PROBLEM STATEMENT

Throughout the years, there have been many artificial intelligence (AI) and machine learning (ML) applications towards the detection of early stages of Alzheimer's disease. There is also research and interest in finding how impactful Alzheimer's disease can be in the presence of other diseases, or even during more severe situations, such as an epidemic or pandemic. This project will focus on and explore datasets that relate to Alzheimer's disease and dementia, as well as their impact on other features within the given datasets.

The main goal and motiviation of this final project has two parts (consisting of two scenarios). For the first scenario, the goal is to determine if Alzheimer's disease is an effective factor with data related to the COVID19 pandemic (when tested with different predicting factors). As part of the second scenario of the project, the goal is more specific, which is to see if there is a strong correlation between age and gender features among individuals who have Alzheimer's disease or dementia (and determining if one can improve model accuracy for an applied machine learning model for this scenario). Results from the first objective will help researchers understand the level of impact that Alzheimer's disease can potentially have in the case of future pandemics. If the second part is acheivable, this will be beneficial for future research when selecting key features to use when analyzing Alzheimer's disease from other datasets.

To achieve both parts of the objective, a multilinear regression model is applied for the first scenario and an applied logistic regression model handles the second scenario. Multilinear regression helps in the first scenario when testing Alzheimer's Disease with other features from the dataset (in this case, most of the other features are represented by other diseases that took place during the COVID19 pandemic). Logistic regression is used for the second case, since one of the column features (gender) for the second scenario consists of categorical data and (in this case) results to one of two values (this helps in simplifying the classication process, since this will mainly be binary classification).

DATA SOURCES

The data sources for this final project include the following sources listed below. Scenario one utilizes federal data from the U.S. Department of Health and Human Services (this dataset is available from a public source). This dataset focuses on the provisional counts of deaths (on a national level) by month between the years of 2020 and 2023, during the time of the COVID19 pandemic.

For scenario two, one of the datasets is also federal data (available from a public source) from the Centers for Disease Control and Prevention. Similar to the dataset for scenario one in terms of subject matter, this dataset contains data for contributing conditions to COVID19-related deaths (which is categorized by age group and gender). Two additional datasets for scenario two come from the Open Access Series of Imaging Studies (OASIS)

Brains project. Both datasets focus on different MRI characteristics of patients and individuals, divided into two categories: demented and nondemented. The final dataset for scenario two is from Kaggle, and it is publicly accessible. This dataset includes information on Alzheimer features identified in a select group of individuals for a case study. Background information on individuals (age, education level, gender) is included as well as specific medical data such as CDR (Clinical Dementia Ratining) and ASF (Atlas Scaling Factor).

1.) U.S. Department of Health and Human Services, Centers for Disease Control and Prevention (2021). Monthly Counts of Deaths by Select Causes (2020-2021) [Data set]. Retrieved from https://catalog.data.gov/dataset/monthly-counts-of-deaths-by-select-causes-2020-2021-2785a

2.) Centers for Disease Control and Prevention. (2021). Conditions Contributing to Deaths Involving Coronavirus Disease 2019 (COVID-19) by Age Group [Data set]. Retrieved from https://catalog.data.gov/dataset/conditions-contributing-to-deaths-involving-coronavirus-disease-2019-covid-19-by-age-group

3.) Open Access Series of Imaging Studies (OASIS) Brains Project. Open Access Series of Imaging Studies (OASIS): Cross-sectional MRI Data in Young, Middle Aged, Nondemented, and Demented Older Adults (2007) [Data set]. Retrieved from https://www.oasis-brains.org/#oasis1

4.) Open Access Series of Imaging Studies (OASIS) Brains Project. Open Access Series of Imaging Studies (OASIS): Longitudinal MRI Data in Nondemented and Demented Older Adults (2010) [Data set]. Retrieved from https://www.oasis-brains.org/#oasis2

5.) Dincer, Baris (2022). Alzheimer Features (2022) [Data set]. Retrieved from https://www.kaggle.com/datasets/brsdincer/alzheimer-features

DATASET DESCRIPTION

The monthly provisional dataset for scenario one is in a tabulated data format with features that include the following: provisional counts of deaths by cause of death, types of causes of death (natural causes, accidents, etc.), dates of deaths, and jurisdiction of where deaths occurred. For column size and row size, these values turn out to be 31 and 42 respectively.

The 'contributing conditions' dataset for scenario two is in a tabulated data format. For column size and row size, these values turn out to be 14 and 583740 respectively.

The OASIS cross-sectional MRI dataset for scenario two is in a tabulated data format. For column size and row size, these values turn out to be 12 and 436 respectively.

The OASIS longitudinal demographics dataset for scenario two is in a tabulated data format. For column size and row size, these values turn out to be 15 and 373 respectively.

The 'alzheimer features' dataset for scenario two is in a tabulated data format. For column size and row size, these values turn out to be 10 and 373 respectively.

GITHUB REPOSITORY

The Github repository link for this final project is displayed below (to run this code and produce the proper results, make sure to follow the instructions in the README file in the Github repository):

https://github.com/IsraelsLibrary/DTSA_5509_Intro_to_Machine_Learning

FINAL PROJECT CODE

Below is the code for the DTSA 5509 Final Project.

In [58]:
```
# Importing required Python libraries for the final project (primarily, plottin
# dataframe libraries)

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import math
import itertools
```

In [59]:
```
# Importing the required datasets for further processing

contributing_conditions_data = pd.read_csv('data/Conditions_Contributing_to_COV
oasis_demographics_data = pd.read_excel('data/oasis_longitudinal_demographics.x
oasis_cross_sectional_data = pd.read_csv('data/oasis_cross-sectional.csv')
alzheimer_data = pd.read_csv('data/alzheimer.csv')
monthly_provisional_data = pd.read_csv('data/Monthly_Provisional_Counts_of_Deat
```

In [60]:
```
# datasets for scenario 1:
## 1.) monthly_provisional_data

# datasets for scenario 2:
## 1.) contributing_conditions_data
## 2.) oasis_demographics_data
## 3.) oasis_cross_sectional_data
## 4.) alzheimer_data
```

DATA CLEANING PROCESS

Upon further inspection of the monthly provisional dataset for scenario one, I discovered many columns that contained null values as well as missing entries. To handle this data, I established a four percent null value threshold to determine which features to drop and which features to impute. I applied the null value threshold for each column feature in the monthly provisional dataset.

In the case of datasets for scenario two, I created two 'helper' functions that removes any rows in the dataset that contains 'nan' values or null values. This process only occurs after data transformations take place in forming the dataset for scenario two (which is done by a third 'helper' function).

