In [207]:
```python
import Plib as pl
import numpy as np
import matplotlib.pyplot as plt
# For making attractive and informative statistical graph
plt.style.use('seaborn-darkgrid')
import pandas as pd
pd.options.mode.chained_assignment = None
```

In [399]:
```python
import calendar
import datetime as dt
from dateutil.relativedelta import relativedelta
from dateutil import parser

def addVix2Future(f):
    VIX2=get_CBOE('VIX')
    VIX2=VIX2[['Close']]
    VIX2.columns=['VIX']
    VIX2 = VIX2.resample('1M', how='mean')
    VIX2['FDate']=getThirdWednesday(2019,1).strftime("%Y-%m-%d")
    for i in range(0,len(VIX2)-1):
        VIX2['FDate'].iloc[i]=getThirdWednesday(int(VIX2.index[i].year),int(VIX2.index[i].month)).strftime("%Y-%m-%d")
    f2=f
    f2['VIX']=0.0
    for i in range(0,len(f2)-1):
        try:
            #print(f2['Future'].iloc[i])
            ret1=VIX2[VIX2['FDate']==f2['Future'].iloc[i].strftime("%Y-%m-%d")]
            #print(float(ret1.iloc[0,0]))
            f2['VIX'].iloc[i]=float(ret1.iloc[0,0])
        except:
            print('')
    f2=f2.set_index('Future')
    return f2

def filterFutCols(futures):
    f1=futures[(futures['Close']!=0) & (futures['Settle']!=0) & (futures['Total Volume']!=0) & (futures['Open Interes
t']!=0)]
    f2=f1.reset_index(drop=True)
    f3=f2.groupby([f2['Group']],as_index=False).mean()
    f3=f3.sort_values(by='Group', ascending=True)
    f3.columns=['Future', 'Open', 'High', 'Low', 'Close', 'Settle', 'Change']
    return f3

def getThirdWednesday(year,month):
    c = calendar.Calendar(firstweekday=calendar.SUNDAY)
    monthcal = c.monthdatescalendar(year,month)
    third_friday = [day for week in monthcal for day in week if day.weekday() == calendar.WEDNESDAY and day.month == m
onth][2]
    return third_friday

def getDatesFutures():
    today=dt.date.today()
    index=0
    df = pd.DataFrame([],columns=['Date'])
    for i in range(-12, 8):
        mydate=today + relativedelta(months=i)
        year=mydate.year
        month=mydate.month
        #print(month,year)
        df.loc[index] = getThirdWednesday(year,month).strftime("%Y-%m-%d")
        index=index+1
    df.sort_values(by='Date', ascending=True)

    return df

def get_CBOE(series):
    url_VVIX='http://www.cboe.com/publish/scheduledtask/mktdata/datahouse/vvixtimeseries.csv'
    url_VIX='http://www.cboe.com/publish/scheduledtask/mktdata/datahouse/vixcurrent.csv'
    url_VIX3='http://www.cboe.com/publish/scheduledtask/mktdata/datahouse/vix3mdailyprices.csv'
    url_VXN='http://www.cboe.com/publish/ScheduledTask/MktData/datahouse/vxncurrent.csv'
    url_SKEW='http://www.cboe.com/publish/scheduledtask/mktdata/datahouse/skewdailyprices.csv'
    url_IC='http://www.cboe.com/publish/scheduledtask/mktdata/datahouse/implied_correlation_hist.csv'
    url_FUTURES='https://markets.cboe.com/us/futures/market_statistics/historical_data/products/csv/VX/'

    if series=='FUTURES':
        data = pd.DataFrame([],columns=['Trade Date', 'Futures', 'Open', 'High', 'Low', 'Close', 'Settle',
       'Change', 'Total Volume', 'EFP', 'Open Interest','Group'])
        md=pd.DataFrame(getDatesFutures())
        md['Date']=pd.to_datetime(md['Date'])
        index=0
        for i in range(0, len(md)):
            try:
                url=url_FUTURES + str(md.iloc[i].Date.date())
                data2 = pd.read_csv(url)
                data2['Group']=md.iloc[i].Date.date()
                data=data.append(data2)
                index=index+1
            except:
                print('Downloaded all data...')
    elif series=='VVIX':
        url=url_VVIX
        data = pd.read_csv(url)
        data2=data
        data2['Date']=pd.to_datetime(data2.iloc[1:,0])
        data2['VVIX']=data2.iloc[1:,1]
        data=data2[['Date','VVIX']]
        data['VVIX']=data2['VVIX'].astype(float)
        data=data.set_index('Date')
        data=data.dropna()
    elif series=='IC':
        #DATE,SPX,VIX,(JCJ) January 2008,(KCJ) January 2009,(ICJ) January 2010,(JCJ)
        url=url_IC
        data = pd.read_csv(url)
        data2=data
        data2['Date']=data.iloc[2:,0]
        data2['Date']=pd.to_datetime(data2['Date'])
        data2['SPX']=data.iloc[2:,1]
        data2['VIX']=data.iloc[2:,2]
        data2['JCJ1']=data.iloc[2:,3]
        data2['KCJ1']=data.iloc[2:,4]
        data2['ICJ1']=data.iloc[2:,5]
        data2['JCJ2']=data.iloc[2:,6]
        data2['KCJ2']=data.iloc[2:,7]
        data2['ICJ2']=data.iloc[2:,8]
        data2['JCJ3']=data.iloc[2:,9]
        data2['KCJ3']=data.iloc[2:,10]
        data2['ICJ3']=data.iloc[2:,11]
        data2['JCJ4']=data.iloc[2:,12]
        data2['KCJ4']=data.iloc[2:,13]
        data2['ICJ4']=data.iloc[2:,14]
        data2['JCJ5']=data.iloc[2:,15]
        data2['KCJ5']=data.iloc[2:,16]
        data2=data2.replace(r'^\s*$', np.nan, regex=True)
        data3=data2[['Date','JCJ1','KCJ1','ICJ1','JCJ2','KCJ2','ICJ2','JCJ3','KCJ3','ICJ3','JCJ4','KCJ4','ICJ4','JCJ
5','KCJ5']]
        data3=data3.set_index('Date')
        data3['JCJ1']=data3.JCJ1.astype(float)
        data3['KCJ1']=data3.KCJ1.astype(float)
        data3['ICJ1']=data3.ICJ1.astype(float)
        data3['JCJ2']=data3.JCJ2.astype(float)
        data3['KCJ2']=data3.KCJ2.astype(float)
        data3['ICJ2']=data3.ICJ2.astype(float)
        data3['JCJ3']=data3.JCJ3.astype(float)
        data3['KCJ3']=data3.KCJ3.astype(float)
        data3['ICJ3']=data3.ICJ3.astype(float)
        data3['JCJ4']=data3.JCJ4.astype(float)
        data3['KCJ4']=data3.KCJ4.astype(float)
        data3['ICJ4']=data3.ICJ4.astype(float)
        data3['JCJ5']=data3.JCJ5.astype(float)
        data3['KCJ5']=data3.KCJ5.astype(float)
        data3=pd.DataFrame(data3.mean(axis=1))
        data3['Corr']=data3.iloc[:,0]
        data3=data3[['Corr']]

        data=data2[['Date','SPX','VIX']]
        data=data.merge(data3,left_on='Date',right_on='Date',how='inner')
        data['SPX']=data.SPX.astype(float)
        data['VIX']=data.VIX.astype(float)
        data=data.set_index('Date')
        data=data.dropna()
    elif series=='SKEW':
        url=url_SKEW
        data = pd.read_csv(url)
        data2=data
        data2['Date']=pd.to_datetime(data2.iloc[1:,0])
        data2['SKEW']=data2.iloc[1:,1]
        data=data2[['Date','SKEW']]
        data['SKEW']=data2['SKEW'].astype(float)
        data=data.set_index('Date')
        data=data.dropna()
    elif series=='VIX':
        url=url_VIX
        data = pd.read_csv(url)
        data2=data
        #print(data2)
        data2['Date']=data.iloc[2:,0]
        data2['Date']=pd.to_datetime(data2['Date'])
        data2['Open']=data.iloc[2:,1]
        data2['High']=data.iloc[2:,2]
        data2['Low']=data.iloc[2:,3]
        data2['Close']=data.iloc[2:,4]
        data=data2[['Date','Open','High','Low','Close']]
        data['Open']=data.Open.astype(float)
        data['High']=data.High.astype(float)
        data['Low']=data.Low.astype(float)
        data['Close']=data.Close.astype(float)
        data=data.set_index('Date')
        data=data.dropna()
    elif series=='VIX3':
        url=url_VIX3
        data = pd.read_csv(url)
        data2=data
        data2['Date']=data.iloc[2:,0]
        data2['Date']=pd.to_datetime(data2['Date'])
        data2['Open']=data.iloc[2:,1]
        data2['High']=data.iloc[2:,2]
        data2['Low']=data.iloc[2:,3]
        data2['Close']=data.iloc[2:,4]
        data=data2[['Date','Open','High','Low','Close']]
        data['Open']=data.Open.astype(float)
        data['High']=data.High.astype(float)
        data['Low']=data.Low.astype(float)
        data['Close']=data.Close.astype(float)
        data=data.set_index('Date')
        data=data.dropna()
    elif series=='VXN':
        url=url_VXN
        data = pd.read_csv(url)
        data2=data
        data2['Date']=data.iloc[2:,0]
        data2['Date']=pd.to_datetime(data2['Date'])
        data2['Open']=data.iloc[2:,1]
        data2['High']=data.iloc[2:,2]
        data2['Low']=data.iloc[2:,3]
        data2['Close']=data.iloc[2:,4]
        data=data2[['Date','Open','High','Low','Close']]
        data['Open']=data.Open.astype(float)
        data['High']=data.High.astype(float)
        data['Low']=data.Low.astype(float)
        data['Close']=data.Close.astype(float)
        data=data.set_index('Date')
        data=data.dropna()
    return data
```

In [400]:
```python
api = pl.EODH()
dt_start="2007-01-01"
dt_end="2019-01-01"
days=30
tick='GSPC'
data = pd.DataFrame(api.get_eod_data_with_date(tick,'INDX', dt_start, dt_end).interpolate())
data.head()

returns = np.log(data/data.shift(1)).dropna()
returns.head()
```

Out[400]:

Date	Open	High	Low	Close	Adjusted_close	Volume
2007-01-04	-0.001009	-0.005317	0.000405	0.001228	0.001228	-0.132217
2007-01-05	0.001228	-0.002465	-0.001905	-0.006103	-0.006103	-0.028720
2007-01-08	-0.006422	-0.002372	-0.001267	0.002218	0.002218	-0.054938
2007-01-09	0.002537	0.000445	0.001032	-0.000517	-0.000517	0.094884
2007-01-10	-0.002935	0.000268	-0.000071	0.001938	0.001938	-0.094407

In [401]:
```python
VIX=get_CBOE('VIX')
VIX.head()
```

Out[401]:

Date	Open	High	Low	Close
2004-01-05	18.45	18.49	17.44	17.49
2004-01-06	17.66	17.67	16.19	16.73
2004-01-07	16.72	16.75	15.50	15.50
2004-01-08	15.42	15.68	15.32	15.61
2004-01-09	16.15	16.88	15.57	16.75

In [402]:
```python
VVIX=get_CBOE('VVIX')
VVIX.head()
```

Out[402]:

Date	VVIX
2007-01-03	87.63
2007-01-04	88.19
2007-01-05	90.17
2007-01-08	92.04
2007-01-09	92.76

In [403]:
```python
R=VIX.merge(VVIX,left_on='Date',right_on='Date',how='inner')
R['Ratio']=R['VVIX']/R['Close']
R.head()
```

Out[403]:

Date	Open	High	Low	Close	VVIX	Ratio
2007-01-03	12.16	12.75	11.53	12.04	87.63	7.278239
2007-01-04	12.40	12.42	11.28	11.51	88.19	7.662033
2007-01-05	11.84	12.25	11.68	12.14	90.17	7.427512
2007-01-08	12.48	12.83	11.78	12.00	92.04	7.670000
2007-01-09	11.86	12.47	11.69	11.91	92.76	7.788413

In [404]:
```python
Corr=get_CBOE('IC')
Corr.head()
```

Out[404]:

Date	SPX	VIX	Corr
2007-01-04	1418.34	11.51	41.755
2007-01-05	1409.71	12.14	42.915
2007-01-08	1412.84	12.00	40.915
2007-01-09	1412.11	11.91	41.555
2007-01-10	1414.85	11.47	40.160

In [405]:
```python
SK=get_CBOE('SKEW')
SK.head()
```

Out[405]:

Date	SKEW
1990-01-02	126.09
1990-01-03	123.34
1990-01-04	122.62
1990-01-05	121.27
1990-01-08	124.12

In [406]:
```python
futures=get_CBOE('FUTURES')
f=filterFutCols(futures)
f
```

```
Downloaded all data...
Downloaded all data...
```

Out[406]:

	Future	Open	High	Low	Close	Settle	Change
0	2018-09-19	16.330164	16.652623	16.050929	16.304536	16.305546	-0.024508
1	2018-10-17	16.757219	17.189037	16.461016	16.765455	16.761257	0.007380
2	2018-11-21	17.127240	17.521667	16.826458	17.135729	17.144896	0.022370
3	2018-12-19	17.300374	17.694866	17.016043	17.330695	17.332353	0.041176
4	2019-01-16	17.948641	18.344674	17.650163	17.956739	17.945897	0.003913
5	2019-02-20	17.912857	18.026429	17.694286	17.827857	17.691071	-0.194643
6	2019-04-17	17.598128	17.910802	17.328128	17.567326	17.556070	-0.029626
7	2019-05-15	17.006000	17.575000	16.401000	16.696000	16.790000	0.447500
8	2019-06-19	17.833261	18.153913	17.556630	17.808478	17.805027	-0.012908
9	2019-07-17	17.765659	18.027308	17.506044	17.728022	17.731813	-0.027253
10	2019-08-21	17.745464	18.078251	17.432459	17.720219	17.703033	-0.021694
11	2019-09-18	17.753488	18.027384	17.452674	17.713023	17.711047	-0.011919
12	2019-10-16	17.589272	17.835762	17.338477	17.570927	17.573013	-0.007947
13	2019-11-20	17.531929	17.748357	17.317929	17.519071	17.514464	-0.000714
14	2019-12-18	17.322500	17.523966	17.112069	17.309914	17.307543	0.001940
15	2020-02-19	17.994242	18.204697	17.807424	18.001212	17.994697	0.007576
16	2020-03-18	18.069333	18.293778	17.859778	18.080889	18.083333	0.025000
17	2020-04-15	18.480312	18.715000	18.274062	18.472812	18.425781	0.034375

In [407]:
```python
f2=addVix2Future(f)
f2
```

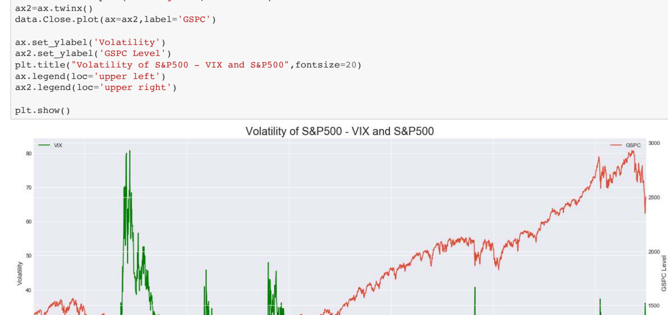
```
/miniconda3/lib/python3.7/site-packages/ipykernel_launcher.py:10: FutureWarning: how in .resample() is deprecated
the new syntax is .resample(...).mean()
  # Remove the CWD from sys.path while we load stuff.
```

Out[407]:

Future	Open	High	Low	Close	Settle	Change	VIX
2018-09-19	16.330164	16.652623	16.050929	16.304536	16.305546	-0.024508	12.910526
2018-10-17	16.757219	17.189037	16.461016	16.765455	16.761257	0.007380	19.352174
2018-11-21	17.127240	17.521667	16.826458	17.135729	17.144896	0.022370	19.389048
2018-12-19	17.300374	17.694866	17.016043	17.330695	17.332353	0.041176	24.953158
2019-01-16	17.948641	18.344674	17.650163	17.956739	17.945897	0.003913	19.572381
2019-02-20	17.912857	18.026429	17.694286	17.827857	17.691071	-0.194643	15.234737
2019-04-17	17.598128	17.910802	17.328128	17.567326	17.556070	-0.029626	12.949048
2019-05-15	17.006000	17.575000	16.401000	16.696000	16.790000	0.447500	16.721818
2019-06-19	17.833261	18.153913	17.556630	17.808478	17.805027	-0.012908	15.836000
2019-07-17	17.765659	18.027308	17.506044	17.728022	17.731813	-0.027253	13.305909
2019-08-21	17.745464	18.078251	17.432459	17.720219	17.703033	-0.021694	18.979091
2019-09-18	17.753488	18.027384	17.452674	17.713023	17.711047	-0.011919	0.000000
2019-10-16	17.589272	17.835762	17.338477	17.570927	17.573013	-0.007947	0.000000
2019-11-20	17.531929	17.748357	17.317929	17.519071	17.514464	-0.000714	0.000000
2019-12-18	17.322500	17.523966	17.112069	17.309914	17.307543	0.001940	0.000000
2020-02-19	17.994242	18.204697	17.807424	18.001212	17.994697	0.007576	0.000000
2020-03-18	18.069333	18.293778	17.859778	18.080889	18.083333	0.025000	0.000000
2020-04-15	18.480312	18.715000	18.274062	18.472812	18.425781	0.034375	0.000000

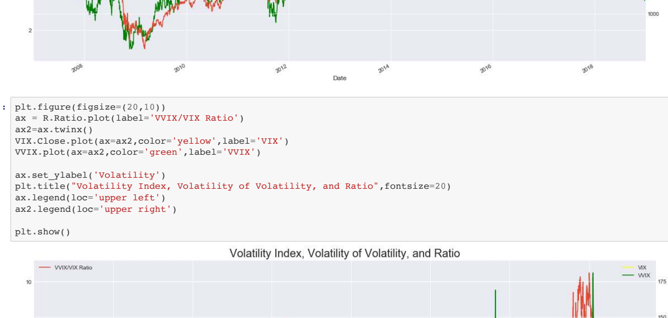
In [ ]:

In [408]:
```python
plt.figure(figsize=(20,10))
ax = Corr.Corr.plot(color='green',label='Implied Correlation')
ax2=ax.twinx()
data.Close.plot(ax=ax2,label='GSPC')

ax.set_ylabel('Implied Correlation')
ax2.set_ylabel('GSPC Level')
plt.title("Implied Correlation of S&P500",fontsize=20)
ax.legend(loc='upper left')
ax2.legend(loc='upper right')

plt.show()
```



In [409]:
```python
plt.figure(figsize=(20,10))
ax = VIX.Close.plot(color='green',label='VIX')
ax2=ax.twinx()
data.Close.plot(ax=ax2,label='GSPC')

ax.set_ylabel('Volatility')
ax2.set_ylabel('GSPC Level')
plt.title("Volatility of S&P500 - VIX and S&P500",fontsize=20)
ax.legend(loc='upper left')
ax2.legend(loc='upper right')

plt.show()
```

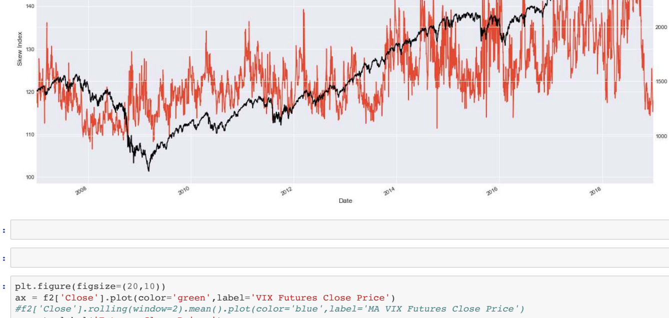
In [410]:
```python
#R.Ratio.plot(figsize=(20,10))
plt.figure(figsize=(20,10))
ax = R.Ratio.plot(color='green',label='VVIX/VIX Ratio')
ax2=ax.twinx()
data.Close.plot(ax=ax2,label='GSPC')

ax.set_ylabel('Ratio')
ax2.set_ylabel('GSPC Level')
plt.title("Volatility Ratio VVIX/VIX and S&P500",fontsize=20)
ax.legend(loc='upper left')
ax2.legend(loc='upper right')

plt.show()
```



In [411]:
```python
plt.figure(figsize=(20,10))
ax = R.Ratio.plot(label='VVIX/VIX Ratio')
ax2=ax.twinx()
VIX.Close.plot(ax=ax2,color='yellow',label='VIX')
VVIX.plot(ax=ax2,color='green',label='VVIX')

ax.set_ylabel('Volatility')
plt.title("Volatility Index, Volatility of Volatility, and Ratio",fontsize=20)
ax.legend(loc='upper left')
ax2.legend(loc='upper right')

plt.show()
```

In [ ]:

In [412]:
```python
plt.figure(figsize=(20,10))
ax = SK.SKEW.plot(label='SKEW Index')
ax.set_ylabel('Skew Index')
ax2=ax.twinx()
data.Close.plot(ax=ax2,color='black',label='GSPC')
plt.title("Skew Index and S&P500",fontsize=20)
ax.legend(loc='upper left')
ax2.legend(loc='upper right')

plt.show()
```



In [ ]:

In [ ]:

In [415]:
```python
plt.figure(figsize=(20,10))
ax = f2['Close'].plot(color='green',label='VIX Futures Close Price')
#f2['Close'].rolling(window=2).mean().plot(color='blue',label='MA VIX Futures Close Price')
ax.set_ylabel('Futures Close Prices')
ax2=ax.twinx()
f2.VIX.plot(ax=ax2,color='yellow',label='VIX')
plt.title("VIX Futures Curve against VIX",fontsize=20)
ax.legend(loc='upper left')
ax2.legend(loc='upper right')
```

Out[415]: <matplotlib.legend.Legend at 0x1a25795978>

In [ ]:

In [ ]:

In [ ]:

In [ ]: