MATH 462 LECTURE NOTES

ADAM M. OBERMAN

CONTENTS

1. Introduction to classification losses 3
1.1. Introduction to classification 3
1.2. Discussion on classification losses 4
1.3. Comparison with regression 6
2. Probability based classifiers 7
2.1. Zero-one loss classifier 7
2.2. False positives and False Negatives 10
2.3. Classes from probabilities 10
2.4. Learning class probabilities 12
2.5. Losses for classification with probabilities 13
3. Score based losses 15
3.1. Define a score function 15
3.2. Classes from score functions 17
3.3. Score-based loss 18
3.4. Margin-based Losses 20

Date: September 22, 2021.

3.5. Binary classification, margin (hinge) loss 21
4. Discussion of classification losses 23
5. Multiclass case 23
5.1. Multiclass classification using cosine similarity 24
6. Smooth classification losses 26
7. Linear models 26
References 27

1. Introduction to classification losses

1.1. **Introduction to classification.** Let's begin our discussion of the classification problem by comparing it to the regression problem.

The main difference is that we are learning one of K discrete classes, which we simply represent as one of the integers, $1,\dots,K$. In order to keep notation similar to the previous case, we write

$$y \in \mathscr{Y} = \mathscr{Y}_K = \{1,\dots,K\}. \tag{K}$$

The notation $\mathscr{Y}_K$ will be used to emphasize that this is a classification problem with K classes. There are conceptual differences depending on the number of classes involved. Two classes is called *binary classification*, more than two is called multi-classification. In fact we can consider the following:

- Binary classification for two classes, where the classes are balanced (e.g. identify a cat/dog picture). We write $\mathscr{Y}_2 = \{1,2\}$ as in (K), above.
- When there are more than two classes, we will usually consider the case where the classes are balanced (same number of samples per class), e.g. MNIST dataset of 10 digits, or CIFAR10/CIFAR100, with 10 and 100 classes, respectively.

- We also consider the case of binary classification consisting of a test, which can be positive of negative, which we can write as $\mathscr{Y}_{\pm} = \{-1, +1\}$. In this case, the probability of each class may be different (e.g. a medical test for a rare medical condition). In this case we care about the *error types*, False Positives or False Negatives.

There is an added complication that our classification losses will be defined on the pair (s, c) for $s \in \mathbb{R}, c \in \mathscr{Y}$.

$$\ell_{class} : \mathbb{R} \times \mathscr{Y} \to \mathbb{R}^+$$

1.2. **Discussion on classification losses.** In this section we are studying classification losses. Most textbooks present the classification losses as already fixed, they don't derive them. The main approaches to (supervised) binary classification are

(DB) Model the Distance to the linear classification Boundary (using a specific notion of distance), usually using Support Vector Machines, [DFO20]. In this case, we combine a margin loss (which we will see later) with a linear model. The loss minizimization problem (EL-C) is piecewise differentiable and convex, it corresponds to a linear program.

(CP) Model class probabilities. In this case, we convert a linear score to a probability, using the logisitic function `https://en.wikipedia.org/wiki/Logistic_function`. The apply a loss on the probability to match the class probabilities for the score. This latter approach has the advatange that the problem (EL-C) is smoothly differentiable.

Both these problems are usually presented in the two class case, and then generalized to multiple classes.

We will see that we can relate the two problems in the following sense: we can show that

(1) The loss in (CP) can be interpreted as a smoothed out margin loss, relating it to (DB)
(2) A version of the SVM model is invariant to relabeling the scores, so we can *calibrate* the scores to be probabilities. In other words we find a non-decreasing map $p(s)$ which gives the class probability as a function of the score. This shows that (DB) can be interpreted as finding class probabilities.

Going from the two class case to the multiclass case introduces an extra dimension to the problem: the number of classes. The multiclass case also allows us to impose extra structure on the loss. In particular, the multiclass version of (DB) and (CP)

impose unusual (and different) notion of distances on the score: (DB) is an infinity norm, and (CP) uses the KL-divergence (a non-symmetric notion of distances on probabilities).

Another approach to the the multiclass case uses the 2-norm, which will be discussed in subsection 5.1.

1.3. **Comparison with regression.** Our approach to classification is *score-based*, which means there are a number of similarities to the case of regression.[1]

At the abstract level we will still have:

- A labelled dataset (S_m) with pairs of (x_i, y_i), $i = 1, \ldots, m$, of data (or features), x_i and labels, y_i.

$$(S_m) \qquad S_m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$$

- A classification loss $\ell(h, y)$, which measures how far our hypothesis is from being correct. It should be (piece-wise) differentiable as a function of h.
- Hypotheses consisting of a family of linear models, $h_w(x)$.

[1] there are other classification methods, for example probabilistic, or clustering, see the discussion which follows. See also `https://en.wikipedia.org/wiki/Statistical_classification`

- We will still train our model to fit data by minimizing the expected loss (EL-C)

$$\widehat{L}(w) = \frac{1}{m}\sum_{i=1}^{m} \ell(h_w(x_i), y_i) \tag{EL-C}$$

These similarities mean that our approach of minimizing the loss (and the gradient based algorithms we will study to find the minimizers) will have many similarities to the case of regression.

However, there are also a number of differences which make the problem more subtle. The first difference is that we need a way to convert our linear model into one of a number of discrete classes. We will study each of the cases (Binary and multi-classification) from this perspective.

2. Probability based classifiers

2.1. **Zero-one loss classifier.** The simplest classification loss is the 0-1 loss, given by

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

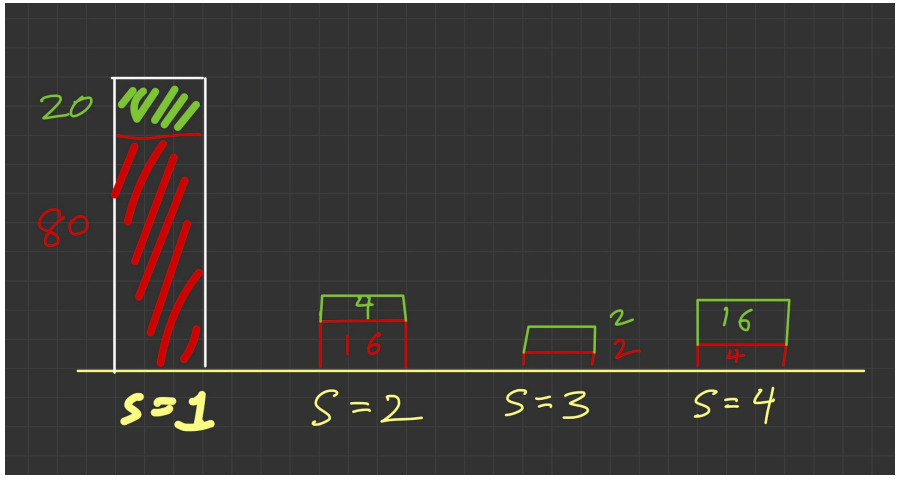

FIGURE 1. Illustration of classification problem

Example 2.1. Consider the example of Figure 1. In this example, the minmizer of the zero one loss is given by majority rule for each score.

$$c(s) = \begin{cases} 0 & s = 1,2 \\ 1, & s = 3,4 \end{cases}$$

The the zero-one loss error count is 30, with $FN = 24$ and $FP = 6$

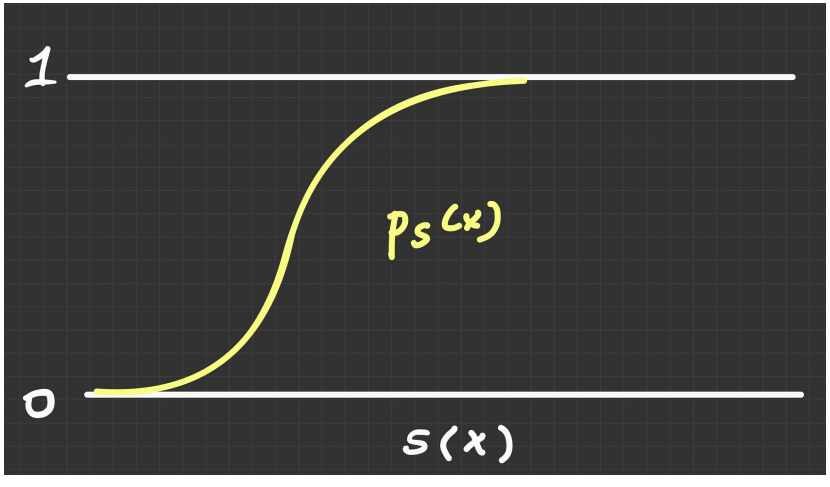

FIGURE 2. Illustration of a score function

However, this loss is not amenable to optimization using gradient based methods, because, if we consider minimizing the loss (EL-C), $\ell_{0-1}(h,y)$ is not even defined as a function of h.

On the other hand, there is a way to define the minimizer of the $0-1$ loss directly (it's just not that useful for training using linear models).

2.2. **False positives and False Negatives.** In the binary classification case, we can define error types. A False Positive (FP) is the error that occurs when $c = 1$ and $y = -1$. Similarly False Negative corresponds to $c = -1$, $y = +1$.

In some cases, we want to design a loss which reflects the relative cost of different types of errors. Typically a False Positve is worse than a False Negative. (E.g when testing for a disease).

In this case the loss function can be extended to measure the cost of different error types. Define C_{FP}, C_{FN} constants, (e.g. $C_{FP} = 10, C_{FN} = 1$).

Then can define the more loss

$$\ell_{FP}(c,y) = \begin{cases} C_{FP} & c = 1, y = -1 \\ C_{FN} & c = -1, y = 1 \\ 0 & \text{otherwise} \end{cases}$$

and we can consider minimization of (1) with this loss.

2.3. **Classes from probabilities.** An important object is the probability of the positive class

$$p(x) = \text{Prob}(y = 1 \mid x) \tag{CP}$$

then the zero-one loss minimization is easily solved.

Theorem 2.2. *The classification problem* (EL-C) *with the zero one loss and the class probability $p(x)$ becomes*

$$\min_{w} \frac{1}{m} \sum_{i=1}^{m} \ell_{0,1}(c(x_i), y_i) \tag{1}$$

The optimal classifier is

$$c_{Bayes}(x) = \begin{cases} 1 & p(x) \geq .5 \\ 0 & ow \end{cases}$$

Proof. We can solve this problem for the optimal value, if we have access to the class probability function $p(y = 1 \mid x)$. In this case, the optimal solution is given the Bayes classifier

□

Exercise 2.3. *Prove the statment above.*

Exercise 2.4. *Generalize the theorem above to the FP loss.*

2.4. **Learning class probabilities.** If the class probabilities (CP) are available, then the loss minimization problem becomes easy to solve. In this section we study how to learn the class probabilities using a linear model.

First, we need a function which converts number $x \in \mathbb{R}$ to probabilies $p(x) \in [0,1]$. In this context, the number are called *logits*.

The function is called the logistic function

$$\sigma(x) = \frac{1}{1+\exp(-x)}$$

Plot looks like Figure 2.

2.4.1. *Properties of the logistic function.* These logistic function has an inverse which maps probabilities to numbers (logits)

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$$

Here write $f(x) = \sigma(x)$.

- $2f(x) = 1 + tanh(x/2)$
- $f(x) = \frac{\exp x}{1+\exp x}$

- $1 - f(x) = f(-x)$ (so $f(x) - 1/2$ is an odd function)
- $f'(x) = f(x)(1 - f(x))$

The ratio $\frac{p}{1-p}$ is called the *odds ratio*.

2.5. **Losses for classification with probabilities.** Now we want to learn $p(x)$. In this case it makes sense to think of x as a random variable (or bin of data) which has a probability $p(x)$ of being a positive example (CP).

In this case, we are given samples (S_m). Think of binning the samples into all those which have x in the same bin. Now we wish to learn $p(x)$

We can do so by simply counting the population fraction in the bin. Let this fraction be q. We can define a loss:

$$\ell_{\text{Prob}}(p, y)$$

which should have the property that: if the fraction of samples $y = 1$ is q, then the loss is non-negative and zero iff $p = q$.

So we can take

$$\ell_{\text{Prob}}(p, y) = \ell_2(p, y) = \frac{1}{2}(p - y)^2$$

or

$$\ell_{\text{Prob}}(p,y) = \ell_{\log}(p,y) = \begin{cases} \log(p) & y = 1 \\ \log(1-p) & y = -1 \end{cases}$$

A loss which will not work is $|p-y|$

Exercise 2.5. *VErify these statements.*

$$\min_p q(1-p)^2 + (1-q)(p)^2$$

gives $p = q$
and

$$\min_p q \log p + (1-q)\log(1-p)$$

gives $p = q$
But

$$\min_{p\in[0,1]} q(1-p) + (1-q)p$$

has gives $p = 0$ *or* $p = 1$ *as minmimizer*

Now consider a loss which has the same shape as the hinge, but is strongly convex, and smooth.

$$\ell_{sc}(s,y) = -\log\left(\frac{1}{1+\exp(-x)}\right)$$

3. SCORE BASED LOSSES

In this section, we go to the primary focus, which is score-based losses. Later we will consider linear models to obtain a score from features. But first we formalize our intuition about what we mean by a score.

3.1. **Define a score function.** Intuitively, a score function (for a given class) means that a higher score corresponds to a higher likelihood of class membership. We formalize this notion with the following definition.

Define the probability density of a function $s(x)$ for the class k to be

$$p_s(t) = Prob(y = k \mid s(x) = t)$$

Definition 3.1. We say $s(x)$ is a score function for the class k if $p_s(t)$ is a non-decreasing function. We say the score function $s(x)$ is *separatating* if $p_s(t)$ only takes the values $0, 1$.

Given a score function, and a threshold, w, define the threshold model and threshold classifier, repspectively, by

$$h_w(s) = w - s \qquad c_w(s) = \mathrm{sgn}(h_w(s)) \tag{2}$$

Exercise 3.2. *In the binary classification case $Y_{\pm}$, show that the 0-1 loss with the threshold classifier* (2) *becomes a step function*

$$\ell_{0-1}(c_w(s), y) = 1_{\{\mathrm{sgn}(w-s)=y\}}$$

Exercise 3.3 (check definition)**.** *Show that if s is a separating score function, the, either (i) there is exact one w for which the threshold classifier has zero error, or (ii) there is half-closed interval I (check endpoints) for which c_w has zero error, for all $w \in I$. Here the error means the 0-1 loss.*

A more typical score function looks like Figure 2.

Exercise 3.4 (Relate score classifier to Bayes classifier)**.** *For a score-based classifier, choosing a threshold which is the solution of*

$$p_s(x) = .5$$

results in the Bayes classifier, defined above.

Prove the statement above. Hint: every lower score has a probabilty of less than .5, and similarly for the higher score.

3.2. **Classes from score functions.** Although eventually we will need to study how to *learn* score functions, first we need to study how to (best) convert scores into classes. These are two difference problems because:

- we can have very effective scoring function, which means high scores are more likely to be in the class
- however, he best way to classify using the score will also depend on the distribution of classes (as well as our preference for error types).

There is more than one way to define a classification based on scores.

Example 3.5 (Grading). Consider the classification problem of converting a grade, $x \in [0,100]$ in one of $K = 5$ letter grades F, D, B, C, A. We can use an absolute rule, e.g. $x \in [85,100]$ converts to A, or we can grade on the curve: For example, if there are five grades in a particular university course, A, B, C, D, and F, where A is reserved for the top 20% of students, B for the next 30%, C for the next 3040%, and D or F for the remaining 10 to 20%.

In each case, the outcomes are difference and there are arguments for and against each method. For example, if a class is particularly strong compared to other classes, the students are penalized by grading on a curve.

In what follows, we will define the classifier based minimizing (EL-C) using a choice of loss function (and classifier). We will then need to study loss design: how the choice of loss affects the solutions of (EL-C).

3.3. **Score-based loss.** Now we want to define a score-based loss which is piecewise differentiable as a function of s.

Define the absolute error loss $\ell_{abs}(s,y)$ as

(LAC)
$$\ell_{abs}(s,y) = \begin{cases} 0 & \text{sgn}(s) = y \\ |y-s| & ow \end{cases}$$

Example 3.6. Consider the example of Figure 1. Find the minimizer of (EL-C) with the score-based threshold classifier (2), and the absolute error loss (LAC). Show that

it corresponds any choice of w between 1 to 2. (TODO check endpoints), and

$$c(s) = \begin{cases} 0 & s = 1 \\ 1, & s = 2,3,4 \end{cases}$$

Note, this is different from the Bayes classifier.

Exercise 3.7. *Show that in Figure 1, if we relabel the scores from* $1,2,3,4$ *to any other non-decreasing values (e.g. try* $10,15,20,25$*), we get the same classifier. (Hint: can check this directly or use the condition for a minimizer).*

Exercise 3.8. *Relate the problelm* (EL-C) *with the score-based threshold classifier* (2)*, and the absolute error loss* (LAC) *to the central value problem with the absolute value loss (from the regression chapter). Explain!*

This loss is amenable to optimization.

Theorem 3.9. *Consider* (EL-C) *with the score-based threshold classifier* (2)*, and the absolute error loss* (LAC)*. A sufficient condition for a minimizer* w^* *is that the number of false positives is equal to the number of false negatives.*

Proof. Now consider mimizing (EL-C) with this loss

We get $\pm$ on each error, depending on if PF / FN
[[details in handwritten class notes, to be filled in]]

$$\sum_{FP} 1 = \sum_{FN} 1$$

So the w^* is the threshold which makes $FP = FN$. □

More generally, we can choose the ratio of false positives to false negatives using the the following generalization of the absolute error loss

(LAC-FP)
$$\ell_{abs}(s,y) = \begin{cases} 0 & \operatorname{sgn}(s) = y \\ |y-s| & y = 1, \operatorname{sgn}(s) = -1 \\ C|y-s| & y = -1, \operatorname{sgn}(s) = +1 \end{cases}$$

Exercise 3.10. *Generalize Theorem 3.9 to the case of* (LAC-FP). *Find the value of C which leads to $FP = 10FN$*

3.4. **Margin-based Losses.** We say an interesting property of the absolute error loss: an increasing relabelling the scores did not change the result. This suggests that the loss may not encourage a wider margin. Later, when we are learnig scores,

we want to encourage scores which better separate classes. One approach to this problme is to design a loss with a *margin*

3.5. **Binary classification, margin (hinge) loss.** The idea is to have a penalty for correct classifications, if the distance to the classification boundary is less than one (before, in the absolute loss, it was zero). See Figure 3, for an illustration.

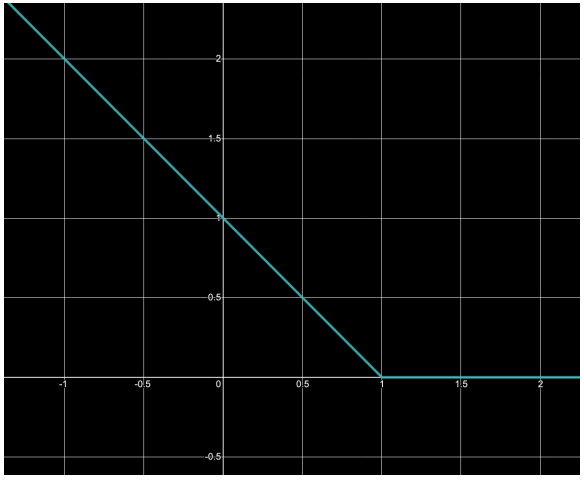

FIGURE 3. Hinge loss, this loss is differentiable except at corner, and lies above the 0-1 loss

So define for $y \in \mathscr{Y}_{\pm}, s \in \mathbb{R}$,

$$g(s,y) = sy$$

Note that $g(s,y)$ is the signed distance to the classification boundary: correct if $g > 0$, incorrect if $g < 0$.

Next define the hinge loss

$$\ell_{hinge}(s,y) = \max(1 - g(s,y), 0) = (1 - g(s,y))^{+}$$

Note

$$\partial_s \ell_{hinge}(s,y) = \begin{cases} +1 & y = 1, s < 1 \\ -1 & y = -1, s > -1 \end{cases}$$

Note, also that,

$$\ell_{hinge}(s,y) \geq \ell_{0-1}(s,y)$$

with equality when $g \geq 1$ and when $g = 0$.

4. Discussion of classification losses

So far we have studied losses for classication

- zero-one loss
- absolute distance loss
- margin loss

Each of these assign a different interpretation to the scores

- zero-one loss: scores have an identity, but are not comparable
- absolute distance loss: scores are ordered, but have no scale
- margin loss: scores are ordered. There is a scale for scores within distance one of the boundary, but nothing beyond. (The loss sees no difference between 1 from boundary and 3 from boundary). This is because the loss is flat (zero) away from the boundary.

Next we will look at losses that are strongly convex - so they continue to give (diminishing) returns the further you go from the boundary.

5. Multiclass case

5.1. **Multiclass classification using cosine similarity.** Another approach to the the multiclass case uses the 2-norm. While is is more naturally geometrically, it is less common. The similarity approach which measures the angle between two feature (score) vectors, is used in Face Recognition and Image Search. The FR problem corresponds to a binary classification: determine if two images represent the same face (for a face/images never seen before). The Image Search problem is: given an image, and a database of images (e.g. the internet), find similar (looking) images. Both compare the cosine similarity

$$\text{sim}(x_1, x_2) = \frac{s_1 \cdot s_2}{\|s_1\|_2 \|s_2\|_2}$$

When the vectors are close to unit length, this similarity is comparable to

$$\text{sim}(x_1, x_2) = 1 - \frac{1}{2}\|x_1 - x_2\|^2, \quad \text{when } \|s_1\| = \|s_2\| = 1$$

(just expand the squared term).

Now we define a classifier by the highest score. Let's assume we have score functions for each classs.

For $s \in \mathbb{R}^K$, define $c : \mathbb{R}^K \to \mathscr{Y}_k$ by

$$c(s) = \arg\max s_y$$

Define the gap (or margin) in the multi-class case by

$$g(s,y) = s_y - \max_{j \neq y} s_j$$

so this is positive if the model is correct, and negative otherwise. As in the binary case, we can regard g as the signed distance (in the maximum norm) to teh classification boundary.

Now we can define, as in the binary case,

$$\ell_{abs}(s,y) = \max(-g(s,y), 0)$$

which is a penalty for the distance to the classification boundary (measured in the maximum norm).

Likewise, if we want to encourage a margin, we define

$$\ell_{hinge}(s,y) = \max(1 - g(s,y), 0) = (1 - g(s,y))^+$$

which penalizes correct points within distance one of the classification boundary.

Exercise 5.1. *Show the multi-class case with $K = 2$ case reduces the binary case, when we set*

$$(s_1, s_2) = (s/2, -s/2)$$

where LHS is the multiclass score function when $K = 2$, and s is the binary score function.

6. Smooth classification losses

We can define smooth classification losses, which are smooth approximations of the margin loss.

7. Linear models

However, in general data is multi-dimensional, and what we want to learn is a consistent score function.

We will do this, in the multi-class case, by learning a score function for each class.

Goal: learn $h(x; w_y)$ a score function for each class. (I.e. w_y is matrix, one row for each y, each row is a weight vector) (Note the features share lots of information, but the final classifier is one for each class) So K functions, one for each class, and K vectors w_y.

$$h_W(x) = (w_1 \cdot x, \ldots, w_K \cdot x)$$

Result / Exercise:

Compute $\nabla_{w_i} \ell_{hinge}(h_W(x), y)$

Complicated: $\ell' = -1$ if $g(s,y) < 1$ and 0 ow. Gradient: x_y for w_y and $-x_j$ for the maximum component. and 0 ow.

REFERENCES

[DFO20] Marc Peter Deisenroth, A Aldo Faisal, and Cheng Soon Ong. *Mathematics for machine learning*. Cambridge University Press, 2020.