Paper Template for COMP30027 Report

Anonymous

1. Introduction

The goal of this project is to predict the sentiment of a set of tweets. The sentiment of a given tweet could be positive, negative, or neutral. A few machine learning models are built based on training and validating on 21802 tweets with their sentiments labelled. The model with the best performance will be selected as the model to predict the sentiment of 6099 tweets in the test dataset.

2. Method

2.1 Data preprocessing

Under the first hypothesis (to be discussed more in-depth in later subsections), hyperlinks, usernames, and non-alphanumeric characters are removed from the tweets. This is followed by stopword removal and lemmatisation.

For the remaining hypotheses, the unprocessed tweets are used. That is, the original version of tweets is used for feature vectorization.

2.2 Feature vectorization

The training and test data are vectorized using the Term Frequency-Inverse Document Frequency (TF-IDF) method to generate features for model training. This unsupervised feature selection method is used as it is useful in capturing important terms while giving less weight to common and rare words. It is also a common technique used in Natural Language Processing (NLP).

2.3 Hypothesis formulation

To find out the most useful features for training classifiers, several hypotheses are formulated.

H0: Lemmatising processed tweets with stopwords removed give a higher accuracy than the baseline.

In this project, we define 'processed tweets' as tweets where hyperlinks, usernames, and non-alphanumeric characters have been removed. We expect common NLP practices such as stopword removal and lemmatisation will contribute to a decent accuracy which is higher than the baseline.

H1: Using features from processed tweets leads to better classifier performance than using unprocessed tweets.

To understand how well processing tweets helps in improving the performance of sentiment classification, this hypothesis is formulated to compare the effect of using processed and unprocessed tweets as features for training. To test this hypothesis, unprocessed tweets are used for feature selection instead.

H2: Using unigrams and bigrams as features improves the performance of sentiment classification.

Following experiments run to test H0 and H1, we then propose H2 as bigrams may contain better information on the sentiment of a tweet (Go, Bhayani, & Huang, 2002). To illustrate, phrases like 'not bad' and 'not good' depict a different sentiment from 'good' and 'bad', respectively. Hence, we expect adding bigrams as features for training will yield better performance than using just unigrams. The unigrams and bigrams are extracted from the unprocessed tweets.

H3: Performing feature selection using chi-square improves the performance of sentiment classification.

As the feature space is large, we hypothesise that using chi-square to select features that are highly correlated to the class will improve the performance of classification. This helps to trim the feature space and only retains features that are highly dependent to the class. We perform feature selection with varying values of k on unprocessed tweets.

2.4 Classifier selection

2.4.1 Baseline

To obtain the baseline accuracy, the zero-R method is used. The most common sentiment in the training dataset is 'neutral'. Using this classification, a baseline accuracy of 0.58 is obtained.

2.4.2 Support Vector Machine (SVM)

A linear support vector classifier (SVC) is used in our project. This classifier is chosen as according to Joachims (1998), SVM is suitable for text classification since it can handle high dimensional feature space and most text categorisation problems are linearly separable.

For each hypothesis, we tune the hyperparameter C, the slack penalty rate, to obtain the value that leads to the highest accuracy. We tune C by first comparing the accuracy when using C = [0.25, 0.5, 0.75, 1.0]. We then further tune C by minimising the range around the value that leads to the previous highest accuracy. For instance, if C=0.5 gives the highest accuracy in the first round of tuning, the second round of tuning will be done in the range of 0.3-0.7, with a difference of 0.05 between every two value.

As this is a multiclass classification problem, we use the one-versus-all method to convert it into binary classification problems and combine the margins to obtain the decision boundaries.

2.4.3 Naïve Bayes

Multinomial Naïve Bayes is another classifier we use as the features are discrete. The prior probability for each class is estimated from the training set. It also calculates the likelihood of each feature occurring given each class. Laplace smoothing is performed when applying this classifier. An important assumption made when using Naïve Bayes is that each feature is independent of each other when conditioned on a class.

2.4.4 K-Nearest Neighbour (KNN) Classifier

The last classifier used is K-Nearest Neighbour classifier. This is an instance-based classifier which can produce flexible decision boundaries.

For each hypothesis, the hyperparameter k, the number of nearest neighbours to compare classification with, is tuned. The tuning is done by comparing the accuracy using k=[3, 5, 7, 9] and choosing the value of k which leads to the highest accuracy. The distance function used is Euclidean distance, and each neighbouring instance is given uniform weight. This means the instance is classified according to the majority class of its k nearest neighbours.

