ETL Project Report

Title: GDP, Unemployment and Health Insurance Analysis for 2008-2018
Members: Ju A Han, Lakshmi Prasanna Rangam

Extraction

Data Sources:
We used the three datasets from three different websites. We collected the data by each county in the States from 2008 to 2018. In terms of the timeframe, the plan was to collect data for recent 10 years, but we decided to extend one more year so that we include the great recession data in the analysis.

- GDP growth Data: Bureau of Economic Analysis (https://www.bea.gov/data/gdp/gdp-county-metro-and-other-areas)
- Unemployment: US Bureau of Labor Statistics (https://www.bls.gov/lau/#cntyaa)
- Health Insurance Coverage: US Census Bureau (https://www.census.gov/data-tools/demo/sahie/#/)

Data Format:
Datasets from the US Census Bureau, Bureau of Economic Analysis and US Bureau of Labor Statistics are in csv format. We downloaded a total of 15 files from the three websites.

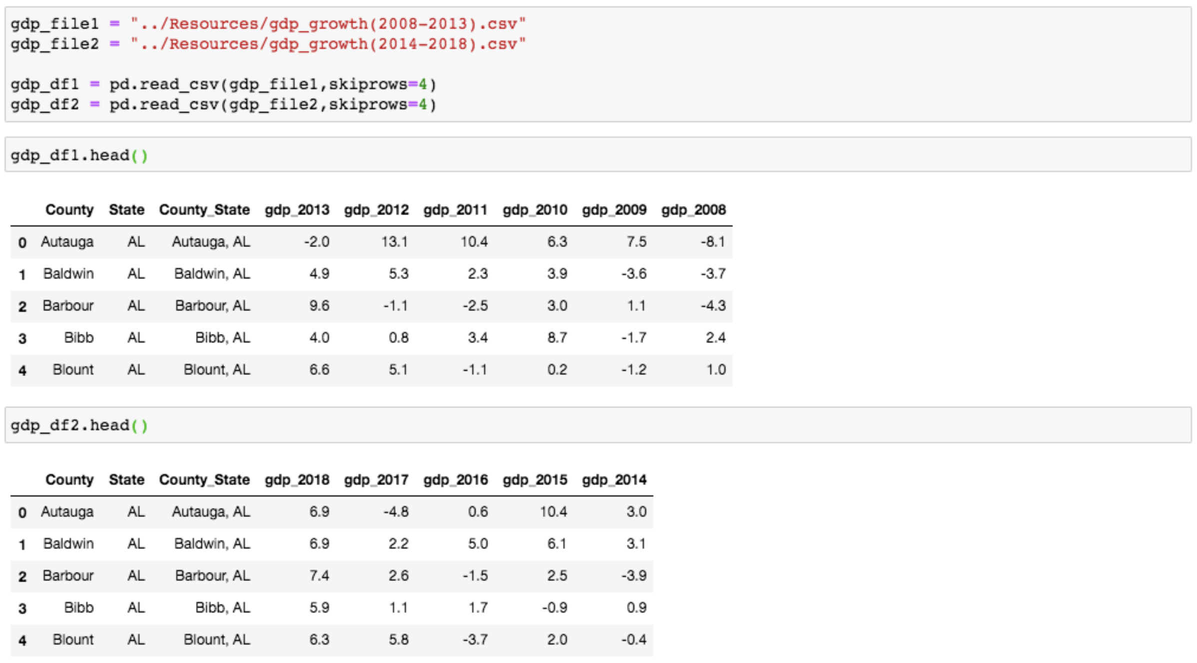
Transformation

We used *pandas* for the transformation of data. Main types of transformation used are filtering, renaming, splitting, dropping, merging, etc.

All the datasets had a column in common, "county-state", which will later be used for merging. However, the columns were not in the same format, so we needed to modify them using replace() method to remove the word "county". Moreover, all three datasets had NA values for some counties since they were newly created or removed during the years. We had to fill the NA values with zero to make sure that no error occurs later. Finally, we dropped all the unnecessary columns and rows in the datasets.

GDP Dataset:

1) Downloaded and loaded two csv files



```
gdp_file1 = "../Resources/gdp_growth(2008-2013).csv"
gdp_file2 = "../Resources/gdp_growth(2014-2018).csv"

gdp_df1 = pd.read_csv(gdp_file1,skiprows=4)
gdp_df2 = pd.read_csv(gdp_file2,skiprows=4)
```

```
gdp_df1.head()
```

