CJ Armbrust

Shion Guha

COSC 4931

11 April 2017

Fake News Write Up

My initial Fake News Write Up gave insight into how to approach the broad and unexplored issue. The dataset of this project has various column values that record numerical and textual values. After some exploratory analysis, I quickly learned that my dataset of all fake news articles required some cleaning. The most important data cleaning process was removing all of the articles that had an article type classified as "BS" because the BS Detector tool used this value when it could not specify the type of fake news. After this step, my dataset had close to 1,500 articles left which still left room for endless computation and inference. After executing additional exploratory analysis and gaining further insight into my dataset, it was time to begin to delve into some of the major algorithms and processes.

The textual values for each article are the title of the article, the website, and the text of the article. For a project that contains a massive amount of text, which is very important to answering some of the final research questions, it still requires processes that minimize the amount of text analyzed. If I could delete or isolate specific text, it would allow for more precise conclusions and classification. These findings would help answer questions such as what political topics of fake news penetrate the most users or which political party benefits most from fake news? So to inquire whether the fake news articles had significant political differences, I wrote different sections of code that search for the terms "Trump" or "Hillary" in the title or the text of the article.

```
for y in range(0, len(df)):
    s = str(df['title'][y]).lower()
    if "trump" not in s:
        df.set_value(y,'trump_title', '0')
    else:
        df.set_value(y,'trump_title', '1')
```

Figure 1: For Loop that traverses through dataframe and looks for "trump" in the article titles.

I plan to expand this to other political terms which will provide a stronger picture of the political tendencies of the articles. I also use similar concepts to record whether an article is popular based upon if its Facebook shares is greater than that of all of the articles in the dataset.

The code explained above had surprisingly successful results, in that nearly 200 articles had "Trump" or "Hillary" in their titles. In addition, nearly one-third of the articles mentioned either "Trump" or "Hillary." This reveals that a lot of the articles have direct political influence and could manipulate their reader's perceptions of the former presidential candidates. To see whether these articles with malice intent had a wide impact on readers, it is important to identify the social media and internet trends of these type of articles. The trends appear similar to the conclusions arrived in the first exploratory analysis, such that Facebook engagement has a strong correlation with domain ranking and an inverse correlation with spam score, although the sample size is relatively limited for articles with consistently high spam scores. I looked into other possibly correlation combinations between spam score, domain ranking, Facebook sharing, and Facebook likes, but this did not provide any information that was different from the initial exploratory analysis. In order to make use of these numeric metrics, I need to continue to break down the articles into smaller subgroups that can form distinct clusters. As a next step for this I will further examine what specific articles had high rates of social media engagement and

analyze their contents to infer whether specific content or themes penetrated networks more successfully.

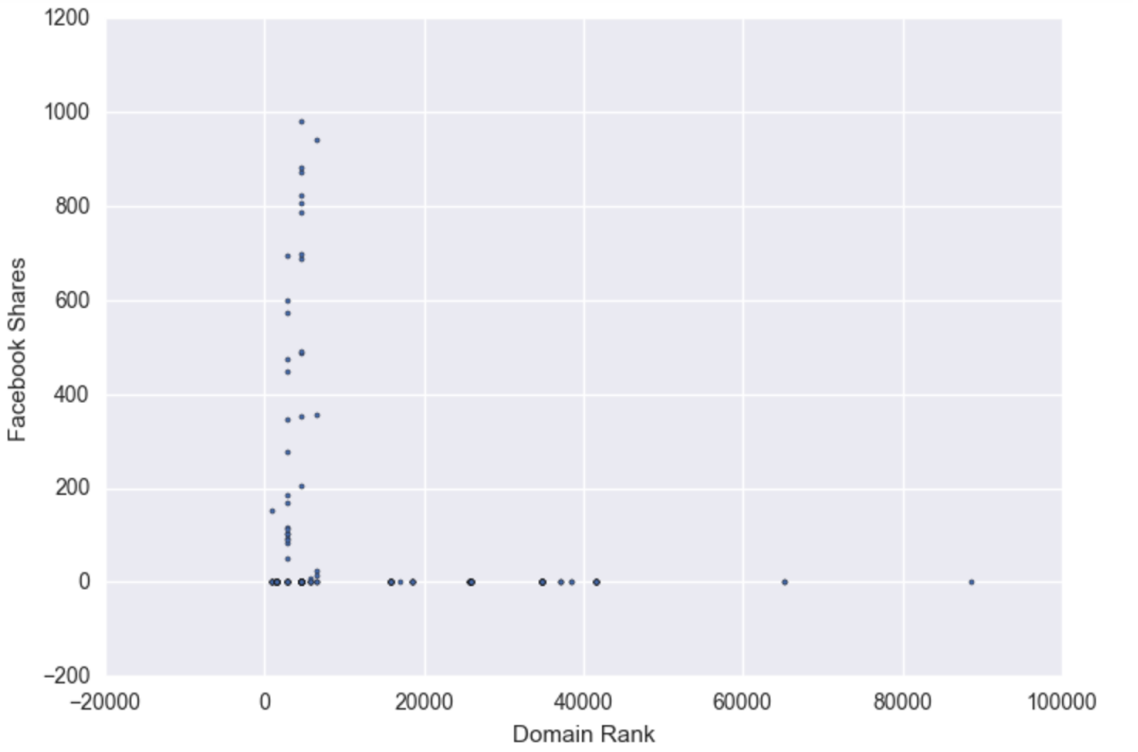

Figure 2: Graph of Facebook Shares vs Domain Rank for articles that discuss Trump or Hillary.

To proceed with the project, I needed to gather all of the articles with Trump and Hillary references in one location. So I used two matrices, one that stores all of the instances of when Trump or Hillary is in the title of an article and another that stores all of the occurrences of when Trump or Hillary is mentioned in the article text. This is the foundation for which I will run various Natural Language Processing algorithms to identify frequently used words, the tone of an article, and other factors that could help determine the article's focus. After running the Natural Language Processing Algorithms from the Natural Language Tool Kit (NLTK) or from Scikit Learn, I will be able to group articles into different clusters based on the aforementioned characteristics. As previously mentioned, forming more distinct and smaller clusters will allow for greater insight into what kind of articles and what topics penetrate users the most.

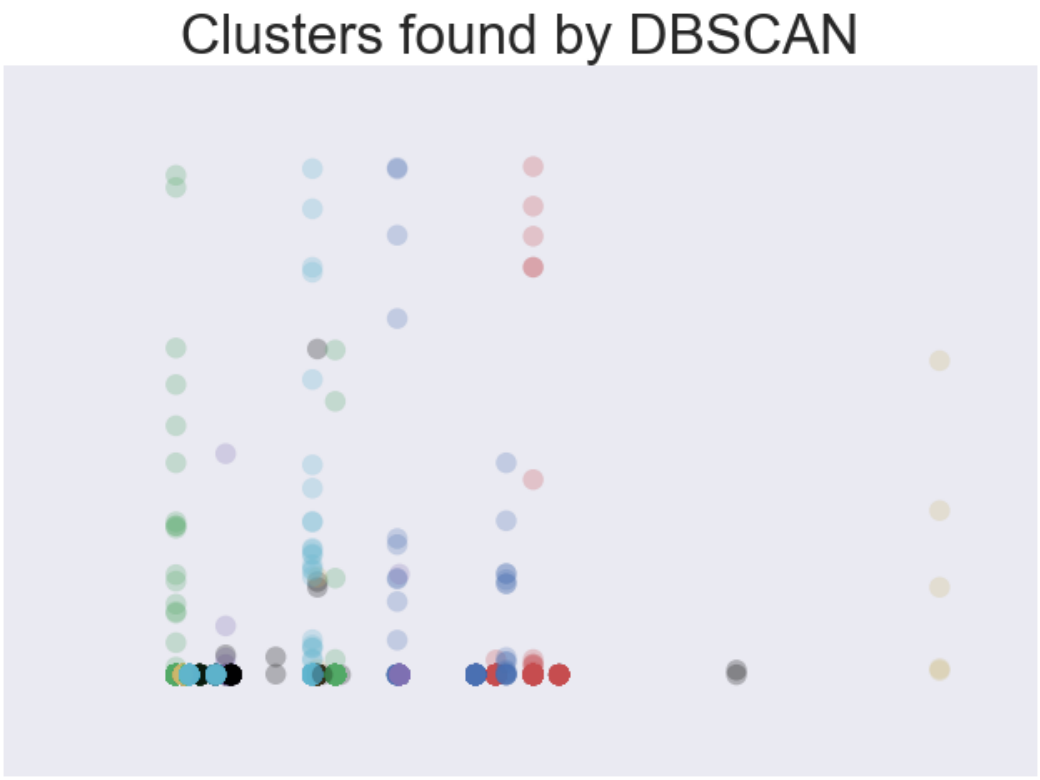

Figure 3: Density Based Clustering of 1500 articles

As seen above, the numerical values for the articles do not provide insightful clustering when using the DBSCAN algorithm. So NLP will be able to provide more useful clustering's because it will help further categorize article data. Once this is done, my final step is to run various classification algorithms that will try to guess whether an article is "popular" or not. This classification tree will use the results from the NLP algorithms and the numerical characteristics mentioned above. Essentially, the groundwork will be laid to infer what kind of articles in my dataset reach the most amount of users and how certain topics affected an article's popularity. This will help answer questions revolving around Fake News' relationship with political parties and political agendas, which is a first step in determining how Fake News had an impact on the 2016 Presidential Election.

Github: https://github.com/cjarmbrust1/cosc4931-introdatascience

References:

1. NLTK Documentation 3.0. Retrieved April 11, 2017, from Natural Language Toolkit: http://www.nltk.org/
2. Stanton, W. Latent Semantic Analysis. Retrieved April 10, 2017, from Data Science Association: http://www.datascienceassn.org/sites/default/files/users/user1/lsa_presentation_final.pdf