MATH3670 Project Proposal
February 3, 2025

Nicholas Stone
903887158
MATH-3670-N

I. Dataset #1: Scatterplot

I.a. Data and Description

The restraints for Dataset #1 are $2000 > n > 100$. The provided Dataset contains 193 entries, as can be assessed by running the following in **dataset1code.py**:

```
print(f"Total number of entries (rows) in the dataset: {len(df_avg_iq)}")
```

Or by checking the total number of rows in the dataset, which can be found here as a .txt file in my Github repository for this project. Please utilize the earlier link to access the full dataset. A snippet of the data can also be found through Reference 8 in the Appendix.

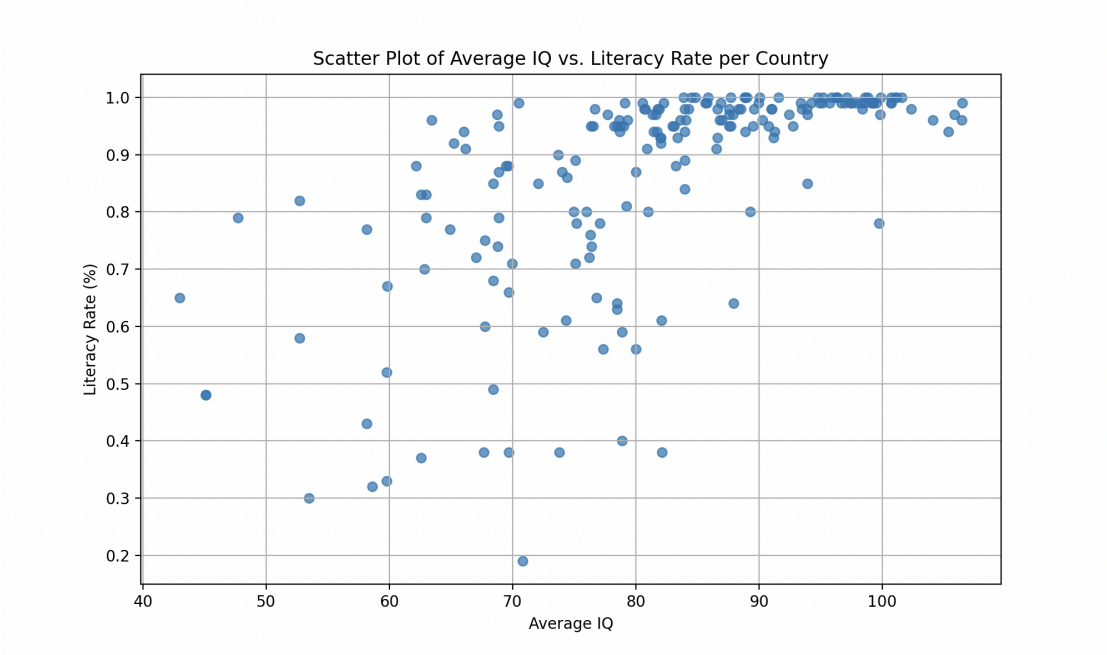
I.b. Data Source

The source to this data can be found here. The data can be downloaded locally by running **dataset1.py** in your terminal. It will provide you with a file path, which is relative to your computer and must be utilized in **dataset1code.py** or else it will not recognize the file path.

Average IQ will be X_i, where Literacy Rate will be Y_i. As you can see throughout the scatter plot in I.c, As a country's average IQ increases, the country's literacy rate tends to as well, indicating a general upward trajectory up to a literacy rate of 1.0, which flattens around an average IQ of 70, where most countries with IQs of 70 or above tend to maintain high literacy rates from ~.9 to 1.0. The data points which are below an IQ of 75 are much more sparse in comparison to those countries which have higher average IQs of 75.

I.c. Scatter Plot of Average IQ vs. Literacy Rate per Country

Provided on the next page.



II. Dataset #2: Histogram

II.a. Data and Description

The restraints for Dataset #2 are $5000 > n > 100$. The provided Dataset contains 270 entries, as can be assessed by checking the total number of rows in the dataset, which can be found here as a .txt file in my Github repository for this project. Please utilize the earlier link to access the full dataset. A snippet of the data can also be found through Reference 9 in the Appendix

The source to this data can be found here. The data can be downloaded locally by running **dataset2.py** in your terminal. It will provide you with a file path, which is relative to your computer and must be utilized in **dataset2code.py** or else it will not recognize the file path.

What you will observe is that the x-axis represents Age, while the y-axis represents the frequency of heart disease. What you will notice is that k = 10 bins were used, effectively showcasing the frequency of heart disease amongst decades of the people who were assessed. The graph showcases a large concentration of heart disease being reported towards late 50's and early 60's, while it tapers off towards younger ages and even drops off towards older ages. The initial frequency histogram shows that the height of cases per age group is approaching 60, and the relative frequency histogram helps put that into perspective by denoting that as a little over .200.

II.b. Sample Mean/Variance for the data set above:

Sample Mean

The following formula for Sample Mean was used:

$$\bar{x} = \frac{\sum x_i}{n}$$

Its code implementation can be seen here. The calculated Sample Mean for this dataset was *54.43333333333333*. You can run this python file by cd'ing into **project_part1/calculations/** and running **python3 calculate_mean.py**.

Sample Variance

The following formula for Sample Variance was used:

$$s^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$$

Its code implementation can be seen here. The calculated Sample Variance for this dataset was *82.97509293680297*. You can run this python file by cd'ing into **project_part1/calculations/** and running **python3 calculate_variance.py**.

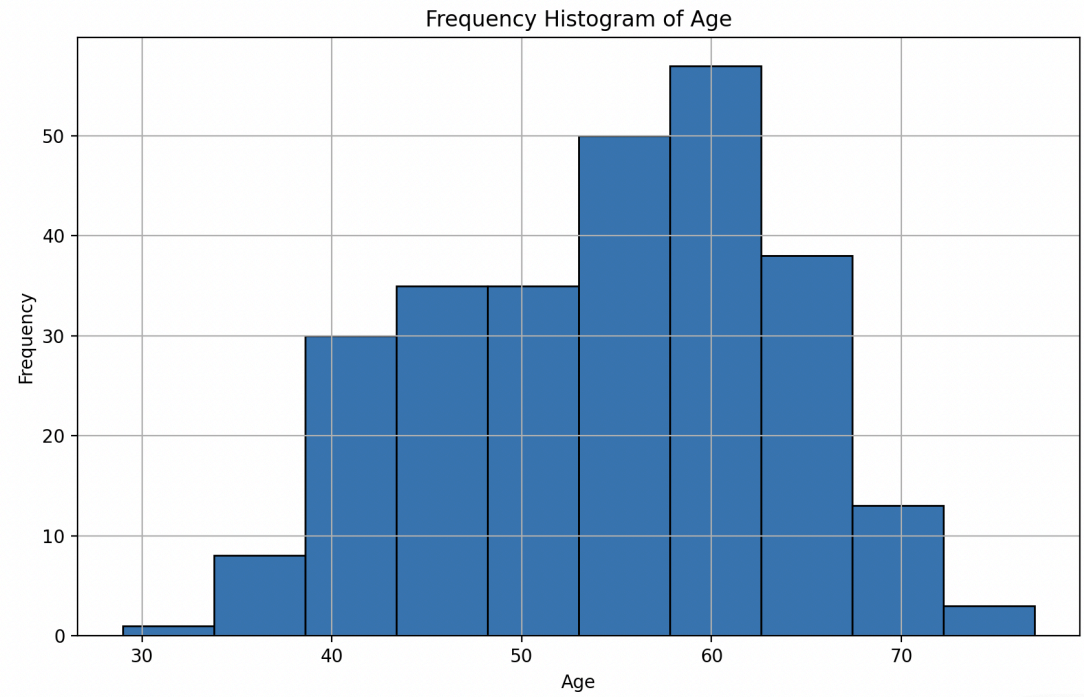
II.c. Quartiles

The following three values separate the four quartiles of the data.

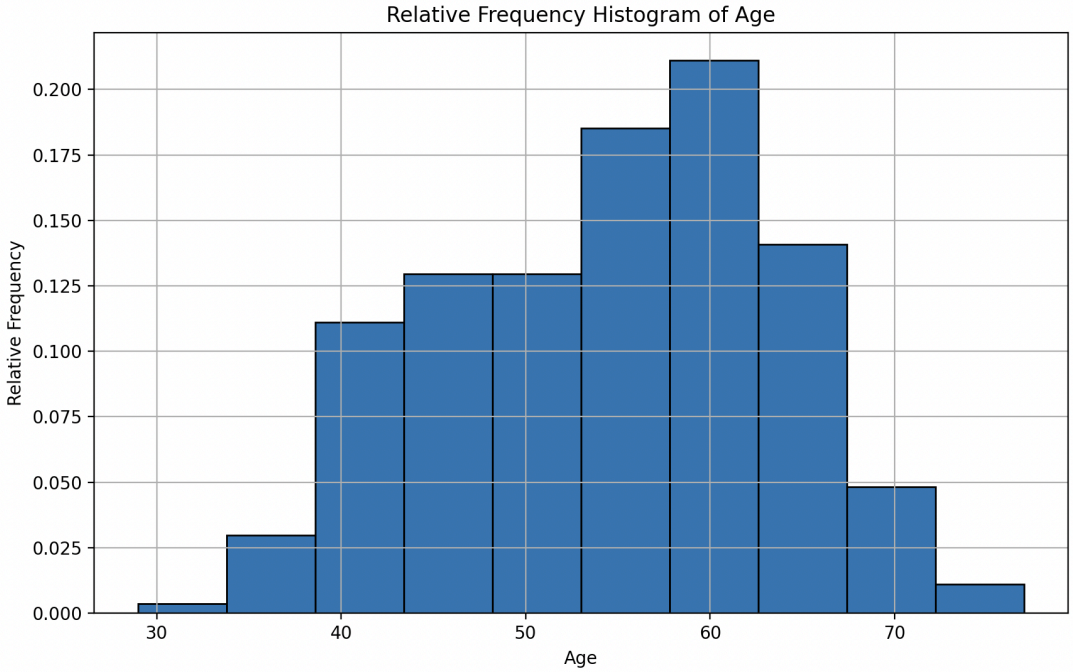
Q1: 48.0
Q2: 55.0
Q3: 61.0

Its code implementation can be seen here. You can run this python file by cd'ing into **project_part1/calculations/** and running **python3 calculate_quartiles.py**.

II.d. Frequency Histogram



II.e. Relative Frequency Histogram



C should take the value 270.0 (Logic can be observed in the code).

III. Code and Documentation

The code and associated documentation can be found here. This link to my Github repository contains the below file structure and project history for your convenience. Else, code is provided in the appendix below.

```
/project_part1
├── /dataset1
│       ├──dataset1.py
│       └──dataset1code.py
├── /dataset2
│       ├──dataset2.py
│       └──dataset2code.py
└──/misc
        ├──/calculations
        │       ├──calculate_mean.py
        │       ├──calculate_quartiles.py
        │       └──calculate_variance.py
        └──/datasets_txt
                ├──dataset1info.txt
                └──dataset2info.txt
```

Appendix

Reference 1: dataset1.py

```python
#Taken from Kaggle, run file to download data locally

import kagglehub

path = kagglehub.dataset_download("mlippo/average-global-iq-per-country-with-other-stats")

print("Path to dataset files:", path)
```

Reference 2: dataset1code.py (File path is cut off in photo but can be observed in Github repo)

```python
# Imports
import pandas as pd
import matplotlib.pyplot as plt

# File Path (relative)
file_path = "/Users/nicholasstone/.cache/kagglehub/datasets/mlippo/average-global-iq-per-country-with-other-stats/versions/3/avgIQp

# Interpreting data as Pandas dataframe
df_avg_iq = pd.read_csv(file_path)

# Looking to see if Average IQ and Literacy Rates occur in the data
if 'Average IQ' in df_avg_iq.columns and 'Literacy Rate' in df_avg_iq.columns:

    # Creates the scatter plot
    plt.figure(figsize=(10, 6))
    plt.scatter(df_avg_iq['Average IQ'], df_avg_iq['Literacy Rate'], alpha=0.7)

    # Labels and Title
    # Label for x-axis
    plt.xlabel('Average IQ')
    # Label for y-axis
    plt.ylabel('Literacy Rate (%)')
    # Title
    plt.title('Scatter Plot of Average IQ vs. Literacy Rate per Country')

    # Background grid initialized for visual purposes
    plt.grid(True)

    # Initialize the scatterplot in a separate frame
    plt.show()
else:
    #'Error' statement
    print("Ensure the dataset has 'Average IQ' and 'Literacy Rate' columns.")
```

Reference 3: dataset2.py

```
#Taken from Kaggle, run file to download data locally

import kagglehub

path = kagglehub.dataset_download("luvharishkhati/heart-disease-patients-details")

print("Path to dataset files:", path)

heart_disease.csv
```

Reference 4: dataset2code.py

```
#Imports
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# File Path (relative)
file_path = "/Users/nicholasstone/.cache/kagglehub/datasets/luvharishkhati/heart-disease-patients-details/versions/1/heart_disease.csv"

# Interprets data as Pandas dataframe
df_heart_disease = pd.read_csv(file_path)

# Looks for Age column in data
if 'age' in df_heart_disease.columns:
    data = df_heart_disease['age'].dropna()

    # Defines k bins
    k = 10  # Can be adjusted, but 10 is ideal number in my opinion, essentially separates by decades,
            # which makes sense in the context of heart disease

    # Creating the Frequency Histogram
    plt.figure(figsize=(10, 6))
    counts, bins, patches = plt.hist(data, bins=k, edgecolor='black')

    # Title initialized
    plt.title('Frequency Histogram of Age')

    # X-axis titled for Age
    plt.xlabel('Age')

    # Y-axis titled for Frequency
    plt.ylabel('Frequency')

    # Initialized grid for visual pruposes
    plt.grid(True)

    # Initialized the histogram in a separate frame
    plt.show()

    # Defining C variable
    C = counts.sum()  # Total number of data points
    relative_frequencies = counts / C

    # Creating the Relative Frequency Histogram
    plt.figure(figsize=(10, 6))
    plt.bar(bins[:-1], relative_frequencies, width=np.diff(bins), edgecolor='black', align='edge')

    # Title initialized
    plt.title('Relative Frequency Histogram of Age')

    # X-axis titled for age
    plt.xlabel('Age')

    # Y-axis titled for relative frequency
    plt.ylabel('Relative Frequency')

    # Initialized grid for visual purposes
    plt.grid(True)
```

```
    # Initialized the relative histogram in a separate frame, keep in mind that the first histogram must be x'd
    #out of to see this second histogram
    plt.show()

    # Display C, must have x'd out of both histograms to see in terinal (i.e. the program has terminated)
    print(f"Value of C (normalizing constant): {C}")
else:
    #'Error' statement
    print("Age not present in data.")
```

Reference 5: calculate_mean.py

```
# Imports
import pandas as pd

#Function
def calculate_mean(data):

    #Return statement for sample mean
    return sum(data) / len(data) if data else None

#Main code execution
if __name__ == "__main__":

    #File path (relative)
    file_path = "/Users/nicholasstone/.cache/kagglehub/datasets/luvharishkhati/heart-disease-patients-details/versions/1/heart_disease.csv"

    #Reads to Pandas datafram
    df = pd.read_csv(file_path)

    #Looking for age in the dataframe
    if "age" in df.columns:

        #Converts age to a list
        data = df["age"].dropna().tolist()

        #Calculates sample mean by referring to python file
        sample_mean_value = calculate_mean(data)

        #Prints to terminal
        print(f"Sample Mean is {sample_mean_value}")
    else:
```

```
        #'Error' message
        print("Age not present in data.")

```

Reference 6: calculate_quartiles.py

```python
#Imports
import pandas as pd

#Function to calculate the quartiles
def calculate_quartiles(data):

    #Listing the quartiles and calculating them
    quartiles = {
        #1st quartile
        "25 percentile": data.quantile(0.25),

        #2nd quartile
        "50 percentile": data.quantile(0.5),

        #3rd quartile
        "75 percentile": data.quantile(0.75)
    }

    #Return statement
    return quartiles

#Main code execution
if __name__ == "__main__":

    #File Path (relative)
    file_path = "/Users/nicholasstone/.cache/kagglehub/datasets/luvharishkhati/heart-disease-patients-details/versions/1/heart_disease.csv"

    #Uses Pandas to read dataframe
    df = pd.read_csv(file_path)
```

```python
    #Checking for age data
    if "age" in df.columns:
        data = df["age"].dropna()
        quartile_values = calculate_quartiles(data)
        print("Quartile Values:")

        #Individually prints quartiles to terminal
        for quartiled, num in quartile_values.items():
            print(f"{quartiled}: {num}")
    else:
        #'Error' statement
        print("Age not present in data.")
```

Reference 7: calculate_variance.py

```python
#Imports
import pandas as pd

#Function to calculate variance
def calculate_variance(data):
    n = len(data)
    if n < 2:
        return None

    #Sample variance calculation (See pdf for rationale)
    mean_value = sum(data) / n
    variance = sum((x - mean_value) ** 2 for x in data) / (n - 1)
    return variance

#Main code execution
if __name__ == "__main__":

    #File Path (relative)
    file_path = "/Users/nicholasstone/.cache/kagglehub/datasets/luvharishkhati/heart-disease-patients-details/versions/1/heart_disease.csv"

    #Reads as a Pandas dataframe
    df = pd.read_csv(file_path)

    #Checks to see if age data exists
    if "age" in df.columns:
        data = df["age"].dropna().tolist()
        variance_value = calculate_variance(data)

        #Prints out sample variance final number
        print(f"Sample Variance: {variance_value}")
    else:
        #'Error' statement
        print("Age not present in data.")
```

Reference 8: Example of dataset1info.txt

```
Data in Rows and Columns:
 Rank                        Country  Average IQ       Continent  Literacy Rate  Nobel Prices  HDI (2021)  Mean years of schooling - 2021
    1                          Japan      106.48            Asia           0.99            29       0.925                            13.4
    2                         Taiwan      106.47            Asia           0.96             4         NaN                             NaN
    3                      Singapore      105.89            Asia           0.97             0       0.939                            11.9
    4                      Hong Kong      105.37            Asia           0.94             1       0.952                            12.2
    5                          China      104.10            Asia           0.96             8       0.768                             7.6
    6                    South Korea      102.35            Asia           0.98             0       0.925                            12.5
    7                        Belarus      101.60          Europe           1.00             2       0.808                            12.1
    8                        Finland      101.20          Europe           1.00             5       0.940                            12.9
    9                  Liechtenstein      101.07          Europe           1.00             0       0.935                            12.5
   10                        Germany      100.74          Europe           0.99           111       0.942                            14.1
   11                    Netherlands      100.74          Europe           0.99            22       0.941                            12.6
   12                        Estonia      100.72          Europe           1.00             0       0.890                            13.5
   13                     Luxembourg       99.87          Europe           1.00             2       0.930                            13.0
   14                          Macao       99.82            Asia           0.97             0         NaN                             NaN
   15                       Cambodia       99.75            Asia           0.78             0       0.593                             5.1
   16                         Canada       99.52   North America           0.99            28       0.936                            13.8
   17                      Australia       99.24         Oceania           0.99            12       0.951                            12.7
   18                        Hungary       99.24          Europe           0.99            13       0.846                            12.2
   19                    Switzerland       99.24          Europe           0.99            27       0.962                            13.9
   20                 United Kingdom       99.12          Europe           0.99           137       0.929                            13.4
   21                   North Korea        98.82            Asia           1.00             0         NaN                             NaN
   22                       Slovenia       98.60          Europe           1.00             1       0.918                            12.8
   23                    New Zealand       98.57         Oceania           0.99             3       0.937                            12.9
   24                        Austria       98.38          Europe           0.98            22       0.916                            12.3
   25                        Iceland       98.26          Europe           0.99             1       0.959                            13.8
   26                        Denmark       97.83          Europe           0.99            13       0.948                            13.0
   27                        Belgium       97.49          Europe           0.99            11       0.937                            12.4
   28                  United States       97.43   North America           0.99           400       0.921                            13.7
```

Reference 9: Example of dataset2info.txt

```
age sex chest   resting_blood_pressure  serum_cholestoral   fasting_blood_sugar resting_electrocardiographic_results    maximum_heart_rate_
70  1   4   130 322 0   2   109 0   2.4 2   3   3   1
67  0   3   115 564 0   2   160 0   1.6 2   0   7   0
57  1   2   124 261 0   0   141 0   0.3 1   0   7   1
64  1   4   128 263 0   0   105 1   0.2 2   1   7   0
74  0   2   120 269 0   2   121 1   0.2 1   1   3   0
65  1   4   120 177 0   0   140 0   0.4 1   0   7   0
56  1   3   130 256 1   2   142 1   0.6 2   1   6   1
59  1   4   110 239 0   2   142 1   1.2 2   1   7   1
60  1   4   140 293 0   2   170 0   1.2 2   2   7   1
63  0   4   150 407 0   2   154 0   4.0 2   3   7   1
59  1   4   135 234 0   0   161 0   0.5 2   0   7   0
53  1   4   142 226 0   2   111 1   0.0 1   0   7   0
44  1   3   140 235 0   2   180 0   0.0 1   0   3   0
61  1   1   134 234 0   0   145 0   2.6 2   2   3   1
57  0   4   128 303 0   2   159 0   0.0 1   1   3   0
71  0   4   112 149 0   0   125 0   1.6 2   0   3   0
46  1   4   140 311 0   0   120 1   1.8 2   2   7   1
53  1   4   140 203 1   2   155 1   3.1 3   0   7   1
64  1   1   110 211 0   2   144 1   1.8 2   0   3   0
40  1   1   140 199 0   0   178 1   1.4 1   0   7   0
67  1   4   120 229 0   2   129 1   2.6 2   2   7   1
48  1   2   130 245 0   2   180 0   0.2 2   0   3   0
43  1   4   115 303 0   0   181 0   1.2 2   0   3   0
47  1   4   112 204 0   0   143 0   0.1 1   0   3   0
54  0   2   132 288 1   2   159 1   0.0 1   1   3   0
48  0   3   130 275 0   0   139 0   0.2 1   0   3   0
46  0   4   138 243 0   2   152 1   0.0 2   0   3   0
51  0   3   120 295 0   2   157 0   0.6 1   0   3   0
58  1   3   112 230 0   2   165 0   2.5 2   1   7   1
```