

# Data Mining: Learning from Large Data Sets

## Final exam

Jan 25, 2016

Time limit: 120 minutes

Number of pages: 15

Total points: 100

You can use the back of the pages if you run out of space. Collaboration on the exam is strictly forbidden. Please show *all* of your work and always *justify* your answers.

Please write your answers with a *pen*.

**(1 point)** Please fill in your name and student ID.

*Please leave the table below empty.*

<b>Problem</b>	<b>Maximum points</b>	<b>Obtained</b>
1.	16	
2.	18	
3.	30	
4.	20	
5.	15	
Total	100	

## 1. Map Reduce

(16 points)

In this task, you are provided with a list of  $n > 2$  sensor measurements  $x_1, x_2, \dots, x_n \in \mathbb{R}$ . Let  $\hat{\mu}$  be the sample mean, i.e.

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^n x_i$$

and  $\hat{\sigma}^2$  the unbiased sample variance, i.e.

$$\hat{\sigma}^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \hat{\mu})^2.$$

(6 points) (a) Show that  $\hat{\sigma}^2$  can be decomposed into the form

$$\hat{\sigma}^2 = \alpha(n) \sum_{i=1}^n x_i^2 - \beta(n) \hat{\mu}^2$$

where  $\alpha(n)$  and  $\beta(n)$  are functions that do not depend on the value of the measurements  $x_i$ .

**(10 points)** (b) The goal of this task is to calculate  $\hat{\sigma}^2$  with a MapReduce program under the following conditions:

- The Map function receives as input a list of measurements  $x_i$ , e.g. (1.0, 2.5, 3.5, 2.0).
- The Map function returns a single  $(key, value)$  pair where  $value$  is a list of three real numbers, e.g.  $value = (5.3, 1.0, 2.1)$ .
- The Reduce function returns  $\hat{\sigma}^2$ .
- You are guaranteed that each measurement  $x_i$  is sent to exactly one mapper.
- The input lists to different mappers may be of different lengths.

Write down both a Map and a Reduce function calculating  $\hat{\sigma}^2$ .

*Hint: Use the decomposition from part (a). You may call the functions  $\alpha(n)$  and  $\beta(n)$  in your code.*

## 2. Near-duplicate Detection using Hashing

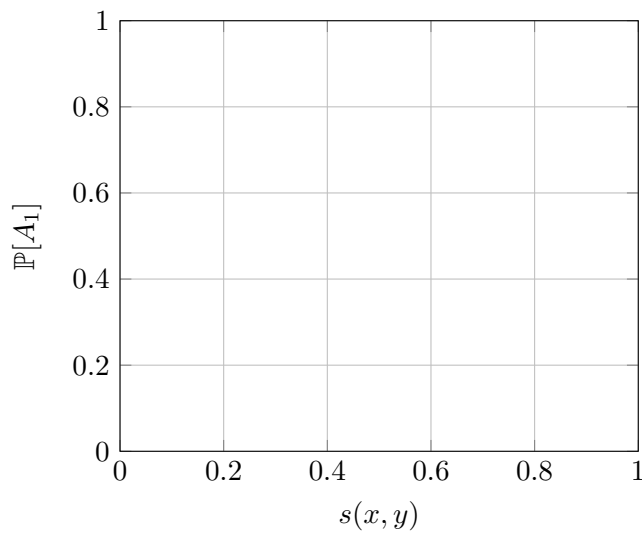
(18 points)

Let  $X$  be a set of  $n$  songs and let  $s : X \times X \rightarrow [0, 1]$  be a symmetric function measuring the similarity between two songs and satisfying  $s(x, x) = 1$  for all  $x \in X$ . Let  $\mathcal{H}$  be a random family of hash functions with the property that for a randomly sampled hash function  $h \in \mathcal{H}$  and all  $x, y \in X$

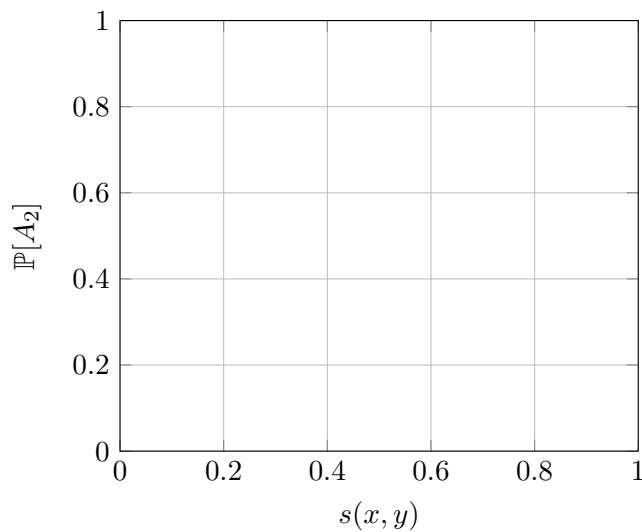
$$\mathbb{P}_{h \in \mathcal{H}} [h(x) = h(y)] = s(x, y).$$

In the following, let  $h_1, h_2, \dots, h_m$  be  $m$  independently sampled hash functions from  $\mathcal{H}$ .

- (3 points) (a) Consider the event  $A_1$  that two songs  $x, y \in X$  **agree on all**  $m$  hash functions, i.e.  $h_i(x) = h_i(y)$  **for all**  $i = 1, 2, \dots, m$ . Plot the relationship between the probability of  $A_1$ , i.e.  $\mathbb{P}[A_1]$ , and the similarity  $s(x, y)$  for the cases  $m = 1$  and  $m \rightarrow \infty$ .



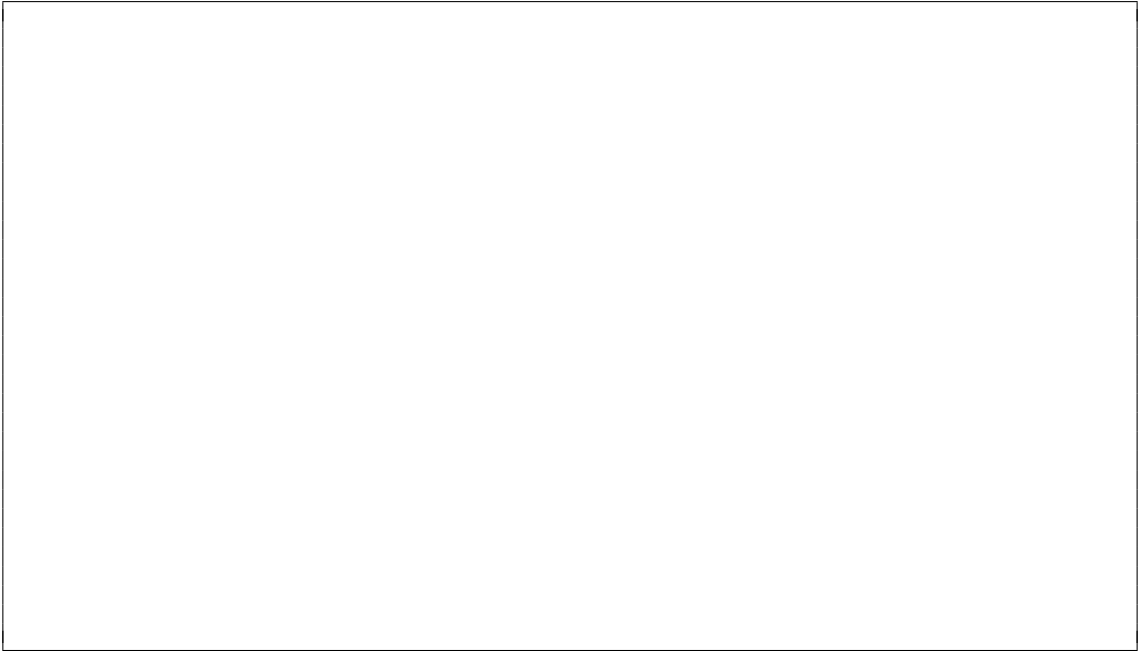
- (3 points) (b) Consider the event  $A_2$  that two songs  $x, y \in X$  **agree on any** of the  $m$  hash functions, i.e.  $h_i(x) = h_i(y)$  **for some**  $i = 1, 2, \dots, m$ . Plot the relationship between the probability of  $A_2$ , i.e.  $\mathbb{P}[A_2]$ , and the similarity  $s(x, y)$  for the cases  $m = 1$  and  $m \rightarrow \infty$ .



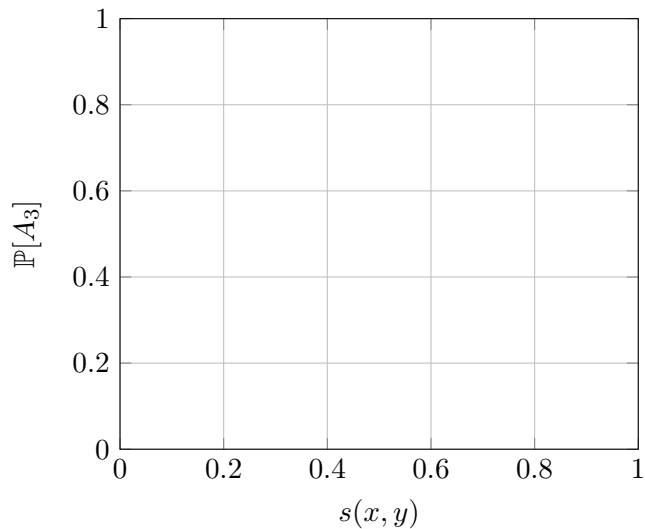
**(4 points)** (c) Our goal is to identify all pairs of songs that have a similarity of at least 0.8, i.e. the set

$$D = \{(x, y) \mid x, y \in X \text{ such that } s(x, y) \geq 0.8\}.$$

Assume that you are penalized for false negatives, but not for false positives. Your goal is to minimize the total penalty. For  $m \rightarrow \infty$ , would you rather output the pairs  $x, y \in X$  that agree on all hash functions as in (a) or on any of the hash functions as in (b)? Justify your answer.



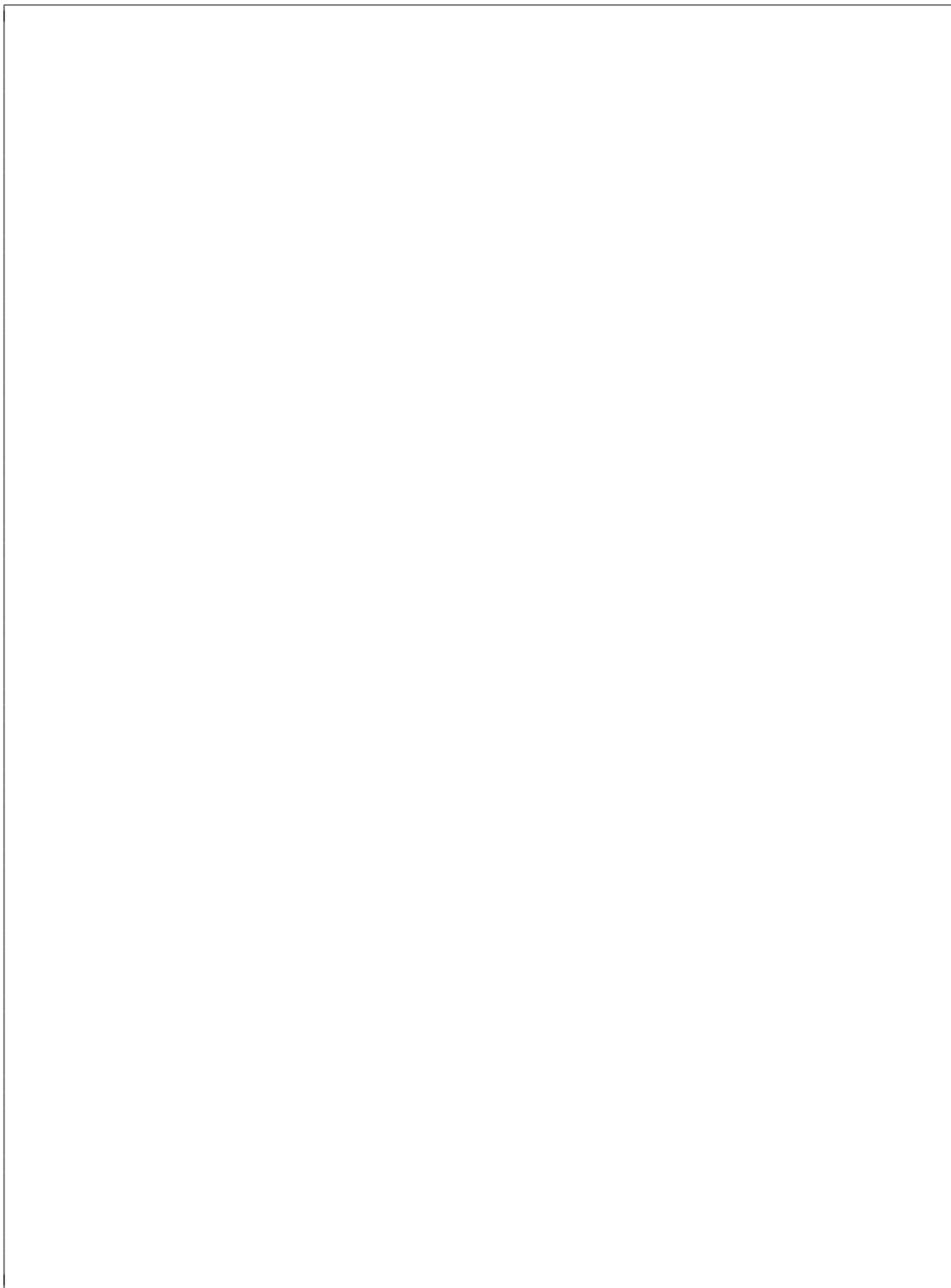
**(2 points)** (d) Consider two elements  $x, y \in X$  with similarity  $s(x, y)$ . Let  $A_3$  be the event that an algorithm labels these two elements as near-duplicates. For a randomized algorithm that perfectly detects pairs with similarity of at least 0.8, plot below the relationship between the probability of  $A_3$ , i.e.  $\mathbb{P}[A_3]$ , and the similarity  $s(x, y)$ .



**(6 points)** (e) Consider four arbitrary songs  $q, r, x, y \in X$  such that  $s(q, r) = s(x, y) = 0.8$ . Sample a random hash function  $h \in \mathcal{H}$  and denote by  $A_4$  the event that both  $h(q) = h(r)$  and  $h(x) = h(y)$ .

Provide an upper bound  $c \in \mathbb{R}$  for  $\mathbb{P}_{h \in \mathcal{H}}[A_4]$  and show that this bound  $\mathbb{P}_{h \in \mathcal{H}}[A_4] \leq c$  is tight.

*Hint: To show tightness, provide an example  $q, r, x, y \in X$  such that  $\mathbb{P}_{h \in \mathcal{H}}[A_4] = c$ .*



### 3. Online Convex Programming

(30 points)

Assume you are given a data set  $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$  with  $\mathbf{x}_i \in \mathbb{R}^d$  and  $y_i \in \{-1, 1\}$ . In class we discussed how to optimize the hinge loss using online convex programming (OCP), where

$$\ell_h(\mathbf{x}, y; \mathbf{w}) = \max(0, 1 - y\mathbf{w}^T \mathbf{x}).$$

Here we will consider two different loss functions,  $\ell_A$  and  $\ell_B$ , defined as follows:

$$\ell_A(\mathbf{x}, y; \mathbf{w}) = \exp\left(-\frac{1}{2}y\mathbf{w}^T \mathbf{x}\right)$$

$$\ell_B(\mathbf{x}, y; \mathbf{w}) = \begin{cases} \exp\left(-\frac{1}{2}y\mathbf{w}^T \mathbf{x}\right), & \text{if } y\mathbf{w}^T \mathbf{x} \geq -2 \ln 2 \\ 1 - y\mathbf{w}^T \mathbf{x}, & \text{otherwise} \end{cases}$$

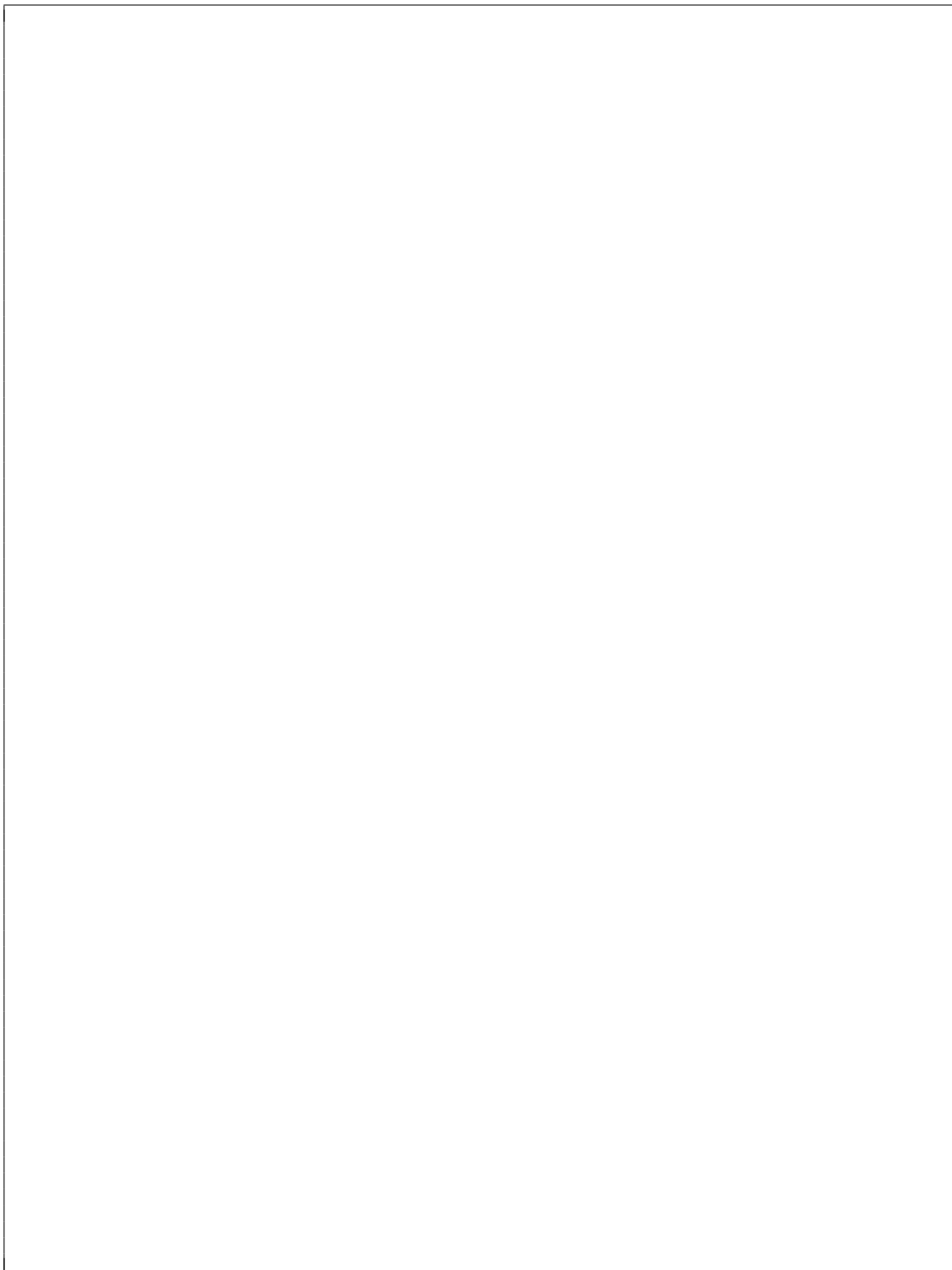
(6 points) (a) Sketch the three loss functions described above in the same plot as functions of  $y\mathbf{w}^T \mathbf{x}$ .



**(8 points)** (b) We want to compute a classifier using online convex programming, that is, by solving

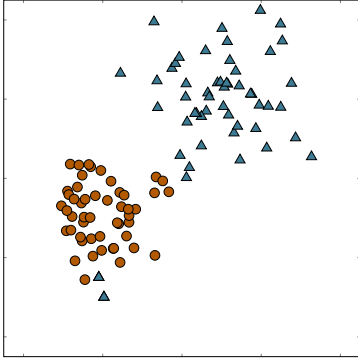
$$\underset{\mathbf{w}}{\text{minimize}} \sum_{i=1}^n \ell_k(\mathbf{x}_i, y_i; \mathbf{w}), \text{ for } k = A, B. \quad (1)$$

Compute a subgradient of  $\ell_A$  and  $\ell_B$  at  $\mathbf{w}$  given fixed  $\mathbf{x}$  and  $y$ .

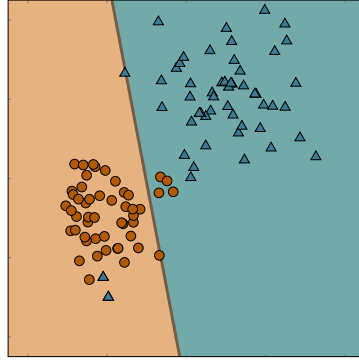




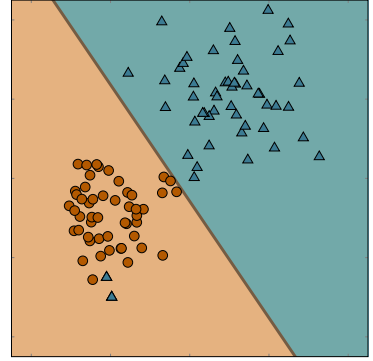
(6 points) (c) For the 2-D data set shown in plot (i) we solve the minimization problem (1) using  $\ell_A$  and  $\ell_B$ , and obtain the classifiers shown in plots (ii) and (iii), respectively. Which classifier corresponds to which loss function? Briefly explain your reasoning.



(i)



(ii)



(iii)



**(10 points)** (d) Finally, we want to add a regularization constraint to our objective to hopefully achieve a smaller generalization error. The regularized version of our problem is

$$\begin{aligned} & \underset{\mathbf{w}}{\text{minimize}} && \sum_{i=1}^n \ell_k(\mathbf{x}_i, y_i; \mathbf{w}) \\ & \text{subject to} && \|\mathbf{w}\|_\infty \leq \lambda, \end{aligned}$$

where  $\|\mathbf{w}\|_\infty = \max_{j=1, \dots, d} |w_j|$ .

Formulate and solve the projection step of the OCP algorithm.

#### 4. Active Learning

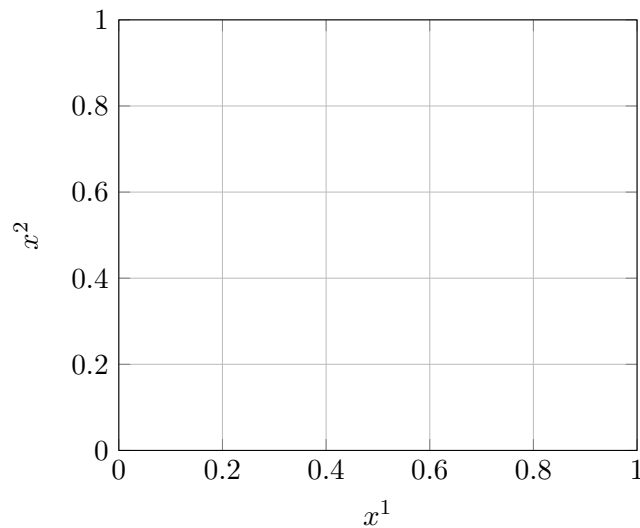
(20 points)

Let  $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  be a set of  $n$  unlabeled examples where each example  $\mathbf{x}_i = (x_i^1, x_i^2)$  is a point within the unit square (i.e.,  $\mathbf{x}_i \in [0, 1]^2$ ). Each example  $\mathbf{x}_i$  has a binary label  $y_i \in \{-1, 1\}$  determined according to

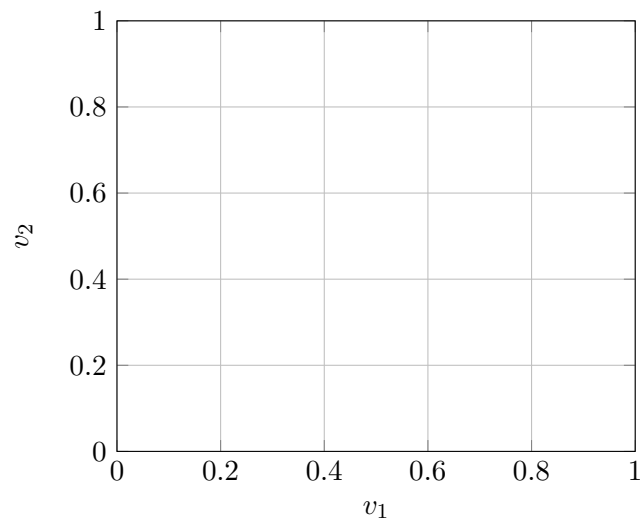
$$y_i = \begin{cases} 1 & \text{if } x_i^1 > v_1 \text{ and } x_i^2 > v_2 \\ -1 & \text{else} \end{cases}$$

for some unknown  $(v_1, v_2) \in [0, 1]^2$ .

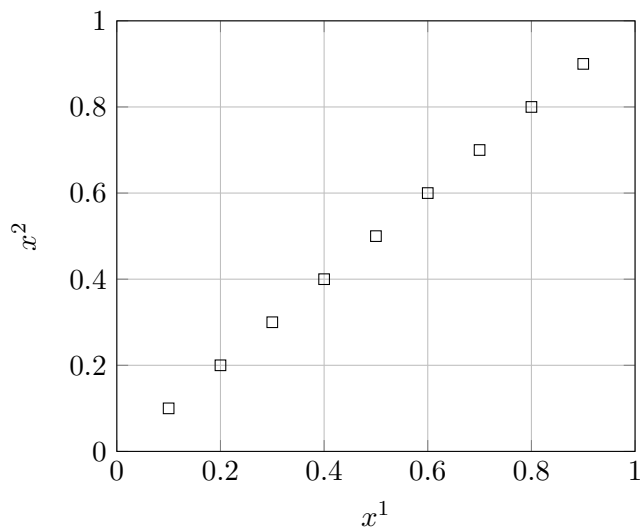
(2 points) (a) For  $v_1 = v_2 = 0.6$ , mark the areas where the labels are 1 and where they are  $-1$ .



(4 points) (b) Let  $X$  consist of the data points  $\mathbf{x}_1 = (0.4, 0.4)$  and  $\mathbf{x}_2 = (0.6, 0.6)$  with labels  $y_1 = -1$  and  $y_2 = 1$ . Mark all values of  $(v_1, v_2)$  which are consistent with the data.



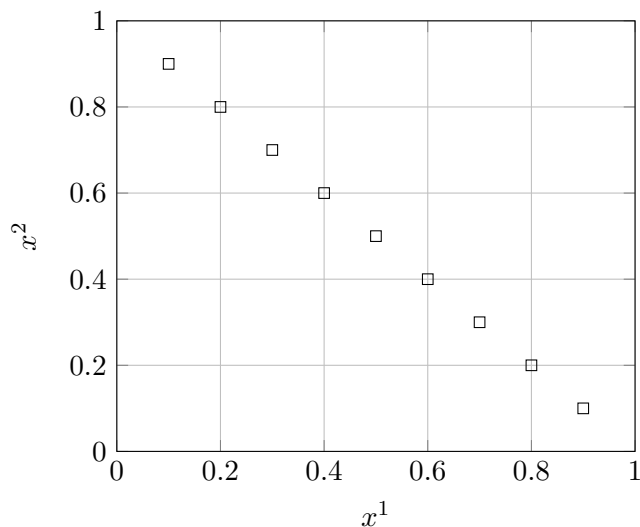
(7 points) (c) Consider a dataset  $X$  of  $n$  examples  $\mathbf{x}_i = (\frac{i}{n+1}, \frac{i}{n+1})$ . For  $n = 9$ , the dataset is plotted below.



In the active learning setting, the labels  $y_i$  of the data points as well as  $v_1$  and  $v_2$  are not known a priori. An active learning algorithm sequentially chooses an example  $\mathbf{x}_i$  and then observes its label  $y_i$ . Based on the label, it then selects subsequent examples to query.

Your task is to provide an algorithm that infers the labels of all  $n$  examples, while performing  $\mathcal{O}(\log n)$  queries. You may use the figure above to illustrate your algorithm.

(7 points) (d) Consider a dataset  $X$  of  $n$  examples  $x_i = (\frac{i}{n+1}, 1 - \frac{i}{n+1})$ . For  $n = 9$ , the dataset is plotted below.



Show that, in the worst case, any active learning algorithm needs to query at least  $n$  points in order to be certain about the labels for all  $n$  points. You may use the figure above to illustrate your proof.

*Hint: Assume that the algorithm has labels for  $n - 1$  points and then consider the last point.*

## 5. Bandits

(15 points)

You are part of a team that has to decide between two different configurations for a new component you have developed. As you do not know the effects of these configurations on the overall system, you pose this as a 2-armed stochastic bandit problem. In each of  $t = 1, 2, \dots, T$  rounds, you can choose a configuration to deploy (i.e., an arm to pull) and then obtain a random pay-off. We denote by  $\mu_1$  and  $\mu_2$  the (unknown) expected pay-offs of the two configurations and assume that all payoffs are independent. The goal is to have a *zero regret* strategy in expectation, i.e. one that satisfies

$$\lim_{T \rightarrow \infty} \frac{\mathbb{E}[R_T]}{T} = 0,$$

where  $R_T$  is the total regret up to time  $T \in \{1, 2, \dots\}$ .

- (7 points) (a) Due to ease of implementation, one of your colleagues suggests to use the  $\epsilon$ -greedy strategy with a *fixed* (not time-varying)  $\epsilon > 0$ . Briefly describe the  $\epsilon$ -greedy strategy and prove that for  $\mu_1 \neq \mu_2$ , you cannot achieve zero regret in expectation.

**(8 points)** (b) Another colleague argues that the zero regret condition might be too restrictive. For example, it becomes harder to achieve as  $\mu_1$  and  $\mu_2$  get closer to one another, while practically it might not matter which configuration you use. To illustrate this, consider the case where the payoff of the first configuration is drawn from a Gaussian with mean  $\mu_1 = 0$  and variance 1 while the payoff of the second configuration is also Gaussian but with mean  $\mu_2 > 0$  and variance 1.

Assume that you have selected each configuration  $n$  times, and from that point onwards you will only deploy the configuration that had the larger *empirical* mean, as computed from these  $2n$  samples. Show that the probability you pick the correct configuration (necessary for achieving zero regret) decreases as  $\mu_2$  decreases.

*Hint: You may use that if  $X \sim \mathcal{N}(\mu, \sigma^2)$  and  $Y \sim \mathcal{N}(\nu, \tau^2)$  are independent, then we have  $Z \sim \mathcal{N}(\mu \pm \nu, \sigma^2 + \tau^2)$  for  $Z = X \pm Y$ .*

