COMP30027 Assignment 2: Report

Anonymous

1 Introduction

Sentiment analysis focuses on classifying text into sentiments. My goal is to apply sentiment analysis to develop and train three reliable sentiment classifiers and one baseline for Twitter posts (tweets). This involves extracting and selecting useful features from a dataset of tweets, then choosing and evaluating highly reliable classifiers.

1.1 Dataset

The dataset provided contains two lists of tweets/instances made prior to 2017 (Rosenthal et al., 2017). The training set in `Train.csv` contains 21802 labelled instances and the testing set in `Test.csv` contains 6099 unlabelled instances. For each instance, included is the text and tweet ID. The tweets included vary in content. For example, some are not in English: "*season in the sun versi nirvana rancak gak..slow rockkk...*". Tweets can either be "positive", "neutral" or "negative", distributed as shown in Figure 1.

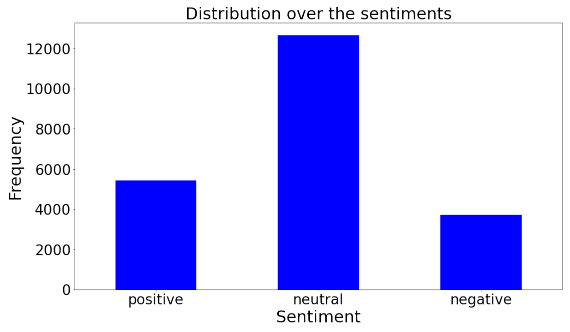

Figure 1: Training sentiment distribution

2 Methodology

The following methods are developed with reference to prior works on Twitter sentiment analysis (Go et al., 2009; Bird et al., 2009; Barbosa and Feng, 2010).

2.1 Instance cleaning

Some features extracted rely on the text in the tweets being cleaned. Cleaning involves removing stopwords (Section 2.1.1), links, hashtags, mentions, numbers, non-alphanumeric characters. Also performed is the reduction of repeated letters with more than two occurrences, as suggested by Go et al. (2009).

2.1.1 Stopwords

Manual stopword list construction is tedious and inexhaustive (consider other languages). Instead, I start with the Python Natural Language Toolkit's (`NLTK`) stopword list, which includes non-English languages (Bird et al., 2009).

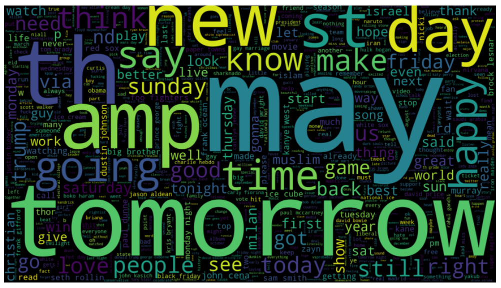

Figure 2: Word cloud with `NLTK` stopword removal

Next, based on Figure 2, some extra words are manually added to the list, leading to the final vocabulary list in Figure 3.

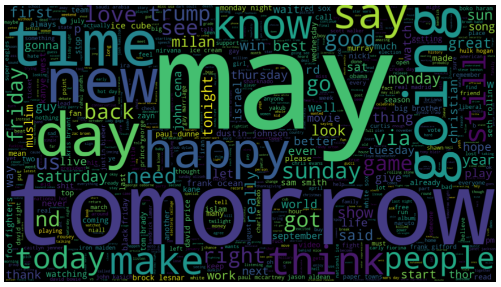

Figure 3: Word cloud with modified `NLTK` stopword removal

2.2 Feature extraction

These features are all chosen for their potential correlations with sentiments.

2.2.1 Twitter features

First, I extract platform-specific features from the tweets: hashtags, mentions, and links (Go et al., 2009; Barbosa and Feng, 2010). These are integrated within the platform, meaning they are widely used (commonplace in tweets).

2.2.2 Linguistic features

Next, linguistic features are extracted and used to tokenize the tweet in different ways: part-of-speech tags, lemmas, lemma 2-grams, phonetics (phenomes of words), punctuation, and emoticons (Barbosa and Feng, 2010; Bird et al., 2009; Weide, 1998). Lemmas are preferred over words, as they group words of the same root (Bird et al., 2009).

Emoticons are identified as string combinations of eye, middle and mouth characters, using a different method to Go et al. (2009), since they miss many emoticons, such as the crying `:'(`. Emoticons are then categorized into happy `:)`, sad `:(`, neutral `:|` or surprised `:o`.

2.2.3 Metric features

These are numeric counts of feature types within a tweet: words, characters, alphabetic characters, links, hashtags, mentions, and emoticons. Also considered are average word length and if a tweet is quoting another.

2.2.4 How many features per feature type is enough?

With over 20,000 unique instances in the training dataset, many feature types generate a large number of unique features (e.g. a large vocabulary of lemmas). Minimizing the feature space with a limit of features per feature type reduces model complexity. To determine this number M_f, the top candidate classifiers are compared for each of $M_f \in \{10, 100, 1000, 5000\}$ by the accuracies found using Section 2.4.1.

2.2.5 Vectorization

At the time that the data was collected, tweets were limited to 140 characters (Pardes, 2017). This means that features appear once (at most twice) in a tweet. Therefore, non-metric features are vectorized by occurrence counts, as TF-IDF is not applicable with near-binary features.

2.3 Classifier Selection

2.3.1 Baseline

The baseline model used is the 0-R: most frequent class (Classifier 4). The three chosen models (Classifiers 1, 2, and 3) need to outperform this.

2.3.2 Classifier Candidates

The following classifier/parameter combinations are considered for the final three models.

Multinomial/Bernoulli Naive Bayes:

Parameter	Options
Include Priors	Yes, No
α Smoothing	0, 1, 10

Table 1: Naive Bayes Parameters

Logistic Regressions: Logistic regressions do not have values modified. For the logistic regressions, `saga` optimization is used, as it is recommended for larger datasets (Pedregosa et al., 2011).

Decision Trees:

Parameter	Options
Maximum Depth	1, 100, 500

Table 2: Decision Tree Parameters

K **Nearest Neighbours**:

Parameter	Options
K	1, 10, 100, 500
Vote Weighting	`uniform`, `distance`

Table 3: K-Nearest Neighbour Parameters

Support Vector Classifiers:

Parameter	Options
Kernel	Linear, Cubic
Regularization C	0.1, 1, 3
Decision Function	One-v-One, One-v-Rest

Table 4: Support Vector Classifier Parameters

2.4 Evaluation

2.4.1 Which Parameter Configurations Are Best?

Of the combinations in Section 2.3.2, the top three (Classifiers 1, 2, and 3) are determined with the following scoring system. First, cross-validation scores are calculated on the training set. Next, cross-validation is performed on the maximum subset of data with a uniform sentiment distribution. The 10 scores are averaged $\bar{x}$, and their standard deviation s found. The best candidate configurations have the highest bound for the right-sided 95% confidence interval of mean cross-validation scores:

$$\begin{aligned}95\%\text{ C.I. Bound} &= \bar{x} - \mathbb{F}_{t_9}^{-1}(0.95) \times \frac{s}{\sqrt{n}} \\ &= \bar{x} - 1.8331 \times \frac{s}{\sqrt{10}}\end{aligned}$$

where t_9 is the t-distribution with 9 degrees of freedom and $n = 10$.

2.4.2 Evaluation Metrics

Accuracies, precisions, recalls and F_1 scores (per sentiment) are generated for each classifier that makes it past Section 2.4.1.

2.4.3 Which models are label distribution agnostic?

It is unknown whether the `Test.csv` and `Train.csv` sentiments are similarly distributed, so evaluation metrics from Section 2.4.2 are calculated on four 4 : 1 train/test splits.

Where "Given" denotes a set of data with sentiment distributions similar to those in Figure 1, and "Uniform" denotes a maximum subset of data with uniformally distributed sentiments, evaluation metrics are generated on the train/test combinations in Table 5. The metrics are then compared to determine which models are agnostic to sentiment distributions.

Training	Testing
Uniform	Uniform
Given	Given
Given	Uniform
Uniform	Given

Table 5: Training and testing splits

3 Results

3.1 Highest cross-validation accuracy per number of maximum features

After generating classifiers for each of the M_f values, the following boasted the highest scores calculated in Section 2.4.1.

Classifier Parameters	Logistic Regression With Intercepts & 500 Max Iterations
$\bar{x}$	0.5524
s	0.04734
95% C.I. Bound	0.5250

Table 6: $M_f = 10$ Top classifier

Classifier Parameters	Linear SVC $C = 0.1$
$\bar{x}$	0.6027
s	0.02625
95% C.I. Bound	0.5875

Table 7: $M_f = 100$ Top classifier

Classifier Parameters	Support Vector Classifier One-v-One Decisions, $C = 0.1$ & Linear Kernel
$\bar{x}$	0.6415
s	0.02002
95% C.I. Bound	0.6299

Table 8: $M_f = 1000$ Top classifier

Classifier Parameters	Logistic Regression With Intercepts & 100 Max Iterations
$\bar{x}$	0.6441
s	0.008317
95% C.I. Bound	0.6392

Table 9: $M_f = 5000$ Top classifier

Classifier Parameters	Logistic Regression Wihout Intercepts & 100 Max Iterations
$\bar{x}$	0.6447
s	0.006991
95% C.I. Bound	0.6406

Table 10: $M_f = 10000$ Top classifier

These results are used to generate Figure 4.

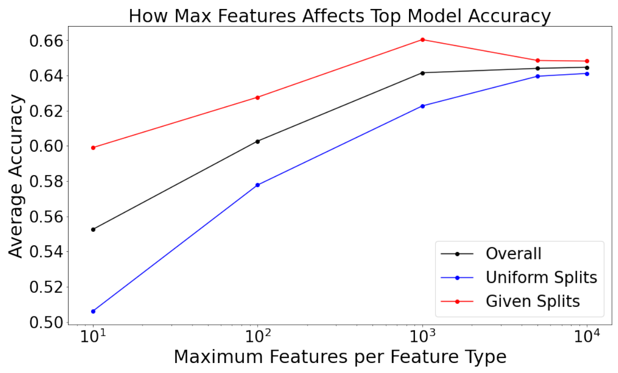

Figure 4: How M_f affects the cross-validation accuracies of the highest-scoring classifiers.

3.2 Top 3 parameter/classifier configurations for $M_f = 10000$

The following classifiers have the highest scores calculated in Section 2.4.1:

Classifier Parameters	Logistic Regression Wihout Intercepts & 100 Max Iterations
$\bar{x}$	0.6447
s	0.006991
95% C.I. Bound	0.6406

Table 11: Classifier 1

Classifier Parameters	Logistic Regression With Intercepts & 100 Max Iterations
$\bar{x}$	0.6449
s	0.008343
95% C.I. Bound	0.6401

Table 12: Classifier 2

Classifier Parameters	Linear SVC One-v-One & $C = 0.1$
$\bar{x}$	0.6491
s	0.01622
95% C.I. Bound	0.6397

Table 13: Classifier 3

3.3 Evaluation metrics for the final 4 models

The metrics describes in Section 2.4.2 are calculated for the final classifiers (Classifiers 1, 2, 3, and 4). Per-class metrics are shown in order of postive (+), neutral (n), then negative (–) sentiments.

Classifier Parameters	Logistic Regression Without Intercepts & 100 Max Iterations
Uniform Training	Uniform Testing
Accuracy	0.64
Precisions (+, n, –)	0.69, 0.54, 0.69
Recalls (+, n, –)	0.68, 0.55, 0.69
F_1 Scores (+, n, –)	0.69, 0.54, 0.69
Given Training	Given Testing
Accuracy	0.65
Precisions (+, n, –)	0.69, 0.65, 0.53
Recalls (+, n, –)	0.40, 0.89, 0.18
F_1 Scores (+, n, –)	0.51, 0.75, 0.27
Given Training	Uniform Testing
Accuracy	0.54
Precisions (+, n, –)	0.81, 0.43, 0.82
Recalls (+, n, –)	0.48, 0.91, 0.24
F_1 Scores (+, n, –)	0.60, 0.58, 0.37
Uniform Training	Given Testing
Accuracy	0.73
Precisions (+, n, –)	0.64, 0.91, 0.57
Recalls (+, n, –)	0.84, 0.62, 0.93
F_1 Scores (+, n, –)	0.73, 0.74, 0.71

Table 14: Classifier 1 metrics

Classifier Parameters	Logistic Regression With Intercepts & 100 Max Iterations
Uniform Training	**Uniform Testing**
Accuracy	0.64
Precisions (+, n, –)	0.69, 0.54, 0.69
Recalls (+, n, –)	0.68, 0.54, 0.69
F_1 Scores (+, n, –)	0.68, 0.54, 0.69
Given Training	**Given Testing**
Accuracy	0.64
Precisions (+, n, –)	0.68, 0.65, 0.52
Recalls (+, n, –)	0.40, 0.89, 0.17
F_1 Scores (+, n, –)	0.50, 0.75, 0.26
Given Training	**Uniform Testing**
Accuracy	0.54
Precisions (+, n, –)	0.81, 0.43, 0.82
Recalls (+, n, –)	0.48, 0.91, 0.23
F_1 Scores (+, n, –)	0.60, 0.58, 0.36
Uniform Training	**Given Testing**
Accuracy	0.73
Precisions (+, n, –)	0.64, 0.91, 0.57
Recalls (+, n, –)	0.84, 0.62, 0.93
F_1 Scores (+, n, –)	0.73, 0.74, 0.71

Table 15: Classifer 2 metrics

Classifier Parameters	Linear SVC One-v-One & $C = 0.1$
Uniform Training	**Uniform Testing**
Accuracy	0.64
Precisions (+, n, –)	0.70, 0.53, 0.68
Recalls (+, n, –)	0.67, 0.55, 0.69
F_1 Scores (+, n, –)	0.69, 0.54, 0.69
Given Training	**Given Testing**
Accuracy	0.65
Precisions (+, n, –)	0.66, 0.66, 0.54
Recalls (+, n, –)	0.46, 0.85, 0.26
F_1 Scores (+, n, –)	0.54, 0.74, 0.35
Given Training	**Uniform Testing**
Accuracy	0.65
Precisions (+, n, –)	0.85, 0.50, 0.92
Recalls (+, n, –)	0.61, 0.91, 0.43
F_1 Scores (+, n, –)	0.71, 0.65, 0.58
Uniform Training	**Given Testing**
Accuracy	0.72
Precisions (+, n, –)	0.66, 0.89, 0.55
Recalls (+, n, –)	0.82, 0.62, 0.92
F_1 Scores (+, n, –)	0.73, 0.74, 0.69

Table 16: Classifer 3 metrics

Classifier	0-R
Uniform Training	**Uniform Testing**
Accuracy	0.33
Precisions (+, n, –)	0.00, 0.00, 0.33
Recalls (+, n, –)	0.00, 0.00, 1.00
F_1 Scores (+, n, –)	0.00, 0.00, 0.50
Given Training	**Given Testing**
Accuracy	0.58
Precisions (+, n, –)	0.00, 0.58, 0.00
Recalls (+, n, –)	0.00, 1.00, 0.00
F_1 Scores (+, n, –)	0.00, 0.73, 0.00
Given Training	**Uniform Testing**
Accuracy	0.33
Precisions (+, n, –)	0.00, 0.33, 0.00
Recalls (+, n, –)	0.00, 1.00, 0.00
F_1 Scores (+, n, –)	0.00, 0.50, 0.00
Uniform Training	**Given Testing**
Accuracy	0.17
Precisions (+, n, –)	0.00, 0.00, 0.17
Recalls (+, n, –)	0.00, 0.00, 1.00
F_1 Scores (+, n, –)	0.00, 0.00, 0.29

Table 17: Baseline (Classifier 4) metrics

4 Analysis

4.1 How M_f modifies the models

It appears that *accuracy* $\propto M_f$. The number of maximum features M_f does affect the accuracy of models, as shown in Figure 4. As M_f increases, the accuracy of the top model increases as well, albeit asymptotically. This confirms that a larger feature space allows for more accurate results to a limit. With 10 features per feature type, the odds of running into an instance without any of these is higher. As the feature space increases in size, more features can be considered when classifying a new instance, meaning more instances are likely to contain at least one feature to consider for classification. Past $M_f = 5000$, the gains in accuracy become negligible. Despite this, the accuracy does increase, leading to the use of $M_f = 10000$ to generate and evalute the final models.

The average accuracy for cross-validation scores over the splits with the given sentiment distributions does not follow the same relationship to M_f as the other accuracy scores. This could potentially suggest that past $M_f = 1000$, training or testing on the given sentiments is unadvisable.

The modal top classifier within this test is the logistic regression. These results highlight the model's versatility with different sizes of the feature space.

One outlier is the linear support vector classifier at $M_f = 1000$. This confirms that selecting the final models required deeper investigation into classifier results than simple cross-validation. Such analysis occurs in Sections 4.2 and 4.3.

4.2 The best classifier/parameter configurations

Of the classifiers evaluated at $M_f = 10000$, the top three models (shown in Tables 11, 12, and 13) are comprised of two logistic regressions and a linear support vector classifier. Interestingly, the Bernoulli naive Bayes classifiers do not appear in the top classifier list. The nature of the features and tweet lengths means that almost all the features are binary events (Bernoulli trials), making the naive Bayes classifier very applicable (Pardes, 2017). On the other hand, logistic regressions by nature are also well suited to binary features, and have improved performance through the use of optimization algorithms.

The Classifiers 1, 2, and 3 all perform better than the baseline model (see Section 3.3). The only exception is in the recalls of the 0-R, in which the baseline model has the best results. However, this is misleading since the 0-R baseline classifier labels all instances with the same class, guarranteeing that one of the classes has no false negative values.

4.3 The most sentiment distribution agnostic classifier

Since the `Test.csv` and `Train.csv` data are not guarranteed to follow the same sentiment distributions, the true best model is selected by comparing F_1 scores. As shown in Section 3.3, the sentiment distribution of the training and testing data influences the evaluation metrics of classifiers.

From the accuracies of the top 3 models on different distribution splits (Tables 14, 15, and 16), training on given data and testing on uniform data yields consistently high precisions and recalls for all sentiments. This is the opposite for training on given data distributions, where precisions are higher than recalls for non-neutral sentiments, suggesting that confirming a non-neutral sentiment at the point of classification proves difficult.

Thus, training on the data with the given sentiment distribution is not preferrable if the true testing data does not follow the same sentiment distribution. In fact, training on uniform data generally yields better accuracy and higher F_1 scores, even if the distribution of sentiments in the test set is not uniform. Therefore, the best-performing models non-neutral sentiments are trained on the maximum uniform subset instead of the full dataset.

The only hyperparameter that differs between Classifier 1 and Classifier 2 is inclusion of intercept terms in the regression. The inclusion of intercepts in the regression models can bias the models towards different sentiments. To measure this bias, the

5 Conclusions

Performing sentiment analysis on tweets is a difficult task. Vectorization of the features yields a large feature space, which contributes to the time and space complexity in training classifiers. While minimizing the feature space reduces complexity, it also sacrifices the accuracy of classifiers (some more than others).

Testing a multitude of different classifier combinations, the highest scoring (calculated by Section 2.4.1) classifiers are naive Bayes classifiers, with the best model relying specifically on a Bernoulli naive Bayes classifier. Bernoulli models make sense in this case, given the small sizes of tweets meaning features appear no more than once in an instance (Pardes, 2017). With little information on the `Test.csv` set, the models generated need to be highly accurate regardless of the distributions of the training sentiments.

Of the classifiers considered, the most efficacious, is a Bernoulli naive Bayes model trained on the maximum subset of data with uniform sentiment distribution with laplace smoothing $\alpha = 1$, without fitting prior distributions.

References

Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 36–44, USA. Association for Computational Linguistics.

Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing*, 150, 01.

Arielle Pardes. 2017. A brief history of the ever-expanding tweet. *Wired*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada, August. Association for Computational Linguistics.

R. L Weide. 1998. Carnegie mellon pronouncing dictionary. www.cs.cmu.edu.