Data Mining – French Dialect

By Shan Devraj, Febin Shahji, James Lawton & Robert Dennis
(ed19sehd, ed192fs, ed18j5l, sc20rmd) – Team Name: X4

Abstract

This paper presents the results of a computational analysis of French dialects from around the world. We present the analysis of a wide range of French corpora using a range of techniques such as utilizing the Waikato Environment for Knowledge Analysis (WEKA) data analytic tool.

1 Introduction – Business Understanding

This paper is part of a joint University of Leeds research project to create and analyse a data set for text and specifically dialect analytics for the CND Corpus of National Dialects. Dialect analytics can be defined as the task of differentiating and examining the variability between written texts of the same language from differing regions (Alshutayri, A. Atwell E; Alosaimy A; Dickins J; Ingleby M; Watson J. 2016. Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts). The French language can be argued to be the most popular Romance language across the world and belongs to the Indo-European language family which has been derived and developed from Vulgar Latin. As the second-most spoken language and the official language in 29 countries, the French language has a myriad of dialects and variations. The common standard of written and spoken French is commonly associated with Parisian French due to Paris' status as the capital and its political and cultural importance in the modern French world. This paper aims to use corpus-based research, lexical and token analysis and sentiment analysis using tools such as SketchEngine, WebBootCat and WEKA to compare and identify dialect features specific to the French language from regions across Europe, North America and Africa.

2 Data Understanding

Data Collection Process: Data collection refers to extracting knowledge from a large amount of text. In the case of this research paper, our task was to see how different countries have different dialects for the same language. For our report we chose French and wanted to analyse the different dialects from Belgium, France, Canada, and Luxembourg. In order to collect data on the different dialects we made use of Sketch Engine where each individual in the group formed their corpus on their respective dialect. We did this by first creating a new corpus and then assigning the chosen language to French, we then chose to find suitable texts from the web. To control what texts Sketch Engine found, we each entered keywords in French linking to "news", "politics", "politicians", and "headlines" to ensure our corpuses were about the same topic to allow for to accurately compare our results. To guarantee that each of our corpuses contained each of our French dialects from our countries we specified the top-level domain (.ca, .be, .fr, and .lu) in the web search settings. We then repeated this process as many times as needed until our corpus was of a size from 50,000 to 70,000.

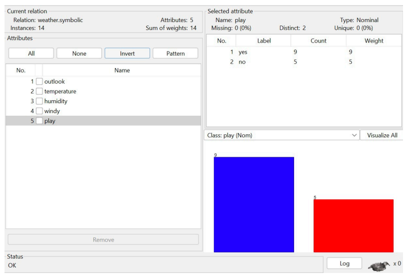
Data Quality Issues: Some factors that we recognised would affect the quality of the data we obtained for our corpus was the inclusion of dates and statistics, these would not be relevant in this task where we were trying to look at the difference between dialects of the same language. It would mean that our corpus would be full of data that is meaningless to our project thus lowering the quality of the data. We overcame this by choosing topics that we thought were less likely to come up with search results full of maths or statistics so we could better analyse that language. Another issue we came across was that not all sources that we mined from the web provided high quality data, some websites were providing data that were not good enough and this drastically lowered the quality. We overcame this by vetting the websites we used trying to make sure that the sites we extracted data from were of a higher quality. Another limitation with our data is that we could only enter a limited number of words and phrases before the number of words in the corpus exceeded the 50,000 to 70,000 range. This means our data will lose quality due to only covering as specific aspect of the dialects.

3 Data Preparation

Now that we all created our individual corpora for our respective dialects of French (Canada, Luxembourg, Belgium, and France), we were able to move onto the next step of the process and analyse these dialects. In order to analyse the dialects, we needed to be able to insert a .arff file into Weka. We can not just upload the individual corpora into Weka but first need to be able to combine all the corpora into a single file so we can analyse the whole thing altogether. Another problem that the current corpora have, is that they contain elements that we do not want it to contain. For example, if we take a look at the corpus, we will see that it contains html tags such as <p> or </p> as well as blank lines, new lines etc. These elements are not needed and would adversely affect the results we receive when running it through Weka. In order to combat this, we created a python script that did 3 main things. The first thing that the python script did was to create a .arff file and then assign a relation which would be the chosen language (French), and 2 attributes which would be the text and the different dialects. After that we needed to combine all of the corpora for the different dialects together. The final step is to open the new .arff file and then to remove all the elements we don't need such as the html tags, blank lines, new lines etc. After running the python script, it meant that we had a .arff file that we could then use inside Weka and apply the different classifiers and filters.