2.5 Evaluation methods and metrics

Two evaluation methods are used in this project. The first method is performing train-test-split on the training dataset. The dataset is split into 80% training set and 20% validation set. The validation set is used to evaluate each classifier's performance after being fitted with the training set. The accuracy on this validation set is then being adopted as one of the evaluation metrics to compare the performance of each classifier.

The second method is performing 10-fold cross validation on the training set. We then take the average cross validation score across all folds as the second evaluation metric.

3 Results

Under all hypotheses, the linear SVC yields the best performance over other classifiers. On the other hand, the Multinomial Naïve Bayes and K-Nearest Neighbours classifiers return evaluation scores that are near to the baseline accuracy. Figures 1 to 3 show the comparison of evaluation scores from each classifier under different hypotheses to the baseline accuracy.

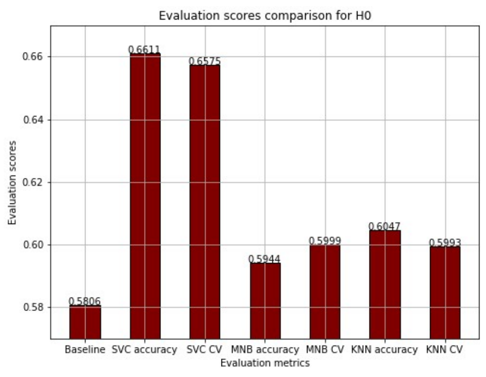

Figure 1 - comparison between evaluation scores under H0.

Figure 1 shows that H0 is supported. All classifiers return an evaluation score higher than the baseline accuracy.

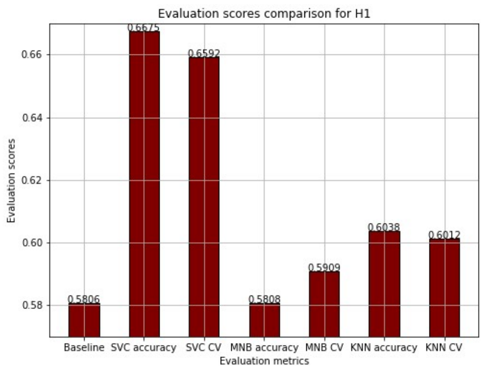

Figure 2 - comparison between evaluation scores under H1.

Based on the evaluation scores of the best performing classifier, linear SVC, H1 is not supported. Thus, unprocessed tweets are used in the following hypotheses for feature engineering as this set of data yields a better result.

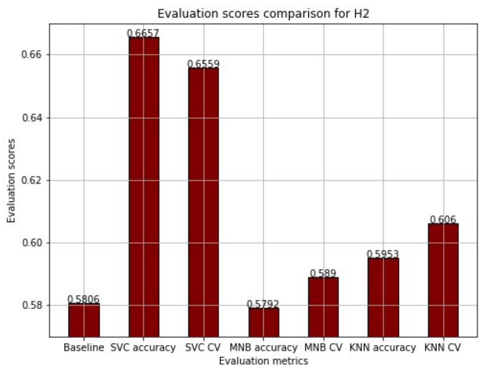

Figure 3 - comparison between evaluation scores under H2.

The overall evaluation scores from H2 are lower than that of H1, which indicates that H2 is not supported. Adding bigrams into the feature space does not improve the performance of classification.

As can be seen from the results, linear SVC is the best performing classifier among all classifiers used. Thus, we decide to focus on linear SVC with C=0.35 (as found to be the optimal value under H1) in H3. Further, the feature selection is applied on the unprocessed tweets. Figure 4 shows the comparison of accuracy, average cross validation, and F1 scores of using chi-square for k-best feature selection.

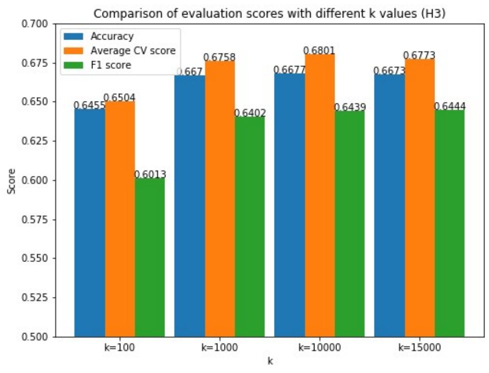

Figure 4 - comparison between evaluation scores under H3.

