DSC540-T302 Data Preparation

Project Milestone 2

Saravanan Janarthanan

In [1]:
```
# Load numpy and pandas modules
import pandas as pd
import numpy as py
```

In [2]:
```
#  Load the data file in a dataframe
sp500_df = pd.read_csv('SNP_500_Performance_data.csv')
sp500_df.head()
```

Out[2]:

	Symbol	Name	Market Cap	P/E	EPS	Net Income	Beta	Dividend	Div Yield	Extended Hr Last Price
0	A	Agilent Technologies	4.223508e+10	26.32	5.37	1240000000	1.11	0.922	0.64%	144.12
1	AAL	American Airlines Gp	9.009457e+09	5.24	2.64	822000000	1.57	NaN	0.00%	13.75
2	AAPL	Apple Inc	2.620000e+12	26.30	6.42	96995000000	1.27	0.960	0.57%	169.47
3	ABBV	Abbvie Inc	3.010000e+11	15.11	11.11	4863000000	0.58	5.990	3.52%	169.99
4	ABNB	Airbnb Inc Cl A	1.050000e+11	36.85	4.31	4792000000	1.25	NaN	0.00%	161.77

5 rows × 30 columns

Step #1- Change header

- View the header names
- identfy the ones that needs to be changed
- Retrieve the current headers in a list
- Perform the changes to the identified headers in the list
- Set the new headers using the modfiied list values

In [3]:
```
header_nms = sp500_df.columns.tolist()
for idx, hdr in enumerate(header_nms):
    print(idx, " : ", hdr)
```

```
0  :  Symbol
1  :  Name
2  :  Market Cap
3  :  P/E
4  :  EPS
5  :  Net Income
6  :  Beta
7  :  Dividend
8  :  Div Yield
9  :  Extended Hr Last Price
10  :  Extended Hr Change
11  :  %Chg (Ext)
12  :  Time
13  :  Last Price
14  :  Change
15  :  %Chg
16  :  Opinion
17  :  20D Rel Str
18  :  20D His Vol
19  :  20D Avg Vol
20  :  52W Low
21  :  52W High
22  :  Wtd Alpha
23  :  YTD %Chg
24  :  1M %Chg
25  :  3M %Chg
26  :  52W %Chg
27  :  Sector
28  :  ISIN
29  :  Volume
```

In [4]:

```python
# Change 4th column (index 3) header from "P/E" to "Price Earnings"
# Change 12th column (index 11) header from "%Chg (Ext)" to "Extended Hr %Change"
# Change 18th column (index 17) header from "20D Rel Str" to "20D Relative Strength"

header_nms[3] = "Price Earnings"
header_nms[11] = "Extended Hr %Change"
header_nms[17] = "20D Relative Strength"

sp500_df.columns = header_nms
for idx, hdr in enumerate(header_nms):
    print(idx, " : ", hdr)
```

```
0  :  Symbol
1  :  Name
2  :  Market Cap
3  :  Price Earnings
4  :  EPS
5  :  Net Income
6  :  Beta
7  :  Dividend
8  :  Div Yield
9  :  Extended Hr Last Price
10  :  Extended Hr Change
11  :  Extended Hr %Change
12  :  Time
13  :  Last Price
14  :  Change
15  :  %Chg
16  :  Opinion
17  :  20D Relative Strength
18  :  20D His Vol
19  :  20D Avg Vol
20  :  52W Low
21  :  52W High
22  :  Wtd Alpha
23  :  YTD %Chg
24  :  1M %Chg
25  :  3M %Chg
26  :  52W %Chg
27  :  Sector
28  :  ISIN
29  :  Volume
```

In [5]:
```
sp500_df.head()
```

Out[5]:

	Symbol	Name	Market Cap	Price Earnings	EPS	Net Income	Beta	Dividend	Div Yield	Extende Hr La Pric
0	A	Agilent Technologies	4.223508e+10	26.32	5.37	1240000000	1.11	0.922	0.64%	144.1
1	AAL	American Airlines Gp	9.009457e+09	5.24	2.64	822000000	1.57	NaN	0.00%	13.7
2	AAPL	Apple Inc	2.620000e+12	26.30	6.42	96995000000	1.27	0.960	0.57%	169.4
3	ABBV	Abbvie Inc	3.010000e+11	15.11	11.11	4863000000	0.58	5.990	3.52%	169.9
4	ABNB	Airbnb Inc Cl A	1.050000e+11	36.85	4.31	4792000000	1.25	NaN	0.00%	161.7

5 rows × 30 columns

Step #2 - Find NAN in columns

- Identify the NAN or null values count present in each column

- If the column is a key column and analyze and remove those records

In [6]:
```
sp500_df.isna().sum()
```

Out[6]:
```
Symbol                    0
Name                      0
Market Cap                0
Price Earnings            0
EPS                       0
Net Income                0
Beta                      2
Dividend                 99
Div Yield                 0
Extended Hr Last Price    0
Extended Hr Change        0
Extended Hr %Change       0
Time                      0
Last Price                0
Change                    0
%Chg                      0
Opinion                   1
20D Relative Strength     0
20D His Vol               0
20D Avg Vol               0
52W Low                   0
52W High                  0
Wtd Alpha                 0
YTD %Chg                  0
1M %Chg                   0
3M %Chg                   0
52W %Chg                  2
Sector                    0
ISIN                      0
Volume                    0
dtype: int64
```

In [7]:
```
sp500_df.isnull().sum()
```

Out[7]:
```
Symbol                     0
Name                       0
Market Cap                 0
Price Earnings             0
EPS                        0
Net Income                 0
Beta                       2
Dividend                  99
Div Yield                  0
Extended Hr Last Price     0
Extended Hr Change         0
Extended Hr %Change        0
Time                       0
Last Price                 0
Change                     0
%Chg                       0
Opinion                    1
20D Relative Strength      0
20D His Vol                0
20D Avg Vol                0
52W Low                    0
52W High                   0
Wtd Alpha                  0
YTD %Chg                   0
1M %Chg                    0
3M %Chg                    0
52W %Chg                   2
Sector                     0
ISIN                       0
Volume                     0
dtype: int64
```

None of the key columns, like 'Symbol', 'Name' or 'Last Price' , 'Change', '%Chg' have null or NAN values, hence no records are removed or deleted

---

Step #3 - Convert float number to int64 to remove expontential listing or displaying

- Column 'Market Cap' displays in exponential form that impacts readability.
- Convert that column values to integer 64 base to display the whole numbers

In [8]:
```
sp500_df["Market Cap"].head(10)
```

Out[8]:
```
0    4.223508e+10
1    9.009457e+09
2    2.620000e+12
3    3.010000e+11
4    1.050000e+11
5    1.930000e+11
6    3.560126e+10
7    2.230000e+11
8    2.170000e+11
9    9.658304e+10
Name: Market Cap, dtype: float64
```

In [9]:
```
sp500_df["Market Cap"] =  sp500_df["Market Cap"].astype('int64')
```

In [10]:
```
sp500_df["Market Cap"].head(10)
```

Out[10]:
```
0      42235084000
1       9009457000
2    2620000000000
3     301000000000
4     105000000000
5     193000000000
6      35601260000
7     223000000000
8     217000000000
9      96583040000
Name: Market Cap, dtype: int64
```

Step #4 - Replace special character, replace with a no character or null character

- Column 'Volume' has '?' character in betweeen the values
- Use the String replace method to replace the '?' character with ''
- Print the column values to validate

In [11]:
```
sp500_df["Volume"].head(10)
```

Out[11]:
```
0     1.656?M
1    30.632?M
2    42.104?M
3      7.03?M
4     2.226?M
5     5.591?M
6     2.591?M
7     3.695?M
8     4.973?M
9     2.087?M
Name: Volume, dtype: object
```

In [12]:
```
sp500_df['Volume'] = sp500_df['Volume'].str.replace('?', '')
```

In [13]:
```
sp500_df["Volume"].head(25)
```

Out[13]:
```
0       1.656M
1      30.632M
2      42.104M
3        7.03M
4       2.226M
5       5.591M
6       2.591M
7       3.695M
8       4.973M
9       2.087M
10      3.348M
11       1.42M
12       1.33M
13      1.585M
14      2.432M
15      7.587M
16      1.888M
17          3M
18    355.516K
19    690.763K
20      919.2K
21      1.901M
22    595.357K
23    999.024K
24    351.903K
Name: Volume, dtype: object
```

Step #5 introduce a new column derive based on 'Opinion' column value

- Convert the confidence percentage into a number rating betweeen 1 to 10)
- For Buy confidence signal convert it to a psotive integer between 1 to 10 and
- for Sell signal convert it to a negative integer ranging between 1 to 10
- Create a method to recieve the stirng value and split them and identify the 'Buy' or'Sell' value and use it to denote positive or negative rating
- use the magnitude value to convert % to 1 to 10 range value

In [14]:
```
import math

# Declare a method to process the input string and return a postive or negative rating
def convertOpinionBuySellToratingScale(x):
    if x == "" or  type(x) == float or  x.find('%') == -1:
        return ""
    else:
        split_val = x.split('%')
        pos_neg_ind = -1 if split_val[1] == " Sell" else 1
        return  round(int(split_val[0])/10)*pos_neg_ind
```

In [15]:
```
# create a new column to set the derived values
sp500_df['BuySell_rating'] = [convertOpinionBuySellToratingScale(x) for x in sp500_df[
sp500_df[['Opinion', 'BuySell_rating']].head(10)
```

Out[15]:

	Opinion	BuySell_rating
0	88% Buy	9
1	8% Sell	-1
2	88% Sell	-9
3	56% Buy	6
4	80% Buy	8
5	40% Buy	4
6	100% Buy	10
7	8% Buy	1
8	72% Sell	-7
9	100% Buy	10

In [16]:

```
# Display the modified dataframe column hearders,  values
# change setting to display all the columns
pd.set_option('display.max_columns', None)

sp500_df.head(100)
```

Out[16]:

	Symbol	Name	Market Cap	Price Earnings	EPS	Net Income	Beta	Dividend	Div Yield	Exter Hr
0	A	Agilent Technologies	42235084000	26.32	5.37	1240000000	1.11	0.922	0.64%	14
1	AAL	American Airlines Gp	9009457000	5.24	2.64	822000000	1.57	NaN	0.00%	1
2	AAPL	Apple Inc	2620000000000	26.30	6.42	96995000000	1.27	0.960	0.57%	16
3	ABBV	Abbvie Inc	301000000000	15.11	11.11	4863000000	0.58	5.990	3.52%	16
4	ABNB	Airbnb Inc Cl A	105000000000	36.85	4.31	4792000000	1.25	NaN	0.00%	16
...	...	...	...	...	...	...	...	...	...	
95	CINF	Cincinnati Financial	19037930000	19.91	6.04	1843000000	0.63	3.060	2.52%	12
96	CL	Colgate-Palmolive Company	72236128000	27.06	3.23	2300000000	0.42	1.920	2.18%	8
97	CLX	Clorox Company	18112030000	25.14	5.83	149000000	0.43	4.780	3.28%	14
98	CMA	Comerica Inc	6974274000	6.85	7.70	881000000	1.27	2.840	5.40%	5
99	CMCSA	Comcast Corp A	162000000000	10.36	3.97	15389000000	0.98	1.180	2.89%	4

100 rows × 31 columns

- What changes were made to the data?
- Are there any legal or regulatory guidelines for your data or project topic?
- What risks could be created based on the transformations done?
- Did you make any assumptions in cleaning/transforming the data?
- How was your data sourced / verified for credibility?
- Was your data acquired in an ethical way?
- How would you mitigate any of the ethical implications you have identified?

Following are the Changes made to the data

- The 4th, 12th and 18th column header were renamed, , Changed "P/E" to "Price Earnings", FChanged "%Chg (Ext)" to "Extended Hr %Change" and changed "20D Rel Str" to "20D Relative Strength"
- Checked for any NaN values in the critical columns; identified NaN values in non-critical columns, therefore no records were removed.

- Changed float numbers to int64 to eliminate exponential notation in the 'Market Cap' column.
- Values in the 'Volume' column contained a special character '?" within the values. Adjustments were made to eliminate this special character.
- The 'Opinion' column contained buy and sell suggestions with a percentage magnitude. A new column titled 'Buy Sell Rating' was created to generate a positive or negative rating ranging from 1 to 10 based on the values in the 'Opinion' column.

Stock information is subject to the regulations of the country where the stock is listed. Companies are typically mandated to disclose specific information to both investors and the public. This information is made available through stock exchanges for public access and scrutiny.

The transformations made were cosmetic and did not change any values or the primary purpose of the data.

No assumptions were while transforming the data.

The data was obtained from a reputable website, which in turn sources it from government-approved stock exchanges. This data is disseminated across numerous commercial websites via feeds received from authoritative channels.

Apart from the 'Opinion' column, which offers buy or sell suggestions, the transformed column is derived from the same. These buy and sell recommendations come with a disclaimer, advising investors to use them as guidance and conduct their due diligence before making any decisions to buy or sell the stock instrument.

In [ ]:

In [ ]: