5. Online learning II

COMP0078: Supervised Learning

Mark Herbster

11 November 2019

University College London
Department of Computer Science

`PartIIOnlinelearning19v1`

Today

- Online Learning with Partial Feedback (Bandits)
- Matrix Completion

Part I

Learning with Partial Feedback

Recall Allocation Setting (Hedge)

Full Feedback Protocol

```
For $t = 1$ To $m$ Do
    Predict $\hat{y}_t \in [n]$
    Observe full loss vector $\boldsymbol{\ell}_t \in [0,1]^n$
```

Goal: Design master algorithms with "small regret".

$$\sum_{t=1}^{m} \ell_t, \hat{y}_t - \sum_{t=1}^{m} \min_{i \in [n]} \ell_{t,i} \leq o(m)$$

In the bandit setting we see only feedback for our prediction/action.

Partial Feedback Protocol

```
For $t = 1$ To $m$ Do
    Predict $\hat{y}_t \in [n]$
    Observe loss of prediction $\ell_{t,\hat{y}_t} \in [0,1]$
```

Goal: as above.

Note: Although (largely) unobserved $\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m$ are assumed to exist.

Partial Feedback Setting – Examples

1. **[Online Advertising]:** We have n potential ads to display. If the user click through on the ad we incur 0 loss.
2. **[Medical Trials]:** We have n potential medical treatments. The better the response the less loss.
3. **[Game tree search]:** We have n potential branches to search, if our "evaluation function" increases we proportionally receive less loss.

Exploration and Exploitation

This *Partial Feedback Setting* is often called the *Bandit* setting. The etymology being that we have a slot/fruit machine (once called one-armed bandits) each with potentially different "payback" rates and we wish to play so as to minimise our loss. Metaphorically, we will think of each prediction/action as pulling one of n arms.

Intuitions (Exploration and Exploitation)

1. We need to *use trials* to explore by trying different arms to estimate the machine with the minimal expected loss.
2. We need to *use trials* to exploit playing the arm with minimal observed loss.
3. The tradeoff: the *number of trials* used to explore limits exploitation and vice versa.

Before considering our "solution" let's discuss a key tool Unbiased Estimators.

Unbiased Estimators

Definition

An **estimator** $\hat{\theta}$ estimates a parameter θ of a distribution from a sample. An estimator is **unbiased** iff $E[\hat{\theta}] = \theta$.

Example 1 (Sample Mean)

Suppose $X_1, \ldots, X_n$ are IID random variables from a distribution with mean μ then $\hat{\theta} := \frac{1}{n}(X_1 + \ldots + X_n)$ is an unbiased estimator of μ.

Example 2 (German tank problem)

Suppose X is a random variable with the discrete uniform distribution over $\{1, \ldots, n\}$. Suppose n is unknown and we wish to estimate it The estimator $\hat{\theta}_1 := X$ is the maximum likelihood estimator, since $\mathcal{L}(\theta; X = x) = \frac{1}{\theta}[\theta \geq x]$ which is maximised when $\theta = x$. However for $\hat{\theta}_1$ we have

$$E[\hat{\theta}_1; \theta = n] = \sum_{x=1}^{n} \frac{1}{n} x = \frac{n+1}{2}$$

However $\hat{\theta}_2 := 2X - 1$ is the unbiased estimator, i.e.,

$$E[\hat{\theta}_2; \theta = n] = \sum_{x=1}^{n} \frac{1}{n}(2x - 1) = 2\sum_{x=1}^{n} \frac{1}{n} x - 1\sum_{x=1}^{n} \frac{1}{n} = n + 1 - 1 = n$$

Assumptions and Estimation

Suppose we have a distribution $\mathcal{D}_i$ over $[0,1]$ for each of $i \in [n]$ arms then each time t we "play" i we receive an IID sample $\ell_{t,i}$ from $\mathcal{D}_i$. Suppose we play i on trials $S_{t,i} \subseteq [t]$ then $\hat{\mu}_{t,i} := \sum_{t \in S_i} \frac{\ell_{t,i}}{|S_{t,i}|}$ is an unbiased estimator of μ_i.

As we get more samples from arm i the law of large numbers implies $\hat{\mu}_i \to \mu_i$ and concentration inequalities (e.g., Hoeffding) allow one to quantatively estimate the likelihood that the estimate differs significantly from the parameter. Using these observations the algorithm UCB balances exploration versus exploitation to obtain good regret bounds for this model. However we would like to consider a more general *adversarial* model. For example suppose $\mathcal{D}_i$ is changing over time (now $\mathcal{D}_{t,i}$) then the estimate $\hat{\mu}_{t,i}$ is biased (now $\mu_{t,i} := \frac{\sum_{j=1}^{t} E[\ell_{j,i}]}{t}$) unless $S_i = [t]$. However if $S_{t,i} = [t]$ then we have no information on the other "arms."

Needed: a method of obtaining a **simultaneous** unbiased estimate for **all** arms!

Importance Weighting – 1

Suppose X is random variable over $\Re$ with a mean μ. By definition $E[X] = \mu$ and $\hat{\theta}_1 = X$ is an unbiased estimator of the mean. Define the biased coin Z_p with outcome H with probability p and T with probability $(1-p)$ then define estimator $\hat{\theta}_p$ as equal to X/p if $Z_p = H$ as equal to 0 if $Z_p = T$. Now observe that,

$$E[\hat{\theta}_p] = \mathbb{P}(Z_p = H)(X/p) + \mathbb{P}(Z_p = T)0 = (p)X/p + (1-p)0 = X$$

and is thus unbiased.

Importance Weighting – 2

We now generalise to obtain an unbiased estimator of $\boldsymbol{\ell}_t$ in the bandit setting. Given $\mathbf{v}_t \in \Delta_n$ by the notation $\hat{y}_t \sim \mathbf{v}_t$ we mean sample $\hat{y}_t$ from a discrete distribution over $[n]$ where $\mathbb{P}(i) := v_{t,i}$.

Definition (Hallucinated Loss Vector)

We define the unbiased estimator $\boldsymbol{\ell}^h_t$ of $\boldsymbol{\ell}_t$ with respect to $\mathbf{v}_t$ as

$$(\ell^h_{t,i} := \frac{\ell_{t,i}}{v_{t,i}}[i = \hat{y}_t])_{i \in [n]}\,,$$

where we have sampled $\hat{y}_t \sim \mathbf{v}_t$.

I.e, $\boldsymbol{\ell}^h_t$ looks like, $\boldsymbol{\ell}^h_t := (0, 0, \ldots, \frac{\ell_{t,\hat{y}_t}}{v_{t,\hat{y}_t}}, 0, \ldots, 0)$. Observe that $\boldsymbol{\ell}^h_t$ is unbiased as for all $i \in [n]$ we have that $E[\ell^h_{t,i}] = \ell_{t,i}$, since

$$E_{\hat{y}_t \sim \mathbf{v}_t}[\ell^h_{t,i}] = \sum_{j=1}^{n} v_{t,j} \frac{\ell_{t,i}}{v_{t,i}}[i = j] = \ell_{t,i}$$

- Amazingly! we have an unbiased estimator for all arms by only observing a single arm.

The Next Steps

Idea

By sampling a single arm we can obtain an unbiased estimate for $\boldsymbol{\ell}_t$. Since we already have an algorithm for the full information setting for arbitrary loss functions (Hedge) our proposed algorithm is to simply apply Hedge to the hallucinated loss vectors.

Todo

1. **Problem:** $\boldsymbol{\ell}_t^h$ is potentially unbounded and Hedge requires bounded loss vectors. **Fix:** Use a more careful analysis of Hedge.
2. **Problem:** We will be giving an *expected* regret bound and there will be some subtleties in the sources of randomness. **Fix:** we will clarify the adversarial model that generates $\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m$.

Exp3 Exponential-weight algorithm for Exploration and Exploitation [ACFS02]

```
Exp3
Parameter $\eta \in (0, \infty)$
Set $\mathbf{v}_1 = (\frac{1}{n}, \ldots, \frac{1}{n})$
For $t = 1$ To $m$ do
    Sample $\hat{y}_t \sim \mathbf{v}_t$ % i.e., treat $\boldsymbol{v}_t$ as a distribution over $[n]$
    Observe loss $\ell_{t,\hat{y}_t} \in [0, 1]$
    Construct hallucinated loss vector
        $\boldsymbol{\ell}_t^h := (\ell_{t,i}^h := \frac{\ell_{t,i}}{v_{t,i}}[i = \hat{y}_t])_{i \in [n]}$
    Update
        $v_{t+1,i} = v_{t,i} \exp(-\eta \ell_{t,i}^h)/Z_t$ for $i \in [n]$
        where $Z_t = \sum_{i=1}^n v_{t,i} \exp(-\eta \ell_{t,i}^h)$
```

Footnote: the original Exp3 algorithm was slightly different but it incorporated the key idea of using Hedge with *hallucinated* loss vectors.

Exp3 – Initial Analysis

Lemma

For any sequence of loss vectors

$$\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m \in [0,1]^n$$

we have the following inequality for Exp3,

$$\sum_{t=1}^{m} \mathbf{v}_t \cdot \boldsymbol{\ell}_t^h - \sum_{t=1}^{m} \mathbf{u} \cdot \boldsymbol{\ell}_t^h \leq \frac{\ln n}{\eta} + \frac{\eta}{2} \sum_{t=1}^{m} \sum_{i=1}^{n} v_{t,i} (\ell_{t,i}^h)^2 \text{ for all } \mathbf{u} \in \Delta_n \,. \tag{1}$$

Proof. The Lemma follows the fact Exp3 is “just” Hedge with $\boldsymbol{\ell}_t$ mapped to $\boldsymbol{\ell}_t^h$ and we proved the above inequality for Hedge on slide “Hedge Theorem - Proof (2).”

To finish our analysis:

1. We need to perform expectations so we may replace hallucinated losses $\boldsymbol{\ell}_t^h$ with the true losses $\boldsymbol{\ell}_t$.
2. In order perform expectations we need understand the sources of randomness in our model, in particular we need a model for how the “adversary” generates $\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m$
3. Finally we need to bound the term $\sum_{t=1}^{m} \sum_{i=1}^{n} v_{t,i} (\ell_{t,i}^h)^2$ and tune η.

Model : Deterministic Oblivious Adversary

Motivation: Our predictions/actions may influence the environment.

Suppose we are using a bandit algorithm to set prices for our products. This may influence the way competitors set prices. If we have an app to display curated news to a user the prior display of new items may effect the user's choices for the next news items. More generally the learners actions may influence the adversary/nature in the future.

Deterministic Oblivious Adversary Model

In this model simply put

$$\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m$$

are determined before the run the algorithm.

However, the process/adversary setting them is assumed to have complete prior knowledge of the learner's algorithm and may set the loss vectors using this knowledge, i.e., if the adversary knows the learner will do (or is likely to) predict something on trial t it may set $\boldsymbol{\ell}_t$ accordingly. The limitation of this near-omniscient adversary is that it is non-adaptive i.e., if the learner will be taking random actions although the adversary may take this into account in setting $\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m$ it cannot however change them "on-the-fly." Observe that this adversary may simulate the *stochastic* model, by simple repeatedly sampling $\mathcal{D}_1, \dots, \mathcal{D}_m$ in advance.

Exp3 - Theorem

Notation: $L_A(S) := \sum_{t=1}^{m} \ell_{t,\hat{y}_t}$ and $L_i(S) := \sum_{t=1}^{m} \ell_{t,i}$

Theorem Exp3 **(Bound)** **[ACFS02]**

For any sequence of loss vectors

$$S = \boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_m \in [0,1]^n$$

the regret of Exp3 with $\eta = \sqrt{2 \ln n / mn}$ is

$$E[L_A(S)] - \min_i L_i \le \sqrt{2mn \ln n}\,.$$

Comparing regrets : Hedge: $\sqrt{2m \ln n}$ and Exp3: $\sqrt{2mn \ln n}$.

EXP3 Theorem - Proof (1) [Not Examined 18/19]

Proof – 1

Observe that the only sources of randomness are the samples $\hat{y}_t \sim \mathbf{v}_t$. As previously argued note that $E[\ell^h_{t,i}] = \ell_{t,i}$ and thus

$$E[\mathbf{v}_t \cdot \boldsymbol{\ell}^h_t] = \sum_{i=1}^{n} E[v_{t,i}\ell^h_{t,i}] = \sum_{i=1}^{n} v_{t,i} E[\ell^h_{t,i}] = \sum_{i=1}^{n} v_{t,i}\ell_{t,i} = E[\ell_{t,\hat{y}_t}] \qquad (2)$$

and we also have,

$$E[(\ell^h_{t,i})^2] = \sum_{j=1}^{n} v_{t,j}\left(\frac{\ell_{t,i}}{v_{t,i}}\right)^2 [j = i]^2 = v_{t,i}\left(\frac{\ell_{t,i}}{v_{t,i}}\right)^2 = \frac{\ell^2_{t,i}}{v_{t,i}} \qquad (3)$$

which implies

$$E\left[\sum_{i=1}^{n} v_{t,i}(\ell^h_{t,i})^2\right] = \sum_{i=1}^{n} v_{t,i}\frac{\ell^2_{t,i}}{v_{t,i}} = \sum_{i=1}^{n} \ell^2_{t,i} \leq n\,. \qquad (4)$$

Exp3 Theorem - Proof (2) [Not Examined 18/19]

Proof – 2

Recalling (1) and taking expectations,

$$E\left[\sum_{t=1}^{m} \mathbf{v}_t \cdot \boldsymbol{\ell}_t^h - \sum_{t=1}^{m} \mathbf{u} \cdot \boldsymbol{\ell}_t^h\right] \leq E\left[\frac{\ln n}{\eta} + \frac{\eta}{2} \sum_{t=1}^{m} \sum_{i=1}^{n} v_{t,i} (\ell_{t,i}^h)^2\right] (\forall \mathbf{u} \in \Delta_n)$$

thus,

$$E\left[\sum_{t=1}^{m} \mathbf{v}_t \cdot \boldsymbol{\ell}_t^h\right] - \min_i E\left[\sum_{t=1}^{m} \ell_{t,i}^h\right] \leq \frac{\ln n}{\eta} + \frac{\eta}{2} E\left[\sum_{t=1}^{m} \sum_{i=1}^{n} v_{t,i} (\ell_{t,i}^h)^2\right]$$

applying (2) to the first term, lower bounding the second term by observing that $\mathbf{u}$ can be any coordinate vector and that $E[\ell_{t,i}^h] = \ell_{t,i}$ then using (4) for the final term gives

$$E\left[L_A(S)\right] - \min_i L_i(S) \leq \frac{\ln n}{\eta} + \frac{\eta}{2} mn$$

now substituting $\eta = \sqrt{2 \ln n / mn}$ proves the theorem. □

Matrix Completion

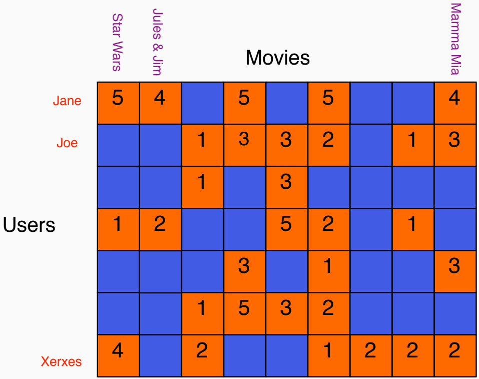
Part II

Matrix Completion

Overview

1. We discuss a generalisation of Winnow to the matrix setting.
2. Winnow, "prediction with expert advice," and Hedge, may all be analysed with an amortised analysis using relative entropy. For the generalisation to matrices this analysis now uses the quantum relative entropy. [We omit the analysis however].
3. We give a bound for Matrix Winnow applied to matrix completion in terms of margin complexity.
4. We show that margin complexity bounds may be interpreted as "learning the kernel" in a multi-task setting.

Matrix Completion



	Star Wars	Jules & Jim							Mamma Mia
Jane	5	4		5		5			4
Joe			1	3	3	2		1	3
			1		3				
	1	2			5	2		1	
				3		1			3
			1	5	3	2			
Xerxes	4		2			1	2	2	2

- **Matrix completion:** fill in the missing values.
- For example, the rows may represent users and the columns movies.
- Netflix had a $1,000,000 challenge with a matrix consisted of 480,189 users × 17,770 users with 100,480,507 rating – won in 2009.
- We will discuss an online approach based on Matrix Winnow.

Online binary matrix completion

- Aim: to predict the entries of a matrix $\boldsymbol{U} \in \{\pm 1\}^{m \times n}$
- Learning proceeds in trials

```
**For** $t = 1, ..., mn$ **do**
    1. Nature selects an entry $(i_t, j_t) \in \{1, ..., m\} \times \{1, ..., n\}$
    2. Learner predicts $\hat{U}_{i_t, j_t} \in \{-1, 1\}$
    3. Nature returns $y_t = U_{i_t, j_t}$
    4. If $\hat{U}_{i_t, j_t} \neq U_{i_t, j_t}$ then mistakes $=$ mistakes $+ 1$
```

- Goal is to bound mistakes $\leq f(\mathit{complexity}(\boldsymbol{U}))$
- Alternately we could aim for *regret* bounds for simplicity we focus on the realizable case.

Matrix Complexity

Rank Complexity

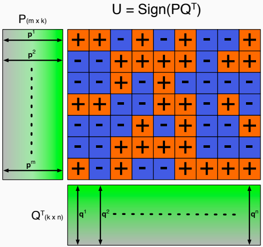
A natural notion of complexity for matrices is that of a low rank decomposition. A matrix $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ has a rank-k decomposition if there exists $\boldsymbol{P} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{Q} \in \mathbb{R}^{n \times k}$ such that $\boldsymbol{U} = \boldsymbol{P}\boldsymbol{Q}^{\top}$. Observe that a rank-$k$ decomposition is specified by $k(m+n)$ parameters.

Factor Models

Consider the Netflix problem. A very simple model is that for each user i we associate a factor p_i which is how positive they are about movies and likewise for movies for each movie j we associate a factor q_j which is popular the movie and thus for any particular (user i,movie j) the score associated is just $U_{ij} = p_i \times q_j$ this is a rank-1 decomposition $\boldsymbol{p} \in \mathbb{R}^n$ and $\boldsymbol{q} \in \mathbb{R}^m$. This is an overly simple model ... slightly more complex we might consider there are a k factors and each user and movie has a score with respect to these factor (e.g., comedy, romance, age, education) and total score is the sum of the scores i.e, $U_{ij} = \sum_{s=1}^{k} p_i^{(s)} q_j^{(s)}$, i.e., a rank-$k$ decomposition.

Rank Complexity

- Since we are considering classification problems we will consider the rank complexity of matrix the minimum rank of any matrix with the same sign pattern.



+	+	-	+	-	+	-	-	+
-	-	+	+	+	+	-	+	+
-	-	+	-	+	-	-	-	-
+	+	-	-	+	+	-	+	-
-	-	-	+	-	+	-	-	+
-	-	+	+	+	+	-	-	-
+	-	+	-	-	+	+	+	+

Definition:

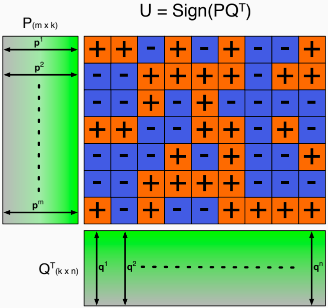
$$\mathrm{dc}(\boldsymbol{U}) := \min_{\substack{\boldsymbol{P}\in\mathbb{R}^{m\times k} \\ \boldsymbol{Q}\in\mathbb{R}^{n\times k} \\ \forall r,s:(\boldsymbol{PQ}^\top)_{rs}\times U_{rs}\geq 1}} k$$

Interpretation

Suppose the columns (movies) are points $x \in \mathcal{X}$ in an input space. And the rows (users) are the hypotheses in $h \in \mathcal{H}$ in hypothesis space so that $h : \mathcal{X} \to \{-1, 1\}$ then $U_{hx} = \mathrm{sign}(h(x))$ and thus $\mathrm{dc}(\boldsymbol{U})$ is the smallest $\mathrm{dc}(\boldsymbol{U})$-dimensional space for which their is a *linear* embedding of for all the data points and hypotheses. I.e., for $\forall x \in \mathcal{X}$ there exists $\mathbf{x}_x \in \mathbb{R}^k$ and $\forall h \in \mathcal{H}$ a $\mathbf{h}_h \in \mathbb{R}^k$ such that $h(x) = \mathrm{sign}(\mathbf{x}_x \cdot \mathbf{h}_h)$.

Note: The $\mathrm{dc}(\boldsymbol{U})$ may be arbitrarily smaller than the $\mathrm{rank}(\boldsymbol{U})$. Why?

Margin Complexity



+	+	-	+	-	+	-	-	+
-	-	+	+	+	+	-	+	+
-	-	+	-	+	-	-	-	-
+	+	-	-	+	+	-	+	-
-	-	-	+	-	+	-	-	+
-	-	+	+	+	+	-	-	-
+	-	+	-	-	+	+	+	+

Interpretation

Rows $h \in \mathcal{H}$ and Columns $x \in \mathcal{X}$ so that $h : \mathcal{X} \to \{-1, 1\}$ then $U_{hx} = \mathrm{sign}(h(x))$ and thus $\mathrm{mc}^2(\boldsymbol{U})$ is the reciprocal of the “maximum margin-squared” × “maximum norm-squared” of a “**x**” for the “worst” pairing $h(x)$. I.e., the best possible “perceptron bound” wrt to the “hardest” hypothesis.

Definition:

$$\mathrm{mc}(\boldsymbol{U}) := \min_{\substack{\boldsymbol{P}\in\mathbb{R}^{m\times k}\\ \boldsymbol{Q}\in\mathbb{R}^{n\times k}\\ k\in\mathbb{N}\\ \forall r,s:(\boldsymbol{P}\boldsymbol{Q}^\top)_{rs}\times U_{rs}\geq 1}} \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}} \|\boldsymbol{p}_i\|_2\|\boldsymbol{q}_j\|_2$$

Without proof we have
$\mathrm{vcd}(\boldsymbol{U}) \leq \mathrm{mc}^2(\boldsymbol{U}) \leq \mathrm{rank}(\boldsymbol{U})$ and $\mathrm{vcd}(\boldsymbol{U}) \leq \mathrm{dc}(\boldsymbol{U}) \leq \mathcal{O}(\mathrm{mc}^2(\boldsymbol{U}) \log(mn))$

Open Problem: Is $\mathrm{dc}(\boldsymbol{U}) \leq \mathcal{O}(\mathrm{mc}^2(\boldsymbol{U}))$?

Matrix Winnow

Parameters: Learning rate $0 < \gamma \leq 1$.

Initialization: $\boldsymbol{W}^{(0)} \leftarrow \frac{\boldsymbol{I}}{(m+n)}$, where $\boldsymbol{I}$ is the identity matrix.

For $t = 1, \ldots, T$

- Get pair $(i_t, j_t) \in \{1, ..., m\} \times \{1, ..., n\}$.
- Define $\boldsymbol{X}^{(t)} := \frac{1}{2}(\boldsymbol{e}_{i_t} + \boldsymbol{e}_{m+j_t})(\boldsymbol{e}_{i_t} + \boldsymbol{e}_{m+j_t})^\top$.
- Predict $\hat{U}_{i_t, j_t} = \begin{cases} 1 & \text{if } \operatorname{Tr}(\boldsymbol{W}^{(t-1)}\boldsymbol{X}^{(t)}) \geq \frac{1}{m+n}, \\ -1 & \text{otherwise.} \end{cases}$
- Receive $U_{i_t, j_t} \in \{-1, 1\}$ and if $U_{i_t, j_t} \neq \hat{U}_{i_t, j_t}$ update

$$\boldsymbol{W}^{(t)} \leftarrow \exp\left(\log\left(\boldsymbol{W}^{(t-1)}\right) + \frac{\gamma}{2}(y_t - \hat{y}_t)\boldsymbol{X}^{(t)}\right).$$

Note: These are matrix log and exp defined for p.d. matrices which is done by computing the eigensystem take the log and exp of the eigenvalues respectively and “rebuilding” the matrix.

Mistake Bounds

Theorem

The mistakes of Matrix Winnow with $\gamma := \frac{1}{\mathrm{mc}(\boldsymbol{U})}$ is bounded by

$$\text{mistakes} \leq 3.55\,\mathrm{mc}^2(\boldsymbol{U})\,(m+n)\log(m+n)$$

- The above bound is tight up to a logarithmic factor.

Theorem (lower bound):

Given any online algorithm $\mathcal{A}$ and an $\ell \in \{1, \ldots, n\}$ there exists a matrix $\boldsymbol{U}$ with margin complexity $\mathrm{mc}^2(\boldsymbol{U}) \leq \ell$ and a sequence of matrix entries such that

$$\ell \times m \leq \text{mistakes}$$

Interpreting the bound – examples

1. Multitask Prediction
2. Biclustered Matrices

Example (Multi-task Prediction)

Multitask Prediction

Single Task Prediction with a Known Kernel K

How can we predict which films Blue will like?

u_1
u_2
u_3
u_4
u_5
u_6

- In the online setup given a kernel $K($ $,$ $) \to \mathbb{R}$ we may use the perceptron to bound the mistakes.

Bound for Single Task Prediction with a Known Kernel K

Model

- Aim: to predict the entries of vector $\mathbf{u} \in \{-1, 1\}^n$
- Learning proceeds in trials

```
for $t = 1, \ldots, n$ do
    1. Nature selects vector entry $j_t \in \{1, \ldots, n\}$
    2. Learner predicts $\hat{u}_{j_t} \in \{-1, 1\}$
    3. Nature returns $u_{j_t}$,
    4. If $\hat{u}_{j_t}, \neq u_{j_t}$ then mistakes $=$ mistakes $+ 1$
```

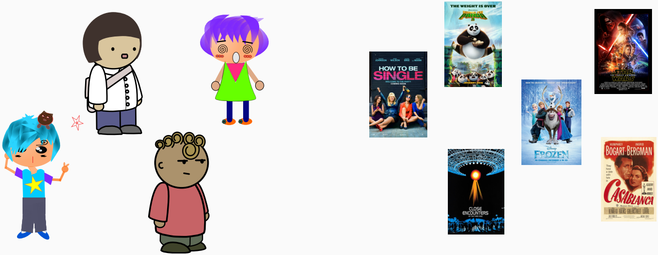
Theorem (Novikoff):

The number of mistakes to predict elements of vector $\mathbf{u}$ is bounded by

$$\text{mistakes} \leq \min_{\boldsymbol{w} \in \mathbb{R}^n} \boldsymbol{w}^\top \boldsymbol{K}^{-1} \boldsymbol{w} \max_{1 \leq j \leq n} K_{jj}$$

where $u_j w_j \geq 1$ for all $j \in \{1, \ldots, n\}$ if $\hat{\mathbf{u}}$ is the predictions of the kernel perceptron. **Note:** $\boldsymbol{K}$ is assumed to be strictly p.d. for simplicity.

Multitask Prediction without a Kernel



- If we have multiple tasks we may learn the “best” kernel.
- Each task is now a row in the matrix to be predicted.

Alternate Formulation of Margin Complexity

- We may reformulate the margin complexity in terms of the kernel perceptron bound.

Proposition

$$\mathrm{mc}^2(\boldsymbol{U}) = \min_{\substack{\boldsymbol{K} \succ 0 \\ \boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(m)} \in \mathbb{R}^n}} \max_{i \in [m], j \in [n]} \boldsymbol{w}^{(i)\top} \boldsymbol{K}^{-1} \boldsymbol{w}^{(i)} K_{jj}$$

where $U_{ij} w_j^{(i)} \geq 1$ for all $i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}$.

Recall that,

$$\text{mistakes} \leq 3.55\, \mathrm{mc}^2(\boldsymbol{U})\, (m+n) \log(m+n)$$

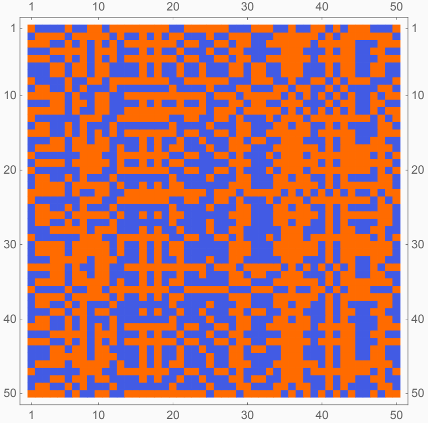
- Thus by applying Matrix Winnow we predict as well as the best kernel on the worst task.
- For this interpretation to make sense $|\text{tasks}| \geq |\text{item}|$.

Example (Biclustering)

Biclustered Matrices

Biclustered Matrices (1)

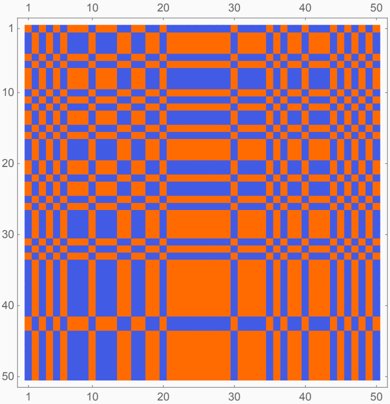
- A matrix is (k, ℓ)-biclustered if after a permutation of the rows and columns the resultant is a $k \times \ell$ grid of rectangles each labeled as either -1 or 1.



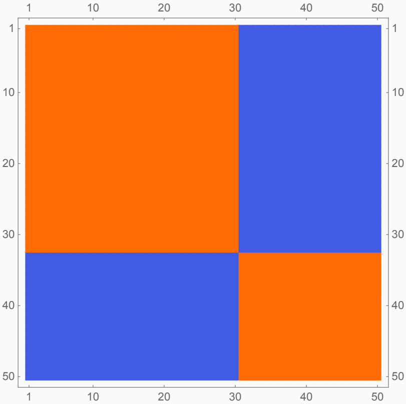
A (9,9)-biclustered matrix

Biclustered Matrices (2)

Another example



A (2,2)-biclustered matrix



After permutation

Biclustered Matrices (3)

Definition

The set of (k,ℓ)-biclustered matrices is

$$\mathbb{B}^{m,n}_{k,\ell} := \left\{\boldsymbol{U}\in\{\pm1\}^{m\times n} : \substack{\boldsymbol{r}\in[k]^m \\ \boldsymbol{c}\in[\ell]^n}, \boldsymbol{F}\in\{\pm1\}^{k\times\ell}, U_{ij} = F_{r_i c_j}, \substack{i\in[m] \\ j\in[n]}\right\}.$$

Theorem

If $\boldsymbol{U}\in\{\pm1\}^{m\times n}$ is a (k,ℓ)-biclustered matrix then

$$\mathrm{mc}^2(\boldsymbol{U}) \le \min(k,\ell).$$

- Observe that VC-dimension$(\mathbb{B}^{m,n}_{k,\ell}) \geq k \times \ell$. This combined with the above theorem implies the previous lower bound.

Problems – 1

1. Prove the “rewrite” of Novikoff’s bound in slide “Bound for Single Task Prediction with a Known Kernel $\boldsymbol{K}$”
2. Suppose $\boldsymbol{U}$ has a (k, ℓ)-biclustering argue that it has a rank-min(k, ℓ) decomposition.
3. There may be an arbitrarily large gap between margin complexity an rank. Argue that the matrix $2I_n - j_n$ (I_n is the $n \times n$ identity matrix, and J_n is matrix of all ones), has a rank of n but a margin complexity of $\mathcal{O}(1)$.
4. Design an inefficient algorithm that achieves a mistake bound of $\mathcal{O}(m \log k + n \log \ell + k\ell)$ for (k, ℓ)-biclustered matrices. If you design an efficient algorithm please submit to NeurIPS :-).

Suggested Readings

Part I follows the presentation from Daniel Hsu's Notes for COMS 6998-4 Fall 2017 Chapter 1 is particularly recommended for more details on Exp3 and Hedge. Part II is based on Mistake bounds for binary matrix completion.

Useful references

1. P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. *The nonstochastic multiarmed bandit problem*, (2002).
2. S. Ben-David, N. Eiron, and H. U. Simon. Limitations of learning via embeddings in euclidean half spaces. *JMLR*, 2003.
3. Nicolò Cesa-Bianchi and Gábor Lugosi, *Prediction, learning, and games.*, (2006), Note this is a book.
4. E. Hazan, S. Kale, and S. Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. In Proc. 23rd Annual Conference on Learning Theory, volume 23:38.1-38.13. *JMLR*, 2012.
5. M. Herbster, S. Pasteris and M. Pontil. *Mistake bounds for binary matrix completion*, (2016).
6. N. Linial, S. Mendelson, G. Schechtman, and A. Shraibman. Complexity measures of sign matrices. *Combinatorica*, 2007.
7. N. Srebro and A. Shraibman. Rank, trace-norm and max-norm. (2005)
8. M. Warmuth *Winnowing subspaces*, (2007)