STA 250 **Optimization** **Fall 2017**

Final Exam (Project)

Lecturer: Cho-Jui Hsieh

Note:

1. Final exam due on Dec 13, 11:59pm (submit the answers and the code online).

Problem 1. Robust PCA

In the non-convex robust PCA problem, given an input matrix $M \in \mathbb{R}^{m\times n}$, we need to solve the following optimization problem:

$$\begin{aligned} \min_{W\in\mathbb{R}^{m\times k}, H\in\mathbb{R}^{n\times k}, S\in\mathbb{R}^{m\times n}} \quad & \frac{1}{2}(\|W\|_F^2 + \|H\|_F^2) + \lambda\|S\|_1, \\ \text{s.t.} \quad & WH^T + S = M. \end{aligned} \tag{1}$$

where $\|S\|_1 = \sum_{i,j} |S_{ij}|$ is the element-wise ℓ_1 norm. Note that k is small (we fix $k = 10$ in this problem), so we expect to decompose the matrix M into the low-rank part WH^T and the sparse part S.

1. (20 pt) Develop an algorithm for solving this problem. Describe your algorithm. What's the time complexity of your algorithm?

2. (20 pt) Generate synthetic data with $n = m = 100$ and $k = 10$ to test your algorithm. Generate W^*, H^* from Gaussian random. Generate S^* to be a sparse matrix with uniform random nonzero entries, and random values for each entry. Generate $M = S^* + W^*(H^*)^T$ and solve the robust PCA problem (1). How well can you recover the low-rank matrix $W^*(H^*)^T$ and the sparse matrix S^*? Try to choose a good parameter λ that can better recover the matrices. Report your findings.

3. (5 pt) Plot time vs objective function figures to see how fast your optimization algorithm converges.

4. (5 pt) Test your algorithm on a real dataset "mnist" (choose a parameter λ which you think gives you better result). Report the final objective function value and discuss your findings.

Problem 2. Extreme classification

In the mutli-label classification problem, given the data matrix $X \in \mathbb{R}^{n\times d}$ (each row is an input data point) and label matrix $Y \in \mathbb{R}^{n\times L}$. L is number of labels, and each row of Y is an L-dimensional 0/1 vector indicating the labels for a data point. We want to predict the label for a given new input data point. In "extreme" multi-label classification, number of labels can be extremely large (e.g., 10,000, or 1 million). Let's develop an algorithm for solving this problem.

We will test our algorithms using the dataset from
`http://manikvarma.org/downloads/XC/XMLRepository.html`.

We solve the following optimization problem to get the model W, H:

$$\min_{W\in\mathbb{R}^{d\times k},H\in\mathbb{R}^{L\times k}} \frac{1}{2}\|Y - XWH^T\|_F^2 + \lambda\|W\|_F^2 + \lambda\|H\|_F^2$$

For this problem we set $k = 50$. After solving the optimization problem, for each testing data $\boldsymbol{x} \in \mathbb{R}^d$, we predict the label $\tilde{y} \in \mathbb{R}^L$ by

$$\tilde{y} = \boldsymbol{x}^T WH^T$$

This is supposed to be close to the true label vector y since we minimize the square loss $\|XWH^T - Y\|_F^2$ in the objective function.

We will evaluate the results using precision@1 and precision@5, where precision@k is

$$\text{(number of true labels in the top-}k\text{ predictions)}/k.$$

Or formally,

$$\text{P@k} := \frac{1}{k}\sum_{l\in\text{rank}_k(\tilde{y})} y_l,$$

where $\text{rank}_k(\tilde{y})$ returns the k largest indices of the predictive label vector $\tilde{y}$, and $y_l = 1$ if it is a correct label.

There will be multiple testing samples, so the P@1, P@5 will be the average among those samples.

1. (10pt) Download the bibtex data. Transform the training data (only samples with training indices) into data matrix X and label matrix Y. Note that in "Bibtex_trSplit.txt" and "Bibtex_tstSplit.txt" there are 10 splits of training and testing data. We will only use the first split (first column in both files) to conduct the experiments.

2. (25pt) Develop an algorithm for solving this problem. Describe your algorithm. What's the time complexity?

3. (15pt) Apply your algorithm to the Bibtex data. Test your algorithm and compute P@1, P@5 of your algorithm. Test different λ values and compare the results. What's your finding? Also, you can compare your results with the state-of-the-art performance on the website `http://manikvarma.org/downloads/XC/XMLRepository.html` (Table 6).

4. (Bonus, 20pt)Try to scale your algorithm to a larger dataset AmazonCat-13K. Report your findings. If you can't scale to that data, you can still get partial credit by explaining why you are not able to do that (which operations in your algorithm are too slow for larger datasets), and what's the potential way to speed up your code.