Visualize the Iris and Air Quality Dataset

April 15, 2021

1 1 Iris Dataset

[1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```python
# read data and get the data of four features
fileURL = 'http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.
↪data'
df = pd.read_csv(fileURL, names=['Sepal Length', 'Sepal Width', 'Petal Length',␣
↪'Petal Width', 'Class'], header=None)
```

[3]:
```python
# display the first five rows of data
df.head(5)
```

[3]:

	Sepal Length	Sepal Width	Petal Length	Petal Width	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

1.1 1.1 Summary Statistics

[4]:
```python
# display summary statistics for each feature (min, max, mean,
# standard deviation, count and 25:50:75% percentiles)
df.describe()
```

[4]:

	Sepal Length	Sepal Width	Petal Length	Petal Width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

[5]:
```python
# range
df[df.columns[0:4]].max()-df[df.columns[0:4]].dropna().min()
```

[5]:
```
Sepal Length    3.6
Sepal Width     2.4
Petal Length    5.9
Petal Width     2.4
dtype: float64
```

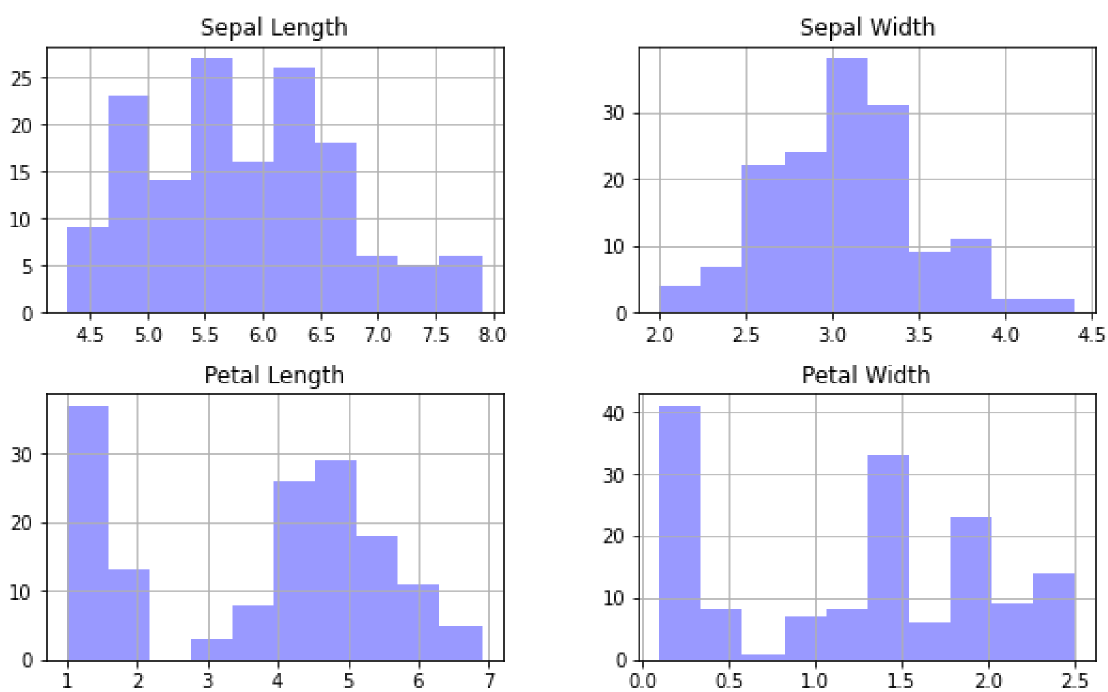
[6]:
```python
# variance
df.var()
```

[6]:
```
Sepal Length    0.685694
Sepal Width     0.188004
Petal Length    3.113179
Petal Width     0.582414
dtype: float64
```

[7]:
```python
## 1.2  Data Visualization
```

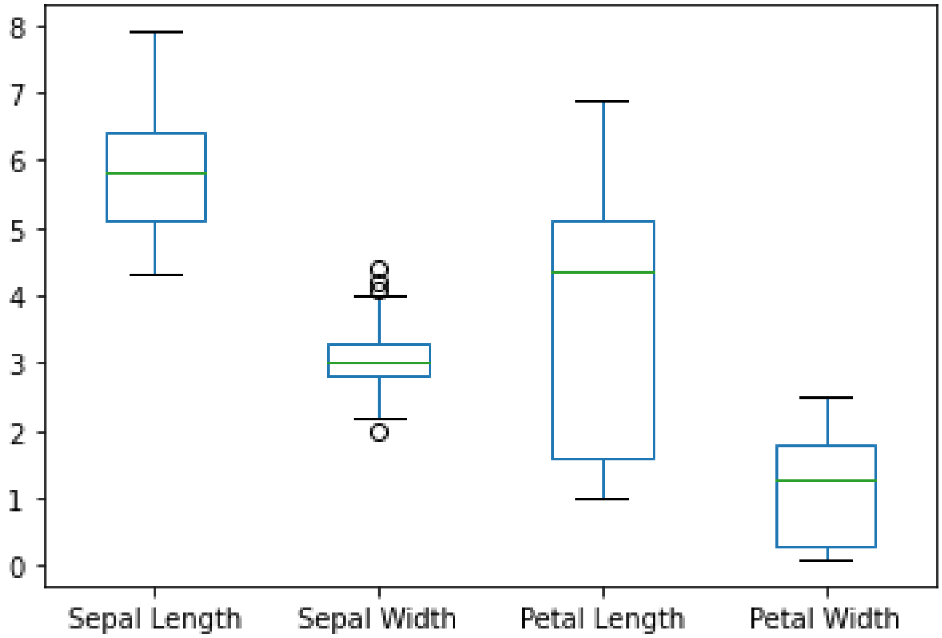
1.1.1 Histograms

[8]:
```python
iris_hist = df.hist(color='b',alpha=0.4,figsize=(10,6))
```



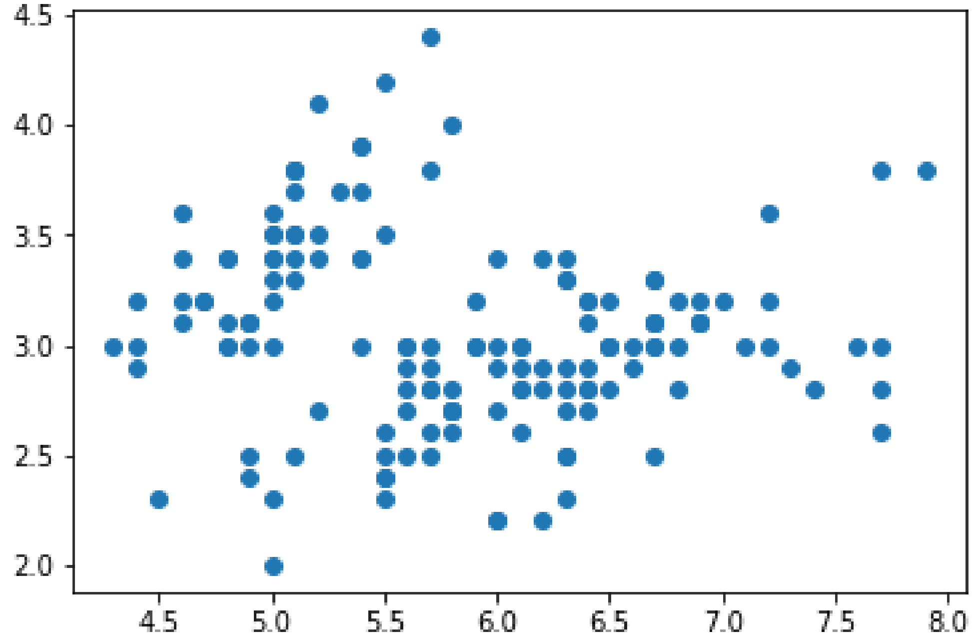
1.1.2 Box Plots

[9]:
```
box = df.boxplot(grid=False, return_type='axes')
```

