INF342: Domain Classification Challenge

Giannis Nikolentzos and Michalis Vazirgiannis

Department of Informatics
Athens University of Economics and Business

May 2019

1 Description of the Challenge

The goal of this project is to study and apply machine learning/data mining techniques to a domain classification problem. Domain classification (or domain categorization) is the task of assigning a domain name into a set of one or more predefined categories (i.e., classes). Such techniques find applications in several fields, such as in managing web hierarchies, and in question answering systems. In this project, you will evaluate your methods on a dataset of greek domain names. More specifically, you are given both a web graph consisting of several thousands of domains, and the textual content of the webpages of a subset of these domains. The problem is very related to the well-studied problems of text categorization and node classification. The pipeline that is typically followed to deal with the problem is similar to the one applied in any classification problem; the goal is to learn the parameters of a classifier from a collection of training domain names (with known class information) and then to predict the class of unlabeled domains.

The challenge is hosted on Kaggle[1], a platform for predictive modelling on which companies, organizations and researchers post their data, and statisticians and data miners from all over the world compete to produce the best models. The challenge is available at the following link: `https://inclass.kaggle.com/c/inf342-datachallenge-2019`. To participate in the challenge, use the following link: `https://www.kaggle.com/t/768e7f75afb84cb7819a9692e353e647`. The edgelist and the textual content of the webpages can be downloaded from this link: `https://www.dropbox.com/sh/i450n1m3fn31dx9/AAAbxgfHiXpP2x76CJ_4E06La?dl=0`.

2 Dataset Description

As mentioned above, you will evaluate your methods on a dataset of greek domains. The dataset was generated from a large crawl of the greek web that was performed by the researchers of the Data & Web Mining group. You are given the following files:

1. **edgelist.txt**: a large part of the greek web graph stored as an edgelist. Nodes correspond to domain names (i.e., *aueb.gr*) and edges to hyperlinks. For example, if there is a hyperlink from domain w to domain v, there is also a directed edge from node w to node v in the graph. The graph consists of $65,208$ vertices and $1,642,073$ directed edges in total.

[1] `https://www.kaggle.com/`

Category	# of Train Domains
athlitismos	158
diaskedasi-psyxagogia	161
eidiseis-mme	160
katastimata-agores	159
pliroforiki-diadiktyo	162

Table 1: Description of the 5 categories

2. **domains.zip**: it contains the textual content of webpages extracted from $40,600$ greek domains. The textual content has been extraced from the HTML source code of the webpages. For each domain, there is a zip file containing the textual content of its webpages.

3. **train.csv**: it contains 800 labeled domain names. Each row of the file contains the name of a domain name and its topic.

4. **test.csv**: this file contains the names of 200 domains. Each of these domains belongs to one of the 5 possible classes. The final evaluation of your methods will be done on these domains and the goal will be to predict the category to which each domain belongs.

As regards the 5 categories, these are illustrated in Table 1.

3 Evaluation

The performance of your models will be assessed using the multi-class logarithmic loss measure. This metric is defined as the negative log-likelihood of the true class labels given a probabilistic classifier's predictions. Specifically, the multi-class log loss is defined as:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij})$$

where N is the number of samples (i.e., web hosts), C is the number of classes (i.e., the 5 different topics), y_{ij} is 1 if sample i belongs to class j and 0 otherwise, and p_{ij} is the predicted probability that sample i belongs to class j.

4 Provided Source Code

You are given two scripts written in Python that will help you get started with the challenge. The first script (`graph_baseline.py`) uses solely graph-based features with a logistic regression classifier for making predictions, and achieves a log loss score of 1.41 on the public leaderboard. The second script (`text_baseline.py`) uses features extracted from the webpages of the domains along with a logistic regression classifier. This model achieves a log loss score of 1.28 on the public leaderboard. As part of this challenge, you are asked to write your own code and build your own models to predict the topic of each domain of the test set. You are advised to use both graph-theoretical and textual information.

5 Useful Python Libraries

In this section, we briefly discuss some tools that can be useful in the project and you are encouraged to use.

- A very powerful machine learning library in Python is `scikit-learn`[2]. It can be used in the pre-processing step (e.g., for feature selection) and in the classification task (a plethora of classification algorithms have been implemented in `scikit-learn`).
- Since you will deal with data represented as a graph, the use of a library for managing and analyzing graphs may be proven important. An example of such a library is the `NetworkX`[3] library of Python that will allow you to create, manipulate and study the structure and several other features of a graph.
- Since you will also deal with textual data, the Natural Language Toolkit (`NLTK`)[4] of Python can also be found useful.

6 Details about the Submission of the Project

Your final evaluation for the project will be based on the presentation you will give, on your position on the leaderboard, on the log loss that will be achieved on the test set, as well as on your total approach to the problem. As part of the project, you have to submit the following:

- A 3-4 pages report, in which you should describe the approach and the methods that you used in the project. Since this is a real classification task, we are interested to know how you dealt with each part of the pipeline, e.g., how you created your representation, which features did you use, if you applied dimensionality reduction techniques, which classification algorithms did you use and why, the performance of your methods (log loss and training time), approaches that finally didn't work but is interesting to present them, and in general, whatever you think that is interesting to report.
- A directory with the code of your implementation.
- Create a `.zip` file containing the code and the report and submit it to the e-class platform.
- **Deadline: 23/6/2019**

[2] `http://scikit-learn.org/`
[3] `http://networkx.github.io/`
[4] `http://www.nltk.org/`