COL 341: Assignment 4 & 5

Notes:

- This assignment has three parts - Support Vector Machine, Decision Tree and K-Means. This assignment has a total weightage of 20%.
- You should submit all your code (including any pre-processing scripts written by you).
- You are advised to use vector operations (wherever possible) for best performance.
- Include any observations and/or plots required by the question in a report of maximum 4 pages . Report must be in **pdf format only**.
- You should use Python/R for all your programming solutions.
- Your assignments will be auto-graded, make sure you test your programs with **'autochecker'** before submitting. If your code fails during autochecking, you will be penalized 40% for correction. We will use your code to train the model on training data and predict on test set.
- Input/output format, submission format and other details are included on 'README.md'. Your programs should be modular enough to accept specified parameters.
- You should submit work of your own. You should cite the source, if you choose to use any external resource. You will be awarded **F grade or DISCO** in case of plagiarism.
- You can use total of 7 buffer days across all assignments.

1. **Support Vector Machine (100 points, Release date: Oct. 13, 2018, Due date: Oct. 26, 2018)** In this problem, we will use Support Vector Machines (SVMs) to build a Devanagari Handwritten Characters classifier. We will be solving the SVM optimization problem using a general purpose convex optimization package. We will use a subset of the dataset containing only 2 classes of Devanagari characters available for download from this link. You have been provided a CSV data file. Each row in the data file corresponds to an image of size 32x32. First entry of each row corresponds to the label(0 or 1) associated with the image followed by the vector of gray-scale pixel intensities of the image.

   (a) **(30 points)**Download and install package CVXOPT for Python and package quadprog for R. Express the SVM dual problem (with a linear kernel) in the a form that the CVXOPT/quadprog package can take. You will have to think about how to express the SVM dual objective in the form $\alpha^T Q\alpha + b^T\alpha + z$ matrix where Q is a $n * n$ matrix (n being the number of training examples),b is an n-sized column vector and z is a constant. For your optimization problem, remember to use the constraints on α_i s in the dual. Use the SVM with soft-margin formulation. The SVM optimisation formulation can be written as:

   $$\min_{w,b} \frac{1}{2} w^T w + C * \sum_{i=1}^{n} \varepsilon_i \tag{1}$$

   subject to $y_i(w^T x_i + b) \geq 1 - \varepsilon_i$,$\varepsilon_i \geq 0$ for all i. Other symbols are as described in the class. Use your learned model to classify the test data. Experiment for various values of C in the set {0.01, 0.1, 1, 10, 100}. Include your observations in report.

   (b) **(20 points)** Now solve the dual SVM problem using a RBF kernel (Gaussian kernel) with the bandwidth parameter γ ,i.e., γ in $K(x, z) = exp^{-\gamma * ||x-z||^2}$. Think about how the matrix Q will be represented. Use your learned model to classify the test data. Experiment for various combinations of C in the set {0.01, 0.1, 1, 10, 100} and γ in the set {0.01, 0.1, 1, 10, 100}. Include your observations in report.

(c) **(35+5 points)**Principal component analysis (PCA) is a technique used to emphasize variation and bring out strong patterns in a dataset. You have to apply PCA on the given dataset to reduce the dimensionality and then use the projected data for the task of training and classification. You should not normalize the data since each pixel value comes from the same scale (0-255).

i. Implement the PCA algorithm using the SVD formulation. You should not call any function to perform PCA directly but rather do it explicitly using SVD. You should be able to perform SVD in Python/R using existing libraries. Note that since data is not normalized to zero mean, you might have to change the co-variance matrix calculation accordingly. Calculate the principal components corresponding to top 50 eigenvalues. For each image in the dataset, project it on to the top 50 dimensions obtained using PCA. Then apply SVM with RBF kernel on the projected data to learn model(as in part b) and classify the test data.

ii. Display the eigenfaces corresponding the top five principal components. You might have to scale the pixel values to display eigenfaces so that maximum value touches 255. Include the eigenfaces in your report.

(d) **(10 points)** Compare the test accuracy obtained by using linear kernel and RBF Kernel with PCA and without PCA. Include your observations in the report.

Evaluation:

- For part-a, part-b, part-c(i), you can get 0 (error), half (code runs fine but predictions are incorrect within some predefined threshold) and full (works as expected).
- For part-c(ii), you can get 0 (wrong plot) and full (plot is as expected).
- For part-d, You can get full (correct observation) or 0 (wrong observation)

2. **Decision Tree (50 points, Release date: Oct. 13, 2018, Due date: Oct. 26, 2018))**
In this problem, you will work with the Adult Dataset available on the UCI repository. Read about the dataset in detail from the link given above. For the purpose of this assignment, you have been provided with a subset of this dataset available for download from this link. You have been also provided a file which specifies which attributes are discrete and which are continuous on the link. Specifically, you have been provided with separate training, validation and testing sets. The first row in each file specifies the type of each attribute. The first entry in each row denotes the class label. You can encode the discrete non-numeric attributes to some numeric value. You have to implement the decision tree algorithm for predicting whether a person is rich (1) or not (0) based on various personal attributes.

(a) **(25+5 points)** Construct a decision tree from first principle for the above prediction problem. You should not use any library available for decision tree. Preprocess (before growing the tree) each numerical attribute into a Boolean attribute by a) computing the threshold value of the attribute in the training data, refer to section 3.7.2 of Machine Learning by Tom Mitchell b) replacing each numerical value by a 1/0 value based on whether the value is greater than the threshold value or not. Note that this process should be repeated for each attribute independently. For non-Boolean (discrete valued) attributes, you should use a multi-way split. Use Gain Ratio as the criterion for choosing the attribute to split on as explained this link . In case of a tie, choose the attribute which appears first in the ordering as given in the training data. Plot the train, validation and test set accuracies against the number of nodes in the tree as you grow the tree. On X-axis you should plot the number of nodes in the tree and Y-axis should represent the accuracy.

(b) **(15+5 points)** One of the ways to reduce over-fitting in decision trees is to grow the tree fully and then use post-pruning based on a validation set. In post-pruning, we greedily prune the nodes of the tree (and sub-tree below them) by iteratively picking a node to prune so that resultant tree gives maximum increase in accuracy on the validation set. In other words, among all the nodes in the tree, we prune the node such that pruning it(and sub-tree below it) results in maximum increase in accuracy over the validation set. This is repeated until any further pruning leads to decrease in accuracy over the validation set. Post prune the tree obtained in step (a) above using the validation set. Again plot the training, validation and test set accuracy against the number of nodes in the tree as you successively prune the tree.

Evaluation:

- Marking will be done in two parts: code (80%) and plot(20%).

- For code: you can get 0 (error), half (code runs fine but predictions are incorrect within some predefined threshold) and full (works as expected).
- For plot: you can get 0 (wrong plot),half (plot is partially correct) and full (plot is as expected).

3. **K-Means Clustering (50 points, Release date: Oct. 31 2018, Due date: Nov. 7 2018)**
In this problem, we will revisit PCA and use K-Means to build a Devanagari Handwritten Characters classifier. We will be using UCI dataset (link) we have provided you with 2 files test and train. Each row in train consists of pixels corresponding to 32x32 image with first element of each row as labels. Each row in test consists of pixels corresponding to 32x32 image with no labels.

   (a) **(25 points)**Implement K-Means to cluster Train data (D) with features $X \in R^{|D| \times M}$ into $C(= C_1, C_2, .., C_n)$ categories. Minimize the given objective function using EM algorithm to get the cluster centers $U = U_1, U_2, .., U_n$

$$\min_{U} \sum_{i=1}^{|D|} \sum_{j=1}^{|C|} 1\{C_i == j\} ||X_i - U_j||_2 \tag{2}$$

   Do not use labels for clustering the data and use majority of class label in a cluster as cluster label. Mention your initialization method (random/k-means++) [Run your code multiple times (at least 5) in case of random and report the values.], cluster quality (NMI, entropy, purity) (link) and convergence criteria in the report. Use the following equation to assign a label to an unknown test point.

$$j^*_{test} = \min_{j \in 1,..,n} ||X_{test} - U_j||_2 \tag{3}$$

   Report test set accuracy in the report.

   (b) **(15 points)** In this question We will use a subset of the dataset containing only 2 classes of Devanagari characters available for download from this link (same as given in SVM). Use your implementation of PCA to reduce the number of dimension of X and try K-Means on these feature space again. Experiment with top 10, 50, 100, 200 principal components and report the results.

   (c) **(10 points)** Compare the test accuracy obtained by K-Means using all the features and K-Means with PCA and best values obtained from SVM on the dataset given in part b. Include your observations in the report.

Evaluation:

- You will be evaluated on time required for your implementation to work.
- Score of part a will be used to evaluate your code.
- Score of part b on PCA components 200 will be used to evaluate your code.