COL 774: Assignment 2 (Semester I, 2022-23)

Due Date: 11:50 pm, Tuesday Oct 4, 2022. Total Points: 60

Notes:

- This assignment has three main parts - text classification using Naïve Bayes (Part 1), modified CIFAR10 binary classification using SVM (Part 2), modified CIFAR10 multi-class classification using SVM (Part 3). All parts will be due at the same time.
- You should submit all your code (including any pre-processing scripts written by you) and any graphs that you might plot.
- Do not submit the datasets. Do not submit any code that we have provided to you for processing.
- Include a **single write-up (pdf) file** which includes a brief description for each question explaining what you did. Include any observations and/or plots required by the question in this single write-up file.
- You should use Python for all your programming solutions.
- Your code should have appropriate documentation for readability.
- You will be graded based on what you have submitted as well as your ability to explain your code.
- Refer to the course website for assignment submission instructions.
- This assignment is supposed to be done individually. You should carry out all the implementation by yourself.
- We plan to run Moss on the submissions. We will also include submissions from previous years since some of the questions may be repeated. Any cheating will result in a zero on the assignment, a penalty of -10 points and possibly much stricter penalties (including a **fail grade** and/or a **DISCO**).

1. **(26 points) Text Classification**
   In this problem, we will use the Naïve Bayes algorithm for text classification. The dataset for this problem is the Large Movie Review Dataset and has been obtained from this website. Given a movie review, the task is to predict the sentiment of the review as either positive or negative. Read the website for more details about the dataset. You have been provided with a subset of the Large Movie Review Dataset, with the training and test splits containing 25,000 reviews (samples) and 15,000 reviews respectively. Data is available at this link. A review comes from one of the two categories (class labels) —positive or negative.

   (a) **(10 points)** Implement the Naïve Bayes algorithm to classify each sample into one of the given categories.

   i. Report the accuracy over the training as well as the test set.
   ii. Read about word cloud. Construct a word cloud representing the most frequent words for each class.

   Notes:

   - Make sure to use the Laplace smoothing for Naïve Bayes (as discussed in class) to avoid any zero probabilities. Use $\alpha = 1$, where α is the parameter which controls the strength of the smoothing.
   - You should implement your algorithm using logarithms to avoid underflow issues.
   - You should implement Naïve Bayes from the first principles and not use any existing Python modules.

(b) **(2 points)**

i. What is the test set accuracy that you would obtain by randomly guessing one of the categories as the target class for each of the reviews (random prediction)?

ii. What accuracy would you obtain if you simply predicted each sample as positive?

iii. How much improvement does your algorithm give over the random/positive baseline?

(c) **(3 points)** Read about the confusion matrix.

i. Draw the confusion matrix for your results in parts (a) and (b) above (for the test data only).

ii. For each confusion matrix, which category has the highest value of the diagonal entry? What does that mean?

iii. In each of the cases above, the entries in the confusion matrix have a specific pattern. Can you observe this pattern and explain the reasons behind it?

(d) **(4 points)** The dataset provided to you is in the raw format i.e., it has all the words appearing in the original set of articles. This includes words such as 'of', 'the', 'and' etc. (called stopwords). Presumably, these words may not be relevant for classification. In fact, their presence can sometimes hurt the performance of the classifier by introducing noise in the data. Similarly, the raw data treats different forms of the same word separately, e.g., 'eating' and 'eat' would be treated as separate words. Merging such variations into a single word is called stemming. Read about stopword removal and stemming (for text classification) online.

i. Perform stemming and remove the stop-words in the training as well as the test data.

ii. Construct word clouds for both classes on the transformed data.

iii. Learn a new model on the transformed data. Report the test set accuracy.

iv. How does your accuracy change over test set? Comment on your observations.

(e) **(5 points)** Feature engineering is an essential component of Machine Learning. It refers to the process of manipulating existing features/constructing new features in order to help improve the overall accuracy on the prediction task. For example, instead of using each word as a feature, you may treat bi-grams (two consecutive words) as a feature.

i. You can read here about Bi-grams. Use word based bi-grams to construct new features. You should construct these features after doing pre-processing described in part d(i) above. Further, these should be added as additional features, on top of existing unigram (single word) based features. Learn your model again, and report test set accuracy.

ii. Come up with at least one additional set of features to further enhance your model. Learn the model after doing pre-processing as described in part d(i) above, and report the test set accuracy.

iii. For both your variations tried above, compare with the test set accuracy that you obtained in parts (a) and parts (d). Do the additional set of features constructed in each case help you improve the overall accuracy? Comment on your observations.

(f) **(2 points)** Read about Precision, Recall and F1-score.

i. For your best performing model obtained above (from parts (a), (d) and (e)), report the precision, recall and F1-score.

ii. Which metric, test accuracy or F1-score, do you think is more suited for this kind of dataset? Why?

2. **(20 points) Binary Image Classification**

In this problem, we will use Support Vector Machines (SVMs) to build a binary image classifier. We will be solving the SVM optimization problem using a general purpose convex optimization package as well as using a scikit-learn library function based on a customized solver known as LIBSVM. This problem is based on the CIFAR-10 image dataset. The original dataset comprises of 50000 training samples and 10000 test samples, belonging to 10 classes. However, we will be using a subset of this, containing 10000 training samples and 5000 test samples belonging to 5 classes. Data is available at this link (part2_data.zip). There are two pickle files, one for train data and the other for test data. Each pickle file contains a dict with two keys, 'data' and 'labels'. Each sample is an RGB image of size $32 \times 32 \times 3$. For working with SVM's, you should flatten each image to create a vector of length 3072.

(a) **(8 points) Binary Classification:**

Let d be the last digit of your entry number. Take the subset of images for the classes d and (d+1)

mod 5) from the train/test data provided to you and perform the following experiments in the context of binary classification. Download and install the CVXOPT package. Express the SVM dual problem (with a linear kernel) in a form that the CVXOPT package can take. You will have to think about how to express the SVM dual objective in the form $\alpha^T P\alpha + q^T\alpha + c$ matrix where P is an $m \times m$ matrix (m being the number of training examples), q is an m-sized column vector and c is a constant. For your optimization problem, remember to use the constraints on α_i's in the dual. Use the SVM formulation which can handle noise and use $C = 1.0$ (i.e. C in the expression $\frac{1}{2}w^T w + C * \sum_i \xi_i$). You can refer this link to get a working overview of cvxopt module and it's formulation.

i. How many support vectors do you get in this case? What percentage of training samples constitute the support vectors?

ii. Calculate the weight vector w and the intercept term b and classify each of the examples in the test file into one of the two labels. Report the test set accuracy.You will need to carefully think about how to represent w and b in this case.

iii. Reshape the support vectors corresponding to the top-5 coefficients to get images of $32 \times 32 \times 3$ and plot these. Similarly, reshape and plot the weight vector w.

(b) **(6 points)** Again use the CVXOPT package to solve the dual SVM problem using a Gaussian kernel. Think about how the P matrix will be represented. Use $C = 1.0$ and $\gamma = 0.001$ (i.e. γ in $K(x, z) = \exp^{-\gamma * \|x-z\|^2}$) for this part.

i. How many support vectors do you get in this case as compared to the linear case above? How many support vectors obtained here match with the linear case above?

ii. Note that you may not be able to explicitly store the weight vector (w) or the intercept term (b) in this case. Use your learned model to classify the test examples and report the test accuracy.

iii. Reshape the support vectors corresponding to the top-5 coefficients to get images of $32 \times 32 \times 3$ and plot these.

iv. Compare the test accuracy obtained here with part (a).

(c) **(6 points)** Repeat parts -(a) & (b) with the scikit-learn SVM function, which is based on the LIBSVM package.

i. Compare the nSV (Number of Support Vectors) obtained here with part (a) for linear kernel and part (b) for Gaussian kernel. How many of the support vectors obtained here match with the support vectors obtained in both these cases?

ii. Compare weight (w), bias (b) obtained here with part (a) for linear kernel.

iii. Report the test accuracy for both linear and Gaussian kernel.

iv. Compare the computational cost (training time) of the CVXOPT with the sklearn implementation in both the linear and Gaussian case.

3. **(14 points) Multi-Class Image Classification:**
In this section, we will work with the entire subset of the data provided to you in question 2 focusing on a multi-class classification problem using SVMs. We will work with Gaussian kernel for this section.

(a) **(4 points)** In class, we described the SVM formulation for a binary classification problem. In order to extend this to the multi-class setting, we train a model on each pair of classes to get $\binom{k}{2}$ classifiers, k being the number of classes (here, k=5). During prediction time, we output the class which has the maximum number of votes from all the $\binom{k}{2}$ classifiers. You can read more about one-vs-one classifier setting at the following link. Using your CVXOPT solver from previous section, implement one-vs-one multi-class SVM. Use a Gaussian Kernel with $C = 1.0$ and $\gamma = 0.001$.

i. Classify the test examples and report test set accuracy. In case of ties, choose the label with the highest score.

(b) **(3 points)** Now train a multi-class SVM on this dataset using the scikit-learn SVM function, which is based on the LIBSVM package. Repeat part (a) using a Gaussian kernel with $\gamma = 0.001$. Use $C = 1.0$ as earlier.

i. Classify the test examples and report test set accuracy.

ii. How do the test set accuracy and the training time obtained here compare with part (a) above?

(c) **(4 points)** Draw the confusion matrix as done in Naive Bayes in this assignment for both of the above parts (2(a) and 2(b)). What do you observe? Which classes are miss-classified into which ones most often? Visualize (and report) 10 examples of mis-classified objects. Do the results make sense? Comment.

(d) **(3 points)** Validation set is typically used to estimate the best value of the model hyper-parameters (e.g., C in our problem with the Gaussian kernel) by randomly selecting a small subset of the training data as validation set and then training on the modified training data (original training data minus the validation set) and making predictions on validation set. For a detailed introduction, you can refer to this video. You can check the correctness of your intuition by trying this test. K-fold cross validation is another such technique in this regard that we use in practice. In this technique we divide our training data into K-folds or parts and then treat each part as our validation set once and train on the remaining K-1 parts. You can read more about cross validation here [1] (see Section 1) for more details. This process is repeated for a range of model hyper-parameter values and the parameters which give best K-fold cross validation accuracy are reported as the best hyper-parameters. We will use scikit-learn SVM function for this part.

For this problem, we will do a 5-fold cross validation to estimate the best value of the C parameter for the Gaussian kernel case. Test data should not be touched.

i. Fix γ as 0.001 and vary the value of C in the set $\{10^{-5}, 10^{-3}, 1, 5, 10\}$ and compute the 5-fold cross validation accuracy and the test accuracy for each value of C.

ii. Now, plot both the 5-fold cross validation accuracy as well as the test set accuracy on a graph as you vary the value of C on x-axis (you may use log scale on x-axis). What do you observe? Which value of C gives the best 5-fold cross validation accuracy? Does this value of the C also give the best test set accuracy? Comment on your observations.

[1] These are from Andrew Ng notes posted on the course website, and the link is available only from the internal IIT Delhi network. Feel free to read additional material online about this topic.