In [1]:

```
import pandas as pd
import numpy as np
from datetime import timedelta
from datetime import datetime
from datetime import *
```

In [2]:

```
# Reading data from csv while parsing date column as datetime and index
mydateparser = lambda x: pd.datetime.strptime(x, "%d-%b-%y")
stockData_df = pd.read_csv('/Users/psai1/spx.csv', infer_datetime_format=True, \
                          parse_dates=['date'], date_parser=mydateparser, index_col ="da
```

```
<ipython-input-2-9622e21e175a>:2: FutureWarning: The pandas.datetime class
is deprecated and will be removed from pandas in a future version. Import
from datetime module instead.
  mydateparser = lambda x: pd.datetime.strptime(x, "%d-%b-%y")
```

In [3]:

```
stockData_df.sample(5)
```

Out[3]:

	close
date	
2004-05-19	1088.68
2013-07-01	1614.96
1992-03-16	406.39
2013-10-10	1692.56
2000-04-14	1356.56

In [4]:

```
stockData_df.shape
```

Out[4]:

```
(8192, 1)
```

In [5]:

```
# Taking latest complete one year data to ease coding
stockData_2017_df = stockData_df[(stockData_df.index.year == 2017)]
```

In [6]:

```
stockData_2017_df.shape
print(stockData_2017_df)
```

```
              close
date
2017-01-03  2257.83
2017-01-04  2270.75
2017-01-05  2269.00
2017-01-06  2276.98
2017-01-09  2268.90
...             ...
2017-12-22  2683.34
2017-12-26  2680.50
2017-12-27  2682.62
2017-12-28  2687.54
2017-12-29  2673.61

[251 rows x 1 columns]
```

In [7]:

```
#Check if the date is in dataframe or not
def Get_Valid_Date(given_date,df):
    try:
        giv_date=str(given_date)
        year=giv_date[0:4]
        month=giv_date[5:7]
        day=giv_date[8:10]
        current_date=datetime(int(year),int(month),int(day))
        cur_date=current_date.date()
        condition=given_date in df.index
        if cur_date>date(2017, 12, 29):
            return date(2017, 12, 29)
        elif cur_date<date(2017, 1, 3):
            return date(2017, 1, 3)
        elif condition==True:
            return cur_date
        else:
            reduced_date = current_date - timedelta(days=1)
            return Get_Valid_Date(reduced_date,df)
    except ValueError:
        giv_date=str(given_date)
        year=giv_date[0:4]
        month=giv_date[5:7]
        int_day=int(giv_date[8:10])-1
        day=str(int_day)
        present_date=year+"-"+month+"-"+day
        return Get_Valid_Date(present_date,df)
```

In [8]:

```
checked_date=Get_Valid_Date("2017-03-03",stockData_2017_df)
checked_date
```

Out[8]:

```
datetime.date(2017, 3, 3)
```

In [9]:

```
# Get closing value for the given date
def Get_closing_Value(given_date,df):
    result_date=str(Get_Valid_Date(given_date,df))
    day_close=df.loc[result_date,["close"]]
    return day_close
```

In [10]:

```
Get_closing_Value("2017-03-03",stockData_2017_df)
```

Out[10]:

```
close    2383.12
Name: 2017-03-03 00:00:00, dtype: float64
```

In [11]:

```
# Calculate percentage growth when given day closing and dataframe
def Cal_per_growth(d_close, df):
    final_close=df.iloc[-1]
    d_close_value=Get_closing_Value(d_close,df)
    return ((final_close-d_close_value)/d_close_value)*100
```

In [12]:

```
Cal_per_growth("2017-03-03",stockData_2017_df)
```

Out[12]:

```
close    12.189483
dtype: float64
```

In [13]:

```
# Calculate average growth of a given day of the month in the given dataframe
def Cal_avg_growth(day_of_month, df):
    sum=0
    for i in range(1,10):
        if day_of_month<10:
            loop_date="2017-0"+str(i)+"-0"+str(day_of_month)
            loop_avg_growth=Cal_per_growth(loop_date,df)
            sum=sum+loop_avg_growth
        else :
            loop_date="2017-0"+str(i)+"-"+str(day_of_month)
            loop_avg_growth=Cal_per_growth(loop_date,df)
            sum=sum+loop_avg_growth
    for i in range(10,13):
        if day_of_month<10:
            loop_date="2017-"+str(i)+"-0"+str(day_of_month)
            loop_avg_growth=Cal_per_growth(loop_date,df)
            sum=sum+loop_avg_growth
        else :
            loop_date="2017-"+str(i)+"-"+str(day_of_month)
            loop_avg_growth=Cal_per_growth(loop_date,df)
            sum=sum+loop_avg_growth

    avg_percent_growth_one_day_full_year=sum/12
    return avg_percent_growth_one_day_full_year
```

In [15]:

```
Cal_avg_growth(30,stockData_2017_df)
```

Out[15]:

```
close    8.705487
dtype: float64
```

In [16]:

```
def Cal_avg_growth_allDays(df):
    final= pd.DataFrame({'day': [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,2
    final.set_index('day')
    result=[]
    for i in range(1,32):
        result.append(Cal_avg_growth(i, df))
    final["Average Growth"]=result
    return final
```

In [17]:

```
Result_table=Cal_avg_growth_allDays(stockData_2017_df)
Result_table
```

Out[17]:

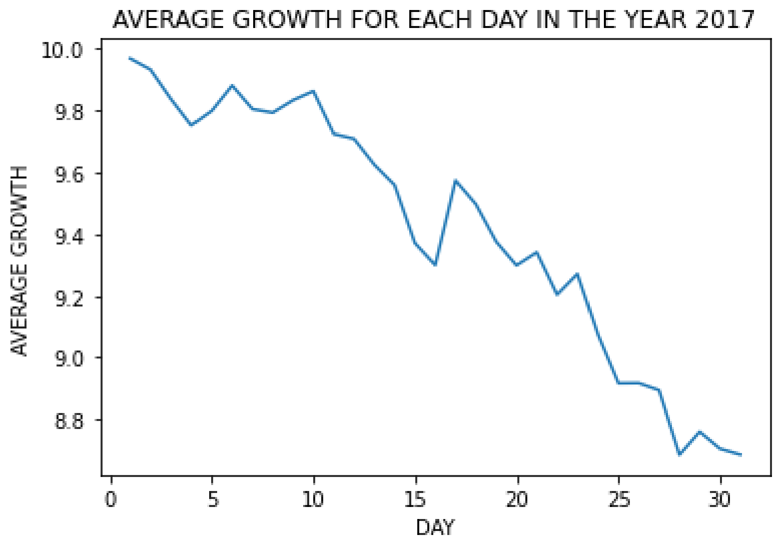
	day	Average Growth
0	1	close 9.968917 dtype: float64
1	2	close 9.933011 dtype: float64
2	3	close 9.838061 dtype: float64
3	4	close 9.752149 dtype: float64
4	5	close 9.799334 dtype: float64
5	6	close 9.881239 dtype: float64
6	7	close 9.804748 dtype: float64
7	8	close 9.793305 dtype: float64
8	9	close 9.833718 dtype: float64
9	10	close 9.862929 dtype: float64
10	11	close 9.724863 dtype: float64
11	12	close 9.709523 dtype: float64
12	13	close 9.625768 dtype: float64
13	14	close 9.559391 dtype: float64
14	15	close 9.37136 dtype: float64
15	16	close 9.299863 dtype: float64
16	17	close 9.574407 dtype: float64
17	18	close 9.497625 dtype: float64
18	19	close 9.376127 dtype: float64
19	20	close 9.299081 dtype: float64
20	21	close 9.341407 dtype: float64
21	22	close 9.203995 dtype: float64
22	23	close 9.271228 dtype: float64
23	24	close 9.073166 dtype: float64
24	25	close 8.916483 dtype: float64
25	26	close 8.916953 dtype: float64
26	27	close 8.894166 dtype: float64
27	28	close 8.685668 dtype: float64
28	29	close 8.760497 dtype: float64
29	30	close 8.705487 dtype: float64
30	31	close 8.686527 dtype: float64

In [28]:

```
import matplotlib.pyplot as plt
import seaborn as sn
%matplotlib inline
```

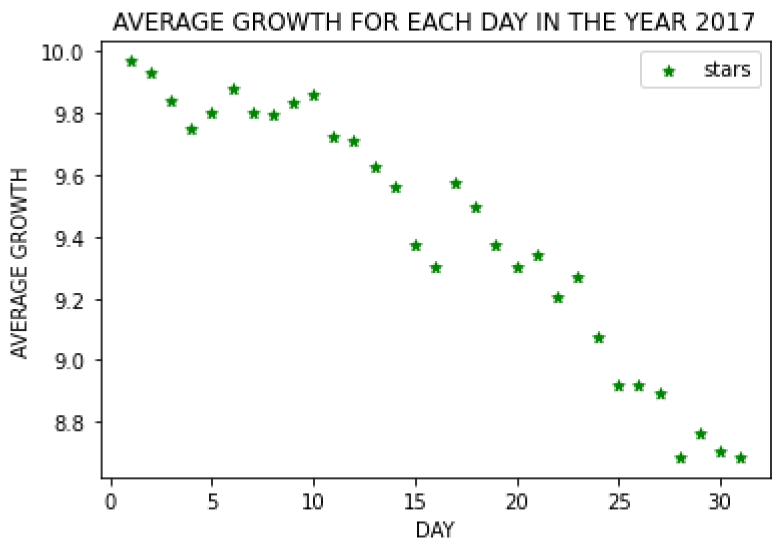
In [31]:

```
x=Result_table["day"]
y=Result_table["Average Growth"]
plt.plot(x, y)
plt.xlabel('DAY')
plt.ylabel('AVERAGE GROWTH')
plt.title('AVERAGE GROWTH FOR EACH DAY IN THE YEAR 2017')
plt.show()
```



In [33]:

```
x=Result_table["day"]
y=Result_table["Average Growth"]
plt.scatter(x, y, label= "stars", color= "green", marker= "*", s=31)
plt.xlabel('DAY')
plt.ylabel('AVERAGE GROWTH')
plt.title('AVERAGE GROWTH FOR EACH DAY IN THE YEAR 2017')
plt.legend()
plt.show()
```



In [17]:

```
Result_table.to_csv('Stocks_data.csv', header=True, index=False)
```

In [1]:

```
import numpy as np
import pandas as pd
```

Loading the data

In [2]:

```
df = pd.read_csv("stock headlines.csv", encoding = 'ISO-8859-1')
```

Exploring the dataset

In [3]:

```
df.columns
```

Out[3]:

```
Index(['Date', 'Label', 'Top1', 'Top2', 'Top3', 'Top4', 'Top5', 'Top6', 'T
op7',
       'Top8', 'Top9', 'Top10', 'Top11', 'Top12', 'Top13', 'Top14', 'Top1
5',
       'Top16', 'Top17', 'Top18', 'Top19', 'Top20', 'Top21', 'Top22', 'Top
23',
       'Top24', 'Top25'],
      dtype='object')
```

In [4]:

```
df.shape
```

Out[4]:

```
(4101, 27)
```

In [5]:

```
df.head(3)
```

Out[5]:

	Date	Label	Top1	Top2	Top3	Top4	Top5	Top6	Top
0	2000-01-03	0	A 'hindrance to operations': extracts from the...	Scorecard	Hughes' instant hit buoys Blues	Jack gets his skates on at ice-cold Alex	Chaos as Maracana builds up for United	Depleted Leicester prevail as Elliott spoils E...	Hungr Spur sens ric picking
1	2000-01-04	0	Scorecard	The best lake scene	Leader: German sleaze inquiry	Cheerio, boyo	The main recommendations	Has Cubie killed fees?	Ha Cubi kille fees
2	2000-01-05	0	Coventry caught on counter by Flo	United's rivals on the road to Rio	Thatcher issues defence before trial by video	Police help Smith lay down the law at Everton	Tale of Trautmann bears two more retellings	England on the rack	Pakista retaliat with ca for vide c Walsl

3 rows × 27 columns

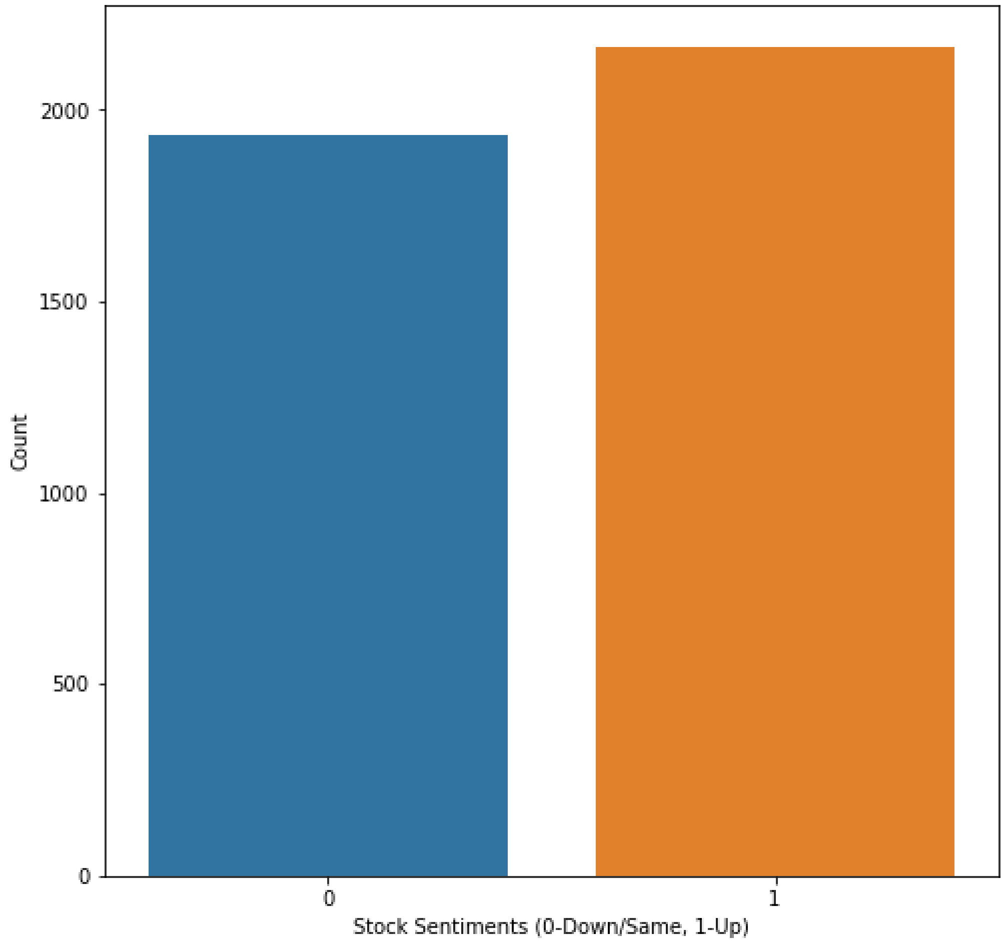
Note: Here 'Label' is a binary attribute which consists 0 - Stock price goes down or stays the same, 1 - Stock price goes up.

In [6]:

```
# Importing essential libraries for visualization
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [7]:

```
# Visualizing the count of 'Label' column from the dataset
plt.figure(figsize=(8,8))
sns.countplot(x='Label', data=df)
plt.xlabel('Stock Sentiments (0-Down/Same, 1-Up)')
plt.ylabel('Count')
plt.show()
```



Data Cleaning and Preprocessing

In [8]:

```
print(df.shape)
```

```
(4101, 27)
```

In [9]:

```
# Finding any NaN values
df.isna().any()
```

Out[9]:

```
Date      False
Label     False
Top1      False
Top2      False
Top3      False
Top4      False
Top5      False
Top6      False
Top7      False
Top8      False
Top9      False
Top10     False
Top11     False
Top12     False
Top13     False
Top14     False
Top15     False
Top16     False
Top17     False
Top18     False
Top19     False
Top20     False
Top21     False
Top22     False
Top23      True
Top24      True
Top25      True
dtype: bool
```

In [10]:

```
# Dropping NaN values
df.dropna(inplace=True)
print(df.shape)
```

```
(4098, 27)
```

In [11]:

```
df_copy = df.copy()
```

In [12]:

```
df_copy.reset_index(inplace=True)
```

In [13]:

```
df.head()
```

Out[13]:

	Date	Label	Top1	Top2	Top3	Top4	Top5	Top6	'
0	2000-01-03	0	A 'hindrance to operations': extracts from the...	Scorecard	Hughes' instant hit buoys Blues	Jack gets his skates on at ice-cold Alex	Chaos as Maracana builds up for United	Depleted Leicester prevail as Elliott spoils E...	Hu S s picl
1	2000-01-04	0	Scorecard	The best lake scene	Leader: German sleaze inquiry	Cheerio, boyo	The main recommendations	Has Cubie killed fees?	C I f
2	2000-01-05	0	Coventry caught on counter by Flo	United's rivals on the road to Rio	Thatcher issues defence before trial by video	Police help Smith lay down the law at Everton	Tale of Trautmann bears two more retellings	England on the rack	Pak retε witl for \ of W
3	2000-01-06	1	Pilgrim knows how to progress	Thatcher facing ban	McIlroy calls for Irish fighting spirit	Leicester bin stadium blueprint	United braced for Mexican wave	Auntie back in fashion, even if the dress look...	Sh ap go th
4	2000-01-07	1	Hitches and Horlocks	Beckham off but United survive	Breast cancer screening	Alan Parker	Guardian readers: are you all whingers?	Hollywood Beyond	A diam

5 rows × 27 columns

In [14]:

```
# Splitting the dataset into train an test set
train = df_copy[df_copy['Date'] < '20150101']
test = df_copy[df_copy['Date'] > '20141231']
print('Train size: {}, Test size: {}'.format(train.shape, test.shape))
```

```
Train size: (3972, 28), Test size: (378, 28)
```

In [15]:

```
train.columns
```

Out[15]:

```
Index(['index', 'Date', 'Label', 'Top1', 'Top2', 'Top3', 'Top4', 'Top5',
       'Top6', 'Top7', 'Top8', 'Top9', 'Top10', 'Top11', 'Top12', 'Top13',
       'Top14', 'Top15', 'Top16', 'Top17', 'Top18', 'Top19', 'Top20', 'Top
21',
       'Top22', 'Top23', 'Top24', 'Top25'],
      dtype='object')
```

In [16]:

```
# Splitting the dataset
y_train = train['Label']
train = train.iloc[:, 3:28]
y_test = test['Label']
test = test.iloc[:, 3:28]
```

In [17]:

```
# Importing essential libraries for performing Natural Language Processing on given data
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
```

```
[nltk_data] Error loading stopwords: <urlopen error [WinError 10060] A
[nltk_data]     connection attempt failed because the connected party
[nltk_data]     did not properly respond after a period of time, or
[nltk_data]     established connection failed because connected host
[nltk_data]     has failed to respond>
```

In [18]:

```
# Removing punctuation and special character from the text
train.replace(to_replace='[^a-zA-Z]', value=' ', regex=True, inplace=True)
test.replace(to_replace='[^a-zA-Z]', value=' ', regex=True, inplace=True)
```

In [19]:

```
# Renaming columns
new_columns = [str(i) for i in range(0,25)]
train.columns = new_columns
test.columns = new_columns
```

In [20]:

```
# Converting the entire text to lower case
for i in new_columns:
  train[i] = train[i].str.lower()
  test[i] = test[i].str.lower()
```

In [21]:

```
# Joining all the columns
train_headlines = []
test_headlines = []

for row in range(0, train.shape[0]):
  train_headlines.append(' '.join(str(x) for x in train.iloc[row, 0:25]))

for row in range(0, test.shape[0]):
  test_headlines.append(' '.join(str(x) for x in test.iloc[row, 0:25]))
```

In [22]:

```
train_headlines[0]
```

Out[22]:

```
'a  hindrance to operations   extracts from the leaked reports scorecard h
ughes  instant hit buoys blues jack gets his skates on at ice cold alex ch
aos as maracana builds up for united depleted leicester prevail as elliott
spoils everton s party hungry spurs sense rich pickings gunners so wide of
an easy target derby raise a glass to strupar s debut double southgate str
ikes  leeds pay the penalty hammers hand robson a youthful lesson saints p
arty like it s      wear wolves have turned into lambs stump mike catches 
testy gough s taunt langer escapes to hit     flintoff injury piles on woe
for england hunters threaten jospin with new battle of the somme kohl s su
ccessor drawn into scandal the difference between men and women sara denve
r  nurse turned solicitor diana s landmine crusade put tories in a panic y
eltsin s resignation caught opposition flat footed russian roulette sold o
ut recovering a title'
```

In [23]:

```
test_headlines[0]
```

Out[23]:

```
'most cases of cancer are the result of sheer bad luck rather than unhealt
hy lifestyles  diet or even inherited genes  new research suggests  random
mutations that occur in dna when cells divide are responsible for two thir
ds of adult cancers across a wide range of tissues  iran dismissed united
states efforts to fight islamic state as a ploy to advance u s  policies i
n the region   the reality is that the united states is not acting to elim
inate daesh  they are not even interested in weakening daesh  they are onl
y interested in managing it  poll  one in   germans would join anti muslim
marches uk royal family s prince andrew named in us lawsuit over underage
sex allegations some    asylum seekers refused to leave the bus when they
arrived at their destination in rural northern sweden  demanding that they
be taken back to malm or  some big city   pakistani boat blows self up aft
er india navy chase  all four people on board the vessel from near the pak
istani port city of karachi are believed to have been killed in the dramat
ic episode in the arabian sea on new year s eve  according to india s defe
nce ministry  sweden hit by third mosque arson attack in a week     cars s
et alight during french new year salaries for top ceos rose twice as fast
as average canadian since recession  study norway violated equal pay law
judge says  judge finds consulate employee was unjustly paid         less
than her male counterpart imam wants radical recruiters of muslim youth in
canada identified and dealt with saudi arabia beheaded    people in
the most in years  a living hell  for slaves on remote south korean island
s   slavery thrives on this chain of rural islands off south korea s rugge
d southwest coast  nurtured by a long history of exploitation and the dema
nds of trying to squeeze a living from the sea  worlds     richest get ric
her  adding    bn in       rental car stereos infringe copyright  music rig
hts group says ukrainian minister threatens tv channel with closure for ai
ring russian entertainers palestinian president mahmoud abbas has entered
into his most serious confrontation yet with israel by signing onto the in
ternational criminal court  his decision on wednesday gives the court juri
sdiction over crimes committed in palestinian lands  israeli security cent
er publishes names of    killed terrorists  concealed by hamas  the year
was the deadliest year yet in syria s four year conflict  with over
killed a secret underground complex built by the nazis that may have been
used for the development of wmds  including a nuclear bomb  has been uncov
ered in austria  restrictions on web freedom a major global issue in
austrian journalist erich mchel delivered a presentation in hamburg at the
annual meeting of the chaos computer club on monday december     detailing
the various locations where the us nsa has been actively collecting and pr
ocessing electronic intelligence in vienna  thousands of ukraine nationali
sts march in kiev chinas new years resolution  no more harvesting executed
prisoners organs authorities pull plug on russia s last politically indepe
ndent tv station'
```

In [24]:

```
# Creating corpus of train dataset
ps = PorterStemmer()
train_corpus = []

for i in range(0, len(train_headlines)):

  # Tokenizing the news-title by words
  words = train_headlines[i].split()

  # Removing the stopwords
  words = [word for word in words if word not in set(stopwords.words('english'))]

  # Stemming the words
  words = [ps.stem(word) for word in words]

  # Joining the stemmed words
  headline = ' '.join(words)

  # Building a corpus of news-title
  train_corpus.append(headline)
```

In [25]:

```
# Creating corpus of test dataset
test_corpus = []

for i in range(0, len(test_headlines)):

  # Tokenizing the news-title by words
  words = test_headlines[i].split()

  # Removing the stopwords
  words = [word for word in words if word not in set(stopwords.words('english'))]

  # Stemming the words
  words = [ps.stem(word) for word in words]

  # Joining the stemmed words
  headline = ' '.join(words)

  # Building a corpus of news-title
  test_corpus.append(headline)
```

In [26]:

```
train_corpus[0:10]
```

```
Out[26]:
```

```
['hindranc oper extract leak report scorecard hugh instant hit buoy blue j
ack get skate ice cold alex chao maracana build unit deplet leicest prevai
l elliott spoil everton parti hungri spur sens rich pick gunner wide easi
target derbi rais glass strupar debut doubl southgat strike leed pay penal
ti hammer hand robson youth lesson saint parti like wear wolv turn lamb st
ump mike catch testi gough taunt langer escap hit flintoff injuri pile woe
england hunter threaten jospin new battl somm kohl successor drawn scandal
differ men women sara denver nurs turn solicitor diana landmin crusad put
tori panic yeltsin resign caught opposit flat foot russian roulett sold re
cov titl',
 'scorecard best lake scene leader german sleaz inquiri cheerio boyo main
recommend cubi kill fee cubi kill fee cubi kill fee hopkin furiou foster l
ack hannib appetit cubi kill fee tale two tail say like like say elbow eye
nippl task forc assess risk asteroid collis found last critic list time li
ve dear doctor irish court halt ira man extradit northern ireland burundi
peac initi fade rebel reject mandela mediat pe point way forward ecb campa
ign keep pressur nazi war crime suspect jane ratcliff yet thing know witho
ut movi millennium bug fail bite',
 'coventri caught counter flo unit rival road rio thatcher issu defenc tri
al video polic help smith lay law everton tale trautmann bear two retel en
gland rack pakistan retali call video walsh cullinan continu cape monopoli
mcgrath put india miseri blair witch bandwagon roll pele turn heat ferguso
n parti divid kohl slush fund scandal manchest unit england women record s
outh pole walk vasco da gama brazil south melbourn australia necaxa mexico
real madrid spain raja casablanca morocco corinthian brazil toni pet proje
ct al nassr saudi arabia ideal holm show pinochet leav hospit test use lin
k',
 'pilgrim know progress thatcher face ban mcilroy call irish fight spirit
leicest bin stadium blueprint unit brace mexican wave aunti back fashion e
ven dress look bit tatti shoaib appeal goe top hussain hurt shambl lay bla
me earlier damag england decad disast reveng sweet jubil cronj choic profi
l former us nazi parti offic william pierc new evid show record war crime
suspect investig rise supernerd written bodi putin admit yeltsin quit give
head start bbc worst hit digit tv begin bite much pay christma glitch upen
d tabl chop line score goal scientif evid unreli defenc claim fusco win ju
dici review extradit case rebel thwart russian advanc blair order shake fa
il nh lesson law hard heart',
 'hitch horlock beckham unit surviv breast cancer screen alan parker guard
[illegible]
fit lawsuit underag sex alleg asylum seeker refus leav bu arriv destin r
[illegible]
ave remot south korean island slaveri thrive chain rural island south k
[illegible]
queen park peril cloud hampden futur waugh hit shoaib repriev knight make
case butcher place scoreboard bond enough star brosnan help peopl blake bl
ast liverpool german parti leader took cash arm dealer children book week
low go like write split vote may offer natwest takeov escap teach stayer s
printer lesson respect everyon know good school wrong give teacher applaus
```

In [27]:

```
test_corpus[0:10]
```

Out[27]:

```
[illegible]
```

```
realis felt sick shock tender stung russian forc inspector warn pressur mi
ght lead offic take easi solv case repair jack hous',
 'newcastl seek new footbal supremo liverpool aim speed heskey deal highla
nd vote edward power play suffer new blow chelsea gambl weah taylor settl
etern tie tenth top flight club fall hodg final word charlton charg top ge
rman parti chief resist call resign beach made man leo pariah irv sue holo
caust author jack straw full common speech batti busi book frock conscious
matter heroin wear arm pakistan like mind co black megabuck luck cabinet b
attl rage ethic foreign polici radio station becom talk sport better breed
dad childish thing kid say hopscotch smoke without fire press reaction spa
in chile argentina',
 'bungl offici carpet red raw corner killer mackenzi unit put shirt englan
d plan home nation reviv donald pois quit test scene adam stare abyss mone
y money money tyson enter britain ga chamber claim imposs say irv union ta
ke mayor vote row court win ticket end affair irv deni deliber portray hit
ler merci gallant fulham flunk shoot hill start leav tranmer one step wemb
ley wean strike right cue dinner plate microwav oven toast one simultan fr
ee yet live fear doubt pakistan arm export anoth fine mess cybershop sport
swear much pay hillari hold late show bye bye american pi tension mount st
raw stand trial plan harrod lose princ philip royal warrant',
 'pompey plump puli work ethic roma fire rolex referee preston park two tol
d take break ok figur rio still chelsea tune weah world top storey await c
ottag west indi unveil fieri next gener donald kill field await edgi engla
nd cronj law run end captainci high counti lose murali shoaib final tell t
ale two citi tv rival jostl soccer club stake blatter unit gave boost worl
d cup bid sunderland recoveri bad news booki england main fault tri hard s
chreiber man would toppl king joe ashton letter resign ashton resign wedne
sday board stravinski rake progress best waterfront scene incompet insult
injuri find time england take six rain shorten first day media sale net ch
ri evan divvi scissor dawn']
```

In [28]:

```
down_words = []
for i in list(y_train[y_train==0].index):
  down_words.append(train_corpus[i])

up_words = []
for i in list(y_train[y_train==1].index):
  up_words.append(train_corpus[i])
```

In [29]:

```
import sys
print(sys.executable)
```

```
C:\Users\psai1\anaconda3\python.exe
```

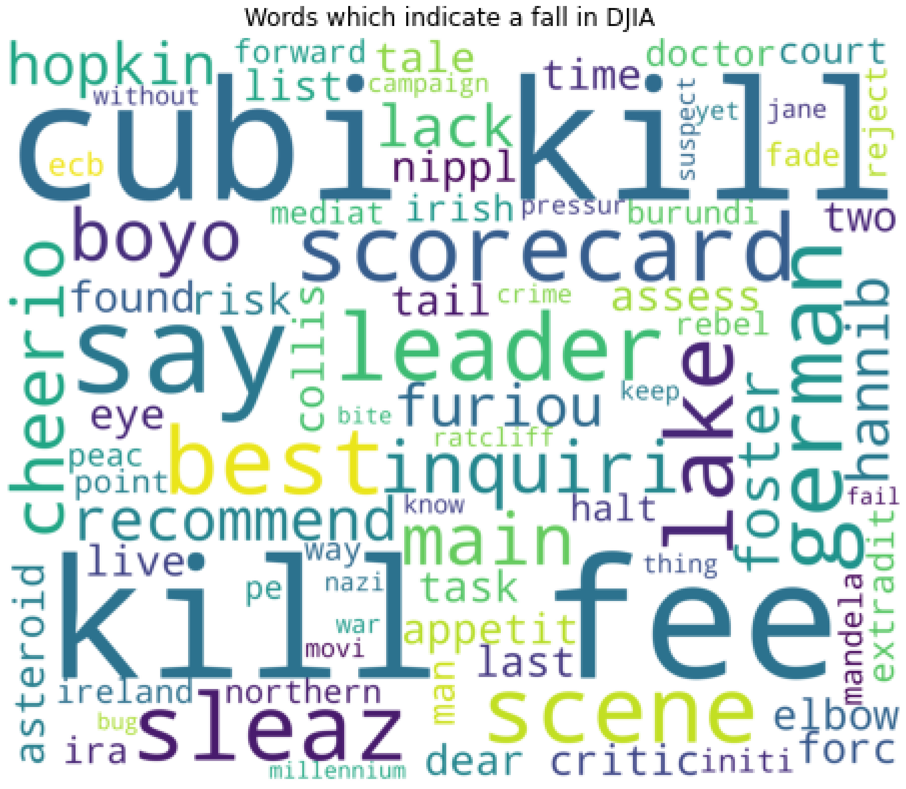
In [30]:

```
!<C:\Users\psai1\anaconda3\python.exe>/python -m pip install wordcloud
```

```
Access is denied.
```

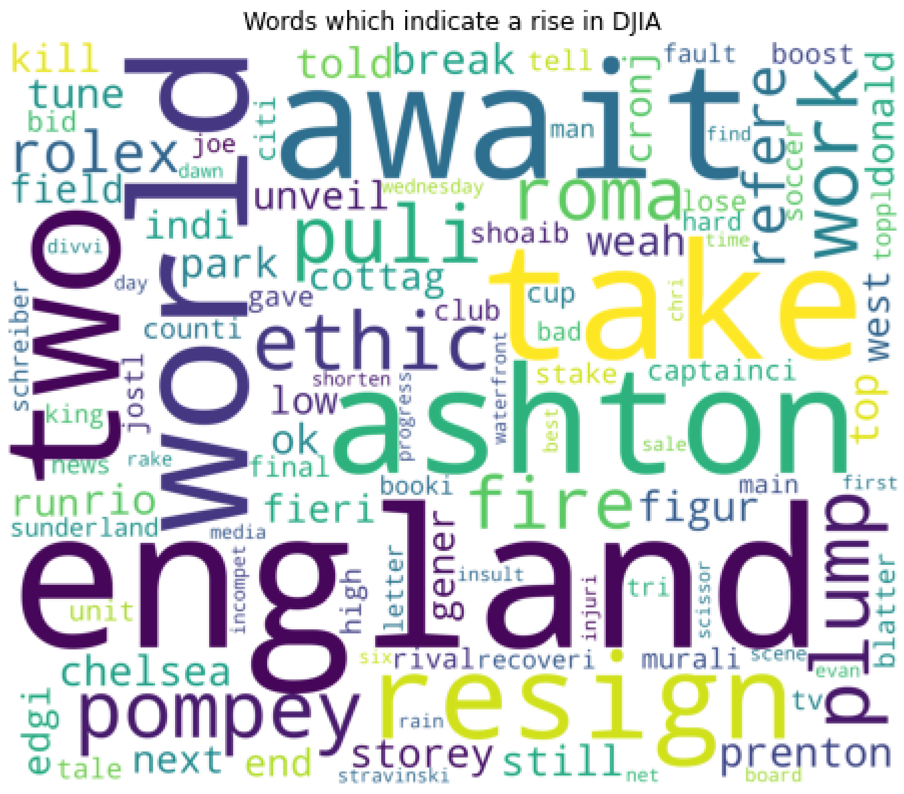
In [31]:

```
# creating wordcloud for down_words
from wordcloud import WordCloud
wordcloud1 = WordCloud(background_color='white', width=3000, height=2500).generate(down_
plt.figure(figsize=(8,8))
plt.imshow(wordcloud1)
plt.axis('off')
plt.title("Words which indicate a fall in DJIA ")
plt.show()
```



In [32]:

```
# Creating wordcloud for up_words
wordcloud2 = WordCloud(background_color='white', width=3000, height=2500).generate(up_wo
plt.figure(figsize=(8,8))
plt.imshow(wordcloud2)
plt.axis('off')
plt.title("Words which indicate a rise in DJIA ")
plt.show()
```



In [33]:

```
# Creating the Bag of Words model
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=10000, ngram_range=(2,2))
X_train = cv.fit_transform(train_corpus).toarray()
```

In [34]:

```
X_test = cv.transform(test_corpus).toarray()
```

Model Building

Logistic Regression

In [35]:

```
from sklearn.linear_model import LogisticRegression
lr_classifier = LogisticRegression()
lr_classifier.fit(X_train, y_train)
```

Out[35]:

```
LogisticRegression()
```

In [36]:

```
lr_y_pred = lr_classifier.predict(X_test)
```

In [37]:

```
# Accuracy, Precision and Recall
from sklearn.metrics import accuracy_score, precision_score, recall_score
score1 = accuracy_score(y_test, lr_y_pred)
score2 = precision_score(y_test, lr_y_pred)
score3 = recall_score(y_test, lr_y_pred)
print("---- Scores ----")
print("Accuracy score is: {}%".format(round(score1*100,2)))
print("Precision score is: {}".format(round(score2,2)))
print("Recall score is: {}".format(round(score3,2)))
```

```
---- Scores ----
Accuracy score is: 85.98%
Precision score is: 0.87
Recall score is: 0.85
```

In [38]:

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
lr_cm = confusion_matrix(y_test, lr_y_pred)
```

In [39]:

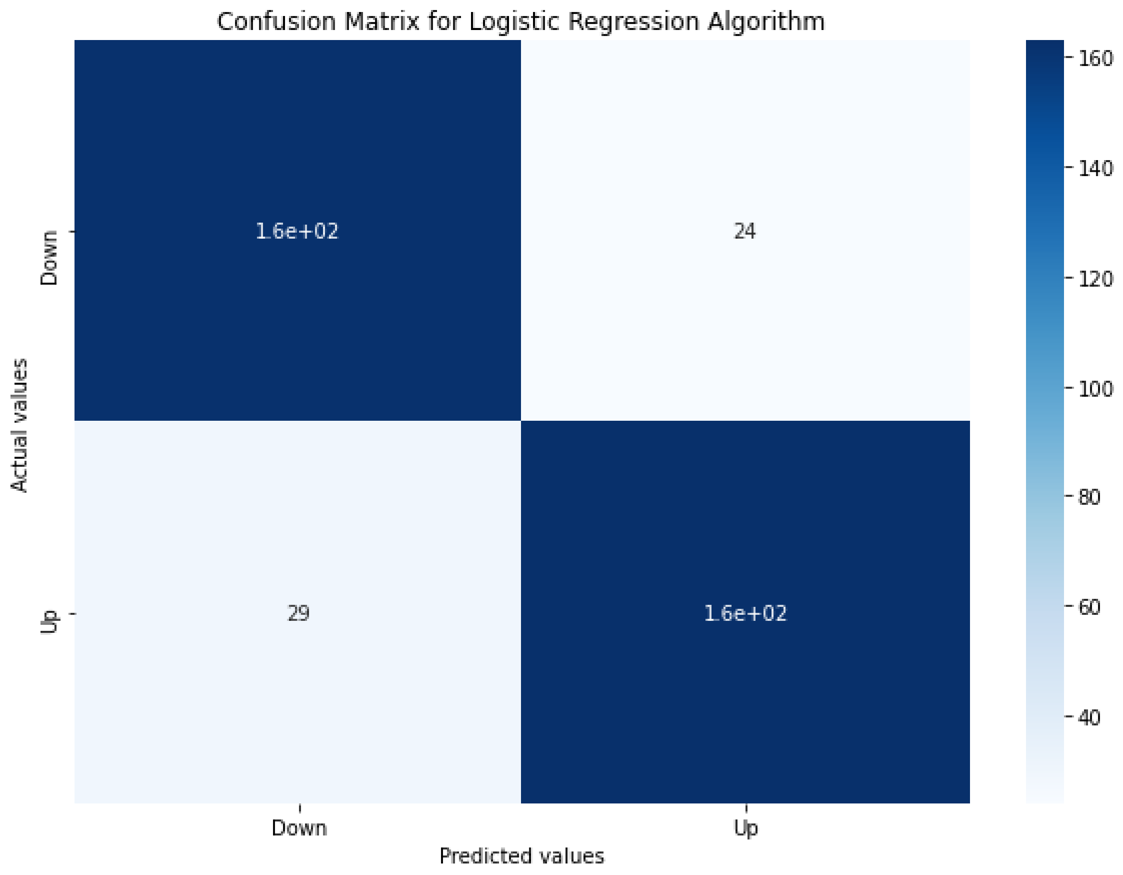
```
lr_cm
```

Out[39]:

```
array([[162,  24],
       [ 29, 163]], dtype=int64)
```

In [40]:

```
# Plotting the confusion matrix
plt.figure(figsize=(10,7))
sns.heatmap(data=lr_cm, annot=True, cmap="Blues", xticklabels=['Down', 'Up'], yticklabel
plt.xlabel('Predicted values')
plt.ylabel('Actual values')
plt.title('Confusion Matrix for Logistic Regression Algorithm')
plt.show()
```



Random Forest Classifier

In [41]:

```
from sklearn.ensemble import RandomForestClassifier
rf_classifier = RandomForestClassifier(n_estimators=100, criterion='entropy')
rf_classifier.fit(X_train, y_train)
```

Out[41]:

```
RandomForestClassifier(criterion='entropy')
```

In [42]:

```
rf_y_pred = rf_classifier.predict(X_test)
```

In [43]:

```
# Accuracy, Precision and Recall
score1 = accuracy_score(y_test, rf_y_pred)
score2 = precision_score(y_test, rf_y_pred)
score3 = recall_score(y_test, rf_y_pred)
print("---- Scores ----")
print("Accuracy score is: {}%".format(round(score1*100,2)))
print("Precision score is: {}".format(round(score2,2)))
print("Recall score is: {}".format(round(score3,2)))
```

```
---- Scores ----
Accuracy score is: 82.8%
Precision score is: 0.82
Recall score is: 0.85
```

In [44]:

```
# Making the Confusion Matrix
rf_cm = confusion_matrix(y_test, rf_y_pred)
```

In [45]:

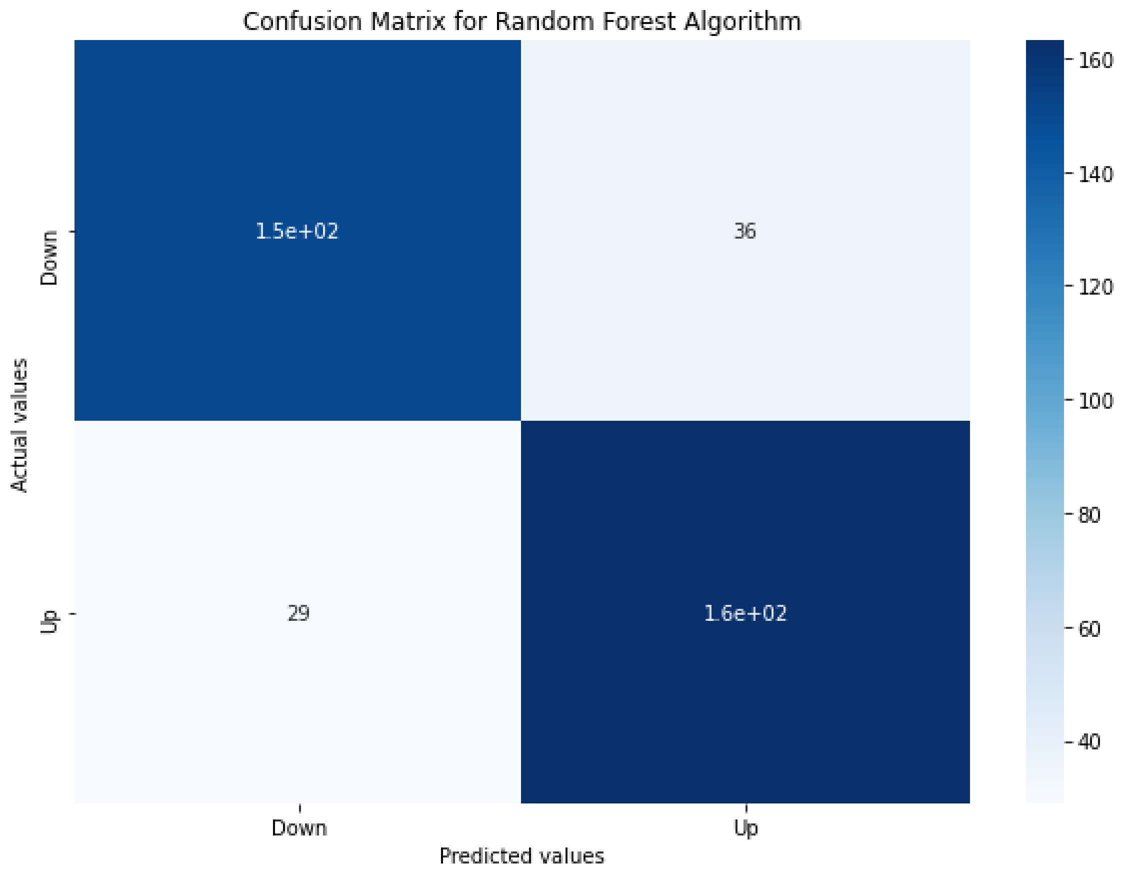
```
rf_cm
```

Out[45]:

```
array([[150,  36],
       [ 29, 163]], dtype=int64)
```

In [46]:

```
# Plotting the confusion matrix
plt.figure(figsize=(10,7))
sns.heatmap(data=rf_cm, annot=True, cmap="Blues", xticklabels=['Down', 'Up'], yticklabel
plt.xlabel('Predicted values')
plt.ylabel('Actual values')
plt.title('Confusion Matrix for Random Forest Algorithm')
plt.show()
```



Multinomial Naive Bayes

In [47]:

```
from sklearn.naive_bayes import MultinomialNB
nb_classifier = MultinomialNB()
nb_classifier.fit(X_train, y_train)
```

Out[47]:

```
MultinomialNB()
```

In [48]:

```
# Predicting the Test set results
nb_y_pred = nb_classifier.predict(X_test)
```

In [49]:

```
# Accuracy, Precision and Recall
score1 = accuracy_score(y_test, nb_y_pred)
score2 = precision_score(y_test, nb_y_pred)
score3 = recall_score(y_test, nb_y_pred)
print("---- Scores ----")
print("Accuracy score is: {}%".format(round(score1*100,2)))
print("Precision score is: {}".format(round(score2,2)))
print("Recall score is: {}".format(round(score3,2)))
```

```
---- Scores ----
Accuracy score is: 83.86%
Precision score is: 0.85
Recall score is: 0.83
```

In [50]:

```
# Making the Confusion Matrix
nb_cm = confusion_matrix(y_test, nb_y_pred)
```

In [51]:

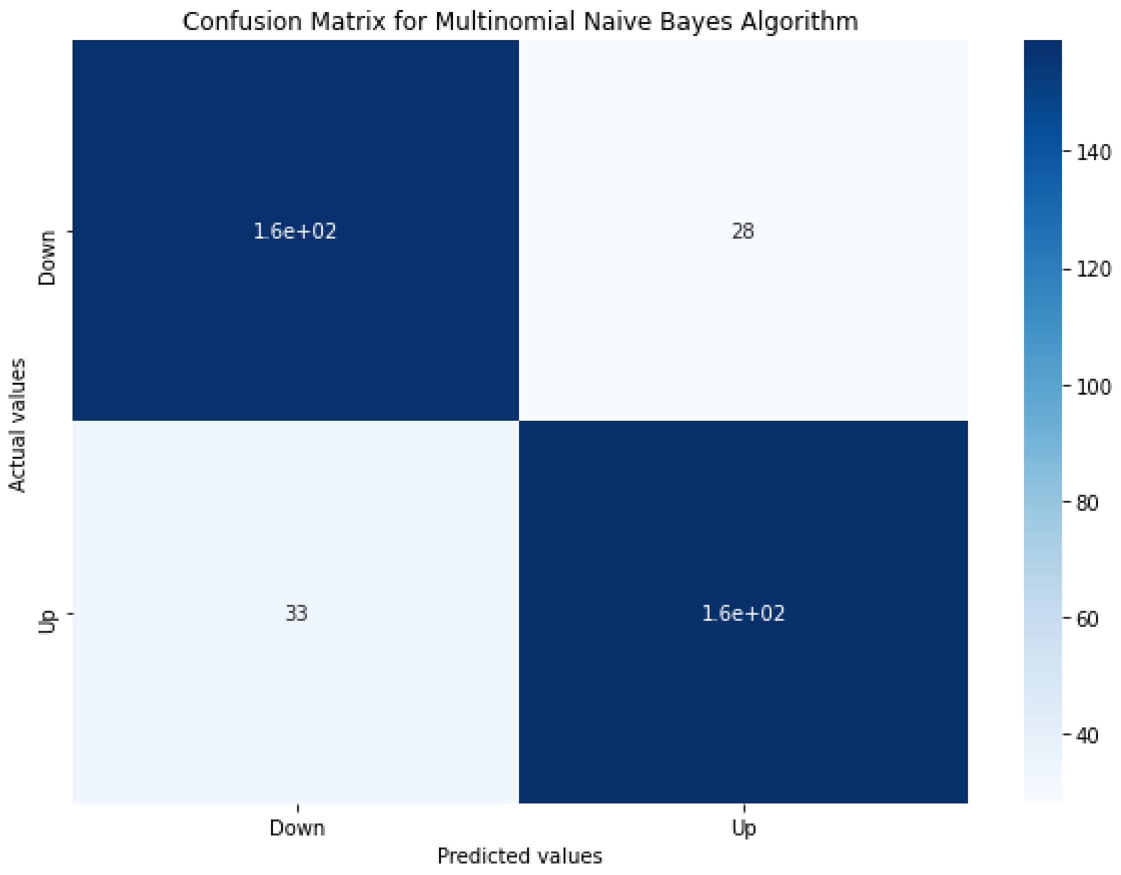
```
nb_cm
```

Out[51]:

```
array([[158,  28],
       [ 33, 159]], dtype=int64)
```

In [52]:

```
# Plotting the confusion matrix
plt.figure(figsize=(10,7))
sns.heatmap(data=nb_cm, annot=True, cmap="Blues", xticklabels=['Down', 'Up'], yticklabel
plt.xlabel('Predicted values')
plt.ylabel('Actual values')
plt.title('Confusion Matrix for Multinomial Naive Bayes Algorithm')
plt.show()
```



Predictions

In [53]:

```
import re

def stock_prediction(sample_news):
  sample_news = re.sub(pattern='[^a-zA-Z]',repl=' ', string=sample_news)
  sample_news = sample_news.lower()
  sample_news_words = sample_news.split()
  sample_news_words = [word for word in sample_news_words if not word in set(stopwords.w
  ps = PorterStemmer()
  final_news = [ps.stem(word) for word in sample_news_words]
  final_news = ' '.join(final_news)

  temp = cv.transform([final_news]).toarray()
  return lr_classifier.predict(temp)
```

In [54]:

```
# For generating random integer
from random import randint
```

In [55]:

```
sample_test = df_copy[df_copy['Date'] > '20141231']
```

In [56]:

```
sample_test.reset_index(inplace=True)
sample_test = sample_test['Top1']
```

In [57]:

```
# Predicting values
row = randint(0,sample_test.shape[0]-1)
sample_news = sample_test[row]

print('News: {}'.format(sample_news))
if stock_prediction(sample_news):
  print('Prediction: The stock price will remain the same or will go down.')
else:
  print('Prediction: The stock price will go up!')
```

```
News: El Chapo' Being Taken to Same Prison He Escaped from Six Months Ago
Prediction: The stock price will remain the same or will go down.
```

In [58]:

```
# Predicting values
row = randint(0,sample_test.shape[0]-1)
sample_news = sample_test[row]

print('News: {}'.format(sample_news))
if stock_prediction(sample_news):
  print('Prediction: The stock price will remain the same or will go down.')
else:
  print('Prediction: The stock price will go up!')
```

```
News: Efficiency up, turnover down: Sweden experiments with six-hour worki
ng day | World news | The Guardian
Prediction: The stock price will remain the same or will go down.
```

In [59]:

```
# Predicting values
row = randint(0,sample_test.shape[0]-1)
sample_news = sample_test[row]

print('News: {}'.format(sample_news))
if stock_prediction(sample_news):
  print('Prediction: The stock price will remain the same or will go down.')
else:
  print('Prediction: The stock price will go up!')
```

```
News: US State Dept declares ISIS is committing genocide in Iraq, Syria
Prediction: The stock price will remain the same or will go down.
```

In [60]:

```
# Predicting values
row = randint(0,sample_test.shape[0]-1)
sample_news = ""
print('News: {}'.format(sample_news))
if stock_prediction(sample_news):
  print('Prediction: The stock price will remain the same or will go down.')
else:
  print('Prediction: The stock price will go up!')
```

```
News:
Prediction: The stock price will remain the same or will go down.
```

In [61]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [62]:

```
df=pd.read_csv('stock headlines.csv', encoding="ISO-8859-1")
df.head()
```

Out[62]:

	Date	Label	Top1	Top2	Top3	Top4	Top5	Top6	T
0	2000-01-03	0	A 'hindrance to operations': extracts from the...	Scorecard	Hughes' instant hit buoys Blues	Jack gets his skates on at ice-cold Alex	Chaos as Maracana builds up for United	Depleted Leicester prevail as Elliott spoils E...	Hu S s pick
1	2000-01-04	0	Scorecard	The best lake scene	Leader: German sleaze inquiry	Cheerio, boyo	The main recommendations	Has Cubie killed fees?	C I f
2	2000-01-05	0	Coventry caught on counter by Flo	United's rivals on the road to Rio	Thatcher issues defence before trial by video	Police help Smith lay down the law at Everton	Tale of Trautmann bears two more retellings	England on the rack	Pak reta with for v of W
3	2000-01-06	1	Pilgrim knows how to progress	Thatcher facing ban	McIlroy calls for Irish fighting spirit	Leicester bin stadium blueprint	United braced for Mexican wave	Auntie back in fashion, even if the dress look...	Sh ap go th
4	2000-01-07	1	Hitches and Horlocks	Beckham off but United survive	Breast cancer screening	Alan Parker	Guardian readers: are you all whingers?	Hollywood Beyond	A diam

5 rows × 27 columns

In [63]:

```
train=df[df['Date']<'20150101']
test=df[df['Date']>'20141231']
train.shape
```

Out[63]:

```
(3975, 27)
```

In [64]:

```
#Removing punctuations
data=train.iloc[:,2:27]
data.replace("[^a-zA-Z]", " ",regex=True, inplace=True)
```

In [65]:

```
data.columns
```

Out[65]:

```
Index(['Top1', 'Top2', 'Top3', 'Top4', 'Top5', 'Top6', 'Top7', 'Top8', 'To
p9',
       'Top10', 'Top11', 'Top12', 'Top13', 'Top14', 'Top15', 'Top16', 'Top
17',
       'Top18', 'Top19', 'Top20', 'Top21', 'Top22', 'Top23', 'Top24', 'Top
25'],
      dtype='object')
```

In [66]:

```
for col in data.columns:
    data[col]=data[col].str.lower()
data.head(1)
```

Out[66]:

	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top
0	a hindrance to operations extracts from the...	scorecard	hughes instant hit buoys blues	jack gets his skates on at ice cold alex	chaos as maracana builds up for united	depleted leicester prevail as elliott spoils e...	hungry spurs sense rich pickings	gunners so wide of an easy target	derby raise a glass to strupar s debut double	southg strik leeds p t pena

1 rows × 25 columns

In [67]:

```
headlines = []
for row in range(0,len(data.index)):
    headlines.append(' '.join(str(x) for x in data.iloc[row,0:25]))
```

Using TF-IDF

In [68]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
```

In [69]:

```
#implement TF-IDF
tfvector=TfidfVectorizer(ngram_range=(2,3))
train_df=tfvector.fit_transform(headlines)
```

In [70]:

```
import pickle
pickle.dump(tfvector, open('tfvector.pkl', 'wb'))
```

RandomForestClassifier

In [71]:

```
# implement RandomForest Classifier
randomclassifier=RandomForestClassifier(n_estimators=200,criterion='entropy')
randomclassifier.fit(train_df,train['Label'])
```

Out[71]:

```
RandomForestClassifier(criterion='entropy', n_estimators=200)
```

plot_confusion_matrix

In [72]:

```
from sklearn import metrics
import itertools
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    See full source and example:
    http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.h

    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
```

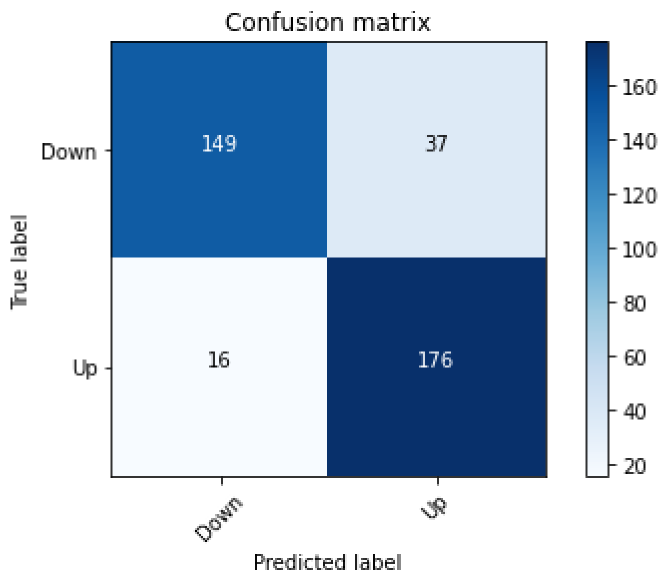
In [73]:

```
# Predict for the Test Dataset
test_transform= []
for row in range(0,len(test.index)):
    test_transform.append(' '.join(str(x) for x in test.iloc[row,2:27]))
test_dataset = tfvector.transform(test_transform)
predictions = randomclassifier.predict(test_dataset)
```

In [74]:

```
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
matrix=confusion_matrix(test['Label'],predictions)
print(matrix)
score=accuracy_score(test['Label'],predictions)
print(score)
report=classification_report(test['Label'],predictions)
print(report)
plot_confusion_matrix(matrix, classes=['Down', 'Up'])
```

```
[[149  37]
 [ 16 176]]
0.8597883597883598
              precision    recall  f1-score   support

           0       0.90      0.80      0.85       186
           1       0.83      0.92      0.87       192

    accuracy                           0.86       378
   macro avg       0.86      0.86      0.86       378
weighted avg       0.86      0.86      0.86       378

Confusion matrix, without normalization
```



MultinomialNB

In [75]:

```
from sklearn.naive_bayes import MultinomialNB
nb=MultinomialNB()
nb.fit(train_df,train['Label'])
```

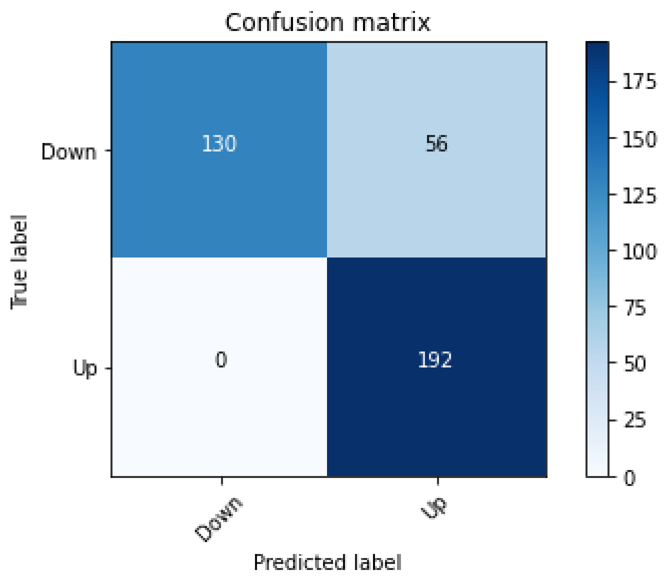
Out[75]:

```
MultinomialNB()
```

In [76]:

```
predictions = nb.predict(test_dataset)
matrix=confusion_matrix(test['Label'],predictions)
print(matrix)
score=accuracy_score(test['Label'],predictions)
print(score)
report=classification_report(test['Label'],predictions)
print(report)
plot_confusion_matrix(matrix, classes=['Down', 'Up'])
```

```
[[130  56]
 [  0 192]]
0.8518518518518519
              precision    recall  f1-score   support

           0       1.00      0.70      0.82       186
           1       0.77      1.00      0.87       192

    accuracy                           0.85       378
   macro avg       0.89      0.85      0.85       378
weighted avg       0.89      0.85      0.85       378

Confusion matrix, without normalization
```



PassiveAggressiveClassifier

In [77]:

```
from sklearn.linear_model import PassiveAggressiveClassifier
pa = PassiveAggressiveClassifier()

pa.fit(train_df,train['Label'])
```

Out[77]:

```
PassiveAggressiveClassifier()
```

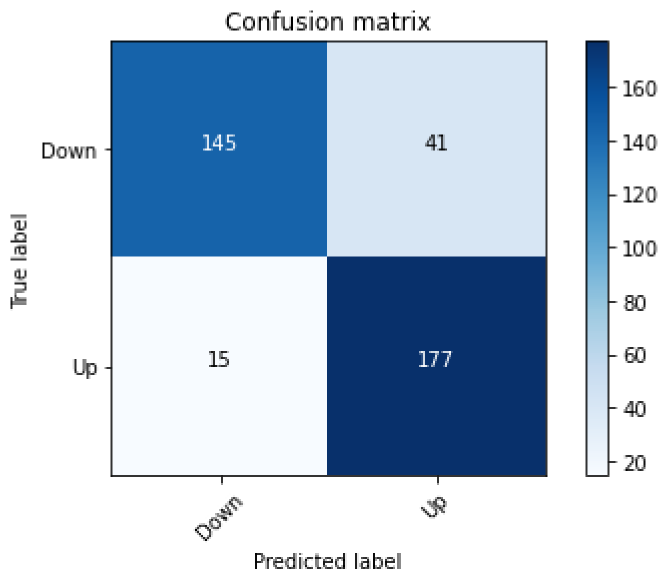
In [78]:

```
import pickle
filename = 'stock_senti.pkl'
pickle.dump(pa, open(filename, 'wb'))
```

In [79]:

```
predictions = pa.predict(test_dataset)
matrix=confusion_matrix(test['Label'],predictions)
print(matrix)
score=accuracy_score(test['Label'],predictions)
print(score)
report=classification_report(test['Label'],predictions)
print(report)
plot_confusion_matrix(matrix, classes=['Down', 'Up'])
```

```
[[145  41]
 [ 15 177]]
0.8518518518518519
              precision    recall  f1-score   support

           0       0.91      0.78      0.84       186
           1       0.81      0.92      0.86       192

    accuracy                           0.85       378
   macro avg       0.86      0.85      0.85       378
weighted avg       0.86      0.85      0.85       378

Confusion matrix, without normalization
```



Using bag of words

In [80]:

```
from sklearn.feature_extraction.text import CountVectorizer
#implement bag of words
bow=CountVectorizer(ngram_range=(2,3))
train_df=bow.fit_transform(headlines)
```

RandomForestClassifier using Bag of words

In [81]:

```
# implement RandomForest Classifier
randomclassifier=RandomForestClassifier(n_estimators=200,criterion='entropy')
randomclassifier.fit(train_df,train['Label'])
```

Out[81]:

```
RandomForestClassifier(criterion='entropy', n_estimators=200)
```

In [82]:

```
predictions = randomclassifier.predict(test_dataset)
matrix=confusion_matrix(test['Label'],predictions)
print(matrix)
score=accuracy_score(test['Label'],predictions)
print(score)
report=classification_report(test['Label'],predictions)
print(report)
```

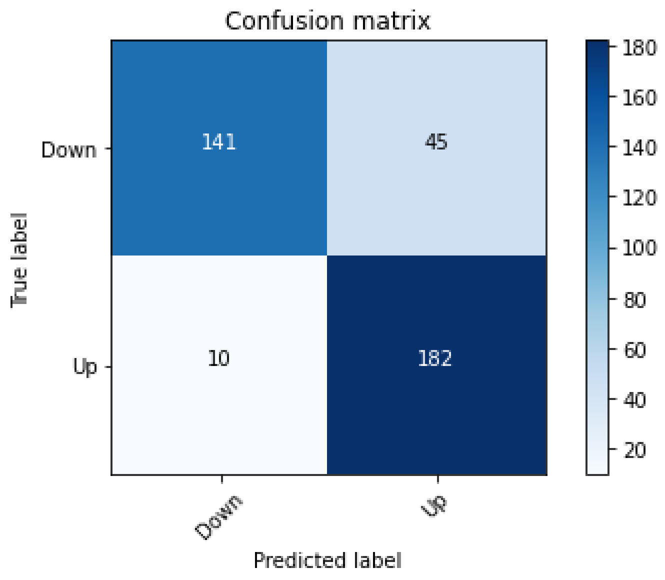
```
[[  0 186]
 [  0 192]]
0.5079365079365079
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       186
           1       0.51      1.00      0.67       192

    accuracy                           0.51       378
   macro avg       0.25      0.50      0.34       378
weighted avg       0.26      0.51      0.34       378


C:\Users\psai1\anaconda3\lib\site-packages\sklearn\metrics\_classificatio
n.py:1248: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\psai1\anaconda3\lib\site-packages\sklearn\metrics\_classificatio
n.py:1248: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\psai1\anaconda3\lib\site-packages\sklearn\metrics\_classificatio
n.py:1248: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

MultinomialNB using Bag of words

In [83]:

```
from sklearn.naive_bayes import MultinomialNB
nb=MultinomialNB()
nb.fit(train_df,train['Label'])

predictions = nb.predict(test_dataset)
matrix=confusion_matrix(test['Label'],predictions)
print(matrix)
score=accuracy_score(test['Label'],predictions)
print(score)
report=classification_report(test['Label'],predictions)
print(report)
plot_confusion_matrix(matrix, classes=['Down', 'Up'])
```

```
[[141  45]
 [ 10 182]]
0.8544973544973545
              precision    recall  f1-score   support

           0       0.93      0.76      0.84       186
           1       0.80      0.95      0.87       192

    accuracy                           0.85       378
   macro avg       0.87      0.85      0.85       378
weighted avg       0.87      0.85      0.85       378

Confusion matrix, without normalization
```

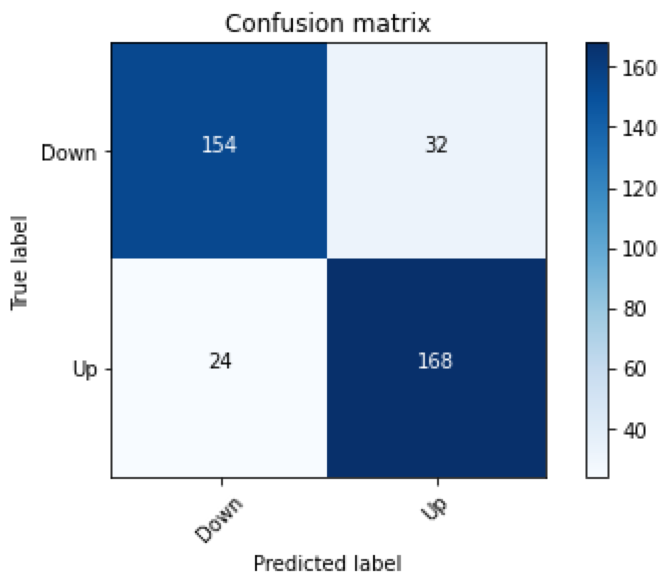


PassiveAggressiveClassifier using Bag of Words

In [84]:

```
from sklearn.linear_model import PassiveAggressiveClassifier
pa = PassiveAggressiveClassifier()
pa.fit(train_df,train['Label'])

predictions = pa.predict(test_dataset)
matrix=confusion_matrix(test['Label'],predictions)
print(matrix)
score=accuracy_score(test['Label'],predictions)
print(score)
report=classification_report(test['Label'],predictions)
print(report)
plot_confusion_matrix(matrix, classes=['Down', 'Up'])
```

```
[[154  32]
 [ 24 168]]
0.8518518518518519
              precision    recall  f1-score   support

           0       0.87      0.83      0.85       186
           1       0.84      0.88      0.86       192

    accuracy                           0.85       378
   macro avg       0.85      0.85      0.85       378
weighted avg       0.85      0.85      0.85       378

Confusion matrix, without normalization
```



Spark

In [85]:

```
import findspark
findspark.init()
import pyspark
from pyspark.sql.session import SparkSession
spark = SparkSession.builder.appName("Yahoo finance").getOrCreate()
```

In [86]:

```
import yfinance as yf
import csv
from pyspark.sql.functions import date_format

from pyspark.sql.functions import dayofmonth, hour, dayofyear, month, year, weekofyear

ticker = '^GSPC' # tsla
start_dt = '2021-1-1'
end_dt = '2022-12-31'
frequency = '1d'

#get data for SPX
data = yf.Ticker(ticker)
#ticker_name = data.info['longName']

#get weekly historical prices for this ticker
df = data.history(interval = frequency, start = start_dt, end = end_dt)

# write data into a csv file
df.to_csv('spx.csv')

# Let Spark know about the header and infer the Schema types!
df = spark.read.csv('spx.csv',inferSchema=True,header=True)

# create new dataframe month_df adding new columns month, diff, high_low_diff and weekof
daily_df = df.withColumn('Month', month(df['Date'])).withColumn('diff', df['Open']-df['C
.withColumn('week_of_year', weekofyear(df.Date)).withColumn('week_day', date_format(df.D
daily_df.show(5)
```

```
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+----------------+------------+---------+
|                Date|             Open|             High|              Lo
w|            Close|    Volume|Dividends|Stock Splits|Month|              di
ff|   high_low_diff|week_of_year| week_day|
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+----------------+------------+---------+
|2021-01-04 00:00:...|3764.610107421875|3769.989990234375|  3662.709960937
5| 3700.64990234375|5015000000|      0.0|         0.0|    1| 63.9602050781
25|107.280029296875|           1|   Monday|
|2021-01-05 00:00:...| 3698.02001953125|   3737.830078125|3695.07006835937
5|3726.860107421875|4591020000|      0.0|         0.0|    1|-28.8400878906
25| 42.760009765625|           1|  Tuesday|
|2021-01-06 00:00:...|3712.199951171875|  3783.0400390625|3705.34008789062
5|3748.139892578125|6064110000|      0.0|         0.0|    1| -35.939941406
25| 77.699951171875|           1|Wednesday|
|2021-01-07 00:00:...|  3764.7099609375|3811.550048828125|  3764.709960937
5|  3803.7900390625|5099160000|      0.0|         0.0|    1|   -39.0800781
25| 46.840087890625|           1| Thursday|
|2021-01-08 00:00:...|3815.050048828125| 3826.68994140625| 3783.6000976562
5|3824.679931640625|4773040000|      0.0|         0.0|    1|   -9.62988281
25|     43.08984375|           1|   Friday|
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+----------------+------------+---------+
only showing top 5 rows
```

