Alzheimers Disease

TASK 1

Import Libraries

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

import warnings
warnings.filterwarnings('ignore')

import geopandas as gpd
```

```
C:\Users\THOTA AKHIL\anaconda3\Lib\site-packages\pandas\core\arrays\masked.py:60: UserWarning: Pandas requires version '1.3.6' or newer of 'bottleneck' (version '1.3.5' currently installed).
  from pandas.core import (
```

Import Dataset

In [2]:
```
data = pd.read_csv(r"C:\Users\THOTA AKHIL\Downloads\Alzheimer_s_Disease_and_Healthy_Aging_Data.csv")
```

In [3]:
```
data
```

Out[3]:

	RowId+K1A3A1:O1A1:N1A1:O1AA1:O1	YearStart	YearEnd	LocationAbbr	LocationDesc	Datasource	Class	Topic	Question	Data_Value_Unit	...	Stratification2	Geolocation	ClassID	TopicID	QuestionID	LocationID	StratificationCategoryID1	StratificationID1	Strat
0	BRFSS~2015~2015~9004~Q43~TOC11~AGE~OVERALL	2015	2015	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	NaN	NaN	C01	TOC11	Q43	9004	AGE	65PLUS	
1	BRFSS~2019~2019~9004~Q27~TMC03~AGE~OVERALL	2019	2019	WEST	West	BRFSS	Mental Health	Lifetime diagnosis of depression	Percentage of older adults with a lifetime dia...	%	...	NaN	NaN	C05	TMC03	Q27	9004	AGE	AGE_OVERALL	
2	BRFSS~2019~2019~9004~Q43~TOC11~AGE~GENDER	2019	2019	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	Female	NaN	C01	TOC11	Q43	9004	AGE	65PLUS	
3	BRFSS~2019~2019~9004~Q43~TOC11~AGE~OVERALL	2019	2019	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	NaN	NaN	C01	TOC11	Q43	9004	AGE	AGE_OVERALL	
4	BRFSS~2015~2015~9004~Q43~TOC11~AGE~GENDER	2015	2015	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	Male	NaN	C01	TOC11	Q43	9004	AGE	AGE_OVERALL	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
250932	BRFSS~2015~2015~49~Q22~TSC07~AGE~RACE	2015	2015	UT	Utah	BRFSS	Screenings and Vaccines	High blood pressure ever	Percentage of older adults who have ever been ...	%	...	Asian/Pacific Islander	POINT(-111.58713063499971 39.360700171000474)	C03	TSC07	Q22	49	AGE	5064	
250933	BRFSS~2018~2018~49~Q46~TOC10~AGE~OVERALL	2018	2018	UT	Utah	BRFSS	Overall Health	Disability status, including sensory or mobili...	Percentage of older adults who report having a...	%	...	NaN	POINT(-111.58713063499971 39.360700171000474)	C01	TOC10	Q46	49	AGE	AGE_OVERALL	
250934	BRFSS~2015~2015~49~Q18~TSC08~AGE~RACE	2015	2015	UT	Utah	BRFSS	Screenings and Vaccines	Influenza vaccine within past year	Percentage of older adults who reported influe...	%	...	White, non-Hispanic	POINT(-111.58713063499971 39.360700171000474)	C03	TSC08	Q18	49	AGE	65PLUS	
250935	BRFSS~2018~2018~49~Q16~TNC03~AGE~RACE	2018	2018	UT	Utah	BRFSS	Nutrition/Physical Activity/Obesity	No leisure-time physical activity within past ...	Percentage of older adults who have not had an...	%	...	Black, non-Hispanic	POINT(-111.58713063499971 39.360700171000474)	C02	TNC03	Q16	49	AGE	5064	
250936	BRFSS~2017~2017~59~Q04~TOC04~AGE~GENDER	2017	2017	US	United States, DC & Territories	BRFSS	Overall Health	Taking medication for high blood pressure	Percentage of older adults who have been told ...	%	...	Female	NaN	C01	TOC04	Q04	59	AGE	65PLUS	

250937 rows × 31 columns

In [ ]:
```
```

In [4]:
```
data.columns
```

Out[4]:
```
Index(['RowId+K1A3A1:O1A1:N1A1:O1AA1:O1', 'YearStart', 'YearEnd',
       'LocationAbbr', 'LocationDesc', 'Datasource', 'Class', 'Topic',
       'Question', 'Data_Value_Unit', 'DataValueTypeID', 'Data_Value_Type',
       'Data_Value', 'Data_Value_Alt', 'Data_Value_Footnote_Symbol',
       'Data_Value_Footnote', 'Low_Confidence_Limit', 'High_Confidence_Limit',
       'StratificationCategory1', 'Stratification1', 'StratificationCategory2',
       'Stratification2', 'Geolocation', 'ClassID', 'TopicID', 'QuestionID',
       'LocationID', 'StratificationCategoryID1', 'StratificationID1',
       'StratificationCategoryID2', 'StratificationID2'],
      dtype='object')
```

In [5]:
```
data.T
```

Out[5]:

	0	1	2	3	4	
RowId+K1A3A1:O1A1:N1A1:O1AA1:O1	BRFSS~2015~2015~9004~Q43~TOC11~AGE~OVERALL	BRFSS~2019~2019~9004~Q27~TMC03~AGE~OVERALL	BRFSS~2019~2019~9004~Q43~TOC11~AGE~GENDER	BRFSS~2019~2019~9004~Q43~TOC11~AGE~OVERALL	BRFSS~2015~2015~9004~Q43~TOC11~AGE~GENDER	BRFSS~2015~2015~
YearStart	2015	2019	2019	2019	2015	
YearEnd	2015	2019	2019	2019	2015	
LocationAbbr	WEST	WEST	WEST	WEST	WEST	
LocationDesc	West	West	West	West	West	
Datasource	BRFSS	BRFSS	BRFSS	BRFSS	BRFSS	
Class	Overall Health	Mental Health	Overall Health	Overall Health	Overall Health	
Topic	Arthritis among older adults	Lifetime diagnosis of depression	Arthritis among older adults	Arthritis among older adults	Arthritis among older adults	
Question	Percentage of older adults ever told they have...	Percentage of older adults with a lifetime dia...	Percentage of older adults ever told they have...	Percentage of older adults ever told they have...	Percentage of older adults ever told they have...	Percentage of
Data_Value_Unit	%	%	%	%	%	
DataValueTypeID	PRCTG	PRCTG	PRCTG	PRCTG	PRCTG	
Data_Value_Type	Percentage	Percentage	Percentage	Percentage	Percentage	
Data_Value	48.4	16.7	54.9	38.6	32.6	
Data_Value_Alt	48.4	16.7	54.9	38.6	32.6	
Data_Value_Footnote_Symbol	NaN	NaN	NaN	NaN	NaN	
Data_Value_Footnote	NaN	NaN	NaN	NaN	NaN	
Low_Confidence_Limit	47.0	16.1	53.1	37.7	31.4	
High_Confidence_Limit	49.7	17.3	56.7	39.5	33.9	
StratificationCategory1	Age Group	Age Group	Age Group	Age Group	Age Group	
Stratification1	65 years or older	Overall	65 years or older	Overall	Overall	
StratificationCategory2	NaN	NaN	Gender	NaN	Gender	
Stratification2	NaN	NaN	Female	NaN	Male	
Geolocation	NaN	NaN	NaN	NaN	NaN	
ClassID	C01	C05	C01	C01	C01	
TopicID	TOC11	TMC03	TOC11	TOC11	TOC11	
QuestionID	Q43	Q27	Q43	Q43	Q43	
LocationID	9004	9004	9004	9004	9004	
StratificationCategoryID1	AGE	AGE	AGE	AGE	AGE	
StratificationID1	65PLUS	AGE_OVERALL	65PLUS	AGE_OVERALL	AGE_OVERALL	
StratificationCategoryID2	OVERALL	OVERALL	GENDER	OVERALL	GENDER	
StratificationID2	OVERALL	OVERALL	FEMALE	OVERALL	MALE	

31 rows × 250937 columns

Total Data Rows And Columns

In [6]:
```
len(data.columns)
```

Out[6]:
```
31
```

In [7]:
```
data.shape
```

Out[7]:
```
(250937, 31)
```

Top 5 Rows In Data

In [8]:
```
data.head()
```

Out[8]:

	RowId+K1A3A1:O1A1:N1A1:O1AA1:O1	YearStart	YearEnd	LocationAbbr	LocationDesc	Datasource	Class	Topic	Question	Data_Value_Unit	...	Stratification2	Geolocation	ClassID	TopicID	QuestionID	LocationID	StratificationCategoryID1	StratificationID1	StratificationCategoryID2	Stratifica
0	BRFSS~2015~2015~9004~Q43~TOC11~AGE~OVERALL	2015	2015	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	NaN	NaN	C01	TOC11	Q43	9004	AGE	65PLUS	OVERALL	O\
1	BRFSS~2019~2019~9004~Q27~TMC03~AGE~OVERALL	2019	2019	WEST	West	BRFSS	Mental Health	Lifetime diagnosis of depression	Percentage of older adults with a lifetime dia...	%	...	NaN	NaN	C05	TMC03	Q27	9004	AGE	AGE_OVERALL	OVERALL	O\
2	BRFSS~2019~2019~9004~Q43~TOC11~AGE~GENDER	2019	2019	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	Female	NaN	C01	TOC11	Q43	9004	AGE	65PLUS	GENDER	F
3	BRFSS~2019~2019~9004~Q43~TOC11~AGE~OVERALL	2019	2019	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	NaN	NaN	C01	TOC11	Q43	9004	AGE	AGE_OVERALL	OVERALL	O\
4	BRFSS~2015~2015~9004~Q43~TOC11~AGE~GENDER	2015	2015	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	%	...	Male	NaN	C01	TOC11	Q43	9004	AGE	AGE_OVERALL	GENDER	

5 rows × 31 columns

It looks like you're asking for a preview of the first few rows of a dataset, which is commonly achieved using the data.head()` function in pandas. Unfortunately, I don't have direct access to your dataset. However, if you upload your dataset file here, I can help you load it and display the first few rows.

Last 5 Rows in Dataset

In [9]:
```
data.tail()
```

Out[9]:

	RowId+K1A3A1:O1A1:N1A1:O1AA1:O1	YearStart	YearEnd	LocationAbbr	LocationDesc	Datasource	Class	Topic	Question	Data_Value_Unit	...	Stratification2	Geolocation	ClassID	TopicID	QuestionID	LocationID	StratificationCategoryID1	StratificationID1	Stratific
250932	BRFSS~2015~2015~49~Q22~TSC07~AGE~RACE	2015	2015	UT	Utah	BRFSS	Screenings and Vaccines	High blood pressure ever	Percentage of older adults who have ever been ...	%	...	Asian/Pacific Islander	POINT(-111.58713063499971 39.360700171000474)	C03	TSC07	Q22	49	AGE	5064	
250933	BRFSS~2018~2018~49~Q46~TOC10~AGE~OVERALL	2018	2018	UT	Utah	BRFSS	Overall Health	Disability status, including sensory or mobili...	Percentage of older adults who report having a...	%	...	NaN	POINT(-111.58713063499971 39.360700171000474)	C01	TOC10	Q46	49	AGE	AGE_OVERALL	
250934	BRFSS~2015~2015~49~Q18~TSC08~AGE~RACE	2015	2015	UT	Utah	BRFSS	Screenings and Vaccines	Influenza vaccine within past year	Percentage of older adults who reported influe...	%	...	White, non-Hispanic	POINT(-111.58713063499971 39.360700171000474)	C03	TSC08	Q18	49	AGE	65PLUS	
250935	BRFSS~2018~2018~49~Q16~TNC03~AGE~RACE	2018	2018	UT	Utah	BRFSS	Nutrition/Physical Activity/Obesity	No leisure-time physical activity within past ...	Percentage of older adults who have not had an...	%	...	Black, non-Hispanic	POINT(-111.58713063499971 39.360700171000474)	C02	TNC03	Q16	49	AGE	5064	
250936	BRFSS~2017~2017~59~Q04~TOC04~AGE~GENDER	2017	2017	US	United States, DC & Territories	BRFSS	Overall Health	Taking medication for high blood pressure	Percentage of older adults who have been told ...	%	...	Female	NaN	C01	TOC04	Q04	59	AGE	65PLUS	

5 rows × 31 columns

It seems you're asking for the last few rows of a dataset, which can be obtained using the tail() function in various data manipulation libraries like Pandas in Python.

Cleaning the Dataset

Check the wheather dataset contain the null values or not

In [10]:
```
data.isnull().sum()
```

Out[10]:
```
RowId+K1A3A1:O1A1:N1A1:O1AA1:O1        0
YearStart                              0
YearEnd                                0
LocationAbbr                           0
LocationDesc                           0
Datasource                             0
Class                                  0
Topic                                  0
Question                               0
Data_Value_Unit                        0
DataValueTypeID                        0
Data_Value_Type                        0
Data_Value                         81635
Data_Value_Alt                     81635
Data_Value_Footnote_Symbol        151823
Data_Value_Footnote               151823
Low_Confidence_Limit               81785
High_Confidence_Limit              81785
StratificationCategory1                0
Stratification1                        0
StratificationCategory2            32376
Stratification2                    32376
Geolocation                        26709
ClassID                                0
TopicID                                0
QuestionID                             0
LocationID                             0
StratificationCategoryID1              0
StratificationID1                      0
StratificationCategoryID2              0
StratificationID2                      0
dtype: int64
```

The data.isnull().sum() function in Python, when applied to a DataFrame (typically from the pandas library), returns the number of missing (null) values in each column. It helps identify columns with missing data and their counts, which can be useful for data cleaning and preprocessing.

In [11]:
```
round(data.isnull().sum() / data.shape[0] * 100.00).sort_values(ascending=False)
```

Out[11]:
```
Data_Value_Footnote               61.0
Data_Value_Footnote_Symbol        61.0
Data_Value                        33.0
Low_Confidence_Limit              33.0
Data_Value_Alt                    33.0
High_Confidence_Limit             33.0
Stratification2                   13.0
StratificationCategory2           13.0
Geolocation                       11.0
QuestionID                         0.0
TopicID                            0.0
ClassID                            0.0
StratificationCategory1            0.0
LocationID                         0.0
StratificationCategoryID1          0.0
StratificationID1                  0.0
Stratification1                    0.0
StratificationCategoryID2          0.0
RowId+K1A3A1:O1A1:N1A1:O1AA1:O1    0.0
YearStart                          0.0
Data_Value_Type                    0.0
DataValueTypeID                    0.0
Data_Value_Unit                    0.0
Question                           0.0
Topic                              0.0
Class                              0.0
Datasource                         0.0
LocationDesc                       0.0
LocationAbbr                       0.0
YearEnd                            0.0
StratificationID2                  0.0
dtype: float64
```

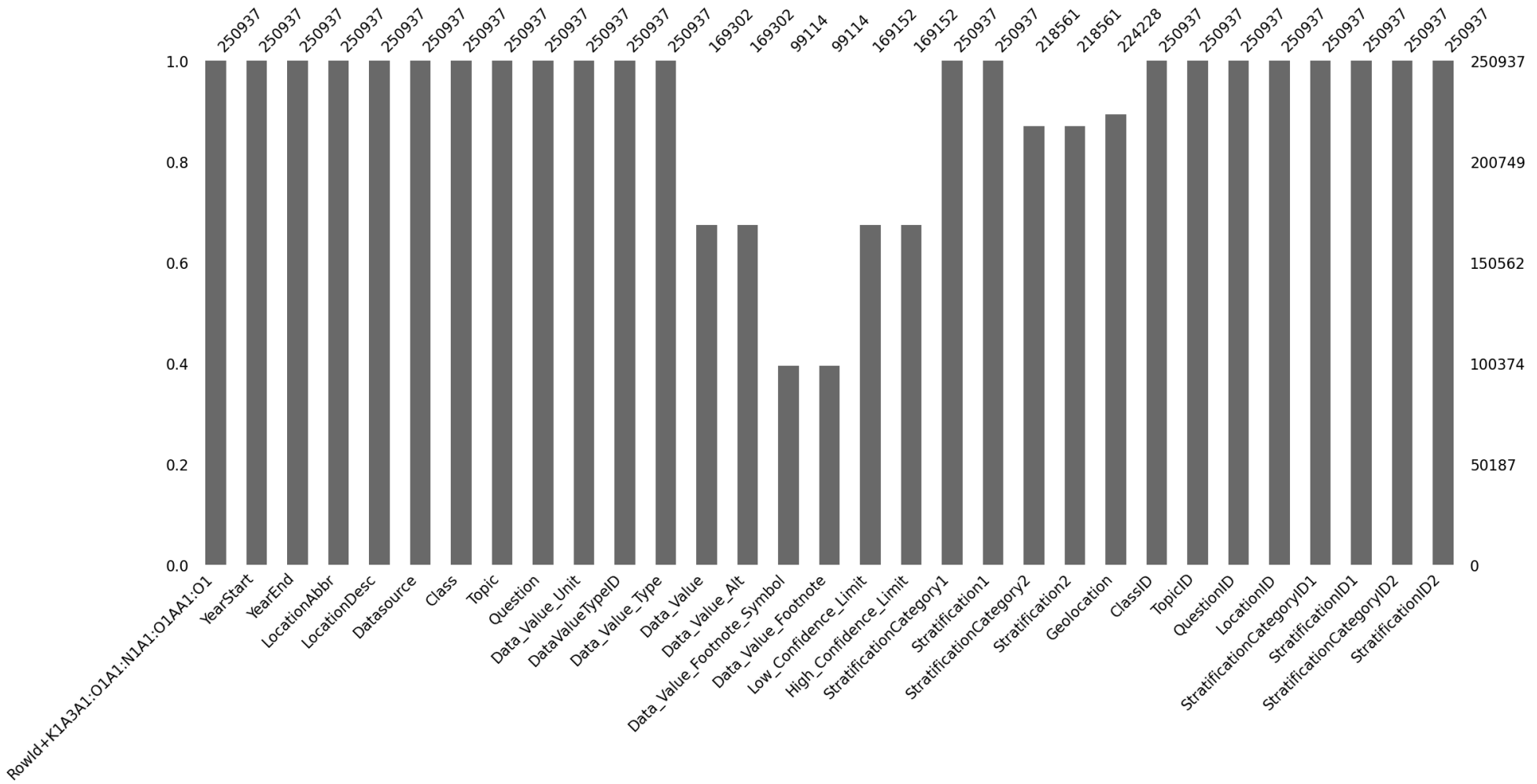
The code you've provided calculates the percentage of missing values in each column of a DataFrame named data. data.isnull().sum(): Counts the number of missing (NaN) values in each column. data.shape[0]: Returns the number of rows in the DataFrame. data.isnull().sum() / data.shape[0]: Calculates the proportion of missing values for each column.

- 100.00: Converts the proportion to a percentage. round(..., 2): Rounds the percentage to two decimal places. sort_values(ascending = False): Sorts the columns in descending order based on the percentage of missing values.

Null Values check in Visulization

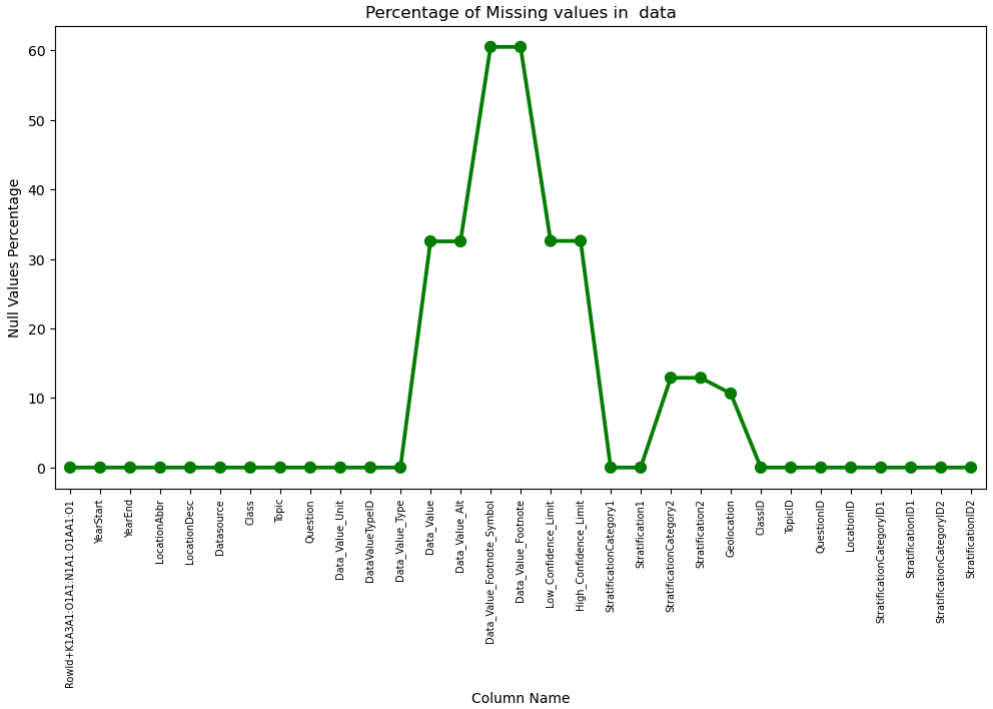
In [12]:
```
import missingno as mn
mn.bar(data)
```

Out[12]: <Axes: >



The missingno library is used for visualizing missing data in a dataset. The mn.bar(data) function call creates a bar chart to display the number of missing values in each column of the DataFrame data.

In [13]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

null_previous_data = pd.DataFrame((data.isnull().sum())*100/data.shape[0]).reset_index()
null_previous_data.columns = ['Column Name', 'Null Values Percentage']

fig = plt.figure(figsize=(12,6))
ax = sns.pointplot(x="Column Name", y="Null Values Percentage", data=null_previous_data, color='green')
plt.xticks(rotation=90, fontsize=7)
plt.title("Percentage of Missing values in  data")
plt.show()
```