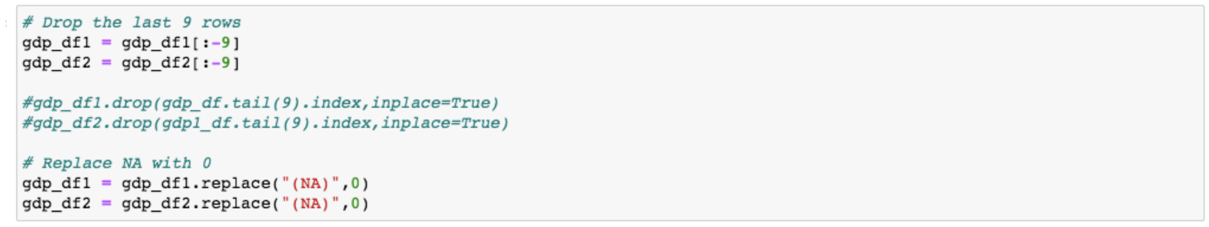
	County	State	County_State	gdp_2013	gdp_2012	gdp_2011	gdp_2010	gdp_2009	gdp_2008
0	Autauga	AL	Autauga, AL	-2.0	13.1	10.4	6.3	7.5	-8.1
1	Baldwin	AL	Baldwin, AL	4.9	5.3	2.3	3.9	-3.6	-3.7
2	Barbour	AL	Barbour, AL	9.6	-1.1	-2.5	3.0	1.1	-4.3
3	Bibb	AL	Bibb, AL	4.0	0.8	3.4	8.7	-1.7	2.4
4	Blount	AL	Blount, AL	6.6	5.1	-1.1	0.2	-1.2	1.0

```
gdp_df2.head()
```

	County	State	County_State	gdp_2018	gdp_2017	gdp_2016	gdp_2015	gdp_2014
0	Autauga	AL	Autauga, AL	6.9	-4.8	0.6	10.4	3.0
1	Baldwin	AL	Baldwin, AL	6.9	2.2	5.0	6.1	3.1
2	Barbour	AL	Barbour, AL	7.4	2.6	-1.5	2.5	-3.9
3	Bibb	AL	Bibb, AL	5.9	1.1	1.7	-0.9	0.9
4	Blount	AL	Blount, AL	6.3	5.8	-3.7	2.0	-0.4

Figure 1 GDP-Data: Reading csv file

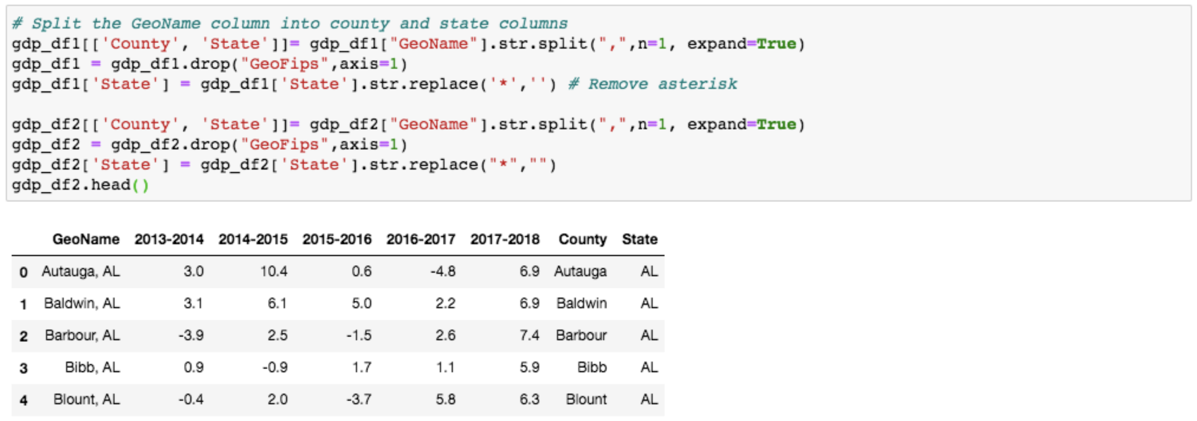
2) Dropped a few rows and columns that aren't needed.
3) Replaced a few NA values with zeros in both csv's.



```
# Drop the last 9 rows
gdp_df1 = gdp_df1[:-9]
gdp_df2 = gdp_df2[:-9]

#gdp_df1.drop(gdp_df.tail(9).index,inplace=True)
#gdp_df2.drop(gdp1_df.tail(9).index,inplace=True)

# Replace NA with 0
gdp_df1 = gdp_df1.replace("(NA)",0)
gdp_df2 = gdp_df2.replace("(NA)",0)
```

Figure 2 GDP-data: Dropping Rows and Replacing NAs

4) Split the county-state column into two different columns county and state.
5) State abbreviations from the state column have some asterisk symbols that need to be removed.

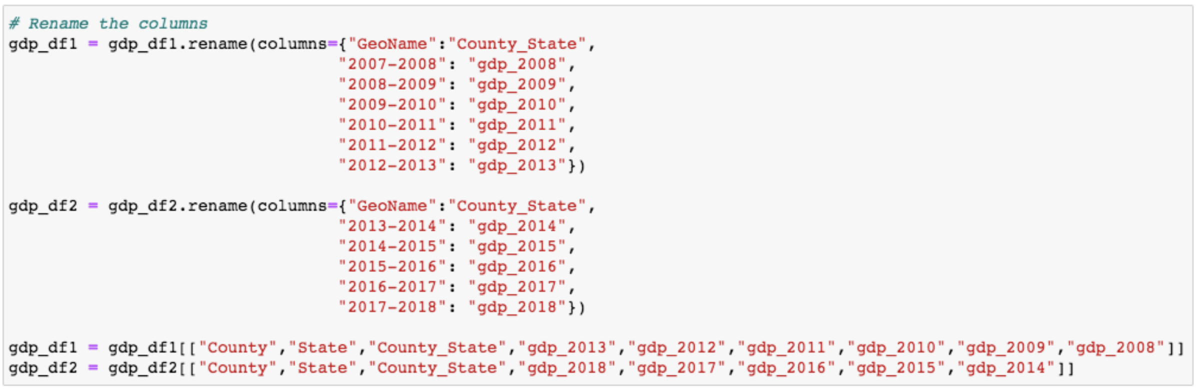


```
# Split the GeoName column into county and state columns
gdp_df1[['County', 'State']]= gdp_df1["GeoName"].str.split(",",n=1, expand=True)
gdp_df1 = gdp_df1.drop("GeoFips",axis=1)
gdp_df1['State'] = gdp_df1['State'].str.replace('*','') # Remove asterisk

gdp_df2[['County', 'State']]= gdp_df2["GeoName"].str.split(",",n=1, expand=True)
gdp_df2 = gdp_df2.drop("GeoFips",axis=1)
gdp_df2['State'] = gdp_df2['State'].str.replace("*","")
gdp_df2.head()
```