In [87]:

```
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((diff > 20 or diff < -20) and (Date > "2022-01-
```

```
+--------------------+----------------+----------------+---------------
-+----------------+----------+---------+------------+-----+------------
----+--------------+------------+---------+
|                Date|            Open|            High|             Lo
w|           Close|    Volume|Dividends|Stock Splits|Month|
diff|  high_low_diff|week_of_year| week_day|
+--------------------+----------------+----------------+---------------
-+----------------+----------+---------+------------+-----+------------
----+--------------+------------+---------+
|2022-01-05 00:00:...|  4787.990234375| 4797.7001953125|4699.4399414062
5|  4700.580078125|4887960000|      0.0|         0.0|    1|     87.4101
5625| 98.26025390625|           1|Wednesday|
|2022-01-07 00:00:...|   4697.66015625| 4707.9501953125|  4662.74023437
5|4677.02978515625|4181510000|      0.0|         0.0|    1|  20.6303710
9375|  45.2099609375|           1|   Friday|
|2022-01-11 00:00:...|4669.14013671875| 4714.1298828125|4638.2700195312
5|4713.06982421875|4101590000|      0.0|         0.0|    1|     -43.929
6875| 75.85986328125|           2|  Tuesday|
|2022-01-13 00:00:...|4733.56005859375| 4744.1298828125| 4650.290039062
```

In [88]:

```
# total days when market was open in 2022
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((Date > "2021-01-01 00:00:00") and (Date < "202
```

Out[88]:

```
252
```

In [89]:

```
# total days when market was open in 2022
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where Date > "2022-01-01 00:00:00"').count()
```

Out[89]:

```
251
```

In [90]:

```
#number of days where spx moved by 20 (difference between open and close) in 2021
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((diff > 20 or diff < -20) \
and ((Date > "2021-01-01 00:00:00") and ((Date < "2022-01-01 00:00:00"))))').count()
```

Out[90]:

```
118
```

In [91]:

```
#number of days where spx moved by 20 (difference between open and close) in 2022
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((diff > 20 or diff < -20)) \
and ((Date > "2022-01-01 00:00:00"))').count()
```

Out[91]:

```
184
```

In [92]:

```
#number of days where spx moved by less 20 (difference between open and close)
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((diff < 20 and diff > -20) and (Date > "2022-01
```

Out[92]:

```
67
```

In [93]:

```
#number of days where spx moved by 20 (difference between high and low of the day)
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((high_low_diff > 20 or high_low_diff < -20) and
```

Out[93]:

```
250
```

In [94]:

```
# number of days when spx moved by 40 (difference between high and low of the day)
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((high_low_diff > 40 or high_low_diff < -40) and
```

Out[94]:

```
227
```

In [95]:

```
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where ((high_low_diff > 40 or high_low_diff < -40) and
+--------------------+----------------+----------------+----------------+-
---------------+----------+---------+------------+-----+----------------+--
-------------+------------+---------+
|                Date|            Open|            High|             Low|
Close|    Volume|Dividends|Stock Splits|Month|            diff|  high_low_d
iff|week_of_year| week_day|
+--------------------+----------------+----------------+----------------+-
---------------+----------+---------+------------+-----+----------------+--
-------------+------------+---------+
|2022-01-04 00:00:...|  4804.509765625| 4818.6201171875|4774.27001953125|
4793.5400390625|4683170000|      0.0|         0.0|    1|  10.9697265625| 4
4.35009765625|           1|  Tuesday|
|2022-01-05 00:00:...|  4787.990234375| 4797.7001953125|4699.43994140625|
4700.580078125|4887960000|      0.0|         0.0|    1|    87.41015625| 9
8.26025390625|           1|Wednesday|
|2022-01-06 00:00:...|4693.39013671875|  4725.009765625|  4671.259765625|
4696.0498046875|4295280000|      0.0|         0.0|    1| -2.65966796875|
53.75|           1| Thursday|
|2022-01-07 00:00:...|   4697.66015625| 4707.9501953125|  4662.740234375|4
677.02978515625|4181510000|      0.0|         0.0|    1| 20.63037109375|
45.2099609375|           1|   Friday|
|2022-01-10 00:00:...|   4655.33984375|4673.02001953125|  4582.240234375|
4670.2900390625|4511810000|      0.0|         0.0|    1| -14.9501953125| 9
0.77978515625|           2|   Monday|
|2022-01-11 00:00:...|4669.14013671875| 4714.1298828125|4638.27001953125|4
713.06982421875|4101590000|      0.0|         0.0|    1|    -43.9296875| 7
5.85986328125|           2|  Tuesday|
|2022-01-12 00:00:...|   4728.58984375|  4748.830078125| 4706.7099609375|4
726.35009765625|4048220000|      0.0|         0.0|    1|  2.23974609375|
42.1201171875|           2|Wednesday|
|2022-01-13 00:00:...|4733.56005859375| 4744.1298828125| 4650.2900390625|4
659.02978515625|4251730000|      0.0|         0.0|    1|  74.5302734375|
93.83984375|           2| Thursday|
|2022-01-14 00:00:...|  4637.990234375| 4665.1298828125|         4614.75|4
662.85009765625|4338490000|      0.0|         0.0|    1|-24.85986328125|
50.3798828125|           2|   Friday|
|2022-01-18 00:00:...|  4632.240234375|  4632.240234375| 4568.7001953125|4
577.10986328125|4748700000|      0.0|         0.0|    1| 55.13037109375|
63.5400390625|           3|  Tuesday|
|2022-01-19 00:00:...|4588.02978515625| 4611.5498046875| 4530.2001953125|
4532.759765625|4465740000|      0.0|         0.0|    1| 55.27001953125|
81.349609375|           3|Wednesday|
|2022-01-20 00:00:...|4547.35009765625|4602.10986328125| 4477.9501953125|4
482.72998046875|4640870000|      0.0|         0.0|    1|  64.6201171875|12
4.15966796875|           3| Thursday|
|2022-01-21 00:00:...| 4471.3798828125|4494.52001953125|   4395.33984375|4
397.93994140625|5589100000|      0.0|         0.0|    1| 73.43994140625| 9
9.18017578125|           3|   Friday|
|2022-01-24 00:00:...|4356.31982421875|4417.35009765625| 4222.6201171875|
4410.1298828125|6928110000|      0.0|         0.0|    1|-53.81005859375|19
4.72998046875|           4|   Monday|
|2022-01-25 00:00:...|4366.64013671875|  4411.009765625|4287.10986328125|
4356.4501953125|5145050000|      0.0|         0.0|    1| 10.18994140625|12
3.89990234375|           4|  Tuesday|
|2022-01-26 00:00:...|4408.43017578125|4453.22998046875| 4304.7998046875|4
349.93017578125|5570640000|      0.0|         0.0|    1|            58.5|14
8.43017578125|           4|Wednesday|
|2022-01-27 00:00:...|  4380.580078125|  4428.740234375|          4309.5|
4326.509765625|5214200000|      0.0|         0.0|    1|     54.0703125|  1
19.240234375|           4| Thursday|
|2022-01-28 00:00:...|4336.18994140625|4432.72021484375| 4292.4599609375|4
```

```
431.85009765625|5031090000|      0.0|          0.0|    1|   -95.66015625|14
0.26025390625|           4|   Friday|
|2022-01-31 00:00:...| 4431.7900390625|4516.89013671875|4414.02001953125|
4515.5498046875|5098610000|      0.0|          0.0|    1|  -83.759765625| 1
02.8701171875|           5|   Monday|
|2022-02-01 00:00:...|4519.56982421875|  4550.490234375|4483.52978515625|
4546.5400390625|4816830000|      0.0|          0.0|    2|-26.97021484375| 6
6.96044921875|           5|  Tuesday|
+--------------------+----------------+----------------+----------------+-
---------------+----------+---------+-------------+-----+---------------+--
-------------+------------+---------+
only showing top 20 rows
```

In [96]:

```
# number of days in each week where spx moved by 40 in either direction
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select week_of_year, count(*) from SPXDaily where ((diff > 40 or diff < -40)
and (Date > "2022-01-01 00:00:00")) \
group by week_of_year \
order by week_of_year').show(50)
```

```
+------------+--------+
|week_of_year|count(1)|
+------------+--------+
|           1|       1|
|           2|       2|
|           3|       4|
|           4|       4|
|           5|       2|
|           6|       4|
|           7|       2|
|           8|       3|
|           9|       2|
|          10|       3|
|          11|       4|
|          12|       2|
|          13|       1|
|          14|       1|
|          15|       4|
|          16|       3|
|          17|       4|
|          18|       2|
|          19|       3|
|          20|       1|
|          21|       4|
|          22|       2|
|          23|       3|
|          24|       2|
|          25|       2|
|          26|       2|
|          27|       1|
|          28|       1|
|          29|       3|
|          30|       3|
|          31|       1|
|          32|       1|
|          34|       3|
|          35|       3|
|          36|       3|
|          37|       1|
|          38|       2|
|          39|       3|
|          40|       3|
|          41|       2|
|          42|       1|
|          43|       2|
|          44|       2|
|          45|       2|
|          48|       2|
|          49|       2|
|          50|       3|
|          52|       2|
+------------+--------+
```

In [97]:

```
# total weeks so far this year
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select max(week_of_year) from SPXDaily').show()
```

```
+------------------+
|max(week_of_year)|
+------------------+
|                52|
+------------------+
```

In [98]:

```
# number of days in each week where spx moved by less than 40 in either direction
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select week_of_year, count(*) from SPXDaily where ((diff < 40 and diff > -40)
and (Date > "2022-01-01 00:00:00")) \
group by week_of_year \
order by week_of_year').show(50)
```

```
+------------+--------+
|week_of_year|count(1)|
+------------+--------+
|           1|       4|
|           2|       3|
|           4|       1|
|           5|       3|
|           6|       1|
|           7|       3|
|           8|       1|
|           9|       3|
|          10|       2|
|          11|       1|
|          12|       3|
|          13|       4|
|          14|       4|
|          16|       2|
|          17|       1|
|          18|       3|
|          19|       2|
|          20|       4|
|          21|       1|
|          22|       2|
|          23|       2|
|          24|       3|
|          25|       2|
|          26|       3|
|          27|       3|
|          28|       4|
|          29|       2|
|          30|       2|
|          31|       4|
|          32|       4|
|          33|       5|
|          34|       2|
|          35|       2|
|          36|       1|
|          37|       4|
|          38|       3|
|          39|       2|
|          40|       2|
|          41|       3|
|          42|       4|
|          43|       3|
|          44|       3|
|          45|       3|
|          46|       5|
|          47|       4|
|          48|       3|
|          49|       3|
|          50|       2|
|          51|       5|
|          52|       2|
+------------+--------+
```

In [99]:

```
# check why in week 47, there were only 2 days over 40/-40 and 2 days were between -40 a
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily where week_of_year = 47').show()
```

```
+--------------------+----------------+----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+---------------+-----------+---------+
|                Date|            Open|            High|              Lo
w|            Close|    Volume|Dividends|Stock Splits|Month|            di
ff|  high_low_diff|week_of_year| week_day|
+--------------------+----------------+----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+---------------+-----------+---------+
|2021-11-22 00:00:...|          4712.0|  4743.830078125|   4682.16992187
5| 4682.93994140625|4441100000|      0.0|         0.0|   11|  29.060058593
75|    61.66015625|          47|   Monday|
|2021-11-23 00:00:...| 4678.47998046875| 4699.39013671875|    4652.6601562
5|  4690.7001953125|4277590000|      0.0|         0.0|   11| -12.220214843
75| 46.72998046875|          47|  Tuesday|
|2021-11-24 00:00:...| 4675.77978515625|  4702.8701171875| 4659.8901367187
5|  4701.4599609375|3418430000|      0.0|         0.0|   11| -25.680175781
25| 42.97998046875|          47|Wednesday|
|2021-11-26 00:00:...|  4664.6298828125|  4664.6298828125| 4585.4301757812
5|  4594.6201171875|3517700000|      0.0|         0.0|   11|    70.0097656
25| 79.19970703125|          47|   Friday|
|2022-11-21 00:00:...| 3956.22998046875|           3962.0|3933.34008789062
5| 3949.93994140625|3850690000|      0.0|         0.0|   11|    6.29003906
25|28.659912109375|          47|   Monday|
|2022-11-22 00:00:...|3965.510009765625|  4005.8798828125|  3956.879882812
5|   4003.580078125|3887990000|      0.0|         0.0|   11|-38.0700683593
75|           49.0|          47|  Tuesday|
|2022-11-23 00:00:...|4000.300048828125|4033.780029296875|3998.65991210937
5|4027.260009765625|3279720000|      0.0|         0.0|   11|  -26.95996093
75|  35.1201171875|          47|Wednesday|
|2022-11-25 00:00:...|4023.340087890625| 4034.02001953125|4020.76000976562
5|  4026.1201171875|1706460000|      0.0|         0.0|   11| -2.7800292968
75|13.260009765625|          47|   Friday|
+--------------------+----------------+----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+---------------+-----------+---------+
```

In [100]:

```
#number of days in each month where SPX moved by 40 in either direction
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select month, count(*) from SPXDaily where ((diff > 40 or diff < -40) \
and (Date > "2022-01-01 00:00:00")) \
group by month \
order by month').show()
```

```
+-----+--------+
|month|count(1)|
+-----+--------+
|    1|      12|
|    2|      10|
|    3|      12|
|    4|      12|
|    5|      10|
|    6|      10|
|    7|       9|
|    8|       7|
|    9|      10|
|   10|       8|
|   11|       6|
|   12|       7|
+-----+--------+
```

In [101]:

```
#number of days in each month where SPX moved by less than 40 in either direction
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select month, count(*) from SPXDaily where ((diff < 40 and diff > -40) \
and (Date > "2022-01-01 00:00:00")) \
group by month \
order by month').show(50)
```

```
+-----+--------+
|month|count(1)|
+-----+--------+
|    1|       8|
|    2|       9|
|    3|      11|
|    4|       8|
|    5|      11|
|    6|      11|
|    7|      11|
|    8|      16|
|    9|      11|
|   10|      13|
|   11|      15|
|   12|      14|
+-----+--------+
```

