Exploring Performance of Classification Models with Count-based Text Representation for Three-Class English Twitter Sentiment Prediction Problem

Anonymous

1. Introduction

Research was conducted on predicting sentiments of English tweets (class label) as ‘Positive’, ‘Neutral’ and ‘Negative’ - a three-class **Natural Language Processing (NLP) supervised-learning classification problem** – given a particular tweet body. This report critically analyses and discusses the pre-processing and performance of various models for this problem.

2. Data Pre-processing

Data of NLP problems seldom come with ready-to-use features; this problem was no different. 21802 instances of this corpus had one raw attribute: **the tweet’s full text (avg. 18 words) -** pre-processed to lower case - hence requiring feature engineering.

We assume the dataset was un-biasedly sampled and is representative of this particular NLP problem.

Text cleaning followed Train-Test-Split, before features were engineered via text representation transformation.

2.1 Train Test Split

A 70-15-15 Train-Validation-Test split was used, ensuring substantial training size (15260) whilst Validation (used to tune hyperparameters) has the same size as Test (both 3271) and hence should not cause evaluation bias between the two sets.

2.2 Text Cleaning

As Twitter is a social media platform and hence predominantly written informally, “the frequency of misspellings and slang in tweets is much higher than…other domains”[1]. Texts also carry Twitter-specific semantics such as hashtags (characterised by #), usernames (@), HTMLs, and retweets. The data also contained emoticons (i.e. :)) which are used by netizens to express emotions in casual writing.

All twitter semantics except hashtags were removed as they are likely mutually distinct and thus unlikely to carry sentimental information. Numerical values were also removed because **numbers alone contain no specific sentiments** – any emotions associated is granted by context, something absent under the count-based representation we chose - and hence will only contribute to noise. Hashtags were retained because we believe they are a unique Twitter feature which places emphasis on the hash-tagged word, and hence should not be treated same in feature engineering as the same-but-un-hash-tagged word.

Meanwhile, we **replaced all common positive emoticons and negative emoticons with specific tags, unifying them in an effort to make two strong predictive features**. In the case where a tweet contained both positive and negative (polar) emoticons, **both would be removed to prevent confusion for learner**.

Shortening same characters successively repeated twice-or-more to two (i.e., ‘huuungry’ to ‘huungry’) removed noise as **English words**

[1] Go, A., et al. 2009. Twitter Sentiment Classification using Distant Supervision.

rarely contain more than two successive same characters; this boosts counts of words and hence likely creates more predictive features.

All non-English-alphabetical characters were removed to reduce noise as we assumed non-English characters do not contribute to sentiments. However, we kept within-word hyphens and dashes to **preserve normal English text and prevent unnecessary information loss**, given the **limited number of training cases compared to past research work on similar topics**.

2.3 Count-based Representation

The choice from the eight combination of stemming/lemmatisation, BoW/TF-IDF, singlegram/singlegram+bigram is essentially a hyperparameter, but tuning them with model hyperparameters is too expensive computationally. Hence, a greedy approach was used whereby each combination was trained for our four chosen models with default parameters and ranked by average accuracy for a validation set. Ultimately singlegram, stemming and TF-IDF was selected.

Our hypothesis for stemming outperforming the supposedly more sophisticated lemmatisation because **the latter took account for word context in transformation**, and given the small dataset severely diluted word counts as the same word potentially mapped to different roots. TF-IDF's victory is unsurprising given its consideration of word frequencies in other tweets, scaling down words that frequently appear in all tweets and thus capturing more information than BoW. Singlegrams' outperformance of single+bigrams was surprising given word ordering should retain valuable information; we offer the hypothesis that **bigrams have contributed more to noise than information due to low counts in this small dataset.**

2.4 Feature Selection

KBest with Chi-squared tests was used to seek the m-sized subset of independently most highly correlated attributes to the discrete class labels.

