11 - Is this website safe for your kids?

Motivation

With greater access to the Internet today among households, parents, it is all the more important for netizens to have a sense of media literacy and discretion when surfing the internet. Children, at an age of formative growth and possessing high impressionability, should be protected from websites that have content potentially harmful to their mental well-being. Hence, a machine-learning classifier which categorises websites into safe-for-work and not-safe for work is very useful to help children avoid potential traps and seeing unwanted things on the World Wide Web. Such a classifier can be useful for automated parental-control software. Furthermore, an ability for such a classifier to gain a rough gauge of a webpage's content is useful for it to generate recommendations for more useful websites for children.

Statement of the Problem/Task

Topic: Information Extraction **Domain**: URLs

We aim to build a machine learning classifier that determines if a website is recommended or unsafe for children based on its uniform resource locator. We will attempt to classify websites into *"recommended for kids"*, *"unsafe for kids"*, and *"neutral"*. Furthermore, among the websites that are deemed "neutral", we aim to categorise the websites according to specific topics of interest to web users, like Health, Computers and Arts. In doing so, we hope to gain insights on the following questions:

- How can we discern whether a website is **unsafe for children** just from the URL?
- How can we decide if a website should be specifically **recommended for children** from its URL?
- How can we get a sensing of the **website's content** from its URL?

General Approach

We will be implementing a classification model that is able to assign the labels '*RECOMMENDED-FOR-KIDS*', '*NOT-SAFE-FOR-KIDS*' and '*NEUTRAL*' to each input URL. For each URL which is assigned the '*NEUTRAL*' label, the language model will further assign a sub-label indicating which subject the webpage corresponds to, like 'HEALTH', 'ARTS' or 'COMPUTERS', for instance. We will be training and testing the model using datasets found on Kaggle which classify URLs into different labels.

1. **Preprocessing**
   - ❏ Implement parser & tokenizer
   - ❏ Clean up datasets obtained from Internet
   - ❏ Conduct label remapping: map various class labels in existing datasets to three categories: '*RECOMMENDED-FOR-KIDS*', '*NOT-SAFE-FOR-KIDS*' and '*NEUTRAL*'
2. **Feature Engineering & Selection**
   - ❏ Observe dataset & brainstorm potential features
   - ❏ Conduct a controlled-variable experiment, modifying features one by one and analyzing effect on result. Keep the subset of features that optimizes classification performance.
3. **Model Comparison**
   - ❏ Use different models (Naive Bayes, Logistic Regression, Neural Networks, etc.) with the same features and compare their performances using the F-Score.
4. **Parameter Tuning & Ablation Study**
   - ❏ After finalising features and models, continue to tune hyperparameters to increase the accuracy.
   - ❏ After finalising on the optimal model and parameters, conduct an ablation study on the effect of each factor on the prediction accuracy.

Evaluation

Satisfactory project outcome (C grade):

- ❏ Reasonable accuracy of classification achieved
- ❏ Clear explanation in posters and videos

Excellent project outcome (A grade):

- ❏ High accuracy of classification
- ❏ Unique extension (sentimental analysis on HTML file) and applicability to real-world problems
- ❏ Clear explanation on why the implemented model is chosen
- ❏ Appealing posters and videos with good content

Resources

Kaggle Datasets: (accessible through open source)

- URL Classification Dataset
- Website Classification Using URL
- News Category Dataset

Readings:
We will make references to the papers written by Prof Min-Yen Kan on the various algorithms used for fast webpage classification using URL features. If the papers are not accessible online, we will email Prof Min-Yen Kan for permission to access these resources.

Software: (accessible through Github/open source libraries)
Pytorch, software that can open large (> 1 million rows) csv files

Schedule / Role Assignment

Week	Task
7	Finalize Project Proposal ***(All Members)***
8	Label remapping ***(In-charge: Team Member 1)*** Implement parser ***(In-charge: Team Member 2)***
9	Brainstorm for possible features ***(All Members)*** Start on model implementation ***(All Members)***
10	Implement at least 2 models concurrently ***(one Team member in-charge of each model)*** Compare the performance of models ***(All Members)***
11	Decide on optimal model ***(All Members)*** Fine-tune parameters ***(In-charge: Team Member 3)***
12	Fine-tune parameters Make Posters and Videos ***(In-charge: Team Member 4)***
13	Improve Posters and Videos ***(All Members)*** Write Final Project Report ***(All Members)***

Appendix

Ahmed, S. (2020, January 30). Website classification using url. Retrieved March 02, 2021, from https://www.kaggle.com/shaurov/website-classification-using-url

Misra, R. (2018, December 02). News category dataset. Retrieved March 02, 2021, from https://www.kaggle.com/rmisra/news-category-dataset

Shawon, A. (2018, August 19). Url classification dataset [dmoz]. Retrieved March 02, 2021, from https://www.kaggle.com/shawon10/url-classification-dataset-dmoz

Min-Yen Kan and Hoang Oanh Nguyen Thi (2005) Fast webpage classification using URL features. In Proc. of Conf. on Info and Knowledge Management (CIKM '05). Bremen, Germany, November 2005. Poster Paper. pp. 325-236.

Min-Yen Kan (2004) Web Page Classification Without the Web Page. In Proceedings of the 13th International WorldWide Web Conference (WWW2004), May 2004. New York, New York, USA. Poster Paper.