All Country Scatter

November 13, 2021

[191]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import dataframe_image as dfi
import numpy as np
```

[198]:
```python
from sklearn.metrics import r2_score
```

0.0.1 Scatter Plot to see Trand between VO Consumption and Heart Disease MR

Pulling two tables to merge and format to plot:

AllCountriesVegetableOilConsumptionPerCapita2007__2011

AllCountriesHeartDiseaseMortality2007__2011

[95]:
```python
acvo = pd.read_csv('AllCountriesVegetableOilConsumptionPerCapita2007_2011.csv')
```

[69]:
```python
acmr = pd.read_csv('AllCountriesHeartDiseaseMortality2007_2011.csv')
```

0.0.2 Analyzing Data to convert to usefull format

Our goal is to merge the tables. Therefore, I have to drop countries not contain in either column, and several other data manipulations.

What I need?

A index based on the Years 2007 to 2011

A top row based on the Countries

Values of MR and VO consumption

All Countries VO consumption

[96]:
```python
acvo.shape
```

[96]:
```
(160, 8)
```

[97]:
```python
acvo.head()
```

[97]:
```
   Unnamed: 0    Countries Units   2007   2008  2009  2010  2011
0           1  Afghanistan    kg   3.48   2.39   2.9   3.4   3.2
```

```
1            2      Albania    kg   7.00   6.30   7.0   5.7   5.4
2            3      Algeria    kg  13.60  13.70  12.3  13.3  14.9
3            4       Angola    kg   9.05   8.63   8.9   8.8   8.8
4            5    Argentina    kg  14.60  14.50  14.4  14.5  14.8
```

```
[98]: acvo.drop(columns=['Unnamed: 0', 'Units'], inplace=True)
```

```
[99]: acvo[acvo.loc[:,'2011'] == '...']
```

```
[99]:     Countries  2007  2008  2009  2010 2011
      133     Sudan  5.62  5.48   6.1   5.5    …
```

```
[100]: acvo.drop(133, axis=0, inplace=True)
```

```
[101]: acvo[acvo.loc[:,'2011'] == '...']
```

```
[101]: Empty DataFrame
       Columns: [Countries, 2007, 2008, 2009, 2010, 2011]
       Index: []
```

```
[102]: acvo['2011'] = acvo['2011'].astype('float64')
```

```
[103]: acvo.dtypes
```

```
[103]: Countries     object
       2007         float64
       2008         float64
       2009         float64
       2010         float64
       2011         float64
       dtype: object
```

```
[104]: acvo.rename(columns={'Countries' : 'Entity'}, inplace=True)
```

```
[105]: acvo.head()
```

```
[105]:         Entity   2007   2008  2009  2010  2011
       0  Afghanistan   3.48   2.39   2.9   3.4   3.2
       1      Albania   7.00   6.30   7.0   5.7   5.4
       2      Algeria  13.60  13.70  12.3  13.3  14.9
       3       Angola   9.05   8.63   8.9   8.8   8.8
       4    Argentina  14.60  14.50  14.4  14.5  14.8
```

```
[106]: acvo.columns
```

```
[106]: Index(['Entity', '2007', '2008', '2009', '2010', '2011'], dtype='object')
```

```
[107]: acvo = pd.melt(acvo, id_vars=["Entity"])
```

```
[109]: acvo.head()
```

```
[109]:        Entity variable  value
       0  Afghanistan     2007   3.48
       1      Albania     2007   7.00
       2      Algeria     2007  13.60
       3       Angola     2007   9.05
       4    Argentina     2007  14.60
```

```
[110]: acvo.rename(columns={'variable' : 'Year', 'value' : 'Vegetable Oil'},␣
        ↪inplace=True)
```

```
[111]: acvo.head()
```

```
[111]:        Entity  Year  Vegetable Oil
       0  Afghanistan  2007           3.48
       1      Albania  2007           7.00
       2      Algeria  2007          13.60
       3       Angola  2007           9.05
       4    Argentina  2007          14.60
```

```
 [62]: country_list = acvo.Entity.unique()
```

```
 [64]: country_list = [country for country in country_list]
```

```
 [68]: len(country_list)
```

```
 [68]: 159
```

```
[113]: acvo['Vegetable Oil'].dtypes
```

```
[113]: dtype('float64')
```

```
[115]: acvo['Vegetable Oil'].head()
```

```
[115]: 0     3.48
       1     7.00
       2    13.60
       3     9.05
       4    14.60
       Name: Vegetable Oil, dtype: float64
```

```
[116]: acvo['Vegetable Oil'] *= 1000
```

```
[118]: acvo['Vegetable Oil'] /= 365
```

```
[119]: acvo['Vegetable Oil'] *= 9
```

```
[119]:         Entity  Year  Vegetable Oil
       0  Afghanistan  2007      85.808219
       1      Albania  2007     172.602740
       2      Algeria  2007     335.342466
       3       Angola  2007     223.150685
       4    Argentina  2007     360.000000
```

```
[120]: # Checking data with other tables
       acvo[acvo['Entity'] == 'France']
```

```
[120]:      Entity  Year  Vegetable Oil
       49   France  2007     507.945205
       208  France  2008     517.808219
       367  France  2009     530.136986
       526  France  2010     520.273973
       685  France  2011     517.808219
```

```
[127]: acvo['Year'] = acvo['Year'].astype('int64')
```

All Countries Heart Disease MR

```
[70]: acmr.shape
```

```
[70]: (1155, 5)
```

```
[72]: acmr.head()
```

```
[72]:    Unnamed: 0       Entity Code  Year      Deaths
      0           0  Afghanistan  AFG  2007  707.188774
      1           1  Afghanistan  AFG  2008  693.448663
      2           2  Afghanistan  AFG  2009  677.845507
      3           3  Afghanistan  AFG  2010  662.616946
      4           4  Afghanistan  AFG  2011  649.725628
```

```
[73]: acmr.drop(columns=['Unnamed: 0', 'Code'], inplace=True)
```

```
[74]: acmr.head()
```

```
[74]:         Entity  Year      Deaths
      0  Afghanistan  2007  707.188774
      1  Afghanistan  2008  693.448663
      2  Afghanistan  2009  677.845507
      3  Afghanistan  2010  662.616946
      4  Afghanistan  2011  649.725628
```

```
[45]: # acmr = acmr.pivot(index='Year', columns='Entity', values='Deaths')
```

0.0.3 Merge Data sets

[128]:
```
acmr.dtypes
```

[128]:
```
Entity     object
Year        int64
Deaths    float64
dtype: object
```

[129]:
```
acvo.dtypes
```

[129]:
```
Entity            object
Year               int64
Vegetable Oil    float64
dtype: object
```

[130]:
```
mergeVOMR = pd.merge(acvo, acmr)
```

[133]:
```
mergeVOMR.shape
```

[133]:
```
(740, 4)
```

[160]:
```
mergeVOMR['Entity'].unique()
```

[160]:
```
array(['Afghanistan', 'Albania', 'Algeria', 'Angola', 'Argentina',
       'Armenia', 'Australia', 'Austria', 'Azerbaijan', 'Bahamas',
       'Bangladesh', 'Belarus', 'Belgium', 'Belize', 'Benin', 'Bolivia',
       'Bosnia and Herzegovina', 'Botswana', 'Brazil', 'Brunei',
       'Bulgaria', 'Burkina Faso', 'Cambodia', 'Cameroon', 'Canada',
       'Central African Republic', 'Chad', 'Chile', 'China', 'Colombia',
       'Congo', 'Costa Rica', 'Croatia', 'Cuba', 'Cyprus', 'Denmark',
       'Djibouti', 'Dominican Republic', 'Ecuador', 'Egypt',
       'El Salvador', 'Estonia', 'Ethiopia', 'Fiji', 'Finland', 'France',
       'Gabon', 'Gambia', 'Georgia', 'Germany', 'Ghana', 'Greece',
       'Guatemala', 'Guinea', 'Guyana', 'Haiti', 'Honduras', 'Hungary',
       'Iceland', 'India', 'Indonesia', 'Iraq', 'Iran', 'Ireland',
       'Israel', 'Italy', 'Jamaica', 'Japan', 'Jordan', 'Kazakhstan',
       'Kenya', 'Kuwait', 'Kyrgyzstan', 'Laos', 'Latvia', 'Lebanon',
       'Lesotho', 'Liberia', 'Lithuania', 'Luxembourg', 'Madagascar',
       'Malawi', 'Malaysia', 'Maldives', 'Mali', 'Malta', 'Mauritania',
       'Mauritius', 'Mexico', 'Moldova', 'Mongolia', 'Montenegro',
       'Morocco', 'Mozambique', 'Myanmar', 'Namibia', 'Nepal',
       'Netherlands', 'New Zealand', 'Nicaragua', 'Niger', 'Nigeria',
       'Norway', 'Oman', 'Pakistan', 'Panama', 'Paraguay', 'Peru',
       'Philippines', 'Poland', 'Portugal', 'Romania', 'Russia', 'Rwanda',
       'Saudi Arabia', 'Senegal', 'Serbia', 'Sierra Leone', 'Slovakia',
       'Slovenia', 'South Africa', 'South Korea', 'Spain', 'Sri Lanka',
       'Suriname', 'Sweden', 'Switzerland', 'Taiwan', 'Tajikistan',
       'Tanzania', 'Thailand', 'Togo', 'Trinidad and Tobago', 'Tunisia',
       'Turkey', 'Turkmenistan', 'Uganda', 'Ukraine',
       'United Arab Emirates', 'United Kingdom', 'Uruguay', 'Uzbekistan',
       'Venezuela', 'Vietnam', 'Yemen', 'Zambia', 'Zimbabwe', 'World'],
      dtype=object)
```

Both Data sets contained a total of 147 countries in common (without counting the world row)

```
[138]: len(mergeVOMR.Entity.unique())
```

```
[138]: 148
```

```
[143]: #No null values them
       mergeVOMR.isnull().sum()
```

```
[143]: Entity           0
       Year             0
       Vegetable Oil    0
       Deaths           0
       dtype: int64
```

0.0.4 Time to plot

```
[144]: #here is how the data looks
       mergeVOMR.head()
```

```
[144]:         Entity  Year  Vegetable Oil      Deaths
       0  Afghanistan  2007      85.808219  707.188774
       1      Albania  2007     172.602740  339.928986
       2      Algeria  2007     335.342466  328.078554
       3       Angola  2007     223.150685  344.017796
       4    Argentina  2007     360.000000  220.586059
```

```
[150]: dfi.export(mergeVOMR.loc[:,'Vegetable Oil':].describe(), 'vegetableOilAndDeaths.
        ↪png')
```

```
[232]: mergeVOMR.loc[:,'Vegetable Oil':].describe()
```

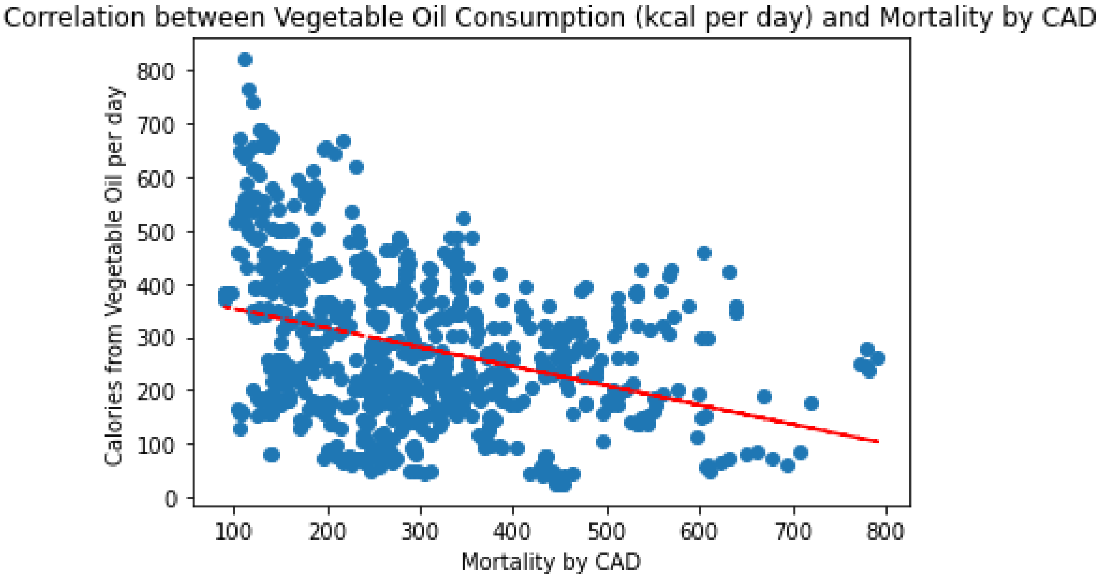
```
[232]:        Vegetable Oil      Deaths
       count     740.000000  740.000000
       mean      284.938467  288.899940
       std       147.470026  136.808543
       min        24.657534   89.784507
       25%       179.691781  181.286112
       50%       261.369863  261.861644
       75%       384.657534  360.329894
       max       821.095890  791.334881
```

[151]:

```
#Setting the X and the Y Variables
x = mergeVOMR['Deaths']
y = mergeVOMR['Vegetable Oil']
```

[197]:

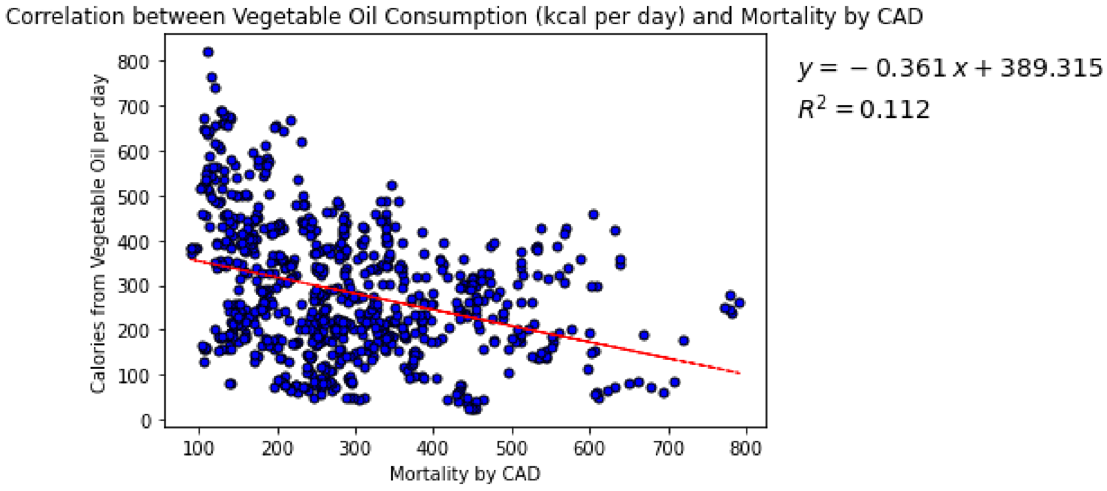
```
plt.title('Correlation between Vegetable Oil Consumption (kcal per day) and␣
 ↪Mortality by CAD')
plt.ylabel('Calories from Vegetable Oil per day')
plt.xlabel('Mortality by CAD')
plt.scatter(x, y)
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"r--")
plt.savefig('CorrelationVOMR.png', dpi=200, bbox_inches='tight')
```



[224]:

```
plt.title('Correlation between Vegetable Oil Consumption (kcal per day) and␣
 ↪Mortality by CAD')
plt.ylabel('Calories from Vegetable Oil per day')
plt.xlabel('Mortality by CAD')
plt.plot(x,y,"b.", ms=10, mec="k")
z = np.polyfit(x, y, 1)
y_hat = np.poly1d(z)(x)

plt.plot(x, y_hat, "r--", lw=1)
text = f"$y={z[0]:0.3f}\;x{z[1]:+0.3f}$\n$R^2 = {r2_score(y,y_hat):0.3f}$"
plt.gca().text(1.05, 0.95, text,transform=plt.gca().transAxes,
     fontsize=14, verticalalignment='top')
plt.savefig('CorrelationVOMR.png', dpi=200, bbox_inches='tight')
```



0.0.5 10 European Countries List

[176]:
```
tenE_list = ['France', 'Germany', 'Italy', 'Netherlands', 'Poland', 'Russia',
        'Spain', 'Switzerland', 'Turkey', 'United Kingdom']
```

[177]:
```
tenE_list
```

[177]:
```
['France',
 'Germany',
 'Italy',
 'Netherlands',
 'Poland',
 'Russia',
 'Spain',
 'Switzerland',
 'Turkey',
 'United Kingdom']
```

[184]:
```
mergeEuro = mergeVOMR[mergeVOMR['Entity'].isin(tenE_list)]
```

[185]:
```
xx = mergeEuro['Deaths']
yy = mergeEuro['Vegetable Oil']
```

[231]:
```
dfi.export(mergeEuro.loc[:, 'Vegetable Oil' :'Deaths'].describe(),␣
 ↪'mergeEuroStats.png')
```

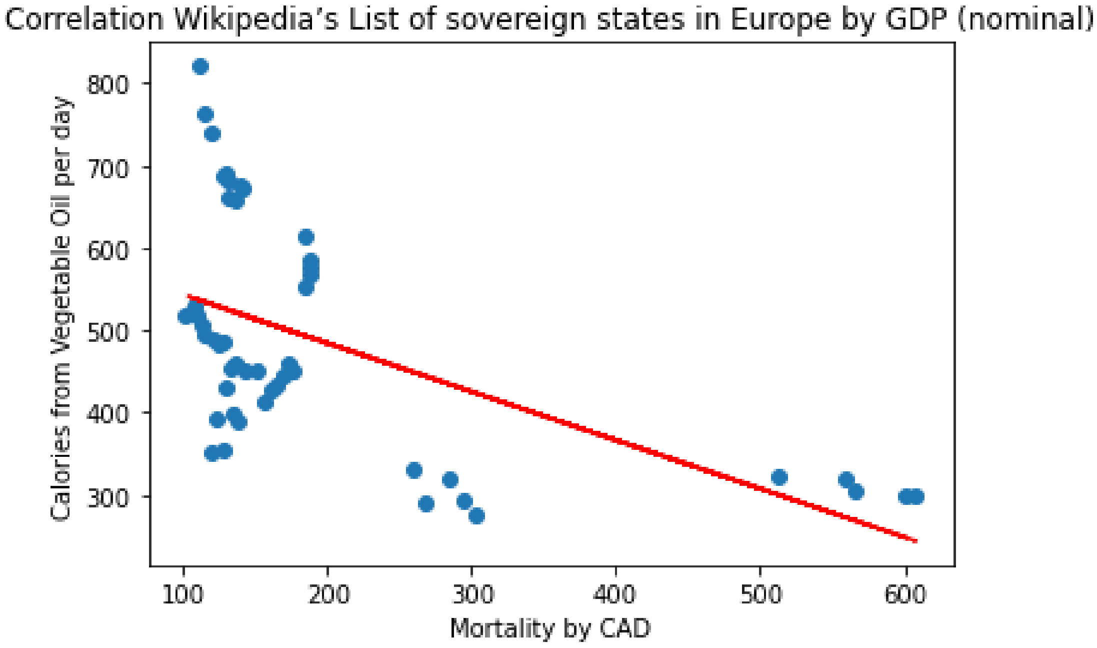
[233]:
```
mergeEuro.loc[:, 'Vegetable Oil' :'Deaths'].describe()
```

[233]:
```
       Vegetable Oil      Deaths
count      50.000000   50.000000
```

```
mean     486.493151  196.486640
std      140.447929  135.247507
min      276.164384  101.182683
25%      390.205479  125.727912
50%      461.095890  138.600611
75%      574.520548  186.434305
max      821.095890  608.128850
```

[194]:

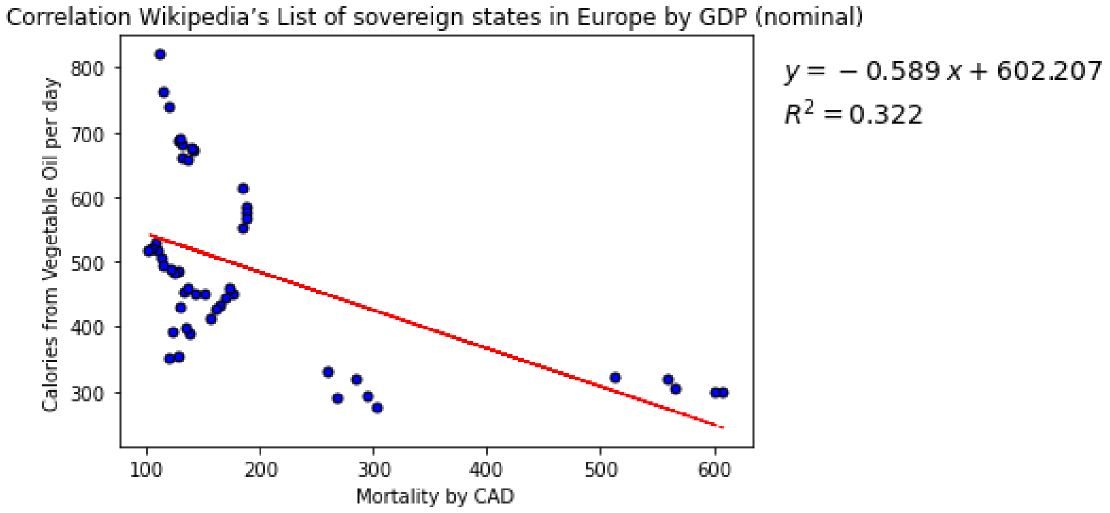
```
plt.title('Correlation Wikipedia’s List of sovereign states in Europe by GDP␣
 ↪(nominal)')
plt.ylabel('Calories from Vegetable Oil per day')
plt.xlabel('Mortality by CAD')
plt.scatter(xx, yy)
z = np.polyfit(xx, yy, 1)
p = np.poly1d(z)
plt.plot(xx,p(xx),"r--")
# plt.savefig('CorrelationEURO.png', dpi=200, bbox_inches='tight')
```



[223]:

```
plt.title('Correlation Wikipedia’s List of sovereign states in Europe by GDP␣
 ↪(nominal)')
plt.ylabel('Calories from Vegetable Oil per day')
plt.xlabel('Mortality by CAD')
plt.plot(xx,yy,"b.", ms=10, mec="k")
z = np.polyfit(xx, yy, 1)
y_hat = np.poly1d(z)(xx)
```

```
plt.plot(xx, y_hat, "r--", lw=1)
text = f"$y={z[0]:0.3f}\;x{z[1]:+0.3f}$\n$R^2 = {r2_score(yy,y_hat):0.3f}$"
plt.gca().text(1.05, 0.95, text,transform=plt.gca().transAxes,
     fontsize=14, verticalalignment='top')
plt.savefig('CorrelationEuro.png', dpi=200, bbox_inches='tight')
```



[ ]: