L4 Data Analysis

Linear Regression

Least Squares regression (OLS)

https://www.socscistatistics.com/pvalues/tdistribution.aspx
T value to P value conversion

'''Linear Regression Estimation of a linear regression model using the Spector and Mazzeo (1980) data set.

Documentation: data on 32 students TUCE scores 5 columns with rows = students

```
 1) Grade ..... post grade
 2) constant .. term
 3) psi ...... participation in program
 4) tuce ...... tuce (test of understanding of college economics) score
 5) gpa ....... grade point average
```

'''

```
# For this we only need to import statsmodels
import statsmodels.api as sm

def main():
    # We load the spector dataset as a pandas dataframe
    # Of course, you can load your own datasets
    data = sm.datasets.spector.load_pandas()

    # We define y as the endogenous variable, and x as the
    # exogenous variable.
    # Note that if you load your own data, the methods endog
    # and exog will not be available and you will have to
    # explicitly define the endogenous and exogenous variables
    y, x = data.endog, data.exog
    print(x)
    print(y)

    # # We do the regression
    reg = sm.OLS(y, x).fit()

    # # And here we can see the results in a very nice looking table
    # print('SUMMARY -------------------------------------------')
    print((reg.summary()))
    # # We can only take a look at the parameter values though
    # print('PARAMETERS -----------------------------------------')
    print((reg.params))

    # # We can also extract the residuals
    # # print('RESIDUALS ------------------------------------------')
    # # print((reg.resid))

    # # This line is just to prevent the output from vanishing when you
    # # run the program by double-clicking
    # # input('Done - Hit any key to finish.')

if __name__ == '__main__':
    main()
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: Futur
  import pandas.util.testing as tm
     GPA  TUCE  PSI
0   2.66  20.0  0.0
1   2.89  22.0  0.0
2   3.28  24.0  0.0
3   2.92  12.0  0.0
4   4.00  21.0  0.0
5   2.86  17.0  0.0
6   2.76  17.0  0.0
7   2.87  21.0  0.0
8   3.03  25.0  0.0
9   3.92  29.0  0.0
10  2.63  20.0  0.0
11  3.32  23.0  0.0
12  3.57  23.0  0.0
13  3.26  25.0  0.0
14  3.53  26.0  0.0
15  2.74  19.0  0.0
16  2.75  25.0  0.0
17  2.83  19.0  0.0
18  3.12  23.0  1.0
19  3.16  25.0  1.0
20  2.06  22.0  1.0
21  3.62  28.0  1.0
22  2.89  14.0  1.0
23  3.51  26.0  1.0
24  3.54  24.0  1.0
25  2.83  27.0  1.0
26  3.39  17.0  1.0
27  2.67  24.0  1.0
28  3.65  21.0  1.0
29  4.00  23.0  1.0
30  3.10  21.0  1.0
31  2.39  19.0  1.0
0     0.0
1     0.0
2     0.0
3     0.0
4     1.0
5     0.0
6     0.0
```

```
7     0.0
8     0.0
9     1.0
10    0.0
11    0.0
12    0.0
13    1.0
14    0.0
15    0.0
16    0.0
17    0.0
18    0.0
19    1.0
20    0.0
21    1.0
```

For a multiple regression model with intercept, we want to test the following null hypothesis and alternative hypothesis:

H0: β1 = β2 = ... = βp-1 = 0

H1: βj ≠ 0, for at least one value of j

This test is known as the overall F-test for regression.

Example Liner Regression

Given the follow dataset

```
data_str = '''Region Alcohol Tobacco
North 6.47 4.03
Yorkshire 6.13 3.76
Northeast 6.19 3.77
East_Midlands 4.89 3.34
West_Midlands 5.63 3.47
East_Anglia 4.52 2.92
Southeast 5.89 3.20
Southwest 4.79 2.71
Wales 5.27 3.53
Scotland 6.08 4.51
Northern_Ireland 4.02 4.56'''
```

Read the data string as dataframe

```
from io import StringIO
df = pd.read_csv(StringIO(data_str), sep=r'\s+')
print(df.head())
```

Plot the data in a scatter plot

```
df.plot('Tobacco', 'Alcohol', style='o')
plt.ylabel('Alcohol')
plt.title('Sales in Several UK Regions')
plt.show()
```

Ordinary Least Squares (OLS) Linear Regression

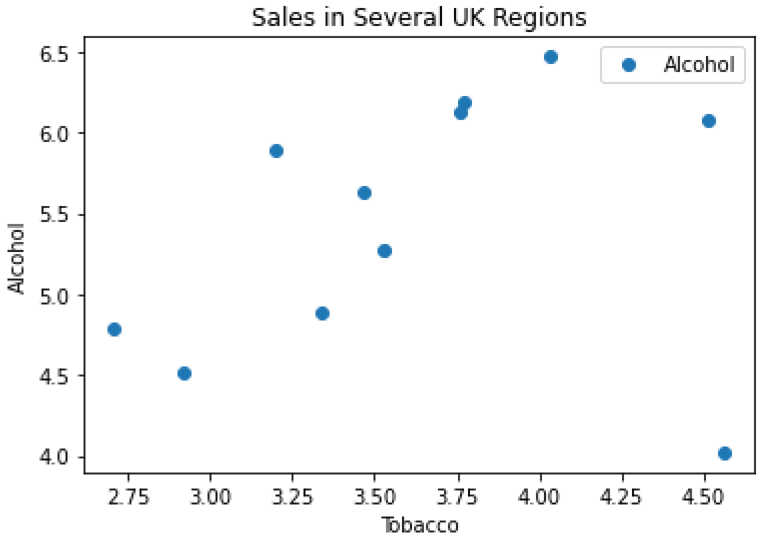
Fit the data using OLS

```
result = sm.OLS( df['Alcohol'],df['Tobacco']).fit()
print(result.summary())
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# import statsmodels.formula.api as sm
from sklearn.linear_model import LinearRegression
from scipy import stats
import statsmodels.api as sm

data_str = '''Region Alcohol Tobacco
North 6.47 4.03
Yorkshire 6.13 3.76
Northeast 6.19 3.77
East_Midlands 4.89 3.34
West_Midlands 5.63 3.47
East_Anglia 4.52 2.92
Southeast 5.89 3.20
Southwest 4.79 2.71
Wales 5.27 3.53
Scotland 6.08 4.51
Northern_Ireland 4.02 4.56'''

#Read the data string as dataframe
#add code
from io import StringIO
df = pd.read_csv(StringIO(data_str), sep=r'\s+')
print(df.head())

# Plot the data using scatter plot
#add code
df.plot('Tobacco', 'Alcohol', style='o')
plt.ylabel('Alcohol')
plt.title('Sales in Several UK Regions')
plt.show()

#Fit the data using OLS
#add code
result = sm.OLS( df['Alcohol'],df['Tobacco']).fit()
```

```
print(result.summary())
```

```
          Region  Alcohol  Tobacco
0          North     6.47     4.03
1      Yorkshire     6.13     3.76
2      Northeast     6.19     3.77
3  East_Midlands     4.89     3.34
4  West_Midlands     5.63     3.47
```



```
                                 OLS Regression Results
=============================================================================
Dep. Variable:                Alcohol   R-squared (uncentered):
Model:                            OLS   Adj. R-squared (uncentered):
Method:                 Least Squares   F-statistic:
Date:                Wed, 11 May 2022   Prob (F-statistic):
Time:                        13:32:00   Log-Likelihood:
No. Observations:                  11   AIC:
Df Residuals:                      10   BIC:
Df Model:                           1
Covariance Type:            nonrobust
=============================================================================
                coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------
Tobacco       1.4761      0.086     17.113      0.000       1.284       1.668
=============================================================================
Omnibus:                       17.342   Durbin-Watson:                  0.673
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              10.940
Skew:                          -1.917   Prob(JB):                     0.00421
Kurtosis:                       6.028   Cond. No.                        1.00
=============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correct
/usr/local/lib/python3.7/dist-packages/scipy/stats/stats.py:1535: UserWarning:
  "anyway, n=%i" % int(n))
```

https://www.youtube.com/watch?v=U7D1h5bbpcs

● ✕