Coding Assignment 3: Ciphertext Classification

Due: March 23, 2022, at 23:59 Pacific Time (11:59 PM)

This assignment counts for 10% of the course grade.

Assignments turned in after the deadline but before the end of March 26 are subject to a 20% grade penalty.

Overview

Privacy-preservation is an important issue in machine learning. In quite a few application scenarios (e.g., healthcare, banking, legal fields, private message processing), real data may not be shown to the developer of a data-driven machine learning model (for NLU or other tasks).

In this assignment, you will experiment with a simple scenario for privacy-preserving NLU: developping a text classification model based on ciphertext. You will be provided with a training set and a development set of (binary) labeled ciphertext, as well as an unlabeled test set. All the sentences in the dataset are encrypted on the word level (hence, words in a sentence are still separated by white spaces).

You are free to experiment with **any methods of representation, encoding and classifying** the ciphertext using the training and dev data, and **submit your prediction** on the unlabeled test set. We will compare your prediction with the ground-truth labels of the test set on our side. Grading will be based on the ranking of your submitted prediction among all of those in the class.

(**Please make sure to read the Q&A part**)

A compressed ZIP file is to be released on Blackboard when the assignment is available. The uncompressed archive contain the following files:

- main.ipynb: The **Python 3** Jupyter notebook you will need to fill in with your training and inference code, and predict the results.
- Dataset:
  - train_enc.tsv: labeled training data. Each line a '\t' separated tuple in the form of (label, ciphertext sentence).
  - dev_enc.tsv: labeled dev data.
  - test_enc_unlabeled.tsv: unlabled test data. Every line is a ciphertext sentence.
- upload_document.txt: the documentation file where you need to fill in the blanks to describe your method.

Programs

The provided Jupyter notebook already contains the beginning part of code for reading the data, and the end part to generate "upload_predictions.txt". You need to fill in the "Main Code Body", and put the 2028 predictioned labels into the *results* list.

Restrictions. Your method needs to be implemented using **python 3**, and should be executable on a single personal machine (e.g., your own laptop). Other than those, you are free to use any python package (e.g., pytorch, gensim, sklearn, fasttext, etc.).

This assignment requests **submitting three files (DO NOT CHANGE the filenames!)**:

1. upload_predictions.txt: The predictions of your model on the 2028 unlabeled test set ciphertext sentences. This file should have **exactly 2028 lines, every line is either 0 or 1.** (submit this on **Vocareum**)
2. upload_document.txt: Fill in the blanks of that file to accordingly describe how your model represents, encodes and classifies the ciphertext, and how you have developed the model. (submit this on **Blackboard**)
3. main.ipynb: This Jupyter notebook already contains the beginning part to read the data, and the end part to generate "upload_predictions.txt". You need to fill in the "Main Code Body" (submit this on **Blackboard**).

Multiple submissions are allowed; only the final submission will be graded. Do not include any other files in your submission. You are encouraged to **submit early and often** in order to iron out any problems, especially issues with the format of the final output.

We will grade using a **leaderboard protocol**: the ground-truth labels on the test set are not released to you, and we will compare your prediction with them to calculate the **accuracy** of your prediction.

The prediction of your predictions will be measured automatically; failure to format your output correctly may result in very low scores, which will not be changed.

For full credit, make sure to submit your assignment well before the deadline. The time of submission recorded by the system is the time used for determining late penalties. If your submission is received late, whatever the reason (including equipment failure and network latencies or outages), it will incur a late penalty.

Grading

We will grade using a **leaderboard protocol**. After the due date, we will calculate the accuracy of your submitted prediction. Then your grade will be decided based on the **standing** of your submission among all **valid ones**. Specifically, the top 20% valid submissions will get a score of 10, then the next 20% will get a score of 9, etc. The bottomline score of a **valid submission** will be 6. However, your submission will be considered invalid and will result in a score lower than the bottomline or of zero if one of the following happens:

- The accuracy of your submission is very much close to a trivial baseline performance (e.g., random guessing).
- The code in your submitted jupyter notebook obviously cannot reproduce your predictions.
- You haven't submitted the documentation file, so we don't know how your results can be reproduced; or, based on the submitted document, we find factual mistakes of your method that should cause it to not work (e.g., giving random or nonsensical predictions).
- Your submission has been redflaged by plagarism checking.

Q&A

Q & A

How is the ciphertext encrypted?

Some encryption algorithm that was much more complicated than a Caesar Cipher was applied to encrypt every word in the original text (punctuations were not encrypted). Words are still separated with whitespaces after the encryption. Labels were unchanged.

How many labels are there?

This is a binary classification task. Only two labels are used: 0 and 1.

Does the ciphertext contain non-ascii characters:

Yes. So make sure you read file using encoding='utf-8', as it has been written in the Jupyter notebook.

Is there a pre-trained word embedding for this ciphertext?

No.

Are the test data from the same distribution of the training/dev data?

Yes, they were splits from the same dataset.

Is the dataset balanced?

Almost balanced.

How do I know if my method is well-performing or not?

How do I know if my method is well-performing or not?

You can try to tell it by the dev set performance.

What's the evaluation metric?

Accuracy.

Can I use bag-of-words + a simple FFNN classifier?

This could be a meaningful baseline method. But you want to try other representation techniques and classifiers to get better performance.

Can I try to decipher the ciphertext?

We do not forbid you to do so. But that is most likely going to be a waste of time (the encryption algorithm is not that easy to hack, and that is also not going to provide you a huge advantage in getting accurate predictions).

Is there a single best solution?

This assignment is an open-ended problem. So there is hardly a single best solution (don't ever try to ask TAs about it: they do not know it either).

Collaboration and external resources

- This is an individual assignment. You may not work in teams or collaborate with other students. You must be the sole author of 100% of the code you turn in.
- You may not look for solutions on the web, or use code you find online or anywhere else.
- Failure to follow the above rules is considered a violation of academic integrity, and is grounds for failure of the assignment, or in serious cases failure of the course.
- You will not locate the test data on the web or anywhere else, since we ourselves encrypted the data, and the raw data were also a mixture of multiple other datasets (including internal ones).
- You are free to use any python 3 packages for this project. This assignment is an open-ended problem, so there is hardly a single best solution.
- We use plagiarism detection software to identify similarities between student assignments, and between student assignments and known solutions on the web. **Any attempt to fool plagiarism detection, for example the modification of code to reduce its similarity to the source, will result in an automatic failing grade for the course.**
- Please discuss any issues you have on the Piazza discussion boards. Do not ask questions about the assignment by email; if we receive questions by email where the response could be helpful for the class, we will ask you to repost the question on the discussion boards.