Media Analysis of Black People & Black Communities in Boston

NAACP - Boston Chapter

Rayyan Nasr, Mahmoud Khalil, Stephanie Forbes, Mani Singh, Dingjie Chen, Manish Patel, and Zheng Hui

December 14, 2020

Table of Contents

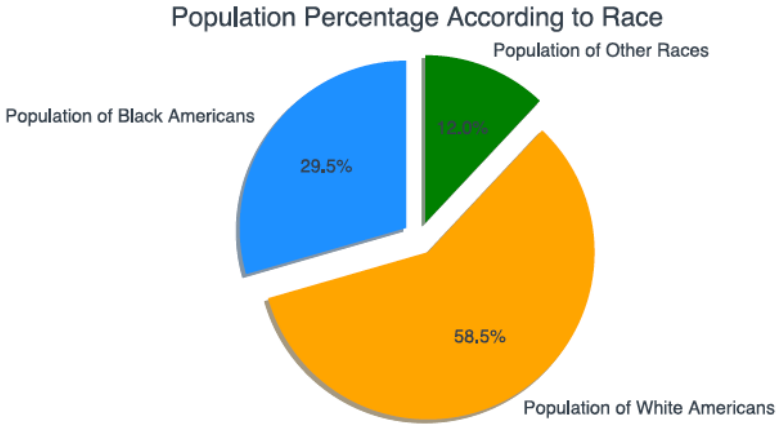
** All code resides at the following repo: https://github.com/alt113/naacp-spark-fall2020*

** All web scraped data resides in the following google drive: https://drive.google.com/drive/u/2/folders/19U0stNPFcuE8EkqnQbSfqQ_E-q0TOFWK*

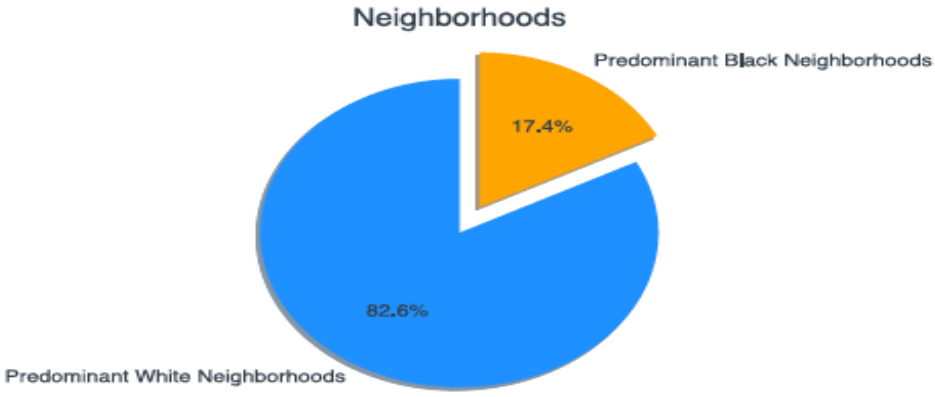
1- Exploratory Data Analysis:

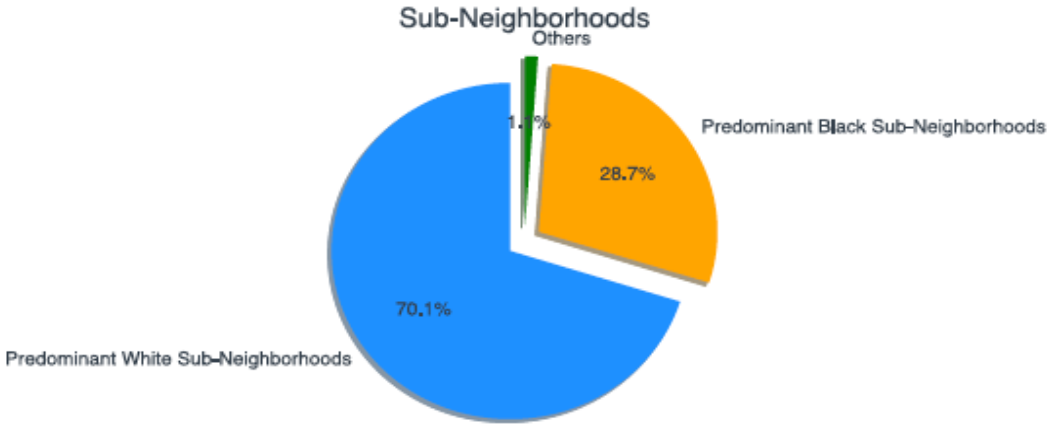
A - Census Data:

The revised census data provided us with a view of racial segregation across the Boston area. Initially we identified the population of Black Americans, White Americans and Other races. It was determined that White Americans constitute 58.5%, while Black Americans 29.5% of the total population. The remaining 12% constitute of American-Indians and Asians. This in turn gave us an understanding of how we can interpret the data we retrieve later in our exploration.

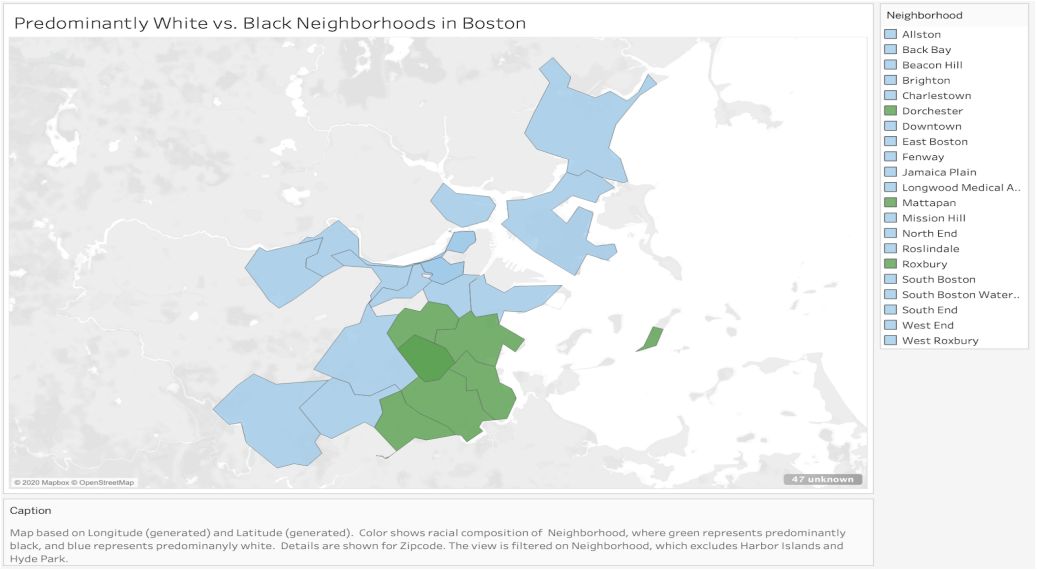


Next, to further understand the segregation. Analysis was done on both neighborhoods and sub-neighborhoods. Out of the total 23 neighborhoods in the area, 19 were found to be predominant White neighborhoods, while the remaining 4 predominant Black neighborhoods. Similarly, out of the 87 sub-neighborhoods, 61 were found to be predominant White sub-neighborhoods, 25 predominant Black sub-neighborhoods, and 1 predominant Asian sub-neighborhood.





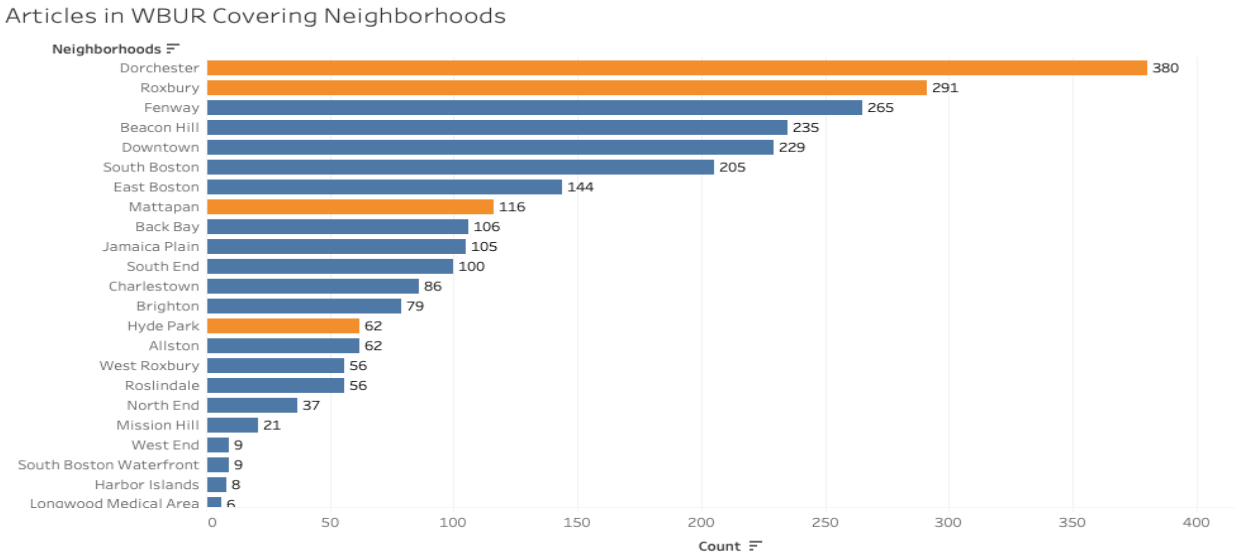
The following map depicts predominantly black and white neighborhoods in Boston, with blue representing predominantly white, while green represents predominantly black neighborhoods.



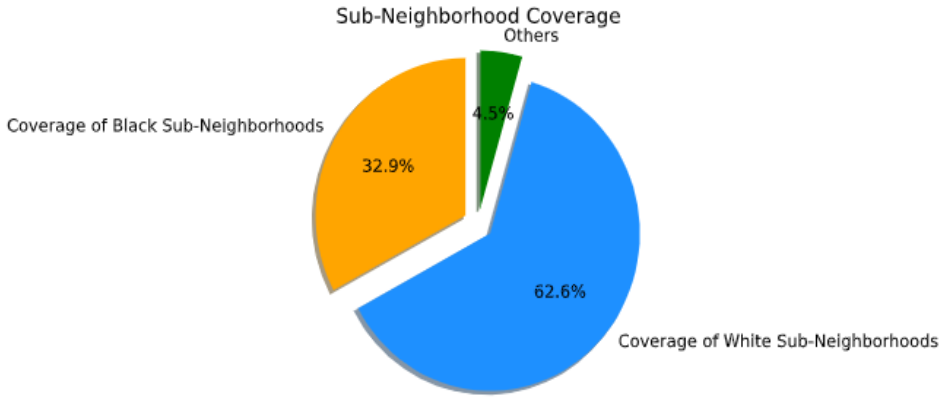
B - WBUR and Census Data:

After aggregating the data for all the five years for the WBUR, a preliminary analysis was done to identify the relationship between the census data and the articles. A total of 12,656 articles were identified between the years 2014 – 2018. Between these years, a total of 1818 articles were about white neighborhoods, and they covered all 19 white neighborhoods. The most prominent white neighborhoods covered were Beacon Hill, Downtown, East Boston, Fenway, Jamaica Plain, South Boston and South End. As for the Black predominant neighborhoods, 849 articles were about Black neighborhoods, and the coverage was spread throughout the 4 predominant black neighborhoods. Therefore, predominantly Black neighborhoods were subject

to 31.8% of news articles that had a geographic mention. Thus, it was observed, the coverage of predominantly Black Neighborhoods compared to predominantly White Neighborhoods was coherent with percentage population (29.5%)

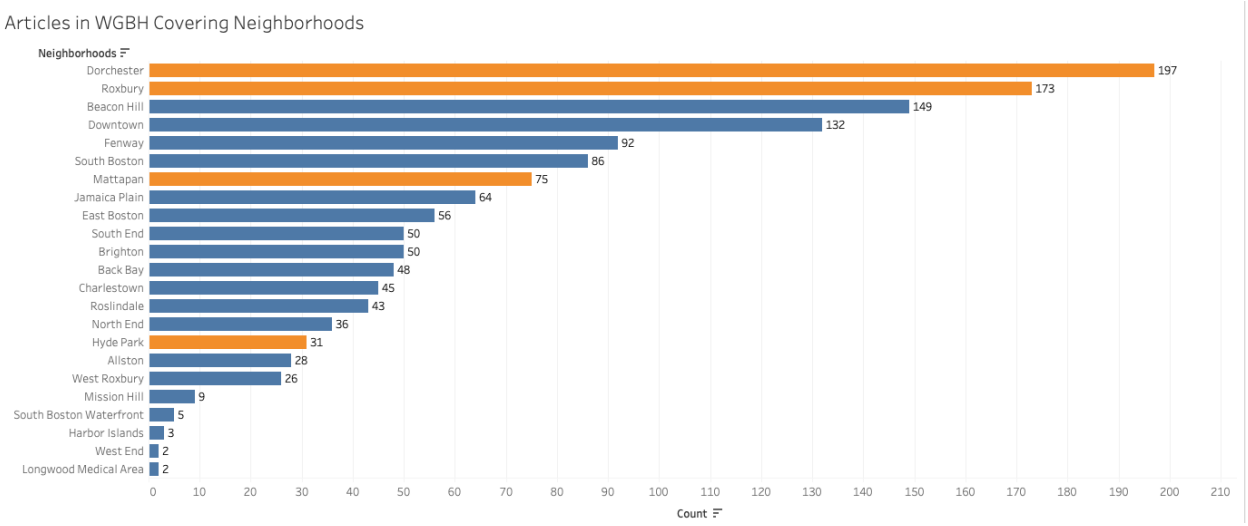


Even though this was the case, we believed that exploring the coverage of sub-neighborhoods will give us an even better understanding. Between the years, there were 1673 articles covering white sub neighborhoods. It was found that out of the 61 predominant white sub neighborhoods 33 sub-neighborhoods were covered. The most prominent white sub-neighborhoods covered were Allston, Brighton, Charlestown, Columbus, and Roslindale. Similar analysis was done for the black sub-neighborhoods, there were 878 articles covering the sub-neighborhoods, and out of the total 25 predominant Black sub-neighborhoods only 9 were covered. The most prominent black sub-neighborhoods covered were found to be Dorchester, Hyde Park, Mattapan, and Roxbury. Predominantly Black sub-neighborhoods were subject to 32.9% of news articles that had a geographic mention, as seen in the figure below.

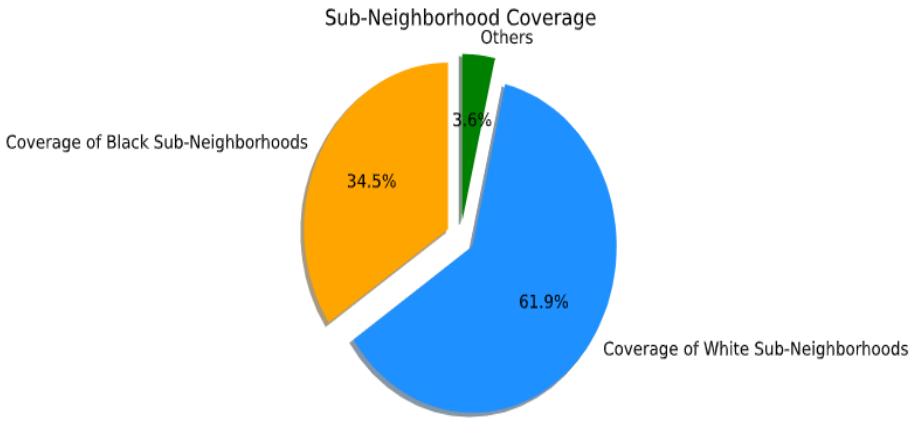


C - WGBH and Census Data:

Similar to what was done using the WGBH scraped data set, an analysis was done to identify the relationship between the census data and the articles. A total of 8,324 articles on local news were identified between the years 2014 – 2018. Between these years, a total of 926 articles were about white neighborhoods, and this covered all 19 white neighborhoods. The most prominent white neighborhoods covered were Beacon Hill, Downtown, East Boston, Fenway, Jamaica Plain, and South Boston. As for the Black predominant neighborhoods, 476 articles were about Black neighborhoods, and the coverage was spread throughout the 4 predominant black neighborhoods. Therefore, predominantly Black neighborhoods were subject to 33.6% of news articles that had a geographic mention. The results were very similar to the ones obtained from the analysis done on the WBUR.



The same process was done on the sub-neighborhoods. Between the years, there were 883 articles covering white sub neighborhoods. It was found that out of the 61 predominant white sub-neighborhoods, 23 sub-neighborhoods were covered. The most prominent white sub-neighborhoods covered were Allston, Boston, Brighton and Charlestown. Similar analysis was done for the black sub-neighborhoods, there were 492 articles covering the sub-neighborhoods, and out of the total 25 predominant Black sub-neighborhoods, only 8 were covered. Predominantly Black sub-neighborhoods were subject to only 34.5% of news articles that had a geographic mention, as seen in the figure below. We concluded even though the amount of articles found on the WBUR is larger than that found on the WGBH during these years, the results were very similar on every aspect. This in turn gave us a solid understanding of our data sets so we can further analyze them.



D - EDA Code Explanation:

The results above, can be reproduced from reproduced from an ipython notebook located under the following directory:
https://github.com/alt113/naacp-spark-fall2020/blob/main/EDA%20Census:%20Articles/EDA_NAACP.ipynb

The code consists of three main parts: The first part deals with analyzing the revised census data, while the second and third applies this analysis on both the WGBH and WBUR data sets.

Analysis of census data:

Initially, after reading the csv file we perform data cleaning in order to remove % signs, cast all columns as floats, and deal with NaNs. Then, after determining the population proportion according to race, we move on to determine the predominant white and black neighborhoods using the following method:

```
# Creating a list of the Predominant white neighborhoods
# ------------------------------------------------------
w_n = new_df_n.groupby('Neighborhood', axis= 0)['White Proportion'].mean()
w_n = w_n.to_frame()
c = w_n[w_n['White Proportion'] > b_n['Black Proportion']]
print(c.count())
w_neighborhoods = c.index.tolist()
w_neighborhoods
```

This will give us the predominant white neighborhoods (white proportion greater than 50%). The same method was used to determine the predominant white and black sub-neighborhoods. The results were then visualized using a pie chart.

WBUR and Census Data Analysis:

The next step is to analyze the coverage of neighborhoods and sub-neighborhoods. The web-scraped articles were scraped year by year, so after concatenating all five years of articles for the WBUR, we cleaned the data to remove any special characters that are found in the text. We also lowercase all the words as seen in the code below:

```
# Cleaning the data set
# ----------------------

# Special characters
spec_chars = ["!","'",'"',"#","%","&","'","(",")",
              "*","+",",","-",".","/",":",";","<",
              "=",">","?","@","[","\\","]","^","_",
              "`","{","|","}","~","—", "\xc2", "\xa0",
             "\x80", "\x9c", "\x99", "\x94", "\xad", "\xe2", "\x9d"]

# Removing Special characters
for char in spec_chars:
    data_frame['text'] = data_frame['text'].str.replace(char, ' ')

# Lowercase all the words
data_frame['text'] = data_frame['text'].apply(lambda x: x.lower())
```

In order to determine the white neighborhoods that were mentioned as well as the frequency of mentions, the following method was used:

```
# Finding the white neighborhoods that were mentioned
# ----------------------------------------------
data_arr = data_frame['text']
# -------------------------------------
wn_df = pd.DataFrame(columns =['Subs'])
wn_df['Subs'] = w_neighborhoods
# -------------------------------------

mentions=[]
for neighborhood in wn_df['Subs'].str.lower():
    count = 0
    for article in data_arr:
        if neighborhood in article:
            count += 1
            continue
    mentions.append(count)

mention= pd.DataFrame(columns =['Count'])
mention['Count'] = mentions
print(mentions)
print(mention.sum())
```

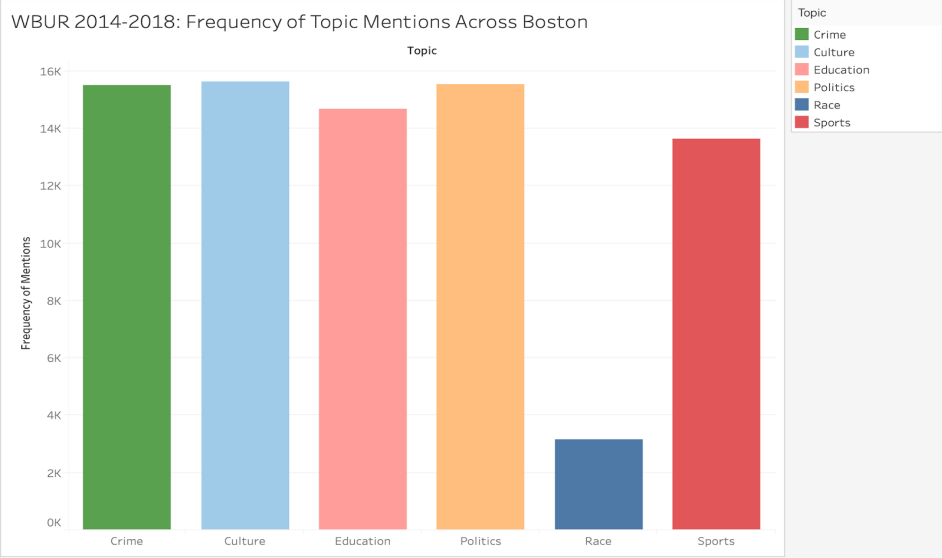
This was done using a simple nested for loop, where we iterate through every predominant white neighborhood, and for every neighborhood, we iterate through every article, if the neighborhood is mentioned, then a counter keeps track of it. This will provide us with the respective frequency of mentions for each neighborhood. Similar analysis was done for the remaining neighborhoods and sub neighborhoods. As well as the analysis done on the WGBH data sets.