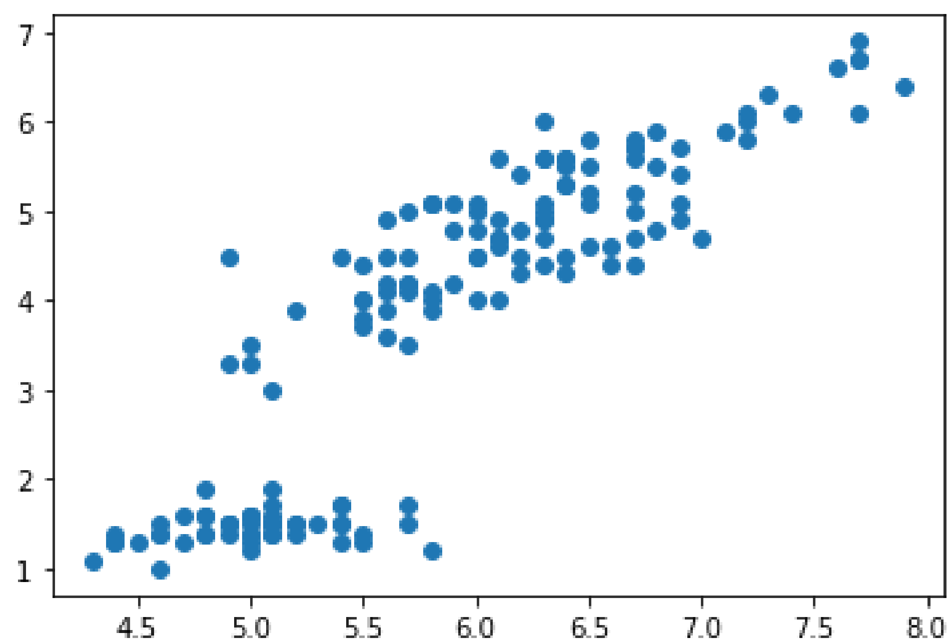


1.1.3 Pairwise Plot (scatter plots)

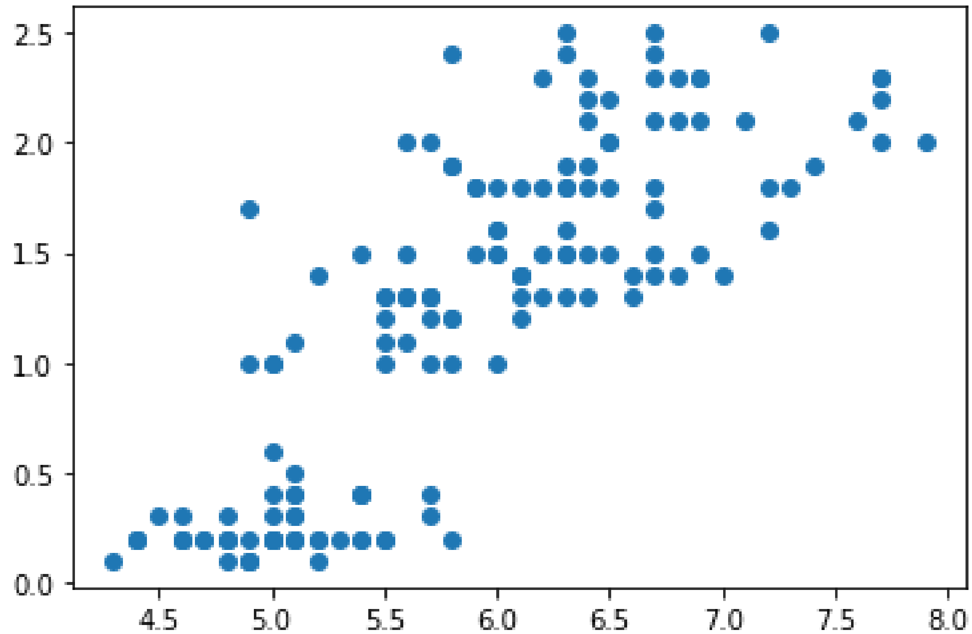
[10]:
```
# 1. scatter plot for sepal length and sepal width
scatter_slen_swid = plt.scatter(df['Sepal Length'], df['Sepal Width'])
```



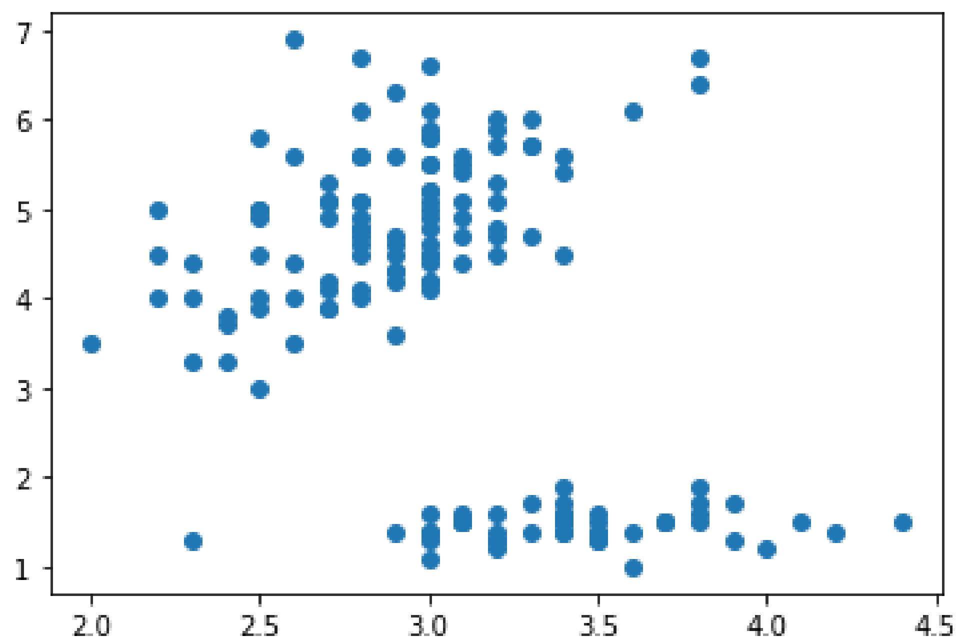
[11]:
```
# 2. scatter plot for sepal length and petal length
scatter_slen_plen = plt.scatter(df['Sepal Length'], df['Petal Length'])
```



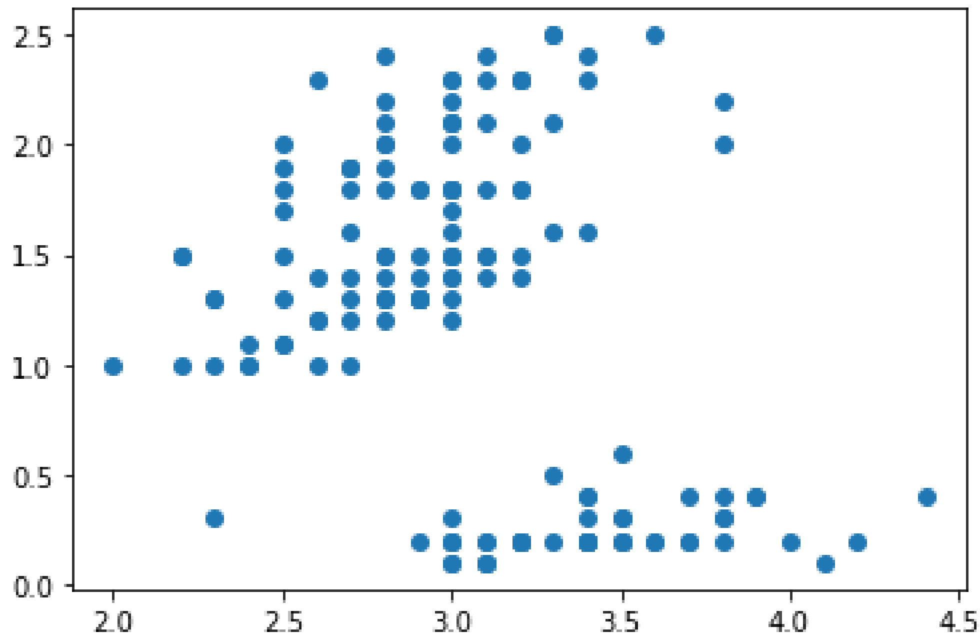
[12]:
```
# 3. scatter plot for sepal length and petal width
scatter_slen_pwid = plt.scatter(df['Sepal Length'], df['Petal Width'])
```



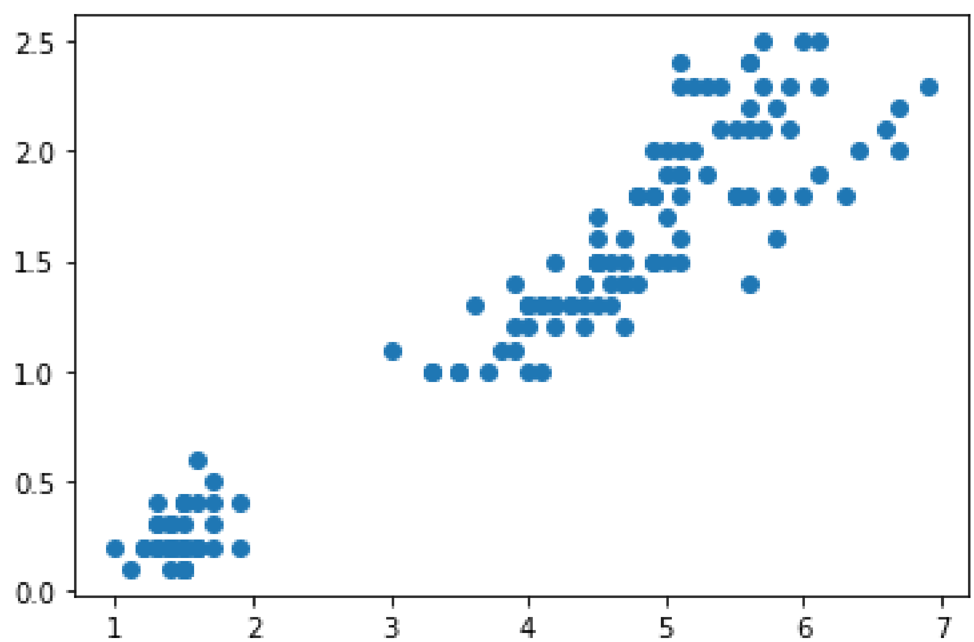
[13]:
```
# 4. scatter plot for sepal width and petal length
scatter_swid_plen = plt.scatter(df['Sepal Width'], df['Petal Length'])
```