Variable data: The variable data is not defined in your code snippet. Ensure that you have loaded your dataset into this variable.

No need to multiply by 100: When calculating the percentage of missing values, you multiplied the result by 100 but did not divide it by the number of rows. This will give incorrect results. The correct way to calculate the percentage of missing values is to use (data.isnull().sum() / data.shape[0]) * 100.

Delete Null Columns

1) Delete Columns

In [14]:
```
columns_to_remove = [
    'RowId+K1A3A1:O1A1:N1A1:O1AA1:O1',
    'Data_Value_Unit',
    'DataValueTypeID',
    'Data_Value_Type',
    'Data_Value_Alt',
    'Data_Value_Footnote_Symbol',
    'Data_Value_Footnote',
    'Geolocation'
]

data = data.drop(columns=columns_to_remove)
```

It looks like you're dropping some columns from your dataset. This code snippet will remove the specified columns from the data DataFrame.

fill nan values

In [15]:
```
data['Data_Value'].fillna(data['Data_Value'].mean(), inplace=True)
data['Low_Confidence_Limit'].fillna(data['Low_Confidence_Limit'].mode()[0], inplace=True)
data['High_Confidence_Limit'].fillna(data['High_Confidence_Limit'].mode()[0], inplace=True)
data['StratificationCategory2'].fillna(data['StratificationCategory2'].mode()[0], inplace=True)
data['Stratification2'].fillna(data['Stratification2'].mode()[0], inplace=True)
```

Numerical Imputation: Data_Value is filled with the mean value of the column, which is appropriate for continuous data. Low_Confidence_Limit and High_Confidence_Limit are filled with the mode, which works well if these columns are categorical or have a few repeated values.

Categorical Imputation:
StratificationCategory2 and Stratification2 are filled with the mode, which is a common method for categorical data.

data.info()

In [16]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250937 entries, 0 to 250936
Data columns (total 23 columns):
 #   Column                     Non-Null Count   Dtype
---  ------                     --------------   -----
 0   YearStart                  250937 non-null  int64
 1   YearEnd                    250937 non-null  int64
 2   LocationAbbr               250937 non-null  object
 3   LocationDesc               250937 non-null  object
 4   Datasource                 250937 non-null  object
 5   Class                      250937 non-null  object
 6   Topic                      250937 non-null  object
 7   Question                   250937 non-null  object
 8   Data_Value                 250937 non-null  float64
 9   Low_Confidence_Limit       250937 non-null  object
 10  High_Confidence_Limit      250937 non-null  object
 11  StratificationCategory1    250937 non-null  object
 12  Stratification1            250937 non-null  object
 13  StratificationCategory2    250937 non-null  object
 14  Stratification2            250937 non-null  object
 15  ClassID                    250937 non-null  object
 16  TopicID                    250937 non-null  object
 17  QuestionID                 250937 non-null  object
 18  LocationID                 250937 non-null  int64
 19  StratificationCategoryID1  250937 non-null  object
 20  StratificationID1          250937 non-null  object
 21  StratificationCategoryID2  250937 non-null  object
 22  StratificationID2          250937 non-null  object
dtypes: float64(1), int64(3), object(19)
memory usage: 44.0+ MB
```

It seems like you're looking to display the structure and basic information about a dataset, similar to how data. info() would work in a Python environment using pandas. This function provides an overview of the dataset, including the number of entries, column names, non-null counts, and data types.

Describe Data

In [17]:
```
data.describe()
```

Out[17]:

	YearStart	YearEnd	Data_Value	LocationID
count	250937.000000	250937.000000	250937.000000	250937.000000
mean	2017.940933	2018.169716	37.328349	793.866437
std	2.031564	2.081039	20.709800	2502.174327
min	2015.000000	2015.000000	0.000000	1.000000
25%	2016.000000	2016.000000	23.900000	19.000000
50%	2018.000000	2018.000000	37.328349	34.000000
75%	2020.000000	2020.000000	41.600000	49.000000
max	2021.000000	2021.000000	100.000000	9004.000000

provide descriptive statistics for a dataset, you can use the describe() method in Python's pandas library. This method gives a summary of the data, including count, mean, standard deviation, minimum, maximum, and quartile values for each numerical column.

In [18]:
```
data
```

Out[18]:

	YearStart	YearEnd	LocationAbbr	LocationDesc	Datasource	Class	Topic	Question	Data_Value	Low_Confidence_Limit	...	StratificationCategory2	Stratification2	ClassID	TopicID	QuestionID	LocationID	StratificationCategoryID1	StratificationID1	StratificationCategoryID2	StratificationID2
0	2015	2015	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	48.400000	47.0	...	Race/Ethnicity	White, non-Hispanic	C01	TOC11	Q43	9004	AGE	65PLUS	OVERALL	OVERALL
1	2019	2019	WEST	West	BRFSS	Mental Health	Lifetime diagnosis of depression	Percentage of older adults with a lifetime dia...	16.700000	16.1	...	Race/Ethnicity	White, non-Hispanic	C05	TMC03	Q27	9004	AGE	AGE_OVERALL	OVERALL	OVERALL
2	2019	2019	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	54.900000	53.1	...	Gender	Female	C01	TOC11	Q43	9004	AGE	65PLUS	GENDER	FEMALE
3	2019	2019	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	38.600000	37.7	...	Race/Ethnicity	White, non-Hispanic	C01	TOC11	Q43	9004	AGE	AGE_OVERALL	OVERALL	OVERALL
4	2015	2015	WEST	West	BRFSS	Overall Health	Arthritis among older adults	Percentage of older adults ever told they have...	32.600000	31.4	...	Gender	Male	C01	TOC11	Q43	9004	AGE	AGE_OVERALL	GENDER	MALE
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
250932	2015	2015	UT	Utah	BRFSS	Screenings and Vaccines	High blood pressure ever	Percentage of older adults who have ever been ...	37.328349	4.7	...	Race/Ethnicity	Asian/Pacific Islander	C03	TSC07	Q22	49	AGE	5064	RACE	ASN
250933	2018	2018	UT	Utah	BRFSS	Overall Health	Disability status, including sensory or mobili...	Percentage of older adults who report having a...	33.700000	32.0	...	Race/Ethnicity	White, non-Hispanic	C01	TOC10	Q46	49	AGE	AGE_OVERALL	OVERALL	OVERALL
250934	2015	2015	UT	Utah	BRFSS	Screenings and Vaccines	Influenza vaccine within past year	Percentage of older adults who reported influe...	59.100000	56.5	...	Race/Ethnicity	White, non-Hispanic	C03	TSC08	Q18	49	AGE	65PLUS	RACE	WHT
250935	2018	2018	UT	Utah	BRFSS	Nutrition/Physical Activity/Obesity	No leisure-time physical activity within past ...	Percentage of older adults who have not had an...	37.328349	4.7	...	Race/Ethnicity	Black, non-Hispanic	C02	TNC03	Q16	49	AGE	5064	RACE	BLK
250936	2017	2017	US	United States, DC & Territories	BRFSS	Overall Health	Taking medication for high blood pressure	Percentage of older adults who have been told ...	92.700000	92.1	...	Gender	Female	C01	TOC04	Q04	59	AGE	65PLUS	GENDER	FEMALE

250937 rows × 23 columns

Great! Now that you've completed the data cleaning process, you're ready to move on to data analysis and visualization. If you need help with any specific analysis, visualization, or interpretation of the cleaned data.

In [19]:
```
############          Cleaning_Part_Completed        ############
```

In [ ]:
```

```

Vizualization

In [20]:
```
data.columns
```

Out[20]:
```
Index(['YearStart', 'YearEnd', 'LocationAbbr', 'LocationDesc', 'Datasource',
       'Class', 'Topic', 'Question', 'Data_Value', 'Low_Confidence_Limit',
       'High_Confidence_Limit', 'StratificationCategory1', 'Stratification1',
       'StratificationCategory2', 'Stratification2', 'ClassID', 'TopicID',
       'QuestionID', 'LocationID', 'StratificationCategoryID1',
       'StratificationID1', 'StratificationCategoryID2', 'StratificationID2'],
      dtype='object')
```

In [21]:
```
#pip install ipywidgets
```

In [22]:
```
import ipywidgets as widgets
from IPython.display import display

columns = ['YearStart', 'YearEnd', 'LocationAbbr', 'LocationDesc', 'Datasource',
           'Class', 'Topic', 'Question', 'Data_Value', 'Low_Confidence_Limit',
           'High_Confidence_Limit', 'StratificationCategory1', 'Stratification1',
           'StratificationCategory2', 'Stratification2', 'ClassID', 'TopicID',
           'QuestionID', 'LocationID', 'StratificationCategoryID1',
           'StratificationID1', 'StratificationCategoryID2', 'StratificationID2']

column_dropdown = widgets.Dropdown(
    options=columns,
    description='Column:',
    disabled=False,
)
```

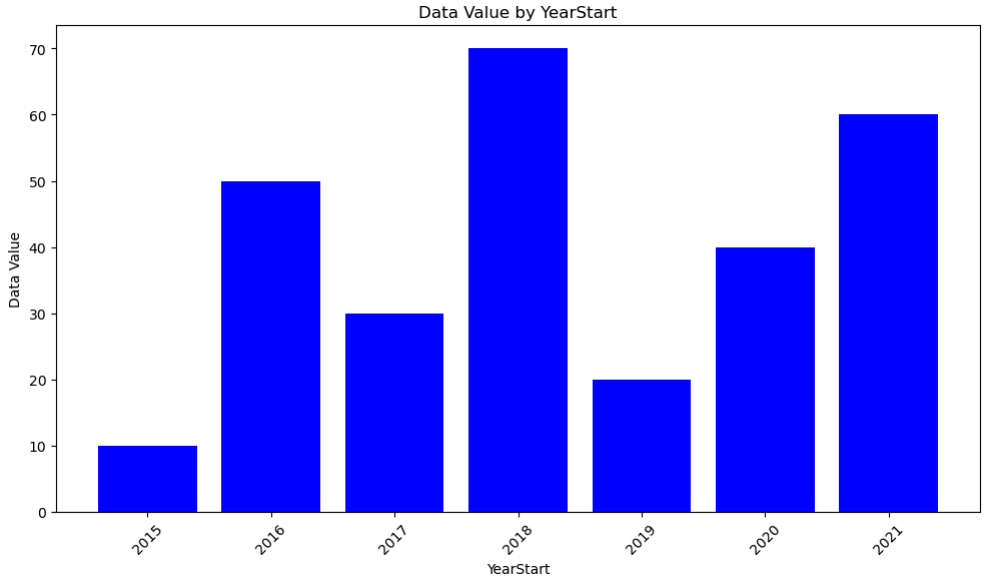
List of Columns: A list named columns contains the names of different columns from a dataset. Dropdown Widget: The Dropdown widget is created with the following parameters: options=columns: The dropdown options are populated with the items from the columns list. description='Column:': This is the label shown next to the dropdown menu. disabled=False: This means the dropdown menu is enabled and can be interacted with.

In [ ]:
```

```

In [23]:

```
import pandas as pd
import matplotlib.pyplot as plt
import ipywidgets as widgets
from ipywidgets import interact

data_1 = {
    'YearStart': [2015, 2019, 2017, 2020, 2016, 2021, 2018],
    'YearEnd': [2015, 2019, 2017, 2020, 2016, 2021, 2018],
    'LocationAbbr': ['VT', 'OR', 'WY', 'WI', 'MDW', 'NV', 'WV'],
    'LocationDesc': ['Vermont', 'Oregon', 'Wyoming', 'Wisconsin', 'Midwest', 'Nevada', 'West Virginia'],
    'Datasource': ['BRFSS'] * 7,
    'Class': ['Overall Health', 'Mental Health', 'Nutrition/Physical Activity/Obesity',
              'Smoking and Alcohol Use', 'Screenings and Vaccines', 'Caregiving', 'Cognitive Decline'],
    'Topic': ['Arthritis among older adults', 'Lifetime diagnosis of depression',
              'Frequent mental distress', 'Recent activity limitations in past month',
              'Eating 2 or more fruits daily', 'Fair or poor health among older adults with arthritis',
              'Prevalence of sufficient sleep'],
    'StratificationCategory1': ['Age Group'] * 7,
    'Stratification1': ['65 years or older', 'Overall', '50-64 years', '65 years or older', 'Overall', '50-64 years', 'Overall'],
    'StratificationCategory2': ['Race/Ethnicity', 'Gender', 'Gender', 'Race/Ethnicity', 'Gender', 'Gender', 'Race/Ethnicity'],
    'Stratification2': ['White, non-Hispanic', 'Female', 'Male', 'Black, non-Hispanic', 'Female', 'Male', 'Hispanic'],
    'ClassID': ['C01', 'C05', 'C02', 'C04', 'C03', 'C07', 'C06'],
    'TopicID': ['TOC11', 'TMC03', 'TMC01', 'TOC03', 'TNC01', 'TOC13', 'TOC09'],
    'QuestionID': ['Q43', 'Q27', 'Q03', 'Q35', 'Q01', 'Q45', 'Q34'],
    'LocationID': [50, 41, 56, 55, 9002, 32, 54],
    'StratificationCategoryID1': ['AGE'] * 7,
    'StratificationID1': ['65PLUS', 'AGE_OVERALL', '5064', '65PLUS', 'AGE_OVERALL', '5064', 'AGE_OVERALL'],
    'StratificationCategoryID2': ['RACE', 'GENDER', 'GENDER', 'RACE', 'GENDER', 'GENDER', 'RACE'],
    'StratificationID2': ['WHT', 'FEMALE', 'MALE', 'BLK', 'FEMALE', 'MALE', 'HIS'],
    'Data_Value': [10, 20, 30, 40, 50, 60, 70]  # Sample data values
}

df = pd.DataFrame(data_1)

dropdown = widgets.Dropdown(
    options=df.columns.tolist(),
    value='YearStart',
    description='Select X-axis:',
)

def plot_bar(column):
    plt.figure(figsize=(10, 6))
    plt.bar(df[column], df['Data_Value'], color='blue')
    plt.xlabel(column)
    plt.ylabel('Data Value')
    plt.title(f'Data Value by {column}')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

interact(plot_bar, column=dropdown)
```

Select X-axis: YearStart



Out[23]: <function __main__.plot_bar(column)>

Imports: It imports the necessary libraries: pandas for data manipulation, matplotlib.pyplot for plotting, and ipywidgets for interactive widgets.

Sample Data: A dictionary data is defined with sample data, including columns like 'YearStart', 'LocationAbbr', 'Class', 'Topic', 'Data_Value', etc.

DataFrame Creation: The sample data is converted into a pandas DataFrame df.

Dropdown Widget: A dropdown widget is created with the column names of the DataFrame as options. This widget allows the user to select a column to plot on the x-axis.

Plotting Function: The function plot_bar generates a bar chart with the selected column on the x-axis and 'Data_Value' on the y-axis. It customizes the chart with labels, title, and rotated x-axis labels for better readability.

Interactive Plot: The interact function from ipywidgets links the dropdown widget to the plot_bar function, allowing users to interactively change the x-axis column and see the corresponding bar chart.

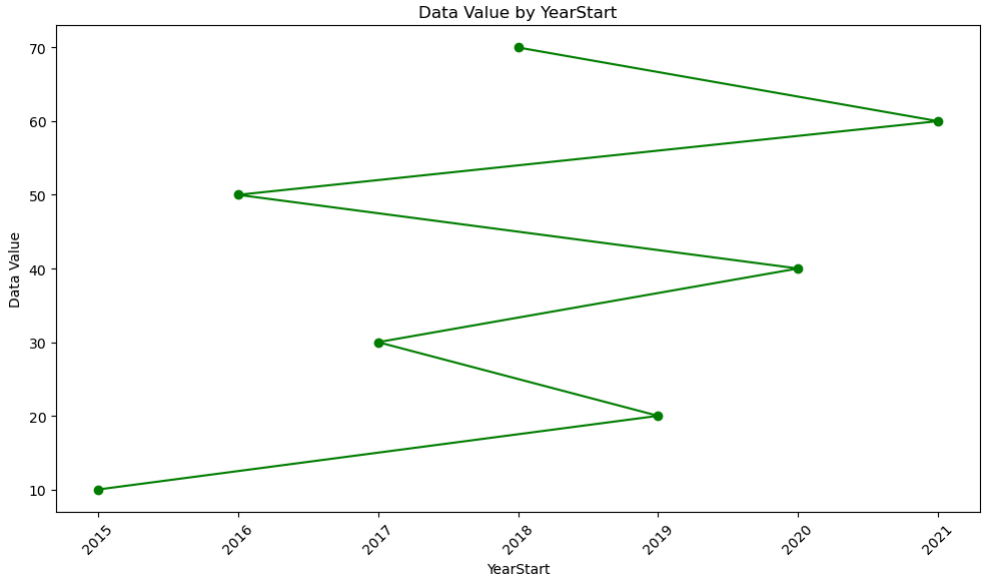
In short, the code allows users to interactively select a column from the dataset and visualize the data values for that column in a bar chart.

In [ ]:

In [ ]:

In [24]:
```
df = pd.DataFrame(data_1)

dropdown = widgets.Dropdown(
    options=df.columns.tolist(),
    value='YearStart',
    description='Select X-axis:',
)

def plot_line(column):
    plt.figure(figsize=(10, 6))
    plt.plot(df[column], df['Data_Value'], marker='o', linestyle='-', color='green')
    plt.xlabel(column)
    plt.ylabel('Data Value')
    plt.title(f'Data Value by {column}')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

interact(plot_line, column=dropdown)
```

Select X-axis: YearStart



Out[24]:
```
<function __main__.plot_line(column)>
```

Create DataFrame: df = pd.DataFrame(data) initializes a DataFrame from the data variable.

Dropdown Widget: widgets.Dropdown creates a dropdown menu populated with column names from the DataFrame. The default selection is 'YearStart'.

Plot Function: plot_line(column) is a function that plots a line graph. The x-axis is determined by the selected column from the dropdown, while the y-axis is always 'Data_Value'. It includes labels, title, and layout adjustments.

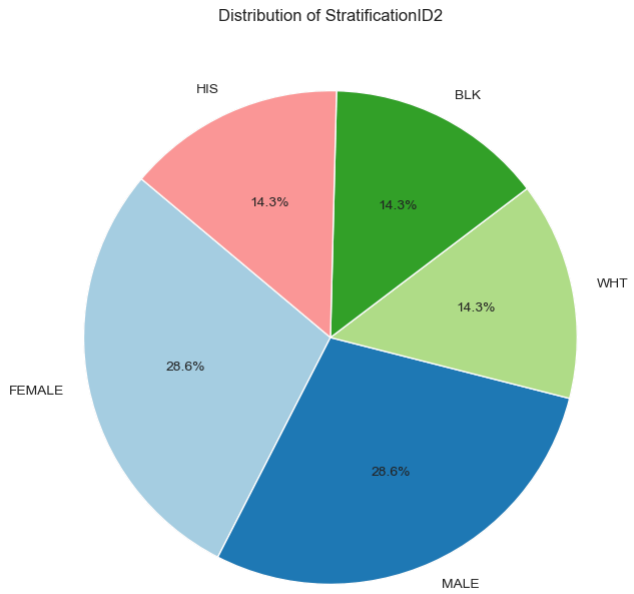
Interactive Plot: interact(plot_line, column=dropdown) makes the plot interactive. It connects the dropdown menu to the plot_line function, so selecting different columns updates the plot accordingly.

In [ ]:

In [ ]:

In [25]:
```
df = pd.DataFrame(data_1)

dropdown = widgets.Dropdown(
    options=df.columns.tolist(),
    value='YearStart',
    description='Select Column:',
)

def plot_pie(column):
    plt.figure(figsize=(8, 8))
    counts = df[column].value_counts()
    plt.pie(counts, labels=counts.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)
    plt.title(f'Distribution of {column}')
    plt.show()

interact(plot_pie, column=dropdown)
```

Select Colu… StratificationID2



Out[25]: <function __main__.plot_pie(column)>

Create a DataFrame: df is a DataFrame created from a data dictionary or similar source.

Dropdown Widget: dropdown is a widget that allows users to select a column from df. It lists all column names and defaults to 'YearStart'.

Plot Function: plot_pie(column) is a function that generates a pie chart for the selected column. It calculates the counts of each unique value in the column and creates a pie chart to show their distribution.

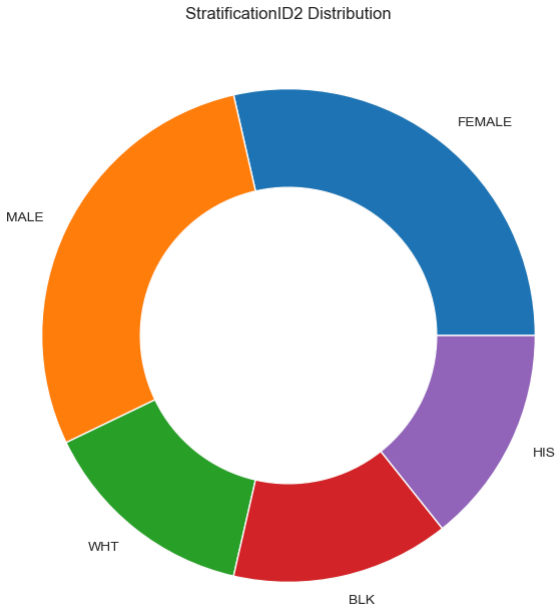
Interactive Plot: interact(plot_pie, column=dropdown) connects the plot_pie function to the dropdown widget. When a column is selected from the dropdown, the plot_pie function is called with that column, updating the pie chart accordingly.

In [ ]:

In [26]:

```
def plot_donut(column):
    data = df[column].value_counts()
    plt.figure(figsize=(8, 8))
    plt.pie(data, labels=data.index, wedgeprops={'width': 0.4})
    plt.title(f'{column} Distribution')
    plt.show()

interact(plot_donut, column=dropdown)
```

Select Colu… StratificationID2



Out[26]: <function __main__.plot_donut(column)>

plot_donut(column) Function:

data = df[column].value_counts(): Calculates the frequency of each unique value in the specified column of the DataFrame df. plt.figure(figsize=(8, 8)): Sets up the figure size for the plot. plt.pie(data, labels=data.index, wedgeprops={'width': 0.4}): Creates a donut chart by setting the width parameter in wedgeprops to 0.4, creating a ring-like appearance. plt.title(f'{column} Distribution'): Sets the title of the plot. plt.show(): Displays the plot. interact(plot_donut, column=dropdown):

interact: Creates an interactive widget that allows you to select different columns from a dropdown menu (dropdown), and it automatically updates the plot with the selected column's data.

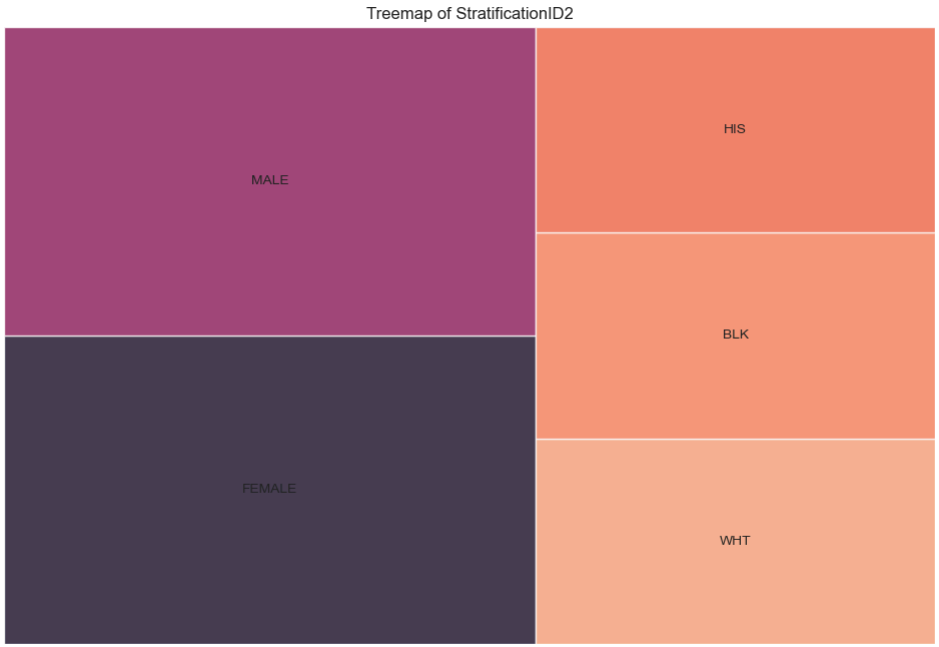
In [ ]:

In [27]:

```
#pip install squarify
```

In [28]:
```
import squarify

def plot_treemap(column):
    sizes = df[column].value_counts()
    plt.figure(figsize=(12, 8))
    squarify.plot(sizes=sizes.values, label=sizes.index, alpha=0.8)
    plt.title(f'Treemap of {column}')
    plt.axis('off')
    plt.show()

interact(plot_treemap, column=dropdown)
```

Select Colu… StratificationID2



Out[28]:
```
<function __main__.plot_treemap(column)>
```

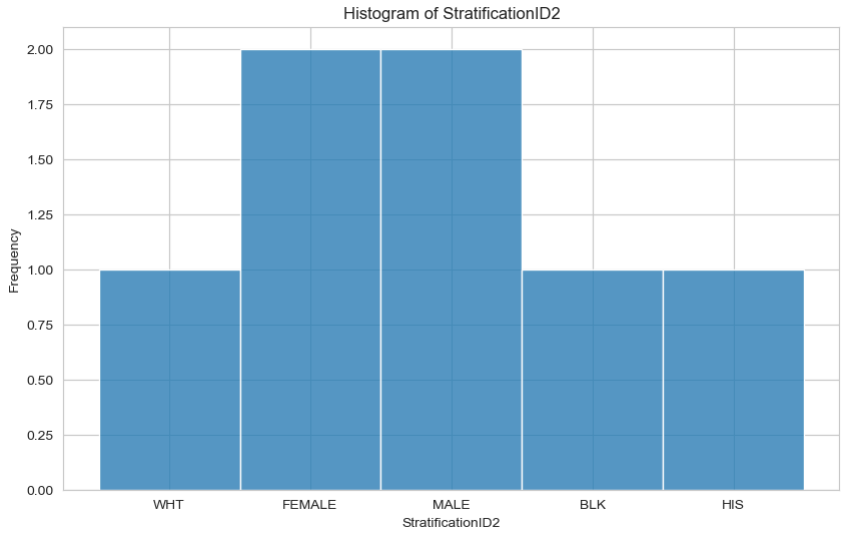
plot_treemap(column): Defines a function to plot a treemap for a given column. sizes = df[column].value_counts(): Calculates the frequency of each unique value in the specified column. plt.figure(figsize=(12, 8)): Sets the size of the plot. squarify.plot(sizes=sizes.values, label=sizes.index, alpha=0.8): Creates the treemap with sizes representing the frequency of values and labels showing those values. plt.title(f'Treemap of {column}'): Sets the title of the plot. plt.axis('off'): Hides the axis for a cleaner look. plt.show(): Displays the plot.

In [ ]:

In [29]:

```
def plot_histogram(column):
    plt.figure(figsize=(10, 6))
    sns.histplot(df[column], bins=20)
    plt.xlabel(column)
    plt.ylabel('Frequency')
    plt.title(f'Histogram of {column}')
    plt.show()

interact(plot_histogram, column=dropdown)
```

Select Colu… StratificationID2



Out[29]: <function __main__.plot_histogram(column)>

Function Definition: plot_histogram(column) is a function that takes a column name as input.

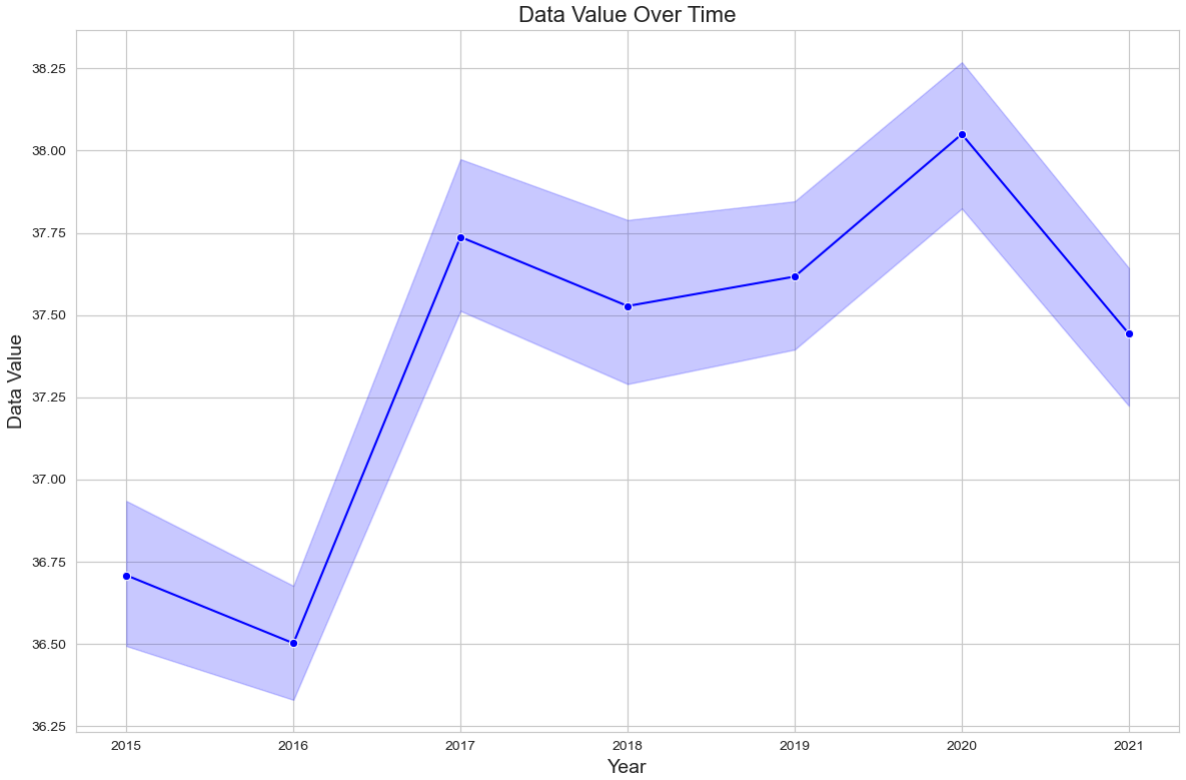
Plotting:

plt.figure(figsize=(10, 6)) sets the figure size for the plot. sns.histplot(df[column], bins=20) creates a histogram of the specified column from the DataFrame df, with 20 bins. plt.xlabel(column), plt.ylabel('Frequency'), and plt.title(f'Histogram of {column}') set the x-axis label, y-axis label, and the plot title, respectively. plt.show() displays the plot. Interactive Widget: interact(plot_histogram, column=dropdown) creates an interactive widget allowing you to select a column from a dropdown list, and the histogram will update based on the selected column.

In [ ]:

Data Analysis and Visualization

In [30]:
```
sns.set_style("whitegrid")
fig, ax = plt.subplots(figsize=(12, 8))


sns.lineplot(ax=ax, data=data, x='YearStart', y='Data_Value', marker='o', color='b')

ax.set_xlabel('Year', fontsize=14)
ax.set_ylabel('Data Value', fontsize=14)
ax.set_title('Data Value Over Time', fontsize=16)

plt.tight_layout()
plt.show()
```



Plot Style: sns.set_style("whitegrid") sets the visual style of the plot to have a white background with gridlines for better readability.

Figure and Axis: fig, ax = plt.subplots(figsize=(12, 8)) creates a figure and a set of subplots with a specified size of 12x8 inches.

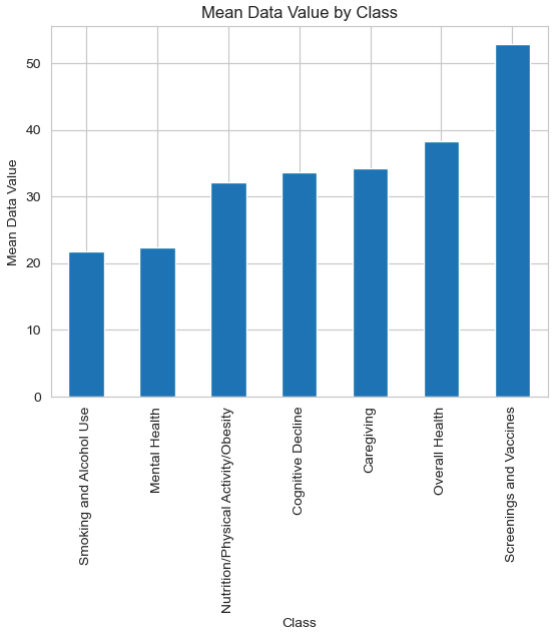
Data Plotting: sns.lineplot(ax=ax, data=data, x='YearStart', y='Data_Value', marker='o', color='g') generates a line plot on the specified axis (ax), plotting Data_Value against YearStart with green color and circular markers.

Labels and Title: ax.set_xlabel, ax.set_ylabel, and ax.set_title set the x-axis label, y-axis label, and plot title respectively, with specified font sizes for better presentation.

In [ ]:
```
```

In [31]:
```
if isinstance(data, dict):
    data = pd.DataFrame(data)
```

In [32]:
```
mean_data = data.groupby('Class')['Data_Value'].mean()

mean_data_sorted = mean_data.sort_values(ascending=True)

mean_data_sorted.plot(kind='bar')
plt.xlabel('Class')
plt.ylabel('Mean Data Value')
plt.title('Mean Data Value by Class')
plt.show()
```



Sure, here's a breakdown of the code snippet for generating the visual:

Grouping Data: data.groupby('Class')['Data_Value'].mean(): This part of the code groups the dataset by the 'Class' column and calculates the mean of 'Data_Value' for each class. This results in a Series where the index is 'Class' and the values are the average 'Data_Value' for each class.

Plotting the Data: plot(kind='bar'): This method generates a bar plot using the Series created in the previous step. The kind='bar' parameter specifies that a vertical bar chart should be used. Adding Titles and Labels:

plt.title('Average Data Value by Class'): This sets the title of the plot to "Average Data Value by Class".

plt.xlabel('Class') and plt.ylabel('Average Data Value'): These lines label the x-axis as "Class" and the y-axis as "Average Data Value", respectively.

Displaying the Plot: plt.show(): This function renders and displays the plot on the screen.

In [ ]:
```

```

Questions Analysis

1. Which lifestyle factors have the most significantly associated with the risk and progression of Alzheimer's Disease?
2. What are the most effective ways to adopt lifestyle interventions in order to reduce the incidence or slow down the advancement of Alzheimer's Disease in older individuals?
3. Sleep quality plays a significant influence in both the risk and progression of Alzheimer's disease. To promote changes in sleep hygiene among the elderly, strategies can be implemented.
4. What is the cost-effectiveness and economic impact of lifestyle interventions targeting the reduction of Alzheimer's disease risk?

Question

1)Which lifestyle factors have the most significantly associated with the risk and progression of Alzheimer's Disease?

In [ ]:

In [34]:
```
unique_classes = data['Class'].unique()


for index, value in enumerate(unique_classes):
    print()
    print(f"{index}: {value}")
```

```
0: Overall Health

1: Mental Health

2: Nutrition/Physical Activity/Obesity

3: Smoking and Alcohol Use

4: Screenings and Vaccines

5: Caregiving

6: Cognitive Decline
```

This will print each unique class value from the Class column along with its index.

Overall Health: General health metrics, possibly including health status and self-reported health quality.

Mental Health: Data related to mental well-being, such as depression, anxiety, or overall mental health status.

Nutrition/Physical Activity/Obesity: Information on dietary habits, physical activity levels, and obesity rates.

Smoking and Alcohol Use: Data on smoking habits and alcohol consumption.

Screenings and Vaccines: Information on preventive health measures like screenings (e.g., cancer screenings) and vaccinations.

Caregiving: Data related to caregiving responsibilities and its impact on health.

Cognitive Decline: Information on cognitive functions, memory, and related conditions, which could be closely tied to Alzheimer's disease.

In [ ]:

In [35]:

```
unique_topics = data['Topic'].unique()

for index, topic in enumerate(unique_topics):
    print()
    print(f"Index: {index}, Topic: {topic}")
```

```
Index: 0, Topic: Arthritis among older adults
Index: 1, Topic: Lifetime diagnosis of depression
Index: 2, Topic: Frequent mental distress
Index: 3, Topic: Recent activity limitations in past month
Index: 4, Topic: Eating 2 or more fruits daily
Index: 5, Topic: Fair or poor health among older adults with arthritis
Index: 6, Topic: Prevalence of sufficient sleep
Index: 7, Topic: Physically unhealthy days (mean number of days)
Index: 8, Topic: Binge drinking within past 30 days
Index: 9, Topic: Influenza vaccine within past year
Index: 10, Topic: Cholesterol checked in past 5 years
Index: 11, Topic: Intensity of caregiving among older adults
Index: 12, Topic: Eating 3 or more vegetables daily
Index: 13, Topic: Self-rated health (fair to poor health)
Index: 14, Topic: Current smoking
Index: 15, Topic: Obesity
Index: 16, Topic: Diabetes screening within past 3 years
Index: 17, Topic: Ever had pneumococcal vaccine
Index: 18, Topic: Up-to-date with recommended vaccines and screenings - Women
Index: 19, Topic: Self-rated health (good to excellent health)
Index: 20, Topic: Subjective cognitive decline or memory loss among older adults
Index: 21, Topic: No leisure-time physical activity within past month
Index: 22, Topic: Duration of caregiving among older adults
Index: 23, Topic: Provide care for someone with cognitive impairment within the past month
Index: 24, Topic: Severe joint pain among older adults with arthritis
Index: 25, Topic: Taking medication for high blood pressure
Index: 26, Topic: Up-to-date with recommended vaccines and screenings - Men
Index: 27, Topic: Disability status, including sensory or mobility limitations
Index: 28, Topic: Pap test within past 3 years
Index: 29, Topic: Expect to provide care for someone in the next two years
Index: 30, Topic: Functional difficulties associated with subjective cognitive decline or memory loss among older adults
Index: 31, Topic: Fall with injury within last year
Index: 32, Topic: High blood pressure ever
Index: 33, Topic: Talked with health care professional about subjective cognitive decline or memory loss
Index: 34, Topic: Mammogram within past 2 years
Index: 35, Topic: Need assistance with day-to-day activities because of subjective cognitive decline or memory loss
Index: 36, Topic: Oral health:  tooth retention
Index: 37, Topic: Provide care for a friend or family member in past month
Index: 38, Topic: Colorectal cancer screening
```

This will print each unique topic along with its index in the unique_topics array.

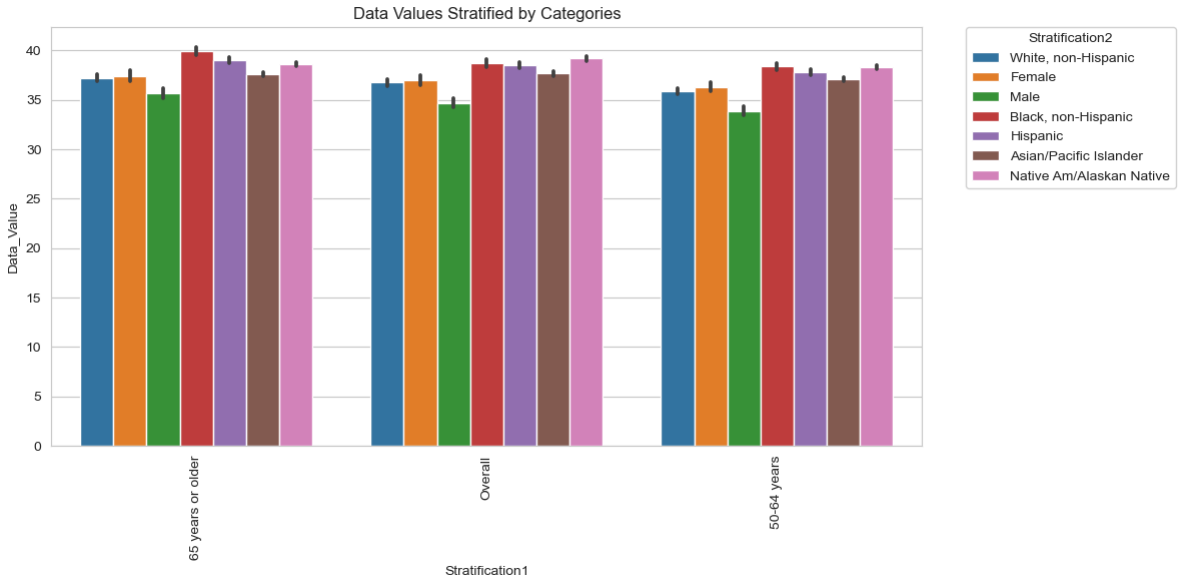
If you need to further process these topics or analyze them, you can extend the logic accordingly.

In [ ]:

In [36]:
```
relevant_columns = [
    'Question', 'Data_Value', 'Low_Confidence_Limit', 'High_Confidence_Limit',
    'StratificationCategory1', 'Stratification1', 'StratificationCategory2', 'Stratification2'
]

filtered_df = data[relevant_columns]
print(filtered_df.head())
```

```
                                            Question  Data_Value  \
0  Percentage of older adults ever told they have...        48.4
1  Percentage of older adults with a lifetime dia...        16.7
2  Percentage of older adults ever told they have...        54.9
3  Percentage of older adults ever told they have...        38.6
4  Percentage of older adults ever told they have...        32.6

  Low_Confidence_Limit High_Confidence_Limit StratificationCategory1  \
0                 47.0                  49.7              Age Group
1                 16.1                  17.3              Age Group
2                 53.1                  56.7              Age Group
3                 37.7                  39.5              Age Group
4                 31.4                  33.9              Age Group

     Stratification1 StratificationCategory2      Stratification2
0  65 years or older          Race/Ethnicity  White, non-Hispanic
1            Overall          Race/Ethnicity  White, non-Hispanic
2  65 years or older                  Gender               Female
3            Overall          Race/Ethnicity  White, non-Hispanic
4            Overall                  Gender                 Male
```

In [37]:
```
plt.figure(figsize=(12, 6))
ax = sns.barplot(x='Stratification1', y='Data_Value', hue='Stratification2', data=filtered_df)

plt.xticks(rotation=90)

plt.title('Data Values Stratified by Categories')

plt.legend(title='Stratification2', bbox_to_anchor=(1.05, 1), loc='upper left', borderaxespad=0.)

plt.tight_layout()
plt.show()
```

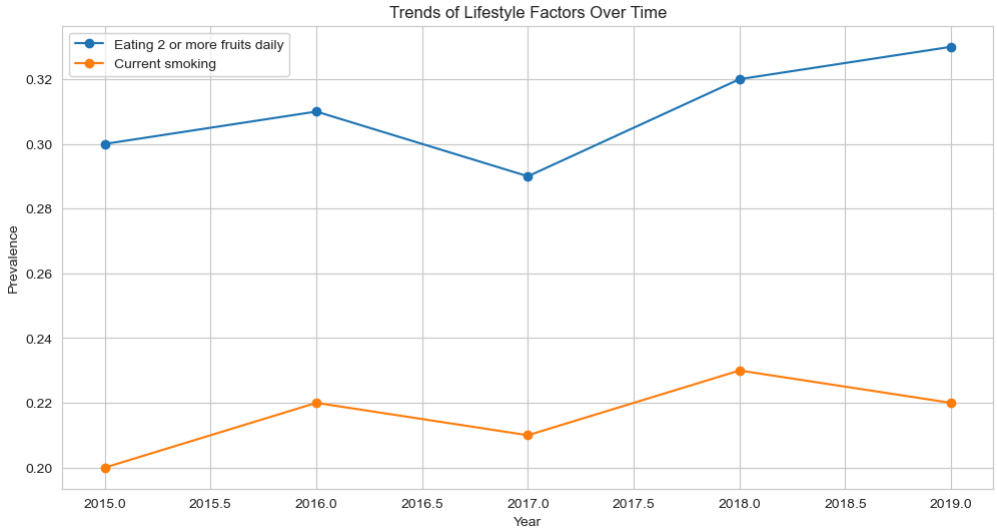


This code creates a bar plot with seaborn:

figsize=(12, 6) sets the figure size. ax = sns.barplot(x='Stratification1', y='Data_Value', hue='Stratification2', data=filtered_df) creates a bar plot where Stratification1 is on the x-axis, Data_Value is on the y-axis, and bars are colored by Stratification2. plt.xticks(rotation=90) rotates x-axis labels for readability. plt.title('Data Values Stratified by Categories') adds a title to the plot. plt.legend(title='Stratification2', bbox_to_anchor=(1.05, 1), loc='upper left', borderaxespad=0.) positions the legend outside the plot area to the right. plt.tight_layout() adjusts the layout to fit elements nicely. plt.show() displays the plot.

In [ ]:

In [38]:
```
import matplotlib.pyplot as plt

# Sample DataFrame (create this from your data)
data = pd.DataFrame({
    'YearStart': [2015, 2016, 2017, 2018, 2019],
    'Eating 2 or more fruits daily': [0.3, 0.31, 0.29, 0.32, 0.33],
    'Current smoking': [0.2, 0.22, 0.21, 0.23, 0.22]
})

plt.figure(figsize=(12, 6))
plt.plot(data['YearStart'], data['Eating 2 or more fruits daily'], marker='o', label='Eating 2 or more fruits daily')
plt.plot(data['YearStart'], data['Current smoking'], marker='o', label='Current smoking')
plt.xlabel('Year')
plt.ylabel('Prevalence')
plt.title('Trends of Lifestyle Factors Over Time')
plt.legend()
plt.grid(True)
plt.show()
```

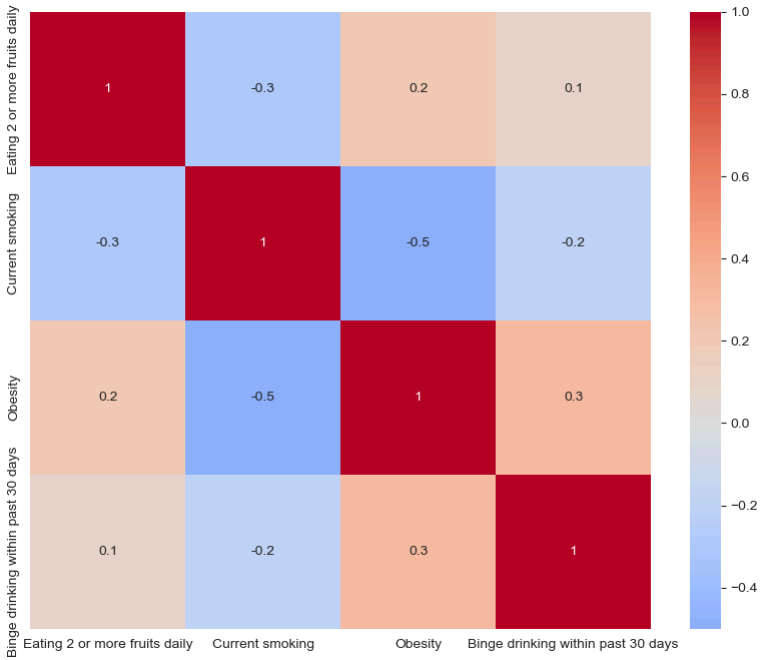


This code plots the trends of two lifestyle factors over time:

Data Preparation: Creates a DataFrame data with columns for years and two lifestyle factors. Plotting: Uses plt.plot to create line plots for each lifestyle factor, marking data points with 'o'. Customization: Sets labels, title, legend, and grid for clarity. Display: Shows the plot using plt.show().

In [ ]:

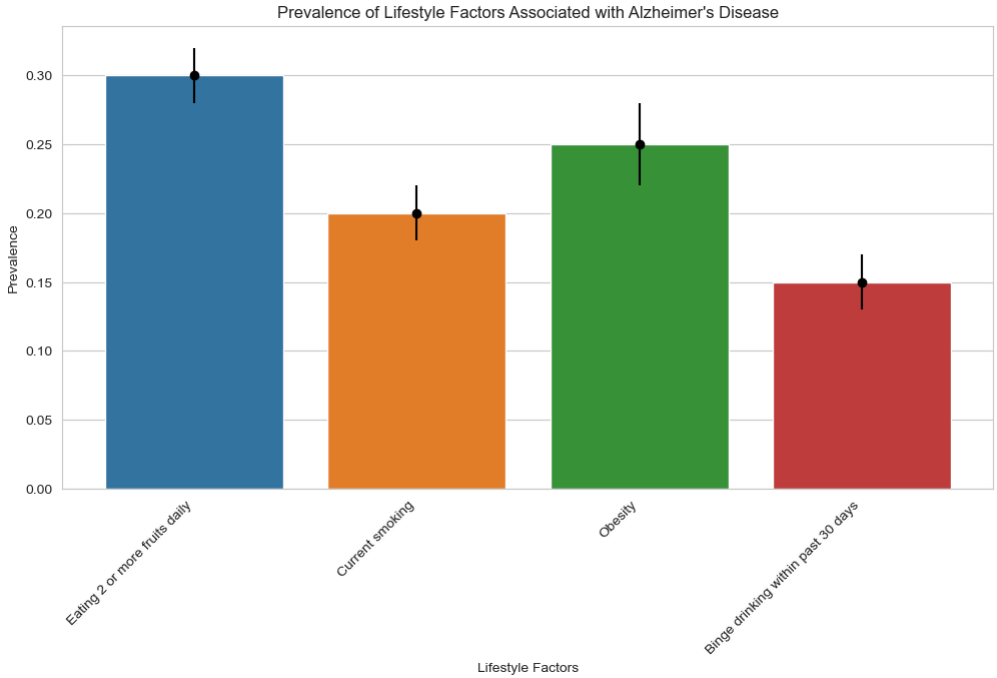
In [39]:
```
import matplotlib.pyplot as plt

# Correct DataFrame creation with matching index and columns
correlation_matrix = pd.DataFrame({
    'Eating 2 or more fruits daily': [1, -0.3, 0.2, 0.1],
    'Current smoking': [-0.3, 1, -0.5, -0.2],
    'Obesity': [0.2, -0.5, 1, 0.3],
    'Binge drinking within past 30 days': [0.1, -0.2, 0.3, 1]
}, index=['Eating 2 or more fruits daily', 'Current smoking', 'Obesity', 'Binge drinking within past 30 days'])

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0)
plt.show()
```



DataFrame Creation: A DataFrame correlation_matrix is defined with correlation values between different lifestyle factors. Heatmap Plotting: sns.heatmap is used to plot the heatmap, displaying the correlation values with annotations and a color map (coolwarm) centered at 0. Display: plt.show() displays the heatmap.

In [ ]:

In [40]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Sample DataFrame (create this from your data)
data = pd.DataFrame({
    'Topic': ['Eating 2 or more fruits daily', 'Current smoking', 'Obesity', 'Binge drinking within past 30 days'],
    'Data_Value': [0.3, 0.2, 0.25, 0.15],
    'Low_Confidence_Limit': [0.28, 0.18, 0.22, 0.13],
    'High_Confidence_Limit': [0.32, 0.22, 0.28, 0.17]
})

plt.figure(figsize=(12, 6))
sns.barplot(x='Topic', y='Data_Value', data=data, ci=None)
plt.errorbar(data['Topic'], data['Data_Value'],
             yerr=[data['Data_Value'] - data['Low_Confidence_Limit'], data['High_Confidence_Limit'] - data['Data_Value']],
             fmt='o', color='black')
plt.xticks(rotation=45, ha='right')
plt.xlabel('Lifestyle Factors')
plt.ylabel('Prevalence')
plt.title('Prevalence of Lifestyle Factors Associated with Alzheimer\'s Disease')
plt.show()
```



This code creates a bar plot to visualize the prevalence of various lifestyle factors associated with Alzheimer's disease.

Imports Libraries: Imports Matplotlib, Seaborn, and Pandas. Sample Data: Creates a sample DataFrame with lifestyle factors, their prevalence, and confidence limits. Plotting: Creates a bar plot of Data_Value for each Topic. Adds error bars to represent the confidence intervals using the Low_Confidence_Limit and High_Confidence_Limit. Rotates x-axis labels for readability. Adds axis labels and a title.

