Ambadipudi, Chaitra (223009560) **Sharma, Aastikta (823007872)**

DETECTING GENDER BIAS IN NEWS ARTICLES

Motivation:

Newspapers are a strong source of data over the years, and more importantly, a timeline of opinions on how the society is functioning. They also help readers form opinions based on what they interpret from the readings. Within such an intricate web of articles, are there any unconscious biases hidden underneath, which may then be unconsciously perceived by the readers? This is what we aim to track. More specifically, we are looking at gender bias, and how it has changed over the years. This report aims at highlighting the choice of dataset, methodologies used, challenges faced and milestones achieved for the project.

Previous Work:

We were inspired to work on this domain after reading the following papers:

- Massive-scale automated analysis of news-content—topics, style and gender by Ilias Flaounasa, Omar Alia, Thomas Lansdall-Welfarea, Tijl De Biea, Nick Mosdellb, Justin Lewisb & Nello Cristianinia: This paper performed a gender bias detection in various news sources, based specifically on writing style. It helped us understand what parameters were looked at for detecting gender bias.
- Automating News Content Analysis: An Application to Gender Bias and Readability by Omar Ali, Ilias Flaounas, Tijl De Bie, Nick Mosdell, Justin Lewis and Nello Cristianini: From this paper, we gathered an idea of the techniques used in Entity Extraction and Gender Labeling.

Choice of Dataset: New York Times

The choice of dataset for this project is inspired from the fact that there are several articles being published on a daily basis in different categories and New York Times is one of the most widely accepted source of latest news. The source of data extraction will be the API that New York Times has made publicly available. The online version of the newspaper provides several APIs that are available here: http://developer.nytimes.com/docs.

We chose "Sports" and "Movies" categories to crawl articles. Our choice of categories was influenced by the fact that most of the articles published in these categories tend to get biased towards one gender. We assume in "Sports" it is male biased and in "Movies" female biased. But, in order to test if our assumptions are correct, we have decided to perform further analysis through information retrieval and visualization. However, it is due to these assumptions that we identified these categories that we wanted to use in the project and proceed with.

Data Crawling:

We created our own crawler (crawler.py) to extract data from the New York Times API. It crawls data from one year and two categories (in our case, Sports and Movies) up to 100 paginations per month, which is the limitation of the API. Thus, for every crawl we were able to gather at a maximum of 1000 articles for one month and one category.

Data Statistics:

Over a period of almost 10 years (2005-2016), we were able to crawl 103,185 data files for the Sports and Movies categories.

We noticed that for the year 2005, there were very few articles available and as the year increased number of articles increased exponentially. From a mere 25 articles in 2005, we were able to crawl 11,162 articles for 2015 under Sports category.

Pre-processing of Data:

The data crawled from website contained a lot of undesired information. Hence, we wrote a script to extract useful parts of it, including, but not limited to:

- Headline
- Snippet
- Persons
- Keywords like subjects/organizations

This information will be used in name entity recognition followed by gender labeling.

Natural Language Processing:

After getting the desired fields, name entity recognition was performed using the Stanford NER server. Every document is processed to extract the names from the articles. The first names are then used for gender labeling. We use the Python-SM library that consists of a huge database of names with over 40000 names and genders.

The first names found in a document are extracted and then passed on to this dictionary of words to correctly identify the gender of the person. The output can be either "male", "female" or "anonymous" (for the names not found in the dictionary).

After getting the gender labeling done, all desired fields are written into JSON format files which are required by Solr for indexing. The field names that make up the JSON files are "news_desk", "lead_paragraph", "word_count", "_id", "snippet", "subsection_name", "pub_date", "persons", "subject", "organizations", "male", "female" and "anonymous".

Solr Indexing:

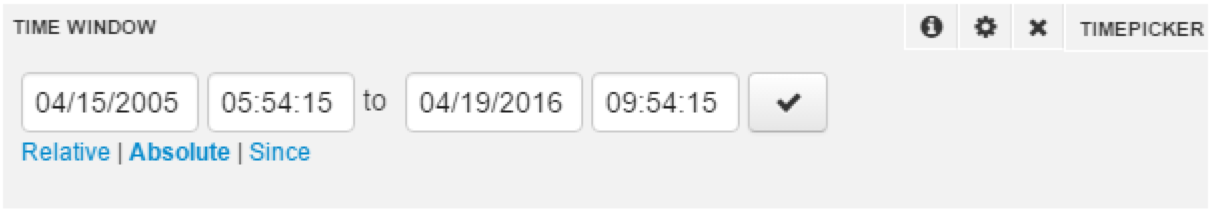
Before indexing the files onto Solr, various configuration files were created and/or updated to accommodate the files. A new workspace has been created on Solr with a new core to store the files. The basic schema was configured to suit the data from the JSON files. Using the command line options, we were then able to push (using bin/post) all the data files to Solr.

We checked if the indexing has been done properly by querying a couple of tasks like querying all the documents containing name "Ken", etc.

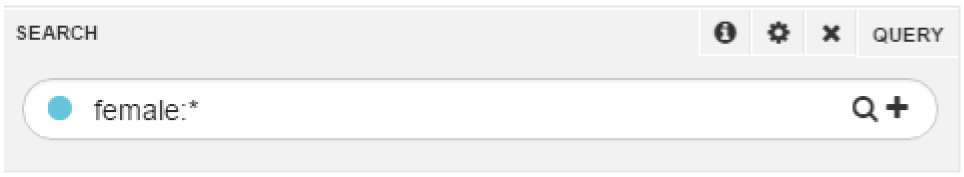
Results and findings:

We were able to index the data to Solr and plot it using Banana JS Framework dashboards. Our dashboard displays a couple of things which are described as follows:

- Time Window: It displays the time picker thereby making it flexible for the user to pick a time frame for which he wants to execute the query.



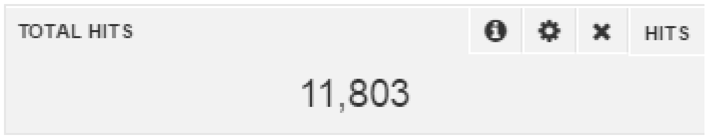
- Search: It is used for querying appropriately.


SEARCH

QUERY

female:*

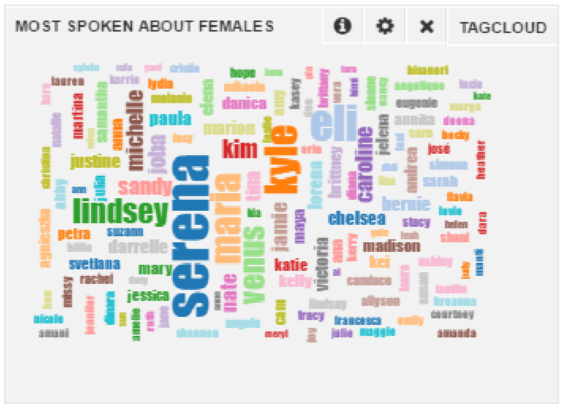
- Hits: The total hits is the number of documents found after the query.


TOTAL HITS

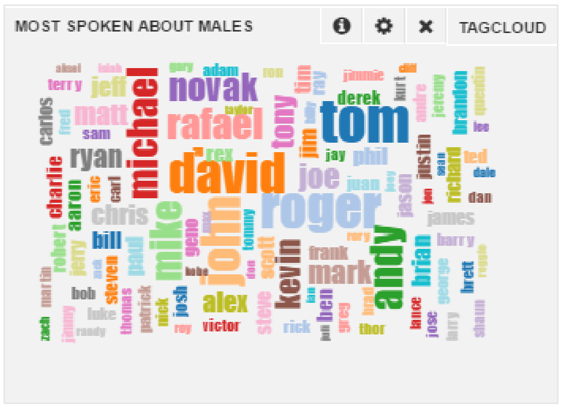
HITS

11,803

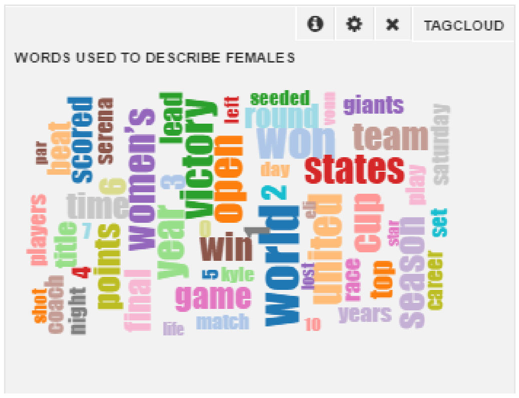
- Word Clouds: They display the most spoken males/females and most common words used to describe them.


MOST SPOKEN ABOUT FEMALES

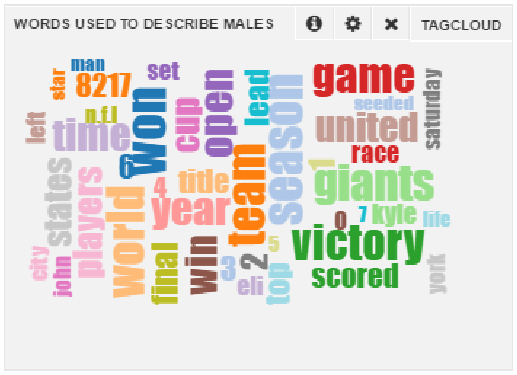
TAGCLOUD


MOST SPOKEN ABOUT MALES

TAGCLOUD

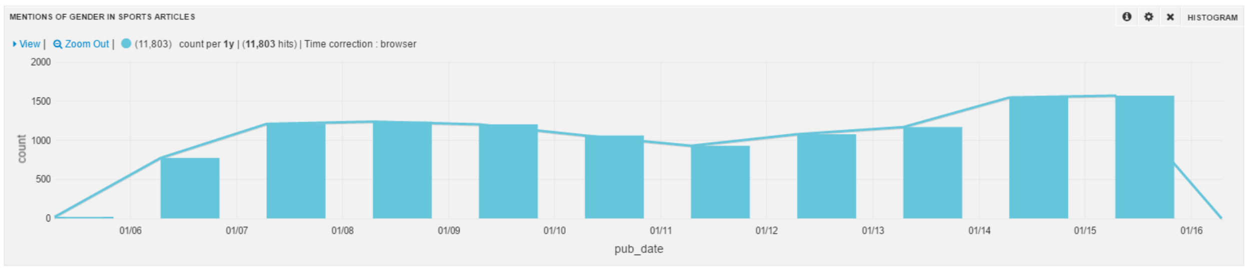

WORDS USED TO DESCRIBE FEMALES

TAGCLOUD

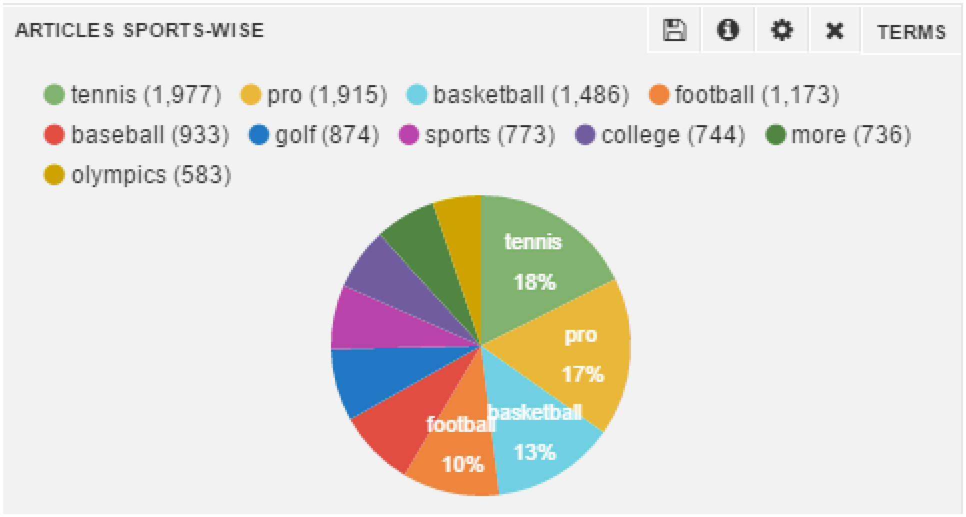

WORDS USED TO DESCRIBE MALES

TAGCLOUD

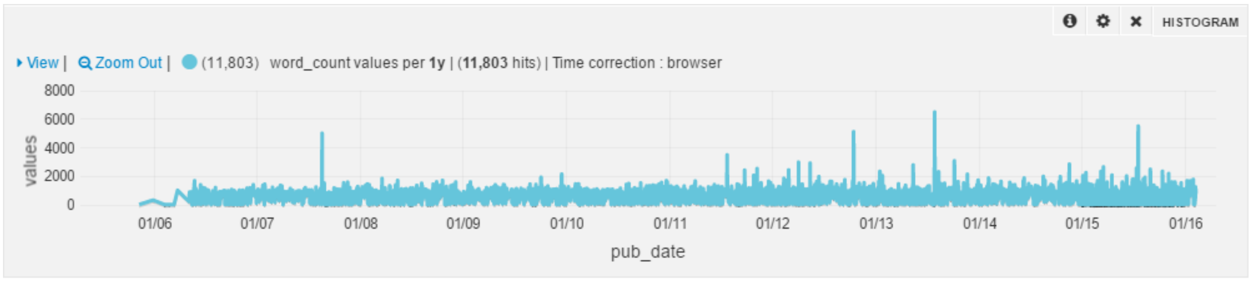
- Gender mentions in Articles: Histogram has been used to display the trend over the period of 10 years. We noticed that over 10 years, the number of articles highlighting women have increased but eventually they were never given the equal importance as men’s articles. We also noticed that during 2010 to 2012 the articles fell again for women.



- Pie Chart: We used these charts to highlight which type of sports were most talked about.



- Histogram: Another histogram has been used to display the total word counts used in articles over a period of 10 years.



Challenges and Solutions:

1. The first challenge was to identify a good exhaustive dictionary of names and gender that accommodates a large number of names with their genders. We finally managed to find a good dictionary called SexMachine(SM) library to suit our needs.
2. The second challenge was to merge the entities provided by default by the NYT API with the entities extracted by the Stanford NER server. We managed to do a basic consistency check by looking for pattern matches in the first and last names of a person, and then merging similar ones.
3. Next, while running the python code to extract names and then perform gender labeling, initially took almost 6 hours for 6 months of data for a year. To resolve this, we created a single instance of the dictionary and called that instance over all the years. This reduced our run time drastically from hours to a couple of minutes.
4. Another challenge, while indexing documents to Solr, was with the schema setup. Initially, there was a managed schema which Solr was using by default to index the data. Since the managed schema could not properly capture all the indexable fields during our JSON data, we were unable to query the documents successfully. Once we created our own schema and configured Solr to use it, we were able to push the data files and query them.
5. Since we didn't come across any documentation regarding the paginations limit per month, we were able to get only 10 articles per month (from the initial pagination). Later we identified that every page has a display limit of 10 articles and we had to increase pagination to 100 to be able to collect all possible articles.
6. While representing data using Banana dashboards, we noticed that the browser was not clearing the cache of the previously stored dashboards. Hence, any changes which were made to the dashboard was not visible even after reloading the page. After reading a couple of articles on Banana and Solr, we finally figured out a way of clearing the dashboard cache, by storing the latest changed dashboard as json and uploading it to replace the existing dashboard, to display the latest changes upon reloading of the page.

Conclusion:

We were able to identify the various trends over a period of 10 years by visualizing properly using Banana and Solr indexing. In future work, it can be extended to add more 'stereotypical' categories such as business, politics, and entertainment, and 'neutral' categories like top stories, or front page stories.