Sentiment Prediction on Movie Reviews

Lim Jing Rui, Nicole Chong,
Shi ShuangQi, Tee Yue Ning

Web Scraping from IMDb

What did we scrape?

60 English language movies

- Sorted by popularity
- **64,052 observations**

7 variables

- Index column
- Movie title
- Date of review
- Username of individual who posted the review
- Rating (number of stars out of 10)
- Headline of review
- Full textual review

Basic Data Cleaning

- Remove duplicate rows
- Remove observations with NA values under ratings column
- Replace NA entries in `full_review` column with `review_title`
- Change variables to appropriate types
- Remove punctuation, numbers, stopwords, contractions and emojis
- Convert all text to lower case and strip whitespaces

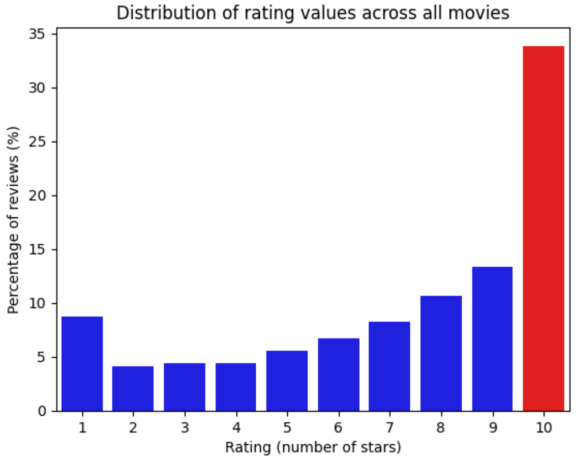
Further Data Preprocessing

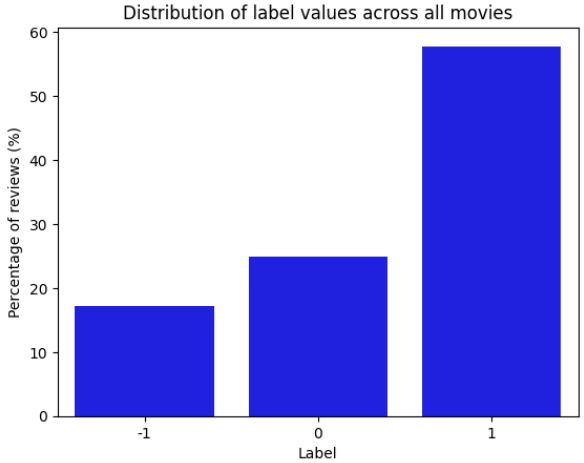
- Remove **accented letters** and **Chinese characters**
- Remove **common** words such as “film” and “movie” → do not convey sentiment and may be deemed as noise by the model
- Kept words that appear more than **twice** in the dataset to prevent overfitting

Implementing Multiclass Classification

Range of stars	Corresponding Class	Corresponding Label
1 - 3	Negative	-1
4 - 7	Neutral	0
8 - 10	Positive	1

Before: left-skewed **VS** **After: slightly more even**





** Data is imbalanced!

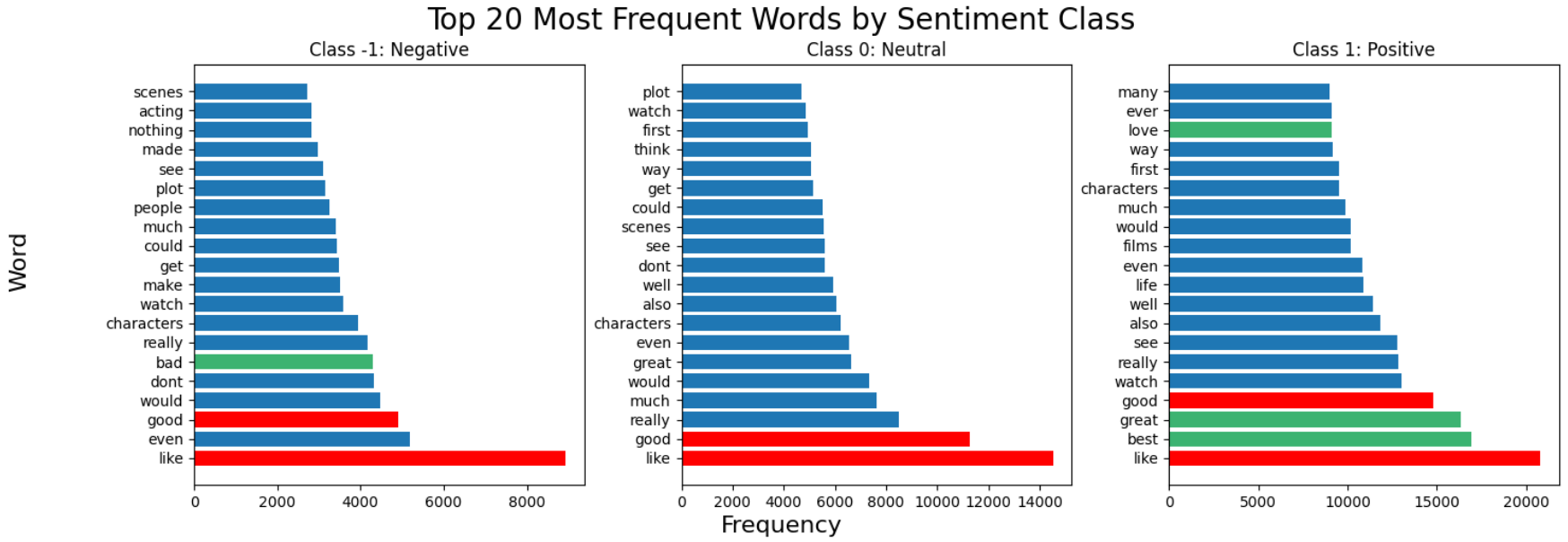
Raw dataset VS Cleaned dataset

	Raw (`imdb`)	**Cleaned (`clean_df`)**
Number of observations	64,052	62,053 (-3%)
Number of columns	6	12

Columns Added:

1. Movie release year
2. Original tokens
3. Final tokens (filter > 2 occurrences)
4. Stemmed tokens
5. Final clean reviews
6. Label (target column)

Most Frequent Words by Sentiment Class



Negative

Neutral

Positive

Do not match corresponding sentiment

Matches corresponding sentiment

Standardised Text Classification Pipeline

CountVectorizer

Convert text data into matrix of token counts

Oversampling

Randomly sample minority class to balance data using `RandomOverSampler`

Label	No. of observations	
	Before	After
-1	2116	7212
0	3083	7212
1	7212	7212

Train-test split

80:20 split for train & test set

Training set size: 49462
Test set size: 12411

TF-IDF transformer

Gives importance to words that are more discriminative

Model

Training and evaluation

Model Selection & Fine-Tuning

	Naive Bayes	Linear SVM
Why?	• Computationally efficient, fast training and predictions • Works well with high-dimensional data (TF-IDF)	• Performs well in high-dimensional spaces (TF-IDF) • Robust to overfitting • Better generalization on unseen data through margin maximisation
Limitations	• Strong assumption of feature independence	• Sensitive to class imbalance
F1 score before VS after fine tuning	73% → 76%	76% → 80%

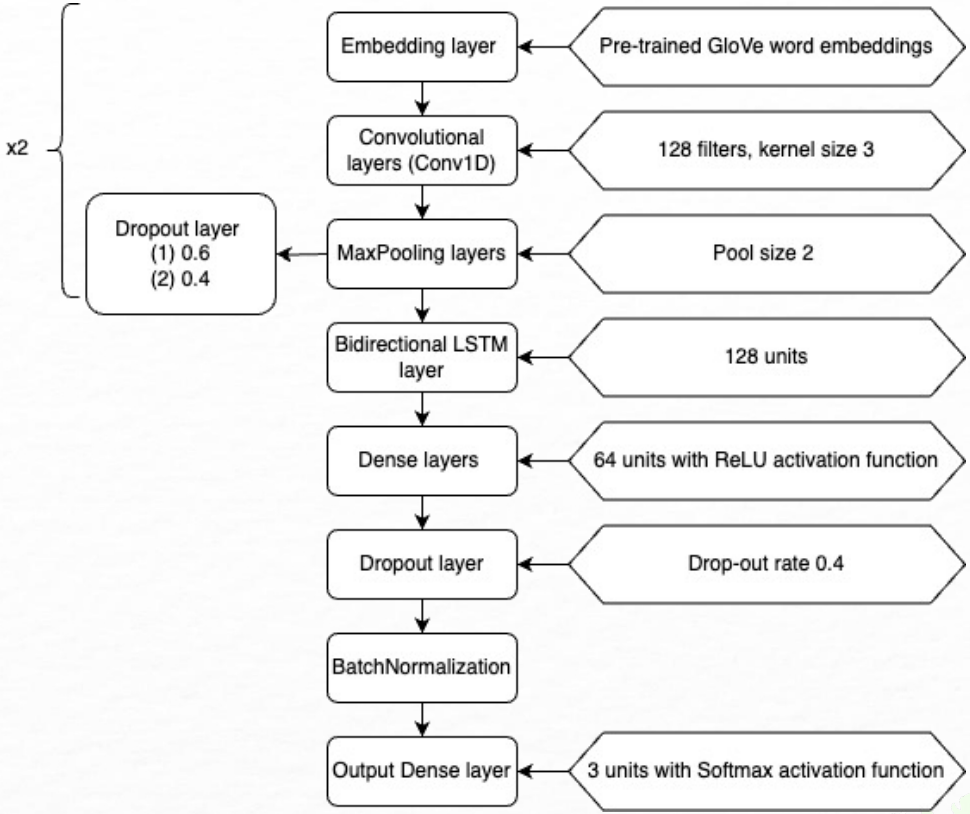
Model Selection & Fine-Tuning

	Logistic Regression	Random Forest
Why?	• Less prone to overfitting • Tends to perform well when there are small number of features relative to number of observations	• Reduces variance (bagging) • Robust to overfitting • Might be able to provide higher predictive accuracy if relationships in data are non-linear
Limitations	• Assumes linear relationship, hence might not be able to capture a complex interaction between data	• Potential for bias predictions towards majority classes
F1 score before VS after fine tuning	77% → 79%	75% → 77%

CNN-BiLSTM Model

CNN and LSTM: powerful architectures commonly used in natural language processing tasks.

Model Architecture



Evaluation Metrics

Micro-Average F1-score

- Accounts for imbalanced data in test set

Macro-Average Precision, Recall, F1-score

- Offers different perspective

SVM & LR > CNN-BiLSTM?

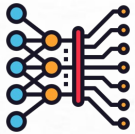
Model	F1 score
CNN-BiLSTM	0.78
Linear SVM (SVM)	0.80
Logistic Regression (LR)	0.79
Random Forest (RF)	0.77
Naïve Bayes (NB)	0.76

Fine-tuned

*Micro-average values!

- Size and quality of data matters in deep learning

1. Approximately 72,000 training points, but ~50% generated from oversampling
2. Oversampling → overfitting when synthetic samples do not represent the true distribution of minority class
3. Model learns noise from these synthetic samples → reduced performance on unseen data
4. Tradeoff in maintaining balanced distribution by oversampling and overfitting



Training accuracy: >0.95

Validation accuracy: ~0.76

- CNN-BiLSTM model more sensitive to imbalanced nature of classes in test set

1. Our model trained on balanced distribution might not perform well when deployed in the real world where distribution of sentiments varies

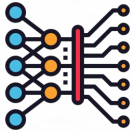
Limitations of BiLSTMs

- Do not perform optimally if data has long-range dependencies

1. Reviews are as long as 250 words after preprocessing → challenge to capture relevant sentiment-bearing words or phrases

- Prone to vanishing gradient issues → difficult for model to learn long-term dependencies

1. Earlier parts of the sequence have little influence on final predictions



CNN-BiLSTM > RF & NB

- NB assumes feature independence

- Structure of RFs

1. Individual trees trained on subsets of samples constrain their ability to comprehend intricate interactions and dependencies

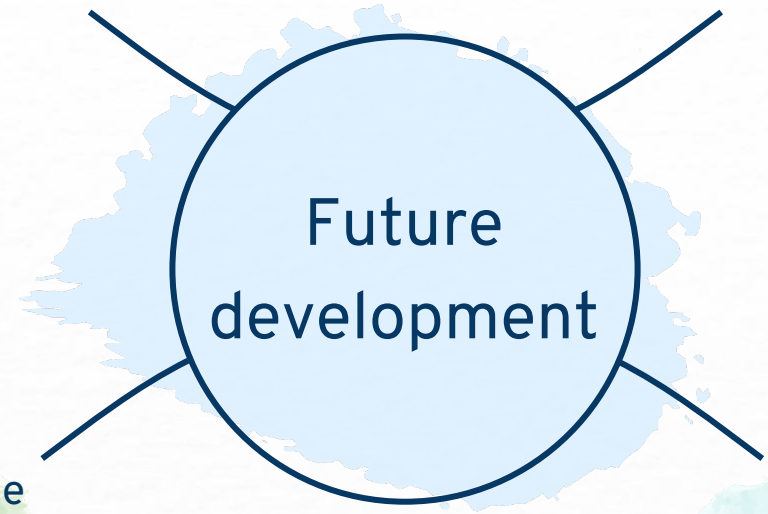
Overall Evaluation of Proposed Solutions

Model	Micro-average	Macro-average		
	F1 score	Precision	Recall	F1 score
CNN-BiLSTM	0.78	0.73	0.73	0.73
Linear SVM (SVM)	0.80	0.75	0.74	0.74
Logistic Regression (LR)	0.79	0.74	0.74	0.75
Random Forest (RF)	0.77	0.72	0.68	0.70
Naïve Bayes (NB)	0.76	0.71	0.72	0.71

Macro-average scores are **LOWER** in general

- Is expected since test set is imbalanced
- Models struggle with correctly predicting minority classes (ie. Neutral, Negative)

Nevertheless, **relative** performance of the models remain consistent across the metrics


Future development
Refining Attention Mechanisms
Start with Bidirectional LSTM
Utilization of Ensemble Methods
Rebalance Data Distribution

THANK YOU