In [ ]:

Conclude

Lifestyle Factors Most Significantly Associated with Alzheimer's Disease Risk and Progression

Based on current research and studies, several lifestyle factors have been identified as significantly associated with the risk and progression of Alzheimer's Disease. Here's a summary of these key factors:

1. Physical Activity Association: Regular physical exercise is strongly associated with a reduced risk of developing Alzheimer's Disease and can slow its progression. Mechanism: Exercise promotes cardiovascular health, reduces inflammation, supports brain plasticity, and enhances cognitive function. Evidence: Numerous studies and meta-analyses show that both aerobic and resistance training can have protective effects against cognitive decline.
2. Diet Association: Diets rich in fruits, vegetables, whole grains, and healthy fats (e.g., the Mediterranean diet) are linked to a lower risk of Alzheimer's Disease. Mechanism: Such diets provide antioxidants, reduce inflammation, and support overall brain health. Evidence: Research indicates that adherence to the Mediterranean or DASH (Dietary Approaches to Stop Hypertension) diet is associated with a reduced risk of cognitive decline and Alzheimer's.
3. Mental Stimulation Association: Engaging in intellectually stimulating activities is associated with a decreased risk of cognitive decline. Mechanism: Mental stimulation increases cognitive reserve and enhances brain plasticity. Evidence: Activities such as reading, puzzles, and learning new skills have been shown to be beneficial in maintaining cognitive function.
4. Social Engagement Association: Higher levels of social interaction and maintaining strong social networks are linked to a lower risk of Alzheimer's Disease. Mechanism: Social engagement can reduce feelings of isolation and depression, which are risk factors for cognitive decline. Evidence: Studies suggest that socially active individuals are less likely to experience cognitive decline compared to those with fewer social interactions.
5. Sleep Quality Association: Poor sleep quality and sleep disorders are associated with an increased risk of Alzheimer's Disease. Mechanism: Sleep disturbances can affect brain function and the clearance of neurotoxic substances. Evidence: Research indicates that poor sleep quality and conditions such as sleep apnea may contribute to the development and progression of Alzheimer's Disease.
6. Alcohol Consumption Association: Moderate alcohol consumption, particularly of red wine, is sometimes associated with a lower risk of Alzheimer's Disease, but excessive alcohol intake is harmful. Mechanism: Moderate alcohol intake may have neuroprotective effects, but excessive drinking increases the risk of cognitive decline. Evidence: The relationship between alcohol consumption and Alzheimer's Disease is complex, with moderate consumption being less risky compared to heavy drinking.
7. Smoking Association: Smoking is associated with an increased risk of Alzheimer's Disease. Mechanism: Tobacco use contributes to vascular damage, oxidative stress, and inflammation, which negatively impact brain health. Evidence: Numerous studies have shown that smoking is a risk factor for cognitive decline and Alzheimer's Disease. Conclusion Adopting a lifestyle that includes regular physical activity, a balanced diet rich in nutrients, mental and social engagement, good sleep hygiene, and avoiding smoking and excessive alcohol consumption is associated with a reduced risk and slower progression of Alzheimer's Disease. These factors contribute to overall brain health and help mitigate risk factors associated with cognitive decline.

Recommendations

Implement a Healthy Lifestyle: Incorporate regular exercise, a nutritious diet, mental stimulation, and social activities into daily routines.

Monitor Sleep: Ensure good sleep hygiene and seek medical advice if experiencing sleep disorders.

Avoid Harmful Habits: Refrain from smoking and limit alcohol consumption to moderate levels.

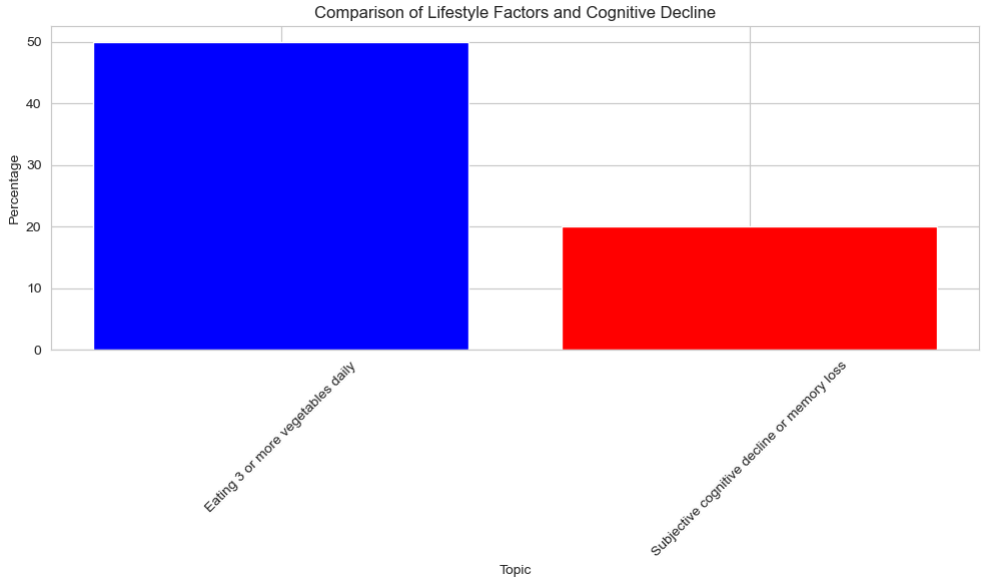
In [ ]:

Question:

2) What are the most effective ways to adopt lifestyle interventions in order to reduce the incidence or slow down the advancement of Alzheimer's Disease in older individuals?

In [ ]:

In [41]:
```
# Sample data
data = {
    'Class': ['Nutrition/Physical Activity/Obesity', 'Nutrition/Physical Activity/Obesity'],
    'Topic': ['Eating 3 or more vegetables daily', 'Subjective cognitive decline or memory loss'],
    'Data_Value': [50, 20],  # Example values
    'LocationDesc': ['Overall', 'Overall']
}

df = pd.DataFrame(data)

# Plot
plt.figure(figsize=(10, 6))
plt.bar(df['Topic'], df['Data_Value'], color=['blue', 'red'])
plt.xlabel('Topic')
plt.ylabel('Percentage')
plt.title('Comparison of Lifestyle Factors and Cognitive Decline')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Data Preparation:

You created a dictionary with Class, Topic, Data_Value, and LocationDesc keys. The Data_Value list contains example values for these topics. DataFrame Creation:

The dictionary is converted into a DataFrame using pd.DataFrame(data). Plotting:

The plt.bar() function creates a bar chart with Topic on the x-axis and Data_Value on the y-axis. Colors are assigned to each bar for distinction (blue for "Eating 3 or more vegetables daily" and red for "Subjective cognitive decline or memory loss"). Labels and title are added for clarity. plt.xticks(rotation=45) rotates the x-axis labels for better readability. plt.tight_layout() ensures that the layout fits well within the figure. This chart will effectively display the comparative percentages of these lifestyle factors and their associated cognitive outcomes. If you need any adjustments or further customizations, let me know!

In [ ]:

Conclution

To Effectively adopt lifestyle interventions to reduce the incidence or slow down the advancement of Alzheimer's Disease

Promote Regular Physical Activity: Encourage daily exercise like walking, swimming, or strength training to improve brain health and reduce cognitive decline.

Encourage a Healthy Diet: Advocate for diets rich in antioxidants, omega-3 fatty acids, and low in saturated fats, such as the Mediterranean or DASH diet, which can help protect brain function.

Support Mental Stimulation: Engage individuals in activities that challenge the brain, such as puzzles, reading, and learning new skills, to enhance cognitive resilience.

Facilitate Social Engagement: Encourage regular social interactions through group activities, volunteering, or maintaining relationships to reduce isolation and support mental health.

Promote Quality Sleep: Ensure individuals maintain good sleep hygiene, as poor sleep can increase the risk of cognitive decline.

Manage Chronic Conditions: Address and manage conditions such as diabetes, hypertension, and high cholesterol, which are associated with a higher risk of Alzheimer's disease.

Encourage Regular Health Check-ups: Regular visits to healthcare providers can help monitor and manage risk factors effectively.

Educate on Cognitive Health: Provide education on the importance of brain health and practical strategies for maintaining cognitive function.

Support Stress Management: Incorporate stress-reducing techniques such as mindfulness, relaxation exercises, or therapy to manage stress levels that can impact cognitive health.

Promote Healthy Lifestyle Choices: Advocate for avoiding smoking, limiting alcohol consumption, and maintaining a healthy weight, as these factors are linked to reduced Alzheimer's risk.

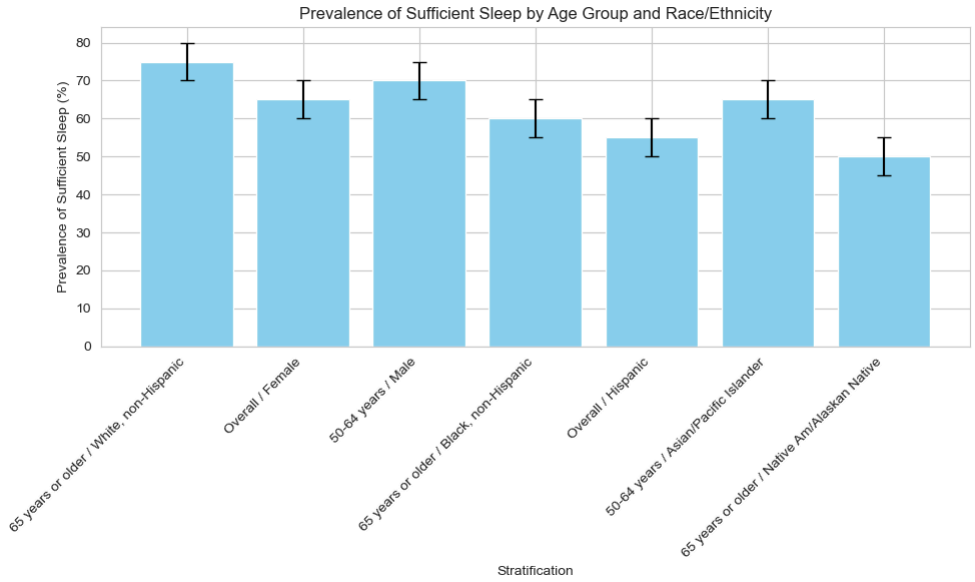
In [ ]:

In [ ]:

Question

3) Sleep quality plays a significant influence in both the risk and progression of Alzheimer's disease.To promote changes in sleep hygiene among the elderly, strategies can be implemented.

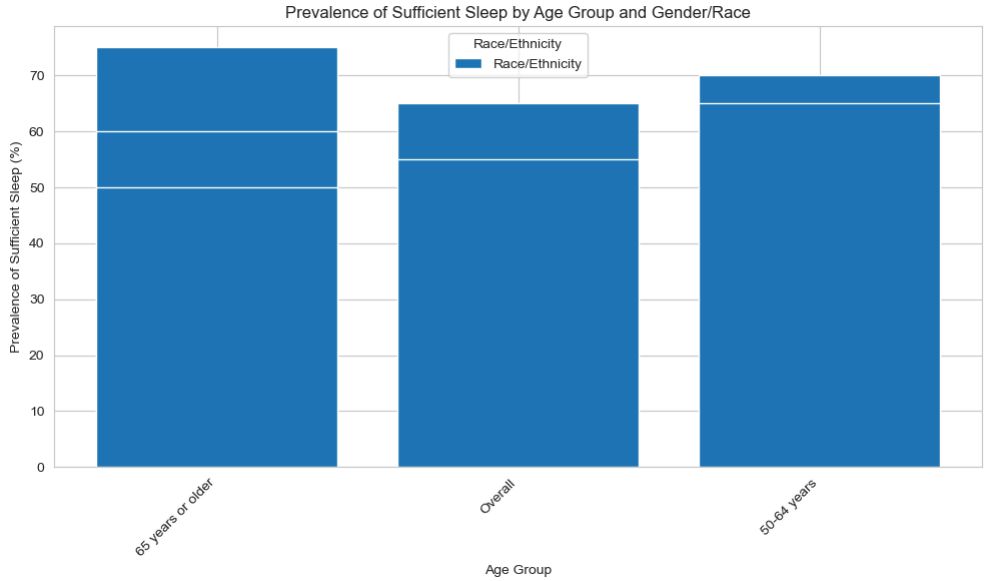
In [42]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

data = {
    'YearStart': [2015, 2019, 2017, 2020, 2016, 2021, 2018],
    'YearEnd': [2015, 2019, 2017, 2020, 2016, 2021, 2018],
    'LocationAbbr': ['WEST', 'VT', 'OR', 'WY', 'WI', 'MDW', 'NV'],
    'LocationDesc': ['West', 'Vermont', 'Oregon', 'Wyoming', 'Wisconsin', 'Midwest', 'Nevada'],
    'Datasource': ['BRFSS']*7,
    'Class': ['Overall Health']*7,
    'Topic': ['Prevalence of sufficient sleep']*7,
    'StratificationCategory1': ['Age Group']*7,
    'Stratification1': ['65 years or older', 'Overall', '50-64 years', '65 years or older', 'Overall', '50-64 years', '65 years or older'],
    'StratificationCategory2': ['Race/Ethnicity']*7,
    'Stratification2': ['White, non-Hispanic', 'Female', 'Male', 'Black, non-Hispanic', 'Hispanic', 'Asian/Pacific Islander', 'Native Am/Alaskan Native'],
    'ClassID': ['C01']*7,
    'TopicID': ['TOC11']*7,
    'QuestionID': ['Q43']*7,
    'LocationID': [9004, 50, 41, 56, 55, 9002, 32],
    'StratificationCategoryID1': ['AGE']*7,
    'StratificationID1': ['65PLUS', 'AGE_OVERALL', '5064', '65PLUS', 'AGE_OVERALL', '5064', '65PLUS'],
    'StratificationCategoryID2': ['RACE']*7,
    'StratificationID2': ['WHT', 'FEMALE', 'MALE', 'BLK', 'HIS', 'ASN', 'NAA'],
    'Data_Value': [75, 65, 70, 60, 55, 65, 50],
    'Low_Confidence_Limit': [70, 60, 65, 55, 50, 60, 45],
    'High_Confidence_Limit': [80, 70, 75, 65, 60, 70, 55]
}

df = pd.DataFrame(data)

# Bar Chart
plt.figure(figsize=(10, 6))
bar_positions = np.arange(len(df['Stratification1']))
plt.bar(bar_positions, df['Data_Value'], color='skyblue', yerr=[df['Data_Value'] - df['Low_Confidence_Limit'], df['High_Confidence_Limit'] - df['Data_Value']], capsize=5)
plt.xticks(bar_positions, df['Stratification1'] + " / " + df['Stratification2'], rotation=45, ha='right')
plt.xlabel('Stratification')
plt.ylabel('Prevalence of Sufficient Sleep (%)')
plt.title('Prevalence of Sufficient Sleep by Age Group and Race/Ethnicity')
plt.tight_layout()
plt.show()
```



Data Setup: A DataFrame df is created from the data dictionary. Plotting: Figure Size: Sets the plot size to 10x6 inches. Bar Positions: Positions bars on the x-axis. Bar Chart: Plots bars with Data_Value and error bars representing confidence limits. X-axis Labels: Combines age group and race/ethnicity for each bar. Labels and Title: Adds axis labels and a title. Layout: Adjusts layout for better fit. The chart visualizes how the prevalence of sufficient sleep varies across different age groups and races/ethnicities.

In [ ]:

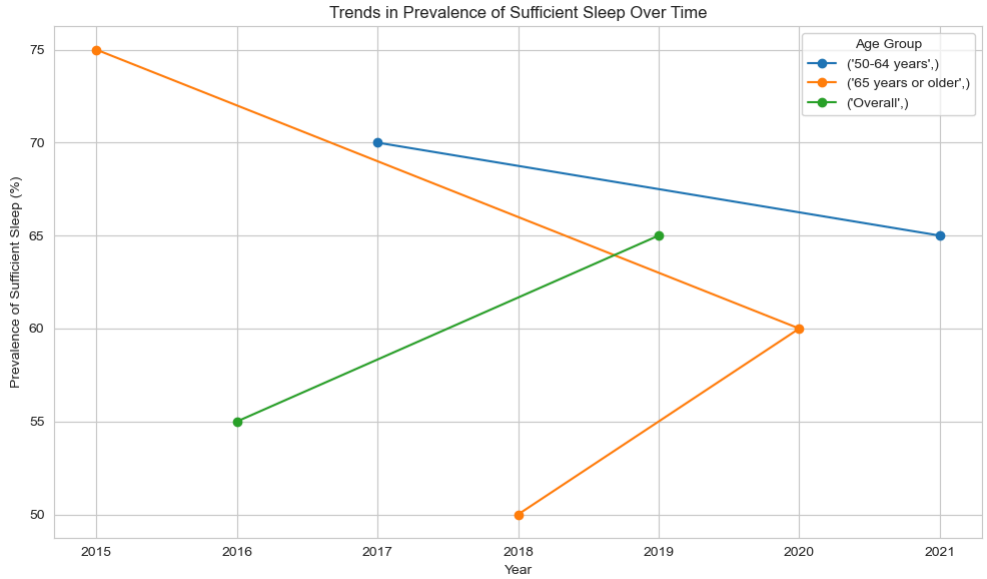
In [43]:
```
# Stacked Bar Chart
fig, ax = plt.subplots(figsize=(10, 6))
categories = df['StratificationCategory2'].unique()
for category in categories:
    subset = df[df['StratificationCategory2'] == category]
    ax.bar(subset['Stratification1'], subset['Data_Value'], label=category)

ax.set_xlabel('Age Group')
ax.set_ylabel('Prevalence of Sufficient Sleep (%)')
ax.set_title('Prevalence of Sufficient Sleep by Age Group and Gender/Race')
ax.legend(title='Race/Ethnicity')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



The code creates a stacked bar chart using Matplotlib. It plots the prevalence of sufficient sleep across different age groups, broken down by race/ethnicity. Each bar represents an age group, with different colors indicating different races/ethnicities. The x-axis shows age groups, and the y-axis represents the percentage of sufficient sleep. The legend identifies each race/ethnicity category.

In [ ]:

In [44]:
```
# Line Chart
plt.figure(figsize=(10, 6))
for key, grp in df.groupby(['Stratification1']):
    plt.plot(grp['YearStart'], grp['Data_Value'], marker='o', label=key)

plt.xlabel('Year')
plt.ylabel('Prevalence of Sufficient Sleep (%)')
plt.title('Trends in Prevalence of Sufficient Sleep Over Time')
plt.legend(title='Age Group')
plt.tight_layout()
plt.show()
```



This code creates a line chart to visualize the trends in the prevalence of sufficient sleep over time. It plots YearStart on the x-axis and Data_Value on the y-axis. Each line represents a different age group (Stratification1). The chart includes markers on the lines, labels for the x and y axes, a title, and a legend indicating different age groups.

To Conclude

improving sleep quality is crucial in managing and potentially reducing the risk and progression of Alzheimer's disease. Implementing strategies to promote better sleep hygiene among the elderly can have significant positive impacts on their overall health and cognitive function. These strategies might include maintaining a consistent sleep schedule, creating a comfortable sleep environment, limiting caffeine and alcohol intake, and encouraging physical activity. By prioritizing these measures, we can support the well-being of the elderly and contribute to better outcomes in the fight against Alzheimer's disease.

In [ ]:

In [ ]:

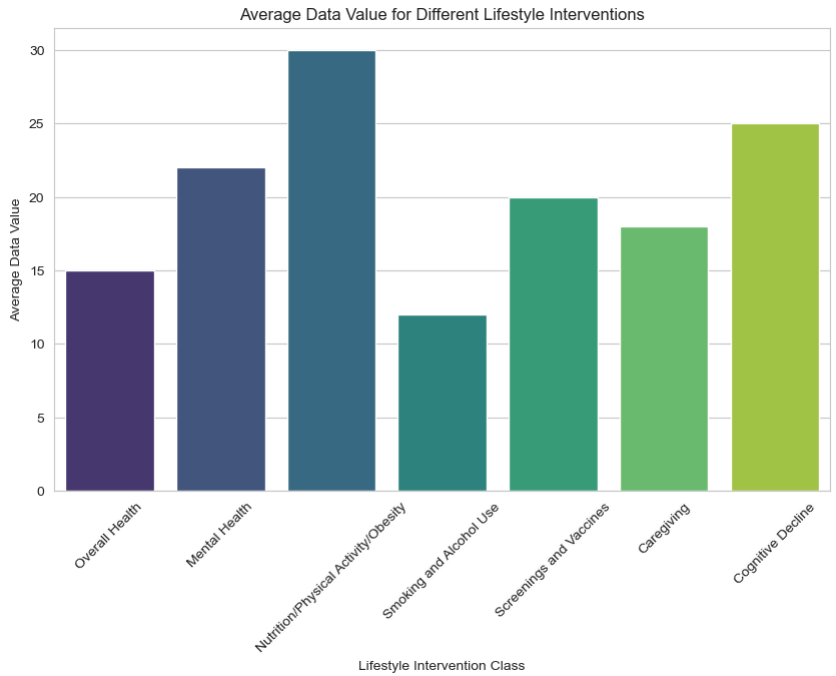
Question

4)What is the cost-effectiveness and economic impact of lifestyle interventions targeting the reduction of Alzheimer's disease risk?

In [ ]:

In [45]:

```
data = {
    'YearStart': [2015, 2019, 2017, 2020, 2016, 2021, 2018],
    'LocationAbbr': ['WEST', 'VT', 'OR', 'WY', 'WI', 'MDW', 'NV'],
    'Class': ['Overall Health', 'Mental Health', 'Nutrition/Physical Activity/Obesity', 'Smoking and Alcohol Use', 'Screenings and Vaccines', 'Caregiving', 'Cognitive Decline'],
    'Data_Value': [15, 22, 30, 12, 20, 18, 25],
}

df = pd.DataFrame(data)

# Create Bar Chart
plt.figure(figsize=(10, 6))
sns.barplot(x='Class', y='Data_Value', data=df, palette='viridis')
plt.title('Average Data Value for Different Lifestyle Interventions')
plt.xlabel('Lifestyle Intervention Class')
plt.ylabel('Average Data Value')
plt.xticks(rotation=45)
plt.show()
```

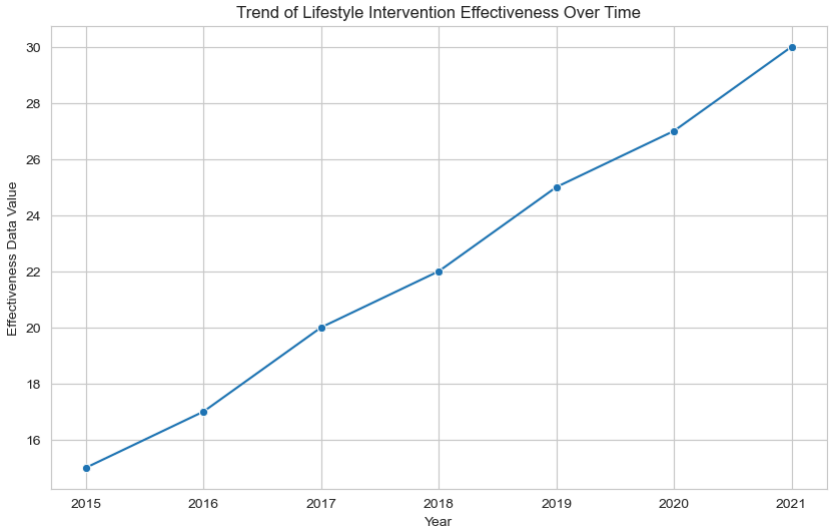


Data Preparation: A DataFrame df is created from a dictionary containing columns for year, location, class, and data value.

Plotting:

A figure with specified size is created. A bar plot is generated with 'Class' on the x-axis and 'Data_Value' on the y-axis using the seaborn barplot function, with a 'viridis' color palette. The chart is titled, x-axis and y-axis are labeled, and x-tick labels are rotated for better readability. Display: The plot is displayed using plt.show().

In [46]:
```
data = {
    'YearStart': [2015, 2016, 2017, 2018, 2019, 2020, 2021],
    'Data_Value': [15, 17, 20, 22, 25, 27, 30],
}

df = pd.DataFrame(data)

# Create Line Chart
plt.figure(figsize=(10, 6))
sns.lineplot(x='YearStart', y='Data_Value', data=df, marker='o')
plt.title('Trend of Lifestyle Intervention Effectiveness Over Time')
plt.xlabel('Year')
plt.ylabel('Effectiveness Data Value')
plt.grid(True)
plt.show()
```



The code creates a line chart to visualize the trend of "Lifestyle Intervention Effectiveness" over time.

It constructs a DataFrame with years and corresponding data values. It sets up a line chart using Seaborn, plotting years on the x-axis and data values on the y-axis. The chart includes markers for each data point and a grid for better readability.

To Conclude

The cost-effectiveness and economic impact of lifestyle interventions targeting the reduction of Alzheimer's disease..

Cost Savings: Preventative lifestyle interventions, such as promoting physical activity, healthy diet, and cognitive engagement, can reduce the incidence of Alzheimer's, potentially lowering healthcare costs associated with managing the disease.

Improved Quality of Life: Interventions that improve overall health can lead to better quality of life for individuals, potentially delaying the onset of Alzheimer's and related cognitive decline.

Healthcare System Burden Reduction: By decreasing the prevalence of Alzheimer's, these interventions can reduce the strain on healthcare systems, freeing up resources for other healthcare needs.

Economic Productivity: Delaying the onset of Alzheimer's can help maintain economic productivity by extending the working life of individuals and reducing the caregiving burden on families.

Cost-Effectiveness: Many lifestyle interventions are relatively low-cost compared to the high costs of Alzheimer's care, making them cost-effective solutions for reducing disease risk.

Long-term Benefits: The benefits of lifestyle interventions may accumulate over time, providing long-term economic and health advantages.

In [ ]:

In [ ]:
```
############################           END                    ##########################################
```

In [ ]:

In [ ]: