MultipleLinearRegression

Program to implement multiple linear regression technique using any standard dataset available in the public domain and evaluate its performance.

The description for all the columns containing data for air pollutants, temperature, relative humidity and absolute humidity is provided below.

Columns	Description
PT08.S1(CO)	PT08.S1 (tin oxide) hourly averaged sensor response (nominally CO targeted)
C6H6(GT)	True hourly averaged Benzene concentration in $\frac{\mu g}{m^3}$
PT08.S2(NMHC)	PT08.S2 (titania) hourly averaged sensor response (nominally NMHC targeted)
PT08.S3(NOx)	PT08.S3 (tungsten oxide) hourly averaged sensor response (nominally NO_x targeted)
PT08.S4(NO2)	PT08.S4 (tungsten oxide) hourly averaged sensor response (nominally NO_2 targeted)
PT08.S5(O3)	PT08.S5 (indium oxide) hourly averaged sensor response (nominally O_3 targeted)
T	Temperature in Â°C
RH	Relative Humidity (%)
AH	AH Absolute Humidity

Multiple Linear Regression Model Using `sklearn` Module

```
1 #Load Dataset & display 1st 5 rows. Github link is as follows:
2 # https://raw.githubusercontent.com/jiss-sngce/air/main/airquality.csv.csv
3 import pandas as pd
4 ds=pd.read_csv('https://raw.githubusercontent.com/jiss-sngce/air/main/airquality.csv.cs
5 ds.head()
```

	DateTime	PT08.S1(CO)	C6H6(GT)	PT08.S2(NMHC)	PT08.S3(NOx)	PT08.S4(NO2)	PT08.S
0	2004-03-10 18:00:00	1360.0	11.9	1046.0	1056.0	1692.0	
1	2004-03-10 19:00:00	1292.0	9.4	955.0	1174.0	1559.0	
2	2004-03-10	1402.0	9.0	939.0	1140.0	1555.0	

```
1 #Display the columns in dataframe
2 ds.columns
```

```
Index(['DateTime', 'PT08.S1(CO)', 'C6H6(GT)', 'PT08.S2(NMHC)', 'PT08.S3(NOx)',
```

```
       'PT08.S4(NO2)', 'PT08.S5(O3)', 'T', 'RH', 'AH', 'Year', 'Month', 'Day',
       'Day Name'],
      dtype='object')
```

```
# Build a linear regression model using the sklearn module by including all the feature
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
features=list(ds.columns.values[1:-1])
features.remove('RH')
f=ds[features]
t=ds['RH']
# Splitting the DataFrame into the train and test sets.
# Test set will have 33% of the values.
f_train,f_test,t_train,t_test=train_test_split(f,t,test_size=0.33,random_state=42)
t_train_reshaped=t_train.values.reshape(-1,1)
t_test_reshaped=t_test.values.reshape(-1,1)

# Build a linear regression model using the 'sklearn.linear_model' module.
sklearn_lin_reg=LinearRegression()
sklearn_lin_reg.fit(f_train,t_train_reshaped)
# Print the value of the intercept .

print("intercept = ",sklearn_lin_reg.intercept_[0])

# Print the names of the features along with the values of their corresponding coeffici
print("\ncoeff = ",sklearn_lin_reg.coef_)
for item in list(zip(f.columns.values,sklearn_lin_reg.coef_[0])):
  print(item[0],item[1])
```

```
intercept =  -15028.451823247718

coeff =  [[ 1.48327948e-02 -9.03464156e-01 -5.88095941e-03  1.50325488e-03
   2.64965020e-02 -1.06574176e-03 -2.35491907e+00  2.95517421e+01
   7.50515310e+00  1.16786097e+00  3.52321248e-02]]
PT08.S1(CO) 0.014832794792690625
C6H6(GT) -0.9034641560183382
PT08.S2(NMHC) -0.005880959405385411
PT08.S3(NOx) 0.0015032548783276978
PT08.S4(NO2) 0.026496502045666503
PT08.S5(O3) -0.001065741763271788
T -2.354919067592639
AH 29.551742104329783
Year 7.505153097892558
Month 1.1678609682998067
Day 0.03523212478929974
```

```
# Evaluate the linear regression model using the 'r2_score', 'mean_squared_error' & 'me
from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error
import numpy as np
t_train_pred=sklearn_lin_reg.predict(f_train)
t_test_pred=sklearn_lin_reg.predict(f_test)

print("**** TRAIN SET ****")
print("R-squared = ",r2_score(t_train_reshaped,t_train_pred))
print("mean squared error = ",mean_squared_error(t_train_reshaped,t_train_pred))
print("root mean squared error = ",np.sqrt(mean_squared_error(t_train_reshaped,t_train
```

```
print("root mean squared error = ",np.sqrt(mean_squared_error(t_train_reshaped,t_train_
print("mean absolute error = ",mean_absolute_error(t_train_reshaped,t_train_pred))
print("\n**** TEST SET ****")
print("R-squared = ",r2_score(t_test_reshaped,t_test_pred))
print("mean squared error = ",mean_squared_error(t_test_reshaped,t_test_pred))
print("root mean squared error = ",np.sqrt(mean_squared_error(t_test_reshaped,t_test_pr
print("mean absolute error = ",mean_absolute_error(t_test_reshaped,t_test_pred))
```

```
**** TRAIN SET ****
R-squared =  0.8785638240066055
mean squared error =  35.11591834141915
root mean squared error =  5.925868572742662
mean absolute error =  4.571994849644625

**** TEST SET ****
R-squared =  0.8787020691681189
mean squared error =  34.702124455429534
root mean squared error =  5.8908509109830245
mean absolute error =  4.5644604329243466
```