Homework 3

Deadline: Thursday, Apr 27th, at 11:59pm.

Submission: In Teams. **DO NOT** submit any zipped files

- Put all your solutions and answers to all questions as a PDF file titled `hw3_writeup.pdf`. You can produce the file however you like (e.g. LATEX, Microsoft Word, scanner), as long as it is readable.
- Your final code for Question 1 should be submitted as python notebook `.ipynb` file.

Late Submission: 50% of the marks will be deducted for any submission beyond the deadline. No submissions will be accepted after two days past the deadline.

Collaboration: Homeworks are individual work. Please refrain from copying.

1. [**4 marks**] **SVM implementation in primal form** For a linear kernel, the primal formulation for Support Vector Classification was written using the hinge loss as:

$$\hat{\underline{\theta}} = \arg\min_{\underline{\theta}\in\mathbb{R}^p} \frac{1}{N}\sum_{i=1}^{N} \max(1 - y^{(i)}\underline{\theta}^T\underline{x}^{(i)}), 0) + \lambda\|\underline{\theta}\|_2^2$$

You will now optimize the above unconstrained formulation of SVM using *Stochastic Sub-Gradient Descent* (SSGD). In this problem, you will be using a binary (two-class) version of MNIST dataset. The data `mnist.mat` can be downloaded from the course website. The `mnist.mat` file contains the train, test and validation datasets.

We slightly modify the above equation and use the following formulation in this problem:

$$\hat{\underline{\theta}} = \arg\min_{\underline{\theta}\in\mathbb{R}^p} \frac{1}{N}\sum_{i=1}^{N} \max(1 - y^{(i)}\underline{\theta}^T\underline{x}^{(i)}), 0) + \frac{\lambda}{2}\|\underline{\theta}\|_2^2,$$

This is only done to simplify calculations. You will optimize this objective using SSGD. In SSGD, we randomly sample a training data point in each iteration and update the parameter vector by taking a small step along the direction of negative "sub-gradient" of the loss[1]. The SSGD algorithm is given by

- Initialize the weight vector $\underline{\theta} = \underline{0}$.
- For $t = 1, \ldots,$ T
  * Choose index $i_t \in \{1, \ldots, N\}$ uniformly at random
  * Set $\eta_t = \frac{1}{\lambda\, t}$
  * `if` $y^{(i_t)}\underline{\theta}^T\underline{x}^{(i_t)} < 1$ `then`:
    - Set $\underline{\theta} \leftarrow (1 - \lambda\eta_t)\underline{\theta} + \eta_t y^{(i_t)} x^{(i_t)}$
  * `else`:
    - Set $\underline{\theta} \leftarrow (1 - \lambda\eta_t)\underline{\theta}$
- Return $\underline{\theta}$

Note that we don't consider the bias/intercept term in this problem.

[1]Sub-gradient generalizes the notion of gradient to non-differentiable functions

Tasks:

(a) Write up a function `train(theta0, Xtrain, ytrain, tot_iters, lamda)` in the `SVM_run.ipynb` notebook file.

- The function `train(theta0, Xtrain, ytrain, tot_iters, lamda)` runs the SSGD algorithm
- It takes in an initial weight vector `theta0`, matrix of covariates `Xtrain`, a vector of labels `ytrain`.
- `tot_iters` is the number of iterations of SSGD
- `lamda` is the hyperparameter in the objective function.
- The function outputs the final learned weight vector `theta`.

(b) Perform training and see the classification accuracy (in percentage) on training and test sets.

(c) Use `validation` dataset for picking a good `lamda`(λ) from the set `{1000, 100, 10, 1, 0.1}`. Plot a graph of validation error vs lamda values and include it in the writeup pdf.

(d) Report the accuracy numbers on train and test datasets obtained using the best `lambda`, after running SSGD for 200 epochs (i.e., `tot_iters` $= 200N$). Generate the training accuracy vs. training time and test accuracy vs. training time plots, and include them in the writeup pdf.

2. **[3 marks] Bagging**: It was told in class that bagging with B models reduces variance but does not change the bias of the ensemble. Prove it mathematically using the concept of bias-variance decomposition.

3. **[3 marks] GMM with EM derivation**: For the case of unsupervised learning using GMM, we try to model the input distribution $p(\underline{x})$ using a family of Gaussian distributions, whose pdf look as follows:

$$\text{GMM}(\underline{x}) = \sum_{m=1}^{M} \pi_m \mathcal{N}\left(\underline{x} \mid \underline{\mu}_m, \underline{\underline{\Sigma}}_m\right)$$

where $\mathcal{N}\left(\cdot \,||\, \underline{\mu}, \underline{\underline{\Sigma}}\right)$ represent a Gaussian pdf with mean $\underline{\mu}$ and covariance $\underline{\underline{\Sigma}}$, and $\{\pi_1, \ldots, \pi_M\}$ are prior weights such that

$$\sum_{m=1}^{M} \pi_m = 1, \quad \pi_m \geq 0 \quad \forall m$$

All the parameters are stacked into $\underline{\theta} = \left\{\underline{\mu}_m, \underline{\underline{\Sigma}}_m, \pi_m\right\}_{m=1}^{M}$. The EM algorithm for learning a GMM (for unsupervised case) from training data $\left\{\underline{x}^{(i)}\right\}_{i=1}^{N}$ is outlined below:

- Initialize $\underline{\theta}$: $\underline{\mu}_m$, $\underline{\underline{\Sigma}}_m$, and π_m for all $m \in \{1, \ldots, M\}$
- Repeat the following until convergence:

(a) Expectation step (E-step):

$$w_m^{(i)} \leftarrow \text{Prob}\left(\underline{x}^{(i)} \in \text{cluster } m \mid \underline{\theta}, \underline{x}^{(i)}\right)$$

(b) Maximization step (M-step):

$$\hat{\underline{\theta}} \leftarrow \arg\max_{\underline{\theta}} \sum_{i=1}^{N} \sum_{m=1}^{M} w_m^{(i)} \left(\ln \mathcal{N}\left(\underline{x}^{(i)} \mid \underline{\mu}_m, \underline{\underline{\Sigma}}_m\right) + \ln \pi_m\right)$$

Consider a simplified scenario where we assume that the covariances of the Gaussian components are the same and equal, i.e., $\underline{\underline{\Sigma}}_m = \underline{\underline{\Sigma}}$. Assume that you have got $w_m^{(i)}$ is the result of the E-step at the mth iteration, and also an estimate $\hat{\underline{\mu}}_m$ from the current M-step. Next you need to update the value of $\underline{\underline{\Sigma}}$. What would be the update formula for $\underline{\underline{\Sigma}}$? Show your derivation.

Hint: Work with precision matrix $\underline{\underline{\Lambda}} = \underline{\underline{\Sigma}}^{-1}$ for easier derivation.