lab1_playground_2

April 1, 2023

Python 3.9.6 (default, Dec 22 2022, 02:58:32)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.11.0 – An enhanced Interactive Python. Type '?' for help.

[ ]:
```
%%markdown

# IST 718
## Laboratory Exercise - 1
```

1 IST 718

1.1 Laboratory Exercise - 1

[ ]:
```
%%markdown

### Import Packages/Dependencies
```

1.1.1 Import Packages/Dependencies

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

[ ]:
```
%%markdown

### Data Import and Cleaning

The 'Coaches9.csv' is read into a Pandas DataFrame. The DataFrame is then␣
↪cleaned by replacing '--' with 0, dropping rows with missing values, and␣
↪replacing commas with empty strings.

The data is then merged with the 'grades.csv' and 'donations_with_conf.csv'␣
↪files to form a single DataFrame.

The output is an overview of the DataFrame via the '.info()' method as well as␣
↪a preview of the DataFrame via the '.head()' method.
```

1.1.2 Data Import and Cleaning

The 'Coaches9.csv' is read into a Pandas DataFrame. The DataFrame is then cleaned by replacing '–' with 0, dropping rows with missing values, and replacing commas with empty strings.

The data is then merged with the 'grades.csv' and 'donations_with_conf.csv' files to form a single DataFrame.

The output is an overview of the DataFrame via the '.info()' method as well as a preview of the DataFrame via the '.head()' method.

[ ]:
```
# Read data
generic_df = pd.read_csv('/Users/pergolicious/Library/CloudStorage/
↪OneDrive-SyracuseUniversity/Syracuse University/Courses/IST 718/Labs/Lab 1/
↪IST_718-master/Coaches9.csv')
coaches_df = pd.read_csv('/Users/pergolicious/Library/CloudStorage/
↪OneDrive-SyracuseUniversity/Syracuse University/Courses/IST 718/Labs/Lab 1/
↪IST_718-master/Coaches9.csv')
grades_csv = pd.read_csv('grades.csv').drop('index', axis=1)
donations_csv = pd.read_csv('donations_with_conf.csv').drop('index', axis=1)

# Clean coaches_df
coaches_df.replace('--', 0, inplace=True)
coaches_df = coaches_df.dropna()
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',␣
↪'Buyout']
for col in money_columns:
    coaches_df[col] = coaches_df[col].str.replace('[^0-9.]', '', regex=True).
↪astype(float)

# Merge data
coaches_df = coaches_df.merge(grades_csv, on='School')
coaches_df = coaches_df.merge(donations_csv, on='School')

# Clean merged data
#coaches_df = coaches_df.drop(['Conference_y', 'Donations (in millions)_x'],␣
↪axis=1)
coaches_df = coaches_df.drop(['Conference_y'], axis=1)
coaches_df = coaches_df.rename(columns={'Conference_x': 'Conference',␣
↪'Donations (in millions)_y': 'Donations (in millions)'})

# Syracuse has NAN values for Bonus, BonusPaid, and Buyout columns.  Remove␣
↪from dataset?
coaches_df = coaches_df.dropna()
coaches_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 87 entries, 0 to 87
Data columns (total 11 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   School                  87 non-null     object
 1   Conference              87 non-null     object
 2   Coach                   87 non-null     object
 3   SchoolPay               87 non-null     float64
 4   TotalPay                87 non-null     float64
 5   Bonus                   87 non-null     float64
 6   BonusPaid               87 non-null     float64
 7   AssistantPay            87 non-null     float64
 8   Buyout                  87 non-null     float64
 9   Graduation Rate (%)     87 non-null     int64
 10  Donations (in millions) 87 non-null     int64
dtypes: float64(6), int64(2), object(3)
memory usage: 8.2+ KB
```

[ ]:
```
coaches_df.head()
```

[ ]:
```
                  School Conference              Coach  SchoolPay   TotalPay  \
0                  Akron        MAC       Terry Bowden   411000.0   412500.0
1                Alabama        SEC         Nick Saban  8307000.0  8307000.0
2  Alabama at Birmingham      C-USA         Bill Clark   900000.0   900000.0
3      Appalachian State   Sun Belt  Scott Satterfield   712500.0   712500.0
4         Arkansas State   Sun Belt     Blake Anderson   825000.0   825000.0

       Bonus  BonusPaid  AssistantPay      Buyout  Graduation Rate (%)  \
0   225000.0    50000.0           0.0    688500.0                   65
1  1100000.0   500000.0           0.0  33600000.0                   77
2   950000.0   165471.0           0.0   3847500.0                   60
3   295000.0   145000.0           0.0   2160417.0                   82
4   185000.0    25000.0           0.0    300000.0                   68

   Donations (in millions)
0                       15
1                       45
2                       10
3                        5
4                        7
```

