COL341 Spring 2023
Assignment 2: Support Vector Machines
(To be done Individually)

Due Date: 30th March 2023, Tuesday, 11:55 PM (No extensions)

Notes:

- Download the starter code and dataset from here.
- Use ***Dual Form*** of the SVM for the whole assignment.
- You should use `Python 3.7+` as your programming language.
- You are allowed to use only `numpy`, `pandas` and built-in python functions and libraries. No third-party library is allowed without explicit permission from the course instructor.
- For solving the Quadratic Programming, use `qpsolver` library. `qpsolver` supports various solvers, you may use the one of your choice.
`https://pypi.org/project/qpsolvers`
- For all parts, use the `Train split` for training your model, and the `Validation split` for evaluating the performance. For binary, and multi-class data, the appropriate splits are mentioned in the questions.
- For final evaluation, we will use a hidden test split which has not been provided to you.
- No extension will be provided. Only exception being medical or other emergencies. Such issues will be dealt in a case-to-case basis once notified to the Instructor.
- Use Piazza for questions and clarifications.
- Plagiarism and other academic integrity violations will lead to penalties as per the course and institute policies.

1 Dataset:

Both the datasets for Binary and Multiclass classification has a train-test-val split of 70-20-10. The test set is held out and final evaluation will be done on this split. The rows denote datapoints and the columns with numerical header are feature vectors and with the header 'y' is the label.

1.1 SVM Hyper-paramters

In general the following hyper-parameters are applicable for SVM.

1. *C:* The regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error. It is a positive scalar and can be tuned to achieve better performance on the training and validation sets.
2. *Kernel function:* The choice of kernel function affects the quality of the decision boundary. Usually the choice depends on the specific problem and the characteristics of the data.
3. *Kernel parameters:* Some kernel functions, such as RBF, have additional parameters (γ) that need to be specified. The choice of kernel parameters can have a significant impact on the performance of the SVM and should be tuned carefully.

2 Kernels [5]

Implement the following kernels for using kernel tricks along with SVMs in the `kernel.py`.

- Linear Kernel
- Polynomial Kernel
- Radial Basis Function (RBF) Kernel
- Sigmoid Kernel
- Laplacian Kernel

3 Binary SVM [35]

Use the dataset from `bi_train.csv` and `bi_val.csv` for this part.

3.1 Implementation [15]

Implement the standard soft margin support vector machine from scratch. Complete the `Trainer.fit()` and `Trainer.predict()` functions in `svm_sm.py`.

3.2 Analysis [20]

You need to run experiments using Linear and RBF kernels in `Trainer` class. Vary the C parameter for $C = \{0.01, 0.1, 1.0, 10.0\}$. Choose $\gamma = 0.1, 0.01, 0.001$ for RBF (for each C). Add the analysis regarding the performance in the `Validation set` for each hyper-parameter set. Plot the accuracy for each hyper-parameter values.

4 Multi-class Classification [20+20]

Use `multi_train.csv` and `multi_val.csv` for this part. You need to use the binary classifier to implement One-vs-One and One-vs-All classifiers by completing `Trainer_OVO` and `Trainer_OVA` classes in `svm_multiclass.py`. Show the analysis for the RBF kernel with $\gamma = 0.1$ and $C = \{0.1, 1.0\}$.

5 Competitive Part [20]

This section will be graded on the rank in the leader-board on the hidden test set. The top-k submissions will receive full marks and rest will receive diminishing marks. The exact scheme will be decided later.

For this part you have to create and return instances of in the given `best.py` by completing the `best_classifier_two_class()` and `best.py` by completing the `best_classifier_multi_class()` functions. In these, create the Trainer class instances with the best set of hyper-parameters that worked for you. The set of hyper-parameters involve C, *Kernel functions*, and *Kernel parameters*. You are free to choose any set of hyper-paramter including any kernel function $f(x)$ for both binary and multi-class classifiers. For the multi-class, you can use either the OVO or OVA trainer, whichever worked best for you.

6 Evaluation

Accuracy and the average of micro and macro F1 scores will be used as the metric for ranking. Try to get the best results on the validation set, the final results will be calculated on the held out test set for the competitive parts.

7 Submission Guidelines

1. Submit all source `.py` files - `best.py, kernel.py, svm_hm.py, svm_sm.py, svm_multiclass.py`.
2. Your code should have appropriate documentation for readability.
3. Submit a report in pdf file (`Report.pdf`). The report should contain the following:
   - Brief explanation of what you did for solving each part.
   - Confusion matrix on the `Validation` split for each part.
   - Best set of Hyper-parameters for each part.
   - Any additional observation/ analysis.
4. DO NOT submit the datasets.
5. Zip the report and source code files, and name the file `<your-entry-number>.zip`. Submit this codezip in Moodle.