FINAL PROJECT - HW2

CENG 3548, Web Mining
Spring 2020 - 2021

Team Members: Gizem Kurnaz & Ayşe Ceren Çiçek

Student IDs: 170709059 & 170709009

WEBSITE CLASSIFICATION

Abstract

Website classification is the process of assigning a website to one or more category labels. In our project, we build some models using machine-learning techniques to classify a website into one of the given categories.

1 Introduction

Website classification is a method of classifying websites' main content or topic according to a set of defined categories. Website classification can help improve the quality of web searches. The general problem of web page classification can be divided into two which are subject classification and function classification. In our project, we classified websites based on their subject or topic.

2 Dataset

We used the DMOZ *dataset*[1] to build a web classification model, a large communally maintained open directory that categorizes web content. DMOZ closed in 2017 because AOL no longer wished to support the project. We found the split version of the dataset in a Github repository. The repository contains 37 CSV files. In total, they contain more than 2 million entries. Due to performance issues, we only used 500 rows from each document and combined them into one CSV file.

The dataset contains 10 columns, which are: id, url, title, description, priority, topic, topicId, stemd, topTerms, topStems. We only used url, title, and topic columns. We added ids and got the general topic from the topic column. We also cleaned the title column from punctuations and stopwords and stored them in a new column called cleaned_text.

3 Data Preprocessing

To clean the data, we dropped duplicate URLs and checked for null values. We added new ids for URLs. After the preprocessing step, we have 18888 rows, 5 columns and 10 categories in total.

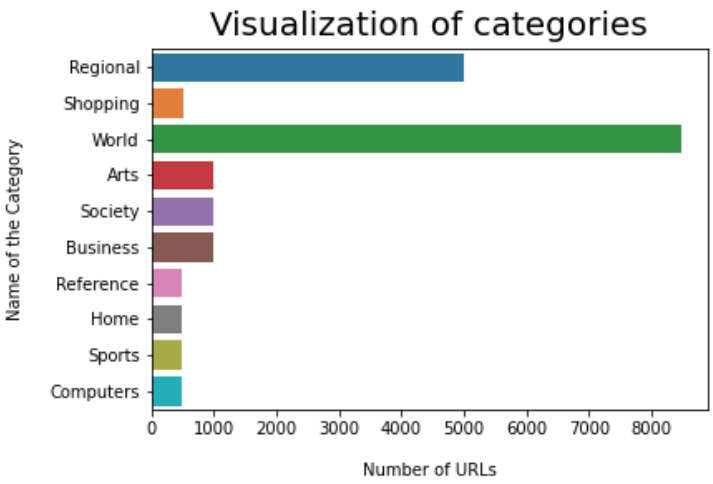

Figure 1: Visualization of number of websites for each categories

4 Building Models

4.1 Part 1 - Ayşe Ceren Çiçek

4.1.1 Text Preprocessing with spaCy

spaCy[2] is a free, open-source library for advanced Natural Language Processing (NLP) in Python. We used it to clean the texts in the 'title' column. We first tokenized the text, segmented it into words, and for each token, we checked if it is a stopword (the, a, an, is...), punctuation, a numeric value, or a null value. If any of those situations is true, we do not add this word to our tokens array. So this word will be removed.

Then we added those cleaned title texts under a new column 'cleaned_text'.

4.1.2 Visualization

For each category, we have shown the most commonly used 20 words.

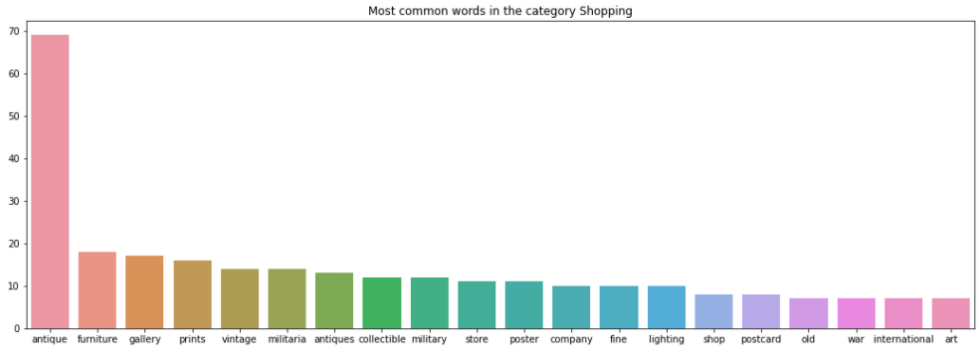

Figure 2: Most commonly used 20 words for Shopping category

4.1.3 Train & Test Data

To build classifiers we split 25% of the data for testing and the other for training. We will classify websites' categories based on their title text so input column (X) is cleaned_text and output column (y) represents categories.

TfidfVectorizer transforms text to feature vectors that can be used as input to the estimator. We will use it to calculate the weight of each word for each text document.

4.1.4 Models

We build a **Logistic Regression Model**, a **Decision Tree Model**, and a **Multinomial Naive Bayes Model** and showed their classification report and confusion matrix.

A) Logistic regression: It aims to measure the relationship between a categorical dependent variable and one or more independent variables (usually continuous) by plotting the dependent variables' probability scores. Our independent variable is 'cleaned_text' and the dependent variable is a category that we will predict.

	precision	recall	f1-score	support
Regional	0.93	0.43	0.59	262
Shopping	0.73	0.38	0.50	263
World	0.89	0.34	0.49	121
Arts	0.94	0.71	0.81	115
Society	0.81	0.48	0.60	128
Business	0.77	0.72	0.75	1204
Reference	0.87	0.39	0.54	124
Home	0.92	0.73	0.81	237
Sports	1.00	0.41	0.58	101
Computers	0.74	0.97	0.84	2167
accuracy			0.77	4722
macro avg	0.86	0.56	0.65	4722
weighted avg	0.79	0.77	0.75	4722

Figure 3: Classification report of Logistic Regression Model

1) Precision is the ability of a classifier not to label an instance positive that is actually negative. In Sports category, precision value is 1.00 which means all of predictions were correct.

2) The recall is the ability of a classifier to find all positive instances. So it defines the percentage of the positive cases we had caught. The macro average (mean average) is 0.56 which shows us that we only caught about half of the positive cases.

3) The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0.

4) Support is the number of actual occurrences of the class in the specified dataset. It seems that our support values are imbalanced, which may indicate weaknesses in the reported scores.

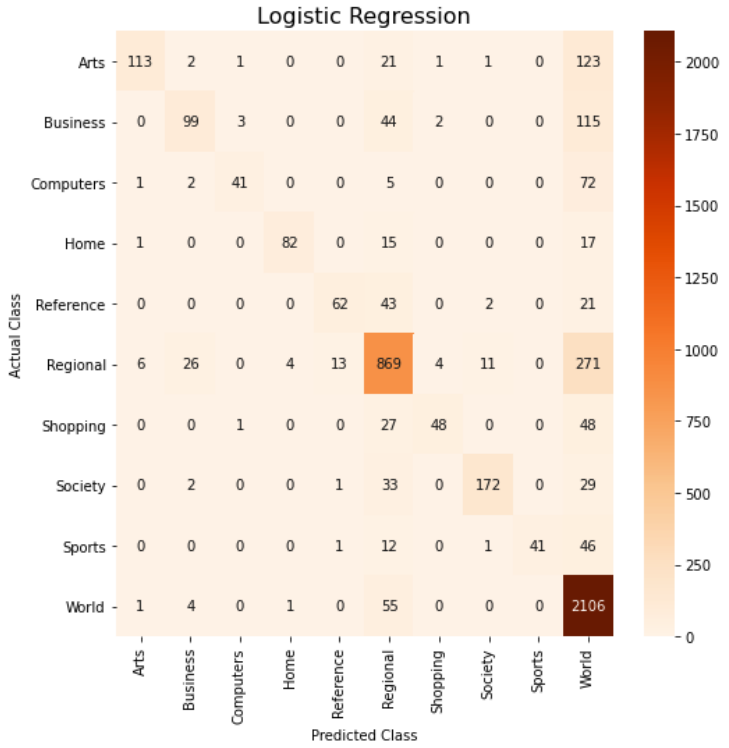

Figure 4: Heatmap of Logistic Regression Model

The diagonal values show the number of correct predicted values. For instance, with this model, 2106 websites have been correctly classified as the 'World' category. But 55 websites have been classified as 'Regional while their correct category was 'World'.

The high number of incorrect predictions are:

- 123 websites with category Arts, predicted as World.
- 115 websites with category Business, predicted as World.
- 72 websites with category Computers, predicted as World.
- 271 websites with category Regional, predicted as World.

```
Accuracy of Logistic Regression:  0.7791190173655231
Accuracy of Decision Tree:  0.650571791613723
Accuracy of Multinomial Naive Bayes:  0.777424819991529
```

Figure 5: Accuracy of our models

When we compared the 3 models, we see that they have close averages of precision, recall, and f1-score values but the decision tree model seems to have a poor performance.

4.2 Part 2 - Gizem Kurnaz

4.2.1 Text Preprocessing with NLTK

NLTK[3] (Natural Language Toolkit) is a suite that contains libraries and programs for statistical language processing. It is one of the most powerful NLP libraries, which contains packages to make machines understand human language and reply to it with an appropriate response.

We removed special characters and numbers, convert document to lowercase, tokenized documents, filtered out the stop-words, reconstructed the document, using some methods in this package.

Then we added those cleaned title texts under a new column 'cleaned_title'.

4.2.2 Visualization with Wordcloud

A word cloud is a novelty visual representation of text data, typically used to depict keyword metadata (tags) on websites, or to visualize free form text. We used wordcloud to find the most frequently occuring words so we saw in which labels, what type of websites are frequently occuring.

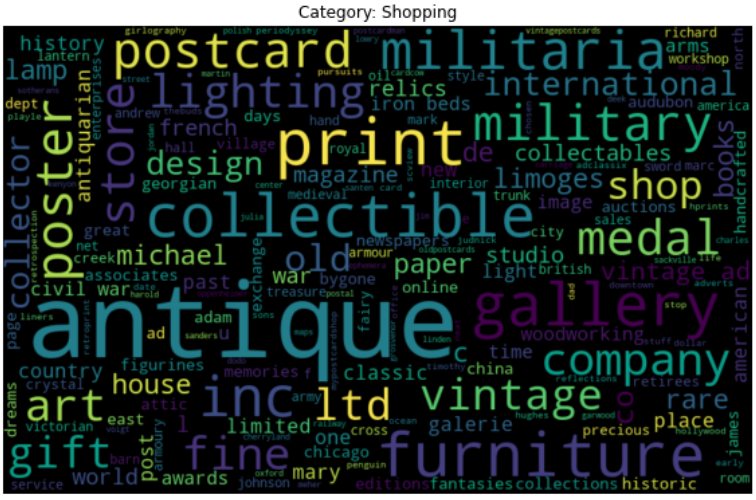

Figure 6: Visualization of most used words in Shopping Category with Wordcloud

4.2.3 Countvectorizer

A vectorizer helps us convert text data to computer understandable numeric data.

4.2.3.1 Count Vectorizer: Counts the frequency of all words in our corpus, sorts them and grabs the most recurring features (using max_features hyperparameter). These are all boolean values. But these results are mostly biased and our model might loose out on some of the important less frequent features.

4.2.3.2 TFIDF Vectorizer TFIDF is a statistical measure have fixed the issues with CountVectorizer. It consists of 2 parts, TF (Term Frequency) multiplied with IDF (Inverse Document Frequency). The main intuition being some words that appear frequently in 1 document and less frequently in other documents could be considered as providing extra insight for that 1 document and could help our model learn from this additional piece of information.These are relative frequencies identified as floating point numbers.

In our project, We first used count vectorizer to convert text data into vectors. However, when we applied the Multinomial Naive Bayes Classifier Model in the next stages, we decided to use a different vectorizer because the scores were very low. That's why we later used TfidfVectorizer and got much better results.

4.2.4 Cross Validation

In classification, we first separate our data set as train and test sets, then create a model on the train data set and test the predictions made on the test data set. However, there may be some problems in train / test separation. We may not have been able to make the dataset separation randomly. We may have chosen only men or women of a certain age, from a certain region, and built a model on them. This will cause overfitting problem. We can solve this problem with Cross Validation.

In K-Folds Cross Validation, we divide our data into k different subsets. We use k-1 subsets to train our data and leave the final subset as test data. The average error value obtained as a result of k experiments indicates the validity of our model.

In this project;
We wrote a function named cv_score that we can validate k-fold. This function takes model as input and define for fitting on the training set for each fold. And then function calculate scores for each fold and print and save. We used four different models as input to this function. These models are:

- Multinomial Naive Bayes Classifier Model
- Logistic Regression Model
- Decision Tree Classifier Model
- Random Forest Classifier Model

And finally we created and visualized a dataframe to compare the results we got for each model.

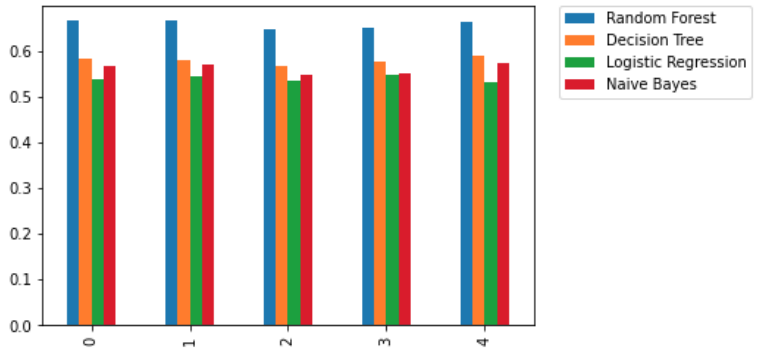

Figure 7: Visualization of most used words in Shopping Category with Wordcloud

When we examine the graph we have created, we can see that the Random Forest Classifier Model has the best performance and the Logistic Regression Model has the worst performance.

5 Conclusion

Website classification is the process of assigning a website to one or more category labels. In this project, we used the DMOZ dataset to build models. We applied text preprocessing to websites' contents with NLTK and SpaCy libraries. Then we found the most used words for a category. With this information, we built classification models with Logistic Regression, Decision Tree, Multinomial Naive Bayes, Random Forest classifiers.

References

[1] DMOZ Dataset

[2] SpaCy

[3] NLTK