Examining Unemployment and Education in the United States ### Mt. SAC CISD 41 Capstone Project Fall 2021 #### By #### Paul Sandeen, "Fiona" Ping Xu, Ping Ju

Introduction

This project examines the connection between unemployment and education level on a county-by-county basis in all 50 states in the United States of America. The data for unemployment and education will first be evaluated separately, then combined together to look for correlations in education level and unemployment.

A step-by-step methodology will be taken to import the data, inspect the data, clean the data, perform an exploratory data analysis, graph the data, and apply statistical methods to analyze the data. All procedures will be performed with the Python programming language and associated libraries.

Intended Audience

This project is intended for students, educators, and anyone interested in a deeper understanding in how education and unemployment are associated in the United States. The ability to read and understand computer programs written in Python is required. Familiarity with fundamental statistical concepts and the ability to interpret graphs is assumed.

Tools Used

This project uses the Python programming language running in the Anaconda environment.

Associated Python libraries used for data analysis: Numpy, pandas, Matplotlib, Seaborn.

The project was composed as a Jupyter Notebook.

DataSet Source

The original dataset for both education level and unemployment is located at:

"USA Unemployment & Education Level"

https://www.kaggle.com/valbauman/student-engagement-online-learning-supplement/version/3?select=UIC_codes.csv

The dataset is maintained by: Val Bauman

The files used for the project: UIC_codes.csv, education.csv, unemployment.csv

Definitions

The terms City, Suburb, Town and Rural areas are defined as follows:

City is a large town, and it usually has the largest civilian labor force. The dataset provider described city as large-in a metro area with at least 1 million residents or more.

Suburb is the residential area of city or town, and it usually has the second largest civilian labor force; it is a micropolitan area adjacent to a large metro area.

Town is usually describing an urban area which is bigger than a village, but it is smaller than a city. A town is a noncore area adjacent to a small metro with a population of at least 2,500 residents.

Rural areas are characterized by a large open land, countryside or farming community. Rural areas are often on the outskirts of a large metro area.

Asking the appropriate questions

This project will attempt to answer the following questions:

1. A charity organization wants to explore which communities have residents that need help in completing their high school education. Which communities should they look at to do the most good? (Paul)

2. Even with states having low or moderate unemployment rates, are there counties with unusually high or low unemployment? (Paul)

3. Does having a bachelor's degree (or higher) or just a high school diploma correlate better with low unemployment? (Paul)

4. Which years have the highest and lowest unemployment rate over the course of 21 years? (Fiona)

5. Which states contribute the most and the least for the unemployment change from year 2019 to 2020? (Fiona)

6. Is there a significant change regarding percentages of people completing different diplomas between year 2000 and year 2015-2019? (Fiona)

7. What is the correlation between the adults with less than a high school diploma and unemployment rate in the year 2000? (Ping)

8. What will happen if the adults complete some college or complete a bachelor's degree or higher in the year 2000? (Ping)

9. How has the civilian labor force changed in City/Suburb/Town/Rural areas from 2000, 2010 and 2020? (Ping)

Import the Required Libraries

In [1]:
```
# Import the numpy and pandas libraries for data analysis methods
import numpy as np
import pandas as pd

# Import the Matplotlib and Seaborn libraries to create plots
import matplotlib.pyplot as plt
import seaborn as sns

# Import libraries required for Choropleth Map plots
import plotly.graph_objs as go
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)

# Import the warnings library and disable warnings from being printed in the
import warnings
warnings.filterwarnings('ignore')

# Display plots inside the Jupypter nodebook
%matplotlib inline

# Import the Z-test libraries used for statistical analysis
from statsmodels.stats.weightstats import ztest

# Import the Python Scipy Statistics library
import scipy.stats as stats

# Import the SciPy Pearson r module
from scipy.stats import pearsonr

# Imprt the libraries for the Chi-squared test
from scipy.stats import chi2_contingency
from scipy.stats import chi2
```

Import the Data

In [2]:
```
# Import the unemployment data from the unemployment.csv file
df_unemployment = pd.read_csv('data/unemployment.csv', sep=',')

# Import the education data from the education.csv file
df_education = pd.read_csv('data/education.csv', sep=',')

# Import the UIC data from the UIC_codes.csv file
df_uic = pd.read_csv('data/UIC_codes.csv', sep=',')
```

Inspect data (head, tail, info, dtype, etc.)

In [3]:
```
# Examine the data present in df_unemployment DataFrame using shape and info(
print('unemployement.csv:')
print('shape (rows, columns):', df_unemployment.shape, '\n')
print(df_unemployment.info(verbose=True, null_counts=True))
```

```
unemployement.csv:
shape (rows, columns): (3275, 93)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3275 entries, 0 to 3274
Data columns (total 93 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   FIPS_Code                        3275 non-null   int64
 1   State                            3275 non-null   object
 2   Area_name                        3275 non-null   object
 3   Rural_urban_continuum_code_2013  3219 non-null   float64
 4   Urban_influence_code_2013        3219 non-null   float64
 5   City/Suburb/Town/Rural           3219 non-null   object
 6   Metro_2013                       3222 non-null   float64
 7   Civilian_labor_force_2000        3270 non-null   object
 8   Employed_2000                    3270 non-null   object
 9   Unemployed_2000                  3270 non-null   object
 10  Unemployment_rate_2000           3270 non-null   float64
 11  Civilian_labor_force_2001        3270 non-null   object
 12  Employed_2001                    3270 non-null   object
 13  Unemployed_2001                  3270 non-null   object
 14  Unemployment_rate_2001           3270 non-null   float64
 15  Civilian_labor_force_2002        3270 non-null   object
 16  Employed_2002                    3270 non-null   object
 17  Unemployed_2002                  3270 non-null   object
 18  Unemployment_rate_2002           3270 non-null   float64
 19  Civilian_labor_force_2003        3270 non-null   object
 20  Employed_2003                    3270 non-null   object
 21  Unemployed_2003                  3270 non-null   object
 22  Unemployment_rate_2003           3270 non-null   float64
 23  Civilian_labor_force_2004        3270 non-null   object
 24  Employed_2004                    3270 non-null   object
 25  Unemployed_2004                  3270 non-null   object
 26  Unemployment_rate_2004           3270 non-null   float64
 27  Civilian_labor_force_2005        3263 non-null   object
 28  Employed_2005                    3263 non-null   object
 29  Unemployed_2005                  3263 non-null   object
 30  Unemployment_rate_2005           3263 non-null   float64
 31  Civilian_labor_force_2006        3263 non-null   object
 32  Employed_2006                    3263 non-null   object
 33  Unemployed_2006                  3263 non-null   object
 34  Unemployment_rate_2006           3263 non-null   float64
 35  Civilian_labor_force_2007        3270 non-null   object
 36  Employed_2007                    3270 non-null   object
 37  Unemployed_2007                  3270 non-null   object
 38  Unemployment_rate_2007           3270 non-null   float64
 39  Civilian_labor_force_2008        3270 non-null   object
 40  Employed_2008                    3270 non-null   object
 41  Unemployed_2008                  3270 non-null   object
```

```
 42  Unemployment_rate_2008                     3270 non-null   float64
 43  Civilian_labor_force_2009                  3270 non-null   object
 44  Employed_2009                              3270 non-null   object
 45  Unemployed_2009                            3270 non-null   object
 46  Unemployment_rate_2009                     3270 non-null   float64
 47  Civilian_labor_force_2010                  3272 non-null   object
 48  Employed_2010                              3272 non-null   object
 49  Unemployed_2010                            3272 non-null   object
 50  Unemployment_rate_2010                     3272 non-null   float64
 51  Civilian_labor_force_2011                  3272 non-null   object
 52  Employed_2011                              3272 non-null   object
 53  Unemployed_2011                            3272 non-null   object
 54  Unemployment_rate_2011                     3272 non-null   float64
 55  Civilian_labor_force_2012                  3272 non-null   object
 56  Employed_2012                              3272 non-null   object
 57  Unemployed_2012                            3272 non-null   object
 58  Unemployment_rate_2012                     3272 non-null   float64
 59  Civilian_labor_force_2013                  3272 non-null   object
 60  Employed_2013                              3272 non-null   object
 61  Unemployed_2013                            3272 non-null   object
 62  Unemployment_rate_2013                     3272 non-null   float64
 63  Civilian_labor_force_2014                  3272 non-null   object
 64  Employed_2014                              3272 non-null   object
 65  Unemployed_2014                            3272 non-null   object
 66  Unemployment_rate_2014                     3272 non-null   float64
 67  Civilian_labor_force_2015                  3272 non-null   object
 68  Employed_2015                              3272 non-null   object
 69  Unemployed_2015                            3272 non-null   object
 70  Unemployment_rate_2015                     3272 non-null   float64
 71  Civilian_labor_force_2016                  3272 non-null   object
 72  Employed_2016                              3272 non-null   object
 73  Unemployed_2016                            3272 non-null   object
 74  Unemployment_rate_2016                     3272 non-null   float64
 75  Civilian_labor_force_2017                  3272 non-null   object
 76  Employed_2017                              3272 non-null   object
 77  Unemployed_2017                            3272 non-null   object
 78  Unemployment_rate_2017                     3272 non-null   float64
 79  Civilian_labor_force_2018                  3272 non-null   object
 80  Employed_2018                              3272 non-null   object
 81  Unemployed_2018                            3272 non-null   object
 82  Unemployment_rate_2018                     3272 non-null   float64
 83  Civilian_labor_force_2019                  3272 non-null   object
 84  Employed_2019                              3272 non-null   object
 85  Unemployed_2019                            3272 non-null   object
 86  Unemployment_rate_2019                     3272 non-null   float64
 87  Civilian_labor_force_2020                  3193 non-null   object
 88  Employed_2020                              3193 non-null   object
 89  Unemployed_2020                            3193 non-null   object
 90  Unemployment_rate_2020                     3193 non-null   float64
 91  Median_Household_Income_2019               3193 non-null   object
 92  Med_HH_Income_Percent_of_State_Total_2019  3192 non-null   float64
dtypes: float64(25), int64(1), object(67)
memory usage: 2.3+ MB
None
```

Conclusion: The unemployment dataset has 93 columns and 3,275 rows of data. With so much data to work with, finding interesting patterns should not be a problem. The problem is that numerical data (such as the number of people unemployed in the year 2000) is stored as an object (a Python string), not as an integer or float. The data will have to be converted from strings to numbers to be usable.

Also, the column names have an underscore character (_) which should be removed.

In [4]:

```
# Display the first five rows of df_unemployment
df_unemployment.head()
```

Out[4]:

_2019	Unemployment_rate_2019	Civilian_labor_force_2020	Employed_2020	Unemployed_2020
268	3.1	8,640	8,067	573
680	2.7	24,661	23,653	1,008
545	2.7	19,592	18,618	974
9,154	2.9	315,957	296,282	19,675
1,107	2.7	40,132	38,146	1,986

In [5]:

```
# Examine the data present in df_education DataFrame using shape and info()
print('education.csv:')
print('shape (rows, columns):', df_education.shape, '\n')
print(df_education.info(verbose=True, null_counts=True))
```

```
education.csv:
shape (rows, columns): (3283, 48)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3283 entries, 0 to 3282
Data columns (total 48 columns):
 #   Column
Non-Null Count  Dtype
---  ------
--------------  -----
 0   FIPS Code
3283 non-null   int64
 1   State
3283 non-null   object
 2   Area name
3283 non-null   object
```

```
 3   2003 Rural-urban Continuum Code
3221 non-null   float64
 4   2003 Urban Influence Code
3221 non-null   float64
 5   2013 Rural-urban Continuum Code
3221 non-null   float64
 6   2013 Urban Influence Code
3221 non-null   float64
 7   City/Suburb/Town/Rural 2013
3221 non-null   object
 8   Less than a high school diploma, 1970
3186 non-null   object
 9   High school diploma only, 1970
3186 non-null   object
 10  Some college (1-3 years), 1970
3186 non-null   object
 11  Four years of college or higher, 1970
3186 non-null   object
 12  Percent of adults with less than a high school diploma, 1970
3186 non-null   float64
 13  Percent of adults with a high school diploma only, 1970
3186 non-null   float64
 14  Percent of adults completing some college (1-3 years), 1970
3186 non-null   float64
 15  Percent of adults completing four years of college or higher, 1970
3186 non-null   float64
 16  Less than a high school diploma, 1980
3267 non-null   object
 17  High school diploma only, 1980
3267 non-null   object
 18  Some college (1-3 years), 1980
3267 non-null   object
 19  Four years of college or higher, 1980
3267 non-null   object
 20  Percent of adults with less than a high school diploma, 1980
3267 non-null   float64
 21  Percent of adults with a high school diploma only, 1980
3267 non-null   float64
 22  Percent of adults completing some college (1-3 years), 1980
3267 non-null   float64
 23  Percent of adults completing four years of college or higher, 1980
3267 non-null   float64
 24  Less than a high school diploma, 1990
3271 non-null   object
 25  High school diploma only, 1990
3271 non-null   object
 26  Some college or associate's degree, 1990
3271 non-null   object
 27  Bachelor's degree or higher, 1990
3271 non-null   object
 28  Percent of adults with less than a high school diploma, 1990
3271 non-null   float64
 29  Percent of adults with a high school diploma only, 1990
3271 non-null   float64
 30  Percent of adults completing some college or associate's degree, 1990
3270 non-null   float64
 31  Percent of adults with a bachelor's degree or higher, 1990
```

```
3271 non-null   float64
 32  Less than a high school diploma, 2000
3272 non-null   object
 33  High school diploma only, 2000
3272 non-null   object
 34  Some college or associate's degree, 2000
3272 non-null   object
 35  Bachelor's degree or higher, 2000
3272 non-null   object
 36  Percent of adults with less than a high school diploma, 2000
3272 non-null   float64
 37  Percent of adults with a high school diploma only, 2000
3272 non-null   float64
 38  Percent of adults completing some college or associate's degree, 2000
3272 non-null   float64
 39  Percent of adults with a bachelor's degree or higher, 2000
3272 non-null   float64
 40  Less than a high school diploma, 2015-19
3273 non-null   object
 41  High school diploma only, 2015-19
3273 non-null   object
 42  Some college or associate's degree, 2015-19
3273 non-null   object
 43  Bachelor's degree or higher, 2015-19
3273 non-null   object
 44  Percent of adults with less than a high school diploma, 2015-19
3273 non-null   float64
 45  Percent of adults with a high school diploma only, 2015-19
3273 non-null   float64
 46  Percent of adults completing some college or associate's degree, 2015-19
3273 non-null   float64
 47  Percent of adults with a bachelor's degree or higher, 2015-19
3273 non-null   float64
dtypes: float64(24), int64(1), object(23)
memory usage: 1.2+ MB
None
```

Conclusion: Much like the Unemployment dataset, the Education dataset has a lot of data (48 columns and 3285 rows). And again, much of the numeric data is stored as type object (a Python string) not an integer or float, so type conversion will be needed.

In [6]:

```
# Display the first five rows of df_education
df_education.head()
```

Out[6]:

	FIPS Code	State	Area name	2003 Rural-urban Continuum Code	2003 Urban Influence Code	2013 Rural-urban Continuum Code	2013 Urban Influence Code	City/Suburb/Town/Rural 2013
0	1007	AL	Bibb County	1.0	1.0	1.0	1.0	City
1	1009	AL	Blount County	1.0	1.0	1.0	1.0	City
2	1021	AL	Chilton County	1.0	1.0	1.0	1.0	City
3	1073	AL	Jefferson County	1.0	1.0	1.0	1.0	City
4	1115	AL	St. Clair County	1.0	1.0	1.0	1.0	City

5 rows × 48 columns

In [7]:

```
# Display the first five rows of df_UIC
df_uic.head()
```

Out[7]:

	FIPS	State	County_Name	Population_2010	UIC_2013	Description	City/Suburb/Town/Rura
0	1007	AL	Bibb County	22,915	1	Large-in a metro area with at least 1 million ...	City
1	1009	AL	Blount County	57,322	1	Large-in a metro area with at least 1 million ...	City
2	1021	AL	Chilton County	43,643	1	Large-in a metro area with at least 1 million ...	City
3	1073	AL	Jefferson County	658,466	1	Large-in a metro area with at least 1 million ...	City
4	1115	AL	St. Clair County	83,593	1	Large-in a metro area with at least 1 million ...	City

In [8]:

```
# Examine the df_uic DataFrame using shape and info()
print('UIC.csv:')
print('shape (rows, columns):', df_uic.shape, '\n')
print(df_uic.info(verbose=True, null_counts=True))
```

```
UIC.csv:
shape (rows, columns): (3221, 7)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3221 entries, 0 to 3220
Data columns (total 7 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   FIPS                   3221 non-null   int64
 1   State                  3221 non-null   object
 2   County_Name            3221 non-null   object
 3   Population_2010        3221 non-null   object
 4   UIC_2013               3221 non-null   int64
 5   Description            3221 non-null   object
 6   City/Suburb/Town/Rural 3221 non-null   object
dtypes: int64(2), object(5)
memory usage: 176.3+ KB
None
```

Conclusion: The UIC.csv file does not contain much information that can be used for the project. The FIPS (Federal Information Processing System) codes are used to give a unique identification to a specific geographic area. The FIPS codes are contained in the other files to identify counties in the U.S. states.

Organizing data

In [9]:

```
# Function #1
# The df_unemployment and df_education DataFrames contains data from
# Puerto Rico (PR) and District of Columbia (DC).
# Ensure the data is only from the 50 US states
def fifty_states(df):
    states = ['AL', 'AK', 'AZ', 'AR', 'CA', 'CO', 'CT', 'DE', 'FL', 'GA', 'HI
              'IL', 'IN', 'IA', 'KS', 'KY', 'LA', 'ME', 'MD', 'MA', 'MI', 'MN
              'MT', 'NE', 'NV', 'NH', 'NJ', 'NM', 'NY', 'NC', 'ND', 'OH', 'OK
              'RI', 'SC', 'SD', 'TN', 'TX', 'UT', 'VT', 'VA', 'WA', 'WV', 'WI
    df_temp = df[df['State'].isin(states)]
    return df_temp
```

In [10]:

```
# Remove rows that are not from one of the 50 US states
df_unemployment_clean = fifty_states(df_unemployment)
# Set the index to FIPS code ()
df_unemployment_clean.set_index('FIPS_Code')
```

Out[10]:

	State	Area_name	Rural_urban_continuum_code_2013	Urban_influence_code_2013
FIPS_Code				
1007	AL	Bibb County, AL	1.0	1.0
1009	AL	Blount County, AL	1.0	1.0
1021	AL	Chilton County, AL	1.0	1.0
1073	AL	Jefferson County, AL	1.0	1.0
1115	AL	St. Clair County, AL	1.0	1.0
...	...	...	...	...
51000	VA	Virginia	NaN	NaN
53000	WA	Washington	NaN	NaN
54000	WV	West Virginia	NaN	NaN
55000	WI	Wisconsin	NaN	NaN
56000	WY	Wyoming	NaN	NaN

3193 rows × 92 columns

In [11]:

```
# Remove rows that are not from one of the 50 US states
df_education_clean = fifty_states(df_education)
# Set the index to FIPS code ()
df_education_clean.set_index('FIPS Code')
```

Out[11]:

	State	Area name	2003 Rural-urban Continuum Code	2003 Urban Influence Code	2013 Rural-urban Continuum Code	2013 Urban Influence Code	City/Suburb/Town/Rura 2013
FIPS Code							
1007	AL	Bibb County	1.0	1.0	1.0	1.0	City
1009	AL	Blount County	1.0	1.0	1.0	1.0	City
1021	AL	Chilton County	1.0	1.0	1.0	1.0	City
1073	AL	Jefferson County	1.0	1.0	1.0	1.0	City
1115	AL	St. Clair County	1.0	1.0	1.0	1.0	City
...	...	...	...	...	...	...	..
51560	VA	Clifton Forge city	6.0	6.0	NaN	NaN	NaN
53000	WA	Washington	NaN	NaN	NaN	NaN	NaN
54000	WV	West Virginia	NaN	NaN	NaN	NaN	NaN
55000	WI	Wisconsin	NaN	NaN	NaN	NaN	NaN
56000	WY	Wyoming	NaN	NaN	NaN	NaN	NaN

3201 rows × 47 columns

Clean data if required (recall replacing null values)

In [12]:

```
# Comment: There are many issues associated when choosing to drop rows or bac
# data with a calculated value such as mean or median. Large states like Cali
# New York will introduce bias when used to fill values from smaller states l
# and Hawaii, so simply dropping the rows containing missing data introduced
# Remove rows from df_unemployment that have empty cells
df_unemployment_clean.dropna(inplace=True)
# Remove Rows from df_unemployment that have empty cells
df_education_clean.dropna(inplace=True)
```

Rename any column that is not named correctly

In [13]:

```
# Rename the df_unemployment column FIPS_Code to FIPS Code to make it consist
#df_unemployment_clean.rename(columns={"FIPS_Code":"FIPS Code"}, inplace=True
df_unemployment_clean.columns = df_unemployment_clean.columns.str.replace("_"
```

In [14]:

```
# df.rename(columns={'oldName1': 'newName1', 'oldName2': 'newName2'}, inplace
# Rename the colunms of df_education to make them more consistent with previo
# Note: It is unclear if "Four years of college or higher" in prevous columns
# so degree status was not included.

df_education_clean.rename(columns={"Less than a high school diploma, 1970":"<
                            "High school diploma only, 1970":"HS Diploma, 197
                            "Some college (1-3 years), 1970":"Some college, 1
                            "Four years of college or higher, 1970":">= Bache
                            "Percent of adults with less than a high school d
                            "Percent of adults with a high school diploma onl
                            "Percent of adults completing some college (1-3 y
                            "Percent of adults completing four years of colle
                            inplace=True)

df_education_clean.rename(columns={"Less than a high school diploma, 1980":"<
                            "High school diploma only, 1980":"HS Diploma, 198
                            "Some college (1-3 years), 1980":"Some college, 1
                            "Four years of college or higher, 1980":">= Bache
                            "Percent of adults with less than a high school d
                            "Percent of adults with a high school diploma onl
                            "Percent of adults completing some college (1-3 y
                            "Percent of adults completing four years of colle
                            inplace=True)

df_education_clean.rename(columns={"Less than a high school diploma, 1990":"<
                            "High school diploma only, 1990":"HS Diploma, 199
                            "Some college or associate's degree, 1990":"Some 
                            "Bachelor's degree or higher, 1990":">= Bachelors
                            "Percent of adults with less than a high school d
                            "Percent of adults with a high school diploma onl
                            "Percent of adults completing some college or ass
                            "Percent of adults with a bachelor's degree or hi
```

```
                            inplace=True)

df_education_clean.rename(columns={"Less than a high school diploma, 2000":"<
                            "High school diploma only, 2000":"HS Diploma, 200
                            "Some college or associate's degree, 2000":"Some 
                            "Bachelor's degree or higher, 2000":">= Bachelors
                            "Percent of adults with less than a high school d
                            "Percent of adults with a high school diploma onl
                            "Percent of adults completing some college or ass
                            "Percent of adults with a bachelor's degree or hi
                            inplace=True)

df_education_clean.rename(columns={"Less than a high school diploma, 2015-19"
                            "High school diploma only, 2015-19":"HS Diploma, 
                            "Some college or associate's degree, 2015-19":"So
                            "Bachelor's degree or higher, 2015-19":">= Bachel
                            "Percent of adults with less than a high school d
                            "Percent of adults with a high school diploma onl
                            "Percent of adults completing some college or ass
                            "Percent of adults with a bachelor's degree or hi
                            inplace=True)
```

Change any data type if required

In [15]:

```
# Remove the ',' character from 'Civilian_labor_force_20xx', 'Employed_20xx',
# of df_unemployment and convert to a dtype float

# Create a list of column headers that contain string data to convert to dtyp
column_to_clean = ['Civilian labor force', 'Employed','Unemployed']

# Iterate through all of the columns in the DataFrame and find the columns th
for col in df_unemployment_clean.columns:
    # The splice col[:len(col)-5] removes the last 5 characters from the colu
    if col[:len(col)-5] in column_to_clean:
        # Drop the ',' character from the data and convert to dtype float
        df_unemployment_clean[col] = df_unemployment_clean[col].str.replace('



# Remove the ',' character from 'Median_Household_Income_2019' and convert to
df_unemployment_clean['Median Household Income 2019'] = df_unemployment_clean
```

In [16]:

```
# Remove the ',' character from 'Less than a high school diploma_year', 'High
# 'Some college (1-3 years)_year' and 'Four years of college or higher_year'
# of education and convert to a dtype float

# Create a list of column headers that contain string data to convert to dtyp
column_to_clean = ['< HS Diploma', 'HS Diploma','Some college', '>= Bachelors

# Iterate through all of the columns in the DataFrame and find the columns th
for col in df_education_clean.columns:
    # The splice col[:len(col)-6] removes the last 6 characters from the colu
    if col[:len(col)-6] in column_to_clean:
        # Drop the ',' character from the data and convert to dtype float
        df_education_clean[col] = df_education_clean[col].str.replace(',', ''
```

Use describe and write your conclusion

In [17]:

```
df_unemployment_clean.describe()
```

Out[17]:

	FIPS Code	Rural urban continuum code 2013	Urban influence code 2013	Metro 2013	Civilian labor force 2000	Employed 2000
count	3128.000000	3128.000000	3128.000000	3128.000000	3.128000e+03	3.128000e+03
mean	30458.390665	5.013107	5.270460	0.369885	4.528774e+04	4.348215e+04
std	15142.828587	2.701932	3.492291	0.482850	1.475892e+05	1.410400e+05
min	1001.000000	1.000000	1.000000	0.000000	4.900000e+01	4.500000e+01
25%	18796.500000	2.000000	2.000000	0.000000	5.080000e+03	4.847750e+03
50%	29188.000000	6.000000	5.000000	0.000000	1.171600e+04	1.119250e+04
75%	45087.500000	7.000000	8.000000	1.000000	3.020775e+04	2.888550e+04
max	56045.000000	9.000000	12.000000	1.000000	4.665167e+06	4.413213e+06

8 rows × 90 columns

In [18]:

```
df_education_clean.describe()
```

Out[18]:

	FIPS Code	2003 Rural-urban Continuum Code	2003 Urban Influence Code	2013 Rural-urban Continuum Code	2013 Urban Influence Code	< HS Diploma, 1970	H
count	3124.000000	3124.000000	3124.000000	3124.000000	3124.000000	3.124000e+03	3.1
mean	30478.789693	5.125160	5.446863	5.002561	5.258003	1.670135e+04	1.0
std	15087.865660	2.678532	3.464017	2.702093	3.489064	5.718080e+04	4.0
min	1001.000000	1.000000	1.000000	1.000000	1.000000	3.300000e+01	8.0
25%	19012.500000	3.000000	2.000000	2.000000	2.000000	2.997000e+03	1.2
50%	29185.500000	6.000000	5.000000	6.000000	5.000000	5.931000e+03	2.1
75%	45083.500000	7.000000	8.000000	7.000000	8.000000	1.196250e+04	6.1
max	56045.000000	9.000000	12.000000	9.000000	12.000000	1.506170e+06	1.2

8 rows × 41 columns

Conclusion: Now all numerical data is stored in numerical format, no columns have the underscore character (_), and rows that are missing data have been dropped.

When dropping rows, it is important to consider the effect this will have on the data. The project group considered multiple options, such as back-filling missing data, or using column means to fill missing data. Filling in missing data presents many issues in this dataset. Can the column mean of data from states like California and New York be comparable to states like Alaska and Wyoming? If a dataset that includes data from California is used to fill information from a state like Alaska, this can introduce bias. In other words, a state like California may not be representative of a state like Alaska.

Another option is to use inner-state data; for example, using data from counties in Alaska to fill missing data from other counties in Alaska. But this poses problems as well, because even within Alaska there are counties with larger city areas and counties that are rural, and again using the mean of inner-state data in Alaska would introduce bias.

Select only required columns

In [19]:

```
# Merge the df_education_clean and df_unemployment_clearn DataFrames into a s
df_merged = pd.merge(df_education_clean, df_unemployment_clean, how='inner', 
```

In [20]:

```
# Save the merged DataSet to a CSV file
df_merged.to_csv('data/merged.csv')
```

Conclusion: Now with all the data cleaned and columns selected, the Unemployment and Education datasets are merged into one unified DataFrame, df_merged, that can be used throughout the project.

Pivot Tables

In [21]:

```
# Create a pivot table stored in a new DataFrame that will provide the total
# the years 2000 and 2010
df_pt1 = pd.pivot_table(df_merged, index='City/Suburb/Town/Rural 2013', value

# Calculate the percent change in total labor force using the formula
# % Difference = [(New_value - Previous_value) / (Previous_value)] * 100%
df_pt1['% Change'] = ((df_pt1["Civilian labor force 2010"] - df_pt1["Civilian
df_pt1
```

Out[21]:

City/Suburb/Town/Rural 2013	Civilian labor force 2000	Civilian labor force 2010	% Change
City	120194652.0	131553875.0	9.450689
Rural	5447956.0	5454747.0	0.124652
Suburb	8545086.0	8708303.0	1.910069
Town	7394947.0	7531767.0	1.850182

Conclusion: The Pivot Table aggregates the civilian labor force into different categories (in this case, counties described by City, Rural, Suburb and Town). Between 2000 and 2010, the City counties had the largest increase (9.45%), while rural areas had the smallest growth at just 0.12 %. Both Suburb and Town counties grew at just under 2%.

Analyzing data with groupby()

In [22]:
```
# The dataset stores data on a per-county basis for each county in the U.S.

# Create a new DataFrame to store per-state aggregate data
df_data_by_state = pd.DataFrame()

# Store the civilian labor force by state for the year 2000
df_data_by_state = df_merged[['Civilian labor force 2000', 'State_x']].groupb

# Store the total unemployed by state for the year 2000
df_data_by_state[['State_x','Unemployed 2000']] = df_merged[['Unemployed 2000

# Calculate the unemployement rate by state for the year 2000
df_data_by_state['Unemployment Rate 2000'] = (df_data_by_state['Unemployed 20

# Totalize individuals with 'Less than a high school diploma, 2000'
df_data_by_state[['State_x','< HS Diploma, 2000']] = df_merged[['< HS Diploma

# Totalize individuals with 'High school diploma, 2000'
df_data_by_state[['State_x','HS Diploma, 2000']] = df_merged[['HS Diploma, 20

# Totalize individuals with 'Some college (1-3 years), 2000'
df_data_by_state[['State_x','Some college, 2000']] = df_merged[['Some college

# Totalize individuals with 'Four years of college or higher, 2000'
df_data_by_state[['State_x','>= Bachelors, 2000']] = df_merged[['>= Bachelors

# Calculate 'Percent of adults with less than a high school diploma, 2000'
df_data_by_state['% < HS Diploma, 2000'] = (df_data_by_state['< HS Diploma, 2

# Calculate 'Percent of adults with a High school diploma only, 2000'
df_data_by_state['% HS Diploma, 2000'] = (df_data_by_state['HS Diploma, 2000'

# Calculate 'Percent of adults completing some college or associate's degree,
df_data_by_state["% Some College, 2000"] = (df_data_by_state['Some college, 2

# Calculate "Percent of adults with a bachelor's degree or higher 2000"
df_data_by_state["% >= Bachelors, 2000"] = (df_data_by_state['>= Bachelors, 2

# Display the DataFrame to verify it was created correctly
df_data_by_state.head()
```

Out[22]:

	State_x	Civilian labor force 2000	Unemployed 2000	Unemployment Rate 2000	< HS Diploma, 2000	HS Diploma, 2000	Some college, 2000	Bache 2
0	AK	305779.0	18936.0	6.192708	40820.0	98565.0	129677.0	909
1	AL	2147180.0	99447.0	4.631517	714081.0	877216.0	746495.0	5496
2	AR	1260517.0	52677.0	4.179000	427449.0	590416.0	424907.0	2884
3	AZ	2502987.0	99302.0	3.967340	615126.0	787024.0	1074683.0	7649
4	CA	16837548.0	825923.0	4.905245	4942743.0	4288452.0	6397739.0	56699

Conclusion: The groupby() function allows multiple columns of data to be aggregated based on a separate column. Percent unemployment is given on a per-county basis in the dataset, but this value cannot be used when aggregating data for an entire state (since certain counties are much larger than others in terms of population). Therefore, the percent unemployment for the state must be calculated separately.

Exploratory Data Analysis with an Emperiacal Cumulative Distribution Function (ECDF)

The ECDF calculates the the fraction of data less than or equal to a specified value, from the lowest value to the largest value. The ECDF can be plotted to show how the data is distributed.

In [23]:

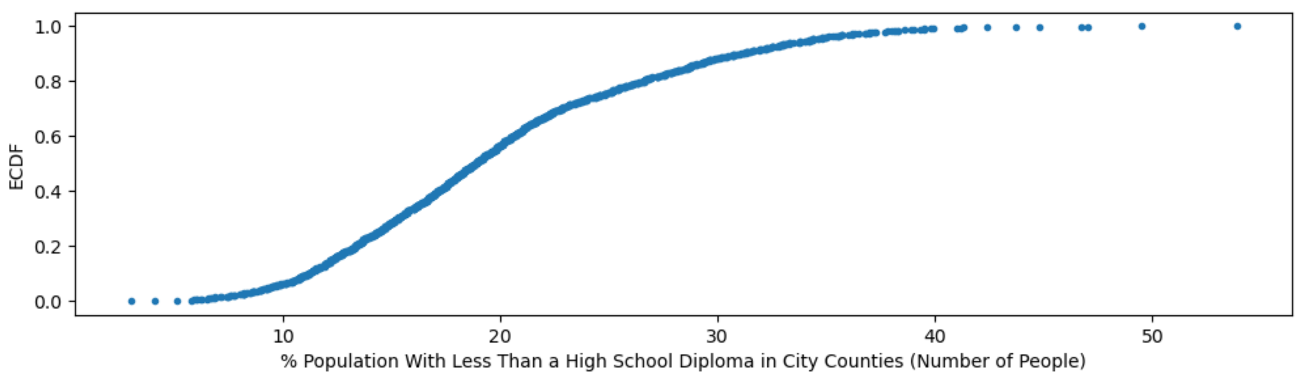
```
# Function #2 ecdf()
# Calculate the ECDF to determine the distibution of the data given a pandas
def ecdf(data):

    '''Calculate ECDF for a one-dimensional pandas column'''

    # Number of data points: n
    n = len(data)

    # x-data for the ECDF: x
    x = data.sort_values()

    # y-data for the ECDF: y
    y = np.arange(1, n+1) / n

    # return x and y values for ECDF:
    return x, y
```

In [24]:

```
# Calculate ECDF values for % people with less than a high school diploma liv

x_rural, y_rural = ecdf(df_merged.loc[df_merged['City/Suburb/Town/Rural 2013'
x_city, y_city = ecdf(df_merged.loc[df_merged['City/Suburb/Town/Rural 2013'] 
x_suburb, y_suburb = ecdf(df_merged.loc[df_merged['City/Suburb/Town/Rural 201
x_town, y_town = ecdf(df_merged.loc[df_merged['City/Suburb/Town/Rural 2013'] 
```

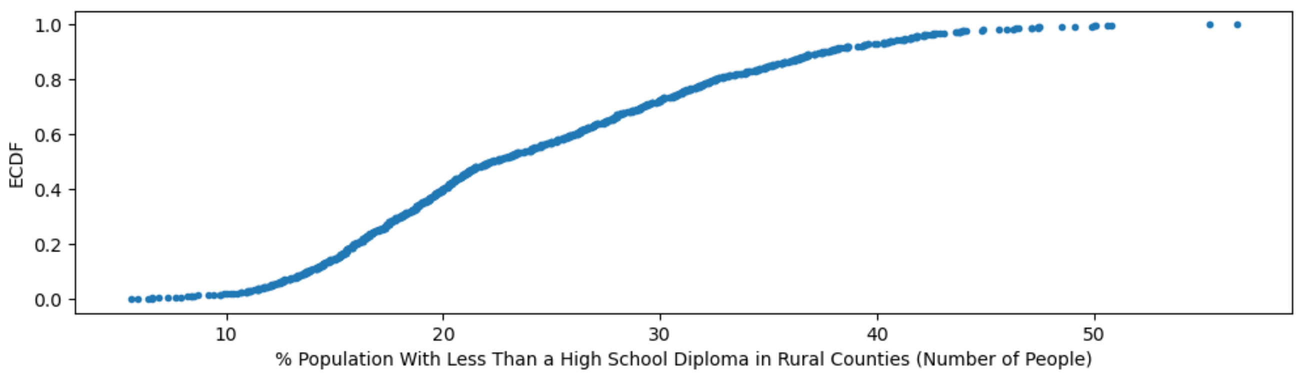
In [25]:

```
# Plot ECDF values for people with less than a high school diploma living in

# Set the default style
plt.style.use('default')

# Adjust the figure size to stretch it horizontally to clearly see the outlie
plt.rcParams["figure.figsize"] = (12,3)

# Label the x-axis and y-axis
_ = plt.xlabel('% Population With Less Than a High School Diploma in City Cou
_ = plt.ylabel('ECDF')

# Set the default style
plt.style.use('default')

# Display the tick labels as whole numbers (the default was scientific notati
plt.ticklabel_format(style='plain', axis='x', scilimits=(0,0))

# Plot the x and y ECDF values
_ = plt.plot(x_city, y_city, marker = '.', linestyle = 'none')
```



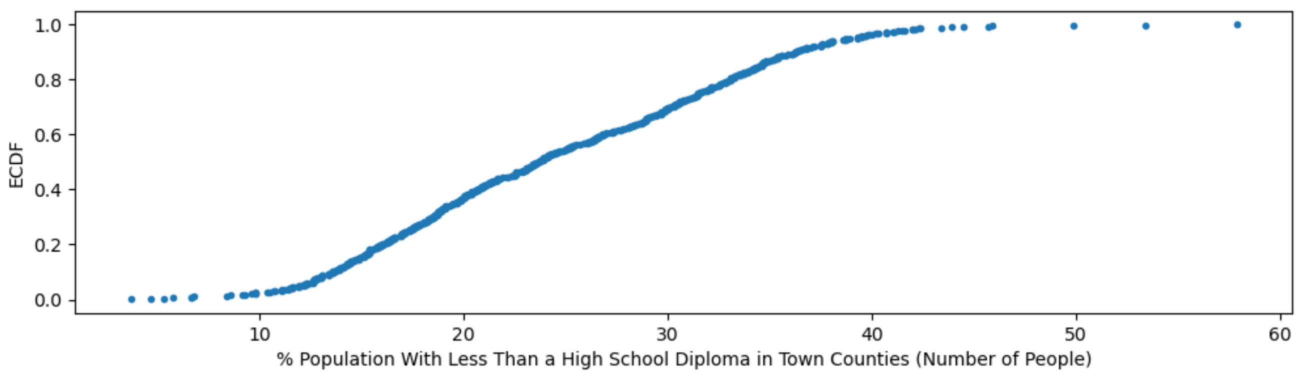
Conclusion: The ECDF plot with percent of the population with less than a high school diploma in the city has outliers on both ends and resembles an s-curve. In the City, 50% of counties have a population where less than 20% of the population has less than a high school diploma.

In [26]:
```
# Plot ECDF values for people with less than a high school diploma living in .

# Set the default style
plt.style.use('default')

# Adjust the figure size to stretch it horizontally to clearly see the outlie
plt.rcParams["figure.figsize"] = (12,3)

# Label the x-axis and y-axis
_ = plt.xlabel('% Population With Less Than a High School Diploma in Rural Co
_ = plt.ylabel('ECDF')

# Display the tick labels as whole numbers (the default was scientific notati
plt.ticklabel_format(style='plain', axis='x', scilimits=(0,0))

# Plot the x and y ECDF values
_ = plt.plot(x_rural, y_rural, marker = '.', linestyle = 'none')
```



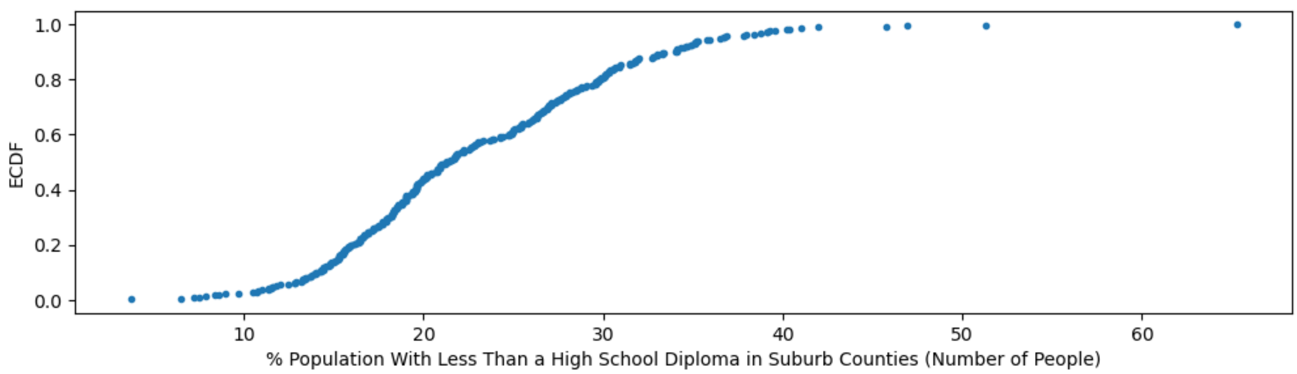
Conclusion: The ECDF plot with percent of the population with less than a high school diploma in the rural areas has outliers on the top end and resembles an s-curve. In rural areas, 50% of counties have a population where less than 25% of the population has less than a high school diploma.

In [27]:
```
# Plot ECDF values for people with less than a high school diploma living in

# Set the default style
plt.style.use('default')

# Adjust the figure size to stretch it horizontally to clearly see the outlie
plt.rcParams["figure.figsize"] = (12,3)

# Label the x-axis and y-axis
_ = plt.xlabel('% Population With Less Than a High School Diploma in Town Cou
_ = plt.ylabel('ECDF')

# Display the tick labels as whole numbers (the default was scientific notati
plt.ticklabel_format(style='plain', axis='x', scilimits=(0,0))

# Plot the x and y ECDF values
_ = plt.plot(x_town, y_town, marker = '.', linestyle = 'none')
```



Conclusion: The ECDF plot with percent of the population with less than a high school diploma in the town areas has outliers on both ends and resembles an s-curve. In town areas, 50% of counties have a population where less than 25% of the population has less than a high school diploma. This is similar to rural areas.

In [28]:
```
# Plot ECDF values for people with less than a high school diploma living in

# Set the default style
plt.style.use('default')

# Adjust the figure size to stretch it horizontally to clearly see the outlie
plt.rcParams["figure.figsize"] = (12,3)

# Label the x-axis and y-axis
_ = plt.xlabel('% Population With Less Than a High School Diploma in Suburb Co
_ = plt.ylabel('ECDF')

# Display the tick labels as whole numbers (the default was scientific notati
plt.ticklabel_format(style='plain', axis='x', scilimits=(0,0))

# Plot the x and y ECDF values
_ = plt.plot(x_suburb, y_suburb, marker = '.', linestyle = 'none')
```



Conclusion: The ECDF plot with percent of the population with less than a high school diploma in the suburb areas has outliers on both ends (with more at the top) and resembles an s-curve. In suburb areas, 50% of counties have a population where less than 25% of the population has less than a high school diploma. This is similar to rural areas and town areas.

In [ ]:
```
```

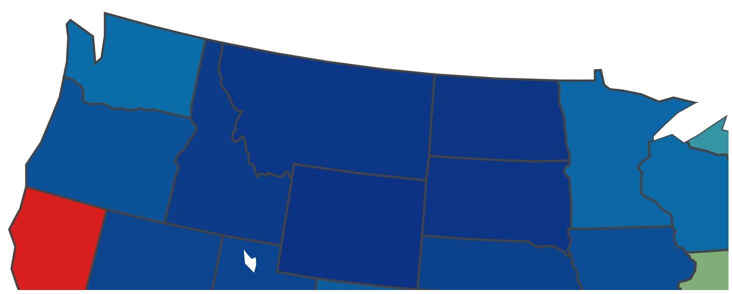
Data Visualization

In [29]:

```
# Display labor force information for the U.S. using Choropleth - a graphics
# for each U.S. State.
# Choropleth requires 'data' (which indicates the type of plot - U.S. States
data = dict(type = 'choropleth',
            locations = df_data_by_state['State_x'],
            locationmode = 'USA-states',
            colorscale = 'Portland',
            text = '(Civilian Labor Force for state)',
            z = df_data_by_state['Civilian labor force 2000'],
            colorbar = {'title':'Civilian Labor Force by State (Year 2000)'})

layout = dict(geo = {'scope':'usa'})
```

In [30]:

```
# Draw the Choropleth plot
choromap = go.Figure(data = [data], layout = layout)
iplot(choromap, validate=False)
```



Conclusion: Looking at the entire U.S., California dominates the civilian labor force with almost 17 million workers. Texas (10.4 million workers) and New York (9.14 million workers) also stand out. Other areas of interest are the Midwest with Illinois (6.49 million workers), Pennsylvania (6.11 million workers) and Ohio (5.78 million workers).

