IST 718 | LAB 1

Matthew L. Pergolski

Import Packages

```
import pandas as pd
import numpy as np
import seaborn as sns
import statsmodels.api as sm
```

Read in Data

```
# Read data
generic_df = pd.read_csv("https://raw.githubusercontent.com/2SUBDA/IST_718/68273222
coaches_df = pd.read_csv("https://raw.githubusercontent.com/2SUBDA/IST_718/68273222
stadiums = pd.read_html("https://www.collegegridirons.com/comparisons-by-capacity/'
teamrecord = pd.read_html("https://www.teamrankings.com/ncf/trends/win_trends/")
grads = pd.read_excel('/content/sample_data/gradyear.xlsx')
```

Inspect Coaches dataset

```python
# Overview of coaches_df dataset

coaches_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 129 entries, 0 to 128
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   School        129 non-null    object
 1   Conference    129 non-null    object
 2   Coach         129 non-null    object
 3   SchoolPay     129 non-null    object
 4   TotalPay      129 non-null    object
 5   Bonus         129 non-null    object
 6   BonusPaid     129 non-null    object
 7   AssistantPay  129 non-null    object
 8   Buyout        129 non-null    object
dtypes: object(9)
memory usage: 9.2+ KB
```

```
coaches_df.head(20)
```

	School	Conference	Coach	SchoolPay	TotalPay	Bonus	BonusPaid
0	Air Force	Mt. West	Troy Calhoun	885000	885000	247000	--
1	Akron	MAC	Terry Bowden	$411,000	$412,500	$225,000	$50,000
2	Alabama	SEC	Nick Saban	$8,307,000	$8,307,000	$1,100,000	$500,000
3	Alabama at Birmingham	C-USA	Bill Clark	$900,000	$900,000	$950,000	$165,471
4	Appalachian State	Sun Belt	Scott Satterfield	$712,500	$712,500	$295,000	$145,000
5	Arizona	Pac-12	Kevin Sumlin	$1,600,000	$2,000,000	$2,025,000	--
6	Arizona State	Pac-12	Herm Edwards	$2,000,000	$2,000,000	$3,010,000	--
7	Arkansas	SEC	Chad Morris	$3,500,000	$3,500,000	$1,000,000	--
8	Arkansas State	Sun Belt	Blake Anderson	$825,000	$825,000	$185,000	$25,000
9	Army	Ind.	Jeff Monken	932521	932521	--	--
10	Auburn	SEC	Gus Malzahn	$6,700,000	$6,705,656	$1,400,000	$375,000
11	Ball State	MAC	Mike Neu	$435,689	$435,689	$380,000	$30,000
12	Baylor	Big 12	Matt Rhule	--	--	--	--
13	Boise State	Mt. West	Bryan	$1,650,010	$1,650,010	$475,000	$145,000

```
coaches_df['Conference'].value_counts()
```

```
SEC         14
C-USA       14
ACC         14
Big Ten     14
Mt. West    12
MAC         12
Pac-12      12
AAC         11
Sun Belt    10
Big 12      10
Ind.         6
Name: Conference, dtype: int64
```

```
# Clean coaches_df
coaches_df.replace('Pac-12', 'Pac 12', inplace=True)
coaches_df.replace('--', 0, inplace=True)

replace_dict = {col: str for col in ['School', 'Conference', 'Coach']}
coaches_df = coaches_df.astype(replace_dict).replace(',', '', regex=True)

money_columns = ['SchoolPay', 'TotalPay', 'Bonus', 'BonusPaid', 'AssistantPay', 'Bu
for col in money_columns:
    coaches_df[col] = coaches_df[col].str.replace('[^0-9.]', '', regex=True).astype

coaches_df.fillna(coaches_df.mean(), inplace=True)
coaches_df.head(20)
```

```
<ipython-input-170-8df8af9d99cb>:12: FutureWarning: The default value of numer
  coaches_df.fillna(coaches_df.mean(), inplace=True)
```

	School	Conference	Coach	SchoolPay	TotalPay	Bonus	Bonus
0	Air Force	Mt. West	Troy Calhoun	885000.000	885000.00	2.470000e+05	149524.2
1	Akron	MAC	Terry Bowden	411000.000	412500.00	2.250000e+05	50000.0
2	Alabama	SEC	Nick Saban	8307000.000	8307000.00	1.100000e+06	500000.0
3	Alabama at Birmingham	C-USA	Bill Clark	900000.000	900000.00	9.500000e+05	165471.0
4	Appalachian State	Sun Belt	Scott Satterfield	712500.000	712500.00	2.950000e+05	145000.0
5	Arizona	Pac 12	Kevin Sumlin	1600000.000	2000000.00	2.025000e+06	149524.2
6	Arizona State	Pac 12	Herm Edwards	2000000.000	2000000.00	3.010000e+06	149524.2
7	Arkansas	SEC	Chad Morris	3500000.000	3500000.00	1.000000e+06	149524.2
8	Arkansas State	Sun Belt	Blake Anderson	825000.000	825000.00	1.850000e+05	25000.0
9	Army	Ind.	Jeff Monken	932521.000	932521.00	8.741782e+05	149524.2
10	Auburn	SEC	Gus Malzahn	6700000.000	6705656.00	1.400000e+06	375000.0
11	Ball State	MAC	Mike Neu	435689.000	435689.00	3.800000e+05	30000.0
12	Baylor	Big 12	Matt Rhule	2410300.712	2417060.76	8.741782e+05	149524.2
13	Boise State	Mt. West	Bryan Harsin	1650010.000	1650010.00	4.750000e+05	145000.0

```
coaches_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 129 entries, 0 to 128
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   School        129 non-null    object 
 1   Conference    129 non-null    object 
 2   Coach         129 non-null    object 
 3   SchoolPay     129 non-null    float64
 4   TotalPay      129 non-null    float64
 5   Bonus         129 non-null    float64
 6   BonusPaid     129 non-null    float64
 7   AssistantPay  129 non-null    float64
 8   Buyout        129 non-null    float64
dtypes: float64(6), object(3)
memory usage: 9.2+ KB
```

```
coaches_df['School'].value_counts().to_csv('coaches_schools.csv')
```

Inspect Stadiums dataset

```
stadiums = stadiums[0]
stadiums.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 131 entries, 0 to 130
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Stadium     131 non-null    object
 1   College     131 non-null    object
 2   Conference  131 non-null    object
 3   Capacity    131 non-null    int64 
 4   Opened      131 non-null    int64 
dtypes: int64(2), object(3)
memory usage: 5.2+ KB
```

```
stadiums.head(20)
```

	Stadium	College	Conference	Capacity	Opened
0	Michigan Stadium	Michigan	Big Ten	107601	1927
1	Beaver Stadium	Penn State	Big Ten	106572	1960
2	Ohio Stadium	Ohio State	Big Ten	104944	1922
3	Kyle Field	Texas A&M	SEC	102733	1904
4	Neyland Stadium	Tennessee	SEC	102521	1921
5	Bryant Denny Stadium	Alabama	SEC	101821	1929
6	Tiger Stadium	LSU	SEC	100500	1924
7	Royal Memorial Stadium	Texas	Big 12	100119	1924
8	Los Angeles Coliseum	USC	Pac 12	93607	1923
9	Sanford Stadium	Georgia	SEC	92746	1929
10	Memorial Stadium	Nebraska	Big Ten	92000	1923
11	Rose Bowl	UCLA	Pac 12	89702	1921
12	Ben Hill Griffin Stadium	Florida	SEC	88548	1930
13	Jordan Hare Stadium	Auburn	SEC	87451	1939
14	Memorial Stadium	Oklahoma	Big 12	84000	1925
15	Doak Campbell Stadium	Florida State	ACC	82300	1950
16	Memorial Stadium	Clemson	ACC	81500	1942
17	Camp Randall Stadium	Wisconsin	Big Ten	80321	1917
18	Williams Brice Stadium	South Carolina	SEC	80250	1934
19	Notre Dame Stadium	Notre Dame	Independent	77622	1930

```
stadiums = stadiums[['College', 'Conference', 'Capacity']]
stadiums.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 131 entries, 0 to 130
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   College     131 non-null    object
 1   Conference  131 non-null    object
 2   Capacity    131 non-null    int64 
dtypes: int64(1), object(2)
memory usage: 3.2+ KB
```

```
stadiums['College'].value_counts().to_csv('stadiums_schools.csv')
```

```
stadiums = stadiums.rename(columns = {'College' : 'School'})
```

```
stadiums.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 131 entries, 0 to 130
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   School      131 non-null    object
 1   Conference  131 non-null    object
 2   Capacity    131 non-null    int64 
dtypes: int64(1), object(2)
memory usage: 3.2+ KB
```

Inspect teamrecord dataset

```
teamrecord = teamrecord[0]
teamrecord.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 131 entries, 0 to 130
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Team             131 non-null    object 
 1   Win-Loss Record  131 non-null    object 
 2   Win %            131 non-null    object 
 3   MOV              131 non-null    float64
 4   ATS +/-          131 non-null    float64
dtypes: float64(2), object(3)
memory usage: 5.2+ KB
```

```
teamrecord.head(20)
```

	Team	Win-Loss Record	Win %	MOV	ATS +/-
0	Georgia	15-0-0	100.0%	26.8	1.6
1	Michigan	13-1-0	92.9%	24.4	2.9
2	TX Christian	13-2-0	86.7%	9.8	5.4
3	Troy	12-2-0	85.7%	8.4	4.2
4	Tulane	12-2-0	85.7%	13.8	7.1
5	Penn State	11-2-0	84.6%	17.5	7.3
6	Washington	11-2-0	84.6%	13.8	3.7
7	Alabama	11-2-0	84.6%	22.9	-1.9
8	Tennessee	11-2-0	84.6%	23.3	9.0
9	Ohio State	11-2-0	84.6%	23.2	-2.2
10	Clemson	11-3-0	78.6%	12.4	-2.8
11	USC	11-3-0	78.6%	12.1	-0.1
12	TX-San Ant	11-3-0	78.6%	10.9	0.6
13	Florida St	10-3-0	76.9%	15.5	4.7
14	Air Force	10-3-0	76.9%	14.5	2.2
15	Oregon	10-3-0	76.9%	11.4	1.3
16	Oregon St	10-3-0	76.9%	12.2	8.2
17	S Alabama	10-3-0	76.9%	9.9	3.0
18	James Mad	8-3-0	72.7%	16.1	5.0
19	LSU	10-4-0	71.4%	12.0	5.4

```
teamrecord['Win %'] = teamrecord['Win %'].str.replace('%', '').astype(float)
teamrecord = teamrecord[['Team', 'Win %']]
teamrecord.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 131 entries, 0 to 130
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Team    131 non-null    object 
 1   Win %   131 non-null    float64
dtypes: float64(1), object(1)
memory usage: 2.2+ KB
```

```
teamrecord['Team'].value_counts().to_csv('teamrecord_schools.csv')
```

```
teamrecord = teamrecord.rename(columns = {'Team' : 'School'})
```

