```
In [1]: import requests
        import pandas as pd
        import numpy as np
        from datetime import datetime
        import re
        import warnings
        #warnings.simplefilter('ignore')

        #Specifying which teams and years to include in data set
        Teams = ['CHC', 'CHW', 'LAA', 'LAD', 'NYM', 'NYY', 'OAK', 'SFG']
        Years = []
        for year in range(2010,2020):
            Years.append(str(year))

        list_of_df = list()

        #Creating dataframe
        for team in Teams:
            for year in Years:
                #Will retrieve data from each year for each team
                url = 'https://www.baseball-reference.com/teams/' + team + '/' +
        year +'-schedule-scores.shtml'
                dfname = team + '_' + year
                html = requests.get(url).content
                df_list = pd.read_html(html)
                df = df_list[-1]

                #Formatting data table
                #rename columns
                df.rename(columns={"Gm#": "GM_Num", "Unnamed: 4": "Home", "Tm":
        "Team", "D/N": "Night"}, inplace = True)
                #turn home, game win, and night into dummy variables
                df['Home'] = df['Home'].apply(lambda x: 0 if x == '@' else 1)
                df['Game_Win'] = df['W/L'].astype(str).str[0]
                df['Game_Win'] = df['Game_Win'].apply(lambda x: 0 if x == 'L' el
        se 1)
                df['Night'] = df['Night'].apply(lambda x: 1 if x == 'N' else 0)
                #quantify streak as number
                df['Streak'] = df['Streak'].apply(lambda x: -1*len(x) if '-' in
        x else len(x))
                df.drop('Unnamed: 2', axis=1, inplace = True)
                df.drop('Orig. Scheduled', axis=1, inplace = True)
                df.drop('Win', axis=1, inplace = True)
                df.drop('Loss', axis=1, inplace = True)
                df.drop('Save', axis=1, inplace = True)
                #Drop rows that do not have data
                df = df[df['GM_Num'].str.isdigit()]
                #Convert W-L column to 4 new numeric columns: Wins, Losses, Net
         Wins (Wins - Losses), Win Percentage (Wins/Total Games)
                WL = df["W-L"].str.split("-", n = 1, expand = True)
                df["Wins"] = WL[0].astype(dtype=np.int64)
                df["Losses"] = WL[1].astype(dtype=np.int64)
                df['Net_Wins'] = df['Wins'] - df['Losses']
                df['Win_Per'] = df['Wins']/(df['Wins']+df['Losses'])
                #Turn date into datetime object
                DayDate = df['Date'].str.split(", ", n = 1, expand = True)
        df['DayOfWeek'] = DayDate[0]
        df['Date'] = DayDate[1] + ', ' + year
        df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
]]
        df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
        #Add to list which will be turned into a dataframe
        list_of_df.append(df)

#Create dataframe
bbattend = pd.concat(list_of_df)
#bbattend
```

```
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:46: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:47: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:48: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:49: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:52: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:53: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:54: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:55: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
```

In [2]:
```
#Create Year Variable for Matching Later
bbattend['Year'] = bbattend.Date.dt.year

#Specify what the same-market team is for matching
bbattend['Same_Mkt_Team'] = bbattend.apply(lambda _: '', axis=1)
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('LAA')] = 'LAD'
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('LAD')] = 'LAA'
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('NYY')] = 'NYM'
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('NYM')] = 'NYY'
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('CHW')] = 'CHC'
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('CHC')] = 'CHW'
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('OAK')] = 'SFG'
bbattend['Same_Mkt_Team'][bbattend['Team'].str.contains('SFG')] = 'OAK'

#Create day of week dummy variables
WeekDays = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Sat
urday', 'Sunday']
for i in WeekDays:
    bbattend[i] = bbattend.apply(lambda _: '', axis=1)
    bbattend[i] = bbattend['DayOfWeek'].apply(lambda x: 1 if x == i else
0)

#Create game_id which will be used to delete duplicates later
bbattend['game_id'] = bbattend['Team'] + bbattend['Date'].astype(str)
#bbattend
```

```
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy

/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  import sys
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy

/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  # Remove the CWD from sys.path while we load stuff.
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  # This is added back by InteractiveShellApp.init_path()
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  if sys.path[0] == '':
/Users/laurel/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.
py:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  del sys.path[0]
```

In [3]:
```
# Create merged table
# Will match all dates of games of team with dates within same year of t
eams from same-market team
merged = bbattend.merge(
    bbattend[["Date", "Year", "Team", "Net_Wins", "Win_Per","Streak", 'W
ins', 'Losses']],
    how="inner",
    left_on=["Year", "Same_Mkt_Team"],
    right_on=["Year", "Team"],
    suffixes=('', '_Same_Mkt_Team')
)

#Measure how far apart the dates of the games are
merged["date_diff"] = (merged.Date - merged.Date_Same_Mkt_Team).dt.days
#Only keep the dates of same-market team that occurred before the date o
f home team's game
merged = merged[merged['date_diff'] > 0]
```

```
In [4]: #Sort by date_diff so closest dates appear first
        merged.sort_values(by='date_diff', inplace = True)

        #Only keep first game_id which will include the data of the same-market
         team for the closest date before the game
        merged.drop_duplicates(subset =['game_id'], keep = 'first', inplace = Tr
        ue)


        merged.sort_values(by=['Team', 'Date'], inplace = True)
        merged.head(20)
```

Out[4]:

	GM_Num	Date	Team	Home	Opp	W/L	R	RA	Inn	W-L	...	Sunday	game_id	Dat
162	2	2010-04-07	CHC	0	ATL	L	2	3	NaN	0-2	...	0	CHC2010-04-07	
325	3	2010-04-08	CHC	0	ATL	W	2	0	NaN	1-2	...	0	CHC2010-04-08	
488	4	2010-04-09	CHC	0	CIN	L	4	5	NaN	1-3	...	0	CHC2010-04-09	
651	5	2010-04-10	CHC	0	CIN	W	4	3	NaN	2-3	...	0	CHC2010-04-10	
814	6	2010-04-11	CHC	0	CIN	L	1	3	NaN	2-4	...	1	CHC2010-04-11	
977	7	2010-04-12	CHC	1	MIL	W	9	5	NaN	3-4	...	0	CHC2010-04-12	
1141	8	2010-04-14	CHC	1	MIL	W	7	6	NaN	4-4	...	0	CHC2010-04-14	
1304	9	2010-04-15	CHC	1	MIL	L	6	8	NaN	4-5	...	0	CHC2010-04-15	
1467	10	2010-04-16	CHC	1	HOU	W	7	2	NaN	5-5	...	0	CHC2010-04-16	
1630	11	2010-04-17	CHC	1	HOU	L	3	4	NaN	5-6	...	0	CHC2010-04-17	
1793	12	2010-04-18	CHC	1	HOU	L	2	3	10	5-7	...	1	CHC2010-04-18	
1956	13	2010-04-19	CHC	0	NYM	L	1	6	NaN	5-8	...	0	CHC2010-04-19	
2118	14	2010-04-20	CHC	0	NYM	L	0	4	NaN	5-9	...	0	CHC2010-04-20	
2281	15	2010-04-21	CHC	0	NYM	W	9	3	NaN	6-9	...	0	CHC2010-04-21	
2444	16	2010-04-22	CHC	0	NYM	L	2	5	NaN	6-10	...	0	CHC2010-04-22	
2607	17	2010-04-23	CHC	0	MIL	W	8	1	NaN	7-10	...	0	CHC2010-04-23	
2770	18	2010-04-24	CHC	0	MIL	W	5	1	NaN	8-10	...	0	CHC2010-04-24	
2933	19	2010-04-25	CHC	0	MIL	W	12	2	NaN	9-10	...	1	CHC2010-04-25	
3096	20	2010-04-26	CHC	1	WSN	W-wo	4	3	10	10-10	...	0	CHC2010-04-26	
3258	21	2010-04-27	CHC	1	WSN	L	1	3	NaN	10-11	...	0	CHC2010-04-27	

20 rows × 40 columns

In [5]:

```
#merged.to_csv('bbattend.csv')
```

```
In [6]: #create df with just home games
        homegames = merged[merged.Home == 1]
        #homegames.to_csv('bbattendhome.csv')
        homegames
```

Out[6]:

	GM_Num	Date	Team	Home	Opp	W/L	R	RA	Inn	W-L	...	Sunday	game_id
977	7	2010-04-12	CHC	1	MIL	W	9	5	NaN	3-4	...	0	CHC2010-04-12
1141	8	2010-04-14	CHC	1	MIL	W	7	6	NaN	4-4	...	0	CHC2010-04-14
1304	9	2010-04-15	CHC	1	MIL	L	6	8	NaN	4-5	...	0	CHC2010-04-15
1467	10	2010-04-16	CHC	1	HOU	W	7	2	NaN	5-5	...	0	CHC2010-04-16
1630	11	2010-04-17	CHC	1	HOU	L	3	4	NaN	5-6	...	0	CHC2010-04-17
1793	12	2010-04-18	CHC	1	HOU	L	2	3	10	5-7	...	1	CHC2010-04-18
3096	20	2010-04-26	CHC	1	WSN	W-wo	4	3	10	10-10	...	0	CHC2010-04-26
3258	21	2010-04-27	CHC	1	WSN	L	1	3	NaN	10-11	...	0	CHC2010-04-27
3421	22	2010-04-28	CHC	1	WSN	L	2	3	NaN	10-12	...	0	CHC2010-04-28
3584	23	2010-04-29	CHC	1	ARI	L	5	13	NaN	10-13	...	0	CHC2010-04-29
3747	24	2010-04-30	CHC	1	ARI	W	11	5	NaN	11-13	...	0	CHC2010-04-30
3910	25	2010-05-01	CHC	1	ARI	W	7	5	NaN	12-13	...	0	CHC2010-05-01
4073	26	2010-05-02	CHC	1	ARI	W	10	5	NaN	13-13	...	1	CHC2010-05-02
5215	33	2010-05-10	CHC	1	FLA	L	2	4	NaN	14-19	...	0	CHC2010-05-10
5377	34	2010-05-11	CHC	1	FLA	L	2	3	NaN	14-20	...	0	CHC2010-05-11
5540	35	2010-05-12	CHC	1	FLA	W	4	3	NaN	15-20	...	0	CHC2010-05-12
5703	36	2010-05-14	CHC	1	PIT	L	6	10	NaN	15-21	...	0	CHC2010-05-14
5866	37	2010-05-15	CHC	1	PIT	L	3	4	NaN	15-22	...	0	CHC2010-05-15
6029	38	2010-05-16	CHC	1	PIT	W	4	3	NaN	16-22	...	1	CHC2010-05-16
6192	39	2010-05-17	CHC	1	COL	W-wo	4	2	11	17-22	...	0	CHC2010-05-17
6354	40	2010-05-18	CHC	1	COL	W	6	2	NaN	18-22	...	0	CHC2010-05-18
7333	46	2010-05-25	CHC	1	LAD	W	3	0	NaN	22-24	...	0	CHC2010-05-25
7496	47	2010-05-26	CHC	1	LAD	L	5	8	NaN	22-25	...	0	CHC2010-05-26

	GM_Num	Date	Team	Home	Opp	W/L	R	RA	Inn	W-L	...	Sunday	game_id
7659	48	2010-05-27	CHC	1	LAD	W	1	0	NaN	23-25	...	0	CHC2010-05-27
7822	49	2010-05-28	CHC	1	STL	L	1	7	NaN	23-26	...	0	CHC2010-05-28
7985	50	2010-05-29	CHC	1	STL	W	5	0	NaN	24-26	...	0	CHC2010-05-29
8148	51	2010-05-30	CHC	1	STL	L	1	9	NaN	24-27	...	1	CHC2010-05-30
9778	61	2010-06-11	CHC	1	CHW	L	5	10	NaN	27-34	...	0	CHC2010-06-11
9941	62	2010-06-12	CHC	1	CHW	L	1	2	NaN	27-35	...	0	CHC2010-06-12
10104	63	2010-06-13	CHC	1	CHW	W	1	0	NaN	28-35	...	1	CHC2010-06-13
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2089739	102	2019-07-23	SFG	1	CHC	W-wo	5	4	13	52-50	...	0	SFG2019-07-23
2089902	103	2019-07-24	SFG	1	CHC	L	1	4	NaN	52-51	...	0	SFG2019-07-24
2091532	113	2019-08-05	SFG	1	WSN	L	0	4	NaN	56-57	...	0	SFG2019-08-05
2091695	114	2019-08-06	SFG	1	WSN	L	3	5	NaN	56-58	...	0	SFG2019-08-06
2091858	115	2019-08-07	SFG	1	WSN	L	1	4	NaN	56-59	...	0	SFG2019-08-07
2092021	116	2019-08-08	SFG	1	PHI	W	5	0	NaN	57-59	...	0	SFG2019-08-08
2092183	117	2019-08-09	SFG	1	PHI	L	6	9	NaN	57-60	...	0	SFG2019-08-09
2092346	118	2019-08-10	SFG	1	PHI	W	3	1	NaN	58-60	...	0	SFG2019-08-10
2092509	119	2019-08-11	SFG	1	PHI	W	9	6	NaN	59-60	...	1	SFG2019-08-11
2092672	120	2019-08-13	SFG	1	OAK	W	3	2	NaN	60-60	...	0	SFG2019-08-13
2092835	121	2019-08-14	SFG	1	OAK	L	5	9	NaN	60-61	...	0	SFG2019-08-14
2094465	131	2019-08-26	SFG	1	ARI	L	4	6	NaN	65-66	...	0	SFG2019-08-26
2094628	132	2019-08-27	SFG	1	ARI	L	2	3	NaN	65-67	...	0	SFG2019-08-27
2094792	133	2019-08-29	SFG	1	SDP	L	3	5	NaN	65-68	...	0	SFG2019-08-29
2094955	134	2019-08-30	SFG	1	SDP	W	8	3	NaN	66-68	...	0	SFG2019-08-30

	GM_Num	Date	Team	Home	Opp	W/L	R	RA	Inn	W-L	...	Sunday	game_id
2095118	135	2019-08-31	SFG	1	SDP	L	1	4	NaN	66-69	...	0	SFG2019-08-31
2095281	136	2019-09-01	SFG	1	SDP	L	4	8	NaN	66-70	...	1	SFG2019-09-01
2096584	144	2019-09-09	SFG	1	PIT	L	4	6	NaN	69-75	...	0	SFG2019-09-09
2096747	145	2019-09-10	SFG	1	PIT	W	5	4	NaN	70-75	...	0	SFG2019-09-10
2096910	146	2019-09-11	SFG	1	PIT	L	3	6	NaN	70-76	...	0	SFG2019-09-11
2097073	147	2019-09-12	SFG	1	PIT	L	2	4	NaN	70-77	...	0	SFG2019-09-12
2097236	148	2019-09-13	SFG	1	MIA	W	1	0	NaN	71-77	...	0	SFG2019-09-13
2097399	149	2019-09-14	SFG	1	MIA	L	2	4	NaN	71-78	...	0	SFG2019-09-14
2097562	150	2019-09-15	SFG	1	MIA	W	2	1	NaN	72-78	...	1	SFG2019-09-15
2098703	157	2019-09-24	SFG	1	COL	L	5	8	16	75-82	...	0	SFG2019-09-24
2098866	158	2019-09-25	SFG	1	COL	W-wo	2	1	NaN	76-82	...	0	SFG2019-09-25
2099029	159	2019-09-26	SFG	1	COL	W	8	3	NaN	77-82	...	0	SFG2019-09-26
2099192	160	2019-09-27	SFG	1	LAD	L	2	9	NaN	77-83	...	0	SFG2019-09-27
2099355	161	2019-09-28	SFG	1	LAD	L	0	2	NaN	77-84	...	0	SFG2019-09-28
2099518	162	2019-09-29	SFG	1	LAD	L	0	9	NaN	77-85	...	1	SFG2019-09-29

6384 rows × 40 columns

In [7]:
```python
def mean_std(cat):
    print(cat + ':')
    for i in Teams:
        TeamTemp = homegames[homegames['Team'] == i]
        TeamTemp = TeamTemp[pd.notnull(TeamTemp[cat])]
        TeamTemp[cat] = TeamTemp[cat].astype(str).astype(int)
        TeamMean = TeamTemp[cat].mean()
        TeamStd = TeamTemp[cat].std()
        print(i + "'s Mean: " + str(TeamMean))
        print(i + "'s Standard Dev: " + str(TeamStd))
```

In [8]:
```
mean_std('Attendance')
#mean_std('Win_Per')
```

```
Attendance:
CHC's Mean: 36891.974968710885
CHC's Standard Dev: 4080.876292558024
CHW's Mean: 22465.715012722645
CHW's Standard Dev: 6536.423707134375
LAA's Mean: 37858.708798017346
LAA's Standard Dev: 4100.180384667041
LAD's Mean: 44832.69937888199
LAD's Standard Dev: 6405.553003267846
NYM's Mean: 29966.022813688214
NYM's Standard Dev: 6223.064231626648
NYY's Mean: 42032.096815286626
NYY's Standard Dev: 4881.165233741245
OAK's Mean: 20075.015132408575
OAK's Standard Dev: 7939.438976393215
SFG's Mean: 40054.90099009901
SFG's Standard Dev: 3294.46544414653
```

In [9]:
```
segment_dummies = pd.get_dummies(homegames['Team'])
homegames = pd.concat([homegames, segment_dummies], axis=1)
homegames.head()
```

Out[9]:

	GM_Num	Date	Team	Home	Opp	W/L	R	RA	Inn	W-L	...	Losses_Same_Mkt_Team
977	7	2010-04-12	CHC	1	MIL	W	9	5	NaN	3-4	...	4
1141	8	2010-04-14	CHC	1	MIL	W	7	6	NaN	4-4	...	5
1304	9	2010-04-15	CHC	1	MIL	L	6	8	NaN	4-5	...	5
1467	10	2010-04-16	CHC	1	HOU	W	7	2	NaN	5-5	...	6
1630	11	2010-04-17	CHC	1	HOU	L	3	4	NaN	5-6	...	7

5 rows × 48 columns

In [ ]:

In [10]:
```
homegames.to_csv('homegames.csv')
```

In [ ]:

```
In [1]: import requests
        import pandas as pd
        import numpy as np
        from datetime import datetime
        import re
        import warnings
        warnings.simplefilter('ignore')

        #Specifying which teams and years to include in data set
        Teams = ['OAK']
        #Teams = ['SFG']
        #Teams = ['SFG', 'OAK', 'LAA', 'LAD', 'CHC', 'CHW']
        Years = ['2019']
        #for year in range(2016,2020):
        #    Years.append(str(year))

        list_of_df = list()

        #Creating dataframe
        for team in Teams:
            for year in Years:
                #Will retrieve data from each year for each team
                url = 'https://www.baseball-reference.com/teams/' + team + '/' +
        year +'-schedule-scores.shtml'
                dfname = team + '_' + year
                html = requests.get(url).content
                df_list = pd.read_html(html)
                df = df_list[-1]

                #Formatting data table
                #rename columns
                df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
                df = df[df['GM_Num'].str.isdigit()]
                df = df[['Date','Attendance']]
                #Turn date into datetime object
                DayDate = df['Date'].str.split(", ", n = 1, expand = True)
                df['Date'] = DayDate[1] + ', ' + year
                df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
        ]]
                df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
                #Add to list which will be turned into a dataframe
                list_of_df.append(df)

        #Create dataframe
        bbattend = pd.concat(list_of_df)
```

In [2]:
```
bbattend.dtypes
bbattend.head()
```

Out[2]:

	Date	Attendance
0	2019-03-20	45787
1	2019-03-21	46451
2	2019-03-28	22691
3	2019-03-29	22585
4	2019-03-30	16051

In [3]:
```
#print(bbattend.Attendance)
```

In [4]:
```
OAKdf = bbattend
OAKdf = OAKdf[pd.notnull(OAKdf['Attendance'])]
OAKdf['Attendance'] = OAKdf['Attendance'].astype(int)
OAKdf.dtypes
```

Out[4]:
```
Date          datetime64[ns]
Attendance             int64
dtype: object
```

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

Out[5]:

	Date	Attendance
0	2019-03-20	45787
1	2019-03-21	46451
2	2019-03-28	22691
3	2019-03-29	22585
4	2019-03-30	16051

In [6]:
```
Teams = ['SFG']
Years = ['2019']


list_of_df = list()

#Creating dataframe
for team in Teams:
    for year in Years:
        #Will retrieve data from each year for each team
        url = 'https://www.baseball-reference.com/teams/' + team + '/' +
year +'-schedule-scores.shtml'
        dfname = team + '_' + year
        html = requests.get(url).content
        df_list = pd.read_html(html)
        df = df_list[-1]

        #Formatting data table
        #rename columns
        df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
        df = df[df['GM_Num'].str.isdigit()]
        df = df[['Date','Attendance']]
        #Turn date into datetime object
        DayDate = df['Date'].str.split(", ", n = 1, expand = True)
        df['Date'] = DayDate[1] + ', ' + year
        df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
]]
        df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
        #Add to list which will be turned into a dataframe
        list_of_df.append(df)

#Create dataframe
bbattend = pd.concat(list_of_df)
```

In [7]:
```
SFGdf = bbattend
SFGdf = SFGdf[pd.notnull(SFGdf['Attendance'])]
#OAKdf['Attendance'] = OAKdf['Attendance'].astype(str).astype(int)
SFGdf['Attendance'] = SFGdf['Attendance'].astype(int)
SFGdf.dtypes
```

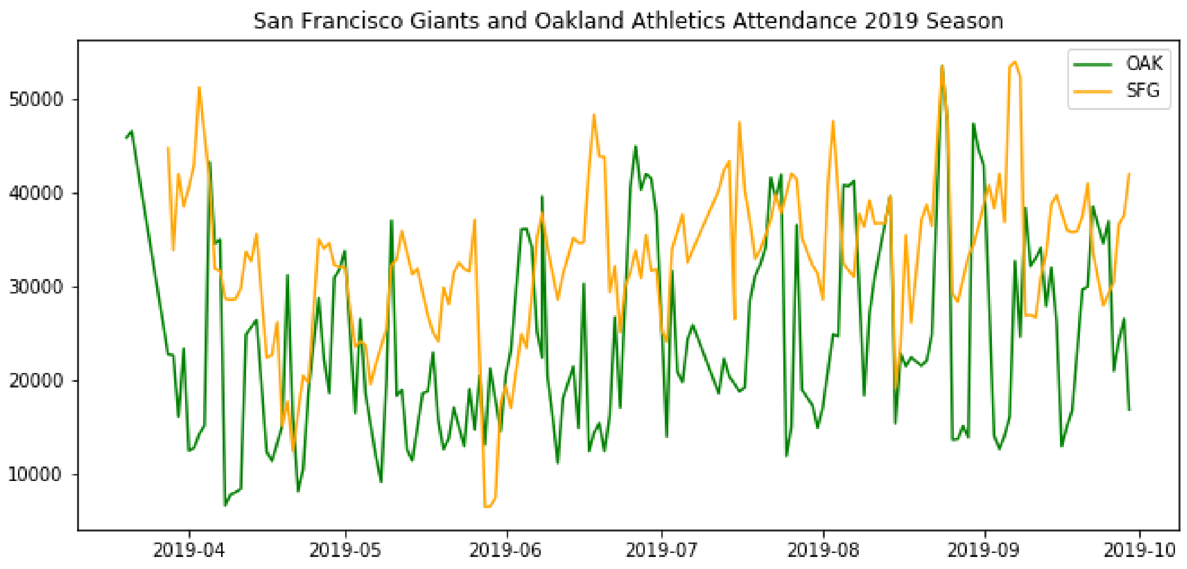
Out[7]:
```
Date          datetime64[ns]
Attendance             int64
dtype: object
```

In [8]:
```
SFGdf = SFGdf.dropna()
SFGdf.head()
```

Out[8]:

	Date	Attendance
0	2019-03-28	44655
1	2019-03-29	33769
2	2019-03-30	41899
3	2019-03-31	38444
5	2019-04-01	40477

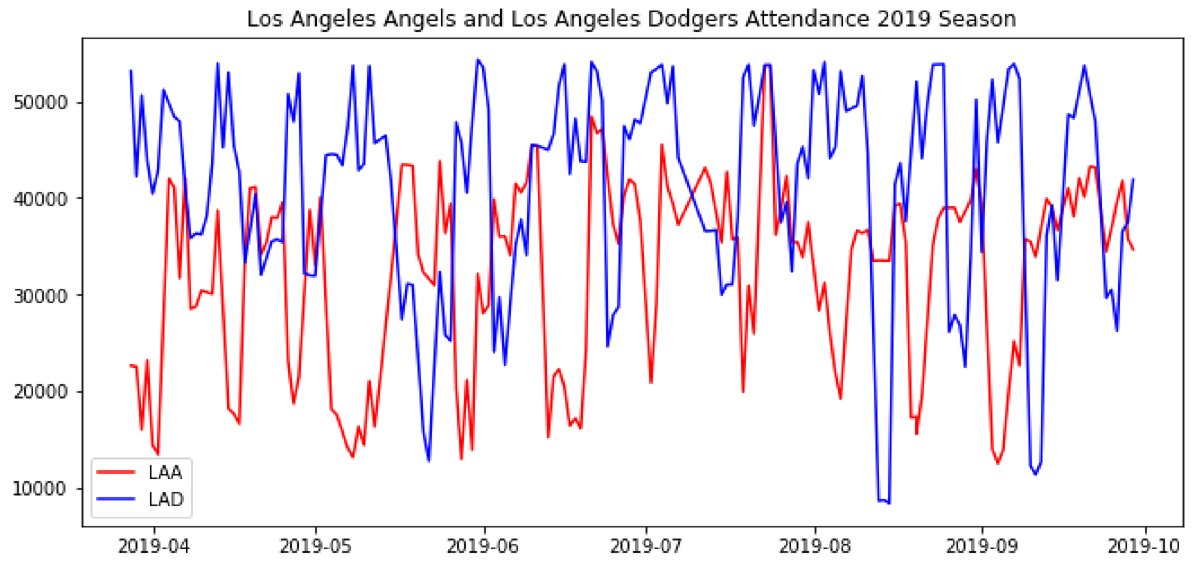
In [20]:
```
import datetime as dt
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import matplotlib.patches as mpatches

fig, ax = plt.subplots(figsize=(11,5))
ax.plot(OAKdf.Date, OAKdf.Attendance, color = 'green', label='OAK')
ax.plot(SFGdf.Date, SFGdf.Attendance, color = 'orange', label='SFG')
ax.set_title('San Francisco Giants and Oakland Athletics Attendance 2019
Season')
ax.legend()

plt.show()
```