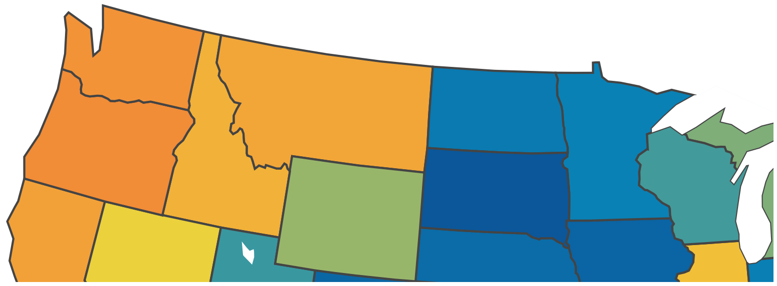
Question 1: A charity organization wants to explore which communities have residents that need help in completing their high school education. Which communities should they look at to do the most good? (Paul)

In [31]:

```
# Display unemployment information for the U.S. using Choropleth - a graphics
# for each U.S. State.
# Choropleth requires 'data' (which indicates the type of plot - U.S. States
data = dict(type = 'choropleth',
            locations = df_data_by_state['State_x'],
            locationmode = 'USA-states',
            colorscale = 'Portland',
            text = 'Unemployment rate by sate (2000)',
            z = df_data_by_state['Unemployment Rate 2000'],
            colorbar = {'title':'Unemployment Rate by State for the Year 2000

layout = dict(geo = {'scope':'usa'})
```

In [32]:

```
# Draw the Choropleth plot
choromap = go.Figure(data = [data], layout = layout)
iplot(choromap, validate=False)
```

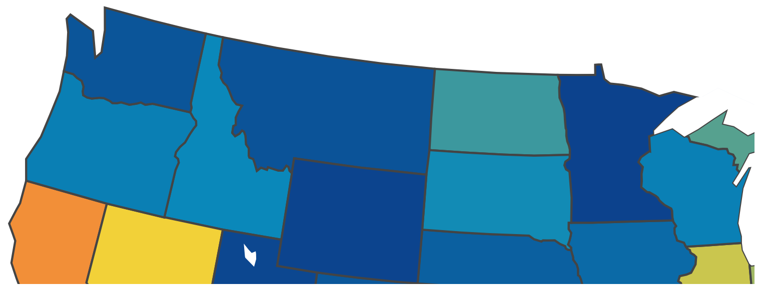


Conclusion: The western states stand out as having higher unemployment than other geographic regions, notably Oregon with 5.20% unemployment. The center of the country has low comparatively low unemployment (Nebraska has only 2.79% unemployment). Hawaii has the highest unemployment rate at 6.19%.

In [33]:

```
# Display education information for the U.S. using Choropleth - a graphics li
# for each U.S. State.
# Choropleth requires 'data' (which indicates the type of plot - U.S. States
data = dict(type = 'choropleth',
            locations = df_data_by_state['State_x'],
            locationmode = 'USA-states',
            colorscale = 'Portland',
            text = '% with less than high school diploma (2000)',
            z = df_data_by_state['% < HS Diploma, 2000'],
            colorbar = {'title':'% with less than high school diploma (2000)'

layout = dict(geo = {'scope':'usa'})
```

In [34]:
```
# Draw the Choropleth plot
choromap = go.Figure(data = [data], layout = layout)
iplot(choromap, validate=False)
```



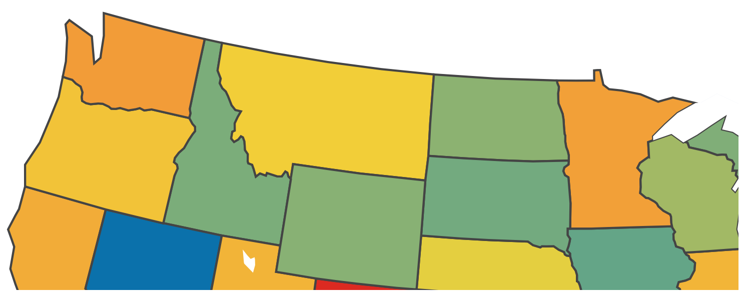
Conclusion: The southeast region has the highest concentration of counties with a population with less than a high school diploma. Kentucky, Mississippi, and Louisiana all have a population with 25% or more who do not have a high school diploma. California follows closely at 23%.

In [35]:

```
# Display education information for the U.S. using Choropleth - a graphics li
# for each U.S. State.
# Choropleth requires 'data' (which indicates the type of plot - U.S. States
data = dict(type = 'choropleth',
            locations = df_data_by_state['State_x'],
            locationmode = 'USA-states',
            colorscale = 'Portland',
            text = "% with bachelor's degree or higher (2000)",
            z = df_data_by_state["% >= Bachelors, 2000"],
            colorbar = {'title':"% with bachelor's degree or higher (2000)"})

layout = dict(geo = {'scope':'usa'})
```

In [36]:

```
# Draw the Choropleth plot
choromap = go.Figure(data = [data], layout = layout)
iplot(choromap, validate=False)
```



Conclusion: The northeast region has a large population with a bachelor's degree or higher, with Massachusetts leading the way at 33%. Colorado stands out near the center of the country with 32.7% of the population having a bachelor's degree or higher.

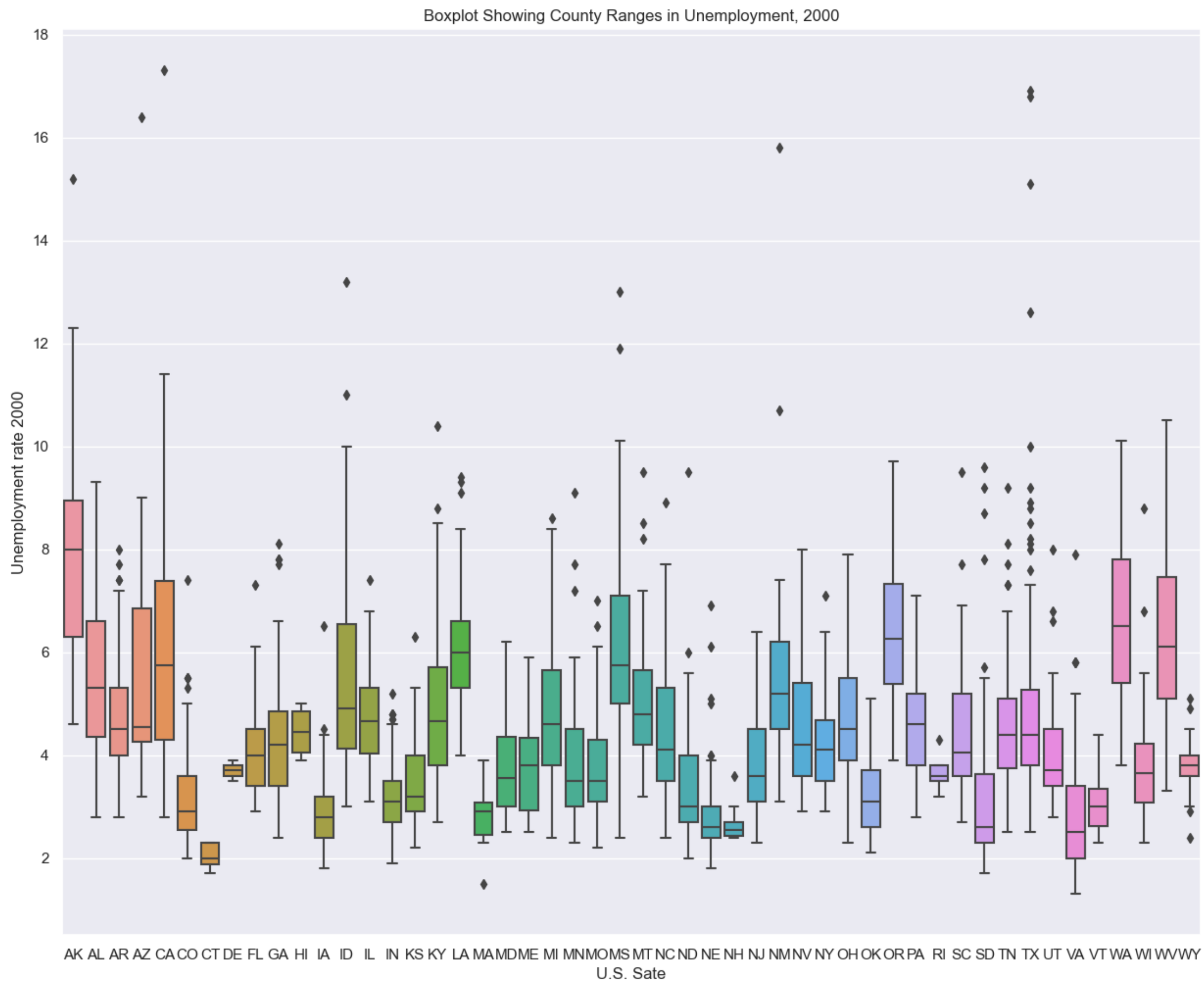
Question 2: Even in U.S. states with low or moderate unemployment rates, are there counties with unusually high or low unemployment? (Paul)

In [37]:

```
# Create a sorted list of state names
state_names = df_merged['State_x'].unique().tolist()
state_names.sort()

# Set the size of the boxplot to 15 by 12
sns.set(rc={'figure.figsize':(15,12)})

# Create a Seaborn boxplot to show ranges of employment by country within a s
bp1 = sns.boxplot(x='State_x', y='Unemployment rate 2000', data=df_merged, or

# Change the X-axis label
bp1.set(xlabel = "U.S. Sate")

# Change the boxplot title
bp1.set_title('Boxplot Showing County Ranges in Unemployment, 2000')
```

Out[37]:

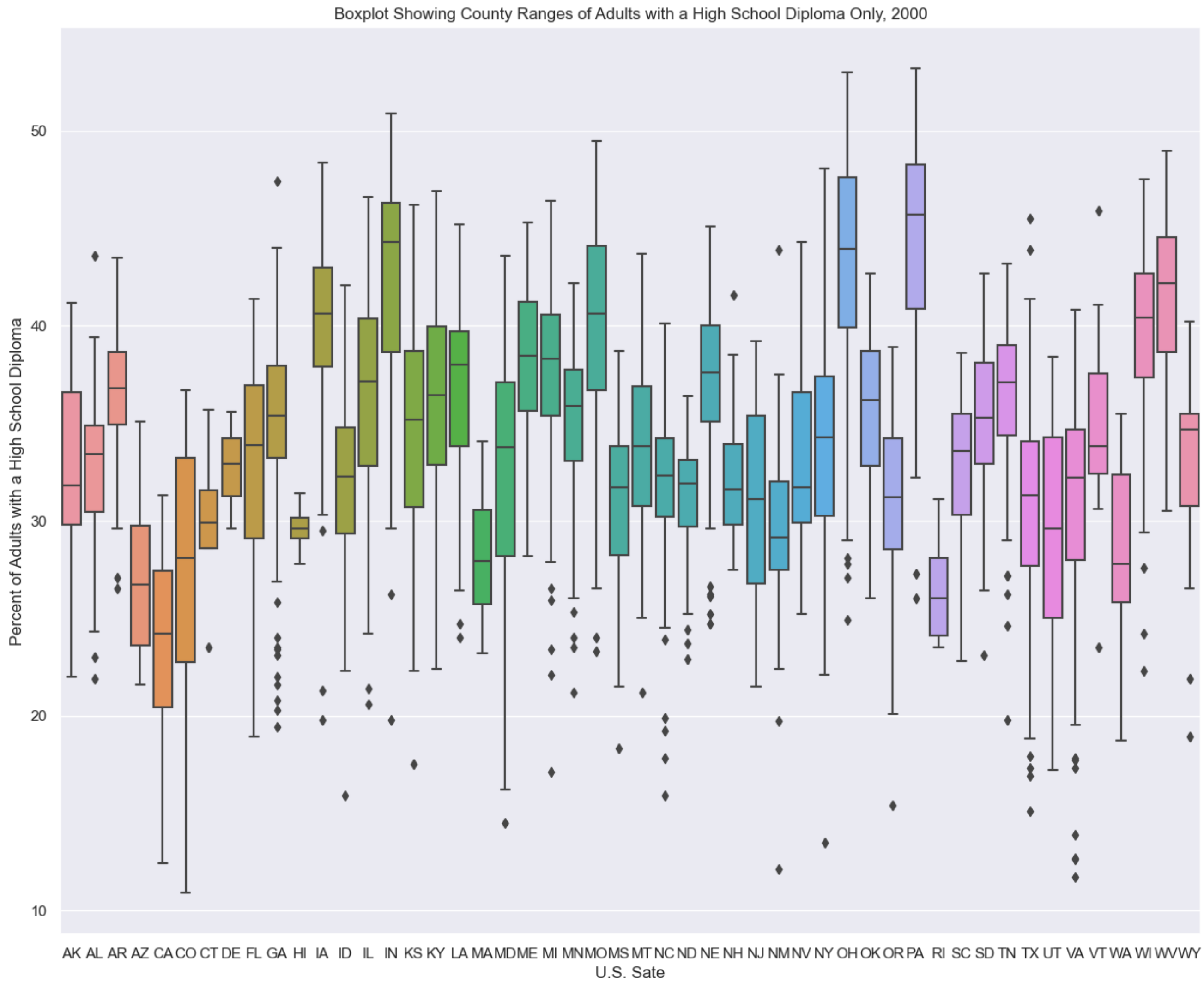
```
Text(0.5, 1.0, 'Boxplot Showing County Ranges in Unemployment, 2000')
```



Conclusion: The maps showed unemployment levels at the state level, but the data is defined at the county level; this allows for examining information inside each state, and boxplots can be used for this. The western states stood out as having high unemployment, with California at 6.17%. Looking at the county data for California, outlier counties can be seen reaching 17% unemployment, but the upper limit on the boxplot (top of the IQR) is approximately 11 % unemployment. Texas in particular has multiple outlier counties, with peak IQR of about 7% but an outlier reaching 17% unemployment.

In [38]:
```
# Create a sorted list of state names
state_names = df_merged['State_x'].unique().tolist()
state_names.sort()

# Set the size of the boxplot to 15 by 12
sns.set(rc={'figure.figsize':(15,12)})

# Create a Seaborn boxplot to show ranges of education by county within a sta
#bp1 = sns.boxplot(x='State_x', y='Percent of adults with a high school diplo
bp1 = sns.boxplot(x='State_x', y='% HS Diploma, 2000', data=df_merged, order=

# Change the X-axis label
bp1.set(xlabel = "U.S. Sate")

# Change the Y-axis label
bp1.set(ylabel = "Percent of Adults with a High School Diploma")

# Change the boxplot title
bp1.set_title('Boxplot Showing County Ranges of Adults with a High School Dip
```

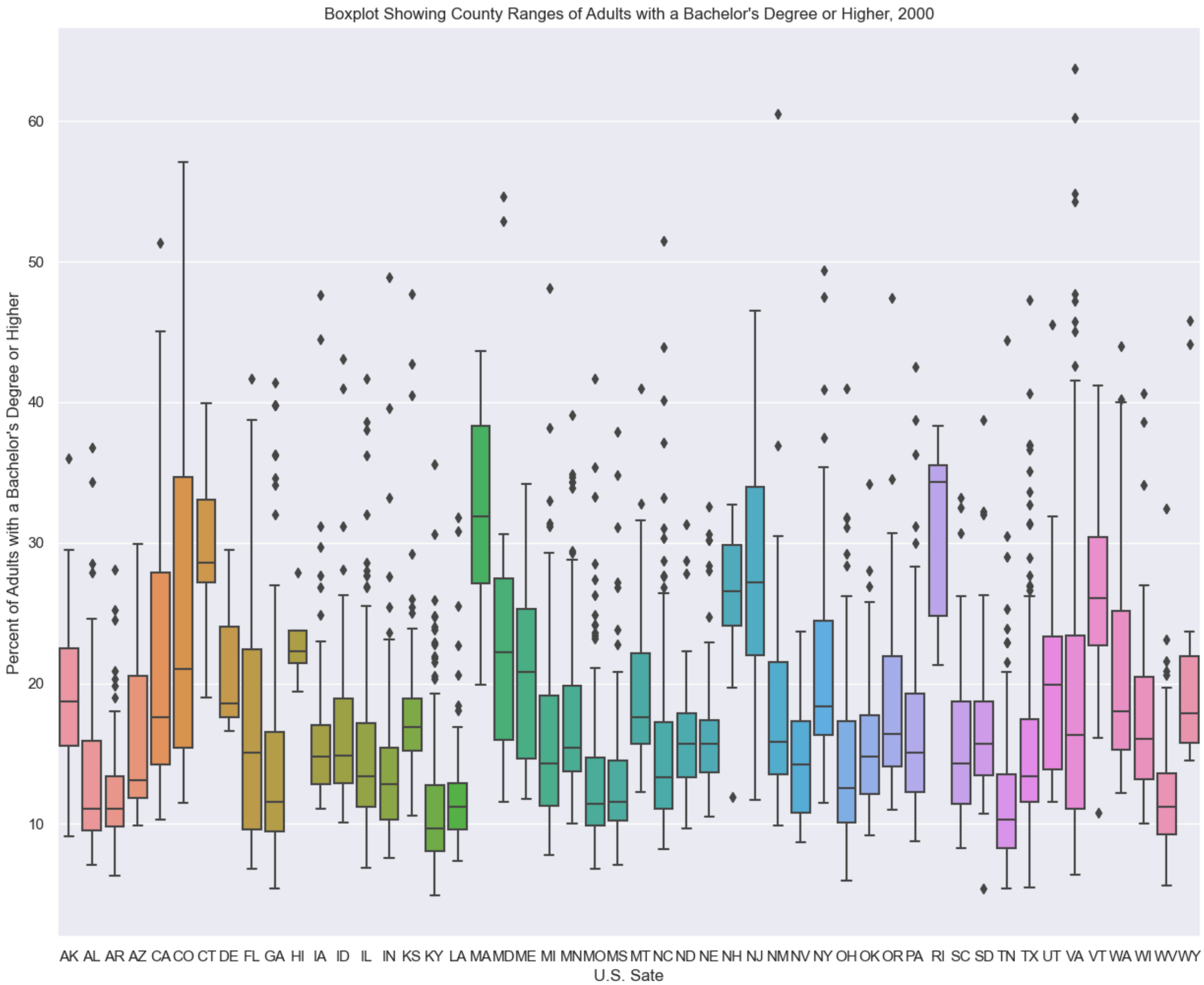
```
Out[38]: Text(0.5, 1.0, 'Boxplot Showing County Ranges of Adults with a High School Dip
loma Only, 2000')
```



Conclusion: Looking at the percent of adults with a high school diploma only, Georgia has a large number of outliers below the lower limit. This is true for other southern states like North Carolina, but also Midwest states like Michigan and Minnesota.

In [39]:
```
# Create a sorted list of state names
state_names = df_merged['State_x'].unique().tolist()
state_names.sort()

# Set the size of the boxplot to 15 by 12
sns.set(rc={'figure.figsize':(15,12)})

# Create a Seaborn boxplot to show ranges of education by county within a sta
bp1 = sns.boxplot(x='State_x', y="% >= Bachelors, 2000", data=df_merged, orde

# Change the X-axis label
bp1.set(xlabel = "U.S. Sate")

# Change the Y-axis label
bp1.set(ylabel = "Percent of Adults with a Bachelor's Degree or Higher")

# Change the boxplot title
bp1.set_title("Boxplot Showing County Ranges of Adults with a Bachelor's Degr
```

Out[39]:
```
Text(0.5, 1.0, "Boxplot Showing County Ranges of Adults with a Bachelor's Degr
ee or Higher, 2000")
```



Conclusion: For the percent of adults with a bachelor's degree or higher, the outliers in nearly every state stand out. In each state, there are counties with the percent of the population with a college degree is much higher than the rest of the state. This is particularly true in southern states like Kentucky, Missouri, Mississippi, and North Carolina.

Quantative Data Exploratory Descriptive Statistics: Correlation coefficients

In [40]:
```
# Examine if a correlation exists between unemployment and having less than a

# Use the df_data_by_state DataFrame to to examine the correlation between Un
# the Percent of adults with less than a high school diploma 2000
df_data_by_state[['State_x', 'Unemployment Rate 2000','% HS Diploma, 2000']].
```

Out[40]:

	State_x	Unemployment Rate 2000	% HS Diploma, 2000
0	AK	6.192708	27.375365
1	AL	4.631517	30.380827
2	AR	4.179000	34.104436
3	AZ	3.967340	24.277414
4	CA	4.905245	20.134617

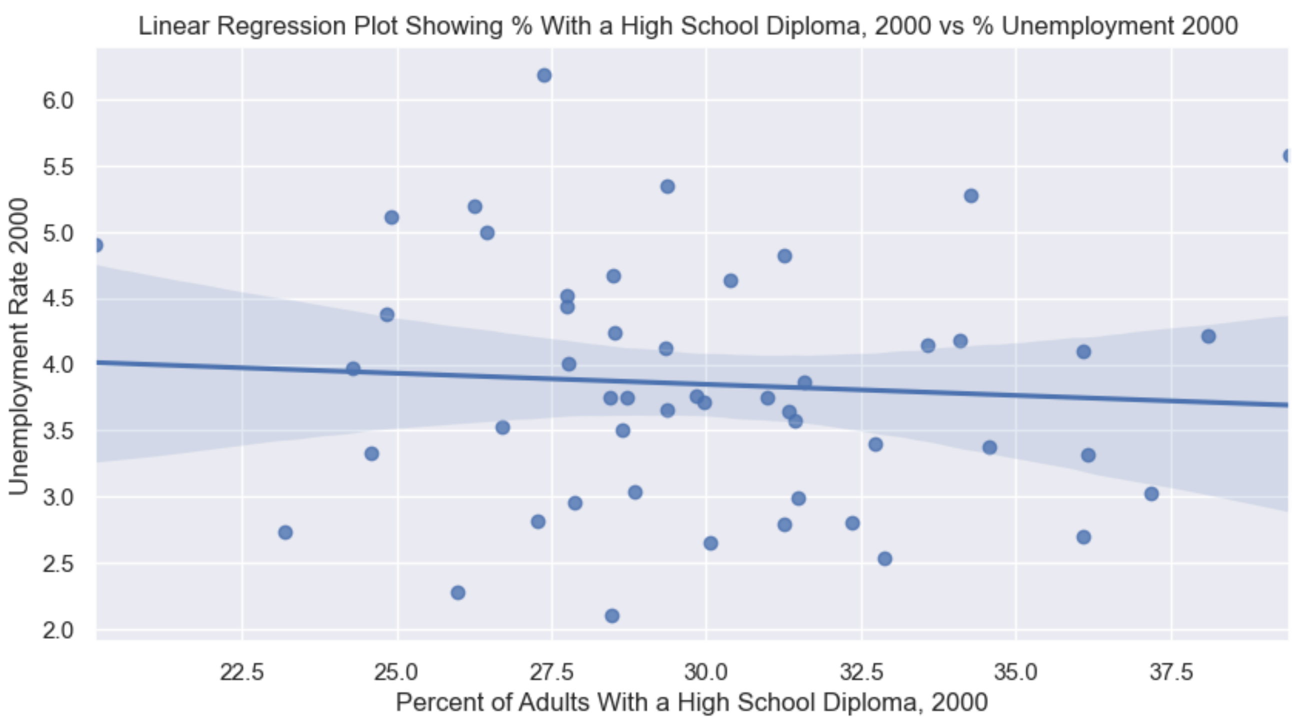
Question 3: Does low unemployment correlate better with having a bachelor's degree (or higher) or just a high school diploma? (Paul)

In [41]:

```
# Set the size of the boxplot to 15 by 12
sns.set(rc={'figure.figsize':(10,5)})

# Create a linear regression plot to visualize unemployment and people with l
corr_p1 = sns.regplot(x="% HS Diploma, 2000", y="Unemployment Rate 2000", dat

# Set the X-axis label
corr_p1.set(xlabel='Percent of Adults With a High School Diploma, 2000')

# Change the boxplot title
corr_p1.set_title("Linear Regression Plot Showing % With a High School Diplom
```

Out[41]:

```
Text(0.5, 1.0, 'Linear Regression Plot Showing % With a High School Diploma, 2
000 vs % Unemployment 2000 ')
```



Conclusion: The linear regression plot of Unemployment and percent of people with less a high school diploma has a negative slope. This indicates that the unemployment and percent of people with only a high school diploma are negatively correlated. The correlation coefficient must be calculated to find the strength of the correlation.