Inspect graduation rates

```
grads = pd.read_excel('/content/sample_data/gradyear.xlsx')
grads.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 257 entries, 0 to 256
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Cohort Year  257 non-null    int64  
 1   School       257 non-null    object 
 2   Conference   257 non-null    object 
 3   Sport        257 non-null    object 
 4   State        257 non-null    object 
 5   GSR          257 non-null    int64  
 6   FGR          237 non-null    float64
dtypes: float64(1), int64(2), object(4)
memory usage: 14.2+ KB
```

```
grads.head(20)
```

	Cohort Year	School	Conference	Sport	State	GSR	FGR
0	2015	Abilene Christian University	ASUN Conference	Football	TX	75	54.0
1	2015	University of Akron	Mid-American Conference	Football	OH	74	76.0
2	2015	Alabama A&M University	Southwestern Athletic Conf.	Football	AL	66	54.0
3	2015	Alabama State University	Southwestern Athletic Conf.	Football	AL	71	41.0
4	2015	University of Alabama	Southeastern Conference	Football	AL	89	66.0
5	2015	University of Alabama at Birmingham	Conference USA	Football	AL	80	44.0
6	2015	University at Albany	Colonial Athletic Association	Football	NY	74	63.0
7	2015	Alcorn State University	Southwestern Athletic Conf.	Football	MS	72	61.0
8	2015	Appalachian State University	Sun Belt Conference	Football	NC	81	57.0
9	2015	Arizona State University	Pac-12 Conference	Football	AZ	87	67.0
10	2015	University of Arizona	Pac-12 Conference	Football	AZ	75	61.0
11	2015	Arkansas State University	Sun Belt Conference	Football	AR	75	52.0
12	2015	University of Arkansas, Fayetteville	Southeastern Conference	Football	AR	87	60.0
13	2015	Auburn University	Southeastern Conference	Football	AL	83	65.0

```
grads['School'].value_counts()
```

```
Abilene Christian University                      1
University of North Carolina, Chapel Hill         1
University of Pittsburgh                          1
Portland State University                         1
Prairie View A&M University                       1
                                                 ..
Jacksonville State University                     1
James Madison University                          1
Kansas State University                           1
University of Kansas                              1
Utah Tech University                              1
Name: School, Length: 257, dtype: int64
```

```
grads.isnull().sum()
```

```
Cohort Year     0
School          0
Conference      0
Sport           0
State           0
GSR             0
FGR            20
dtype: int64
```

```
grads = grads.dropna()
grads.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 237 entries, 0 to 256
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Cohort Year  237 non-null    int64
 1   School       237 non-null    object
 2   Conference   237 non-null    object
 3   Sport        237 non-null    object
 4   State        237 non-null    object
 5   GSR          237 non-null    int64
 6   FGR          237 non-null    float64
dtypes: float64(1), int64(2), object(4)
memory usage: 14.8+ KB
```

Merge data into one

```
!pip install -Uqq fuzzywuzzy
```

```
from fuzzywuzzy import fuzz
from fuzzywuzzy import process
import pandas as pd

# find the best match for each school name in the stadiums, teamrecord, and grads d
stadiums['Match'] = stadiums['School'].apply(lambda x: process.extractOne(x, coache
teamrecord['Match'] = teamrecord['School'].apply(lambda x: process.extractOne(x, co
grads['Match'] = grads['School'].apply(lambda x: process.extractOne(x, coaches_df['

# replace school names in the stadiums, teamrecord, and grads data frames with the
stadiums['School'] = stadiums['Match']
teamrecord['School'] = teamrecord['Match']
grads['School'] = grads['Match']

# drop the Match column
stadiums.drop('Match', axis=1, inplace=True)
teamrecord.drop('Match', axis=1, inplace=True)
grads.drop('Match', axis=1, inplace=True)

# merge data frames on the School column
df = pd.merge(coaches_df, stadiums, on='School')
df = pd.merge(df, teamrecord, on='School')
df = pd.merge(df, grads, on='School')

# drop duplicate rows based on the School column
df = df.drop_duplicates(subset=['School'])

# print information about the data frame
df.info()

# find dropped schools
original_schools = set(coaches_df['School']).union(set(stadiums['School'])).union(s
merged_schools = set(df['School'])
dropped_schools = original_schools - merged_schools
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 82 entries, 0 to 441
Data columns (total 18 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   School        82 non-null     object 
 1   Conference_x  82 non-null     object 
 2   Coach         82 non-null     object 
 3   SchoolPay     82 non-null     float64
 4   TotalPay      82 non-null     float64
 5   Bonus         82 non-null     float64
 6   BonusPaid     82 non-null     float64
 7   AssistantPay  82 non-null     float64
 8   Buyout        82 non-null     float64
 9   Conference_y  82 non-null     object 
 10  Capacity      82 non-null     int64  
 11  Win %         82 non-null     float64
 12  Cohort Year   82 non-null     int64  
 13  Conference    82 non-null     object 
 14  Sport         82 non-null     object 
 15  State         82 non-null     object 
 16  GSR           82 non-null     int64  
 17  FGR           82 non-null     float64
dtypes: float64(8), int64(3), object(7)
memory usage: 12.2+ KB
```

```
print(len(original_schools))
original_schools
```

```
 'Nebraska',
 'Nevada',
 'Nevada-Las Vegas',
 'New Mexico',
 'New Mexico State',
 'North Carolina',
 'North Carolina State',
 'North Texas',
 'Northern Illinois',
 'Northwestern',
 'Notre Dame',
 'Ohio',
 'Ohio State',
 'Oklahoma',
 'Oklahoma State',
 'Old Dominion',
 'Oregon',
 'Oregon State',
 'Penn State',
 'Pittsburgh',
 'Purdue',
 'Rice',
 'Rutgers',
 'San Diego State'
```

```
      'San Diego State',
      'San Jose State',
      'South Alabama',
      'South Carolina',
      'South Florida',
      'Southern California',
      'Southern Methodist',
      'Southern Mississippi',
      'Stanford',
      'Syracuse',
      'Tennessee',
      'Texas',
      'Texas A&M',
      'Texas Christian',
      'Texas State',
      'Texas Tech',
      'Texas–El Paso',
      'Texas–San Antonio',
      'Toledo',
      'Troy',
      'Tulane',
      'Tulsa',
      'UCLA',
      'Utah',
      'Utah State',
      'Vanderbilt',
      'Virginia',
      'Virginia Tech',
      'Wake Forest',
      'Washington',
      'Washington State',
      'West Virginia',
      'Western Kentucky',
      'Western Michigan',
      'Wisconsin',
      'Wyoming'}
```

```
print(len(merged_schools))
merged_schools
```

```
      'Duke',
      'East Carolina',
      'Florida',
      'Georgia',
      'Hawaii',
      'Houston',
      'Illinois',
      'Indiana',
      'Iowa',
      'Kansas',
      'Kent State',
      'Kentucky',
      'LSU',
      'Louisiana Lafayette',
      'Louisiana–Lafayette',
      'Louisiana–Monroe',
      'Louisville',
      'Marshall',
      'Maryland',
      'Massachusetts',
      'Memphis',
      'Michigan',
      'Middle Tennessee',
      'Minnesota',
      'Mississippi',
      'Missouri',
      'Navy',
      'Nebraska',
      'Nevada',
      'New Mexico',
      'North Texas',
      'Northwestern',
      'Notre Dame',
      'Ohio',
      'Oklahoma',
      'Old Dominion',
      'Oregon',
      'Penn State',
      'Pittsburgh',
      'Purdue',
      'Rice',
      'San Diego State',
      'San Jose State',
      'Stanford',
      'Syracuse',
      'Tennessee',
      'Texas',
      'Texas A&M',
      'Texas State',
      'Toledo',
      'Troy',
      'Tulane',
      'Tulsa',
      'Utah',
      'Vanderbilt',
      'Virginia',
      'Wake Forest',
      'Washington',
      'Wisconsin',
      'Wyoming'}
```

```python
print(len(dropped_schools))
dropped_schools
```

```
47
{'Air Force',
 'Alabama at Birmingham',
 'Arizona State',
 'Arkansas State',
 'Colorado State',
 'Eastern Michigan',
 'Florida Atlantic',
 'Florida International',
 'Florida State',
 'Fresno State',
 'Georgia Southern',
 'Georgia State',
 'Georgia Tech',
 'Iowa State',
 'Kansas State',
 'Liberty',
 'Louisiana Tech',
 'Miami (Fla.)',
 'Miami (Ohio)',
 'Michigan State',
 'Mississippi State',
 'Nevada-Las Vegas',
 'New Mexico State',
 'North Carolina',
 'North Carolina State',
 'Northern Illinois',
 'Ohio State',
 'Oklahoma State',
 'Oregon State',
 'Rutgers',
 'South Alabama',
 'South Carolina',
 'South Florida',
 'Southern California',
 'Southern Methodist',
 'Southern Mississippi',
 'Texas Christian',
 'Texas Tech',
 'Texas-El Paso',
 'Texas-San Antonio',
 'UCLA',
 'Utah State',
 'Virginia Tech',
 'Washington State',
 'West Virginia',
 'Western Kentucky',
 'Western Michigan'}
```

```
df.to_csv('df.csv')
df.head(40)
```

