Web Scraping and Data Analytics_Assignment 4

March 17, 2024

[30]:

```
import requests
from bs4 import BeautifulSoup
import os
import time


os.makedirs("hockey data")
#creating a directory hockey data

def scrape_page(page_number):
    html_page = requests.get(f'https://www.scrapethissite.com/pages/forms/?
 ↪page_num={page_number}').text
    soup = BeautifulSoup(html_page, 'lxml')
    return soup.find_all('tr', class_='team')

'''In the above function, we have customised the url by passing a integer to␣
 ↪the fuction. We can invoke this function to
move through various pages and then using the beautiful soup we then find the␣
 ↪<tr> tags of the html page associated with the
class named "team" '''

def hockey_data():
    for page_number in range(1, 25):
        all_data = scrape_page(page_number)
        save_data(page_number, all_data)

'''We have created another function which we will use to pass page_number␣
 ↪values to the function above and invoke it
and this function which has another function invocation of the save_data␣
 ↪function which is invoked next'''


def save_data(page_number, all_data):
        complt_data = []
        for index, data in enumerate(all_data):
            team_name = data.find('td', class_='name').text.strip()
            year = data.find('td', class_='year').text.strip()
            wins = data.find('td', class_='wins').text.strip()
            losses = data.find('td', class_='losses').text.strip()
            overtime_losses = data.find('td', class_='ot-loses').text.strip()␣
 ↪if data.find('td', class_='ot-loses') else ''
            win_percentage = data.find('td', class_='pct text-success').text.
 ↪strip() if data.find('td', class_='pct text-success') else ''
            goals_for = data.find('td', class_='gf').text.strip()
            goals_against = data.find('td', class_='ga').text.strip()
            diff_between_goals = data.find('td', class_='diff text-success').
 ↪text.strip() if data.find('td', class_='diff text-success') else ''


            complt_data.append({
            "Team Name": team_name,
            "Year": year,
            "Wins": wins,
            "Losses": losses,
            "Overtime Losses": overtime_losses,
            "Win Percentage": win_percentage,
            "Goals For": goals_for,
            "Goals Against": goals_against,
            "Difference Between Goals": diff_between_goals
        })

        df = pd.DataFrame(complt_data)
        df.to_csv(f'hockey data/hockeydata_page{page_number}.csv', index=False)
        print(f'Data saved for page {page_number} ')



'''Next we have created the actual logic for moving through each data point in␣
 ↪the pagenated web pages with a table of data.
we create the function save_data to which we pass the values of page number␣
 ↪through page_number and all_data which
has the returned value/'result' of the invoked function␣
 ↪scrape_page(page_number). It is through the scrape_page function we
find the tr and td tags associated with a particular page. For page 1 we have a␣
 ↪csv file named hockeydata_page1
and similarly for page 2 we have a csv file named hockeydata_page2. In the for␣
 ↪loop, we use the td tags and
their corresponding classes to extract the cell values and write them to the␣
 ↪csv files. We have also included a print statement
which tells us that a particular page's data is saved in a particular file'''

'''It is important to note that we have used if else to ignore the empty values␣
 ↪in the tables of the pages and to create
```

```
an empty string instead if and when we encounter them.'''



if __name__ == '__main__':
    hockey_data()
    time.sleep(1)
#we add a small delay of 2 second between each request
'''Here finally we invoke hockey_data function. The flow of the program is such␣
 ↪that by invoking hockey data function we
pass he page value to scrape_page function (from 1 to 24 which are the pages of␣
 ↪the pagenated webpage) whose returned
value/'result' is stored in the all_data object and then save_data function is␣
 ↪invoked with the values page_number and
all_data passed to it which is where actual extraction and writing of the data␣
 ↪into the csv files takes place.'''
```

```
Data saved for page 1
Data saved for page 2
Data saved for page 3
Data saved for page 4
Data saved for page 5
Data saved for page 6
Data saved for page 7
Data saved for page 8
Data saved for page 9
Data saved for page 10
Data saved for page 11
Data saved for page 12
Data saved for page 13
Data saved for page 14
Data saved for page 15
Data saved for page 16
Data saved for page 17
Data saved for page 18
Data saved for page 19
Data saved for page 20
Data saved for page 21
Data saved for page 22
Data saved for page 23
Data saved for page 24
```

[30]:

```
"Here finally we invoke hockey_data function. The flow of the program is such
that by invoking hockey data function we\npass he page value to scrape_page
function (from 1 to 24 which are the pages of the pagenated webpage) whose
returned \nvalue/'result' is stored in the all_data object and then save_data
function is invoked with the values page_number and \nall_data passed to it
```

```
which is where actual extraction and writing of the data into the text file
takes place."
```

[31]:

```
os.getcwd()
'''We use getcwd of os module to get an idea of where the current directory is␣
 ↪present locally in the PC
when the program is run through Jupyter Notebook.'''
```

[31]:

```
'We use getcwd of os module to get an idea of where the current directory is
present locally in the PC \nwhen the program is run through Jupyter Notebook.'
```

[36]:

```
import os



directory = 'hockey data'
output_file = 'combined_data.csv'

#Open output file in write mode
with open(output_file, 'w') as outfile:
    # Iterate through CSV files in the directory
    for filename in os.listdir(directory):
        if filename.endswith(".csv"):
            file_path = os.path.join(directory, filename)
            # Open each CSV file in read mode and append its contents to the␣
 ↪output file
            with open(file_path, 'r') as dat_file:
                outfile.write(dat_file.read())



print("combined CSV file created")

'''We open the output file in write mode since we do need to write all the csv␣
 ↪data into it.
We move through the csv files in the hockey data folder present in the local␣
 ↪working directory of mine and find all the files
which end with .csv extension and join those filenames with the directory name␣
 ↪to create a "path". We use this path to actually
write data to the output file which is the "combined_data.csv" file.'''
```

```
combined CSV file created
```

[36]:

```
'We open the output file in write mode since we do need to write all the csv
data into it. \nWe move through the csv files in the hockey data folder present
in the local working directory of mine and find all the files\nwhich end with
.csv extension and join those filenames with the directory name to create a
"path". We use this path to actually\nwrite data to the output file which is the
"combined_data.csv" file.'
```

[250]:

```
hock_dat = pd.read_csv(r'C:\Users\bvsro\combined_data.csv')
#using 'r' to read raw string literals since I get unicode error
print(hock_dat)
```

```
                   Team Name  Year Wins Losses Overtime Losses Win Percentage  \
0              Boston Bruins  1990   44     24             NaN           0.55
1             Buffalo Sabres  1990   31     30             NaN            NaN
2             Calgary Flames  1990   46     26             NaN          0.575
3         Chicago Blackhawks  1990   49     23             NaN          0.613
4           Detroit Red Wings  1990   34     38             NaN            NaN
..                       …   …  …    …             …            …
600       Tampa Bay Lightning  1998   19     54             NaN            NaN
601       Toronto Maple Leafs  1998   45     30             NaN          0.549
602         Vancouver Canucks  1998   23     47             NaN            NaN
603       Washington Capitals  1998   31     45             NaN            NaN
604     Mighty Ducks of Anaheim  1999   34     33             NaN            NaN

    Goals For Goals Against Difference Between Goals
0         299           264                       35
1         292           278                       14
2         344           263                       81
3         284           211                       73
4         273           298                      NaN
..        …          …                      …
600       179           292                      NaN
601       268           231                       37
602       192           258                      NaN
603       200           218                      NaN
604       217           227                      NaN

[605 rows x 9 columns]
```

[251]:

```
print(hock_dat.dtypes)
print(hock_dat.count())
```

```
Team Name                   object
Year                        object
Wins                        object
Losses                      object
Overtime Losses             object
Win Percentage              object
Goals For                   object
Goals Against               object
Difference Between Goals    object
dtype: object
Team Name                   605
Year                        605
Wins                        605
```

```
Losses                      605
Overtime Losses              23
Win Percentage              255
Goals For                   605
Goals Against               605
Difference Between Goals    342
dtype: int64
```

[252]:
```
hock_dat= hock_dat.drop(columns = ['Overtime Losses'])
hock_dat = hock_dat.drop(columns = ['Win Percentage'])

'''Dropping two of the above columns since there is a lot of missing data. But␣
 ↪it should be possible to draw win percentage ourselves from
the givne data.'''
```

[252]:
```
'Dropping two of the above columns since there is a lot of missing data. But it
should be possible to draw win percentage ourselves from\nthe givne data.'
```

[253]:
```
print(hock_dat['Losses'].unique())

'''checking the unique value of losses we see that it has a string "Losses" in␣
 ↪it which needs to be removed'''
```

```
['24' '30' '26' '23' '38' '37' '39' '33' '45' '31' '50' '22' '46' '43'
 '36' '32' '29' 'Losses' '57' '35' '28' '27' '34' '40' '48' '19' '47' '41'
 '16' '20' '51' '21' '25' '42' '17' '44' '58' '52' '18' '15' '70' '71'
 '54' '61' '11' '13' '59' '55']
```

[253]:
```
'checking the unique value of losses we see that it has a string "Losses" in it
which needs to be removed'
```

