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.

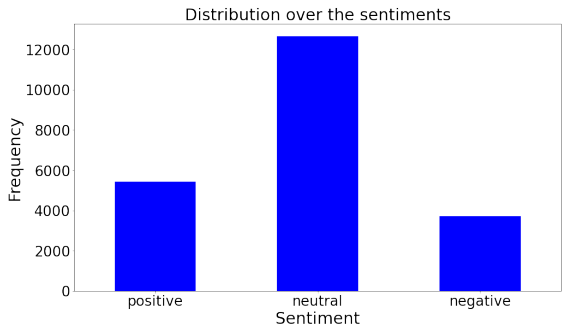

Figure 1: Training sentiment distribution

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 omits stopwords from other languages. Instead, I start with the Python Natural Language Toolkit's (`NLTK`) stopword list, which includes non-English languages (Bird et al., 2009).

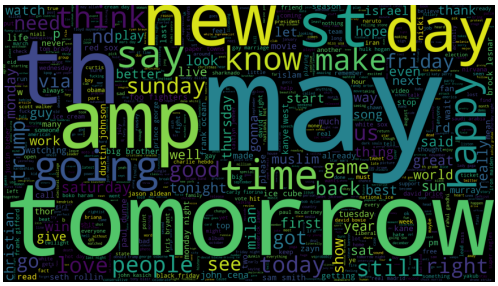

Figure 2: Word cloud with `NLTK` stopword removal

Next, referring to Figure 2, some extra words are manually added to the final stopword list, leading to the final training vocabulary in Figure 3.

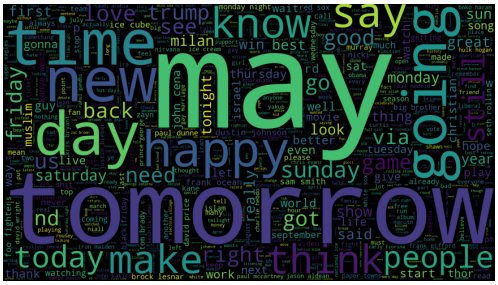

Figure 3: Word cloud with modified `NLTK` stopword removal

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 (phonemes 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 Binary features

Binary features track the appearance of other feature types within a tweet: whether the tweet contains quotes, a hashtag, mention, link, or emoticons. These binary values may correlate to non-neutral sentiments.

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 $M_f \in \{10, 100, 1000, 5000, 10000\}$ 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 parameters 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

Linear Support Vector Classifiers:

Parameter	Options
Regularization C	0.1, 1, 3

Table 4: Linear 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. Cross-validation scores are calculated on the training set with 10 folds. 10 folds allow for a random assortment of large subsets to be scored . 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 (5th percentile accuracy):

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

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 uniformly 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	Logistic Regression
$\bar{x}$	0.6009
s	0.03539
95% C.I. Bound	0.5804

Table 6: $M_f = 10$ Top classifier

Classifier	Logistic Regression
$\bar{x}$	0.6334
s	0.01925
95% C.I. Bound	0.6222

Table 7: $M_f = 100$ Top classifier

Classifier	Logistic Regression
$\bar{x}$	0.6599
s	0.008548
95% C.I. Bound	0.6549

Table 8: $M_f = 1000$ Top classifier

Classifier	Logistic Regression
$\bar{x}$	0.6662
s	0.006952
95% C.I. Bound	0.6621

Table 9: $M_f = 5000$ Top classifier

Classifier	Logistic Regression
$\bar{x}$	0.6676
s	0.006673
95% C.I. Bound	0.6637

Table 10: $M_f = 10000$ Top classifier

These results are used to generate Figure 4.

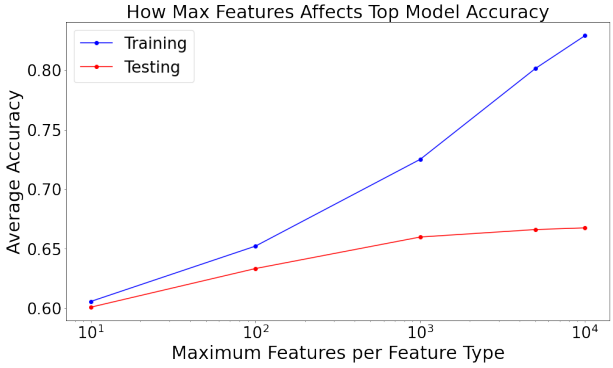

Figure 4: How M_f affects the cross-validation accuracies of the highest-scoring classifiers.

