http://goo.gl/svgV0n

KYOTO UNIVERSITY

Statistical Machine Learning Theory

Lecture 11

On-line Learning

Hisashi Kashima
kashima@i.Kyoto-u.ac.jp

DEPARTMENT OF INTELLIGENCE SCIENCE AND TECHNOLOGY

Topics:
Online learning algorithms and theoretical guarantees

- On-line learning problem
- Halving algorithm, its theoretical mistake bound, and its limitation
- Regret analysis as a performance measure of online learning algorithms
- Analyses of:
  - Follow-the-leader (FTL) and follow-the-regularized-leader (FTRL) algorithms
  - Online gradient descent algorithm
  - Perceptron algorithm

Most of the contents in this lecture are based on:
Shalev-Shwartz, S. (2011). Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, *4*(2), 107-194.

On-line learning problem:
Learning to make periodical decisions

- In standard (batch) learning settings,
  1. Given training dataset $\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$
  2. Make predictions for test dataset $\{\mathbf{x}^{(N+1)}, \ldots, \mathbf{x}^{(N+M)}\}$
  3. Get feedbacks (reward or loss)
- In online learning,
  1. At each round, make a prediction for an arriving data
  2. Get a feedback for the prediction
  3. Return to 1
  - Training and test are done with the same data

On-line learning applications:
Real-time modeling and prediction

- You continuously have to make decisions (and get feedbacks)
- (Somewhat ambitious) examples
  - Weather forecasting
  - Stock price prediction
- Sometimes considered as an efficient alternative to batch learning (for big data!)

On-line learning problem formulation: Guaranteed strategy to minimize cumulative loss

- At each round $t = 1, 2, \dots, T$
  1. Receive input $\mathbf{x}^{(t)} \in \mathcal{X}$
  2. Make prediction $p^{(t)} \in \mathcal{Y}$
  3. Observe true answer $y^{(t)} \in \mathcal{Y}$
  4. Suffer loss $l\left(p^{(t)}, y^{(t)}\right)$

the environment chooses $y^{(t)}$

- Our goal
  - Find a prediction strategy to minimize cumulative loss $\sum_{t=1}^{T} l\left(p^{(t)}, y^{(t)}\right)$
  - Theoretical guarantees of the performance of the strategy

A simple online learning problem example : Two-class classification with a finite set of predictors

- Consider an on-line two-class classification problem
  - At each round $t = 1, 2, \dots, T$
    1. Receive input $\mathbf{x}^{(t)} \in \mathcal{X}$
    2. Make prediction $p^{(t)} \in \{+1, -1\}$
    3. Observe true answer $y^{(t)} \in \{+1, -1\}$
    4. Suffer loss $l\left(p^{(t)}, y^{(t)}\right) = 0$ (if $p^{(t)} = y^{(t)}$) or 1 (if $p^{(t)} \neq y^{(t)}$)
- Assumption:
  1. Finite hypotheses:
     A finite set of predictors $\mathcal{H}$ ($|\mathcal{H}| < \infty$) is available
  2. Realizability: True answers are generated by some $h^* \in \mathcal{H}$

Halving algorithm : Majority vote prediction with version space

- Initialization: $V_1 = \mathcal{H}$ (V_t is called a version space)
  - maintains predictors consistent with past observations
- At each round $t = 1, 2, \dots, T$
  1. Receive input $\mathbf{x}^{(t)} \in \mathcal{X}$
  2. Predict $p^{(t)} = \operatorname{argmax}_{p \in \{+1,-1\}} \left| \left\{ h \in V_t \mid h\left(\mathbf{x}^{(t)}\right) = p \right\} \right|$
     - Take a majority vote with the current version space
  3. Observe true answer $y^{(t)} \in \{+1, -1\}$
  4. Update $V_{t+1} = \left\{ h \in V_t \mid h\left(\mathbf{x}^{(t)}\right) = y^{(t)} \right\}$
     - Correct hypotheses survive to next round

Theoretical guarantee of the halving algorithm : Logarithmic mistake bound

- Halving algorithm makes at most $\log_2(|\mathcal{H}|)$ wrong predictions
- Proof:
  - Whenever the algorithm makes a mistake, more than a half of the members in the current version space V_t make mistakes
    - Size of the next version space $|V_{t+1}| \leq \frac{|V_t|}{2}$
  - After making M mistakes, $|V_t| \leq \frac{|\mathcal{H}|}{2^M}$
  - Since at least one predictor survives, $1 \leq |V_t|$
  - Rearranging $1 \leq \frac{|\mathcal{H}|}{2^M}$ concludes the proof

realizability assumption

Limitations of the current setting: Adversarial environments do not allow mistake bounds

- The halving algorithm cannot enjoy the logarithmic bound
  - when $\mathcal{H}$ is an infinite set (e.g. $\mathbf{w} \in \mathbb{R}^D$)
  - when the true predictor is not in $\mathcal{H}$
- Even worse when the environment is adversarial
  - The environment can decide the true answer after observing an algorithm's prediction
  - Number of mistakes can be T

Regret: Relative performance in a particular class of predictors

- Adversarial environments can always make wrong predictions
  - Impossible to guarantee mistake bounds
- Regret: relative performance in a particular class of predictors

$$\mathrm{Regret}_T(\mathcal{H}) = \underbrace{\sum_{t=1}^{T} l\left(p^{(t)}, y^{(t)}\right)}_{\text{cumulative loss by the algorithm}} - \underbrace{\min_{h\in\mathcal{H}} \sum_{t=1}^{T} l\left(h\left(\mathbf{x}^{(t)}\right), y^{(t)}\right)}_{\text{minimum cumulative loss in } \mathcal{H}}$$

  - h^* is the predictor achieving the minimum cumulative loss
  - Even with an adversarial environment, regret will not be large if all members of $\mathcal{H}$ perform poorly

Regret bound:
Sublinear regret bound guarantees relative performance

- If $\mathrm{Regret}_T(\mathcal{H}) = \overset{\text{sublinear}}{o(T)}$ (e.g. $\sqrt{T}$), $\frac{\mathrm{Regret}_T(\mathcal{H})}{T} \to 0$ as $T \to \infty$
  - Your algorithm is asymptotically guaranteed to perform as well as the best predictor in $\mathcal{H}$!

$$\sum_{t=1}^{T} l\left(p^{(t)}, y^{(t)}\right) \leq \min_{h\in\mathcal{H}} \sum_{t=1}^{T} l\left(h\left(\mathbf{x}^{(t)}\right), y^{(t)}\right) + o(T)$$

On-line learning problem formulation II:
Online learning of general models with parameters

- Consider of a specific class of online learning problems
  - to design online learning algorithms of models with parameters (e.g. linear classifiers)
- At each round $t = 1, 2, \dots, T$
  1. Submit a parameter vector $\mathbf{w}^{(t)} \in \mathcal{S}$ (e.g. $\mathbb{R}^D$)
  2. Receive a loss function $l^{(t)}: \mathcal{S} \to \mathbb{R}$
  3. Suffer loss $l^{(t)}\left(\mathbf{w}^{(t)}\right)$
  - Loss function $l^{(t)}$ can be different at each round
- $\mathrm{Regret}_T(\mathcal{S}) = \sum_{t=1}^{T} l^{(t)}\left(\mathbf{w}^{(t)}\right) - \min_{\mathbf{w}\in\mathcal{S}} \sum_{t=1}^{T} l^{(t)}(\mathbf{w})$

Some examples of loss function: Convex and non-convex loss functions

- Convex loss functions
  - Squared loss (Online regression)

$$l^{(t)}\left(\mathbf{w}^{(t)}\right) = l\left(\mathbf{w}^{(t)\top}\mathbf{x}^{(t)}, y^{(t)}\right) = \left(\mathbf{w}^{(t)\top}\mathbf{x}^{(t)} - y^{(t)}\right)^2$$

  - Linear function (Online linear optimization)

$$l^{(t)}\left(\mathbf{w}^{(t)}\right) = \left\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)} \right\rangle$$

- Non-convex loss function
  - 0-1 loss (Online classification)

prediction is wrong

$$l^{(t)}\left(\mathbf{w}^{(t)}\right) = 1_{\left[y^{(t)}\left\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)} \right\rangle \le 0\right]}$$

Follow-the-leader: An online algorithm with regret bound

- An online algorithm specifies $\mathbf{w}^{(t)}$
- Follow-the-Leader (FTL) submits $\mathbf{w}^{(t)}$ which has the minimum cumulative loss for the past rounds
  - i.e. $\mathbf{w}^{(t)} = \mathrm{argmin}_{\mathbf{w}\in\mathcal{S}} \sum_{\tau=1}^{t-1} l^{(\tau)}(\mathbf{w})$
- Lemma: ${}^{\forall}\mathbf{u}$,

decrease of $l^{(t)}$ by the update

$$\sum_{t=1}^{T}\left(l^{(t)}\left(\mathbf{w}^{(t)}\right) - l^{(t)}(\mathbf{u})\right) \le \sum_{t=1}^{T}\left(l^{(t)}\left(\mathbf{w}^{(t)}\right) - l^{(t)}\left(\mathbf{w}^{(t+1)}\right)\right)$$

  - This holds for $\mathbf{u} = \mathrm{argmin}_{\mathbf{w}\in\mathcal{S}} \sum_{t=1}^{T} l^{(t)}(\mathbf{w})$, so gives an upper bound of $\mathrm{Regret}_T(\mathcal{S})$

Proof of the FTL lemma: Proof by induction

- What we want to show $^\forall \mathbf{u},\ \sum_{t=1}^{T} l^{(t)}\left(\mathbf{w}^{(t+1)}\right) \leq \sum_{t=1}^{T} l^{(t)}(\mathbf{u})$
- For $T=1$, $l^{(1)}\left(\mathbf{w}^{(2)}\right) \leq l^{(1)}(\mathbf{u})$ holds since $\mathbf{w}^{(2)}$ is determined so that $l^{(1)}$ is minimized
- Suppose the inequality holds for $T-1$, i.e. $\sum_{t=1}^{T-1} l^{(t)}\left(\mathbf{w}^{(t+1)}\right) \leq \sum_{t=1}^{T-1} l^{(t)}(\mathbf{u})$
- Adding $l^{(T)}\left(\mathbf{w}^{(t+1)}\right)$ to both sides yields $\sum_{t=1}^{T} l^{(t)}\left(\mathbf{w}^{(t+1)}\right) \leq l^{(T)}\left(\mathbf{w}^{(T+1)}\right) + \sum_{t=1}^{T-1} l^{(t)}(\mathbf{u})$
- Since this holds even for $\mathbf{u}=\mathbf{w}^{(T+1)}$,

$\mathbf{w}^{(T+1)}$ is taken to satisfy this

$$\sum_{t=1}^{T} l^{(t)}\left(\mathbf{w}^{(t+1)}\right) \leq \sum_{t=1}^{T} l^{(t)}\left(\mathbf{w}^{(T+1)}\right) \leq \sum_{t=1}^{T} l^{(t)}(\mathbf{u})$$

Follow-the-regularized-leader: An online algorithm with regret bound

- Too aggressive updates might increase regret of FTL
  - Regret bound depends on the sum of decreases of $l^{(t)}$ so far
- Follow-the-Regularized-Leader (FTRL) performs “milder” updates

$$\mathbf{w}^{(t)} = \mathrm{argmin}_{\mathbf{w}\in\mathcal{S}} \sum_{\tau=1}^{t-1} l^{(\tau)}(\mathbf{w}) + R(\mathbf{w})$$

regularization term

- Lemma:

$$^\forall \mathbf{u},\ \sum_{t=1}^{T}\left(l^{(t)}\left(\mathbf{w}^{(t)}\right) - l^{(t)}(\mathbf{u})\right) \leq R(\mathbf{u}) - R\left(\mathbf{w}^{(1)}\right) + \sum_{t=1}^{T}\left(l^{(t)}\left(\mathbf{w}^{(t)}\right) - l^{(t)}\left(\mathbf{w}^{(t+1)}\right)\right)$$

Proof of the FTRL lemma: Reuse of the FTL lemma

- FTRL on $l^{(1)}, l^{(2)}, \ldots \overset{\text{equivalent}}{\Longleftrightarrow}$ **FTL** on $l^{(0)} = R(\mathbf{w}), l^{(1)}, l^{(2)}, \ldots$
  - Since the **FTL** update is
$$\mathbf{w}^{(t)} = \operatorname{argmin}_{\mathbf{w}\in\mathcal{S}} \sum_{\tau=0}^{t-1} l^{(\tau)}(\mathbf{w})$$
$$= \operatorname{argmin}_{\mathbf{w}\in\mathcal{S}} \sum_{\tau=1}^{t-1} l^{(\tau)}(\mathbf{w}) + R(\mathbf{w})$$
- Applying the previous FTL lemma, we have additional terms
$$l^{(0)}(\mathbf{u}) - l^{(0)}\left(\mathbf{w}^{(1)}\right) = R(\mathbf{u}) - R\left(\mathbf{w}^{(1)}\right)$$

Example of FTRL update: Online linear optimization

- Assume:
  - Linear loss function: $l^{(t)}\left(\mathbf{w}^{(t)}\right) = \left\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)} \right\rangle$
  - Standard L_2-regularization term: $R(\mathbf{w}) = \frac{1}{2\eta}\|\mathbf{w}\|_2^2$
- FTRL update: $\mathbf{w}^{(t+1)} = \operatorname{argmin}_{\mathbf{w}\in\mathbb{R}^d} \sum_{\tau=1}^{t} \left\langle \mathbf{w}, \mathbf{z}^{(\tau)} \right\rangle + \frac{1}{2\eta}\|\mathbf{w}\|_2^2$
  - i.e. $\mathbf{w}^{(t+1)} = -\eta \sum_{\tau=1}^{t} \mathbf{z}^{(\tau)} = \mathbf{w}^{(t)} - \eta\mathbf{z}^{(t)}$
  - With no regularization term, $\mathbf{w}^{(t+1)} = -\infty \cdot \operatorname{sign}\left(\sum_{\tau=1}^{t} \mathbf{z}^{(\tau)}\right)$
    - suffers infinite loss

Regret bound for online linear optimization: FTRL enjoys sublinear regret bound

- $\mathrm{Regret}_T(\mathcal{S}) \leq \frac{1}{2\eta}\|\mathbf{w}^*\|_2^2 + \sum_{t=1}^{T}\left(\left\langle \mathbf{w}^{(t)}, \mathbf{z}^{(t)}\right\rangle - \left\langle \mathbf{w}^{(t+1)}, \mathbf{z}^{(t)}\right\rangle\right)$

$$= \frac{1}{2\eta}\|\mathbf{w}^*\|_2^2 + \sum_{t=1}^{T}\left\langle \mathbf{w}^{(t)} - \mathbf{w}^{(t+1)}, \mathbf{z}^{(t)}\right\rangle$$

$$= \frac{1}{2\eta}\|\mathbf{w}^*\|_2^2 + \sum_{t=1}^{T}\left\langle \eta\mathbf{z}^{(t)}, \mathbf{z}^{(t)}\right\rangle = \frac{1}{2\eta}\|\mathbf{w}^*\|_2^2 + \eta\sum_{t=1}^{T}\left\|\mathbf{z}^{(t)}\right\|_2^2$$

- Optimizing η, $\eta = \frac{\|\mathbf{w}^*\|_2^2}{L\sqrt{2T}}$, where $\frac{1}{T}\sum_{t=1}^{T}\left\|\mathbf{z}^{(t)}\right\|_2^2 \leq L$, gives a sublinear bound: $\mathrm{Regret}_T(\mathcal{S}) \leq \|\mathbf{w}^*\|_2^2\, L\sqrt{2T}$

Doubling trick: Getting rid of dependence of the regret bound on T

- Obtaining $\mathrm{O}\left(\sqrt{2T}\right)$ regret bound requires us to know the total number of rounds T; we would get rid the dependence

- Suppose we have an algorithm A with regret bound of $\alpha\sqrt{T}$

- Doubling trick:
  - Make T double when the round reaches T
  - i.e. for each epoch $m = 1, 2, \dots,$ run A for $\tilde{T} = 2^m$ rounds

- Total regret is bounded by

$$\sum_{m=1}^{\lceil \log_2 T\rceil} \alpha\sqrt{\tilde{T}} = \sum_{m=1}^{\lceil \log_2 T\rceil} \alpha\sqrt{2^m} \leq \frac{\sqrt{2}}{\sqrt{2}-1}\alpha\sqrt{T}$$

Online gradient descent: Online learning algorithm with convex loss function

- Online gradient descent
  - Hyper-parameter (learning rate): $\eta > 0$
  - Initialization: $\mathbf{w}^{(t)} = \mathbf{0}$
- At each round $t = 1, 2, \dots, T$
  1. Submit a parameter vector $\mathbf{w}^{(t)} \in \mathcal{S}$ (convex set e.g. $\mathbb{R}^D$)
  2. Receive a convex loss function $l^{(t)}: \mathcal{S} \to \mathbb{R}$ and suffer loss $l^{(t)}\left(\mathbf{w}^{(t)}\right)$
  3. Update parameter $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{z}^{(t)}$, where $\mathbf{z}^{(t)} \in \partial l^{(t)}\left(\mathbf{w}^{(t)}\right)$ (subgradients)

[Supplement]: Subgradient

- A function f: S (convex set) $\to \mathbb{R}$ is a convex function iff $\forall \mathbf{u} \in S$, there exists $\mathbf{z}$ such that
$$\forall \mathbf{u} \in S, f(\mathbf{u}) \geq f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \mathbf{z} \rangle$$
- $\mathbf{z}$ is called a *subgradient* of f at $\mathbf{w}$, and denote the set of subgradients by $\partial f(\mathbf{w})$
- If f is differentiable at $\mathbf{w}$, $\partial f(\mathbf{w})$ has only a single element $\nabla l(\mathbf{w})$ called gradient

Regret bound of online gradient descent: OGD also enjoys sublinear regret bound

- Lemma: Regret bound of online gradient descent is

$$\mathrm{Regret}_T(\mathcal{S}) \le \frac{1}{2\eta}\|\mathbf{w}^*\|_2^2 + \eta\sum_{t=1}^{T}\left\|\mathbf{z}^{(t)}\right\|_2^2$$

optimal $\mathbf{w}$

norm of subgradient

- Optimizing η, $\eta = \frac{\|\mathbf{w}^*\|_2^2}{L\sqrt{2T}}$, where $\frac{1}{T}\sum_{t=1}^{T}\left\|\mathbf{z}^{(t)}\right\|_2^2 \le L$, we have a sublinear bound: $\mathrm{Regret}_T(\mathcal{S}) \le \|\mathbf{w}^*\|_2^2\, L\sqrt{2T}$
- Same results as those for regret bounds for online linear optimization

Proof of regret bound of online gradient descent: Reduction to online linear optimization

optimal $\mathbf{w}$

- For convex loss l,
  $l(\mathbf{w}^*) \ge l(\mathbf{w}) + \langle \mathbf{w}^* - \mathbf{w}, \mathbf{z}\rangle, \mathbf{z} \in \partial l(\mathbf{w}) \Rightarrow l(\mathbf{w}) - l(\mathbf{w}^*) \le \langle \mathbf{w} - \mathbf{w}^*, \mathbf{z}\rangle$
- Regret is bounded above

$$\mathrm{Regret}_T(\mathcal{S}) = \sum_{t=1}^{T}\left(l^{(t)}(\mathbf{w}^{(t)}) - l^{(t)}(\mathbf{w}^*)\right) \le \sum_{t=1}^{T}\left(\langle \mathbf{w}^{(t)}, \mathbf{z}^{(t)}\rangle - \langle \mathbf{w}^*, \mathbf{z}^{(t)}\rangle\right)$$

  – This is exactly what we bounded in the online linear optimization using FTRL
- OGD is equivalent to FTRL by taking $\mathbf{z}^{(t)} \in \partial l^{(t)}\left(\mathbf{w}^{(t)}\right)$, results in the same regret bounds as those of FTRL
  – Remember the FTRL update: $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta\mathbf{z}^{(t)}$

Convex surrogate: Regret bound for non-convex loss

- Our analysis relied on the convexity of $l^{(t)}$; what if it is not?
- Consider a *convex* upper bound $\hat{l}^{(t)}$ such that $l^{(t)} \leq \hat{l}^{(t)}$
- Running the online gradient descent using $\hat{l}^{(t)}$ gives regret bound $\sum_{t=1}^{T}\left(\hat{l}^{(t)}\left(\mathbf{w}^{(t)}\right) - \hat{l}^{(t)}(\mathbf{w}^{*})\right) \leq \|\boldsymbol{w}^{*}\|_2^2\, L\sqrt{2T}$
- Combined with $l^{(t)}\left(\mathbf{w}^{(t)}\right) \leq \hat{l}^{(t)}\left(\mathbf{w}^{(t)}\right)$, we get

$$\sum_{t=1}^{T} l^{(t)}\left(\mathbf{w}^{(t)}\right) \leq \sum_{t=1}^{T} \hat{l}^{(t)}(\mathbf{w}^{*}) + \|\boldsymbol{w}^{*}\|_2^2\, L\sqrt{2T}$$

Perceptron algorithm: Online classification learning with mistake bound

- Perceptron update:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + y^{(t)}\mathbf{x}^{(t)} \cdot 1_{\left[y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle \leq 0\right]}$$

- Non-convex loss function 0-1 loss (Online classification)

$$l^{(t)}\left(\mathbf{w}^{(t)}\right) = 1_{\left[y^{(t)}\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle \leq 0\right]}$$

- Lemma: If there exists $\mathbf{w}^{*}$ such that $\forall t, y^{(t)}\left\langle \mathbf{w}^{*}, \mathbf{x}^{(t)}\right\rangle \geq 1$, mistake bound of perceptron is

$$m \leq 2R^2\|\mathbf{w}^{*}\|_2^2,$$

where $\left\|\mathbf{x}^{(t)}\right\|_2^2 \leq R^2$

number of mistakes

Perceptron algorithm:
Equivalent to ODG with surrogate loss

- Define convex surrogate $\hat{l}^{(t)}$ as $\hat{l}^{(t)} = 1 - y^{(t)}\langle \mathbf{w}^{(t)},\ \mathbf{x}^{(t)}\rangle$ if the perceptron makes a mistake, and $\hat{l}^{(t)} = 0$ if not
- Online gradient descent with $\hat{l}^{(t)}$ is equivalent to perceptron
  – OGD:
  $$\begin{aligned}\mathbf{w}^{(t+1)} &= \mathbf{w}^{(t)} + \eta y^{(t)}\mathbf{x}^{(t)} \cdot 1_{[y^{(t)}\langle \mathbf{w}^{(t)},\mathbf{x}^{(t)}\rangle \le 0]} \\ &= \eta \textstyle\sum_{t=1}^{T} y^{(t)}\mathbf{x}^{(t)} \cdot 1_{[y^{(t)}\langle \mathbf{w}^{(t)},\mathbf{x}^{(t)}\rangle \le 0]}\end{aligned}$$
  – Perceptron:
  $$\begin{aligned}\mathbf{w}^{(t+1)} &= \mathbf{w}^{(t)} + y^{(t)}\mathbf{x}^{(t)} \cdot 1_{[y^{(t)}\langle \mathbf{w}^{(t)},\mathbf{x}^{(t)}\rangle \le 0]} \\ &= \textstyle\sum_{t=1}^{T} y^{(t)}\mathbf{x}^{(t)} \cdot 1_{[y^{(t)}\langle \mathbf{w}^{(t)},\mathbf{x}^{(t)}\rangle \le 0]}\end{aligned}$$
  no effect on prediction
  – We can take arbitrary η since $\mathrm{sign}(\langle \mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle) = \mathrm{sign}(\langle \eta\mathbf{w}^{(t)}, \mathbf{x}^{(t)}\rangle)$

Proof of perceptron mistake bound (1/2):
Use regret bound of OGD with surrogate loss

- Online gradient descent with $\hat{l}^{(t)}$ gives
$$\mathrm{Regret}_T(\mathcal{S}) \le \frac{1}{2\eta}\|\mathbf{w}^*\|_2^2 + \eta \sum_{t=1}^{T} \left\|y^{(t)}\mathbf{x}^{(t)}\right\|_2^2 \cdot 1_{[y^{(t)}\langle \mathbf{w}^{(t)},\mathbf{x}^{(t)}\rangle \le 0]}$$
$$\left\|y^{(t)}\mathbf{x}^{(t)}\right\|_2^2 = \left\|\mathbf{x}^{(t)}\right\|_2^2 \le R^2$$
- On the other hand,
$$\mathrm{Regret}_T(\mathcal{S}) = \sum_{t=1}^{T}\left(\hat{l}^{(t)}(\mathbf{w}^{(t)}) - \hat{l}^{(t)}(\mathbf{w}^*)\right) \ge m$$
  – since $\sum_t \hat{l}^{(t)}(\mathbf{w}^{(t)}) \ge \sum_t l^{(t)}(\mathbf{w}^{(t)}) = m$, and $\sum_{t=1}^{T} \hat{l}^{(t)}(\mathbf{w}^*) = 0$ (since $\forall t, y^{(t)}\langle \mathbf{w}^*,\ \mathbf{x}^{(t)}\rangle \ge 1$)
- Connecting the two inequalities yields $m \le \frac{1}{2\eta}\|\mathbf{w}^*\|_2^2 + \eta m R^2$

Proof of perceptron mistake bound (2/2): Optimize the bound

- We have $m \leq \frac{1}{2\eta} \|\mathbf{w}^*\|_2^2 + \eta m R^2$
- Minimizing the r.h.s. finds $\eta = \frac{\|\mathbf{w}^*\|_2}{R\sqrt{2m}}$, which results in $m \leq R\sqrt{2m}\, \|\mathbf{w}^*\|_2$
  - Remember we do not have to determine η actually
- $m \leq 2R^2 \|\mathbf{w}^*\|_2^2$