Surrogate Loss Minimization

Alon Cohen

Submitted in partial fulfilment of the requirements
of the degree of Master of Science

Under the supervision of
Prof. Daphna Weinshall
Prof. Shai Shalev-Shwartz

September 2014

Rachel and Selim Benin
School of Computer Science and Engineering
The Hebrew University of Jerusalem
Israel

Abstract

We survey the problem of learning linear models, in the binary and multiclass settings. In both cases, our goal is to find a linear model with least probability of mistake.

This problem is known to be NP-hard and even NP-hard to learn improperly (under relevant assumptions). Nonetheless, under certain assumptions about the input the problem has an algorithm with worst-case polynomial time complexity.

At first glance these assumptions seem to vary greatly. Starting from the realizable assumption, which entails that the labeling is deterministic and can be realized by a linear function, and ending with simply the existence of a predictor with margin and a low error rate. However all of these methods can be seen as *generalized linear models*, namely the optimal classifier can be used to estimate the distribution of the labels given any example.

On a different note, all of these methods are based on convex optimization and they are are generally split into two groups, offline, or batch, methods and online methods. Offline methods are based on minimizing a *surrogate* convex loss function over a sample taken from the distribution. This function is called a surrogate since it replaces the zero-one loss (number of mistakes). Online methods include Perceptron-like algorithms and can all be seen as special cases of online linear optimization. We show that all known online methods are in fact approximations to offline methods.

Contents

List of Figures

Chapter 1

Introduction

1.1 Learning linear models

This section briefly introduces statistical learning and linear models for binary and multiclass classification.

Let us start with an introduction to statistical learning. We are given a *sample space* $\mathcal{X}$, a *label space* $\mathcal{Y}$ and an unknown distribution $\mathcal{D}$ over the product space $\mathcal{X} \times \mathcal{Y}$. The goal is to find a function $h : \mathcal{X} \to \mathcal{Y}$ that minimizes the probability of error,

$$\Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$$

Clearly, in order to estimate the function h we need to have some kind of access to the distribution $\mathcal{D}$. This access is realized by an oracle which samples independently from $\mathcal{D}$.

Let us say a learning problem is *learnable* if there exists an algorithm A with access to the oracle that returns a function h which satisfies

$$\Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y] \leq \inf_{h'} \Pr_{(x,y)\sim\mathcal{D}}[h'(x) \neq y] + \epsilon$$

for a small $\epsilon > 0$ with high probability.

Note that the goal of A is to estimate the optimal h from a finite number of examples drawn from the distribution. As such, the No-Free-Lunch theorem tells us that if we let h be any function, then for any algorithm A there exists distributions $\mathcal{D}$ such that A will fail to estimate the optimal h with high probability.

One approach for overcoming this difficulty is to say that h belongs to a hypothesis class $\mathcal{H}$. Then the output of A is compared to the optimal function in $\mathcal{H}$.

A popular hypothesis class is a class of linear functions. It is here that this section will split into two parts, binary and multiclass classification.

1.1.1 Binary

Here we assume that $\mathcal{X} \subseteq \mathbb{R}^d$, $\mathcal{Y} = \{-1, 1\}$ and that

$$\mathcal{H} = \{\mathbf{x} \mapsto \operatorname{sign}(\langle \mathbf{w}, \mathbf{x} \rangle) : \mathbf{w} \in \mathbb{R}^d\}$$

Let $\mathbf{w} \in \mathbb{R}^d$ by a vector which represents a halfspace. Any example $\mathbf{x} \in \mathcal{X}$ satisfying $\langle \mathbf{w}, \mathbf{x} \rangle > 0$ is classified as 1 and otherwise as -1. In fact, $\mathbf{w}$ is perpendicular to the hyperplane which bounds the halfspace.

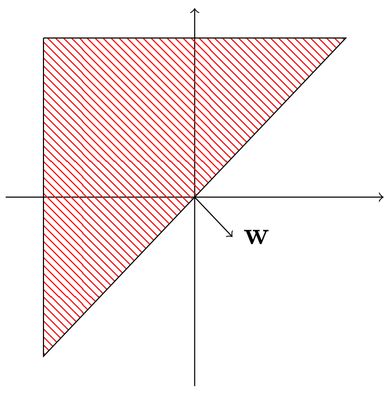


Figure 1.1: An example of a halfspace in $\mathbb{R}^2$. The red area is one which is classified as -1 and the rest of the space is classified as 1.

1.1.2 Multiclass

Once again assume that $\mathcal{X} \subseteq \mathbb{R}^d$ except now $\mathcal{Y} = [k] := \{1, ..., k\}$, k is the number of classes, and

$$\mathcal{H} = \{\mathbf{x} \mapsto \arg\max_{z \in [k]} (W\mathbf{x})_z : W \in \mathbb{R}^{k \times d}\}$$

For a geometric intuition, the space of $\mathbb{R}^d$ is split into k tangential cones. Each y'th cone is pointed and convex and contains all points satisfying a set of linear inequalities,

$$\forall z \in [k]\backslash\{y\}\ (W\mathbf{x})_y > (W\mathbf{x})_z$$

Let $\mathbf{e}_z$ is the z'th canonical basis vector of $\mathbb{R}^k$. An equivalent viewpoint is that for every two classes, z and y, the halfspace represented by $W^\top(\mathbf{e}_z - \mathbf{e}_y)$ classifies between the two classes. For every example of class z we have $\langle W^\top(\mathbf{e}_z - \mathbf{e}_y), \mathbf{x}\rangle > 0$ and for every example of class y we have $\langle W^\top(\mathbf{e}_z - \mathbf{e}_y), \mathbf{x}\rangle \leq 0$.

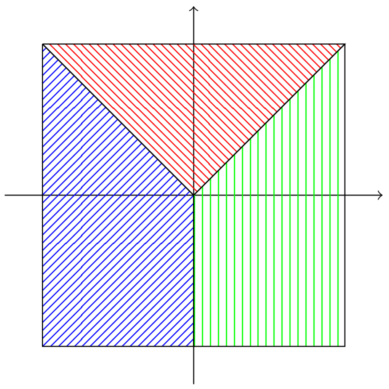


Figure 1.2: An example of a multiclass linear classifier in $\mathbb{R}^2$. The red, green and blue areas represent a partition of the space into three classes.

1.2 Hardness of learning

In this section we will state results showing that under relevant assumptions learning linear models is computationally hard. The section will focus on learning halfspaces, but since halfspaces are a special class of multiclass linear models the results hold for multiclass as well.

Guruswami and Raghavendra ([21]) have shown that given $\epsilon, \delta > 0$ and a set of examples from $\{0,1\}^d$, it is hard to distinguish between the case that $(1-\epsilon)$-fraction could be explained by a halfspace and the case that $(1/2+\delta)$-fraction could be explained by a halfspace. Since any δ-weak learning algorithm could be used to distinguish between the two cases, the result follows. In a follow-up paper ([19]) Feldman et al. have shown an analogous result for when the sample space is $\mathbb{Q}^d$.

Both results are based on hardness of maximizing satisfiability of linear inequalities. They prove hardness of proper learning, meaning when the algorithm has to return a hypothesis which is itself a halfspace.

The work of Daniely et al. ([17]) is based on a stronger assumption than P $\neq$ NP, that it is hard to refute random SAT formulas. They show that it is hard to distinguish between a realizable sample and a random one. These random samples are unrealizable with high probability and this entails that it is hard to distinguish between a realizable sample and an unrealizable one (w.h.p.). Specifically, any improper weak learning algorithm can be used to distinguish between the different kinds of samples and the results follows.

1.3 A motivating example

After presenting the problem and showing that it is generally hard, we now tend to present how machine learning literature had tried to overcome this hardness. As a motivating example let us now present the idea behind soft-SVM ([14]).

Define the zero-one loss:

$$\ell_{0-1}(\alpha) = \mathbb{1}_{[\alpha \le 0]}$$

For a halfspace $\mathbf{w}$ and an example $(\mathbf{x}, y)$, $\ell_{0-1}(y\langle \mathbf{w}, \mathbf{x}\rangle)$ is 1 if $\mathbf{w}$ mistakes on $(\mathbf{x}, y)$ and zero otherwise. Since the expected zero-one loss is the probability of mistake, one would like to find a halfspace with minimum zero-one loss.

The above is NP-hard in general. The approach of soft-SVM is to replace the zero-one loss with a different loss function, the hinge loss:

$$\ell_{\text{hinge}}(\alpha) = \max\{0, 1 - \alpha\}$$

The hinge loss is convex, bounded from below and we can find its minima efficiently. Another important property is that it upper bounds the zero-one loss. Since it replaces the zero-one loss it is one type of a *surrogate loss function.*

If we could learn with respect to the hinge loss, meaning we could find a halfspace $\mathbf{w}$ for which

$$\mathbb{E}[\ell_{\text{hinge}}(y\langle \mathbf{w}, \mathbf{x}\rangle)] \le \min_{\mathbf{w}'} \mathbb{E}[\ell_{\text{hinge}}(y\langle \mathbf{w}', \mathbf{x}\rangle)] + \epsilon$$

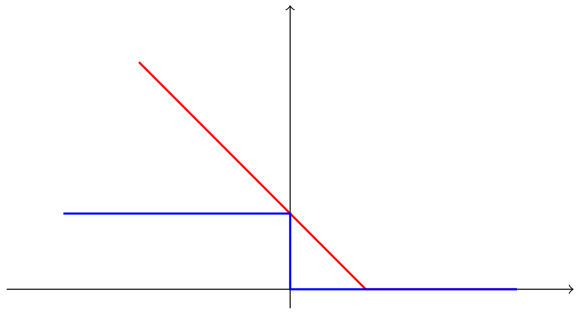


Figure 1.3: In red the hinge loss and in blue the zero-one loss.

then also

$$\mathbb{E}[\ell_{0-1}(y\langle \mathbf{w}, \mathbf{x}\rangle)] \leq \min_{\mathbf{w}'} \mathbb{E}[\ell_{\text{hinge}}(y\langle \mathbf{w}', \mathbf{x}\rangle)] + \epsilon$$

What does this mean?

- If the minimizer of the hinge loss is minimizer of the zero-one loss then we have found the optimal halfspace.
- If the minimum expected hinge loss is small, then we have found a good classifier.

That brings up the question, can we guarantee any of these conditions? If we can, under what assumptions? These are the type of questions that we would try and answer for the remainder of this thesis.

1.4 Contributions

Our contributions are as following:

1. We have compiled a survey of major known results, both positive and negative, regarding surrogate convex loss minimization.
2. We show simple algorithms for learning halfspaces with random noise. Both in the batch (section 4.4) and online (chapter 9) settings.
3. In chapter 11 we show that known online algorithms for learning linear models are in fact approximations of the batch case, i.e. minimizing a surrogate convex loss over a sample from the distribution.

1.5 Preliminaries

1.5.1 Notation

We will denote scalars as non-capital letters, vectors as bold non-capital letters and matrices and random variables as capital letters. Denote the indicator function as $\mathbb{1}_{[P]}$ for a predicate P, which is 1 if P holds and 0 otherwise.

Our examples are taken from $\mathbb{R}^d$. Abbreviating $[k]$ for $\{1, ..., k\}$, the labels are taken from $\{-1, 1\}$ in the binary case and from $[k]$ in the multiclass case, where k is the number of classes.

Loss functions are denoted as ℓ subscripted by their name. We find it easier to use a few equivalent notations for loss functions, each useful in different chapters:

1. In the binary case as functions from the real line, for example: $\ell_{\text{hinge}}(\alpha) = \max\{0, 1 - \alpha\}$. This form is easier to visualize and for analyzing notions of consistency. Given an example $(\mathbf{x}, y)$ for a halfspace $\mathbf{w} \in \mathbb{R}^d$ the actual loss is $\ell(y\langle \mathbf{w}, \mathbf{x}\rangle)$.

2. In the multiclass case as functions from $\mathbb{R}^k$. Each label $y \in [k]$ has a loss associated with it, denoted as $\ell^{(y)}$. For example: $\ell^{(y)}_{\text{cs}}(\boldsymbol{\alpha}) = \max_{z\in[k]}\{\mathbb{1}_{[z\neq y]} + \boldsymbol{\alpha}_z - \boldsymbol{\alpha}_y\}$. The actual loss is $\ell^{(y)}(W\mathbf{x})$ for a matrix $W \in \mathbb{R}^{k\times d}$.

3. As functions from $\mathbb{R}^d$ (or $\mathbb{R}^{k\times d}$) parametrized by an example, e.g. $\ell_{\text{hinge}}(\mathbf{w}; (\mathbf{x}, y)) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}$. This form is useful as some loss functions cannot be written in the previous forms.

4. In the online setting, given the t'th example $(\mathbf{x}^{(t)}, y^{(t)})$ we will abbreviate $\ell^{(t)}(\mathbf{w})$ for $\ell^{(t)}(\mathbf{w}; (\mathbf{x}^{(t)}, y^{(t)}))$.

1.5.2 Margin

In some parts of this manuscript we will assume the existence margin. Most researchers in machine learning know the definition of margin in the realizable case, but here we will define it more generally.

Assumption 1 (Margin)**.** Define the margin loss:

$$\ell_{\text{margin}}(\alpha) = \mathbb{1}_{[\alpha\leq 1]}$$

Suppose for all $\mathbf{x} \in \mathcal{X}$, $\|\mathbf{x}\| \leq 1$, we will say the distribution is separable with error ν and margin γ if there exists a halfspace $\|\mathbf{w}\| = 1/\gamma$ with an expected margin loss of ν.

Equivalently, if instead we have a bound $\|\mathbf{x}\| \leq R$ for all $\mathbf{x} \in \mathcal{X}$ and $\|\mathbf{w}\| = B$ then we say that the margin is $1/(RB)$.

Part I

Surrogate Loss Minimization

Chapter 2

Consistency

The notion of consistency deals with statistical properties of surrogate loss functions. We would like to show that if the expected surrogate loss converges to its minimum value then the same is true for the zero-one loss.

We will allow the hypotheses to be any function, not only linear. This point will prove itself to be crucial to consistency results. We will discuss this further in the last part of the section.

From now on this chapter will split into two, discussing the binary and multiclass cases. For each case, we will present different loss functions and show whether they are consistent or not.

2.1 Binary

In this section we will survey the papers [38, 5].

Definition 1 (Consistency)**.** A surrogate loss $\ell : \mathbb{R} \to \mathbb{R}$ is *consistent* if for any distribution and for any sequence of functions $f_n : \mathcal{X} \to \mathbb{R}$ such that

$$\lim_{n\to\infty} \mathbb{E}\ell(yf_n(\mathbf{x})) = \inf_{f'} \mathbb{E}\ell(yf'(\mathbf{x}))$$

then

$$\lim_{n\to\infty} \mathbb{E}\ell_{0-1}(yf_n(\mathbf{x})) = \inf_{f'} \mathbb{E}\ell_{0-1}(yf'(\mathbf{x}))$$

2.1.1 Classification Calibration

Let us now define the notion of classification calibration and show equivalence to consistency.

Start by considering the expected conditional loss

$$\begin{aligned}&\mathbb{E}[\ell(Yf(X))|X=\mathbf{x}] = \\ &\quad \Pr[Y=1|X=\mathbf{x}]\ell(f(\mathbf{x})) + \Pr[Y=-1|X=\mathbf{x}]\ell(-f(\mathbf{x}))\end{aligned}$$

and denote $\eta = \Pr[Y = 1|X = \mathbf{x}]$. Since f can be any function $f(\mathbf{x})$ can attain any value and so the $\mathbf{x}$ has no influence on the value above. Using this idea we can define the general conditional risk

$$C_\eta(\alpha) = \eta\ell(\alpha) + (1-\eta)\ell(-\alpha)$$

Intuitively, for a "good" loss function we expect that the sign of a α attaining the minimum of C_η to correspond with the most probable label. More concretely, if $\eta > 1/2$ then any minimal α must be positive and if $\eta < 1/2$ then that α must be negative.

This is formalized by the following definitions. First, define the functions H and H^-:

$$\begin{aligned}H(\eta) &= \inf_\alpha C_\eta(\alpha) \\ H^-(\eta) &= \inf_{\alpha:\alpha(\eta-1/2)\leq 0} C_\eta(\alpha)\end{aligned}$$

where H is the minimum value of the general conditional risk for a given η and H^- is the minimum value conditioned on the sign of α not corresponding with the most probable label.

Definition 2 (Classification Calibration)**.** A loss function $\ell : \mathbb{R} \to \mathbb{R}$ is *classification calibrated* if for all $\eta \in [0, 1]$ such that $\eta \neq 1/2$

$$H^-(\eta) > H(\eta)$$

The proof of the following theorem can be found in the papers cited at the beginning of the chapter.

Theorem 1. *The following statements are equivalent:*

1. *ℓ is classification calibrated.*

2. *ℓ is consistent.*

2.1.2 Simplified conditions

We will state a couple of results which simplify the characterization of classification calibration. The first is a theorem stating a necessary and sufficient condition for classification calibration of convex loss functions.

Theorem 2. *Let ℓ be convex. Then ℓ is classification calibrated if and only if it is differentiable at* 0 *and* $\ell'(0) < 0$.

The following lemma states that any nonnegative classification calibrated loss must upper bound the zero-one loss.

Lemma 1. *If $\ell : \mathbb{R} \to [0, \infty)$, then there exists a $\gamma > 0$ such that $\gamma\ell(\alpha) \geq \mathbb{1}_{[\alpha \leq 0]}$ for all $\alpha \in \mathbb{R}$.*

2.1.3 Examples

Based on the previous results, let us review some common consistent loss functions.

Exponential $\ell_{\text{exp}}(\alpha) = \exp(-\alpha)$

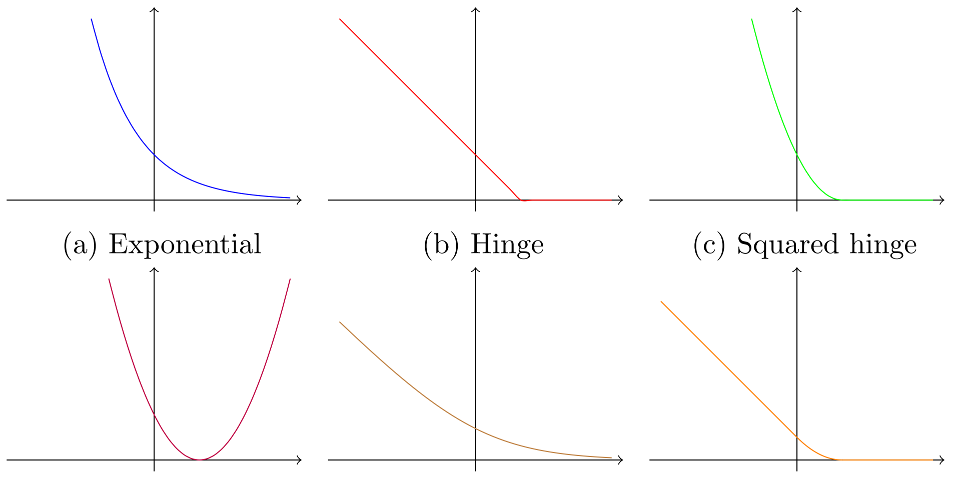
Hinge $\ell_{\text{hinge}}(\alpha) = \max\{0, 1 - \alpha\}$

Squared Hinge $\ell_{\text{sqr-hinge}}(\alpha) = \max\{0, 1 - \alpha\}^2$

Square $\ell_{\text{square}}(\alpha) = (1 - \alpha)^2$

Logistic $\ell_{\text{logistic}}(\alpha) = \log(1 + \exp(-\alpha))$

One-sided Huber $\ell_{\text{osh}}(\alpha) = \begin{cases} 0, & \alpha \geq 1 \\ \frac{1}{2}(1-\alpha)^2, & 0 \leq \alpha < 1 \\ \frac{1}{2} - \alpha, & \text{otherwise} \end{cases}$


(a) Exponential (b) Hinge (c) Squared hinge

(d) Square (e) Logistic (f) One-sided Huber

Figure 2.1: Consistent binary loss functions

2.2 Multiclass

This section will present the papers [37, 35].

We will consider functions of the form $\ell^{(y)} : \mathbb{R}^k \to \mathbb{R}$, where k is the number of classes and $y \in [k]$ is a label. Define the zero-one loss as:

$$\ell_{0-1}^{(y)}(\boldsymbol{\alpha}) = \mathbb{1}_{[\exists z \in [k] \;\; \boldsymbol{\alpha}_z \geq \boldsymbol{\alpha}_y]}$$

Let us define the notion of consistency similarly to the binary case.

Definition 3 (Consistency)**.** A family of surrogate loss functions $\{\ell^{(y)} : \mathbb{R}^k \to \mathbb{R}\}_{y=1}^k$ is *consistent* if for any distribution and for any sequence of functions $\mathbf{f}_n : \mathcal{X} \to \mathbb{R}^k$ such that

$$\lim_{n\to\infty} \mathbb{E}\ell^{(y)}(\mathbf{f}_n(\mathbf{x})) = \inf_{\mathbf{f}'} \mathbb{E}\ell^{(y)}(\mathbf{f}'(\mathbf{x}))$$

then

$$\lim_{n\to\infty} \mathbb{E}\ell_{0-1}^{(y)}(\mathbf{f}_n(\mathbf{x})) = \inf_{\mathbf{f}'} \mathbb{E}\ell_{0-1}^{(y)}(\mathbf{f}'(\mathbf{x}))$$

2.2.1 Classification Calibration

Define the expected conditional loss as:

$$C_\ell(\boldsymbol{\alpha}, \mathbf{q}) = \sum_{y=1}^{k} \mathbf{q}_y \ell^{(y)}(\boldsymbol{\alpha})$$

and the suboptimality with respect to the zero-one loss:

$$W(\boldsymbol{\alpha}, \mathbf{q}) = C_{\ell_{0-1}}(\boldsymbol{\alpha}, \mathbf{q}) - \inf_{\boldsymbol{\alpha'}} C_{\ell_{0-1}}(\boldsymbol{\alpha}', \mathbf{q})$$

then similarly to the binary case, we can define the notion of classification calibration:

Definition 4 (Classification Calibration)**.** A family of surrogate loss functions $\{\ell^{(y)} : \mathbb{R}^k \to \mathbb{R}\}_{y=1}^k$ is *classification calibrated* if for any $\epsilon > 0$ and any distribution $\mathbf{q}$ over $[k]$:

$$\inf_{\boldsymbol{\alpha}: W(\boldsymbol{\alpha}, \mathbf{q}) > \epsilon} C_\ell(\boldsymbol{\alpha}, \mathbf{q}) > \inf_{\boldsymbol{\alpha'}} C_\ell(\boldsymbol{\alpha}', \mathbf{q})$$

Intuitively, a loss function is classification calibrated if any $\boldsymbol{\alpha}$ which attains the minimum expected conditional loss has $\boldsymbol{\alpha}_y > \boldsymbol{\alpha}_z$ for all $z \neq y$.

To complete the analogy to the binary case, we have the following theorem.

Theorem 3. *The following are equivalent:*

1. $\{\ell^{(y)}\}_{y=1}^k$ *is classification calibrated.*

2. $\{\ell^{(y)}\}_{y=1}^k$ *is consistent.*

2.2.2 Examples

Let us review some common multiclass loss functions, the first three are not consistent and the rest are consistent.

Crammer and Singer [15] $\ell_{\text{cs}}^{(y)}(\boldsymbol{\alpha}) = \max_{z \in [k]} \{\mathbb{1}_{[z \neq y]} + \boldsymbol{\alpha}_z - \boldsymbol{\alpha}_y\}$

Weston and Watkins [36] $\ell_{\text{ww}}^{(y)}(\boldsymbol{\alpha}) = \sum_{z \neq y} \max\{0, 1 + \boldsymbol{\alpha}_z - \boldsymbol{\alpha}_y\}$

Lee, Lin and Wahba [24] $\ell^{(y)}_{\text{llw}}(\boldsymbol{\alpha}) = \sum_{z \neq y} \max\{0, 1 + \boldsymbol{\alpha}_z\}$ subject to $\sum_{z=1}^{k} \boldsymbol{\alpha}_z = 0$.

Logistic $\ell^{(y)}_{\text{logistic}}(\boldsymbol{\alpha}) = \log(\sum_{z=1}^{k} \exp(\boldsymbol{\alpha}_z)) - \boldsymbol{\alpha}_y$

Smooth max-of-hinge [33] $\ell^{(y)}_{\text{smh}}(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha} - \mathbf{e}_y\|^2 - \min_{\mathbf{u} \in \Delta} \|\boldsymbol{\alpha} - \mathbf{u}\|^2$, where $\mathbf{e}_y$ is the y'th canonical basis vector of $\mathbb{R}^k$ and Δ is the $(k-1)$-dimensional probability simplex.

Square $\ell^{(y)}_{\text{sqr}}(\boldsymbol{\alpha}) = (1 - \boldsymbol{\alpha}_y)^2 + \sum_{z \neq y} \boldsymbol{\alpha}_z^2$

Let us show proofs for a couple of these examples.

Claim 1. *The Crammer and Singer loss is not consistent because it is not classification calibrated.*

Proof. For the distribution $\mathbf{q} = (3/7, 2/7, 2/7)$ on three classes, a minimum of the expected conditional loss, $\min_{\boldsymbol{\alpha}} \sum_{y=1}^{k} \mathbf{q}_y \ell^{(y)}_{cs}(\boldsymbol{\alpha})$, is attained at $\boldsymbol{\alpha} = \mathbf{0}$. This solution has an expected conditional zero-one loss of 1 instead of the optimal 4/7.

□

Claim 2. *The logistic loss is classification calibrated and therefore consistent.*

Proof. Let $\mathbf{q}$ be any distribution over $[k]$. Assume WLOG that $\mathbf{q}_1$ is the largest and $\mathbf{q}_2$ is the largest such that $\mathbf{q}_1 - \mathbf{q}_2 > \epsilon$. First, the minimum expected conditional loss is

$$\min_{\boldsymbol{\alpha}} \sum_{y=1}^{k} \mathbf{q}_y \ell^{(y)}_{\text{logistic}}(\boldsymbol{\alpha}) = -\sum_{y=1}^{k} \mathbf{q}_y \log \mathbf{q}_y$$

Secondly,

$$\min_{W(\boldsymbol{\alpha}, \mathbf{q}) > \epsilon} \sum_{y=1}^{k} \mathbf{q}_y \ell^{(y)}_{\text{logistic}}(\boldsymbol{\alpha})$$

by our assumption is the same as

$$\min_{\boldsymbol{\alpha}: \boldsymbol{\alpha}_2 \geq \boldsymbol{\alpha}_1} \sum_{y=1}^{k} \mathbf{q}_y \ell^{(y)}_{\text{logistic}}(\boldsymbol{\alpha}) = -(\mathbf{q}_1 + \mathbf{q}_2) \log((\mathbf{q}_1 + \mathbf{q}_2)/2) - \sum_{y=3}^{k} \mathbf{q}_y \log \mathbf{q}_y$$

which is always larger than $-\sum_{y=1}^{k} \mathbf{q}_y \log \mathbf{q}_y$. Therefore the logistic loss is classification calibrated.

□

2.3 Discussion

The definition of consistency assumes that the function f is such that the surrogate loss ℓ can attain any value over any point in its domain. However, by the no-free-lunch theorem we know that we cannot learn unless we restrict f to be from a "small" hypothesis class. Later on, we will show that unless the surrogate is free to use any function, even for cases in which the Bayes-optimal predictor is linear, convergence to an optimal predictor is not guaranteed. More surprisingly, the latter is true even when it is possible to minimize the zero-one loss efficiently such as in the realizable case.

Chapter 3

The realizable case

This is a simple case where the labeling function is linear and deterministic. In this section we will show that under the realizable assumption it is possible to learn linear models efficiently. We will show that the minimizer of certain surrogate losses is bound to be the optimal predictor, however for other surrogate losses, albeit consistent, the minimizer is not guaranteed to be optimal.

The results in this chapter are as following. First, for the binary case.

Proposition 1. *Under the realizable assumption:*

1. *Any minimizer of the expected hinge loss is guaranteed to be an optimal predictor.*
2. *Minimizing a consistent surrogate loss function does not guarantee a small zero-one loss. In particular, there exists a distribution under which the minimizer of the square loss fails to be an optimal predictor.*

An analogous result for the multiclass case.

Proposition 2. *Under the realizable assumption:*

1. *Any minimizer of the expected Crammer and Singer loss is guaranteed to be an optimal predictor.*
2. *Minimizing a consistent surrogate loss function does not guarantee a small zero-one loss. In particular, there exists a distribution under which the minimizer of the multiclass square loss fails to be an optimal predictor.*

The results in these prepositions are well known in machine learning folklore and are made rigorous here. The reader should note that with regard to the second part of proposition 2, Long and Serevido ([25]) have shown an example in which the Lee, Lin and Wahba loss fails in the realizable case.

3.1 Binary

Assumption 2 (Realizable)**.** There exists a halfspace $\mathbf{w}^\star \in \mathbb{R}^d$ such that for all $\mathbf{x} \in \mathcal{X}$

$$\Pr[Y = 1|X = \mathbf{x}] = \mathbb{1}_{[\langle \mathbf{w}^\star, \mathbf{x}\rangle > 0]}$$

We will start by showing the hinge loss can always find an optimal predictor. We assume that there is a $\mathbf{w}^\star$ for which $y\langle \mathbf{w}^\star, \mathbf{x}\rangle > 0$ for all pairs $(\mathbf{x}, y)$. Up to scale, we can assume $y\langle \mathbf{w}^\star, \mathbf{x}\rangle \geq 1$ and therefore $\ell_{\text{hinge}}(\mathbf{w}^\star, (\mathbf{x}, y)) = 0$. Since the hinge loss is non-negative, $\mathbf{w}^\star$ is a minimizer. Any other $\bar{\mathbf{w}}$ for which $\ell_{\text{hinge}}(\bar{\mathbf{w}}, (\mathbf{x}, y)) = 0$ must satisfy $y\langle \bar{\mathbf{w}}, \mathbf{x}\rangle \geq 1$ for all $(\mathbf{x}, y)$. This proves the first part of proposition 1.

On the other hand, consider the square loss and the following distribution on $\mathbb{R}^2$. The distribution is concentrated on three points $\mathbf{x}_1 = (-1, 1)$, $\mathbf{x}_2 = (0.9, 1)$ and $\mathbf{x}_3 = (1, 1)$. $\mathbf{x}_1$ and $\mathbf{x}_2$ are labeled -1 and $\mathbf{x}_3$ is labeled 1. $\mathbf{x}_1$ and $\mathbf{x}_3$ appear with probability 0.45 and $\mathbf{x}_2$ with probability 0.1. A minimizer the expected hinge loss is $\bar{\mathbf{w}} = (20, -19)$. A minimizer of expected square loss is $\hat{\mathbf{w}} = (910/1081, -190/1081)$ with an expected zero-one loss of 0.1. This proves the second part of proposition 1.

3.2 Multiclass

Assumption 3 (Realizable)**.** There exists a matrix $W^\star \in \mathbb{R}^{k\times d}$ such that for all $\mathbf{x} \in \mathcal{X}$

$$\mathbb{E}[\mathbf{e}_Y|X = \mathbf{x}] = \arg\max_{\mathbf{e}_z : z\in[k]} \langle \mathbf{e}_z, W^\star \mathbf{x}\rangle$$

where $W^\star$ is such that the maximum on the right hand side is always unique.

For part of proposition 2, consider the Crammer and Singer loss. Similarly to the binary hinge loss it can be shown that a minimizer of the expected Crammer and Singer loss has zero-one loss of 0.

Now consider the multiclass square loss $\ell^{(y)}(\boldsymbol{\alpha}) = (\boldsymbol{\alpha}_y - 1)^2 + \sum_{z\neq y} \boldsymbol{\alpha}_z^2$. It can be validated that this loss is in fact consistent. Also consider the

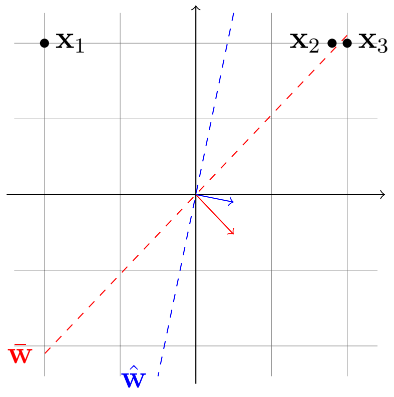


Figure 3.1: The halfspace returned by minimizing the expected hinge loss is depicted in red, while the one returned by minimizing the expected square in blue.

following distribution over $\mathbb{R}^2$. The distribution is uniform on three points $\mathbf{x}_1 = (0, 1)$, $\mathbf{x}_2 = (0.5, 0.5)$ and $\mathbf{x}_3 = (1, 0)$. Each $\mathbf{x}_i$ is labeled i.

A minimizer the expected Crammer and Singer loss is

$$\bar{W} = \begin{pmatrix} -7/3 & 5/3 \\ 2/3 & 2/3 \\ 5/3 & -7/3 \end{pmatrix}$$

while a minimizer of expected ℓ is

$$\hat{W} = \begin{pmatrix} -1/6 & 5/6 \\ 1/3 & 1/3 \\ 5/6 & -1/6 \end{pmatrix}$$

$\hat{W}$ classifies $\mathbf{x}_1$ and $\mathbf{x}_3$ correctly but misclassifies $\mathbf{x}_2$ as either a 1, a 2 or a 3, so the zero-one loss of $\hat{W}$ is $1/3$. This completes the second part of proposition 2.

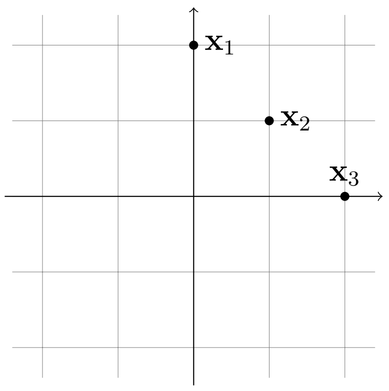


Figure 3.2: Distribution on which the multiclass square loss fails.

Chapter 4

Random noise

In this chapter we will focus on binary classification. We assume that the distribution over the data was a realizable distribution which has been corrupted with uniform random noise. In other words there is a halfspace $\mathbf{w}^\star$ such that for any $\mathbf{x} \in \mathcal{X}$

$$\Pr[Y = 1 | X = \mathbf{x}] = \begin{cases} 1 - \eta, & \langle \mathbf{w}^\star, \mathbf{x} \rangle > 0 \\ \eta, & \langle \mathbf{w}^\star, \mathbf{x} \rangle \leq 0 \end{cases}$$

for a parameter $\eta \in [0, 1/2)$.

The outline for this chapter is as following. We review Kearns' statistical query model and the methods of [8, 12] for learning under random noise in the statistical query model. We will then finish by showing a much simpler method based on surrogate loss minimization.

Throughout this chapter, we will assume that $\mathcal{X}$ is bounded, namely that $\|\mathbf{x}\| \leq 1$ for all $\mathbf{x} \in \mathcal{X}$. Our results are as following:

Proposition 3. *Under the random noise assumption, the optimal halfspace can be found in polynomial time. Moreover, by assuming margin as well a simple surrogate loss minimization procedure suffices to find the optimal halfspace.*

The fact that halfspaces are learnable with random noise is well known by works of [8, 12]. The surrogate loss for random noise is novel and is based on the article of [26].

4.1 Statistical queries

Let us start by defining the notion of a statistical query ([23]).

Definition 5 (Statistical Query)**.** A *Statistical Query* is a pair (χ, τ), where $\chi : \mathcal{X} \times \{-1, 1\} \to [0, 1]$ is a polynomial-time computable function and τ an accuracy parameter. Namely, a statistical query is a request for an estimator $\hat{P}_\chi$ of $P_\chi = \mathbb{E}[\chi(\mathbf{x}, y)]$ satisfying:

$$P_\chi - \tau \leq \hat{P}_\chi \leq P_\chi + \tau$$

with high probability.

Such a query can be implemented in polynomial time with confidence $1-\delta$ by simply estimating the expectation over $\text{poly}(1/\tau, \log(1/\delta))$ samples. Then clearly, any class efficiently learnable using statistical queries is efficiently learnable in the PAC model. However, the converse is not true. For example learning parity functions, which are efficiently learnable in the PAC model, can be shown to be hard using statistical queries.

4.1.1 Statistical queries in the presence of noise

Aslam and Decatur ([3]) had showed that in the random noise model, any statistical query with respect to the original distribution can be cast as a linear function of several estimates. Each such estimate can be evaluated as a statistical query over the noisy distribution.

Lemma 2. *For any noise rate $\eta < 1/2$, any statistical query (χ, τ) can be computed with confidence $1 - \delta$ in time and sample complexities of* $\text{poly}(1/\tau, \log(1/\delta), 1/(1 - 2\eta))$.

The manner of computing such statistical queries will be presented when we will present Cohen's algorithm. In the mean time, note that the above lemma entails that any algorithm which uses statistical queries can be automatically made robust to random noise.

4.2 Blum's algorithm

The algorithm of [8] improperly learns halfspaces under random noise. It returns a decision list of halfspaces. Each halfspace is capable of classifying an example as 1, -1 or "I don't know". Any example to be classified is checked against the first halfspace, if it returns 1 or -1 the example is classified as such. If it returns "I don't know" the example is checked against the second halfspace and so on. The process continues until one of the halfspaces returns a label or until the decision list is exhausted. In the latter case a random label is returned.

We shall begin by introducing the different subroutines which we will later combine to the full algorithm.

4.2.1 Outlier removal

Assume the data is given to a finite precision of b bits and that the it lies in the Euclidean unit ball. The following lemma states that points can be removed from the training set S, such that the resulting training S' is still fairly large and no point in S' is too far from any halfspace.

Lemma 3 (Outlier Removal)**.** *For any set $S \subseteq \mathbb{R}^d$ and any $\epsilon > 0$ there exists a subset $S' \subseteq S$ such that:*

- $|S'| \geq (1 - \epsilon - 2^{-db})|S|$
- *for every* $\mathbf{w} \in \mathbb{R}^d$*,* $\max_{\mathbf{x} \in S'} \langle \mathbf{w}, \mathbf{x} \rangle^2 \leq \beta \mathbb{E}_{\mathbf{x} \in S'}[\langle \mathbf{w}, \mathbf{x} \rangle^2]$

where $\beta = O(d^7 b/\epsilon)$. Moreover, such a set S' can be computed in polynomial time.

The expectations above are taken with respect to the empirical distribution.

The algorithm from the lemma computes the second-moment matrix $A = \mathbb{E}_{\mathbf{x} \in S}[\mathbf{x}\mathbf{x}^\top]$ and transforms the data by applying $A^{-1/2}\mathbf{x}$ (assuming A is full rank). In the new space, we have for all $\|\mathbf{w}\| = 1$, $\mathbb{E}_{\mathbf{x} \in S}[\langle \mathbf{w}, \mathbf{x} \rangle^2] = 1$. The algorithm removes all points for which $\|\mathbf{x}\|$ is large. Then the process is repeated until the conditions of the theorem are satisfied.

A key point is that after the algorithm is done, a large fraction of the examples in S' have margin in the transformed space. To see this, note that for all $\mathbf{w}$ of unit norm, $\mathbb{E}_{\mathbf{x} \in S'}[\langle \mathbf{w}, \mathbf{x} \rangle^2] = 1$ and that implies that $\max_{\mathbf{x} \in S'} \|\mathbf{x}\|^2 \leq \beta d$. Thus for any $\mathbf{w}$ (of unit norm),

$$\mathbb{E}_{\mathbf{x} \in S}[\cos(\mathbf{w}, \mathbf{x})^2] = \mathbb{E}_{\mathbf{x} \in S'}[\frac{\langle \mathbf{w}, \mathbf{x} \rangle^2}{\|\mathbf{x}\|^2}] \geq \frac{\mathbb{E}_{\mathbf{x} \in S'}[\langle \mathbf{w}, \mathbf{x} \rangle^2]}{\max_{\mathbf{x} \in S'} \|\mathbf{x}\|^2} \geq \frac{1}{\beta d}$$

This means that at least a $1/(2\beta d)$-fraction of points in S' satisfy $\cos(\mathbf{w},\mathbf{x})^2 \geq 1/(2\beta d)$.

4.2.2 Modified Perceptron

We now proceed to present the next building block of Blum's algorithm, the *modified Perceptron algorithm*. It defers from the original Perceptron (8) in two ways. First, the algorithm returns a vector $\mathbf{w}$ such that all misclassified samples $\mathbf{x} \in S'$ satisfy $|\cos(\mathbf{w},\mathbf{x})| \leq \sigma$ with high probability, for a parameter σ. Second, the algorithm uses statistical queries to estimate the update step. Note that statistical queries can be used to estimate conditional expectations by expanding the expectation to a quotient of two other expectations.

With the output of the outlier removal algorithm (3), by setting $\sigma = \sqrt{1/(2\beta d)}$ we can guarantee that a $1/(2\beta d)$-fraction of the points would be classified correctly. The rest of the points would be classified as "I don't know".

Algorithm 1 Modified Perceptron

```
1) Input: $S$, $\sigma$
2) Denote $\tau = \epsilon/(6\beta d)$
3) Sample $\mathbf{w}$ at random uniformly from the unit sphere
4) Perform a statistical query
       $$\left((\mathbf{x},y) \mapsto \mathbb{1}_{[\mathrm{sign}(\langle \mathbf{w}^\star,\mathbf{x}\rangle)\cos(\mathbf{w},\mathbf{x}) \leq -\sigma]}, \tau\right)$$
   and halt if the result is at most $2\tau$.
5) Perform statistical queries to estimate
       $$\mathbb{E}_{(\mathbf{x},y)\in S}[\mathrm{sign}(\langle \mathbf{w}^\star,\mathbf{x}\rangle)\mathbf{x} | \mathrm{sign}(\langle \mathbf{w}^\star,\mathbf{x}\rangle)\cos(\mathbf{w},\mathbf{x}) \leq -\sigma]$$
   with confidence $\sigma^3/(16\sqrt{d}\log(d))$. Denote $\mathbf{x}_{\text{update}}$ as the result and execute:
       $$\mathbf{w} \leftarrow \mathbf{w} - \frac{\langle \mathbf{w}, \mathbf{x}_{\text{update}}\rangle}{\|\mathbf{x}_{\text{update}}\|^2}\mathbf{x}_{\text{update}}$$
6) If performed more than $(1/\sigma^2)\log(d)$ updates, go back to step (3).
```

In the original paper, the authors show a direct approach in estimating the quantities in steps (4) and (5) of algorithm 1. It is similar to the approach of Cohen's algorithm which we will review in the next section.

Algorithm 2 Blum's Algorithm

```
Input: $S$, accuracy $\epsilon$
Denote $\sigma = \sqrt{1/(2\beta d)}$
Create an empty decision list
**repeat**
    Apply the outlier removal lemma on $S$.
    Run the modified Perceptron on the transformed $S'$ with parameter $\sigma$
    Remove all points classified correctly by $\mathbf{w}$ from $S'$, add $\mathbf{w}$ to the decision list.
**until** reached the desired accuracy.
```

4.3 Cohen's algorithm

Unlike algorithm 2, the following algorithm properly learns halfspaces with random noise. Meaning the hypothesis returned by the algorithm is itself a halfspace.

The algorithm employs the ellipsoid algorithm as a black box in order to find an ϵ-correct hypothesis with high probability. At each iteration, the algorithm implements the separation oracle as finding a *witness* for the current hypothesis. Denote the current hypothesis as $\mathbf{w}$, a witness $\mathbf{v}$ is any vector that satisfies two inequalities with high probability:

1. $\mathbf{w}^\star$ is correct on $\mathbf{v}$: $\langle \mathbf{w}^\star, \mathbf{v} \rangle > 0$
2. $\mathbf{w}$ mistakes on $\mathbf{v}$: $\langle \mathbf{w}, \mathbf{v} \rangle < 0$

In order to explain how a witness is found, first define the following quantities:

$$\begin{aligned}
\tilde{\mathbf{m}}^- &= \mathbb{E}[\mathbb{1}_{[y\langle \mathbf{w},\mathbf{x}\rangle \le 0]} y\mathbf{x}] \\
\tilde{\mathbf{m}}^+ &= \mathbb{E}[\mathbb{1}_{[y\langle \mathbf{w},\mathbf{x}\rangle \ge 0]} y\mathbf{x}] \\
\mathbf{m}^- &= \mathbb{E}[\mathbb{1}_{[\text{sign}(\langle \mathbf{w}^\star,\mathbf{x}\rangle)\langle \mathbf{w},\mathbf{x}\rangle \le 0]} \text{sign}(\langle \mathbf{w}^\star, \mathbf{x}\rangle)\mathbf{x}] \\
\mathbf{m}^+ &= \mathbb{E}[\mathbb{1}_{[\text{sign}(\langle \mathbf{w}^\star,\mathbf{x}\rangle)\langle \mathbf{w},\mathbf{x}\rangle \ge 0]} \text{sign}(\langle \mathbf{w}^\star, \mathbf{x}\rangle)\mathbf{x}]
\end{aligned}$$

The quantity $\mathbf{m}^-$ is the one we would like to estimate. It is the one we would use if we were running the Perceptron algorithm with respect to the original distribution (before it has been corrupted with random noise). We cannot estimate $\mathbf{m}^-$ directly (since we do not know $\mathbf{w}^\star$) but instead we can use estimates of $\tilde{\mathbf{m}}^-$ and $\tilde{\mathbf{m}}^+$ since

$$\mathbf{m}^- = \frac{(1-\eta)\tilde{\mathbf{m}}^- + \eta\tilde{\mathbf{m}}^+}{1-2\eta}$$

Unfortunately, $\tilde{\mathbf{m}}^-$ and $\tilde{\mathbf{m}}^+$ cannot be estimated with high probability as the estimates depend on $\Pr[y\langle\mathbf{w},\mathbf{x}\rangle \leq 0]$ and $\Pr[y\langle\mathbf{w},\mathbf{x}\rangle \geq 0]$ and these can be arbitrarily small. Let us introduce a new small positive constant ν and define the following:

$$\begin{aligned}\mathbf{m}^{-,\nu} &= \frac{(1-\eta-\nu)\tilde{\mathbf{m}}^- + (\eta+\nu)\tilde{\mathbf{m}}^+}{1-2\eta} \\ &= (1-\frac{\nu}{1-2\eta})\mathbf{m}^- + \frac{\nu}{1-2\eta}\mathbf{m}^+\end{aligned}$$

Any accurate estimate of $\mathbf{m}^{-,\nu}$ is guaranteed to fulfill the first witness property, however not the second.

That brings us to this next idea. Note that $\mathbf{w}^\star$ is still the optimal hypothesis even if we restrict the distribution to a subset of the original domain. Therefore, we can restrict the distribution to a slice of the form $\{\mathbf{x} \in \mathcal{X} : a \leq |\langle\mathbf{w},\mathbf{x}\rangle| \leq a(1+\mu)\}$, which guarantees that on the restricted distribution $\mathbf{m}^{-,\nu}$ satisfies the second witness property.

Such a slice could be identified in the following manner. Sample enough points from $\mathcal{D}$ and throw away all slices from which not enough points have been sampled. Then, return a slice with a maximum proportion of mistakes made by $\mathbf{w}$ on the samples from that slice. One could show that on such a slice can be identified with high probability, and that $\mathbf{m}^{-,\nu}$ satisfies the second witness property.

Finally, throughout the execution of the algorithm, the lemma 3 (outlier removal) is used each time the distribution is restricted. This is done so that $\mathbf{m}^{-,\nu}$ could be estimated by a number of samples which is independent of the margin.

4.4 Surrogate loss

The previous couple of algorithms are quite complicated. The main reason is that their sample and runtime complexities do not depend on the margin.

In this section we will show that if one is willing to assume margin then a far simpler algorithm suffices to solve the problem. We will follow an approach similar to [26].

Consider the loss function:

$$\ell_{\text{rn}}(\alpha) = \frac{1}{1-2\eta} \max\{(1-\eta)(1-\alpha), -\eta(1+\alpha)\}$$

This loss function is an unbiased estimator for a hinge-like function with respect to the original distribution (without noise):

$$(1-\eta)\ell_{\text{rn}}(\alpha) + \eta\ell_{\text{rn}}(-\alpha) = \begin{cases} 0, & \alpha \geq \frac{1}{1-2\eta} \\ (1-\eta)(\frac{1}{1-2\eta} - \alpha), & -\frac{1}{1-2\eta} < \alpha < \frac{1}{1-2\eta} \\ 1-\alpha, & \text{otherwise} \end{cases}$$

and for any $\mathbf{x}$ and any $\mathbf{w}$ the expected conditional loss is

$$(1-\eta)\ell_{\text{rn}}(\text{sign}(\langle \mathbf{w}^\star, \mathbf{x}\rangle)\langle \mathbf{w}, \mathbf{x}\rangle) + \eta\ell_{\text{rn}}(-\text{sign}(\langle \mathbf{w}^\star, \mathbf{x}\rangle)\langle \mathbf{w}, \mathbf{x}\rangle)$$

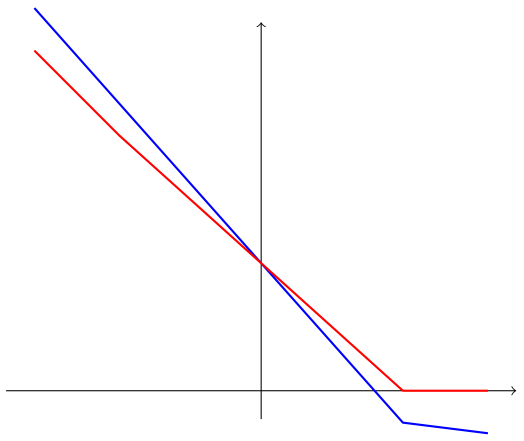


Figure 4.1: In blue, the loss ℓ_{rn} for $\eta = 0.1$. Its conditional expectation is depicted in red.

If we assume margin, that for any $\mathbf{x}$, $|\langle \mathbf{w}^\star, \mathbf{x}\rangle| \geq 1/(1-2\eta)$, then $\mathbf{w}^\star$ is the minimizer of the expected loss and any other minimizer has the same predictions as $\mathbf{w}^\star$.

By standard bounds from the general setting of learning, a minimizer $\hat{\mathbf{w}}$ of the empirical ℓ_{rn} with m samples has

$$\mathbb{E}[\ell_{\mathrm{rn}}(\hat{\mathbf{w}})] \leq \mathbb{E}[\ell_{\mathrm{rn}}(\mathbf{w}^\star)] + O\left(\sqrt{\frac{1}{m}}\right)$$

with high probability. By our construction $\mathbb{E}[\ell_{\mathrm{rn}}(\hat{\mathbf{w}})] \geq \Pr[\mathrm{sign}(\langle \hat{\mathbf{w}}, \mathbf{x}\rangle) \neq \mathrm{sign}(\langle \mathbf{w}^\star, \mathbf{x}\rangle)]$ and $\mathbb{E}[\ell_{\mathrm{rn}}(\mathbf{w}^\star)] = 0$, so

$$\Pr[\mathrm{sign}(\langle \hat{\mathbf{w}}, \mathbf{x}\rangle) \neq \mathrm{sign}(\langle \mathbf{w}^\star, \mathbf{x}\rangle)] \leq O\left(\sqrt{\frac{1}{m}}\right)$$

with high probability. This completes the proof of proposition 3.

Chapter 5

Generalized linear models

In this chapter we will present the *generalized linear model* assumption which is a generalization of the realizable and random noise assumptions. This will be both for binary and multiclass settings. In essence, this assumption entails that the distribution of the label given any example is a function of the optimal predictor. This function would have certain properties which would enable us to learn efficiently. Moreover, unlike the previous chapters, here we are interested in estimating these conditional distributions, namely it becomes a problem of regression rather than classification.

We have the following results in this chapter.

Proposition 4. *A generalized linear model assumption entails the existence of a convex surrogate loss functions, such that its minimizer is an optimal predictor.*

The proof of this proposition is based on the article of [1].

5.1 Binary

Throughout this section we will define the label space to be $\mathcal{Y} = \{0, 1\}$ instead of $\{-1, 1\}$ as it was defined thus far.

Definition 6 (Generalized linear model)**.** A *generalized linear model* (GLM) is a generative assumption which entails there exists a monotone nondecreasing function $g : \mathbb{R} \to [0, 1]$ and a halfspace represented by $\mathbf{w}^\star$ such that for all $\mathbf{x}$,

$$\Pr[Y = 1 | X = \mathbf{x}] = g(\langle \mathbf{w}^\star, \mathbf{x} \rangle)$$

The function g is called a *link function*.

5.1.1 Examples

Before moving on to show how we could learn such models let us show some examples.

Realizable The realizable assumption is a special case of GLMs with the link function $g(\alpha) = \mathbb{1}_{[\alpha>0]}$.

Random noise This is also a GLM with

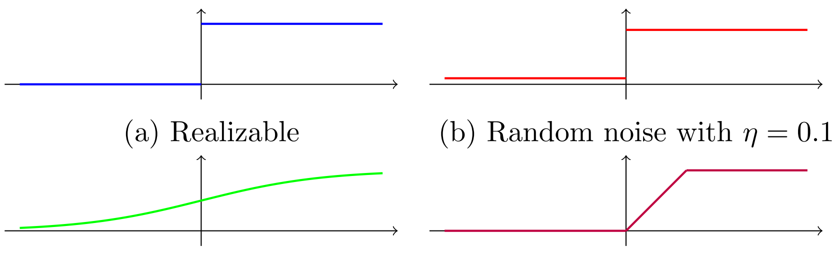
$$g(\alpha) = \begin{cases} 1-\eta, & \alpha > 0 \\ \eta, & \text{otherwise} \end{cases}$$

Logistic The logistic regression model is a GLM with

$$g(\alpha) = \frac{1}{1+\exp(-\alpha)}$$

Ramp This GLM is given by the link function

$$g(\alpha) = \begin{cases} 0, & \alpha \leq 0 \\ \alpha, & 0 < \alpha < 1 \\ 1, & \alpha \geq 1 \end{cases}$$


(a) Realizable (b) Random noise with $\eta = 0.1$

(c) Logistic (d) Ramp

Figure 5.1: Examples

These examples are in part there to show that the GLM assumption is strictly weaker than the realizable and random noise assumptions.

5.1.2 Learning generalized linear models

Our goal is to find a predict the probability that $Y = 1$ with good accuracy. Meaning we want to find a $\mathbf{w}$ that minimizes $\mathbb{E}[(g(\langle \mathbf{w}, \mathbf{x}\rangle) - g(\langle \mathbf{w}^\star, \mathbf{x}\rangle))^2]$. As one can see for the realizable and random noise models this is similar to minimizing the zero-one loss, but in general this is a regression problem rather than a classification problem.

We will focus on the case where g is continuous and L-Lipschitz. These additional assumptions entail that there exists a convex differentiable function $\Phi : \mathbb{R} \to \mathbb{R}$ such that $g \equiv \Phi'$. Now we can define the following surrogate loss:

$$\ell(\mathbf{w}; (\mathbf{x}, y)) = \Phi(\langle \mathbf{w}, \mathbf{x}\rangle) - y\langle \mathbf{w}, \mathbf{x}\rangle$$

Let us see a couple of examples of such loss functions.

Logistic The loss given for the logistic link function is the logistic loss:

$$\ell_{\text{logistic}}(\mathbf{w}; (\mathbf{x}, y)) = \log(1 + \exp(\langle \mathbf{w}, \mathbf{x}\rangle)) - y\langle \mathbf{w}, \mathbf{x}\rangle$$

Ramp The loss given for the ramp link function is the one-sided Huber loss (up to constants):

$$\ell_{\text{osh}}(\mathbf{w}; (\mathbf{x}, y)) = \begin{cases} y(1/2 - \langle \mathbf{w}, \mathbf{x}\rangle), & \langle \mathbf{w}, \mathbf{x}\rangle \le 0 \\ 1/2(\langle \mathbf{w}, \mathbf{x}\rangle - y)^2, & 0 < \langle \mathbf{w}, \mathbf{x}\rangle < 1 \\ (1-y)(\langle \mathbf{w}, \mathbf{x}\rangle - 1/2), & \text{otherwise} \end{cases}$$

The following lemma shows that a minimizer of the expected surrogate loss is also a Bayes-optimal predictor.

Lemma 4 ([1])**.** *Suppose $g : \mathbb{R} \to [0, 1]$ is a monotone nondecreasing and continuous function and that the GLM assumption holds, then $\mathbf{w}^\star$ is a minimizer of $\mathbb{E}[\ell(\mathbf{w}; (\mathbf{x}, y))]$. Furthermore, if $\bar{\mathbf{w}}$ is a (different) minimizer of $\mathbb{E}[\ell(\mathbf{w}; (\mathbf{x}, y))]$, then $\Pr[Y = 1|X = \mathbf{x}] = g(\langle \bar{\mathbf{w}}, \mathbf{x}\rangle)$.*

Finally, since we assumed g is L-Lipschitz then Φ is L-strongly smooth and therefore ℓ is L-strongly smooth with respect to $\langle \mathbf{w}, \mathbf{x}\rangle$. Because $\mathbf{w}^\star$ is a bayes-optimal predictor then for all $\mathbf{x}$, $\langle \mathbf{w}^\star, \mathbf{x}\rangle$ is a pointwise minimizer of the conditional expected loss. That means that for any other $\mathbf{w}$ (Theorem 2.1.5 in [27]),

$$\mathbb{E}[\ell(\mathbf{w}; (X, Y)) - \ell(\mathbf{w}^\star; (X, Y))|X = \mathbf{x}] \ge \frac{1}{2L}(g(\langle \mathbf{w}, \mathbf{x}\rangle) - g(\langle \mathbf{w}^\star, \mathbf{x}\rangle))^2$$

and taking the expectation over X,

$$\mathbb{E}[\ell(\mathbf{w};(X,Y))] - \ell(\mathbf{w}^\star;(X,Y))] \geq \frac{1}{2L}\mathbb{E}[(g(\langle \mathbf{w}, X\rangle) - g(\langle \mathbf{w}^\star, X\rangle))^2]$$

Once again by bounds from the general setting of learning, given enough samples we can ensure that for a $\mathbf{w}$ which minimizes the empirical loss, the quantity on the right hand side is small with high probability.

5.2 Multiclass

The definition of multiclass GLMs is an extension of binary GLMs.

Definition 7 (Generalized linear model)**.** A *generalized linear model* (GLM) is a generative assumption which entails there exists a convex differentiable function $\Phi : \mathbb{R}^k \to \mathbb{R}$ and a matrix $W^\star \in \mathbb{R}^{k\times d}$ such that for all $\mathbf{x}$,

$$\mathbb{E}[\mathbf{e}_Y | X = \mathbf{x}] = g(W^\star \mathbf{x})$$

where $g \equiv \nabla\Phi$.

The link function now maps $W^\star\mathbf{x}$ to a k-dimensional distribution vector, in which the i'th entry is the probability of $Y = i$ given $\mathbf{x}$.

To learn multiclass GLMs, as in the binary case we can define the following consistent loss function:

$$\ell(W;(\mathbf{x},y)) = \Phi(W\mathbf{x}) - (W\mathbf{x})_y$$

and once again if we assume that g is L-Lipschitz that entails the Φ is L-smooth. That means that for any matrix W,

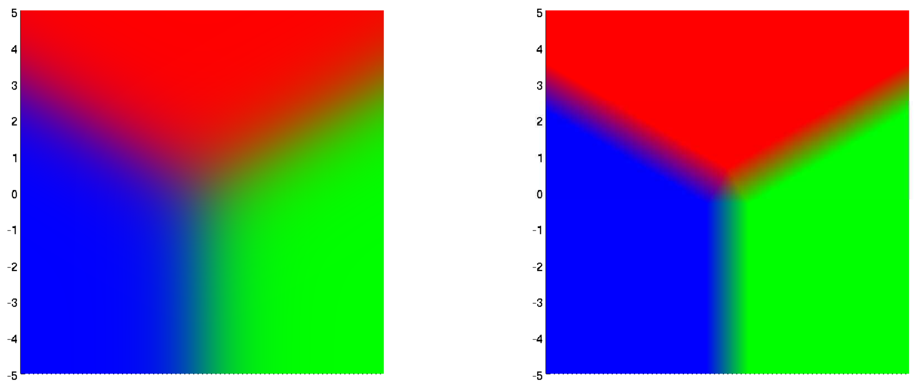
$$\mathbb{E}[\ell(W;(X,Y))] - \ell(W^\star;(X,Y))] \geq \frac{1}{2L}\mathbb{E}[\|g(WX) - g(W^\star X)\|^2]$$

5.2.1 Examples

Here are a couple of examples of multiclass GLMs.

Logistic This model corresponds to multinomial logistic regression.

$$g(\boldsymbol{\alpha})_y = \frac{\exp(\boldsymbol{\alpha}_y)}{\sum_{z=1}^k \exp(\boldsymbol{\alpha}_z)}$$



(a) Logistic (b) Projection

Figure 5.2: Multiclass link functions depicted for three class in $\mathbb{R}^2$. The probability of each class is assigned a color: red, green and blue - the higher the probability of the class the stronger its color component.

Projection This GLM is given by $g(\boldsymbol{\alpha}) = \arg\min_{\mathbf{v}\in\Delta} \|\mathbf{v} - \boldsymbol{\alpha}\|$, where Δ is the $(k-1)$-dimensional probability simplex.

For the above link functions we have the following loss functions:

Logistic $\ell(W;(\mathbf{x},y)) = \log(\sum_{z=1}^{k} \exp(\langle \mathbf{e}_z, W\mathbf{x}\rangle)) - (W\mathbf{x})_y$

Projection $\ell(W;(\mathbf{x},y)) = \min_{\mathbf{v}\in\Delta} \|W\mathbf{x} - \mathbf{v}\|^2 - \|W\mathbf{x} - \mathbf{e}_y\|^2$ (up to constants)

Chapter 6

Learning halfspaces with margin

In this section, we focus on binary classification. We will first show an example from [6] proving that without any assumptions on the distribution, minimizing a convex and consistent surrogate loss can result in an arbitrarily bad predictor.

We will continue by adding an assumption of margin and state results from [6, 16] showing that any learner based on minimizing a surrogate convex loss can approximate the optimal zero-one loss by a factor of $\Omega(1/\gamma)$.

Proposition 5. *We have the following results in this chapter:*

1. *Without margin, there exists a distribution under which minimizing any convex and consistent surrogate loss would result in an arbitrarily bad predictor.*
2. *Convex surrogate loss functions give an $\Omega(1/\gamma)$ approximation factor.*
3. *There is an algorithm which returns the optimal halfspace with time and sample complexities of $O(\exp(1/\gamma^2))$.*

The example of the first part of the proposition is due to [6]. The proofs in the second part are from [6, 16]. Finally, the algorithm of the third part is due to [32].

6.1 Failure in the general case

In this section we will show a distribution over which a minimizer of any convex consistent surrogate loss will return a predictor with high zero-one loss.

Let $\nu < 1/2$. Consider a distribution over $\mathbb{R}$ concentrated on four points:

- -1 labeled 1 with probability $\nu/2$.
- $-\beta$ labeled -1 with probability $(1-\nu)/2$.
- β labeled 1 with probability $(1-\nu)/2$.
- 1 labeled -1 with probability $\nu/2$.

An optimal halfspace would be one which classifies the right of zero as 1, and suffer a zero-one loss of ν. However for any minimizer of a consistent and convex surrogate loss there's a β small enough such that it would classify the right of zero as -1, because the points close to zero suffer a small loss compared to the ones far from zero. Such a halfspace would have a zero-one loss of $1-\nu$. By reducing ν and β appropriately, we can make sure that the zero-one loss of halfspace minimizing the surrogate loss is arbitrarily close to 1.

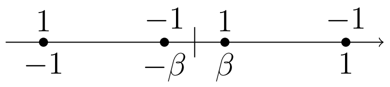


Figure 6.1: Any minimizer of a consistent and convex surrogate loss would classify the right of zero as -1.

6.2 Approximation under margin

For this section we will assume that a margin exists (1). We assume there is a predictor $\mathbf{w}^\star$ with a margin loss of ν and margin γ. We have the following trivial claim.

Claim 3. *For any minimizer* $\mathbf{w}$ *of the expected hinge loss:*

$$\mathbb{E}[\ell_{\text{hinge}}(y\langle \mathbf{w}, \mathbf{x}\rangle)] \leq \left(1 + \frac{1}{\gamma}\right)\nu$$

This next theorem states that the approximation ratio of the hinge loss is optimal up to a factor of 2, compared to all other convex and consistent surrogate loss functions.

Theorem 4. *Let ℓ be any convex and consistent surrogate loss. For any $\gamma > 0$, there exists a distribution such that for any minimizer $\mathbf{w}$ of the expected loss:*

$$\mathbb{E}[\ell(y\langle \mathbf{w}, \mathbf{x}\rangle)] \geq \frac{1}{2}\min\left\{\left(1+\frac{1}{\gamma}\right)\nu, 1\right\}$$

6.3 Learning kernel-based halfspaces

In this section we will present the work of [32]. This is an algorithm for improperly learning halfspaces in the agnostic setting under the assumption of margin. Unlike the previous section, this algorithm returns an exact solution but requires sample and runtime complexities which are exponential in the margin.

Define the label space to be $\mathcal{Y} = \{0, 1\}$. Then we can define the error of an hypothesis $h : \mathcal{X} \to [0, 1]$ as:

$$\text{err}(h) = \mathbb{E}[|h(\mathbf{x}) - y|]$$

Suppose h is of the form $\mathbf{x} \mapsto \phi_{0-1}(\langle \mathbf{w}, \mathbf{x}\rangle)$ where $\phi_{0-1}(\alpha) = \mathbb{1}_{[\alpha>0]}$, then $\text{err}(h)$ is exactly the zero-one loss. The function ϕ_{0-1} is called a *transfer function.*

The first idea behind the algorithm is to approximate ϕ_{0-1} by an L-Lipschitz transfer function:

$$\phi_{\text{sig}}(\alpha) = \frac{1}{1+\exp(-4L\alpha)}$$

for which we can define a hypothesis class $\mathcal{H}_{\text{sig}} = \{\mathbf{x} \mapsto \phi_{\text{sig}}(\langle \mathbf{w}, \mathbf{x}\rangle) : \mathbf{w} \in \mathcal{X}\}$. Unfortunately it is NP-hard to learn this hypothesis class.

This is where this next idea comes into play. We will improperly learn $\mathcal{H}_{\text{sig}}$ by a larger hypothesis class $\cup_p \mathcal{H}_p$ for $\mathcal{H}_p = \{\mathbf{x} \mapsto p(\langle \mathbf{w}, \mathbf{x}\rangle) : \mathbf{w} \in \mathcal{X}\}$. The transfer functions p are polynomials chosen in such a way as to approximate ϕ_{sig}.

In order to learn with respect to these polynomials we can use the following kernel mapping:

$$k(\mathbf{x}, \mathbf{x}') = \frac{1}{1 - \langle \mathbf{x}, \mathbf{x}'\rangle/2}$$

It implements an inner product of an RKHS $\mathbb{V}$ given by a feature mapping ψ, which maps each $\mathbf{x}$ to a vector of its monomials (up to constants). We can also define an appropriate hypothesis class:

$$\mathcal{H}_B = \{\mathbf{x} \mapsto \langle \mathbf{v}, \psi(\mathbf{x}) \rangle : \mathbf{v} \in \mathbb{V},\ \|\mathbf{v}\|^2 \leq B\}$$

The algorithm involves sampling m examples and returning the hypothesis h which minimizes

$$\min_{h \in \mathcal{H}_B} \frac{1}{m} \sum_{i=1}^{m} |h(\mathbf{x}_i) - y_i|$$

The hypothesis h is called an ERM predictor. Using the Representer theorem ([30]), we know h is of the form $h(\cdot) = \sum_{i=1}^{m} \alpha_i k(\mathbf{x}_i, \cdot)$. Then to find h we actually need to minimize

$$\min_{\alpha_1, \ldots, \alpha_m} \frac{1}{m} \sum_{i=1}^{m} |\sum_{j=1}^{m} \alpha_j k(\mathbf{x}_j, \mathbf{x}_i) - y_i| \ s.t. \ \sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \leq B$$

Let us finish this section by presenting a theoretical guarantee for the algorithm.

Theorem 5. *Let $\epsilon, \delta \in (0,1)$. For any $L \geq 3$, let*

$$B = 6L^4 + \exp(9L \log\left(\frac{2L}{\epsilon}\right) + 5)$$

and let m be a sample size satisfying

$$m \geq \frac{8B}{\epsilon^2} \left(2 + 9\sqrt{\log\left(\frac{8}{\delta}\right)}\right)^2$$

Then for any distribution $\mathcal{D}$, with probability at least $1-\delta$, any ERM predictor $\hat{h} \in \mathcal{H}_B$ with respect to $\mathcal{H}_B$ satisfies

$$\mathrm{err}(\hat{h}) \leq \min_{h \in \mathcal{H}_{\mathrm{sig}}} \mathrm{err}(h) + \epsilon$$

6.3.1 Lower bound

One could think if maybe a different algorithm could do better and this is an open question. However for kernel-based learners, as the one presented above, we will show that this result is tight. The results presented here are due to [16].

This next theorem is an extension of theorem 4. It includes the case where one is allowed to use kernels. It states that if the sample complexity is at most polynomial in the inverse margin, then the best approximation factor achievable is at least an order of $1/\gamma$.

Theorem 6. *Let ℓ be an convex surrogate loss. Let A be an kernel-based learning algorithm w.r.t ℓ with sample complexity at most* $\text{poly}(1/\gamma)$*. Then for every $\gamma > 0$ there exists a distribution such that w.p. at least* $1 - 10\exp(-1/\gamma)$ *the algorithm A returns a hypothesis h for which*

$$\mathbb{E}[\ell_{0-1}(h)] \geq \nu \cdot \Omega\left(\frac{1}{\gamma \cdot \text{poly}(\log(1/\gamma))}\right)$$

The next theorem states that if a kernel-based algorithm is able get a better approximation, then its sample complexity is exponential in $1/\gamma$.

Theorem 7. *Let ℓ be an convex surrogate loss, let $\epsilon > 0$ and let A be an kernel-based learning algorithm w.r.t ℓ, such that for every $\gamma > 0$ there exists a distribution for which w.p. at least* $1/2$ *the algorithm A returns a hypothesis h such that*

$$\mathbb{E}[\ell_{0-1}(h)] \leq \nu \cdot \left(\frac{1}{\gamma}\right)^{1-\epsilon}$$

then the sample complexity of A is $\Omega(\exp((1/\gamma)^{a(\epsilon)})$ *for some* $a(\epsilon) > 0$.

Chapter 7

Square Loss

Consider the binary case. In this section we will show that the square loss returns the optimal classifier for certain class of distributions. The reader is reminded that the square loss is defined as

$$\ell_{\text{square}}(\alpha) = (\alpha - 1)^2$$

Suppose the expectation $\boldsymbol{\mu} = \mathbb{E}[y\mathbf{x}]$ and the covariance $\Sigma = \mathbb{E}[\mathbf{x}\mathbf{x}^\top] - \boldsymbol{\mu}\boldsymbol{\mu}^\top$ are well defined. For simplicity, let us further assume that Σ is invertible but note that the results for when it is not invertible are similar. We will show the following result:

Proposition 6. *Let $\hat{\mathbf{w}}$ be the halfspace returned by minimizing the expected square loss, then*

$$\sup_{\mathcal{D}} \Pr_{(\mathbf{x},y)\sim\mathcal{D}}[y\langle\hat{\mathbf{w}}, \mathbf{x}\rangle \leq 0] = \frac{1}{1 + \langle\mu, \Sigma^{-1}\mu\rangle}$$

where the supremum is taken with respect to all distributions with mean $\boldsymbol{\mu}$ and covariance Σ.

Once again, this is yet another result which is well known in machine learning folklore. As far as we know, this is the first time it has been proven rigorously.

For any halfspace $\mathbf{w} \in \mathbb{R}^d$, the random variable $y\langle\mathbf{w}, \mathbf{x}\rangle$ has an expectation of $\langle\mathbf{w}, \boldsymbol{\mu}\rangle$ and variance $\mathbf{w}^\top\Sigma\mathbf{w}$. By Cantelli's inequality ([10]):

$$\Pr[y\langle\mathbf{w}, \mathbf{x}\rangle \leq 0] \leq \tfrac{1}{1+\langle\mathbf{w},\boldsymbol{\mu}\rangle^2/\langle\mathbf{w},\Sigma\mathbf{w}\rangle}$$

One can check that the $\mathbf{w}$ that minimizes the right hand side is proportional to $\Sigma^{-1}\mu$. Since Cantelli's inequality is tight, such a $\mathbf{w}$ would be the optimal classifier if μ and Σ are sufficient statistics for the distribution of $y\mathbf{x}$, for example when $y\mathbf{x}$ is distributed normally. With that being said, for arbitrary distributions the above bound can be loose.

Claim 4. *The halfspace returned by minimizing the square loss is proportional to $\Sigma^{-1}\mu$.*

Proof. We would like to find a $\mathbf{w}$ for which the gradient of the expected square loss is zero,

$$\mathbb{E}[\mathbf{x}\mathbf{x}^\top \mathbf{w} - y\mathbf{x}] = 0$$

Plugging in the definition of $\boldsymbol{\mu}$ and Σ and re-arranging,

$$\mathbf{w} = (\Sigma + \boldsymbol{\mu}\boldsymbol{\mu}^\top)^{-1}\boldsymbol{\mu}$$

By the Sherman-Morrison formula ([34]) we have

$$(\Sigma + \boldsymbol{\mu}\boldsymbol{\mu}^\top)^{-1} = \Sigma^{-1} - \frac{1}{1 + \langle \boldsymbol{\mu}, \Sigma^{-1}\boldsymbol{\mu}\rangle}\Sigma^{-1}\boldsymbol{\mu}\boldsymbol{\mu}^\top\Sigma^{-1}$$

plugging that back in

$$\mathbf{w} = \frac{1}{1 + \langle \boldsymbol{\mu}, \Sigma^{-1}\boldsymbol{\mu}\rangle}\Sigma^{-1}\boldsymbol{\mu}$$

which is proportional to $\Sigma^{-1}\boldsymbol{\mu}$ as required. □

The proof of proposition 6 is complete.

Part II

Online Methods

In the previous part the learning procedure were of the form:

- Sample from the distribution.
- Minimize an expected convex loss, with the expectation taken over the empirical distribution generated by the sample.

The methods that we will present in this part are online, namely an adversary generates the examples, rather than being samples IID from some distribution, but the adversary is confined to label these examples in a certain manner (to be defined later). We will not discuss online versions of algorithms mentioned in the previous part.

The reader should note that online algorithms are applicable to the batch setting as well. This is done by online-to-batch conversion. Generally if the algorithm is run on a certain sample then one of its iterates has that its relative error is comparable to the average regret. This iterate can be found with high probability by returning the one which minimizes its error on a validation set.

As in the previous part, the methods that we shall discuss will gradually allow for weaker assumptions. Starting from the Perceptron algorithm, that finds an optimal halfspace in the realizable setting, and ending with the Isotron algorithm, which is an extension of generalized linear models.

Finally, we will show how all of these methods can be seen as instances of online convex optimization. As such, each such method could be converted to an equivalent offline method.

Chapter 8

Perceptron

For this chapter we assume that the distribution is separable with margin (2, 1). We will present the binary version first and then its multiclass extension. Lastly, we will relax the separability assumption.

Proposition 7. *We show the following:*

1. *Under the realizable assumption, the Perceptron algorithm has a mistake bound of* $1/\gamma^2$.
2. *Without the realizable assumption, the Perceptron's mistake bound depends on the performance of the best predictor w.r.t the appropriate surrogate loss function.*
3. *The Perceptron algorithm is easily extended to the multiclass setting.*

The first part of the proposition is due to [2, 29]. Part 2 is taken from [31]. Part 3 is based on the article of [13].

8.1 Perceptron

The Perceptron algorithm ([2, 29]) is an algorithm for finding a point in a polyhedron, where the polyhedron is given by a set of linear inequalities. In our case these inequalities are of the form $y\langle \mathbf{w}, \mathbf{x}\rangle > 0$ for all examples $(\mathbf{x}, y)$.

To facilitate the analysis of the algorithm first we need to assume the following:

1. The sample space is bounded – for all $\mathbf{x} \in \mathcal{X}$, $\|\mathbf{x}\| \leq R$.

Algorithm 3 Perceptron

```
Initialize: $\mathbf{w}^{(1)} = 0$
for $t = 1, 2, ..., T$ do
    Receive an example $(\mathbf{x}^{(t)}, y^{(t)})$.
    if the example is misclassified, meaning $y^{(y)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)} \rangle \leq 0$ then
        $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + y^{(t)}\mathbf{x}^{(t)}$
    else
        $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)}$
    end if
end for
return $\mathbf{w}^{(T+1)}$
```

2. There exists a halfspace $\mathbf{w}^\star$ with $\|\mathbf{w}^\star\| \leq B$ such that $\mathbf{w}^\star$ separates the data with margin. Namely for each example $(\mathbf{x}, y)$, $y\langle \mathbf{w}^\star, \mathbf{x} \rangle \geq 1$.

The next theorem implies that after a bounded number of mistakes, the Perceptron algorithm returns the optimal predictor.

Theorem 8 ([28, 7])**.** *Suppose the conditions above hold, then Perceptron makes at most $(RB)^2$ mistakes.*

8.1.1 Dependence on margin

The reader should be aware that there are versions of Perceptron (not online) which find an optimal halfspace without their runtime being dependent on the margin. For example, see [18].

8.2 Analysis for the inseparable case

In this section we will present an analysis for Perceptron which is based on online learning bounds ([31]). In particular, we will define the notion of a *data-dependent surrogate loss.*

The key notion is that the Perceptron algorithm can be seen as gradient descent over a series of loss functions (with a constant step size of 1). At time t we have our current predictor $\mathbf{w}^{(t)}$ and are given an example $(\mathbf{x}^{(t)}, y^{(t)})$. Let us define the loss function $\ell^{(t)}_{\text{Perc}}$ as $1 - y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)} \rangle$ if $\mathbf{w}^{(t)}$ mistakes on $(\mathbf{x}^{(t)}, y^{(t)})$ and as zero otherwise.

These loss functions have a couple of important properties:

1. At $\mathbf{w}^{(t)}$ we have $\ell^{(t)}_{\text{Perc}}(\mathbf{w}^{(t)}) \geq \mathbb{1}_{[y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle \leq 0]}$.
2. Let $\mathbf{z}^{(t)}$ be the gradient of $\ell^{(t)}_{\text{Perc}}$ at $\mathbf{w}^{(t)}$. Then $\mathbf{z}^{(t)} = y^{(t)}\mathbf{x}^{(t)}$ if $\mathbf{w}^{(t)}$ mistakes on $(\mathbf{x}^{(t)}, y^{(t)})$ and $\mathbf{z}^{(t)} = 0$ otherwise.

By applying any standard online convex optimization algorithm (such as gradient descent or RFTL) on these loss functions, we have the following regret bound:

$$\sum_{t=1}^{T} \ell^{(t)}_{\text{Perc}}(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} \ell^{(t)}_{\text{Perc}}(\mathbf{u}) \leq \frac{1}{2\beta}\|\mathbf{u}\|^2 + \frac{\beta}{2}\sum_{t=1}^{T}\|\mathbf{z}^{(t)}\|^2$$

for any $\|\mathbf{u}\| \leq B$ and β is a parameter of the algorithm.

Let M be the number of mistakes the algorithm makes. Then,

$$M - \sum_{t=1}^{T} \ell^{(t)}_{\text{Perc}}(\mathbf{u}) \leq \frac{1}{2\beta}\|\mathbf{u}\|^2 + \frac{\beta}{2}MR^2$$

We can now optimize over β, rearrange and get the expression

$$M - R\|\mathbf{u}\|\sqrt{M} - \sum_{t=1}^{T} \ell^{(t)}_{\text{Perc}}(\mathbf{u}) \leq 0$$

which entails that

$$M \leq \sum_{t=1}^{T} \ell^{(t)}_{\text{Perc}}(\mathbf{u}) + R\|\mathbf{u}\|\sqrt{\sum_{t=1}^{T} \ell^{(t)}_{\text{Perc}}(\mathbf{u})} + R^2\|\mathbf{u}\|^2$$

As a special case, if there exists a $\mathbf{u}$ such that $y^{(t)}\langle \mathbf{u}, \mathbf{x}^{(t)}\rangle \geq 1$ for all t, then

$$M \leq R^2\|\mathbf{u}\|^2$$

This completes the proof of the second part of proposition 7.

8.3 Multiclass Perceptron

In this section we will show an extension of Perceptron to the multiclass setting.

The first way to approach this problem is to reduce to the binary case. Suppose we have k classes, note that the optimal predictor must satisfy $k-1$ linear inequalities for each example $(\mathbf{x}, y)$:

$$\forall z \neq y \ (W\mathbf{x})_y > (W\mathbf{x})_z$$

We can replace a matrix $W \in \mathbb{R}^{k\times d}$ with a (kd)-dimensional vector by concatenating its rows. Then replace each example $(\mathbf{x}, y)$ with $k-1$ (kd)-dimensional vectors denoted $\{\mathbf{x}_z\}_{z\neq y}$. Each $\mathbf{x}_z$ is split into k d-dimensional blocks. The y'th block contains the original $\mathbf{x}$, the z'th block contains $-\mathbf{x}$ and the rest of the blocks are zeros:

$$\mathbf{x}_z = (\mathbf{0}, \cdots, \underbrace{\mathbf{x}}_{y}, \mathbf{0}, \cdots, \underbrace{-\mathbf{x}}_{z}, \mathbf{0}, \cdots)$$

After applying this transformation we could run the Perceptron algorithm. Under the same assumptions we get that it makes at most $(RB)^2$ mistakes.

The reader should note that when machine learning researchers mention the multiclass Perceptron, they usually refer to the following algorithm, which stems from [13].

Algorithm 4 Multiclass Perceptron

```
Initialize: $W^{(1)} = 0$
for $t = 1, 2, ..., T$ do
    Receive an example $(\mathbf{x}^{(t)}, y^{(t)})$.
    Calculate $r = \max_{z \in [k]} (W\mathbf{x}^{(t)})_z$
    if the example is misclassified, meaning $(W\mathbf{x}^{(t)})_{y^{(t)}} \leq r$ then
        Set $\hat{y} \in \arg\max_{z \in [k]} (W\mathbf{x}^{(t)})_z$
        $W^{(t+1)} \leftarrow W^{(t)} + \mathbf{e}_{y^{(t)}} (\mathbf{x}^{(t)})^\top - \mathbf{e}_{\hat{y}} (\mathbf{x}^{(t)})^\top$
    else
        $W^{(t+1)} \leftarrow W^{(t)}$
    end if
end for
return $W^{(T+1)}$
```

Chapter 9

Random Noise

In this chapter, we will apply the technique of *data dependent surrogate* loss functions, introduced in the previous chapter, to learning halfspaces with random noise (4). Our goal is to obtain a mistake bound for this setting.

Proposition 8. *Our results are the following:*

1. *We show a simple extension of Perceptron to the random noise case.*
2. *This algorithm has a mistake bound of $O(\sqrt{T}/(\gamma(1-2\eta)))$ with high probability.*

The algorithm presented in this chapter and its analysis are novel.

9.1 Noisy Perceptron

Define the following loss family:

$$\ell^{(t)}_{\mathrm{prn}}(\mathbf{w}) = \begin{cases} \frac{1-\eta}{1-2\eta}(1 - y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle), & y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle \le 0 \\ -\frac{\eta}{1-2\eta}(1 + y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle), & y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle > 0 \end{cases}$$

We claim that this loss family is an unbiased estimate for the Percetpron loss w.r.t the original distribution. Namely, for any $\mathbf{w}$ we have that

$$\mathbb{E}_{y^{(t)}}[\ell_{\mathrm{prn}}(\mathbf{w}; (\mathbf{x}^{(t)}, y^{(t)}))|\{y^{(i)}\}_{i=1}^{t}] = \ell_{\mathrm{Perc}}(\mathbf{w}; (\mathbf{x}^{(t)}, \mathrm{sign}(\langle \mathbf{w}^{\star}, \mathbf{x}^{(t)}\rangle))) \quad (9.1)$$

Let us start by obtaining a regret bound for these loss functions. Once again by applying gradient descent, for example, with parameter β we have the following bound:

$$\sum_{t=1}^{T} \ell_{\text{prn}}^{(t)}(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} \ell_{\text{prn}}^{(t)}(\mathbf{u}) \leq \frac{1}{2\beta}\|\mathbf{u}\|^2 + \frac{\beta}{2}\sum_{t=1}^{T}\|\mathbf{z}^{(t)}\|^2$$

for any $\|\mathbf{u}\| \leq B$.

Note that we have that $\|\mathbf{z}^{(t)}\| \leq RB/(1-2\eta)$ for all $t \in [T]$. Plugging that in, setting $\mathbf{w}^\star$ for $\mathbf{u}$ and optimizing over β we get

$$\sum_{t=1}^{T} \ell_{\text{prn}}^{(t)}(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} \ell_{\text{prn}}^{(t)}(\mathbf{w}^\star) \leq \frac{RB\sqrt{T}}{1-2\eta} \tag{9.2}$$

We would now like to use that regret bound in order to obtain a high probability mistake bound. For this purpose define the following random variables:

$$V^{(t)} = \mathbb{E}_{y^{(t)}}[\ell_{\text{prn}}^{(t)}(\mathbf{w}^{(t)}) - \ell_{\text{prn}}^{(t)}(\mathbf{w}^\star)|\{y^{(i)}\}_{i=1}^{t}] - (\ell_{\text{prn}}^{(t)}(\mathbf{w}^{(t)}) - \ell_{\text{prn}}^{(t)}(\mathbf{w}^\star))$$

The sequence $\{V^{(t)}\}_{t=1}^{T}$ forms a martingale difference sequence with respect to $\{y^{(t)}\}_{t=1}^{T}$ since

$$\mathbb{E}[V^{(t)}|\{y^{(i)}\}_{i=1}^{t}] = 0$$

also, each $V^{(t)}$ is bounded since by our assumptions

$$|V^{(t)}| \leq \frac{4RB}{1-2\eta}$$

Under these conditions we can apply the Hoeffding-Azuma ([4]) inequality, such that with probability at least $1-\delta$ the following concentration bound holds:

$$\sum_{t=1}^{T} V^{(t)} \leq \frac{RB}{1-2\eta}\sqrt{8T\log\frac{1}{\delta}} \tag{9.3}$$

Let us now combine equations 9.1, 9.2 and 9.3. With probability at least $1-\delta$,

$$\sum_{t=1}^{T} \ell_{\text{Perc}}^{(t)}(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} \ell_{\text{Perc}}^{(t)}(\mathbf{w}^\star) \leq \frac{RB\sqrt{T}}{1-2\eta} + \frac{RB}{1-2\eta}\sqrt{8T\log\frac{1}{\delta}}$$

To complete the mistake bound, let us notice that $\ell^{(t)}_{\text{Perc}}(\mathbf{w}^\star) \le 0$ for all $t \in [T]$ and that $\sum_{t=1}^{T} \ell^{(t)}_{\text{Perc}}(\mathbf{w}^{(t)}) \ge M$ where M is the number of mistakes. We got:

$$M \le \frac{RB\sqrt{T}}{1-2\eta} + \frac{RB}{1-2\eta}\sqrt{8T\log\frac{1}{\delta}}$$

This completes proposition 8.

Chapter 10

Isotron

The results from chapter 5 assume that the link function g is known to the learner. However it is still possible to learn the model even when g is unknown. The *Isotron* ([22]) handles this case by combining the Perceptron algorithm with isotonic regression.

Algorithm 5 Isotron

```
Initialize: $\mathbf{w}^{(1)} = 0$
for $t = 1, 2, ..., T$ do
    Sample fresh $m$ examples $\{\mathbf{x}_i, y_i\}_{i=1}^m \subset \mathbb{R}^d \times \{0, 1\}$
    $g^{(t)} = \mathrm{PAV}((\langle \mathbf{w}^{(t)}, \mathbf{x}_1 \rangle, y_1), ..., (\langle \mathbf{w}^{(t)}, \mathbf{x}_m \rangle, y_m))$
    $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \frac{1}{m} \sum_{i=1}^m [y_i - g^{(t)}(\langle \mathbf{w}^{(t)}, \mathbf{x}_i \rangle)]\mathbf{x}_i$
end for
```

At each time t, the Isotron algorithm alternates between estimating $g^{(t)}$ given $\mathbf{w}^{(t)}$, and updating $\mathbf{w}^{(t+1)}$ given $g^{(t)}$. We will show that after enough iterations, the algorithm allows us to recover a good estimate of $\mathbf{w}^\star$ and g for which the GLM assumption holds.

Before showing the convergence guarantee, let us explain how the Isotron estimates $g^{(t)}$ at every iteration. This is done using the PAV routine: Initially, the values of $\langle \mathbf{w}, \mathbf{x}_i \rangle$ are sorted in a nondecreasing order. Each sample is assigned to its own pool. The prediction in each pool is the average label over all samples in the pool. The pools are then arbitrarily merged and the predictions updated until the resulting function is nondecreasing. Finally, linear interpolation is performed between the different pools.

Theorem 9. *Suppose* $\{\mathbf{x}_i, y_i\}_{i=1}^{mT} \subseteq \mathbb{B}^d \times \{0,1\}$ *satisfy the GLM assumption for monotonic nondecreasing L-Lipschitz* $g^\star$ *and* $\|\mathbf{w}^\star\| \leq 1$. *Suppose* $L \geq 1$, $T \geq 1$ *and* $m \geq (6T\log(eT)/L)^2$. *Then for all* $h^{(t)}(\mathbf{x}) = g^{(t)}(\langle \mathbf{w}^{(t)}, \mathbf{x}\rangle)$ *computed by the Isotron:*

$$\mathbb{E}[\sum_{t=1}^{T} \mathrm{err}(h^{(t)}) - \sum_{t=1}^{T} \mathrm{err}(f)] \leq 8L^2$$

where $\mathrm{err}(h) = (1/m)\sum_{i=1}^{m}(h(\mathbf{x}_i) - y_i)^2$, $f(\mathbf{x}) = g^\star(\langle \mathbf{w}^\star, \mathbf{x}\rangle)$ *and* $\mathbb{B}^d$ *is the d-dimensional unit ball.*

10.1 Extension to multiclass

In this section we will present an extension of Isotron to the multiclass setting.

Recall that in the multiclass setting the link function g was a gradient of a k-dimensional convex function Φ. That means that estimating g is equivalent to estimating Φ. Unfortunately, this is an extremely rich class; the sample complexity of estimating a uniformly bounded convex, Lipschitz function in k dimensions grows exponentially with k ([9]).

To overcome this problem, the approach taken in [1] is to assume that g^{-1} is a linear combination of a finite basis of functions. In their experiments, they typically take this basis to be the set of all monomials of degree at most 3.

They then give an algorithm for learning under this setting. The algorithm does not estimate g^{-1} but only the conditional probabilities $\mathbb{E}[\mathbf{e}_y|\mathbf{x}]$ for each example in the training set. Also, the algorithm does not come with proper theoretical guarantees.

Chapter 11

Online-to-Offline conversion

In this chapter we show the following:

Proposition 9. *For certain learning problems, online algorithms which optimize data dependent surrogates can converted to batch convex optimization problems.*

This novel result reveals a certain connection between online and batch learning algorithms, as well as a nontrivial re-parametrization of Isotron as a convex surrogate loss minimization problem.

11.1 Setup

Throughout the chapter we will consider a series of data dependent surrogates $\{\ell^{(t)}\}_{t=1}^{T}$. These are surrogates for the true error measure we would like to minimize, denoted as $\mathrm{err}^{(t)}(\mathbf{w})$ for any halfspace $\mathbf{w}$. We assume that $\mathbf{w}^{\star}$ has an error of 0.

Assume that each surrogate $\ell^{(t)}$ is fully determined by the current example $(\mathbf{x}^{(t)}, y^{(t)})$ and the previous iterate $\mathbf{w}^{(t)}$. Note that the previous iterates are themselves random variables determined by $y^{(1)}, ..., y^{(t)}$. We need our surrogates to fulfill two important properties:

1. At any time, the expected loss of $\mathbf{w}^{\star}$ is not much larger than the optimal error independently of the previous actions of the learner. Formally,

$$\mathbb{E}_{y^{(t)}}\left[\sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\mathbf{w}^{\star})\right] \leq \epsilon$$

where $\epsilon = O(\sqrt{1/T})$ (usually $\epsilon = 0$).

2. At any time t, given any sequence of labels $y^{(1)}, ..., y^{(t-1)}$, the expected loss of $\mathbf{w}^{(t)}$ upper bounds the error at $\mathbf{w}^{(t)}$:

$$\mathbb{E}_{y^{(t)}}[\ell^{(t)}(\mathbf{w}^{(t)})|\{y^{(i)}\}_{i=1}^{t-1}] \geq \text{err}^{(t)}(\mathbf{w}^{(t)})$$

In the online setting, the goal of the learner is to have sublinear regret in terms of the error, namely:

$$\sum_{t=1}^{T} \text{err}^{(t)}(\mathbf{w}^{(t)}) \leq T\epsilon + O(\sqrt{T})$$

for any sequence $\mathbf{x}^{(1)}, ..., \mathbf{x}^{(T)}$.

In the remainder of the chapter we will show that the properties presented above are enough to ensure that. We will also that this setup is enough to create loss function such that when minimized in the batch setting will ensure

$$\mathbb{E}\text{err}^{(1)}(\hat{\mathbf{w}}) \leq \epsilon + O\left(\sqrt{\frac{1}{T}}\right)$$

where $\hat{\mathbf{w}}$ is the minimizer of the surrogate loss. In this setting we have some underlying distribution generating our sample $\mathbf{x}^{(1)}, ..., \mathbf{x}^{(T)}$ and we would like the above inequality to hold in expectation. Note that here we assume that the expected err are interchangeable.

11.2 Examples

In this section we will see that those properties hold for the methods presented in this part.

11.2.1 Perceptron

The Perceptron requires the realizable assumption and so the labels $\{y^{(t)}\}_{t=1}^{T}$ and the iterates $\{\mathbf{w}^{(t)}\}_{t=1}^{T}$ are all deterministic. Our error measure is the zero-one loss $\text{err}^{(t)}(\mathbf{w}^{(t)}) = \mathbb{1}_{[y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle \leq 0]}$. Recall that the Perceptron loss is

$$\ell_{\text{Perc}}^{(t)}(\mathbf{w}) = \begin{cases} 1 - y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle, & y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle \leq 0 \\ 0, & \text{otherwise} \end{cases}$$

For the first property:

$$\sup_{\mathbf{w}^{(t)}} \ell^{(t)}_{\text{Perc}}(\mathbf{w}^\star) = \max\{0, 1 - y^{(t)}\langle \mathbf{w}^\star, \mathbf{x}^{(t)}\rangle\} = 0$$

where $\epsilon = 0$, and for the second property:

$$\ell^{(t)}_{\text{Perc}}(\mathbf{w}^{(t)}) \geq \text{err}^{(t)}(\mathbf{w}^{(t)})$$

11.2.2 Noisy Perceptron

Recall the loss functions for the noisy Perceptron were:

$$\ell^{(t)}_{\text{prn}}(\mathbf{w}) = \begin{cases} \frac{1-\eta}{1-2\eta}(1 - y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle), & y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle \leq 0 \\ -\frac{\eta}{1-2\eta}(1 + y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle), & y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle > 0 \end{cases}$$

Our error measure is $\text{err}^{(t)}(\mathbf{w}^{(t)}) = \mathbb{1}_{[\text{sign}(\langle \mathbf{w}^\star, \mathbf{x}^{(t)}\rangle)\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle \leq 0]}$.

For the first property:

$$\begin{aligned} &\mathbb{E}_{y^{(t)}}[\sup_{\mathbf{w}^{(t)}} \ell^{(t)}_{\text{prn}}(\mathbf{w}^\star)] \\ &= \mathbb{E}_{y^{(t)}} \frac{1}{1-2\eta} \max\{(1-\eta)(1 - y^{(t)}\langle \mathbf{w}^\star, \mathbf{x}^{(t)}\rangle), -\eta(1 + y^{(t)}\langle \mathbf{w}^\star, \mathbf{x}^{(t)}\rangle)\} \\ &\leq 0 \end{aligned}$$

where, once again, $\epsilon = 0$. For the second property,

$$\begin{aligned} \mathbb{E}_{y^{(t)}}[\ell^{(t)}_{\text{prn}}(\mathbf{w}^{(t)}; (\mathbf{x}^{(t)}, y^{(t)})) | y^{(1)}, ..., y^{(t-1)}] &= \ell^{(t)}_{\text{Perc}}(\mathbf{w}^{(t)}; (\mathbf{x}^{(t)}, \text{sign}(\langle \mathbf{w}^\star, \mathbf{x}^{(t)}\rangle))) \\ &\geq \text{err}^{(t)}(\mathbf{w}^{(t)}) \end{aligned}$$

11.2.3 Isotron

This example is somewhat more complex than the previous two. Each loss function $\ell^{(t)}$ observes m examples (instead of just one). Recall that we assume there is a halfspace $\mathbf{w}^\star$ and a monotone nondecreasing and L-Lipschitz function $g^\star$ such that

$$\Pr[y^{(t)}_i = 1] = g^\star(\langle \mathbf{w}^\star, \mathbf{x}^{(t)}_i\rangle)$$

for all $t \in [T]$ and $i \in [m]$. We also assume that $\|\mathbf{w}^\star\| = 1$ and that $\|\mathbf{x}\| \leq 1$ for all $\mathbf{x} \in \mathcal{X}$.

For simplicity, denote $\hat{y}_i^{(t)} = g^{(t)}(\langle \mathbf{w}^{(t)}, \mathbf{x}_i^{(t)} \rangle)$. Then the error measure is

$$\mathrm{err}^{(t)}(\mathbf{w}^{(t)}) = \frac{1}{Lm}\sum_{i=1}^{m}(g^\star(\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)} \rangle) - \hat{y}_i^{(t)})^2$$

and the loss function is

$$\ell^{(t)}(\mathbf{w}) = \mathrm{err}^{(t)}(\mathbf{w}^{(t)}) + \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i^{(t)} - y_i^{(t)})\langle \mathbf{w}, \mathbf{x}_i^{(t)} \rangle$$

We would now like to show that our two properties hold. To show the first property, first we need to extend $g^\star$ such that it is tightly bounded in $[0,1]$. We will extend it to the interval $[-2,2]$ (by linear interpolation) such that $g^\star(-2) = 0$ and $g^\star(2) = 1$. The extension allows us to define its inverse:

$$v(y) = \inf\{s \in [-2,2] : g^\star(s) = y\}$$

By lemma 3 from [22] we have the following import statements:

$$\sum_{i=1}^{m}(\hat{y}_i^{(t)} - y_i^{(t)})v(\hat{y}_i^{(t)}) = 0 \tag{11.1}$$

$$(g^\star(\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)} \rangle) - \hat{y}_i^{(t)})(\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)} \rangle - v(\hat{y}_i^{(t)})) \geq \frac{1}{L}(g^\star(\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)} \rangle) - \hat{y}_i^{(t)})^2 \tag{11.2}$$

where 11.1 is a property of the PAV algorithm and 11.2 comes from the fact that $g^\star$ is nondecreasing and Lipchitz. Then we have that

$$\begin{aligned}
&\mathbb{E}_{\{y_i^{(t)}\}_{i=1}^m}\left[\sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\mathbf{w}^\star)\right] \\
&= \mathbb{E}_{\{y_i^{(t)}\}_{i=1}^m} \sup_{\mathbf{w}^{(t)}} \left\{ \mathrm{err}^{(t)}(\mathbf{w}^{(t)}) + \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i^{(t)} - y_i^{(t)})\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)} \rangle \right\} \\
&= \mathbb{E}_{\{y_i^{(t)}\}_{i=1}^m} \sup_{\mathbf{w}^{(t)}} \left\{ \mathrm{err}^{(t)}(\mathbf{w}^{(t)}) + \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i^{(t)} - g^\star(\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)} \rangle))\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)} \rangle \right\}
\end{aligned}$$

focusing on the second summand,

$$\begin{aligned}&\sum_{i=1}^{m}(\hat{y}_i^{(t)} - g^{\star}(\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle))\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle \\ &\quad= \sum_{i=1}^{m}(\hat{y}_i^{(t)} - g^{\star}(\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle))(\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle - v(\hat{y}_i^{(t)})) \\ &\quad+ \sum_{i=1}^{m}(\hat{y}_i^{(t)} - g^{\star}(\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle))v(\hat{y}_i^{(t)})\end{aligned}$$

By inequality 11.2 the first summand is at most $-\mathrm{err}^{(t)}(\mathbf{w}^{(t)})$. By equation 11.1 the second summand equals to

$$\sum_{i=1}^{m}(y_i^{(t)} - g^{\star}(\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle))v(\hat{y}_i^{(t)})$$

Putting it all together, so far we have got that

$$\mathbb{E}_{\{y_i^{(t)}\}_{i=1}^m}\left[\sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\mathbf{w}^{\star})\right] \le \mathbb{E}_{\{y_i^{(t)}\}_{i=1}^m}\left[\sup_{\mathbf{w}^{(t)}} \frac{1}{m}\sum_{i=1}^{m}(y_i^{(t)} - g^{\star}(\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle))v(\hat{y}_i^{(t)})\right]$$

Note that the sequence $y_i^{(t)} - g^{\star}(\langle \mathbf{w}^{\star}, \mathbf{x}_i^{(t)}\rangle)$ is bounded IID with mean zero and that the $v(\hat{y}_i^{(t)})$ is a bounded and nondecreasing sequence. Then to bound the above expression we need the following concentration inequality.

Lemma 5 ([22]). *For any integer $m \ge 1$ and reals $a < b$ let $X_1, ..., X_m$ be independent random variables, with $\mathbb{E}[X_i] = 0$, each in the bounded range $[-1, 1]$. Let $A_1, ..., A_m$ be a sequence of random variables such that $a \le A_1 \le A_2 \le ... \le A_m \le b$. Note that the sequence A_i need not be independent and may depend on the X_is as well. Then,*

$$\mathbb{E}\left[\frac{A_1X_1 + ... + A_mX_m}{m}\right] \le (b-a)\sqrt{\frac{2}{m}}$$

Even though the lemma isnt stated in this manner, we could take a supremum over $A_1, ..., A_m$ inside the expectation as long as they are monotone nondecreasing and bounded in $[a, b]$. The proof of the lemma would remain the same.

Finally, the first property holds for $\epsilon = 4\sqrt{2/m}$ since

$$\mathbb{E}_{\{y_i^{(t)}\}_{i=1}^m}[\sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\mathbf{w}^\star)] \le \epsilon$$

Note that here $m = \Omega(T)$ and our assumption about ϵ holds.

To show that the second property holds, notice that the PAV algorithm solves

$$\min_{\{\hat{y}_i^{(t)}\}_{i=1}^m \text{nondecreasing}} \sum_{i=1}^m (\hat{y}_i^{(t)} - y_i^{(t)})^2$$

and that the solution $\hat{y}_i^{(t)} + \langle \mathbf{w}^{(t)}, \mathbf{x}_i^{(t)} \rangle$ is feasible since it is nondecreasing with respect to $\langle \mathbf{w}^{(t)}, \mathbf{x}_i^{(t)} \rangle$. Then by optimality conditions for convex programs we have that

$$\sum_{i=1}^m (\hat{y}_i^{(t)} - y_i^{(t)})(\hat{y}_i^{(t)} - (\hat{y}_i^{(t)} + \langle \mathbf{w}^{(t)}, \mathbf{x}_i^{(t)} \rangle)) \le 0$$

and so

$$\sum_{i=1}^m (\hat{y}_i^{(t)} - y_i^{(t)}) \langle \mathbf{w}^{(t)}, \mathbf{x}_i^{(t)} \rangle \ge 0$$

That entails that the second property holds since

$$\mathbb{E}_{\{y_i^{(t)}\}_{i=1}^m}[\ell^{(t)}(\mathbf{w}^{(t)}) | \{y_i^{(1)}\}_{i=1}^m, ..., \{y_i^{(t-1)}\}_{i=1}^m] \ge \mathrm{err}^{(t)}(\mathbf{w}^{(t)})$$

11.3 Online setting

In this section we will show how the two properties are enough for online learning. For that we assume that $\|\mathbf{z}^{(t)}\| \le R$ (the gradients of $\ell^{(t)}$ at $\mathbf{w}^{(t)}$ are bounded), $\|\mathbf{w}^\star\| \le B$ and $|\ell^{(t)}(\mathbf{w})| \le C$ for all $\|\mathbf{w}\| \le B$ (the losses are bounded).

Recall that we can obtain the following regret bound

$$\sum_{t=1}^T \ell^{(t)}(\mathbf{w}^{(t)}) - \sum_{t=1}^T \ell^{(t)}(\mathbf{w}^\star) \le \sqrt{2RBT} \tag{11.3}$$

Define the following random variable:

$$U^{(t)} = \mathbb{E}_{y^{(t)}}[\ell^{(t)}(\mathbf{w}^{(t)}) - \ell^{(t)}(\mathbf{w}^\star) | \{y^{(i)}\}_{i=1}^{t-1}] - (\ell^{(t)}(\mathbf{w}^{(t)}) - \ell^{(t)}(\mathbf{w}^\star))$$

The sequence $\{U^{(t)}\}_{t=1}^T$ for a bounded martingale difference sequence. By the Hoeffding-Azuma inequality with probability at least $1-\delta$:

$$\sum_{t=1}^T U^{(t)} \le C\sqrt{8T\log\frac{1}{\delta}} \tag{11.4}$$

Combining inequalities 11.3 and 11.4 we have that

$$\sum_{t=1}^T \mathbb{E}_{y^{(t)}}[\ell^{(t)}(\mathbf{w}^{(t)}) - \ell^{(t)}(\mathbf{w}^\star)|\{y^{(i)}\}_{i=1}^{t-1}] \le \sqrt{2RBT} + C\sqrt{8T\log\frac{1}{\delta}}$$

then our two properties entail that

$$\sum_{t=1}^T \mathrm{err}^{(t)}(\mathbf{w}^{(t)}) \le \epsilon T + \sqrt{2RBT} + C\sqrt{8T\log\frac{1}{\delta}}$$

as required.

11.4 Offline setting

In this section we will show how the two properties relate two the offline setting. Formally, we would like to solve the following convex program:

$$\min_{\mathbf{w}} \frac{1}{T}\sum_{t=1}^T \sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\mathbf{w})$$

over a sample $S = \{(\mathbf{x}^{(t)}, y^{(t)})\}_{t=1}^T$ taken independently from the distribution $\mathcal{D}$.

Let $\hat{\mathbf{w}}$ be a minimizer of the above program. Then,

$$\mathbb{E}_S\left[\frac{1}{T}\sum_{t=1}^T \sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\hat{\mathbf{w}})\right] \le \mathbb{E}_S\left[\frac{1}{T}\sum_{t=1}^T \sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\mathbf{w}^\star)\right] \le \epsilon$$

by the first property. Also,

$$\begin{aligned}
\mathbb{E}_S\left[\frac{1}{T}\sum_{t=1}^T \sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\hat{\mathbf{w}})\right] &= \mathbb{E}_S\left[\frac{1}{T}\sum_{t=1}^T \mathbb{E}_{y^{(t)}}\left[\sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\hat{\mathbf{w}})|\{y^{(i)}\}_{i=1}^{t-1}\right]\right] \\
&\ge \mathbb{E}_S\left[\frac{1}{T}\sum_{t=1}^T \mathrm{err}^{(t)}(\hat{\mathbf{w}})\right] \\
&= \mathbb{E}\mathrm{err}^{(1)}(\hat{\mathbf{w}})
\end{aligned}$$

Here the first equality is by the tower property of expectation and the second equality holds since we assume that the error measures are interchangeable in expectation. Since we are taking the supremum over $\mathbf{w}^{(t)}$, replacing it with $\hat{\mathbf{w}}$ would only decrease the expression. Afterwards the inequality follows by the second property.

11.5 More examples

In this section we will show what the batch version of the three examples look like.

Perceptron For this we have for all $t \in [T]$,

$$\sup_{w^{(t)}} \ell^{(t)}(\mathbf{w}) = \max\{0, 1 - y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle\} = \ell_{\text{hinge}}(\mathbf{w})$$

Noisy Perceptron Here:

$$\begin{aligned}\sup_{w^{(t)}} \ell^{(t)}(\mathbf{w}) &= \max\left\{\frac{1-\eta}{1-2\eta}(1 - y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle), -\frac{\eta}{1-2\eta}(1 + y^{(t)}\langle \mathbf{w}, \mathbf{x}^{(t)}\rangle)\right\} \\ &= \ell_{\text{rm}}(\mathbf{w})\end{aligned}$$

Isotron For each $\mathbf{w}$ denote $g_{\mathbf{w}}^{(t)}$ as the function returned from the PAV algorithm for $\mathbf{w}$ over the sample $\{(\mathbf{x}_i^{(t)}, y_i^{(t)})\}_{i=1}^m$.

Denote $\mathcal{G}^{(t)} = \{\mathbf{x} \mapsto g_{\mathbf{w}}^{(t)}(\langle \mathbf{w}, \mathbf{x}\rangle) : \|\mathbf{w}\| \leq 1\}$, then

$$\sup_{\mathbf{w}^{(t)}} \ell^{(t)}(\mathbf{w}) =$$

$$\sup_{g \in \mathcal{G}^{(t)}} \left\{\frac{1}{Lm}\sum_{i=1}^m (g(\mathbf{x}_i^{(t)}) - g^\star(\langle \mathbf{w}^\star, \mathbf{x}_i^{(t)}\rangle))^2 + \frac{1}{m}\sum_{i=1}^m (g(\mathbf{x}_i^{(t)}) - y_i^{(t)})\langle \mathbf{w}, \mathbf{x}_i^{(t)}\rangle\right\}$$

Appendix A

Online learning primer

This will be a short primer on online learning, based on [31]. The reader is also referred to [11] for a more thorough introduction.

A.1 Online learning

Online learning is usually thought of as a game between a learner and an adversary. The game goes on for T rounds, in each round the adversary supplies the learner with an example $\mathbf{x}^{(t)}$ and the learner has to predict its label as $p^{(t)}$. The learner then receives the true label $y^{(t)}$ from the adversary. The goal of the learner is to make the least number of mistakes, typically sublinear in T.

Algorithm 6 Online learning

```
for $t = 1, 2, ..., T$ do
    Receive example $\mathbf{x}^{(t)} \in \mathcal{X}$.
    Predict $p^{(t)} \in \mathcal{Y}$.
    Receive true label $y^{(t)} \in \mathcal{Y}$.
    Suffer loss $\mathbb{1}_{[y^{(t)} \neq p^{(t)}]}$.
end for
```

A.2 Online convex optimization

We will also require the setup of online convex optimization. It can be formulated in a similar manner to the online learning game. At each iteration t the learner presents a predictor $\mathbf{w}^{(t)}$ residing in some convex set S. It then receives a convex loss function $\ell^{(t)} : S \to \mathbb{R}$ and suffers a loss $\ell^{(t)}(\mathbf{w}^{(t)})$.

Algorithm 7 Online convex optimization

```
for $t = 1, 2, ..., T$ do
    Predict $\mathbf{w}^{(t)} \in S$.
    Receive loss function $\ell^{(t)} : S \to \mathbb{R}$.
    Suffer loss $\ell^{(t)}(\mathbf{w}^{(t)})$.
end for
```

In this setting we will require the learner to have a small regret (typically sublinear in T) with respect to any vector in S. The regret of the learner with respect to $\mathbf{u} \in S$ is

$$\text{regret}(\mathbf{u}) = \sum_{t=1}^{T} \ell^{(t)}(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} \ell^{(t)}(\mathbf{u})$$

Suppose the $\ell^{(t)}$'s are differentiable, one of the main algorithms for learning is gradient descent. Let $\beta > 0$ be a step size parameter, the predictor at time t is $\mathbf{w}^{(t)} = -\beta \sum_{i=1}^{t} \mathbf{z}_i$ where $\mathbf{z}_i = \nabla \ell^{(i)}(\mathbf{w}^{(i)})$. This achieves a regret with respect to any $\mathbf{u} \in S$:

$$\text{regret}(\mathbf{u}) \leq \frac{1}{2\beta} \|\mathbf{u}\|^2 + \beta \sum_{t=1}^{T} \|\mathbf{z}_t\|^2$$

Notes

Bibliography

[1] Alekh Agarwal, Sham M Kakade, Nikos Karampatziakis, Le Song, and Gregory Valiant. Least squares revisited: Scalable approaches for multi-class prediction. *arXiv preprint arXiv:1310.1949*, 2013.

[2] Shmuel Agmon. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6(3):382–392, 1954.

[3] Javed A Aslam and Scott E Decatur. Improved noise-tolerant learning and generalized statistical queries. Technical report, Citeseer, 1994.

[4] Kazuoki Azuma et al. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal*, 19(3):357–367, 1967.

[5] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

[6] Shai Ben-David, David Loker, Nathan Srebro, and Karthik Sridharan. Minimizing the misclassification error rate using a surrogate convex loss. *arXiv preprint arXiv:1206.6442*, 2012.

[7] HD Block. The perceptron: A model for brain functioning. i. *Reviews of Modern Physics*, 34(1):123, 1962.

[8] Avrim Blum, Alan Frieze, Ravi Kannan, and Santosh Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22(1-2):35–52, 1998.

[9] EM Bronshtein. ε-entropy of convex sets and functions. *Siberian Mathematical Journal*, 17(3):393–398, 1976.

[10] FP Cantelli. Intorno ad un teorema fondamentale della teoria del rischio. *Boll. Assoc. Attuar. Ital.(Milan)*, pages 1–23, 1910.

[11] Nicolo Cesa-Bianchi, Gábor Lugosi, et al. *Prediction, learning, and games*, volume 1. Cambridge University Press Cambridge, 2006.

[12] Edith Cohen. Learning noisy perceptrons by a perceptron in polynomial time. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 514–523. IEEE, 1997.

[13] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 1–8. Association for Computational Linguistics, 2002.

[14] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[15] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292, 2002.

[16] Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. The complexity of learning halfspaces using generalized linear methods. *arXiv preprint arXiv:1211.0616*, 2012.

[17] Amit Daniely and Shai Shalev-Shwartz. Complexity theoretic limitations on learning dnf's. *CoRR*, abs/1404.3378, 2014.

[18] John Dunagan and Santosh Vempala. A simple polynomial-time rescaling algorithm for solving linear programs. *Mathematical Programming*, 114(1):101–114, 2008.

[19] Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM Journal on Computing*, 39(2):606–645, 2009.

[20] David A Freedman. On tail probabilities for martingales. *the Annals of Probability*, pages 100–118, 1975.

[21] Venkatesan Guruswami and Prasad Raghavendra. Hardness of learning halfspaces with noise. *SIAM Journal on Computing*, 39(2):742–765, 2009.

[22] Adam Tauman Kalai and Ravi Sastry. The isotron algorithm: High-dimensional isotonic regression. In *COLT*, 2009.

[23] Michael Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 45(6):983–1006, 1998.

[24] Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

[25] Phil Long and Rocco Servedio. Consistency versus realizable h-consistency for multiclass classification. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 801–809, 2013.

[26] Nagarajan Natarajan, Inderjit Dhillon, Pradeep Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in Neural Information Processing Systems*, pages 1196–1204, 2013.

[27] Yurii Nesterov and IU E Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer, 2004.

[28] Albert BJ Novikoff. On convergence proofs for perceptrons. Technical report, DTIC Document, 1963.

[29] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[30] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *Computational learning theory*, pages 416–426. Springer, 2001.

[31] Shai Shalev-Shwartz. Online learning and online convex optimization. 2011.

[32] Shai Shalev-Shwartz, Ohad Shamir, and Karthik Sridharan. Learning kernel-based halfspaces with the 0-1 loss. *SIAM Journal on Computing*, 40(6):1623–1646, 2011.

[33] Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *arXiv preprint arXiv:1309.2375*, 2013.

[34] Jack Sherman and Winifred J Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, pages 124–127, 1950.

[35] Ambuj Tewari and Peter L Bartlett. On the consistency of multiclass classification methods. *The Journal of Machine Learning Research*, 8:1007–1025, 2007.

[36] Jason Weston and Chris Watkins. Multi-class support vector machines. Technical report, Citeseer, 1998.

[37] Tong Zhang. Statistical analysis of some multi-category large margin classification methods. *The Journal of Machine Learning Research*, 5:1225–1251, 2004.

[38] Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, pages 56–85, 2004.