Projects in Applied Data Science: Fall 2019

Editors:
Caleb Phillips and Lindy Williams

Authors:
Rahul Aedula*
Ally Arenson
Yash Gandhi*
Steven Hobbs*
Parth Jawale*
Holden Kjerland-Nicoletti
Israel Miles
Jacob Munoz
Karthik Palavalli*
Srishti Rawal*
Thanika Reddy*
Lakshya Sharma*
Nimra Sharnez
Orgil Sugar
Tyler Tokumoto
Telly Umada*

* Graduate Students

University of Colorado, Computer Science Department
Boulder, Colorado, 80309-0430
December 31, 2019

Foreword

This document contains semester projects for students in CSCI 4381/7000 Data Science Projects. This course explores concepts and techniques for design, formulation and execution of practical, applied data science. Topics covered include experimental design, statistical analysis and predictive modeling, machine learning, data visualization, scientific writing and presentation. During the class, students selected a semester-long project to acquire, analyze, and understand data in support of a research question. In addition to traditional lectures, students read and discussed published papers on data science topics, practiced skills in recitation sessions, and entertained guest lectures from expert data scientists in the field. Outside of these readings and recitations, students were allowed to work on their projects exclusively and were supported with meetings, peer-discussion and copyediting.

In terms of the scope of the final product, undergraduate students were asked to perform a research or engineering task of some complexity while graduate students were additionally required to perform a survey of related work, demonstrate some novelty in their approach, and describe the position of their contribution within the broader literature. All students who performed at or above these expectations were offered the opportunity to contribute their paper for publication in this technical summary.

The diversity of the papers herein is representative of the diversity of interests of the students in the class. There is no common trend among the papers submitted and each takes a different topic to task. Students made use of open data or worked with organizations to acquire data. Several students pivoted their projects early on due to limitations and difficulties in data access --- a real-world challenge in practical data science. The projects herein range from analyzing traffic in cities, restaurant trends and Facebook responses to smartphone accelerometer data, scaling laws in higher education, and bicycle trends in Boulder, Colorado. Analysis approaches are similarly varied: visualization, statistical analysis and modeling, machine learning, reinforcement learning, etc.. Most papers can be understood as exploratory data analysis, although some emphasize interactive visualization and others emphasize statistical modeling and prediction aimed at testing a well-defined research question. To inform the style of their approach, students read papers from a broad sampling of original research. They used these

readings to build an understanding of approaches to presentation and analysis in the modern scientific literature. One paper was held out from this compendium so that it could be submitted for publication to a peer-reviewed venue.

Please direct questions/comments on individual papers to the student authors when contact information has been made available.

Table of Contents

Predicting Traffic Congestion in Cities
Srishti Rawal
7 pages

An Analysis of the Popularity of Facebook News Posts
Thanika Reddy
14 pages

Opinion Fraud Detection in Amazon Reviews
Lakshya Sharma
7 pages

A Demographic Analysis of Fatal Encounter with Law
Nimra Sharnez
11 pages

Customer Demographics Study
Orgil Sugar
8 pages

Analysis of the Spread of Restaurant Trends in Las Vegas, Nevada
Tyler Tokumoto
9 pages

Comparing Yelp Ratings and Food Trends across the United States
Tetsumichi (Telly) Umada
8 pages

Scalable collaborative filtering based recommendation on Yelp

Rahul Aedula
University of Colorado Boulder
rahul.aedula@colorado.edu

Abstract

Recommendation systems have become commonplace in today's user experience online. From Netflix's shows to Amazon's products almost any service which can monetize a user's involvement has a recommendation system tuned to enhance the overall experience. The biggest advantage that these systems provide is an opportunity for the user to discover new products by means of crowdsourcing. Collaborative filtering is an essential part of a recommender system as it gives insights into data which can be useful in finding items which are similar to other users. Most collaborative filtering techniques rely heavily on finding closely matching items with the use of similarity metrics such as Pearson's correlation and Cosine similarity. These methods are fairly effective in finding products that are liked by similar users but perform poorly with high sparsity. Due to the large skew in the number of users over the total products that are available the problem with sparsity becomes very relevant. Using Yelp, as a case study we will try to address some of the domain specific fixes which can not only help in managing the scale of the data but also improve the overall performance of the recommender systems. We will also selectively contrast the existing recommender systems to capture their performance with these new improvements.

I. Introduction

Yelp has become the leading source to find comprehensive reviews and ratings for a wide variety of businesses in North America. The user experience is the most important aspect of Yelp, finding businesses in the surrounding proximity to narrow down is its primary function. The ratio of businesses to users is very low, making this a perfect problem to make changes tackle scale and sparsity. As aforementioned Yelp holds reviews for a broad spectrum of businesses with categories like entertainment, nightlife, fitness etc. We will be primarily focusing on food related businesses so that we can utilize the majority of the data and keep our computation times more manageable within the allocated resources. However, the methods we will be using are fairly generalized and are equally applicable for any of the categories.

To breakdown the sparsity problem we must first understand why this problem occurs in the first place. The users subscribing to a service such as Yelp are more prone to be browsing items and may not necessarily be invested in giving ratings or reviews of any kind. This brings us to our first cause of sparsity which is non-engaging users. The next biggest problem is that as a user they may have limited capabilities in terms of time or money to review anything else apart from their usual set of items which can be classified as engaging users. Most recommendation systems suffer due to the lack of data as a result of non-engaging users. However the few insights which we can get are completely attributed to the engaging user. Our goal in this paper is to find a way to restrict the size of the search based on locations and friends so as to reduce the overall sparsity and get more focused ratings which we can use to make our recommendations.

Filtering the data before performing any calculations is the key to make more effective and faster recommendations. This will ensure that the data is not only easily handled in memory but also improve our chances of getting useful recommendations for the user. Vectorization is also a very big part of the process this will

ensure that the compute times for these filtering methods stay very low. The two methods we will be primarily using are the weighted locations estimate to find the estimated location of the user and a consecutive user-friend filter which will add to the relevancy of the recommendation. We will then proceed to juxtapose SVD and K means based recommender systems along with their variations against a baseline recommender to see how well they perform.

For the purpose of this case study we will not be focusing on the cold start problem but just users who already exist in the database to give relevant recommendations. We will consider each entry for a review made by a user as a user's entry into our database (essentially all the users in the review table) which we will later split into training and testing.

II. Data

The data used in this analysis is courtesy of the Yelp dataset challenge. The data spans a wide spectrum of characteristics in regards to the Yelp users and its registered businesses. It ranges from user information to reviews about various businesses while also providing some image data of various categories posted on Yelp. Since the focus is to create a robust recommendation system which effectively filters down businesses on a user to user basis, the data which is most preferred for this analysis will be that pertaining to business data, user data and review information.

When dealing with Yelp data the most notable aspect is its scale, most of the user and review data prove to be a challenge in terms of computation of expected results. The review data alone is approximately 5GB consisting of 6.6 million comprehensive user reviews on businesses making this a very memory intensive dataset. This scale makes loading and handling data from the native format a challenge and quite difficult to work with. Usage of efficient data structures and coding practices make a big difference when dealing with such large amounts of data.

The default data format given by Yelp is JavaScript Object Notation (JSON), but this has proven to be inefficient for repetitive read and write times. A good approach to use would be to convert the data into the parquet or CSV file type as this allows for faster read times for future analysis. Further optimizations such as reading only the required columns and the usage of pyarrow library in python to read parquet files also drastically reduce reading and preprocessing times.

In this case study we will be using parquet to store the review JSON since it is the largest and also the most relevant dataset in our analysis. As mentioned before the dataset of the review frame is over 6 million rows, a simple operation like converting it into a parquet file without a powerful RAM will have wait times of several hours. The method we used to solve this issue is batch reading. By reading around 400k rows in a batch and clearing them with efficient garbage collecting at every cycle we eliminated any memory errors that may arise and maintain the overall time for the operation at around 3 hours.

[jmcarpenter, 2019] A big part of data management is vectorization. We will talk in more detail about this in the methods section but it is very useful to use pandas and numpy at this stage as they are very well optimized to read and hold the batches which is being written into the parquet file in a memory efficient way due to their well vectorized routines.

Yelp dataset is inherently very clean, most of the entries are of good significance and well structured. The only few places where there is any need to cleaning are the names of the cities and other geographical labels. However, in this case study none of these are a really big issue as we will be using these locations to shorten the search. The ending result of all these data transformations are fast read formats which allow us to perform multiple reads with very less times. In the parquet format, pandas can read the review frame with selected columns in under a minute which is a considerable improvement over the JSON.

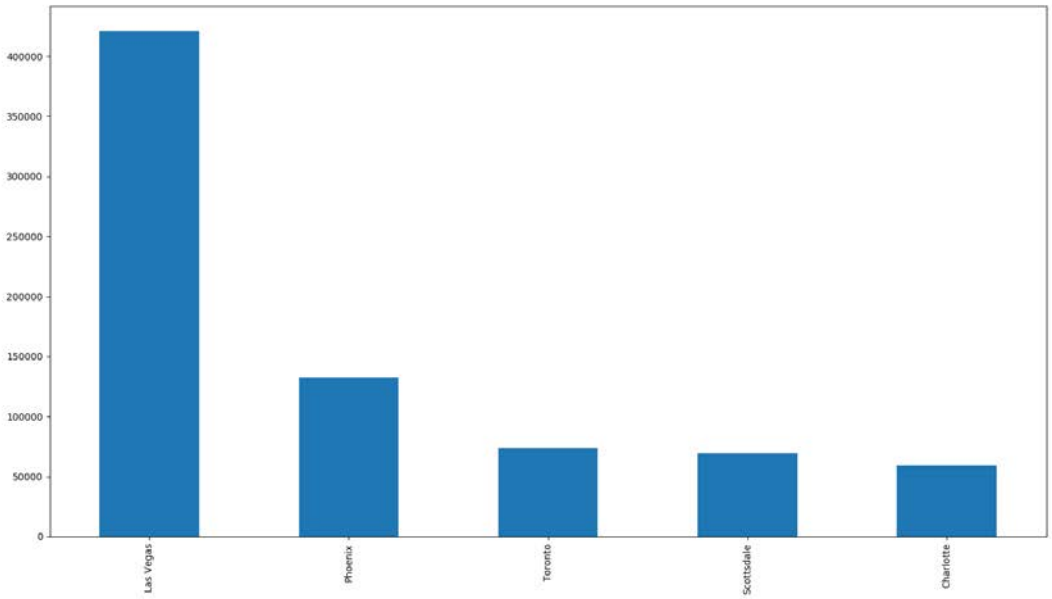


Figure 1: *Frequency plot of most reviewed cities*

III. Methods

Generating relevant recommendations is a challenge in itself. The lack of ground truth of the results places stress on the methods to ensure interpretability of the model. This is necessary to make any decisions inside the model as cogent and lucid as possible. As discussed earlier we are trying to filter out data based on location and friends the users may have. Upon closer inspection of the data we notice that most of the users have only one review which means that we most of their locations can be estimated by taking into account where the review was made. Another noticeable thing is that a fair share of the users do not have any friends on Yelp. This would imply that if we picked any one criteria we may fall short on recommendations for most of the users. Hence making it a composite filter is the best way to go. By filtering on location and friends we can ensure that most of the users have atleast some recommendations and they are most likely to be relevant. As aforementioned vectorization plays an important part in making this whole system scalable, the use of libraries like numpy and pandas improve computation speeds during filtering.

For the first method we will try to estimate the city in which the user is located based on their reviews. Since the review data gives us the information about the business which was reviewed we can use that as an indicator to see which city was the review given in. Similarly the same can be done for all the users and all the businesses they reviewed, This gives us a list of business ids which are mapped to the cities in which they are located in. The following mapped cities can be checked for majority to estimate the city of the user. While this method is not really precise we can confine a user's business recommendation to a city which they are most likely to live in or visit. A few methods were tested where the weights of the friends cities were also considered but the issue stated earlier that most of the users on Yelp don't have friends made it easier to just a frequency for the weighted location estima-

Figure 2: *User - User correlation of a sample*

tion. Vectorizing your code with libraries like swifter [jmcarpenter, 2019] while performing this analysis improves the speed by 72x.

We can extract the user's friend list from the user data and use it as a part of this analysis. By adding a mix of businesses reviewed by friends in the estimated city (and possibly other cities) we are adding a sense of randomness and serendipity to the recommendation system. The biggest advantage of utilizing friends as a part of the recommendation system is that we are adding a bunch of users to the ratings matrix which can become very useful to give recommendations to the user.

We can now proceed to split the resulting frame which was generated by location and friends into training and testing. We can treat users who exist in the training frame as users to recommend and the ones who don't as cold start users. Since we want to highlight scale in this case study we will be using around 4 million rows as train set and around 200k rows as test set. Upon performing our location filtering we can effectively reduce this to a 1.2 million data frame where each single user is mapped with locations as a field.

Upon completing the filtering stage we are now focused on making a user-item matrix (also referred as utility matrix). Since we emphasized on using collaborative filtering early on we will not be using an item-item matrix for our recommendation. Our goal is to find users with similar interests and recommend items or in our case businesses similar to their liked items. To make this stage efficient we can again vectorize it with Swifter [jmcarpenter, 2019] to pass through each row of the data frame quickly while performing all the operations.

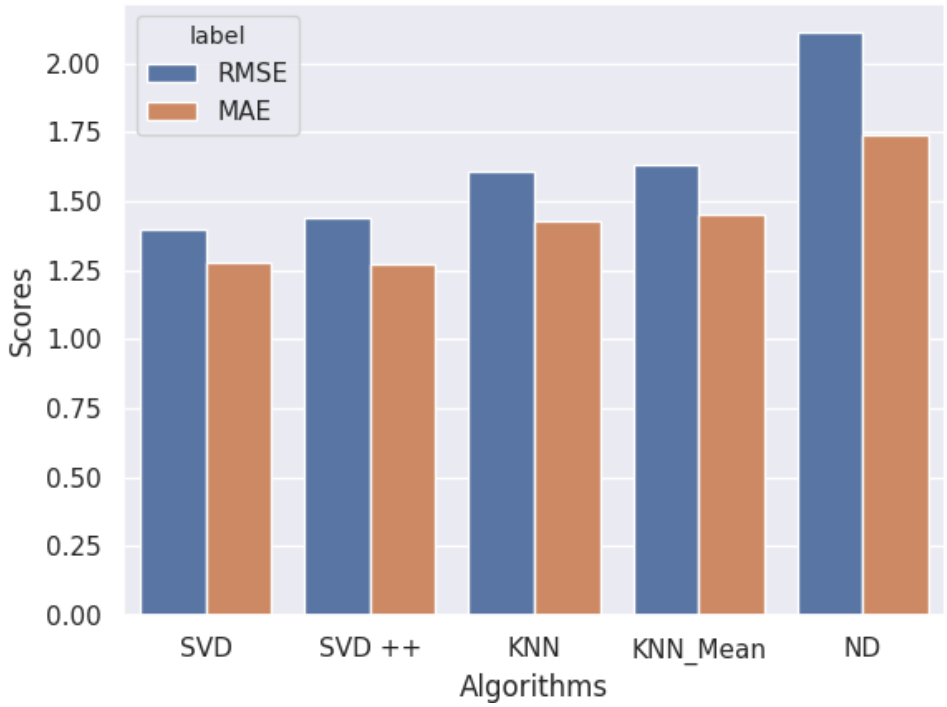


Figure 3: *RMSE vs MAE comparison chart*

Ideally this operation should generate the user-item matrix where the values are the ratings but if we are using libraries like scikit-learn surprise we can skip the formation of the matrix and feed the user-item data frame with ratings as another column.

Visualizing this matrix with a correlation plot we can clearly see that the matrix is majority sparse while maintaining some correlation with the users. This means that we can see some relationships exist within the users while also seeing most of this matrix is sparse. The first step would be to reduce sparsity and the recommended way to do that is by filling in the missing values by the mean of the column. However we will be skipping this step since we are using scikit-learn surprise to implement the recommender algorithms which implicitly handles missing data.

The above methods give us the base set of filters before we perform our recommendation, these filters not only increase the relevancy of the results seen by the users but also optimizes the way we deal with really large scale data. As emphasized before the data is very sparse, hence making any type of similarity function may not give is clear cut results about what to recommend. This is where Singular Value Decomposition really shines.

$$A_{m\times n} = U_{m\times k} \times \Sigma_{k\times k} \times V^T_{k\times n}$$

SVD splits a matrix into three different matrices. The U and V^T are the most important matrices for this problem as U corresponds to the users in the utility matrix and V^T corresponds to the businesses in the utility matrix [Sawant, 2013]. The dot product of a given user row to a business column gives us the estimated rating of a particular business for a user. [Brand, 2003] If the estimated ratings go over the scale we then prune the ratings to fit the scale we have defined. Scikit-learn surprise

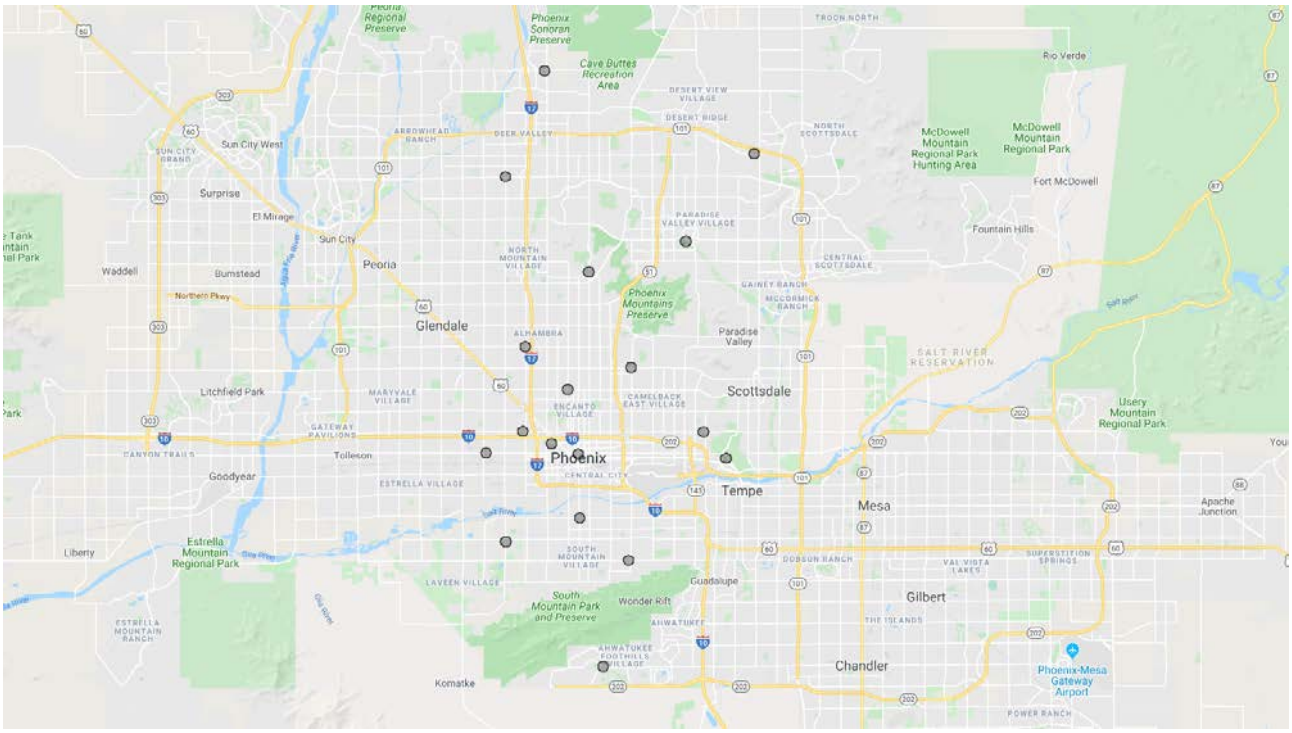


Figure 4: *Sample recommendation at scale*

takes care of all this implicitly as a parameter.

Now that we filled all the values of our rating matrix with estimated ratings from SVD, a similarity metric such as Pearson's correlation or Cosine Similarity can be used effectively to choose a user who has the highest correlation with the user in being recommended for. The top n rated businesses from the similar user can then be used as recommendations for the test user.

IV. Results

Evaluating recommender systems is a challenging task. One of the most famous evaluation methods for recommender systems is A/B testing where different groups of people are exposed to different products to ascertain how well the overall system works. This might be a bit difficult with our case since we do not have access to live users where we can perform a controlled experiment on. The next best thing we can look for is offline methods of evaluation. Though these are not as accurate as that of A/B testing they give us some basic insight on how our recommender systems perform. The two metrics we will be using to evaluate our recommender systems are MAE and RMSE. These can show the quantitative difference between our predicted value and the true value and give us a sense of how our recommender systems perform. This is a very regression like approach but has some benefits on understanding the flaws of our recommender system. Since we are using Scikit-learn surprise we can also go ahead and check how well our recommender system works against other recommender models. In figure 3 we have our SVD recommender compared to SVD++, KNN, KNN with mean and a random normal distribution recommender over a few thousand iterations on cross validation splits of 5. The normal distribution recommender is used as a baseline since it basically measures the distribution of the ratings and then recommends randomly. As expected the SVD based recommender has

lesser RMSE and MAE error and performs really well. The overall estimated time of recommendation on average per user is less than 2 seconds with an average RMSE of 1.3 and MAE of 1.28 which basically does a little better than some of the other results [Sawant, 2013] obtained and is faster. The remarkable thing is it scales really well with memory of a system hence making this method extremely scalable.

V. Discussion and Conclusions

The scalable recommender system not only performed very well in terms of accuracy but also speed. With results such as average RMSE of 1.3 and MAE of 1.28 we have shown that the SVD is on par and perhaps a little better than the other results. However, this is still not the best recommender system in terms of predicting values. The following filters can be used in succession along with other graph methods to achieve a much higher accuracy. In terms of implementing matrix factorization based recommender system this was fairly straightforward, more norm based techniques can be used to ensure that the ratings predicted by them are as close to the true value as possible.

NLP techniques like Latent Dirichlet Allocation (LDA) especially LDA2VEC [Cemoody, 2016] allow us to discover hidden topics inside text, this can be used to identify the categories which a user may like from their reviews and match them with businesses of similar categories. For the purpose of this project we only filtered food reviews but such an approach can be used to perform recommendations on any type businesses at scale.

In conclusion more methods could be tested in sequence with these filter to further demonstrate the results. One major improvement that could be done is use A/B testing to see how well it actually performs on a real world setting.

References

[Cemoody, 2016] Cemoody. (2016, May 31). cemoody/lda2vec. Retrieved from https://github.com/cemoody/lda2vec.

[jmcarpenter, 2019] jmcarpenter2. (2019, September 26). jmcarpenter2/swifter. Retrieved from https://github.com/jmcarpenter2/swifter.

[Sawant, 2013] Sawant, Sumedh, and Gina Pai. "Yelp food recommendation system." (2013).

[Sarwar, 2002] Sarwar, Badrul, et al. "Incremental singular value decomposition algorithms for highly scalable recommender systems." Fifth international conference on computer and information science. Vol. 27. 2002.

[Brand, 2003] Brand, Matthew. "Fast online svd revisions for lightweight recommender systems." Proceedings of the 2003 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 2003.

[Scikit-Learn Surprise] "Welcome to Surprise' Documentation!¶." Welcome to Surprise' Documentation! - Surprise 1 Documentation, https://surprise.readthedocs.io/en/stable/.

Lean Into Your Strengths: An Analysis of CliftonStrengths Data at CU Boulder

Alyssa Arenson
University of Colorado Boulder, CU
alyssa.arenson@colorado.edu

Abstract

This study examines the impacts of personal strengths, via the CliftonStrengths assessment, on an individual's chosen area of study in at the University of Colorado at Boulder. Understanding one's inherit strengths and where they best apply can help students capitalize on their natural capabilities and pick a field of study in which they can thrive. Data was collected from CU Boulder's 2019 freshmen class and analyzed to identify any correlations between certain strength and schools/majors. Although these strengths are neither defining nor psychic, there are certain data points that future students might want to take into account as they are deciding upon a specific area of study to start guiding their career.

I. Introduction

No one knows you better than you. However, understanding and breaking down the complexities of the human psyche is no easy task, even if you are doing it to yourself. This is where a well researched series of questions can be extremely beneficial in helping people breakdown that psyche, and identify their natural tendencies, strengths, weaknesses, and more. The questions vary and are packaged in all sorts of different personality tests, some popular ones being Myers Briggs, the Disc Assessment, and the Enneagram. Assuming the tests are well tested and researched, people can use the results from these assessments to better understand themselves, and make more well-informed decisions in their daily lives.

One popular personality test is CliftonStrengths. It is a 177 question test that asses an individual's talents, natural thinking patterns, feelings, and behaviors – and categorizes them into the 34 CliftonStrengths themes. The nuances of all 34 themes are difficult to understand, so they are also categorized into four overall domains: Executing, Relationship Building, Influencing, and Strategic. Each domain is defined differently; the "Executing" domain categorizes all themes that will work tirelessly to complete their goal, the "Influencing" domain categorizes all themes that relate to effective communicating and the ability to persuade, the "Relationship Building" domain categorizes all themes that foster meaningful relationships and unite others, and the "Strategic Thinking" domain categorizes all themes that can process and innovate new solutions [Gallup, 2019].

CliftonStrengths is a highly effective assessment that has been taken by over 22 million people, and is used by over 90% of Fortune 500 companies to improve company culture. The entire purpose of CliftonStrengths is to study what is right with people and help people identify their strengths so they can lean into them and exercise them more effectively. Research data shows that people that have utilized CliftonStrengths are six times more likely to be engaged in their jobs, six times more likely to believe more productive, and three times more likely to report having an excellent quality of life [Gallup 2019].

II. Data

This study was performed on a data set of University of Colorado at Boulder students. Every

student at CU Boulder has access to and is encouraged to take a CliftonStrengths assessment via freshman orientations and different academic success programs. However the statistical analysis in this study were only performed on CU Boulder's 2019 Freshman class.

The size of CU Boulder's freshman class in Fall 2019 is 7,133, and of those, 4,115 have completed the CliftonStrengths assessment, which is about 58% of the freshman class [Fall Census, 2019]. Upon completion students are given their top 5 ranked strengths, which are stored in a school database. The database was retrieved in an excel document format from CU Boulder's Career Services Center. The database includes the name, student id, school email, college, and declared major of each student along with their top five strengths. Each strength has a name that is self-explanatory except for "Woo", which is an acronym that stands for "winning over others". There are seven different undergraduate colleges at CU Boulder, and all students with majors declared across multiple disciplines are categorized under the "MULTU" category which stands for "Multidisciplinary". The data also had to scrubbed, as many students began the assessment, but did not finish, resulting in a lot of blank strengths. Student IDs and emails also had to eliminated due to CU Boulder FERPA regulations to protect student privacy [Shaff, 2019].

The 2019 freshman class is broken down by frequency of 34 strengths and the frequency of strengths in the four domains (See Figures 1 2). Each college has a unique set of prevalent strengths. For example, the most prevalent strength in the Leeds School of Business is "Competition", in the College of Education is "Empathy", etc.

III. Methods

Identifying the correlation between categorical variables is best determined by a chi-squared "goodness of fit" test. The chi-squared test determines whether there is a significant difference between the expected frequencies and the observed frequencies between the 34 different strengths categories broken down by college [McHugh, 2013]. To do this an 8 row by 34 column contingency table must be made to display the observed frequency distribution of the variables.

Then the chi-squared test is used to identify correlation between observed and expected frequency data sets using the equation below. A high χ^2 value indicates a lack of correlation, whereas a low χ^2 value indicates a high correlation between observed and expected data sets.

$$\chi^2 = \sum_{k=1}^{n} \frac{(O_k - E_k)^2}{E_k}$$

The degrees of freedom for a chi-square test are calculated given the equation below, where r = number of rows and c = number of columns.

$$df = (r-1)(c-1)$$

The degrees of freedom calculated above can now be used to determine if the CliftonStrength variables are not normally distributed across each college by calculating the the p-value. The p-value reveals whether or not the null hypothesis can be rejected by evaluating if it is less than the significance level, which is set to 0.05.

This can all be efficiently done in python with the $chi2_contingency$ command can be used to output a chi-squared value, p-value, and degrees of freedom.

The chi-squared test is beneficial in determining significance between two groups of variables. However, determining the strength of association between variables must be determined with Cramer's V. The statistic varies from 0 to 1, where closer to zero corresponds to no association between the variables and closer to one corresponds to complete association [Kearney, 2017]. It can be calculated using the equation below where n = total number of observations, c = number of columns, and r = number of rows.

$$V = \sqrt{\frac{\chi^2/n}{min(c-1)(r-1)}}$$

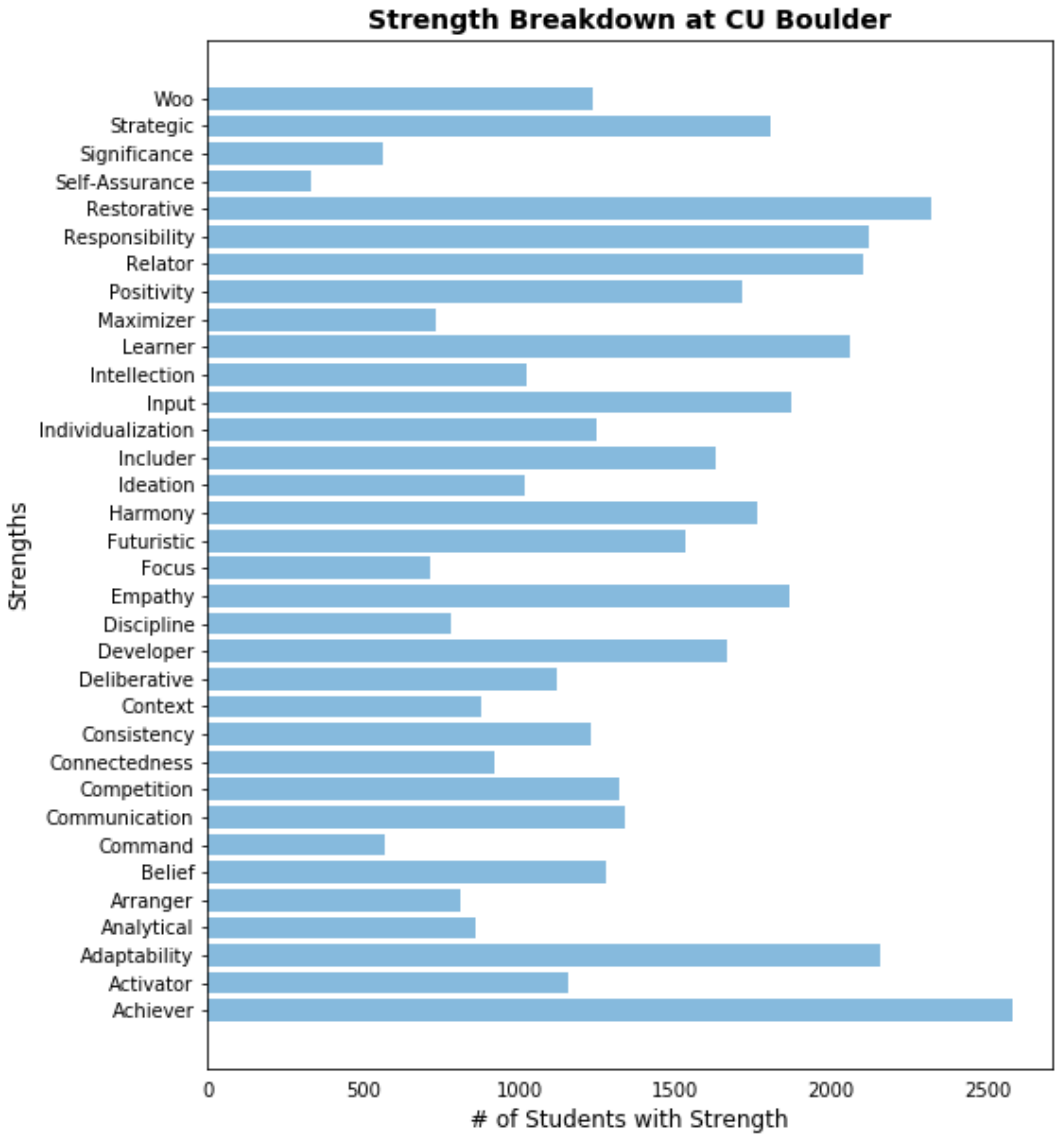


Figure 1: *The distribution of CliftonStrengths among CU Boulder freshman in 2019.*

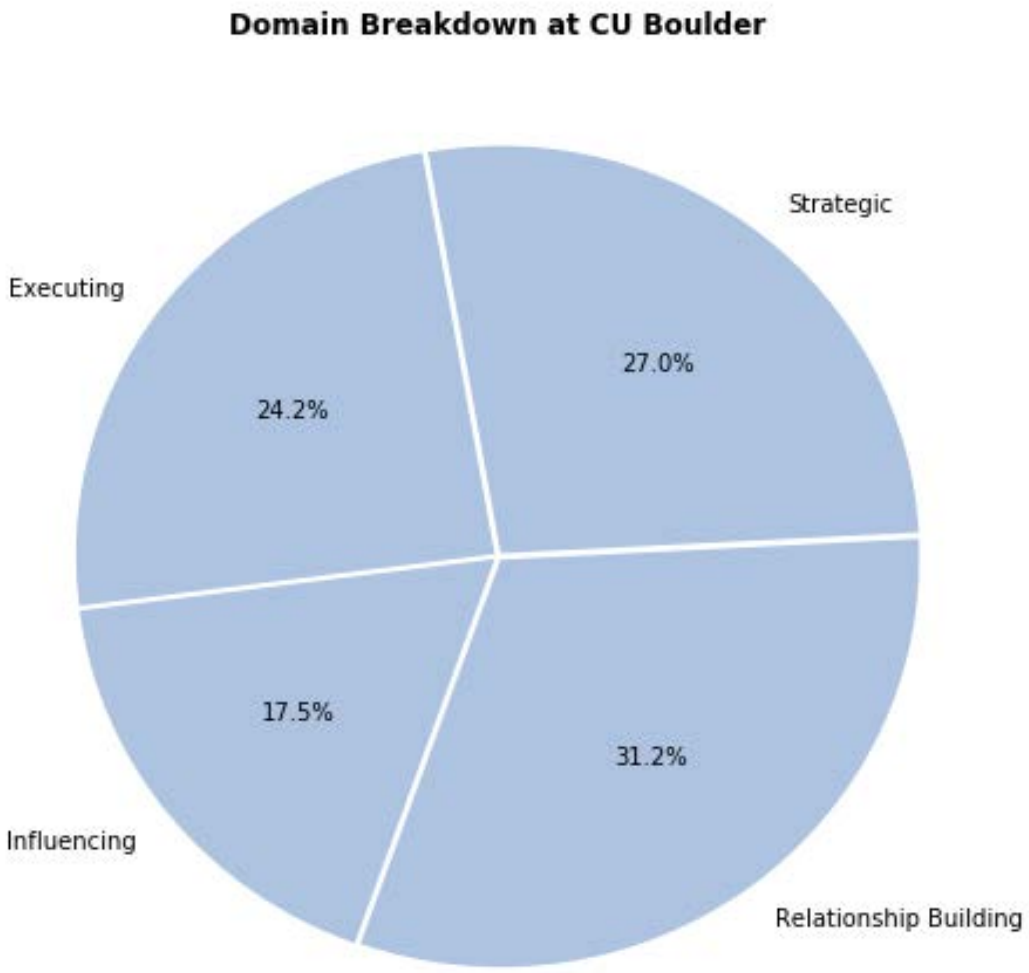


Figure 2: *The breakdown of CliftonStrengths domains among CU Boulder freshman in 2019.*

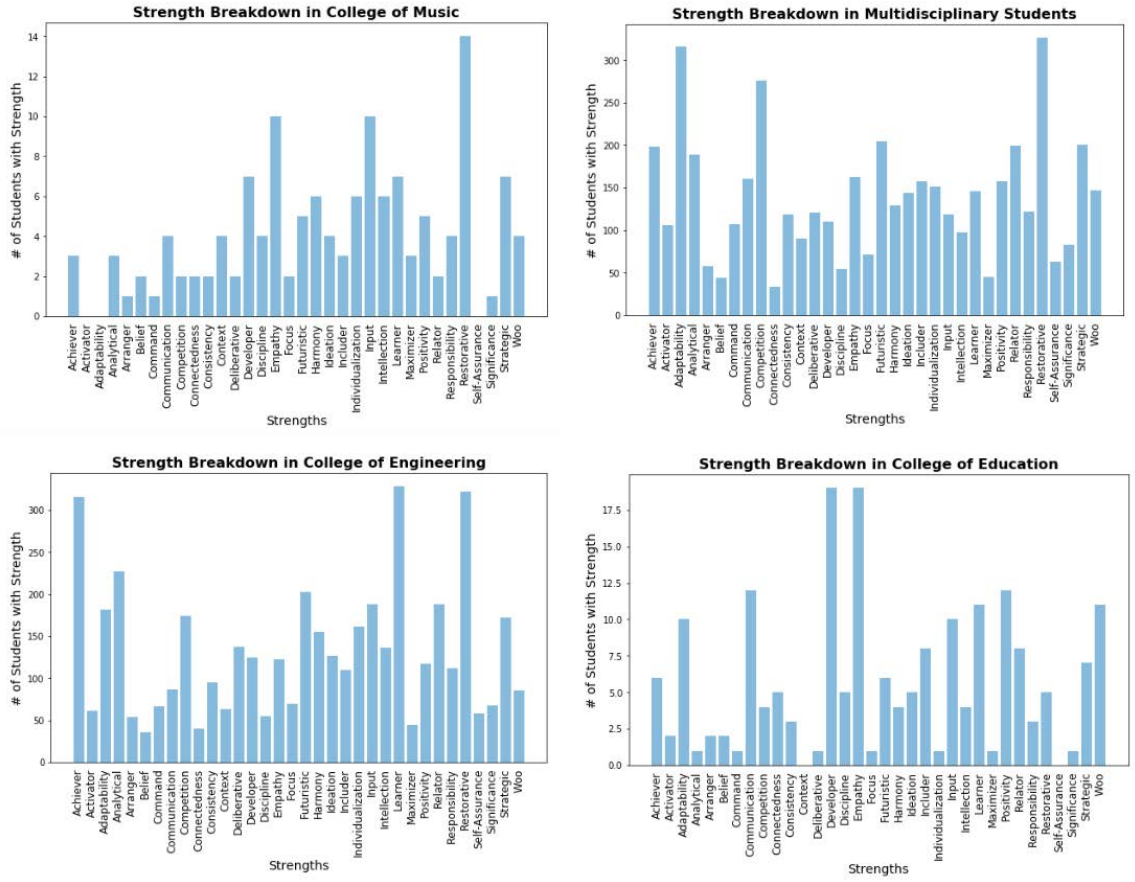


Figure 3: *The distribution of CliftonStrengths among 2019 CU Boulder freshman in different colleges.*

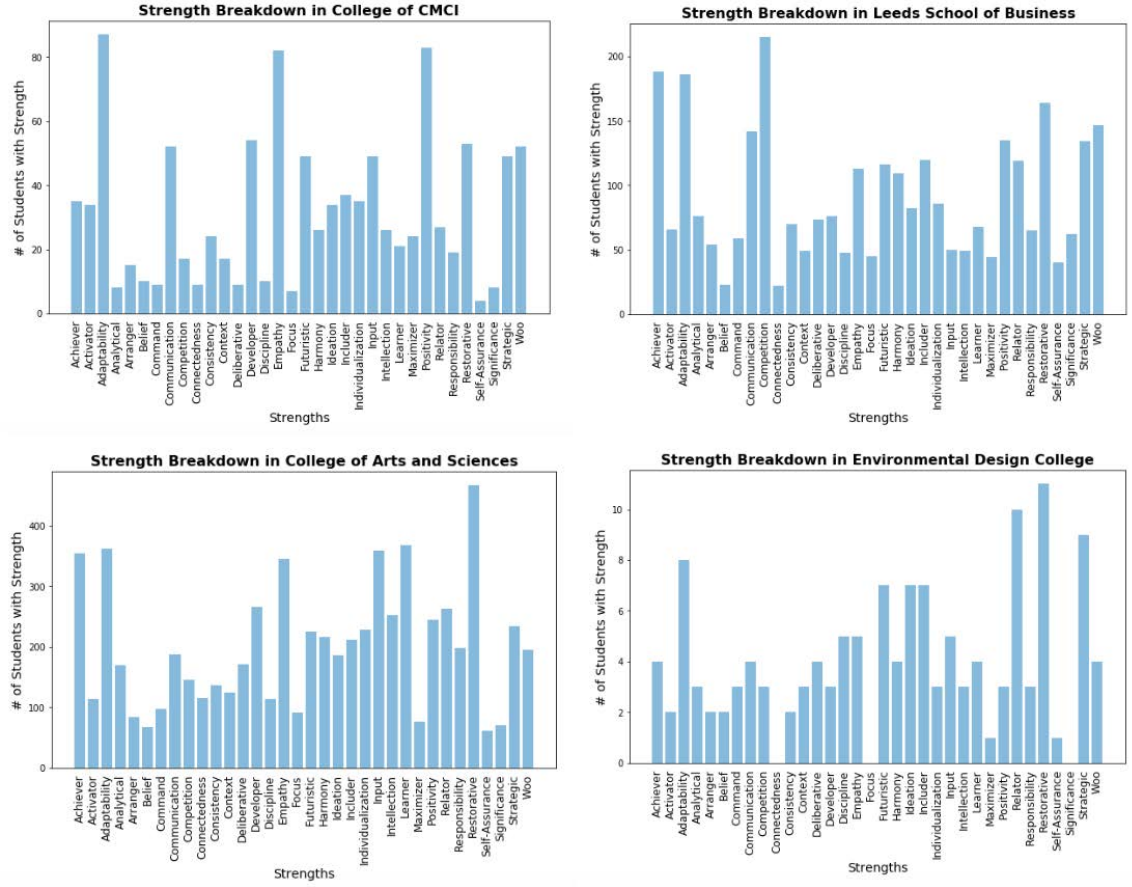


Figure 4: *The distribution of CliftonStrengths among 2019 CU Boulder freshman in different colleges.*

Chi-squared and Cramer's V asses the relationship of a single strength; however, in order to see if there is a certain top five strength combination that is popular, the order/ranking of the strengths must be taken into account, so Kendall's Tau must be used. The test outputs a statistic called the Tau correlation coefficient that returns a value of zero to one, where: zero is no relationship and one is a perfect relationship. It can be calculated using the equation below where C = the number of concordant pairs and D = the number of discordant pairs. Concordant pairs are the number of data points that are ranked in the same direction , and discordant pairs are the number of data points that are ranked in opposite directions [Glen, 2017].

$$\tau = \frac{C - D}{C + D}$$

IV. Results

After a preliminary look at the data there are a couple of key takeaways to notice. "Restorative" is the most prevalent strength across the entire CU freshman class. However, different strengths correlate with different colleges depending on the prevalence. Figures 3 and 4 contain bar graphs that show the different strength distributions for each college. Each graph displays a different distribution, where certain strengths are more popular than others. The College of Music's top five strengths are "Restorative", "Empathy", "Input", "Developer", and "Learner". Multidisciplinary students' top five strengths are "Restorative", "Adaptability", "Competition", "Futuristic", and "Strategic". The College of Engineering and Applied Science's top five strengths are "Learner", "Achiever", "Restorative", "Analytical", and "Futuristic". The College of Education's top five strengths are "Empathy", "Developer", "Communication", "Realtor", and "Woo" (See Figure 3). The College of Communication, Media, and Information's top five strength's are "Adaptability", "Positivity", "Empathy", "Developer", and "Communication". The Leeds School of Business's top five strengths are "Competition", "Achiever", "Adaptability", "Restorative" and "Woo". The College of Arts and Science's top five strengths are "Restorative", "Adaptability", "Learner", "Input", and "Achiever". The College of Environmental Design's top five strengths are "Restorative", "Realtor", "Strategic", "Adaptability", and "Futuristic" (See Figure 4).

When students are given their CliftonStrengths report, they are given their top five strengths, not just their number one strength. Although the five strengths have different weights of importance, they are all indicated by the CliftonStrengths assessment to hold a lot of value for an individual. So adjusting from the first analysis, a chi-squared analysis for the CliftonStrengths data set was performed by observing whether or not a strength was in any of an individual's top five strengths, and with the expected values equally distributed across each strength. This analysis output a chi-squared value 2441.277, which indicates that the observed distribution of data does not correlate with the distribution that is expected and therefore the sets of variables are not independent. The analysis also calculated the p-value extremely close to 0.0, and since the p-value is less than any given significance level, therefore the null hypothesis cannot be accepted, which means the CliftonStrength variables are not equally distributed across each college. The Cramer's V statistic is 0.5345, which indicates a moderate strength of correlation between the variables.

However, the analysis above inherently implies that every strength is equally common due to the expected values generated by the *chi*2$_c$*ontingency* command. The *chi*2$_c$*ontingency* command runs a chi-squared independence test by creating an observed value table that assumes equal distribution among all variables. However, some strengths are more common than others the real-world distribution must be taken into account for more accurate Chi-squared and Cramer's V tests. CliftonStrengths has data from 2016 that shows the distribution of strengths and domains among students in higher education (See Figure 5 6)[Gallup 2016]. Using this data,

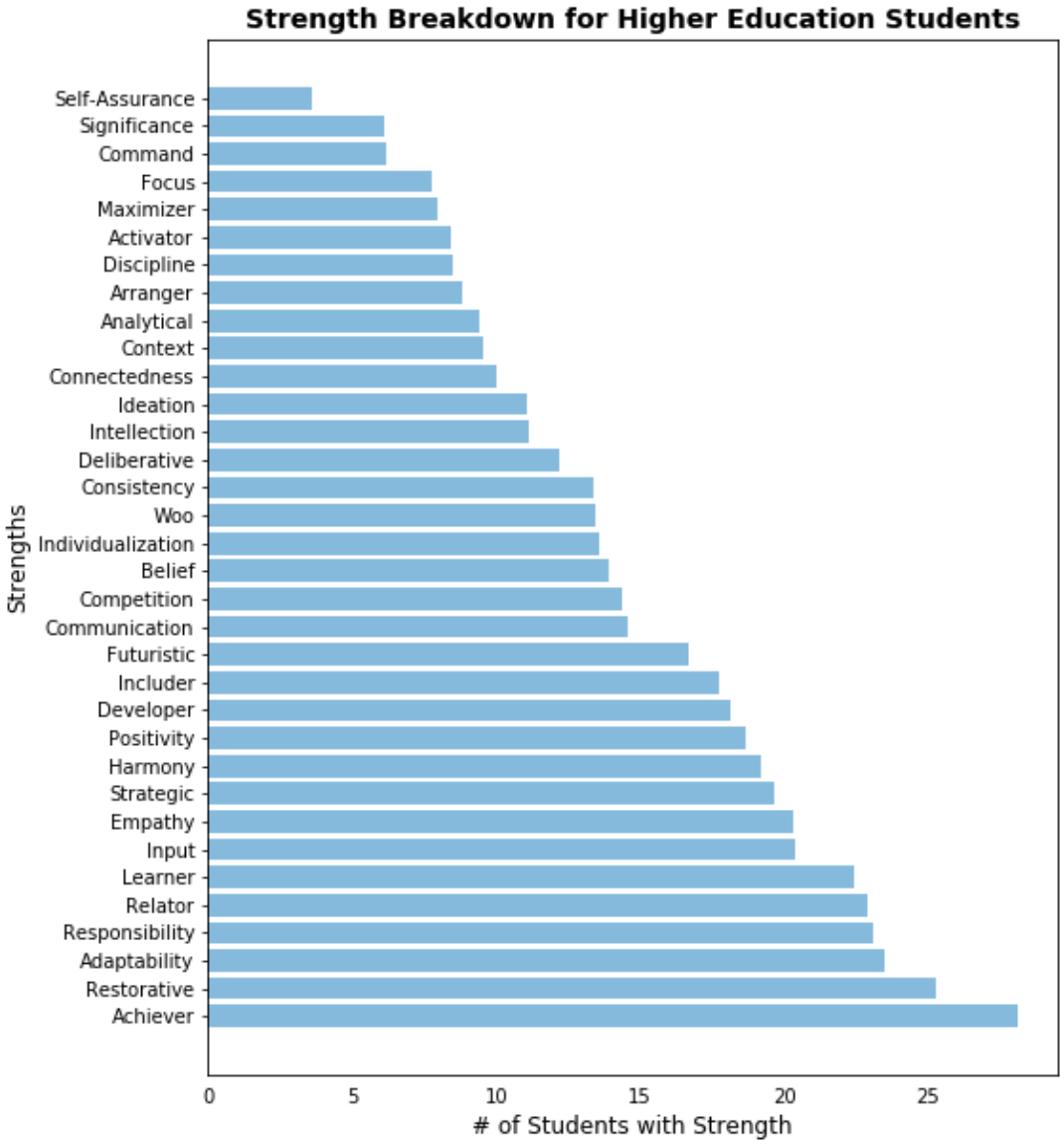


Figure 5: *CliftonStrengths breakdown for higher education students in general.*

there is a new chart of expected values used to calculate a chi-squared value, which yields a different result of 1493.13. The analysis also calculated the p-value extremely close to 0.0, and since the p-value is less than any given significance level, the null hypothesis must be rejected, which means the CliftonStrengths one has a significant relationship with what area they choose to study. The Cramer's V statistic is similar to the last analysis at 0.54, which indicates a moderate correlation between the two variables.

The 34 strengths can also be broken down into 4 domains, which reduces the contingency table (See Table 1), and therefore the degrees of freedom. The lower the degrees of freedom the more accurate the Cramer's V test. The chi-squared value is 1493.1301, p-value extremely close to 0.0, and Cramer's V statistic is 0.8164, which indicates a strong correlation between

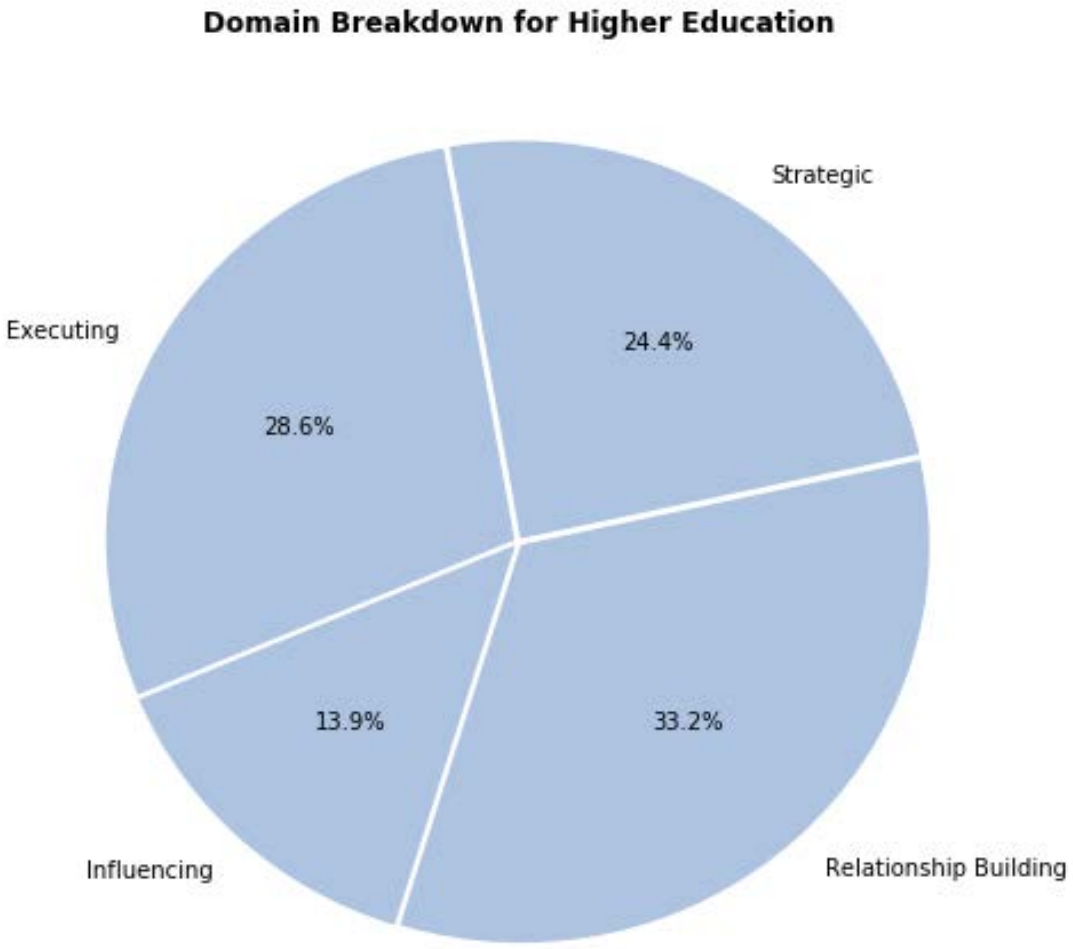


Figure 6: *CliftonStrengths breakdown for higher education students in general.*

	Executing	Influencing	Relationship Building	Strategic Thinking	Total
Environmental Design	33	18	43	41	133
Arts and Sciences	1686	948	2255	1921	4558
Leeds School of Business	730	775	966	624	787
CMCI	182	200	440	253	424
Education	28	32	86	44	119
Engineering	1197	643	1201	1444	1172
MULTU	1113	988	1415	1189	1906
Music	34	15	41	46	78
Total	5003	3619	6447	5562	9177

Table 1: *Contingency table for CliftonStrength Domains*

the two variables. Meaning, that the domain of one's strengths is strongly related to what one

chooses to study.

To perform the Kendall's Tau test for this analysis, the categorical variables must be replaced by number rankings. The ranking each variable is assigned does not matter, as long as the rankings are consistent. For example, "Achiever" must always have the ranking of 1 for all students in the study. The importance isn't what the number is but the order in which the numbers appear for each student's top five strengths. This test is meant to prove whether or not there is a common ranking/combination of top five strengths in each college. The Kendall's Tau correlation coefficients for each college all yielded results close to zero, which means there is no strong relationship or similarity among rankings.

V. Discussion and Conclusions

The results from this paper do not perfectly predict the optimal area of study for an individual based on their personality; however, it does give an insight into some statistically significant trends.

In summary, there is evidence that shows people with strengths such as "Competition" and "Woo" study business. People with strengths such as "Learner", "Achiever", and "Analytical" are correlated with engineering majors. People interested in educating the future minds of America tend to have strengths such as "Empathy", "Developer", "Realtor". People interested in working in media and communication tend to have strengths such as "Positivity", "Empathy", and "Communication". People studying Environmental Design tend to be "Restorative", "Strategic", and "Futuristic". However, there is no power combination of top strengths among any college. This means that although some strengths are strongly correlated with certain areas of study, that there is no magical combination of strengths that equates to the perfect or most common business or engineering school student etc.

Fall of 2019 was CU Boulder's first year collecting CliftonStrengths data in an organized and effective format. There was limited data to work with in this study, and many freshman have not declared their majors, but going forward, as more data is collected different more in depth analysis could be done. There could be a deeper look into correlations between areas of study with specific majors and programs and across all grade levels. There could also be a study that looks at the correlations between strengths and career paths by looking at internships and jobs students earn down the line.

References

[Fall Census, 2019] "Fall Census." Data Analytics, 22 Oct. 2019, www.colorado.edu/oda/student-data/enrollment/fall-census.

[Shaff, 2019] Shaff, Cori. "Fall2019." Excel, Spreadsheet, Sept. 2019.

[Gallup, 2019] Gallup, Inc. "Learn About the History of CliftonStrengths." Gallup.com, Gallup, 4 Oct. 2019, www.gallup.com/cliftonstrengths/en/253754/history-cliftonstrengths.aspx.

[Kearney, 2017] Kearney, Michael. (2017). Cramér's V. 10.4135/9781483381411.n107.

[Glen, 2017] Glen, Stephanie. "Kendall's Tau (Kendall Rank Correlation Coefficient)." Statistics How To, 14 Nov. 2017, www.statisticshowto.datasciencecentral.com/kendalls-tau/.

[McHugh, 2013] McHugh, Mary L. "The chi-square test of independence." Biochemia medica vol. 23,2 (2013): 143-9. doi:10.11613/BM.2013.018

[Gallup, 2016] Gallup, Inc. "CliftonStrengths Theme Frequency Report (November 2016)." Gallup.com, Nov. 2016, www.strengthsquest.com/198482/strengthsquest-theme-frequency-report-november-2016.aspx.

Augmenting Deep Reinforcement Learning on Toribash with Expert Matches

Yash Gandhi
University of Colorado Boulder
yash.gandhi@colorado.edu

Abstract

There are a variety of environments and simulators to test deep reinforcement learning algorithms. One of the less studied is a a game-based environment called Toribash. We believe Toribash to a useful platform for learning about common research problems in reinforcement learning such as high dimensional tabular input and complex control structures with highly coupled variables. Also, previous works in Toribash have focused on winning matches or evolving specific moves. Here, we focus on generating matches using deep reinforcement learning that behave similarly to human played matches. We do this by introducing augmentations informed by a data set of expert matches. Finally, we measure the results of our models using hidden markov models.[1]

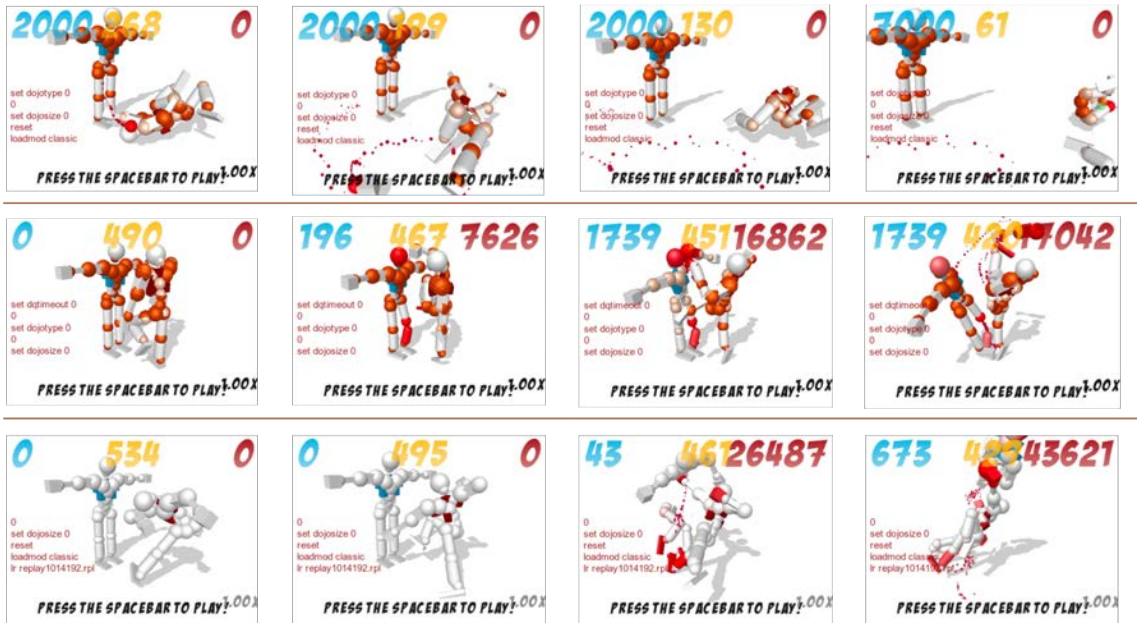


Figure 1: *We visualize certain frames from models and experts to compare their behaviors. (Top) A full multi-discrete model must learn all 22 joints simultaneously. This proves to be difficult for the deep reinforcement learning model and thus these frames demonstrate seemingly random actions and a losing strategy. (Middle) By limiting the action space to only 30 possible actions chosen from the set of expert trajectories, the agent learns to kick the opponent player. (Bottom) Human matches demonstrate much finer control and often have more build up to larger, more destructive actions.*

[1]The code is available at https://git.cs.colorado.edu/yaga6341/csci-4831-7000.

I. Introduction

Current applications of deep reinforcement learning (RL) vary from robotics, resource management, autonomous driving, and many others [1], [2], [3], [4]. Through the use of neural network, deep RL methods can process high dimensional inputs and produce skilled control. For example, Gu et. al. were able to train a robust robotic arm policy to open a door, Cheng et. al. were able to learn manipulation of occluded objects by controlling both the robotic arm and its gaze, and Zhu et. al provided fine control over multiple fingers for rotating valves, flipping boxes, and, again, opening doors [5], [6], [7]. Unfortunately, this can be a costly endeavour. Neural network approaches require many thousands of samples to learn and many more iterations to learn separable latent spaces. This drawback is magnified by the dynamics of an environment in deep RL and can make many robotic applications impractical. Thus, games can act as a simple, yet powerful proxy. Since game-based environments can be run indefinitely, researchers can tackle difficult research problems in simulation. The most common environment, the Arcade Learning Environment (ALE), offers a variety of platforms from the Atari 2600 to test genreal RL agents and models and demonstrated the abilities of the deep Q-Network (DQN) [8] [9], [10]. One such environment that has not been as intensely studied is Toribash. We believe Toribash to a another powerful testbed for deep RL and train many different augmentations to the original learning environment [11].

Toribash allows players to control a human-like figure consisting of 22 different control surfaces - joints. Each of the joints affects a different part of the body. Twenty of the joints can be in four possible states while two of the joints, hand grips, can be either on or off. Each turn, the player changes any subset of the joints and then moves the game forward a certain number of frames. The goal of the game is to fight a similarly controlled opponent and inflict the most damage [2]. Against a static opponent, this is a fairly simple task and can be learned without much modification to the original environment. Although, this results in nearly random movement that looks nothing like matches run by actual players. We use a number of techniques and methods to create action trajectories that are visually similar to matches played by humans.

II. Data

Deep RL relies heavily on the reward function because it dictates the direction of positive - and negative - growth. In certain situations, this can be implied by the goal of the agent or environment. If the goal is to reach a specific position, then inverse distance is an effective reward function. If the goal is only to win the game, the agent may learn to maximize the score. On the other hand, more complex goals can make engineering a reward function much more difficult. Even though the purpose of Toribash is to win the match, we want the agent to learn a constrained set of actions that are similar to the movements of human players. So, we used a set of expert trajectories to further improve our methods. Many of the methods we describe in section IV either use expert trajectories to reduce the effort of the learning algorithm or are inspired by observed trends among the many matches.

Toribash has a built in match saving feature and a large community of competitive players and match builders. The builders can create fights by moving the game a single frame at a time. Since the games development, many hundreds of thousands of replays have been shared among the community. For our project, we have chosen 47 matches that demonstrate a wide array of skill and precision. Each match averages around 1000 frames, or 1000 samples. Each frame represents a vector of values which we consider to be our state space. The state is a concatenation of player 1 and player 2's state

[2]We note that the game actually has hundreds of different game types and each with their own specific rule set. For this project we will only consider the basic rules defined at the start of the game.

values where each players state is comprised of different components shown in table 1.

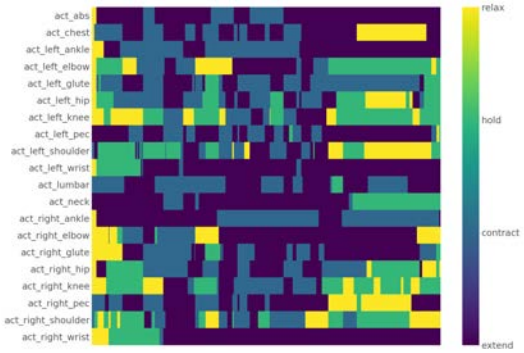


Figure 2: *Expert matches are highly dynamic and require constant fine tuning to all of the different joints throughout the duration of the game.*

From a heatmap of a specific expert match shown in figure 2, we can see exactly how dynamic the trajectories are. Futhermore, this demonstrates that a well played match consists of constantly fined-tuned actions and knowledge of how each joint will effect the overall motion of the Toribash agent. With regards to our learned agent, this implies that it must learn how every possible action can effect the outcome of the next step and of the match. To alleviate this intractable search space, we develop a set of tools that leverage these matches to simplify this problem in section IV.

III. Background

Reinforcement learning is commonly used for solving complex control task where the goal is to let an agent observe certain state values and choose an action guided by a reward signal. More concretely, RL is a solution to a Markov Decision Process (MDP).

$$\begin{aligned} R &: SxAxS \to \mathbb{R} \\ R(s_t, a_t, s_{t+1}) &= r_t \end{aligned} \tag{1}$$

An MDP is defined as a 5-tuple: (S, A, P, R, γ). The space of observations or states, S, represents the information given to our agent. The available control variables, or actions, and their domains are defined by A. An MDP exists within an environment, which is constrained by some transition function P defined by equation 2. This allows us to define the probability of moving to any other state given our current state and chosen action. The goal of the control task is represented as a reward signal function on the current state, action, and next state (equation 1). Finally, γ is a discount factor that weights the importance of future rewards. A larger γ will be more considerate of future rewards while a smaller γ will create a short-sighted RL agent.

$$P(s', a, s) = P(s_{t+1} = s' | s_t = s, a_t = a) \tag{2}$$

Solving an MDP with RL means finding some optimal control sequence that satisfies our goal. Formally, an agent's policy, π, must maximize our agent's reward. The expected return of a agents current policy is usually defined as a value function on the state shown in equation 3. The optimal policy is one that maximizes equation 3 over all possible states. In classical methods, the policy at any step s_t was a greedy algorithm on the plausible future states ranked using their value function. While the classical methods offer powerful solutions for some environments, other considerations make it difficult to use naively. First, the state space grows exponentially with the number of features. If any of the features are continuous, then the summation over possible states may be intractable. Futhermore, a greedy policy assumes memorization of all possible state and action policies which is infeasible, again, with a growing number of continuous state features and actions. This has inspired many different algorithms that replace the value and policy with neural networks [9], [12], [13], [14].

IV. Methods

Because of the complexity of this environment, we employ a number of different methods and techniques to simplify learning and generate

Components	Number of Components
Position X	21
Position Y	21
Position Z	21
Velocity X	21
Velocity Y	21
Velocity Z	21
Joints	22

Table 1: *A players state consist of seven different segments that define its position, velocity, and current joint configuration.*

more human-like actions.

$$V : S \to \mathbb{R},$$
$$V^{\pi}(s_t) = \mathbb{E}\left[\sum_i \sum_{a \in A} \gamma^i R(s_{t+i}, a_{t+i})\right] \text{ or}$$
$$V^{\pi}(s_t) = \max_a \left[r_t + \gamma \sum_{s'} P(s_t, a_t, s') V(s')\right] \tag{3}$$

I. PPO and TRPO

We use the proximal policy optimization (PPO) algorithm to train all of the deep RL models [14]. PPO is a method that simplifies the implementation of the trust region policy optimization (TRPO) algorithm [13], [14]. If the network parameters are rapidly changing, then it may take longer for the agent to learn how to properly control its environment, so both TRPO and PPO introduce a policy update that limits the change in the policy. TRPO introduces a hard constraint on the distance between the policies and their KL-divergence while providing monotonic updates [13]. On the other hand, PPO forgoes the theoretical monotonicity, and clips the ratio of new to old policies to both simplify implementation and also stabilize the TRPO criterion [14].

II. Reward Functions

Defining a reward function that promotes Toribash agents to learn human-like actions is a difficult design task. So, we tested different candidate functions against one another. Some of these functions leveraged the data set of expert matches, while others were based solely on observed heuristics. Here, we describe some of the candidate functions used during testing.

I Score Reward

The simplest reward function is one that maximizes the agent's in-game score. Although, the in-game scoring system can sometimes give unreasonably high scores, so to stabilize training we return the log of the score. Furthermore, we noticed that most experts would only change at most half of their joints during any one turn, so we append a L_1 penalty on the change in action. This reward type served as a baseline to measure against others and also served as the only sparse reward function. Most expert matches have a very sparse distribution of the score as seen in figure 3.

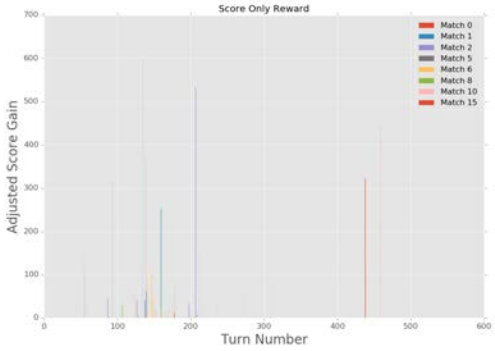


Figure 3: *In game score gains during different matches. Each spike represents a change in the score of the game.*

II Weighted Linear Reward

Because our goal is to produce policies that mimic human matches, we can leverage the saved replays to learn a reward function. This is a common method to learn a reward function given human demonstrations. This method, called inverse reinforcement learning, has an expansive literature on different methods to learn a function from previously demonstrated samples [15], [16], [17], [18]. For this project, we use a similar formulation as Hadfield-Menell et. al. and their Inverse Reward Design method [19]. In their paper, they define the reward as a linear combination of weights and a function on the state (top equation 4). We assume that $|phi$ is a function condition on some transformation matrix, and, here, we use the identity matrix for simplicity. To solve for the weight vector, w, we used the expert trajectories to solve for the normalized increase in score during a match rather than purely the score itself. We solved for w using ridge regression and by tunning the tradeoff parameter.

$$\begin{aligned} r(s_t) &= w^T \phi(s_t) \\ \phi(s_t; \mathbb{I}) &= \mathbb{I} s_t \end{aligned} \tag{4}$$

III Distance Reward

While exploring the expert trajectories, we noticed that the distance between the players and the score were often inversely correlated. Because of this observation, we introduced a reward function that negatively penalized the score reward with the distance between each players center of motion.

IV Curriculum Reward

Since the goal of Toribash - winning - is sparse, then we employ curriculum learning. Curriculum learning allows the agent to gradually learn harder tasks by changing the reward function during training. Originally introduced for classification techniques, this process can be thought of as learning gradually less smooth objectives [20]. We have four different segments based on previous reward functions that defines our curriculum. First, the agent should learn to win the game regardless of the performance. To make this a dense function, we penalize each step where the second player's score is higher than the agent's score. Next, the agent should learn to accrue more points. Instead of just adding the score to agent's reward, subtract the log of the difference between the score at time t and time $t+1$. This way, the agent has to learn how to regain its lost points and stops it from remaining stagnant. The third segment subtracts the distance to second player. This integrates the correlation we observed earlier and forces the agent to stay close to the second player. Finally, we add in the weighted linear reward to encourage movements similar to the expert trajectories.

III. Action Spaces

I Single Agent

With 22 different joints and four possible states for each joint, the space of total configurations for a single turn is exactly 4^{213}. Because of this, we introduced a few simple augments to the action space to make it more feasible to learn. First, instead of choosing from the entire space, just pick N random actions. While it drastically decreases the expressiveness, the RL model can more appropriately explore its own action space. Although, now, performance across models depends heavily on the random actions chosen. So, since we have matches played by experts, we can use that information to choose a more appropriate set of N actions. One method is to just pick the N most frequent actions from all the matches. Although, this can skew the actions towards beginning moves which are typically not useful for dealing damage to other opponent. Rather, most beginning

[3] We refer to full action space as Multi-Discrete.

moves represent a build-up phase. To remedy this, we can use a gaussian kernel that weights moves closer to the center of the match more than those at the beginning and end. Some matches use the end of a match to prepare a final pose rather than using those moves to deal damage. Although, these discrete spaces severely limit the space of possible actions.

Here, we describe a method that drastically decreases the number of actions the model has to learn, but still allows for access to the entire action space. For the continuous case, the model learns two actions x, y where both $x, y \in \mathbb{R}_+$. These two values represent a point on the cartesian plane. We can divide the plane into bins based on the total number of discrete actions and whatever bin the point falls into decides which action to take. Although, since our discrete action space consist of 4^{21} possible actions, iterating over each one can become difficult. So, we employ the following procedure to greatly speed up this calculation. First, we can calculate the bin number because the total number of actions is known. Then, we calculate the base 4 value of that number. Since each joint can be in only one of four possible configurations, we can take the digits of the base 4 value to be the next configurations for the joints.

II Multi-Limb Model

Instead of trying to solve the entire problem with a single model, we also tried to compartmentalize the action space by segmenting the Toribash figure. The agent is first broken up into six different sections: left and right leg, left and right arm, and upper and lower body. Each one of those networks is only responsible for a small number of actions[4]. A higher level model receives the normal state input and sends a K dimensional signal to each lower level. The limbs learn to interpret this signal and send back commands for their respective components. The higher level aggregates the actions and sends it to the environment.

$$r(s_0) = c_0 P(s_0|\sigma) - c_1 \tag{5}$$

There are two possible routes to training this set of models. One is to train all of the models at once. This is computationally difficult because it essentially amounts to training seven different networks - six lower levels and one higher level. Also, it requires a more complex communication structure between the models. The other way is to to train the limbs first and then train the higher level. The limb models can be trained by sending random permutations of signals to the lower level and then measuring how well they interpreted the signal. During training, each limb model receives some random signal as a vector $\sigma \in \mathbb{R}^4_+$. The first two values of σ are the center of a 2D distribution and the second two values are the diagonal of the covariance matrix. The lower limb models output a x and y value, similar to the continuous actions defined for the single agent model, and that defines a configuration of each limbs respective joints. Although, since each model is only concerned with a few joints, there is less granularity for each bin. The reward for the lower limb models is the probability of the chosen point given the distribution defined by the signal (equation 5, figure 4).

The controller model, now, acts as an intermediary between the limbs and the state input receiving rewards based on performance in the game.

III Embedded Models

In the game Toribash, based on domain expertise and observing the expert trajectories, there is a clear bias for certain joints for larger moves and certain joints for finer control. We incorporated this thinking into another model type we call the embedded model. Here, we train subsets of the joints in a gradually embedded style. First, we train joints that are used for larger movements: turning the entire body, jumping, rotating entire limbs. Then, we train

[4] All of the models either controlled 3 or 4 joints which means 64 or 256 actions for any one model

a separate set of joints that are used for a finer set of attack moves: punching, kicking. Finally, we noticed that a smaller set of joints were used for precision control to increase the advantage during the matches. These models are the major, minor, and detail models respectively. The major model is trained completely independently using any of single agent action types from direct state input and any reward function. Next, we train the minor model, again with any of the single agent actions and any of the reward functions, but also include the trained major model. Finally, the detail model is trained in the same way, but with both the major and minor models.

IV. Stochastic Similarity

Since the score of the match does not evaluate the components of the trajectory generated by the model and how similar it is to human players, we use a different measure for success of the model. We implement a stochastic similarity measure as described by Nechyba and Xu [21]. This paper uses a pair of Hidden Markov Models (HMM) to determine the similarity of two observations. A HMM allows us to reason about the probability of sequences given influence by some set of hidden states. A HMM, H, is defined as a 4-tuple given by equation 6. The Q represents the states of the model, Π is the distribution of the initial states where each $\pi_i \in \Pi$ is the probability of starting in state $q_i \in Q$, and O is the sequence of observations seen by the observer. Here, $\lambda = (A, B)$ where A is the transition matrix such that the probability of going from state q_i to state q_j is a_{ij} and B is the likelihood of emitting observation o_t at state q_i or $b_i(o_t)$.

$$H = (\lambda, Q, \Pi, O) \tag{6}$$

It is possible to learn the transition probabilities, A, and the emission probabilities, B, by using the forward-backward (Baum-Welch) algorithm and a set of observations [22]. Before measuring similarity across different experimental runs, Nechyba and Xu process their multi-dimensional trajectories using a codebook [21]. We use a similar strategy, but before discretizing the data, we reduce the dimensionality of our data using principle component analysis (PCA). Then, we use a K-Means algorithm to generate our codebook. We perform this process separately for the data from the experts and the data generated from executing a learned policy. We use these discretized data sets to learn a HMM for each of the models: human players and learned policy. From here, we sample matches from both sets and measure the similarity using equation 7 where α represents either the generated match or expert match and β represents the HMM learned from either the generated matches or expert matches.

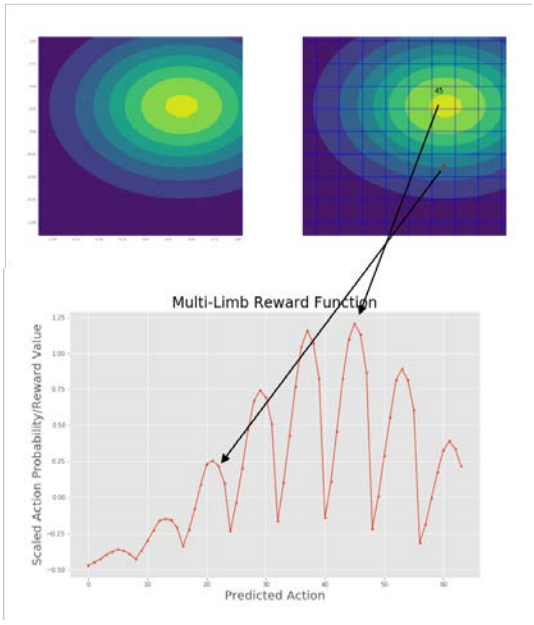


Figure 4: *The upper left image represents a distribution defined by some signal. The upper right image shows that same distribution, but with bins placed on top of the distribution. The red dot represent the action taken by the limb model. The lower graph shows what the reward values is for the given point and what it would have been if the point made it closer to the signaled bin.*

$$P_{\alpha,\beta} = 10^{\frac{Pr(\alpha|\beta)}{|\alpha|}}$$
$$\sigma(O_1, O_2) = \sqrt{\frac{P_{12}P_{21}}{P_{11}P_{22}}} \tag{7}$$

The final score is an average over the number of samples. A detailed outline of this algorithm can be found in the appendix A.

V. Results

Here we compare the many different model types, action types, and rewards against one another. Each of the single agent methods and the multi-limb model were trained for 100,000 iterations. For the multi-limb environment, each individual limb was trained until the last limb showed convergence on its reward value at about 75,000 iterations. To be able to compare models against one another, reward parameters were fine-tuned for the first model in which they appeared and then propogated to other models. For example, constant multipliers tuned in the score only reward for random actions were the ones used in the random actions model with a curriculum reward. We report similarity scores for all of these models in table 2.

We can see from the table that the curriculum reward well at creating policies with high stochastic similarities. One particular example of the curriculum reward that has a higher similarity to the human matches is the weighted frequent actions model. When observing this model, it had learned to kick the opponent player well enough to detach its arm. This example can be seen in the middle row of figure 1. From the table, we can see that certain reward functions tended to perform poorly. Most models using the weighted linear reward function alone failed to even cross a similarity of 0.001. Often, though, we would see many of the agents learn to deal a small amount of damage and then either retreat away or stay stationary on the ground.

VI. Conclusion

The values in table 2 demonstrate that many of the methods described fall short of generating human-like actions. Curriculum rewards, though, were the most useful for accomplishing this goal. These models were able to learn through a more stable process rather, which allowed for the agent to learn a more stable set of actions to take. The multi-limb environment obtained the worst results and we believe it to be the fact that the upper level controller had to learn a general enough signals for all of the models to interpret. In future work, we would have a upper controller that sends an individualized signal to each of the lower level limbs. Stable models like the embedded model and the continuous model all demonstrated similar behaviors across the reward functions. This leads us to believe that the reward engineering was a much more vital part of learning.

While still a relatively new environment for testing deep reinforcement learning methods on, certain aspects of Toribash have been studied before. Kanervisto and Hautamäki created the original environment and trained some simple agents on a variety of different task [11]. They built a communication strategy to allow for state-of-the-art algorithms to interface with the original game software and also tested some initial competitive training environment like self-play and playing against human players [11]. Other works also include some genetic algorithm work to try and generate offensive opening moves [23]. These papers further prove that Toribash is a good environment for training learning agents. While our focus has been on generating policies to play in the standard game, Toribash offers a massive set of further modifications developed by community members.

Learning general agents is a costly endeavour and games offer a simple way to approximate common research problems while being able to be run indefinitely. We have shown that Toribash offers a number of desirable properties. Toribash is an interesting single agent platform with high dimensional input and con-

Action Type	Score	Weighted Linear	Distance	Curriculum
Multi-Discrete	0.004	0.030	0	**0.058**
Random	0.021	0	0.012	**0.067**
Frequent	0.057	0	0	**0.089**
Weighted Frequent	0	0	0	**0.273**
Continuous	0.094	0.120	**0.203**	0.001
Multi-Limb	0.001	0.001	**0.003**	0
Embedded Model	0.022	0	0.072	**0.137**

Table 2: *Stochastic similarity values for all of the single agent policy methods and the multi-limb method using all of the different reward types. Scores closer to 1 represent more similar models. Each model was trained for approximately 100,000 iterations and each learned policy was run for 50 episodes for PCA, K-Means, and HMM training. The models were tested on 25 random samples pairs from the expert matches and generated runs. Any score lower than 0.001 was floored to 0 for the purposes of this table.*

trol. It can be sectioned off into components to test multi-agent learners. By choosing specific subsets of actions, a hierarchical model can be trained where each level can either be independent or dependent. The extensive list of mods available make it simple to train a Toribash model in a new scenarios with meta-learning algorithms. And, while the basic goal of the game is to win, the purpose of generating human-like can be useful for training more complex goal oriented methods.

References

[1] Yu Fan Chen, Michael Everett, Miao Liu, and Jonathan P How. "Socially aware motion planning with deep reinforcement learning". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 1343–1350.

[2] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 3389–3396.

[3] Ning Liu et al. "A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning". In: *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE. 2017, pp. 372–382.

[4] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. "Benchmarking deep reinforcement learning for continuous control". In: *International Conference on Machine Learning*. 2016, pp. 1329–1338.

[5] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 3389–3396.

[6] Ricson Cheng, Arpit Agarwal, and Katerina Fragkiadaki. "Reinforcement Learning of Active Vision for Manipulating Objects under Occlusions". In: *arXiv preprint arXiv:1811.08067* (2018).

[7] Henry Zhu, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar. "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3651–3657.

[8] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. "The ar-

cade learning environment: An evaluation platform for general agents". In: *Journal of Artificial Intelligence Research* 47 (2013), pp. 253–279.

[9] Volodymyr Mnih et al. "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602* (2013).

[10] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (2015), p. 529.

[11] Anssi Kanervisto and Ville Hautamäki. "ToriLLE: Learning Environment for Hand-to-Hand Combat". In: *arXiv preprint arXiv:1807.10110* (2018).

[12] Timothy P Lillicrap et al. "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971* (2015).

[13] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. "Trust region policy optimization". In: *International conference on machine learning*. 2015, pp. 1889–1897.

[14] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[15] Andrew Y Ng, Stuart J Russell, et al. "Algorithms for inverse reinforcement learning." In: *Icml*. Vol. 1. 2000, p. 2.

[16] Pieter Abbeel and Andrew Y Ng. "Apprenticeship learning via inverse reinforcement learning". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 1.

[17] Faraz Torabi, Garrett Warnell, and Peter Stone. *Behavioral Cloning from Observation*. 2018. eprint: arXiv:1805.01954.

[18] Jonathan Ho and Stefano Ermon. "Generative adversarial imitation learning". In: *Advances in neural information processing systems*. 2016, pp. 4565–4573.

[19] Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart J Russell, and Anca Dragan. "Inverse reward design". In: *Advances in neural information processing systems*. 2017, pp. 6765–6774.

[20] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. "Curriculum learning". In: *Proceedings of the 26th annual international conference on machine learning*. ACM. 2009, pp. 41–48.

[21] M. C. Nechyba and Yangsheng Xu. "Stochastic similarity for validating human control strategy models". In: *IEEE Transactions on Robotics and Automation* 14.3 (1998), pp. 437–451. DOI: 10.1109/70.678453.

[22] Frederick Jelinek, Lalit Bahl, and Robert Mercer. "Design of a linguistic statistical decoder for the recognition of continuous speech". In: *IEEE Transactions on Information Theory* 21.3 (1975), pp. 250–256.

[23] Jonathan Byrne, Michael Neill, and Anthony Brabazon. "Optimising offensive moves in toribash using a genetic algorithm". In: (Jan. 2010).

A. Stochastic Similarity Algorithm

Here we provide the stochastic similarity algorithm for measuring model performance.

Algorithm 1: Stochastic Similarity

Result: Similarity between trained model and expert matches

$\xi_E \leftarrow$ EXPERT TRAJECTORIES
$L \leftarrow$ NUMBER OF SAMPLES TO AVERAGE
$C \leftarrow$ K-MEAN CLUSTERS
$d \leftarrow$ NUMBER OF HIDDEN STATES
$h \leftarrow$ MODEL
$\xi_G \leftarrow [\,]$
for $k \in K$ EPISODES **do**
  $\tau \leftarrow [\,]$
  for $t \in T$ TIMESTEPS **do**
    $\tau_t \leftarrow h(s_t)$
  end
  $\xi_G^k \leftarrow \tau$
end
/* $d << |S|$ where S is the state space */
$\mathbf{P}_e = \text{PCA}(\xi_E, d)$
$\mathbf{P}_g = \text{PCA}(\xi_G, d)$
$\mathbf{K}_e = \text{KMEANS}(\mathbf{P}_e, C)$
$\mathbf{K}_g = \text{KMEANS}(\mathbf{P}_g, C)$
/* $\mathbf{K}_{e/g} \doteq \mathbf{N}^{Kx1}$ */
$\lambda_e \doteq \text{BAUM-WELCH}(\mathbf{K}_e)$
$\lambda_g \doteq \text{BAUM-WELCH}(\mathbf{K}_g)$
$s = 0$
for $l \in \{1, 2, ..., L\}$ SAMPLES **do**
  $O_e = \mathbf{K}_e^l$
  $O_g = \mathbf{K}_g^l$
  $P_{11} = 10^{log\left(\frac{P(O_e|\lambda_e)}{\text{LEN}(O_e)}\right)}$
  $P_{12} = 10^{log\left(\frac{P(O_e|\lambda_g)}{\text{LEN}(O_e)}\right)}$
  $P_{21} = 10^{log\left(\frac{P(O_g|\lambda_e)}{\text{LEN}(O_g)}\right)}$
  $P_{22} = 10^{log\left(\frac{P(O_g|\lambda_g)}{\text{LEN}(O_g)}\right)}$
  $s \leftarrow s + \sqrt{\frac{P_{12}P_{21}}{P_{11}P_{22}}}$
end
return $\frac{s}{L}$

Voting Equity in the 2018 General Election in Wisconsin

Steven L Hobbs
University of Colorado Boulder
steven.hobbs@colorado.edu

Abstract

In the state of Wisconsin, the 2016 presidential election was decided by 22,748 votes in a state of 5.8 million. In cooperation with the Native American Rights Fund, this study sought to determine if Native Americans in the state of Wisconsin, a group of currently about 69,000, have been disadvantaged or discriminated against by the distribution and specific locations of polls in the most recent 2018 general election. Towards this end, travel duration by car between block group population centroids and 2018 general election polling locations were calculated and travel time to polls estimated for Native American, Asian, Black and African American, Hispanic or Latino Origin and White racial groups. While this study found significant differences in median travel times to the nearest polling locations between racial groups ($p < .001$), those differences are small, generaly less than 1 minute, and appear to be primarily explained by differences in land area and population densities across block groups, which together accounted for all but 1% of the explainable variation in travel time to the nearest polls. After accounting for block group land area and population density, no racial group appears to be grossly disadvantaged by the location and distribution of polling locations. While inequities and discriminatory polling locations may still exist in the state of Wisconsin, this study suggest that such phenomenon are not observable at the block group level of analysis.

I. Introduction

Recent years have been characterized by the passing of highly contentious rules and legislation pertaining to voting in national elections. For example, in April 2017 House Bill 1369 passed in North Dakota requiring citizens who wish to vote to present a valid ID with a residential street address. Homes and other buildings on tribal lands and reservations often lack both street names and addresses, rendering such identification cards unattainable for many Native Americans. While Bill 1369 can be argued to prevent fraudulent voting practices and protect the integrity of elections, no evidence exists that voter fraud related to voter-ID actually occurs, and that laws such as Bill 1369 are passed strategically to exclude segments of the population from the voting process.

The location of polls on election day is also a concern for many Native Americans. In Alaska, Nevada, Utah, Arizona and New Mexico Native Americans have reportedly been forced to drive 34 (New Mexico) to 163 miles (Utah) to reach the nearest polling location [Native News Online.Net]. Absentee ballots are not necessarily a solution to long travel times as many reservations don't receive regular mail, or sending and recieving mail may still require a long drive to the nearest post office. Furthermore, one could argue that if the US government is to provide physical polling locations to Americans for convenience or to encourage voting or for any other reason, such polling locations should be distributed as equitably as possible and so as to avoid marginalizing any group of people.

In the state of Wisconsin the 2016 presidential election was decided by approximately one third of the State's current Native American population. Given the potential for Wisconsin Native Americans to influence the impending 2020 presidential election with their vote, this study sought to determine if the location and distribution of polling locations in the most recent 2018 general election, served to disadvan-

tage Native Americans, or other racial groups, primarily by imposing long travel times to the nearest polling locations.

II. Data

Population Estimates and Physical Geography

2017 American Community Survey (ACS) 5-year population estimates were obtained for 4,489 block groups in the state of Wisconsin from the US Census bureau application programming interface (API) [ACS, 2015] using the tidycensus R package [tidycensus]. The population estimates included groupings for all demographics combined, white alone, American Indian and Alaskan Native alone (Native Americans), black and African-American alone, Asian alone and Hispanic or Latino origin alone. Population estimates of people claiming two or more races were not included.

Physical geography including land area and water area, as well as 12 digit geographical identifier (GEOID) numbers for each block group were obtained from TIGER/Line® 2017 shapefiles for Wisconsin downloaded from the US Census Bureau [TIGER/Line Shapefiles, 2017]. These files were analyzed using open source QGIS software and the data merged with the 2017 ACS population estimates.

Polling Locations

Polling locations for the 2018 General Election in the state of Wisconsin were downloaded as an .xlsx file from the Wisconsin Elections Commission [WEC, 2018]. The location information included street addresses, latitude and longitude. Although 3,681 polling locations were obtained, only 2,526 involved unique addresses and unique latitude and longitude coordinates. Polling locations with identical addresses and coordinates were regarded as a single polling location for the purposes of this analysis.

Nearest Polls for each Block Group

Population centroids for all 4489 Wisconsin block groups were downloaded from the US Census Bureau [Centers, 2010]. Because population centroids are calculated from full census data, block group centroids are based on 2010 U.S. census data. For each block group centroid, the euclidean distance to every polling location was calculated and the three nearest polling locations for each block group were retained for further analysis.

Shortest Poll Travel Duration for each Block Group

The list of block group centroids and nearest polling locations were submitted to the Google Maps Platform API [Google Maps] to obtain travel duration and travel distance between each centroid and the three closest polling locations. For each block group centroid, the polling location with the shortest travel duration was retained. 13 block group centroids were eliminated for failing the Google API query because they exist entirely over water (e.g. lake Superior, lake Michigan) and have no residents. 4 additional block group centroids were eliminated for having no population and were composed mainly of water areas (e.g. lake Menona, Lake Mendota). The resulting 4,472 block group centroids were connected to 1,970 unique polling locations, each of which was determined to have the shortest travel duration to at least one block group. 555 polling locations were not the closest polling location to any block group centroids, but were considered along with all other unique polling locations when counting the number of polling locations per block group.

Geolocating Polls and Block Group Centroids

Out of 2,526 unique polling locations, 2520 were geolocated within ACS block groups using Shapely[Shapely] and the NAD 27 Wisconsin Central coordinate system (EPSG:32053). 6 polling locations were not found within shapefile block group boundaries, and were manually located to block groups using the US Census Geocoder[Geocoder]. Population centroids were localized to block groups by combining their state, county, tract and block group iden-

tifiers into 12-digit block group GEOID numbers.

To identify block groups with polls on Native American Lands, the 2,526 unique polling locations were compared to shapefiles for all American Indian and Native Alaskan lands in the United States in 2017 downloaded from the US Census Bureau [TIGER/Line Shapefiles, 2017]. The polling locations that exist on Native American lands were geolocated to 2017 ACS block groups using Python's shapely package [Shapely]. The total 2017 Native American land area in Wisconsin was also obtained from these shapefiles.

All block groups were given a categorical label based loosely on population density. Urban bock groups were those that intersected with urban area or urban cluster polygons as defined in the US Census Bureau's shapefile for US urban areas[TIGER/Line Shapefiles, 2017]. Because the Census Bureau's urban area polygons do not follow block group boundaries, if any part of a block group intersected with an urban area, the entire block group was designated as urban. The remaining block groups were designated as rural or remote, a subset of rural with population densities less than or equal to 100 residents per square mile.

III. Statistical Methods

To examine differences between duration of travel to polls between races, the nonparametric Kruskal Wallis test was performed followed by Bonferroni corrected pairwise Wilcoxon-rank sum tests. Both tests were conducted for urban, rural and remote block groups. Additionally, and to control for the influence of land area and population density, multiple regression was performed with land area, population density and population estimates for each race as predictors of the duration of travel to polls. Multiple regression equations were performed for urban, rural and remote categories of population density. Because our analysis only seeks to characterize the Wisconsin population using census data, rather than to make predictions or generalize to a larger population, mild violations of assumptions (heteroscedasticity and non-normally distributed residuals) were ignored.

IV. Results

Block Group Population Distributions

The ACS 5-year 2017 Wisconsin population estimate is 5,763,217 and is composed of 50,094 (.9%) Native American only, 152,325 (2.6%) Asian only, 365,884 (6.5%) Black or African American only, 380,590 (6.6%) Hispanic or Latino origin only and 4,950,577 (85.9%) White only residents. Histograms showing population distributions for each racial group across all of Wisconsin as well as within rural and remote block groups are displayed in figure 1. Notable observations include the relatively high concentration of Native Americans in the lowest category of population density, remote, while Black and African American and to a lesser extent Asian, show a stronger preference for urban block groups.

Population estimates, block group means and standard deviations for each race at each block group density category are shown in table 1. By percent of the total single-race population, the White demographic is clearly dominant in urban, 84%, rural, 91%, and remote, 95%, block groups. Nearly a third of the Native American only population live in remote block groups, where they comprise 2.6% of the population. All other demographics have less than 10% of their population in remote block groups, while Black and African American and Hispanic or Latino origin have around 1% of their population in remote blocks. White only and Native American only racial groups make up higher percentages of the remote populations, while Asian, Black and Hispanic or Latino origin make up higher percentages of the urban and rural populations.

Travel Time to Polls

Histograms for travel time to polls in minutes for each demographic are shown in figure 2. While the distribution centers are similar

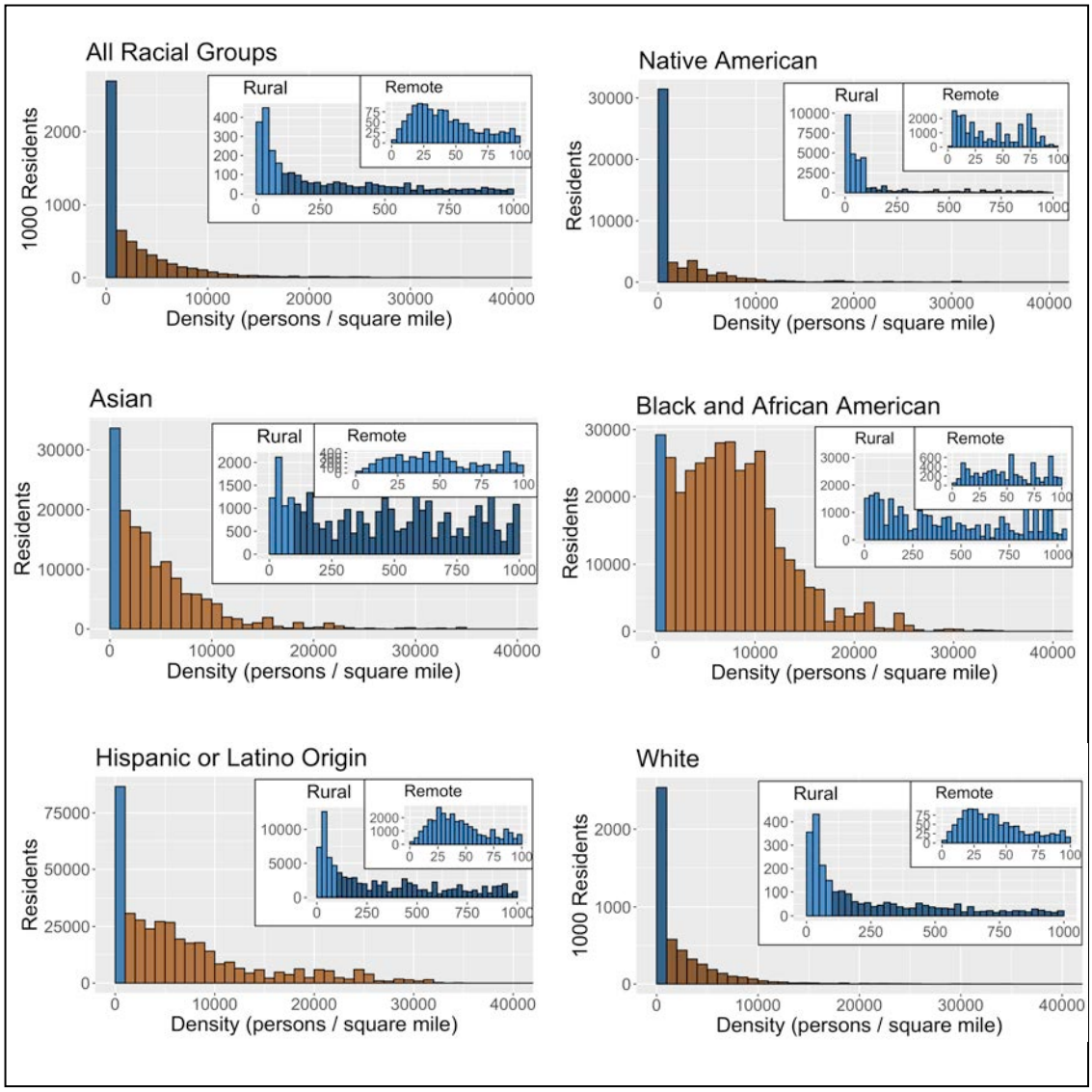


Figure 1: *Population distributions for all racial groups across for all Wisconsin and for rural and remote block groups.*

across all demographics, bimodality is suggested for Native Americans where a positively skewed sub-population shows longer travel time to the nearest poll location. While all demographics show positive skew, the Black and African American population has notably less representation in the longer travel times, possibly reflective of a more urban distribution within block groups.

Table 2 shows data on the number of polls per 1000 residents and per 100 square miles, as well as the number of residents per poll. Over the entire state of Wisconsin, the number of polls per 1000 residents has a mean of .50 and SD .67. This ratio decreased for urban (mean = .41, SD = .67) and rural block groups (mean = .44, SD = .67), but increased substantially for remote block groups (mean = 1.23, SD = .95). The mean number of residents per poll followed a similar trend, with 2285, 2815, 2633 and 868 residents per poll in all Wisconsin, urban, rural and remote block groups respectively. Stan-

	All Races	Native American	Asian	Black or African American	Hispanic or Latino	White
All						
Total, (%)	5,763,217	50,094 (.9)	152,325 (2.6)	365,884 (6.3)	380,590 (6.6)	4,950,577 (86.9)
Block Group Mean (SD)	1,288 (642)	11 (62)	34 (79)	82 (205)	85 (154)	1,107 (636)
Urban						
N, (% of total)	4,606,027	31,434 (.7)	136,442 (3.0)	346,075 (7.5)	337,510 (7.3)	3,879,119 (84)
Block Group Mean (SD)	1,296 (660)	9 (43)	38 (84)	97 (222)	95 (163)	1,092 (658)
Rural						
Total	579,376	3,729 (.6)	13,756 (2.4)	17,518 (3.0)	29,463 (5.1)	524,660 (91)
Block Group Mean (SD)	1,149 (398)	30 (188)	4 (9)	5 (12)	27 (41)	1,087 (384)
Remote						
Total	577,814	14,931 (2.6)	2,127 (.4)	2291 (.4)	19,617 (3.4)	546,798 (95)
Block Group Mean (SD)	1,149 (398)	30 (188)	4 (9)	5 (12)	27 (41)	1,087 (384)

Table 1: *2017 Population estimates, block group means and standard deviations (SD) by race for all Wisconsin and at urban, rural and remote population densities.*

dard deviations for residents per poll could not be calculated because many block groups had no polling locations.

The density of polling locations more than doubled going from urban to rural block groups at 6.1 and 14.5 polls per 100 square miles respectively, but dropped substantially for remote blocks groups at 2.6 polls per square mile. Given the modest increase in travel time to the nearest poll for remote block groups (table 3 , the polling locations in remote block groups are likely to be closer to population

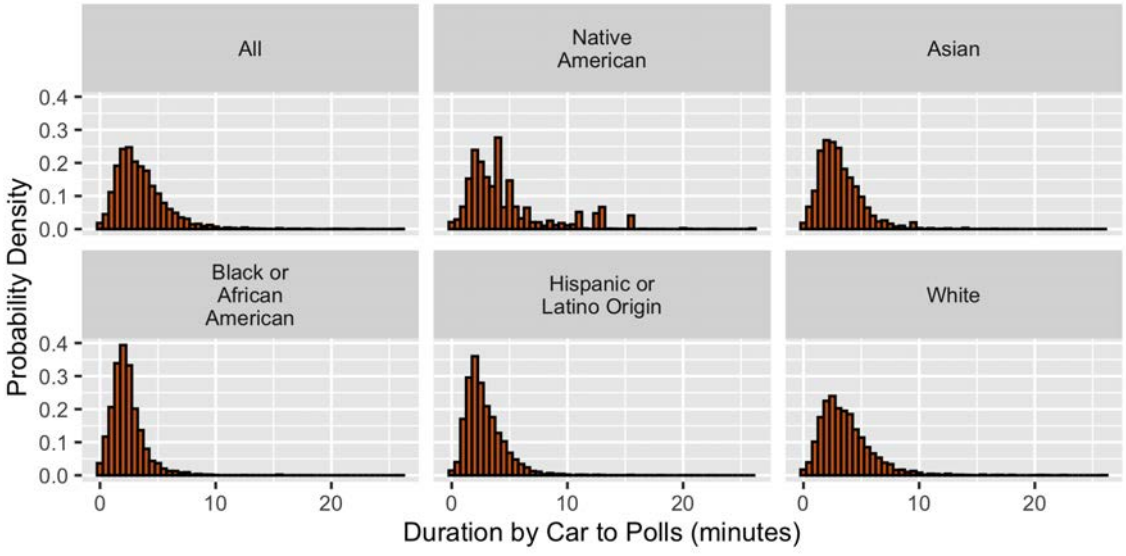


Figure 2: *Probability density plots showing the travel time by car to the nearest poll in minutes across all Wisconsin and for each racial group.*

Density	Polls per 1000 residents	Polls per 100 square miles	Residents per poll
All	.50 (.75)	4.7	2285
Urban	.41 (.67)	6.1	2815
Rural	.44 (.67)	14.5	2633
Remote	1.23 (.95)	2.6	868

Table 2: *The mean and standard deviation (SD) for the number of polling locations per 1000 residents for all Wisconsin and at each block group population density.*

centroids than in rural and urban densities.

Table 3 shows the mean and standard deviation of the shortest travel time to polls for each demographic for the entire population and at each block population density. As parametric statistics on these data were not possible, means are provided for descriptive purposes only. While small differences exist between racial groups, the overall trend is clearly for longer travel times to polls with decreasing population density.

Table 4 shows median values for the shortest travel time to polls (travel times) for all races across all of Wisconsin and at all population densities. Across all Wisconsin, Native Americans had the longest median travel time to polls at 3.76 minutes, followed by Whites (3.25 minutes), Asians (2.82 minutes), Hispanic or Latino origin (2.48 minutes) and Black or African American (2.13 minutes). Median travel times for all pairwise comparisons between races were statistically significant ($p < .001$).

In figure 3, boxplots for travel times by race for all Wisconsin show that while all races are similar in their overall distribution characteristics, the small but statistically differences in medians between races are visible, as are differences in the interquartile ranges, particularly for Black and Hispanic or Latino origin, presumably reflecting the concentration of these demographics in urban areas with shorter distances to polls.

Median travel times in urban and rural populations were qualitiatively and quantitiavely very similar to those across all of Wisconsin, with the exceptions that Native Americans have shorter travel times in urban (3.28 minutes) and rural block groups (2.71 minutes) and Asians have notably shorter travel times in remote block groups (2.28 minutes). The most notable differences appear at the remote block group level, with sizable increases in travel times for all demographics (table 4).

Statistically significant differences in median travel times over the entire population were maintained at the urban block group level ($p \leq .001$). However, in rural block groups significant differences between White, Native American and Asian races disappeared ($p \geq .83$), while in remote block groups significant differences between White and Native American and between Asian and Hispanic or Latino disappeared ($p \geq .32$). Qualitative differences in travel time medians, interquartile ranges and other distribution characteristics are visible in figure 4.

Regression analysis results for travel duration predicted by population density, land area and population estimates for each race at each level of population density, and for block groups with poll travel time greater than 10 minutes are presented in table 5. Regression models with land area and population density only as predictors explained 20%, 10% and 20% of the variation in poll travel time for urban, rural and remote block groups respectively and the models were highly significant ($p<.001$). Adding population estimates for each race yielded a very small increase in predictive power of 2% and 3% for urban and remote block groups respectively and made no difference for rural block groups. Population estimates for individual races had mostly non-significant influences on nearest poll travel time, with the exception that increasing white population estimates slightly increased near-

	All Races	American Indian	Asian	Black or African American	Hispanic or Latino	White
All	3.55 (2.24)	4.65 (3.59)	3.16 (1.89)	2.36 (1.44)	2.84 (1.69)	3.67 (2.27)
Urban	3.28 (2.05)	3.52 (2.69)	2.94 (1.86)	2.24 (1.39)	2.61 (1.59)	3.33 (2.08)
Rural	5.16 (1.91)	7.01 (1.35)	5.44 (1.78)	4.94 (1.7)	4.89 (1.38)	5.11 (1.92)
Remote	5.16 (3.23)	7.01 (4.43)	5.44 (3.11)	4.94 (3.12)	4.89 (2.93)	5.11 (3.19)

Table 3: *Mean and standard deviation for the shortest travel time to the nearest polling location in minutes for all Wisconsin and at each block group population density.*

	All Races	Native American	Asian	Black or African American	Hispanic or Latino	White
All	3.08	3.76	2.81	2.13	2.48	3.25
Urban	2.97	3.28	2.82	2.11	2.43	3.1
Rural	3.08	2.71	2.28	2.28	2.53	3.12
Remote	4.68	5.84	5.11	4.52	4.64	4.67

Table 4: *The median shortest travel time to the nearest polling location in minutes for all Wisconsin and at each block group population density*

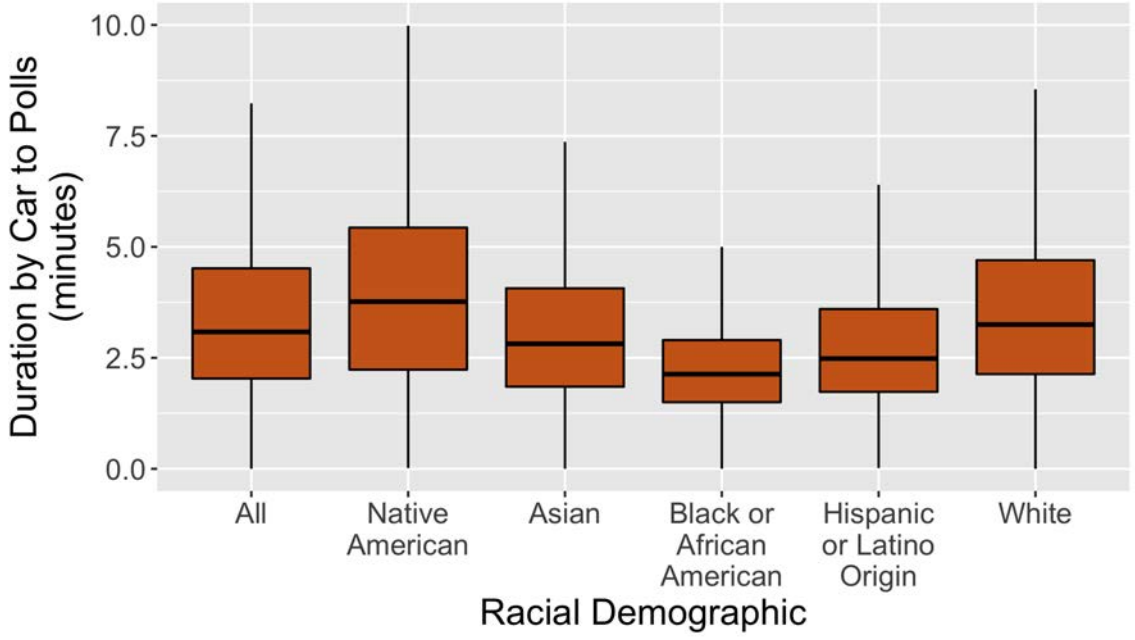


Figure 3: *Boxplots for the shortest travel duration (minutes) to nearest polling location in all Wisconsin.*

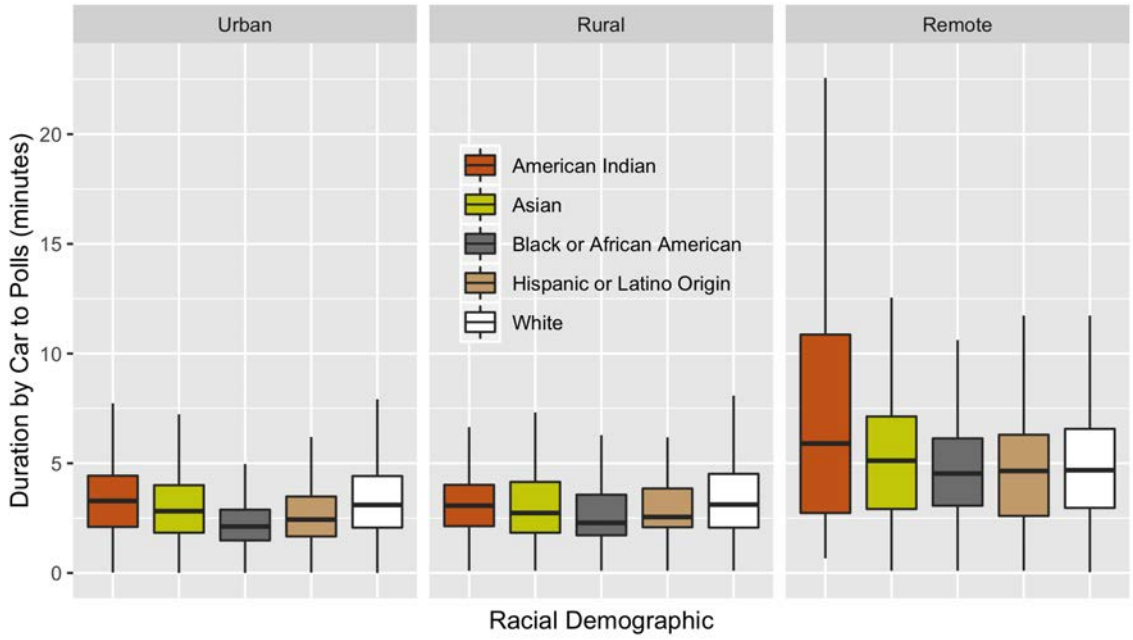


Figure 4: *Boxplots for travel time in minutes to the nearest polling location by race and by block group population density.*

est poll travel time in urban (B = 0.11, p<.001) and rural block groups (B=0.023, p = .015), but decreased nearest poll travel time in remote (B = -.10,p<.001) block groups. Additionally, in urban block groups, increasing estimates of Black and African Americans predicted a small decrease in nearest poll travel time (B = -.029, p=.002).

Regression analysis were also performed on a subset of 72 block groups with a travel time to the nearest poll of more than 10 minutes. Within these block groups, the average population density is 100 persons per square mile, 34 fold less than the average block group population density of Wisconsin at 3,411 persons per square mile. Native Americans are concentrated in these longer travel time block groups, representing 5.4% of the single-race population, an increase of 632% compared to their representation in all of Wisconsin. However, after controlling for land area and population density, population estimates of all races had no measurable influence on travel time to the nearest polls within these longer travel time block groups ($p \geq .33$).

Number of Polls per Block Group

Characteristics of block groups with more than one polling location are presented in table 6. The average number of polls per block is .57 with standard deviation .76. While 2543 block groups had no polling location, 479 block groups had more than 1 polling location, with a maximum of 7 (n=1). Block groups with multiple polling locations have the potential to favor one race over another if they disproportionately reduce travel time for certain racial groups and not others. Furthermore, this affect may not be detected by methods that focus on travel time to the nearest polls, because many of the polls in these block groups are not the nearest polling location to any block group centroid. However, block groups with multiple polling locations do not appear to discriminate against Native Americans. Native Americans make up about .9% of the single race population of Wisconsin, but are 1% or more of the single race population in block groups with

A. Urban Densities

Model 0: R^2 =.20 , F = 445, p < .001				
	B	β	t value	P value
Land Area	1.95	.32	21	< .001***
Population Density	$-5.21 * 10^{-3}$	-.24	-16	< .001***
Model 1: R^2 = .22, F =142 , p < .001				
	B	β	t value	P value
Land Area	1.92	.31	20	< .001***
Population Density	$-4.16 * 10^{-3}$	-.20	-11	< .001***
Native American	-.016	$-5.8 * 10^{-3}$	6.7	.7
Asian Population	-.035	.024	-.38	.12
Black or African American	-.029	-.052	1.5	.002**
Hispanic or Latino Origin	-.021	-.028	-3.2	.08·
White Population	.021	.11	-1.8	< .001***

B. Rural Densities

Model 0: R^2 =.10 , F = 23, p < .001				
	B	β	t value	P value
Land Area	5.8	.25	4.9	<.001***
Population Density	$-3.5 * 10^{-3}$	-.11	-2.1	.036 *
Model 1: R^2 = .10, F = 7.5, p < .001				
	B	β	t value	P value
Land Area	3.97	.17	2.85	.005 **
Population Density	$-3.30 * 10^{-3}$	-.10	-1.77	.077·
Native American	.031	$8.56 * 10^{-3}$	.18	.85
Asian Population	-.033	-.025	-.48	.63
Black or African American	$-2.4 * 10^{-3}$	$-3.09 * 10^{-3}$	-.061	.95
Hispanic or Latino Origin	-.022	-.027	-.53	.60
White Population	.023	.15	2.46	.015 *

C. Remote Densities

Model 0: R^2 =.20, F = 64 , p < .001				
	B	β	t value	P value
Land Area	.45	2.0	8.861	<.001***
Population Density	$-4.7 * 10^{-3}$	-.041	-.093	.93
Model 1: R^2 = .23, F =22 , p < .001				
	B	β	t value	P value
Land Area	2.2	.50	9.3	<.001***
Population Density	.73	.084	1.5	.13
Native American	-.032	-.022	-.51	.61
Asian Population	1.65	-.063	1.53	.13
Black or African American	-.56	-.026	-.65	.52
Hispanic or Latino Origin	-.29	-.056	-1.4	.17
White Population	-.10	-.19	-4.1	<.001***

D. Poll Travel Time > 10 minutes

Model 0: R^2 = .025 , F =1.892 , p =.16				
	B	β	t value	P value
Land Area	.53	.23	.04	.06
Population Density	$2.7 * 10^{-3}$	.005	1.89	.97
Model 1: R2 = .0190, F = 1.197, p = .3176				
	B	β	t value	P value
Land Area	.59	.26	1.96	.054·
Population Density	.06	.12	.49	.627
Native American	-.10	-.13	-.88	.38
Asian Population	-.85	-14	-.57	.57
Black or African American	.59	-.12	-.86	.39
Hispanic or Latino Origin	-.97	-.16	-.98	.33
White Population	-.03	-.09	-.59	.56

Table 5: *Multiple regression results for travel time to the nearest polling location predicted by population density, land area and population estimates for each race.*

more than 1 polling location, with the exception of one block group containing 6 polling locations and no Native American residents. Additionally, multiple polling locations in a block group clearly serve large geographical areas, as the land area per poll is greater in all block groups with multiple polling locations compared to block groups with only 1 polling location.

Regression analysis results for the number of polls per block group predicted by population density, land area and population estimates for each race are presented in table 7. Land area and population density accounted for 31% (R^2 = .31) of the variation in the number of polls per block group and both predictors and the regression model were highly significant (p<.001). Adding population estimates for each race to the model yielded a 1% increase in predictive power (R^2 = .32). The Native American population estimate had a small, positive, highly significant influence on the number of polls per block group (B = .10, p<.001), while White and Black or African American population estimates had small, negative and highly significant influences (B = $1.5*10^{-4}$, p=.003 and B=$1.14*10^{-4}$, p<.001 respectively). Taken together, the results suggest that high poll number block groups exist in sparsely populated block groups with large land areas, and which tend to have stronger representations from White and Native American populations.

Number of Polls in Block Group	0	1	2	3	4	5	6	7
Block Groups	2543	1450	389	73	13	2	1	1
Land area / poll	Undefined	16	19	29	29	28	18	59
Residents / poll	Undefined	1354	683	447	357	281	182	199
Native American	.72%	.96%	1%	2.2%	4.5%	1%	0%	1.2%
Asian	3.1%	2.3%	1.6%	1.8%	.2%	.3%	0%	.4%
Black or African American	7.7%	5.5%	2.6%	.7%	2.3%	.1%	.6%	0%
Hispanic or Lartino Origin	7.9%	5.6	3.7%	2.6%	2.8%	.6%	.8%	1%
White	84%	88%	92%	93%	90%	97%	97%	97%

Table 6: *Characteristics of block groups with more than one polling location.*

Model 0: R^2 =.31, F = 1015 , p < .001				
	B	β	t value	P value
Land Area	.015	.47	31	<.001***
Population Density	$-1.2*10^{-5}$	-.06	-4.3	<.001***
Model 1: R^2 = .32, F =304 , p < .001				
	B	β	t value	P value
Land Area	.015	.54	41	<.001***
Population Density	$-9.9*10^{-6}$	-.07	-4.9	<.001***
Native American	$-6.0*10^{-4}$	-.05	-3.9	<.001***
Asian Population	$-5.5*10^{-5}$	-.006	-.4	.66
Black or African American	$1.5*10^{-4}$	.04	3.0	.003**
Hispanic or Latino Origin	$-2.9*10^{-5}$	-.006	-.4	.66
White Population	$1.1*10^{-4}$	.10	6.9	<.001***

Table 7: *Regression analysis of block groups with more than one polling location.*

V. Discussion and Conclusions

This study examined the distribution of polling locations in the State of Wisconsin with the intent of identifying inequities based on race. The primary endpoint, travel time to the nearest polling location, does not appear to be heavily influenced by race, especially after taking into consideration the residency preferences among racial groups for urban, rural and remote block groups. While Native Americans do have slightly longer travel times to the nearest polls, this trend correlates well with higher percentages of Native Americans in remote block groups. Similarly, travel times to the nearest polls tended to be shorter for Black and African American populations compared to all other racial groups, which also correlates with the higher percentages of this racial group in more densely populated block groups. The most parsimonious explanation for the small differences in travel time to polls is that these differences merely mirror the differential distribution across urban, rural and remote block groups of the racial groupings.

However, several limitations to our study bear a detailed explanation. Foremost, is issue of granularity. Representing an entire block group population by a single latitude and longitude in the form of a population centroid, is likely to grossly misrepresent the actual travel time to the nearest poll of racial groupings within a block group. Consider a hypothetical block group with a Native American reservation located at one end of a block group, and a town or urban center at the other end. If the nearest polling location is in a neighboring block group near the urban center, the shortest travel time to the nearest poll will be an overestimate for the urban center and an underestimate for the Native American reservation. This limitation would be would be minimized by obtaining population centroids based on smaller divisions within block groups, namely census blocks. Unfortunately, the US Census does not provide block-level centroids.

A second limitation to this study, also related to granularity, is that many block groups had multiple polling locations. The assumption made for the purpose of analysis, is that all residents of a block group utilize the polling location closest to the block group centroid, and that all travel from the block group centroid to the nearest polling location. These assumptions were made out of necessity and because more detailed (i.e. more granular) data on where residents live is not available. Clearly, the entire block group population is not located at the centroid, but instead distributed around the land area within the block group, such that all polling locations within the block group as well polling locations in neighboring block groups are likely to be utilized, regardless of whether they appear as a "nearest polling location" in this study.

A final limitation is that this study is unable to identify differences within block groups, with respect to where people from different racial groupings live. With block groups up to 410 square miles and containing over 8000 residents, many block groups are likely to contain pockets where racial groups are concentrated, such as Native Americans on reservations. These within block group pockets may have very different travel times to the nearest polls than what is presented in this study. Furthermore, this study is not equipped to address the possibility that within a block group, a larger population of Native Americans may have to travel to a polling location in a smaller less populated town to case votes in person. Again, block-level data and population centroids would help address this concern.

In spite of the limitations, this study suggests that egregious state-wide inequities in travel time to the nearest polls are not at play in the state of Wisconsin. Certainly the types of extreme inequities observed in states such as Alaska, Nevada, Utah, Arizona and New Mexico where Native American tribes must drive up to 163 miles to cast a vote[Native News Online.Net] do not exist in Wisconsin. This is certainly not to say that inequities don't exist though. To identify inequities that manifest within block groups, or that are otherwise unobservable by this study,

the recommendation is to focus on specific regions of interest and pursue a highly localized and detailed examination of nearby polling locations and population demographics.

References

[ACS, 2015] American Community Survey 2015. *https://www.census.gov/data/developers/data-sets/acs-5year.html*

[TIGER/Line Shapefiles, 2017] Wisconsin TIGER/Line® Shapefiles, 2017, US Census Bureau, *https://www.census.gov/cgi-bin/geo/shapefiles/index.php*

[R Core Team, 2015] R Core Team (2015). R: A language and environment for statistical computing. *Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.*

[tidycensus] Kyle Walker, *Return tidy data frames from the US Census Bureau API*, R Package, version 0.9.2

[WEC, 2018] Wisconsin Elections Commission, 2018 General Election Polling Places.xlsx, *https://elections.wi.gov/elections-voting/2018/fall*

[Centers, 2010] US Census Bureau, Centers of Population, 2010 *https://www.census.gov/geographies/reference-files/time-series/geo/centers-population.html*

[Google Maps] Google Maps Platform, 2019 *https://developers.google.com/maps/documentation*

[Shapely] Sean Gillies, *Manipulation and analysis of geometric objects in the Cartesian plane* Python Package, version 1.6.4

[Geocoder] US Census Bureau *https://geocoding.geo.census.govl*

[Native News Online.Net] *Tribal Equal Access to Voting Act of 2015* Native News Online Staff, 21 May 2015 *http://nativenewsonline.net/currents/tribal-equal-access-to-voting-act-of-2015/*

Understanding Insincere Questions on Online Q/A platforms

Parth Anand Jawale
University of Colorado Boulder
parth.jawale@colorado.edu

Abstract

As online communities grow in size, it is only self-evident that more number of users are subjected to content that clearly violates the policies of the forum. All of these communities operate on mutual trust, faith and support of its users. This project uses a data-oriented approach to understanding what insincerity means for an online Q/A platform, Quora. The approach consists of analyzing data to understand the interplay between question construction and insincerity. Models were developed to predict sincere as well as insincere questions and present a comparative analysis of what works best for the given problem. It is fashioned as text classification problem, as opposed to a clustering problem wherein the attempt would be to cluster sincere and insincere questions. In this project, the models of text classification are developed using Natural Language Processing (NLP), Machine Learning and Deep Learning approaches. The models are compared using F1 scores, Precision and Recall. The impact of word-embeddings on modeling approaches has been measured and suggestions have been made for ways in which document-coverage can be used as a measure of evaluating the use of embeddings on a given downstream NLP task.

I. Introduction

An existential problem for any major website today is how to handle toxic and divisive content. This is a highly relevant problem in online social interactions, as the internet, and especially social media platforms where people attempt to resolve their curiosities, are subjected to content that clearly violates the policies of the forum. To build a human-in-the-loop system which assists in flagging of this violative content is a hard problem, because the criteria is often non-comprehensive and because of the ubiquitous presence of meaning-overloaded words which could be seen as having negative connotation on first-sight but might not be insincere e.g., words such as *bad, worthless* etc. Prior work on this problem, especially using neural approaches focuses on use of Convolutional Neural Networks [1] to detect toxicity in comments. We focus on the use of statistical methods, embeddings and langauge models on similar downstream tasks.

We seek to understand what causes questions to be insincere and build models to predict potentially insincere questions. As a matter of procedure, moderators exist on online Q/A forums to manually tag insincere questions, however with gradual increase in the number of users, these platforms find that the number of questions to check far outweigh the number and capacity of forum moderators. With this project, we hope to flag disingenious, malicious, discriminatory content or as Quora defines them "insincere questions" and deter potential users from posting such content.

II. Data

In this research, we use a data-set of questions and noisy labels provided by Quora. Insincere questions, according to Quora are defined as ones that meet any of the following (non-comprehensive) criteria:

- has a non-neutral tone, such as an exaggeration to emphasize a comment about a particular group of people;
- is disparaging or inflammatory, such as an attempt to seek confirmation of a

stereotype or present a discriminatory remark, a comment based on an outlandish premise about a group of people, or the disparaging of a natural, not fixable, or immeasurable characteristic;

- Is not grounded in reality, such as a reference to false information or absurd assumptions;
- Uses sexual content for shock value, such as references to pedophilia, incest, or bestiality outside of an attempt to find genuine answers.

The data-set was provided by Kaggle and contained a training set of over 1,300,000 labeled examples [2]. Each example in the training set has a unique ID, the text of the question, an a label of '0' or '1' to represent 'sincere' or 'insincere'. Although the data is very large, there is a large discrepancy in the class distribution within the training set. This is one of the biggest challenges of this project, since we are essentially trying to predict if a question is 'insincere', and there are very few actual 'insincere' questions to train on. This is also the reason why we use F1 score as the metric to evaluate our models on.

III. Background

We start modeling simpler, computationally inexpensive models and gradually increase model complexity and conduct ablation tests with respect to features and text processing.

TFIDF [3]: TF-IDF score represents the relative importance of a term in the document and the entire corpus. TF-IDF score has two components: the first computes the normalized Term Frequency (TF), the second term is the Inverse Document Frequency (IDF), computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

TF(t) = (Number of times term 't' appears in a document) / (Total number of terms in the document)

IDF(t) = log_e (Total number of documents) / (Number of documents with term 't' in it)

Logistic Regression [4]: Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (insincere question) or 0 (sincere question). In other words, the logistic regression model predicts P(Y=1) as a function of X, whilst preserving the marginal probabilities of the data.

NB-SVM (Naïve Bayes - Support Vector Machines) [5]: This is an SVM variant using NB log-count ratios as feature values. Naïve Bayes (NB) and Support Vector Machine (SVM) are widely used as baselines in text-related tasks but their performance varies significantly across variants, features and data-sets. Wang et al. suggest that word bigrams are useful for sentiment analysis, but not so much for topical text classification tasks but NB does better than SVM for short snippet sentiment tasks, while SVM outperforms NB for longer documents.

Support Vector Machines (SVM) [6]: A Support Vector Machine (SVM) is a discriminating classifier formally defined by a separating hyper-plane. Given labeled training data (supervised learning), the algorithm outputs an optimal hyper-plane which categorizes new examples (in this case, categorizes questions asked as sincere or insincere). In two dimensional space, this hyper-plane is a line dividing a plane in two parts wherein data points from each class lie on either side.

Bidirectional LSTMs [7]: LSTM preserves information from inputs that has already passed through it using the hidden state. RNNs suffered from a popular 'vanishing gradient' problem that caused those nets not to learn so much because of disproportionately small weight updates. Using Bidirectional LSTMs, we feed the learning algorithm with the original data once from beginning to the end and once from end to beginning. In each pass (forward and backward), the reading of

input words is modeled as a recurrent process with a single hidden state.

Visualization using T-SNE [8]: t-distributed Stochastic Neighbor Embedding (T-SNE) is a machine learning algorithm primarily used for visualization. It is non-linear technique, used for dimensionality reduction and is used to visualize high-dimension data in two or three dimensions. It models each high-dimensional object by a two or three-dimensional point. This is done in a way that similar objects are modeled by points closer to each other and dissimilar objects are modeled by distant points with high probability. This is reminiscent of the idea behind word-embeddings, and higher dimensional embeddings can be visualized using t-sne.

Embeddings [9]: An embedding is a form of representing words, characters, n-grams and documents using dense vector representations. The position of a word/n-gram/character within the vector space is learned from text and is based on the words/n-grams/characters that surround the word when it is used. Word embeddings can be trained using any input corpus or can be dynamically generated using pre-trained word embeddings such as GloVE, FastText, and Word2Vec. Most of them are used as transfer learning layers in neural architectures. In this work, we have used GloVE and FastText as an embedding layer to neural models.

IV. Methods

We evaluate the results using all the algorithms listed above in the Background section. We also use word-embeddings as a feature in our neural BiLSTM model.

Using embeddings as a feature:

- Load the pretrained word-vectors (GloVe, FastText, Paragram etc.)
- Create a tokenizer object
- Transform documents to sequence of tokens (along with padding)
- Create a mapping of tokens (and subsequently obtain the embedding matrix)

We apply text processing techniques to stem, lemmatize, remove stop-words etc. to make the classifier learn over texts that are of uniform format. Another goal of this project is to implement lexical processing techniques to compare and improve quality of word-embeddings for a given corpus based on text coverage. We primarily use two word-embeddings - Global Vectors for Word Representation (GloVe) [11] and FastText [12] to test performances of the embeddings themselves and the models that use them. Once the baselines are established, we create more complex, computationally expensive models to improve upon them. We use Bidirectional LSTMs (BiLSTM), architecturally similar to TextCNNs [13]. Bi-LSTMs usually provide slightly better results than using a single LSTM for most NLP tasks, because a word's context in a sentence includes future words as well as previous words. To test this claim, we will try to compare the results of a bidirectional model to a unidirectional model. We will also use FastText's shallow neural text classification model along with its embeddings to compare performance.

Finally, we conduct an analytical and comparative study of questions being asked online and what topics can be tagged to insincere questions. For this study, we use data analysis and Latent Dirichlet Allocation (LDA). The "topics" produced by topic models are clusters of similar words. LDA captures the mathematical intuition behind this, and can be used to explain what topics might the document have, and what the balance of topics and words is, within the document. This technique is used in this project to assess primary topics that seem to be the cause of insincerity, as a conjugate to analyzing word bigrams or trigrams. LDA is harder to interpret, but some implementations, as the one used in this project (called PyLDAvis) provide ways to interpret, understand and fine-tune LDA models based word saliency.

We also conduct an analysis on different word-embeddings. As opposed to using pre-

trained vectors as they are, we present an analysis which could help increase the document-coverage on different kinds of word vectors. The reason this helps is because some embeddings are trained only from uncased vocabularies. To add to that, there is often anomalous behavior relating to numbers/digits. The ability of an embedding to capture the context of a rare or out-of-vocabulary (OOV) word depends on how it was trained. But by pre-processing these embeddings, we can increase the number of OOV words it can cover, which we claim would increase the performance of the model.

V. Results

Data Analysis: The data is extremely imbalanced (only 6.2% are insincere questions and 93.8% are sincere questions). This is the primary cause of complexity in this problem. [10] With further analysis of (uni/-bi/-tri)-grams, we were able to understand the search terms and the latent topics they belong to, which causes them to be insincere. This is an attempt to explain why the models would predict a question to be sincere or otherwise.

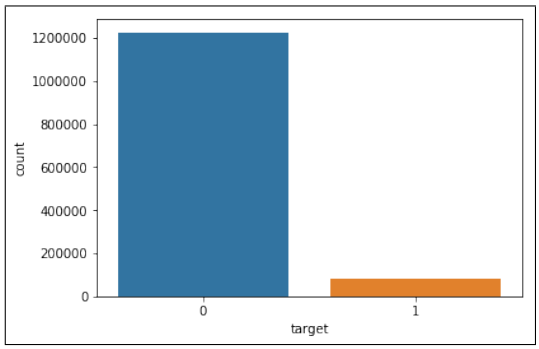


Figure 1: *Distribution of insincere vs sincere questions where 'target' refers to the two classes and 'counts' refer to number of examples in each class*

We also visualize a word-cloud of the insincere questions to understand common words, phrases etc. which helps us to gauge what topics might be causing questions to be insincere. We can see that the lot of common words are about race, countries, political positionings etc. and tend to be more controversial. The word-cloud for sincere questions (not shown here) has words relating to topics on jobs, career, life etc. implying that there is a pattern between questions asked and the topics they belong to.

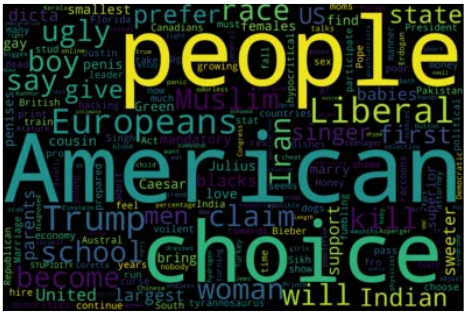


Figure 2: *Wordcloud visualizing insincere questions*

A technique that visualizes higher dimensional data into two dimensions is T-SNE. We have used T-SNE as a way to find out if the classes in the given data are trivially separable or not. After randomly sampling and count vectorizing 25000 data points, we apply TruncatedSVD before injesting the data into a T-SNE model which takes our target values in a high-dimensional data and maps it to a lower dimension. The results from the model indicate that the data is not trivially separable. This begs the question of how easy therefore, is it in terms of semantic meaning (to a human) to distinguish between an insincere and a sincere question. It also helps to realize that cluster sizes as well as distances from one cluster to another are not meaningful to the overall global geometry of the picture, since T-SNE preserves patterns in data, not density or distance properties. Some parts of the graph have heavier clustering of one target class, and others do not, but we cannot reflect upon these as separable patterns because they are not explainable. But we can say that the patterns are at least not such that the classes seem trivially discernable.

Figure 3 illustrates a T-SNE visualization that shows non-separability of the data.

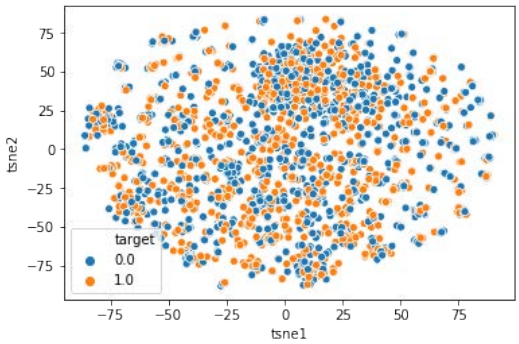


Figure 3: *TSNE Visualization - 'Target' refers to class variable 0 (sincere) or 1 (insincere)*

Modeling: After some preprocessing, we established baselines with TFIDF and simple Logistic Regression. F1 score was the metric of choice, due to the extreme class imbalance. The resultant pipeline gives an F1 score of 58%. Upon further experimentation with word-embeddings, we tried fitting a Bi-LSTM model which gives an F1 score of 60.2%. This is expected to be more, however computational constraints (we ran it for 4 epochs) have caused us to stop training much before convergence was reached. This problem can be addressed using computational resources like cloud platforms and GPU clusters. We also implemented TF-IDF + Linear SVM model and it performs surprisingly well with an F1 score of 61%. Our best performing model seems to be NB-SVM, with an F1 score of 63%. We expect other neural models, like Bi-LSTM and TextCNN (used along with word-embeddings) to perform better, given access to ample computational resources.

Another notable observation is that the Bi-LSTM has the highest precision in identifying the insincere questions (71%) which implies that it is the best at modeling and understanding language, phraseology and sentence structure, something other models lack at. Even the best performing model has precision of 58% at identifying insincere questions in spite of higher F1 values. We argue that in problems where false positives/negatives can be rectified by human-in-the-loop mechanisms such as this problem, Precision@1 should be used as a metric of choice instead of F1-Score. This is because the model was built to identify insincere questions, and precision on that very downstream task of should be the popular metric of choice. We report the F1 scores in this paper, but present an argument for precision as the metric of choice for class imbalanced problems where classifying one class (in this case, insincere) is the task at hand.

Model	F1-Score
NB-SVM	0.63
TF-IDF + Linear SVM	0.61
BiLSTM.	0.602
TF-IDF + Log. Reg	0.58

Word-Embeddings - Another goal of this study was to understand how word-embeddings and their impact can be evaluated on a given corpus. The objective of this is to get our corpus vocabulary as close as possible to the word-embeddings. The way to do this is to find the intersection between the embeddings and our vocabulary, and the higher this intersection - the higher will be the ability of the embeddings to cover our entire corpus (document-coverage). With the GloVE embeddings, the original document-coverage was 88.21%. With Paragram embeddings, this number was significantly lower (72.19%) because Paragram embeddings do not understand upper letters. Similarly, FastText embeddings have a document-coverage of around 87%.

The higher the document coverage, the better the model building. This is because the model is not losing out on information which may be in the embeddings and involves dealing with upper-casing, numbers/digits, out-of-vocabulary (OOV) words, misspellings, contractions, punctuations etc. To increase this document coverage, we pre-process the text heavily. After all of the operations, we were able to achieve a document-coverage of almost 99% for all of the word-embeddings. S. Rezaeinia [14] suggests that neural models built from these embeddings would be superior, however that claim is not tested as a part of this research so far. In this work, these exercises are done to hypothesize and understand which

word-embeddings might perform better for a given corpus.

Embedding	Coverage_i	Coverage_f
GloVe	71.38%	99.13%
Paragram	72.21%	99.62%
FastText	87.66%	99.44%

VI. Discussion and Conclusions

We analyzed the Quora dataset for insincere questions and provided insight into why these questions might be insincere. Some of the observations can be summarized as follows: Sentence lengths matter. The average lengths of sincere and insincere questions are significantly different. Other exploratory data analysis centers primarily around word (uni-/bi-/tri-)grams and word-clouds. We use these visual tools to understand what words or sequences of words are causing questions to be insincere. This also gives an insight into the topics that cause people to phrase questions that may be defamatory, inflammatory or intended for shock value. Examples of some of these can be found repeatedly in the dataset for e.g, examples on politics, elections etc.

We also tried several models to predict whether we can learn to identify potentially insincere questions. We used several supervised learning models along with other models that utilize word-embeddings, neural networks and transfer learning. We also tried using two ensemble models. Results from most of the modeling exercises show that ensemble models work well. Our results show that a combination of using log features from Naïve Bayes as an input to an SVM performs the best (NB-SVM), followed by neural models like BiLSTM. Other models also do fairly well. If we just consider the precision as a metric of comparison, which would be the ideal choice if we were to just classify insinceres, we can see that neural approaches far outweigh other models. With a BiLSTM, we were able to achieve a Precision of 71% on identifying insincere questions. Understanding this interplay between precision and F1 score proves important in judging which model is the best performing one.

We attempted to improve word-embeddings by improving their document coverage. Word-embeddings have the ability to capture words of certain kind as they appear within different contexts depending on word frequencies, word neighbors and other more linguistically fine-grained approaches. However, word-embeddings operate on a word-level and not on character-level (unless explicitly designed), so it becomes important that the vocabulary in the corpus be similar attuned to way they were trained. Some embeddings do not cover contractions, misspellings, rare words etc. and learning from them becomes possible if we pre-process the vocabulary such that embeddings can cover these words, and still has a possibility of learning. To this end, we calculate initial document-coverage and pre-process our corpus to try and increase this coverage up to 100%, achieving results up to a highest coverage of 99.62% (for Paragram embeddings). The impact of increased coverage on neural models is yet to be analyzed.

VII. Future Work

We could resources on cloud like GCP, Colab, AWS to be able to train computationally expensive models like BiLSTMs and TextCNNs. Recent advances in NLP have seen a meteoric rise in the popularity of combining transfer learning techniques with large-scale Transformer language models. [15] Using Transformer models like BERT, ELMo, XL-Net or pre-trained language models like ULM-FIT to learn from insincere questions would better help understand the linguistic phenomena that makes a question on Quora insincere. This requires training on an extremely large data-set. That can also be achieved by scraping Quora for other insincere questions. Another possibility is to use the improved word-embeddings with increased coverage and use them as an embedding layer of a neural architecture. MOE (Misspellings Oblivious Embeddings) [16] can

be used to tackle possible misspellings. Character level embeddings can help capture inflectional forms. Architectures involving sub-word models (like Wordpiece, Byte-Pair Encoding, subword-NMT) can be used to capture basically any possible word and create a more robust embedding layer. [17]

References

[2] Quora Insincere Questions Classification *Kaggle. 2018 https://www.kaggle.com/c/quora-insincere-questions-classification/*

[1] Convolutional Neural Networks for Toxic Comment Classification *Georgakopoulos et al. 2018. https://arxiv.org/pdf/1802.09957.pdf*

[3] Understanding Inverse Document Frequency: On theoretical arguments for IDF *Robertson. 2004. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.438.2284&rep=rep1&type=pdf*

[4] An Introduction to Logistic Regression Analysis and Reporting *Peng et al. 2002. https://datajobs.com/data-science-repo/Logistic-Regression-[Peng-et-al].pdf*

[5] Baselines and Bigrams: Simple, Good Sentiment and Topic Classification *Wang et al. 2012. https://nlp.stanford.edu/pubs/sidaw12_simple_sentiment.pdf*

[6] A Practical Guide to Support Vector Classification *Hsu et al. 2016. https://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf*

[7] Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network *Sherstinsky. 2018. https://arxiv.org/abs/1808.03314*

[8] Visualizing Data using t-SNE, *L. Maaten et al. 2008 http://www.jmlr.org/papers/v9/vandermaaten08a.html*

[9] Estimation of Word Representations in Vector Space, *Mikolov et al. 2013 https://arxiv.org/abs/1301.3781*

[10] The Class Imbalance Problem: Significance and Strategies, *N. Japkowicz. 2002 https://content.iospress.com/articles/intelligent-data-analysis/ida00103*

[11] GloVe: Global Vectors for Word Representation, *Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014 https://nlp.stanford.edu/pubs/glove.pdf.*

[12] Advances in Pre-Training Distributed Word Representations, *T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, A. Joulin. 2017 https://arxiv.org/abs/1712.09405)*

[13] What does TextCNN learn? *Linyuan Gong, Ruyi Ji. 2018 https://arxiv.org/abs/1801.06287*

[14] Improving the Accuracy of Pre-trained Word Embeddings for Sentiment Analysis, *S. M. Rezaeinia. 2017 https://arxiv.org/abs/1711.08609*

[15] Transformer: A Novel Neural Network Architecture for Language Understanding, *J. Uszkoreit. 2017 https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html*

[16] Mispelling Oblivious Word Embeddings, *B. Edizel et al. 2018 https://arxiv.org/abs/1905.09755*

[17] Neural Machine Translation of Rare Words with Subword Units *R. Sennrich et al. 2015. https://arxiv.org/abs/1508.07909*

Boulder County Internet

Holden Kjerland-Nicoletti
University of Colorado Boulder
hokj6625@colorado.edu

Abstract

In Boulder County, Colorado, internet availability and it's relation with certain demographics is an interesting and important question. In this paper the Federal Communication Commission's census internet data is analyzed with corresponding census demographic data for Boulder County, especially demographics pertaining to mobile home communities. The methods for this analysis include regression, t-tests, transformations and Wilcoxon-Mann-Whitney tests. The results show that there is not a strong correlation between mobile home communities and internet speed, nor for household income. However, there is a positive correlation for population.

I. Introduction

In today's world, internet availability is a necessary resource to be able to do many things. For example, for a high schooler applying for college, this is almost impossible to do without internet nowadays. This paper takes a look to see if certain demographics have a correlation with internet availability in Boulder county.

There are five main demographics being compared with internet speeds. At the census block group level the demographics analyzed are whether or not the block group contains a mobile home community, average household yearly income, population, and distance of center of population to center of Boulder county population. For the census tract level there is an analysis on the percentage of the population living in mobile home communities.

First, is a comparison between census block groups with a mobile home community and without a mobile home community compared with their internet download speeds. This will see if mobile communities recieve the same internet availability as non-mobile home communities.

Next, is a comparison at the census tract level of percentage of population living in a mobile home community compared with internet download speed. This comparison will explore if areas with more mobile home living residents recieve different internet availability.

Then, is a comparison at the census block group level of average yearly houeshold income and internet download speed. This will see if communities of higher wealth receive different internet availability.

An exploration into the relationship between population of census block group and internet. This will see if areas with higher population recieve different internet availability.

Finally, is an exploration on the center of the population of each block group and it's Euclidian distance to the center of population for all of Boulder county. Center of population is the "balancing point" of an area if all of the people are weighted equally. This looks to see if areas farther from the "center" of Boulder recieve different interney availability.

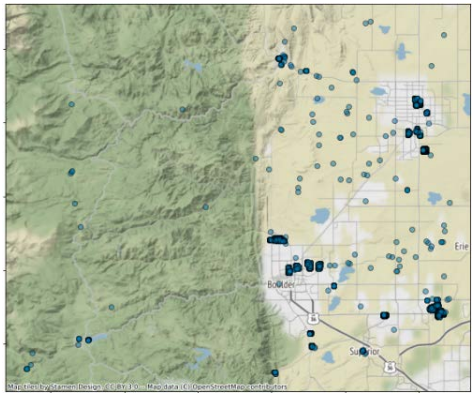


Figure 1: *Map of Boulder county, and all of the mobile home addresses in Boulder County represented as points.*

II. Data

Four different sources were used to obtain the data for this paper. The first, was the Federal Communications Commission's Fixed Broadband Deployment Data [1]. All facilities-based broadband providers are required to file data with the FCC twice a year (Form 477) on where they offer Internet access service at speeds exceeding 200 kbps in at least one direction. This data gives these internet speeds down to the census block level. The data was filtered to only have the census blocks in Boulder County, CO. The maximum advertised downstream speed/bandwidth offered by the provider in the block for Consumer service was used as the main measurement for "internet speed".

The second data used is the US Census data for Boulder County [2]. The main statistics from the census data that are used are household income and population. Next, the "Mobile Home Parks" dataset [3] from data.gov was used, which has all of the mobile home parks in the United States. This data was filtered to Boulder County, and then these mobile home parks were matched to their census block based on address. At the census tract level, the mobile home percent was solved for, which is the percentage of the population living in mobile home parks for that census tract. Finally, the 2010 US Census dataset for Centers of Population was used [4]. The center of population is the "balancing point" if the area was converted to a flat plate, and all of the people were equal weights on the plate.

In the end, the aggregated data used for this paper is: average household income, mobile home park (True or False), mobile home percentage, and max advertised downstream speed. This data was aggregated into both census tracts and census block groups using means. A visual of the table can be seen in Figure 2.

There were also no obvious outliers that needed to be considered.

III. Methods

The first method used was a two-sample t-test on mobile home park internet speeds compared with non-mobile home park internet speeds using the census block group level. This test compares the means of both samples along with each samples variance to see how likely one sample could have come from the other. However, t-tests rely on the data being normally distributed, which this data does not appear to be, so this test could not be used.

The size of the non-mobile home block groups was significantly larger than mobile home block groups, so to adjust for this downsampling was performed on the non-mobile home block groups. This means rows from the non-mobile home block groups were randomly selected to match the same size as the mobile-home data. This creates a more fair comparison between the two distributions.

For an additional test on this data, a Wilcoxon-Mann-Whitney test was performed. Unlike the t-test, this test does not rely on the data being normally distributed (which the data does not appear to be), and this test looks at the likelihood that the two sets of data are from the same distribution. This test relies on some assumptions about the data, which this data passes those assumptions.

The next method used linear regression on mobile home percentage for census tracts compared with internet speed to see if there was correlation. Linear regression finds the least

Data Table (Tract Level)						
Census Tract Code	Max Advertised Down Stream Speed/ Bandwidth	Average Household Income (USD)	Contains Mobile Home	Percentage of Population in Mobile Homes	Population	Distance of Center of Population to Boulder County Population Center

Figure 2: *Aggregated data table used for the analysis on census tracts. Aggregated data table used for census block groups is same but rows correspond to census tract level.*

squared errors coefficient. In two dimensions like this, this means that it tries to find the line that minimizes the squared errors of each point from the line.

It was found that the model was being skewed because the majority of points lied on the 0% x-coordinate (meaning most census tracts had a 0% mobile home population). To account for this the linear regression model was recreated without the the 0% census tracts, meaning only census tracts with at least some mobile home population were analyzed. There also appeared to possibly be some other hidden curve in the scatter plot, so two transformations were performed on the data, first was a log transformation which took the natural log of the population percentage to fit the following equation

$$y = a * ln(x) + b$$

and also a 1/x transformation to fit the following equation

$$y = \frac{a}{x} + b$$

Where a, b, c were the constants that were fitted for the data. Also, because all census block groups with mobile home percentage of 0 removed, there is no issue with dividing by zero.

Next, the relationship between average household income and internet speeds was analyzed. Again, linear regression is trying to minimize the squared errors between each point and the regression line. Looking at the scatter plot, there does not appear to be a hidden shape to these points that is not linear, so linear regression is probably the best choice for the analysis of these two variables.

For population, regression is again performed, and the data appears to have a slight linear fit, so no transformations were performed.

Finally, for distance of center of populations we use regression. The data appears to have a logarithmic fit, so a log transform on the distances is used for the regression.

IV. Results

For the first method, the 2-sample t-test on mobile home parks compared with non mobile home parks and their internet speeds, it was found that mobile home parks did in fact have lower average internet speeds. Mobile home parks had internet speeds 3% lower than non-mobile home parks. However, this difference was too small to say that it was significant, having a p-value over 0.9 (two-sided p-test), and for this paper a standard alpha level of 0.05 is used.

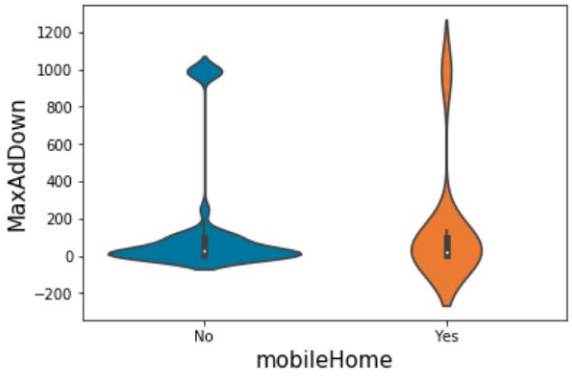


Figure 3: *Violin plot of the all census block groups with x-axis representing containing mobile home park or not, and their corresponding internet download speed.*

For the Wilcoxon-Mann-Whitney test with a smaller random subset of non-mobile home census block groups, it was found that the significance level of the test was 0.032, this means we can reject that the two different datasets have the same distribution.

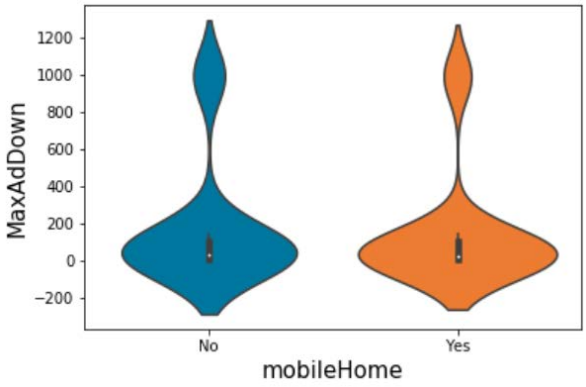


Figure 4: *Violin plot of random subset of non-mobile home census block groups and mobile home block groups, with their corresponding internet download speed.*

Next was the linear regression on mobile home population percent compared with internet download speed. The coefficient between mobile home percent and internet speeds was -31.52, meaning that an increase in mobile home population by 1 percent meant a decrease in internet speed by 0.3152kbps. However, again, this coefficient was found to have a very high p-value of 0.756, so it could not be considered significant.

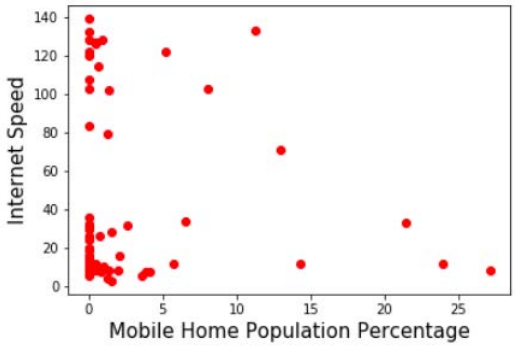


Figure 5: *Plot of all mobile home park percentages compared with their internet download speed. Also, has line of best fit from linear regression.*

For the repeated linear regression model of only census tracts with at least some mobile home population, there was a larger negative correlation found to be -65.84. With a lower p-value of 0.588, but this is too high to be considered significant at the 0.05 alpha level. The R-squared value of this model was 0.010.

Using the log and 1/x transformations discussed in the methods section, it was found that the log transform fit the data better with an R-squared of 0.082 and 0.036 respectively, however neither of these fit the data as well as the simple linear regression.

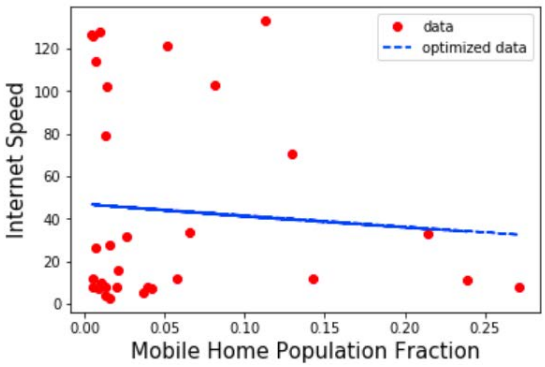


Figure 6: *Plot of all mobile home park percentages greater than 0% compared with their internet download speed. Also, has line of best fit from log transform regression.*

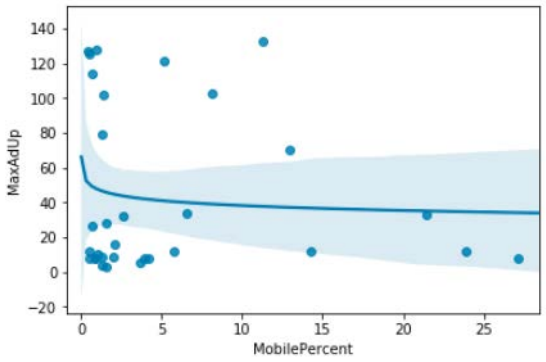


Figure 7: *Plot of all mobile home park percentages greater than 0% compared with their internet download speed. Also, has line of best fit from log transform regression.*

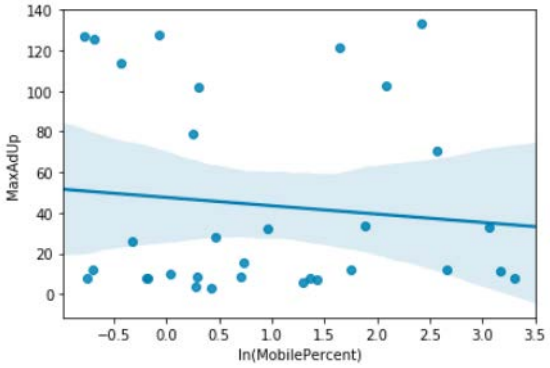


Figure 8: *Plot log transform of population percentage, along with line of best fit from simple linear regression.*

There was the linear regression between average income at the census block group level and internet speed. The results from this regression found that the correlation between income and internet speed was actually slightly negative with a value of -0.0004 meaning an increase in $1000 for household income, means a decrease of 0.4kbps. This coefficient had a p-value of 0.05 (two-sided). This means that block groups with higher average incomes have sligthly lower internet speeds. The R-squared value for the line of best fit was 0.039.

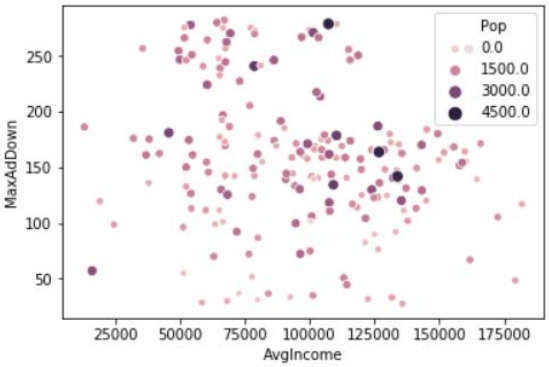


Figure 9: *Plot of average yearly household income compared with internet download speed at the census block group level. The size and color of points is based of the points' population according to the plot legend.*

For the regression on population compared with internet speeds, there is a positive correlation of 0.0181, meaning adding one person to the block groups leads to an increase of 0.0181kbps on average. The coefficient also has a p-value of 0.003, so we can conclude significance at the 0.05 alpha level. No clear hidden shape appears to be in the data, so only a linear regression was performed.

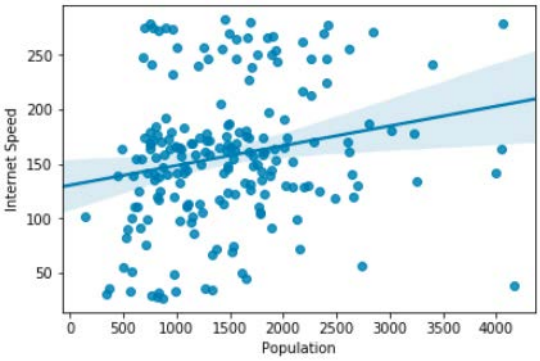


Figure 10: *Plot of population of census block group and internet speeds, including line of best fit.*

Finally, for the log transformation on the distance of centers of populations for each block group to the center of Boulder county's population, the linear regression has a R-squared of 0.131, gives a correlation of -2.63, and a p-value less than 0.001, so we can con-

clude significance again at the 0.05 alpha level.

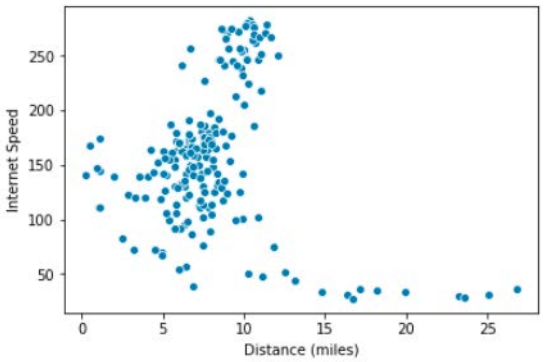


Figure 11: *Plot of distance of centers of populations for each block group to the center of Boulder county's population and internet speeds.*

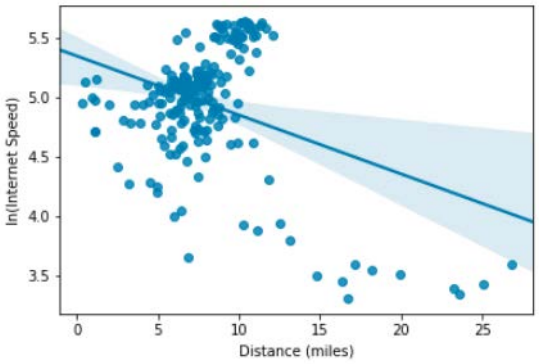


Figure 12: *Plot of log transformation of distance of centers of populations for each block group to the center of Boulder county's population and internet speeds, including line of best fit from simple linear regression.*

V. Discussion and Conclusions

When comparing all census block groups that have mobile home communities and don't have mobile home communities, the mobile home census block groups had lower internet download speeds on average. However, the average for mobile home block groups was only 3% lower than non-mobile home communities block groups. It is good to know that at an initial glance mobile home communities don't appear to be lacking very far behind in internet availability.

However, when downsampling and using the Wilcoxon-Mann-Whitney it is found to be unlikely both types of block groups have the same distribution. This tells us that if we dig deeper, there actually is a bigger difference in internet availability of mobile home census block groups and non-mobile home census block groups. At the very least, the two different block groups have a different distribution of internet download speeds. However, using this test, we can't claim anything more than that.

Looking at the comparison between percentage of population living in a mobile home community with internet download speeds, we see there is only a very slight correlation between the population percentage and internet download speed. However, it makes sense to remove the census tracts with 0% mobile home population because these census tracts are irrelevant to this analysis.

When looking at census tracts with some mobile home population percentage, we see much more negative correlations. Although, the linear regression again did not have a significantly significant coefficient. This means that it is certainly possible that this negative correlation could have occured by chance.

From looking at the scatter plot, two curves that appear could better fit the data were the inverse exponential, and the inverese function. However, neither of these transformations gave a better R-squared than the linear regression, so not much can be used from these transformations.

It seems that the main takeaway from this analysis is that there is a decrease in internet download speeds for communities that have a higher percentage of their population living in mobile home communities, but this negative correlation is small enough that it is not significant at the 0.05 alpha level. Also, despite what would seem to be the case at first glance, there does not appear to be a hidden shape in the data in the form of some sort of inverse or inverse exponential.

Looking at the census block group level again and the average yearly household income compared with it's internet download speed there are some interesting results. The correlation between average yearly household income and internet download speed was in fact negative.

The linear regression model between average yearly household income and internet download speeds has an R-squared value of 0.039, however the p-value is exactly 0.05 which is the alpha level of this paper. Looking at figure 9, there does not appear to be a strong correlation in the data, in fact, the data looks pretty random and does not appear to have any shape at all.

There are some potential explanations for this correlation. One is that communities of higher wealth might live in more rural areas which are less likely recieve as good of internet availability as more urban areas. Wealthier communities might also live deeper in the mountains. These communities have a higher distance to the population center of Boulder, which means that they likely have less internet availability from looking at figures 11 and 12.

For population the positive correlation between internet speeds and population has a likely business explanation. Areas with more people have more customers, and therefore internet companies are more likely to invest more resources in that area.

To conclude, there does not appear any strong statistics/evidence that mobile home communities recieve less internet availability than non-mobile home communities. However, the two different community types do likely have different distributions of internet availability. There is also a slight negative correlation between average yearly household income and internet availability, a positive correlation between population and internet speeds, and a negative correlation between distance from the population center of Boulder county and internet speeds.

References

[1] . [FCC Form 477, 2018]FCC2018 Fixed Broadband Deployment Data from FCC Form 477 *U.S. Federal Communications Commission https://www.fcc.gov/general/broadband-deployment-data-fcc-form-477*

[2] . [U.S. Census Bureau, 2015]USCB2017 HOUSEHOLD INCOME IN THE PAST 12 MONTHS (IN 2017 INFLATION-ADJUSTED DOLLARS) *2017 U.S. Census Bureau https://factfinder.census.gov*

[3] . [U.S. Census Bureau, 2015]MHP2019 Mobile Home Parks *2019 U.S. https://catalog.data.gov/dataset/mobile-home-parks*

[4] . [U.S. Census Bureau, 2010]MHP2010 Centers of Population for the 2010 Census *2010 U.S. https://www.census.gov/geographies/reference-files/2010/geo/2010-centers-population.html*

Activity Re-Identification Using Time Series Classification Techniques

Israel J Miles
University of Colorado Boulder
israel.miles@colorado.edu

Abstract

Privacy is becoming a growing concern at the forefront of technological development. Both data collection and the algorithms to describe it are becoming increasingly precise. This can lead to serious privacy and security concerns such as user re-identification. In this report, multiple methods of time series analysis will be explored to demonstrate the ability to expose an individual's daily activities from cell phone accelerometer data. This work joins a rapidly growing literature discussing privacy considerations for mobile devices.

I. Introduction

Time-series classification is an increasingly applied field in several domains, such as health informatics, weather patterns and production line throughput. As the number of applications to time series increase, so does the data that serves as input to the classification models. Among the most common are k-nearest neighbors and specialized neural networks. In this report, feature extraction will take place of complicated models in order to offer high interpretability with methods such as logistic regression, random forests and linear support vector classifiers. Furthermore, classification performance will represent the privacy risks of exposed smartphone data. Accelerometer and gyroscope data will serve as the input for our models in order to showcase the ability to re-identify user activities given nothing more than simple time stamps of acceleration and rotation magnitudes. This study shows that basic feature extraction and trivial models are capable of identifying user behavior, and the current risks of exposing user senstitive data.

II. Data

The MotionSense Dataset [1] is composed of 24 participants performing multiple actions such as sitting, walking, jogging, climbing up or down stairs and standing. The experiments involved giving each participant an iPhone 6 that would be stored in a front pocket. The phone would then record common gyroscopic and acceleromoter sensors that are found in most phones today. There are 15 total trials split up into two types of trials, long and short for each subject.

- Long Trials: Numbered 1 through 9 lasting 2-3 minutes.
- Short Trials: Numbered 11 through 16 lasting 30 seconds to 1 minute.

This data thus includes the magnitude of acceleration described in $\mathbb{R}^3$ as well as the magnitude of rotation. The dataset also includes gravity features along with the phones' pitch, roll and yaw. In total, there are 12 features describing an individuals time series. Each participant also included demographic information such as age, gender and weight. The resulting dataset 'A_DeviceMotion_data' contains timeseries respective to each participant and their performed activities.

III. Methods

Multiple methods of time series analysis will be used on the dataset. We have multiple useful data attributes that describe their own time

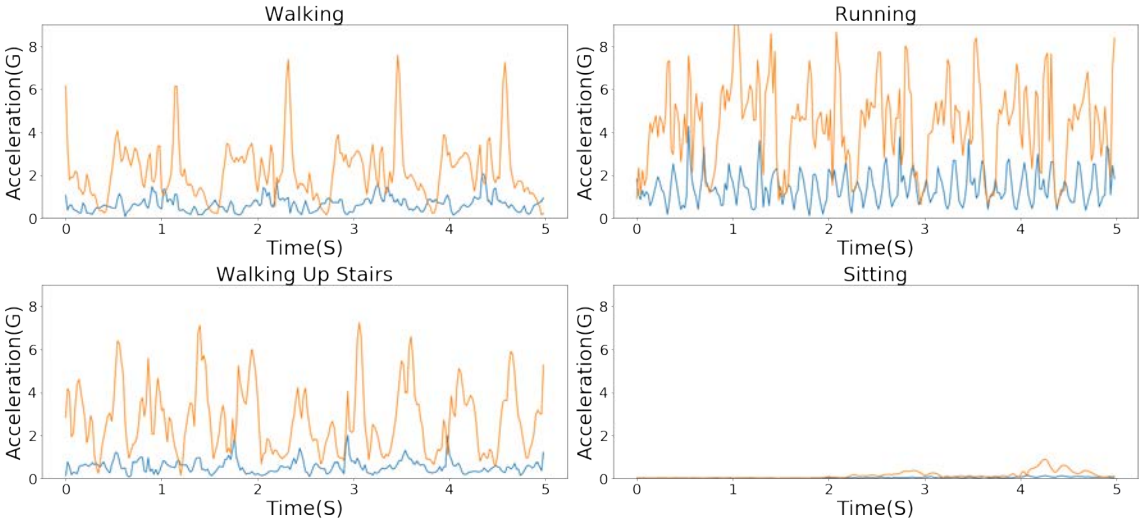


Figure 1: *Multiple activities for subject 1*

series. Our first step in feature selection will be to select only the xyz features for user rotation and acceleration. The reason for this is that the phone's sensors can differentiate between gravity and user-specific motions, and we want to classify based solely on user information. Furthermore, a straightforward method of normalizing the data involves taking the the Euclidean norm of the acceleration and rotation components. Four of the activities from a single individual are plotted above in figure one.

We can discover further information by combining their acceleration and rotation components. Notice that the acceleration norms between standing and sitting are not largely different, yet their rotation vectors vary widely.

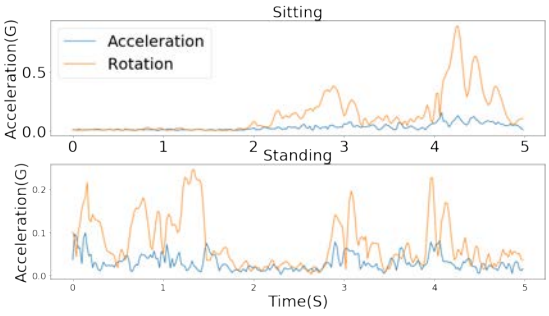


Figure 2: *Walking vs Standing Activities*

For the first method for time series classification, we will investigate the performance of a Linear Support Vector Classifier (LSVC). A LSVC is the simplest form of a support vector machine. It is a supervised learning algorithm capable of classification and regression tasks, seeking to separate the data into groups by maximizing the distance between a decision boundary and the nearest data point, this distance is known as the margin.

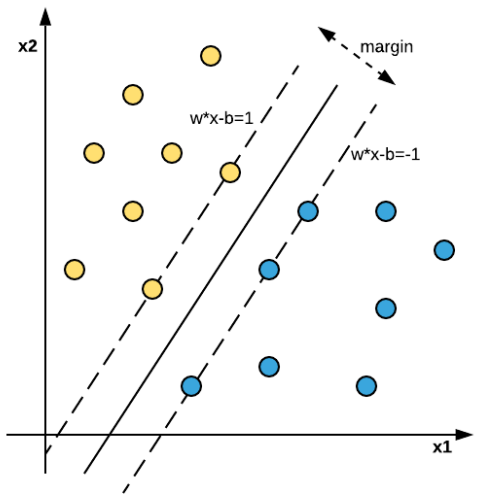


Figure 3: *Linear Support Vector Classifier*

The second algorithm will be a Random

Forest Classifier (RFC). A RFC is an ensemble learning method that constructs an array of decision trees at training time. The predicted class is based off of a voting schema from the trained random generated trees. This is a robust method as it seeks to correct the over-fitting problem inherent to traditional tree based classifiers.

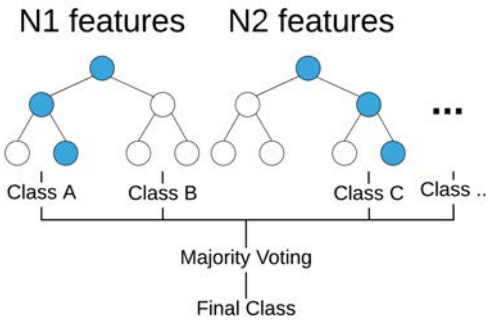


Figure 4: *Random Forest Classifier*

The third and final algorithm is the classic Logistic Regression Classifier (LRC). The LRM has similar mechanics to that of minimizing the least squares error in linear regression. However, LRC seeks to find a maximum likelihood of classifying separate by predicting the mean of a Bernoulli distribution for each sample. The activation function of the LRC is the threshold boundary between each class. This process is often modeled by the Sigmoid function.

$$Sig(t) = \frac{1}{1 + e^{-t}}$$

The next set of methods to compare against include basic feature extraction to be used as descriptive features for the time series. This will include the maximum, mean, standard deviation and skewness for each time series feature. Recall that these features are defined as follows.

$$\mu = \frac{\Sigma_{i=1}^{n} x_i}{n} \tag{1}$$

$$\sigma = \sqrt{\frac{\Sigma_{i=1}^{n}(x_i - \mu)^2}{n}} \tag{2}$$

$$Skew = \frac{[(N-1)(N-2)]}{N} \Sigma_{j=1}^{N} \frac{(x_j - \bar{x})^3}{\sigma_x^3} \tag{3}$$

A key tradeoff to address is that we will also be combining similar tasks to improve model performance. This comes with the cost that we will have fewer activities to differentiate from, since the activity pairs sitting and standing in addition to walking up or down the stairs will be combined into a single timeseries each.
Finally, we will experiment with smoothing each time series via a rolling average. This will help remove outliers and improve classification performance. Large time series will also be broken apart into smaller chunks in order to create uniform-length training and testing datasets in addition to avoiding the loss of data. The cut-off length will be set to 2,000 with shorter time series being padded with repetitions of their existing patterns. Once the models have been trained cross validation will be used to gain better understanding of true model performance.

IV. Results

After processing the time series to extract features and create a uniform-length dataset, basic descriptors of the time series were found. The average length of the raw time series was 2,193, with a minimum length of 377 and a maximum of 8,212. Similar time series were also combined into single classes. This includes the two pairs walking up and down stairs as well as sitting and standing. This results in the four timeseries walking, jogging, climbing stairs (up/down), and being at some form of a standstill (walking/standing). We will classify against these four groups, setting aside twenty percent of the data for testing. Using only the norm of the acceleration vectors, the LSVC showed a promising baseline of 66% test accuracy. Cross validation showed the algorithm to actually be even more accurate, with an average of of 72% accuracy across five data splits.
It is important to note that the total class counts are not evenly distributed. The post-processed dataset primarily contains samples from stair climbing and standing/sitting due to our processing efforts. The total count for

each activity is described below.

Stairs	Walking	Jogging	Still
234	143	75	192

A simple improvement to the LSVC was found by simply smoothing the time series samples. This was done through the sliding window technique in which window sizes of 10 averaged out the time series. Smoothing the series improves performance in the models by removing outliers and noise as displayed below.

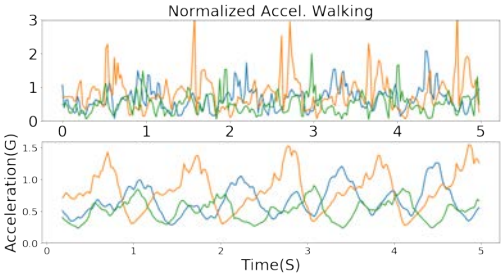


Figure 5: *Walking Signatures of Subjects 1,2,3 Before and After Smoothing*

After smoothing, feature extraction included finding the mean, standard deviation and maximum values for each time series. This resulted in an improvement in the LSVC from 72% to 76%. Finally this same process was applied while also incorporating the rotation rate across each test subject. This dramatically improved performance across all models, resulting in the final cross validation scores listed below.

LSVC	RFC	LRC
91.9%	91.5%	89.1%

The ROC curves for the LSVC is displayed below. The classes are correlated as zero, one, two, and three to walking, jogging, climbing stairs, and idleness, respectively. Note that while all classes showed quality performance, activities pertaining to climbing stairs and idleness had perfect ROC scores due to their significant differences in overall magnitudes of acceleration and rotation. In this example, a logistic regression classification threshold of approximately 0.15 could have upwards of 90% TPR with less than 10% FPR.

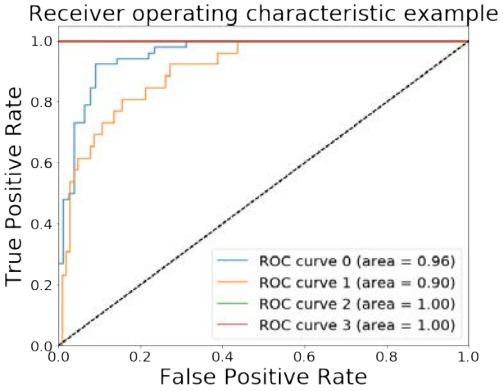


Figure 6: *ROC Curve for Each Class*

As seen in the ROC plot above, some of the activities are classified with 100% true positive rate (standing/sitting for example). All other models had very similar ROC curves, which is most likely due to our limited feature space.

V. Discussion and Conclusions

This report analyzed the performance between three separate models to classify user specific behavioral data collected from smartphone sensor devices. The results showed promising performance with over 90% accuracy for two of the models. This should also be of concern considering these trivial methods are able to identify user activities, imposing on privacy and user security. Simple improvements such as more sophisticated feature extraction and normalization of the data could lead to considerable improvement. Furthermore, streaming data techniques could hold even higher risks for concern as intercepted user data could be used to re-identify an individual, regardless if the data has already been anonymized or not. While there are many benefits to the continued improvements to data analysis and classification, user privacy threats are at an all-time high. Ethical practices must be met with the highest technical standards in order to maintain

user privacy and anonymity in today's modern world.

References

[1] ACM, 2019]IoTDI 2019 Proceedings of the International Conference on Internet of Things Design and Implementation. *Mobile Sensor Data Anonymization, https://www.kaggle.com/malekzadeh/motionsense-dataset*

[2] IEEE Access, 2017]Digital Object Identifier 10.1109/ACCESS.2017 *LSTM Fully Convolutional Networks for Time Series Classification. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8141873*

[3] Thessaloniki 2001]Data Engineering Lab, Department of Informatics. *Feature-based Classification of Time-series Data.*

[4] Massachusetts Institute of Technology, 2001]Media Laboratory *Folk Music Classification Using Hidden Markov Models.*

[5] Remote Sens. 2015]A Hidden Markov Models Approach for Crop Classification.*Linking Crop Phenology to Time Series of Multi-Sensor Remote Sensing Data.*

[6] IEEE 2012]12th International Conference on Hybrid Intelligent Systems (HIS). *Improving Time Series Classification Using Hidden Markov Models.*

Boulder Bicycle Traffic Forecasting

Jacob Munoz
University of Colorado Boulder
jamu0075@colorado.edu

Abstract

Boulder, Colorado, has consistently been recognized as one of the best cities for cyclists. This is based on factors such as the number of riders, safety, path network, and availability of the paths. Boulder has a dedicated riding community that may one day require traffic monitoring to improve safety and commuting time while providing the city of Boulder valuable information on the trends of their riders. The driving hypothesis of this research is that cyclists ride in predictable patterns based on their environment with an end goal of predicting bicycle traffic.

I. Introduction

Boulder, Colorado, has consistently been recognized as one of the best cities for bicyclists in the country[1]. The judging criteria for the ranking includes features such as ridership, safety, network, reach, and acceleration. Network being how well the bike network connects people to destinations, how long it takes to navigate Boulder. Reach is how well the bike network serves everyone equally. Acceleration is the city's commitment to growing bicycling quickly, maintaining and updating infrastructure. In 2019, Boulder was named one of the best cities in the U.S. for bikes out of 500 communities[2].

The goal of this research is to help the city of Boulder understand its cyclists' trends and to forecast bicycle traffic using recent observations and weather data. Cycling is a fantastic environmentally sustainable mode of transportation that should be constantly improving to encourage more riders. There may one day be a need for bicycle traffic mapping similar to that of current car traffic to promote safe and timely commuting.

While the results of this research provide some insight to cyclist patterns, there is plenty of room for forecasting improvements and expansion on location tracking. This is the first step towards real-time bicycle traffic modeling.

II. Data

Two data sets were used throughout this project. The first being bicycle counts at various intersections throughout Boulder and the second being daily weather data. The bicycle data is obtained from the City of Boulder website that has publicly available data that is updated regularly[3]. The weather data is obtained from the National Oceanic and Atmospheric Administration (NOAA) website and is also updated regularly[4].

The bicycle data includes the count of bikes observed at each intersection every 15 minutes. The intersections being observed are highly trafficked and capture many common routes in Boulder. Every intersection has data up until the current day and begins at various dates. Some sets begin early 2015 and others early 2016 but the longest, Folsom & Boulder Creek Path, begins 8/8/2011. It is uncertain how the data is collected but it is assumed to be a simple count of objects passing through each intersection's bicycle lane. This would capture other modes of transport such as skateboards or scooters but the vast majority of traffic in these lanes are assumed to be bicycles.

The weather data includes the daily temperature minimum and maximum in degrees Fahrenheit, snow cover in inches, and precipitation in inches. NOAA has records of weather readings beginning in 1897, although most features where not recorded until early 1900's.

This information is updated on a monthly basis and comes directly from NOAA's observations here in Boulder, Colorado.

While Boulder provides good coverage of data with their bike counts, this research will focus on one. Folsom & Boulder Creek Path is an important intersection given its central location and longevity of data collection. More importantly, this is the only intersection that is connected to the multi-use bike lanes that navigate Boulder. This is an important feature because these multi-use paths are more popular and practical than bike lanes, however Boulder is not yet collecting data on these paths. This intersection will capture some of the traffic that uses multi-use lanes to get into the heart of Boulder.

	date	total	tmax	tmin	precip	snow	snowcover	dayofweek
0	2015-01-03	47	35	15	0.03	0.70	8.0	5
1	2015-01-04	978	25	0	0.01	0.40	8.0	6
2	2015-01-05	813	56	2	0.00	0.00	5.0	0

Figure 1: *Folsom & Boulder Creek Path count with features*

III. Methods

To begin, a monthly count plot was created to view yearly seasonality. Figure 2 shows a clear seasonal correlation that repeats each year. The monthly counts remained mostly constant year after year. This supports the idea that bicycle traffic is a predictable variable.

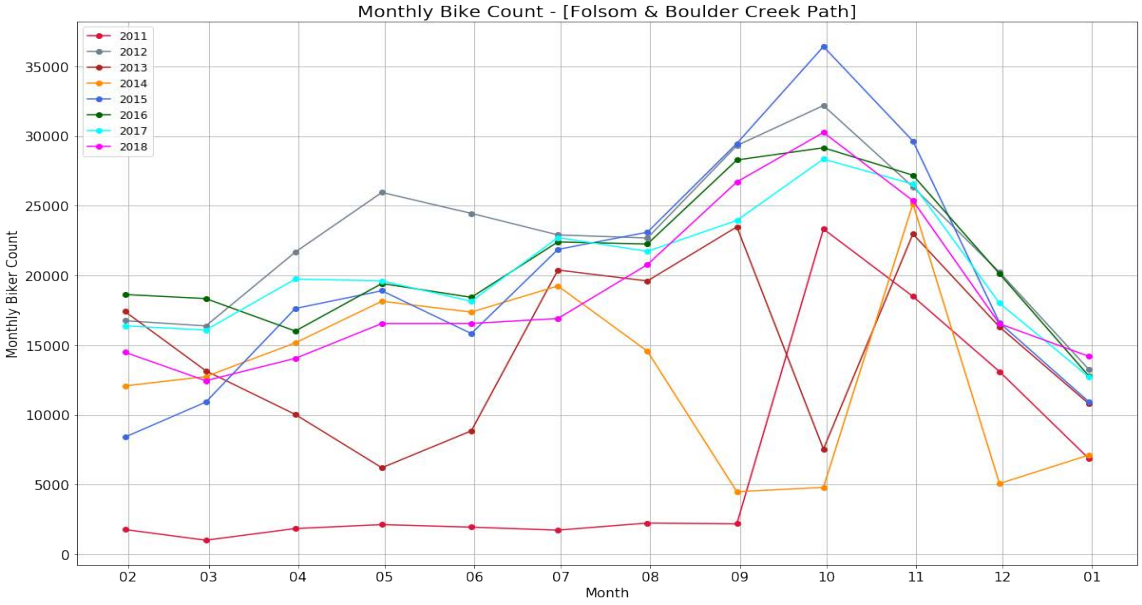


Figure 2: *Monthly bike count at Folsom & Boulder Creek Path 2011-2018*

Figure 3 shows daily temperature and monthly means that follows a very consistent climate pattern. For the last 10 years the monthly means for low and high temperature has remained very consistent with only a few degrees of variation. There is a clear correlation between the temperature and bike count throughout the year that follows the seasons.

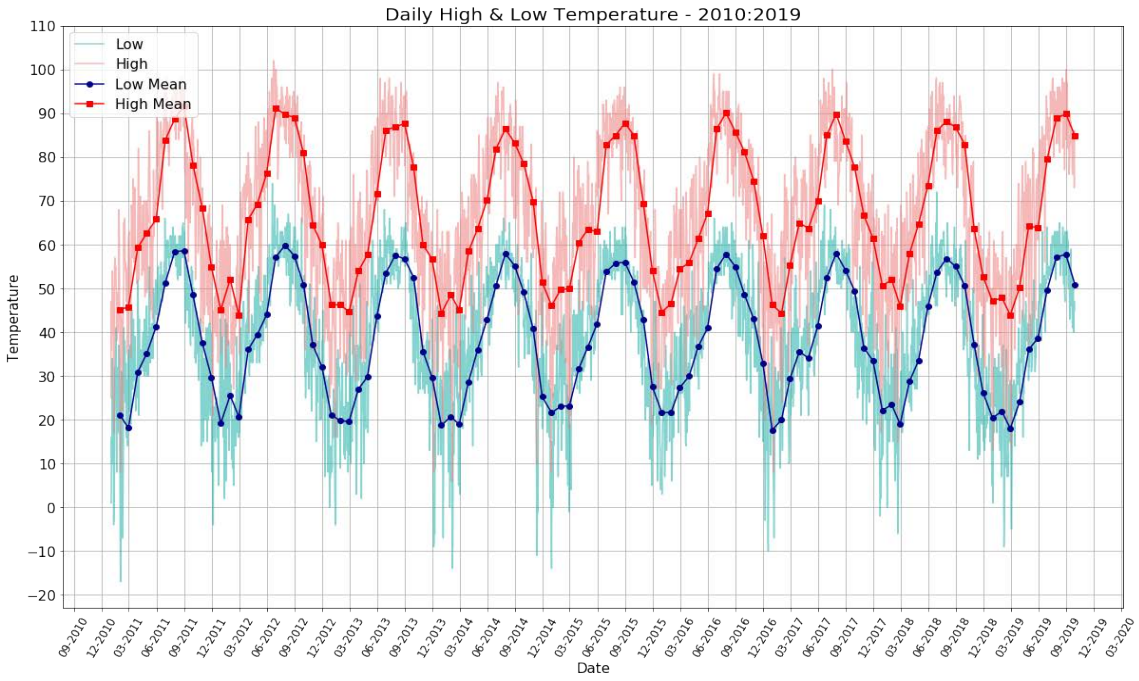


Figure 3: *Monthly average temperature(degrees Fahrenheit) with daily observations 2010-2019*

Before a model could be trained the bicycle count data had to be cleaned. Folsom & Boulder Creek Path's data longevity was appealing, however, after cleaning the data it became comparable to the other intersections (2015-2019). The years 2011-2014 were removed due to wildly different monthly counts compared to more recent years, as shown in figure 2. This may be as a result of physical changes to the bike lane or something else early on but for the purpose of forecasting, that will be saved for later exploration.

The number of daily zero-counts was 41 (2.25% of 2015-2018). When looking at the data the zero-counts could be observed to be grouped by three or more days in a row, and repeated monthly. This is highly unlikely in reality and may suggest the counting mechanism was offline for some scheduled maintenance. For this reason all zero-counts (within the daily grouping) were removed. Furthermore the data had only two outliers and were three or more standard deviations out. These dates did not appear to be any known holiday or event (i.e., Boulder bike-to-work day) and thus were removed. After these changes the skew for the Folsom & Boulder Creek Path count was 0.53. The daily counts was significantly right skewed and the data was standardized in an attempt to mitigate this skew. The data was standardized using sklearn[5] and resulted in a standardized skew of 0.49. The skew is difficult to interpret but it seems likely due to the fact that the counts vary drastically and within a wider range than initially thought. At this point regression was used on the Folsom & Boulder Creek Path data set to make simple predictions.

A simple linear regression model was built using the daily temperature high as the dependent variable with sklearn LinearRegressor[6]. Daily high was used after observing the correlation table, figure 4, for the total and features: daily low, daily high, precipitation, snow, snow cover, and day of the week. Daily max temperature outperformed all other features.

	total	tmax	tmin	precip	snow	snowcover	dayofweek
total	1.000000	0.392215	0.358899	-0.170514	-0.205226	-0.249324	-0.222049
tmax	0.392215	1.000000	0.890617	-0.197196	-0.319635	-0.476324	0.004508
tmin	0.358899	0.890617	1.000000	-0.044057	-0.252350	-0.461541	-0.018285
precip	-0.170514	-0.197196	-0.044057	1.000000	0.541321	0.268217	-0.013865
snow	-0.205226	-0.319635	-0.252350	0.541321	1.000000	0.633644	0.010305
snowcover	-0.249324	-0.476324	-0.461541	0.268217	0.633644	1.000000	0.003632
dayofweek	-0.222049	0.004508	-0.018285	-0.013865	0.010305	0.003632	1.000000

Figure 4: *Folsom & Boulder Creek Path correlation amongst features*

After examining each feature individually, multiple linear regression was used to complicate the model slightly. Features where chosen based on correlation to the total count and observed model performance. The best model is most accurate when using maximum temperature, and day of the week as features.

Ultimately, given the observed behaviour of the data, time series analysis became the focus[7]. When observing the auto correlation and partial auto correlation in figure 5 we can see clear seasonal trends amongst the daily groupings.

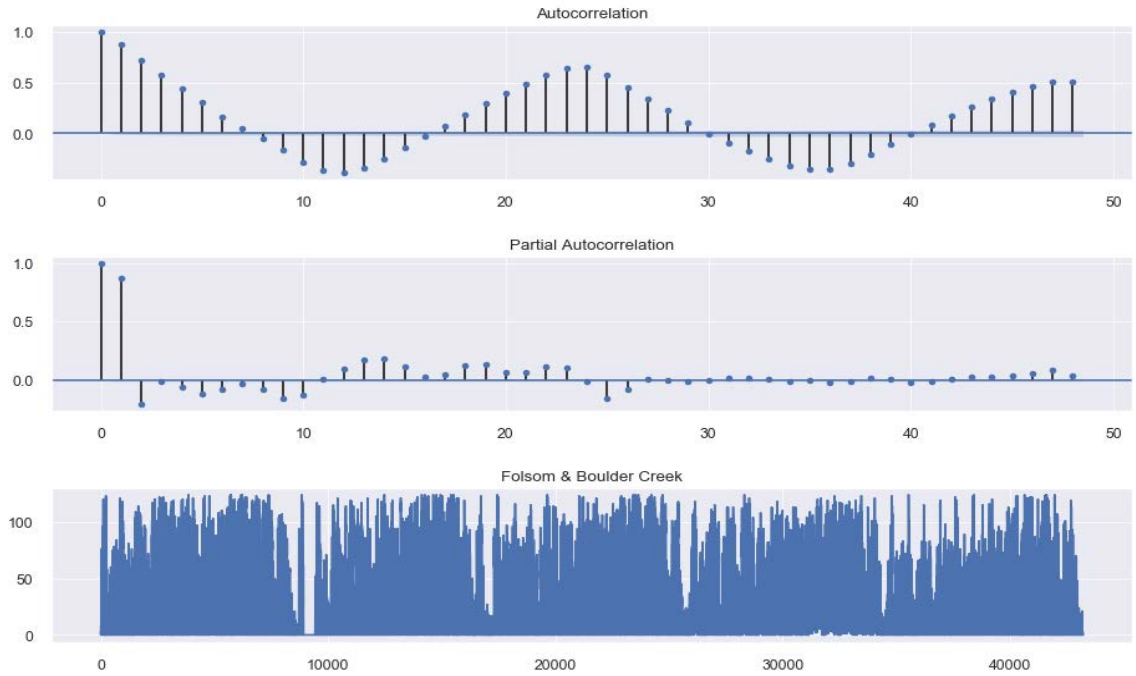


Figure 5: *Autocorrelation and Partial Aurocorrelation plots for Daily counts at Folsom & Boulder Creek Path*

FBprohpet[8] was used for time series forecasting due to its versatility and customization options. It is capable of capturing multiple seasonalities at once including sub-daily, daily, weekly, and yearly. It is also robust in its ability to handle outliers and trend change over time. Sub-daily forecasting was explored with some levels of success but the majority of the modeling was spent working with daily forecasting. That being said, sub-daily forecasting has great potential for success and practical use.

IV. Results

The results of univariate linear regression were not great but showed promise, specifically for maximum daily temperature as a feature. A p-value test was conducted using python StatsModels[9] to confirm correlation between bike count and features. The null hypothesis, there is no correlation between weather and bicycle count, was safely rejected after receiving a p-value of less than 0.01 with each weather feature.

The assumptions of linear regression that were acknowledged include a linear relationship, multivariate normality, no or little multicollinearity, no auto-correlation, and homoscedasticity. Figure 6 shows the correlation between maximum daily temperature and bike count and there appears to be some amount of linear relationship, though not overwhelmingly. As for multivariate normality, the bike count and daily maximum temperature were both standardized to be on the same scale. Multicollinearity was not an issue for univariate analysis but was considered when exploring multivariate regression. Auto-correlation is however present in the data. As observed previously in figure 5, there is clear auto-correlation which negatively affected the performance of linear regression and led to time series analysis. As for homoscedasticity, the residuals in figure 7 appear to have a skew similar to that of the observed values which indicates room for improvement. The best univariate linear model had a root mean squared error of 358.12 and an R-squared value of 0.15. Thus, the model has room for improvement and a multivariate linear regression model was the next step.

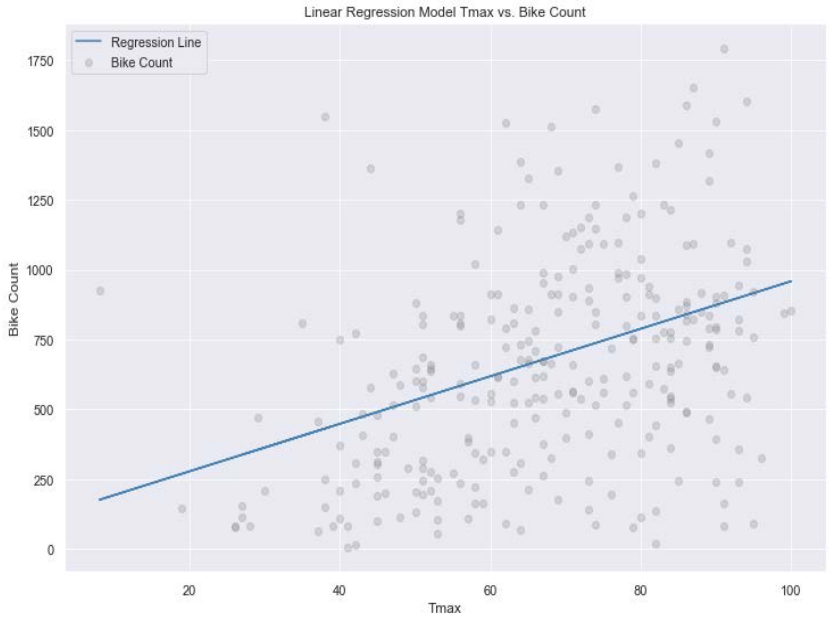


Figure 6: *Daily Count vs. Daily maximum temperature with a univariate regression line*

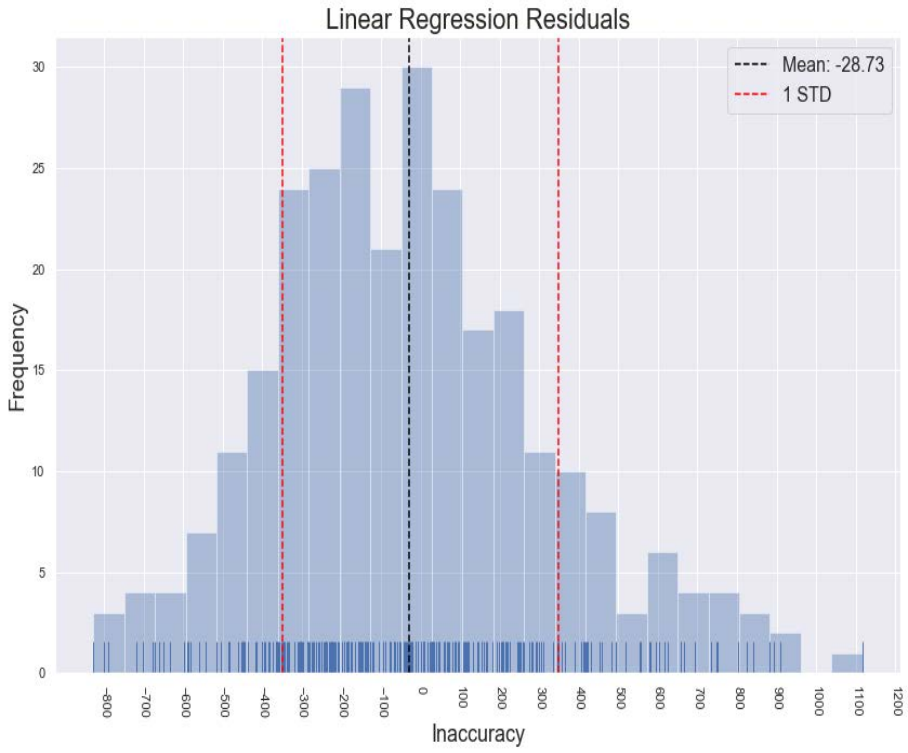


Figure 7: *Residuals for univariate regression predictions*

Many of the features were explored, but the best model was produced with the maximum temperature alongside day of the week as features. Surprisingly, the number of cyclists drops 25% on Saturday and Sunday. This suggests that there are a large number of cyclist who are commuter only, but there are other possible explanations. The root mean squared error got down to 338.85 with an R-squared value of 0.198. The residuals followed the same distribution of the univariate regression, confirming room for improvement. While still not an ideal model, there is significant improvement from the univariate linear model.

Although there is plenty of room for improvements with the regression models, focus shifted towards time series forecasting given the data auto-correlation.

The most success was found within daily forecasting. Using FBprophet, it was possible to forecast based not only on seasonalities but also extra regressors such as daily maximum temperature and day of the week. These extra regressors proved to provide a small increase in accuracy. The model is a multiplicative seasonality with a changepoint-prior-scale of 0.1 and daily, weekly, and yearly seasonality values of ten, eight, and fifteen respectively. This yielded a model that made the prediction in figure 8 below with the most success. Figure 8 shows the predicted values against the observed values. The model is able to capture the general trend quite well however there are several outliers. Daily bike counts certainly follow seasonalities however the unpredictable nature of humans results in dramatic shifts in bicycle count from one day to the next. This results in a model that fits for general trends but struggles with dramatic shifts in activity.

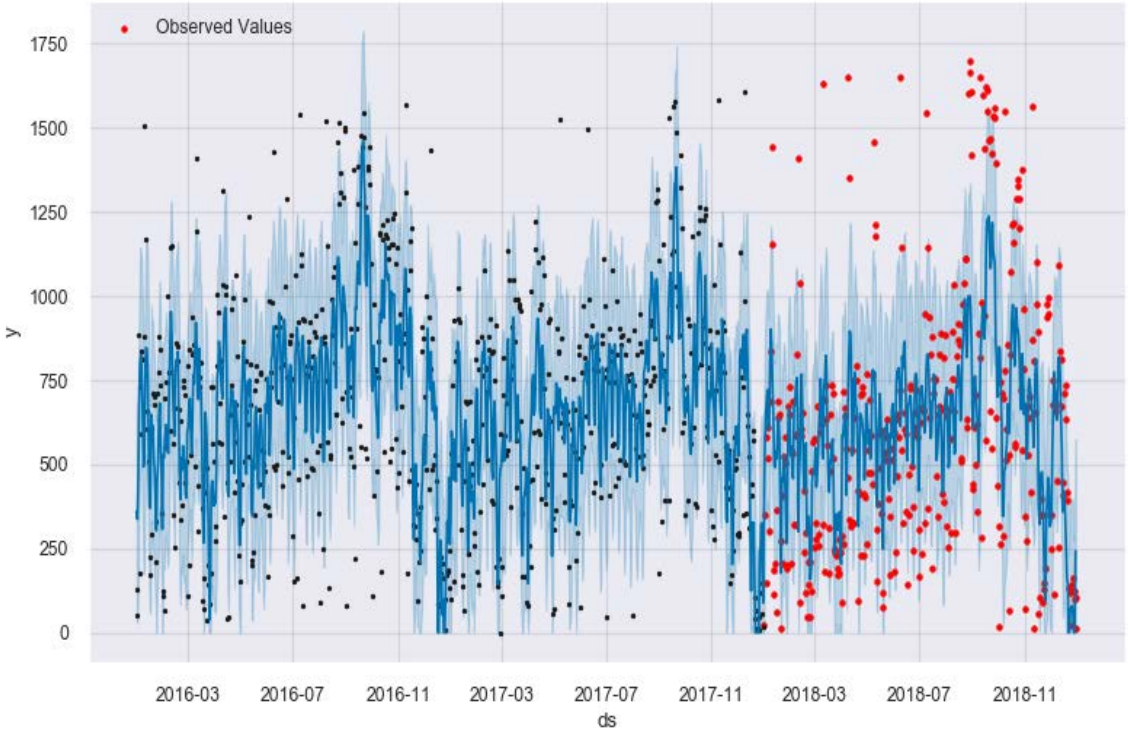


Figure 8: *The FBprophet model trained on 2016 and 2017 to predict 2018*

Figure 9, below, shows the observed values versus the predicted values with the range of possible predictions. It showcases the model's ability to capture the general trend while struggling to account for drastic peaks in activity. This could likely be improved with more time adjusting the models sensitivity to change. The residuals shown in figure 10 for the forecasting model show a much more normal distribution with a mean close to zero. This suggests the time series forecasting model more accurately captures the behaviour of cyclists with a more uniform error.

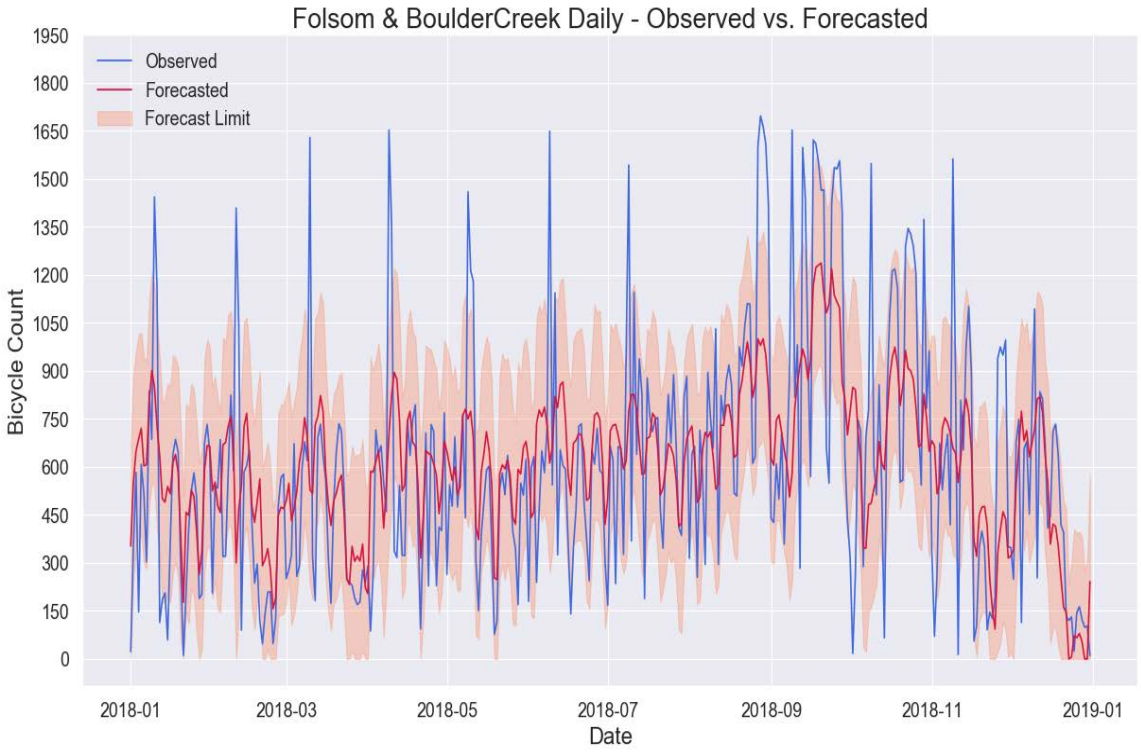


Figure 9: *The FBprophet model predicted values vs. observed values for 2018*

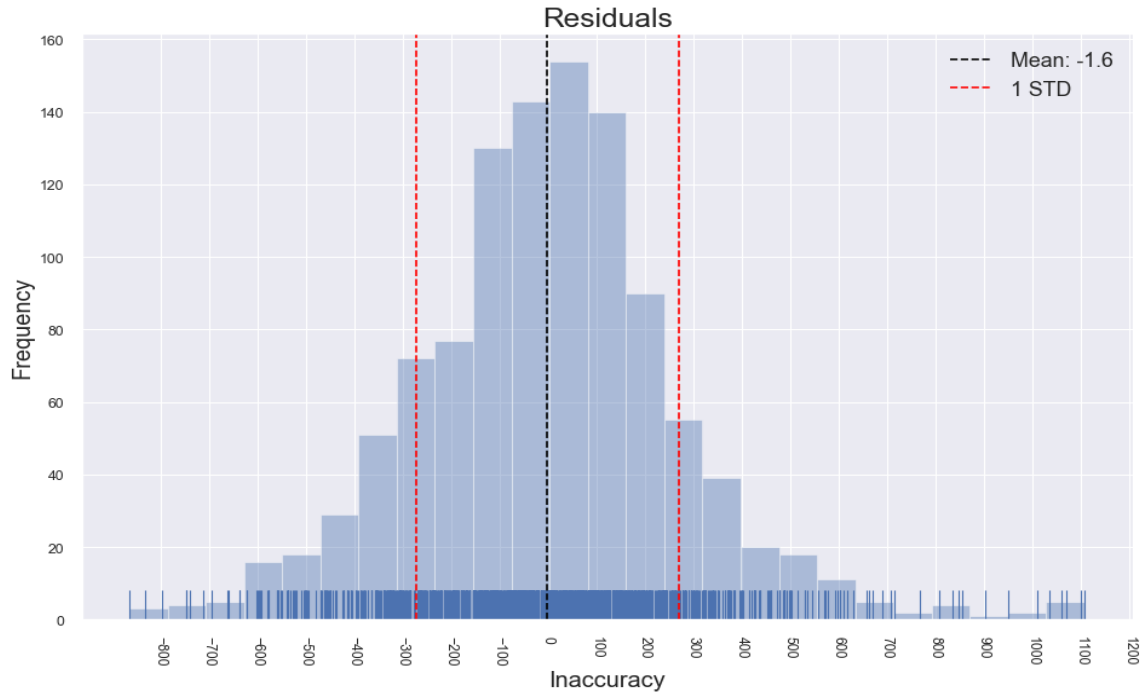


Figure 10: *The FBprophet model residuals*

When comparing the observed values to the predicted values they show a definite correlation, a calculated correlation coefficient[10] of 0.67. The model is able to more accurately predict counts between 450 and 700, possibly because the daily average falls in this range. Finally, the calculated root mean squared error of 270.90, a significant improvement upon the multivariate regression model.

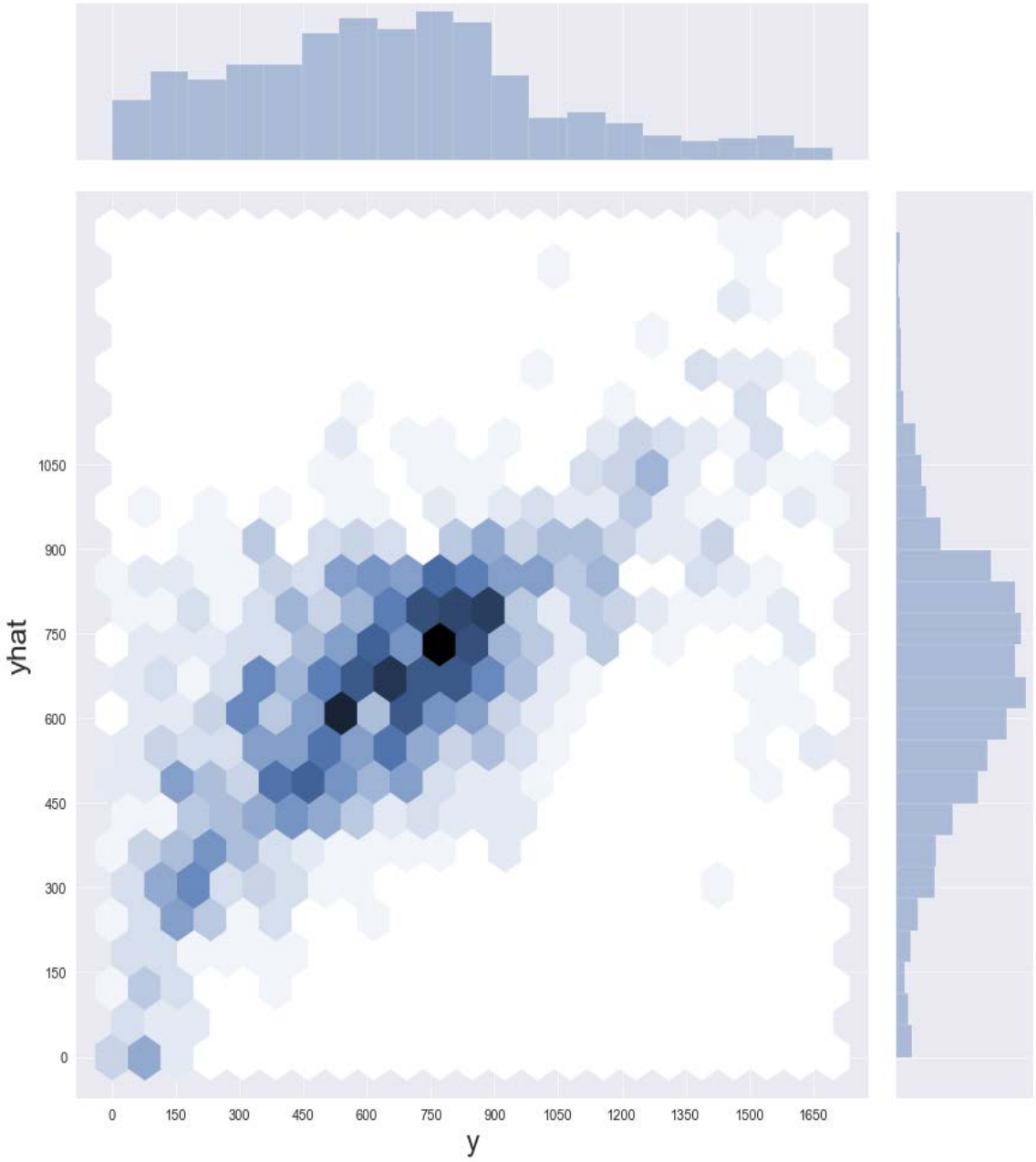


Figure 11: *Correlation plot of observed values vs. predicted values.*

V. Discussion

Boulder, Colorado, has a great cycling community that is ever growing and the data can be used to further improve the infrastructure in place. Forecasting bicycle traffic has provided an insight into when and where people bike given a small amount of information on the environment. Originally, the goal of this research was to provide a broad analysis of every intersection provided by the data. This could provide a more specific analysis of when and where people ride including northbound and southbound traffic during morning and evening work commutes or if the quality of the bike lane affects traffic. The shift in focus to forecasting came with the hopes of predicting traffic by understanding how the weather affects cyclists. In the future there may be a practical use for real-time bicycle traffic monitoring and predicting to improve the commuting experience for riders and to collect useful rider data to improve the infrastructure. While the daily forecasting returned decent results, there is plenty of room for improvement in the future. With more time we would like to explore more features such as the length of the day, or hourly weather data to better predict sub-daily traffic. The focus of this research was daily forecasting, however, sub-daily forecasting is possible in the future given more time and could be more useful. This research was the first step towards better understanding the cyclist community in Boulder, Colorado, and we are excited to continue this work into the future.

References

[1] [AC Shilton and the Bicycling Magazine Editors, 2018]
The Best Bike Cities in America,
https://www.bicycling.com/culture/a23676188/best-bike-cities-2018/

[2] [People For Bikes, 2019]
2019 City Ratings: Top Overall Cities,
https://peopleforbikes.org/blog/2019-city-ratings-top-5-overall-cities/

[3] [City of Boulder, 2019]
City of Boulder Bicycle Traffic Counts,
https://bouldercolorado.gov/open-data/bicycle-traffic-counts/

[4] [NOAA, 2019]
NOAA Boulder Daily Data,
https://www.esrl.noaa.gov/psd/boulder/getdata.html

[5] [Scikit Learn, 2019]
sklearn preprocessing StandardScaler,
https://scikitlearn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[6] [Scikit Learn, 2019]
sklearn LinearRegressor,
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegres

[7] [Hyndman, R.J., Athanasopoulos, G, 2018]
Forecasting: principles and practice, 2nd edition, OTexts: Melbourne, Australia,
OTexts.com/fpp2

[8] [Prophet, 2019]
Facebook Prophet,
https://facebook.github.io/prophet/

[9] [StatsModels, 2019]
StatsModels p-values,
https://www.statsmodels.org/stable/generated/statsmodels.regression.linear_model.OLSResults.pvalues.html

[10] [Pandas, 2019]
Pandas Correlation,
https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html

Detection of Duplicate Sentences in Online Resource Platforms using Deep Embeddings

Karthik Palavalli
University of Colorado Boulder
karthik.palavalli@colorado.edu

Abstract

The advent of online resource platforms such as Stack Overflow, Stack Exchange, Research Gate etc., has made information more readily available to the common public. These platforms also enable new users to ask a wide range of questions and provide a great marketplace for the community to help each other out. As the data blooms, one of the key aspects to maintaining such large collections is to have a structure in it, in our case having all the discussion related to a topic in one place. In order to achieve this, it is important to detect duplicates and prevent multiple discussions of the same topic. We propose methods to address this critical issue in such large scale information systems. This paper surveys methods from unsupervised to deep learning based embeddings to detect such duplicates in a question corpus. We achieve all of this using the raw text with the help of pre-trained models on the language corpuses.

I. Introduction

In an era of massive data growth on online platforms, duplicates are a very common pattern to observe, be it text, images or sometimes even videos. There have been various efforts to understand and detect these redundancies in data [1] [2].

For images, simple methods such as pixel level matching to more complex architectures such as scene prediction have been experimented to understand if two images are representing the same information in other words [3] [4]. For text on the other hand, there have been various algorithms ranging from topic modelling [5] to deep learning based approaches [6] to the detect if two sentences convey the same information.

In this paper, we explore a transformer architecture based deep learning approach to embed sentences [7]. We then look at a pairwise cosine distance for each of these sentence embeddings to predict if they are duplicates are not, experimentally we found out the score to be 0.05 for the dataset in use. We also look at various unsupervised methods to group these embeddings and cluster purity metrics to optimize these clustering techniques.

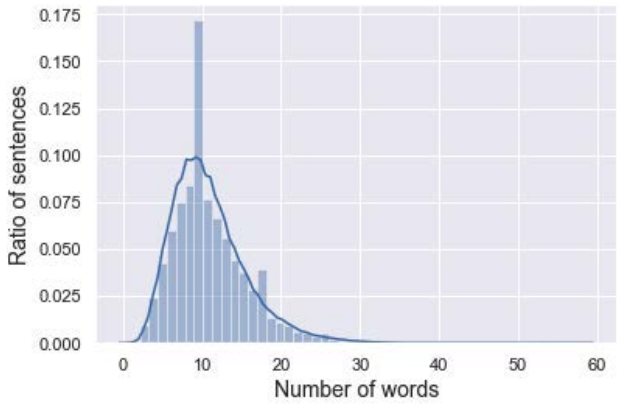
II. Data

The dataset used in this research is the Stack Overflow Data [8] published on Kaggle. It is a 190 GB dataset that can be accessed using the Big Query API provided. The partial schema of the data, used for this research is shown below:

Each entry consists of a unique id, title of the question, body of the question, answers among many others listed in the database schema table 1. This research aims to find duplicate/similar questions using just the questions in the natural form. For the scope of this research a sample of the data is being used due to the computational limits. We consider 3000 queries from the complete dataset. These 3000 queries produce 4,498,500 question pairs whose similarity we analyze using the methods described below.

Name	Type	Description
id	INTEGER	Unique Identifier
title	STRING	Title of the question
body	STRING	Question Description
answer id	INTEGER	Matched answer
comment id	INTEGER	Comment Identifier
date	DATE	Creation Date
owner	STRING	Creator Info
score	INTEGER	Question rank
tags	STRING	Topic tags
count	INTEGER	View count

Table 1: *Stack OverFlow Data Scheme*




```
count    100000.000000
mean         10.913890
std           4.707936
min           1.000000
25%           8.000000
50%          10.000000
75%          13.000000
max          58.000000
dtype: float64
```

Figure 1: *Distribution of sentence lengths*

III. Methods

Upon initial analysis of the sentences, we observed that that the length of sentences were distributed close to normal with a good number of outliers, depicted in figure 1. For any standard sentence embedding to work good it is important that the sentence lengths to be close near each other. As a part of the first pre-processing step we removed sentence pairs that had word lengths greater than 20.

The second stage of pre-processing was to filter out words with low statistical significance [9]. The process of stop word removal used here was based on the standard set of statistically insignificant words.

Further pre-processing was done to normalize the words before embedding. English language has a lot of word forms, lemmatized [10] was done to make them variant neutral. This has shown to improve accuracies of word embeddings, by treating all of forms of the same word to have the same distributional respresentation.

After the pre-processing stage, we produce the embeddings using the BERT [7] which has a transformer architecture as shown in figure 2. We generate sentence level embeddings of 1024 dimensions. K means [11] was used to get an overview of the similar question groups. This method was used as first check to see if there were actually duplicates in the given dataset, and if the current embedding method was good enough to capture these duplicates. K means is a parametric unsupervised clustering method which given a set of observations$(x_1, x_2, ...x_n)$, where each individual observation is a vector in a m dimensional space, groups them into K clusters by the following convergence metric (μ represents the cluster centres):

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2$$

Silhouette score [12] III is used in conjunction with K means to predict accurate parameters for the clustering method. Silhouette is a method to measure the purity of the cluster at a particular configuration. From the figure we can observe there was a high accuracy of silhouette score at K = 15.

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i)))}$$

The clusters showed the data contained duplicates, hence once this was affirmed it was important to accurately tag the exact sentences as the duplicates. These sentence vectors are compared against each other for closeness, figure 3 shows the scores for the 4,498,500 pairs.

As shown in the figure 3, there are good number of queries which are very close to each other (distance less than 0.05). And upon spot checking they were found to duplicates semantically.

All of the above methods aimed at detecting duplicates in an already present corpus, while it is a great way to tag duplicates, it does not predict questions in real-time. We built a K-NN model to predict the close sentences for a new incoming sentence. This model suggests the user if his question meant something that was already present in the database and hence prevent duplicates at the source.

IV. Results

Figure 4 shows that cluster state of the 3000 question pairs and the most of the close sentences in the figure were in-fact semantically similar, also we observe that the silhouette helped in estimating the K for K-means, and from figure 4 we also see that for K=15 the purity is at the peak.

At the moment, there is low threshold on cosine score and the accuracy is judged based on spot checking. On every run 10 question pair are chosen in random and analyzed for the duplicates. This process was repeated 15 times for each run, and the computer predicted duplicates at an accuracy of 83.33%.

Finally, we also have a small annotated dataset of 15 duplicate pairs and 15 non duplicate pairs handpicked. The K-NN model predicts the duplicate within the top 3, for 10 out of 15 pairs, making the accuracy 66.67%.

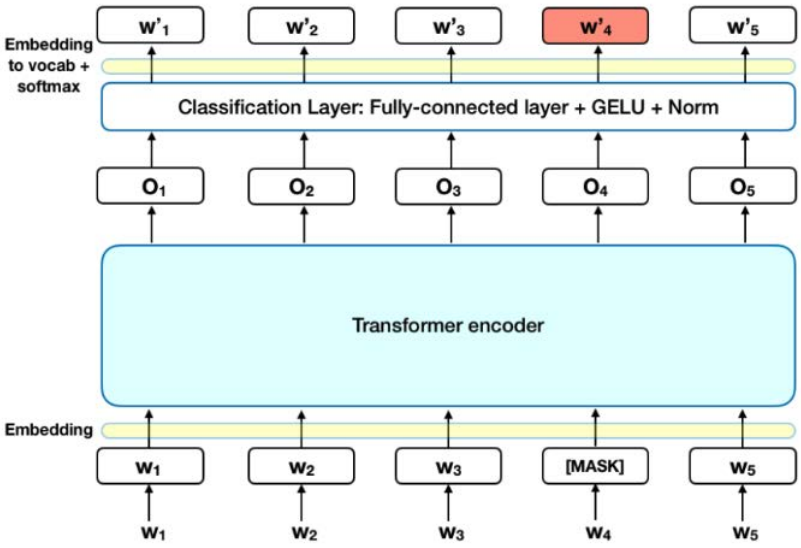


Figure 2: *BERT Architecture*

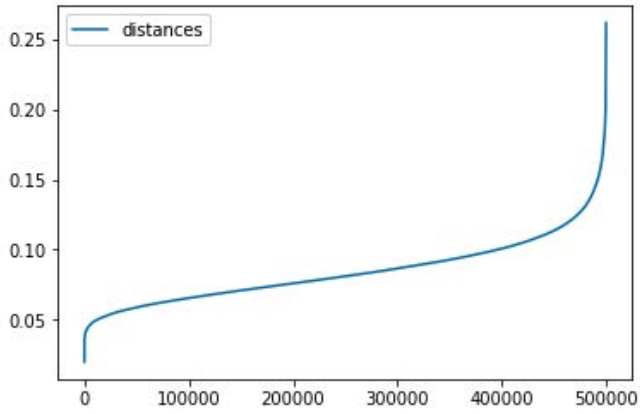


Figure 3: *Question similarity using Cosine Distance*

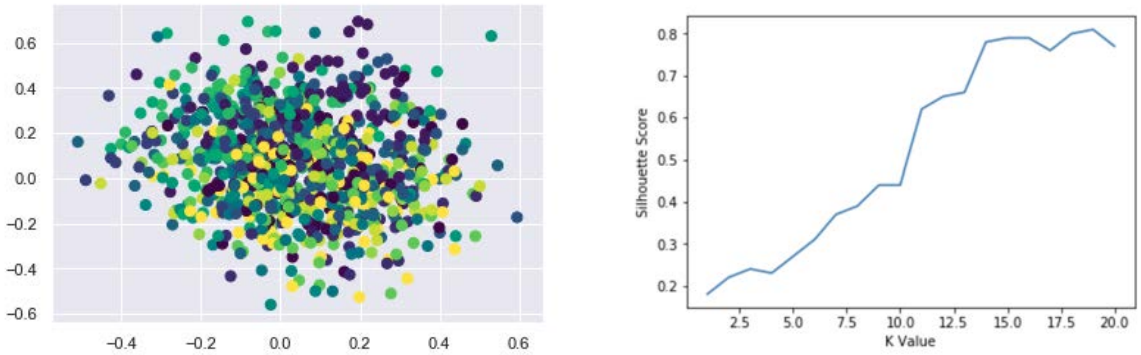


Figure 4: *Clusters and their convergence based on K Value*

Source Sentence	Neighbouring Sentences
Generate a line graph	Make adversarial images with foolbox I want to add image share button in my android app Unable to reconnect named pipe from nodejs
Error while loading the test	Custom throttle unit test Is there any way to add html file/report to JIRA Issue

Table 2: *KNN Predictions*

Some of the model predictions is shown in table 2.

V. Discussion and Conclusions

As a part of future work, we aim at fine-tuning the BERT embeddings to better suit the database.

Context and intent recognition [13] is a technique that has shown to produce high accuracies for sentences with a lot of contextual words, and it is also a majorly used technique in industry for factoid questions. This is something we hope to explore on the Stack Over-Flow dataset.

Finally, we aim to increase the number of annotated data-points and compute classification metrics such as precision, recall, F1 etc to better measure the performance of the models.

References

[1] Sujith Viswanathan, Nikhil Damodaran, Anson Simon, Anon George, M Anand Kumar, and KP Soman. Detection of duplicates in quora and twitter corpus. In *Advances in Big Data and Cloud Computing*, pages 519–528. Springer, 2019.

[2] Andrada Maria Pumnea. Master thesis advancing duplicate question detection with deep learning. 2018.

[3] Yan Ke, Rahul Sukthankar, Larry Huston, Yan Ke, and Rahul Sukthankar. Efficient near-duplicate detection and sub-image retrieval. In *Acm Multimedia*, volume 4, page 5. Citeseer, 2004.

[4] Xunyu Pan and Siwei Lyu. Region duplication detection using image feature matching. *IEEE Transactions on Information Forensics and Security*, 5(4):857–867, 2010.

[5] Alexandra Schofield, Laure Thompson, and David Mimno. Quantifying the effects of text duplication on semantic models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2737–2747, 2017.

[6] Yushi Homma, Stuart Sy, and Christopher Yeh. Detecting duplicate questions with deep learning. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016). IEEE*, pages 1–8, 2016.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[8] Stack Overflow. Stack Overflow Data, howpublished = `https://www.kaggle.com/stackoverflow/stackoverflow/metadata`. Accessed: 2019-09-26.

[9] Hassan Saif, Miriam Fernández, Yulan He, and Harith Alani. On stopwords, filtering and data sparsity for sentiment analysis of twitter. 2014.

[10] Giorgio Maria Di Nunzio and Federica Vezzani. A linguistic failure analysis of classification of medical publications: A study on stemming vs lemmatization.

[11] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *JSTOR: Applied Statistics*, 28(1):100–108, 1979.

[12] Peter Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65, November 1987.

[13] Richard Kelley, Alireza Tavakkoli, Christopher King, Amol Ambardekar, Monica Nicolescu, and Mircea Nicolescu. Context-based bayesian intent recognition. *IEEE Transactions on Autonomous Mental Development*, 4(3):215–225, 2012.

Predicting Traffic Congestion in Cities

Srishti Rawal
University of Colorado Boulder
srishti.rawal@colorado.edu

Abstract

Road traffic is an inevitable obstacle for commuters in densely populated cities. On an average, a commuter spends fifty four hours being stuck in traffic. In addition to costing large amount of time, traffic congestion leads to more usage of fuel and causes pollution. Identifying patterns in traffic is necessary to improve road conditions to support the growing population in cities. These would enable traffic operator and engineers to proactively take appropriate measures such as improving road infrastructure and changing traffic light strategies. This paper proposes a machine learning based strategy to identify highly congested intersections in four major cities of the United States. Among the three regression models used in this paper, Random Forest Regressor predicted the Total Wait Time and Total Distance Stopped most accurately. The predicted root mean square error values averaged for Total Wait Time and Total Distance Stopped across the three percentiles i.e 20th, 50th and 80th was 27.06. This means that the predicted values deviate from actual values of the six variables by 27.07 units.

I. Introduction

Over the last few decades, there has been exponential growth in population and urbanization. Millions of people move to the bigger cities to get better jobs and facilities. With this movement, the bigger cities are becoming denser in population day by day. Despite the ever increasing need for road development, many places are suffering from the problem of regular traffic in certain areas.

It is not uncommon to see people getting stuck at the same junction every day during their commute. This calls for improvements in road infrastructure, analysis of traffic light strategies or better public transport to lower traffic congestion in cities.

Several road traffic analysis, models and prediction methods have been developed to understand traffic conditions and predict congestion. Some works have utilized deep learning, hidden markov and time series models. For instance, in [1] deep learning models using GPUs and in memory computing have been utilized to predict traffic congestion points. Support Vector Regression and Discrete Fourier Transform have been used in [2] to predict traffic flow during holidays.

In this paper, a data driven approach has been proposed to predict the congestion patterns at intersections in four cities of United States. Three different machine learning models have been utilized for prediction and comparative analysis is done to find the most optimal algorithm for this scenario.

The rest of this paper is organized as follows. Section 2 gives description of the dataset used. In section 3, methods used for predicting congestion are described in detail. Section 4 presents and describes the results obtained. Finally, conclusion and direction for future research for provided in Section 5.

II. Data

The data being used for this study is comprised of information collected by BigQuery from commercial vehicles for four cities, namely, Atlanta, Philadelphia, Boston and Chicago. It contains metrics which have been aggregated and grouped by intersection, month, hour of the day, direction of the vehicle and whether the day is a weekend or not. Metrics used to identify congestion are in the form of percentiles of total time stopped at an intersection, time from first stop and distance from first stop

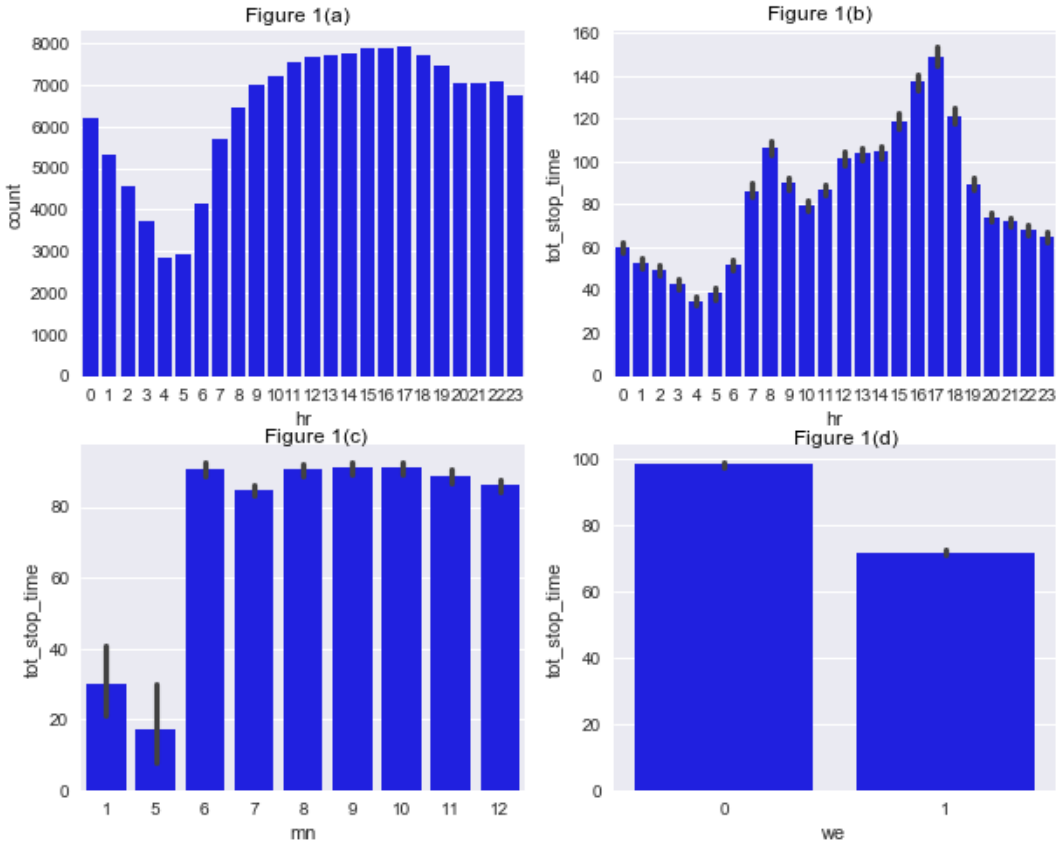


Figure 1: *Atlanta data*

at the intersection.

In the dataset, there are a total of 857,409 entries out of which 150,231 are for Atlanta, 180,398 for Boston, 132,944 for Chicago and 385,647 for Philadelphia. Each entry has the coordinates of the intersection which are utilized to identify a network of traffic by plotting them on an interactive map to visualize the propagation of vehicles. The training data is split into test and train sets.

The models are run on the test set obtained after splitting. Congestion is found by predicting the 20th, 50th and 80th percentiles of total time stopped and total distance from first stop for entries in the test set.

Figure 1 shows the plots of training data for the city Atlanta. In 1(a), the counts of entries for each hour of the day is displayed. It can be observed that data recorded between the early morning hours is substantially less compared to data recorded between 7am to 12am. 1(b) has total time stopped for each row on the y axis and hour of the day on x axis. From 1(a) and 1(b), it can be inferred that congestion might not be directly related to the number of vehicles on the road at a particular time. Vehicles stop for maximum duration at intersections between 5pm to 6pm. Figure 1(c) plots total time stopped vs month of the year and 1(d) plots total time stopped vs whether the day was a weekend or week day.

III. Methods

Data obtained in flat csv files was converted to dataframes using Pandas[3] as it provides multiple in built functionalities for cleaning, transforming and analyzing data.

Basic descriptive methods were used to find the means and standard deviation of all features. This was done to find out the range and check the variation in values of different variables. It was observed that the data was highly zero inflated for the columns being predicted. Also, features that represented similar data were removed from training. For example, the feature "path" depended on features "entry heading", "exit heading", "entryStreetName" and "exitStreetName" was removed the the set. To find the correlation between coloums, spearman correlation was used to identify relationship between the variables. High correlation was found between time based features.

To visualize relation between time based metric with other features, barplots were plotted as total time stopped on the y-axis and hour, month, weekend on the x axis. Also, a plot between directions and total time stopped was plotted to identify whether a particular entry and exit heading cause delay for vehicle to move.

The regression algorithms used in this study have been described below:

Linear Regression: A simple but powerful technique that fits the data into a linear equation in which the dependent variables are used to predict the independent(unknown) information.

Random Forest Regressor: A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True (default).

Multi Layered Perceptron Regressor: MLPRegressor trains iteratively since at each time step the partial derivatives of the loss function with respect to the model parameters are computed to update the parameters. It can also have a regularization term added to the loss function that shrinks model parameters to prevent overfitting.This model optimizes the squared-loss using LBFGS or stochastic gradient descent.

Evaluation of the models is done using the metrics Mean Absolute Error(MAE) and Mean Squared Error(MSE).
Mean Absolute Error: This is found by calculating the absolute prediction error for each entry in the test set. Mean of the absolute prediction errors gives Mean Absolute Error.

$$\text{MAE}= \frac{1}{n}\sum_{t=1}^{n}|e_t|$$

Root Mean Squared Error: This metric is calculated by taking the square root of Mean Squared Error which basically measures the average squared error of the predictions.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}e_t^2}$$

Visualization of results over the maps is done by using plotly.express[4] using the mapbox layout option. The were made interactive by varying the results over hours. Using the visualization, the propagation of traffic over a network of intersections can be identified. Also, the points which are highly congested at odd times like before 7am were found.

IV. Results

The value of root mean squared error and mean absolute error obtained by using the three regressors are mentioned in Tables 1, 2 and 3.

Variable	MAE	RMSE
Total Time Stopped - 20th Percentile	3.13	7.18
Total Time Stopped - 50th Percentile	10.45	15.56
Total Time Stopped - 80th Percentile	19.91	27.91
Distance to First Stop - 20th Percentile	11.82	27.63
Distance to First Stop - 50th Percentile	38.11	71.05
Distance to First Stop - 80th Percentile	38.24	71.10

Table 1: *Linear Regression*

Figure 2(a)

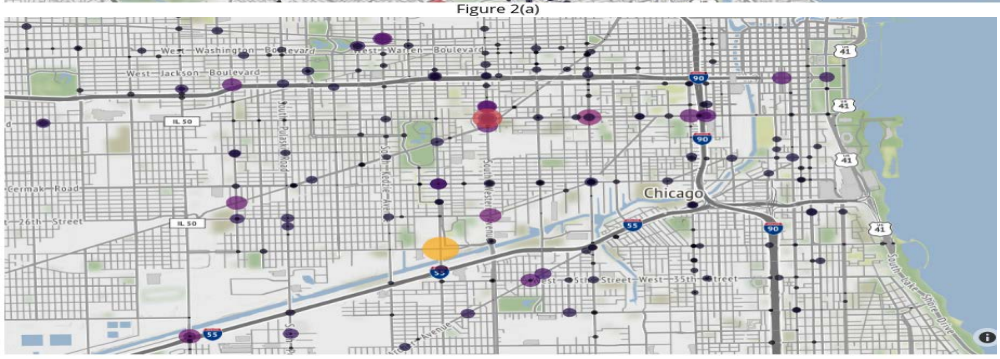


Figure 2(b)

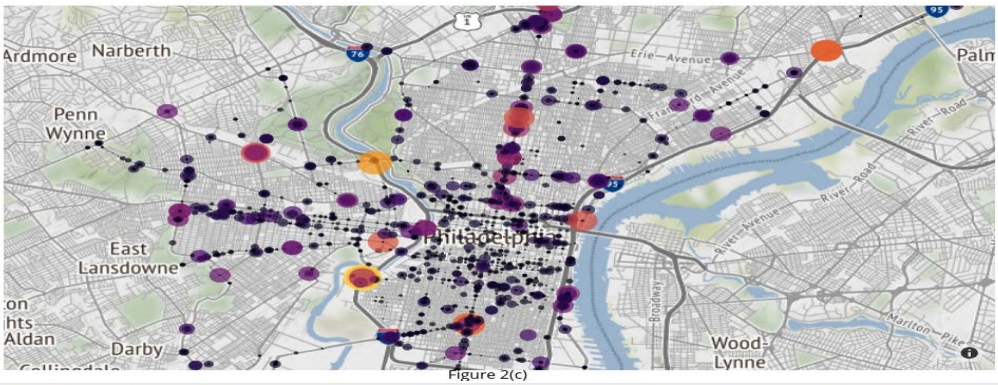


Figure 2(c)

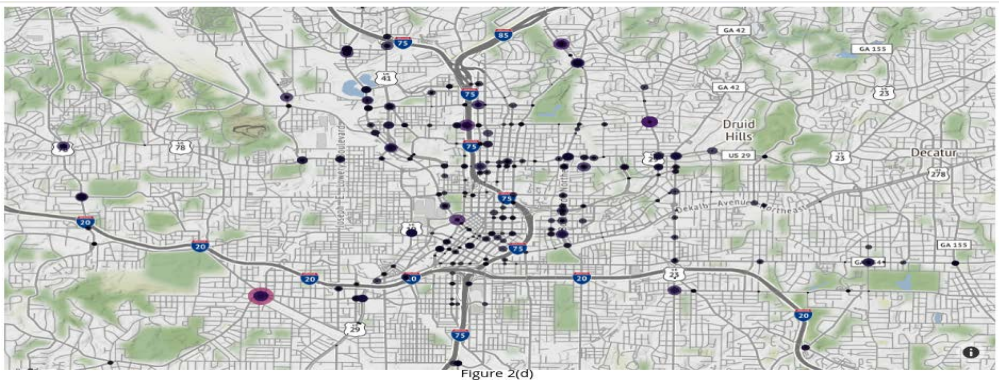


Figure 2(d)

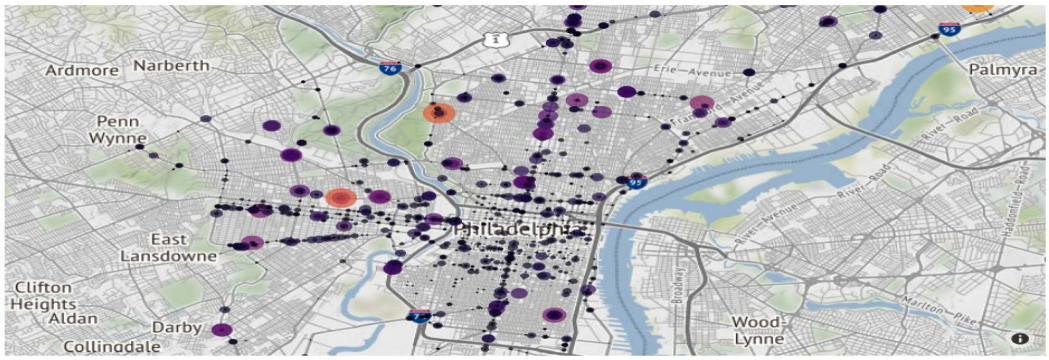


Figure 2(a)

Figure 2(b)

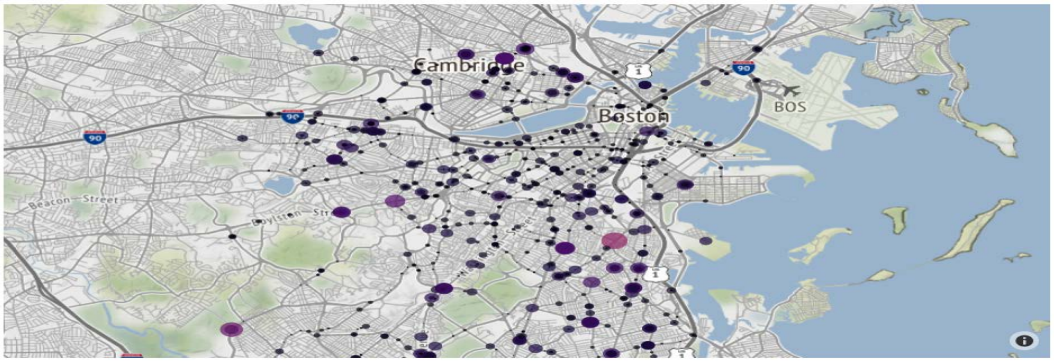


Figure 2(c)

Figure 2(d)

Variable	MAE	RMSE
Total Time Stopped - 20th Percentile	2.17	6.03
Total Time Stopped - 50th Percentile	5.75	10.46
Total Time Stopped - 80th Percentile	10.77	17.88
Distance to First Stop - 20th Percentile	8.31	24.70
Distance to First Stop - 50th Percentile	21.13	51.56
Distance to First Stop - 80th Percentile	21.15	51.70

Table 2: *Random Forest Regressor*

From the tables 1, 2 and 3 it can be concluded that Random Forest Regressor results in least error while prediction. Figure two plots the predicted 20th percentile of total time stopped for the four cities. The color scale for the points varies from black to yellow where yellow stands for the highest value. The visuals were made interactive to display the variation in congestion points at different hours.

Variable	MAE	RMSE
Total Time Stopped - 20th Percentile	3.32	7.20
Total Time Stopped - 50th Percentile	14.19	17.26
Total Time Stopped - 80th Percentile	169.76	173.12
Distance to First Stop - 20th Percentile	175.82	178.08
Distance to First Stop - 50th Percentile	97.71	111.89
Distance to First Stop - 80th Percentile	32.31	73.30

Table 3: *Multi Layered Perceptron Regressor*

Similarly, Figure 3 shows visualization of Distance to First Stop for the vehicles in first 20th percentile.

V. Discussion and Conclusions

The results obtained so far can enable us to identify the location of highly congested as well as low congestion intersections in the cities. Experiments indicate that Random Forest Regressor is the best among the three regression techniques used for predicting congestion. Through the visualizations, a pattern of network in traffic can also be observed. Better insights into the causes of road congestion are vital for growth of modern cities. Focusing on these areas to look for potential problems in road infrastructure or traffic signal timings can lead to significant improvements.

The paper addresses the problem of predicting congested intersection in four cities ahead of time. Eventually, the goal is to look for infrastructure improvement in such intersections. Since the outliers are not necessarily the points where cause of traffic is dense population but they have more to do with infrastructure issues, a traffic engineer can study these points to determine which roads need more maintenance and define better road systems.

The future work will focus on hypermeter tuning for the models. This can be especially useful for the predictions made by Multi Layered Perceptron. Also, we'd like to gather a larger dataset and make use of Big Data Technologies. Traffic conditions are impacted by weather, road conditions which have not been considered in the current study. These features can be included in to improve the results.

References

[1] Muhammad Aqib; Rashid Mehmood; Ahmed Alzahrani; Iyad Katib; Aiiad Albeshri; Saleh M. Altowaijri, "Smarter Traffic Prediction Using Big Data, In-Memory Computing, Deep Learning and GPUs", 2019 May, PubMed Central

[2] Xianglong Luo; Danyang Li; Shengrui Zhang, "Traffic Flow Prediction during the Holidays Based on DFT and SVR", Journal of Sensors, vol. 2019, Article ID 6461450, 2019.

[3] Wes McKinney, "Data Structures for Statistical Computing in Python", Proceedings of the 9th Python in Science Conference, 51-56, 2010

[4] Plotly Technologies Inc., "Collaborative data science"

[5] Sun Ye, "Research on Urban Road Traffic Congestion Charging Based on Sustainable Development", 2012 International Conference on Applied Physics and Industrial Engineering

[6] Jerome H Friedman, "Multivariate

adaptive regression splines", The annals of statistics, 1991.

[7] S.A.M. Ostring ; H. Sirisena, "The influence of long-range dependence on traffic prediction", ICC 2001. IEEE International Conference on Communications..

[8] BigQuery-Geotab Intersection Congestion, web site: www.kaggle.com

An Analysis of the Popularity of Facebook News Posts

Thanika Reddy
University of Colorado Boulder
thanika.reddy@colorado.edu

Abstract

This study uses the Facebook News Dataset to assess the effect of various factors on the popularity of news posts on Facebook. The insights gained can aid in the creation of posts that reach a wider audience and can also lead to a better understanding of what social media users look for in a post. Predictive analysis was performed using multivariate linear regression, MARS and SVR using an RBF kernel. MARS improves the R^2 score by 12.7% over Multivariate Linear Regression, and SVR further improves it by 14.2%. Multivariate Linear Regression chooses the number of reactions received by a post to be the most predictive feature, while MARS takes the sentiment of the post into account and chooses the number of "sad" reactions received by a post per second to be the most predictive feature.

I. Introduction

Content on social media is shared in many different forms, such as status updates, tweets, images, posts and pages. Some shared content becomes more popular and reaches a larger audience than some other content. The popularity of this content can be measured by the number of likes, shares and comments it receives. There can also be different factors that influence the popularity of content, such as the number of followers or friends the creator of the content has, how active those followers are, the number of followers a page the content is posted to has, the day a post is created on, the time it is created at and its sentiment.

The popularity of content posted on social media has been widely studied. In [1], hierarchical regression is used to study the categories of antecedents that can be related to a Facebook post's popularity. The popularity of news posts on Twitter is analyzed and an exponential regression model is used to predict this popularity in [2]. The sentiment of posted content has also been found to affect its popularity and reach. The rate of information diffusion is higher for content with negative sentiment [3]. Sentiment features extracted from images posted on social media are used to predict the popularity of the image in [4].

This paper studies the factors that determine the popularity of news posts on Facebook. The distributions of various predictor variables are first studied. Following this, the relationships between the predictor variables and outcome are studied. Then, three predictive models (multivariate linear regression, MARS and SVR) are fit to the data. Their performance and the importance they assign to features are used to further analyze what factors determine the popularity of posts and in what manner.

By indicating what posts should contain and where they should be posted in order to gain more popularity, this study can aid in the creation of posts that reach a wider audience. By indicating what a user looks for in a post before they share or "like" it, these insights can also aid in studies of the psychology of social media users.

II. Data

The Facebook News Dataset [5] is employed in this study. It contains 19,850 Facebook posts from 83 different news organizations and personalities representing up to the last 250 page posts made as of July 14th, 2017. Each post has up to 100 comments for a total of 1,025,403

comments.

Each post is represented by the following features in the dataset: post creation time, post scrape time, description, link, contents of the post, page ID (of the page the post is on), post ID, number of "angry", "haha", "like", "love", "sad" and "wow" reactions the post had at the time it was scraped and the number of shares the post had.

Additional features of interest were derived from these features. The post creation time was used to determine the day of the week each post was created, which was one-hot encoded to create seven features. The post creation time was also used to determine the segment of the day each post was created. This created four additional one-hot encoded features that indicate if a post was created between 1 AM and 7 AM, 7 AM and 10 AM, 10 AM and 5 AM or 5 AM and 1 AM.

The duration for which a post existed before being scraped was calculated using the difference between the post creation time and post scrape time. This was then used to calculate the average number of shares and average number of reactions (of each type) the post received per second, which resulted in a total of seven additional features. The fraction of reactions that were a certain type ("angry", "haha", "like", "love", "sad" or "wow") on each post were also calculated and this resulted in six additional features.

The Python VADER library was used to perform sentiment analysis on each post, and this resulted in four additional features - the sentiment, positivity, negativity and neutrality of each post.

Each comment is represented by the following features in the dataset: the parent post ID, the comment creation time, the name and ID of the user who created the comment, and the contents of the comment. The comments on each post were sorted by their creation time. Using the post creation time and the creation time of the 100^{th} comment, an additional feature, the time it takes for a post to get 100 comments, was derived for each post. Similarly, the time it takes for each post to get its first comment was also derived as an additional feature. All existing and derived features are summarized in Table 1.

The time it takes for a post to get 100 comments was the feature chosen to represent the popularity of a post. A post that gets 100 comments in fewer minutes than another post is more popular because it means it reached a wider audience in a shorter span of time.

Table 1: *List of existing and derived features*

Existing Features (for each post)
Creation time
Scrape time
Description
Link
Contents
Page ID
Post ID
Number of "angry", "haha", "like", "love", "sad" and "wow" reactions (6 separate features)
Number of shares
Existing Features (for each comment)
Parent post ID
Creation time
Name of the user who created the comment
Contents
Derived features (for each post)
Day of the week (one-hot encoded)
Time of the day (one-hot encoded)
Average number of "angry", "haha", "like", "love", "sad" and "wow" reactions per second (6 separate features)
Average number of shares per second
Fraction of "angry", "haha", "like", "love", "sad" and "wow" reactions (6 separate features)
Sentiment
Positivity
Negativity
Neutrality

III. Methods

First, the distributions of and relationships between features were studied. The number of reactions (of each type) received by a post per second had long-tailed distributions. Log-scaling them resulted in normal distributions. The number of minutes until a post gets its first and 100^{th} comment, the number of times it is shared per second and the total number of reactions it gets per second had heavily skewed distributions and needed a more powerful transform [6] to make them normally distributed. The reciprocal root transform was chosen, with a different fractional power for each feature.

To study the relationship between post sentiment and some other features, posts were binned by sentiment (bins of width 0.2) and the average value of the feature in question was calculated, over all posts in that bin.

Second, each of the transformed features was plotted against the (transformed) number of minutes until a post gets its 100^{th} comment to determine if there exists a relationship between them. These scatter plots indicated linear relationships. The mean number of minutes until a post gets its 100^{th} comment for each day and time segment, plotted against all the days and time segments respectively indicated linear and bimodal relationships respectively.

Univariate linear regression analyses were then conducted on pairs of features mentioned above, to further examine relationships between them. This was followed by multivariate linear regression analysis over 100 iterations. Each iteration began with a multivariate linear regression model being trained on just one feature. Features were added one at a time until the model was trained on all features. At each step, i.e. for each set of features in an iteration, the model was evaluated based on its R^2 score and mean squared error.

Out of a total of thirteen features, the six features that led to the largest increase in the R^2 score in each iteration were noted. This was done over all 100 iterations. In each iteration, the order in which features were added to the model, was randomized. The six features that occurred the most number of times in the top six of each iteration, over all 100 iterations, were chosen as the most significant features.

Following this, Multivariate Adaptive Regression Spline (MARS) analyses and Support Vector Regression (SVR) analyses were carried out using the most significant features (chosen previously). Hyper-parameters were tuned using 20% of the dataset.

IV. Results

Considering all reactions across all posts, "likes" occur most frequently. They constitute 62.9% of all reactions, followed by "angry" reactions which constitute 10.8%. This is shown in Figure 1.

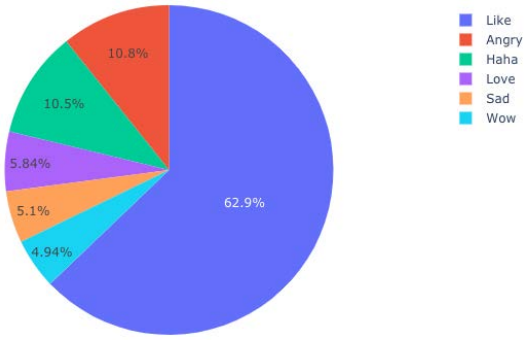
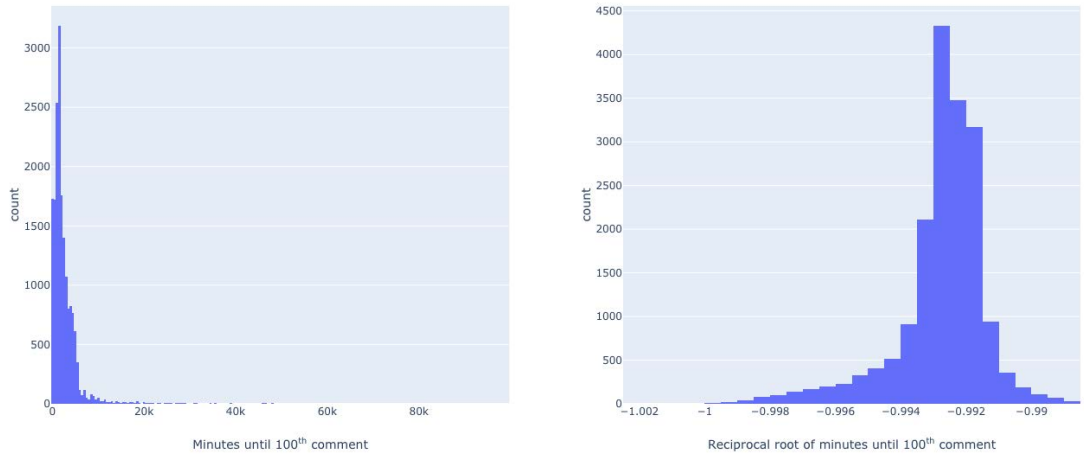


Figure 1: *Percentage of each type of reaction in the dataset.*

The number of minutes it takes for a post to get 100 comments appears to have a heavily skewed distribution with a long tail, as shown in Figure 2 (a). 80.9% of the posts get their first 100 comments within 67 hours. After applying the reciprocal root transform, the distribution appears to be normal. This is shown in Figure 2 (b). The distributions of other features are also similarly skewed.

In Figure 3, the rows indicating the fraction of "like", "angry" and "haha" reactions for each post have a higher number of brightly colored cells as compared to the rows indicating the fraction of "love", "sad" and "wow" reactions. The majority of reactions on each

post are therefore "like", "angry" and "haha" reactions. This is in agreement with Figure 1, which indicates that "love", "sad" and "wow" reactions occur less frequently across the entire dataset.



(a) **(b)**

Figure 2: *Distribution of the minutes it takes for a post to get 100 comments (a) before transformation (b) after transformation*

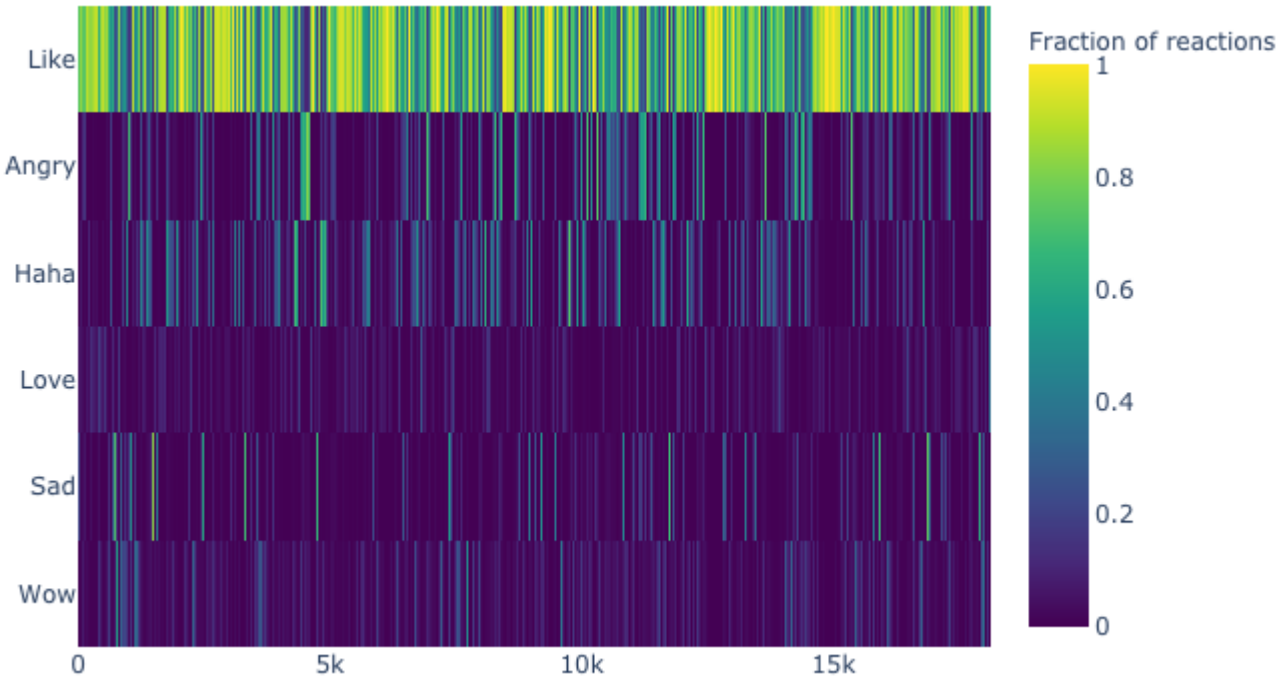


Figure 3: *Fraction of each type of reaction (y-axis) for each post in the dataset (x-axis)*

The fraction of each type of reaction on a post indicates the sentiment of a post. As shown in Table 2, posts tagged with positive sentiment (>= 0.5) by VADER have a smaller fraction of "angry" and "sad" reactions, and a larger fraction of "love" reactions than posts tagged with negative sentiment (<=-0.5).

Table 2: *Relationship between sentiment of posts and the average fraction (over all posts) of reactions of each type*

VADER Sentiment	**"angry" and "sad" fraction**	**"love" fraction**	**"haha" fraction**
>=0.5	0.107	0.052	0.079
<=-0.5	0.233	0.027	0.073

Figure 4 shows that the number of "sad" and "angry" reactions per second reduce, and the number of "love" reactions per second increase, for posts with more positive sentiment. "Wow" reactions seem to indicate negative sentiment since the rate at which they are obtained reduces as posts become more positive. Similarly, "haha" reactions seem to indicate positive sentiment.

Figure 5 shows box plots of the minutes until a post gets it's first and it's 100^{th} comment, for sentiment bins of width 0.1. The maximum number of minutes until both the first and 100^{th} comment for each bin appear to increase for bins with more positive sentiment. The median number of minutes until both the first and 100^{th} comment for each bin also appear to increase for bins with more positive sentiment, but not as much as the maximum. Posts therefore appear to be more popular if they are associated with more negative sentiment. Posts which get their first comment later than 6 minutes and their 100^{th} later than 4000 minutes (67 hours) (i.e. 20% of the posts) contribute more to this trend than the other 80% of posts.

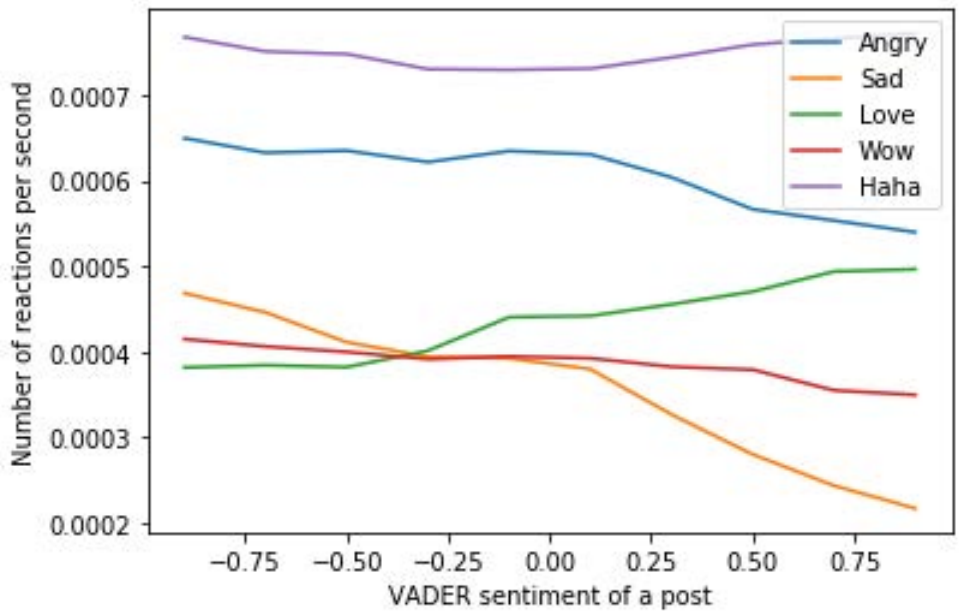


Figure 4: *Variation of the number of reactions of each type obtained per second, with the VADER sentiment of a post*

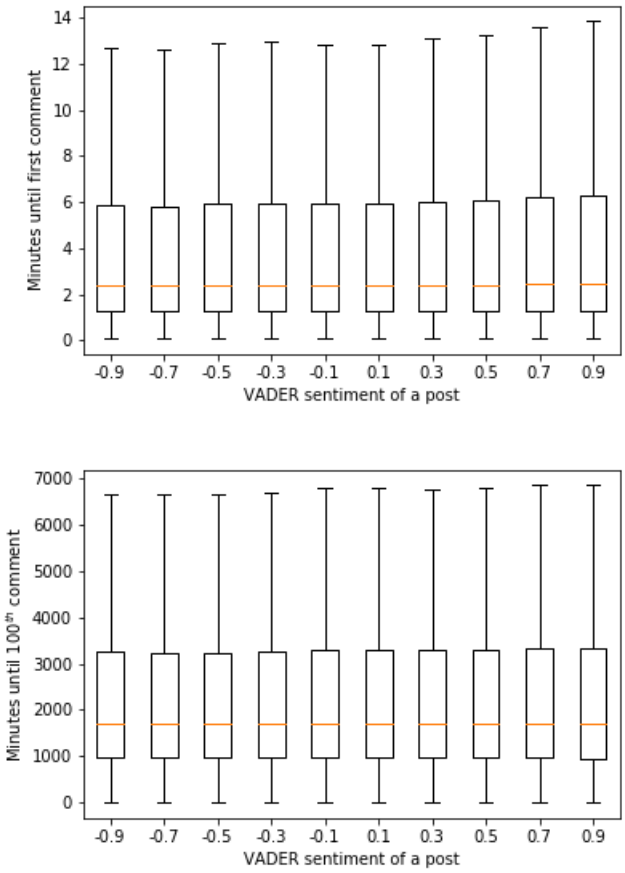
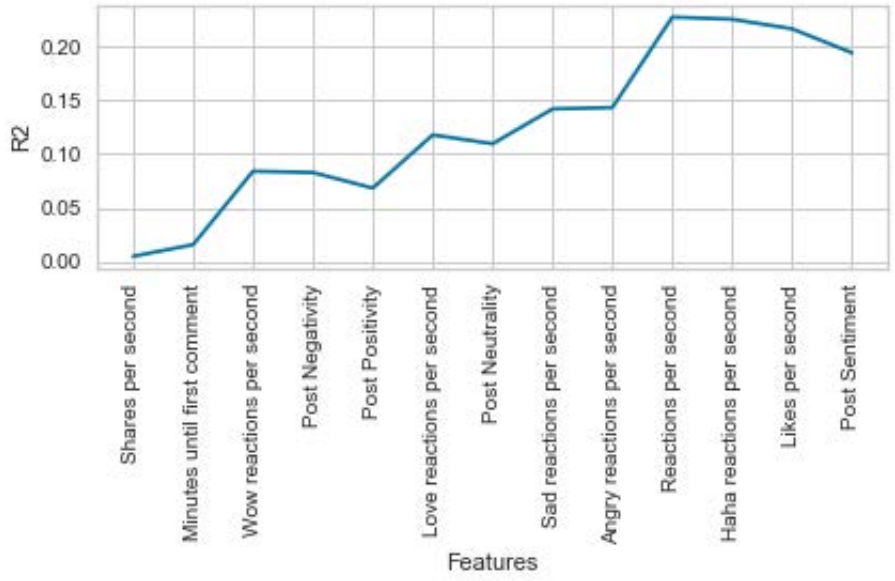


Figure 5: *Variation of the number of the number of minutes until the first and 100th comment, with the VADER sentiment of a post*

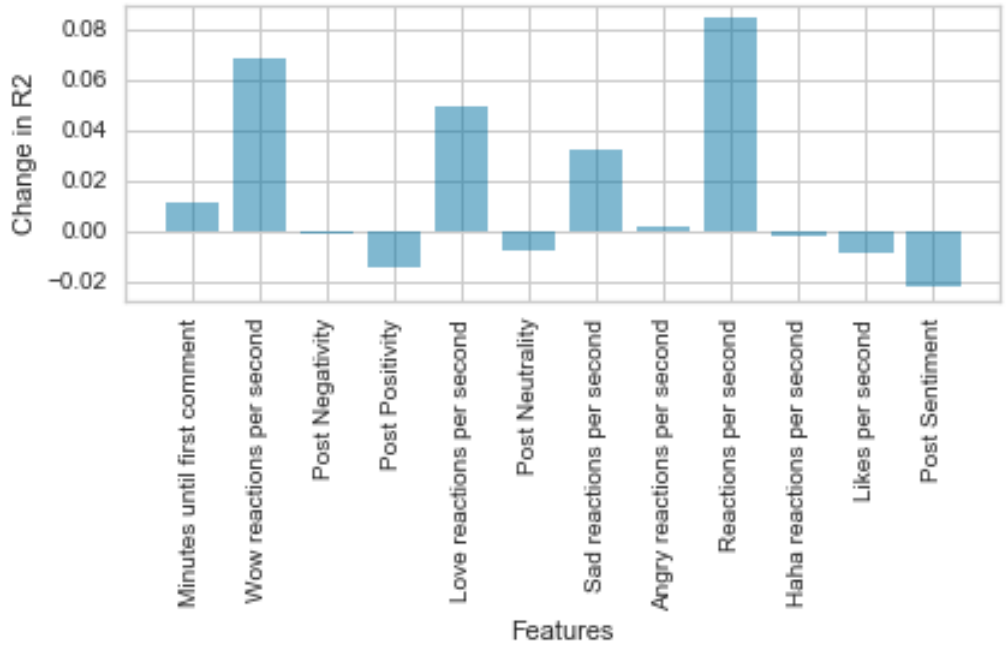
Figures 6 (a) and 7 (a) show the variation in the R^2 score and RMSE of the multivariate linear regression model (in one of the hundred iterations) as each feature is added. The R^2 score sees an overall increase and the RMSE sees an overall decrease as features are added. This indicates the model performs better as more features are added.

Some features increase the R^2 score more than the others (and decrease the RMSE more than the others), as shown in Figures 6 (b) and 7 (b). It is interesting to note that the feature that leads to the largest increase in the R^2 score (the number of "sad" reactions a post receives per second) is not the same as the features that decreases the RMSE the most (the number of "love" reactions a post receives per second).

Across all 100 iterations, the six features which increase the R^2 score by the largest magnitude, the maximum number of times, are the log-scaled number of "love", "sad", "like" and "wow" reactions received by a post per second, the reciprocal root transformed number of reactions per second, and minutes until a post gets its first comment.

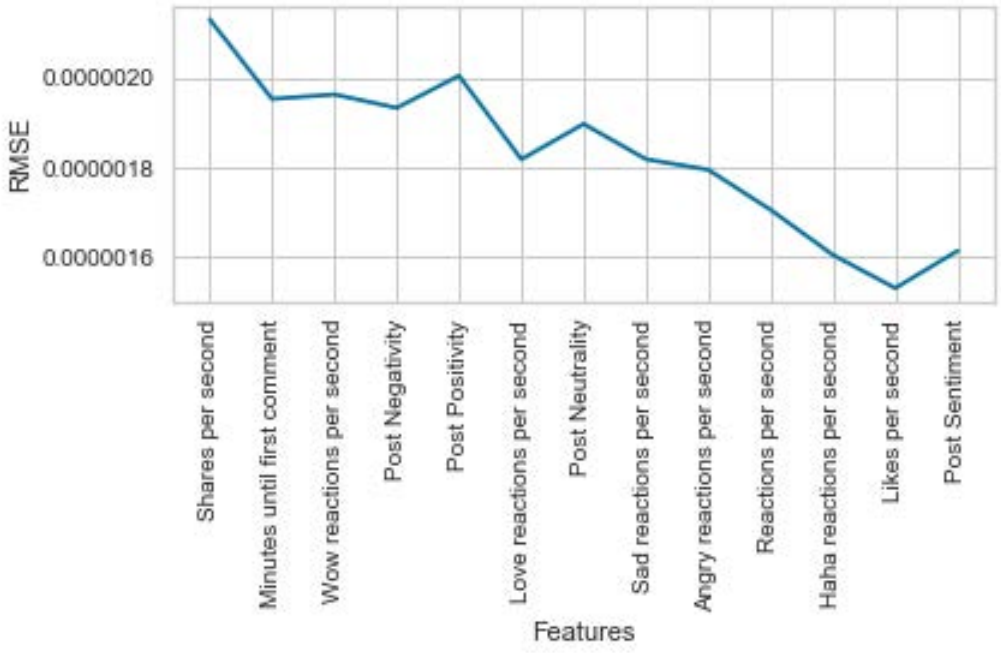


(a)

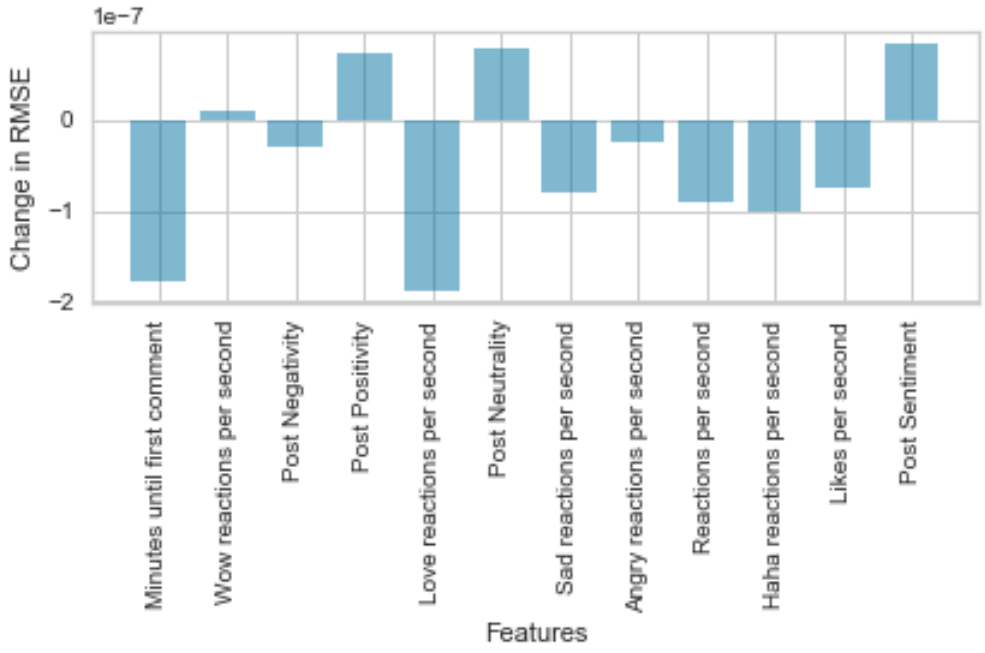


(b)

Figure 6: *(a) Variation in the R^2 score of the multivariate linear regression model as each feature is added (b) Magnitude of change in the R^2 due to each feature*



(a)



(b)

Figure 7: *(a) Variation in the RMSE of the multivariate linear regression model as each feature is added (b) Magnitude of change in the RMSE due to each feature*

Table 3: *Multivariate linear regression coefficients and p-values for one iteration*

Feature	Coefficient (in reciprocal root transformed minutes)	P-value
Likes per second	-0.0010	$\ll 0.001$
Post negativity	-0.0176	0.701
Reactions per second	0.2091	$\ll 0.001$
Shares per second	0.0157	$\ll 0.001$
Minutes until first comment	-0.0009	$\ll 0.001$
"Wow" reactions per second	-0.0004	$\ll 0.001$
"Angry" reactions per second	-0.0002	$\ll 0.001$
"Love" reactions per second	-0.0008	$\ll 0.001$
"Haha" reactions per second	-0.0003	$\ll 0.001$
Post positivity	-0.0180	0.695
"Sad" reactions per second	-0.0005	$\ll 0.001$
Post neutrality	-0.0177	0.700
Post sentiment	4.58×10^{-5}	0.297

Table 3 shows the coefficients and the p-values for multivariate linear regression for one iteration. The features with p-values $\gg \alpha$ ($\alpha = 0.05$) are post negativity, positivity, neutrality and sentiment. These are the same features that lead to a decrease in the R^2 score and an increase in the RMSE (Figures 6 and 7).

Figure 8 shows the log-scaled absolute values of the same coefficients. Features with significant coefficients don't always lead to an increase in the R^2 score (for example, the post positivity) and features with less significant and negative coefficients do lead to an increase in the R^2 score (for example, minutes until the first comment and the number of "wow" reactions per second).

For SVR, a grid search was performed over three kernels (linear, RBF and polynomial), four values of C (0.1, 1, 100, 1000), eleven values of ϵ (0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10) and seven values of γ (0.0001, 0.001, 0.005, 0.1, 1, 3, 5). The RBF kernel with values $C = 0.1$, $\gamma = 0.1$ and $\epsilon = 0.0001$ performed the best.

A comparison of the performance of the three chosen models (Table 4) indicates that MARS improves the R^2 score by 12.7% over Multivariate Linear Regression, and SVR further improves it by 14.2%. MARS reduces the RMSE by 5.4%.

Table 4: *Performance of regression models*

Model	R^2 Score	RMSE
Multivariate Linear Regression	0.212	1.65×10^{-6}
MARS	0.239	1.56×10^{-6}
SVR	0.273	1.58×10^{-6}

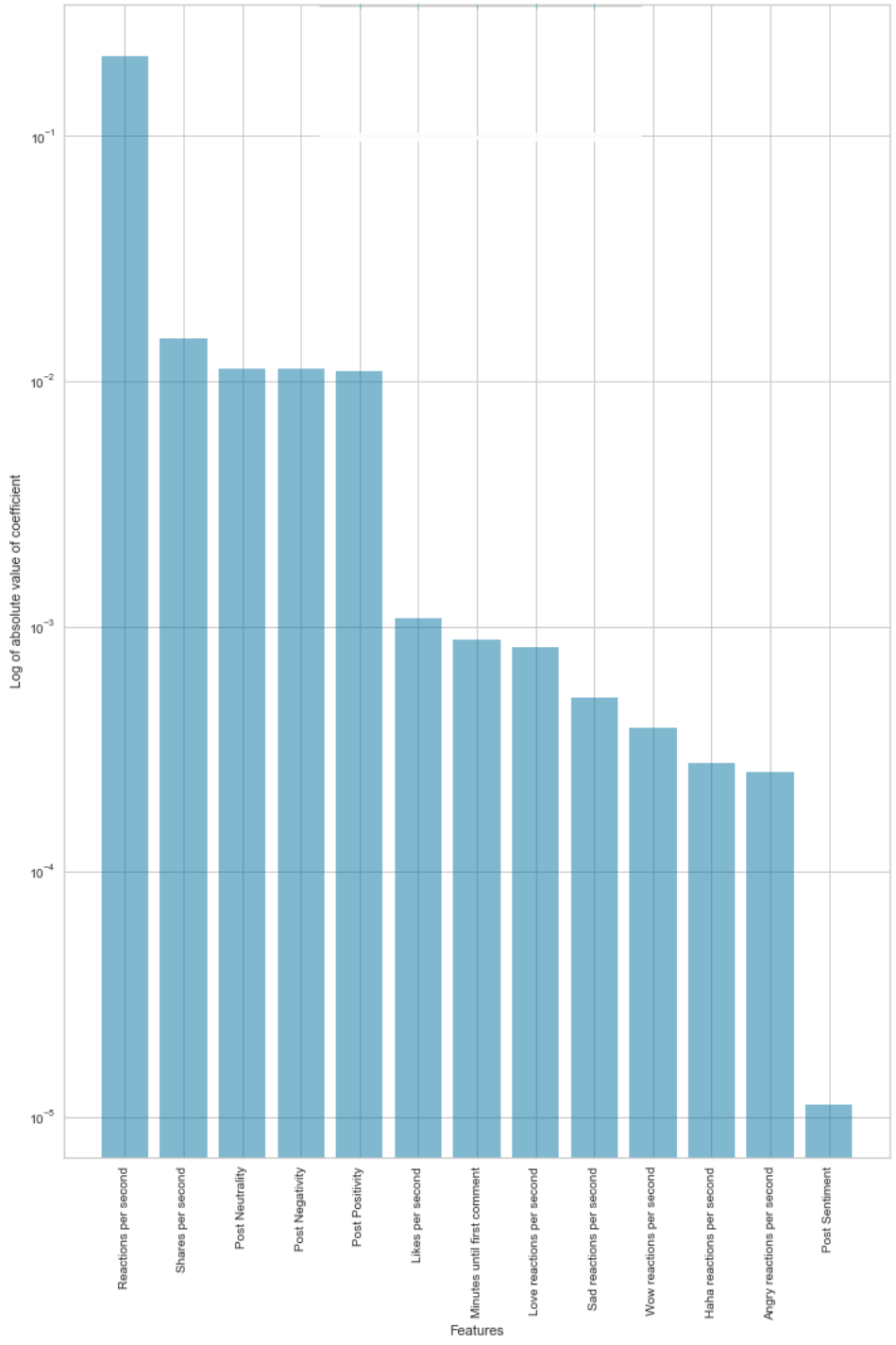
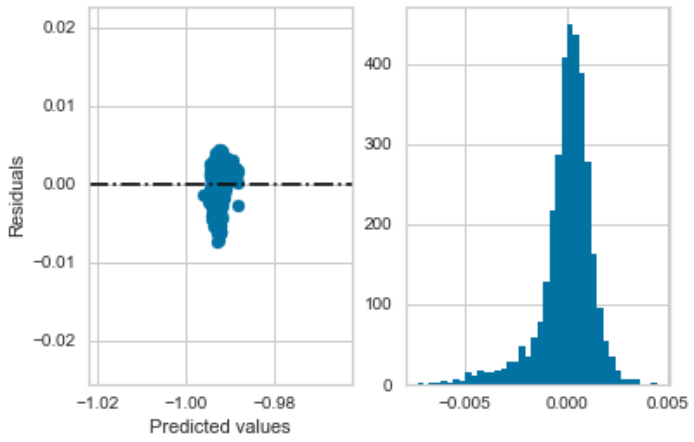
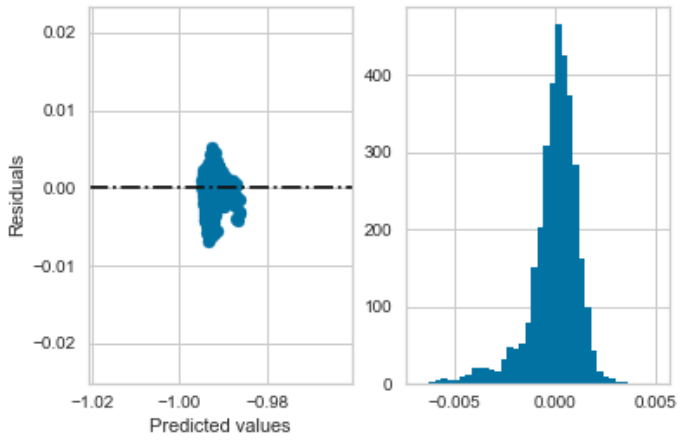


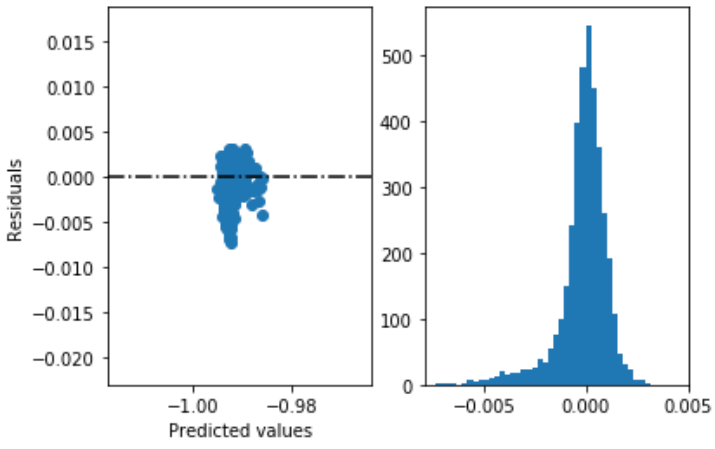
Figure 8: *Log-scaled absolute values of coefficients for multivariate linear regression (one iteration)*



(a)



(b)



(c)

Figure 9: *Residuals and their distribution for each of the three models (a) Multivariate linear regression (b) MARS (c) SVR*

Figure 9 shows that the distributions of residuals become progressively narrower and more heavy tailed. This could indicate that multivariate linear regression gives the outliers just as much importance as the other observations, thus leading to more data points having larger residuals (a wider distribution) and the outliers having smaller residuals (a distribution with no long tails). MARS and SVR, on the other hand, give the outliers less importance. The non-outliers therefore fit better (smaller residuals, narrower distribution) but the outliers have large residuals, which leads to the heavy tails in the distribution.

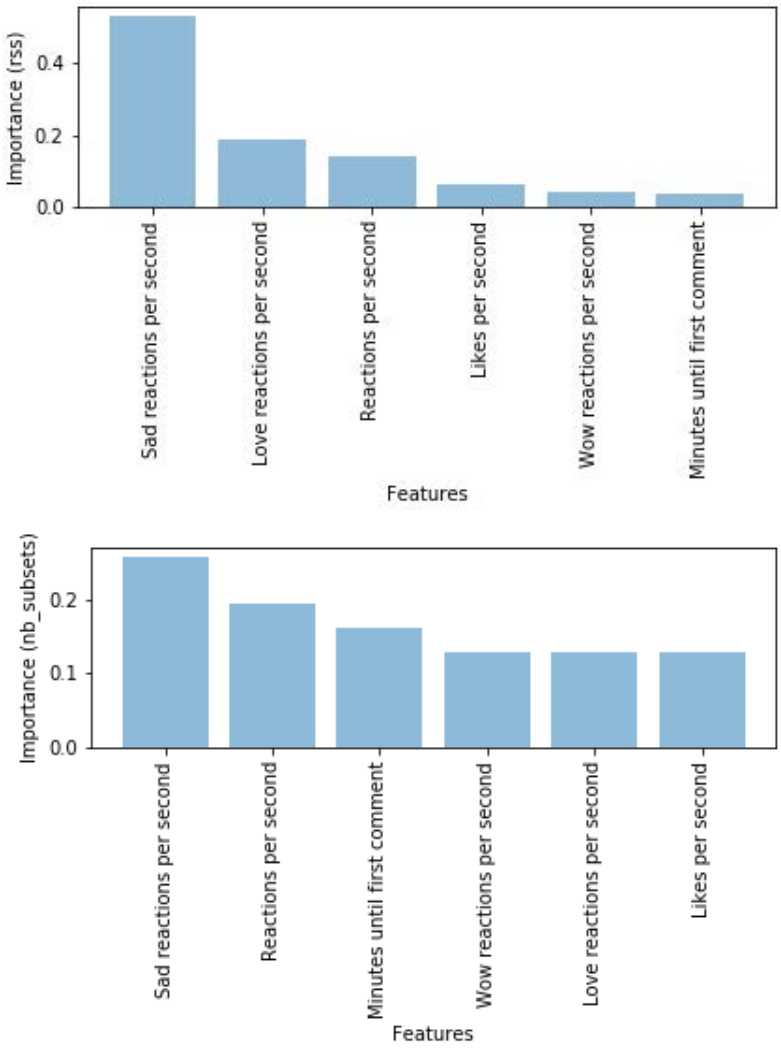


Figure 10: *Features that MARS chose to be the most predictive with RSS and number of model subsets as measures of importance respectively*

For the MARS model, feature importance can be measured either by the residual sum of squares (RSS) or the number of model subsets that include the feature [7]. Figure 10 shows that the number of "sad" reactions received by a post per second is chosen to be the most predictive feature, using both RSS and number of subsets as the criteria.

On the other hand, the multivariate linear regression model chose the number of (all types of) reactions to be the most predictive, thereby disregarding the sentiment of the post in determining its popularity (Figure 8). The total number of reactions a post receives per second could be a measure of the number of followers a news page has and the number of friends each of those followers has. The greater these two are, the greater the number of feeds on which the post shows up, which leads to more people reacting to the post. The multivariate linear regression model therefore seems to use the popularity of a news page and its followers to determine the popularity of a post.

The MARS model assigns small weights to reactions per second, likes per second and minutes until the first comment, all of which do not take the sentiment of a post into account. The number of "sad" reactions per second takes both of the page popularity and post sentiment into account. The MARS model therefore appears to determine the popularity of a post not only by the popularity of the news page the post is on and the popularity of its followers, but also by the sentiment of the post.

It is also interesting that the MARS model considers the number of "sad" reactions to be more predictive than the number of "love" reactions per second to predict popularity. This could mean people react to posts with negative sentiment but don't react to posts with positive sentiment. Positive sentiment is then indicated by the absence of negative reactions, rather than by the presence of positive ones. It is also interesting to note that the multivariate linear regression coefficients with largest magnitude include the number of reactions per second, the positivity, negativity and the neutrality of a post, i.e. its sentiment (Figure 8). By choosing the number of "sad" reactions per second to be the most predictive, MARS, in some sense, chooses a feature that is a combination of the most predictive features of the linear regression model.

V. Discussion and Conclusions

This study assessed the effects of various features such as the number and type of reactions received by a post per second, the sentiment of the post, etc. on the popularity of the post. The sentiment of a post was found to influence the popularity of a post more than the popularity of the page the post was created on or the popularity of its followers. In line with previous research, posts with negative sentiment were found to be more popular.

The feature importances assigned by the RBF SVR kernel could not be calculated as it transforms the features to a more complex space and then assigns weights. A possible extension to this study would be to determine what features the RBF kernel chose to be the most predictive. In addition to this, the six features that lead to the most increase in the R^2 score of the multivariate linear regression model were used to train the MARS and SVR models. Choosing the features with the highest values of multivariate linear regression coefficients could lead to different results and provide different insights into the popularity of posts.

References

[1] Banerjee, S. and Chua, A.Y.K. J Brand Manag (2019) 26: 621. *https://doi-org.colorado.idm.oclc.org/10.1057/s41262-019-00157-7*.

[2] Bo Wu, Haiying Shen, (2015). Analyzing and predicting news popularity on Twitter, International Journal of Information Management. *https://doi.org/10.1016/j.ijinfomgt.2015.07.003*..

[3] Ferrara E, Yang Z. 2015. Quantifying the effect of sentiment on information diffusion in social media. PeerJ Computer Science 1:e26 *https://doi.org/10.7717/peerj-cs.26*.

[4] Francesco Gelli, Tiberio Uricchio, Marco Bertini, Alberto Del Bimbo, and Shih-Fu Chang. 2015. Image Popularity Pre-

diction in Social Media Using Sentiment and Context Features. In Proceedings of the 23rd ACM international conference on Multimedia (MM '15). ACM, New York, NY, USA, 907-910. *DOI: https://doi.org/10.1145/2733373.2806361*

[5] John Bencina, 2017. *Facebook News Scraper https://github.com/jbencina/facebook-news*.

[6] James Kirchner, 2001. *Data Analysis Toolkit #3: Tools for Transforming Data http://seismo.berkeley.edu/kirchner/eps_120 /Toolkits/Toolkit_03.pdf*.

[7] Stephen Milborrow, 2019. *Notes on the earth package http://www.milbo.org/doc/earth-notes.pdf*.

Opinion Fraud Detection in Amazon Reviews

Lakshya Sharma
University of Colorado Boulder
lakshya.sharma@colorado.edu

Abstract

Product reviews are an integral component of E-Commerce websites. Users make a decision on whether to go through with a purchase based on product reviews. Opinion spammers often target review systems in order to cause a misrepresentation of product qualities by posting fraudulent reviews. Review text, rating, metadata such as review time, helpfulness of review are all important features that can be used to identify deceitful reviews. This study aims to classify such "fake" reviews by using unsupervised machine learning approaches.

I. Introduction

Amazon is one of the biggest E-Commerce websites. Amazon holds a massive 49% market share and accounts for 5% of all retail sales in USA [Tech Crunch 2018]. In today's world, people rely more on ordering products from Amazon rather than physically going to the store and buying. In February 2019, non-store retail sales accounted for 11.813 percent of the total, compared with 11.807 percent for general merchandise [CNBC 2019]. The purchases can vary from small day to day items to extravagant expensive items. Due to a physical disconnect from the product, buyers can scrutinize the product images, description etc, but tend to rely heavily on what previous buyers' experience has been. This information is typically found in the form of product ratings and product reviews. Good product reviews can drive the success of a product's sale and consequently impact a business' profit. Large proportion of positive reviews can attract more customers and proliferate financial gains whereas if negative reviews for a product in abundance can defame products and cause the sales to plummet. Due to the reputation and financial incentive, imposters may be hired to deliberately write fake or deceptive reviews to promote or demote the reputation for their target products or services. Such imposters are called review or opinion spammers and their review texts are called review spams.

With the rise of e-commerce websites, opinion spammers are on the rise. A review spam is a form of review text aimed at misrepresenting the nature of interaction the reviewer has had with a business to tarnish the reputation of the business. Spam reviews can be categorized in three types. Type 1, which are deliberately posted to mislead readers by giving overly positive reviews to promote objects or malicious negative reviews to damage their reputation. Type 2, are reviews which are not targeted at the particular product but only at the brands, sellers and manufacturers. Type 3, are reviews that are not reviews at all and contain random texts, questions, advertisements etc [Jindal 2008].

The goal of this study is to use unsupervised machine learning techniques to identify such fraudulent reviews. Rating and review text are the primary features used to do this analysis. Natural Language Processing techniques are used to clean and featurize the review text to perform sentiment analysis and find outliers by using the review text semantics and sentiment. Metadata like helpfulness of a review, review time, reviews by the same reviewer on several products are primarily use to check for anomalous patterns opinion spammers employ to target multiple reviews at once. After getting sentiment scores for each review

the data is fed through an isolation forest from sklearn which returns an anomaly score for each sample. The predictions are then manually observed to see if they show a fraudulent review pattern.

II. Related Work

Previously supervised learning models have been employed to perform spam detection. Researches from [Jindal 2008] have used the Amazon dataset and labelled the reviews manually. They first detect duplicates and near duplicates and label them as spam reviews of Type 1. For type 2 and type 3 type reviews they have tagged the reviews manually. They used logistic regression for classifying the reviews as "spam" or "not spam".

There have been unsupervised learning methods employed in detecting spam reviews. A "Unified Review Spamming Model" (URSM) [Xu et al, 2015] is proposed for detecting "suspicious" review spams, the review spammers and the manipulated offerings in an unsupervised manner. They explore the idea of exploiting text generality in addition to the existing features for improving spam detection.

Auto encoders have been recently used for doing anomaly detection even in Opinion Fraud Detection [Dong et al, 2018] which is an approach to consider. Isolation forests [Liu et al, 2018] have been used widely in anomaly detection which could also be explored. There is a FRAUDEAGLE framework [Akoglu et al, 2015] which has been tested on synthetic and SWM dataset and seemed to have performed adequately.

III. Data

The data set used for this project is Amazon Review Data [Ni et al, 2019]. This dataset consists of product reviews, which include ratings, text-reviews, helpfulness votes, as well as product metadata of products such as description, price, brand etc. It has a total of 233.1 million reviews ranging from May 1996 - October 2018. The reviews have been divided based on their category. For the purpose of this study the focus is on the "Books" category.

The data is formatted as one-review-per-line in json. Below is an example of a review. The json tags are self descriptive. The tag `"reviewText"` is an important feature in our dataset which can be used to identify how a fraudulent review is typically worded. Additionally, the field `"overall"` can be used to see if there is a discrepancy between the sentiment of the `"reviewText"` and the rating of the review. We can also use the field `"vote"` to identify how useful other users have found the review to be. An example record is provided below.


```
{
  "reviewerID": "A2SUAM1J3GNN3B",
  "asin": "0000013714",
  "reviewerName": "J. McDonald",
  "vote": 5,
  "style": {
    "Format:": "Hardcover"
  }
  "reviewText": "I bought this for
      my husband who plays the
     piano.  He is having a
     wonderful time playing these
      old hymns.  The music  is
     at times hard to read
     because we think the book
     was published for singing
     from more than playing from.
       Great purchase though!",
  "overall": 5.0,
  "summary": "Heavenly Highway
     Hymns",
  "unixReviewTime": 1252800000,
  "reviewTime": "09 13, 2009"
}
```


The data was pre-processed and filtered to test the methods on a smaller subset of data. Several methods were employed to filter the dataset for getting an appropriate distribution of data. Research by [Jindal 2008] shows the relationships between number of reviews to number of members on Amazon. A power law relationship exists between number of reviews

and the reviewers. An interesting observation is that large number of reviewers write only a few reviews and a few reviewers use write a large number of reviews. We grouped the reviews by `reviewerID` and sorted by descending order to only consider reviewers that post several reviews. We assume that opinion spammers will not write one off reviews and will look to write several reviews to target products.

IV. Methods

A major chunk of this study focuses on exploratory data analysis. A histogram of ratings can be plotted on a per product basis to see the distribution of the ratings for a product. An example for histogram of ratings for a product is as provided in figure 1. We can see that the distribution of product ratings can be a factor in understanding fraud review distribution. We see that primarily reviews have a high rating. Therefore, solely focussing on rating is not a good approach to classify.

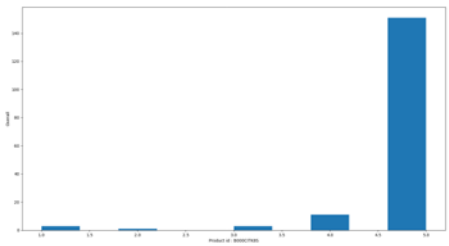


Figure 1: *Histogram of ratings for a product*

This leads us into utilizing the `reviewText` to featurize any insights from the way fraud reviews are worded. We performed sentiment analysis on the review text. Before getting the sentiment score for each review text cleaning operations are performed.

- Convert text to lowercase : Convert all the characters to lowercase.
- Tokenize : Extract individual words by ignoring whitespaces and punctuations.
- Remove stop words like a, an, the etc and other less useful words which contain numbers.
- Part-Of-Speech tagging : That is marking up the words as corresponding to a particular part of speech, based on both its definition and its context—i.e., its relationship with adjacent and related words in a phrase, sentence, or paragraph.
- Lemmatization : Grouping together the inflected forms of a word so they can be analysed as a single item, identified by the word's lemma, or dictionary form, eg : Ask, Asking, Asked all will be lemmatized to ask.

`SentimentIntensityAnalyzer` from `nltk` was used to get a positive, negative and neutrality score for each review text. The histogram of features is as shown in figure 2 and the correlation matrix between features is as shown in figure 3. We can see that there is high correlation between positive and compound scores. This also suggests that reviews tend to be positively worded in general. The Compound score is a metric that calculates the sum of all the lexicon ratings which have been normalized between -1(most extreme negative) and +1 (most extreme positive). We see that the distribution of compound also skews towards a positive score.

The combined sentiment score and the overall rating of the product was used as features to do a baseline k-means clustering. We hope that reviews similar to each other will cluster together. Therefore, we expect spam reviews to have similar underlying features to cluster together into one cluster and similarly fraudulent reviews to cluster into another cluster. The number of clusters(k) was defined as 4. This number was chosen based on the expectation that reviews will fall under one of the 4 categories : Non-spam, Type 1 spam, Type 2 spam and Type 3 spam. The results of a k = 4 can be seen as below. We notice that, k-means clustering gives a lot of weight to rating and hence is not a good method to identify spams.

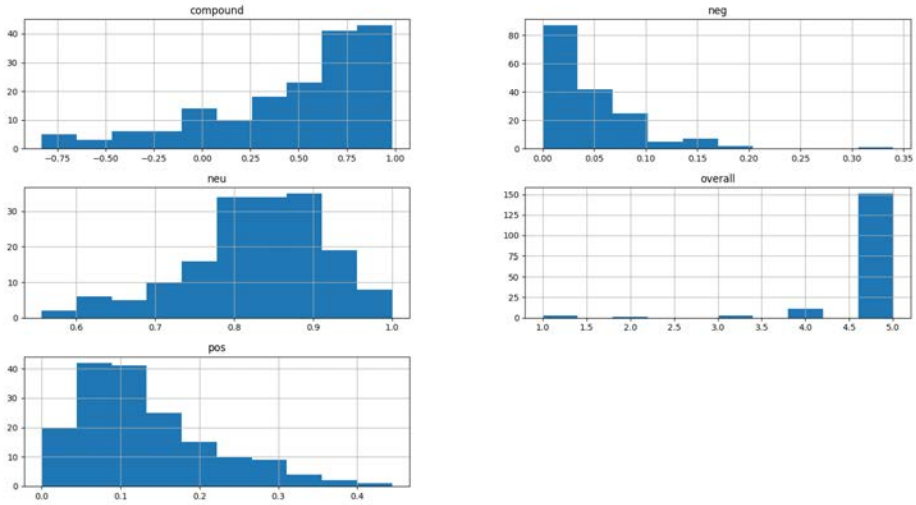


Figure 2: *Histogram of sentiment scores*

Figure 3: *Correlation matrix of features*

An approach to filter out more reviews is based on the pattern in which users post reviews. We plot the `unixReviewTime` of the review for each reviewer and observe that certain reviews for different products are posted at exactly the same timestamp by the same user. Figure 4 shows this behavior. It seems like this could be a key motivating factor in determining fraudulent reviews. A user posting reviews for multiple products typically would not post reviews within a few minutes of posting another review let alone post at the exact same timestamp. We proceed by finding reviews posted within a time window of two minutes from the previous review. With this method of pre-processing we hope to get a more even distribution of spam and non spam reviews in our data.

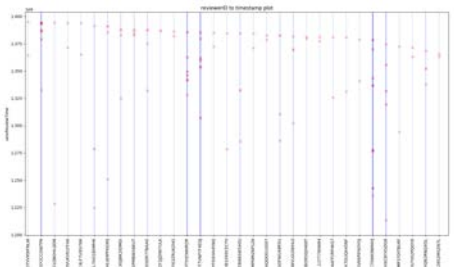


Figure 4: *reviewerID to unixReviewTime plot*

We look at another predominantly used anomaly detection technique, Isolation Forests. Isolation Forests employ a different approach than trivial clustering and anomaly detection algorithms. Previous approaches would build a profile of a "normal" sample and then go on to identify samples that do not match this profile as an abnormal behavior and classify it as an anomaly. Isolation Forests fundamentally do the opposite of this. They explicitly try to isolate anomalies instead of profiling "normal" behavior. The primary goal of an isolation forest is to find samples that are "few and different", hence making them more susceptible to isolation than "normal" samples [Liu et al, 2008].

Isolation forest builds an ensemble of iTrees for a given data set, then anomalies are those instances which have short average path lengths on the iTrees. Figure 5 shows a representation of how random partitioning of an anomaly differs from a normal point. The anomaly score for a sample for Isolation Forest is defined as

$$s(x,n) = 2^{\frac{-E(h(x))}{c(n)}}$$

where $h(x)$ is the path length of the observation x and $c(n)$ is the average path length of unsuccessful search in a Binary Search Tree and n is the number of external nodes.

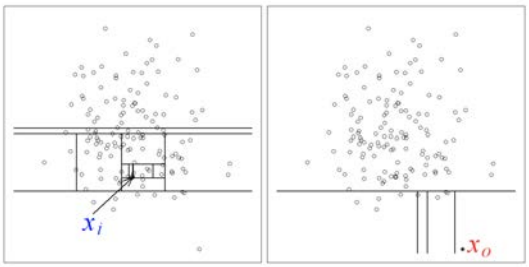


Figure 5: *Isolation forest, normal vs abnormal observations*

V. Results

The results of using sklearn's Isolation Forest on the cleaned and filtered dataset is as shown. Since we do not have labelled data, the results are inspected manually. We notice that the reviews classified by our model are not necessarily spam and the model performance is low. Additional ethods are discussed in the next section to make the model more robust. Nevertheless, our model does identify some interesting patterns. Our model identified reviews from the categories as shown below.

The snippet of review below shows that this review is a culmination of type 1 and type 3 reviews. It is abnormally negative and doesn't really talk much about the book.


```
{
"reviewerID": "A279FA8OYQQRP9",
"asin": "0310331935",
"helpful": [
  0,
```

```
    0
  ],
  "reviewText": "After eating the
     chicken and spitting out the
      bones the chicken that I
     take away from this book is
     to grow in your sacrificial
     love and devotion to Christ.
      That said I see a lot of
     bones in this book to be
     rejected. It seems like this
      books ideas may have an
     underlying Calvinist
     theology. I have noticed
     that Calvinists have a
     backwards belief regarding
     losing your salvation
     compared to the Armenian
     church of Christ
     denomination I grew up in.
     The man who does not work
     but trusts in him who saves
     the ungodly, that mans faith
      is credited as
     righteousness. The one who
     relys on works will not be
     counted righteous.",
  "overall": 2,
  "unixReviewTime": 1399420800,
  "compound": 0.9988,
  "anomaly": -1
}
```


Similarly, the next review is overly negative without giving much information on why they disliked the book.


```
{
  "reviewerID": "A3DKM0STAF66Z5",
  "asin": "310344298",
  "helpful": [
    1,
    11
  ],
  "reviewText": "I read the first
     chapter and lost 5 lbs!
     Didn't do anything. Didn't
     change the way I ate or
     moved. It's not necessary!
     God wants you slim and it
     will happen. After all I
     can do all things.... Read
     the rest of the book and
     gained 15 lbs... The first 5
      lbs was my wallet getting
     lighter.Kudo's to those who
     made this work to loose
     weight. You have my utmost
     respect. It's amazing that
     anyone could stick to this
     diet let alone maintain it
     as a life change.",
  "overall": 1,
  "unixReviewTime": 1397260800,
  "compound": 0.9025,
  "anomaly": -1
}
```


One of the most interesting observations was that we found examples of several reviews that were posted by the same reviewer for different products at the exact same timestamp. More interestingly, the reviews were quite similar and had no relevance to the book. They seemed to be slightly different in which they were phrased but had the same review content which had no relation to the product. This verifies that reviews posted at the exact same timestamp do tend to be anomalous.

VI. Discussion and Conclusions

There is a lot of scope in opinion spam detection. For the scope of this research, we only utilized review timestamp, sentiment score and rating of a product for predicting fraudulent reviews. We realized that a lot of the reviews classified as fraudulent had very common words in their vocabulary. This motivated the thought process of utilizing text data within the review text to make the model more robust. The text can be featurized further by using word embeddings to find a pattern in the way fraudulent reviews are worded.

We noticed that when opinion spammers post similar reviews for different products, they are not exactly the same text. The review texts are worded and phrased slightly different to

avoid looking like duplicates, however they still contain the same content. Variational auto encoders can be utilized to deal with this problem. Variational Autoencoders (VAEs) have one fundamentally unique property that separates them from vanilla autoencoders, and it is this property that makes them so useful for generative modeling: their latent spaces are, by design, continuous, allowing easy random sampling and interpolation [TDS 2018]. This makes them especially useful for training on data that has slight differences from the input. One can tweak the input slightly and the encoder-decoder learns to identify these as the same samples.

Finally, this research was a step towards exploring unsupervised learning approaches to do anomaly detection in Amazon reviews to find fraudulent reviews. This study shows a significant correlation between the posting patterns of opinion spammers such as posting several reviews at the same time stamp and posting similar reviews for different products.

References

[Tech Crunch 2018] Amazon's share of the US e-commerce market is now 49% *https://techcrunch.com/2018/07/13/amazons-share-of-the-us-e-commerce-market-is-now-49-or-5-of-all-retail-spend/*

[CNBC 2019] Online shopping overtakes a major part of retail for the first time ever *https://www.cnbc.com/2019/04/02/online-shopping-officially-overtakes-brick-and-mortar-retail-for-the-first-time-ever.html*

[Jindal 2008] Nitin Jindal, Bing Liu : Opinion Spam and Analysis

[Ni et al, 2019] Jianmo Ni, Jiacheng Li, Julian McAuley : Justifying recommendations using distantly-labeled reviews and fined-grained aspects Empirical Methods in Natural Language Processing (EMNLP), 2019 *https://nijianmo.github.io/amazon/index.html*

[Dong et al, 2018] Manqing Dong, Lina Yao, Xianzhi Wang, Boualem Benatallah, Chaoran Huang, Xiaodong Ning : Opinion Fraud Detection via Neural Autoencoder Decision Forest

[Liu et al, 2018] Fei Tony Liu, Kai Ming Ting, Zhi-Hua Zhou : Isolation Forest

[Akoglu et al, 2015] Leman Akoglu, Rishi Chandy, Christos Faloutsos : Opinion Fraud Detection in Online Reviews by Network Effects

[Xu et al, 2015] Yinqing Xu, Bei Shi, Wentao Tian, Wai Lam : A Unified Model for Unsupervised Opinion Spamming Detection Incorporating Text Generality

[Liu et al, 2008] Fei Tony Liu, Kai Ming Ting, Zhi-Hua Zhou : Isolation Forest

[TDS 2018] Intuitively Understanding Variational Autoencoders% *https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf*

A Demographic Analysis of Fatal Encounter with Law

Nimra Sharnez
University of Colorado Boulder
nimra.sharnez@colorado.edu
December 23, 2019

Abstract

***Aims**: To understand the similarities in fatal encounters with law enforcement by race throughout American counties. **Design**: This analysis utilizes the 2017 American Community Survey US Census Demographic Data to understand county demographics and economic structures. The crowd-sourced database from fatalencounters.org documented fatal encounters by race from the years 2000 to 2019 was traced back to the county and state of occurrence. Statistical tests were then performed to relate race, location demographic, and socioeconomic structures of location. **Results**: There were significant differences throughout the races and fatal encounter with law enforcement. Further analysis is to be done on the predictive features of spatial distribution and opportunities in the United States amongst racial communities.*

I. Introduction

Police-minority interactions in the United States have been a controversial subject since the founding of the country. For decades, the varying characteristics of American cities produced differences in policing agencies. The encounters with law enforcement between citizens plagues urban America (Holmes, Malcolm D Brad W. Smith).

In the 1960s, the largest form of interaction between law enforcement and citizens were during race riots (Holmes, Malcolm D Brad W. Smith). In 1968, the Kerner Commission reported riots were fueled by frustration in regards to the lack of economic opportunities for the black community. The presidential panel at the time presented the conflict, "White institutions created [the ghettos], white institutions maintain it, and white society condones it," (The Kerner Report). It became well understood that black communities felt police were oppressors, maintaining the imbalance in structure.

The Kerner Commission provided many recommendations to abolish the frustration felt in the community, however, President Lyndon B. Johnson rejected the report all together. Police began to react more aggressively towards these communities than before, and the ignored report was followed by separate and unequal communities. As a result, the targeted communities began to lose trust and became increasingly bitter towards law enforcement. In 1998, former Democratic United States Senator Fred R. Harrison attributed America's debts to the hidden report, "today, thirty years after the Kerner Report, there is more poverty in America, it is deeper, blacker and browner than before, and it is more concentrated in the cities, which have become America's poorhouses"(Fred R. Harris).

Similar to the Kerner Commission findings, this paper aims to examine the socioeconomic characteristics that contribute to the fatal encounters with law enforcement from the years 2000 to 2019. The source of fatal encounter data is driven by crowd-sourced documentations, "creating an impartial, comprehensive, and searchable national database of people killed during interactions with law enforcement" (`https://fatalencounters.org/`). Understanding which race was the highest fatal encounter in a county paired with the loca-

tion's characteristics will provide further insight on the reports findings from the 1960s. In counties where majority encounters are amongst minority groups, neighborhood characteristics can substantiate the frustrations of the black and other minority communities.

II. Data

Four data sets were used during this study: one illustrates the demographics of each United States county, one provides the residential segregation index between black and white Americans per county, one shows the Median Household Income per race by state and, finally, one shows the fatal encounters with law enforcement. The data set describing demographics per county was obtained from the Kaggle website providing US Census Demographic Data, 2017 `https://www.kaggle.com/muonneutrino/us-census-demographic-data#acs2017_county_data.csv`. The variables of the data set utilized were total population, population by race, employment rate and poverty rate.

Segregation index was found through the County Health Rankings and Road maps website (`https://www.countyhealthrankings.org/`). Higher segregation indices represent larger segregation between residents. The index can be understood as the percent residents (Black or White) that would need to relocate to have an integrated Black/White community.

The residential segregation of an area is an important feature to aid in the explanation of the frustrations felt amongst minority communities regarding limited opportunities, such as education, healthcare and employment (`https://www.countyhealthrankings.org/`).

The median household income data serves to understand the varying incomes between the black and white American communities throughout the states, created through the United States Census American Community Survey (ACS). The median income and margin of error is categorized by state by the race of householder (`https://www.census.gov/`).

The Fatal Encounters with Law Enforcement (FELE) data that was used in analysis described race, county, state and year of occurrence. The database includes more features related to the encounters and can be found on the Fatal Encounters website (`https://fatalencounters.org/`).

Although robust, the vast majority of reported incidents fail to identify the race involved in the encounter, therefore reducing the number of encounters usable in analysis. This website catalogues the death of person who has died after an interaction with the police in America. Much of the data is unrefined and contains little information regarding the characteristics of the deceased persons. 211 of the 1180 FELEs in Los Angeles alone did not contain data on the race of the victim.

Additionally, due to the limited occurrences of fatal interactions between citizens and law enforcement documented, all 20 years recorded from 2000 to 2019 (n=26,975) were used for analysis. This report does however aim to understand the *locations* of occurrences and therefore may find value in the cumulative effect of economic structures.

Of the 50 states, California, Illinois and Texas had the counties with the highest number of FELE. These three counties were used throughout the analysis. Figure 1 describes the count, percent of FELE in the state compared to all American FELE and the counties of highest FELEs within the states.

Highest FELEs by the State by County			
	California	Texas	Illinois
Count	4,383	2,383	1,004
Percent of US FELE (%)	16.25	8.83	3.72
County	Los Angeles	Harris	Cook

Figure 1

III. Methods

To begin, basic descriptive analyses on the FELE dataset were collected. Highest occurrences of death by state, county, and race were recorded. Using the census data of state and county demographic, the race variable was adjusted for skew through standardizing FELEs. A test statistic for each race was calculated to analyze the differences in population proportions between the races, ages and genders. This analysis was done on the three counties seen in Figure 1.

Secondly, a multi-variate analysis was conducted to understand how a states' segregation between the black and white communities affect chances of a FELE. The three states in Figure 1 were used along with 3 additional states: New York, New Jersey and Maryland. Plotted along the x axis, residential segregation indices ranged from 54 to 74. The y axis compared the black and white median income per household by the state. Fatal encounters and demographic populations amongst the two communities for each state were additional features in the multivariate analysis (Figure 13).

Through these methods, the hypothesis *there will be a balanced distribution of FELE among a populations races* will be rejected if there is a significant difference between the population proportions of a race and FELE by race. With a county sampling of a large enough expected count (expected value > roughly 5% of population) that is less than 10% of the overall population, a χ^2 goodness of fit analysis is valid (The Chi-square test of independence, 2013). The multi-variate analysis serves to provide additional insight to the differing opportunities offered to the black and white communities through understanding the varying incomes amongst the two communities.

IV. Results

I. Study Population Characteristics

From the 10,105,722 Los Angeles residents, the majority demographic with an estimate of 4,891,169.45 (population proportion) was the Latinx community (48.4%). The second largest community was European-American/White (26.5%) and the smallest demographic population is the Native American community(0.2%). Further sizes and population proportions of other races throughout the counties are presented in figures 2-4.

II. Normalized FELE by the Race

The two variables, county demographic population and FELE demographic count, are on different scales. Figures 2-4 below eliminate the FELE : population ratio. The normalized data reduces the effect of population density by one FELE for every 10,000 people of a given race.

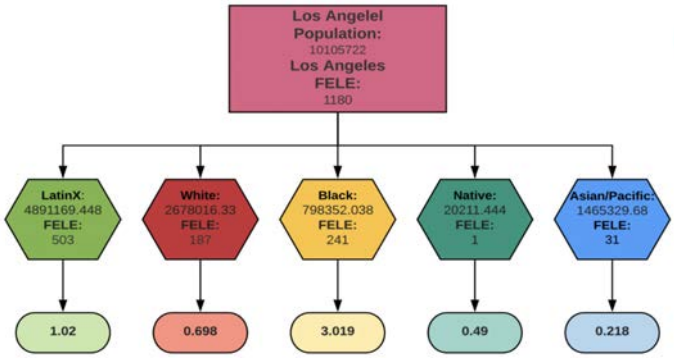


Figure 2: *Los Angeles, CA Normalized Data*

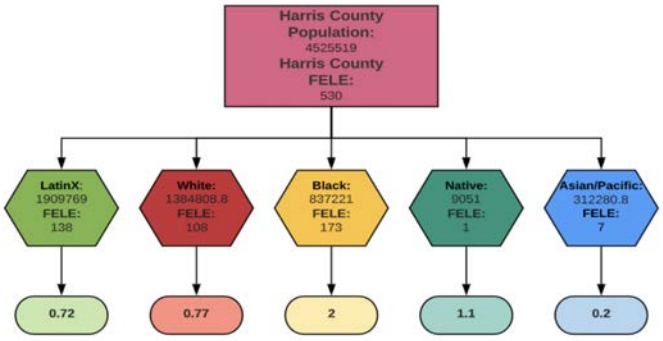


Figure 3: *Harris, TX Normalized Data*

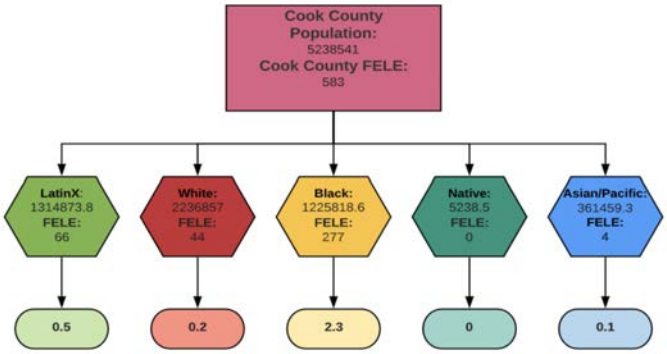


Figure 4: *Cook, IL Normalized Data*

III. Los Angeles Demographic Population

The normalized Los Angeles FELE data shows the highest death per 10,000 people of any given race was the African American/Black race at 3 FELE per 10000 African American/ Black people. It is worth mentioning that this community makes up just 7.9% of the county's population. This community is followed by the largest occupying demographic in Los Angeles (LatinX) at 1 FELE per 10,000.

Given the distributional assumption that the FELE by race in Los Angeles mirror that of the demographic population of Los Angeles, the frequencies across the races were measured through the county sample size of 1180 Los Angeles FELE.

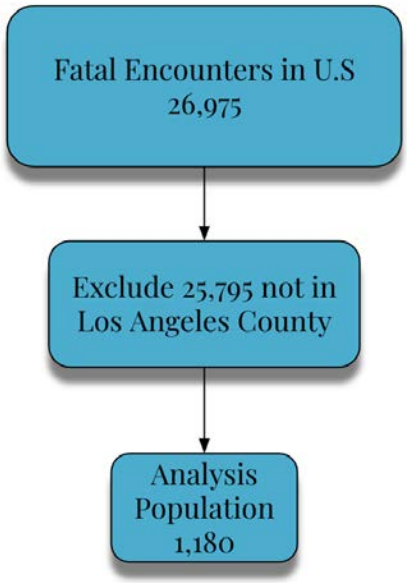


Figure 5

Figure 5 describes the county sample extracted from the 26,975 FELE. The χ^2 goodness of fit test was conducted to understand how the FELE resembled the data pertaining to the demographic population collected from the census data. Due to the small Native American population in Los Angeles and that expected value is determined by population, the Native American expected count (n = 0.2) was combined with unspecified races (n = 5.5).

1180 FELE Sample in LA		
Race	Expected Number	Actual Number
Latinx	571.1	503
White	312.7	187
Black	93.2	241
Asian/Pacific	171.1	32
Native American	23.6	1

Figure 6: χ^2*test results(p<0.00001)*

Figure 6 presents the expected versus actual FELE based on the demographic population of Los Angeles. With the large χ^2 statistic of 33.6 (discluding unspecified races), our p-value (< 0.00001) is less than the significance level ($\alpha = 0.05$). We can reject the null hypothesis for a mirrored distribution in the FELE throughout the races with the demographic population.

IV. Harris, TX Cook, IL Demographic Population

Figures 7 and 8 represent the χ^2 results for Harris County, TX and Cook County, IL.

530 FELE Sample in Harris County, TX		
Race	Expected Number	Actual Number
Latinx	223.7	138
White	162.18	108
Black	98.05	173
Asian/Pacific	36.57	7
Native American	10.6	1

Figure 7: χ^2*test results(p<0.00001)*

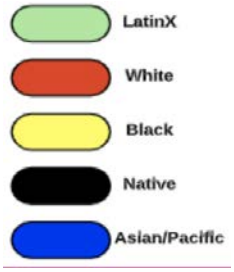
583 FELE Sample in Cook County, IL		
Race	Expected Number	Actual Number
Latinx	145.8	66
White	248.9	44
Black	136.422	277
Asian/Pacific	40.2	4
Native American	0	0

Figure 8: χ^2*test results(p<0.00001)*

In both counties, the p-value is less than the significance level (p<0.000001), rejecting the null hypothesis as well.

With the three highest FELE counties proving a significant difference in the amount of residents by race and victims of fatal encounters by race, it is reasonable to begin a deeper dive and look into these locations of occurrences.

Figures 9, 10 and 11 represent the FELEs and race in the three states analyzed (California, Texas and Illinois). Each marker represents one FELE. The color legend specifying race by color can be seen.



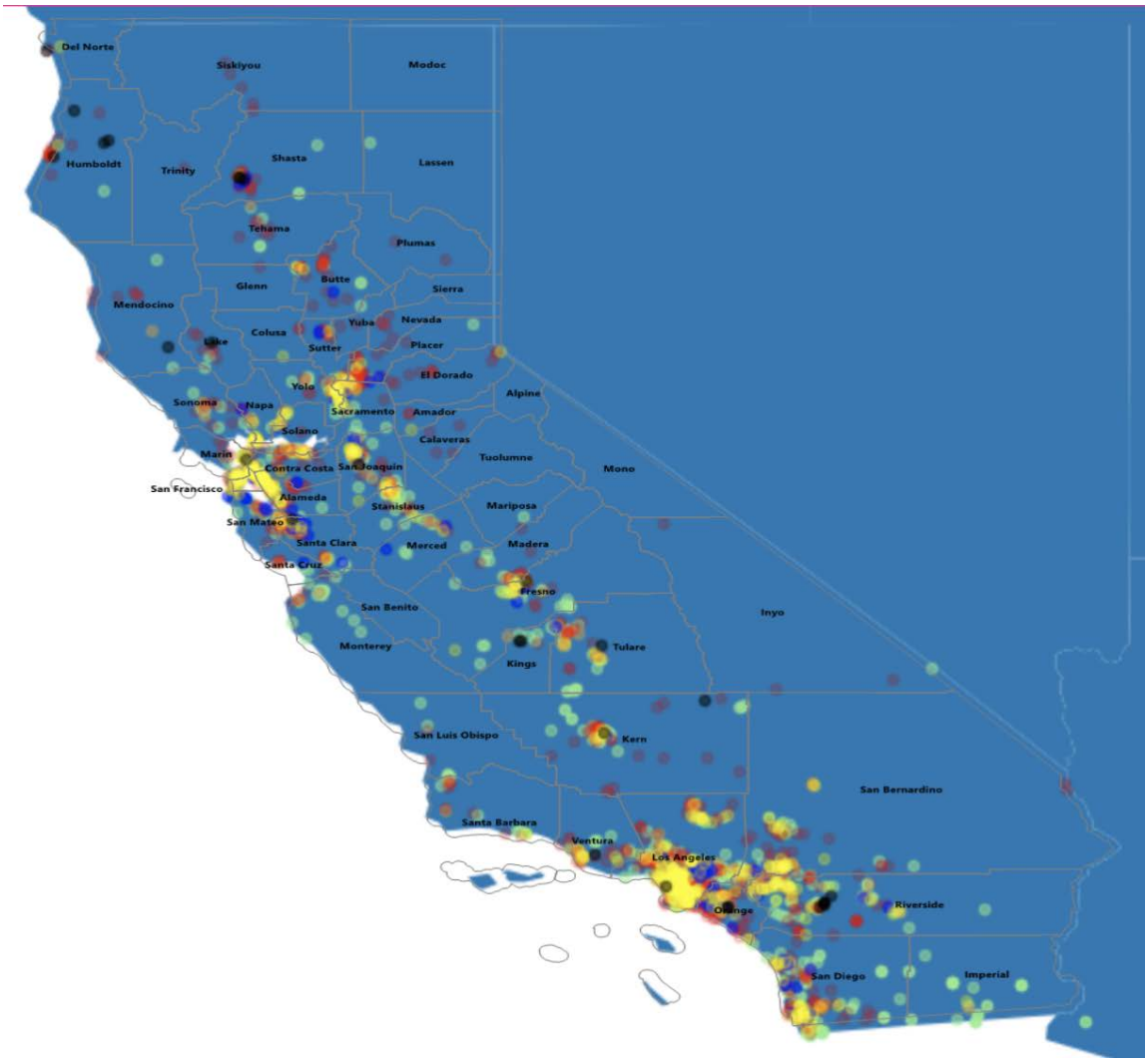

Del Norte
Siskiyou
Modoc
Humboldt
Trinity
Shasta
Lassen
Tehama
Plumas
Glenn
Butte
Sierra
Mendocino
Yuba
Nevada
Colusa
Lake
Sutter
Placer
El Dorado
Yolo
Alpine
Sonoma
Napa
Sacramento
Amador
Solano
Calaveras
Marin
Tuolumne
Contra Costa
San Joaquin
Mono
San Francisco
Alameda
Stanislaus
Mariposa
San Mateo
Santa Clara
Merced
Madera
Santa Cruz
Fresno
San Benito
Inyo
Monterey
Tulare
Kings
San Luis Obispo
Kern
San Bernardino
Santa Barbara
Ventura
Los Angeles
Orange
Riverside
San Diego
Imperial

Figure 9

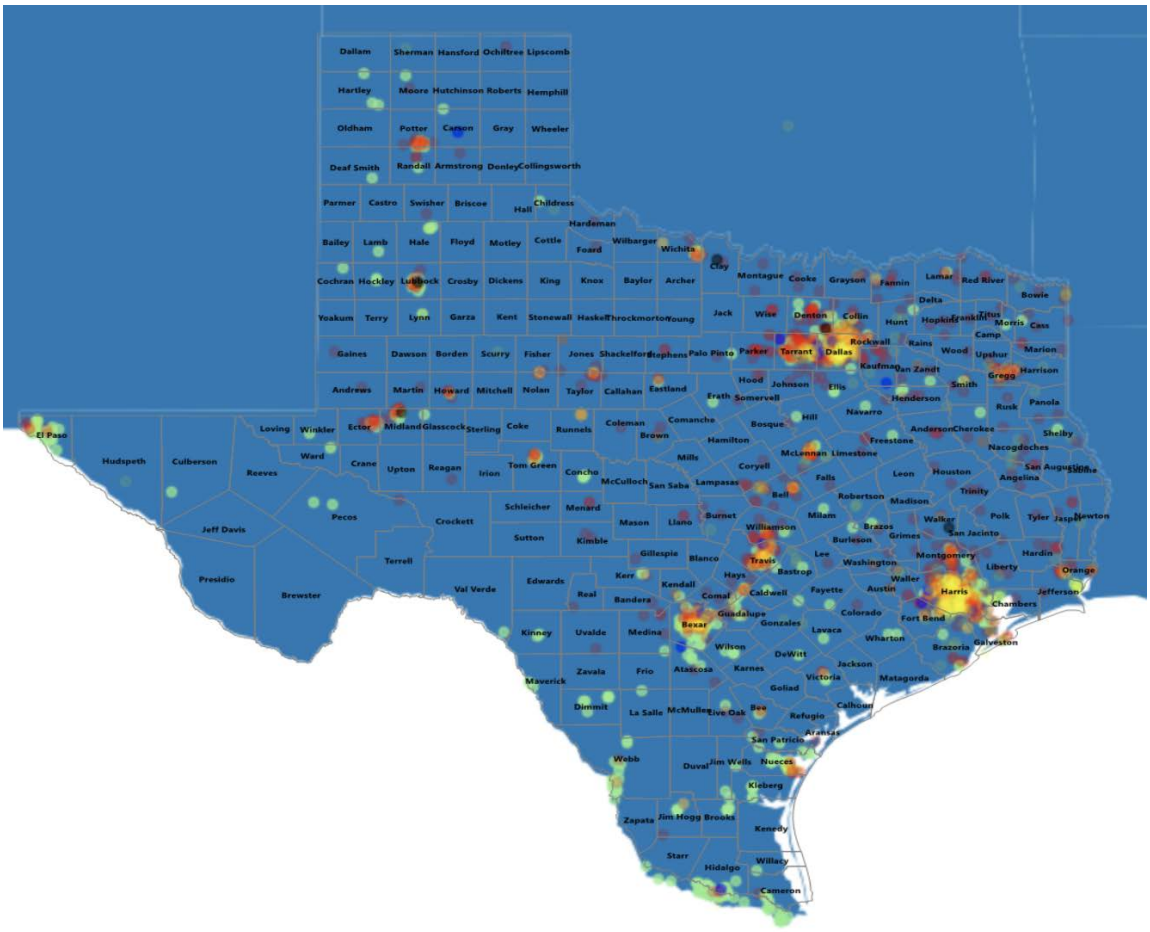

Dallam
Sherman
Hansford
Ochiltree
Lipscomb
Hartley
Moore
Hutchinson
Roberts
Hemphill
Oldham
Potter
Carson
Gray
Wheeler
Deaf Smith
Randall
Armstrong
Donley
Collingsworth
Parmer
Castro
Swisher
Briscoe
Hall
Childress
Hardeman
Bailey
Lamb
Hale
Floyd
Motley
Cottle
Foard
Wilbarger
Wichita
Clay
Cochran
Hockley
Lubbock
Crosby
Dickens
King
Knox
Baylor
Archer
Montague
Cooke
Grayson
Fannin
Lamar
Red River
Bowie
Yoakum
Terry
Lynn
Garza
Kent
Stonewall
Haskell
Throckmorton
Young
Jack
Wise
Denton
Collin
Hunt
Delta
Hopkins
Franklin
Titus
Morris
Cass
Camp
Gaines
Dawson
Borden
Scurry
Fisher
Jones
Shackelford
Stephens
Palo Pinto
Parker
Tarrant
Dallas
Rockwall
Rains
Wood
Upshur
Marion
Kaufman
Van Zandt
Gregg
Harrison
Andrews
Martin
Howard
Mitchell
Nolan
Taylor
Callahan
Eastland
Erath
Somervell
Hood
Johnson
Ellis
Henderson
Smith
Rusk
Panola
Loving
Winkler
Ector
Midland
Glasscock
Sterling
Coke
Runnels
Coleman
Comanche
Brown
Bosque
Hill
Navarro
Freestone
Anderson
Cherokee
Shelby
Nacogdoches
El Paso
Hudspeth
Culberson
Reeves
Ward
Crane
Upton
Reagan
Irion
Tom Green
Concho
McCulloch
San Saba
Lampasas
Mills
Hamilton
McLennan
Limestone
Coryell
Falls
Leon
Houston
Angelina
San Augustine
Sabine
Trinity
Ward
Pecos
Crockett
Schleicher
Menard
Mason
Llano
Burnet
Bell
Milam
Robertson
Madison
Walker
Polk
Tyler
Jasper
Newton
Jeff Davis
Sutton
Kimble
Williamson
Brazos
Grimes
Burleson
San Jacinto
Terrell
Gillespie
Blanco
Travis
Lee
Washington
Montgomery
Hardin
Presidio
Brewster
Val Verde
Edwards
Kerr
Hays
Bastrop
Waller
Liberty
Orange
Real
Kendall
Bandera
Comal
Caldwell
Fayette
Austin
Harris
Jefferson
Chambers
Guadalupe
Colorado
Fort Bend
Kinney
Uvalde
Medina
Bexar
Gonzales
Lavaca
Wharton
Brazoria
Galveston
Wilson
DeWitt
Zavala
Frio
Atascosa
Karnes
Jackson
Victoria
Matagorda
Maverick
Goliad
Dimmit
La Salle
McMullen
Live Oak
Bee
Refugio
Calhoun
San Patricio
Aransas
Webb
Duval
Jim Wells
Nueces
Kleberg
Zapata
Jim Hogg
Brooks
Kenedy
Starr
Hidalgo
Willacy
Cameron

Figure 10

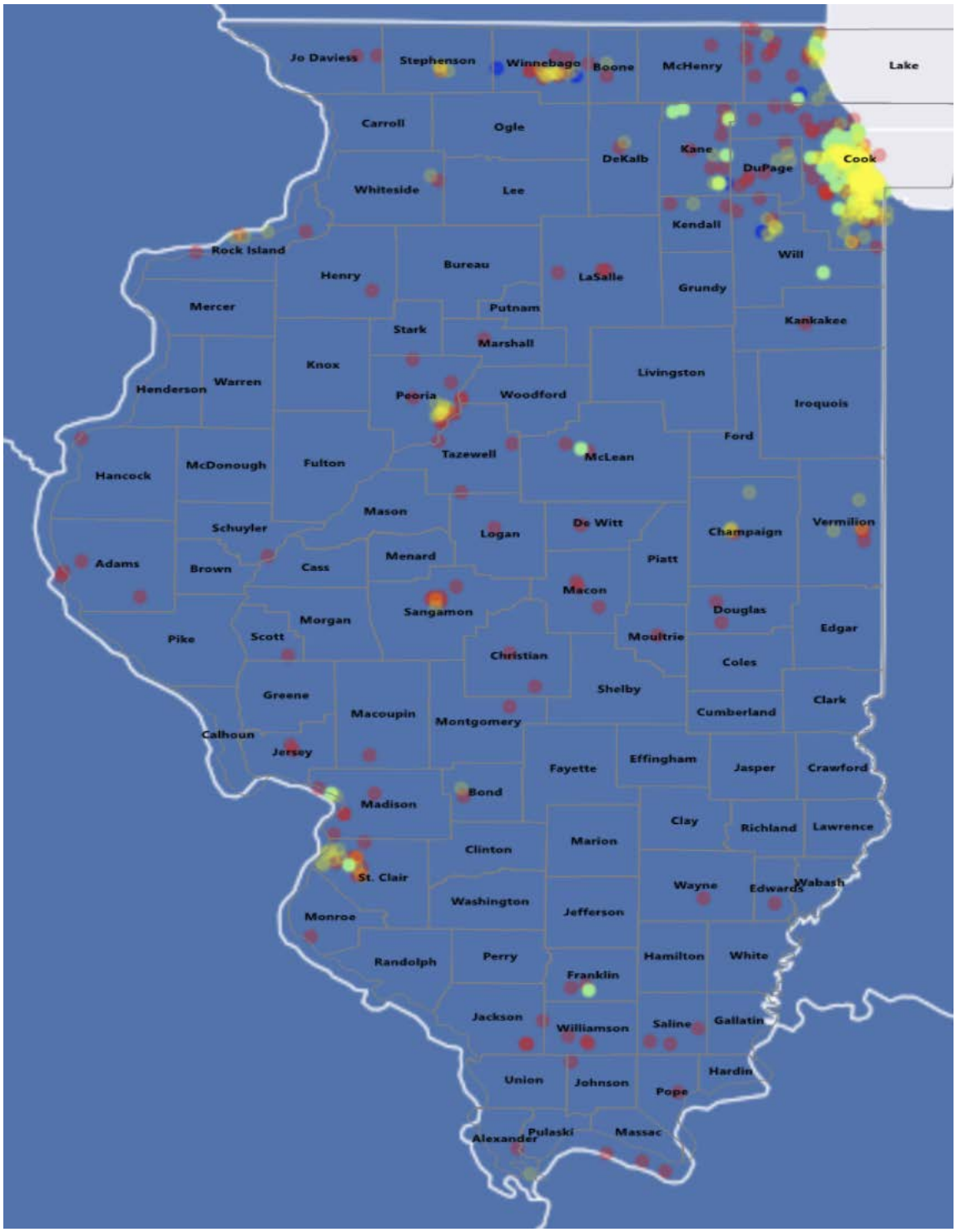

Jo Daviess
Stephenson
Winnebago
Boone
McHenry
Lake
Carroll
Ogle
DeKalb
Kane
DuPage
Cook
Whiteside
Lee
Kendall
Will
Rock Island
Henry
Bureau
LaSalle
Grundy
Mercer
Putnam
Kankakee
Stark
Marshall
Knox
Livingston
Henderson
Warren
Peoria
Woodford
Iroquois
Ford
Tazewell
McLean
Hancock
McDonough
Fulton
Mason
Schuyler
Logan
De Witt
Champaign
Vermilion
Adams
Brown
Cass
Menard
Piatt
Macon
Douglas
Morgan
Sangamon
Edgar
Pike
Scott
Moultrie
Christian
Coles
Shelby
Greene
Clark
Macoupin
Montgomery
Cumberland
Calhoun
Jersey
Fayette
Effingham
Jasper
Crawford
Madison
Bond
Clay
Richland
Lawrence
Clinton
Marion
St. Clair
Wayne
Edwards
Wabash
Washington
Jefferson
Monroe
Randolph
Perry
Hamilton
White
Franklin
Jackson
Williamson
Saline
Gallatin
Union
Johnson
Hardin
Pope
Alexander
Pulaski
Massac

Figure 11

V. Neighborhood Structure and Race

The model to analyze the effect of a racial background's likelihood of a FELE given the neighborhood structure of occurrence utilized the 2017 census, census median income by race and segregation index data. These features allowed for a multi-variate analysis of the counts of a given race's FELE in a location.

VI. County State Features/ Disadvantages

To begin, a descriptive analysis was conducted on the three sample counties to understand the poverty and unemployment rates compared to the mean average county poverty and unemployment rates throughout the state (figure 12).

Although the poverty rates were higher in the counties sampled than that county state averages, there was no significant difference in either poverty or unemployment rates (p-values > 0.5; χ^2).

Moving past the local demographics, the spatial distribution of black and white communities may provide additional predictive insights to the unequal distributions of FELE and race.

The segregation index of the three states we analyzed are visualized (x-axis) in figure 13 with the addition of three more states, New York, New Jersey and Maryland. These three states were selected based on their black/white segregation indices providing more insight to the feature. Along the vertical axis, the median household income for the black and white communities are displayed with their corresponding error. Bubbles on the same segregation index are the same state. The size of the bubble is determined by the number of residents in that given state. The larger bubbles on any given state is the white population given there are more residents of that demographic for Texas, California, New Jersey, Maryland, Illinois and New York. Finally, the color-scale represents the numbers of fatal encounters for that race as seen on the legend to the right of the graph. An interactive graph can also be found here: `https://colab.research.google.com/drive/1W2dgiQbAS4TWQ-hFwQXEKhUmTsCpcQRa`.

The higher the segregation index of a state is, there seems to be a similarity in the number of FELE for both races, regardless of the demographic population. Further analysis is to be conducted regarding this multi-variable graph.

Poverty and Unemployment (%)			
	Los Angeles, California	Harris, Texas	Cook, Illinois
Poverty (County, State)	17, 15.73	16.8, 16.3	15.9, 13.95
Unemployment (County, Average)	7.8, 8.26	6.4, 6	8.7, 6.6

Figure 12

Black/ White Median Income & Segregation Index Throughout the States & FELEs

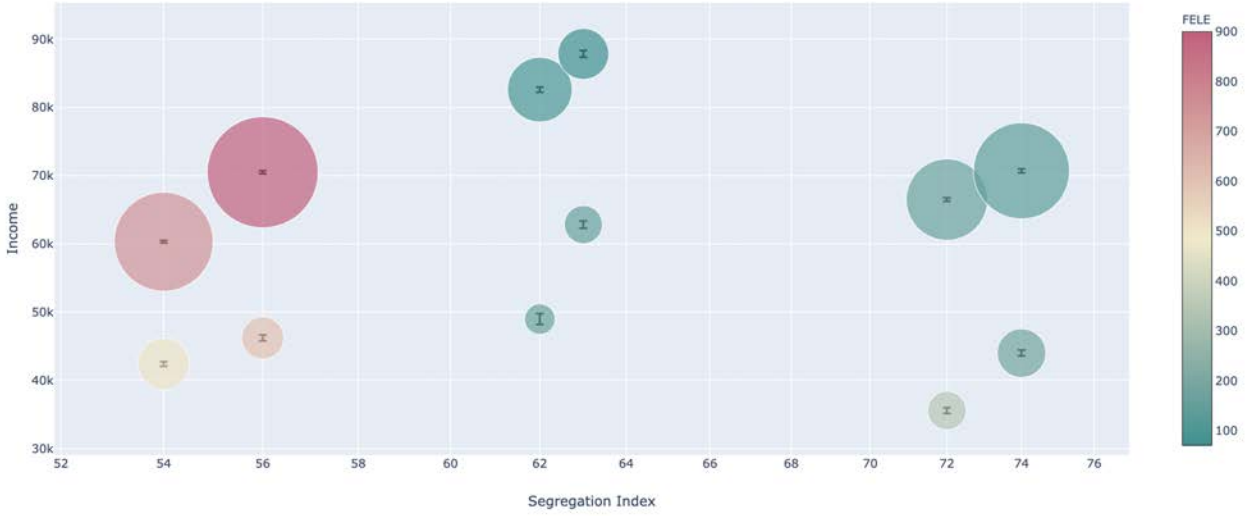


Figure 13 (Left to Right: TX, CA, NJ, MD, IL, NY)

V. Discussion and Conclusions

According to the Kerner Commission, the frustration in the black and minority communities stems from the disadvantages in their hometowns. This study broke down the states in the United States to understand the socioeconomic structures on a county level.

Although these results provided insight to the problem in America, further analysis would go a step further and analyze the neighborhood conditions on a smaller level. Perhaps by taking high FELE counties and breaking those into smaller neighborhoods to understand if race is still significant on a smaller scale.

To go further, once finding locations of unequal FELE by race distributions, similar disadvantages (below average county income, unemployment and poverty) can be understood. Lastly, moving away from local demographics and looking into spatial distribution may provide a deeper analysis of the problem.

This research and future analysis of the subject would serve to provide statistical verification for the frustration felt amongst those in disadvantaged communities. It is important to understand the racial backgrounds of the communities as it represents the repetitive cycle of oppression felt amongst races.

This report was heavily focused on confirmation of the FELE issue in the years 2000 to 2019 summed together. Understanding how targeted communities have remained oppressed would be the next steps in understanding the trends in the frustration. It is also important to note that there is a lot of missing data and unspecified races in this data set, especially because fatal encounters are less likely than violent encounters. Future analysis can understand and further verify any correlation through data sets where the encounters are non fatal but still violent.

References

[ACS, 2019] American Community Survey "US Census Demographic Data." *Kaggle*, 3 Mar. 2019, *www.kaggle.com/muonneutrino/us-census-demographic-dataacs2017*$_c$*ounty*$_d$*ata.csv*.

[County Health Rankings, 2019] County Health Rankings. "Residential Segregation - Black/White*." County Health Rankings amp; Roadmaps, 2017, *www.countyhealthrankings.org/explore-health-rankings/measures-data-sources/county-health-rankings-model/health-factors/social-and-economic-factors/family-*

social-support/residential-segregation-blackwhite.

[FELE, 2019] *"Fatal Encounters." Fatal Encounters, fatalencounters.org/.*

[Median Household Income, 2017] United States Census Bureau. United Stated Median Household Income By State by Race of Householder, 2013-2017 *https://www.census.gov/programs-surveys/acs/*

[Race and Police Brutality, 2008] Holmes, Malcolm D.., and Brad W.. Smith. Race and Police Brutality Roots of an Urban Dilemma. State University of New York Press, 2008.

[The Chi-square test of independence, 2013] McHugh, Mary L. "The Chi-Square Test of Independence." Biochemia Medica, Croatian Society of Medical Biochemistry and Laboratory Medicine, *15 June 2013, www.ncbi.nlm.nih.gov/pmc/articles/PMC3900058/.*

[Kerner Commision, 1968] "'Our Nation Is Moving Toward Two Societies, One Black, One White–Separate and Unequal': Excerpts from the Kerner Report." HISTORY MATTERS - The U.S. Survey Course on the Web, *historymatters.gmu.edu/d/6545/.*

Customer Demographics Study

Orgil Sugar
University of Colorado Boulder
orgil.sugar@colorado.edu

Abstract

One of the local museums has been trying to reach out to local communities by doing various activities among them. The museum's main goal is to educate the younger generation and expand their reach. They will need to increase their annual funding in order to host effective and artistic events. Thus, their customer data and with help from the most recent us census data will reveal the patterns that locals make, especially their audience.

I. Introduction

In the Colorado state, especially in the Boulder county, there is not an uniform-age distribution due to the increasing number of college students where the younger age dominates the most of the population in the area. Due to that increase, the most businesses target the most obvious customers which is college students but one of the local museums requested us to make study on their customer demographics. This study will hopefully help them for making future decisions and aiming for correct customers who show a great deal of respect and interest in the museum.

Our goal for this study is to explore the data as much as we can and give some useful intuitions for the museum. There has not been any study like this done for the museum before except the local county's annual reports on the local communities where they do surface level analysis.

II. Data

The main datasets employed in the study are: '2016 US Census data[1]' , 'Personal communication data[2]' and various 'Google APIs[3]'. The museum's personal communication data has been provided by them where some sensitive private information are trimmed prior to the study. Each data set has a granularity of five digit zipcode and we will explore customers' demographic on state, county and five-digit zipcode level. There are 1022 unique zipcodes in the personal communication data which covers the 50 states. 157 of these are in Colorado where they receive larger number of audience from. The US census data is collected and sanitized by one of the python libraries. The google APIs are used for additional study related problems, such as travel distance between 2 zipcodes.

1.https://pypi.org/project/uszipcode/
2.Data from the local museum
3.On Google Cloud Platform

III. Methods

Our main goal is exploration of data and revealing the patterns that are hidden. Basic descriptive analyses (means and standard deviations) were calculated and all variables were then transformed to their logarithmic values to adjust for skew depending on the values. Next, bivariate and multivariate regression analyses were conducted to examine the relationship between various data fields. Multivariate regression analyses was used to produce the most parsimonious sociostructural models for each state. As the bivariate procedure can lead to biased estimates because of the other variables that should have been taken into consideration. During the exploration, the elimination and inclusion of vari-ables can affect the explanatory power of theresult. The most important candidate for thisstudy is the income data which can be found from the US Census data. Before performing the regression methods, the distribution of variables was checked to make sure they were not omitted in the selection process due to tight range of values. The state of Colorado has the largest audience.

Analyses of small audiences are subject to a variety of biases due to lack of data.In order to detect and underempahize the biases, sorting the zipcodes and states based onthe number of audience, will give a reasonable area of interest. As for the museum, they are only interested in the local communities where they receive the most of their audience and that will align with the suggested filtering.

The US Census data has a multiple flaws and one of which is the distribution of number of returns in every zipcode level and, that should be taken into consideration. It is impossible to know the exact number of population from the number of returns. Thus, instead of using the exact number of population, assigning weights to the zipcodes depending on their metadata is the most reasonable approach. Another challenge is to take account of the difference between data sets. Meaning that the census data is out-dated compared to the personal communation data and consideration of the average relocation is the last crucial part in the methods section. Considering the other similar studies done with the US census data. This difference is not our most concerned part of the dataset.

IV. Results

First of all, we should start analyzing the attendance from different states and see if there is any correlation between state-level demographics and the attendance. As you see from Figure 1.0 where the attendace from the state of Colorado is the largest because of the museum's location. After Colorado, the attendance drops by two digits but California, Texas and New York have higher attendance rate than others. You can see the the logarithmic value of the attendace visualized on the map of the USA (Figure 1.1) on the next page where the state of Colorado is excluded to show the difference in other states.

	State	Attendance
1	CO	10877
2	CA	298
3	TX	243
4	NY	236
5	IL	120
6	MD	116
7	FL	114
8	MI	95
9	PA	81
10	MA	77
11	MN	76
12	IN	75
13	NC	65
14	NJ	64
15	AZ	64
16	OH	58
17	WI	49
18	MO	47
19	DC	46
20	WA	43
21	IA	42
22	NE	41
23	CT	38
24	GA	36
25	OR	33
26	VA	32
27	VT	30
28	KS	30
29	TN	29
30	KY	29
31	SC	28
32	ME	23
33	UT	22
34	LA	22
35	AR	20
36	NM	18
37	WY	16
38	SD	13
39	NH	13
40	NV	12
41	AL	12
42	HI	11
43	OK	10
44	MT	7
45	AK	6
46	RI	5
47	ND	4
48	DE	4
49	ID	2
50	MS	1

Figure 1.0
Attendance from each state.

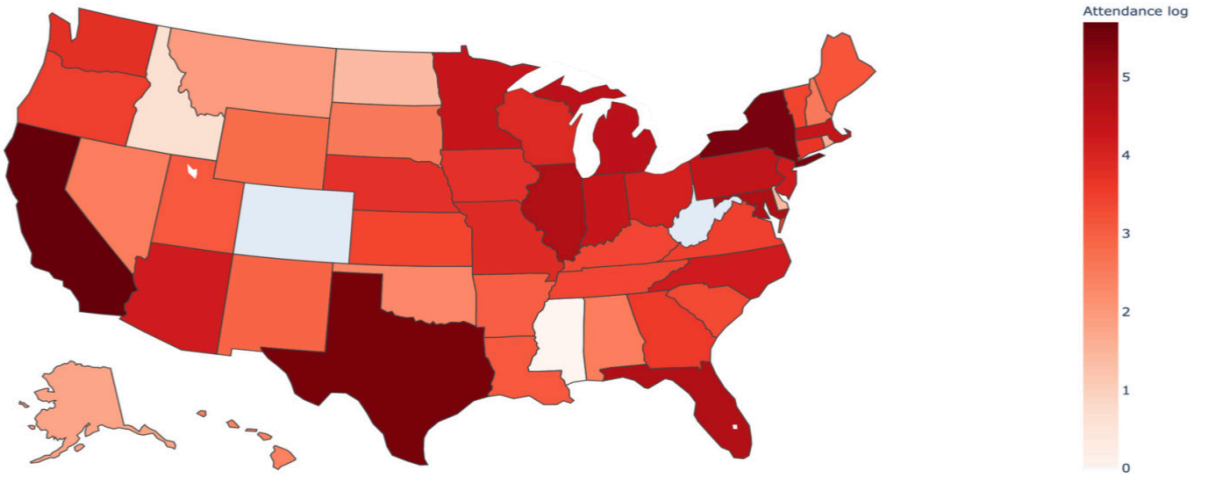


Figure 2.0
Attendance from each state.

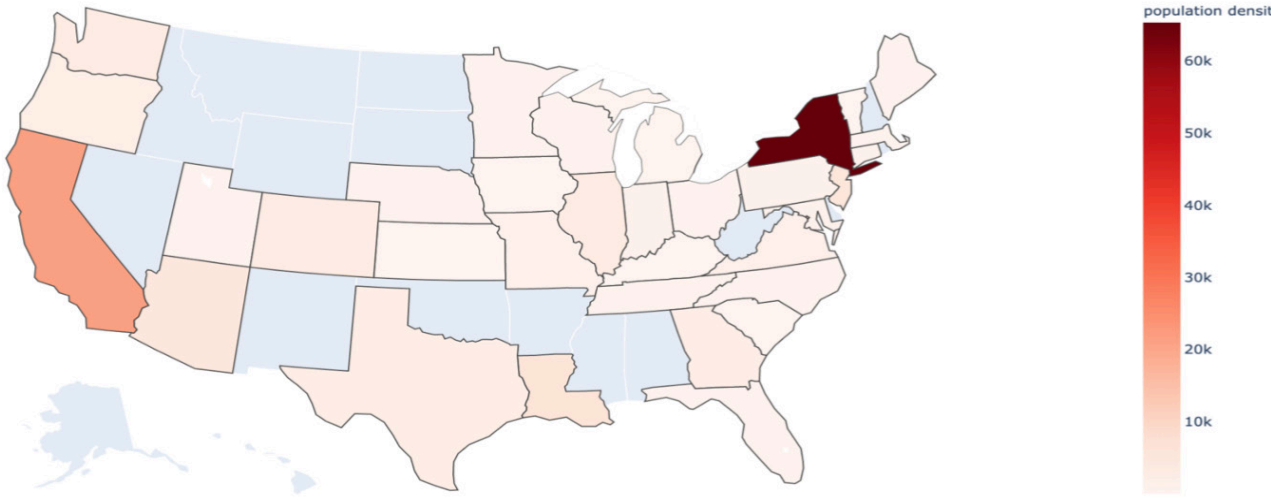


Figure 2.1
Population density on possible states

As you can see from the figures above that the high attendance from the states mentioned before are correlated to their population density. Higher the population, it is more likely to have a customer from that state whereas the chances increase with the population density. Furthermore the following visualizations will show how much does the median home get valued where the occupation of those houses are displayed below. Meaning that there is some correlation for some states where the housing value affect the occupation in every unit. If the housing unit is cheaper then it's highly likely to be empty.

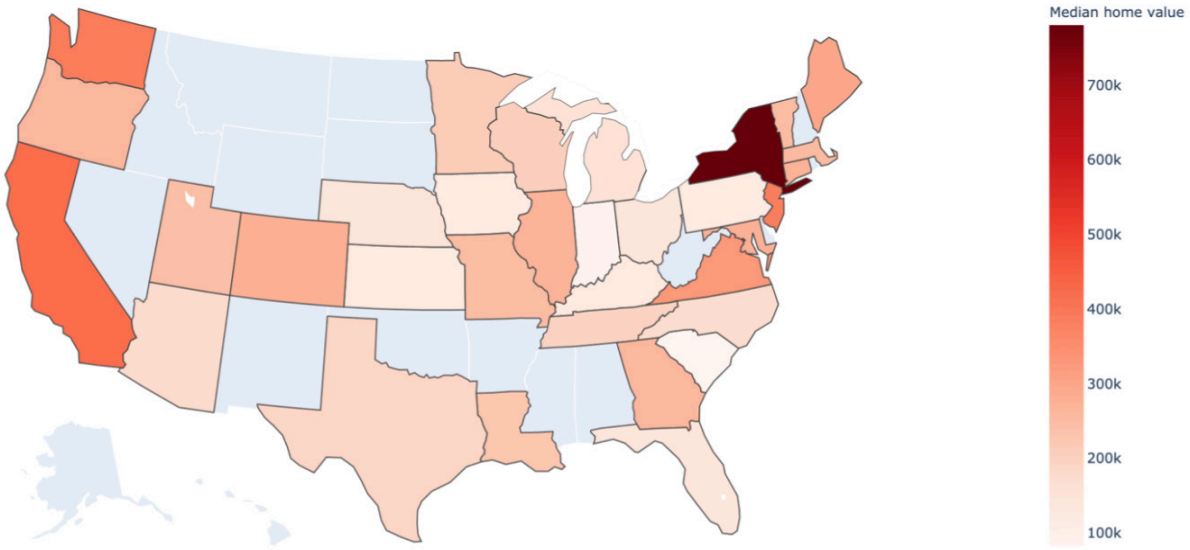


Figure 2.2
Median home value across the USA

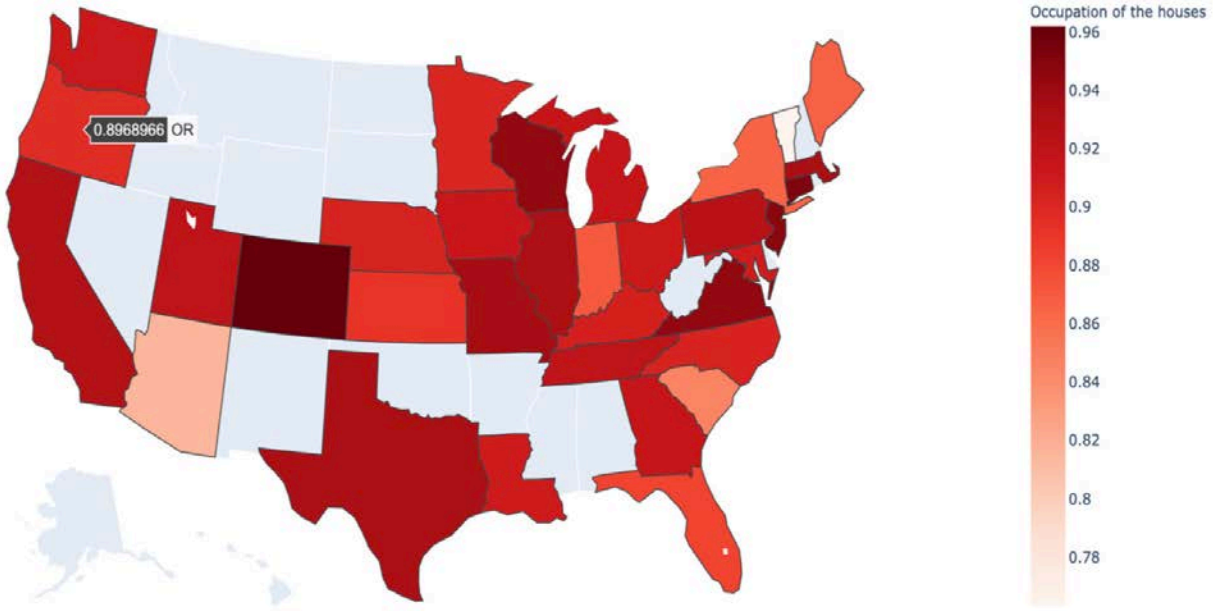


Figure 2.3
Occupation of the houses across the USA. (Higher means more occupation for every housing unit.)

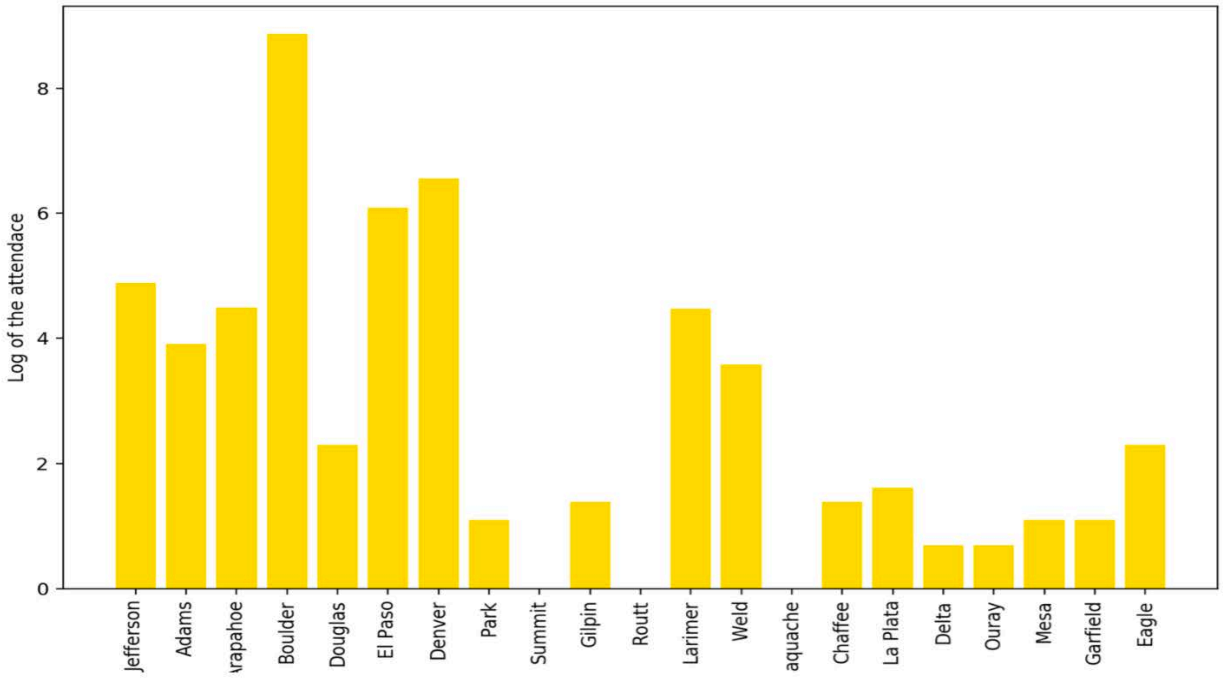


Figure 3.0
Attendance from counties in Colorado in log-scale

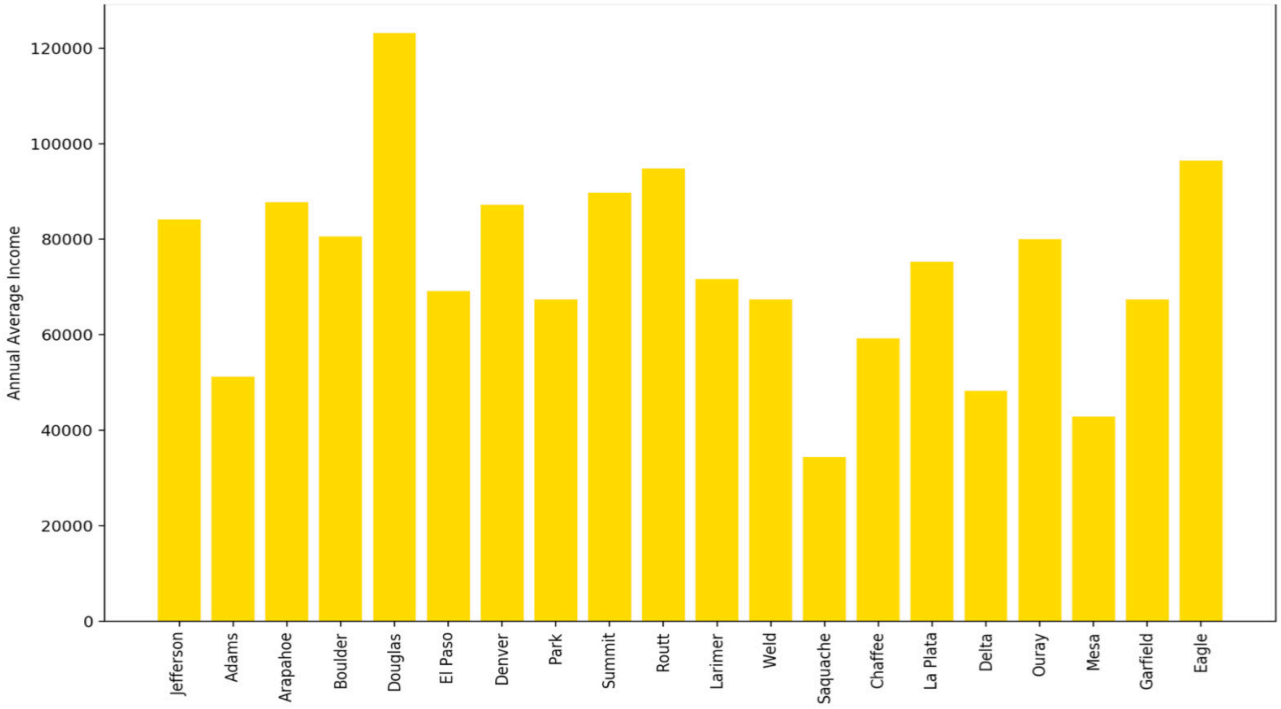


Figure 3.1
Annual Income for Colorado counties

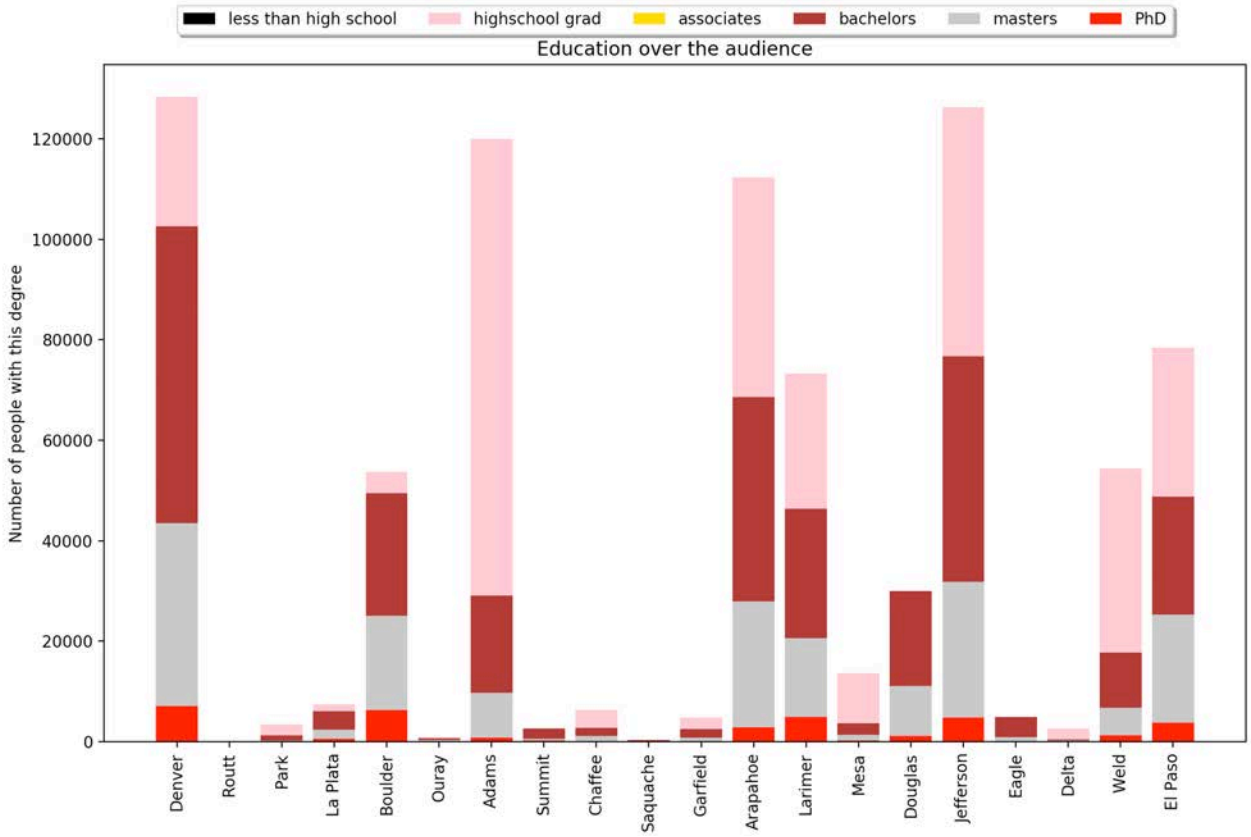


Figure 3.3
Education in the audience in log-scale

In Figure 3.0 above, we are looking at the attendance rate from Colorado counties where some counties have zero attendance. We should be aware of that the museum is located in Boulder county where they will get their most attendance from. The attendance will be likely decay based on the distance from the Boulder county but we are hoping to find some outliying pattern where educated people are putting more interest in art. The universities should be also taken into account while analyzing the visualizations because there are more likely to be more academics around universities. Also, the average annual income might affect people's interest in art. Both of the insights are also visualized above as Figure 3.2 and 3.3.

In the next visualization, the average amount of time people spend traveling from their homes to the museum which will be obviously decaying based on the distance from the Boulder county. However, as the driving period increases we might find people who are most likely to drive from their homes because of their daily commute is considerably higher than traveling duration to the Boulder county. The shape of the road (congestion) and traffic laws should be mentioned because they affect the traveling distance and period in their own ways. The average traveling period is calculated for every zipcode to Boulder county and put into county bins where the average you are seeing above is extracted.

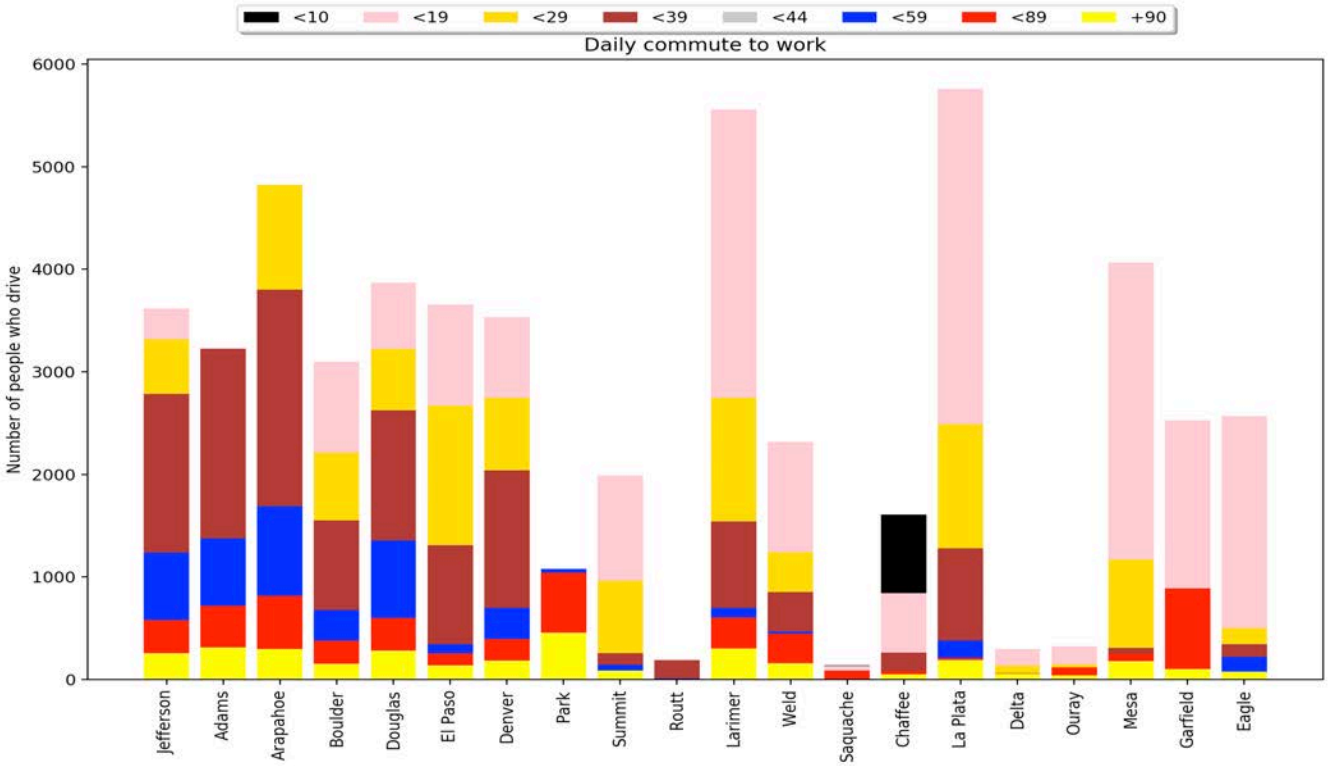


Figure 4.1
Daily commute that people make to work in minutes in log-scale

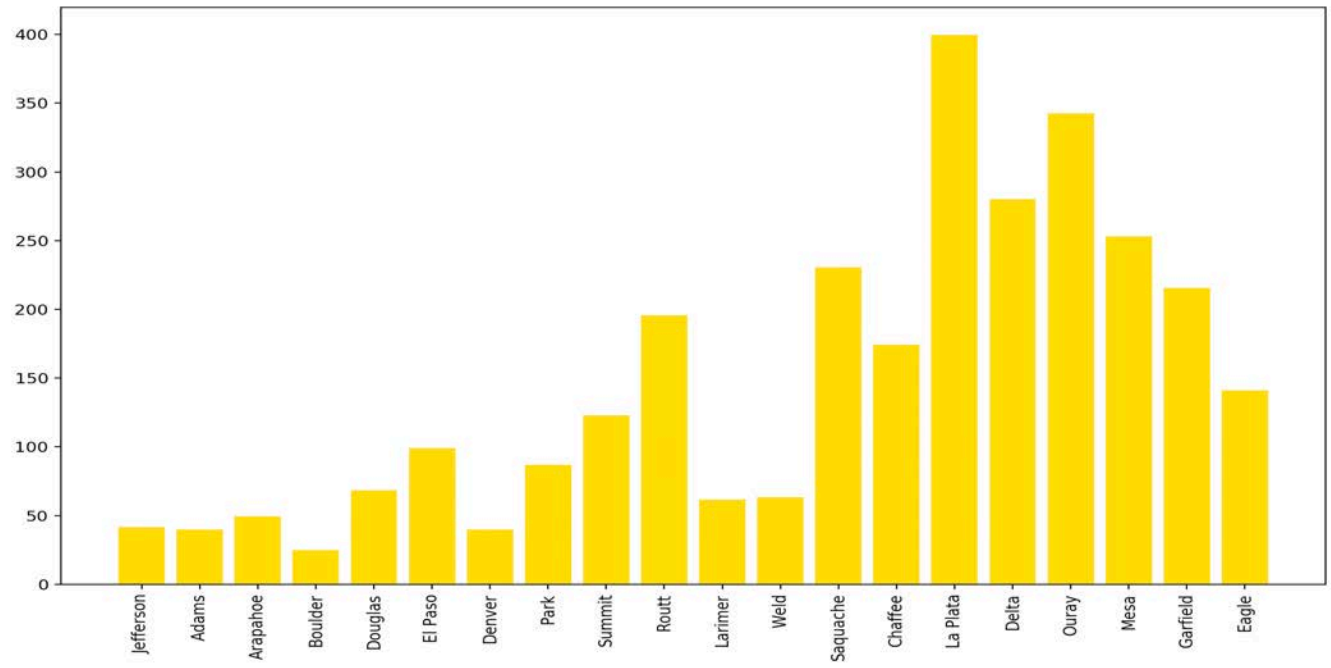


Figure 4.2
Travel duration from county to the museum in minutes (bar graph)

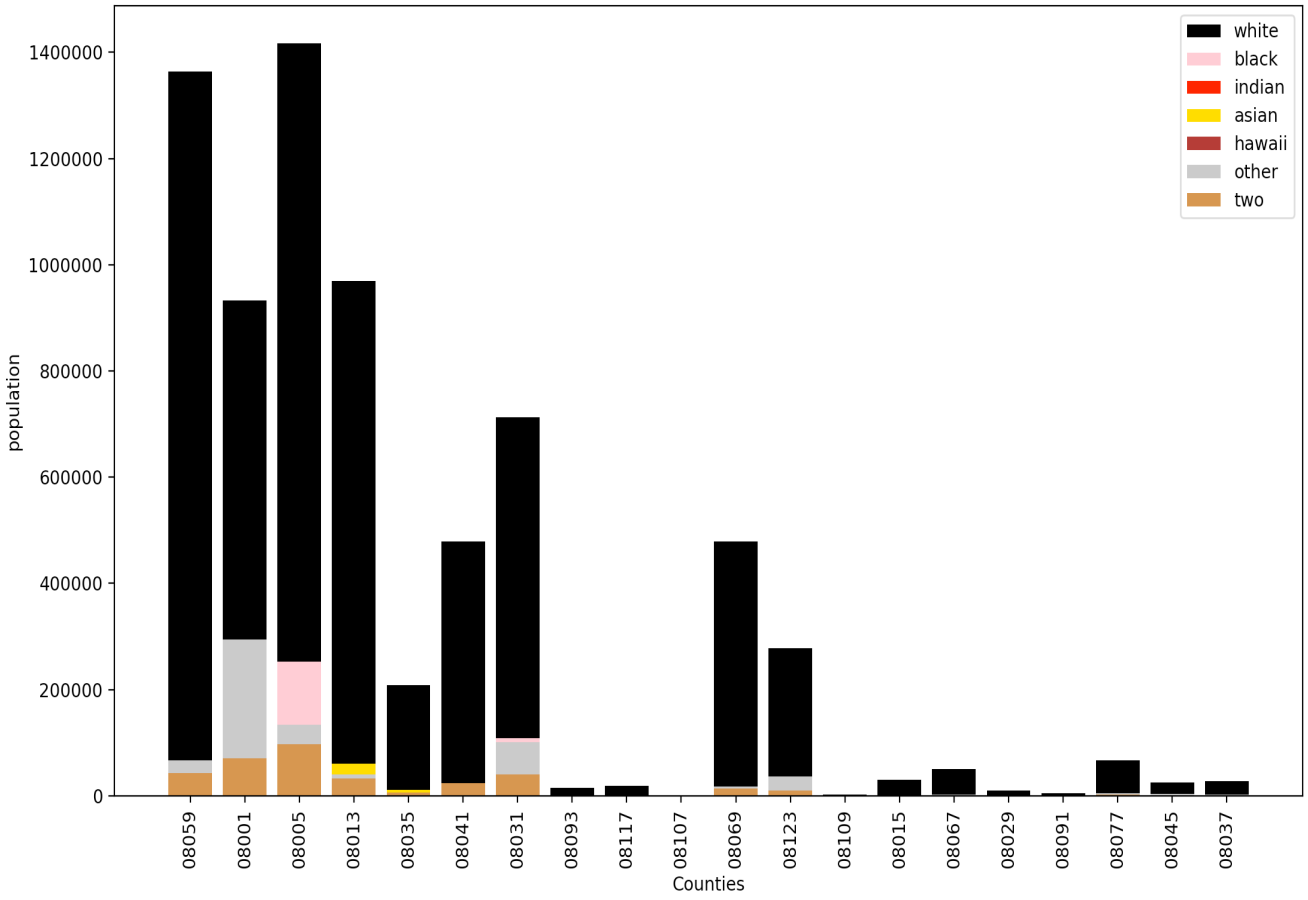


Figure 5.0
Racial diversity in the counties in log-scale

In the figure 5.0, the racial divesity in the Colorado counties are presented in log-scale. As you can see from the visualization, after the white race the majority of the population have two or more racial description. Before that description, 'white' communities mostly dominate the rest of the population and we noticed that there are even zero occurence in some racial groups. According to this visualization, the museum might be interested in presenting events for majority of the population or hosting event where the minority commnuties can attend and that can be correlated to their culture which will catch their attention and interest more.Ultimately, the figures above will be useful for the museum to make future decisions such as relocation to a different community while maintaining the current attendance rate and possibly targeting more. As I mentioned before, the museum is a non-profit organization that is looking for potential donators and more audience whom they can gather a good amount of their annual budget. Also, this report will hopefully help them for their annual report and more insights for the current donators whereas showing them how much is the museum affecting the communities around Boulder county and attracting more people, even from El Paso country . The attendance rate from El Paso country , Texas state were the surprising insights that they can leverage and present. Finally, o ur journey through the US Census data and Personal Communication data from the museum will end here .

Analysis of the Spread of Restaurant Trends in Las Vegas, Nevada

Tyler Tokumoto

University of Colorado, Boulder
tntokum@gmail.com

December 14, 2019

Abstract

Trend analysis, or the study of how patterns permeate through a population, is an increasingly prevalent field of study in today's data-driven world. We used trend analysis techniques on the Yelp! dataset – a free information package provided by the Yelp! company – to find any discernable patterns in the way restaurant categories move over time. With this information, we can determine when and where a certain kind of restaurant will be built in the future.
It was found that, over the years, the times at which restaurants tended to be built occurred in a fairly linear fashion across many of the popular categories. Types of food observed include standard American fare, Mexican restaurants, Italian food, and Chinese food. Furthermore, restaurants of certain categories tended to stay clustered in one location, and generally did not cross certain boundaries close to other restaurants' clusters. There were also many issues that arose with the approaches taken in this study which are discussed further in.

I. Introduction

Trend analysis can be summarized as the process of finding patterns in data. These patterns may manifest as traits, such as finding that a pair of features is highly correlated or the distribution of items over space is non-random. In this vein, there are many applications of trend analysis, including climate studies [1] and the digestion of data collected by private web companies [2]. The techniques used to these ends may be applied to any dataset that exhibits non-random behavior, such as the one used in this study: the Yelp! Open Dataset.

This dataset is suitable for trend analysis due to it being represented in both time and space. Businesses are tracked by Yelp! using city, state, and coordinates, and reviews are tagged by time of creation. They are also further divided by the kind of business, which will be limited to restaurants for the purposes of this study. Finally, each restaurant may also serve a different kind of food, which will also be accounted for. Using these features, we can effectively monitor and predict the state of categories, showing where they hold the most influence and when they may be built next.

Spatiotemporal trend analysis is generally done separately, per component, as a general all-encompassing algorithm does not exist [3]. Therefore, we performed geospatial analysis first, then followed up with a predictor model for temporal patterns. A technique used for clustering events on a real-world map, the mean-shift algorithm [4], was implemented for the geospatial analysis. This gives clusters of restaurants in specific categories, scaled by number of restaurants in a certain area. To analyze the time series data, a regression was implemented to give a prediction of when restaurants are likely to be built.

With the implementation of these techniques, it was found that the categories of restaurants

analyzed in this study – American, Mexican, Chinese, and Italian – have their own spheres of influence which rarely overlap. Furthermore, the relation between the number of restaurants per category and time was found to be both correlated and fairly linear, giving way to a linear model for American restaurants. However, both approaches were found to have their issues regarding the available information, the optimality of the parameters, and the certainty of future behavior.

II. Data

The data used for analysis comes from the Yelp dataset [5]. This is a set of `JSON`-formatted files, where information is sorted according to a key and a corresponding value related to the key. In this manner, it behaves much like a dictionary or a map data structure. These files have information pertaining to many features of Yelp, such as reviews, business details, and photos. For the purposes of this analysis, emphasis is placed on the files that contain business details in `business.json` and the reviews left on Yelp for businesses in `review.json`. All analysis performed is done in Python.

Another dataset is also leveraged to represent the data geospatially. These are the shapefiles that detail the outlines of state counties, which are pulled from the US Census website [6]. These files contain the polygon math that is interpreted by the `geopandas` [7] library and displayed using `geoplot` [8].

The first step is to identify the necessary data and reformat it into the shape we need. There are many fields in both the `business.json` and `review.json` files, but we will only need a few of these for the purposes of trend-finding. The `pandas` [9] library for Python is able to load this data automatically, but due to the relatively large size of the `review.json` set, the data is discretized into chunks. `business.json` is small enough to not require this treatment, so we may keep all present fields. However, for the reviews, we selected only `review_id`, `business_id`, `text`, and `date`. When analyzing trends for patterns over time, the only important variable is the timestamp. In the case of the current analysis methods, the current assumption is that the earliest timestamp of a review is a good-enough indicator of the opening of a business.

Next, the space itself must be represented accordingly in data. Only regions as large as metropolitan areas should be considered for appropriate analysis, as a frame that is too large involves too many variables to accurately model with a dataset this small, and an area too small will not yield any meaningful conclusions. Therefore, the data is filtered to use cities that are well-represented in the dataset. The city with the most entries is Las Vegas, so we will use this city as the point of analysis; however, the methods outlined here generalize to any city in this dataset.

Since the data in `business.json` is spatially tagged with coordinates, we only need to convert this data into mappable form by using a `geopandas` dataframe. Finally, we can use the shapefiles and this geotagged data to plot both at the same time, resulting in a representation of restaurants in a city as shown in figure 1:

III. Methods

Spatiotemporal trend analysis is best accomplished by viewing each dimension as a separate component, i.e., to analyze space and time distinctly, then combining the results after the fact. For the purposes of this study, two main analysis methods will be used for time and space respectively to determine restaurant-spreading trends: the mean-shift clustering algorithm [4] and the Mann-Kendall Test for Monotonic Trend [10] [11] in conjunction with a linear regression.

The mean-shift clustering algorithm is used to determine likely centers of restaurant activity using calculated nodes. It is also well-suited for this problem because it relies on the natural parameter of scale to determine the number of clusters to calculate, rather than a fixed number of k-clusters, as most typical clustering algorithms implement. Therefore, we will get a reasonable number of clusters given our region

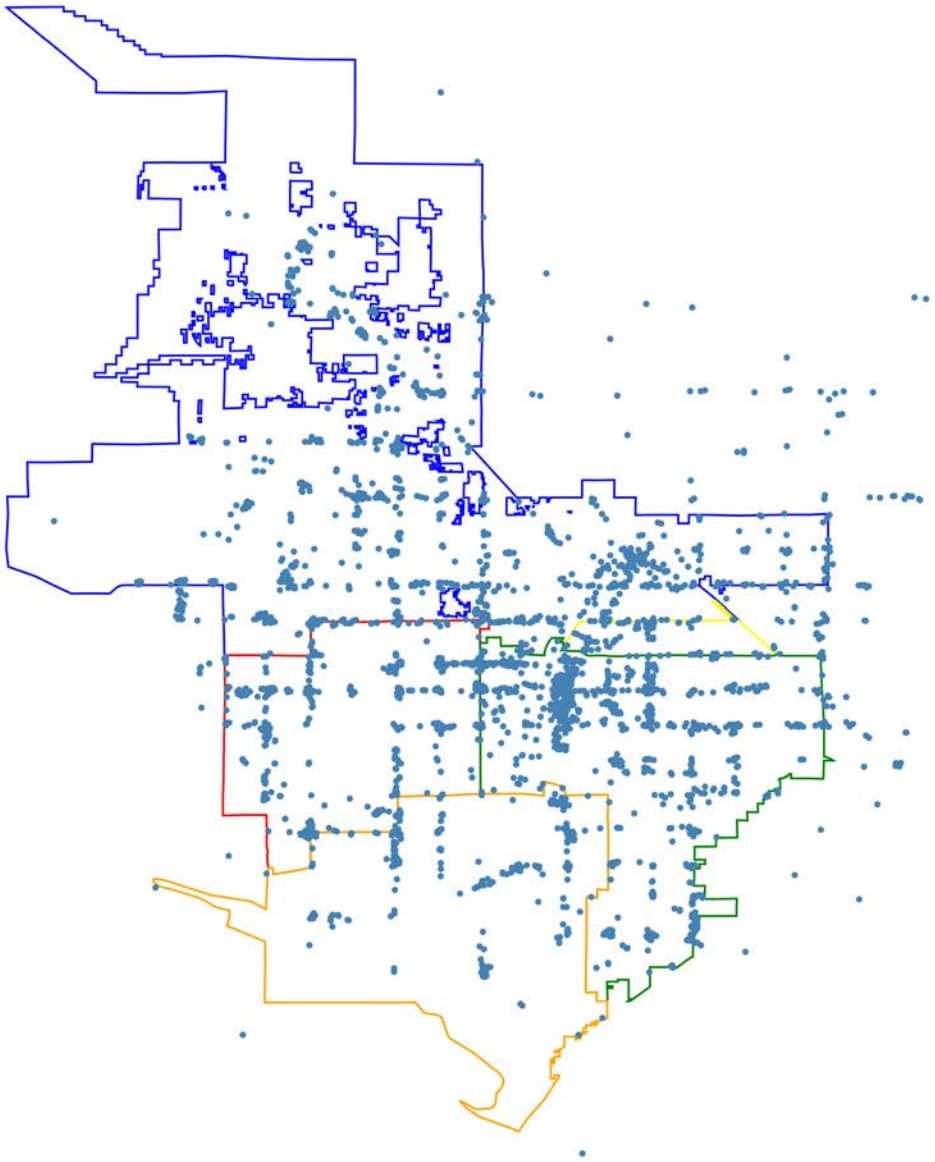


Figure 1: *Figure of visited restaurants in Las Vegas. Each point represents a restaurant.*

of interest being a metropolitan area, and from the clusters, gather a reasonable estimate of where the next restaurant may appear based on population density and lack of clusters in a given area.

The mean-shift vector is calculated with the following:

$$m_{h,G}(x) = \frac{\sum_{i=1}^{n} x_i g\|(x - x_i)/h\|^2}{\sum_{i=1}^{n} g\|(x - x_i)/h\|^2} - x,$$

where x_i are the data values, g is the weight for the data point taken from the kernel function G, h is the bandwidth parameter of the kernel, and x is the center of the kernel. We can use a uniform function for the kernel, much like [4] does.

The mean-shift sequence is then calculated using an initial location $x^{(1)}$:

$$x^{(i+1)} = x^{(i)} + m_{h,G}(x^{(i)}),$$

where $x^{(1)}$ is taken to be many random points on the map to generate the most representative clusters. In this implementation, the data values being used are the number of restaurants of a given category in an area around a point x_j.

To give the nodes some representative meaning, we will use the neighbors that fall within the bandwidth threshold to rank the mean shift in order of most likely to least likely for a new restaurant of that category to be built at that location. Using this technique will yield "hotspots" of restaurant trend activity; however, it does not necessarily imply any cause behind the trend. Therefore, predictions made using this technique must be careful, as although they would be informed, they would not be reasoned with confidence.

The next part of the analysis determines `when` a restaurant is likely to spread. For this, we utilize the Mann-Kendall Test for Monotonic Trend [10] [11], hereafter known as the MK test. It is most often used in conjunction with climate trend analysis over time [1], which is why it may be a viable analytic to consider when attempting to find if there is a temporal correlation within restaurant categories. As explained in [12], the null hypothesis of the MK test is that the data is independent and identically distributed, i.e. there is no observable trend.

The MK test uses the signs of the differences between data points to first calculate a value, S:

$$S = \sum_{k-1}^{n-1} \sum_{j-k+1}^{n} sgn(x_j - x_k), \tag{1}$$

where function

$$sgn(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0 \end{cases}$$

With S, the MK test statistic can be calculated as the following:

$$Z_{MK} = \begin{cases} \frac{S-1}{\sqrt{\text{var}(S)}} & S > 0 \\ 0 & S = 0 \\ \frac{S+1}{\sqrt{\text{var}(S)}} & S < 0 \end{cases}$$

The flexibility of the statistic implies that any value may be chosen to be analyzed for trending behavior; therefore, multiple statistics will be calculated to give a better idea of what is contributing to the spread of restaurants. First, we will use the population of an arbitrary area around a restaurant to calculate our x_j and x_k to discover how population in an area influences the spread of restaurants. Since it already seems likely that restaurants will naturally spread to more densely-populated areas, this value should be preprocessed to give a more accurate result. A second way to use the statistic would be to analyze how other restaurants influence the spread of one category of restaurants. In this case, the number of restaurants around the target category of restaurants will be used to determine the statistic. Finally, each of these statistics will be limited to the category of restaurant, but will also be used holistically for initial results. Calculating these values will give us an idea if there is a temporal trend to the spread of the

restaurants, which, when used in conjunction with our regression algorithm, informs us of *when* a restaurant trend is likely to spread to a new location.

Other considerations also come into play: since restaurants are likely to mirror the population of the area around them, we must be careful to control for the population in the areas we analyze. This is to ensure we are not making statements about the population instead of the restaurant trends. After the controllable factors are considered, we can then combine the results of the techniques to have a prediction of when and where a restaurant of a particularly category will be placed next.

Finally, we will use a regression to complete the analysis of the time series data after confirmation of the trending nature of restaurant growth. This will allow us to better predict its behavior over time, as we may use the model to both extrapolate into the future and quantify the past. After examining the data, a linear regression was selected to serve as a model, since it fit the data best. This model takes in the dates at which a restaurant was first reviewed and the number of restaurants at that time. The latter quantity is also scaled to represent the number of restaurants per population at the time step to accurately compensate for population growth over time, with data taken from the U.S. Census Bureau from their "Cities and Towns" population estimates [13] [14]. Finally, the model will be plotted to show how well it fits the data.

IV. Results

To apply our methodology, we will focus on three distinct categories in our data: "American (Traditional)" (213 restaurants), "Mexican" (149 restaurants), and "Italian" (100 restaurants). These categories were chosen due to their relatively high representation in the dataset and their specificity in detailing the exact kind of food to expect at these establishments.

The likely locations for restaurants to be built were calculated using the mean shift algorithm with an adjusted bandwidth of 0.025, which was selected due to the relatively sparse label set it returns and the scale of the data we're working with. By building the model using `sklearn`'s `MeanShift` class, we can begin to analyze and predict the locations of restaurants around Las Vegas, as seen in figure 2.

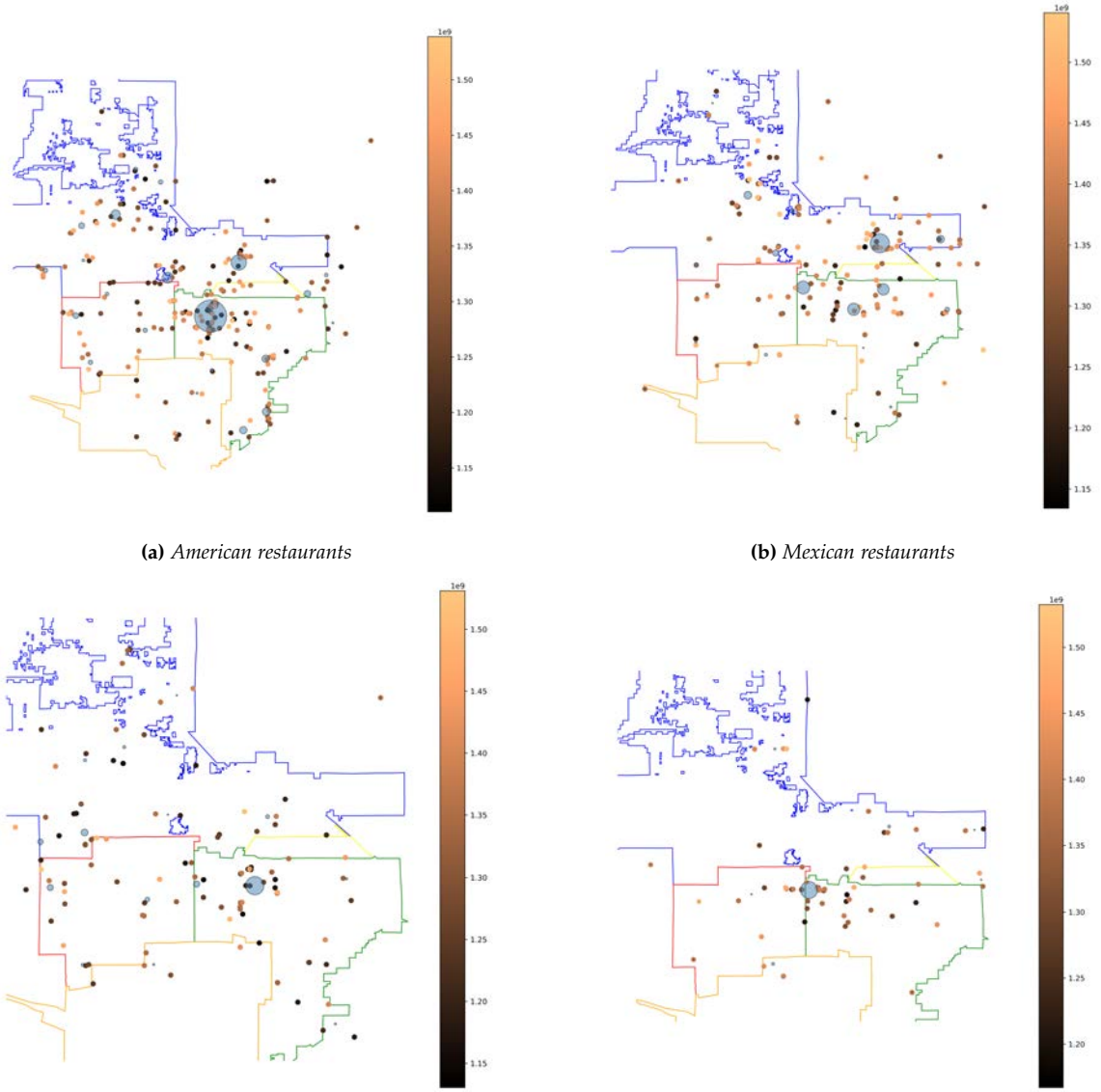
Taking figure 2a as an example, nodes have been plotted in blue at areas that have established restaurant presences for each category, along with the restaurants themselves in that category that were visited over time. The nodes, having been scaled to the number of neighbors in the bandwidth, represent the likelihood of a restaurant being placed at a specific location. A larger node implies a higher likelihood, which is why nodes are larger in denser areas.

Following these graphs, we can see that the area of highest likelihood is centered around the strip in the middle of the map. This follows the principle of agglomeration economies, where businesses are known to cluster in areas with (a) similar businesses and (b) many people [15]. We can also see outlets of activity around the strip and along highways that lead to the center.

Since the cluster sizes are scaled to the number of surrounding neighbors, there are many instances in which nodes exist but are not relevant when compared with the surrounding data. This has the effect of eliminating outliers from the graphs and allows us to concentrate on areas of highest likelihood to be further populated by restaurants.

In the analysis of the time series data, we first observe that the MK test results indicate that our data does exhibit a strong positive trending behavior, as evidenced by its Z-values of roughly around 4 (calculated values for American restaurants show to be $Z = 4.286$). Having confirmed this, we can then see in figure 3 that the plot of number of restaurants over time forms a relatively linear relationship. As a starting point, we can model this data using a basic linear regression to predict when restaurants may be built, resulting in the model at figure 4.

For instance, with regard to American restau-



(a) *American restaurants*

(b) *Mexican restaurants*

(c) *Italian restaurants*

(d) *Chinese restaurants*

Figure 2: *Calculated nodes for specific categories of restaurants, as well as the restaurants themselves. Small, lighter dots are recently-built restaurants. Blue nodes indicate centers of high restaurant activity for that category.*

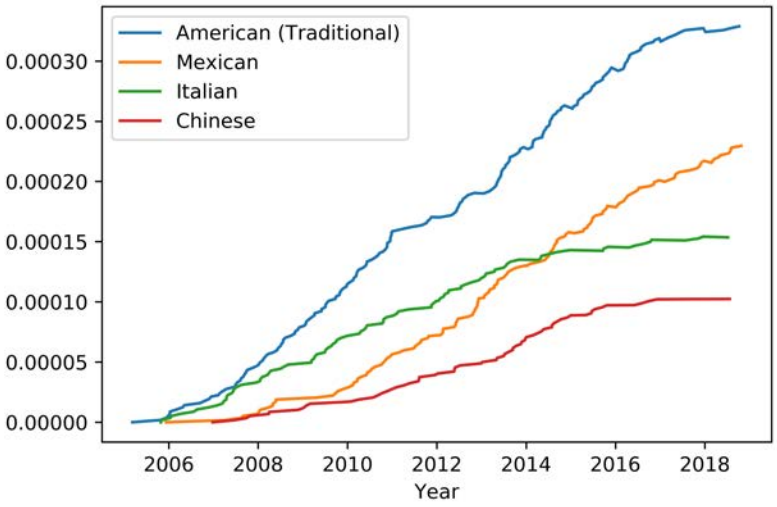


Figure 3: *Growth of restaurant categories scaled to population growth*

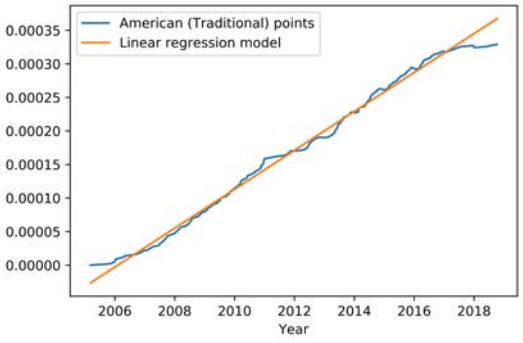


Figure 4: *Regression model using the number of restaurants per timestep*

rants, our model roughly has the form

$$y = 5.816e - 07t - 666.912$$

, where the coefficient represents the number of predicted restaurants per seconds since the UNIX epoch. Plugging in seconds since epoch for t gives us predicted number of restaurants at that time. However, this approach is taken with a fair amount of assumptions, as presented in the discussion section.

V. Discussion

i. Spatial Analysis

As shown in the generated graphs, each category of food has its own sphere of influence around the Las Vegas strip. For example, Chinese restaurants maintain a community mostly centered northwest of where many American restaurants reside. The four kinds of restaurants tested in general seem to each have their own major space with not too much overlap, with the exception of a strong American restaurant presence where many Mexican restaurants are as well.

A perk of using the mean-shift clustering algorithm is that there is no k-hyperparameter that the user must tune. This is because the scale parameter – the bandwidth of the kernel – is used instead to determine the number of nodes to use. However, this in itself is also a hyperparameter that we must consider and control throughout our analysis of the spatial distribution of restaurants. If this varies from run to run, we may end up with clusters that misrepresent the activity of restaurants in that area. This lead to testing of multiple bandwidths in order to determine a sufficiently

sparse set of clusters as to not overfit the algorithm, yet populated enough to display some relevant information.

Therefore, it is possible that the bandwidth is "unoptimized", as the tuning was done with a human in the loop. However, the calculated result is satisfactory, since it relates the information of interest, and the bandwidth value makes intuitive sense given the scale of the data.

One more property to note is that due to the scaling method used, the mean-shift algorithm is not kind to outliers; this is due to both the kernel bandwidth and the scaling metric. Due to the mean-shift algorithm's use of a kernel, the map is divided into chunks of certain size. Therefore, it is possible for nodes that may have been grouped together using a *k*-means algorithm to be uncounted for in the mean-shift, since, in the eyes of the algorithm, the outlier nodes fell outside of the kernel. Furthermore, there may be a more comprehensive metric than simply using the number of restaurants in bandwidth size to determine the size of the node.

ii. Temporal Analysis

As seen from figure 3, the growth of each restaurant category over time (adjusted for population growth) is highly positively correlated, indicating activity is relatively high for each category. Furthermore, it is worth nothing that the American and Mexican restaurants exhibit a relatively high pattern of linearity that matches the population growth, whereas the Italian and Chinese restaurants are tapering off in number, suggesting a lower demand. Therefore, a linear model would be a good match for the American restaurants, whereas Chinese restaurants may be more accurately modeled with a logistic regression.

The model itself found in figure 4 contradicts the tail end of the data found in years 2017-2018, where the number of American restaurants seems to taper off with respect to the population growth. This makes sense with regard to the nature of all populations, which follow a logistic pattern of growth due to the carrying capacity of a given area.

Regarding the this regression analysis, there are a couple issues regarding scope and assumptions. These trends could be modeled with a linear regression with moderate accuracy, but a linear model will grow to be less and less accurate over time due to the inherent maximum number of restaurants that can be present in a given area, factoring in physical construction space as an upper limit and the demand for each kind of restaurant. Therefore, a logistic model would be ideal for predicting the next time a restaurant will be built, but the accuracy of such a model depends heavily on external variables.

There is also the issue of Yelp! itself being the crux of the matter. It is very probable that, for some time, this data models the number of people on Yelp!. This can be seen towards the start of figure 3, where the number of restaurants is supposedly zero. This cannot be, and is therefore a result of Yelp!'s popularity at the time as a new app. Therefore, to further increase the accuracy of our model, data should only be considered after a certain time where Yelp! became more integrated into society.

VI. Conclusion

Using the spatiotemporal analysis techniques commonly found in climate studies and social networking sites, we found that restaurants do behave in a certain manner and tend to cluster in groups related to the kind of food they serve. The specifics of these interactions are also derivable using these techniques, as we can predict where and when a restaurant is likely to pop up. However, the applications of techniques were also found to be flawed, as there are many external variables that contribute to the shape of our results, such as popular topics at the time and the physical upper limit of restaurants a location can hold but given enough information, to name a few.

With the information from this study, consumers may have an easier time finding areas where restaurants they like are popular, busi-

ness owners may see opportunities to expand their region of influence or capitalize on the draw similar businesses in the area, and demographic analysts may use the results to further correlate with studies on ethnic, social, or financial enclaves of people based on what they eat.

References

[1] Pingping Luo, Bin He, Kaoru Takara, Bam H. N. Razafindrabe, Daniel Nover, and Yosuke Yamashiki. Spatiotemporal trend analysis of recent river water qualityconditions in japan., August 2011.

[2] Twitter. Twitter trends faqs. https://help.twitter.com/en/using-twitter/twitter-trending-faqs.

[3] Columbia University. Spatiotemporal analysis. https://www.mailman.columbia.edu/research/population-health-methods/spatiotemporal-analysis.

[4] David Crandall, Lars Backstrom, Daniel Huttenlocher, and Jon Kleinberg. Mapping the world's photos, 2009.

[5] Yelp. Yelp open dataset. https://www.yelp.com/dataset, November 2018.

[6] U. S. Census Bureau. Tiger/line shapefiles. https://www.census.gov/cgi-bin/geo/shapefiles/index.php, 2019.

[7] GeoPandas contributors. geopandas, 10 2019. [Online; accessed 2019-11-05].

[8] Aleksey Bilogur. geoplot, 11 2019. [Online; accessed 2019-11-05].

[9] Wes McKinney. Data structures for statistical computing in python, 2010.

[10] H. B. Mann. Non-parametric tests against trend, 1945.

[11] M. G. Kendall. Rank correlation metods, 4th edition, 1975.

[12] Pacific Northwest National Laboratory. Mann-kendall test for monotonic trend. https://vsp.pnnl.gov/help/Vsample/Design_Trend_Mann_Kendall.htm.

[13] U. S. Census Bureau. City and town intercensal datasets: 2000-2010. https://www.census.gov/data/datasets/time-series/demo/popest/intercensal-2000-2010-cities-and-towns.html, 2010.

[14] U. S. Census Bureau. City and town population totals: 2010-2018. https://www.census.gov/data/tables/time-series/demo/popest/2010s-total-cities-and-towns.html, 2010.

[15] Edward L. Glaeser. Agglomeration economics. https://www.nber.org/chapters/c7977.pdf, February 2010.

Comparing Yelp Ratings and Food Trends across the United States

Tetsumichi(Telly) Umada
University of Colorado Boulder
Tetsumichi.Umada@colorado.edu

Abstract

We analyze the Yelp reviews between the states to see whether there are any significant differences. The results show that there are no parasitical differences. However, the statistical test indicates there is a difference. Also, we analyze the trends of ramen and pizza restaurants over time and find increasing trends over the years.

I. Introduction

Is a restaurant in California better than the one in Colorado? When people travel to different countries or different cities, mostly they can have different types of cuisine. In the United States, many people originally come from different countries such as Europe, Asia, or South America. When they migrated, they also brought their own cultures including food. In particular, New York City is known as a place with ethnic diversities. Over the years, millions of immigrants entered through the city and over 800 different languages are spoken [1]. Because of the diversity of people and languages, there are many restaurants with various ethnic foods. By walking around a couple of blocks, it's possible to eat different types of authentic ethnic foods.

The increase in immigrants may not only create a diversity of food but also generate a trend of food. National food is considered as a food that forms the identity of a country and strongly associated with it. In the United States, hamburgers are popular and classific food, and it is considered as a national food [2]. On the other hand, some of the foods might become popular over the decades. For example, some restaurants such as shusi or ramen probably might not exist much in the United States. However, nowadays, it's more accessible and possible to eat them.

With the development of websites, nowadays, people can easily search for restaurants and check their reviews. For this project, we use the Yelp dataset that contains the reviews of the restaurants and their ratings. By using the data, we investigate if the restaurants in the particular state are better then the ones in the other states and analyze the trend of the food.

II. Data

For this data science project, we use the Yelp data set.[3]. The dataset is publicly available on Yelp Open Dataset. The data set includes the business data from ten states in the United States and three states in Canada.

Country	State	Count
Canada	AB	8008
Canada	ON	33422
Canada	QC	9223
USA	AR	1
USA	AZ	56712
USA	IL	1930
USA	NC	14722
USA	NV	36337
USA	NY	18
USA	OH	14698
USA	PA	11218
USA	SC	1160
USA	VT	2
USA	WI	5158

Table 1: *Business count per state*

Table 1 shows the overall distribution of the business data for each state in the United States and Canada. It contains 192,610 business, 21,719,148 reviews, 1,637,139 users, in total.

The downloaded dataset consists of five JavaScript Object Notation(JSON) files, one for each data object: business, review, user, check-in, and tip. Each line in the JSON file is an object. In business. JSON, a business is represented as a JSON object. The data contains business id, name, city, hours, is open, latitude, longitude, review count, stars, categories, attributes (such as parking, wifi, smoking). In the review JSON, a text review is represented as a JSON object which specifies business id, user id, stars, review text, date.

We pre-processed the data using python[4] and pandas[5]. There are mainly two data preprocessing steps. The business JSON file contains a name of cities where the business is located. However, many city names are misspelled and inconsistent (such as "Las Vegas, NV", "Las Vegass", "Las Vergas"). To correct the misspellings and keep the name of cities consistent, we use the Google Map API to decode latitude and longitude to reverse geocode and make the values of cities and states consistent. To eliminate the variances between the countries and focus on the restaurants in the United States, we use the data in the United States. Also, Arkansas (AR), New York (NY), and Vermont (VI) do not have many data points, so we remove them from the analysis.

After cleaning up the city names, we select the restaurants in the data. In the business data, there are many different types of businesses such as entertainment, fitness, and restaurants, etc. To eliminate the variances between the business and be able to compare within the same kinds of businesses, we focus on the restaurants. The data points are selected if the category contains the keywords, "Restaurants" or "food" in the category in case insensitive.

The pre-processed data contains the only restaurant businesses in the eight states in the United States. The number of restaurants in the selected dataset is 45,698.

III. Methods

In this paper, we investigate the Yelp stars to understand if there are any differences in the Yelp star ratings between states. We also see if there is a trend of some types of cuisine based on the number of reviews over the years.

To analyze the Yelp stars and validate the hypothesizes, we use statistical tests. Analysis of variance (ANOVA) is used to determine whether there are any statistical differences between the two or more means of groups [6]. It tests the null hypothesis, $H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$ where μ is group mean and k is the number of groups. We also use the Kruskal–Wallis H test that is used to test whether the median of the two or more groups are different.[7] It tests the null hypothesis, H_0: population medians are equal. We use $\alpha = 0.05$ to compare the p-value and evaluate the statistical test.

To analyze the trend of the particular food, we create a plot by taking a proportion by dividing the number of reviews by the number of the overall reviews for each year. The trend percentage would be helpful to limit some other related trends over time. Then, we perform χ^2 test that allows testing the associations between variables. To be able to perform a χ^2 test, we create a contingency table[8]. We use $\alpha = 0.05$ to compare the p-value and evaluate the statistical test.

IV. Results

In this section, we present the results for statistical analysis for each hypothesis.

The Yelp dataset contains names of the restaurants, their average star ratings, and cities, and states where the businesses are located. In the selected data sets, there are 45,698 restaurants in 8 different states including Arizona (AZ), Nevada (NV), Ohio (OH), North Carolina (NC), Pennsylvania (PA), Wisconsin (WI), Illinois (IL), South Carolina (SC). The overall average star rating is 3.48 out of 5, and the standard deviation is 0.85. The figure 1

shows the distribution of the overall star ratings in the selected data set. The star rating for each restaurant is rounded to half-stars.

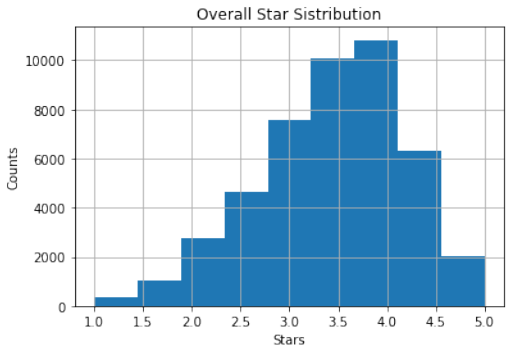


Figure 1: *Overall distribution of the Yelp starts*

We break down the overall Yelp rating starts to each state to investigate that the restaurants in one state are better than the ones in the other states. We compute the mean and standard deviations of the Yelp star ratings per state.

State	Mean	Std
AZ	3.4756	0.8560
IL	3.3881	0.8468
NC	3.4588	0.8492
NV	3.4892	0.8377
OH	3.4674	0.8639
PA	3.5532	0.8186
SC	3.4189	0.8832
WI	3.5159	0.8174

Table 2: *Yelp ratings per state*

By comparing the mean ratings per state and standard deviations in table 2, the means are almost similar to each other. We also perform a statistical analysis using ANOVA. The null hypothesis would be that there are no significant differences between the states' mean Yelp ratings. For the alternative, there is a significant difference between the states' mean Yelp ratings. F score is 8.51, and the p-value is 0. The resulting p-value is less than 0.05. We can reject the null hypothesis and conclude that there is a significant difference between the Yelp ratings for each state.

Although the p-value form the ANOVA shows that there is a statistically significant differences for star ratings between the states, we are not able to observe practical significant differences based on the means and standard deviations on the above table. To further investigate, we perform the Kruskal-Wallis H-test for the pairs of the states to see if there is a statistically significant difference. The results are shown in table 3.

State A	State B	Kruskal-Wallis	pvalue
NC	PA	29.9912	0.0000
AZ	PA	28.5636	0.0000
PA	IL	28.2531	0.0000
OH	PA	22.6426	0.0000
NV	PA	19.2568	0.0000
WI	IL	16.3354	0.0001
NV	IL	11.5944	0.0007
PA	SC	10.7679	0.0010
AZ	IL	9.4709	0.0021
OH	IL	8.7924	0.0030
NC	WI	7.4854	0.0062
NC	IL	6.1911	0.0128
WI	SC	5.7238	0.0167
AZ	WI	4.6870	0.0304
OH	WI	4.2839	0.0385
NC	NV	3.3527	0.0671
NV	SC	3.0001	0.0833
NV	WI	2.5897	0.1076
PA	WI	2.1788	0.1399
AZ	SC	2.1596	0.1417
OH	SC	1.9896	0.1584
NC	AZ	1.3585	0.2438
NC	SC	1.1339	0.2870
NC	OH	0.8816	0.3478
AZ	NV	0.8772	0.3490
OH	NV	0.7562	0.3845
IL	SC	0.5069	0.4765
AZ	OH	0.0124	0.9114

Table 3: *Kruskal-Wallis H statistics and p-value*

The above table shows the Kruskal-Wallis H statistics and p-value between the pairs of states, sorted by the p-value. The p-value for the first ten pairs of the states on the table are lower than the 0.05, so we can conclude that the star ratings for the some states such as NC and PA are significantly different, statistically.

To further analyze the data, we compare the restaurants serving the same type of foods in different states. Particularly, we picked restaurants serving ramen and pizza to compare the ratings between the states to see if there are any statistically significant differences. The table below shows the mean and standard deviations of the yelp ratings for each state.

Table 4 shows the mean ratings and their standard deviations of the ramen restaurants in each state. By comparing the mean ratings, Illinois is relatively lower than the restaurants in the other states. On the other hand, the ramen restaurants in Arizona and Nevada are relatively high ratings. The ANOVA returns 136.83 for F statistics and 0 for p-value. Since the p-value is less than 0.05, we can conclude that there are statistically significant differences. For pizza restaurants, the mean ratings for pizza restaurants seem to be very similar to each other as well as the standard deviations. The ANOVA results show that F statistics is 120.32 and the p-value is 0. Since the p-value is less than 0.05, we can conclude that there are statistically significant differences.

	Ramen		Pizza	
State	mean	std	mean	std
AZ	3.9355	0.4786	3.3779	0.7963
IL	2.8333	0.7638	3.2553	0.8222
NC	3.9444	0.5270	3.2542	0.8509
NV	4.0000	0.4160	3.2960	0.8592
OH	3.5000	0.7071	3.3224	0.8537
PA	3.5000	0.5477	3.1585	0.7940
WI	3.7500	0.2887	3.2632	0.7996

Table 4: *Yelp ratings for ramen and pizza restaurants*

For both pizza and ramen restaurants, it's possible to conclude that there are statistically significant differences. However, we are not able to observe the practically significant differences by comparing the mean and standard deviations.

We also investigate the trends of Yelp reviews in terms of the number of reviews and food categories. Yelp has been becoming popular for the past few years. As shown in the graph below, the number of reviews for the restaurants in the data set has been increasing over time.

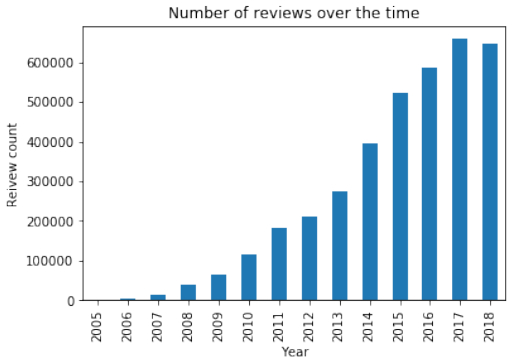


Figure 2: *Overall review counts over the years*

The figure 2 shows the total number of reviews for the restaurants in the data set. The Yelp website started in 2004. In 2006, the number of reviews is 3,727. It has exponentially increased. Then, in the year 2018, the number of reviews for the restaurants reaches 648,021.

To investigate the trend furthermore by breaking down it with a food category. In particular, we compare the restaurants serving ramen and pizza. We compute the proportion by dividing the number of ramen reviews by the total number of restaurant reviews to disclose the other related trends over the years.

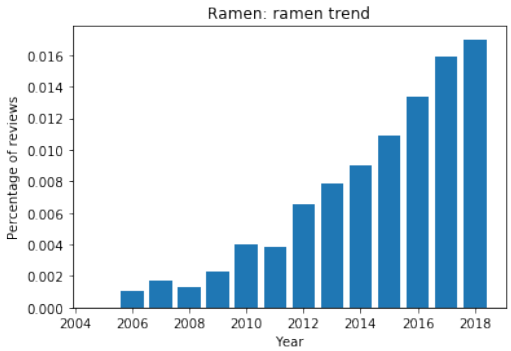


Figure 3: *Ramen trend*

Figure 3 shows the trend of ramen reviews over time. As shown in the graph, there are not many reviews in 2005. In the year of 2006, the Yelp users started to write reviews on the Yelp, meaning that the ramen restaurants are

	Year			
	2004-2008	2009 - 2013	2014 - 2018	total
Ramen	78	4860	38570	43508
Not Ramen	56831	845525	2772431	3674787
Total	56909	850385	2811001	3718295

Table 5: *contingency table for the ramen trend*

	Year			
	2004-2008	2009 - 2013	2014 - 2018	total
Pizza	4270	83739	306419	394428
Not Pizza	52639	766646	2504582	3323867
Total	56909	850385	2811001	3718295

Table 6: *contingency table for the pizza trend*

	Year			
	2004-2008	2009 - 2013	2014 - 2018	total
Ramen	78	4860	38570	43508
Pizza	4270	83739	306419	394428
Total	4348	88599	2811001	437936

Table 7: *contingency table for the ramen and pizza trend*

getting popular. Since then, the number of reviews on the Yelp has increased linearly as well as proportional to the overall yelp reviews. By comparing to the bar plot for the overall trend, the slop looks steeper and the counts are consistently increasing over the years.

To analyze the trend statistically, we perform a χ^2 test by finding the total number of reviews over the four years of the period so that we can create a contingency table, as shown in table 5. We perform χ^2_{tred} for trend analysis using the data in table 1, and it returns χ^2_{tred} value of 4152.04 which at df = 2 yields a p-value of 0. Since the p-value is less than 0.05, it indicates that the increasing trend of ramen reviews is statistically significant and likely to be true.

We also perform the same analysis for pizza restaurants to see if there is another kind of trend. To observe the trends over the years, we plot the proportion over the years by dividing the number of reviews for pizza restaurants by the total number of reviews.

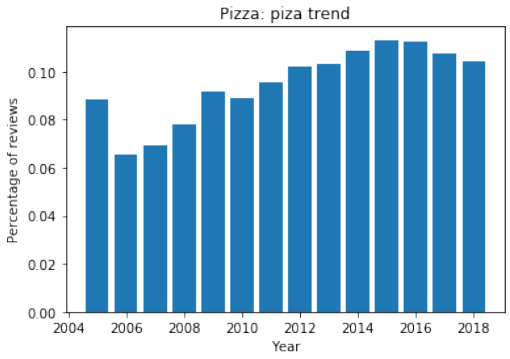


Figure 4: *Pizza trend*

As show in figure 4, the number of reviews for piazza also started gradually. In the year of 2004, there are 4 reviews regarding the pizza restaurants, but they increase the counts of reviews over the years. Comparing to the plot for the ramen trend, the trend increases gradually and fairly smooth over time. Between the years 2005 and 2018, on average, 2,8173.07 number of reviews (standard deviation is 2,6578.41) per year has been posted.

In the same way as the ramen trend, we

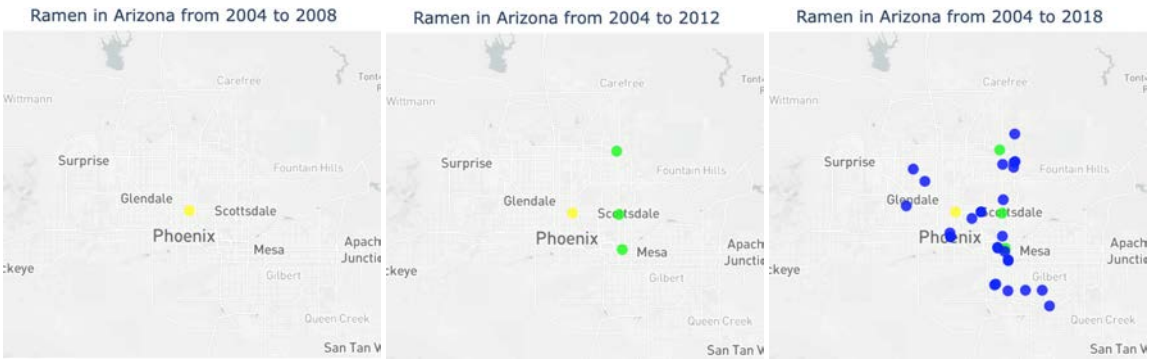


Figure 5: *Ramen trend in Arizona*

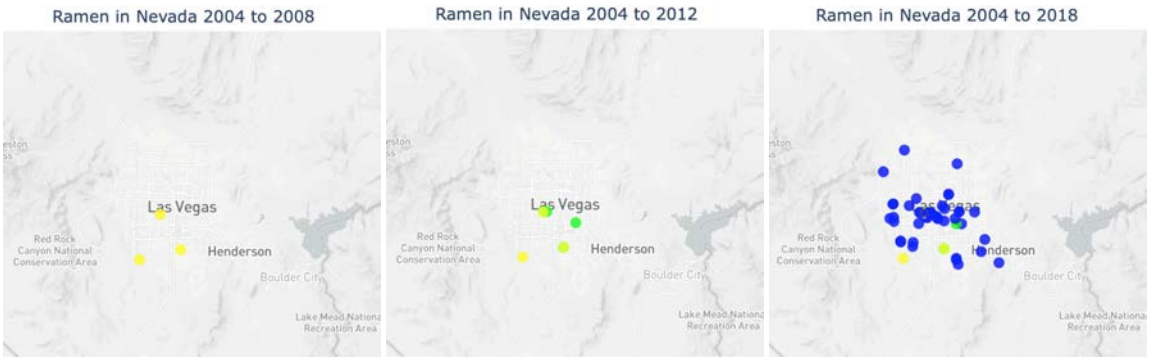


Figure 6: *Ramen trend in Nevada*

perform a χ^2 test for a trend by creating a contingency table for the pizza trend. Based on table 6, χ^2_{trend} returns the value of 4152.03 for the χ^2_{trend} which yields that the p-value of 0 at df = 2. Since the p-value is less than 0.05, we can conclude that the increasing trend of pizza reviews is statistically significant and likely to be true.

To compare between the trend of ramen restaursnts and pizza restaurants, we perform a χ^2 test for a trend using the contingency table 7. χ^2_{trend} value is 2880.21 which yields the that the p-value of 0 at df = 2. Since the p-value is less than 0.05, we can conclude that the increasing trend of ramen and pizza reviews is statistically significant and likely to be true.

After investigating the trend over the years, we also observe the trend in terms of the geographical locations for ramen and pizza restaurants. Table 8 shows the distribution of the ramen and pizza restaurants of each state in the dataset.

By comparing the number of ramen and pizza restaurants in each state, Arizona and Nevada have the highest number of restaurants for ramen. Also, both states have a sufficient number of pizza restaurants. We select those two states and plot their geographical trends over time.

State	Ramen	Pizza
AZ	31	1409
IL	3	94
NC	9	472
NV	53	848
OH	8	870
PA	11	780
SC	0	41
WI	4	190

Table 8: *Review counts for ramen and pizza restaurants*

Figure 5 shows how the ramen restaurants spread over the time in Arizona. In the dataset,

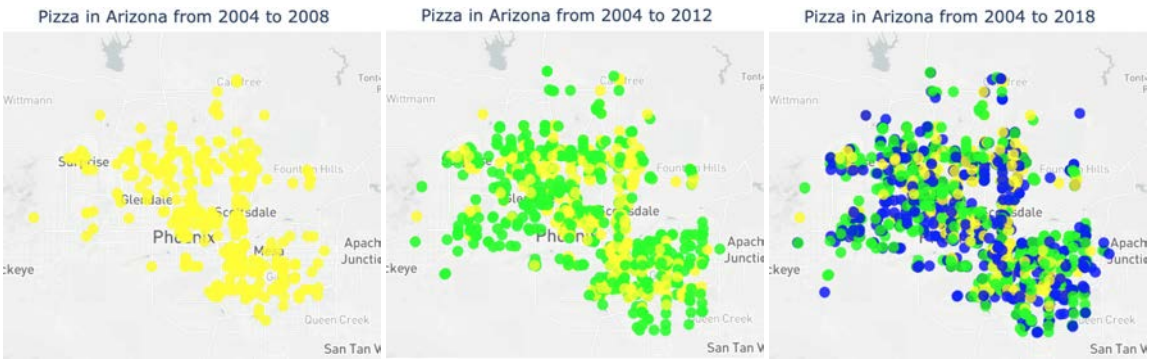


Figure 7: *Pizza trend in Arizona*

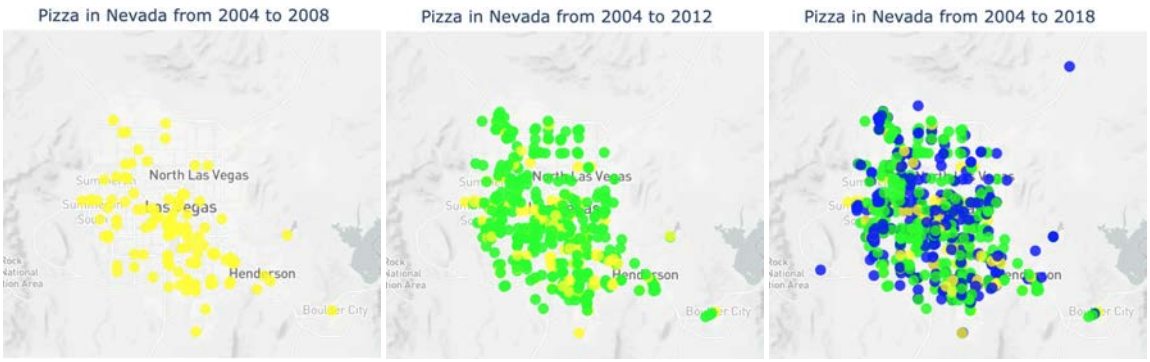


Figure 8: *Pizza trend in Nevada*

the ramen restaurants in Arizona are mainly Phoenix area. From 2004 to 2008, there are only a few restaurants near Phoenix area. As the time progress, the restaurants are started from the Phoenix and its surrounding area.

Figure 6 shows geographical trends of ramen restaurants in Nevada. Similar to the geographical trend of the reamen restaurants in Arizona, the ramen restaurants in Nevada is also started from the large city, Las Vegas. Then, over the years, the ramen restaurants increased gradually. Comparing to the plots for Arizona, the restaurants in Nevada are relatively close to each other. For the both plots in Nevada and Arizona, the ramen restaurants start in the center of the cities such as Phoenix, AZ or Las Vegas, NV.

To compare the geographical trends between the food types, we also plot the pizza restaurants to observe the geographical trend for pizza restaurants over the years in Nevada and Arizona.

Since the many pizza restaurants exist before the yelp started, the number of restaurants between 2004 and 2008 is higher. As a result, it looks that the restaurants are very close to each other. By comparing the spacey of the restaurants between Arizona and Nevada, the pizza restaurants in Nevada are located more close to each and they are mostly in the Las Vegas area. On the other hand, in Arizona, pizza restaurants are relatively far and spread out.

V. Discussion and Conclusions

In this project, we analyzed the yelp ratings between the states and trends of the ramen and pizza based on the Yelp reviews over the years. For the rating reviews, the ANOVA showed that there are statistically significant differences between the states. However, we were not able to observe the practical differences. By per-

forming the pairwise the Kruskal-Wallis H-test, some of the pairs between the states show the statistically significant differences.

For the trend of the food, we focused to analyze ramen and pizza to see if there is a trend over time. We plotted the trend by taking the proportions of the data. While the trend for ramen is more increasing, the trend for pizza is very flat and smooth. It indicates that the ramen is imported and becoming popular in the past few years. We also plotted to see the geographical spread of the restaurants. For both ramen and pizza restaurants, they tend to start from major cities such as Phoenix, AZ and Las Vegas, NV. However, it seems there are some other reasons that how far they spread from the center of the area.

The statistical analysis for this project is limited because the Yelp data set is a very small subset of the data set created from the Yelp website. By collecting larger sample data across the states and over time, we might be able to compare across the different states. For example, the ramen restaurants on the West Coast of the United States are better than the ones in the Midwest of the United States because immigrants from Asian countries more likely to live on the West Coast.

As future work, it would be more interesting to use the restaurant reviews to predict the economic states or demographics of the particular area. In a recent study, some researchers can characterize the neighborhoods [9]. Since the update on the Yelp is more frequent than the census data, it could be possible to predict or realize the small change of economics and demographics.

References

[1] Nigel Amaya. "How Many Languages Are Spoken in NYC?" WorldAtlas, Nov. 9, 2018. worldatlas.com/articles/how-many-languages-are-spoken-in-nyc.

[2] National Geographic. "Top 10 National Dishes" National Geographic, Sept. 13, 2011. nationalgeographic.com/travel/top-10/national-food-dishes/.

[3] Yelp. Yelp Dataset Challenge. 2019. yelp.com/dataset/challenge.

[4] Python Software Foundation. Python Language Reference, version 3.8. python.org.

[5] McKinney, Wes. Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 51-56 (2010).

[6] Heiberger, Richard M., and Erich Neuwirth. "One-way anova." In R through excel, pp. 165-191. Springer, New York, NY, 2009.

[7] Laerd Statistics. Kruskal–Wallis H Test using SPSS Statistics. statistics.laerd.com/spss-tutorials/kruskal-wallis-h-test-using-spss-statistics

[8] Hazra, Avijit, and Nithya Gogtay. "Biostatistics Series Module 4: Comparing Groups - Categorical Variable." Indian journal of dermatology 61, no. 4 (2016).

[9] Dong, Lei, Carlo Ratti, and Siqi Zheng. "Predicting neighborhoods' socioeconomic attributes using restaurant data." Proceedings of the National Academy of Sciences 116, no. 31 (2019): 15447-15452.