Multi-class Online Learning with Bandit Feedback

Wenxi Chen & Wei Zhang

Columbia University

December 11, 2017

Introduction

Setting

- Multi-class online learning
- Bandit feedback (partial feedback)
- Agnostic Setting

Goal

The goal is to achieve an expected regret bound by $\tilde{O}(\sqrt{dTK})$, where d is some sort of dimension (e.g. VC-dimension, Littlestone-dimension).

Inspiration

Theorem 3.6 from [Shalev-Shwartz et al., 2012] uses an algorithm that achieves $O(\sqrt{\mathrm{Ldim}(\mathcal{H})\ln(T)T})$ expected regret in agnostic binary online learning setting.

Difficulties

- Littlestone-dimension and VC-dimension are in binary cases.
- We have bandit feedback.

Method Summary

1. Generalize Littlestone-dimension to K-classes $\Rightarrow$ *BL*-dim.
2. Generalize *SOA* algorithm from [Shalev-Shwartz et al., 2012] to K-classes $\Rightarrow$ *BSOA*.
3. Generalize Theorem 3.6 from [Shalev-Shwartz et al., 2012] to multi-class bandit feedback.
   - Initialize a set of experts S at the beginning of the algorithm.
   - Initialize in such a way: for every behavior from $\mathcal{H}$, there is at least 1 expert has the same behavior.
     $\Rightarrow$ *Expert2, Modified Exp4*
   - Conceptualize the "fork" trick.
   - Use *Exp4* instead of regular *Weighted Majority*.

BL-dim [Daniely et al., 2011]

A tree $\mathcal{T}$ which is BL-shattered by $\mathcal{H}$ is a shattered tree that generalizes to a K-ary tree in the multi-class setting where there are K classes.

BL-dim [Daniely et al., 2011]

A tree $\mathcal{T}$ which is BL-shattered by $\mathcal{H}$ is a shattered tree that generalizes to a K-ary tree in the multi-class setting where there are K classes.

We introduce the bandit Littlestone dimension of $\mathcal{H}$, denoted $BL\text{-Dim}(\mathcal{H})$, as the maximal depth of a complete K-ary tree that is BL-Shattered by $\mathcal{H}$.

BL-dim [Daniely et al., 2011]

A tree $\mathcal{T}$ which is BL-shattered by $\mathcal{H}$ is a shattered tree that generalizes to a K-ary tree in the multi-class setting where there are K classes.

We introduce the bandit Littlestone dimension of $\mathcal{H}$, denoted BL-Dim($\mathcal{H}$), as the maximal depth of a complete K-ary tree that is BL-Shattered by $\mathcal{H}$.

For any hypothesis class $\mathcal{H}$, the number of mistakes that $BSOA$ will make is at most BL-Dim($\mathcal{H}$).

BL-dim [Daniely et al., 2011]

A tree $\mathcal{T}$ which is BL-shattered by $\mathcal{H}$ is a shattered tree that generalizes to a K-ary tree in the multi-class setting where there are K classes.

We introduce the bandit Littlestone dimension of $\mathcal{H}$, denoted $BL\text{-Dim}(\mathcal{H})$, as the maximal depth of a complete K-ary tree that is BL-Shattered by $\mathcal{H}$.

For any hypothesis class $\mathcal{H}$, the number of mistakes that *BSOA* will make is at most $BL\text{-Dim}(\mathcal{H})$.

Expert2 creates a set of experts U that at least one of them mimic the behavior of a given hypothesis $h \in \mathcal{H}$ between round 1 and T inclusively, such that $|U| = (K-1)^L$.

Modified Exp4

Algorithm 1 Modified Exp4

Require: $\eta > 0$; a hypothesis class $\mathcal{H}$

Initialize $w_1 = (w_{1,1}, w_{1,2}, \cdots, w_{1,N}) = (1, 1, \cdots, 1)$

Initialize the set for experts S with the size of N where

$$N = \sum_{(U,L)} (K-1)^L$$

for t=1,2,... **do**

  Generate action and update version space for each expert as in round t in *Expert2*

  Receive experts' actions: $\forall i \in [N]$, $b_{t,i} \in [K]$

  ... (regular *Exp4*)

end for

Final Bound

Let $D = BL\text{-Dim}(\mathcal{H})$,

$$N = \sum_{L=0}^{D} \binom{T}{L} (K-1)^L \leq (K-1)^D (\frac{eT}{D})^D.$$

By using *Exp4*'s theorem from lecture note:

$$\mathbb{E}[\text{Regret}_T] = \sqrt{2KT \ln N} = \tilde{O}(\sqrt{2KTD}).$$

Appendix: *BSOA* [Daniely et al., 2011]

Algorithm 2 Bandit Standard Optimal Algorithm(BSOA)

Require: a hypothesis class $\mathcal{H}$
init: $V_0 = \mathcal{H}$
for t=1,2,... **do**
    Receive x_t
    For $y \in [k]$, let $V_t^{(y)} = \{f \in V_{t-1} : f(x_t) \neq y\}$
    Predict $a_t = \operatorname*{argmin}_y BL\text{-Dim}(V_t^y)$
    if $c_t = 1$ **then**
        Update $V_t = V_t^{(a_t)}$
    else
        $V_t = V_{t-1}$
    end if
end for

Appendix: *Expert2*

Algorithm 3 Expert2

```
Require: A hypothesis class $\mathcal{H}$; indices $i_1 < i_2 < \cdots < i_L$
init: $V_0 = \mathcal{H}$
for t=1,2,..., $T$ do
    Receive $x_t$
    For $y \in [k]$, let $V_t^{(y)} = \{f \in V_{t-1} : f(x_t) \neq y\}$
    Define $\hat{y}_t = \underset{y}{\operatorname{argmin}}\, BL\text{-Dim}(V_t^y)$
    if $t \in \{i_1, i_2, \ldots, i_L\}$ then
        Replace this program with $k-1$ clones, each with distinct prediction $a_t \in [k] \setminus \hat{y}_t$
        Update $V_t = V_t^{(a_t)}$
    else
        Predict $a_t = \hat{y}_t$
    end if
end for
```

Appendix: *Modified Exp4*

Algorithm 4 Modified Exp4

Require: $\eta > 0$; a hypothesis class $\mathcal{H}$

Initialize $w_1 = (w_{1,1}, w_{1,2}, \cdots, w_{1,N}) = (1, 1, \cdots, 1)$

Initialize the set for experts S with the size of N where

$$N = \sum_{L=0}^{BL\text{-Dim}(\mathcal{H})} \binom{T}{L} (K-1)^L$$

for t=1,2,... **do**

  Generate action and update version space for each expert as in round t in *Expert2*

  Receive experts' actions: $\forall i \in [N]$, $b_{t,i} \in [K]$

  ... (regular *Exp4*)

end for

References I

Amit Daniely, Sivan Sabato, Shai Ben-David, and Shai Shalev-Shwartz. Multiclass learnability and the erm principle. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 207–232, 2011.

Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4 (2):107–194, 2012.

Through the Lens of Oracle: Efficient Algorithm for Finding Selective Classifier

Presented by Yanlin (Alan) Duan, Rong Zhou

Review of selective classifier

Recall the definition of a selective classifier:

Review of selective classifier

Recall the definition of a selective classifier:

A selective classifier C is a tuple $(h, (\gamma_1, \cdots, \gamma_m))$, where h lies in a hypothesis class $\mathcal{H}$, and $0 \leq \gamma_i \leq 1$ for all $i = 1, \cdots, m$. For any $x_{n+j} \in U$, $C(x_{n+j}) = h(x_{n+j})$ wp γ_j and 0 wp $1 - \gamma_j$.

Review of selective classifier

Below is the algorithm (optimization problem) for finding selective classifier:

Algorithm 1: Optimization Problem for the Selective Classifier (3)

1 Pick an arbitrary $h_0 \in V$, where V is the version space with respect to the labelled samples in S.

2 Solve the linear program:

$$\max \sum_{i=1}^{m} \gamma_i$$

subject to:

$$\forall i, 0 \leq \gamma_i \leq 1 \tag{1}$$

$$\forall h \in V, \sum_{i:h(x_{n+i}) \neq h_0(x_{n+i})} \gamma_i \leq \epsilon m \tag{2}$$

3 Output the selective classifier:

$$(h_0, (\gamma_1, \gamma_2, \ldots, \gamma_m))$$

Review of selective classifier

Below is the algorithm (optimization problem) for finding selective classifier:

Algorithm 1: Optimization Problem for the Selective Classifier (3)

1. Pick an arbitrary $h_0 \in V$, where V is the version space with respect to the labelled samples in S.
2. Solve the linear program:

$$\max \sum_{i=1}^{m} \gamma_i$$

subject to:

$$\forall i, 0 \leq \gamma_i \leq 1 \quad (1)$$

$$\forall h \in V, \sum_{i: h(x_{n+i}) \neq h_0(x_{n+i})} \gamma_i \leq \epsilon m \quad (2)$$

3. Output the selective classifier:

$$(h_0, (\gamma_1, \gamma_2, \ldots, \gamma_m))$$

How many constraints do we have?

