AQI REPORT

In [88]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import os
```

In [89]:
```
pwd
```

Out[89]:
```
'C:\\Users\\MIDO'
```

In [90]:
```
df = pd.read_csv("C:\\Users\\MIDO\\Downloads\\project.csv")
df.head(10)
```

Out[90]:

	Date	Didian	SO2	NO2	PM10	PM2.5	O3	CO
0	2019/4/16	监测点A	5.0	56.0	36.0	31.0	103.0	0.8
1	2019/4/17	监测点A	9.0	63.0	67.0	53.0	206.0	1.1
2	2019/4/18	监测点A	6.0	37.0	31.0	25.0	83.0	0.6
3	2019/4/19	监测点A	5.0	50.0	23.0	15.0	71.0	0.6
4	2019/4/20	监测点A	6.0	68.0	33.0	26.0	32.0	0.7
5	2019/4/21	监测点A	4.0	53.0	41.0	27.0	35.0	0.7
6	2019/4/22	监测点A	4.0	32.0	32.0	16.0	64.0	0.6
7	2019/4/23	监测点A	6.0	27.0	35.0	17.0	65.0	0.5
8	2019/4/24	监测点A	5.0	20.0	NaN	16.0	85.0	0.6
9	2019/4/25	监测点A	7.0	23.0	36.0	19.0	79.0	0.7

In [91]:
```
df.tail(5)
```

Out[91]:

	Date	Didian	SO2	NO2	PM10	PM2.5	O3	CO
814	2021/7/8	监测点A	5.0	15.0	18.0	3.0	64.0	0.4
815	2021/7/9	监测点A	7.0	17.0	28.0	12.0	146.0	0.4
816	2021/7/10	监测点A	6.0	13.0	20.0	5.0	81.0	0.4
817	2021/7/11	监测点A	6.0	11.0	20.0	3.0	63.0	0.3
818	2021/7/12	监测点A	6.0	11.0	17.0	5.0	81.0	0.4

In [92]:
```
df.shape
```

Out[92]:
```
(819, 8)
```

In [93]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 819 entries, 0 to 818
Data columns (total 8 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Date    819 non-null    object
 1   Didian  819 non-null    object
 2   SO2     815 non-null    float64
 3   NO2     813 non-null    float64
 4   PM10    808 non-null    float64
 5   PM2.5   814 non-null    float64
 6   O3      812 non-null    float64
 7   CO      813 non-null    float64
dtypes: float64(6), object(2)
memory usage: 51.3+ KB
```

In [94]:
```
df.describe()
```

Out[94]:

	SO2	NO2	PM10	PM2.5	O3	CO
count	815.000000	813.000000	808.000000	814.000000	812.000000	813.000000
mean	7.011043	32.824108	44.121287	24.228501	98.554187	0.715375
std	3.061792	18.727494	23.259436	22.269055	51.992371	0.206775
min	1.000000	4.000000	5.000000	2.000000	2.000000	0.300000
25%	5.000000	20.000000	27.000000	11.000000	61.000000	0.600000
50%	6.000000	29.000000	38.000000	21.000000	87.000000	0.700000
75%	9.000000	41.000000	56.000000	32.750000	128.250000	0.800000
max	20.000000	132.000000	143.000000	465.000000	296.000000	1.500000

In [95]:
```
missing_values_count = df.isnull().sum()
missing_values_count[0:10]
```

