COMS 4771 FA18 HW2

Due: Fri Oct 26, 2018 at 11:59pm

You are allowed to write up solutions in groups of (at max) three students. These group members don't necessarily have to be the same from previous homeworks. Only one submission per group is required by the due date on Gradescope. Name and UNI of all group members must be clearly specified on the homework. No late homeworks are allowed. To receive credit, a typesetted copy of the homework pdf must be uploaded to Gradescope by the due date. You must show your work to receive full credit. Discussing possible solutions for homework questions is encouraged on piazza and with peers outside your group, but every group must write their own individual solutions. You must cite all external references you used (including the names of individuals you discussed the solutions with) to complete the homework.

1 **[Empirical study of various gradient based optimization procedures]** There are several gradient based optimization procedures that have been proposed over the years for finding local optima of a smooth non-convex function. A few of them are listed below.

- (Full) Gradient Descent (GD)
- Stochastic Gradient Descent (SGD)
- SGD with Momentum (SGDM)
- Adaptive Gradient (AdaGrad)
- RMSProp
- AdaDelta
- Adaptive Momentum (ADAM)

Using online resources, books, etc. your goal is to learn about what each technique is about and what makes each technique different. (make sure to properly cite all resources used!)

(i) Describe each of the techniques in detail. Compare and contrast the relative advantages and disadvantages of each of the techniques. Are there specific types of datasets where one technique is expected to perform better than the other? if so, why? Provide detailed justifications for each.

(ii) Implement each of the optimization procedures and experimentally verify your findings in part (i). Using synthetic dataset(s) compare and contrast the relative performance of each of the techniques.

You must submit your code to Courseworks to receive full credit.

2 **[A variant of Perceptron Algorithm]** One is often interested in learning a *disjunction model*. That is, given binary observations from a d feature space (i.e. each datapoint $x \in \{0,1\}^d$), the output is an 'OR' of few of these features (e.g. $y = x_1 \vee x_{20} \vee x_{33}$). It turns out that the following variant of the perceptron algorithm can learn such disjunctions well.

```
Perceptron OR Variant
learning:
- Initialize $w := 1$
- for each datapoint $(x, y)$ in the training dataset
-   $\hat{y} := \mathbf{1}[w \cdot x > d]$
-   if $y \neq \hat{y}$ and $y = 1$
-       $w_i \leftarrow 2w_i$      (for $\forall i : x_i = 1$)
-   if $y \neq \hat{y}$ and $y = 0$
-       $w_i \leftarrow w_i/2$    (for $\forall i : x_i = 1$)

classification:
$f(x) := \mathbf{1}[w \cdot x > d]$
```

We will prove the following interesting result: The **Perceptron OR Variant** algorithm makes at most $2 + 3r(1 + \log d)$ mistakes when the target concept is an OR of r variables.

(i) Show that for any positive example, **Perceptron OR Variant** makes $M_+ \leq r(1 + \log n)$ mistakes.

(ii) Show that each mistake made on a negative example decreases the total weight $\sum_i w_i$ by at least $d/2$.

(iii) Let M_- denote the total number of mistakes on negative examples, and $TW(t)$ denote the total weight of w at iteration t. Observing that

$$0 < TW(t) \leq TW(0) + dM_+ - (d/2)M_-,$$

conclude that the total number of mistakes (i.e., $M_+ + M_-$) is at most $2 + 3r(1 + \log d)$.

3 **[Recommender systems]** Suppose that the CEO of some startup company calls you and tells you that she sells d items on her web site. She has access to ratings given by n users, each of whom have rated a subset of the the d items. Assume that each rating is a real-number. Your task is to estimate the missing ratings. (Think of the data as $n \times d$ matrix in which some entries are missing and your task is to estimate the missing entries). Being a good machine learner, you have come up with the following generative model for the rating:

$$r_{i,j} = u_i \cdot v_j + \epsilon_{i,j}$$

- $r_{i,j}$ is the rating of the jth product by user i
- $u_i, v_j \in \mathbb{R}^k$ are k-dimensional vectors encoding user i's preferences, and item j's attributes respectively.
- $u_i \cdot v_j$ models how much user i prefers item j.
- $\epsilon_{i,j}$ is distributed as independent zero-mean and σ^2-variance Gaussian noise.

The semantics of the rating model are as follows: the rating i, j is a noisy realization of the dot product between user i's preference and item j's attribute.

Given this model, the goal obviously is to estimate the user preference vectors u_i and item attribute vectors v_j for all users and items. These could then be used to estimate the missing ratings and potentially suggest relevant items to interested users.

(i) Given the above ratings model, what are the total number of parameters that need to be estimated (give the answer in terms of n, d and k)?

(ii) One way to estimate the parameters of this generative ratings model is to do maximum likelihood estimation (MLE). Cast the parameter estimation problem as MLE and write down the objective (and constraints, if any) that need to be optimized.

(iii) An easy way to optimize for the parameters (in part (ii)) is to actually optimize the *negative log likelihood* function. Write down the negative log likelihood optimization problem and simplify it as much as possible.

(iv) Is the negative log-likelihood problem formulation (in part (iii)) a convex optimization problem in with respect to the parameter u_i? How about with respect to the parameter v_j? Justify your answer.

(v) Is the negative log-likelihood problem formulation (in part (iii)) *jointly* convex in the parameters u_i and v_j simultaneously? Justify your answer.

(vi) What is the optimal setting of the parameters u_i? (derive the answer in terms of v_j, $r_{i,j}$). Similarly, derive the optimal setting of the parameters v_j?

(vii) Show that this model cannot be used to predict the rating given by a new/previously unseen user $\tilde{u}$ or a new/previously unseen item $\tilde{v}$. Suggest some approaches on how this can be fixed.

4 **[Making data linearly separable by feature space mapping]** Consider the infinite dimensional feature space mapping

$$\Phi_{\alpha,\beta} : \mathbb{R} \to \mathbb{R}^\infty$$
$$x \mapsto \Big(\mathbf{1}[\beta > x > \gamma - \alpha]\Big)_{\gamma \in \mathbf{R}}.$$

(i) Given some $B > 0$, show that for any n distinct points $x_1, \ldots, x_n \in [-B, B]$, there exists $\alpha > 0$ such that the mapping $\Phi_{\alpha,B}$ can linearly separate *any* binary labeling of the n points.

(ii) Show that one can efficiently compute the dot products in this feature space, by giving an analytical formula for $\Phi_{\alpha,B}(x) \cdot \Phi_{\alpha,B}(x')$ for arbitrary points x and x'.

5 **[Movie Recommendation System Case Study]** Download the datafile `movie_ratings.csv`. This datafile contains 100,000 movie ratings for 9,000 movies given by 600 users. Your goal is to design a good prediction model for this dataset by (a) using the model suggested in Q3, (b) come up with your own model by doing some research on the internet (make sure to properly cite all resources used!).

(i) Describe the mathematical details of your model. How does it differ from the Q3 model? What potential advantages and disadvantages do you expect?

(ii) Implement the model suggested in Q3 and your proposed model.

(you may use builtin functions for performing basic linear algebra and probability calculations but you should develop the models from scratch.)

You must submit your code to Courseworks to receive full credit.

(iii) Which of the two models is better for movie recommendation dataset? You must justify your answer with appropriate performance graphs demonstrating the superiority of one model over the other. Example things to consider: you should evaluate how the model behaves on a holdout test sample for various splits of the data; how does the training sample size affects the classification performance? how can the two models perform on new user/movie?