Technical Report

Logistic Regression in Datastreams

Chris Schwiegelshohn, Christian Sohler

01/2014

technische universität
dortmund

SFB 876 Verfügbarkeit von
Information durch Analyse unter
Ressourcenbeschränkung

Part of the work on this technical report has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis", project A2, C4.

Speaker: Prof. Dr. Katharina Morik
Address: TU Dortmund University
Joseph-von-Fraunhofer-Str. 23
D-44227 Dortmund
Web: http://sfb876.tu-dortmund.de

Abstract

Learning from data streams is a well researched task both in theory and practice. As remarked by Clarkson, Hazan and Woodruff [12], many classification problems cannot be very well solved in a streaming setting. For previous model assumptions, there exist simple, yet highly artificial lower bounds prohibiting space efficient one-pass algorithms. At the same time, several classification algorithms are often successfully used in practice. To overcome this gap, we give a model relaxing the constraints that previously made classification impossible from a theoretical point of view and under these model assumptions provide the first $(1+\epsilon)$-approximate algorithms for sketching the objective values of logistic regression and perceptron classifiers in data streams.

1 Introduction

Mining large datasets has become a highly relevant task with the massive increase in data available. Much theoretical research has been devoted to studying such problems either for data streams [19] where points are processed one by one with limited memory, or recently for distributed settings [18]. In this paper we discuss the possibility of learning binary linear classifiers, that is classifiers separating two labeled point sets via a hyperplane, in data streams. More precisely, we study loss-based linear classifiers, including logistic regression and perceptron classifiers, where each point from the data set is assigned a loss based on the relative position towards the hyperplane. The optimal solution, that is the best-fit hyperplane minimizes the sum of losses incurred for each point.

The usual approach for learning tasks in data streams is to summarize the data such that the model learned from the summary is equal to or, far more likely, approximates the model learned from the entire data set. For optimization problems, the model computed on the summary is required to have an objective value with an $(1+\epsilon)$-factor of the objective value of the optimum model computed on the entire dataset. Clustering [7, 15] in particular has been subject to intensive research, though other topics such as subspace approximation [23] and regression [13] have also been studied. Similar approaches are not applicable for binary classification. Consider as an example the objective function of logistic regression $L(w) := \sum_{x_i \in X} \ln(1 + e^{-y_i \langle w, x_i \rangle})$, where w is a hyperplane, x_i a point from the dataset X and $y_i \in \{-1, 1\}$ a class label. If the data is linearly separable, that is, if there exists a hyperplane such that all points of different labels are on opposing sides of the hyperplane, the objective function approaches 0, otherwise the objective function is at least $\ln(2)$. Hence, in order for the objective function of a hyperplane w to be within an $(1+\epsilon)$-factor of the optimal objective function the summary is required to determine whether or not the datasets are linearly separable. It is intuitively clear that no algorithm can achieve this in $\text{polylog}(|X|)$ space streaming algorithms aim for, as an adversarial chosen input sequence can first submit all points from the class labeled 1 and then submit a point from class -1. The last point can query the entire convex hull of the thus far submitted points, which has a description size of $O(|X|)$, see formal derivation of the lower bound in Section 3.

Instead of worst-case analysis, one might study suitable relaxations of the problem. Randomly permuting the input sequence of a stream often yields better results, yet even then linear classification remains infeasible. Another approach of relaxing the problem is smoothed analysis, first introduced by Spielman and Teng [24], where the performance of an algorithm is measured against a random, polynomial small Gaussian perturbation of any given input. Our approach is similar, but differs in that our space guarantees do not hold on expectation for a random perturbation, but for at least one perturbation. Though this guarantee is weaker compared to smoothed analysis, any algorithm with a feasible, i.e. sublinear smoothed-space complexity will likely require weaker performance measurement, see Section X. To formalize our claims, let $w \in \mathbb{R}^d$ be an arbitrary unit vector. For every point $x' \in X$ we denote the average squared distance of x' by $C_X(x') := \frac{1}{n}\sum_{x\in X} ||x - x'||_2^2$ and the w-projected average squared distance by $C_X(w, x') := \frac{1}{n}\sum_{x\in X}\left(w^{\mathrm{T}}(x - x')\right)^2$. We say a perturbation $f(X) : X \to \mathbb{R}^d$ of X is γ-bounded if for all unit vectors w and points $x \in X$ it holds $|\langle w, f(x) - x\rangle| \le \gamma \cdot \max_{x\in X} C_X(w, x)$. The set of all γ-bounded perturbations of X is denoted by $F_\gamma(X)$. We now aim to study the following problem in in a data stream:

Problem 1 (γ-relaxed Logistic Regression)**.** Let $X = X^+ \biguplus X^-$ be a d-dimensional set of points, labeled $\{-1, 1\}$, and let $\gamma > 0$ be a parameter. Let

$$L(w, x) := \begin{cases} \ln\left(1 + e^{-\langle w,x\rangle}\right) & \text{if } x \in X^+ \\ \ln\left(1 + e^{\langle w,x\rangle}\right) & \text{if } x \in X^- \end{cases}$$

and let $L(w) := \sum_{x\in X} L(w, x)$ be the error function of logistic regression. Then the γ-relaxed error function for a non-zero hyperplane w is defined as

$$L_\gamma(w) := \max_{f\in F_\gamma(X^+)} \sum_{x\in X^+} L(w, f(x)) + \max_{f\in F_\gamma(X^-)} \sum_{x\in X^-} L(w, f(x)).$$

The γ relaxed logistic regression aims to find a hyperplane w with

$$L(w) \le \min_{v\in\mathbb{R}^d} L_\gamma(w).$$

For this problem we are able to obtain the following results.

Theorem 1. *Let X be a set of n d-dimensional labeled points arriving in an insertion-only stream in arbitrary order and let $\epsilon, \gamma > 0$ be parameters. Then there exists a 1-pass algorithm such that for any vector w, we can sketch the γ-relaxed error function of Logistic Regression using $\tilde{O}\left(\frac{\log^2 n}{\epsilon^3} \cdot \frac{1}{\gamma^d}\right)$ memory with high probability.*

Theorem 2. *Let X be a set of n d-dimensional labeled points arriving in an insertion/deletion stream in arbitrary order and let $\epsilon, \gamma > 0$ be parameters. Then there exists a 1-pass algorithm such that for any vector w, we can sketch the γ-relaxed error function of Logistic Regression using $\tilde{O}\left(\frac{\log^2 n}{\epsilon^3} \cdot \left(\frac{\log n}{\gamma}\right)^d\right)$ memory with high probability.*

Related Work

Classification for Data Streams Various learning tasks such as clustering [7, 15], regression [13, 23] and classification have been studied in the streaming model. Specifically regarding binary classification, there has been extensive work on support vector machines. Assuming the data to be separable, support vector machines aim to find a hyperplane with maximum margin. In a streaming setting, algorithms produce summaries of the data called coresets, such that a hyperplane with maximum margin on the summary has an ϵ-approximate maximum margin on the original data. Coresets were originally proposed in [2] interchangeably with extent approximations (see Section 2), though now they have become a general design concept for various algorithmic problems, see a survey by Agarwal, Har-Peled and Varadarajan [4]. Roughly speaking, coresets generally summarize the data in a way that any query on the summary produces an $(1+\epsilon)$-approximate answer to the query on the entire dataset. Assuming the data to be separable, there exist maximum margin coresets for support vector machines [16]. More generally, the optimization of support vector machines can be formulated in terms of the minimum enclosing ball problem [25]. Coresets for the minimum enclosing ball problem have been widely studied and algorithms have either storage requirements exponential in d [2, 8], do not compute ϵ-approximate coresets [9, 21] or require multiple passes over the data [12, 25].

Online Gradient Descent The original perceptron algorithm by Rosenblatt [22] cycles through the dataset multiple times, updating the current model hyperplane w by $w_{new} := w_{old} + y_i \cdot x_i$ whenever w_{old} misclassifies a point x_i. Each pass through the data therefore takes time $O(n \cdot d)$ and the algorithm converges after a finite number of passes if the data is separable and obtains a hyperplane with ϵ-approximate maximum margin after $O(1/\epsilon^2)$ many passes. Clarkson, Hazan and Woodruff [12] gave an algorithm with $O(\log n/\epsilon^2)$ passes and a total running time of $O(\log n\epsilon^2(n+d))$. For non-separable data and using only one scan over the data, the number of misclassified points can be bounded with large (i.e. non-constant) approximation factors for some descent schemes with stopping criteria, see for instance [6, 14]. There also exists a very large body of work focusing on the performance of an online gradient descent for convex optimization problems such as logistic regression, see for instance [17, 26]. These bounds tend to better in terms of misclassified points and/or value of the objective function than those obtained for perceptron classification, but are also not independent of n.

2 Preliminaries

We denote a point set by X and the ith point by x_i. If labellings are required, the corresponding label is denoted by $y_i \in \{-1, 1\}$ and $X := X^- \biguplus X^+$, where X^- and X^+ the set of all points labeled -1 and 1, respectively. Further, some of the subroutines assume the points to placed on a d-dimensional discrete grid $\{-\Delta, \Delta\}^d$, so we similarly assume the points to be placed. We will also refer to x_i as the d-dimensional vector associated with the ith point. Further, let n be the number of points and d the dimension.

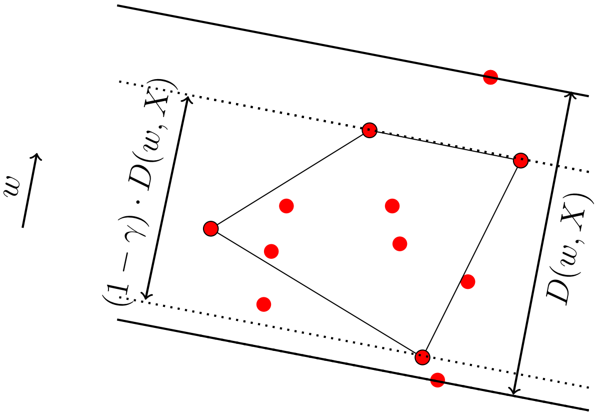

Figure 1: For any vector w, the extent of the polyhedron approximates the extent of the entire point set with respect to w.

We denote by $||v||_2 = \sqrt{\sum_{i=1}^{d} v_i}$ the l_2 norm of a vector v. The unit sphere S^{d-1} is the set of all vectors v with $||v||_2 = 1$ and $||M||_2 := \max_{v\in S^{d-1}} \frac{||Mv||_2}{||v||_2}$ the spectral norm of a matrix M. For a point set X, $H(X) := \{p \in \mathbb{R}^d \mid \exists x_1, x_2 \in X \wedge \lambda \in [0\ldots1]\ .\ p = \lambda x_1 + (1-\lambda)x_2)\}$ is defined as the convex hull of X and $|H(x)|$ the space required to store the convex hull of X. For a given statement E we denote by $\mathbb{1}[E]$ the indicator variable that is 1 if and only if the statement E holds. We characterize hyperplanes by their normal w. Offsets to a hyperplane can be modeled by adding a further dimension to w and an entry 1 appended to each point. The normals of a hyperplane are not not normalized i.e. $||w||_2$ can be arbitrary. Given a hyperplane w, we say the halfspace $h := \{p \in \mathbb{R}^d \mid \langle w, p\rangle \geq 0\}$ is induced by w. We denote the complementary halfspace of h by $c := \{p \in \mathbb{R}^d \mid \langle w, p\rangle \leq 0\}$. The mean of a point set X is denoted by $\mu_X = \frac{1}{n}\sum_{x\in X} x$ and the population variance $\sigma_X^2 = \frac{1}{n}\sum_{x\in X} ||x-\mu_X||_2^2$. We similarly define the w-projected population variance of X as $\sigma_X^2(w) := \frac{1}{n}\sum_{x\in X}\left(w^{\mathrm{T}}(x-\mu_X)\right)^2$. The average squared distance (ASD) of a point $a \in X$ is denoted by $C_X(a) := \frac{1}{n}\sum_{x\in X} ||x-a||_2^2$ and the w-projected average squared distance by $C_X(w,a) := \frac{1}{n}\sum_{x\in X}\left(w^{\mathrm{T}}(x-a)\right)^2$. We drop X from the subscript if the context is clear.

Our algorithms heavily rely on data structures approximating the extent in any direction. The extent of a point set in direction $w \in S^{d-1}$ is defined as $D(w,X) := \max_{x\in X} w^{\mathrm{T}}x - \min_{x\in X} w^{\mathrm{T}}x$. Intuitively, a subset of points approximating $D(w,X)$ can be thought of as a polyhedron with roughly the shape of the convex hull of a point set, see Figure 2.

Definition 1 (γ-Approximation of a Point Set)**.** Let $X \in \mathbb{R}^d$ be a set of points and $\gamma > 0$ a parameter. Then a subset $Q \subseteq X$ is an γ–approximation of X if for any vector $w \in \mathbb{R}^d \setminus \{\mathbf{0}\}$

$$D(w,X) \leq D(w,Q)\cdot(1+\gamma).$$

Sketching the extents of a point set in datastreams has been subject to extensive research. The algorithm with the currently smallest space requirement (and to our knowledge the only algorithm without any space dependency on n) of $O\left(\frac{1}{\gamma^{d-1}}\log^{d-1}\frac{1}{\gamma}\right)$ maintaining the extent in insertion-only datastreams is due to Chan [10]. The only known sketch that can be maintained for deletions from a discrete grid is due to Andoni and Nguyen [5] with a space complexity of $O\left(\left(\frac{\log n}{\gamma}\right)^d\right)$. γ-approximate sketches for the extent with polynomial dependency on d are not possible, see Agarwal and Sharathkumar [3]. For the remainder of this paper, we use such extent sketching algorithms as a black box, denoting by $APPROX(\gamma)$ the space required to maintain an γ-approximation. The time required to query the extent from a sketch is typically constant, though specific queries such as the vector u with minimum width have varying running times depending on the sketch.

The following lemma relates the extent of point set to our perturbation measure.

Lemma 1. *Let X be a set of points in $\mathbb{R}^d$ and let w be a hyperplane. Then*

$$\underset{x\in X}{max}\, C_X(w,x) \geq \frac{1}{4}D(w,X)^2$$

Proof. Recall the bias variance decomposition:

$$C_X(w,a) = \frac{1}{n}\sum_{x\in X}(w^T(x-a))^2 = \frac{1}{n}\sum_{x\in X}(w^T(x-\mu_X))^2 + (w^T(\mu_X-a))^2. \qquad (1)$$

Let $a,b \in X$ be the pair of points such that $D(w,X) = D(w,\{a,b\})$. Then

$$\max(D(w,\{a,\mu_X\}), D(w,\{b,\mu_X\})) \geq \frac{1}{2}D(w,\{a,b\}).$$

Assume without loss of generality that $D(w,\{a,\mu_X\}) \geq \frac{1}{2}D(w,\{a,b\})$. Then with Equation 1 we have

$$C_X(w,a) \;\; \geq \;\; (w^T(\mu_X-a))^2 = (D(w,\{a,\mu_X\}))^2 \geq \frac{1}{4}D(w,X)^2$$

□

3 Lower Bounds for Binary Linear Classification

We reduce the problem for logistic regression from the one-way 2-party indexing problem, where Alice has a binary bit string of length n elements and Bob an index $k \in \{1,\ldots,n\}$. Alice is allowed to send one message to Bob, whereupon Bob has to output the kth index. The number of bits of the transmitted message required by any randomized protocol succeeding with probability at least 2/3 over the random choices of the players is in $\Omega(n)$, see [1].

Lemma 2. *Let A be a subset of n points and let b be a single point in 2-dimensional Euclidean space arriving after another in a data stream. Then any single pass randomized algorithm deciding with probability $2/3$ whether $C(A)$ and b intersect requires at least $\Omega(n)$ space.*

Proof. Let $x \in \{0,1\}^n$ be Alice' bit string. For each $i \in \{1,\dots,n\}$, define the point $p_i = (\sin(\pi \cdot i/n), \cos(\pi \cdot 1/n))$. By construction, p_k is in the convex hull of any point set $\bigcup_{i \in I} p_i$ with $I \subseteq \{1,\dots,n\}$ if and only if $k \in I$. Alice computes the convex hull C over all points p_i with $x_i = 1$ and transmits a message to Bob. Similarly, Bob computes p_k and checks whether $p_k \in C$. □

Corollary 1. *Let A and B be two sets of a total n points in 2-dimensional space arriving in a data stream. Then any single pass randomized algorithm deciding with probability 2/3 whether A and B are linearly separable requires at least $\Omega(n)$ space.*

Proof. Since the error function of logistic regression approaches 0 for linearly separable data and is at least $\ln(2)$ otherwise, any streaming algorithm with multiplicative error and constant success probability requires $\Omega(n)$ space. □

For random order streams, we consider an instance where a single point $p \in X^-$ is in the convex hull $C(X^+)$. The expected position of p in the stream is $(n+1)/2$, hence we are still required to store $\Omega(n)$ space. For a regularization term $\lambda \cdot ||w|$ added to the error function, which prevents $||w||$ from reaching ∞, we can rescale the input without significantly increasing the description size of the points. If the set of input points are required to be distinct, i.e. if no two points have the same coordinates, Alice computes points $p_i = (\sin(\pi \cdot i/(n+1)), \cos(\pi \cdot i/(n+1)))$ for $i \in \{1, \dots n\}$ and additional points $a = (0,1) = (\sin(0), \cos(0))$ and $b = (1,0) = (\sin(1), \cos(0))$. Vice versa, Bob can alternatively compute an point p such that $p \in C \cup \{a,b\}$ if and only if $p \in C(\{a,b,p_k\})$.

4 Streaming Classification Algorithms

The main tool we use to sketch the point set is by counting points contained in halfspaces. We start by studying the following

Definition 2 (Approximate Perturbed Halfspace Counting)**.** Let X be a set of points in d dimensional space, $\epsilon, \gamma > 0$ be parameters and let F be the set of γ-bounded perturbations. Then Q is an (ϵ,γ)-*approximate halfspace sketch* if for every halfspace h in $\mathbb{R}^d$ we have

$$Q(|h \cap X|) \leq (1+\epsilon) \cdot \max_{f \in F(X)} |h \cap f(X)|.$$

Our aim is to sketch $|h \cap X|$ via uniform sampling. In general when uniformly sampling a subset A from X with probability p, we can estimate the number of points in $h \cap X$ as $\frac{|A \cap h|}{p}$. The major drawback of this approach is that if $|h \cap X|$ is very small compared to $|X|$, then the probability of sampling a point from $h \cap X$ is either negligibly small, or the number of sampled points will not be sublinear in n. Here the following simple observation is helpful.

Fact 1. Let X be a point set and let h be a halfspace induced by a hyperplane w. Then $h \cap X \neq \emptyset$ if and only if $h \cap H(X) \neq \emptyset$.

The key feature of our algorithm is the estimation of $|h \cap X|$ by counting intersections of convex hulls of multiple sampled subsets of X with h. We do not estimate $|h \cap X|$ directly, rather we choose a probability dependent on a choice number of possibly contained points t and infer from the number of intersections whether $|h \cap X|$ is sufficiently close to t or not.

When randomly picking each point with probability p, the probability of picking at least one point from a subset containing t elements is $1 - (1-p)^t$. Setting $1 - (1-p)^t = \frac{1}{2} \Leftrightarrow p = 1 - \frac{1}{2}^{\frac{1}{t}}$, we define a family of probabilities $p(t)$ for every possible number of points $t \in \{1, \dots, n\}$ contained in $h \cap M$. Using Chernoff-bounds, we will be able to tell whether a given halfspace h contains close to t and in particular less than $(1+\epsilon) \cdot t$ points from X. Any implementation of this estimation strategy would require storing the convex hull of each subsample of X, which leads to a prohibitively large amount of space requirement. However, if we allow perturbations on X, storing extent approximations become a feasible alternative.

The section is now organized as follows...

Lemma 3. *Let X be a set of points in $\mathbb{R}^d$ and let w be a hyperplane. Then there exists a constant*

Algorithm 1: HalfspaceCount

```
input: halfspace $h$, $\gamma$-approximations $A_j$ of $r$ subsets $X_j$ of $X$ independently
       drawn with probability $p(i)$, $\gamma$-approximation $A$ of $X$
if $h \cap A(X) = \emptyset$ then
    return 0;
$i \leftarrow 1$;
while $i \leq O(\log_{1+\epsilon} n)$ do
    $y \leftarrow 0$;
    for $j \leftarrow 1$ to $r$ do
        if $h \cap A_j] \neq \emptyset$ then
            $y \leftarrow y + 1$;
    if $y \leq (1 + \epsilon/4) \cdot \frac{r}{2}$ then
        return $p^{-1}(f(i))$;
    $i \leftarrow i + 1$;
```

Lemma 4. *Let X be a set of n points in d-dimensional space, let h be an arbitrary halfspace and let $\epsilon, \gamma, \delta > 0$ be parameters. Then there exists an algorithm such that with probability $1 - \delta$ we estimate a number a with*

$$a \leq (1+\epsilon) \cdot \underset{f}{max}\ |h \cap f(X)|,$$

where f is taken over all weakly γ-bounded perturbations of X. The algorithm uses $O(\frac{\log n}{\epsilon^3}(\log\log n + \log\frac{1}{\epsilon} + \log\frac{1}{\delta}) \cdot APPROX(\gamma))$ space, where $APPROX(\gamma)$ is the size of a γ-approximation.

Proof. We first analyse the estimation where we store the subsample X_j and subsequently argue the existence of a weakly γ-bounded perturbation f for which X_j may be summarized with A_j.

Let $p(i)$ be the probability for which Algorithm 1 produced an output. We declare a failure event F_i if, given that $(1-\epsilon)\cdot a_i \le |h\cap X| \le (1+\epsilon)\cdot a_i$ we have

$$\left|\sum_{j=1}^{r} \mathbb{1}[h\cap X_j \neq \emptyset] - \frac{r}{2}\right| \ge \epsilon/4\cdot\frac{r}{2}.$$

We say that the sketch fails indicated by the binary variable F if $F_i = 1$ for some i.

First, let $|h\cap X| \ge (1+\epsilon)\cdot a_i$. Then the probability of sampling one of the points and therefore with Observation 1, the probability $\mathbb{P}[h\cap X_j\neq\emptyset]$ is at least

$$1-(1-p(i))^{(1+\epsilon)\cdot a_i} = 1-(1-(1-\frac{1}{2}^{\frac{1}{a_i}}))^{(1+\epsilon)\cdot a_i} = 1-\frac{1}{2}^{1+\epsilon}.$$

Hence, $\mathbb{E}[\sum_{j=1}^{r}\mathbb{1}[h\cap X_j\neq\emptyset]] \ge \left(1-\frac{1}{2}^{1+\epsilon}\right)r$. Using the Chernoff bound and some straightforward, though tedious calculation, we can bound the deviation probability of $\sum \mathbb{1}[h\cap X_j\neq\emptyset]$ around its expectation by

$$\begin{aligned}
&\mathbb{P}\left[\sum_{i=1}^{r}\mathbb{1}[h\cap X_j\neq\emptyset] \le (1+\epsilon/4)\cdot\frac{r}{2}\right]\\
&= \mathbb{P}\left[\sum_{i=1}^{r} Y_{i,j} \le \left(1-\left(1-\frac{1+\epsilon/4}{2\cdot\left(1-\frac{1}{2}^{1+\epsilon}\right)}\right)\right)\left(1-\frac{1}{2}^{1+\epsilon}\right)r\right]\\
&\le \exp\left(-\left(1-\frac{1+\epsilon/4}{2\cdot\left(1-\frac{1}{2}^{1+\epsilon}\right)}\right)^2\frac{1}{2}\left(1-\frac{1}{2}^{1+\epsilon}\right)r\right)\\
&= \exp\left(-\left(\frac{1-\frac{1}{2}^{\epsilon}-\epsilon/4}{2\cdot\left(1-\frac{1}{2}^{1+\epsilon}\right)}\right)^2\frac{1}{2}\left(1-\frac{1}{2}^{1+\epsilon}\right)r\right)\\
&\le \exp\left(-\left(\frac{\left(\ln(2)-\frac{\ln^2(2)}{2}-\frac{1}{4}\right)\epsilon}{2\cdot\left(1-\frac{1}{2}^{1+\epsilon}\right)}\right)^2\frac{1}{2}\left(1-\frac{1}{2}^{1+\epsilon}\right)r\right) \qquad (2)\\
&\le \exp\left(-\left(\frac{\left(\ln(2)-\frac{\ln^2(2)}{2}-\frac{1}{4}\right)^2\epsilon^2}{8\cdot\left(1-\frac{1}{2}^{1+\epsilon}\right)}\right)r\right) \le \exp\left(-\frac{\epsilon^2}{98}\cdot r\right).
\end{aligned}$$

where equation 2 follows from the Taylor expansion

$$0+\left(\ln(2)-\frac{1}{3}\right)x-\frac{\ln^2(2)}{2}x^2+\sum_{k=3}^{\infty}\frac{\ln^k(2)}{k!}x^k(-1)^{k+1}$$

of $1 - \frac{1}{2}^x - x/3$ at $x = 0$.

The case where $|h \cap X|$ is less than $(1-\epsilon) \cdot a_i$ can be argued analogously. Since a_i is exponentially growing in $1+\epsilon$, we have $O(\log n/\epsilon)$ choices of i. By setting $r \in O\left(\frac{\log\log n + \log\frac{1}{\epsilon} + \log\frac{1}{\delta}}{\epsilon^2}\right)$ the probability that our algorithm 'fails' is

$$\mathbb{P}[F] = \mathbb{P}\left[\bigcup_i F_i\right] \leq 3\log n/\epsilon \cdot \exp\left(-\frac{\epsilon^2}{98} \cdot r\right) \leq \delta.$$

Now we return to the original algorithm using γ-approximations. Let w be the hyperplane inducing the halfspace h. Let a_i be the estimated number of points contained in h. Since A_j is a γ-approximation of X_j, there exists a function $f_j : X \to \mathbb{R}^d$ such that for any point $x \in X_j \backslash H(A_j)$ we have $f_j(x) \in H(A_j)$ and $||f_j(x) - x||_2 \leq 2\gamma \cdot D(w, X_j)^2 \leq 2\gamma \cdot D(w, X)^2$ and any point $x \notin X_j \backslash H(A_j)$ we have $f_j(x) = x$. Further by Lemma 1

$$D(w, X)^2 \leq 4 \cdot \max_{x \in X} C_X(w, x).$$

Define $f : X \to \mathbb{R}^d$ such that $f(x) = \underset{f_j(x)}{\operatorname{argmax}} \left(w^{\mathrm{T}} f_j(x)\right)^2$. Then f is weakly 4γ-bounded and conditioned on the event that the algorithm does not fail

$$a_i \leq (1+\epsilon) \cdot \sum_{x \in X} \mathbb{1}[w^{\mathrm{T}} f(x) > 0] \leq (1+\epsilon) \cdot \max_{f'} \sum_{x \in X} \mathbb{1}[w^{\mathrm{T}} f'(x) > 0],$$

where f' is taken over all weakly 4γ-bounded perturbations.

The total space requirement consists of r γ-approximations, each of size $APPROX(\gamma)$ for all $O(\log n/\epsilon)$ choices of i. □

To complete the proof of Theorems 1 and 2, we now show how $L(w)$ can be written in terms of points contained in halfspaces.

Lemma 5. *Let X be a set of points labeled -1 or 1. Then for all $w \in \mathbb{R}^d$, $L(w)$ can be approximated up to an $(1 \pm \epsilon)$ factor by counting points contained in $O(n^{d+1})$ many halfspaces up to a $(1+\epsilon)$ factor.*

Proof. If w has infinite norm, then $L(w) = \infty$, if there exists a misclassified point and $L(w) = 0$ otherwise.

We therefore assume from now on that w has finite norm. Let $p_{\max} := \underset{x \in X}{\operatorname{argmax}}\ w^{\mathrm{T}} x$ and $p_{\min} := \underset{x \in X}{\operatorname{argmin}}\ w^{\mathrm{T}} x$. Since the norm of w is finite, the interval $(p_{\min}, p_{\max})$ has finite length and the derivative of $\ln(1+e^x)$ is finite within that interval. Hence, we can approximate $\ln(1+e^x)$ within that interval with finite many expansions of 0th degree. For each expansion at point $a \in (p_{\min}, p_{\max})$, consider the halfspace $h := \{p \in \mathbb{R}^d \mid \frac{1}{||w||} \cdot w^{\mathrm{T}} p \geq$

$a\}$. The contribution $L(w,x)$ of every point x to $L(w)$ can be decomposed into

$$\begin{aligned} L(w,x) &\geq (1-\epsilon)\cdot(\ln(1+e^{a_1})+\sum_{j=2}^{\ell}(\ln(1+e^{a_j})-\ln(1+e^{a_{j-1}}))\cdot \mathbb{1}[w^{\mathrm{T}}x\geq a_j]) \quad (3)\\ L(w,x) &\leq (1+\epsilon)\cdot(\ln(1+e^{a_1})+\sum_{j=2}^{\ell}(\ln(1+e^{a_j})-\ln(1+e^{a_{j-1}}))\cdot \mathbb{1}[w^{\mathrm{T}}x\geq a_j]), \end{aligned}$$

where a_j is the jth expansion point, ℓ is finite and $a_1 = p_{\min}$. Using Equation 3, we have

$$L(w) \leq (1+\epsilon)(\ln(1+e^{a_1})\cdot \sum_{x\in X}\mathbb{1}[w^{\mathrm{T}}x\geq a_1] + \sum_{j=1}^{\ell}(a_j - a_{j-1})\cdot \mathbb{1}[w^{\mathrm{T}}x\geq a_j]),$$

and a similar lower bound.

Further, the number of subsets of n points induced by a d dimensional hyperplane is in $O(n^{d+1})$, see for instance [11] on range spaces and VC-dimension. With a datastructure querying the (approximate) number of points in any halfspace, we can approximate $\sum_{x\in X}\mathbb{1}[w^{\mathrm{T}}x \geq a]$ for any expansion point a up to an $1\pm\epsilon$ factor, resulting in an $(1+\epsilon)^2 \leq (1+3\epsilon)$ approximation of $L(w)$. □

All random coins used by the algorithm can be sampled from a random seed of length $k\cdot\log n$ using pseudorandom generators for bounded space computation, see Nisan [20], where k is the size of our data structure.

For the perceptron classifier, where the loss function is defined as

$$L(w,x_i) := \begin{cases} 0 & \text{if } y_i\cdot\langle w,x_i\rangle > 0\\ 1 & \text{if } y_i\cdot\langle w,x_i\rangle \leq 0 \end{cases},$$

that is the loss function is essentially the points contained on the wrong side of the hyperplane, the bounds carry over:

Corollary 2. *Let X be a set of n d-dimensional labeled points arriving in an insertion/deletion stream in arbitrary order and let $\epsilon,\gamma>0$ be parameters. Then there exists a 1-pass algorithm such that for any hyperplane w, we can sketch the γ-relaxed error function of Perceptron Classification using $\tilde{O}\left(\frac{\log n^2}{\epsilon^3}\cdot\left(\frac{\log n}{\gamma}\right)^d\right)$ memory with high probability.*

5 Conclusion and Discussion

In this paper we introduced a relaxed model for the previously infeasible task of learning binary classifiers from datasets. The model is similar to smoothed analysis in that we do not consider worst-case inputs but perturbations, but differs in a few crucial details. Unlike smoothed analysis we do not allow random perturbations, but show that there exists a perturbation for which we can produce a feasible summary, making our model seemingly weaker. Perhaps smoothed analysis can be applied in some way to measure the expected space complexity of a summary, but the probability by which we are required to store (close to) the entire point set is constant. Therefore, if smoothed analysis can indeed by applied in some way, it will require other restrictions to handle the probability space over all perturbations or a different way of measuring the performance of an algorithm.

Our algorithm exhibits two major drawbacks. Firstly, our perturbation measure is not very robust with respect to outliers. Secondly, though we are able to query the objective function of every candidate solution, we are not able to efficiently compute is on the summary as the point set is store implicitly. Future work will address both questions.

References

[1] Farid Ablayev. Lower bounds for one-way probabilistic communication complexity and their application to space complexity. *Theoretical Computer Science*, 157(2):139 – 159, 1996.

[2] P. Agarwal, S. Har-Peled, and K. Varadarajan. Approximating extent measures of points. *J. ACM*, 51(4):606–635, 2004.

[3] P. Agarwal and R. Sharathkumar. Streaming algorithms for extent problems in high dimensions. In *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2010.

[4] Pankaj K. Agarwal, Sariel Har-peled, Kasturi, and R. Varadarajan. Geometric approximation via coresets. In *Combinatorial and Computational Geometry, MSRI*, pages 1–30. University Press, 2005.

[5] A. Andoni and H. Nguyen. Width of points in the streaming model. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 447–452. SIAM, 2012.

[6] P. Auer and M. Warmuth. Tracking the best disjunction. *Machine Learning*, 32(2):127–150, 1998.

[7] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings of the 34th ACM Symposium on Theory of Computing (STOC)*, pages 250–257, 2002.

[8] T. Chan. Faster core-set constructions and data stream algorithms in fixed dimensions. In *Comput. Geom. Theory Appl*, pages 152–159, 2003.

[9] T. Chan and V. Pathak. Streaming and dynamic algorithms for minimum enclosing balls in high dimensions. In *Proceedings of the 12th international conference on Algorithms and data structures*, WADS'11, pages 195–206, Berlin, Heidelberg, 2011. Springer-Verlag.

[10] Timothy M. Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Comput. Geom.*, 35(1-2):20–35, 2006.

[11] Bernard Chazelle. *The discrepancy method - randomness and complexity.* Cambridge University Press, 2001.

[12] K. Clarkson, E. Hazan, and D. Woodruff. Sublinear optimization for machine learning. *J. ACM*, 59(5):23, 2012.

[13] K. Clarkson and D. Woodruff. Numerical linear algebra in the streaming model. In *STOC*, pages 205–214, 2009.

[14] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Mach. Learn.*, 37(3):277–296, December 1999.

[15] S. Har-Peled and A. Kushal. Smaller coresets for k-median and k-means clustering. *Discrete & Computational Geometry*, 37(1):3–19, 2007.

[16] Sariel Har-Peled, Dan Roth, and Dav Zimak. Maximum margin coresets for active and noise tolerant learning. In *IJCAI*, pages 836–841, 2007.

[17] E. Hazan and S. Kale. Beyond the regret minimization barrier: an optimal algorithm for stochastic strongly-convex optimization. *Journal of Machine Learning Research - Proceedings Track*, 19:421–436, 2011.

[18] Ravindran Kannan and Santosh Vempala. Nimble algorithms for cloud computing. *CoRR*, abs/1304.3162, 2013.

[19] S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.

[20] N. Nisan. Pseudorandom generators for space-bounded computations. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, STOC '90, pages 204–212, New York, NY, USA, 1990. ACM.

[21] P. Rai, H. Daumé, and S. Venkatasubramanian. Streamed learning: one-pass svms. In *Proceedings of the 21st international jont conference on Artifical intelligence*, IJCAI'09, pages 1211–1216, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[22] Frank Rosenblatt. The perceptron: A perceiving and recognizing automaton. Technical report, 1957.

[23] T. Sarlós. Improved approximation algorithms for large matrices via random projections. In *FOCS*, pages 143–152, 2006.

[24] Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, 51(3):385–463, 2004.

[25] I. Tsang, J. Kwok, and P. Cheung. Core vector machines: Fast svm training on very large data sets. *J. Mach. Learn. Res.*, 6:363–392, December 2005.

[26] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pages 928–936, 2003.