Assignment 2 (Due: April 9) (No late submissions accepted) (10 pts)

In this project, you will implement several different classifiers in Python manually as well as using **scikit-learn** APIs. The project has two phases. In Phase I, you will build classifiers and run them on the same modified 20 Newsgroup dataset. In Phase II, you will evaluate these classifiers to elucidate their quality of predictions (e.g., using accuracy and confusion matrix).

The project may be done in a team of two or three members like before, to promote discussions and insights. All the team members are expected to contribute to all aspects of the project: design, implementation, documentation, and testing, for their own good.

Phase I

scikit-learn provides a mature set of APIs for building models using regression, classification and clustering techniques, and has been used extensively for prediction tasks.

Classification on Newsgroups

For this project, you will use a subset of the 20 Newsgroups dataset. The full data set contains 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups and has been used for experiments in text applications of machine learning techniques, such as text classification and text clustering. This assignment dataset contains a pre-processed subset of 1000 documents and a vocabulary (dictionary) of 5,500 terms. As you are already familiar with the text pre-processing pipeline for parsing and converting text into sequence of terms, we are providing associated term-document matrix representation of the dataset as input. Each document belongs to one of two classes: Hockey (class label 1) and Microsoft Windows (class label 0). The data has already been split (80%, 20%) into training data and test data. The class labels for training data and test data are also provided in separate files. The training data and test data, in `term x document` format, contains a row for each term in the vocabulary and a column for each document. The values in the table represent raw term occurrence counts. The data has already been preprocessed to extract tokens, remove stop words and perform stemming. So, the terms in the vocabulary are stems, not full terms. Consult the `readme.txt` file in the distribution for details.

Your task is to exercise several different classifiers available in **scikit-learn** on the given 2Newsgroups dataset, and code one of them manually. Specifically, you must manually code K-Nearest-Neighbor (kNN), and use Naive Bayes (NB) and Support Vector Machine (SVM) classifiers via APIs. You may additionally use Pandas, NumPy, standard Python libraries, and Matplotlib in your programs.

Milestones for Phase I

- Read and understand the input dataset format.
- Implement kNN from scratch manually. Now exercise kNN, and using scikit-learn APIs in Python, Naive Bayes (Multinomial) and SVM classifiers on the given dataset.

Phase I resources

- Dataset: `2Newsgroup.zip`
- Relevant tutorials and descriptions
  - https://scikit-learn.org/stable/supervised_learning.html
  - https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html
  - https://scikit-learn.org/stable/datasets/real_world.html

Phase I environment setup

- Python 3 or higher
- Install scikit-learn:
  - https://www.activestate.com/resources/quick-reads/how-to-install-scikit-learn/
- Name your well-documented Python script as `assignment2.py`.
- You are required to use a Python virtual environment. For those unfamiliar with Python virtual environments here are two reference articles explaining how to use and activate:
  - https://uoa-eresearch.github.io/eresearch-cookbook/recipe/2014/11/26/python-virtual-env/
  - https://realpython.com/lessons/creating-virtual-environment/
- When you are ready to submit the completed assignment to Pilot, you need to create a `requirements.txt` file listing all the libraries used in your development. Use the following command only after you have activated your Python environment.
  - `source venv/bin/activate`
  - `pip list --format=freeze  requirements.txt`
  - `deactivate`
  - https://openclassrooms.com/en/courses/6900846-set-up-a-python-environment/6990546-manage-virtual-environments-using-requirements-files

Phase II

Briefly discuss the evaluation metrics used to elucidate and quantify the quality of classifier predictions.

Milestones for Phase II

- For each classifier, at the minimum, provide confusion matrix and accuracy.
- Compare and contrast K-Nearest-Neighbor (kNN), Naive Bayes and SVM.

Deliverables

TURN IN: Upload one tar archive file per team that contains the following files created during Phases 1 and 2:

Code and accompanying documentation: Include well-documented source code for the entire project.

Evaluation information: Provide comparative analysis of the different classifier performance using the chosen evaluation metrics.

ReadMe.txt: Briefly explain your application, how to launch the application, any external libraries used, all team members names and UIDs, and any other relevant information. The more information you provide in this document the easier it is to grade and give partial credit if something does not work. *Make sure to include names and email addresses of all team members.*

Input/Output files: Read the input data in the order given: `trainMatrixModified.txt, trainClasses.txt, testMatrixModified.txt, testClasses.txt, modifiedterms.txt` and generate output data per classifier as: `kNN.txt, NB.txt, SVM.txt, ....`

Application execution: Make sure that your Python program takes the inputs in the order described, from the command line. Additionally, make sure all the outputs are saved in the appropriate (working) directory.

File Location: All file(s) including all generated output file(s) must be placed in your working directory. Make sure to keep verbatim copies of all the classifier outputs in the subdirectory `backup` in your home directory.

Caution: Points will be deducted if you do not follow the naming conventions and use standard file extensions, or if your program does not run in the terminal/command line. **Use/monitor *discord* to seek clarifications and for updates.** If there are several small changes due to these discussions, I may even email an updated assignment consolidating all the changes.

TAR archive: Build the project file you submit using the following command to tar up your working directory:

- `tar -zcvf assignment2.tgz yourWorkDirectoryName/`

Uploading to Pilot: Upload the archive asg2.zip to Pilot > Dropbox > Assignment 2 folder by April 9. Upload only one submission per team making sure to list team member names and UIDs as mentioned earlier. You are also expected to demo your program to us if necessary and be prepared to answer questions about its design, implementation, and comparative evaluation.

Grading Criteria

You must obtain a PASS on this assignment to PASS the course. At the minimum, your code should compile, process the dataset, and return reasonable results for at least one classifier.