# 6.7900 Machine Learning (Fall 2024)

## Lecture 9: online learning, regret

## (supporting slides)

# Outline

- We have discussed statistical models for linear regression and classification, the concepts of empirical risk minimization and Bayesian estimation

- These formulations assume that we are given a training set ahead of time, we learn from it (e.g., minimize empirical risk), then use the resulting predictor on test examples that are revealed later

- Today we focus on **on-line learning** where examples come one at a time and we have to make a prediction (and incur a loss) for each new example as they come. The correct answer is revealed after each prediction and we can learn from these answers to improve our predictions.
  - e.g., trading, advertising, adapting to changing circumstances, etc.

# On-line prediction game

- We can think of on-line learning as a prediction game between nature (who selects the examples to come) and learner (that must adjust to do well)

For t = 1,2,3,…

Nature reveals input x

Learner makes a prediction on x

Nature reveals the correct output

Learner suffers loss

Learner updates its model

- **We do not make any statistical assumptions about the sequence of examples / labels**; they can be chosen adversarially even with the knowledge of how we learn and make predictions!

# On-line learning: regret

- Since the examples can be chosen adversarially, we need to redefine how we measure learner's performance
- E.g., if learner has to predict a $\{0,1\}$ label, nature could always choose the label to be the opposite of what we predict in each round. The learner would suffer cumulative loss proportional to T after T rounds…

- We therefore measure learner's performance not in absolute terms but in terms of **regret,** i.e., how well it does relative to some comparison class of methods, e.g., another method that has access to more information

# On-line learning: regret example

- At round t, learner has parameters $\theta^t \in B \subseteq \mathbb{R}^d$ (e.g., bounded subset)
- Nature chooses $x^t \in \mathbb{R}^d$
- Learner predicts using, e.g., logistic model: $P(y = 1 \,|\, x^t, \theta^t) = \sigma({\theta^t}^T x^t)$
- Nature reveals the correct output $y^t \in \{0,1\}$
- Learner suffers loss $L(x^t, y^t, \theta^t) = -\log P(y^t \,|\, x^t, \theta^t)$
- Learner updates parameters to be $\theta^{t+1}$ for the next round

- Learner's **regret** after T rounds:

$$R_T = \underbrace{\sum_{t=1}^{T} L(x^t, y^t, \theta^t)}_{\text{Learner's loss at each round}} - \underbrace{\min_{\theta \in B} \sum_{t=1}^{T} L(x^t, y^t, \theta)}_{\text{loss of the best fixed predictor in the same class chosen with hindsight}}$$

# A warm-up exercise

- As a warm-up, let’s try to constrain the problem enough so that it becomes easily learnable with simple methods