	GeoName	2013-2014	2014-2015	2015-2016	2016-2017	2017-2018	County	State
0	Autauga, AL	3.0	10.4	0.6	-4.8	6.9	Autauga	AL
1	Baldwin, AL	3.1	6.1	5.0	2.2	6.9	Baldwin	AL
2	Barbour, AL	-3.9	2.5	-1.5	2.6	7.4	Barbour	AL
3	Bibb, AL	0.9	-0.9	1.7	1.1	5.9	Bibb	AL
4	Blount, AL	-0.4	2.0	-3.7	5.8	6.3	Blount	AL

Figure 3 GDP-Data: Creating County and State Columns

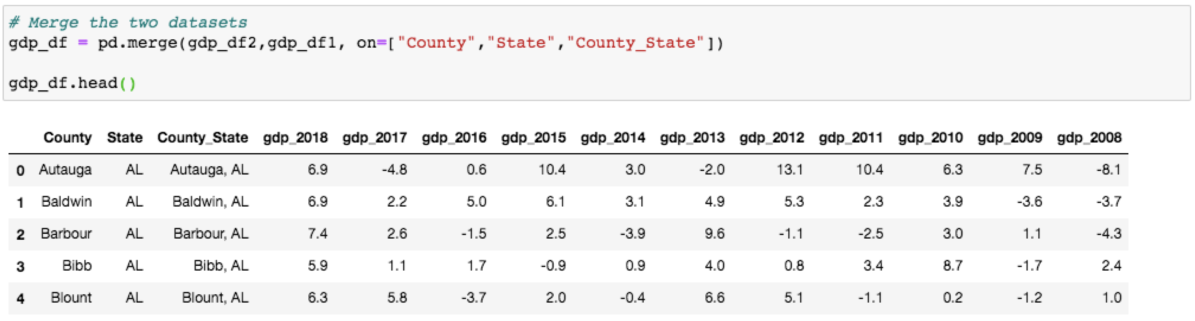
6) Renamed the columns as we needed.



```
# Rename the columns
gdp_df1 = gdp_df1.rename(columns={"GeoName":"County_State",
                                  "2007-2008": "gdp_2008",
                                  "2008-2009": "gdp_2009",
                                  "2009-2010": "gdp_2010",
                                  "2010-2011": "gdp_2011",
                                  "2011-2012": "gdp_2012",
                                  "2012-2013": "gdp_2013"})

gdp_df2 = gdp_df2.rename(columns={"GeoName":"County_State",
                                  "2013-2014": "gdp_2014",
                                  "2014-2015": "gdp_2015",
                                  "2015-2016": "gdp_2016",
                                  "2016-2017": "gdp_2017",
                                  "2017-2018": "gdp_2018"})

gdp_df1 = gdp_df1[["County","State","County_State","gdp_2013","gdp_2012","gdp_2011","gdp_2010","gdp_2009","gdp_2008"]]
gdp_df2 = gdp_df2[["County","State","County_State","gdp_2018","gdp_2017","gdp_2016","gdp_2015","gdp_2014"]]
```

Figure 4 GDP-Data: Renaming and Reordering

7) Two csv’s were merged into the final GDP dataframe.