2- Topic Modeling:

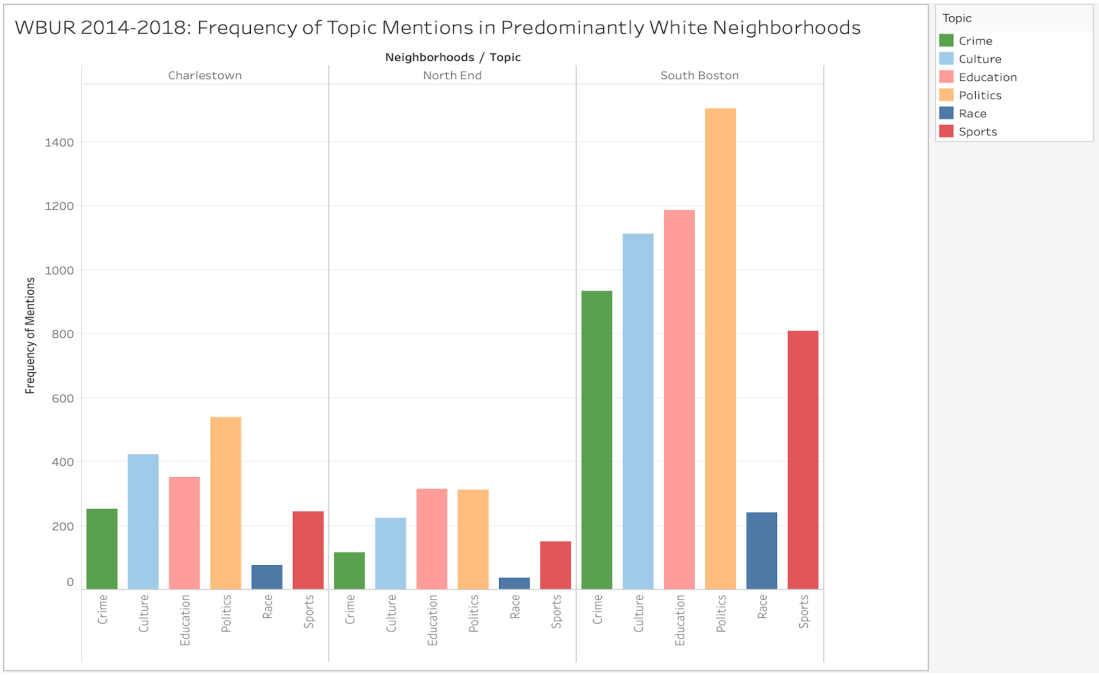
A- WBUR: Topics Covered

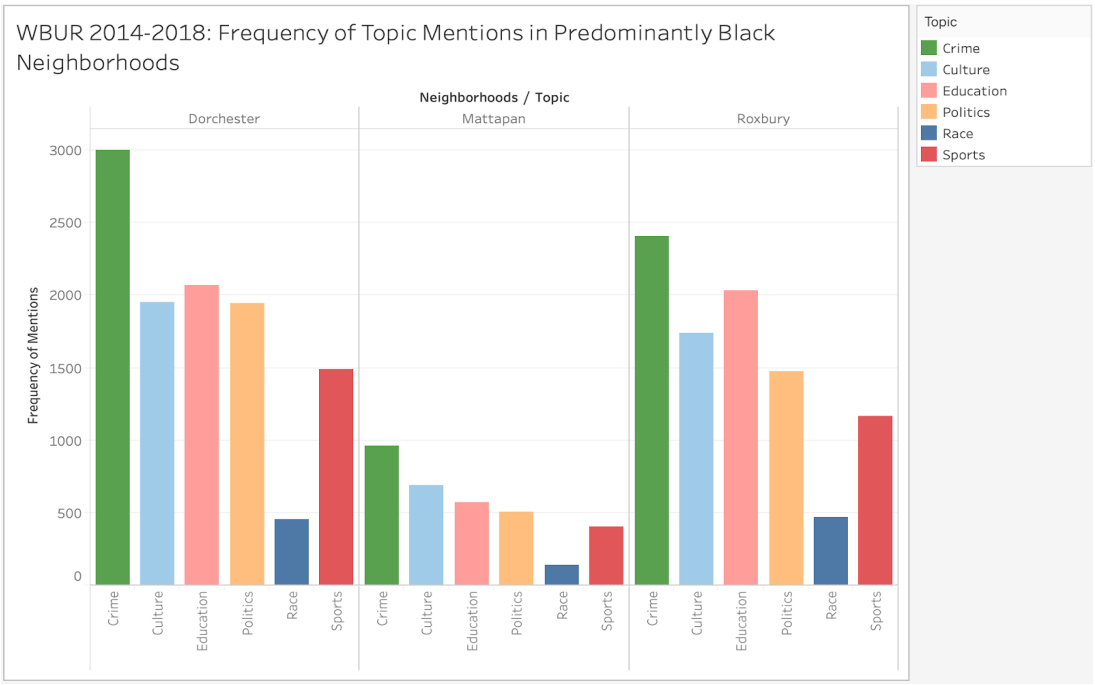
What topics are being covered in predominantly white versus predominantly black neighborhoods?

In order to look at what topics were being covered in different neighborhoods of Boston, we used a simple model to look for mentions of both the neighborhood in question, and various keywords for several different topics. We looked at crime, culture, education, politics, race, and sports as topics, and attributed 30 unique keywords to search for in each article to be considered a "mention". We used LDA and NMF to influence some of the keywords that we attributed to these topics. The histogram below shows the distribution of mentions across these topics. Note that some articles can have multiple topics attributed to them in the model that we had created.



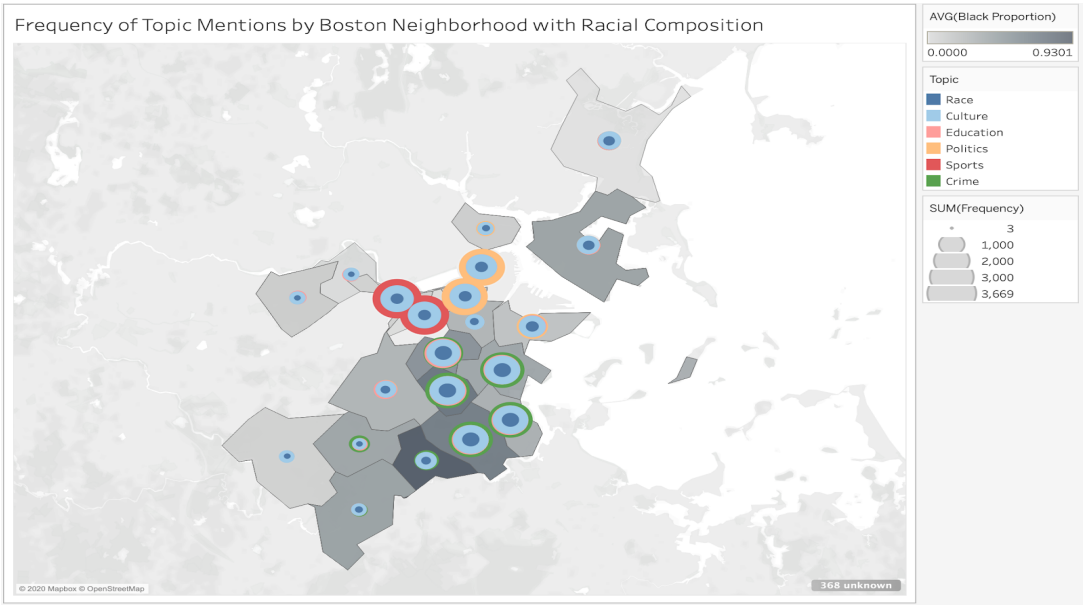
The top 3 predominantly black neighborhoods are Dorchester, Mattapan, and Roxbury, while 3 of the largest, most predominantly white neighborhoods are Charleston, the North End, and South Boston. We filtered out these neighborhoods in particular to look at the topic mention breakdown more specifically to see if there was, in fact a difference in the topics being covered and the frequency among them. The next figure shows the breakdown for Charleston, the North End, and South Boston followed by the breakdown for Dorchester, Mattapan, and Roxbury.





These histograms showed us that crime was being covered significantly more in black neighborhoods relative to other topics when compared to white neighborhoods. A next step would be to look into the actual crime rate in these neighborhoods, as there is actual data for that, relative to the frequency of articles and mentions of crime in these areas. This could be used to indicate if there is actually a bias present in WBUR's reporting.

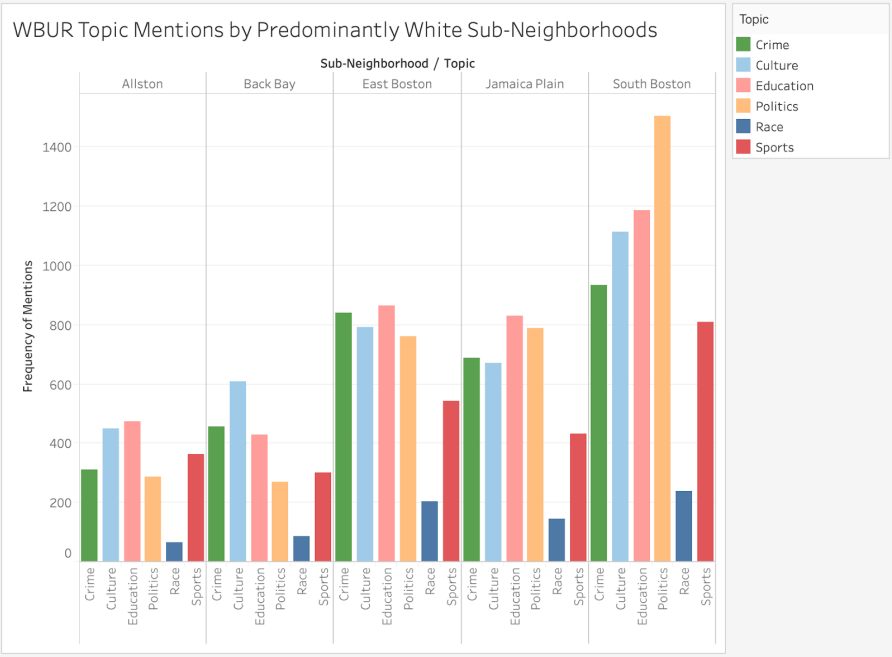
Given these topics and the racial distribution across Boston, we layered the frequency of mentions of these topics over a map of Boston showing the proportion of blacks living in each area. The following figure gives more insight into what topics are covered in each neighborhood of Boston for further analysis. The size of each point represents the frequency of mentions of that respective topic. Some of the major trends that this figure highlights is that sports are covered heavily in the Fenway area, politics are covered heavily in the financial district, and crime is covered heavily in the Dorchester/Roxbury areas.



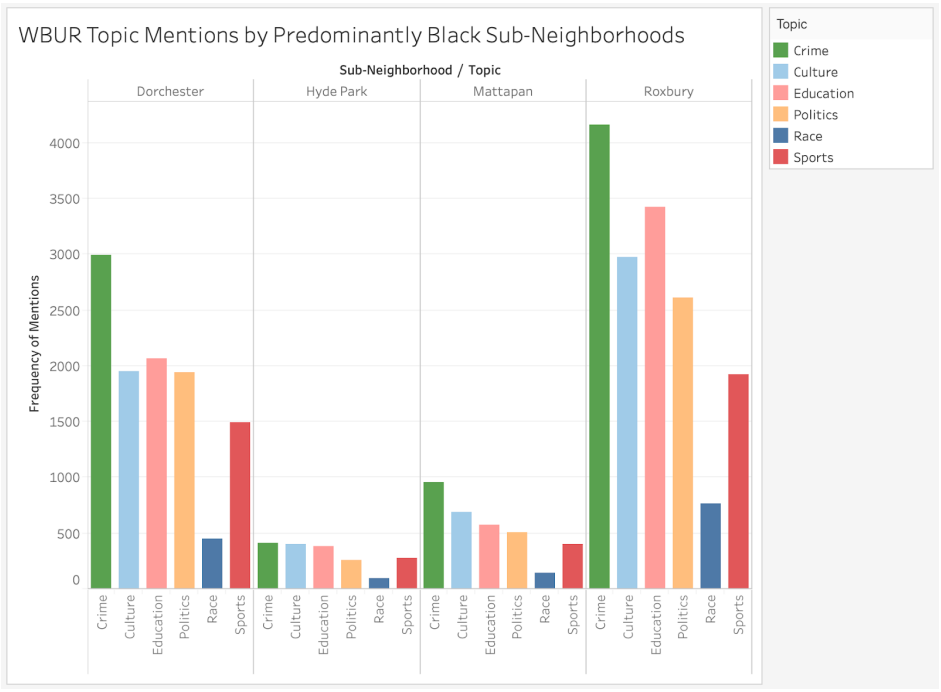
Topic Modeling by Sub-Neighborhoods

We performed the same analysis on the sub-neighborhoods of Boston to see with more specificity the topics that were being covered throughout the city of Boston by these sources. Using similar criteria as we had with neighborhoods to determine which predominantly white and black neighborhoods to analyze, we picked the top few based on articles mentioning these neighborhoods, as well as by size and general proportion of black or white residents.

We analyzed Allston, Back Bay, East Boston, Jamaica Plain, and South Boston for the top white sub-neighborhoods and found a general consistency of topics being mentioned, with politics being particularly high in South Boston. The following figure depicts the frequency of topics mentioned by sub-neighborhood.

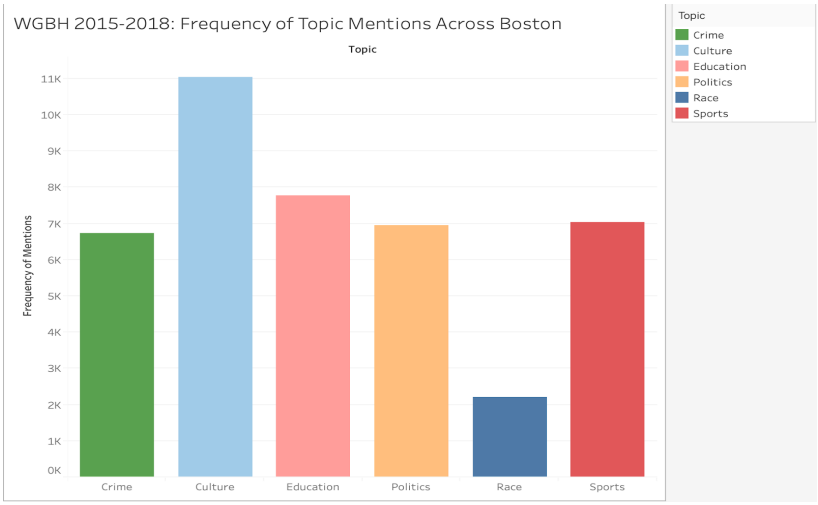


The following figure shows the same analysis conducted on the predominantly black sub-neighborhoods of Boston with a similar result to what we saw for the predominantly black neighborhoods themselves. There is a heavier emphasis on crime-related articles in these sub-neighborhoods, followed by an even spread of culture, education, and politics. Again, further analysis could be conducted in these areas to see if the crime rates match the high focus on crime in these areas.

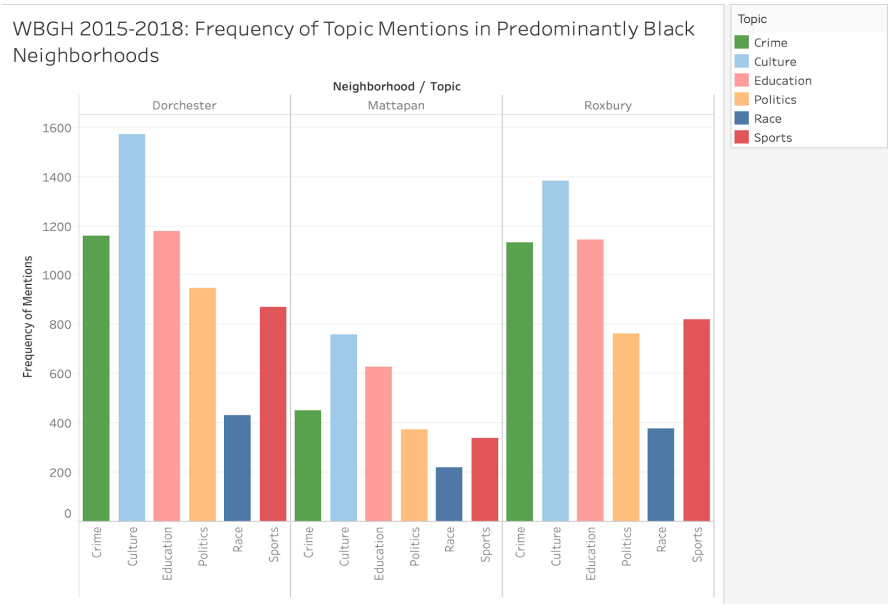


B- WGBH: Topics Covered

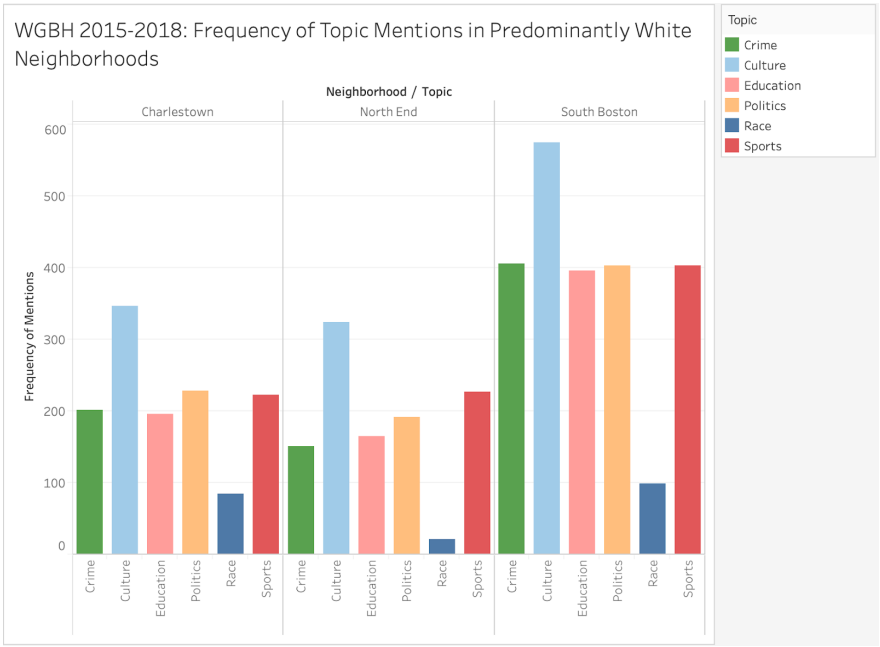
Using the same methodologies on the WGBH data set as were used on the WBUR dataset, we were able to analyze the topics that were covered by this media source. We used the same topics: race, culture, education, politics, sports, and crime, and this is the resulting histogram depicting the distribution of topics covered by WGBH. Compared to the WBUR dataset, WGBH mentions culture related topics most frequently, followed by an even spread of crime, education, politics, and sports related articles. Again, note that some articles can have multiple topics attributed to them.



Using the same population breakdown as above for the WBUR dataset, we looked at the top 3 predominantly black and white neighborhoods to analyze the distribution of topic mentions closer, to see if there is a racial bias associated with reporting in these areas. The top 3 areas from above: Dorchester, Mattapan, and Roxbury, had the following distribution of topic mentions.

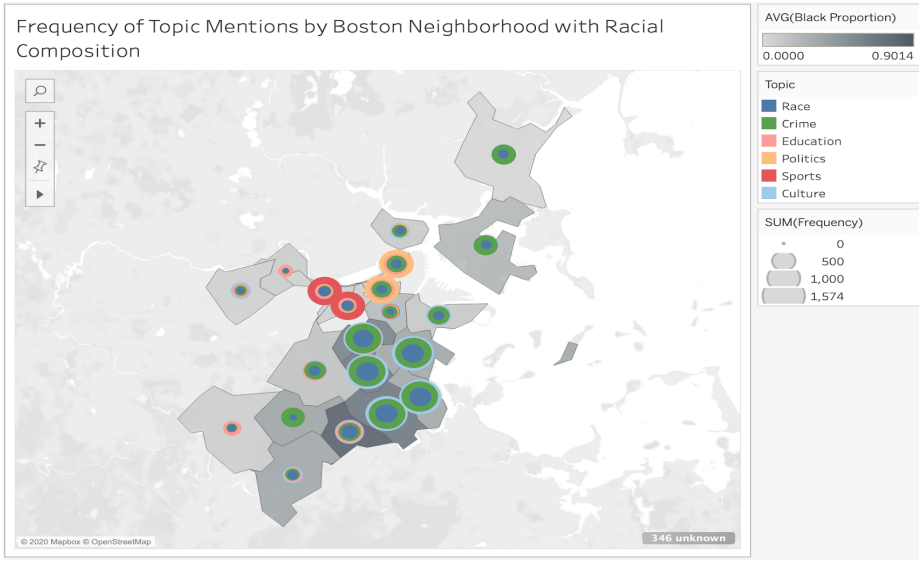


The top 3 predominantly white areas from above; Charleston, the North End, and South Boston, had the following distribution of topic mentions.



Though the predominantly black neighborhoods appeared to be covered more in general than the predominantly white neighborhoods based on this sampling approach, it again appears that crime is covered more heavily in these neighborhoods, along with politics, than the predominantly white neighborhoods. As mentioned before, comparing this information to the actual crime rates in these areas could be a point for further analysis and research to determine if there is actually a racial bias in reporting crime more heavily in these areas.

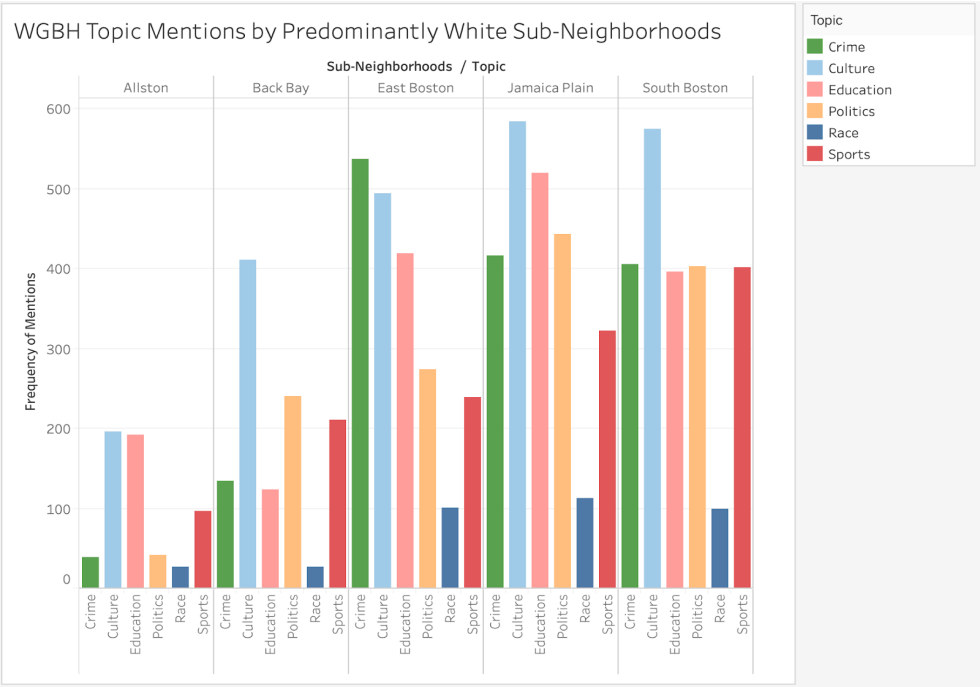
Again, we layered data points depicting the frequency of topic mentions over a map of Boston that shows the racial composition by zip code. The colors of the points represent the topics, while the size corresponds with the frequency of the mentions. Some of the same trends that we



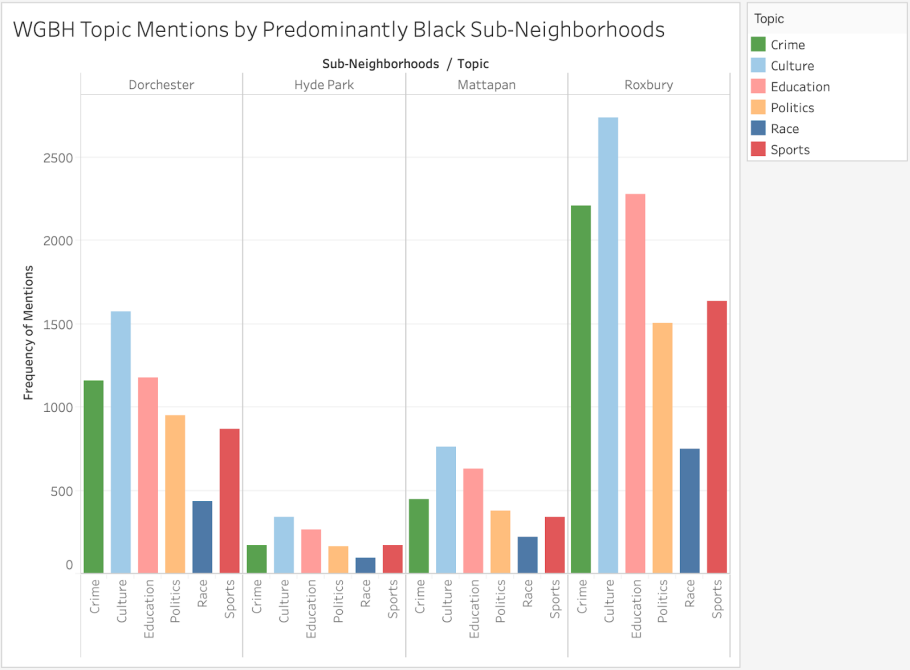
found in the WBUR dataset were evident here, where sports were covered most heavily in the Fenway area, while politics were covered heavier in the Financial District area.

Topic Modeling by Sub-Neighborhoods

Using the same neighborhoods as above with the WBUR dataset, and the same method of analyzing them, we took a look at the topics that were being mentioned by WGBH in these areas. The first figure below shows the same, predominantly white, or non-black, sub-neighborhoods: Allston, Back Bay, East Boston, Jamaica Plain, and South Boston. It appears that culture is covered very heavily in all of these sub-neighborhoods, which makes sense due to the heavy emphasis on culture in the neighborhoods as well, and for the entire news source in general. Interestingly, crime is covered quite heavily in East Boston as well.



The next figure shows the same analysis on the predominantly black sub-neighborhoods of Boston : Dorchester, Hyde Park, Mattapan, and Roxbury. Here, culture again is the most covered topic. Since some of the same neighborhoods are also considered as sub-neighborhoods, this analysis looks almost identical to that of the predominantly black neighborhoods figure for this news source.



C - Code Explanation

Using the pandas package in Python, we were able to import a merged dataset from both WBUR and WGBH, as well as the neighborhood and sub-neighborhood information from census data. Next, we isolated a list of either neighborhoods or sub-neighborhoods depending on the analysis being conducted. We created a list of 5 hypothetical topics as well based on both our intuition and the previous analysis about some of the frequently mentioned phrases and topics in relation to race. We created 30 unique keywords to attribute for each topic and imported this data file into our code.

Next, we iterated through each article, neighborhood, and topic keyword and kept a tally of the number of times a specific topic was mentioned (based on keywords from our topic set, including multiple mentions of each word), in each neighborhood from that dataset. The results were appended to a table in this format, for example: [‘fenway’, ‘sports’, 3].

This was then exported into a CSV file for further processing in Tableau to visualize and compile our results in order to draw conclusions. The format of the data from this CSV file made it easy to sum the mentions in tableau for each respective topic in each neighborhood. This process was repeated for both WGBH and WBUR separately to create two different output CSV files. We also used this process to analyze gun-violence mentions by creating a separate topic file as will be discussed later in this report.

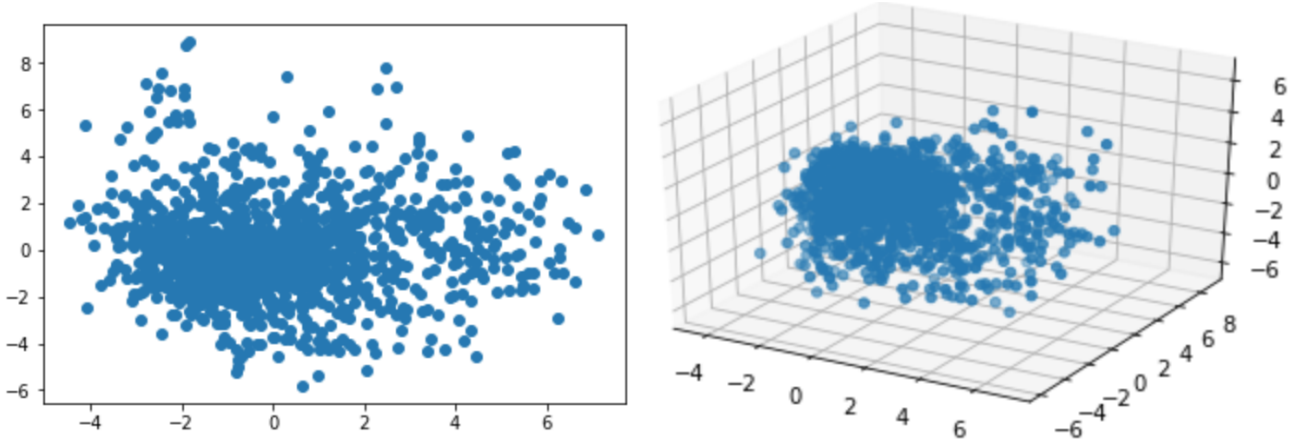
3- Exploring Different Modeling Techniques

WGBH Web-Scraped Data: Is there a "better" way to represent the data?

At the beginning of the semester our PM's showed us what they had previously done in order to capture clusters of news articles for downstream processing; they used LDA (Latent Dirichlet Allocation) clustering to capture the topics of news articles. The purpose of clustering these articles was to study what topics were being associated with whichever race was being mentioned. The problem here is that the LDA model uses the BOW (bag-of-words) approach to cluster text data which neglects word ordering; this can result in mediocre results at best.

One of the things we sought to improve was the feature representation of the textual data (i.e. news articles) and capture more meaning behind them in order to unlock more meaningful results. We focused on representing the data using document level embeddings via an algorithm called Doc2Vec. This algorithm is an extension of the famous Word2Voc model but instead of learning word embeddings it takes in entire documents and learns document embeddings that represent the underlying meaning of each document.

However, it turned out that these document embeddings were not as helpful in categorizing the articles as we hoped. Below is a 2D and 3D visualization of the document embeddings obtained from one of the web-scraped data sets we collected (WGBH 2017):



It is evident that the data is forming a cluster around the bottom left of the 2D graph even though there is a large number of points scattered away. This is most probably due to the underlying nature of news articles these days and how they can be convoluted with many topics (e.g. an article about how NBA players kneel during the national anthem can be about both sports and politics). This proved that there is more to be done in regards to better topic classification of news articles.

The aforementioned results can be viewed and reproduced from an ipython notebook located under the following directory
`naacp-spark-fall2020/notebooks/doc2vec_similarity_query.ipynb`.
What follows is a high-level overview of the gist of the code within that notebook along with documentation and explanations of code blocks/sections.

After cleaning the data from special characters such as # and **`\xa0`** that show up as a result from web-scraping we go on to split the words in each document within the corpus into individual words and drop so-called stop-words such as 'and' or 'the' that appear frequently within text phrases yet add no special meaning:

```
import gensim

def tokenize(text, stopwords, max_len = 20):
    return [token for token in gensim.utils.simple_preprocess(text, max_len=max_len) if token not in
stopwords]
```

```
list_of_articles = df.values.tolist()

flat_list_of_articles = [item for sublist in list_of_articles for item in sublist]

tagged_docs = [gensim.models.doc2vec.TaggedDocument(tokenize(text, [], max_len=200), [i]) for i, text
in enumerate(flat_list_of_articles)]
```

Towards the end of the you can notice that we perform document tagging for each processed document in our corpus using the `TaggedDocument` class from the gensim library. This helps later on in order to keep track of which document belongs to the output embedding.

Towards the end of the notebook you will see the performance of Principal Component Analysis in order to project our high-dimensional document embeddings into 2D and 3D spaces for graphing purposes:

```
from sklearn.decomposition import PCA

pca = PCA(n_components=3)
principalComponents = pca.fit_transform(X)

# Create a new dataset from principal components
df = pd.DataFrame(data = principalComponents,
                  columns = ['PC1', 'PC2', 'PC3'])
```

From then on it is just a matter of passing the principal components into graphing tools (e.g. `matplotlib` for 2D and `mpl_toolkits` for 3D).

Again, all of this can be reproduced by simply running the notebook from start to finish in any IDE that supports ipython notebooks such as Jupyter or Google Colab.

WGBH Web-Scraped Data: Can we at least capture sentiment?

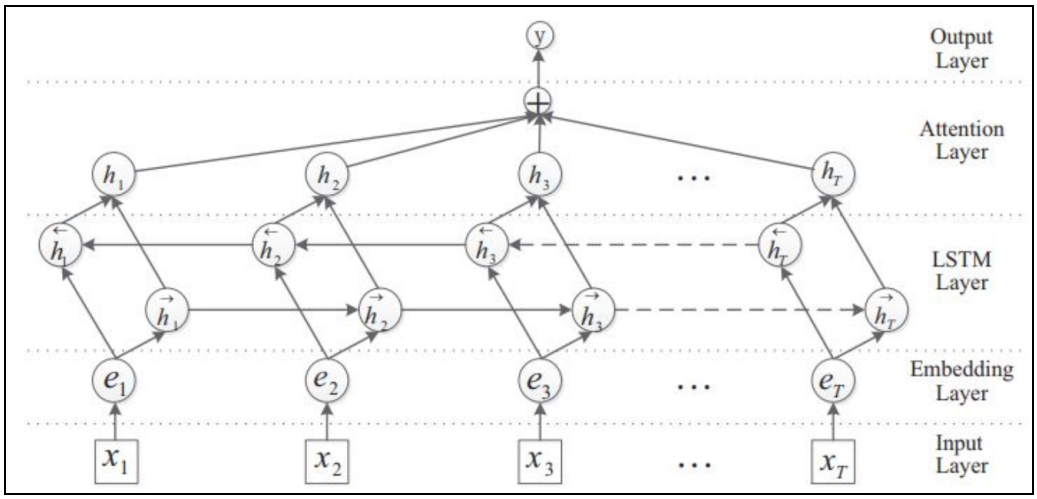
We then focused back on the traditional Word2Vec embeddings to represent the news articles in order to perform sentiment analysis. By transforming word documents into embeddings and then feeding them into a recurrent neural network we were able to successfully perform sentiment analysis on the data with an average accuracy score of around 57% for unseen data:

```
The following is the classification score report on novel data:
              precision    recall  f1-score   support

           0       0.57      0.61      0.59     12500
           1       0.58      0.54      0.56     12500

    accuracy                           0.57     25000
   macro avg       0.57      0.57      0.57     25000
weighted avg       0.57      0.57      0.57     25000
```

The specific architecture we used was a bidirectional LSTM (Long Short Term Memory) neural network with the option of specifying an attention layer; the attention layer is nothing more than one-layer neural network appended to the LSTM's outputs to help the neural network better focus on the right words when performing sentiment analysis. The following is a high-level depiction of the neural network architecture:



To build the neural network you would have to call the helper function `build_bidirectional_lstm` located in the `naacp-spark-fall2020/modules/utils.py` file. Two types of neural networks can be returned from the specified function depending on whether or not the user would want to have an attention layer on top of the bidirectional LSTM or not. In

the case that the user does which to utilize an attention layer then the following code blocks executes:

```
sequence_input = Input(shape=(N,))
embedded_sequences = Embedding(max_features, embed_size)(sequence_input)

(bilstm, forward_h, forward_c, backward_h, backward_c) = Bidirectional(
   LSTM(196,
        dropout=0.2,
        return_state=True,
       return_sequences=True)
)(embedded_sequences)

state_h = Concatenate()([forward_h, backward_h])
state_c = Concatenate()([forward_c, backward_c])

context_vector, attention_weights = Attention(10)(bilstm, state_h)

dense = Dense(20, activation="relu")(context_vector)

dropout = Dropout(0.05)(dense1)
output = Dense(1, activation="sigmoid")(dropout)

model = keras.Model(inputs=sequence_input, outputs=output)
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=METRICS)
```

The general workflow behind the above code block is to start off by creating two LSTMs and tying them together in both the forward and backward direction; this is taken care of by passing a single LSTM to the Bidirectional class constructor at the very beginning. Then the outputs from both directions (i.e. both LSTMs) need to be concatenated together and fed into the Attention layer seeing that its sole purpose is to tell our network what words of the input document to focus on when coming up with a prediction. This can be seen in the back-to-back calls of the Concatenate class for variables `state_h` and `state_c`.

Finally, seeing that this is a classification task with only two target sentiments (positive and negative) our output layer is made up of a single node with a sigmoid activation function and the model is compiled with the binary cross entropy loss.

In the case that the user does not wish to augment an attention layer on top of the bidirectional LSTM then the following code block will execute:

```
model = Sequential()

model.add(Embedding(max_features, embed_size, input_length=N))
model.add(Bidirectional(LSTM(196, dropout=0.2, return_state=True)))
model.add(Dense(1, activation="sigmoid"))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=METRICS)
```

This is very similar to the previous code block with the exception of no attention layer and hence no need for performing concatenation to the bidirectional outputs. How a user can specify the need for an attention layer is very simple; passing either True or False to the `build_bidirectional_lstm` function's parameter argument `is_attention` which has a default value set to False.

4 - Sentiment Analysis

A - WBUR:

Is topic modeling better than explicit race mention for sentiment analysis?

Previously, explicit race mention was used to create two datasets: one where the articles are about black people and another where the articles are about white people. Then sentiment analysis can be used on each dataset to determine whether or not the media is negatively reporting on black people. However, we decided that instead of using explicit race mention to construct these two datasets, we would use topic modeling. We developed a Doc2Vec that takes as input an array of words and returns a similarity score along with the indices of articles that embody the input array of words. We used words like “black”, “african american”, and “haitian” to create a topic that encompasses articles discussing black people. We did the same for white people as well. The similarity score is a number between 0 and 1 so the closer the score is to 1, the better the article embodies the topics. We decided on a threshold by manually experimenting with values in the range of 0.1 to 0.9 and then randomly sampling ten articles to check to see if they were about the given topic. We found that 250 was the optimal number of articles with a minimum threshold of 0.23. We had five WBUR datasets, so we ended up with 1250 articles about black people and 1250 articles about white people.

Since the WBUR data was not labeled, we tried unsupervised methods for sentiment analysis. Specifically, we used VADER and TextBlob polarity. VADER uses a list of lexical features, like words, that are already labeled as positive or negative in order to determine the sentiment of the entire text. TextBlob uses polarity scores, which take into account the semantic relations and frequency of each word in a sentence. VADER returns three scores, the positive, negative, and neutral score. Each range from 0 to 1 where values close to 1 mean the articles exhibit a particular sentiment. So a negative score of 0.87 would be highly negative while a positive score of 0.87 would be highly positive. For TextBlob, articles with scores greater than 0 are considered positive while scores less than 0 are negative. The results can be seen in the tables below.

VADER

Score Type	Black Articles	White Articles
Positive	0.0567	0.0591
Negative	0.0835	0.0822
Neutral	0.8598	0.8586

TextBlob

Score Type	Black Articles	White Articles
Positive	0.1168	0.1164
Negative	-0.0663	-0.0622

The results from VADER indicate that the sentiment for white and black articles is roughly the same. Also, the coverage for both black and white articles is very neutral. The results from TextBlob also show that the sentiment for black and white articles is very similar. However, the TextBlob results show a slightly positive sentiment for both black and white articles. The negative score for both black and white articles is very close to zero so they are not highly negative. These results match the results that were produced last year when running these algorithms on articles with explicit race mention.

B- WGBH:

This process was repeated for the WGBH data. The results are shown in the tables below:

VADER

Score Type	Black Articles	White Articles
Positive	0.0536	0.0538
Negative	0.0734	0.0712
Neutral	0.8727	0.8640

TextBlob

Score Type	Black Articles	White Articles
Positive	0.134	0.096
Negative	-0.084	-0.098

The results are very similar to those of WBUR. We still see that in general, the media sentiment towards both groups is leaning towards neutral. For the TextBlob method, we can see that WGBH articles for both black and white mentions are more negative than in WBUR. However, the change is not by a large amount. Similarly, the positive sentiments for black and white mentions for the WGBH data is also less positive than in WBUR. For the VADER method, there is not any difference between the sentiment of black and white mentions for WBUR and WGBH.

C - Code Explanation:

The Doc2Vec model is used to obtain articles of a particular topic. Specifically, we use the `model.infer` method to pass a list of topic keywords to the Doc2Vec model. To get articles discussing black people, we used the following words: “black”, “african-american”, “african american”, “haitian”, “jamaican”, “west indian”, and “dominican”. To get articles discussing white people, we used the words “white”. “caucasian”, “italian”, and “irish”. After getting the articles, we save them to two separate CSV files. We performed the unsupervised sentiment analysis in the “sentiment_analysis” notebook. We first read in the two CSV files that we obtained from the Doc2Vec model. Then we iterate through all the black articles, calculate the sentiment scores using VADER, store them in a list, and then calculate the mean. We repeat the same procedure for the white articles. This entire process is then repeated again using TextBlob to calculate the sentiment.

5 - Crime Analysis

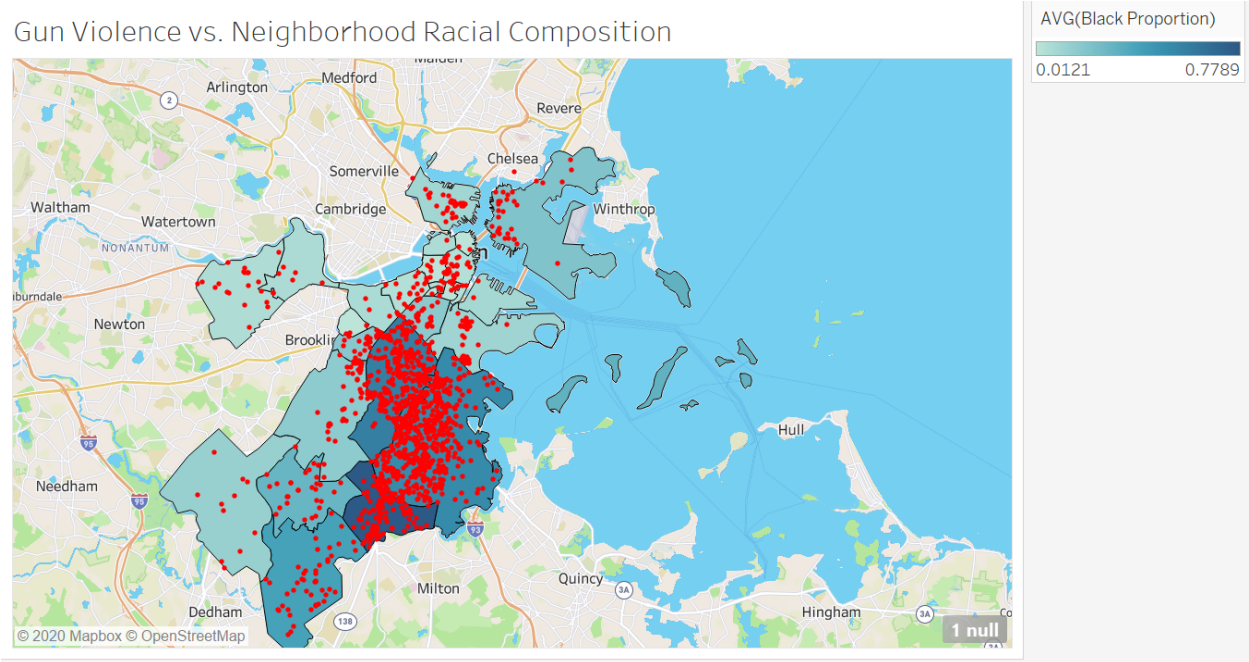
Is coverage of gun violence (including homicides) consistent with higher incidence or are there variations across different neighborhoods?

Originally, we were given a list of homicide victims from Boston, with each individual's name, age, race, etc. However, this information on its own lacked an element to connect it to news coverage by neighborhood. Thus, using the open database known as the *Gun Violence Archive* (https://www.gunviolencearchive.org/), we were able to create a query to extract a CSV file containing all incidents of gun violence in Boston from 2014 to 2018. This is easily reproducible by accessing the online database and extracting the incidents for the desired time frame and location. Notably, each incident also had an address included (amongst other new information) so now we had a geographical component to work with. Yet this address on its own would still not be able to yield a specific point on a data visualization, so we created an instance of a geocoder in python using the geopy package. This would give us the ability to go through each of the roughly 2000 incidents and pinpoint it to an exact latitude and longitude (and altitude, although that is certainly not relevant for our purposes), which could then be put onto Tableau. One thing to note is that the geocoder requires us to timeout between each entry (that is, a RateLimiter built into geopy, where we must specify how long we delay between each iteration) so that we do not get locked out of access to the server, so going through every single entry requires quite a bit of time.

With the average proportion of the Black residents in each of Boston's neighborhoods previously found by the aforementioned Census data, we would be able to see how incidents of gun violence varied across the neighborhoods. Furthermore, and more importantly, we could then combine this with the output from code that goes through each article, and look for mentions of specific neighborhoods and a new set of key words relating to gun violence (similar to the method used in the previous section). The significance here is that we could now see if greater incidence of gun violence in a particular neighborhood is correlated to greater coverage of it in news articles or if there is some discrepancy, and if this discrepancy can be attributed to the racial compositions of the neighborhoods. However, moving forward, applying this same logic and visualization model to future gun violence incidents (and hopefully, other news outlets) can provide a basis for evaluating bias in reporting gun violence in areas depending on their racial composition. Moreover, if we wanted to focus solely on homicides, the Gun Violence Archive CSV provides a column that represents the number of people killed in each incident, so in the future, it would be possible to use the same database and simply filter out the entries for which this column has a value of 0.

In more technical terms, this visualization (shown below) begins with the map of Boston first layered with the proportion of Black residents in each neighborhood, then with red colored dots to indicate each of the approximately 2000 incidents of gun violence and where they occurred.

The darkness of the blue hue of each neighborhood block represents the proportion of Black residents there. Evidently, the neighborhoods of Roxbury, Dorchester, and Matapan have the most incidents of gun violence within them and/or the densest clusters of them. The task now was to take such information and compare it to coverage by the WBUR and WGBH to see if a proportional amount of articles were written about these neighborhoods regarding gun violence.

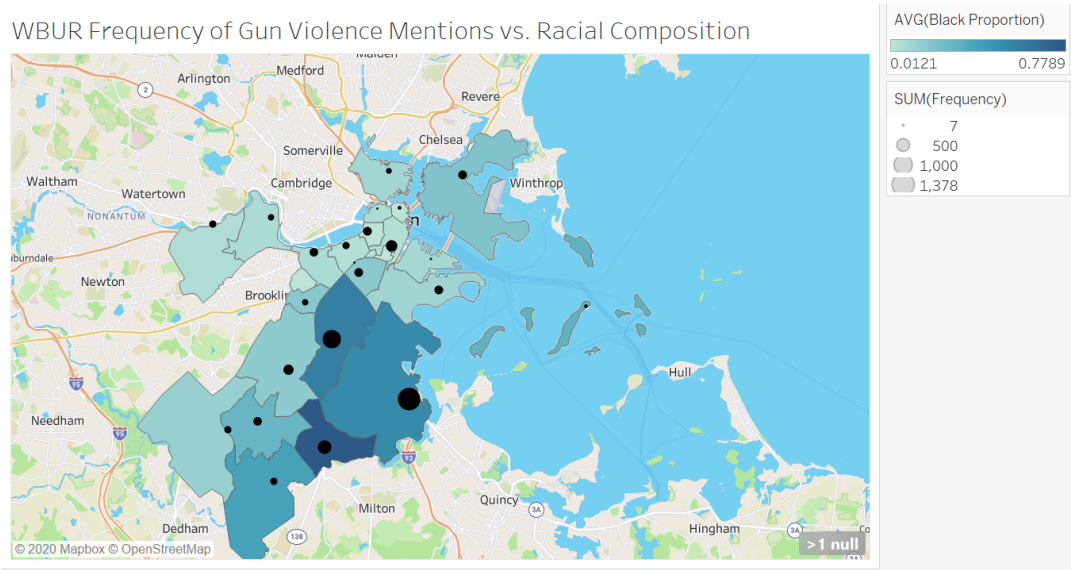


Incidents of Gun Violence (2014-2018) Compared to Racial Composition of Boston Neighborhoods

A - WBUR: Crime Coverage:

Using the web-scraped data from the WBUR between 2014 and 2018, we were able to generate how many articles for each neighborhood had mentions of gun violence. In conjunction with the figure shown above, we could look for a discrepancy between incidence of gun violence and coverage of it in different neighborhoods. The presence of such a discrepancy could be indicative of a bias present (or lack thereof) in the WBUR's reporting.

As such, the visualization for this aspect begins with the same first layer of the map of Boston with the proportion of Black residents in each neighborhood, and then contains a larger black mark above or right near each neighborhood to specify how frequently gun violence was mentioned in the WBUR articles.



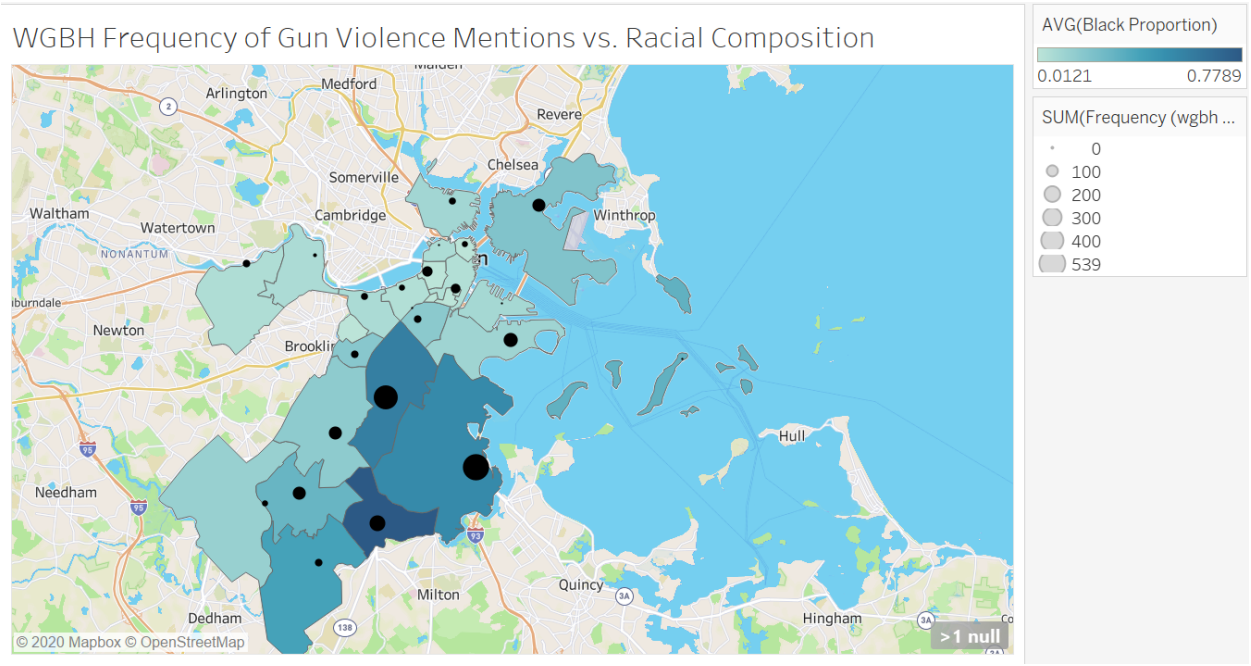
Frequency of Gun Violence Mentions in WBUR (2014-2018) Articles

To reiterate, in the above image, each black dot indicates the frequency of gun violence (through a set of keywords related to the topic) being mentioned for articles in the WBUR (2014 - 2018) for each neighborhood. The larger the dot, the more frequently gun violence was mentioned in articles about neighborhoods at that point. To the naked eye, there seems to be no discrepancy between the density of these incidents and frequency of being talked about by the WBUR. The only slight mismatch that appears is that despite having a comparable density of gun violence incidents as Dorchester and Roxbury, the neighborhood of Matapan (the darkest blue block towards the bottom of the map) seems to get significantly less gun violence coverage by the WBUR. Although this could be attributed to Matapan having a far smaller size than the other two amongst other factors, it is interesting to note this because Matapan is the neighborhood with the highest proportion of Black residents on the map.

B - WGBH: Crime Coverage

Using the web-scraped data from the WGBH between 2014 and 2018, we were able to generate how many articles for each neighborhood had mentions of gun violence in the same way as previously mentioned. In conjunction with the first figure shown above, we could look for a discrepancy between incidence of gun violence and coverage of it in different neighborhoods. The presence of such a discrepancy now could be indicative of a bias present (or lack thereof) in the WGBH's reporting.

The visualization for this aspect begins with the same first layer of the map of Boston with the proportion of Black residents in each neighborhood, and the same larger black mark above or right near each neighborhood to specify how frequently gun violence was mentioned in the WGBH articles.



Frequency of Gun Violence Mentions in WGBH (2014-2018) Articles

To reiterate, in the above image, each black dot indicates the frequency of gun violence (through a set of keywords related to the topic) being mentioned for articles in the WGBH (2014 - 2018) for each neighborhood. The larger the dot, the more frequently gun violence was mentioned in articles about neighborhoods at that point. Again, interestingly, there seems to be no straightforward discrepancy between the density of these incidents and frequency of being talked about by the WGBH nor does the proportion of gun violence mentions for each neighborhood differ drastically between the WBUR and WGBH. However, as mentioned for the WBUR, we see the same mismatch regarding the neighborhood of Matapan and it having significantly less gun violence coverage by the WGBH than Dorchester and Roxbury. While there is no way to immediately jump to conclusions about the cause of this disparity, it is still intriguing to observe and possibly try to further investigate later through other means.

C - Code Explanation:

The preliminary code necessary for this was the geocoder necessary to take a CSV of gun violence incidents with their addresses and find out each one's latitude and longitude. The code is located in the file with the path `data visualization/gun_violence_geocoder.py`. Using the geopy package in Python, we were able to create a geocoder from the Nominatim API and as previously mentioned, had to define the RateLimiter to delay each geocoding request by 1 second to prevent being access restricted. Since the CSV provided the address components in separate columns, we needed to concatenate them into a single *Full Address* column, to which we could apply the geocoder and retrieve a tuple in the form of (*Latitude, Longitude, Altitude*). One drawback of the Nominatim API is that its geocoder is not good at handling addresses that

indicate street intersections (e.g. “Parkman St and Dorchester Ave”) so the output for many of those came out to be *None.* Thus, it was necessary to filter those out of the DataFrame so we could split the tuples into respective columns without running into any errors. With an updated dataset containing the exact location of each incident, we could turn that back into its own CSV, which Tableau could then utilize as desired.

The other code necessary for this analysis was that as previously mentioned which went through the dataset of articles for each news source and output the frequency of gun violence being mentioned (using a set of keywords related to the topic) for articles about each neighborhood. In other words, it would go through the articles, figure out which neighborhood(s) the article relates to, and see how many mentions of gun violence (or its keywords) are present. Then, we can create a CSV that contains which Neighborhood(s) each article is about and a value for the topic frequency. Tableau is able to sum up the frequencies for each neighborhood and represent it as a dot, where the size of the dot varies based on the summation of the frequencies for that neighborhood.

6 - Entity Recognition

A - WBUR/WGBH

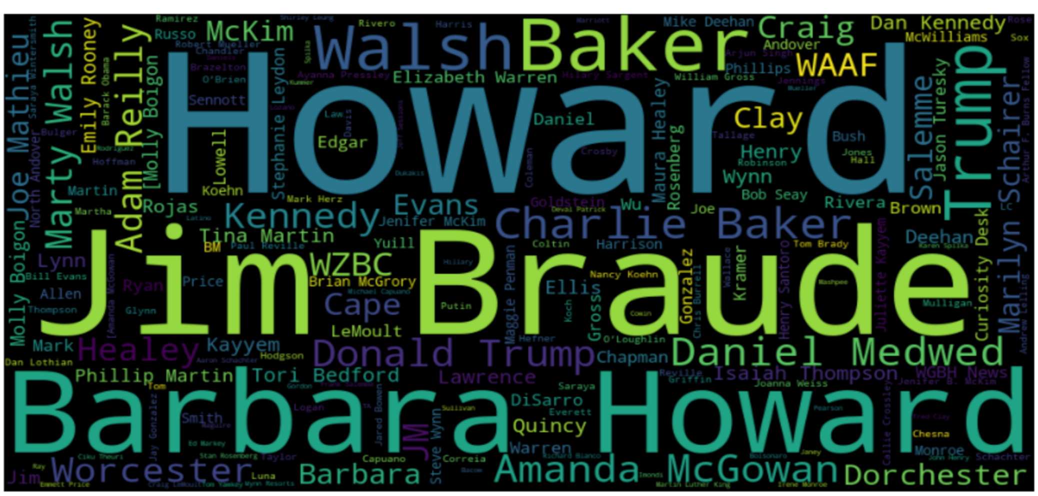
We implemented an entity recognition method as the client asked. Entity recognition is a subtask of information extraction that seeks to locate and classify named entities mentioned in unstructured text into predefined categories such as person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc. Our clients want us to find all entities that in the given Boston dataset articles with the label person's name and company. We used NLP and python to achieve the goal. By using python package SpaCy, we are able to identify the entities in the given dataset, showing as the figure below.

This algorithm will successfully identify entities that exist in a given dataset with an accuracy almost nearly 100%. But sometimes it will give an entity not fully correct like include some . or [ ] in the result it returns.

B - Code Explanation

The first step is to take the CSV file of every article in WBUR and WGBH as the input source file. By using the SpaCy package in Python, we were able to identify nouns, verbs, and adjectives. From all the words we only want to keep the nouns containing important information. We predetermine the entity such as Organization, Street name, and Person name and classify every noun into these entities. Besides that, we also implemented a static function and a word cloud function to make the data more informative and clear to people with little computer science background. The static function returns a result that tells people how many times an entity appears in the dataset. Also have the ability to rank.

Every entity is independent from each other right now. For a given entity it will output a word cloud which the more a word appears in the database, the bigger the size is shown in word cloud.



7 - Limitations and Future Work

A - Limitations:

For the gun-violence incidents data, there were many incidents that did not include an exact address. Intersections or just street names were provided so the data could not be geocoded with the correct location with the current implementation. This could lead to some inaccuracy in the visualizations as some of the incidents are not marked in the exact area that they occured. There were also some data quality issues that were hard to control. For example, the WBUR included articles that discussed national news. Since the goal was to analyze media bias on a local level, removing these articles is important but it's difficult to differentiate between local and national news. In addition, since the data that we web-scraped was not labeled, we couldn't train our LSTM for sentiment analysis on the actual articles. We used the IMDB dataset to train the model so we cannot be certain on how it will perform on the actual data. Also, the inherent nature of news articles leads to subject embedding overlap, which could present difficulties when performing topic-modeling. When dealing with sensitive topics like racism, there could possibly be existing confirmation bias as well when implementing ideas and interpreting results.

B - Future Work:

We believe that future groups can continue to make improvements on how to better visualize the topic modeling results. Perhaps one way could be to use interactive plots, so the user can add and remove topics from the map. In addition to this, we think that creating a simple user interface or web app for the visualizations could be a great way for non-technical users to interact with the visualizations. Libraries like Plotly and Dash can be used to do this quickly. We found a tutorial that outlines how to create a dashboard with Plotly and Dash. Streamlit can also be used to do this. For entity recognition, future work can be focused on using the Linkedin and Google Places API to link the recognized entities to people and businesses. Creating a simple web app with Streamlit could also be beneficial for entity detection as well. For the gun violence analysis, future groups can make use of a more robust geocoder or data that already has geographical coordinates for all incidents of gun violence/homicide to analyze their coverage. For sentiment analysis, groups can focus on supervised sentiment analysis methods. We created an LSTM model for supervised sentiment analysis so if any of three datasets can be labeled, future groups can train the model with the labeled data.