INF554 - Machine and Deep Learning
Data Challenge: Retweet Prediction

November 2022

1 Introduction

The goal of this data challenge is to study and apply machine learning/artificial intelligence techniques to a real-world regression problem. In this regression problem, your mission is to build a model that can **accurately predict the number of retweets a tweet will get**. We provide a novel Twitter dataset with the 2022 French presidential election as the central topic. The tweets used for building this dataset date from February 14, 2022 to April 5, 2022. This dataset[1] corresponds to a large and coherent corpus consisting of small pieces of text related to the election candidates and major events during their campaigns. More specifically, it contains tweet-related information, such as the text and the number of hashtags, mentions and URLs contained in the tweet, and user-related information, such as the number of followers and tweets that the user has published. This setting poses several challenges ranging from data cleaning/preprocessing to model design and hyperparameter tuning. Your solution can be based on both supervised and unsupervised learning techniques. You should aim for a **minimum Mean Absolute Error (MAE)**, i.e., the MAE will be the loss function we use to evaluate your models. This data challenge is hosted on Kaggle as an in-class competition. In order to access the competition you must have a Kaggle account. If you do not have an account you can create one for free. The URL to register for the competition and have access to all necessary material is the following:

https://www.kaggle.com/t/781ae6df09734303a73185dd3aafe0fd

2 Dataset Description

As part of the challenge, you are given the following files:

1. **train.csv** - 353.969 tweets for which we know the number of retweets. Each row has the following fields, 'TweetID', 'timestamp', 'retweets_count', 'verified', 'statuses_count', 'followers_count', 'friends_count', 'mentions', 'urls', 'hashtags', 'favorites_count' and 'text'.
2. **evaluation.csv** - 240.744 tweets. The number of retweets is not available (your task is to predict it). Each row has the following fields, 'TweetID', 'timestamp', 'verified',

[1] Abdine, Guo, Rennard & Vazirgiannis. "Political Communities on Twitter: Case Study of the 2022 French Presidential Election." *arXiv preprint arXiv:2204.07436* (2022).

'statuses_count', 'followers_count', 'friends_count', 'mentions', 'urls', 'hashtags', 'favorites_count' and 'text'.

3. **mean_predictions.txt, constant_predictions.txt** - sample submission files in the correct format (the predictions have been generated by dummy baselines as described in Section 4 below).

3 Task and Evaluation

For each tweet in the test set, your model should predict the number of retweets it will get after its publication. The evaluation metric for this competition is the Mean Absolute Error (MAE). The MAE metric is calculated by dividing the sum of absolute differences between the predicted number of retweets (p_i) and the observed number of retweets (a_i) by the number of observations (N), i.e.,

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|p_i - a_i\right|$$

4 Provided Source Code

You are given two baseline scripts written in Python that will help you get started with the challenge. The first script **dummy_baseline.py** is a Python script with two baselines, one that given each row predicts the number of retweets is equal to the mean value of the training dataset and one that constantly predicts zero. These baselines achieve MAE of 26.13779 and 15.55762, respectively. The second script **baseline.py** is a Python script containing a simple baseline method that only uses the text of the tweet. The method transforms the text using the TF-IDF vectors and then trains a Gradient Boosting Regressor. The MAE score of this baseline is approximately 25.85580.

5 Useful Python Libraries

In this section, we briefly discuss some packages that can be useful in the challenge:

- A very powerful machine learning library in Python is scikit-learn[2]. It can be used in the preprocessing step (e.g., for feature selection) and in the classification task (several regression algorithms have been implemented in scikit-learn).
- A very popular deep learning library in Python is PyTorch[3]. The library provides a simple and user-friendly interface to build and train deep learning models.

[2] http://scikit- learn.org/
[3] https://pytorch.org/

- Since you will deal with textual data, the Natural Language Toolkit (NLTK)[4] of Python can also be found useful.
- Gensim[5] is a Python library for unsupervised topic modeling and natural language processing, using modern statistical machine learning. The library provides all the necessary tools for learning word and document embeddings. An alternative to it is FastText[6].
- If you do not want to spend a lot of time producing the word embeddings, you can use transfer learning. You can download a pretrained set of word embeddings on GoogleNews[7], Glove[8] or Numberbatch[9]).
- In case you want to work with the data represented as a graph, the use of a library for managing and analyzing graphs may prove to be important. An example of such a library is the NetworkX[10] library of Python that will allow you to create, manipulate and study the structure and several other features of a graph.

6 Grading Scheme

Each team has to be composed of 2 or 3 students. No larger or smaller teams will be accepted. Please join and use the slack channel (”team-finding”) to organise yourselves into groups of 3 or coordinate offline. Group choices need to be submitted to us by **Friday 25th November at 17:00** at the below link:

https://forms.gle/8qYsWF6knpN5CGR18

Grading will be out of 100 points in total. Each team should deliver:

A submission on the Kaggle competition webpage. (20 points) will be allocated based on raw performance only, provided that the results are reproducible. That is, using only your code, the data provided on the competition page and *any additional resources you are able to reference and demonstrate understanding of*, the jury should be able to train your final model and use it to generate the predictions you submitted for scoring.

A zipped folder in moodle (30 points) including:

1. A folder named ”code” containing all the scripts needed to reproduce your submission.

[4] http://www.nltk.org/
[5] https://radimrehurek.com/gensim/
[6] https://fasttext.cc/
[7] https://code.google.com/archive/p/word2vec/
[8] https://nlp.stanford.edu/projects/glove/
[9] https://github.com/commonsense/conceptnet-numberbatch
[10] http://networkx.github.io/

2. A README file with brief instructions on how to run your code and where it expects the original data files.
3. A report (.pdf file), of max 3 pages, excluding the cover page and references. In addition to your self-contained 3-page report, you can use up to 3 extra pages of the appendix (for extra explanations, algorithms, figures, tables, etc.). Please ensure that both your real name(s) and the name of your Kaggle team appear on the cover page.

The 3-page report should include the following sections (in that order):

- **Section 1: Feature Selection/Extraction (10 points).** Independent of the prediction performance achieved, the jury will reward the research effort done here. Best submissions will capture both graph and text information. You are expected to:
    1. Explain the motivation and intuition behind each feature. How did you come up with the feature (e.g., are you following the recommendation of a research paper)? What is it intended to capture?
    2. Rigorously report your experiments about the impact of various combinations of features on predictive performance, and, depending on the regressor, how you tackled the task of feature selection.
- **Section 2: Model Choice, Tuning and Comparison (15 points).** Best submissions will:
    1. Compare your model against different regression models(e.g., Neural Network, Support Vector Regression, Random Forest...).
    2. For each regressor, explain the procedure that was followed to tackle parameter tuning and prevent overfitting.
- Report and code completeness, organization and readability will be worth **5 points**. Best submissions will (1) clearly deliver the solution, providing detailed explanations of each step, (2) provide clear, well-organized and commented code, and (3) refer to research papers. You are free to search for relevant papers and articles and try to incorporate their ideas and approaches into your code and report as long as (a) it is clearly cited within the report, (b) it is not a direct copy of code and (c) you are able to demonstrate understanding of the content you incorporated.

An oral presentation of your project and the achieved results (50 points) Oral presentations will be scheduled in the week of the 12th of December. A schedule on which you can choose presentation slots will be released after the team submissions have been made. We ask you to prepare 15-minute presentations, which will then be followed by questions from the examiners. It is required that all group members are present and take a speaking role during the presentation.

7 Submission Process

Submission files should be in **.txt format**, and contain two columns respectively named "TweetID" and "retweets_count". The "TweetID" column should contain the tweet id as given in the evaluation.csv. The “retweets_count" column should contain the predictions (non-negative integer). Note that a sample submission file is available for download **(mean_predictions.txt)**. You can use it to test that everything works fine.

The competition ends on **Wednesday 7th December at 17:00**. This is the deadline for you to submit a compressed file containing your source code and final report, explaining your solutions and discussing the scores you have achieved. Until then, you can submit your solution to Kaggle and get a score at most 5 times per day. There must be one final submission per team.