DATA1002 [PROJECT STAGE 1]

Crime Factors in the US by State
From years 2013-2019

PREPARED BY

SID: 530023821
SID: 530508591
SID: 510644297
SID: 530511063

[10/09/23]

Introduction

Historically, crime rates in the United States have experienced many fluctuations over the years, varying by the type of crime, geographic location and other factors. However, these crime rates, with the high level of public attention and scrutiny they receive, still remain a subject of significant concern given their large impact on communities and society as a whole. The statistics carry great importance in managing public safety and security by drawing awareness towards the prevalence of criminal activities in a given area.

This report aims to draw observations regarding the correlation between crime rates and socioeconomic factors in the United States, such as income, employment rates, and education attainment, whilst factoring in the distinction between correlation and causation. Effective crime prevention and reduction strategies can be employed by addressing underlying such socio economic issues. To explore the relationship between the multiple contributing factors, we performed an analysis on four respective datasets, each containing a wide variety of data values which we filter according to their relevance to our project aim.

The datasets used in this report include:

- Dataset1 - Unemployment level data obtained from U.S.DEPARTMENT OF AGRICULTURE (USDA)
- Dataset2 - Education attainment level data obtained from the“National Center for Education Statistics”. The Institute of Education Sciences (IES) is the statistics, research, and evaluation arm of the U.S. Department of Education.
- Dataset3 - Income data obtained from ‘The Bureau of Economic Analysis’ (BEA)
- Dataset4 - Crime data obtained from FBI: UCR.

Victims: Individuals and families who have experienced crimes, or are subject to crime such as burglary, violent assault, or vandalism, are likely to be directly affected physically, emotionally, and suffer economically.

Educational institutions: many students and teachers can have their safety at risk, leading to major concerns about security measures and the educational environment.

Business Owners: High crime rates can lead to increased security costs and potential losses due to thrift or property damage.

Local Economy: Crime can deter businesses from investing in areas with high crime rates, potentially impacting local economic development.

Police: A high crime rate places increased demands on police departments, requiring more resources and personnel for crime prevention, investigation, and enforcement.

Courts: Courts may become overwhelmed with cases, leading to backlogs, delays, and challenges in providing timely justice.

Corrections: Prisons and correctional facilities may become overcrowded due to high crime rates, impacting inmate management and rehabilitation efforts.

DATASET 1 - Unemployment rate

Explanation + Metadata + Limitations/Strengths

Our first data set was retrieved from the official U.S.DEPARTMENT OF AGRICULTURE (USDA), Economic Research Service. The original source of data comes from the U.S. Department of Labor, Bureau of Labor Statistics and U.S. Department of Commerce, Bureau of the Census. The Economic Research Service collected the data and prepared the final table (3277 rows x 100 columns).

Data presented on the USDA Website is considered public domain information that can be freely distributed or copied with the appropriate credits requested. As we have correctly cited the source in the appendix at the end of the report, our use is within the scope set by the copyright owners.

The aim of our project is to investigate the factors leading to crime rate differences across different regions in the US, therefore, unemployment data grouped by states between 2013-2019 is desired. Before choosing this dataset, we also compared relevant sources directly from the Bureau of Labor Statistics, the Bureau of Economic Analysis and the National Conference of State Legislatures (NCSL). Among all sources, this dataset provides the most relevant information and has the best quality by containing the least amount of missing value and the clearest explanations of each attribute. Other strengths of the data include its strong reliability (given that USDA is a federal executive department) and clarity (Classification is carefully stated and there is a separate worksheet especially prepared for variable descriptions).

However, only a limited number of variables in this dataset can be used as we only focus on the state-level unemployment within the 7-year range (2013-2019) while the dataset offers information beginning from 2000 and is detailly grouped into county level. The breadth of the data has decreased after dropping more than half of the columns and rows.

Below is the name and description for all variables mentioned in the cleaned dataset.

Variable	**Description**
Postal Abbr.	State abbreviation
State	State or county name
Year	The specific calendar year in which the associated statistics or data were recorded
Civilian_labor_force	Civilian labor force annual average
Employed	Number employed annual average
Unemployed	Number unemployed annual average
Unemployment_rate	Unemployment rate

Data quality check + Cleaning process

The data cleaning/checking we did using pandas included:

- Use ***list(df.columns)*** to return column names as a list and ***df.types*** to check if the data types are all correct. The result shows that all categorical data is presented as 'object' and all numerical data is stored as 'float64'or 'int64', which is appropriate.
- Use ***df.drop_duplicates()*** to delete repetitive rows. The exact same information on the District of Columbia appeared twice in the dataset and was removed.
- Use ***df.isnull().mean()*** to find the missing values distribution, turns out that there's no missing value in the dataset.

```
#Check if the data types are correct
print("Column datatypes: ")
print(df1.dtypes)
```

```
Column datatypes:

Postal Abbr.              object
State                     object
Year                       int64
Civilian_labor_force     float64
Employed                 float64
Unemployed               float64
Unemployment_rate        float64
dtype: object
```

```
#Check if there are missing values
print("Missing values distribution: ")
print(df.isnull().mean())
print("")
```

```
Missing values distribution:
State                     0.0
Area_Name                 0.0
Civilian_labor_force_2013 0.0
Employed_2013             0.0
Unemployed_2013           0.0
Unemployment_rate_2013    0.0
Civilian_labor_force_2014 0.0
Employed_2014             0.0
Unemployed_2014           0.0
Unemployment_rate_2014    0.0
Civilian_labor_force_2015 0.0
Employed_2015             0.0
```

```
#Check if there are duplicate rows and drop them if there are any
df=df.drop_duplicates()
df
```

The filtering and reformatting we did included:

- Use ***df.drop(columns= '')*** to pull out the data within the year range and drop the rest as we are only interested in the statistics from year 2013 to 2019.
- Use ***df=df['Area_Name'].isin(state_full)]***, unnecessary rows describing the county are dropped because we aim to investigate state-level unemployment. Note that 'Puerto Rico' is dropped because it is not a state but a US territory. Meanwhile, D.C. is kept due to its political and cultural importance.
- Convert the dataset from wide to long format so that the dataset is easier to read and integrate. We achieve this by using Pandas ***stack() and unstack(), rsplit(), and pivot_table*** to reshape the DataFrame.
- We sort the value based on state and year using ***df.sort_values(by=['State','Year'])***.

```
df=df.drop(columns=['FIPS_Code','Rural_Urban_Continuum_Code_2013','Urban_Influence_Code_2013','Metro_2013'
 'Employed_2000','Unemployed_2000','Unemployment_rate_2000','Civilian_labor_force_2001','Employed_2001','U
```

```
state_full=['Alabama','Alaska','American Samoa','Arizona','Arkansas','California','Colorado','
df = df[df['Area_Name'].isin(state_full)]
```

```
#Rename relevent columns
df1=df1.rename(columns={'level_2': 'Type',0: 'Rate'} )
```

```
#Seperate the year from the 'Type' column using the rsplit function targeting at the last underscore
#Create a new column called 'Year' to store the data
df1[["Type","Year"]]=df1["Type"].str.rsplit("_",n=1,expand=True)
df1
```

```
#Move the columns
df1 = df1[['Postal Abbr.','State','Year','Type','Rate']]
df1['Year']=df1['Year'].astype('int')
```

```
#Create a pivot table using 'Postal Abbr.','State','Year' as index
df1=df1.pivot_table(index=['Postal Abbr.','State','Year'], columns='Type', values='Rate')
df1
```

```
#Pivot table to normal table, sorted by year.
df1=df1.sort_values(by=['State','Year'])
df1 = df1.reset_index()
df1
```

Analysis using Python

The cleaned dataset has 357 rows and 7 columns, all together with 2499 value, containing labor statistics for all 50 states plus D.C (a district, not state) in the US between 2013-and 2019.

```
# computing number of rows
rows = len(df.axes[0])
# computing number of columns
cols = len(df.axes[1])
# computing the size of the dataframe
values = df.size
print("Number of Rows: ", rows)
print("Number of Columns: ", cols)
print("Number of values: ", values)
```

```
Number of Rows:  357
Number of Columns:  7
Number of values:  2499
```

```
list(df.columns)
```

```
['Postal Abbr.',
 'State',
 'Year',
 'Civilian_labor_force'
 'Employed',
 'Unemployed',
 'Unemployment_rate']
```

```
state=df['State'].unique()
print(state)
print(len(state))
```

```
['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']
51
```

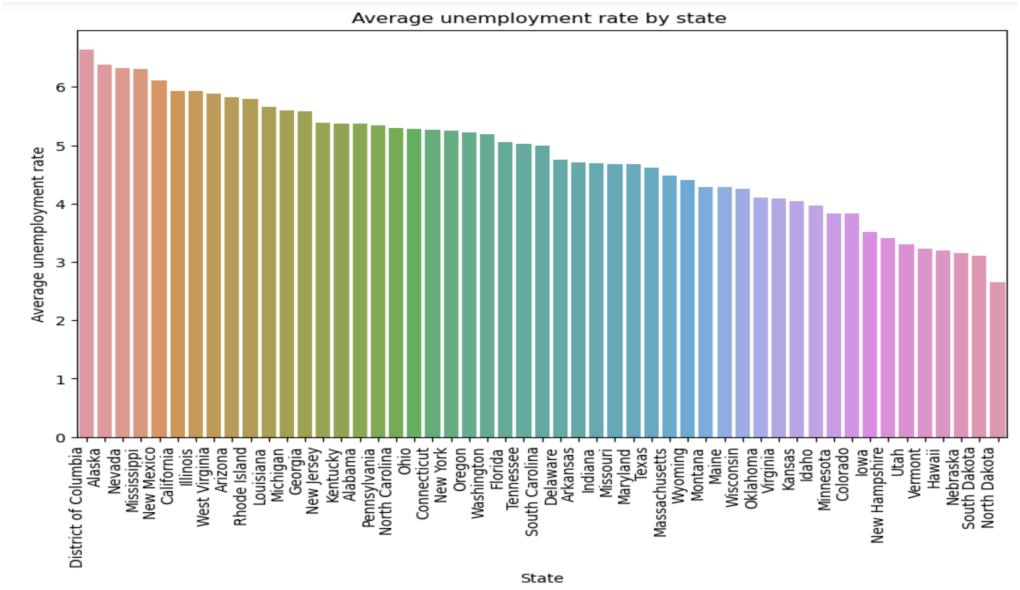
Group the ***unemployment rate*** data by State using ***groupby*** function and ***sort by 'mean'***

- The states/Districts with the highest average unemployment rates in 2013-2019 are the **District of Columbia(6.64), Alaska(6.39) and Nevada(6.33).**

```
#Which states/District have the highest average unemployment rate during 2013-2019?
df2=df.groupby(['State']).Unemployment_rate.agg(['mean'])
df2.sort_values(by=['mean'],ascending=False,inplace=False)


print("States/Districts with the highest unemployment rate")
print("Sorted by average rate during 2013-2019")

df2.head()
```

State	mean
District of Columbia	6.642857
Alaska	6.385714
Nevada	6.328571
Mississippi	6.300000
New Mexico	6.114286

- A basic visualisation of ***average unemployment by state*** using Matplotlib and Seaborn is performed



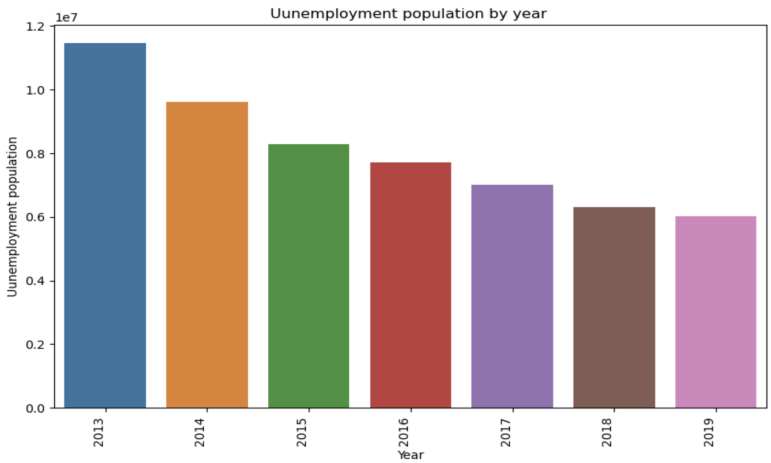
Group the ***Unemployment population*** data by Year using ***groupby*** function and ***Pivot_table***

- The unemployment population in the US(sum of all states) decreased from 2013-2019.

```
#For each year, What is the sum,maximum,minimum,avergae of the unemployment population in different states of the US?

print("Uunemployment population(across states)")
df.groupby(['Year']).Unemployed.agg(['sum','median','max','min'])
```

```
Uunemployment population(across states)
```

Year	sum	median	max	min
2013	11457241	149276.0	1677544	11590
2014	9617207	128967.0	1412174	10720
2015	8285731	106299.0	1176732	11468
2016	7725602	96696.0	1046610	10808
2017	7000411	84878.0	927302	10687
2018	6311939	83561.0	819595	8790
2019	6009990	72498.0	795311	7490



- California had the most Civilian labour force throughout the years

```
#Which state has the most Civilian labor force throughout the years?
df3=df.pivot_table(index='State',columns='Year',values='Civilian_labor_force')

df3.head(5)
```

State \ Year	2013	2014	2015	2016	2017	2018	2019
Alabama	2172102	2164715	2152295	2155729	2203458	2240109	2272935
Alaska	363544	364281	362425	362017	360589	355499	353161
Arizona	3064554	3118441	3184445	3253468	3240111	3325721	3430766
Arkansas	1331609	1327959	1339301	1343891	1348088	1343862	1354936
California	18565408	18676692	18824141	19012017	19185366	19289505	19413244

```
print("The state with the most Civilian labor force:")
df3.idxmax(axis='rows')
```

```
The state with the most Civilian labor force:

Year
2013    California
2014    California
2015    California
2016    California
2017    California
2018    California
2019    California
dtype: object
```

```
print("Number of the Civilian labor force in California:")
df3.max(axis='rows')
```

```
Number of the Civilian labor force in California:

Year
2013    18565408
2014    18676692
2015    18824141
2016    19012017
2017    19185366
2018    19289505
2019    19413244
dtype: int64
```

DATASET 2 - Levels of Education across the States (530511063)

Explanation + Limitations/Strengths + Metadata

- This dataset is obtained from the "National Center for Education Statistics". The Institute of Education Sciences (IES) is the statistics, research, and evaluation arm of the U.S. Department of Education. These statistics were originally sourced from the U.S. Department of Commerce, Census Bureau across from 2013 - 2019 American Community Survey (ACS).
- According to the "IES Policy Regarding Public Access to Research", any work that is funded by IES needs to be submitted to ERIC (Institute of Education Sciences) for review. Otherwise, it is considered public domain information and can be accessed with the necessary and appropriate credits requested.
- To complement the goal/objective of our project, we want to find appropriate data regarding levels of education across the States of America between the timeframe of 2013 to 2019, and through analysis see the relationship between crime rates and level of education. Through our research we found that IES provides the appropriate data, especially tidily grouped in years, through integrating multiple datasets (one from each year 2019 to 2019), we were able to obtain a larger datasets with the appropriate information from 2013 to 2019. All datasets from IES are obtained from the U.S. Department of Commerce, Census Bureau, in the form of surveys.

- Strengths:

- Data in the datasets are sourced from the U.S. Department of Commerce, an official government monitored website/source. Furthermore, the data quality of these datasets from 2013 - 2019 are of a high quality. Through filtering and cleaning, no missing, default and incorrect values were found, and the overall contour of the trend was relatively smooth, no unexpected outliers and unreasonable data were present.

- Limitations:

- The limitations of these datasets can mainly be recognised as how these data are obtained.
- As mentioned in the footnote of "Table 104.88: Number of persons age 25 and over and rates of high school completion and bachelor's degree attainment among persons in this age group, by sex and state: 2019" , "data are based on sample surveys of the entire population residing within the U.S., including both non-institutionalised persons (for example: those living in households, college housing, or military housing located within the U.S.) and institutionalised persons (for example: those living in prisons, nursing facilities, or other healthcare facilities). "
- As a result, the consequences below follows:
    - There could potentially be individuals who do not respond or participate fully. This could result in the introduction of bias if certain groups are more likely to avoid participation, leading to under-representation or even over-representation of specific demographics (for example: people may be unwilling to complete the survey/questionnaire due to personal reasons, like privacy concerns).
    - Certain groups may provide inaccurate or incomplete information. This can be intentional and unintentional, as a result of various reasons, such as privacy concerns, social desirability bias, or lack of knowledge and understanding of the survey/questionnaire (for example: people may be unwilling to reveal the accurate measure of their level of education, and thus providing inaccurate responses to the survey/questionnaire).

- Furthermore, the dataset focuses on “persons aged 25 and over”. Thus, it does not take into consideration people below the age of 25. This can be seen as a significant limitation of the dataset as it does not include people 18 or 19 years of age who have just completed high school or those who completed a bachelor degree of some sort before the age of 25.

Dictionary (by appearance from left to right)

State: Name of State (or county)
Year: The specific calendar year in which the associated statistics or data were recorded
No. of persons 25 and over (total): Number of persons 25 years of age and over in total
(for example: 462 for Alaska in 2013 means that the population of Alaska over the age of 25 was 462000)
No. of persons 25 and over (male): Number of males 25 years of age and over
(for example: 239 for Alaska in 2013 means that the population of Alaska over the age of 25 was 239000)
No. of persons 25 and over (female): Number of females 25 years of age and over
(for example: 222 for Alaska in 2013 means that the population of Alaska over the age of 25 was 222000)
Percentage with high school completion or higher (total): percentage of population over 25 years of age that have completed high school or higher in total
(for example: 91.6 for Alaska in 2013 means that 91.6% of the 462000 has completed high school of higher)
Percentage with high school completion or higher (male): percentage of male over 25 years of age that have completed high school or higher
(for example: 91 for Alaska in 2013 means that 91% of the 239000 males has completed high school of higher)
Percentage with high school completion or higher (female): percentage of female over 25 years of age that have completed high school or higher
(for example: 92.3 for Alaska in 2013 means that 92.3% of the 222000 females has completed high school of higher)
Percentage with bachelor’s or higher degree (total): percentage of population over 25 years of age that have completed at least a bachelor’s degree or higher in total
(for example: 29.2 for Alaska in 2013 means that 29.2% of the 462000 has completed at least a bachelor’s degree or higher)
Percentage with bachelor’s or higher degree (male): percentage of male over 25 years of age that have completed at least a bachelor’s degree or higher
(for example: 25.5 for Alaska in 2013 means that 25.5% of the 239000 has completed at least a bachelor’s degree or higher)
Percentage with bachelor’s or higher degree (female): percentage of female over 25 years of age that have completed at least a bachelor’s degree or higher
(for example: 33.2 for Alaska in 2013 means that 33.2% of the 222000 has completed at least a bachelor’s degree or higher)

