

COMP30027 Assignment 2 Report

Anonymous

1 Introduction

The goal of this project was to predict the star ratings of a review of a restaurant based on the review text. The data available for learning and evaluation consisted of approximately 250000 Yelp reviews, and is made available by (Medhat et al., 2014) and (Rayana and Akoglu, 2015). Each review consists of the review text, a rating (1, 3 or 5 stars) and review metadata (which was not used). We sought to investigate to what extent it was possible to predict the rating of a review from the review text alone.

2 Background

This task is a type of sentiment analysis: the task of predicting sentiment from text. [discussion of background literature](#)

3 Method

We investigated models based on logistic regression (detailed in Section 3.1) and support vector machine (detailed in Section 3.2) techniques, in each case using the implementations made available through the Scikit Learn Python library (Pedregosa et al., 2011). These models used paragraph vector encodings of the review text, which was produced using the implementation made available through the Gensim Python library (Řehůřek and Sojka, 2010).

[Move to results?](#) In addition to model-specific hyperparameters (which will be discussed in the respective sections) the dimension of the parameter vector space embedding is a hyperparameter for all of our models. This can be interpreted as a measure of complexity of the model. As the dimension of the paragraph vector encoding a review increases, the degree to which the paragraph vector can capture the information in the review will increase too. Not all of the information in a review is likely to be useful for this classification task however¹, so there is a point

at which increasing the dimension of the feature space will lead to over-fitting and degrade the performance of the classifier.

3.1 Logistic regression

[Explain machine learning models used, and why they were used.](#)

3.2 Support vector machine

We investigated three variants of support vector machine (SVM) classifiers: a SVM with a linear kernel and one-verses-rest multi-class classification (Linear-SVM), a SVM model with a radial basis function (RBF) kernel and one-verses-rest multi-class classification (RBF-SVM), and a SVM with a linear kernel separating the positive and negative sentiment classes, in which marginal instances are classified as having neutral sentiment (Binary-SVM).

All three SVM-based model have a regularisation coefficient C as a hyperparameter, which determines the trade-off between margin maximisation and training error minimisation. As C increases the margin is increased, however more training instances are allowed to be misclassified. The RBF-SVM also has the kernel hyperparameter γ , which in this case was used to scale $\frac{1}{\text{Var}(X) \cdot \text{numfeatures}}$ ([explain this better](#)). The Binary-SVM has the probability threshold hyperparameter, which is the probability a negative of positive sentiment prediction must exceed to be classified as that class [awkward wording here](#).

Justifying SVM model selection

Paragraph vector encoding is well suited to classification by an SVM model since paragraph vectors have meaningful geometric relationships to one another (Le and Mikolov, 2014), and SVM models attempt to exploit the geometry

the cuisine of the restaurant, the time-of-day of the visit and the gender of the wait-staff, much of which is likely irrelevant to the task at hand.

¹Review text also may contain information such as

of the feature space by fitting a separating hyperplane. For example, the relative distance between points in the paragraph vector space is relevant to their relative meaning. This is in contrast Bag-of-Words vectors, in which the vector encodings of the (one word) texts “horrible”, “terrible” and “great” are equidistant in the Bag-of-Words feature space.

The paragraph vector encoding of text is also designed to transfer relationships in the meaning of text to linear relationships between paragraph vectors: “king” - “man” + “woman” = “queen” (Le and Mikolov, 2014). If we suppose that negative and positive sentiment reviews have exactly opposite meanings (with neutral sentiment reviews somewhere in between) then we would expect the review text of these classes to be linearly separable with the paragraph vector encoding. Preliminary testing of different kernel functions (without tuning any hyperparameters) indeed showed that an SVM classifier with a linear kernel performed the best in every metric, however an SVM classifier came quite close in performance. For this reason we chose to continue with extended analysis of both.