[ ]:
```
%%markdown

### Data Exploration
```

```
The following code block explores the data by calculating the correlation matrix and plotting the correlation matrix as a heatmap.

The overall findings are that the most correlated variables with TotalPay are Buyout, Graduation Rate, and Donations.  The least correlated variables are Bonus, BonusPaid, and AssistantPay.
```

1.1.3 Data Exploration

The following code block explores the data by calculating the correlation matrix and plotting the correlation matrix as a heatmap.

The overall findings are that the most correlated variables with TotalPay are Buyout, Graduation Rate, and Donations. The least correlated variables are Bonus, BonusPaid, and AssistantPay.

[ ]:

```python
# Explore the data
correlations = coaches_df.corr()

# Remove The AssistantPay column from the correlations DataFrame
correlations = correlations.drop('AssistantPay', axis=0
                                 ).drop('AssistantPay', axis=1)
correlations['TotalPay']
correlations
```

```
<ipython-input-8-4cb110aa4184>:3: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlations = coaches_df.corr()
```

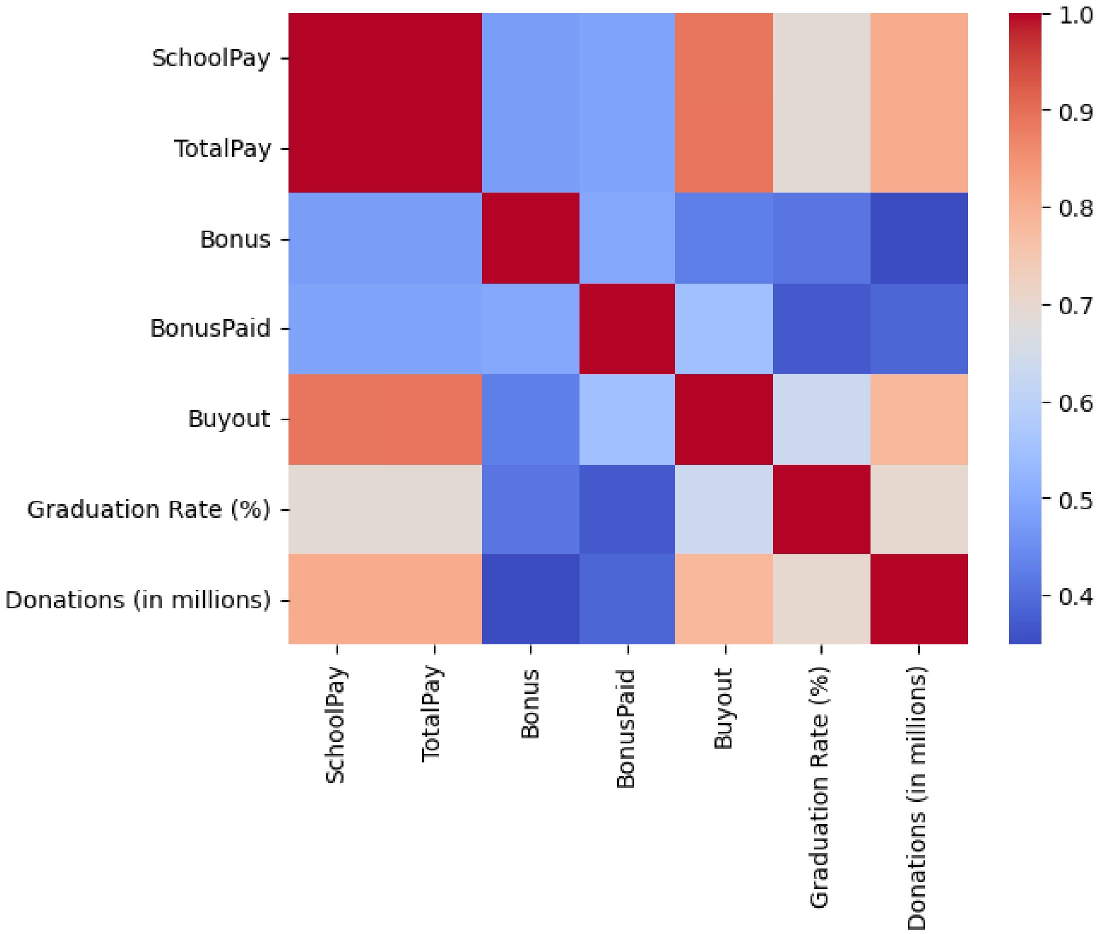
[ ]:

	SchoolPay	TotalPay	Bonus	BonusPaid	Buyout	\
SchoolPay	1.000000	0.999836	0.476402	0.489034	0.890432	
TotalPay	0.999836	1.000000	0.476084	0.489863	0.892823	
Bonus	0.476402	0.476084	1.000000	0.496681	0.425647	
BonusPaid	0.489034	0.489863	0.496681	1.000000	0.547452	
Buyout	0.890432	0.892823	0.425647	0.547452	1.000000	
Graduation Rate (%)	0.691090	0.691689	0.409931	0.369707	0.632668	
Donations (in millions)	0.804861	0.805701	0.348452	0.387060	0.782478	

	Graduation Rate (%)	Donations (in millions)
SchoolPay	0.691090	0.804861
TotalPay	0.691689	0.805701
Bonus	0.409931	0.348452
BonusPaid	0.369707	0.387060
Buyout	0.632668	0.782478
Graduation Rate (%)	1.000000	0.699119
Donations (in millions)	0.699119	1.000000

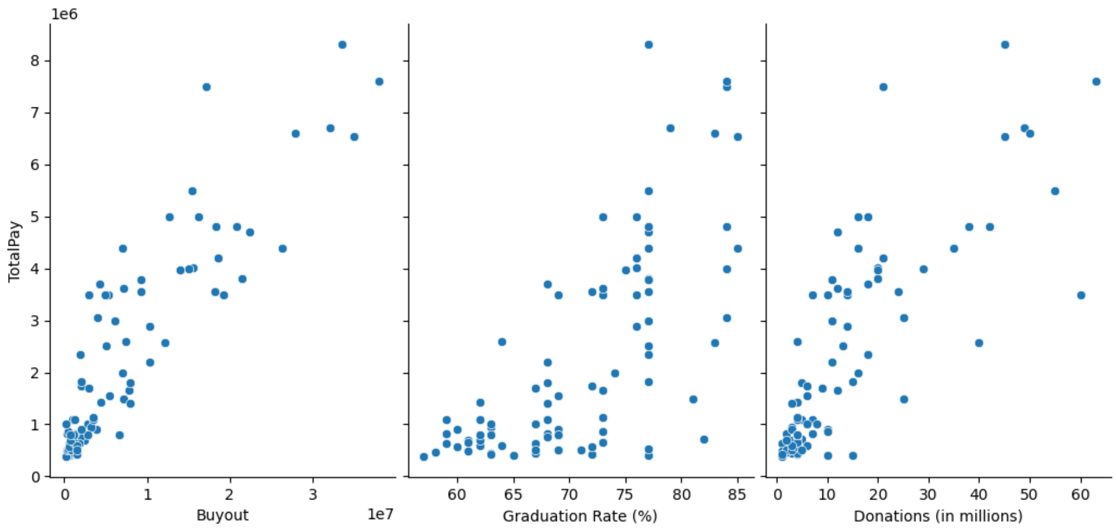
[ ]:
```python
import seaborn as sns

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

'''
Correlation Matrix shows the following are the most correlated with TotalPay␣
↪(Ignoring SchoolPay):
    - Buyout
    - Graduation Rate
    - Donations

'''
```

[ ]:
```
'\nCorrelation Matrix shows the following are the most correlated with TotalPay
(Ignoring SchoolPay):\n    - Buyout\n    - Graduation Rate\n    - Donations\n\n'
```



```
[ ]: sns.pairplot(coaches_df,
                 x_vars=['Buyout', 'Graduation Rate (%)', 'Donations (in␣
     ↪millions)'],
                 y_vars='TotalPay',
                 height=5,
                 aspect=0.7)
```

```
[ ]: <seaborn.axisgrid.PairGrid at 0x29c282070>
```




```
[ ]: %%markdown

     ### 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.
```

1.1.4 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.

```
[ ]: # Prepare the data
     # X = coaches_df[['Bonus', 'BonusPaid', 'AssistantPay', 'Buyout', 'Graduation␣
     ↪Rate (%)', 'Donations (in millions)']]
     X = coaches_df[['Buyout', 'Graduation Rate (%)', 'Donations (in millions)']]
     y = coaches_df['TotalPay']
     y = coaches_df['TotalPay']
```

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

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

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

[ ]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:               TotalPay   R-squared:                       0.837
Model:                            OLS   Adj. R-squared:                  0.831
Method:                 Least Squares   F-statistic:                     141.9
Date:                Sat, 01 Apr 2023   Prob (F-statistic):           1.43e-32
Time:                        13:00:41   Log-Likelihood:                -1306.0
No. Observations:                  87   AIC:                             2620.
Df Residuals:                      83   BIC:                             2630.
Df Model:                           3
Covariance Type:            nonrobust
================================================================================
===========
                              coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
-----------
const                   -1.852e+06   1.11e+06     -1.666      0.100   -4.06e+06
3.59e+05
Buyout                      0.1422      0.016      8.890      0.000       0.110
0.174
Graduation Rate (%)      3.792e+04   1.67e+04      2.265      0.026    4618.651
7.12e+04
Donations (in millions)  2.715e+04   1.06e+04      2.555      0.012    6019.114
4.83e+04
==============================================================================
Omnibus:                       24.047   Durbin-Watson:                   1.845
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               38.355
Skew:                           1.144   Prob(JB):                     4.69e-09
Kurtosis:                       5.313   Cond. No.                     1.49e+08
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

```
[2] The condition number is large, 1.49e+08. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[ ]:
```
%%markdown

### 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:
```

1.1.5 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(coaches_df['Buyout']),
    'Graduation Rate (%)': np.mean(coaches_df['Graduation Rate (%)']),
    'Donations (in millions)': np.mean(coaches_df['Donations (in millions)'])
}

# 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: $2,303,989.60
```

[ ]:
```
%%markdown
### 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, you can first calculate the average values for each
↪independent variable in the model based on the conference. Then, you can use
↪these averages to predict the coach's salary in different conferences.
```

1.1.6 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, you can first calculate the average values for each independent variable in the model based on the conference. Then, you can 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, coaches_df=coaches_df,␣
↪model=model):
    # Filter the dataframe for the specific conference
    conference_df = coaches_df[coaches_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(),
        'Graduation Rate (%)': conference_df['Graduation Rate (%)'].mean(),
        'Donations (in millions)': conference_df['Donations (in millions)'].
↪mean()
    }

    # 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_salary[0], 2))

    return formatted_conference_salary

# Create a list of conferences
conferences = coaches_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', columns=['Predicted Salary'])
```

```
predicted_salaries_df
```

[ ]:

```
           Predicted Salary
MAC           $1,026,513.07
SEC           $4,859,856.42
C-USA         $1,080,383.80
Sun Belt        $977,113.31
Mt. West      $1,290,274.89
Pac-12        $3,287,019.00
AAC           $1,355,372.37
ACC           $4,085,613.08
Big Ten       $3,796,105.94
Big 12        $3,061,884.01
Ind.            $860,087.82
```

[ ]:

```
def print_salary(conference, predicted_salaries_df=predicted_salaries_df):
    print(f"Predicted average salary for {conference} football coach:␣
 ↪{predicted_salaries_df.loc[conference]['Predicted Salary']}")

print_salary('Big Ten')
print_salary('ACC')
```

```
Predicted average salary for Big Ten football coach: $3,796,105.94
Predicted average salary for ACC football coach: $4,085,613.08
```

[ ]:

```
%%markdown

### QUESTION 3
- What schools did we drop from our data and why?

Overall, 42 schools were dropped:
- 'Air Force',
- 'Arizona',
- 'Arizona State',
- 'Arkansas',
- 'Army',
- 'Baylor',
- 'Boston College',
- 'Brigham Young',
- 'Central Florida',
- 'Duke',
- 'Florida',
- 'Florida State',
- 'Georgia Southern',
- 'Kent State',
- 'Liberty',
```

```
- 'Louisiana-Lafayette',
- 'Miami (Fla.)',
- 'Mississippi',
- 'Mississippi State',
- 'Navy',
- 'Nebraska',
- 'Northwestern',
- 'Notre Dame',
- 'Oregon',
- 'Oregon State',
- 'Pittsburgh',
- 'Rice',
- 'South Alabama',
- 'Southern California',
- 'Southern Methodist',
- 'Stanford',
- 'Syracuse',
- 'Tennessee',
- 'Texas A&M',
- 'Texas Christian',
- 'Texas-El Paso',
- 'Tulane',
- 'Tulsa',
- 'UCLA',
- 'Vanderbilt',
- 'Wake Forest',
- 'Wisconsin'

