MATH 462 LECTURE NOTES: LECTURE OCT 24, 2022
BINARY CLASSIFICATION

ADAM M. OBERMAN

1. Vector features

The most effective machine learning models are based on linear models using vector features. In other words, the raw data d_i are mapped to vectors $x_i \in [0,1] \subset \mathbb{R}^n$. Then these features are used for

There are several different settings of features we need to consider:

1.1. **human designed features.** We have raw data, d_i and we work with a (human designed) feature map, which might involve taking a measurement. In this section we consider vector features. When the features are fixed, usually just write $x_i \in \mathbb{R}^d$ for the features. For human designed features we usually assume

> The features are low dimensional ($d = 1, 2, 3$) and semantically meaningful: similar features correspond to similar data).

Example 1.1. The data could be different people, and the features could be their height (in cm) and their weight (in kilograms).

1.2. **math features.** In this case, we have low dimensional raw data, and we have a model for the target function. For example, polynomial, or oscillatory. Then we can use a vector of polynomial features,

$$f(x) = (1, x, x^2, \dots x^{d-1}), \quad x \in \mathbb{R}$$

Example 1.2. We can have raw data $x \in \mathbb{R}$ and features $f_i(x) = x^i$, for $i = 1, \dots, d$, which corresponds to fitting one dimensional data with a polynomial. Here we are looking for a nonlinear classification boundary. See Figure 2.

1.3. **Signal processing.** In this case x is a physical waveform, e.g. sound wave, electrical current, and the features come from a digital measurement device,

$$f(x) = (a_0, \dots, a_d), \quad x \approx \sum_i a_i \sin(2\pi\theta_i)$$

1.4. **Machine generated features.**

Example 1.3. Let d_i be handwritten digits (on paper) and let x_i be the vector features of a digital image. Then d_i is given by `https://en.wikipedia.org/wiki/MNIST_database`. However x_i is just long list of $n = 32 \times 32$ digits

Or x can be light hitting a digital camera, and $f(x)$ is the RGB vector color intensities, a digital image.

Date: November 2, 2022.

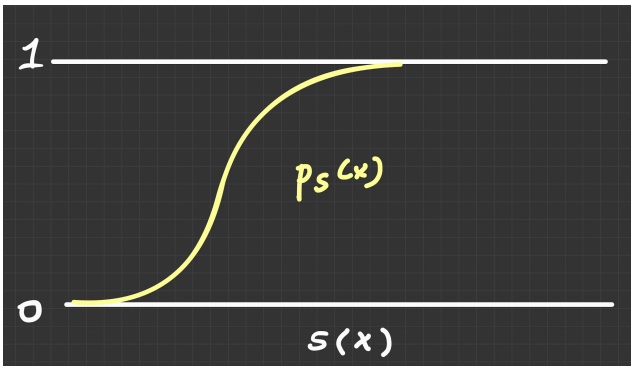

FIGURE 1. Illustration of the logistic function

Machine generated data is high dimensional and not semantically meaningful. (Nearby vectors do not correspond to similar images).

2. Linear models and Neural Networks

Write a general hypothesis class:

$$\mathcal{H} = \{s_w : \mathcal{X} \to \mathbb{R} \mid w \in \mathcal{W}\}$$

The main models to consider are

- linear models (when the features are meaningful)
- neural network models (for machine data)

2.1. **linear models.** In this setting, for $x \in \mathbb{R}^d$, we consider the hypothesis class consisting of linear (actually affine) models

$$\mathcal{H} = \{s_w(x) \mid s_w(x) = w \cdot x + w_0\}$$

Here $s_w(x)$ corresponds to the score of x. We want higher scores to corresponds to higher probability of classification, as in Figure 1.

Remark 2.1. Notational trick: we can absorb the w_0 by adding an extra dimension, whose value is always equal to one. $x \in \mathbb{R}^d \to (x, 1) \in \mathbb{R}^{d+1}$. $w \in \mathbb{R}^d \to (w, w_0) \in \mathbb{R}^{d+1}$

$$s_w(x) = w \cdot x = \sum_{i=1}^{d} w_i x_i, \quad w \in \mathbb{R}^d$$

2.2. **Neural Networks.** Neural Networks consist of one more (e.g 32) linear layers, followed by componentwise-nonlinearity.

$$L^{(j)}(x) = W^{(j)} \cdot x + b^{(j)},$$

with

$$F^{(j)}(x) = \sigma_{Relu}(L^{(j)}(x))$$

where $\sigma_{Relu} : \mathbb{R}^d \to R^d$, is componentwise function $t \mapsto \max(t, 0)$. Other nonlinearities can be used.

Then the neural network features are given by composition

$$f(x) = F^{(L)} \circ \dots F^{(1)}(x)$$

Then the final classification is performed by a linear model using the features above.

> The cool thing is the the neural network will learn the features and the linear classifier, at the same time, using the same method as we would for a linear classifier. So first we will explain linear classification. Then we will reiterate it for a neural network. So first x_i will be (semantically meaningful) vector data. Then, in the high dimensional case, we will replace $x_i = f(x_i)$ with (hopefully) semantically meaningful features.

3. Introduction to binary classification: errors

Reference for this section

- [Mur12, Chapter 8] (mostly the first equation) or [Mur22, Section 5.1.2].
- review earlier notes on logistic and softmax. Will be used in this material.

3.1. **Binary classification setup.** In the general classification problem, the target set $\mathcal{Y}$ is a set of discrete labels. Here we consider the case of binary classification consisting, so there are two labels, which we denote by $-1, +1$, and we write

$$\mathcal{Y} = \mathcal{Y}_{\pm} = \{-1, +1\}$$

Sometimes the target set will instead be

$$\mathcal{Y} = \mathcal{Y}_2 = \{0, 1\}$$

(Because for K classification, we will use $\mathcal{Y}_K = \{0, \dots, K-1\}$.)

is used instead, for convenience. It's important to keep track of which one is used.

We are given a dataset (S_m) consisting of m pairs of (x_i, y_i), $i = 1, \dots, m$, of data, $x_i \in \mathcal{X}$ and labels, $y_i \in \mathcal{Y}$,

$$S_m = \{(x_1, y_1), \dots, (x_m, y_m)\} \tag{S_m}$$

Definition 3.1 (Error)**.** The error, or zero-one loss, is given by $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, 1]$,

$$\ell_{0,1}(y_1, y_2) = \begin{cases} 0 & y_1 = y_2 \\ 1 & \text{otherwise} \end{cases}$$

Given a function $c : \mathcal{X} \to \mathcal{Y}$ and the dataset (S_m), the error of the model on the dataset is given by

$$\widehat{L}_{0-1}(c) = \frac{1}{m} \sum_{i=1}^{m} \ell_{0-1}(c(x_i), y_i) \tag{EE}$$

We also call (EE) the zero-one loss.

Note: when (S_m) is a training, or test set, we call (EE) the training or test error, respectively. When expressed as a percentage, it's called the test/training *accuracy*

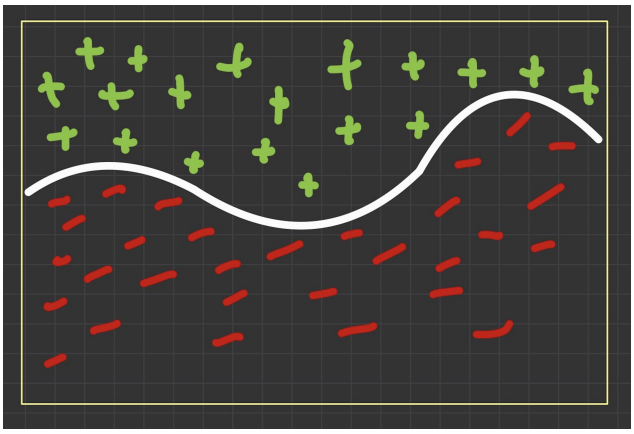

FIGURE 2. Classification problem: nonlinear feature map leads to a nonlinear boundary. However, the model $h_w(x) = w \cdot f(x)$ is still linear as a function of w. White line: the classification boundary $h_w(x) = 0$.

3.2. **Task versus loss for binary classification.** In binary classification the machine learning *task* is to train a model h using the dataset (S_m) to accurately predict classes on unseen data.

When we assume that the training dataset is drawn iid from the same distribution $\rho(x)$ as the unseen data. This is called *in-distribution* generalization. In distribution generalization is mathematically defined, and we can prove that certain (e.g. linear) models generalize (in a probabilistic sense).

Definition 3.2 (Classification Task)**.** The task for binary classification is to train a model on (S_m) to have high test accuracy.

As we saw earlier, using the zero-one loss and discrete models may not work well.

Definition 3.3 (Classification Method)**.** The method(s) used for classification are to (i) use differentiable models (ii) differentiable surrogate losses to achieve the classification task.

There are several reasons for changing the loss. These can be

(1) Algorithmic: differentiable models are easier to train
(2) Statistical: linear models generalize better
(3) Generative: logistic regression also gives a model of the data, so can generate new points.
(4) Geometric: the SVM model/loss gives a geometric picture of the classification boundary, so we can talk about marginal and confident data points.

All these reasons can be confusing. There are different things we can choose to focus on. In this class, want to learn the parts which are also useful for deep learning. So we focus on differentiable losses and models. But we will also cover the other parts briefly.

Remark 3.4. In deep learning, we may also want to assume that the test data is drawn from a different distribution. This is not mathematically well-defined (yet). But the idea for the performance of the model to degrade gracefully as the data is changed. This is called *out-of-distribution* generalization. Currently, we can have two deep learning models with very similar test and training accuracy, but one can perform much better on OOD data. Very little is known about this, although it is an active area of study. E.g. Out-of-distribution generalization and adaptation in natural and artificial intelligence `https://nips.cc/Conferences/2021/Schedule?showEvent=21852`

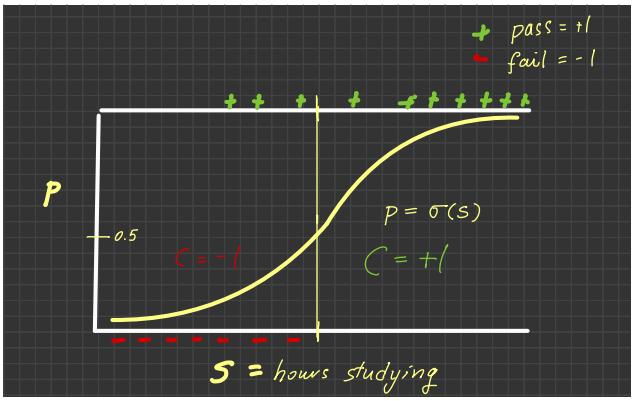

FIGURE 3. Illustration of logistic classifier: probability of passing ($y = +1$) an exam, as a function of hours studying. $p = \sigma(s)$. From wikipedia `https://en.wikipedia.org/wiki/Logistic_regression`. The model is an approximation of the probability of positive class, as a function of s.

4. Binary classification Losses

4.1. **Differentiable losses.** The approach to (supervised) binary classification we take is score based, differentiable loss. The main advantage of this approach

- a differentiable loss is amenable to optimization
- Using the score-based perspective, we can bound the error in terms of the loss
- We get better generalization using a loss like this

Remark 4.1. We already interpreted the gain for decision trees, as the KL-divergence loss. This will be an important loss for classification in general.

4.2. **Standard log-logistic loss.** In this section we study how to classify using the standard log-logistic loss. The classifier is given by (2). The loss is defined for $s \in \mathbb{R}$ by

$$\ell_{\log}(s, y) = \begin{cases} -\log(\sigma(s)) & y = +1 \\ -\log(1 - \sigma(s)) & y = -1 \end{cases}$$

For a function $s : \mathbb{R}^d \to \mathbb{R}$, and a dataset (S_m), define the empirical loss,

$$\widehat{L}_{\log}(s) = \frac{1}{m} \sum_{i=1}^{m} \ell_{\log}(s(x_i), y_i)$$

4.3. **Standard margin loss.** In this section we study the standard margin (or hinge) loss.

For $s \in \mathbb{R}$, use the classifier (2). For $s \in \mathbb{R}, y \in \mathcal{Y}_{\pm}$, define the margin loss

$$\ell_{margin}(s, y) = \max(0, 1 - ys) \tag{1}$$

This loss is designed to score which lead to incorrect classification, as well as marginal scores. See Figure 4. See also Figure 6

For a function $s : \mathbb{R}^d \to \mathbb{R}$, and a dataset (S_m), define the empirical loss,

$$\widehat{L}_{margin}(s) = \frac{1}{m} \sum_{i=1}^{m} \ell_{margin}(s(x_i), y_i) \tag{ELM}$$

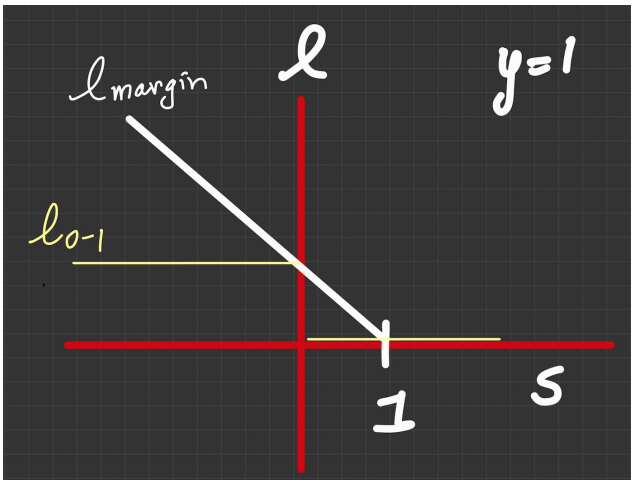

FIGURE 4. Margin loss, this loss is differentiable except at the corner, and lies above the 0-1 loss

4.4. **General classification losses.** We need to use a *surrogate loss*, define over real-valued hypotheses.

Definition 4.2 (Surrogate loss)**.** classification losses will be defined on the pair (s, c) for $s \in \mathbb{R}, c \in \mathcal{Y}$.

$$\ell_{class} : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}^+ \tag{ℓ_{class}}$$

Given $s \in \mathbb{R}$, define

$$c_{\text{sgn}} : \mathbb{R} \to \mathcal{Y}_{\pm}, \qquad c_{\text{sgn}}(s) = \text{sgn}(s) \tag{2}$$

to be the sign of s (undefined when $s = 0$).

Usually we have the additional requirement that the loss is convex, and monotone, according to the following definition.

Definition 4.3 (convex classification loss)**.** The loss, (ℓ_{class}) is convex if $\ell_{class}(s, y)$ is convex as a function of s for every $y \in \mathcal{Y}_{\pm}$. The loss is monotone if ℓ_{class} is monotone as a function of s for each $y \in \mathcal{Y}_{\pm}$.

Define the empirical loss of the score based model, s, by

$$\widehat{L}_{class}(s) = \frac{1}{m} \sum_{i=1}^{m} \ell_{class}(s(x_i), y_i) \tag{EL-C}$$

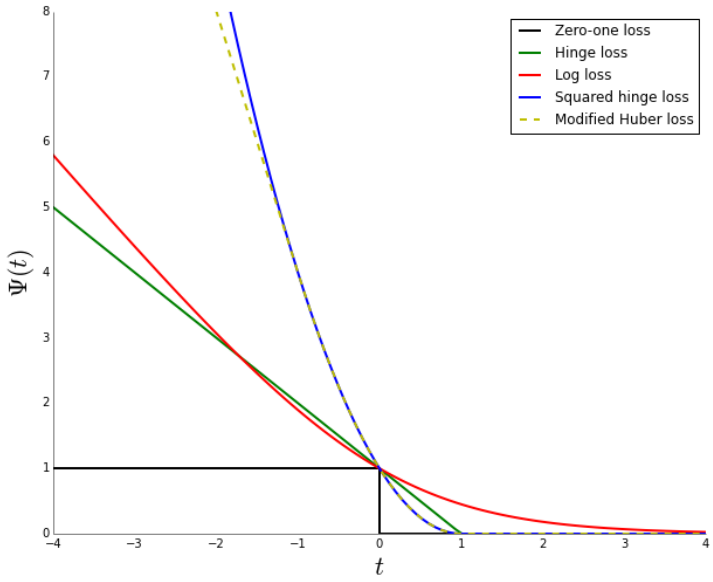

FIGURE 5. Plot of loss functions, image takes from `https://fa.bianp.net/blog/2014/surrogate-loss-functions-in-machine-learning/`

Binary Classification via loss minimization (score-based)

Inputs:

- Dataset (S_m) with binary labels $y_i \in \mathcal{Y}$ and $x_i \in \mathbb{R}^d$ features.
- Hypothesis class of models $h_w(x) : \mathbb{R}^d \to \mathbb{R}$, parameterized by w.
- Classifier: $c : \mathbb{R} \to \mathcal{Y}_{\pm}$, maps scores $h(x)$ to classes $y = c(h(x)) \in \mathcal{Y}$
- Surrogate classification loss, (ℓ_{class}).

Goal:

- Given x, predict $y = c(x)$, in other words, minimize the test error (EE) of the classifier $c(h_{w^*})$.

Method:

- Minimize (EL-C) to find the model which minimizes the loss, h_{w^*}.
- Set $y = c(h_{w^*}(x))$.

5. Error bounds from the loss

We defined the classification task to be (in-distribution) generalization. For this purpose, both the losses work equally well. So does any abstract loss which satisfies the properties above.

5.1. **Errors from losses.** When we use the score-based approach, we need to check that our loss minimization (which is defined $h \in \mathbb{R}$) results in an effective *classification*. In other words we care about the average classification error (the 0-1 loss, defined below).

We need to characterize the effect of the loss on the errors. There are two approaches to this question: (i) we can prove *a priori* the properties of the classifier, or (ii) we can simply check

a posteriori that the results are good. The purpose of understanding classification losses is to achieve the first goal.

One possible answer is given by the following idea.

Definition 5.1. Given the triple $(\ell_{class}, c, C_{class})$ consisting of a classification loss, a classification map and a constant $C_{class} > 0$. The triplet is an upper bound for the classification error, if

$$\ell_{class}(h, y) \geq C_{class}\ \ell_{0-1}(c(h), y), \quad \text{for all } h \in \mathbb{R},\ y \in \mathcal{Y}_{\pm} \tag{LvE}$$

We say the loss is an upper bound for the error when the constant and the classifier are understood. Note: if (LvE) holds for C_{class}, and $C_{class} \geq C$, then it also holds for C. Define the best C in (LvE) to be the largest constant for which (LvE) holds.

Theorem 5.2. *Suppose $(\ell_{class}, c, C_{class})$ is an upper bound for the error. Show that for any function $h : \mathcal{X} \to \mathbb{R}$, and any dataset S_m*

$$\widehat{L}_{0-1}(c(h)) \leq \frac{1}{C_{class}} \widehat{L}_{class}(h)$$

In particular, the bound above holds for a minimizer h_{w^*} of $\widehat{L}_{class}(h)$.

We summarize the application of the error bounds as follows.

Error bounds on the minimizer

Inputs:

- Model h_{w^*} which minimizes the empirical loss (EL-C).
- Classifier: $c : \mathbb{R} \to \mathcal{Y}_{\pm}$, a rule which converts the model value to a class.
- Classification loss ℓ_{class}, which (along with the constant C_{class}) is an upper bound for the error

Outputs:

- The empirical error, (EE), is bounded by $\frac{1}{C_{class}} \widehat{L}_{class}(h)$

Exercise 5.1. *Prove Theorem 5.2.*

Exercise 5.2. *In this exercise, use c_{sgn}. (i) Is the loss $\ell(h, y) = (h - y)^2$ an upper bound for the zero one loss? If so, what is the best (largest) constant for which* (LvE) *holds.*

(ii) Show that $\ell(h, y) = |h + y|$ is not an upper bound for the zero one loss.

(iii) Given the function $\ell(h, y)$, suppose there is an $h < 0$ with $\ell(h, 1) = 0$. Show this function cannot be an upper bound for the zero one loss.

(iv) Converse. Given $\ell(h, y)$, suppose (1) $\ell(h, y) \geq 0$ for all h, y, (2) there is some $c > 0$ such that $\ell(h, y) \geq c$ for all h with $(h) \neq y$. Prove that there is a C_{class} which makes ℓ and upper bound for the zero one loss. What is the best value of C_{class}?

5.2. **Error analysis for the standard margin loss.** We have the following result, which follows from Theorem 5.2. Given any function $h : \mathcal{X} \to \mathcal{Y}_{\pm}$, and any dataset S_m

$$\widehat{L}_{margin}(s) \geq \widehat{L}_{0-1}(c_{\text{sgn}}(s)) \tag{3}$$

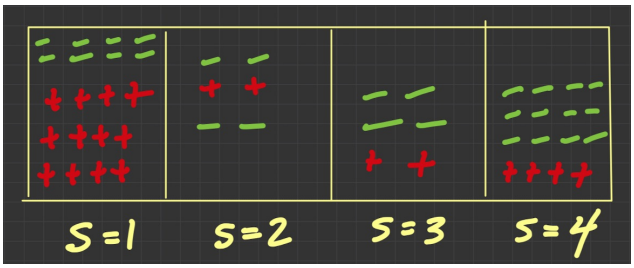

FIGURE 6. Score based classification example

Error bounds for margin classification

Inputs:

- Model h_w which minimizes the empirical loss (ELM).

Outputs:

- The empirical error, (EE), is bounded by the empirical loss $\widehat{L}_{margin}(h_w)$

5.3. **Error bounds for the log-sigma loss.**

Definition 5.3. Define the the scaled loss by

$$\ell_{\log -2}(s,y) = -\frac{1}{\log 2}\ell_{\log}(s,y)$$

Then

$$\ell_{log-2}(s,y) < 1 \implies c(s) = y \tag{4}$$
$$\ell_{log-2}(s,y) > 1 \implies c(s) \neq y$$

The proof of this is an exercise.

In particular, Theorem 5.2 holds for this loss.

5.4. **Exercises.**

Exercise 5.3. *Prove* (4).

Exercise 5.4. *Consider the example of score-based classification illustrated in Figure 6. Find the minimizer of the empirical loss using the score-based absolute value loss*

$$\ell_{abs}(s,y) = \begin{cases} \max(s,0) & y = -1 \\ \max(-s,0) & y = +1 \end{cases}$$

the threshold model $s_w(x) = x - w$, and the sign classifier $c(s) = \operatorname{sgn}(s)$. Compare to the majority classifier which chooses the most popular class in each bin. Show that in Figure 6, if we relabel the scores from $1,2,3,4$ to any other non-decreasing values (e.g. try $10,15,20,25$), and use the absolute value loss, we get the same classifier. (Hint: can check this directly or use the condition for a minimizer).

Exercise 5.5. *Show that with the margin loss* (1)*, the cases in* **??** *correspond to*

$$\ell_{margin}(s,y) \begin{cases} [1,\infty) & \textit{incorrect} \\ \in [0,1] & \textit{marginal} \\ = 0 & \textit{confident} \end{cases}$$

Exercise 5.6. *Show that* (LvE) *holds for the* $\ell_{margin-t}$ *with* $C_{class} = 1$ *and the* $c = \operatorname{sgn}$ *classifier. Justify* (3)*.*

Definition 5.4. Given a threshold $t > 0$. Define the t-margin loss,

$$\ell_{margin,t}(s,y) = \begin{cases} \max(0, 1 - s/t) & y = 1 \\ \max(0, 1 + s/t) & y = -1 \end{cases} \tag{5}$$

Exercise 5.7. *(i) Show that setting* $t = 1$ *in* (5) *recovers that standard margin loss. (ii) Generalize the definitions of the error types* **??***.*

Exercise 5.8. *Plot the loss* (5) *for* $y = 1$ *and* $t > 1$*. Show symmetry of loss* $\ell_{margin,t}(-s,-y) = \ell(s,y)$*. Use this to plot loss for* $y = -1$*.*

References

[Mur12] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.
[Mur22] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2022.