Classification of French dialect of Canada using WEKA

Anonymous ACL submission

Abstract

This paper describes the experiments carried out in order to build a classifier that can classify the French dialect of a particular country in a mixed corpus of French dialects. We have used the different machine learning classifiers offered by WEKA in order to come up with one classifier and its best feature-sets that can perform this task with greater accuracy. The paper also demonstrates how the different phases of **CRISP-DM** methodology has been applied to accomplish this data mining task.

1 Introduction:

French is spoken in 29 different countries. The dialect of French just like most popular languages changes from place to place. For this piece of research, we have selected French dialects from 6 different countries and have built a machine learning classifier that can distinguish one dialect from the rest.

2 Business Understanding:

This data mining problem involves building a classifier that could correctly identify with considerable accuracy and thereby classify the French dialect of Canada in a mixed corpus of French dialects. This involves collecting data from websites using SketchEngine's WebBootCat features, cleaning the data to get rid of unwanted elements, classifying the data for training and testing of the algorithms, processing it to be able to load into the mining tool and then experimenting with different machine learning algorithms, features and parameters in order to come up with a suitable classifier with high performance accuracy for this task.

3 Data Understanding:

The raw data was collected from SketchEngine (Alfaifi and Atwell, 2016; Kilgarriff, A., Baisa, V., Bušta, J, 2014) and produced in .txt file format. Four seed words were used in the process which are **Tourisme**, **Voyage**, **Magasinage** and **Facture** of which the latter two, are unique to the dialect of Canada. This French corpus is collection of 70142 words of the TLD of Canada. The rest of the team collected corpuses of French dialects from Ivory Coast, Morocco, Belgium, Switzerland and France. More details on the same are present in the report submitted in CW0.

Below are the main data quality issues identified in the collected corpuses –

- The data collected in the corpuses, were unstructured and not of the format which would allow loading into WEKA.
- The presence of numerous HTML tags like , , <doc> and </doc>
- The presence of several URLs in the collected text.

A corpus comparison exercise in SketchEngine of these 6 corpuses yields the below result where a score of 1 pertains to a complete match between the comparing corpuses and the scores get higher, as the difference between the corpuses increases.

	CH	BE	CA	FR	CI	MA
FrenchCH	1.00	4.40	5.06	5.59	4.01	4.48
FrenchBE	4.40	1.00	3.64	3.61	4.06	4.11
FrenchCA	5.06	3.64	1.00	4.15	4.75	4.64
FrenchFR	5.59	3.61	4.15	1.00	5.40	5.45
FrenchCI	4.01	4.06	4.75	5.40	1.00	3.72
FrenchMA	4.48	4.11	4.64	5.45	3.72	1.00

Fig 1.1

4 Data Preparation:

In order to deal with the data quality issues, python programing was used. The in the raw data was replaced with a single quote and was replaced with a colon followed by a single quote, as the ARFF file format needs the instances to be presented in this format for WEKA to be able to load them. The actual text body which is of linguistic importance also contained single quotes in several places. These were preceded by a skip character i.e. a backslash in order to maintain their linguistic relevance. The web URLs and the HTML tags- <doc> and </doc> where removed as well using python. The instances in the corpus of French dialect of Canada were given the class value 1 whereas, those that of all other dialects were given 0. The datasets were then appended and then segregated into test and training set with 80% and 20% data in them respectively.

These two datasets where then converted into WEKA - ARFF file format which contains a header section having the description of the attributes and the labels used for class. The rest of the file is data in the format –Text, Class Label. Mentioned below are the before and after picture of the data cleaning respectively.

```
<doc url="https://ici.radio-canada.ca/tele/la-facture/2015-2016/segments/reportage/4674/
assurance-voyage-complementaire-touriste-sud-snowbird" parent_folder="web1"
id="file19370856" filename="assurance-voyage-complementaire-touriste-sud-snowbird">
<p> Nous utilisons les témoins de navigation (cookies) afin d'opérer et d'améliorer nos
services ainsi qu'à des fins publicitaires. Le respect de votre vie privée est important
```

Before-

```
@relation 'French Canada Train - Novices'

@attribute Text string
@attribute class-att {0,1}

@data
' Nous utilisons les témoins de navigation (cookies) afin d\'opérer et d'améliorer nos
services ainsi qu\'à des fins publicitaires. Le respect de votre vie privée est important
pour nous. Si vous n\'êtes pas à l\'aise avec l\'utilisation de ces informations,
veuillez revoir vos paramètres avant de poursuivre votre visite.Gérer vos témoins de
navigationEn savoir plus ;',1
```

After-

5 Modelling:

In order to meet the objective of this task, first a baselining was done to identify potential classifiers from a selection of classifiers, which could fulfil this task with greater accuracy. J48, JRIP, Naïve Bayes, SMO and ZeroR provided by WEKA (Hall et al., 2009) were selected to carry out this experiment. The performance accuracy of these classifiers were captured in four different test methods namely : Use Training Set – where the same data used to train the classifier is used to test it, 10 Fold Cross Validation – in which the classifier is run 10 times with 90 percent of the data from the training set each time and setting aside the remaining 10 percent for prediction at the end, Percentage Split – where 66% of the supplied training data set is used for training and 34% for testing and the lastly the Supplied Test Set where a separate dataset is supplied for the classifiers to make predictions upon. The below table – 1.1 shows the performance of the classifiers across these four test strategies. WEKA StringToWordVector filter with WordTokenizer was used to extract words as features from the input strings. It converts the words present in an input text/string into a set of attributes which represents the occurrence of the words in a dataset (Alshutayri et al., 2016). For each class, n top words are kept. This was set to 1000 words per class.

Classifier Name	Test Strategy			
	Use Training Set	10 Fold Cross Val	Percentage Split	Supplied Test Set
SMO	95.61	91.94	91.59	86.52
J48	93.39	88.67	88.00	83.36
JRIP	85.34	84.65	84.12	80.54
Naïve Bayes	83.50	83.29	84.00	74.77
ZeroR	78.16	78.16	78.31	75.36

Table 1.1

From the results of the first experiment in table 1.1, it could be observed that the SMO and J48 were the two best performing classifiers which were considered for further experimentation.

To further improve the performance accuracies of these two classifiers, term weighting schemes – **Term Frequency (TF)** and **Inverse Document Frequency (IDF)** were used.

TF – TF or term frequency refers to the number of times a particular term appears in a text. This is used with the intuition that a term which occurs more frequently in a text, identifies the particular text better than any other term that occurs less frequently (Gebre et al., 2013).

IDF – IDF or the inverse document frequency, quantifies the intuition that a term which occurs

frequently in several texts is not a good identifier for the texts and should be given less weights (Gebre et al., 2013).

With the parameters, **TFTransform, IDFTransform** set to true and **lowercaseTokens** (converts word tokens to lower case before adding to the dictionary) set to true, the performance of these two classifiers were tested. The results for the same are captured in table 1.2 below.

Classifier Name	Test Strategy			
	Use Training Set	Percentage Split	10 Fold Cross Val	Supplied Test Set
SMO	96.5081	91.8705	92.331	85.8045
J48	94.082	88.787	88.683	83.5299

Table 1.2

It can be observed from the table above that the changes in the parameter settings stated previously improved the accuracy of the classifiers by a negligible proportion which is < 1%. However, the ranking order of these two classifiers in terms of accuracy remained the same.

In order to experiment further for performance improvement, the parameter **IDFTransform** was set to false which was earlier set to true. This was done with the intuition that the corpuses already differed from each other to a great extent as seen the figure - Fig 1.0 and had less terms in common. Also, the word frequency of the instances was normalized by setting **normalizeDocLength** to Normalize all data. With these settings the classifier's performances were evaluated again and results are presented in the below table.

Classifier Name	Test Strategy			
	Use Training Set	Percentage Split	10 Fold Cross Val	Supplied Test Set
SMO	97.7081	93.0173	93.6317	87.1946
J48	94.1946	88.6263	89.0038	84.3724

Table 1.3

It can be observed from the table above that there is a slight improvement in the accuracy of J48 but in case of SMO the improvement was > 1% Also the performance accuracy, of the SMO classifier turned out to be significantly better than J48.

We further carried out another experiment in Supplied Test Set mode with the SMO classifier by carrying out step wise feature reduction. With the parameter values – **IDFTransform** = False, **TFTransform** = True, **lowerCaseTokens** = True, **normalizeDocLength** = Normalize all data, we gradually increased the number of top n words per class starting with 1000 words and increasing by 1000 words for every run. The figure below shows the performance accuracies obtained at the different feature thresholds.

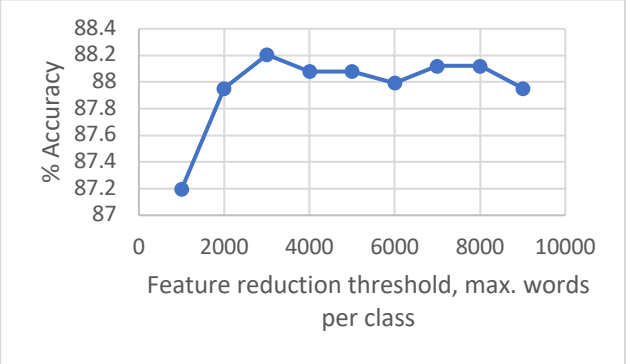

Fig 1.2

It can be seen that initially the accuracy increases as the number of feature increases. It reaches the maximum of 88.2 % performance accuracy at 3000 top features per class, then flattens out at 4000 to 5000 features, dips a little at 6000 features, increases slightly between 6000 to 7000, flattens out again between 7000 to 8000 and shows a downward trend from then onwards.

