SpringBoard Data Science Career Track
Capstone Project 1

Predicting Unemployment Rates

Merle Glick

Table of Contents

I. Introduction

Unemployment rates vary from one area to another, and they also vary across time. In 2008 and 2009, the United States economy experienced a recession, often referred to as the Great Recession. During this time period, unemployment rates increased dramatically across most of the country. On average, the unemployment rate almost doubled, although some areas were affected much more than others. Other parts of the world were affected as well, but for this report, I will focus on the unemployment rates in the United States, and various factors that are related to unemployment.

The goal is to be able to predict the unemployment rate for a specific area in the United States, based on population, education level, median income, and other factors. Knowing how these other factors are related to unemployment rates will make it easier to determine how to solve the unemployment problem in this country, and possibly how to prevent a major crisis, such as was experienced in 2008 – 2009.

To address this problem, I gathered data from several different sites, including the USDA government site and the United States Census Bureau. I used the Python programming language, with its many statistical tools and libraries, to analyze the data and build a model to predict the unemployment rates. I looked at past trends, and investigated the correlation between various factors that may affect unemployment rates.

II. Data Acquisition and Wrangling

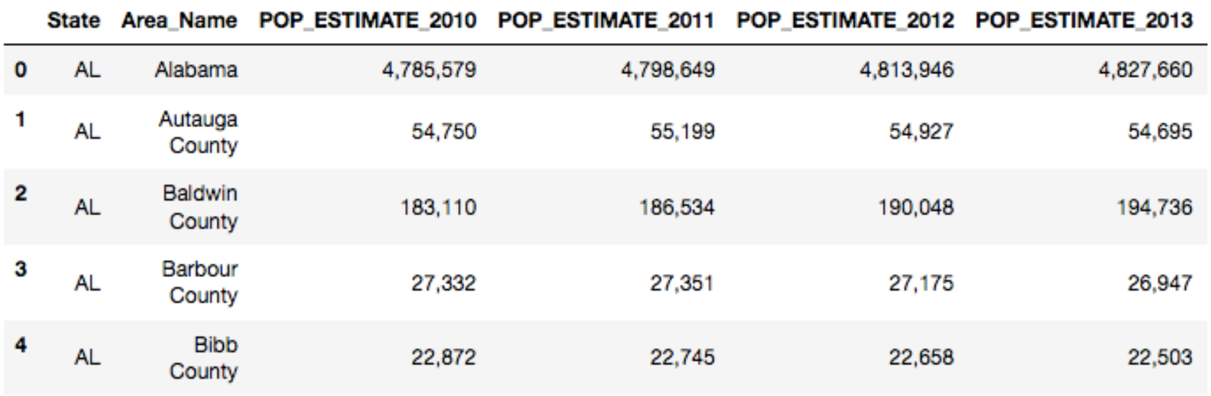
The majority of the data used for this project was obtained from the USDA government site and contains unemployment rates, population estimates, median household income, educational attainment, and other data for all states in the U.S. It also contains some data for all counties within each state. I was able to find additional income data from the United State Census Bureau site. All the data is contained in downloadable excel files. Since the data was in several different files, I had to merge the desired data from each one. The data also required some cleaning and reformatting so that it could be plotted and analyzed. There are many variables contained in this data set, so I had to determine which ones are needed to answer the questions I wanted to answer.

The first step was to import the data from the csv files into a Pandas dataframe. This makes it easy to look at the overall structure of the data, and the format of the columns. In the Pandas dataframe structure, each column is a variable, and each row contains a value for each variable. The Pandas library has many useful tools that allow us to determine the datatype of the values in each column, find missing values, and many other processes. Some of the data files contained data from U.S. territories in addition to the state data. I wanted only data for the 50 states and the District of Columbia, so I used Pandas to drop all rows containing data for the territories.

After extracting the desired data, I investigated the individual columns. Most of the columns contained numeric data, but it was formatted as ‘string’ type data, which means it cannot be used to perform mathematical calculations. To be able to use the data for analysis and plotting, it needs to be integers or ‘floating point’ decimal numbers. Also, some of the values included commas and/or the dollar sign, which I needed to eliminate so that the values are recognized as numbers by the computer. To clean up these values, I defined a function called ‘remove_non_digits()’, that takes an entire column as input, then removes all non-digits from each value and converts the value to a ‘float’, which can be used for the analysis and plotting that needs to be done. The function returns a series containing these new values, which I then put into the dataframe in place of the original column. I used this function several times throughout the process of cleaning the data.

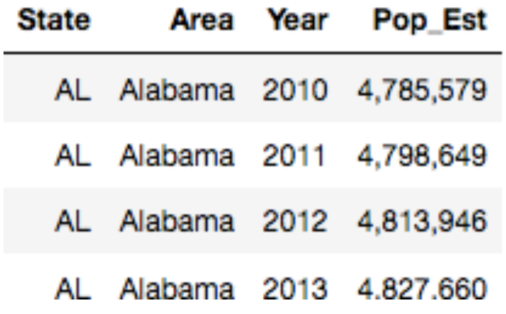
Another issue I had was that one of the data files contained a column with only the county names, while another data set had the county and state in the column. For example, one file had ‘Autauga County’ while the other had ‘Autauga County, Alabama’. There was a separate column containing the state name, so I did not need the state name in the ‘County’ column. Also, I needed these columns to contain the same values in both data sets so that I could merge the data sets for further analysis. To clean this column, I used a built-in function that splits each string on the comma. I then saved only the first part (the county name) so that the data was in the same format as the other dataframe.

As part of the analysis, I wanted to compart different regions of the U.S. (Northeast, South, Midwest, and West). I created a list of the states in each region, then used this to subset the dataframe, and created a column with the region. I also wanted a column with the year, so that I could use this as a feature. The years were contained in some of the column names, as shown below. However, I wanted one column with only the years, and a second column with the population values.



	State	Area_Name	POP_ESTIMATE_2010	POP_ESTIMATE_2011	POP_ESTIMATE_2012	POP_ESTIMATE_2013
0	AL	Alabama	4,785,579	4,798,649	4,813,946	4,827,660
1	AL	Autauga County	54,750	55,199	54,927	54,695
2	AL	Baldwin County	183,110	186,534	190,048	194,736
3	AL	Barbour County	27,332	27,351	27,175	26,947
4	AL	Bibb County	22,872	22,745	22,658	22,503

To achieve this result, I used the 'stack()' function in pandas to combine all the population columns into two columns, where the first contains the previous column titles and the second contains the values from the previous columns. I then used the previously mentioned function, remove_non_digits(), to clean the 'Year' column so that it contained only the year. The result is shown below. I used a similar technique for the Median Income data, and the Education data, which were contained in separate files.