	School	Conference_x	Coach	SchoolPay	TotalPay	Bonus	B
0	Akron	MAC	Terry Bowden	411000.000	412500.00	2.250000e+05	5.00
1	Alabama	SEC	Nick Saban	8307000.000	8307000.00	1.100000e+06	5.00
43	Appalachian State	Sun Belt	Scott Satterfield	712500.000	712500.00	2.950000e+05	1.45
163	Arizona	Pac 12	Kevin Sumlin	1600000.000	2000000.00	2.025000e+06	1.49
169	Arkansas	SEC	Chad Morris	3500000.000	3500000.00	1.000000e+06	1.49
177	Army	Ind.	Jeff Monken	932521.000	932521.00	8.741782e+05	1.49
178	Auburn	SEC	Gus Malzahn	6700000.000	6705656.00	1.400000e+06	3.75
182	Ball State	MAC	Mike Neu	435689.000	435689.00	3.800000e+05	3.00
183	Baylor	Big 12	Matt Rhule	2410300.712	2417060.76	8.741782e+05	1.49
184	Boise State	Mt. West	Bryan Harsin	1650010.000	1650010.00	4.750000e+05	1.45
185	Boston College	ACC	Steve Addazio	2514859.000	2514859.00	8.741782e+05	1.49
191	Bowling Green	MAC	Mike Jinks	437228.000	437228.00	2.450000e+05	8.12
192	Brigham Young	Ind.	Kalani Sitake	2410300.712	2417060.76	8.741782e+05	1.49
193	Buffalo	MAC	Lance Leipold	455500.000	455500.00	3.810000e+05	0.00
194	California	Pac 12	Justin	1500000.000	1500000.00	9.000000e+05	7.50

194	California	Pac 12	Wilcox	1500000.000	1500000.00	9.000000e+05	7.50
201	Central Florida	AAC	Josh Heupel	1700000.000	1700000.00	2.500000e+05	1.49
203	Central Michigan	MAC	John Bonamego	655000.000	655000.00	4.150000e+05	4.50
204	Charlotte	C-USA	Brad Lambert	625000.000	625000.00	1.200000e+05	0.00
205	Cincinnati	AAC	Luke Fickell	2000000.000	2000000.00	6.250000e+05	0.00
206	Clemson	ACC	Dabo Swinney	6205000.000	6543350.00	1.125000e+06	5.00
207	Coastal Carolina	Sun Belt	Joe Moglia	400000.000	400000.00	8.000000e+05	2.50
211	Colorado	Pac 12	Mike	2878500.000	2878500.00	2.150000e+06	2.97

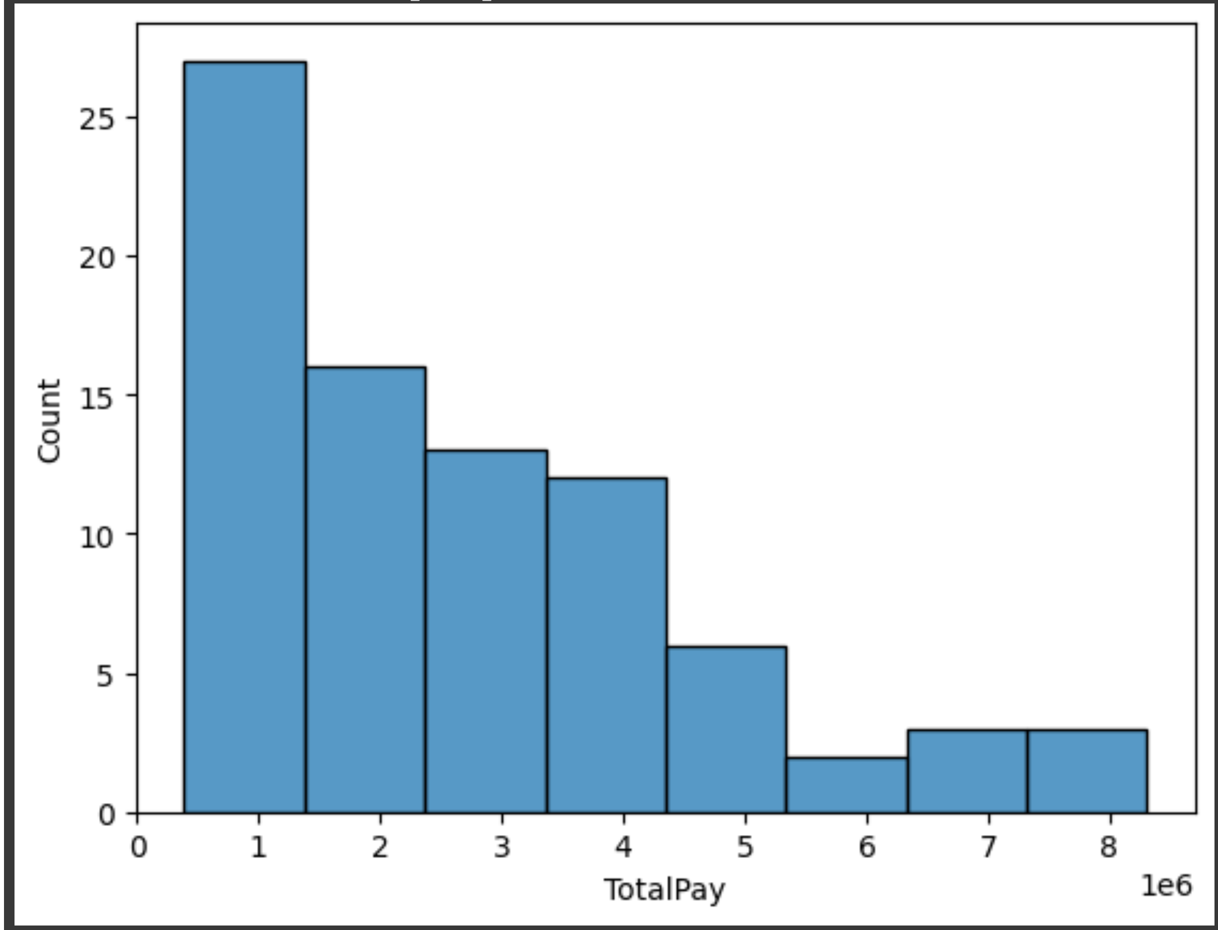
```
df = df.drop(['Conference_x'], axis = 1)
#df = df.rename(columns={'Conference_y': 'Conference'})
df = df.drop(['Conference_y'], axis = 1)
df = df.dropna()
#df = df.drop('matched_school', axis=1)
df = df.drop('Cohort Year', axis=1)
df['Capacity'] = df['Capacity'].astype(float)
#df.rename(columns={'Conference': 'Conference2'}, inplace=True)
#df = df.drop('Conference2', axis=1)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 82 entries, 0 to 441
Data columns (total 15 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   School        82 non-null     object 
 1   Coach         82 non-null     object 
 2   SchoolPay     82 non-null     float64
 3   TotalPay      82 non-null     float64
 4   Bonus         82 non-null     float64
 5   BonusPaid     82 non-null     float64
 6   AssistantPay  82 non-null     float64
 7   Buyout        82 non-null     float64
 8   Capacity      82 non-null     float64
 9   Win %         82 non-null     float64
 10  Conference    82 non-null     object 
 11  Sport         82 non-null     object 
 12  State         82 non-null     object 
 13  GSR           82 non-null     int64  
 14  FGR           82 non-null     float64
dtypes: float64(9), int64(1), object(5)
memory usage: 10.2+ KB
```

Data Exploration

```
# View Distribution of TotalPay
sns.histplot(df['TotalPay'])
```

```
<Axes: xlabel='TotalPay', ylabel='Count'>
```



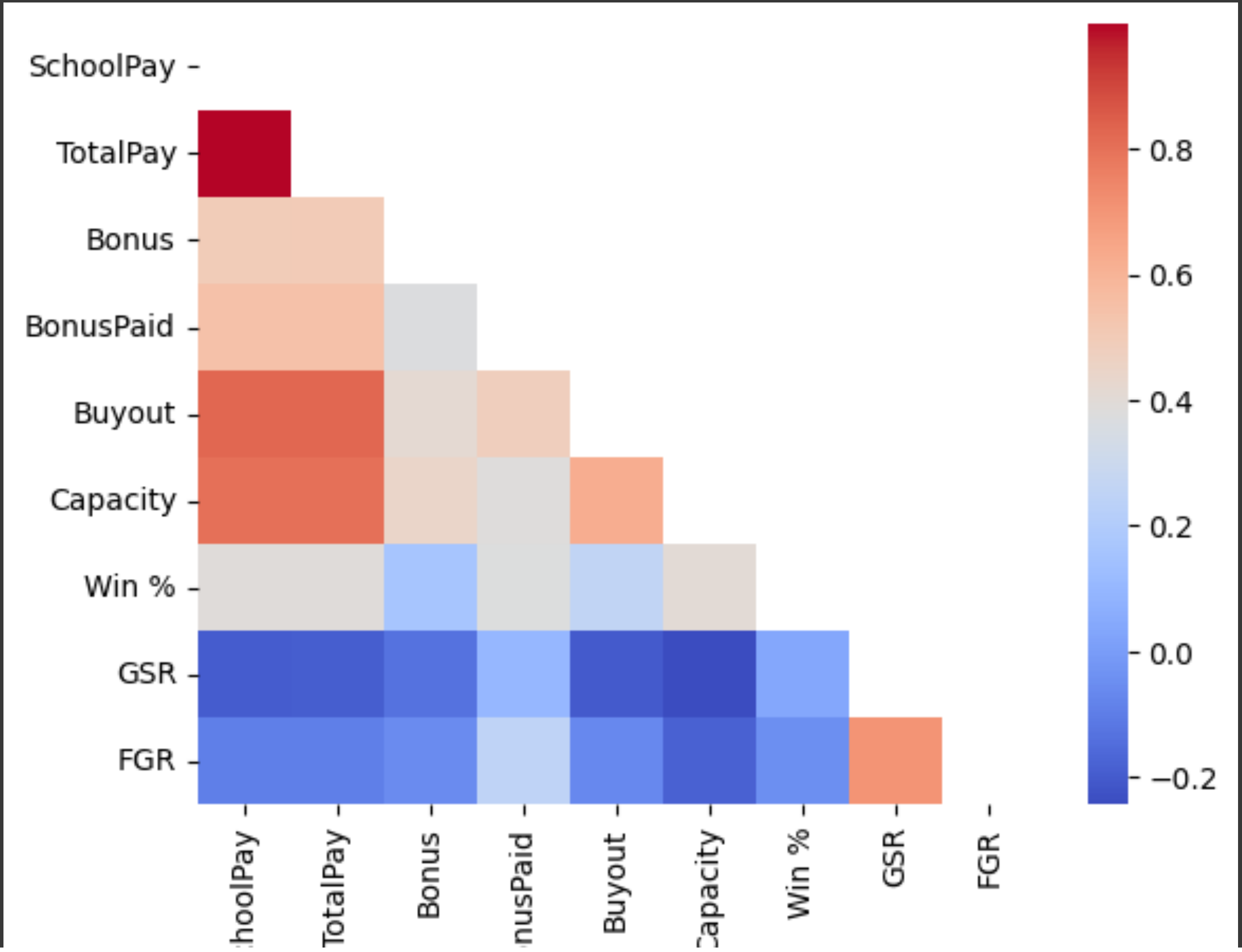
```
import numpy as np

# Explore the data
correlations = df.corr()

# Remove The AssistantPay column from the correlations DataFrame
correlations = correlations.drop('AssistantPay', axis=0).drop('AssistantPay', axis=

# Zero out the lower triangle of the matrix
mask = np.triu(np.ones_like(correlations, dtype=bool))
correlations = correlations.mask(mask)

correlations
```

```
<ipython-input-196-86c7bff2e180>:4: FutureWarning: The default value of numeri
  correlations = df.corr()
```