6 Evaluation:

It can be seen from the experiments carried out above, that the SMO classifier has the best performance accuracy in this task.

The Sequential Minimal Optimization (SMO) is WEKA's implementation of the Support Vector Machine classifier (SVM) (Tarmom et al., 2020). SVM is a learning approach introduced in 1995 by Vapnik in order to resolve two class pattern recognition problem. The method is defined over a vector space where the goal is to find a decision surface that separates the data into two classes in the best possible way (Ayodele, 2010). SVM can achieve the best performance in text classification due to its ability to remove the need for feature selection. (Alshutayri et al., 2016)

Mentioned below are the results of SMO classifier from all the four test methods with 3000 top features per class and with the parameter values – **IDFTransform** = False, **TFTransform** = True, **lowerCaseTokens** = True,

normalizeDocLength = Normalize all data, **wordsToKeep** = 3000. This feature-set with SMO, gives the best performance accuracy for this task. For all other parameter settings, the performance accuracy is lower.

Training set- % accuracy = 98.7696 %

Confusion Matrix-

a	b	Classified As
8989	32	a = 0
110	2410	b = 1

Table 2.1

Detailed accuracy by class –

Class	TP Rate	FP Rate	Precision	Recall
0	0.996	0.044	0.988	0.996
1	0.956	0.004	0.987	0.956

Table 2.2

Percentage Split- % accuracy = 93.9348 %

Confusion Matrix-

a	b	Classified As
3008	65	a = 0
173	678	b = 1

Table 2.3

Detailed accuracy by class –

Class	TP Rate	FP Rate	Precision	Recall
0	0.979	0.203	0.946	0.979
1	0.797	0.021	0.913	0.797

Table 2.4

10 Fold Cross Validation- % accuracy = 94.3073 %

Confusion Matrix-

a	b	Classified As
8792	229	a = 0
428	2092	b = 1

Table 2.5

Detailed accuracy by class –

Class	TP Rate	FP Rate	Precision	Recall
0	0.975	0.17	0.954	0.975
1	0.83	0.025	0.901	0.83

Table 2.6

Supplied test set- % accuracy = 88.2056 %

Confusion Matrix-

a	b	Classified As
1748	41	a = 0
239	346	b = 1

Table 2.7

Detailed accuracy by class –

Class	TP Rate	FP Rate	Precision	Recall
0	0.977	0.409	0.88	0.977
1	0.591	0.023	0.894	0.591

Table 2.8

The substantial increase in the performance accuracy of this classifier as seen in fig 1.2 suggests that applying feature reduction prior to learning is beneficial. This increase may be related to the removal of noisy features and additional information presented to the classifier due to feature reduction (Danso et al., 2013).

7 Conclusion:

With the help of the experiments conducted above, we have been able to build a classifier which could perform this task with a high level of accuracy. The improvement in performance accuracy of the **SMO classifier** at the beginning and after these series of experiments can be seen in the table below.

	Before	After	% Improvement
Use Training Set	95.61	98.7696	3.16
Percentage Split	91.59	93.9348	2.34
10 Fold Cross Val	91.94	94.3073	2.37
Supplied Test Set	86.52	88.2056	1.69

Table 3.1

An average of 2.39% improvement has been achieved across all test strategies.

Reference:

A. Alfaifi and E. Atwell. 2016. Comparative evaluation of tools for Arabic corpora search and analysis. *International Journal of Speech Technology.* 19(2), pp. 347-357

A. Alshutayri, E. Atwell, A. Alosaimy, J. Dickins, M. Ingleby, and J. Watson. 2016. Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts. In *Proceedings of VarDial'2016 Workshop on NLP for Similar Languages, Varieties and Dialects*, pp. 204-211

Ayodele, Taiwo. (2010). *Types of Machine Learning Algorithms*. 10.5772/9385.

Binyam Gebrekidan Gebre, Marcos Zampieri, Peter Wittenburg, and Tom Heskes. 2013. Improving native lan- guage identification with tf-idfweighting. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 216–223. Association for Computational Linguistics.

Danso, SO, Atwell, ES and Johnson, O (2013). *A comparative study of machine learning methods for verbal autopsy text classification*. IJCSI International Journal of Computer Science Issues, 10 (6). ISSN 1694-0784

Hall, Mark & Frank, Eibe & Holmes, Geoffrey & Pfahringer, Bernhard & Reutemann, Peter & Witten, Ian. (2008). *The WEKA data mining software: An update*. SIGKDD Explor. Newsl.. 11. 10-18.

Kilgarriff, A., Baisa, V., Bušta, J. et al. The Sketch Engine: ten years on. Lexicography ASIALEX 1, 7–36 (2014). https://doi.org/10.1007/s40607-014-0009-9

T. Tarmom, W. Teahan, E. Atwell, and M.A. Alsalka. 2020. Compression versus traditional machine learning classifiers to detect code-switching in varieties and dialects: Arabic as a case study. *Natural Language Engineering Journal.*