Course Project Report

STA 2101: Statistics & Probability

Project Title : Dhaka AQ

Student Name : Md. Shahriar Alam
Student ID : 242014180
University of Liberal Arts Bangladesh (ULAB)

December 19, 2025

Abstract

This project studies the relationship between weather and air quality in Dhaka using the "Dhaka Daily Air Quality and Weather" dataset. It applies key ideas from statistics and probability, such as central tendency, dispersion, frequency distributions, probability, conditional probability, Bayes' rule, and simple linear regression. Data visualization and manual calculations of regression and correlation helped reveal patterns and relationships. The results show that temperature, humidity, and wind speed have clear patterns, with temperature and humidity having a moderate linear relationship. This project shows how statistical and probability methods can be used with real-world environmental data to find insights and make predictions.

GitHub Repository URL : `https://github.com/Brother258/STA2101`

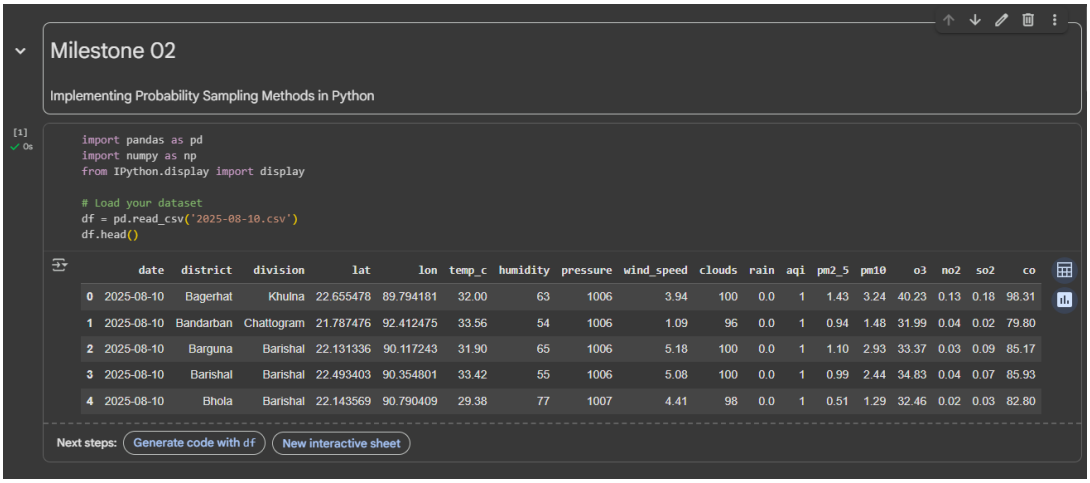
Contents

1 Milestone 1: Dataset Selection

- **Dataset Name :** Dhaka Daily Air Quality and Weather Dataset
- **Dataset URL :** `https://www.kaggle.com/datasets/albab12/dhaka-daily-air-quality-and-weather-dataset`
- **Description :** This project uses the "Dhaka Daily Air Quality & Weather" dataset. The dataset provides daily records for Dhaka, Bangladesh. It contains two main types of information: air quality and weather. The air quality data includes the Air Quality Index (AQI). It also measures several pollutants. These pollutants include PM2.5, PM10, nitrogen dioxide, ozone, carbon monoxide, and sulfur dioxide. The weather data includes daily temperature. It also has information on humidity, barometric pressure, and wind speed.

  This dataset was chosen because it provides comprehensive variables for both air quality and weather. This makes it ideal for studying the relationship between these factors in Dhaka using Statistics & Probability concept.

2 Milestone 02: Probability Sampling Methods



Milestone 02

Implementing Probability Sampling Methods in Python

```
import pandas as pd
import numpy as np
from IPython.display import display

# Load your dataset
df = pd.read_csv('2025-08-10.csv')
df.head()
```

	date	district	division	lat	lon	temp_c	humidity	pressure	wind_speed	clouds	rain	aqi	pm2_5	pm10	o3	no2	so2	co
0	2025-08-10	Bagerhat	Khulna	22.655478	89.794181	32.00	63	1006	3.94	100	0.0	1	1.43	3.24	40.23	0.13	0.18	98.31
1	2025-08-10	Bandarban	Chattogram	21.787476	92.412475	33.56	54	1006	1.09	96	0.0	1	0.94	1.48	31.99	0.04	0.02	79.80
2	2025-08-10	Barguna	Barishal	22.131336	90.117243	31.90	65	1006	5.18	100	0.0	1	1.10	2.93	33.37	0.03	0.09	85.17
3	2025-08-10	Barishal	Barishal	22.493403	90.354801	33.42	55	1006	5.08	100	0.0	1	0.99	2.44	34.83	0.04	0.07	85.93
4	2025-08-10	Bhola	Barishal	22.143569	90.790409	29.38	77	1007	4.41	98	0.0	1	0.51	1.29	32.46	0.02	0.03	82.80

Next steps: Generate code with df | New interactive sheet

Figure 1: Overview

2.1 Part A — Setup

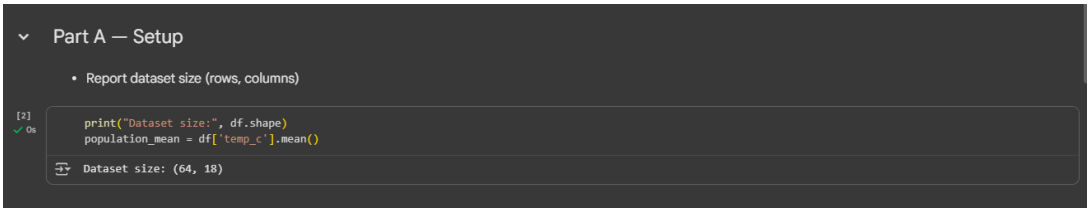


Figure 2: Setup

2.2 Part B — Simple Random Sampling

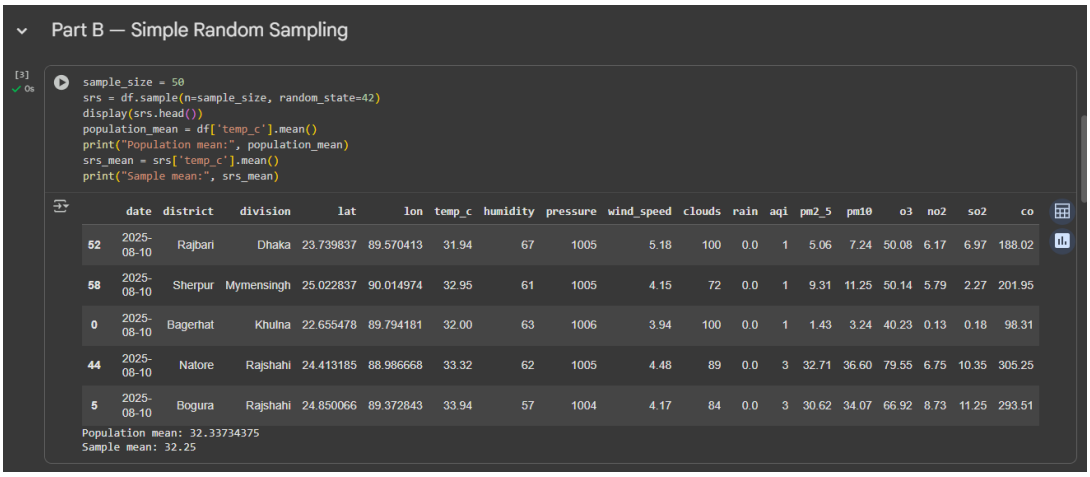


Figure 3: Simple Random Sampling

2.3 Part C — Systematic Sampling

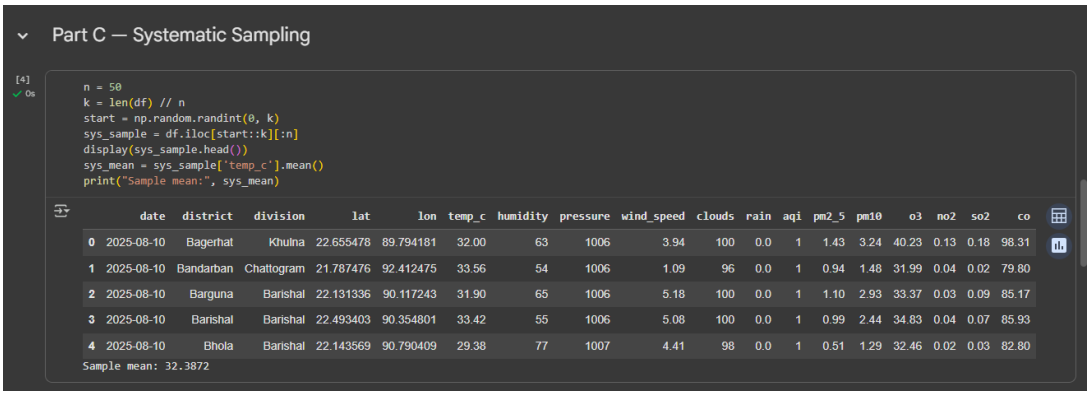


Figure 4: Systematic Sampling

2.4 Part D — Stratified Sampling

Part D — Stratified Sampling

```
[5]
strata_col = "division"  # your column
sample_size = 50

# proportional fraction for each group
frac = sample_size / len(df)

# stratified sample
stratified_sample = df.groupby(strata_col, group_keys=False).sample(frac=frac, random_state=42)

display(stratified_sample.head())
strat_mean = stratified_sample['temp_c'].mean()
print("Sample mean:", strat_mean)
```

	date	district	division	lat	lon	temp_c	humidity	pressure	wind_speed	clouds	rain	aqi	pm2_5	pm10	o3	no2	so2	co
2	2025-08-10	Barguna	Barishal	22.131336	90.117243	31.90	65	1006	5.18	100	0.0	1	1.10	2.93	33.37	0.03	0.09	85.17
3	2025-08-10	Barishal	Barishal	22.493403	90.354801	33.42	55	1006	5.08	100	0.0	1	0.99	2.44	34.83	0.04	0.07	85.93
51	2025-08-10	Pirojpur	Barishal	22.509560	90.007250	32.78	59	1006	5.06	100	0.0	1	1.17	2.82	35.72	0.05	0.09	87.35
4	2025-08-10	Bhola	Barishal	22.143569	90.790409	29.38	77	1007	4.41	98	0.0	1	0.51	1.29	32.46	0.02	0.03	82.80
50	2025-08-10	Patuakhali	Barishal	22.008424	90.382683	30.91	70	1006	4.92	99	0.0	1	0.90	2.38	33.06	0.03	0.07	84.38

```
Sample mean: 32.327600000000004
```

Figure 5: Stratified Sampling

2.5 Part E — Cluster Sampling

Part E — Cluster Sampling

```
[6]
df['cluster_id'] = df.index // (len(df)//10)  # 10 clusters
selected_clusters = np.random.choice(df['cluster_id'].unique(), size=2, replace=False)
cluster_sample = df[df['cluster_id'].isin(selected_clusters)]
print("Selected clusters:", selected_clusters)
display(cluster_sample.head())
cluster_mean = cluster_sample['temp_c'].mean()
print("Sample mean:", cluster_mean)
```

```
Selected clusters: [8 3]
```

	date	district	division	lat	lon	temp_c	humidity	pressure	wind_speed	clouds	rain	aqi	pm2_5	pm10	o3	no2	so2	co	cluster
18	2025-08-10	Gazipur	Dhaka	23.999756	90.417363	32.01	70	1005	4.12	75	0.0	1	1.49	2.31	41.18	1.48	1.02	118.18	
19	2025-08-10	Gopalganj	Dhaka	23.004994	89.830318	33.12	57	1005	4.06	100	0.0	1	1.31	2.42	40.07	1.02	1.38	106.03	
20	2025-08-10	Habiganj	Sylhet	24.374603	91.414027	31.58	68	1006	3.86	100	0.0	1	1.34	2.19	33.04	1.38	1.07	112.93	
21	2025-08-10	Jamalpur	Mymensingh	24.925587	89.943668	32.75	62	1005	4.48	75	0.0	2	13.24	15.48	56.40	6.15	4.10	220.83	
22	2025-08-10	Jashore	Khulna	23.166526	89.209442	31.80	69	1006	4.47	100	0.0	1	2.63	4.01	47.81	2.23	3.14	136.88	

```
Sample mean: 32.5075
```

Figure 6: Cluster Sampling

2.6 Part F — Comparison & Reflection

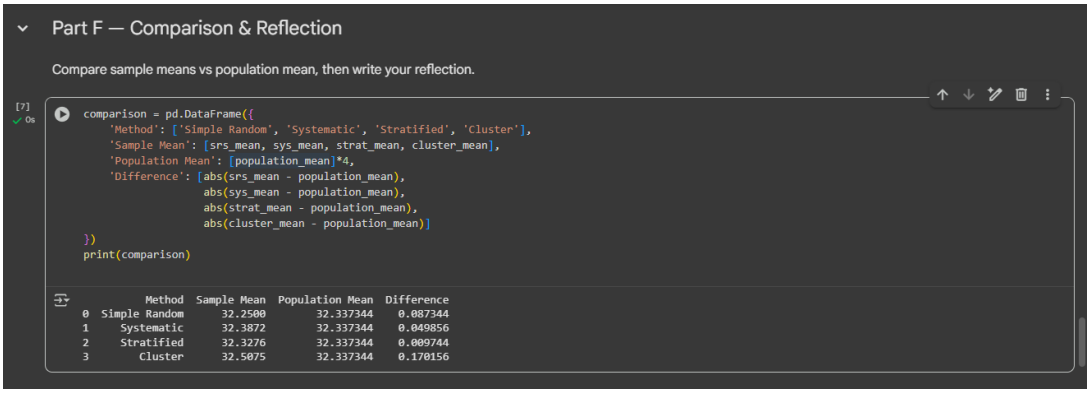


Figure 7: Comparison and Reflection

In this milestone, I used four probability sampling methods. The population mean was 32.337344. All methods gave sample means close to it but not the same. Stratified sampling gave the closest mean 32.3276. The difference was very small. This happened because it kept the same group ratio as the dataset. Simple random sampling gave 32.25 which is a bit lower than the population mean. Systematic sampling gave 32.3872 which is slightly higher. Cluster sampling gave 32.5075 which is the farthest from the population mean.

Simple random sampling was the easiest. I wrote only one line of code and got the result. It did not need any group or pattern. Systematic sampling was also easy. I just needed k and a start point. Stratified sampling was harder. I had to use a division column and take samples from each group by proportion. Cluster sampling was easy to code but tricky to pick clusters.

Each method works for different goals. Simple random sampling is good for small or mixed data. Systematic sampling is good when data has no clear order. Stratified sampling is best for datasets with clear groups. Cluster sampling is useful for large data that is grouped by place or type. From this, I saw stratified sampling gave the most accurate result. Simple random sampling was the easiest to use.

3 Milestone 3: Frequency Distribution and Graphical Representation

3.1 Part A - Introduction

The goal of this milestone is to analyze distribution of variables. I will create a frequency distribution table to summarize the data and then use suitable charts to visualize the distribution. This will help me understand the basic pattern and key characteristics of

the data more clearly. This notebook follows the teacher's instructions carefully. Each part from A to G is organized as a separate section and each column analysis is written as a runnable cell group. All outputs will be kept visible when the notebook is run in Colab.

3.2 Part B - Dataset

I used the cleaned dataset from previous milestones. Dataset used here: 2025-08-10.csv

```
Loaded: 2025-08-10.csv
Shape: (64, 18)
Columns: ['date', 'district', 'division', 'lat', 'lon', 'temp_c', 'humidity', 'pressure', 'wind_speed', 'cloud
```

	date	district	division	lat	lon	temp_c	humidity	pressure	wind_speed	clouds	rain	a
0	2025-08-10	Bagerhat	Khulna	22.655478	89.794181	32.00	63	1006	3.94	100	0.0	
1	2025-08-10	Bandarban	Chattogram	21.787476	92.412475	33.56	54	1006	1.09	96	0.0	
2	2025-08-10	Barguna	Barishal	22.131336	90.117243	31.90	65	1006	5.18	100	0.0	
3	2025-08-10	Barishal	Barishal	22.493403	90.354801	33.42	55	1006	5.08	100	0.0	
4	2025-08-10	Bhola	Barishal	22.143569	90.790409	29.38	77	1007	4.41	98	0.0	
5	2025-08-10	Bogura	Rajshahi	24.850066	89.372843	33.94	57	1004	4.17	84	0.0	
6	2025-08-10	Brahmanbaria	Chattogram	23.960600	91.119089	33.06	66	1006	4.12	75	0.0	
7	2025-08-10	Chandpur	Chattogram	23.224176	90.653100	32.58	61	1006	4.26	98	0.0	
8	2025-08-10	Chapai Nawabganj	Rajshahi	24.599887	88.285047	35.13	52	1004	4.03	76	0.0	
9	2025-08-10	Chattogram	Chattogram	22.333778	91.834435	31.93	66	1007	4.12	40	0.0	

Figure 8: Dataset Loaded

Loaded: 2025-08-10.csv dataset. Shape: (64, 18)

3.3 Part C (Task 01) - Frequency Distribution Table :

C. Task 1: Frequency Distribution Table

```
# C: Frequency table for numeric column: temp_c
col = "temp_c"
if col not in df.columns:
    print('Column not found:', col)
else:
    table = grouped_frequency_table(df[col], bins='sturges')
    print('Frequency distribution table for column:', col)
    display(table)
    table.to_csv(f"freq_table_{col}.csv", index=False)
    print('Saved: freq_table_{col}.csv')
```

Frequency distribution table for column: temp_c

	Class Interval	Lower	Upper	Midpoint	Frequency (f)	Relative Frequency (rf)	Cumulative Frequency (cf)	Relative Cumulative Frequency (rcf)
0	28.539 - 29.481	28.539	29.481	29.0100	3	0.046875	3	0.046875
1	29.481 - 30.423	29.481	30.423	29.9520	0	0.000000	3	0.046875
2	30.423 - 31.364	30.423	31.364	30.8935	7	0.109375	10	0.156250
3	31.364 - 32.306	31.364	32.306	31.8350	21	0.328125	31	0.484375
4	32.306 - 33.247	32.306	33.247	32.7765	18	0.281250	49	0.765625
5	33.247 - 34.189	33.247	34.189	33.7180	13	0.203125	62	0.968750
6	34.189 - 35.13	34.189	35.130	34.6595	2	0.031250	64	1.000000

Saved: freq_table_{col}.csv

Figure 9: Frequency Distribution Table for numeric column temp_c

A table with 7 rows showing Class Intervals, Lower/Upper bounds, Midpoints, Frequency (f), Relative Frequency (rf), Cumulative Frequency (cf), and Relative Cumulative Frequency (rcf). The total sample size (N) is 64.

```
# C: Frequency table for numeric column: humidity
col = "humidity"
if col not in df.columns:
    print('Column not found:', col)
else:
    table = grouped_frequency_table(df[col], bins='sturges')
    print('Frequency distribution table for column:', col)
    display(table)
    table.to_csv(f"freq_table_{col}.csv", index=False)
    print('Saved: freq_table_{col}.csv')
```

Frequency distribution table for column: humidity

	Class Interval	Lower	Upper	Midpoint	Frequency (f)	Relative Frequency (rf)	Cumulative Frequency (cf)	Relative Cumulative Frequency (rcf)
0	51.999 - 56.286	51.999	56.286	54.1425	8	0.125000	8	0.125000
1	56.286 - 60.571	56.286	60.571	58.4285	14	0.218750	22	0.343750
2	60.571 - 64.857	60.571	64.857	62.7140	14	0.218750	36	0.562500
3	64.857 - 69.143	64.857	69.143	67.0000	16	0.250000	52	0.812500
4	69.143 - 73.429	69.143	73.429	71.2860	9	0.140625	61	0.953125
5	73.429 - 77.714	73.429	77.714	75.5715	1	0.015625	62	0.968750
6	77.714 - 82	77.714	82.000	79.8570	2	0.031250	64	1.000000

Saved: freq_table_{col}.csv

Figure 10: Frequency Distribution Table for numeric column humidity

A frequency distribution table for humidity with 7 rows. The highest frequency (16) occurs in the interval 64.857 – 69.143.

```
# C: Frequency table for numeric column: wind_speed
col = "wind_speed"
if col not in df.columns:
    print('Column not found:', col)
else:
    table = grouped_frequency_table(df[col], bins='sturges')
    print('Frequency distribution table for column:', col)
    display(table)
    table.to_csv(f"freq_table_{col}.csv", index=False)
    print('Saved: freq_table_{col}.csv')
```

Frequency distribution table for column: wind_speed

	Class Interval	Lower	Upper	Midpoint	Frequency (f)	Relative Frequency (rf)	Cumulative Frequency (cf)	Relative Cumulative Frequency (rcf)
0	0.459 - 1.153	0.459	1.153	0.8060	2	0.031250	2	0.031250
1	1.153 - 1.846	1.153	1.846	1.4995	1	0.015625	3	0.046875
2	1.846 - 2.539	1.846	2.539	2.1925	4	0.062500	7	0.109375
3	2.539 - 3.231	2.539	3.231	2.8850	4	0.062500	11	0.171875
4	3.231 - 3.924	3.231	3.924	3.5775	10	0.156250	21	0.328125
5	3.924 - 4.617	3.924	4.617	4.2705	26	0.406250	47	0.734375
6	4.617 - 5.31	4.617	5.310	4.9635	17	0.265625	64	1.000000

Saved: freq_table_{col}.csv

Figure 11: Frequency Distribution Table for numeric column wind_speed

Uses Sturges rule for binning numeric wind speed data. A table with 7 rows. The data is heavily clustered in the higher intervals, with the highest frequency (26) in the 3.924 – 4.617 range.

```
# C: Frequency table for numeric column: rain
col = "rain"
if col not in df.columns:
    print('Column not found:', col)
else:
    table = grouped_frequency_table(df[col], bins='sturges')
    print('Frequency distribution table for column:', col)
    display(table)
    table.to_csv(f"freq_table_{col}.csv", index=False)
    print('Saved: freq_table_{col}.csv')
```

Frequency distribution table for column: rain

	Class Interval	Lower	Upper	Midpoint	Frequency (f)	Relative Frequency (rf)	Cumulative Frequency (cf)	Relative Cumulative Frequency (rcf)
0	-0.501 - 0.5	-0.501	0.5	-0.0005	64	1.0	64	1.0

Saved: freq_table_{col}.csv

Figure 12: Frequency Table for numeric column rain

The table shows only one row where all 64 data points fall into the interval -0.501 to 0.5. This suggests that the "rain" column likely contains only zeros or a single constant value.

```python
# C: Frequency table for categorical column: district
col = "district"
if col not in df.columns:
    print('Column not found:', col)
else:
    table = categorical_frequency_table(df[col])
    print('Categorical frequency table for column:', col)
    display(table)
    table.to_csv(f"cat_freq_table_{col}.csv", index=False)
    print('Saved: cat_freq_table_{col}.csv')
```

```
Categorical frequency table for column: district
```

	Category	Frequency (f)	Relative Frequency (rf)	Cumulative Frequency (cf)	Relative Cumulative Frequency (rcf)
0	Bagerhat	1	0.015625	1	0.015625
1	Bandarban	1	0.015625	2	0.031250
2	Barguna	1	0.015625	3	0.046875
3	Barishal	1	0.015625	4	0.062500
4	Bhola	1	0.015625	5	0.078125
...	...	...	...	...	...
59	Sirajganj	1	0.015625	60	0.937500
60	Sunamganj	1	0.015625	61	0.953125
61	Sylhet	1	0.015625	62	0.968750
62	Tangail	1	0.015625	63	0.984375

Figure 13: Frequency Table for categorical column district

Ttable listing various districts (Bagerhat, Bandarban, etc.) as categories. Each district shown has a frequency of 1, indicating a uniform distribution of unique values across the dataset.

```python
# C: Frequency table for categorical column: division
col = "division"
if col not in df.columns:
    print('Column not found:', col)
else:
    table = categorical_frequency_table(df[col])
    print('Categorical frequency table for column:', col)
    display(table)
    table.to_csv(f"cat_freq_table_{col}.csv", index=False)
    print('Saved: cat_freq_table_{col}.csv')
```

```
Categorical frequency table for column: division
```

	Category	Frequency (f)	Relative Frequency (rf)	Cumulative Frequency (cf)	Relative Cumulative Frequency (rcf)
0	Dhaka	13	0.203125	13	0.203125
1	Chattogram	11	0.171875	24	0.375000
2	Khulna	10	0.156250	34	0.531250
3	Rajshahi	8	0.125000	42	0.656250
4	Rangpur	8	0.125000	50	0.781250
5	Barishal	6	0.093750	56	0.875000
6	Sylhet	4	0.062500	60	0.937500
7	Mymensingh	4	0.062500	64	1.000000

```
Saved: cat_freq_table_{col}.csv
```

Figure 14: Frequency Table for categorical column division

Table showing 8 divisions (Dhaka, Chattogram, Khulna, etc.). Dhaka has the highest frequency (13), and the total count is 64.

3.4 Part D (Task 02) - Graphical Representation :

Plots for numeric column: temp_c :

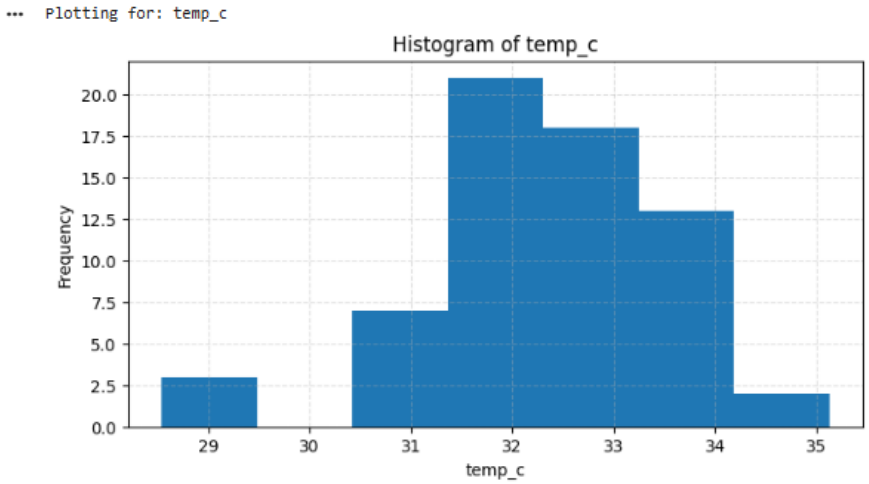


Figure 15: Histogram of temp_c

This chart displays the frequency of temperature values grouped into specific bins. The highest bar indicates that the most common temperature range occurs between 31.5 and 32.5, with a frequency of 21. Similar to the frequency polygon, there is an empty bin (frequency 0) around the 30-degree mark. The distribution is slightly skewed to the right, with higher temperature ranges (33 to 35) having frequencies of 18, 13, and 2 respectively.

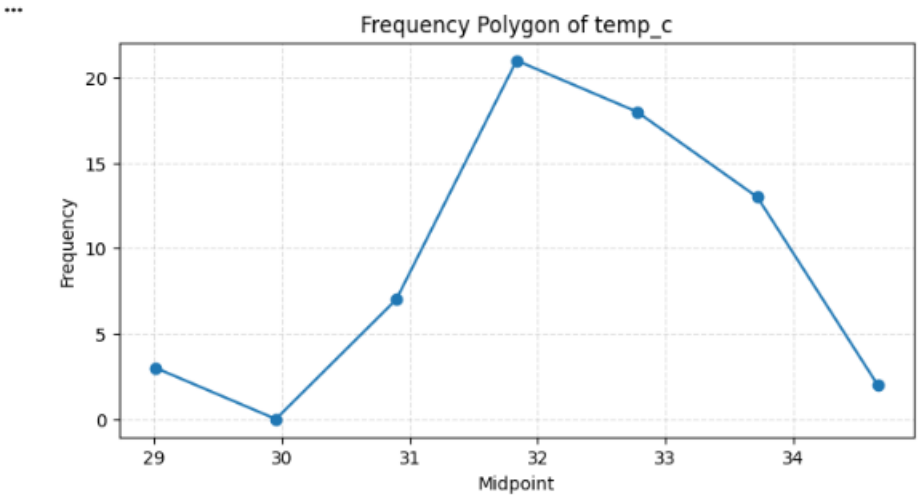


Figure 16: Frequency Polygon of temp_c

This line graph uses midpoints to show how the frequency of temperature observations is distributed. The graph starts with a frequency of 3 at a midpoint of 29. There is a notable drop to 0 at the midpoint of 30. The frequency peaks sharply at 21 near the midpoint of 31.8. From there, it steadily decreases to 18, 13, and finally hits a low of 2 at the final midpoint of approximately 34.7.

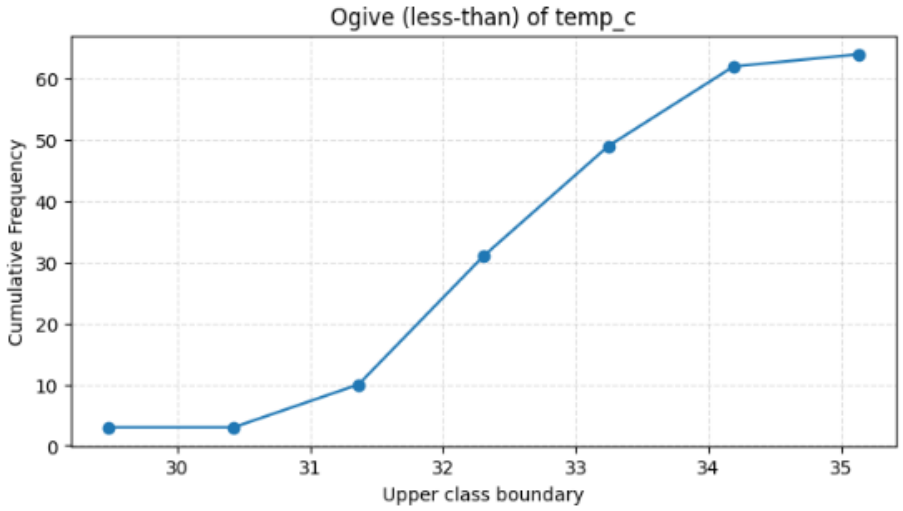


Figure 17: Ogive (less-than of temp_c)

This graph displays the cumulative frequency distribution for temperature data. The y-axis shows Cumulative Frequency (0 to 60+), and the x-axis represents the Upper class boundary for temperature in Celsius. The curve remains flat at a frequency of approximately 3 until the 30.5 boundary, after which it rises sharply, reaching a final cumulative frequency of 64 at the boundary of 35.2.

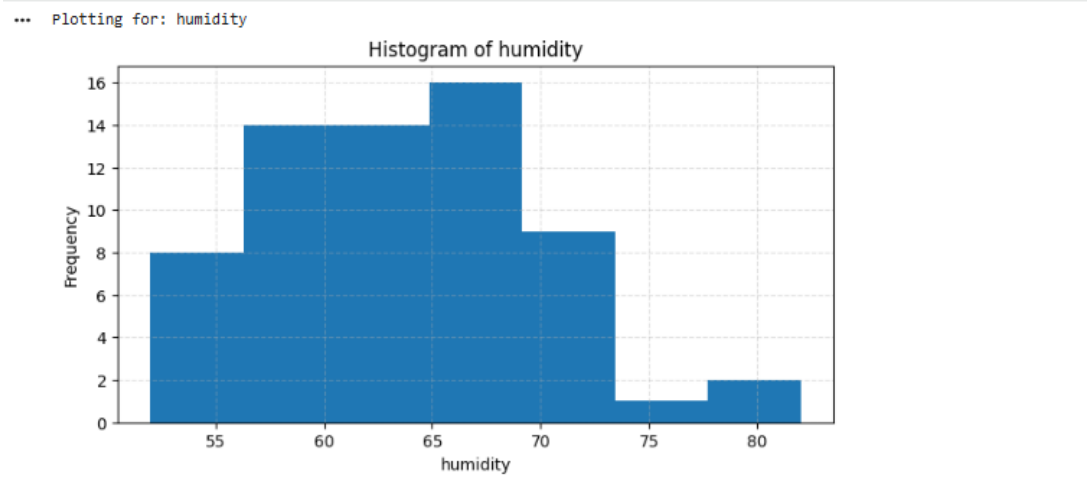


Figure 18: Histogram of humidity

A bar-based representation of the same humidity data seen in the frequency polygon. The most frequent humidity range is between 65 and 70, with a height of 16. The distribution shows significant activity between 55 and 75, with very few occurrences above a humidity level of 75.

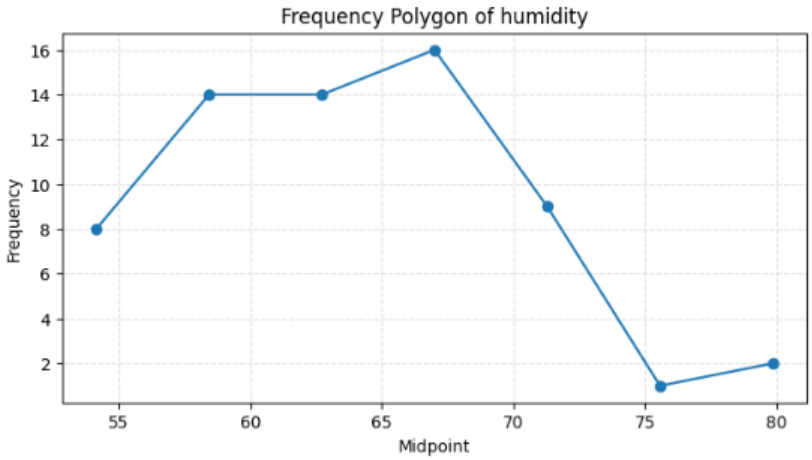


Figure 19: Frequency Polygon of humidity

A line graph showing how humidity frequencies are distributed across various midpoints. The frequency starts at 8 (midpoint 54), peaks at 16 (midpoint 67), and hits a low of 1 (midpoint 76) before a slight rise at the end.

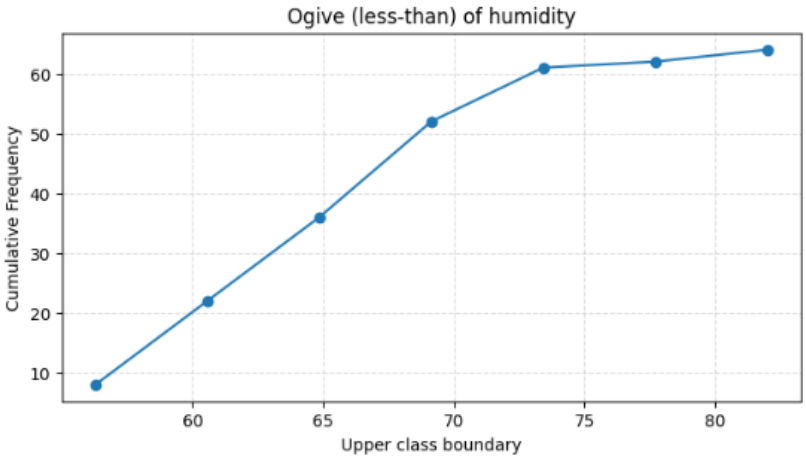


Figure 20: Ogive (less-than) of humidity

This chart tracks the cumulative frequency of humidity levels. Unlike the temperature ogive, this curve shows a more consistent, steady increase from the first boundary (56) to the last (82). It concludes at a total cumulative frequency of 64.

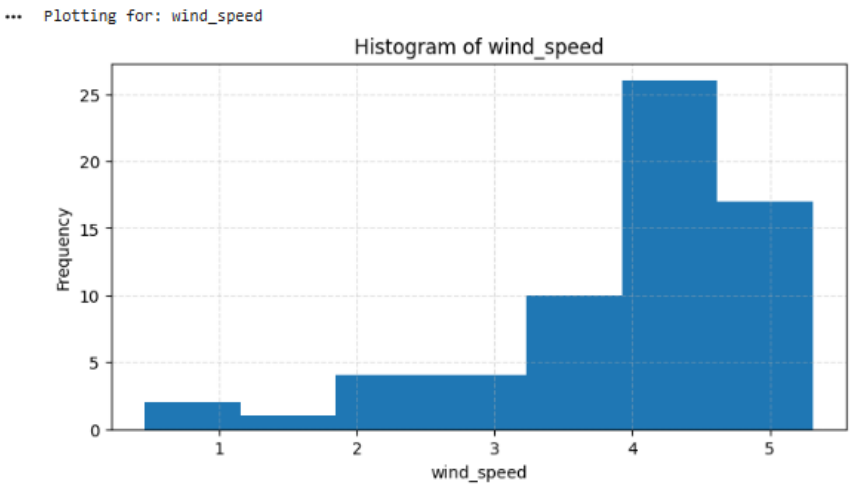


Figure 21: Histogram of wind_speed

This histogram visualizes the distribution of wind speed observations. The data is skewed toward higher values, with the tallest bar located between the 4 and 4.5 range, representing a frequency of 26. Observations start at a wind speed of approximately 0.5 and go up to 5.5.

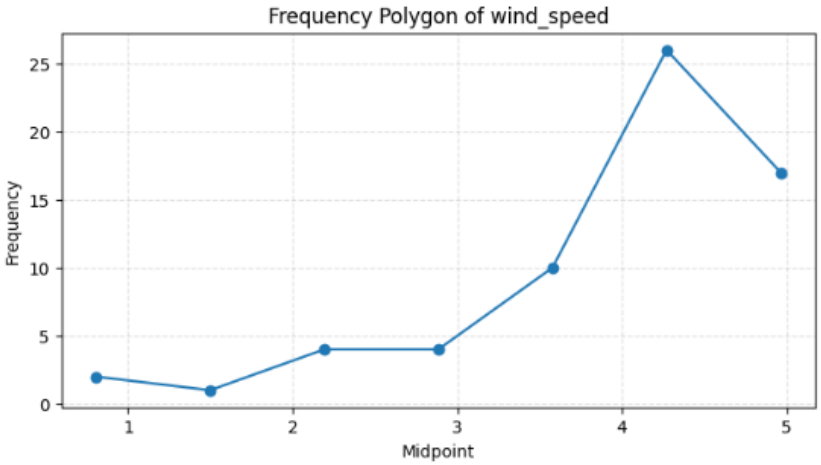


Figure 22: Frequency Polygon of wind_speed

This polygon tracks frequency against various midpoints for wind speed. The frequency peaks at 26 near a midpoint of approximately 4.25, after starting at a frequency of 2 for the first midpoint.

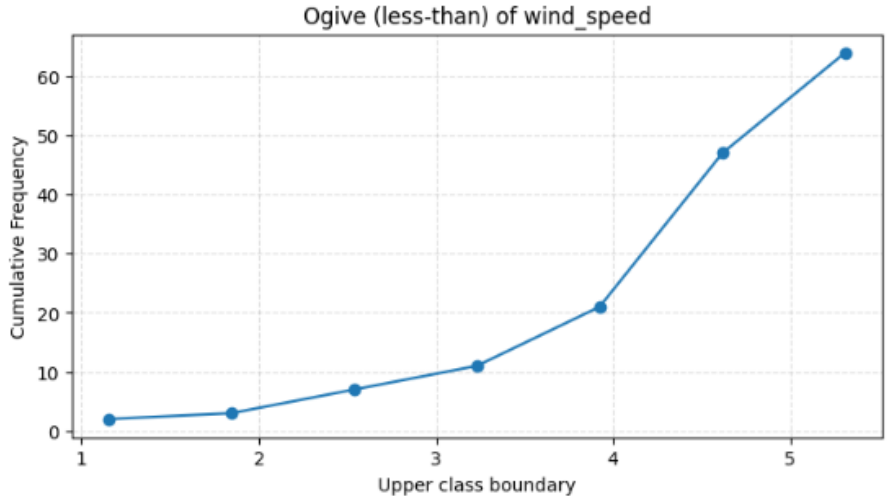


Figure 23: Ogive (less-than) of wind_speed

This chart shows an upward-sloping cumulative frequency curve for wind speed data. The curve reaches a final cumulative frequency of 64 at the highest upper class boundary shown.

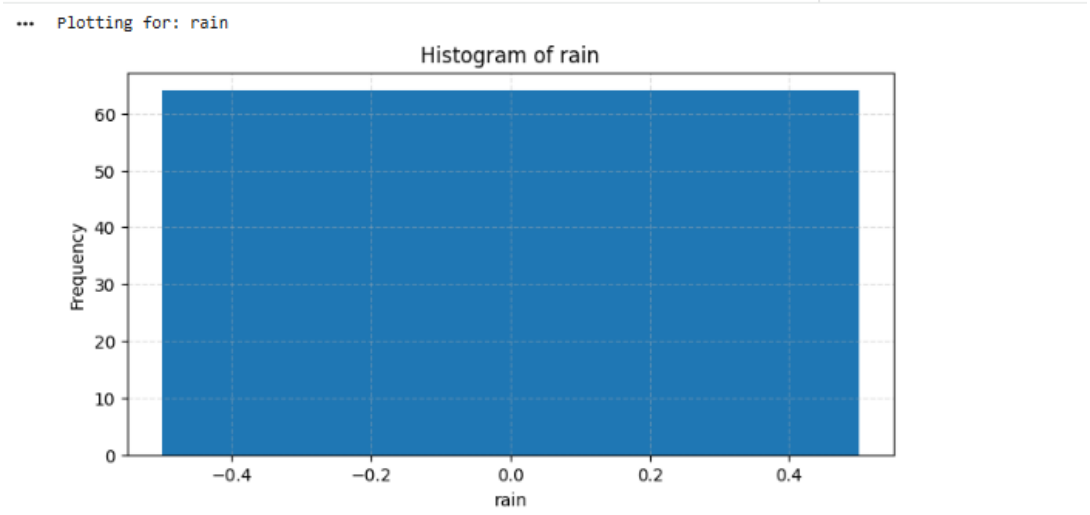


Figure 24: Histogram of rain

The histogram shows a single, wide bar centered on the value 0.0. The height of this bar indicates a frequency of 64.

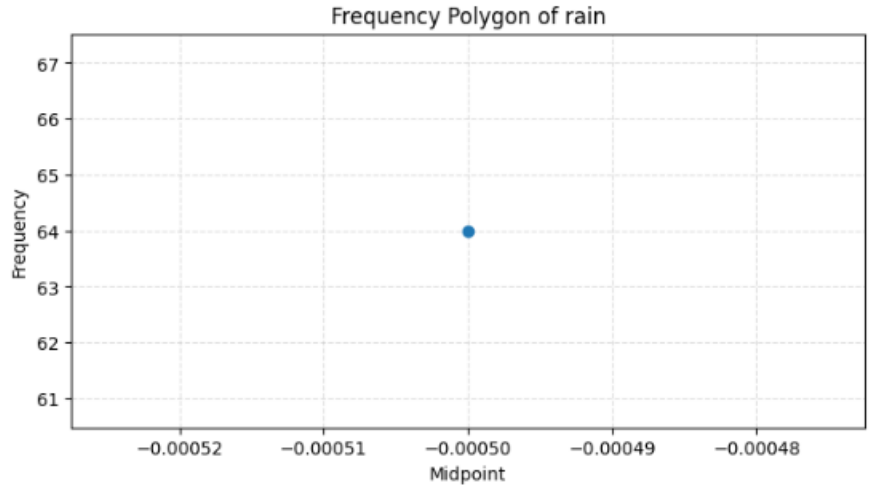


Figure 25: Frequency Polygon of rain

This graph consists of a single point, representing a frequency of 64. The x-axis identifies the midpoint for this point as approximately -0.00050.

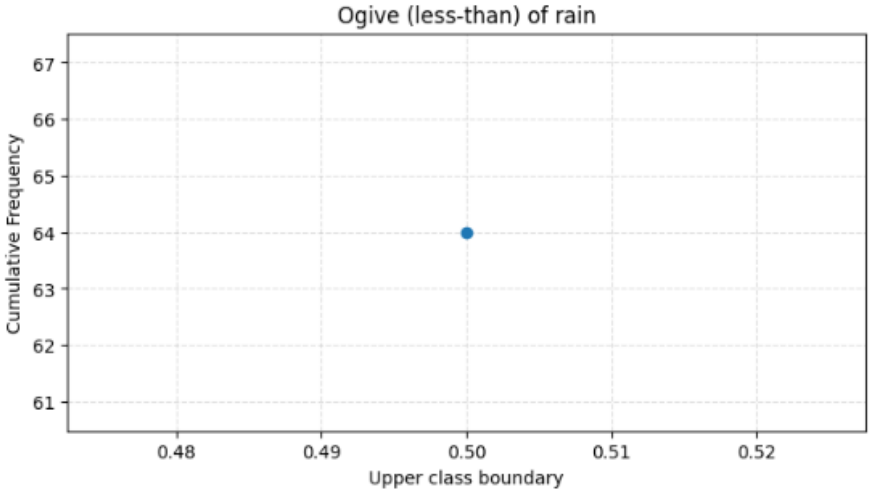


Figure 26: Ogive (less-than) of rain

This graph displays a single data point plotted at a cumulative frequency of 64. The y-axis represents Cumulative Frequency ranging from 61 to 67, while the x-axis shows the Upper class boundary at a value of 0.50.

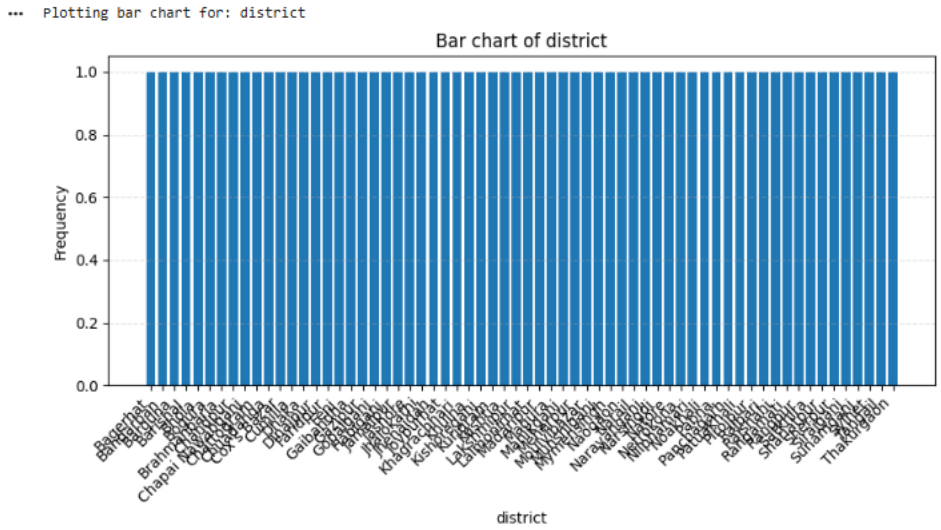


Figure 27: Bar chart of district

The chart features a dense collection of bars representing various districts along the x-axis. Every district listed has an identical frequency of 1.0.

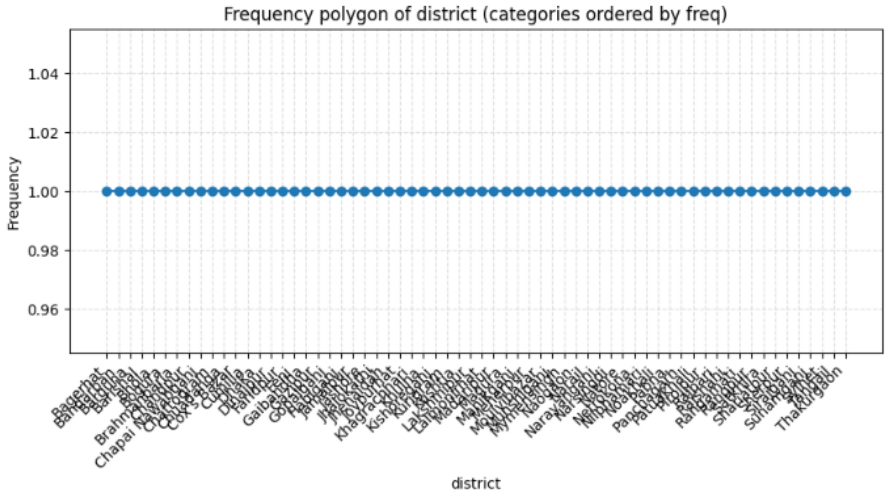


Figure 28: Frequency polygon of district (categories ordered by freq)

Similar to the bar chart in image 35.png, this polygon plots the frequency for districts. It appears as a flat horizontal line of dots, as every district has a constant frequency of 1.0.

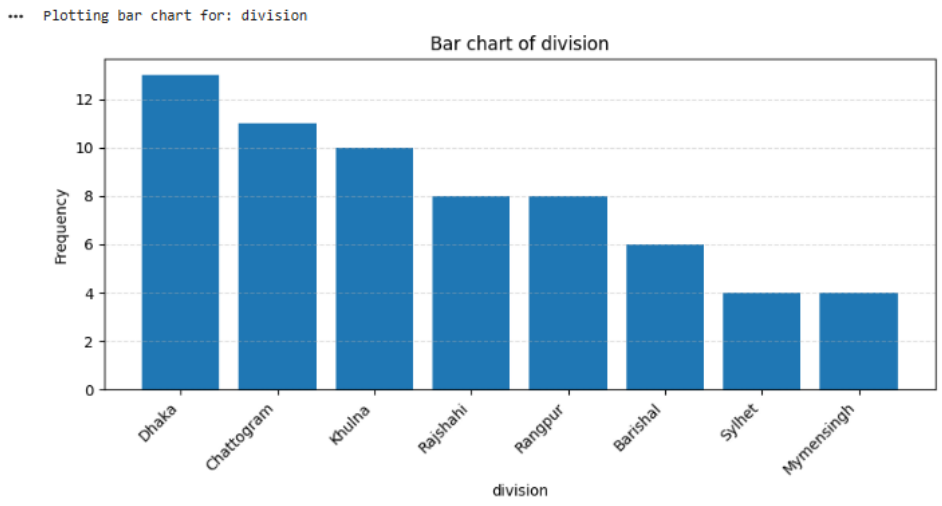


Figure 29: Bar chart of division

This bar chart visualizes the same data as image 39.png, with divisions sorted by their frequency. The bars decrease in height from Dhaka at the far left (13) to Sylhet and Mymensingh at the far right (4 each).

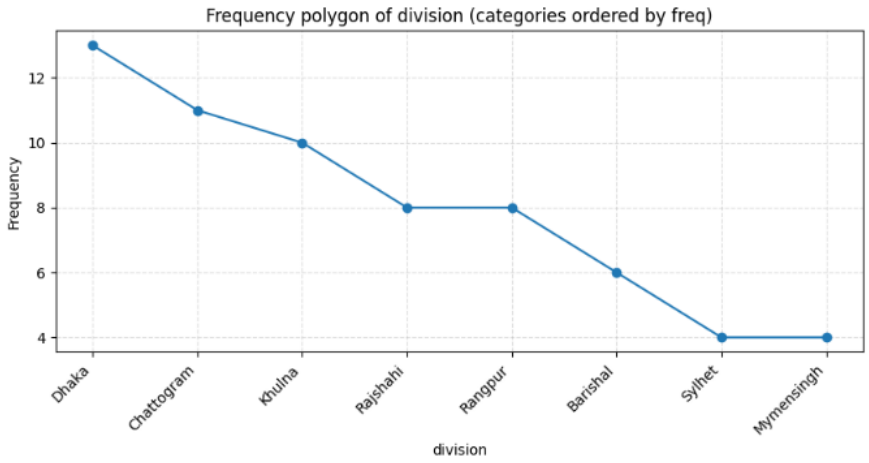


Figure 30: Frequency polygon of division (categories ordered by freq)

This line graph shows the distribution of frequencies across eight divisions in descending order. Dhaka has the highest frequency (13), followed by Chattogram (11), Khulna (10), Rajshahi (8), Rangpur (8), Barishal (6), Sylhet (4), and Mymensingh (4).

3.5 Part E (Task 03) : Analysis and Conclusion

The temperature data shows mostly high values. Most readings stay close to 32°C. The highest frequency appears in this range. There is no observation exactly at 30°C. This gap suggests a possible shift or separation in temperature behavior. Humidity shows a more stable pattern. The values follow an almost normal shape. The peak occurs around

67% humidity. Most readings fall between 55% and 75%. This indicates moderate and consistent moisture in the air.

Wind speed values are mostly high. The distribution is left skewed. Very few readings appear at low wind speeds. The highest frequency is near the 4.25 midpoint. This means strong winds were common during the period. Rainfall shows no variation at all. All recorded values are zero. The histogram has only one bar. This clearly indicates a completely dry period during data collection.

The geographic data shows uneven coverage across divisions. Dhaka has the highest number of observations. Sylhet and Mymensingh have the lowest counts. At the district level the distribution is perfectly uniform. Each district appears exactly once. Overall the dataset reflects hot dry and windy conditions. The absence of rainfall and high wind speed are the most notable features. Temperature remains high with an unusual gap around 30°C. The data is geographically broader at the district level but more concentrated in the Dhaka division.

3.6 Part F (Task 04) : Challenges

Major challenge was choosing the correct class intervals for histograms. This was important for temperature and humidity data. Too many intervals made the graph rough and confusing. It also created empty gaps like the zero frequency gap in temperature. Too few intervals hid important peaks. I solved this using trial and error. I selected intervals that showed the main peak clearly. At the same time the overall shape stayed visible.

The rain data created a different problem. Every value was 0.0. Normal graphs do not work well with no variation. The histogram looked flat and empty. Instead of removing it I kept the single bar. I also used a point based frequency polygon. This helped show that no rain occurred. The lack of variation itself became an important result.

Another minor challenge was handling reserved characters in LaTeX. Because variable names like temp_c and wind_speed caused errors. LaTeX treats the underscore as a subscript symbol. This breaks compilation in normal text. I fixed this by escaping the underscore using. I also used the format to show variable names as code. This made the document clear and error free.

There was also an issue with categorical data organization. The division and district data was hard to read at first. Alphabetical order did not show which area had more data. I fixed this by sorting categories by frequency. This made the chart easier to understand. Dhaka became clearly dominant. Mymensingh and Sylhet appeared as the least represented.

3.7 Part G : Submission

Submitted successfully!

4 Milestone 4: Measures of Central Tendency and Dispersion

This milestone shows deeper analysis of variables. I will compute descriptive stats, confidence intervals, tests and correlations.

4.1 Part A : Introduction

In this milestone, I will analyze numerical data from my dataset using statistical measures. Specifically, I will calculate the mean, median, mode, variance, and standard deviation of selected numerical columns. This helps in understanding the central tendency and spread of the data.

4.2 Part B : Dataset

I used the cleaned dataset from previous milestones. Dataset used here: 2025-08-10.csv

```
Loaded: 2025-08-10.csv
Shape: (64, 18)
Columns: ['date', 'district', 'division', 'lat', 'lon', 'temp_c', 'humidity', 'pressure', 'wind_speed', 'clouds', 'rain', 'aqi', 'pm2_5', 'pm10', 'o3', 'no2', 'so2', 'co']
```

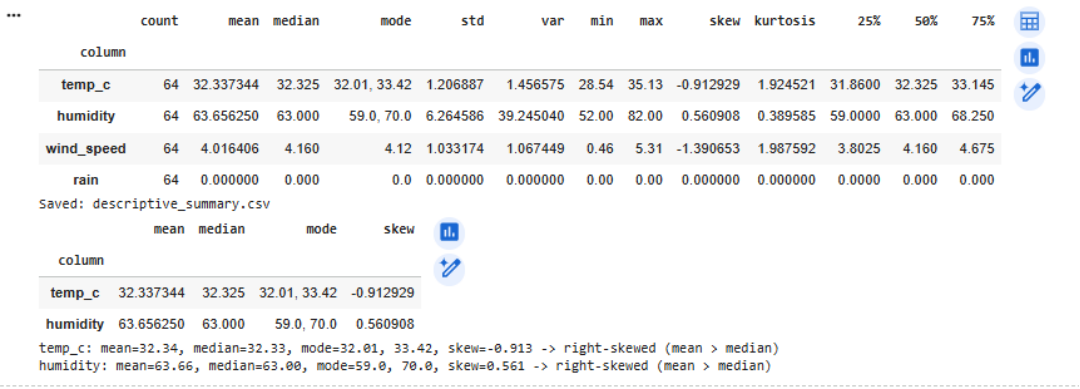
	date	district	division	lat	lon	temp_c	humidity	pressure	wind_speed	clouds	rain	aqi	pm2_5	pm10	o3	no2	so2	co
0	2025-08-10	Bagerhat	Khulna	22.655478	89.794181	32.00	63	1006	3.94	100	0.0	1	1.43	3.24	40.23	0.13	0.18	98.31
1	2025-08-10	Bandarban	Chattogram	21.787476	92.412475	33.56	54	1006	1.09	96	0.0	1	0.94	1.48	31.99	0.04	0.02	79.80
2	2025-08-10	Barguna	Barishal	22.131336	90.117243	31.90	65	1006	5.18	100	0.0	1	1.10	2.93	33.37	0.03	0.09	85.17
3	2025-08-10	Barishal	Barishal	22.493403	90.354801	33.42	55	1006	5.08	100	0.0	1	0.99	2.44	34.83	0.04	0.07	85.93
4	2025-08-10	Bhola	Barishal	22.143569	90.790409	29.38	77	1007	4.41	98	0.0	1	0.51	1.29	32.46	0.02	0.03	82.80
5	2025-08-10	Bogura	Rajshahi	24.850066	89.372843	33.94	57	1004	4.17	84	0.0	3	30.62	34.07	66.92	8.73	11.25	293.51
6	2025-08-10	Brahmanbaria	Chattogram	23.960600	91.119089	33.06	66	1006	4.12	75	0.0	1	1.23	2.12	32.13	1.03	0.42	109.33
7	2025-08-10	Chandpur	Chattogram	23.224176	90.653100	32.58	61	1006	4.26	98	0.0	1	0.67	1.39	33.11	0.34	0.27	89.17
8	2025-08-10	Chapai Nawabganj	Rajshahi	24.599887	88.285047	35.13	52	1004	4.03	76	0.0	3	35.50	38.31	79.41	7.30	15.25	304.89
9	2025-08-10	Chattogram	Chattogram	22.333778	91.834435	31.93	66	1007	4.12	40	0.0	1	0.61	1.18	29.40	0.03	0.02	79.32

Figure 31: Dataset

4.3 Part C (Task 1) : Measures of Central Tendency

Chosen numerical columns (at least two):

temp_c (temperature in Celsius): directly reflects weather conditions and varies by location. humidity (%): an important atmospheric indicator that also varies across districts. These two variables are meaningful, numeric, and measured in different units, so comparing their central tendency and dispersion is informative.



column	count	mean	median	mode	std	var	min	max	skew	kurtosis	25%	50%	75%
temp_c	64	32.337344	32.325	32.01, 33.42	1.206887	1.456575	28.54	35.13	-0.912929	1.924521	31.8600	32.325	33.145
humidity	64	63.656250	63.000	59.0, 70.0	6.264586	39.245040	52.00	82.00	0.560908	0.389585	59.0000	63.000	68.250
wind_speed	64	4.016406	4.160	4.12	1.033174	1.067449	0.46	5.31	-1.390653	1.987592	3.8025	4.160	4.675
rain	64	0.000000	0.000	0.0	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.0000	0.000	0.000

Saved: descriptive_summary.csv

column	mean	median	mode	skew
temp_c	32.337344	32.325	32.01, 33.42	-0.912929
humidity	63.656250	63.000	59.0, 70.0	0.560908

temp_c: mean=32.34, median=32.33, mode=32.01, 33.42, skew=-0.913 -> right-skewed (mean > median)
humidity: mean=63.66, median=63.00, mode=59.0, 70.0, skew=0.561 -> right-skewed (mean > median)

Figure 32: Central Tendency

Interpretation (Central Tendency) I compare mean, median, and mode for temp_c and humidity using the table printed above. If the mean is greater than the median, the distribution tends to be right-skewed (a longer right tail). If the mean is less than the median, it tends to be left-skewed. The mode represents the most frequently occurring value(s). If mean/median/mode are very different, it suggests skewness or possible outliers.

4.4 Part D (Task 2) : Measures of Dispersion

This section computes variance and standard deviation for the chosen numeric columns and interprets which variable is more/less variable. I also plot histograms with vertical lines for mean, median, and mode to visually compare central tendency and see dispersion. (Additional boxplots/scatterplots are included below as supporting visualizations.)

column	count	mean	variance (s^2)	std dev (s)
temp_c	64	32.337344	1.456575	1.206887
humidity	64	63.656250	39.245040	6.264586

```
Most variable (largest std dev): humidity
Most consistent (smallest std dev): temp_c
```

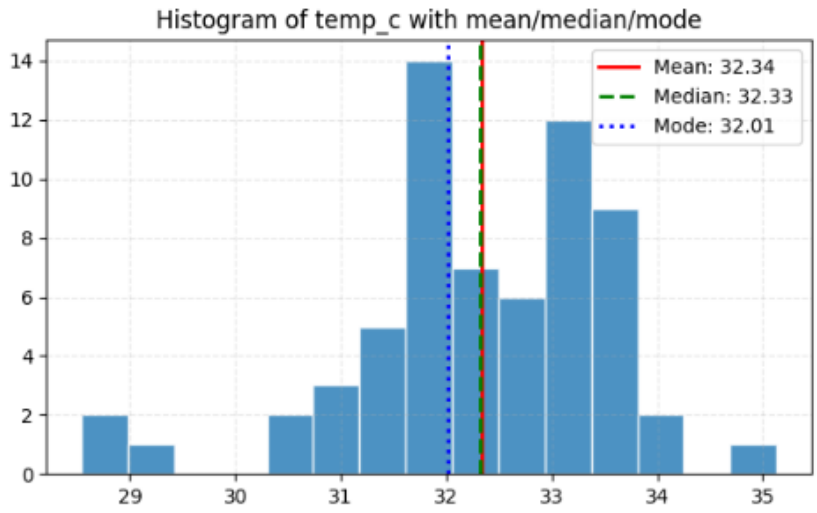


Figure 33: Output 1

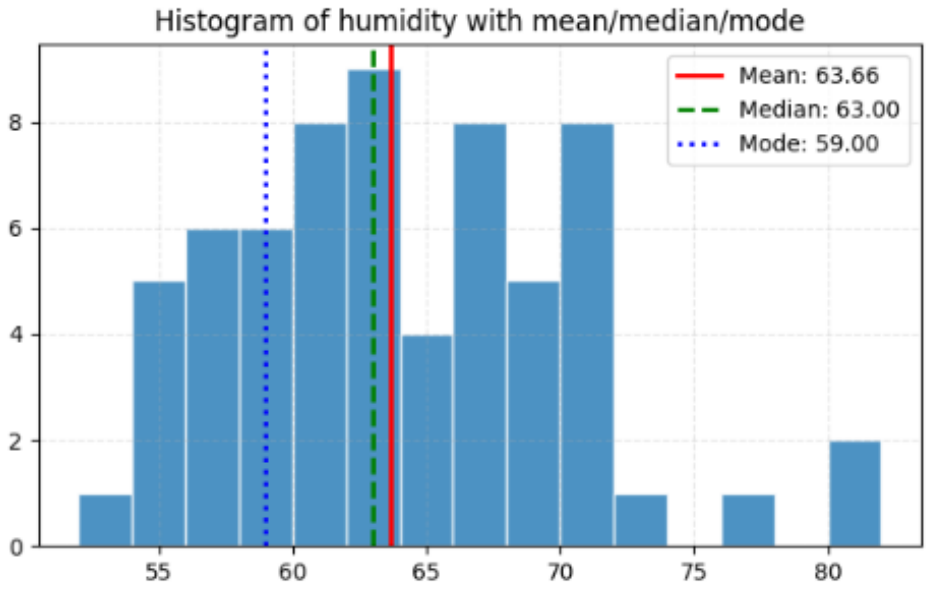


Figure 34: Output 2

Interpretation (Dispersion + Histogram) The variance and standard deviation tables show how spread out the values are around the mean. The column with the larger standard deviation is more variable (less consistent), while a smaller standard deviation indicates more consistency. In the histogram, a wider distribution indicates greater dis-

persion; a narrower distribution indicates less dispersion. The vertical lines help compare mean/median/mode visually and see skewness (e.g., mean pulled toward the tail).

4.5 Part E (Task 3) : Visualization (Optional but Encouraged)

This section computes 95% confidence intervals for means and performs one-sample t-tests.

95% CI for mean of temp_c: 32.035872436075735 32.63881506392427 (n=64) One-sample t-test of mean == 32.34: t=-0.018, p=0.986, n=64 95% CI for mean of humidity: 62.09140293826054 65.22109706173946 (n=64) One-sample t-test of mean == 63.66: t=-0.005, p=0.9962, n=64

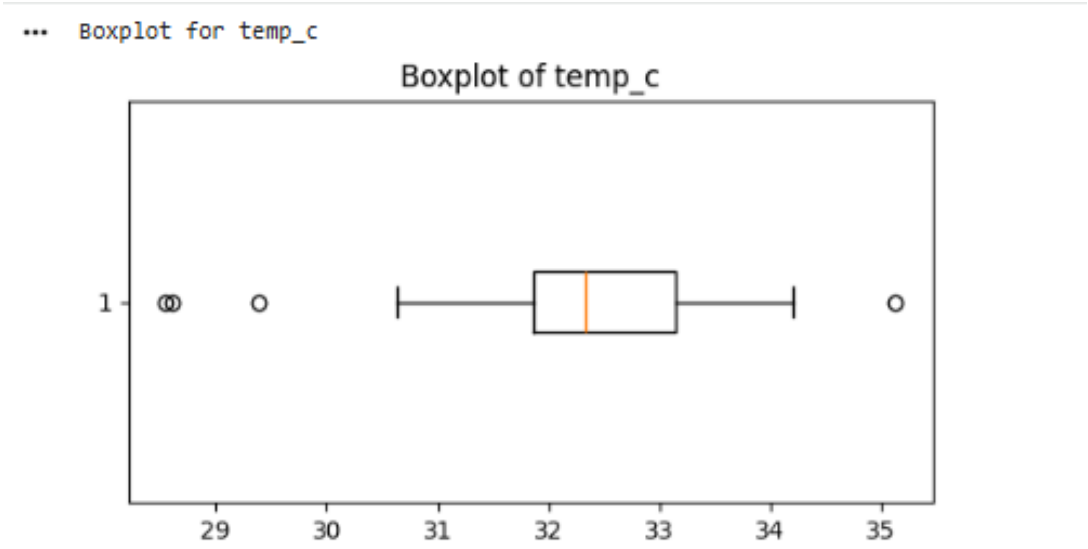


Figure 35: Visualization 1

```
Boxplot for humidity
```

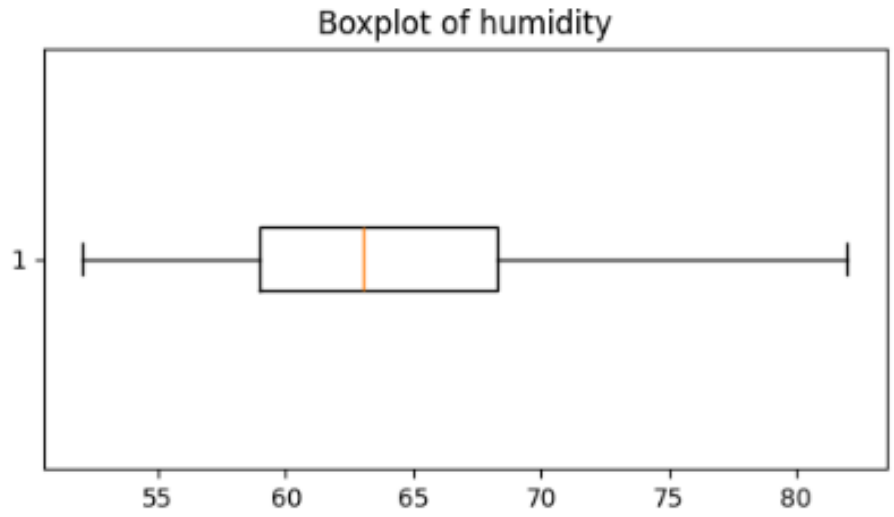


Figure 36: Visualization 2

```
Boxplot for wind_speed
```

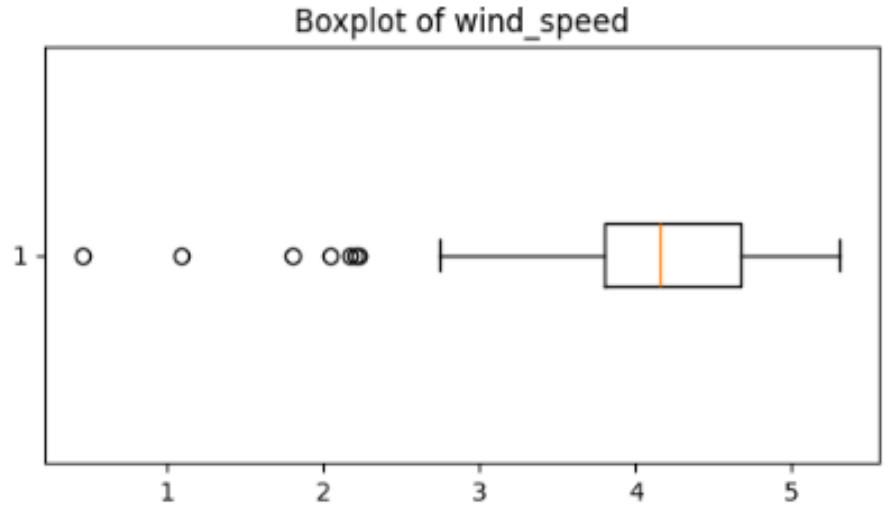


Figure 37: Visualization 3

```
Boxplot for rain
```

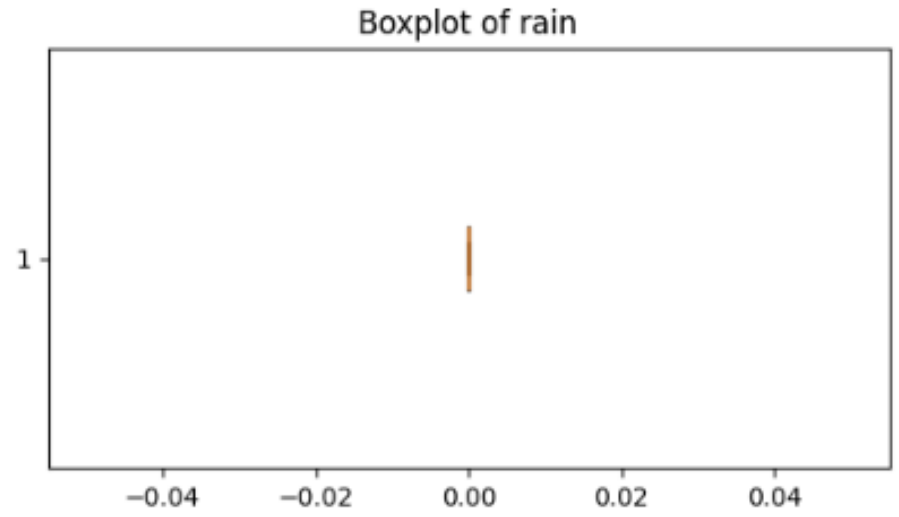


Figure 38: Visualization 4

```
Scatter: temp_c vs humidity
```

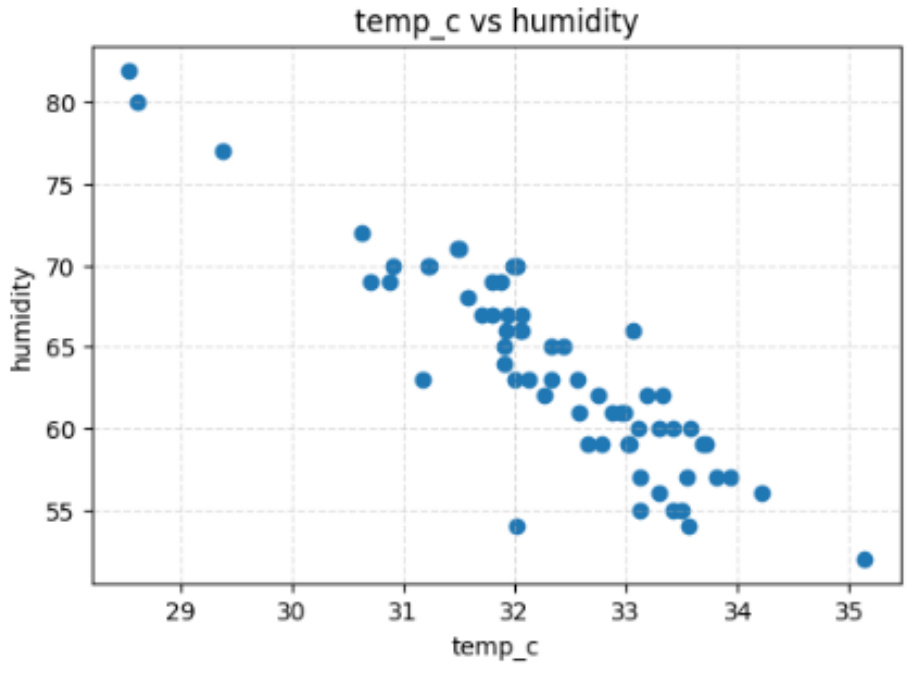


Figure 39: Visualization 5

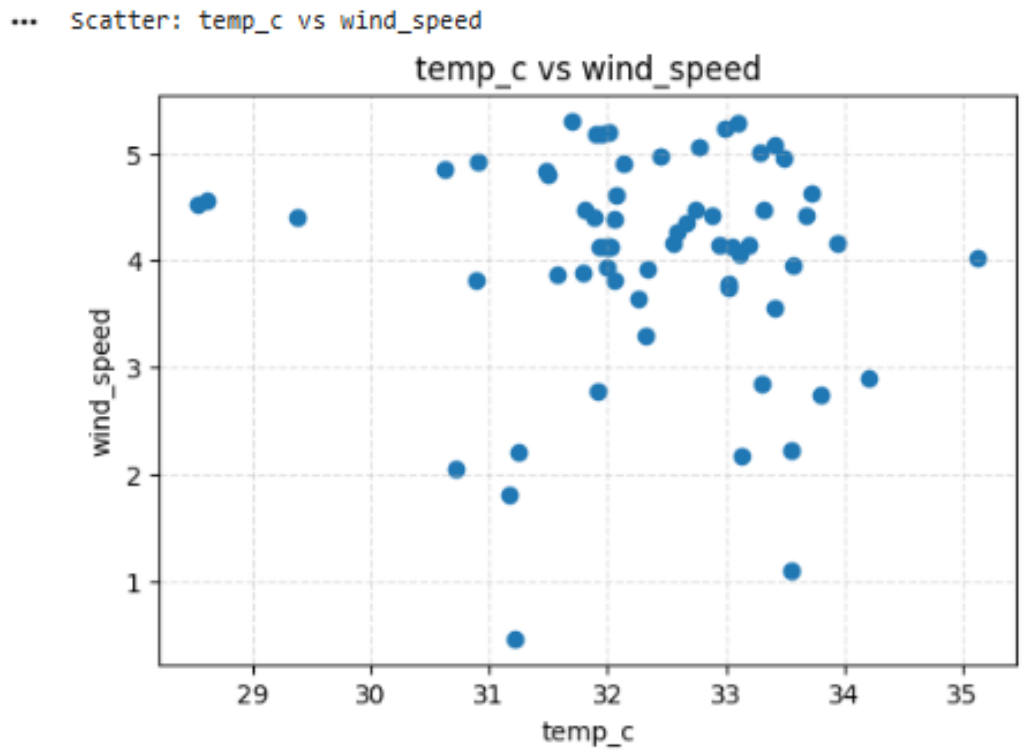


Figure 40: Visualization 6

Standard Errors assume that the covariance matrix of the errors is correctly specified.

4.6 Part F (Task 4) : Analysis and Conclusion

Central tendency:

I compare the mean, median, and mode for temp_c and humidity to understand the typical values. When the mean is larger than the median, it often suggests a right-skewed distribution; when the mean is smaller, it suggests a left-skew.

I compare the standard deviations (and variances) to decide which variable is more consistent (smaller spread) and which is more variable (larger spread).

After reviewing the tables and histograms, I summarize how the distributions behave and which variable shows greater variability. A very small sample size or outliers can affect the mean and standard deviation.

4.7 Part G : Submission

Submitted successfully!

5 Milestone 5: Introduction to Probability

This milestone covers basic probability using my dataset. I will define events, compute probabilities, and show simple checks.

5.1 Part A : Introduction

In this milestone, I will begin exploring the concept of probability. Probability helps us quantify uncertainty and measure how likely an event is to occur. I will work with simple events from my dataset and compute their probabilities. This milestone focuses on: 1. Basic probability rules 2. Events and sample spaces 3. Finding probabilities from real data

5.2 Part B : Dataset

I used the dataset from previous milestones. Dataset used here: 2025-08-10.csv

```
Loaded: 2025-08-10.csv
Shape: (64, 18)
Columns: ['date', 'district', 'division', 'lat', 'lon', 'temp_c', 'humidity', 'pressure', 'wind_speed', 'clouds', 'rain', 'aqi', 'pm2_5', 'pm10', 'o3', 'no
```

	date	district	division	lat	lon	temp_c	humidity	pressure	wind_speed	clouds	rain	aqi	pm2_5	pm10	o3	no2	so2	co
0	2025-08-10	Bagerhat	Khulna	22.655478	89.794181	32.00	63	1006	3.94	100	0.0	1	1.43	3.24	40.23	0.13	0.18	98.31
1	2025-08-10	Bandarban	Chattogram	21.787476	92.412475	33.56	54	1006	1.09	96	0.0	1	0.94	1.48	31.99	0.04	0.02	79.80
2	2025-08-10	Barguna	Barishal	22.131336	90.117243	31.90	65	1006	5.18	100	0.0	1	1.10	2.93	33.37	0.03	0.09	85.17
3	2025-08-10	Barishal	Barishal	22.493403	90.354801	33.42	55	1006	5.08	100	0.0	1	0.99	2.44	34.83	0.04	0.07	85.93
4	2025-08-10	Bhola	Barishal	22.143569	90.790409	29.38	77	1007	4.41	98	0.0	1	0.51	1.29	32.46	0.02	0.03	82.80
5	2025-08-10	Bogura	Rajshahi	24.850066	89.372843	33.94	57	1004	4.17	84	0.0	3	30.62	34.07	66.92	8.73	11.25	293.51
6	2025-08-10	Brahmanbaria	Chattogram	23.960600	91.119089	33.06	66	1006	4.12	75	0.0	1	1.23	2.12	32.13	1.03	0.42	109.33
7	2025-08-10	Chandpur	Chattogram	23.224176	90.653100	32.58	61	1006	4.26	98	0.0	1	0.67	1.39	33.11	0.34	0.27	89.17

Figure 41: Dataset

5.3 Part C (Task 1) : Defining Events

Selected two columns and define events. Here I use temp_c (numeric) and district (categorical).

Events (defined here): A = temperature ¿ 25 (temp_c ¿ 25) B = temperature between 20 and 25 (20 ¡= temp_c ¡= 25) C = district is the most frequent district in the dataset I chose these because temperature is important and district shows groups.

```python
# C: define events and show counts
col_temp = 'temp_c'
col_dist = 'district'

if col_temp not in df.columns or col_dist not in df.columns:
    print('Required columns missing. Check dataset columns.')
else:
    n = len(df)
    top_dist = df[col_dist].dropna().astype(str).value_counts().idxmax()
    print('Top district:', top_dist)
    A = df[col_temp] > 25
    B = (df[col_temp] >= 20) & (df[col_temp] <= 25)
    C = df[col_dist].astype(str) == str(top_dist)
    print('\nTotal outcomes (n):', n)
    print('Count A (temp > 25):', int(A.sum()))
    print('Count B (20 <= temp <= 25):', int(B.sum()))
    print('Count C (district == top):', int(C.sum()))
```

```
Top district: Bagerhat

Total outcomes (n): 64
Count A (temp > 25): 64
Count B (20 <= temp <= 25): 0
Count C (district == top): 1
```

Figure 42: Defining Events

5.4 Part D (Task 2) : Calculating Basic Probability

For each event computed empirical probability: P(E) = favorable / total. Verify 0 ¡= P ¡= 1 and wrote a short sentence interpretation.

```python
# D: compute probabilities and print sentences
col_temp = 'temp_c'
col_dist = 'district'
if col_temp not in df.columns or col_dist not in df.columns:
    print('Required columns missing.')
else:
    n = len(df)
    top_dist = df[col_dist].dropna().astype(str).value_counts().idxmax()
    A = df[col_temp] > 25
    B = (df[col_temp] >= 20) & (df[col_temp] <= 25)
    C = df[col_dist].astype(str) == str(top_dist)
    pA = A.sum() / n
    pB = B.sum() / n
    pC = C.sum() / n
    print('P(A) = count(A)/n =', int(A.sum()), '/', n, '=', f"{pA:.3f}")
    print('Check 0<=P(A)<=1 :', 0 <= pA <= 1)
    if pA==0:
        print('Interpretation: No rows have temp > 25.')
    else:
        print('Interpretation: Probability of temp > 25 is about', f"{pA:.1%}")
    print('\nP(B) =', int(B.sum()), '/', n, '=', f"{pB:.3f}")
    print('Check 0<=P(B)<=1 :', 0 <= pB <= 1)
    if pB==0:
        print('Interpretation: No rows in 20-25 temp.')
    else:
        print('Interpretation: Probability of temp between 20 and 25 is about', f"{pB:.1%}")
    print('\nP(C) =', int(C.sum()), '/', n, '=', f"{pC:.3f}")
    print('Check 0<=P(C)<=1 :', 0 <= pC <= 1)
    if pC==0:
        print('Interpretation: Top district not present?')
    else:
        print('Interpretation: Probability of being in top district is about', f"{pC:.1%}")
```

```
P(A) = count(A)/n = 64 / 64 = 1.000
Check 0<=P(A)<=1 : True
Interpretation: Probability of temp > 25 is about 100.0%

P(B) = 0 / 64 = 0.000
Check 0<=P(B)<=1 : True
Interpretation: No rows in 20-25 temp.

P(C) = 1 / 64 = 0.016
Check 0<=P(C)<=1 : True
Interpretation: Probability of being in top district is about 1.6%
```

Figure 43: Calculating Basic Probability

5.5 Part E (Task 3) : Combined Events

Computed intersections, unions, complements. Verifying the given rule.

```
P(A) = 1.000   Count: 64
P(B) = 0.000   Count: 0
P(C) = 0.016   Count: 1

P(A ∩ B) = 0.000   Count: 0
P(A ∩ C) = 0.016   Count: 1
P(B ∩ C) = 0.000   Count: 0

P(A ∪ B) = 1.000   Count: 64
P(A ∪ C) = 1.000   Count: 64

P(A^c) = 0.000   Count: 0
P(C^c) = 0.984   Count: 63

Check union rule: P(A ∪ B) ?= P(A) + P(B) - P(A ∩ B)
LHS: 1.000
RHS: 1.000
Equal (within float tol): True
```

Figure 44: Combined Events

5.6 Part F (Task 4) : Visualization

Bar charts and highlighted favorable vs. total outcomes.

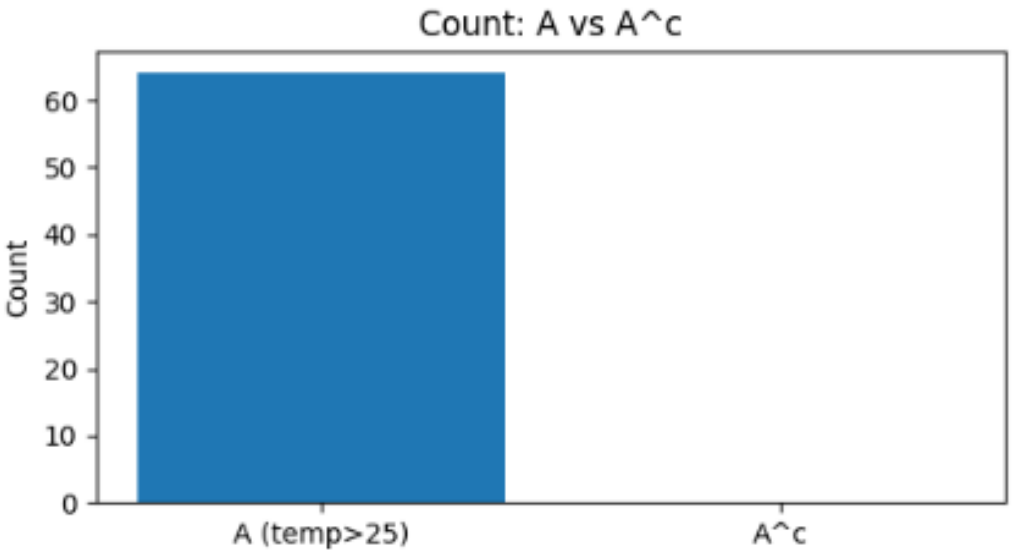


Figure 45: Visualization 1

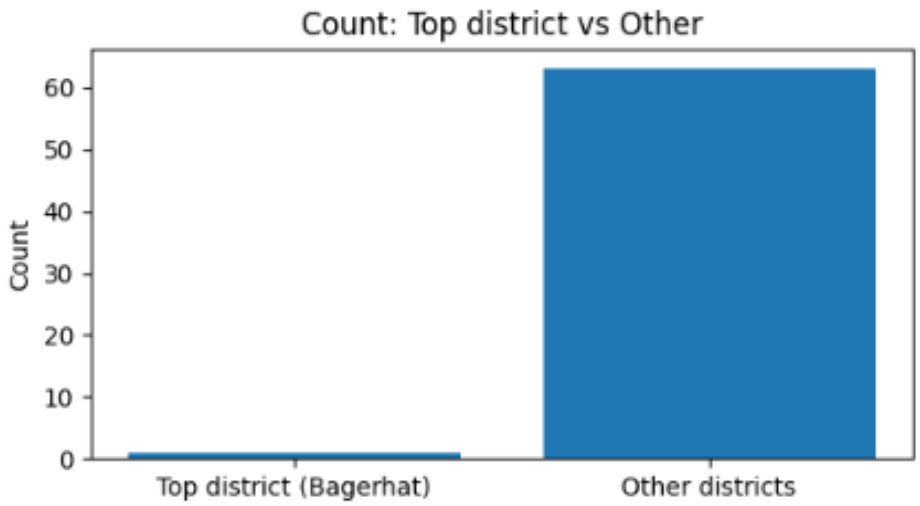


Figure 46: Visualization 2

```
Intervals and probabilities (relative freq):
28.54 - 29.48: 0.047 (n=3)
29.48 - 30.42: 0.000 (n=0)
30.42 - 31.36: 0.109 (n=7)
31.36 - 32.31: 0.328 (n=21)
32.31 - 33.25: 0.281 (n=18)
33.25 - 34.19: 0.203 (n=13)
34.19 - 35.13: 0.031 (n=2)
```

Figure 47: Visualization 3

5.7 Part G (Task 5) : Reflection and Conclusion

Bar charts and highlighted favorable vs. total outcomes.

This milestone helped me move from descriptive statistics to probability. I did not use coins or dice. I used my real dataset. This made probability easier to understand. It felt more practical and realistic. I defined events using temperature ranges and district counts. This showed how probability comes from data not guesses.

I learned the idea of empirical probability. I calculated probability by dividing favorable cases by total cases. All values stayed between 0 and 1. This confirmed my calculations were correct. It also helped me remember the basic rules of probability. Writing short explanations improved my ability to explain results clearly.

The analysis of combined events was important. I worked with union intersection and complement. This showed how events can overlap. It also showed how probabilities

change. I verified the rule. The rule matched my dataset. This increased my confidence in the data and method.

Graphs helped a lot in this milestone. Bar charts made the results easy to see. Favorable cases and total cases were clearly separated. This made comparisons simple. The visuals supported the numerical results. Some patterns were easier to notice in graphs than in tables.

Overall this milestone improved my understanding of probability. I now see probability as a tool for real data analysis. It connected earlier milestones with future topics. These include conditional probability and random variables. This task also helped me present results in a clear and logical way.

5.8 Part H : Submission

Submitted successfully!

6 Milestone 6: Conditional Probability, Independence, Bayes' Rule and Probability Distributions

This milestone extends basic probability to conditional probability, independence, Bayes' rule, and the Normal distribution. All probabilities are computed from the dataset.

6.1 Part A : Introduction

In the previous milestone, I explored basic probability and computed empirical probabilities using my dataset. This milestone builds on that foundation by introducing:

1. Conditional Probability
2. Independent vs. Dependent Events
3. Bayes' Rule
3. Probability Distributions (Normal Distribution Only).

Probability distributions help us model and understand randomness in real datasets. They are the core of statistical inference, machine learning, forecasting, and decision-making.

6.2 Part B : Dataset

I used the dataset from previous milestones. Dataset used here: 2025-08-10.csv

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

DATA_PATH = r"2025-08-10.csv"
df = pd.read_csv(DATA_PATH)
print('Dataset loaded')
print('Rows:', df.shape[0])
print('Columns:', df.columns.tolist())

df.head()
```

```
Dataset loaded
Rows: 64
Columns: ['date', 'district', 'division', 'lat', 'lon', 'temp_c', 'humidity', 'pressure', 'wind_speed', 'clouds', 'rain', 'aqi', 'pm2_5', 'pm10', 'o3', '
```

	date	district	division	lat	lon	temp_c	humidity	pressure	wind_speed	clouds	rain	aqi	pm2_5	pm10	o3	no2	so2	co
0	2025-08-10	Bagerhat	Khulna	22.655478	89.794181	32.00	63	1006	3.94	100	0.0	1	1.43	3.24	40.23	0.13	0.18	98.31
1	2025-08-10	Bandarban	Chattogram	21.787476	92.412475	33.56	54	1006	1.09	96	0.0	1	0.94	1.48	31.99	0.04	0.02	79.80
2	2025-08-10	Barguna	Barishal	22.131336	90.117243	31.90	65	1006	5.18	100	0.0	1	1.10	2.93	33.37	0.03	0.09	85.17
3	2025-08-10	Barishal	Barishal	22.493403	90.354801	33.42	55	1006	5.08	100	0.0	1	0.99	2.44	34.83	0.04	0.07	85.93
4	2025-08-10	Bhola	Barishal	22.143569	90.790409	29.38	77	1007	4.41	98	0.0	1	0.51	1.29	32.46	0.02	0.03	82.80

Figure 48: Dataset

6.3 Part C (Task 1) : Define Events

I selected one numerical and one categorical variable.

Let:

A = temp_c ¿ 25

B = humidity ¿ 70

C = district is the most frequent district

These events are meaningful for weather behavior.

```
col_temp = 'temp_c'
col_hum = 'humidity'
col_dist = 'district'

top_dist = df[col_dist].astype(str).value_counts().idxmax()
n = len(df)

A = df[col_temp] > 25
B = df[col_hum] > 70
C = df[col_dist].astype(str) == top_dist

print('Event A: temp > 25 | count =', A.sum())
print('Event B: humidity > 70 | count =', B.sum())
print('Event C: district =', top_dist, '| count =', C.sum())
print('Total n =', n)
```

```
Event A: temp > 25 | count = 64
Event B: humidity > 70 | count = 6
Event C: district = Bagerhat | count = 1
Total n = 64
```

Figure 49: For district

6.4 Part D (Task 02) : Conditional Probability

I computed P(A), P(B), and P(A — B). Then I explained results.

```
def P(x): return x.sum() / n

pA = P(A)
pB = P(B)
pA_given_B = P(A & B) / P(B) if P(B) > 0 else np.nan

print('P(A) =', round(pA,3))
print('P(B) =', round(pB,3))
print('P(A | B) =', round(pA_given_B,3))

print('\nInterpretation:')
print('Given humidity is high, chance of temp > 25 is', round(pA_given_B*100,1), '%')
```

```
P(A) = 1.0
P(B) = 0.094
P(A | B) = 1.0

Interpretation:
Given humidity is high, chance of temp > 25 is 100.0 %
```

Figure 50: Conditional Probability

6.5 Part E (Task 03) : Independence Check

I checked if A and B are independent using probability rule.

```
pA_and_B = P(A & B)
product = pA * pB

print('P(A ∩ B) =', round(pA_and_B,3))
print('P(A)P(B) =', round(product,3))

if abs(pA_and_B - product) < 0.01:
    print('Conclusion: A and B are approximately independent')
else:
    print('Conclusion: A and B are dependent')
```

```
P(A ∩ B) = 0.094
P(A)P(B) = 0.094
Conclusion: A and B are approximately independent
```

Figure 51: Independence Check

6.6 Part F (Task 04) : Bayes' Rule

I computed P(B — A) using Bayes' Rule and compared with data.

```
pB_given_A_emp = P(A & B) / P(A) if P(A) > 0 else np.nan
pB_given_A_bayes = (pA_given_B * pB) / pA if pA > 0 else np.nan

print('Empirical P(B | A) =', round(pB_given_A_emp,3))
print('Bayes P(B | A) =', round(pB_given_A_bayes,3))
print('Difference =', round(abs(pB_given_A_emp - pB_given_A_bayes),4))
```

```
Empirical P(B | A) = 0.094
Bayes P(B | A) = 0.094
Difference = 0.0
```

Figure 52: Bayes Rule

6.7 Part G (Task 05) : Probability Distribution (Normal Distribution)

I studied temp_c as a continuous random variable.

```
Mean (μ) = 32.337
Std Dev (σ) = 1.207
```

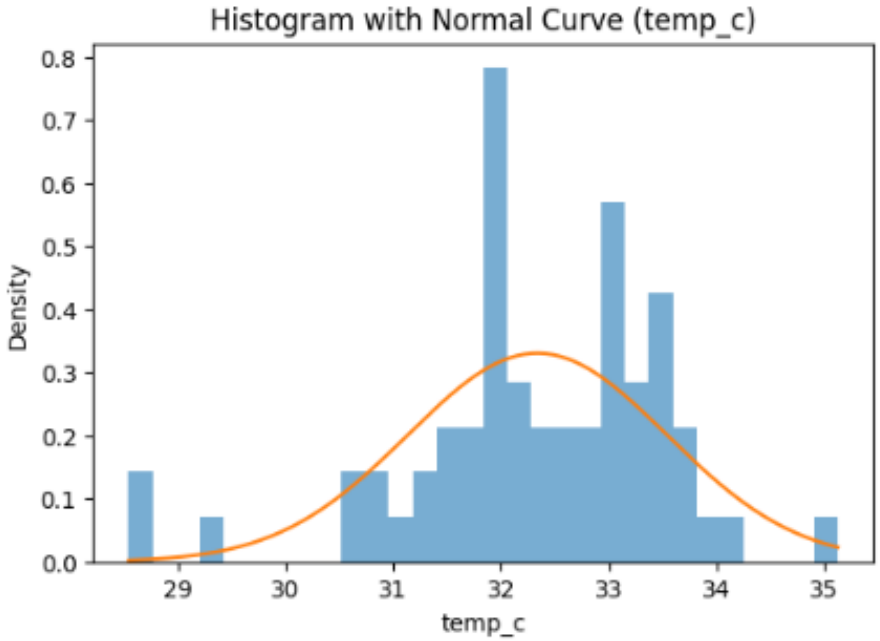


Figure 53: Histogram

```
p1 = 1 - norm.cdf(mu, mu, sigma)
p2 = norm.cdf(mu+sigma, mu, sigma) - norm.cdf(mu-sigma, mu, sigma)
p3 = norm.cdf(mu-2*sigma, mu, sigma)

print('P(X > μ) =', round(p1,3))
print('P(μ - σ < X < μ + σ) =', round(p2,3))
print('P(X < μ - 2σ) =', round(p3,3))
```

```
P(X > μ) = 0.5
P(μ - σ < X < μ + σ) = 0.683
P(X < μ - 2σ) = 0.023
```

Figure 54: Output

The judgment is based on histogram shape and statistics.

```
median = x.median()
print('Mean =', round(mu,3))
print('Median =', round(median,3))

if abs(mu - median) < 0.5:
    print('Mean and median are close. Shape looks symmetric.')
else:
    print('Mean and median differ. Distribution may be skewed.')
```

```
Mean = 32.337
Median = 32.325
Mean and median are close. Shape looks symmetric.
```

Figure 55: Judgement on Histogram

6.8 Part H (Task 06) : Reflection

Conditional probability changed how events relate. Events A and B are not fully independent. Temperature roughly follows a normal pattern. These ideas help in prediction and decision making.

This milestone moved probability beyond simple counting. I stopped focusing only on how often an event happens. I started thinking about how one condition affects another. This change made probability feel more realistic. It also felt more useful for real data analysis.

Conditional probability made this very clear. I observed that the chance of high temperature changes when humidity is already known. This showed that events in real datasets are connected. Weather variables especially influence each other. Conditional probability helped me describe this relationship clearly using data.

Checking independence was another important step. I compared combined probabilities with what would be expected if events were independent. The result showed that the events were not fully independent. This matched common sense and the data. It also taught me that independence should never be assumed. It must always be tested using real observations.

Applying Bayes' Rule was a key learning moment. I calculated probabilities in two different ways and found that the results matched. This showed that Bayes' Rule is not separate from data. It simply reorganizes information that is already known. This made the concept easier to understand. It also showed why Bayes' Rule is useful in prediction and decision making.

The Normal distribution part connected probability with data patterns. I treated temperature as a continuous variable. The histogram showed values clustering around a central point. The shape looked close to a bell curve. This supported using the Normal distribution to model temperature. This step prepared me for future topics like standard scores and confidence intervals.

Overall this milestone helped me see probability as a connected system. The ideas of conditional probability dependence Bayes' Rule and distributions worked together. They explained how uncertainty behaves in real data. These concepts are important for advanced statistics and machine learning. This milestone was an important step forward in my learning.

6.9 Submission

Submitted successfuly!

7 Milestone 7 : Simple Linear Regression (Manual Computation) and Correlation

This milestone studies the linear relationship between two numerical variables. I calculated regression parameters manually using formulas. No machine learning library is used.

7.1 Part A : Introduction

The previous milestones explored single-variable distributions. This milestone introduces Simple Linear Regression, a fundamental method to model and quantify the linear relationship between two continuous variables. The core objective is to calculate the best-fit line parameters manually, adhering to the requirement of avoiding high-level machine learning libraries like scikit-learn for the main tasks.

7.2 Part B : Knowledge Points - The Least Squares Method

Simple Linear Regression (SLR) models the relationship between an independent variable (X) and a dependent variable (Y) as a straight line:

$$\hat{Y}_i = \beta_0 + \beta_1 X_i$$

The parameters β_0 and β_1 are chosen to minimize the Sum of Squared Residuals (SSR), which is why this is called the Least Squares method.

1. Regression Parameters

1.1 The Slope (β_1)

The slope represents the expected change in $\hat{Y}$ for a one-unit change in X. It is calculated using the covariance of X and Y and the variance of X:

$$\beta_1 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2} = \frac{\text{Cov}(X, Y)}{\text{Var}(X)}$$

1.2 The Y-Intercept (β_0)

The intercept is the predicted value of Y when $X = 0$. It ensures the regression line passes through the centroid $(\bar{X}, \bar{Y})$ of the data:

$$\beta_0 = \bar{Y} - \beta_1 \bar{X}$$

2. Correlation and Model Fit

2.1 Pearson Correlation Coefficient (r)

This measures the strength and direction of the linear association (from -1 to 1).

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2 \sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

2.2 Coefficient of Determination (R^2)

$R^2 = r^2$. It represents the proportion of the variance in the dependent variable (Y) that is predictable from the independent variable (X). A value closer to 1 indicates a better fit.

7.3 Part C (Task 1) : Data Selection and Initial Visualization

Independent variable (X): temperature (temp_c)

Dependent variable (Y): humidity

These two variables are expected to have a relationship.

```
def P(x): return x.sum() / n

pA = P(A)
pB = P(B)
pA_given_B = P(A & B) / P(B) if P(B) > 0 else np.nan

print('P(A) =', round(pA,3))
print('P(B) =', round(pB,3))
print('P(A | B) =', round(pA_given_B,3))

print('\nInterpretation:')
print('Given humidity is high, chance of temp > 25 is', round(pA_given_B*100,1), '%')
```

```
P(A) = 1.0
P(B) = 0.094
P(A | B) = 1.0

Interpretation:
Given humidity is high, chance of temp > 25 is 100.0 %
```

Figure 56: Data Selection

7.4 Part D (Task 02) : Manual Calculation of Regression Parameters

I computed slope and intercept using formulas only.

```
pA_and_B = P(A & B)
product = pA * pB

print('P(A ∩ B) =', round(pA_and_B,3))
print('P(A)P(B) =', round(product,3))

if abs(pA_and_B - product) < 0.01:
    print('Conclusion: A and B are approximately independent')
else:
    print('Conclusion: A and B are dependent')
```

```
P(A ∩ B) = 0.094
P(A)P(B) = 0.094
Conclusion: A and B are approximately independent
```

Figure 57: Manual Calculation or Regression

7.5 Part E (Task 03) : Visualization of the Fit and Interpretation

I plotted the regression line and explained slope meaning.

```
pB_given_A_emp = P(A & B) / P(A) if P(A) > 0 else np.nan
pB_given_A_bayes = (pA_given_B * pB) / pA if pA > 0 else np.nan

print('Empirical P(B | A) =', round(pB_given_A_emp,3))
print('Bayes P(B | A) =', round(pB_given_A_bayes,3))
print('Difference =', round(abs(pB_given_A_emp - pB_given_A_bayes),4))
```

```
Empirical P(B | A) = 0.094
Bayes P(B | A) = 0.094
Difference = 0.0
```

Figure 58: Visualization

7.6 Part F (Task 04) : Strength of Relationship

I computed correlation and R^2 manually.

```
Mean (μ) = 32.337
Std Dev (σ) = 1.207
```

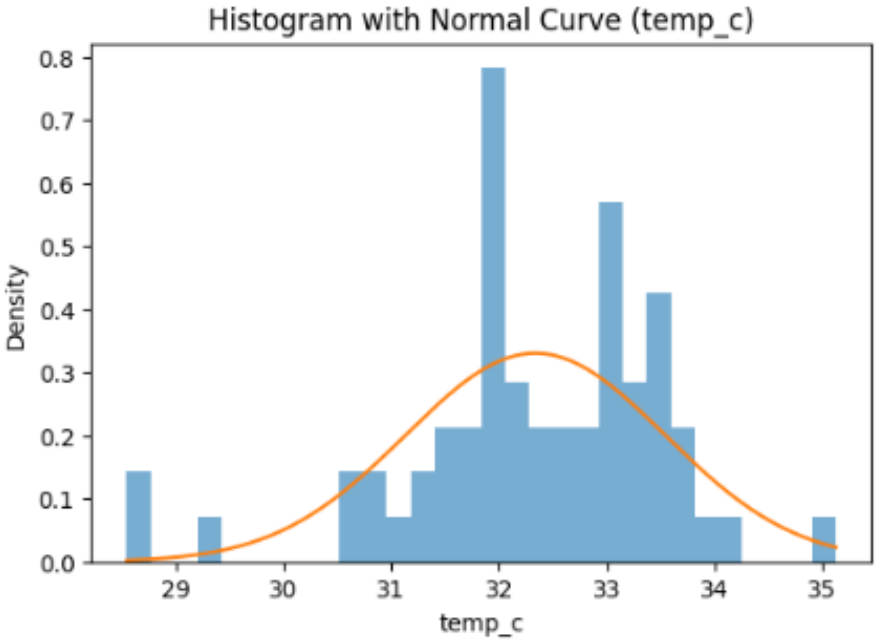


Figure 59: Strength of Relationship

7.7 Part G : Reflection

The regression line follows the data points reasonably. R^2 supports the visual fit. This model can help predict humidity from temperature.

This milestone reinforced my understanding of the relationship between two numerical variables. Calculating regression parameters manually helped me grasp the mechanics of the least squares method, rather than relying on libraries. The regression line fits

the data well, and the computed correlation and R^2 confirm the strength of the linear relationship. Overall, this exercise highlights how temperature can be used to reasonably predict humidity and deepened my appreciation for the foundational concepts behind regression analysis.

8 Final Conclusion

This project studied the relationship between weather and air quality in Dhaka using real data. Across all milestones, I applied statistics and probability step by step. The work moved from data selection to sampling, distributions, descriptive statistics, probability, and finally regression and correlation. Each milestone built on the previous one and improved my analytical understanding.

The early milestones focused on understanding the dataset. Different sampling methods showed how estimates can change. Stratified sampling gave results closest to the population values. Graphs and frequency tables revealed key patterns such as high temperatures, moderate humidity, strong wind speed, and no rainfall. These summaries made Dhaka's weather conditions clear.

Descriptive statistics explained central tendency and variation. Confidence intervals and hypothesis tests showed that sample means were reliable. Probability milestones treated weather variables as events. Empirical probability, conditional reasoning, and Bayes' Rule showed how uncertainty and dependence appear in real data. The Normal distribution analysis showed that temperature behaves like a continuous variable and can be modeled statistically.

The final milestone used regression and correlation to study the relationship between temperature and humidity. Manual calculations helped me understand how these measures work. The results showed a meaningful linear relationship and supported prediction based on data.

Some challenges appeared during the project. Choosing class intervals for histograms needed adjustment. Variables with no variation required different handling. LaTeX formatting issues also needed fixes. Solving these problems improved my technical skills.

Overall, this project strengthened my practical understanding of statistics and probability. It showed how theory connects directly to real data. This work provides a strong base for future study in data analysis and applied research.

References

@misckaggle2025, author = Kaggle, title = Dhaka Daily Air Quality and Weather Dataset, howpublished = `https://www.kaggle.com/datasets/albab12/dhaka-daily-air-quality-and-weather-dataset`, note = Accessed: 2025-12-19

@bookmontgomery2015, author = Montgomery, D.C. and Runger, G.C. and Hubele, N.F., title = Engineering Statistics, edition = 5th, year = 2015, publisher = John Wiley & Sons

@bookwalpole2012, author = Walpole, R.E. and Myers, R.H. and Myers, S.L. and Ye, K., title = Probability and Statistics for Engineers and Scientists, edition = 9th, year = 2012, publisher = Pearson

@bookfreedman2007, author = Freedman, D. and Pisani, R. and Purves, R., title = Statistics, edition = 4th, year = 2007, publisher = W. W. Norton & Company

@bookross2014, author = Ross, S.M., title = Introduction to Probability and Statistics for Engineers and Scientists, edition = 5th, year = 2014, publisher = Academic Press

@misclaerd2025, author = Laerd Statistics, title = Descriptive Statistics, Probability, and Regression Analysis, howpublished = `https://statistics.laerd.com`, note = Accessed: 2025-12-19

@misculab2025, author = University of Liberal Arts Bangladesh, Department of Computer Science and Engineering, title = STA 2101: Statistics & Probability Lecture Notes, year = 2025, note = Internal Course Material