Report of Data Mining CW3

Group: I'm Not a Thief but a Treasure Hunter

Team members: Leshan Wang 201388927 Yujia Lei 201388911 Jiayuan Li 201388919 Yixuan Liutang 201388954 Xinyu Li 201388943

Abstract

This document is the formal report of data mining coursework 3, Spanish dialect classification task, in which the task, classifiers and evaluation will be discussed in detail. The project is conducted by a group of 5 students from SWJTU-Leeds Joint School.

1 Business Understanding

With the ever-increasing amount of data in life, a variety of storage devices can make it easy to postpone decisions about what to do with all. Based on this situation, the importance of data mining and machine learning becomes more and more obvious (Frank et al., 2017).

The goal of this team project is to understand a certain language, here we have chosen Spanish. Samples of language variants or dialects of Chile, Spain, Argentina, Colombia, and the United Kingdom were collected and analyzed. The goal is to use data mining methods, collect corpora, and apply algorithms to build an optimal classifier for Spanish dialects from different countries.

Data mining refers to the automated process of sorting through massive data sets to identify trends and patterns through data analysis and build relationships to solve business problems. There are two layers of definitions, one is the technical definition, that is, from a large amount of incomplete, fuzzy, random practical application data, extract the hidden, people do not know in advance, but is a process of potentially useful information and knowledge. The second time is the definition of business perspective. It is also a new business information processing technology. Its main feature is to extract, transform, analyze, and model a large amount of business data in the business database, and extract and assist business decision-making key data.

2 Data Understanding

According to the number of first language users, about 437 million people use Spanish as their mother tongue, making it the second largest language in the world after Chinese. Mainly used in Spain and Latin America. Spanish is known as Spanish (Español) in Spain, the United States, Mexico, Central America, the Caribbean, Colombia, Ecuador, and Uruguay; in other regions it is primarily known as Spanish (Castellano).

The formation of Spanish is closely related to the development and evolution of Spanish history. Spanish has widely accepted the influence of other dialects (Catalan, Galician, Basque). Since the 18th century, English has emerged as a force, almost all major languages and has become a global vocabulary. Spanish of course also absorbs a lot of English vocabulary, some of these words have been Sanitized, and some of these words have been transplanted intact. Latin American Spanish is an important part of the development and evolution of Spanish, as it is the product of the fusion of Castilian languages from the Iberian Peninsula with the cultural, natural, and social environment of the Americas. This fusion is mainly reflected in the aspects of vocabulary, phonetics, and semantics.

However, Latin American countries have their own characteristics of Spanish due to differences in cultural and social development. These are all things to consider when collecting a corpus of Spanish dialect texts. The United Kingdom can serve as the representative of English forces, and Colombia, Chile, and Argentina can serve as representatives of Latin American. Therefore, we choose to collect Spanish dialect samples from five countries: Spain, Britain, Colombia, Chile, and Argentina for analysis.

First, we used Sketch Engine to organize corpora from five countries. The set keywords are es (is), hacer (do), ciudad (town), culture (culture), música (music), amor (love), es and hacer are common prepositions and verbs respectively, used to expand the search Scope, the remaining four nouns are related to life and culture, and are used to be close to the different cultural aspects of each country. Then set search scope as url: 7, Seed: 3, which is used to control the size of the corpus to be about 50,000 to 70,000 words. Finally set TLD for each country and start creating corpus.

After constructing the corpus, we compared the data, analyzed the reliability of the corpus content, and set the N-grams length to 2 to 3 and 3 to 4. Observe the top 30 words by frequency, then observe the lemma and noun of the Word List, and finally set the focus on to 0.001, 10 and 100000 respectively, and observe the SINGLE-WORDS and MULTI-WORD TERMS of the Keywords. By comparing the observed data, we obtained important features of these corpus which ensure they won't cause potential overfitting errors. Therefore, by doing this, the result of classification accuracy might not be high but is guaranteed to be reasonable.

3 Data Preparation

Before doing any classifications, the raw data needs to be processed and transformed into arff files. First, delete " " ", " ' " and some irregular and meaningless symbols from the row data downloaded from sketchEngine. Then use python code to read line by line, modify the format line by line, and add attributes, etc. The project converts txt format documents into arff format documents so that Weka can read the data.

After opening the arff document with Weka, it is found that Chile, Spain, and Argentina all contain about 3000 lines of sentences, the United Kingdom has 8000 lines of sentences on average 2.5 times that of the three countries, and Colombia has only 1900 lines of sentences, which is about 0.66 times that of the three countries. Considering that the five country categories are unbalanced, and the dataset sizes vary widely, this may lead to over accuracy for one country and extremely under accuracy for others. Such predictions are not universal and therefore meaningless. Therefore, we use the smote method to up sample the Colombian dataset by about 65 percent to about 2900 rows of data. The dataset for all countries is then filtered using the SpreadSubSample filter, resulting in a random subsample that controls for frequency differences between the rarest and most common categories. The problem of imbalanced classes of these two operations is addressed, and the datasets for all countries are controlled to around 3000 lines of sentences.

In order to improve the accuracy of the classifier model, on the one hand, the method of removing stop words is considered to remove some meaningless words in the dataset in order to improve the accuracy.

The application of natural language processing techniques to information retrieval, indexing, topic modeling, and text classification is increasing in engineering contexts. A standard component of such tasks is the removal of stop words, which are uninformative components of the data. (Sarica at el., 2021) nltk in python is a commonly used natural language processing tool, which collects many common Spanish stop words such as personal pronouns, prepositions, adverbs of degree and other words that do not have actual meanings. Therefore, a dedicated python program was written to use nltk functions to remove stop words in five countries and generate arff format. On the other hand, we've extracted best features and worst features to optimize the data further.

4 Modelling

We selected the best classifiers for the Spanish dialect recognition task from a set of classifiers provided by WEKA. They are NaiveBayes, IBK, BayesNet and SMO. Below we will introduce the principles of these classifiers and the reason why we choose them.

4.1 NaiveBayes

The classification principle of Bayesian classifier is to use the Bayesian formula to calculate the posterior probability of an object through the prior probability of an object and select the class with the largest posterior probability as the class to which the object belongs. The advantages of the Bayesian classifier are that the logic of the algorithm is relatively simple, easy to implement, and the time and space overhead of the algorithm in the classification process is small. In addition, the Bayesian classifier has stable classification efficiency, requires few parameters for the model, is less sensitive to the lack of data, and has strong robustness, so we choose it for our dialect recognition task.

4.2 IBK

The algorithm of IBK classifier belongs to a kind of KNN algorithm, which is a kind of Lazy Learning method. The idea of the KNN algorithm is very simple and intuitive: find the k training objects closest to the test object from the training set, and then find the dominant category from the k training objects and assign them to the test object. The reason we choose the IBK algorithm is that it is simple and effective, easy to understand and implement, has a low retraining cost, and is suitable for the sample set to be classified where the class domain crosses and overlaps a lot like our task.

4.3 BayesNet

BayesNet is a directed acyclic graph with probability annotations. Each node in the graph represents a random variable. If there is an arc between two nodes in the graph, it represents the random variable corresponding to the two nodes. are probabilistically dependent, otherwise the two random variables are conditionally independent.

The reason for choosing BayesNet is that the Bayesian network is different from the general knowledge representation method in modeling the problem domain. When the conditions or behaviors change, the model does not need to be revised. Moreover, it can graphically represent the joint probability between random variables and can handle various uncertain information.

4.4 SMO

The SMO algorithm is an algorithm to solve the quadratic optimization problem, and its most classic application is to solve the SVM problem. A remarkable feature of SMO is that it is also extremely easy to implement. Platt’s comparative testing against other algorithms has shown that SMO is often much faster and has better scaling properties. (Keerthi at el., 2001) By understanding the optimization goal of SVM, we can find that its ultimate purpose is to calculate a set of optimal values of alpha and constant term b. The central idea of the SMO algorithm is to select two alphas for optimization each time, and then fix the other alpha values.