We treated this as a hyperparameter, beginning at *m*=¼ *#attributes*, and iteratively adding 1000 until *m*>21734 (original #attributes) to find the optimal #attributes for each model.

2.5 Models Chosen

Logistic Regression (LogR), Support Vector Machines (SVM), Extreme Gradient Boosting (XGB), Random Forest (RF), and a Stacker based on these models were explored; the first two common supervised classification models whilst XGBoost incorporating boosting and RF incorporating bagging.

Most-common-class 0R was our baseline.

3. Performance Analysis

We determined the best performer of the six models using multiple evaluation statistics.

3.1 Cross-Validated Accuracy

Accuracy measures the percentage of all correct predictions from all predictions. Five-fold Cross-Validation (CV) was performed to reduce evaluation variance and bring the accuracy scores closer to their underlying 'theoretical values' (by Central Limit Theorem taking average of many observations from the 'same' distribution reduces variance of the mean, and it limits to the 'true theoretical mean').

	0R	LogR	RF	SVM	XGB	Stacker
Train	0.581	0.730	0.633	0.727	0.742	0.741
Test	0.581	0.680	0.621	0.677	0.660	0.682

Table 1 CV Train Test Accuracies

All models outperformed the 0R, demonstrating improvement on predictability. Stacker topped the CV Test score with 0.682 whilst LogR and SVM trailed closely. RF was the worst performer, only outperforming 0R by 4%.

All of the CV Train and Test accuracy were within 8% of each other; suggesting **these {pre-process, model} pairs do not tend to suffer significantly from overfitting when well-tuned.**

3.2 (Weighted) Precision, Recall and F1_score

	0R	LogR	RF	SVM	XGB	Stacker
Precision	0.342	0.665	0.625	0.653	0.646	0.670
Recall	0.584	0.672	0.613	0.660	0.653	0.675
F1	0.431	0.653	0.520	0.635	0.616	0.652

Table 2 Test Precisions, Recalls, F1s

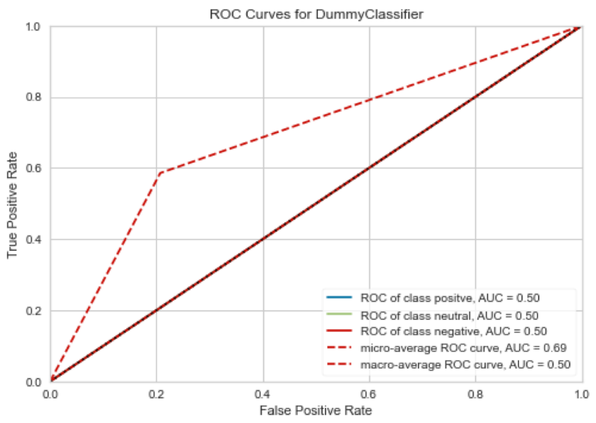
Precision measures the within-class percentage of accurate predictions from all predictions; Recall measures the within-class percentage of accurate predictions out of all observations. 'Weighted' means the statistics are averaged with consideration for the observed class distribution, more representative of this problem than any unweighted stat averages since this problem's data label distribution is non-uniform. The order of importance depends on which type of errors the actual use case of this problem wishes to avoid. These statistics provide more insights than point-estimate accuracy – the latter does not accurately reflect performance of models for non-uniform datasets.

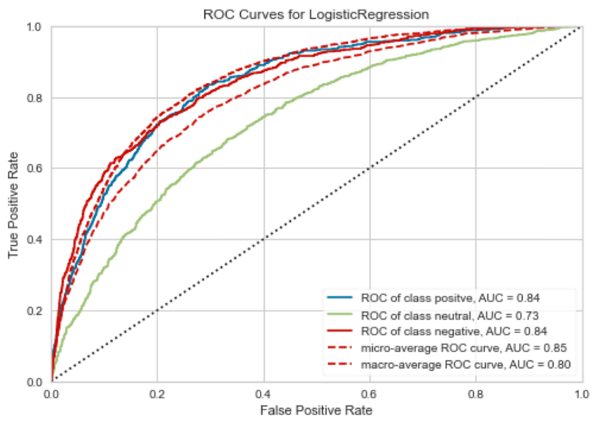
As our research prioritise neither error class, we chose to compare models by F1 – the aggregate of recall and precision. All models outperformed 0R, with LogR narrowly beating Stacker as the top performer by 0.001. RF slumped to 0.520, more than 0.09 below second last SVM but 0.081 above 0R, with the baseline let down by terrible precision as it predicts all instances as 'neutral'.

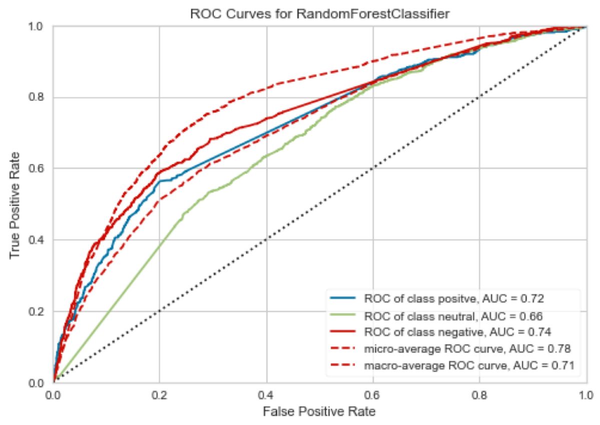
3.3 Receiver Operator Curve (ROC) and Area Under [RO]Curve (AUC)

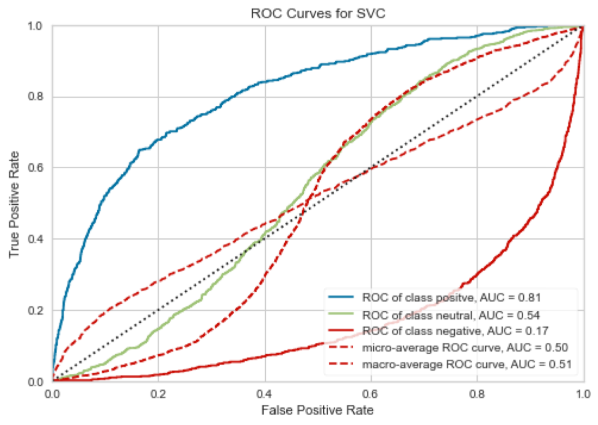
	0R	LogR	RF	SVM	XGB	Stacker
Macro Average AUC	0.5	0.8	0.71	0.51	0.78	0.52

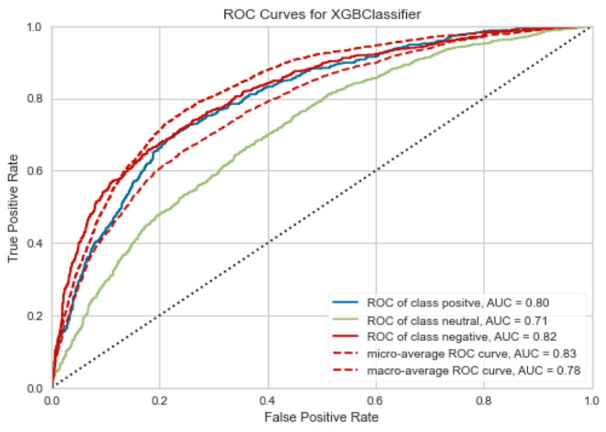
Table 3 AUC scores











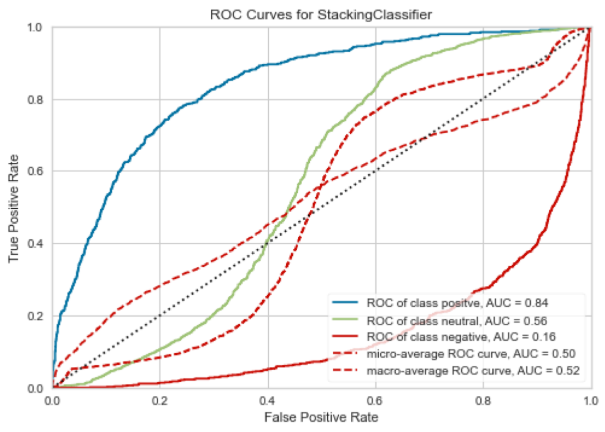


Figure 1-7 ROCs for all models

The ROC measures True Positives against False Positives at different decision boundaries/thresholds of a model for a particular dataset, typically for binary classifiers but can be modified for multinominal cases by averaging performance of each binary meta-classifier. Meanwhile, the AUC statistic can **indicate the usefulness of a model for this particular problem, as opposed to performance just on this dataset; hence it is a broader yet more abstract evaluative statistic.**

By AUC, LogR was once again the best model at 0.8, whilst Stacker and SVM slipped, barely outperforming 0R at 0.52 and 0.51 respectively; XGB is a firm second and RF third. This indicates that SVM-class solutions more are prone to false positives for this problem compared to other models, whilst Stacker's low AUC is likely due to its decision boundary/threshold being difficult as it is utilising multiple classes of underlying level-0 models, resulting in its low performance.

3.4 Summary

Overall, LogR has the best performance by three different measures. Stacker and SVM are second and third by dataset-specific scores but have disappointing AUCs, showing their enhanced suitability for this specific dataset over the actual problem.

4. Interpretation and Critical Analysis

4.0.1 Tuning

When tuning for hyperparameters, we used a brute force approach where all combinations of some discrete values of interested hyperparameters were trained and the combination producing the highest validation score was chosen.

4.0.2 Overall Trends

Although CV test accuracy failed to surpass 0.68, these models still provide valuable insights into these models' behaviour for this problem - even if ignoring that **0.68 more than doubles the random 0R baseline accuracy=0.33**.

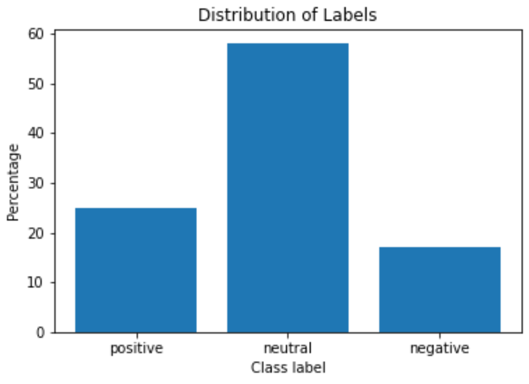


Figure 8 Full-Dataset Class-Label Distribution

Ignoring the problem of small dataset (Go's research used 1.4 million tweets), the highly uneven distribution of class labels in the full dataset (24%-58%-17%) severely hindered the ability to train models capable of learning the polar classes. Recurrent themes in the confusion matrices are the **strong tendencies to predict for and correctly predict the 'neutral' class, whilst correct predictions for the polar classes are less satisfactory**.

4.1 LogR

LogR is a binary classifier; under our choice to build a one-v-rest model, three meta-classifiers were built, and the **predicted label would be the 'one' class where the instance achieved highest probability**. LASSO regression was deployed to remove ineffective features to reduce overfitting, moderately successful as the difference in CV train and test accuracy was not large (5%).

Tuning returned the hyperparameters *C=1/λ=2* and *#features=10433* (48% of 21734). The choice of inverse LASSO regularisation strength *C=2* means weak regularisation (as they are inversely proportionate and *0<C<1* is considered small), suggesting **this problem suffers from using too few (even if highly predictive) features and requires sufficient even-if-moderately-effective features for good performance**. This is supported by the observation that **NLP problems have sparse features: count-based features are abundant due to the variety in natural language; but individual instances will have many '0-attribute-values'**, particularly for this problem as tweets have maximum 280 characters, and hence **retaining a substantial number of features is vital for learning**.