[14]:
```
# 5. scatter plot for sepal width and petal width
scatter_swid_pwid = plt.scatter(df['Sepal Width'], df['Petal Width'])
```

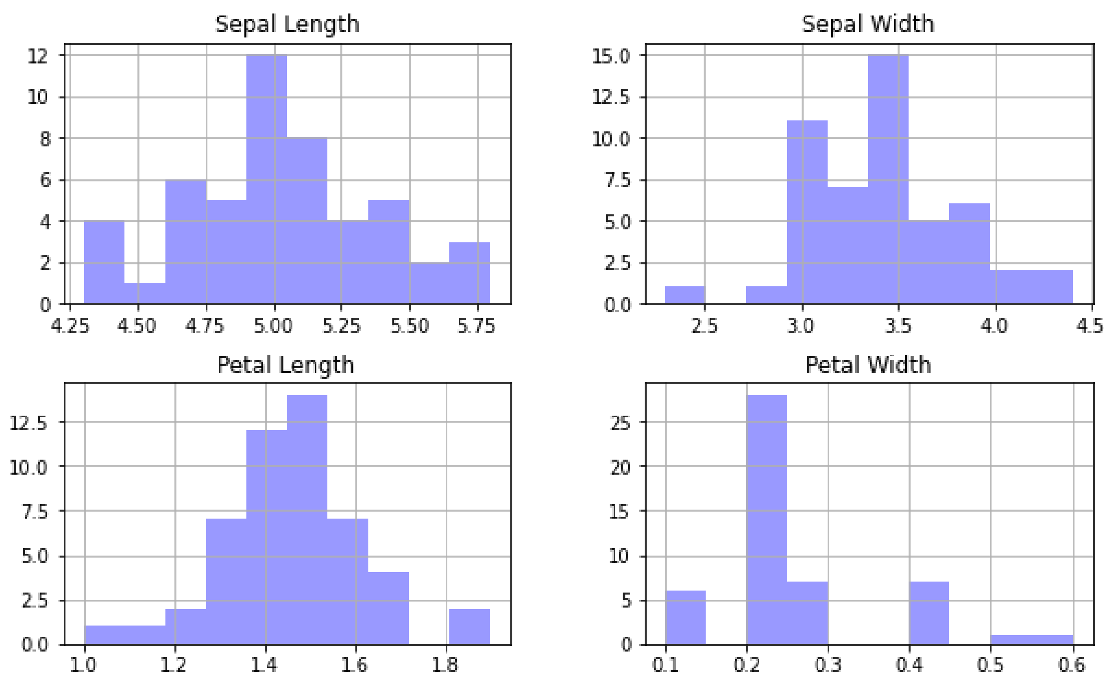


[15]:
```
# 6. scatter plot for petal length and petal width
scatter_plen_pwid = plt.scatter(df['Petal Length'], df['Petal Width'])
```

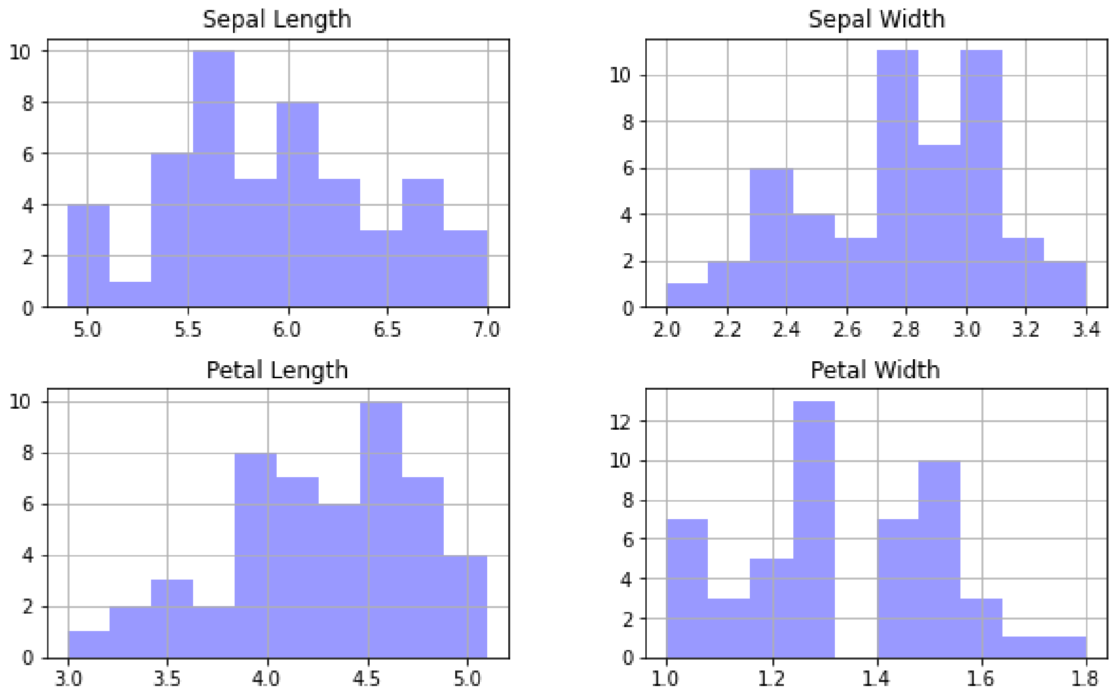


1.1.4 Class-wise Visualization

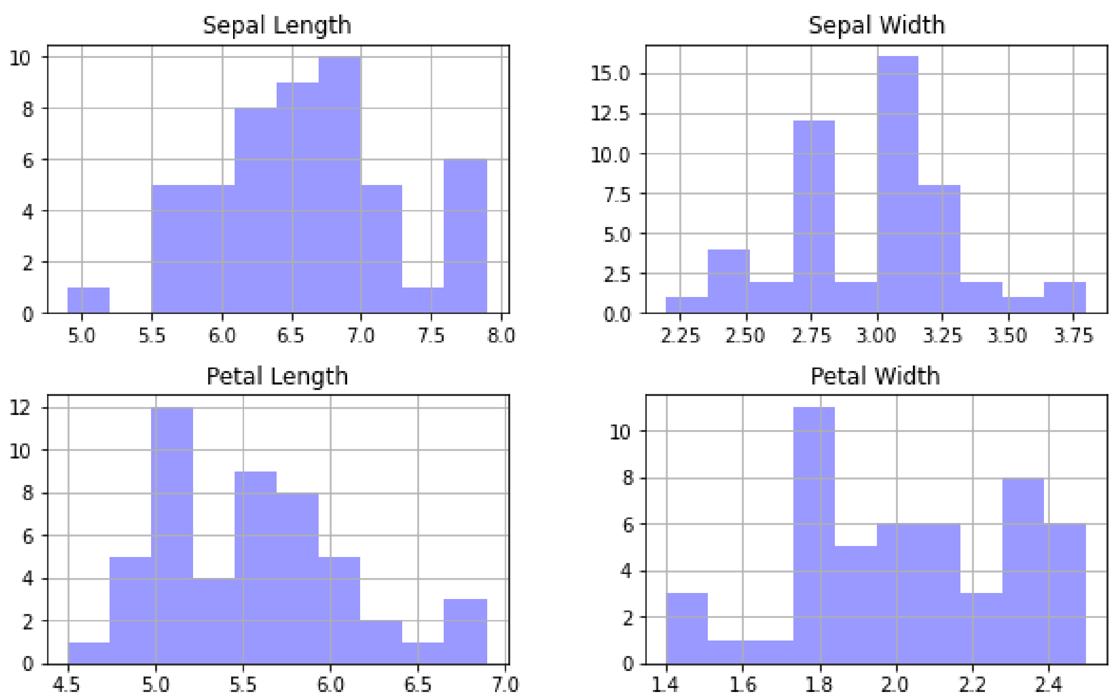
[16]:
```
# Histograms for Iris-setosa class
setosa_df = df.loc[df['Class'] == 'Iris-setosa']
setosa_hist = setosa_df.hist(color='b',alpha=0.4,figsize=(10,6))
```



[17]:
```
# Histograms for Iris-versicolor class
versicolor_df = df.loc[df['Class'] == 'Iris-versicolor']
versicolor_hist = versicolor_df.hist(color='b',alpha=0.4,figsize=(10,6))
```



[18]:
```
# Histograms for Iris-virginica class
virginica_df = df.loc[df['Class'] == 'Iris-virginica']
virginica_hist = virginica_df.hist(color='b',alpha=0.4,figsize=(10,6))
```



1.2 1.3 Analysis

1. There are five features. Petal length, petal width, sepal length and sepal width are numeric features. Class is a nominal feature.
2. The plots for petals are discontinuous, the histograms for petal length can be segmented at around 2.5 (2 - 3). There's a drastic drop between 0.5-1.0 for petal width. But the histograms for sepal length and width are continuous and more close to a bell curve.
3. Sepal length and petal width have significantly different medians as the boxplots for these two features have the smallest overlap degree. Petal length has the greatest amount of data (largest range).
