Introduction to Machine Learning (67577)

IML Hackathon 2017

Teacher: Prof. Shai Shalev-Shwartz
TA: Michal Moshkovitz, David Yakira, Yoav Wald

General Instructions

- The hackthon starts on Thursday, 22.6, 3pm, and ends on Friday, 23.6, 12pm.
- You may work in teams of one, two, three or four students.
- In the next page you will find two learning tasks. Choose one of them, and solve it as well as you can. All the materials for the tasks are available on the course moodle.
- To enable us to run your code without errors, we require all submissions to use `virtualenv` and define an environment for the project. Then submit a `requirements.txt` file, generated using the `freeze` command, that summarizes the environment you use. Instructions on and the `virtualenv` and the `freeze` command can be found here.
- Submit through the course moodle a tar file named `iml_hackathon.tar` (using one of the accounts in the team) that includes:
  - All your code and and supplementary files. Notice that each task may have different code requirements. Follow those carefully, as there will be automatic tests. Make sure to follow best coding practices, including good documentation of your code.
  - `README` – contains a file list and a brief description of each file.
  - `USERS` – contains the logins and IDs of all the team members. Use one line per student, in the format `login, ID`.
  - `project.pdf` – a written description of your project, up to 2 pages long, as a PDF file. Explain your solution, describe your work process, and do your best to justify any design decisions you have made. Feel free to include supporting figures if you want.
  - `requirements.txt` – the required packages for your virtual environment, as described earlier.
- We advise that you start by finding a basic solution that works. Then try to improve it as much as you can, until you're out of time.

Good Luck!

1 Whose Headline Is It Anyway?

Task: You are given a dataset of newspaper headlines from “Haaretz” and “Israel Hayom” in English. Your task is to learn a predictor that takes a headline and returns which newspaper published it. The test set will be taken from newspaper headlines published after the hackathon is over.

Dataset: The headlines can be found in `haaretz.csv` and `israelhayom.csv` , where each record corresponds to a single headline. You can load the dataset using the script `load_headlines.py`.
Note: The training set in imbalanced, as it contains more headlines from “Israel Hayom” than “Haaretz”, but the test set will be balanced.

Code Requirements: You need to implement the class `Classifier` in the module `classifier`. This class has a method `classify` that receives a list of headlines and predicts for each one which newspaper published it. Do not change the signature of this method.

הארץ ישראל היום

2 Spelling Correction

Task: You are given a set of typing and spelling mistakes made by a user typing on a standard (QWERTY) keyboard.
Your task is to learn how to correct future mistakes this user will make. In our case the “user” is a simulator that gets a correctly spelled word and randomly corrupts it with:

- Typing mistakes such as pressing an adjacent key (e.g “t” instead of “g”), omitting a key etc.
- Spelling mistakes like omitting the second letter in a double-letter (e.g “gren” instead of “green”), substituting a vowel for a different one (e.g “translatur” instead of “translator”) and more . . .

The test set will be consisted of newly generated mistakes from the same simulator, corrupting the same words it did in the training set.

Dataset and Output: The training set can be found in the file `data_50000.pickle`. It contains tuples where the first element is a correctly spelled word and the second is a misspelled version of the word, as generated by the simulator.
We also include the file `dictionary_5000.pickle` which lists the unique (correctly spelled) words that appear in the training set. The items in this list are tuples of size 2, first item being the *word’s index* and the second the word itself. The test set will consist of misspellings of these words, so we can look at the problem as a multi-class problem with 5000 classes. Labels are then given by the word’s index and are in the range $[0, 4999]$.

Code Requirements: You may predict **3 candidates** for each misspelled word, a prediction will be considered correct if the correct word is in the set of candidates. Implement the class `Classifier` in the module `classifier`. This class has a method `classify` that receives a list of misspelled words and returns a numpy matrix of size $m \times 3$ with ints in the range $[0, 4999]$, representing the predicted candidates. Do not change the signature of this method.