1. Problem Statement E-commerce Review Analysis Team 5 G1

Currently, the default value implemented into the rating system mismatches the ratings and reviews, thereby misleading the customers. The platform is unable to find the key issues purely based on these ratings. Our goal is to predict the class labels of sentiment level. We also intend to extract the reasons behind these reviews, i.e., the root cause identification.

The reviews & ratings are very important components for buyers and for the platform to evaluate the quality of products and their sellers. The better the review and the higher the rating of the product, the more exposure and privileges will it enjoy. The current state does not permit this to happen and instead hampers the business of the sellers and misguides buyers. Also, the inability of the platform to find the root cause is leading to different extents of manual work on a case by case basis, which further leads to long delays and increased unhappiness of customers.

A proper review sentiment analysis will allow the company to not respond to every negative or positive review in the same way and instead provide insights that help craft customised responses to them. These insights include useful information like what customers want, what customers like and dislike about products, what their buying signals and so on. This can really help understand a customer better and at the same time help boost the business by making targeted approaches.

Some challenges faced in this process are the imbalance of the rating categories in the data along with the presence of emojis, abbreviations and spelling errors.

Our pre-processing techniques for the reviews include tokenization to split the text into smaller units and lemmatization to normalize the text. Stop word removal was also used. Also, we had to conduct an emoji removal technique since emojis are a common part of reviews.

2. Evaluation Metrics

2.1 Classification

Our group used the multi-class confusion matrix, which is a performance measure for classification problems. Given that our dataset consists of multi class data, we will be using this to compute a confusion matrix for each class. Even though the confusion matrix contains all the information for the outcome of the classifier, this information is rarely used since it is difficult to compare and discuss (Sklearn, n.d.). Thus, we used the parameters given to compute the weighted F1 for all models.

Weighted F1 score, which is a matrix to measure the predictors performance, as a measure since it conveys the balance between precision and recall by using their harmonic mean. This is a suitable measure for our data set since it has an uneven class distribution. By using the weighted F1 score the majority class is favoured since, each class is calculated independently, but is added together using a weight that depends on the number of true labels of each class (Shmueli, 2020).

2.2 Clustering

For clustering, our group chose to use **Silhouette score**, which measures how similar an object is to its own cluster compared to other clusters. This measure has a range between -1 and 1 where a silhouette coefficient nearer to 1 indicates that the sample is far away from the neighbouring cluster while a value nearer to -1 indicates that those samples might have been assigned to the wrong cluster, and a value of - indicates that the sample is on or very close to the decision boundary between two neighbouring clusters. In addition, the silhouette score also helps to provide a common metric for evaluating the different clustering models (Sklearn, n.d.).

3. ML Techniques to solve the problem

S/N	Techniques
1	TF-IDF and Word Embedding (Word2Vec)
2	Clustering Methods (DBSCAN,OPTICS,K-means,Birch,Spectral Clustering, Affinity propagation.,Gaussian mixtures and Mean-shift)
3	Statistical Models: Naive Bayes, Logistic Regression, SVM
4	Neural Network Classification (LSTM, Bi-LSTM)
5	Ensemble Classifiers (Light GBM, RandomForest, ExtraTrees, Voting)

4. Comparison of these techniques

4.1 TF-IDF

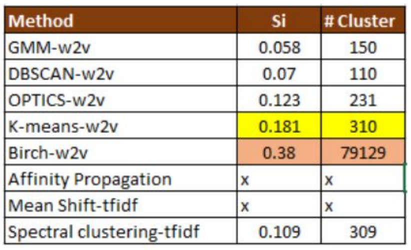
TF-IDF vectoriser - parameter used: min_df = 0.001, max_df = 0.5, ngram_range = (1,5), max_features = 3000
We have around 200k reviews in our dataset and no other features. Hence, in order to get as many important features as possible we use a lower min_df and a high max_df to retrieve the important words as our features. An ngram_range of 1 to 5 is also to include features with multiple words which are near each other in the review. Max feature is being used to select the most important words from the TF-IDF word vector to prevent overfitting since the number of features can be very large.

4.2 Word2Vec

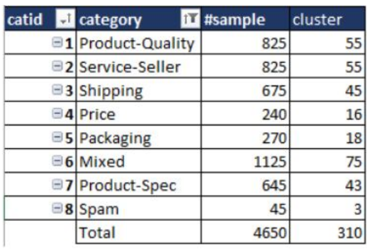
Word2Vec vectoriser - parameters used: size=300, window=7, min_count=10, worker=8
As an alternative vectoriser to TF-IDF, we used word2vec to find important words to act as our features. Using a size of 300 for the vector is concluded by testing higher and lower sizes when using the word2vec model.

4.3 Clustering Models

As the data given without the true labels of review reason categories, the team then had to conduct the label generating process with heavy involvement of the clustering models. The clustering model guided the team to discover the hidden pattern among the text samples. Every method has been tested on both TF-IDF and word2vec metrics and then find the averaged vectors and distance from the sentence for clustering. The final method selected is Kmeans clustering with word2vec data, the results given **310 clusters and being manually combined into 8 categories for next stage classification of review reasons**.
K-means - to compute Euclidean distance on n-dimensional Euclidean space, cluster the samples based on the means distance computed which ensure the more similarity between points within the group and vice versa outside of the groups. The Number of clusters K has been used for tuning, the best result reached Silhouette score 0.181 with 310 clusters.
DBSCAN,OPTICS - The density based clustering methods which group the points based on the neighbours of each point. Tuning parameter is the number of minimum sample per group, the best result would be 0.07 for DBSCAN and 0.123 for OPTICS
Brich hierarchical clustering is performed using the default parameters. It came with the best Silhouette score of 0.38 but it has given us 79 thousand of clusters making it not feasible for manual labelling.



Method	Si	# Cluster
GMM-w2v	0.058	150
DBSCAN-w2v	0.07	110
OPTICS-w2v	0.123	231
K-means-w2v	0.181	310
Birch-w2v	0.38	79129
Affinity Propagation	x	x
Mean Shift-tfidf	x	x
Spectral clustering-tfidf	0.109	309



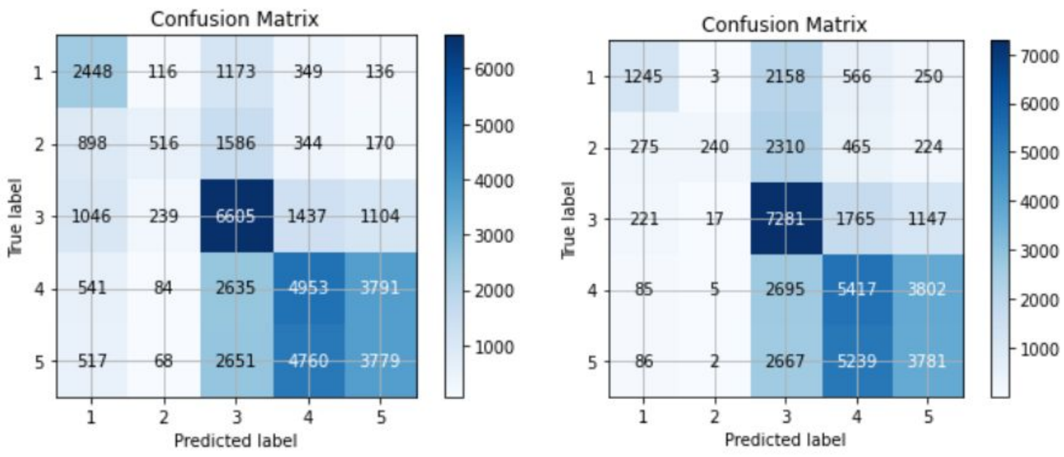
catid	category	#sample	cluster
1	Product-Quality	825	55
2	Service-Seller	825	55
3	Shipping	675	45
4	Price	240	16
5	Packaging	270	18
6	Mixed	1125	75
7	Product-Spec	645	43
8	Spam	45	3
	Total	4650	310

4.4 Statistical Model

For all the statistical models listed, we have trained it with both TF-IDF vectoriser but due to time constraint not all models can be trained using Word2Vec embedding.

4.4.1 Naive Bayes

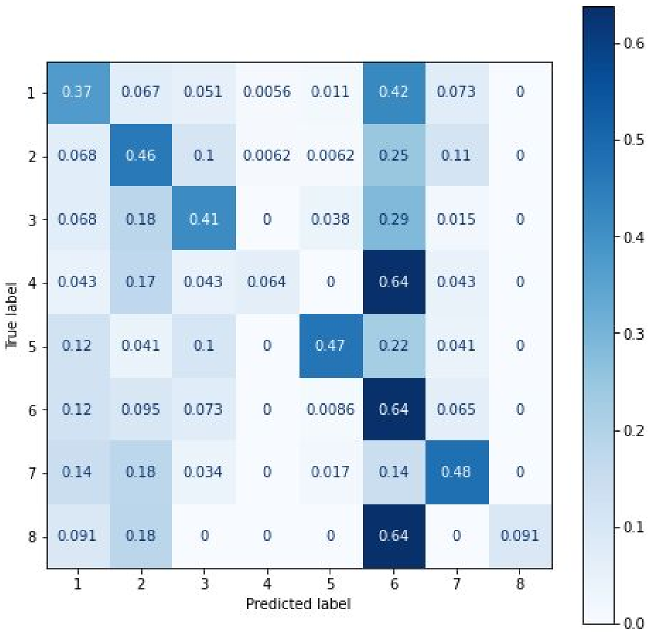
This is our benchmark model where we determine if another model is suitable for our dataset. However, we still perform parameter tuning of alpha to determine which alpha value better suits our benchmark model. We train the Naive bayes across 5 alpha value 0.001, 0.01, 0.01, 1, 10. We choose our benchmark by comparing its train accuracy, test accuracy and log loss. Although alpha = 10 provides a better train and test accuracy, we decided to use alpha = 1 as it has the lowest log loss of 1.21. As you can compare with the confusion matrix below, although alpha = 10 did a good job in its overall accuracy, its prediction for class 1 is a lot worse compared to alpha = 1. In order for us to identify bad reviews we need to be able to classify both the good and the bad reviews but in alpha = 10 its ability to distinguish between a bad (1 and 2) and neutral (3) is a lot worse compared to alpha = 1. Therefore despite the accuracy, we would continue to use alpha = 1 as our benchmark model which has a weighted F1 score of 43.02%.

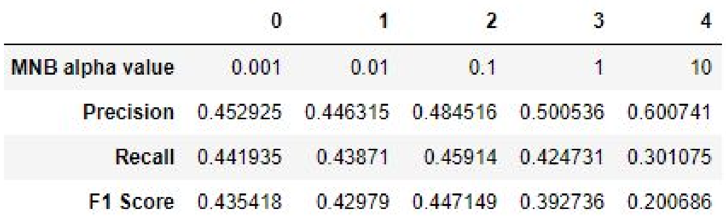


Alpha = 1 on the left, alpha = 10 on the right

As for Naive bayes for word2vec it performs very badly across the various values of alpha, hence we would exclude it from our analysis.

Review reason analysis:
For review reason analysis, the team did apply the similar methods for the making the benchmark model. By tuning the value of the alpha from 0.001 to 10, the naive bayes model performs on the TF-IDF matrix , the best in terms of the weight F1 score while taking into account the imbalanced categories at 44.71%. A weakness of prediction occurred between class 6 and the all others where the 6 is the categories of mixed reasons.





	0	1	2	3	4
MNB alpha value	0.001	0.01	0.1	1	10
Precision	0.452925	0.446315	0.484516	0.500536	0.600741
Recall	0.441935	0.43871	0.45914	0.424731	0.301075
F1 Score	0.435418	0.42979	0.447149	0.392736	0.200686

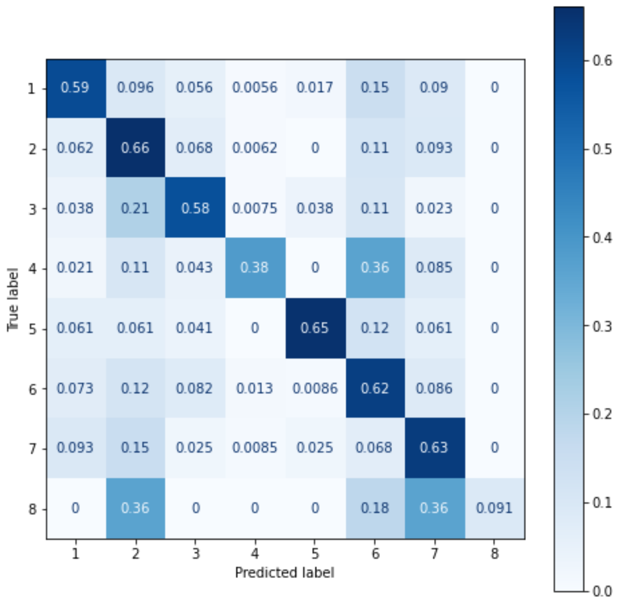
4.4.2 Logistic Regression

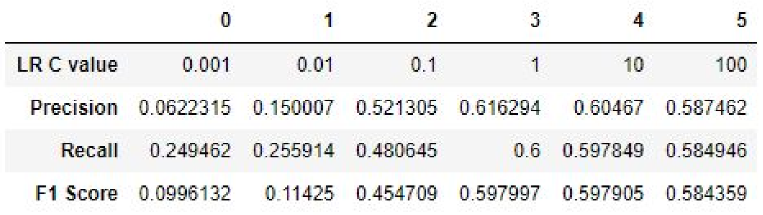
Sentiment analysis:

Logistic Regression performs slightly better than naive bayes. We also perform parameter tuning on the across 4 C values 0.01, 0.1, 1, 10 on the logistic regression for both TF-IDF vectors and word2vec. Out of the 4 models, we have chosen to use the model with C value = 1 for the model that is trained with TF-IDF vectors as it has the lowest log loss and the highest test accuracy of 46.37% and weighted F1 score of 45.79%. As for Word2Vec, the performance is not as good and the time taken to train the model is a lot longer compared to TF-IDF with a weighted F1 score of 44.83%.

Review reason analysis:

For review reason analysis, the Logistic regression also has been considered by the team, as usual the C values of the inverse regularization strength have been tuned for this model to predict the review reasons. Among the all variations here, the C value at 1 has given the best result of Precision, recall and F1 score at approx. of 59.79%.





	0	1	2	3	4	5
LR C value	0.001	0.01	0.1	1	10	100
Precision	0.0622315	0.150007	0.521305	0.616294	0.60467	0.587462
Recall	0.249462	0.255914	0.480645	0.6	0.597849	0.584946
F1 Score	0.0996132	0.11425	0.454709	0.597997	0.597905	0.584359

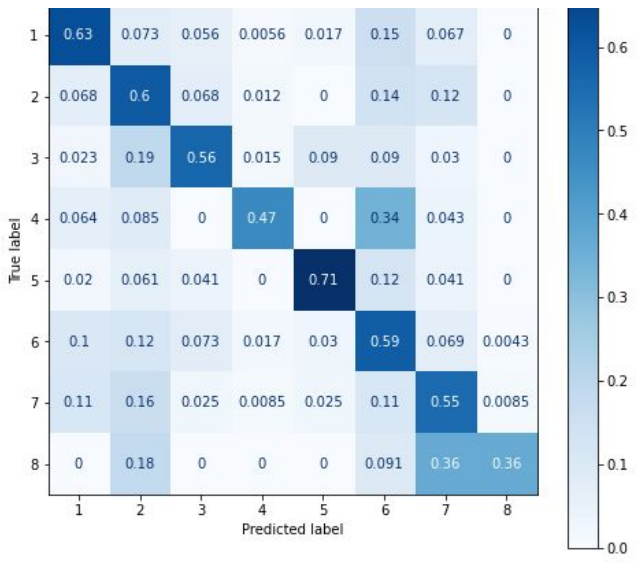
4.4.3 SVM

Sentiment analysis:

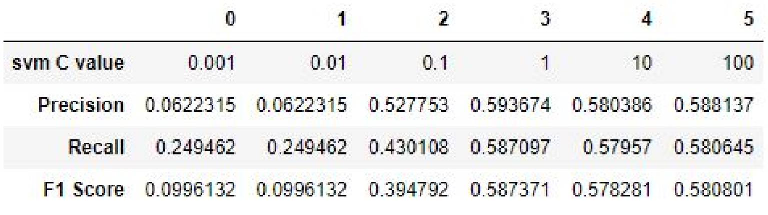
The parameters we used for our SVC model is using a linear kernel, while the rest of the parameters are the default value. Due to time and computational strength, we did not optimize our SVC model as our dataset is considerably large to be used for SVM. Using TF-IDF, we have gotten a weighted F1 score of 45.52%.

Review reason analysis:

As the review reason analysis used the lesser samples that derived from the clustering results, the team then decided to tune the C values for SVM with linear kernel. The best F1 score achieved at 58.73% with a C value of 1 which is slightly lesser than logistic regression. However, this model is doing better to distinguish the category 8 spam and 7.



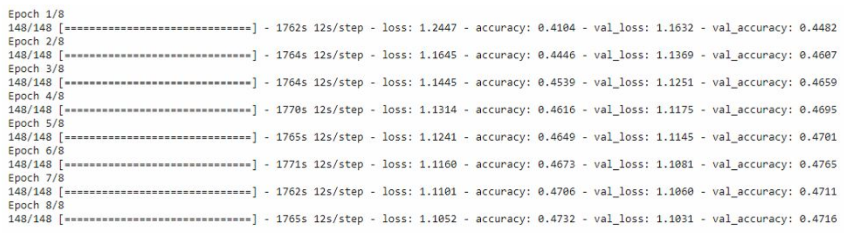
True label \ Predicted label	1	2	3	4	5	6	7	8
1	0.63	0.073	0.056	0.0056	0.017	0.15	0.067	0
2	0.068	0.6	0.068	0.012	0	0.14	0.12	0
3	0.023	0.19	0.56	0.015	0.09	0.09	0.03	0
4	0.064	0.085	0	0.47	0	0.34	0.043	0
5	0.02	0.061	0.041	0	0.71	0.12	0.041	0
6	0.1	0.12	0.073	0.017	0.03	0.59	0.069	0.0043
7	0.11	0.16	0.025	0.0085	0.025	0.11	0.55	0.0085
8	0	0.18	0	0	0	0.091	0.36	0.36



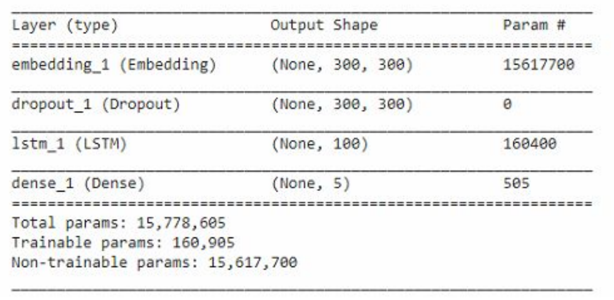
	0	1	2	3	4	5
svm C value	0.001	0.01	0.1	1	10	100
Precision	0.0622315	0.0622315	0.527753	0.593674	0.580386	0.588137
Recall	0.249462	0.249462	0.430108	0.587097	0.57957	0.580645
F1 Score	0.0996132	0.0996132	0.394792	0.587371	0.578281	0.580801

4.5 Neural Network Classification - LSTM

Using Keras library, we built a LSTM neural network that consists of an embedding layer, which is built using Word2Vec, Dropout with 0.5, the LSTM layer with 100 neurons, dropout of 0.2, as well as a Dense layer for the output with an activation function of "SoftMax" and ran it for 8 epochs. With LSTM, we achieve an accuracy of 47.70%, but upon comparing with a weighted F1 score, it is lower at 44.75%.



```
Epoch 1/8
148/148 [==============================] - 1762s 12s/step - loss: 1.2447 - accuracy: 0.4104 - val_loss: 1.1632 - val_accuracy: 0.4482
Epoch 2/8
148/148 [==============================] - 1764s 12s/step - loss: 1.1645 - accuracy: 0.4446 - val_loss: 1.1369 - val_accuracy: 0.4607
Epoch 3/8
148/148 [==============================] - 1764s 12s/step - loss: 1.1445 - accuracy: 0.4539 - val_loss: 1.1251 - val_accuracy: 0.4659
Epoch 4/8
148/148 [==============================] - 1770s 12s/step - loss: 1.1314 - accuracy: 0.4616 - val_loss: 1.1175 - val_accuracy: 0.4695
Epoch 5/8
148/148 [==============================] - 1765s 12s/step - loss: 1.1241 - accuracy: 0.4649 - val_loss: 1.1145 - val_accuracy: 0.4701
Epoch 6/8
148/148 [==============================] - 1771s 12s/step - loss: 1.1160 - accuracy: 0.4673 - val_loss: 1.1081 - val_accuracy: 0.4765
Epoch 7/8
148/148 [==============================] - 1762s 12s/step - loss: 1.1101 - accuracy: 0.4706 - val_loss: 1.1060 - val_accuracy: 0.4711
Epoch 8/8
148/148 [==============================] - 1765s 12s/step - loss: 1.1052 - accuracy: 0.4732 - val_loss: 1.1031 - val_accuracy: 0.4716
```



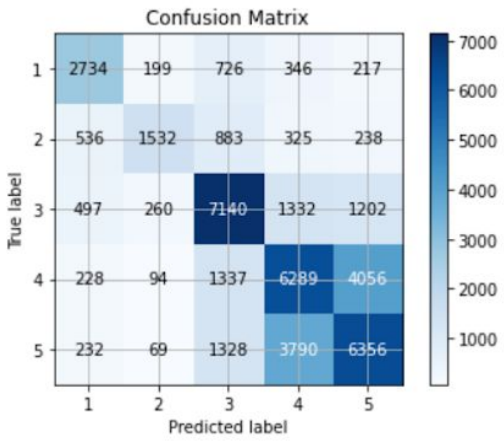
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 300, 300)          15617700
_________________________________________________________________
dropout_1 (Dropout)          (None, 300, 300)          0
_________________________________________________________________
lstm_1 (LSTM)                (None, 100)               160400
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 505
=================================================================
Total params: 15,778,605
Trainable params: 160,905
Non-trainable params: 15,617,700
_________________________________________________________________
```

4.6 Ensemble Learning

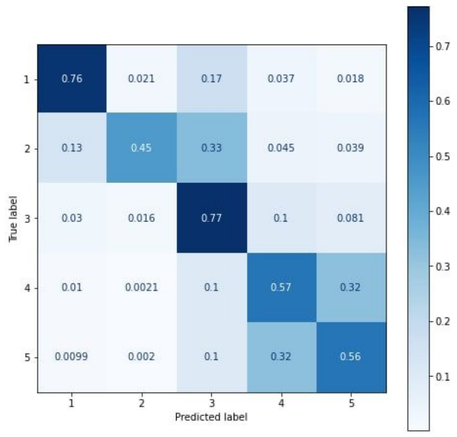
Used random forest and extra trees from the sklearn library as the ensemble learning models which to be explored in the final stage of the project development. The ensemble models created multiple sub-models for prediction and combined the results through voting. Although the both random forest and extra tree are tree based ensemble techniques that shared a lot of similarities, there are 2 main differences. The first difference is the sample input, the Random forest used subsamples with replacement while the extra tree used the entire original sample set. The second difference is extra tree construction trees with randomly node splitting logics while the random forest used an optimal split algorithm. Based on their characteristics, the extra tree classifier has a similar performance of random forest at a faster speed.

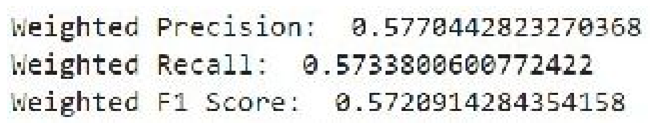
Sentiment analysis:
For Random Forest Classifier, a weakness of prediction occurred between class 4 and class 5, where class 4 labels were predicted as class 5 labels and vice versa. The F1 score achieved at 57.21%, whereas Extra Trees Classifier achieved a F1 score of 62.22%. Overall, the ensemble models did better prediction across all the categories from the sample given with no considering the execution time used. As such the team decided that extra tree classifier shall be chosen as the most suitable classifier for our sentiment analysis.

Random Forest:

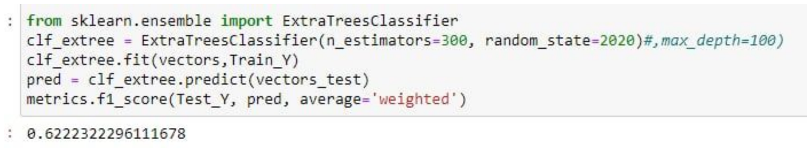


Extra Trees:





```
Weighted Precision:  0.5770442823270368
Weighted Recall:  0.5733800600772422
Weighted F1 Score:  0.5720914284354158
```

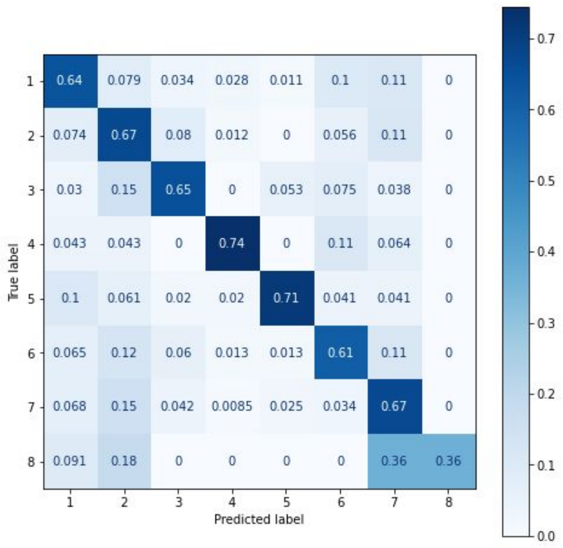


```
from sklearn.ensemble import ExtraTreesClassifier
clf_extree = ExtraTreesClassifier(n_estimators=300, random_state=2020)#,max_depth=100)
clf_extree.fit(vectors,Train_Y)
pred = clf_extree.predict(vectors_test)
metrics.f1_score(Test_Y, pred, average='weighted')
```

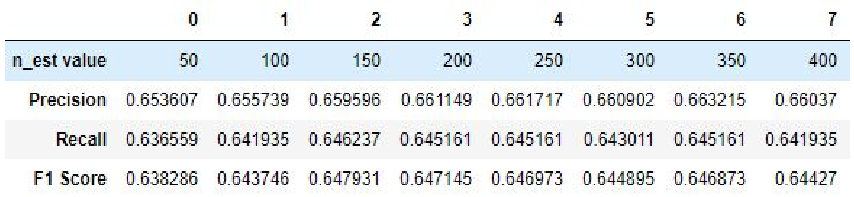
```
0.6222322296111678
```

Review reason analysis:

The parameter used for fine tuning is the number of estimators, both extra tree and random forest models have been fed with n of from 50 to 400 with a step value of 50. The result showed that for extratree with 200 meta estimators and random forest with 400 estimators provided the best prediction result at 64.71% and 65.63% respectively. In addition, the ensemble models did better prediction across all the categories from the sample given with no considering the execution time used. As such the team decided that a random forest classifier shall be chosen as the most suitable classifier for our review reason analysis.

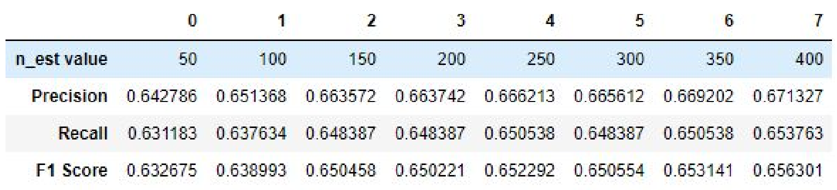


ExtraTree:



	0	1	2	3	4	5	6	7
n_est value	50	100	150	200	250	300	350	400
Precision	0.653607	0.655739	0.659596	0.661149	0.661717	0.660902	0.663215	0.66037
Recall	0.636559	0.641935	0.646237	0.645161	0.645161	0.643011	0.645161	0.641935
F1 Score	0.638286	0.643746	0.647931	0.647145	0.646973	0.644895	0.646873	0.64427

Random Forest:



	0	1	2	3	4	5	6	7
n_est value	50	100	150	200	250	300	350	400
Precision	0.642786	0.651368	0.663572	0.663742	0.666213	0.665612	0.669202	0.671327
Recall	0.631183	0.637634	0.648387	0.648387	0.650538	0.648387	0.650538	0.653763
F1 Score	0.632675	0.638993	0.650458	0.650221	0.652292	0.650554	0.653141	0.656301

4.6.1 LGBM

Sentiment Analysis:

Compared to the common statistical methods, a gradient boosting method provides slightly better performance where the LGBM trained with TF-IDF Vector obtain a weighted F1 score of 49.81%. Most statistical methods perform badly when we are using Word2Vec. However, when we are using LGBM with Word2Vec its performance surpasses TF-IDF vectors with a weighted F1 score of 56.79%. Both LGBM uses the same model parameters which may be the results of too little features in the TF-IDF vectors. We tried a silo test where we did not limit the max features of the TF-IDF vectoriser and retrain the LGBM model and the result is only slightly better.

Although LGBM took a longer time to train, we managed to tune the parameter a few times. They are feature fraction, bagging fraction, bagging frequency and learning rate. The first 3 parameters are used to prevent overfitting while learning rate is used to make sure that the model learns “sufficiently” before converging or underfit. Feature fraction is to limit the number of features used for each iteration. Bagging fraction refers to how much data is being used in each iteration and bagging frequency determines how often the bag of data used to train the model changes.

Other methods used for Review reason analysis:

Methods	SGD	RidgeClassifier	KNeighbors	VotingClassifier	GradientBoosting
F1 Score	57.55%	55.46%	52.39%	64.62%	61.53%

5. Interpretation of the Results

5.1 Why some technique worked, but not the others

5.1.1 SVM

As SVM does not work well with dataset with large dimensions, thus we were not able to do parameter tuning due to the high computational complexity and lack of computational power.

5.1.2 Naive Bayes with Word2Vec

As Naive Bayes does not accept negative inputs, Word2Vec could not be used directly. Hence, we perform a min max scaler to make sure that all the inputs are positive and we can train the naive bayes model. However, Word2Vec is a complex model and by converting the scale of the word vector might result in information loss. Hence, the model does not work as well.

5.1.3 SGD

We have excluded this model from the analysis as it performs poorer than the benchmark model.

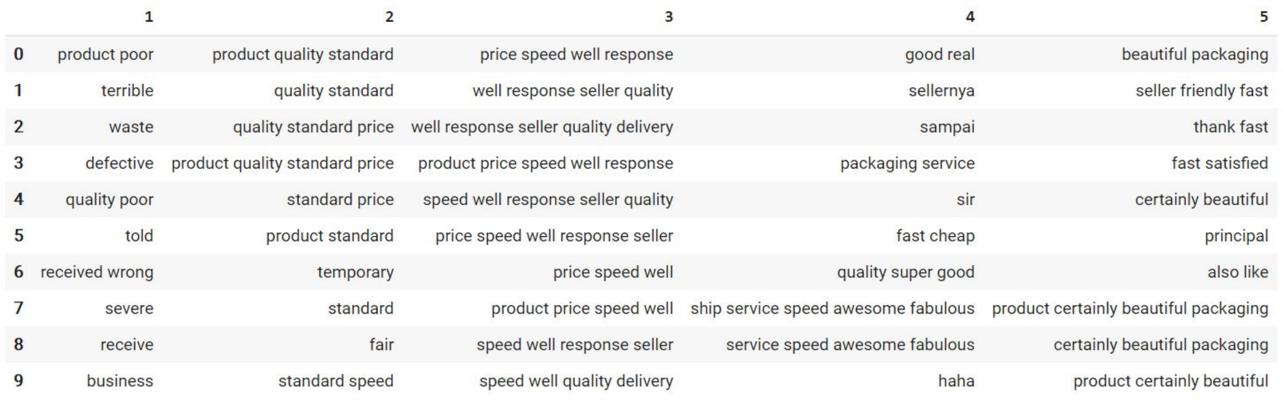
5.1.4 Clustering

As the both word2vec and TF-IDF vectors contained a high number of dimensions of data, the team faced increasingly difficult to perform the clustering due to either high computational complexity or RAM required. For the clustering method like Spectral , meanshift , agglomerative and affinity propagation clustering with word2vec data are not executable due to 200 GB Ram required for clustering our dataset. For the density based clustering like OPTICS and DBSCAN required a long execution period of more than 2 days for each round of clustering.

5.2 what feature are important to solve your problem, are they interpretable

5.2.1 TF IDF/ Word2vec

The chart below shows the top 10 most important features for Naive bayes with TF-IDF vectors with respect to each star rating. The features are not limited to only 1 word as we use an ngram_range between 1 to 5. We can see that for better ratings, they are filled with compliments such as beautiful packaging and fast. While for the poorer ratings tends to be complaints such as terrible and defective. However, for 2 star ratings, the features do not seem to explain much.



	1	2	3	4	5
0	product poor	product quality standard	price speed well response	good real	beautiful packaging
1	terrible	quality standard	well response seller quality	sellernya	seller friendly fast
2	waste	quality standard price	well response seller quality delivery	sampai	thank fast
3	defective	product quality standard price	product price speed well response	packaging service	fast satisfied
4	quality poor	standard price	speed well response seller quality	sir	certainly beautiful
5	told	product standard	price speed well response seller	fast cheap	principal
6	received wrong	temporary	price speed well	quality super good	also like
7	severe	standard	product price speed well	ship service speed awesome fabulous	product certainly beautiful packaging
8	receive	fair	speed well response seller	service speed awesome fabulous	certainly beautiful packaging
9	business	standard speed	speed well quality delivery	haha	product certainly beautiful

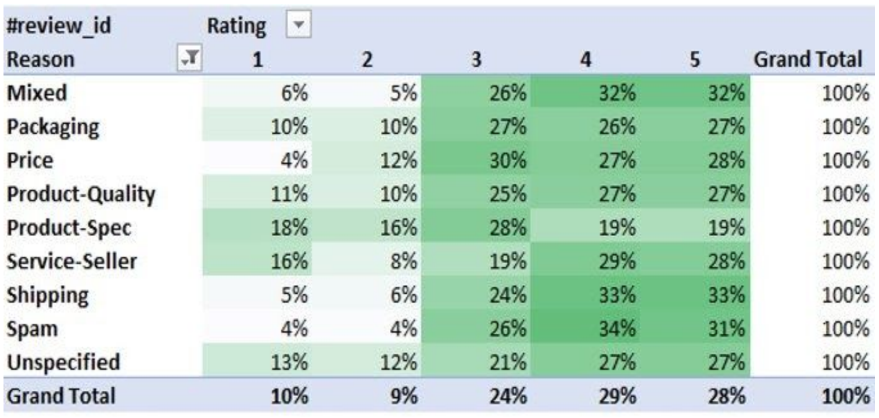
5.2.2 Top 10 most important features for LGBM with TF-IDF Vectors

Unlike Naive Bayes we are unable to get the importance based on each star rating. This might be the reason that not all the features in the top 10 make sense. Most of the features seem to come from compliments which may be a result of using an imbalance dataset as there are more good ratings compared to bad ratings. There are still words like "Severe" which may be a good indicator for bad reviews. As for "kaka" it is the malay word for sister which does not add much value to our analysis.

```
kaka
done
product use
response seller quality
good seller
product response
severe
took
price original speed response
time quality delivery
```

6. Future Extension

Customer experience is key to a company's success, and with sentiment analysis not having time is no longer a valid excuse. We have tried to incorporate a heatmap to determine the priority that should be given to an existing issue. This helps summarize reviews into actionable insights so that the company can make well-informed decisions. This helps us enhance the customer experience by regulating product quality and maintaining genuinely of sellers based on these decisions. If we had access to more data, we'd be able to increase the overall accuracy of our model which would help give the company better insights. The company can see what to boost and what to lose without wasting any time. According to Reevoo stats, reviews produce an average 18% uplift in sales. Companies have a lot to gain from positive reviews and a lot more to lose from negative reviews (Paxcom, 2020). This number is only going up with the increase in popularity of e-commerce due to the ongoing Covid-19 pandemic and has further boosted the importance of a review sentiment analysis tool. There is an underlying emotion for every action taken by a customer, whether it's a purchase or a negative review. This tool helps the company capture that emotion and act upon it. It is also a pivotal factor when it comes to optimizing marketing strategies and monitoring company reputation.



#review_id	Rating					
Reason	1	2	3	4	5	Grand Total
Mixed	6%	5%	26%	32%	32%	100%
Packaging	10%	10%	27%	26%	27%	100%
Price	4%	12%	30%	27%	28%	100%
Product-Quality	11%	10%	25%	27%	27%	100%
Product-Spec	18%	16%	28%	19%	19%	100%
Service-Seller	16%	8%	19%	29%	28%	100%
Shipping	5%	6%	24%	33%	33%	100%
Spam	4%	4%	26%	34%	31%	100%
Unspecified	13%	12%	21%	27%	27%	100%
Grand Total	**10%**	**9%**	**24%**	**29%**	**28%**	**100%**

References:

Sklearn. (n.d.). Retrieved November 22, 2020, from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.multilabel_confusion_matrix.html

Sklearn. (n.d.). Retrieved November 22, 2020, from https://https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Shmueli, B. (2020, July 03). Multi-Class Metrics Made Simple, Part II: The F1-score. Retrieved November 22, 2020, from https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1

Paxcom (2020). Why Is Customer Sentiment Analysis Important For Your Brand. paxcom.net