Review of selective classifier

Below is the algorithm (optimization problem) for finding selective classifier:

Algorithm 1: Optimization Problem for the Selective Classifier (3)

1. Pick an arbitrary $h_0 \in V$, where V is the version space with respect to the labelled samples in S.
2. Solve the linear program:

$$\max \sum_{i=1}^{m} \gamma_i$$

subject to:

$$\forall i, 0 \leq \gamma_i \leq 1 \tag{1}$$

$$\forall h \in V, \sum_{i:h(x_{n+i}) \neq h_0(x_{n+i})} \gamma_i \leq \epsilon m \tag{2}$$

3. Output the selective classifier:

$$(h_0, (\gamma_1, \gamma_2, \ldots, \gamma_m))$$

How many constraints do we have?

As many as $|V|$!

Motivation of our project

Motivation of our project

Can we solve the optimization problem more efficiently by accessing the hypothesis class not directly through (large amount of) constraints, but through some ERM oracle?

Motivation of our project

Can we solve the optimization problem more efficiently by accessing the hypothesis class not directly through (large amount of) constraints, but through some ERM oracle?

Yes!

Motivation of our project

Can we solve the optimization problem more efficiently by accessing the hypothesis class not directly through (large amount of) constraints, but through some ERM oracle?

Yes!

Definition of ERM oracle we use: For a set of hypothesis $\mathcal{H}$, the *Weighted ERM Oracle* is an algorithm, which for any sequence $(x_1, y_1, w_1), (x_2, y_2, w_2), \ldots (x_t, y_t, w_t) \in Z \subseteq \mathcal{X} \times Y \times \mathbb{R}_{\geq 0}$, returns $\operatorname{argmin}_{h \in \mathcal{H}} \sum_{(x,y,w) \in Z} \mathbb{1}\{h(x) \neq y\} \cdot w$

Main Algorithm

Algorithm 2: Coordinate ascent algorithm for solving the optimization problem

```
Inputs: Accuracy parameter ε > 0. Initialize parameter λ_1, λ_2, ..., λ_N s.t. γ_i = 1, ∀i.
Find h_0 using the Weighted ERM Oracle with input S and weights ∞ for each sample
  x_1, x_2 ... x_n ∈ S.
while true do
    Rescale: λ ← s · λ where s = arg max_{s∈[0,1]} D(s · λ).
    Find h_p using the the Weighted ERM Oracle(Algorithm 3).
    if ∀i, Σ_{i:h_p(x_{n+i})≠h_0(x_{n+i})} γ_i > εm then
        Update λ_{h_p} ← λ_{h_p} + δ, where δ = 2 [Σ_{i=1}^m I^i_{h_p} (Σ_{j=1}^N λ_j I^i_{h_j})^{-0.5} − εm] / [Σ_{i=1}^m I^i_{h_p} (Σ_{j=1}^N λ_j I^i_{h_j})^{-1.5}].
    else
        return λ_1, λ_2, ... λ_N.
    end
end
```

$$\delta = 2\frac{\sum_{i=1}^{m} \mathcal{I}^i_{h_p}\left(\sum_{j=1}^{N} \lambda_j \mathcal{I}^i_{h_j}\right)^{-0.5} - \epsilon m}{\sum_{i=1}^{m} \mathcal{I}^i_{h_p}\left(\sum_{j=1}^{N} \lambda_j \mathcal{I}^i_{h_j}\right)^{-1.5}}$$

Algorithm 3: Finding the most violated hypothesis using the *Weighted ERM Oracle*

1 **Inputs:** hypothesis h_0, labelled samples in S, and unlabelled samples in U.

2 Label all the m unlabelled points as $y_{n+i} = -h_0(x_{n+i})$ for all $i = 1, 2, \ldots m$.

3 Use the ERM oracle to find the empirical risk minimizer of the n labelled samples in S with weights ∞, and m samples labeled as step 1 with weights of each sample γ_i.

4 **Return** the hypothesis in step 3.

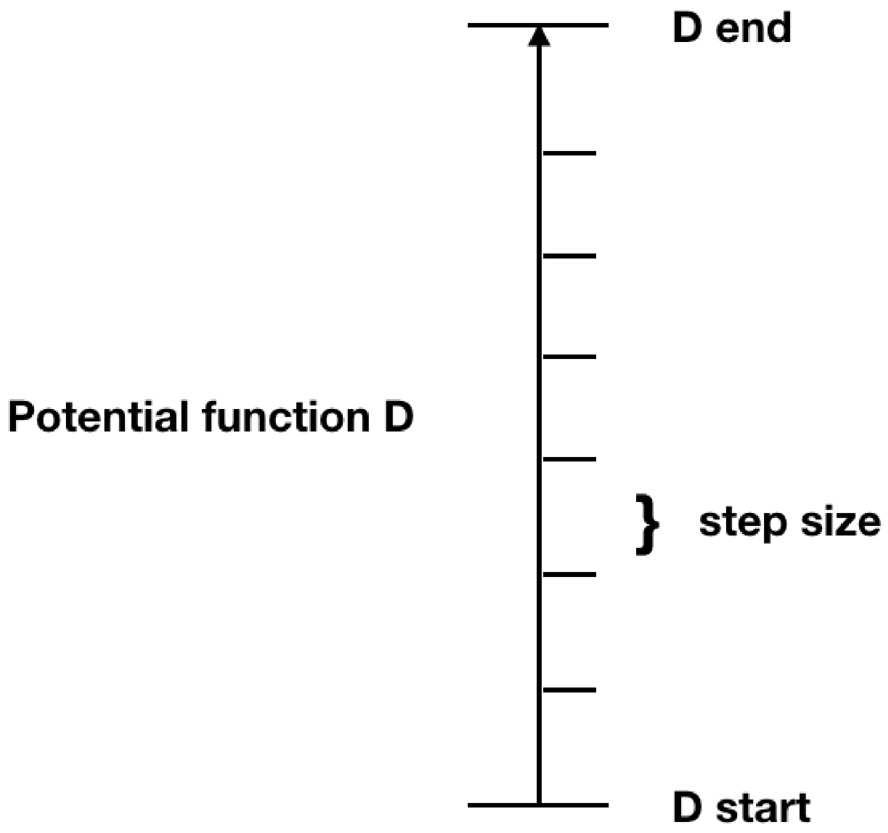
Analysis of algorithm

Question: After how many rounds will the algorithm halt?

Analysis of algorithm

Question: After how many rounds will the algorithm halt?

High level idea:



Find step size

Transform from primal to dual by introducing Lagrange multipliers:

$$\mathcal{L} = \sum_{i=1}^{m} \frac{1}{\gamma_i} + \sum_{j=1}^{N} \lambda_j \left(\left(\sum_{i=1}^{m} \gamma_i \mathbb{1}_{h_j(x_{n+i}) \neq h_0(x_{n+i})} \right) - \epsilon m \right)$$

Find step size

Transform from primal to dual by introducing Lagrange multipliers:

$$\mathcal{L} = \sum_{i=1}^{m} \frac{1}{\gamma_i} + \sum_{j=1}^{N} \lambda_j \left(\left(\sum_{i=1}^{m} \gamma_i \mathbb{1}_{h_j(x_{n+i}) \neq h_0(x_{n+i})} \right) - \epsilon m \right)$$

Find derivative and set to zero. Substitute γ_i using the following:

$$\gamma_i^2 = \left(\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i}) \neq h_0(x_{n+i})} \right)^{-1}$$

Find step size

Transform from primal to dual by introducing Lagrange multipliers:

$$\mathcal{L} = \sum_{i=1}^{m} \frac{1}{\gamma_i} + \sum_{j=1}^{N} \lambda_j \left(\left(\sum_{i=1}^{m} \gamma_i \mathbb{1}_{h_j(x_{n+i}) \neq h_0(x_{n+i})} \right) - \epsilon m \right)$$

Find derivative and set to zero. Substitute γ_i using the following:

$$\gamma_i^2 = \left(\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i}) \neq h_0(x_{n+i})} \right)^{-1}$$

Now dual looks like:

$$D(\lambda) = \sum_{i=1}^{m} \frac{2}{\sqrt{Q_i(\lambda)}} - \sum_{j=1}^{N} \lambda_j \epsilon m$$

where

$$Q_i(\lambda) = \left(\sum_{j=1}^{N} \lambda_j \mathcal{I}_{h_j}^{i} \right)^{-1} = \gamma_i^2$$

$$\mathcal{I}_{h_j}^{i} = \mathbb{1}_{h_j(x_{n+i}) \neq h_0(x_{n+i})}$$

Find step size (cont'd)

Denote

$$\lambda' = \begin{cases} \lambda_p + \delta, & \text{if } h_p \text{ is the most violated constraint} \\ \lambda, & \text{otherwise} \end{cases}$$

Find step size (cont'd)

Denote

$$\boldsymbol{\lambda}' = \begin{cases} \lambda_p + \delta, & \text{if } h_p \text{ is the most violated constraint} \\ \lambda, & \text{otherwise} \end{cases}$$

We examine the difference in dual since update: $D(\boldsymbol{\lambda}') - D(\boldsymbol{\lambda})$ (our potential function)

Find step size (cont'd)

Denote

$$\boldsymbol{\lambda}' = \begin{cases} \lambda_p + \delta, & \text{if } h_p \text{ is the most violated constraint} \\ \lambda, & \text{otherwise} \end{cases}$$

We examine the difference in dual since update: $D(\boldsymbol{\lambda}') - D(\boldsymbol{\lambda})$ (our potential function)

$D(\boldsymbol{\lambda}') - D(\boldsymbol{\lambda}) \geq A$

Find step size (cont'd)

Denote

$$\boldsymbol{\lambda}' = \begin{cases} \lambda_p + \delta, & \text{if } h_p \text{ is the most violated constraint} \\ \lambda, & \text{otherwise} \end{cases}$$

We examine the difference in dual since update: $D(\boldsymbol{\lambda}') - D(\boldsymbol{\lambda})$ (our potential function)

$$D(\boldsymbol{\lambda}') - D(\boldsymbol{\lambda}) \geq A$$

Find δ that maximizes A (lower bound of dual difference). Then find how much the lower bound will increase with that δ -- this is our step size!

Find initial and end point

We initialize each $\gamma_i = 1$, so:

$D(\lambda_{\text{start}}) = 2m - \sum_{j=1}^{N} \lambda_j \epsilon m \geq 2m - N\epsilon m.$

We know that $\gamma_1 = \gamma_2 = \cdots = \gamma_m = \epsilon$ is a feasible solution, so:

$D(\lambda_{\text{end}}) \leq 2\sqrt{\epsilon} mN - N\epsilon^2 m$

Final result and future work

The final bound we get looks like this:

$$\frac{D(\lambda_{\text{end}})-D(\lambda_{\text{start}})}{t} = \frac{m(N(2\sqrt{\epsilon}-\epsilon^2+\epsilon)-2)\cdot\sum_{i=1}^{m} \frac{\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}}{(\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i})\neq h_0(x_{n+i})})^{1.5}}}{\left(\sum_{i=1}^{m} \frac{\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}}{\sqrt{\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i})\neq h_0(x_{n+i})}}}-\epsilon m\right)^2}$$

Final result and future work

The final bound we get looks like this:

$$\frac{D(\lambda_{\text{end}})-D(\lambda_{\text{start}})}{t} = \frac{m(N(2\sqrt{\epsilon}-\epsilon^2+\epsilon)-2)\cdot\sum_{i=1}^{m} \frac{\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}}{(\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i})\neq h_0(x_{n+i})})^{1.5}}}{\left(\sum_{i=1}^{m} \frac{\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}}{\sqrt{\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i})\neq h_0(x_{n+i})}}}-\epsilon m\right)^2}$$

Note that it contains $\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}$.

Final result and future work

The final bound we get looks like this:

$$\frac{D(\lambda_{\text{end}})-D(\lambda_{\text{start}})}{t} = \frac{m(N(2\sqrt{\epsilon}-\epsilon^2+\epsilon)-2)\cdot\sum_{i=1}^{m} \frac{\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}}{\left(\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i})\neq h_0(x_{n+i})}\right)^{1.5}}}{\left(\sum_{i=1}^{m} \frac{\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}}{\sqrt{\sum_{j=1}^{N} \lambda_j \mathbb{1}_{h_j(x_{n+i})\neq h_0(x_{n+i})}}} - \epsilon m\right)^2}$$

Note that it contains $\mathbb{1}_{h_p(x_{n+i})\neq h_0(x_{n+i})}$.

We tried to get rid of this, but so far no luck. Maybe the future work will be to finish up the bound by elinimating this term.

References

1. T.-K. Huang, A. Agarwal, D. J. Hsu, J. Langford, and R. E. Schapire, “E!cient and parsimonious agnostic active learning,” in Advances in Neural Information Processing Systems, pp. 2755–2763, 2015.

2. A. Agarwal, D. Hsu, S. Kale, J. Langford, L. Li, and R. E. Schapire, “Taming the Monster: A Fast and Simple Algorithm for Contextual Bandits,” ArXiv e-prints, Feb. 2014.

3. K. Chaudhuri and C. Zhang, “Improved algorithms for confidence-rated prediction with error guarantees,” 2013.

Thanks!

Any questions?

THe S^2 algorithm

Arushi Gupta ag3309

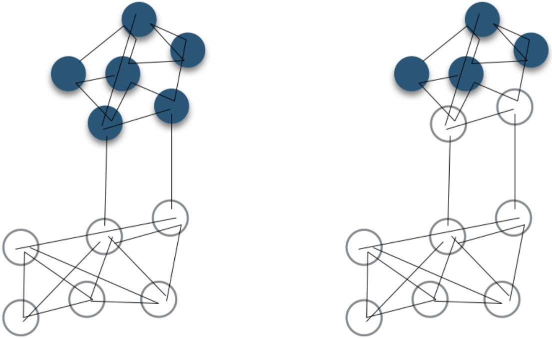
Dec 11

Active learning on graphs

- G = (V,E) is our graph
- There is a labeling function, f: V $\rightarrow \{+1, -1\}$
- we wish to query as few vertices as possible

Comparison with clustering

- S^2 determines the precise decision boundaries given high enough labeling budget



The algorithm [Dasarathy et al]

- sample random pts until you find two with different labels

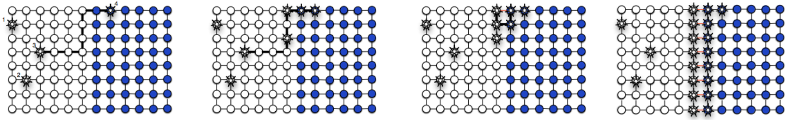
Algorithm 1: S^2 : Shortest Shortest Path

Input Graph $G = (V, E)$, BUDGET $\leq n$
$L \leftarrow \emptyset$
while 1 **do**
  $x \leftarrow$ Randomly chosen unlabeled vertex
  do
    Add $(x, f(x))$ to L
    Remove from G all edges whose two ends have different labels.
    if $|L| =$ BUDGET **then**
      Return LABELCOMPLETION(G, L)
    end if
  while $x \leftarrow$ MSSP(G, L) exists
end while

Sub-routine 2: MSSP

Input Graph $G = (V, E)$, $L \subseteq V$
for each $v_i, v_j \in L$ such that $f(v_i) \neq f(v_j)$ **do**
  $P_{ij} \leftarrow$ shortest path between v_i and v_j in G
  $\ell_{ij} \leftarrow$ length of P_{ij} (∞ if no path exists)
end for
$(i^*, j^*) \leftarrow \arg\min_{v_i, v_j \in L : f(v_i) \neq f(v_j)} \ell_{ij}$
if (i^*, j^*) exists **then**
  Return mid-point of $P_{i^* j^*}$ (break ties arbitrarily).
else
  Return $\emptyset$
end if

Sample run



(*a*) Random sampling ends. One shortest shortest path is shown with thickened edges.

(*b*) This path bisected and S^2 finds one cut edge.

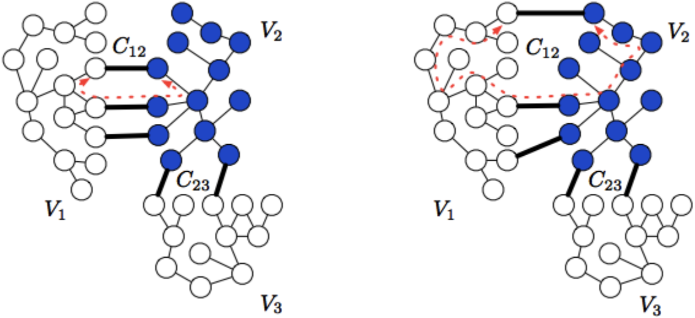
(*c*) Next shortest-shortest path is bisected.

(*d*) This continues till S^2 unzips through the cut boundary.

Figure 1: A sample run of the S^2 algorithm on a grid graph. The shaded and unshaded vertices represent two different classes (say $+1$ and -1 respectively). See text for explanation.

Conditions which make problem solvable

- balancedness $\min \frac{|V_i|}{k}$
- κ-clustering



(a) (b)

Maximal Planar graphs

- Saying a graph is a maximal planar graph and plane triangulation is the same thing
- $\kappa = 1$ here.

Conclusions

- Difference between active learning on graph and clustering
- κ as a measure of how difficult problem is to solve

Teaching with Partial Knowledge to Learner's Hypothesis Set

COMS 6998 Final Project Summary

Qinyao He Xuefeng Hu

December, 2017

Computer Science Department
Columbia University

Outline

Problem Statement

Problem Statement

As defined in Goldman and Kearns' *On the Complexity of Teaching*, the Teaching Dimension of a Concept Class C is defined as

$$\mathrm{TD}(C) := \max_{c \in C} \left(\min_{\tau \in T(c)} |\tau| \right) \tag{1}$$

where $T(c)$ is the set of all teaching sequence τ that uniquely specify concept c in Concept Class C.

Teaching Dimension as defined as, for any concept $c \in C$, the minimum necessary samples to teach the learner in order to let learner to uniquely identify c among his/her/its Hypothesis Set. Therefore, it is useful only when Teacher knows Learner's Hypothesis Class.

Question: What if Teacher does not know exactly what the learner's hypothesis class is?

Main Results

Setting

- Sample Space $\mathcal{X} \times \mathcal{Y} \sim \mathcal{D}$, where $\mathcal{X}$ is the set of the observations, $\mathcal{Y}$ is the set of the corresponding labels, and $\mathcal{D}$ is the sample distribution.

Unconstrained Learner
Setting

- Learner
  - Learner using a unknown but realizable Hypothesis Set $\mathcal{H}_L$ to learn.
  - Consistent Learner.
- Teacher
  - Aims on teaching Concept Class $C \in \mathcal{H}_L$ to the Learner.
  - TEACH: teach learner with sample $(x, y) \in \mathcal{X} \times \mathcal{Y}$
  - ASK: ask learner to give a prediction $\hat{y}$ on a selected sample point $x \in \mathcal{X}$.

- Main Result: If there is no other constrain on the Learner, we claim that it is not likely to achieve a better result than the trivial teaching dimension bound $|\mathcal{X}|$.
- Proved by showing that TEACH is more informative than ASK in this setting.

- Learner
  - Learner uses Hypothesis Set $\mathcal{H}_L$ from n possible Hypothesis Sets $\{\mathcal{H}_1, \ldots, \mathcal{H}_n\}$.
  - Consistent Learner.
  - Leaner can inform Teacher when there is not hypothesis consistent with the current samples.
- Teacher
  - Teacher has access to an Oracle $\mathcal{Q}(\mathcal{H})$ to get the teaching dimension of any hypothesis class and also the optimal teaching sequence for any hypothesis $h \in \mathcal{H}$.
  - Teacher has knowledge of $\{\mathcal{H}_1, \ldots, \mathcal{H}_n\}$, but does not know which one is $\mathcal{H}_L$.
  - Teacher can inform Leaner that a teaching session is ended to have a fresh restart.

Finite Hypothesis Classes
A Trivial Approach

- **Observation:** $\mathcal{TD}(\cup_{i=1}^{n}\mathcal{H}_i)$ can be very large even each $\mathcal{TD}(\mathcal{H}_i)$ is bounded.
- **Trivial Approach:** Eliminate $\mathcal{H}_i$ one by one, by teach the optimal teaching sequence and check if the learner successfully learned the target concept (let the learner tell if it's version space contains only one).
- **Trivial Bound:** Worst case

$$\sum_{i=1}^{n} \mathcal{TD}(\mathcal{H}_i)$$

samples to teach.

- **Observation:** Notice in some case $\mathcal{TD}(\mathcal{H}_1 \cup \mathcal{H}_2) < \mathcal{TD}(\mathcal{H}_1) + \mathcal{TD}(\mathcal{H}_2)$.
- **Example:** $h_{k,m} = \{n | n \equiv k \mod m\}$.
- **Improvement:** Find all possible pairs and merge them iteratively, with at most $O(n^3)$ call to $\mathcal{Q}$.

Compact Hypothesis Representation Setting

- Learner
  - Learner using a unknown but realizable Hypothesis Set $\mathcal{H}_L$ to learn.
  - Consistent Learner.
- Teacher
  - Aims on teaching Concept Class $C \in \mathcal{H}_L$ to the Learner. Not aims on unique identification, but rather an *probably approximately correct* estimation.
  - TEACH: teach learner with sample $(x, y) \in \mathcal{X} \times \mathcal{Y}$
  - ASK2: ask learner to give return a hypothesis $h^{(t)}$ in the current Version Space in a compact representation, so that it does not require large amount of information to be transferred.

Compact Hypothesis Representation
Inspiration From Active Learning

- Recall in CAL active learning algorithm, learner query labels when it is "confused". Only queried sample make progress to learning (reduce the version space).
- Can we only teach those sample make actual progress?
- We claim that teacher also only need to teach

$$O(\theta(h^*, \mathcal{H}, \mathcal{D}) d \log^2(n))$$

samples to let the learner learns an ϵ-approximation of the true concept with at least $1 - \delta$ probability, where $n = \frac{1}{\epsilon}(d \log \frac{1}{\epsilon} + \log \frac{1}{\delta})$ is the sample complexity of the classic PAC learning.

Algorithm 1 HH algorithm

Input: ϵ the error rate and δ the confidence level, and d is the upper bound of learner's hypothesis class' VC dimension.

1: Teacher draw n i.i.d. samples $\mathcal{X}_n := \{(x_1, h^*(x_1)), ..., (x_n, h^*(x_n))\}$ from $\mathcal{X} \times \mathcal{Y}$ according to $\mathcal{D}$, where

$$n = \frac{1}{\epsilon}(d \log \frac{1}{\epsilon} + \log \frac{1}{\delta})$$

2: **for** $i \leftarrow 1, 2, ..., n$ **do**
3: Teacher gets Learner's hypothesis class $h^{(t)}$.
4: Let

$$V_t := \{x \in \mathcal{X}_n | h^{(t)}(x) \neq h^*(x)\}$$
$$S_t := \mathcal{X} \backslash V_t$$

5: **if** $V_t = \emptyset$ **then**
6: record $k = i$ and terminate the teaching.
7: **end if**
8: Randomly pick $x^{(t)} \in V_t$, and teach learner with $(x^{(t)}, h^*(x^{(t)}))$.
9: **end for**

Summary

Summary

- We propose three different setting for teaching with partial knowledge by introducing interaction.
- In general it's not possible to do anything without proper constraint on the problem structure.
- Some relationship between active learning and teaching?
- Discussion
    - Proof of "Active Teaching" is not completed.
    - What kind of interaction is actually allowed to make sense in practice.

Goldman, Sally A and Kearns, Michael J
On the complexity of teaching.
Journal of Computer and System Sciences, 50(1):20–21,, 1995.

Cohn, David and Atlas, Les and Ladner, Richard
Improving generalization with active learning
Machine learning, 15(2):201–221, 1994.

Regret Bound of multi-arm bandits

Yiyang Li yl3789

December 11, 2017

Overview

1. Introduction
   - Problem
   - Former work
2. Algorithm
   - Settings
   - Algorithm
3. Conclusion

Introduction

As for the problem of online decision-making problem in full-feedback setting, we can reach the regret bound of $O\left(\sqrt{L_* logN} + logN\right)$ where N denotes the number of experts, and L_* is the minimum total loss of the experts throughout T rounds. We can see this result from the point of probabilistic. Since the variance of the loss of an expert will be bound by its total loss, the square root here can represent the upper bound for the variance.

The problem is how to do this with partial feedback (i.e., bandit setting). There are some algorithms known for the case where there are no experts (or equivalently, there are exactly K constant experts, one per possible action). So that might be a good starting point to consider.

Introduction

The problem is how to do this with partial feedback (i.e., bandit setting). There are some algorithms known for the case where there are no experts (or equivalently, there are exactly K constant experts, one per possible action). So that might be a good starting point to consider.
In the Allenberg et al., 2006, they brought up an idea of the Green clipping trick. By ignoring low-probability action, they somehow managed to reduce the regret bound to $O(L_*^{2/3} poly(K, ln(N/\delta)))$ with probability of $1-\delta$.

Green Algorithm

Algorithm GREEN

Let $\eta_1, \eta_2, \ldots > 0$, $\varepsilon_1, \varepsilon_2, \ldots > 0$ and $\gamma_1, \gamma_2, \ldots \geq 0$.

Initialization: $\widetilde{L}_{i,0} = 0$ for all $i = 1, \ldots, N$.

For each round $t = 1, 2, \ldots$

(1) Calculate the probability distribution

$$p_{i,t} = \frac{e^{-\eta_t \widetilde{L}_{i,t-1}}}{\sum_{i=1}^N e^{-\eta_t \widetilde{L}_{i,t-1}}} \qquad i = 1, \ldots, N \,.$$

(2) Calculate the modified probabilities

$$\widetilde{p}_{i,t} = \begin{cases} 0 & \text{if } p_{i,t} < \gamma_t, \\ c_t \cdot p_{i,t} & \text{if } p_{i,t} \geq \gamma_t, \end{cases}$$

where $c_t = 1 / \sum_{p_{i,t} \geq \gamma_t} p_{i,t}$.

(3) Select an action $I_t \in \{1, \ldots, N\}$ according to $\widetilde{\mathbf{p}}_\mathbf{t} = (\widetilde{p}_{1,t}, \ldots, \widetilde{p}_{N,t})$.

(4) Draw a Bernoulli random variable S_t such that $\mathbb{P}(S_t = 1) = \varepsilon_t$.

(5) Compute the estimated loss for all $i = 1, \ldots, N$

$$\widetilde{\ell}_{i,t} = \begin{cases} \frac{\ell_{i,t}}{\widetilde{p}_{i,t}\varepsilon_t} & \text{if } I_t = i \text{ and } S_t = 1; \\ 0 & \textit{otherwise}. \end{cases}$$

(6) For all $i = 1, \ldots, N$ update the cumulative estimated loss

$$\widetilde{L}_{i,t} = \widetilde{L}_{i,t-1} + \widetilde{\ell}_{i,t}.$$

Figure: Algorithm *Green*

Yiyang Li
yl3789

Setting

Assume we also have $|\mathcal{K}| = K$ actions and $N = \binom{K}{k}$ experts. Here $K > k > 0$ is the number of actions that each expert will choose in a uniform distribution. And every expert has a unique combination of actions choose. For convenience, we assume that each action has a fixed loss, and
$l_1 < l_2 < \cdots < l_{K-1} < l_K$.
Thinking of $K = 3$ and $k = 2$, then there will be $\binom{3}{2} = 3$ experts. And expert 1 will always give the advice vector like

$$\xi^1 = \left(\frac{1}{2}, \frac{1}{2}, 0\right)$$

Similarly, expert 2 and 3 will have

$$\xi^2 = \left(\frac{1}{2}, 0, \frac{1}{2}\right), \xi^3 = \left(0, \frac{1}{2}, \frac{1}{2}\right)$$

Definitions

Expert Similarity

$$ES(i,j) = \frac{1}{T}\sum_{t=1}^{T} \|\xi_t^i - \xi_t^j\|_1.$$

Minimum Loss differnece

$$dl_{min} = \min_{i \neq j} |l_i - l_j|.$$

Exp4.L Algorithm

Algorithm 2 Exp4.L

Require: K actions, N experts, $\xi^1, \xi^2, \ldots, \xi^N$ advice vectors.

Set all weight&modified weight = 1
for $t = 1, 2, \ldots, T$ **do**
  Calculate probability distribution P using modified weights

$$p_{j,t} = \frac{\sum_{i=1}^{N} \widetilde{w}_{i,t} \cdot \xi^i_{j,t}}{\widetilde{W}_t}, \text{ where } \widetilde{W}_t = \sum_{i=1}^{N} \widetilde{W}_{i,t}$$

  Draw the action $i_t \sim P$
  Receive the loss $l_{i_t,t} \in [0,1]$
  for $j = 1, 2, \ldots, K$ **do**
    Set

$$\hat{l}_{j,t} = \begin{cases} \frac{l_{j,t}}{p_{j,t}} & \text{, if } j = i_t \\ 0 & \text{,o.w.} \end{cases}$$

  for $i = 1, 2, \ldots, N$ **do**
    update the weights as

$$w_{i,t+1} = w_{i,t} \exp\left(-\eta \xi^i \cdot \hat{l}_t\right)$$

  Find the best expert $a_{t+1} = \arg\max_i w_{i,t+1}$
  Calculate the modified weight with population parameter $d_t \in \{0, 1, 2, \ldots, k\}$

$$\widetilde{w}_{i,t+1} = \begin{cases} w_{i,t+1} & \text{,if } ES(i, a_{t+1}) \geq d_t \\ 0 & \text{, o.w.} \end{cases}$$

Conclusion

Yiyang Li
yl3789

Exp4 Regret Bound

$$L_{Exp4} - L_{min} \leq \frac{\ln N}{\eta} + \frac{\eta N T}{4k^2} \leq \frac{\sqrt{TN \ln N}}{k}.$$

Exp4.L Regret Bound

$$L_{Exp4.L} - L_{min} \leq \frac{\ln K - \sqrt{\ln K^2 - 2dl_{min}^2(\ln T + \ln \frac{1}{\delta} + \ln K)}}{dl_{min}^2} \cdot \frac{(\ln \frac{1}{\delta} + d \ln K)}{\epsilon^2}$$

The End

Relative Power of Disagreement- and Diameter-Based Active Learning

Xingyu Niu (xn2126) and Geelon So (ags2191)

December 11, 2017 (COMS 6998-4)

Goal of Presentation

- Diameter $\leq$ Disagreement

Goal of Presentation

- Diameter $\leq$ Disagreement
- Diameter $\stackrel{?}{\equiv}$ Disagreement

Goal of Presentation

- Diameter $\leq$ Disagreement
- Diameter $\stackrel{?}{\equiv}$ Disagreement
- Combining disagreement and diameter: SEARLIT

Setting

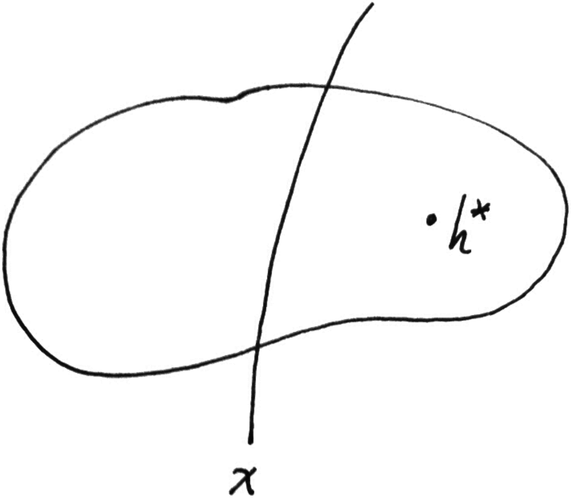


Figure 1: The hypothesis class $\mathcal{H}$ is being split by some $x \in \mathcal{X}$.

Disagreement

Definition

Let $S \subset \mathcal{H}$ be a (countable) collection of hypotheses. Their **region of disagreement** is the set of points

$$\mathrm{DIS}(S) := \{x \in \mathcal{X} : h(x) \neq h'(x)\}.$$

Disagreement

Definition

Let $S \subset \mathcal{H}$ be a (countable) collection of hypotheses. Their **region of disagreement** is the set of points

$$\mathrm{DIS}(S) := \{x \in \mathcal{X} : h(x) \neq h'(x)\}.$$

Definition

Let the **disagreement** of S be the size:

$$\mathrm{dis}(S) := \mathbb{P}\big[\mathrm{DIS}(S)\big].$$

Diameter

Definition

The **standard distance** on $\mathcal{H}$ is:

$$\begin{aligned} d(h, h') &:= \mathbb{P}\big[h(x) \neq h'(x)\big] \\ &= \int_{\mathcal{X}} |h - h'| \,\mathrm{d}\mu. \end{aligned}$$

Diameter

Definition

The **standard distance** on $\mathcal{H}$ is:

$$d(h, h') := \mathbb{P}\big[h(x) \neq h'(x)\big] = \int_{\mathcal{X}} |h - h'| \, \mathrm{d}\mu.$$

Definition

The **diameter** of S is:

$$\mathrm{diam}(S) := \sup_{h, h' \in S} d(h, h').$$

Diameter $\leq$ Disagreement

$$\operatorname{diam}(S) = \sup_{h,h' \in S} \int_{\mathcal{X}} |h - h'| \, \mathrm{d}\mu$$

Diameter $\leq$ Disagreement

$$\begin{aligned}
\operatorname{diam}(S) &= \sup_{h,h' \in S} \int_{\mathcal{X}} |h - h'| \, \mathrm{d}\mu \\
&= \sup_{h,h' \in S} \int_{\mathrm{DIS}(S)} |h - h'| \, \mathrm{d}\mu
\end{aligned}$$

Diameter $\leq$ Disagreement

$$\begin{aligned}\mathrm{diam}(S) &= \sup_{h,h' \in S} \int_{\mathcal{X}} |h - h'| \, \mathrm{d}\mu \\ &= \sup_{h,h' \in S} \int_{\mathrm{DIS}(S)} |h - h'| \, \mathrm{d}\mu \ \leq \ \int_{\mathrm{DIS}(S)} \mathrm{d}\mu \ = \ \mathrm{dis}(S).\end{aligned}$$

Relative Power

"Corollary"

Any disagreement-reducing algorithm is also a diameter-reducing algorithm.

No Reverse Implication

Example

For interval classifiers on $[0, 1]$, the disagreement region of the ball $B(0, \epsilon)$ is $\mathcal{X}$.

Proof. Just consider the collection of classifiers

$$h = \mathbb{1}_{[\alpha, \alpha+\epsilon/2]}.$$

Equivalence of Norms

We say that two norms $||\cdot||_1$ and $||\cdot||_2$ are **equivalent** if there exists $c, C > 0$ such that

$$c||\cdot||_1 \leq ||\cdot||_2 \leq C||\cdot||_2.$$

Equivalence of Diameter and Disagreement

In the spirit of the equivalence of norms, when is:

$$\mathrm{diam}(S) \leq \mathrm{dis}(S) \leq C \cdot \mathrm{diam}(S).$$

Bounded Ratio

Diameter and disagreement are equivalent when

$$\sup_{S \subset \mathcal{H}} \frac{\mathrm{dis}(S)}{\mathrm{diam}(S)} < \infty.$$

Disagreement Coefficient

Definition

The **disagreement coefficient** of $\mathcal{H}$ is

$$\theta_{\mathcal{H}} := \sup_{h \in \mathcal{H}} \sup_{r>0} \frac{\mathrm{dis}(B(h, r))}{r}.$$

Disagreement Coefficient

Definition

The **disagreement coefficient** of $\mathcal{H}$ is

$$\theta_{\mathcal{H}} := \sup_{h \in \mathcal{H}} \sup_{r > 0} \frac{\mathrm{dis}(B(h, r))}{r}.$$

Note: a finite disagreement coefficient is equivalent to a bounded disagreement-diameter ratio.

Relative Power II

“Corollary”

If $\theta_{\mathcal{H}} < \infty$, then any diameter-reducing algorithm can be made into a disagreement-reducing algorithm.

Relative Power III

Does not imply that diameter-based methods are weaker.

Relative Power III

Does not imply that diameter-based methods are weaker.

- In particular, consider the *splitting index*.

Splitting Index

Definition

We say that $\mathcal{H}$ is $(\boldsymbol{\rho}, \boldsymbol{\epsilon}, \boldsymbol{\tau})$**-splittable** if for all finite edge sets $Q \subset \binom{\mathcal{H}}{2}$ of length greater than ϵ,

$$\mathbb{P}\left[x\ \rho\text{-splits}\ Q_\epsilon\right] \geq \tau.$$

Splitting Index Intuition

Fix ρ and $h \in \mathcal{H}$.

- Consider the ϵ-sphere $S(h, \epsilon)$.

Splitting Index Intuition

Fix ρ and $h \in \mathcal{H}$.

- Consider the ϵ-sphere $S(h, \epsilon)$.
- Union $\{h\}$ with any finite collection of h_i's in $S(h, \epsilon)$,

$$S := \{h\} \cup \{h_1, \ldots, h_n\}.$$

Splitting Index Intuition

Fix ρ and $h \in \mathcal{H}$.

- Consider the ϵ-sphere $S(h, \epsilon)$.
- Union $\{h\}$ with any finite collection of h_i's in $S(h, \epsilon)$,

$$S := \{h\} \cup \{h_1, \ldots, h_n\}.$$

- Heuristically, the ρ-agreement of S is at least τ:

$$\mathrm{agr}_\rho(S) = \mathbb{P}\big[h(x) \text{ agree for on } \rho\text{-fraction of } S\big] \geq \tau.$$

Analogy to Equivalence?

If "apartness" of S from h is ϵ and $\rho > 0$, then:

$$\mathrm{agr}_\rho(S) \leq \mathrm{apart}_h(S).$$

Analogy to Equivalence?

If "apartness" of S from h is ϵ and $\rho > 0$, then:

$$\mathrm{agr}_\rho(S) \leq \mathrm{apart}_h(S).$$

We might consider the analogy:

$$\mathrm{agr}_\rho(S) \leq \mathrm{apart}_h(S) \leq C \cdot \mathrm{agr}_\rho(S).$$

Open Question

Does the 'ρ-agreement-h-apartness' equivalence imply lower bounds on agreement region?

Open Question

Does the 'ρ-agreement-h-apartness' equivalence imply lower bounds on agreement region?

- Implies disagreement method from splitting index.

Combining Methods

For now, it seems that combined method is stronger.

Recall LARCH

Input: $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots$, where $h^* \in \mathcal{H}_{k^*}$.

1. Use CAL to bound diameter of version space

Recall LARCH

Input: $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots$, where $h^* \in \mathcal{H}_{k^*}$.

1. Use CAL to bound diameter of version space
2. Use disagreement coefficient to bound disagreement

Recall LARCH

Input: $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots$, where $h^* \in \mathcal{H}_{k^*}$.

1. Use CAL to bound diameter of version space
2. Use disagreement coefficient to bound disagreement
3. Use SEARCH oracle to efficiently traverse $\mathcal{H}_i$'s

SEARLIT (or, SPLEARCH)

Input: $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots$, where $h^* \in \mathcal{H}_{k^*}$.

1. Use SPLIT to bound diameter of version space

SEARLIT (or, SPLEARCH)

Input: $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots$, where $h^* \in \mathcal{H}_{k^*}$.

1. Use SPLIT to bound diameter of version space
2. Use disagreement coefficient to bound disagreement

SEARLIT (or, SPLEARCH)

Input: $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots$, where $h^* \in \mathcal{H}_{k^*}$.

1. Use SPLIT to bound diameter of version space
2. Use disagreement coefficient to bound disagreement
3. Use SEARCH oracle to efficiently traverse $\mathcal{H}_i$'s

Label Complexity

- LARCH:

$$O\left(\left(k^* + \log\frac{1}{\epsilon}\right) \cdot d_{k^*} \cdot \sup_{k \leq k^*} \theta_k \cdot \log^2 \frac{1}{\epsilon}\right).$$

- SEARLIT (or, SPLABEARCH):

$$O\left(\frac{k^*}{\rho} \cdot d_{k^*} \cdot \log \frac{\sup_{k<k^*} \theta_k}{\epsilon}\right)$$

Sample Complexity

- LARCH:

$$O\left(\frac{k^* \cdot d_{k^*}}{\epsilon}\right).$$

- SPLEARCH (or, SEARPLITEL):

$$O\left(\frac{k^* \cdot d_{k^*}}{\rho\tau} \cdot \log \frac{\sup_{k<k^*} \theta_k}{\epsilon} \cdot \log \frac{d_{k^*}}{\rho\delta}\right).$$

References

(B2007) Balcan, Maria-Florina, et al. "Asymptotic active learning." Workshop on Principles of Learning Design Problem. 2007.

(BH2016) Beygelzimer, Alina, et al. "Search improves label for active learning." Advances in Neural Information Processing Systems. 2016.

(CAL1994) Cohn, David, Les Atlas, and Richard Ladner. "Improving generalization with active learning." Machine learning 15.2 (1994): 201-221.

(D2006) Dasgupta, Sanjoy. "Coarse sample complexity bounds for active learning" Advances in Neural Information Processing Systems. 2006.

(H2014) Hanneke, Steve. "Theory of Active Learning." 2014.

(H2010) Hsu, Daniel J. "Algorithms for active learning." 2010.

(HD1992) Haussler, David. "Decision theoretic generalizations of the PAC model for neural net and other learning applications." Information and computation 100.1 (1992): 78-150.

An Efficient and General Interactive Clustering Algorithm

Kiran Vodrahalli
knv2109@columbia.edu

Columbia University

December 11, 2017

The Problem

Interactive Clustering

Interactive clustering organizes data points into clusters with the help of a (human?) oracle

- We focus on split-merge oracles [Balcan & Blum '08]
  - split: the cluster is impure
  - merge: merge two pure clusters
- Clusters parametrized by concept classes (rectangles, logical formula, etc.)
- Give guarantees in terms of query complexity
  - general algorithm presented by [Balcan & Blum '08] gets $\mathcal{O}(k^3 \log |\mathcal{C}|)$
  - k: $\#$ of clusters in a clustering
  - $\mathcal{C}$: concept class
- Also care about computational complexity
  - Algorithm by [Balcan & Blum '08] is intractable

Original Algorithm

due to [Balcan & Blum '08]

Key idea: Remove significant chunk of the clustering version space with each query

Definition

A cluster c is α-consistent if an α-fraction of the clusterings in the version space contain a cluster which contains c.

Progress guarantee (regardless of split/merge-response)

- cluster is at least α-consistent
- cluster is at most $(1-\alpha)$-consistent

Feedback response

- split(c_i): remove clusterings inconsistent with c_i from version space
- merge(c_i, c_j): remove clusterings inconsistent with $c_i \cup c_j$ from version space

Key idea: Replace version space operations with a cluster sampler

Lemma

α-consistency over clusters $\implies$ α-consistency over k-clusterings.

Proof.

$1 - (1-\alpha)^k \geq \alpha$ $\square$

Lemma

$\mathcal{O}\left(\frac{1}{\epsilon^2}\log(\frac{1}{\delta})\right)$ samples are required to approximate frequency of a Bernoulli r.v. with error $\leq \epsilon$ with probability $\geq 1-\delta$.

Proof.

Let $Y = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}[\text{cluster } i \text{ consistent with } c]$ for a cluster c. By Hoeffding, $\mathbb{P}\left\{|Y - \hat{Y}| > \epsilon\right\} \leq 2e^{-2n\epsilon^2}$. $\square$

New Algorithm

cluster sampling distribution

How to sample with respect to constraints:

$$\mathbb{P}\left\{c|E_{\text{feedback}}, X\right\} = \frac{\mathbb{P}\left\{E_{\text{feedback}}|c, X\right\}\mathbb{P}\left\{c|X\right\}}{Z} \tag{1}$$

where X is the data, E_{feedback} is merge-split feedback, Z normalizes.
To-do:

1. Determine whether calculating $\mathbb{P}\left\{E_{\text{feedback}}|c, X\right\}$ and the normalization constant is tractable
2. How to efficiently update?
3. How to sample?

New Algorithm

efficiency conditions

Two assumptions for efficient computation:

1. $\mathcal{C}$ is optimizable: Can efficiently find $\bigcap_{c:Q\subseteq c, c\in\mathcal{C}} c$ for query cluster $Q \subseteq X$.
2. $\mathcal{C}$ is intersectable: Can efficiently calculate $c_i \cap c_j$ for $c_i, c_j \in \mathcal{C}$.

Definition

Define a validity function $\mathbb{V}$ given query Q and concept c. Let $\hat{c}_Q$ be the result of optimization to find smallest $c \in \mathcal{C}$ containing Q. Let α_Q be the response of a merge-split query with respect to Q. Then, $\mathbb{V}_{\alpha_Q}(\hat{c}_Q, c)$ tells us if c is a valid cluster to sample from given query Q and its feedback.

- Calculate using $\mu_v(\hat{c}_Q, c) := \frac{|\hat{c}_Q \cap c|}{|\hat{c}_Q|}$
- Generalizes to multiple queries: $\mathbb{V}(c) := \left\lfloor \frac{1}{N} \sum_{i=1}^{N} \mathbb{V}_{\alpha_{Q_i}}(\hat{c}_{Q_i}, c) \right\rfloor$
- Requires remembering $\{(\hat{c}_{Q_i}, \alpha_{Q_i})\}_{i=1}^{N}$

New Algorithm

sampling algorithm

We can sample from this distribution using [Kim, Sabharwal, Ermon '16].

- Difficulty in sampling due to calculation of normalization Z.
- Sampling = Optimization: Use the Gumbel-max noise trick to turn sampling into an optimization problem.
- Integer linear program relaxation
- Can use optimized-heuristic solvers to solve in practice.

Implications

rectangular concept classes

This algorithm yields an efficient algorithm for $\mathcal{C} = d$-dimensional rectangles.

- Query complexity of our algorithm: $\mathcal{O}(k^3 d \log m)$
- Beats [Awasthi & Zadeh '10]: They have query complexity $\mathcal{O}((kd \log m)^d)$
- $k = \#$ of clusters, $m = \#$ data points.

Why?

1. Can easily optimize over rectangles in time $\mathcal{O}(md)$.
2. Can easily calculate $\mu_v(\hat{c}_Q, c) = \prod_{i=1}^{d} \frac{\mathbf{1}[d_i > b_i, w_i < b_i](b_i - w_i) + \mathbf{1}[z_i < b_i, a_i < z_i](z_i - a_i)}{b_i - a_i}$

Thank you for your attention!

Project: Finite Sample Analysis for Leave-one-out Approximation

Ji Xu

Dec 9th, 2017

Model

Model:

- Training data are $(\boldsymbol{x}_{i\cdot}, y_i)_{i=1,\cdots n} \in \mathbb{R}^d \times \mathbb{R}$ where $(\boldsymbol{x}_{i\cdot}, y_i)$ are i.i.d following distribution of $D(\boldsymbol{x}, y)$ with $\mathbb{E}\boldsymbol{x} = \boldsymbol{0}$.
- Consider the following optimization problem

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \frac{p}{n} \sum_{i=1}^{n} l(\boldsymbol{x}_{i\cdot}^\top \boldsymbol{\beta}, y_i) + \lambda R(\boldsymbol{\beta}), \tag{1}$$

where p is a tuning parameter such that $\mathbb{E}x_{ij}^2 = O(1/p)$.

Model

Model:

- Training data are $(\boldsymbol{x}_{i\cdot}, y_i)_{i=1,\cdots n} \in \mathbb{R}^d \times \mathbb{R}$ where $(\boldsymbol{x}_{i\cdot}, y_i)$ are i.i.d following distribution of $D(\boldsymbol{x}, y)$ with $\mathbb{E}\boldsymbol{x} = \boldsymbol{0}$.
- Consider the following optimization problem

$$\hat{\beta} = \arg\min_{\beta} \frac{p}{n} \sum_{i=1}^{n} l(\boldsymbol{x}_{i\cdot}^\top \beta, y_i) + \lambda R(\beta), \tag{1}$$

where p is a tuning parameter such that $\mathbb{E}x_{ij}^2 = O(1/p)$.

Reason for introducing p:

- it connects both low dimension regime $d = o(n)$ and high dimension regime $d = \Omega(n)$.
- it bounds the norm of the estimates with high probability.

Leave One Out Approximation:

Let $\tilde{\boldsymbol{\beta}}^{\setminus i}$ denotes the leave-ith-out estimate. In Koh & Liang's work, the approximation is

$$\tilde{\boldsymbol{\beta}}^{\setminus i} \approx \hat{\boldsymbol{\beta}} + \frac{p}{n} H^{-1} \nabla_{\boldsymbol{\beta}} l(\boldsymbol{x}_{i\cdot}^{\top} \hat{\boldsymbol{\beta}}, y_i), \tag{2}$$

where $H := \frac{p}{n} \sum_{j=1}^{n} \nabla_{\boldsymbol{\beta}}^2 l(\boldsymbol{x}_{i\cdot}^{\top} \hat{\boldsymbol{\beta}}, y_i) + \lambda \nabla^2 R(\hat{\boldsymbol{\beta}})$.

Leave One Out Approximation:

Let $\tilde{\boldsymbol{\beta}}^{\setminus i}$ denotes the leave-ith-out estimate. In Koh & Liang's work, the approximation is

$$\tilde{\boldsymbol{\beta}}^{\setminus i} \approx \hat{\boldsymbol{\beta}} + \frac{p}{n} H^{-1} \nabla_{\boldsymbol{\beta}} l(\boldsymbol{x}_{i\cdot}^{\top} \hat{\boldsymbol{\beta}}, y_i), \tag{2}$$

where $H := \frac{p}{n} \sum_{j=1}^{n} \nabla_{\boldsymbol{\beta}}^2 l(\boldsymbol{x}_{i\cdot}^{\top} \hat{\boldsymbol{\beta}}, y_i) + \lambda \nabla^2 R(\hat{\boldsymbol{\beta}})$.
However, (2) is only accurate for $d = o(n)$. More accurate approximation is

$$\tilde{\boldsymbol{\beta}}^{\setminus i} \approx \hat{\boldsymbol{\beta}} - \frac{\frac{p}{n} H^{-1} \nabla_{\boldsymbol{\beta}} l(\boldsymbol{x}_{i\cdot}^{\top} \hat{\boldsymbol{\beta}}, y_i)}{1 - \boldsymbol{x}_{i\cdot}^{\top} H^{-1} \boldsymbol{x}_{i\cdot} \cdot \frac{p}{n} l''(\boldsymbol{x}_{i\cdot}^{\top} \hat{\boldsymbol{\beta}}, y_i)}. \tag{3}$$

Main Assumptions

- Loss function l and regularizer R are smooth enough. e.g. $l'(u, y)$ and $l''(u, y)$ are pseudo-Lipchitz and R'' is Lipchitz.

Main Assumptions

- Loss function l and regularizer R are smooth enough. e.g. $l'(u, y)$ and $l''(u, y)$ are pseudo-Lipchitz and R'' is Lipchitz.

- Assume $\boldsymbol{x}$ is either has independent components with sub-Gaussian tail or $\boldsymbol{x} \sim N(0, \Sigma)$ where maximal eigenvalue of Σ is $O(1/p)$. Assume $\max_i |y_i| = O(n^{\alpha_y})$.

Main Assumptions

- Loss function l and regularizer R are smooth enough. e.g. $l'(u, y)$ and $l''(u, y)$ are pseudo-Lipchitz and R'' is Lipchitz.
- Assume $\boldsymbol{x}$ is either has independent components with sub-Gaussian tail or $\boldsymbol{x} \sim N(0, \Sigma)$ where maximal eigenvalue of Σ is $O(1/p)$. Assume $\max_i |y_i| = O(n^{\alpha_y})$.
- Suitable (λ, d, n) such that H^{-1} exists.

Examples

- Ridge regression: the approximation is exact for any choice of (d, n, p, λ) with $\lambda > 0$.

Examples

- Ridge regression: the approximation is exact for any choice of (d, n, p, λ) with $\lambda > 0$.
- Logistic regression with $R = \|\cdot\|_2^2$:

  Let $p = 1$ and $\lambda = \Omega\left(\frac{\sqrt{d}}{n^{\frac{5}{6}-\epsilon}} \cdot \max\left(1, \left(\frac{d}{n}\right)^{\frac{1}{3}}\right)\right), \forall \epsilon > 0,$

  then

$$\sup_i \|\Delta_i\| \quad \leq \quad \begin{cases} \tilde{O}\left(\frac{d}{\lambda^3 n^2} + \frac{d^2}{\lambda^5 n^{3.5}}\right), & d \leq O(n) \\ \tilde{O}\left(\frac{d^2}{\lambda^3 n^3} + \frac{d^{3.5}}{\lambda^5 n^5}\right), & \Omega(n) \leq d \leq O(n^{1.4-\epsilon}) \end{cases},$$

  with probability at least $1 - o(1)$.

Examples

- Poisson regression regression with $R = \|\cdot\|_2^2$:
  Let $p = \left(\frac{d}{\lambda}\right)^2$, $d \leq O(n^{0.5-\alpha_y})$ and $\lambda \geq \Omega(\frac{d}{n^{0.5-\alpha_y}})$, then

$$\sup_i \|\Delta_i\| \quad \leq \quad \tilde{O}\left(\frac{d^2}{\lambda^4 n^{2-3\alpha_y}}\right),$$

  with probability at least $1 - o(1) - \tilde{O}\left(\frac{d^2}{\lambda^2 n}\right)$.

Teaching with Partial Knowledge

Jiefu Ying, Jinyi Zhang

December 11, 2017

Outline

Project Summary

- Setting
- Key Points
- Inheritance Model & School Model

Application details

- Threshold Function
- Monotone Monomial

Setting

- Target : to teacher a specific concept h^* to the learner.
- The version space of teacher and learner, V_T and V_L, are no longer identical. Instead, we have $V_T \supset V_L$, where $|V_T| \gg |V_L|$.
- Consider the realizable case, $h^* \in V_L$.
- A consistent learner.

Key Points

- Explore V_T to locate V_L

  - Quiz

- Correct wrong predictions

  - Tradeoffs

- Stop condition / learning status is not clear

  - Guaranteed way: reduce V_T to $\{h^*\}$

  - Interactive way: extended quiz

Inheritance Model

- Stop condition: completely sure that $V_L \to \{h^*\}$
- Operations:

  - extended quiz

- Selection function adaptive to V_T and learning status

Algorithm 1 Deterministic teaching algorithm for inheritance model - extended quiz

Initialize $\mathcal{V}_T := \mathcal{H}_T,\ \mathcal{V}_L := \mathcal{H}_L,\ m_0 = |\mathcal{H}_T|,\ n_0 = 0, \mathcal{D} = \{\}$.

for t = 1, 2, ... : **do**

  Set $(x_t, y_t) = E(\mathcal{V}_T, m_{t-1})$.

  Quiz the learner with x_t, who returns m_t correct answers, and n_t wrong answers.

  if $n_t > 0$ **then**

    Update $\mathcal{V}_L$ using (x_t, y_t), $\mathcal{D} = \mathcal{D} \cup \{(x_t, y_t)\}$.

  end if

  if $m_t = 1$ **then**

    Break.

  end if

  Update $\mathcal{V}_T$ using (x_t, y_t).

end for

return $\mathcal{D}$.

School Model

- Intuitions from school education
- Stop condition:

  - run out of budget / good enough
- Operations:

  - normal quiz & black box test
- Randomized behavior adaptive to V_T, learning status, and test

  - $R(D)$, $R(T)$ and ε

  - $k = \frac{R(D)+\log(\varepsilon)R(T)+\gamma}{(1+\log(\varepsilon))+\gamma}$, $\alpha = 1 - k^{\eta_1}$, $\beta = k^{\eta_2}$

School Model

Algorithm 2 Randomized teaching algorithm for school model

Input: A teaching sequence $s(h^*, \mathcal{H}_T) = \{x^{(1)}, y^{(1)}), ..., (x^{(|s(h^*,\mathcal{H}_T)|)}, y^{(|s(h^*,\mathcal{H}_T)|)}\}$, target error $\epsilon \in (0, 1)$, teacher's hypothesis class $\mathcal{H}_T$, teaching budge B, and parameters γ, η_1, η_2.

Initialize $\mathcal{V}_L := \mathcal{H}_L$, cost $c := 0$, $j := 0$, $\mathcal{D} = \{\}$, $T = \{\}$.

```
while c < B do
    Let k = (R(T) − log_10(ε)R(D) + γ) / ((1 − log_10(ε))|H_T| + γ), β = k^{η_2}.
    Toss a coin with head probability β.
    if Head then
        Conduct the test operation using 1/ε data points on V_L.
        j = j + 1.
        if the learner passes the test then
            Break.
        end if
    end if
    Pick an example (x, y) ∈ s(h*, H_T) that has not been used for quizzing or teaching.
    Quiz the learner using x, who returns q(x) as the answer.
    if q(x) == h*(x) then
        Let α = 1 − k^{η_1}.
        Update V_L using (x, y), D = D ∪ {(x, y)} with probability α.
    else
        Update V_L using (x, y), D = D ∪ {(x, y)}.
    end if
    T = T ∪ {(x, y)}.
    Update c in terms of j, |T|, and |D|.
end while
return D
```

Threshold Function

- $X = \{1, 2, \dots, 100\}$
- $H_T = \{h_n : n = 1,2, \dots, 100; h_n(x) = 1\ if\ x > n\ else\ 0\}$
- $h^* = h_{50}$
- $s(h^*, H_T) = \{(51, 1), (50, 0)\}$
- $H_L = \{n_1, n_2, 50, n_4, n_5\}$
- $\eta_1 = \eta_2 = 2, \epsilon = 0.01$
- Adversary pick: 74.3% probability to reduce H_L to $\{h^*\}$
- Random pick: 82.7% probability

School Model Results

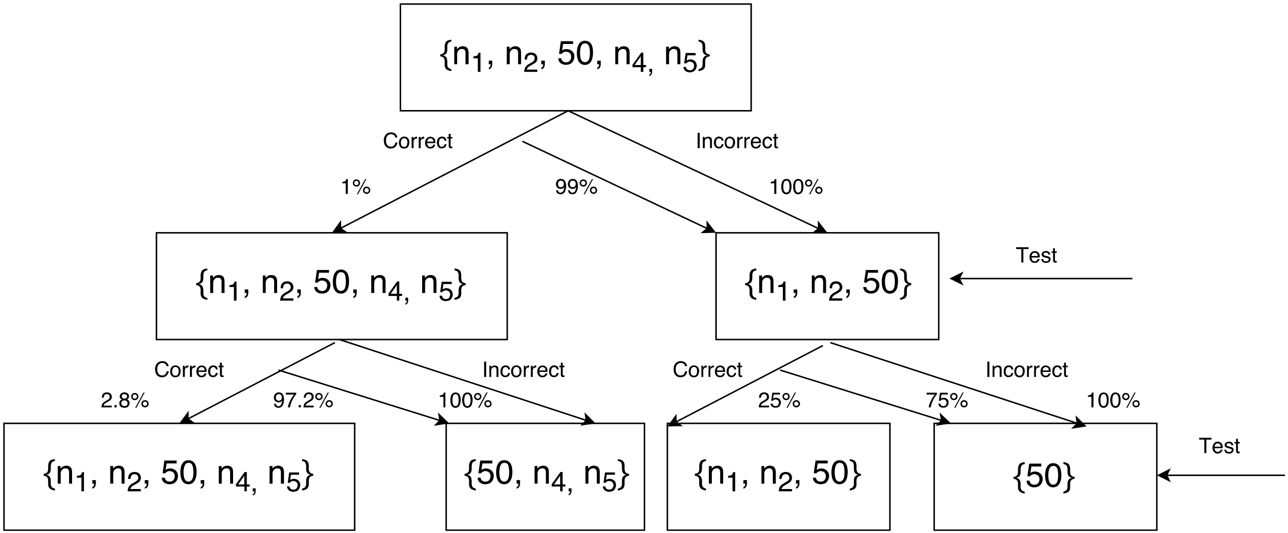


Figure 1. At the first round, the teacher quizzes the example (51, 1). If the learner makes an incorrect prediction, the teacher teaches the example; otherwise, the teacher has a probability of 99% to teach the example. At the second round, teacher quizzes the example (50, 0). If the learner makes a correct prediction, the teacher has a probability of 97.2% to teach the example if she didn't teach the first example; if she has taught the first example, the teacher has a probability of 75% to teach the second example.

Monotone Monomial

- n variables, r variables in h^*
  - Let $n = 100, r = 10$
- $TD(h^*, H_T) = \min(r+1, n)$
  - One positive example and r negative examples generated by flipping each relevant bit of the positive example one at a time
- Using Inheritance Model
  - Use the positive example first.
  - Same probability return each negative example

Q & A

Splitting Index Bounds for Decision Trees

Che Shen, Yuemei Zhang

December 11, 2017

Problem Statement

We focus on splitting index bounds for decision trees under uniform distribution on $[0,1]^d$ with a given size s.

Decision trees

- We define the *size* of the tree to be the number of leaves.
- A tree of size s partitions the domain $[0,1]^d$ into s axis aligned rectangles.

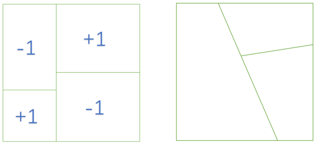


Figure: **Left:** a decision tree with $d = 2, s = 4$. **Right:** an invalid decision tree.

Recall: splitting index

Splitting index

- A point x is said to *ρ-split* the edge-set $Q \subset \binom{\mathcal{H}}{2}$ if it can eliminate at least a fraction ρ of edges. That is:

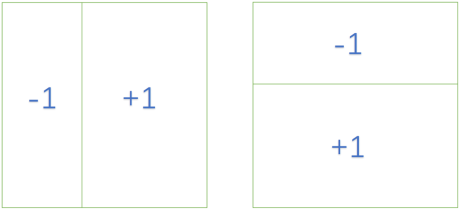
$$\max\left\{\left|Q \cap \binom{\mathcal{H}_x^+}{2}\right|, \left|Q \cap \binom{\mathcal{H}_x^-}{2}\right|\right\} \leq (1-\rho)|Q|.$$

- A subset of hypotheses $\mathcal{S} \subset \mathcal{H}$ is (ρ, ϵ, τ)-*splittable* if for any finite edge-set $Q \subset \binom{\mathcal{S}}{2}$, $\mathbb{P}\{x : x\ \rho\text{-splits } Q_\epsilon\} \geq \tau$.
- Splitting index gives bounds on sample and label complexity in active learning.

Main Results

Decision stumps over $[0,1]^d$

The hypothesis space consists of decision trees of size $s = 2$ over $[0,1]^d$ has global splitting index with $\rho = \Omega(1/d)$ and $\tau = \Omega(\epsilon)$.



Decision trees of size $s \geq 3$ over $[0,1]$

When the version space is reduced to a reasonably small size, it has splitting index with $\rho = \Omega(1/s)$ and $\tau = \Omega(\epsilon)$.

Decision Stumps over $[0,1]^d$

Claim

Given any dimension d, let $\mathcal{H}$ be decision trees of size $s = 2$ over $[0,1]^d$. Then $\mathcal{H}$ is $\left(\frac{1}{16d}, \epsilon, \frac{\epsilon}{8}\right)$-splittable.

Upper bounds

- Sample complexity: $\tilde{O}\left(\frac{d}{\epsilon} \log d \cdot \log \frac{1}{\epsilon}\right)$
- Label complexity: $\tilde{O}\left(d \log d \cdot \log \frac{1}{\epsilon}\right)$

Decision trees of size $s \geq 3$ over $[0,1]$

Unions of intervals

Let $\mathcal{H}^1_s$ denote the hypothesis space on $[0,1]$ of decision trees of size s, and $\mathcal{F}_t$ denote the hypothesis space of union of at most t intervals on $[0,1]$. Then $\mathcal{H}^1_s \subset \mathcal{F}_{[\frac{s}{2}]+1} \subset \mathcal{H}^1_{s+2}$. So it suffices to determine the splitting index of $\mathcal{F}_t$.

Claim

Let h be the union of t intervals $[a_1, a_2], \ldots, [a_{2t-1}, a_{2t}]$ on $[0,1]$. Let $p = \min_{i=1,2,\ldots,2n-1}(a_{i+1} - a_i)$. For any $\epsilon > 0$, if $p > 4\epsilon$ then $B(h, 4\epsilon)$ is $(\frac{1}{16t}, \epsilon, \frac{\epsilon}{2})$-splittable.

Figure: Union of 2 intervals, or decision tree with $s = 4$