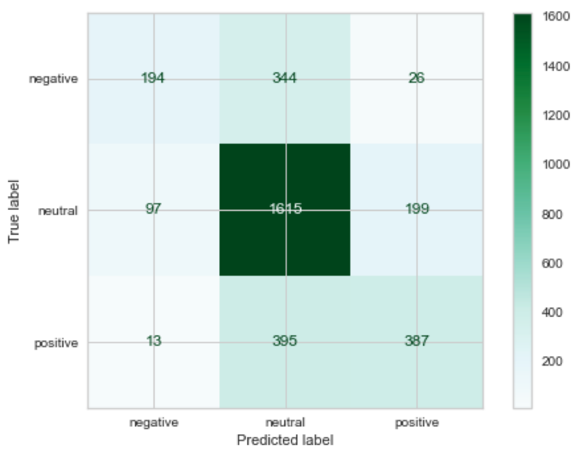


Figure 9 LogR Confusion Matrix

Overall, 72% of the instances were predicted to be 'neutral', and the *#(pred=neutral, true=polar)* outweighed *#(pred=polar, true = polar)*, indicating that the model has learnt the 'neutral' class well but not so much the other classes.

4.2 SVM

SVM classifies by building maximised margins using support vectors (the most difficult to categorise training instances). The 'one-v-rest' solution to the multinomial problem was chosen, and like LogR, three meta-classifiers were built, and the final prediction is the 'one' class furthest away from the boundary on the correct side. SVM has inbuilt feature selection that weighs down the ineffective features, hence no feature selection tuning required.

Margin trade-off constant *C=10* and a linear kernel were selected via tuning, indicating **preference for lower errors as increased *C* means harder boundaries**, and that the **classes are linearly separable**. This implicitly suggests higher dimensions would lead to overfitting which decreases validation accuracy).

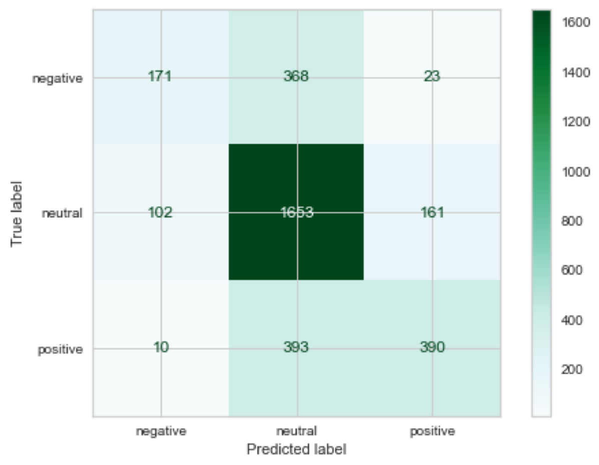


Figure 10 SVM Confusion Matrix

SVM's confusion matrix is even more 'neutral focussed' than LogR, with the predicted 'neutral's increasing to 73.8% of all test instances. Given validation data preferred a linear decision boundary, one explanation for this observation combined with the optimised hyperparameter *C=10* is that **many tweets sit on the boundary of the theoretical 'true' polar-neutral classes**, but because of the dominance of the 'neutral' class in the distribution, **the 'true boundary' given infinite dataset is heavily influenced by the 'neutral' boundary cases**, hence the large number of 'neutral's predicted. A softer boundary may lead to more 'neutral' train cases becoming wrongly classified and outweigh increase of polar classes correctly classified, and hence a **relatively hard boundary has been chosen to prioritise the accuracy of 'neutral' classes**.

SVM having inbuilt feature selection and worse accuracy than LogR supports the aforementioned 'moderately-useful features hypothesis'. Without LASSO's λ hyperparameter which LogR can tune, SVM has likely optimised for the training set and hence led to the same but more significant *#feature* problem discussed in LogR.

4.3 RF

RF is an ensemble classifier based on Decision Trees (which chooses the best feature and feature-value split at, using entropy, and has in-built feature selection), trained simultaneously with bootstrapped instances and randomly selected subset of features (after feature selection) for different trees that together classify by voting. Bagging theory suggests an ensemble of weaker classifiers should out-perform a single strong classifier, provided each tree learns different things and hence makes uncorrelated errors. Reasonable accuracy of underlying trees nonetheless remains essential.

Tuning returned *#trees=50*, *max_depth=10*, *max_features%=0.5*, *max_samples%=0.25* and *#features=5433* (25%).

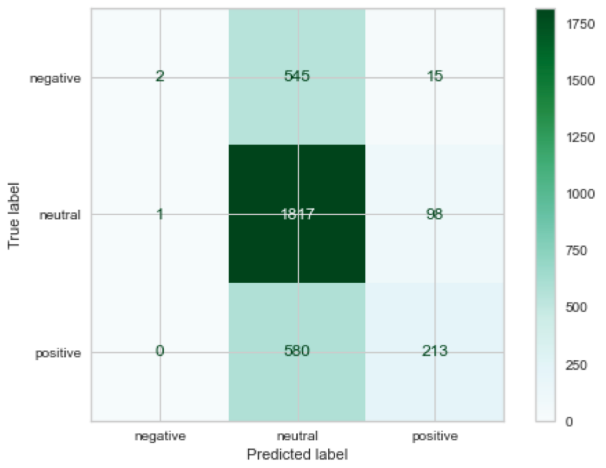


Figure 11 RF Confusion Matrix

RF's predictive pattern is highly skewed – almost no negatives and 89.9% 'neutral'. Recall for polar classes was extremely low, with accuracy predominantly supported by the correct predictions of 'neutral's. One reasonable explanation is that **underlying trees have learnt poorly**, which can be related again to the hypothesis of usefulness of moderately-predictive features: **most trees are likely struggling with reduced features and have all learnt predominantly to predict the dominant class. Hence, they lack diversity** which is a key assumption required for good bagging performance. This is supported by the fact that *max_features%>max_samples%*, meaning that trees rather seek diversity from selecting random instances than random features.

4.4 XGB

Gradient Boosted Trees (GBT) is the technique of **training an ensemble of Decision Trees in succession, weighting up the wrongly classified training instances in the previous iteration in the next bootstrapping.** Each tree's vote is weighted by its strength/accuracy. XGB is an improvement on GBT with **better penalisation of #trees, proportional shrinking of #leaf nodes and an extra randomisation parameter**.

Tuning returned *γ=1*, *max_features=0.5*, *max_features%=0.5*, *max_depth=6*, *eta=0.25*, *#features=9433*. A lower *eta* leads to stronger for regularisation, while a higher *γ* reduces regularisation.

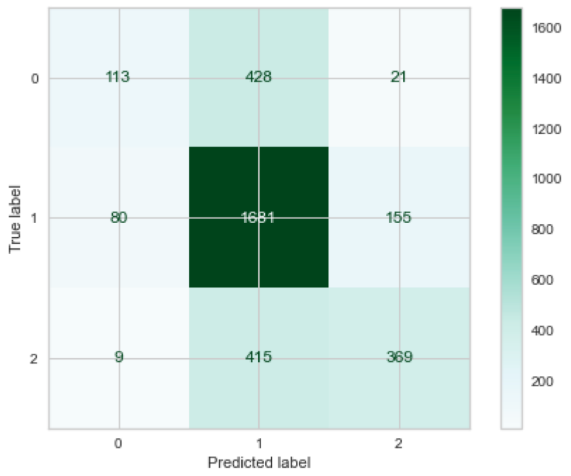


Figure 12 XGB Confusion Matrix

The performance of the XGB is slightly more 'neutral'-centric than SVM, and hence the two tree models were worst performance-wise. As XGB is also tree-based, arguments for its disappointment are largely same as RF.