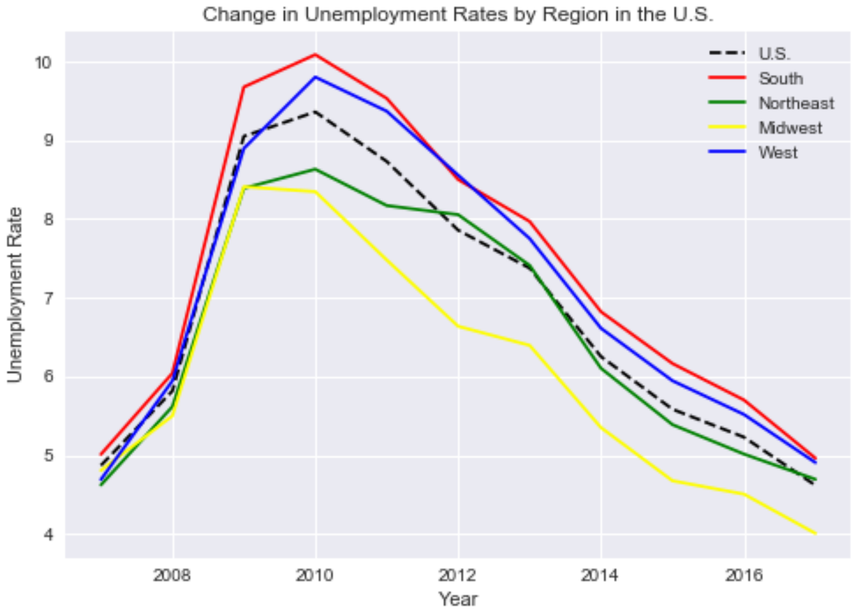


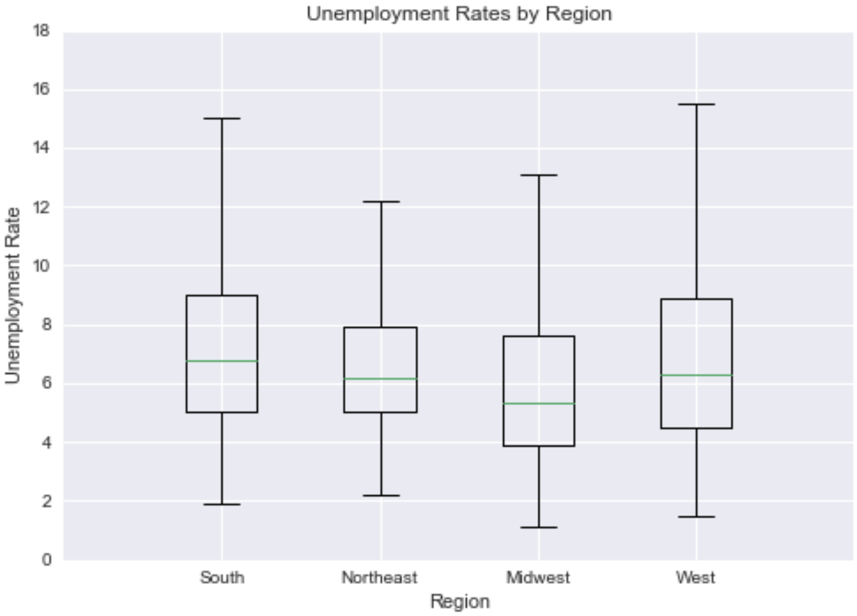
State	Area	Year	Pop_Est
AL	Alabama	2010	4,785,579
AL	Alabama	2011	4,798,649
AL	Alabama	2012	4,813,946
AL	Alabama	2013	4.827.660

The next step was to combine all this data into a single dataframe to be used for analysis. I exported each file, containing unemployment rates, education data, population data, and median income data. I then imported each file as a new dataframe, and used the Pandas 'merge' function to combine all the dataframes into one. After merging the data, I still had more data wrangling and cleaning to do before I could begin the analysis. I had some missing values that would need to be imputed so that the columns could be used in the machine learning algorithms. For now, I filled in these value with 'NaN'. During the analysis, I would determine the best way to deal with these values. I now had a dataframe with 15 columns and just over 35,000 rows.

III. Exploratory Data Analysis

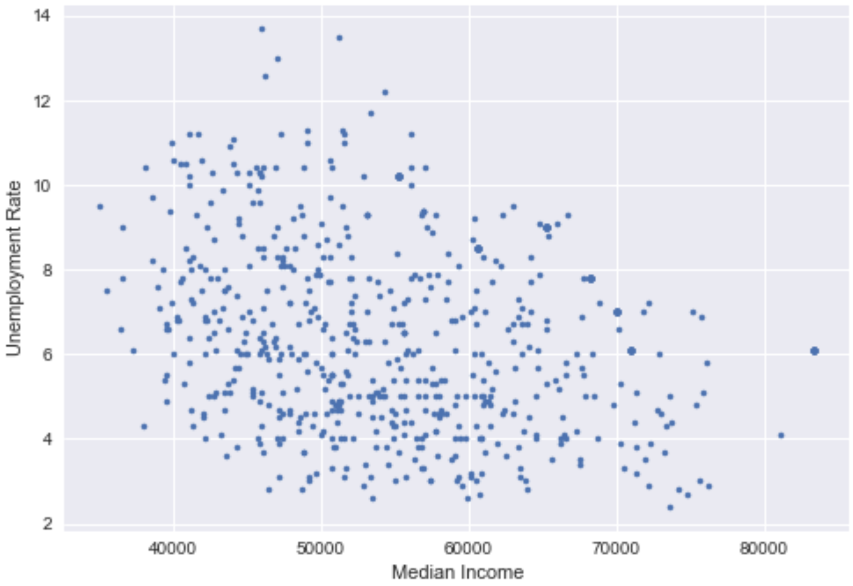
I wanted to see how unemployment rates had changed over the period from 2007 to 2017, so I constructed a time series plot for each region in the U.S., and also for the entire U.S. I noticed a very similar trend in each graph, a sharp increase from 2007 to 2010 (during the economic recession), then a gradual decline in the years since then. Further analysis indicated that overall there has been a slightly negative trend in unemployment rates. We can also see the difference between the regions of the U.S. In the years immediately following the recession, the differences between the regions are obvious. As the rates continue to decrease, the graphs are closer together, so there is not as much of a distinction between the regions in these years. It looks like the South and the West were affected the most by the economic recession, and in general, these regions have higher unemployment rates then the Northeast and the Midwest. The time-series plot is shown below, along with boxplots comparing the four regions.



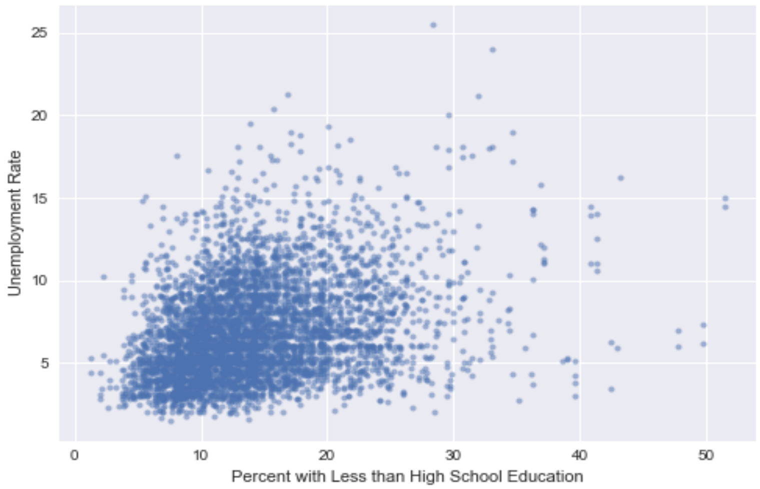


I also wanted to know how the individual variables might be correlated with each other, and especially how they were correlated with Unemployment Rates. I used the pairplots function in the Seaborn library to construct scatterplots of all possible pairs of variables in the dataset. This would allow me to see possible correlations between any of the variables, so that I can determine which ones will be most useful in predicting Unemployment Rates. Before constructing the plots, I had to deal with the missing values in the dataset, because the pairplots function will only work if there is no missing data. Since I had median income data for the states, but not for all the counties within the state, I filled in the county values with the median income for the state for that year. For the education data, I had estimates for the period from 2012 to 2016, so I used this same value for the other years. I had population data for 2011 through 2017. To estimate the population for the other years, I calculated the rate of change from 2011 to 2012, then used that rate to estimate the population for each year prior to 2011. There were also several categorical variables that might be useful for predicting unemployment rates. However, many of the machine learning algorithms can only deal with numerical values, so I converted these to numerical values, so that each value of the variable is represented by a numerical code.

After looking at the pairplots for all the variables, I constructed a few individual plots to investigate the relationships further. The plot showing the relationship between Median Income and Unemployment Rates is shown below. I used only the data from the states for this plot. There is a noticeable downward trend, which indicates that states with higher median incomes generally have lower unemployment rates. So it appears that median income might be a good predictor to include in the model.

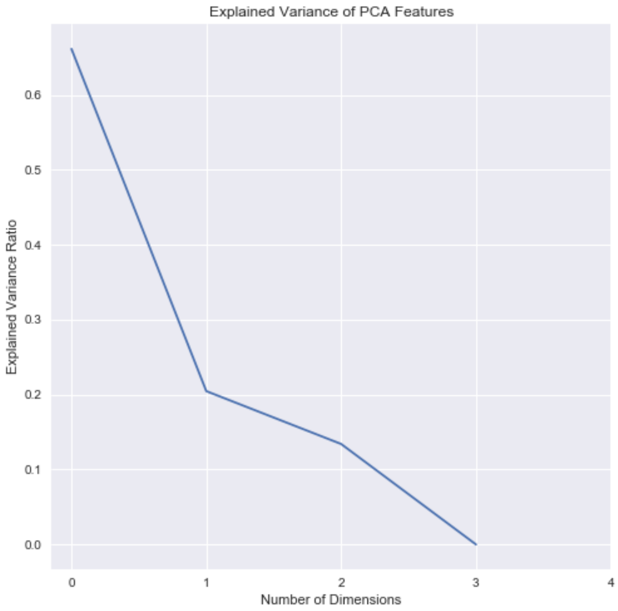


I was also interested in the correlation between education level and unemployment rates. The education level variables appeared to be more strongly correlated with unemployment rates than most of the other variables. One example is shown in the plot below, which shows the relationship between ‘Percentage with Less Than a High School Education’ and ‘Unemployment Rate’. I used a random sample of 5000 of the data points to construct this plot, so that the plot is easier to interpret. There is an obvious upward trend, which indicates that areas where a higher percentage of the population has less than a high school education tend to have higher unemployment rates.



IV. Machine Learning

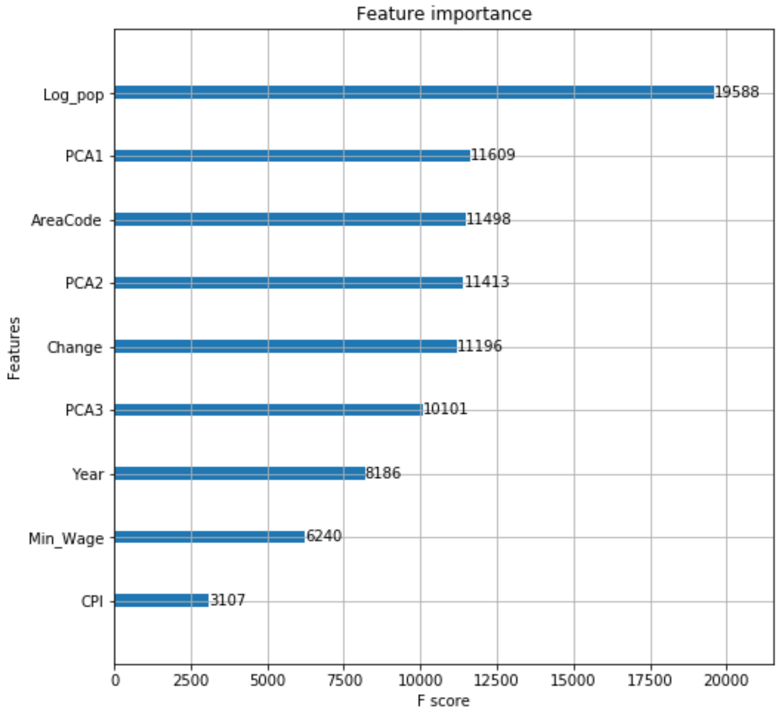
The next step is preprocessing the data for the machine learning algorithms. There are four predictor variables related to education: ‘Percent with Less than High School’, ‘Percent with a High School Education’, ‘Percent with Some College’, and ‘Percent with a Bachelor’s Degree’. From the previous analysis, it is obvious that these features are strongly correlated with each other. Because of this, I used Principal Component Analysis (PCA) to analyze these features and reduce the number of features used. I plotted the explained variance of the features (shown below), and it was obvious that almost all the variance in the data was explained by three of the features. These three features will be included in the model.



Before fitting a model, I scaled all the variables, and printed the first few rows of the dataframe to make sure it was formatted in the way I wanted it. I then split the data into training and test sets. The training set is used to 'train' the model, and then the test set is used to determine how well the model performs. The goal is to build a model that will perform well on new data that it has not seen before, so we keep some of the data as the test set, and do not use it until the model has been built.

I used a linear model as a first attempt. To test the performance of each model, I used the mean absolute error, which measures how close the predicted values are to the actual values. With the linear model, I got a mean absolute error of 0.07. The linear model can be used as a benchmark to compare the results of other models.

Next I used XGBoost, a machine learning algorithm which uses gradient boosting, and has many parameters that can be tuned to get optimal results. XGBoost uses decision trees to build a model, and outperforms many other machine learning algorithms. I used a randomized search algorithm from the sklearn library to tune the hyperparameters and build the best model. After fitting this model to the data, I got a mean absolute error of 0.023, which is significantly lower than the error with the linear model. I then constructed a plot showing the feature importance for this model. From this plot, we can see that the population was the most important feature in predicting unemployment using this model.



I also used CatBoost, another machine learning algorithm, to fit a model. This library derives its name from its ability to use categorical variables, and from the fact that it uses boosting. I again used the randomized search to tune the hyperparameters, then fit a model using the optimal parameters found. After adjusting the parameter search several times and fitting the model, I was able to get a mean absolute error of 0.033, which is slightly higher than what I got with XGBoost, but still much better than the linear model.

V. Conclusion

By using these optimized machine learning algorithms, I was able to build a good model to predict unemployment rates. It might be possible to improve this model slightly by adjusting some of the hyperparameters, or by adding additional data. However, this could also increase the complexity of the model, and slow down the algorithm. I believe this model gives us a good prediction without being too complex. This model could be useful for agencies that are working to reduce unemployment rates. If they have information about the factors that could affect unemployment rates, they can be more effective in implementing strategies to minimize unemployment rates.