JudithJeyafreeda@Dravidian-CodeMix-FIRE2020: Sentiment Analysis of YouTube Comments for Dravidian Languages

Judith Jeyafreeda Andrew

GREYC, CNRS, Université de Caen, Normandie, France

Abstract

Sentiment Analysis is a process of identifying the tone of a given sentiment/opinion. It focuses on the polarity of the sentiments - positive, negative or neutral. In this paper, the goal is to classify sentiments from a dataset of comments/posts into pre-defined classes belonging to two code-mixed Dravidian Languages (code mixed Malayalam-English and code mixed Tamil-English) collected from social media. A new gold standard corpus for sentiment analysis for code-mixed Dravidian languages (Malayalam-English and Tamil-English) is used in this paper. The classification is performed as a multi-class classification problem. For this purpose, several well known machine learning models have been used. Based on the accuracy of each model obtained from the development set, the best model is chosen for prediction.

Keywords

Sentiment Analysis, Multi-class classification, Machine Learning, Dravidian Languages

1. Introduction

Sentiment Analysis aims at identifying the polarity of the text. The task in FIRE 2020([1], [2]) is to classify sentiments from the YouTube comments in the code-mixed Dravidian languages of Tamil and Malayalam. Thus for this purpose, a Multiclass classification approach is used. Multiclass text classification is a process of classifying an instance into one of the multiple classes possible. In a multi class classification problem, an instance can belong only to one class. There are several machine learning models for these type of classification problems. Some of the models are better suited for a certain problem than others. Thus the goal of this work is to find the best suited model for the particular task at hand. This technique allows for experimenting with several machine learning models to obtain better predictive performance. The task discussed in this paper uses datasets from two Dravidian languages - Tamil and Malayalam. The classification is a multi-class classification. In the sense that there are several classes to choose from to classify an instance. Several machine learning algorithms have been trained for this purpose. The accuracy of each model is then calculated from the development datasets and based on this measure, the model with the highest accuracy is chosen for prediction.

2. Related Work

[3] presents an improvement of word sense translation for under-resourced languages. It focuses on cleaning the noisy corpus in the form of code-mixed content at word-level based on orthographic

FIRE 2020: Forum for Information Retrieval Evaluation, December 16-20, 2020, Hyderabad, India

EMAIL: judith-jeyafreeda.andrew@unicaen.fr (J.J. Andrew)

ORCID: 0000-0002-2305-1439 (J.J. Andrew)

information which results in improvement of Dravidian languages. It also proposes to alleviate the problem of different scripts by transcribing the native script into a common representation such as the Latin script or the International Phonetic Alphabet (IPA).

[4] presents the two popular methods which decompose multi-class problems into many binary class problems, the “one-against-all” and the “one-against-one” approaches. The "one-against-all" approach is fairly simple. Suppose there are K classes, these K classes are partitioned into a two class problem - one for a "true" class and the other for the "others" class. Thus there are K two way classifiers, one for each class. In a "one-against-all" approach, a two way classifier is trained for all possible pairs of classes leading to $K(K\text{-}1)/2$ two way classifiers for a problem with K classes.

Random Forests are popular machine learning models for classification [5], [6]. A random forest is an ensemble of tree structure classifiers where every tree gives a unit vote assigning the most probable class label to the input [7]. [8] describes an improved random forest classification approach for improving the classification accuracy for a multi-class classification problem. Improved-RFC approach uses Random Forest algorithm, an attribute evaluator method and an instance filter method-Resample for the problem of multi-class classification problem.

A Naïve Bayes classifier is a simple probabilistic model that allows prediction of probablities of class membership ([9]). This classifier handles missing attribute values by omitting the corresponding probabilities for those attributes when calculating the likelihood of membership for each class. It also requires the class conditional independence. [10] uses a hidden naïve Bayes (HNB) classifier to a network intrusion detection system. The HNB is an extended version of the basic naïve Bayes classifier which relaxes the conditional independence assumption. The HNB creates a new layer which represents the hidden parent of each attribute. Thus the influences from all of the other attributes can combined through conditional probabilities by estimating the attributes from the training dataset.

[11] compares various machine learning algorithms - Naïve Bayes, Random Forest, Decision Tree, Logistic Regression and support vector machines for the classification of text reviews. The findings indicate that the Logistic Regression for multi-class classification for product reviews is the best method in terms of accuracy. It should also be noted that the overall classification accuracy in combination with uni/bi/tri-gram models increases the average of classification accuracy.

3. Data

The dataset used here is from the Dravidian-CodeMix - FIRE 2020, which is a collection of YouTube comments in two different languages - Tamil (data from [12]) and Malayalam (data from [13]). The datasets of both the languages (Tamil and Malayalam) are in a way that the text is in Latin script. The Tamil training dataset contains 11,335 instances, the validation/development dataset contains 1,260 instances and the test dataset contains 3,149 instances. The classes are "positive", "negative", "unknown_state", "mixed_feeling" and "not_Tamil". The Malayalam training dataset contains 4,851 instances, the validation/development dataset contains 674 instances and the test dataset contains 1,348 instances. The classes are "positive", "negative", "unknown_state", "mixed_feeling" and "not_Malayalam".

4. Text Representation

To be able to train a supervised classifier, every comment in the dataset is represented by a numerical feature vector. One common approach for extracting features from text is to use the bag of words model. In this model, the frequency of the words is taken into consideration, but the order in which they occur is ignored. The Term Frequency, Inverse Document Frequency (tf-idf) measure is calculated for each term in the dataset (individually for Tamil and Malayalam).

5. Machine Learning Models

In this section, several machine learning methods designed for the task at hand are explained. The models are Logistic Regression, Naïve Bayes, Support Vector Machines and Random Forests.

5.1. Logistic Regression

The well established multi-class logistic regression model is implemented for the task at hand [14]. The model of logistic regression for a multi-class classification problem forces the output layer to have discrete probability distributions over the possible k classes. This is accomplished by using the softmax function. Given the input vector(z), the softmax function works as follows:

$$softmax(z) = \frac{e^{z}}{\sum_{i=1}^{k} e^{z_i}} \tag{1}$$

At this point, there are k outputs and thus there is a necessity to impose weights connecting each input to each output. The model thus is as follows:

$$\hat{y} = softmax(xW + b) \tag{2}$$

where, W is the weight matrix between the input and output, x being the input and b is the bias.

5.2. Random Forest

Random Forest is a collection of large number of individual decision trees. Decision Trees for samples from the training data sets are constructed. Following this, each decision tree predicts a class. A vote is performed on all predicted results. The class with the maximum vote is decided on to be the output class. For the training process, the random subspace method is used (i.e) if one or a few features are very strong predictors for the target output, these features will be selected in many of the decision trees. This makes the features more correlated.

5.3. Support Vector Machines

SVMs are very good classification algorithm. The idea is to identify hyper-planes that will separate the various features. A linear SVM is used in this paper. The classification decision is thus performed as follows:

$$f(x) = sign(W^{*}.x + b^{*}) \tag{3}$$

where x represents the input feature, W represents the model weight and b represents the bias. For the multi-class classification problem, a one-vs-rest (also known as one-vs-all) approach is used. It involves splitting the dataset into multiple binary classification problems. Thus a binary classification boundary is constructed to train each binary SVMs and the one with the highest confidence is used to solve the multi-class classification problem.

Model	Parameters
SVM	multi_class=ovr; max_iter=1000; penalty=l2; loss=squared_hinge
Logistic Regression	solver=liblinear; penalty=l2; tol=0.0001; max_iter=100
Naïve Bayes	alpha=1.0; fit_prior=True; class_prior=None
Random Forest Classifier	n_estimators=200; max_depth=3; random_state=0

Table 1
Implementation details for the various machine learning models

Model	Accuracy	Language
SVM	0.64	Tamil
Logistic Regression	0.65	Tamil
Naïve Bayes	0.69	Tamil
Random Forest Classifier	0.67	Tamil
SVM	0.60	Malayalam
Logistic Regression	0.64	Malayalam
Naïve Bayes	0.55	Malayalam
Random Forest Classifier	0.43	Malayalam

Table 2
Accuracy of the different models.

5.4. Naïve Bayes

Naïve Bayes [15] is based on the Bayes theorem. For a given training dataset, the joint probability distribution (P(X,Y)) is learned. When using Naïve Bayes for classification for an input x, the posterior probability is calculated by the classification model. The class with the highest posterior probability is the predicted class.

6. Implementation

The scikit-learn[1] package in Python is used for the feature extraction and model training. The TfidfVectorizer from the scikit-learn is used to convert the text data into TF-IDF feature vectors. The models of logistic regression, linear support vector classification, multinomialNB and random Forest provided by the scikit-learn toolkit are used for the training of the machine learning models discussed in 5. The implementation details for these models are shown in table 1.

Every model described in section 5 is trained using the training sets for both the Tamil and Malayalam languages. The accuracy of each model for the two languages are calculated. The accuracy of the model is calculated using the development set. The accuracy of each model for the different languages are presented in table 2.

As seen from table 2, all models are quite closer to each other in terms of accuracy. However, the highest accurate model to use for the classification of YouTube comments in the Tamil language is the Naïve Bayes classifier. Thus this classifier is used for the classification of the test data set. As for the Malayalam language, table 2, the Logistic Regression is the most accurate model which is thus used for the classification of YouTube comments in the Malayalam language.

[1] https://scikit-learn.org/

Language	Precision	Recall	F-score	Algorithm
Tamil	0.57	0.66	0.54	NB classifier
Malayalam	0.68	0.62	0.58	Logistic Regression

Table 3
Results.

Language	Positive	Negative	Mixed_Feelings	unknown_state	not_Tamil/Malayalam
Tamil	7,627	1,448	1,283	6,09	368
Malayalam	2,022	549	289	1,344	647

Table 4
Number of instances in each category from the training set of the two different languages.

7. Results and Conclusion

The weighted averages for the precision, recall and F-score for the task at hand is shown in table 3. A precision of 0.57, a recall of 0.66 and a F-score of 0.54 is achieved by the method presented in this paper for the Tamil language. A precision of 0.68, a recall of 0.62 and a F-score of 0.58 is achieved by the method presented in this paper for the Malayalam language. The model for the Malayalam language achieves a better performance than the model for the Tamil language. However, it has to be noted that the same model has not been chosen for both the languages and thus comparing the results between languages is not fair. It is seen from table 2 that the accuracy of the Naïve Bayes, which has been chosen as the model for the Tamil language, has a very low accuracy for the Malayalam language. On the other hand, the Linear Regression, which has been chosen as the model for the Malayalam language, has a low accuracy for the Tamil language. This reassures the fact that the same model does not suit very well for all the languages.

The higher accuracy of the various models for the Tamil language could be because of the fact that the training and development data for the Tamil language is larger than that for the Malayalam language. This could mean that the models are very well trained for the Tamil language when compared to the Malayalam language. The number of instances for each case from the training set for both languages are shown in table 4. It can also be noted from table 4 that the dataset for Tamil is quite imbalanced with too many instances for the "Positive" category but not too many instances for the other categories. For the Malayalam training dataset, the number of instances for the "Positive" and "unknown_state" category are greater than the number of instances for the other categories. The imbalance in the dataset for the various categories makes it harder for predicting the minor classes (categories with fewer instances).

Thus in this paper, multiple models are investigated for the languages - Tamil and Malayalam. It is noticed that two different models are chosen for the two languages. The models presented in this paper are language independent. All of these models can be applied to any language as the model with the best accuracy is chosen as final model for the classification of the sentences. Link to code [2].

[2] https://github.com/JudithJeyafreeda/FIRE2020.git

References

[1] B. R. Chakravarthi, R. Priyadharshini, V. Muralidaran, S. Suryawanshi, N. Jose, E. Sherly, J. P. McCrae, Overview of the track on Sentiment Analysis for Dravidian Languages in Code-Mixed Text, in: Proceedings of the 12th Forum for Information Retrieval Evaluation, FIRE '20, 2020.

[2] B. R. Chakravarthi, R. Priyadharshini, V. Muralidaran, S. Suryawanshi, N. Jose, E. Sherly, J. P. McCrae, Overview of the track on Sentiment Analysis for Dravidian Languages in Code-Mixed Text, in: Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2020). CEUR Workshop Proceedings. In: CEUR-WS. org, Hyderabad, India, 2020.

[3] B. R. Chakravarthi, Leveraging orthographic information to improve machine translation of under-resourced languages, Ph.D. thesis, NUI Galway, 2020.

[4] M. P. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, M. Ares, D. Haussler, Knowledge-based analysis of microarray gene expression data by using support vector machines, Proceedings of the National Academy of Sciences 97 (2000) 262–267.

[5] D. R. Cutler, T. C. Edwards Jr, K. H. Beard, A. Cutler, K. T. Hess, J. Gibson, J. J. Lawler, Random forests for classification in ecology, Ecology 88 (2007) 2783–2792.

[6] B. Ghimire, J. Rogan, J. Miller, Contextual land-cover classification: incorporating spatial dependence in land-cover classification models using random forests and the getis statistic, Remote Sensing Letters 1 (2010) 45–54.

[7] L. Breiman, Random forests, Machine learning 45 (2001) 5–32.

[8] A. Chaudhary, S. Kolhe, R. Kamal, An improved random forest classifier for multi-class classification, Information Processing in Agriculture 3 (2016) 215–222.

[9] J. Chen, H. Huang, S. Tian, Y. Qu, Feature selection for text classification with naïve bayes, Expert Systems with Applications 36 (2009) 5432–5435.

[10] L. Koc, T. A. Mazzuchi, S. Sarkani, A network intrusion detection system based on a hidden naïve bayes multiclass classifier, Expert Systems with Applications 39 (2012) 13492–13500.

[11] T. Pranckevičius, V. Marcinkevičius, Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification, Baltic Journal of Modern Computing 5 (2017) 221.

[12] B. R. Chakravarthi, V. Muralidaran, R. Priyadharshini, J. P. McCrae, Corpus creation for sentiment analysis in code-mixed Tamil-English text, in: Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL), European Language Resources association, Marseille, France, 2020, pp. 202–210. URL: https://www.aclweb.org/anthology/2020.sltu-1.28.

[13] B. R. Chakravarthi, N. Jose, S. Suryawanshi, E. Sherly, J. P. McCrae, A sentiment analysis dataset for code-mixed Malayalam-English, in: Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL), European Language Resources association, Marseille, France, 2020, pp. 177–184. URL: https://www.aclweb.org/anthology/2020.sltu-1.25.

[14] Multiclass logistic regression from scratch, 2017. URL: https://gluon.mxnet.io/chapter02_supervised-learning/softmax-regression-scratch.html.

[15] A. Y. Ng, M. I. Jordan, On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes, in: Advances in neural information processing systems, 2002, pp. 841–848.