Classifying the Geolocation of Tweets

Anonymous

1 Introduction

This report analyses different machine learning classification algorithms to output the geolocation of users according to their tweets. We have three datasets in our disposition, namely the training dataset- to train the classifiers and tune parameters, the development dataset- to evaluate the performance of the classifiers trained on the training dataset, and the testing dataset- consisting of only features and no labels. The training dataset has 133796 instances, the development dataset has 11476 instances and the test dataset has 12019 instances. The classes are regions in the United States labelled as 'MIDWEST', 'NORTHEAST', 'SOUTH', 'WEST'.

This task is relevant in the social media industry, mainly in the data analytics field. It would be relatively easy to predict the geolocation of a user given their IP address, Wifi footprints or GPS data (Rahimi et al., 2018). However, this information is rarely publicly available and it would be a breach of privacy data if we were to access them without permission (Boehme-Neßler, 2016).

2 Literature Review

There are many research papers based on the association between language and geography (Pavalanathan & Eisenstein, 2015). For instance, Pavalanathan & Eisenstein (2015) incorporated age and gender in their analysis of the tweet-location relationship using a 'latent variable model'. They demonstrated how twitter texts vary depending on population demographic variables. However, their findings (Pavalanathan & Eisenstein, 2015) were highly focused on the given names of twitter users, which is not always readily available.

In addition, Rahimi et al. (2018) tackled social media user geolocation by proposing a semi-supervised geolocation model- Graph Convolutional Networks (GCN). They found that 'highway network gates' were key to controlling the amount of useful neighbourhood expansion in their model. Moreover, they showed that modelling text and network together performed better than using text and network separately.

The dataset used in this report was obtained from the resource published by Eisenstein et al. (2010).

3 Method

In this section, we explore the features and models used in our classification and the reasoning behind their selection.

3.1 Features

This report uses the 'count' dataset provided, as features for our classification.

The rationale is that we are more interested in the grammar, terms and expressions used, references to local topics and places etc. in each tweet (HaCohen-Kerner et al., 2020). We are less interested in the importance of each word in a tweet or the overall meaning of a tweet. Therefore using word frequencies as features is preferred to using tfidf values (Schütze et al., 2008) or glove300 vectors (Pennington et al., 2014). Note however that the words have been filtered using tfidf so that only the frequency of relevant words are used in the analysis. Each word is mapped to a unique ID.

The representation of the training data, as used in the code, is shown in Table 1 below. Assume there are n tweets and m distinct relevant words in the training dataset where x_{ij} represents the frequency of word j in tweet i and y_i is the region of tweet i.

Tweet ID	Word 1	Word 2	...	Word m	Region
1	x_{11}	x_{12}	...	x_{1m}	y_1
2	x_{21}	x_{22}	...	x_{2m}	y_2
...	...	...	...	...	...
n	x_{n1}	x_{n2}	...	x_{nm}	y_n

Table 1: Training data representation

Upon inspection of the 2038 tfidf-filtered words, we still notice a significant number of irrelevant features such as emojis, ellipsis, words like 'awww' and 'hahahah' etc. We perform further feature selection by picking the 500 features with the highest χ^2 values. We use χ^2 instead of Mutual Information since χ^2 reflects the dependence between a feature and the labels whereas Mutual Information shows how much information a feature gives about the labels. χ^2 examines the raw counts, i.e. takes into account the sample size, which increases the confidence that some dependence relationships exist (Li, 2017). Contrarily, Mutual Information only examines the marginal and joint probability distributions.

Since there are a significant number of zeros in the table, we represent the features as a sparse matrix in the code, instead of a dense matrix. This allows us to save some memory and improve the time complexity of our algorithms.

Trivially, we use the same feature selection and transformation for the development and test data (without the labels in the test data).

3.2 Baseline

We use the Zero Rules (majority class) baseline in this report.

We prefer it to the random baseline since Zero-R is consistent, whereas the results for a random baseline has variance- it changes each time the function is called since it is built on a probability distribution. In this case, Zero-R is also better than One-R because our data representation consists of a large amount of features and several zero-valued features for each tweet. Hence, no single feature would be a decent depiction for the label, even for a baseline.

3.3 Candidate Models

3.3.1 Multinomial Naive Bayes

We choose Multinomial Naive Bayes as a candidate model because of its simplistic approach, generally good performance empirically and easy interpretability. It is also popular in document classification problems using the frequency of words as features (Patil, 2017). Since the features in our dataset are discrete, we use the multinomial distribution when implementing the Naive Bayes classifier.

