

# Reduction to Binary Classification

He He

CDS, NYU

March 30, 2021

# Overview

- So far, most algorithms we've learned are designed for binary classification.
- Many real-world problems have more than two classes.
- [discussion] What are some potential issues when we have a large number of classes?

# Today's lecture

- How to *reduce* multiclass classification to binary classification?
- How do we *generalize* binary classification algorithm to the multiclass setting?
- Example of very large output space: structured prediction.

## Reduction to Binary Classification

# One-vs-All / One-vs-Rest

## Setting

- Input space:  $\mathcal{X}$
- Output space:  $\mathcal{Y} = \{1, \dots, k\}$

## Training

- Train  $k$  binary classifiers, one for each class:  $h_1, \dots, h_k : \mathcal{X} \rightarrow \mathbb{R}$ .
- Classifier  $h_i$  distinguishes class  $i$  (+1) from the rest (-1).

## Prediction

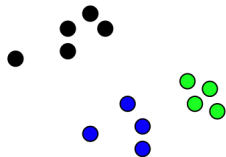
- Majority vote:

$$h(x) = \arg \max_{i \in \{1, \dots, k\}} h_i(x)$$

- Ties can be broken arbitrarily.

## OvA: 3-class example

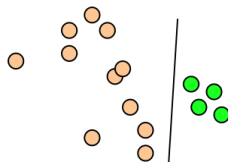
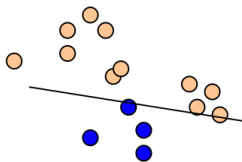
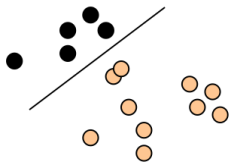
Consider a dataset with three classes:



**Assumption:** each class is linearly separable from the rest.

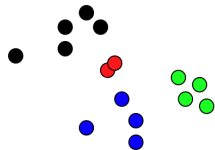
Ideal case: only target class has positive score.

Train OvA classifiers:



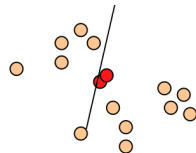
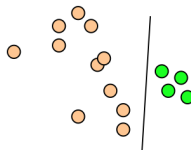
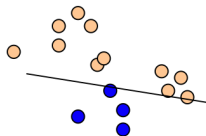
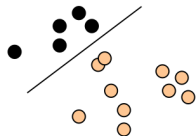
## OvA: 4-class non-separable example

Consider a dataset with four classes:



Cannot separate **red** points from the rest.  
Which classes might have low accuracy?

Train OvA classifiers:





# All vs All / One vs One / All pairs

## Setting

- Input space:  $\mathcal{X}$
- Output space:  $\mathcal{Y} = \{1, \dots, k\}$

## Training

- Train  $\binom{k}{2}$  binary classifiers, one for each pair:  $h_{ij} : \mathcal{X} \rightarrow \mathbb{R}$  for  $i \in [1, k]$  and  $j \in [i+1, k]$ .
- Classifier  $h_{ij}$  distinguishes class  $i$  (+1) from class  $j$  (-1).

## Prediction

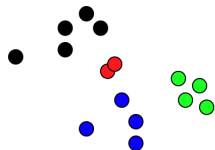
- Majority vote (each class gets  $k-1$  votes)

$$h(x) = \arg \max_{i \in \{1, \dots, k\}} \sum_{j \neq i} \underbrace{h_{ij}(x) \mathbb{I}\{i < j\}}_{\text{class } i \text{ is } +1} - \underbrace{h_{ji}(x) \mathbb{I}\{j < i\}}_{\text{class } i \text{ is } -1}$$

- Tournament
- Ties can be broken arbitrarily.

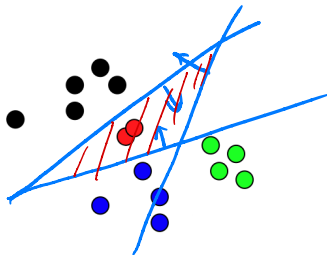
## AvA: four-class example

Consider a dataset with four classes:



**Assumption:** each pair of classes are linearly separable.  
More expressive than OvA.

What's the decision region for the red class?



## [discussion] OvA vs AvA

		OvA	AvA
computation	train	$O(kB_{\text{train}}(n))$	$O(k^2B_{\text{train}}(n/k))$
	test	$O(kB_{\text{test}})$	$O(k^2B_{\text{test}})$
challenges	train	class imbalance	small training set
	test	calibration / scale tie breaking	

Lack theoretical justification but simple to implement and works well in practice (when # classes is small).

**Question:** When would you prefer AvA / OvA?

## Code word for labels

Using the reduction approach, can you train fewer than  $k$  binary classifiers?

**Key idea:** Encode labels as binary codes and predict the code bits directly.

OvA encoding:

class	$h_1$	$h_2$	$h_3$	$h_4$
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

$\lceil \log_2 k \rceil$

OvA uses  $k$  bits to encode each label, what's the minimal number of bits you can use?

# Error correcting output codes (ECOC)

Example: 8 classes, 6-bit code

class	$h_1$	$h_2$	$h_3$	$h_4$	$h_5$	$h_6$
1	0	0	0	1	0	0
2	1	0	0	0	0	0
3	0	1	1	0	1	0
4	1	1	0	0	0	0
5	1	1	0	0	1	0
6	0	0	1	1	0	1
7	0	0	1	0	0	0
8	0	1	0	1	0	0

**Training** Binary classifier  $h_i$ :


- +1: classes whose  $i$ -th bit is 1
- -1: classes whose  $i$ -th bit is 0

**Prediction** Closest label in terms of Hamming distance.

$h_1$	$h_2$	$h_3$	$h_4$	$h_5$	$h_6$
0	1	1	0	1	1

**Code design** Want good binary classifiers.

## Error correcting output codes: summary

- Computationally more efficient than OvA (a special case of ECOC). Better for large  $k$ .
- Why not use the minimal number of bits ( $\log_2 k$ )?
  - If the minimum Hamming distance between any pair of code word is  $d$ , then it can correct  $\lfloor \frac{d-1}{2} \rfloor$  errors.
  - In plain words, if rows are far from each other, ECOC is robust to errors.
- Trade-off between code distance and binary classification performance.
- Nice theoretical results [Allwein et al., 2000] (also incorporates AvA).

Reduction-based approaches:

- Reducing multiclass classification to binary classification: OvA, AvA, ECOC.
- Key is to design “natural” binary classification problems without large computation cost.

But,

- Unclear how to generalize to extremely large # of classes.
- ImageNet: >20k labels; Wikipedia: >1M categories.

Next, generalize previous algorithms to multiclass settings.