance\_of\_companies. Now we can use these results for classification and then store them in a file which we will do bellow.



## Classify model

We have already done predictions now we can use these predictions to Classify a company performance whether it is Good, worst or neutral. For classification what we are doing is we have true data from 2010 to 2019 and then we have prediction results from 2020 to 2025, so our classification formula is if predicted value of a company is increased by 5% of its current value then it is consider as Good and it drops by 5% then worst other wise neutral, for performing this task I am using 2019 value as a current value and taking average of 2020 to 2025 values as a expected value and by using these values I am classifying a company performance, now let us see of it works in a code.

```
performances_classes = []
Expected_values = []
current_values = []

for Pc in performance_of_companies:
    current_value = Pc[9][0]
    new_number = find_average(Pc[10:],0)
    performances_class = find_class(current_value, new_number)

    performances_classes.append(performances_class)
    Expected_values.append(new_number)
    current_values.append(current_value)

# Make a dictionary from the data.
per_based_ROE_dic = {"Company": list(same_companies), "Current ROE": current_values, "Expected ROE":Expected_values, "Performance":performances_classes}
# Make a data frame using above dictionary.
ROE_Based_Performance = pd.DataFrame(data=per_based_ROE_dic)
# Store the Performances data into excel file.
ROE_Based_Performance.to_excel('Performance Based On ROE.xlsx')

# Store the Forecast results in excel file.
df_roe = store_results(performance_of_companies,"ROE",same_companies)
df_roe.to_excel('ROE Forcast Results.xlsx')
```

In this above code first, I am classifying the companies and then making dictionary using this data and then making a data frame using dictionary and after that I stored the Performance base classification results as well as forecasting results in excel file. We can see in above code for performing this task I am using three functions one for finding the average ok why I am finding the average and average of whom. As I mention above for classification we need two ROE current and expected so we have 2010 to 2019 true ROE and 2020 to 2025 predicted ROE so I am taking 2019 ROE as current ROE and then finding the average of 2020 to 2025 ROE and considering it as expected ROE, so now I can use current ROE and Expected ROE for finding the class of company for this purpose I am using function named as find\_class which take current and new ROE and return its class by computing the formula you will see below in the code and third function for just arranging the data in required format so I can store into excel file function name is make\_result\_dataframe. Let's see the code bellow

```

# This function used for finding the average of 2020 to 2025 values
def find_average(numbers):
    total_len = len(numbers)
    sum_of_all = 0
    for num in numbers:
        sum_of_all += num

    return sum_of_all/total_len

# This function is used for classifying a class based on curren value and new value
def find_class(curent_value, new_value):
    percentage = ((new_value-curent_value)/abs(curent_value))*100
    if percentage >= 5:
        return 'Good'
    elif percentage <= -5:
        return 'Worst'
    else:
        return 'Neutral'

# This function is used to make a data frame from the results of forecast of ROA,ROE and Revenue
def make_result_dataframe(raw_data,name,company_names):
    Year_list = [2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019,2020,2021,2022,2023,2024,2025]
    col_names = ["Company"]
    for x in Year_list:
        col_names.append(name+" "+str(x))
    df = pd.DataFrame(columns=col_names)
    for x in range(len(raw_data)):
        company_data = raw_data[x]
        df = df.append({
            "Company": company_names[x],
            col_names[1]:company_data[0][0],
            col_names[2]:company_data[1][0], col_names[3]:company_data[2][0],
            col_names[4]:company_data[3][0], col_names[5]:company_data[4][0],
            col_names[6]:company_data[5][0], col_names[7]:company_data[6][0],
            col_names[8]:company_data[7][0], col_names[9]:company_data[8][0],
            col_names[10]:company_data[9][0], col_names[11]:company_data[10][0],
            col_names[12]:company_data[11][0],col_names[13]:company_data[12][0],
            col_names[14]:company_data[13][0],col_names[15]:company_data[14][0],
            col_names[16]:company_data[15][0]
        },ignore_index=True)

    return df

```

## Visualize Results

As we seen above, we train models then get forecasting results and on bases of them we classify companies and also store results in excel format, now in this phase we will visualize the results and get some interesting information so let's start.

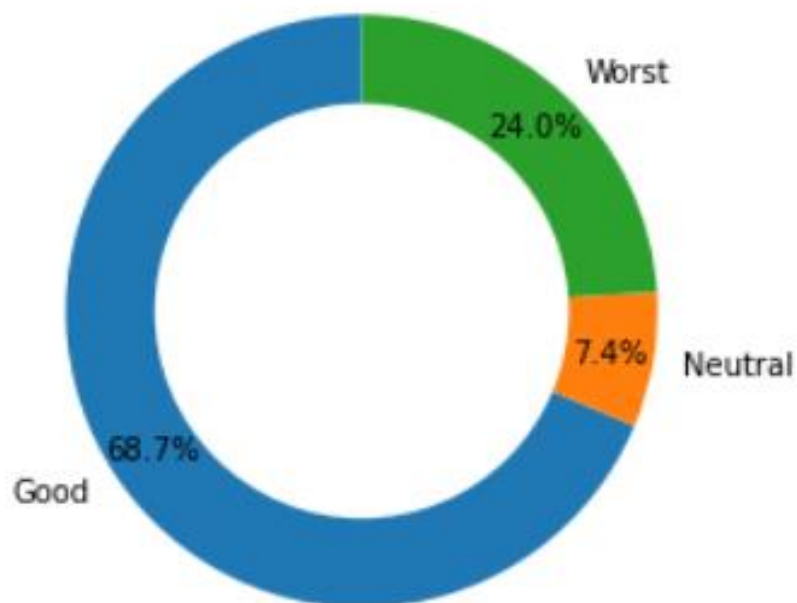
The approach I am using for visualize results is first I will generate and interesting question and then we will see the answer of that question in the form of Graph. I have generated similar question for ROE, ROA, Revenue and Both Eps, so here in this report I will show you all answers for Revenue if you want to see answers for others ROE, ROE and both EPS you can see it in source code.



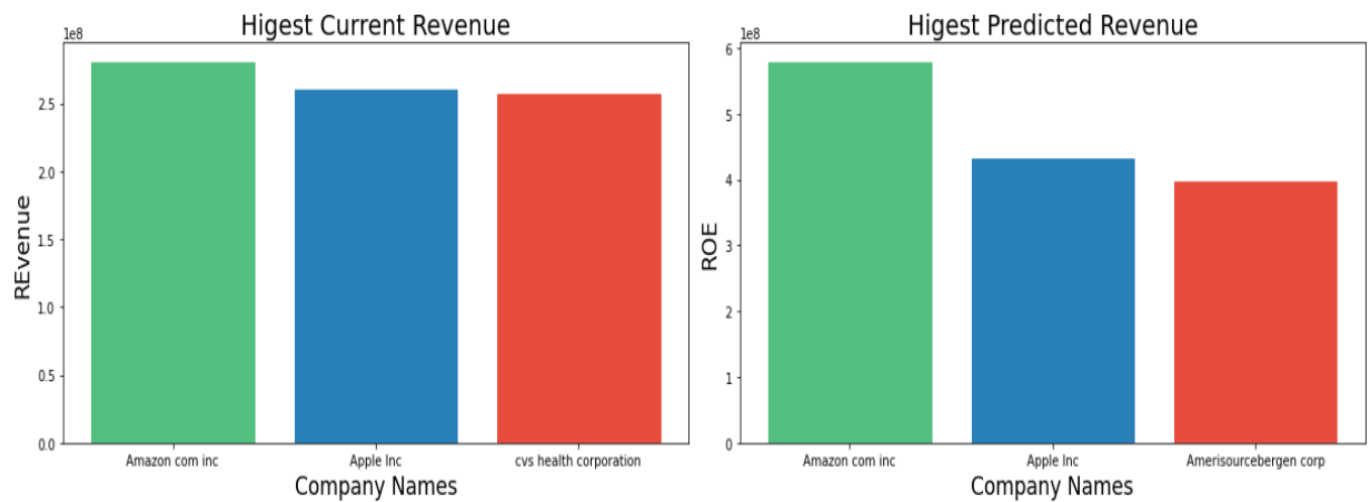
**Q no 1: How many companies lies in Good, Worst or Neutral class according to Revenue?**

	Company	Current REVENUE	Expected REVENUE
<b>Performance</b>			
<b>Good</b>	149	149	149
<b>Neutral</b>	16	16	16
<b>Worst</b>	52	52	52

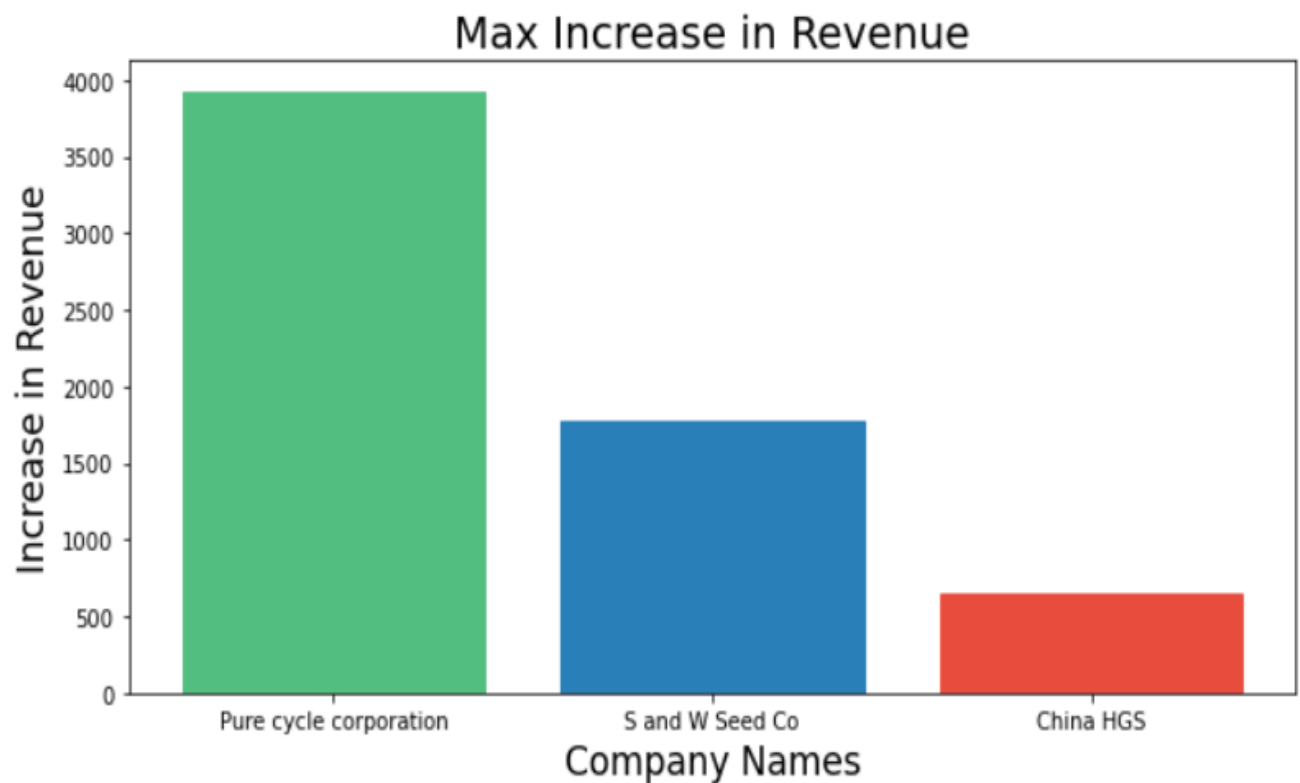
## Peformance Measure Based on Revenue



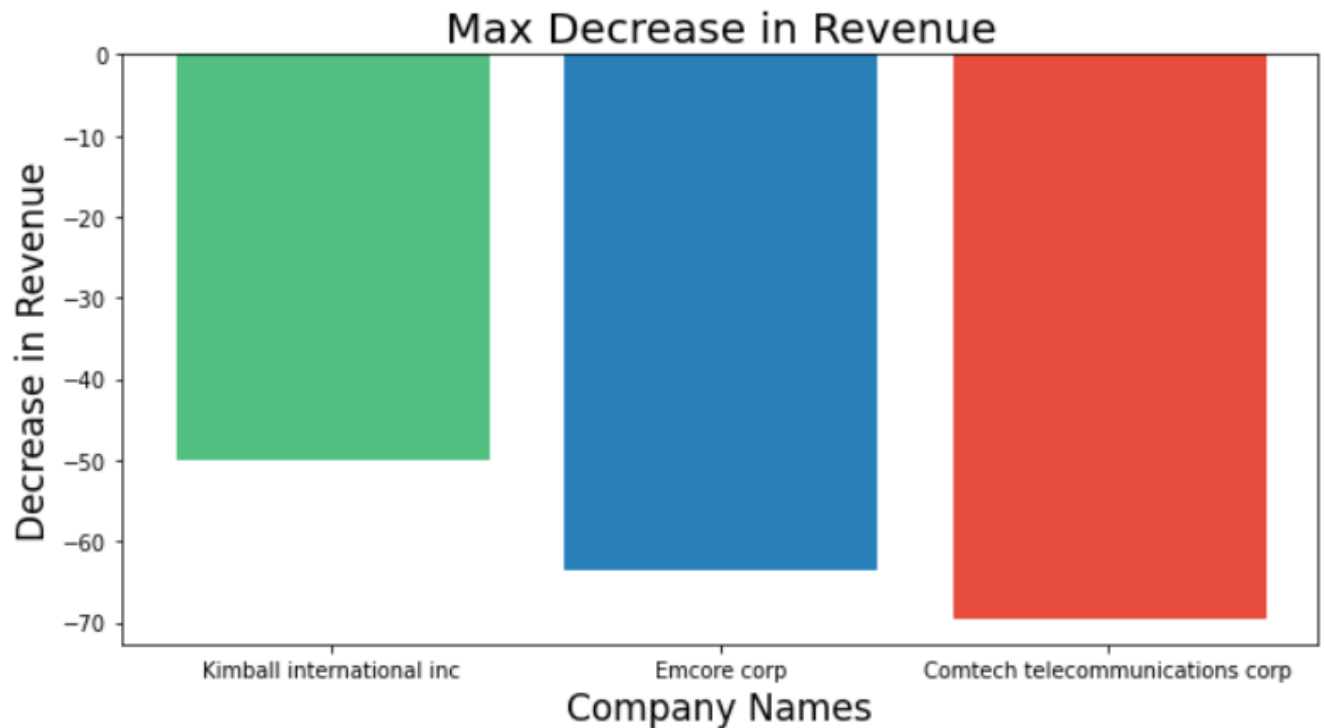
**Q no 2: Which company have highest Revenue now and who will have Highest Revenue in future?**



**Q no 3: Tell me Top 3 companies whose Revenue will most increase.**



**Q no 4: Tell me Top 3 companies whose Revenue will most decrease.**



## ROE and EPS Prediction Using LSTM

(Time Series Forecasting)

### Description:

I have a data for past 10 years of 244 companies in which I have ROE and w.r.t per year and same as BASIC and DILUTED EPS. I made LSTM models for predicting ROE and EPS, for each I made two models in one model I use univariate approach and in other multivariate. I train the model using 2010 to 2019 data and the predict 2019 to 2025 and predict the ROE and EPS.

### Steps:

9. Separate data for individual company.
10. Split data into train and test.

- 11.Transform data into required format in which our model needs for training and testing.
- 12.Build model
- 13.Predict model
- 14.Visualize Results

## Separate for individual company:

Data was in this format

C	D	E	F	G	H	I	J	K	L	M	N	
Company	ROA 2019	ROA 2018	ROA 2017	ROA 2016	ROA 2015	ROA 2014	ROA 2013	ROA 2012	ROA 2011	ROA 2010	REVENUE 2019	REVENUE 2010
Alico inc	9.06	3.08	-2.26	1.54	2.87	3.69	9.88	9.99	3.94	-0.33	122,251	81
Cal Maine foods inc	4.69	10.95	-7.19	28.43	17.36	13.46	6.76	12.36	9.49	10.74	1,361,188	1,500
S and W Seed Co	-6.36	-3.43	-10.10	0.29	-2.59	0.42	-2.94	1.70	-5.35	2.47	109,723	64
America's car mart inc.	9.66	8.01	4.75	2.84	7.36	5.81	8.97	10.60	10.19	10.67	669,122	61
Autozone	16.34	14.31	13.83	14.43	14.32	14.23	14.75	14.85	14.46	13.25	11,863,743	11,200
Marinemax inc	4.59	6.14	3.68	4.13	10.33	2.80	3.93	0.30	-3.17	0.74	1,237,153	1,100
Advaxis inc	-36.71	n.s.	-99.78	-43.51	-39.32	-70.69	-87.09	n.s.	n.s.	n.s.	20,884	6
Alexion Pharmaceuticals Inc.	12.42	1.74	4.03	4.35	3.80	20.76	15.86	15.21	16.47	14.72	4,991,100	4,100
Applied DNA Sciences inc	n.s.	n.s.	n.s.	-78.22	-76.39	-74.55	-72.71	-70.87	-69.04	-67.20	5,389	3
Avid Bioservices inc	-6.81	-21.47	1.18	3.30	-51.67	-39.06	-66.09	n.s.	-98.23	-49.41	53,603	53
Biogen Inc	21.62	17.52	10.74	16.19	18.19	20.50	15.70	13.62	13.64	12.40	14,377,900	13,400
Bio-technie corporation	5.94	7.91	7.19	13.06	14.50	18.71	20.65	22.55	26.71	30.15	714,006	64
Cabot corp	5.13	-3.51	7.43	4.84	-10.86	4.87	3.61	8.82	7.51	5.34	3,337,000	3,200
Cabot microelectronics corp	1.73	14.09	10.43	8.23	8.50	8.44	9.53	7.74	8.22	8.65	1,037,696	59
Cardinal ethanol LLC	-4.80	5.28	8.43	8.78	24.59	52.45	15.62	1.20	14.50	12.36	260,669	26
Clorox co	16.03	16.27	15.33	14.37	13.93	13.11	13.27	12.42	13.38	13.26	6,214,000	6,100
Estee lauder companies Inc.	13.57	8.82	10.80	12.09	13.24	15.30	14.27	13.00	11.17	8.96	14,863,000	13,600
Grante falls energy inc	-7.91	2.45	9.04	7.18	11.14	34.92	8.90	0.26	19.38	13.99	208,777	21
H.B. Fuller company	3.28	4.10	1.36	5.92	4.24	2.66	5.17	7.03	7.26	6.15	2,897,000	3,000
Innovation pharmaceuticals inc.	n.s.	n.s.	n.s.	n.s.	-91.80	-76.00	-60.19	-44.38	-28.58	-12.77	0	
Landec corp	0.08	6.14	2.95	-3.40	3.91	6.10	7.76	4.57	1.90	1.99	557,559	52
Lannett company inc.	-22.92	1.82	-0.04	2.54	29.47	16.66	7.94	2.77	-0.19	5.59	655,407	68

We converted into this format for each company

	A	B	C	D	E
1	datetime	ROE	Company	Year	
2	2010	-12.697	113	2010	
3	2011	-12.722	113	2011	
4	2012	-57.835	113	2012	
5	2013	-10.988	113	2013	
6	2014	-0.342	113	2014	
7	2015	-33	113	2015	
8	2016	-1.899	113	2016	
9	2017	-2.534	113	2017	
10	2018	0.597	113	2018	
11	2019	6.438	113	2019	
12					
13					

Source Code for conversion is given in the source code folder

## Split data into train and test:

Then I split data into train and test, for training I used 80-70% data and 20-30% data for testing

Transform data into required format in which our model needs for training and testing

Data transformation is one of the major task because in time series data we need to prepare data using walk-forward validation

Input                      -> predict

Year1                      Year2

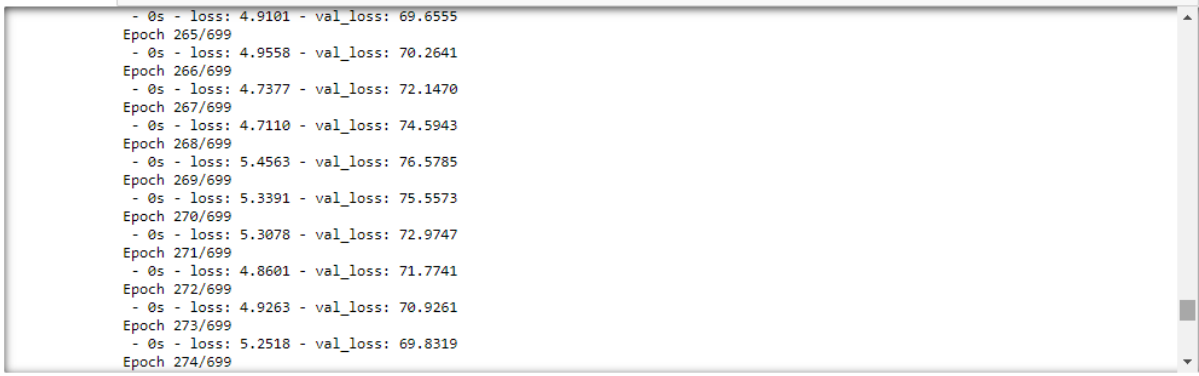
Year1+Year2              Year3

Year1+Year2+Year3      Year4

For this purpose I use to functions `split_dataset ()` and `to_supervised ()`

Build Model

I am attaching some of the screen shorts of building model.