Data quality check + Cleaning process

For the appropriate dataset(s), the following checks were performed to ensure the quality of the dataset(s):

1. Checking and Remedy for Missing values

2. Checking and Remedy for Incorrect values

Checking and Remedy for Missing Values

```
def chec_remedy_missing(df):
    """
    Detects missing values and imputes where there are missing values.
    Arguments:
    df: the dataset we are referencing to
    Returns:
    NONE
    """
    missing = False
    for c in df.columns:
        if df[c].isnull().any():
            missing = True
            for r in df[c].isnull().index:
                print(f"There is a missing value located at row {r} in column {c}.")
                """
                ^Provide the location of which the missing values are located.
                """

            if df[c].dtype.name == "float64":
                df[c].fillna(float(df[c].mean()), inplace = True)
            elif df[c].dtype.name == "int64":
                print(f"A population is missing at row {r} in column {c}.")
            else:
                print(f"A state is missing at at row {r} in column {c}")
                """
                ^Filling in the missing values.
                """
    if not missing:
        print("There are no missing values in this dataset.")
        """
        ^No missing values are detected.
        """
    else:
        print("All missing values were remedied and filled.")
        """
        ^Remedy completed.
        """
```

Initially, the function takes in the argument ‘df’ which is the dataset we are trying to check.
For loop loops through each column and identifies any columns with any missing values using the isnull() method. Then, another for loop loops through to identify the exact location of the missing value (with the use of .index for each row within that column).
For remedy, if the missing value is a percentage (float in this case), the average percentage of that column will be imputed in using the fillna() method with .mean(). If the missing value is either an integer or a string, the function prints out the location of that missing value. (There is no point finding the average population of all states then imputing it as a value, because each state can be drastically different to one another in size. For the name of each state, it does not make sense to impute a random state in.)

Checking and Remedy for Incorrect Values:

Again, the functions take in ‘df’ as the file we are checking, along with ‘column_name’, a valid range of values and the data type of that column. For any percentage (float), the function will check whether that float (selected using the .loc() method) is between 0 and 100 and whether the data is in the form of a float (using .dtype attribute and whether that is the same as the argument data_type that is provided). For integers (total population in this case), a custom valid range (v_range) needs to be provided as an argument (the default valid range is for floats. I.e. percentages between 0 and 100). For the names of state the function will simply check if the values are of the same type. Similar to before, it does not make sense to impute a value for the population and the name of a statement, so the function will simply identify where the incorrect value is.

```
def chec_remedy_incorrect(df, column_name, v_range = list([0,100]), data_type = "float64"):
    """
    Detects incorrect values and imputes where there are default values.
    Arguments:
    df: the Pandas dataset we are referencing to
    column_name(string): the title of the column to check
    v_range(list): a list of valid values
    data_type(string/data type): the datatype of the data in the column.
    Returns:
    NONE
    """
    for j in range(len(df)):
        value = df.loc[j, column_name]
        if data_type == "float64":
            if not (v_range[0] <= value <= v_range[1]):
                df.loc[j, column_name] = float(df[column_name].mean())
                print(f"For the column named {column_name}, in row {j}, the value {value} is not in the valid range.")
            if value.dtype != data_type:
                df.loc[j, column_name] = float(df[column_name].mean())
                print(f"For the column named {column_name}, in row {j}, the value {value} is not the correct data type.")
        elif data_type == "int64":
            if not value in v_range:
                print(f"For the column named {column_name}, in row {j}, the value {value} is not in the valid range.")
            if value.dtype != data_type:
                print(f"For the column named {column_name}, in row {j}, the value {value} is not the correct data type.")
        else:
            if isinstance(value, data_type) == False:
                print(f"For the column named {column_name}, in row {j}, the value is not the correct data type.")

    print("Detections and remedies for incorrect values are complete.")
```

Note: As previously discussed and mentioned in the Strengths of the dataset, there were no missing or incorrect values to begin with, as the dataset was already top quality and in good condition and no cleaning was necessary.

Analysis using Python

Some basic information:

The function 'csv_info' presents basic information about the dataset. This includes the size, the name and the data type of each column, the number of non-null tiles and some other information. The output is shown below:

```python
def csv_info(df):
    """
    Prints and presents the fundamental information about the dataset.
    Arguments:
    df: the dataset we are referencing to
    Returns:
    NONE
    """
    r, c = df.shape
    print(f"In total, there are {r} rows and {c} columns in this dataset.")
    """
    ^Prints the number of rows and columns of the dataset.
    """

    print("The other basic information of this dataset is shown below")
    df.info()
    """
    ^Prints the other information about the data set.
    """
```

```
In total, there are 357 rows and 11 columns in this dataset.
The other basic information of this dataset is shown below
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 357 entries, 0 to 356
Data columns (total 11 columns):
 #   Column                                                Non-Null Count  Dtype
---  ------                                                --------------  -----
 0   State                                                 357 non-null    object
 1   Year                                                  357 non-null    int64
 2   No. of persons 25 and over (in thou)  Total           357 non-null    int64
 3   No. of persons 25 and over (in thou)  Male            357 non-null    int64
 4   No. of persons 25 and over (in thou)  Female          357 non-null    int64
 5   Percentage with high school completion or higher (%)  total   357 non-null    float64
 6   Percentage with high school completion or higher (%)  Male    357 non-null    float64
 7   Percentage with high school completion or higher (%)  Female  357 non-null    float64
 8   Percentage with bachelor's or higher degree (%)  total        357 non-null    float64
 9   Percentage with bachelor's or higher degree (%)  Male         357 non-null    float64
 10  Percentage with bachelor's or higher degree (%)  Female       357 non-null    float64
dtypes: float64(6), int64(4), object(1)
memory usage: 30.8+ KB
```

