OEIT6 - Data Analytics

Experiment 4: Linear Regression
Name: Gaurav Panchal

UID: 2019120046

```
#Importing the librariesimport pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
```

```
#Reading the dataset
dataset = pd.read_csv("climate_change.csv")
```

```
dataset.head()
```

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosols
0	1983	5	2.556	345.96	1638.59	303.677	191.324	350.113	1366.1024	0.0863
1	1983	6	2.167	345.52	1633.71	303.746	192.057	351.848	1366.1208	0.0794
2	1983	7	1.741	344.15	1633.22	303.795	192.818	353.725	1366.2850	0.0731
3	1983	8	1.130	342.25	1631.35	303.839	193.602	355.633	1366.4202	0.0673
4	1983	9	0.428	340.17	1648.40	303.901	194.392	357.465	1366.2335	0.0619

```
dataset.columns
```

```
Index(['Year', 'Month', 'MEI', 'CO2', 'CH4', 'N2O', 'CFC-11', 'CFC-12', 'TSI',
       'Aerosols', 'Temp'],
      dtype='object')
```

```
Q1 = dataset.quantile(0.25)
Q3 = dataset.quantile(0.75)
IQR = Q3 - Q1
#print(IQR)
dataset = dataset[~((dataset < (Q1 - 1.5 * IQR)) |(dataset > (Q3 + 1.5 * IQR))).any(axis=1
```

```
import statsmodels.api as sm
x = dataset[['MEI', 'CO2', 'CH4', 'N2O', 'CFC-11', 'CFC-12', 'TSI','Aerosols']]
y = dataset[['Temp']]
x2 = sm.add_constant(x)
est = sm.OLS(y,x2)
est2 = est.fit()
print(est2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Temp   R-squared:                       0.703
Model:                            OLS   Adj. R-squared:                  0.692
Method:                 Least Squares   F-statistic:                     69.11
Date:                Tue, 05 Apr 2022   Prob (F-statistic):           2.36e-57
Time:                        09:43:49   Log-Likelihood:                 251.36
No. Observations:                 243   AIC:                            -484.7
Df Residuals:                     234   BIC:                            -453.3
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        -60.8378     23.736     -2.563      0.011    -107.600     -14.075
MEI            0.0665      0.007      9.650      0.000       0.053       0.080
CO2            0.0033      0.002      1.389      0.166      -0.001       0.008
CH4           -0.0005      0.001     -0.895      0.372      -0.002       0.001
N2O           -0.0033      0.010     -0.319      0.750      -0.023       0.017
CFC-11        -0.0032      0.002     -1.319      0.188      -0.008       0.002
CFC-12         0.0027      0.001      2.173      0.031       0.000       0.005
TSI            0.0449      0.018      2.532      0.012       0.010       0.080
Aerosols      -8.2339      2.042     -4.032      0.000     -12.257      -4.211
==============================================================================
Omnibus:                        3.269   Durbin-Watson:                   1.015
Prob(Omnibus):                  0.195   Jarque-Bera (JB):                2.996
Skew:                           0.194   Prob(JB):                        0.224
Kurtosis:                       3.381   Cond. No.                     9.94e+06
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
[2] The condition number is large, 9.94e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:117: FutureWarning
  x = pd.concat(x[::order], 1)
```

Spliting the data into a training set, consisting of all the observations up to and including 2006, and a testing set consisting of the remaining years

```
df_train = dataset[dataset.iloc[:,0]<=2006]
df_train.head()
```

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosols
29	1985	10	-0.140	343.08	1681.56	305.395	215.327	390.676	1365.5269	0.0101
30	1985	11	-0.050	344.40	1680.68	305.530	216.282	392.714	1365.6289	0.0097
31	1985	12	-0.293	345.82	1677.99	305.653	217.326	394.539	1365.6794	0.0122
32	1986	1	-0.307	346.54	1675.82	305.775	218.382	396.082	1365.6746	0.0146
33	1986	2	-0.191	347.13	1666.83	305.911	219.379	397.345	1365.5475	0.0158

```
df_test = dataset[dataset.iloc[:,0]>2006]
df_test.head()
```

	Year	Month	MEI	CO2	CH4	N2O	CFC-11	CFC-12	TSI	Aerosol
284	2007	1	0.974	382.93	1799.66	320.561	248.372	539.206	1365.7173	0.005
285	2007	2	0.510	383.81	1803.08	320.571	248.264	538.973	1365.7145	0.005
286	2007	3	0.074	384.56	1803.10	320.548	247.997	538.811	1365.7544	0.004
287	2007	4	-0.049	386.40	1802.11	320.518	247.574	538.586	1365.7228	0.004
288	2007	5	0.183	386.58	1795.65	320.445	247.224	538.130	1365.6932	0.004

linear regression model of traing set

```
#Setting the value for X and Y
x_train = df_train[['MEI', 'CO2', 'CH4', 'N2O', 'CFC-11', 'CFC-12', 'TSI','Aerosols']]
y_train = df_train['Temp']
x2_train = sm.add_constant(x_train)
est_train = sm.OLS(y_train,x2_train)
est2_train = est_train.fit()
print(est2_train.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Temp   R-squared:                       0.722
Model:                            OLS   Adj. R-squared:                  0.711
Method:                 Least Squares   F-statistic:                     68.15
Date:                Tue, 05 Apr 2022   Prob (F-statistic):           3.37e-54
Time:                        09:43:49   Log-Likelihood:                 229.49
No. Observations:                 219   AIC:                            -441.0
Df Residuals:                     210   BIC:                            -410.5
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        -51.0320     24.469     -2.086      0.038     -99.268      -2.796
MEI            0.0622      0.007      8.508      0.000       0.048       0.077
CO2            0.0050      0.002      1.995      0.047    5.82e-05       0.010
CH4           -0.0004      0.001     -0.689      0.491      -0.001       0.001
N2O            0.0018      0.012      0.156      0.876      -0.021       0.025
CFC-11        -0.0011      0.003     -0.406      0.685      -0.007       0.004
CFC-12         0.0014      0.001      0.940      0.348      -0.002       0.004
TSI            0.0360      0.019      1.931      0.055      -0.001       0.073
Aerosols      -8.4359      2.024     -4.167      0.000     -12.427      -4.445
==============================================================================
Omnibus:                        6.330   Durbin-Watson:                   0.994
Prob(Omnibus):                  0.042   Jarque-Bera (JB):                6.027
Skew:                           0.363   Prob(JB):                       0.0491
Kurtosis:                       3.366   Cond. No.                     9.82e+06
==============================================================================

Warnings:
```

```
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
[2] The condition number is large, 9.82e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:117: FutureWarning
  x = pd.concat(x[::order], 1)
```

1.1

The value of R-squared : 0.722

1.2

Which variables are significant in the model?

Ans. : MEI, CO2, TSI ,Aerosols (p-value <0.05)

```
#Setting the value for X and Y
x_test = df_test[['MEI', 'CO2', 'CH4', 'N2O', 'CFC-11', 'CFC-12', 'TSI',
       'Aerosols']]
y_test = df_test['Temp']
```

```
x_train.size
```

```
1752
```

```
y_train.size
```

```
219
```

```
x_test.size
```

```
192
```

```
y_test.size
```

```
24
```

```
from sklearn.linear_model import LinearRegression
```

```
mlr = LinearRegression()
mlr.fit(x_train,y_train)
```

```
LinearRegression()
```

```
print("Intercept: ", mlr.intercept_)
print("Coefficients:")
list(zip(x_train, mlr.coef_))
```

```
Intercept:  -51.031969159858036
Coefficients:
[('MEI', 0.06223569777302381),
 ('CO2', 0.0049606987940408465),
 ('CH4', -0.00038810727802363575),
 ('N2O', 0.0018262419311547184),
 ('CFC-11', -0.0011344993284555694),
 ('CFC-12', 0.0014013277032073551),
 ('TSI', 0.03604734063953119),
 ('Aerosols', -8.435947559286046)]
```

```
from scipy.stats import pearsonr
list1 = df_train['MEI']
list2 = df_train['N2O']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: -0.062
```

```
from scipy.stats import pearsonr
list1 = df_train['CO2']
list2 = df_train['N2O']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: 0.975
```

```
from scipy.stats import pearsonr
list1 = df_train['CH4']
list2 = df_train['N2O']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: 0.890
```

```
from scipy.stats import pearsonr
list1 = df_train['CFC-11']
list2 = df_train['N2O']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: 0.327
```

```
from scipy.stats import pearsonr
list1 = df_train['CFC-12']
list2 = df_train['N2O']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: 0.865
```

```
from scipy.stats import pearsonr
list1 = df_train['TSI']
list2 = df_train['N2O']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: 0.160
```

```
from scipy.stats import pearsonr
list1 = df_train['Aerosols']
list2 = df_train['N2O']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: -0.661
```

2.1

Which of the following is the simplest correct explanation for this contradiction?

I. Climate scientists are wrong that N2O and CFC-11 are greenhouse gases - this regression analysis constitutes part of a disproof.

II. There is not enough data, so the regression coefficients being estimated are not accurate.

III. All of the gas concentration variables reflect human development - N2O and CFC.11 are correlated with other variables in the data set.

Ans:All of the gas concentration variables reflect human development - N2O and CFC.11 are correlated with other variables in the data set.

Correlation

Which of the following independent variables is N2O highly correlated with (absolute correlation greater than 0.7)?

ans. b) CO2 c) CH4 f) CFC.12

[ ] ↳ *14 cells hidden*

Conclusion

Problem 1.1 - Creating Our First Model Enter the model R2 (the "Multiple R-squared" value): 0.722

Problem 1.2 - Creating Our First Model

Which variables are significant in the model? We will consider a variable significant only if the p-value is below 0.05. (Select all that apply.)

a) MEI b) CO2 c) CH4 d) N2O e) CFC.11 f) CFC.12 g) TSI h) Aerosols

Ans.: MEI, CO2, TSI ,Aerosols (p-value <0.05)

Problem 2.1 - Understanding the Model

Which of the following is the simplest correct explanation for this contradiction?

I. Climate scientists are wrong that N2O and CFC-11 are greenhouse gases - this regression analysis constitutes part of a disproof.

II. There is not enough data, so the regression coefficients being estimated are not accurate.

III. All of the gas concentration variables reflect human development - N2O and CFC.11 are correlated with other variables in the data set.

Ans. : All of the gas concentration variables reflect human development - N2O and CFC.11 are correlated with other variables in the data set.

Compute the correlations between all the variables in the training set. Which of the following independent variables is N2O highly correlated with (absolute correlation greater than 0.7)? Select all that apply.

a) MEI b) CO2 c) CH4 d) N2O e) CFC.11 f) CFC.12 g) TSI h) Aerossol

Ans: b) CO2 c) CH4 f) CFC.12

Which of the following independent variables is CFC.11 highly correlated with? Select all that apply.

a)MEI b) CO2 c) CH4 d) N2O e) CFC.12 f) TSI g) Aerosols

ans.: CFC.12

Problem 3 - Simplifying the Model

Given that the correlations are so high, let us focus on the N2O variable and build a model with only MEI, TSI, Aerosols and N2O as independent variables. Remember to use the training set to build the model.

Enter the coefficient of N2O in this reduced model: 0.0217

(How does this compare to the coefficient in the previous model with all of the variables?)

Enter the model R2: 0.706

Inference:

Often when we get a dataset, we might find a plethora of features in the dataset. All of the features we find in the dataset might not be useful in building a machine learning model to make the necessary prediction. Using some of the features might even make the predictions worse. So, feature selection plays a huge role in building a machine learning model.

Features with high correlation are more linearly dependent and hence have almost the same effect on the dependent variable. So, when two features have high correlation, we can drop one of the two features.