Sentiment Analysis Using Twitter Data

A sentiment prediction algorithm for tweets scraped from the Twitter API

Capstone 3 Presentation
Damilola T. Olaiya

Proposal

- Hypothesis → How can a given company use tweets about itself and its products or employees to (i) gauge overall sentiment about its products, services, employees or overall branding to make decisions about demand and/or marketing?

- Criteria for success
  - Creating a model that can accurately predict the [sentiment of tweets about a given entity scraped from Twitter's API.

Data Wrangling

- The dataset contains 1.6 million tweets and was obtained from kaggle.

- There were 6 features/columns which were:
  - ids
  - date
  - flag
  - user
  - text
  - sentiment

- The target feature is sentiment which was the polarity of the tweet
  - (0 = negative, 4 = positive)

Data Wrangling

- The entries/samples in the data evenly corresponded to 800K entries each for positive and negative sentiment.

- There were no missing or duplicated values. The data was clean.

- This suggested that there were no tweets with exactly the same information from exactly the same users on exactly the same dates.

Data Wrangling

- To improve readability, the positive sentiment values were mapped from 4 to 1.

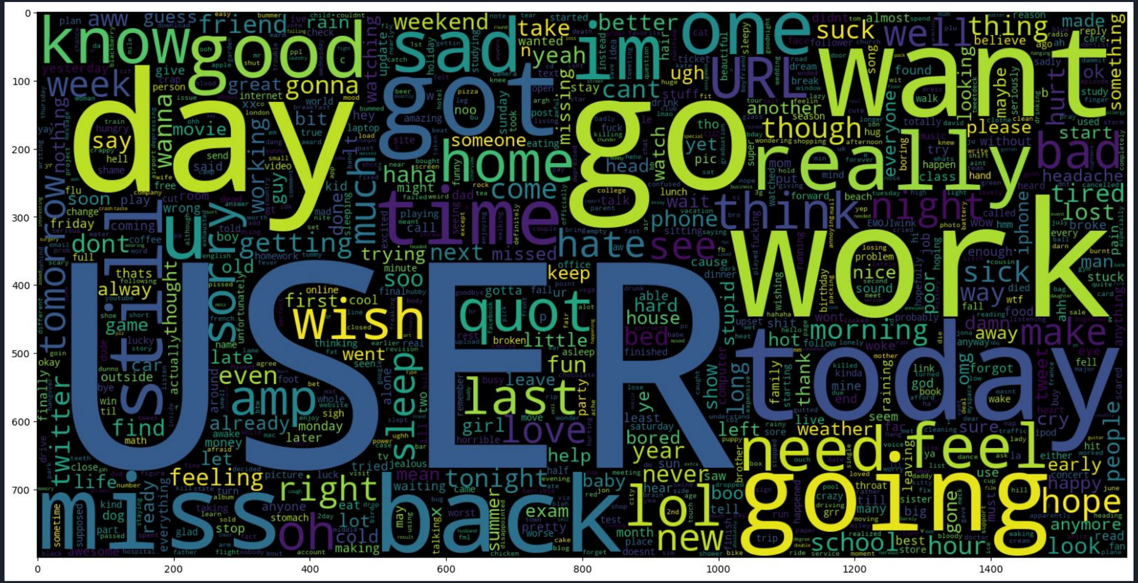
- Also, all contractions were “de-contracted” eg
  - n’t to not
  - ‘ll to will
  - ‘d to would
  - ‘s to is
  - ‘re to are
  - ‘ve to have
  - ‘m to am

Data Wrangling

- It should be noted that stop words were left in the corpus although there is the option of removing them.
    - Model tuning suggested that accuracy was worsened by the removal of stop words.
    - This is because stop words include words that indicate negation (e.g. can't, wouldn't, not, don't etc) which strongly affect sentiment.

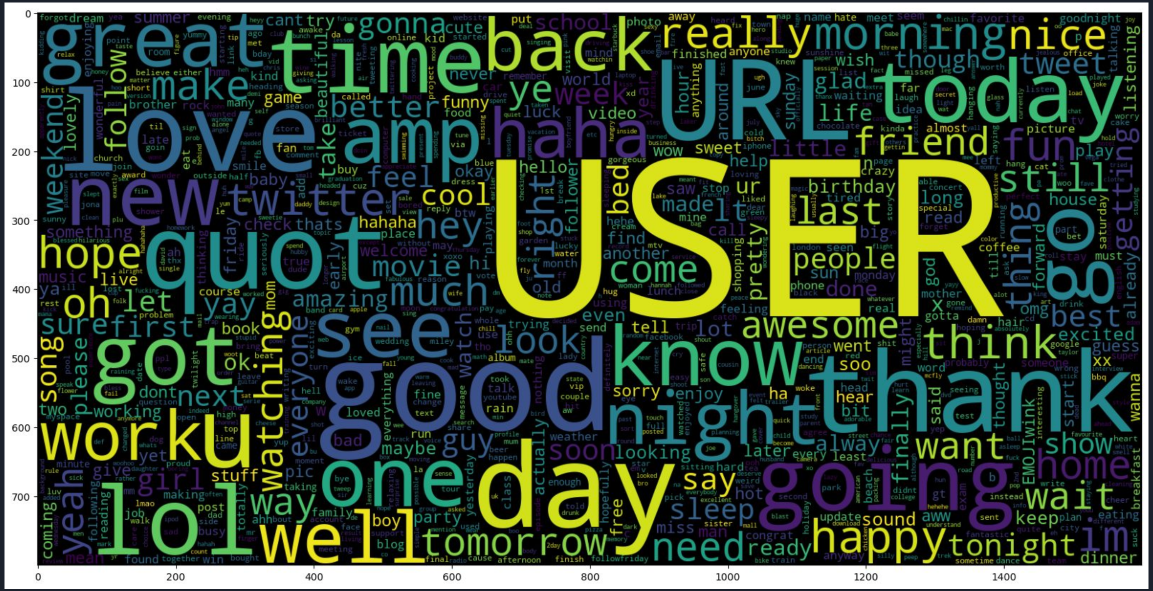
Exploratory Data Analysis (EDA)

- A word cloud was generated for tweets with a negative sentiment.



Exploratory Data Analysis (EDA)

- A word cloud was generated for tweets with a positive sentiment.



Pre-processing and Training

- All features besides the text and sentiment were dropped.
- The data was split into X_train, X_test, y_train and y_test.
- It was then transformed using Tfidf Vectorization.
- The vectorizer included both single words and bigrams.
- The vectorizer was set to use only the 500K most populous features/words.
- Note that the vectorizer was fit and transformed using X_train only then used to transform X_test. This was done so that the model would be completely unaffected by the testing data.

Modeling

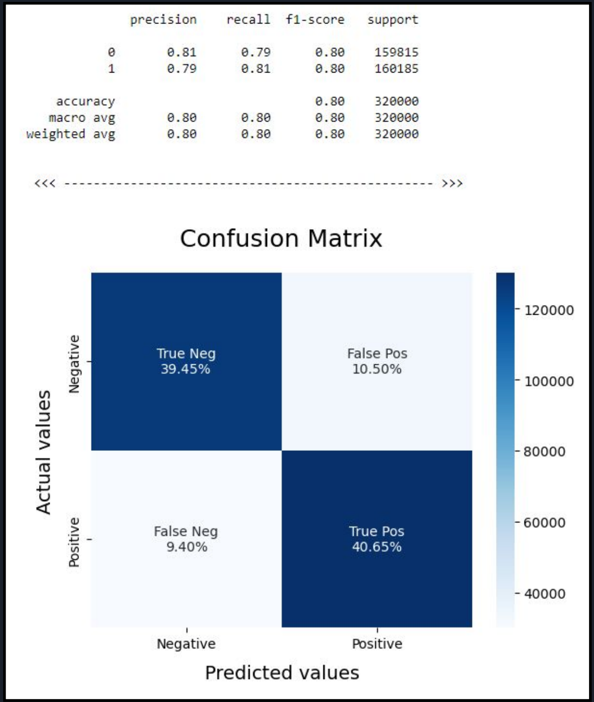
- The following models were be used:
  - Bernoulli Naive Bayes (BernoulliNB)
  - Linear Support Vector Classification (LinearSVC)
  - Logistic Regression (LR)

- Since our data was not skewed, accuracy was chosen as the evaluation metric.

- Confusion matrices and Classification Reports were used to get an understanding of how our models were performing in both classes.

Modeling

- Bernoulli Naive Bayes model results

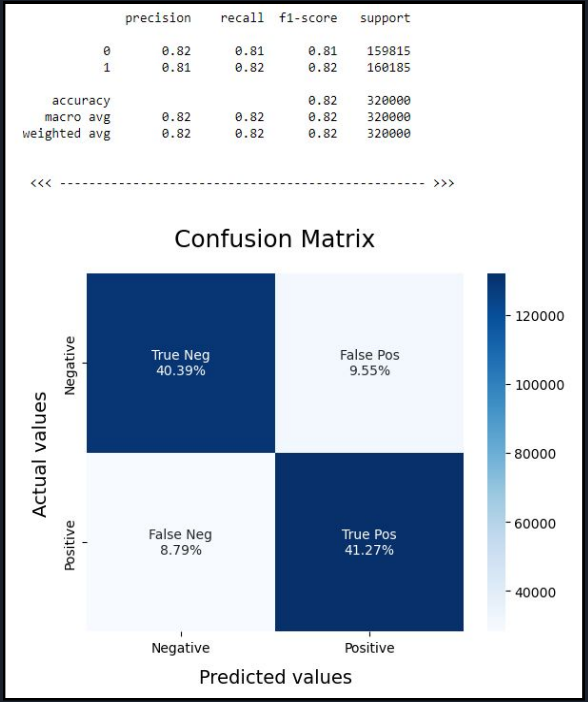


```
              precision    recall  f1-score   support

           0       0.81      0.79      0.80    159815
           1       0.79      0.81      0.80    160185

    accuracy                           0.80    320000
   macro avg       0.80      0.80      0.80    320000
weighted avg       0.80      0.80      0.80    320000

 <<< ------------------------------------------------ >>>
```

Confusion Matrix

Modeling

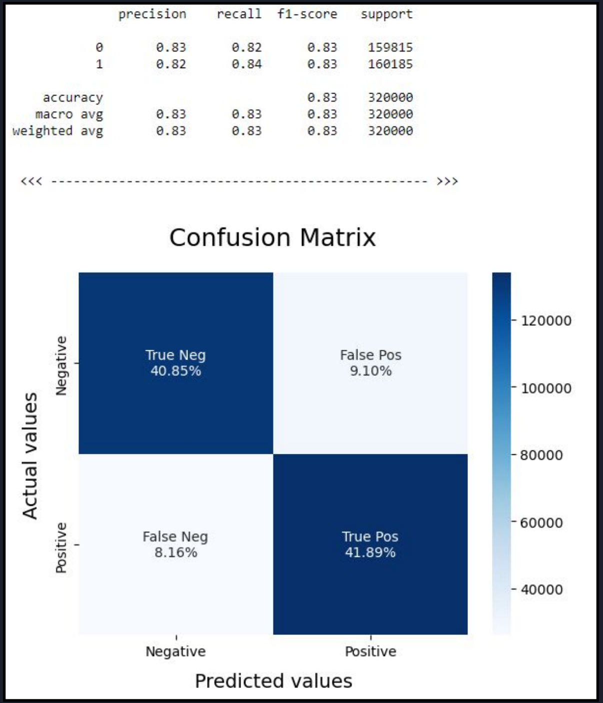
- Linear SVC model results



```
              precision    recall  f1-score   support

           0       0.82      0.81      0.81    159815
           1       0.81      0.82      0.82    160185

    accuracy                           0.82    320000
   macro avg       0.82      0.82      0.82    320000
weighted avg       0.82      0.82      0.82    320000

 <<< ---------------------------------------------- >>>
```

Modeling

- Logistic Regression model results



```
              precision    recall  f1-score   support

           0       0.83      0.82      0.83    159815
           1       0.82      0.84      0.83    160185

    accuracy                           0.83    320000
   macro avg       0.83      0.83      0.83    320000
weighted avg       0.83      0.83      0.83    320000


<<< -------------------------------------------------- >>>
```

Inference and Conclusion

- The Logistic Regression model performed the best out of all the different models that were tried. It achieved 83% accuracy.

- This is followed by the LinearSVC model with 82% accuracy and the BernoulliNB model with 80% accuracy.

- All three models performed adequately but Logistic Regression was the best performer and we will proceed with it.

Further Steps to Consider

1. Using a larger dataset
2. Using regularization
3. Using cross validation on model parameters
4. Using real time scrapping of Twitter's API and adjusting the model using batch machine learning
5. Using more models
6. Excluding stop words