ANALYZING THE SIMILARITY OF U.S. CITIES

ADAMM HOCKMAN

IBM DATA SCIENCE CERTIFICATION

COURSERA.ORG

25 MAY 2020

0 TABLE OF CONTENTS

1 INTRODUCTION

1.1 BACKGROUND

There are tens of thousands of cities across the U.S. with a wide array of characteristics, many of which are quantifiable. What exactly gives a city its character is a surprisingly complicated question. There are many stakeholders when it comes to this type of question. Suppose a travel agency must make a recommendation on which city to visit based only on a list of cities the client has indicated they enjoy the most. Imagine a new job requires you to work remotely and hence will pay moving expenses for any city in the U.S. as a perk. The applicant may want to move to a new city that is similar to cities they know they like. A restaurant may be looking at expanding into new cities and wants to make sure the new cities are similar to the cities where they have had success. In all of these situations, it is not clear how to define similarity between cities.

1.2 PROBLEM

Here we will utilize an unsupervised learning algorithm to cluster U.S. cities based primarily on the distribution of different venues in each city. We will also consider how additional factors such as population, politics of the state they reside in, as well as health statistics for the state influence this clustering.

1.3 INTEREST

There is both personal interest as well as corporate interest in looking at this clustering of U.S. cities. Meaningful conclusions will improve the quality of life for individuals looking for a new city to call home. They will also help expanding businesses have more success in unknown cities across the U.S.

2 DATA

2.1 DATA ACQUISITION

2.1.1 Data Sources

The U.S. Census Bureau provides population data from 2010 to 2019 for every census recognized area in the U.S. We use this to build our primary dataset based on individual cities in the U.S. After getting the names and populations, we will need coordinates for each city. Unfortunately, GeoPy does not allow heavy use of their library to return coordinates for a city. Hence, we will need to cut down the size of our dataset by removing cities with too small of a population.

As every city resides within a state, certain characteristics of the state will naturally influence the metrics for a city. Given this observation, we will use a combination of sources to build a dataset of all U.S. states with the following information: (a) population, (b) political affiliation, (c) health statistics, and (d) unemployment. Population data on the states also comes from the U.S. Census Bureau. For political affiliation, we scrape the Wikipedia page containing presidential election results for all U.S. states. Health statistics are scraped directly from the Centers for Disease Control. Finally, the unemployment rates come from the U.S. Bureau of Labor Statistics.

We will use Foursquare to obtain a list of the top 100 venues within a 500 ??? radius. This will give us a distribution of certain venue categories across each city. We are using the free developer account for API calls and may need to spread out the data collection over several days to not hit the limit of number of calls allowed per day.

2.1.2 Wrangling Method

We will primarily rely on the webpage scraping tools provided by the pandas library. Using the Excel Reader and the HTML Reader, we can get all of our data besides the venue information. To get the venue distribution for each city we will make API calls to Foursquare using the free edition of a Foursquare developer account.

2.2 Data Preparation

After obtaining the population figures from the U.S. Census, we need to get the latitude and longitude for each city. To achieve this, we rely on the free GeoPy package offered in Python. However, this service does not allow heavy use. We cannot make more than one call per second. The census data consists of 19,500 cities and would take approximately five and a half hours to complete. To reduce this overhead, we choose to remove those cities with a population of less than 20,000. This reduces our sample to 1,800 cities. In fact, since our venue data comes from Foursquare, this step is necessary to obtain venue data.

Next, we can add the state data to our data frame using an inner join on the state. After some cleaning (i.e. removing redundant columns, re-labelling, and sorting), we have each city as well as the auxiliary statistics associated with it.

Our final step is to obtain the venue data. Using the Foursquare API calls, we get the top 100 venues in each city, within a 5,000 meter radius of the coordinates. We then apply one-hot encoding to the venue category column, group the results by city, and take the sum over each group. Hence, we are left with the total number of venues of various categories (approximately 500) within each city.

If a city's coordinates could not be found using GeoPy, or if a city returns no venues, then we will drop these rows. This does not affect our outcome since we will be using an unsupervised learning algorithm.

2.3 Feature Selection

After processing the data, we end up with 1,737 total cities, with 558 features. The features we use to train our model include the following:

Table 1. Feature selection for primary dataset (city-level).

Column	Description
'Location'	"City, State" (e.g. "St. Louis, Missouri")
'City Population 2019'	The cities estimated population for 2019, by the U.S. Census Bureau,
'City Growth (% since 2010)'	The total population in 2019 divided by the total population in 2010.
'City Latitude'	City latitude returned by geopy.

'City Longitude'	City longitude returned by geopy.
'Total Venues'	Total venues returned by Foursquare.
'ATM'	Total number of ‘ATM’ for city.
'Accessories Store'	Total number of 'Accessories Store' for city.
'Adult Boutique'	Total number of 'Adult Boutique' for city.
'Advertising Agency'	Total number of 'Advertising Agency' for city.
'Afghan Restaurant'	Total number of 'Afghan Restaurant' for city.
…	…
"Women's Store"	Total number of " Women's Store" for city.
'Yoga Studio'	Total number of 'Yoga Studio' for city.

Table 2. Feature selection for secondary dataset (state-level).

Column	Description
'State'	State name (e.g. ‘Missouri’)
'State Population 2019'	Population estimated by the census.
'State Growth (% since 2010)'	Population growth for state.
'State Birth Rate'	Births per 100k
'State Death Rate'	Deaths per 100k
'State Election 2000'	One-hot encoded with _D and _R
'State Election 2004'	One-hot encoded with _D and _R
'State Election 2008'	One-hot encoded with _D and _R
'State Election 2012'	One-hot encoded with _D and _R
'State Election 2016'	One-hot encoded with _D and _R
'State Unemployment'	Rate as of April 2020 by the BLS.

The columns with the election results and the various venue categories have been one-hot encoded, with indicator variables in place of categorical variable values.

In the process of cleaning the data, we encountered several instances of features that contain redundant information. For example, total births is a less meaningful feature than birth rate because rates are independent of the how long the window was open to record totals, hence a more objective comparison. We also condensed the population totals for 2010-2018 into one feature—population growth across that period.

3 METHODOLOGY

3.1 HEADING

BULK OF THE REPORT

Include: exploratory data analysis, inferential statistical testing, if any, and what machine learning algorithms were used and why.

4 RESULTS

4.1 HEADING

Include: pictures, models, brief discussions of the takeaways from the study.

5 DISCUSSION

5.1 OBSERVATIONS

discuss observations here

5.2 RECOMMENDATIONS

discuss any recommendations to the target for further study

6 CONCLUSION

if applicable

7 REFERENCES

if applicable

8 ACKNOWLEDGEMENTS

if applicable

9 APPENDICES

if applicable