ASSIGNMENT - 4

Finding the highest correlation factors

In [1]:
```
# importing packages pandas, numpy, matplotlib and seaborn
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
# loading the dataset
airQuality = pd.read_excel('AirQualityUCI.xlsx')
```

In [3]:
```
airQuality.head()
```

Out[3]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(
0	2004-03-10	18:00:00	2.6	1360.00	150	11.881723	1045.50	166.0	1056.25	113.0	1692.00	1267
1	2004-03-10	19:00:00	2.0	1292.25	112	9.397165	954.75	103.0	1173.75	92.0	1558.75	972
2	2004-03-10	20:00:00	2.2	1402.00	88	8.997817	939.25	131.0	1140.00	114.0	1554.50	1074
3	2004-03-10	21:00:00	2.2	1375.50	80	9.228796	948.25	172.0	1092.00	122.0	1583.75	1203
4	2004-03-10	22:00:00	1.6	1272.25	51	6.518224	835.50	131.0	1205.00	116.0	1490.00	1110

In [4]:
```
airQuality.tail()
```

Out[4]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S
9352	2005-04-04	10:00:00	3.1	1314.25	-200	13.529605	1101.25	471.7	538.50	189.8	1374.25	1
9353	2005-04-04	11:00:00	2.4	1162.50	-200	11.355157	1027.00	353.3	603.75	179.2	1263.50	1
9354	2005-04-04	12:00:00	2.4	1142.00	-200	12.374538	1062.50	293.0	603.25	174.7	1240.75	1
9355	2005-04-04	13:00:00	2.1	1002.50	-200	9.547187	960.50	234.5	701.50	155.7	1041.00	
9356	2005-04-04	14:00:00	2.2	1070.75	-200	11.932060	1047.25	265.2	654.00	167.7	1128.50	

In [5]:
```
#showing number of rows and columns in the dataset
airQuality.shape
```

Out[5]:
```
(9357, 17)
```

In [6]:
```
# generating descriptive statistics
airQuality.describe()
```

Out[6]:

	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S
count	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.0
mean	-34.207524	1048.869652	-159.090093	1.865576	894.475963	168.604200	794.872333	58.135898	1391.363266	974.9
std	77.657170	329.817015	139.789093	41.380154	342.315902	257.424561	321.977031	126.931428	467.192382	456.9
min	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.0
25%	0.600000	921.000000	-200.000000	4.004958	711.000000	50.000000	637.000000	53.000000	1184.750000	699.7
50%	1.500000	1052.500000	-200.000000	7.886653	894.500000	141.000000	794.250000	96.000000	1445.500000	942.0
75%	2.600000	1221.250000	-200.000000	13.636091	1104.750000	284.200000	960.250000	133.000000	1662.000000	1255.2
max	11.900000	2039.750000	1189.000000	63.741476	2214.000000	1479.000000	2682.750000	339.700000	2775.000000	2522.7

In [7]:
```
# deleting last two columns
airQuality.drop(airQuality.columns[[15, 16]], axis = 1, inplace = True)
```

In [8]:
```
# generating descriptive statistics
airQuality.describe()
```

Out[8]:

	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S
count	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.0
mean	-34.207524	1048.869652	-159.090093	1.865576	894.475963	168.604200	794.872333	58.135898	1391.363266	974.9
std	77.657170	329.817015	139.789093	41.380154	342.315902	257.424561	321.977031	126.931428	467.192382	456.9
min	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.0
25%	0.600000	921.000000	-200.000000	4.004958	711.000000	50.000000	637.000000	53.000000	1184.750000	699.7
50%	1.500000	1052.500000	-200.000000	7.886653	894.500000	141.000000	794.250000	96.000000	1445.500000	942.0
75%	2.600000	1221.250000	-200.000000	13.636091	1104.750000	284.200000	960.250000	133.000000	1662.000000	1255.2
max	11.900000	2039.750000	1189.000000	63.741476	2214.000000	1479.000000	2682.750000	339.700000	2775.000000	2522.7

In [9]:
```
#information about dataset
airQuality.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9357 entries, 0 to 9356
Data columns (total 15 columns):
 #   Column         Non-Null Count  Dtype         
---  ------         --------------  -----         
 0   Date           9357 non-null   datetime64[ns]
 1   Time           9357 non-null   object        
 2   CO(GT)         9357 non-null   float64       
 3   PT08.S1(CO)    9357 non-null   float64       
 4   NMHC(GT)       9357 non-null   int64         
 5   C6H6(GT)       9357 non-null   float64       
 6   PT08.S2(NMHC)  9357 non-null   float64       
 7   NOx(GT)        9357 non-null   float64       
 8   PT08.S3(NOx)   9357 non-null   float64       
 9   NO2(GT)        9357 non-null   float64       
 10  PT08.S4(NO2)   9357 non-null   float64       
 11  PT08.S5(O3)    9357 non-null   float64       
 12  T              9357 non-null   float64       
 13  RH             9357 non-null   float64       
 14  AH             9357 non-null   float64       
dtypes: datetime64[ns](1), float64(12), int64(1), object(1)
memory usage: 1.1+ MB
```

In [10]:
```
# outliers in dataset are considered to be -200
airQuality.isin([-200]).any()
```

Out[10]:
```
Date             False
Time             False
CO(GT)            True
PT08.S1(CO)       True
NMHC(GT)          True
C6H6(GT)          True
PT08.S2(NMHC)     True
NOx(GT)           True
PT08.S3(NOx)      True
NO2(GT)           True
PT08.S4(NO2)      True
PT08.S5(O3)       True
T                 True
RH                True
AH                True
dtype: bool
```

In [11]:
```
# replacing -200 with NaN values
airQuality.replace(to_replace = -200, value =np.nan,inplace=True)
```

In [12]:
```
# checking and counting for missing data points for each column
airQuality.isnull().sum()
```

Out[12]:
```
Date                0
Time                0
CO(GT)           1683
PT08.S1(CO)       366
NMHC(GT)         8443
C6H6(GT)          366
PT08.S2(NMHC)     366
NOx(GT)          1639
PT08.S3(NOx)      366
NO2(GT)          1642
PT08.S4(NO2)      366
PT08.S5(O3)       366
T                 366
RH                366
AH                366
dtype: int64
```

In [13]:
```
# replacing the null values with 0
airQuality.fillna(0,inplace=True)
```

In [14]:
```
# now we can see all columns have zero missing values
airQuality.isnull().sum()
```

Out[14]:
```
Date             0
Time             0
CO(GT)           0
PT08.S1(CO)      0
NMHC(GT)         0
C6H6(GT)         0
PT08.S2(NMHC)    0
NOx(GT)          0
PT08.S3(NOx)     0
NO2(GT)          0
PT08.S4(NO2)     0
PT08.S5(O3)      0
T                0
RH               0
AH               0
dtype: int64
```

In [15]:
```
# generating descriptive statistics
airQuality.describe()
```

Out[15]:

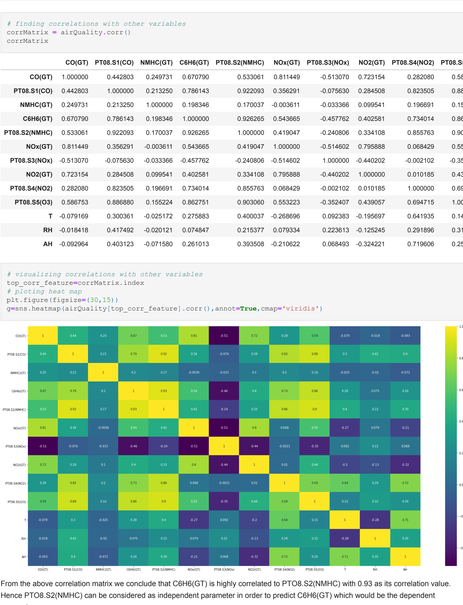
	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S
count	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.0
mean	1.765545	1056.692672	21.373731	9.688596	902.298983	203.636796	802.695353	93.232617	1399.186287	982.7
std	1.554264	301.232260	91.103489	7.559609	318.681183	214.984126	299.341439	61.468588	441.442059	438.0
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	0.600000	921.000000	0.000000	4.004958	711.000000	50.000000	637.000000	53.000000	1184.750000	699.7
50%	1.500000	1052.500000	0.000000	7.886653	894.500000	141.000000	794.250000	96.000000	1445.500000	942.0
75%	2.600000	1221.250000	0.000000	13.636091	1104.750000	284.200000	960.250000	133.000000	1662.000000	1255.2
max	11.900000	2039.750000	1189.000000	63.741476	2214.000000	1479.000000	2682.750000	339.700000	2775.000000	2522.7

In [16]:
```
# finding correlations with other variables
corrMatrix = airQuality.corr()
corrMatrix
```

Out[16]:

	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(
CO(GT)	1.000000	0.442803	0.249731	0.670790	0.533061	0.811449	-0.513070	0.723154	0.282080	0.586
PT08.S1(CO)	0.442803	1.000000	0.213250	0.786143	0.922093	0.356291	-0.075630	0.284508	0.823505	0.886
NMHC(GT)	0.249731	0.213250	1.000000	0.198346	0.170037	-0.003611	-0.033366	0.099541	0.196691	0.155
C6H6(GT)	0.670790	0.786143	0.198346	1.000000	0.926265	0.543665	-0.457762	0.402581	0.734014	0.862
PT08.S2(NMHC)	0.533061	0.922093	0.170037	0.926265	1.000000	0.419047	-0.240806	0.334108	0.855763	0.903
NOx(GT)	0.811449	0.356291	-0.003611	0.543665	0.419047	1.000000	-0.514602	0.795888	0.068429	0.553
PT08.S3(NOx)	-0.513070	-0.075630	-0.033366	-0.457762	-0.240806	-0.514602	1.000000	-0.440202	-0.002102	-0.352
NO2(GT)	0.723154	0.284508	0.099541	0.402581	0.334108	0.795888	-0.440202	1.000000	0.010185	0.439
PT08.S4(NO2)	0.282080	0.823505	0.196691	0.734014	0.855763	0.068429	-0.002102	0.010185	1.000000	0.694
PT08.S5(O3)	0.586753	0.886880	0.155224	0.862751	0.903060	0.553223	-0.352407	0.439057	0.694715	1.000
T	-0.079169	0.300361	-0.025172	0.275883	0.400037	-0.268696	0.092383	-0.195697	0.641935	0.149
RH	-0.018418	0.417492	-0.020121	0.074847	0.215377	0.079334	0.223613	-0.125245	0.291896	0.318
AH	-0.092964	0.403123	-0.071580	0.261013	0.393508	-0.210622	0.068493	-0.324221	0.719606	0.259

In [17]:
```
# visualizing correlations with other variables
top_corr_feature=corrMatrix.index
# plotting heat map
plt.figure(figsize=(30,15))
g=sns.heatmap(airQuality[top_corr_feature].corr(),annot=True,cmap='viridis')
```



From the above correlation matrix we conclude that C6H6(GT) is highly correlated to PT08.S2(NMHC) with 0.93 as its correlation value. Hence PT08.S2(NMHC) can be considered as independent parameter in order to predict C6H6(GT) which would be the dependent parameter

In [18]:
```
X = airQuality[['PT08.S2(NMHC)']] # taking independent parameter as X
y = airQuality['C6H6(GT)'] # taking dependent parameter as y
```