In [102]:

```
# number of days by week_day where spx moved by 40 in either direction
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select week_day, count(*) from SPXDaily where ((diff > 40 or  diff < -40) \
and (Date > "2022-01-01 00:00:00")) \
group by week_day \
order by week_day').show()
```

```
+---------+--------+
| week_day|count(1)|
+---------+--------+
|   Friday|      24|
|   Monday|      15|
| Thursday|      28|
|  Tuesday|      22|
|Wednesday|      24|
+---------+--------+
```

In [103]:

```
# number of days by week_day where spx moved less than 40 in either direction
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select week_day, count(*) from SPXDaily where ((diff < 40 and diff > -40) \
and (Date > "2022-01-01 00:00:00")) \
group by week_day \
order by week_day').show()
```

```
+---------+--------+
| week_day|count(1)|
+---------+--------+
|   Friday|      27|
|   Monday|      30|
| Thursday|      23|
|  Tuesday|      30|
|Wednesday|      28|
+---------+--------+
```

In [104]:

```
from pyspark.sql.functions import dayofmonth, hour, dayofyear, month, year, weekofyear,
from pyspark.sql.functions import date_format
from pyspark.sql.functions import col
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily \
          where ((diff between -10 and 10) \
                  and (date_format(Date, "EEEE") == "Monday") \
                  and year(Date) == 2022)').show()
```

```
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
+---------------+-----------+--------+
|                Date|             Open|             High|              Lo
w|            Close|    Volume|Dividends|Stock Splits|Month|          diff
|  high_low_diff|week_of_year|week_day|
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
+---------------+-----------+--------+
|2022-03-21 00:00:...| 4462.39990234375|          4481.75|  4424.299804687
5| 4461.18017578125|4869820000|      0.0|         0.0|    3|  1.2197265625
|  57.4501953125|          12|  Monday|
|2022-04-18 00:00:...|  4385.6298828125| 4410.31005859375|  4370.299804687
5| 4391.68994140625|3910490000|      0.0|         0.0|    4|-6.06005859375
| 40.01025390625|          16|  Monday|
|2022-05-16 00:00:...| 4013.02001953125|  4046.4599609375|3983.98999023437
5|4008.010009765625|4415030000|      0.0|         0.0|    5|5.010009765625
|62.469970703125|          20|  Monday|
|2022-07-25 00:00:...|3965.719970703125|3975.300048828125|  3943.459960937
5|3966.840087890625|3568340000|      0.0|         0.0|    7| -1.1201171875
|31.840087890625|          30|  Monday|
|2022-08-01 00:00:...|  4112.3798828125|  4144.9501953125| 4096.0200195312
5|  4118.6298828125|4202810000|      0.0|         0.0|    8|         -6.25
| 48.93017578125|          31|  Monday|
|2022-08-29 00:00:...|   4034.580078125|4062.989990234375|   4017.41992187
5|4030.610107421875|3396510000|      0.0|         0.0|    8|3.969970703125
|45.570068359375|          35|  Monday|
|2022-10-31 00:00:...| 3881.85009765625| 3893.72998046875|3863.17993164062
5| 3871.97998046875|4820620000|      0.0|         0.0|   10|  9.8701171875
|30.550048828125|          44|  Monday|
|2022-11-21 00:00:...| 3956.22998046875|           3962.0|3933.34008789062
5| 3949.93994140625|3850690000|      0.0|         0.0|   11|  6.2900390625
|28.659912109375|          47|  Monday|
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
+---------------+-----------+--------+
```

In [105]:

```
from pyspark.sql.functions import dayofmonth, hour, dayofyear, month, year, weekofyear,
from pyspark.sql.functions import date_format
from pyspark.sql.functions import col
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select * from SPXDaily \
           where ((date_format(Date, "EEEE") == "Monday") and \
                   year(Date) == 2022)').show(200)
```

```
+--------------------+----------------+-----------------+---------------
-+-----------------+----------+---------+------------+-----+------------
--+----------------+------------+--------+
|                Date|            Open|             High|             Lo
w|            Close|    Volume|Dividends|Stock Splits|Month|           di
ff|   high_low_diff|week_of_year|week_day|
+--------------------+----------------+-----------------+---------------
-+-----------------+----------+---------+------------+-----+------------
--+----------------+------------+--------+
|2022-01-03 00:00:...| 4778.14013671875| 4796.64013671875|   4758.16992187
5| 4796.56005859375|3831020000|      0.0|         0.0|    1|   -18.4199218
75|  38.47021484375|           1|  Monday|
|2022-01-10 00:00:...|    4655.33984375| 4673.02001953125|   4582.24023437
5|  4670.2900390625|4511810000|      0.0|         0.0|    1|  -14.95019531
25|  90.77978515625|           2|  Monday|
|2022-01-24 00:00:...| 4356.31982421875| 4417.35009765625|  4222.620117187
5|  4410.1298828125|6928110000|      0.0|         0.0|    1| -53.810058593
75| 194.72998046875|           4|  Monday|
|2022-01-31 00:00:...|  4431.7900390625| 4516.89013671875| 4414.0200195312
5|  4515.5498046875|5098610000|      0.0|         0.0|    1|   -83.7597656
25|  102.8701171875|           5|  Monday|
|2022-02-07 00:00:...|          4505.75| 4521.85986328125| 4471.4702148437
5|  4483.8701171875|4228480000|      0.0|         0.0|    2|   21.87988281
25|   50.3896484375|           6|  Monday|
|2022-02-14 00:00:...| 4412.60986328125| 4426.22021484375|    4364.8398437
5|   4401.669921875|4600390000|      0.0|         0.0|    2|  10.939941406
25|  61.38037109375|           7|  Monday|
|2022-02-28 00:00:...|   4354.169921875|    4388.83984375|  4315.120117187
5| 4373.93994140625|6071370000|      0.0|         0.0|    2| -19.770019531
25|   73.7197265625|           9|  Monday|
|2022-03-07 00:00:...|   4327.009765625|   4327.009765625| 4199.8500976562
5|    4201.08984375|6940470000|      0.0|         0.0|    3|   125.9199218
75| 127.15966796875|          10|  Monday|
|2022-03-14 00:00:...|          4202.75| 4247.56982421875| 4161.7202148437
5| 4173.10986328125|5574920000|      0.0|         0.0|    3|  29.640136718
75|    85.849609375|          11|  Monday|
|2022-03-21 00:00:...| 4462.39990234375|          4481.75|  4424.299804687
5| 4461.18017578125|4869820000|      0.0|         0.0|    3|    1.21972656
25|   57.4501953125|          12|  Monday|
|2022-03-28 00:00:...|    4541.08984375| 4575.64990234375| 4517.6899414062
5| 4575.52001953125|4312260000|      0.0|         0.0|    3| -34.430175781
25|   57.9599609375|          13|  Monday|
|2022-04-04 00:00:...| 4547.97021484375|           4583.5|  4539.209960937
5| 4582.64013671875|4547350000|      0.0|         0.0|    4|   -34.6699218
75|   44.2900390625|          14|  Monday|
|2022-04-11 00:00:...| 4462.64013671875| 4464.35009765625|  4408.379882812
5| 4412.52978515625|4266290000|      0.0|         0.0|    4|   50.11035156
25|  55.97021484375|          15|  Monday|
|2022-04-18 00:00:...|  4385.6298828125| 4410.31005859375|  4370.299804687
5| 4391.68994140625|3910490000|      0.0|         0.0|    4|  -6.060058593
75|  40.01025390625|          16|  Monday|
|2022-04-25 00:00:...|    4255.33984375| 4299.02001953125| 4200.8198242187
5|  4296.1201171875|5240040000|      0.0|         0.0|    4|  -40.78027343
75|   98.2001953125|          17|  Monday|
|2022-05-02 00:00:...| 4130.60986328125| 4169.81005859375|4062.51000976562
5|  4155.3798828125|5163790000|      0.0|         0.0|    5| -24.770019531
25|107.300048828125|          18|  Monday|
|2022-05-09 00:00:...| 4081.27001953125| 4081.27001953125| 3975.4799804687
5|3991.239990234375|5954520000|      0.0|         0.0|    5| 90.0300292968
75|  105.7900390625|          19|  Monday|
|2022-05-16 00:00:...| 4013.02001953125|  4046.4599609375|3983.98999023437
```

```
5|4008.010009765625|4415030000|      0.0|           0.0|    5|  5.0100097656
25| 62.469970703125|          20|  Monday|
|2022-05-23 00:00:...|   3919.419921875|  3981.8798828125|  3909.040039062
5|          3973.75|4420030000|      0.0|           0.0|    5|  -54.3300781
25|     72.83984375|          21|  Monday|
|2022-06-06 00:00:...| 4134.72021484375| 4168.77978515625| 4109.1801757812
5| 4121.43017578125|4332700000|      0.0|           0.0|    6|   13.29003906
25|    59.599609375|          23|  Monday|
|2022-06-13 00:00:...| 3838.14990234375| 3838.14990234375|3734.30004882812
5|  3749.6298828125|5636890000|      0.0|           0.0|    6|  88.520019531
25|103.849853515625|          24|  Monday|
|2022-06-27 00:00:...|3920.760009765625|3927.719970703125|3889.65991210937
5|3900.110107421875|4325310000|      0.0|           0.0|    6|  20.649902343
75|  38.06005859375|          26|  Monday|
|2022-07-11 00:00:...| 3880.93994140625| 3880.93994140625|3847.21997070312
5|3854.429931640625|3423480000|      0.0|           0.0|    7| 26.5100097656
25| 33.719970703125|          28|  Monday|
|2022-07-18 00:00:...|  3883.7900390625| 3902.43994140625|  3818.629882812
5| 3830.85009765625|4046870000|      0.0|           0.0|    7|  52.939941406
25|  83.81005859375|          29|  Monday|
|2022-07-25 00:00:...|3965.719970703125|3975.300048828125|  3943.459960937
5|3966.840087890625|3568340000|      0.0|           0.0|    7|   -1.12011718
75| 31.840087890625|          30|  Monday|
|2022-08-01 00:00:...|  4112.3798828125|  4144.9501953125| 4096.0200195312
5|  4118.6298828125|4202810000|      0.0|           0.0|    8|           -6.
25|  48.93017578125|          31|  Monday|
|2022-08-08 00:00:...| 4155.93017578125|  4186.6201171875| 4128.9702148437
5| 4140.06005859375|4221090000|      0.0|           0.0|    8|   15.87011718
75|  57.64990234375|          32|  Monday|
|2022-08-15 00:00:...|  4269.3701171875|  4301.7900390625| 4256.8999023437
5| 4297.14013671875|3696830000|      0.0|           0.0|    8| -27.770019531
25|  44.89013671875|          33|  Monday|
|2022-08-22 00:00:...|   4195.080078125|   4195.080078125| 4129.8598632812
5|   4137.990234375|3907430000|      0.0|           0.0|    8|     57.089843
75|  65.22021484375|          34|  Monday|
|2022-08-29 00:00:...|   4034.580078125|4062.989990234375|   4017.41992187
5|4030.610107421875|3396510000|      0.0|           0.0|    8|  3.9699707031
25| 45.570068359375|          35|  Monday|
|2022-09-12 00:00:...|   4083.669921875| 4119.27978515625|   4083.66992187
5|    4110.41015625|3814200000|      0.0|           0.0|    9|   -26.7402343
75|  35.60986328125|          37|  Monday|
|2022-09-19 00:00:...|3849.909912109375|3900.449951171875|           3838.
5|3899.889892578125|3766850000|      0.0|           0.0|    9| -49.979980468
75| 61.949951171875|          38|  Monday|
|2022-09-26 00:00:...|3682.719970703125|   3715.669921875|3644.76000976562
5|  3655.0400390625|4886140000|      0.0|           0.0|    9| 27.6799316406
25| 70.909912109375|          39|  Monday|
|2022-10-03 00:00:...|3609.780029296875| 3698.35009765625|3604.92993164062
5|3678.429931640625|4806680000|      0.0|           0.0|   10| -68.649902343
75| 93.420166015625|          40|  Monday|
|2022-10-10 00:00:...|3647.510009765625|   3652.169921875| 3588.1000976562
5|3612.389892578125|3834320000|      0.0|           0.0|   10|   35.12011718
75|  64.06982421875|          41|  Monday|
|2022-10-17 00:00:...| 3638.64990234375| 3689.72998046875| 3638.6499023437
5|3677.949951171875|4352780000|      0.0|           0.0|   10|-39.3000488281
25|    51.080078125|          42|  Monday|
|2022-10-24 00:00:...|3762.010009765625|3810.739990234375| 3741.6499023437
5|3797.340087890625|4747930000|      0.0|           0.0|   10|   -35.3300781
25| 69.090087890625|          43|  Monday|
|2022-10-31 00:00:...| 3881.85009765625| 3893.72998046875|3863.17993164062
5| 3871.97998046875|4820620000|      0.0|           0.0|   10|    9.87011718
```

```
75| 30.550048828125|           44|  Monday|
|2022-11-07 00:00:...|  3780.7099609375|3813.949951171875|3764.69995117187
5|3806.800048828125|4341620000|      0.0|            0.0|    11|-26.0900878906
25|            49.25|           45|  Monday|
|2022-11-14 00:00:...|3977.969970703125|4008.969970703125| 3956.3999023437
5|           3957.25|4561930000|      0.0|            0.0|    11| 20.7199707031
25| 52.570068359375|           46|  Monday|
|2022-11-21 00:00:...| 3956.22998046875|           3962.0|3933.34008789062
5| 3949.93994140625|3850690000|      0.0|            0.0|    11|     6.29003906
25| 28.659912109375|           47|  Monday|
|2022-11-28 00:00:...|4005.360107421875| 4012.27001953125| 3955.7700195312
5| 3963.93994140625|3615430000|      0.0|            0.0|    11| 41.4201660156
25|             56.5|           48|  Monday|
|2022-12-05 00:00:...| 4052.02001953125|4052.449951171875|3984.48999023437
5|3998.840087890625|4280820000|      0.0|            0.0|    12| 53.1799316406
25|   67.9599609375|           49|  Monday|
|2022-12-12 00:00:...|  3939.2900390625|  3990.7099609375|3935.30004882812
5| 3990.56005859375|3904130000|      0.0|            0.0|    12| -51.270019531
25| 55.409912109375|           50|  Monday|
|2022-12-19 00:00:...|  3853.7900390625|3854.860107421875|  3800.040039062
5|3817.659912109375|3969610000|      0.0|            0.0|    12| 36.1301269531
25| 54.820068359375|           51|  Monday|
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+----------------+-----------+--------+
```

In [106]:

```python
from pyspark.sql.functions import dayofmonth, hour, dayofyear, month, year, weekofyear,
from pyspark.sql.functions import date_format
from pyspark.sql.functions import col
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select *, date_format(Date, "EEEE") from SPXDaily \
          where ((date_format(Date, "EEEE") in ("Monday", "Friday")) and \
                  year(Date) == 2022)').show(200)
```

```
+--------------------+-----------------+-----------------+-------------
----+-----------------+----------+---------+------------+-----+--------
--------+----------------+------------+--------+----------------------
--------------------+
|                Date|             Open|             High|
Low|            Close|    Volume|Dividends|Stock Splits|Month|
diff|   high_low_diff|week_of_year|week_day|date_format(CAST(Date AS TI
MESTAMP), EEEE)|
+--------------------+-----------------+-----------------+-------------
----+-----------------+----------+---------+------------+-----+--------
--------+----------------+------------+--------+----------------------
--------------------+
|2022-01-03 00:00:...| 4778.14013671875| 4796.64013671875|   4758.16992
1875| 4796.56005859375|3831020000|      0.0|            0.0|    1|   -18.4
19921875|  38.47021484375|           1|  Monday|
Monday|
|2022-01-07 00:00:...|    4697.66015625|  4707.9501953125|   4662.74023
4375| 4677.02978515625|4181510000|      0.0|            0.0|    1|  20.630
37109375|   45.2099609375|           1|  Friday|
```

In [107]:

```
spark.sql('select month(Date), date_format(Date, "EEEE"), avg(diff) from SPXDaily \
           where year(Date) == 2022 \
           group by month(Date), date_format(Date, "EEEE") \
           order by month(Date), date_format(Date, "EEEE")').show(200)
```

```
+------------------------+-------------------------------------------+----
---------------+
|month(CAST(Date AS DATE))|date_format(CAST(Date AS TIMESTAMP), EEEE)|
avg(diff)|
+------------------------+-------------------------------------------+----
---------------+
|                       1|                                     Friday|   -
6.6124267578125|
|                       1|                                     Monday|  -4
2.7349853515625|
|                       1|                                   Thursday|   4
7.6402587890625|
|                       1|                                    Tuesday|
8.090087890625|
|                       1|                                  Wednesday|
50.85498046875|
|                       2|                                     Friday|
4.8299560546875|
|                       2|                                     Monday|  4.
349934895833333|
|                       2|                                   Thursday|   1
2.5001220703125|
|                       2|                                    Tuesday|   -
20.574951171875|
|                       2|                                  Wednesday|
4.2501220703125|
|                       3|                                     Friday|
3.12744140625|
|                       3|                                     Monday|
30.58740234375|
|                       3|                                   Thursday|
-3.45615234375|
|                       3|                                    Tuesday|
-11.2080078125|
|                       3|                                  Wednesday|
-25.97392578125|
|                       4|                                     Friday|
59.050048828125|
|                       4|                                     Monday|   -
7.8499755859375|
|                       4|                                   Thursday|   1
5.3900146484375|
|                       4|                                    Tuesday|   2
9.7073974609375|
|                       4|                                  Wednesday|
-5.97509765625|
|                       5|                                     Friday|
-27.392578125|
|                       5|                                     Monday|
3.9849853515625|
|                       5|                                   Thursday|
5.5999755859375|
|                       5|                                    Tuesday|
0.3958984375|
|                       5|                                  Wednesday|
3.76763916015625|
|                       6|                                     Friday|
0.9073486328125|
|                       6|                                     Monday| 40.
819986979166664|
|                       6|                                   Thursday|
8.683984375|
|                   6|                                                 Tuesday|
1.45001220703125|
|                   6|                                               Wednesday|
6.843994140625|
|                   7|                                                  Friday|   -
21.353955078125|
|                   7|                                                  Monday| 26.
109944661458332|
|                   7|                                                Thursday|  -3
9.9849853515625|
|                   7|                                                 Tuesday|
-12.35498046875|
|                   7|                                               Wednesday|  -3
2.9925537109375|
|                   8|                                                  Friday|  2
3.61517333984375|
|                   8|                                                  Monday|
8.581982421875|
|                   8|                                                Thursday|
-8.357666015625|
|                   8|                                                 Tuesday|
13.673876953125|
|                   8|                                               Wednesday|
-7.72412109375|
|                   9|                                                  Friday|
23.073876953125|
|                   9|                                                  Monday|-16.
346761067708332|
|                   9|                                                Thursday|
5.12197265625|
|                   9|                                                 Tuesday|
46.39501953125|
|                   9|                                               Wednesday| -1
5.33758544921875|
|                  10|                                                  Friday|  -
3.50750732421875|
|                  10|                                                  Monday|   -
19.657958984375|
|                  10|                                                Thursday| -1
7.85748291015625|
|                  10|                                                 Tuesday| -2
2.71002197265625|
|                  10|                                               Wednesday|  -
3.22747802734375|
|                  11|                                                  Friday|   -
8.6275634765625|
|                  11|                                                  Monday|  1
0.58502197265625|
|                  11|                                                Thursday|-36.
806722005208336|
|                  11|                                                 Tuesday|
3.553955078125|
|                  11|                                               Wednesday|
4.7439453125|
|                  12|                                                  Friday|
-2.668017578125|
|                  12|                                                  Monday| 12.
680013020833334|
|                  12|                                                Thursday|
8.902001953125|
|                       12|                                  Tuesday|   2
7.00994873046875|
|                       12|                                Wednesday|
6.7425537109375|
+-------------------------+------------------------------------------+----
----------------+
```

In [108]:

```
from pyspark.sql.functions import dayofmonth, hour, dayofyear, month, year, weekofyear,
from pyspark.sql.functions import date_format
from pyspark.sql.functions import col
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select *, date_format(Date, "EEEE") from SPXDaily \
           where ((date_format(Date, "EEEE") in ("Thursday")) and \
                   year(Date) == 2022) and \
                   month(Date) == 7').show(200)
```

```
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+---------------+------------+--------+---------------------------------
---------+
|                Date|             Open|             High|              Lo
w|            Close|    Volume|Dividends|Stock Splits|Month|            di
ff|  high_low_diff|week_of_year|week_day|date_format(CAST(Date AS TIMESTAM
P), EEEE)|
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+---------------+------------+--------+---------------------------------
---------+
|2022-07-07 00:00:...| 3858.85009765625|  3910.6298828125| 3858.8500976562
5|  3902.6201171875|4057770000|      0.0|         0.0|    7| -43.770019531
25| 51.77978515625|          27|Thursday|
Thursday|
|2022-07-14 00:00:...|3763.989990234375|3796.409912109375| 3721.5600585937
5|  3790.3798828125|4199690000|      0.0|         0.0|    7|-26.3898925781
25|74.849853515625|          28|Thursday|
Thursday|
|2022-07-21 00:00:...|3955.469970703125|  3999.2900390625|3927.63989257812
5|3998.949951171875|4132790000|      0.0|         0.0|    7| -43.479980468
75|71.650146484375|          29|Thursday|
Thursday|
|2022-07-28 00:00:...|  4026.1298828125|4078.949951171875|3992.96997070312
5|4072.429931640625|4413000000|      0.0|         0.0|    7|-46.3000488281
25| 85.97998046875|          30|Thursday|
Thursday|
+--------------------+-----------------+-----------------+----------------
-+-----------------+----------+---------+------------+-----+--------------
--+---------------+------------+--------+---------------------------------
---------+
```

In [109]:

```
daily_df.createOrReplaceTempView("SPXDaily")
spark.sql('select min(Date), max(Date) from SPXDaily where diff < 20 or diff < -20' ).sh
```

```
+--------------------+--------------------+
|           min(Date)|           max(Date)|
+--------------------+--------------------+
|2021-01-05 00:00:...|2022-12-30 00:00:...|
+--------------------+--------------------+
```

In [110]:

```
import yfinance as yf
import csv

ticker = 'TSLA'
#^GSPC #TSLA
start_dt = '2022-8-15'
end_dt = '2022-8-24'
frequency = '1d'

#get data for TSLA
data = yf.Ticker(ticker)
#ticker_name = data.info['longName']

#get weekly historical prices for this ticker
df = data.history(interval = frequency, start = start_dt, end = end_dt)
# write data into a csv file
df.to_csv('spx.csv')

# Let Spark know about the header and infer the Schema types!
df = spark.read.csv('spx.csv',inferSchema=True,header=True)
df.head(5)
```

Out[110]:

```
[Row(Date='2022-08-15 00:00:00-04:00', Open=301.78668212890625, High=313.1
333312988281, Low=301.2300109863281, Close=309.32000732421875, Volume=8935
9200, Dividends=0.0, Stock Splits=0.0),
 Row(Date='2022-08-16 00:00:00-04:00', Open=311.6666564941406, High=314.66
66564941406, Low=302.8833312988281, Close=306.5633239746094, Volume=881364
00, Dividends=0.0, Stock Splits=0.0),
 Row(Date='2022-08-17 00:00:00-04:00', Open=303.39666748046875, High=309.6
5667724609375, Low=300.0333251953125, Close=303.9966735839844, Volume=6876
6000, Dividends=0.0, Stock Splits=0.0),
 Row(Date='2022-08-18 00:00:00-04:00', Open=306.0, High=306.5, Low=301.853
33251953125, Close=302.8699951171875, Volume=47500500, Dividends=0.0, Stoc
k Splits=0.0),
 Row(Date='2022-08-19 00:00:00-04:00', Open=299.0, High=300.3599853515625,
Low=292.5, Close=296.6666564941406, Volume=61395300, Dividends=0.0, Stock
Splits=0.0)]
```

In [111]:

```
df.head()[4]
```

Out[111]:

```
309.32000732421875
```

In [112]:

```
from pyspark.sql.functions import dayofmonth, hour, dayofyear, month, year, weekofyear,
from pyspark.sql.functions import date_format, col
df.select(dayofyear(df['Date']), weekofyear(df['Date']), dayofweek(df['Date']), date_for
```

```
+---------------+----------------+---------------+-----------------------+
|dayofyear(Date)|weekofyear(Date)|dayofweek(Date)|date_format(Date, EEEE)|
+---------------+----------------+---------------+-----------------------+
|            227|              33|              2|                 Monday|
|            228|              33|              3|                Tuesday|
|            229|              33|              4|              Wednesday|
|            230|              33|              5|               Thursday|
|            231|              33|              6|                 Friday|
|            234|              34|              2|                 Monday|
|            235|              34|              3|                Tuesday|
+---------------+----------------+---------------+-----------------------+
```

In [113]:

```
df.filter((dayofmonth(df['Date']) > 20) & (month(df['Date']) > 3)).show()
```

```
+--------------------+-----------------+-----------------+----------------
-+-----------------+--------+---------+------------+
|                Date|             Open|             High|              Lo
w|            Close|  Volume|Dividends|Stock Splits|
+--------------------+-----------------+-----------------+----------------
-+-----------------+--------+---------+------------+
|2022-08-22 00:00:...| 291.913330078125|292.3999938964844|286.296661376953
1| 289.913330078125|55843200|      0.0|         0.0|
|2022-08-23 00:00:...|291.4533386230469|  298.82666015625|  287.9233398437
5|296.4533386230469|63984900|      0.0|         0.0|
+--------------------+-----------------+-----------------+----------------
-+-----------------+--------+---------+------------+
```

In [114]:

```
#df.filter((df['Volume'].between (40000000, 42780400))).show()
#df.where((df['Volume'].between (40000000, 42780400))).show()
df.filter((df['Volume'] >= 40000000) & (df['Volume'] < 63984900)).show()
```

```
+--------------------+----------------+-----------------+-----------------
-+-----------------+--------+---------+------------+
|                Date|            Open|             High|               Lo
w|            Close|  Volume|Dividends|Stock Splits|
+--------------------+----------------+-----------------+-----------------
-+-----------------+--------+---------+------------+
|2022-08-18 00:00:...|           306.0|            306.5|301.8533325195312
5|302.8699951171875|47500500|      0.0|         0.0|
|2022-08-19 00:00:...|           299.0|300.3599853515625|             292.
5|296.6666564941406|61395300|      0.0|         0.0|
|2022-08-22 00:00:...|291.913330078125|292.3999938964844| 286.296661376953
1| 289.913330078125|55843200|      0.0|         0.0|
+--------------------+----------------+-----------------+-----------------
-+-----------------+--------+---------+------------+
```

In [115]:

```
month_df = df.withColumn('Month', month(df['Date']))
month_df.groupBy('Month').mean()[['avg(Month)', 'avg(Volume)']].orderBy('avg(Month)').sh

# newdf = df.withColumn("Year",year(df['Date']))
# newdf.groupBy("Year").mean()[['avg(Year)','avg(Close)']].show()
```

```
+----------+------------------+
|avg(Month)|       avg(Volume)|
+----------+------------------+
|       8.0|6.785507142857143E7|
+----------+------------------+
```

In [116]:

```
month_df = df.withColumn('Month', month(df['Date']))
month_df.groupBy('Month').mean()[['avg(Month)', 'avg(Volume)']].sort('avg(Month)').show(
```

```
+----------+------------------+
|avg(Month)|       avg(Volume)|
+----------+------------------+
|       8.0|6.785507142857143E7|
+----------+------------------+
```

In [117]:

```
month_df = df.withColumn('Month', month(df['Date']))
month_df.groupBy('Month').mean()[['avg(Month)', 'avg(Volume)']].orderBy('avg(Month)').sh
```

```
+----------+------------------+
|avg(Month)|       avg(Volume)|
+----------+------------------+
|       8.0|6.785507142857143E7|
+----------+------------------+
```