IITM-CS5011 : Introduction to Machine Learning	Given on: Oct 01, 10pm
Programming Assignment #2	Due on : Oct 16 , 5pm

- The goal of this assignment is to experiment with feature extraction methods and classification methods.
- This is an individual assignment. Collaborations and discussions with others are strictly prohibited.
- You may use Matlab, Octave or Python for your implementation. If you are using any other languages, please contact Harini before you proceed.
- You have to turn in the well documented code along with a detailed report of the results of the experiment electronically in Moodle. Typeset your report in Latex.
- Be precise for your explanations in the report. Unnecessary verbosity will be penalized.
- You have to check the Moodle discussion forum regularly for updates regarding the assignment.

---

1. You have been provided with training instances for an image classification problem (DS2). You have to train an SVM to classify the test images into either of the following four categories: coast, forest, inside-city, mountain.

   Follow the instructions below for extracting the features from the images.

   Instructions for Feature Extraction

   You are given a set of scene images. In this step, the requirement is to extract some features from the images that can be used as input to our SVM. There are many feature extraction techniques. For this assignment, we will follow a color histogram based approach. This is not the best technique for feature extraction, but most likely, the easiest.

   Matlab-

   1. Read the image into a variable using imread(), e.g. im = imread(filename).
   2. Extract red, green and blue channels from the variable you read into in 1. The sequence is r-g-b, e.g. r = im(:,:,1).
   3. For every channel divide it into 32 bins and find frequency using imhist(), e.g. f1 = imhist(r,32).
   4. Concatenate these 32 dimensional feature vectors for every channel to find a 96D vector for the whole image. (sequence r-g-b)
   5. Normalize the features before using them.

Python-

Use the Python Image Library. Load the image, use the histogram function to convert it into a pixel count list (histogram). An RGB image will produce a $3 \times 256 = 768$ length vector hist[0:255] corresponds to R, hist[256:511] corresponds to G and the rest correspond to the B channel. Re-bin these into 32 bins for each of R,G,B (aggregate appropriate bins).

1. import Image
2. img = Image.open(imagepath)
3. hist = img.histogram() – to get the histogram
4. Re-bin by aggregating to get a 96D list (histogram)
5. Normalize the features before using them.

Use the training data to build classification models using the following kernels.

1. Linear kernel
2. Polynomial kernel
3. Gaussian kernel
4. Sigmoid kernel

Come up with the kernel parameters for the various models. You can use a fraction of data supplied to do a n-fold cross validation to find the best model parameters.

Important Notes:

1. You have to use libsvm in matlab.
2. Name the models as modelx, where x is the number of the corresponding model given above, e.g., model1
3. Put only these models in a single .mat file, name it as your roll no.mat, and submit it, e.g., CS11S016.mat (roll no, in uppercase)
4. Please do not jumble up the r-g-b sequence while building the feature vectors or the modelx while building the classifiers.
5. We are planning to automate evaluation of this question. And hence not following with the conventions might result in undesired evaluation results.

2. Implement original back-propagation algorithm. Use DS2 for training your neural network. Report per-class precision, recall and f-measure on the test data used in Question-1. Now consider the following alternate error function for training neural networks.

$$R(\theta) = \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K}(y_{ik} - f_k(x_i))^2 + \gamma(\sum_{k}\sum_{m}\beta_{km}^2 + \sum_{m}\sum_{l}\alpha_{ml}^2)$$

where N is the number of training instances, K is the number of output features, $f_k(x)$ is the predicted output vector, y is the original output vector, α and β are the weights and γ is a regularization parameter. Derive the gradient descent update rule for this definition of R. Now train your neural network with this new error function. Report per-class precision, recall and f-measure on the same test data. What will happen when you vary the value of γ ? Vary the value of γ from 10^{-2} to 10^2 in multiples of 10 and repeat the experiment and report the results. Can you figure out the effect of γ in the results? Look at the weights learnt using the new error function. What do you infer from them?

3. For this experiment, use only forest and mountain classes in DS2. Perform 2-class Logistic Regression on it. Report per-class precision, recall and f-measure on the same test data you used to test the neural net work. Now perform L_1-regularized Logistic Regression on the same dataset and report similar performance results. Use *l1_logreg* code provided by Boyds Group (`http://www.stanford.edu/~boyd/l1_logreg/`).

4. We have discussed about Linear Discriminant Analysis(LDA) in the class. We will see how different variants of this technique works. For this experiment, you have to use Iris Dataset (`http://archive.ics.uci.edu/ml/datasets/Iris`). Use only petal width and petal length features and perform LDA. Visualize the boundaries learnt. Also read about Quadratic Discriminant Analysis (QDA) and Regularized Discriminant Analysis(RDA) from the text book. Do QDA and RDA on the same data set and visualize the boundaries. You have to submit all the three plots. Please refer to section of 4.3 of Elements of Statistical Learning.

Using external libraries

- You can use PMTK (`https://code.google.com/p/pmtk3/`) for PCA, LDA and QDA.
- Use LIBSVM (`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`) for SVM.
- For L1-regularized Logistic Regression use the code provided by Stephen Boyds group. Link is provided in the question. You should NOT use any other external libraries or toolkit.
- If you are using Python, then you can use PCA, LDA, QDA, and libsvm in sklearn package.

Submission Instructions

Submit a single tarball/zip file containing the following files in the specified directory structure. Use the following naming convention: 'cs5011_a2_rollno.tar.gz'.

cs5011_a2_rollno

Dataset

DS2_train.csv
DS2_test.csv

Report

rollno-report.pdf

Code

all your code files