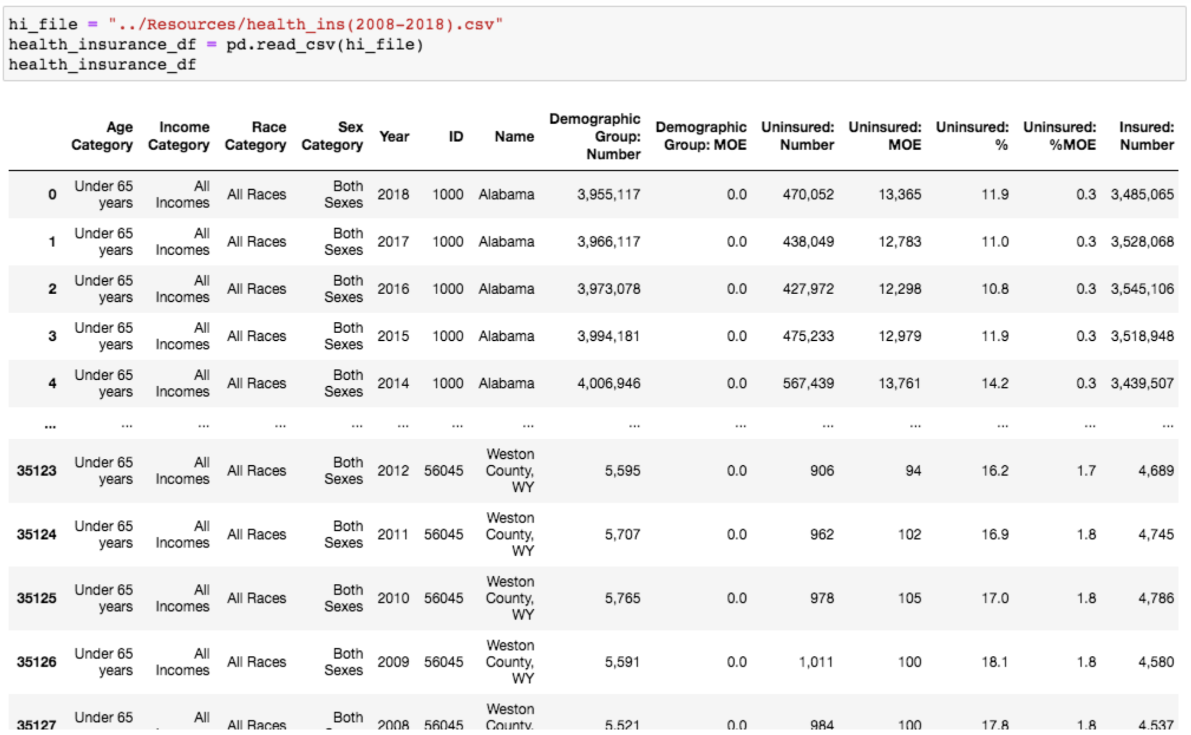
```
# Merge the two datasets
gdp_df = pd.merge(gdp_df2,gdp_df1, on=["County","State","County_State"])

gdp_df.head()
```

	County	State	County_State	gdp_2018	gdp_2017	gdp_2016	gdp_2015	gdp_2014	gdp_2013	gdp_2012	gdp_2011	gdp_2010	gdp_2009	gdp_2008
0	Autauga	AL	Autauga, AL	6.9	-4.8	0.6	10.4	3.0	-2.0	13.1	10.4	6.3	7.5	-8.1
1	Baldwin	AL	Baldwin, AL	6.9	2.2	5.0	6.1	3.1	4.9	5.3	2.3	3.9	-3.6	-3.7
2	Barbour	AL	Barbour, AL	7.4	2.6	-1.5	2.5	-3.9	9.6	-1.1	-2.5	3.0	1.1	-4.3
3	Bibb	AL	Bibb, AL	5.9	1.1	1.7	-0.9	0.9	4.0	0.8	3.4	8.7	-1.7	2.4
4	Blount	AL	Blount, AL	6.3	5.8	-3.7	2.0	-0.4	6.6	5.1	-1.1	0.2	-1.2	1.0

Figure 5 GDP-Data: Merging

Health Insurance Coverage Dataset:

1) Downloaded and loaded one csv file which contained all the data

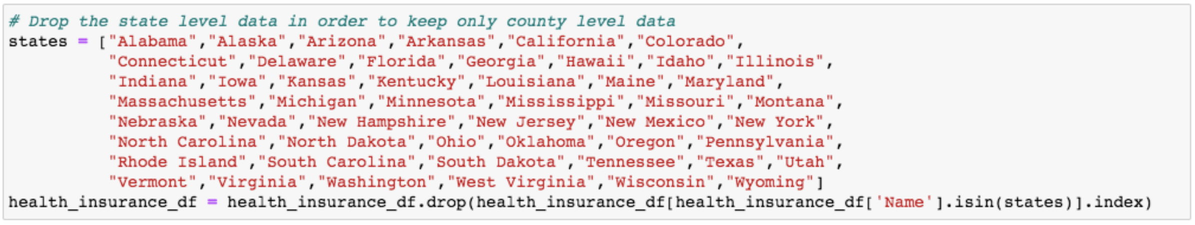


```
hi_file = "../Resources/health_ins(2008-2018).csv"
health_insurance_df = pd.read_csv(hi_file)
health_insurance_df
```

	Age Category	Income Category	Race Category	Sex Category	Year	ID	Name	Demographic Group: Number	Demographic Group: MOE	Uninsured: Number	Uninsured: MOE	Uninsured: %	Uninsured: %MOE	Insured: Number
0	Under 65 years	All Incomes	All Races	Both Sexes	2018	1000	Alabama	3,955,117	0.0	470,052	13,365	11.9	0.3	3,485,065
1	Under 65 years	All Incomes	All Races	Both Sexes	2017	1000	Alabama	3,966,117	0.0	438,049	12,783	11.0	0.3	3,528,068
2	Under 65 years	All Incomes	All Races	Both Sexes	2016	1000	Alabama	3,973,078	0.0	427,972	12,298	10.8	0.3	3,545,106
3	Under 65 years	All Incomes	All Races	Both Sexes	2015	1000	Alabama	3,994,181	0.0	475,233	12,979	11.9	0.3	3,518,948
4	Under 65 years	All Incomes	All Races	Both Sexes	2014	1000	Alabama	4,006,946	0.0	567,439	13,761	14.2	0.3	3,439,507
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
35123	Under 65 years	All Incomes	All Races	Both Sexes	2012	56045	Weston County, WY	5,595	0.0	906	94	16.2	1.7	4,689
35124	Under 65 years	All Incomes	All Races	Both Sexes	2011	56045	Weston County, WY	5,707	0.0	962	102	16.9	1.8	4,745
35125	Under 65 years	All Incomes	All Races	Both Sexes	2010	56045	Weston County, WY	5,765	0.0	978	105	17.0	1.8	4,786
35126	Under 65 years	All Incomes	All Races	Both Sexes	2009	56045	Weston County, WY	5,591	0.0	1,011	100	18.1	1.8	4,580
35127	Under 65	All	All Races	Both	2008	56045	Weston County.	5.521	0.0	984	100	17.8	1.8	4.537