Other analysis:

The function to the right and the snippets of code below prints the highest and the lowest value of a selected column. By importing pandas, we can simply make the function return 2 values: the highest and the lowest in the selected column. It would not make sense to try to work out the highest and the lowest population, because that does not contribute to our needs to find patterns within the data (as each state has drastically different demographics). Instead, it makes the most sense for us to find the state with the highest and lowest percentage on whichever criteria, each year. Before merging the data from 2013 to 2019, we can use the initial datasets and run the columns we want through this function (Grouping the dataset with respect to year, we use 2019 and "Percentage with high school completion or higher: total"). Below is the output for the 2019 dataset, and to the right is a table of this action repeated for the other years.

Year	State with Highest percentage of persons completing h.s and higher	State with lowest percentage of persons completing h.s and higher
2013	Wyoming (93.6%)	California (81.7%)
2014	North Dakota (92.9%)	California (82.1%)
2015	Montana (93.4%)	California (82.2%)
2016	Wyoming (93.2%)	California (82.4%)
2017	New Hampshire (93.2%)	California (83.4%)
2018	Alaska (94%)	Cali, Texas (83.9%)
2019	Wyoming (94.5%)	California (84.1%)

```python
def find_highest_and_lowest(df, column_name):
    """
    Prints and presents the highest and lowest value in each column.
    Arguments:
    df: the dataset we are referencing to
    column_name: the column we are referencing to
    Returns:
    NONE
    """
    highest_value = df[column_name].max()
    lowest_value = df[column_name].min()

    return highest_value, lowest_value

target_column = "Percentage with high school completion or higher (%)  total"
highest_value, lowest_value = find_highest_and_lowest(df, target_column)

print("For all states in the year 2019:")
print(f"The highest value in column '{target_column}' is: {highest_value}")
print(f"The lowest value in column '{target_column}' is: {lowest_value}")
```

```
[user@sahara ~]$ python highlow.py
For all states in the year 2019:
The highest value in column 'Percentage with high school completion or higher (%)  total' is: 94.5
The lowest value in column 'Percentage with high school completion or higher (%)  total' is: 84.1
```

DATASET 3 - Income and economic activity across the states

SID: 530508591

Explanation + Limitations/Strengths + Metadata

The US regional economic summary dataset, obtained from 'The Bureau of Economic Analysis' (BEA) is a comprehensive collection of various economic activities, measured annually across each state in the US. As an authoritative and official US government agency, data sourced from BEA is considered highly reliable and credible. The original dataset, structured with 781 rows and 17 columns, provides a comprehensive coverage of economic data over a duration of 26 years (1998 to 2022), including nominal values and inflation-adjusted values, allowing for analysis of trends at various levels of granularity. Especially concerning our project question, all the data entries that fall within the years 2013 and 2019 are of high quality, where no missing or unreasonable values were detected and trends across the variables for each state were relatively similar.

The data was originally collected through a combination of surveys, data sources, and administrative records, gathered from federal agencies such as the U.S. Census Bureau and the Bureau of Labor Statistics (BLS). Intercensal population statistics were produced using the 'Census Bureau Das Gupta' method on the Census Bureau decennial counts for 2010 and 2020 to create consistent time series that are used to prepare per capita personal income statistics. The method was utilised to account for the unreleased official Census population data for particular years.

According to the official website of the BEA, any information retrieved directly from the website is in the public domain and may be used or reproduced without specific permission. With appropriate citations and references provided, the dataset is able to be used securely by the general public and use is kept within the scope of the copyright owners.

Although data obtained from The Census Bureau has undergone extensive quality checks to ensure data accuracy, it is still possible for underlying errors to occur due to self-reporting, political or social factors, or human errors in surveys. This limits the extent of the data's reliability and has to be taken into account when justifying the inferences made based on the values given. Moreover, by relying on estimation techniques to compile data (such as the 'Census Bureau Das Gupta' method), margins of error are introduced as data figures reported over time are subject to change and may not be consistent.

Summary of Metadata:

Variable Name	Description	Measurement Unit	Data Type	Minimum value	Maximum value
Real GDP	Inflation-adjusted measure of each state's gross product based on national prices for the goods and services in the state.	*Millions of chained 2012 dollars	Float	28681.5	2729225.8
Real PI	Personal income divided by the RPPs and national PCE price index	*Millions of constant 2012 dollars	Float	28647.1	2108670.2

Variable Name	Description	Measurement Unit	Data Type	Minimum value	Maximum value
Real PCE	Personal consumption expenditures divided by the RPPs and national PCE price index.	Millions of constant 2012 dollars	Float	22246.0	1571307.1
Gross domestic product (GDP)	Measure of the market value of all final goods and services produced in a state in a specific time period.	*Millions of current dollars	Float	29289.6	3042694.1
Personal income (PI)	Income that a person receives in return for the provision of labour, land, and capital used in current production as well as other income.	Millions of current dollars	Float	28964.3	2567425.6
Disposable PI	The income available to a person for spending or saving. Equal to personal income without personal current taxes.	Millions of current dollars	Float	26087.0	2198931.7
Personal consumption expenditures (PCE)	A measure of spending on goods and services purchased by, and on behalf of, households based on households' state of residence.	Millions of current dollars	Float	21807.3	1917201.3
Real per capita PI	Real personal income divided by midyear population	Constant 2012 dollars	Integer	36310	70457
Real per capita PCE	Total real personal consumption expenditures divided by total midyear population.	Constant 2012 dollars	Integer	29407	59334
Per capita PI	Personal income of a given area divided by the resident population of the area.	Dollars	Integer	34259	84671
Per capita disposable PI	Disposable personal income of a given area divided by the resident population of the area.	Dollars	Integer	31523	71186
Per capita PCE	Personal consumption expenditures of a given area divided by the resident population of the area.	Dollars	Integer	28020	71454
Regional price parities (RPPs)	Regional price levels determined by the average prices paid by consumers for goods and services consumed, expressed as a percentage of the overall national price level.	Index	Float	86.095	113.385
Implicit regional price deflator	Product of the region's RPP and the US PCE price index.	Index	Float	90.97	122.722
Total employment (number of jobs)	A count of jobs, both full-time and part-time. It includes wage and salary jobs, sole proprietorships, and individual general partners, but not unpaid family workers nor volunteers.	Number of jobs	Integer	398130	24227775

*Millions of chained 2012 dollars: Values expressed in units of one million dollars after an adjustment for inflation is completed against the base year 2012. Calculations are performed to take into consideration the changes in the composition of goods and services consumed over time.