Under the assumption that positive and negative sentiment reviews are exactly opposite in meaning and that neutral sentiment reviews are some mixture (i.e. a linear combination in paragraph vector space) of positive and negative meanings, it seems plausible that the neutral sentiment reviews will lie close to the hyperplane which separates the positive and negative sentiment reviews. This encourages the consideration of a binary SVM classifier which distinguishes positive and negative reviews, where instances which fall in the margin are classified as neutral. Rather than using the raw distance to the hyperplane, opted to use a probabilistic variant of SVM which estimates the probability of an instance belonging to each class based on its distance to the hyperplane (Platt, 1999). This allowed us to use a probability threshold for neutral classification as a hyperparameter, rather than a distance threshold, which means the value of this threshold is independent of the specific paragraph vector embedding². A probability threshold also has the benefit of making the model more interpretable. Despite its apparent theoretical foundations, results in the

literature suggest that such a classifier will always result in worse performance as compared to using a standard multi-class approach (Koppel and Schler, 2006). Nevertheless we chose to pursue this approach as well.

3.3 Stacking model

4 Results

discussion of why we used stratified splits? In order to identify the optimal feature space dimension for each model, we first evaluated our model (using what metric?) (with hyperparameters untuned) on paragraph vector of dimensions 25 to 300 in steps of 25 encodings using an 80:20 stratified random holdout, only using the training set in each case to compute the paragraph vector embedding function. A random holdout method, rather than cross-validation, was used due to the computation expense of computing paragraph vector encodings: producing cross-validation splits for many dimensions was out of the question. For the logistic regression, RBF-SVM and Binary-SVM models this dimension was 125, and for the Linear-SVM model this dimension was 150.

We then computed the paragraph vector encodings for a 5-fold stratified cross validation split at that dimension, again only using the training set of each split to produce the encoding. This pre-computed split was then used to tune the remaining hyperparameters. Since each cross validation run took a long time parameter values were initially adjusted in large increments, and then a finer full grid search was done on regions of interest. The results of this can be seen in Figure insert grid search figure. What are the best hyperparameters? with reference to the figures Why cross validation? (to prevent overfitting the hyperparameters)

After tuning the hyperparameters, we evaluated the final model trained on paragraph vectors of dimensions from 25 to 300 in steps of 25 to produce the learning curve in Figure insert figure of learning curves. As discussed in Section 3 the dimension of the paragraph vector encoding is a measure of the complexity of our model. We also evaluated our model on the 5-fold cross validation split at the optimal dimension, the results of which can be seen in Figure insert figure of confusion matrix, and any other stuff we want to talk about in analysis.

I think the stacking model would have been treated differently, need to talk about this here.

Describe and compare model performance.

²The specific embedding of a particular text is only relevant in the context of the training corpus used to compute the embedding function (Le and Mikolov, 2014).

(i.e. just say what the figures show)

5 Analysis

Explain and interpret results. Identify weaknesses in methodology (more cross validation runs, more kernels for SVM). include comparison to baseline classifier (Zero-R, we don't actually need to compute this can just look at class proportions), error analysis via confusion matrix (put the figure in the results section though)

- All models have best performance on feature vectors of a similar dimension. Suggest this is because this is the best dimension for this text corpus for representing the information relevant to the rating. It is a good thing this is independent of the model since...
- As expected the Binary-SVM performed worse than the SVM classifiers using a ovr approach to multi-class classification. This is consistent with literature. Note that is is a lot faster though.
- All classifiers cap at around the same performance accross all metrics, including the stacking classifier. It is possible that this is just the best that can be done with this feature representation of the review text. Also can discuss how it is not reasonable to expect perfect prediction even given a perfect representation of text features: the review score gives a (human) reader context for the sentiment of the review text. eg "chips were good" with a 1 and 5 star review gives very different impressions (maybe think of a better example).

6 Conclusion

Summarise main points and (possibly) future work.

References

- Moshe Koppel and Jonathan Schler. 2006. The importance of neutral examples for learning sentiment. 22(2):100–109.
- Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents.
- Walaa Medhat, Ahmed Hassan, and Hoda Korashy. 2014. Sentiment analysis algorithms and applications: A survey. 5(4):1093–1113.

- Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. 2013. What Yelp fake review filter might be doing? In *7th International AAAI Conference on Weblogs and Social Media*.
- 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. 12:2825–2830.
- John Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. 10(3):61–74. Publisher: Cambridge, MA.
- Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 985–994.
- Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50. ELRA.