Predicting_Apple_Stock_Linear_Regression_Updated

August 12, 2019

FACTORS IN DATASET AND THEIR IMPORTANCE 1. Debt to Equity Ratio 2. ROI 3. ROE 4. ROA 5. Net Income 6. Revenue 7. Current Ratio 8. Gross Margin 9. Free Cash Flow

In [1]:
```
import requests
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
from mpl_toolkits.mplot3d import Axes3D
```

In [2]:
```
#Import BeautifulSoup
from bs4 import BeautifulSoup
```

In [3]:
```
#Scraping Data Data
page = requests.get("https://ycharts.com/companies/AAPL/net_income_ttm")
page

page2 = requests.get("https://ycharts.com/companies/AAPL/gross_profit_margin")
page2

page3 = requests.get("https://ycharts.com/companies/AAPL/free_cash_flow")
page3
```

Out[3]:
```
<Response [200]>
```

In [4]:
```
#Website Structure
soup = BeautifulSoup(page.content)
soup2 = BeautifulSoup(page2.content)
soup3 = BeautifulSoup(page3.content)
```

In [5]:
```
#Left DataTable and Right DataTable
My_table = soup.find('table',{'class':'histDataTable'})
My_table_right = soup.find('div',{'class':'dataColRt'})
```

In [6]:
```
#index array
index = [0]*50
for i in range(50):
```

```
        index[i] = i

    for i in range(len(index)):
        index[i] += 1

    index = pd.DataFrame(index)
    index.columns = ['index']
```

In [7]:
```
    #2006-2019 Quarter Dates List
    dates = []
    for tag in My_table.select(".col1 "):
      text = tag.get_text()
      dates.append(text)
    dates = dates[1:]
    dates = pd.Series(dates)

    dates2 =[]
    for tag in My_table_right.select(".col1 "):
      text = tag.get_text()
      dates2.append(text)
    dates2 = dates2[1:]
    dates2 = pd.Series(dates2)

    quarters = dates.append(dates2)
    quarters = pd.DataFrame(quarters)
    quarters.index = range(50)
    quarters.columns = ['Date']
```

In [8]:
```
    #2006-2019 Quarter NI list
    NI = []
    for tag in My_table.select(".col2 "):
      text = tag.get_text()
      NI.append(text)
    NI = NI[1:]
    NI = pd.Series(NI)
    NI = NI.str.strip(" ")
    NI = NI.str.strip("\n")
    NI = NI.str.strip(" ")
    NI = NI.str.strip("\n")
    NI = NI.str.strip(" ")
    NI = NI.str.strip("B")

    NI2 =[]
    for tag in My_table_right.select(".col2 "):
      text = tag.get_text()
      NI2.append(text)
    NI2 = NI2[1:]
    NI2 = pd.Series(NI2)
```

```
        NI2 = NI2.str.strip(" ")
        NI2 = NI2.str.strip("\n")
        NI2 = NI2.str.strip(" ")
        NI2 = NI2.str.strip("\n")
        NI2 = NI2.str.strip(" ")
        NI2 = NI2.str.strip("B")

        NIFinal = NI.append(NI2)
        NIFinal = pd.DataFrame(NIFinal)
        NIFinal.index = range(50)
        NIFinal.columns = ['Net Income']
```

```
In [9]: #2006-2019 revenue list
        revenue = pd.read_csv("applerev.csv")
        revenue = revenue['revenue'].dropna()
        revenue = pd.DataFrame(revenue)
        revenue.index = range(50)
        revenue.columns = ['Gross Revenue']
```

```
In [10]: #2006-2019 stock open price
         stockprice = pd.read_csv("appledata2.csv")
         stockprice = stockprice['Stock Price']
         stockprice = stockprice.dropna()
         stockprice = pd.DataFrame(stockprice)
         stockprice.index = range(50)
         stockprice.columns = ['Stock Price']
```

```
In [11]: #2006-2019 p/e ratio
         peratio = pd.read_csv("appledata2.csv")
         peratio = peratio['PE Ratio']
         peratio = peratio.dropna()
         peratio = pd.DataFrame(peratio)
         peratio.index = range(50)
         peratio.columns = ['P/E Ratio']
```

```
In [12]: #2006-2019 return on equity
         roe = pd.read_csv("appledata3.csv")
         roe = roe['Return on Equity']
         roe = roe.dropna()
         roe = pd.DataFrame(roe)
         roe.index = range(50)
         roe.columns = ['Return_On_Equity']
         roe = roe['Return_On_Equity'].str.rstrip('%')
```

```
In [13]: #2006-2019 return on investment
         roi = pd.read_csv("appledata4.csv")
         roi = roi['Return on Investment']
         roi = roi.dropna()
         roi = pd.DataFrame(roi)
```

```
roi.index = range(50)
roi.columns = ['Return_On_Investment']
roi = roi['Return_On_Investment'].str.rstrip('%')
```

In [14]:
```
# 2006-2019 return on assets
roa = pd.read_csv("appledata9.csv")
roa = roa['Return on Assets']
roa = roa.dropna()
roa = pd.DataFrame(roa)
roa.index = range(50)
roa.columns = ['Return_On_Assets']
roa = roa['Return_On_Assets'].str.rstrip('%')
```

In [15]:
```
#2006-2019 current ratio
currratio = pd.read_csv("appledata6.csv")
currratio = currratio['Current Ratio']
currratio = currratio.dropna()
currratio = pd.DataFrame(currratio)
currratio.index = range(50)
currratio.columns = ['Current_Ratio']
```

In [16]:
```
#2006-2019 gross margin
grossmargin = pd.read_csv("appledata7.csv")
grossmargin = grossmargin['Gross Margin']
grossmargin = grossmargin.dropna()
grossmargin = pd.DataFrame(grossmargin)
grossmargin.index = range(50)
grossmargin.columns = ['Gross_Margin']
grossmargin = grossmargin['Gross_Margin'].str.rstrip('%')
```

In [17]:
```
#2006-2019 free cash flow
freecashflow = pd.read_csv("appledata8.csv")
freecashflow = freecashflow['Free Cash Flow']
freecashflow = freecashflow.dropna()
freecashflow = pd.DataFrame(freecashflow)
freecashflow.index = range(50)
freecashflow.columns = ['Free_Cash_Flow']
```

In [18]:
```
#Left DataTable and Right DataTable
My_table2 = soup2.find('div',{'class':'dataColLeft'})
My_table_right2 = soup2.find('div',{'class':'dataColRt'})

#2009-2016 gross profit margin
GPM = []
for tag in My_table2.select(".col2 "):
  text = tag.get_text()
  GPM.append(text)

GPM = GPM[1:]
```

```
GPM = pd.Series(GPM)
GPM = GPM.str.strip("\n")
GPM = GPM.str.strip(" ")
GPM = GPM.str.strip("\n")
GPM = GPM.str.strip(" ")
GPM = GPM.str.strip("\n")
GPM = GPM.str.strip("%")


GPM2 = []
for tag in My_table_right2.select(".col2 "):
  text = tag.get_text()
  GPM2.append(text)

GPM2 = GPM2[1:]
GPM2 = pd.Series(GPM2)
GPM2 = GPM2.str.strip("\n")
GPM2 = GPM2.str.strip(" ")
GPM2 = GPM2.str.strip("\n")
GPM2 = GPM2.str.strip(" ")
GPM2 = GPM2.str.strip("\n")
GPM2 = GPM2.str.strip("%")

GPMFinal = GPM.append(GPM2)
GPMFinal = pd.DataFrame(GPMFinal)
GPMFinal.index = range(50)
GPMFinal.columns = ['Gross_Profit_Margin']
```