In [42]:

```
# Calcualte the Pearson standard correlation coefficient (r)
r = df_data_by_state["% HS Diploma, 2000"].corr(df_data_by_state["Unemploymen
```

In [43]:

```
print("The correlation coefficient comparing High School Diploma 2000 (%) wit
```

```
The correlation coefficient comparing High School Diploma 2000 (%) with Unempl
oyment 2000 (%):  -0.07256875257203949
```

Conclusion: As shown in the previous plot, there is a slightly negative correlation between unemployment and percent of people with only than a high school diploma. The correlation coefficient was calculated to be approximately -0.0726. This is lower than -0.7, so there is a negative but weak correlation between unemployment rate and percent of people with only a high school diploma.

In [44]:

```
# Examine if a correlation exists between unemployment and having a bachelor'

# Use the df_data_by_state DataFrame to to examine the correlation between Un
# the Percent of adults with a bachelor's degree or higher 2000
df_data_by_state[['State_x', 'Unemployment Rate 2000',"% >= Bachelors, 2000"]
```

Out[44]:

	State_x	Unemployment Rate 2000	% >= Bachelors, 2000
0	AK	6.192708	25.270935
1	AL	4.631517	19.034703
2	AR	4.179000	16.660582
3	AZ	3.967340	23.596865
4	CA	4.905245	26.620933

In [45]:
```
# Set the size of the boxplot to 15 by 12
sns.set(rc={'figure.figsize':(10,5)})

# Create a linear regression plot to visualize Unemployment Rate 2000 and Per
corr_p1 = sns.regplot(x="% >= Bachelors, 2000", y="Unemployment Rate 2000", d

# Set the X-axis label
corr_p1.set(xlabel="Percent of Adults With a Bachelor's Degree or Higher, 200

# Change the boxplot title
corr_p1.set_title("Linear Regression Plot Showing % of adults with a bachelor
```

Out[45]:
```
Text(0.5, 1.0, "Linear Regression Plot Showing % of adults with a bachelor's d
egree or higher, 2000 vs % Unemployment 2000 ")
```

Conclusion: The linear regression plot of Unemployment and percent of people with a bachelor's degree or higher has a negative slope. This indicates that the unemployment rate and percent of people with a bachelor's degree are negatively correlated. The correlation coefficient must be calculated to find the strength of the correlation.

In [46]:
```
# Calcualte the Pearson standard correlation coefficient (r)
r = df_data_by_state["% >= Bachelors, 2000"].corr(df_data_by_state["Unemploym
```

In [47]:
```
print("The correlation coefficient comparing % Bachelor's Degree or Higher wi
```

```
The correlation coefficient comparing % Bachelor's Degree or Higher with Unemp
loyment 2000 (%):  -0.3809390258255998
```

Conclusion: As shown in the previous plot, there is a negative correlation between unemployment and percent of people with a bachelor's degree or better. The correlation coefficient was calculated to be approximately -0.38. This is closer to 0 than -0.7, so there is a negative but weak correlation between unemployment rate and percent of people with a bachelor's degree or higher.

Hypothesis Testing

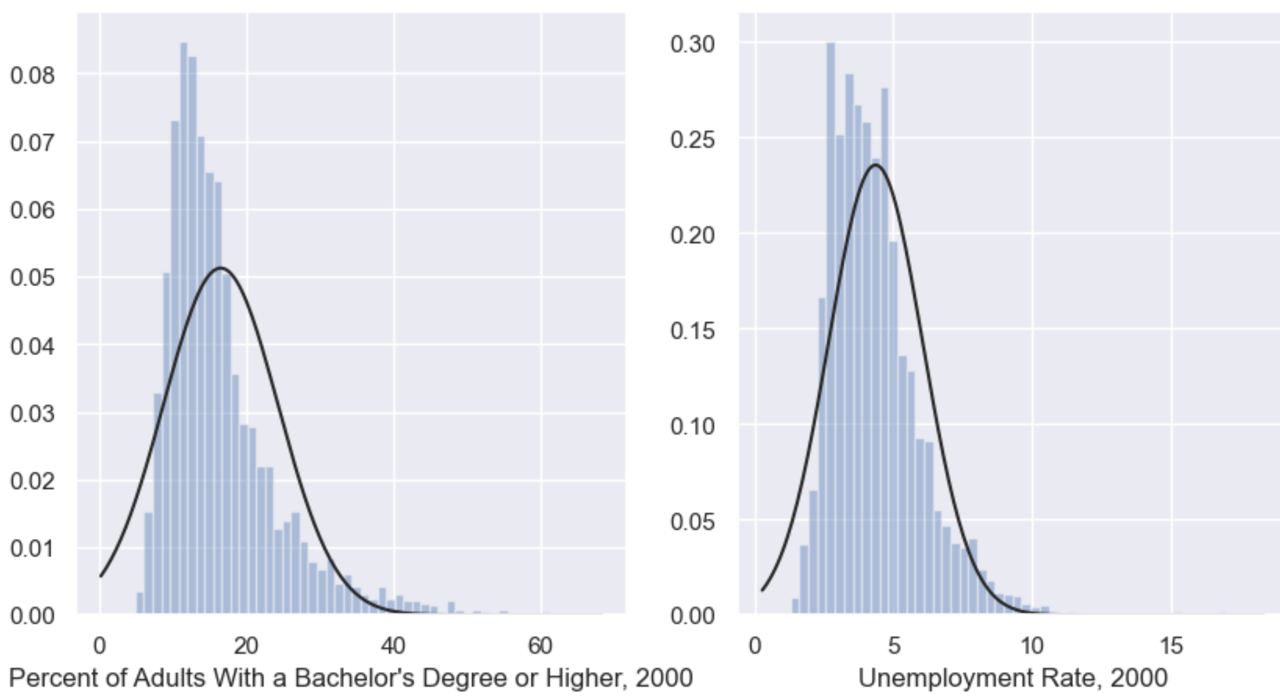
Normal Test

In [48]:

```
# Prepare two subplots to show the disbtribution of data for "Percent of adul
# and "Unemployment rate 2000"
fig, ax = plt.subplots(1,2)

# Create a distplot for "Percent of adults with a bachelor's degree or higher
sns.distplot(df_merged["% >= Bachelors, 2000"],fit=stats.norm,kde=False, ax=a
# Set the X-axis label
ax[0].set(xlabel="Percent of Adults With a Bachelor's Degree or Higher, 2000"

# Create a distplot for "Unemployment rate 2000"
sns.distplot(df_merged["Unemployment rate 2000"],fit=stats.norm,kde=False, ax
ax[1].set(xlabel="Unemployment Rate, 2000")
```

Out[48]:

```
[Text(0.5, 0, 'Unemployment Rate, 2000')]
```



Conclusion: The data "Percent of adults with a bachelor's degree or higher, 2000" and "Unemployment rate 2000" visually appear to be close to normally distributed, but the normal test should be run to verify.

In [49]:
```
# Use stats.normaltest() to see if the data for "Percent of adults with a bac
# distibuted.
stats.normaltest(df_merged["% >= Bachelors, 2000"])
```

Out[49]:
```
NormaltestResult(statistic=1103.6947814470946, pvalue=2.1663237251888e-240)
```

In [50]:
```
# Use stats.normaltest() to see if the data for "Unemployment rate 2000" if n
stats.normaltest(df_merged["Unemployment rate 2000"])
```

Out[50]:
```
NormaltestResult(statistic=1204.350632751332, pvalue=3.010114111302691e-262)
```

Conclusion: The 'statistic' returns s^2 + k^2; s is the z-score from the skew test, while k is the z-score from the kurtosis test. The p-value is the two-sided chi-squared probability of the hypothesis test. The null hypothesis (H_0) is "The data comes from a normal distribution" while the alternate hypothesis (H_a) is "The data does not come from a normal distribution."

H_0: "The data comes from a normal distribution"
H_a: "The data does not come from a normal distribution."

Since $p < 0.05$ for both columns of data, we reject the null hypothesis (H_0) that the data comes from a normal distribution. However, because there are so many data points (about 3,000 rows of data), the Central Limit Theorem can be applied and the data can be treated as a normal distribution.

Z-test

In [51]:
```
# Utilize the Z-test to test the mean of the distribution for "Percent of adu
# The Z-test can be used because the popultion standard deviation is known an

# H0: The average percentage of adults with a bachelor's degree or higher in
#     is 17%

# H1: The average percentage of adults with a bachelor's degree or higher in
#     is less than 17%

test_stat, pval = ztest(df_merged["% >= Bachelors, 2000"], value=17, alternat
print(f"Test statistic: {test_stat}, P-value: {pval}")
```

```
Test statistic: -3.4290387934866184, P-value: 0.0003028614711704304
```

Conclusion: The Z-test is a statistical test on data that can be treated as normally distributed. It produces a test statistic on how the mean of the data test against a null hypothesis. The Z-test was selected over the Student's T-test because the critical values used for the T-Test are defined by the sample size and is less convenient.

The null hypothesis under test (H_0) states "The average percentage of adults with a bachelor's degree or higher in the year 2000 for all counties in the USA is 17%", while the alternate hypothesis (H_a) states "The average percentage of adults with a bachelor's degree or higher in the year 2000 for all counties in the USA is less than 17%."

H_0: "The average percentage of adults with a bachelor's degree or higher in the year 2000 for all counties in the USA is 17%".

H_a: "The average percentage of adults with a bachelor's degree or higher in the year 2000 for all counties in the USA is less than 17%."

Because the P-value of the Z-test is less than 0.05 (0.0003028614711704304), there is evidence to reject the null hypothesis. This means that there is not enough evidence to support the claim that the average percentage of adults with a bachelor's degree or higher in the year 2000 for all counties in the USA is 17%.

Correlation Test

In [52]:

```
# the pandas.DataFrame.corr() method returns the correlation between the colu
# Examine the correlation between "Unemployment rate 2000" and "Percent of ad

df_merged["Unemployment rate 2000"].corr(df_merged["% < HS Diploma, 2000"], m
```

Out[52]:

```
0.48525791224159837
```

In [53]:
```
# The SciPy pearsonr() method returns a tuple with two values: the correlatio
# the probability of two unrelated processes producing that correlation coeff

# The Correlation Test examines two series in the DataFrame to see how they a
# by the Correlation Test provides insight into the evidence of there being a

# P-value < 0.001: The evidence for correlation is strong
# P-value < 0.05: The evidence for correlation is moderate
# P-value < 0.1: The evidece for correlation is weak
# P-value > 0.1: There is no evidence of correlation

# H0: There is no correlation between "Unemployment rate 2000" and "Percent o

# H1: There is a correlation between "Unemployment rate 2000" and "Percent of

corr, p = pearsonr(df_merged["Unemployment rate 2000"], df_merged["% < HS Dip


print(f"Correlation coefficient: {corr}, P-value: {p}")
```

```
Correlation coefficient: 0.4852579122415984, P-value: 7.097382052376354e-184
```

Conclusion: The correlation coefficient for the pandas.DataFrame.corr() method and the scipy.stats.pearsonr() method both returned a value of 0.485, which is a moderate positive correlation. The P-value returned from scipy.stats.pearsonr() is a small value close to 0. When p > 0.05, the data sets being studied are probably independent.

H_0: There is no correlation between "Unemployment rate 2000" and "Percent of adults with less than a high school diploma, 2000"

H_a: There is a correlation between "Unemployment rate 2000" and "Percent of adults with less than a high school diploma, 2000"

There is enough evidence to reject the null hypothesis (H_0) that there is no correlation between "Unemployment rate 2000" and "Percent of adults with less than a high school diploma, 2000." Both the correlation coefficient and P-value indicate that the data is correlated.

Chi-squared Test

In [54]:
```
# The Chi-squared Test compares an expected value with an observed value of c
# be used to evaulate a null hypothesis (H0).

# H0: "Percent of adults with less than a high school diploma, 2000" and "Une

# H1: "Percent of adults with less than a high school diploma, 2000" and "Une
```

In [55]:
```
# Build a table to hold the data. The first element in this example is treate
# while the second value is the observed value.

table = df_merged["% < HS Diploma, 2000"], df_merged["Unemployment rate 2000"

# The scipy.stats Chi-square Test returns a tuple with the following values:
# stat: The test statistic
# p: The P-value of the test
# dof: The degrees of freedom of the data, which is the number of categories
#      The Chi-squared test is used with categorical data, and in this examol
# expected: The expected frequencies calculated from the margin totals of the
stat, p, dof, expected = chi2_contingency(table)

# Print out the returned values
print('Test statistic: ', stat)
print('P-value: ', p)
print('Degrees of freedom (dof):', dof)
print('expected values:', expected)

# Set the probability at 95%
prob = 0.95

# Calculate the critical value based on the desired probability and degrees o
# (this is equivalent to using a Chi-squared table lookup to find the critica
critical_value = chi2.ppf(prob, dof)

# Print the probability, critical value
print('Probability: ', prob)
print('Critical value: ', critical_value)

# Evaluate the test statistic
print('Evaluate the test statistic:')
if abs(stat) >= critical_value:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')

# Evaluate the P-value
print('Evaluate the P-value:')
alpha = 1.0 - prob
print('Significance: ', alpha)
if p <= alpha:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')
```

```
Test statistic:  1713.9652606813252
P-value:  1.0
Degrees of freedom (dof): 3116
expected values: [[35.37540301 27.74705781 31.93845627 ... 16.346454   18.1906
6932
  13.66395898]
 [ 6.82459699  5.35294219  6.16154373 ...  3.153546    3.50933068
   2.63604102]]
Probability:  0.95
Critical value:  3246.976756354243
Evaluate the test statistic:
Independent (fail to reject H0)
Evaluate the P-value:
Significance:  0.050000000000000044
Independent (fail to reject H0)
```

Conclusion: The Chi-squared test produced a test statistic and P-value. If the P-value is less than or equal to the the significance level (0.05), then there is evidence to reject the null hypothesis ($\boldsymbol{H_0}$).

$\boldsymbol{H_0}$: "Percent of adults with less than a high school diploma, 2000" and "Unemployment rate 2000" are independent."

$\boldsymbol{H_a}$: "Percent of adults with less than a high school diploma, 2000" and "Unemployment rate 2000" are dependent."

The test statistic (1713.96) was less than the critical value (3246.97) and the P-value is 1, so we fail to reject the null hypothesis ($\boldsymbol{H_0}$) that "Percent of adults with less than a high school diploma, 2000" and "Unemployment rate 2000" are independent.

ANOVA: Analysis of Variance

Comment: ANOVA (Analysis of Variance) is a technique used to compare the means of more than two sets of data. The means of the data sets do not have to be equal; ANOVA measures how likely the means represent data from the same overall population. It is still possible to use pair-wise statistical tests on data to compare the sample means, but this causes the errors to compound for each test. Put another way, ANOVA is a variability ratio where the variance between means is divided by the variance within the means. If the variance between means is large comparted to a smaller variance within means, the null hypothesis ($\boldsymbol{H_0}$) is rejected. If the variance between means is similar to the variance within means, or if the variance between means is small and the variance within means is large, then we fail to reject the null hypothesis ($\boldsymbol{H_0}$).

In [56]:

```
# Create a DataFrame to use with the ANOVA test using "Percent of adults with
# and area ("City/Suburb/Town/Rural 2013")
df_anova = df_merged[["City/Suburb/Town/Rural 2013","% < HS Diploma, 2000"]]

# Group the annova dataframe by area type (City/Suburb/Town/Rural 2013)
df_anova_groupby_area = df_anova.groupby(['City/Suburb/Town/Rural 2013'])
```

In [57]:

```
# Perform the ANOVA calculation using the SciPy f_oneway method to calculate
# ANOVA test and create the null hypothesis

# H0: There is no difference in the mean percentage of adults with less than
#     down by City, Suburb, Rural, and Town areas.

# Ha: There is a difference in the mean percentage of adults with less than a
#     down by City, Suburb, Rural, and Town areas.

anova_result_1 = stats.f_oneway(df_anova_groupby_area.get_group("City")["% < 
                                df_anova_groupby_area.get_group("Suburb")["% 
                                df_anova_groupby_area.get_group("Rural")["% <
                                df_anova_groupby_area.get_group("Town")["% < 

print( "ANOVA results: F=",anova_result_1)
```

```
ANOVA results: F= F_onewayResult(statistic=60.470231199169056, pvalue=5.416271
238168328e-38)
```

Conclusion: If the variance between means is large comparted to a smaller variance within means, the null hypothesis (H0) is rejected.

H_0: There is no difference in the mean percentage of adults with less than a high school diploma when broken down by City, Suburb, Rural, and Town areas.

H_a: There is a difference in the mean percentage of adults with less than a high school diploma when broken down by City, Suburb, Rural, and Town areas.

The result of the one-way ANOVA calculation produced a F-statistic significantly larger than 1 and a P-value much less than 0.05; this is statistically significant and is evidence to reject the null hypothesis (***H_0***) that there is no difference in the mean percentage of adults with less than a high school diploma when broken down by City, Suburb, Rural, and Town areas.

In [ ]:

In [ ]:

Question 4: Which years have the highest and lowest unemployment rate over the course of 21 years? (Fiona)

In [128…

```
# Create a function to return the unemployment rate for each year, name it un
def unemplyRate(year):
    unemployment_rate= df_merged['Unemployed {}'.format(year)].sum()/df_merge
    return unemployment_rate

# Create arrays for year and unemployment rate separately.
array_year=np.arange(2000, 2021)
array_unemplyRate=[unemplyRate(2000),unemplyRate(2001),unemplyRate(2002),unem
                  unemplyRate(2005),unemplyRate(2006),unemplyRate(2007),unemp
                  unemplyRate(2010),unemplyRate(2011),unemplyRate(2012),unemp
                   unemplyRate(2015),unemplyRate(2016),unemplyRate(2017),unem

# Convert a dictionary with column names year and unemployment rate,
# and the two arrays containing their values into a DataFrame.
df_yearly=pd.DataFrame({'Year': array_year, 'Unemployment Rate': array_unempl

# Display the DataFrame
df_yearly
```

Out[128...

	Year	Unemployment Rate
0	2000	3.987268
1	2001	4.729557
2	2002	5.786635
3	2003	5.982835
4	2004	5.524438
5	2005	5.096690
6	2006	4.627300
7	2007	4.622688
8	2008	5.790927
9	2009	9.259746
10	2010	9.643024
11	2011	8.956002
12	2012	8.073261
13	2013	7.376400
14	2014	6.163897
15	2015	5.276563
16	2016	4.863503
17	2017	4.349288
18	2018	3.888465
19	2019	3.662809
20	2020	8.047782

In [59]:

```
# Display the year which has the maxmium unemployment rate and minimum unempl
df_yearly_data.loc[df_yearly_data['Unemployment Rate']==df_yearly_data['Unemp
```

Out[59]:

	Year	Unemployment Rate
10	2010	9.643024

In [60]:

```
df_yearly_data.loc[df_yearly_data['Unemployment Rate']==df_yearly_data['Unemp
```

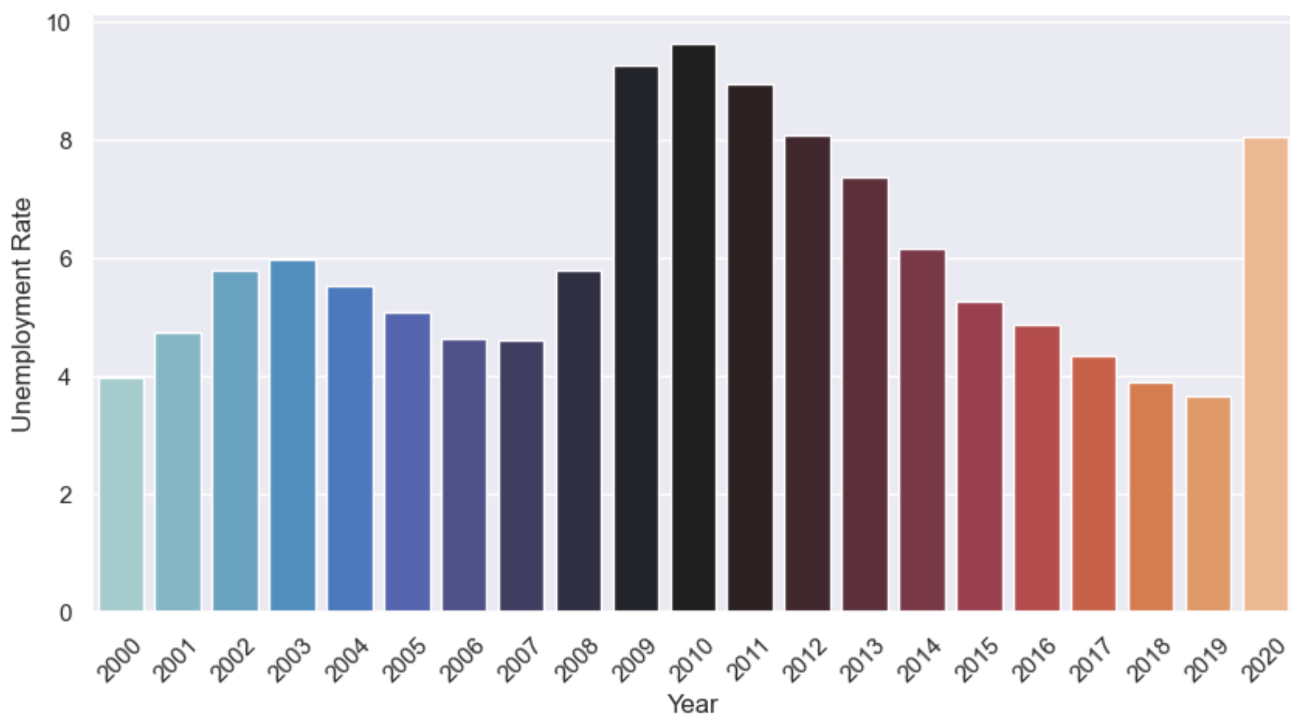
Out[60]:

	Year	Unemployment Rate
19	2019	3.662809

In [123…

```
# Use barplot to visualize the unemployment rate for each year.
fig, ax=plt.subplots(figsize=(10, 5))
sns.barplot(x='Year', y='Unemployment Rate', data=df_yearly_data, palette='ic
plt.xticks(rotation=45)
```

Out[123…
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20]),
 [Text(0, 0, '2000'),
  Text(1, 0, '2001'),
  Text(2, 0, '2002'),
  Text(3, 0, '2003'),
  Text(4, 0, '2004'),
  Text(5, 0, '2005'),
  Text(6, 0, '2006'),
  Text(7, 0, '2007'),
  Text(8, 0, '2008'),
  Text(9, 0, '2009'),
  Text(10, 0, '2010'),
  Text(11, 0, '2011'),
  Text(12, 0, '2012'),
  Text(13, 0, '2013'),
  Text(14, 0, '2014'),
  Text(15, 0, '2015'),
  Text(16, 0, '2016'),
  Text(17, 0, '2017'),
  Text(18, 0, '2018'),
  Text(19, 0, '2019'),
  Text(20, 0, '2020')])
```



Conclusion: From year 2000 to 2020, 2010 has the highest unemployment rate 9.64%, while 2019 has the lowest unemployment rate 3.66%. However, there was a great increase on unemployment rate from year 2008 to 2009 and 2019 to 2020.

Question 5: Which states contribute the most and the least for the unemployment change from year 2019 to 2020? (Fiona)

In [62]:
```
# create a dataframe df_sum which aggregates the sum of civilian labor force
df_sum=pd.pivot_table(df_merged, index='State_x', values=['Civilian labor for
                                                          'Unemployed 2019', '
```

In [63]:
```
# show the head of df_sum
df_sum.head()
```

Out[63]:

	State_x	Civilian labor force 2019	Civilian labor force 2020	Unemployed 2019	Unemployed 2020
0	AK	338294.0	332648.0	18177.0	25995.0
1	AL	2237287.0	2230132.0	67888.0	131065.0
2	AR	1365272.0	1354299.0	48109.0	81952.0
3	AZ	3529442.0	3561240.0	171507.0	281433.0
4	CA	19353742.0	18821176.0	803218.0	1908093.0

In [64]:
```
# define a function to return the unemployment rate for any year.
def df_rate(year):
    return df_sum['Unemployed {}'.format(year)]/df_sum['Civilian labor force
```

In [65]:
```
# add two columns to show the unemployment rate for each state for both year
df_sum['Unemployment rate 2019']=df_rate(2019)
df_sum['Unemployment rate 2020']=df_rate(2020)
```

In [66]:
```
# show the head of df_sum with only columns of unemployment rate, assign it t
df_rate=df_sum[['State_x', 'Unemployment rate 2019', 'Unemployment rate 2020'
df_rate.head()
```

Out[66]:

	State_x	Unemployment rate 2019	Unemployment rate 2020
0	AK	5.373137	7.814567
1	AL	3.034389	5.877006
2	AR	3.523767	6.051249
3	AZ	4.859323	7.902669
4	CA	4.150195	10.138012

In [67]:
```
# add another column to show the unemployment rate change
df_rate['% Unemployment rate change']=(df_rate['Unemployment rate 2020']-df_r
```

In [68]:

```
# show the head of df_rate
df_rate.head()
```

Out[68]:

	State_x	Unemployment rate 2019	Unemployment rate 2020	% Unemployment rate change
0	AK	5.373137	7.814567	45.437698
1	AL	3.034389	5.877006	93.680032
2	AR	3.523767	6.051249	71.726711
3	AZ	4.859323	7.902669	62.628994
4	CA	4.150195	10.138012	144.277967

In [69]:

```
# show 5 states that have the highest unemployment rate change
df_rate.sort_values(by='% Unemployment rate change', ascending=False).head(5)
```

Out[69]:

	State_x	Unemployment rate 2019	Unemployment rate 2020	% Unemployment rate change
10	HI	2.454310	11.631602	373.925476
32	NV	3.904095	12.833004	228.706267
18	MA	3.023465	8.861822	193.101503
30	NJ	3.425938	9.786199	185.650179
5	CO	2.659245	7.271170	173.429859

In [70]:

```
# show 5 states that have the lowest unemployment rate change
df_rate.sort_values(by='% Unemployment rate change', ascending=True).head(5)
```

Out[70]:

	State_x	Unemployment rate 2019	Unemployment rate 2020	% Unemployment rate change
28	NE	2.989404	4.230367	41.512080
0	AK	5.373137	7.814567	45.437698
24	MS	5.546770	8.084132	45.744861
40	SD	2.992517	4.643643	55.175173
49	WY	3.715453	5.842239	57.241624

Concluson: The sorted dataframe shows that all the states have positive unemployment rate changes, from 42% to 374%. The five states that have the highest unemployment rate change from 2019 to 2020 are Hawaii, Nevada, Massachusetts, New Jersey, Colorado, with changes of more than 170%. And the five states that have the lowest unemployment rate change are Nebraska, Alaska, Mississippi, South Dakota, and Wyoming, with changes of less than 60%.

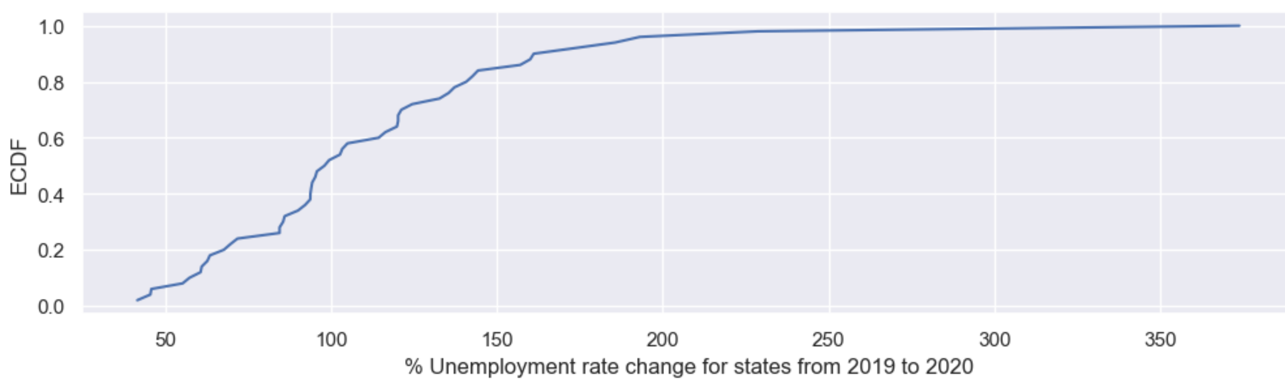
In [71]:

```
# Since so many states have unemployment rate change that are greater than 10
# let's examine the fraction of states that have unemployment rate equal to o
# We can use the ECDF which Paul already defined to plot the cumulative data.
# First, let's caculate ECDF values for the percent change of unemployment ra
x, y=ecdf(df_rate['% Unemployment rate change'])
```

In [72]:

```
# Plot ECDF values for the unemployment rate change from year 2019 to 2020.
# set figure size of the plot so it's easier to see the line trend
plt.rcParams['figure.figsize']=(12, 3)
# Label the x-axis and y-axis
plt.xlabel('% Unemployment rate change for states from 2019 to 2020')
plt.ylabel('ECDF')
# Plot the x and y ECDF values
plt.plot(x, y)
```

Out[72]: [<matplotlib.lines.Line2D at 0x7fddbe6ce6a0>]

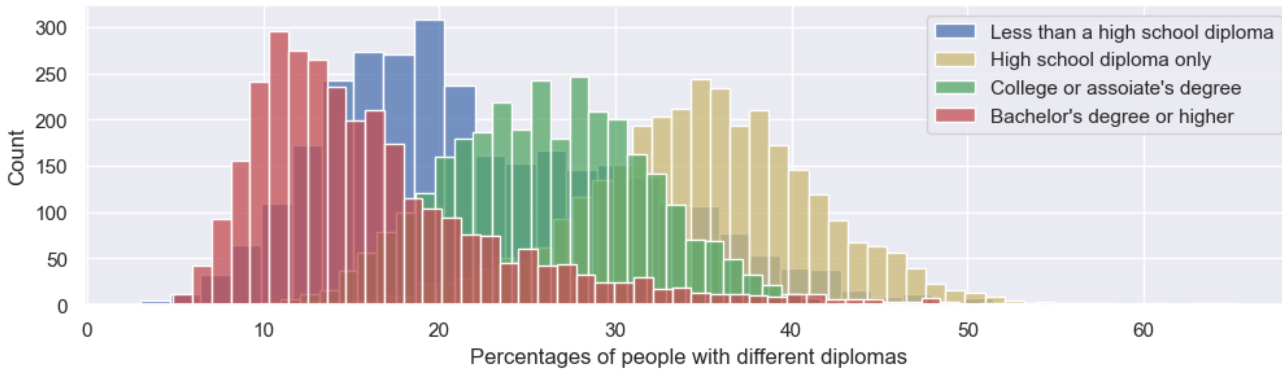


Conclusion: The ECDF plot shows that approximate 50% of the states have unemployment rate change equal to or less than 100%. This means that another 50% of the states have unemployment rate change larger than 100%.

Question 6: Is there a significant change regarding percentages of people completing different diplomas between year 2000 and year 2015-2019? (Fiona)

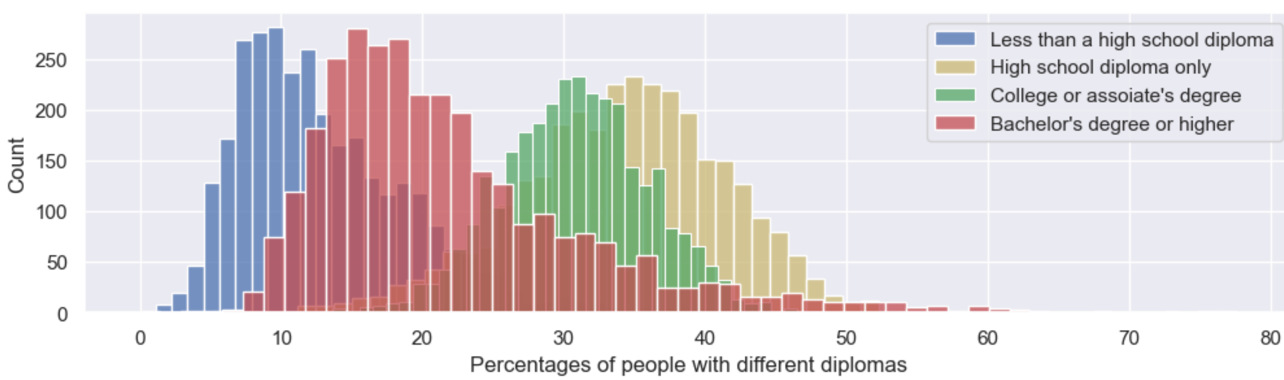
In [73]:
```
sns.histplot(df_merged["% < HS Diploma, 2000"], color='b')
sns.histplot(df_merged["% HS Diploma, 2000"], color='y')
sns.histplot(df_merged["% Some College, 2000"], color='g')
sns.histplot(df_merged["% >= Bachelors, 2000"], color='r')
plt.xlabel('Percentages of people with different diplomas')
plt.legend(['Less than a high school diploma', 'High school diploma only', 'C
            'Bachelor\'s degree or higher'], loc='upper right')
```

Out[73]: <matplotlib.legend.Legend at 0x7fddc41df1f0>



In [74]:
```
sns.histplot(df_merged["% < HS Diploma, 2015-19"], color='b')
sns.histplot(df_merged["% HS Diploma, 2015-19"], color='y')
sns.histplot(df_merged["% Some College, 2015-19"], color='g')
sns.histplot(df_merged["% >= Bachelors, 2015-19"], color='r')
plt.xlabel('Percentages of people with different diplomas')
plt.legend(['Less than a high school diploma', 'High school diploma only', 'C
            'Bachelor\'s degree or higher'], loc='upper right')
```

Out[74]: <matplotlib.legend.Legend at 0x7fddc4521910>



Conclusion: There is a significant change in the percentages of people completing different diplomas between year 2000 and year 2015-2019, with more people having bachelor's degree or higher, and fewer people having less than a high school diploma. However, the percentages of people with high school diploma only and college or associate's degree don't have too much difference.

Hypothesis Testing

Normal Test

Test if the unemployment rate of year 2020 is normally distributed

- H0: Distribution is normal.
- H1: Distribution is not normal.

In [75]:

```
# In order to check if a data is normally distributed, we use the built-in fu
# Normaltest returns a 2-tuple of the chi-squared statistic, and the associat
from scipy.stats import normaltest
normaltest(df_merged['Unemployment rate 2020'])
```

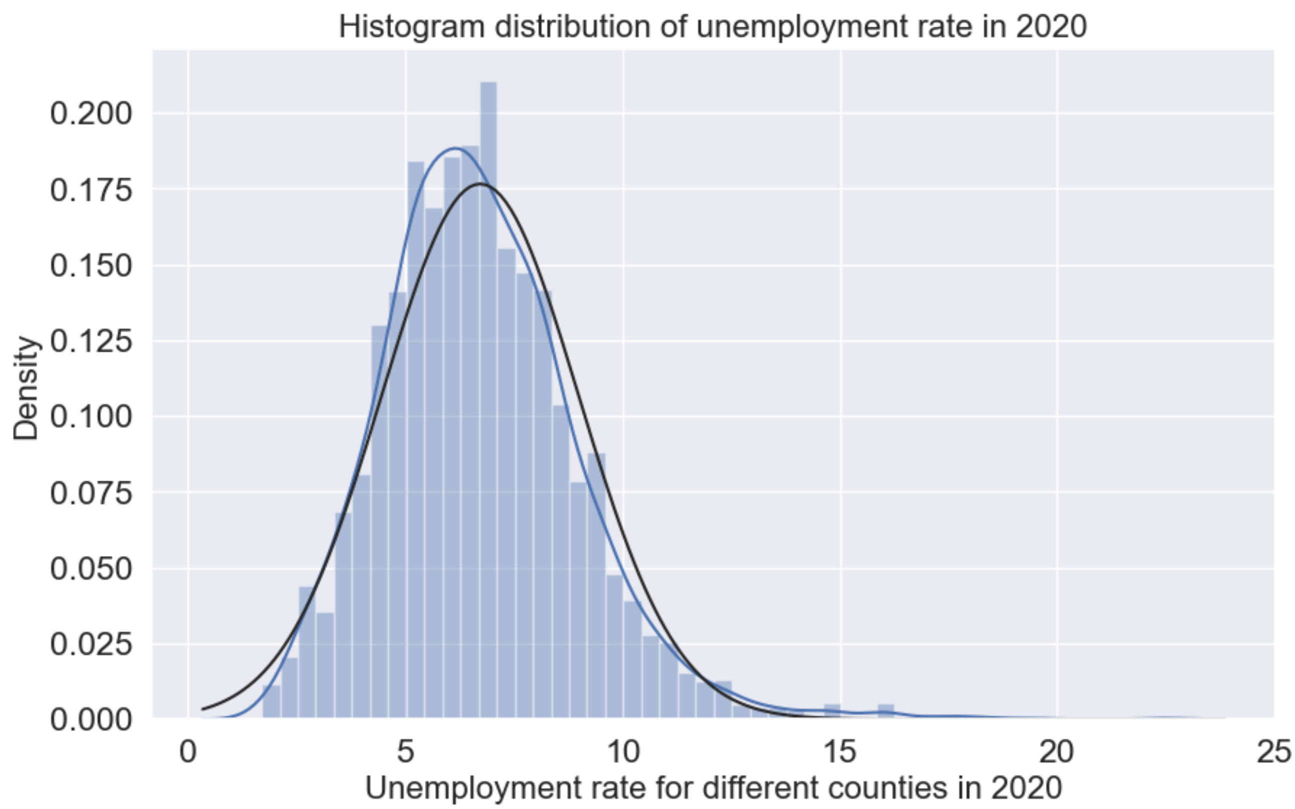
Out[75]:

```
NormaltestResult(statistic=491.7981758483595, pvalue=1.6120667093233896e-107)
```

Conclusion: Since p-value<0.05, the unemployment rate of 2020 does not follow a normal distribution (Reject H0).

In [76]:

```
# We can also visualize the data through distplot to check our result.

# Set the figure width of 10 and height of 6
plt.rcParams["figure.figsize"] = [10,6]
sns.set_style("darkgrid")
#set context , font scale and font size
sns.set_context("notebook", font_scale=1.5, rc={"font.size":16,"axes.titlesiz
# the fit will impose a normal curve to the histogram
# we set kde to false because by default it uses the kde
sns.distplot(df_merged['Unemployment rate 2020'],fit=stats.norm,kde=True)
# add title, xlabel to the plot
plt.title('Histogram distribution of unemployment rate in 2020')
plt.xlabel('Unemployment rate for different counties in 2020')
plt.show()
```



Conclusion: The kde is skewed left, so we reject the null hypothesis that the unemployment rate for 2020 follows a normal distribution.

In [ ]:

```
```

Z-test

Test if the mean of the percent of adults with high school diploma only in 2015-19 is 35 against the alternative that it is not $\begin{align}

& {{H}*{0}}*:|,|*mu* =|,*{{|mu* }{0}} \ & {{H}*{1}}*:|,|*mu* |*ne* *{{|mu* }{0}} \ \end{align}$

In [77]:

```
df_merged['% HS Diploma, 2015-19'].mean()
```

Out[77]: 34.17847289059988

In [78]:

```
(test_statistic, p_value) = ztest(df_merged['% HS Diploma, 2015-19'], value=3
```

In [79]:
```
print("The test statistic is: ", test_statistic)
print("The p-value is: ", p_value)
```

```
The test statistic is:  -6.345513987373989
The p-value is:  2.216840537231483e-10
```

Conclusion: p-value is less than 0.05, so at alpha =0.05 level of significance we can reject the null hypothesis. This means that there is not enough evidence to support the claim that the average percentage of adults with high school diploma only in the year 2015-19 for all counties in the USA is 35%.

Testing the hypothesis that the mean is 35 against the alternative that it is GREATER

$$H_0 : \mu <= \mu_0 \quad (1)$$
$$H_1 : \mu > \mu_0 \quad (2)$$

In [80]:
```
(test_statistic, p_value) = ztest(df_merged['% HS Diploma, 2015-19'], value=3
```

In [81]:
```
print("The test statistic is: ", test_statistic)
print("The p-value is: ", p_value)
```

```
The test statistic is:  -6.345513987373989
The p-value is:  0.999999999889158
```

Conclusion: p-value is very close to 1, so at alpha =0.05 level of significance we can not reject the null hypothesis. This means that there's stong evidence to support that the average percentage of adults with high school diploma only in the year 2015-19 for all counties in the USA is less than 35%.

Testing the hypothesis that the mean is 35 against the alternative that it is SMALLER

$$H_0 : \mu >= \mu_0 \quad (3)$$
$$H_1 : \mu < \mu_0 \quad (4)$$

In [82]:
```
(test_statistic, p_value) = ztest(df_merged['% HS Diploma, 2015-19'], value=3
```

In [83]:
```
print("The test statistic is: ", test_statistic)
print("The p-value is: ", p_value)
```

```
The test statistic is:  -6.345513987373989
The p-value is:  1.1084202686157414e-10
```

Conclusion:p-value is less than 0.05, so at alpha =0.05 level of significance we can reject the null hypothesis. This means that there is not enough evidence to support the claim that the average percentage of adults with high school diploma only in the year 2015-19 for all counties in the USA is larger than 35%. From all the hypothesis testings we can conclude that the z-test indicates that the average percentage of adults with high school diploma only in the year 2015-19 is less than 35%.

Correlation test

Testing the correlation between the percent of adults with less than a high school diploma and the unemployment rate in year 2015-2019.

- H0: the two samples are independent.
- H1: there is a dependency between the samples.

In [84]:

```
# Calculate the average unemployment rate for all counties from 2015-2019.
df_merged['Unemployed 2015-2019']=df_merged['Unemployed 2015']+df_merged['Une
+df_merged['Unemployed 2018']+df_merged['Unemployed 2019']
df_merged['Civilian labor force 2015-2019']=df_merged['Civilian labor force 2
+df_merged['Civilian labor force 2017']+df_merged['Civilian labor force 2018'
df_merged['Unemployment rate 2015-19']= df_merged['Unemployed 2015-2019']/df_
# Show the column Unemploument rate 2015-2019
df_merged['Unemployment rate 2015-19']
```

Out[84]:

```
0       8.817506
1       7.523158
2       7.707645
3       7.974111
4       7.294081
          ...
3112    6.110381
3113    9.241071
3114    8.381459
3115    6.821677
3116    4.519774
Name: Unemployment rate 2015-19, Length: 3117, dtype: float64
```

In [85]:

```
stat, p = pearsonr(df_merged['Unemployment rate 2015-19'], df_merged["% < HS 
```

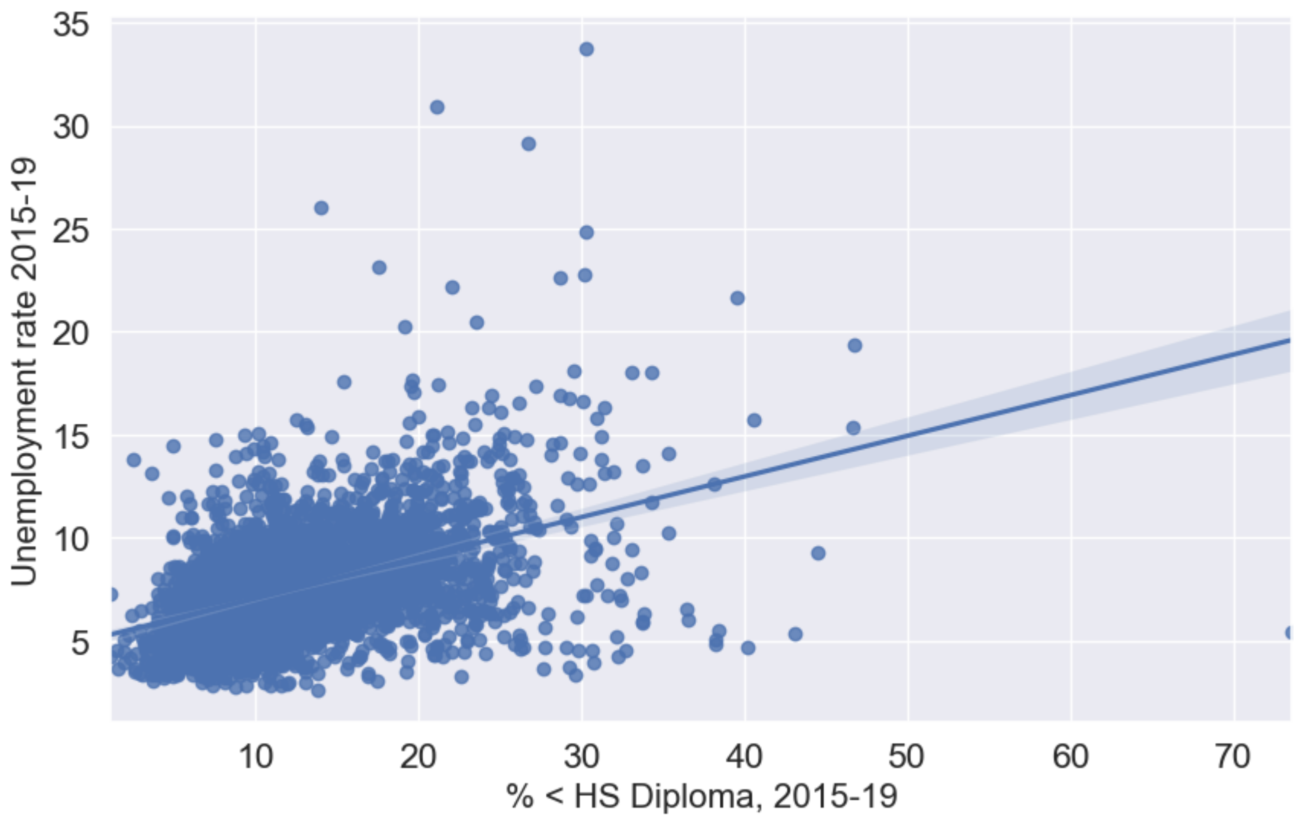
In [86]:

```
print(stat, p)
if p > 0.05:
  print('Probably independent')
else:
 print('Probably dependent')
```

```
0.45652019698009155 2.5633905632932428e-160
Probably dependent
```

In [87]:
```
# Let's visualize the data to check if it shows the same result
sns.regplot(x="% < HS Diploma, 2015-19", y='Unemployment rate 2015-19',
            data=df_merged)
```

Out[87]: <AxesSubplot:xlabel='% < HS Diploma, 2015-19', ylabel='Unemployment rate 2015-19'>



Conclusion: P-value<0.05, so the unemployment rate and percent of adults with less than a high school diploma in year 2015-2019 are probably dependent. Since the pearson correlation coeffient is 0.46, we can say that they are positively correlated, and the correlation between them are moderate strong.

Chi-squared Test

Testing if the unemployment rate matches the percent of adults with a bachelor's degree or higher in year 2015-2019

- H0: the unemployment rate and the percent of adults with a bachelor's degree or higher are independent.
- H1: there is a dependency between the samples.

In [88]:
```
# contingency table
table= df_merged['Unemployment rate 2015-19'], df_merged["% >= Bachelors, 201
print(table)
stat, p, dof, expected = chi2_contingency(table)
print('dof=%d' % dof)
print(expected)
# interpret test-statistic
prob = 0.95
critical = chi2.ppf(prob, dof)
print('probability=%.3f, critical=%.3f, stat=%.3f' % (prob, critical, stat))
if abs(stat) >= critical:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')
# interpret p-value
alpha = 1.0 - prob
print('significance={}, p={}'.format(alpha, p))
if p <= alpha:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')
```

```
(0       8.817506
1       7.523158
2       7.707645
3       7.974111
4       7.294081
          ...
3112    6.110381
3113    9.241071
3114    8.381459
3115    6.821677
3116    4.519774
Name: Unemployment rate 2015-19, Length: 3117, dtype: float64, 0       10.4
1       13.1
2       12.7
3       33.4
4       16.1
        ...
3112    16.5
3113    24.2
3114    29.1
3115    19.0
3116    18.0
Name: % >= Bachelors, 2015-19, Length: 3117, dtype: float64)
dof=3116
[[ 4.97176678  5.33542326  5.27966784 ...  9.69683952  6.68033385
   5.82609744]
 [14.24573951 15.28773438 15.12797685 ... 27.78461983 19.14134353
  16.69367657]]
probability=0.950, critical=3246.977, stat=6891.142
Dependent (reject H0)
significance=0.050000000000000044, p=1.5561516234184176e-285
Dependent (reject H0)
```

Conclusion: P-value<0.05, reject the null hypothesis, so the unemployment rate and percent of adults with a bachelor's degree or higher in year 2015-19 are dependent.

ANOVA: Analysis of Variance

Testing whether median household income differs based on areas(city/suburb/town/rural)

- H0: the median household income in different areas has no significant difference
- H1: at least one area has the median houshold income that differs significantly from others

In [89]:
```
#extract only the columns of interest weight and group
df_anova = df_merged[['City/Suburb/Town/Rural 2013','Median Household Income 
#display the dataframe head
df_anova.head()
```

Out[89]:

	City/Suburb/Town/Rural 2013	Median Household Income 2019
0	City	47918.0
1	City	52902.0
2	City	49692.0
3	City	54127.0
4	City	65403.0

In [90]:
```
#find the unique group values assign it to grps
grps = pd.unique(df_anova['City/Suburb/Town/Rural 2013'].values)
grps
```

Out[90]:
```
array(['City', 'Suburb', 'Rural', 'Town'], dtype=object)
```

In [91]:
```
# group Median Houshold Income by areas
dict_anova={grp: df_anova['Median Household Income 2019'][df_anova['City/Subu
dict_anova
```

Out[91]:
```
{'City': 0       47918.0
 1       52902.0
 2       49692.0
 3       54127.0
 4       65403.0
          ...
 1148    61624.0
 1149    62108.0
 1150    59643.0
 1151    69613.0
 1152    66104.0
 Name: Median Household Income 2019, Length: 1153, dtype: float64,
 'Suburb': 1153    42024.0
 1154    50897.0
 1155    47719.0
 1156    45980.0
 1157    39990.0
          ...
 1667    58248.0
 1668    63752.0
 1669    58982.0
 1670    57325.0
 1671    52216.0
 Name: Median Household Income 2019, Length: 371, dtype: float64,
 'Rural': 1283    51276.0
 1284    29572.0
 1285    42922.0
 1286    40827.0
 1287    45273.0
          ...
 3112    48761.0
 3113    53908.0
 3114    55576.0
 3115    53018.0
 3116    48513.0
 Name: Median Household Income 2019, Length: 980, dtype: float64,
 'Town': 1672    35972.0
 1673    31906.0
 1674    39944.0
 1675    45982.0
 1676    44836.0
          ...
 2441    57953.0
 2442    64030.0
 2443    80639.0
 2444    98837.0
 2445    70756.0
 Name: Median Household Income 2019, Length: 613, dtype: float64}
```

In [92]:
```python
#find the statistic F and P value calling the stats.f_oneway method from scip
F, p = stats.f_oneway(dict_anova['City'], dict_anova['Suburb'], dict_anova['T
```

In [93]:
```
#print the p-value
print("p-value for significance is: ", p)
```

```
p-value for significance is:  3.944965924079782e-150
```

Conclusion: p-value<0.05, reject the null hypothesis. At least one area does not have the same mean.

Separately: city, suburb and town

In [94]:
```
f_val, p_val = stats.f_oneway(dict_anova['City'], dict_anova['Suburb'], dict_

print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 208.85695705451874 , P = 1.4478259314252543e-83
```

Separately: city, suburb and rural

In [95]:
```
f_val, p_val = stats.f_oneway(dict_anova['City'], dict_anova['Suburb'], dict_

print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 310.3855867034588 , P = 3.914159101923696e-121
```

Separately: city, town, and rural

In [96]:
```
f_val, p_val = stats.f_oneway(dict_anova['City'], dict_anova['Town'], dict_an
print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 362.58297379265696 , P = 1.904249005060578e-140
```

Separately: suburb, town and rural

In [97]:
```
f_val, p_val = stats.f_oneway(dict_anova['Suburb'], dict_anova['Town'], dict_
print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 11.51236925810701 , P = 1.0699554695614907e-05
```

Conclusion: Based on the separate tests, p_values among any three of the areas are less than 0.05, meaning at least one area among any three of the areas has different median houshold income mean.

In [98]:
```
# Since the p-value is much less in the last separate test,
# let's examine the last group and see if there's significant difference betw
```

Separate test: suburb and town

In [99]:

```
f_val, p_val = stats.f_oneway(dict_anova['Suburb'], dict_anova['Town'])
print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 9.22708142147127 , P = 0.0024476361838968327
```

Separate test: suburb and rural

In [100…

```
f_val, p_val = stats.f_oneway(dict_anova['Suburb'], dict_anova['Rural'])
print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 23.73476570516828 , P = 1.2364270478373443e-06
```

Separate test: town and rural

In [101…

```
f_val, p_val = stats.f_oneway(dict_anova['Town'], dict_anova['Rural'])
print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 2.8774182752758586 , P = 0.09002556319094654
```

Conclusion: When you examine any two areas among suburb, town and rural, the p-values are not that small, especially the one for town and rural. Since 0.05<p<0.1 in the last group, there is weak evidence against the null hypothesis.

In [ ]:

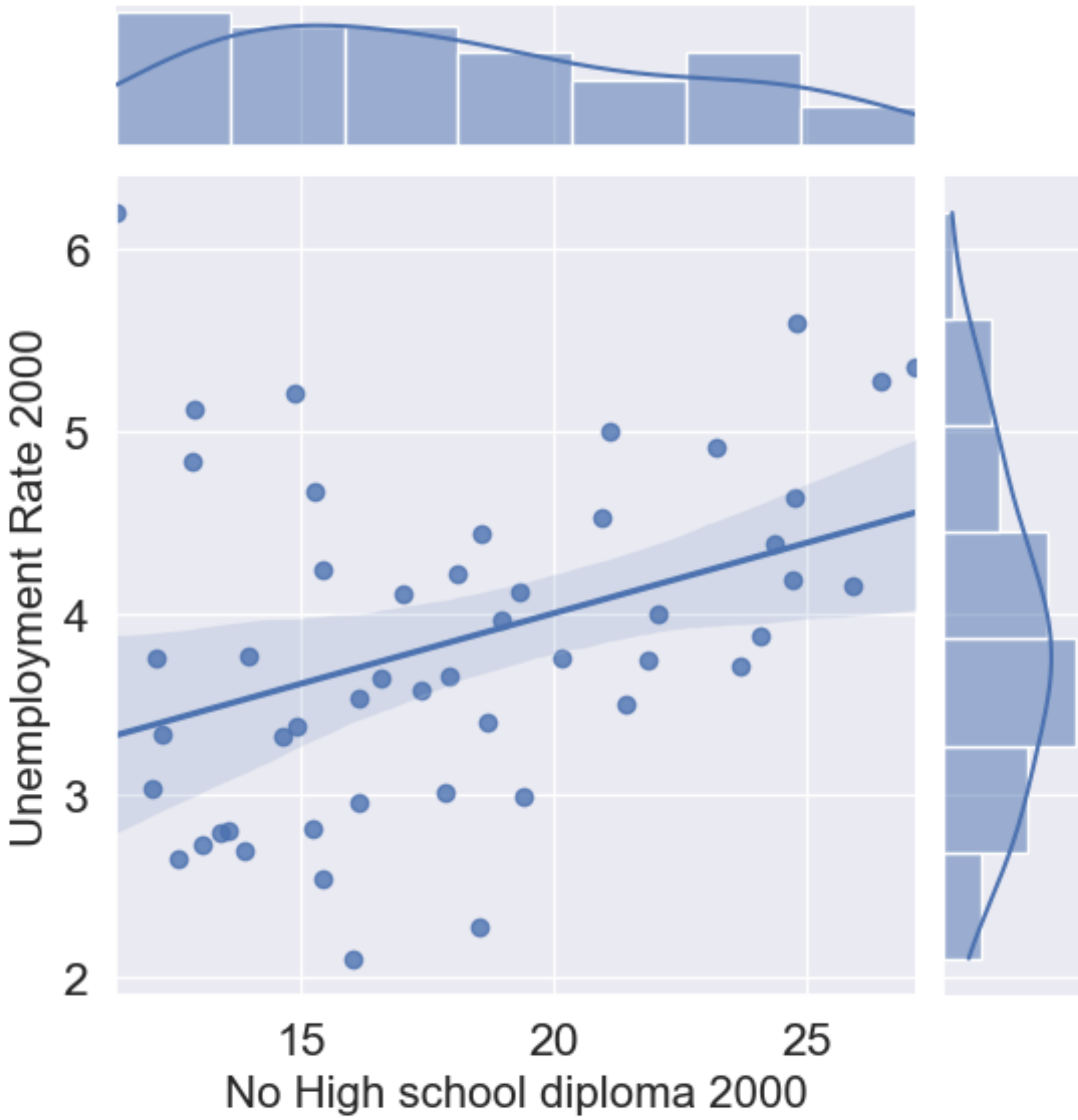
In [ ]:

Question 7: What is the correlation between the adults with less than a high school diploma and unemployment rate in the year 2000? (Ping)

In [102…

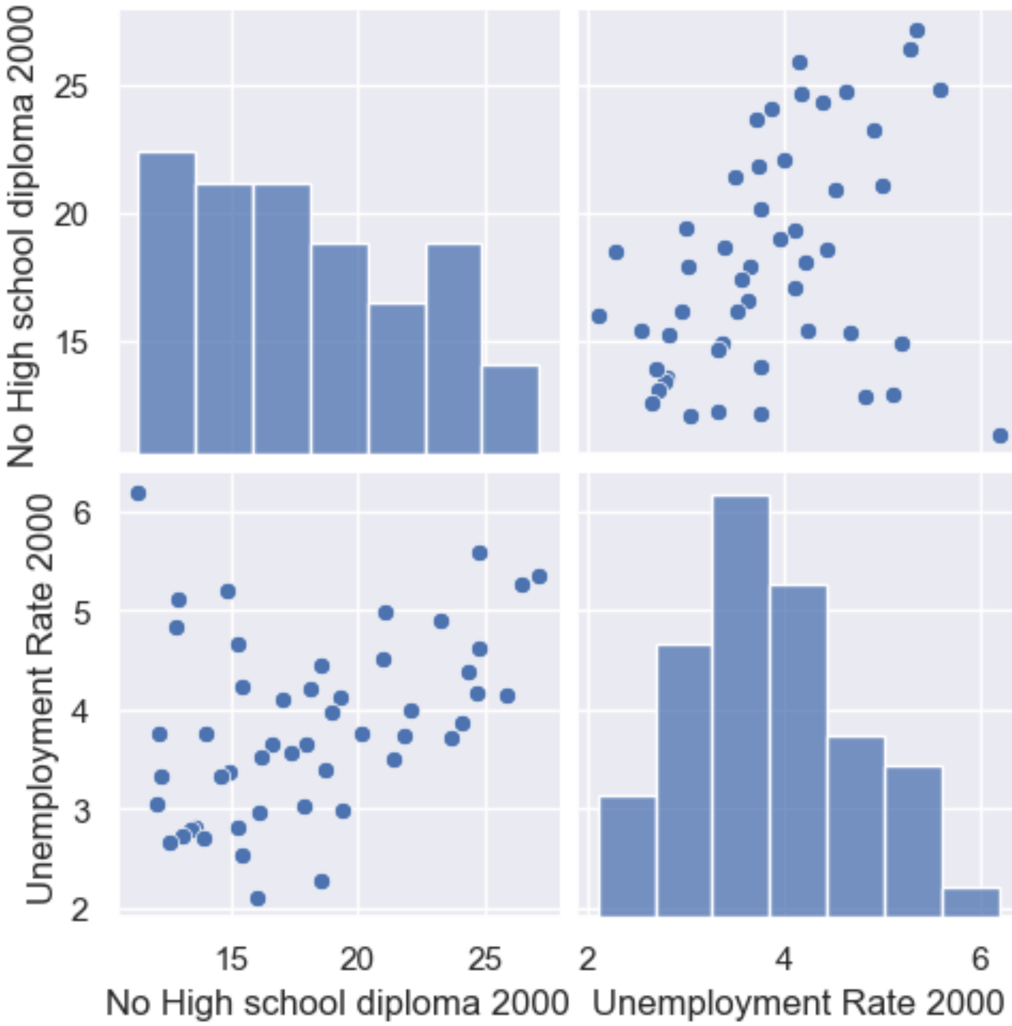
```
# PING JU
# Use Jointplot to show correlation between Unemployment Rate 2000 and Percen

df1 = df_data_by_state[['State_x', '% < HS Diploma, 2000','Unemployment Rate 

# Rename for X Axis
df1 = df1.rename(columns = {"% < HS Diploma, 2000":"No High school diploma 20

#Print Jointplot
sns.jointplot(x='No High school diploma 2000',y='Unemployment Rate 2000',data
```

Out[102… <seaborn.axisgrid.JointGrid at 0x7fddc3f0a220>



In [103…

```
#Define function
def draw_pairplot(df):
    sns.pairplot(df)

# Set the font size
sns.set(font_scale = 1)

#Draw Plot
draw_pairplot(df1)
```



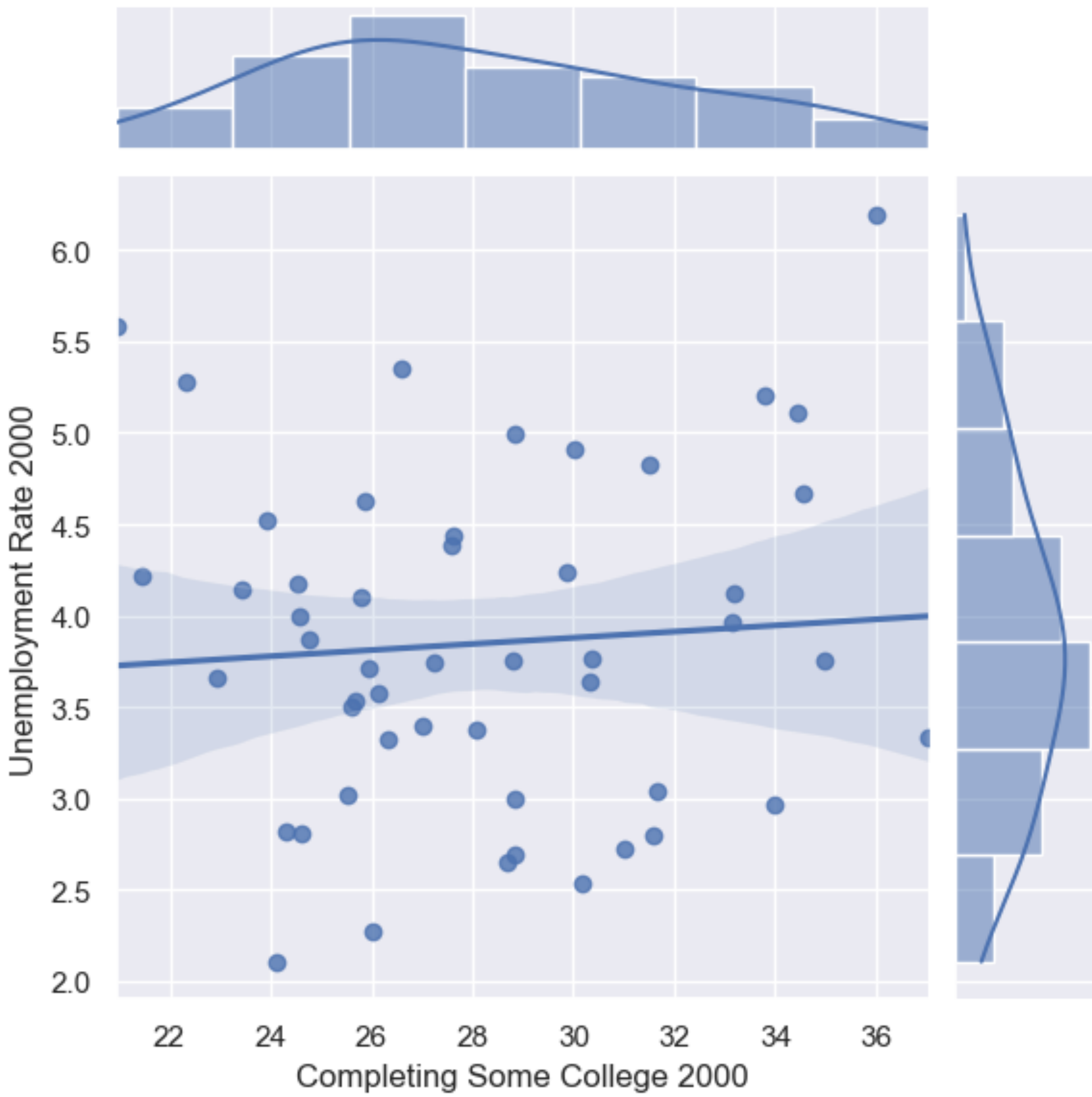
Conclusion: According to the jointplot and pairplot, we acknowledge the positive correlation between the percentage of adults with less than a high school diploma and unemployment rate in the year 2000. It displays a positive slope. This shows as one variable increases the other one increases also. We see that as the percent of adults with less than a high school diploma increases, the unemployment rate increases.

Question 8: What will happen if the adults complete some college or complete a bachelor's degree or higher in the year 2000? (Ping)

In [104…

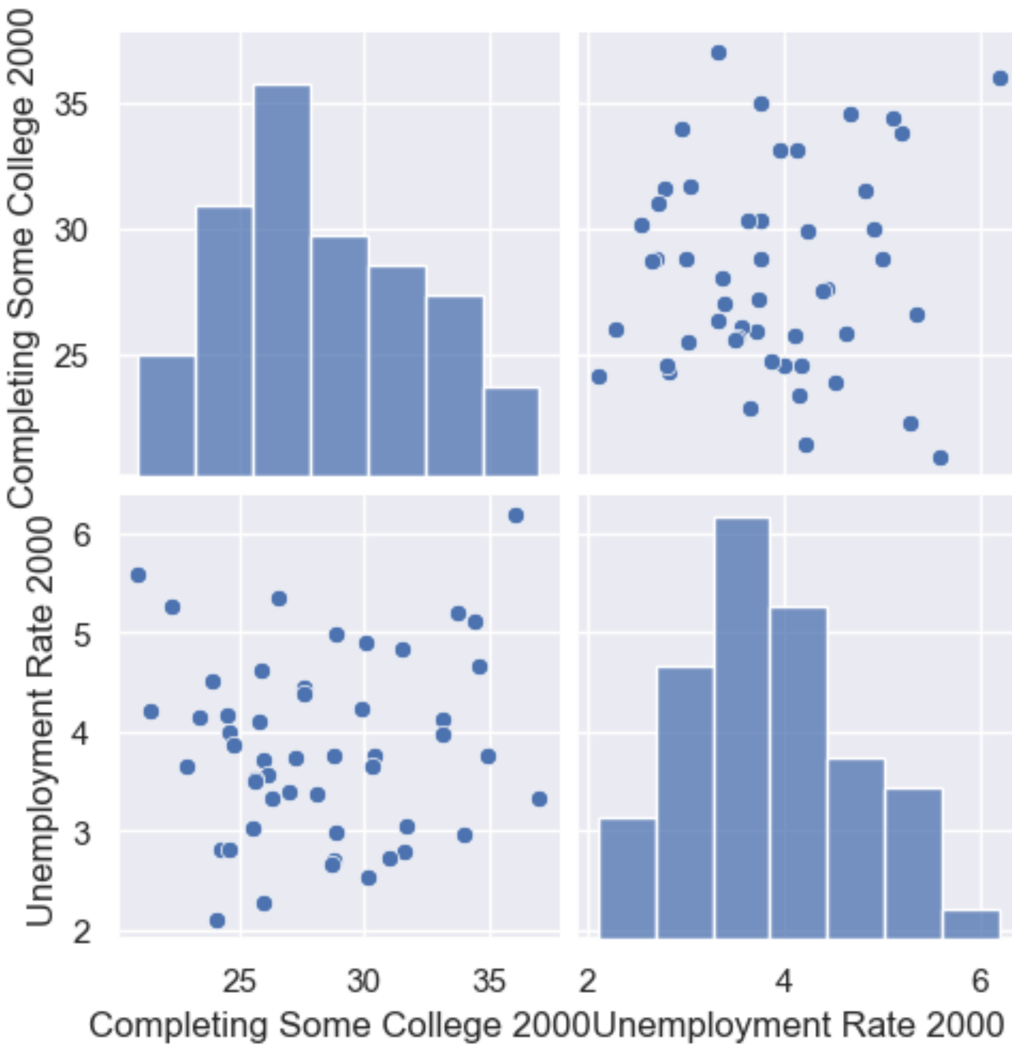
```
# Use Jointplot to show correlation between Unemployment Rate 2000 and Percen
df2 = df_data_by_state[['State_x', '% Some College, 2000','Unemployment Rate 
# Rename X-axis
df2 = df2.rename(columns = {"% Some College, 2000":"Completing Some College 2
# Draw Jointplot
sns.jointplot(x='Completing Some College 2000',y='Unemployment Rate 2000',dat
```

Out[104… <seaborn.axisgrid.JointGrid at 0x7fddbf2e4400>



In [105…

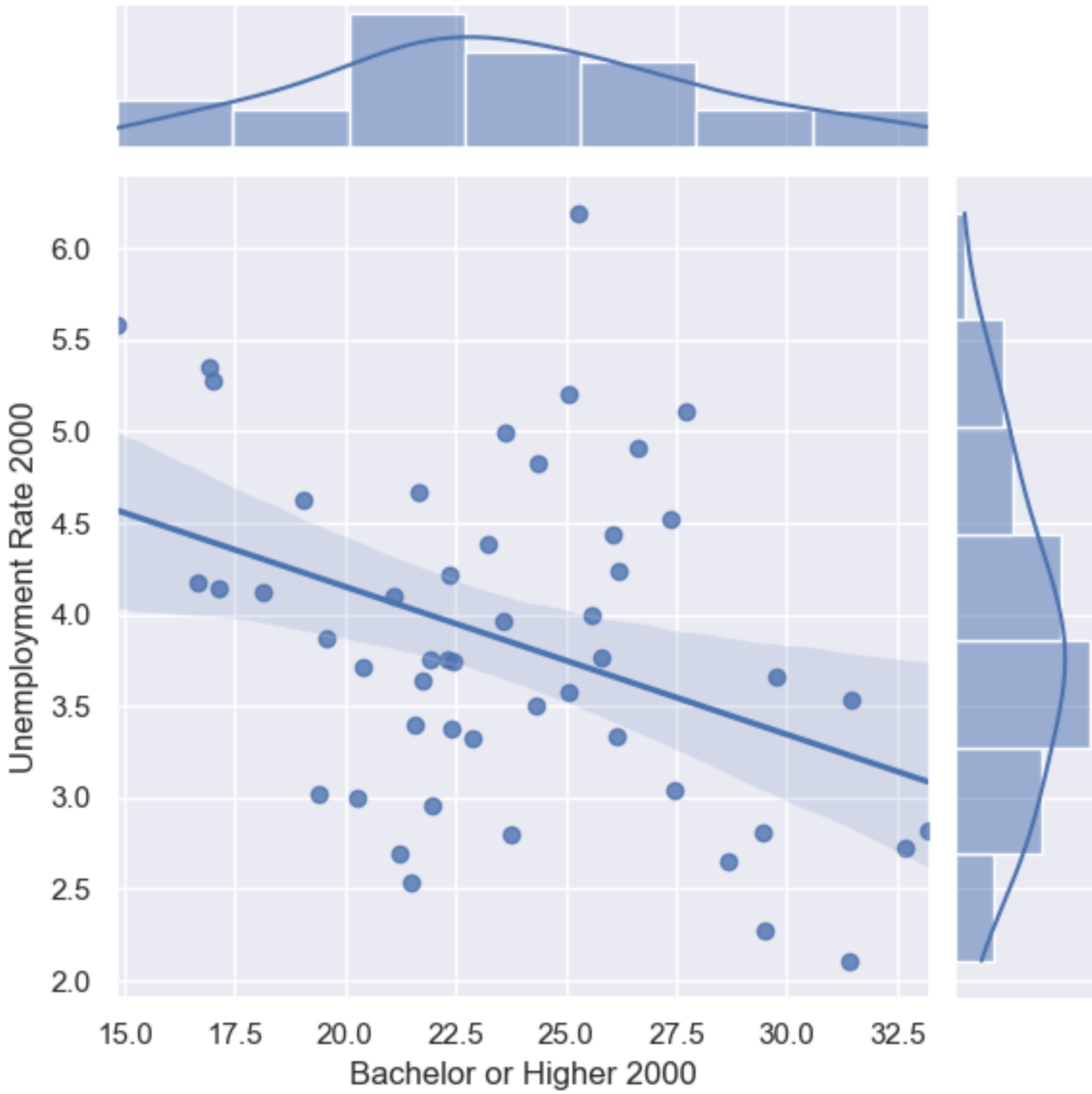
```
#Define Function for the the adults completing some college
def draw_pairplot(df):
    sns.pairplot(df)

