Machine learning theory

Active learning

Hamid Beigy

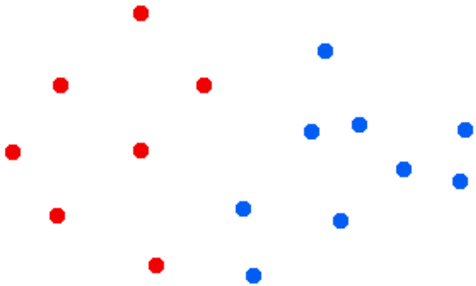
Sharif university of technology

June 13, 2020

Table of contents

Introduction

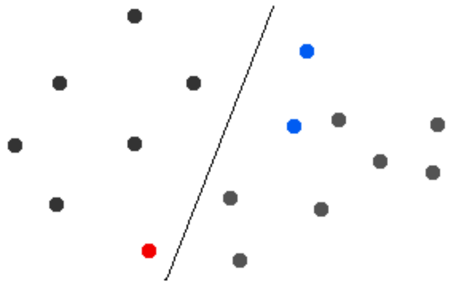
- We have studied the passive supervised learning methods.
- Given access to a labeled sample of size m (drawn iid from an unknown distribution $\mathcal{D}$), we want to learn a classifier $h \in H$ such that $\mathbf{R}(h) \leq \epsilon$ with probability higher than $(1 - \delta)$.



- We need m to be roughly $\frac{VC(H)}{\epsilon}$ in realizable case and $\frac{VC(H)}{\epsilon^2}$ in urealizable case.
- In many applications such as web-page classification, there are a lot of unlabeled examples but obtaining their labels is a costly process.

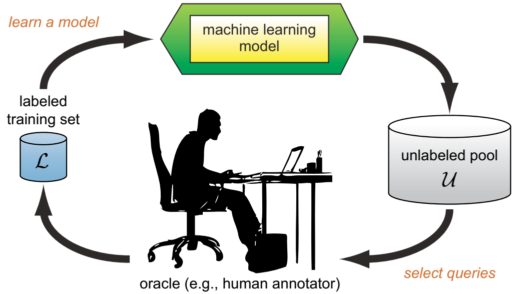
Active learning

- In many applications **unlabeled data is cheap and easy to collect**, but **labeling it is very expensive** (e.g., requires a hired human).
- Considering the problem of web page classification.
  1. A basic web crawler can very quickly collect millions of web pages, which can serve as the unlabeled pool for this learning problem.
  2. In contrast, obtaining labels typically requires a human to read the text on these pages to determine its label.
  3. Thus, the time-bottleneck in the data-gathering process is the time spent by the human labeler.
- The idea is to let the classifier/regressor pick which examples it wants labeled.

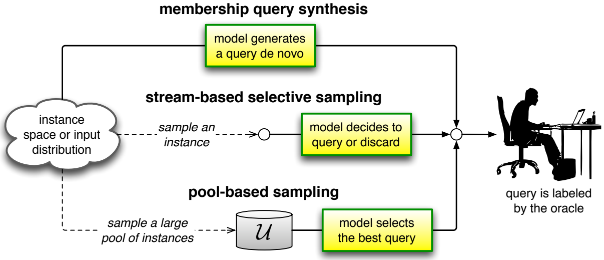


- The hope is that by directing the labeling process, we can pick a good classifier at low cost.
- It is therefore desirable to minimize the number of labels required to obtain an accurate classifier.

- In passive supervised learning setting, we have
  1. There is a set $\mathcal{X}$ called the **instance space**.
  2. There is a set $\mathcal{Y}$ called the **label space**.
  3. There is a distribution $\mathcal{D}$ called the **target distribution**.
  4. Given a training sample $S \subset \mathcal{X} \times \mathcal{Y}$, the goal is to find a classifier $h : \mathcal{X} \mapsto \mathcal{Y}$ with acceptable error rate $\mathbf{R}(h) = \underset{(\mathbf{x},y)\sim\mathcal{D}}{\mathbb{P}}[h(x) \neq y]$.
- In active learning, we have
  1. There is a set $\mathcal{X}$ called the **instance space**.
  2. There is a set $\mathcal{Y}$ called the **label space**.
  3. There is a distribution $\mathcal{D}$ called the **target distribution**.
  4. The learner have access to sample $S_X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\infty\} \subset \mathcal{X}$.
  5. There is an oracle that labels each instant $\mathbf{x}$.
  6. There is a budget m.
  7. The learner chooses an instant and gives it to the oracle and receives its label.
  8. After a number of these label requests not exceeding the budget m, the algorithm halts and returns a classifier h.



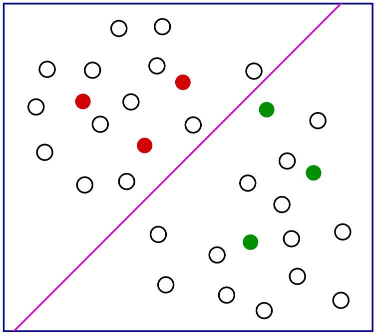
- There are three main scenarios where active learning has been studied.



- In all scenarios, at each iteration a model is fitted to the current labeled set and that model is used to decide which unlabeled example we should label next.
- In membership query synthesis, the active learner is expected to produce an example that it would like us to label.
- In stream based selective sampling, the learner gets a stream of examples from the data distribution and decides if a given instance should be labeled or not.
- In pool-based sampling, the learner has access to a large pool of unlabeled examples and chooses an example to be labeled from that pool. This scenario is most useful when gathering data is simple, but the labeling process is expensive.

Typical heuristics for active learning[5]

1: Start with a pool of unlabeled data.
2: Pick a few points at random and get their labels.
3: **repeat**
4: Fit a classifier to the labels seen so far.
5: Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty,...)
6: **until** forever

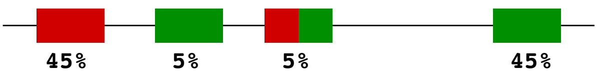


Biased sampling: the labeled points are not representative of the underlying distribution!

Typical heuristics for active learning

Start with a pool of unlabeled data.
Pick a few points at random and get their labels.
repeat
    Fit a classifier to the labels seen so far.
    Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty,...)
until forever

Example (Samplin bias)

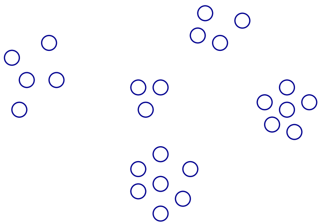


Even with infinitely many labels, converges to a classifier with 5% error instead of the best achievable, 2.5%. Not consistent!

Can adaptive querying really help?

- There are two distinct narratives for explaining how adaptive querying can help
  1. Exploiting (cluster) structure in data
  2. Efficient search through hypothesis space

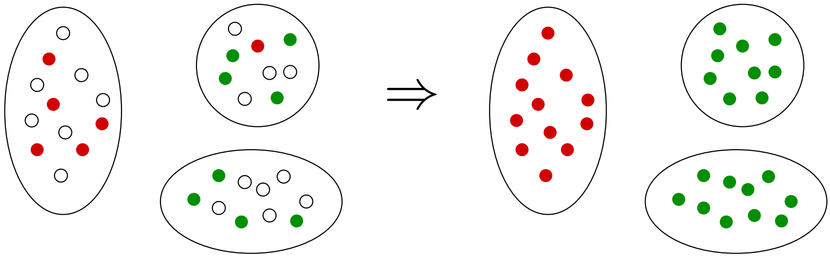
- Exploiting (cluster) structure in data
  1. Suppose the unlabeled data looks like this



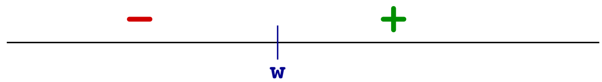
  2. Then perhaps we just need five labels!
- In general, the cluster structure has the following challenges
  1. It is not so clearly defined
  2. There exists at many levels of granularity.
- The clusters themselves might not be pure in their labels.
- How to exploit whatever structure happens to exist?

- Efficient search through hypothesis space
  1. Ideal case is when each query cuts the version space in two.
  2. Then perhaps we need just $\log|H|$ labels to get a perfect hypothesis!
- In general, the efficient search through hypothesis space has the following challenges
  1. Do there always exist queries that will cut off a good portion of the version space?
  2. If so, how can these queries be found?
  3. What happens in the non-separable case?

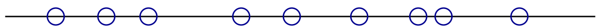
- Find a clustering of the data
- Sample a few randomly-chosen points in each cluster
- Assign each cluster its majority label
- Now use this fully labeled data set to build a classifier



- Threshold functions on the real line: $H = \{h_w \mid w \in \mathbb{R}\}$ and $h_w(x) = \mathbb{I}[x \geq w]$.



- **Passive learning:** we need $\Omega\left(\frac{1}{\epsilon}\right)$ labeled points to have $\mathbf{R}(h_w) \leq \epsilon$.
- **Active learning:** start with $\frac{1}{\epsilon}$ unlabeled points.

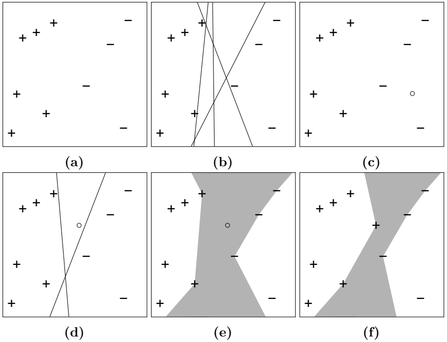


- **Binary search:** need just $\log \frac{1}{\epsilon}$ labels, from which the rest can be inferred. **Exponential improvement in label complexity!**
- **Challenges:**
  1. Nonseparable data?
  2. Other hypothesis classes?

Algorithm CAL [1]

1: Let $h : \mathcal{X} \mapsto \{-1, +1\}$ and $h^* \in H$.
2: Initialize $i = 1$ and $H_1 = H$.
3: **while** $(|H_i| > 1)$ **do**
4: Select $\mathbf{x}_i \in \{\mathbf{x} \mid h \in H_1 \text{ disagrees}\}$. ▷ Region of disagreement
5: Query with $\mathbf{x}_i$ to obtain $y_i = h^*(\mathbf{x}_i)$. ▷ Query the oracle
6: Set $H_{i+1} \leftarrow \{h \in H_i \mid h(\mathbf{x}_i) = y_i\}$. ▷ Version space
7: Set $i \leftarrow i + 1$.
8: **end while**

CAL example



Definition (Label complexity[4, 3])

Active learning algorithm A achieves label complexity m_A if, for every $\epsilon \geq 0$ and $\delta \in [0, 1]$, every distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, and every integer m higher than $m_A(\epsilon, \delta, \mathcal{D})$, if h is the classifier produced by running A with budget m, then with probability at least $(1 - \delta)$, we have $\mathbf{R}(h) \leq \epsilon$.

Definition (Disagreement coefficient (separable case)[4, 3])

Let $\mathcal{D}_{\mathcal{X}}$ be the underlying probability distribution on input space $\mathcal{X}$. Let H_ϵ be all hypotheses in H with error less than ϵ. Then,

1. disagreement region is defined as

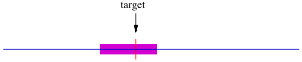
$$DIS(H_\epsilon) = \left\{\mathbf{x} \;\middle|\; \exists h, h' \in H_\epsilon \text{ such that } h(\mathbf{x}) \neq h'(\mathbf{x})\right\}.$$

2. Then, disagreement coefficient is defined as

$$\theta = \sup_{\epsilon} \frac{\mathcal{D}_{\mathcal{X}}(DIS(H_\epsilon))}{\epsilon}.$$

Example (Threshold classifier)

Let H be the set of all threshold functions in real line $\mathbb{R}$. Show that $\theta = 2$.



Example (Threshold classifier)

1. Let $\mathcal{X} = [0, 1]$ and $H = \left\{ h_{[z,1]} : \mathcal{X} \mapsto \{-1, +1\} \mid z \in (0, 1) \right\}$, where

$$h_{[z,1]}(x) = \begin{cases} +1 & \text{if } x \in [z, 1] \\ -1 & \text{if } x \notin [z, 1] \end{cases}$$

2. One simple passive learning algorithm for the realizable case would simply return z as the midpoint between the smallest positive example and the largest negative example.

3. Let $\mathcal{D}$ be uniform distribution over $\mathcal{X}$ and let also

$$h^*_{[z^*,1]}(x) = \begin{cases} +1 & \text{if } x \in [z^*, 1] \\ -1 & \text{if } x \notin [z^*, 1] \end{cases}$$

   where $\epsilon < z^* < 1 - \epsilon$ to guarantee $\mathbf{R}(h) \leq \epsilon$, it suffices to have some $x_i \in [z^* - \epsilon, z^*]$ and another $x_j \in [z^*, z^* + \epsilon]$.

4. Each of these regions has probability ϵ, so the probability this happens is at least $1 - 2(1-\epsilon)^m$ (by a union bound);
5. Since $1 - \epsilon \leq e^{-\epsilon}$, this is at least $1 - 2e^{-\epsilon m}$.
6. For this to be greater than $(1 - \delta)$, it suffices to take $m \geq \frac{1}{\epsilon} \ln \frac{2}{\delta}$.