Figure 6 Health-Insurance-Data: Reading CSV

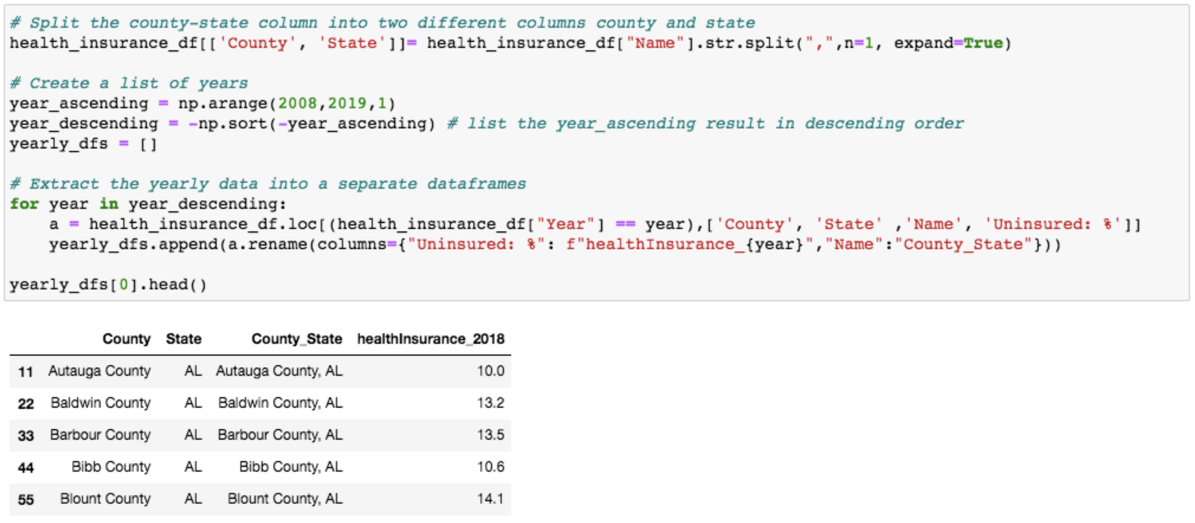
2) Dropped the state-level data



```
# Drop the state level data in order to keep only county level data
states = ["Alabama","Alaska","Arizona","Arkansas","California","Colorado",
          "Connecticut","Delaware","Florida","Georgia","Hawaii","Idaho","Illinois",
          "Indiana","Iowa","Kansas","Kentucky","Louisiana","Maine","Maryland",
          "Massachusetts","Michigan","Minnesota","Mississippi","Missouri","Montana",
          "Nebraska","Nevada","New Hampshire","New Jersey","New Mexico","New York",
          "North Carolina","North Dakota","Ohio","Oklahoma","Oregon","Pennsylvania",
          "Rhode Island","South Carolina","South Dakota","Tennessee","Texas","Utah",
          "Vermont","Virginia","Washington","West Virginia","Wisconsin","Wyoming"]
health_insurance_df = health_insurance_df.drop(health_insurance_df[health_insurance_df['Name'].isin(states)].index)
```

Figure 7 Health-Insurance-Data: Dropping State-level Data

3) Extracted the yearly data into each data frame

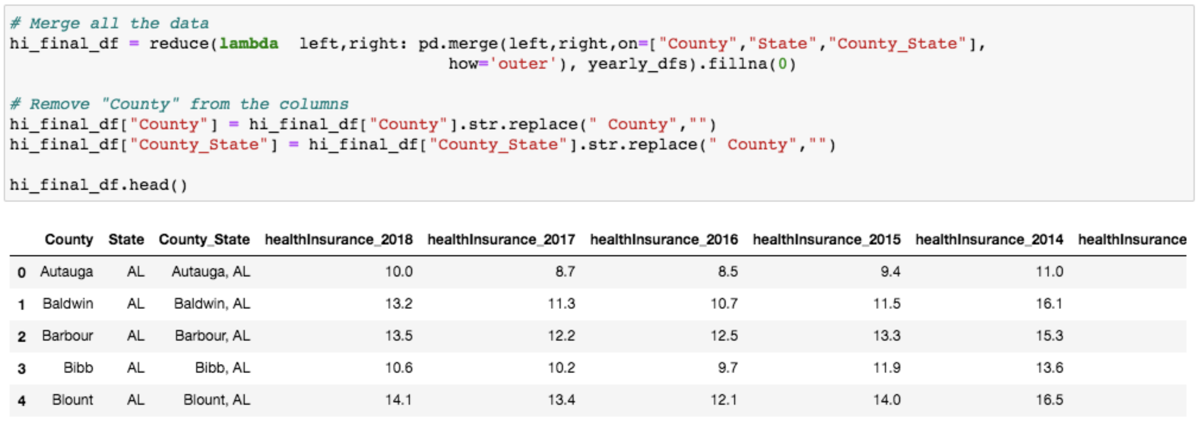


```
# Split the county-state column into two different columns county and state
health_insurance_df[['County', 'State']]= health_insurance_df["Name"].str.split(",",n=1, expand=True)

# Create a list of years
year_ascending = np.arange(2008,2019,1)
year_descending = -np.sort(-year_ascending) # list the year_ascending result in descending order
yearly_dfs = []

# Extract the yearly data into a separate dataframes
for year in year_descending:
    a = health_insurance_df.loc[(health_insurance_df["Year"] == year),['County', 'State' ,'Name', 'Uninsured: %']]
    yearly_dfs.append(a.rename(columns={"Uninsured: %": f"healthInsurance_{year}","Name":"County_State"}))

yearly_dfs[0].head()
```

	County	State	County_State	healthInsurance_2018
11	Autauga County	AL	Autauga County, AL	10.0
22	Baldwin County	AL	Baldwin County, AL	13.2
33	Barbour County	AL	Barbour County, AL	13.5
44	Bibb County	AL	Bibb County, AL	10.6
55	Blount County	AL	Blount County, AL	14.1

Figure 8 Health-Insurance-Data: Extracting Yearly Data

4) Replaced NA values with zero
5) Split the county-state column into two different columns county and state.
6) Renamed the columns
7) Merged all the annual datasets



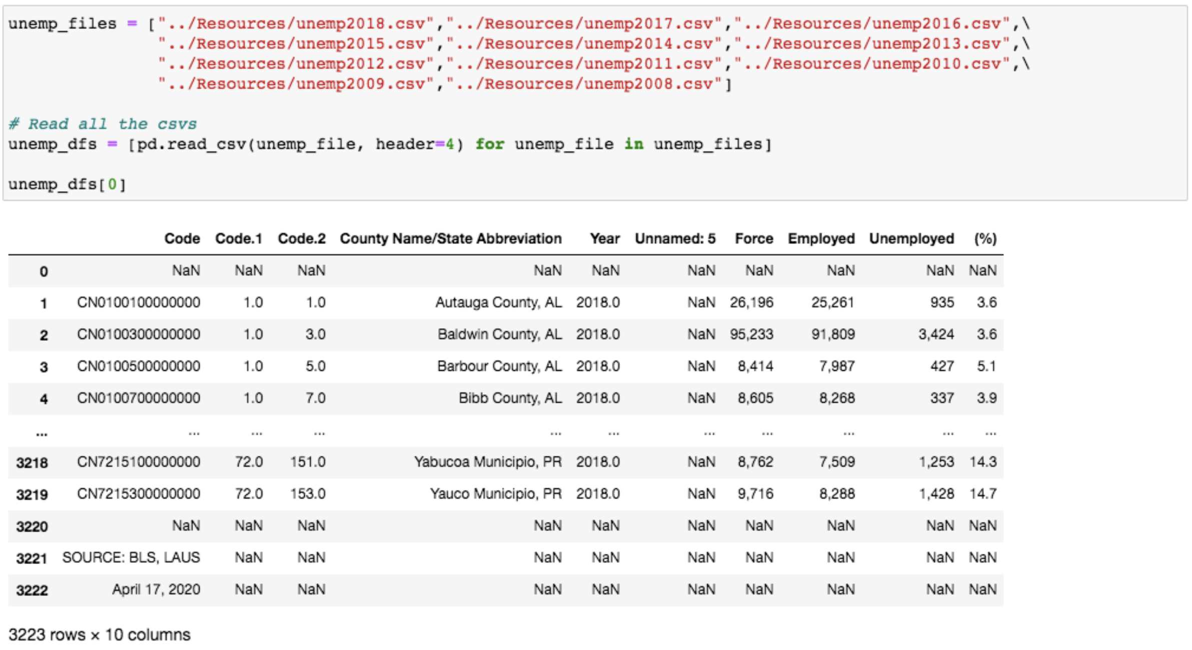
```
# Merge all the data
hi_final_df = reduce(lambda  left,right: pd.merge(left,right,on=["County","State","County_State"],
                                            how='outer'), yearly_dfs).fillna(0)

# Remove "County" from the columns
hi_final_df["County"] = hi_final_df["County"].str.replace(" County","")
hi_final_df["County_State"] = hi_final_df["County_State"].str.replace(" County","")

hi_final_df.head()
```

	County	State	County_State	healthInsurance_2018	healthInsurance_2017	healthInsurance_2016	healthInsurance_2015	healthInsurance_2014	healthInsurance
0	Autauga	AL	Autauga, AL	10.0	8.7	8.5	9.4	11.0	
1	Baldwin	AL	Baldwin, AL	13.2	11.3	10.7	11.5	16.1	
2	Barbour	AL	Barbour, AL	13.5	12.2	12.5	13.3	15.3	
3	Bibb	AL	Bibb, AL	10.6	10.2	9.7	11.9	13.6	
4	Blount	AL	Blount, AL	14.1	13.4	12.1	14.0	16.5	

Figure 9 Health-Insurance-Data: Merging and Renaming

Unemployment Dataset:

1) Downloaded and loaded 11 csv files for annual data.



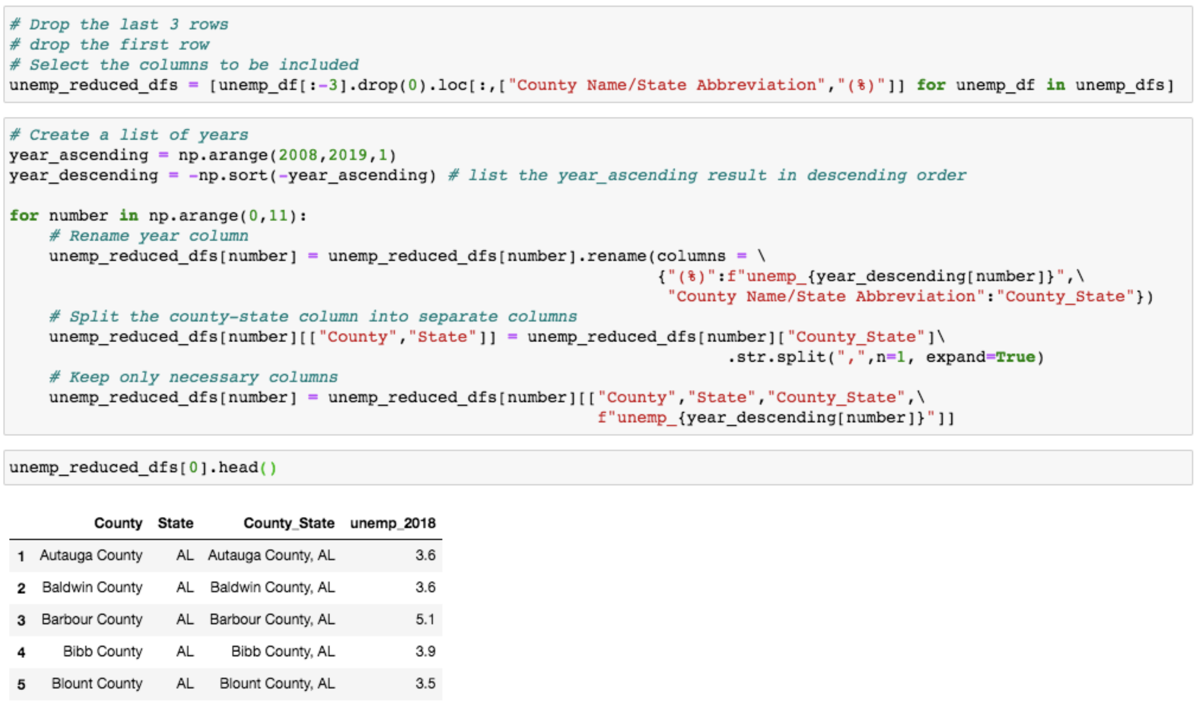
```
unemp_files = ["../Resources/unemp2018.csv","../Resources/unemp2017.csv","../Resources/unemp2016.csv",\
               "../Resources/unemp2015.csv","../Resources/unemp2014.csv","../Resources/unemp2013.csv",\
               "../Resources/unemp2012.csv","../Resources/unemp2011.csv","../Resources/unemp2010.csv",\
               "../Resources/unemp2009.csv","../Resources/unemp2008.csv"]

# Read all the csvs
unemp_dfs = [pd.read_csv(unemp_file, header=4) for unemp_file in unemp_files]

unemp_dfs[0]
```

	Code	Code.1	Code.2	County Name/State Abbreviation	Year	Unnamed: 5	Force	Employed	Unemployed	(%)
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	CN0100100000000	1.0	1.0	Autauga County, AL	2018.0	NaN	26,196	25,261	935	3.6
2	CN0100300000000	1.0	3.0	Baldwin County, AL	2018.0	NaN	95,233	91,809	3,424	3.6
3	CN0100500000000	1.0	5.0	Barbour County, AL	2018.0	NaN	8,414	7,987	427	5.1
4	CN0100700000000	1.0	7.0	Bibb County, AL	2018.0	NaN	8,605	8,268	337	3.9
...	...	...	...	...	...	...	...	...	...	...
3218	CN7215100000000	72.0	151.0	Yabucoa Municipio, PR	2018.0	NaN	8,762	7,509	1,253	14.3
3219	CN7215300000000	72.0	153.0	Yauco Municipio, PR	2018.0	NaN	9,716	8,288	1,428	14.7
3220	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3221	SOURCE: BLS, LAUS	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3222	April 17, 2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

3223 rows × 10 columns

Figure 10 Unemployment-Data: Reading CSV

2) Dropped unnecessary columns and rows.
3) Replaced NA values with zero.
4) Split the county-state column into two different columns county and state.



```
# Drop the last 3 rows
# drop the first row
# Select the columns to be included
unemp_reduced_dfs = [unemp_df[:-3].drop(0).loc[:,["County Name/State Abbreviation","(%)"]] for unemp_df in unemp_dfs]
```

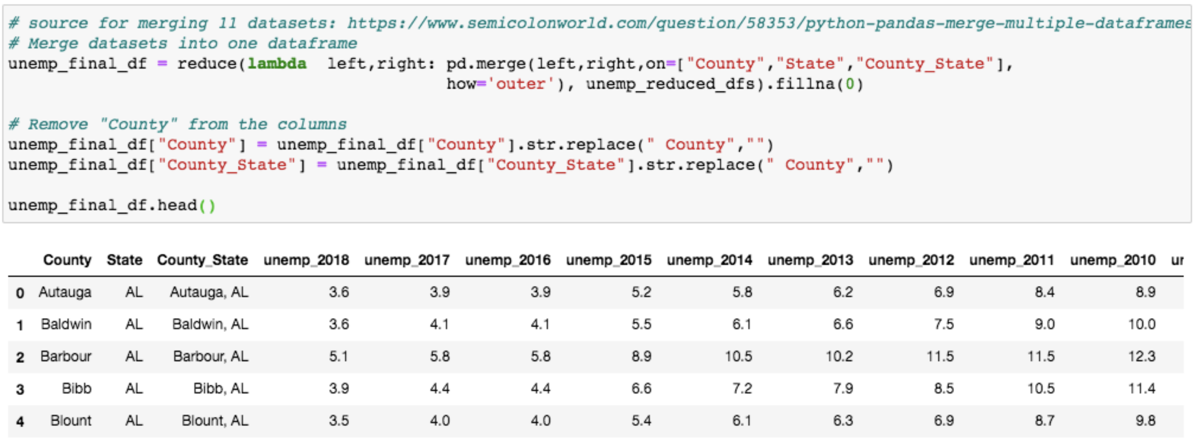
```
# Create a list of years
year_ascending = np.arange(2008,2019,1)
year_descending = -np.sort(-year_ascending) # list the year_ascending result in descending order

for number in np.arange(0,11):
    # Rename year column
    unemp_reduced_dfs[number] = unemp_reduced_dfs[number].rename(columns = \
                                                                {"(%)":f"unemp_{year_descending[number]}",\
                                                                 "County Name/State Abbreviation":"County_State"})

    # Split the county-state column into separate columns
    unemp_reduced_dfs[number][["County","State"]] = unemp_reduced_dfs[number]["County_State"]\
                                                              .str.split(",",n=1, expand=True)

    # Keep only necessary columns
    unemp_reduced_dfs[number] = unemp_reduced_dfs[number][["County","State","County_State",\
                                                           f"unemp_{year_descending[number]}"]]
```

```
unemp_reduced_dfs[0].head()
```

	County	State	County_State	unemp_2018
1	Autauga County	AL	Autauga County, AL	3.6
2	Baldwin County	AL	Baldwin County, AL	3.6
3	Barbour County	AL	Barbour County, AL	5.1
4	Bibb County	AL	Bibb County, AL	3.9
5	Blount County	AL	Blount County, AL	3.5

Figure 11 Unemployment-Data: Dropping, Renaming, Extracting

5) Merged datasets and renamed the columns.



```
# source for merging 11 datasets: https://www.semicolonworld.com/question/58353/python-pandas-merge-multiple-dataframes
# Merge datasets into one dataframe
unemp_final_df = reduce(lambda  left,right: pd.merge(left,right,on=["County","State","County_State"],
                                            how='outer'), unemp_reduced_dfs).fillna(0)

# Remove "County" from the columns
unemp_final_df["County"] = unemp_final_df["County"].str.replace(" County","")
unemp_final_df["County_State"] = unemp_final_df["County_State"].str.replace(" County","")

unemp_final_df.head()
```

	County	State	County_State	unemp_2018	unemp_2017	unemp_2016	unemp_2015	unemp_2014	unemp_2013	unemp_2012	unemp_2011	unemp_2010	ur
0	Autauga	AL	Autauga, AL	3.6	3.9	3.9	5.2	5.8	6.2	6.9	8.4	8.9	
1	Baldwin	AL	Baldwin, AL	3.6	4.1	4.1	5.5	6.1	6.6	7.5	9.0	10.0	
2	Barbour	AL	Barbour, AL	5.1	5.8	5.8	8.9	10.5	10.2	11.5	11.5	12.3	
3	Bibb	AL	Bibb, AL	3.9	4.4	4.4	6.6	7.2	7.9	8.5	10.5	11.4	
4	Blount	AL	Blount, AL	3.5	4.0	4.0	5.4	6.1	6.3	6.9	8.7	9.8	

Figure 12 Unemployment-Data: Merging and Renaming

Loading

Data Storage:
We used a relational database (PostgreSQL) to store and link the three datasets using County_State column as our common identifier. We used PostgreSQL over MongoDB as PostgreSQL stores data in table and supports relational databases while MongoDB stores data like documents and does not support well-defined relationships. Using MongoDB may end up with having a lot of duplicate data which might result in having corrupted data.

Data Loading:
Finally, we created a connection from SQL database, and loaded the final data in a Jupyter Notebook using *SQLAlchemy*.