CSE 847 (Spring 2021): Machine Learning— Homework 6

The homework is optional and questions are for bonus.

Instructor: Jiayu Zhou

Due on Friday, April 30th, 11:59 PM (Absolutely No Extension).

1 Sparse Learning

1. Recall that we have learned the maximize a posterior (MAP) estimation in the class to encode our prior knowledge about the distribution of model parameters.

   Given a data matrix $X \in \mathbb{R}^{n\times d}$, our model $w \in \mathbb{R}^d$, and the response y is generated according to the distribution $y \sim \mathcal{N}(Xw, I_n)$.

   - If we assume the model parameters are drawn from a *Laplacian prior*, i.e., $w_i \sim \frac{\lambda}{2}e^{-\lambda|w_i|}$, can you derive the optimization problem to get its MAP estimator?
   - Given the assumption in 1), when $X^TX = I$, can you derive the MAP estimator? Show detailed steps.

2. Many machine learning problems can be formulated into a first-order constrained optimization problem of the form:

$$\min_x \mathcal{L}(x), \quad \text{subject to: } x \in \mathcal{S}$$

   where $\mathcal{L}(x)$ is convex and differentiable and $\mathcal{S}$ is convex set. One way to optimize this problem is through the projected gradient, which iteratively solves:

$$x_{i+1} = \arg\min_x \left\{ \mathcal{L}(x_i) + \nabla\mathcal{L}(x_i)^T(x - x_i) + \frac{1}{2\gamma_i}\|x - x_i\|_2^2 \right\}, \quad \text{subject to: } x \in \mathcal{S},$$

   where γ_i is the step size given by a certain line search criteria. Show that this problem is equivalent to the following Euclidean projection problem:

$$x_{i+1} = \arg\min_x \|x - (x_i - \gamma_i\nabla\mathcal{L}(x_i))\|_2^2. \quad \text{subject to: } x \in \mathcal{S}$$

   Briefly explain the geometry interpretation of this algorithm.

3. Implement *stability selection* on top of *sparse logistic regression* to rank features. Use the Alzheimer's dataset[1] in Homework 4. Use 1000 bootstrap samples to compute the stability scores and lambda values in Homework 4 for each feature. Show the feature names and stability score of the top 20 features. The feature names can be found in the MATLAB data file `feature_name.mat` in the data folder.

2 Matrix Completion

1. The HARDIMPUTE is an approximated algorithm to solve the following rank constrained problem:

$$\min_Z \|P_\Omega(X) - P_\Omega(Z)\|_F^2, \quad \text{s.t. } \text{rank}(Z) = r.$$

[1] `https://github.com/jiayuzhou/CSE847/tree/master/data/alzheimers`

by iteratively compute the following update:

$$Z^{\text{new}} \leftarrow \mathbf{H}_r(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$$

where $\mathbf{H}_r(X)$ is the best rank-r approximation to X.

- Implement a simple version (you don't have to implement the low-rank + sparse trick to accelerate the computation) of HardImpute in Matlab:

```
function [X_complete] = hardimpute(X_missing, Omega, r)
% Input:
%    X_missing -- a m-by-n input matrix, only values at Omega
%    Omega     -- a m-by-n binary matrix, indicating location of the missing values
%    r         -- rank
```

- Experiment with one of your favorite picture landscape and buildings. Crop and scale the picture to 128 pixel by 128 pixel. Use only one channel (out of the R,G,B channels) and discard other channels. Randomly remove 50% of the pixels. Set the missing values to be 0, and record the missing location in the Ω matrix. Compute the recovery images given different rank values:

```
r_arr = [1, 5, 10, 15, 20, 25, 30];
```

  Show the corresponding recovery errors for different rank and their recovered images.

- Repeat the experiment with all three channels, assuming that the missing patterns of three channels are the same (the Ω is the same for three channels). Recovery of each image now requires to solve 3 independent matrix completion problems. Show the recovered color images.