[254]:
```
hock_dat['Losses'] = hock_dat['Losses'].replace('Losses', float('nan'))
print(hock_dat['Losses'].unique())

"We replace Losses with float-NaN value so that we can drop the rows containing␣
 ↪it from not just this column but the entire data"
```

```
['24' '30' '26' '23' '38' '37' '39' '33' '45' '31' '50' '22' '46' '43'
 '36' '32' '29' nan '57' '35' '28' '27' '34' '40' '48' '19' '47' '41' '16'
 '20' '51' '21' '25' '42' '17' '44' '58' '52' '18' '15' '70' '71' '54'
 '61' '11' '13' '59' '55']
```

[254]:
```
'We replace Losses with float-NaN value so that we can drop the rows containing
it from not just this column but the entire data'
```

[255]:
```
hock_dat = hock_dat.dropna(axis = 0)
print(hock_dat['Losses'].unique())
print(hock_dat.count())
```

```
'''We use drop all the not float-nan values from the data'''
```

```
['24' '30' '26' '23' '37' '33' '31' '22' '36' '29' '32' '35' '28' '27'
 '34' '19' '16' '20' '21' '25' '17' '18' '15' '40' '11' '13']
Team Name                   319
Year                        319
Wins                        319
Losses                      319
Goals For                   319
Goals Against               319
Difference Between Goals    319
dtype: int64
```

[255]:

```
'We use drop all the not float-nan values from the data'
```

[256]:

```
hock_dat.count()
```

[256]:

```
Team Name                   319
Year                        319
Wins                        319
Losses                      319
Goals For                   319
Goals Against               319
Difference Between Goals    319
dtype: int64
```

[257]:

```
hock_dat['Year'] = hock_dat['Year'].astype(int)
print(hock_dat['Year'].dtype)
```

```
int32
```

[258]:

```
print(hock_dat.dtypes)
```

```
Team Name                   object
Year                         int32
Wins                        object
Losses                      object
Goals For                   object
Goals Against               object
Difference Between Goals    object
dtype: object
```

[259]:

```
hock_dat['Wins'] = hock_dat['Wins'].astype(int)
hock_dat['Losses'] = hock_dat['Losses'].astype(int)
hock_dat['Goals For'] = hock_dat['Goals For'].astype(int)
hock_dat['Goals Against'] = hock_dat['Goals Against'].astype(int)
hock_dat['Difference Between Goals'] = hock_dat['Difference Between Goals'].
  ↪astype(int)
```

```
#convering into appropriate data types
```

[272]:

```
hock_dat['Win Percentage New'] = ((hock_dat['Wins'])/(hock_dat['Wins'] +␣
↪hock_dat['Losses']))*100
#creating our own win percentage from the data available at this point since we␣
↪dropped the entire Win percentage column earlier
print(hock_dat['Win Percentage New'])
hock_dat['Win Percentage New'] = hock_dat['Win Percentage New'].round()
print(hock_dat['Win Percentage New'])
#Rounding the values to their nearest integer
```

```
0      64.705882
1      50.819672
2      63.888889
3      68.055556
5      50.000000
         …
596    55.714286
597    55.882353
598    48.437500
599    53.623188
601    60.000000
Name: Win Percentage New, Length: 319, dtype: float64
0      65.0
1      51.0
2      64.0
3      68.0
5      50.0
       …
596    56.0
597    56.0
598    48.0
599    54.0
601    60.0
Name: Win Percentage New, Length: 319, dtype: float64
```

[261]:

```
import matplotlib.pyplot as pt
import seaborn as sns
import colorcet as cc
```

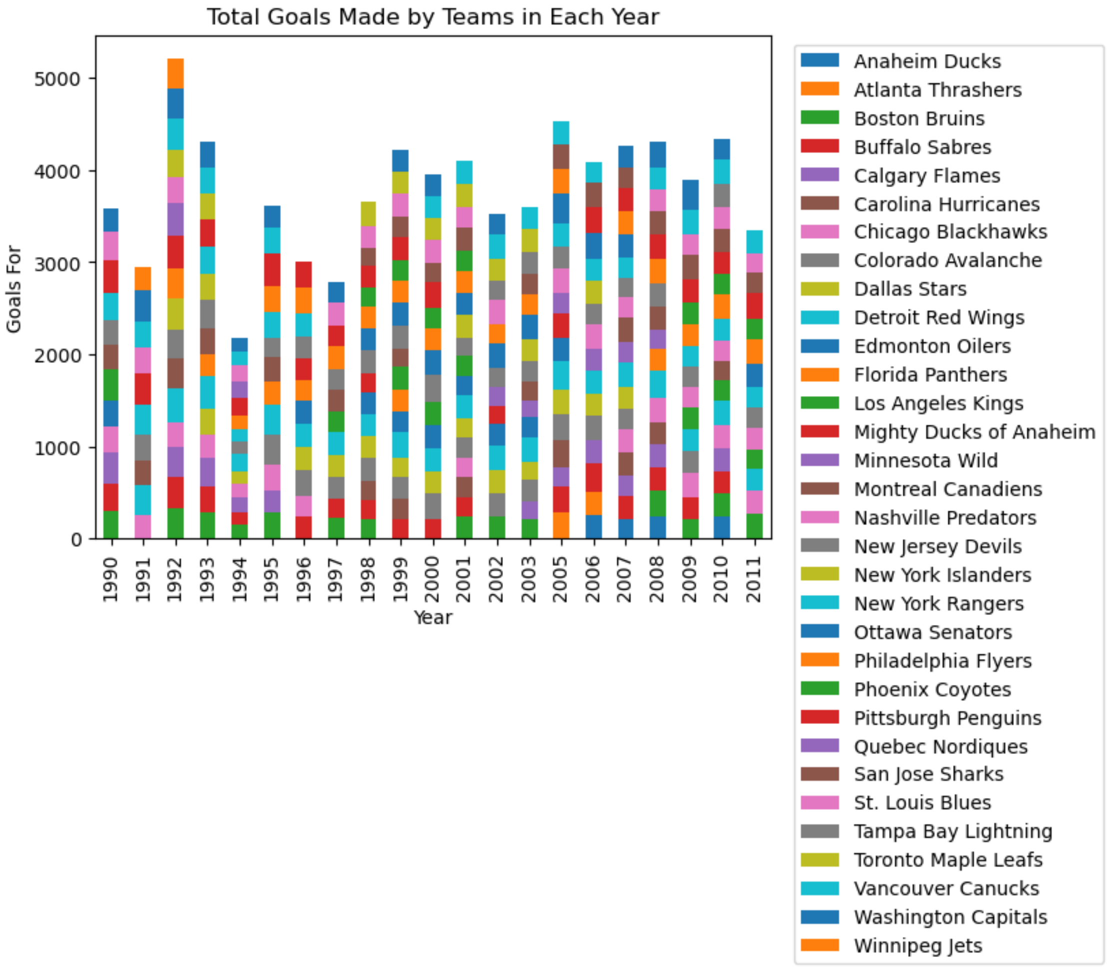
[262]:

```
goals_by_team_year = hock_dat.pivot_table(index='Year', columns='Team Name',␣
↪values='Goals For', aggfunc='sum')
```

[263]:

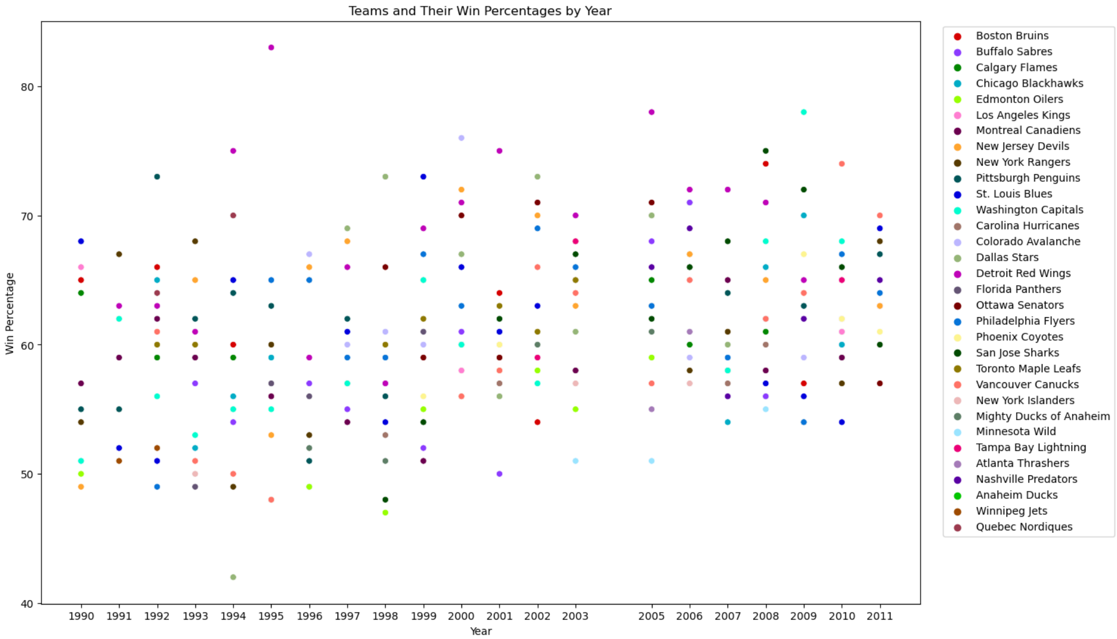
```
pt.figure(figsize=(12,8))
goals_by_team_year.plot(kind='bar', stacked=True)
pt.title('Total Goals Made by Teams in Each Year')
pt.xlabel('Year')
pt.ylabel('Goals For')
pt.legend(loc = 'upper left', bbox_to_anchor=(1.02, 1))
pt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



```
[337]: palette_col = sns.color_palette(cc.glasbey, n_colors=32)
       pt.figure(figsize=(15,10))
       scatter_1 = sns.scatterplot(data = hock_dat, x = 'Year', y = 'Win Percentage␣
         ↪New', hue = 'Team Name', palette = palette_col)
       scatter_1.set_xticks(hock_dat['Year'].unique())
       pt.title('Teams and Their Win Percentages by Year')
       pt.xlabel('Year')
       pt.ylabel('Win Percentage')
       pt.legend(loc = 'upper left', bbox_to_anchor=(1.02, 1))
       pt.show()
```

```
'''In the chart it would seem that Detriot Red Wings, represented by pink color, seem to have relatively higher
win percentages over the years. The lowest win percentage is Mighty Ducks of Anaheim while the highest win percentage
recorded in the data available is by Detroit Red Wings. It is possible that there is some other team with higher win percentage
since some of the rows were deleted/dropped from the original scraped data.'''
```



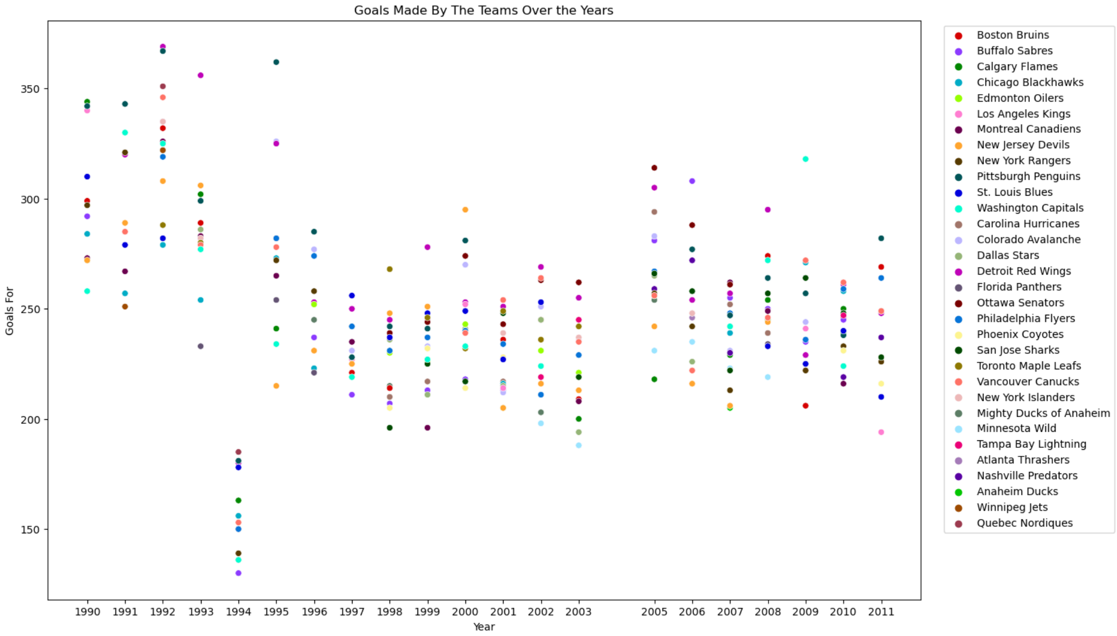
[337]:

```
'In the chart it would seem that Detriot Red Wings, represented by pink color,
seem to have relatively higher \nwin percentages over the years. The lowest win
percentage is Mighty Ducks of Anaheim while the highest win percentage\nrecorded
in the data available is by Detroit Red Wings. It is possible that there is some
other team with higher win percentage\nsince some of the rows were
deleted/dropped from the original scraped data.'
```

[339]:

```
palette_col = sns.color_palette(cc.glasbey, n_colors=32)
pt.figure(figsize=(15,10))
scatter_1 = sns.scatterplot(data = hock_dat, x = 'Year', y = 'Goals For', hue = 'Team Name', palette = palette_col)
scatter_1.set_xticks(hock_dat['Year'].unique())
pt.title('Goals Made By The Teams Over the Years')
pt.xlabel('Year')
pt.ylabel('Goals For')
pt.legend(loc = 'upper left', bbox_to_anchor=(1.02, 1))
pt.show()
```

```python
'''It would seem that there is some correlation between Goals For and Win Percentage since even in terms of goals
scored by teams Detriot Red Wings seems to be doing relatively better than other teams.'''
```



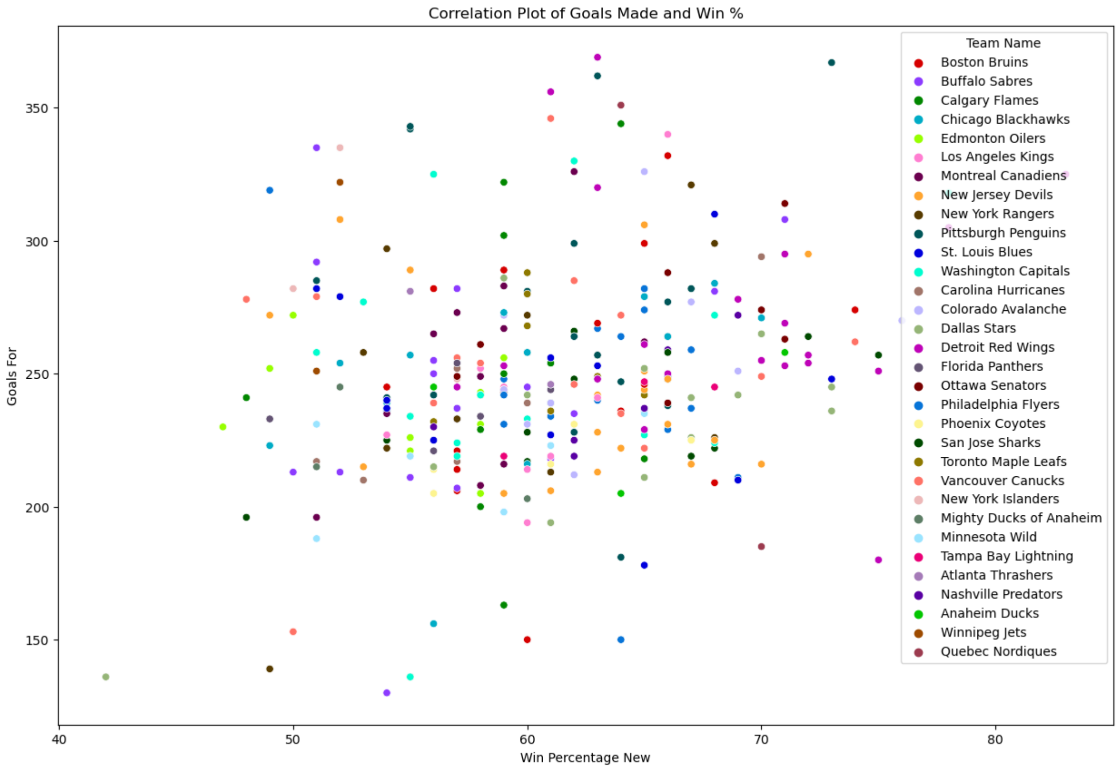
[339]:

```
'It would seem that there is some correlation between Goals For and Win
Percentage since even in terms of goals \nscored by teams Detriot Red Wings
seems to be doing relatively better than other teams.'
```

[341]:

```python
pt.figure(figsize=(15, 10))
sns.scatterplot(data = hock_dat, x = 'Win Percentage New', y = 'Goals For', hue = 'Team Name', palette = palette_col)
pt.title('Correlation Plot of Goals Made and Win %')
pt.show()

'''Checking if there is any correlation between Win % and Goals scored. On first glance, it seems like there might be some correlation
so it may be worth investigating.'''
```



```
[341]: 'Checking if there is any correlation between Win % and Goals scored. On first
       glance, it seems like there might be some correlation\nso it may be worth
       investigating.'
```

```
[342]: hock_dat_newframe = pd.DataFrame({'Win Percentage New' : hock_dat['Win Percentage New'], 'Goals For' : hock_dat['Goals For']})
       print(hock_dat_newframe)
       print(hock_dat['Win Percentage New'])

       #creating a new dataframe for later use
```

```
     Win Percentage New  Goals For
0                  65.0        299
1                  51.0        292
2                  64.0        344
3                  68.0        284
5                  50.0        272
..                  …          …
596                56.0        205
597                56.0        242
598                48.0        196
599                54.0        237
601                60.0        268
```

```
[319 rows x 2 columns]
0      65.0
1      51.0
2      64.0
3      68.0
5      50.0
       …
596    56.0
597    56.0
598    48.0
599    54.0
601    60.0
Name: Win Percentage New, Length: 319, dtype: float64
```

[343]:

```
print(hock_dat_newframe.corr(method = 'pearson'))
print(hock_dat_newframe.corr(method = 'kendall'))
print(hock_dat_newframe.corr(method = 'spearman'))