In [10]:
```
Teams = ['LAA']

list_of_df = list()

#Creating dataframe
for team in Teams:
    for year in Years:
        #Will retrieve data from each year for each team
        url = 'https://www.baseball-reference.com/teams/' + team + '/' + 
year +'-schedule-scores.shtml'
        dfname = team + '_' + year
        html = requests.get(url).content
        df_list = pd.read_html(html)
        df = df_list[-1]

        #Formatting data table
        #rename columns
        df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
        df = df[df['GM_Num'].str.isdigit()]
        df = df[['Date','Attendance']]
        #Turn date into datetime object
        DayDate = df['Date'].str.split(", ", n = 1, expand = True)
        df['Date'] = DayDate[1] + ', ' + year
        df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
]]
        df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
        #Add to list which will be turned into a dataframe
        list_of_df.append(df)

#Create dataframe
bbattend = pd.concat(list_of_df)
LAAdf = bbattend
LAAdf = LAAdf[pd.notnull(LAAdf['Attendance'])]
LAAdf['Attendance'] = LAAdf['Attendance'].astype(int)
```

```
In [11]: Teams = ['LAD']
         #Teams = ['SFG', 'OAK', 'LAA', 'LAD', 'CHC', 'CHW']
         #for year in range(2016,2020):
         #    Years.append(str(year))

         list_of_df = list()

         #Creating dataframe
         for team in Teams:
             for year in Years:
                 #Will retrieve data from each year for each team
                 url = 'https://www.baseball-reference.com/teams/' + team + '/' +
         year +'-schedule-scores.shtml'
                 dfname = team + '_' + year
                 html = requests.get(url).content
                 df_list = pd.read_html(html)
                 df = df_list[-1]

                 #Formatting data table
                 #rename columns
                 df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
                 df = df[df['GM_Num'].str.isdigit()]
                 df = df[['Date','Attendance']]
                 #Turn date into datetime object
                 DayDate = df['Date'].str.split(", ", n = 1, expand = True)
                 df['Date'] = DayDate[1] + ', ' + year
                 df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
         ]]
                 df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
                 #Add to list which will be turned into a dataframe
                 list_of_df.append(df)

         #Create dataframe
         bbattend = pd.concat(list_of_df)
         LADdf = bbattend
         LADdf = LADdf[pd.notnull(LADdf['Attendance'])]
         LADdf['Attendance'] = LADdf['Attendance'].astype(int)
```

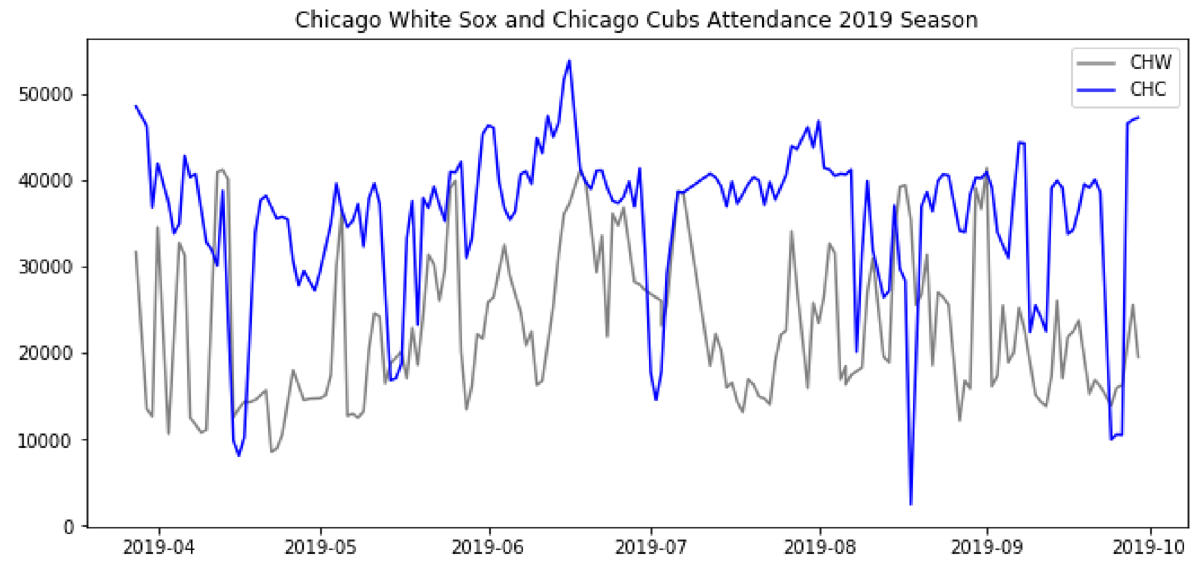
In [12]:
```
fig, ax = plt.subplots(figsize=(11,5))
ax.plot(LAAdf.Date, LAAdf.Attendance, color = 'red', label='LAA')
ax.plot(LADdf.Date, LADdf.Attendance, color = 'blue', label='LAD')
ax.set_title('Los Angeles Angels and Los Angeles Dodgers Attendance 2019
Season')
ax.legend()
plt.show()
```



In [13]:
```python
import requests
import pandas as pd
import numpy as np
from datetime import datetime
import re
import warnings
warnings.simplefilter('ignore')

#Specifying which teams and years to include in data set
#Teams = ['OAK']
Teams = ['CHC']
#Teams = ['SFG', 'OAK', 'LAA', 'LAD', 'CHC', 'CHW']
#for year in range(2016,2020):
#    Years.append(str(year))

list_of_df = list()

#Creating dataframe
for team in Teams:
    for year in Years:
        #Will retrieve data from each year for each team
        url = 'https://www.baseball-reference.com/teams/' + team + '/' +
year +'-schedule-scores.shtml'
        dfname = team + '_' + year
        html = requests.get(url).content
        df_list = pd.read_html(html)
        df = df_list[-1]

        #Formatting data table
        #rename columns
        df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
        df = df[df['GM_Num'].str.isdigit()]
        df = df[['Date','Attendance']]
        #Turn date into datetime object
        DayDate = df['Date'].str.split(", ", n = 1, expand = True)
        df['Date'] = DayDate[1] + ', ' + year
        df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
]]
        df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
        #Add to list which will be turned into a dataframe
        list_of_df.append(df)

#Create dataframe
bbattend = pd.concat(list_of_df)
CHCdf = bbattend
CHCdf = CHCdf[pd.notnull(CHCdf['Attendance'])]
CHCdf['Attendance'] = CHCdf['Attendance'].astype(int)
```

In [14]:
```
import requests
import pandas as pd
import numpy as np
from datetime import datetime
import re
import warnings
warnings.simplefilter('ignore')

#Specifying which teams and years to include in data set
#Teams = ['OAK']
Teams = ['CHW']
#Teams = ['SFG', 'OAK', 'LAA', 'LAD', 'CHC', 'CHW']
#for year in range(2016,2020):
#    Years.append(str(year))

list_of_df = list()

#Creating dataframe
for team in Teams:
    for year in Years:
        #Will retrieve data from each year for each team
        url = 'https://www.baseball-reference.com/teams/' + team + '/' + 
year +'-schedule-scores.shtml'
        dfname = team + '_' + year
        html = requests.get(url).content
        df_list = pd.read_html(html)
        df = df_list[-1]

        #Formatting data table
        #rename columns
        df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
        df = df[df['GM_Num'].str.isdigit()]
        df = df[['Date','Attendance']]
        #Turn date into datetime object
        DayDate = df['Date'].str.split(", ", n = 1, expand = True)
        df['Date'] = DayDate[1] + ', ' + year
        df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
]]
        df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
        #Add to list which will be turned into a dataframe
        list_of_df.append(df)

#Create dataframe
bbattend = pd.concat(list_of_df)
CHWdf = bbattend
CHWdf = CHWdf[pd.notnull(CHWdf['Attendance'])]
CHWdf['Attendance'] = CHWdf['Attendance'].astype(int)
```

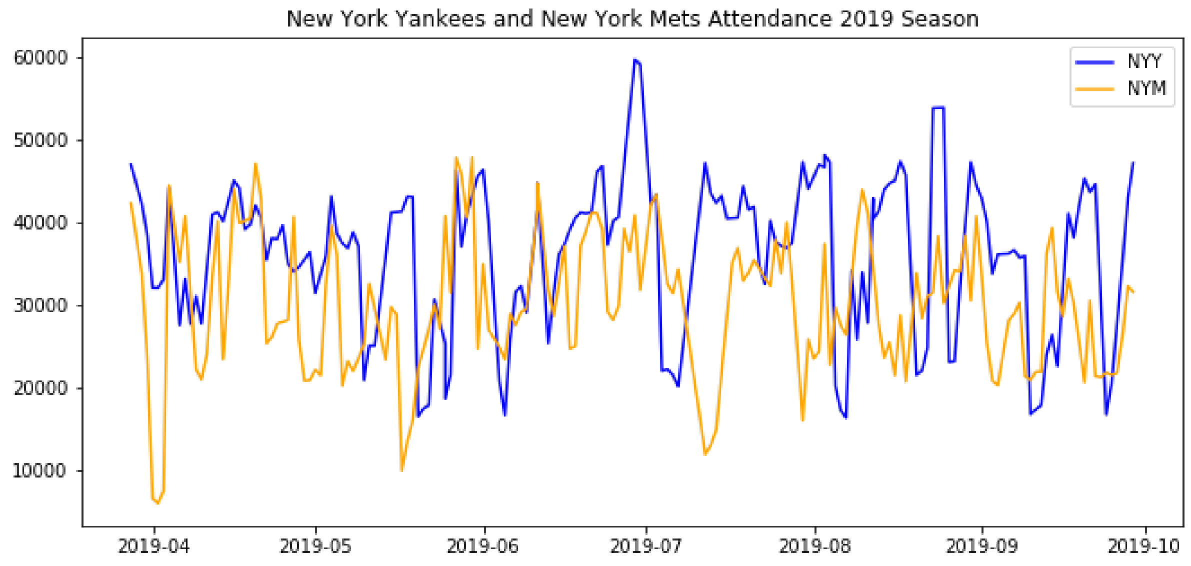
```
In [15]: fig, ax = plt.subplots(figsize=(11,5))
         ax.plot(CHWdf.Date, CHWdf.Attendance, color = 'gray', label='CHW')
         ax.plot(CHCdf.Date, CHCdf.Attendance, color = 'blue', label='CHC')
         ax.set_title('Chicago White Sox and Chicago Cubs Attendance 2019 Season'
         )
         ax.legend()
         plt.show()
```



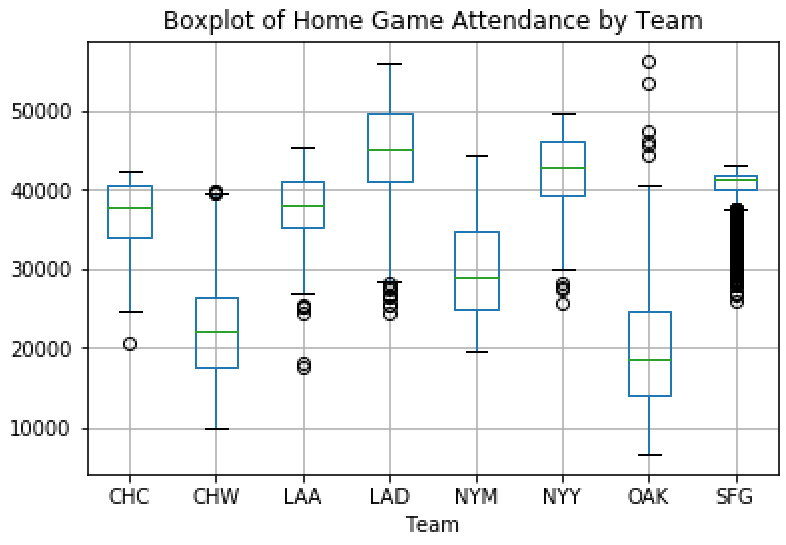
In [16]:
```
Teams = ['NYY']
#Teams = ['SFG', 'OAK', 'LAA', 'LAD', 'CHC', 'CHW']
#for year in range(2016,2020):
#    Years.append(str(year))

list_of_df = list()

#Creating dataframe
for team in Teams:
    for year in Years:
        #Will retrieve data from each year for each team
        url = 'https://www.baseball-reference.com/teams/' + team + '/' +
year +'-schedule-scores.shtml'
        dfname = team + '_' + year
        html = requests.get(url).content
        df_list = pd.read_html(html)
        df = df_list[-1]

        #Formatting data table
        #rename columns
        df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
        df = df[df['GM_Num'].str.isdigit()]
        df = df[['Date','Attendance']]
        #Turn date into datetime object
        DayDate = df['Date'].str.split(", ", n = 1, expand = True)
        df['Date'] = DayDate[1] + ', ' + year
        df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
]]
        df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
        #Add to list which will be turned into a dataframe
        list_of_df.append(df)

#Create dataframe
bbattend = pd.concat(list_of_df)
NYYdf = bbattend
NYYdf = NYYdf[pd.notnull(NYYdf['Attendance'])]
NYYdf['Attendance'] = NYYdf['Attendance'].astype(int)
```

In [17]:
```
Teams = ['NYM']

list_of_df = list()

#Creating dataframe
for team in Teams:
    for year in Years:
        #Will retrieve data from each year for each team
        url = 'https://www.baseball-reference.com/teams/' + team + '/' +
year +'-schedule-scores.shtml'
        dfname = team + '_' + year
        html = requests.get(url).content
        df_list = pd.read_html(html)
        df = df_list[-1]

        #Formatting data table
        #rename columns
        df.rename(columns={"Gm#": "GM_Num"}, inplace = True)
        df = df[df['GM_Num'].str.isdigit()]
        df = df[['Date','Attendance']]
        #Turn date into datetime object
        DayDate = df['Date'].str.split(", ", n = 1, expand = True)
        df['Date'] = DayDate[1] + ', ' + year
        df['Date'] = [re.sub("\s\(\d+\)", "", str(x)) for x in df['Date'
]]
        df['Date'] = pd.to_datetime(df['Date'], format='%b %d, %Y')
        #Add to list which will be turned into a dataframe
        list_of_df.append(df)

#Create dataframe
bbattend = pd.concat(list_of_df)
NYMdf = bbattend
NYMdf = NYMdf[pd.notnull(NYMdf['Attendance'])]
NYMdf['Attendance'] = NYMdf['Attendance'].astype(int)
```

```
In [18]: fig, ax = plt.subplots(figsize=(11,5))
         ax.plot(NYYdf.Date, NYYdf.Attendance, color = 'blue', label='NYY')
         ax.plot(NYMdf.Date, NYMdf.Attendance, color = 'orange', label='NYM')
         ax.set_title('New York Yankees and New York Mets Attendance 2019 Season'
         )
         ax.legend()
         plt.show()
```



```
In [19]: bbattendhome =pd.read_csv("bbattendhome.csv")
         bbattendhome['Year'] = pd.DatetimeIndex(bbattendhome['Date']).year
         #bbattendhome.head()
         bbattendhome.boxplot(column='Attendance', by='Team')
         plt.title("Boxplot of Home Game Attendance by Team")
         plt.suptitle("")

         plt.show()
```



```
In [ ]:
```

In [22]:
```
import pandas as pd
import numpy as np

homegames = pd.read_csv("homegames.csv")
homegames.drop(columns=['Unnamed: 0', 'Inn'], inplace = True)
homegames['Win_Per'] = 100*homegames['Win_Per']
homegames['Win_Per_Same_Mkt_Team'] = 100*homegames['Win_Per_Same_Mkt_Tea
m']
homegames.dropna(inplace = True)
homegames.head()
```

Out[22]:

	GM_Num	Date	Team	Home	Opp	W/L	R	RA	W-L	Rank	...	Losses_Same_Mkt_Team	da
0	7	2010-04-12	CHC	1	MIL	W	9	5	3-4	3	...	4	
1	8	2010-04-14	CHC	1	MIL	W	7	6	4-4	3	...	5	
2	9	2010-04-15	CHC	1	MIL	L	6	8	4-5	3	...	5	
3	10	2010-04-16	CHC	1	HOU	W	7	2	5-5	2	...	6	
4	11	2010-04-17	CHC	1	HOU	L	3	4	5-6	3	...	7	

5 rows × 47 columns

```
In [2]: list(homegames.columns)
```

```
Out[2]: ['GM_Num',
         'Date',
         'Team',
         'Home',
         'Opp',
         'W/L',
         'R',
         'RA',
         'W-L',
         'Rank',
         'GB',
         'Time',
         'Night',
         'Attendance',
         'Streak',
         'Game_Win',
         'Wins',
         'Losses',
         'Net_Wins',
         'Win_Per',
         'DayOfWeek',
         'Year',
         'Same_Mkt_Team',
         'Monday',
         'Tuesday',
         'Wednesday',
         'Thursday',
         'Friday',
         'Saturday',
         'Sunday',
         'game_id',
         'Date_Same_Mkt_Team',
         'Team_Same_Mkt_Team',
         'Net_Wins_Same_Mkt_Team',
         'Win_Per_Same_Mkt_Team',
         'Streak_Same_Mkt_Team',
         'date_diff',
         'CHC',
         'CHW',
         'LAA',
         'LAD',
         'NYM',
         'NYY',
         'OAK',
         'SFG']
```

```
In [3]: homegames.Win_Per.unique()
```

```
Out[3]: array([42.85714286, 50.        , 44.44444444, ..., 49.61832061,
               48.87218045, 47.77070064])
```

```
In [4]: import statsmodels.formula.api as smf

        attend_ols1 = smf.ols('Attendance ~  Streak_Same_Mkt_Team', data=homegam
        es).fit(cov_type = 'HC3')
        print(attend_ols1.summary())
```

```
                            OLS Regression Results                            
=======================================================================
=======
Dep. Variable:             Attendance   R-squared:                     
0.000
Model:                            OLS   Adj. R-squared:                
-0.000
Method:                 Least Squares   F-statistic:                   
0.7714
Date:                Thu, 19 Dec 2019   Prob (F-statistic):            
0.380
Time:                        10:01:57   Log-Likelihood:                
-67866.
No. Observations:                6372   AIC:                         1.
357e+05
Df Residuals:                    6370   BIC:                         1.
358e+05
Df Model:                           1                                  
Covariance Type:                  HC3                                  
=======================================================================
=================
                           coef    std err          z      P>|z|      
[0.025      0.975]
-----------------------------------------------------------------------
-----------------
Intercept             3.432e+04    128.038    268.065      0.000     3.4
1e+04    3.46e+04
Streak_Same_Mkt_Team   -43.6365     49.684     -0.878      0.380     -14
1.015      53.742
=======================================================================
=======
Omnibus:                      396.406   Durbin-Watson:                 
0.306
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              
394.759
Skew:                          -0.563   Prob(JB):                      
1.90e-86
Kurtosis:                       2.530   Cond. No.                      
2.50
=======================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

In [5]:
```
#attend_ols2 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team', data=homeg
ames).fit(cov_type = 'HC3')
attend_ols2 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team', data=homega
mes).fit(cov_type = 'HC3')
print(attend_ols2.summary())
```

```
                            OLS Regression Results
========================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.004
Model:                            OLS   Adj. R-squared:
0.004
Method:                 Least Squares   F-statistic:
20.23
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
6.98e-06
Time:                        10:01:57   Log-Likelihood:
-67854.
No. Observations:                6372   AIC:                          1.
357e+05
Df Residuals:                    6370   BIC:                          1.
357e+05
Df Model:                           1
Covariance Type:                  HC3
========================================================================
==================
                            coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------
------------------
Intercept               3.76e+04    748.734     50.220      0.000     3.
61e+04    3.91e+04
Win_Per_Same_Mkt_Team   -64.0997     14.250     -4.498      0.000      -
92.030     -36.170
========================================================================
=======
Omnibus:                      393.193   Durbin-Watson:
0.307
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
404.744
Skew:                          -0.577   Prob(JB):
1.29e-88
Kurtosis:                       2.560   Cond. No.
275.
========================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

In [26]:
```
#attend_ols2 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team', data=homeg
ames).fit(cov_type = 'HC3')
attend_ols2 = smf.ols('Attendance ~  Wins_Same_Mkt_Team', data=homegames
).fit(cov_type = 'HC3')
print(attend_ols2.summary())
```

```
                            OLS Regression Results                            
======================================================================
=======
Dep. Variable:             Attendance   R-squared:                       
0.000
Model:                            OLS   Adj. R-squared:                  
-0.000
Method:                 Least Squares   F-statistic:                   
0.09760
Date:                Thu, 19 Dec 2019   Prob (F-statistic):              
0.755
Time:                        12:08:38   Log-Likelihood:                
-67867.
No. Observations:                6372   AIC:                         1.
357e+05
Df Residuals:                    6370   BIC:                         1.
358e+05
Df Model:                           1                                         
Covariance Type:                  HC3                                         
======================================================================
===============
                         coef    std err          z      P>|z|      [0.
025      0.975]
----------------------------------------------------------------------
---------------
Intercept           3.425e+04    255.893    133.860      0.000    3.38e
+04    3.48e+04
Wins_Same_Mkt_Team     1.6404      5.251      0.312      0.755      -8.
651      11.932
======================================================================
=======
Omnibus:                      396.619   Durbin-Watson:                   
0.306
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              
394.197
Skew:                          -0.562   Prob(JB):                     
2.52e-86
Kurtosis:                       2.529   Cond. No.                     
94.3
======================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

```
In [27]: attend_ols2 = smf.ols('Attendance ~  Wins_Same_Mkt_Team + Losses_Same_Mk
t_Team', data=homegames).fit(cov_type = 'HC3')
print(attend_ols2.summary())
```

```
                            OLS Regression Results
==============================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.011
Model:                            OLS   Adj. R-squared:
0.011
Method:                 Least Squares   F-statistic:
36.07
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
2.65e-16
Time:                        12:09:08   Log-Likelihood:
-67832.
No. Observations:                6372   AIC:                             1.
357e+05
Df Residuals:                    6369   BIC:                             1.
357e+05
Df Model:                           2
Covariance Type:                  HC3
==============================================================================
=================
                          coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
-----------------
Intercept            3.374e+04    264.341    127.621      0.000    3.3
2e+04    3.43e+04
Wins_Same_Mkt_Team    -68.8691      9.427     -7.305      0.000     -8
7.346     -50.392
Losses_Same_Mkt_Team   87.5573     10.320      8.484      0.000      6
7.331    107.784
==============================================================================
=======
Omnibus:                      391.387   Durbin-Watson:
0.309
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
408.035
Skew:                          -0.582   Prob(JB):
2.49e-89
Kurtosis:                       2.573   Cond. No.
133.
==============================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

```
In [6]: attend_ols3 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team + Win_Per', d
        ata=homegames).fit(cov_type = 'HC3')
        print(attend_ols3.summary())
```

```
                            OLS Regression Results
==========================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.098
Model:                            OLS   Adj. R-squared:
0.097
Method:                 Least Squares   F-statistic:
210.9
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
1.97e-89
Time:                        10:01:58   Log-Likelihood:
-67540.
No. Observations:                6372   AIC:                           1.
351e+05
Df Residuals:                    6369   BIC:                           1.
351e+05
Df Model:                           2
Covariance Type:                  HC3
==========================================================================
==================
                            coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------
------------------
Intercept              1.918e+04   1240.088     15.464      0.000    1.
67e+04    2.16e+04
Win_Per_Same_Mkt_Team   -44.8331     15.826     -2.833      0.005      -
75.852     -13.815
Win_Per                 340.6958     16.962     20.086      0.000      3
07.451     373.941
==========================================================================
=======
Omnibus:                      374.171   Durbin-Watson:
0.361
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
441.523
Skew:                          -0.642   Prob(JB):
1.33e-96
Kurtosis:                       2.881   Cond. No.
577.
==========================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

```
In [7]: from sklearn import linear_model                          # ols, ridge,
         lasso
        from sklearn.preprocessing import StandardScaler

        var_list = ['Night', 'Streak', 'Wins', 'Losses', 'Net_Wins', 'Win_Per',
        'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sun
        day', 'Net_Wins_Same_Mkt_Team', 'Win_Per_Same_Mkt_Team', 'Streak_Same_Mk
        t_Team', 'LAA', 'LAD', 'CHC', 'CHW', 'NYY', 'NYM', 'SFG', 'OAK']
        #var_list = ['Win_Per', 'Win_Per_Same_Mkt_Team']
        X = StandardScaler().fit_transform(homegames[var_list])
        lasso_model_best_alpha = linear_model.LassoCV(cv=10).fit(X, homegames['A
        ttendance'])
        lasso_model_alpha = lasso_model_best_alpha.alpha_
        print('The best alpha from the candidate alphas is {0}.'.format(lasso_mo
        del_best_alpha.alpha_))
        print(lasso_model_best_alpha.coef_)

        #Drop Losses, Thursday
```

```
/Users/laurel/anaconda3/lib/python3.7/site-packages/sklearn/preprocessi
ng/data.py:645: DataConversionWarning: Data with input dtype int64, flo
at64 were all converted to float64 by StandardScaler.
  return self.partial_fit(X, y)
/Users/laurel/anaconda3/lib/python3.7/site-packages/sklearn/base.py:46
4: DataConversionWarning: Data with input dtype int64, float64 were all
converted to float64 by StandardScaler.
  return self.fit(X, **fit_params).transform(X)

The best alpha from the candidate alphas is 5.37129426477929.
[ 5.48184784e+01 -1.43178023e+02  2.62506606e+02  0.00000000e+00
  1.46291000e+03  1.59847904e+02 -2.89230773e+02 -1.47103284e+02
 -1.50288663e+02 -0.00000000e+00  9.03450631e+02  1.66058123e+03
  1.04431001e+03 -2.04533722e+02 -9.35266522e+01  3.58201884e+01
  3.55785387e+02  2.23385916e+03 -5.37902474e+00 -4.50008148e+03
  1.17062955e+03 -2.13763152e+03  9.61136069e+02 -5.59203093e+03]
```

```
In [28]: attend_ols4 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team + Streak_Same
         _Mkt_Team + Wins_Same_Mkt_Team + Losses_Same_Mkt_Team', data=homegames).
         fit(cov_type = 'HC3')
         print(attend_ols4.summary())
```

```
                            OLS Regression Results
========================================================================
=======
Dep. Variable:              Attendance   R-squared:
0.011
Model:                             OLS   Adj. R-squared:
0.010
Method:                  Least Squares   F-statistic:
18.29
Date:                 Thu, 19 Dec 2019   Prob (F-statistic):
5.98e-15
Time:                         12:43:09   Log-Likelihood:
-67831.
No. Observations:                 6372   AIC:                         1.
357e+05
Df Residuals:                     6367   BIC:                         1.
357e+05
Df Model:                            4
Covariance Type:                   HC3
========================================================================
==================
                            coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------
------------------
Intercept              3.349e+04   1105.867     30.281      0.000    3.
13e+04    3.57e+04
Win_Per_Same_Mkt_Team     4.8615     20.703      0.235      0.814     -
35.716      45.439
Streak_Same_Mkt_Team     48.4382     51.982      0.932      0.351     -
53.444     150.320
Wins_Same_Mkt_Team      -73.3412     13.691     -5.357      0.000    -1
00.175     -46.507
Losses_Same_Mkt_Team     92.2478     14.914      6.185      0.000
63.016     121.479
========================================================================
=======
Omnibus:                   390.514   Durbin-Watson:
0.309
Prob(Omnibus):               0.000   Jarque-Bera (JB):
407.303
Skew:                       -0.582   Prob(JB):
3.59e-89
Kurtosis:                    2.574   Cond. No.
597.
========================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

```
In [29]: attend_ols5 = smf.ols('Attendance ~  Wins_Same_Mkt_Team + Losses_Same_Mk
         t_Team + Wins + Losses', data=homegames).fit(cov_type = 'HC3')
         print(attend_ols5.summary())
```

```
                            OLS Regression Results
==============================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.151
Model:                            OLS   Adj. R-squared:
0.150
Method:                 Least Squares   F-statistic:
316.4
Date:                Thu, 19 Dec 2019   Prob (F-statistic):          1.
23e-248
Time:                        12:45:57   Log-Likelihood:
-67346.
No. Observations:                6372   AIC:                         1.
347e+05
Df Residuals:                    6367   BIC:                         1.
347e+05
Df Model:                           4
Covariance Type:                  HC3
==============================================================================
=================
                          coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
-----------------
Intercept            3.388e+04    258.195    131.218      0.000    3.3
4e+04    3.44e+04
Wins_Same_Mkt_Team     54.2327     83.088      0.653      0.514    -10
8.618     217.083
Losses_Same_Mkt_Team   97.9457     82.590      1.186      0.236     -6
3.927     259.819
Wins                  223.7920     82.471      2.714      0.007      6
2.153     385.431
Losses               -377.9694     82.613     -4.575      0.000    -53
9.887    -216.051
==============================================================================
=======
Omnibus:                      400.352   Durbin-Watson:
0.361
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
479.629
Skew:                          -0.671   Prob(JB):                    7.
08e-105
Kurtosis:                       2.933   Cond. No.
207.
==============================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

```
In [16]: attend_ols6 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team + Win_Per + C
         HC + CHW + LAA + LAD + NYM + NYY + OAK', data=homegames).fit(cov_type =
         'HC3')
         print(attend_ols6.summary())
```

```
                            OLS Regression Results
======================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.709
Model:                            OLS   Adj. R-squared:
0.709
Method:                 Least Squares   F-statistic:
1359.
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
0.00
Time:                        10:04:37   Log-Likelihood:
-63929.
No. Observations:                6372   AIC:                         1.
279e+05
Df Residuals:                    6362   BIC:                         1.
279e+05
Df Model:                           9
Covariance Type:                  HC3
======================================================================
==================
                            coef    std err          z      P>|z|
[0.025      0.975]
----------------------------------------------------------------------
------------------
Intercept              3.507e+04    733.235     47.827      0.000    3.
36e+04    3.65e+04
Win_Per_Same_Mkt_Team   -28.0054      9.064     -3.090      0.002      -
45.770     -10.241
Win_Per                 124.5626     10.377     12.004      0.000      1
04.225     144.901
CHC                   -3048.7062    178.790    -17.052      0.000    -33
99.127   -2698.285
CHW                     -1.7e+04    263.839    -64.447      0.000    -1.
75e+04   -1.65e+04
LAA                   -1796.9956    190.447     -9.436      0.000    -21
70.264   -1423.727
LAD                    4225.1934    251.065     16.829      0.000     37
33.114    4717.273
NYM                   -9757.2031    257.812    -37.846      0.000    -1.
03e+04   -9251.900
NYY                    1267.3238    212.099      5.975      0.000      8
51.617    1683.031
OAK                    -1.98e+04    301.801    -65.599      0.000    -2.
04e+04   -1.92e+04
======================================================================
=======
Omnibus:                      190.822   Durbin-Watson:
1.046
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
408.096
Skew:                           0.178   Prob(JB):
2.42e-89
Kurtosis:                       4.187   Cond. No.
701.
======================================================================
=======
Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```

In [21]:
```
attend_ols7 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team + Win_Per + C
HC + CHW + LAA + LAD + NYM + NYY + OAK + Wins + Losses', data=homegames)
.fit(cov_type = 'HC3')
print(attend_ols7.summary())
```

```
                            OLS Regression Results
========================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.721
Model:                            OLS   Adj. R-squared:
0.721
Method:                 Least Squares   F-statistic:
1246.
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
0.00
Time:                        10:54:16   Log-Likelihood:
-63795.
No. Observations:                6372   AIC:                          1.
276e+05
Df Residuals:                    6360   BIC:                          1.
277e+05
Df Model:                          11
Covariance Type:                  HC3
========================================================================
==================
                            coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------
------------------
Intercept              3.974e+04    816.761     48.652      0.000    3.
81e+04    4.13e+04
Win_Per_Same_Mkt_Team   -20.1713      8.518     -2.368      0.018      -
36.867      -3.476
Win_Per                  12.2678     13.782      0.890      0.373      -
14.744      39.279
CHC                   -2924.3251    176.635    -16.556      0.000    -32
70.522   -2578.128
CHW                   -1.655e+04    262.371    -63.076      0.000    -1.
71e+04    -1.6e+04
LAA                   -1889.7400    190.963     -9.896      0.000    -22
64.020   -1515.460
LAD                    3832.8077    246.212     15.567      0.000     33
50.241    4315.374
NYM                   -9455.9259    252.469    -37.454      0.000    -99
50.755   -8961.096
NYY                     702.1230    208.925      3.361      0.001      2
92.638    1111.608
OAK                   -1.987e+04    299.565    -66.320      0.000    -2.
05e+04   -1.93e+04
Wins                    127.2774      8.741     14.562      0.000      1
10.146     144.409
Losses                 -116.3835      9.158    -12.709      0.000     -1
34.332     -98.435
========================================================================
=======
Omnibus:                      209.598   Durbin-Watson:
1.085
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
401.127
Skew:                           0.247   Prob(JB):
7.88e-88
```

```
Kurtosis:                       4.126   Cond. No.
1.02e+03
==============================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
[2] The condition number is large, 1.02e+03. This might indicate that t
here are
strong multicollinearity or other numerical problems.
```

```
In [17]: attend_ols8 = smf.ols('Attendance ~  Win_Per_Same_Mkt_Team + Win_Per + C
         HC + CHW + LAA + LAD + NYM + NYY + OAK + Wins + Losses + Night + Tuesday
         + Wednesday + Thursday + Friday + Saturday + Sunday', data=homegames).fi
         t(cov_type = 'HC3')
         print(attend_ols8.summary())
```

```
                            OLS Regression Results
=======================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.758
Model:                            OLS   Adj. R-squared:
0.757
Method:                 Least Squares   F-statistic:
980.0
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
0.00
Time:                        10:16:30   Log-Likelihood:
-63344.
No. Observations:                6372   AIC:                         1.
267e+05
Df Residuals:                    6353   BIC:                         1.
269e+05
Df Model:                          18
Covariance Type:                  HC3
=======================================================================
==================
                            coef    std err          z      P>|z|
[0.025      0.975]
-----------------------------------------------------------------------
------------------
Intercept              3.744e+04    822.133     45.534      0.000    3.
58e+04     3.9e+04
Win_Per_Same_Mkt_Team   -21.3123      8.053     -2.647      0.008     -
37.095      -5.529
Win_Per                  12.5437     13.358      0.939      0.348     -
13.637      38.725
CHC                   -2889.3826    179.820    -16.068      0.000   -32
41.823   -2536.942
CHW                   -1.659e+04    248.760    -66.698      0.000   -1.
71e+04   -1.61e+04
LAA                   -1894.3620    188.835    -10.032      0.000   -22
64.472   -1524.252
LAD                    3809.1754    245.882     15.492      0.000    33
27.255    4291.095
NYM                   -9491.0096    239.800    -39.579      0.000   -99
61.009   -9021.010
NYY                     697.2456    204.560      3.409      0.001     2
96.316    1098.175
OAK                   -1.988e+04    281.160    -70.707      0.000   -2.
04e+04   -1.93e+04
Wins                    124.3029      8.375     14.841      0.000     1
07.887     140.718
Losses                 -114.4338      8.752    -13.075      0.000    -1
31.587     -97.280
Night                   147.3633    176.375      0.836      0.403    -1
98.326     493.053
Tuesday                 518.1997    275.013      1.884      0.060     -
20.815    1057.215
Wednesday               519.4735    275.668      1.884      0.060     -
20.826    1059.773
Thursday                946.2944    302.032      3.133      0.002     3
54.323    1538.266
```

```
Friday                 3424.7178    263.764     12.984      0.000    29
07.749    3941.686
Saturday               5459.1360    280.593     19.456      0.000    49
09.184    6009.088
Sunday                 3811.2748    299.608     12.721      0.000    32
24.055    4398.495
=======================================================================
=======
Omnibus:                      354.212   Durbin-Watson:
1.129
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
733.638
Skew:                           0.384   Prob(JB):                    4.
93e-160
Kurtosis:                       4.475   Cond. No.
1.10e+03
=======================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
[2] The condition number is large, 1.1e+03. This might indicate that th
ere are
strong multicollinearity or other numerical problems.
```

```
In [25]: attend_ols9 = smf.ols('Attendance ~ Win_Per_Same_Mkt_Team + Win_Per + CH
C + CHW + LAA + LAD + NYM + NYY + OAK + Wins + Losses + Night + Tuesday
 + Wednesday + Thursday + Friday + Saturday + Sunday + Wins_Same_Mkt_Tea
m + Losses_Same_Mkt_Team', data=homegames).fit(cov_type = 'HC3')
print(attend_ols9.summary())
```

```
                                OLS Regression Results
=======================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.759
Model:                            OLS   Adj. R-squared:
0.758
Method:                 Least Squares   F-statistic:
885.7
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
0.00
Time:                        12:08:08   Log-Likelihood:
-63336.
No. Observations:                6372   AIC:                         1.
267e+05
Df Residuals:                    6351   BIC:                         1.
269e+05
Df Model:                          20
Covariance Type:                  HC3
=======================================================================
==================
                            coef    std err          z      P>|z|
[0.025      0.975]
-----------------------------------------------------------------------
------------------
Intercept              3.665e+04    934.075     39.233      0.000    3.
48e+04    3.85e+04
Win_Per_Same_Mkt_Team    -8.9123     11.175     -0.797      0.425     -
30.816      12.991
Win_Per                  13.1137     13.491      0.972      0.331     -
13.328      39.555
CHC                   -2960.6463    178.323    -16.603      0.000   -33
10.153   -2611.139
CHW                   -1.663e+04    247.928    -67.069      0.000   -1.
71e+04   -1.61e+04
LAA                   -1868.7716    188.662     -9.905      0.000   -22
38.542   -1499.001
LAD                    3815.9625    244.410     15.613      0.000    33
36.928    4294.997
NYM                   -9436.3690    239.629    -39.379      0.000   -99
06.033   -8966.705
NYY                     712.1878    205.424      3.467      0.001     3
09.564    1114.812
OAK                   -1.991e+04    280.351    -71.019      0.000   -2.
05e+04   -1.94e+04
Wins                    271.4754     44.637      6.082      0.000     1
83.989     358.962
Losses                   37.1647     44.754      0.830      0.406     -
50.552     124.881
Night                   154.5391    176.510      0.876      0.381    -1
91.414     500.492
Tuesday                 497.0178    274.671      1.810      0.070     -
41.327    1035.362
Wednesday               499.9431    275.505      1.815      0.070     -
40.037    1039.924
Thursday                960.3424    301.715      3.183      0.001     3
68.992    1551.693
```

```
Friday                 3399.1808    263.730     12.889      0.000    28
82.279    3916.083
Saturday               5436.2385    280.771     19.362      0.000    48
85.937    5986.540
Sunday                 3791.5175    299.676     12.652      0.000    32
04.163    4378.872
Wins_Same_Mkt_Team     -164.3464     44.570     -3.687      0.000    -2
51.701     -76.992
Losses_Same_Mkt_Team   -133.7164     44.566     -3.000      0.003    -2
21.064     -46.369
=======================================================================
=======
Omnibus:                      369.875   Durbin-Watson:
1.131
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
766.531
Skew:                           0.399   Prob(JB):                    3.
55e-167
Kurtosis:                       4.500   Cond. No.
1.42e+03
=======================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
[2] The condition number is large, 1.42e+03. This might indicate that t
here are
strong multicollinearity or other numerical problems.
```

In [30]:
```
attend_ols7 = smf.ols('Attendance ~  Wins_Same_Mkt_Team + Losses_Same_Mk
t_Team + Wins + Losses + CHC + CHW + LAA + LAD + NYM + NYY + OAK', data=
homegames).fit(cov_type = 'HC3')
print(attend_ols7.summary())
```

```
                            OLS Regression Results
========================================================================
=======
Dep. Variable:             Attendance   R-squared:
0.722
Model:                            OLS   Adj. R-squared:
0.722
Method:                 Least Squares   F-statistic:
1248.
Date:                Thu, 19 Dec 2019   Prob (F-statistic):
0.00
Time:                        12:49:46   Log-Likelihood:
-63785.
No. Observations:                6372   AIC:                          1.
276e+05
Df Residuals:                    6360   BIC:                          1.
277e+05
Df Model:                          11
Covariance Type:                  HC3
========================================================================
=================
                           coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------
-----------------
Intercept             3.916e+04    175.897    222.646      0.000    3.8
8e+04    3.95e+04
Wins_Same_Mkt_Team    -209.8621     46.683     -4.496      0.000    -30
1.358    -118.366
Losses_Same_Mkt_Team  -170.4447     47.043     -3.623      0.000    -26
2.647     -78.242
Wins                   321.6648     46.797      6.874      0.000     22
9.944     413.386
Losses                  70.1336     46.801      1.499      0.134     -2
1.594     161.861
CHC                  -3006.7073    174.712    -17.209      0.000   -334
9.137   -2664.277
CHW                   -1.66e+04    261.115    -63.564      0.000   -1.7
1e+04   -1.61e+04
LAA                  -1893.7148    190.489     -9.941      0.000   -226
7.067   -1520.362
LAD                   3852.1668    244.505     15.755      0.000    337
2.945    4331.388
NYM                  -9399.3493    252.768    -37.186      0.000   -989
4.765   -8903.933
NYY                    728.2551    209.107      3.483      0.000     31
8.412    1138.098
OAK                  -1.992e+04    298.236    -66.800      0.000   -2.0
5e+04   -1.93e+04
========================================================================
=======
Omnibus:                      224.629   Durbin-Watson:
1.087
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
429.410
Skew:                           0.266   Prob(JB):
5.69e-94
```

```
Kurtosis:                       4.155   Cond. No.
789.
======================================================================
=======

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC3)
```