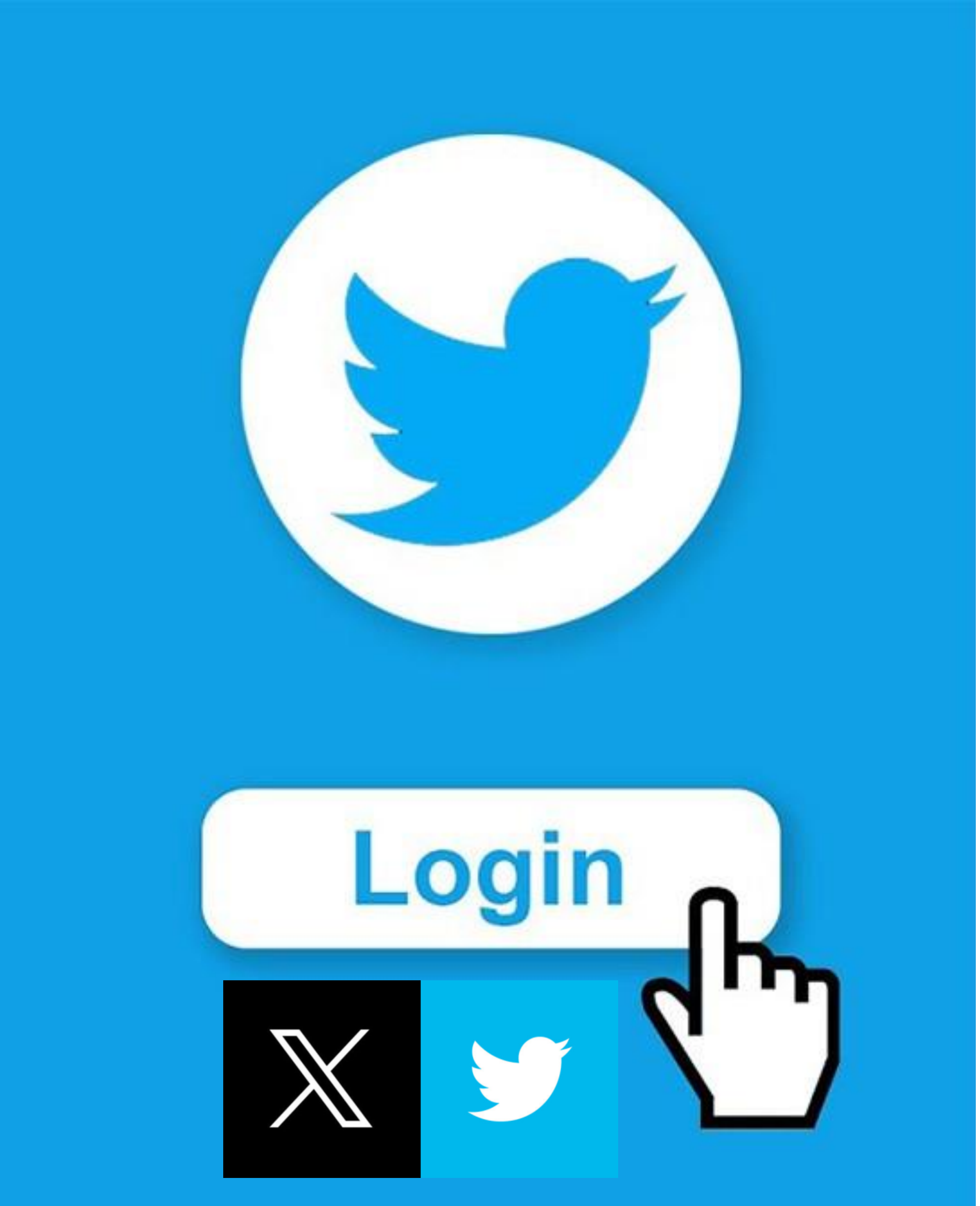
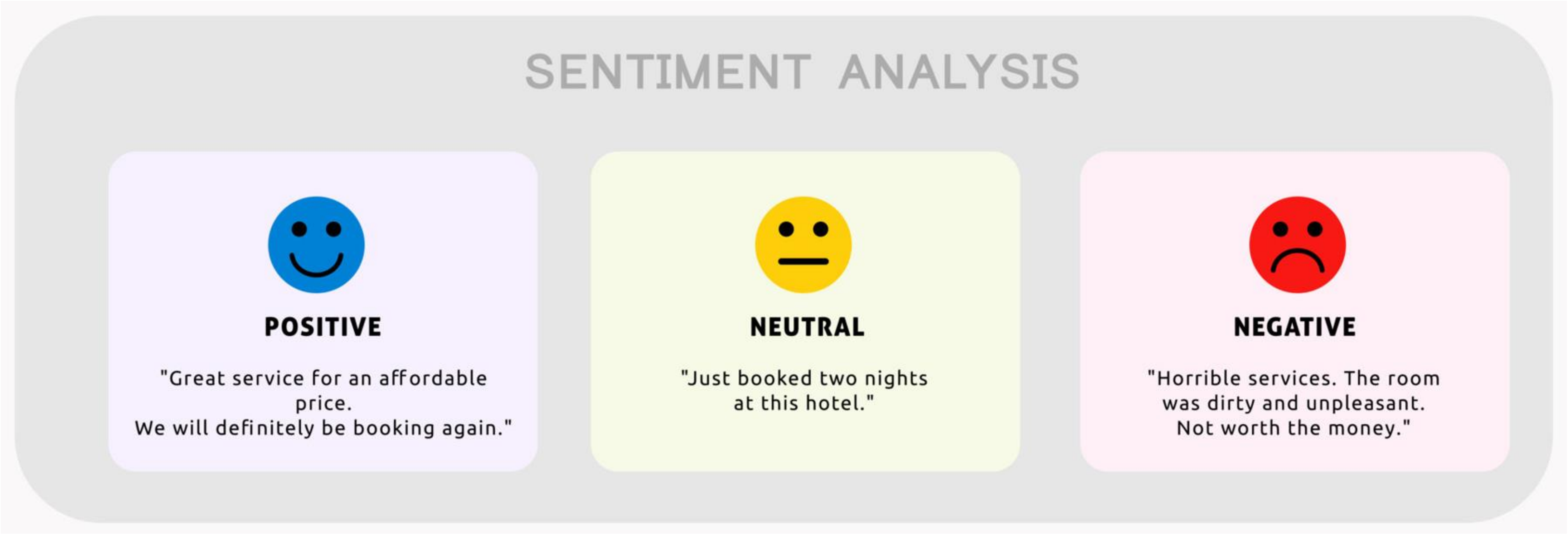


Tweets Sentiment Analysis

Tan Yue Hang
NLP Supervised ML
Mini Project 3

Introduction

- This project aims to build a NLP model that can effectively classify the tweets.
- Gain actionable insights, analyze factors impacting sales, monitor, social media marketing and analyze customer sentiment in real-time.

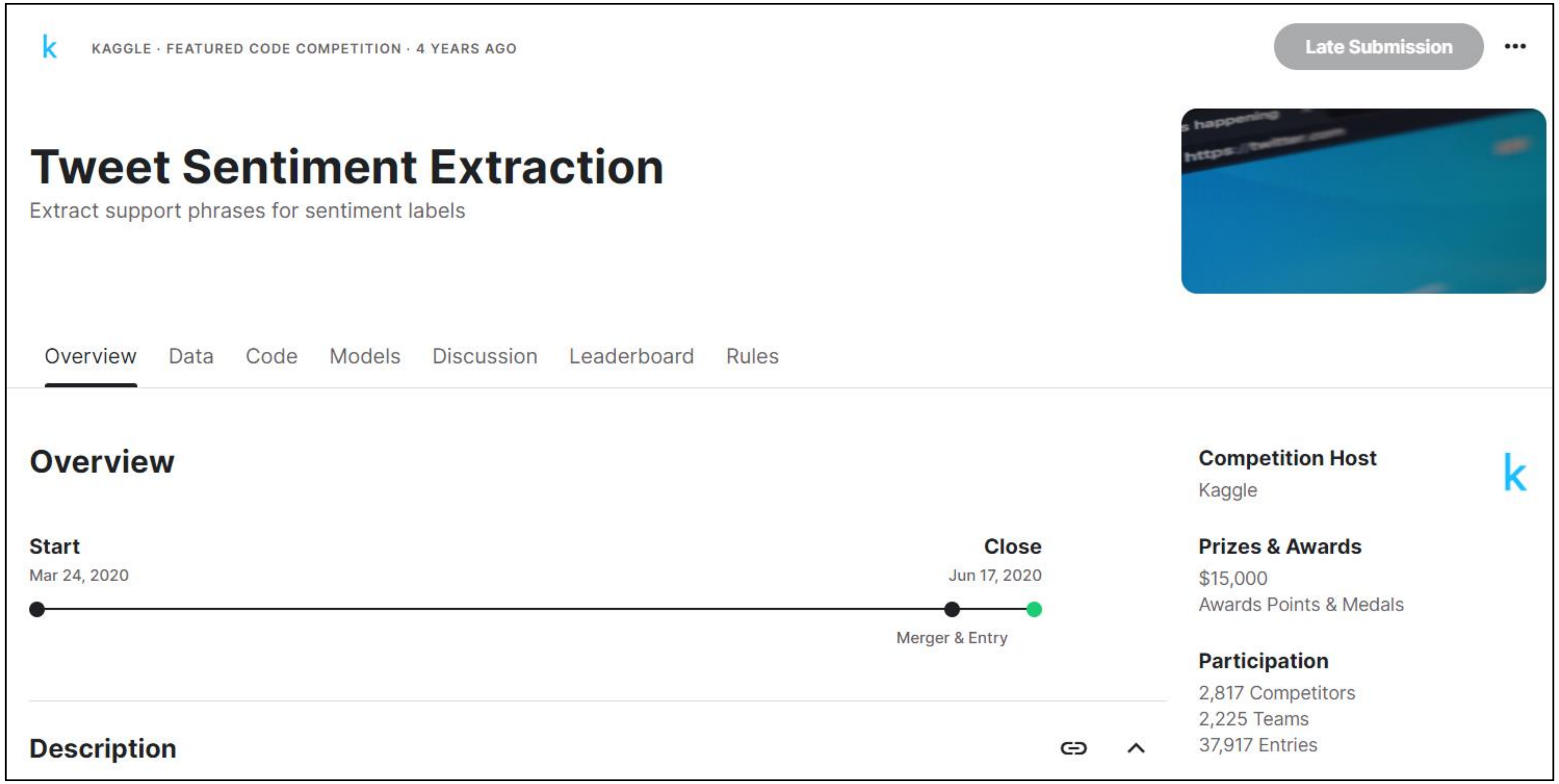


Potential Business Application

- Find out the overall sentiment towards the product based on text reviews.
- Identify the specific features of product that are frequently associated with negative sentiment.
- Identify the trends, patterns or changes of sentiments towards the product over time.
- Competing product sentiments analysis and comparisons.
- Identify potential influencers (users with many followers) who have tweeted positively about the product.
- Utilize the sentiment analysis results as features to build new ML model and predict the future sales trends.
    - E.g. When a product receives more positive sentiments by certain %, it will generally sell more by certain %.

Source of Dataset

- The dataset was acquired from Kaggle, which was also held as a competition in year 2020.
- The dataset consists of two csv files, the training data (27,281 rows) and the test data (3,535 rows). For the simplicity of experimentation purposes, only training data is used in this mini project 3.