4. Sepal length and petal length, sepal length and petal width, petal length and petal width are most correlated as the scatterplots are more linear.
5. For petal length, the histograms are more like bimodal distribution for the whole dataset, while more like normal distribution (bell curve) for each class. For sepal length, all the histograms are like normal distribution (bell curve), but have more outliers for each class. For sepal width, the histograms for the whole dataset is more like normal distribution, while the histograms for each class are more like multimodal distribution.

2 2 Air Quality Dataset

[19]:
```
# read data and display the first five rows of the data
keys = df.columns.values
df0 = pd.read_csv("AirQualityUCI.csv", sep=";", decimal=',')
df1 = df0.dropna(how='all', axis=1)
df1.head(5)
```

[19]:
```
         Date      Time  CO(GT)  PT08.S1(CO)  NMHC(GT)  C6H6(GT)  \
0  10/03/2004  18.00.00     2.6       1360.0     150.0      11.9
1  10/03/2004  19.00.00     2.0       1292.0     112.0       9.4
2  10/03/2004  20.00.00     2.2       1402.0      88.0       9.0
3  10/03/2004  21.00.00     2.2       1376.0      80.0       9.2
4  10/03/2004  22.00.00     1.6       1272.0      51.0       6.5

   PT08.S2(NMHC)  NOx(GT)  PT08.S3(NOx)  NO2(GT)  PT08.S4(NO2)  PT08.S5(O3)  \
0         1046.0    166.0        1056.0    113.0        1692.0       1268.0
1          955.0    103.0        1174.0     92.0        1559.0        972.0
2          939.0    131.0        1140.0    114.0        1555.0       1074.0
3          948.0    172.0        1092.0    122.0        1584.0       1203.0
4          836.0    131.0        1205.0    116.0        1490.0       1110.0

      T    RH      AH
0  13.6  48.9  0.7578
1  13.3  47.7  0.7255
```

```
2  11.9  54.0  0.7502
3  11.0  60.0  0.7867
4  11.2  59.6  0.7888
```

2.1 2.1 Summary Statistics

[20]:
```python
# display summary statistics for each feature (min, max, mean,
# standard deviation, count and 25:50:75% percentiles)
df1.describe()
```

[20]:
```
            CO(GT)  PT08.S1(CO)     NMHC(GT)     C6H6(GT)  PT08.S2(NMHC)  \
