# My Latest Project (Not mentioned in resume):

K-Means Clustering and Analysis for Leading Causes of Death in United States of America - Case Study  
  
1. Importing necessary libraries and dataset  
  
2. Data Preprocessing  
  
3. Analysis of United States  
  
4. Analysis of New York City  
  
5. Time Series Analysis  
  
6. K-Means Clustering  
  
7. Results Analysis

In [1]:

**import** pandas **as** pd

In [2]:

df **=** pd.read\_csv('causes-of-death.csv')

In [3]:

df.head()

Out[3]:

**Year 113 Cause Name Cause Name State Deaths Age-adjusted**

**Death Rate**

1. 2017 Accidents (unintentional injuries)

(V01-X59,Y8...

1. 2017 Accidents (unintentional injuries)

Unintentional

injuries

Unintentional

United States

169936 49.4

(V01-X59,Y8...

1. 2017 Accidents (unintentional injuries)

(V01-X59,Y8...

1. 2017 Accidents (unintentional injuries)

(V01-X59,Y8...

1. 2017 Accidents (unintentional injuries)

(V01-X59,Y8...

injuries Alabama 2703 53.8

Unintentional Alaska 436 63.7

injuries

Unintentional Arizona 4184 56.2

injuries

Unintentional Arkansas 1625 51.8

injuries

In [4]:

df.tail()

Out[4]:

**113 Cause Name Cause Name**

|  |  |
| --- | --- |
|  | **Year** |
| **10863** | 1999 |
| **10864** | 1999 |
| **10865** | 1999 |
| **10866** | 1999 |
| **10867** | 1999 |

**State Deaths Age-adjusted Death Rate**

Nephritis, nephrotic syndrome

and nephrosis (N...

Nephritis, nephrotic syndrome

and nephrosis (N...

Kidney disease

Kidney disease

Virginia 1035 16.9

Washington 278 5.2

Nephritis, nephrotic syndrome

and nephrosis (N...

Kidney disease

West Virginia

345 16.4

Nephritis, nephrotic syndrome

and nephrosis (N...

Nephritis, nephrotic syndrome

Kidney disease

Kidney

Wisconsin 677 11.9

and nephrosis (N...

disease Wyoming 30 6.8

In [5]:

df.shape

Out[5]:

(10868, 6)

In [6]:

df.columns

Out[6]:

Index(['Year', '113 Cause Name', 'Cause Name', 'State', 'Deaths', 'Age-adjusted Death Rate'],

dtype='object')

In [7]:

df.duplicated().sum()

Out[7]:

0

In [8]:

df.isnull().sum()

Out[8]:

Year 0

113 Cause Name 0

Cause Name 0

State 0

Deaths 0

Age-adjusted Death Rate 0

dtype: int64

In [9]:

df.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 10868 entries, 0 to 10867 Data columns (total 6 columns):

# Column Non-Null Count Dtype

1. Year 10868 non-null int64
2. 113 Cause Name 10868 non-null object
3. Cause Name 10868 non-null object
4. State 10868 non-null object
5. Deaths 10868 non-null int64
6. Age-adjusted Death Rate 10868 non-null float64 dtypes: float64(1), int64(2), object(3)

memory usage: 509.6+ KB

In [10]:

df.describe()

Out[10]:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Year** | **Deaths** | **Age-adjusted Death Rate** |
| **count** | 10868.000000 | 1.086800e+04 | 10868.000000 |
| **mean** | 2008.000000 | 1.545991e+04 | 127.563894 |
| **std** | 5.477478 | 1.128760e+05 | 223.639771 |
| **min** | 1999.000000 | 2.100000e+01 | 2.600000 |
| **25%** | 2003.000000 | 6.120000e+02 | 19.200000 |
| **50%** | 2008.000000 | 1.718500e+03 | 35.900000 |
| **75%** | 2013.000000 | 5.756500e+03 | 151.725000 |
| **max** | 2017.000000 | 2.813503e+06 | 1087.300000 |

In [11]:

df.nunique()

Out[11]:

Year 19

113 Cause Name 11

Cause Name 11

State 52

Deaths 5964

Age-adjusted Death Rate 2490

dtype: int64

In [12]:

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

In [13]:

**import** warnings

warnings.filterwarnings("ignore")

In [14]:

df['Year'].unique()

Out[14]:

array([2017, 2007, 2016, 2015, 2014, 2013, 2012, 2011, 2010, 2009, 2008,

2006, 2005, 2004, 2003, 2002, 2001, 2000, 1999], dtype=int64)

|  |  |
| --- | --- |
| 2017 | 572 |
| 2008 | 572 |
| 2000 | 572 |
| 2001 | 572 |
| 2002 | 572 |
| 2003 | 572 |
| 2004 | 572 |
| 2005 | 572 |
| 2006 | 572 |
| 2009 | 572 |
| 2007 | 572 |
| 2010 | 572 |
| 2011 | 572 |
| 2012 | 572 |
| 2013 | 572 |
| 2014 | 572 |
| 2015 | 572 |
| 2016 | 572 |
| 1999 | 572 |

Name: Year, dtype: int64

In [16]:

df['113 Cause Name'].unique()

Out[16]:

array(['Accidents (unintentional injuries) (V01-X59,Y85-Y86)', 'All Causes', "Alzheimer's disease (G30)",

'Cerebrovascular diseases (I60-I69)',

'Chronic lower respiratory diseases (J40-J47)', 'Diabetes mellitus (E10-E14)',

'Diseases of heart (I00-I09,I11,I13,I20-I51)', 'Influenza and pneumonia (J09-J18)',

'Intentional self-harm (suicide) (\*U03,X60-X84,Y87.0)', 'Malignant neoplasms (C00-C97)',

'Nephritis, nephrotic syndrome and nephrosis (N00-N07,N17-N19,N25-N2

7)'],

dtype=object)

Out[17]:

Accidents (unintentional injuries) (V01-X59,Y85-Y86) 988

All Causes 988

Alzheimer's disease (G30) 988

Cerebrovascular diseases (I60-I69) 988

Chronic lower respiratory diseases (J40-J47) 988

Diabetes mellitus (E10-E14) 988

Diseases of heart (I00-I09,I11,I13,I20-I51) 988

Influenza and pneumonia (J09-J18) 988

Intentional self-harm (suicide) (\*U03,X60-X84,Y87.0) 988

Malignant neoplasms (C00-C97) 988

Nephritis, nephrotic syndrome and nephrosis (N00-N07,N17-N19,N25-N27) 988

Name: 113 Cause Name, dtype: int64

In [18]:

df['Cause Name'].unique()

Out[18]:

array(['Unintentional injuries', 'All causes', "Alzheimer's disease", 'Stroke', 'CLRD', 'Diabetes', 'Heart disease',

'Influenza and pneumonia', 'Suicide', 'Cancer', 'Kidney disease'], dtype=object)

In [19]:

df['Cause Name'].value\_counts()

Out[19]:

Unintentional injuries 988

All causes 988

Alzheimer's disease 988

Stroke 988

CLRD 988

Diabetes 988

Heart disease 988

Influenza and pneumonia 988

Suicide 988

Cancer 988

Kidney disease 988

Name: Cause Name, dtype: int64

df['State'].unique()

Out[20]:

array(['United States', 'Alabama', 'Alaska', 'Arizona', 'Arkansas', 'California', 'Colorado', 'Connecticut', 'Delaware',

'District of Columbia', 'Florida', 'Georgia', 'Hawaii', 'Idaho', 'Illinois', 'Indiana', 'Iowa', 'Kansas', 'Kentucky', 'Louisiana', 'Maine', 'Maryland', 'Massachusetts', 'Michigan', 'Minnesota',

'Mississippi', 'Missouri', 'Montana', 'Nebraska', 'Nevada', 'New Hampshire', 'New Jersey', 'New Mexico', 'New York',

'North Carolina', 'North Dakota', 'Ohio', 'Oklahoma', 'Oregon',

'Pennsylvania', 'Rhode Island', 'South Carolina', 'South Dakota', 'Tennessee', 'Texas', 'Utah', 'Vermont', 'Virginia', 'Washington', 'West Virginia', 'Wisconsin', 'Wyoming'], dtype=object)

df['State'].value\_counts()

Out[21]:

United States 209

Alabama 209

Nebraska 209

Nevada 209

New Hampshire 209

New Jersey 209

New Mexico 209

New York 209

North Carolina 209

North Dakota 209

Ohio 209

Oklahoma 209

Oregon 209

Pennsylvania 209

Rhode Island 209

South Carolina 209

South Dakota 209

Tennessee 209

Texas 209

Utah 209

Vermont 209

Virginia 209

Washington 209

West Virginia 209

Wisconsin 209

Montana 209

Missouri 209

Mississippi 209

Hawaii 209

Alaska 209

Arizona 209

Arkansas 209

California 209

Colorado 209

Connecticut 209

Delaware 209

District of Columbia 209

Florida 209

Georgia 209

Idaho 209

Minnesota 209

Illinois 209

Indiana 209

Iowa 209

Kansas 209

Kentucky 209

Louisiana 209

Maine 209

Maryland 209

Massachusetts 209

Michigan 209

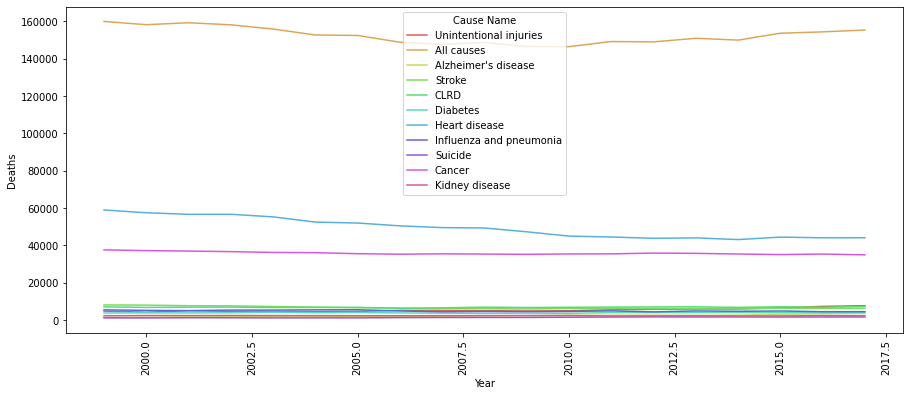
Wyoming 209

Name: State, dtype: int64

plt.figure(figsize**=**(15,6))

sns.lineplot(data**=**df[df['State']**==**'New York'], x**=**'Year', y**=**'Deaths', hue**=**'Cause Name', palette **=** 'hls')

plt.xticks(rotation **=** 90) plt.show()

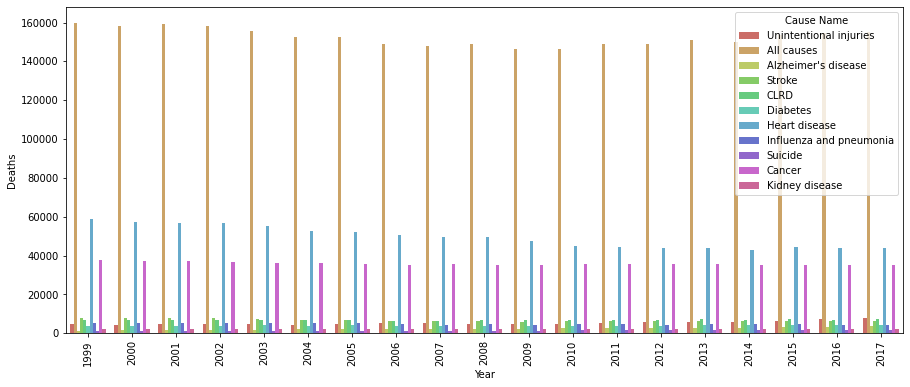


In [23]:

plt.figure(figsize**=**(15,6))

sns.barplot(data**=**df[df['State']**==**'New York'], x**=**'Year', y**=**'Deaths', hue**=**'Cause Name', palette **=** 'hls')

plt.xticks(rotation **=** 90) plt.show()



In [24]:

**import** plotly.express **as** px

fig **=** px.line(df[df['State']**==**'New York'], x**=**'Year', y**=**'Deaths', color **=**'Cause Name') fig.show()

160k

140k

120k

100k

Deaths

80k 60k

Cause Name

Unintentional injuries All causes Alzheimer's disease Stroke

CLRD

Diabetes Heart disease

Influenza and pneumonia Suicide

Cancer

Kidney disease

40k

20k

0

2000 2005 2010 2015

Year

fig **=** px.bar(df[df['State']**==**'New York'], x**=**'Year', y**=**'Deaths', color **=**'Cause Name') fig.show()

300k

250k

200k

150k

Deaths

Cause Name

Unintentional injuries All causes Alzheimer's disease Stroke

CLRD

Diabetes Heart disease

Influenza and pneumonia Suicide

Cancer

Kidney disease

100k

50k

0 2000 2005 2010 2015

Year

plt.figure(figsize**=**(15,6))

sns.barplot(data**=**df[df['State']**==**'New York'], x**=**'Cause Name', y**=**'Deaths', hue**=**'Year', palette **=** 'hls')

plt.xticks(rotation **=** 90) plt.show()

Chart, histogram

Description automatically generated

In [28]:

plt.figure(figsize**=**(15,6))

sns.lineplot(data**=**df[df['State']**==**'New York'], x**=**'Cause Name', y**=**'Deaths', hue**=**'Year', palette **=** 'hls')

plt.xticks(rotation **=** 90) plt.show()

Line chart

Description automatically generated

fig **=** px.line(df[df['State']**==**'New York'], x**=**'Cause Name', y**=**'Deaths', color **=**'Year') fig.show()

160k

140k

120k

100k

Deaths

80k 60k 40k 20k 0

Cancer

Suicide

Influenza and pneumonia

Heart disease

Diabetes

CLRD

Stroke

Alzheimer's disease

All causes

Unintentional injuries

Year

2017

2016

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

2015

2014

2013

2012

2011

2010

2009

2007

2008

2006

2005

2004

2003

2002

2001

Kidney disease

Cause Name

fig **=** px.bar(df[df['State']**==**'New York'], x**=**'Cause Name', y**=**'Deaths', color **=**'Year') fig.show()

3M

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |
|  |
|  |  |
|  |
|  |
|  |
|  |  |  |
|  |
|  |
|  |  |  |
|  |
|  |
|  |
|  |  |
|  |
|  |
|  |
|  |  |  |
|  |
|  |

2.5M

2M

Deaths

1.5M

1M

0.5M

0

Kidney disease

Cancer

Suicide

Influenza and pneumonia

Heart disease

Diabetes

CLRD

Stroke

Alzheimer's disease

All causes

Unintentional injuries

Year



2015

2010

2005

2000

Cause Name

In [31]:

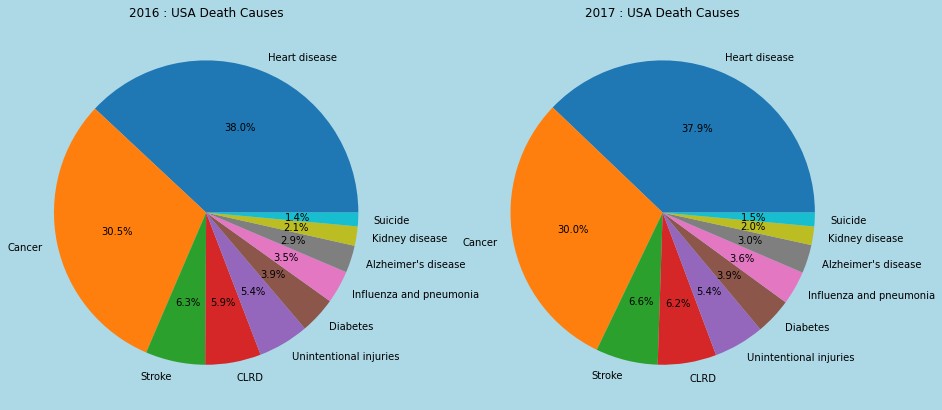
**import** numpy **as** np

|  |  |  |
| --- | --- | --- |
| label**=**['Heart disease', 'Cancer', 'Stroke', 'CLRD',  'Unintentional injuries', 'Diabetes', 'Influenza and pneumonia', "Alzheimer's disease", 'Kidney disease', 'Suicide']  y1**=**df[(df['State']**==**'United States') **&** (df['Year']**==**1999)].sort\_values(by**=**'Deaths',  ascending**=False**) y2**=**df[(df['State']**==**'United States') **&** (df['Year']**==**2017)].sort\_values(by**=**'Deaths',  ascending**=False**)  x **=** np.arange(len(label)) *# the label locations*  width **=** 0.4 *# the width of the bars*  fig, ax **=** plt.subplots(figsize **=** (15,10), facecolor**=**'lightpink')  rects1 **=** ax.bar(x **-** width**/**2, y1, width, label**=**'1999 Deaths by Cause Name') rects2 **=** ax.bar(x **+** width**/**2, y2, width, label**=**'2017 Deaths by Cause Name')  ax.set\_ylabel('The number of death') ax.set\_title('Deaths by Cause Name') ax.set\_xticks(x, label, rotation**=**45) ax.legend()  ax.bar\_label(rects1, padding**=**3) ax.bar\_label(rects2, padding**=**3)  fig.tight\_layout() plt.show() | | |
|  |  |  |

