PYTHON HINTS, TRICKS, AND STYLE (V.1.)

COMPUTER SCIENCE 187

Read through this handout prior to working on the first assignment.

1. Getting Started

We will be using Python 2.7.3 for CS 187. If you are completely new to Python, take a look at Google's Python Class, which provides a quick introduction: `https://developers.google.com/edu/python/`. A more extensive tutorial is available at `http://docs.python.org/2/tutorial/`. The official Python documentation will also be a useful reference throughout the semester: `http://docs.python.org/2/`.

2. Using Python in the Appliance

2.1. **EPDFree.** We have installed EPDFree, which is the free Enthought Python distribution[1], in the Appliance. It contains a number of useful libraries for scientific computing, including, among others, tools for generating plots (Matplotlib) and a language extension for efficient array manipulation (NumPy). In a future handout, we will provide hints on how to generate plots. Unless we specify otherwise, use of these external libraries is not required to complete the assignments. In particular, you are not required to use NumPy arrays; the built-in Python data structures (e.g., dictionaries, lists, sets, etc.) will work well for most of our tasks.

[1]For future use, if you would like to install this on your home machine outside of the Appliance, note that you can get the full Enthought distribution for academic use at `http://www.enthought.com/products/edudownload.php`. The version in the Appliance, EPDFree, is smaller but has everything we will need for the course.

2.2. **Komodo Edit.** For a Python editor, we have installed Komodo Edit 7, which has an icon on the Desktop. Feel free to use your preferred editor, but we've found this editor to work well. We've added a run script macro[2], so once you have a Python file open in Komodo, just hit F5 and the script will run in the console window in Komodo. After executing the script, the interpreter will be in interactive mode in the console window, which can be quite useful for debugging.

2.3. **Terminal.** You can also start an interactive mode by typing

```
% python
```

in a terminal window. From a terminal window, you can run a Python script[3] by typing

```
% python NAME_OF_SCRIPT.py
```

3. Coding Style

3.1. **Consistency.** Prior to embarking on a particular assignment, take time to consider the style conventions you will use in your code, and once you start coding, be consistent throughout the assignment. The following are good starting points:

(1) PEP 8 – Style Guide for Python Code `http://www.python.org/dev/peps/pep-0008/`
(2) Google Python Style Guide `http://google-styleguide.googlecode.com/svn/trunk/pyguide.html`

3.2. **Whitespace.** With Python, since whitespace is significant, it is particularly important to not mix tabs and spaces. Unless you have a compelling desire to do otherwise, we would recommend that you

[2]This run script is just a few short lines of JavaScript, curtesy of the following reference, which is useful in case you want to add this to a future install: `http://www.cs.unc.edu/~gb/Comp116Spring2011/blog/2011/02/09/adding-a-run-command-to-komodo/`. Note that if you install Enthought, you will have to be careful to specify the correct Python path.

[3]It is also possible to make these scripts executable if you provide the proper shebang header, but you do not need to do that for the assignment scripts.

use four spaces for indentation. By default, Komodo will insert four spaces when you hit tab in a Python script.

3.3. **Documentation.** Documentation is so important that it deserves its own full, top-level section below for emphasis.

4. Documentation

While you are generally free to (consistently) employ your Python coding style of choice, we ask that you use the following particular convention for your Python Documentation Strings (a.k.a. "doc strings"). A Python doc string is the first statement (in our case, denoted by three double-quotes) of a class or method/function that describes the syntax and use of the applicable class or method/function. (Implementation details should be specified with in-line comments, as opposed to the doc string.) We ask that you include a doc string for every class and method/function that you include in your assignment scripts. The first line should be a short descriptive phrase. For functions/methods, also include lines titled Args, Returns, and Raises. For classes, include lines for Attributes. This convention is demonstrated in the following functions (with implementation code omitted):

```
def gen_features(vocab, reg_tokenizer, training_flag):
    """
    Generate list of binary feature vectors (lists)

    Args:
        vocab: dictionary mapping vocabulary items to unique indices
        reg_tokenizer: regular expression object
        training_flag: True for generating training features;
            False for generating test features
    Returns:
        list of lists containing binary feature vectors for each
            training/test review
    Raises:
        Error if the training/test directory/files are inaccessible
    """

def train_perceptron(features, vocab_size, labels, num_iter=10, alpha=1):
```

```
    """
    Train the perceptron model

    Args:
        features: list of lists containing binary feature vectors
            for each training review
        vocab_size: size of the vocabulary
        labels: list of labels for the training set
        num_iter: number of iterations
        alpha: step size/learning rate
    Returns:
        List of weights
    Raises:
        None
    """
```

5. Additional Python Highlights

5.1. **Importing Python Modules.** The outermost statements in a Python "module", which is a synonym for "script" or "file" in our usage, are executed whenever the module is imported or run directly. As such, it is common to use the following boilerplate code to control what code is executed on an import, as opposed to the result of running the module directly (as via the command line in a terminal window):

```
def main():
    # the code included here will be executed when the module
    # is run directly, but not when the module
    # is imported

if __name__ == "__main__":
    main()
```

We ask that you organize your code using this convention for the purposes of the assignments.

5.2. **Built-in Methods and Functions.** Python has a number of built-in methods and functions that are quite useful for string and list processing. Be sure to familiarize yourself with, among others, the string

methods strip(), split(), join(), find(), and lower(). The built-in functions sorted(), zip(), range(), float(), sum(), max(), len(), and open() are also worth checking out. If you're new to the language, figuring out what these do is a good place to start. Consult the Python Documentation for more information, and experiment in interactive mode with the Python interpreter in Komodo or in a terminal window.

6. Submitting Python Assignments

All assignments will be run in the Appliance for evaluation, so if you decide to develop in another environment, ensure that your Python code runs in the Appliance prior to submitting on iSites.