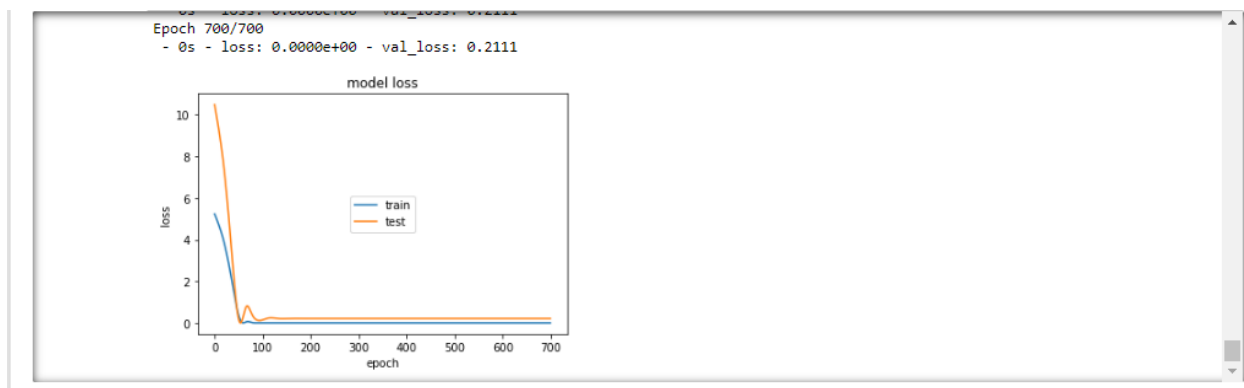


```
- 0s - loss: 4.9101 - val_loss: 69.6555
Epoch 265/699
- 0s - loss: 4.9558 - val_loss: 70.2641
Epoch 266/699
- 0s - loss: 4.7377 - val_loss: 72.1470
Epoch 267/699
- 0s - loss: 4.7110 - val_loss: 74.5943
Epoch 268/699
- 0s - loss: 5.4563 - val_loss: 76.5785
Epoch 269/699
- 0s - loss: 5.3391 - val_loss: 75.5573
Epoch 270/699
- 0s - loss: 5.3078 - val_loss: 72.9747
Epoch 271/699
- 0s - loss: 4.8601 - val_loss: 71.7741
Epoch 272/699
- 0s - loss: 4.9263 - val_loss: 70.9261
Epoch 273/699
- 0s - loss: 5.2518 - val_loss: 69.8319
Epoch 274/699
```

```

Epoch 246/699
- 0s - loss: 2.0166 - val_loss: 3.5890
Epoch 247/699
- 0s - loss: 2.1359 - val_loss: 3.8766
Epoch 248/699
- 0s - loss: 2.0067 - val_loss: 3.9239
Epoch 249/699
- 0s - loss: 1.9947 - val_loss: 4.0660
Epoch 250/699
- 0s - loss: 2.1048 - val_loss: 3.9269
Epoch 251/699
- 0s - loss: 2.0108 - val_loss: 3.6446
Epoch 252/699
- 0s - loss: 1.9855 - val_loss: 3.2155
Epoch 253/699
- 0s - loss: 1.9143 - val_loss: 2.9891
Epoch 254/699
- 0s - loss: 2.0328 - val_loss: 3.0032
Epoch 255/699
- 0s - loss: 2.0486 - val_loss: 3.1406

```



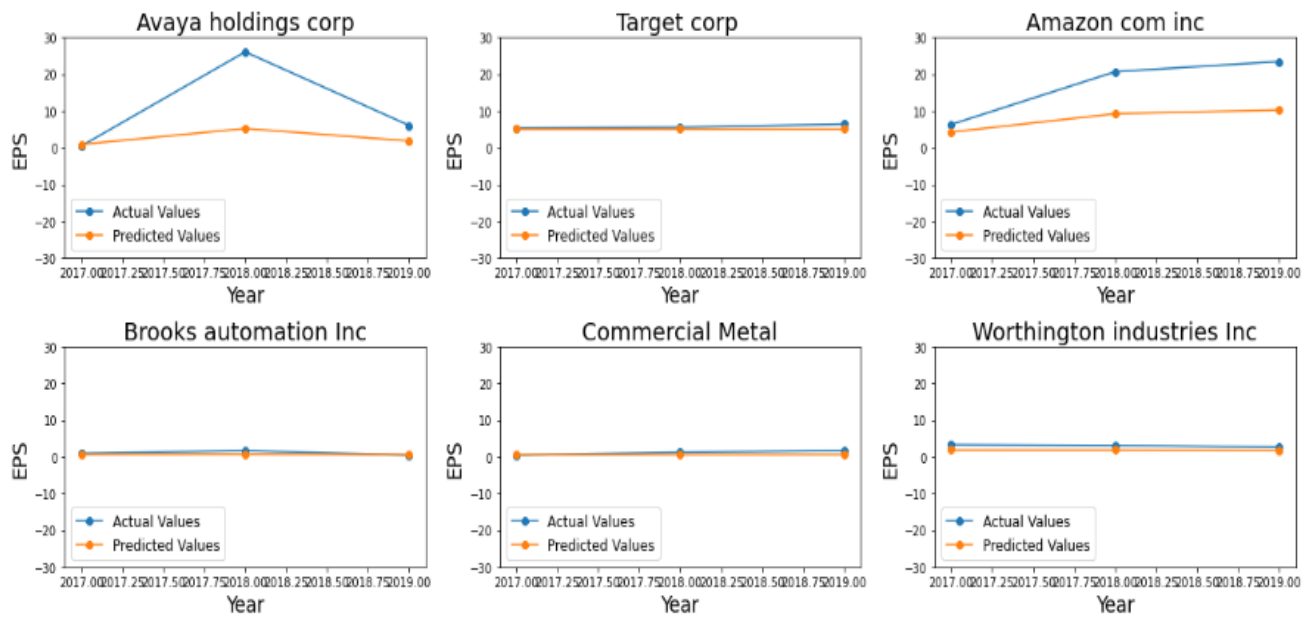
In the pic shows ho model is tearing and reducing the loss on each epoch and in the end graphical representation of training neural network.

## Predict mode:

In predicting model first I predict the test set in which I have samples of 2018 and 2019 year and after that I took dummy data for years 2020 to 2025 and give to model for prediction and then in visualization part visualized some results in the form of graph and some in printed text output.

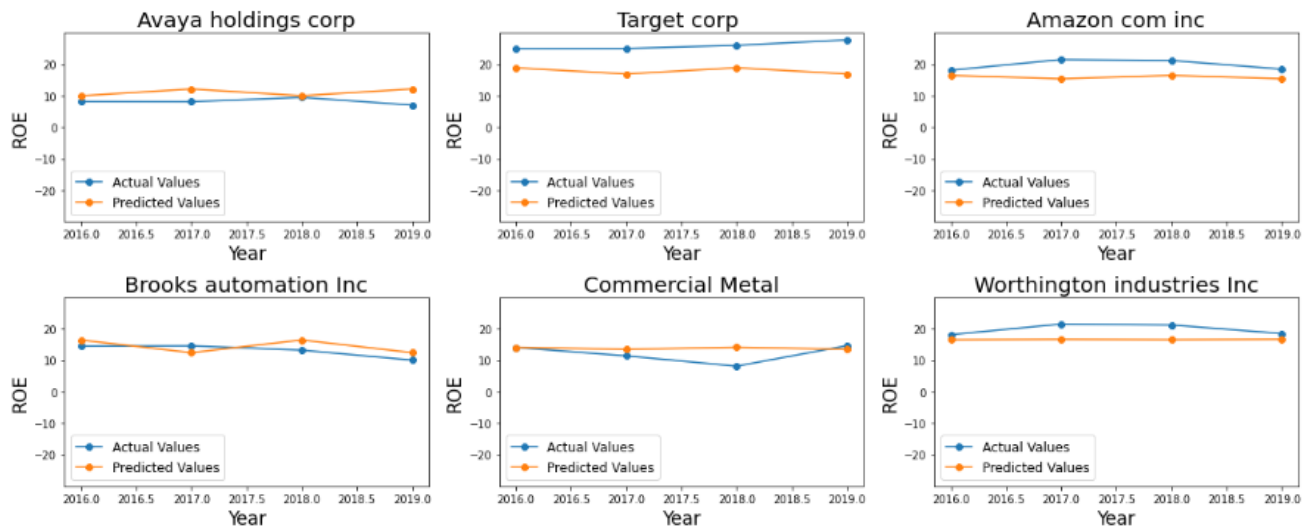
## Visualization:



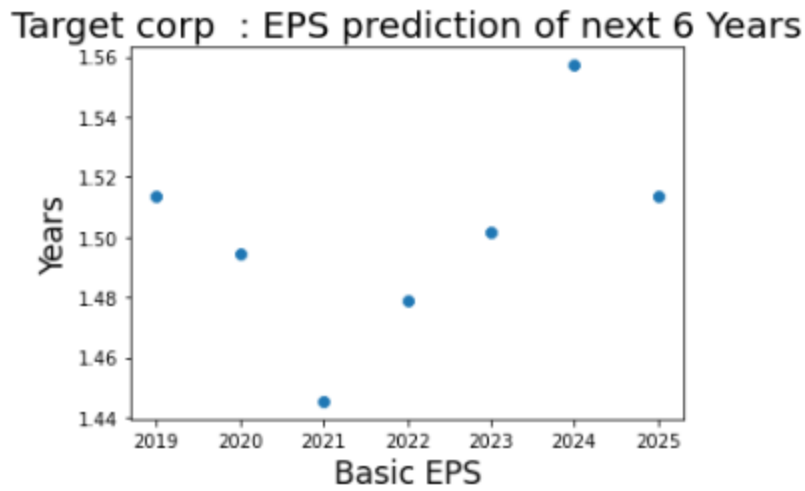


This figure shows that I took 6 random companies data and give it to the model and model gave the prediction then I graph the actual and predicted results. One y-axis we have EPS and in x-axis we have Years. 2017, 2018 and 2019

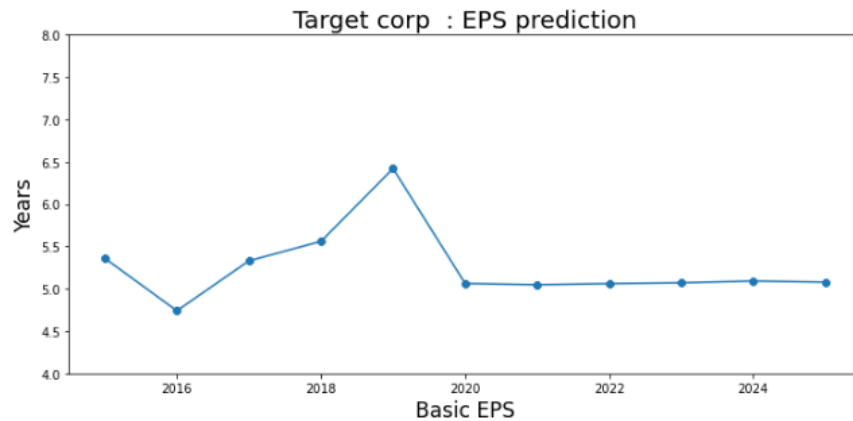
Then same did for ROE



After that I predict EPS and ROE for 2020 to 2025 for random taken a company named as Target Crop and we got these results.



After that I plot the 2020 to 2025 prediction with the past 5 year so we can get clear picture



```

***** : Avaya holdings corp : *****
Better Performance

***** : Target corp : *****
Better Performance

***** : Amazon com inc : *****
Better Performance

***** : Brooks automation Inc : *****
Better Performance

***** : Commercial Metal : *****
Better Performance

***** : Worthington industries Inc : *****
Better Performance

*****

```

Some results related to better, worst or neutral performance.

I took the average of all predict performance and find the increase or decrees w.r.t to 2019 year performance.

## Code Explanation:

### Extracting the data for individual company and storing in csv

```

In [3]: comany_names = df['Company ']
unique_comp_names = set(comany_names)
df_temp = ""
dir_name = 'company_data_EPS/'
for com_name in unique_comp_names:
    df_temp = df[df['Company '] == com_name]
    df_temp = df_temp.iloc[:, 1:]
    path_file = dir_name + com_name + '.csv'
    df_temp.to_csv(path_file)

```

In this code I am extracting each individual company data from over all data set  
And storing into csv file

### Transform data for each company into required format

```
In [4]: transform_dir = 'company_transform_data_EPS/'

com_label = 1
year_list = [2015, 2016, 2017, 2018, 2019]
E_i_list = []
D_E_i_list = []
or com in unique_comp_names:
    Diluted_EPS_list = []
    Basic_EPS_list = []
    com_label_list = []
    path_file = dir_name + com + '.csv'
    df = pd.read_csv(path_file)
    df = df.loc[0]
    index_list = list(df.index)
    D_E_i_list = index_list[3:8]
    B_E_i_list = index_list[8:]
    D_E_list = df[D_E_i_list]
    B_E_list = df[B_E_i_list]
    D_E_list = D_E_list.values.tolist()
    B_E_list = B_E_list.values.tolist()
    for i in range(len(Year_list)):
        com_label_list.append(com_label)
    com_label += 1
my_dic = {'datetime': Year_list, 'Basic EPS': B_E_list, 'Diluted EPS': D_E_list, 'Company': com_label_list, 'Year': Year_list}
df_temp = pd.DataFrame(my_dic)
path_file = transform_dir + com + '.csv'
df_temp.to_csv(path_file, index=False)
```

In this code I am converting data into Time series shape data were date and time

Is stored as index and values of that time are store in that row so we in our case we have index suppose 2010 and store all 2010 ROE or EPS recorders in that row who's index is 2010

```
[22]: def split_dataset(data):
    train, test = data[0:8], data[8:]
    train = array(split(train, len(train)/2))
    test = array(split(test, len(test)/2))
    return train, test

def to_supervised(train, n_input, n_out=2):
    data = train.reshape((train.shape[0]*train.shape[1], train.shape[2]))
    X, y = list(), list()
    in_start = 0
    for _ in range(len(data)):
        in_end = in_start + n_input
        out_end = in_end + n_out
        if out_end <= len(data):
            x_input = data[in_start:in_end, 0]
            x_input = x_input.reshape((len(x_input), 1))
            X.append(x_input)
            y.append(data[in_end:out_end, 0])
        in_start += 1
    return array(X), array(y)

def forecast(model, history, n_input):
    data = array(history)
    data = data.reshape((data.shape[0]*data.shape[1], data.shape[2]))
    input_x = data[-n_input:, 0]
    input_x = input_x.reshape((1, len(input_x), 1))
    yhat = model.predict(input_x, verbose=0)
    yhat = yhat[0]
    return yhat
```

Ok now after our data has been converted according to Time series format then before training our model we need to split data into train and test and then we need to convert the shape, the shape is required by our model, here shape of data means our data dimension for example 2d, 3d. Then after training model we need our model to predict so for these 3 task we have 3 functions.

First function named as split\_dataset is used for first task splitting the data into train and test and return it. I am using 80% data for training and 20% data for testing.

Second function to\_supervised is used for 2<sup>nd</sup> task it will do two thing first it split x and y and then make it like our model need to be given as input for if I take example related to year and ROE for univariate then data will be like this

Train X (ROE)	Train Y (ROE)
2010	2011
2010, 2011	2012
2010, 2011, 2012	2013

Third forecast function take model and history and change the shape of data and give model to predict and the values and return the predicted values.

Ok now we know what these three function do but one this is really important to understand that meaning and motive of these function is same for ROE and EPS or

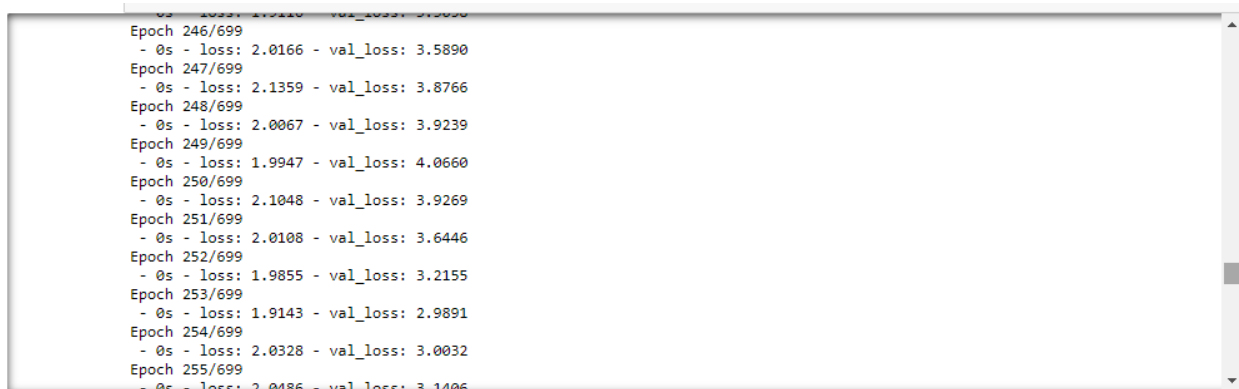
Univariate ROE or multivariate ROE or EPS. (In univariate we use single value like train on prior ROE as input and Predict next ROE and in multivariate we are using multiple columns for training and we can predict one for more value)

So motive is same but for each of above mentioned model we need to change these function and adjust according to the needed data.

```
model = Sequential()
model.add(LSTM(200, activation='relu', input_shape=(n_timesteps, n_features)))
model.add(Dense(100, activation='relu'))
model.add(Dense(n_outputs))
model.compile(loss='mse', optimizer='adam')
history = model.fit(train_x, train_y, validation_split=0.33, epochs=epochs, batch_size=batch_size, verbose=verbose)

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='center')
plt.show()
```

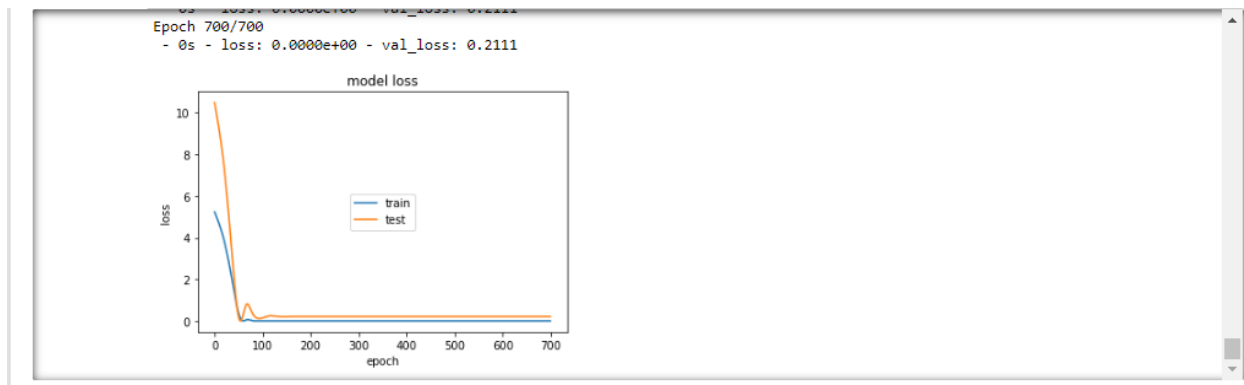
So now when everything is ready as our algorithms need we need to build model and train it on train x and train y and after training we are also printing the graph how our model change values and reduce the loss and train on each epochs(you can say as step or iteration). Here I am also sharing some pic while training model and graph of it while training how neural networks change the weights and reduce the loss.



```
Epoch 246/699
- 0s - loss: 2.0166 - val_loss: 3.5890
Epoch 247/699
- 0s - loss: 2.1359 - val_loss: 3.8766
Epoch 248/699
- 0s - loss: 2.0067 - val_loss: 3.9239
Epoch 249/699
- 0s - loss: 1.9947 - val_loss: 4.0660
Epoch 250/699
- 0s - loss: 2.1048 - val_loss: 3.9269
Epoch 251/699
- 0s - loss: 2.0108 - val_loss: 3.6446
Epoch 252/699
- 0s - loss: 1.9855 - val_loss: 3.2155
Epoch 253/699
- 0s - loss: 1.9143 - val_loss: 2.9891
Epoch 254/699
- 0s - loss: 2.0328 - val_loss: 3.0032
Epoch 255/699
- 0s - loss: 2.0486 - val_loss: 3.1406
```

Shows reducing the loss against each epoch





Visualize the training procedure.

```
history = [x for x in train]

predictions = list()
for i in range(len(test)):
    yhat_sequence = forecast(model, history, n_input)
    predictions.append(yhat_sequence)
    history.append(test[i, :])

yhat_sequence = forecast(model, history, n_input)
predictions.append(yhat_sequence)
```

After model is trained we need to do predictions so we are doing prediction using forecast function and appending the prediction and actual values and storing then in required format so we can use them to make graphs and visualize the results.

As visualization the result I have already explain above.

## ROA and Revenue Prediction Using LSTM

(Time Series Forecasting)

### Description:

I have a data for past 10 years of 244 companies in which I have ROA and revenue. I made LSTM models for predicting ROA and revenue, for each I made model in which I use univariate approach. I train the model using 2010 to 2019 data and the predict 2019 to 2025 and predict the ROA and revenue.

### Steps:

1. Separate data for individual company.
2. Split data into train and test.
3. Transform data into required format in which our model needs for training and testing.
4. This step is only for Revenue, in this step I normalize the revenue so we can build our model more efficient. We will discuss this in detail later
5. Build model
6. Predict model
7. Visualize Results

## Separate for individual company:

Data was in this format

C	D	E	F	G	H	I	J	K	L	M	N	O
Company	ROA 2019	ROA 2018	ROA 2017	ROA 2016	ROA 2015	ROA 2014	ROA 2013	ROA 2012	ROA 2011	ROA 2010	REVENUE 2019	REVENUE 2010
Alico inc	9.06	3.08	-2.26	1.54	2.87	3.69	9.88	9.99	3.94	-0.33	122,251	81
Cal Maine foods inc	4.69	10.95	-7.19	28.43	17.36	13.46	6.76	12.36	9.49	10.74	1,361,188	1,54
S and W Seed Co	-6.36	-3.43	-10.10	0.29	-2.59	0.42	-2.94	1.70	-5.35	2.47	109,723	64
America's car mart inc.	9.66	8.01	4.75	2.84	7.36	5.81	8.97	10.60	10.19	10.67	669,122	61
Autozone	16.34	14.31	13.83	14.43	14.32	14.23	14.75	14.85	14.46	13.25	11,863,743	11,2
Marinemax inc	4.59	6.14	3.68	4.13	10.33	2.80	3.93	0.30	-3.17	0.74	1,237,153	1,1
Advaxis inc	-36.71	n.s.	-99.78	-43.51	-39.32	-70.69	-87.09	n.s.	n.s.	n.s.	20,884	6
Alexion Pharmaceuticals Inc.	12.42	1.74	4.03	4.35	3.80	20.76	15.86	15.21	16.47	14.72	4,991,100	4,1
Applied DNA Sciences inc	n.s.	n.s.	n.s.	-78.22	-76.39	-74.55	-72.71	-70.87	-69.04	-67.20	5,389	3
Avid Bioservices inc	-6.81	-21.47	1.18	3.30	-51.67	-39.06	-66.09	n.s.	-98.23	-49.41	53,603	53
Biogen Inc	21.62	17.52	10.74	16.19	18.19	20.50	15.70	13.62	13.64	12.40	14,377,900	13,4
Bio-technie corporation	5.94	7.91	7.19	13.06	14.50	18.71	20.65	22.55	26.71	30.15	714,006	64
Cabot corp	5.13	-3.51	7.43	4.84	-10.86	4.87	3.61	8.82	7.51	5.34	3,337,000	3,2
Cabot microelectronics corp	1.73	14.09	10.43	8.23	8.50	8.44	9.53	7.74	8.22	8.65	1,037,696	59
Cardinal ethanol LLC	-4.80	5.28	8.43	8.78	24.59	52.45	15.62	1.20	14.50	12.36	260,669	26
Clorox co	16.03	16.27	15.33	14.37	13.93	13.11	13.27	12.42	13.38	13.26	6,214,000	6,1
Estee lauder companies Inc.	13.57	8.82	10.80	12.09	13.24	15.30	14.27	13.00	11.17	8.96	14,863,000	13,6
Grante falls energy inc	-7.91	2.45	9.04	7.18	11.14	34.92	8.90	0.26	19.38	13.99	208,777	21
H.B. Fuller company	3.28	4.10	1.36	5.92	4.24	2.66	5.17	7.03	7.26	6.15	2,897,000	3,0
Innovation pharmaceuticals inc.	n.s.	n.s.	n.s.	n.s.	-91.80	-76.00	-60.19	-44.38	-28.58	-12.77	0	
Landec corp	0.08	6.14	2.95	-3.40	3.91	6.10	7.76	4.57	1.90	1.99	557,559	52
Lannett company inc.	-22.92	1.82	-0.04	2.54	29.47	16.66	7.94	2.77	-0.19	5.59	655,407	68