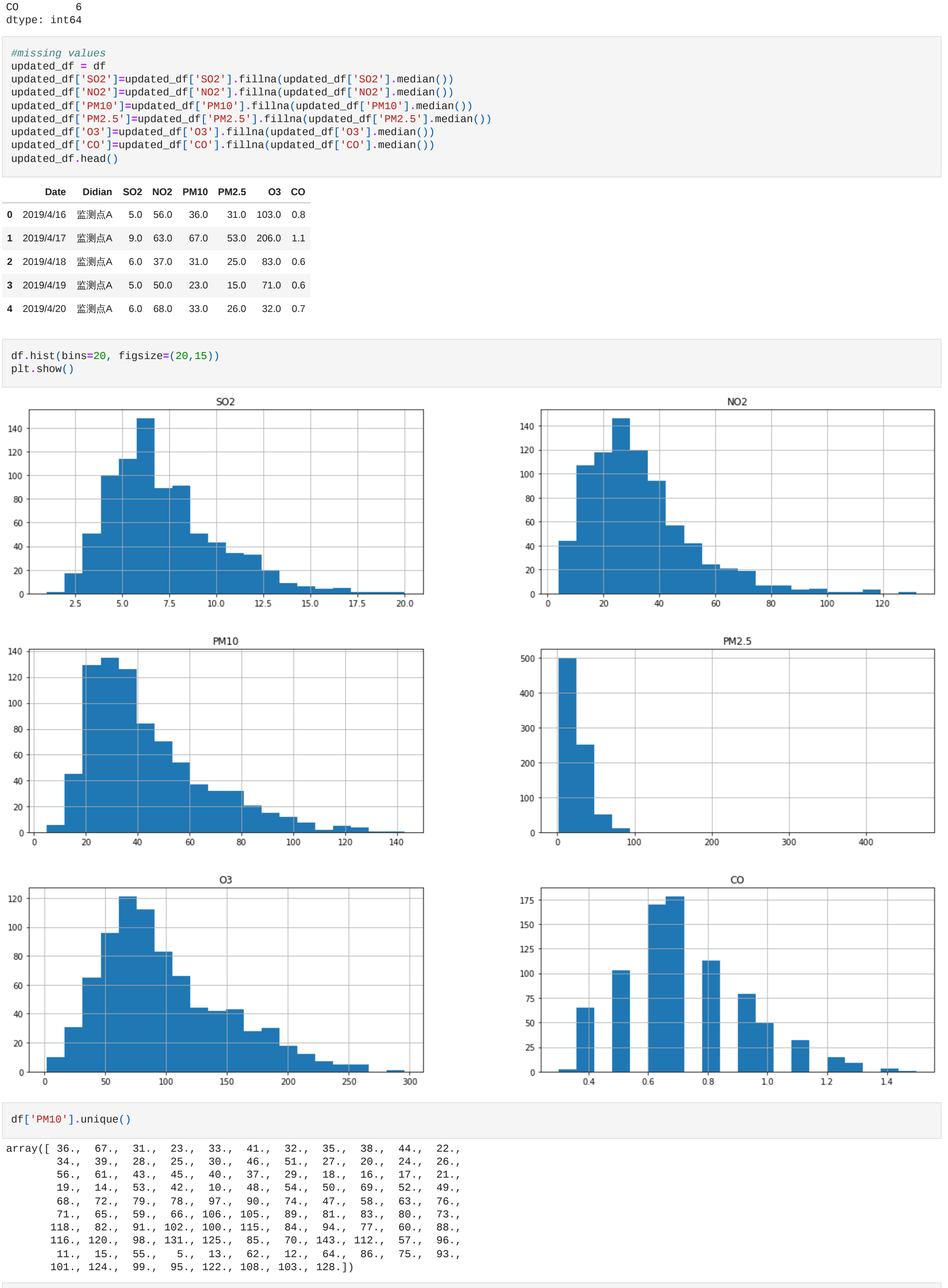
Out[95]:
```
Date       0
Didian     0
SO2        4
NO2        6
PM10      11
PM2.5      5
O3         7
CO         6
dtype: int64
```

In [96]:
```
#missing values
updated_df = df
updated_df['SO2']=updated_df['SO2'].fillna(updated_df['SO2'].median())
updated_df['NO2']=updated_df['NO2'].fillna(updated_df['NO2'].median())
updated_df['PM10']=updated_df['PM10'].fillna(updated_df['PM10'].median())
updated_df['PM2.5']=updated_df['PM2.5'].fillna(updated_df['PM2.5'].median())
updated_df['O3']=updated_df['O3'].fillna(updated_df['O3'].median())
updated_df['CO']=updated_df['CO'].fillna(updated_df['CO'].median())
updated_df.head()
```

Out[96]:

	Date	Didian	SO2	NO2	PM10	PM2.5	O3	CO
0	2019/4/16	监测点A	5.0	56.0	36.0	31.0	103.0	0.8
1	2019/4/17	监测点A	9.0	63.0	67.0	53.0	206.0	1.1
2	2019/4/18	监测点A	6.0	37.0	31.0	25.0	83.0	0.6
3	2019/4/19	监测点A	5.0	50.0	23.0	15.0	71.0	0.6
4	2019/4/20	监测点A	6.0	68.0	33.0	26.0	32.0	0.7

In [97]:
```
df.hist(bins=20, figsize=(20,15))
plt.show()
```



In [98]:
```
df['PM10'].unique()
```

Out[98]:
```
array([ 36.,  67.,  31.,  23.,  33.,  41.,  32.,  35.,  38.,  44.,  22.,
        34.,  39.,  28.,  25.,  30.,  46.,  51.,  27.,  20.,  24.,  26.,
        56.,  61.,  43.,  45.,  40.,  37.,  29.,  18.,  16.,  17.,  21.,
        19.,  14.,  53.,  42.,  10.,  48.,  54.,  50.,  69.,  52.,  49.,
        68.,  72.,  79.,  78.,  97.,  90.,  74.,  47.,  58.,  63.,  76.,
        71.,  65.,  59.,  66., 106., 105.,  89.,  81.,  83.,  80.,  73.,
       118.,  82.,  91., 102., 100., 115.,  84.,  94.,  77.,  60.,  88.,
       116., 120.,  98., 131., 125.,  85.,  70., 143., 112.,  57.,  96.,
        11.,  15.,  55.,   5.,  13.,  62.,  12.,  64.,  86.,  75.,  93.,
       101., 124.,  99.,  95., 122., 108., 103., 128.])
```

In [99]:
```
## PM2.5 Sub-Index calculation
def get_PM25_subindex(x):
    if x <= 30:
        return x * 50 / 30
    elif x <= 60:
        return 50 + (x - 30) * 50 / 30
    elif x <= 90:
        return 100 + (x - 60) * 100 / 30
    elif x <= 120:
        return 200 + (x - 90) * 100 / 30
    elif x <= 250:
        return 300 + (x - 120) * 100 / 130
    elif x > 250:
        return 400 + (x - 250) * 100 / 130
    else:
        return 0

df["PM2.5_SubIndex"] = df["PM2.5"].apply(lambda x: get_PM25_subindex(x))
```

In [100…
```
## PM10 Sub-Index calculation
def get_PM10_subindex(x):
    if x <= 50:
        return x
    elif x <= 100:
        return x
    elif x <= 250:
        return 100 + (x - 100) * 100 / 150
    elif x <= 350:
        return 200 + (x - 250)
    elif x <= 430:
        return 300 + (x - 350) * 100 / 80
    elif x > 430:
        return 400 + (x - 430) * 100 / 80
    else:
        return 0

df["PM10_SubIndex"] = df["PM10"].apply(lambda x: get_PM10_subindex(x))
```

In [101…
```
## SO2 Sub-Index calculation
def get_SO2_subindex(x):
    if x <= 40:
        return x * 50 / 40
    elif x <= 80:
        return 50 + (x - 40) * 50 / 40
    elif x <= 380:
        return 100 + (x - 80) * 100 / 300
    elif x <= 800:
        return 200 + (x - 380) * 100 / 420
    elif x <= 1600:
        return 300 + (x - 800) * 100 / 800
    elif x > 1600:
        return 400 + (x - 1600) * 100 / 800
    else:
        return 0

df["SO2_SubIndex"] = df["SO2"].apply(lambda x: get_SO2_subindex(x))
```

In [102…
```
## NOx Sub-Index calculation
def get_NO2_subindex(x):
    if x <= 40:
        return x * 50 / 40
    elif x <= 80:
        return 50 + (x - 40) * 50 / 40
    elif x <= 180:
        return 100 + (x - 80) * 100 / 100
    elif x <= 280:
        return 200 + (x - 180) * 100 / 100
    elif x <= 400:
        return 300 + (x - 280) * 100 / 120
    elif x > 400:
        return 400 + (x - 400) * 100 / 120
    else:
        return 0

df["NO2_SubIndex"] = df["NO2"].apply(lambda x: get_NO2_subindex(x))
```

In [103…
```
## CO Sub-Index calculation
def get_CO_subindex(x):
    if x <= 1:
        return x * 50 / 1
    elif x <= 2:
        return 50 + (x - 1) * 50 / 1
    elif x <= 10:
        return 100 + (x - 2) * 100 / 8
    elif x <= 17:
        return 200 + (x - 10) * 100 / 7
    elif x <= 34:
        return 300 + (x - 17) * 100 / 17
    elif x > 34:
        return 400 + (x - 34) * 100 / 17
    else:
        return 0

df["CO_SubIndex"] = df["CO"].apply(lambda x: get_CO_subindex(x))
```