In [19]:
```
#Left DataTable and Right DataTable
My_table3 = soup3.find('div',{'class':'dataColLeft'})
My_table_right3 = soup3.find('div',{'class':'dataColRt'})

#2009-2016 free cash flow
FCF = []
for tag in My_table3.select(".col2 "):
  text = tag.get_text()
  FCF.append(text)

FCF = FCF[1:]
FCF = pd.Series(FCF)
FCF = FCF.str.strip("\n")
FCF = FCF.str.strip(" ")
FCF = FCF.str.strip("\n")
FCF = FCF.str.strip(" ")
FCF = FCF.str.strip("\n")
FCF = FCF.str.strip("B")
```

```
FCF2 = []
for tag in My_table_right3.select(".col2 "):
  text = tag.get_text()
  FCF2.append(text)

FCF2 = FCF2[1:]
FCF2 = pd.Series(FCF2)
FCF2 = FCF2.str.strip("\n")
FCF2 = FCF2.str.strip(" ")
FCF2 = FCF2.str.strip("\n")
FCF2 = FCF2.str.strip(" ")
FCF2 = FCF2.str.strip("\n")
FCF2 = FCF2.str.strip("B")
FCF2 = FCF2.str.strip("M")


FCFFinal = FCF.append(FCF2)
FCFFinal = pd.DataFrame(FCFFinal)
FCFFinal.index = range(50)
FCFFinal.columns = ['Free_Cash_Flow']
```

In [20]:
```
#combining features into one dataframe
from typing import List, Any, Union

stock_feature_frames = [quarters,NIFinal,revenue,stockprice,peratio,roe,roi,roa,currra
stock_features = pd.concat(stock_feature_frames, axis=1)
stock_features.columns: List[Union[str, Any]] = ['Date','Net_Income','Gross_Revenue',
stock_features = stock_features.drop(columns=['Free_Cash_Flow2'])
stock_features.head()
```

Out[20]:

	Date	Net_Income	Gross_Revenue	Stock_Price	P/E_Ratio	\
0	June 30, 2019	55.70	58015.0	189.95	15.98	
1	March 31, 2019	57.17	62900.0	157.07	12.92	
2	Dec. 31, 2018	59.43	53265.0	223.99	18.87	
3	Sept. 30, 2018	59.53	61137.0	183.03	16.59	
4	June 30, 2018	56.12	84310.0	165.26	15.95	

	Return_On_Equity	Return_On_Investment	Return_On_Assets	Current_Ratio	\
0	51.29	31.95	15.98	1.32	
1	50.92	31.91	16.33	1.30	
2	48.68	32.03	15.99	1.13	
3	43.50	29.66	14.98	1.31	
4	39.97	28.54	14.27	1.46	

	Gross_Margin	Free_Cash_Flow	Gross_Profit_Margin
0	38.05	32127.0	37.59
1	38.21	23335.0	37.61
2	38.34	64121.0	37.99

```
3        38.27         47639.0              38.29
4        38.30         36418.0              38.34
```

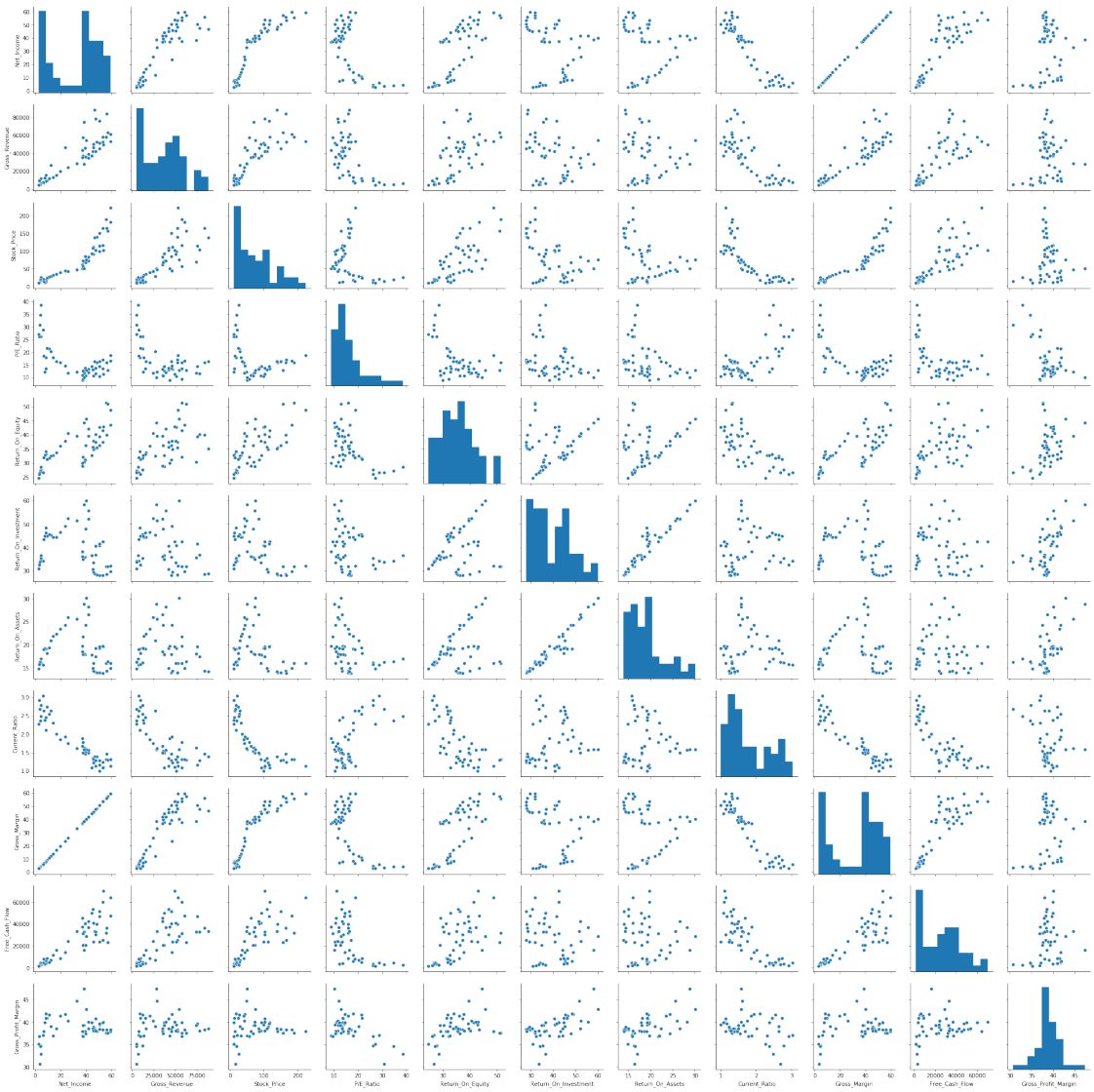
```
In [21]: #CHANGING DATA TYPES OF ALL COLUMNS TO FLOATS
         stock_features["Net_Income"] = pd.to_numeric(stock_features["Net_Income"])
         stock_features["Return_On_Equity"] = pd.to_numeric(stock_features["Return_On_Equity"])
         stock_features["Return_On_Investment"] = pd.to_numeric(stock_features["Return_On_Inves
         stock_features["Return_On_Assets"] = pd.to_numeric(stock_features["Return_On_Assets"])
         stock_features["Gross_Margin"] = pd.to_numeric(stock_features["Net_Income"])
         stock_features["Gross_Profit_Margin"] = pd.to_numeric(stock_features["Gross_Profit_Mar
         stock_features.dtypes
```

```
Out[21]: Date                    object
         Net_Income             float64
         Gross_Revenue          float64
         Stock_Price            float64
         P/E_Ratio              float64
         Return_On_Equity       float64
         Return_On_Investment   float64
         Return_On_Assets       float64
         Current_Ratio          float64
         Gross_Margin           float64
         Free_Cash_Flow         float64
         Gross_Profit_Margin    float64
         dtype: object
```

DATA VISUALIZATIONS

```
In [22]: #pairplot
         sns.pairplot(stock_features)
```

```
Out[22]: <seaborn.axisgrid.PairGrid at 0x1a25cc65f8>
```

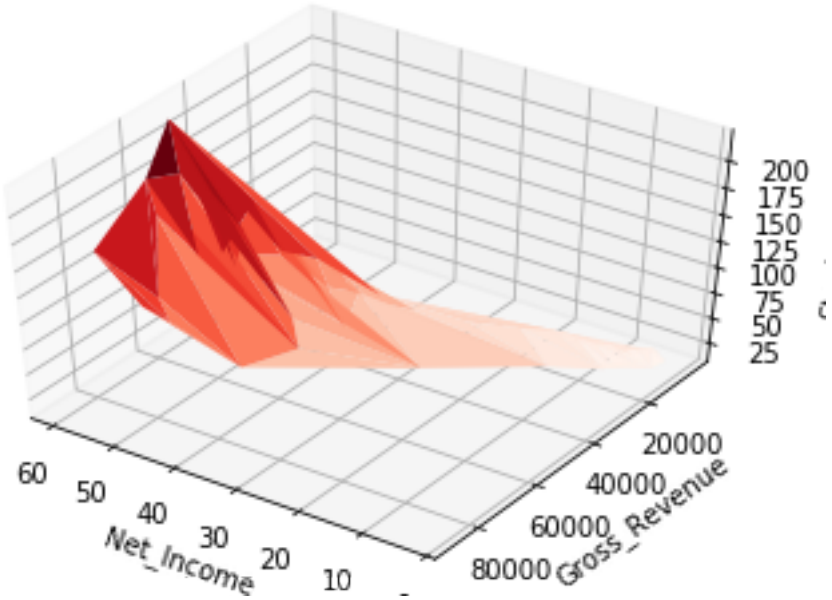


In [23]:
```
#plotting correlation heatmap
corr = stock_features.corr()
sns.heatmap(corr)
```

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x1a295c9b00>

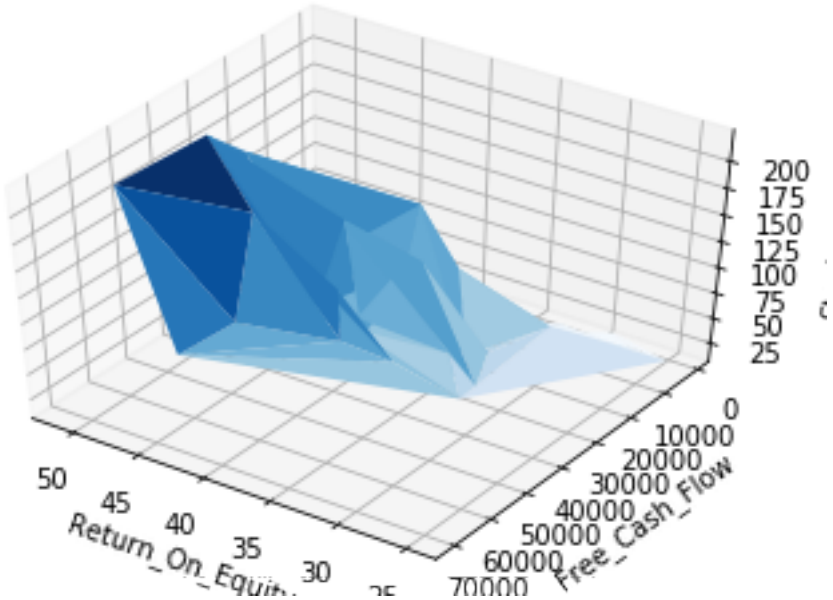
In [24]:
```
#plotting multiple variables
fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot_trisurf(stock_features['Net_Income'], stock_features['Gross_Revenue'], stock_
ax.view_init(45, 125)
ax.set_xlabel('Net_Income')
ax.set_ylabel('Gross_Revenue')
ax.set_zlabel('Stock_Price')
plt.show()

#CONCLUSION: as net income and gross-revenue go up, stock price goes up
#            as net income goes up gross revenue goes up also
```



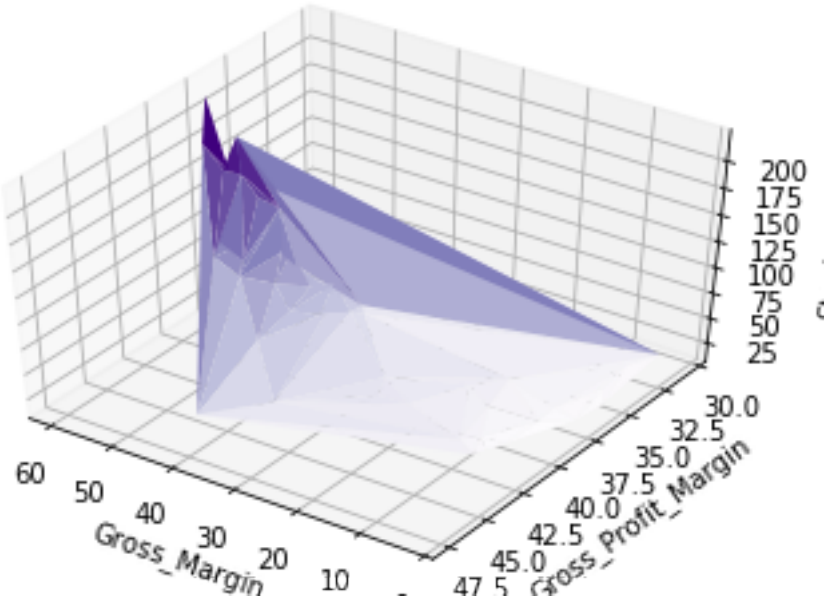
In [25]:
```
fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot_trisurf(stock_features['Return_On_Equity'], stock_features['Free_Cash_Flow'],
ax.view_init(45, 125)
ax.set_xlabel('Return_On_Equity')
ax.set_ylabel('Free_Cash_Flow')
ax.set_zlabel('Stock_Price')
plt.show()

#CONCLUSION: as roe and free cash flow go up, stock price goes up
#            as roe goes up free cash flow goes up also
```



In [26]:
```
fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot_trisurf(stock_features['Gross_Margin'], stock_features['Gross_Profit_Margin']
ax.view_init(45, 125)
ax.set_xlabel('Gross_Margin')
ax.set_ylabel('Gross_Profit_Margin')
ax.set_zlabel('Stock_Price')
plt.show()

#CONCLUSION: as gross margin and gross-profit-margin go up, stock price goes up
#            as gross margin goes up gross-profit-margin goes up also
```



In [27]:
```
#time series data
date = pd.to_datetime(stock_features['Date'])
date = pd.DataFrame(date)
date.columns = ['date']
```

In [28]:
```
stock_features = pd.concat([stock_features, date], axis=1)
stock_features
```

Out[28]:

	Date	Net_Income	Gross_Revenue	Stock_Price	P/E_Ratio	\
0	June 30, 2019	55.700	58015.0	189.95	15.98	
1	March 31, 2019	57.170	62900.0	157.07	12.92	
2	Dec. 31, 2018	59.430	53265.0	223.99	18.87	
3	Sept. 30, 2018	59.530	61137.0	183.03	16.59	
4	June 30, 2018	56.120	84310.0	165.26	15.95	
5	March 31, 2018	53.320	52579.0	166.02	17.06	
6	Dec. 31, 2017	50.520	45408.0	150.65	16.38	
7	Sept. 30, 2017	48.350	52896.0	140.21	15.93	
8	June 30, 2017	46.650	88293.0	139.29	16.29	
9	March 31, 2017	45.730	46852.0	111.82	13.39	
10	Dec. 31, 2016	45.220	42358.0	108.58	13.13	
11	Sept. 30, 2016	45.690	50557.0	91.33	10.67	
12	June 30, 2016	47.800	78351.0	103.48	11.51	
13	March 31, 2016	50.680	51501.0	99.41	10.55	
14	Dec. 31, 2015	53.730	49605.0	103.39	11.24	
15	Sept. 30, 2015	53.390	58010.0	117.41	13.56	
16	June 30, 2015	50.740	75872.0	116.00	14.34	
17	March 31, 2015	47.810	42123.0	102.50	13.81	

