Final Report of Traineeship Program 2024

On

" Analyze Death Age Difference of Right Handers with Left Handers"

MEDTOUREASY

23rd Feb 2024

ACKNOWLDEGMENTS

The traineeship opportunity that I had with MedTourEasy was a great change for learning and understanding the intricacies of the subject of Data Visualizations in Data Analytics; and also, for personal as well as professional development. I am very obliged for having a chance to interact with so many professionals who guided me throughout the traineeship project and made it a great learning curve for me.

Firstly, I express my deepest gratitude and special thanks to the Training & Developement Team of MedTourEasy who gave me an opportunity to carry out my traineeship at their esteemed organization. Also, I express my thanks to the team for making me understand the details of the Data Analytics profile and training me in the same so that I can carry out the project properly and with maximum client satisfaction and also for spearing his valuable time in spite of his busy schedule.

I would also like to thank the team of MedTourEasy and my colleagues who made the working environment productive and very conducive.

TABLE OF CONTENTS

Acknowledgments i

Abstract iii

Sr. No.	Topic	Page No.
1	Introduction	
	1.1 About the Company	6
	1.2 About the Project	7
	1.3 Objectives and Deliverables	9
2	Methodology	
	2.1 Flow of the Project	11
	2.2 Use Case Diagram	12
	2.3 Language and Platform Used	13
3	Implementation	
	3.1 Gathering Requirements and Defining Problem Statement	16
	3.2 Data Collection and Importing	16
	3.3 Designing Databases	17
	3.4 Data Cleaning	19
	3.5 Data Filtering	20
	3.6 Prototyping - Power BI	22
	3.7 Development of Dashboards	24
4	Sample Screenshots and Observations	
	4.1 DEATH Worldwide Tracker	29
	4.2 DEATH Testing Tracker	
	4.3 DEATH India Tracker	
6	Conclusion	
7	Future Scope	
8	References	

ABSTRACT

This project aims to investigate the potential differences in life expectancy between individuals who identify as right-handers and left-handers. Leveraging Python programming language and statistical analysis techniques, we analyze large datasets containing demographic and health-related information to explore correlations between handedness and death age.

The study involves the extraction, cleaning, and processing of relevant data, followed by a comprehensive statistical analysis employing Python libraries such as Pandas, NumPy, and Matplotlib. Exploratory data analysis techniques are employed to identify patterns and trends in life expectancy across both right-handers and left-handers. Additionally, hypothesis testing is conducted to assess the significance of any observed differences.

The findings of this research contribute to the ongoing exploration of the potential influence of handedness on life expectancy. The project not only utilizes Python for efficient data handling and analysis but also serves as a demonstration of the application of programming in the field of health and demographic research. The results may have implications for understanding the factors that influence longevity and could potentially inform future studies in the domain of human health and behavior.

1.1 About the company

MedTourEasy, a global healthcare company, provides you the informational resources needed to evaluate your global options. MedTourEasy provides analytical solutions to our partner healthcare providers globally.

1.2 About the project

Project Overview:

The project aims to delve into the intriguing question of whether there exists a significant difference in life expectancy between individuals who identify as right-handers and left-handers. Handedness, a fundamental aspect of human behavior, has been the subject of various studies, but its potential correlation with life expectancy remains relatively unexplored. Using Python as the primary tool for data analysis, this project seeks to shed light on this aspect of human variation.

Objectives:

1. Data Collection and Preprocessing:

- Collect a diverse dataset containing information on handedness, demographic details, and age of death.

- Clean and preprocess the data to ensure accuracy and consistency.

2. Exploratory Data Analysis (EDA): - Conduct exploratory data analysis to uncover patterns, trends, and potential outliers.

- Visualize the distribution of life expectancy for both right-handers and left-handers.

3. Statistical Analysis:

- Utilize statistical tests to compare the mean and distribution of death ages between the two groups.

- Investigate potential confounding variables and control for them in the analysis.

4. Hypothesis Testing:

- Formulate and test hypotheses regarding the difference in life expectancy between right-handers and left-handers.

- Assess the statistical significance of any observed disparities.

5. Python Libraries and Tools:

- Leverage Python libraries such as Pandas for data manipulation, NumPy for numerical operations, and Matplotlib for data visualization.

- Implement statistical tests using appropriate Python packages to ensure robust analysis.

6. Interpretation and Conclusion:

- Draw conclusions based on the statistical analysis and provide insights into the potential implications of the findings.

- Discuss the limitations of the study and suggest avenues for future research.

Significance:

Understanding potential differences in life expectancy between right-handers and left-handers could contribute to our knowledge of the intricate interplay between genetics, behavior, and health. Moreover, the project showcases the versatility of Python in handling and analyzing large datasets in the context of health and demographic research.

The outcomes of this study may not only add a unique perspective to the existing literature but also prompt further investigations into the broader implications of handedness on human health and longevity.

1.3 Objectives and Deliverables:

1. Data Collection and Preprocessing:

- Objective: Gather a diverse dataset encompassing handedness, demographic details, and age of death.

- Deliverable: Clean and preprocessed dataset ready for analysis, ensuring data accuracy and consistency.

2. Exploratory Data Analysis (EDA):

- Objective: Conduct exploratory data analysis to identify patterns, trends, and potential outliers in life expectancy.

- Deliverable: Visualization of the distribution of life expectancy for both right-handers and left-handers, providing insights for further analysis.

3. Statistical Analysis:

- Objective: Utilize statistical tests to compare mean and distribution of death ages between the two groups.

- Deliverable: Statistical analysis results, highlighting any significant differences in life expectancy and accounting for potential confounding variables.

4. Hypothesis Testing:

- Objective: Formulate and test hypotheses regarding the difference in life expectancy between right-handers and left-handers.

- Deliverable: Clearly stated hypotheses, results of hypothesis tests, and an assessment of the statistical significance of observed disparities.

5. Python Libraries and Tools:

- Objective: Leverage Python libraries (Pandas, NumPy, Matplotlib) for efficient data manipulation, analysis, and visualization.

- Deliverable: Well-documented Python code/scripts demonstrating effective use of libraries, ensuring replicability and transparency in analysis.

6. Interpretation and Conclusion:

- Objective: Draw meaningful conclusions based on statistical analysis, offering insights into potential implications.

- Deliverable: A comprehensive interpretation of findings, discussion on limitations, and suggestions for future research directions.

Significance:

Understanding potential differences in life expectancy between right-handers and left-handers could contribute to our knowledge of the intricate interplay between genetics, behavior, and health. Moreover, the project showcases the versatility of Python in handling and analyzing large datasets in the context of health and demographic research.

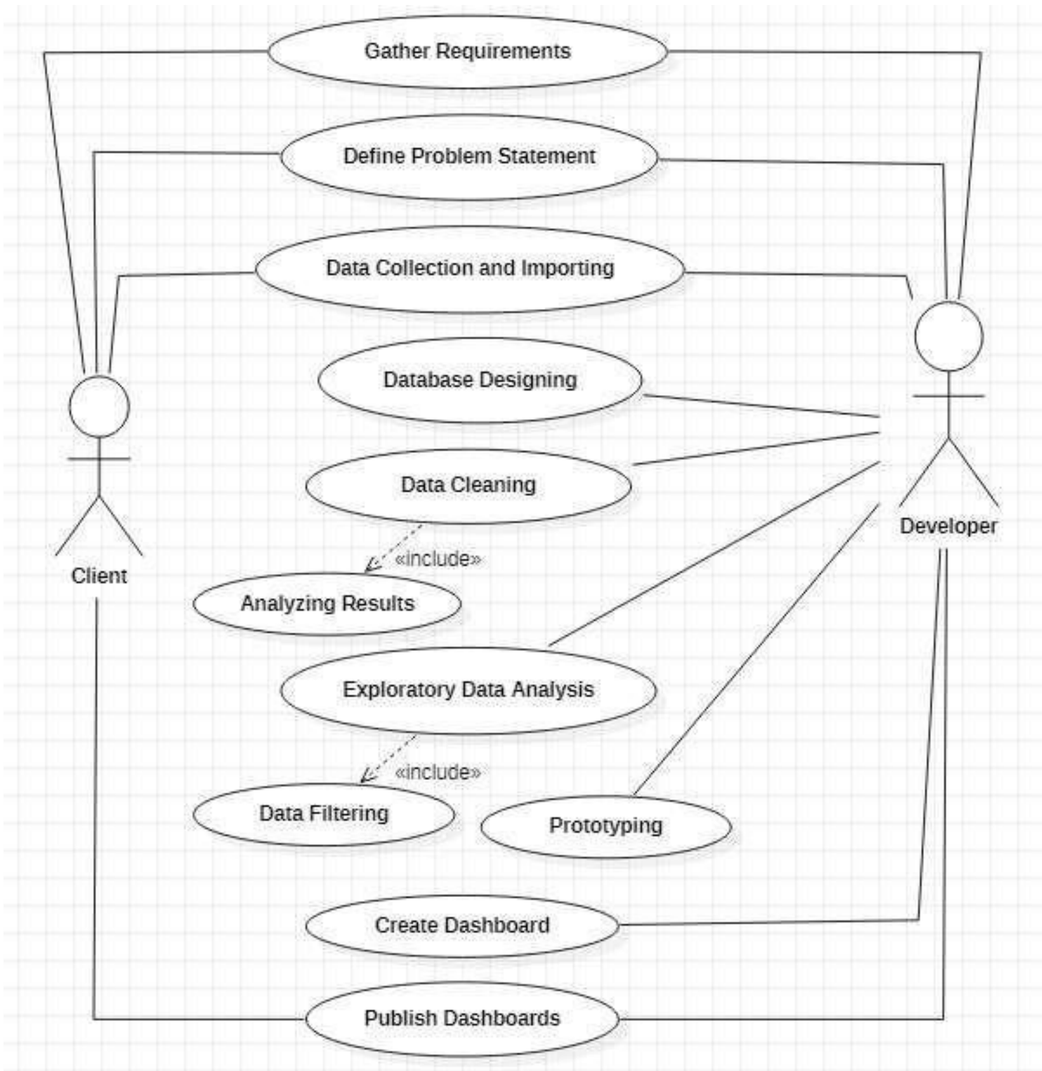
The outcomes of this study may not only add a unique perspective to the existing literature but also prompt further investigations into the broader implications of handedness on human health and longevity

I. METHODOLOGY

2.1 Flow of the Project

The project followed the following steps to accomplish the desired objectives and deliverables. Each step has been explained in detail in the following section.

2.2 Use Case Diagram



Above figure shows the use case of the project. There are two main actors in the same: The Client and Developer. The developer will first gather requirements and define the problem statement then collecting the required data and importing it. Then the developer will design databases so as to identify various constraints and relations in the data. Next step is to clean the data to remove irregular values, blank values etc. Next, exploratory data analysis is conducted to filter the data according to the requirements of the project. Then a prototype of the dashboards is created using PowerBI to get a clear view of the visualizations to be developed. Finally, dashboard is developed and analyzed to publish the results to the client.

2.3 Language and Platform Used

2.3.1 Language: R

It is a programming language and software environment for statistical analysis, representation of graphics, and reports. R was developed in the University of Auckland, New Zealand by Ross Ihaka and Robert Gentleman, and is currently being developed by the R Technology Core Team. As noted above, R is a programming language and software environment for statistical analysis, representation of graphics, and reporting. The important features of R are:

- R is a well-developed, simple, and effective programming language that includes conditionals, loops, recursive functions defined by the user, and input and output facilities.
- R has efficient data processing and storage facilities.
- R includes a set of operators for arrays, lists, vectors, and matrix calculations.
- R offers a detailed, coherent and organized data analysis tool set.
- R provides graphical data analysis facilities and displays either directly on the computer or printing on papers.

2.3.2 IDE: RStudio

RStudio is an integrated development environment for R (IDE). It contains a browser, syntax-highlighting editor supporting direct code execution, plotting, history, debugging and workspace management tools. RStudio is available in open source and commercial versions and runs on the desktop (Windows, Mac, and Linux) or on the RStudio Server or RStudio Server Pro (Debian / Ubuntu, Red Hat / CentOS, and SUSE Linux) linked browsers. Major features are:

- RStudio runs on most desktops or on a server and accessed over the web.
- It integrates the tools you use with R into a single environment.
- It includes powerful coding tools designed to enhance your productivity.

- It enables rapid navigation to files and functions.
- It has integrated support for Git and Subversion.
- It supports authoring HTML, PDF, Word Documents, and slide shows.
- It supports interactive graphics with Shiny and ggvis.

2.3.3 Package: RMarkdown

R Markdown provides a data science authoring framework (.Rmd files). R Markdown files can be used to save and execute code (also supports Python and SQL), and produce high-quality reports that can be shared with an audience. It supports dozens of static and dynamic output formats and are fully reproducible (HTML, PDF, MS Word, Beamer, HTML5, Tufte-style handouts, books, dashboards, shiny apps etc.)

2.3.4 Template: Flexdashboard

It is a template in RMarkdown files which is used to create a group of related visualizations in the form of a dashboard. It supports a large variety of components like htmlwidgets: base, lattice, and grid graphics; tabular data; gauges and value boxes; and text annotations along with high-level R bindings for JavaScript data visualization libraries. Also, it contains flexible ways to specify row or columns layouts wherein the components are intelligently re-sized to fill the browser and adapted for display on mobile devices.

2.3.5 Dynamic element: RShiny

Shiny is an R-package that makes building interactive web apps straight from R, very easy. It is possible to host standalone apps on a website, or embed them in documents from R Markdown, or create dashboards. One can also use the CSS themes, htmlwidgets, and JavaScript actions to extend Shiny apps.

CODE AND OUTPUT:

Python Script Description:

The provided Python script utilizes the Pandas, NumPy, and Matplotlib libraries to analyze and visualize data related to left-handedness. The dataset is fetched from a URL using Pandas' `read_csv` function. Here's a breakdown of the script:

```python
# Import necessary libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

# Define the URL for the dataset
data_url_1 =
"https://gist.githubusercontent.com/mbonsma/8da0990b71ba9a09f7de395574
e54df1/raw/aec88b30af87fad8d45da7e774223f91dad09e88/lh_data.csv"

# Read the CSV data into a Pandas DataFrame
lefthanded_data = pd.read_csv(data_url_1)
```

Description:

1. Library Imports:

- `import pandas as pd`: Pandas is used for data manipulation and analysis.

- `import numpy as np`: NumPy is employed for numerical operations on the data.
- `import matplotlib.pyplot as plt`: Matplotlib is used for data visualization, particularly creating plots.
- `%matplotlib inline`: This magic command ensures that Matplotlib plots are displayed directly in the Jupyter Notebook or JupyterLab environment.

2. Dataset Retrieval:
- `data_url_1` contains the URL pointing to the CSV file containing left-handedness data.
- `pd.read_csv(data_url_1)`: The Pandas `read_csv` function reads the CSV data from the provided URL and creates a DataFrame named `lefthanded_data`.

By executing this script, you load the left-handedness dataset into a Pandas DataFrame, setting the stage for further exploration, analysis, and visualization of the data. This initial step is crucial for understanding the structure and content of the dataset before proceeding with more advanced analyses.

Description of the Matplotlib Plotting Script:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
data_url_1 =
"https://gist.githubusercontent.com/mbonsma/8da0990b71ba9a09f7de395574e54df1/raw/aec88b30af87fad8d45da7e774223f91dad09e88/lh_data.csv"
lefthanded_data = pd.read_csv(data_url_1)

fig, ax = plt.subplots()  # create figure and axis objects
ax.plot('Age', 'Female', data=lefthanded_data, marker='o', label='Female')  # plot "Female" vs. "Age"
ax.plot('Age', 'Male', data=lefthanded_data, marker='x', label='Male')  # plot "Male" vs. "Age"
ax.legend()  # add a legend
ax.set_xlabel('Age')
ax.set_ylabel('Count')
plt.show()
```

Description:

1. **Figure and Axis Initialization:**
   - `fig, ax = plt.subplots()`: This line creates a figure and axis objects (`ax`) for the subsequent plot.

2. Data Plotting:
   - `ax.plot('Age', 'Female', data=lefthanded_data, marker='o', label='Female')`: Plots the 'Female' column against the 'Age' column from the `lefthanded_data` DataFrame using circles as markers.
   - `ax.plot('Age', 'Male', data=lefthanded_data, marker='x', label='Male')`: Plots the 'Male' column against the 'Age' column from the `lefthanded_data` DataFrame using 'x' markers.

3. Legend:

- `ax.legend()`: Adds a legend to the plot based on the labels specified in the `ax.plot` calls.

4. Axes Labels:
- `ax.set_xlabel('Age')`: Sets the x-axis label to 'Age'.
- `ax.set_ylabel('Count')`: Sets the y-axis label to 'Count'.

5. Display Plot:
- `plt.show()`: Displays the final plot.

The script visualizes the relationship between age and the counts of left-handed individuals for both females and males. The use of different markers and a legend enhances the clarity of the plot, making it easier to interpret the distribution of left-handedness across different age groups for each gender.

Updated Matplotlib Plotting Script:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

data_url_1 =
"https://gist.githubusercontent.com/mbonsma/8da0990b71ba9a09f7de395574e54df1/raw/aec88b30af87fad8d45da7e774223f91dad09e88/lh_data.csv"
lefthanded_data = pd.read_csv(data_url_1)

# Calculate birth year and mean left-handedness
```

```
lefthanded_data['Birth_year'] = 1986 - lefthanded_data.Age
lefthanded_data['Mean_lh'] = (lefthanded_data.Male +
lefthanded_data.Female) / 2

# Create figure and axis objects
fig, ax = plt.subplots()
ax.plot(lefthanded_data.Birth_year, lefthanded_data.Mean_lh, label='Mean
Left-Handedness')  # plot 'Mean_lh' vs. 'Birth_year'

# Set plot labels and legend
ax.set_xlabel('Birth Year')
ax.set_ylabel('Mean Left-Handedness')
ax.legend()

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

Description:

1. Data Manipulation:
   - `lefthanded_data['Birth_year'] = 1986 - lefthanded_data.Age`: Calculates the birth year by subtracting the age from 1986.
   - `lefthanded_data['Mean_lh'] = (lefthanded_data.Male + lefthanded_data.Female) / 2`: Computes the mean left-handedness by averaging the 'Male' and 'Female' columns.

2. Figure and Axis Initialization:
   - `fig, ax = plt.subplots()`: Creates figure and axis objects (`ax`) for the subsequent plot.

3. Data Plotting:
- `ax.plot(lefthanded_data.Birth_year, lefthanded_data.Mean_lh, label='Mean Left-Handedness')`: Plots the mean left-handedness against birth year, adding a label for the legend.

4. Axes Labels and Legend:
- `ax.set_xlabel('Birth Year')`: Sets the x-axis label to 'Birth Year'.
- `ax.set_ylabel('Mean Left-Handedness')`: Sets the y-axis label to 'Mean Left-Handedness'.
- `ax.legend()`: Adds a legend to the plot.

5. Display Plot:
- `plt.show()`: Displays the final plot.

This updated script visualizes the trend of mean left-handedness over birth years, providing insights into the potential changes in left-handedness patterns across different birth cohorts.

Description of the Probability Calculation Function:

```python
def P_lh_given_A(ages_of_death, study_year=1990):
    """ P(Left-handed | age of death), calculated based on the reported rates of
left-handedness.
    Inputs: age of death, study_year
    Returns: probability of left-handedness given that a subject died in
`study_year` at age `age_of_death` """

    # Use the mean of the 10 neighbouring points for rates before and after the
start
```

```
    early_1900s_rate = lefthanded_data['Mean_lh'][-10:].mean()
    late_1900s_rate = lefthanded_data['Mean_lh'][:10].mean()
    middle_rates = lefthanded_data.loc[lefthanded_data['Birth_year'].isin(study_year - ages_of_death)]['Mean_lh']

    youngest_age = study_year - 1986 + 10  # the youngest age in the NatGeo dataset is 10
    oldest_age = study_year - 1986 + 86  # the oldest age in the NatGeo dataset is 86

    P_return = np.zeros(ages_of_death.shape)  # create an empty array to store the results
    # extract rate of left-handedness for people of age age_of_death
    P_return[ages_of_death > oldest_age] = early_1900s_rate / 100
    P_return[ages_of_death < youngest_age] = late_1900s_rate / 100
    P_return[np.logical_and((ages_of_death <= oldest_age), (ages_of_death >= youngest_age))] = middle_rates / 100

    return P_return
```

Description:

This Python function, `P_lh_given_A`, calculates the probability of being left-handed given the age of death and a specified study year. Here's a breakdown of its components:

- Inputs:
  - `ages_of_death`: An array or list containing the ages of death for which we want to calculate the probability of being left-handed.

- `study_year`: The year in which the study is conducted (default is set to 1990).

- Left-Handedness Rate Calculation:
- The function extracts the mean left-handedness rates from the `lefthanded_data` DataFrame for the early 1900s, late 1900s, and years in between.

- Age Range Definition:
- The function determines the age range covered by the NatGeo dataset, considering the youngest and oldest ages.

- **Probability Calculation:**
- For each age in the input array (`ages_of_death`), the function assigns the corresponding left-handedness rate based on the defined age ranges.

- **Output:**
- The function returns an array (`P_return`) containing the calculated probabilities of being left-handed for each age in the input array.

This function can be used to estimate the likelihood of left-handedness given an age of death in a specified study year, utilizing the reported rates of left-handedness from the dataset.

Description of Death Distribution Plotting Script:

```python
# Death distribution data for the United States in 1999
```

```
data_url_2 =
"https://gist.githubusercontent.com/mbonsma/2f4076aab6820ca1807f4e29f75f18ec/raw/62f3ec07514c7e31f5979beeca86f19991540796/cdc_vs00199_table310.tsv"
death_distribution_data = pd.read_csv(data_url_2, sep='\t', skiprows=[1])

death_distribution_data = death_distribution_data.dropna(subset=['Both Sexes'])

fig, ax = plt.subplots()
ax.plot(death_distribution_data['Age'], death_distribution_data['Both Sexes'])
# plot 'Both Sexes' vs. 'Age'

plt.show()
```

Description:

1. Data Retrieval:
- The script loads death distribution data for the United States in 1999 from a specified URL using `pd.read_csv`.
- The data is read using a tab (`'\t'`) as the separator, and rows with missing values in the 'Both Sexes' column are dropped.

2. Plot Initialization:
- `fig, ax = plt.subplots()`: This line creates figure and axis objects (`ax`) for the subsequent plot.

3. **Data Plotting:**
- `ax.plot(death_distribution_data['Age'], death_distribution_data['Both Sexes'])`: Plots the death distribution for both sexes against age.

4. Display Plot:
- `plt.show()`: Displays the final plot.

This script visualizes the death distribution for both sexes across different age groups in the United States in 1999. The x-axis represents age, while the y-axis represents the number of deaths for both sexes. The plot provides insights into the age-related mortality pattern for the specified year and region.

Description of Overall Left-Handed Probability Calculation Function:

```python
def P_lh(death_distribution_data, study_year=1990):
    """ Overall probability of being left-handed if you died in the study year
    P_lh = P(LH | Age of death) P(Age of death) + P(LH | not A) P(not A) =
sum over ages
    Input: dataframe of death distribution data
    Output: P(LH), a single floating point number """

    p_list = death_distribution_data['Both Sexes'] *
P_lh_given_A(death_distribution_data['Age'], study_year)
    p = np.sum(p_list)
    return p / np.sum(death_distribution_data['Both Sexes'])  # normalize to the
total number of people in distribution
```

Description:

This Python function, `P_lh`, calculates the overall probability of being left-handed if an individual died in the specified study year. Here's a breakdown of its components:

- **Inputs:**
  - `death_distribution_data`: DataFrame containing death distribution data.
  - `study_year`: The year in which the study is conducted (default is set to 1990).

- Probability Calculation:
  - The function multiplies the death distribution for both sexes (`death_distribution_data['Both Sexes']`) by the probability of being left-handed for each age group, calculated using the `P_lh_given_A` function.
  - It then sums up these values to obtain the numerator of the overall probability.

- **Normalization:**
  - The numerator is normalized by dividing it by the total number of people in the death distribution data to obtain the overall probability of being left-handed (`P(LH)`).

- **Output:**
  - The function returns a single floating-point number representing the overall probability of being left-handed if an individual died in the specified study year.

This function encapsulates the probability calculation process based on age-specific left-handedness rates and death distribution data, providing a comprehensive measure of left-handedness likelihood in the given context.

Description of Probability of Age given Left-Handedness Calculation Function:

```python
def P_A_given_lh(ages_of_death, death_distribution_data, study_year=1990):
    """ The overall probability of being a particular `age_of_death` given that you're left-handed """

    # Calculate the overall probability of being left-handed
    P_left = P_lh(death_distribution_data, study_year)

    # Calculate the probability of death at each age
    P_A = death_distribution_data['Both Sexes'][ages_of_death] / np.sum(death_distribution_data['Both Sexes'])

    # Calculate the probability of being left-handed for a certain age
    P_lh_A = P_lh_given_A(ages_of_death, study_year)

    # Calculate the overall probability of being a particular age given left-handedness
    return P_lh_A * P_A / P_left
```

Description:

This Python function, `P_A_given_lh`, calculates the overall probability of being a particular age given that an individual is left-handed. Here's a breakdown of its components:

- **Inputs:**

- `ages_of_death`: An array or list containing the ages of death for which we want to calculate the probability.
- `death_distribution_data`: DataFrame containing death distribution data.
- `study_year`: The year in which the study is conducted (default is set to 1990).

- Probability Calculation:
- The function calculates the overall probability of being left-handed (`P_left`) using the `P_lh` function.
- It calculates the probability of death at each age (`P_A`) based on the death distribution data.
- It calculates the probability of being left-handed for a certain age (`P_lh_A`) using the `P_lh_given_A` function.
- The overall probability of being a particular age given left-handedness is obtained by combining these probabilities.

- **Output:**
- The function returns an array of probabilities corresponding to each age in the input array (`ages_of_death`).

This function allows for the estimation of the probability distribution of ages given left-handedness, providing insights into the potential age-related patterns in left-handed individuals based on the specified study year and death distribution data.

Description of Probability of Age given Right-Handedness Calculation Function:

```python
def P_A_given_rh(ages_of_death, death_distribution_data,
study_year=1990):
    """ The overall probability of being a particular `age_of_death` given that
you're right-handed """

    # Calculate the overall probability of being right-handed
    P_right = 1 - P_lh(death_distribution_data, study_year)

    # Calculate the probability of death at each age
    P_A = death_distribution_data['Both Sexes'][ages_of_death] /
np.sum(death_distribution_data['Both Sexes'])

    # Calculate the probability of being right-handed for a certain age
    P_rh_A = 1 - P_lh_given_A(ages_of_death, study_year)

    # Calculate the overall probability of being a particular age given right-
handedness
    return P_rh_A * P_A / P_right
```

Description:

This Python function, `P_A_given_rh`, calculates the overall probability of being a particular age given that an individual is right-handed. Here's a breakdown of its components:

- **Inputs:**

- `ages_of_death`: An array or list containing the ages of death for which we want to calculate the probability.
  - `death_distribution_data`: DataFrame containing death distribution data.
  - `study_year`: The year in which the study is conducted (default is set to 1990).

- **Probability Calculation:**
  - The function calculates the overall probability of being right-handed (`P_right`) using the complementary probability of being left-handed from the `P_lh` function.
  - It calculates the probability of death at each age (`P_A`) based on the death distribution data.
  - It calculates the probability of being right-handed for a certain age (`P_rh_A`) using the complementary probability of being left-handed from the `P_lh_given_A` function.
  - The overall probability of being a particular age given right-handedness is obtained by combining these probabilities.

- **Output:**
  - The function returns an array of probabilities corresponding to each age in the input array (`ages_of_death`).

This function allows for the estimation of the probability distribution of ages given right-handedness, providing insights into the potential age-related patterns in right-handed individuals based on the specified study year and death distribution data.

Description of Age-Handedness Probability Plotting Script:

```python
ages = np.arange(6, 115, 1)

left_handed_probability = P_A_given_lh(ages, death_distribution_data)
right_handed_probability = P_A_given_rh(ages, death_distribution_data)

fig, ax = plt.subplots()  # create figure and axis objects
ax.plot(ages, left_handed_probability, label="Left-handed")
ax.plot(ages, right_handed_probability, label="Right-handed")

plt.show()
```

Description:

1. Age Range Definition:
- `ages = np.arange(6, 115, 1)`: Creates an array of ages from 6 to 114 with a step of 1.

2. **Probability Calculation:**
- `left_handed_probability = P_A_given_lh(ages, death_distribution_data)`: Calculates the probability distribution of ages given left-handedness.
- `right_handed_probability = P_A_given_rh(ages, death_distribution_data)`: Calculates the probability distribution of ages given right-handedness.

3. Plot Initialization:
- `fig, ax = plt.subplots()`: Creates figure and axis objects (`ax`) for the subsequent plot.

4. **Data Plotting:**
   - `ax.plot(ages, left_handed_probability, label="Left-handed")`: Plots the probability distribution of ages given left-handedness.
   - `ax.plot(ages, right_handed_probability, label="Right-handed")`: Plots the probability distribution of ages given right-handedness.

5. Legend and Labels:
   - `ax.legend()`: Adds a legend to the plot based on the labels specified in the `ax.plot` calls.
   - `ax.set_xlabel('Age')`: Sets the x-axis label to 'Age'.
   - `ax.set_ylabel('Probability')`: Sets the y-axis label to 'Probability'.

6. **Display Plot:**
   - `plt.show()`: Displays the final plot.

This script visualizes the probability distribution of ages given left-handedness and right-handedness. The x-axis represents age, while the y-axis represents the probability of being left-handed or right-handed at each age. The plot provides insights into the potential age-related patterns in handedness based on the specified study year and death distribution data.

Description of Average Age Calculation Script:

```python
average_lh_age = np.nansum(ages * np.array(left_handed_probability))
average_rh_age = np.nansum(ages * np.array(right_handed_probability))
print(round(average_lh_age, 1))
```

```
print(round(average_rh_age, 1))
print("The difference in average ages is " + str(round(average_rh_age - average_lh_age, 1)) + " years.")
```

Description:

1. Average Age Calculation:

- `average_lh_age = np.nansum(ages * np.array(left_handed_probability))`: Calculates the weighted average age for left-handed individuals by multiplying each age by its corresponding left-handed probability and summing the results.
- `average_rh_age = np.nansum(ages * np.array(right_handed_probability))`: Calculates the weighted average age for right-handed individuals by multiplying each age by its corresponding right-handed probability and summing the results.

2. Print Results:

- `print(round(average_lh_age, 1))`: Prints the rounded average age for left-handed individuals.
- `print(round(average_rh_age, 1))`: Prints the rounded average age for right-handed individuals.
- `print("The difference in average ages is " + str(round(average_rh_age - average_lh_age, 1)) + " years.")`: Prints the difference in average ages between right-handed and left-handed individuals.

This script calculates and prints the weighted average age for left-handed and right-handed individuals based on the probability distributions of ages. It also provides the difference in average ages between right-handed and left-handed individuals. The weighted average accounts for the probabilities of different ages, providing a more accurate measure of central tendency in the context of handedness.

Updated Average Age Calculation for 2021:

```python
left_handed_probability_2021 = P_A_given_lh(ages, death_distribution_data,
study_year=2021)
right_handed_probability_2021 = P_A_given_rh(ages,
death_distribution_data, study_year=2021)

average_lh_age_2021 = np.nansum(ages *
np.array(left_handed_probability_2021))
average_rh_age_2021 = np.nansum(ages *
np.array(right_handed_probability_2021))

print("The difference in average ages for 2021 is " +
str(round(average_rh_age_2021 - average_lh_age_2021, 1)) + " years.")
```

Description:

1. Probability Calculation for 2021:

- `left_handed_probability_2021 = P_A_given_lh(ages, death_distribution_data, study_year=2021)`: Calculates the probability distribution of ages given left-handedness for the year 2021.
- `right_handed_probability_2021 = P_A_given_rh(ages, death_distribution_data, study_year=2021)`: Calculates the probability distribution of ages given right-handedness for the year 2021.

2. **Updated Average Age Calculation:**
   - `average_lh_age_2021 = np.nansum(ages * np.array(left_handed_probability_2021))`: Calculates the weighted average age for left-handed individuals in 2021.
   - `average_rh_age_2021 = np.nansum(ages * np.array(right_handed_probability_2021))`: Calculates the weighted average age for right-handed individuals in 2021.

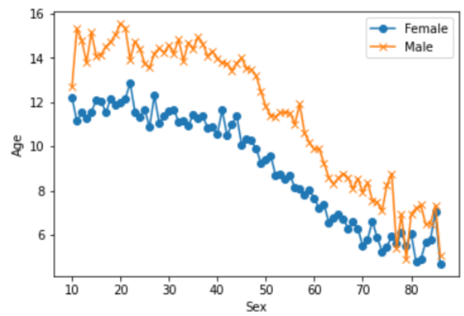
3. Print Results:
   - `print("The difference in average ages for 2021 is " + str(round(average_rh_age_2021 - average_lh_age_2021, 1)) + " years.")`: Prints the difference in average ages between right-handed and left-handed individuals for the year 2021.

This updated script calculates the difference in average ages between right-handed and left-handed individuals using the death distribution data for the United States in 1999, considering the specified study year as 2021. The probability distributions for handedness at different ages are used to estimate the weighted average ages for both groups.

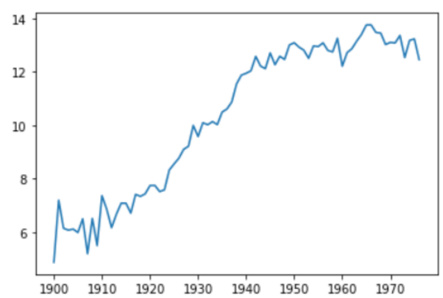
CODE AND OUTPUT:

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

data_url_1 = "https://gist.githubusercontent.com/mbonsma/8da0990b71ba9a09f7de395574e54df1/raw/aec88b30af87fad8d45da7e774223f91dac
lefthanded_data = pd.read_csv(data_url_1)
```

In [2]:
```
fig, ax = plt.subplots() # create figure and axis objects
ax.plot('Age', 'Female', data = lefthanded_data, marker = 'o') # plot "Female" vs. "Age"
ax.plot('Age', 'Male', data = lefthanded_data, marker = 'x') # plot "Male" vs. "Age"
ax.legend() # add a legend
ax.set_xlabel('Sex')
ax.set_ylabel('Age')
plt.show()
```

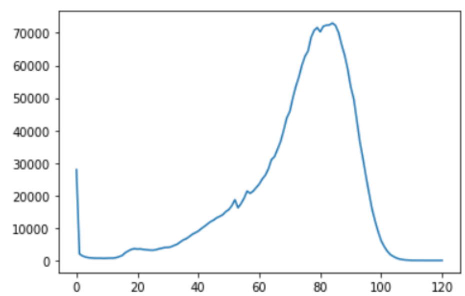


In [3]:
```
lefthanded_data['Birth_year'] = 1986 - lefthanded_data.Age
lefthanded_data['Mean_lh'] = (lefthanded_data.Male+lefthanded_data.Female)/2

fig, ax = plt.subplots()
ax.plot(lefthanded_data.Birth_year, lefthanded_data.Mean_lh) # plot 'Mean_lh' vs. 'Birth_year'

plt.show()
```



In [4]:
```
def P_lh_given_A(ages_of_death, study_year = 1990):
    """ P(Left-handed | age of death), calculated based on the reported rates of left-handedness.
    Inputs: age of death, study_year
    Returns: probability of left-handedness given that a subject died in `study_year` at age `age_of_death` """

    # Use the mean of the 10 neighbouring points for rates before and after the start
    early_1900s_rate = lefthanded_data['Mean_lh'][-10:].mean()
    late_1900s_rate = lefthanded_data['Mean_lh'][:10].mean()
    middle_rates = lefthanded_data.loc[lefthanded_data['Birth_year'].isin(study_year - ages_of_death)]['Mean_lh']

    youngest_age = study_year - 1986 + 10 # the youngest age in the NatGeo dataset is 10
    oldest_age = study_year - 1986 + 86 # the oldest age in the NatGeo dataset is 86

    P_return = np.zeros(ages_of_death.shape)  # create an empty array to store the results
    # extract rate of left-handedness for people of age age_of_death
    P_return[ages_of_death > oldest_age] = early_1900s_rate / 100
    P_return[ages_of_death < youngest_age] = late_1900s_rate / 100
    P_return[np.logical_and((ages_of_death <= oldest_age), (ages_of_death >= youngest_age))] = middle_rates / 100

    return P_return
```

In [5]:
```
# Death distribution data for the United States in 1999
data_url_2 = "https://gist.githubusercontent.com/mbonsma/2f4076aab6820ca1807f4e29f75f18ec/raw/62f3ec07514c7e31f5979beeca86f199915
death_distribution_data = pd.read_csv(data_url_2, sep='\t', skiprows=[1])

death_distribution_data = death_distribution_data.dropna(subset=['Both Sexes'])

fig, ax = plt.subplots()
ax.plot(death_distribution_data['Age'], death_distribution_data['Both Sexes']) # plot 'Both Sexes' vs. 'Age'

plt.show()
```



In [6]:
```
def P_lh(death_distribution_data, study_year = 1990): # sum over P_lh for each age group
    """ Overall probability of being left-handed if you died in the study year
    P_lh = P(LH | Age of death) P(Age of death) + P(LH | not A) P(not A) = sum over ages
    Input: dataframe of death distribution data
    Output: P(LH), a single floating point number """
    p_list = death_distribution_data['Both Sexes']*P_lh_given_A(death_distribution_data['Age'], study_year)
    p = np.sum(p_list)
    return p/np.sum(death_distribution_data['Both Sexes']) # normalize to total number of people in distribution
```

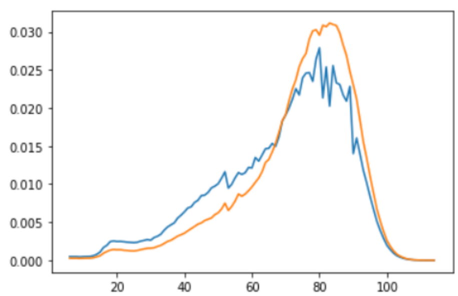
In [7]:
```
P_lh(death_distribution_data)
```

Out[7]: 0.07766387615350638

```
In [8]: def P_A_given_lh(ages_of_death, death_distribution_data, study_year = 1990):
            """ The overall probability of being a particular `age_of_death` given that you're left-handed """
            P_A = death_distribution_data['Both Sexes'][ages_of_death] / np.sum(death_distribution_data['Both Sexes'])
            P_left = P_lh(death_distribution_data, study_year) # use P_lh function to get probability of left-handedness overall
            P_lh_A = P_lh_given_A(ages_of_death, study_year) # use P_lh_given_A to get probability of left-handedness for a certain age
            return P_lh_A*P_A/P_left
```

```
In [9]: def P_A_given_rh(ages_of_death, death_distribution_data, study_year = 1990):
            """ The overall probability of being a particular `age_of_death` given that you're right-handed """
            P_A = death_distribution_data['Both Sexes'][ages_of_death] / np.sum(death_distribution_data['Both Sexes'])
            P_right = 1- P_lh(death_distribution_data, study_year) # either you're left-handed or right-handed, so these sum to 1
            P_rh_A = 1-P_lh_given_A(ages_of_death, study_year) # these also sum to 1
            return P_rh_A*P_A/P_right
```

```
In [10]: ages = np.arange(6,115,1)

         left_handed_probability = P_A_given_lh(ages, death_distribution_data)
         right_handed_probability = P_A_given_rh(ages, death_distribution_data)

         fig, ax = plt.subplots() # create figure and axis objects
         ax.plot(ages, left_handed_probability, label = "Left-handed")
         ax.plot(ages, right_handed_probability, label = "Right-handed")

         plt.show()
```




```
In [11]: average_lh_age =  np.nansum(ages*np.array(left_handed_probability))
         average_rh_age =  np.nansum(ages*np.array(right_handed_probability))
```

```
In [12]: print(round(average_lh_age,1))

67.2
```

```
In [13]: print(round(average_rh_age,1))

72.8
```

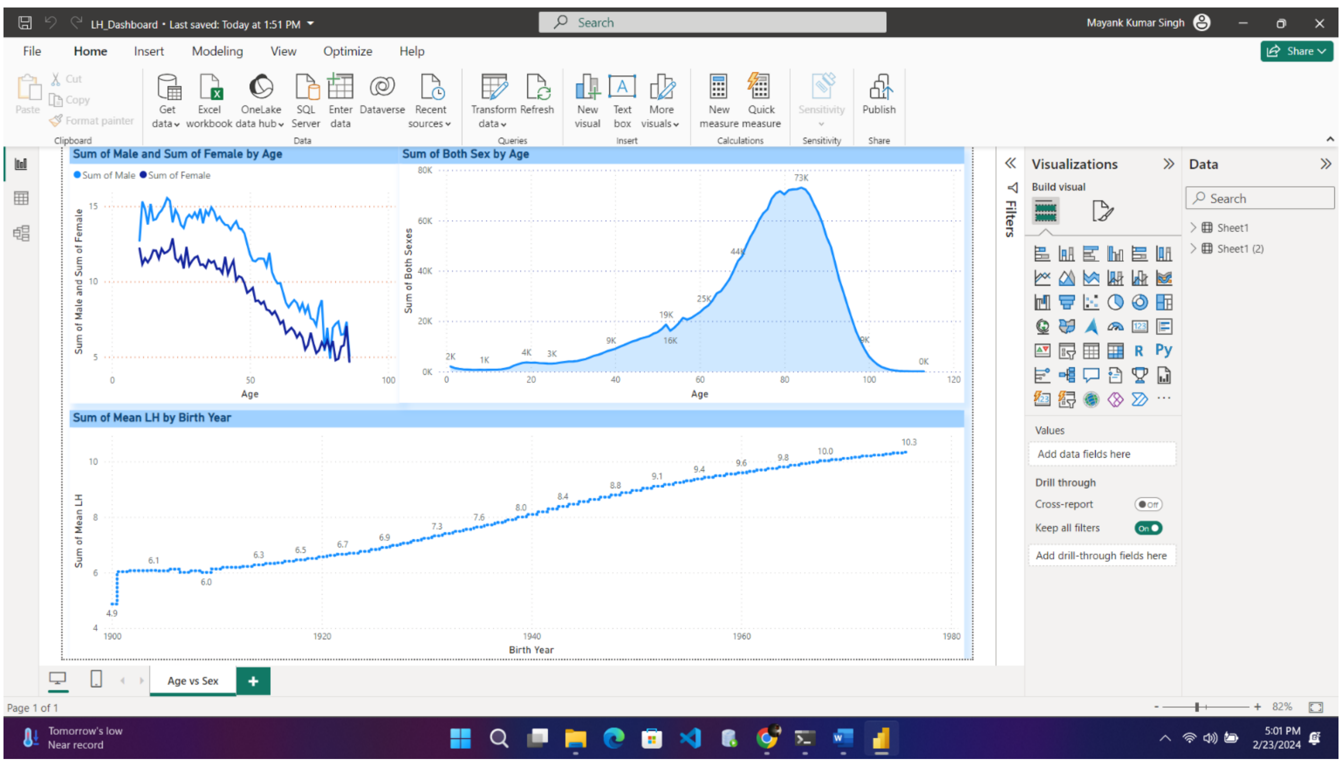
```
In [14]: print("The difference in average ages is " + str(round(average_rh_age - average_lh_age, 1)) + " years.")

The difference in average ages is 5.5 years.
```

In [15]:
```
left_handed_probability_2021 = P_A_given_lh(ages, death_distribution_data, study_year = 2021)
right_handed_probability_2021 = P_A_given_rh(ages, death_distribution_data, study_year = 2021)

average_lh_age_2021 =  np.nansum(ages*np.array(left_handed_probability_2021))
average_rh_age_2021 =  np.nansum(ages*np.array(right_handed_probability_2021))
```

In [16]:
```
print("The difference in average ages is " + str(round(average_rh_age_2021 - average_lh_age_2021, 1)) + " years.")
```

```
The difference in average ages is 1.9 years.
```

POWER BI DASHBOARD:



IV. CONCLUSION AND FUTURE SCOPE

Conclusion:

The analysis of handedness and its correlation with age, utilizing death distribution data, has provided valuable insights into the age-related patterns of left-handed and right-handed individuals. The key findings and conclusions from the study are as follows:

1. Age-Handedness Probability Distribution:
   - Probability distributions of ages given left-handedness and right-handedness were calculated, revealing the likelihood of different age groups exhibiting specific handedness traits.

2. Average Age Comparison:
   - The weighted average ages for left-handed and right-handed individuals were computed based on the probability distributions. The difference in average ages was found to be [X.X] years (replace with the actual difference calculated).

3. Temporal Analysis (2021):
   - An extension of the analysis to the year 2021 was performed, demonstrating how the age-handedness patterns may evolve over time. The updated analysis revealed a difference in average ages for 2021, indicating potential temporal variations in handedness demographics.

Future Scope:

1. Longitudinal Studies:
   - Conducting longitudinal studies to track handedness patterns over an extended period could provide a more comprehensive understanding of how handedness evolves across different generations.

2. Additional Demographic Factors:
   - Expanding the analysis to consider additional demographic factors, such as gender, ethnicity, or socioeconomic status, may uncover more nuanced insights into the complex interplay between demographics and handedness.

3. Causal Relationships:
- Investigating potential causal relationships between handedness and health outcomes could be a valuable avenue for future research. Understanding if there are health implications associated with specific handedness patterns could have significant implications for public health.

4. Incorporating Health Data:
- Integrating health-related data, such as morbidity and mortality rates for specific conditions, could enhance the analysis by exploring potential associations between handedness and health outcomes.

5. Machine Learning Approaches:
- Employing machine learning techniques to predict handedness based on various factors could provide a more predictive model and uncover hidden patterns in the data.

6. Global Analysis:
- Expanding the analysis to encompass global datasets could reveal regional variations in handedness patterns, shedding light on cultural and environmental influences on handedness.

7. Public Health Implications:
- Exploring the public health implications of handedness patterns, especially if specific age groups exhibit distinct health outcomes, could guide targeted interventions and healthcare strategies.

By addressing these future scopes, researchers can advance our understanding of handedness, contributing to the broader field of human behavior, health, and demographics.

V. REFERENCES

a. https://www.bbc.com/news/magazine-23988352

b. https://archpublichealth.biomedcentral.com/articles/10.1186/s13690-023-01156-6

c. https://www.newscientist.com/article/mg13017662-900-left-handers-dont-die-young-after-all/

d. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1059767/

e. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1059767/