Data Mining – Learning from Large Data Sets
Final Exam

Date:	18 August 2014
Time limit:	120 minutes
Number of pages:	10
Maximum score:	100 points

You can use the back of the pages if you run out of space. Most subproblems can be solved independently from each other. Collaboration on the exam is strictly forbidden.

[1 point] Please fill in your name and student ID:

1 [18 points] 1-Gram Markov Model Estimation

A language model is a probability distribution over sequences of words that come from some dictionary $\mathcal{D}$. More precisely, for every $n \in \mathbb{N}$ and sequence of words $w_1, \ldots, w_n$, where $w_i \in \mathcal{D}$, the language model assigns some probability $P(w_1, \ldots, w_n)$. One such model is the *1-gram markov model*, which assumes that w_i is independent of $w_1, \ldots, w_{i-2}$, when conditioned on its *immediate* predecessor w_{i-1}. Then, the probability factorizes as follows

$$P(w_1, w_2, \ldots, w_n) = P(w_1 \mid \texttt{BEG}) P(w_2 \mid w_1) \cdots P(w_n \mid w_{n-1}),$$

where we have introduced the symbol `BEG` to denote the beginning of the document.

In this question, we ask you to estimate such a model from a corpus of documents using Map-Reduce. Your mappers will get documents represented as lists of words. Your reducers should output for each word `w` in the corpus a list of pairs `[(w_1, p_1), (w_2, p_2), ..., (w_n, p_n)]`, where `p_i` is the *fraction* of times `w` was followed by word `w_i` in the *whole* corpus. You should include only those words `w_i` for which `p_i > 0`. Do not forget to include the symbol `BEG`, which preceeds the first word of the document. For example, given documents

```
[``peter'', ``likes'', ``chocolate''], [``you'', ``and'', ``peter'', ``ski''],
[``peter'', ``likes'', ``apples''],
```

for the word `` ``peter'' `` you should output `[(``likes'', 2/3), (``ski'', 1/3)]` and for the symbol `BEG` you should output `[(``peter'', 2/3), (``you'', 1/3)]`.

You can assume to have the default Python types and functions at your disposal, and that the symbol `BEG` is defined and different from any word appearing in the documents. Please write the *pseudo-code* for your `map` and `reduce` functions.

```
# Use emit(key, value) to emit a key-value pair.
map(document):
```

```
# Use output(word, list) to output the results.
# Order of the list is irrelevant.
reduce(key, values):
```

2 [20 points] Online Support Vector Regression

For this question you will derive a regression version of online SVMs called *support vector regression (SVR)*. In contrast to regular SVMs, which are used for classification, SVRs are used to learn a linear mapping $y \approx \mathbf{w}^T\mathbf{x}$, where $y \in \mathbb{R}$.

Instead of the *hinge loss* (used in SVMs), we will use SVR with the *squared ε-insensitive loss*: For a training example $(y_i, \mathbf{x}_i)$ and candidate coefficients $\mathbf{w}$, this loss is defined as

$$\ell_\varepsilon(y_i, \mathbf{x}_i; \mathbf{w}) = \begin{cases} 0 & \text{if } |y_i - \mathbf{w}^T\mathbf{x}_i|^2 \le \varepsilon \\ |y_i - \mathbf{w}^T\mathbf{x}_i|^2 - \varepsilon & \text{otherwise} \end{cases}$$

Similar as for the SVM, in order to avoid overfitting, a regularization term needs to be added. We will consider SVR using the ℓ_∞ norm, i.e., $||\mathbf{w}||_\infty = \max_j |w_j|$.

Given a data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ and constant $\lambda > 0$, support vector regression with *squared ε-insensitive loss and ℓ_∞ norm regularization* solves the program

$$\min_{\mathbf{w}} \sum_{i=1}^{n} \ell_\varepsilon(y_i, \mathbf{x}_i; \mathbf{w}) \text{ such that } ||\mathbf{w}||_\infty \le \frac{1}{\lambda}$$

(a) **[4 points]** Plot the *squared ε-insensitive loss* as a function of $\xi = y - \mathbf{w}^T\mathbf{x}$.

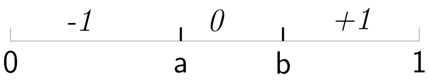
(b) **[6 points]** Derive the sub-gradient of the loss function $\ell_\varepsilon(y, \mathbf{x}; \mathbf{w})$ with respect to the weights $\mathbf{w}$ for a fixed data point $(\mathbf{x}, y)$.

(c) **[10 points]** Derive the projection step for the vector $\mathbf{w} \in \mathbb{R}^d$ w.r.t the ℓ_∞ norm. I.e., show how to find $\arg\min\{\mathbf{w}' : ||\mathbf{w}' - \mathbf{w}||_2 \text{ s.t. } ||\mathbf{w}'||_\infty \leq 1/\lambda\}$ for a fixed vector $\mathbf{w}$.

