Matplotlib

Data sorting using Matplotlib

In [4]:

```
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

Stats of all coloumns

Visualisation of dataset

In [5]:

```
report = pd.read_csv('district.csv')
report
#showing stats of all coloumns
```

Out[5]:

	district	active cases	confirmed cases	deceased	recovered
0	Ahmadnagar	17	42	2	23
1	Yavatmal	69	79	0	10
2	Washim	1	2	0	1
3	Solapur	93	99	6	0
4	Sindhudurg	1	2	0	1
5	Satara	21	32	2	9
6	Sangli	3	29	1	25
7	Ratnagiri	2	8	1	5
8	Raigarh	44	71	3	24
9	Parbhani	1	2	0	1
10	Palghar	119	169	4	46
11	Osmanabad	0	3	0	3
12	Nashik	179	197	12	6
13	Nandurbar	10	11	1	0
14	Nanded	3	3	0	0
15	Nagpur	100	139	2	37
16	Latur	3	12	1	8
17	Kolhapur	10	14	0	4
18	Buldana	3	21	1	17
19	Jalgaon	30	40	9	1
20	Hingoli	14	15	0	1
21	Gondiya	0	1	0	1
22	Dhule	22	25	3	0
23	Chandrapur	0	2	0	2
24	Buldana	3	21	1	17
25	Bid	0	1	0	1
26	Bhandara	1	1	0	0
27	Aurangabad	102	131	7	22
28	Amravati	17	28	7	4
29	Akola	30	39	1	8
30	Ahmadnagar	17	42	2	23
31	Mumbai	5679	7061	290	1092
32	Thane	755	943	16	172
33	Pune	912	1248	88	248

In [6]:

```
report.describe
#describing the data
```

Out[6]:

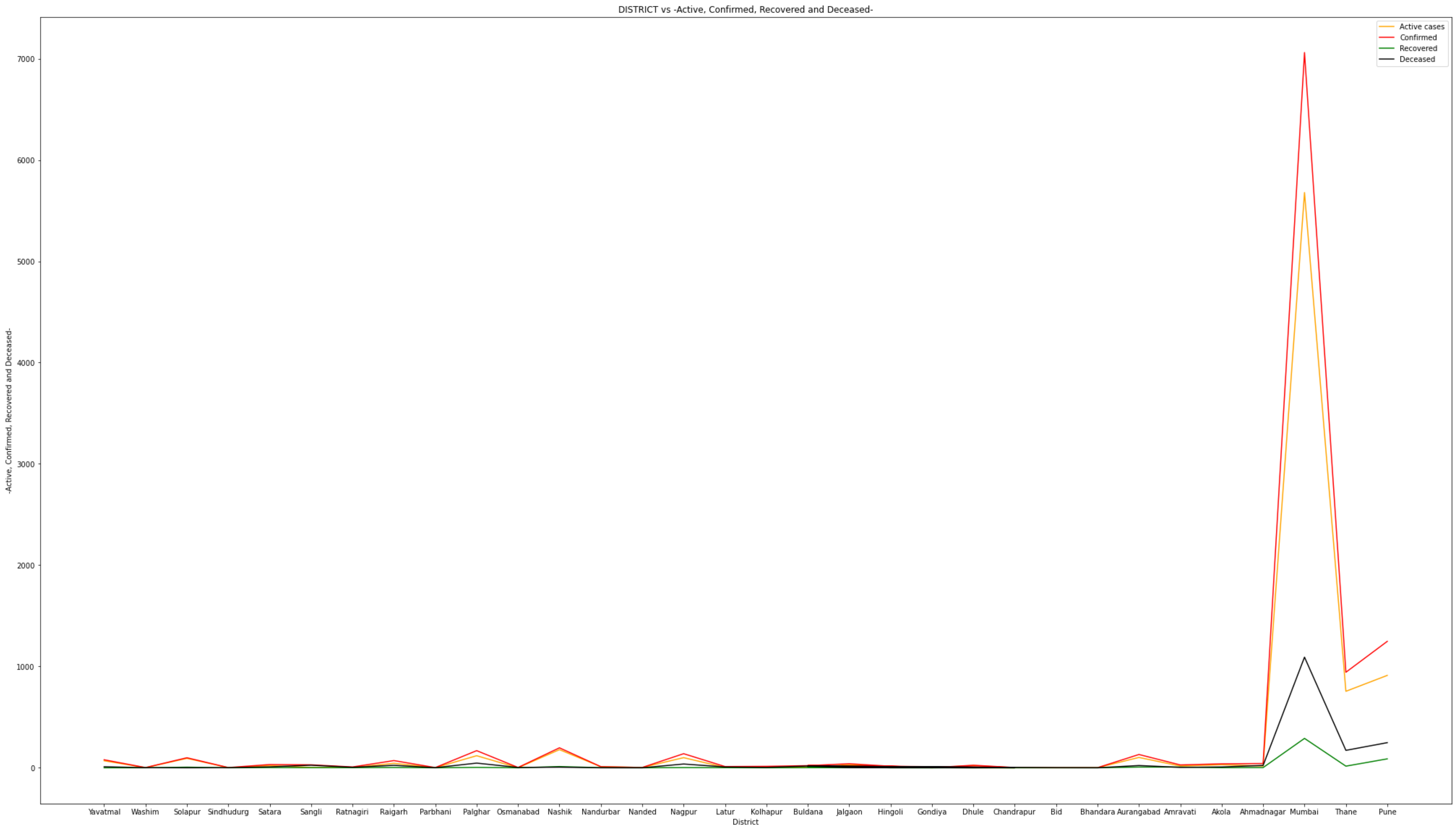
```
<bound method NDFrame.describe of       district  active cases  confirmed cases  deceased  recovered
0   Ahmadnagar            17                42         2         23
1     Yavatmal            69                79         0         10
2       Washim             1                 2         0          1
3      Solapur            93                99         6          0
4   Sindhudurg             1                 2         0          1
5       Satara            21                32         2          9
6       Sangli             3                29         1         25
7    Ratnagiri             2                 8         1          5
8      Raigarh            44                71         3         24
9     Parbhani             1                 2         0          1
10     Palghar           119               169         4         46
11   Osmanabad             0                 3         0          3
12      Nashik           179               197        12          6
13   Nandurbar            10                11         1          0
14      Nanded             3                 3         0          0
15      Nagpur           100               139         2         37
16       Latur             3                12         1          8
17    Kolhapur            10                14         0          4
18     Buldana             3                21         1         17
19     Jalgaon            30                40         9          1
20     Hingoli            14                15         0          1
21     Gondiya             0                 1         0          1
22       Dhule            22                25         3          0
23  Chandrapur             0                 2         0          2
24     Buldana             3                21         1         17
25         Bid             0                 1         0          1
26    Bhandara             1                 1         0          0
27  Aurangabad           102               131         7         22
28    Amravati            17                28         7          4
29       Akola            30                39         1          8
30  Ahmadnagar            17                42         2         23
31      Mumbai          5679              7061       290       1092
32       Thane           755               943        16        172
33        Pune           912              1248        88        248>
```

Line plot

Comparative visualization of active, confirmed, deceased and recovered cases districtwise

In [10]:

```
Y = report.iloc[1:,1].values
Z = report.iloc[1:,2].values
A = report.iloc[1:,3].values
B = report.iloc[1:,4].values
X = report.iloc[1:,0]
#assigning each parameter to an alphabate
#(X=district,Y=active cases, Z=confirmed cases, A=deceased, B=recovered)

plt.figure(figsize=(35,20))
#fixing the figure size so that graph can be properly viewed


plt.plot(X, Y, label="Active cases", color ="orange")
# District vs Active cases, indicating with orange color


plt.plot(X, Z, label="Confirmed", color = "red")
# District vs Confirmed cases, indicating with red color


plt.plot(X, A, label="Recovered " , color ="green")
# District vs Recovered cases, indicating with green color

plt.plot(X, B, label="Deceased", color ="black")
# District vs Deceased patients, indicating with black color

plt.xlabel('District')
#labeling x axis as DISTRICT

plt.ylabel('-Active, Confirmed, Recovered and Deceased-')
#labeling y axis

plt.title('DISTRICT vs -Active, Confirmed, Recovered and Deceased-')
#assigning title

plt.legend()
#legend visulization

plt.show()
#showing the plot
```



Bar diagram

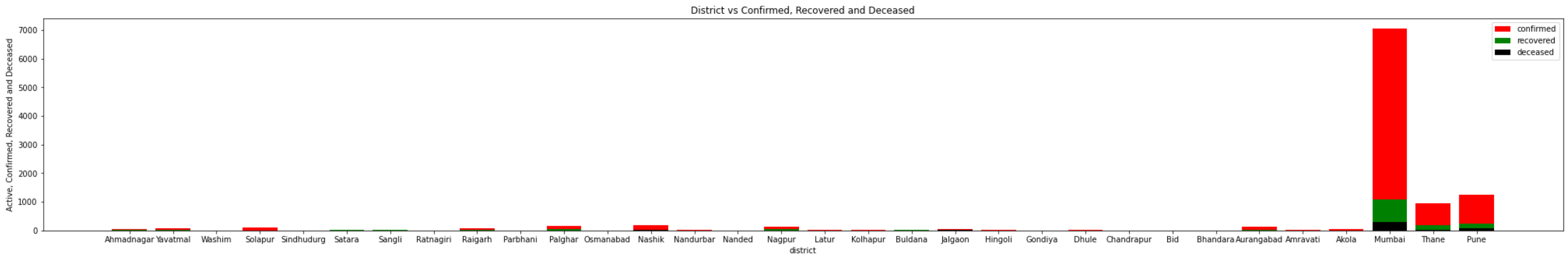
Bar diagram of confirmed, recovered and deceased cases districtwise

In [18]:

```
report = pd.read_csv('district.csv')
#datacalling

x=report['district']
y=report['confirmed cases']
h=report['recovered']
j=report['deceased']
#assigning each parameter to an alphabate

plt.figure(figsize=(35,5))
#assigning graph size

plt.bar(x,y,color=['red'],label="confirmed")
#confirmed cases districtwise indicating with red color

plt.bar(x,h,color=['green'],label="recovered")
#recovered cases districtwise indicating with green color

plt.bar(x,j,color=['black'],label="deceased")
#deceased cases districtwise indicating with black color

plt.xlabel("district")
#labeling x-axis

plt.ylabel("Active, Confirmed, Recovered and Deceased")
#labeling y-axis

plt.title('District vs Confirmed, Recovered and Deceased')
#giving title to graph

plt.legend()
#visulization of legend
```

Out[18]:

```
<matplotlib.legend.Legend at 0x166390b5970>
```



Sorting and plotting

sorting the data according to top 5 active cases districtwise and then depicting comparision of active & recovered cases in those districts

In [15]:

```
report = pd.read_csv('district.csv')
report.groupby(by="district")["active cases"].nlargest(5)
#sorting of districtwise active cases, picking highest 5 active cases
```

Out[15]:

```
district
Ahmadnagar  0       17
            30      17
Akola       29      30
Amravati    28      17
Aurangabad  27     102
Bhandara    26       1
Bid         25       0
Buldana     18       3
            24       3
Chandrapur  23       0
Dhule       22      22
Gondiya     21       0
Hingoli     20      14
Jalgaon     19      30
Kolhapur    17      10
Latur       16       3
Mumbai      31    5679
Nagpur      15     100
Nanded      14       3
Nandurbar   13      10
Nashik      12     179
Osmanabad   11       0
Palghar     10     119
Parbhani    9        1
Pune        33     912
Raigarh     8       44
Ratnagiri   7        2
Sangli      6        3
Satara      5       21
Sindhudurg  4        1
Solapur     3       93
Thane       32     755
Washim      2        1
Yavatmal    1       69
Name: active cases, dtype: int64
```

In [16]:

```
report.groupby(by="active cases")["recovered"].nlargest(5)
#sorting of recovered cases from active cases
```

Out[16]:

```
active cases
0             11       3
              23       2
              21       1
              25       1
1             2        1
              4        1
              9        1
              26       0
2             7        5
3             6       25
              18      17
              24      17
              16       8
              14       0
10            17       4
              13       0
14            20       1
17            0       23
              30      23
              28       4
21            5        9
22            22       0
30            29       8
              19       1
44            8       24
69            1       10
93            3        0
100           15      37
102           27      22
119           10      46
179           12       6
755           32     172
912           33     248
5679          31    1092
Name: recovered, dtype: int64
```

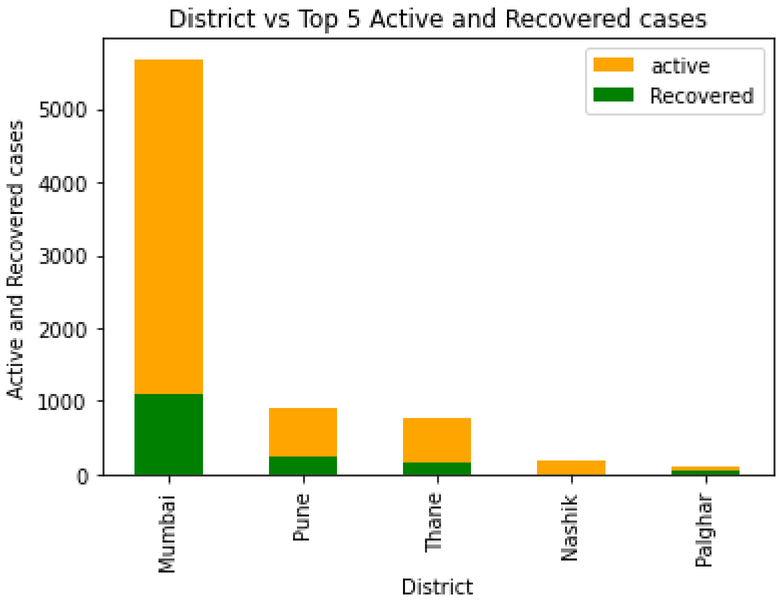
In [20]:

```
report = pd.DataFrame({'District': ['Mumbai', 'Pune', 'Thane', 'Nashik', 'Palghar'],'active':[5679,912,755,179,119],
                       'Recovered':[1092,248,172,6,46]})
#assigning values of top 5 districts with active cases

ax = report.plot(x="District",y="active",kind="bar", color="orange")
#ploting active cases destrictwise

report.plot(x="District", y="Recovered", kind="bar", ax=ax, color="green")
#ploting recovered cases destrictwise

plt.ylabel("Active and Recovered cases")
```

```
#labeling y-axis

plt.title('District vs Top 5 Active and Recovered cases')
#giving title to graph
```

Out[20]:

```
Text(0.5, 1.0, 'District vs Top 5 Active and Recovered cases')
```



In [ ]: