Project 4
CS 6375: Machine Learning
130 Points

Vibhav Gogate

1 K-means clustering on images [30 points]

You can use either Java or Python to implement this part.

In this problem, you will use K-means clustering for image compression. We have provided you with two images.

- Display the images after data compression using K-means clustering for different values of K (2, 5, 10, 15, 20).
- What are the compression ratios for different values of K? Note that you have to repeat the experiment multiple times with different initializations and report the average as well as variance in the compression ratio.
- Is there a tradeoff between image quality and degree of compression. What would be a good value of K for each of the two images?

We have provided you a java template "KMeans.java" which implements various image input/output operations. You have to implement the function kmeans in the template. If you want to use python, you will have to replicate the code in KMeans.java in python (will take roughly 10-15 minutes). Note that you cannot use scikit learn implementation of k-means for this part.

What to Turn in for Part 1

In a single zip file (part1.zip):

- Your source code for the kmeans algorithm.

- A PDF report containing your write up.

Note that your program must compile and we should be able to replicate your results. Otherwise no credit will be given.

2 Tractable Probabilistic Models (100 points)

In this part, you will implement the following three algorithms and test their performance on the 10 datasets available on MS Teams. Each dataset has three files: ".ts.data" (training data); ".valid.data" (validation data) and ".test.data" (test data).

1. **Tree Bayesian networks.** (20 points) Use the Chow-Liu algorithm to learn the structure and parameters of the Bayesian network. Use 1-Laplace smoothing to ensure that you don't have any zeros when computing the mutual information as well as zero probabilities in the model. See section 2 in [Meila and Jordan, 2001]. An implementation of Chow-Liu tree is provided in the code base provided to you.

2. **Mixtures of Tree Bayesian networks using EM.** (40 points) The model is defined as follows. We have one latent variable having k values and each mixture component is a Tree Bayesian network. Thus, the distribution over the observed variables, denoted by $\mathbf{X}$ (variables in the data) is given by:

$$P(\mathbf{X} = \mathbf{x}) = \sum_{i=1}^{k} p_i T_i(\mathbf{X} = \mathbf{x})$$

   where p_i is the probability of the i-th mixture component and T_i is the distribution represented by the i-th Tree Bayesian network. Write code to learn the structure and parameters of the model using the EM-algorithm (in the M-step each mixture component is learned using the Chow-Liu algorithm). Run the EM algorithm until convergence or until 50 iterations whichever is earlier. See section 3 in [Meila and Jordan, 2001]. Use the following values for $k \in \{2, 5, 10, 20\}$. Test performance using the "test set."

   In the code provided, see file MIXTURE_CLT.py, you have to write two functions "learn(..)" and "computeLL(...)"

3. **Mixtures of Tree Bayesian networks using Random Forests.** (40 points) The model is defined as above (see Item (3)). Learn the structure and parameters of the model using the following Random-Forests style approach.

Given two hyper-parameters (k, r), generate k sets of Bootstrap samples and learn the i-th Tree Bayesian network using the i-th set of the Bootstrap samples by randomly setting exactly r mutual information scores to 0 (as before use the Chow-Liu algorithm with r mutual information scores set to 0 to learn the structure and parameters of the Tree Bayesian network). Select k and r using the validation set and use 1-Laplace smoothing. You can either set $p_i = 1/k$ for all i or use any reasonable method (reasonable method is extra credit). Describe your (reasonable) method precisely in your report. Does it improve over the baseline approach that uses $p_i = 1/k$.

Report Test-set Log-Likelihood (LL) score on the 10 datasets available on MS Teams. For EM and Random Forests (since they are randomized algorithms), choose the hyper-parameters (k and r) using the validation set and then run the algorithms 5 times and report the average and standard deviation. Can you rank the algorithms in terms of accuracy (measured using test set LL) based on your experiments? Comment on why you think the ranking makes sense.

What to turn in for Part 2

In a single zip file (part2.zip):

- Your report in PDF format describing your experimental evaluation.
- Code along with a Readme file on how to compile and run your code.

References

[Meila and Jordan, 2001] Meila, M. and Jordan, M. I. (2001). Learning with mixtures of trees. *Journal of Machine Learning Research*, 1:1–48.