3 [22 points] Active learning for interval indicator functions

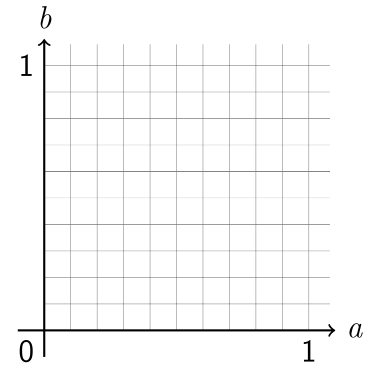
Consider the problem of learning single-variable interval functions $f : [0,1] \mapsto \{-1, 0, +1\}$, where

$$f(x) := \begin{cases} -1, & \text{if } 0 \le x < a, \\ 0, & \text{if } a \le x < b, \\ +1, & \text{if } b \le x \le 1, \end{cases}$$

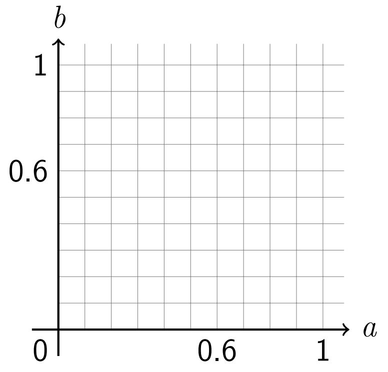


and $0 < a < b < 1$ (see above figure on the right). Suppose you are given a pool $X = \{x_1, ..., x_n\}$ of n unlabeled examples where each $x_i \in [0,1]$. You may assume that each interval $[0,a)$, $[a,b)$, and $[b,1]$ contains at least one example from X. We would like to develop a pool-based active learning strategy for this hypothesis space. Assuming there is no label noise, the algorithm sequentially selects one of the n examples and obtains its true label.

(a) [**3 points**] Illustrate the version space on the 2-D coordinate system shown below (i.e. set of hypotheses consistent with the observed labels) before observing the label of any example. Note that each hypothesis is precisely represented by (a,b), hence the version space can always be represented as a subset of $[0,1]^2$.



(b) [**3 points**] Suppose we pick an example $x = 0.6$ from X, and observe that its label is 0. Illustrate the version space after incorporating this observation.



(c) **[10 points]** Devise an active learning algorithm with sublinear label complexity, i.e., the number of labeled examples needed to learn the labels of all n samples has to be strictly *sublinear* in n. Provide *pseudo-code* for your algorithm. How many labeled examples does it need?

(d) **[6 points]** Now suppose the label acquisition cost of the data is label-dependent. In particular, the cost to label $x_i \in [a, b)$ is 1, while the cost to label an example in $[0, a)$ and $[b, 1]$ is 0. Devise an active learning scheme with the smallest label cost. What is the label cost of your algorithm?

4 [20 Points] Online l_1 K-means clustering

The goal of the l_1 K-means clustering problem is: Given a set of points $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, \ldots, n$, find the centers of k clusters $\mu = [\mu_1, \ldots, \mu_k], \mu_i \in \mathbb{R}^d$ minimizing the average l_1 distance from the points to the closest cluster center according to the following cost function:

$$\mu^* = \arg\min_{\mu} \sum_{i=1}^{n} l(\mathbf{x}_i, \mu)$$

where $l(\mathbf{x}_i, \mu) = \min_{j \in 1,\ldots,k} ||\mathbf{x}_i - \mu_j||_1$ is the loss function associated with data point $\mathbf{x}_i$.

In this question you are asked to design an online algorithm for the l_1 K-means clustering problem, similar to the online algorithm for the regular K-means problem discussed in class. The difference is that instead of using the squared Euclidean (l_2) norm $||\mathbf{x}_i - \mu_j||_2^2$, we use the l_1 norm $||\mathbf{x}_i - \mu_j||_1$.

(a) **[6 Points]** Compute the gradient of the loss function, i.e. $\bigtriangledown_\mu \, l(\mathbf{x}_i, \mu)$ where differentiable.

(b) **[10 Points]** Write pseudo-code for an online algorithm for the $l1$ K-means problem based on stochastic gradient descent, similar to the online K-means algorithm discussed in class.

(c) **[4 Points]** When would you prefer the l_1 distance over the Euclidean (l_2) distance as used in the standard K-means algorithm, and why?

5 [12 points] Bob's Choice

Bob has been offered two different jobs. One of them is in tele-marketing and pays a fixed 0.9 CHF per minute. The other job is in tele-sales which pays a base 0.85 CHF per minute plus commission for products sold. While Bob does not know the expected commission, we know that the expected commission is 0.03 CHF per minute (with maximum commission fixed at 1 CHF per minute).

Bob takes the following strategy. Initially, he decides to try the sales job for a fixed length of 1000 minutes (trial period) and then decides between the two jobs based on the mean pay at the end of the trial period.

(a) [**5 Points**] Recall that a version of Hoeffding's inequality states that if $X_1, X_2, \ldots, X_n$ are independent random variables bounded in $[0, 1]$

$$P(\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}(X_i) \geq a) \leq e^{-2na^2}$$

Use the inequality to derive an upper bound B on the probability that Bob makes the wrong choice at the end of his trial period of 1000 minutes. Write an expression for B.

(b) [**7 Points**] Provide the tightest lower bound you can on the expected earnings of Bob for a time period of 2000 minutes. *Hint: You may wish to state your bound in terms of the quantity B from part a).*

6 [7 points] Submodular functions

Let V be a finite set and $G : 2^V \to \mathbb{R}$ be a submodular function. Let $S \subseteq V$ be drawn at random from some probability distribution $P(S)$ over 2^V. Then show that $F : 2^V \to \mathbb{R}$ defined as

$$F(A) = \sum_{S \subseteq V} P(S) G(V \setminus (A \cap S))$$

is a submodular function. You may use the closedness properties of submodular functions discussed in class. Clearly state if you use any such property.