#Draw pairplot
draw_pairplot(df2)
```



In [106…

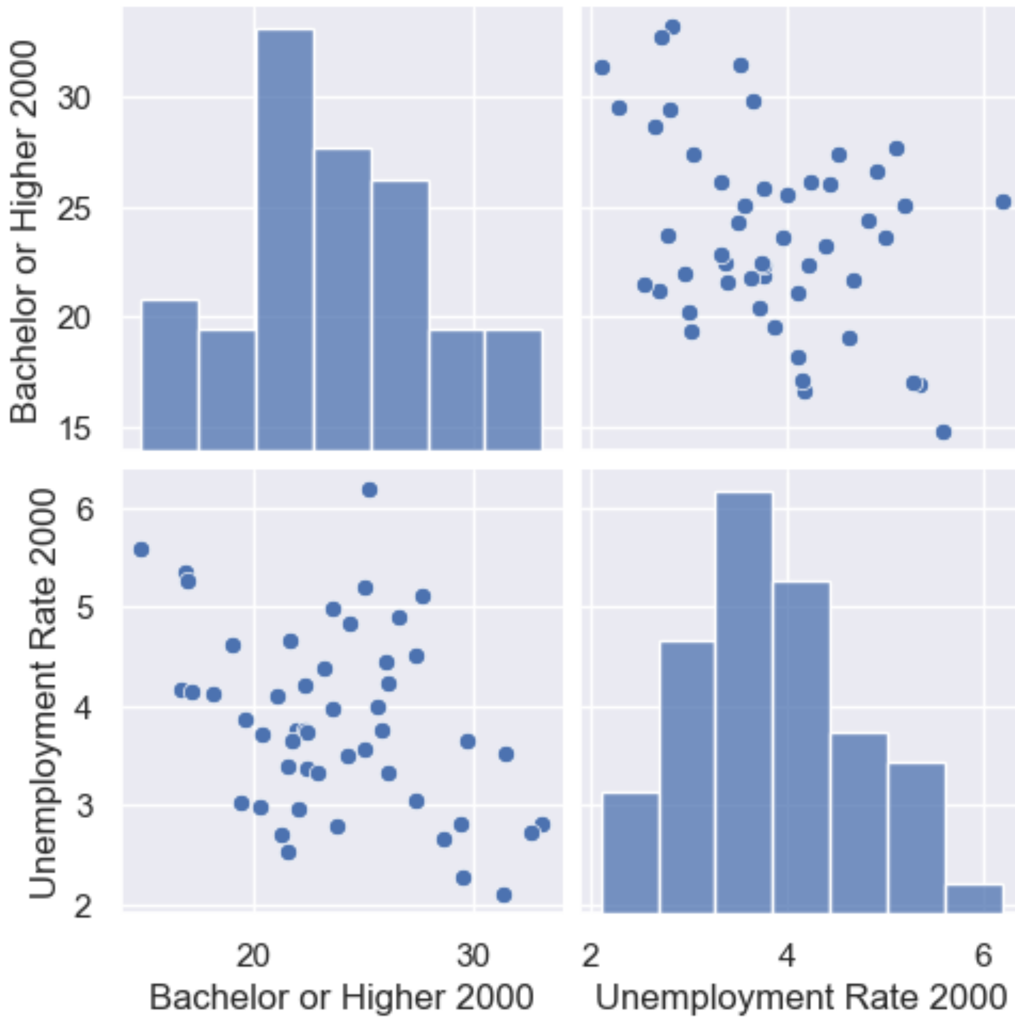
```
# Use Jointplot to show correlation between Unemployment Rate 2000 and Percen

df3 = df_data_by_state[["State_x", "% >= Bachelors, 2000","Unemployment Rate 

# Rename it to a more formal name for the X Axis
df3 = df3.rename(columns = {"% >= Bachelors, 2000":"Bachelor or Higher 2000"}

# Draw the jointplot
sns.jointplot(x= "Bachelor or Higher 2000",y="Unemployment Rate 2000",data=df
```

Out[106… <seaborn.axisgrid.JointGrid at 0x7fddc5987df0>



In [107…

```
#Define Function
def draw_pairplot(df):
    sns.pairplot(df)

#Draw the pairplot
draw_pairplot(df3)
```



Conclusion: From the jointplot and pairplot of the percentage of adults who complete some college versus unemployment rate, we can see the weak correlation between the variables.In a way, we can see the slope slowly change toward the opposite direction.Obviously, the correlation between the percent of adults with a bachelor's degree or higher and unemployment rate displays a negative linear relationship. From the positive correlation between the percentage of adults with less than a high school diploma and unemployment rate to the negative correlation between percentage of the adults with a bachelor's degree or higher and unemployment rate, we can conclude there is correlation between the variables.

Question 9: How has the civilian labor force changed in City/Suburb/Town/Rural areas from 2000, 2010 and 2020?

In [108…

```
# Calculate the percent change in total labor force using the formula
# % Difference = [(New_value - Previous_value) / (Previous_value)] * 100%
# Define Function
def calculate(df_pt1):
    df_pt1['% 2000 to 2010 Change'] = ((df_pt1["Civilian labor force 2010"] -
    df_pt1['% 2010 to 2020 Change'] = ((df_pt1["Civilian labor force 2020"] -
    df_pt1['% 2000 to 2020 Change'] = ((df_pt1["Civilian labor force 2020"] -

    return df_pt1
```

In [109…

```
# Create a pivot table stored in a new DataFrame that will provide the total
# the years 2000 and 2010
df_pt1 = pd.pivot_table(df_merged, index='City/Suburb/Town/Rural 2013', value

# Print Results
df_pt1
calculate(df_pt1)
```

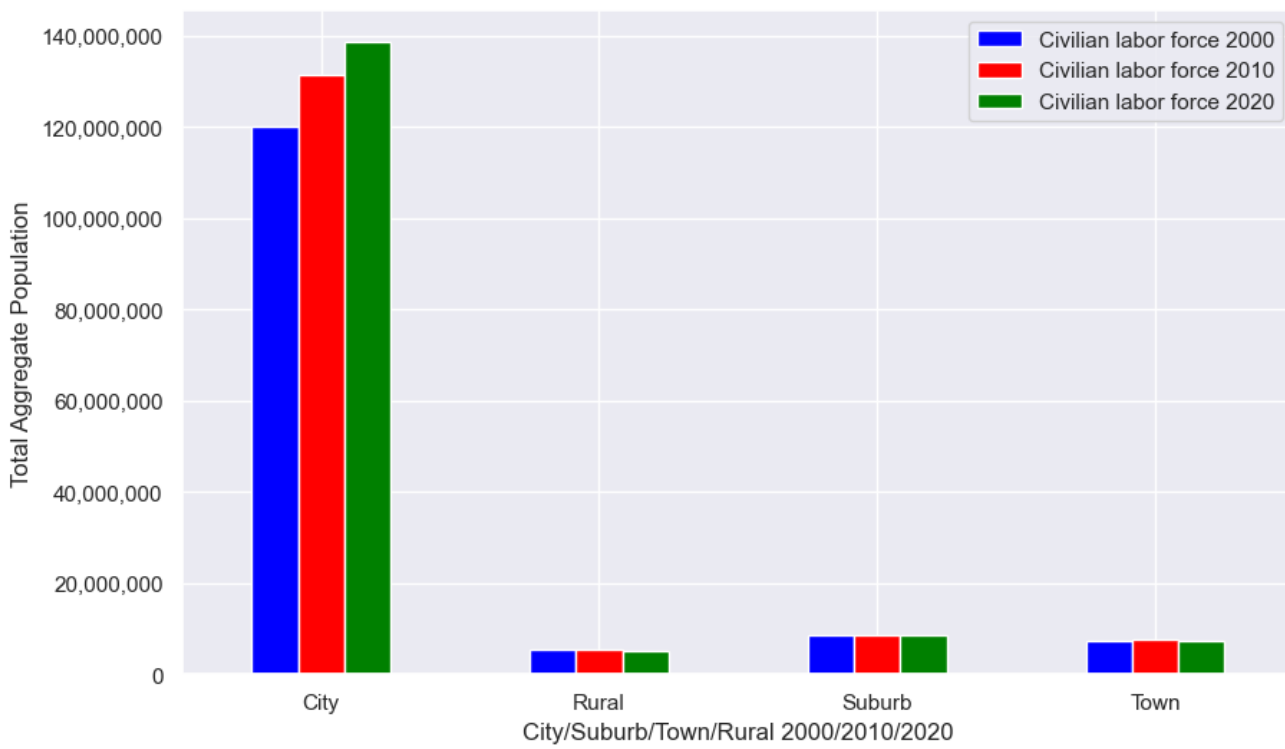
Out[109…

City/Suburb/Town/Rural 2013	Civilian labor force 2000	Civilian labor force 2010	Civilian labor force 2020	% 2000 to 2010 Change	% 2010 to 2020 Change	% 200 to 202 Chang
City	120194652.0	131553875.0	138665536.0	9.450689	5.405892	15.36747
Rural	5447956.0	5454747.0	5153247.0	0.124652	-5.527296	-5.40953
Suburb	8545086.0	8708303.0	8439451.0	1.910069	-3.087306	-1.23620
Town	7394947.0	7531767.0	7249178.0	1.850182	-3.751962	-1.97119

In [110…

```
#Import requried library
import matplotlib as mpl

# Create a DataFrame to hold the aggregate population data aggregated by area
# Show the total aggregate population for civilian labor force 2000 (blue), c
df_pt1 = df_merged.groupby('City/Suburb/Town/Rural 2013')["Civilian labor forc

# Plot the aggregated population values based on area (City/Suburb/Town/Rural
ax = df_pt1.sum().plot.bar(color=['blue', 'red', 'green', 'cyan'])


# Modify the plot layout parameters
plt.ticklabel_format(axis="y", style="plain", scilimits=(0,0))
plt.xticks(rotation=0)
ax.yaxis.set_major_formatter(mpl.ticker.StrMethodFormatter('{x:,.0f}'))

#Label X and Y axes
ax.set_xlabel("City/Suburb/Town/Rural 2000/2010/2020")
ax.set_ylabel("Total Aggregate Population")
```

Out[110… Text(0, 0.5, 'Total Aggregate Population')



Conclusion: From the graph, we see the civilian labor force in City is increasing in a stable rate, but the civilian labor force in Suburb, Town and Rural are stay almost unchanged. In the last 10 years, our technology has grown rapidly. It may correlated to the growth in city population since most of the technology jobs are available in the city. People may need to relocate to cities in order to work in the technology industry. At the same time, higher education become essential.

Hypothesis Testing

Normal Test

Testing whether the data for "Unemployment rate 2000" is normally distributed or not.

H0 (Null Hypothesis): The data is normal distributed

H1 (Alternative Hypothesis): The data is not normal distributed

In [111…

```
#Null Hypothesis can include =, <=, or => sign
#A general statement or default position that there is no relationship betwee
#or no association among groups

#Alternative Hypothesis can include NOT= or !=, >, or < sign
#It is the hypothesis used in hypothesis testing that is contrary to the Null

#Reference Website: https://towardsdatascience.com/hypothesis-testing-in-mach

(Statistic, p_value) = stats.normaltest(df_merged['Unemployment rate 2000'])
```

In [112…

```
#Print Result
print("Normal Test Result")

# 10 and 100 are the numbers behind decimal for Test Statistic and P-Value
print("Statistic is: ", round(Statistic,10))
print("P-Value is: ", round(p_value, 100))

#Print result which is either Accept or Reject, and it determined by P-Value.
if p_value < 0.05:
       print("Reject Null Hypothesis")
else:
       print("Accept Null Hypothesis")
```

```
Normal Test Result
Statistic is:  1204.3506327513
P-Value is:  0.0
Reject Null Hypothesis
```

Conclusion: The reason behind rejected Null Hypothesis is because P-Value is less than 0.05, so the unemployment rate for the year 2000 is not normal distributed.

Z-Test

Sample size is greater than 30. (N > 30)

Data points should be independent from each other

Data should be normally distributed, but for a large sample size (>30) this does not always matter

If the population standard deviation, sigma is known, then we use Z-Test

$$H_0 : \mu = \mu_0 \tag{5}$$
$$H_1 : \mu \neq \mu_0 \tag{6}$$

In [113…
```
#Test Z-Test
#Test the mean of the labor force in 2000 is 45,000
#The Alternative Hypothesis is that the mean of the labor force in 2000 is NO

(Statistic, p_value) = ztest(df_merged['Civilian labor force 2000'], value=45
```

In [114…
```
#Print results
print("Z-Test")

# 10 and 100 are the numbers behind decimal for Test Statistic and P-Value
#Evaluate Z-TEST STATISTIC AND P-VALUE
print("Test Statistic is: ", round(Statistic,10))
print("P-Value is: ", round(p_value, 100))

#Print result which is either Accept or Reject, and it determined by P-Value.
if p_value < 0.05:
       print("Reject Null Hypothesis")
else:
       print("Accept Null Hypothesis")
```

```
Z-Test
Test Statistic is:  0.1596475346
P-Value is:  0.8731587319927478
Accept Null Hypothesis
```

Conclusion: Since this P-Value of the Z-Test is greater than 0.05, we do not have sufficient evidence to reject the null hypothesis. In other words, we conclude that the mean of the labor force in 2000 is 45,000.

Correlation Test

I have tested from Q1 and Q2. I also show the correlation test from calculation.

Question 1: What is the correlation between the adults with less than a high school diploma 2000 and unemployment rate in the year of 2000?

Question 2: What will happen if the adults completing some college or even completing a bachelor's degree or higher 2000?

Null Hypothesis (H0): the two samples are independent.

Alternative Hypothesis (H1): there is a dependency between the samples.

Tested the correlation between

1. The percent of adults with less than a high school diploma and unemployment rate in the year of 2000

Null Hypothesis (H0): The percent of adults with less than a high school diploma and unemployment rate in the year of 2000 are independent.

Alternative Hypothesis (H1): There is a dependency between the percent of adults with less than a high school diploma and unemployment rate in the year of 2000.

1. The percent of adults completing some college and unemployment rate in the year of 2000

Null Hypothesis (H0): The percent of adults completing some college and unemployment rate in the year of 2000 are independent.

Alternative Hypothesis (H1): There is a dependency between the percent of adults completing some college and unemployment rate in the year of 2000.

1. The percent of adults with a bachelor's degree or higher and unemployment rate in the year of 2000

Null Hypothesis (H0): The percent of adults with a bachelor's degree or higher and unemployment rate in the year of 2000 are independent.

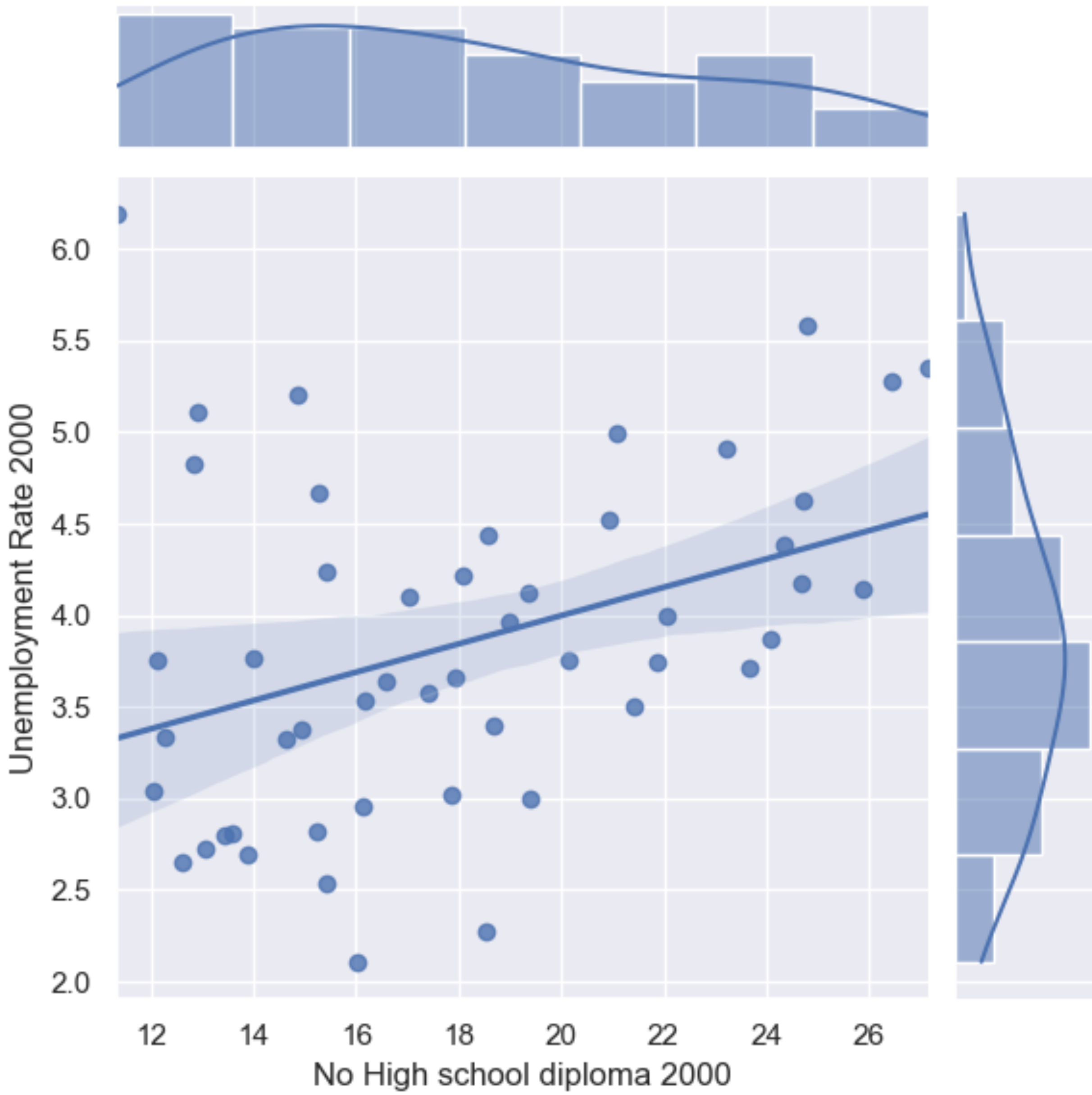
Alternative Hypothesis (H1): There is a dependency between the percent of adults with a bachelor's degree or higher and unemployment rate in the year of 2000.

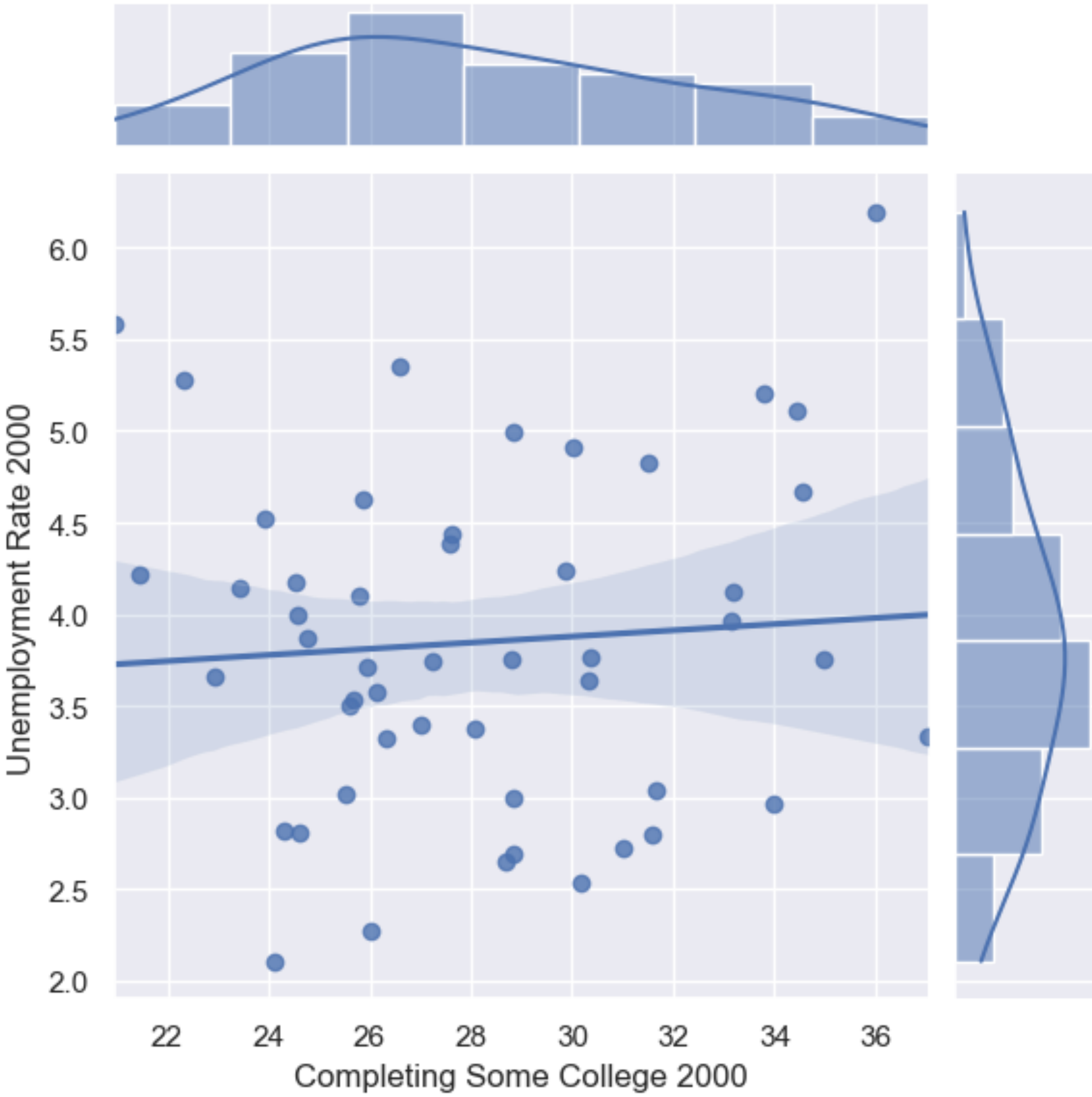
In [115…

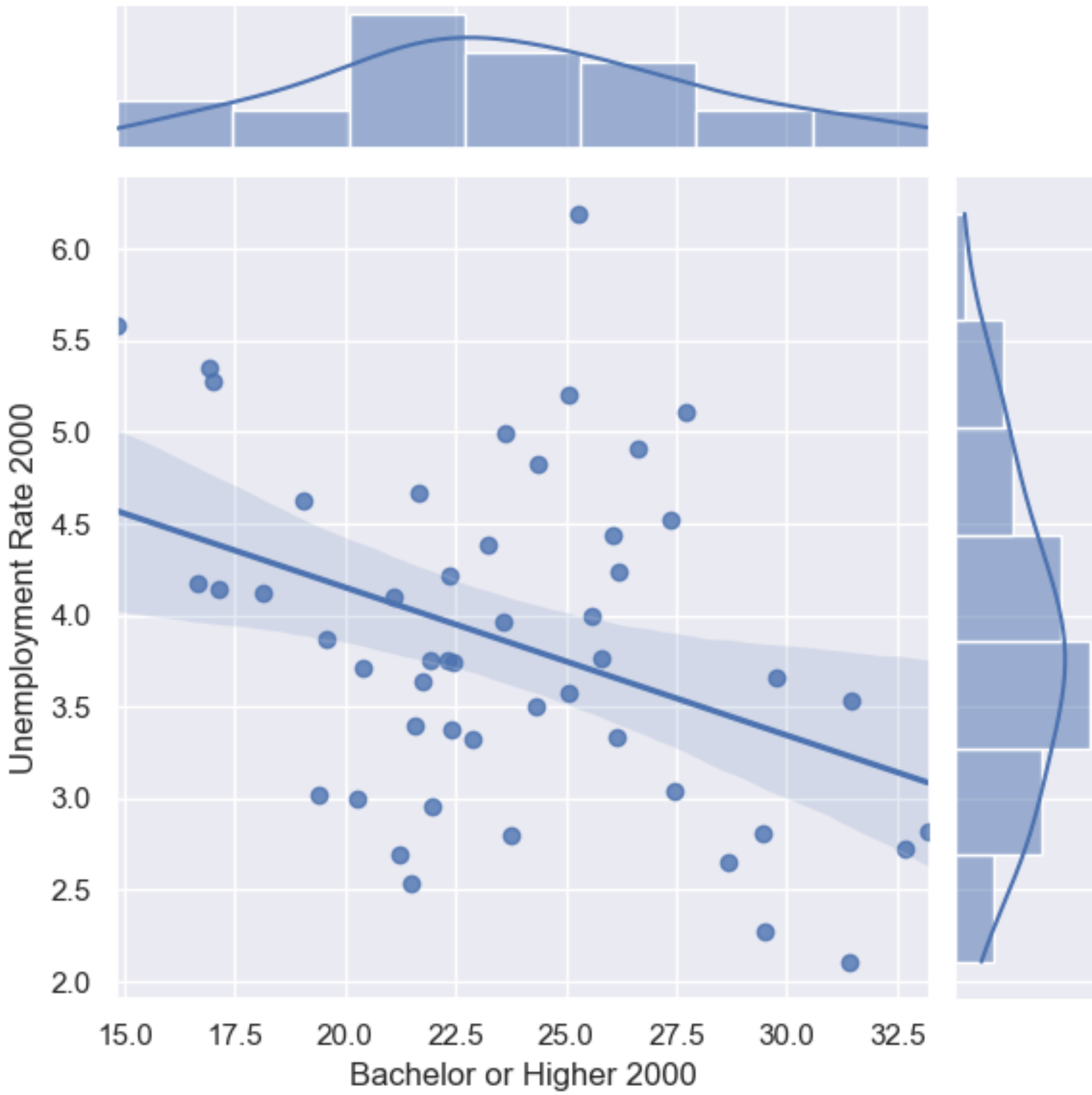
```
# Show Correlation between
# 1. The percent of adults with less than a high school diploma and unemploym
df1 = df_data_by_state[['State_x', '% < HS Diploma, 2000','Unemployment Rate 

df1 = df1.rename(columns = {"% < HS Diploma, 2000":"No High school diploma 20
sns.jointplot(x='No High school diploma 2000',y='Unemployment Rate 2000',data
# Positive Correlation


# Show Correlation between
# 2. The percent of adults completing some college and unemployment rate in t
df2 = df_data_by_state[['State_x', '% Some College, 2000','Unemployment Rate 

df2 = df2.rename(columns = {"% Some College, 2000":"Completing Some College 2
sns.jointplot(x='Completing Some College 2000',y='Unemployment Rate 2000',dat
#Very slight positive correlation to almost no correlation


# Show Correlation between
# 3. The percent of adults with a bachelor's degree or higher and unemploymen
df3 = df_data_by_state[["State_x", "% >= Bachelors, 2000","Unemployment Rate 

df3 = df3.rename(columns = {"% >= Bachelors, 2000":"Bachelor or Higher 2000"}
sns.jointplot(x= "Bachelor or Higher 2000",y="Unemployment Rate 2000",data=df
#Negative Correlation
```

Out[115…

```
<seaborn.axisgrid.JointGrid at 0x7fddbf3022b0>
```


6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
Unemployment Rate 2000
12
14
16
18
20
22
24
26
No High school diploma 2000


6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
Unemployment Rate 2000
22
24
26
28
30
32
34
36
Completing Some College 2000



In [116…

```
# Tested the correlation between
# 1. The percent of adults with less than a high school diploma and unemploym
corr, p = pearsonr(df_merged["Unemployment rate 2000"], df_merged["% < HS Dip

print(f"Correlation coefficient: {corr}, P-value: {p}")
```

```
Correlation coefficient: 0.4852579122415984, P-value: 7.097382052376354e-184
```

In [117…

```
# 2. The percent of adults completing some college and unemployment rate in t
corr, p = pearsonr(df_merged["Unemployment rate 2000"], df_merged["% Some Col

print(f"Correlation coefficient: {corr}, P-value: {p}")
```

```
Correlation coefficient: -0.24359426787193705, P-value: 2.4492868410404643e-43
```

In [118…

```
# 3. The percent of adults with a bachelor's degree or higher and unemploymen
corr, p = pearsonr(df_merged["Unemployment rate 2000"], df_merged["% >= Bache

print(f"Correlation coefficient: {corr}, P-value: {p}")
```

```
Correlation coefficient: -0.3856131212048477, P-value: 4.719921080791591e-111
```

Conclusion from the Statistics side:

All of the P-Value are extremely close to zero. Since P-Value is less than 0.001, so there is strong eveidence for correlation.

Conclusions of the correlation between two samples:

1. The percent of adults with less than a high school diploma and unemployment rate in the year of 2000 Reject the Null Hypothesis (H0) and accept Alternative Hypothesis (H1) because it does reveal a positive correlation betwee the two samples in the graph.

2. The percent of adults completing some college and unemployment rate in the year of 2000 Accept the Null Hypothesis (H0) and reject the Alternative Hypothesis (H1) because it does reveal a very slightly positive correlation to almost no correlation betwee the two samples in the graph.

3. The percent of adults with a bachelor's degree or higher and unemployment rate in the year of 2000 Reject the Null Hypothesis (H0) and accept Alternative Hypothesis (H1) because it does reveal a negative correlation betwee the two samples in the graph.

Chi-Square Test

H0: "Percent of adults completing some college 2000" and "Unemployment rate 2000" are independent.

H1: "Percent of adults completing some college 2000" and "Unemployment rate 2000" are dependent.

H0: "Percent of adults with a bachelor's degree or higher 2000" and "Unemployment rate 2000" are independent.

H1: "Percent of adults with a bachelor's degree or higher 2000" and "Unemployment rate 2000" are dependent.

In [119…

```
# Reference Website (https://docs.scipy.org/doc/scipy/reference/generated/sci
# Reference Website (https://docs.scipy.org/doc/scipy/reference/generated/sci

# import required libraries
from scipy.stats import chi2_contingency
from scipy.stats import chisquare

table = df_merged["% Some College, 2000"], df_merged["Unemployment rate 2000"

#print(table)
stat, p, dof, expected = chi2_contingency(table)

print('Test statistic: ', stat)
print('P-value: ', p)
print('Degrees of freedom (dof):', dof)
print('expected values:', expected)

#Set probability equal to 95%
prob = 0.95
critical_value = chi2.ppf(prob, dof)

# Print the probability, critical value
print('Probability: ', prob)
print('Critical value: ', critical_value)

# Evaluate the test statistic
print('Evaluate the test statistic:')
if abs(stat) >= critical_value:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')

# Evaluate the P-value
print('Evaluate the P-value:')
alpha = 1.0 - prob
print('Significance: ', alpha)
if p <= alpha:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')
```

```
Test statistic:  2870.8631293077046
P-value:  0.9992689958224099
Degrees of freedom (dof): 3116
expected values: [[22.1115117  24.25410004 21.25447636 ... 28.28216613 32.5673
4282
  31.45319688]
 [ 3.6884883   4.04589996  3.54552364 ...  4.71783387  5.43265718
   5.24680312]]
Probability:  0.95
Critical value:  3246.976756354243
Evaluate the test statistic:
Independent (fail to reject H0)
Evaluate the P-value:
Significance:  0.050000000000000044
Independent (fail to reject H0)
```

In [120…

```
# Reference Website (https://docs.scipy.org/doc/scipy/reference/generated/sci
# Reference Website (https://docs.scipy.org/doc/scipy/reference/generated/sci

# import required libraries
from scipy.stats import chi2_contingency
from scipy.stats import chisquare

table = df_merged["% >= Bachelors, 2000"], df_merged["Unemployment rate 2000"

#print(table)
stat, p, dof, expected = chi2_contingency(table)

print('Test statistic: ', stat)
print('P-value: ', p)
print('Degrees of freedom (dof):', dof)
print('expected values:', expected)

#Set probability equal to 95%
prob = 0.95
critical_value = chi2.ppf(prob, dof)

# Print the probability, critical value
print('Probability: ', prob)
print('Critical value: ', critical_value)

# Evaluate the test statistic
print('Evaluate the test statistic:')

#Print Result (Reject or Fail to Reject H0)
if abs(stat) >= critical_value:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')

# Evaluate the P-value
print('Evaluate the P-value:')
alpha = 1.0 - prob

# Evaluate Significance Value
print('Significance: ', alpha)

# Print results (Reject or Fail to reject H0)
if p <= alpha:
  print('Dependent (reject H0)')
else:
  print('Independent (fail to reject H0)')
```

```
Test statistic:  4742.658884176255
P-value:  1.8973530558983913e-71
Degrees of freedom (dof): 3116
expected values: [[ 9.88848472 10.36313198 11.23331864 ... 17.79927249 16.4544
3857
  14.95138889]
 [ 2.61151528  2.73686802  2.96668136 ...  4.70072751  4.34556143
   3.94861111]]
Probability:  0.95
Critical value:  3246.976756354243
Evaluate the test statistic:
Dependent (reject H0)
Evaluate the P-value:
Significance:  0.050000000000000044
Dependent (reject H0)
```

Conclusion:

According to Paul's work, we fail to reject the null hypothesis which is "Percent of adults with less than a high school diploma, 2000" and "Unemployment rate 2000". Extending from there, by Chi-Square Test, we fail to reject Null Hypothesis which is "Percent of adults completing some college 2000" and "Unemployment rate 2000" are independent. We reject the "Percent of adults with a bachelor's degree or higher 2000" and "Unemployment rate 2000" are independent (H0), so I was able to confirm that "Percent of adults with a bachelor's degree or higher 2000" and "Unemployment rate 2000" are dependent.

ANOVA

ANOVA test returns two values:

F-test score: variation between sample group means divided by variation within sample group.

P-value: Confidence degree. The p-value shows whether the obtained result is statistically significant

Null Hypothesis (H0):In City, Suburb, Rural, and Town areas, there is no difference in the mean percentage of adults with the adults who completing Some College 2000.

Alternative Hypothesis (H1): In City, Suburb, Rural, and Town areas, there is difference in the mean percentage of adults with the adults who completing Some College 2000.

Null Hypothesis (H0):In City, Suburb, Rural, and Town areas, there is no difference in the mean percentage of adults with the adults with a bachelor's degree or higher 2000.

Alternative Hypothesis (H1): In City, Suburb, Rural, and Town areas, there is difference in the mean percentage of adults with the adults with a bachelor's degree or higher 2000.

In [121…

```
#Show Groups
df_anova = df_merged[["City/Suburb/Town/Rural 2013","% Some College, 2000"]]

# Group the annova dataframe by area type (City/Suburb/Town/Rural 2013)
df_anova_groupby_area = df_anova.groupby(['City/Suburb/Town/Rural 2013'])

anova_result_1 = stats.f_oneway(df_anova_groupby_area.get_group("City")["% So
                                df_anova_groupby_area.get_group("Suburb")["% 
                                df_anova_groupby_area.get_group("Rural")["% S
                                df_anova_groupby_area.get_group("Town")["% So

print( "ANOVA results: F=",anova_result_1)
```

```
ANOVA results: F= F_onewayResult(statistic=21.864460890602306, pvalue=5.211724
676158826e-14)
```

In [122…

```
#Dataframe
df_anova = df_merged[["City/Suburb/Town/Rural 2013","% >= Bachelors, 2000"]]

# Group the annova dataframe by area type (City/Suburb/Town/Rural 2013)
df_anova_groupby_area = df_anova.groupby(['City/Suburb/Town/Rural 2013'])

anova_result_1 = stats.f_oneway(df_anova_groupby_area.get_group("City")["% >=
                                df_anova_groupby_area.get_group("Suburb")["% 
                                df_anova_groupby_area.get_group("Rural")["% >
                                df_anova_groupby_area.get_group("Town")["% >=

#Print Result
print( "ANOVA results: F=",anova_result_1)
```

```
ANOVA results: F= F_onewayResult(statistic=166.12955659067728, pvalue=6.804715
210427925e-100)
```

Conclusion:

According to the ANOVA results, both F-statistic is larger than 1 and both P-Value is much less than 0.05. This reveals the evidence to reject the null hypotheses.

Reject -> Null Hypothesis (H0):In City, Suburb, Rural, and Town areas, there is no difference in the mean percentage of adults with the adults who completing Some College 2000.

Acccept -> Alternative Hypothesis (H1): In City, Suburb, Rural, and Town areas, there is difference in the mean percentage of adults with the adults who completing Some College 2000.

Reject -> Null Hypothesis (H0):In City, Suburb, Rural, and Town areas, there is no difference in the mean percentage of adults with the adults with a bachelor's degree or higher 2000.

Acccept -> Alternative Hypothesis (H1): In City, Suburb, Rural, and Town areas, there is difference in the mean percentage of adults with the adults with a bachelor's degree or higher 2000.

Project Summary: Answers to the Stated Questions

1. A charity organization wants to explore which communities have residents that need help in completing their high school education. Which communities should they look at to do the most good? (Paul)

The results of the ANOVA test show that there is a significant difference in City, Suburb, Rural and Town areas when looking for people with less than a high school diploma. The results of the Pearson Correlation test show a correlation between a county's unemployment rate and residents having less than a high school diploma. Looking at the U.S. map, one can see that Louisiana, Mississippi, Tennessee and Kentucky all have a large percentage of residents with less than a high school diploma, and also have high unemployment. The residents in those states show the greatest need.

2. Even in U.S. states with low or moderate unemployment rates, are there counties with unusually high or low unemployment? (Paul)

Yes. In the chart "Boxplot Showing County Ranges of Unemployment, 2000" it is clear that several states have counties representing outliers for high unemployment. Notably Texas with a median unemployment rate of about 5%, there are counties with double-digit unemployment, with one county as high as 17%. However, in all fifty states, only Massachusetts and Wyoming had outliers representing counties with unusually low unemployment rates compared with the state median unemployment rate.

3. Does having a bachelor's degree (or higher) or just a high school diploma correlate better with low unemployment? (Paul)

Having a bachelor's degree (or higher) correlates better with lower unemployment. Unemployment is negatively correlated with having a bachelor's degree (-0.381) and having only a high school diploma (-0.0725). but the negative correlation is stronger for having the bachelor’s degree or higher.

In other words, the higher the level of education in a county, the lower the unemployment rate will be.

4. Which years have the highest and lowest unemployment rate over the course of 21 years? (Fiona)

From year 2000 to 2020, 2010 has the highest unemployment rate 9.64%, while 2019 has the lowest unemployment rate 3.66%. However, there was a great increase on unemployment rate in year 2020.

5. Which states contribute the most and the least for the unemployment change from year 2019 to 2020? (Fiona)

All The states have increased unemployment rate from year 2019 to 2020. The five states that have the highest unemployment rate increases are Hawaii, Nevada, Massachusetts, New Jersey, Colorado. While the five states that have the lowest unemployment rate increases are Nebraska, Alaska, Mississippi, South Dakota, and Wyoming. The ECDF plot shows almost 50% of the states have unemployment rate changes which are higher than 100%.

6. Is there a significant change regarding percentages of people completing different diplomas between year 2000 and year 2015-2019? (Fiona)

There is a significant change in the percentages of people completing different diplomas between year 2000 and year 2015-2019, with more people having bachelor's degree or higher, and fewer people having less than a high school diploma. However, the percentages of people with high school diploma only and college or associate's degree don't have too much difference. hat is the correlation between the adults with less than a high school diploma 2000 and unemployment rate in the year of 2000? (Ping) (Answer)

7. What is the correlation between the adults with less than a high school diploma and unemployment rate in the year 2000? (Ping)

From the jointplot, pairplot, and hypothesis test, I found a positive correlation between the percentage of adults with less than a high school diploma and unemployment rate. It displays a positive slope. We cannot conclude that one affects the other, but we do see that as the percentage of adults with less than a high school diploma increases, the unemployment rate increases. We can conclude that there is dependency (Correlation) between the two variables. At the end, the Chi-Square Test’s result also confirmed that the two variables are dependent.

8. What will happen if the adults complete some college or complete a bachelor's degree or higher in the year 2000? (Ping)

From the jointplot and pairplot of the percentage of adults completing some college versus unemployment rate, we can see the weak positive correlation between the variables. In a way, we can see the slope slowly change toward the opposite direction. Then, the correlation between the percent of adults with a bachelor's degree or higher and unemployment rate displays a negative linear relationship. Also, I did Hypothesis Tests and Chi-Square Tests to support my conclusion from the plots. From the positive correlation between the percentage of adults with less than a high school diploma and unemployment rate to the negative correlation between the percentage of the adults with a bachelor's degree or higher and unemployment rate. We can see strong evidence of dependency.

9. How has the civilian labor force changed in City/Suburb/Town/Rural areas from 2000, 2010 and 2020? (Ping)

Civilian labor force for cities reveals a steady growth, but there is almost no changes in Suburb/Town/Rural areas. I want to find the reasons behind the results of the dependency for the variables from my previous two questions, so I start to search for the reasons from a different angle. In the last 20 years, our technology has grown rapidly which led to the changes in the job market. Many people are forced to go back to school in order to be able to get technology related jobs. Tech companies are usually located in cities, and it might be a major reason leading to the increasing population in cities. From this perspective, the more educated people get, then they will have higher job opportunities. This supports the correlation between the various education levels versus unemployment rate.

In [ ]:

```
```