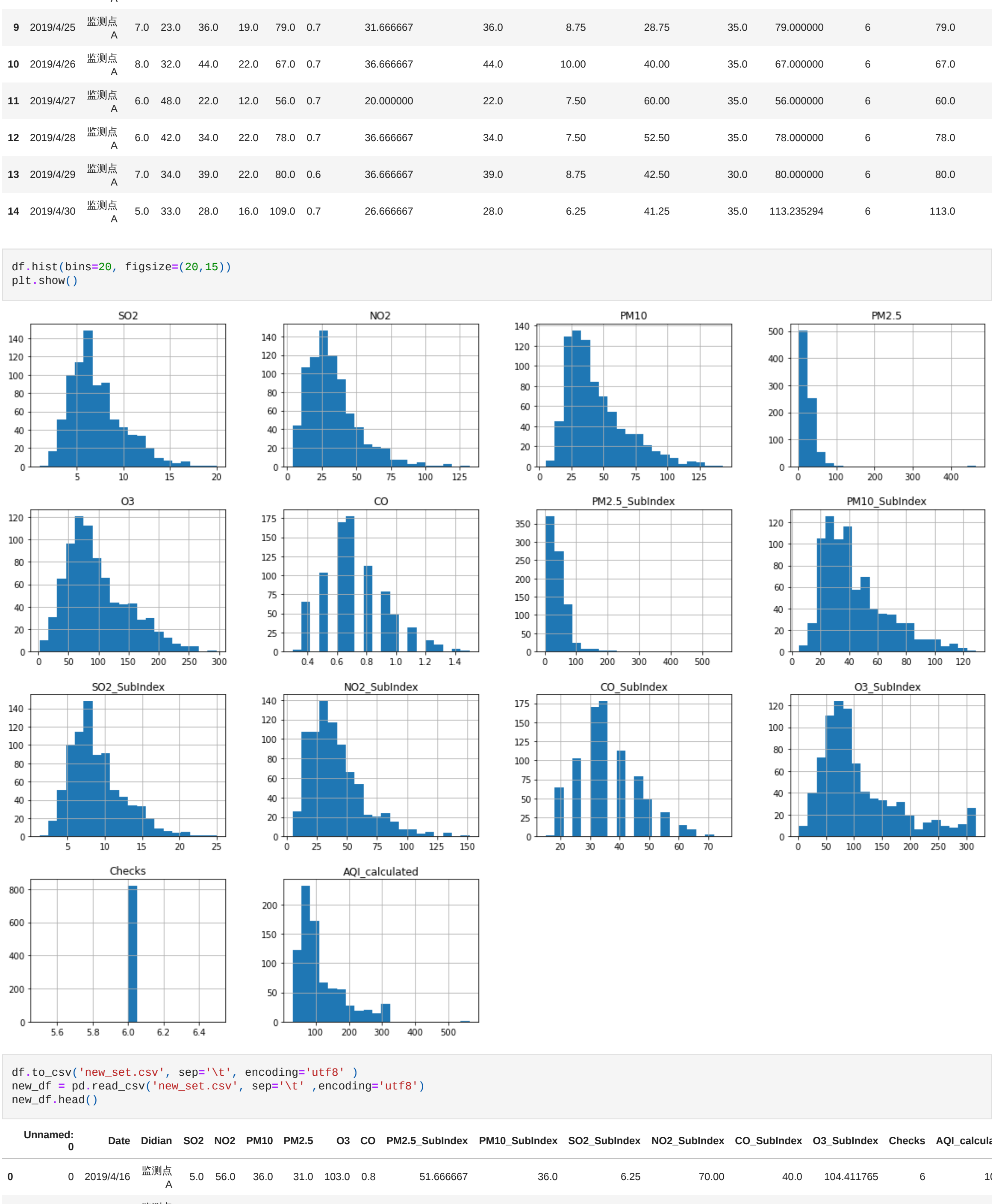
In [104…
```
## O3 Sub-Index calculation
def get_O3_subindex(x):
    if x <= 50:
        return x * 50 / 50
    elif x <= 100:
        return 50 + (x - 50) * 50 / 50
    elif x <= 168:
        return 100 + (x - 100) * 100 / 68
    elif x <= 208:
        return 200 + (x - 168) * 100 / 40
    elif x <= 748:
        return 300 + (x - 208) * 100 / 539
    elif x > 748:
        return 400 + (x - 400) * 100 / 539
    else:
        return 0

df["O3_SubIndex"] = df["O3"].apply(lambda x: get_O3_subindex(x))
```

In [105…
```
## AQI bucketing
def get_AQI_bucket(x):
    if x <= 50:
        return "优 Good"
    elif x <= 100:
        return "良 Satisfactory"
    elif x <= 150:
        return "轻度污染 Moderate"
    elif x <= 200:
        return "中度污染 Poor"
    elif x <= 300:
        return "重度污染"
    elif x > 300:
        return "严重污染 Severe"
    else:
        return np.NaN

df["Checks"] = (df["PM2.5_SubIndex"] > 0).astype(int) + \
               (df["PM10_SubIndex"] > 0).astype(int) + \
               (df["SO2_SubIndex"] > 0).astype(int) + \
               (df["NO2_SubIndex"] > 0).astype(int) + \
               (df["CO_SubIndex"] > 0).astype(int) + \
               (df["O3_SubIndex"] > 0).astype(int)

df["AQI_calculated"] = round(df[["PM2.5_SubIndex", "PM10_SubIndex", "SO2_SubIndex", "NO2_SubIndex",
                                 "CO_SubIndex", "O3_SubIndex"]].max(axis = 1))

df.loc[df["PM2.5_SubIndex"] + df["PM10_SubIndex"] <= 0, "AQI_calculated"] = np.NaN
df.loc[df.Checks < 3, "AQI_calculated"] = np.NaN

df["AQI_bucket_calculated"] = df["AQI_calculated"].apply(lambda x: get_AQI_bucket(x))
df[~df.AQI_calculated.isna()].head(15)
```

Out[105…

	Date	Didian	SO2	NO2	PM10	PM2.5	O3	CO	PM2.5_SubIndex	PM10_SubIndex	SO2_SubIndex	NO2_SubIndex	CO_SubIndex	O3_SubIndex	Checks	AQI_calculated	AQI_l
0	2019/4/16	监测点A	5.0	56.0	36.0	31.0	103.0	0.8	51.666667	36.0	6.25	70.00	40.0	104.411765	6	104.0	
1	2019/4/17	监测点A	9.0	63.0	67.0	53.0	206.0	1.1	88.333333	67.0	11.25	78.75	55.0	295.000000	6	295.0	
2	2019/4/18	监测点A	6.0	37.0	31.0	25.0	83.0	0.6	41.666667	31.0	7.50	46.25	30.0	83.000000	6	83.0	
3	2019/4/19	监测点A	5.0	50.0	23.0	15.0	71.0	0.6	25.000000	23.0	6.25	62.50	30.0	71.000000	6	71.0	
4	2019/4/20	监测点A	6.0	68.0	33.0	26.0	32.0	0.7	43.333333	33.0	7.50	85.00	35.0	32.000000	6	85.0	
5	2019/4/21	监测点A	4.0	53.0	41.0	27.0	35.0	0.7	45.000000	41.0	5.00	66.25	35.0	35.000000	6	66.0	
6	2019/4/22	监测点A	4.0	32.0	32.0	16.0	64.0	0.6	26.666667	32.0	5.00	40.00	30.0	64.000000	6	64.0	
7	2019/4/23	监测点A	6.0	27.0	35.0	17.0	65.0	0.5	28.333333	35.0	7.50	33.75	25.0	65.000000	6	65.0	
8	2019/4/24	监测点A	5.0	20.0	38.0	16.0	85.0	0.6	26.666667	38.0	6.25	25.00	30.0	85.000000	6	85.0	
9	2019/4/25	监测点A	7.0	23.0	36.0	19.0	79.0	0.7	31.666667	36.0	8.75	28.75	35.0	79.000000	6	79.0	
10	2019/4/26	监测点A	8.0	32.0	44.0	22.0	67.0	0.7	36.666667	44.0	10.00	40.00	35.0	67.000000	6	67.0	
11	2019/4/27	监测点A	6.0	48.0	22.0	12.0	56.0	0.7	20.000000	22.0	7.50	60.00	35.0	56.000000	6	60.0	
12	2019/4/28	监测点A	6.0	42.0	34.0	22.0	78.0	0.7	36.666667	34.0	7.50	52.50	35.0	78.000000	6	78.0	
13	2019/4/29	监测点A	7.0	34.0	39.0	22.0	80.0	0.6	36.666667	39.0	8.75	42.50	30.0	80.000000	6	80.0	
14	2019/4/30	监测点A	5.0	33.0	28.0	16.0	109.0	0.7	26.666667	28.0	6.25	41.25	35.0	113.235294	6	113.0	

In [106…
```
df.hist(bins=20, figsize=(20,15))
plt.show()
```



In [107…
```
df.to_csv('new_set.csv', sep='\t', encoding='utf8' )
new_df = pd.read_csv('new_set.csv', sep='\t' ,encoding='utf8')
new_df.head()
```

Out[107…

	Unnamed: 0	Date	Didian	SO2	NO2	PM10	PM2.5	O3	CO	PM2.5_SubIndex	PM10_SubIndex	SO2_SubIndex	NO2_SubIndex	CO_SubIndex	O3_SubIndex	Checks	AQI_calcula
0	0	2019/4/16	监测点A	5.0	56.0	36.0	31.0	103.0	0.8	51.666667	36.0	6.25	70.00	40.0	104.411765	6	1
1	1	2019/4/17	监测点A	9.0	63.0	67.0	53.0	206.0	1.1	88.333333	67.0	11.25	78.75	55.0	295.000000	6	2
2	2	2019/4/18	监测点A	6.0	37.0	31.0	25.0	83.0	0.6	41.666667	31.0	7.50	46.25	30.0	83.000000	6	8
3	3	2019/4/19	监测点A	5.0	50.0	23.0	15.0	71.0	0.6	25.000000	23.0	6.25	62.50	30.0	71.000000	6	7
4	4	2019/4/20	监测点A	6.0	68.0	33.0	26.0	32.0	0.7	43.333333	33.0	7.50	85.00	35.0	32.000000	6	8

How do we calculate AQI?

1.The Sub-indices for individual pollutants at a monitoring location are calculated using its24-hourly average concentration value (8-hourly in case of CO and O3) and health breakpoint concentration range. The worst sub-index is the AQI for that location.

2.All the six pollutants may not be monitored at all the locations. Overall AQI iscalculated only if data are available for minimum three pollutants out of which one should necessarily be either PM2.5 or PM10. Else, data are considered insufficient for calculating AQI. Similarly, a minimum of 16 hours' data is considered necessary for calculating subindex.

3.The sub-indices for monitored pollutants are calculated and disseminated, even if data are inadequate for determining AQI. The Individual pollutant-wise sub-index will provide air quality status for that pollutant.

4.The web-based system is designed to provide AQI on real time basis. It is an automated system that captures data from continuous monitoring stations without human intervention, and displays AQI based on running average values (e.g. AQI at 6am on a day will incorporate data from 6am on previous day to the current day).

5.For manual monitoring stations, an AQI calculator is developed wherein data can be fed manually to get AQI value.

analyses of project

According to this project , the monthly average of the Air Quality Index for monitoring the heating season in this city and the monthly concentration of major atmospheric pollutants in this three years, the relationship between five major atmospheric pollutants and meteorological conditions was analyzed, the main results are as follows:

(1) in the past three years, the air quality in this city heating season was mild pollution, and the most serious degree of pollution was in January each year.

(2) The main pollutants in this city were PM2.5 and PM10, but also the content of pollutants NO2 and O3 has risen sharply, so we should strengthen the protection and the treatment of NO2 and O3.

(3) The precipitation has a certain impact on air quality index.

(4) The air quality in city is closely related to meteorological elements.

In [108…
```
df.describe()
```

Out[108…

	SO2	NO2	PM10	PM2.5	O3	CO	PM2.5_SubIndex	PM10_SubIndex	SO2_SubIndex	NO2_SubIndex	CO_SubIndex	O3_SubIndex	Checks	A
count	819.000000	819.000000	819.000000	819.000000	819.000000	819.000000	819.000000	819.000000	819.000000	819.000000	819.000000	819.000000	819.0	
mean	7.006105	32.796093	44.039072	24.208791	98.455433	0.715263	40.696440	43.903541	8.757631	40.890415	35.763126	109.744451	6.0	
std	3.055111	18.661538	23.113273	22.202318	51.780370	0.206019	34.331254	22.667357	3.818889	22.918649	10.300975	71.029457	0.0	
min	1.000000	4.000000	5.000000	2.000000	2.000000	0.300000	3.333333	5.000000	1.250000	5.000000	15.000000	2.000000	6.0	
25%	5.000000	20.000000	27.000000	11.000000	61.000000	0.600000	18.333333	27.000000	6.250000	25.000000	30.000000	61.000000	6.0	
50%	6.000000	29.000000	38.000000	21.000000	87.000000	0.700000	35.000000	38.000000	7.500000	36.250000	35.000000	87.000000	6.0	
75%	9.000000	41.000000	56.000000	32.000000	128.000000	0.800000	53.333333	56.000000	11.250000	51.250000	40.000000	141.176471	6.0	
max	20.000000	132.000000	143.000000	465.000000	296.000000	1.500000	565.384615	128.666667	25.000000	152.000000	75.000000	316.326531	6.0	