4 Modelling

For the purpose of evaluating the French dialect, we used 2 classifiers in WEKA, namely the Naïve Bayes Multinomial Text Classifier and the ZeroR classifier. We also used features such as StringToWordVector also found in WEKA. Below we will explain how both of these classifiers work as well as give example outputs with an example data set provided with WEKA (weather.nomial.arff). Below is a screenshot when the example data set is loaded into WEKA for pre-processing.



Current relation
Relation: weather.symbolic   Attributes: 5
Instances: 14   Sum of weights: 14

Selected attribute
Name: play   Type: Nominal
Missing: 0 (0%)   Distinct: 2   Unique: 0 (0%)

No.	Label	Count	Weight
1	yes	9	9
2	no	5	5

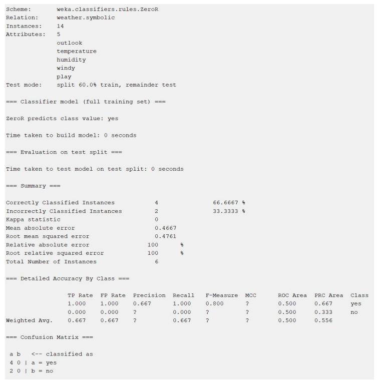
Attributes

No.		Name
1	☐	outlook
2	☐	temperature
3	☐	humidity
4	☐	windy
5	☐	play

Class: play (Nom)

Status
OK

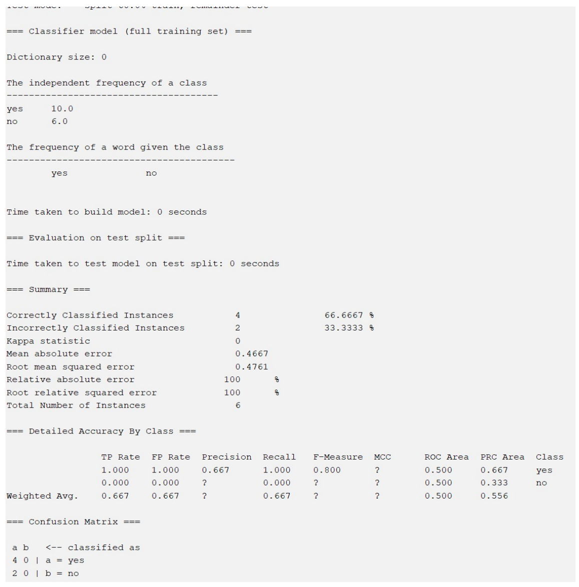
ZeroR:
ZeroR is the simplest classification method and relies on the target whilst disregarding predictors. It does not base its decision on any particular attributes and works by taking the majority class of the target attribute. Despite its relative simplicity, this classifier can produce surprisingly high levels of accuracy, but this might not be a good thing.
Below is a screenshot showing the ZeroR classifier used on the example data set mentioned previously highlighting its accuracy and performance.



```
Scheme:       weka.classifiers.rules.ZeroR
Relation:     weather.symbolic
Instances:    14
Attributes:   5
              outlook
              temperature
              humidity
              windy
              play
Test mode:    split 60.0% train, remainder test

=== Classifier model (full training set) ===

ZeroR predicts class value: yes

Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances           4               66.6667 %
Incorrectly Classified Instances         2               33.3333 %
Kappa statistic                          0
Mean absolute error                      0.4667
Root mean squared error                  0.4761
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances                6

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.667      1.000    0.800      ?        0.500     0.667     yes
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.333     no
Weighted Avg.    0.667    0.667    ?          0.667    ?          ?        0.500     0.556

=== Confusion Matrix ===

 a b   <-- classified as
 4 0 | a = yes
 2 0 | b = no
```

Naïve Bayes Multinomial Text:
Naïve Bayes Multinomial Text is specifically designed for text and has faster performance than the standard Naïve Bayes and is ideal for small sample sizes. This classifier is often used for document classification and uses the frequency of words for predictors. (Alshutayri, A. Atwell E; Alosaimy A; Dickins J; Ingleby M; Watson J.

2016. Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts). Below is a screenshot showing the Naïve Bayes Multinomial Text classifier used on the example data set mentioned previously highlighting its accuracy and performance.



```
=== Classifier model (full training set) ===

Dictionary size: 0

The independent frequency of a class
--------------------------------------
yes     10.0
no      6.0

The frequency of a word given the class
-----------------------------------------
        yes              no


Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances           4               66.6667 %
Incorrectly Classified Instances         2               33.3333 %
Kappa statistic                          0
Mean absolute error                      0.4667
Root mean squared error                  0.4761
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances                6

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.667      1.000    0.800      ?        0.500     0.667     yes
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.333     no
Weighted Avg.    0.667    0.667    ?          0.667    ?          ?        0.500     0.556

=== Confusion Matrix ===

 a b   <-- classified as
 4 0 | a = yes
 2 0 | b = no
```

StringToWordVector:
The StringToWordVector in WEKA is a filter which converts string attributes to sets of numeric attributes which are used to represent word occurrence. Specifically, this feature filters strings into N-grams and it can also remove stop-words. (Alshutayri, A. Atwell E; Alosaimy A; Dickins J; Ingleby M; Watson J. 2016. Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts). Below is a screenshot which shows how the filter works in a more intuitive manner:

```
@relation dataset
 @attribute Text string
 @date
 'I'm a movie lover and this is one of the best museums in which ...
```

```
@relation dataset1
@attribute word numeric
...
{13 2, 19 2, 30 2, 33 1, 53 1, 55 4, 60 1, 61 2, 72 3, 78 1, 89 1, 90 1, 99
1, 106 1,120 1,121 1,123 2,124 5,126 2,136 1,140 1,147 5,148 2,160 1,186
1,198 1,202 1,248 9,253 1, ...}
```

(https://stackoverflow.com/questions/55697318/stringtowordvector-weka-output)

5 Evaluation

For us to evaluate the French dialects across different native speaking countries we combined all corpora from each group member and transferred it into one .arff file. We applied the StringToWordVector to the ZeroR classifier and applied the Naive Bayes Multinomial Text classifier to the raw .arff file. When applying the StringToWordVector the only value changed from default was the number of words to keep, this was increased to 5000 per class.

Classifier	Correctly Classified Instances	Incorrectly Classified Instances	Weighted Average Precision	Mean Absolute Error
Naive Bayes Multinomial Text (2 fold)	43%	57%	0.708	0.2843
Naive Bayes Multinomial Text (5 fold)	71%	29%	0.829	0.1489
Naive Bayes Multinomial Text (10 fold)	74%	26%	0.868	0.1306
Naive Bayes Multinomial Text (20 fold)	74%	26%	0.848	0.1287
Naive Bayes Multinomial Text (25 fold)	75%	25%	0.875	0.1228
Naïve Bayes Multinomial Text (30 fold)	73%	27%	0.859	0.1336
ZeroR	36%	64%	N/A	0.3625

The best classifier we found was the Naive Bayes Multinomial Text with 25 folds. It had the highest percentage of correctly classified instances at 75%, along with the highest weighted average precision of 0.875 and lowest mean absolute error of 0.1228, making it the overall best, well rounded classifier. It's high average precision and low absolute error made it the most accurate. We found that an increase in folds would lead to an increase in accuracy and correctly classified instances up to 25 folds. This can be seen as going from 2 folds to 5 folds increases correct classification by 28%, while also improving weighted average precision and mean absolute error. This occurs as increasing folds increases the amount the data set is divided in to and the increases the number of tests used for training. After 25 folds the percentage of correctly classified instances begins to decrease along with accuracy as weighted average precision decreases and mean absolute error increases.

Out of all the classifiers we tested, we found that ZeroR was the worst classifier with the highest percentage of incorrectly classified instances and highest mean absolute error, meaning that it has very low accuracy. ZeroR predicts the class based off which class has the highest number of instances, which was Canada in this case. This is likely a consequence of Sketch Engine struggling to find websites in French under the ".ca" domain with large amounts of suitable text. ZeroR doesn't have a weighted average precision as it only tests one class, Canada in this case.

References:

Alshutayri, A. Atwell E; Alosaimy A; Dickins J; Ingleby M; Watson J. 2016. Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts. In proceedings of VarDial'2016 Workshop on Natural Language Processing for Similar Languages, Varieties and Dialects.
http://aclweb.org/anthology/W16-4826