Example (Threshold classifier (cont.))

7. The same results can be obtained for $z^* \in [0, \epsilon) \cup (1-\epsilon, 1]$, hence $m_H(\epsilon, \delta) = \frac{1}{\epsilon} \ln \frac{2}{\delta}$.
8. Consider the simple active learning algorithm, which returns $h_{[\hat{z},1]}$ when given budget m.

```
Let m_0 = 2^{m-1} and let {j_k}_{k=1}^{m_0} be the sequence such that x_{j_1} ≤ x_{j_2} ≤ ... ≤ x_{j_{m_0}}.
Initialize l = 0 and u = m_0 + 1.
repeat
    Let k = ⌊(l + u)/2⌋, request label y_{j_k} of point x_{j_1}.
    if y_{j_k} = 1 then
        Set u ← k
    else
        Set l ← k
    end if
until (l = u − 1)
if (l > 0) and (u < m_0 + 1) then
    Set ẑ ← [x_{j_l} + x_{j_u}] /2
else if (l = 0) then
    Set ẑ ← x_{j_u}/2
else if (u = m + 1) then
    Set ẑ ← [x_{j_l} + 1] /2
end if
```

Example (Threshold classifier (cont.))

9. Note that,
   9.1 k is median of l and u, and either l or u is set to k after each label request, the total number of label requests is at most $\log_2 m_0 + 1 = m$, so this algorithm stays within the indicated budget.
   9.2 The algorithm requests the largest value of x for which its label -1 and the smallest value of x for which its label $+1$.
10. Hence, this active learner outputs the same result as the passive learner.
11. This is remarkable, since $m_0 = 2^{m-1}$, then the label complexity of this algorithm for realizable case equals to

$$m_A(\epsilon, \delta, \mathcal{D}) \leq 1 + \left\lceil \log_2 \left(\frac{1}{\epsilon} \ln \frac{2}{\delta} \right) \right\rceil$$

12. This is an exponential improvement over passive learning.
13. We have shown that $VC(H) = 1$.
14. It can also be easy to show that $\theta \leq 2$.

Label complexity of CAL

Algorithm CAL [1]

Let $h : \mathcal{X} \mapsto \{-1, +1\}$ and $h^* \in H$.
Initialize $i = 1$ and $H_1 = H$.
while $(|H_i| > 1)$ **do**
    Select $\mathbf{x}_i \in \{\mathbf{x} \mid h \in H_1 \text{ disagrees}\}$. ▷ Region of disagreement
    Query with $\mathbf{x}_i$ to obtain $y_i = h^*(\mathbf{x}_i)$. ▷ Query the oracle
    Set $H_{i+1} \leftarrow \{h \in H_i \mid h(\mathbf{x}_i) = y_i\}$. ▷ Version space
    Set $i \leftarrow i + 1$.
end while

- The label complexity of CAL can be captured by $VC(H) = d$ and disagreement coefficient θ.
  1. For realizable case, label complexity of CAL equals to

$$\theta d \log(1/\epsilon).$$

  2. For unrealizable case, label complexity of CAL equals to (If best achievable error rate is v)

$$\theta \left(d \log^2 \frac{1}{\epsilon} + \frac{dv^2}{\epsilon^2} \right).$$

Summary

- We considered active learning problems:
- There are different scenarios of active learning.
- We defined two different measures of label complexity and disagreement coefficient.
- We showed that the label complexity is characterized by $VC(H)$ of hypothesis space and disagreement coefficient θ.
- It was shown that active learning decreases the label complexity in an exponential improvement over passive learning.

David Cohn, Les Atlas, and Richard Ladner. "Improving Generalization with Active Learning". In: *Machine Learning* 15.2 (May 1994), pp. 201–221.

Sanjoy Dasgupta and Daniel J. Hsu. "Hierarchical sampling for active learning". In: *Proceedings of the 25 International Conference on Machine Learning (ICML)*. Vol. 307. 2008, pp. 208–215.

Steve Hanneke. *Theory of Active Learning*. Tech. rep. Pennsylvania State University, 2014.

Steve Hanneke. "Theory of Disagreement-Based Active Learning". In: *Foundations and Trends in Machine Learning* 7.2-3 (2014), pp. 131–309.

SanjoyDasgupta. "Two faces of active learning". In: *Theoretical Computer Science* 412.19 (Apr. 2011), pp. 1767–1781.

Burr Settles. *Active Learning*. Morgan & Claypool Publishers, 2012.

Questions?