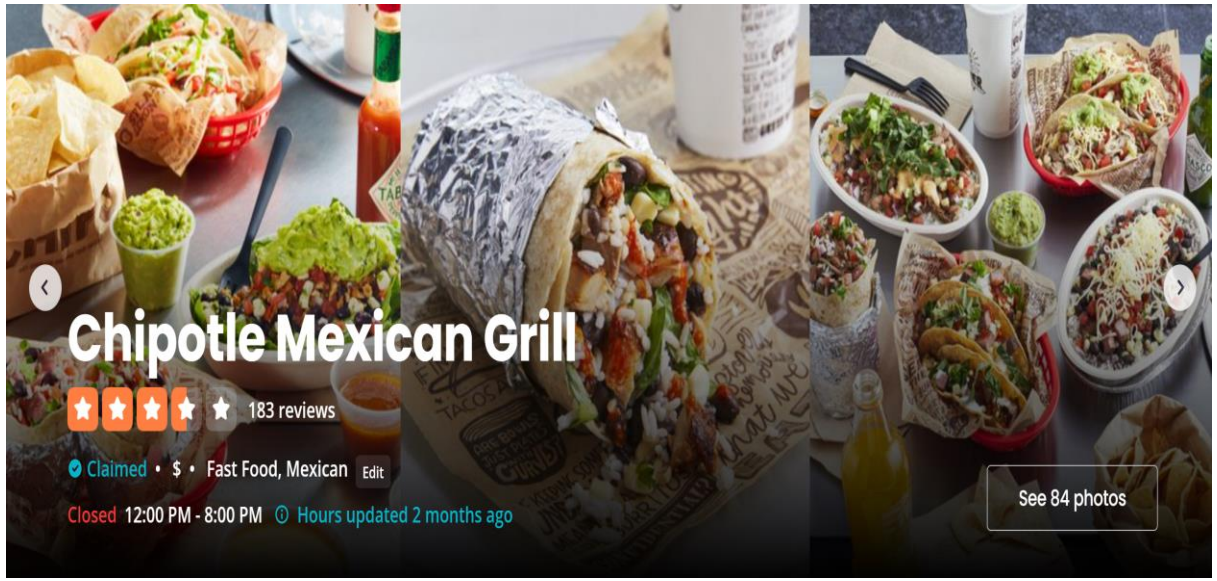
Unsupervised Sentiment Analysis of Yelp Reviews Using Natural Language Processing

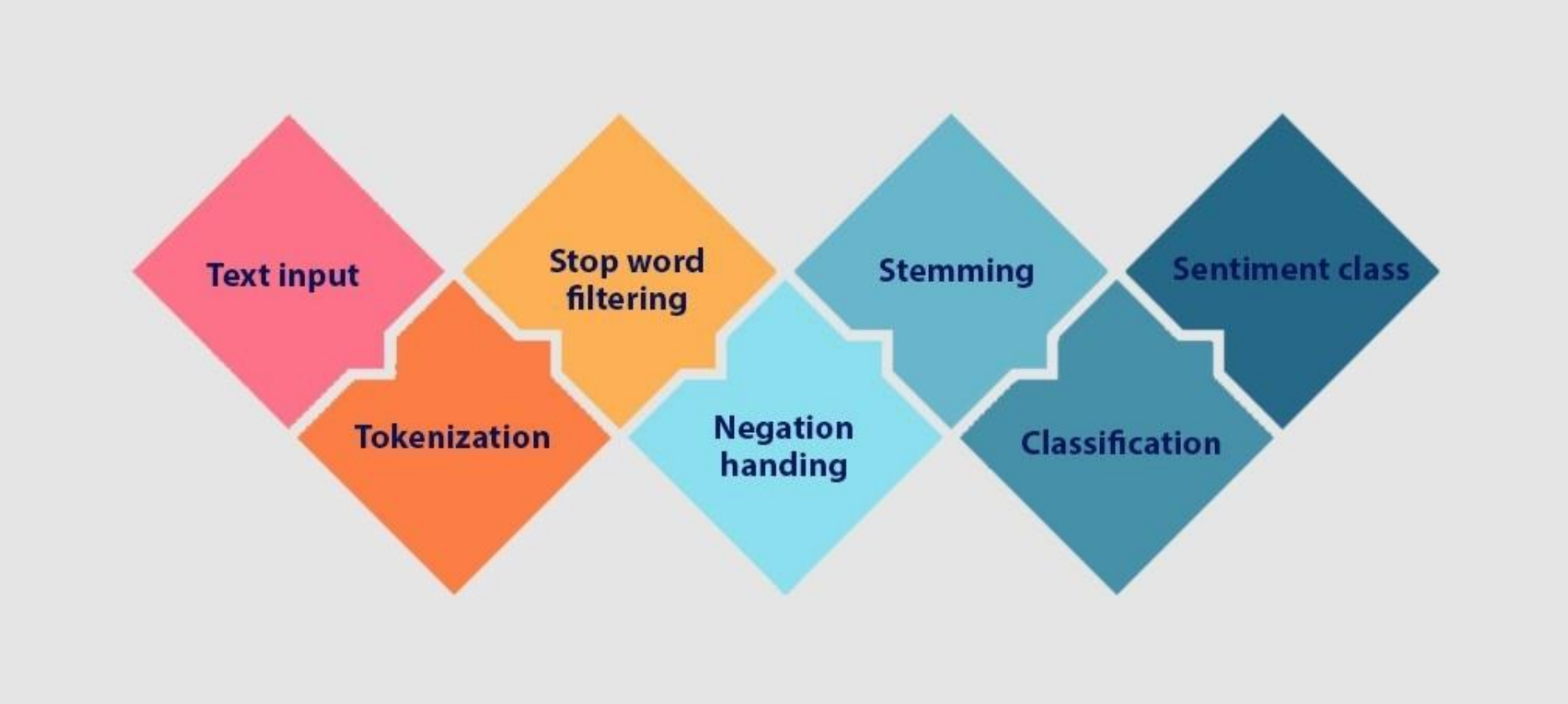
Prepared by: Rabia Tariq
10/19/2021

Problem

- Online reviews are one of the most common ways for a customer to share their opinion on a service or a product.
- With large amount of data, it is very difficult for a human to read through all the reviews.
- Most websites have 5-star rating system, which can be inconsistent.
- We can build a machine learning model to determine the sentiment based on the text within the review.



Sentiment Analysis Workflow



Data Wrangling

- Used Yelp API to access the URLs for each restaurant.
- Used NYC Open Data to create a list of 190+ restaurants in New York.
- Used python to scrape reviews from each restaurants.
- Made the reviews more readable by filtering out html script using BeautifulSoup.
- Ended with approximately 10,000 reviews from 190+ restaurants.

clean_reviews

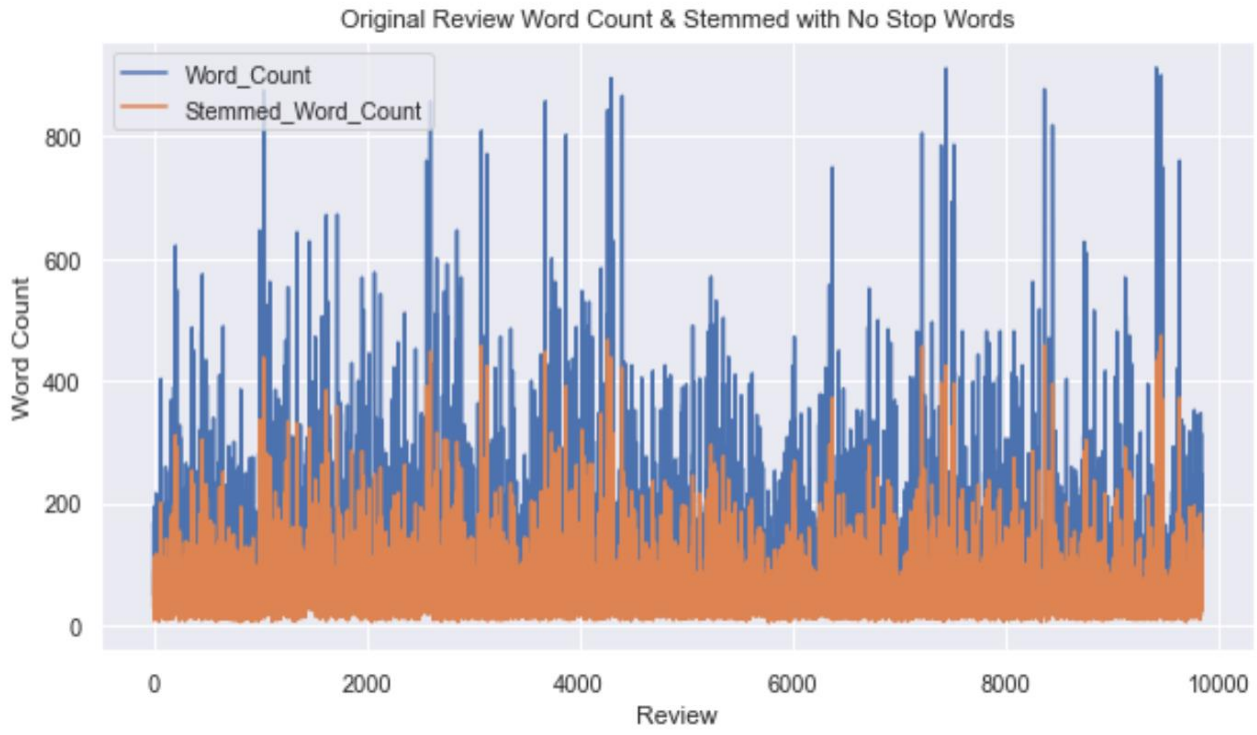
['During Cinco de Mayo and National Nurses Week in May, I wasnt surprised to see two of Chipotles All-Stars, Murphy and Fran k, providing a welcoming environment in their financial district location. As a Health Care Worker, Chipotle has offered VIP treatment to frontline workers throughout the pandemic and we will not forget their kindness and generosity.Thank you for an other free "Burrito Day" during Nurses Week!',

Word Cloud

- Created a word cloud to represent text data in a way where the size of each word indicates its frequency or importance.

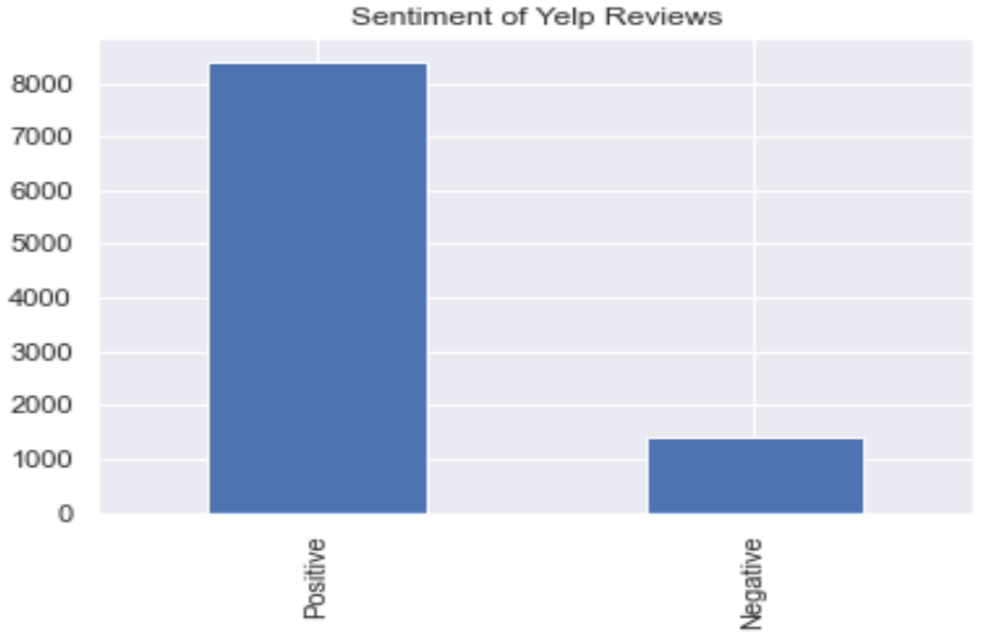
Stemming

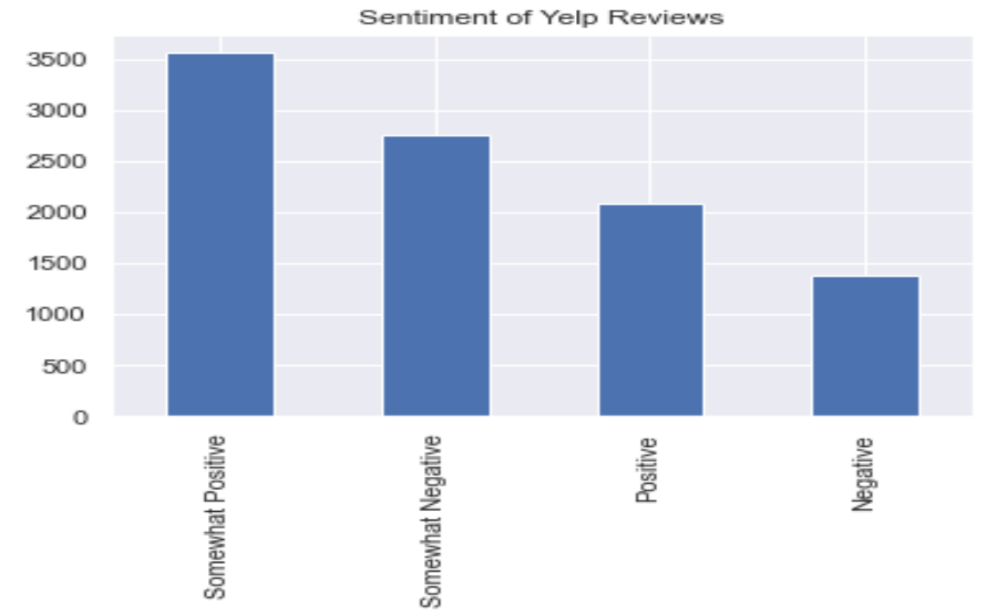
- Removed stop words (the, and, an, for, that, etc.) to save time and computing time.
- To reduce the ambiguity for a machine learning algorithm, its is important to implement stemming to reduce the words to their base forms and make sure all word forms are conjugated the same.
- The plot shows the total word count of the reviews in blue and with stemming in orange.



Classification/Labeling

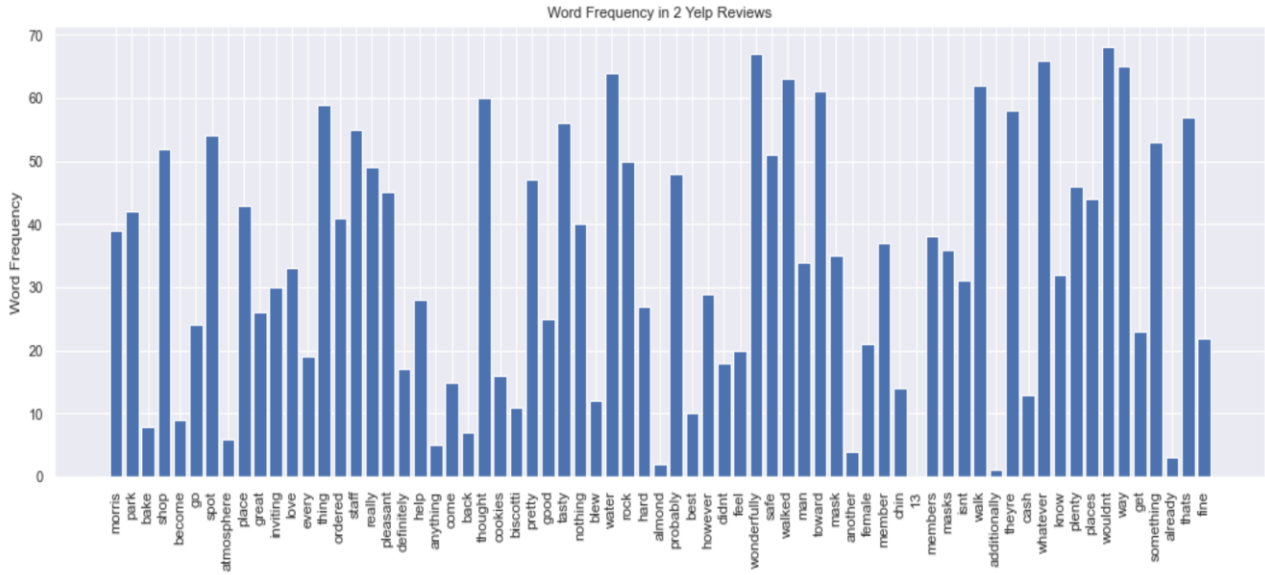
- The data we obtained from web scraping was unlabeled.
- We had to assign the label/sentiment to our reviews using TextBlob's Polarity Value
- Uses Text2Vector to compare word vectors and find the most similar to known positive and negative word vectors
- We first split the reviews into just 2 classes but ended up splitting into 4 classes





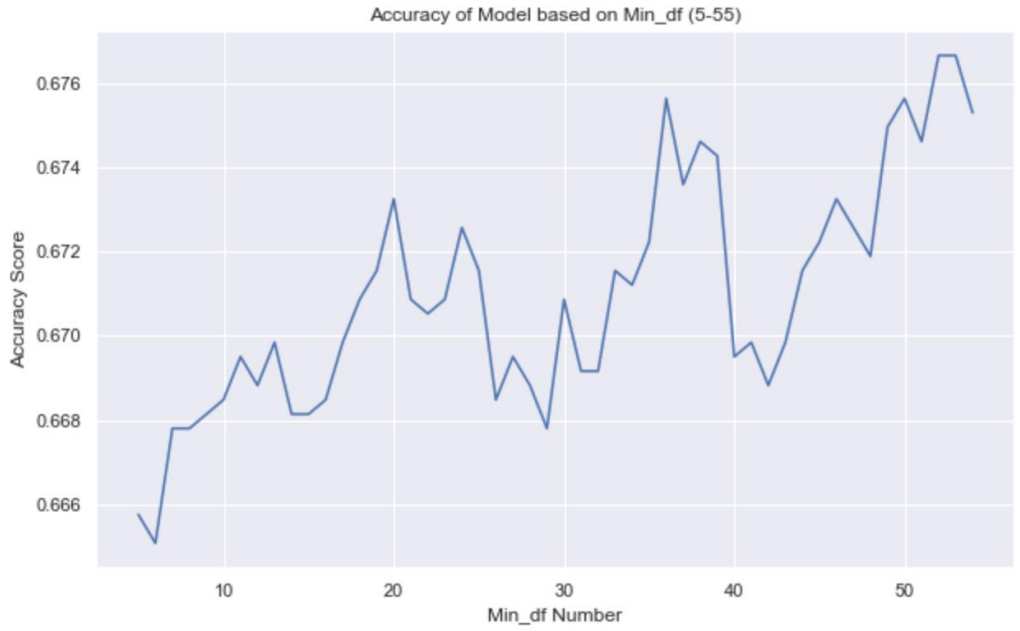
Feature Processing

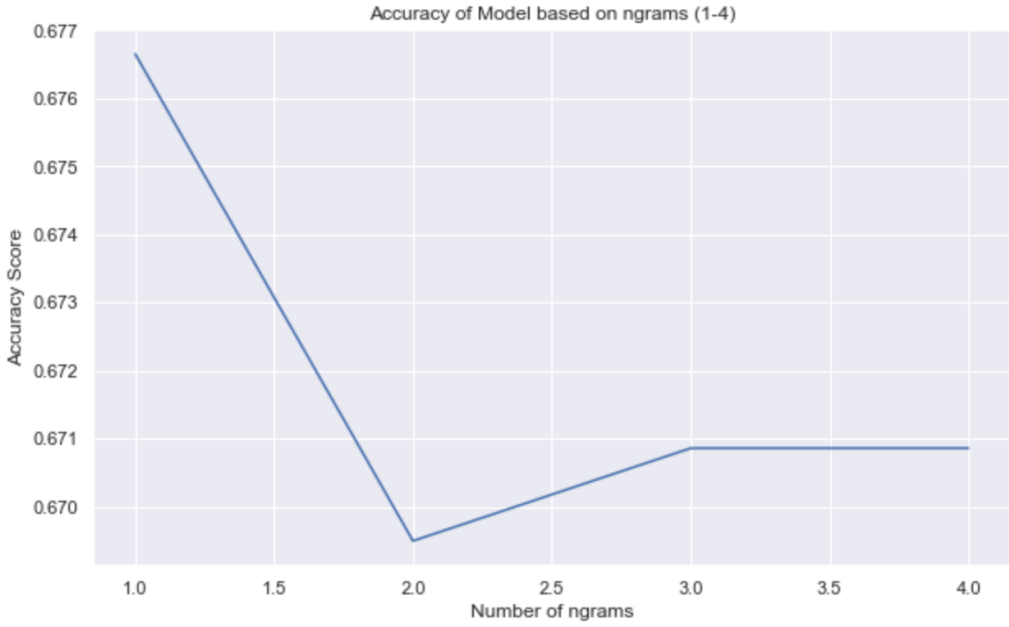
- **TF-IDF:** Term Frequency Inverse Document Frequency takes a count of words for all reviews and assigns a score based on how often they appear in one review.
- Selected different number of n-grams to see if accuracy of model increases.
- To avoid overfitting, we also checked for different minimum word frequency in a document.



Feature Processing

- We set the minimum word frequency in a document to 52.
- Reduced the number of features from 10,000+ to around 1,600.





Model Selection

- Tested 9 models for the sentiment analysis.
- **Models using extracted features:**
  - Logistic Regression
  - Random Forest
  - Gradient Boosted Classifier
  - Support Vector Classifier
- **Word Embedding: Bag of Words:**
  - Naïve Bayes with TF-IDF
  - Naïve Bayes with CountVectorizer
  - Gradient Boosted Classifier with TF-IDF
- **Ensembles:**
  - Naïve Bayes Probability with Dense/Sparse
  - Stacked Model (GBC + NB)

	Model	Accuracy	F1_Macro	F1_Micro	F1_Weighted
1	Stacked Model	0.906	0.900	0.886	0.926
3	SVC	0.693	0.693	0.693	0.692
2	GBC	0.691	0.695	0.691	0.690
0	Logistic Regression	0.690	0.690	0.690	0.689
1	Random Forest	0.639	0.647	0.639	0.639
1	NB_CV	0.589	0.594	0.589	0.586
2	GBC_TF	0.583	0.591	0.583	0.582
0	NB_TF	0.581	0.562	0.581	0.568
0	NB Probability Dense/Sparse	0.581	0.590	0.581	0.576

Stacked Model

- Stacked model was a combination of 2 models:
  - Naïve Bayes with Dense/Sparse text matrix as input.
  - Gradient Boosted Classifier with TFIDF vectors and percentage of positive words as input

- Used Naïve Bayes Model to calculate the probability of the sparse matrix being either Positive, Somewhat Positive, Somewhat Negative and Negative.

- Added this as a feature for our Gradient Boosted Classifier

Conclusion

- With our stacked model we were able to improve the accuracy from 68% to 90%, which is a 22% increase in the accuracy.

- Sentiment analysis has major business implications and could be very useful for any service that receives mostly test-based reviews

- The potential for natural language processing is incredibly high. This sentiment analysis shows just one of the many applications of utilizing text-based data.