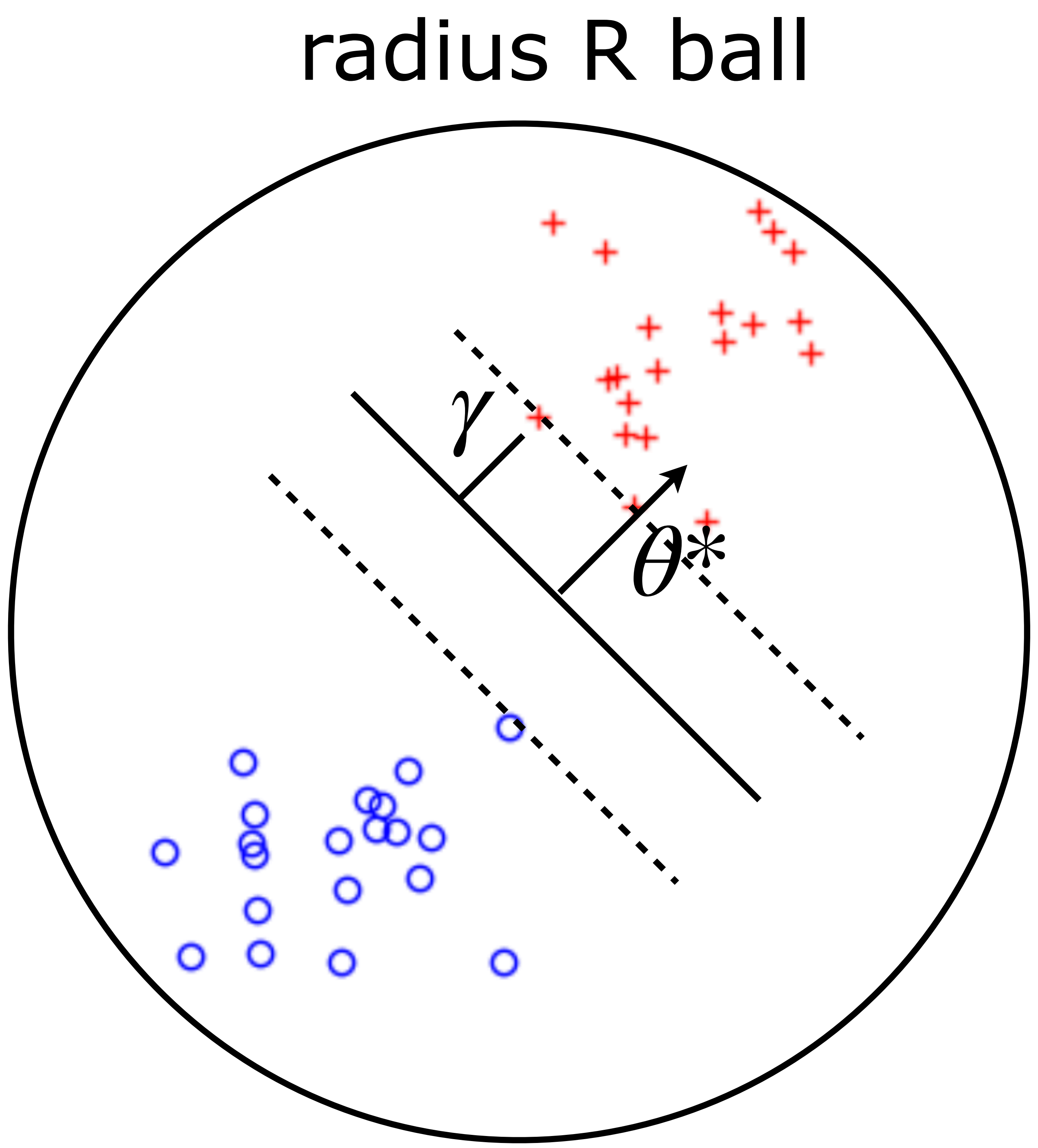
# A warm-up exercise

- Consider a restricted on-line classification task where we dictate ahead of time that a good solution must exists. E.g.,

- $\|x^i\| \leq R$, $y^i \in \{-1,1\}$ for all $i = 1,2,3,\ldots$
- Examples are linearly separable and there exists $\theta*$ (unknown to us) such that

$$\gamma \leq \frac{y^i(\theta*)^T x^i}{\|\theta*\|}, \ \ \forall i$$

i.e., the smallest distance of any example to the decision boundary defined by $\theta*$ is $\gamma$ (margin)



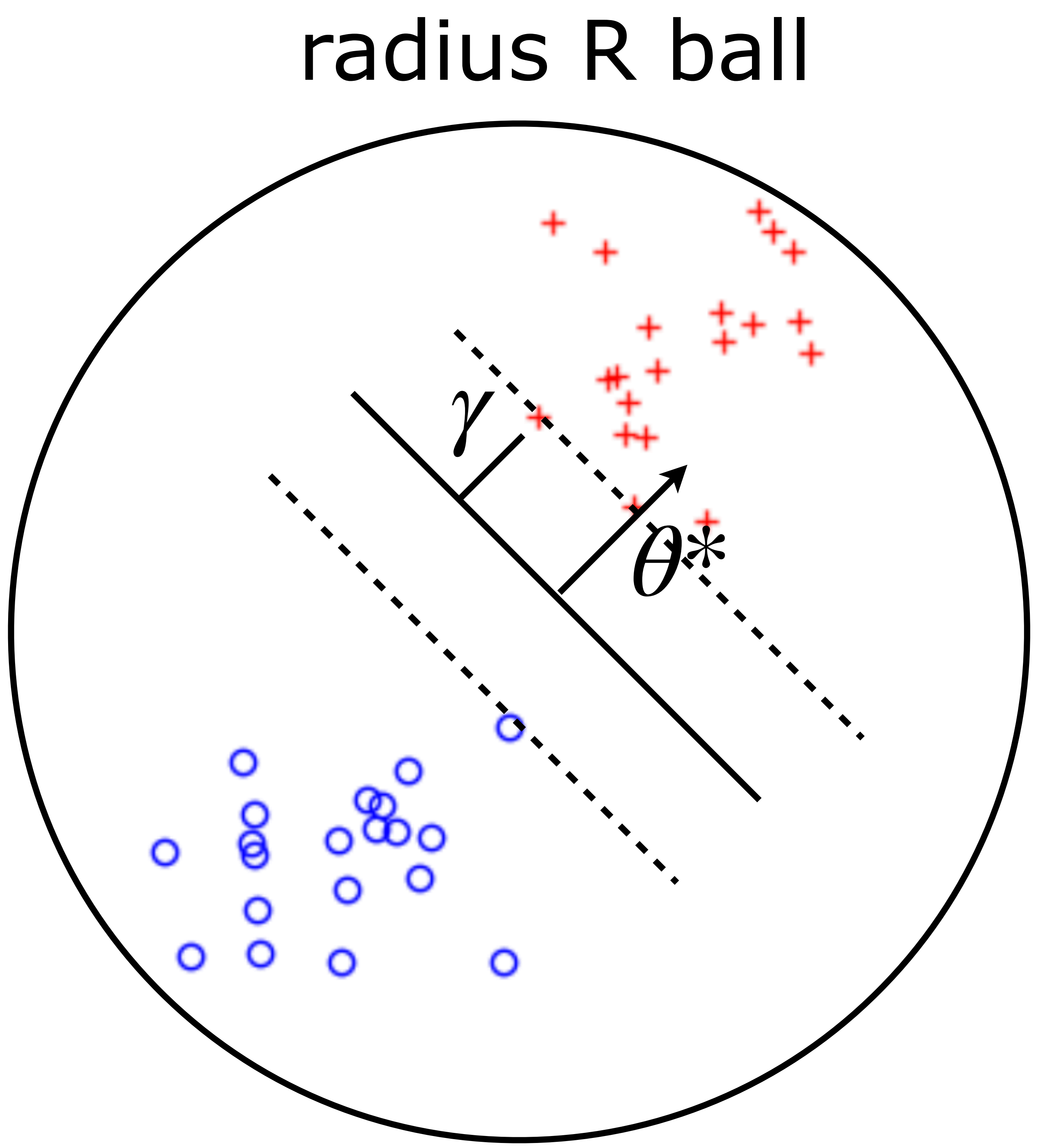
# A warm-up exercise

- Consider a restricted on-line classification task where we dictate ahead of time that a good solution must exists. E.g.,

- $\|x^i\| \leq R$, $y^i \in \{-1,1\}$ for all $i = 1,2,3,\ldots$
- Examples are linearly separable and there exists $\theta*$ (unknown to us) such that

$$\gamma \leq \frac{y^i(\theta*)^T x^i}{\|\theta*\|}, \ \forall i$$

i.e., the smallest distance of any example to the decision boundary defined by $\theta*$ is $\gamma$ (margin)




- Clearly, the best linear classifier chosen with hindsight makes zero errors on any sequence consistent with these assumptions

# Simple perceptron algorithm (1950's)

- Start with $\theta^1 = 0$ (vector)
- For t=1,2,3,…

  Nature gives $x^t$

  Learner predicts $\hat{y}^t = \text{sign}\left((\theta^t)^T x^t\right)$

  Nature reveals $y^t$

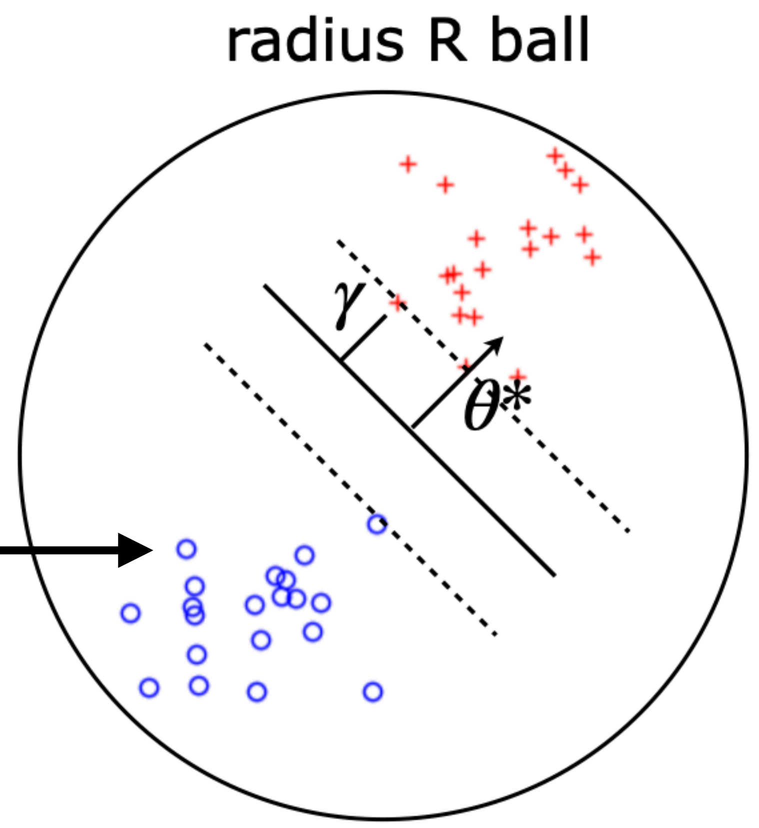
  If $\hat{y}^t \neq y^t$ (mistake)

  $\theta^{t+1} = \theta^t + y^t x^t$

  else

  $\theta^{t+1} = \theta^t$

# Simple perceptron algorithm (1950's)




- Start with $\theta^1 = 0$ (vector)
- For t=1,2,3,…

  Nature gives $x^t$ (outside of the margin)

  Learner predicts $\hat{y}^t = \text{sign}\big((\theta^t)^T x^t\big)$

  Nature reveals $y^t$ $= \text{sign}\big((\theta^*)^T x^t\big)$ (nature has to abide by our assumptions)

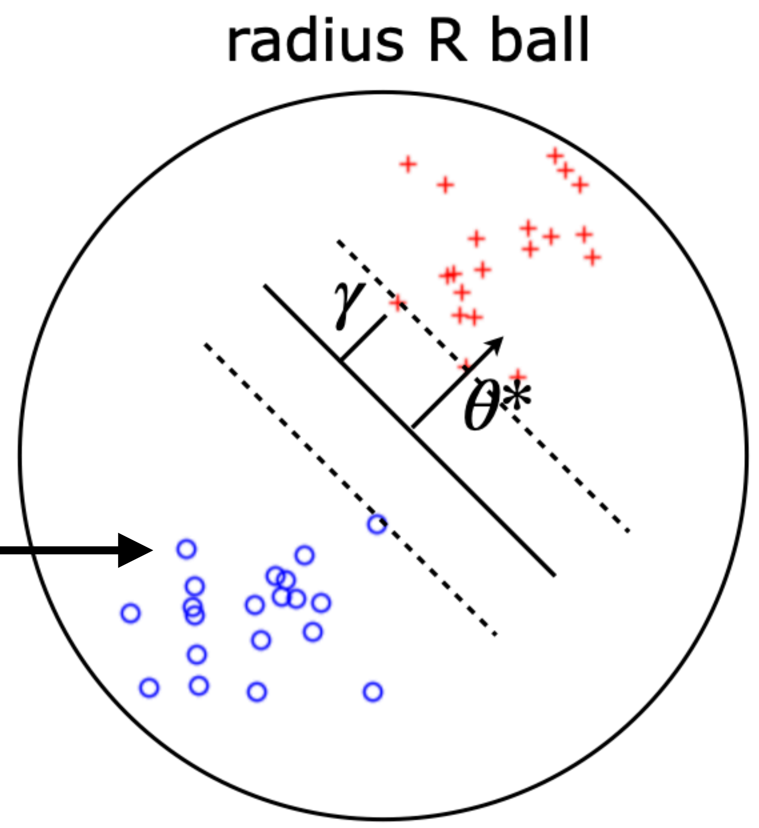
  If $\hat{y}^t \neq y^t$ (mistake)

  $$\theta^{t+1} = \theta^t + y^t x^t$$

  else

  $$\theta^{t+1} = \theta^t$$

# Simple perceptron algorithm (1950's)




- Start with $\theta^1 = 0$ (vector)
- For t=1,2,3,…

  Nature gives $x^t$ (outside of the margin)

  Learner predicts $\hat{y}^t = \text{sign}\left((\theta^t)^T x^t\right)$

  Nature reveals $y^t$ $= \text{sign}\left((\theta^*)^T x^t\right)$ (nature has to abide by our assumptions)

  If $\hat{y}^t \neq y^t$ (mistake)

  $$\theta^{t+1} = \theta^t + y^t x^t$$

  else

  $$\theta^{t+1} = \theta^t$$

**Theorem:** The perceptron algorithm makes at most

$$\frac{R^2}{\gamma^2}$$ independent of dim!!

mistakes on any sequence of examples and labels satisfying our assumptions

# Generalization to online convex optimization

- At round t, learner has parameters $\theta^t \in B \subseteq \mathbb{R}^d$ (e.g., bounded subset)
- Nature chooses $x^t \in \mathbb{R}^d$
- Learner predicts using, e.g., logistic model: $P(y=1\,|\,x^t,\theta^t) = \sigma({\theta^t}^T x^t)$
- Nature reveals the correct output $y^t \in \{0,1\}$
- Learner suffers loss $L(x^t, y^t, \theta^t) = -\log P(y^t\,|\,x^t, \theta^t)$
- Learner updates parameters to be $\theta^{t+1}$ for the next round

# Generalization to online convex optimization

- At round t, learner has parameters $\theta^t \in B \subseteq \mathbb{R}^d$ (e.g., bounded subset)
- Nature chooses $x^t \in \mathbb{R}^d$
- Learner predicts using, e.g., logistic model: $P(y = 1 \mid x^t, \theta^t) = \sigma(\theta^{tT} x^t)$
- Nature reveals the correct output $y^t \in \{0,1\}$
- Learner suffers loss $L(x^t, y^t, \theta^t) = -\log P(y^t \mid x^t, \theta^t)$
- Learner updates parameters to be $\theta^{t+1}$ for the next round

- By selecting $(x^t, y^t)$ nature effectively selects a loss function for us

$$L(x^t, y^t, \theta) = -\log P(y^t \mid x^t, \theta) \equiv f_t(\theta)$$

convex in $\theta$

- So we can redefine the online learning problem as learner choosing parameters, nature selecting convex loss functions in response

# A modern version: online convex optimization

- Let $\theta \in B$ where $B$ is a set of possible parameters

- For t=1,2,3,…

  Learner selects $\theta^t \in B$

  Nature reveals a convex loss function $f_t(\cdot)$

  Learner incurs loss $f_t(\theta^t)$

- Learner's goal is then to minimize regret relative to the best fixed $\theta \in B$ chosen with hindsight for the same sequence of losses

$$R_T = \sum_{t=1}^{T} f_t(\theta^t) - \min_{\theta \in B} \sum_{t=1}^{T} f_t(\theta)$$

# Online convex optimization: restrictions

- Similar to assuming a margin earlier, we need to impose some additional restrictions on the loss functions as well as the parameter space so as to make this online task learnable (sub-linear regret)

- For example, we can assume

1. $\theta \in B \subseteq \mathbb{R}^d$, $B$ is a convex, bounded set (bounded by radius R ball)
2. $f_i(\theta)$ are convex, Lipschitz continuous with constant L (e.g., if smooth, gradients remain bounded by L)

   $$\|f_i(\theta) - f_i(\theta')\| \leq L\|\theta - \theta'\|, \quad \forall \theta, \theta' \in B$$

# Would ERM solve our problem?

- For t=1,2,3,…

  Learner selects $\theta^t = \text{argmin}_{\theta \in B} \sum_{s=1}^{t-1} f_s(\theta)$ minimize empirical losses revealed so far

  Nature reveals a convex loss function $f_t(\cdot)$

  Learner incurs loss $f_t(\theta^t)$

- Should be pretty effective as a procedure for iid examples, i.e., when the convex loss functions that the nature selects are random of the form

$$f_t(\theta) = -\log P(y^t \,|\, x^t, \theta), \quad (x^t, y^t) \sim P(x, y)$$

  (nature samples a test point at random and calculates the loss based on it)
- Does it work if this assumption is violated?

# Would ERM solve our problem?

## (a.k.a "follow the leader")

- For t=1,2,3,…

  Learner selects $\theta^t = \text{argmin}_{\theta \in B} \sum_{s=1}^{t-1} f_s(\theta)$ minimize empirical losses revealed so far

  Nature reveals a convex loss function $f_t(\cdot)$

  Learner incurs loss $f_t(\theta^t)$

- When nature doesn't conform to iid losses this ERM-like procedure can be pretty bad. E.g.,

- Let $\theta \in [-1,1]$ (scalar), $\theta^1 = 0$
- Nature's choices are: $f_1(\theta) = 0.5\theta$ and thereafter for even t, $f_t(\theta) = -\theta$, and for odd t, $f_t(\theta) = \theta$ (these are well-behaving, convex losses)
- Learner's losses become: 0, 1, 1, 1, 1, ….
- The competitor's cumulative loss after T steps is always -0.5 … so $R_T = T - 1 + 0.5$

# A projected gradient approach

## (~ "follow the regularized leader")

- Initialize $\theta^1 \in B$
- For t=1,2,3,…

  Nature reveals a convex loss function $f_t(\cdot)$

  Learner incurs loss $f_t(\theta^t)$

  Learner updates parameters according to projected gradient descent

  $\hat{\theta} = \theta^t - \eta_t \nabla_\theta f_t(\theta)_{|\theta=\theta^t}$ gradient update (may go outside B)

  $\theta^{t+1} = \text{argmin}_{\theta \in B} ||\hat{\theta} - \theta||$ projection back to B

# A projected gradient approach

## (~ "follow the regularized leader")

- Initialize $\theta^1 \in B$
- For t=1,2,3,…

  Nature reveals a convex loss function $f_t(\cdot)$

  Learner incurs loss $f_t(\theta^t)$

  Learner updates parameters according to projected gradient descent

  $\hat{\theta} = \theta^t - \eta_t \nabla_\theta f_t(\theta)_{|\theta=\theta^t}$ gradient

  $\theta^{t+1} = \text{argmin}_{\theta \in B} ||\hat{\theta} - \theta||$ proj

**Theorem:** Under the assumptions stated earlier, and if $\eta_t = R/(L\sqrt{t})$, the projected gradient algorithm has regret bounded by

$$R_T \leq LR\sqrt{T}$$

# Other examples of regret problems

- **Bernoulli bandit problem:**
- We have k choices (arms) that we can select. Each arm k, if selected, gives a random binary {0,1} reward with mean $\mu_k$; the underlying means are unknown
- We would like to quickly learn to select the best arm $\mu^* = \max_k \mu_k$

- At each round t=1,2,3,…

  Learner can select $k_t \in \{1,\ldots,K\}$

  The outcome $r_t \in \{0,1\}$ is chosen at random with mean $\mu_{k_t}$

  Learner receives $r_t$ and can update its strategy

- The learner's regret (here averaged over nature's choices) is

$$E\{R_T\} = \sum_{t=1}^{T} (\mu^* - \mu_{k_t})$$

# Thompson sampling

- We estimate the underlying mean responses with parameters $\theta_1, \ldots, \theta_K$
- We use a Beta distribution to represent our probability over each $\theta_k$

  $B(\theta_k \mid \alpha_k, \beta_k) \propto \theta_k^{\alpha_k - 1}(1 - \theta_k)^{\beta_k - 1}$

  where initially $\alpha_k = \beta_k = 1$ (uniform prior)

- **TS sampling**

  Learner samples $\hat{\theta}_k \sim B(\theta_k \mid \alpha_k, \beta_k),\ \ k = 1, \ldots, K$

  Learner selects $k_t = \mathrm{argmax}_{k=1,\ldots,K}\ \{\ \hat{\theta}_k\ \}$

  Nature selects $r_t \in \{0,1\}$ by sampling from the underlying Bernoulli model for arm $k_t$

  Learner updates $(\alpha_{k_t}, \beta_{k_t}) \leftarrow (\alpha_{k_t}, \beta_{k_t}) + (r_t, 1 - r_t)$

# Thompson sampling

- We estimate the underlying mean responses with parameters $\theta_1, \ldots, \theta_K$
- We use a Beta distribution to represent our probability over each $\theta_k$

$$B(\theta_k \mid \alpha_k, \beta_k) \propto \theta_k^{\alpha_k - 1}(1 - \theta_k)^{\beta_k - 1}$$

where initially $\alpha_k = \beta_k = 1$ (uniform prior)

- **TS sampling**

Learner samples $\hat{\theta}_k \sim B(\theta_k \mid \alpha_k, \beta_k),\ \ k = 1, \ldots, K$

Learner selects $k_t = \text{argmax}_{k=1,\ldots,K} \{\ \hat{\theta}_k\ \}$

Nature selects $r_t \in \{0,1\}$ by sampling from

Learner updates $(\alpha_{k_t}, \beta_{k_t}) \leftarrow (\alpha_{k_t}, \beta_{k_t}) + (r_t, 1 - r_t$

**Theorem:** The expected regret from using TS sampling over T rounds is

$$E\{R_t\} = O(\sqrt{KT \log(T)})$$

where the expectation is over both nature's and learner's random choices

# References


- Novikoff, “On convergence proofs on perceptrons”, In Proceedings of the Symposium on the Mathematical Theory of Automata, Vol. XII, pages 615–622, 1962.
- E. Hazan, “Introduction to online convex optimization”, https://arxiv.org/abs/1909.05207
- Russo et al., “A tutorial on Thompson Sampling”, https://web.stanford.edu/~bvr/pubs/TS_Tutorial.pdf
- Agrawal et al., “Analysis of Thompson Sampling for the Multi-armed Bandit Problem”, https://proceedings.mlr.press/v23/agrawal12/agrawal12.pdf