

```
import pandas as pd
```

▼ Importing the dataset.

The dataset is taken from kaggle(<https://www.kaggle.com/datasets/fedesoriano/air-quality-data-in-india>). The dataset presents to us data for air quality index from the year 2017 to 2022 in India. For each date(from 2017-11-07 to 2022-06-04) the air quality index is measured on an hourly basis. So there are 24 entries for each day starting from 00:00 hrs to 24:00 hrs with an interval of 1 hour.

What is an AQI?

An AQI number is assigned based on the air pollutant with the highest AQI number at the moment the air quality is measured.

The index represents air pollutant concentrations with a number falling within a range of air quality categories. Within each category and number range, elevated health risks associated with rising air pollutant concentrations are identified.

The air quality index ranges from 0 to 500, though air quality can be indexed beyond 500 when there are higher levels of hazardous air pollution. Good air quality ranges from 0 to 50, while measurements over 300 are considered hazardous.

IQAir AirVisual platform AQI readings are based on the U.S. Environmental Protection Agency (EPA) National Ambient Air Quality Standards (NAAQS) to calculate AQI and to attribute code color.4,5 AirVisual Series air quality monitors measure PM2.5, PM1, PM10, and carbon dioxide levels and use PM2.5, or fine particulate matter, to determine the AQI.

The AirVisual Series monitors AQI using PM2.5 measurements as the determinant for AQI readings because PM2.5 is widely available and considered the most hazardous air pollutant impacting human health.

PM2.5 is measured by micrograms per cubic meter (µg/m3). According to the U.S. EPA NAAQS, any measurement greater than 12.0 µg/m3 (US AQI 50) can be dangerous to human health.

(Source: <https://www.iqair.com/blog/air-quality/what-is-aqi>)

```
from google.colab import files
```

```
uploaded = files.upload()
```

Choose Files

No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.


Saving air-quality-india.csv to air-quality-india (2).csv

```
import io
aq_data = pd.read_csv(io.BytesIO(uploaded['air-quality-india.csv']))
```

```
aq_data.head(10)
```

	Timestamp	Year	Month	Day	Hour	PM2.5	
0	2017-11-07 12:00:00	2017	11	7	12	64.51	
1	2017-11-07 13:00:00	2017	11	7	13	69.95	
2	2017-11-07 14:00:00	2017	11	7	14	92.79	
3	2017-11-07 15:00:00	2017	11	7	15	109.66	
4	2017-11-07 16:00:00	2017	11	7	16	116.50	
5	2017-11-07 17:00:00	2017	11	7	17	124.21	
6	2017-11-07 18:00:00	2017	11	7	18	123.45	
7	2017-11-07 19:00:00	2017	11	7	19	120.31	
8	2017-11-07 20:00:00	2017	11	7	20	108.75	
9	2017-11-07 21:00:00	2017	11	7	21	97.80	

```
aq_data.tail(10)
```

	Timestamp	Year	Month	Day	Hour	PM2.5	
36182	2022-06-04 06:00:00	2022	6	4	6	41.89	
36183	2022-06-04 07:00:00	2022	6	4	7	38.34	
36184	2022-06-04 08:00:00	2022	6	4	8	36.63	
36185	2022-06-04 09:00:00	2022	6	4	9	35.84	
36186	2022-06-04 10:00:00	2022	6	4	10	34.71	
36187	2022-06-04 11:00:00	2022	6	4	11	35.89	

`aq_data.info()` # brief information about the data

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36192 entries, 0 to 36191
Data columns (total 6 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Timestamp    36192 non-null  object
1   Year         36192 non-null  int64
2   Month        36192 non-null  int64
3   Day          36192 non-null  int64
4   Hour         36192 non-null  int64
5   PM2.5        36192 non-null  float64
dtypes: float64(1), int64(4), object(1)
memory usage: 1.7+ MB
```

So, there are 36192 non-null hours and for each hour the PM2.5 count is recorded.

https://www2.igair.com/sites/default/files/inline-images/AQI%29Chart_US.png

The link given above is used to convert the PM2.5 count to US AQI level. For simplicity and modelling purpose I have considered the levels as integers. The code for the same is given below.

The mapping is as follows:

1:Good; 2:Moderate; 3:Unhealthy for Sensitive Groups; 4:Unhealthy; 5:Very unhealthy; 6:Hazardous. (Compare these with the newly created column "AQI_level")

```
AQI_level=[]
for i in aq_data.index:
    if aq_data['PM2.5'][i] >=0 and aq_data['PM2.5'][i]<=12:
        AQI_level.append(1)
    elif aq_data['PM2.5'][i]>12 and aq_data['PM2.5'][i]<=35.4:
        AQI_level.append(2)
    elif aq_data['PM2.5'][i]>35.4 and aq_data['PM2.5'][i]<=55.4:
        AQI_level.append(3)
    elif aq_data['PM2.5'][i]>55.4 and aq_data['PM2.5'][i]<=150.4:
        AQI_level.append(4)
    elif aq_data['PM2.5'][i]>150.4 and aq_data['PM2.5'][i]<=250.4:
        AQI_level.append(5)
    else:
        AQI_level.append(6)
print(len(AQI_level))

36192
```

```
aq_data.insert(6,"AQI_level",AQI_level,True) #inserts the new column AQI_level
aq_data.head(10)
```

	Timestamp	Year	Month	Day	Hour	PM2.5	AQI_level	
0	2017-11-07 12:00:00	2017	11	7	12	64.51	4	
1	2017-11-07 13:00:00	2017	11	7	13	69.95	4	
2	2017-11-07 14:00:00	2017	11	7	14	92.79	4	

To model this data, I need to see that how many hours in a day does the "AQI_level" remain equal to 1(i.e the AQI level is Good). For this I have followed two steps:

Step 1: Combine the Year, Month and the Day to form a single date.

Step 2: Mark the hours in a day where the level is equal to1 and count the number of such hours for that day (among 24 hours)

```
3 2017-11-07 15:00:00 2017 11 7 15 109.66 4
#combine Year, Month and Day to form date
Date=[]
for i in aq_data.index:
    date_str = str(aq_data['Year'][i])+'-'+str(aq_data['Month'][i])+'-'+str(aq_data['Day'][i])
    Date.append(date_str)
print(len(Date))

36192
```

```
aq_data.insert(5,"Date",Date,True) #inserts new column Date
aq_data.head(10)
```

	Timestamp	Year	Month	Day	Hour	Date	PM2.5	AQI_level	
0	2017-11-07 12:00:00	2017	11	7	12	2017-11-7	64.51	4	
1	2017-11-07 13:00:00	2017	11	7	13	2017-11-7	69.95	4	
2	2017-11-07 14:00:00	2017	11	7	14	2017-11-7	92.79	4	
3	2017-11-07 15:00:00	2017	11	7	15	2017-11-7	109.66	4	
4	2017-11-07 16:00:00	2017	11	7	16	2017-11-7	116.50	4	
5	2017-11-07 17:00:00	2017	11	7	17	2017-11-7	124.21	4	
6	2017-11-07 18:00:00	2017	11	7	18	2017-11-7	123.45	4	
7	2017-11-07 19:00:00	2017	11	7	19	2017-11-7	120.31	4	
8	2017-11-07 20:00:00	2017	11	7	20	2017-11-7	108.75	4	
9	2017-11-07 21:00:00	2017	11	7	21	2017-11-7	97.80	4	

```
# Mark the hours with '1' if they have the level equal to 1
AQI_level_01=[]
for i in aq_data.index:
    if aq_data['AQI_level'][i] ==1:
        AQI_level_01.append(1)
    else:
        AQI_level_01.append(0)
print(len(AQI_level_01))

36192
```

```
aq_data.insert(8,"PM2.5=1?",AQI_level_01,True)
aq_data.tail(10)
```

	Timestamp	Year	Month	Day	Hour	Date	PM2.5	AQI_level	PM2.5=1?	
36182	2022-06-04 06:00:00	2022	6	4	6	2022-6-4	41.89	3	0	
36183	2022-06-04 07:00:00	2022	6	4	7	2022-6-4	38.34	3	0	

aq_data.head(5)

	Timestamp	Year	Month	Day	Hour	Date	PM2.5	AQI_level	PM2.5=1?	
0	2017-11-07 12:00:00	2017	11	7	12	2017-11-7	64.51	4	0	
1	2017-11-07 13:00:00	2017	11	7	13	2017-11-7	69.95	4	0	
2	2017-11-07 14:00:00	2017	11	7	14	2017-11-7	92.79	4	0	
3	2017-11-07 15:00:00	2017	11	7	15	2017-11-7	109.66	4	0	
4	2017-11-07 16:00:00	2017	11	7	16	2017-11-7	116.50	4	0	

aq_data['PM2.5=1?'].value_counts()

```
0    36140
1         52
Name: PM2.5=1?, dtype: int64
```

So there are 12568 hours in total where the level is les than 3 and for the rest of the hours (i.e. 36192-12569=23624) the level is greater than equal to 3 .

p=52/36192
p

```
0.0014367816091954023
```

Here $p = 0.0014367816091954023$ which is the probability that the AQI level will be less than 3 in an hour of a day.


```
# to find the number of hours in a given date where the level is less than 3
pivot = aq_data.pivot_table(index =['Date'],
                             values =['PM2.5=1?'],
                             aggfunc = 'sum')

print(pivot)
```

	PM2.5=1?
Date	
2017-11-10	0
2017-11-11	0
2017-11-12	0
2017-11-13	0
2017-11-14	0
...	...
2022-5-9	0
2022-6-1	0
2022-6-2	0
2022-6-3	0
2022-6-4	0

[1616 rows x 1 columns]

pivot = pivot.reset_index().rename_axis(None, axis=1) #convert pivot to a dataframe
pivot

	Date	PM2.5=1?	
0	2017-11-10	0	
1	2017-11-11	0	
2	2017-11-12	0	
3	2017-11-13	0	

So, there are 1616 dates

```
pivot['PM2.5=1?'].sum() # to check

52

1616 0000 0 0
unique_AQI_level = pivot['PM2.5=1?'].unique()
```

```
X=list(unique_AQI_level)
X.sort()
print(X)

[0, 1, 2, 3, 5, 6, 8, 10]
```


```
seen=[]
data_dict={}
for i in pivot.index:
    for j in X:
        if pivot['PM2.5=1?'][i]==j:
            if pivot['PM2.5=1?'][i] in seen:
                data_dict[j]+=1
                break
            else:
                data_dict[j]=1
                seen.append(j)
                break
print(data_dict)

{0: 1601, 2: 6, 1: 2, 6: 1, 3: 3, 10: 1, 5: 1, 8: 1}
```

```
sorted(data_dict.items()) #used to sort

[(0, 1601), (1, 2), (2, 6), (3, 3), (5, 1), (6, 1), (8, 1), (10, 1)]
```

```
Model_data = pd.DataFrame(sorted(data_dict.items()), columns=["X", "Freq."])
Model_data
```

	X	Freq.	
0	0	1601	
1	1	2	
2	2	6	
3	3	3	
4	5	1	
5	6	1	
6	8	1	
7	10	1	

X is the random variable and Freq. denotes the number of times X takes a particular value x among the 1616 days.

```
probability=[]
for i in Model_data.index:
    cal=float(Model_data['Freq.'][i])/1616
    probability.append(cal)
print(probability)
```

[0.9907178217821783, 0.0012376237623762376, 0.0037128712871287127, 0.0018564356435643563, 0.0006188118811881188, 0.000618811881

```
Model_data.insert(2,"P(X=x)",probability,True)
Model_data
```

	X	Freq.	P(X=x)	
0	0	1601	0.990718	
1	1	2	0.001238	
2	2	6	0.003713	
3	3	3	0.001856	
4	5	1	0.000619	
5	6	1	0.000619	
6	8	1	0.000619	
7	10	1	0.000619	

```
import numpy as np
import matplotlib.pyplot as plt
```

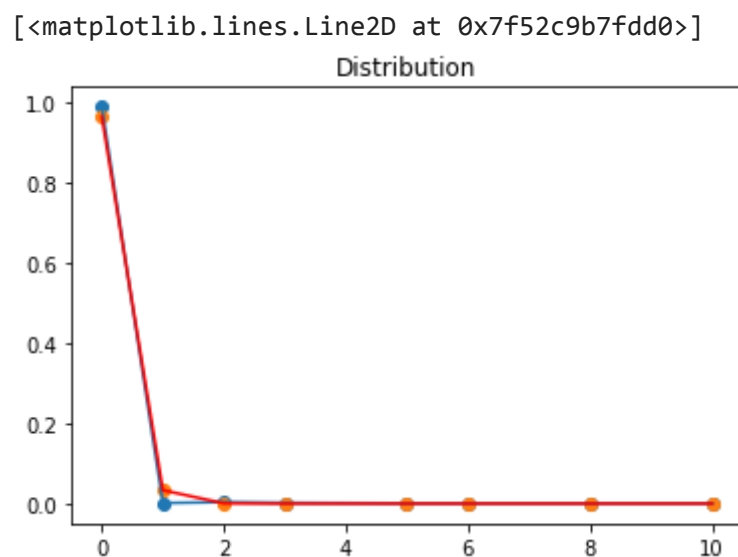
```
X_axis=X
Y_axis= probability
```

```
from scipy.stats import binom
dist = [binom.pmf(r, 24, p) for r in X_axis] # binomial(24,p)
```

```
# set the title of a plot
plt.title("Distribution")
```

```
# plot scatter plot with x and y data
plt.scatter(X_axis , Y_axis)
plt.scatter(X_axis , dist)
```

```
# plot with x and y data
plt.plot(X_axis, Y_axis)
plt.plot(X_axis, dist,color="red")
```



```
Model_data.insert(3,"P(X=x)~bin",dist,True)
Model_data
```

	X	Freq.	P(X=x)	P(X=x)~bin
0	0	1601	0.990718	9.660810e-01
1	1	2	0.001238	3.336107e-02
3	3	3	0.001856	5.824648e-06
4	5	1	0.000619	2.532324e-10
5	6	1	0.000619	1.153817e-12
6	8	1	0.000619	1.305271e-17
7	10	1	0.000619	7.206092e-23

Model_data.to_excel('Model_Data_1.xlsx', sheet_name = 'Sheet1')