Presentation Contents

Section 1: Data Cleaning & EDA

Section 2: ML Modelling

- *Conventional ML Algorithms*
- *Deep Neural Networks*

Section 1:
Data Cleaning & EDA

Dataset Information

- In this Mini Project 3, only train.csv is used which comprises of 27,281 data rows and 4 columns in total before data cleaning for the simplicity of experimentations.
- The target (response) of the analysis would be “sentiment”, which comes with three categories namely “neutral”, “positive”, and “negative”.
- The column “text ID” is rather meaningless, and the column “selected_text” seems to be a cleaned text data. Will drop these two columns and perform separate text data cleaning.
- There is only one row with missing value in “text” column. We will just drop this row and proceed with the rest.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27481 entries, 0 to 27480
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   textID         27481 non-null  object
 1   text           27480 non-null  object
 2   selected_text  27480 non-null  object
 3   sentiment      27481 non-null  object
dtypes: object(4)
memory usage: 858.9+ KB
```

	textID	text	selected_text	sentiment
314	fdb77c3752	NaN	NaN	neutral

Drop this row with missing data.

	textID	text	selected_text	sentiment
0	cb774db0d1	I`d have responded, if I were going	I`d have responded, if I were going	neutral
1	549e992a42	Sooo SAD I will miss you here in San Diego!!!	Sooo SAD	negative
2	088c60f138	my boss is bullying me...	bullying me	negative
3	9642c003ef	what interview! leave me alone	leave me alone	negative
4	358bd9e861	Sons of ****, why couldn`t they put them on t...	Sons of ****,	negative

Original Data

Text Data Cleaning & Features Engineering

Cleaned Data with Features Engineering

	text	sentiment	Lower Case	Punctuations Removed	Spaces Cleaned	Char Counts	No Stop Words	Lemmatized	Tokenized	Word Counts	Word Density	Punctuation Counts
0	I`d have responded, if I were going	neutral	i`d have responded, if i were going	i d have responded if i were going	i d have responded if i were going	27	d responded going	d respond go	[d, respond, go]	8	3.375	2
1	Sooo SAD I will miss you here in San Diego!!!	negative	sooo sad i will miss you here in san diego!!!	sooo sad i will miss you here in san diego	sooo sad i will miss you here in san diego	33	sooo sad miss san diego	sooo sad miss san diego	[sooo, sad, miss, san, diego]	10	3.300	3
2	my boss is bullying me...	negative	my boss is bullying me...	my boss is bullying me	my boss is bullying me	18	boss bullying	boss bully	[boss, bully]	5	3.600	3

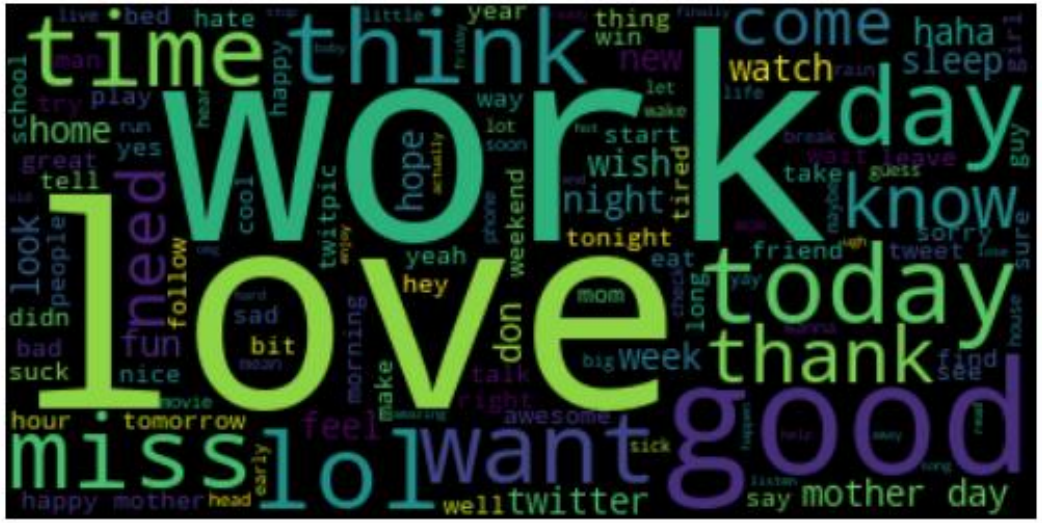
Target

Potential Features

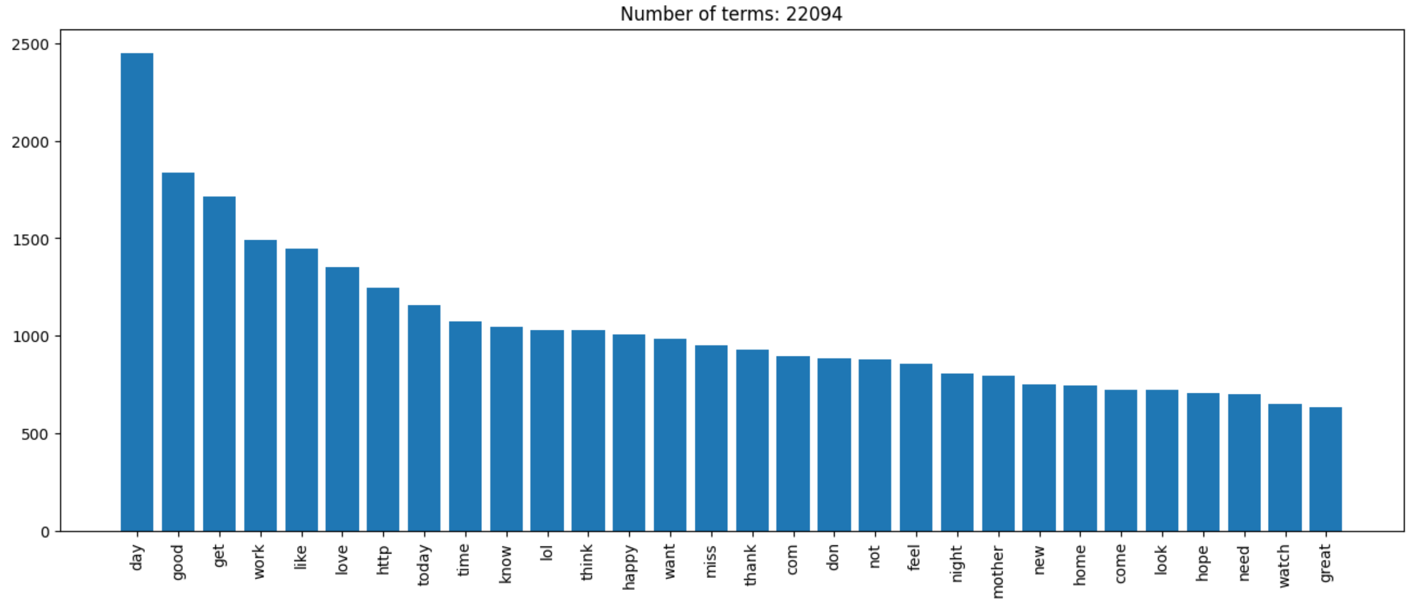
- 8 new features have been created after data cleaning and features engineering, namely:
  - Cleaned text in sentence form
  - Characters counts
  - Sentence without stop words
  - Lemmatized sentence
  - Tokenized sentence stored in list. For the usage of some vectorizers.
  - Word counts
  - Word density
  - Punctuation counts

Features Exploration

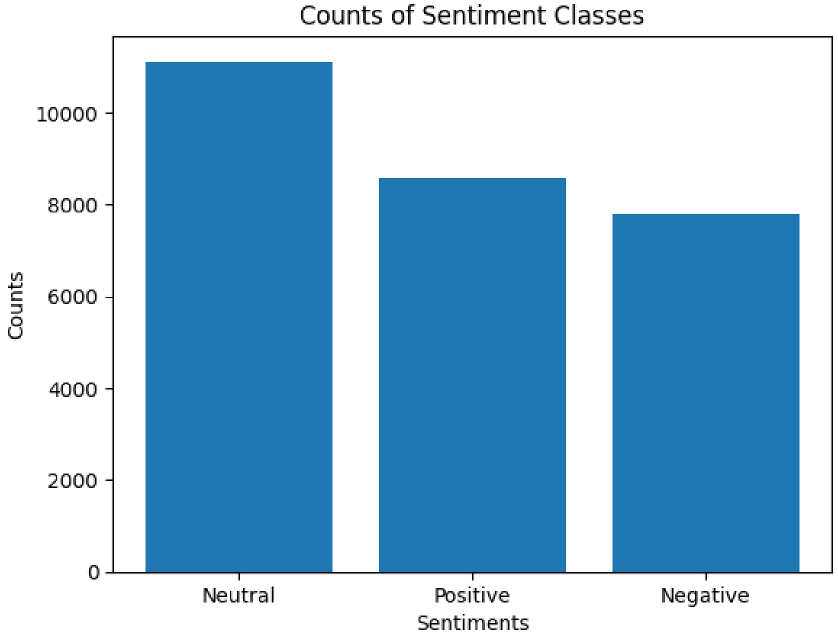
- There is no imbalance in classes observed.
- Some of the highest occurring tokens are shown in the bar plot and illustrated in the word cloud.



WordCloud



Tokens of top 30 occurrences



Sentiment classes

Scaling
(Continuous Data)

- By reviewing the numerical columns, it appears that the scale of these columns are significantly different.
  - Large variation in scale in between features could potentially lead to misleading weights in the model.
- Apply **StandardScaler** to bring all the numerical columns to the same scale.

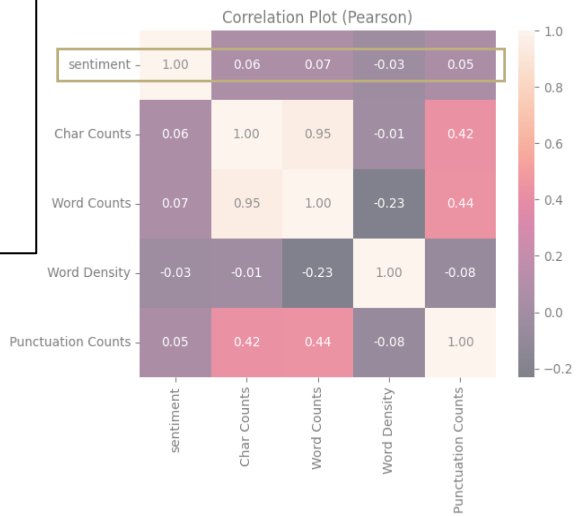
	Char Counts	Word Counts	Word Density	Punctuation Counts
0	-0.918621	-0.756577	-0.709990	-0.465604
1	-0.698465	-0.477051	-0.797283	-0.168549
2	-1.248855	-1.175867	-0.448111	-0.168549
3	-0.992006	-1.175867	1.181357	-0.762659
4	0.108775	0.082002	-0.065685	0.722616

After scaling

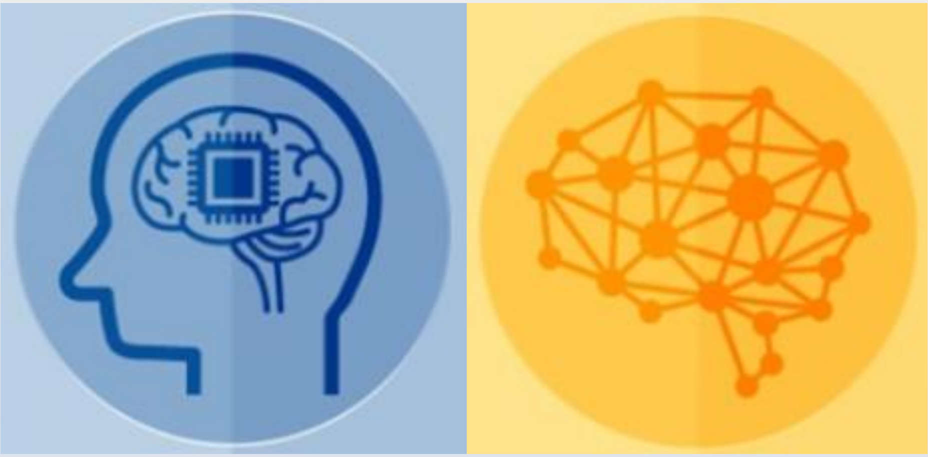
Correlation
(Continuous Data only)

- Apparently all the numerical features do not have significant correlation with the target with Pearson's r.
    - Is Pearson's r a suitable candidate for this use case?
- In this case since the features are continuous data, while the target is categorical data, the more suitable method to determine the correlation is to determine the coefficient of each feature with LogisticRegression model.
    - Train a LogisticRegression model, and return the coef_
- "Char Counts" appears to have certain correlation with the targets.
- While "Word Counts" also seems to have correlation with the targets, "Char Counts" is derived from "Word Counts", so only include one of them to avoid multi-collinearity issue.

Correlation observed



Section 2: ML Modelling



Cartoon source: https://www.futurefundamentals.com/

Data Structure

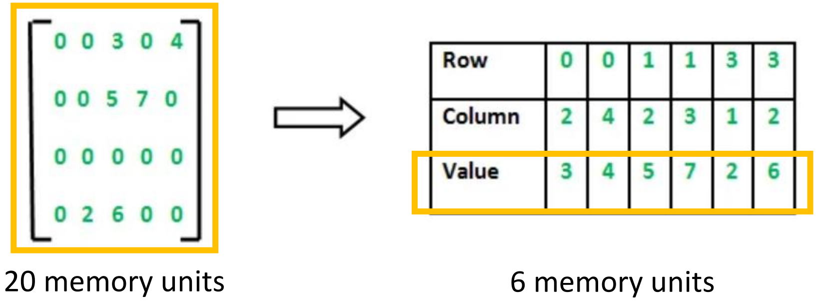
Sparse matrix:

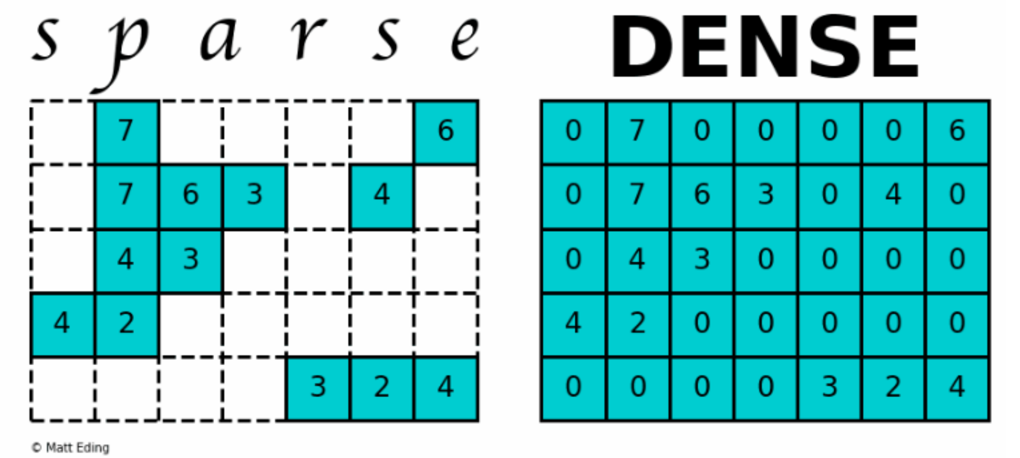
- Contains mostly **zero** values, which are treated as meaningless and not stored / processed at all.
- Common in ML especially in data encoding, e.g. one-hot encoding, some NLP encoding methods and etc.
- Save memory space in storage as machine doesn’t store zero values but by indices (dictionary data structure), good for large datasets.
- Can be processed by conventional ML algorithms directly.
- Memory efficient as zero values are not stored.
- ***Traditionally*** cannot be processed directly by some deep neural networks, [though there are growing interest in using sparse matrix for DNN in recent years due to its memory efficiency].

Dense matrix:

- Zero values are treated as meaningful as non-zero values, and hence processed.
- Suitable for continuous data, where magnitude is important (e.g. image pixel).
- Consume more memory and computational resources.
- Can be processed directly by most deep neural networks traditionally since dense matrix is suitable for direct matrix calculations.

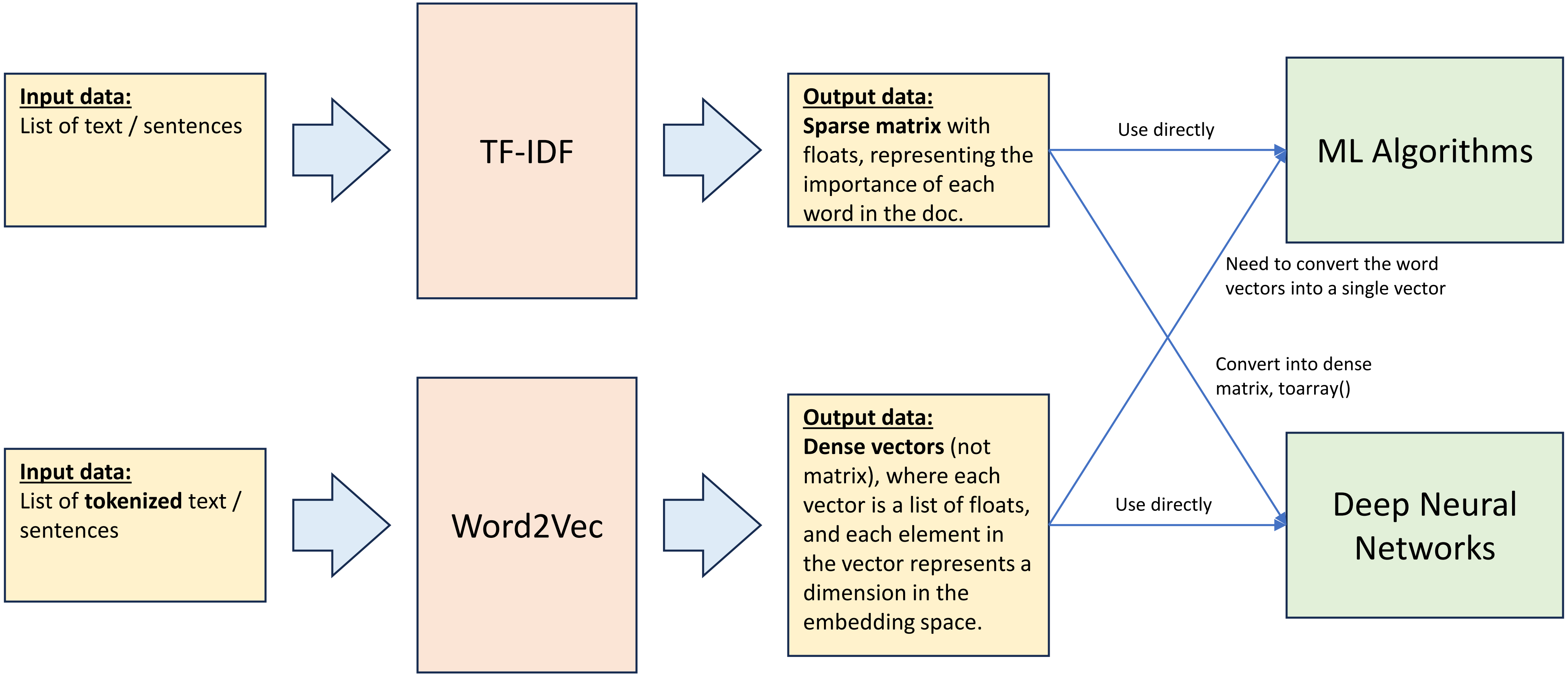
Note: There is no strict rule to differentiate whether a matrix is sparse or dense, it depends on the computational context.





Credit: Source

Data Structure



2.1 Experimentation

Text Data : Lemmatized (w/o Stop Words)
Vectorizer : TF-IDF
Modelling : ML Algorithms
Metric : Recall

Processed Text Data

	text	sentiment	Lower Case	Punctuations Removed	Spaces Cleaned	Char Counts	No Stop Words	Lemmatized	Tokenized	Word Counts	Word Density	Punctuation Counts
0	I`d have responded, if I were going	neutral	i`d have responded, if i were going	i d have responded if i were going	i d have responded if i were going	27	d responded going	d respond go	[d, respond, go]	8	3.375	2
1	Sooo SAD I will miss you here in San Diego!!!	negative	sooo sad i will miss you here in san diego!!!	sooo sad i will miss you here in san diego	sooo sad i will miss you here in san diego	33	sooo sad miss san diego	sooo sad miss san diego	[sooo, sad, miss, san, diego]	10	3.300	3
2	my boss is bullying me...	negative	my boss is bullying me...	my boss is bullying me	my boss is bullying me	18	boss bullying	boss bully	[boss, bully]	5	3.600	3

Results of Different Base Models (Training Data)

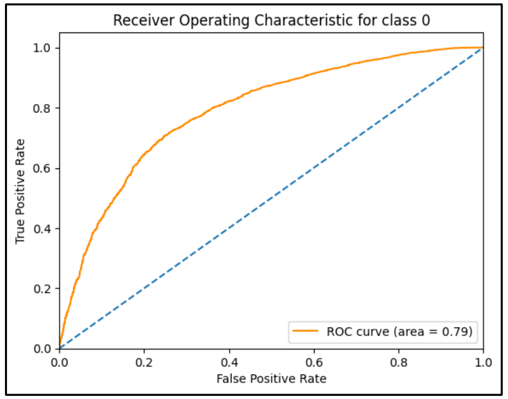
Base Model	Accuracy	Precision	Recall	F1	Elapsed Time
LogisticRegression	0.6756	NaN	NaN	NaN	21.903120
RandomForestClassifier	0.6938	NaN	NaN	NaN	479.752745
XGBClassifier	0.6825	NaN	NaN	NaN	62.041388

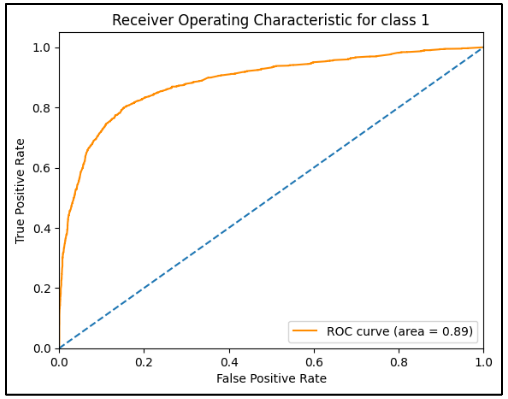
Results with Different Data on the Tuned RandomForestClassifier Model

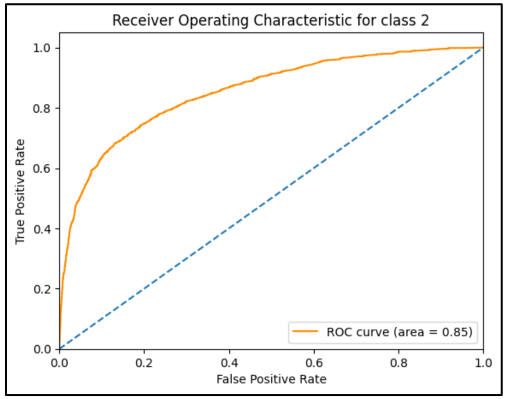
Data	Accuracy	Precision	Recall	F1
Training	0.6916	-	-	-
Validation	0.7082			
Test	0.7029	0.7103	0.6974	0.7014

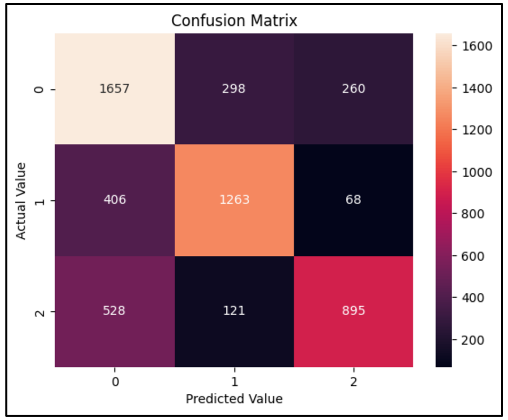
Results with Different Data on the Base Model

Data	Accuracy	Precision	Recall	F1
Training	0.6938	-	-	-
Validation	0.7066	-	-	-
Test	0.6974	0.7074	0.6850	0.6918









- Recall is a more robust metric compared to accuracy, as recall returns the true positive rates over the total actual positives, which is more meaningful to aim for providing accurate sentimental classification as much as possible.
- The goal of this attempt is to simplify and generalize the text as much as possible (reduce noises) in order to achieve better ML model generalization.
- It appears that the RandomForestClassifier works well on this dataset.

2.2 Experimentation

Text Data : Cleaned original text
Vectorizer : TF-IDF
Modelling : ML Algorithms
Metric : Recall

Processed Text Data

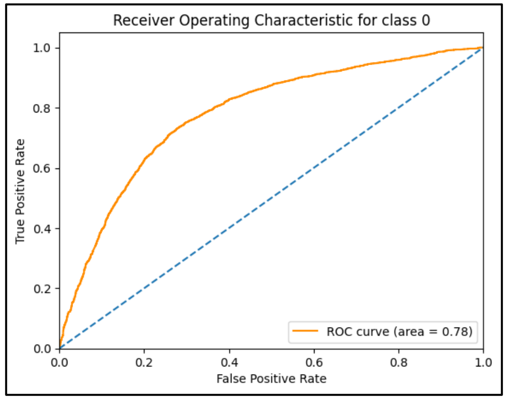
	text	sentiment	Lower Case	Punctuations Removed	Spaces Cleaned	Char Counts	No Stop Words	Lemmatized	Tokenized	Word Counts	Word Density	Punctuation Counts
0	I`d have responded, if I were going	neutral	i`d have responded, if i were going	i d have responded if i were going	i d have responded if i were going	27	d responded going	d respond go	[d, respond, go]	8	3.375	2
1	Sooo SAD I will miss you here in San Diego!!!	negative	sooo sad i will miss you here in san diego!!!	sooo sad i will miss you here in san diego	sooo sad i will miss you here in san diego	33	sooo sad miss san diego	sooo sad miss san diego	[sooo, sad, miss, san, diego]	10	3.300	3
2	my boss is bullying me...	negative	my boss is bullying me...	my boss is bullying me	my boss is bullying me	18	boss bullying	boss bully	[boss, bully]	5	3.600	3

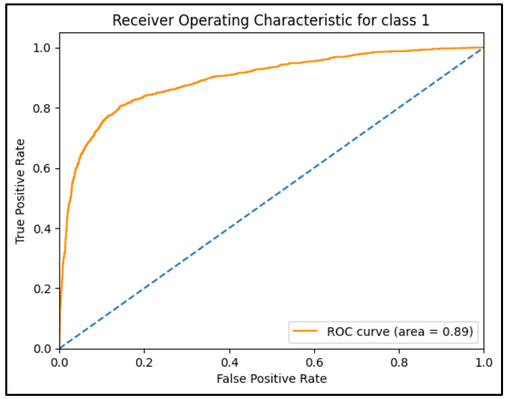
Results of Different Base Models (Training Data)

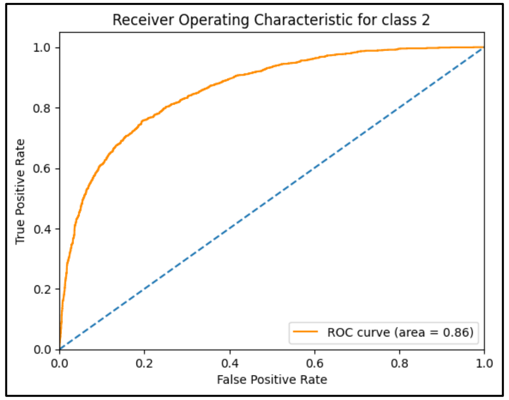
Base Model	Accuracy	Precision	Recall	F1	Elapsed Time
LogisticRegression	0.6757	NaN	NaN	NaN	20.811764
RandomForestClassifier	0.6600	NaN	NaN	NaN	440.694196
XGBClassifier	0.6735	NaN	NaN	NaN	102.323427

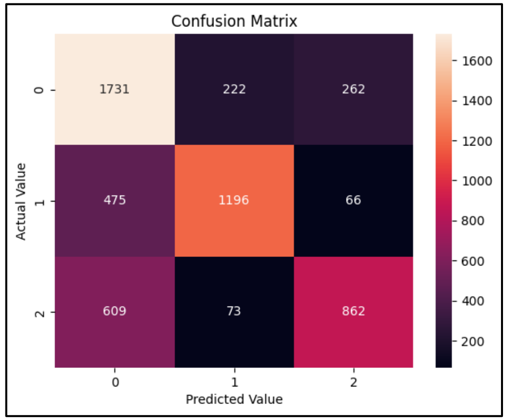
Results with Different Data on the Tuned LogisticRegression Model

Data	Accuracy	Precision	Recall	F1
Training	0.6834	-	-	-
Validation	0.7009			
Test	0.6894	0.7138	0.6761	0.6866









- The goal of this attempt is to try with original text as much as possible which is to aim for minimizing information loss. As a result this attempt only converted the text into lower case as away to standardize, removed punctuations and cleaned up the spaces.
- Continued to use the 3 same ML algorithms so have apple-to-apple comparisons.
- In this case LogisticRegression appears to perform better on the dataset, obtaining the recall score of 67.61%. But this is still not as good as lemmatizing the text and removing the stop words in the first attempt.

2.3 Experimentation

Text Data : Lemmatized (w/o Stop Words) + Numerical Features
Vectorizer : TF-IDF
Modelling : ML Algorithms
Metric : Recall

Processed Text Data

	text	sentiment	Lower Case	Punctuations Removed	Spaces Cleaned	Char Counts	No Stop Words	Lemmatized	Tokenized	Word Counts	Word Density	Punctuation Counts
0	I`d have responded, if I were going	neutral	i`d have responded, if i were going	i d have responded if i were going	i d have responded if i were going	27	d responded going	d respond go	[d, respond, go]	8	3.375	2
1	Sooo SAD I will miss you here in San Diego!!!	negative	sooo sad i will miss you here in san diego!!!	sooo sad i will miss you here in san diego	sooo sad i will miss you here in san diego	33	sooo sad miss san diego	sooo sad miss san diego	[sooo, sad, miss, san, diego]	10	3.300	3
2	my boss is bullying me...	negative	my boss is bullying me...	my boss is bullying me	my boss is bullying me	18	boss bullying	boss bully	[boss, bully]	5	3.600	3

Results of Different Base Models (Training Data)

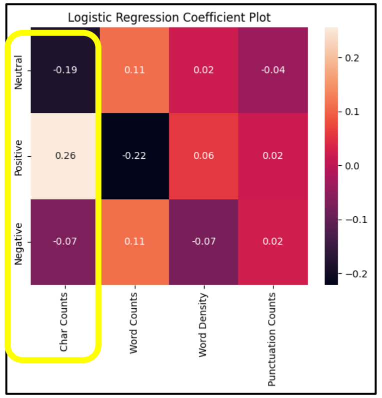
Base Model	Accuracy	Precision	Recall	F1	Elapsed Time
LogisticRegression	0.6810	NaN	NaN	NaN	65.686097
RandomForestClassifier	0.6840	NaN	NaN	NaN	390.348354
XGBClassifier	0.6825	NaN	NaN	NaN	55.793267

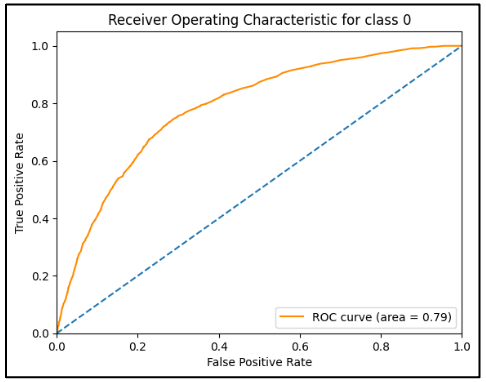
Results of the Tuned RandomForestClassifier Models (Training Data)

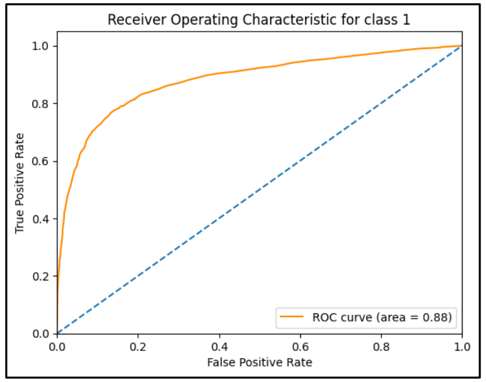
Data	Accuracy	Precision	Recall	F1
Training	0.6867	-	-	-

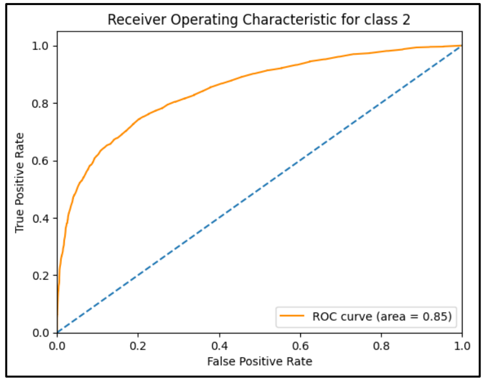
Results with Different Data on the Base RandomForestClassifier Model

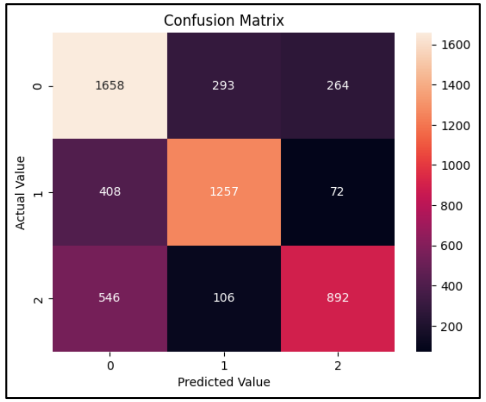
Data	Accuracy	Precision	Recall	F1
Training	0.6840	-	-	-
Validation	0.6937			
Test	0.6927	0.7067	0.6833	0.6905











- Knowing that lemmatized text performs better, the goal of this attempt is to combine with numerical features, to study whether these numerical features could improve the model performance.
- From the coefficients of LogisticRegression, apparently only has certain correlation with the target, so Char Counts is included. Char Counts is derived from Word Counts, so Word Counts shouldn't be included to avoid multi-collinearity issue.
- RandomForestClassifier performs the best on the dataset, obtaining the recall score of 68.33%, but it's still not as good as having lemmatized text alone.
- Numerical features are not needed.

2.4 Experimentation

Text Data : Lemmatized (w/o Stop Words)
Vectorizer : TF-IDF
Modelling : Dense Neural Network
Metric : Recall

Processed Text Data

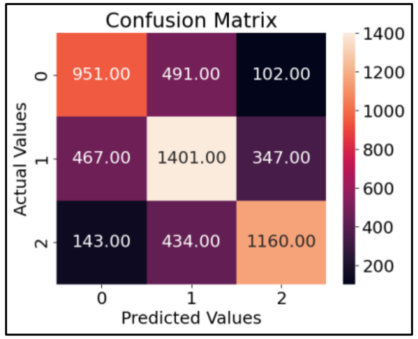
	text	sentiment	Lower Case	Punctuations Removed	Spaces Cleaned	Char Counts	No Stop Words	Lemmatized	Tokenized	Word Counts	Word Density	Punctuation Counts
0	I`d have responded, if I were going	neutral	i`d have responded, if i were going	i d have responded if i were going	i d have responded if i were going	27	d responded going	d respond go	[d, respond, go]	8	3.375	2
1	Sooo SAD I will miss you here in San Diego!!!	negative	sooo sad i will miss you here in san diego!!!	sooo sad i will miss you here in san diego	sooo sad i will miss you here in san diego	33	sooo sad miss san diego	sooo sad miss san diego	[sooo, sad, miss, san, diego]	10	3.300	3
2	my boss is bullying me...	negative	my boss is bullying me...	my boss is bullying me	my boss is bullying me	18	boss bullying	boss bully	[boss, bully]	5	3.600	3

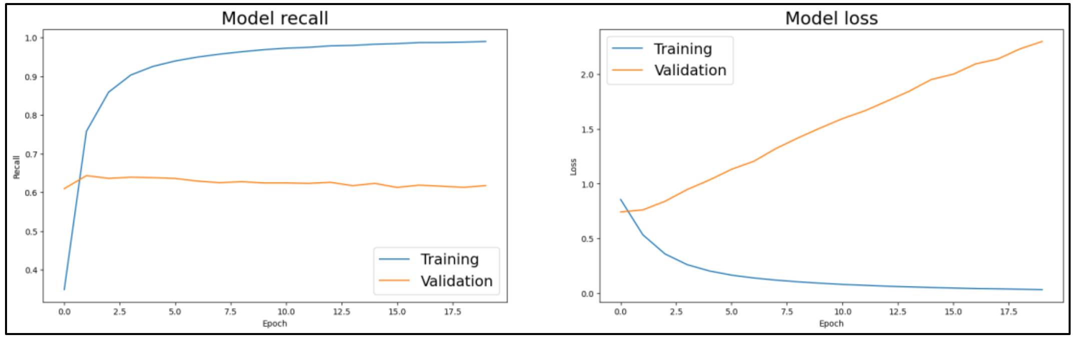
Results with different epochs

Epochs	Data	Accuracy	Precision	Recall	F1
20	Training	-	-	0.9899	-
	Test	0.6683	0.6782	0.6620	0.6671
2	Training	-	-	0.7580	-
	Test	0.6636	0.6719	0.6645	0.6664

DL Model structure

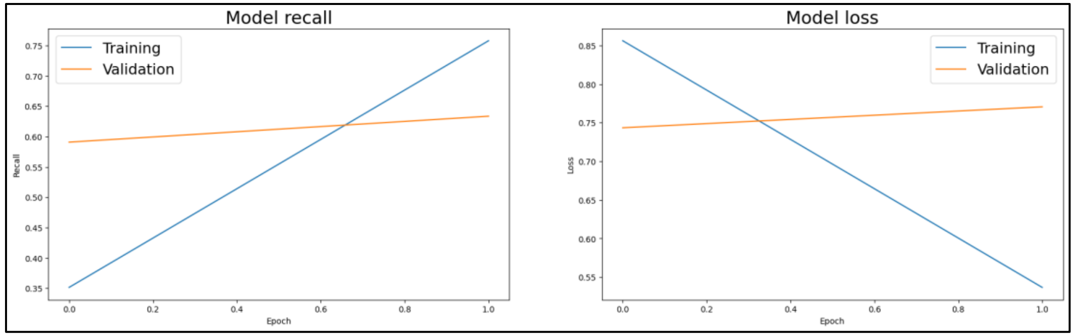
```
# Instantiate and add dense layers to the model
model = Sequential()
model.add(Dense(100, activation="relu", input_dim=22723))
model.add(Dense(3, activation="softmax"))