The schools were dropped from the state becasue they had '--' values that were
 ↪turned to NaN via the data cleansing process.
```

1.1.7 QUESTION 3

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

Overall, 42 schools were dropped: - 'Air Force', - 'Arizona', - 'Arizona State', - 'Arkansas', - 'Army', - 'Baylor', - 'Boston College', - 'Brigham Young', - 'Central Florida', - 'Duke', - 'Florida', - 'Florida State', - 'Georgia Southern', - 'Kent State', - 'Liberty', - 'Louisiana-Lafayette', - 'Miami (Fla.)', - 'Mississippi', - 'Mississippi State', - 'Navy', - 'Nebraska', - 'Northwestern', - 'Notre Dame', - 'Oregon', - 'Oregon State', - 'Pittsburgh', - 'Rice', - 'South Alabama', - 'Southern California', - 'Southern Methodist', - 'Stanford', - 'Syracuse', - 'Tennessee', - 'Texas A&M', - 'Texas Christian', - 'Texas-El Paso', - 'Tulane', - 'Tulsa', - 'UCLA', - 'Vanderbilt', - 'Wake Forest', - 'Wisconsin'

The schools were dropped from the state becasue they had '–' values that were turned to NaN via the data cleansing process.

```
[ ]: # Create dropped schools data frame
```

```python
generic_df = pd.read_csv('/Users/pergolicious/Library/CloudStorage/
↪OneDrive-SyracuseUniversity/Syracuse University/Courses/IST 718/Labs/Lab 1/
↪IST_718-master/Coaches9.csv')
generic_df.replace('--', np.nan, inplace=True)
null = generic_df[generic_df.isnull().any(axis=1)]
null.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 42 entries, 0 to 127
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   School        42 non-null     object
 1   Conference    42 non-null     object
 2   Coach         42 non-null     object
 3   SchoolPay     38 non-null     object
 4   TotalPay      38 non-null     object
 5   Bonus         20 non-null     object
 6   BonusPaid     1 non-null      object
 7   AssistantPay  42 non-null     object
 8   Buyout        20 non-null     object
dtypes: object(9)
memory usage: 3.3+ KB
```

[ ]:
```python
# Identify dropped schools
null_schools = null['School'].unique().tolist()
null_schools
```

[ ]:
```
['Air Force',
 'Arizona',
 'Arizona State',
 'Arkansas',
 'Army',
 'Baylor',
 'Boston College',
 'Brigham Young',
 'Central Florida',
 'Duke',
 'Florida',
 'Florida State',
 'Georgia Southern',
 'Kent State',
 'Liberty',
 'Louisiana-Lafayette',
 'Miami (Fla.)',
 'Mississippi',
 'Mississippi State',
```

```
 'Navy',
 'Nebraska',
 'Northwestern',
 'Notre Dame',
 'Oregon',
 'Oregon State',
 'Pittsburgh',
 'Rice',
 'South Alabama',
 'Southern California',
 'Southern Methodist',
 'Stanford',
 'Syracuse',
 'Tennessee',
 'Texas A&M',
 'Texas Christian',
 'Texas-El Paso',
 'Tulane',
 'Tulsa',
 'UCLA',
 'Vanderbilt',
 'Wake Forest',
 'Wisconsin']
```

[ ]:
```
%%markdown
### 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.

For the graduation rate coefficient, the p-value is less than the standard
 ↪threshold of 0.05. This means that the coefficient is statistically
 ↪significant. The coefficient value is 3.792e+04 (37920), which means that
 ↪for every 1% increase in graduation rate, the predicted salary increases by
 ↪~$37,920. It is the single largest contributor to the predicted salary
 ↪(TotalPay).
```

1.1.8 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.

For the graduation rate coefficient, the p-value is less than the standard threshold of 0.05. This means that the coefficient is statistically significant. The coefficient value is 3.792e+04 (37920), which means that for every 1% increase in graduation rate, the predicted salary increases by ~$37,920. It is the single largest contributor to the predicted salary (TotalPay).

[ ]:
```
model.summary()
```

[ ]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:               TotalPay   R-squared:                       0.837
Model:                            OLS   Adj. R-squared:                  0.831
Method:                 Least Squares   F-statistic:                     141.9
Date:                Sat, 01 Apr 2023   Prob (F-statistic):           1.43e-32
Time:                        13:00:49   Log-Likelihood:                -1306.0
No. Observations:                  87   AIC:                             2620.
Df Residuals:                      83   BIC:                             2630.
Df Model:                           3
Covariance Type:            nonrobust
================================================================================
===========
                              coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
-----------
const                   -1.852e+06   1.11e+06     -1.666      0.100   -4.06e+06
3.59e+05
Buyout                      0.1422      0.016      8.890      0.000       0.110
0.174
Graduation Rate (%)      3.792e+04   1.67e+04      2.265      0.026    4618.651
7.12e+04
Donations (in millions)  2.715e+04   1.06e+04      2.555      0.012    6019.114
4.83e+04
==============================================================================
Omnibus:                       24.047   Durbin-Watson:                   1.845
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               38.355
Skew:                           1.144   Prob(JB):                     4.69e-09
Kurtosis:                       5.313   Cond. No.                     1.49e+08
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.49e+08. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[ ]:
```
%%markdown
### 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 (1.43e-32), it can be determiend that␣
↪the model can be interpreted.

It's then advisable to move onto the R-squared value.  It's seen that the value␣
↪for this is 0.837, which tells the reader that ~83.7% of the change in our Y␣
↪variable (TotalPay) is explained by the change in our independent (X)␣
↪variables -- which is 'Buyout', 'Graduation Rate (%)', and 'Donations (in␣
↪millions)'.

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

The p-values for each indepdnent variable look to be under the 0.05 threshold,␣
↪which means that they are statistically significant to the model.
```

1.1.9 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 (1.43e-32), it can be determiend that the model can be interpreted.

It's then advisable to move onto the R-squared value. It's seen that the value for this is 0.837, which tells the reader that ~83.7% of the change in our Y variable (TotalPay) is explained by the change in our independent (X) variables – which is 'Buyout', 'Graduation Rate (%)', and 'Donations (in millions)'.

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

The p-values for each indepdnent variable look to be under the 0.05 threshold, which means that they are statistically significant to the model.

[ ]:
```
model.summary()
```

[ ]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:               TotalPay   R-squared:                       0.837
Model:                            OLS   Adj. R-squared:                  0.831
Method:                 Least Squares   F-statistic:                     141.9
```

```
Date:                  Sat, 01 Apr 2023   Prob (F-statistic):           1.43e-32
Time:                          13:00:50   Log-Likelihood:                -1306.0
No. Observations:                    87   AIC:                             2620.
Df Residuals:                        83   BIC:                             2630.
Df Model:                             3
Covariance Type:              nonrobust
================================================================================
===========
                             coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
-----------
const                  -1.852e+06   1.11e+06     -1.666      0.100   -4.06e+06
3.59e+05
Buyout                     0.1422      0.016      8.890      0.000       0.110
0.174
Graduation Rate (%)     3.792e+04   1.67e+04      2.265      0.026    4618.651
7.12e+04
Donations (in millions)  2.715e+04   1.06e+04      2.555      0.012    6019.114
4.83e+04
==============================================================================
Omnibus:                       24.047   Durbin-Watson:                   1.845
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               38.355
Skew:                           1.144   Prob(JB):                     4.69e-09
Kurtosis:                       5.313   Cond. No.                     1.49e+08
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.49e+08. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[ ]:
```
%%markdown
### QUESTION 6
- What is the single biggest impact on salary size?

As mentioned above, the single biggest impact on salary size is the graduation
 ↪rate.  This is seen in the coefficient for the graduation rate variable,
 ↪which is 3.792e+04 (37920).  This means that for every 1% increase in
 ↪graduation rate, the predicted salary increases by ~$37,920.
```

1.1.10 QUESTION 6

- What is the single biggest impact on salary size?

As mentioned above, the single biggest impact on salary size is the graduation rate. This is seen in the coefficient for the graduation rate variable, which is 3.792e+04 (37920). This means that for every 1% increase in graduation rate, the predicted salary increases by ~$37,920.

[ ]:
```
# Get the coefficients from the model
coefficients = model.params
coefficients
```

[ ]:
```
const                       -1.851800e+06
Buyout                       1.421501e-01
Graduation Rate (%)          3.792381e+04
Donations (in millions)      2.715142e+04
dtype: float64
```