km5188_ds4e_hw3

November 11, 2022

1 DS4E: Homework 3

[1]:
```
pip install statsmodels
```

```
Collecting statsmodels
  Using cached
statsmodels-0.13.5-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(9.9 MB)
Requirement already satisfied: scipy>=1.3 in /opt/conda/lib/python3.10/site-
packages (from statsmodels) (1.9.0)
Requirement already satisfied: numpy>=1.17 in /opt/conda/lib/python3.10/site-
packages (from statsmodels) (1.23.2)
Collecting pandas>=0.25
  Using cached
pandas-1.5.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.1
MB)
Requirement already satisfied: packaging>=21.3 in
/opt/conda/lib/python3.10/site-packages (from statsmodels) (21.3)
Collecting patsy>=0.5.2
  Using cached patsy-0.5.3-py2.py3-none-any.whl (233 kB)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in
/opt/conda/lib/python3.10/site-packages (from packaging>=21.3->statsmodels)
(3.0.9)
Requirement already satisfied: python-dateutil>=2.8.1 in
/opt/conda/lib/python3.10/site-packages (from pandas>=0.25->statsmodels) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-
packages (from pandas>=0.25->statsmodels) (2022.2.1)
Requirement already satisfied: six in /opt/conda/lib/python3.10/site-packages
(from patsy>=0.5.2->statsmodels) (1.16.0)
Installing collected packages: patsy, pandas, statsmodels
Successfully installed pandas-1.5.1 patsy-0.5.3 statsmodels-0.13.5
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
pip install pandas
```

```
Requirement already satisfied: pandas in /opt/conda/lib/python3.10/site-packages
(1.5.1)
Requirement already satisfied: numpy>=1.21.0 in /opt/conda/lib/python3.10/site-
```

```
packages (from pandas) (1.23.2)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-
packages (from pandas) (2022.2.1)
Requirement already satisfied: python-dateutil>=2.8.1 in
/opt/conda/lib/python3.10/site-packages (from pandas) (2.8.2)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-
packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

[3]:
```
# import libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import statsmodels.formula.api as smf
```

1.1 Question 1

1(a)

support for canceling student debt

1(b)

people's support of the plan

1(c)

The researcher conceptualizes support by studying the thoughts of passers-by in Washington Square Park about whether they support the abolition of student debt.

1(d)

He asks respondents in Washington Square Park on Saturday afternoon to rate their support for the plan on a scale from 1 to 5, 1 means "strongly oppose the plan" and 5 means "strongly support the plan."

1(e)

Such a classification of people's opinions will make it easier for the researcher to record data, store it and manipulate it in the future. In addition, it will also make answers comparable to each other.

1(f)

Since almost every person has a different scale of plan support, measuring support in only 5 discrete categories may not be enough. In addition, one person's subjective perception of "strongly support the plan" may differ from others' subjective perception of the same scale.

1(g)

The researcher misrecording the data.

1(h)

Response bias is when people consciously or unconsciously lie in their answers. There may be various reasons for this, such as social desirability, poor survey design, etc. For example, in this

case, people may be inclined to overestimate their support in order to appear nice and caring about students.

1(i)

Firstly, the student conducted the study only in Washington Square Park. Since there are many NYU buildings located there, there is a high probability that a student will meet a student or a person associated with NYU. If a person is from NYU, then it is quite possible that he will be more supportive of the plan, since it is also likely that he values higher education highly.

Secondly, the researcher asks only those people who voluntarily stop to talk to them. Thus, they create a sample of a voluntary response. In this case, only people with a strong opinion will want to stop to talk to a stranger. In this way, the researcher can get many answers that either "strongly oppose the plan" or "strongly support the plan".

1(j)

Validity Error. When we think we are measuring one thing, but actually we are measuring something else.

1.2 Question 2

2(a)

[4]:
```
import pandas as pd
df = pd.read_csv("forbes_athletes.csv")
display = pd.read_csv("forbes_athletes.csv", nrows = 15)
display
```

[4]:

	Name	Nationality	Current Rank	Sport	Year	earnings ($ million)
0	Mike Tyson	USA	1	boxing	1990	28.6
1	Buster Douglas	USA	2	boxing	1990	26.0
2	Sugar Ray Leonard	USA	3	boxing	1990	13.0
3	Ayrton Senna	Brazil	4	auto racing	1990	
4	Alain Prost	France	5	auto racing	1990	
5	Jack Nicklaus	USA	6	golf	1990	
6	Greg Norman	Australia	7	golf	1990	
7	Michael Jordan	USA	8	basketball	1990	
8	Arnold Palmer	USA	8	golf	1990	
9	Evander Holyfield	USA	8	boxing	1990	
10	Evander Holyfield	USA	1	boxing	1991	
11	Mike Tyson	USA	2	boxing	1991	
12	Michael Jordan	USA	3	basketball	1991	
13	George Foreman	USA	4	boxing	1991	
14	Ayrton Senna	Brazil	5	auto racing	1991	

```
3                    10.0
4                     9.0
5                     8.6
6                     8.5
7                     8.1
8                     8.1
9                     8.1
10                   60.5
11                   31.5
12                   16.0
13                   14.5
14                   13.0
```

2(b)

Earnings

2(c)

[5]:
```
df = df.rename(columns={'Name':'name',
                        'Nationality': 'nationality',
                        'Current Rank':'current_rank',
                        'Sport':'sport',
                        'Year':'year',
                        'earnings ($ million)':'earnings'})
df
```

[5]:
```
                  name nationality  current_rank              sport  year  \
0          Mike Tyson         USA             1             boxing  1990
1      Buster Douglas         USA             2             boxing  1990
2   Sugar Ray Leonard         USA             3             boxing  1990
3        Ayrton Senna      Brazil             4        auto racing  1990
4         Alain Prost      France             5        auto racing  1990
..                 …         …            …                 …    …
296     Stephen Curry         USA             6         Basketball  2020
297      Kevin Durant         USA             7         Basketball  2020
298       Tiger Woods         USA             8               Golf  2020
299      Kirk Cousins         USA             9  American Football  2020
300      Carson Wentz         USA            10  American Football  2020

     earnings
0        28.6
1        26.0
2        13.0
3        10.0
4         9.0
..        …
296      74.4
```

```
297      63.9
298      62.3
299      60.5
300      59.1

[301 rows x 6 columns]
```

2(d)

[6]:
```
df['sport'] = df['sport'].replace(['NFL'],'American Football')
df
```

[6]:
```
                  name nationality  current_rank              sport  year  \
0           Mike Tyson         USA             1             boxing  1990
1       Buster Douglas         USA             2             boxing  1990
2    Sugar Ray Leonard         USA             3             boxing  1990
3         Ayrton Senna      Brazil             4        auto racing  1990
4          Alain Prost      France             5        auto racing  1990
..                  …         …            …                  …    …
296      Stephen Curry         USA             6         Basketball  2020
297       Kevin Durant         USA             7         Basketball  2020
298        Tiger Woods         USA             8               Golf  2020
299       Kirk Cousins         USA             9  American Football  2020
300       Carson Wentz         USA            10  American Football  2020

     earnings
0        28.6
1        26.0
2        13.0
3        10.0
4         9.0
..        …
296      74.4
297      63.9
298      62.3
299      60.5
300      59.1

[301 rows x 6 columns]
```

2(e)

[7]:
```
df['year'].value_counts()
```

[7]:
```
2002    11
1990    10
2007    10
2019    10
```

```
2018    10
2017    10
2016    10
2015    10
2014    10
2013    10
2012    10
2011    10
2010    10
2009    10
2008    10
2006    10
1991    10
2005    10
2004    10
2003    10
2000    10
1999    10
1998    10
1997    10
1996    10
1995    10
1994    10
1993    10
1992    10
2020    10
Name: year, dtype: int64
```

For some reason, all years have 10 observations, except for 2002, which has 11 observations.

2(f)

[8]:
```
top = df.sort_values(by = ['earnings'], ascending = False)
top[['name','year','earnings']].head()
```

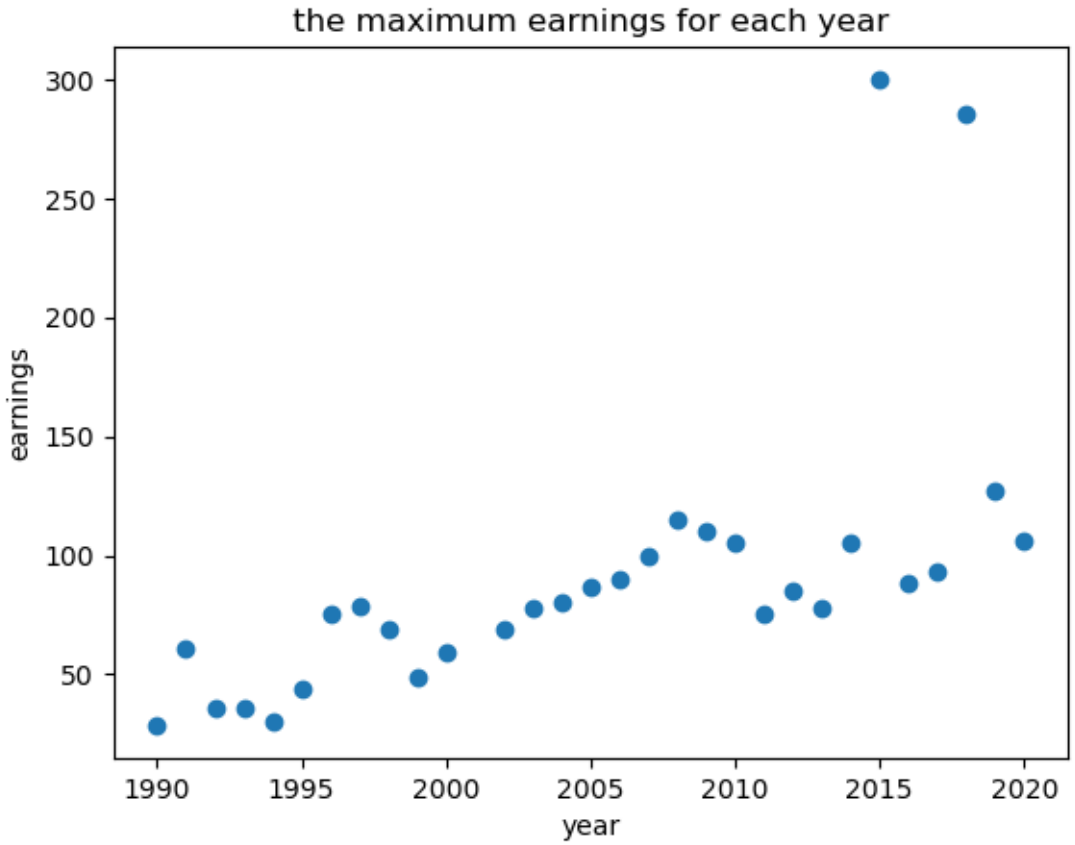
[8]:

	name	year	earnings
241	Floyd Mayweather	2015	300.0
271	Floyd Mayweather	2018	285.0
242	Manny Pacquiao	2015	160.0
281	Lionel Messi	2019	127.0
171	Tiger Woods	2008	115.0

2(g)

[9]:
```
# find max earnings per year
# m=df.groupby('year')['earnings'].transform('max') == df['value']
df_max = df.sort_values('earnings',
```

```
                        ascending = False).drop_duplicates(['year']).
↪sort_values(by =
                                                                    ␣
↪ ['year'])
plt.scatter(df_max['year'], df_max['earnings'])
plt.xlabel('year')
plt.ylabel('earnings')
plt.title('the maximum earnings for each year')
frame = plt.text(0,0,"")
```



This graph shows that the maximum earnings of prominent athletes have been increasing over the past 30 years.

2(h)

[10]:
```
df.groupby('nationality')['earnings'].sum().sort_values(ascending= False)
```

[10]:
```
nationality
USA                 8786.3
Portugal             787.1
```

```
Switzerland          781.1
Argentina            715.5
Germany              639.0
UK                   443.2
Brazil               422.0
Philippines          242.0
Finland              129.0
Italy                128.0
Canada                99.1
Ireland               99.0
Mexico                94.0
Filipino              62.0
Serbia                55.8
Northern Ireland      50.0
Spain                 44.5
France                36.0
Dominican             35.0
Russia                29.8
Austria               13.5
Australia              8.5
Name: earnings, dtype: float64
```

1.3 Question 3

3(a)

[11]:
```
df2 = pd.read_csv("chicago_salary_sample.csv")
mean2 = df2["annual_salary"].mean()
mean2
```

[11]:
```
99217.66344
```

3(b)

[12]:
```
df3 = pd.read_csv("chicago_salary_full.csv")
my_pop_mean = df3["annual_salary"].mean()
my_pop_mean
```

[12]:
```
98915.8253718593
```

3(c)

[13]:
```
df2
```

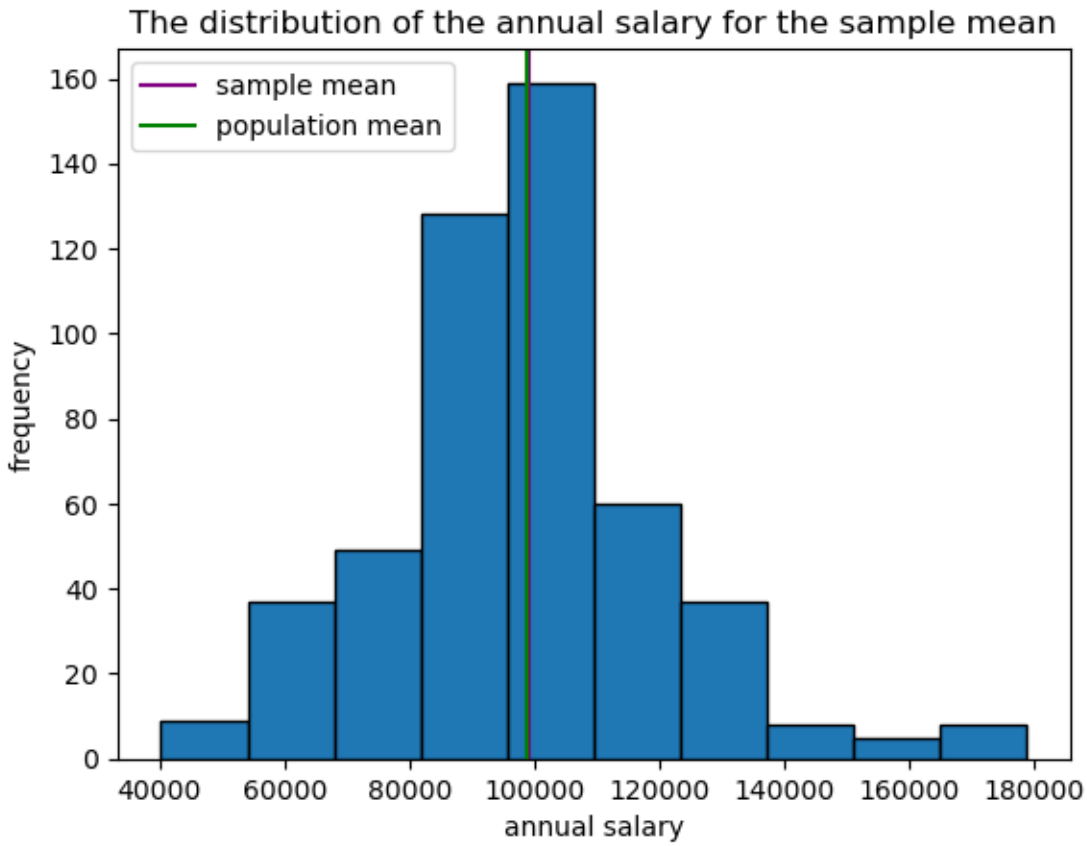
[13]:

	job_titles	department	annual_salary
0	ASST CORPORATION COUNSEL II	LAW	70104.0
1	POLICE OFFICER	POLICE	95586.0
2	LIEUTENANT-EMT	FIRE	135144.0

```
3            FIRE COMMUNICATIONS OPERATOR I              OEMC        72108.0
4                         AREA COORD - CAPS            POLICE       114012.0
..                                      …                 …            …
495  POLICE OFFICER (ASSIGNED AS DETECTIVE)            POLICE       110796.0
496                          FIREFIGHTER-EMT              FIRE        94152.0
497                           LIEUTENANT-EMT              FIRE       126840.0
498        BUSINESS COMPLIANCE INVESTIGATOR  BUSINESS AFFAIRS        70152.0
499                          POLICE OFFICER            POLICE        82458.0

[500 rows x 3 columns]
```

[14]:
```
plt.hist(df2["annual_salary"], ec = 'black')
plt.axvline(x=99217.66344, color ='purple',
            label = 'sample mean')
plt.axvline(x=98915.8253718593, color ='green',
            label = 'population mean')
plt.legend(loc = 'upper left')
plt.xlabel('annual salary')
plt.ylabel('frequency')
plt.title("The distribution of the annual salary for the sample mean")
plt.show()
```



3(d)

[15]:
```
sampled = df2["annual_salary"].sample(frac = 1, replace = True)
mean_sampled = round(sampled.mean(),2)

# print("sampled salary", sampled)
print("mean sampled:",mean_sampled)
```

```
mean sampled: 98534.19
```

3(e)

[16]:
```
means = ([])
for i in np.arange(1000):
    sampled = df2["annual_salary"].sample(frac = 1, replace = True)
    means = np.append(means, round(sampled.mean(), 2))

sample_mean = round(means.mean(),2)
left = np.percentile(means, 2.5)
right = np.percentile(means, 97.5)

print("sample mean:", sample_mean)
print("Does the 95% confidence interval around the sample mean contain the true␣
 ↪mean?:",
      left < my_pop_mean < right)
```

```
sample mean: 99197.78
Does the 95% confidence interval around the sample mean contain the true mean?:
True
```

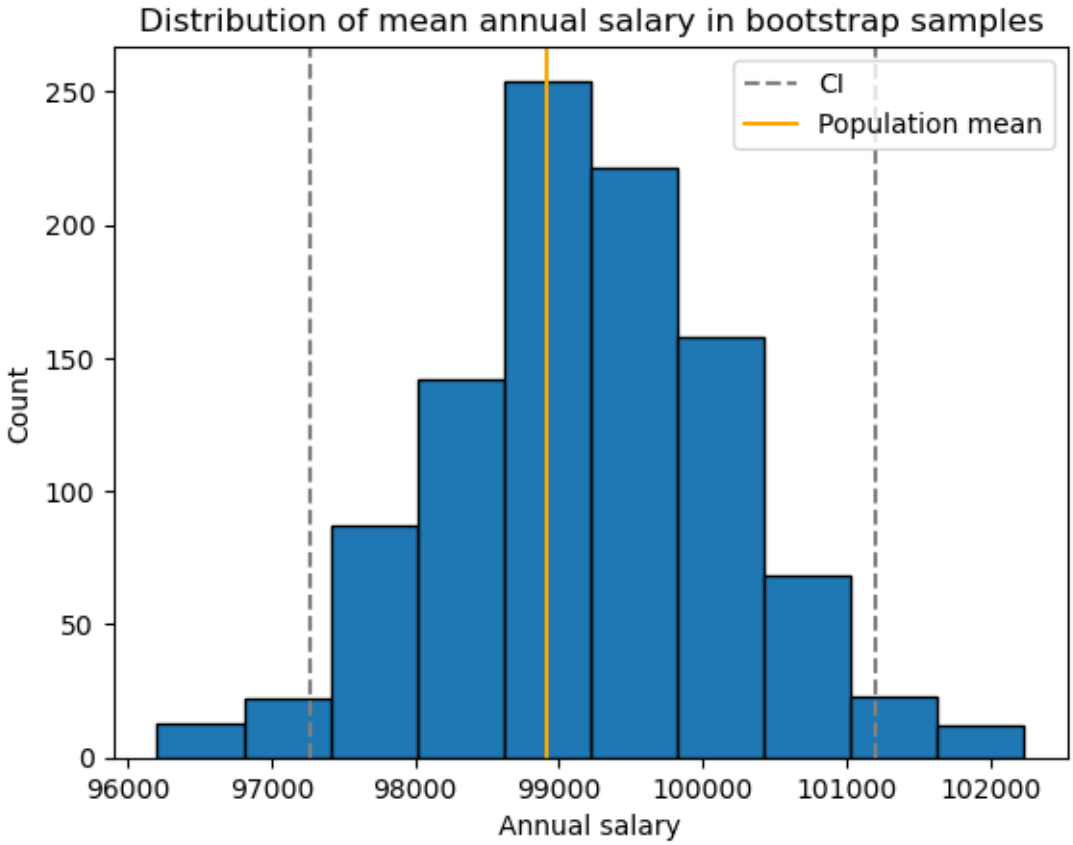
3(f)

[17]:
```
## you can use this function to help with question 3f
def plot_conf(means, population_mean = my_pop_mean, conf=95):
  lower_pct = (100-conf)/2
  upper_pct = 100-((100-conf)/2)
  lower = np.percentile(means, lower_pct)
  upper = np.percentile(means, upper_pct)

  plt.hist(means, ec="black", bins=10)
  plt.axvline(lower, color='grey', linestyle='--', label='CI')
  plt.axvline(upper, color='grey', linestyle='--')
  plt.axvline(population_mean, color='orange', label='Population mean')
  plt.title("Distribution of mean annual salary in bootstrap samples")
  plt.xlabel("Annual salary")
  plt.ylabel("Count")
  plt.legend()
```

```
  plt.show()

  print("Interval:", lower, ",", upper)
plot_conf(means, my_pop_mean)
```




```
Interval: 97273.41475 , 101202.15299999999
```

1.4 Question 4

4(a)

[18]:
```
import pandas as pd
```

[19]:
```
df = pd.read_csv("chicago_salary_full.csv")
df4 = df[df['department'].isin(['POLICE', 'FIRE'])]
df4.head()
```

[19]:

	job_titles	department	annual_salary
0	SERGEANT	POLICE	122568.0
1	POLICE OFFICER (ASSIGNED AS DETECTIVE)	POLICE	110796.0

```
3                          POLICE OFFICER     POLICE        86730.0
4                       FIRE ENGINEER-EMT       FIRE       118830.0
5                          POLICE OFFICER     POLICE       109236.0
```

4(b)

[20]:
```
police = df4[df4['department']=='POLICE']
fire = df4[df4['department']=='FIRE']
print("police annual salary:", police['annual_salary'].mean())
print("fire annual salary:", fire['annual_salary'].mean())
```

```
police annual salary: 101170.56398454837
fire annual salary: 106580.96719082378
```

4(c)

[21]:
```
import statsmodels.formula.api as smf
```

[22]:
```
df_r = df4.replace('POLICE',0)
df_r = df_r.replace('FIRE',1)
df_r
```

[22]:
```
                                  job_titles  department  annual_salary
0                                   SERGEANT           0       122568.0
1        POLICE OFFICER (ASSIGNED AS DETECTIVE)           0       110796.0
3                              POLICE OFFICER           0        86730.0
4                           FIRE ENGINEER-EMT           1       118830.0
5                              POLICE OFFICER           0       109236.0
…                                         …          …             …
23874                          POLICE OFFICER           0        95586.0
23875                          POLICE OFFICER           0        90990.0
23876                          POLICE OFFICER           0        95586.0
23877                          POLICE OFFICER           0       102372.0
23878                          POLICE OFFICER           0       109236.0

[16962 rows x 3 columns]
```

[23]:
```
# X = police["annual_salary"]
# Y = fire["annual_salary"]
model = smf.ols(formula = 'annual_salary ~ department', data = df_r).fit()
model.summary()
```

[23]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:          annual_salary   R-squared:                       0.014
Model:                            OLS   Adj. R-squared:                  0.014
```

```
Method:                 Least Squares   F-statistic:                     248.6
Date:                Fri, 11 Nov 2022   Prob (F-statistic):           1.29e-55
Time:                        19:48:01   Log-Likelihood:            -1.9215e+05
No. Observations:               16962   AIC:                         3.843e+05
Df Residuals:                   16960   BIC:                         3.843e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1.012e+05    182.439    554.546      0.000    1.01e+05    1.02e+05
department  5410.4032    343.132     15.768      0.000    4737.829    6082.977
==============================================================================
Omnibus:                     1268.984   Durbin-Watson:                   1.921
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             4084.504
Skew:                           0.366   Prob(JB):                         0.00
Kurtosis:                       5.290   Cond. No.                         2.44
==============================================================================

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

The coefficient for the department is 5410.4032, which is high when compared with average annual salaries: annual salary of the police: 101170.56398454837; annual salary of fire department: 106580.96719082378. This is because we only have two values for the independent variable: police and fire departments. The R-squared is 0.014, which means that the correlation between the annual salary and the department is very small or nonexistent.

[ ]: