Bruce Clark
Data Mining 347
4/17/2020

NOTE TO TEACHER: I see that this assignment was due on 4/15, however, I was looking at the "due date" in D2L that said 4/17. I now believe that this is the time when the D2L submission folder closes. I hope you can understand the misunderstanding.

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.

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. This is the current list of attributes that I have chosen, although it is subject to change in the future: 'population', 'born out of state', "income of > $75k", 'under 18', 'grad', 'undergrad', 'total in poverty', 'zip code'.

I then built this query into Python And into a pandas data frame. I'm using the following function to accomplish this:

```
def call_census_api(zip_codes: list):
    zip_codes_list = ','.join(zip_codes)
    variables = ["B01003_001E", "B06001_025E", "B06010_011E", "B09001_001E",
"B14001_009E", "B14001_008E", "B17001_001E"]
    variables = ','.join(variables)

    URL= ("https://api.census.gov/data/2017/acs/acs5?key=" + apiKey + "&get=" +
variables + "&for=zip%20code%20tabulation%20area:" + zip_codes_list)
    response = json.loads(requests.get(URL).text)[1:]

    df = pd.DataFrame(columns=['population', 'bornOutOfState', "income of > $75k",
'under 18', 'grad', 'undergrad', 'total in poverty', 'zipcode'], data=response)
```

Summarizes the data set (how many instances and attributes, how many categorical and numerical features, how many nodes and edges if using graph data, …)

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:

	population	bornOutOfState	income of > $75k	under 18	grad	undergrad	total in poverty	zipcode
count	361	361	361	361	361	361	361	361
unique	324	284	173	243	60	110	318	361
top	142	20	0	0	0	0	142	59244
freq	3	6	26	19	189	118	3	1

Lists data mining techniques you would like to use to help solve this problem

Because I am trying to predict the number of out of state people in each zip code, I will have to choose a regression algorithm. For this reason, I am considering using the regression-based version of the naive bayes algorithm or simple linear regression.

Describes what part of your proposed solution may need to be left for future work if you run out of time

Because there are so many attributes to play with, I don't expect to find the best attributes or combination of attributes. That would take an incredible amount of effort and would require more time than I can give. The best that I can do right now, is run a few tests and give it my best guess as to which combination of attributes would be most predictive.