Machine learning

Multi-class Classifiers

Hamid Beigy

Sharif University of Technology

April 16, 2023

Table of contents

Introduction

1. In classification, the goal is to find a mapping from inputs X to outputs $t \in \{1, 2, \ldots, C\}$ given a labeled set of input-output pairs.
2. We can extend the binary classifiers to C class classification problems or use multiple binary classifiers.
3. For C-class, we have four extensions for using binary classifiers.

   Single $C-$class discriminant
   One-against-all
   One-against-one
   Hierarchical classification
   Error correcting coding

$C-$class discriminant function

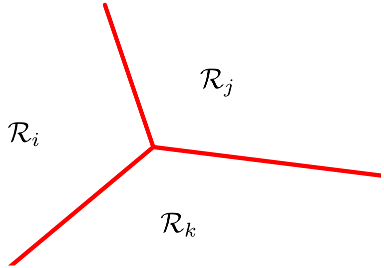
1. We can consider a single $C-$class discriminant comprising C linear functions of the form

$$g_k(x) = w_k^T x + w_{k0}$$

2. Then assigning a point x to class C_k if $g_k(x) > g_j(x)$ for all $j \neq k$.
3. The decision boundary between class C_k and class C_j is given by $g_k(x) = g_j(x)$ and corresponds to hyperplane

$$(w_k - w_j)^T x + (w_{k0} - w_{j0}) = 0$$

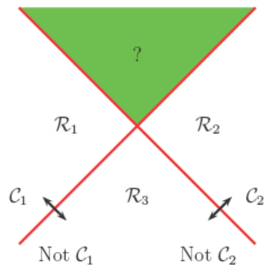
4. This has the same form as decision boundary for the two-class case.



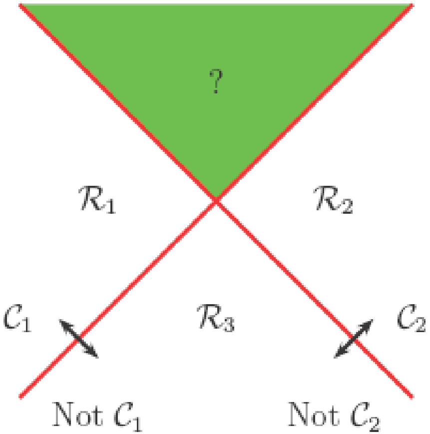
One-against-all classification

1. This extension is to consider a set of C two-class problems.
2. For each class, we seek to design an optimal discriminant function, $g_i(x)$ (for $i = 1, 2, \ldots, C$) so that $g_i(x) > g_j(x)$, $\forall j \neq i$, if $x \in C_i$.
3. Adopting the SVM methodology, we can design the discriminant functions so that $g_i(x) = 0$ to be the optimal hyperplane separating class C_i from all the others. Thus, each classifier is designed to give $g_i(x) > 0$ for $x \in C_i$ and $g_i(x) < 0$ otherwise.
4. Classification is then achieved according to the following rule:

$$\text{Assign } x \text{ to class } C_i \text{ if} \quad i = \underset{k}{argmax}\, g_k(x)$$



1. The number of classifiers equals to C.
2. Each binary classifier deals with a rather asymmetric problem in the sense that training is carried out with many more negative than positive examples. This becomes more serious when the number of classes is relatively large.
3. This technique, however,may lead to indeterminate regions, where more than one $g_i(x)$ is positive



1. The implementation of OVA is easy.
2. It is not robust to errors of classifiers. If a classifier make a mistake, it is possible that the entire prediction is errorneous.

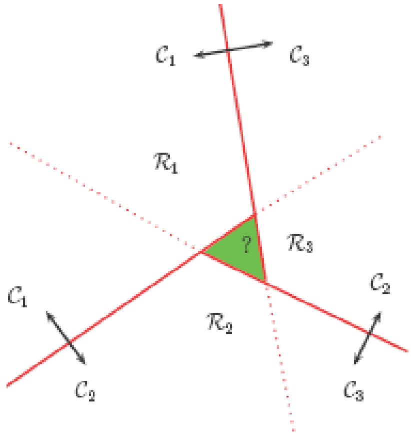
Theorem (OVA error bound)

Suppose the average binary error of C binary classifiers is ϵ. Then the error rate of the OVA multi–class classifier is at most $(C-1)\epsilon$.

3. Please prove the above theorem.

One-against-one classification

1. In this case, $C(C-1)/2$ binary classifiers are trained and each classifier separates a pair of classes.
2. The decision is made on the basis of a majority vote.



3. The obvious disadvantage of the technique is that a relatively large number of binary classifiers has to be trained.

1. This technique, however,may lead to indeterminate regions, where more than one $g_{ij}(x)$ is positive

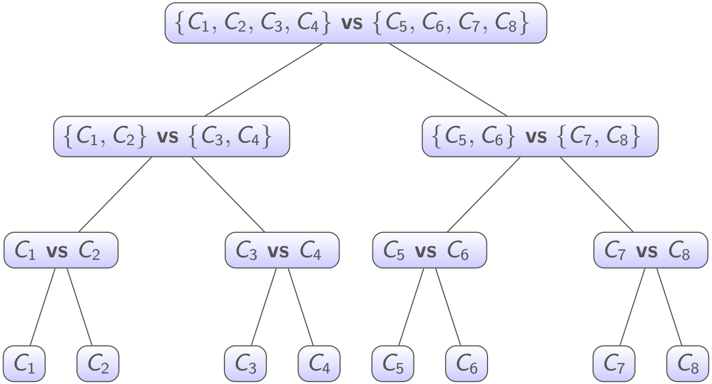
Theorem (AVA error bound)

Suppose the average binary error of the $C(C-1)/2$ binary classifiers is at most ϵ. Then the error rate of the AVA multi–class classifier is at most $2(C-1)\epsilon$.

2. Please prove the above theorem.
3. The bound for AVA is $2(C-1)\epsilon$ and the bound for OVA is $(C-1)\epsilon$. Does this mean that OVA is neccessarily better than AVA? Why or why not? Please do it as a homework.

Hierarchical classification

1. In hierarchical classification, the output space is hierarchically divided i.e. the classes are arranged into a tree.



1. One thing to keep in mind with hierarchical classifiers is that you have control over how the tree is defined.
2. In OVA and AVA you have no control in the way that classification problems are created.
3. In hierarchical classifiers, the only thing that matters is that, at the root, half of the classes are considered positive and half are considered negative.
4. You want to split the classes in such a way that this classification decision is as easy as possible.

Theorem (Hierarchical classification error bound)

Suppose the average binary classifiers error is ϵ. *Then the error rate of the hierarchical classifier is at most* $\lceil \log_2 C \rceil \epsilon$.

5. Can you do better than $\lceil \log_2 C \rceil \epsilon$? Yes. Using error-correcting codes.

Error correcting coding classification

1. In this approach, the classification task is treated in the context of error correcting coding.
2. For a $C-$class problem a number of, say, L binary classifiers are used,where L is appropriately chosen by the designer.
3. Each class is now represented by a binary code word of length L.
4. During training of i^{th} classifier, the desired labels are chosen from $\{-1, +1\}$.
5. For each class, the desired labels may be different for the various classifiers.
6. This is equivalent to constructing a matrix $C \times L$ of desired labels. For example, if $C = 4$ and $L = 6$, such a matrix can be

$$\begin{bmatrix} -1 & -1 & -1 & +1 & -1 & +1 \\ +1 & -1 & +1 & +1 & -1 & -1 \\ +1 & +1 & -1 & -1 & -1 & +1 \\ -1 & -1 & +1 & -1 & +1 & +1 \end{bmatrix}$$

1. For example, if $C = 4$ and $L = 6$, such a matrix can be

$$\begin{bmatrix} -1 & -1 & -1 & +1 & -1 & +1 \\ +1 & -1 & +1 & +1 & -1 & -1 \\ +1 & +1 & -1 & -1 & -1 & +1 \\ -1 & -1 & +1 & -1 & +1 & +1 \end{bmatrix}$$

2. During training, the first classifier (corresponding to the first column of the previous matrix) is designed in order to respond $(-1, +1, +1, -1)$ for examples of classes C_1, C_2, C_3, C_4, respectively.
3. The second classifier will be trained to respond $(-1, -1, +1, -1)$, and so on.
4. The procedure is equivalent to grouping the classes into L different pairs, and, for each pair, we train a binary classifier accordingly.
5. Each row must be distinct and corresponds to a class.

1. When an unknown pattern is presented, the output of each one of the binary classifiers is recorded, resulting in a code word.
2. Then,the Hamming distance of this code word is measured against the C code words, and the pattern is classified to the class corresponding to the smallest distance.
3. This feature is the power of this technique. If the code words are designed so that the minimum Hamming distance between any pair of them is, say, d, then a correct decision will still be reached even if the decisions of at most $\lfloor \frac{d-1}{2} \rfloor$ out of the L, classifiers are wrong.

Theorem (Error-correcting error bound)

Suppose the average binary classifiers error is ϵ. Then the error rate of the classifier created using error correcting codes is at most 2ϵ .

4. You can prove a lower bound that states that the best you could possible do is $\frac{\epsilon}{2}$.

Learning with Imbalanced Data

1. An imbalanced data set is one in which there are too many positive examples and too few negative examples (or vice versa).
2. Examples of imbalanced data set are
   - Fraud detection
   - Intrusion detection
   - Spam detection
3. Let 99.9% of samples are labeled as negative and only 0.1% of samples are labeled as positive.
   - Then, minimizing training error will result in a classifier that labels all samples as negative.
   - The error of this classifier is only 0.1%.
   - This results in a **bad classifier**.
4. The problem is not with the data, but with the definition of the learning problem.
5. If we have a good binary classification algorithm, can we use it for imbalanced dataset?
6. To use such a classifier, we use the following transformations of dataset.
   - Sub-sampling
   - Oversampling
   - Weighting

1. In this problem, we believe that the positive class is α-times as important as the negative class.
2. The learning problem is now

$$\text{Minimize} \underset{(\mathbf{x},y)\sim\mathcal{D}}{\mathbb{E}} \left[\alpha^{y=+1} \left[h(\mathbf{x}) \neq y\right]\right]$$

3. The above objective function is identical to standard binary classification.
4. The only difference is that the cost of misprediction for $y = +1$ is α, while the cost of misprediction for $y = -1$ is 1.
5. The question we will ask is:
   Suppose that having a good binary classification algorithm. Can I turn that into a good algorithm for solving the α-weighted binary classification algorithm?
6. To do this, we need to define a transformation that maps a concrete weighted problem into a concrete unweighted problem.
7. This transformation must be done both at training time and at test time.
8. **The transformation is explicitly turning the distribution over weighted examples into a distribution over binary examples**.

1. The sub-sampling down the training samples the majority class by factor of α.

```
procedure SubsampleMap(D^w, α)
    while true do
        (x, y) ~ D^w
        u ~ [0, 1]
        if y = +1 and u < 1/α then
            return (x, y)
        end if
    end while
end procedure
```

Theorem (Subsampling Optimality)

Suppose the binary classifier trained using the above algorithm achieves a binary error rate of ϵ. Then the error rate of the weighted predictor is equal to $\alpha\epsilon$.

2. This theorem states that if your binary classifier does well (on the new distribution), then the learned predictor will also do well (on the original distribution).

Proof.

1. Let $\mathcal{D}^w$ be the original distribution.
2. Let $\mathcal{D}^b$ be the induced distribution.
3. Let h be the binary classifier trained on data from $\mathcal{D}^b$ and achieves error of ϵ^b on $\mathcal{D}^b$.
4. We will compute the expected error ϵ^w of h on the weighted problem:

$$\begin{aligned}
\epsilon^w &= \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}^w}\left[\alpha^{y=+1}\left[h(\mathbf{x}) \neq y\right]\right] \\
&= \sum_{\mathbf{x}\in\mathcal{X}} \sum_{y\in\pm 1} \mathcal{D}^w(\mathbf{x}, y)\alpha^{y=+1}\left[h(\mathbf{x}) \neq y\right] \\
&= \alpha \sum_{\mathbf{x}\in\mathcal{X}} \left(\mathcal{D}^w(\mathbf{x}, +1)\left[h(\mathbf{x}) \neq +1\right] + \mathcal{D}^w(\mathbf{x}, -1)\frac{1}{\alpha}\left[h(\mathbf{x}) \neq -1\right]\right) \\
&= \alpha \sum_{\mathbf{x}\in\mathcal{X}} \left(\mathcal{D}^b(\mathbf{x}, +1)\left[h(\mathbf{x}) \neq +1\right] + \mathcal{D}^b(\mathbf{x}, -1)\left[h(\mathbf{x}) \neq -1\right]\right) \\
&= \alpha \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}^b}\left[h(\mathbf{x}) \neq y\right] \\
&= \alpha\epsilon^b
\end{aligned}$$

□

Reading

1. Section 4.1.2 of Pattern Recognition and Machine Learning Book (Bishop 2006).
2. Chapter 6 of A Course in Machine Learning (Daume III 2012).

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning*. Springer-Verlag.

Daume III, Hal (2012). *A Course in Machine Learning*. ciml.info.

Questions?