Online learning and prediction: just play along!

A pre-Antaragni talk on online learning!

SIGML

Special Interest Group in Machine Learning

Purushottam Kar

Department of CSE

IIT Kanpur

Learning Problems

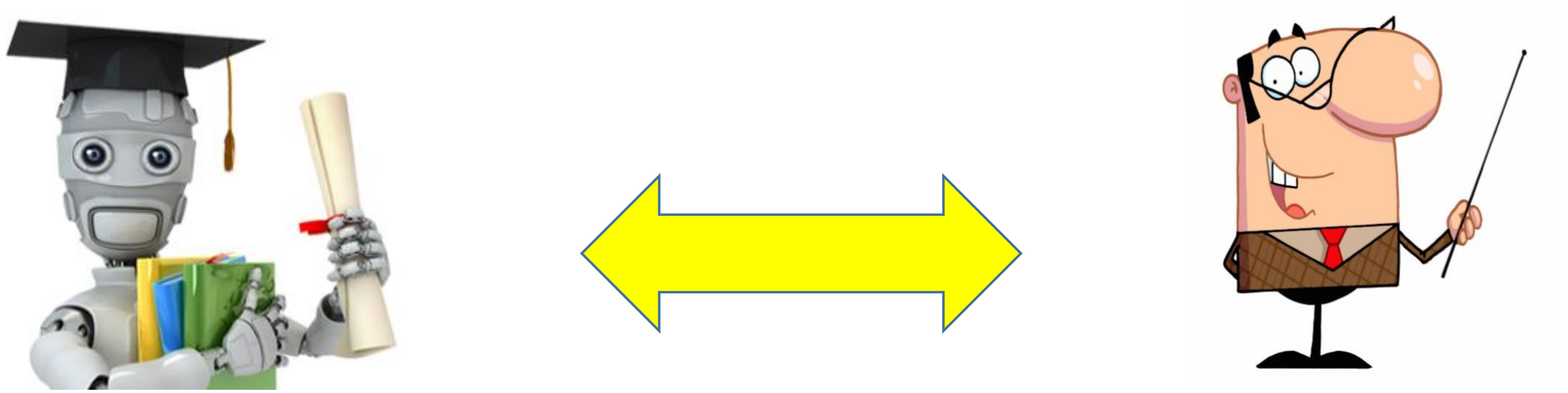
- Portfolio selection:

- Branch prediction:

```
if (num > 0) {
    printf("%d is a positive
    if (num % 2 == 0)
        printf("%d is an even
    else
        printf("%d is an odd
}
else
    printf("%d is a negative
```

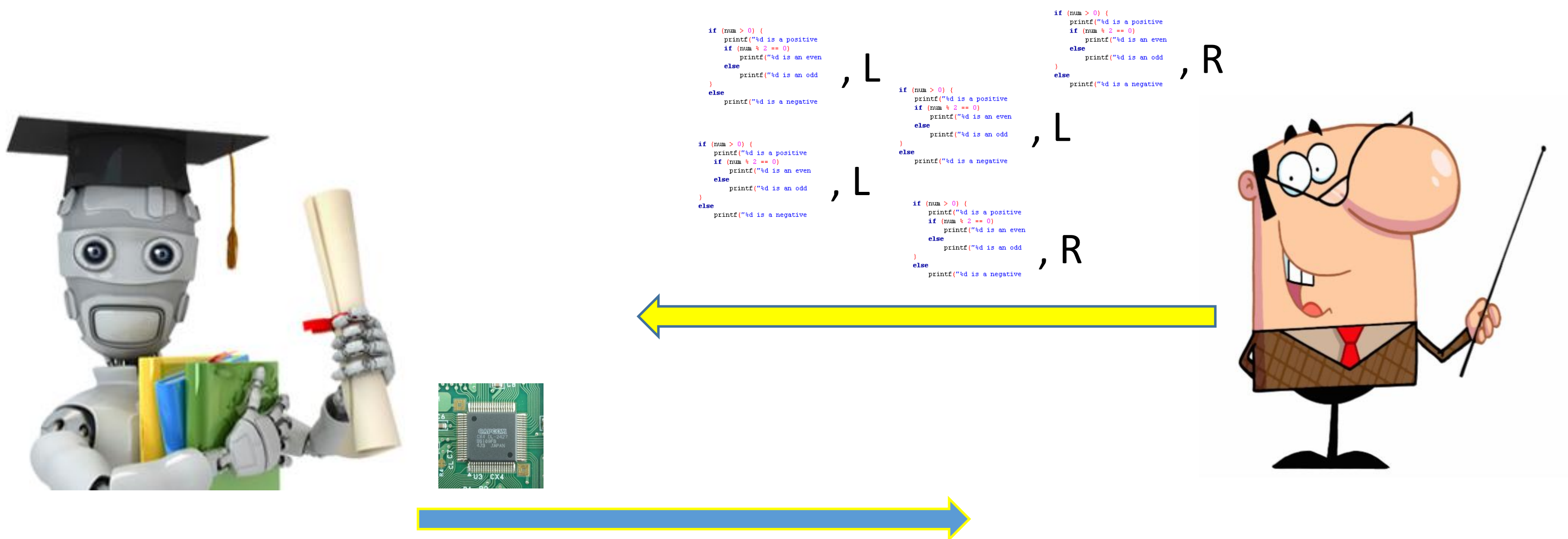
- Click prediction:

Supervised Learning



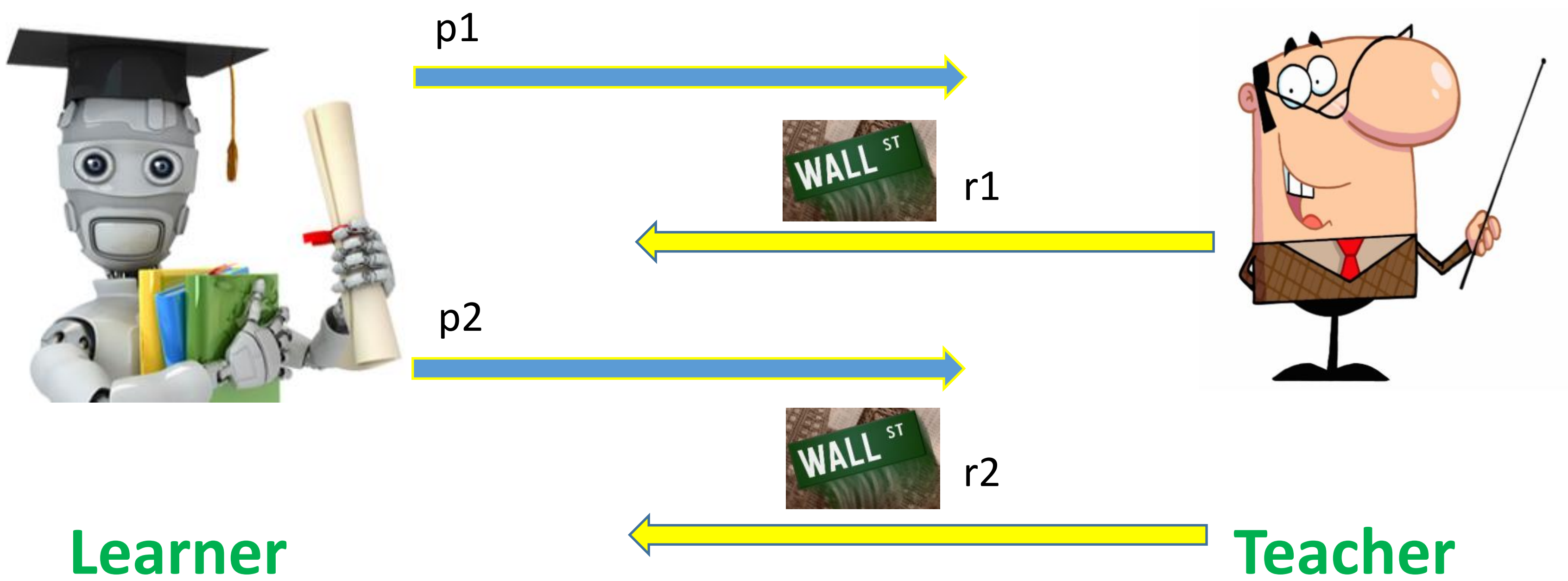
Learner **Teacher**

Passive Supervised Learning



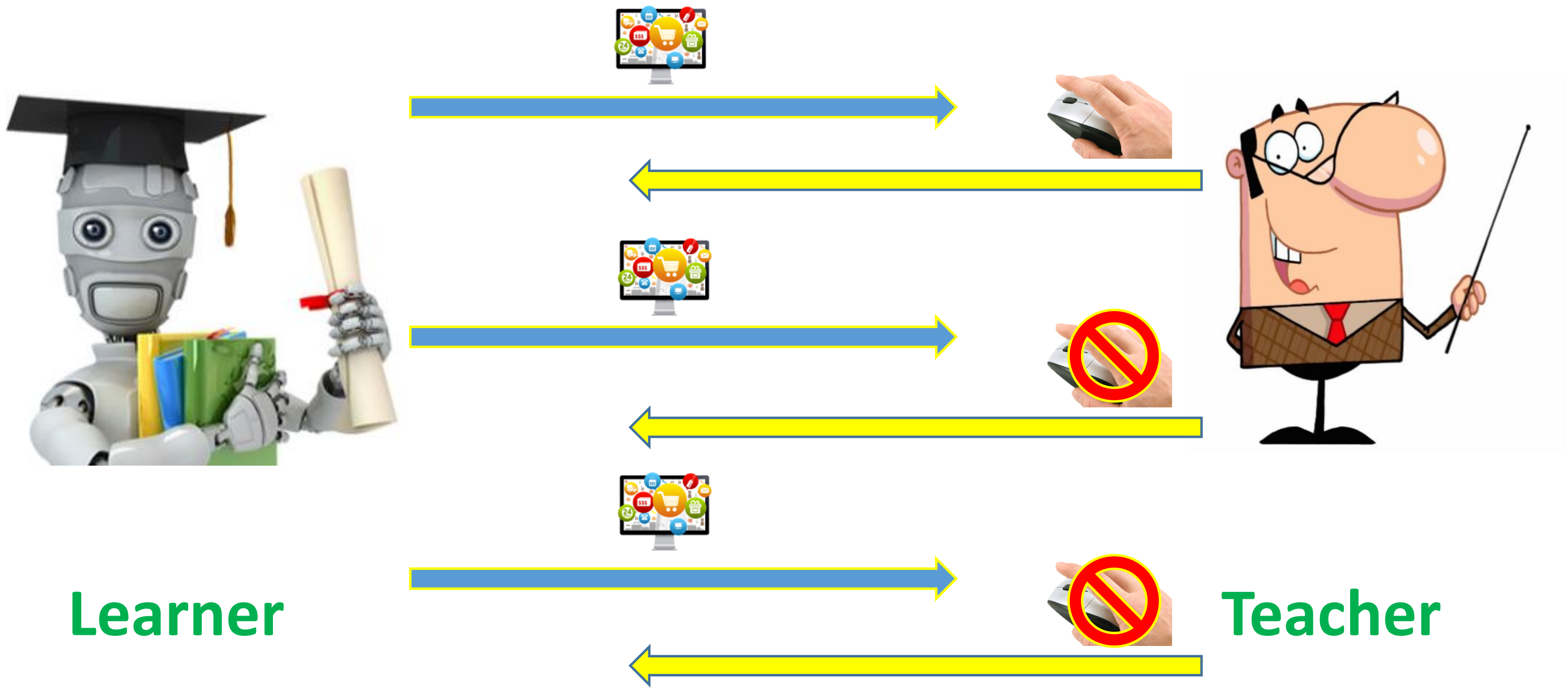
Learner **Teacher**

Online Supervised Learning



Corpus = <p1,r1> * <p2,r2> * … <pT,rT>

Online Supervised Learning



Active Supervised Learning

Learner

```
if (num > 0) {
    printf("%d is a positive
    if (num % 2 == 0)
        printf("%d is an even
    else
        printf("%d is an odd
}
else
    printf("%d is a negative
```

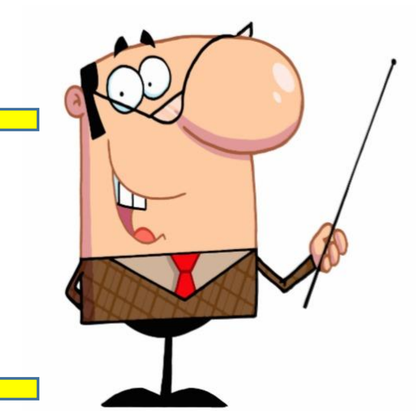
L

```
if (num > 0) {
    printf("%d is a positive
    if (num % 2 == 0)
        printf("%d is an even
    else
        printf("%d is an odd
}
else
    printf("%d is a negative
```

L

```
if (num > 0) {
    printf("%d is a positive
    if (num % 2 == 0)
        printf("%d is an even
    else
        printf("%d is an odd
}
else
    printf("%d is a negative
```

R



Teacher

The Online Learning Model

How we assess Online Learning Algorithms

The Online Learning Model

- An attempt to model an interactive and adaptive environment
  - We have a set of actions $\mathcal{A}$
  - Environment has a set of loss functions $\mathcal{L} = \{\ell: A \to \Re_+\}$

- In each round t
  - We play some action $a_t \in \mathcal{A}$
  - Environment responds with a loss function $\ell_t \in \mathcal{L}$
  - We are forced to incur a loss $\ell_t(a_t)$
  - Environment can adapt to our actions (or even be adversarial)

- Our goal: minimize cumulative loss $\sum_{t=1}^{T} \ell_t(a_t)$
  - Can cumulative loss be brought down to zero : mostly no !
  - More reasonable measure of performance: single best action in hindsight
  - Regret: $R_T \coloneqq \sum_{t=1}^{T} \ell_t(a_t) \; - \min_{a \in \mathcal{A}} \sum_{t=1}^{T} \ell_t(a)$
  - Why is this a suitable notion of performance ?

Making it big in the stock market

- Learning investment profiles
  - Set of actions is the d-dimensional simplex $\mathcal{A} = \{p \in \Re^d, p \geq 0, \|p\|_1 = 1\}$
  - Reward received at t^{th} step is $\langle p^t, r^t \rangle$ where r^t is the return given by market
  - Total reward (assume w.l.o.g. initial corpus is $D = 1$)

$$\prod_{t=1}^{T} \langle p_t, r_t \rangle = \exp\left(\sum_{t=1}^{T} \log\langle p_t, r_t \rangle \right)$$

  - Returns affected by investment, other market factors (adaptive, adversarial)
  - Can think of $\ell(p, r) = -\log\langle p, r \rangle$ as a negative reward or a loss

$$\ell_t(p_t) = -\log\langle p_t, r_t \rangle$$

  - Regret (equivalently) given by

$$\mathcal{R}_T = \sum_{t=1}^{T} \ell(p_t, r_t) - \min_{p \in \mathcal{A}} \sum_{t=1}^{T} \ell(p, r_t)$$

  - Goal: make as much profit as the single best investment profile in hindsight

Simple Online Algorithms

What makes online learning click ?

Online Linear Classification

- Perceptron Algorithm

1. Start with $w_0 = 0$
2. Classify o_t as $\text{sign}\left(w_{t-1}^\top x_{o_t}\right)$
3. If correct classification i.e. $y_t = \text{sign}\left(w_t^\top x_{o_t}\right)$, then let $w_t = w_{t-1}$
4. Else $w_t = w_{t-1} + y_t x_{o_t}$

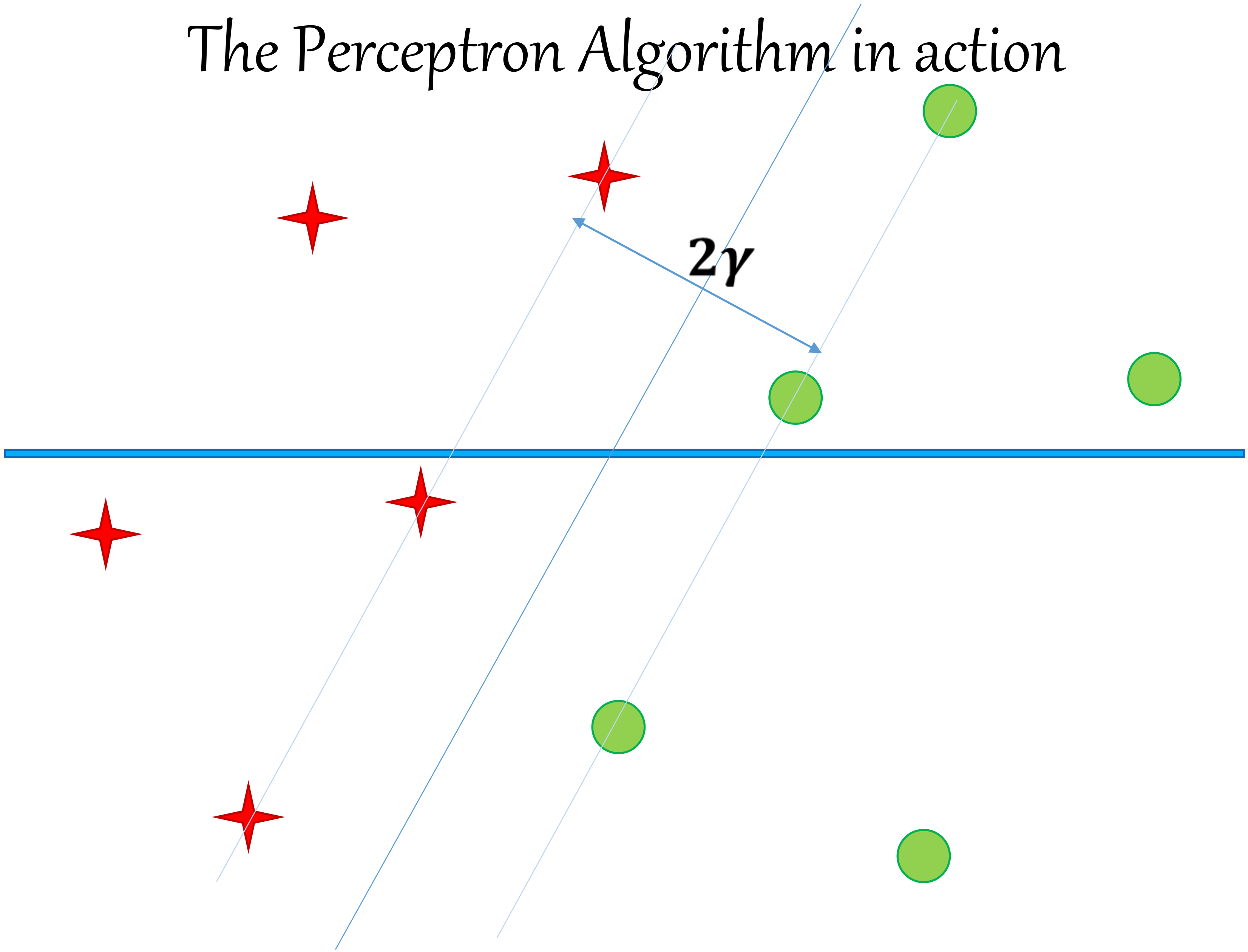
- Loss function $\ell_{0/1}(w, o) = \mathbb{I}\{y_o w^\top x_o < 0\}$ i.e. 1 iff w misclassifies o
- If there exists a perfect linear separator w^* such that $\mathrm{y_t w^*{}^\top x_{o_t}} \geq \gamma$,

$$\mathcal{R}_T = \sum \ell_{0/1}(w_t, o_t) - \sum \ell_{0/1}(w^*, o_t) \leq \frac{1}{\gamma^2}$$

- If there exists an imperfect separator w^* such that $y_t w^{*\top} x_{o_t} \geq \gamma - \xi_t$,

$$\mathcal{R}_T = \sum \ell_{0/1}(w_t, o_t) - \sum \ell_{0/1}(w^*, o_t) \leq \frac{1}{\gamma^2} + \frac{1}{\gamma} \sum \xi_t$$

The Perceptron Algorithm in action



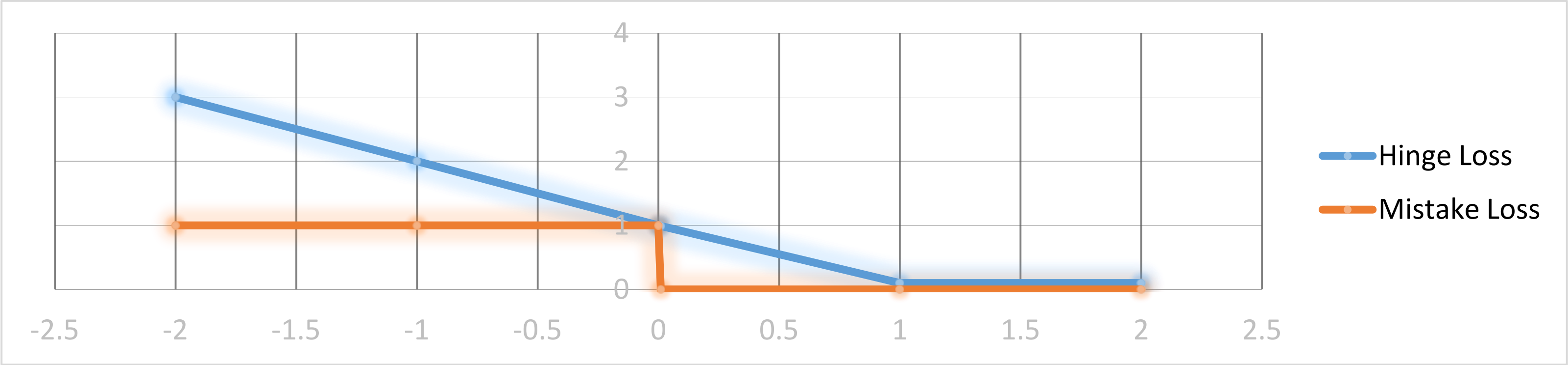
Online Regression

- The Perceptron Algorithm was (almost) a gradient descent algorithm
- Consider the loss function

$$\ell_{\text{hinge}}(w, x) = \max\{1 - yw^\top x, 0\}$$

- $\tilde{\ell}$ is a convex *surrogate* to the mistake function $\ell_{0/1}(w, x) = \mathbb{I}\{yw^\top x < 0\}$

$$\ell_{\text{hinge}}(w, x) \geq \ell_{0/1}(w, x)$$



- When perceptron makes a mistake i.e. $\ell_{0/1}(w, x) = 1$, we have

$$\nabla_w \ell_{hinge}(w, x) = -yx$$

- Thus the perceptron update step $w_t = w_{t-1} + y_t x_{o_t}$ is a gradient step !

Online Regression via Online Gradient Descent

- Suppose we are taking actions $a_t \in \mathcal{A}$ and receiving losses $\ell_t \in \mathcal{L}$
  - Assume that all loss function $\ell_t: \mathcal{A} \to \Re_+$ are convex and Lipchitz
  - Examples $\ell_t(a) = (a^\top x_t - y_t)^2$, $\ell_t(a) = -\log(a^\top x_t)$, $\ell_t(a) = [1 - y_t a^\top x_t]_+$

- Online Gradient Descent (for linear predictions problems)

1. Start with $a_0 = 0$
2. Receive object x_t and predict value $a_{t-1}^\top x_t$ for object x_t
3. Receive loss function ℓ_t and update $a_t = a_{t-1} - \frac{1}{\sqrt{t}} \nabla_a \ell_t(a_{t-1})$
   - Some more work needed to ensure that $a_t \in \mathcal{A}$ as well

- We can ensure that

$$\mathrm{R_T} = \sum_{t=1}^{T} \ell_t(a_t) - \min_{a \in \mathcal{A}} \sum_{t=1}^{T} \ell_t(a) \leq \mathcal{O}(\sqrt{T})$$