# Compile the model
model.compile(loss="categorical_crossentropy", optimizer="Adam", metrics=[Recall(name="recall")])
```





With 20 epochs, severe overfitting and poor performance

- At this point we see that Lemmatized text alone is the best. The goal of this attempt is to see whether dense neural network can perform any better than the conventional ML algorithms.
- Started with 20 epochs, and the overfitting becomes too much while the prediction made on validation data is poor.
- It seems that low counts of epoch may do just the same. Repeated the run with only 2 epochs and managed to obtain similar accuracy, precision, recall and F1 scores.
- Simple 2-layer dense neural network is weak for this data set, and its performance is even not as good as RandomForestClassifier and LogisticRegression. Applying early stopping could give similar action.



With 2 epochs, less overfitting but still with poor performance

2.5 Experimentation

Text Data : Lemmatized (w/o Stop Words)
Vectorizer : Word2Vec
Modelling : LSTM
Metric : Recall

Processed Text Data

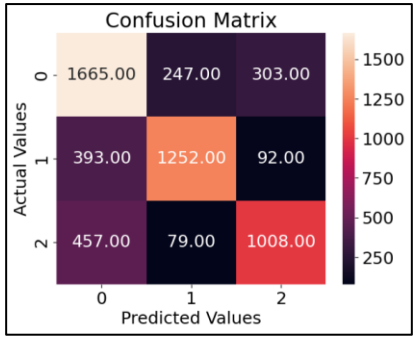
	text	sentiment	Lower Case	Punctuations Removed	Spaces Cleaned	Char Counts	No Stop Words	Lemmatized	Tokenized	Word Counts	Word Density	Punctuation Counts
0	I`d have responded, if I were going	neutral	i`d have responded, if i were going	i d have responded if i were going	i d have responded if i were going	27	d responded going	d respond go	[d, respond, go]	8	3.375	2
1	Sooo SAD I will miss you here in San Diego!!!	negative	sooo sad i will miss you here in san diego!!!	sooo sad i will miss you here in san diego	sooo sad i will miss you here in san diego	33	sooo sad miss san diego	sooo sad miss san diego	[sooo, sad, miss, san, diego]	10	3.300	3
2	my boss is bullying me...	negative	my boss is bullying me...	my boss is bullying me	my boss is bullying me	18	boss bullying	boss bully	[boss, bully]	5	3.600	3

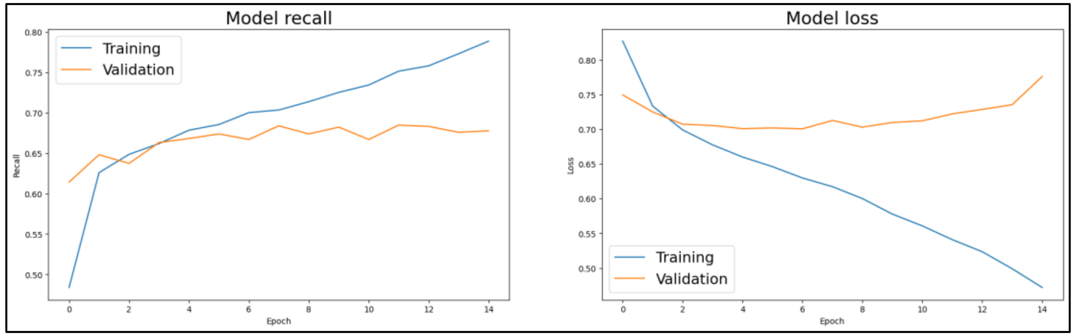
Results with 15 epochs

Data	Accuracy	Precision	Recall	F1
Training	-	-	0.7883	-
Test	0.7142	0.7246	0.7084	0.7145

DL Model structure

```
# Build the model
model = Sequential()

# Add embedding
model.add(Embedding(len(word_index) + 1, word_vectors.vector_size, weights=[embedding_matrix], input_length=100, trainable=False))
model.add(LSTM(128, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(3, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=[Recall(name="recall")])
```





- The goal of this attempt is to try to slightly more advanced vectorizing method, namely **Word2Vec**, and **LSTM** neural network which is a type of **RNN**.
- From the results of 2.4, we learned that overfitting is an issue. Therefore in this trial a dropout rate of 0.2 is added in LSTM.
- By picking the best weights from a total of 15 epochs, the recall that can be obtained is 0.7084, which is also the best results manage to get in this mini project 3.
- While the overfitting issue is not as bad as in 2.4 trial, it’s still bad. More fine tuning (e.g. more dropout layers) is needed to improve the model performance while reducing overfitting issue as much as possible.

Summary of Scores with Various Combinations

Experimentation	Text Data	Vectorizer	Modelling Technique	Best Recall Score
2.1	Lemmatized (w/o Stop Words)	TF-IDF	RandomForestClassifier	0.6974
2.2	Cleaned original text	TF-IDF	LogisticRegression	0.6761
2.3	Lemmatized (w/o Stop Words) + Numerical Features	TF-IDF	RandomForestClassifier	0.6833
2.4	Lemmatized (w/o Stop Words)	TF-IDF	Dense Neural Network	0.6645
2.5	Lemmatized (w/o Stop Words)	Word2Vec	LSTM Neural Network	0.7084

kaggle Leaderboard

Prize Winners

#	△	Team	Members	Score	Entries	Last	Solution
1	▲ 2	Dark of the Moon		0.73615	279	4y	
2	▲ 3	Y. O. & m.y. & hiromu		0.73471	227	4y	
3	▲ 1	Muggles united		0.73332	190	4y	
4	▲ 11	Podpall		0.73208	66	4y	
5	▲ 4	to be twitter masters		0.73152	134	4y	
6	▼ 4	T.M.P		0.73132	357	4y	

Summary of Results & Discussions

- Reduce the complexities of the text down to its “root” form and reducing the unnecessary / not meaningful text as much as possible appears to yield better model performance.
  - Standardize in lower case, remove the punctuations, unnecessary spaces, stop words and etc.
- Dense neural network performs poorly while having overfitting issue too.
- LSTM (one of the RNN), performs well in general, but still at the cost of overfitting.
- Conventional ML algorithms:
  - Pros: Performs pretty well, and there doesn’t seem to be any overfitting issue at all for LogisticRegression, RandomForestClassifier and XGBClassifier. Much shorter training time taken as well compared to DNN.
  - Cons: They reach saturated point quickly and getting difficult to improve after that.
- If the sample size of the text data isn’t large and there no need of very high accuracy, it’s better to go with conventional ML algorithms.

Summary & Final Thoughts

- Less advanced modelling techniques may not be able to catch detail information in the text, hence simpler and more generalized text data may help the performance of the models.
- With conventional ML algorithms, it's good to go with tree-based algorithms (e.g. RandomForestClassifier) and a few other algorithms that do not assume ordinal relationship on label encoded target variable. This way we can use label encoding instead of one-hot encoding in order to avoid high dimensionality issue.
- Deep neural networks tend to overfit the training quite easily. Further tweaking on the model is needed by adding more dropouts, weight regularization and etc.
- Kaggle leaderboard scores are accuracy scores, just for reference.

<u>Future Works:</u>

- Tune the neural networks model further to reduce overfitting while improve the recall scores.
  - Add more LSTM layers and neurons.
  - Apply early stopping to avoid overfitting, coupled with ModelCheckpoint().
  - Add more dropout layers to the model.
- Attempt with state-of-the-art pre-trained LLM like BERT and GPT.
- Hardware: Utilize external GPU to increase the computational speed, as a result able to add more layers and do more experimentations.

End of Presentation