Chart, bar chart

Description automatically generated

|  |  |  |
| --- | --- | --- |
| fig **=** plt.figure(figsize **=** (15,10), facecolor**=**'lightblue')  ax1 **=** fig.add\_subplot(1, 2, 1)  ax2 **=** fig.add\_subplot(1, 2, 2)  ax1.pie(df[(df['State']**==**'New York') **&** (df['Year']**==**2016)].sort\_values(by**=**'Deaths',  ascending**=False**)  ax2.pie(df[(df['State']**==**'New York') **&** (df['Year']**==**2017)].sort\_values(by**=**'Deaths',  ascending**=False**)  ax1.set\_title("2016 : USA Death Causes") ax2.set\_title("2017 : USA Death Causes") plt.show(); | | |
|  |  |  |



In [34]:

df\_state**=**df[df['State']**!=**'United States'].sort\_values(by**=**['Year'])

df\_state

Out[35]:

**113 Cause Name Cause Name State Deaths Age-adjusted**

|  |  |
| --- | --- |
|  | **Year** |
| **10867** | 1999 |
| **10475** | 1999 |
| **10476** | 1999 |
| **10477** | 1999 |
| **10478** | 1999 |
| **...** | ... |
| **380** | 2017 |
| **379** | 2017 |
| **378** | 2017 |
| **384** | 2017 |
| **1** | 2017 |

**Death Rate**

Nephritis, nephrotic syndrome

and nephrosis (N...

Cerebrovascular diseases (I60-

I69)

Cerebrovascular diseases (I60-

I69)

Cerebrovascular diseases (I60-

I69)

Cerebrovascular diseases (I60-

I69)

Kidney disease Wyoming 30 6.8

Stroke Minnesota 2997 59.6

Stroke Mississippi 1854 69.2

Stroke Missouri 3950 65.6

Stroke Montana 595 62.2

... ... ... ... ...

Influenza and pneumonia (J09-

J18)

Influenza and pneumonia (J09-

J18)

Influenza and pneumonia (J09-

J18)

Influenza and pneumonia (J09-

J18)

Accidents (unintentional injuries) (V01-X59,Y8...

Influenza and Iowa 578 13.2

pneumonia

Influenza and Indiana 1078 13.8

pneumonia

Influenza and Illinois 2402 15.6

pneumonia

Influenza and Maine 301 15.2

pneumonia

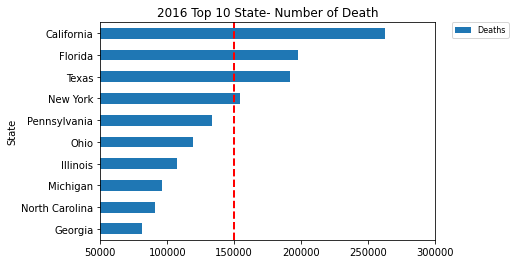
Unintentional Alabama 2703 53.8

injuries

10659 rows × 6 columns

|  |  |  |
| --- | --- | --- |
| df\_state[(df\_state['Cause Name']**==**'All causes')**&**(df\_state['Year']**==**2016)].sort\_values(by**=**'De plt.xlim(50000,300000 )  plt.legend(bbox\_to\_anchor**=**(1.05, 1), loc**=**'upper left', borderaxespad**=**0, fontsize**=**8) plt.axvline(x**=**150000,color**=**'red',lw**=**2,ls**=**'--',alpha**=**1)  plt.title('2016 Top 10 State- Number of Death')  df\_state[(df\_state['Cause Name']**==**'All causes')**&**(df\_state['Year']**==**2017)].sort\_values(by**=**'De plt.xlim(50000, 300000)  plt.legend(bbox\_to\_anchor**=**(1.05, 1), loc**=**'upper left', borderaxespad**=**0, fontsize**=**8) plt.axvline(x**=**150000,color**=**'red',lw**=**2,ls**=**'--',alpha**=**1)  plt.title('2017 Top 10 State- Number of Death'); | | |
|  |  |  |

Chart, bar chart

Description automatically generated

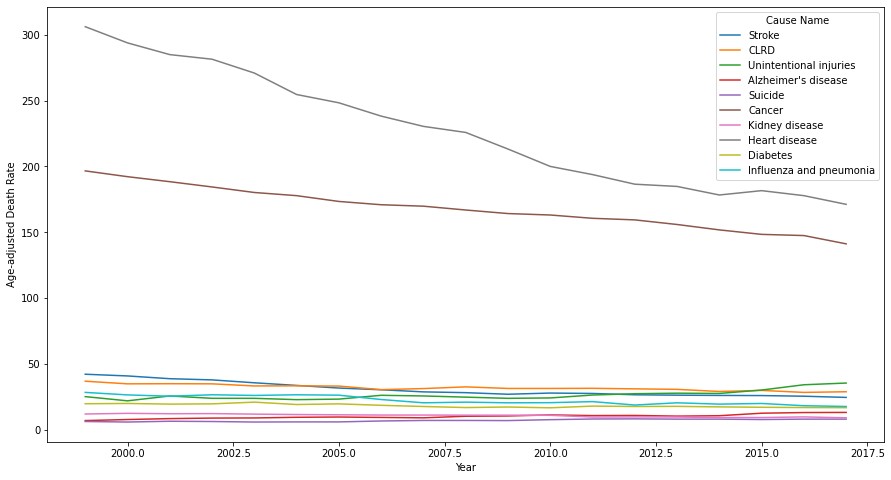
|  |  |  |
| --- | --- | --- |
| df\_state[(df\_state['Cause Name']**==**'All causes')**&**(df\_state['Year']**==**2016)].sort\_values(by**=**'Ag plt.xlim(800,1200 )  plt.legend(bbox\_to\_anchor**=**(1.05, 1), loc**=**'upper left', borderaxespad**=**0, fontsize**=**8) plt.axvline(x**=**1000,color**=**'red',lw**=**2,ls**=**'--',alpha**=**1)  plt.title('2016 Top 10 State- Age-adjusted Death Rate')  df\_state[(df\_state['Cause Name']**==**'All causes')**&**(df\_state['Year']**==**2017)].sort\_values(by**=**'Ag plt.xlim(800, 1200)  plt.legend(bbox\_to\_anchor**=**(1.05, 1), loc**=**'upper left', borderaxespad**=**0, fontsize**=**8) plt.axvline(x**=**1000,color**=**'red',lw**=**2,ls**=**'--',alpha**=**1)  plt.title('2017 Top 10 State- Age-adjusted Death Rate'); | | |
|  |  |  |

Chart

Description automatically generated with low confidenceA picture containing chart

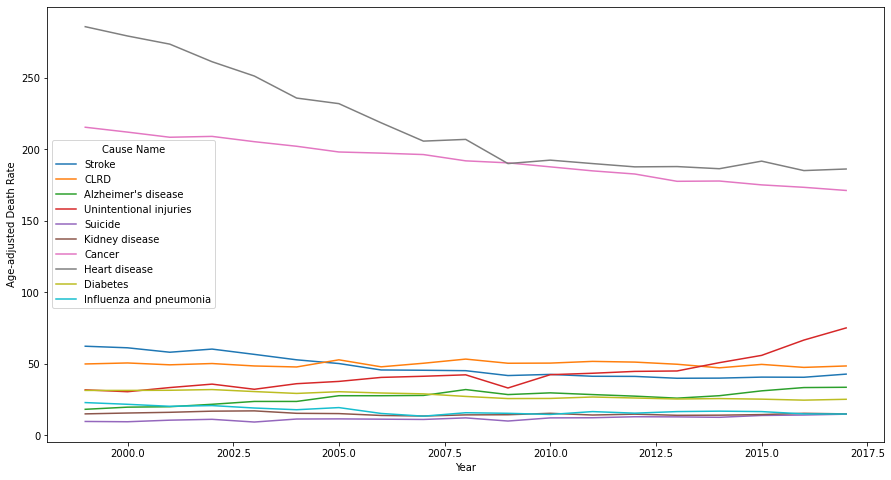
Description automatically generated

|  |  |  |
| --- | --- | --- |
| plt.figure(figsize**=**(15,8))  sns.lineplot(data**=**df\_state[(df\_state['State']**==**'New York')**&**(df\_state['Cause Name']**!=**'All cau x**=**'Year',y**=**'Age-adjusted Death Rate',hue**=**'Cause Name');  plt.show() | | |
|  |  |  |



In [39]:

|  |  |  |
| --- | --- | --- |
| plt.figure(figsize**=**(15,8))  sns.lineplot(data**=**df\_state[(df\_state['State']**==**'Ohio')**&**(df\_state['Cause Name']**!=**'All causes' x**=**'Year',y**=**'Age-adjusted Death Rate',hue**=**'Cause Name');  plt.show() | | |
|  |  |  |



|  |  |  |
| --- | --- | --- |
| label**=**["Alzheimer's disease",'CLRD','Cancer','Diabetes','Heart disease','Influenza and pneum 'Stroke','Suicide','Unintentional injuries']  y1**=**df[(df['State']**==**'United States') **&** (df['Year']**==**2017)**&**(df['Cause Name']**!=**'All causes')].  ascending**=True**)[ y2**=**df[(df['State']**==**'New York') **&** (df['Year']**==**2017)**&**(df['Cause Name']**!=**'All causes')].sort\_  ascending**=True**)[  x **=** np.arange(len(label)) *# the label locations*  width **=** 0.4 *# the width of the bars*  fig, ax **=** plt.subplots(figsize **=** (15,10), facecolor**=**'lightpink')  rects1 **=** ax.bar(x **-** width**/**2, y1, width, label**=**'USA 2017 Age-adjusted Death Rate')  rects2 **=** ax.bar(x **+** width**/**2, y2, width, label**=**'New York 2017 Age-adjusted Death Rate')  ax.set\_ylabel('Age-adjusted Death Rate') ax.set\_title('Deaths by Cause Name')  ax.set\_xticks(x, label, rotation**=**45) ax.legend()  ax.bar\_label(rects1, padding**=**3) ax.bar\_label(rects2, padding**=**3)  fig.tight\_layout() plt.show() | | |
|  |  |  |

Chart, bar chart

Description automatically generated

|  |  |  |
| --- | --- | --- |
| fig **=** plt.figure(figsize **=** (15,10), facecolor**=**'lightblue')  ax1 **=** fig.add\_subplot(1, 2, 1)  ax2 **=** fig.add\_subplot(1, 2, 2)  ax1.pie(df[(df['State']**==**'New York') **&** (df['Year']**==**2016)].sort\_values(by**=**'Deaths',  ascending**=False**)  ax2.pie(df[(df['State']**==**'New York') **&** (df['Year']**==**2017)].sort\_values(by**=**'Deaths',  ascending**=False**)  ax1.set\_title("2016 : New York Death Causes") ax2.set\_title("2017 : New York Death Causes") plt.show(); | | |
|  |  |  |

Chart, pie chart

Description automatically generated

In [42]:

df\_clus**=**df\_state[df\_state['Year']**==**2017]

df\_clus**=**df\_clus.drop(['Year','113 Cause Name'],axis**=**1)

Out[43]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Cause Name** | **State** | **Deaths** | **Age-adjusted Death Rate** |
| **193** | Stroke | Oklahoma | 1947 | 43.3 |
| **192** | Stroke | Ohio | 6425 | 42.8 |
| **191** | Stroke | North Dakota | 337 | 35.4 |
| **190** | Stroke | North Carolina | 5098 | 43.0 |
| **189** | Stroke | New York | 6264 | 24.6 |
| **...** | ... | ... | ... | ... |
| **380** | Influenza and pneumonia | Iowa | 578 | 13.2 |
| **379** | Influenza and pneumonia | Indiana | 1078 | 13.8 |
| **378** | Influenza and pneumonia | Illinois | 2402 | 15.6 |
| **384** | Influenza and pneumonia | Maine | 301 | 15.2 |
| **1** | Unintentional injuries | Alabama | 2703 | 53.8 |

561 rows × 4 columns

In [44]:

df\_clus**=**df\_clus.pivot(index**=**'State', columns**=**'Cause Name', values**=**['Deaths', 'Age-adjusted D

In [45]:

df\_clus

Out[45]:

**Cause Name**

**Deaths**

**All causes**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **State** |  |  |  |  |  |  | **pneumonia** |  |
| **Alabama** | 53238.0 | 2563.0 | 3484.0 | 10410.0 | 1173.0 | 13110.0 | 1176.0 | 980.0 |
| **Alaska** | 4411.0 | 98.0 | 204.0 | 926.0 | 130.0 | 814.0 | 66.0 | 54.0 |
| **Arizona** | 57758.0 | 3058.0 | 3802.0 | 12008.0 | 2054.0 | 12398.0 | 876.0 | 540.0 |
| **Arkansas** | 32588.0 | 1436.0 | 2517.0 | 6517.0 | 1180.0 | 8270.0 | 720.0 | 725.0 |
| **California** | 268189.0 | 16238.0 | 13881.0 | 59516.0 | 9595.0 | 62797.0 | 6340.0 | 3887.0 |
| **Colorado** | 38063.0 | 1830.0 | 2604.0 | 7829.0 | 1017.0 | 7060.0 | 577.0 | 503.0 |
| **Connecticut** | 31312.0 | 1077.0 | 1471.0 | 6608.0 | 694.0 | 7138.0 | 675.0 | 554.0 |
| **Delaware** | 9178.0 | 377.0 | 526.0 | 2085.0 | 244.0 | 1990.0 | 184.0 | 205.0 |
| **District of Columbia** | 4965.0 | 125.0 | 137.0 | 1031.0 | 138.0 | 1284.0 | 79.0 | 67.0 |
| **Florida** | 203636.0 | 6980.0 | 12619.0 | 45131.0 | 6172.0 | 46440.0 | 3057.0 | 3172.0 |
| **Georgia** | 83098.0 | 4290.0 | 4866.0 | 17135.0 | 2348.0 | 18389.0 | 1400.0 | 1942.0 |
| **Hawaii** | 11390.0 | 465.0 | 378.0 | 2456.0 | 299.0 | 2575.0 | 637.0 | 208.0 |
| **Idaho** | 14011.0 | 672.0 | 925.0 | 3020.0 | 394.0 | 3084.0 | 255.0 | 170.0 |
| **Illinois** | 109721.0 | 4021.0 | 5732.0 | 24150.0 | 2927.0 | 25394.0 | 2402.0 | 2565.0 |
| **Indiana** | 65597.0 | 2771.0 | 4375.0 | 13462.0 | 2096.0 | 14445.0 | 1078.0 | 1440.0 |
| **Iowa** | 30530.0 | 1597.0 | 1939.0 | 6449.0 | 918.0 | 7180.0 | 578.0 | 367.0 |
| **Kansas** | 27063.0 | 894.0 | 1832.0 | 5494.0 | 874.0 | 5723.0 | 546.0 | 541.0 |
| **Kentucky** | 48212.0 | 1765.0 | 3480.0 | 10145.0 | 1474.0 | 10343.0 | 932.0 | 1024.0 |
| **Louisiana** | 45804.0 | 2188.0 | 2467.0 | 9513.0 | 1272.0 | 11260.0 | 785.0 | 1076.0 |
| **Maine** | 14676.0 | 601.0 | 982.0 | 3391.0 | 395.0 | 2844.0 | 301.0 | 260.0 |
| **Maryland** | 49926.0 | 1191.0 | 2079.0 | 10796.0 | 1439.0 | 11653.0 | 990.0 | 830.0 |
| **Massachusetts** | 58803.0 | 1841.0 | 2842.0 | 12934.0 | 1321.0 | 12140.0 | 1433.0 | 1193.0 |
| **Michigan** | 97602.0 | 4428.0 | 5688.0 | 20671.0 | 2798.0 | 25187.0 | 1798.0 | 1874.0 |
| **Minnesota** | 44371.0 | 2474.0 | 2464.0 | 9896.0 | 1312.0 | 8230.0 | 697.0 | 537.0 |
| **Mississippi** | 32280.0 | 1626.0 | 2037.0 | 6526.0 | 1164.0 | 7944.0 | 782.0 | 741.0 |
| **Missouri** | 61876.0 | 2545.0 | 3940.0 | 12971.0 | 1605.0 | 14820.0 | 1281.0 | 1515.0 |
| **Montana** | 10200.0 | 285.0 | 752.0 | 2145.0 | 292.0 | 2164.0 | 185.0 | 129.0 |
| **Nebraska** | 16878.0 | 698.0 | 1224.0 | 3502.0 | 575.0 | 3581.0 | 393.0 | 226.0 |
| **Nevada** | 24657.0 | 779.0 | 1633.0 | 5283.0 | 609.0 | 6417.0 | 636.0 | 299.0 |
| **New Hampshire** | 12504.0 | 436.0 | 755.0 | 2760.0 | 340.0 | 2721.0 | 230.0 | 169.0 |
| **New Jersey** | 74846.0 | 2829.0 | 3227.0 | 16264.0 | 1908.0 | 18840.0 | 1337.0 | 1591.0 |
| **New Mexico** | 18673.0 | 572.0 | 1143.0 | 3620.0 | 673.0 | 3896.0 | 338.0 | 330.0 |
| **New York** | 155358.0 | 3521.0 | 7258.0 | 34956.0 | 4176.0 | 44092.0 | 4517.0 | 2296.0 |
| **North Carolina** | 93157.0 | 4289.0 | 5540.0 | 19474.0 | 2903.0 | 18808.0 | 2076.0 | 2040.0 |

**Alzheimer's CLRD Cancer Diabetes disease**

**Heart disease**

**Influenza and**

**Kidney disease**

**Cause Name**

**State**

**Deaths**

**All causes**

**Alzheimer's CLRD Cancer Diabetes disease**

**Heart disease**

**Influenza and pneumonia**

**Kidney disease**

**North Dakota** 6415.0 387.0 358.0 1280.0 194.0 1326.0 147.0 122.0

**Ohio** 123648.0 5117.0 7312.0 25643.0 3740.0 28008.0 2243.0 2237.0

**Oklahoma** 40452.0 1752.0 3035.0 8203.0 1398.0 10772.0 625.0 461.0

**Oregon** 36624.0 1850.0 2088.0 8083.0 1243.0 6942.0 573.0 377.0

**Pennsylvania** 135656.0 4213.0 6667.0 28387.0 3704.0 32312.0 2718.0 2898.0

**Rhode Island** 10157.0 435.0 521.0 2154.0 275.0 2339.0 206.0 176.0

**South Carolina** 49441.0 2549.0 2983.0 10356.0 1535.0 10418.0 723.0 950.0

**South Dakota** 7996.0 444.0 505.0 1715.0 263.0 1710.0 217.0 75.0

**Tennessee** 70096.0 3522.0 4657.0 14302.0 1915.0 16019.0 1656.0 1140.0

**Texas** 198106.0 9545.0 10650.0 40668.0 5832.0 45346.0 2954.0 4256.0

**Utah** 18035.0 991.0 826.0 3161.0 596.0 3749.0 334.0 384.0

**Vermont** 6007.0 370.0 375.0 1434.0 163.0 1332.0 86.0 29.0

**Virginia** 68579.0 2549.0 3363.0 15064.0 1967.0 14861.0 1245.0 1618.0

**Washington** 56995.0 3710.0 3177.0 12664.0 1812.0 11582.0 1041.0 439.0

**West Virginia** 23276.0 770.0 1681.0 4654.0 864.0 4849.0 458.0 436.0

In [46]:

**Wisconsin** 52681.0

2428.0 2834.0 11318.0 1433.0 11860.0 974.0 922.0

**def** get\_converted\_multi\_columns(df\_clus, **\***, to\_snake\_case**=True**):

**if** to\_snake\_case:

**Wyoming** 4768.0 212.0 366.0 948.0 121.0 1001.0 115.0

63.0

**return** [col[0] **+** '\_' **+** col[1] **for** col **in** df.columns.values]

51 ro**e**w**l**s**s**×**e**:22 columns

**return** [col[0] **+** col[1].capitalize() **for** col **in** df\_clus.columns.values]

In [47]:

print(get\_converted\_multi\_columns(df\_clus, to\_snake\_case**=False**))

['DeathsAll causes', "DeathsAlzheimer's disease", 'DeathsClrd', 'DeathsCance r', 'DeathsDiabetes', 'DeathsHeart disease', 'DeathsInfluenza and pneumonia', 'DeathsKidney disease', 'DeathsStroke', 'DeathsSuicide', 'DeathsUnintentional

injuries', 'Age-adjusted Death RateAll causes', "Age-adjusted Death RateAlzhei mer's disease", 'Age-adjusted Death RateClrd', 'Age-adjusted Death RateCance

r', 'Age-adjusted Death RateDiabetes', 'Age-adjusted Death RateHeart disease', 'Age-adjusted Death RateInfluenza and pneumonia', 'Age-adjusted Death RateKidn ey disease', 'Age-adjusted Death RateStroke', 'Age-adjusted Death RateSuicid

e', 'Age-adjusted Death RateUnintentional injuries']

In [48]:

df\_clus.columns**=** get\_converted\_multi\_columns(df\_clus, to\_snake\_case**=False**)

In [49]:

df\_clus

Out[49]:

**DeathsAll causes**

**DeathsAlzheimer's disease**

**DeathsClrd DeathsCancer DeathsDiabetes DeathsH**

**disease**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **State** |  | | | | | |
| **Alabama** | 53238.0 | 2563.0 | 3484.0 | 10410.0 | 1173.0 | 131 |
| **Alaska** | 4411.0 | 98.0 | 204.0 | 926.0 | 130.0 | 8 |
| **Arizona** | 57758.0 | 3058.0 | 3802.0 | 12008.0 | 2054.0 | 123 |
| **Arkansas** | 32588.0 | 1436.0 | 2517.0 | 6517.0 | 1180.0 | 82 |
| **California** | 268189.0 | 16238.0 | 13881.0 | 59516.0 | 9595.0 | 627 |
| **Colorado** | 38063.0 | 1830.0 | 2604.0 | 7829.0 | 1017.0 | 70 |
| **Connecticut** | 31312.0 | 1077.0 | 1471.0 | 6608.0 | 694.0 | 71 |
| **Delaware** | 9178.0 | 377.0 | 526.0 | 2085.0 | 244.0 | 19 |
| **District of Columbia** | 4965.0 | 125.0 | 137.0 | 1031.0 | 138.0 | 12 |
| **Florida** | 203636.0 | 6980.0 | 12619.0 | 45131.0 | 6172.0 | 464 |
| **Georgia** | 83098.0 | 4290.0 | 4866.0 | 17135.0 | 2348.0 | 183 |
| **Hawaii** | 11390.0 | 465.0 | 378.0 | 2456.0 | 299.0 | 25 |
| **Idaho** | 14011.0 | 672.0 | 925.0 | 3020.0 | 394.0 | 30 |
| **Illinois** | 109721.0 | 4021.0 | 5732.0 | 24150.0 | 2927.0 | 253 |
| **Indiana** | 65597.0 | 2771.0 | 4375.0 | 13462.0 | 2096.0 | 144 |
| **Iowa** | 30530.0 | 1597.0 | 1939.0 | 6449.0 | 918.0 | 71 |
| **Kansas** | 27063.0 | 894.0 | 1832.0 | 5494.0 | 874.0 | 57 |
| **Kentucky** | 48212.0 | 1765.0 | 3480.0 | 10145.0 | 1474.0 | 103 |
| **Louisiana** | 45804.0 | 2188.0 | 2467.0 | 9513.0 | 1272.0 | 112 |
| **Maine** | 14676.0 | 601.0 | 982.0 | 3391.0 | 395.0 | 28 |
| **Maryland** | 49926.0 | 1191.0 | 2079.0 | 10796.0 | 1439.0 | 116 |
| **Massachusetts** | 58803.0 | 1841.0 | 2842.0 | 12934.0 | 1321.0 | 121 |
| **Michigan** | 97602.0 | 4428.0 | 5688.0 | 20671.0 | 2798.0 | 251 |
| **Minnesota** | 44371.0 | 2474.0 | 2464.0 | 9896.0 | 1312.0 | 82 |
| **Mississippi** | 32280.0 | 1626.0 | 2037.0 | 6526.0 | 1164.0 | 79 |
| **Missouri** | 61876.0 | 2545.0 | 3940.0 | 12971.0 | 1605.0 | 148 |
| **Montana** | 10200.0 | 285.0 | 752.0 | 2145.0 | 292.0 | 21 |
| **Nebraska** | 16878.0 | 698.0 | 1224.0 | 3502.0 | 575.0 | 35 |
| **Nevada** | 24657.0 | 779.0 | 1633.0 | 5283.0 | 609.0 | 64 |
| **New Hampshire** | 12504.0 | 436.0 | 755.0 | 2760.0 | 340.0 | 27 |
| **New Jersey** | 74846.0 | 2829.0 | 3227.0 | 16264.0 | 1908.0 | 188 |
| **New Mexico** | 18673.0 | 572.0 | 1143.0 | 3620.0 | 673.0 | 38 |
| **New York** | 155358.0 | 3521.0 | 7258.0 | 34956.0 | 4176.0 | 440 |
| **North Carolina** | 93157.0 | 4289.0 | 5540.0 | 19474.0 | 2903.0 | 188 |

**DeathsAll causes**

**DeathsAlzheimer's disease**

**DeathsClrd DeathsCancer DeathsDiabetes DeathsH**

**disease**

sc **=** StandardScaler()

df\_sc **=** sc.fit\_transform(df\_clus)

d5f1\_rsocws**=**×p2d2.DcoaltuamFnrsame(df\_sc, columns**=**df\_clus.columns)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **State** |  | | | | | |
| **North Dakota** | 6415.0 | 387.0 | 358.0 | 1280.0 | 194.0 | 13 |
| **Ohio** | 123648.0 | 5117.0 | 7312.0 | 25643.0 | 3740.0 | 280 |
| **Oklahoma** | 40452.0 | 1752.0 | 3035.0 | 8203.0 | 1398.0 | 107 |
| **Oregon** | 36624.0 | 1850.0 | 2088.0 | 8083.0 | 1243.0 | 69 |
| **Pennsylvania** | 135656.0 | 4213.0 | 6667.0 | 28387.0 | 3704.0 | 323 |
| **Rhode Island** | 10157.0 | 435.0 | 521.0 | 2154.0 | 275.0 | 23 |
| **South Carolina** | 49441.0 | 2549.0 | 2983.0 | 10356.0 | 1535.0 | 104 |
| **South Dakota** | 7996.0 | 444.0 | 505.0 | 1715.0 | 263.0 | 17 |
| **Tennessee** | 70096.0 | 3522.0 | 4657.0 | 14302.0 | 1915.0 | 160 |
| **Texas** | 198106.0 | 9545.0 | 10650.0 | 40668.0 | 5832.0 | 453 |
| **Utah** | 18035.0 | 991.0 | 826.0 | 3161.0 | 596.0 | 37 |
| **Vermont** | 6007.0 | 370.0 | 375.0 | 1434.0 | 163.0 | 13 |
| **Virginia** | 68579.0 | 2549.0 | 3363.0 | 15064.0 | 1967.0 | 148 |
| **Washington** | 56995.0 | 3710.0 | 3177.0 | 12664.0 | 1812.0 | 115 |
| **West Virginia** | 23276.0 | 770.0 | 1681.0 | 4654.0 | 864.0 | 48 |
| In [50]:  **Wisconsin** | 52681.0 | 2428.0 | 2834.0 | 11318.0 | 1433.0 | 118 |
| **from** sklearn.preprocessing **import** StandardScaler  **Wyoming** 4768.0 212.0 366.0 | | | | 948.0 | 121.0 | 10 |

In [51]:

df\_sc

Out[51]:

**DeathsAll**

**DeathsAlzheimer's**

**DeathsClrd DeathsCancer DeathsDiabetes DeathsHeart**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **causes** | **disease** |  |  | **disease** | **and p** |
| **0** -0.034961 | 0.067802 0.114881 | -0.111244 | -0.266131 | 0.031436 |  |
| **1** -0.920035 | -0.847844 -0.984321 | -0.900222 | -0.862413 | -0.900516 |  |
| **2** 0.046971 | 0.251674 0.221450 | 0.021695 | 0.237535 | -0.022528 |  |
| **3** -0.409278 | -0.350832 -0.209182 | -0.435104 | -0.262129 | -0.335402 |  |
| **4** 3.861396 | 5.147503 3.599152 | 3.973905 | 4.548714 | 3.797369 |  |
| **5** -0.310034 | -0.204477 -0.180027 | -0.325958 | -0.355316 | -0.427112 |  |
| **6** -0.432408 | -0.484186 -0.559721 | -0.427534 | -0.539975 | -0.421200 |  |
| **7** -0.833625 | -0.744207 -0.876411 | -0.803804 | -0.797239 | -0.811383 |  |
| **8** -0.909993 | -0.837815 -1.006774 | -0.891487 | -0.857839 | -0.864893 |  |
| **9** 2.691261 | 1.708536 3.176227 | 2.777211 | 2.591790 | 2.557621 |  |
| **10** 0.506302 | 0.709312 0.578021 | 0.448212 | 0.405615 | 0.431548 |  |
| **11** -0.793529 | -0.711519 -0.926010 | -0.772940 | -0.765796 | -0.767044 |  |
| **12** -0.746018 | -0.634627 -0.742698 | -0.726021 | -0.711485 | -0.728465 |  |
| **13** 0.988890 | 0.609389 0.868237 | 1.031793 | 0.736628 | 0.962479 |  |
| **14** 0.189067 | 0.145066 0.413476 | 0.142654 | 0.261547 | 0.132620 |  |
| **15** -0.446583 | -0.291027 -0.402883 | -0.440761 | -0.411914 | -0.418017 |  |
| **16** -0.509428 | -0.552163 -0.438741 | -0.520208 | -0.437069 | -0.528447 |  |
| **17** -0.126066 | -0.228622 0.113541 | -0.133289 | -0.094050 | -0.178283 |  |
| **18** -0.169716 | -0.071495 -0.225938 | -0.185865 | -0.209533 | -0.108781 |  |
| **19** -0.733964 | -0.661000 -0.723596 | -0.695157 | -0.710913 | -0.746656 |  |
| **20** -0.094997 | -0.441839 -0.355966 | -0.079132 | -0.114059 | -0.078994 |  |
| **21** 0.065914 | -0.200391 -0.100267 | 0.098729 | -0.181520 | -0.042083 |  |
| **22** 0.769212 | 0.760573 0.853492 | 0.742373 | 0.662879 | 0.946790 |  |
| **23** -0.195691 | 0.034742 -0.226944 | -0.154003 | -0.186665 | -0.338434 |  |
| **24** -0.414861 | -0.280255 -0.370041 | -0.434355 | -0.271277 | -0.360111 |  |
| **25** 0.121617 | 0.061116 0.267697 | 0.101807 | -0.019158 | 0.161043 |  |
| **26** -0.815099 | -0.778381 -0.800674 | -0.798812 | -0.769798 | -0.798195 |  |
| **27** -0.694049 | -0.624969 -0.642496 | -0.685923 | -0.608007 | -0.690796 |  |
| **28** -0.553041 | -0.594881 -0.505431 | -0.537761 | -0.588569 | -0.475847 |  |
| **29** -0.773335 | -0.722291 -0.799668 | -0.747650 | -0.742356 | -0.755978 |  |
| **30** 0.356721 | 0.166610 0.028755 | 0.375753 | 0.154067 | 0.465731 |  |
| **31** -0.661512 | -0.671773 -0.669641 | -0.676107 | -0.551981 | -0.666921 |  |
| **32** 1.816140 | 0.423660 1.379634 | 1.930749 | 1.450680 | 2.379659 |  |
| **33** 0.688639 | 0.708940 0.803894 | 0.642794 | 0.722907 | 0.463305 |  |
| **34** -0.883709 | -0.740492 -0.932712 | -0.870772 | -0.825824 | -0.861710 |  |
| **35** 1.241341 | 1.016508 1.397731 | 1.155996 | 1.201419 | 1.160602 |  |

**Death**

**DeathsAll**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **causes** | **disease** |  |  |  | **disease** | **and p** |
| **36** -0.266730 | -0.233451 | -0.035589 | -0.294845 | -0.137499 | -0.145768 |  |
| **37** -0.336119 | -0.197048 | -0.352950 | -0.304828 | -0.226112 | -0.436056 |  |
| **38** 1.459007 | 0.680709 | 1.181577 | 1.384271 | 1.180838 | 1.486816 |  |
| **39** -0.815879 | -0.722662 | -0.878087 | -0.798064 | -0.779517 | -0.784931 |  |
| **40** -0.103789 | 0.062602 | -0.053015 | -0.115736 | -0.059176 | -0.172599 |  |
| **41** -0.855051 | -0.719319 | -0.883449 | -0.834584 | -0.786377 | -0.832605 |  |
| **42** 0.270619 | 0.424031 | 0.507980 | 0.212534 | 0.158069 | 0.251918 |  |
| **43** 2.591021 | 2.661329 | 2.516370 | 2.405932 | 2.397412 | 2.474703 |  |
| **44** -0.673077 | -0.516131 | -0.775875 | -0.714291 | -0.596001 | -0.678063 |  |
| **45** -0.891105 | -0.746807 | -0.927015 | -0.857961 | -0.843547 | -0.861255 |  |
| **46** 0.243121 | 0.062602 | 0.074332 | 0.275925 | 0.187797 | 0.164150 |  |
| **47** 0.033141 | 0.493866 | 0.011999 | 0.076268 | 0.099184 | -0.084376 |  |
| **48** -0.578074 | -0.598224 | -0.489345 | -0.590088 | -0.442786 | -0.594691 |  |
| **49** -0.045058  In [52]:  **50** -0.913564 | 0.017655  -0.805498 | -0.102948  -0.930031 | -0.035707  -0.898391 | -0.117490  -0.867558 | -0.063305  -0.886343 |  |
| **from** sklearn.cluster **import** KMeans  51 rows × 22 columns | | | | | | |

**DeathsAlzheimer's**

**DeathsClrd DeathsCancer DeathsDiabetes DeathsHeart**

**Death**

distortions **=** []

**for** i **in** range(1,11):

km **=** KMeans(n\_clusters**=**i,

init**=**'k-means++', n\_init**=**10,

max\_iter**=**300,

random\_state**=**0) km.fit(df\_sc)

distortions.append(km.inertia\_)

plt.plot(range(1,11),distortions,marker**=**'o') plt.xlabel('Number of clusters')

plt.ylabel('Distortion') plt.show()

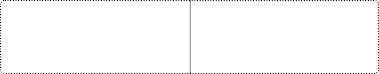
Chart, line chart

Description automatically generated

In [54]:

model **=** KMeans(n\_clusters**=**4, random\_state**=**1) model.fit(df\_sc)

Out[54]:



▾

KMeans

KMeans(n\_clusters=4, random\_state=1)

In [55]:

cluster **=** model.labels\_

In [56]:

df\_clus['Cluster']**=**cluster

df\_clus.groupby('Cluster').mean().style.bar(axis**=**0)

Out[57]:

**DeathsAll**

**DeathsAlzheimer's**

**DeathsClrd DeathsCancer DeathsDiabetes Deaths**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cluster** | **causes** | **disease** |  | **dis** |
| **0** | 24251.678571 | 1067.750000 1383.642857 | 5178.214286 707.785714 | 5169.03 |
| **1** | 104629.444444 | 3917.444444 5517.000000 | 22416.000000 2941.222222 | 25099.00 |
| **2** | 223310.333333 | 10921.000000 12383.333333 | 48438.333333 7199.666667 | 51527.66 |
| **3** | 47532.727273 | 2135.181818 3150.545455 | 9732.636364 1425.090909 | 11113.63 |

In [58]:

df\_clus**=**df\_clus.reset\_index()

In [59]:

col**=**['DeathsAll causes', "DeathsAlzheimer's disease", 'DeathsClrd', 'DeathsCancer', 'DeathsDiabetes', 'DeathsHeart disease',

'DeathsInfluenza and pneumonia', 'DeathsKidney disease', 'DeathsStroke', 'DeathsSuicide', 'DeathsUnintentional injuries',

"Age-adjusted Death RateAll causes",

"Age-adjusted Death RateAlzheimer's disease",

'Age-adjusted Death RateClrd', 'Age-adjusted Death RateCancer', 'Age-adjusted Death RateDiabetes',

'Age-adjusted Death RateHeart disease',

'Age-adjusted Death RateInfluenza and pneumonia', 'Age-adjusted Death RateKidney disease',

'Age-adjusted Death RateStroke', 'Age-adjusted Death RateSuicide', 'Age-adjusted Death RateUnintentional injuries']

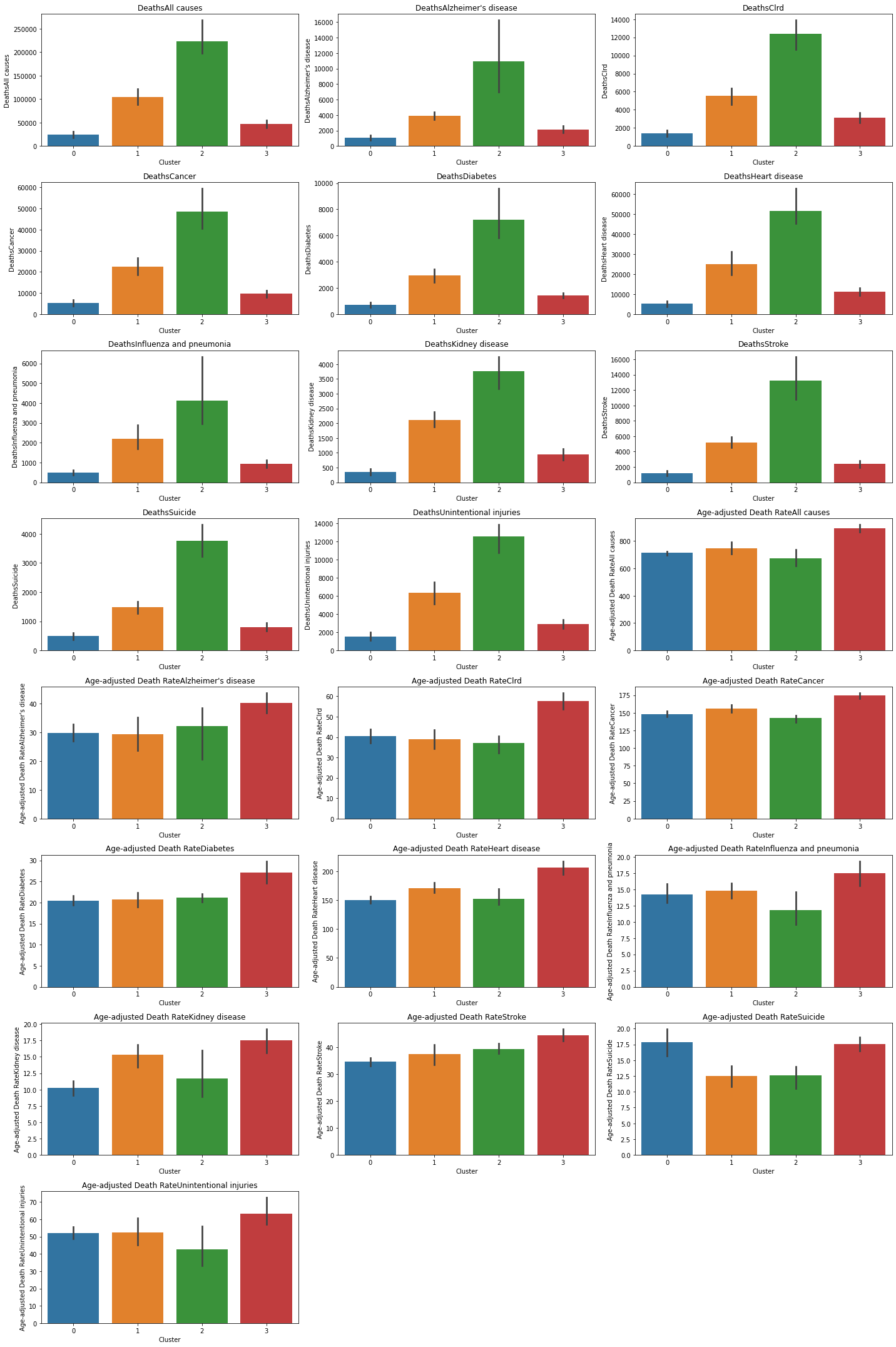
In [60]:

fig **=** plt.figure(figsize**=**(20,30))

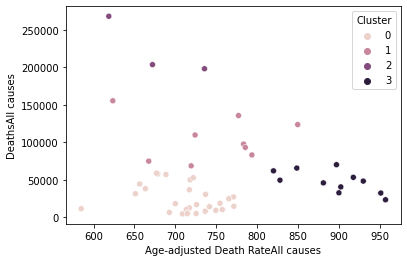
**for** i **in** range(len(col)): plt.subplot(8,3,i**+**1) plt.title(col[i])

sns.barplot(data**=**df\_clus,y**=**df\_clus[col[i]],x**=**df\_clus['Cluster'])

plt.tight\_layout() plt.show()

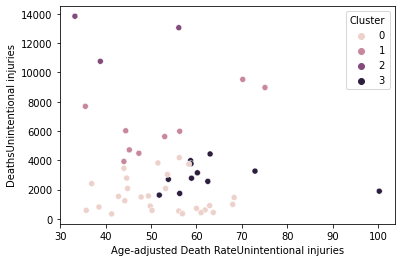


In [61]:



t(data**=**df\_clus, y**=**'DeathsAll causes',x**=**"Age-adjusted Death RateAll causes",hue**=**'Cluster');

In [62]:



hsUnintentional injuries',x**=**'Age-adjusted Death RateUnintentional injuries',hue**=**'Cluster');

State

1. Alaska
2. Arizona
3. Colorado
4. Connecticut
5. Delaware
6. District of Columbia
7. Hawaii
8. Idaho
9. Iowa
10. Kansas
11. Maine
12. Maryland
13. Massachusetts

23 Minnesota

1. Montana
2. Nebraska
3. Nevada
4. New Hampshire

31 New Mexico

34 North Dakota

37 Oregon

39 Rhode Island

41 South Dakota

1. Utah
2. Vermont

47 Washington

1. Wisconsin
2. Wyoming

In [64]:

print(df\_clus[df\_clus['Cluster']**==**1][['State']])

State

10 Georgia

13 Illinois

22 Michigan

30 New Jersey

1. New York
2. North Carolina

35 Ohio

38 Pennsylvania

46 Virginia

In [65]:

print(df\_clus[df\_clus['Cluster']**==**2][['State']])

State

4 California

9 Florida

43 Texas

State

0 Alabama

3 Arkansas

14 Indiana

1. Kentucky
2. Louisiana
3. Mississippi
4. Missouri

36 Oklahoma

40 South Carolina

42 Tennessee

48 West Virginia

In [ ]:

# Email Classification using BERT model:

## Importing Libraries

import tensorflow as tf

import tensorflow\_hub as hub

import tensorflow\_text as text

## Kaggle Dataset

import pandas as pd

df = pd.read\_csv("spam.csv", encoding = "ISO-8859-1")

df.head(5)

|  | **Category** | **Message** | **Unnamed: 2** | **Unnamed: 3** | **Unnamed: 4** |
| --- | --- | --- | --- | --- | --- |
| **0** | ham | Go until jurong point, crazy.. Available only ... | NaN | NaN | NaN |
| **1** | ham | Ok lar... Joking wif u oni... | NaN | NaN | NaN |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | NaN | NaN | NaN |
| **3** | ham | U dun say so early hor... U c already then say... | NaN | NaN | NaN |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | NaN | NaN | NaN |

## Removing Useless Columns

df = df.drop(["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"], axis=1)

df.head(5)

## Count Unique Mails

df.groupby("Category").describe()

## Creating spam Column (new)

df["spam"] = df["Category"].apply(lambda x:1 if x=='spam' else 0)

df.head()

|  | **Category** | **Message** | **spam** |
| --- | --- | --- | --- |
| **0** | ham | Go until jurong point, crazy.. Available only ... | 0 |
| **1** | ham | Ok lar... Joking wif u oni... | 0 |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | 1 |
| **3** | ham | U dun say so early hor... U c already then say... | 0 |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | 0 |

## model and data splitting

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(df['Message'], df['spam'], test\_size=0.2, stratify=df['spam'])

## using our pre-trained model(bert) to create embeddings

## downloading bert model(it is pre-trained model, trained on wikipedia and book corpus)

bert\_preprocessing\_url = hub.KerasLayer("https://tfhub.dev/tensorflow/bert\_en\_uncased\_preprocess/3")

bert\_mainencoder\_url = hub.KerasLayer("https://tfhub.dev/tensorflow/bert\_en\_uncased\_L-12\_H-768\_A-12/4")

## this function takes couple of sentences(array) as input and returns an embedding vector for each sentence

def get\_sentence\_embedding(sentences):

preprocessed\_text = bert\_preprocessing\_url(sentences)

## returns a dictionary from which we need to use the pooled output

return bert\_mainencoder\_url(preprocessed\_text)['pooled\_output']

## pooled output is the encoding for the entire sentence(each)

## generated embedding vector of size 768 for each sentence

get\_sentence\_embedding(["500$ discount! Hurry up.", "Bhavin, are you up for a volleyball game tomorrow?"])

## its embedding vectors

others = get\_sentence\_embedding(["Banana", "Mango", "Grapes", "Jeff Bezos", "Bill Gates", "Elon Musk"])

others

## using cosine similarity(arguments placed inside a 2-d array ie [others[0]]) to compare two vectors

from sklearn.metrics.pairwise import cosine\_similarity

## comparing Banana and Jeff Bezos

cosine\_similarity([others[0]], [others[3]])

## as output is 0.847 so not that much similar

## comparing Mango and Grapes

cosine\_similarity([others[1]], [others[2]])

## as output 0.985 is so highly similar

## comparing Bill Gates and Elon Musk

cosine\_similarity([others[4]], [others[5]])

## cosine similarity is not an exact vector similarity so we might get some unexpected result

## Building our model now

## So far in our deep learning series we have built tensor flow models using sequential model

## Now we are going to use functional model to build our models

## In the sequential model we add layers one by one as a sequence but

## in the functional model first we create a input layer, then we create a hidden layer and then supply input as a

## function argument and then we create another hidden(layer) one and supply that into hidden two layer argument and

## then finally we create a model using multiple input and output layers

## using functional model(using shape=() like this as sentence length is varying)

text\_input\_layer = tf.keras.layers.Input(shape=(), dtype=tf.string, name='input') ## input layer

preprocessed\_text = bert\_preprocessing\_url(text\_input\_layer)

output = bert\_mainencoder\_url(preprocessed\_text)

output['pooled\_output'] ## sentence encoding vector

## creating a dropout(tackles the overfitting) layer and then

## feeding the pooled output into a dropout layer

## and finally the last layer would be a one neuron dense layer

## dropping 10% of neuron

dropout\_layer = tf.keras.layers.Dropout(0.1, name="dropout")(output['pooled\_output'])

## binary(0,1) classification hence sigmoid and we are using functional api hence treat it like a

## function and pass in the previous layer here (one neuron dense layer)

dense\_layer = tf.keras.layers.Dense(1, activation="sigmoid", name="output")(dropout\_layer)

model = tf.keras.Model(inputs=[text\_input\_layer], outputs=[dense\_layer])

model.summary()

## trainable parameters is 769 ie 768neurons+1(1-output neuron), similarly for the non trainable parameters and the total parameters for our already trained bert model

## compiling our builded model(as binary classification so using binary cross entropy)

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

## training the model (takes time)

model.fit(x\_train, y\_train, epochs=10)

## the mean accuracy is 0.95

model.evaluate(x\_test, y\_test)

## Inference(first 3 are spams and rest two are not spams)

emails = ["XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message orclick here>> http://wap. xxxmobilemovieclub.com?n=QJKGIGHJJGCBL",

"Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030",

"WINNER!! As a valued network customer you have been selected to receivea å£900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.",

"Ffffffffff. Alright no way I can meet up with you sooner?",

"I HAVE A DATE ON SUNDAY WITH WILL!!"]

model.predict(emails)

## in sigmoid if value is greater than 0.5 then the email is spam