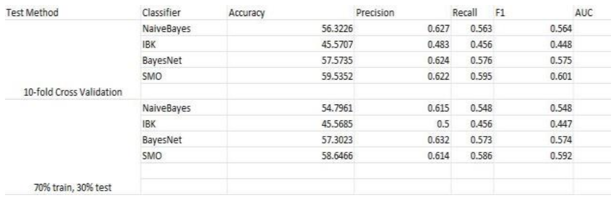
The biggest feature of SMO is that it can use the analytical method and completely avoid the complex iterative process of the quadratic programming numerical solution. This not only saves a lot of computation time, but also does not involve the accumulation of errors caused by the iterative method (which has caused great trouble in some other algorithms).

5 Evaluation

The project is evaluated by both classification result and running performance. The best accuracy result between two countries is around 70 percent, therefore, the accuracy result of classifying 5 countries cannot surpass 70 percent, since there are more interferences. In the evaluation stage, our team has chosen several classifiers, many of them had relatively awful outcomes and are not included in the final result form.

Before training classifiers, the corpus data went through several stages of pre-processing. As it was mentioned, the best features were extracted by using AttributeSelection in Weka with InfoGainAttributeEval, Ranker threshold set to 0.001. By doing this we obtained the best features as: profesionales, padres, cuidadores, cultura and amor. By eliminating worst features, we gained a little better classification result.

Following is result display and analysis:



Test Method	Classifier	Accuracy	Precision	Recall	F1	AUC
	NaiveBayes	56.3226	0.627	0.563	0.564	
	IBK	45.5707	0.483	0.456	0.448	
	BayesNet	57.5735	0.624	0.576	0.575	
	SMO	59.5352	0.622	0.595	0.601	
10-fold Cross Validation						
	NaiveBayes	54.7961	0.615	0.548	0.548	
	IBK	45.5685	0.5	0.456	0.447	
	BayesNet	57.3023	0.632	0.573	0.574	
	SMO	58.6466	0.614	0.586	0.592	
70% train, 30% test						

The corpus itself has a significant effect on results. Although we have done several levels of processing to the data, the Spanish corpus of UK still has negative effects on the result, as it includes many non-Spanish word and the content of it is deviated from what we've expected. This could only be improved until we change corpus or keywords. We've tried editing weights of errors to achieve balance, however it didn't work well so we abandoned this idea.

NaiveBayes, IBK, BayesNet and SMO have relatively better results than other classifiers. We implemented both 10-fold cross validation and percentage split (70% for training, 30% for testing) to evaluate the results. Accuracy, precision, recall, F1, AUC and running time are considered as evaluation standards. Among the four, SMO has the best accuracy of 59.5352 with 10-fold cross validation strategy, however, it costs over 100 minutes to finish classification. Meanwhile, SMO with percentage split strategy achieves an accuracy of 58.6466 with 10 minutes running time, which is acceptable. It also has the best F1 measurement, which means the classifier model is better than the rest. The worst performed classifier is IBK, which only has an accuracy around 45. NaiveBayes and BayesNet can also be considered as decent solutions, since they have good accuracy and best running performance among the four. Finally, our team consider SMO the best classifier and to have the best result.

6 Summary

Data mining should not be seen as a simple one-time exercise. Huge data collections may be analyzed and examined in an unlimited number of ways. As time progresses, so new kinds of structures and patterns may attract interest and may be worth seeking in the data. (Hand at el., 2001)

As this coursework is a simplified data mining task, it expects us to learn the key ideas of data science; the formal data mining process shall involve more complex and advanced techniques to process and evaluate. Through this project, we have learned basic knowledge about NLP, corpus collecting and processing, unbalanced sample processing e.g., SMOTE, features of corpus language, Weka operations, properties, and theories of different classifiers. By eliminating many negative interferences, we've improved the result to a great extent. We are also aware of the importance of teamwork, with which incredible things could be achieved; as in scientific research field, the spirit of cooperation is very necessary.

References

Witten, Frank, E., Hall, M. A., & Pal, C. J. (2017). Data mining: practical machine learning tools and techniques (Fourth Edition.). Elsevier.

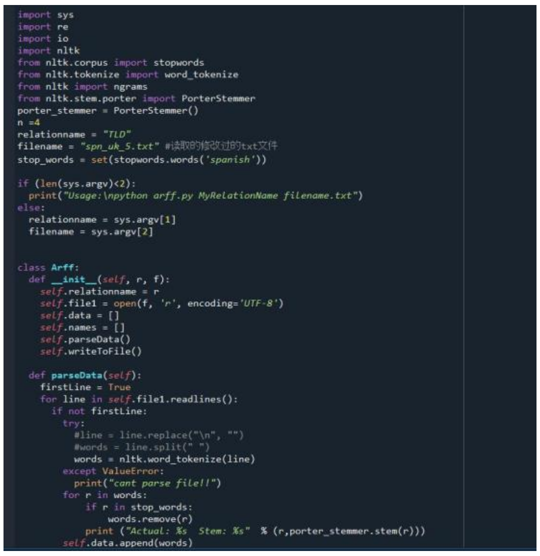
Hand, Mannila, H., & Smyth, P. (2001). Principles of data mining. MIT Press.

Keerthi, Shevade, S. K., Bhattacharyya, C., & Murthy, K. R. K. (2001). Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation, 13(3), 637–649.https://doi.org/10.1162/089976601300014493

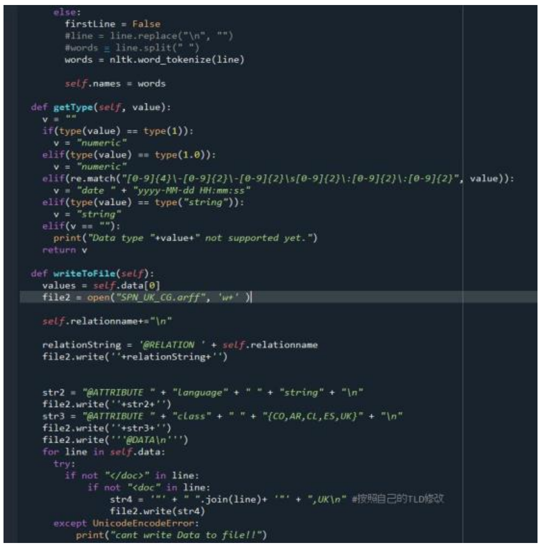
Sarica, & Luo, J. (2021). Stopwords in technical language processing. PloS One, 16(8), e0254937–e0254937.

Appendices

1. txt – arff python file

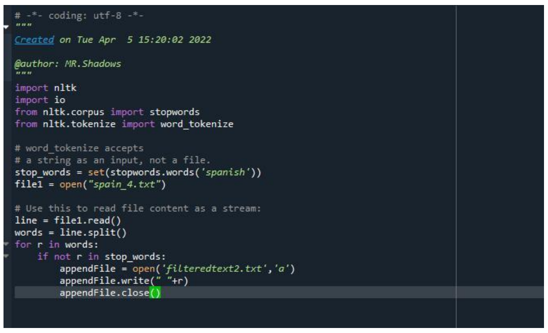


```python
import sys
import re
import io
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk import ngrams
from nltk.stem.porter import PorterStemmer
porter_stemmer = PorterStemmer()
n =4
relationname = "TLD"
filename = "spn_uk_5.txt" #读取的修改过的txt文件
stop_words = set(stopwords.words('spanish'))

if (len(sys.argv)<2):
  print("Usage:\npython arff.py MyRelationName filename.txt")
else:
  relationname = sys.argv[1]
  filename = sys.argv[2]


class Arff:
  def __init__(self, r, f):
    self.relationname = r
    self.file1 = open(f, 'r', encoding='UTF-8')
    self.data = []
    self.names = []
    self.parseData()
    self.writeToFile()

  def parseData(self):
    firstLine = True
    for line in self.file1.readlines():
      if not firstLine:
        try:
          #line = line.replace("\n", "")
          #words = line.split(" ")
          words = nltk.word_tokenize(line)
        except ValueError:
          print("cant parse file!!")
        for r in words:
            if r in stop_words:
                words.remove(r)
            print ("Actual: %s  Stem: %s"  % (r,porter_stemmer.stem(r)))
        self.data.append(words)
```



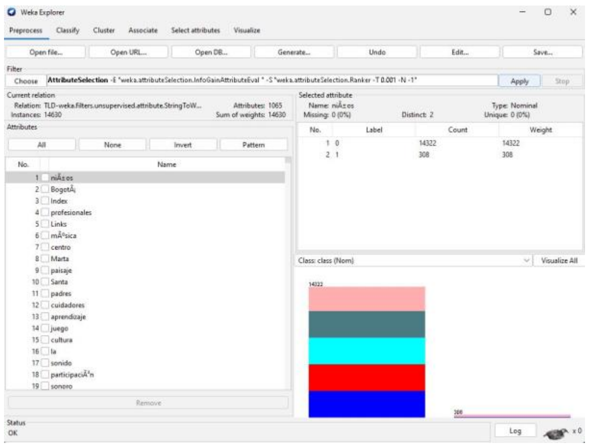
```python
      else:
        firstLine = False
        #line = line.replace("\n", "")
        #words = line.split(" ")
        words = nltk.word_tokenize(line)

        self.names = words

  def getType(self, value):
    v = ""
    if(type(value) == type(1)):
      v = "numeric"
    elif(type(value) == type(1.0)):
      v = "numeric"
    elif(re.match("[0-9]{4}\-[0-9]{2}\-[0-9]{2}\s[0-9]{2}\:[0-9]{2}\:[0-9]{2}", value)):
      v = "date " + "yyyy-MM-dd HH:mm:ss"
    elif(type(value) == type("string")):
      v = "string"
    elif(v == ""):
      print("Data type "+value+" not supported yet.")
    return v

  def writeToFile(self):
    values = self.data[0]
    file2 = open("SPN_UK_CG.arff", 'w+' )

    self.relationname+="\n"

    relationString = '@RELATION ' + self.relationname
    file2.write(''+relationString+'')


    str2 = "@ATTRIBUTE " + "language" + " " + "string" + "\n"
    file2.write(''+str2+'')
    str3 = "@ATTRIBUTE " + "class" + " " + "{CO,AR,CL,ES,UK}" + "\n"
    file2.write(''+str3+'')
    file2.write('''@DATA\n''')
    for line in self.data:
      try:
        if not "</doc>" in line:
            if not "<doc" in line:
                str4 = '"' + " ".join(line)+ '"' + ",UK\n" #按照自己的TLD修改
                file2.write(str4)
      except UnicodeEncodeError:
          print("cant write Data to file!!")
```

2. Nltk stopword removal python file



```python
# -*- coding: utf-8 -*-
"""
Created on Tue Apr  5 15:20:02 2022

@author: MR.Shadows
"""
import nltk
import io
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

# word_tokenize accepts
# a string as an input, not a file.
stop_words = set(stopwords.words('spanish'))
file1 = open("spain_4.txt")

# Use this to read file content as a stream:
line = file1.read()
words = line.split()
for r in words:
    if not r in stop_words:
        appendFile = open('filteredtext2.txt','a')
        appendFile.write(" "+r)
        appendFile.close()
```

3. Best features



Filter: AttributeSelection -E "weka.attributeSelection.InfoGainAttributeEval " -S "weka.attributeSelection.Ranker -T 0.001 -N -1"

Current relation
Relation: TLD-weka.filters.unsupervised.attribute.StringToW...   Attributes: 1065
Instances: 14630   Sum of weights: 14630

Selected attribute
Name: niÃ±os   Type: Nominal
Missing: 0 (0%)   Distinct: 2   Unique: 0 (0%)

No.	Label	Count	Weight
1	0	14322	14322
2	1	308	308

Class: class (Nom)

Attributes

No.	Name
1	niÃ±os
2	BogotÃ¡
3	Index
4	profesionales
5	Links
6	mÃºsica
7	centro
8	Marta
9	paisaje
10	Santa
11	padres
12	cuidadores
13	aprendizaje
14	juego
15	cultura
16	la
17	sonido
18	participaciÃ³n
19	sonoro

Status
OK