

```
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats
```

```
#reading all the files
```

```
disease_df = pd.read_csv('SAHD_county_data.csv')
county_df = pd.read_csv('co-est2019-alldata.csv', encoding='ISO-8859-1')
diet_df = pd.read_csv('diet_by_county.csv')
```

```
<ipython-input-2-acaf4d3b4708>:3: DtypeWarning: Columns (11,12) have mixed types. Specify dtype option on import or set low_
disease_df = pd.read_csv('SAHD_county_data.csv')
```

```
#modifying the SAHD dataset to only include data from 2013 and ages 35-64 years
#and removing all unnecessary cols and only keeping heart disease info not stroke info
```

```
disease_df = disease_df[disease_df['Year'] == '2013']
disease_df = disease_df[disease_df['Stratification1'] == 'Ages 35-64 years']
cols_keep = ['Year', 'LocationID', 'LocationAbbr', 'LocationDesc', 'Topic', 'Data_Value', 'Stratification1']
disease_df = disease_df[cols_keep]
heart_df = disease_df[disease_df['Topic'] != "Stroke"]
```

```
#becuase diet_df.csv has the fewest rows (2605 in comparison to 3136 for heart_df.csv and 3193
#for county_df.csv) left joins were used on diet_df.csv
```

```
#merging the SNAP data with the census data
merged = diet_df.merge(county_df, how='left')
```

```
#District of Columbia is the only county that doesn't end with the word county or
#parish so added it in so we could chop it off in the next line
merged.loc[merged['CTYNAME'] == 'District of Columbia', ['CTYNAME']] = 'District of Columbia County'
```

```
#removes the word county from all the counties
merged['CTYNAME'] = merged['CTYNAME'].str.rsplit(' ', n=1).str[0]
```

```
#created a col with county and state name to avoid the problem of duplicate county names
merged['County State'] = merged['CTYNAME'] + merged['STATE']
heart_df['County State'] = heart_df['LocationDesc'] + heart_df['LocationAbbr']
```

```
#merged census, SNAP, and SAHD data together
merged_final = merged.merge(heart_df, how='left')
```

```
#multiplying the percentage of people with the CAD (logged as instances per 100000)
#by 100000 and then dividing by the population to get the total number of cases per county
merged_final['Total Heart Disease'] = merged_final['Data_Value']*100000/merged_final['POPESTIMATE2013']
```

```
#converted all diet variables to total numbers for the entire county
merged_final['Total_FV'] = merged_final['Fresh_FV'] * merged_final['POPESTIMATE2013']
merged_final['Total_soda'] = merged_final['soda'] * merged_final['POPESTIMATE2013']
merged_final['Total_fast_food'] = merged_final['fastfood'] * merged_final['POPESTIMATE2013']
```

```
#added together all variables for every county to get a state total
state_df = merged_final.groupby('STATE')[['POPESTIMATE2013', 'Total Heart Disease', 'Total_FV', 'Total_soda', 'Total_fast_food']]
```

```
<ipython-input-4-7dff0221c10d>:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view)

```
heart_df['County State'] = heart_df['LocationDesc'] + heart_df['LocationAbbr']
```

```
#calculate as percent of state populations
state_df['Heart Disease'] = state_df['Total Heart Disease']/state_df['POPESTIMATE2013']*100000
state_df['FV'] = state_df['Total_FV']/state_df['POPESTIMATE2013']
state_df['Soda'] = state_df['Total_soda']/state_df['POPESTIMATE2013']
state_df['Fast Food'] = state_df['Total_fast_food']/state_df['POPESTIMATE2013']
```

```
#remove puerto rico from the dataset
state_df = state_df.dropna()
```

```
state_df = state_df.sort_values(by=['Heart Disease'])
```

```
#5 states with the least heart disease (higher as you go down)
```

```
#Heart Disease is numbers per 100,000
```

```
state_df.head(10)
```

	POPESTIMATE2013	Total Heart Disease	Total_FV	Total_soda	Total_fast_food	Heart Disease	FV	Soda	Fast Food
STATE									
DC	650581.0	12.342814	4.946054e+05	10907.514833	1.683714e+05	1.897199	0.760252	0.016766	0.258802
CT	3594841.0	135.984915	2.640964e+06	55369.343528	1.178653e+06	3.782780	0.734654	0.015402	0.327873
NJ	8310947.0	504.733907	5.480146e+06	139311.470381	2.448086e+06	6.073121	0.659389	0.016762	0.294562
DE	923576.0	78.906774	5.647180e+05	23353.781608	3.264666e+05	8.543615	0.611447	0.025286	0.353481
MA	6713315.0	613.945195	5.223661e+06	97495.082394	2.285703e+06	9.145187	0.778105	0.014523	0.340473
HI	1408154.0	135.659586	8.982469e+05	36038.945108	4.923157e+05	9.633860	0.637890	0.025593	0.349618
CA	38234059.0	3737.441739	2.420173e+07	821275.290119	1.204430e+07	9.775163	0.632989	0.021480	0.315015
AZ	6603326.0	673.737679	4.132560e+06	175566.220339	2.255255e+06	10.203005	0.625830	0.026588	0.341533
RI	1055081.0	209.488311	7.676778e+05	16361.653045	3.920067e+05	19.855187	0.727601	0.015507	0.371542

```
state_df = state_df.sort_values(by=['Heart Disease'], ascending=False)
```

```
#5 states with the most heart disease (higher as you go down)
```

```
state_df.head(10)
```

```
#note for ryka = get rid of total cols bc they are dependent on populations
```

	POPESTIMATE2013	Total Heart Disease	Total_FV	Total_soda	Total_fast_food	Heart Disease	FV	Soda	Fast Food
STATE									
ND	585189.0	21635.555777	3.375371e+05	20869.960927	2.146408e+05	3697.191126	0.576800	0.035664	0.366789
SD	694738.0	23642.611806	4.164872e+05	25273.337118	2.793522e+05	3403.097543	0.599488	0.036378	0.402097
NE	1716517.0	34833.171180	9.947860e+05	58892.420258	7.245255e+05	2029.293691	0.579538	0.034309	0.422091
KS	2761598.0	46080.614008	1.504584e+06	103422.916039	1.222059e+06	1668.621356	0.544824	0.037450	0.442519
KY	4341171.0	66957.195557	2.041231e+06	201098.903908	2.438527e+06	1542.376367	0.470203	0.046324	0.561721
MS	2626707.0	38435.130265	1.154731e+06	98618.920976	1.308978e+06	1463.243912	0.439612	0.037545	0.498334
IA	3021772.0	43886.824728	1.735781e+06	117618.186359	1.188575e+06	1452.353941	0.574425	0.038924	0.393337
AR	2861377.0	40409.546038	1.319663e+06	115995.542659	1.460821e+06	1412.241240	0.461199	0.040538	0.510531
MT	838634.0	10610.202797	5.863473e+05	21784.118103	2.570130e+05	1265.176799	0.699169	0.025976	0.306466

```
df = state_df.sort_values(by=['Soda'])
```

```
df.tail()
```

	POPESTIMATE2013	Total Heart Disease	Total_FV	Total_soda	Total_fast_food	Heart Disease	FV	Soda	Fast Food
STATE									
AL	4796791.0	17970.869444	2.191078e+06	189222.668894	2.436427e+06	374.643578	0.456780	0.039448	0.507929
AR	2861377.0	40409.546038	1.319663e+06	115995.542659	1.460821e+06	1412.241240	0.461199	0.040538	0.510531
OK	3695110.0	27255.028804	1.772054e+06	150019.203034	1.990309e+06	737.597225	0.479567	0.040599	0.538633
WV	1732647.0	15093.340140	8.531874e+05	76410.748437	8.254989e+05	871.114551	0.492418	0.044101	0.476438
KY	4341171.0	66957.195557	2.041231e+06	201098.903908	2.438527e+06	1542.376367	0.470203	0.046324	0.561721

```
!pip install geopandas
import geopandas as gpd
```

```

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting geopandas
  Downloading geopandas-0.13.0-py3-none-any.whl (1.1 MB)
    1.1/1.1 MB 19.5 MB/s eta 0:00:00
Collecting fiona>=1.8.19 (from geopandas)
  Downloading Fiona-1.9.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (16.0 MB)
    16.0/16.0 MB 56.8 MB/s eta 0:00:00
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from geopandas) (23.1)
Requirement already satisfied: pandas>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from geopandas) (1.5.3)
Collecting pyproj>=3.0.1 (from geopandas)
  Downloading pyproj-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (7.7 MB)
    7.7/7.7 MB 72.4 MB/s eta 0:00:00
Requirement already satisfied: shapely>=1.7.1 in /usr/local/lib/python3.10/dist-packages (from geopandas) (2.0.1)
Requirement already satisfied: attrs>=19.2.0 in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (23.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (2022.12.7)
Requirement already satisfied: click~8.0 in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (8.1.3)
Collecting click-plugins>=1.0 (from fiona>=1.8.19->geopandas)
  Downloading click_plugins-1.1.1-py2.py3-none-any.whl (7.5 kB)
Collecting cligj>=0.5 (from fiona>=1.8.19->geopandas)
  Downloading cligj-0.7.2-py3-none-any.whl (7.1 kB)
Collecting munch>=2.3.2 (from fiona>=1.8.19->geopandas)
  Downloading munch-2.5.0-py2.py3-none-any.whl (10 kB)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (2.8.1)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (2022.1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (1.24.0)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from munch>=2.3.2->fiona>=1.8.19->geopandas) (1.16.0)
Installing collected packages: pyproj, munch, cligj, click-plugins, fiona, geopandas
Successfully installed click-plugins-1.1.1 cligj-0.7.2 fiona-1.9.3 geopandas-0.13.0 munch-2.5.0 pyproj-3.5.0

```

```

country = gpd.read_file('cb_2018_us_state_20m.shp')
states_merged = country.merge(state_df, left_on='STUSPS', right_index=True, how='left')

```

```

fig, [[ax1, ax2], [ax3, ax4]] = plt.subplots(2, 2, figsize=(12, 7))

states_merged.plot(color='#EEEEEE', ax=ax1)
states_merged.plot(column='Heart Disease', legend=True,
                    ax=ax1, vmin=0, vmax=3697.1911258071505)
ax1.set(xlim=(-190, -60))
ax1.set_title('Heart Disease')

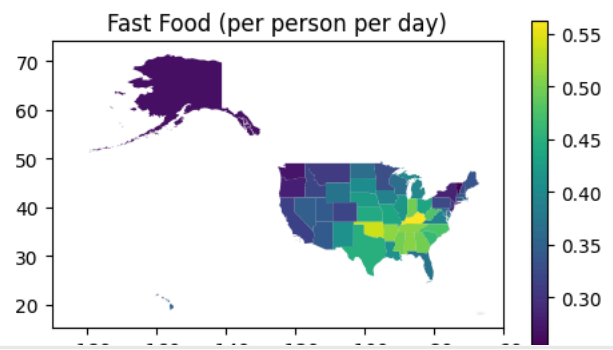
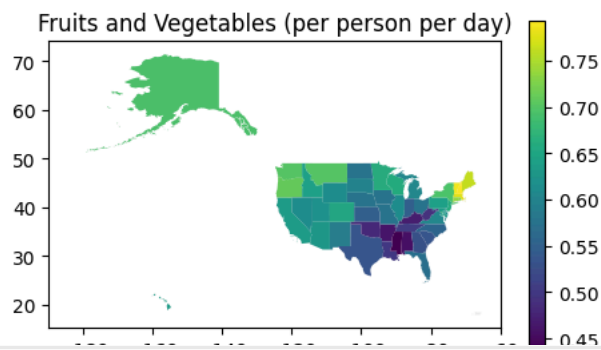
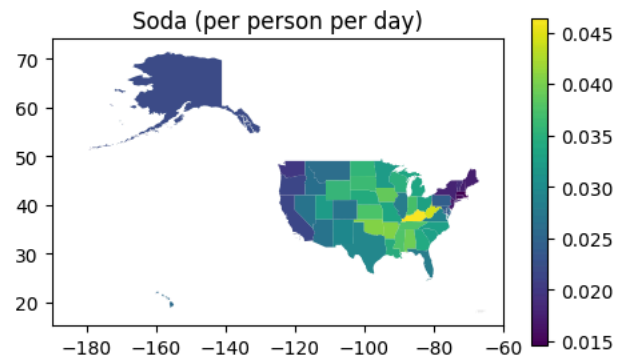
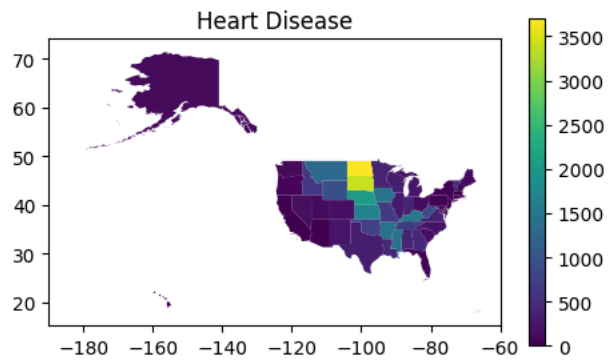
states_merged.plot(color='#EEEEEE', ax=ax3)
states_merged.plot(column='FV', legend=True,
                    ax=ax3, vmin=0.4396115274255892, vmax=0.7924542675892629)
ax2.set(xlim=(-190, -60))
ax3.set_title('Fruits and Vegetables (per person per day)')

states_merged.plot(color='#EEEEEE', ax=ax2)
states_merged.plot(column='Soda', legend=True,
                    ax=ax2, vmin=0.014522643789798782, vmax=0.04632365412655134)
ax3.set(xlim=(-190, -60))
ax2.set_title('Soda (per person per day)')

states_merged.plot(color='#EEEEEE', ax=ax4)
states_merged.plot(column='Fast Food', legend=True,
                    ax=ax4, vmin=0.2517673537811804, vmax=0.5617211256032667)
ax4.set(xlim=(-190, -60))
ax4.set_title('Fast Food (per person per day)')

```


Text(0.5, 1.0, 'Fast Food (per person per day)')

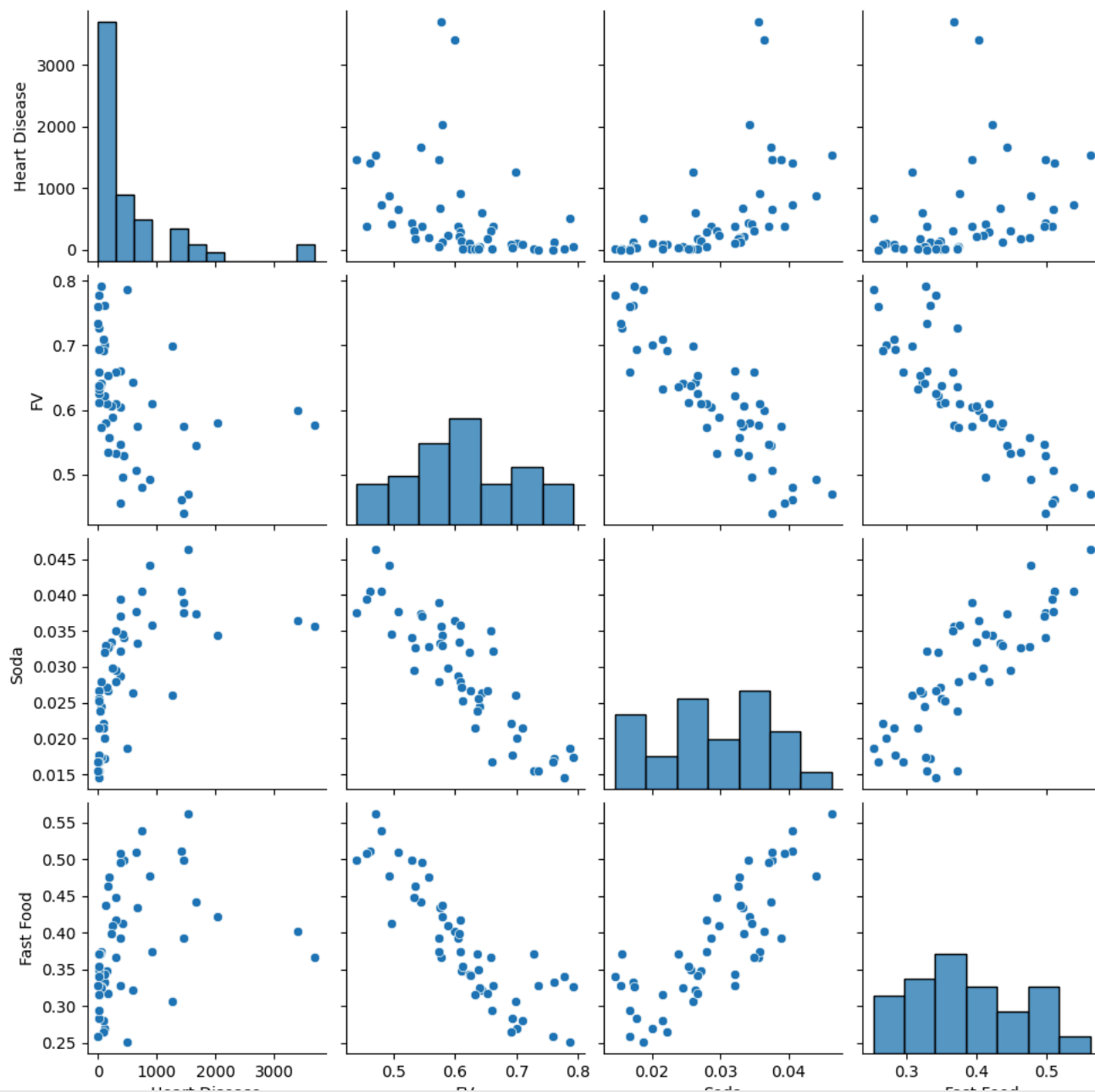


```
import seaborn as sns
```

```
#concatenate data  
final_df = state_df[['Heart Disease', 'FV', 'Soda', 'Fast Food']]
```

```
#pairplot on features and Heart Disease  
sns.pairplot(final_df)
```

 <seaborn.axisgrid.PairGrid at 0x7f1cc8994a90>



```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeRegressor, plot_tree
import numpy as np
```

```
features = state_df[['FV', 'Soda', 'Fast Food']]
labels = state_df['Heart Disease']
```

```
#used standard scaling to scale all features to have a mean of 0 and a standard deviation of 1
scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)
```

```
#transformed features
```