	SchoolPay	TotalPay	Bonus	BonusPaid	Buyout	Capacity	Win %
SchoolPay	NaN	NaN	NaN	NaN	NaN	NaN	NaN
TotalPay	0.999566	NaN	NaN	NaN	NaN	NaN	NaN
Bonus	0.496905	0.501814	NaN	NaN	NaN	NaN	NaN
BonusPaid	0.545730	0.547483	0.370240	NaN	NaN	NaN	NaN
Buyout	0.826087	0.828953	0.412215	0.483947	NaN	NaN	NaN
Capacity	0.802511	0.802756	0.444124	0.385656	0.622141	NaN	NaN
Win %	0.390897	0.390198	0.162289	0.373447	0.257871	0.406655	NaN
GSR	-0.197069	-0.193242	-0.133699	0.097298	-0.204554	-0.243810	0.033822
FGR	-0.094980	-0.094176	-0.057573	0.250936	-0.065918	-0.184339	-0.044788

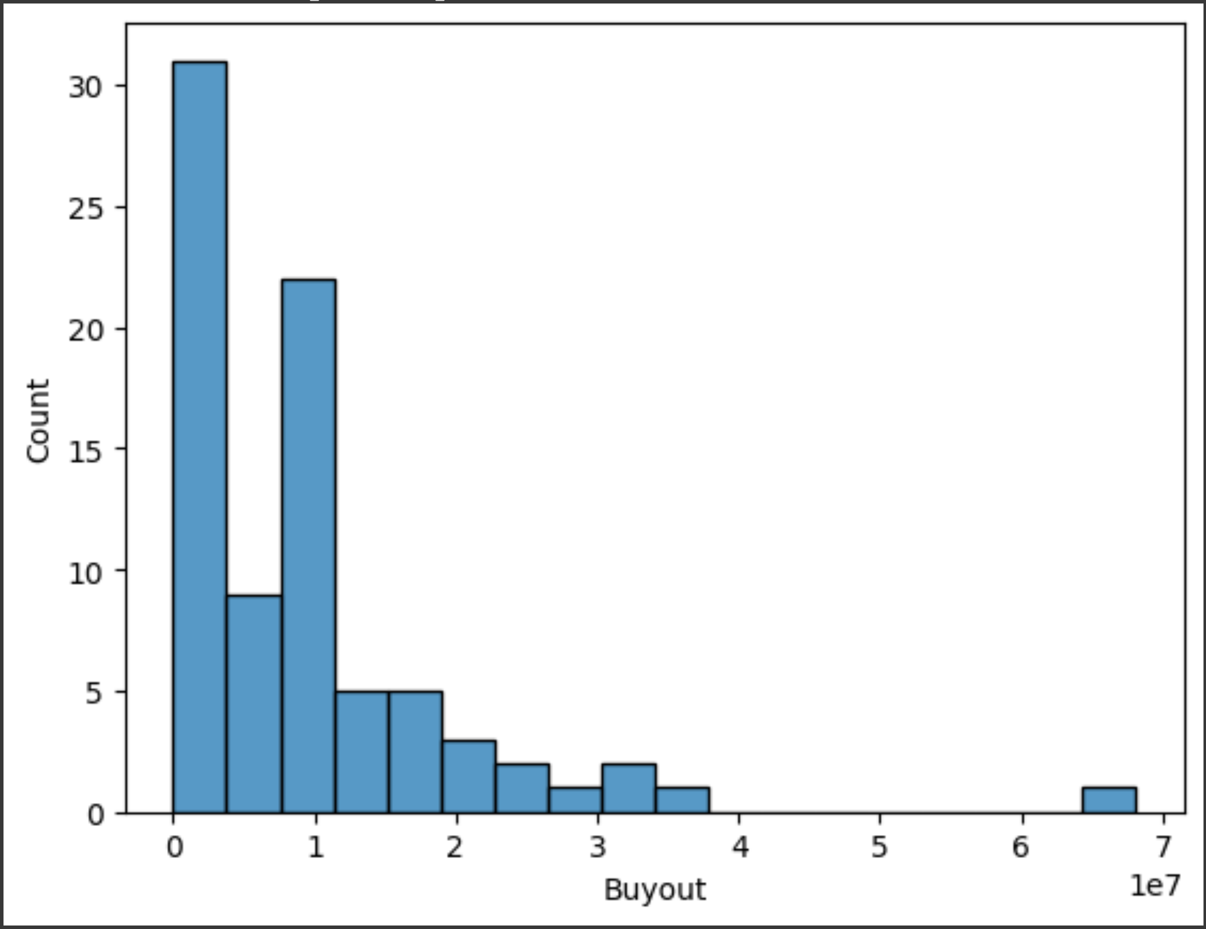
```
import seaborn as sns

sns.heatmap(correlations, cmap='coolwarm')

'''
Correlation Matrix shows the following are the most correlated with TotalPay (Ignor
    - Buyout
    - Capacity
    - Bonus
    - BonusPaid
    - Win %
'''
```

```
'\nCorrelation Matrix shows the following are the most correlated with TotalP
ay (Ignoring SchoolPay):\n    - Buyout\n    - Capacity\n    - Bonus\n    - Bo
nusPaid\n    - Win %\n'
```



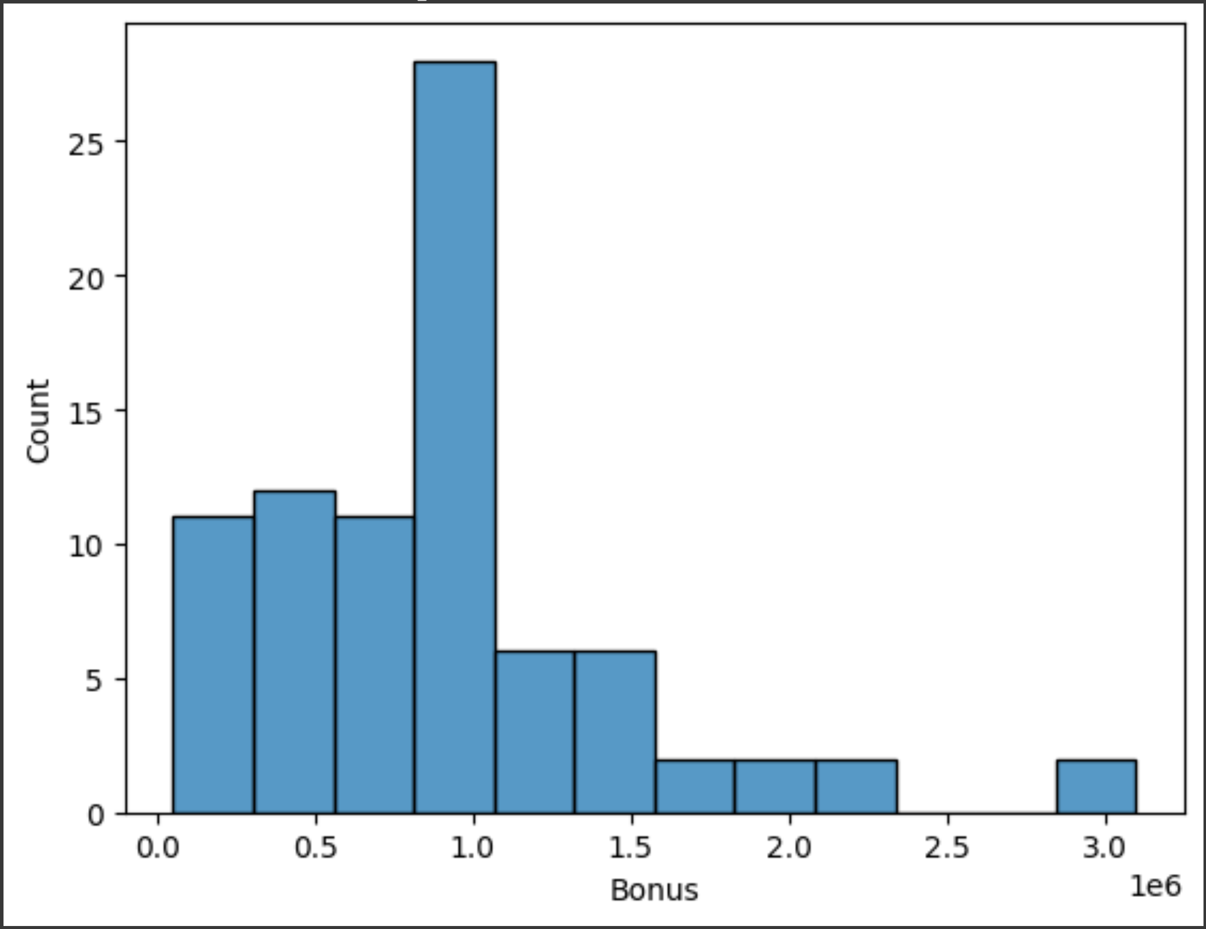
```
# View Distribution of of Buyout, Graduation Rate, and Donations
sns.histplot(df['Buyout'])
```

```
<Axes: xlabel='Buyout', ylabel='Count'>
```



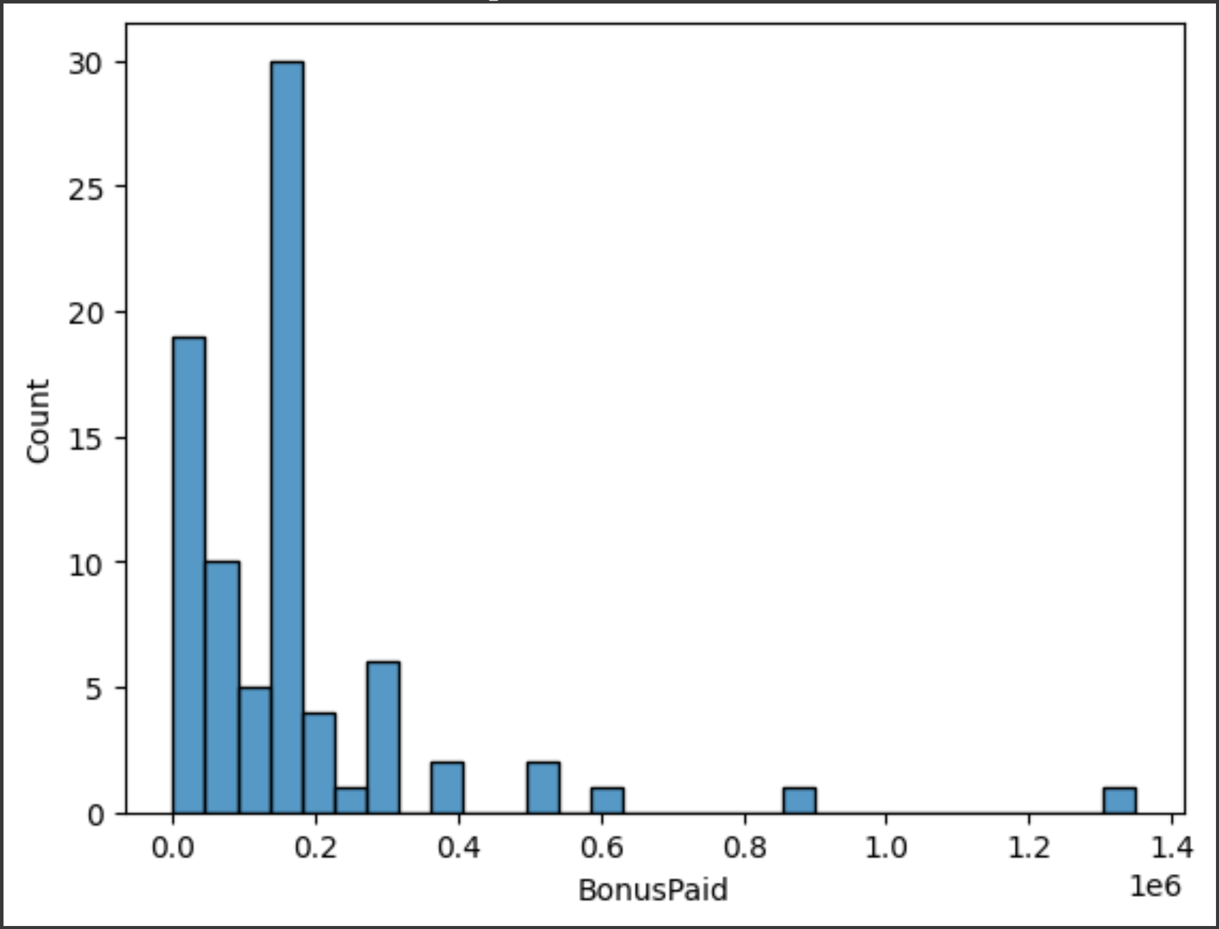
```
sns.histplot(df['Bonus'])
```

```
<Axes: xlabel='Bonus', ylabel='Count'>
```



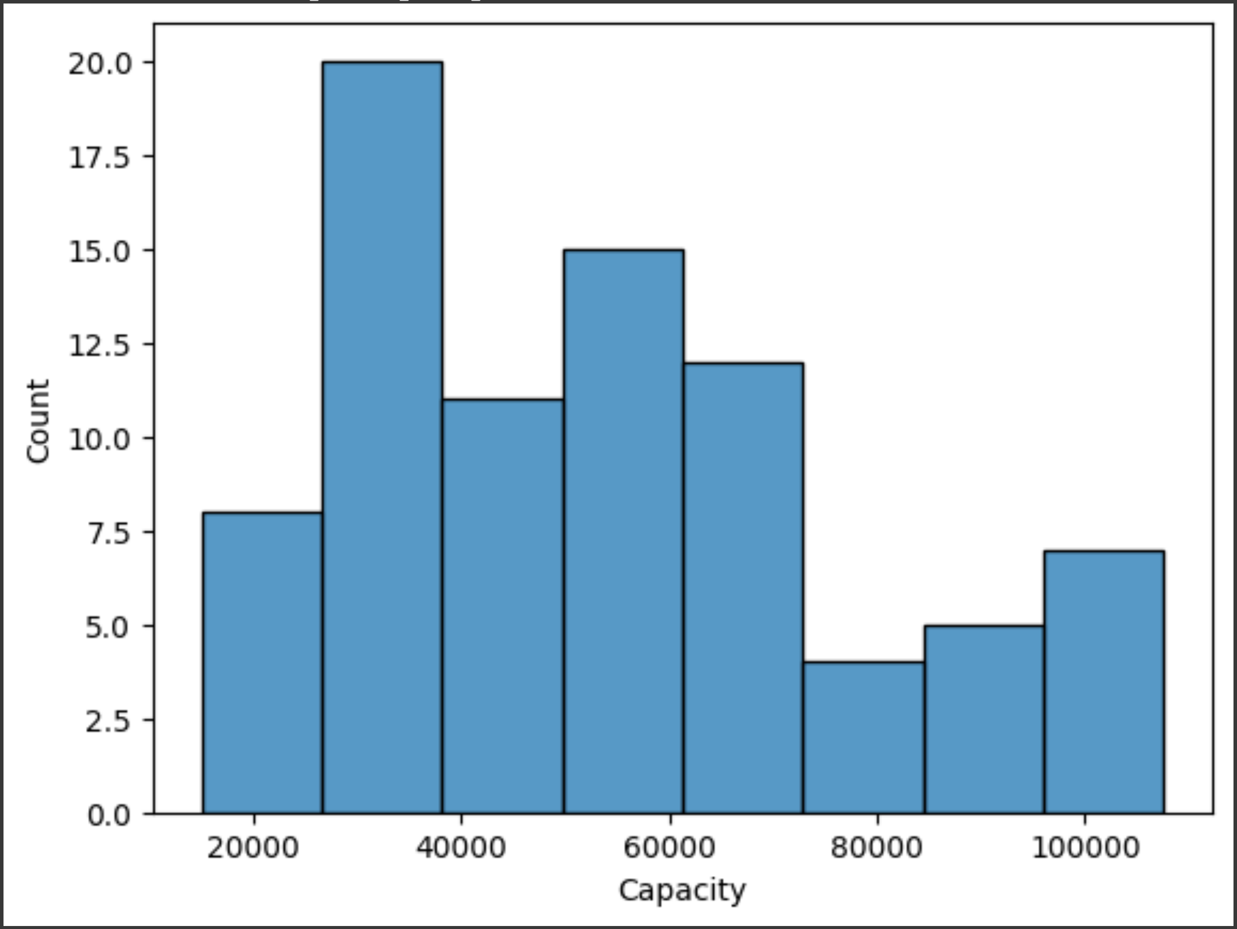
```
sns.histplot(df['BonusPaid'])
```

```
<Axes: xlabel='BonusPaid', ylabel='Count'>
```



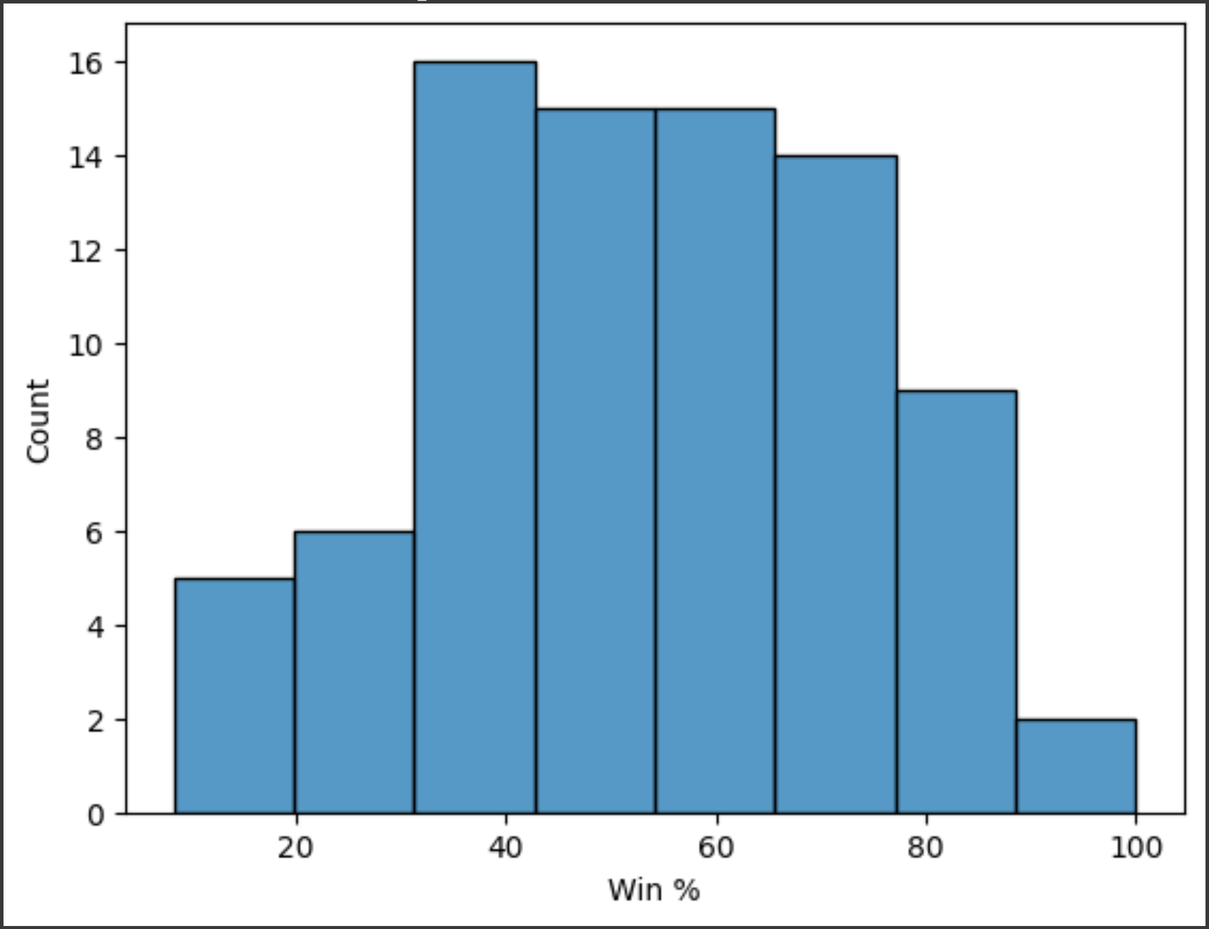
```
sns.histplot(df['Capacity'])
```

```
<Axes: xlabel='Capacity', ylabel='Count'>
```