'''While there does to be some correlation, it doesn't seem to be strong. So,␣
 ↪higher goals do not mean high win % and vice versa'''
```

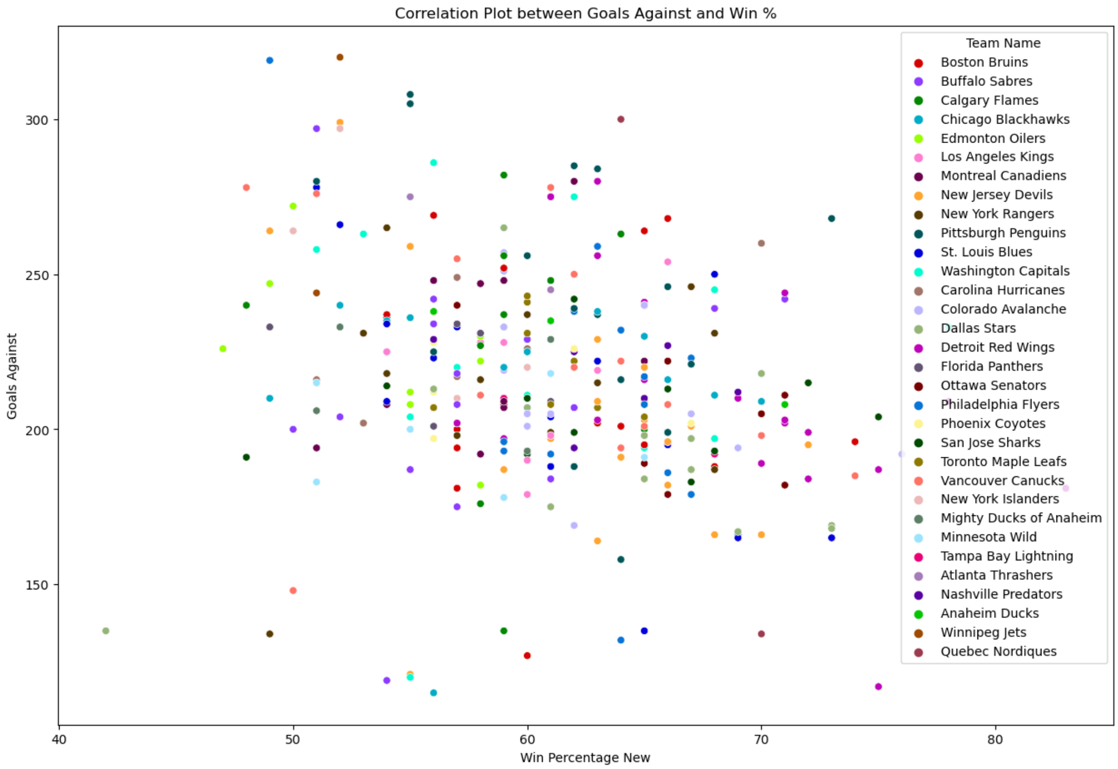
```
                    Win Percentage New  Goals For
Win Percentage New            1.000000   0.210828
Goals For                     0.210828   1.000000
                    Win Percentage New  Goals For
Win Percentage New            1.000000   0.141691
Goals For                     0.141691   1.000000
                    Win Percentage New  Goals For
Win Percentage New             1.00000    0.19952
Goals For                      0.19952    1.00000
```

[343]:

```
"While there does to be some correlation, it doesn't seem to be strong. So,
higher goals do not mean high win % and vice versa"
```

[344]:

```
pt.figure(figsize=(15, 10))
sns.scatterplot(data = hock_dat, x = 'Win Percentage New', y = 'Goals Against',␣
 ↪hue = 'Team Name', palette = palette_col)
pt.title('Correlation Plot between Goals Against and Win %')
pt.show()
```



[345]:
```
hock_dat_newframe_2 = pd.DataFrame({'Win Percentage New 2' : hock_dat['Win␣
 ↪Percentage New'], 'Goals Against' : hock_dat['Goals Against']})
print(hock_dat_newframe_2)

#creating a new dataframe for later use
```

```
     Win Percentage New 2  Goals Against
0                    65.0            264
1                    51.0            278
2                    64.0            263
3                    68.0            211
5                    50.0            272
..                    …              …
596                  56.0            197
597                  56.0            225
598                  48.0            191
599                  54.0            209
601                  60.0            231

[319 rows x 2 columns]
```

[346]:

```
print(hock_dat_newframe_2.corr(method = 'pearson'))
print(hock_dat_newframe_2.corr(method = 'kendall'))
print(hock_dat_newframe_2.corr(method = 'spearman'))

'''Evidently the correlation between Win % and Goals Against is weak.'''
```

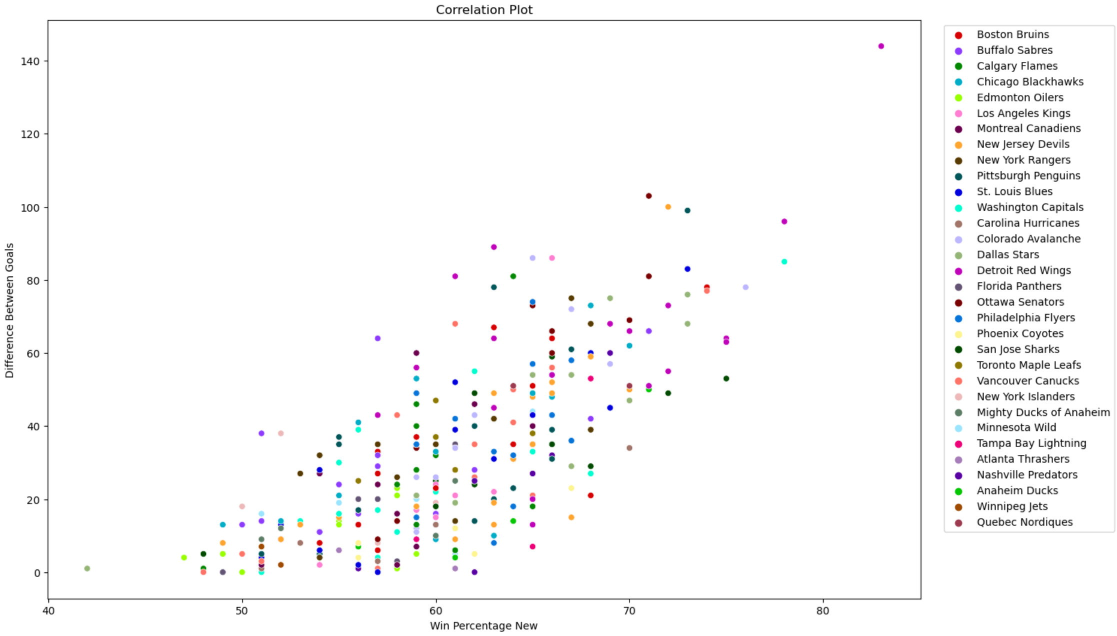
```
                    Win Percentage New 2  Goals Against
Win Percentage New 2             1.000000      -0.270776
Goals Against                   -0.270776       1.000000
                    Win Percentage New 2  Goals Against
Win Percentage New 2             1.000000      -0.219195
Goals Against                   -0.219195       1.000000
                    Win Percentage New 2  Goals Against
Win Percentage New 2             1.000000      -0.305936
Goals Against                   -0.305936       1.000000
```

[346]:

```
'Evidently the correlation between Win % and Goals Against is weak.'
```

[347]:

```
pt.figure(figsize=(15, 10))
sns.scatterplot(data = hock_dat, x = 'Win Percentage New', y = 'Difference␣
 ↪Between Goals', hue = 'Team Name', palette = palette_col)
pt.legend(loc = 'upper left', bbox_to_anchor=(1.02, 1))
pt.title('Correlation Plot')
pt.show()
```



[348]:

```
hock_dat_newframe_3 = pd.DataFrame({'Win Percentage New 3' : hock_dat['Win␣
 ↪Percentage New'], 'Difference Between Goals' : hock_dat['Difference Between␣
 ↪Goals']})
print(hock_dat_newframe_3)
```

```
     Win Percentage New 3  Difference Between Goals
0                    65.0                        35
1                    51.0                        14
2                    64.0                        81
3                    68.0                        73
5                    50.0                         0
..                    …                        …
596                  56.0                         8
597                  56.0                        17
598                  48.0                         5
599                  54.0                        28
601                  60.0                        37

[319 rows x 2 columns]
```

[349]:

```
print(hock_dat_newframe_3.corr(method = 'pearson'))
print(hock_dat_newframe_3.corr(method = 'kendall'))
print(hock_dat_newframe_3.corr(method = 'spearman'))

'''There is a high correlation between difference between goals scored and win␣
 ↪percentage. Could be the case that the
more there are players in a team who can score, the greater the chances are␣
 ↪there for a team to win which is also something
that can be thought of just by common sense but this may confirm it.'''
```

```
                          Win Percentage New 3  Difference Between Goals
Win Percentage New 3                  1.000000                  0.738272
Difference Between Goals              0.738272                  1.000000
                          Win Percentage New 3  Difference Between Goals
Win Percentage New 3                  1.000000                  0.545415
Difference Between Goals              0.545415                  1.000000
                          Win Percentage New 3  Difference Between Goals
Win Percentage New 3                   1.00000                   0.72678
Difference Between Goals               0.72678                   1.00000
```