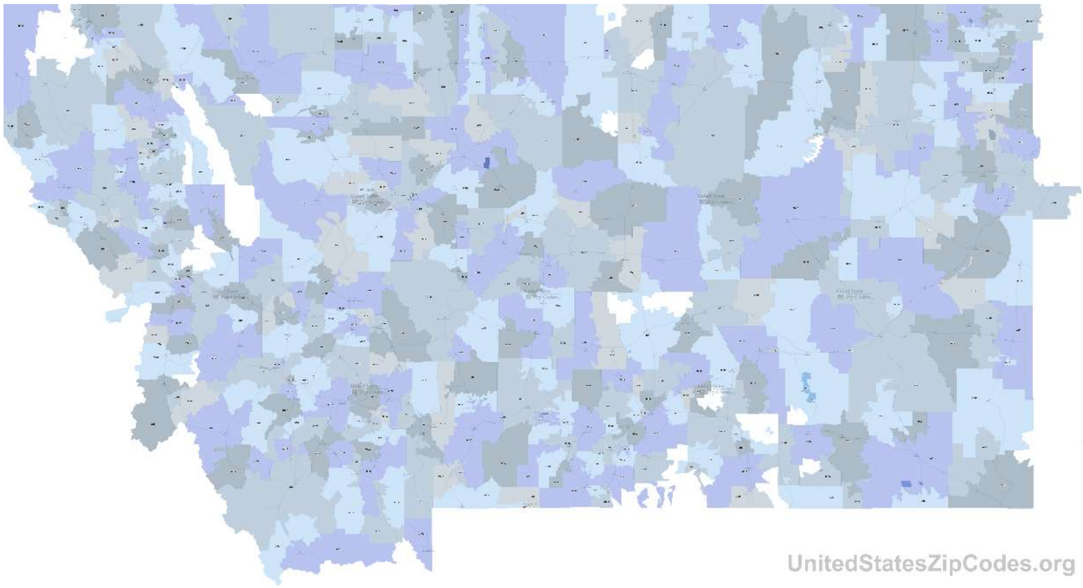
Predicting People Born Out of State in each Zip code

Data Mining 347

Bruce Clark

5/5/2020



UnitedStatesZipCodes.org

Summarizes the problem

In Montana, there is a stigma against people who have moved to Montana from other states. Montanans are very protective of their state and do not want to see it overpopulated. Many are also distrustful of nonnative Montanans and see places like Bozeman as a place that is caught up in industry. As a result, many Montanans look down on people from Bozeman and have even taken to calling the town Boz Angles as they believe it resembles a large city.

I would like to see if we could predict the number of people in every zip code who where born out of state (OOS) by looking at several attributes that describe that zip codes that we get through publicly available census data. I will then compare the predictions with the actual values given by the census data and will be able to determine how accurate my algorithm was.

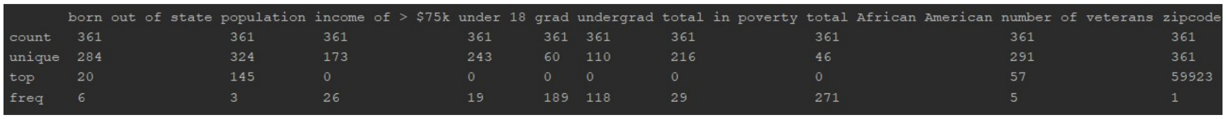
Gathering the Data

In order to get the data, I must interact with the API. There is a special API built for developers where you can query and get census data that you are after. First, I had to request a key here: https://www.census.gov/developers/. Then, I had to build my query to breakdown all the attributes that I would be asking for in terms of zip codes in Montana. I did this by finding a list of every zip code in Montana and building that into my query. Next, I selected a number of variables from this list: https://api.census.gov/data/2017/acs/acs5/variables.html, incorporated my selected variables into my query, and was able to pull json data back from the request. I selected my variables with a great deal of thought, after hours of exploring many of the available variables (there were over a couple thousand to choose from) and what they represented. After careful consideration, I choose the following variables to pull down from the API: number of people born out of state, total population, total with income greater than 75K, total under the age of 18, total graduate students, total undergraduate students, total in poverty, total African American, number of veterans, and the zip code. These are referenced throughout the report as the following: 'born out of state', 'population', "income of > $75k", 'under 18', 'grad', 'undergrad', 'total in poverty', 'total African American', 'number of veterans', 'zipcode'.

Data Description

There is no missing data in my data set and all attributes are numerical. Right now, I have 361 zip codes that I am evaluating in Montana, out of the total 404 zip codes. (This mean that 43 zip codes in Montana have no census data available to them)

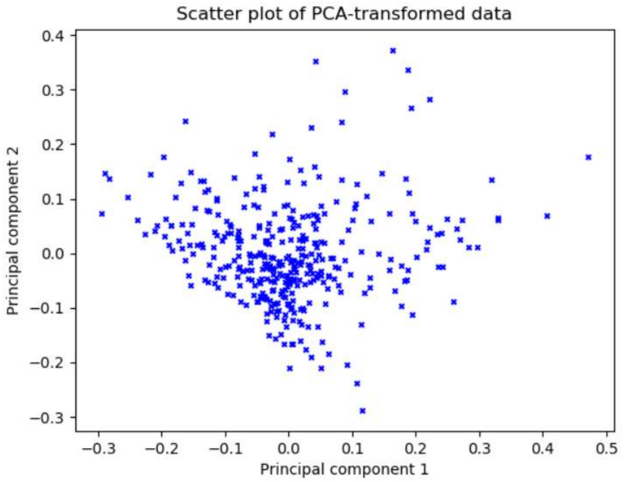
Here is a pandas' description of my data:



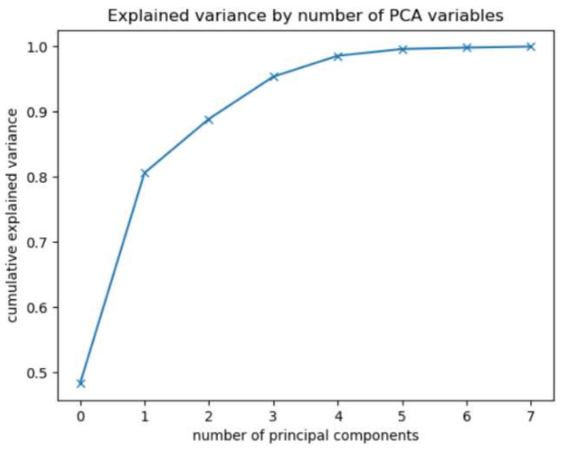
```
       born out of state population income of > $75k under 18 grad undergrad total in poverty total African American number of veterans zipcode
count   361              361        361              361      361  361       361              361                    361                361
unique  284              324        173              243      60   110       216              46                     291                361
top     20               145        0                0        0    0         0                0                      57                 59923
freq    6                3          26               19       189  118       29               271                    5                  1
```

Data Preprocessing

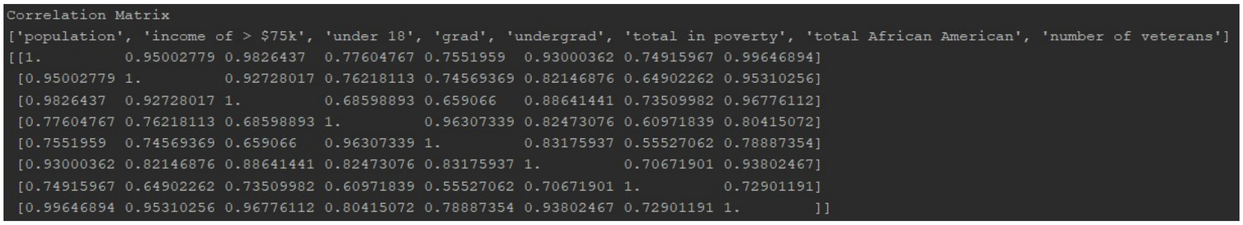
Figure 1



After I gathered the data, I started experimenting with different methods of preprocessing. One of the first things I wanted to do was visualize my data. To do this, I used the PCA algorithm to condense my data down to two principle components. I then plotted these two components on they scatter plot (Figure 1). Next, I wanted to explore PCA and how much variability I would be losing by reducing the dimensionality of my data set. To do this I created a rolling sum of the amount of variance caught by adding dimensions and plotted that on a line chart (Explained variance by number of PCA variables). This shows that we can retain most of our variability an reduce our data set to five components. However, I am was hesitant to do so as seven variables is not a ton in the first place. So, I decided I would experiment with it a little farther along in the process.



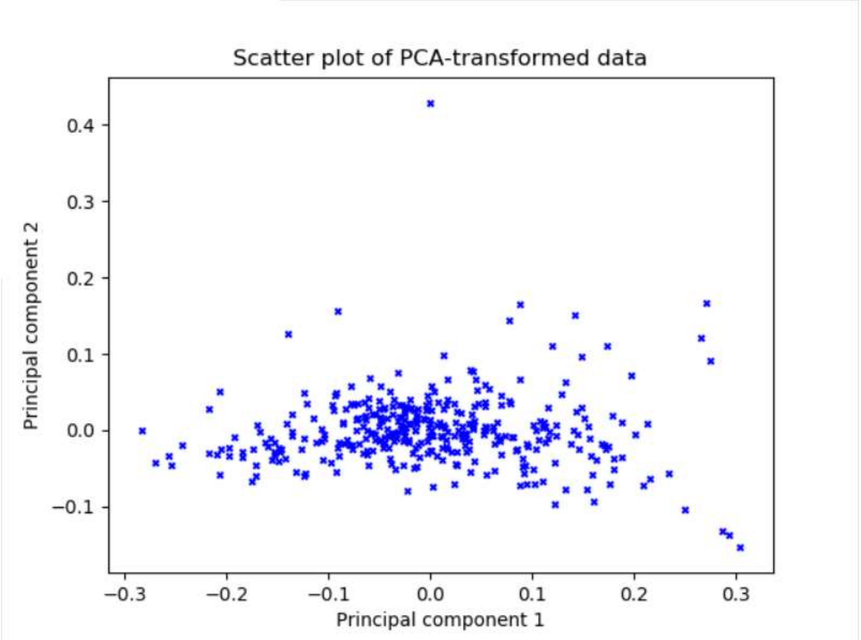
I also wanted to look at the correlation matrix so I could see if two variables were incredibly correlated, then I would be more likely to combined them with the PCA algorithm. Here are the results of the correlation matrix:



```
Correlation Matrix
['population', 'income of > $75k', 'under 18', 'grad', 'undergrad', 'total in poverty', 'total African American', 'number of veterans']
[[1.         0.95002779 0.9826437  0.77604767 0.7551959  0.93000362 0.74915967 0.99646894]
 [0.95002779 1.         0.92728017 0.76218113 0.74569369 0.82146876 0.64902262 0.95310256]
 [0.9826437  0.92728017 1.         0.68598893 0.659066   0.88641441 0.73509982 0.96776112]
 [0.77604767 0.76218113 0.68598893 1.         0.96307339 0.82473076 0.60971839 0.80415072]
 [0.7551959  0.74569369 0.659066   0.96307339 1.         0.83175937 0.55527062 0.78887354]
 [0.93000362 0.82146876 0.88641441 0.82473076 0.83175937 1.         0.70671901 0.93802467]
 [0.74915967 0.64902262 0.73509982 0.60971839 0.55527062 0.70671901 1.         0.72901191]
 [0.99646894 0.95310256 0.96776112 0.80415072 0.78887354 0.93802467 0.72901191 1.        ]]
```

A few of these values stick out to me. One, there seems to be a high level of correlation between the population and the number of veterans (0.996). Two, there is also a high level of correlation between population and the total number of people under 18 (0.9826). This tells me that the number of veterans and kids is equally distributed by population throughout the geographical locations of Montana.

Then, I did some more experimentation. I wanted to see if I could eliminate some variables in order to get our data looking more linear. My thoughts going into this were to use Naive Bayes (NB) or Simple Linear Regression (SLR), both of which are linear algorithms. By eliminating the “total number of people in poverty” and the “total number of African American people” variables, I was able to make the scatterplot look more linear.



This also changed my total variables from seven to five. Because I had already reduced the dimensionality of my data, I decided against using any further PCA algorithm on my data.

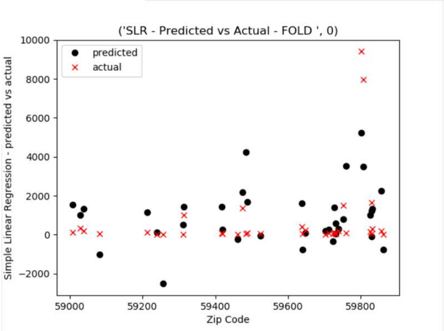
The Algorithms

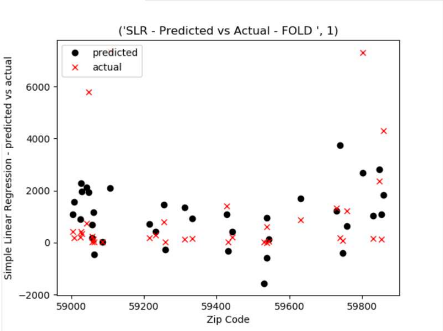
With my data processed and analyzed, it was time to start putting it to the test. I first wanted to test the data out with simple linear regression. I decided to perform ten-fold validation on my data set and record the r^2 (coefficient of determination). Here are the results of SLR for each fold:

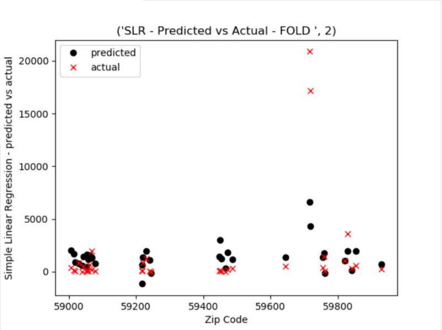
```
SLR - fold 1 r^2:  0.32
SLR - fold 2 r^2:  0.22
SLR - fold 3 r^2:  0.39
SLR - fold 4 r^2:  0.04
SLR - fold 5 r^2:  0.55
SLR - fold 6 r^2:  3.62
SLR - fold 7 r^2:  0.03
SLR - fold 8 r^2:  0.03
SLR - fold 9 r^2:  0.15
SLR - fold 10 r^2:  0.31
```

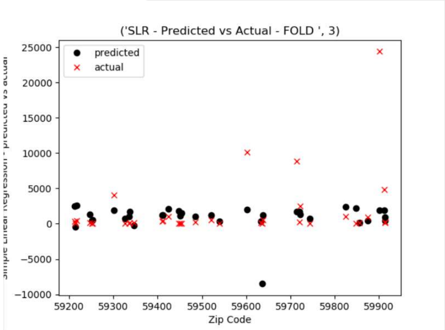
As you can see the results are mediocre with the average r^2 value being 0.566 which was skewed by the 3.62 value. Speaking of the 3.62 value, I am not sure if this is a legitimate value since r^2 values are supposed to range from -1 to 1, with values closer to 1 or -1 meaning the model works better and values closer to zero meaning the model is worse. Honestly, I am confused by this 3.62 and might want to check the values going in for the sixth fold.

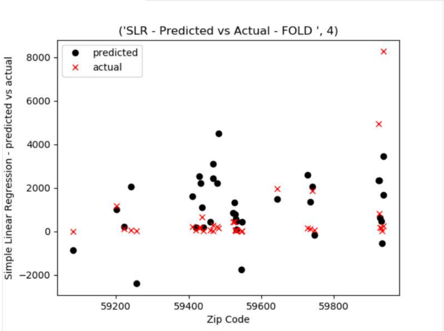
I then wanted to visualize these results. so, I plotted the results of each fold on a scatterplot with the predicted values and actual values of each zip code. Here are the results:

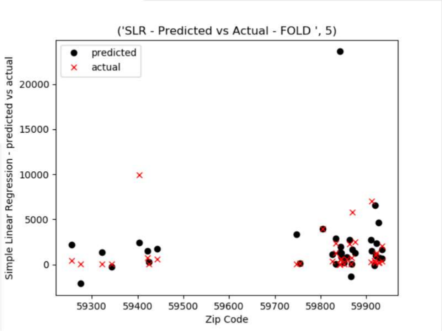

('SLR - Predicted vs Actual - FOLD ', 0)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
10000
8000
6000
4000
2000
0
−2000
59000
59200
59400
59600
59800

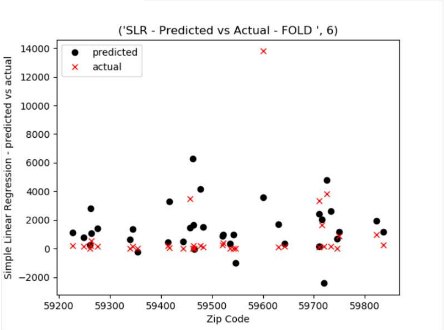

('SLR - Predicted vs Actual - FOLD ', 1)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
6000
4000
2000
0
−2000
59000
59200
59400
59600
59800

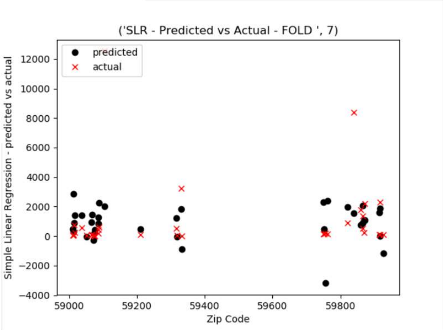

('SLR - Predicted vs Actual - FOLD ', 2)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
20000
15000
10000
5000
0
59000
59200
59400
59600
59800

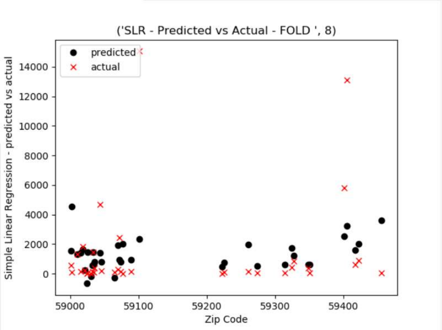

('SLR - Predicted vs Actual - FOLD ', 3)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
25000
20000
15000
10000
5000
0
−5000
−10000
59200
59300
59400
59500
59600
59700
59800
59900

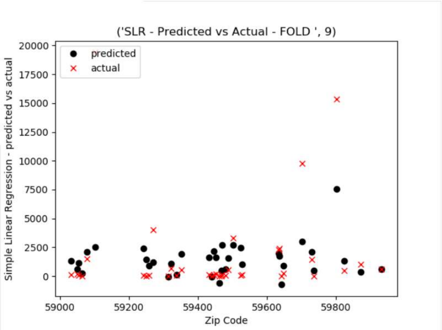

('SLR - Predicted vs Actual - FOLD ', 4)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
8000
6000
4000
2000
0
−2000
59200
59400
59600
59800


('SLR - Predicted vs Actual - FOLD ', 5)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
20000
15000
10000
5000
0
59300
59400
59500
59600
59700
59800
59900


('SLR - Predicted vs Actual - FOLD ', 6)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
14000
12000
10000
8000
6000
4000
2000
0
−2000
59200
59300
59400
59500
59600
59700
59800


('SLR - Predicted vs Actual - FOLD ', 7)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
12000
10000
8000
6000
4000
2000
0
−2000
−4000
59000
59200
59400
59600
59800


('SLR - Predicted vs Actual - FOLD ', 8)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
14000
12000
10000
8000
6000
4000
2000
0
59000
59100
59200
59300
59400

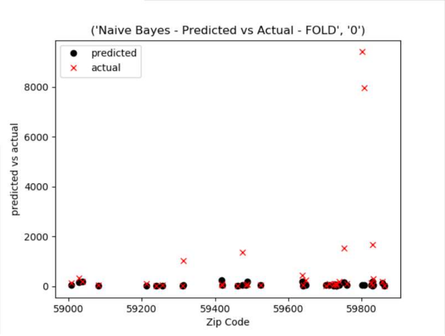

('SLR - Predicted vs Actual - FOLD ', 9)
predicted
actual
Simple Linear Regression - predicted vs actual
Zip Code
20000
17500
15000
12500
10000
7500
5000
2500
0
59000
59200
59400
59600
59800

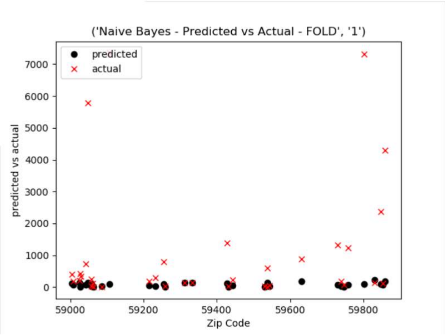
You can see here from the visualizations that simple linear regression was trying to fit the model as best it can. In some cases, it did better, and other cases it did worse. This led me to wonder if I had chosen a bad algorithm to match with my data. so, I decided to run a Gaussian Naive Bayes algorithm to see if the results were different.

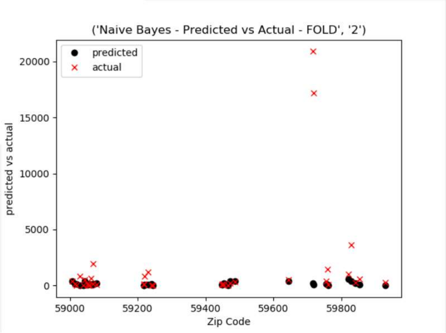
The accuracy function built into the equation was mean accuracy. However, upon examining the results, I am almost positive that the algorithm was only couting instances as accurate if they were able to exactly predict the number of people born out of state in each tested zip code. For this reason, the mean accuracies were incredibly low.

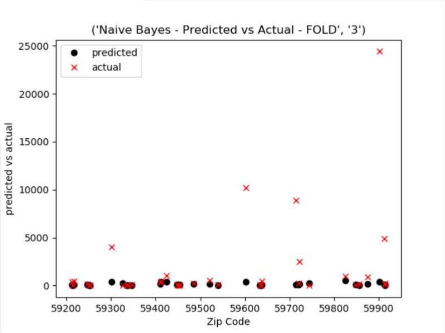
```
NB - fold 1 Mean Accuracy:  0.0
NB - fold 2 Mean Accuracy:  0.0
NB - fold 3 Mean Accuracy:  0.028
NB - fold 4 Mean Accuracy:  0.0
NB - fold 5 Mean Accuracy:  0.0
NB - fold 6 Mean Accuracy:  0.028
NB - fold 7 Mean Accuracy:  0.0
NB - fold 8 Mean Accuracy:  0.0
NB - fold 9 Mean Accuracy:  0.0
NB - fold 10 Mean Accuracy:  0.0
```

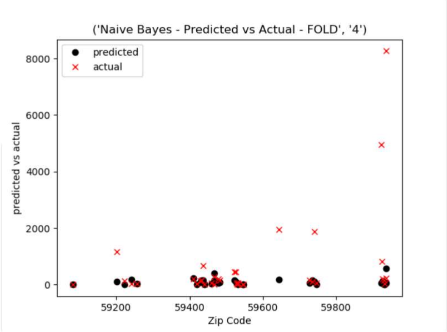
I also wanted to visualize the results of the Naive Bayes algorithm. So, much like I had done for SLR, I plotted the actual and predicted results by zip code for each of the folds. Here are the results:

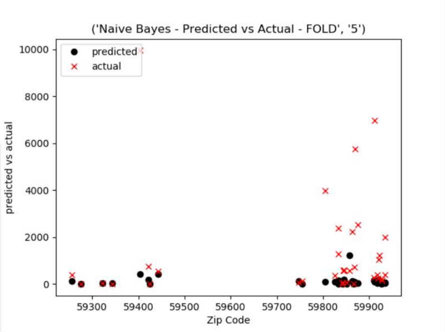

('Naive Bayes - Predicted vs Actual - FOLD', '0')
● predicted
× actual
predicted vs actual
Zip Code
0
2000
4000
6000
8000
59000
59200
59400
59600
59800

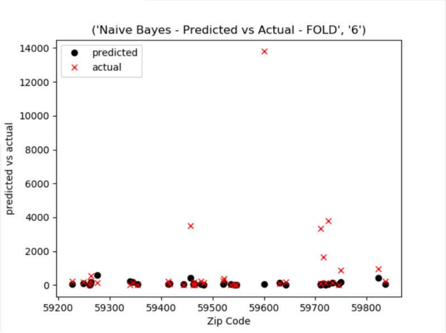

('Naive Bayes - Predicted vs Actual - FOLD', '1')
● predicted
× actual
predicted vs actual
Zip Code
0
1000
2000
3000
4000
5000
6000
7000
59000
59200
59400
59600
59800

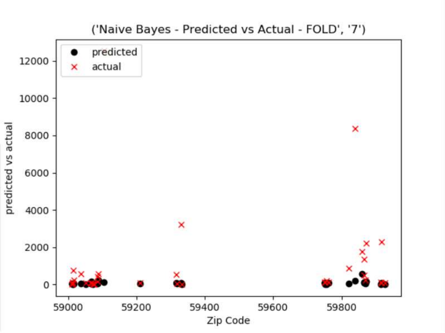

('Naive Bayes - Predicted vs Actual - FOLD', '2')
● predicted
× actual
predicted vs actual
Zip Code
0
5000
10000
15000
20000
59000
59200
59400
59600
59800

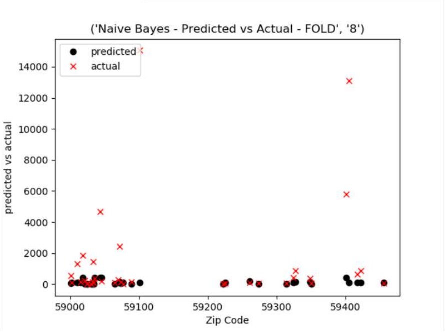

('Naive Bayes - Predicted vs Actual - FOLD', '3')
● predicted
× actual
predicted vs actual
Zip Code
0
5000
10000
15000
20000
25000
59200
59300
59400
59500
59600
59700
59800
59900

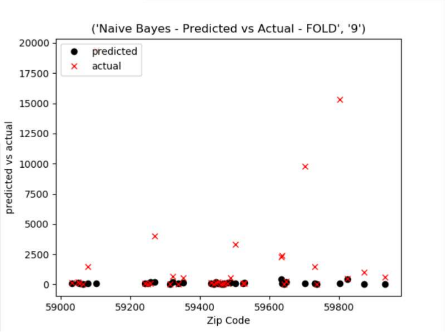

('Naive Bayes - Predicted vs Actual - FOLD', '4')
● predicted
× actual
predicted vs actual
Zip Code
0
2000
4000
6000
8000
59200
59400
59600
59800


('Naive Bayes - Predicted vs Actual - FOLD', '5')
● predicted
× actual
predicted vs actual
Zip Code
0
2000
4000
6000
8000
10000
59300
59400
59500
59600
59700
59800
59900


('Naive Bayes - Predicted vs Actual - FOLD', '6')
● predicted
× actual
predicted vs actual
Zip Code
0
2000
4000
6000
8000
10000
12000
14000
59200
59300
59400
59500
59600
59700
59800


('Naive Bayes - Predicted vs Actual - FOLD', '7')
● predicted
× actual
predicted vs actual
Zip Code
0
2000
4000
6000
8000
10000
12000
59000
59200
59400
59600
59800


('Naive Bayes - Predicted vs Actual - FOLD', '8')
● predicted
× actual
predicted vs actual
Zip Code
0
2000
4000
6000
8000
10000
12000
14000
59000
59100
59200
59300
59400


('Naive Bayes - Predicted vs Actual - FOLD', '9')
● predicted
× actual
predicted vs actual
Zip Code
0
2500
5000
7500
10000
12500
15000
17500
20000
59000
59200
59400
59600
59800

Conclusion

I was incredibly surprised by the poor metrics of the results. I was expecting that the number of people born out of state would be a fixed function or ratio of the population (and other variables) of the zip code. However, I was not able to create model that demonstrated this. If I had more time, I would want to build a mean squared error function to better determine the results of naive Bayes Algorithm. I would also like to experiment with multiple linear regression to see if that will yield better results than simple linear regression.

I would say the highlight of this project for me was the ability to pull census data and be able to visualize it. There is a lot of information available in the census, and I would love to be able to play around and build models with it. This project is one step in that direction.