A hypothesis for the failure of boosting to improve performance is that the dominance of the 'neutral' instances in training meant even when the weights of polar classes were increased in bootstrapping, the resulting bag-of-instances was still dominated by 'neutral's.

Based on the results, we further hypothesise that **tree-based classifiers are not suited for NLP problems**. Unlike LogR or SVM, trees successively take in attribute values of an instance; and with NLP problems having so many features where semi-useful instances may be essential for minor classes, **those latter class of attributes could be deep inside the tree or never reached upon *max_depth*, with tree nodes largely composed of attributes that are useful for the majority 'neutral' class**. Hence, tree-class base-models suffer from lacking sufficient learning for 'positive' and 'negative'.

4.5 Stacking

The Stacker is an ensemble method, where a final estimator uses predicted results of other estimators (based on input) to train and make its prediction. The *level-0 classifiers* are usually of different models, which provides uncorrelated but strong predictions that should help the *level-1 classifier* be better.

Tuning picked SVM as the *level-1 classifier* (and the other 3 as *level-0 classifiers*) and 13433 features for level-0 input.

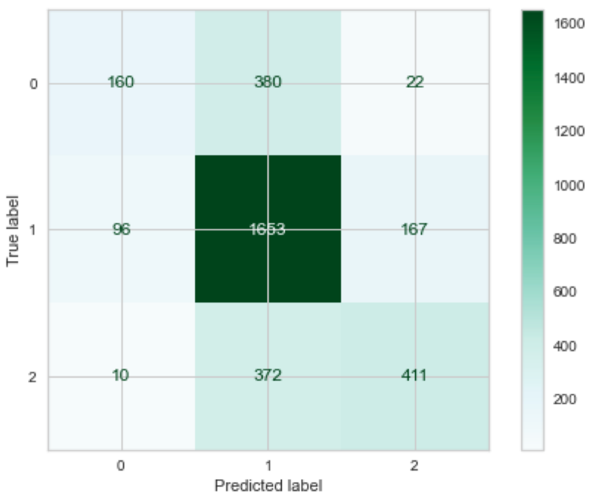


Figure 13 Stacker Confusion Matrix

Against regular SVM and LogR (our previous best), Stacker performed better for 'positives' whilst 'negatives' much the same, indicating the base classifiers all generally learnt 'positives' better than 'negatives'. This improvement also suggests **the three classifiers are behaving differently, where when one classifier cannot confidently predict a 'positive' instance correctly, one or more other base-classifiers frequently influence the prediction enough to amend to the correct classification**.

5. Evaluation and suggested improvements

More data will always improve model learning and reduce model variance. It would also make bi-grams and larger-gram counts less sparse and let them play a more meaningful role, as **NLP models should improves when word context is retained**. Currently, the lack of instances is the likely reason for the inclusion of bigrams causing more harm than good.

Other pre-processing methods (i.e., pre-trained word embeddings[2]) and more advanced Neural Network-based models (i.e., BERT, BiLSTM) should be implemented for more advanced research and superior performance. Information loss occurs when word order is eliminated in count-based representation, and **pre-trained transformations such as word-to-vector which**

[2] Le, Q, Mikolov, T., 2014. Distributed Representations of Sentences and Documents.

allows the tweet to be parsed as continuous phrases will capture more information; meanwhile, Neural Networks and other deep learning models will suit more to the complex task of NLP than the simpler models we used, **which are all likely underfitting the problem**.

6. Conclusion

LogR was found to be the best model when using {count-based representation, simple models} for this problem, with the best CV test accuracy 0.68.

Hypothesis for flaws in all evaluated models was discussed, the most prominent hypothesis on the importance of moderately-useful features in NLP problems. This research contributes to understanding the nature of NLP problems and forms a basis for researching more advanced models and pre-processing methods.

7. References

Go, A., et al. 2009. Twitter Sentiment Classification using Distant Supervision.

Le, Q. and Mikolov, T., 2014. Distributed Representations of Sentences and Documents.