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	Logistic Regression
$\bar{x}$	0.6676
s	0.006673
95% C.I. Bound	0.6637

Table 11: Classifier 1

Classifier Parameters	Linear SVC $C = 0.1$
$\bar{x}$	0.6615
s	0.005878
95% C.I. Bound	0.6581

Table 12: Classifier 2

Classifier Parameters	Multinomial Naive Bayes Without Priors & $\alpha = 1$
$\bar{x}$	0.6434
s	0.006020
95% C.I. Bound	0.6399

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	Logistic Regression
Uniform Training	Uniform Testing
Accuracy	0.63
Precisions (+, n, –)	0.68, 0.53, 0.68
Recalls (+, n, –)	0.66, 0.54, 0.69
F_1 Scores (+, n, –)	0.67, 0.54, 0.69
Given Training	Given Testing
Accuracy	0.66
Precisions (+, n, –)	0.64, 0.67, 0.56
Recalls (+, n, –)	0.49, 0.84, 0.29
F_1 Scores (+, n, –)	0.55, 0.74, 0.38
Given Training	Uniform Testing
Accuracy	0.65
Precisions (+, n, –)	0.84, 0.50, 0.88
Recalls (+, n, –)	0.60, 0.90, 0.44
F_1 Scores (+, n, –)	0.70, 0.65, 0.59
Uniform Training	Given Testing
Accuracy	0.75
Precisions (+, n, –)	0.66, 0.92, 0.60
Recalls (+, n, –)	0.85, 0.64, 0.96
F_1 Scores (+, n, –)	0.74, 0.76, 0.74

Table 14: Classifier 1 metrics

Classifier Parameters	Linear SVC $C = 0.1$
Uniform Training	Uniform Testing
Accuracy	0.64
Precisions (+, n, –)	0.67, 0.54, 0.68
Recalls (+, n, –)	0.68, 0.52, 0.71
F_1 Scores (+, n, –)	0.68, 0.53, 0.70
Given Training	Given Testing
Accuracy	0.65
Precisions (+, n, –)	0.60, 0.68, 0.56
Recalls (+, n, –)	0.52, 0.80, 0.31
F_1 Scores (+, n, –)	0.56, 0.73, 0.40
Given Training	Uniform Testing
Accuracy	0.73
Precisions (+, n, –)	0.85, 0.58, 0.92
Recalls (+, n, –)	0.71, 0.89, 0.58
F_1 Scores (+, n, –)	0.77, 0.70, 0.72
Uniform Training	Given Testing
Accuracy	0.74
Precisions (+, n, –)	0.65, 0.92, 0.59
Recalls (+, n, –)	0.86, 0.63, 0.96
F_1 Scores (+, n, –)	0.74, 0.75, 0.73

Table 15: Classifer 2 metrics

Classifier Parameters	Multinomial NB Without Priors & $\alpha = 1$
Uniform Training	Uniform Testing
Accuracy	0.62
Precisions (+, n, –)	0.65, 0.56, 0.63
Recalls (+, n, –)	0.71, 0.40, 0.75
F_1 Scores (+, n, –)	0.68, 0.47, 0.69
Given Training	Given Testing
Accuracy	0.63
Precisions (+, n, –)	0.65, 0.63, 0.59
Recalls (+, n, –)	0.35, 0.91, 0.10
F_1 Scores (+, n, –)	0.46, 0.75, 0.16
Given Training	Uniform Testing
Accuracy	0.59
Precisions (+, n, –)	0.84, 0.45, 0.95
Recalls (+, n, –)	0.52, 0.93, 0.31
F_1 Scores (+, n, –)	0.65, 0.61, 0.46
Uniform Training	Given Testing
Accuracy	0.64
Precisions (+, n, –)	0.56, 0.89, 0.48
Recalls (+, n, –)	0.83, 0.48, 0.89
F_1 Scores (+, n, –)	0.67, 0.63, 0.63

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. These results merit the use of $M_f = 10000$ to generate and evaluate the final models.

The highest 5th percentile accuracy models for all M_f values tested are consistently linear regressions. This is a testament to the classifier's versatility with both small and large feature spaces.

The accuracies on the training data are consistently higher than those of the testing data (Figure 4). As M_f increases, the difference between the training and testing accuracies widens. This may be an indicator of overfitting in the data.

Despite the risk of overfitting, the final models are trained with $M_f = 10000$ so as to maximise the result accuracy. My theory is that using any larger value of M_f will result in diminishing returns in testing accuracy, as the model starts to overfit the training data.

4.2 The best classifier/parameter configurations

With the selected feature space $M_f = 10000$, the highest 5th percentile accuracies come from:

1. the logistic regression classifier,
2. the linear support vector classifier (SVC) with regularization $C = 0.1$,
3. the multinomial naive Bayes classifier with laplace smoothing $\alpha = 1$ and without priors

as per Section 3.2.

The logistic regression relies on optimization, mapping instances to a multinomial class distribution. This classifier scores the highest of all candidates, likely due to its optimization algorithm. While optimization is slow (and not guaranteed to find the ideal fit), the model weights features proportionally to their importance sentiment classification.

The linear SVC also relies on optimization to find supports to generate lines separating the instances by sentiment. It applies a one-vs-rest decision function, generating three lines separating each sentiment from the other two. This makes the model effective at determining classes which are very different from the rest. However, a weakness of the SVC is in distinguishing between classes with similar features. Furthermore, the model is highly reliant on a small subset of instance vectors to classify all other instances. While this simplicity reduces memory use, it can also abstract away useful distinctions for specific features.

The multinomial naive Bayes performs better than the Bernoulli naive Bayes classifier. This result is not expected due to the expectation of binary features explained in Section 2.2.5. This naive Bayes model also performs best without fitting sentiment priors. The consideration of priors is heavily reliant on whether the sentiment distribution in the `Test.csv` matches the `Train.csv`. The naive Bayes classifier relies on simple laplace smoothing $\alpha = 1$. As α increases or decreases, missing features become either too underrepresented or overrepresented (negatively affecting accuracy).

The Classifiers 1, 2, and 3 all perform better than the baseline model (Table 17). The only exception is the recall of the 0-R, in which the baseline model has the best results for some classes. This is misleading since the 0-R baseline classifier labels all instances with the same sentiment, guaranteeing that one of the classes has no false negatives.

4.3 The most sentiment-distribution-agnostic classifier

Since the `Test.csv` and `Train.csv` data are not guaranteed to follow the same sentiment distributions, the true best model is selected by comparing the evaluation metrics over different train/test distribution splits. As shown in Section 3.3, the sentiment distribution of the training and testing data influences the evaluation metrics of classifiers.

Based on the accuracies only, training on a uniform sentiment distribution is preferable. However, certain models may perform better for certain sentiments. Since neutral sentiments are most prominent in the training set, models that do not consider the prior sentiment distributions (either due to a parameter or due to the training set used) generally yield higher neutral precisions than recalls, indicating more false non-neutral than false neutral classifications.

For most applications of sentiment analysis the priority is to identify non-neutral instances. Therefore, high precisions and recalls for non-neutral labels are prioritised. This can be simplified into finding the models which generate the highest non-neutral F_1 scores. Comparing Tables 14, 15, and 16, the linear SVC and logistic regression have the best F_1 scores in these sentiments. Specifically the best scores are generated on uniform training sets, where both models perform similarly in precision and recall. Therefore, either the linear SVC or the logistic regression can reliably be used on the `Test.csv` data, given they are trained on a maximal subset of the training data with uniform sentiment distribution.

5 Conclusions

Performing accurate sentiment analysis on tweets is difficult. Vectorization of the features yields a large feature space, which contributes to the time and space complexity in training classifiers and poses the risk of overfitting results. While minimizing the feature space reduces model training complexity, it also sacrifices the accuracy of classifiers.

Testing a multiple classifier combinations, the most accurate classifiers by estimated 5th percentile are the logistic regression, linear SVC ($C = 0.1$), and multinomial naive Bayes ($\alpha = 1$ and without priors). With little information on the `Test.csv` data, the chosen models need to be highly accurate regardless of distributions of the testing or training sets. Both the linear SVC and logistic regression can be used classify the `Test.csv` instances. However, given the disadvantages of the linear SVC's reliance on a one-versus-rest decision function, **I choose to generate the final predictions on a logistic regression trained on a maximals subset of instances with uniform sentiment distribution.**

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.