Data Analysis and Visualisation Practical - 6

Shad Jamil CSC/21/45

In [1]:

```
import numpy as np
import pandas as pd

data=pd.read_csv("weatherHistory.csv")
fram=data.head(15)
frame=fram.drop(["Formatted Date","Summary",'Loud Cover','Pressure (millibars)', 'Dail
x=pd.date_range('01/08/23',periods=15);
frame.insert(value=x,column="Date",loc=0)
frame
```

Out[1]:

	Date	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)
0	2023-01-08	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8263
1	2023-01-09	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8263
2	2023-01-10	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9569
3	2023-01-11	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8263
4	2023-01-12	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8263
5	2023-01-13	rain	9.222222	7.111111	0.85	13.9587	258.0	14.9569
6	2023-01-14	rain	7.733333	5.522222	0.95	12.3648	259.0	9.9820
7	2023-01-15	rain	8.772222	6.527778	0.89	14.1519	260.0	9.9820
8	2023-01-16	rain	10.822222	10.822222	0.82	11.3183	259.0	9.9820
9	2023-01-17	rain	13.772222	13.772222	0.72	12.5258	279.0	9.9820
10	2023-01-18	rain	16.016667	16.016667	0.67	17.5651	290.0	11.2056
11	2023-01-19	rain	17.144444	17.144444	0.54	19.7869	316.0	11.4471
12	2023-01-20	rain	17.800000	17.800000	0.55	21.9443	281.0	11.2700
13	2023-01-21	rain	17.333333	17.333333	0.51	20.6885	289.0	11.2700
14	2023-01-22	rain	18.877778	18.877778	0.47	15.3755	262.0	11.4471

a) Compute the mean of a series grouped by another series :

In [2]:

```
frame.groupby("Humidity")["Visibility (km)"].mean(numeric_only=True)
```

Out[2]:

```
Humidity
0.47    11.4471
0.51    11.2700
0.54    11.4471
0.55    11.2700
0.67    11.2056
0.72     9.9820
0.82     9.9820
0.83    15.8263
0.85    14.9569
0.86    15.8263
0.89    13.5884
0.95     9.9820
Name: Visibility (km), dtype: float64
```

b)Fill an intermittent time series to replace all missing dates with values of previous non-missing date.

In [3]:

```
value_to_replace=pd.date_range("01/10/2023",periods=5)
frame.replace(value_to_replace,np.nan,inplace=True)
frame
```

Out[3]:

	Date	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)
0	2023-01-08	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8263
1	2023-01-09	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8263
2	NaT	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9569
3	NaT	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8263
4	NaT	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8263
5	NaT	rain	9.222222	7.111111	0.85	13.9587	258.0	14.9569
6	NaT	rain	7.733333	5.522222	0.95	12.3648	259.0	9.9820
7	2023-01-15	rain	8.772222	6.527778	0.89	14.1519	260.0	9.9820
8	2023-01-16	rain	10.822222	10.822222	0.82	11.3183	259.0	9.9820
9	2023-01-17	rain	13.772222	13.772222	0.72	12.5258	279.0	9.9820
10	2023-01-18	rain	16.016667	16.016667	0.67	17.5651	290.0	11.2056
11	2023-01-19	rain	17.144444	17.144444	0.54	19.7869	316.0	11.4471
12	2023-01-20	rain	17.800000	17.800000	0.55	21.9443	281.0	11.2700
13	2023-01-21	rain	17.333333	17.333333	0.51	20.6885	289.0	11.2700
14	2023-01-22	rain	18.877778	18.877778	0.47	15.3755	262.0	11.4471

In [4]:

```
frame["Date"].replace(np.nan,method='ffill',inplace=True)
frame
```

Out[4]:

	Date	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)
0	2023-01-08	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8263
1	2023-01-09	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8263
2	2023-01-09	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9569
3	2023-01-09	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8263
4	2023-01-09	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8263
5	2023-01-09	rain	9.222222	7.111111	0.85	13.9587	258.0	14.9569
6	2023-01-09	rain	7.733333	5.522222	0.95	12.3648	259.0	9.9820
7	2023-01-15	rain	8.772222	6.527778	0.89	14.1519	260.0	9.9820
8	2023-01-16	rain	10.822222	10.822222	0.82	11.3183	259.0	9.9820
9	2023-01-17	rain	13.772222	13.772222	0.72	12.5258	279.0	9.9820
10	2023-01-18	rain	16.016667	16.016667	0.67	17.5651	290.0	11.2056
11	2023-01-19	rain	17.144444	17.144444	0.54	19.7869	316.0	11.4471
12	2023-01-20	rain	17.800000	17.800000	0.55	21.9443	281.0	11.2700
13	2023-01-21	rain	17.333333	17.333333	0.51	20.6885	289.0	11.2700
14	2023-01-22	rain	18.877778	18.877778	0.47	15.3755	262.0	11.4471

c)Perform appropriate year-month string to dates conversion.

In [5]:

```python
import datetime as dt
frame["Date"]=pd.to_datetime(frame["Date"]).dt.strftime("%d-%m-%Y")
frame
```

Out[5]:

	Date	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)
0	08-01-2023	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8263
1	09-01-2023	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8263
2	09-01-2023	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9569
3	09-01-2023	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8263
4	09-01-2023	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8263
5	09-01-2023	rain	9.222222	7.111111	0.85	13.9587	258.0	14.9569
6	09-01-2023	rain	7.733333	5.522222	0.95	12.3648	259.0	9.9820
7	15-01-2023	rain	8.772222	6.527778	0.89	14.1519	260.0	9.9820
8	16-01-2023	rain	10.822222	10.822222	0.82	11.3183	259.0	9.9820
9	17-01-2023	rain	13.772222	13.772222	0.72	12.5258	279.0	9.9820
10	18-01-2023	rain	16.016667	16.016667	0.67	17.5651	290.0	11.2056
11	19-01-2023	rain	17.144444	17.144444	0.54	19.7869	316.0	11.4471
12	20-01-2023	rain	17.800000	17.800000	0.55	21.9443	281.0	11.2700
13	21-01-2023	rain	17.333333	17.333333	0.51	20.6885	289.0	11.2700
14	22-01-2023	rain	18.877778	18.877778	0.47	15.3755	262.0	11.4471

d)Split a dataset to group by two columns and then sort the aggregated results within the groups.

In [6]:
```
grp=frame.groupby(["Humidity","Visibility (km)"])
grp.agg({"Wind Speed (km/h)":'mean'}).sort_values(by="Wind Speed (km/h)")
```

Out[6]:

		Wind Speed (km/h)
Humidity	**Visibility (km)**	
0.89	**14.9569**	3.9284
0.82	**9.9820**	11.3183
0.95	**9.9820**	12.3648
0.72	**9.9820**	12.5258
0.83	**15.8263**	12.5741
0.85	**14.9569**	13.9587
0.89	**15.8263**	14.1197
	9.9820	14.1519
0.86	**15.8263**	14.2646
0.47	**11.4471**	15.3755
0.67	**11.2056**	17.5651
0.54	**11.4471**	19.7869
0.51	**11.2700**	20.6885
0.55	**11.2700**	21.9443

e)Split a given dataframe into groups with bin counts.

In [7]:
```
grp2=frame.groupby(['Visibility (km)', pd.cut(frame['Temperature (C)'],4)])
grp2.size().unstack()
```

Out[7]:

Temperature (C)	(7.722, 10.519]	(10.519, 13.306]	(13.306, 16.092]	(16.092, 18.878]
Visibility (km)				
9.9820	2	1	1	0
11.2056	0	0	1	0
11.2700	0	0	0	2
11.4471	0	0	0	2
14.9569	2	0	0	0
15.8263	4	0	0	0

In [ ]:
```
```