Final Report of Traineeship Program 2023

On

"DEATH AGE DIFFERENCE OF RIGHT HANDERS WITH LEFT HANDERS"

MEDTOUREASY

29th June 2023

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 sparing their valuable time in spite of their 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

ABSTRACT

This case study examines the age difference at death between right-handers and left-handers, considering the changing rates of left-handedness over time. The study is based on a National Geographic survey conducted in 1986, which collected data on hand preference for writing and throwing from over a million respondents aged 10 to 86.

The analysis of this extensive dataset revealed that rates of left-handedness were approximately 13% among individuals younger than 40 but decreased to around 5% by the age of 80. Researchers Avery Gilbert and Charles Wysocki concluded that this age-dependence was primarily influenced by the social acceptability of left-handedness, rather than being a direct effect of age. They proposed that the rates of left-handedness were determined by the year of birth, suggesting that if the same survey were conducted today, a similar age-dependent distribution would be observed but shifted in relation to the current generation.

The study also observed additional effects related to hand preference. Among left-handers, concordance for writing and throwing was more prevalent than combinations involving right-handed writing and left-handed throwing. The prevalence of sinistrality displayed distinct and stable patterns before the age of 50, but changing patterns emerged thereafter.

By exploring the age-related changes in left-handedness rates, this study aims to investigate the impact of these changing rates on the apparent mean age at death of left-handed individuals. The findings from this case study contribute to a deeper understanding of the complex relationship between handedness, age, and mortality, highlighting the influence of societal factors on these associations.

1. INTRODUCTION

1.1 About the Company

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

1.2 About the Project

This project aims in reproducing the average age difference at death purely from the changing rates of left-handedness overtime, refuting the claim of early death for left-handers.

A National Geographic survey in 1986 resulted in over a million responses that included age, sex, and hand preference for throwing and writing. Researchers Avery Gilbert and Charles Wysocki analyzed this data and noticed that rates of left-handedness were around 13% for people younger than 40 but decreased with age to about 5% by the age of 80. They concluded based on analysis of a subgroup of people who throw left-handed but write right-handed that this age-dependence was primarily due to changing social acceptability of left-handedness. This means that the rates aren't a factor of *age* specifically but rather of the *year you were born*, and if the same study was done today, we should expect a shifted version of the same distribution as a function of age. Ultimately, we'll see what effect this changing rate has on the apparent mean age of death of left-handed people.

This study includes using pandas library and Bayesian statistics to analyze the probability of being a certain age at death given that you are reported as left-handed or right-handed. Cleaning of the dataset, plotting charts by importing matplotlib (python plotting library) and numpy to perform efficient and fast numeric computing.

Hence, this project aims at collecting and analyzing large data sets to create intuitive visualizations representing Death age difference of right-handers with left-handers in order to gain meaningful insights.

Each of this tasks and visualizations displayed are created using Python language on Jupyter Notebook (IDE) and using pandas and numpy libraries, which are then used to analyze the related data and draw conclusions about the same.

1.3 Objective

This project focuses on reproducing a difference in average age at death purely from the changing rates of left-handedness over time, refuting the claim of early death for left-handers collecting data from various sources like Death distribution data of US from 1999, 1992 paper by Gilberth and Wysocki, etc. and using the coding language Python and packages like pandas and numpy and Bayesian statistics which will enable to analyze draw conclusions.

2. 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 Language and Platform Used

2.2.1 Language: Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation via the off-side rule. Python is dynamically typed and garbage-collected. The important features of R are:

- Easy to code.
- Free and Open-Source.
- Robust Standard Library.
- Easy to read.

2.2.2 IDE: Jupyter Notebook

JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality. Major feature of allows you to create and share documents that contain live code, mathematical equations, graphics, maps, plots, visualizations, and narrative text. It integrates with many programming languages like Python, PHP, R, etc.

2.2.3 Package: Pandas, Numpy and Matplotlib

Pandas is a python library used for working with datasets. It has functions for analyzing, cleaning, exploring, and manipulating data. The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. Create publication quality plots.

Installation:

```
pip install pandas

pip install numpy

pip install matplotlib
```

Importing:

```
import pandas as pd

import numpy as np

import mstplotlib.pyplot as  plt
```

- The **import pandas as pd** statement imports the pandas library and assigns it the alias 'pd'. This allows you to use pandas functions and classes by prefixing them with 'pd'.

- The **import numpy as np** statement imports the numpy library and assigns it the alias 'np'. This allows you to use numpy functions and classes by prefixing them with 'np'.

- The **import matplotlib.pyplot as plt** statement imports the pyplot module from the matplotlib library and assigns it the alias 'plt'. This allows you to create plots and visualizations using the pyplot module.