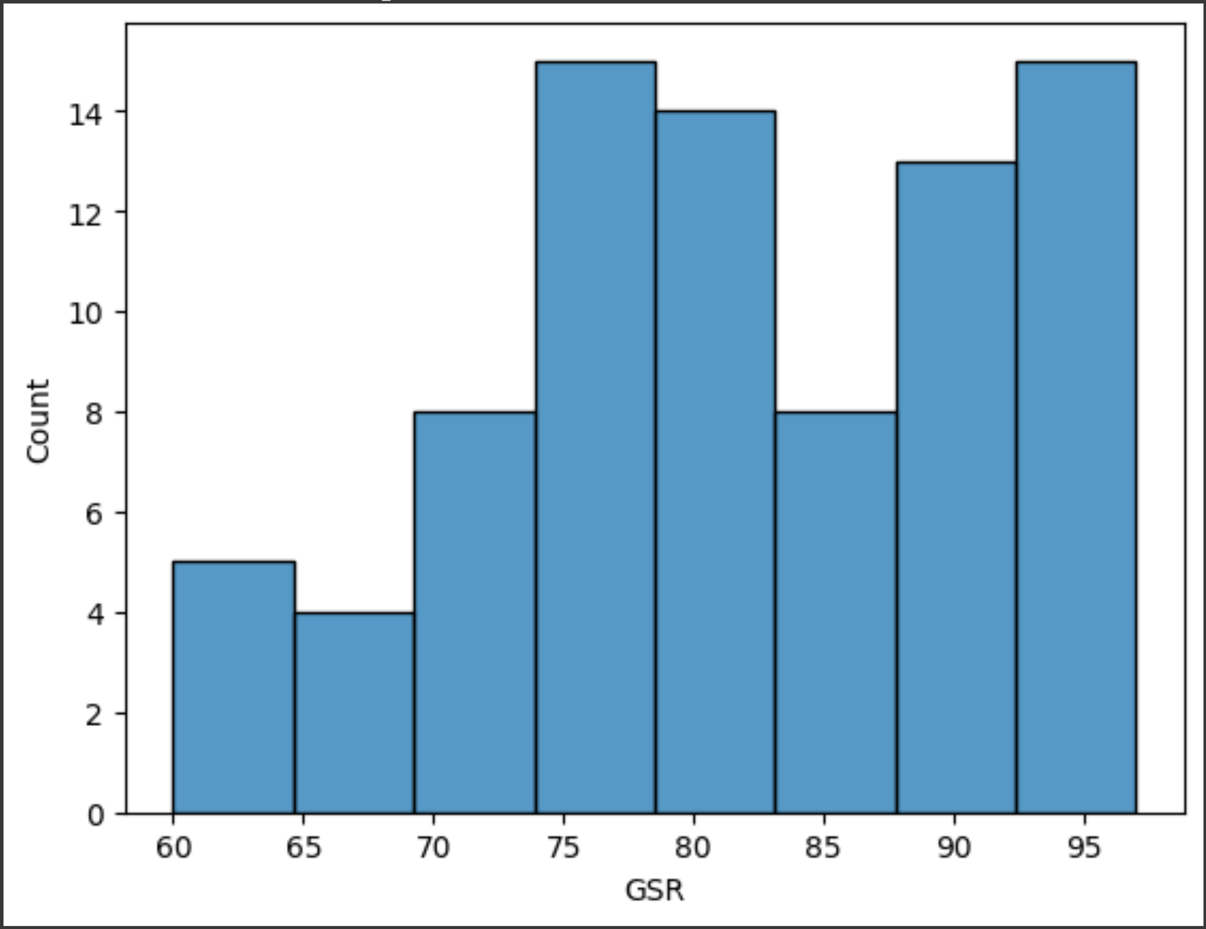
```
sns.histplot(df['Win %'])
```

```
<Axes: xlabel='Win %', ylabel='Count'>
```



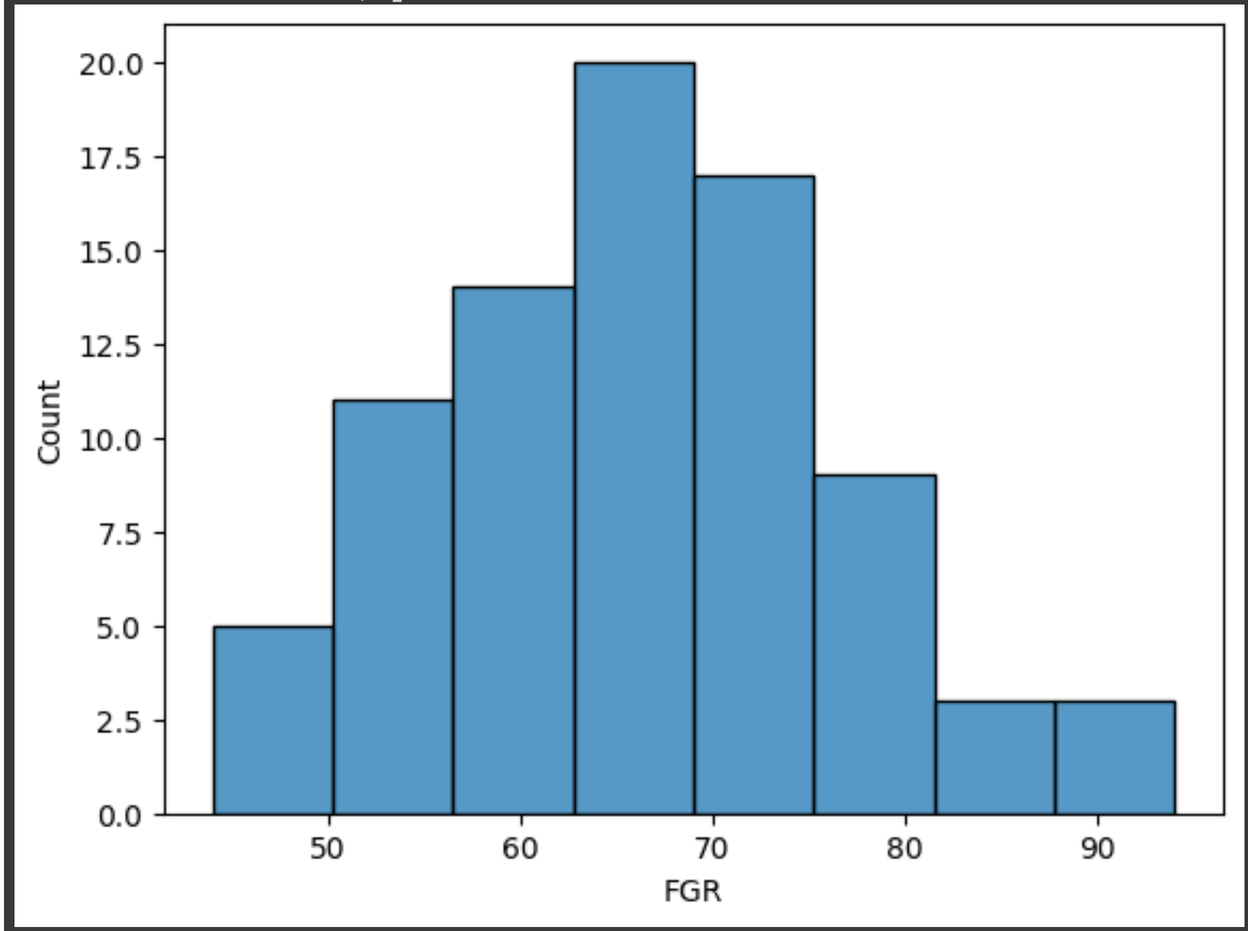
```
sns.histplot(df['GSR'])
```

```
<Axes: xlabel='GSR', ylabel='Count'>
```



```
sns.histplot(df['FGR'])
```

```
<Axes: xlabel='FGR', ylabel='Count'>
```



```
df.info()
```

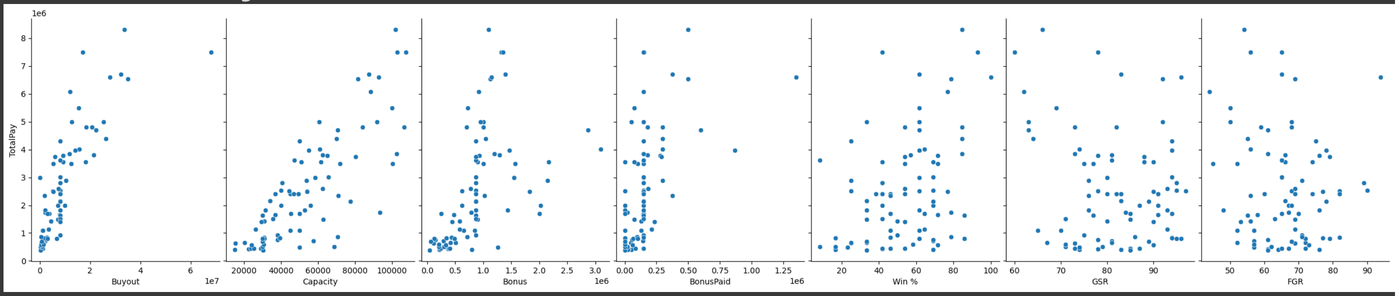
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 82 entries, 0 to 441
Data columns (total 15 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   School        82 non-null     object 
 1   Coach         82 non-null     object 
 2   SchoolPay     82 non-null     float64
 3   TotalPay      82 non-null     float64
 4   Bonus         82 non-null     float64
 5   BonusPaid     82 non-null     float64
 6   AssistantPay  82 non-null     float64
 7   Buyout        82 non-null     float64
 8   Capacity      82 non-null     float64
 9   Win %         82 non-null     float64
 10  Conference    82 non-null     object 
 11  Sport         82 non-null     object 
 12  State         82 non-null     object 
 13  GSR           82 non-null     int64  
 14  FGR           82 non-null     float64
dtypes: float64(9), int64(1), object(5)
memory usage: 10.2+ KB
```

```
print(df[['Buyout', 'Capacity', 'TotalPay','Bonus','BonusPaid','GSR','FGR']].isnull
print(df.columns.duplicated())
df_no_duplicates = df.drop_duplicates()
df_no_duplicates.info()
```

```
Buyout       0
Capacity     0
TotalPay     0
Bonus        0
BonusPaid    0
GSR          0
FGR          0
dtype: int64
[False False False False False False False False False False False False
 False False False]
<class 'pandas.core.frame.DataFrame'>
Int64Index: 82 entries, 0 to 441
Data columns (total 15 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   School        82 non-null     object 
 1   Coach         82 non-null     object 
 2   SchoolPay     82 non-null     float64
 3   TotalPay      82 non-null     float64
 4   Bonus         82 non-null     float64
 5   BonusPaid     82 non-null     float64
 6   AssistantPay  82 non-null     float64
 7   Buyout        82 non-null     float64
 8   Capacity      82 non-null     float64
 9   Win %         82 non-null     float64
 10  Conference    82 non-null     object 
 11  Sport         82 non-null     object 
 12  State         82 non-null     object 
 13  GSR           82 non-null     int64  
 14  FGR           82 non-null     float64
dtypes: float64(9), int64(1), object(5)
memory usage: 10.2+ KB
```

```
sns.pairplot(df_no_duplicates,
             x_vars=['Buyout', 'Capacity','Bonus','BonusPaid','Win %','GSR','FGR'],
             y_vars='TotalPay',
             height=5,
             aspect=0.7)
```

```
<seaborn.axisgrid.PairGrid at 0x7ffb9acc9c10>
```



Modeling (Linear Regression)

With the above data exploration findings in mind, a linear regression model will be fit with the following independenet variables: Buyout, Graduation Rate, and Donations. The predicting variable will be TotalPay.

```
# Model without log variables

X = df[['Buyout', 'Capacity','Bonus','BonusPaid','Win %','GSR','FGR']]
y = df['TotalPay']

# Add a constant term to the predictor variables (X)
X1 = sm.add_constant(X)

# Create the linear model and fit it to the data
model = sm.OLS(y, X).fit()

# Print the model summary
print(model.summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:               TotalPay   R-squared (uncentered):
Model:                            OLS   Adj. R-squared (uncentered):
Method:                 Least Squares   F-statistic:
Date:                Mon, 17 Apr 2023   Prob (F-statistic):
Time:                        22:48:04   Log-Likelihood:
No. Observations:                  82   AIC:
Df Residuals:                      75   BIC:
Df Model:                           7
Covariance Type:            nonrobust
=======================================================================================
                 coef    std err          t      P>|t|      [0.025      0.975
---------------------------------------------------------------------------------------
Buyout         0.0876      0.012      7.443      0.000       0.064       0.11
Capacity      31.6866      5.009      6.325      0.000      21.707      41.66
Bonus          0.2265      0.178      1.269      0.208      -0.129       0.58
BonusPaid      1.3174      0.573      2.298      0.024       0.176       2.45
Win %       3479.3999   5010.850      0.694      0.490   -6502.725    1.35e+0
GSR        -1664.6599   1.14e+04     -0.146      0.884   -2.44e+04     2.1e+0
FGR        -6119.9941   1.32e+04     -0.464      0.644   -3.24e+04    2.02e+0
=======================================================================================
Omnibus:                        9.985   Durbin-Watson:                         2.47
Prob(Omnibus):                  0.007   Jarque-Bera (JB):                     10.74
Skew:                           0.658   Prob(JB):                            0.0046
Kurtosis:                       4.189   Cond. No.                            2.69e+0
=======================================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not co
[2] Standard Errors assume that the covariance matrix of the errors is correc
[3] The condition number is large, 2.69e+06. This might indicate that there a
strong multicollinearity or other numerical problems.
```

QUESTION 1

- What is the predicted salary for Syracuse's next football coach?

To predict the recommended salary for Syracuse's next football coach, the predict() will be called on our model variable to estimate the salary:

```
# Create a dictionary of data for Syracuse
syracuse_data = {
    #'const': 1,
    'Buyout': np.mean(df['Buyout']),
    'Capacity': np.mean(df['Capacity']),
    'Bonus': np.mean(df['Bonus']),
    'BonusPaid': np.mean(df['BonusPaid']),
    'Win %': np.mean(df['Win %']),
    'GSR': np.mean(df['GSR']),
    'FGR': np.mean(df['FGR'])
}

# Convert the dictionary to a DataFrame
syracuse_df = pd.DataFrame(syracuse_data, index=[0])

# Predict the salary
predicted_salary = model.predict(syracuse_df)
formatted_salary = "${:,.2f}".format(round(predicted_salary[0], 2))
print(f"Predicted salary for Syracuse football coach: {formatted_salary}")
```

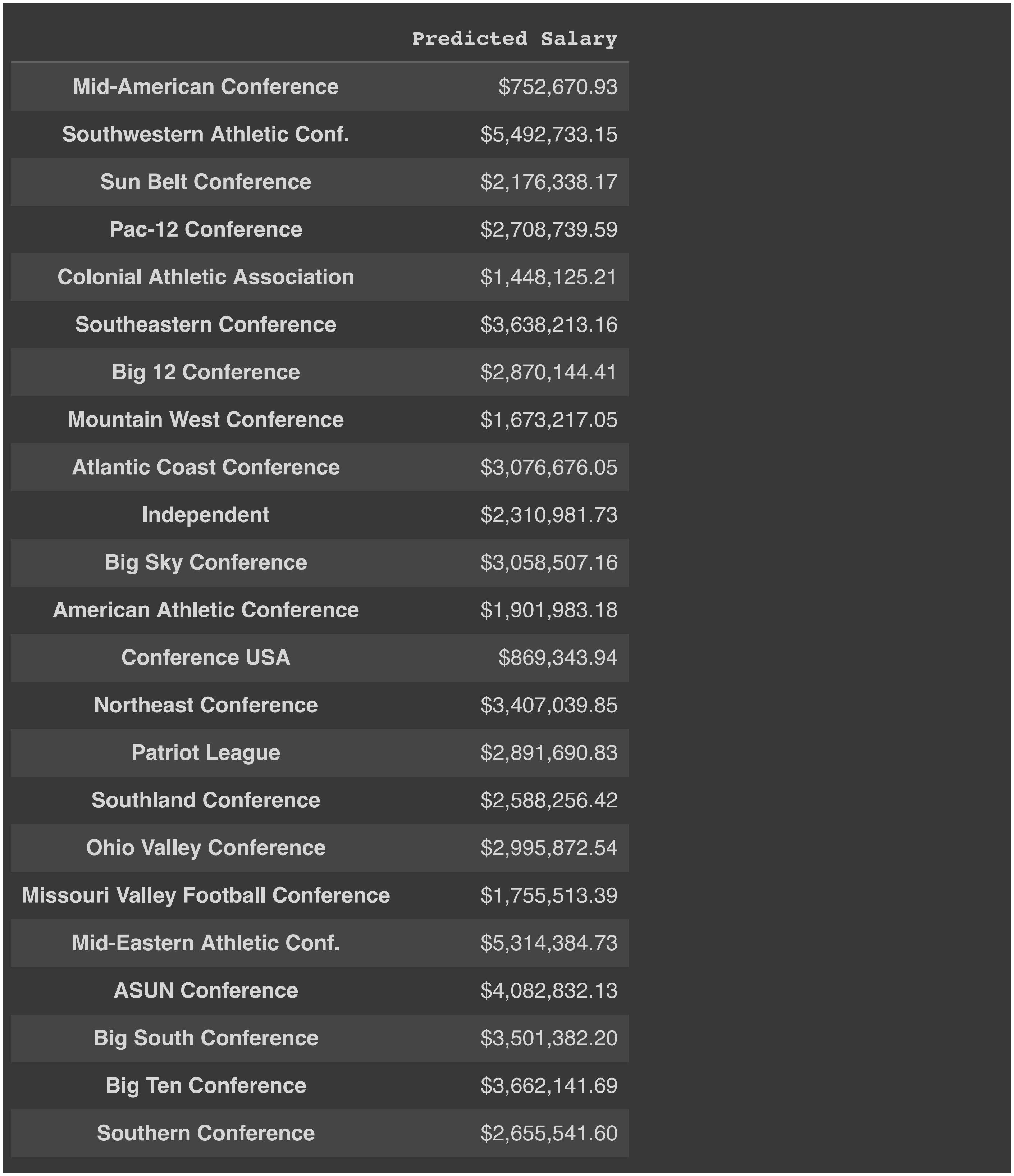
```
Predicted salary for Syracuse football coach: $2,559,590.59
```

QUESTION 2

- What would his salary be if we were still in the Big East? What if we went to the Big Ten?

To answer this question, we'll first calculate the average values for each independent variable in the model based on the conference (i.e., group by conference). Then, we'll use these averages to predict the coach's salary in different conferences.

```
# Create a function to predict the salary by conference
def predict_salary_by_conference(conference, df=df, model=model):
    # Filter the dataframe for the specific conference
    conference_df = df[df['Conference'] == conference]
    conference = conference
    # Calculate the average values of the independent variables for the conference
    conference_averages = {
        #'const': 1,
        'Buyout': conference_df['Buyout'].mean(),
        'Capacity': conference_df['Capacity'].mean(),
        'Bonus': conference_df['Bonus'].mean(),
        'BonusPaid': conference_df['BonusPaid'].mean(),
        'Win %': conference_df['Win %'].mean(),
        'GSR': np.mean(df['GSR']),
        'FGR': np.mean(df['FGR'])
    }

    # Convert the dictionary to a DataFrame
    conference_averages_df = pd.DataFrame(conference_averages, index=[0])

    # Predict the salary
    predicted_conference_salary = model.predict(conference_averages_df)
    formatted_conference_salary = "${:,.2f}".format(round(predicted_conference_sala

    return formatted_conference_salary

# Create a list of conferences
conferences = df['Conference'].unique()

# Create an empty dictionary to store the predicted salaries
predicted_salaries = {}

# Loop through each conference and predict the salary
for conference in conferences:
    predicted_salary = predict_salary_by_conference(conference)
    predicted_salaries[conference] = predicted_salary

# Convert the dictionary to a DataFrame
predicted_salaries_df = pd.DataFrame.from_dict(predicted_salaries, orient='index',

predicted_salaries_df
```



	Predicted Salary
Mid-American Conference	$752,670.93
Southwestern Athletic Conf.	$5,492,733.15
Sun Belt Conference	$2,176,338.17
Pac-12 Conference	$2,708,739.59
Colonial Athletic Association	$1,448,125.21
Southeastern Conference	$3,638,213.16
Big 12 Conference	$2,870,144.41
Mountain West Conference	$1,673,217.05
Atlantic Coast Conference	$3,076,676.05
Independent	$2,310,981.73
Big Sky Conference	$3,058,507.16
American Athletic Conference	$1,901,983.18
Conference USA	$869,343.94
Northeast Conference	$3,407,039.85
Patriot League	$2,891,690.83
Southland Conference	$2,588,256.42
Ohio Valley Conference	$2,995,872.54
Missouri Valley Football Conference	$1,755,513.39
Mid-Eastern Athletic Conf.	$5,314,384.73
ASUN Conference	$4,082,832.13
Big South Conference	$3,501,382.20
Big Ten Conference	$3,662,141.69
Southern Conference	$2,655,541.60

```
def print_salary(conference, predicted_salaries_df=predicted_salaries_df):
    print(f"Predicted average salary for {conference} football coach: {predicted_sa

print_salary('Big Ten Conference')
print_salary('Atlantic Coast Conference')
```

```
Predicted average salary for Big Ten Conference football coach: $3,662,141.69
Predicted average salary for Atlantic Coast Conference football coach: $3,076,
```

QUESTION 3

- What schools did we drop from our data and why?

All in all 47 schools were dropped either because of NaN values, or enough dissimilarity that the `fuzzywuzzy` python package library could not detect enough similarity to merge into the main `df`. Additional measures that could be sought after next time would be to manually ensure the names of the schools are the same (i.e., go cell by cell and make manual changes to the spelling and/or abbreviations of each school name).

For the purposes of this assignment, the `fuzzywuzzy` match was deemed to be satisfactory for this analysis

```python
print(len(dropped_schools))
dropped_schools
```

```
47
{'Air Force',
 'Alabama at Birmingham',
 'Arizona State',
 'Arkansas State',
 'Colorado State',
 'Eastern Michigan',
 'Florida Atlantic',
 'Florida International',
 'Florida State',
 'Fresno State',
 'Georgia Southern',
 'Georgia State',
 'Georgia Tech',
 'Iowa State',
 'Kansas State',
 'Liberty',
 'Louisiana Tech',
 'Miami (Fla.)',
 'Miami (Ohio)',
 'Michigan State',
 'Mississippi State',
 'Nevada-Las Vegas',
 'New Mexico State',
 'North Carolina',
 'North Carolina State',
 'Northern Illinois',
 'Ohio State',
 'Oklahoma State',
 'Oregon State',
 'Rutgers',
 'South Alabama',
 'South Carolina',
 'South Florida',
 'Southern California',
 'Southern Methodist',
 'Southern Mississippi',
 'Texas Christian',
 'Texas Tech',
 'Texas-El Paso',
 'Texas-San Antonio',
 'UCLA',
 'Utah State',
 'Virginia Tech',
 'Washington State',
 'West Virginia',
 'Western Kentucky',
 'Western Michigan'}
```

QUESTION 4

- What effect does graduation rate have on the projected salary?

To answer this question, viewing the output of the linear regression model will be required.

The graduation rate coefficient is divided into two statistics: GSR, FGR. Both appear to have a p-value of well above the threshold of 0.05 (GSR = 0.884; FGR = 0.644). This means we cannot determine that the impact is anything different than zero and the results are not statistically significant.

```
print(model.summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:               TotalPay   R-squared (uncentered):
Model:                            OLS   Adj. R-squared (uncentered):
Method:                 Least Squares   F-statistic:
Date:                Mon, 17 Apr 2023   Prob (F-statistic):
Time:                        22:48:15   Log-Likelihood:
No. Observations:                  82   AIC:
Df Residuals:                      75   BIC:
Df Model:                           7
Covariance Type:            nonrobust
=======================================================================================
                 coef    std err          t      P>|t|      [0.025      0.975
---------------------------------------------------------------------------------------
Buyout         0.0876      0.012      7.443      0.000       0.064       0.11
Capacity      31.6866      5.009      6.325      0.000      21.707      41.66
Bonus          0.2265      0.178      1.269      0.208      -0.129       0.58
BonusPaid      1.3174      0.573      2.298      0.024       0.176       2.45
Win %       3479.3999   5010.850      0.694      0.490   -6502.725    1.35e+0
GSR        -1664.6599   1.14e+04     -0.146      0.884   -2.44e+04     2.1e+0
FGR        -6119.9941   1.32e+04     -0.464      0.644   -3.24e+04    2.02e+0
=======================================================================================
Omnibus:                        9.985   Durbin-Watson:                        2.47
Prob(Omnibus):                  0.007   Jarque-Bera (JB):                    10.74
Skew:                           0.658   Prob(JB):                           0.0046
Kurtosis:                       4.189   Cond. No.                           2.69e+0
=======================================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not co
[2] Standard Errors assume that the covariance matrix of the errors is correc
[3] The condition number is large, 2.69e+06. This might indicate that there a
strong multicollinearity or other numerical problems.
```

Double-click (or enter) to edit

QUESTION 5

- How good is our model?

To answer this question, it will be necessary to again review the model summary.

When first reviewing the output of a linear regression model, the first step is to evaluate the p-value of the f-statistic itself. Since this appears to be well under the standard 0.05 threshold (1.22e-43), it can be determiend that the model can be interpreted and is statistically significant.

It's then advisable to move onto the R-squared value. It's seen that the value for this is 0.942, which tells the reader that ~94.2 of the change in our Y variable (TotalPay) is explained by the change in our independent (X) variables -- which is 'Buyout', 'Capacity', 'Bonus', 'BonusPaid', 'Win %', 'FGR', and 'GSR'.

The coefficients for these varialbes, as well as their respective p-values, can be viewed in the output as well.

The only p-values for the indepdnent variables that look to be under the 0.05 threshold is Buyout, Capacity, and BonusPaid, which means that they are statistically significant to the model.

```
print(model.summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:               TotalPay   R-squared (uncentered):
Model:                            OLS   Adj. R-squared (uncentered):
Method:                 Least Squares   F-statistic:
Date:                Mon, 17 Apr 2023   Prob (F-statistic):
Time:                        22:48:17   Log-Likelihood:
No. Observations:                  82   AIC:
Df Residuals:                      75   BIC:
Df Model:                           7
Covariance Type:            nonrobust
=======================================================================================
                 coef    std err          t      P>|t|      [0.025      0.975
---------------------------------------------------------------------------------------
Buyout         0.0876      0.012      7.443      0.000       0.064       0.11
Capacity      31.6866      5.009      6.325      0.000      21.707      41.66
Bonus          0.2265      0.178      1.269      0.208      -0.129       0.58
BonusPaid      1.3174      0.573      2.298      0.024       0.176       2.45
Win %       3479.3999   5010.850      0.694      0.490   -6502.725    1.35e+0
GSR        -1664.6599   1.14e+04     -0.146      0.884   -2.44e+04     2.1e+0
FGR        -6119.9941   1.32e+04     -0.464      0.644   -3.24e+04    2.02e+0
=======================================================================================
Omnibus:                        9.985   Durbin-Watson:                         2.47
Prob(Omnibus):                  0.007   Jarque-Bera (JB):                     10.74
Skew:                           0.658   Prob(JB):                            0.0046
Kurtosis:                       4.189   Cond. No.                            2.69e+0
=======================================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not co
[2] Standard Errors assume that the covariance matrix of the errors is correc
[3] The condition number is large, 2.69e+06. This might indicate that there a
strong multicollinearity or other numerical problems.
```

QUESTION 6

- What is the single biggest impact on salary size?

The single biggest statistically significant impact on salary size is more than likely due to Stadium capcity. For every 1 unit increase in stadium capcity (i.e., for each person added), the coaches TotalPaid salary increases by ~ $31

```
coefficients = model.params
coefficients
```

```
Buyout          0.087625
Capacity       31.686578
Bonus           0.226475
BonusPaid       1.317356
Win %        3479.399912
GSR         -1664.659910
FGR         -6119.994080
dtype: float64
```

```

```