data-science-lab-2-ui22cs03

August 12, 2024

[153]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[154]:
```
#import the data in excel in pandas
data = pd.read_excel('UI22CS03_DS_LAB_2_DATA .xlsx')
```

[155]:
```
data.head()
```

[155]:

	Year	Departure\n(millions)	Accidents	Fatalities
0	1985	6.1	4	197
1	1986	6.4	2	5
2	1987	6.6	4	231
3	1988	6.7	3	285
4	1989	6.6	11	278

[156]:
```
#Inforamtion about the dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 4 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Year                  22 non-null     int64
 1   Departure
(millions)  22 non-null     float64
 2   Accidents             22 non-null     int64
 3   Fatalities            22 non-null     int64
dtypes: float64(1), int64(3)
memory usage: 832.0 bytes
```

[157]:
```
print(data)
```

```
    Year  Departure\n(millions)  Accidents  Fatalities
0   1985                    6.1          4         197
1   1986                    6.4          2           5
```

2	1987	6.6	4	231
3	1988	6.7	3	285
4	1989	6.6	11	278
5	1990	7.8	6	39
6	1991	7.5	4	62
7	1992	7.5	4	33
8	1993	7.7	1	1
9	1994	7.8	4	239
10	1995	8.1	2	166
11	1996	7.9	3	342
12	1997	9.9	3	3
13	1998	10.5	1	1
14	1999	10.9	2	12
15	2000	11.1	2	89
16	2001	10.6	6	531
17	2002	10.3	0	0
18	2003	10.2	2	22
19	2004	109.0	1	13
20	2005	10.9	3	22
21	2006	11.2	2	50

#(a) Represent the number of yearly airline accidents in a frequency table.

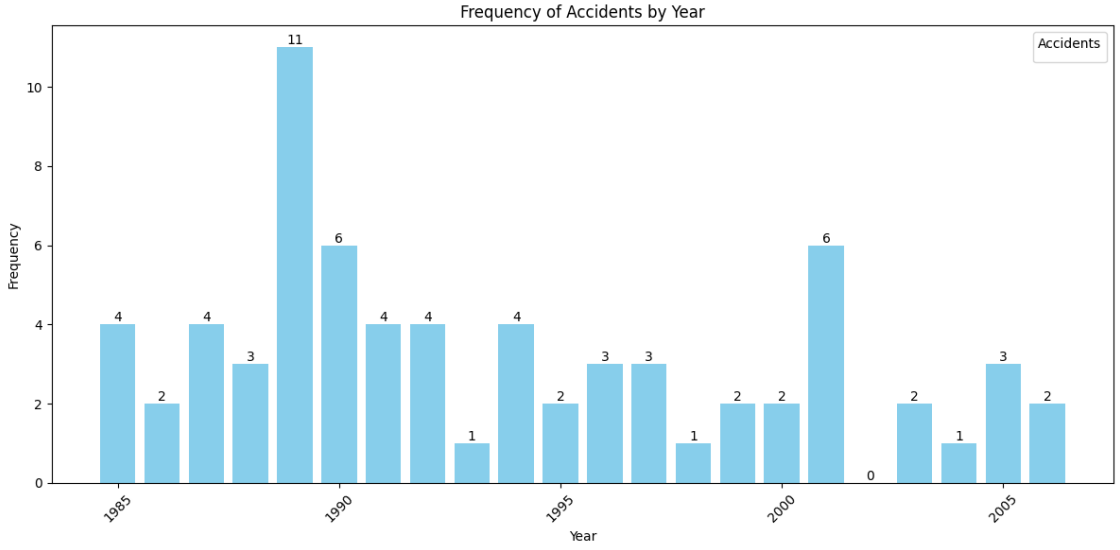
[158]:
```
freq_table = pd.DataFrame(data['Year'], data['Accidents'])
print(freq_table)
```

```
           Year
Accidents
4          1989
2          1987
4          1989
3          1988
11         1996
6          1991
4          1989
4          1989
1          1986
4          1989
2          1987
3          1988
3          1988
1          1986
2          1987
2          1987
6          1991
0          1985
2          1987
1          1986
```

```
3          1988
2          1987
```

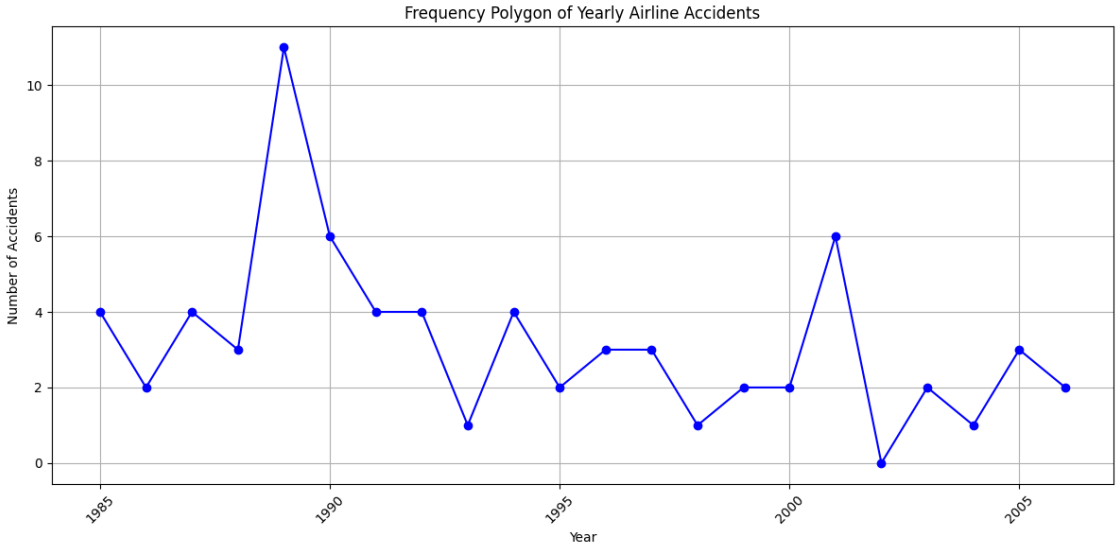
[177]:
```
#Lets plot that above as well
plt.figure(figsize=(12, 6))
bars = plt.bar(data['Year'], data['Accidents'], color='skyblue')
plt.title('Frequency of Accidents by Year')
plt.legend(title='Accidents')
plt.xlabel('Year')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.tight_layout()

# Add frequency labels above each bar
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval, int(yval), va='bottom',␣
 ↪ha='center')

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

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note
that artists whose label start with an underscore are ignored when legend() is
called with no argument.
```



1 (b) Give a frequency polygon graph of the number of yearly airline accidents.

[179]:
```
#Let's plot the graph
plt.figure(figsize=(12, 6))
plt.plot(data['Year'], data['Accidents'], marker='o', linestyle='-', color='b')
plt.xlabel('Year')
plt.ylabel('Number of Accidents')
plt.title('Frequency Polygon of Yearly Airline Accidents')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



#(c) Give a cumulative relative frequency plot of the number of yearly airline accidents.

[196]:
```
# Calculate the cumulative relative frequency
data_sorted = data.sort_values(by='Accidents')
print("Data sorted by Accident:- \n",data_sorted)

data_sorted['Cumulative Frequency of Accident'] = data_sorted['Accidents'].
 ↪cumsum()
print("\nCummulative Frequency :- \n",data_sorted['Cumulative Frequency of␣
 ↪Accident'])

data_sorted['Cumulative Relative Frequency of Accident'] =␣
 ↪data_sorted['Cumulative Frequency of Accident'] / data_sorted['Accidents'].
 ↪sum()
```

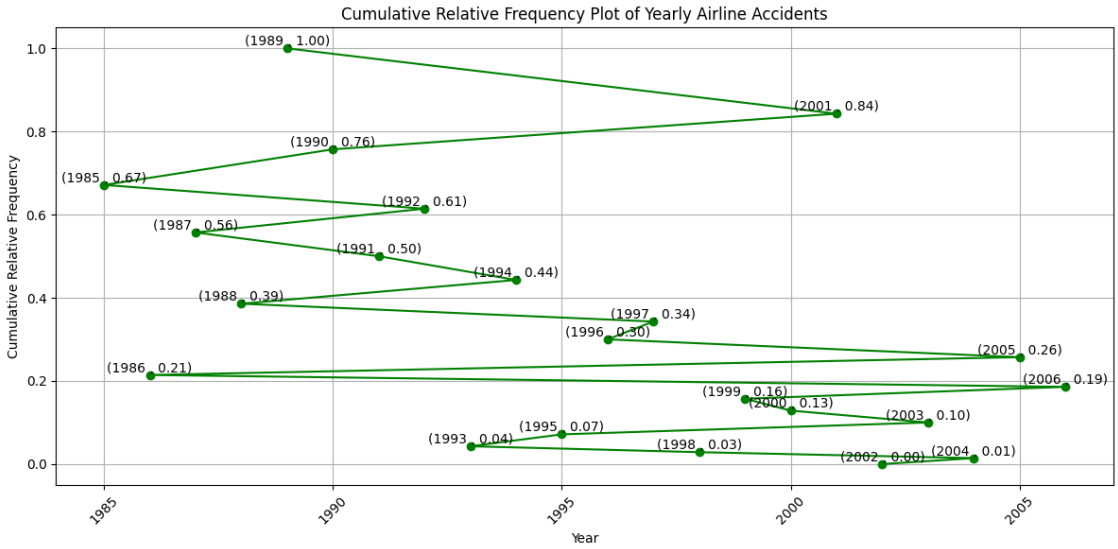
```
print("\nCumulative Relative Freq :- \n", data_sorted['Cumulative Relative␣
 ↪Frequency of Accident'])
```

```
Data sorted by Accident:-
     Year  Departure\n(millions)  Accidents  Fatalities
17   2002                   10.3          0           0
19   2004                  109.0          1          13
13   1998                   10.5          1           1
8    1993                    7.7          1           1
10   1995                    8.1          2         166
18   2003                   10.2          2          22
15   2000                   11.1          2          89
14   1999                   10.9          2          12
21   2006                   11.2          2          50
1    1986                    6.4          2           5
20   2005                   10.9          3          22
11   1996                    7.9          3         342
12   1997                    9.9          3           3
3    1988                    6.7          3         285
9    1994                    7.8          4         239
6    1991                    7.5          4          62
2    1987                    6.6          4         231
7    1992                    7.5          4          33
0    1985                    6.1          4         197
5    1990                    7.8          6          39
16   2001                   10.6          6         531
4    1989                    6.6         11         278

Cummulative Frequency :-
 17     0
19      1
13      2
8       3
10      5
18      7
15      9
14     11
21     13
1      15
20     18
11     21
12     24
3      27
9      31
6      35
2      39
7      43
```

```
0     47
5     53
16    59
4     70
Name: Cumulative Frequency of Accident, dtype: int64

Cumulative Relative Freq :-
 17    0.000000
19    0.014286
13    0.028571
8     0.042857
10    0.071429
18    0.100000
15    0.128571
14    0.157143
21    0.185714
1     0.214286
20    0.257143
11    0.300000
12    0.342857
3     0.385714
9     0.442857
6     0.500000
2     0.557143
7     0.614286
0     0.671429
5     0.757143
16    0.842857
4     1.000000
Name: Cumulative Relative Frequency of Accident, dtype: float64
```

[209]:
```
# Plot the cumulative relative frequency
plt.figure(figsize=(12, 6))
plt.plot(data_sorted['Year'], data_sorted['Cumulative Relative Frequency of␣
 ↪Accident'], marker='o', linestyle='-', color='g')
plt.xlabel('Year')
plt.ylabel('Cumulative Relative Frequency')
plt.title('Cumulative Relative Frequency Plot of Yearly Airline Accidents')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()

# Add cumulative relative frequency labels above each point
for x, y in zip(data_sorted['Year'], data_sorted['Cumulative Relative Frequency␣
 ↪of Accident']):
    plt.text(x, y, f'({x:.0f} , {y:.2f})', ha='center', va='bottom')
```

```
plt.show()
```



2 (d) Find the sample mean of the number of yearly airline accidents.

[162]:
```
# Calculate the sample mean
mean_accidents = statistics.mean(data['Accidents'])
print(f'Sample Mean of the Number of Yearly Airline Accidents:␣
  ↪{mean_accidents}')
```

```
Sample Mean of the Number of Yearly Airline Accidents: 3.1818181818181817
```

3 (e) Find the sample median of the number of yearly airline accidents.

[163]:
```
# Calculate the sample median
import statistics
median_accidents = statistics.median(data['Accidents'])
print(f'Sample Median of the Number of Yearly Airline Accidents:␣
  ↪{median_accidents}')
```

```
Sample Median of the Number of Yearly Airline Accidents: 3.0
```

#(f) Find the sample mode of the number of yearly airline accidents.

[164]:
```
# Calculate the sample mode
mode_accidents = statistics.mode(data['Accidents'])
```

```
print(f'Sample Mode of the Number of Yearly Airline Accidents:␣
  ↪{mode_accidents}')
```

```
Sample Mode of the Number of Yearly Airline Accidents: 2
```

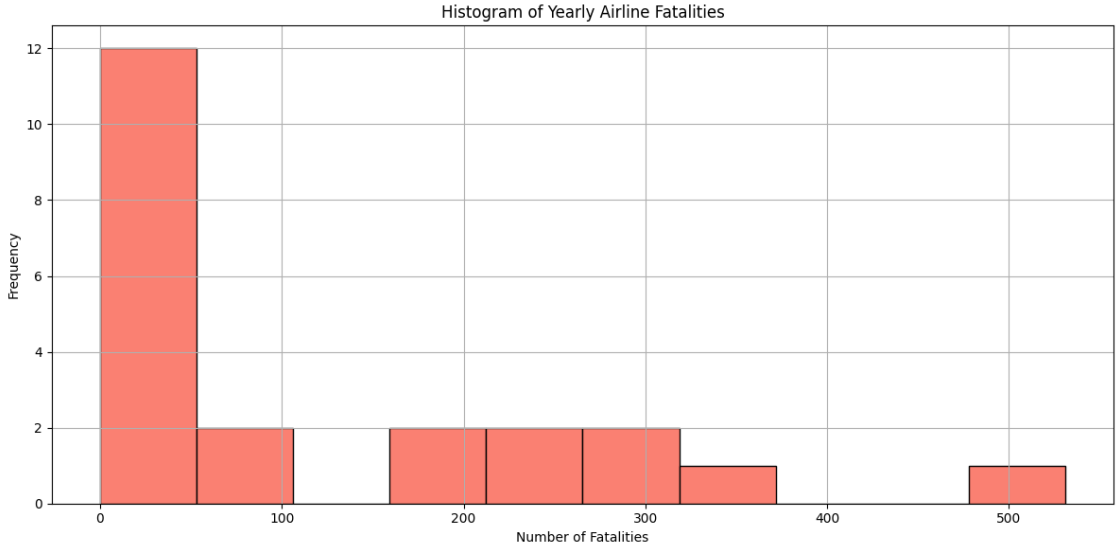
4 (g) Find the sample standard deviation of the number of yearly airline accidents.

[165]:
```
# Calculate the sample standard deviation
std_dev_accidents = statistics.stdev(data['Accidents'])
print(f'Sample Standard Deviation of the Number of Yearly Airline Accidents:␣
  ↪{std_dev_accidents}')
```

```
Sample Standard Deviation of the Number of Yearly Airline Accidents:
2.3224856068314814
```

5 (h) Represent the number of yearly airline fatalities in a histogram.

[166]:
```
# Plot a histogram of the number of fatalities per year
plt.figure(figsize=(12, 6))
plt.hist(data['Fatalities'], bins=10, color='salmon', edgecolor='black')
plt.xlabel('Number of Fatalities')
plt.ylabel('Frequency')
plt.title('Histogram of Yearly Airline Fatalities')
plt.grid(True)
plt.tight_layout()
plt.show()
```



6 (i) Represent the number of yearly airline fatalities in a stem and leaf plot.

[167]:
```
# Extracting the fatalities and sorting them
fatalities = sorted(data['Fatalities'])

# Function to create a stem-and-leaf plot
def stem_and_leaf(data, scale=10):
    stems = [x // scale for x in data]
    leaves = [x % scale for x in data]
    stem_leaf = {}
    for stem, leaf in zip(stems, leaves):
        if stem not in stem_leaf:
            stem_leaf[stem] = []
        stem_leaf[stem].append(leaf)

    # Print the stem-and-leaf plot
    for stem in sorted(stem_leaf.keys()):
        leaf_str = ' '.join(str(leaf) for leaf in sorted(stem_leaf[stem]))
        print(f"{stem} | {leaf_str}")

print("Stem-and-Leaf Plot of Yearly Airline Fatalities:")
stem_and_leaf(fatalities)
```

```
Stem-and-Leaf Plot of Yearly Airline Fatalities:
0 | 0 1 1 3 5
1 | 2 3
2 | 2 2
3 | 3 9
5 | 0
6 | 2
8 | 9
16 | 6
19 | 7
23 | 1 9
27 | 8
28 | 5
34 | 2
53 | 1
```

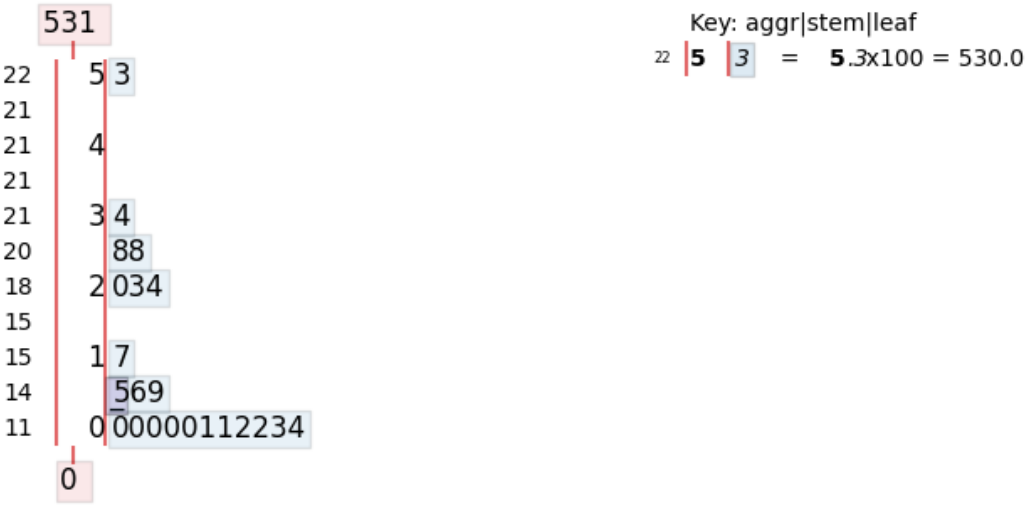
[168]:
```
# (i) Represent the number of yearly airline fatalities in a stem and leaf plot␣
  ↪using steamgraph libray
!pip install stemgraphic==0.9.1
```

```
import stemgraphic

# Extract the fatalities data into a list
fatalities = data['Fatalities'].tolist()

# Create the stem and leaf plot
stemgraphic.stem_graphic(fatalities)
```

```
Requirement already satisfied: stemgraphic==0.9.1 in
/usr/local/lib/python3.10/dist-packages (0.9.1)
Requirement already satisfied: docopt in /usr/local/lib/python3.10/dist-packages
(from stemgraphic==0.9.1) (0.6.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-
packages (from stemgraphic==0.9.1) (3.7.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(from stemgraphic==0.9.1) (2.1.4)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-
packages (from stemgraphic==0.9.1) (0.13.1)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->stemgraphic==0.9.1)
(1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib->stemgraphic==0.9.1) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->stemgraphic==0.9.1)
(4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->stemgraphic==0.9.1)
(1.4.5)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-
packages (from matplotlib->stemgraphic==0.9.1) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->stemgraphic==0.9.1)
(24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-
packages (from matplotlib->stemgraphic==0.9.1) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->stemgraphic==0.9.1)
(3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->stemgraphic==0.9.1)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->stemgraphic==0.9.1) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->stemgraphic==0.9.1) (2024.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
```

```
packages (from python-dateutil>=2.7->matplotlib->stemgraphic==0.9.1) (1.16.0)
```

[168]:
```
(<Figure size 750x350 with 1 Axes>, <Axes: >)
```



7 (j) Find the sample mean of the number of yearly airline fatalities.

[169]:
```
# Calculate the sample mean of fatalities
mean_fatalities = statistics.mean(data['Fatalities'])
print(f'Sample Mean of the Number of Yearly Airline Fatalities:␣
 ↪{mean_fatalities}')
```

```
Sample Mean of the Number of Yearly Airline Fatalities: 119.13636363636364
```

8 (k) Find the sample median of the number of yearly airline fatalities.

[170]:
```
# Calculate the sample median of fatalities
median_fatalities = statistics.median(data['Fatalities'])
print(f'Sample Median of the Number of Yearly Airline Fatalities:␣
 ↪{median_fatalities}')
```

```
Sample Median of the Number of Yearly Airline Fatalities: 44.5
```

9 (k) Find the sample standard deviation of the number of yearly airline fatalities

[171]:
```
# Calculate the sample median of fatalities
stdev_fatalities = statistics.stdev(data['Fatalities'])
print(f'Sample Median of the Number of Yearly Airline Fatalities:␣
 ↪{stdev_fatalities}')
```

```
Sample Median of the Number of Yearly Airline Fatalities: 144.78499509684974
```

10 CODE BY ADITYA KUMAR (UI22CS03)