In [191…
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [827…
```
df=pd.read_csv("Casestudy2.csv")
```

In [829…
```
df
```

Out[829…

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
0	1.0	Cedar Falls University	4165.0	2.0	120.0	47,420	21.0	0
1	2.0	Oklahoma A&M	3746.0	3.0	58.0	47,420	21.0	0
2	3.0	Urbana College	4943.0	5.0	67.0	47,420	21.0	0
3	4.0	University of Bloomington	2366.0	9.0	83.0	47,420	21.0	0
4	5.0	Indiana A&M	1796.0	10.0	74.0	47,420	21.0	0
...	...	...	...	...	...	...	...	
72	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
73	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
74	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
75	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
76	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

77 rows × 72 columns

In [831…
```
df = df.dropna(axis=1, how='all')
df
```

Out[831…

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
0	1.0	Cedar Falls University	4165.0	2.0	120.0	47,420	21.0	0
1	2.0	Oklahoma A&M	3746.0	3.0	58.0	47,420	21.0	0
2	3.0	Urbana College	4943.0	5.0	67.0	47,420	21.0	0
3	4.0	University of Bloomington	2366.0	9.0	83.0	47,420	21.0	0
4	5.0	Indiana A&M	1796.0	10.0	74.0	47,420	21.0	0
…	…	…	…	…	…	…	…	
72	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
73	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
74	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
75	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
76	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

77 rows × 18 columns

In [833…

```
df = df.dropna(axis=0, how='any')
df
```

Out[833...

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
0	1.0	Cedar Falls University	4165.0	2.0	120.0	47,420	21.0	0
1	2.0	Oklahoma A&M	3746.0	3.0	58.0	47,420	21.0	0
2	3.0	Urbana College	4943.0	5.0	67.0	47,420	21.0	0
3	4.0	University of Bloomington	2366.0	9.0	83.0	47,420	21.0	0
4	5.0	Indiana A&M	1796.0	10.0	74.0	47,420	21.0	0
5	6.0	Minneapolis State University	1979.0	13.0	68.0	47,420	21.0	0
6	7.0	Mt Pleasant College	3866.0	1.0	109.0	46,198	57.0	0
7	8.0	University of Ames	5194.0	2.0	99.0	46,198	57.0	0
8	9.0	Ann Arbor University	1909.0	5.0	6.0	46,198	57.0	0
9	10.0	Evanston University	2523.0	6.0	55.0	46,198	57.0	0
10	11.0	Madison University	2734.0	8.0	17.0	46,198	57.0	0
11	12.0	Columbus University	2034.0	11.0	3.0	46,198	57.0	0
12	13.0	Lincoln University	6463.0	1.0	6.0	44,211	73.0	0
13	14.0	DeKalb College	3128.0	3.0	99.0	44,211	73.0	0
14	15.0	Pennsylvania A&M	2972.0	6.0	1.0	44,211	73.0	0
15	16.0	University of Bloomington	2158.0	8.0	68.0	44,211	73.0	0
16	17.0	Urbana College	2120.0	10.0	78.0	44,211	73.0	0
17	18.0	Minneapolis State University	1434.0	12.0	38.0	44,211	73.0	0

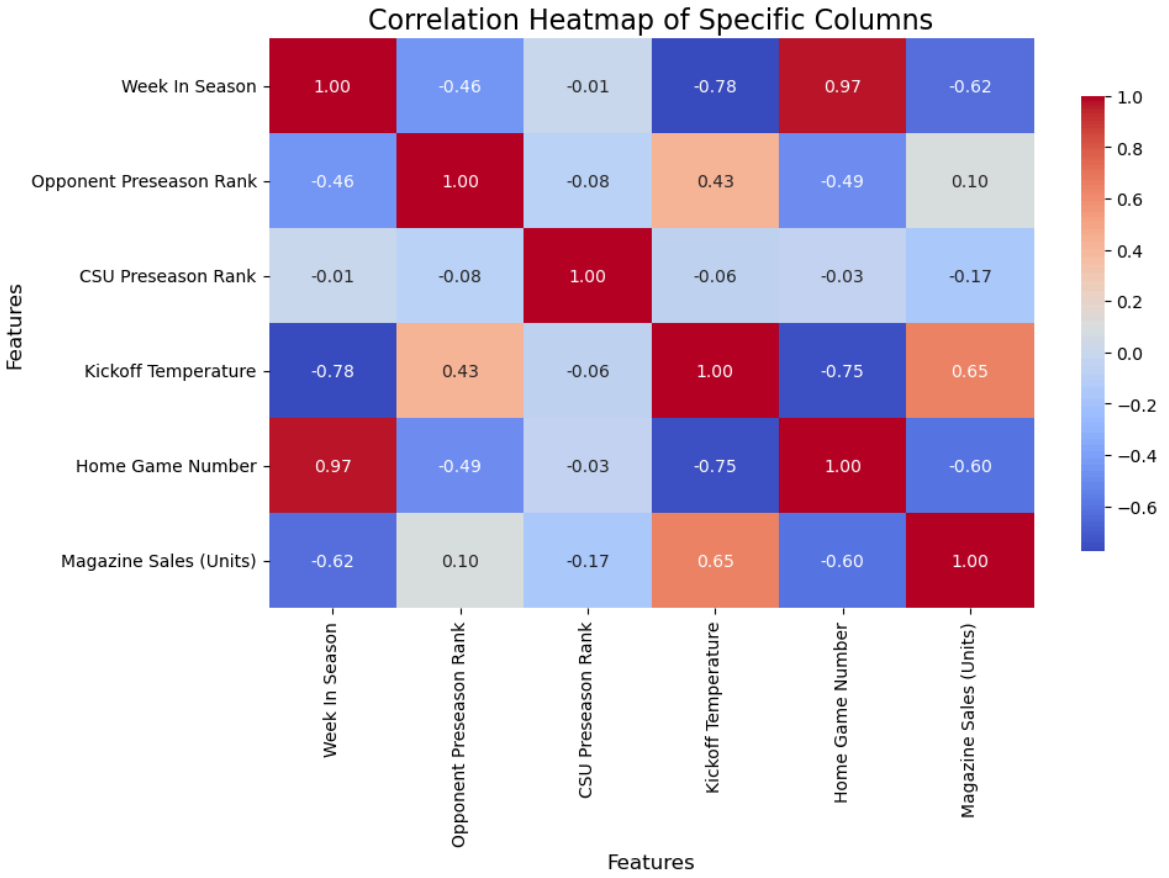
	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
18	19.0	University of Kalamazoo	2691.0	2.0	83.0	37,851	71.0	0
19	20.0	University of Ames	3202.0	3.0	79.0	37,851	71.0	0
20	21.0	Michigan A&M	1669.0	6.0	24.0	37,851	71.0	0
21	22.0	Columbus University	2194.0	8.0	15.0	37,851	71.0	0
22	23.0	Madison University	1536.0	9.0	4.0	37,851	71.0	0
23	24.0	Evanston University	763.0	11.0	94.0	37,851	71.0	0
24	25.0	Ohio A&M	3059.0	1.0	109.0	40,549	76.0	0
25	26.0	Northern Cincinnati University	2390.0	2.0	79.0	40,549	76.0	0
26	27.0	Pennsylvania A&M	3056.0	5.0	39.0	40,549	76.0	0
27	28.0	University of Bloomington	2298.0	8.0	93.0	40,549	76.0	0
28	29.0	Ann Arbor University	2956.0	9.0	17.0	40,549	76.0	0
29	30.0	Minneapolis State University	2324.0	12.0	59.0	40,549	76.0	0
30	31.0	LBJ University	2885.0	1.0	84.0	41,362	46.0	0
31	32.0	University of Ames	3677.0	3.0	52.0	41,362	46.0	0
32	33.0	University of Logan	1911.0	4.0	114.0	41,362	46.0	0
33	34.0	Indiana A&M	2404.0	6.0	22.0	41,362	46.0	0
34	35.0	Michigan A&M	2113.0	7.0	23.0	41,362	46.0	0
35	36.0	Madison University	2345.0	10.0	32.0	41,362	46.0	0
36	37.0	Evanston University	1969.0	11.0	69.0	41,362	46.0	0

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
37	38.0	Northern Cincinnati University	3634.0	1.0	68.0	42,843	31.0	0
38	39.0	Western New York University	2500.0	2.0	117.0	42,843	31.0	0
39	40.0	University of Tempe	2810.0	4.0	29.0	42,843	31.0	0
40	41.0	Ann Arbor University	3961.0	6.0	16.0	42,843	31.0	0
41	42.0	Pennsylvania A&M	2440.0	9.0	44.0	42,843	31.0	0
42	43.0	Urbana College	2017.0	10.0	45.0	42,843	31.0	0
43	44.0	Minneapolis State University	2173.0	12.0	26.0	42,843	31.0	0
44	45.0	Ohio A&M	4382.0	1.0	95.0	47,035	19.0	1
45	46.0	University of Ames	4037.0	2.0	77.0	47,035	19.0	0
46	47.0	Michigan A&M	2930.0	5.0	37.0	47,035	19.0	0
47	48.0	Columbus University	2757.0	7.0	11.0	47,035	19.0	0
48	49.0	Indiana A&M	2678.0	10.0	32.0	47,035	19.0	0
49	50.0	Madison University	2445.0	12.0	15.0	47,035	19.0	0
50	51.0	Letterman University	2985.0	1.0	111.0	54,584	19.0	0
51	52.0	Cedar Falls University	2800.0	3.0	120.0	54,584	11.0	0
52	53.0	Urbana College	2910.0	5.0	69.0	54,584	11.0	0
53	54.0	University of Bloomington	2500.0	7.0	86.0	54,584	11.0	0
54	55.0	Ann Arbor University	3721.0	8.0	7.0	54,584	11.0	0

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
55	56.0	Minneapolis State University	2500.0	12.0	23.0	54,584	11.0	0

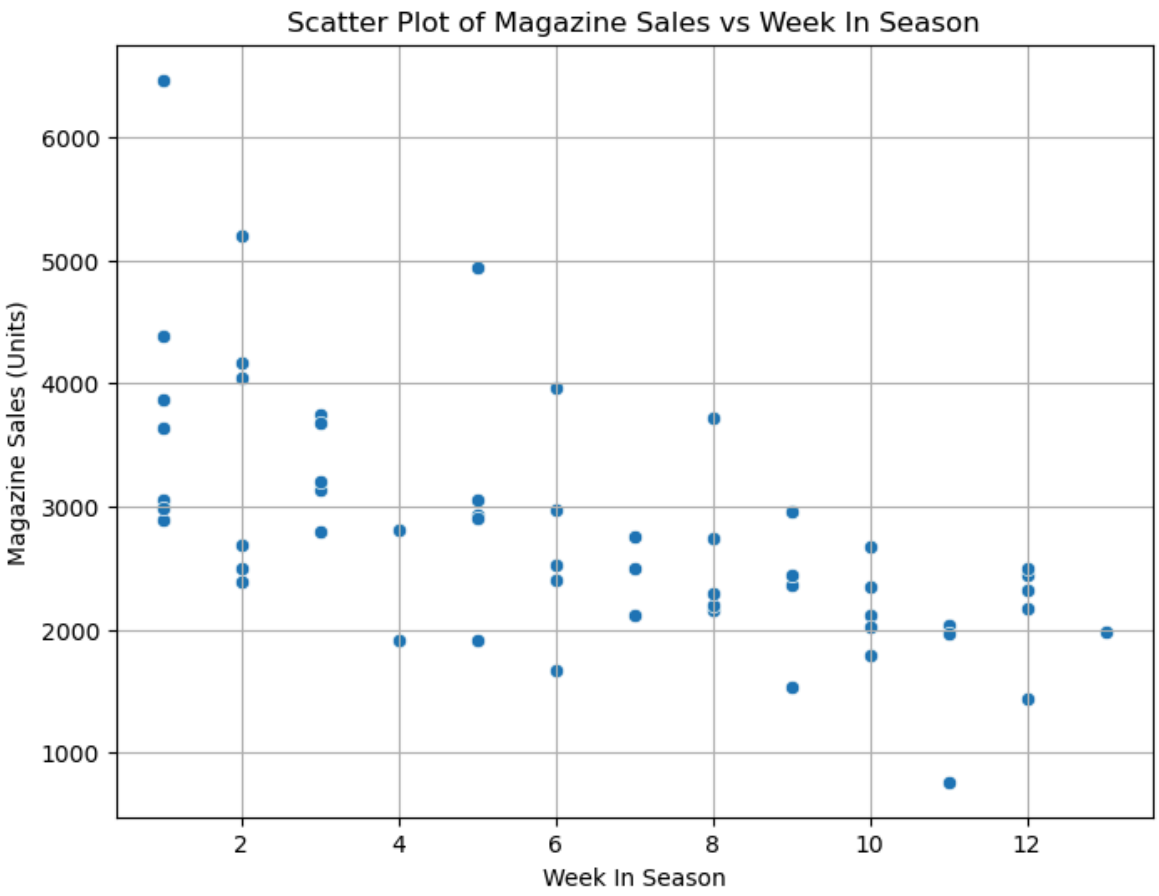
In [841…

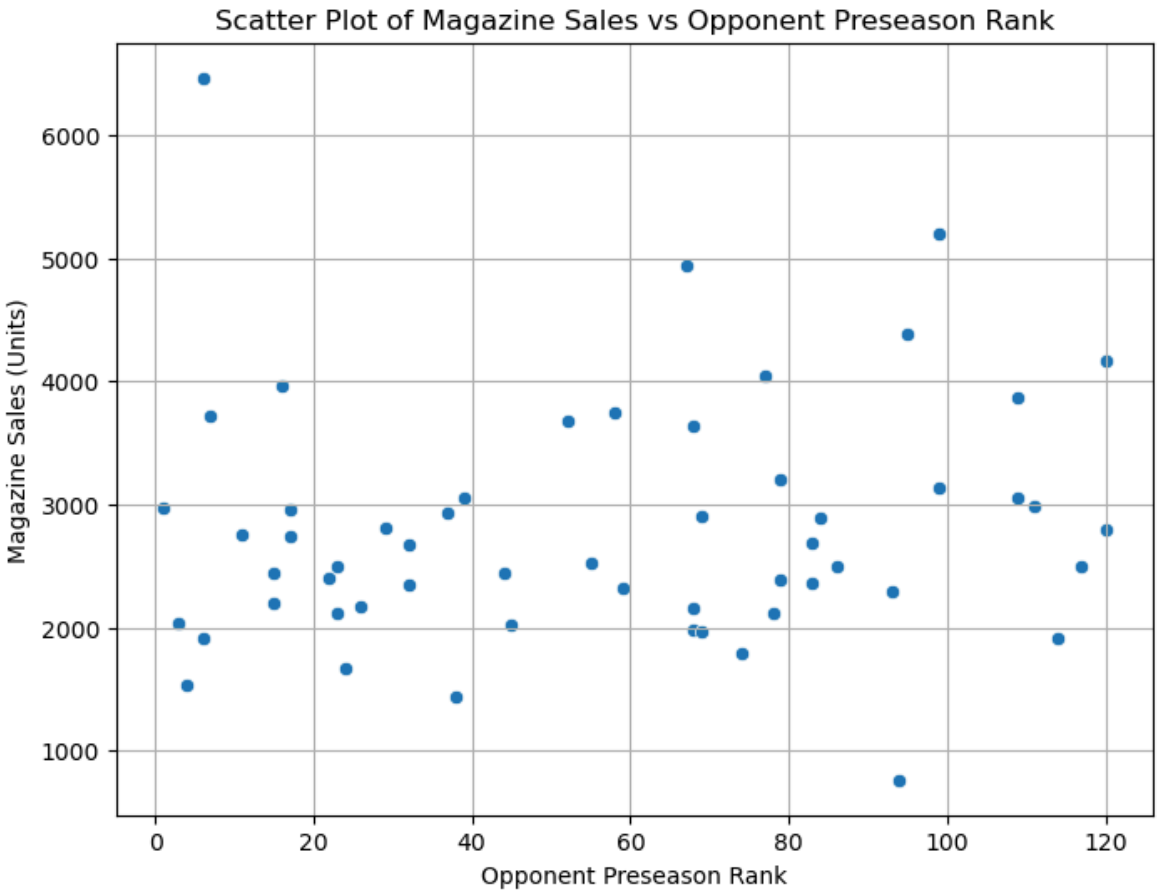
```
import seaborn as sns
import matplotlib.pyplot as plt

# List the specific columns you want to include in the heatmap
specific_columns = ['Week In Season', 'Opponent Preseason Rank', 'CSU Preseason
                    'Kickoff Temperature', 'Home Game Number', 'Magazine Sales (

# Filter the DataFrame to only include those specific columns
filtered_df = df[specific_columns]

# Ensure all selected columns are numeric
filtered_df = filtered_df.select_dtypes(include=[float, int])

# Calculate the correlation matrix for the specific columns
plt.figure(figsize=(10, 6))
sns.heatmap(filtered_df.corr(), annot=True, cmap='coolwarm', fmt='.2f',
            annot_kws={"size": 10}, cbar_kws={"shrink": .8})

# Customize labels and title
plt.xlabel('Features', fontsize=12)
plt.ylabel('Features', fontsize=12)
plt.title('Correlation Heatmap of Specific Columns', fontsize=16)
plt.show()
```

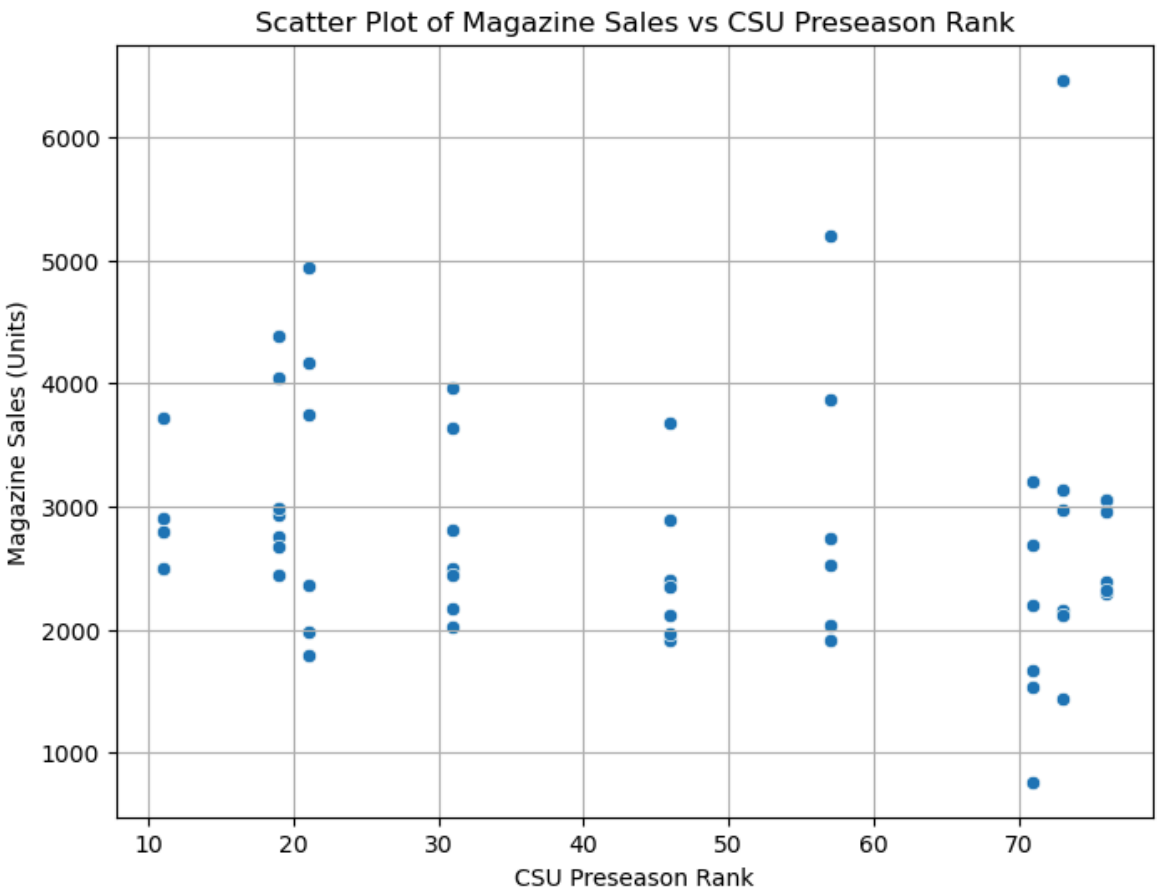


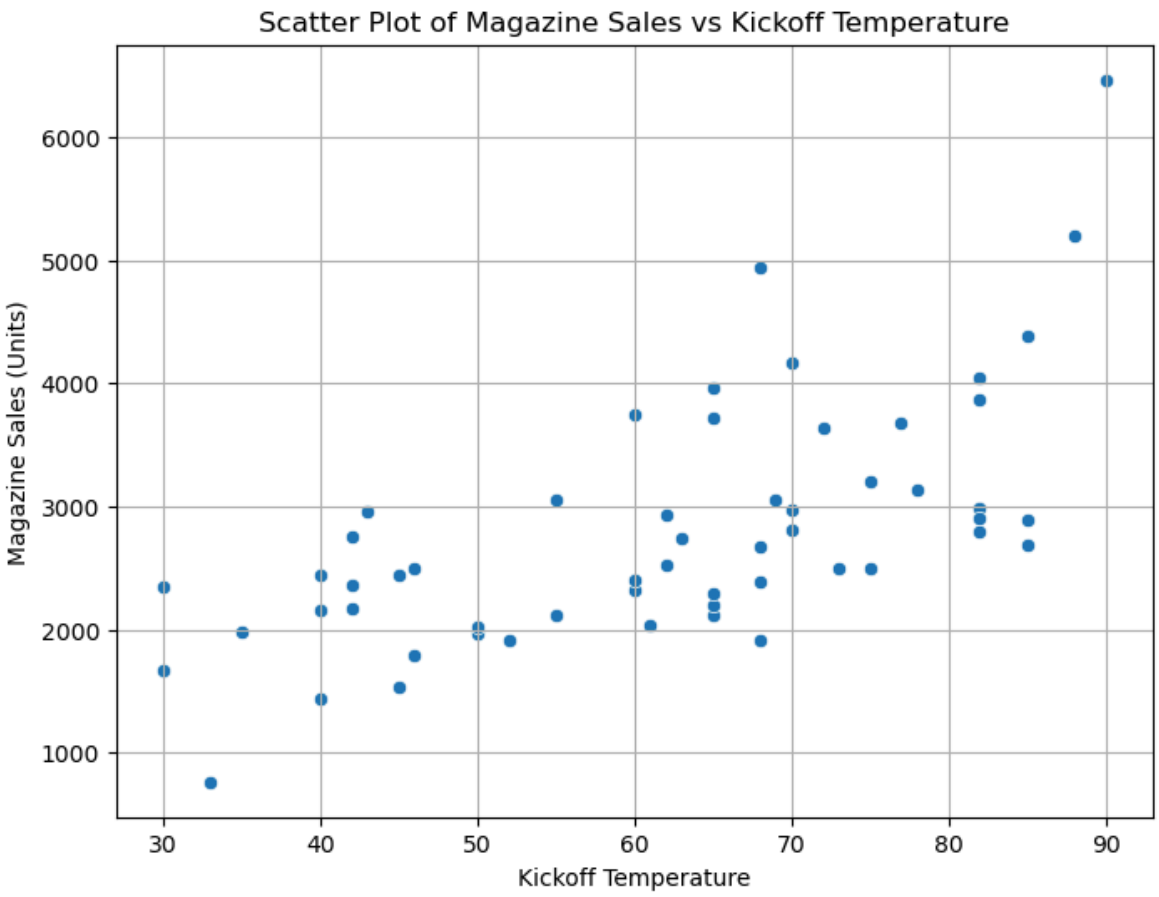
In [784…

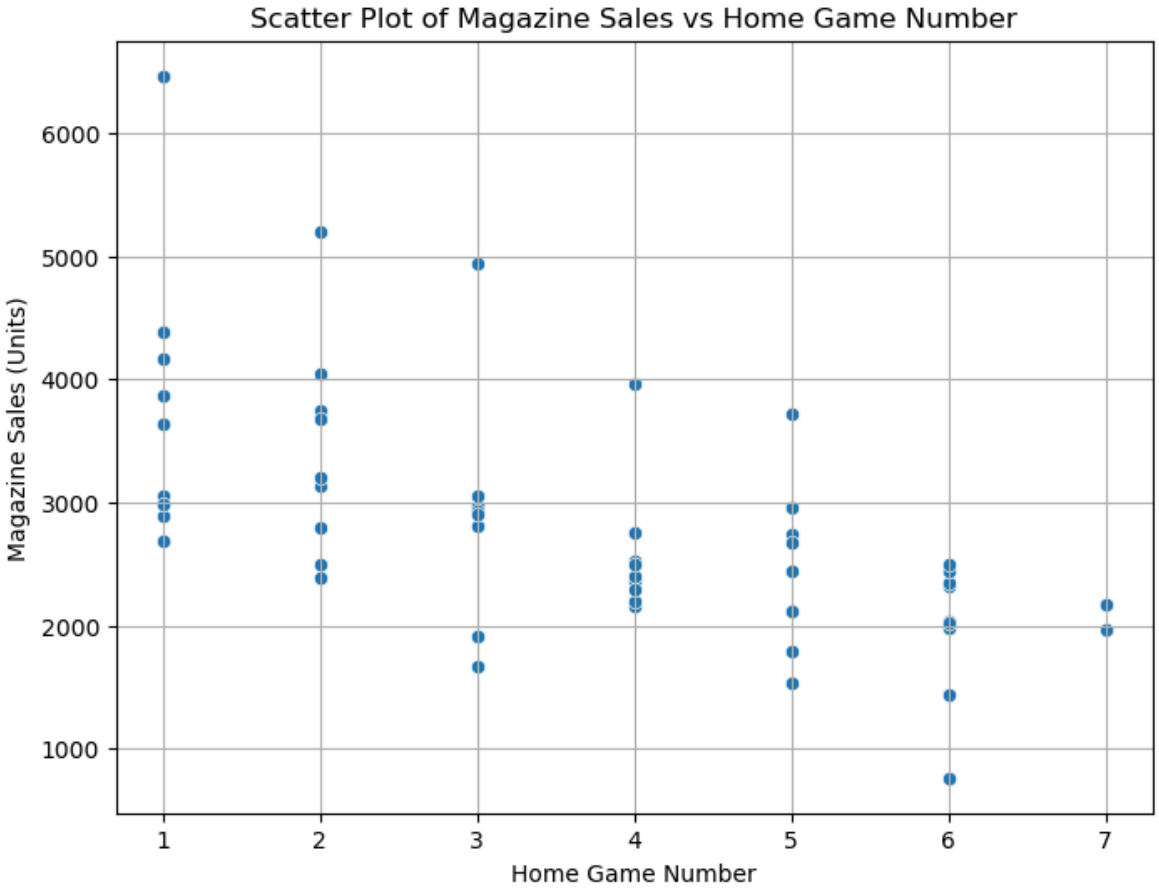
```
# List of quantitative columns
quantitative_columns = [
    'Week In Season', 'Opponent Preseason Rank',
    'CSU Preseason Rank', 'Kickoff Temperature', 'Home Game Number',
    'Opponent\'s Previous Season Number of Losses',
    'CSU\'s Previous Season Number of Wins', 'CSU\'s Previous Season Number of L
]

# Scatter plots
for col in quantitative_columns:
    plt.figure(figsize=(8, 6))
    sns.scatterplot(data=df, x=col, y='Magazine Sales (Units)')
    plt.title(f'Scatter Plot of Magazine Sales vs {col}')
    plt.xlabel(col)
    plt.ylabel('Magazine Sales (Units)')
    plt.grid(True)
    plt.show()
```

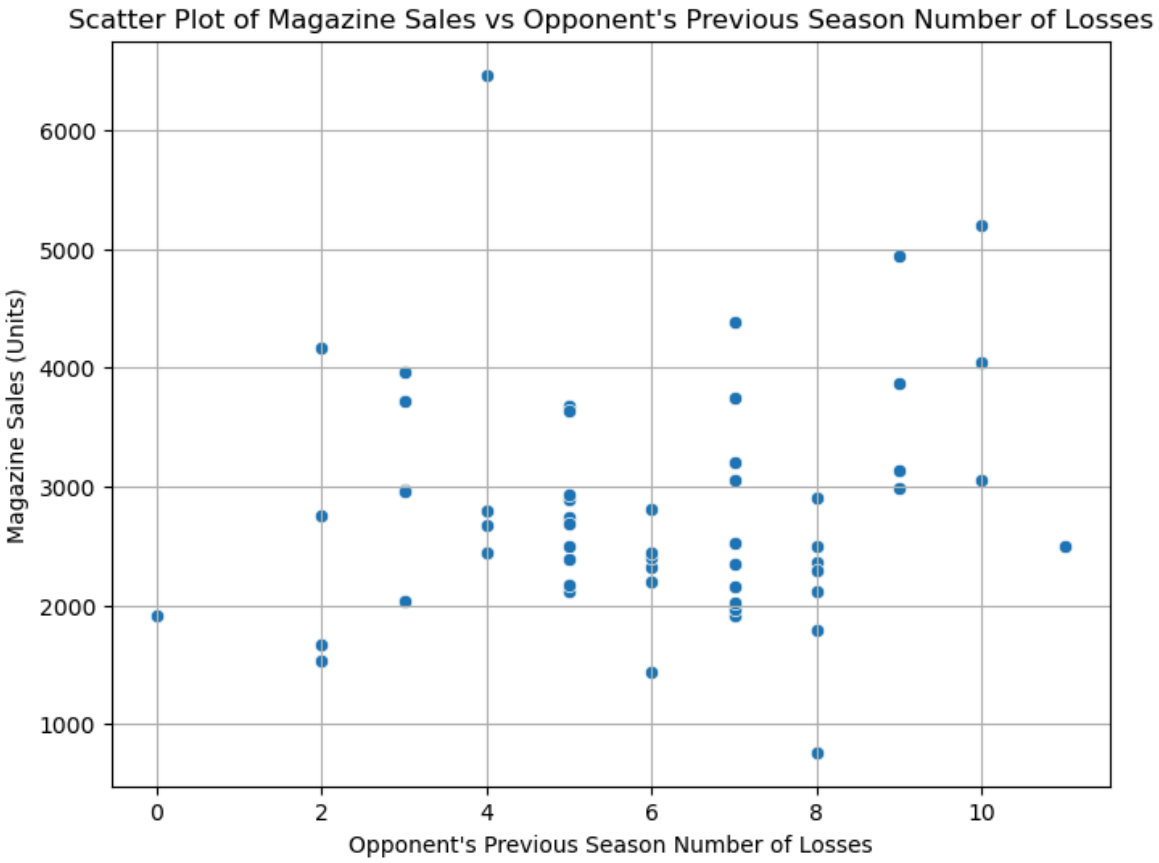

Scatter Plot of Magazine Sales vs Week In Season
Magazine Sales (Units)
Week In Season
1000
2000
3000
4000
5000
6000
2
4
6
8
10
12

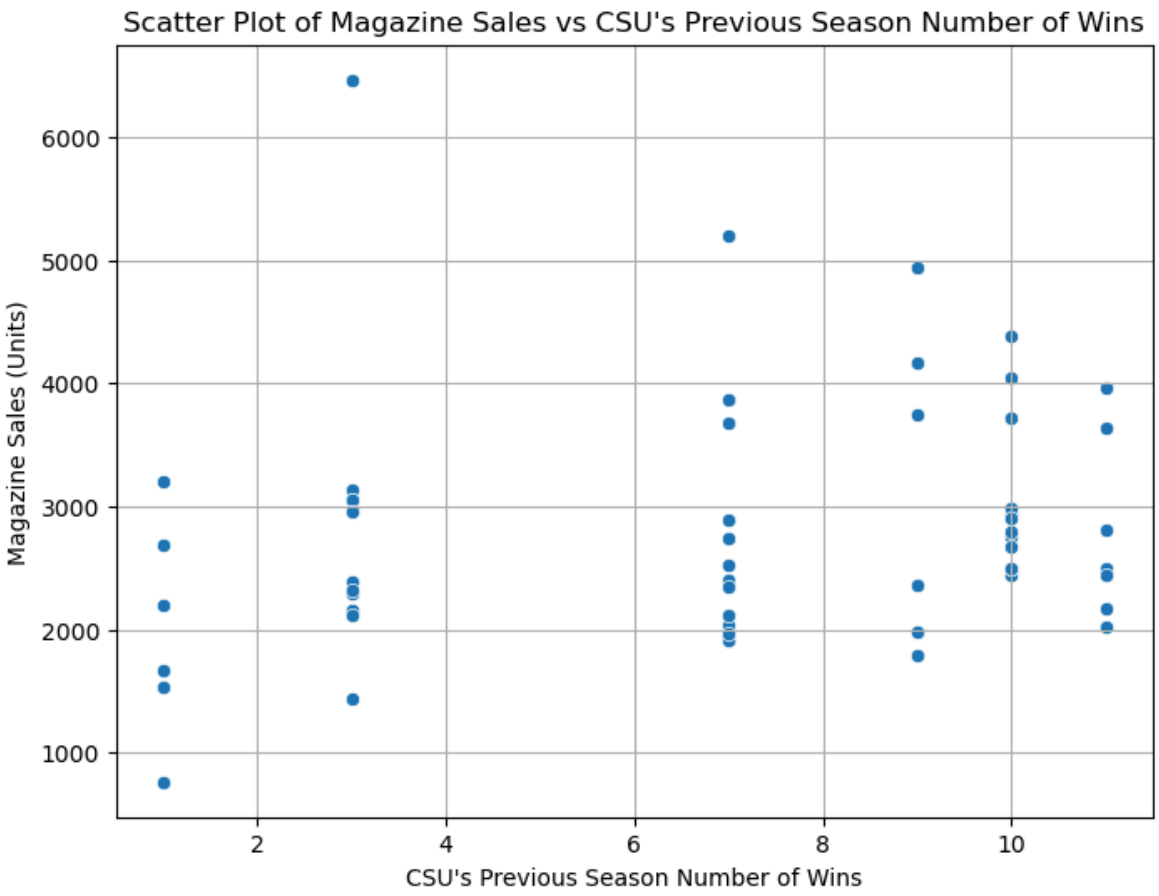

Scatter Plot of Magazine Sales vs Opponent Preseason Rank
Magazine Sales (Units)
Opponent Preseason Rank
1000
2000
3000
4000
5000
6000
0
20
40
60
80
100
120

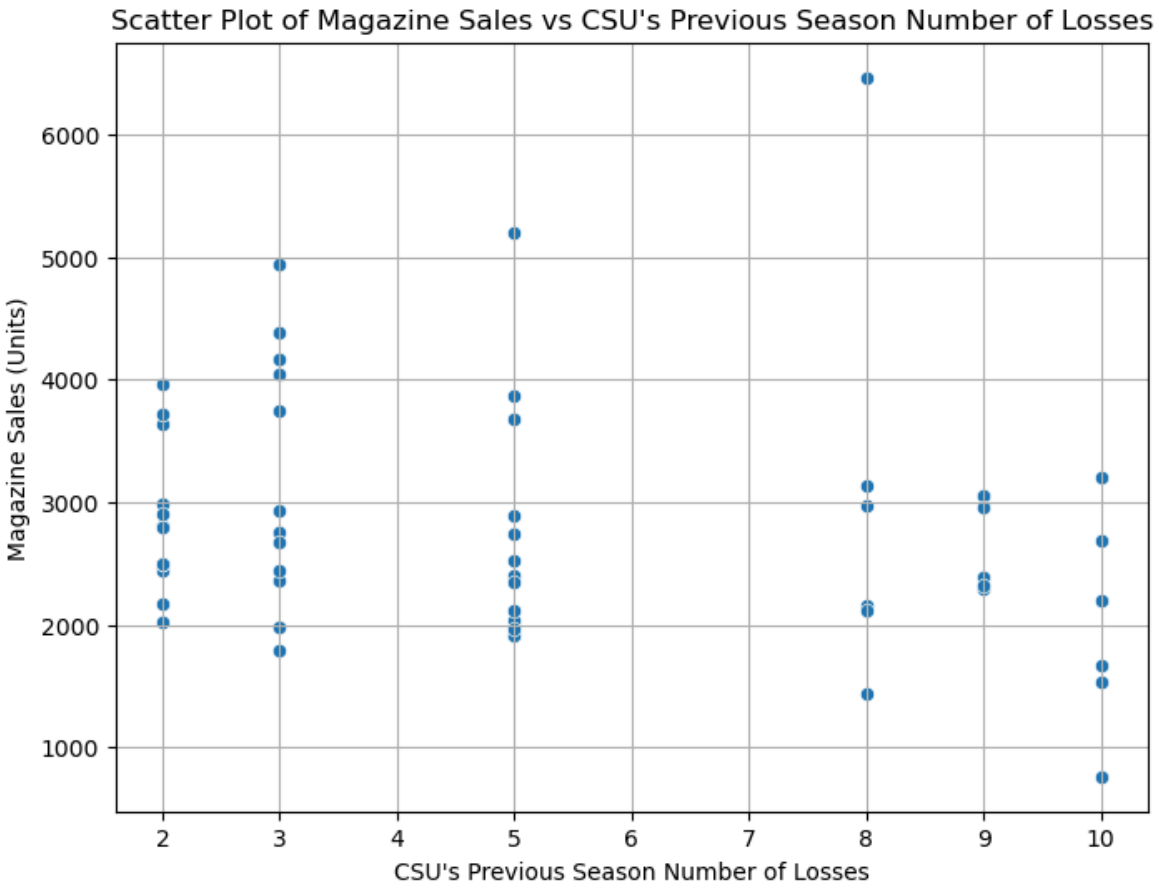

Scatter Plot of Magazine Sales vs CSU Preseason Rank
Magazine Sales (Units)
CSU Preseason Rank
1000
2000
3000
4000
5000
6000
10
20
30
40
50
60
70


Scatter Plot of Magazine Sales vs Kickoff Temperature
Magazine Sales (Units)
Kickoff Temperature
1000
2000
3000
4000
5000
6000
30
40
50
60
70
80
90


Scatter Plot of Magazine Sales vs Home Game Number
Magazine Sales (Units)
Home Game Number
1000
2000
3000
4000
5000
6000
1
2
3
4
5
6
7


Scatter Plot of Magazine Sales vs Opponent's Previous Season Number of Losses
Magazine Sales (Units)
Opponent's Previous Season Number of Losses
1000
2000
3000
4000
5000
6000
0
2
4
6
8
10

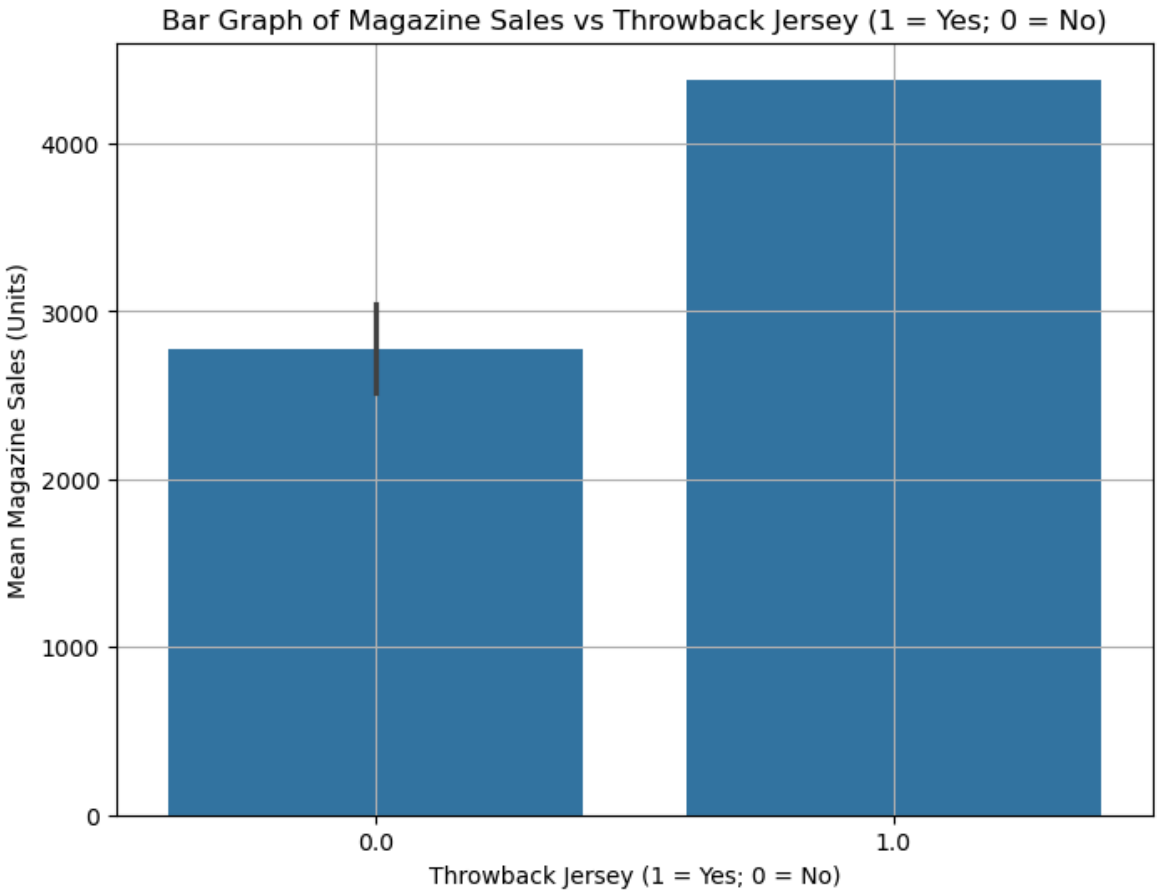


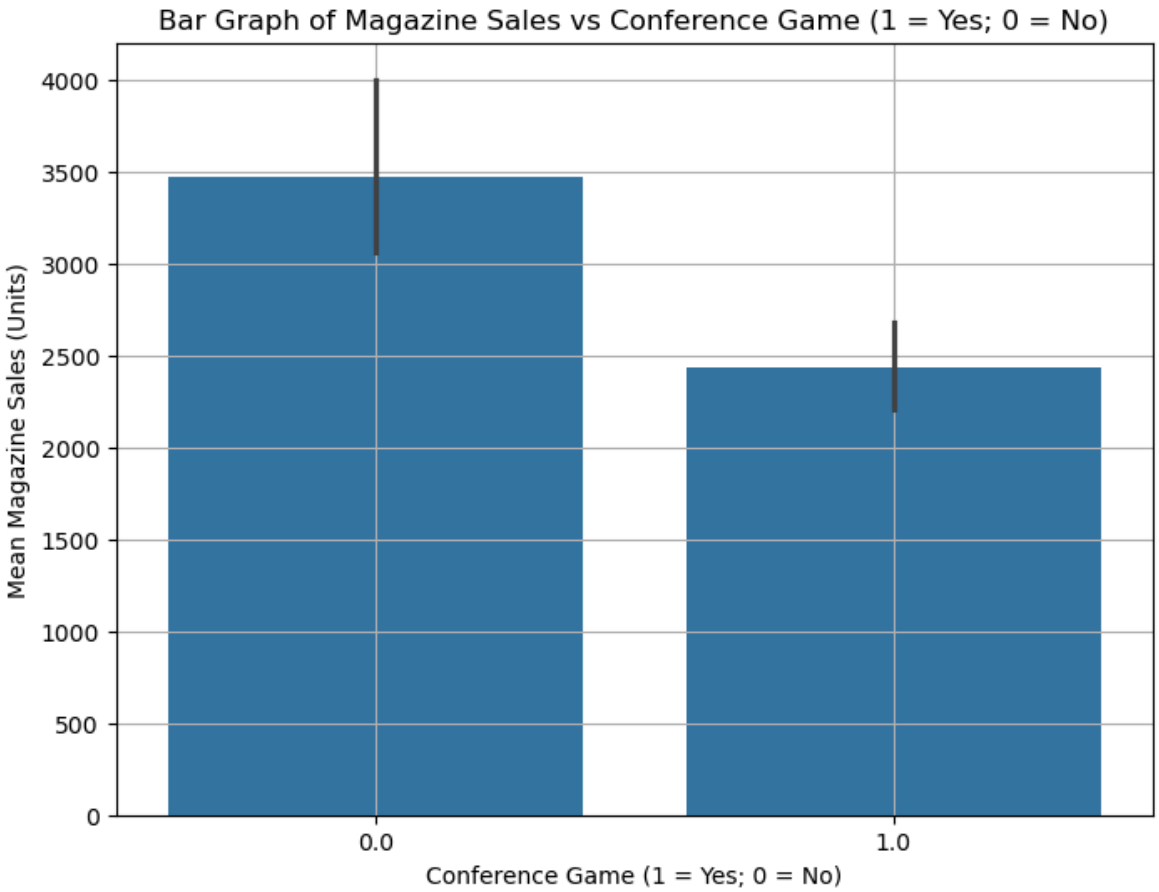


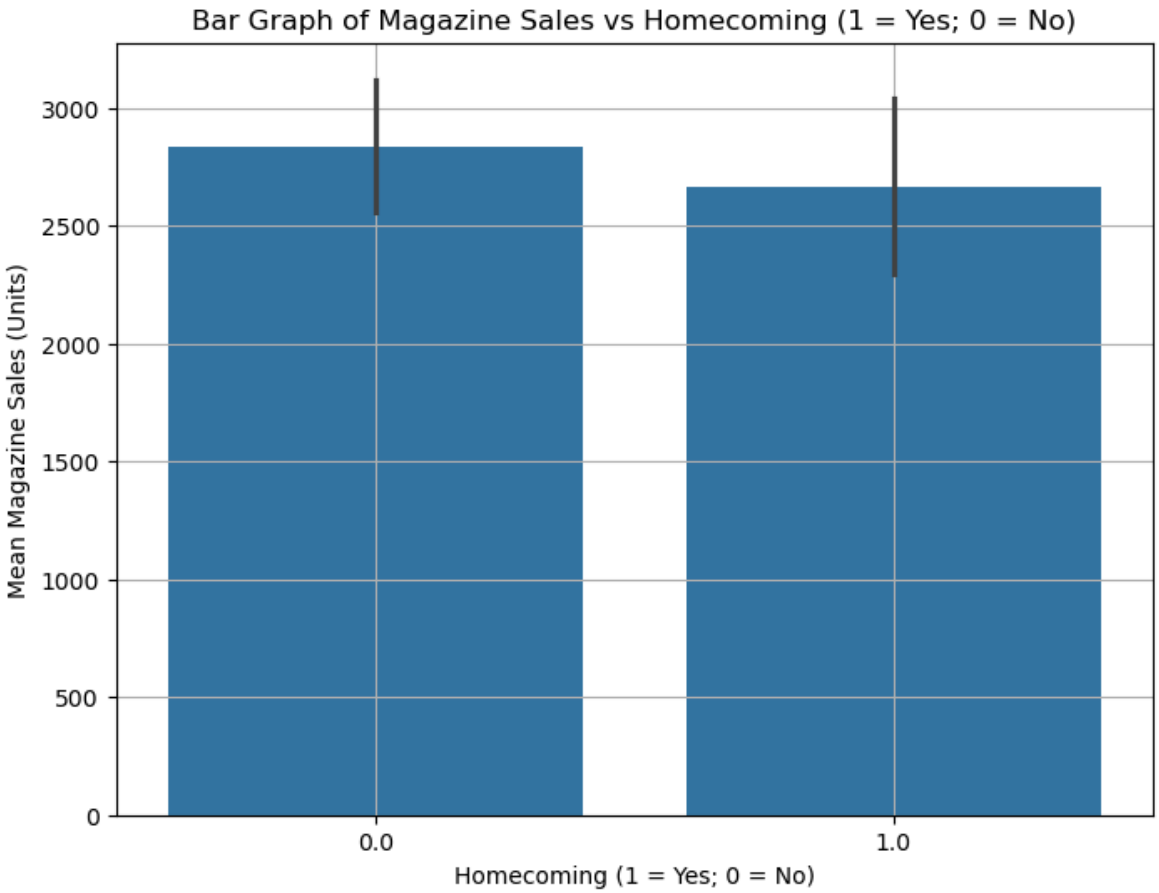
In [786…

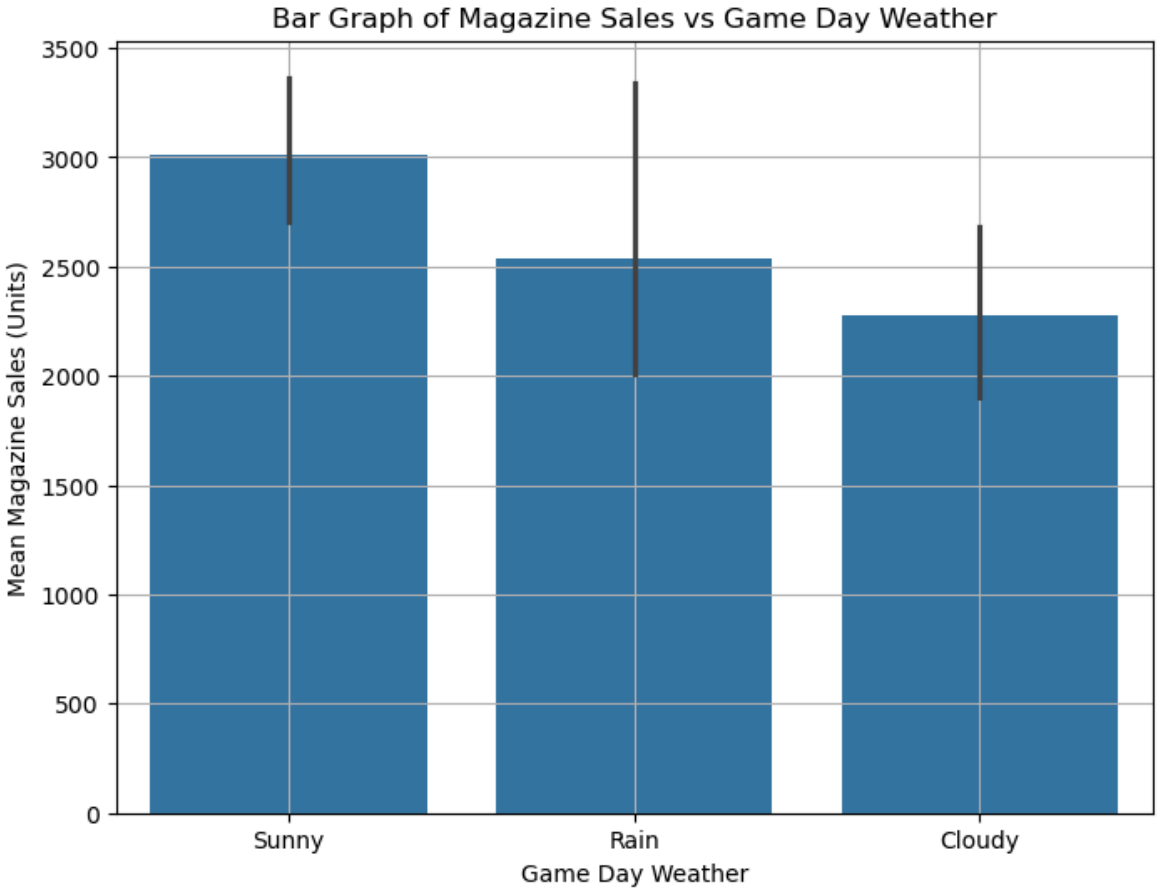
```
qualitative_columns = [
    'Throwback Jersey (1 = Yes; 0 = No)', 'Conference Game (1 = Yes; 0 = No)',
    'Homecoming (1 = Yes; 0 = No)', 'Game Day Weather' ,'CSU\'s Previous Season
]
```

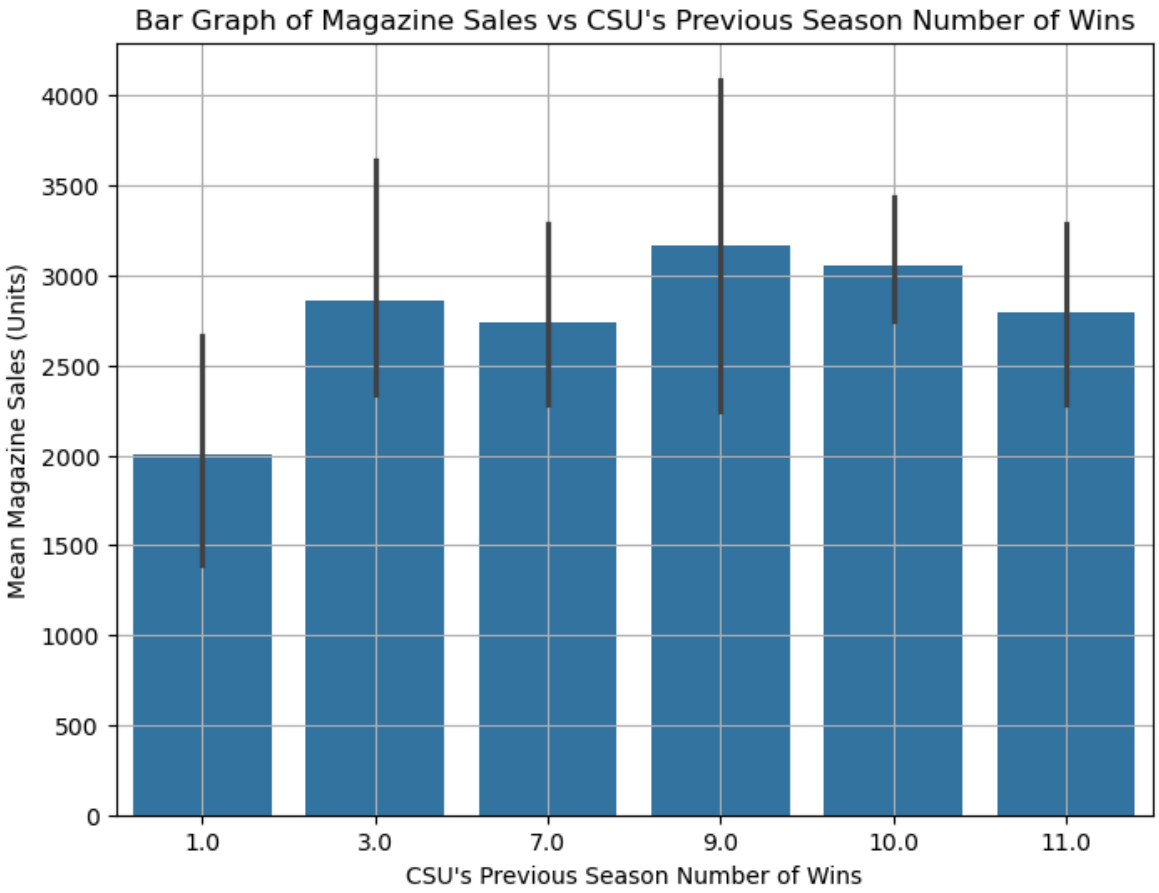
```
# Bar graphs
for col in qualitative_columns:
    plt.figure(figsize=(8, 6))
    sns.barplot(data=df, x=col, y='Magazine Sales (Units)', estimator='mean')
    plt.title(f'Bar Graph of Magazine Sales vs {col}')
    plt.xlabel(col)
    plt.ylabel('Mean Magazine Sales (Units)')
    plt.grid(True)
    plt.show()
```

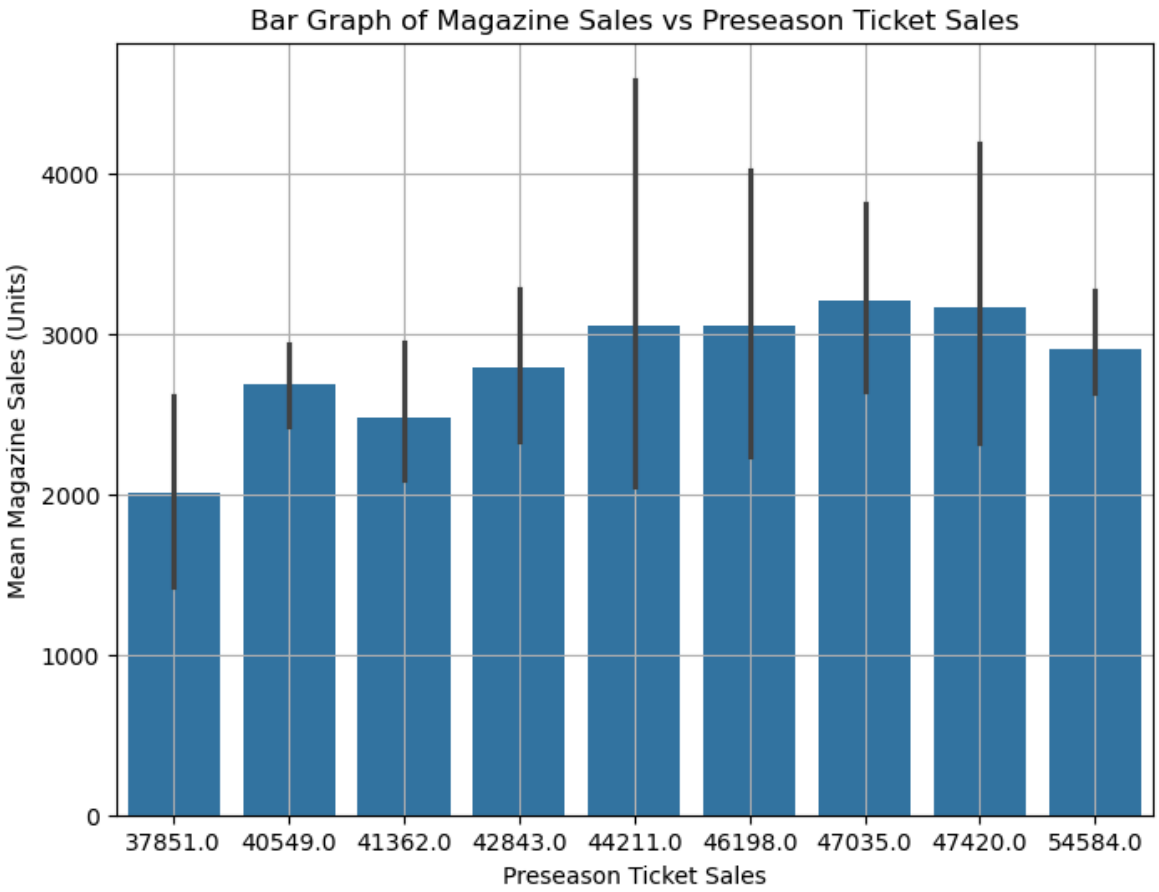


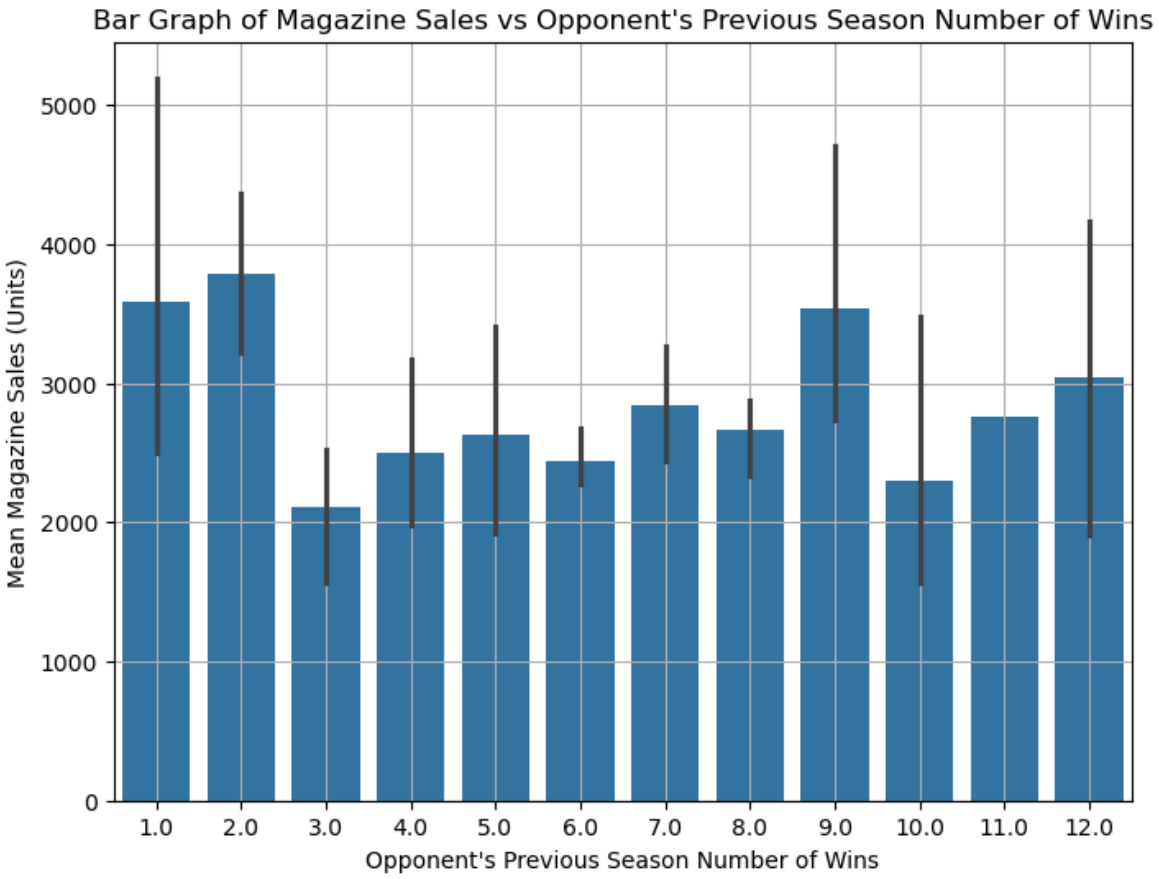

Bar Graph of Magazine Sales vs Conference Game (1 = Yes; 0 = No)
Mean Magazine Sales (Units)
0
500
1000
1500
2000
2500
3000
3500
4000
0.0
1.0
Conference Game (1 = Yes; 0 = No)

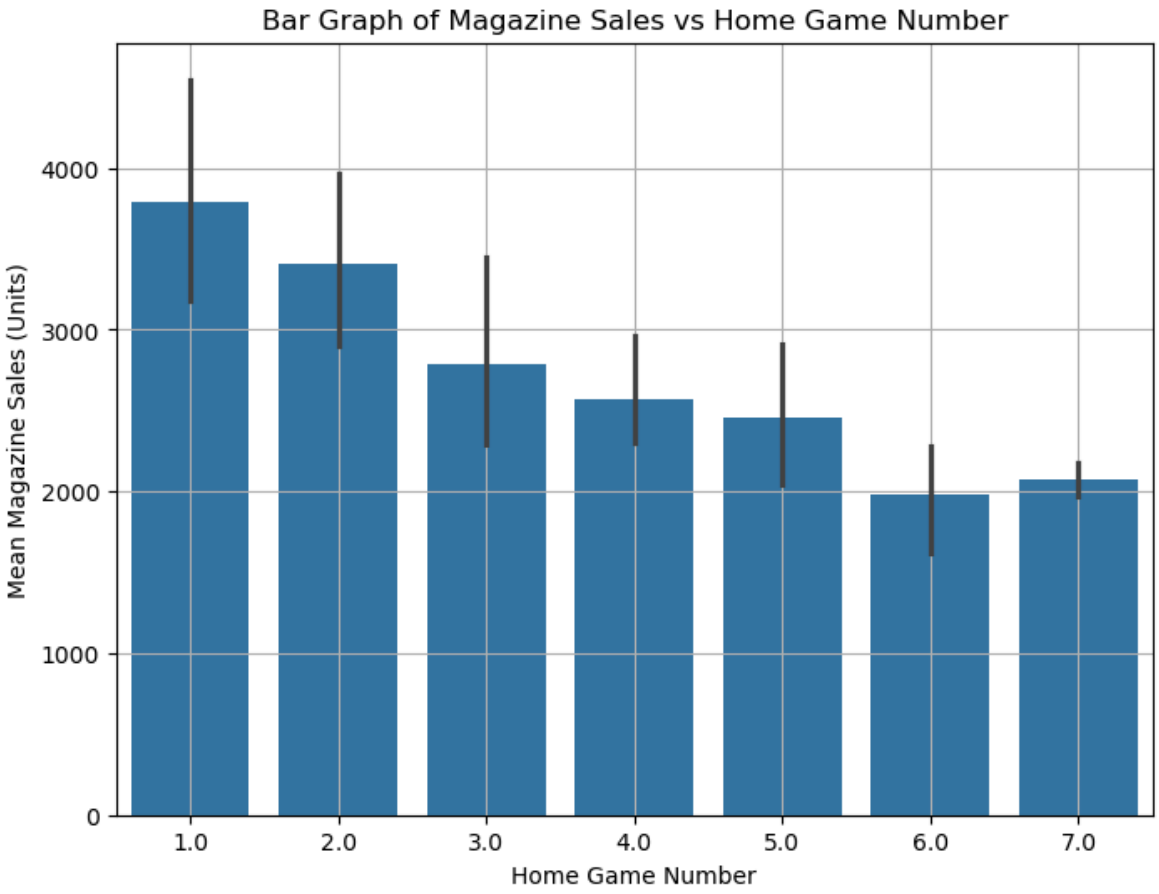

Bar Graph of Magazine Sales vs Homecoming (1 = Yes; 0 = No)
Mean Magazine Sales (Units)
0
500
1000
1500
2000
2500
3000
0.0
1.0
Homecoming (1 = Yes; 0 = No)


Bar Graph of Magazine Sales vs Game Day Weather
Mean Magazine Sales (Units)
3500
3000
2500
2000
1500
1000
500
0
Sunny
Rain
Cloudy
Game Day Weather


Bar Graph of Magazine Sales vs CSU's Previous Season Number of Wins
Mean Magazine Sales (Units)
4000
3500
3000
2500
2000
1500
1000
500
0
1.0
3.0
7.0
9.0
10.0
11.0
CSU's Previous Season Number of Wins

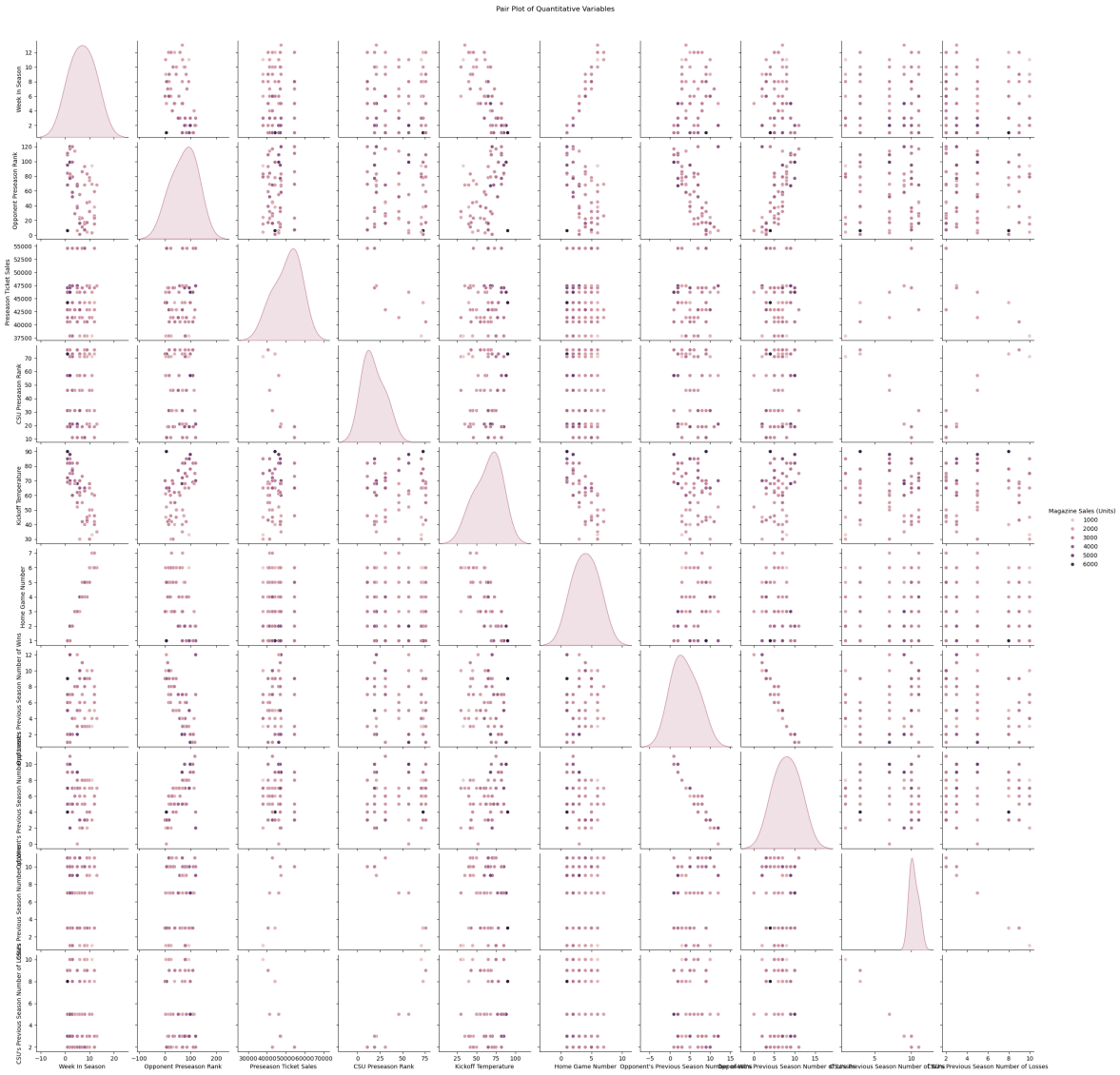

Bar Graph of Magazine Sales vs Preseason Ticket Sales
Mean Magazine Sales (Units)
0
1000
2000
3000
4000
37851.0
40549.0
41362.0
42843.0
44211.0
46198.0
47035.0
47420.0
54584.0
Preseason Ticket Sales


Bar Graph of Magazine Sales vs Opponent's Previous Season Number of Wins
Mean Magazine Sales (Units)
0
1000
2000
3000
4000
5000
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
11.0
12.0
Opponent's Previous Season Number of Wins



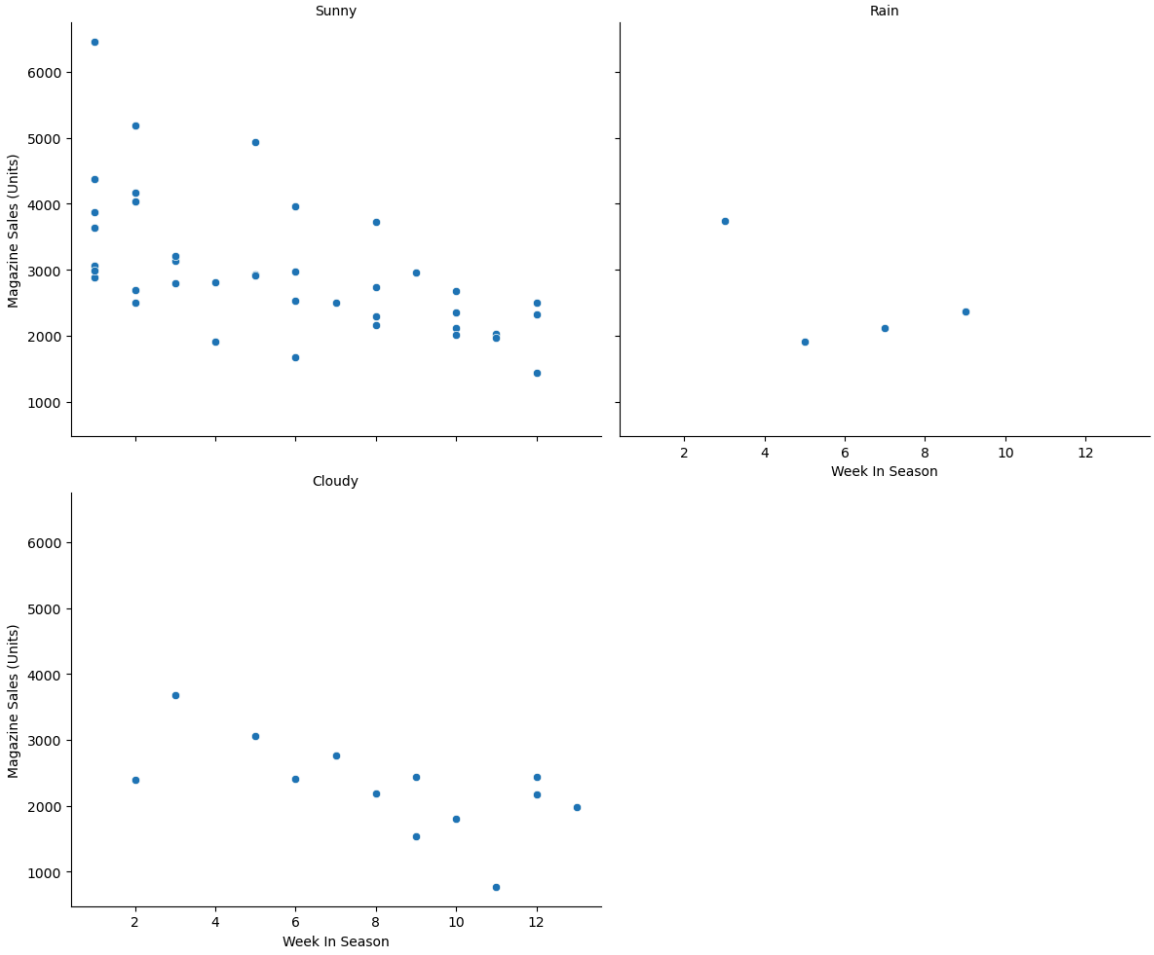
In [789…

```
# List of quantitative columns
quantitative_columns = [
    'Week In Season', 'Opponent Preseason Rank', 'Preseason Ticket Sales',
    'CSU Preseason Rank', 'Kickoff Temperature', 'Home Game Number',
    'Opponent\'s Previous Season Number of Wins', 'Opponent\'s Previous Season N
    'CSU\'s Previous Season Number of Wins', 'CSU\'s Previous Season Number of L
]

# Pair plot
pair_data = df[quantitative_columns + ['Magazine Sales (Units)']]
sns.pairplot(pair_data, diag_kind='kde', hue='Magazine Sales (Units)')
plt.suptitle('Pair Plot of Quantitative Variables', y=1.02)
plt.show()
```



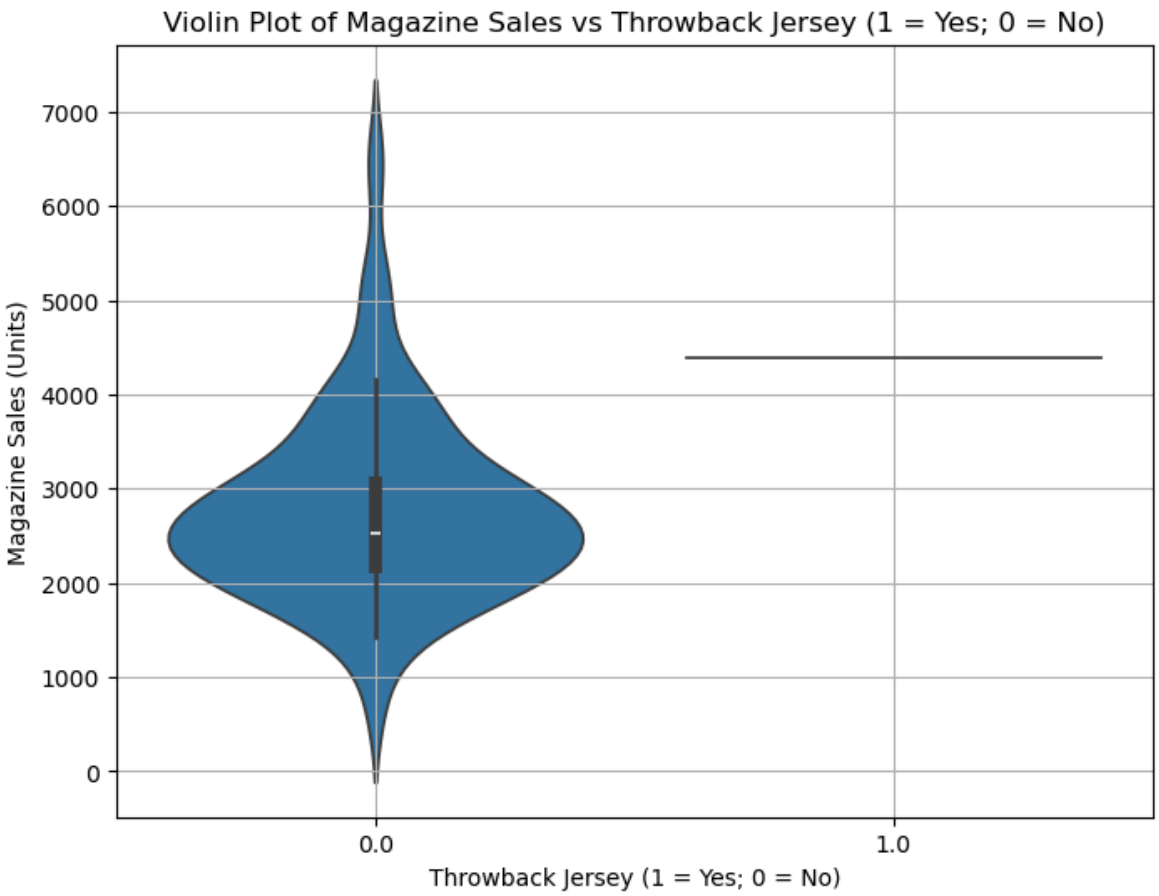
In [791…

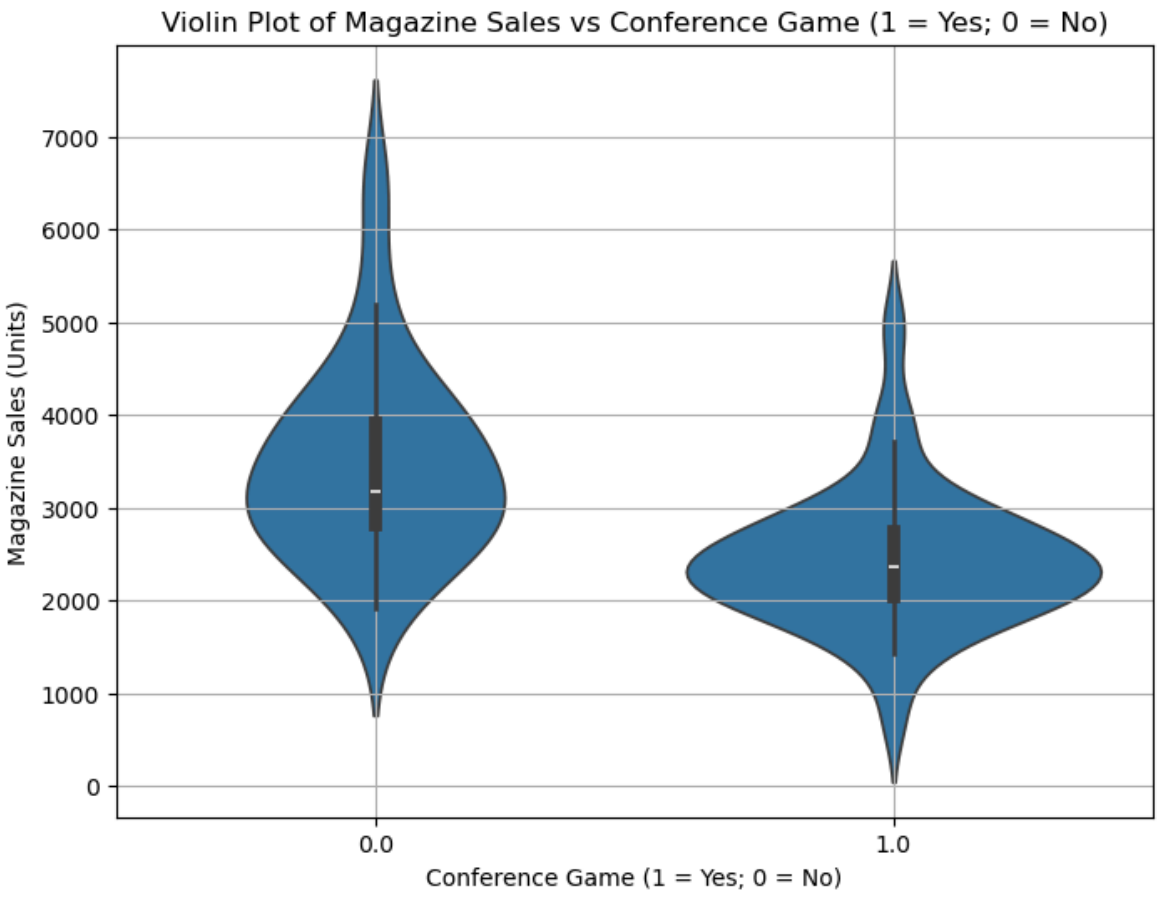
```
# Define the qualitative columns and create facets
qualitative_columns = [
    'Throwback Jersey (1 = Yes; 0 = No)', 'Conference Game (1 = Yes; 0 = No)',
    'Homecoming (1 = Yes; 0 = No)', 'Game Day Weather'
]

# Create a facet grid
g = sns.FacetGrid(df, col='Game Day Weather', col_wrap=2, height=5, aspect=1.2)
g.map(sns.scatterplot, 'Week In Season', 'Magazine Sales (Units)')
g.set_axis_labels('Week In Season', 'Magazine Sales (Units)')
g.set_titles(col_template="{col_name}")
g.add_legend()
plt.show()
```

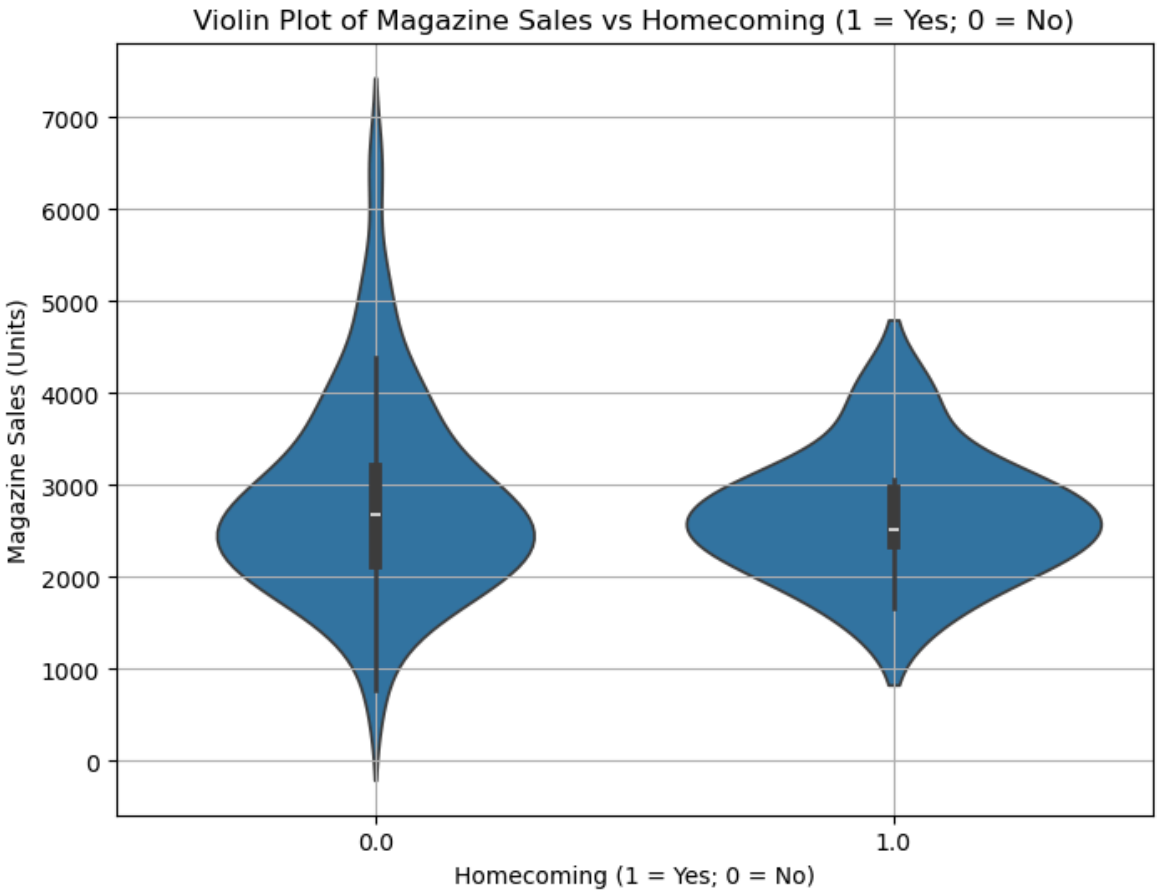


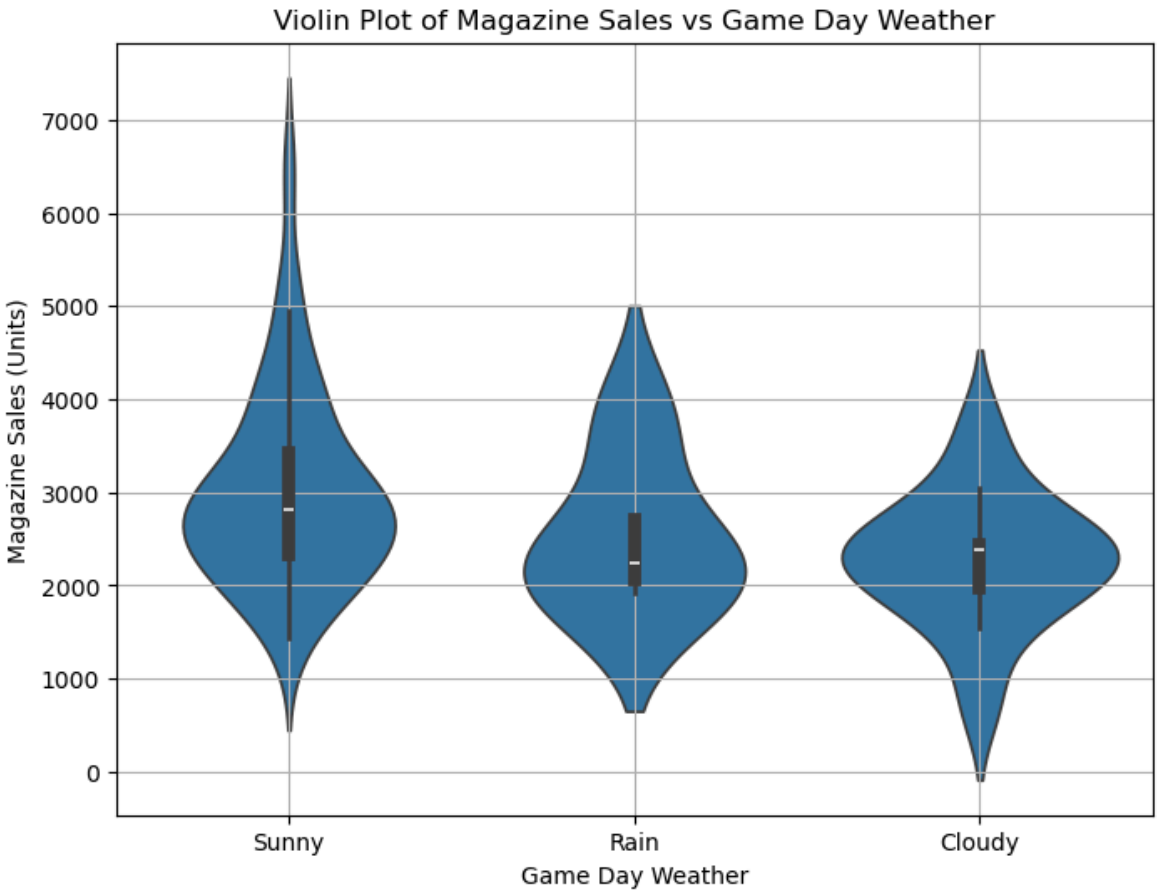
In [793…

```
# List of qualitative columns
qualitative_columns = [
    'Throwback Jersey (1 = Yes; 0 = No)', 'Conference Game (1 = Yes; 0 = No)',
    'Homecoming (1 = Yes; 0 = No)', 'Game Day Weather'
]

# Violin plots
for col in qualitative_columns:
    plt.figure(figsize=(8, 6))
    sns.violinplot(data=df, x=col, y='Magazine Sales (Units)')
    plt.title(f'Violin Plot of Magazine Sales vs {col}')
    plt.xlabel(col)
    plt.ylabel('Magazine Sales (Units)')
    plt.grid(True)
    plt.show()
```


Violin Plot of Magazine Sales vs Throwback Jersey (1 = Yes; 0 = No)
Magazine Sales (Units)
7000
6000
5000
4000
3000
2000
1000
0
0.0
1.0
Throwback Jersey (1 = Yes; 0 = No)


Violin Plot of Magazine Sales vs Conference Game (1 = Yes; 0 = No)
Magazine Sales (Units)
7000
6000
5000
4000
3000
2000
1000
0
0.0
1.0
Conference Game (1 = Yes; 0 = No)





In [700…

```
df_encoded = pd.get_dummies(df, columns=['Opponent', 'Game Day Weather'])
df_encoded
```

Out[700...

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
0	1.0	4165.0	2.0	120.0	47420.0	21.0	0.0	1.0	
1	2.0	3746.0	3.0	58.0	47420.0	21.0	0.0	1.0	
2	3.0	4943.0	5.0	67.0	47420.0	21.0	0.0	1.0	
3	4.0	2366.0	9.0	83.0	47420.0	21.0	0.0	1.0	
4	5.0	1796.0	10.0	74.0	47420.0	21.0	0.0	1.0	
5	6.0	1979.0	13.0	68.0	47420.0	21.0	0.0	1.0	
6	7.0	3866.0	1.0	109.0	46198.0	57.0	0.0	2.0	
7	8.0	5194.0	2.0	99.0	46198.0	57.0	0.0	2.0	
8	9.0	1909.0	5.0	6.0	46198.0	57.0	0.0	2.0	
9	10.0	2523.0	6.0	55.0	46198.0	57.0	0.0	2.0	
10	11.0	2734.0	8.0	17.0	46198.0	57.0	0.0	2.0	
11	12.0	2034.0	11.0	3.0	46198.0	57.0	0.0	2.0	
12	13.0	6463.0	1.0	6.0	44211.0	73.0	0.0	3.0	
13	14.0	3128.0	3.0	99.0	44211.0	73.0	0.0	3.0	
14	15.0	2972.0	6.0	1.0	44211.0	73.0	0.0	3.0	
15	16.0	2158.0	8.0	68.0	44211.0	73.0	0.0	3.0	
16	17.0	2120.0	10.0	78.0	44211.0	73.0	0.0	3.0	
17	18.0	1434.0	12.0	38.0	44211.0	73.0	0.0	3.0	
18	19.0	2691.0	2.0	83.0	37851.0	71.0	0.0	4.0	
19	20.0	3202.0	3.0	79.0	37851.0	71.0	0.0	4.0	
20	21.0	1669.0	6.0	24.0	37851.0	71.0	0.0	4.0	
21	22.0	2194.0	8.0	15.0	37851.0	71.0	0.0	4.0	
22	23.0	1536.0	9.0	4.0	37851.0	71.0	0.0	4.0	
23	24.0	763.0	11.0	94.0	37851.0	71.0	0.0	4.0	
24	25.0	3059.0	1.0	109.0	40549.0	76.0	0.0	5.0	
25	26.0	2390.0	2.0	79.0	40549.0	76.0	0.0	5.0	
26	27.0	3056.0	5.0	39.0	40549.0	76.0	0.0	5.0	
27	28.0	2298.0	8.0	93.0	40549.0	76.0	0.0	5.0	
28	29.0	2956.0	9.0	17.0	40549.0	76.0	0.0	5.0	
29	30.0	2324.0	12.0	59.0	40549.0	76.0	0.0	5.0	
30	31.0	2885.0	1.0	84.0	41362.0	46.0	0.0	6.0	

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
31	32.0	3677.0	3.0	52.0	41362.0	46.0	0.0	6.0	
32	33.0	1911.0	4.0	114.0	41362.0	46.0	0.0	6.0	
33	34.0	2404.0	6.0	22.0	41362.0	46.0	0.0	6.0	
34	35.0	2113.0	7.0	23.0	41362.0	46.0	0.0	6.0	
35	36.0	2345.0	10.0	32.0	41362.0	46.0	0.0	6.0	
36	37.0	1969.0	11.0	69.0	41362.0	46.0	0.0	6.0	
37	38.0	3634.0	1.0	68.0	42843.0	31.0	0.0	7.0	
38	39.0	2500.0	2.0	117.0	42843.0	31.0	0.0	7.0	
39	40.0	2810.0	4.0	29.0	42843.0	31.0	0.0	7.0	
40	41.0	3961.0	6.0	16.0	42843.0	31.0	0.0	7.0	
41	42.0	2440.0	9.0	44.0	42843.0	31.0	0.0	7.0	
42	43.0	2017.0	10.0	45.0	42843.0	31.0	0.0	7.0	
43	44.0	2173.0	12.0	26.0	42843.0	31.0	0.0	7.0	
44	45.0	4382.0	1.0	95.0	47035.0	19.0	1.0	8.0	
45	46.0	4037.0	2.0	77.0	47035.0	19.0	0.0	8.0	
46	47.0	2930.0	5.0	37.0	47035.0	19.0	0.0	8.0	
47	48.0	2757.0	7.0	11.0	47035.0	19.0	0.0	8.0	
48	49.0	2678.0	10.0	32.0	47035.0	19.0	0.0	8.0	
49	50.0	2445.0	12.0	15.0	47035.0	19.0	0.0	8.0	
50	51.0	2985.0	1.0	111.0	54584.0	19.0	0.0	9.0	
51	52.0	2800.0	3.0	120.0	54584.0	11.0	0.0	9.0	
52	53.0	2910.0	5.0	69.0	54584.0	11.0	0.0	9.0	
53	54.0	2500.0	7.0	86.0	54584.0	11.0	0.0	9.0	
54	55.0	3721.0	8.0	7.0	54584.0	11.0	0.0	9.0	
55	56.0	2500.0	12.0	23.0	54584.0	11.0	0.0	9.0	

56 rows × 43 columns

In [704…
```
# df_encoded['Preseason Ticket Sales'] = df_encoded['Preseason Ticket Sales'].st
# df_encoded
```

In [529…
```
df_encoded['Preseason Ticket Sales'] = df_encoded['Preseason Ticket Sales'].asty
df_encoded
```

Out[529...

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
0	1.0	4165.0	2.0	120.0	47420.0	21.0	0.0	1.0	
1	2.0	3746.0	3.0	58.0	47420.0	21.0	0.0	1.0	
2	3.0	4943.0	5.0	67.0	47420.0	21.0	0.0	1.0	
3	4.0	2366.0	9.0	83.0	47420.0	21.0	0.0	1.0	
4	5.0	1796.0	10.0	74.0	47420.0	21.0	0.0	1.0	
5	6.0	1979.0	13.0	68.0	47420.0	21.0	0.0	1.0	
6	7.0	3866.0	1.0	109.0	46198.0	57.0	0.0	2.0	
7	8.0	5194.0	2.0	99.0	46198.0	57.0	0.0	2.0	
8	9.0	1909.0	5.0	6.0	46198.0	57.0	0.0	2.0	
9	10.0	2523.0	6.0	55.0	46198.0	57.0	0.0	2.0	
10	11.0	2734.0	8.0	17.0	46198.0	57.0	0.0	2.0	
11	12.0	2034.0	11.0	3.0	46198.0	57.0	0.0	2.0	
12	13.0	6463.0	1.0	6.0	44211.0	73.0	0.0	3.0	
13	14.0	3128.0	3.0	99.0	44211.0	73.0	0.0	3.0	
14	15.0	2972.0	6.0	1.0	44211.0	73.0	0.0	3.0	
15	16.0	2158.0	8.0	68.0	44211.0	73.0	0.0	3.0	
16	17.0	2120.0	10.0	78.0	44211.0	73.0	0.0	3.0	
17	18.0	1434.0	12.0	38.0	44211.0	73.0	0.0	3.0	
18	19.0	2691.0	2.0	83.0	37851.0	71.0	0.0	4.0	
19	20.0	3202.0	3.0	79.0	37851.0	71.0	0.0	4.0	
20	21.0	1669.0	6.0	24.0	37851.0	71.0	0.0	4.0	
21	22.0	2194.0	8.0	15.0	37851.0	71.0	0.0	4.0	
22	23.0	1536.0	9.0	4.0	37851.0	71.0	0.0	4.0	
23	24.0	763.0	11.0	94.0	37851.0	71.0	0.0	4.0	
24	25.0	3059.0	1.0	109.0	40549.0	76.0	0.0	5.0	
25	26.0	2390.0	2.0	79.0	40549.0	76.0	0.0	5.0	
26	27.0	3056.0	5.0	39.0	40549.0	76.0	0.0	5.0	
27	28.0	2298.0	8.0	93.0	40549.0	76.0	0.0	5.0	
28	29.0	2956.0	9.0	17.0	40549.0	76.0	0.0	5.0	
29	30.0	2324.0	12.0	59.0	40549.0	76.0	0.0	5.0	
30	31.0	2885.0	1.0	84.0	41362.0	46.0	0.0	6.0	

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
31	32.0	3677.0	3.0	52.0	41362.0	46.0	0.0	6.0	
32	33.0	1911.0	4.0	114.0	41362.0	46.0	0.0	6.0	
33	34.0	2404.0	6.0	22.0	41362.0	46.0	0.0	6.0	
34	35.0	2113.0	7.0	23.0	41362.0	46.0	0.0	6.0	
35	36.0	2345.0	10.0	32.0	41362.0	46.0	0.0	6.0	
36	37.0	1969.0	11.0	69.0	41362.0	46.0	0.0	6.0	
37	38.0	3634.0	1.0	68.0	42843.0	31.0	0.0	7.0	
38	39.0	2500.0	2.0	117.0	42843.0	31.0	0.0	7.0	
39	40.0	2810.0	4.0	29.0	42843.0	31.0	0.0	7.0	
40	41.0	3961.0	6.0	16.0	42843.0	31.0	0.0	7.0	
41	42.0	2440.0	9.0	44.0	42843.0	31.0	0.0	7.0	
42	43.0	2017.0	10.0	45.0	42843.0	31.0	0.0	7.0	
43	44.0	2173.0	12.0	26.0	42843.0	31.0	0.0	7.0	
44	45.0	4382.0	1.0	95.0	47035.0	19.0	1.0	8.0	
45	46.0	4037.0	2.0	77.0	47035.0	19.0	0.0	8.0	
46	47.0	2930.0	5.0	37.0	47035.0	19.0	0.0	8.0	
47	48.0	2757.0	7.0	11.0	47035.0	19.0	0.0	8.0	
48	49.0	2678.0	10.0	32.0	47035.0	19.0	0.0	8.0	
49	50.0	2445.0	12.0	15.0	47035.0	19.0	0.0	8.0	
50	51.0	2985.0	1.0	111.0	54584.0	19.0	0.0	9.0	
51	52.0	2800.0	3.0	120.0	54584.0	11.0	0.0	9.0	
52	53.0	2910.0	5.0	69.0	54584.0	11.0	0.0	9.0	
53	54.0	2500.0	7.0	86.0	54584.0	11.0	0.0	9.0	
54	55.0	3721.0	8.0	7.0	54584.0	11.0	0.0	9.0	
55	56.0	2500.0	12.0	23.0	54584.0	11.0	0.0	9.0	

56 rows × 43 columns

In [706…

```
df_encoded
```

Out[706...

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
0	1.0	4165.0	2.0	120.0	47420.0	21.0	0.0	1.0	
1	2.0	3746.0	3.0	58.0	47420.0	21.0	0.0	1.0	
2	3.0	4943.0	5.0	67.0	47420.0	21.0	0.0	1.0	
3	4.0	2366.0	9.0	83.0	47420.0	21.0	0.0	1.0	
4	5.0	1796.0	10.0	74.0	47420.0	21.0	0.0	1.0	
5	6.0	1979.0	13.0	68.0	47420.0	21.0	0.0	1.0	
6	7.0	3866.0	1.0	109.0	46198.0	57.0	0.0	2.0	
7	8.0	5194.0	2.0	99.0	46198.0	57.0	0.0	2.0	
8	9.0	1909.0	5.0	6.0	46198.0	57.0	0.0	2.0	
9	10.0	2523.0	6.0	55.0	46198.0	57.0	0.0	2.0	
10	11.0	2734.0	8.0	17.0	46198.0	57.0	0.0	2.0	
11	12.0	2034.0	11.0	3.0	46198.0	57.0	0.0	2.0	
12	13.0	6463.0	1.0	6.0	44211.0	73.0	0.0	3.0	
13	14.0	3128.0	3.0	99.0	44211.0	73.0	0.0	3.0	
14	15.0	2972.0	6.0	1.0	44211.0	73.0	0.0	3.0	
15	16.0	2158.0	8.0	68.0	44211.0	73.0	0.0	3.0	
16	17.0	2120.0	10.0	78.0	44211.0	73.0	0.0	3.0	
17	18.0	1434.0	12.0	38.0	44211.0	73.0	0.0	3.0	
18	19.0	2691.0	2.0	83.0	37851.0	71.0	0.0	4.0	
19	20.0	3202.0	3.0	79.0	37851.0	71.0	0.0	4.0	
20	21.0	1669.0	6.0	24.0	37851.0	71.0	0.0	4.0	
21	22.0	2194.0	8.0	15.0	37851.0	71.0	0.0	4.0	
22	23.0	1536.0	9.0	4.0	37851.0	71.0	0.0	4.0	
23	24.0	763.0	11.0	94.0	37851.0	71.0	0.0	4.0	
24	25.0	3059.0	1.0	109.0	40549.0	76.0	0.0	5.0	
25	26.0	2390.0	2.0	79.0	40549.0	76.0	0.0	5.0	
26	27.0	3056.0	5.0	39.0	40549.0	76.0	0.0	5.0	
27	28.0	2298.0	8.0	93.0	40549.0	76.0	0.0	5.0	
28	29.0	2956.0	9.0	17.0	40549.0	76.0	0.0	5.0	
29	30.0	2324.0	12.0	59.0	40549.0	76.0	0.0	5.0	
30	31.0	2885.0	1.0	84.0	41362.0	46.0	0.0	6.0	

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
31	32.0	3677.0	3.0	52.0	41362.0	46.0	0.0	6.0	
32	33.0	1911.0	4.0	114.0	41362.0	46.0	0.0	6.0	
33	34.0	2404.0	6.0	22.0	41362.0	46.0	0.0	6.0	
34	35.0	2113.0	7.0	23.0	41362.0	46.0	0.0	6.0	
35	36.0	2345.0	10.0	32.0	41362.0	46.0	0.0	6.0	
36	37.0	1969.0	11.0	69.0	41362.0	46.0	0.0	6.0	
37	38.0	3634.0	1.0	68.0	42843.0	31.0	0.0	7.0	
38	39.0	2500.0	2.0	117.0	42843.0	31.0	0.0	7.0	
39	40.0	2810.0	4.0	29.0	42843.0	31.0	0.0	7.0	
40	41.0	3961.0	6.0	16.0	42843.0	31.0	0.0	7.0	
41	42.0	2440.0	9.0	44.0	42843.0	31.0	0.0	7.0	
42	43.0	2017.0	10.0	45.0	42843.0	31.0	0.0	7.0	
43	44.0	2173.0	12.0	26.0	42843.0	31.0	0.0	7.0	
44	45.0	4382.0	1.0	95.0	47035.0	19.0	1.0	8.0	
45	46.0	4037.0	2.0	77.0	47035.0	19.0	0.0	8.0	
46	47.0	2930.0	5.0	37.0	47035.0	19.0	0.0	8.0	
47	48.0	2757.0	7.0	11.0	47035.0	19.0	0.0	8.0	
48	49.0	2678.0	10.0	32.0	47035.0	19.0	0.0	8.0	
49	50.0	2445.0	12.0	15.0	47035.0	19.0	0.0	8.0	
50	51.0	2985.0	1.0	111.0	54584.0	19.0	0.0	9.0	
51	52.0	2800.0	3.0	120.0	54584.0	11.0	0.0	9.0	
52	53.0	2910.0	5.0	69.0	54584.0	11.0	0.0	9.0	
53	54.0	2500.0	7.0	86.0	54584.0	11.0	0.0	9.0	
54	55.0	3721.0	8.0	7.0	54584.0	11.0	0.0	9.0	
55	56.0	2500.0	12.0	23.0	54584.0	11.0	0.0	9.0	

56 rows × 43 columns

In [87]:
```
print(df_encoded.describe())
```

	Index	Magazine Sales (Units)	Week In Season
count	56.000000	56.000000	56.000000
mean	28.500000	2806.285714	6.250000
std	16.309506	989.430501	3.738011
min	1.000000	763.000000	1.000000
25%	14.750000	2169.250000	3.000000
50%	28.500000	2600.500000	6.000000
75%	42.250000	3076.250000	9.250000
max	56.000000	6463.000000	13.000000

	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank
count	56.000000	56.000000	56.000000
mean	56.571429	44580.767857	44.910714
std	36.344242	4601.066252	23.639933
min	1.000000	37851.000000	11.000000
25%	23.000000	41362.000000	21.000000
50%	58.500000	44211.000000	46.000000
75%	83.250000	47035.000000	71.000000
max	120.000000	54584.000000	76.000000

	Throwback Jersey (1 = Yes; 0 = No)	Year	Kickoff Temperature
count	56.000000	56.000000	56.000000
mean	0.017857	5.053571	61.750000
std	0.133631	2.575547	16.367651
min	0.000000	1.000000	30.000000
25%	0.000000	3.000000	46.000000
50%	0.000000	5.000000	65.000000
75%	0.000000	7.000000	73.500000
max	1.000000	9.000000	90.000000

	Home Game Number	Conference Game (1 = Yes; 0 = No)
count	56.000000	56.000000
mean	3.625000	0.642857
std	1.814713	0.483494
min	1.000000	0.000000
25%	2.000000	0.000000
50%	4.000000	1.000000
75%	5.000000	1.000000
max	7.000000	1.000000

	Homecoming (1 = Yes; 0 = No)
count	56.000000
mean	0.160714
std	0.370591
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

	Opponent's Previous Season Number of Wins
count	56.000000
mean	5.857143
std	2.969105
min	1.000000
25%	3.000000
50%	6.000000
75%	8.000000
max	12.000000

```
       Opponent's Previous Season Number of Losses  \
count                                     56.000000
mean                                       5.982143
std                                        2.408252
min                                        0.000000
25%                                        4.750000
50%                                        6.000000
75%                                        8.000000
max                                       11.000000

       CSU's Previous Season Number of Wins  \
count                             56.000000
mean                               6.857143
std                                3.440024
min                                1.000000
25%                                3.000000
50%                                7.000000
75%                               10.000000
max                               11.000000

       CSU's Previous Season Number of Losses
count                              56.000000
mean                                5.160714
std                                 2.903144
min                                 2.000000
25%                                 3.000000
50%                                 5.000000
75%                                 8.000000
max                                10.000000
```

In [574…

```python
## TO TRAIN ON KICK OFF TEMPERATURE
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Features (X) and Target (y)
X = df_encoded[['Week In Season','Year']]
y = df_encoded['Kickoff Temperature']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Initialize and train the model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared Score: {r2:.2f}")

# If you want to see the coefficients
```

```
coefficients = model.coef_
intercept = model.intercept_

print(f"Coefficients: {coefficients}")
print(f"Intercept: {intercept}")
```

```
Mean Squared Error: 63.99
R-squared Score: 0.55
Coefficients: [-3.50140477  1.10975665]
Intercept: 77.62047320171311
```

In [843…

```
##TO TRAIN ON MAGAZINE SALES
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Features (X) and Target (y)
X = df_encoded[['Week In Season','Home Game Number']]
y = df_encoded['Magazine Sales (Units)']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Initialize and train the model
mo = LinearRegression()
mo.fit(X_train, y_train)

# Make predictions
y_pred = mo.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared Score: {r2:.2f}")

# If you want to see the coefficients
coefficients = mo.coef_
intercept = mo.intercept_

print(f"Coefficients: {coefficients}")
print(f"Intercept: {intercept}")
```

```
Mean Squared Error: 685572.25
R-squared Score: 0.21
Coefficients: [-122.98468407  -97.60485746]
Intercept: 3896.198880468626
```

In [646…

```
#######ENDDDDDDDDDDD

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

In [648…

```
df_new=pd.read_csv("Casestudy2.csv")
```

In [650…
```
df_new= df_new.dropna(axis=1, how='all')
df_new= df_new.dropna(axis=0, how='all')
df_new
```

Out[650…

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
0	1.0	Cedar Falls University	4165.0	2.0	120.0	47,420	21.0	0
1	2.0	Oklahoma A&M	3746.0	3.0	58.0	47,420	21.0	0
2	3.0	Urbana College	4943.0	5.0	67.0	47,420	21.0	0
3	4.0	University of Bloomington	2366.0	9.0	83.0	47,420	21.0	0
4	5.0	Indiana A&M	1796.0	10.0	74.0	47,420	21.0	0
…	…	…	…	…	…	…	…	
58	59.0	Columbus University	NaN	5.0	4.0	54,584	16.0	0
59	60.0	Indiana A&M	NaN	6.0	30.0	54,584	16.0	0
60	61.0	DeKalb College	NaN	9.0	56.0	54,584	16.0	0
61	62.0	Evanston University	NaN	10.0	65.0	54,584	16.0	0
62	63.0	Madison University	NaN	11.0	47.0	54,584	16.0	0

63 rows × 18 columns

In [652…
```
df_new = pd.get_dummies(df_new, columns=['Opponent', 'Game Day Weather'])
df_new
```

Out[652...

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
0	1.0	4165.0	2.0	120.0	47,420	21.0	0.0	1.0	
1	2.0	3746.0	3.0	58.0	47,420	21.0	0.0	1.0	
2	3.0	4943.0	5.0	67.0	47,420	21.0	0.0	1.0	
3	4.0	2366.0	9.0	83.0	47,420	21.0	0.0	1.0	
4	5.0	1796.0	10.0	74.0	47,420	21.0	0.0	1.0	
...	...	...	...	...	...	...	...	...	
58	59.0	NaN	5.0	4.0	54,584	16.0	0.0	10.0	
59	60.0	NaN	6.0	30.0	54,584	16.0	0.0	10.0	
60	61.0	NaN	9.0	56.0	54,584	16.0	0.0	10.0	
61	62.0	NaN	10.0	65.0	54,584	16.0	0.0	10.0	
62	63.0	NaN	11.0	47.0	54,584	16.0	0.0	10.0	

63 rows × 44 columns

In [654...

```
df_new['Preseason Ticket Sales'] = df_new['Preseason Ticket Sales'].str.replace(
df_new
```

Out[654...

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
0	1.0	4165.0	2.0	120.0	47420	21.0	0.0	1.0	
1	2.0	3746.0	3.0	58.0	47420	21.0	0.0	1.0	
2	3.0	4943.0	5.0	67.0	47420	21.0	0.0	1.0	
3	4.0	2366.0	9.0	83.0	47420	21.0	0.0	1.0	
4	5.0	1796.0	10.0	74.0	47420	21.0	0.0	1.0	
...	...	...	...	...	...	...	...	...	
58	59.0	NaN	5.0	4.0	54584	16.0	0.0	10.0	
59	60.0	NaN	6.0	30.0	54584	16.0	0.0	10.0	
60	61.0	NaN	9.0	56.0	54584	16.0	0.0	10.0	
61	62.0	NaN	10.0	65.0	54584	16.0	0.0	10.0	
62	63.0	NaN	11.0	47.0	54584	16.0	0.0	10.0	

63 rows × 44 columns

In [656...
```
df_new['Preseason Ticket Sales'] = df_new['Preseason Ticket Sales'].astype(float
```

In [555...
```
df_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 63 entries, 0 to 62
Data columns (total 44 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Index                                         63 non-null     float64
 1   Magazine Sales (Units)                        56 non-null     float64
 2   Week In Season                                63 non-null     float64
 3   Opponent Preseason Rank                       63 non-null     float64
 4   Preseason Ticket Sales                        63 non-null     float64
 5   CSU Preseason Rank                            63 non-null     float64
 6   Throwback Jersey (1 = Yes; 0 = No)            63 non-null     float64
 7   Year                                          63 non-null     float64
 8   Kickoff Temperature                           56 non-null     float64
 9   Home Game Number                              63 non-null     float64
 10  Conference Game (1 = Yes; 0 = No)             63 non-null     float64
 11  Homecoming (1 = Yes; 0 = No)                  63 non-null     float64
 12  Opponent's Previous Season Number of Wins     63 non-null     float64
 13  Opponent's Previous Season Number of Losses   63 non-null     float64
 14  CSU's Previous Season Number of Wins          63 non-null     float64
 15  CSU's Previous Season Number of Losses        63 non-null     float64
 16  Opponent_Ann Arbor University                 63 non-null     bool
 17  Opponent_Cedar Falls University               63 non-null     bool
 18  Opponent_Columbus University                  63 non-null     bool
 19  Opponent_DeKalb College                       63 non-null     bool
 20  Opponent_Evanston University                  63 non-null     bool
 21  Opponent_Indiana A&M                          63 non-null     bool
 22  Opponent_LBJ University                       63 non-null     bool
 23  Opponent_Letterman University                 63 non-null     bool
 24  Opponent_Lincoln University                   63 non-null     bool
 25  Opponent_Madison University                   63 non-null     bool
 26  Opponent_Michigan A&M                         63 non-null     bool
 27  Opponent_Minneapolis State University         63 non-null     bool
 28  Opponent_Mt Pleasant College                  63 non-null     bool
 29  Opponent_Northern Cincinnati University       63 non-null     bool
 30  Opponent_Ohio A&M                             63 non-null     bool
 31  Opponent_Oklahoma A&M                         63 non-null     bool
 32  Opponent_Pennsylvania A&M                     63 non-null     bool
 33  Opponent_University of Ames                   63 non-null     bool
 34  Opponent_University of Bloomington            63 non-null     bool
 35  Opponent_University of Kalamazoo              63 non-null     bool
 36  Opponent_University of Logan                  63 non-null     bool
 37  Opponent_University of Missoula               63 non-null     bool
 38  Opponent_University of Tempe                  63 non-null     bool
 39  Opponent_Urbana College                       63 non-null     bool
 40  Opponent_Western New York University          63 non-null     bool
 41  Game Day Weather_Cloudy                       63 non-null     bool
 42  Game Day Weather_Rain                         63 non-null     bool
 43  Game Day Weather_Sunny                        63 non-null     bool
dtypes: bool(28), float64(16)
memory usage: 10.1 KB
```

In [676…

```
df_new = df_new.tail(7)
df_new
```

Out[676...

	Index	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)	Year	Tem
56	57.0	NaN	1.0	120.0	54584.0	16.0	1.0	10.0	
57	58.0	NaN	3.0	46.0	54584.0	16.0	0.0	10.0	
58	59.0	NaN	5.0	4.0	54584.0	16.0	0.0	10.0	
59	60.0	NaN	6.0	30.0	54584.0	16.0	0.0	10.0	
60	61.0	NaN	9.0	56.0	54584.0	16.0	0.0	10.0	
61	62.0	NaN	10.0	65.0	54584.0	16.0	0.0	10.0	
62	63.0	NaN	11.0	47.0	54584.0	16.0	0.0	10.0	

7 rows × 44 columns

In [678...

```
##SELECTING VARIABLES TO PREDICT THE KICK OFF TEMPERATURE
df_temp = df_new[['Week In Season','Year']]
df_temp
```

Out[678...

	Week In Season	Year
56	1.0	10.0
57	3.0	10.0
58	5.0	10.0
59	6.0	10.0
60	9.0	10.0
61	10.0	10.0
62	11.0	10.0

In [662...

```
predicted_temp = model.predict(df_temp)
print(predicted_temp)
```

```
[85.21663497 78.21382543 71.2110159  67.70961113 57.20539682 53.70399206
 50.20258729]
```

In [845...

```
##SELECTING VARIABLES TO PREDICT THE MAGAZINE SALES
df_sales = df_new[['Week In Season','Home Game Number']]
df_sales
```

Out[845…

	Week In Season	Home Game Number
56	1.0	1.0
57	3.0	2.0
58	5.0	3.0
59	6.0	4.0
60	9.0	5.0
61	10.0	6.0
62	11.0	7.0

In [847…

```
predicted_sales = mo.predict(df_sales)
print(predicted_sales)
```

```
[3675.60933894 3332.03511334 2988.46088774 2767.87134621 2301.31243653
 2080.722895   1860.13335347]
```

In [766…

```
df_weather = pd.read_csv('Case_Study_3.csv')
df_weather
```

Out[766...

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
0	1	Cedar Falls University	4165	2	120	47,420	21	
1	2	Oklahoma A&M	3746	3	58	47,420	21	
2	3	Urbana College	4943	5	67	47,420	21	
3	4	University of Bloomington	2366	9	83	47,420	21	
4	5	Indiana A&M	1796	10	74	47,420	21	
...	...	...	...	...	...	...	...	
58	59	Columbus University	3397	5	4	54,584	16	
59	60	Indiana A&M	2984	6	30	54,584	16	
60	61	DeKalb College	2262	9	56	54,584	16	
61	62	Evanston University	1961	10	65	54,584	16	
62	63	Madison University	1837	11	47	54,584	16	

63 rows × 18 columns

In [768...

```
df_weathertrain = df_weather.dropna(axis=0, how='any')
df_weathertrain
```

Out[768...

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
0	1	Cedar Falls University	4165	2	120	47,420	21	
1	2	Oklahoma A&M	3746	3	58	47,420	21	
2	3	Urbana College	4943	5	67	47,420	21	
3	4	University of Bloomington	2366	9	83	47,420	21	
4	5	Indiana A&M	1796	10	74	47,420	21	
5	6	Minneapolis State University	1979	13	68	47,420	21	
6	7	Mt Pleasant College	3866	1	109	46,198	57	
7	8	University of Ames	5194	2	99	46,198	57	
8	9	Ann Arbor University	1909	5	6	46,198	57	
9	10	Evanston University	2523	6	55	46,198	57	
10	11	Madison University	2734	8	17	46,198	57	
11	12	Columbus University	2034	11	3	46,198	57	
12	13	Lincoln University	6463	1	6	44,211	73	
13	14	DeKalb College	3128	3	99	44,211	73	
14	15	Pennsylvania A&M	2972	6	1	44,211	73	
15	16	University of Bloomington	2158	8	68	44,211	73	
16	17	Urbana College	2120	10	78	44,211	73	
17	18	Minneapolis State University	1434	12	38	44,211	73	

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
18	19	University of Kalamazoo	2691	2	83	37,851	71	
19	20	University of Ames	3202	3	79	37,851	71	
20	21	Michigan A&M	1669	6	24	37,851	71	
21	22	Columbus University	2194	8	15	37,851	71	
22	23	Madison University	1536	9	4	37,851	71	
23	24	Evanston University	763	11	94	37,851	71	
24	25	Ohio A&M	3059	1	109	40,549	76	
25	26	Northern Cincinnati University	2390	2	79	40,549	76	
26	27	Pennsylvania A&M	3056	5	39	40,549	76	
27	28	University of Bloomington	2298	8	93	40,549	76	
28	29	Ann Arbor University	2956	9	17	40,549	76	
29	30	Minneapolis State University	2324	12	59	40,549	76	
30	31	LBJ University	2885	1	84	41,362	46	
31	32	University of Ames	3677	3	52	41,362	46	
32	33	University of Logan	1911	4	114	41,362	46	
33	34	Indiana A&M	2404	6	22	41,362	46	
34	35	Michigan A&M	2113	7	23	41,362	46	
35	36	Madison University	2345	10	32	41,362	46	
36	37	Evanston University	1969	11	69	41,362	46	

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
37	38	Northern Cincinnati University	3634	1	68	42,843	31	
38	39	Western New York University	2500	2	117	42,843	31	
39	40	University of Tempe	2810	4	29	42,843	31	
40	41	Ann Arbor University	3961	6	16	42,843	31	
41	42	Pennsylvania A&M	2440	9	44	42,843	31	
42	43	Urbana College	2017	10	45	42,843	31	
43	44	Minneapolis State University	2173	12	26	42,843	31	
44	45	Ohio A&M	4382	1	95	47,035	19	
45	46	University of Ames	4037	2	77	47,035	19	
46	47	Michigan A&M	2930	5	37	47,035	19	
47	48	Columbus University	2757	7	11	47,035	19	
48	49	Indiana A&M	2678	10	32	47,035	19	
49	50	Madison University	2445	12	15	47,035	19	
50	51	Letterman University	2985	1	111	54,584	19	
51	52	Cedar Falls University	2800	3	120	54,584	11	
52	53	Urbana College	2910	5	69	54,584	11	
53	54	University of Bloomington	2500	7	86	54,584	11	
54	55	Ann Arbor University	3721	8	7	54,584	11	

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
55	56	Minneapolis State University	2500	12	23	54,584	11	

In [770…
```
df_weather
```

Out[770…

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwba Jersey (1 Yes; 0 N
0	1	Cedar Falls University	4165	2	120	47,420	21	
1	2	Oklahoma A&M	3746	3	58	47,420	21	
2	3	Urbana College	4943	5	67	47,420	21	
3	4	University of Bloomington	2366	9	83	47,420	21	
4	5	Indiana A&M	1796	10	74	47,420	21	
…	…	…	…	…	…	…	…	
58	59	Columbus University	3397	5	4	54,584	16	
59	60	Indiana A&M	2984	6	30	54,584	16	
60	61	DeKalb College	2262	9	56	54,584	16	
61	62	Evanston University	1961	10	65	54,584	16	
62	63	Madison University	1837	11	47	54,584	16	

63 rows × 18 columns

In [774…
```
df_weathertest = df_weather.tail(7)
df_weathertest
```

Out[774…

	Index	Opponent	Magazine Sales (Units)	Week In Season	Opponent Preseason Rank	Preseason Ticket Sales	CSU Preseason Rank	Throwback Jersey (1 = Yes; 0 = No)
56	57	University of Missoula	3429	1	120	54,584	16	1
57	58	University of Ames	3518	3	46	54,584	16	0
58	59	Columbus University	3397	5	4	54,584	16	0
59	60	Indiana A&M	2984	6	30	54,584	16	0
60	61	DeKalb College	2262	9	56	54,584	16	0
61	62	Evanston University	1961	10	65	54,584	16	0
62	63	Madison University	1837	11	47	54,584	16	0

In [778…

```
df_weat = df_weathertest[['Year', 'Kickoff Temperature']]
df_weat
```

Out[778…

	Year	Kickoff Temperature
56	10	85
57	10	78
58	10	71
59	10	67
60	10	57
61	10	53
62	10	50

In [780…

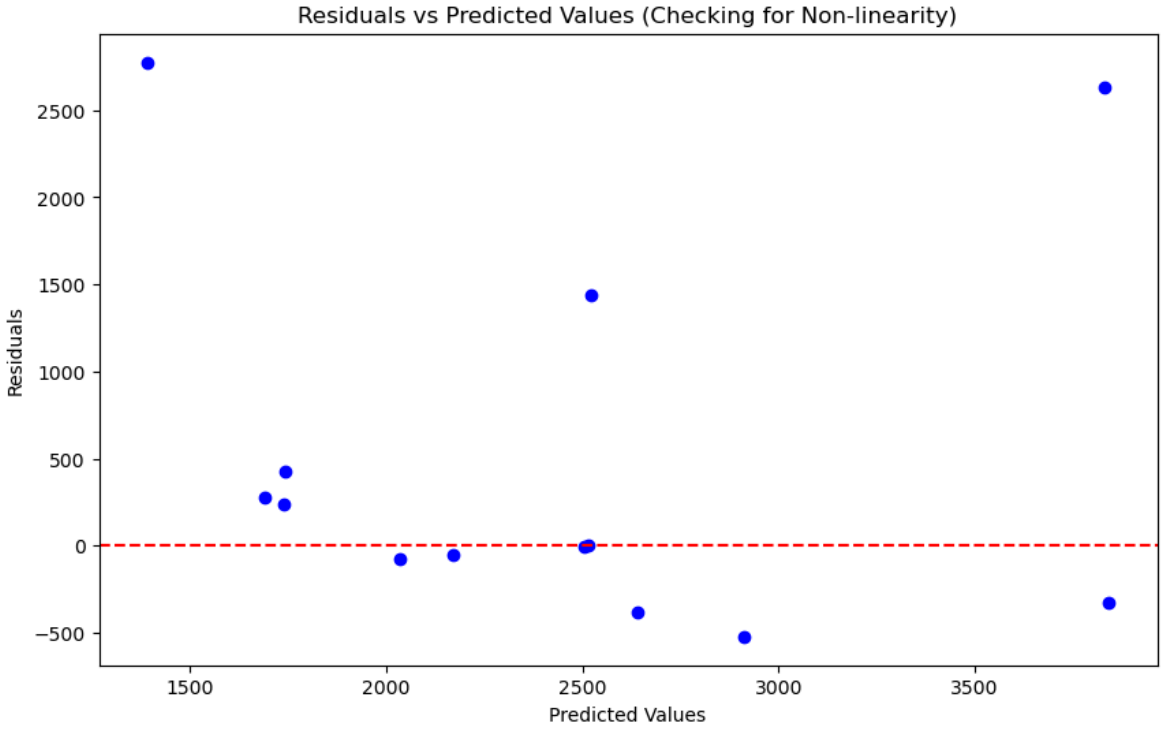
```
predicted_weather =model_weather.predict(df_weat)
print(predicted_weather)
```

```
['Sunny' 'Sunny' 'Sunny' 'Sunny' 'Sunny' 'Sunny' 'Sunny']
```

In [811…

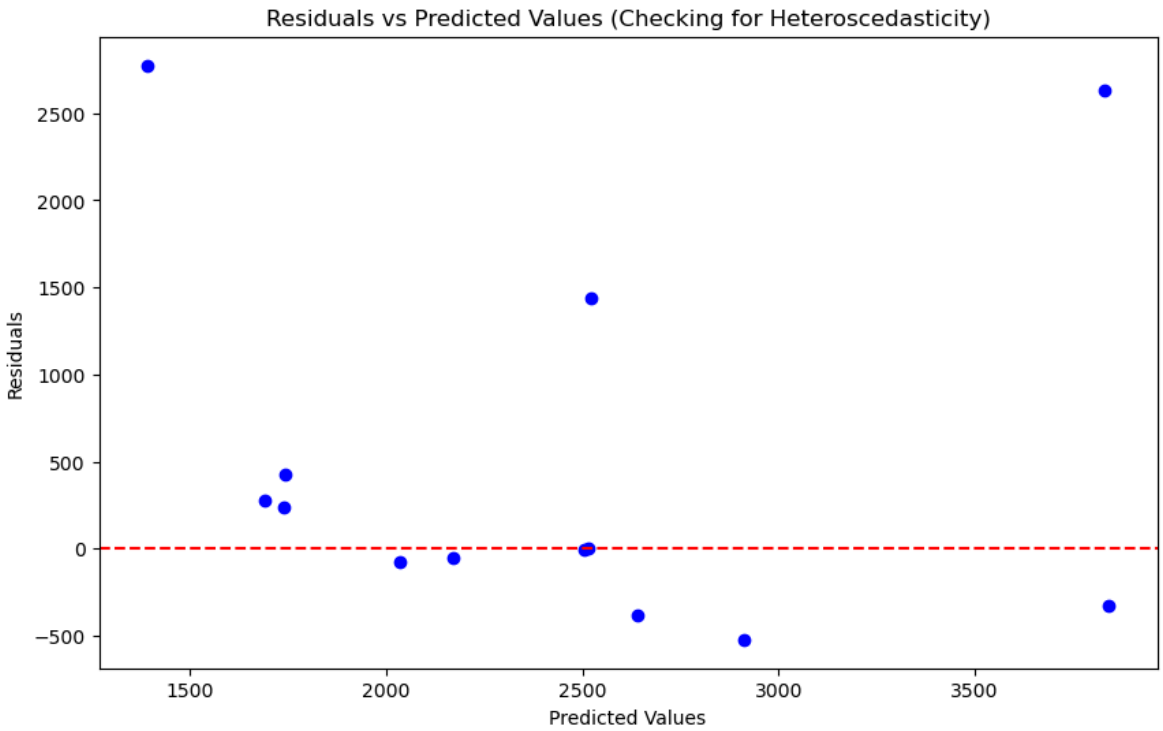
```
# Residuals plot to check for non-linearity
plt.figure(figsize=(10, 6))
plt.scatter(y_pred, y_test - y_pred, color='blue')
plt.axhline(y=0, color='red', linestyle='--')
plt.title('Residuals vs Predicted Values (Checking for Non-linearity)')
plt.xlabel('Predicted Values')
```

```
plt.ylabel('Residuals')
plt.show()
```



In [815…

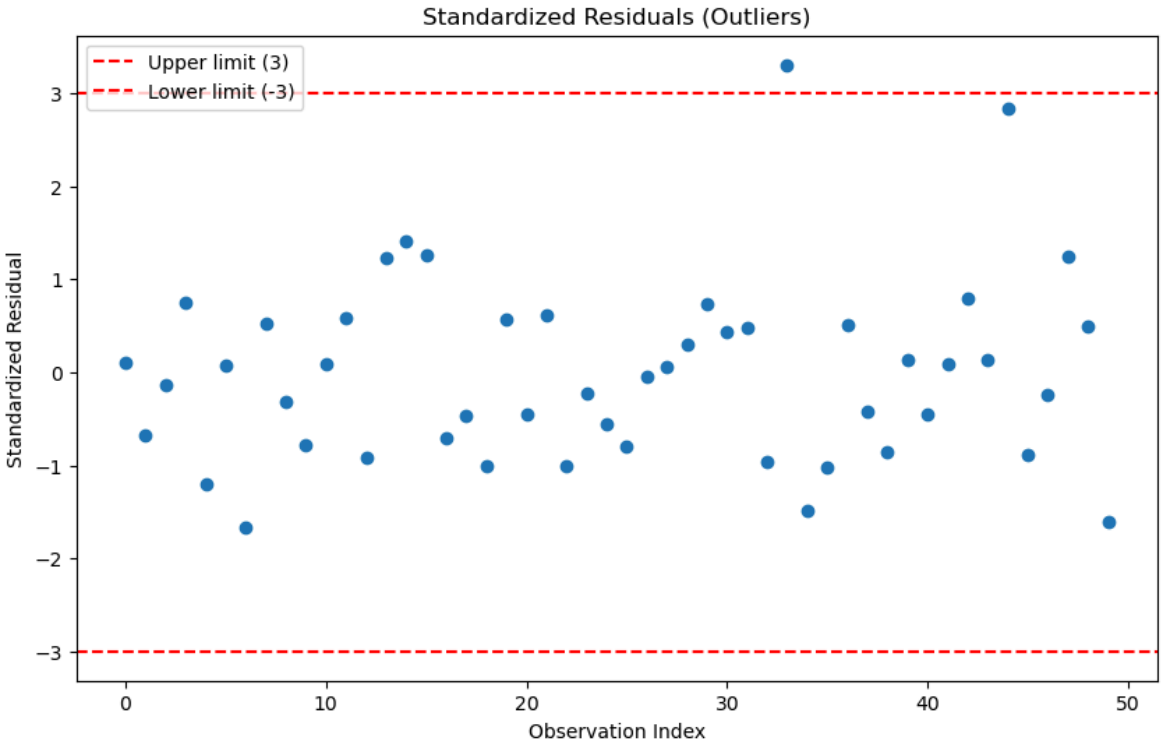
```
# Residuals vs Predicted plot to check for heteroscedasticity
plt.figure(figsize=(10, 6))
plt.scatter(y_pred, residuals, color='blue')
plt.axhline(y=0, color='red', linestyle='--')
plt.title('Residuals vs Predicted Values (Checking for Heteroscedasticity)')
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.show()
```



In [817…

```
#OUTLIERS

import numpy as np
```

```
import statsmodels.api as sm
import matplotlib.pyplot as plt

# Fit the linear regression model
X_train_with_const = sm.add_constant(X_train)  # Add constant term (intercept)
model = sm.OLS(y_train, X_train_with_const).fit()

# Calculate standardized residuals (z-scores)
influence = model.get_influence()
standardized_residuals = influence.resid_studentized_internal

# Plot standardized residuals
plt.figure(figsize=(10, 6))
plt.scatter(np.arange(len(standardized_residuals)), standardized_residuals)
plt.axhline(y=3, color='r', linestyle='--', label='Upper limit (3)')
plt.axhline(y=-3, color='r', linestyle='--', label='Lower limit (-3)')
plt.title('Standardized Residuals (Outliers)')
plt.xlabel('Observation Index')
plt.ylabel('Standardized Residual')
plt.legend()
plt.show()

# Print outlier indices
outliers = np.where(np.abs(standardized_residuals) > 3)[0]
print("Outlier indices:", outliers)
```

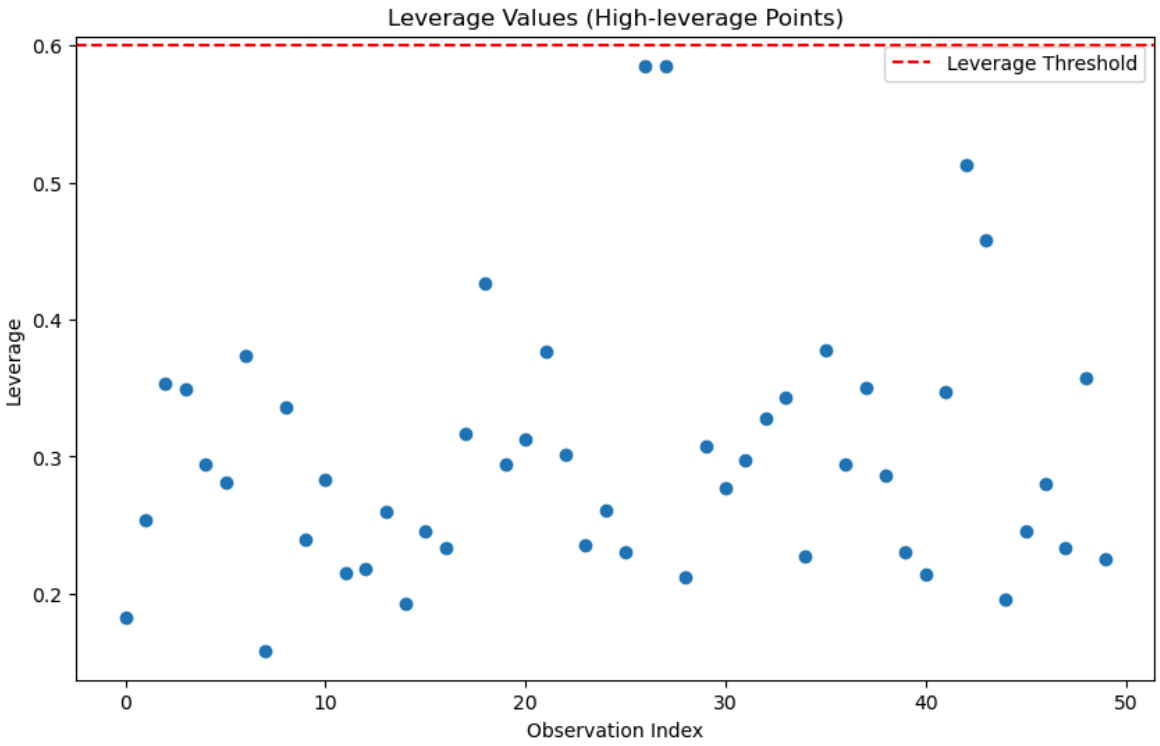



```
Outlier indices: [33]
```

In [819…

```
#HIGH LEVERAGE POINTS
# Calculate leverage values (Hat values)
leverage = influence.hat_matrix_diag

# Plot leverage values
plt.figure(figsize=(10, 6))
plt.scatter(np.arange(len(leverage)), leverage)
plt.axhline(y=2 * (X_train.shape[1] + 1) / len(y_train), color='r', linestyle='-
plt.title('Leverage Values (High-leverage Points)')
plt.xlabel('Observation Index')
```

```
plt.ylabel('Leverage')
plt.legend()
plt.show()

# Print high-leverage points
high_leverage_points = np.where(leverage > 2 * (X_train.shape[1] + 1) / len(y_tr
print("High-leverage points:", high_leverage_points)
```




```
High-leverage points: []
```

In [821…

```
#CO LINEARITY
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Calculate VIF for each predictor variable
X_train_with_const = sm.add_constant(X_train)  # Add constant term (intercept)
vif = pd.DataFrame()
vif["Variable"] = X_train.columns
vif["VIF"] = [variance_inflation_factor(X_train_with_const.values, i+1) for i in

print(vif)
```

	Variable	VIF
0	Week In Season	25.617486
1	Opponent Preseason Rank	5.125421
2	Preseason Ticket Sales	4.753499
3	CSU Preseason Rank	8.615561
4	Throwback Jersey (1 = Yes; 0 = No)	1.493685
5	Kickoff Temperature	3.741122
6	Home Game Number	22.699167
7	Conference Game (1 = Yes; 0 = No)	7.564752
8	Homecoming (1 = Yes; 0 = No)	1.748205
9	Opponent's Previous Season Number of Wins	56.266338
10	Opponent's Previous Season Number of Losses	49.483641
11	CSU's Previous Season Number of Wins	126.402775
12	CSU's Previous Season Number of Losses	126.145851
13	Game Day Weather	1.488528

In [ ]:

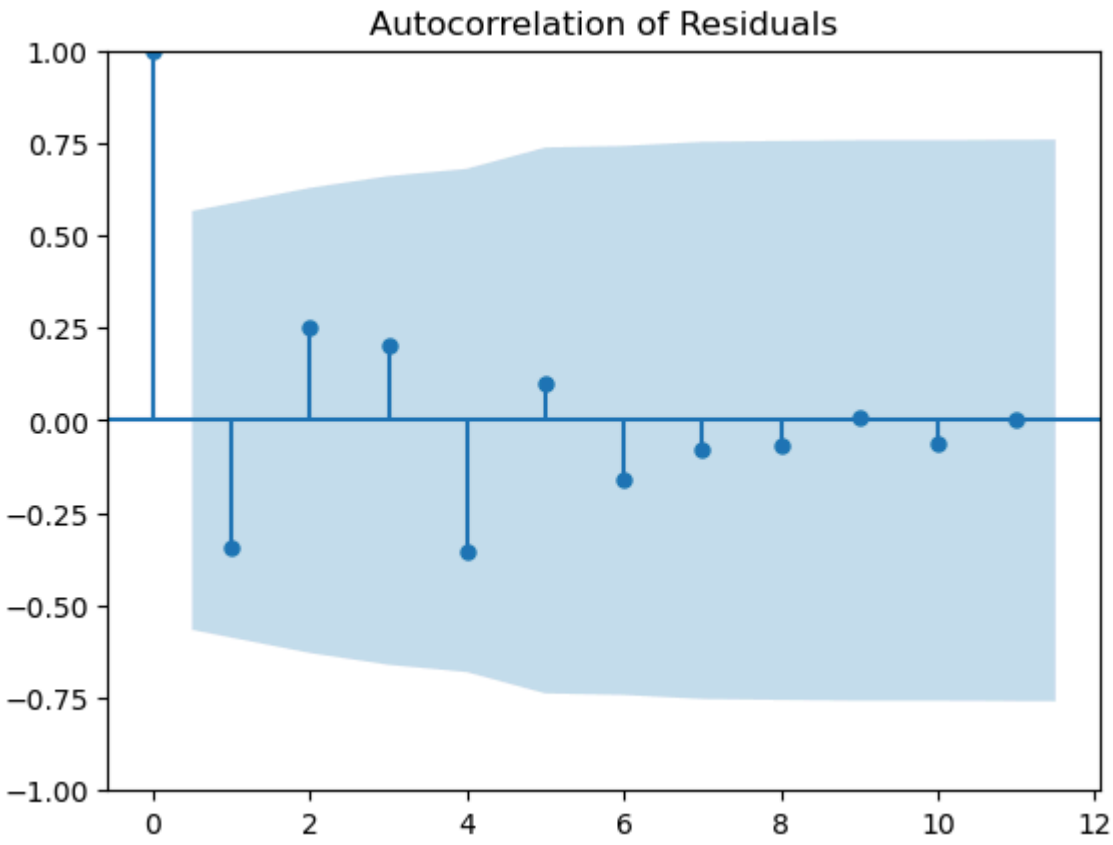
```
#Variance Inflation Factor (VIF) quantifies how much the variance of a regressio
```

In [823…

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import statsmodels.api as sm
from statsmodels.stats.stattools import durbin_watson
from statsmodels.graphics.tsaplots import plot_acf
import matplotlib.pyplot as plt

# Assuming 'df_encoded' is already prepared from your previous steps.
# Features (X) and Target (y) for Magazine Sales Prediction
X = df_encoded[['Week In Season', 'Opponent Preseason Rank', 'CSU Preseason Rank
y = df_encoded['Magazine Sales (Units)']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Initialize and train a basic Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Residuals
residuals = y_test - y_pred

# 1. Durbin-Watson Test for Autocorrelation
dw_stat = durbin_watson(residuals)
print(f'Durbin-Watson Statistic: {dw_stat}')

# 2. Autocorrelation Plot of Residuals
plt.figure(figsize=(10, 6))
plot_acf(residuals, lags=min(len(residuals) - 1, 20))
plt.title('Autocorrelation of Residuals')
plt.show()

# 3. Generalized Least Squares (GLS) Model
# Add a constant to the features for GLS
X_train_const = sm.add_constant(X_train)
X_test_const = sm.add_constant(X_test)

# Fit GLS model
model_gls = sm.GLS(y_train, X_train_const).fit()

# Make predictions using GLS
y_pred_gls = model_gls.predict(X_test_const)

# Evaluate GLS Model
mse_gls = mean_squared_error(y_test, y_pred_gls)
r2_gls = r2_score(y_test, y_pred_gls)
print(f"GLS Model - Mean Squared Error: {mse_gls:.2f}")
print(f"GLS Model - R-squared Score: {r2_gls:.2f}")

# 4. Adding Lagged Dependent Variable (Magazine Sales Lag)
df_encoded['Magazine_Sales_Lag'] = df_encoded['Magazine Sales (Units)'].shift(1)

# Drop rows where either X or y contains NaN values due to shifting
```

```
lagged_data = df_encoded[['Magazine Sales (Units)', 'Week In Season', 'Opponent
                          'CSU Preseason Rank', 'Year', 'Home Game Number', 'Mag

# Now split the data into features (X_with_lag) and target (y_with_lag)
X_with_lag = lagged_data[['Week In Season', 'Opponent Preseason Rank', 'CSU Pres
y_with_lag = lagged_data['Magazine Sales (Units)']

# Split the data into training and testing sets
X_train_lag, X_test_lag, y_train_lag, y_test_lag = train_test_split(X_with_lag,

# Train the linear regression model with lagged data
model_lag = LinearRegression()
model_lag.fit(X_train_lag, y_train_lag)

# Make predictions and evaluate
y_pred_lag = model_lag.predict(X_test_lag)
mse_lag = mean_squared_error(y_test_lag, y_pred_lag)
r2_lag = r2_score(y_test_lag, y_pred_lag)

print(f"Lag Model - Mean Squared Error: {mse_lag:.2f}")
print(f"Lag Model - R-squared Score: {r2_lag:.2f}")

# 5. Linear Model with Newey-West Standard Errors (HAC)
# Fit an OLS model but adjust for HAC errors (heteroscedasticity and autocorrela
model_ols_hac = sm.OLS(y_train, X_train_const).fit(cov_type='HAC', cov_kwds={'ma

# Predict and evaluate HAC model
y_pred_hac = model_ols_hac.predict(X_test_const)

# Calculate performance metrics for HAC model
mse_hac = mean_squared_error(y_test, y_pred_hac)
r2_hac = r2_score(y_test, y_pred_hac)

print(f"HAC Model - Mean Squared Error: {mse_hac:.2f}")
print(f"HAC Model - R-squared Score: {r2_hac:.2f}")
```

```
Durbin-Watson Statistic: 2.6387638360650207
<Figure size 1000x600 with 0 Axes>
```




```
GLS Model - Mean Squared Error: 452733.99
GLS Model - R-squared Score: 0.48
Lag Model - Mean Squared Error: 1019242.55
Lag Model - R-squared Score: 0.37
HAC Model - Mean Squared Error: 452733.99
HAC Model - R-squared Score: 0.48
```

In [851…

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import LabelEncoder
import warnings

# Suppress specific warnings
warnings.simplefilter(action='ignore', category=pd.errors.SettingWithCopyWarning
warnings.simplefilter(action='ignore', category=UserWarning)

# Step 1: Load the data from a CSV file
df = pd.read_csv('Magazine.csv')
df['Preseason Ticket Sales'] = df['Preseason Ticket Sales'].str.replace(',', '',
df['Preseason Ticket Sales'] = df['Preseason Ticket Sales'].astype(float)

# Step 2: Feature selection
# We'll select relevant features to predict magazine sales
features = ['Week In Season', 'Opponent Preseason Rank', 'Preseason Ticket Sales
            'CSU Preseason Rank', 'Throwback Jersey (1 = Yes; 0 = No)', 'Kickoff
            'Home Game Number', 'Conference Game (1 = Yes; 0 = No)', 'Homecoming
            'Opponent\'s Previous Season Number of Wins',
            'Opponent\'s Previous Season Number of Losses',
            'CSU\'s Previous Season Number of Wins',
            'CSU\'s Previous Season Number of Losses']

X = df[features].copy()  # Create a copy of the selected features
```

```
y = df['Magazine Sales (Units)']  # Target variable

# Step 3: Preprocessing (convert categorical columns if any)
# Assuming 'Throwback Jersey', 'Conference Game', 'Homecoming' are already binar
# If 'Game Day Weather' is a categorical feature, we need to encode it:
label_encoder = LabelEncoder()
df['Game Day Weather'] = label_encoder.fit_transform(df['Game Day Weather'])
X.loc[:, 'Game Day Weather'] = df['Game Day Weather']  # Use .loc to assign

# Step 4: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Step 5: Build the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Step 6: Evaluate the model on the test set
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")

# Step 7: Predict magazine sales based on average conditions across the season
# Instead of inputting new data, we use the average of the features to get the p
X_mean = X.mean().values.reshape(1, -1)  # Average values of all features
predicted_sales = model.predict(X_mean)
print(f"Predicted magazine sales (average across season): {predicted_sales[0]}")

# Step 8: Calculate the optimal number of magazines to order
cost_per_unit = 10
selling_price = 30
salvage_value = 5

# Using the predicted sales as the forecasted demand
optimal_order_quantity = predicted_sales[0]  # Simplified assumption
print(f"Optimal number of magazines to order: {int(optimal_order_quantity)}")
```

```
Mean Squared Error: 1349178.243046289
Predicted magazine sales (average across season): 2699.903022551285
Optimal number of magazines to order: 2699
```

In [ ]: