Bandit Multiclass Linear Classification: Efficient Algorithms for the Separable Case

Alina Beygelzimer (Yahoo)
David Pal (Expedia)
Balazs Szorenyi (Yahoo)
Devanathan Thiruvenkatachari (NYU)
Chen-Yu Wei (USC)
Chicheng Zhang (UArizona)

ICML 2019

Bandit Classification

For $t = 1, 2, \ldots, T$:

1. Adversary chooses (x_t, y_t), where

   $x_t \in \mathbb{R}^d$ is the feature vector

   $y_t \in [K]$ is the label

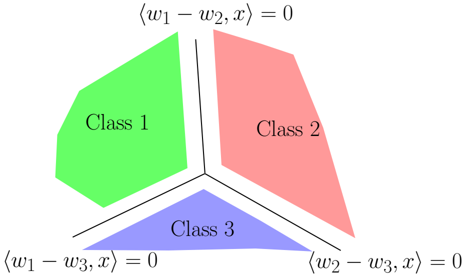
   and reveal x_t to the learner
2. Learner predicts a label $\widehat{y}_t \in [K]$.
3. Learner observes feedback $\mathbb{1}[\widehat{y}_t \neq y_t]$.

Goal: minimize the total number of mistakes

$$\sum_{t=1}^{T} \mathbb{1}[\widehat{y}_t \neq y_t]$$

Linearly Separable Data

Consider the ideal case: assume the incoming samples are linearly separable with a margin γ:



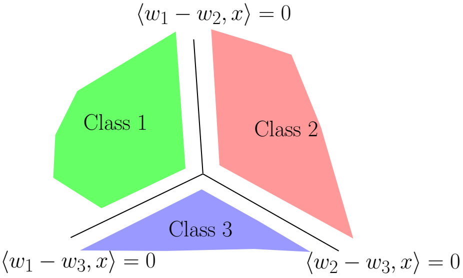
Linearly Separable Data

Consider the ideal case: assume the incoming samples are linearly separable with a margin γ:

$$\exists w_1, w_2, \ldots, w_K \in \mathbb{R}^d, \quad \sum_j \|w_j\|^2 \leq 1, \quad \text{such that}$$

for all (x, y) in the dataset,

$$w_{y}^\top x > w_{y'}^\top x + \gamma, \qquad \text{for all } y' \neq y$$



Mistake Bounds for Linearly Separable Data

Bounds on #mistakes:

1. [Kakade et al'08]: $\widetilde{O}\left(K^2 d \ln \frac{1}{\gamma}\right)$
2. [Daniely and Helbertal'13]: $\widetilde{O}\left(\frac{K}{\gamma^2}\right)$
3. [Kakade et al'08, Beygelzimer et al'17, Foster et al'18]: $\widetilde{O}\left(\frac{1}{\gamma}\sqrt{KT} + \frac{K}{\gamma^2}\right)$

$\widetilde{O}(f) \triangleq O\left(f \cdot \text{polylog}(f)\right)$

Mistake Bounds for Linearly Separable Data

Bounds on #mistakes:

1. [Kakade et al'08]: $\widetilde{O}\left(K^2 d \ln \frac{1}{\gamma}\right)$
2. [Daniely and Helbertal'13]: $\widetilde{O}\left(\frac{K}{\gamma^2}\right)$
3. [Kakade et al'08, Beygelzimer et al'17, Foster et al'18]: $\widetilde{O}\left(\frac{1}{\gamma}\sqrt{KT} + \frac{K}{\gamma^2}\right)$

$\widetilde{O}(f) \triangleq O\left(f \cdot \text{polylog}(f)\right)$

1&2. finite #mistakes, but exponential running time

3. polynomial-time algorithm, but infinite #mistakes

Mistake Bounds for Linearly Separable Data

Bounds on #mistakes:

1. [Kakade et al'08]: $\widetilde{O}\left(K^2 d \ln \frac{1}{\gamma}\right)$
2. [Daniely and Helbertal'13]: $\widetilde{O}\left(\frac{K}{\gamma^2}\right)$
3. [Kakade et al'08, Beygelzimer et al'17, Foster et al'18]: $\widetilde{O}\left(\frac{1}{\gamma}\sqrt{KT} + \frac{K}{\gamma^2}\right)$

$\widetilde{O}(f) \triangleq O\left(f \cdot \text{polylog}(f)\right)$

1&2. finite #mistakes, but exponential running time

3. polynomial-time algorithm, but infinite #mistakes

$\Rightarrow$ is there a polynomial-time algorithm with finite mistake bound?

Result Overview

- First polynomial-time algorithm with finite mistake bound
  - far from optimal — the mistake bound is exponential in some parameters
- Some negative results characterizing the difficulty of this problem

Result Overview

- First polynomial-time algorithm with finite mistake bound
  - far from optimal — the mistake bound is exponential in some parameters
- Some negative results characterizing the difficulty of this problem

Open Problem Is there a **polynomial** time algorithm with a **finite** and **polynomial** mistake bound?

Result Overview

	#mistake	running time
some previous works	finite and polynomial	exponential
other previous works	infinite	polynomial
this work	finite and exponential	polynomial
our hope	finite and polynomial	polynomial

Table: Bandit classification with linearly separable data

Outline

- Review of previous approaches
- Our approach

The Matrix Representation of Linear Classifiers

$$W \in \mathbb{R}^{K \times d}$$

K: #classes, d: feature dimension

$$Wx = \underbrace{\begin{bmatrix} \text{——} & w_1^\top & \text{——} \\ \text{——} & w_2^\top & \text{——} \\ & \vdots & \\ \text{——} & w_K^\top & \text{——} \end{bmatrix}}_{W} \begin{bmatrix} | \\ x \\ | \end{bmatrix} = \underbrace{\begin{bmatrix} w_1^\top x \\ w_2^\top x \\ \vdots \\ w_K^\top x \end{bmatrix}}_{\text{scores}}$$

For a feature vector x, the linear classifier W chooses the label

$$\operatorname*{argmax}_{i \in [K]} (Wx)_i$$

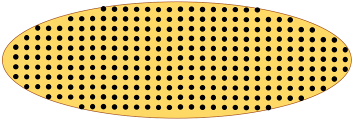
A set of data is linearly separable

$$\Updownarrow$$

$\exists$ linear classifer W^* that always chooses the correct label

i.e., for all (x, y) in the dataset, $\text{argmax}_{i \in [K]} (W^* x)_i = y$

Halving Algorithm [Kakade et al'08]



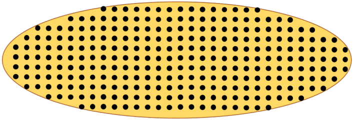
In each round t:

- Majority vote:

$$\widehat{y}_t = \underset{i \in [K]}{\operatorname{argmax}} \left| \{W \in \mathcal{H}_t : W \text{ chooses label } i\} \right|.$$

- If $\widehat{y}_t \neq y_t$: $\mathcal{H}_{t+1} \leftarrow \{W \in \mathcal{H}_t : W \text{ does not choose } \widehat{y}_t\}$

Halving Algorithm [Kakade et al'08]



In each round t:

- Majority vote:

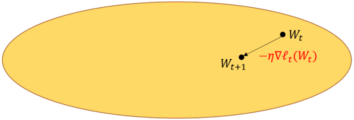
$$\hat{y}_t = \underset{i \in [K]}{\text{argmax}} \left| \{ W \in \mathcal{H}_t : W \text{ chooses label } i \} \right|.$$

- If $\hat{y}_t \neq y_t$: $\mathcal{H}_{t+1} \leftarrow \{ W \in \mathcal{H}_t : W \text{ does not choose } \hat{y}_t \}$

Every time $\hat{y}_t \neq y_t$, $|\mathcal{H}_{t+1}| \leq \left(1 - \frac{1}{K}\right) |\mathcal{H}_t|$

$|\mathcal{H}_1| = O\left(\frac{d}{\gamma}\right)^{Kd}$

Bandit Perceptron Approaches [Kakade et al'08, etc.]

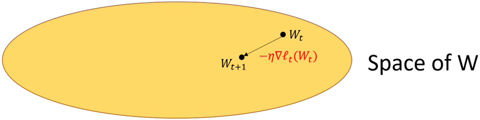


Space of W

online surrogate loss minimization

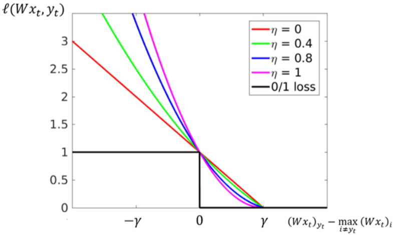
e.g. $\ell_t(W) = [\gamma - (Wx_t)_{y_t} + \max_{i \neq y_t}(Wx_t)_i]_+$

Bandit Perceptron Approaches [Kakade et al'08, etc.]

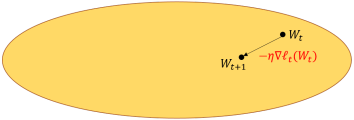


online surrogate loss minimization

e.g. $\ell_t(W) = [\gamma - (Wx_t)_{y_t} + \max_{i \neq y_t}(Wx_t)_i]_+$



Bandit Perceptron Approaches [Kakade et al'08, etc.]



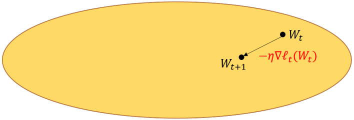
Space of W

online surrogate loss minimization
e.g. $\ell_t(W) = [\gamma - (Wx_t)_{y_t} + \max_{i \neq y_t}(Wx_t)_i]_+$

In each round t:

- $\widehat{y}_t = \begin{cases} W_t\text{'s choice of label} & \text{with probability } 1-\epsilon \\ \text{Uniform}([K]) & \text{with probability } \epsilon \end{cases}$
- If $\widehat{y}_t = y_t$, create surrogate loss $\ell_t(\cdot)$, and update $W_{t+1} \leftarrow W_t - \eta\nabla\ell_t(W_t)$.

Bandit Perceptron Approaches [Kakade et al'08, etc.]



Space of W

online surrogate loss minimization
e.g. $\ell_t(W) = [\gamma - (Wx_t)_{y_t} + \max_{i \neq y_t}(Wx_t)_i]_+$

In each round t:

- $\widehat{y}_t = \begin{cases} W_t\text{'s choice of label} & \text{with probability } 1-\epsilon \\ \text{Uniform}([K]) & \text{with probability } \epsilon \end{cases}$
- If $\widehat{y}_t = y_t$, create surrogate loss $\ell_t(\cdot)$, and update $W_{t+1} \leftarrow W_t - \eta\nabla\ell_t(W_t)$.

Fact: It is difficult to design a *convex* surrogate loss if you only have a wrong label but do not know the true label. (Why?)

A Difference between Halving and Bandit Perceptron

- Halving makes *great* progress when it makes a mistake: $|\mathcal{H}_{t+1}| \leq (1 - 1/K)|\mathcal{H}_t|$
- Bandit Perceptron makes *no* progress when it makes a mistake

A Difference between Halving and Bandit Perceptron

- Halving makes *great* progress when it makes a mistake: $|\mathcal{H}_{t+1}| \leq (1 - 1/K)|\mathcal{H}_t|$
- Bandit Perceptron makes *no* progress when it makes a mistake

We showed: if an algorithm does not update itself when it makes a mistake, then the adversary can force

$$\#\text{mistake} \geq \min\left\{\sqrt{T}, 2^{\Omega(d)}\right\}.$$

A Difference between Halving and Bandit Perceptron

- Halving makes *great* progress when it makes a mistake: $|\mathcal{H}_{t+1}| \leq (1 - 1/K)|\mathcal{H}_t|$
- Bandit Perceptron makes *no* progress when it makes a mistake

We showed: if an algorithm does not update itself when it makes a mistake, then the adversary can force

$$\#\text{mistake} \geq \min\left\{\sqrt{T}, 2^{\Omega(d)}\right\}.$$

Lesson learned: our algorithm should update when it makes a mistake

Our Algorithm

The simple idea of our algorithm:

- When can we efficiently update when $\widehat{y}_t \neq y_t$?

Our Algorithm

The simple idea of our algorithm:

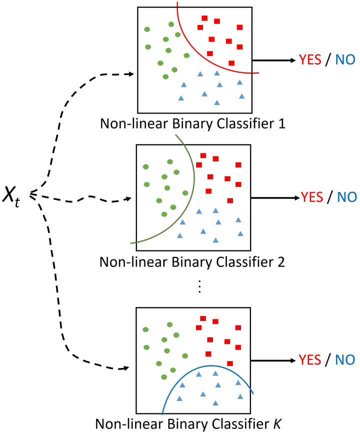
- When can we efficiently update when $\widehat{y}_t \neq y_t$?
  $\Rightarrow$ binary classification (can know y_t when only seeing $\mathbb{1}[\widehat{y}_t \neq y_t]$)

Our Algorithm

The simple idea of our algorithm:

- When can we efficiently update when $\widehat{y}_t \neq y_t$?
  $\Rightarrow$ binary classification (can know y_t when only seeing $\mathbb{1}[\widehat{y}_t \neq y_t]$)
- Reduce our problem to binary classification

Our Algorithm: Bandit-OvA



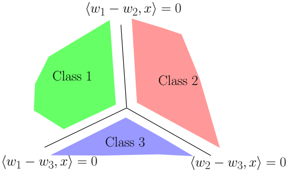
Case 1: ≥ 1 of them respond YES
$\widehat{y}_t \leftarrow$ any one of those YES labels
If $\widehat{y}_t \neq y_t$, update $\widehat{y}_t$-th sub-learner

Case 2: all of them respond NO
$\widehat{y}_t \leftarrow$ uniform from $\{1, \ldots, K\}$
If $\widehat{y}_t = y_t$, update $\widehat{y}_t$-th sub-learner

$$\mathbb{E}[\#\text{mistakes(alg)}] \leq K \sum_i \#\text{mistakes}(i)$$

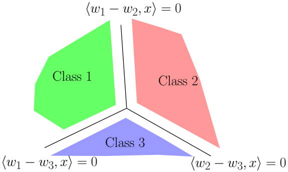
Our Algorithm: Bandit-OvA

- Each sub-learner learns the support of class i, which lies in an intersection of $K - 1$ halfspaces with a margin.



Our Algorithm: Bandit-OvA

- Each sub-learner learns the support of class i, which lies in an intersection of $K-1$ halfspaces with a margin.

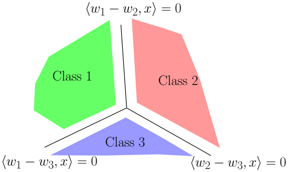


- Sub-learner = **2-class Kernel Perceptron** with rational kernel [Klivans and Servedio'04, Shalev-Shwartz et al'11]:

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

Our Algorithm: Bandit-OvA

- Each sub-learner learns the support of class i, which lies in an intersection of $K-1$ halfspaces with a margin.



- Sub-learner = **2-class Kernel Perceptron** with rational kernel [Klivans and Servedio'04, Shalev-Shwartz et al'11]:

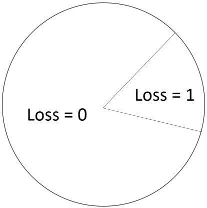
$$K(x, x') = \frac{1}{1 - \frac{1}{2}\langle x, x'\rangle}$$

- $\#\text{mistakes(sub-learner)} \leq O\left(\frac{1}{\gamma'^2}\right) = 2^{\widetilde{O}\left(\min\left\{K \log^2(1/\gamma), \sqrt{1/\gamma} \log K\right\}\right)}$

Difficulty of designing a surrogate loss when $\widehat{y}_t \neq y_t$

- When the learner makes a mistake ($\widehat{y}_t \neq y_t$), the set of W's we want to **penalize** is

$$\left\{ W : (Wx_t)_{\widehat{y}_t} > (Wx_t)_i,\ \forall i \neq \widehat{y}_t \right\}$$



Difficult to design a convex surrogate loss $\ell_t(W)$.

A Side Result Indicating the Difficulty

- The **offline problem** is NP-hard:

  Given a mixed-labeled dataset which consists of two types of samples:

  $$(x, y) : x \text{ belongs to class } y$$
  $$(x, \overline{y}) : x \text{ does not belong to class } y$$

  Given that this dataset is separable with $\gamma = \frac{1}{2}$ and $K = 3$. Find a linear classifier W^*.

Summary

- We studied the problem of bandit multiclass classification with linearly separable data
- We developed the first polynomial time algorithm that has finite number of mistakes
- It remains open how to make the number of mistakes polynomial (or proving that this is computationally hard)

Thank you!