Peyton Skwarczynski

CSI 4180 – Natural Language Processing

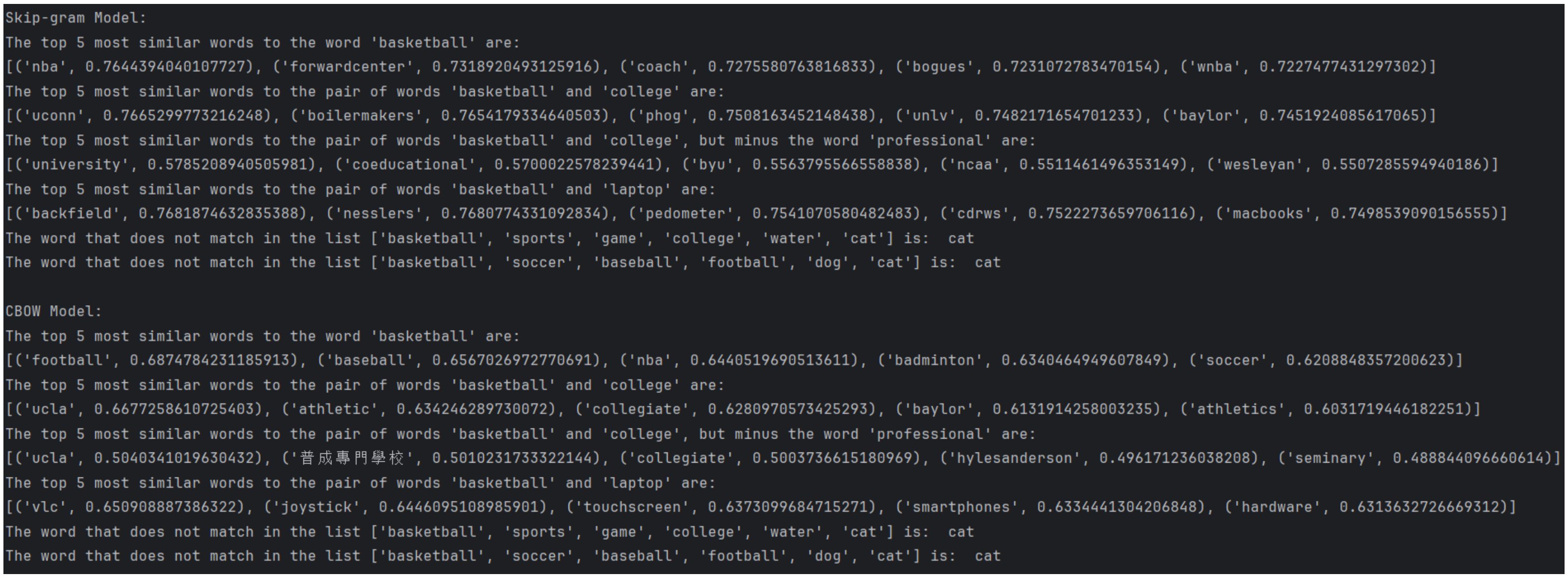
Dr. Steven Wilson

NLP Homework 2: Word Embeddings

1. Embeddings:

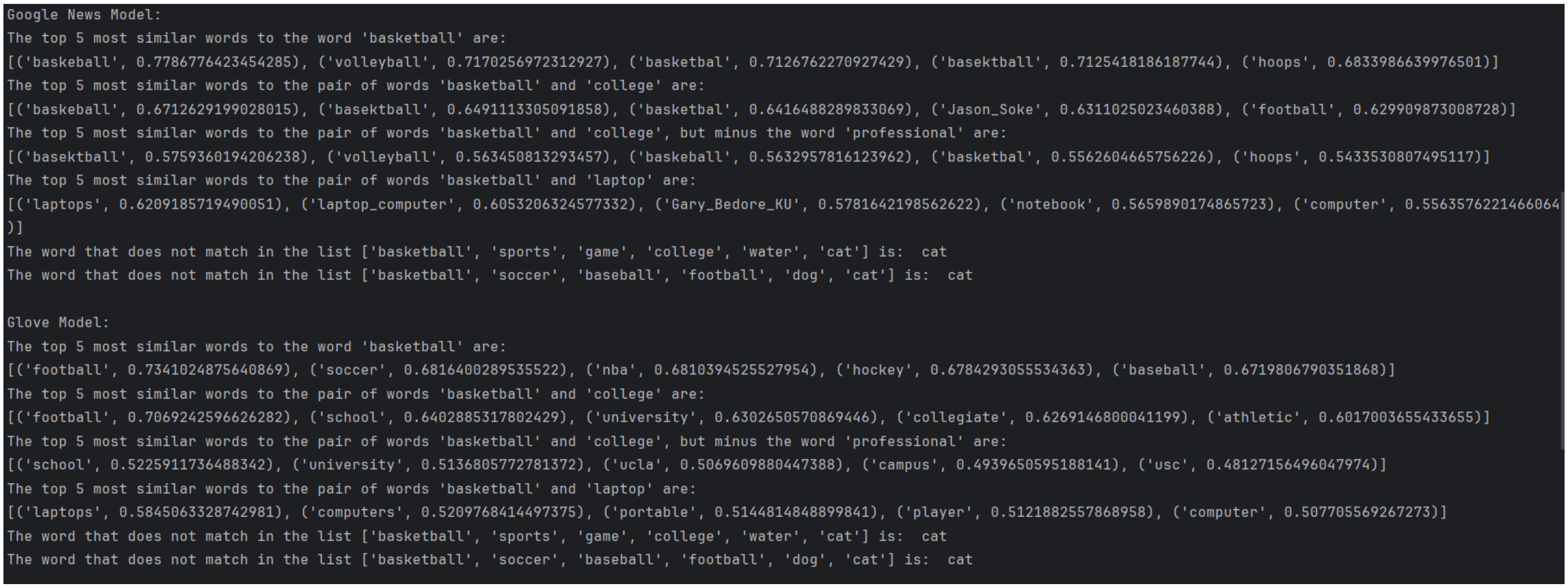
The two main variations of word embeddings I trained were the Skip-Gram and Continuous Bag of Words (CBOW) models. The package I used to train them is Word2Vec from the Gensim library. The only difference in how the models are trained is the training algorithm itself. In the Skip-Gram model , the algorithm tries to predict the many context words based off of the main input word. The CBOW model works in the complete opposite direction, where the input context words are used to predict the main word. The most important thing that I learned during the training process is that depending on the size of your original dataset, the training process can be extremely lengthy. From researching the Word2Vec function, I realized that you could increase the "workers" parameter so that multiple cores train the model. This simple, but important change can significantly reduce model training time.

Query Results:



```
Skip-gram Model:
The top 5 most similar words to the word 'basketball' are:
[('nba', 0.7644394040107727), ('forwardcenter', 0.7318920493125916), ('coach', 0.7275580763816833), ('bogues', 0.7231072783470154), ('wnba', 0.7227477431297302)]
The top 5 most similar words to the pair of words 'basketball' and 'college' are:
[('uconn', 0.7665299773216248), ('boilermakers', 0.7654179334640503), ('phog', 0.7508163452148438), ('unlv', 0.7482171654701233), ('baylor', 0.7451924085617065)]
The top 5 most similar words to the pair of words 'basketball' and 'college', but minus the word 'professional' are:
[('university', 0.5785208940505981), ('coeducational', 0.5700022578239441), ('byu', 0.5563795566558838), ('ncaa', 0.5511461496353149), ('wesleyan', 0.5507285594940186)]
The top 5 most similar words to the pair of words 'basketball' and 'laptop' are:
[('backfield', 0.7681874632835388), ('nesslers', 0.7680774331092834), ('pedometer', 0.7541070580482483), ('cdrws', 0.7522273659706116), ('macbooks', 0.7498539090156555)]
The word that does not match in the list ['basketball', 'sports', 'game', 'college', 'water', 'cat'] is:  cat
The word that does not match in the list ['basketball', 'soccer', 'baseball', 'football', 'dog', 'cat'] is:  cat

CBOW Model:
The top 5 most similar words to the word 'basketball' are:
[('football', 0.6874784231185913), ('baseball', 0.6567026972770691), ('nba', 0.6440519690513611), ('badminton', 0.6340464949607849), ('soccer', 0.6208848357200623)]
The top 5 most similar words to the pair of words 'basketball' and 'college' are:
[('ucla', 0.6677258610725403), ('athletic', 0.634246289730072), ('collegiate', 0.6280970573425293), ('baylor', 0.6131914258003235), ('athletics', 0.6031719446182251)]
The top 5 most similar words to the pair of words 'basketball' and 'college', but minus the word 'professional' are:
[('ucla', 0.5040341019630432), ('普成專門學校', 0.5010231733322144), ('collegiate', 0.5003736615180969), ('hylesanderson', 0.496171236038208), ('seminary', 0.488844096660614)]
The top 5 most similar words to the pair of words 'basketball' and 'laptop' are:
[('vlc', 0.650908887386322), ('joystick', 0.6446095108985901), ('touchscreen', 0.6373099684715271), ('smartphones', 0.6334441304206848), ('hardware', 0.6313632726669312)]
The word that does not match in the list ['basketball', 'sports', 'game', 'college', 'water', 'cat'] is:  cat
The word that does not match in the list ['basketball', 'soccer', 'baseball', 'football', 'dog', 'cat'] is:  cat
```



```
Google News Model:
The top 5 most similar words to the word 'basketball' are:
[('baskeball', 0.7786776423454285), ('volleyball', 0.7170256972312927), ('basketbal', 0.7126762270927429), ('basektball', 0.7125418186187744), ('hoops', 0.6833986639976501)]
The top 5 most similar words to the pair of words 'basketball' and 'college' are:
[('baskeball', 0.6712629199028015), ('basektball', 0.6491113305091858), ('basketbal', 0.6416488289833069), ('Jason_Soke', 0.6311025023460388), ('football', 0.629909873008728)]
The top 5 most similar words to the pair of words 'basketball' and 'college', but minus the word 'professional' are:
[('basektball', 0.5759360194206238), ('volleyball', 0.563450813293457), ('baskeball', 0.5632957816123962), ('basketbal', 0.5562604665756226), ('hoops', 0.543353080749511 7)]
The top 5 most similar words to the pair of words 'basketball' and 'laptop' are:
[('laptops', 0.6209185719490051), ('laptop_computer', 0.6053206324577332), ('Gary_Bedore_KU', 0.5781642198562622), ('notebook', 0.5659890174865723), ('computer', 0.5563576221466064
)]
The word that does not match in the list ['basketball', 'sports', 'game', 'college', 'water', 'cat'] is:  cat
The word that does not match in the list ['basketball', 'soccer', 'baseball', 'football', 'dog', 'cat'] is:  cat

Glove Model:
The top 5 most similar words to the word 'basketball' are:
[('football', 0.7341024875640869), ('soccer', 0.6816400289535522), ('nba', 0.6810394525527954), ('hockey', 0.6784293055534363), ('baseball', 0.6719806790351868)]
The top 5 most similar words to the pair of words 'basketball' and 'college' are:
[('football', 0.7069242596626282), ('school', 0.6402885317802429), ('university', 0.6302650570869446), ('collegiate', 0.6269146800041199), ('athletic', 0.6017003655433655)]
The top 5 most similar words to the pair of words 'basketball' and 'college', but minus the word 'professional' are:
[('school', 0.5225911736488342), ('university', 0.5136805772781372), ('ucla', 0.5069609880447388), ('campus', 0.4939650595188141), ('usc', 0.4812715649604797 4)]
The top 5 most similar words to the pair of words 'basketball' and 'laptop' are:
[('laptops', 0.5845063328742981), ('computers', 0.5209768414497375), ('portable', 0.5144814848899841), ('player', 0.5121882557868958), ('computer', 0.507705569267273)]
The word that does not match in the list ['basketball', 'sports', 'game', 'college', 'water', 'cat'] is:  cat
The word that does not match in the list ['basketball', 'soccer', 'baseball', 'football', 'dog', 'cat'] is:  cat
```

The two additional pre-trained word embedding models that I used were the Google News and GloVe model. I ended up running each model through each of the following six queries:

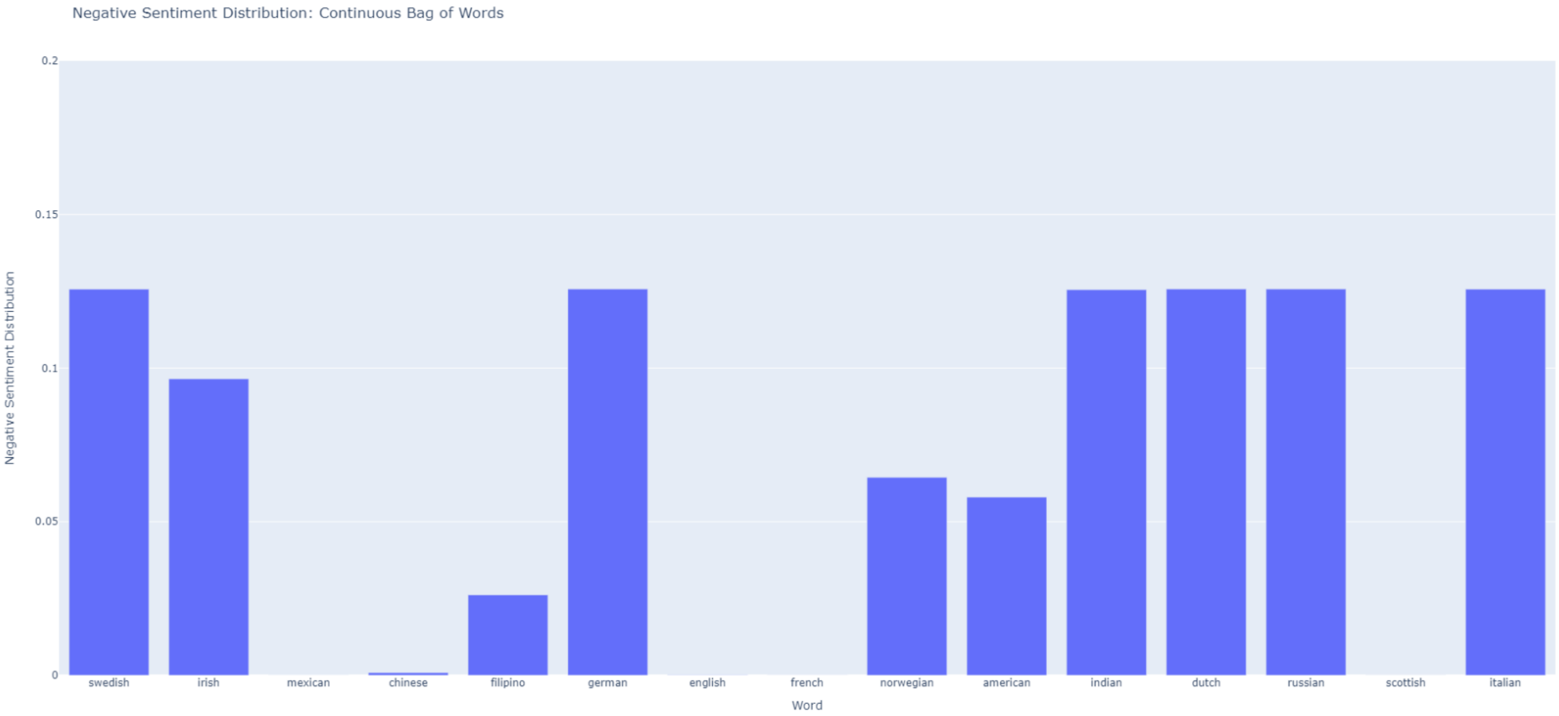
1. The top 5 words most similar to the word "basketball"
2. The top 5 words most similar to the words "basketball" and "college"
3. The top 5 words most similar to the words "basketball" and "college", but minus the word "professional"
4. The top 5 words most similar to the words "basketball" and "laptop". These two words are not that similar to each other.
5. Select the word in the list that does not match. Where there is one obvious answer.
6. Select the word in the list that does not match. Where there are two obvious answers.

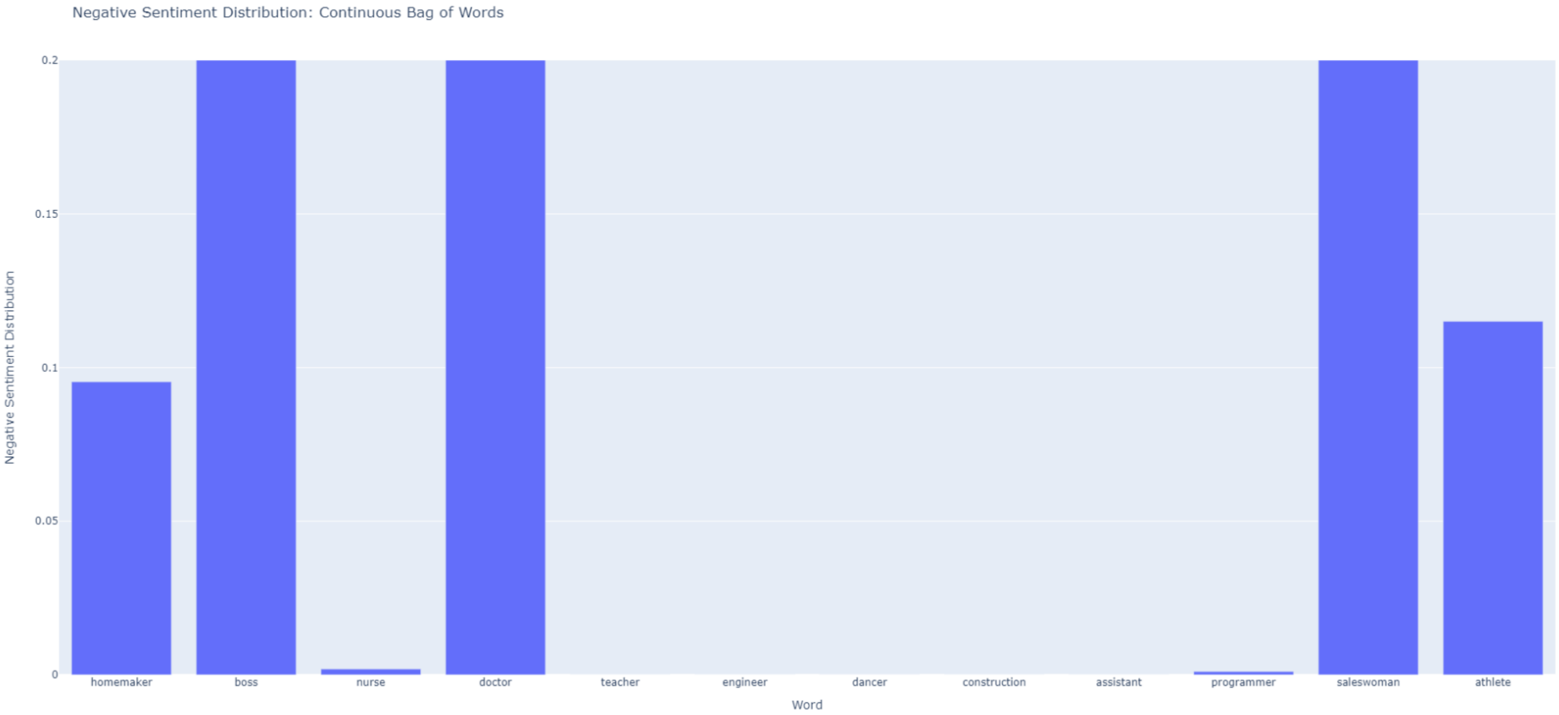
The results for the first query are fairly consistent. Each model printed words that are related to basketball or other sports. The only weird finding for this query is in the Google News model, where some of the top related words are misspellings of the word basketball. This misspelling is also found in the first three queries as well. I think it is purely due to the dataset of the Google News model, as everything found on Google does not have to be a professional paper that is thoroughly spell-checked. The results for the second query are also very consistent and expected. The results for all of the models are related to collegiate basketball, either the sport in general or college team names. Even though the third query should contain results fairly similar to the second query, this proved to not be the case. By removing the word professional from this test, the most similar words for all models were more closely related to the word college than basketball. This is seen through the words such as "coeducational", "seminary", "campus", and "school". For my generated CBOW model, the third query also somehow managed to have a Chinese word in it, "普成專門學校". I am not sure how this was one of the most related words, but when putting this word into Google Translate, you get "Pucheng Vocational School". My best guess is that it could be a popular basketball school. As for the fourth query, I did not know what results to expect. After testing, the outcome from this query showed words that are more closely related to the word "laptop" than "basketball". In fact, very few words had any visible correlation to "basketball" at all, but rather contained words that describe computer hardware. The results of the fifth and six queries were consistent with all models. Each model identified the least similar word to be “cat”. For the fifth query, this makes obvious sense, but for the sixth query, I would have thought that “dog” was going to be selected, as there are less sports teams that are named after dogs than cats. If you look at any professional or collegiate sports league, there are tons of teams called the Bengals, Lions, Panthers, etc., all of which are cats.

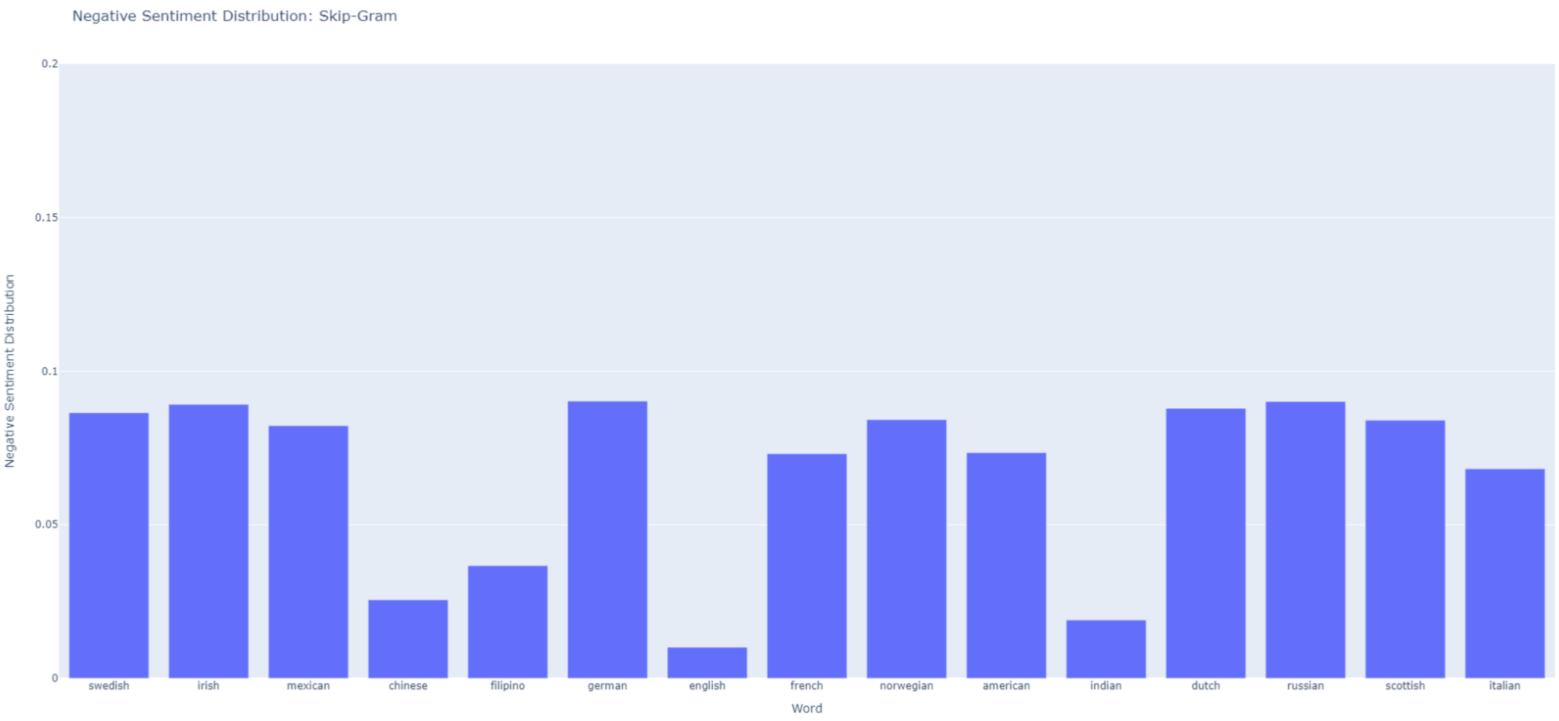
2. Bias:

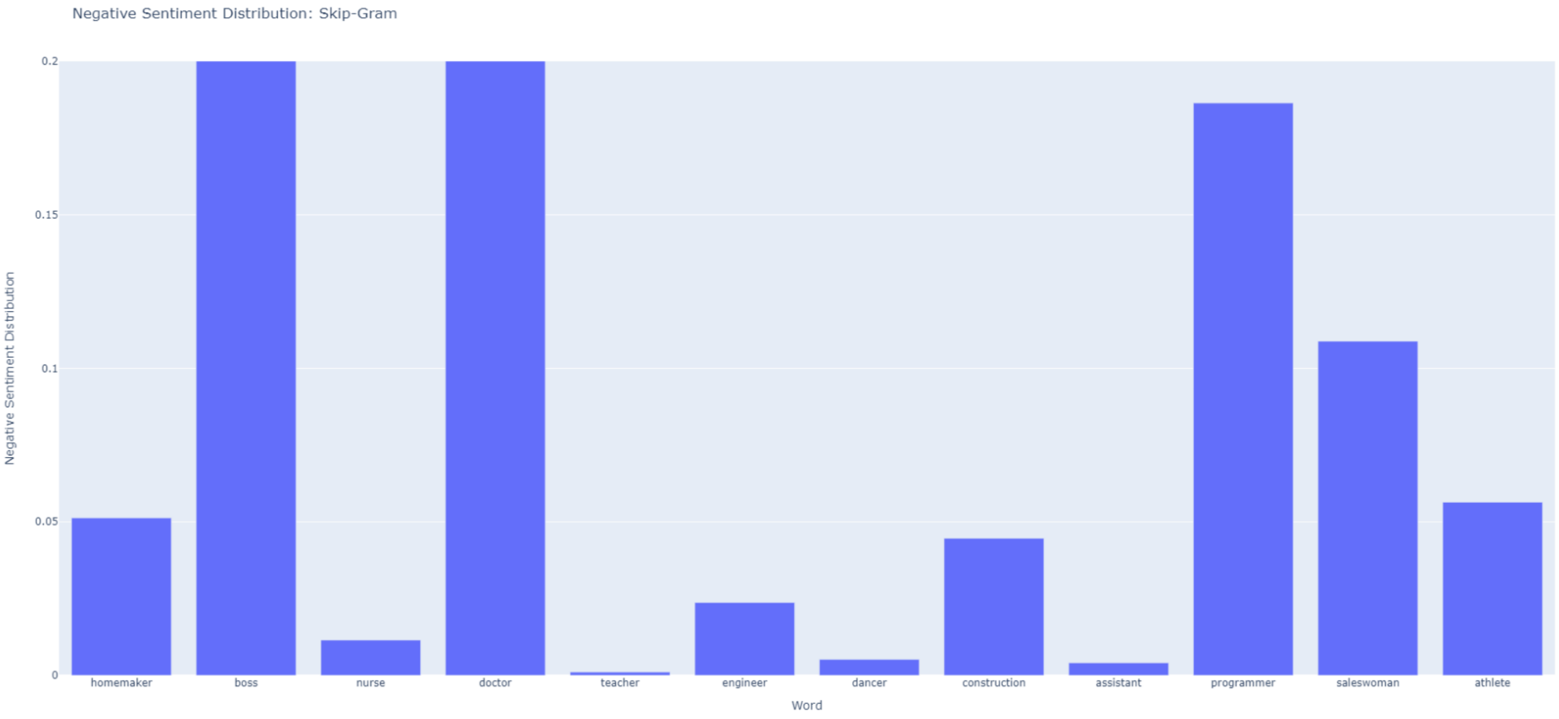
The bias study that I extended was the RNSB example. The original RNSB words that I used were different nationalities. The additional biased words that I also used in this analysis were different professions that are typically stereotyped to be dominantly held by a certain gender. Throughout each of these two queries, both the nationalities and professions, the negative sentiment distribution changes significantly depending on the word embedding model you are testing. As for the nationality query, the CBOW, Skip-Gram, and Google News models contained results that had around a 0.1 negative sentiment distribution for each different nationality. The only model that contained extremely high negative sentiment was the GloVe model, which showed great bias in relation to the Mexican and Indian nationalities. The one nationality that showed extremely little bias across each model was English, with some of the other European nationalities also resulting in some of the lowest levels of bias. The negative sentiment distribution for this query was not entirely expected. I initially thought that there would be a much higher and consistent rate of bias for the nationalities like Mexican, German, and Indian, where the actual results proved to be much tamer. When looking at the profession query, the overall amount of bias across the four models is significantly higher than what was seen in the nationality query. In each model, there are a couple professions that contain a large amount of negative sentiment. In the CBOW and Skip-Gram models, the results show a large bias towards the professions boss, doctor, programmer, and saleswoman. In the GloVe and Google News models, there exists a large amount of bias towards the professions assistant, construction, nurse, doctor, and teacher. Across each of the four models, the professions dancer, homemaker, teacher, engineer, and athlete contain very little amount of negative sentiment. These results were also quite surprising, as I specifically chose to include professions that are typically referred to as dominated by a certain gender. Additionally, as I look back on my sample query, the profession saleswoman may not be the best to signify negative sentiment, as the profession has the gender of its employee in the job title. For future testing, this should either be removed or changed to a profession regarding sales in general.

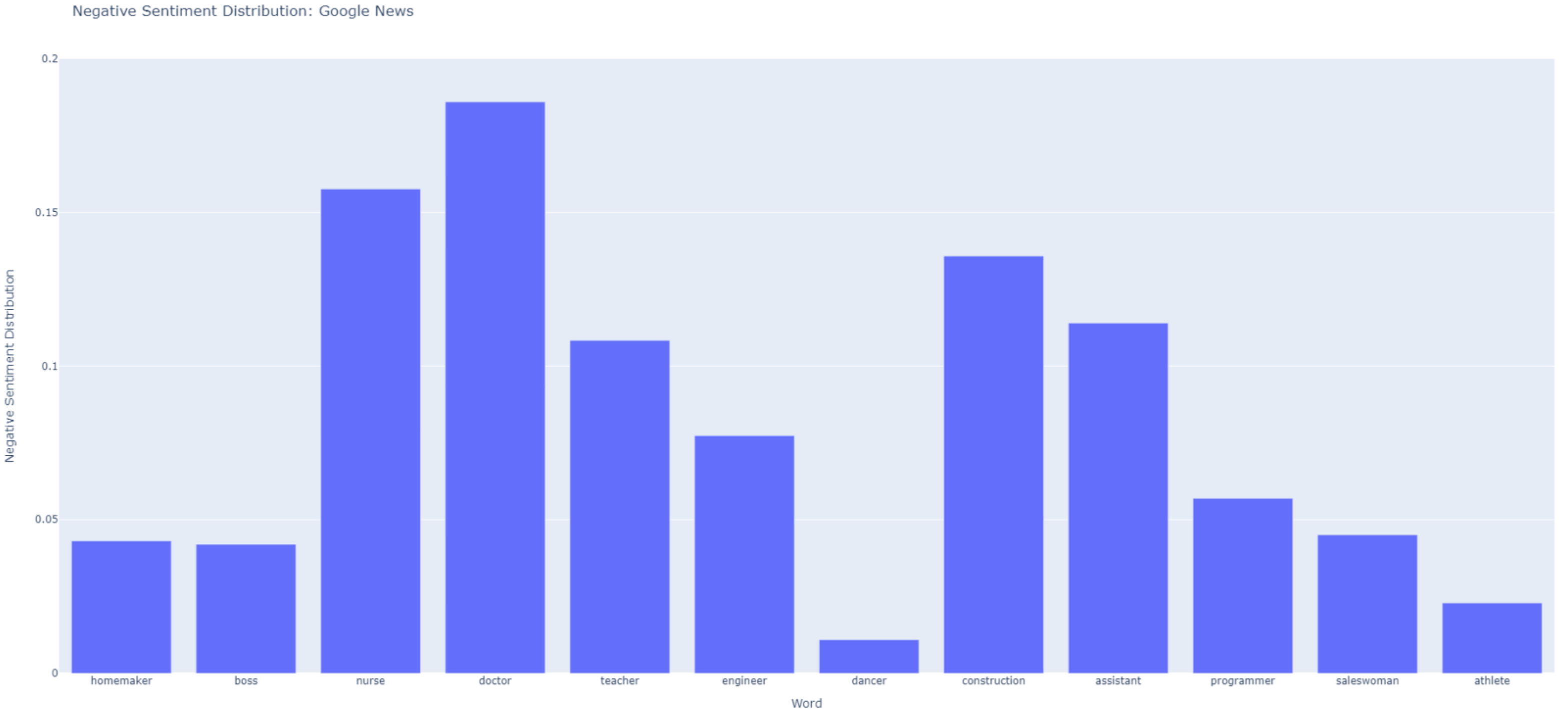
After analyzing the results of bias testing, I believe depending on the specific model being used in a machine learning system, there could be large consequences. In general, you want your machine learning model to train on data that is as bias free as possible, since this will have a large effect on the output information of the system. If you are only using a single model to train your machine learning system, this could result in a large bias issue. No matter the model being used in my analysis, each contained their own way of displaying large amounts of bias on certain topics. If only one model is used to train the system, you can expect the machine learning to also have its own data that is heavily biased. The way I believe you can get around this is by training the machine learning model on a great variety of data. One of the points in my analysis was that not all models contained bias towards the same topics. If you were to train your machine learning system on multiple models, this bias will be out weighted and could be eventually eliminated from the model in its entirety. You would just have to make sure the models you are training it on contain different points of bias, so that each model does not overlap and contain the same type of bias.

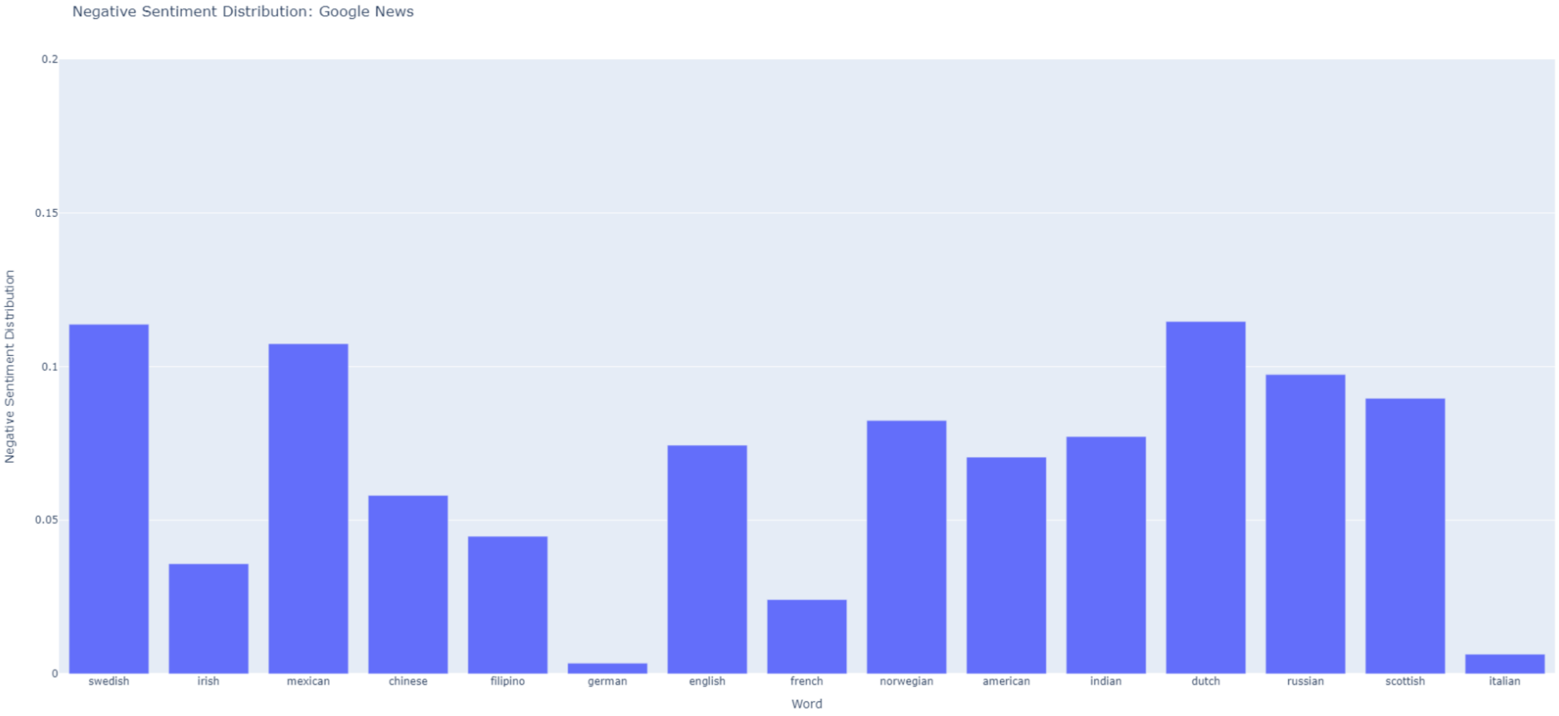


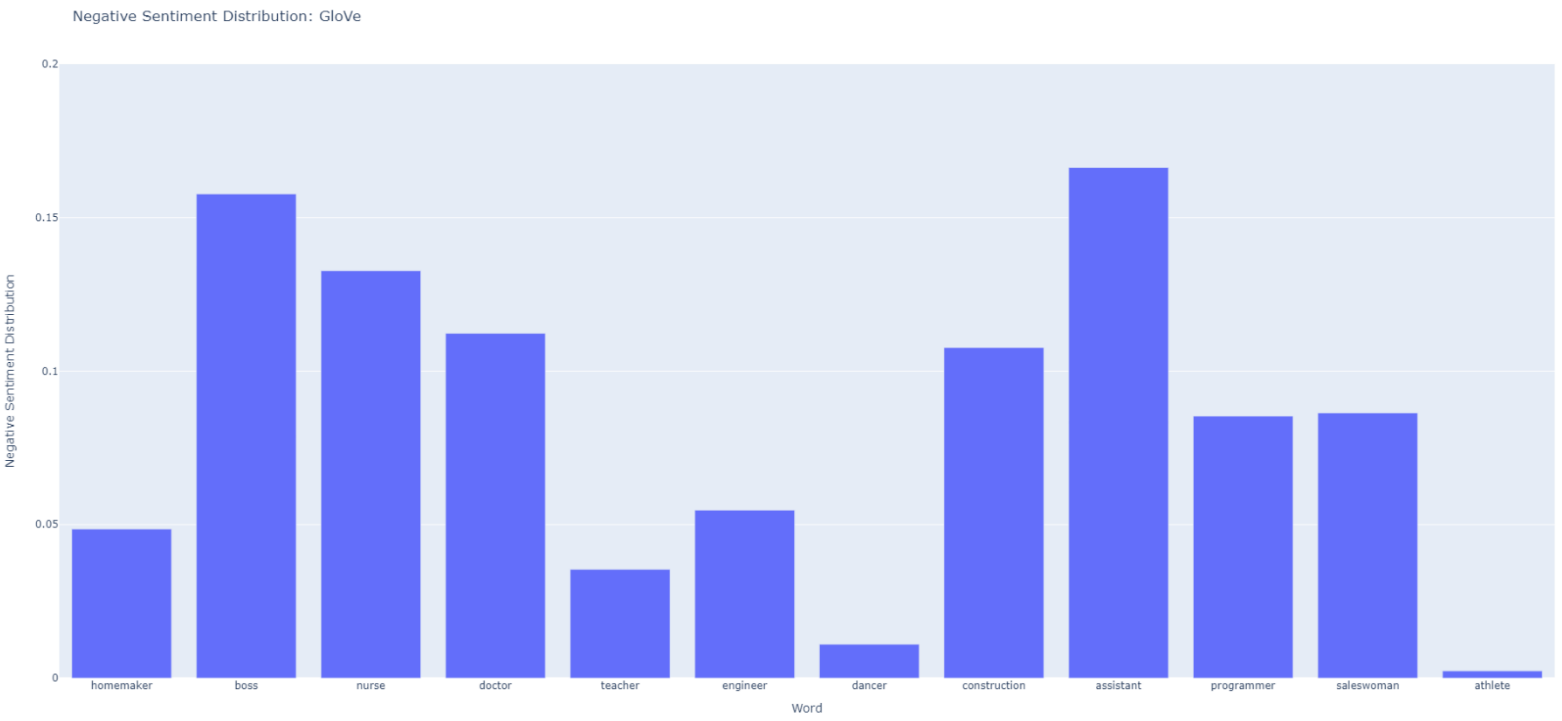


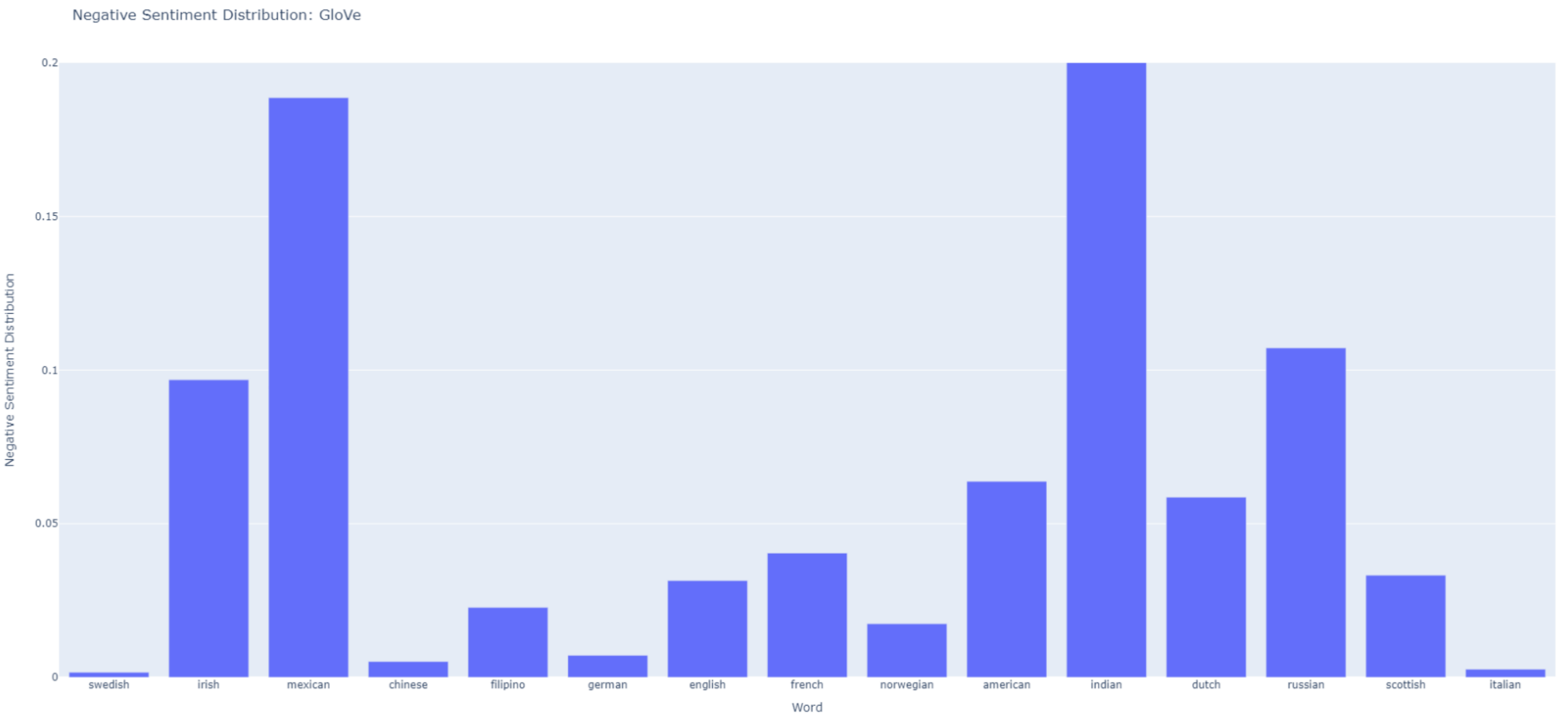

Negative Sentiment Distribution: Skip-Gram


Negative Sentiment Distribution: Skip-Gram


Negative Sentiment Distribution: Google News

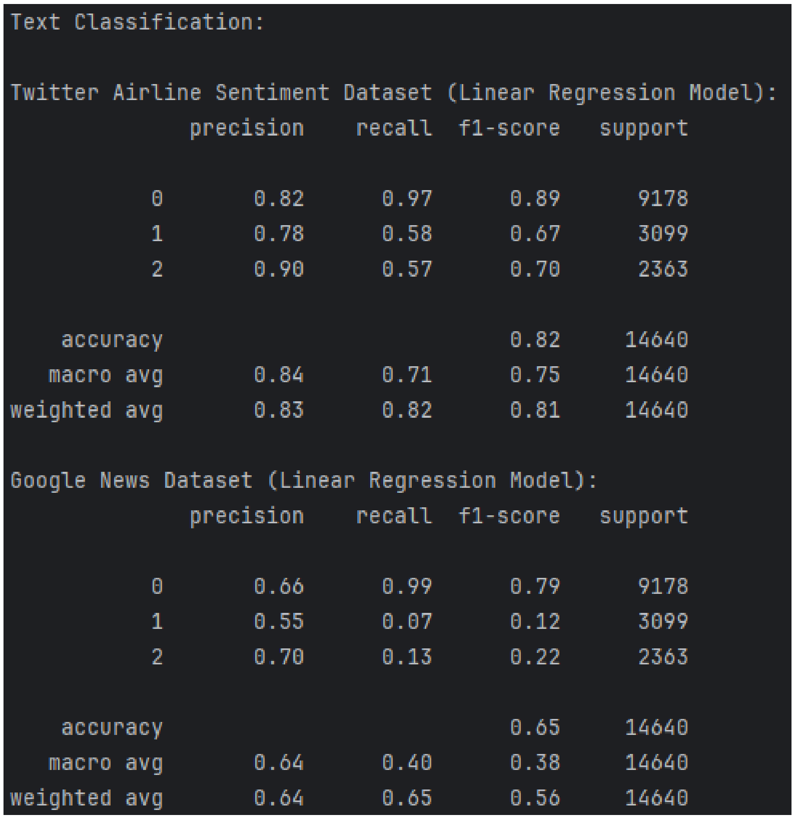

Negative Sentiment Distribution: Google News


Negative Sentiment Distribution: GloVe
Negative Sentiment Distribution
0.2
0.15
0.1
0.05
0
homemaker
boss
nurse
doctor
teacher
engineer
dancer
construction
assistant
programmer
saleswoman
athlete
Word


Negative Sentiment Distribution: GloVe
Negative Sentiment Distribution
0.2
0.15
0.1
0.05
0
swedish
irish
mexican
chinese
filipino
german
english
french
norwegian
american
indian
dutch
russian
scottish
italian
Word

3. Classification:

The embedding model that I selected to run the linear regression classification testing was the Google News model. I also used the Twitter Airline Sentiment dataset as a “test run” so that I could have something to compare with the results of the Google News model. As seen through the F1-scores throughout the Google News linear regression, this dataset had quite poor classification predictions. This is especially true when comparing these results to that of the Twitter Airline Sentiment results. Throughout all of the F1-scores for the Twitter dataset, the results maintained a correct prediction rate of around 0.75, when the F1-scores for the Google News model had a much greater difference between test sets. I believe the main thing that could cause such a great difference in classification predictability is the fact that the Google News dataset contains a much wider range of topics, when compared to the Twitter Airline Sentiment data. Since the Google News dataset is so much broader, predictability metrics can be greatly swayed because there are so many possible options of what words could possibly come next.



```
Text Classification:

Twitter Airline Sentiment Dataset (Linear Regression Model):
              precision    recall  f1-score   support

           0       0.82      0.97      0.89      9178
           1       0.78      0.58      0.67      3099
           2       0.90      0.57      0.70      2363

    accuracy                           0.82     14640
   macro avg       0.84      0.71      0.75     14640
weighted avg       0.83      0.82      0.81     14640

Google News Dataset (Linear Regression Model):
              precision    recall  f1-score   support

           0       0.66      0.99      0.79      9178
           1       0.55      0.07      0.12      3099
           2       0.70      0.13      0.22      2363

    accuracy                           0.65     14640
   macro avg       0.64      0.40      0.38     14640
weighted avg       0.64      0.65      0.56     14640
```

4. Reflection:

One of the most challenging parts of this assignment was getting familiar with the libraries to do some of the complex model training and comparisons. All of these libraries I am still quite unfamiliar with, mainly due to my lack of Python experience, so each assignment has come with a large learning curve where I have to figure out which library to use, and how to use them in the particular way I want to. Other than the mathematical topics covered in this assignment, the main thing I learned were these various Python libraries. Additionally, another challenge I faced was the runtime of the program and my laptop's memory. Due to the large datasets being loaded for the computations, the runtime of the program was initially extremely long. In order to troubleshoot my code and see what exactly is being printed out, I needed to figure out a way to organize my code so that not as many datasets are loaded at the same time. At first, when I would try and run the program, the runtime was so long I could not get to the point where the graphs would be created. What would happen is that a new tab would open for the graphs to be plotted but would eventually time out due to long execution time. I had to make several adjustments to the program's organization in order to only load the files needed at that particular time, in order to significantly lessen the runtime of the application.