18	Dec. 31, 2014	44.460	37432.0	93.16	14.48
19	Sept. 30, 2014	39.510	45646.0	85.50	13.81
20	June 30, 2014	38.560	74599.0	70.15	11.73
21	March 31, 2014	37.710	37472.0	72.89	12.66
22	Dec. 31, 2013	37.030	35323.0	61.58	10.88
23	Sept. 30, 2013	37.040	43603.0	50.88	8.90
24	June 30, 2013	37.750	57594.0	56.43	9.43
25	March 31, 2013	39.670	35966.0	67.46	10.71
26	Dec. 31, 2012	41.750	35023.0	84.16	13.34
27	Sept. 30, 2012	41.730	39186.0	73.36	12.07
28	June 30, 2012	40.130	54512.0	75.32	12.86
29	March 31, 2012	38.620	28270.0	50.88	10.14
30	Dec. 31, 2011	32.980	28571.0	47.90	12.12
31	Sept. 30, 2011	25.920	24667.0	42.17	11.68
32	June 30, 2011	23.610	46333.0	43.78	14.61
33	March 31, 2011	19.550	20343.0	40.52	15.84
34	Dec. 31, 2010	16.640	15700.0	35.65	16.47
35	Sept. 30, 2010	14.010	13499.0	31.60	17.94
36	June 30, 2010	12.240	26741.0	29.52	20.32
37	March 31, 2010	10.810	12207.0	26.47	21.40
38	Dec. 31, 2009	9.358	9734.0	23.28	21.70
39	Sept. 30, 2009	8.235	9084.0	17.89	18.02
40	June 30, 2009	8.124	15683.0	13.21	13.61
41	March 31, 2009	7.368	12907.0	10.72	12.24
42	Dec. 31, 2008	6.793	7464.0	14.28	18.65
43	Sept. 30, 2008	6.119	7512.0	21.03	28.81
44	June 30, 2008	4.601	11880.0	18.03	26.13
45	March 31, 2008	4.347	6789.0	24.88	38.62
46	Dec. 31, 2007	4.072	5410.0	19.28	34.78
47	Sept. 30, 2007	3.495	5264.0	15.33	30.75
48	June 30, 2007	3.134	9608.0	11.67	26.19
49	March 31, 2007	2.788	4837.0	10.66	27.13

	Return_On_Equity	Return_On_Investment	Return_On_Assets	Current_Ratio	\
0	51.29	31.95	15.98	1.32	
1	50.92	31.91	16.33	1.30	
2	48.68	32.03	15.99	1.13	
3	43.50	29.66	14.98	1.31	
4	39.97	28.54	14.27	1.46	
5	37.37	28.05	13.83	1.24	
6	36.29	27.94	13.95	1.28	
7	35.40	28.21	14.00	1.39	
8	35.09	28.81	14.15	1.39	
9	34.94	29.42	14.31	1.23	
10	35.59	30.76	14.91	1.35	
11	37.89	33.56	16.01	1.31	
12	40.24	36.78	17.44	1.28	
13	42.79	40.87	19.22	1.00	

14	42.94	42.48	19.66	1.11
15	41.46	42.45	19.74	1.09
16	39.44	41.58	19.56	1.16
17	37.36	40.64	19.28	1.13
18	32.76	36.57	17.85	1.08
19	31.20	35.78	17.92	1.47
20	30.36	35.63	18.00	1.63
21	28.93	35.00	17.92	1.49
22	29.06	36.05	18.57	1.68
23	29.93	38.29	19.69	1.88
24	32.20	42.44	21.74	1.78
25	36.32	47.94	24.34	1.54
26	39.51	52.30	26.56	1.50
27	42.14	55.66	28.22	1.57
28	45.63	60.04	30.13	1.58
29	44.35	58.22	28.89	1.58
30	39.56	51.57	25.62	1.61
31	40.48	52.34	25.97	1.75
32	37.77	49.04	24.32	1.93
33	35.99	46.48	23.46	1.85
34	33.76	44.30	22.34	2.01
35	32.66	44.36	21.93	2.31
36	32.60	45.29	20.93	2.64
37	31.83	45.45	19.41	2.55
38	31.45	44.83	18.13	2.74
39	34.06	48.43	19.08	2.11
40	33.06	46.26	19.15	2.46
41	32.78	44.89	19.25	2.38
42	31.88	43.38	19.06	2.64
43	26.67	34.15	15.65	3.04
44	27.69	35.29	16.17	2.78
45	28.57	36.57	17.01	2.49
46	27.19	34.28	16.42	2.37
47	26.74	33.76	16.28	2.68
48	26.05	32.54	15.82	2.92
49	24.76	30.85	14.78	2.27

	Gross_Margin	Free_Cash_Flow	Gross_Profit_Margin	date
0	55.700	32127.0	37.59	2019-06-30
1	57.170	23335.0	37.61	2019-03-31
2	59.430	64121.0	37.99	2018-12-31
3	59.530	47639.0	38.29	2018-09-30
4	56.120	36418.0	38.34	2018-06-30
5	53.320	25483.0	38.31	2018-03-31
6	50.520	51774.0	38.41	2017-12-31
7	48.350	39890.0	37.91	2017-09-30
8	46.650	33495.0	38.51	2017-06-30
9	45.730	23900.0	38.93	2017-03-31

```
10        45.220         53497.0                38.51 2016-12-31
11        45.690         40941.0                38.02 2016-09-30
12        47.800         33116.0                38.02 2016-06-30
13        50.680         23851.0                39.40 2016-03-31
14        53.730         70019.0                40.10 2015-12-31
15        53.390         60162.0                39.90 2015-09-30
16        50.740         47217.0                39.68 2015-06-30
17        47.810         30505.0                40.78 2015-03-31
18        44.460         50142.0                39.87 2014-12-31
19        39.510         40718.0                38.01 2014-09-30
20        38.560         32841.0                39.36 2014-06-30
21        37.710         20685.0                39.32 2014-03-31
22        37.030         45501.0                37.93 2013-12-31
23        37.040         37548.0                37.02 2013-09-30
24        37.750         31605.0                36.87 2013-06-30
25        39.670         21109.0                37.50 2013-03-31
26        41.750         42561.0                38.63 2012-12-31
27        41.730         36886.0                40.04 2012-09-30
28        40.130         28753.0                42.81 2012-06-30
29        38.620         16233.0                47.37 2012-03-31
30        32.980         33269.0                44.68 2011-12-31
31        25.920         24485.0                40.25 2011-09-30
32        23.610         14154.0                41.73 2011-06-30
33        19.550          8559.0                41.42 2011-03-31
34        16.640         16590.0                38.51 2010-12-31
35        14.010         11667.0                36.93 2010-09-30
36        12.240          7461.0                39.08 2010-06-30
37        10.810          5405.0                41.67 2010-03-31
38         9.358          9015.0                40.88 2009-12-31
39         8.235          6364.0                41.82 2009-09-30
40         8.124          4340.0                40.92 2009-06-30
41         7.368          3599.0                39.93 2009-03-31
42         6.793          8505.0                37.94 2008-12-31
43         6.119          4613.0                37.12 2008-09-30
44         4.601          3596.0                34.83 2008-06-30
45         4.347          2563.0                32.93 2008-03-31
46         4.072          4735.0                34.68 2007-12-31
47         3.495          3244.0                30.76 2007-09-30
48         3.134          2300.0                36.88 2007-06-30
49         2.788          1671.0                35.13 2007-03-31
```

```
In [29]: stock_features = stock_features.drop(columns = ['Date'])
         stock_features['Gross_Revenue'] = stock_features['Gross_Revenue']/1000
         stock_features.columns = ['Net_Income','Gross_Revenue','Stock_Price', 'PE_Ratio','Retu
         stock_features
```

```
Out[29]:    Net_Income  Gross_Revenue  Stock_Price  PE_Ratio  Return_On_Equity  \
         0      55.700         58.015       189.95     15.98             51.29
```

1	57.170	62.900	157.07	12.92	50.92
2	59.430	53.265	223.99	18.87	48.68
3	59.530	61.137	183.03	16.59	43.50
4	56.120	84.310	165.26	15.95	39.97
5	53.320	52.579	166.02	17.06	37.37
6	50.520	45.408	150.65	16.38	36.29
7	48.350	52.896	140.21	15.93	35.40
8	46.650	88.293	139.29	16.29	35.09
9	45.730	46.852	111.82	13.39	34.94
10	45.220	42.358	108.58	13.13	35.59
11	45.690	50.557	91.33	10.67	37.89
12	47.800	78.351	103.48	11.51	40.24
13	50.680	51.501	99.41	10.55	42.79
14	53.730	49.605	103.39	11.24	42.94
15	53.390	58.010	117.41	13.56	41.46
16	50.740	75.872	116.00	14.34	39.44
17	47.810	42.123	102.50	13.81	37.36
18	44.460	37.432	93.16	14.48	32.76
19	39.510	45.646	85.50	13.81	31.20
20	38.560	74.599	70.15	11.73	30.36
21	37.710	37.472	72.89	12.66	28.93
22	37.030	35.323	61.58	10.88	29.06
23	37.040	43.603	50.88	8.90	29.93
24	37.750	57.594	56.43	9.43	32.20
25	39.670	35.966	67.46	10.71	36.32
26	41.750	35.023	84.16	13.34	39.51
27	41.730	39.186	73.36	12.07	42.14
28	40.130	54.512	75.32	12.86	45.63
29	38.620	28.270	50.88	10.14	44.35
30	32.980	28.571	47.90	12.12	39.56
31	25.920	24.667	42.17	11.68	40.48
32	23.610	46.333	43.78	14.61	37.77
33	19.550	20.343	40.52	15.84	35.99
34	16.640	15.700	35.65	16.47	33.76
35	14.010	13.499	31.60	17.94	32.66
36	12.240	26.741	29.52	20.32	32.60
37	10.810	12.207	26.47	21.40	31.83
38	9.358	9.734	23.28	21.70	31.45
39	8.235	9.084	17.89	18.02	34.06
40	8.124	15.683	13.21	13.61	33.06
41	7.368	12.907	10.72	12.24	32.78
42	6.793	7.464	14.28	18.65	31.88
43	6.119	7.512	21.03	28.81	26.67
44	4.601	11.880	18.03	26.13	27.69
45	4.347	6.789	24.88	38.62	28.57
46	4.072	5.410	19.28	34.78	27.19
47	3.495	5.264	15.33	30.75	26.74
48	3.134	9.608	11.67	26.19	26.05

```
49       2.788          4.837        10.66     27.13             24.76

    Return_On_Investment  Return_On_Assets  Current_Ratio  Gross_Margin  \
0                  31.95             15.98           1.32        55.700
1                  31.91             16.33           1.30        57.170
2                  32.03             15.99           1.13        59.430
3                  29.66             14.98           1.31        59.530
4                  28.54             14.27           1.46        56.120
5                  28.05             13.83           1.24        53.320
6                  27.94             13.95           1.28        50.520
7                  28.21             14.00           1.39        48.350
8                  28.81             14.15           1.39        46.650
9                  29.42             14.31           1.23        45.730
10                 30.76             14.91           1.35        45.220
11                 33.56             16.01           1.31        45.690
12                 36.78             17.44           1.28        47.800
13                 40.87             19.22           1.00        50.680
14                 42.48             19.66           1.11        53.730
15                 42.45             19.74           1.09        53.390
16                 41.58             19.56           1.16        50.740
17                 40.64             19.28           1.13        47.810
18                 36.57             17.85           1.08        44.460
19                 35.78             17.92           1.47        39.510
20                 35.63             18.00           1.63        38.560
21                 35.00             17.92           1.49        37.710
22                 36.05             18.57           1.68        37.030
23                 38.29             19.69           1.88        37.040
24                 42.44             21.74           1.78        37.750
25                 47.94             24.34           1.54        39.670
26                 52.30             26.56           1.50        41.750
27                 55.66             28.22           1.57        41.730
28                 60.04             30.13           1.58        40.130
29                 58.22             28.89           1.58        38.620
30                 51.57             25.62           1.61        32.980
31                 52.34             25.97           1.75        25.920
32                 49.04             24.32           1.93        23.610
33                 46.48             23.46           1.85        19.550
34                 44.30             22.34           2.01        16.640
35                 44.36             21.93           2.31        14.010
36                 45.29             20.93           2.64        12.240
37                 45.45             19.41           2.55        10.810
38                 44.83             18.13           2.74         9.358
39                 48.43             19.08           2.11         8.235
40                 46.26             19.15           2.46         8.124
41                 44.89             19.25           2.38         7.368
42                 43.38             19.06           2.64         6.793
43                 34.15             15.65           3.04         6.119
44                 35.29             16.17           2.78         4.601
```

45	36.57	17.01	2.49	4.347
46	34.28	16.42	2.37	4.072
47	33.76	16.28	2.68	3.495
48	32.54	15.82	2.92	3.134
49	30.85	14.78	2.27	2.788

	Free_Cash_Flow	Gross_Profit_Margin	date
0	32127.0	37.59	2019-06-30
1	23335.0	37.61	2019-03-31
2	64121.0	37.99	2018-12-31
3	47639.0	38.29	2018-09-30
4	36418.0	38.34	2018-06-30
5	25483.0	38.31	2018-03-31
6	51774.0	38.41	2017-12-31
7	39890.0	37.91	2017-09-30
8	33495.0	38.51	2017-06-30
9	23900.0	38.93	2017-03-31
10	53497.0	38.51	2016-12-31
11	40941.0	38.02	2016-09-30
12	33116.0	38.02	2016-06-30
13	23851.0	39.40	2016-03-31
14	70019.0	40.10	2015-12-31
15	60162.0	39.90	2015-09-30
16	47217.0	39.68	2015-06-30
17	30505.0	40.78	2015-03-31
18	50142.0	39.87	2014-12-31
19	40718.0	38.01	2014-09-30
20	32841.0	39.36	2014-06-30
21	20685.0	39.32	2014-03-31
22	45501.0	37.93	2013-12-31
23	37548.0	37.02	2013-09-30
24	31605.0	36.87	2013-06-30
25	21109.0	37.50	2013-03-31
26	42561.0	38.63	2012-12-31
27	36886.0	40.04	2012-09-30
28	28753.0	42.81	2012-06-30
29	16233.0	47.37	2012-03-31
30	33269.0	44.68	2011-12-31
31	24485.0	40.25	2011-09-30
32	14154.0	41.73	2011-06-30
33	8559.0	41.42	2011-03-31
34	16590.0	38.51	2010-12-31
35	11667.0	36.93	2010-09-30
36	7461.0	39.08	2010-06-30
37	5405.0	41.67	2010-03-31
38	9015.0	40.88	2009-12-31
39	6364.0	41.82	2009-09-30
40	4340.0	40.92	2009-06-30