We converted into this format for each company

	A	B	C	D
1	datetime	Revenue	Company	Year
2	2010	1626900	20	2010
3	2011	1795700	20	2011
4	2012	1933700	20	2012
5	2013	2089100	20	2013
6	2014	2393500	20	2014
7	2015	2706700	20	2015
8	2016	3291300	20	2016
9	2017	3505100	20	2017
10	2018	3680100	20	2018
11	2019	3672700	20	2019

	A	B	C	D
1	datetime	ROA	Company	Year
2	2010	19.561	166	2010
3	2011	-0.02	166	2011
4	2012	-30.425	166	2012
5	2013	-1.706	166	2013
6	2014	-10.565	166	2014
7	2015	-20.947	166	2015
8	2016	-13.49	166	2016
9	2017	-0.225	166	2017
10	2018	7.243	166	2018
11	2019	6.171	166	2019

Source Code for conversion is given in the source code folder

## Split data into train and test:

Then I split data into train and test, for training I used 80-70% data and 20-30% data for testing

Transform data into required format in which our model needs for training and testing

Data transformation is one of the major task because in time series data we need to prepare data using walk-forward validation

Input -> predict

Year1 Year2

Year1+Year2 Year3

Year1+Year2+Year3 Year4

For this purpose I use to functions `split_dataset ()` and `to_supervised ()`

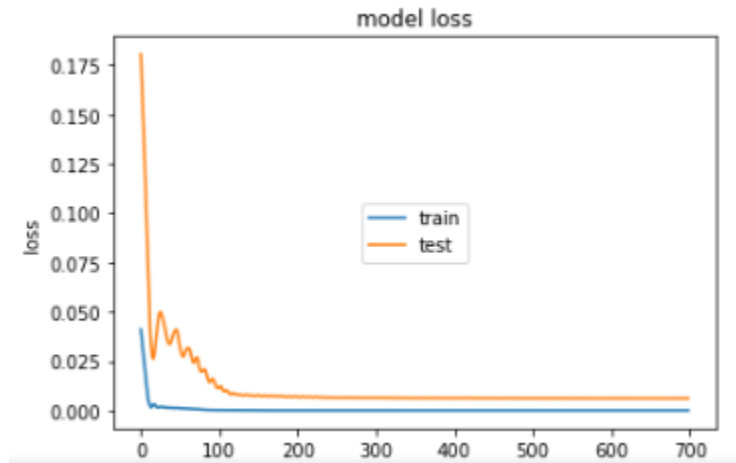
Build Model

I am attaching some of the screen shorts of building model.

```
Train on 3 samples, validate on 2 samples
Epoch 1/699
- 0s - loss: 0.3720 - val_loss: 0.0948
Epoch 2/699
- 0s - loss: 0.3465 - val_loss: 0.0848
Epoch 3/699
- 0s - loss: 0.3224 - val_loss: 0.0756
Epoch 4/699
- 0s - loss: 0.2970 - val_loss: 0.0665
Epoch 5/699
- 0s - loss: 0.2753 - val_loss: 0.0571
Epoch 6/699
- 0s - loss: 0.2480 - val_loss: 0.0476
Epoch 7/699
- 0s - loss: 0.2217 - val_loss: 0.0382
Epoch 8/699
- 0s - loss: 0.1939 - val_loss: 0.0290
Epoch 9/699
- 0s - loss: 0.1676 - val_loss: 0.0204
Epoch 10/699
```

Here we can see in starting loss is high but on each epoch it reduce the loss this shows that our model is training and getting better after each epoch. Here I have attached some snapshots of model training.

```
- 0s - loss: 0.0044 - val_loss: 0.0403
Epoch 53/699
- 0s - loss: 0.0045 - val_loss: 0.0414
Epoch 54/699
- 0s - loss: 0.0045 - val_loss: 0.0411
Epoch 55/699
- 0s - loss: 0.0047 - val_loss: 0.0397
Epoch 56/699
- 0s - loss: 0.0045 - val_loss: 0.0391
Epoch 57/699
- 0s - loss: 0.0043 - val_loss: 0.0389
Epoch 58/699
- 0s - loss: 0.0042 - val_loss: 0.0379
Epoch 59/699
- 0s - loss: 0.0041 - val_loss: 0.0364
Epoch 60/699
- 0s - loss: 0.0044 - val_loss: 0.0348
Epoch 61/699
- 0s - loss: 0.0042 - val_loss: 0.0346
Epoch 62/699
```



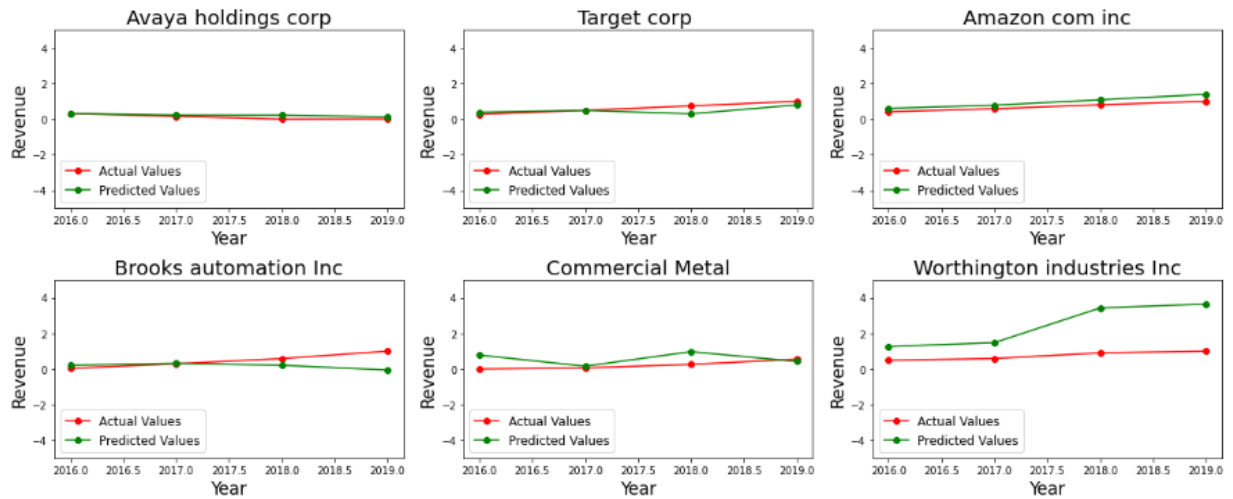
Finally when model is train if we plot the training graph we can see how model has been train after each epoch.

### **Predict mode:**

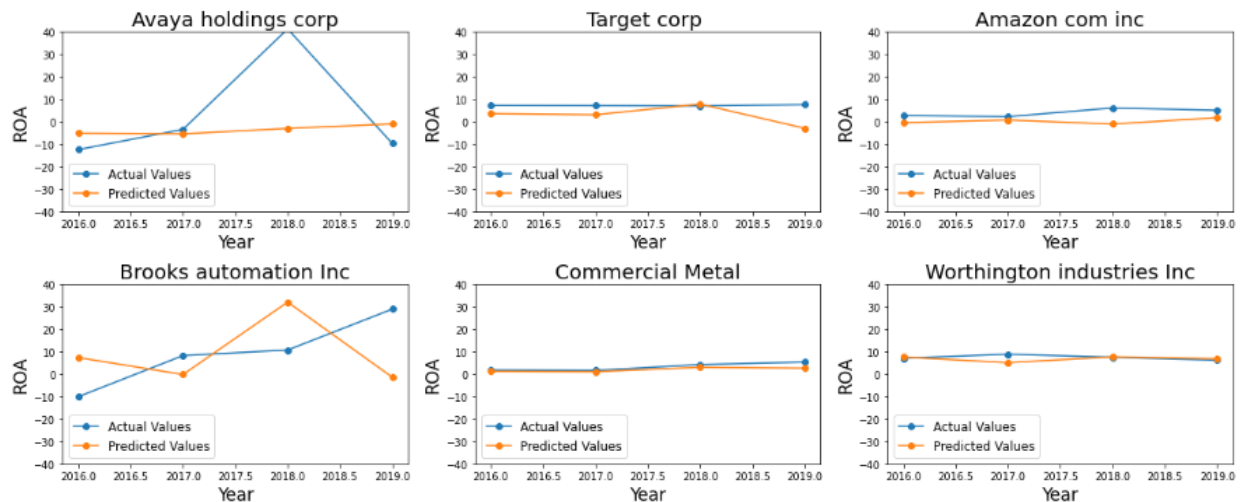
In predicting model first I predict the test set in which I have samples of 2018 and 2019 year and after that I took dummy data for years 2020 to 2025 and give to model for prediction and then in visualization part visualized some results in the form of graph and some in printed text output.

### **Visualization:**

Revenue



## ROA

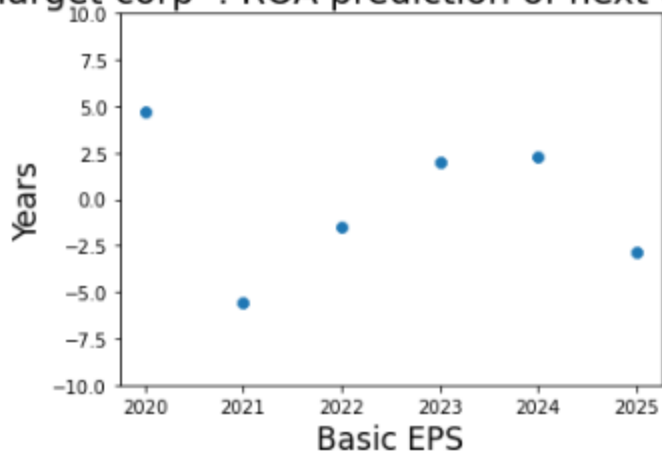


These figures shows that I took 6 random companies data and give it to the model and model gave the prediction then I graph the actual and predicted results. On Y-axis we have ROA, Revenue and in x-axis we have Years. 2017, 2018 and 2019.

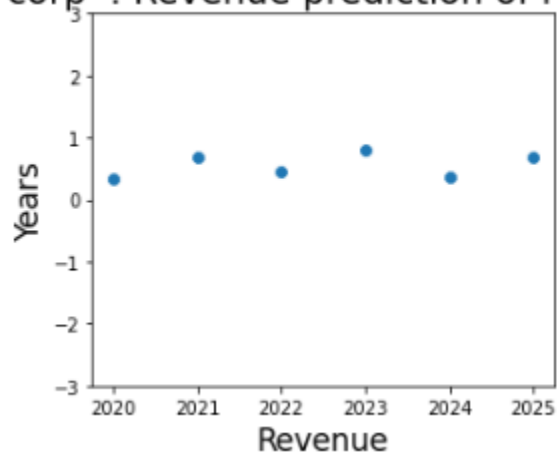
After that I predict ROA and Revenue for 2020 to 2025 for random taken a company named as Target Crop and we got these results.



Target corp : ROA prediction of next 6 Years

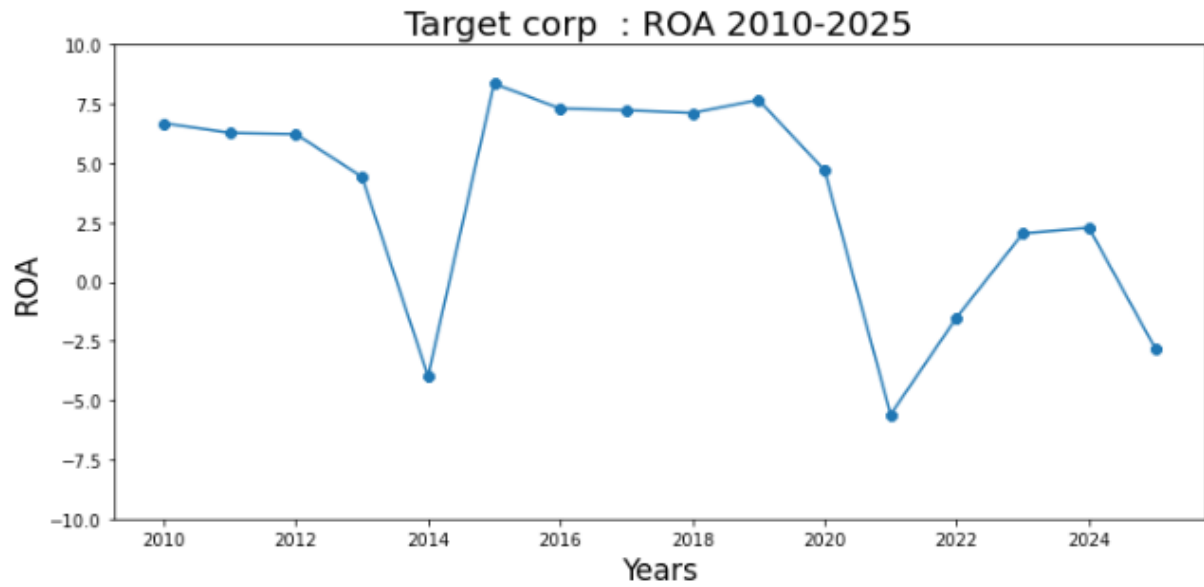


Target corp : Revenue prediction of next 6 Years



After that I plot the 2020 to 2025 prediction with the past 5 year so we can get clear picture

ROA Graph:



Revenue Graph:



**Now Let's Talk about Data Normalization which I used for Revenue**

As we see the revenue data was in high numbers.

	A	B	C	D
1	datetime	Revenue	Company	Year
2	2010	34204000	59	2010
3	2011	48077000	59	2011
4	2012	61093000	59	2012
5	2013	74452000	59	2013
6	2014	88988000	59	2014
7	2015	107006000	59	2015
8	2016	135987000	59	2016
9	2017	177866000	59	2017
10	2018	232887000	59	2018
11	2019	280522000	59	2019

So, it is better we can normalize it and bring data into common scale, without distorting differences in the ranges of values. So, it's easy to train our model so I use this formula for normalization.

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

**Thanks**