*Millions of chained 2012 dollars: [see ‘Millions of chained 2012 dollars’] except adjustment is made using a fixed set of prices from the reference year 2012.

*Millions of current dollars: Values expressed in units of one million dollars without accounting for inflation or changes in the value of money in previous years.

Data Cleaning (completed in pandas):

Step one (data overview):

Check the data variables (columns) in the dataset and decide which ones are appropriate for usage by generating a list containing column headers.

```
#Create a list of all column headers
print('Original dataframe contains columns for:')
columns = list(df.columns)
print(columns)
```

Using `'df.info()'` a summary of the dataset, including data types of the columns with a count, and size of the dataframe, is returned. This is helpful in getting a quick overview on which sections of the dataset need changing.

Step two (filtering dataset):

To remove columns, `'df.drop([''])'` was used, where inside the list are the column names of the columns to remove.

```
#remove any irrelevant columns and rows that don't contain state or year information
df = df.drop(['GeoFIPS','Region','TableName','LineCode','IndustryClassification','Unit'], axis = 1)
```

Purely for convenience, instead of removing two sections of the columns in order to keep only the range of years from 2013 to 2019, `'df.loc[:['']]'` was used where the wanted column names could be entered in the list .

```
#keep only wanted columns including state, description, and columns with years 2013-2019
df = df.loc[:, ["GeoName","Description","2013","2014","2015","2016","2017","2018","2019"]]
```

Similarly to remove rows, the drop function is used, however, instead of providing a list of column names (given the rows do not have names) we enter in a range of the rows to remove through the default index values `('df.drop(np.r_[:])')`.

```
df = df.drop(np.r_[0:15,780:784])
```

As an added line of code, specific column headers were renamed for identification purposes, making future analysis with the dataset easier as variables are more clearly described.

```
#rename column headers to a more recognisable name
df = df.rename(columns = {'GeoName':'State', 'Description':'Variables'})
```

Step three (checking data values):

From the filtered dataset, we checked any potential missing values using `'df.isnull(),sum()'` that still remains and whether there were duplicates using `'df[df.duplicates()]'`

```
#check for any NULL values that need replacing or removing
print('Number of missing cells is: ')
missing_counts = df.isnull().sum()
print(missing_counts)
```

```
Number of missing cells is:
State        0
Variables    0
2013         0
2014         0
2015         0
2016         0
2017         0
2018         0
2019         0
```

```
#check for duplicates in data frame
duplicate = df[df.duplicated()]
print(len(duplicate))
```

```
0
```

Taking into account our later stage of integration, we decided to format every individual dataset in the same or very similar ways, where the US states would be the fixed index column, each state having separate rows for the years 2013-2019 in that order for the different variables. To achieve this for this particular dataset, a combination of the melt function `'pd.melt(df,)'`, pivot function `'df.pivot()',` and reset index method `'df.reset_index()'` was used.

```
# Melt the DataFrame to convert years (2013-2019) to a single column
melted_df = pd.melt(df, id_vars=['State', 'Variables'], var_name='Year', value_name='Value')

# Pivot the melted DataFrame to have 'Variables' as columns and 'Year' as a separate column
pivot_df = melted_df.pivot(index=['State','Year'], columns='Variables', values='Value')

# Reset the index to make 'Year' a separate column
pivot_df.reset_index(inplace=True)

# Rename the index column to 'Years'
pivot_df.rename(columns={'Year': 'Year'}, inplace=True)
```

Now that the dataset has been reformatted accordingly, we can change the data types shown from the initial line of code (all 'objects') to its actual type. We defined a dictionary containing all the column names as the keys and the wanted data type as their values. By using 'df.astype()' on the dictionary, all the columns will have an adjusted type in place.

Aggregation summaries (completed in pandas):

Performing an analysis on the trend of the Real Gross Domestic Product instead of the GDP provides a more accurate statistical measure of the US economy as adjustments have been made according to inflation rates at the time. By using `'df.groupby()'`, the entire dataset could be grouped into years, then applying `'.sum()'` on the grouped data enabled us to obtain the sum of the values across each state in each year.

```
# Group by year and calculate the sum of GDP for each year
gdp_sum_by_year = df.groupby('Year')['Real GDP'].sum().reset_index()

# Print or work with the resulting DataFrame
print(gdp_sum_by_year)
```

```
   Year     Real GDP
0  2013  16391324.6
1  2014  16773587.9
2  2015  17239274.1
3  2016  17530096.3
4  2017  17920705.3
5  2018  18434037.7
6  2019  18858550.9
```

From the printed result, we can conclude that the real GDP over the period of 6 years increased at a relatively steady rate. This serves as relevant information we can compare other economic data values against as many variables rely on the change of the country's economy over time.

Other data summaries include finding the maximum and minimum average values for the 'real per capita personal income' and 'real per capita personal consumption expenditures'. Computation of these values allow for more meaningful comparisons of economic indicators over different years as it enables us to assess whether there has been genuine growth or decline without inflation effects. Like with the GDP, the `'groupby()'` function was used and followed by `'.mean()'` to get the average. To print out the corresponding state with the max and min mean values, `'.loc[].values[]'` were used where the arguments contain the values and the index of the state column.

```
# Find the maximum and minimum values in the "Real per capita personal income" column
max_value = R_PI["Real per capita personal income"].max()
min_value = R_PI["Real per capita personal income"].min()

# Find the corresponding names for the maximum and minimum values
max_name = R_PI.loc[R_PI["Real per capita personal income"] == max_value, "State"].values[0]
min_name = R_PI.loc[R_PI["Real per capita personal income"] == min_value, "State"].values[0]

print(f"Maximum Value: {max_value} (State: {max_name})")
print(f"Minimum Value: {min_value} (State: {min_name})")
```

```
Maximum Value: 65914.42857142857 (State: District of Columbia)
Minimum Value: 39317.142857142855 (State: Mississippi)
```

```
print("States with the maximum and minimum average value for 'Real per capita PCE'")
print(f"Maximum Value: {max_value} (State: {max_name})")
print(f"Minimum Value: {min_value} (State: {min_name})")
```

```
States with the maximum and minimum average value for 'Real per capita PCE'
Maximum Value: 55249.0 (State: District of Columbia)
Minimum Value: 32171.85714285714 (State: Utah)
```

DATASET 4 —- State Crime - https://corgis-edu.github.io/corgis/csv/state_crime/

- This data set was retrieved from the CORGIS Dataset Project by Ryan Whitcomb, Joung Min Choi, and Bo Guuan on GitHub.
- The original source of data comes from the website of FBI: UCR. https://ucr.fbi.gov/crime-in-the-u.s/2019/crime-in-the-u.s.-2019/downloads/download-printable-files.
- The provided dataset contains information regarding crime rates and totals of states across the United States from 1960 to 2019. In this dataset, there are 21 variables. The crime types are divided into two main categories: property and violent crime. Property crimes refer to burglary, larceny, and motor-related crime while violent crimes refer to assault, murder, rape, and robbery.
- Limitations: The data is used within the 7-year range (2013-2019) while the dataset offers information beginning from 1960 to 2019. Rows containing 'United States' values have been removed because it was unclear which specific state 'United States' referred to. This will affect the accuracy of the analysis.

Data quality + cleaning process

```
df = pd.read_csv("/Users/mac/Downloads/state_crime.csv")
print(df.head())
```

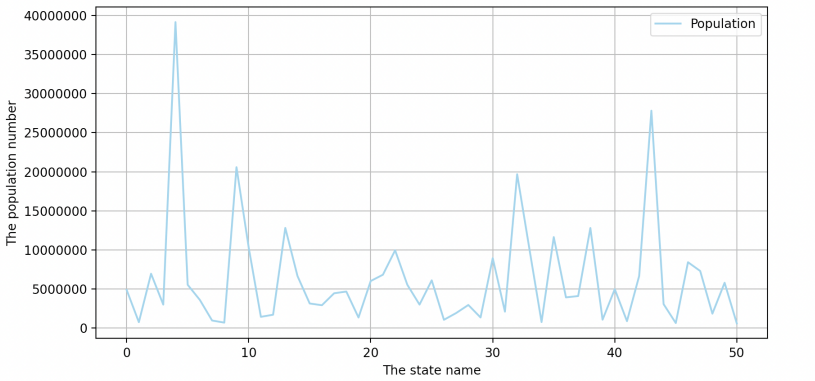
```
    State  Year  Data.Population  ...  Data.Totals.Violent.Murder  Data.Totals.Violent.Rape  Data.Totals.Violent.Robbery
0  Alabama  1960          3266740  ...                         406                       281                          898
1  Alabama  1961          3302000  ...                         427                       252                          630
2  Alabama  1962          3358000  ...                         316                       218                          754
3  Alabama  1963          3347000  ...                         340                       192                          828
4  Alabama  1964          3407000  ...                         316                       397                          992

[5 rows x 21 columns]
```

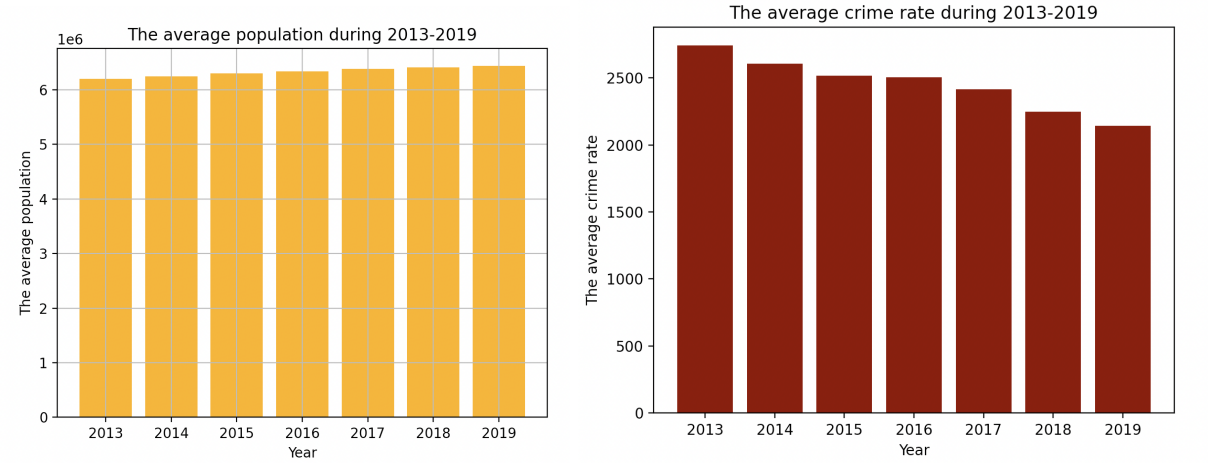
```
# Checking the data types
print(df.dtypes)

# Checking missing values. There are no missing values in the dataset.
miss = df.isnull().sum()
print(miss)

# Checking duplicates in data frame
duplicate = df[df.duplicated()]
print(len(duplicate))

# Creating a DataFrame and indexing values
index = pd.Index(range(len(df)))
y = df.set_index('Year')
df = pd.DataFrame(df)
```

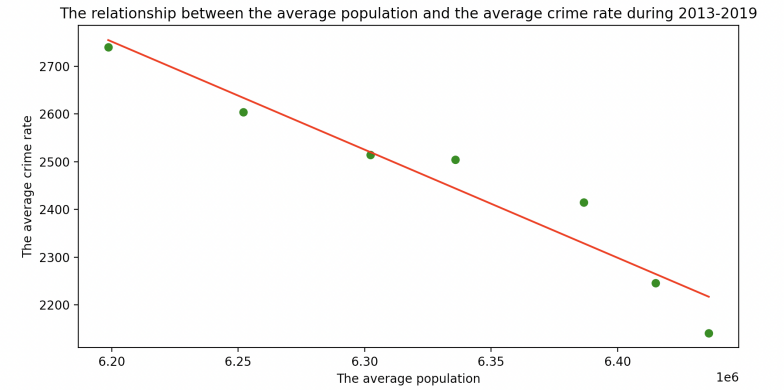
```
# Renaming column names
df.rename(columns = {'Data.Population':'Population', 'Data.Rates.Property.All':'Rates of all crimes',
                     'Data.Rates.Property.Burglary':'Rates of burglary', 'Data.Rates.Property.Larceny':'Rates of larceny',
                     'Data.Rates.Property.Motor':'Rates of motor', 'Data.Rates.Violent.All':'Rates of violent crimes',
                     'Data.Rates.Violent.Assault':'Rates of violent assault', 'Data.Rates.Violent.Murder':'Rates of violent murder',
                     'Data.Rates.Violent.Rape':'Rates of violent rape', 'Data.Rates.Violent.Robbery':'Rates of violent robbery',
                     'Data.Totals.Property.All':'Total crimes', 'Data.Totals.Property.Burglary':'Total burglary',
                     'Data.Totals.Property.Larceny':'Total larceny', 'Data.Totals.Property.Motor':'Total motor',
                     'Data.Totals.Violent.All':'Total violent crimes', 'Data.Totals.Violent.Assault':'Total violent assault',
                     'Data.Totals.Violent.Murder':'Total violent murder', 'Data.Totals.Violent.Rape':'Total violent rape',
                     'Data.Totals.Violent.Robbery':'Total violent robbery'}, inplace=True)
```

Loading...

```python
# Show all rows in VS Code terminal and find indexs of rows contained 'United States' value
pd.set_option('max_row', None)
search_year.head(None)
print(search_year)

# Delete the rows contains the United States value
search_year.drop(labels = range(2688, 2695), axis = 0, inplace = True)
print(search_year)
```







Summary:

Through these graphs, the average crime rate for all states in the US was decreasing and the population was increasing from 2013 to 2019. The correlation between the average crime rate and the population is negative.

Integrating datasets

Merging (screenshot of Python code)

```
df1['Year'] = df1['Year'].astype(object)
df2['Year'] = df2['Year'].astype(object)
df3['Year'] = df3['Year'].astype(object)
df4['Year'] = df4['Year'].astype(object)

df_merged = pd.merge(pd.merge(df1, df2, on=["State", "Year"]), df3, on=["State", "Year"])
df_merged = df_merged.merge(df4, on=["State", "Year"])
df_merged
```

```
#Remove the columns in df4 containing the standard deviations

columns_to_drop = [col for col in df_merged.columns if col.startswith('Unnamed:') and int(col.split(':')[1]) % 2 == 1]
df_merged = df_merged.drop(columns=columns_to_drop)
df_merged
```

Metadata

Dataset 1: Our first data set was retrieved from the official U.S.DEPARTMENT OF AGRICULTURE (USDA), Economic Research Service. The original source of data comes from the U.S. Department of Labor, Bureau of Labor Statistics and U.S. Department of Commerce, Bureau of the Census. The Economic Research Service collected the data and prepared the final table (3277 rows x 100 columns).

Dataset 2: This dataset is obtained from the “National Center for Education Statistics”. The Institute of Education Sciences (IES) is the statistics, research, and evaluation arm of the U.S. Department of Education. These statistics were originally sourced from the U.S. Department of Commerce, Census Bureau across from 2013 - 2019 American Community Survey (ACS). This dataset contains the total population of each state over the age of 25 years of age and corresponding percentage of high school completion and bachelor degrees completion.

Dataset 3: The data was originally collected through a combination of surveys, data sources, and administrative records, gathered from federal agencies such as the U.S. Census Bureau and the Bureau of Labor Statistics (BLS). Intercensal population statistics were produced using the ‘Census Bureau Das Gupta’ method on the Census Bureau decennial counts for 2010 and 2020 to create consistent time series that are used to prepare per capita personal income statistics. The method was utilised to account for the unreleased official Census population data for particular years.

Dataset 4: The provided dataset contains information regarding crime rates and totals of states across the United States from 1960 to 2019. In this dataset, there are 21 variables. The crime types are divided into two main categories: property and violent crime. Property crimes refer to burglary, larceny, and motor-related crime while violent crimes refer to assault, murder, rape, and robbery.

REFERENCES:

USDA ERS - County-level Data Sets: Download Data. (2023). Www.ers.usda.gov. https://www.ers.usda.gov/data-products/county-level-data-sets/county-level-data-sets-download-data/

Digest of Education Statistics, 2014. (2013). Ed.gov; National Center for Education Statistics. https://nces.ed.gov/programs/digest/d14/tables/dt14_104.88.asp

Digest of Education Statistics, 2015. (2014). Ed.gov; National Center for Education Statistics. https://nces.ed.gov/programs/digest/d15/tables/dt15_104.88.asp

Digest of Education Statistics, 2016. (2015). Ed.gov; National Center for Education Statistics. https://nces.ed.gov/programs/digest/d16/tables/dt16_104.88.asp

Digest of Education Statistics, 2017. (2016). Ed.gov; National Center for Education Statistics. https://nces.ed.gov/programs/digest/d17/tables/dt17_104.88.asp

Digest of Education Statistics, 2018. (2017). Ed.gov; National Center for Education Statistics. https://nces.ed.gov/programs/digest/d18/tables/dt18_104.88.asp

Digest of Education Statistics, 2019. (2018). Ed.gov; National Center for Education Statistics. https://nces.ed.gov/programs/digest/d19/tables/dt19_104.88.asp

Digest of Education Statistics, 2020. (2019). Ed.gov; National Center for Education Statistics. https://nces.ed.gov/programs/digest/d20/tables/dt20_104.88.asp

Regional Economic Accounts: Download (no date) *BEA*. Available at: https://apps.bea.gov/regional/downloadzip.cfm (Accessed: 10 September 2023).