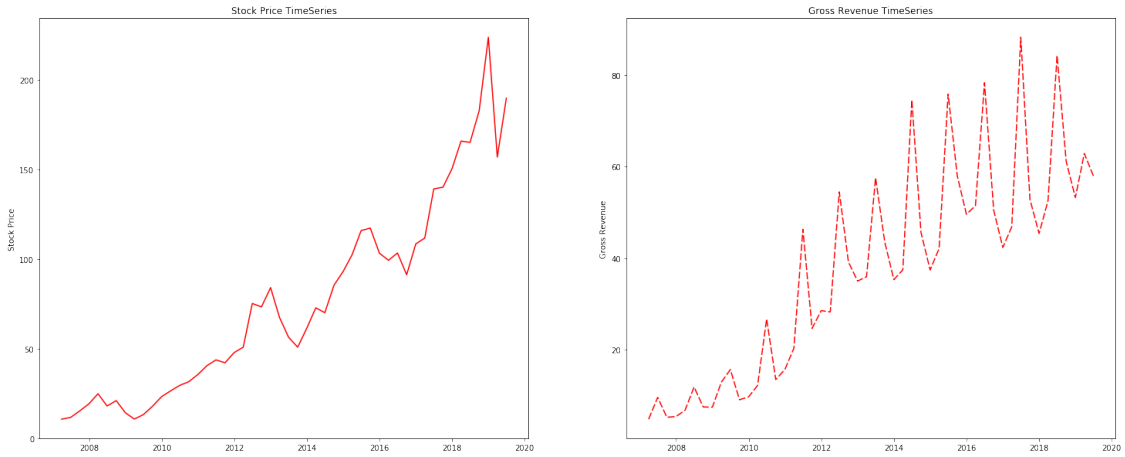
```
41          3599.0              39.93 2009-03-31
42          8505.0              37.94 2008-12-31
43          4613.0              37.12 2008-09-30
44          3596.0              34.83 2008-06-30
45          2563.0              32.93 2008-03-31
46          4735.0              34.68 2007-12-31
47          3244.0              30.76 2007-09-30
48          2300.0              36.88 2007-06-30
49          1671.0              35.13 2007-03-31
```

In [30]:
```
cols = stock_features.columns.tolist()
cols = cols[-1:] + cols[:-1]
stock_features = stock_features[cols]
```

In [31]:
```
fig = plt.figure(figsize=(25,10))
ax1 = fig.add_subplot(1,2,1)
plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Red")
plt.title("Stock Price TimeSeries")
plt.ylabel("Stock Price")


ax2 = fig.add_subplot(1,2,2)
plot2 = ax2.plot(stock_features.date, stock_features.Gross_Revenue, dashes=[6, 2], co
plt.title("Gross Revenue TimeSeries")
plt.ylabel("Gross Revenue")
```

Out[31]:
```
Text(0, 0.5, 'Gross Revenue')
```

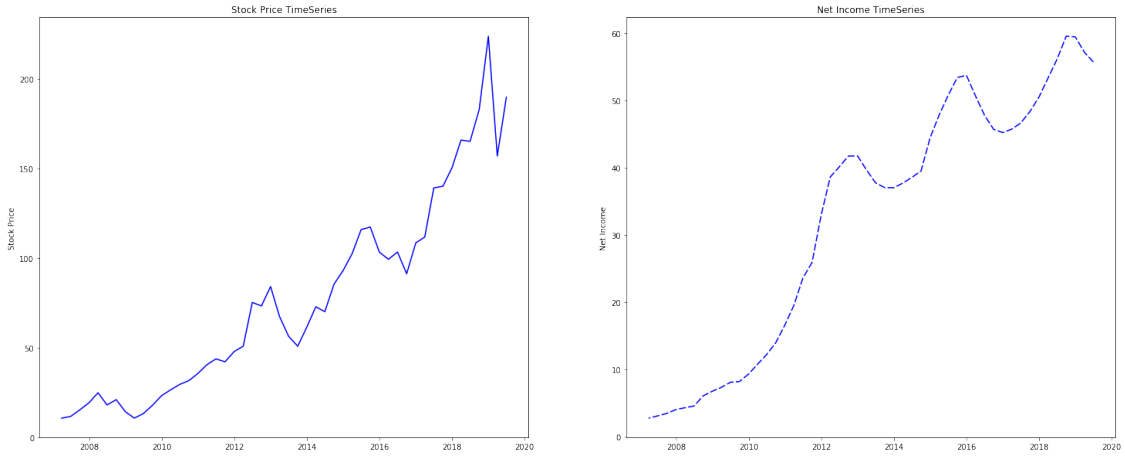


In [32]:
```
fig = plt.figure(figsize=(25,10))
ax1 = fig.add_subplot(1,2,1)
plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Blue")
plt.title("Stock Price TimeSeries")
```

```
plt.ylabel("Stock Price")


ax2 = fig.add_subplot(1,2,2)
plot2 = ax2.plot(stock_features.date, stock_features.Net_Income, dashes=[6, 2], color=
plt.title("Net Income TimeSeries")
plt.ylabel("Net Income")
```

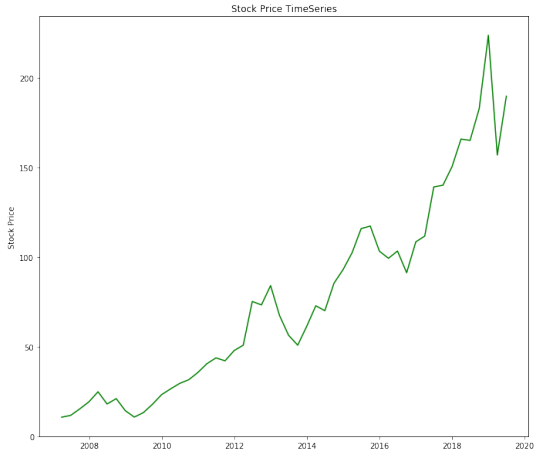
Out[32]: Text(0, 0.5, 'Net Income')

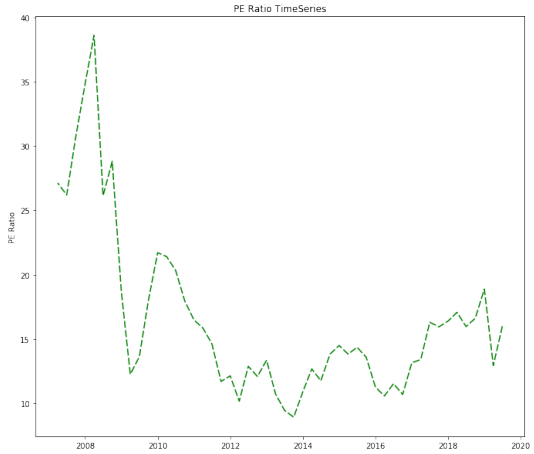


In [33]:
```
fig = plt.figure(figsize=(25,10))
ax1 = fig.add_subplot(1,2,1)
plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Green")
plt.title("Stock Price TimeSeries")
plt.ylabel("Stock Price")


ax2 = fig.add_subplot(1,2,2)
plot2 = ax2.plot(stock_features.date, stock_features.PE_Ratio, dashes=[6, 2], color="
plt.title("PE Ratio TimeSeries")
plt.ylabel("PE Ratio")
```

Out[33]: Text(0, 0.5, 'PE Ratio')

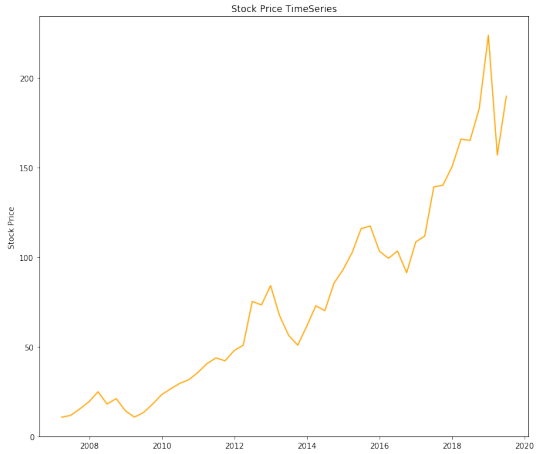





```
In [34]: fig = plt.figure(figsize=(25,10))
         ax1 = fig.add_subplot(1,2,1)
         plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Orange")
         plt.title("Stock Price TimeSeries")
         plt.ylabel("Stock Price")


         ax2 = fig.add_subplot(1,2,2)
         plot2 = ax2.plot(stock_features.date, stock_features.Free_Cash_Flow, dashes=[6, 2], co
         plt.title("FCF TimeSeries")
         plt.ylabel("Free Cash Flow")
```

```
Out[34]: Text(0, 0.5, 'Free Cash Flow')
```

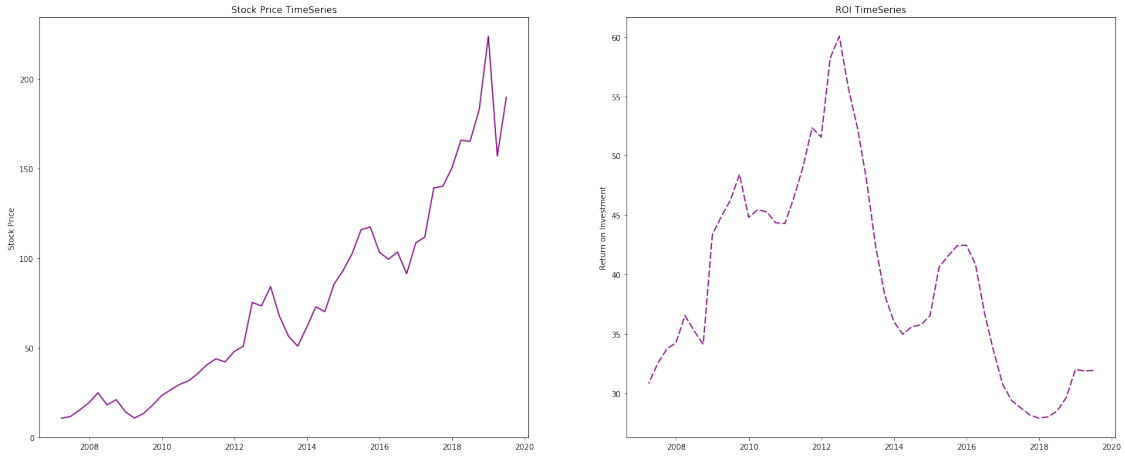



```
In [35]: fig = plt.figure(figsize=(25,10))
         ax1 = fig.add_subplot(1,2,1)
```

```
plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Purple")
plt.title("Stock Price TimeSeries")
plt.ylabel("Stock Price")


ax2 = fig.add_subplot(1,2,2)
plot2 = ax2.plot(stock_features.date, stock_features.Return_On_Investment, dashes=[6,
plt.title("ROI TimeSeries")
plt.ylabel("Return on Investment")
```

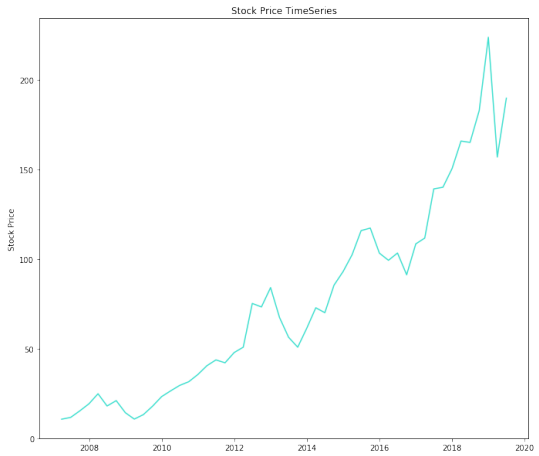
```
Out[35]: Text(0, 0.5, 'Return on Investment')
```

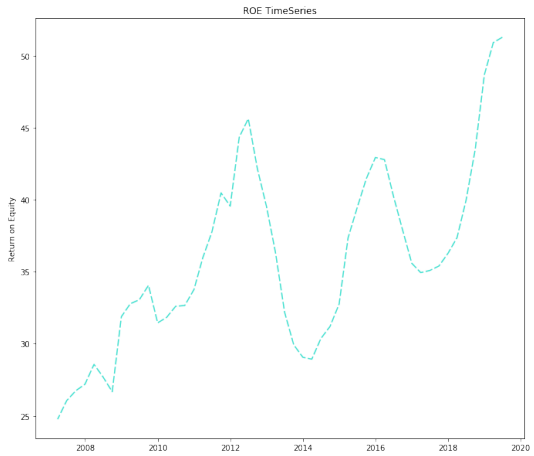



```
In [36]: fig = plt.figure(figsize=(25,10))
         ax1 = fig.add_subplot(1,2,1)
         plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Turquoise")
         plt.title("Stock Price TimeSeries")
         plt.ylabel("Stock Price")


         ax2 = fig.add_subplot(1,2,2)
         plot2 = ax2.plot(stock_features.date, stock_features.Return_On_Equity, dashes=[6, 2],
         plt.title("ROE TimeSeries")
         plt.ylabel("Return on Equity")
```

```
Out[36]: Text(0, 0.5, 'Return on Equity')
```

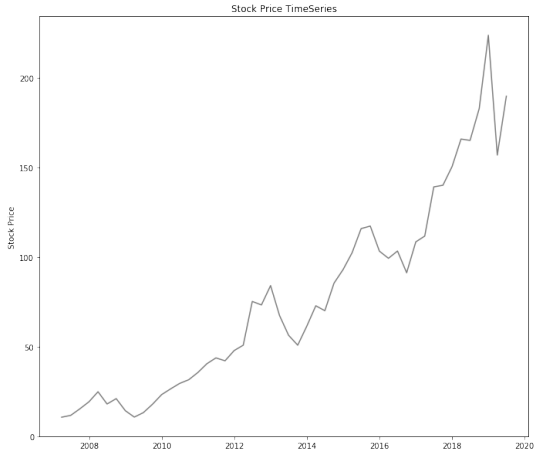


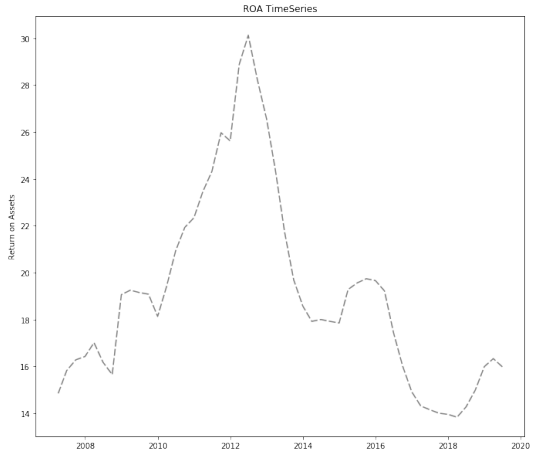


In [37]:
```
fig = plt.figure(figsize=(25,10))
ax1 = fig.add_subplot(1,2,1)
plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Grey")
plt.title("Stock Price TimeSeries")
plt.ylabel("Stock Price")


ax2 = fig.add_subplot(1,2,2)
plot2 = ax2.plot(stock_features.date, stock_features.Return_On_Assets, dashes=[6, 2],
plt.title("ROA TimeSeries")
plt.ylabel("Return on Assets")
```

Out[37]: Text(0, 0.5, 'Return on Assets')