count  9357.000000  9357.000000  9357.000000  9357.000000    9357.000000
mean    -34.207524  1048.990061  -159.090093     1.865683     894.595276
std      77.657170   329.832710   139.789093    41.380206     342.333252
min    -200.000000  -200.000000  -200.000000  -200.000000    -200.000000
25%       0.600000   921.000000  -200.000000     4.000000     711.000000
50%       1.500000  1053.000000  -200.000000     7.900000     895.000000
75%       2.600000  1221.000000  -200.000000    13.600000    1105.000000
max      11.900000  2040.000000  1189.000000    63.700000    2214.000000

           NOx(GT)  PT08.S3(NOx)      NO2(GT)  PT08.S4(NO2)  PT08.S5(O3)  \
count  9357.000000   9357.000000  9357.000000   9357.000000  9357.000000
mean    168.616971    794.990168    58.148873   1391.479641   975.072032
std     257.433866    321.993552   126.940455    467.210125   456.938184
min    -200.000000   -200.000000  -200.000000   -200.000000  -200.000000
25%      50.000000    637.000000    53.000000   1185.000000   700.000000
50%     141.000000    794.000000    96.000000   1446.000000   942.000000
75%     284.000000    960.000000   133.000000   1662.000000  1255.000000
max    1479.000000   2683.000000   340.000000   2775.000000  2523.000000

                 T           RH           AH
count  9357.000000  9357.000000  9357.000000
mean      9.778305    39.485380    -6.837604
std      43.203623    51.216145    38.976670
min    -200.000000  -200.000000  -200.000000
25%      10.900000    34.100000     0.692300
50%      17.200000    48.600000     0.976800
75%      24.100000    61.900000     1.296200
max      44.600000    88.700000     2.231000
```

[21]:
```python
# range
df1[df1.columns[2:15]].max()-df1[df1.columns[2:15]].dropna().min()
```

[21]:
```
CO(GT)             211.900
PT08.S1(CO)       2240.000
NMHC(GT)          1389.000
C6H6(GT)           263.700
```

```
PT08.S2(NMHC)    2414.000
NOx(GT)          1679.000
PT08.S3(NOx)     2883.000
NO2(GT)           540.000
PT08.S4(NO2)     2975.000
PT08.S5(O3)      2723.000
T                 244.600
RH                288.700
AH                202.231
dtype: float64
```

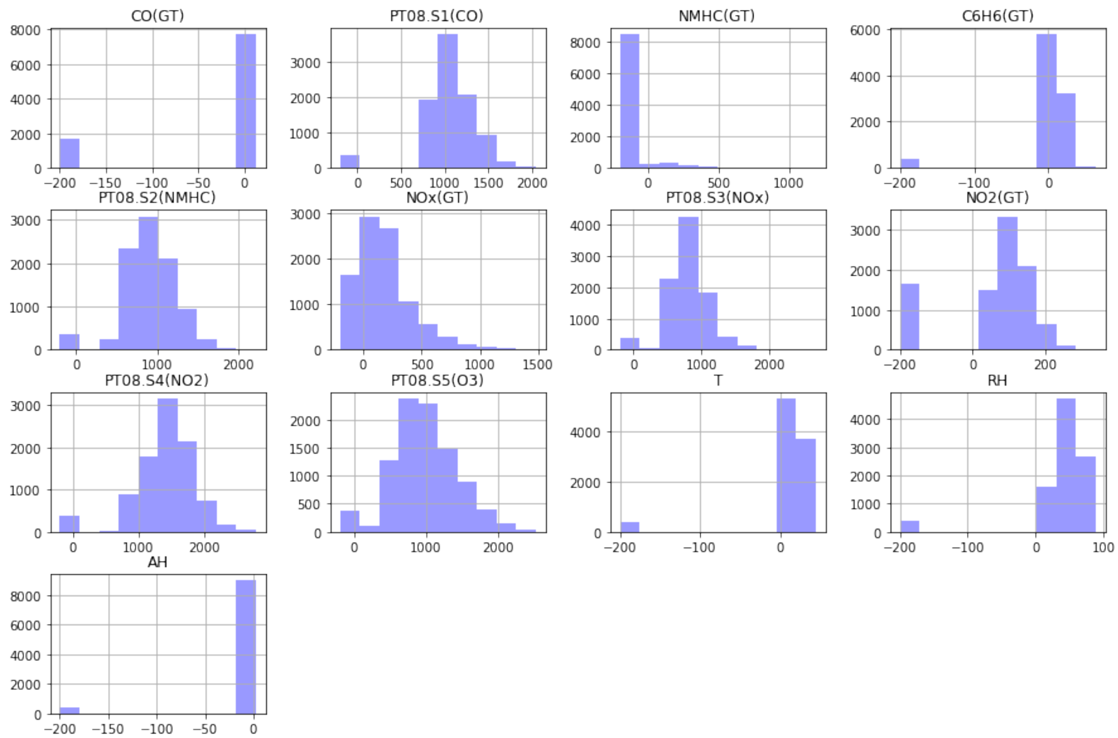
[22]:
```
# variance
df1.var()
```

[22]:
```
CO(GT)             6030.636106
PT08.S1(CO)      108789.616511
NMHC(GT)          19540.990493
C6H6(GT)           1712.321485
PT08.S2(NMHC)    117192.055185
NOx(GT)           66272.195514
PT08.S3(NOx)     103679.847274
NO2(GT)           16113.879181
PT08.S4(NO2)     218285.300489
PT08.S5(O3)      208792.504430
T                  1866.553046
RH                 2623.093506
AH                 1519.180817
dtype: float64
```

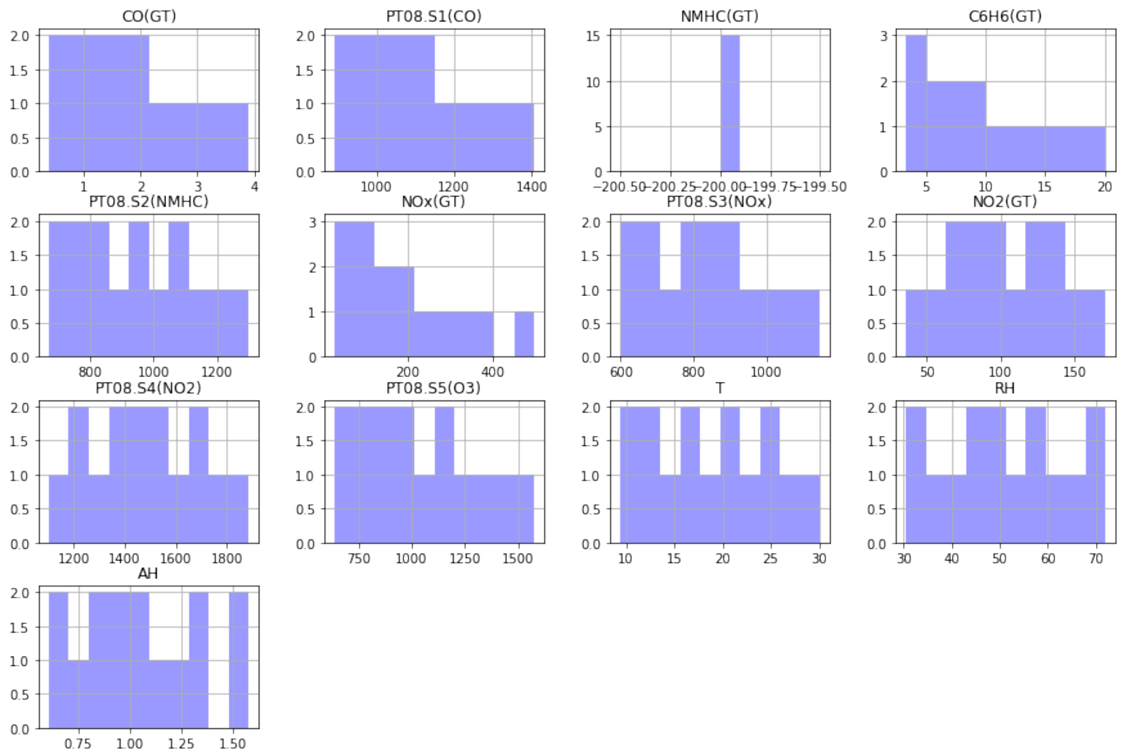
2.2 2.2 Data Visualization

2.2.1 Histograms

[23]:
```
# histograms with ouliers
air_quality_hist = df1.hist(color='b',alpha=0.4,figsize=(15,10))
```

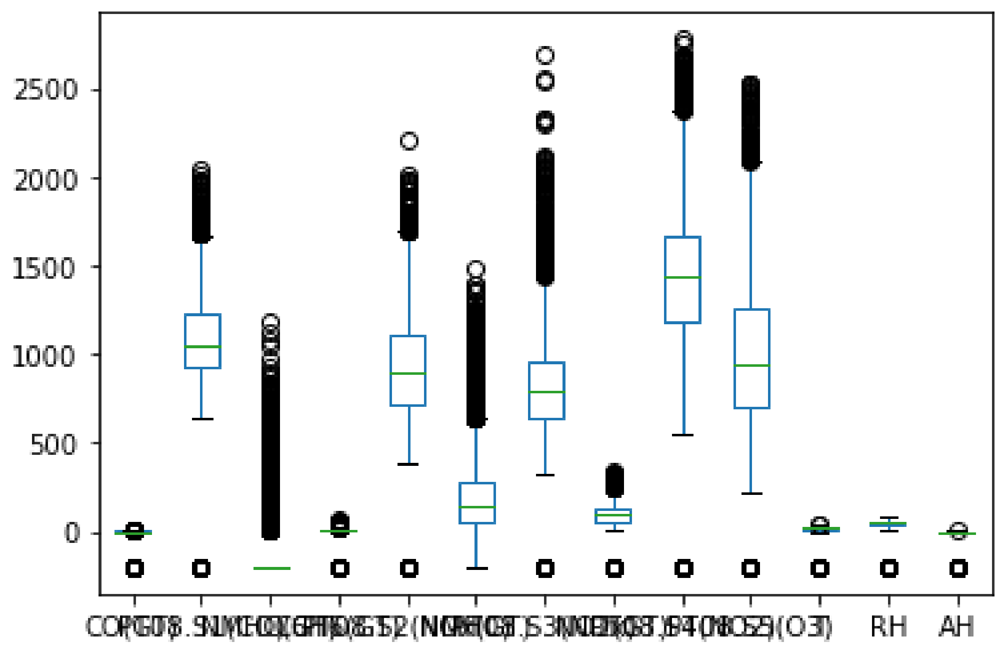


[24]:
```python
# histograms without outliers
# set the lower and upper bound
lower_bound = 0.20
upper_bound = 0.95
# eliminate the outliers outside the bounds
df2 = df1.quantile(np.arange(lower_bound, upper_bound, 0.05))
air_quality_hist_no_outliers = df2.hist(color='b',alpha=0.4,figsize=(15,10))
```



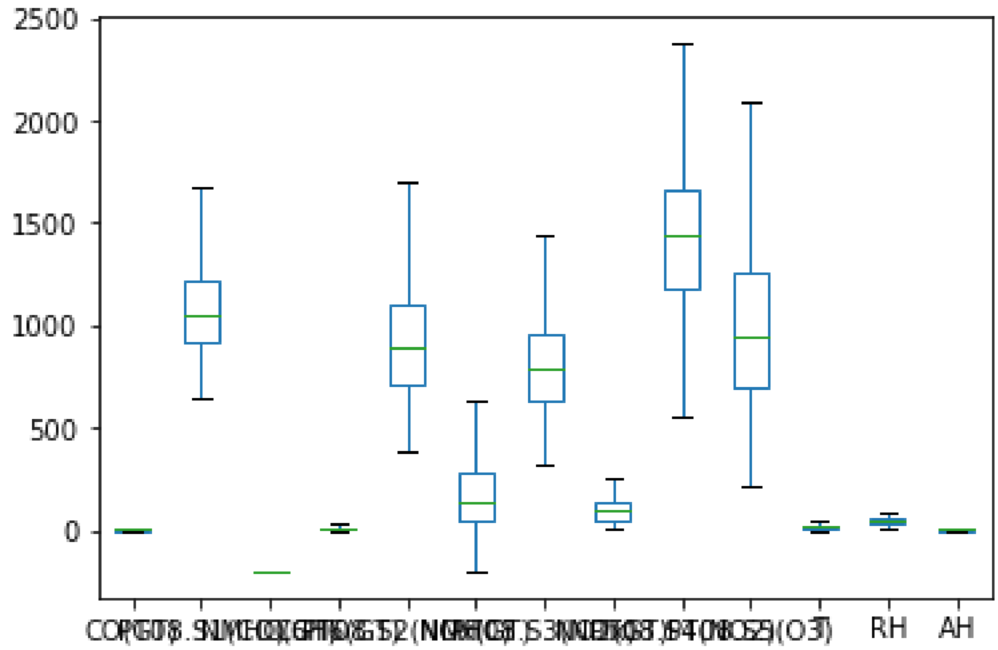
2.2.2 Boxplot

[25]:
```
# boxplots with outliers
box_outliers = df1.boxplot(grid=False, return_type='axes')
```



[26]:

```python
# boxplots without outliers
box_no_outliers = df1.boxplot(grid=False, return_type='axes', showfliers=False)
```



2.3 2.3 Analysis

1. From the histograms: AH, C6H6(GT), CO(GT), NMHC(GT), RH and T are not like normal distributions, the data concentrate on certain range of amount; for NOx(GT), the distribution is skewed; for NO2(GT), PT08.S1(CO), PT08.S2(NMHC), PT08.S3(NOx), PT08.S4(NO2) and PT08.S5(O3), there are several obvious outliers.
2. From the summary statistics: for CO(GT), NO2(GT), and AH, the differences between mean and 50%(median) are large, which means the distributions of data are skewed.For NMHC(GT), the data range is large but 25%-50%-75% and min are all the same.
3. By the elimination of the outliers from the data.
4. The histograms should have a bell shape (normal distribution) after removing the abnormalities from the data.