Considering the accuracy and average cross validation scores, selecting 10000 features that are most correlated with the class results in the highest overall performance. The F1 score of k=10000 feature selection does not differ much from k=1000 and k=15000 as well.

As the linear SVC gives the highest accuracy, we decide to use it as our classifier on the test set. To find out under which hypothesis does the classifier gives the best result, we sum up the accuracy and average cross validation scores and take the average of them for each hypothesis. This gives us the final evaluation score for each hypothesis. Figure 5 shows the comparison of this average evaluation score across all hypotheses.

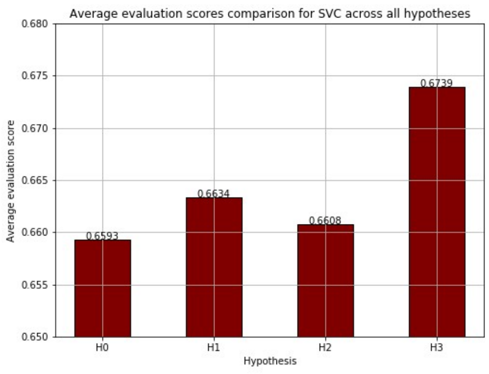

Figure 5 - comparison between the average evaluation score from linear SVC under all hypotheses.

As shown in Figure 5, we get the highest average evaluation score when applying linear SVC under H3, where we use unprocessed tweets for feature engineering and select 10000 features most correlated with the class using chi-square. Hence, we decide to fit a linear SVC with C=0.35 on 10000 best features obtained from unprocessed tweets vectorization using the TF-IDF method on the test set.

4 Discussion

Across all experiments, linear SVC gives the best performance among all classifiers implemented. This might be due to the linear separability of the dataset. Another observation is that the optimal values of the slack penalty rate found by tuning for each linear SVC is relatively small (lying in the range of 0.25-0.4). This is in contrary to the notion that a smaller C value leads to more misclassifications as it allows a larger margin. This also suggests that the instances are inherently quite linearly separable.

On the other hand, the Multinomial Naïve Bayes classifier returns the lowest accuracy and average cross validation scores among all classifiers. As Naïve Bayes is a probabilistic classifier which makes predictions based on prior probability and likelihood calculations, there might be features in the test set that was not seen in the training set. Therefore, these features would have missing likelihoods, which could cause inaccuracies. Moreover, the features might be somewhat correlated, which violates the assumption that all features are independent.

Further, the performance of the K-Nearest Neighbour classifier is also relatively poor. This could be due to the large feature space generated from the variety of words in the tweets. As each instance only contains a small portion of the features, instances from the same class could be far from each other in the feature space. This results in a sparse feature space, and thus the majority of the nearest neighbours of a target instance could be from different classes.

Comparing the results of H0 and H1, it is observed that using unprocessed tweets for feature vectorization results in slightly better performance than using processed tweets. Given that the difference in the results is minimal, this could be due to the effect of the randomised train-test-splits. Yet, in accordance with the results, we decide to further our experiments using the unprocessed data.

The results of H2 are in contradiction to our hypothesis, where adding bigrams as features does not improve the classifiers' performance. This could be due to the increased dimensionality reducing the linear separability of the instances. Using a SVM with polynomial kernel might improve the performance.

Applying feature selection improves the performance of the linear SVC as it reduces the dimension of data and noise from unpredictive features. K=10000 features is about one-third of the original size of the feature space, which allows more correlated features to be considered in the prediction process than k=100 or k=1000 features but includes less noise than when using k=15000 features.

The distribution of the training dataset should be noted as well. The training set consists of 58% neutral instances, 25% positive instances, and 17% negative instances. This imbalanced sampling could lead to a bias towards classifying the test instances by the majority class, which is the neutral class. A more balanced dataset could remedy this bias.

5 Conclusion

In conclusion, linear SVC is a fairly suitable classifier for text classification problems such as sentiment analysis as these problems are generally linearly separable (Joachims, 1998). Further, feature selection is important for reducing the dimensionality of the feature space and identify predictive features from the dataset.

6 References

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N project report, Stanford, 1*(12), 2009.

Joachims, T. (1998, April). Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning* (pp. 137-142). Springer, Berlin, Heidelberg.

Rosenthal, S., Farra, N., & Nakov, P. (2017, August). SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)* (pp. 502-518).