3. IMPLEMENTATION AND INSIGHTS

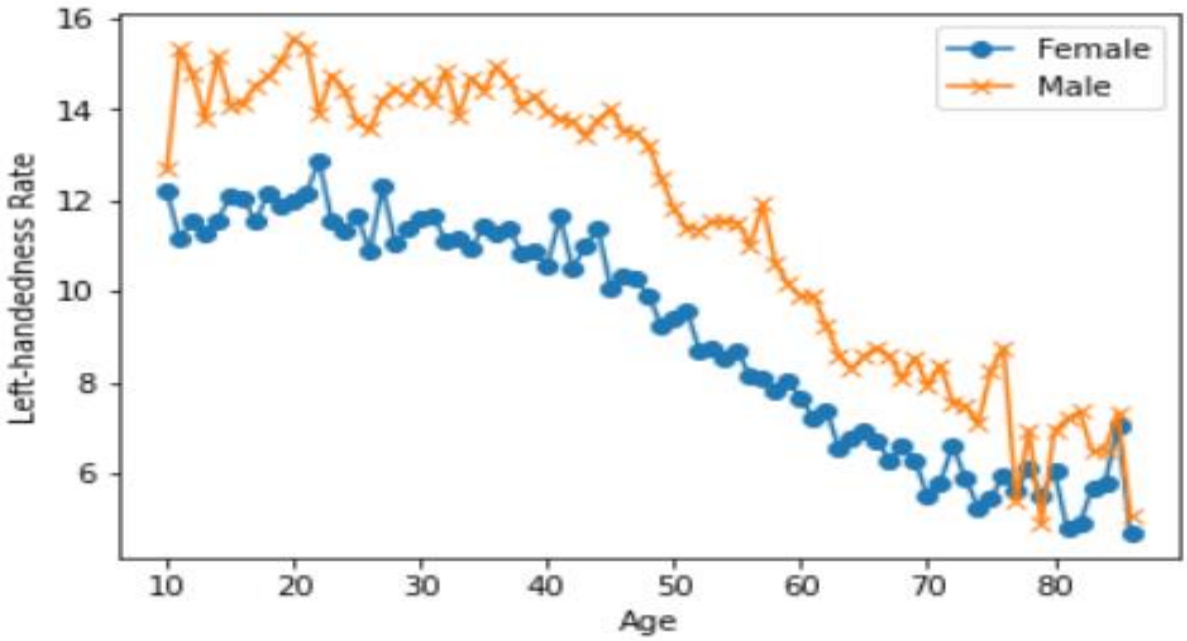
3.1 Data loading and Visualization:

In the first step, we import the necessary libraries, including pandas and matplotlib, and load the left-handedness data from an online source. The data contains information on left-handedness rates categorized by age and gender. We plot the left-handedness rates for males and females against age to visualize any trends or patterns.

```
# import libraries
import pandas as pd
import matplotlib.pyplot as plt

# load the data
data_url_1 = "https://gist.githubusercontent.com/mbonsma/8da0990b71ba9a09f7de395574e54df1/raw/aec88b30af87fad8d45da7e774223f91dad09e88/lh_dat
lefthanded_data = pd.read_csv(data_url_1)

# plot male and female left-handedness rates vs. age
# matplotlib inline
fig, ax = plt.subplots() # create figure and axis objects
ax.plot(lefthanded_data['Age'], lefthanded_data['Female'], label = 'Female', marker = 'o') # plot "Female" vs. "Age"
ax.plot(lefthanded_data['Age'], lefthanded_data['Male'], label = 'Male', marker = 'x') # plot "Male" vs. "Age"
ax.legend() # add a legend
ax.set_xlabel('Age')
ax.set_ylabel('Left-handedness Rate')

plt.show()
```

- The code begins by importing the pandas library and assigning it the alias 'pd'. It also imports the pyplot module from the matplotlib library and assigns it the alias 'plt'. These libraries will be used for data manipulation and creating plots, respectively.
- The code specifies the URL of a CSV file containing left-handedness data. The **pd.read_csv()** function is used to read the data from the CSV file and store it in a pandas DataFrame called 'lefthanded_data'. The DataFrame will contain columns such as 'Age', 'Female', and 'Male', representing different variables related to left-handedness rates.
- The code creates a figure and axis object using **plt.subplots()**. Then, it plots the left-handedness rates for females and males against age using the **ax.plot()** function. The 'Age' column from the DataFrame is used as the x-axis, while the 'Female' and 'Male' columns are used as the y-axis data. The 'Female' data points are plotted as circles ('o') and labeled as 'Female', while the 'Male' data points are plotted as crosses ('x') and labeled as 'Male'. The **ax.legend()** function adds a legend to the plot. The **ax.set_xlabel()** and **ax.set_ylabel()** functions set the labels for the x-axis and y-axis, respectively.
- Finally, **plt.show()** is called to display the plot.

INSIGHTS:



The output of this code is a line plot with two lines representing the left-handedness rates for females and males as a function of age. The x-axis represents the age, while the y-axis represents the left-handedness rate.

- ✓ The plot shows how the left-handedness rates vary with age for both females and males.
- ✓ From the plot, we can observe the general trends in left-handedness rates for each gender. It allows us to see if there are any significant differences or similarities between the left-handedness rates of females and males across different age groups.
- ✓ The plot provides a visual representation of the relationship between age and left-handedness rates, which can be useful for analysing patterns or making comparisons.
- ✓ By examining the slopes and patterns of the lines, we can identify age ranges where left-handedness rates are higher or lower for each gender.
- ✓ The plot can also serve as a starting point for further analysis or investigation into the factors that might influence left-handedness rates, such as cultural or societal changes over time.

3.2 Data Manipulation and Visualization:

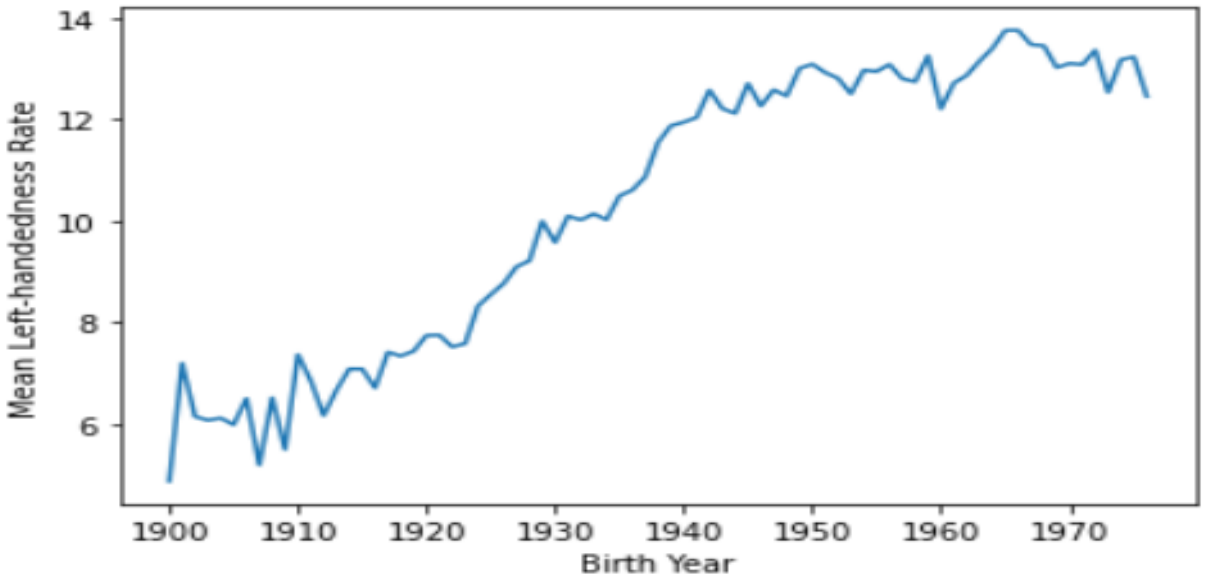
In step 2, we create new columns in the data to represent the birth year and the average left-handedness rate. These calculations provide additional insights into the data. We plot the average left-handedness rate against birth year to understand how it changes over time.

```
# create a new column for birth year of each age
lefthanded_data['Birth_year'] = 1986 - lefthanded_data['Age']

# create a new column for the average of male and female
lefthanded_data['Mean_lh'] = (lefthanded_data['Female'] + lefthanded_data['Male'])/2

# create a plot of the 'Mean_lh' column vs. 'Birth_year'
fig, ax = plt.subplots()
ax.plot(lefthanded_data['Birth_year'], lefthanded_data['Mean_lh']) # plot 'Mean_lh' vs. 'Birth_year'
ax.set_xlabel('Birth Year') # set the x label for the plot
ax.set_ylabel('Mean Left-handedness Rate') # set the y label for the plot

plt.show()
```

- First line of code subtracts the 'Age' column from 1986 to calculate the birth year for each age in the DataFrame. It creates a new column called 'Birth_year' and assigns the calculated birth years to it.
- Second line of code calculates the mean left-handedness rate by taking the average of the 'Female' and 'Male' columns. It creates a new column called 'Mean_lh' and assigns the calculated mean values to it.
- The code creates a figure and axis object using **plt.subplots()**. Then, it plots the mean left-handedness rate ('Mean_lh') against birth year ('Birth_year') using **ax.plot()**. The 'Birth_year' column is used as the x-axis, and the 'Mean_lh' column is used as the y-axis data. The **ax.set_xlabel()** and **ax.set_ylabel()** functions set the labels for the x-axis and y-axis, respectively. Finally, **plt.show()** is called to display the plot.

INSIGHTS:

Plot visualizes the relationship between birth year and the mean left-handedness rate, allowing for insights into the historical patterns and trends of left-handedness prevalence.



- ✓ The mean left-handedness rate fluctuates over different birth years, indicating variations in left-handedness prevalence across different generations.

- ✓ The plot shows the general trend of left-handedness rates over time. You can observe whether the rates increase, decrease, or remain relatively stable across birth years.

- ✓ The slope or steepness of the line can provide information about the rate of change in left-handedness prevalence over generations. Steeper slopes indicate more significant changes in prevalence.
- ✓ By analyzing the plot, you can identify periods or birth cohorts with higher or lower left-handedness rates, which may be of interest for further investigation or analysis.

3.3 Probability Calculation:

Next, we define a function called ‘P_lh_given_A’ that calculates the probability of being left-handed given the age of death. This function utilizes the left-handedness rates from the data and handles different age ranges. The function takes an array of ages of death and a study year as inputs and returns the corresponding probabilities.

```python
# import library
import numpy as np

# create a function for P(LH | A)
def P_lh_given_A(ages_of_death, study_year = 1990):
    """ P(Left-handed | ages of death), calculated based on the reported rates of left-handedness.
    Inputs: numpy array of ages of death, study_year
    Returns: probability of left-handedness given that subjects died in `study_year` at ages `ages_of_death` """

    # Use the mean of the 10 last and 10 first points for left-handedness rates before and after the start
    early_1900s_rate = lefthanded_data['Mean_lh'].tail(11).values[::-1]
    late_1900s_rate = lefthanded_data['Mean_lh'].head(11).values
    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 is 10
    oldest_age = study_year - 1986 + 86 # the oldest age 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 ages 'ages_of_death'
    P_return[ages_of_death > oldest_age] = early_1900s_rate.mean()/100
    P_return[ages_of_death < youngest_age] = late_1900s_rate.mean()/100
    P_return[np.logical_and((ages_of_death <= oldest_age), (ages_of_death >= youngest_age))] = middle_rates / 100

    return P_return
```

- The function signature is defined with the required input parameters `ages_of_death` and an optional parameter `study_year` with a default value of 1990.
- The function assumes the existence of a dataset named `lefthanded_data`, which contains information about the mean rates of left-handedness (`Mean_lh`) for different birth years. This dataset is not provided in the code snippet, so we can assume it is defined elsewhere.
- The function initializes an empty array `P_return` to store the calculated probabilities of left-handedness for each age of death.

- The code then extracts the left-handedness rates for three different time periods:

  - `early_1900s_rate`: The mean left-handedness rates for the 10 most recent birth years before the `study_year` (from the `lefthanded_data` dataset). The `tail(11)` function retrieves the last 11 rows of the `Mean_lh` column (in reverse order using `[::-1]` to get the earliest years first), and `values` returns the corresponding values as a NumPy array.

  - `late_1900s_rate`: The mean left-handedness rates for the 10 earliest birth years after the `study_year`. The `head(11)` function retrieves the first 11 rows of the `Mean_lh` column, and `values` returns the corresponding values as a NumPy array.

  - `middle_rates`: The left-handedness rates for birth years corresponding to the ages of death in `ages_of_death`. It filters the `lefthanded_data` dataset to include only rows where the `Birth_year` column matches the difference between `study_year` and each age in `ages_of_death`. The `Mean_lh` column values are extracted.

- The code determines the youngest and oldest possible ages in `ages_of_death` based on the assumption that the youngest age is 10 and the oldest age is 86. These values are calculated as `study_year - 1986 + 10` and `study_year - 1986 + 86`, respectively.

- The calculated left-handedness probabilities are assigned to the corresponding elements in `P_return`:

  - Ages greater than the oldest possible age are set to the mean of `early_1900s_rate` divided by 100 (converted to a probability).

  - Ages less than the youngest possible age are set to the mean of `late_1900s_rate` divided by 100.

  - Ages between the youngest and oldest possible ages are set to the corresponding values in `middle_rates` divided by 100.

- Finally, the function returns the array `P_return` containing the probabilities of left-handedness corresponding to the ages of death provided in `ages_of_death`.

INSIGHTS:

- ✓ The function calculates the probabilities of left-handedness based on the provided age of death and the reported rates from the ‘lefthanded_data’ DataFrame.

- ✓ The rates used for the early 1900s and late 1900s are based on the mean left-handedness rates from the available data points.

- ✓ For ages within the range of available birth years, the function uses the specific rates reported in the 'lefthanded_data' DataFrame.

- ✓ The output of this function provides insights into the estimated probabilities of left-handedness for individuals who died at different ages in the specified study year.

3.4 Death Distribution Data Loading

In this step, we load death distribution data for the United States in 1999. The data provides information on the number of deaths at different ages. We clean the data by dropping any NaN values and prepare it for further analysis.

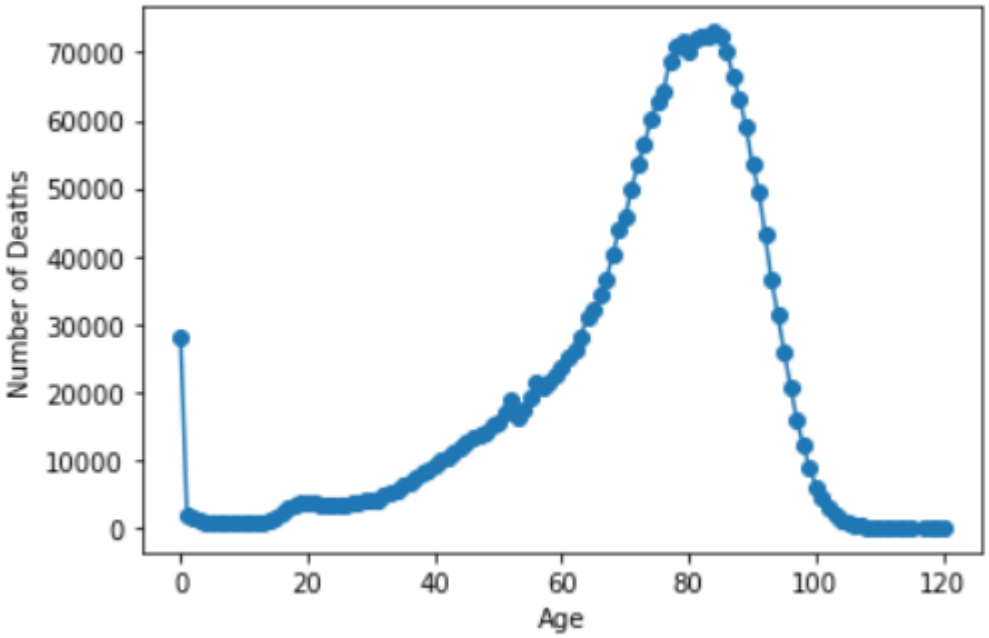
```
# Death distribution data for the United States in 1999
data_url_2 = "https://gist.githubusercontent.com/mbonsma/2f4076aab6820ca1807f4e29f75f18ec/raw/62f3ec07514c7e31f5979beeca86f19991540796/cdc_vs

# load death distribution data
death_distribution_data = pd.read_csv(data_url_2, sep = '\t', skiprows = [1])

# drop NaN values from the `Both Sexes` column
death_distribution_data = death_distribution_data.dropna(subset = ['Both Sexes'])

# plot number of people who died as a function of age
fig, ax = plt.subplots()
ax.plot(death_distribution_data['Age'], death_distribution_data['Both Sexes'], marker='o') # plot 'Both Sexes' vs. 'Age'
ax.set_xlabel('Age')
ax.set_ylabel('Number of Deaths')

plt.show()
```

- The URL `data_url_2` points to a file hosted on GitHub Gist, which contains the death distribution data for the United States in 1999. The file is in TSV (tab-separated values) format.
- The code uses the `pd.read_csv()` function from the pandas library to read the data from the specified URL. The `sep='\t'` parameter is set to indicate that the data is tab-separated. The `skiprows=[1]` parameter is used to skip the second row of the file, which contains additional information about the data.
- The loaded data is stored in the `death_distribution_data` DataFrame.
- The code then drops rows that contain NaN (missing) values in the 'Both Sexes' column. The `dropna()` function with the `subset=['Both Sexes']` parameter ensures that only rows with NaN values in the 'Both Sexes' column are dropped.
- A plot is created using `plt.subplots()`, which returns a figure (`fig`) and an axes object (`ax`). The figure and axes are used to customize the plot.
- The `ax.plot()` function is used to plot the number of people who died (`death_distribution_data['Both Sexes']`) as a function of age (`death_distribution_data['Age']`). The `marker='o'` parameter specifies that a marker

should be placed at each data point.

- Axes labels are set using `ax.set_xlabel()` and `ax.set_ylabel()` to label the x-axis as 'Age' and the y-axis as 'Number of Deaths', respectively.
- Finally, `plt.show()` is called to display the plot.

INSIGHTS:



- ✓ The output of this code is a plot that shows the number of people who died as a function of age in the United States in 1999. The x-axis represents age, and the y-axis represents the number of deaths.

- ✓ The plot displays a curve that typically starts at a lower value for younger ages, increases gradually, and peaks at a certain age range (often referred to as the "mortality peak"), before gradually declining for older ages.

- ✓ The mortality peak in the plot indicates the age range around 80 has the highest number of deaths occurred.

- ✓ The shape of the curve provide insights into patterns of mortality and age-related diseases or conditions prevalent in the population.

- ✓ Analysing the plot in conjunction with other data and factors helps identify trends and patterns in mortality rates and inform public health policies and interventions.

3.5 Overall Probability Calculation:

We define a function called ‘P_lh’, which calculates the overall probability of being left-handed if someone died in the study year. This function uses the death distribution data and the previously defined ‘P_lh_given_A’ function. It multiplies the number of deaths at each age by the corresponding left-handedness probability and returns the normalized probability.

```
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
    Input: dataframe of death distribution data, study year
    Output: P(LH), a single floating point number """
    p_list = death_distribution_data['Both Sexes'] * P_lh_given_A(death_distribution_data['Age'].values, study_year) # multiply number of dea
    p = p_list.sum() # calculate the sum of p_list
    return p / death_distribution_data['Both Sexes'].sum() # normalize to total number of people (sum of death_distribution_data['Both Sexes'

print(P_lh(death_distribution_data))
```

- The function `P_lh` takes two input parameters: `death_distribution_data`, which is a DataFrame containing death distribution data, and `study_year`, which is an optional parameter with a default value of 1990.

- Within the function, a list `p_list` is created by multiplying the number of dead people at each age group (`death_distribution_data['Both Sexes']`) with the corresponding probability of being left-handed (`P_lh_given_A(death_distribution_data['Age'].values, study_year)`). The `P_lh_given_A` function is called with the ages of death (`death_distribution_data['Age'].values`) and the study year to calculate the probability of left-handedness for each age group.

- The code then calculates the sum of all elements in `p_list` using the `sum()` function and assigns the result to the variable `p`.

- Finally, the function returns the value of `p` divided by the total number of people (sum of `death_distribution_data['Both Sexes']`). This normalizes the probability to account for the total population.

- The `print(P_lh(death_distribution_data))` statement calls the `P_lh` function with the `death_distribution_data` DataFrame as input and prints the resulting overall probability of being left-handed for individuals who died in the study year.

INSIGHTS:

The output of this code is a single floating-point number, which represents the overall probability of being left-handed if a person died in the specified study year.

- ✓ The output probability represents the estimated likelihood of being left-handed among individuals who died in the study year, based on the death distribution data and the calculated probabilities of left-handedness for different age groups.
- ✓ The probability value can range from 0 to 1, where 0 indicates no probability of being left-handed and 1 indicates a certainty of being left-handed.
- ✓ The output allows for quantitative comparison of left-handedness probabilities across different study years or datasets, providing insights into potential variations in left-handedness rates over time.
- ✓ It can be used to analyze and understand the prevalence of left-handedness within a specific context, such as a particular year or population.
- ✓ The output value can be interpreted as an estimation of the proportion of left-handed individuals among those who died in the study year, based on the available data.
- ✓ Comparing the output probabilities for different study years or datasets can reveal trends or patterns in left-handedness rates over time.
- ✓ It's important to note that the accuracy of the output probability depends on the quality and representativeness of the death distribution data and the underlying assumptions used in calculating the probabilities of left-handedness for different age groups.

3.6 Age Given LH Probability Calculation:

In this step, we define a function called ‘P_A_given_lh’, which calculates the probability of being a particular age at death given that the person is left-handed. The function utilizes the death distribution data, ‘P_lh’ function, and the previously defined ‘P_lh_given_A’ function. It computes the probability by multiplying the left-handedness probability, the number of deaths at each age, and the overall probability of being left-handed.

```
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.loc[death_distribution_data['Age'].isin(ages_of_death), 'Both Sexes'] / 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
```

- The function `P_A_given_lh` takes three input parameters: `ages_of_death`, which is a list or array of ages at death for which the probability is calculated, `death_distribution_data`, which is a DataFrame containing death distribution data, and `study_year`, which is an optional parameter with a default value of 1990.
- Within the function, the code calculates the probability of being a particular `age_of_death` (`P_A`) by dividing the number of dead people at each `age_of_death` in the `death_distribution_data` DataFrame by the total number of people (sum of `death_distribution_data['Both Sexes']`). This is done using the `loc` function to filter the DataFrame based on matching ages in `ages_of_death`.
- The code then calculates the overall probability of being left-handed (`P_left`) by calling the `P_lh` function, passing the `death_distribution_data` and `study_year` as inputs. This provides the probability of being left-handed for individuals who died in the study year.
- Next, the code calculates the probability of being left-handed (`P_lh_A`) for each specific `age_of_death` using the `P_lh_given_A` function and passing `ages_of_death` and `study_year` as inputs.
- Finally, the function returns the product of `P_lh_A`, `P_A`, and the inverse of `P_left` (`P_lh_A * P_A / P_left`). This gives the overall probability of being a particular `age_of_death` given that an individual is left-handed.

INSIGHTS:

The output of this code is an array of probabilities, where each element represents the probability of a particular ‘age_of_death’ given that an individual is left-handed.

- ✓ The output probabilities provide insights into the distribution of ages at death among left-handed individuals based on the death distribution data and left-handedness probabilities.
- ✓ The probabilities are conditional on an individual being left-handed. They represent the likelihood of dying at a specific age among the left-handed population.
- ✓ By comparing the probabilities for different ages, you can identify age groups where left-handed individuals have a higher or lower likelihood of dying compared to others.
- ✓ The output can be used to analyze and understand the relationship between left-handedness and age at death, providing insights into potential associations or patterns.
- ✓ It's important to note that the accuracy of the output probabilities depends on the quality and representativeness of the death distribution data and the underlying assumptions used in calculating the probabilities of left-handedness for different age groups. Additionally, the output assumes that left-handedness and age at death are independent variables, which may not always be the case in reality.

3.7 Age Given RH Probability Calculation:

Similar to step 6, we define a function called 'P_A_given_rh', which calculates the probability of being a particular age at death given that the person is right-handed. This function uses the same approach as 'P_A_given_lh' but calculates the probability for right-handed individuals.

```
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.loc[death_distribution_data['Age'].isin(ages_of_death), 'Both Sexes'] / death_distribution_data['Both Sexes
    P_right = 1 - P_lh(death_distribution_data, study_year) # either you're left-handed or right-handed, so P_right = 1 - P_left
    P_rh_A = 1 - P_lh_given_A(ages_of_death, study_year) # P_rh_A = 1 - P_lh_A

    return P_rh_A*P_A/P_right
```

- The function `P_A_given_rh` takes three input parameters: `ages_of_death`, which is a list or array of ages at death for which the probability is calculated, `death_distribution_data`, which is a DataFrame containing death distribution data, and `study_year`, which is an optional parameter with a default value of 1990.
- Within the function, the code calculates the probability of being a particular `age_of_death` (`P_A`) by dividing the number of dead people at each `age_of_death` in the `death_distribution_data` DataFrame by the total number of people (sum of `death_distribution_data['Both Sexes']`). This is done using the `loc` function to filter the DataFrame based on matching ages in `ages_of_death`.
- The code then calculates the overall probability of being right-handed (`P_right`) by subtracting the probability of being left-handed (`P_left`) from 1. This is because the individual is assumed to be either left-handed or right-handed, and the sum of the probabilities should be 1.
- Next, the code calculates the probability of being right-handed (`P_rh_A`) for each specific `age_of_death` by subtracting the probability of being left-handed (`P_lh_A`) from 1. This assumes that if an individual is not left-handed, they must be right-handed.
- Finally, the function returns the product of `P_rh_A`, `P_A`, and the inverse of `P_right` (`P_rh_A * P_A / P_right`). This gives the overall probability of being a particular `age_of_death` given that an individual is right-handed.

OUTPUT:

The output of this code is an array of probabilities, where each element represents the probability of a particular ‘age_of_death’ given that an individual is right-handed.

- ✓ The output probabilities provide insights into the distribution of ages at death among right-handed individuals based on the death distribution data and right-handedness probabilities.
- ✓ The probabilities are conditional on an individual being right-handed. They represent the likelihood of dying at a specific age among the right-handed population.
- ✓ By comparing the probabilities for different ages, you can identify age groups where right-handed individuals have a higher or lower likelihood of dying compared to others.
- ✓ The output can be used to analyze and understand the relationship between right-handedness and age at death, providing insights into potential associations or patterns.
- ✓ It's important to note that the accuracy of the output probabilities depends on the quality and representativeness of the death distribution data and the underlying assumptions used in calculating the probabilities of right-handedness for different age groups. Additionally, the output assumes that right-handedness and age at death are independent variables, which may not always be the case in reality.

3.8 Visualization of Probabilities:

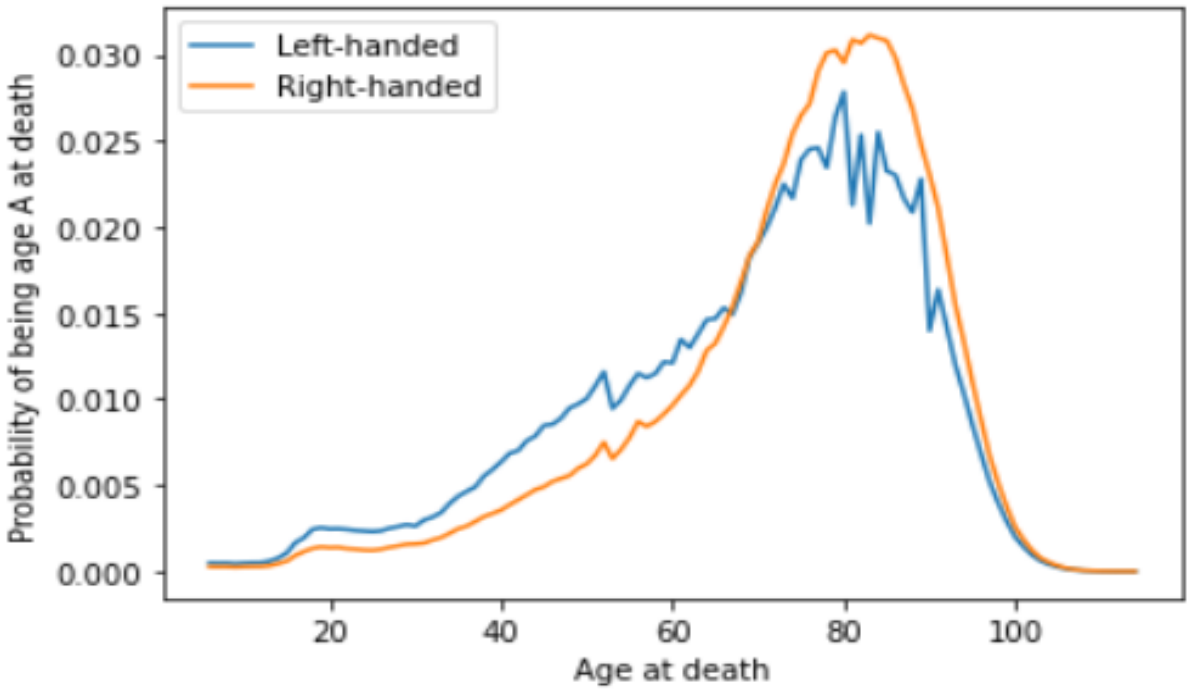
We plot the probabilities of being left-handed and right-handed at different ages of death. This visualization helps us understand the age distributions for left-handed and right-handed individuals.

```
ages = np.arange(6, 115, 1) # make a list of ages of death to plot

# calculate the probability of being left- or right-handed for each
left_handed_probability = P_A_given_lh(ages, death_distribution_data)
right_handed_probability = P_A_given_rh(ages, death_distribution_data)

# create a plot of the two probabilities vs. age
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")
ax.legend() # add a legend
ax.set_xlabel("Age at death")
ax.set_ylabel(r"Probability of being age A at death")
```

- The line `ages = np.arange(6, 115, 1)` creates an array `ages` containing values from 6 to 114 (inclusive) with a step size of 1. This represents the range of ages at death for which the probabilities will be calculated.
- The line `left_handed_probability = P_A_given_lh(ages, death_distribution_data)` calls the `P_A_given_lh` function with `ages` and `death_distribution_data` as inputs. This calculates the probabilities of being left-handed for each age in the `ages` array based on the death distribution data.
- The line `right_handed_probability = P_A_given_rh(ages, death_distribution_data)` calls the `P_A_given_rh` function with `ages` and `death_distribution_data` as inputs. This calculates the probabilities of being right-handed for each age in the `ages` array based on the death distribution data.
- The code then creates a plot using `**plt.subplots()**`, which returns a figure (`fig`) and an axes object (`ax`). The figure and axes are used to customize the plot.
- The line `**ax.plot(**ages, left_handed_probability, label="Left-handed"**)**` plots the left-handed probabilities on the axes. It uses the `ages` array as the x-values and `left_handed_probability` as the y-values. The `label` parameter sets the label for the left-handed probability line in the legend.

- The line `**ax.plot(**ages, right_handed_probability, label="Right-handed"**)**` plots the right-handed probabilities on the axes. It uses the `ages` array as the x-values and `right_handed_probability` as the y-values. The `label` parameter sets the label for the right-handed probability line in the legend.
- The line `**ax.legend()**` adds a legend to the plot, displaying the labels specified for the left-handed and right-handed lines.
- The lines `**ax.set_xlabel(**"Age at death"**)**` and `**ax.set_ylabel(**r"Probability of being age A at death"**)**` set the x-axis and y-axis labels, respectively.
- Finally, the plot is displayed using `**plt.show()**`.

INSIGHTS:



The output of this code is a plot that visualizes the probabilities of being a certain age at death given left-handedness or right-handedness. The x-axis represents the age at death, and the y-axis represents the probability of being that age at death.

- ✓ The plot allows for a visual comparison of the probabilities of being a certain age at death for left-handed and right-handed individuals.
- ✓ By examining the plotted lines, we can observe how the probabilities change across different ages.
- ✓ Any differences in the shapes of the lines can provide insights into potential associations between handedness and age at death. For example, if the left-handed probability line consistently stays above the right-handed probability line, it suggests a higher likelihood of being a certain age at death for left-handed individuals compared to right-handed individuals (and vice versa).
- ✓ Peaks or dips in the lines indicates age ranges where left-handed or right-handed individuals have a higher or lower probability of being age A at death.
- ✓ The plot helps identify any age groups where handedness may be associated with variations in the distribution of ages at death.
- ✓ It's important to consider the underlying data and assumptions used to calculate these probabilities, as well as the limitations of the dataset and analysis methods employed.

3.9 Average Age Calculation and Comparison:

We calculate the average ages for the left-handed and right-handed groups based on the probabilities calculated in step 6 and step 7. By multiplying each age with its corresponding probability and summing the results, we obtain the average ages for each group. We compare the average ages and calculate the difference between them.

```
# calculate average ages for left-handed and right-handed groups
# use np.array so that two arrays can be multiplied
average_lh_age =  np.nansum(left_handed_probability*np.array(ages))
average_rh_age =  np.nansum(right_handed_probability*np.array(ages))

# print the average ages for each group
print("Average age for left-handed group:", round(average_lh_age, 1))
print("Average age for right-handed group:", round(average_rh_age, 1))

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

- The line `average_lh_age = **np.nansum(**left_handed_probability***np.array(**ages**))**` calculates the average age for the left-handed group. It multiplies the `left_handed_probability` array by the `ages` array element-wise using `**np.array**` so that the arrays can be multiplied. The `**np.nansum**` function is then used to sum the resulting array, ignoring any **NaN** (Not a Number) values that might be present. This provides the sum of the products of probabilities and ages.
- The line `average_rh_age = **np.nansum(**right_handed_probability***np.array(**ages**))**` calculates the average age for the right-handed group using the same approach as above.
- The lines `**print(**"Average age for left-handed group:", **round(**average_lh_age, 1**))**` and `**print(**"Average age for right-handed group:", **round(**average_rh_age, 1**))**` print the calculated average ages for the left-handed and right-handed groups, respectively. The `round` function is used to round the average ages to one decimal place.
- The line `**print(**"The difference in average ages is " + **str(round(**average_lh_age - average_rh_age, 1**))** + " years.")` calculates the difference in average ages between the left-handed and right-handed groups and prints it as a string. The `round` function is used to round the difference to one decimal place.

INSIGHTS:

- ✓ The average age for the left-handed group represents the average age at death for individuals who are left-handed, calculated based on the given probabilities and age values, which comes out to be 67.3 years.
- ✓ Similarly, the average age for the right-handed group represents the average age at death for individuals who are right-handed, which is 72.8 years.
- ✓ The difference in average ages between the left-handed and right-handed groups indicates whether there is a notable disparity in the average ages at death between these two groups.
- ✓ A positive difference suggests that the average age of the left-handed group is higher than that of the right-handed group, while a negative difference suggests the opposite.
- ✓ The difference is a negative value -5.5 years, which suggests that the average age of the left-handed group is lesser than that of the right-handed group.
- ✓ The magnitude of the difference indicates the extent of the disparity in average ages between the two groups.
- ✓ By comparing the average ages and their difference, you can gain insights into potential associations between handedness and age at death. These insights can be further explored and analyzed to understand the underlying factors that contribute to the observed differences.

3.10 Analysis for a Different Study Year:

We repeat steps 5 to 9 for a different study year, in this case, 2018. We calculate the probabilities and average ages for left-handed and right-handed individuals. Finally, we compare the average ages between the two.

```
# Calculate the probability of being left- or right-handed for all ages
left_handed_probability_2018 = P_A_given_lh(ages, death_distribution_data, study_year = 2018)
right_handed_probability_2018 = P_A_given_rh(ages, death_distribution_data, study_year = 2018)

# calculate average ages for left-handed and right-handed groups
average_lh_age_2018 = np.nansum(ages*np.array(left_handed_probability_2018))
average_rh_age_2018 = np.nansum(ages*np.array(right_handed_probability_2018))

print("The difference in average ages is " +
      str(round(average_rh_age_2018 - average_lh_age_2018, 1)) + " years.")
```

- The lines `left_handed_probability_2018 = P_A_given_lh(ages, death_distribution_data, study_year = 2018)` and `right_handed_probability_2018 = P_A_given_rh(ages, death_distribution_data, study_year = 2018)` calculate the probabilities of being left-handed and right-handed for all ages in the `ages` array, specifically for the year 2018. These probabilities are based on the death distribution data.

- The line `average_lh_age_2018 = **np.nansum(**ages*__np.array(__left_handed_probability_2018**))**` calculates the average age for the left-handed group in 2018. It multiplies the `ages` array element-wise by the `left_handed_probability_2018` array and then uses `**np.nansum**` to sum the resulting array, ignoring any **NaN** values. This provides the sum of the products of ages and probabilities for the left-handed group in 2018.

- Similarly, the line `average_rh_age_2018 = **np.nansum(**ages*__np.array(__right_handed_probability_2018**))**` calculates the average age for the right-handed group in 2018 using the same approach as above.

- The line `**print(**"The difference in average ages is " + **str(round(**average_rh_age_2018 - average_lh_age_2018, 1**))** + " years.")` calculates the difference in average ages between the right-handed and left-handed groups in 2018 and prints it as a string. The difference is obtained by subtracting the average age of the left-handed group from the average age of the right-handed group. The `round` function is used to round the difference to one decimal place.

OUTPUT:

The output of this code provides insights into the difference in average ages between the left-handed and right-handed groups in the year 2018:

- ✓ The difference in average ages indicates whether there is a notable disparity in the average ages at death between the left-handed and right-handed groups specifically for the year 2018.
- ✓ The difference in average ages comes out to be 2.4 years for 2018, which is much lesser compared to the study done as per 1990 data.
- ✓ By examining the difference in average ages, we can gain insights into potential variations in the relationship between handedness and age at death over time. This information can be valuable for understanding any shifts or trends in the observed differences between the two groups across different years or time periods.

4. CONCLUSION AND FUTURE SCOPE

We got a pretty big age gap between left-handed and right-handed people purely as a result of the changing rates of left-handedness in the population, which is good news for left-handers: you probably won't die young because of your sinisterness. The reported rates of left-handedness have increased from just 3% in the early 1900s to about 11% today, which means that older people are much more likely to be reported as right-handed than left-handed, and so looking at a sample of recently deceased people will have more old right-handers.

Our number is still less than the 9-year gap measured in the study. It's possible that some of the approximations we made are the cause:

- We used death distribution data from almost ten years after the study (1999 instead of 1991), and we used death data from the entire United States instead of California alone (which was the original study).
- We extrapolated the left-handedness survey results to older and younger age groups, but it's possible our extrapolation wasn't close enough to the true rates for those ages.

One thing can be done next is to figure out how much variability can be expected to encounter in the age difference purely because of random sampling: if we take a smaller sample of recently deceased people and assign handedness with the probabilities of the survey, what does that distribution look like? How often can we encounter an age gap of nine years using the same data and assumptions?

Calculating the age gap for 2018 instead of in 1990, the gap turns out to be much smaller since rates of left-handedness haven't increased for people born after about 1960. Both the National Geographic study and the 1990 study happened at a unique time - the rates of left-handedness had been changing across the lifetimes of most people alive, and the difference in handedness between old and young was at its most striking.

5. REFERENCES

Data Collection

The following websites have been referred to obtain the input data and statistics:

a. NVSS - Mortality Tables (cdc.gov)

b. Deaths By Single Years of Age, Race, and Sex: United States, 1999 (cdc.gov)

c. Hand preference and age in the United States - PubMed (nih.gov)

Programming References

The following websites have been referred for Python coding and Bayesian statistics:

a. https://www.python.org/

b. https://www.w3schools.com/python/pandas/default.asp

c. https://www.w3schools.com/python/numpy/default.asp

d. https://www.w3schools.com/python/matplotlib_intro.asp

e. https://www.quantstart.com/articles/Bayesian-Statistics-A-Beginners-Guide/