In [61]:
```
# A printout of the dataset for the first scenario, to show columns that contai
print('monthly_provisional_data')
print(monthly_provisional_data)
```

```
monthly_provisional_data
     Data As Of  Start Date    End Date Jurisdiction of Occurrence  Year  \
0   06/21/2023  01/01/2020  01/31/2020              United States  2020
1   06/21/2023  02/01/2020  02/29/2020              United States  2020
2   06/21/2023  03/01/2020  03/31/2020              United States  2020
3   06/21/2023  04/01/2020  04/30/2020              United States  2020
4   06/21/2023  05/01/2020  05/31/2020              United States  2020
5   06/21/2023  06/01/2020  06/30/2020              United States  2020
6   06/21/2023  07/01/2020  07/31/2020              United States  2020
7   06/21/2023  08/01/2020  08/31/2020              United States  2020
8   06/21/2023  09/01/2020  09/30/2020              United States  2020
9   06/21/2023  10/01/2020  10/31/2020              United States  2020
10  06/21/2023  11/01/2020  11/30/2020              United States  2020
11  06/21/2023  12/01/2020  12/31/2020              United States  2020
12  06/21/2023  01/01/2021  01/31/2021              United States  2021
13  06/21/2023  02/01/2021  02/28/2021              United States  2021
14  06/21/2023  03/01/2021  03/31/2021              United States  2021
15  06/21/2023  04/01/2021  04/30/2021              United States  2021
16  06/21/2023  05/01/2021  05/31/2021              United States  2021
17  06/21/2023  06/01/2021  06/30/2021              United States  2021
18  06/21/2023  07/01/2021  07/31/2021              United States  2021
19  06/21/2023  08/01/2021  08/31/2021              United States  2021
20  06/21/2023  09/01/2021  09/30/2021              United States  2021
21  06/21/2023  10/01/2021  10/31/2021              United States  2021
22  06/21/2023  11/01/2021  11/30/2021              United States  2021
23  06/21/2023  12/01/2021  12/31/2021              United States  2021
24  06/21/2023  01/01/2022  01/31/2022              United States  2022
25  06/21/2023  02/01/2022  02/28/2022              United States  2022
26  06/21/2023  03/01/2022  03/31/2022              United States  2022
27  06/21/2023  04/01/2022  04/30/2022              United States  2022
28  06/21/2023  05/01/2022  05/31/2022              United States  2022
29  06/21/2023  06/01/2022  06/30/2022              United States  2022
30  06/21/2023  07/01/2022  07/31/2022              United States  2022
31  06/21/2023  08/01/2022  08/31/2022              United States  2022
32  06/21/2023  09/01/2022  09/30/2022              United States  2022
33  06/21/2023  10/01/2022  10/31/2022              United States  2022
34  06/21/2023  11/01/2022  11/30/2022              United States  2022
35  06/21/2023  12/01/2022  12/31/2022              United States  2022
36  06/21/2023  01/01/2023  01/31/2023              United States  2023
37  06/21/2023  02/01/2023  02/28/2023              United States  2023
38  06/21/2023  03/01/2023  03/31/2023              United States  2023
39  06/21/2023  04/01/2023  04/30/2023              United States  2023
40  06/21/2023  05/01/2023  05/31/2023              United States  2023
41  06/21/2023  06/01/2023  06/17/2023              United States  2023

    Month  All Cause  Natural Cause  Septicemia  Malignant Neoplasms  ...  \
0       1     264681         242914        3687                52635  ...
1       2     244966         224343        3324                48764  ...
2       3     269806         247634        3669                51640  ...
3       4     322424         300780        3366                48773  ...
4       5     280564         255489        3085                49012  ...
5       6     250456         225455        3036                47962  ...
6       7     279012         252481        3127                50626  ...
7       8     277282         251071        3268                51209  ...
8       9     257190         232827        3136                49671  ...
9      10     273906         249366        3250                51255  ...
10     11     302583         278769        3388                49658  ...
11     12     367209         342681        3778                51907  ...
12      1     373642         348017        3688                51602  ...
13      2     282548         259720        3366                46084  ...
```

```
14     3    271042        245457      3392              50318  ...
15     4    257047        231263      3046              49280  ...
16     5    258360        231635      3111              50587  ...
17     6    245269        218347      3158              49258  ...
18     7    257929        230220      3419              51281  ...
19     8    304052        276341      3405              52355  ...
20     9    312514        285695      3544              50314  ...
21    10    300219        273380      3763              52119  ...
22    11    289077        263564      3615              50326  ...
23    12    320033        293773      3834              52506  ...
24     1    370087        343560      4026              53166  ...
25     2    290122        265820      3328              46279  ...
26     3    268525        242498      3560              50963  ...
27     4    246402        221751      3290              48912  ...
28     5    255097        228830      3352              50658  ...
29     6    248969        222510      3288              49217  ...
30     7    260946        233086      3367              51199  ...
31     8    259736        232811      3319              51588  ...
32     9    251591        225540      3356              50321  ...
33    10    265623        239226      3513              52098  ...
34    11    268689        243608      3696              51123  ...
35    12    301072        274898      4169              53385  ...
36     1    288918        264949      4058              52379  ...
37     2    248422        227826      3290              47614  ...
38     3    267404        246550      3745              51655  ...
39     4    248416        230905      3404              49224  ...
40     5    232815        222411      3075              48925  ...
41     6     71615         71585       981              16744  ...

    Intentional Self-Harm (Suicide)  Assault (Homicide)  Drug Overdose  \
0                            4040.0              1708.0         6547.0
1                            3672.0              1471.0         6435.0
2                            3952.0              1693.0         7268.0
3                            3480.0              1756.0         7938.0
4                            3769.0              2067.0         9466.0
5                            3985.0              2261.0         8212.0
6                            4184.0              2426.0         8583.0
7                            4055.0              2348.0         8351.0
8                            3925.0              2191.0         7589.0
9                            3804.0              2368.0         7486.0
10                           3716.0              2242.0         7417.0
11                           3581.0              2230.0         7760.0
12                           3893.0              2154.0         8858.0
13                           3568.0              1826.0         7761.0
14                           3992.0              1935.0         9339.0
15                           3720.0              2068.0         9490.0
16                           4093.0              2361.0         9334.0
17                           4077.0              2296.0         9073.0
18                           4090.0              2407.0         9441.0
19                           4366.0              2304.0         9415.0
20                           4232.0              2272.0         8994.0
21                           4267.0              2332.0         8809.0
22                           3952.0              2022.0         8325.0
23                           3898.0              2136.0         8626.0
24                           4153.0              2030.0         9050.0
25                           3815.0              1844.0         8589.0
26                           4108.0              1959.0         9294.0
27                           4032.0              2073.0         8656.0
28                           4404.0              2253.0         8958.0
29                           4287.0              2227.0         8983.0
```

```
30                      4366.0              2365.0          9598.0
31                      4354.0              2236.0          9249.0
32                      4111.0              2061.0          8880.0
33                      4075.0              2085.0          9055.0
34                      3882.0              1883.0          8695.0
35                         NaN                 NaN             NaN
36                         NaN                 NaN             NaN
37                         NaN                 NaN             NaN
38                         NaN                 NaN             NaN
39                         NaN                 NaN             NaN
40                         NaN                 NaN             NaN
41                         NaN                 NaN             NaN

    COVID-19 (Multiple Cause of Death)  COVID-19 (Underlying Cause of Death)
\
0                                    6                                      4
1                                   25                                     20
2                                 7175                                   6785
3                                65553                                  62014
4                                38330                                  35279
5                                18026                                  15827
6                                31135                                  28279
7                                29913                                  27031
8                                19158                                  16858
9                                24930                                  22083
10                               53250                                  48037
11                               98175                                  89612
12                              105566                                  96438
13                               48570                                  42857
14                               23268                                  19633
15                               18805                                  16020
16                               14989                                  12744
17                                8024                                   6561
18                               11222                                   9741
19                               48822                                  45466
20                               63443                                  59212
21                               42606                                  38771
22                               32328                                  29008
23                               45623                                  41417
24                               83989                                  72635
25                               50235                                  40989
26                               15620                                  10986
27                                6265                                   3672
28                                7634                                   4823
29                                9533                                   6236
30                               13389                                   8944
31                               14128                                   9349
32                               11119                                   7049
33                                9709                                   6082
34                               10021                                   6378
35                               14349                                   9735
36                               14801                                   9980
37                                8933                                   5781
38                                7491                                   4746
39                                5053                                   3121
40                                3271                                   1978
41                                 845                                    505

                  flag_accid                               flag_mva  \
0                        NaN                                    NaN
```

```
1                         NaN                         NaN
2                         NaN                         NaN
3                         NaN                         NaN
4                         NaN                         NaN
5                         NaN                         NaN
6                         NaN                         NaN
7                         NaN                         NaN
8                         NaN                         NaN
9                         NaN                         NaN
10                        NaN                         NaN
11                        NaN                         NaN
12                        NaN                         NaN
13                        NaN                         NaN
14                        NaN                         NaN
15                        NaN                         NaN
16                        NaN                         NaN
17                        NaN                         NaN
18                        NaN                         NaN
19                        NaN                         NaN
20                        NaN                         NaN
21                        NaN                         NaN
22                        NaN                         NaN
23                        NaN                         NaN
24                        NaN                         NaN
25                        NaN                         NaN
26                        NaN                         NaN
27                        NaN                         NaN
28                        NaN                         NaN
29                        NaN                         NaN
30                        NaN                         NaN
31                        NaN                         NaN
32                        NaN                         NaN
33                        NaN                         NaN
34                        NaN                         NaN
35  Data not shown (6 month lag)  Data not shown (6 month lag)
36  Data not shown (6 month lag)  Data not shown (6 month lag)
37  Data not shown (6 month lag)  Data not shown (6 month lag)
38  Data not shown (6 month lag)  Data not shown (6 month lag)
39  Data not shown (6 month lag)  Data not shown (6 month lag)
40  Data not shown (6 month lag)  Data not shown (6 month lag)
41  Data not shown (6 month lag)  Data not shown (6 month lag)

                    flag_suic                  flag_homic  \
0                         NaN                         NaN
1                         NaN                         NaN
2                         NaN                         NaN
3                         NaN                         NaN
4                         NaN                         NaN
5                         NaN                         NaN
6                         NaN                         NaN
7                         NaN                         NaN
8                         NaN                         NaN
9                         NaN                         NaN
10                        NaN                         NaN
11                        NaN                         NaN
12                        NaN                         NaN
13                        NaN                         NaN
14                        NaN                         NaN
15                        NaN                         NaN
16                        NaN                         NaN
```

```
17                         NaN                         NaN
18                         NaN                         NaN
19                         NaN                         NaN
20                         NaN                         NaN
21                         NaN                         NaN
22                         NaN                         NaN
23                         NaN                         NaN
24                         NaN                         NaN
25                         NaN                         NaN
26                         NaN                         NaN
27                         NaN                         NaN
28                         NaN                         NaN
29                         NaN                         NaN
30                         NaN                         NaN
31                         NaN                         NaN
32                         NaN                         NaN
33                         NaN                         NaN
34                         NaN                         NaN
35  Data not shown (6 month lag)  Data not shown (6 month lag)
36  Data not shown (6 month lag)  Data not shown (6 month lag)
37  Data not shown (6 month lag)  Data not shown (6 month lag)
38  Data not shown (6 month lag)  Data not shown (6 month lag)
39  Data not shown (6 month lag)  Data not shown (6 month lag)
40  Data not shown (6 month lag)  Data not shown (6 month lag)
41  Data not shown (6 month lag)  Data not shown (6 month lag)

                   flag_drugod
0                          NaN
1                          NaN
2                          NaN
3                          NaN
4                          NaN
5                          NaN
6                          NaN
7                          NaN
8                          NaN
9                          NaN
10                         NaN
11                         NaN
12                         NaN
13                         NaN
14                         NaN
15                         NaN
16                         NaN
17                         NaN
18                         NaN
19                         NaN
20                         NaN
21                         NaN
22                         NaN
23                         NaN
24                         NaN
25                         NaN
26                         NaN
27                         NaN
28                         NaN
29                         NaN
30                         NaN
31                         NaN
32                         NaN
```

```
33                            NaN
34                            NaN
35  Data not shown (6 month lag)
36  Data not shown (6 month lag)
37  Data not shown (6 month lag)
38  Data not shown (6 month lag)
39  Data not shown (6 month lag)
40  Data not shown (6 month lag)
41  Data not shown (6 month lag)

[42 rows x 31 columns]
```

In [62]:
```
# A printout of the datasets for the second scenario, to show columns that cont
print('contributing_conditions_data')
print(contributing_conditions_data)
print('oasis_demographics_data')
print(oasis_demographics_data)
print('oasis_cross_sectional_data')
print(oasis_cross_sectional_data)
print('alzheimer_data')
print(alzheimer_data)
```

```
contributing_conditions_data
       Data As Of  Start Date    End Date     Group    Year  Month  \
0       06/11/2023  01/01/2020  06/10/2023  By Total     NaN    NaN
1       06/11/2023  01/01/2020  06/10/2023  By Total     NaN    NaN
2       06/11/2023  01/01/2020  06/10/2023  By Total     NaN    NaN
3       06/11/2023  01/01/2020  06/10/2023  By Total     NaN    NaN
4       06/11/2023  01/01/2020  06/10/2023  By Total     NaN    NaN
...            ...         ...         ...       ...     ...    ...
583735  06/11/2023  02/01/2023  02/28/2023  By Month  2023.0    2.0
583736  06/11/2023  03/01/2023  03/31/2023  By Month  2023.0    3.0
583737  06/11/2023  04/01/2023  04/30/2023  By Month  2023.0    4.0
583738  06/11/2023  05/01/2023  05/31/2023  By Month  2023.0    5.0
583739  06/11/2023  06/01/2023  06/10/2023  By Month  2023.0    6.0

                State      Condition Group               Condition  \
0       United States  Respiratory diseases  Influenza and pneumonia
1       United States  Respiratory diseases  Influenza and pneumonia
2       United States  Respiratory diseases  Influenza and pneumonia
3       United States  Respiratory diseases  Influenza and pneumonia
4       United States  Respiratory diseases  Influenza and pneumonia
...               ...                   ...                      ...
583735    Puerto Rico              COVID-19                 COVID-19
583736    Puerto Rico              COVID-19                 COVID-19
583737    Puerto Rico              COVID-19                 COVID-19
583738    Puerto Rico              COVID-19                 COVID-19
583739    Puerto Rico              COVID-19                 COVID-19

       ICD10_codes Age Group  COVID-19 Deaths  Number of Mentions Flag
0          J09-J18      0-24           1553.0              1629.0  NaN
1          J09-J18     25-34           5773.0              5995.0  NaN
2          J09-J18     35-44          15019.0             15636.0  NaN
3          J09-J18     45-54          37323.0             38782.0  NaN
4          J09-J18     55-64          82334.0             85353.0  NaN
...            ...       ...              ...                 ...  ...
583735        U071  All Ages             99.0                99.0  NaN
583736        U071  All Ages             50.0                50.0  NaN
583737        U071  All Ages             43.0                43.0  NaN
583738        U071  All Ages             56.0                56.0  NaN
583739        U071  All Ages             18.0                18.0  NaN

[583740 rows x 14 columns]
oasis_demographics_data
    Subject ID         MRI ID        Group  Visit  MR Delay M/F Hand  Age  \
0    OAS2_0001  OAS2_0001_MR1  Nondemented      1         0   M    R   87
1    OAS2_0001  OAS2_0001_MR2  Nondemented      2       457   M    R   88
2    OAS2_0002  OAS2_0002_MR1     Demented      1         0   M    R   75
3    OAS2_0002  OAS2_0002_MR2     Demented      2       560   M    R   76
4    OAS2_0002  OAS2_0002_MR3     Demented      3      1895   M    R   80
..         ...            ...          ...    ...       ...  ..  ...  ...
368  OAS2_0185  OAS2_0185_MR2     Demented      2       842   M    R   82
369  OAS2_0185  OAS2_0185_MR3     Demented      3      2297   M    R   86
370  OAS2_0186  OAS2_0186_MR1  Nondemented      1         0   F    R   61
371  OAS2_0186  OAS2_0186_MR2  Nondemented      2       763   F    R   63
372  OAS2_0186  OAS2_0186_MR3  Nondemented      3      1608   F    R   65

     EDUC  SES  MMSE  CDR         eTIV      nWBV       ASF
0      14  2.0  27.0  0.0  1986.550000  0.696106  0.883440
1      14  2.0  30.0  0.0  2004.479526  0.681062  0.875539
2      12  NaN  23.0  0.5  1678.290000  0.736336  1.045710
3      12  NaN  28.0  0.5  1737.620000  0.713402  1.010000
```

```
4       12  NaN  22.0  0.5  1697.911134  0.701236  1.033623
..     ...  ...   ...  ...          ...       ...       ...
368     16  1.0  28.0  0.5  1692.880000  0.693926  1.036690
369     16  1.0  26.0  0.5  1688.009649  0.675457  1.039686
370     13  2.0  30.0  0.0  1319.020000  0.801006  1.330540
371     13  2.0  30.0  0.0  1326.650000  0.795981  1.322890
372     13  2.0  30.0  0.0  1332.944463  0.801248  1.316634

[373 rows x 15 columns]
oasis_cross_sectional_data
                ID M/F Hand  Age  Educ  SES  MMSE  CDR  eTIV   nWBV    ASF  \
0    OAS1_0001_MR1   F    R   74   2.0  3.0  29.0  0.0  1344  0.743  1.306
1    OAS1_0002_MR1   F    R   55   4.0  1.0  29.0  0.0  1147  0.810  1.531
2    OAS1_0003_MR1   F    R   73   4.0  3.0  27.0  0.5  1454  0.708  1.207
3    OAS1_0004_MR1   M    R   28   NaN  NaN   NaN  NaN  1588  0.803  1.105
4    OAS1_0005_MR1   M    R   18   NaN  NaN   NaN  NaN  1737  0.848  1.010
..             ...  ..  ...  ...   ...  ...   ...  ...   ...    ...    ...
431  OAS1_0285_MR2   M    R   20   NaN  NaN   NaN  NaN  1469  0.847  1.195
432  OAS1_0353_MR2   M    R   22   NaN  NaN   NaN  NaN  1684  0.790  1.042
433  OAS1_0368_MR2   M    R   22   NaN  NaN   NaN  NaN  1580  0.856  1.111
434  OAS1_0379_MR2   F    R   20   NaN  NaN   NaN  NaN  1262  0.861  1.390
435  OAS1_0395_MR2   F    R   26   NaN  NaN   NaN  NaN  1283  0.834  1.368

     Delay
0      NaN
1      NaN
2      NaN
3      NaN
4      NaN
..     ...
431    2.0
432   40.0
433   89.0
434    2.0
435   39.0

[436 rows x 12 columns]
alzheimer_data
           Group M/F  Age  EDUC  SES  MMSE  CDR  eTIV   nWBV    ASF
0    Nondemented   M   87    14  2.0  27.0  0.0  1987  0.696  0.883
1    Nondemented   M   88    14  2.0  30.0  0.0  2004  0.681  0.876
2       Demented   M   75    12  NaN  23.0  0.5  1678  0.736  1.046
3       Demented   M   76    12  NaN  28.0  0.5  1738  0.713  1.010
4       Demented   M   80    12  NaN  22.0  0.5  1698  0.701  1.034
..           ...  ..  ...   ...  ...   ...  ...   ...    ...    ...
368     Demented   M   82    16  1.0  28.0  0.5  1693  0.694  1.037
369     Demented   M   86    16  1.0  26.0  0.5  1688  0.675  1.040
370  Nondemented   F   61    13  2.0  30.0  0.0  1319  0.801  1.331
371  Nondemented   F   63    13  2.0  30.0  0.0  1327  0.796  1.323
372  Nondemented   F   65    13  2.0  30.0  0.0  1333  0.801  1.317

[373 rows x 10 columns]
```

In [63]:

```
# Establishing a four percent null value threshold to apply to the data cleanin
threshold = int(0.04 * len(monthly_provisional_data))

throw = []

for col in monthly_provisional_data.columns:
    if monthly_provisional_data[col].isnull().sum()!=0:
```

```
        if monthly_provisional_data[col].isnull().sum()>threshold:
            throw.append(col)


old_monthly_provisional_data = monthly_provisional_data

monthly_provisional_data = monthly_provisional_data.drop(columns=throw, axis=1)
```

In [64]:
```
# 'Helper' functions that perform data cleaning for the datasets in scenarios c

def drop_columns(columns_to_drop, dataset):
    for col in columns_to_drop:
        if col in dataset.columns:
            dataset = dataset.drop(columns=col, axis=1)
    return dataset

def clean_data(dataset):
    dataset = dataset.dropna()
    if dataset.isnull().sum().sum() > 0:
        clean_data(dataset)
```

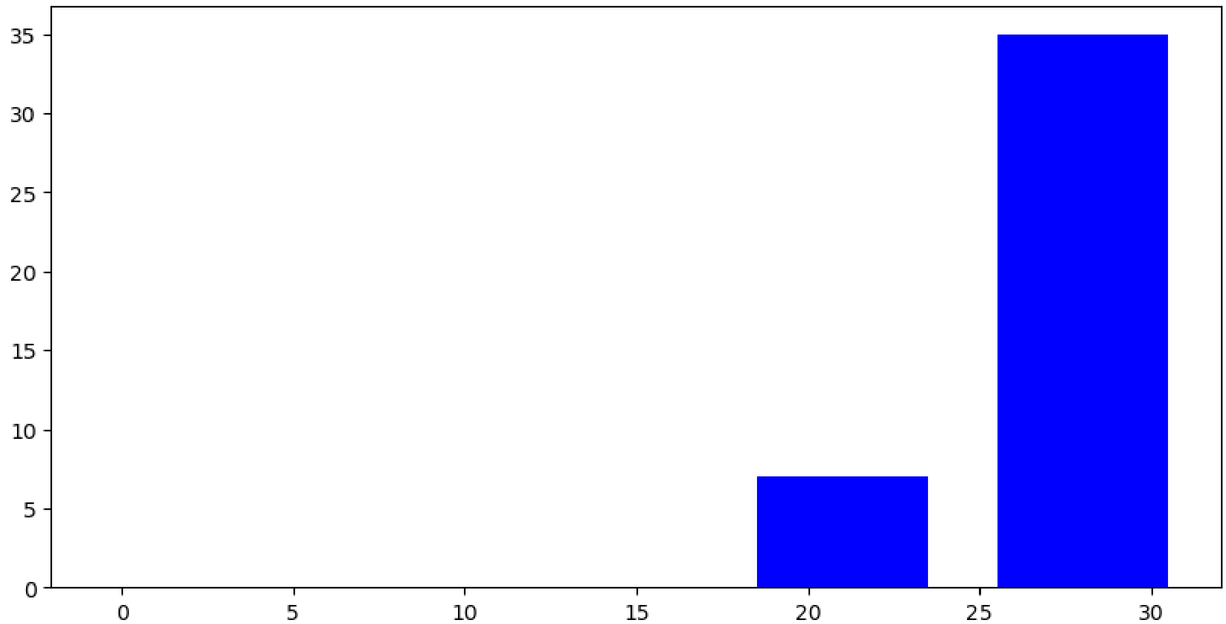
DATA CLEANING VISUALIZATIONS (SCENARIO ONE)

The following visualization and output information will show a comparison of the scenario one dataset, counting the number of null values before and after the cleaning process.

In [65]:
```
# Comparing the null count between two versions of the scenario one
# dataset (before and after data cleaning).

old_count = old_monthly_provisional_data.isna().sum()
new_count = monthly_provisional_data.isna().sum().sum()
print("Number of null values per column in raw data: \n%s\n\n" %old_count)


# Plotting the null value count of the raw dataset, prior to
# data cleaning.

x = [old_count.tolist(), new_count.tolist()]
fig = plt.figure(figsize = (10, 5))
# creating the bar plot
plt.bar(range(len(old_count)), x[0], color='blue',width = 1)
plt.bar(range(len(old_count)), x[1], color='orange')

plt.show()

print("Number of null values per column in cleaned data: %s" %new_count)
```

```
Number of null values per column in raw data:
Data As Of
0
Start Date
0
End Date
0
Jurisdiction of Occurrence
0
Year
0
Month
0
All Cause
0
Natural Cause
0
Septicemia
0
Malignant Neoplasms
0
Diabetes Mellitus
0
Alzheimer Disease
0
Influenza and Pneumonia
0
Chronic Lower Respiratory Diseases
0
Other Diseases of Respiratory System
0
Nephritis, Nephrotic Syndrome and Nephrosis
0
Symptoms, Signs and Abnormal Clinical and Laboratory Findings, Not Elsewhere C
lassified     0
Diseases of Heart
0
Cerebrovascular Diseases
0
Accidents (Unintentional Injuries)
7
Motor Vehicle Accidents
7
Intentional Self-Harm (Suicide)
7
Assault (Homicide)
7
Drug Overdose
7
COVID-19 (Multiple Cause of Death)
0
COVID-19 (Underlying Cause of Death)
0
flag_accid
35
flag_mva
35
flag_suic
35
flag_homic
```

```
35
flag_drugod
35
dtype: int64
```




```
Number of null values per column in cleaned data: 0
```

In [66]:
```
# Initial data cleaning and data transformations to prepare the datasets for sc
# Dropping unnecessary columns from relevant datasets.

col3_to_remove = ['ICD10_codes', 'Start Date', 'End Date', 'Flag', 'Number of N
contributing_conditions_data = drop_columns(col3_to_remove, contributing_condit

oasis_demographics_data = oasis_demographics_data[oasis_demographics_data['Grou

col5_to_remove = ['Educ','Delay']
oasis_cross_sectional_data = drop_columns(col5_to_remove, oasis_cross_sectional

alzheimer_data = alzheimer_data[alzheimer_data['Group']=='Demented']


datasets2 = {'Contributing Contributions Dataset': contributing_conditions_data
             'Oasis Longitudinal Demographics Dataset': oasis_demographics_data
             'Oasis Cross-sectional Demographics Dataset': oasis_cross_sectiona
             'Alzheimver Features Dataset': alzheimer_data}
```

In [67]:
```
# Removing special characters from monthly provisional dataset feature names. T
# processing and model training.

monthly_provisional_data.columns = monthly_provisional_data.columns.str.replace
monthly_provisional_data.columns = monthly_provisional_data.columns.str.replace
monthly_provisional_data.columns = monthly_provisional_data.columns.str.replace
monthly_provisional_data.columns = monthly_provisional_data.columns.str.replace
monthly_provisional_data.columns = monthly_provisional_data.columns.str.replace
monthly_provisional_data.columns = monthly_provisional_data.columns.str.replace
```

```
/var/folders/k1/f3l1yym17tl9cvqg5972snr00000gn/T/ipykernel_13055/2519984467.p
y:6: FutureWarning: The default value of regex will change from True to False
in a future version. In addition, single character regular expressions will *n
ot* be treated as literal strings when regex=True.
  monthly_provisional_data.columns = monthly_provisional_data.columns.str.repl
ace('(', '')
/var/folders/k1/f3l1yym17tl9cvqg5972snr00000gn/T/ipykernel_13055/2519984467.p
y:7: FutureWarning: The default value of regex will change from True to False
in a future version. In addition, single character regular expressions will *n
ot* be treated as literal strings when regex=True.
  monthly_provisional_data.columns = monthly_provisional_data.columns.str.repl
ace(')', '')
```

CONCLUSION OF THE DATA CLEANING PROCESS

Previous efforts of data cleaning led to failure when it came to applying the model approach for both scenarios. The reason for this was because missing entries were still present in the modified datasets for both scenarios, even after data cleaning. To conduct more thorough data cleaning, I established the previously mentioned null threshold value and 'helper' functions to improve the cleaning process for all involved datasets.

EXPLORATORY DATA ANALYSIS (EDA) FOR SCENARIO ONE

Regarding the EDA process for scenario one, the first step is to generate a correlation matrix and a series of pair plots for the updated montly provisional dataset. The goal of these visualizations is to see the correlations between features and determine where strong correlations and collinearity resides. As a result, the analysis concludes that there is strong correlation between many of the features in the dataset, as indicated in the following correlation matrix and pair plots. Also, according to the following correlation matrix, there is strong multicollinearity as well between features.

Further evaluation and metrics is needed to determine the effectiveness of different disease types as column features when tested with the types of causes of deaths. These features (including "AlzheimerDisease") are used in the formation of the associated multilinear regression models. The various disease features are tested with multiple predicting factors, including 'AllCause', 'COVID19MultipleCauseofDeath', and 'COVID19UnderlyingCauseofDeath'.

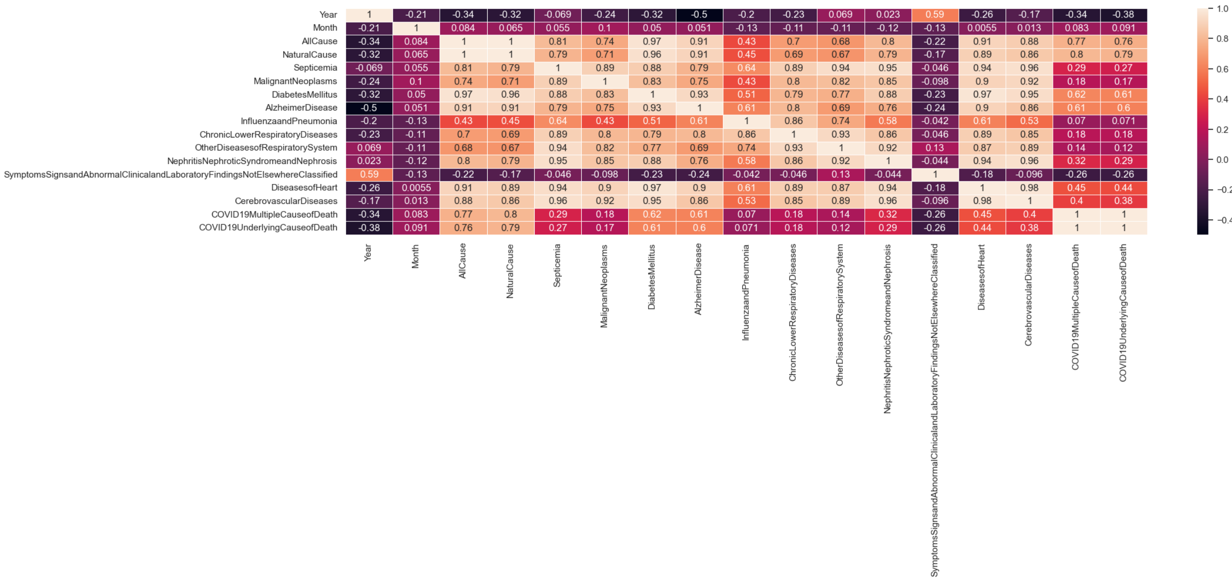
The reason there are multiple predicting factors (and in turn, multiple iterations of the multilinear regression model) is because I specifically want to see how effective the column feature "AlzheimerDisease" is in the case of different types of causes of death (i.e. natural causes, as an underlying cause related to COVID19, etc.).

In [68]:
```
# Importing addition Python libraries to create visualizations as well as creat
# for both scenarios

from sklearn.linear_model import LogisticRegression
from sklearn import model_selection
from sklearn import linear_model
```

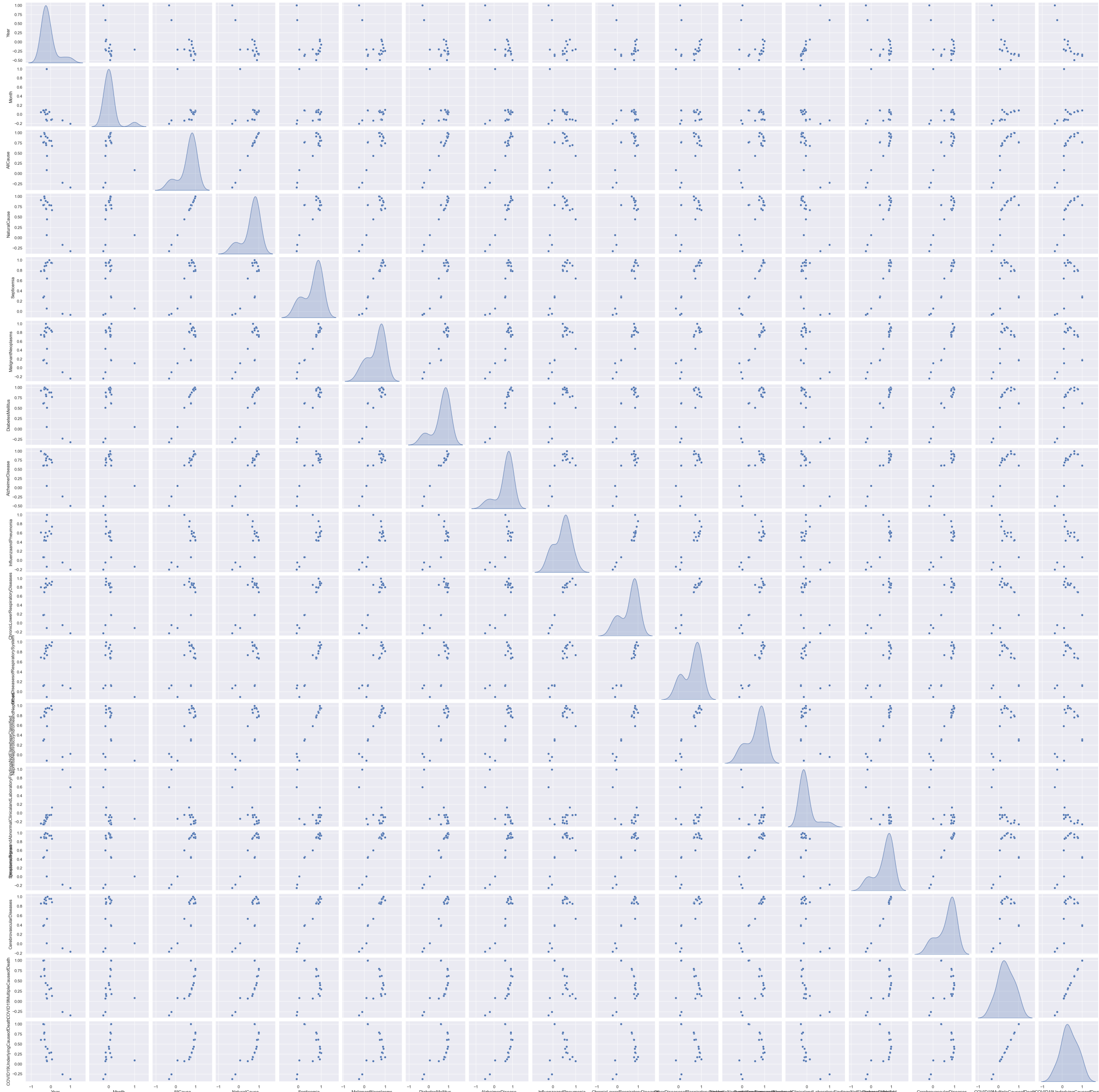
```
from sklearn.model_selection import train_test_split
import seaborn as sns
sns.set()
import statsmodels.formula.api as smf
import statsmodels.api as sm
import matplotlib.pyplot as plt
```

In [69]:
```
# Plotting the correlation matrix for the scenario one dataset.
corr_matrix = monthly_provisional_data.corr()
fig, ax = plt.subplots(figsize=(22, 5))
sns.heatmap(corr_matrix, annot=True, linewidth=0.5)
plt.savefig('correlation_matrix.png', dpi = 300, bbox_inches = 'tight')
```

```
/var/folders/k1/f3l1yym17tl9cvqg5972snr00000gn/T/ipykernel_13055/1529644853.p
y:2: FutureWarning: The default value of numeric_only in DataFrame.corr is dep
recated. In a future version, it will default to False. Select only valid colu
mns or specify the value of numeric_only to silence this warning.
  corr_matrix = monthly_provisional_data.corr()
```



In [70]:
```
# Plotting a series of pair plots for the scenario one dataset
sns.pairplot(corr_matrix, diag_kind='kde')
plt.savefig('pair_plot.png', dpi = 300, bbox_inches = 'tight')
```



MULTILINEAR REGRESSION MODEL ANALYSIS FOR SCENARIO ONE

For the next part, I use a series of multilinear regression models to test different types of death causes with 'AlzheimerDisease' and other disease types found as dataset features. The main focus here is on the model summary results for 'p-values' for the different features and discover any patterns that may envelop. Any features that are found with a 'p-value' less than 0.05 are considered ineffective and not considered good factors when tested with the corresponding predicting features.

In [71]:
```
# Creating the first iteration of the Multilinear Regression model with 'AllCau
x = monthly_provisional_data[['MalignantNeoplasms', 'DiabetesMellitus','Disease
                              'InfluenzaandPneumonia', 'ChronicLowerRespiratoryI
                              'DiseasesofHeart']]
y = monthly_provisional_data[['AllCause']]
regr = linear_model.LinearRegression()
regr.fit(x, y)
```

```
# with statsmodels
x = sm.add_constant(x) # adding a constant

model = sm.OLS(y, x).fit()
predictions = model.predict(x)
```

In [72]:
```
# Generating the model summary and evaluation metrics for scenario one
model.summary()
```

Out[72]:

OLS Regression Results

Dep. Variable:	AllCause	**R-squared:**	0.966
Model:	OLS	**Adj. R-squared:**	0.958
Method:	Least Squares	**F-statistic:**	136.3
Date:	Mon, 26 Jun 2023	**Prob (F-statistic):**	5.63e-23
Time:	20:59:17	**Log-Likelihood:**	-439.56
No. Observations:	42	**AIC:**	895.1
Df Residuals:	34	**BIC:**	909.0
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	7829.8612	1.41e+04	0.554	0.583	-2.09e+04	3.65e+04
MalignantNeoplasms	-0.7712	0.969	-0.796	0.432	-2.740	1.197
DiabetesMellitus	49.9709	9.938	5.028	0.000	29.775	70.167
DiseasesofHeart	-2.0285	1.151	-1.762	0.087	-4.368	0.311
AlzheimerDisease	7.0316	3.323	2.116	0.042	0.279	13.785
InfluenzaandPneumonia	-4.7917	3.937	-1.217	0.232	-12.792	3.208
ChronicLowerRespiratoryDiseases	0.3633	5.382	0.068	0.947	-10.574	11.301
OtherDiseasesofRespiratorySystem	16.7025	9.921	1.683	0.101	-3.460	36.865
DiseasesofHeart	-2.0285	1.151	-1.762	0.087	-4.368	0.311

Omnibus:	1.254	**Durbin-Watson:**	1.389
Prob(Omnibus):	0.534	**Jarque-Bera (JB):**	1.117
Skew:	0.382	**Prob(JB):**	0.572
Kurtosis:	2.764	**Cond. No.**	1.43e+18

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.94e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

In [73]:
```
# Creating the second iteration of the Linear Regression model with 'COVID19Mul
x = monthly_provisional_data[['MalignantNeoplasms', 'DiabetesMellitus','Disease
                              'InfluenzaandPneumonia', 'ChronicLowerRespiratoryD
                              'DiseasesofHeart']]
y = monthly_provisional_data[['COVID19MultipleCauseofDeath']]
regr = linear_model.LinearRegression()
regr.fit(x, y)

x = sm.add_constant(x) # adding a constant

model = sm.OLS(y, x).fit()
```

In [74]:
```
# Generating the model summary and evaluation metrics for scenario one (second
model.summary()
```

Out[74]:

OLS Regression Results

Dep. Variable:	COVID19MultipleCauseofDeath	**R-squared:**	0.883
Model:	OLS	**Adj. R-squared:**	0.859
Method:	Least Squares	**F-statistic:**	36.65
Date:	Mon, 26 Jun 2023	**Prob (F-statistic):**	4.93e-14
Time:	20:59:18	**Log-Likelihood:**	-441.40
No. Observations:	42	**AIC:**	898.8
Df Residuals:	34	**BIC:**	912.7
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.583e+04	1.48e+04	1.073	0.291	-1.42e+04	4.58e+04
MalignantNeoplasms	-3.3477	1.012	-3.308	0.002	-5.405	-1.291
DiabetesMellitus	48.2608	10.383	4.648	0.000	27.160	69.362
DiseasesofHeart	-2.8382	1.203	-2.360	0.024	-5.282	-0.394
AlzheimerDisease	7.2680	3.472	2.093	0.044	0.212	14.324
InfluenzaandPneumonia	-6.3079	4.113	-1.534	0.134	-14.666	2.050
ChronicLowerRespiratoryDiseases	1.1252	5.623	0.200	0.843	-10.302	12.553
OtherDiseasesofRespiratorySystem	9.8939	10.366	0.954	0.347	-11.172	30.960
DiseasesofHeart	-2.8382	1.203	-2.360	0.024	-5.282	-0.394

Omnibus:	1.176	**Durbin-Watson:**	1.306
Prob(Omnibus):	0.555	**Jarque-Bera (JB):**	1.190
Skew:	0.347	**Prob(JB):**	0.552
Kurtosis:	2.556	**Cond. No.**	1.43e+18

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.94e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

In [75]:

```
# Creating the third iteration of the Linear Regression model with 'COVID19Unde
x = monthly_provisional_data[['MalignantNeoplasms', 'DiabetesMellitus','Disease
                              'InfluenzaandPneumonia', 'ChronicLowerRespiratoryI
                              'DiseasesofHeart']]
y = monthly_provisional_data[['COVID19UnderlyingCauseofDeath']]
regr = linear_model.LinearRegression()
regr.fit(x, y)

# with statsmodels
```

```
x = sm.add_constant(x) # adding a constant

model = sm.OLS(y, x).fit()

# Generating the model summary and evaluation metrics for scenario one (third i
model.summary()
```

Out[75]:

OLS Regression Results

Dep. Variable:	COVID19UnderlyingCauseofDeath	**R-squared:**	0.872
Model:	OLS	**Adj. R-squared:**	0.845
Method:	Least Squares	**F-statistic:**	33.04
Date:	Mon, 26 Jun 2023	**Prob (F-statistic):**	2.25e-13
Time:	20:59:20	**Log-Likelihood:**	-440.29
No. Observations:	42	**AIC:**	896.6
Df Residuals:	34	**BIC:**	910.5
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.522e+04	1.44e+04	1.059	0.297	-1.4e+04	4.44e+04
MalignantNeoplasms	-2.7476	0.986	-2.788	0.009	-4.751	-0.745
DiabetesMellitus	49.1374	10.111	4.860	0.000	28.588	69.686
DiseasesofHeart	-3.2329	1.171	-2.760	0.009	-5.613	-0.853
AlzheimerDisease	6.4606	3.381	1.911	0.064	-0.410	13.331
InfluenzaandPneumonia	-5.8816	4.005	-1.468	0.151	-14.021	2.258
ChronicLowerRespiratoryDiseases	3.2700	5.476	0.597	0.554	-7.858	14.398
OtherDiseasesofRespiratorySystem	6.1158	10.095	0.606	0.549	-14.399	26.631
DiseasesofHeart	-3.2329	1.171	-2.760	0.009	-5.613	-0.853

Omnibus:	0.911	**Durbin-Watson:**	1.318
Prob(Omnibus):	0.634	**Jarque-Bera (JB):**	0.974
Skew:	0.290	**Prob(JB):**	0.615
Kurtosis:	2.532	**Cond. No.**	1.43e+18

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.94e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

MULTILINEAR REGRESSION EVALUATION METRICS RESULTS FOR SCENARIO ONE

Evaluation metrics show that, from the three iterations of the multilinear regression model, the first and second iterations (with 'AllCause' and 'COVID19MultipleCauseofDeath' as the predicting features) showed 'AlzheimerDisease' as an significant feature. In addition to this result, the first and second iterations also produced high r-squared values of 0.966 and 0.883, respectively. Future tests can be done to explore other predicting factors, but as of now, I have been able to show effectiveness of the 'AlzheimerDisease' feature with this model approach.

DATA CLEANING READOUTS AND RESULTS (SCENARIO TWO)

The following output information will show a comparison of the scenario two datasets, counting the number of null values before and after the cleaning process.

In [76]:
```
# 'Helper' function that forms the final version of the dataset for scenario 2.
def transform_dataset2(datasets, limit):
    new_dataset = {}

    new_dataset['age'] = []
    new_dataset['sex'] = []

    for dataset in datasets.values():
        new_dataset['age'].extend(dataset[col].tolist()[:limit] for col in data
        new_dataset['sex'].extend(dataset[col].tolist()[:limit] for col in data

    all_ages = []
    genders = []
    group = []

    for age_,sex_ in zip(new_dataset['age'], new_dataset['sex']):
        all_ages += age_
        genders += sex_

    for ind in range(len(genders)):
        if genders[ind] == 'M':
            genders[ind] = 0
        else:
            genders[ind] = 1

    new_dataset['age'] = all_ages
    new_dataset['sex'] = genders

    return pd.DataFrame.from_dict(new_dataset)
```

In [77]:
```
scenario2_data = transform_dataset2(datasets2, 300)
```

In [78]:
```
# Prints out information on the null value counts for the scenario 2 dataset (b
temp = []

for key,dataset in datasets2.items():
    print("Total number of null count values for dataset: %s is the following:
```

```
    temp.append(dataset.isna().sum().tolist())

print("Total number of null count values for MODIFIED dataset is the following:
```

```
Total number of null count values for dataset: Contributing Contributions Data
set is the following:
Data As Of                0
Group                     0
Year                  12420
Month                 62100
State                     0
Condition Group           0
Condition                 0
Age Group                 0
COVID-19 Deaths      170909
dtype: int64

Total number of null count values for dataset: Oasis Longitudinal Demographics
Dataset is the following:
Subject ID     0
MRI ID         0
Group          0
Visit          0
MR Delay       0
M/F            0
Hand           0
Age            0
EDUC           0
SES           19
MMSE           2
CDR            0
eTIV           0
nWBV           0
ASF            0
dtype: int64

Total number of null count values for dataset: Oasis Cross-sectional Demograph
ics Dataset is the following:
ID        0
M/F       0
Hand      0
Age       0
SES     220
MMSE    201
CDR     201
eTIV      0
nWBV      0
ASF       0
dtype: int64

Total number of null count values for dataset: Alzheimver Features Dataset is
the following:
Group     0
M/F       0
Age       0
EDUC      0
SES      19
MMSE      2
CDR       0
eTIV      0
nWBV      0
ASF       0
dtype: int64
```

```
Total number of null count values for MODIFIED dataset is the following:

age    0
sex    0
dtype: int64
```

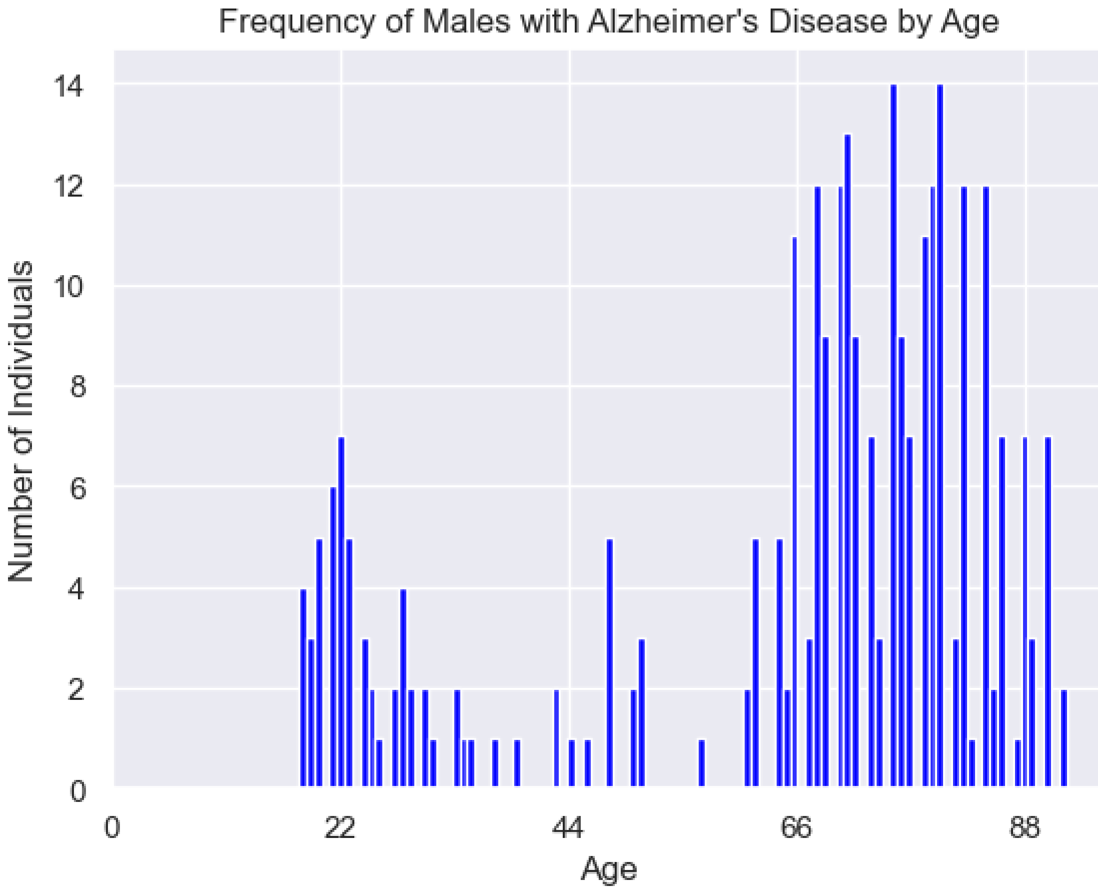
EXPLORATORY DATA ANALYSIS (EDA) FOR SCENARIO TWO

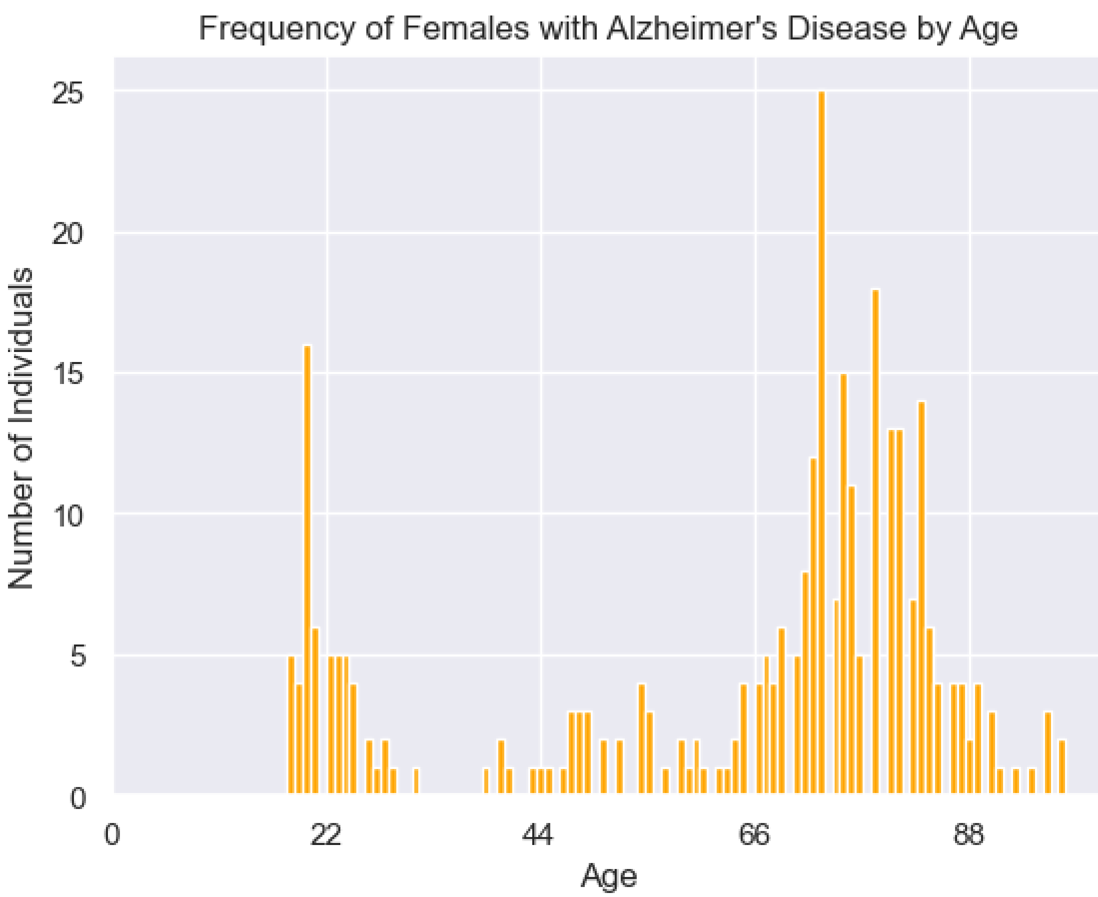
For scenario two, I utilize a 'helper' function (as shown above) to extract desired features from the three selected datasets (in this case, I am focusing on age and gender as our features). To further explore the modified data, I generated histograms to observe the frequency of male and female patients with Alzheimer's disease across different ages.

The histograms reveal that there are a higher number of male patients than female patients who have Alzheimer's disease, and that the highest number of patients (male and female) that have Alzheimer's disease are between the ages of 66 and 88. Unfortunately, the correlation was found to be not good between 'age' and 'gender' features. However, I decided to apply a logistic regression model since the 'age' feature results (in this case) to a binary outcome. For future datasets, a different model approach may be needed, since not all patients identify as 'male' nor 'female'. However, in this case, all three datasets that shared the 'gender' feature had binary results, so it seems appropriate to utilize a logistic regression model.

In [91]:
```
# Generates a set of histograms to display the frequency of patients with Alzhe
# varies by age.

scenario2_data[scenario2_data['sex']==0].hist(column='age', color='blue', bins=
plt.xlabel('Age')
plt.ylabel('Number of Individuals')
plt.xticks(np.arange(0, 100, 22))
plt.title("Frequency of Males with Alzheimer's Disease by Age")
scenario2_data[scenario2_data['sex']==1].hist(column='age', color='orange', bir
plt.xticks(np.arange(0, 100, 22))
plt.xlabel('Age')
plt.ylabel('Number of Individuals')
plt.title("Frequency of Females with Alzheimer's Disease by Age")
```

Out[91]:
```
Text(0.5, 1.0, "Frequency of Females with Alzheimer's Disease by Age")
```





LOGISTIC REGRESSION MODEL ANALYSIS FOR SCENARIO TWO

In [110…

```
# Forms the first iteration of the logistic regression for scenario 2.

X = scenario2_data.iloc[:, :-1].values
y = scenario2_data.iloc[:,-1].values
x_train, x_test, y_train, y_test = model_selection.train_test_split(X,y,test_si

LogReg = LogisticRegression(solver='liblinear')
LogReg.fit(x_train, y_train)
```

Out[110]:

```
▾ LogisticRegression
LogisticRegression(solver='liblinear')
```

LOGISTIC REGRESSION EVALUATION METRICS RESULTS FOR SCENARIO ONE

For evaluation metrics purposes, I am setting up not only one, but two iterations of logistic regression models with the dataset for scenario two. Multiple logistic regression models will help to determine which hyperparameters to tune or modify so that I can find a way to improve the model accuracy, if necessary.

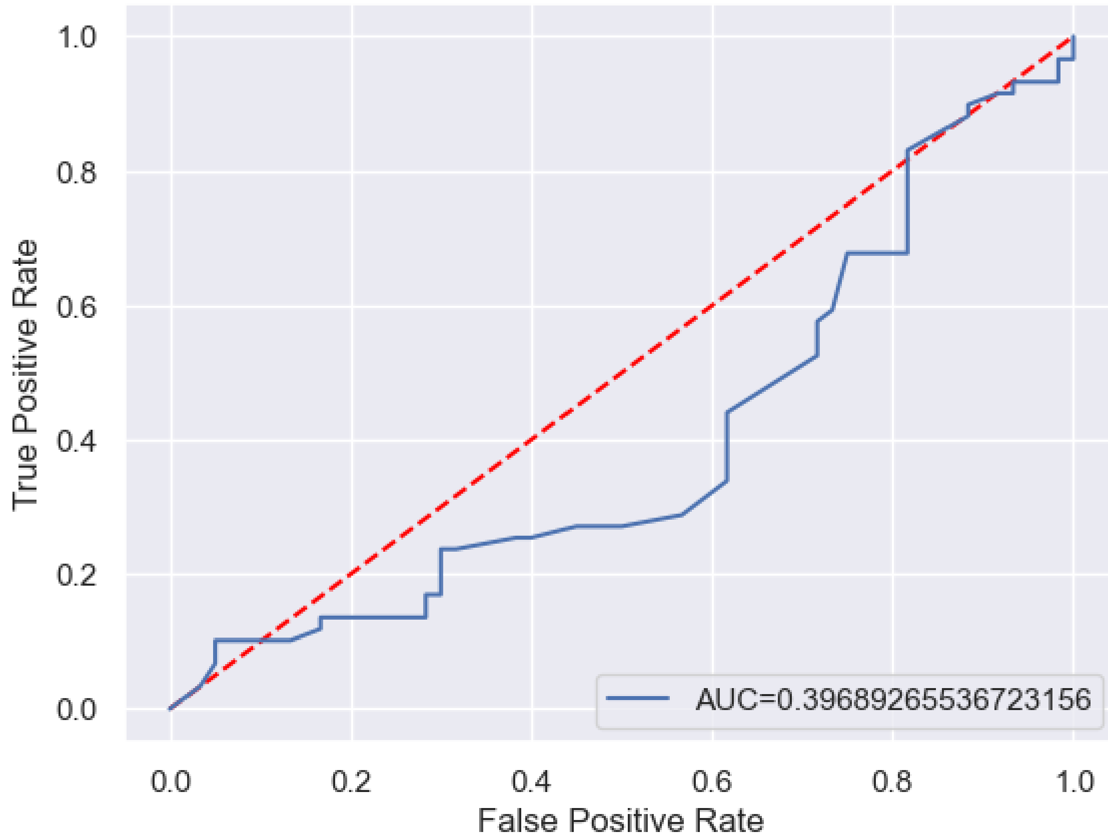
Initally, I use the entire modified scenario 2 dataset with a random state value of '5' when the model is created. However, when plotting the calculated true and false positive rates, the associated AUC value results to be 0.40 (lower than the standard of 0.50, which makes it an ineffective model). The model accuracy is also found to be 0.50.

To improve model performance and accuracy, I decided to change the size of the training and test data as well as the random state when recreating the logistic regression model. The following changes were made in an attempt to increase model performance: conducted dataset pruning to decrease the size of the training and test data, and increased the random state value in order to further shuffle the data before splitting it.

As a result of the changes, the logistic regression model improved in performance and the metrics revealed a higher AUC value of 0.64 (better than 0.50, which makes it a more improved model). Model accuracy increased as well to a value of 0.61.

In [111…

```
# Ploting the ROC curve and AUC metrics for the logistic regression model
from sklearn import metrics

# your code here
y_pred_score = LogReg.predict_proba(x_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_score)
auc = metrics.roc_auc_score(y_test, y_pred_score)

plt.figure()
plt.plot([0, 1], [0, 1], linestyle='--', color='red')
plt.plot(fpr,tpr,label="AUC="+str(auc))
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.legend(loc=4)
plt.show()
```



In [112…

```
# Computing the model accuracy score
y_pred = LogReg.predict(x_test)
print(metrics.accuracy_score(y_test,y_pred))
```

```
0.5042016806722689
```

In [105…

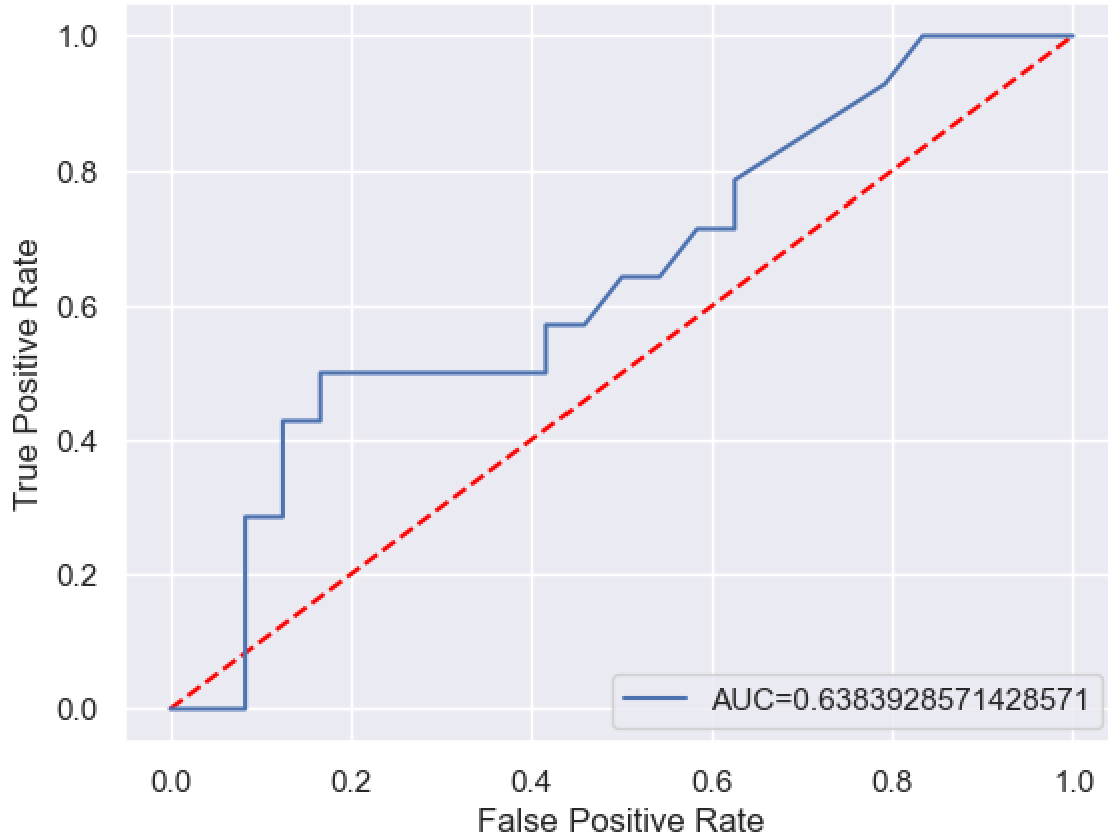
```
# Forms the second iteration of the logistic regression model, with a decreased
b = transform_dataset2(datasets2, 50)
X = b.iloc[:, :-1].values
y = b.iloc[:,-1].values
x_train, x_test, y_train, y_test = model_selection.train_test_split(X,y, test_s

LogReg = LogisticRegression(solver='liblinear')
LogReg.fit(x_train, y_train)
```

Out[105]:

```
▾ LogisticRegression
LogisticRegression(solver='liblinear')
```

In [106…

```
y_pred_score = LogReg.predict_proba(x_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_score)
auc = metrics.roc_auc_score(y_test, y_pred_score)

plt.figure()
plt.plot([0, 1], [0, 1], linestyle='--', color='red')
plt.plot(fpr,tpr,label="AUC="+str(auc))
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.legend(loc=4)
plt.show()
```



In [107…

```
y_pred = LogReg.predict(x_test)
print(metrics.accuracy_score(y_test,y_pred))
```

```
0.6052631578947368
```

SUMMARY OF RESULTS AND ANALYSIS

For both scenarios, the option was chosen to iterate and tune hyperparameters for both respective models until I noticed an improvement in model performance or achieved a desired outcome. In the case of scenario one, I changed the predicting features based on the given conditions (determine if Alzheimer's disease is impactful for any of the chosen predicting features. As a result, I found that the 'AlzheimerDisease' feature was significant in the case where 'COVID19MultipleCauseofDeath' and 'AllCause' were the predicting features.

In scenario two, I witnessed how the dataset size, random state, and how the input data is split could affect the logistic regression model accuracy and AUC metrics. Dataset pruning and increasing the random state led to an improvement in model performance in accuracy. However, further tests will need to be made in the future. Even though I witnessed an increase in model performance, I need to perform future tests with larger scales of data to make sure model performance is maintainable. In the likelihood that model performance does decrease with larger scales of data, I know that tuning the hyperparameters and data pruning are effective methods in case I need to improve model performance.

DISCUSSION AND CONCLUSION

In the end, I was able to achieve initial objectives: confirm if Alzheimer's Disease is an impactful feature in the multilinear regression model in scenario one, see if whether or not age and gender have a good correlation in scenario two, and verify if there are methods to improve the logistic regression model in scenario two.

The main takeaways and lessons learned from scenario one are the impact of data cleaning and utilizing multiple predicting features for the associated regression model. Before achieving success with the multilinear regression model in its current state, I encountered errors with certain features that remained in the input dataset. Occurring errors were a result of missing values still present in the dataset, and my previous efforts of data cleaning were not working properly. This led me to establish a null threshold value in order to limit the number of features that remained in the main dataset. As a result, I have learned how important the data cleaning process can be and how critical of a role it plays in data preparation and in data transformations.

For scenario two, I encountered a challenge with the lack of correlation between my selected features for patients with Alzheimer's disease: age and gender. Unfortunately, the correlation between both features was not good, even though these were shared features that were extracted from three datasets with the same associated time period. For future tests and improvement, I would test a similar scenario but with different data to verify if I get different correlation results with similar features. I have learned that even though a scenario did not achieve a desired outcome with the given data, it always helps to verify scenario results with different datasets. Even though I could not find good correlation here with the given features, I did find two methods to improve the model accuracy and performance: dataset pruning and increased shuffling of data. These are methods that I plan to keep in mind for future scenarios in case I encounter similar challenges with model accuracy and performance.

In [ ]:

```

```