Sentiment Analysis Methods

An Evaluation of NLP Techniques for Sentiment Analysis

- What is Sentiment Analysis
- **In the context of this paper, Sentiment analysis is a method of classifying "*sentiment*" of a corpus of text.**

  (Def.) Sentiment -- a. **:** an attitude, thought, or judgment prompted by feeling, :

  b.: a specific view or notion :

  Sentiment – human response to a body of text given context (ie., opinion on matter/thing/event [Eg., a review].
- Number of Factor levels: binary (good, bad), 3-factor (good,bad,neutral), etc.

NLP – Natural Language Processing

NLP Methods (for text representation/feature extraction):

- BoW – Bag of Words
- TF-IDF – Term Frequency – Inverse Document Frequency

ML (Machine Learning) methods (classification/supervised learning):

- Random Forest Classifier
- Naive Bayes

**** All methods above are in Python libraries and are readily available to implement ****

Datasets (from kaggle):

Youtube Comment Datasets:
https://www.kaggle.com/datasets/atifaliak/youtube-comments-dataset

Twitter dataset: (with addional field on context (topic tweet is about):
https://www.kaggle.com/datasets/jp797498e/twitter-entity-sentiment-analysis

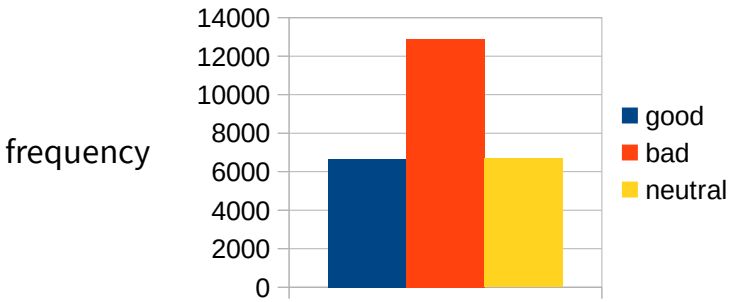
Chat gpt sentiment analysis on twitter:
https://www.kaggle.com/datasets/charunisa/chatgpt-sentiment-analysis

market sentiment of financial commentaries:
https://www.kaggle.com/datasets/sbhatti/financial-sentiment-analysis

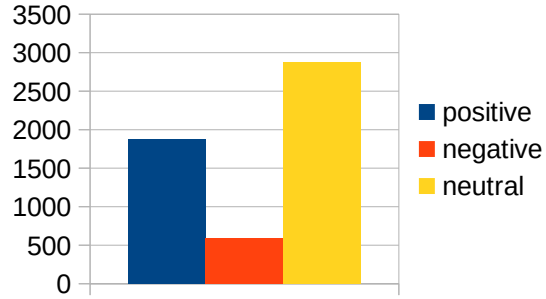
IMDB ratings sentiment on comments from rotten tomatoes:
https://www.kaggle.com/datasets/yasserh/imdb-movie-ratings-sentiment-analysis

Datasets (sentiment frequency histograms):

1. youtube comments

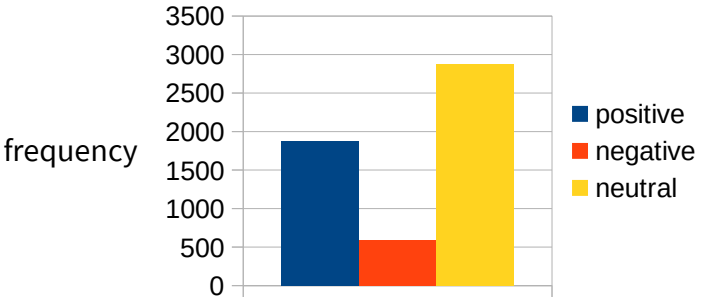


2. chatgpt comments

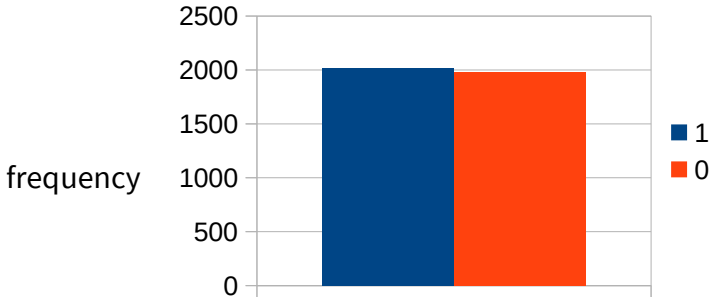


Datasets (sentiment frequency histograms, continue...):

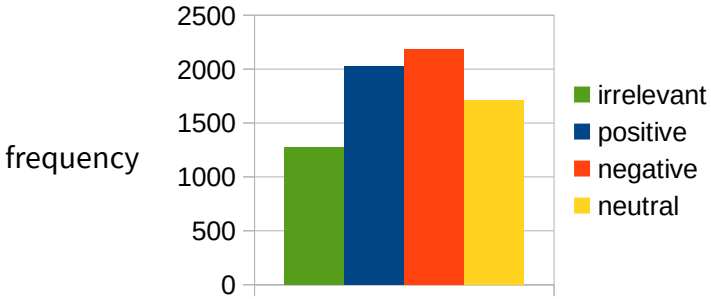
3. financial sentiment



4. imdb (movie sentiment)



5. twitter comments



Pre-processing Step:
Data cleaning and pre-processing of text corpus:

- ***Cleaning (1)*** – Remove Special characters, drop missing values/duplicates
- ***Cleaning (2)***– Drop non-English comments, drop english 'stop' words *
- ***Normalization*** – convert text to standard form (ie. all lowercase or uppercase)
- ***Lemmanization*** – convert inflected words into standard root form**
- ***Tokenization*** – split text into words and/or phrases (1-gram and 2-grams)
- ***Vectorization*** – transformation of tokens into the vector space
  that can be processed by the ML method for prediction.

*article, common proposition words (ie, a, the, an, of, etc etc)
** eg. convert running, ran, or runs into root word: run (Provisionally)

Tools:
Language: Python, R
Python Libraries: (nltk, scikitlearn, keyBERT, etc) [for modeling]
R libraries: R base, [for statistics and hypothesis testing]
ggplot [for graphical plots]

Evaulation:

- Split 80/20 training/test dataset
- Consists of 100 runs per dataset
- Accuracy Score from sklearn.metrics

Evaluation Results:

Dataset 1 (Youtube):
no features: 3 (positive, negative, neutral)
number of rows: 14577
Dimension (no tokens[words]): 27029

Pre-processing method	ML model technique	Avg. Accuracy (100 runs)
BoW	Rforest	70.77%
BoW	NBayes	73.17%
TF-IDF	Rforest	70.61%
TF-IDF	NBayes	66.0%

Dataset 2 (chatgpt 12% sample):
no features: 3 (good, bad, neutral)
number of rows: 25149
Dimension (no tokens[words]): 55522

Pre-processing method	ML model technique	Avg. Accuracy (100 runs)
BoW	Rforest	70.38%
BoW	NBayes	67.53%
TF-IDF	Rforest	68.63%
TF-IDF	NBayes	60.41%

Dataset 3 (financial):
no features: 3 (positive, negative, neutral)
number of rows: 4425
Dimension (no tokens[words]): 10622

Pre-processing method	ML model technique	Avg. Accuracy (100 runs)
BoW	Rforest	69.28
BoW	NBayes	70.98
TF-IDF	Rforest	68.61
TF-IDF	NBayes	68.43

Dataset 4 (movie 10% sample):
no features: 2 (0,1)
number of rows: 3998
Dimension (no tokens[words]): 41246

Pre-processing method	ML model technique	Avg. Accuracy (100 runs)
BoW	Rforest	73.05%
BoW	NBayes	82.83%
TF-IDF	Rforest	71.9%
TF-IDF	NBayes	83.72%

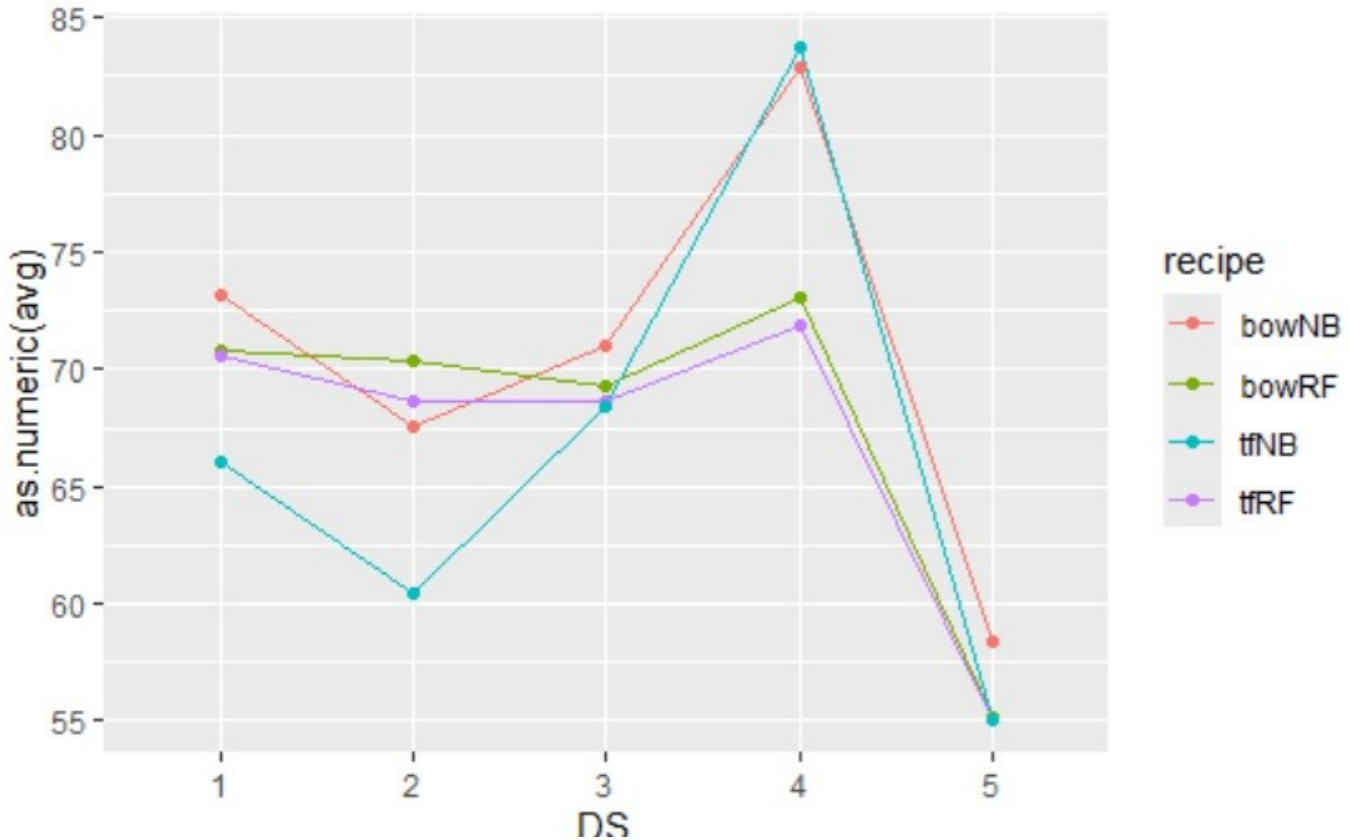
Dataset 5 (twitter):
no features: 4 (positive, negative, neutral,irrelevant)
number of rows: 5953
Dimension (no tokens[words]): 13129

Pre-processing method	ML model technique	Avg. Accuracy (100 runs)
BoW	Rforest	55.12%
BoW	NBayes	58.36%
TF-IDF	Rforest	55.08%
TF-IDF	NBayes	55.09%

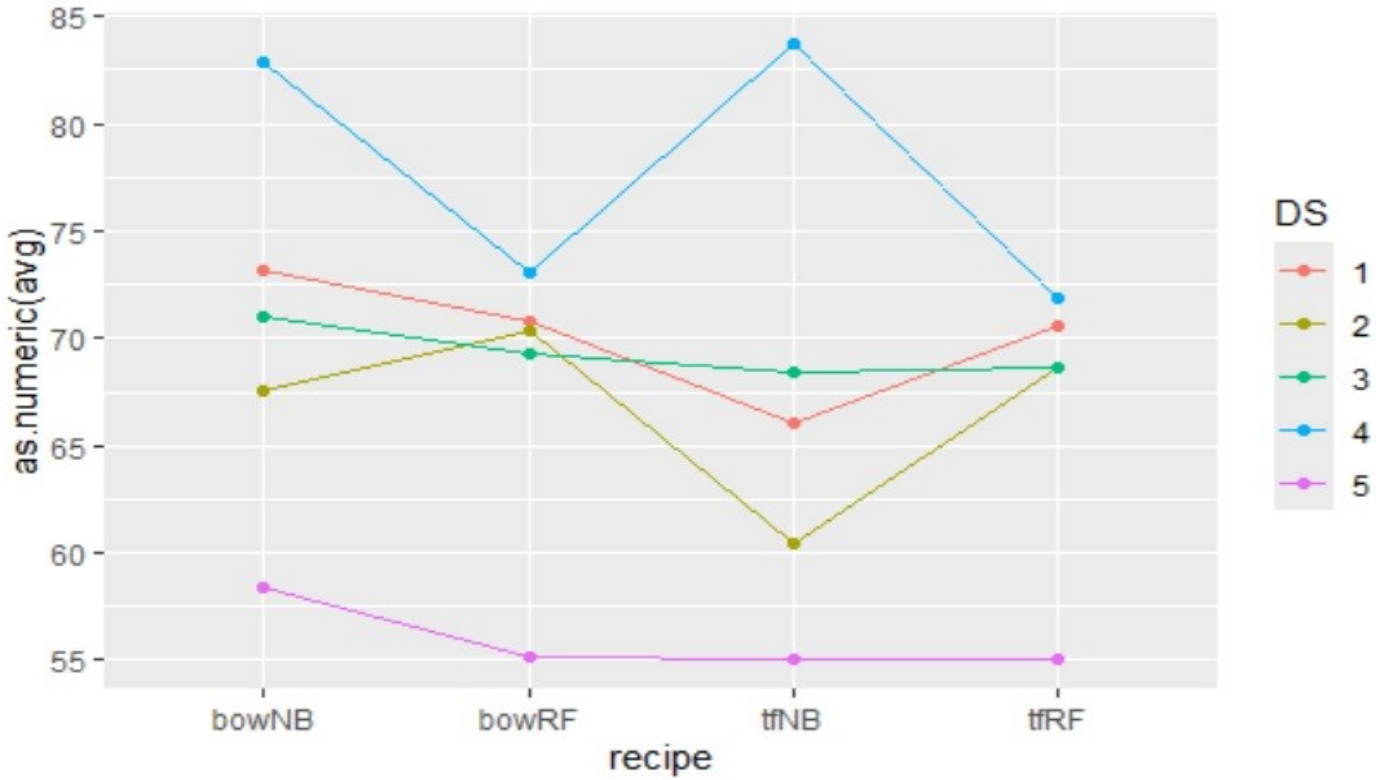
Aggregated Average Results:

Aggregated results:	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5	Avg.
bow/RF	70.77	70.38	69.28	73.05	55.12	67.72
bow/NB	73.17	67.53	70.98	82.83	58.36	70.57
TFIDF/ RF	70.61	68.63	68.61	71.9	55.08	66.96
TFIDF/ NB	66.0	60.41	68.43	83.72	55.09	66.73
Avg.	70.14	66.74	69.32	77.87	55.91	*68.0%* ← total avg accuracy

Plot of avg accuracy over DS (dataset) grouped by method



Plot of average accuracy over methods grouped by Dataset



Shapiro-wilks test p-value for each of the run (n=100)

P-VALUE*	bow/ RF	bow/ NB	tf/RF	tf/NB
DS1 (youtube)	0.1795	0.6521	0.6814	0.0947
DS2 (chatgpt)	0.1581	0.5887	0.8558	0.9345
DS3 (financial)	0.8799	0.6213	0.6201	0.0685
DS4 (movies)	0.2034	0.1402	0.7594	0.3659
DS5 (twitter)	0.4933	0.5722	0.3854	0.0539

* All p-values > 0.05 indicate all runs samples of 100 elements come from a normally distributed population.

Pairwise Welch t-test by method

Recipe pair/dataset	1	2	3	4	5
bowRF – bowNB	T	T	T	T	T
bowRF – tfRF	F	T	T	T	F
bowRF – tfNB	T	T	T	T	F
bowNB - tfRF	T	T	T	T	T
bowNB - tfNB	T	T	T	T	T
tfRF - tfNB	T	T	F	T	F

Pairwise Welch t-test by dataset

Dataset Pair/Recipe	bow/RF	bow/NB	tf/RF	tf/NB
1(youtube) – 2 (chatgpt)	T	T	T	T
1(youtube) – 3 (financial)	T	T	T	T
1(youtube) – 4 (movies)	T	T	T	T
1(youtube) – 5 (twitter)	T	T	T	T
2 (chatgpt) – 3 (financial)	T	T	F	T
2 (chatgpt) – 4 (movies)	T	T	T	T
2 (chatgpt) – 5 (twitter)	T	T	T	T
3 (financial) – 4 (movies)	T	T	T	T
3 (financial) – 5 (twitter)	T	T	T	T
4 (movies) – 5 (twitter)	T	T	T	T

T – (true, reject null) there is a statistically significant difference in the means

F – (false, fail to reject null) there is no statistically significant difference in the means

Key takeaways:

* As the number of sentiment factors increase, accuracy decreases
* dataset 5 averages are much lowers than the rest of datasets at (55.12 58.36 55.08 55.09)
* NB for DS4 stands out as the highest results (82.83,83.72 for bow/NB and tf-idf/NB respectively)
*Naive Bayes see to have excellent (>80%) scores in regard to dataset #4 (movies) with binary factors.
*no statistical difference for (bowRF,tfRF,tfNB) with essentially the same worst performance at ≈ 55.1% for dataset 5
* Naive Bayes seem to work better with count Vectorization of BoW.

Limits:

* Study limited due to current hardware limitations:
    ** SVM as ML method dropped
    ** Keybert dropped
* Accuracy only considered, no Confusion Matrix -- no measure for False Positives and False Negative, and subsequently no recall, precision.
* no hyperparameter tuning of ML methods, “vanilla” or “default” ml methods implemented (note: rf param n=10, instead of default 100)
* limited to 1-gram tokens [words], no 2-grams [two word phrases as tokens] due to hardware limitations.
* methods are classifiers (supervised learning), requiring pre-labeled datasets for training.

Future direction:

*With a large uptick in interest in sentiment analysis, demand for “fast”
And efficient methods could be important.

*methods here can be implemented not only on “sentiment analysis”, might easily be implemented in other types of text analysis like spam detection
Or fraud detection (as long as response is a discrete factor response
Appropriate for ML classification.

CONCLUSION:

* avg overall prediction accuracy: 68%

* biggest takeaway of study is Naive bayes on a binary sentiment dataset yielded >80% accuracy.

*despite hardware limitation, I hope this was a good presentation of a basic implementation of NLP and supervised learning.