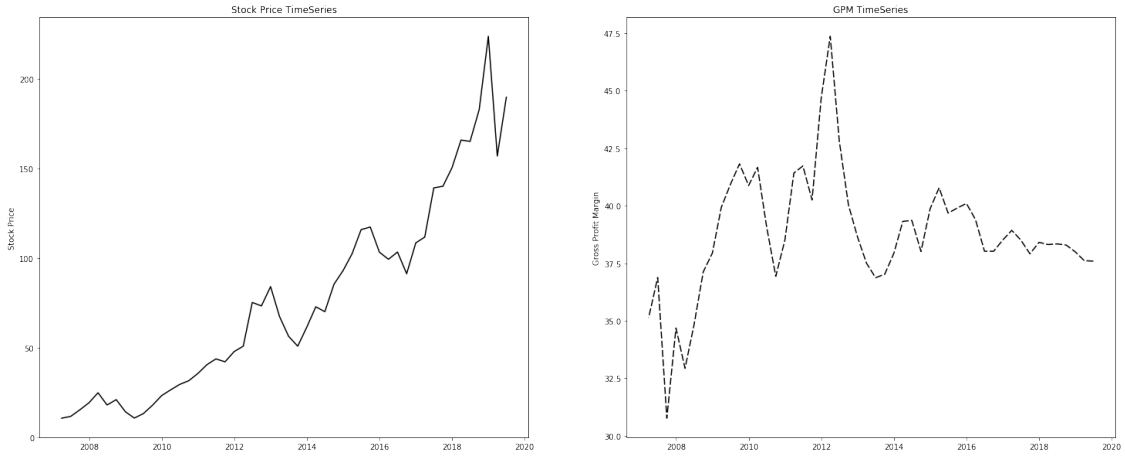


In [38]:
```
fig = plt.figure(figsize=(25,10))
ax1 = fig.add_subplot(1,2,1)
```

```
plot1 = ax1.plot(stock_features.date, stock_features.Stock_Price, color="Black")
plt.title("Stock Price TimeSeries")
plt.ylabel("Stock Price")


ax2 = fig.add_subplot(1,2,2)
plot2 = ax2.plot(stock_features.date, stock_features.Gross_Profit_Margin, dashes=[6, 
plt.title("GPM TimeSeries")
plt.ylabel("Gross Profit Margin")
```

Out[38]: Text(0, 0.5, 'Gross Profit Margin')



DOING REGRESSION ANALYSIS

In [39]:
```
#simple linear regression
import sklearn
from sklearn.model_selection import train_test_split
```

In [40]:
```
stock_features_copy = stock_features.copy()
stock_features_copy = stock_features_copy.drop(columns=['date'])
X = stock_features_copy.drop(columns=['Stock_Price'])
Y = stock_features_copy['Stock_Price']
```

In [41]:
```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
```

In [42]:
```
from sklearn import linear_model
lm = linear_model.LinearRegression()
model = lm.fit(X_train,y_train)
```

In [43]:
```
predictions = lm.predict(X_test)
```

In [44]:
```
lm.score(X_test,y_test)
```

Out[44]: 0.9211021729291289

```
In [45]: #trying ridge regression
         from sklearn.linear_model import Ridge

In [46]: clf = Ridge(alpha=1.0)
         clf.fit(X_train, y_train)

Out[46]: Ridge(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=None,
            normalize=False, random_state=None, solver='auto', tol=0.001)

In [47]: predictions_1 = clf.predict(X_test)

In [48]: clf.score(X_test,y_test)

Out[48]: 0.919775213635698

In [49]: #trying lasso regression
         from sklearn import linear_model

In [50]: clf_2 = linear_model.Lasso(alpha=1.0)
         clf_2.fit(X_train, y_train)

Out[50]: Lasso(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=1000,
            normalize=False, positive=False, precompute=False, random_state=None,
            selection='cyclic', tol=0.0001, warm_start=False)

In [51]: predictions_2 = clf_2.predict(X_test)

In [52]: clf_2.score(X_test,y_test)

Out[52]: 0.9120437609061818

In [53]: #trying elasticnet regression
         from sklearn.linear_model import ElasticNet

In [54]: regr = ElasticNet(random_state=0)
         regr.fit(X_train, y_train)

/Users/saiteja_suvarna/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/coordinate_de
  ConvergenceWarning)


Out[54]: ElasticNet(alpha=1.0, copy_X=True, fit_intercept=True, l1_ratio=0.5,
               max_iter=1000, normalize=False, positive=False, precompute=False,
               random_state=0, selection='cyclic', tol=0.0001, warm_start=False)

In [55]: predictions_3 = regr.predict(X_test)

In [56]: regr.score(X_test, y_test)

Out[56]: 0.9095743355441822
```

```
In [57]: #comparing forecasted data to actual data
         lin_reg_preds = pd.DataFrame({'Actual': y_test, 'Predicted': predictions})
         lin_reg_preds
```

Out[57]:

	Actual	Predicted
18	93.16	93.143166
41	10.72	3.905585
10	108.58	114.021353
24	56.43	60.772413
2	223.99	179.919911
3	183.03	171.421480
16	116.00	116.265164
11	91.33	111.669957
15	117.41	122.680327
25	67.46	62.111016

```
In [58]: ridge_reg_preds = pd.DataFrame({'Actual': y_test, 'Predicted': predictions_1})
         ridge_reg_preds
```

Out[58]:

	Actual	Predicted
18	93.16	94.093639
41	10.72	4.214858
10	108.58	114.280606
24	56.43	60.315703
2	223.99	179.811402
3	183.03	170.674947
16	116.00	116.786213
11	91.33	112.023853
15	117.41	123.041210
25	67.46	61.919699

```
In [59]: lasso_reg_preds = pd.DataFrame({'Actual': y_test, 'Predicted': predictions_2})
         lasso_reg_preds
```

Out[59]:

	Actual	Predicted
18	93.16	94.756134
41	10.72	5.209955
10	108.58	113.888522
24	56.43	61.937330
2	223.99	177.686779
3	183.03	168.447823
16	116.00	116.964973
11	91.33	112.735947
15	117.41	122.619553
25	67.46	64.142629

```
In [60]: elasticnet_reg_preds = pd.DataFrame({'Actual': y_test, 'Predicted': predictions_3})
         elasticnet_reg_preds = elasticnet_reg_preds.reset_index()
         elasticnet_reg_preds
```

Out[60]:

	index	Actual	Predicted
0	18	93.16	95.011994
1	41	10.72	4.970172
2	10	108.58	113.564945
3	24	56.43	62.210396
4	2	223.99	176.811518
5	3	183.03	168.152528
6	16	116.00	117.327310
7	11	91.33	112.485001
8	15	117.41	122.699475
9	25	67.46	64.484423

In [61]:
```
simple_list = []
stock_features['date'].iloc[28]
stock_features['date'].iloc[21]
stock_features['date'].iloc[41]
stock_features['date'].iloc[7]
stock_features['date'].iloc[43]
stock_features['date'].iloc[14]
stock_features['date'].iloc[23]
stock_features['date'].iloc[16]
stock_features['date'].iloc[32]
stock_features['date'].iloc[13]

simple_list.append(stock_features['date'].iloc[28])
simple_list.append(stock_features['date'].iloc[21])
simple_list.append(stock_features['date'].iloc[41])
simple_list.append(stock_features['date'].iloc[7])
simple_list.append(stock_features['date'].iloc[43])
simple_list.append(stock_features['date'].iloc[14])
simple_list.append(stock_features['date'].iloc[23])
simple_list.append(stock_features['date'].iloc[16])
simple_list.append(stock_features['date'].iloc[32])
simple_list.append(stock_features['date'].iloc[13])

simple_list = pd.Series(simple_list)
simple_list

fin_actual_predicted = pd.concat([simple_list, elasticnet_reg_preds], axis=1)
fin_actual_predicted = fin_actual_predicted.drop(columns=['index'])
fin_actual_predicted.columns = ['date', 'Actual','Predicted']
fin_actual_predicted
```

Out[61]:

	date	Actual	Predicted
0	2012-06-30	93.16	95.011994
1	2014-03-31	10.72	4.970172
2	2009-03-31	108.58	113.564945
3	2017-09-30	56.43	62.210396

```
4 2008-09-30  223.99  176.811518
5 2015-12-31  183.03  168.152528
6 2013-09-30  116.00  117.327310
7 2015-06-30   91.33  112.485001
8 2011-06-30  117.41  122.699475
9 2016-03-31   67.46   64.484423
```

In [62]:
```
#Graphing actual vs predicted stock price
ax = sns.lineplot(x='date',y='Actual',data=fin_actual_predicted, color='purple')
ax = sns.lineplot(x='date',y='Predicted',data=fin_actual_predicted, color='red')
plt.xlabel("Date")
plt.ylabel("Price")
plt.title("Actual vs. Predicted Stock Price")
plt.legend(labels=['Actual','Predicted'])
```

Out[62]:
```
<matplotlib.legend.Legend at 0x1a2a7e7dd8>
```