Yet, the Naive Bayes assumption tells us that given the geolocation of a tweet, all word counts are independent to each other. This is important to point out since it is not applicable with the way we represent our data- a large number of attributes, especially numerical ones, introduce new dependencies in the data. Hence, the conditional independence assumption is violated. However, the model still works well empirically despite the strong assumption it imposes (Patil, 2017).

Because of all the zeros, we use Laplace smoothing with $\alpha = 1$. Even though this adds bias to the classifier, as we no longer have a true maximum likelihood estimator, it still generally works well (Patil, 2017) and is better than rendering meaningful observations irrelevant.

3.3.2 Logistic Regression

Contrary to Naive Bayes, Logistic Regression makes no assumption about the distribution of labels in feature space (Stoltzfus, 2011). It also adjusts the model coefficients to indicate the importance of each feature- the coefficient of an important feature would have a high magnitude whereas an irrelevant feature would have a coefficient close to zero. The classifier is particularly well suited for frequency-based attributes, which we are using (Stoltzfus, 2011). In addition, Logistic Regression is still easy to implement, and efficient to train.

The main limitation of Logistic Regression is that it has a linear decision surface. This poses a problem since it is very unlikely that our data set is linear separable since it is based on real-world data (Tsai & Lin, 2011). Further, regularisation is required to avoid over-fitting, which is highly probable in a high dimensional dataset. We use the default 'l2' regularisation technique implemented in the *sklearn* package in Python.

3.3.3 Multilayer Perceptron

We implement a Multilayer Perceptron (neural network) because it is a powerful and complex classification algorithm. Similarly to Logistic Regression, is has automatic feature learning- this means that it will assign appropriate coefficients to each feature and this is done at each layer too. It also works well with non-linear data (using non-linear activation functions), hence the algorithm can learn more complex decision boundaries, which is more suitable for our real-word data. This provides us with a different perspective as compared to the previous two classifiers, who have strictly linear decision boundaries.

On the other hand, large scale neural networks require powerful GPU's for them to work effectively (Toliušis & Kurasova, 2017). As a re-

sult, they can take a long time to compute outputs. Further, neural networks demand that we hand-select the number of hidden layers and the number of neurons per hidden layer, which can only be done through trial and error (Toliušis & Kurasova, 2017). We implement a Multilayer Perceptron with one hidden layer containing 100 neurons. This makes sense as a convergence pattern since we have 500 features-$500 \rightarrow 100 \rightarrow \hat{y}$. Figure 1 illustrates our model.

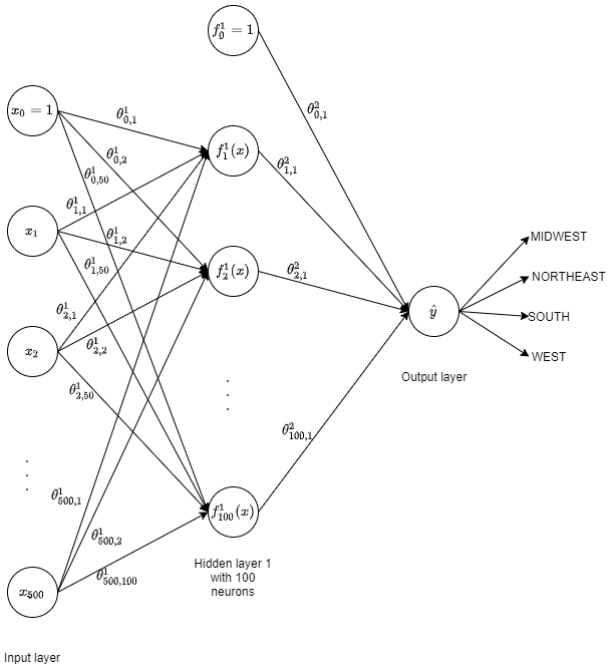

Figure 1: Multilayer Perceptron with 1 hidden layer

We use the sigmoid function as the activation function for the hidden layer and the output function is the softmax function since we are looking at a multi-class classification.

3.3.4 Decision Tree

The main reason we are using a Decision Tree is because its structure is very similar to how we (humans) tackle decision-making problems. Also, its decision boundary is non-linear, which again, gives us another perspective about the decision boundary of the data. The results from a decision tree is arguably the easiest to interpret since it is a white-box model (Tofan, 2014). It also imposes no assumption on the shape of our data.

In contrast, a decision tree is highly prone to overfitting. This is because at each node, the algorithm looks at every possible split of every single variable and hence overcomplicates its logic. It mostly happens at deep parts of the tree. To avoid overfitting, we limit the depth of the tree to 50, which is fairly little considering we have 500 features. In addition, information gain (the criterion that we use in the code) tends to have a bias towards features with a high dimension.

3.4 Evaluation Metrics

The accuracy and F1 scores, using macro-averaging, are used to evaluate our models. We choose accuracy since we want to know how often a model correctly classifies a tweet geolocation. The F1 score provides us with a harmonic mean of the model's precision and recall.

4 Results

We train each model on the training dataset and use the features in the development dataset to predict their labels. Then, the results are computed in terms of accuracy score and Macro F1 score which are shown in Table 2 below. Note that for comparison, we also include the models' counts for each label, as well as the actual empirical count.

The barchart in Figure 2 displays a visual representation of the evaluation metrics for each model.

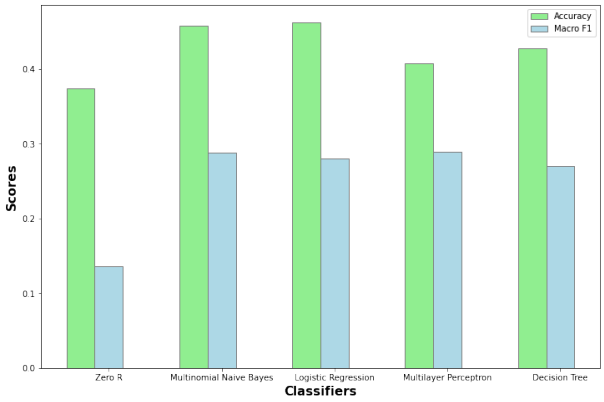

Figure 2: Barchart of evaluation metrics

5 Critical Analysis

5.1 Contextualisation

Overall, a good sign is that all candidate classifiers outperform the Zero Rules baseline in terms of both accuracy and Macro F1 scores. On another note, we deduct that all four models are biased towards the 'NORTHEAST' and 'SOUTH' labels and hence misjudge the labelling of the 'MIDWEST' and 'WEST' classes.

Classifier	'MIDWEST' Count	'NORTHEAST' Count	'SOUTH' Count	'WEST' Count	Accuracy Score	Macro F1 Score
Zero R	0	11475	0	0	0.37	0.14
Multinomial Naive Bayes	82	5236	5838	319	0.46	0.29
Logistic Regression	25	5464	5819	167	0.46	0.28
Multilayer Perceptron	632	4733	5372	738	0.41	0.29
Decision Tree	211	5420	5580	264	0.43	0.27
Empirical	*1484*	*4295*	*4266*	*1430*		

Table 2: Classification reults on development dataset

This calls attention to the imbalanced empirical distribution of the dataset itself- it is highly skewed towards the 'NORTHEAST' and 'SOUTH' classes.

In particular, the Multinomial Naive Bayes and Logistic Regression clearly outperform the Multilayer Perceptron and the Decision Tree classifiers. This is illustrative of the overfitting that comes along with the two latter models. Tweaking the number of hidden layers and number of neurons in each layer of the Neural Network would most likely improve the model's performance. In theory, the Multilayer Perceptron is a more advanced and improved version of the Logistic Regression classifier since they both use sigmoid functions, but the former has 100 neurons whereas the latter has only one. We can observe from our results that a more complex model does not necessary imply a better empirical performance.

The Decision Tree cannot be improved further marginally, even with pruning- removing irrelevant and redundant branches of the tree (Tofan, 2014). This is because the tree always favours the attributes on the top section of the tree. As a result, the attributes that are further down lose relevancy in the model. This poses a problem because several words usually have an equal importance in determining the geolocation of as tweet.

Multinomial Naive Bayes and Logistic Regression both have an equal accuracy score of 0.46 and Multinomial Naive Bayes has a higher Macro F1 score than Logistic Regression but the difference is not big enough to conclude that one is clearly better than the other. We then look at the underlying assumptions and logistics of each model. Due to the conditional independence assumption in the Multinomial Naive Bayes model, the classifier becomes oversensitive to redundant attributes, which are still present in the features data despite the χ^2 feature selection process. A high number of attributes randomises the Naive Bayes model (Patil, 2017), as discussed in the previous section. Further, the Laplace smoothing implemented in Multinomial Naive Bayes adds bias to the model, which explains the imbalance in the output. Even though the smoothing also reduces the variance, it does not matter if we have a high bias.

In contrast, the Logistic Regression model does not make any assumption on the features of the data. Instead, it optimises the conditional probabilities directly and hence optimally distinguishes between tweets in different regions. As a result, the model is more flexible and will adapt better to new instances and features coming in. Also, Logistic Regression has feature selection embedded in its algorithm- assigning low weights to insignificant attributes and high weights to significant ones.

5.2 Ethical Issues

In the case of a geolocation classifier, we are more concerned about the matter of privacy. The major question about privacy is whether the benefits stemming from a geolocation classifier overrule a person's right to privacy. The only ethical reason to record a person's geolocation is if they are potentially involved in unlawful acts. For example, if they commit a crime or if a child goes missing. In these less likely instances, the geolocation of the parties potentially involved would be extremely useful. However, it still raises the question: can breaching the privacy of millions of people be justified if there is the slight probability that it may be

useful in a small fraction of cases?

An issue more specific to the report is that the classifier is trained to create assumptions about individuals based on their vocabulary. This may cause issues within a political realm. For instance, the classifier identifies that users in a specific region use vocabulary that are more left or right leaning. This then creates a gateway for leaders and officials from that political party to target those in that region with propaganda or politically biased messages. Hence, there is a potential to manipulate users' political views. This issue is one that has now been brought to attention to many social-media users and has spurred conversations about the lack of ethics involved in user tracking (Pavalanathan & Eisenstein, 2015).

6 Conclusion

We propose the Logistic Regression model as our final model to classify the geolocation of tweets, using word counts as attributes. Even though the accuracy score is only at 46%, we believe that the model can do a better job for predictions with more training data. Further work can be done in terms on feature selection to improve the accuracy of the model, which is the main issue when dealing with Natural Language Processing classification (Eisenstein et al., 2010). The next step is to test the accuracy of this Logistic Regression classifier with the unlabelled test dataset on Kaggle.

7 Bibliography

Boehme-Neßler, V. (2016). Privacy: a matter of democracy. Why democracy needs privacy and data protection. International Data Privacy Law, 6(3), 222-229. https://doi.org/10.1093/idpl/ipw007

Eisenstein, J., O'Connor, B., Smith, N., & Xing, E. (2010). A latent variable model for geographic lexical variation. 2010 Conference On Empirical Methods In Natural Language Processing, 1277-1287.

HaCohen-Kerner, Y., Miller, D., & Yigal, Y. (2020). The influence of preprocessing on text classification using a bag-of-words representation. PLOS ONE, 15(5), e0232525. https://doi.org/10.1371/journal.pone.0232525

Li, Y. (2017). Text feature selection algorithm based on Chi-square rank correlation factorization. Journal Of Interdisciplinary Mathematics, 20(1), 153-160. https://doi.org/10.1080/09720502.2016.1259769

Patil, R. (2017). A Comparative Study of Centroid-Based and Naïve Bayes Classifiers for Document Categorization. International Journal Of Engineering Research And Applications, 07(03), 59-63. https://doi.org/10.9790/9622-0703045963

Pavalanathan, U., & Eisenstein, J. (2015). Confounds and consequences in geotagged twitter data. 2015 Conference On Empirical Methods In Natural Language Processing, 2138-2148.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. 2014 Conference On Empirical Methods In Natural Language Processing (EMNLP), 1532-1543.

Rahimi, A., Cohn, T., & Baldwin, T. (2018). Semi-supervised user geolocation via graph convolutional networks. 56Th Annual Meeting Of The Association For Computational Linguistics, 1, 2009-2019.

Schütze, H., Manning, C., & Raghavan, P. (2008). Introduction to information retrieval. Cambridge University Press Cambridge, 39.

Stoltzfus, J. (2011). Logistic Regression: A Brief Primer. Academic Emergency Medicine, 18(10), 1099-1104. https://doi.org/10.1111/j.1553-2712.2011.01185.x

Tofan, C. (2014). Optimization Techniques of Decision Making - Decision Tree. Advances In Social Sciences Research Journal, 1(5), 142-148. https://doi.org/10.14738/assrj.15.437

Toliušis, R., & Kurasova, O. (2017). Multilayer perceptron for face recognition. Lietuvos Matematikos Rinkinys, 58. https://doi.org/10.15388/lmr.b.2017.11

Tsai, D., & Lin, C. (2011). Fuzzy C-means based clustering for linearly and nonlinearly separable data. Pattern Recognition, 44(8), 1750-1760. https://doi.org/10.1016/j.patcog.2011.02.009