solution-test-excercise-2

August 13, 2023

[1]:
```
import pandas as pd
import numpy as np
import math
import matplotlib.pyplot as plt
import statsmodels.api as sm
from scipy import stats
```

[2]:
```
data = pd.read_excel('DataSet.xls')
data.head()
```

[2]:

	Observation	FGPA	SATM	SATV	FEM
0	1	2.518	4.0	4.0	1
1	2	2.326	4.9	3.1	0
2	3	3.003	4.4	4.0	1
3	4	2.111	4.9	3.9	0
4	5	2.145	4.3	4.7	0

1 Multiple Regression

1.1 Question a

[3]:
```
y = data['FGPA']
X = data['SATV']
X = sm.add_constant(X)
model = sm.OLS(y, X)
results = model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   FGPA   R-squared:                       0.008
Model:                            OLS   Adj. R-squared:                  0.007
Method:                 Least Squares   F-statistic:                     5.201
Date:                Sun, 13 Aug 2023   Prob (F-statistic):             0.0229
Time:                        13:48:22   Log-Likelihood:                -388.44
No. Observations:                 609   AIC:                             780.9
Df Residuals:                     607   BIC:                             789.7
```

```
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.4417      0.155     15.747      0.000       2.137       2.746
SATV           0.0631      0.028      2.280      0.023       0.009       0.117
==============================================================================
Omnibus:                       11.335   Durbin-Watson:                   1.949
Prob(Omnibus):                  0.003   Jarque-Bera (JB):                7.694
Skew:                           0.138   Prob(JB):                       0.0213
Kurtosis:                       2.524   Cond. No.                         48.2
==============================================================================

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

[4]:
```
pvalue = results.pvalues[1]
```

[5]:
```
# From model
se = results.bse[1]
m = results.params[1]
confidence_interval = list(results.conf_int(alpha=0.05).iloc[1, :])


# From manual Calculation:

# dist = stats.norm
# alpha = 0.05
# z = dist.ppf(1 - alpha/ 2)

# lower_limit = m - z*se
# upper_limit = m + z*se
# lower_limit, upper_limit
```

1.2 Answer a

[6]:
```
print('Answer of question a:')
print( f'The coefficient of SATV is {round(m, 3)}')
print( f'The standard error is {round(se, 3)}')
print( f'The p-value is {round(pvalue, 3)}')
print( f'Confidence interval for the effect on FGPA of an increase by 1 point␣
 ↪in SATV is {confidence_interval}')
```

```
Answer of question a:
The coefficient of SATV is 0.063
```

```
The standard error is 0.028
The p-value is 0.023
Confidence interval for the effect on FGPA of an increase by 1 point in SATV is
[0.008757813110037933, 0.11741387764565256]
```

1.2.1 Summarize solution 1 into Function

[7]:
```python
def calc_simple_regression(data, tag):
    y = data['FGPA']
    if tag != 'Const':
        X = data[tag]
        X = sm.add_constant(X)
        col_i = 1
    else:
        X = np.ones(len(y))
        col_i = 0
    model = sm.OLS(y, X)
    results = model.fit()

    pvalue = results.pvalues[col_i]

    # dist = stats.norm
    # alpha = 0.05
    # z = dist.ppf(1 - alpha/ 2)
    # se = results.bse[col_i]
    # m = results.params[col_i]
    # lower_limit = m - z*se
    # upper_limit = m + z*se

    confidence_interval = list(results.conf_int(alpha=0.05).iloc[col_i, :])


    print(f'Answer of question a for {tag}:')
    print( f'The coefficient of SATV is {round(m, 3)}')
    print( f'The standard error is {round(se, 3)}')
    print( f'The p-value is {round(pvalue, 3)}')
    print( f'Confidence interval for the effect on FGPA of an increase by 1␣
 ↪point in SATV is [{lower_limit}, {upper_limit}]')
```

1.3 Question b

[8]:
```python
X = data[['SATM', 'SATV', 'FEM']]
X = sm.add_constant(X)
y = data['FGPA']
ml_results = sm.OLS(y, X).fit()
```

[9]:
```python
ml_results.summary()
```

[9]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                   FGPA   R-squared:                       0.083
Model:                            OLS   Adj. R-squared:                  0.078
Method:                 Least Squares   F-statistic:                     18.24
Date:                Sun, 13 Aug 2023   Prob (F-statistic):           2.41e-11
Time:                        13:48:22   Log-Likelihood:                -364.67
No. Observations:                 609   AIC:                             737.3
Df Residuals:                     605   BIC:                             755.0
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.5570      0.216      7.205      0.000       1.133       1.981
SATM           0.1727      0.032      5.410      0.000       0.110       0.235
SATV           0.0142      0.028      0.507      0.612      -0.041       0.069
FEM            0.2003      0.037      5.358      0.000       0.127       0.274
==============================================================================
Omnibus:                        7.757   Durbin-Watson:                   1.912
Prob(Omnibus):                  0.021   Jarque-Bera (JB):                5.727
Skew:                           0.118   Prob(JB):                       0.0571
Kurtosis:                       2.588   Cond. No.                         103.
==============================================================================

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

[10]:
```
col_i = 2
m = ml_results.params[col_i]
pvalue = ml_results.pvalues[col_i]
confidence_interval = list(ml_results.conf_int(alpha=0.05).iloc[col_i, :])
```

1.4 Answer b

[11]:
```
print('Answer of question a:')
print( f'The coefficient of SATV is {round(m, 3)}')
print( f'The standard error is {round(se, 3)}')
print( f'The p-value is {round(pvalue, 3)}')
print( f'Confidence interval for the effect on FGPA of an increase by 1 point␣
 ↪in SATV is {confidence_interval}')
```

```
Answer of question a:
```

```
The coefficient of SATV is 0.014
The standard error is 0.028
The p-value is 0.612
Confidence interval for the effect on FGPA of an increase by 1 point in SATV is
[-0.040683678193648265, 0.06900747127434032]
```

1.5 Question C

[12]:
```
cov_matrix = ml_results.cov_params()
cov_matrix
```

[12]:

	const	SATM	SATV	FEM
const	0.046697	-0.004977	-0.002649	-0.001390
SATM	-0.004977	0.001019	-0.000265	0.000215
SATV	-0.002649	-0.000265	0.000780	-0.000089
FEM	-0.001390	0.000215	-0.000089	0.001397

[13]:
```
GPA = data[['FGPA', 'SATM', 'SATV', 'FEM']]
GPA.corr()
```

[13]:

	FGPA	SATM	SATV	FEM
FGPA	1.000000	0.195040	0.092167	0.176491
SATM	0.195040	1.000000	0.287801	-0.162680
SATV	0.092167	0.287801	1.000000	0.033577
FEM	0.176491	-0.162680	0.033577	1.000000

1.6 Answer C

1. The total effect SATV is signifgicant, as p-value = 0.028; however, the partial effect of SATV is not significant in the multiple regression model.
2. The correlation between SATV and SATM reduces the effectness of SATV. The effect of SATM can be absorted by SATV. We can exclude SATM from the model.
3. FEM and SATV dont have significant correlation, so gender can be kept in the model.

1.7 Question D

[14]:
```
X = data[['SATM', 'FEM']]
X = sm.add_constant(X)
y = data['FGPA']

restricted_results = sm.OLS(y, X).fit()
restricted_results.summary()
```

[14]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
```

```
Dep. Variable:                   FGPA   R-squared:                       0.083
Model:                            OLS   Adj. R-squared:                  0.080
Method:                 Least Squares   F-statistic:                     27.27
Date:                Sun, 13 Aug 2023   Prob (F-statistic):           4.56e-12
Time:                        13:48:22   Log-Likelihood:                -364.80
No. Observations:                 609   AIC:                             735.6
Df Residuals:                     606   BIC:                             748.8
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.6051      0.194      8.272      0.000       1.224       1.986
SATM           0.1776      0.030      5.828      0.000       0.118       0.237
FEM            0.2019      0.037      5.424      0.000       0.129       0.275
==============================================================================
Omnibus:                        8.121   Durbin-Watson:                   1.909
Prob(Omnibus):                  0.017   Jarque-Bera (JB):                5.977
Skew:                           0.124   Prob(JB):                       0.0504
Kurtosis:                       2.582   Cond. No.                         70.0
==============================================================================

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

[15]:
```
R_0 = restricted_results.rsquared
R_1 = ml_results.rsquared
k = 4
degreef_model = len(y) - 4
g = 1
F = (R_1 - R_0)*degreef_model/(1-R_1)/g
tvalues_SATV = ml_results.tvalues[2]
t_square = tvalues_SATV**2
isEqual = round(t_square, 3) == round(F, 3)
```

1.8 Answer D

[16]:
```
print(f'F statistic rounded to 3 decimals is: {round(F,3)}')
print(f't square rounded to 3 decimals is: {round(t_square,3)}')
print('There for F = t_square.')
```

```
F statistic rounded to 3 decimals is: 0.257
t square rounded to 3 decimals is: 0.257
There for F = t_square.
```