CMSC5741 Big Data Tech. & Apps.

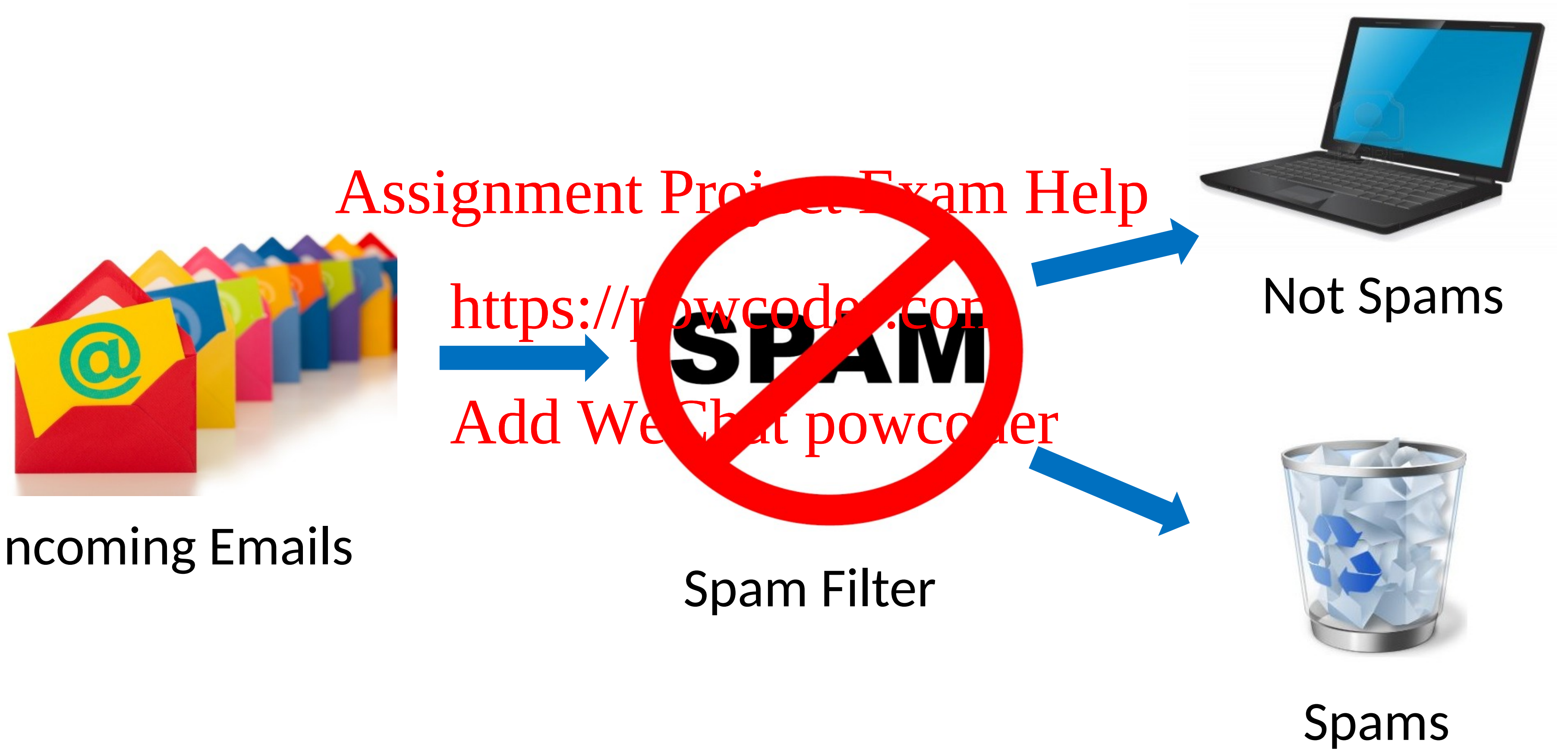
Lecture 11: Online Learning

Prof. Michael R. Lyu

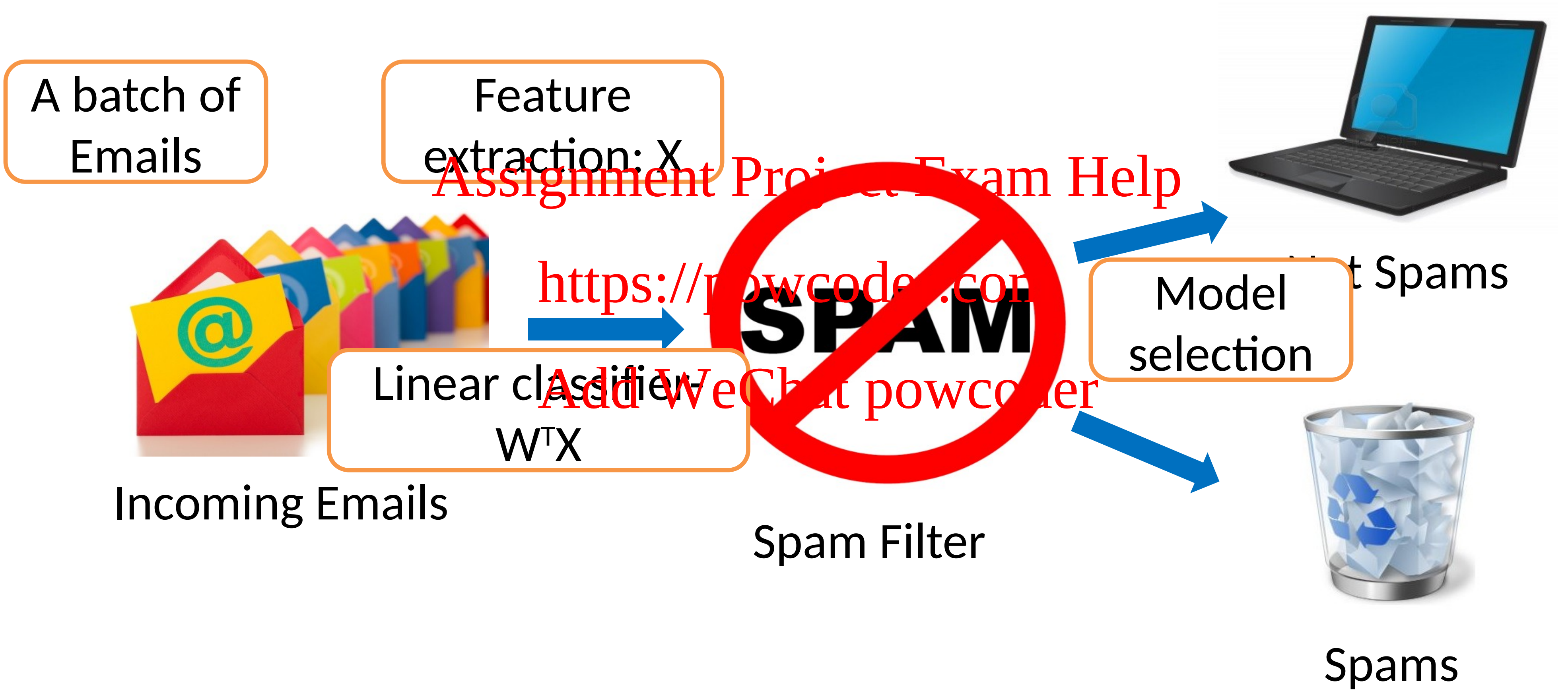
Computer Science & Engineering Dept.

The Chinese University of Hong Kong

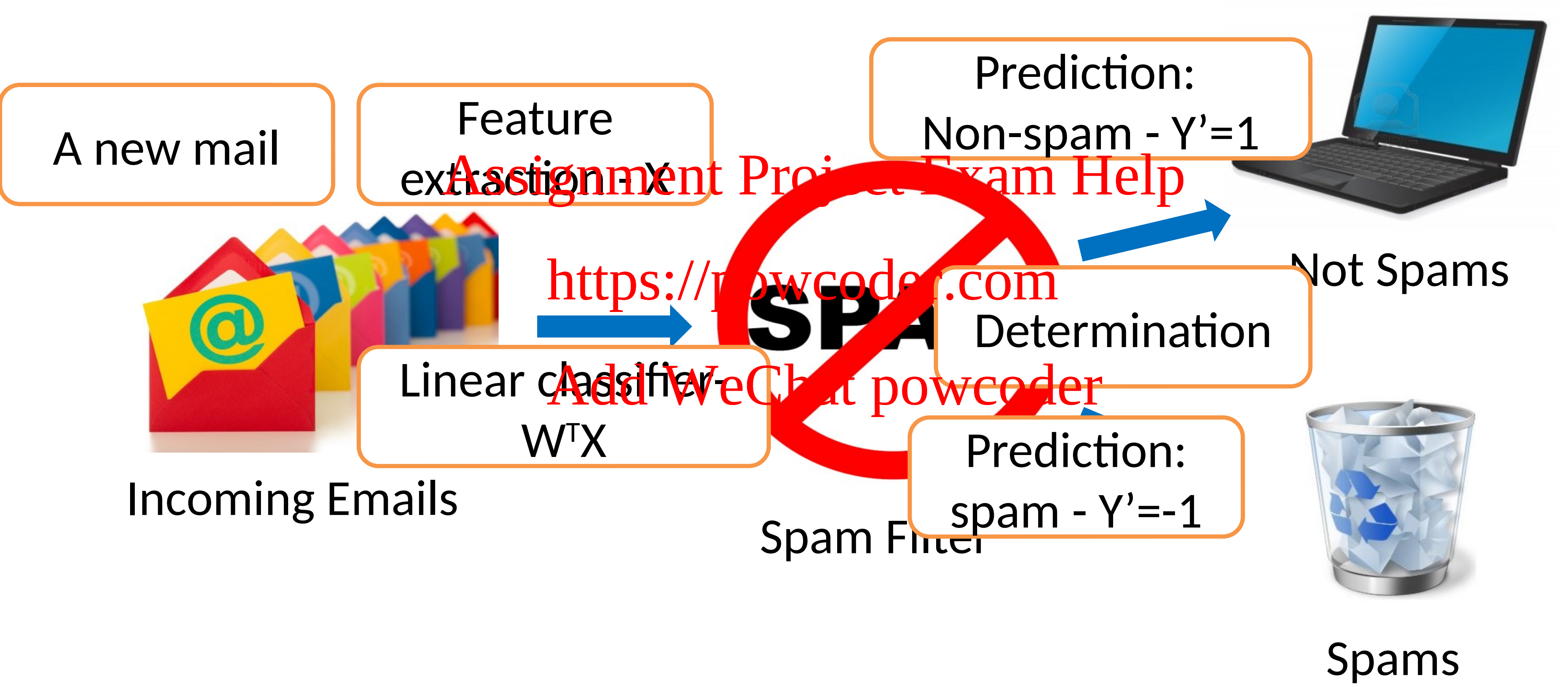
A Motivating Example– Spam Filtering



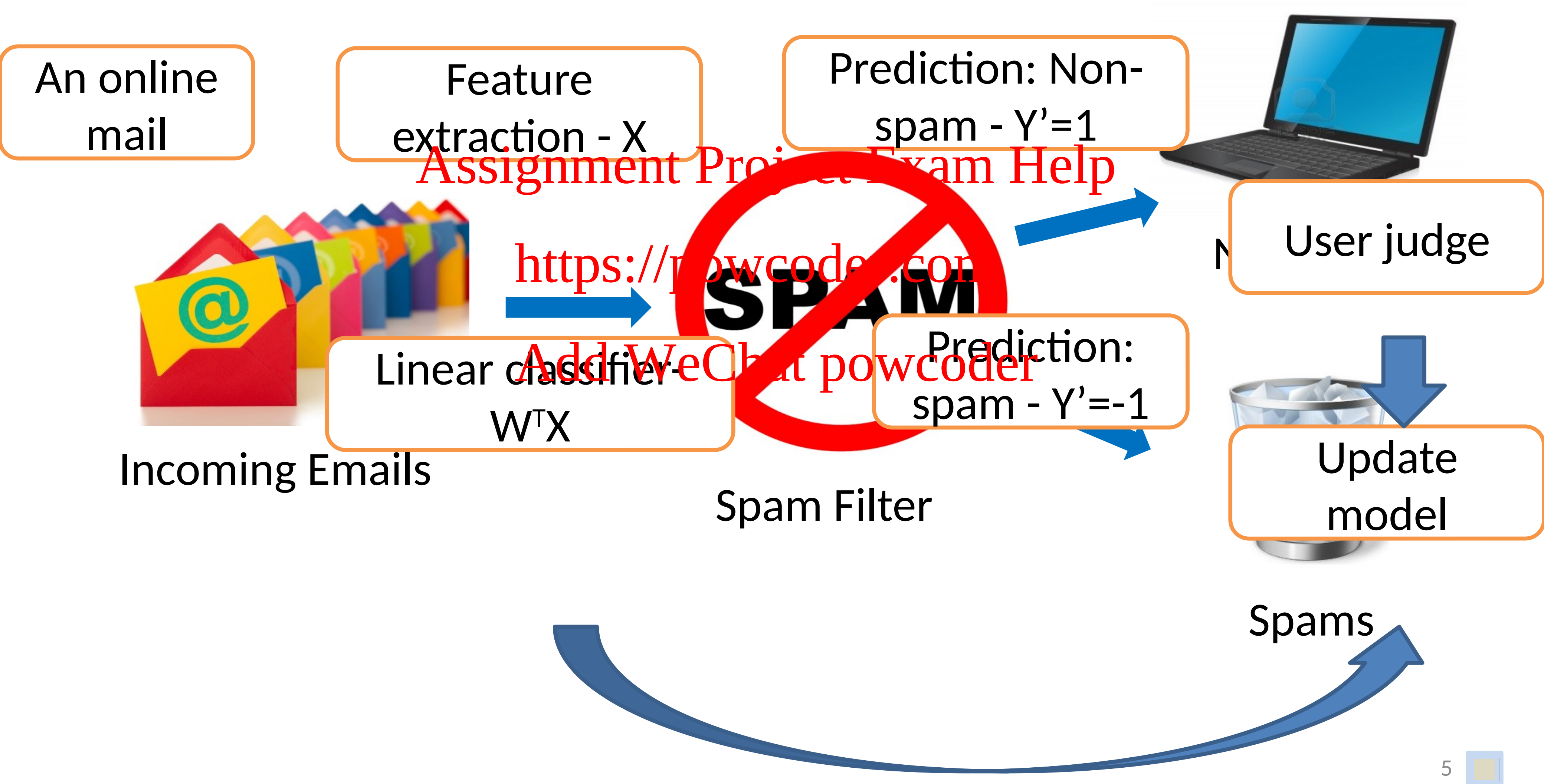
Traditional Method: Training



Traditional Method: Test



Online Protocol



Outline

- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

Outline

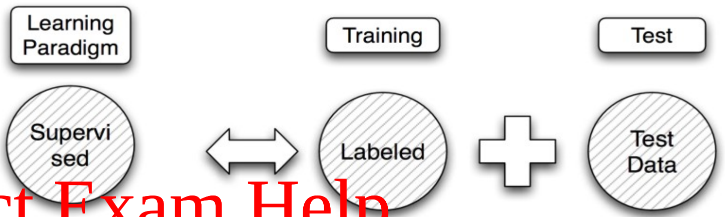
- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

Outline

- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

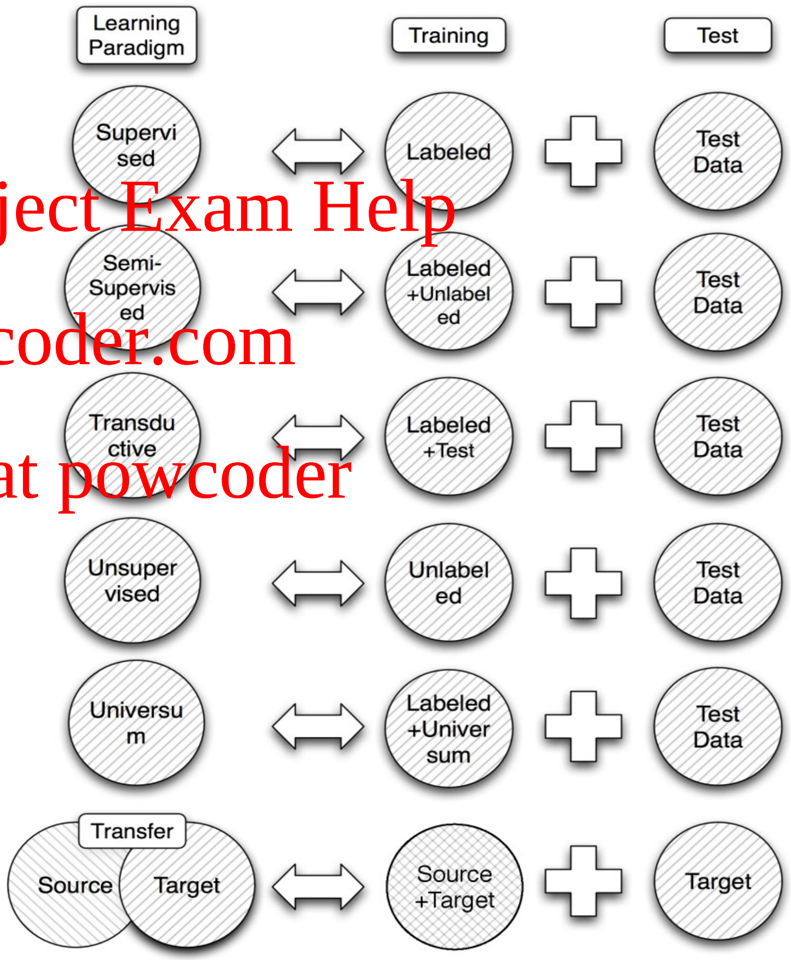
Learning Paradigms Overview

- Learning paradigms
  - **Supervised learning**



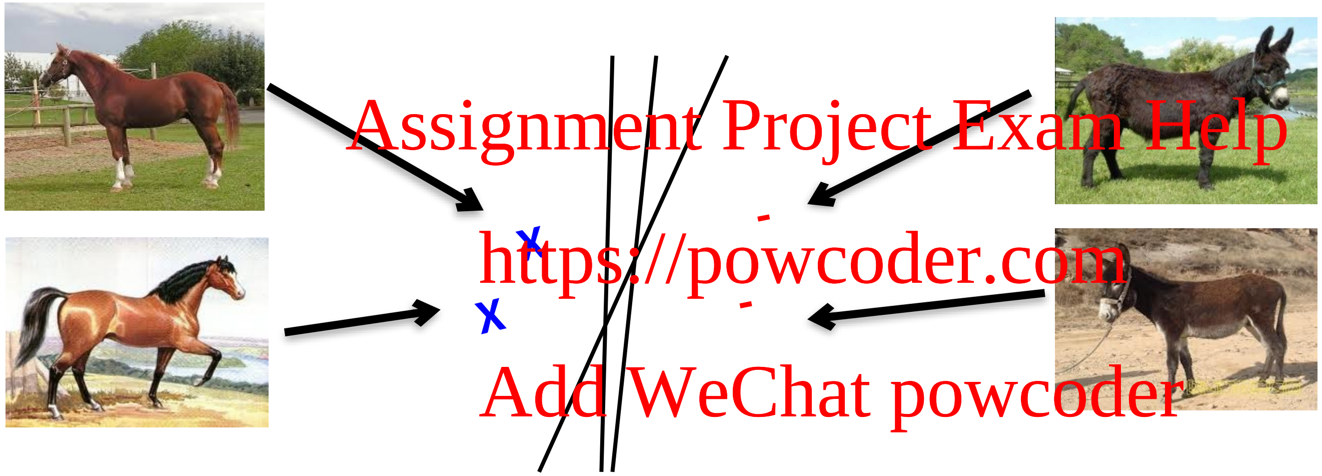
Learning Paradigms Overview

- Learning paradigms
  - **Supervised learning**
  - Semisupervised learning
  - Transductive learning
  - **Unsupervised learning**
  - Universum learning
  - Transfer learning



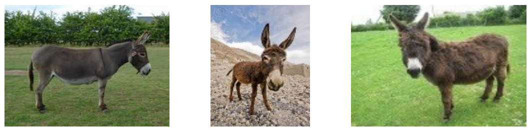
Supervised Learning

- Train on labeled data



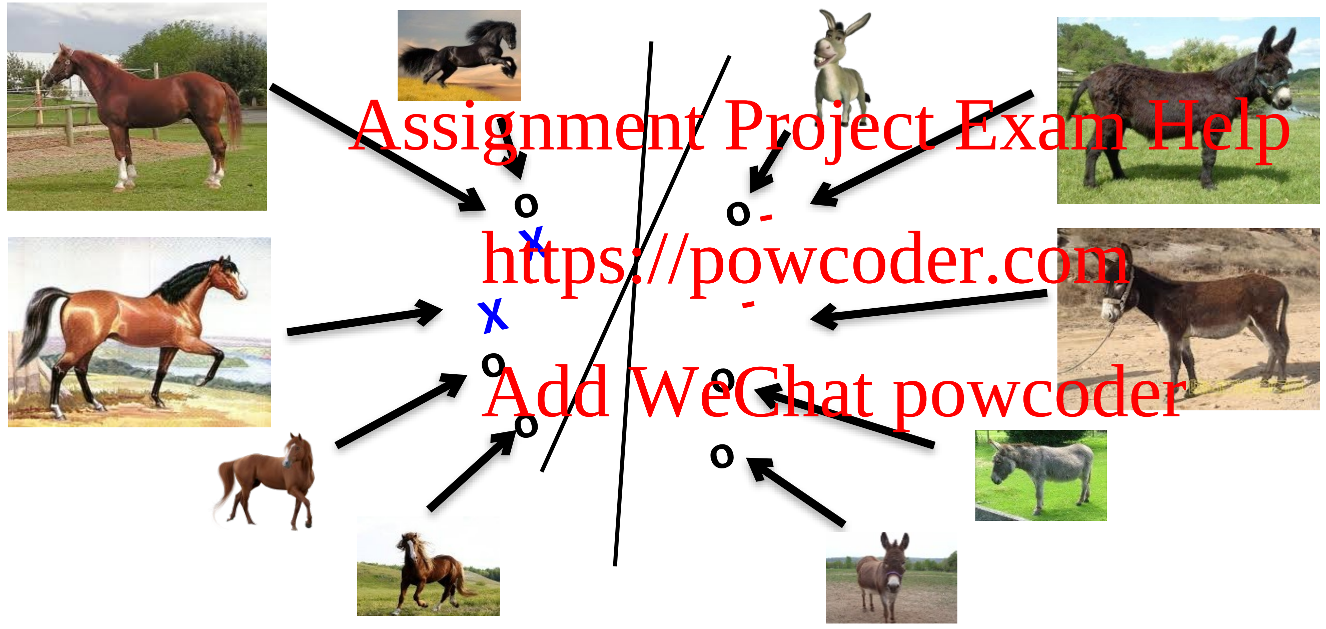
- Test on test data

?


?

Semisupervised Learning

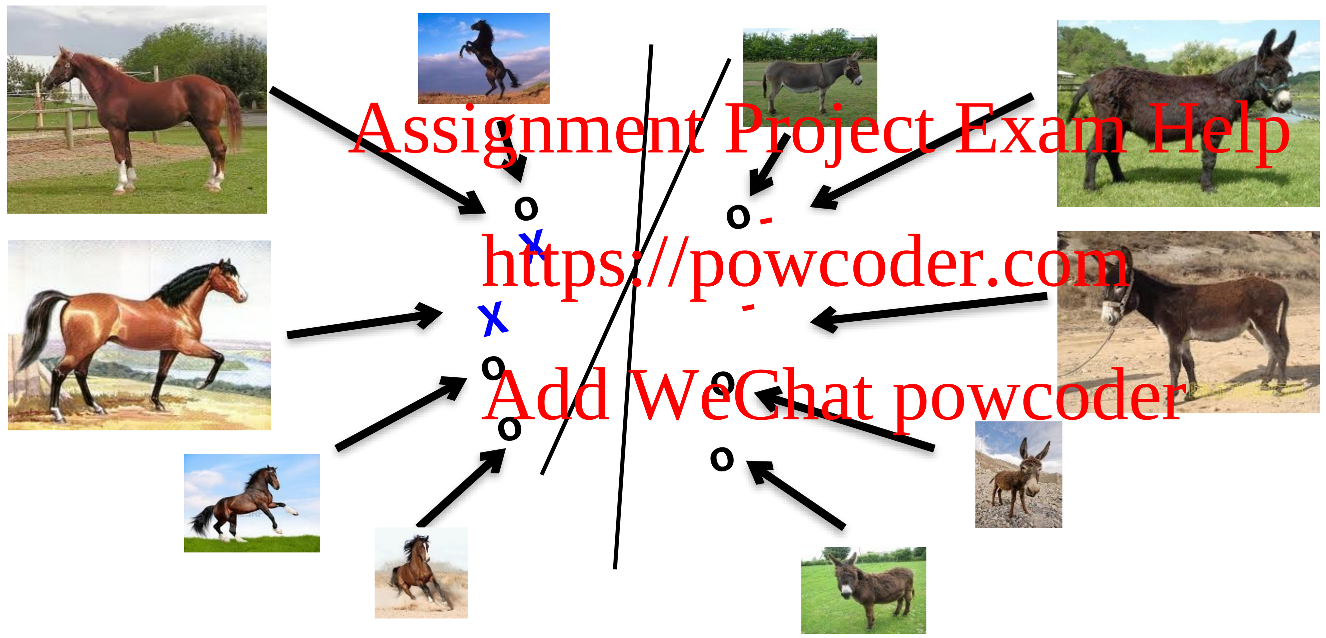
- Train on labeled and unlabeled data



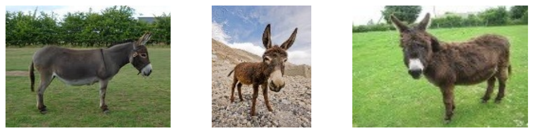
- Test on test data

Transductive Learning

- Train on labeled and test data

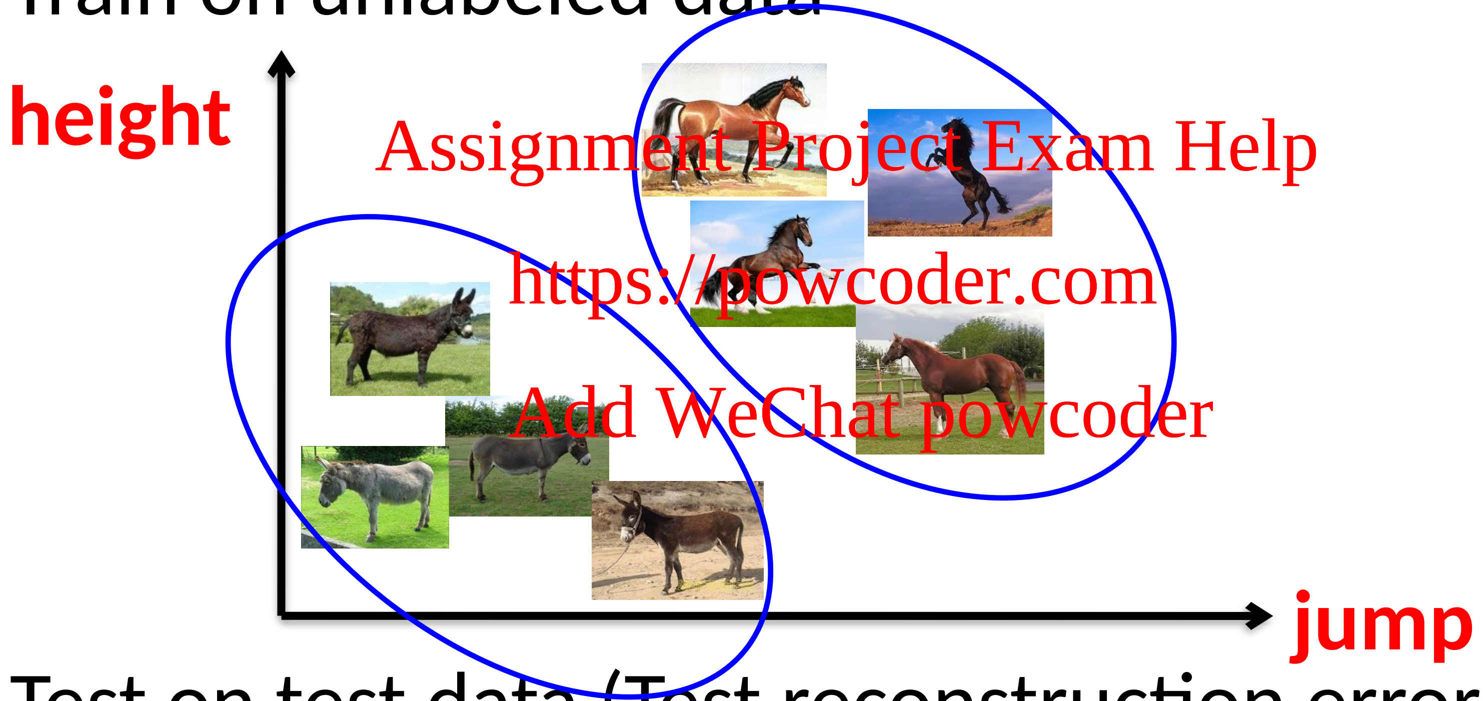


- Test on test data



Unsupervised Learning

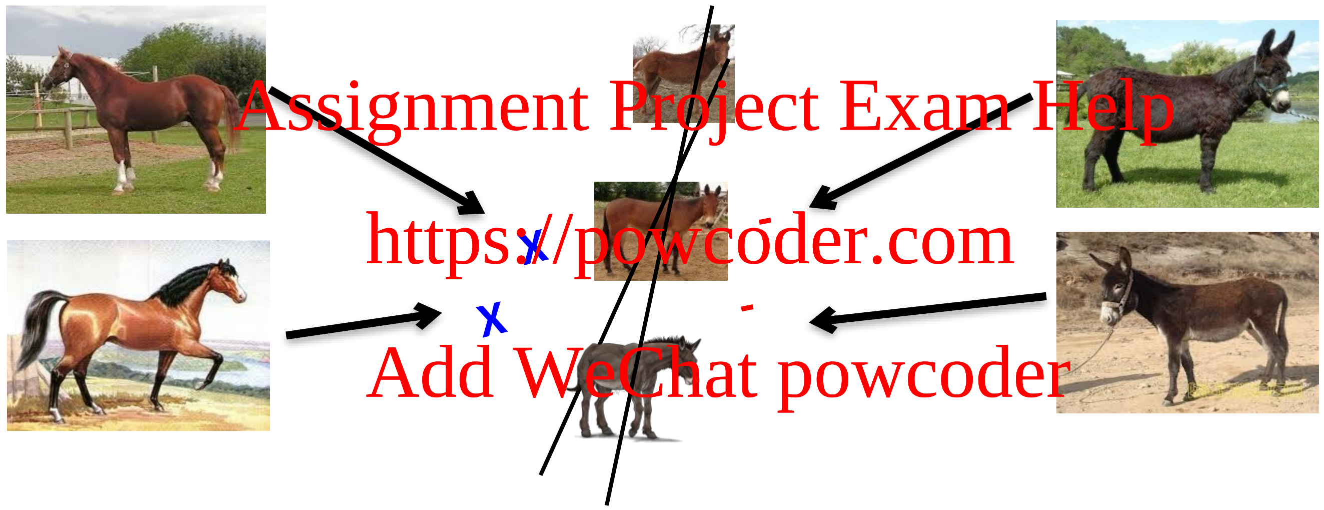
- Train on unlabeled data



- Test on test data (Test reconstruction error)

Universum Learning

- Train on labeled and universum data



- Test on test data

Transfer Learning

- Train on labeled from source and target domains

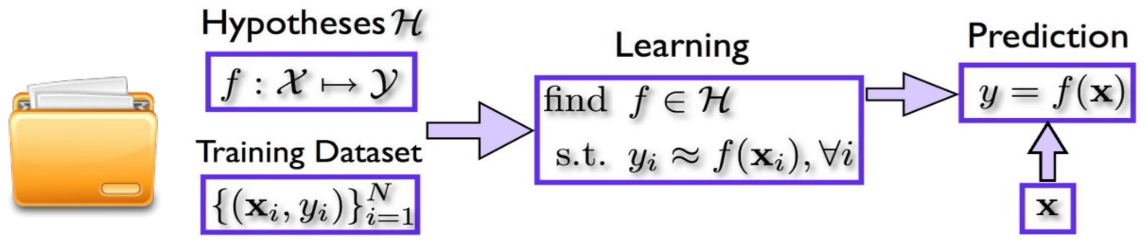
- Test on test data

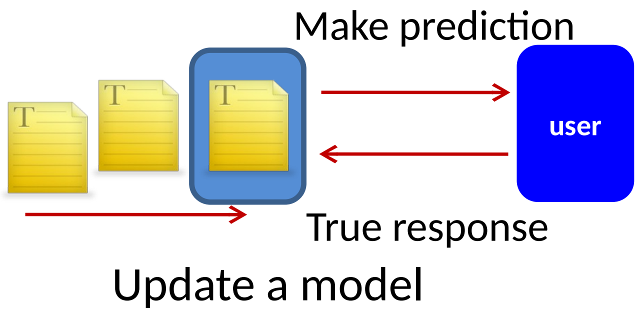
Outline

- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

What is Online Learning?

- Batch/Offline learning
  - Observe a **batch** of training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$
  - Learn a model from them
  - Predict new samples accurately
- Online learning
  - Observe a **sequence** of data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_t, y_t)$
  - Learn a model **incrementally** as instances come
  - Make the sequence of online predictions accurately

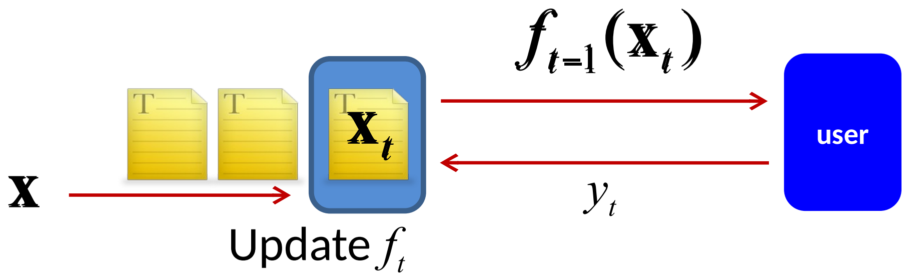

Hypotheses $\mathcal{H}$
$f: \mathcal{X} \mapsto \mathcal{Y}$
Training Dataset
$\{(\mathbf{x}_i, y_i)\}_{i=1}^N$
Learning
find $f \in \mathcal{H}$
s.t. $y_i \approx f(\mathbf{x}_i), \forall i$
Prediction
$y = f(\mathbf{x})$
$\mathbf{x}$



Online learning is the process of answering a sequence of questions given (maybe partial) knowledge of the correct answers to previous questions and possibly additional available information. [Shal11]

Online Prediction Algorithm

- An initial prediction rule $f_0(\cdot)$
- For $t = 1, 2, \ldots$
  - We observe $\mathbf{x}_t$ and make a prediction $f_{t-1}(\mathbf{x}_t)$
  - We observe the true outcome y_t and then compute a loss $l(f_{t-1}(\mathbf{x}_t), y_t)$
  - The online algorithm updates the prediction rule using the new example and construct $f_t(\mathbf{x})$

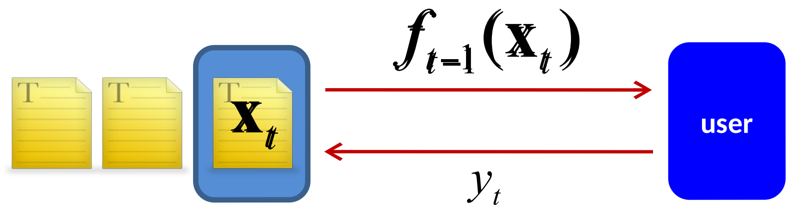


Online Prediction Algorithm

- The total error of the method is

$$\sum_{t=1}^{T} l(f_{t-1}(\mathbf{x}_t), y_t)$$

- Goal: this error to be as small as possible
- Predict unknown future one step a time: similar to generalization error



Regret Analysis

- $f_*(\cdot)$: optimal prediction function from a class H, e.g., the class of linear classifiers

$$f_*(\cdot) = \arg\min_{f \in H} \sum_{t=1}^{T} l(f(\mathbf{x}_t), y_t)$$

with minimum error after seeing all examples

- Regret for the online learning algorithm

$$\text{regret} = \frac{1}{T}\sum_{t=1}^{T}\left[l(f_{t-1}(\mathbf{x}_t), y_t) - l(f_*(\mathbf{x}_t), y_t)\right]$$

We want regret as small as possible

Why Low Regret?

- Regret for the online learning algorithm

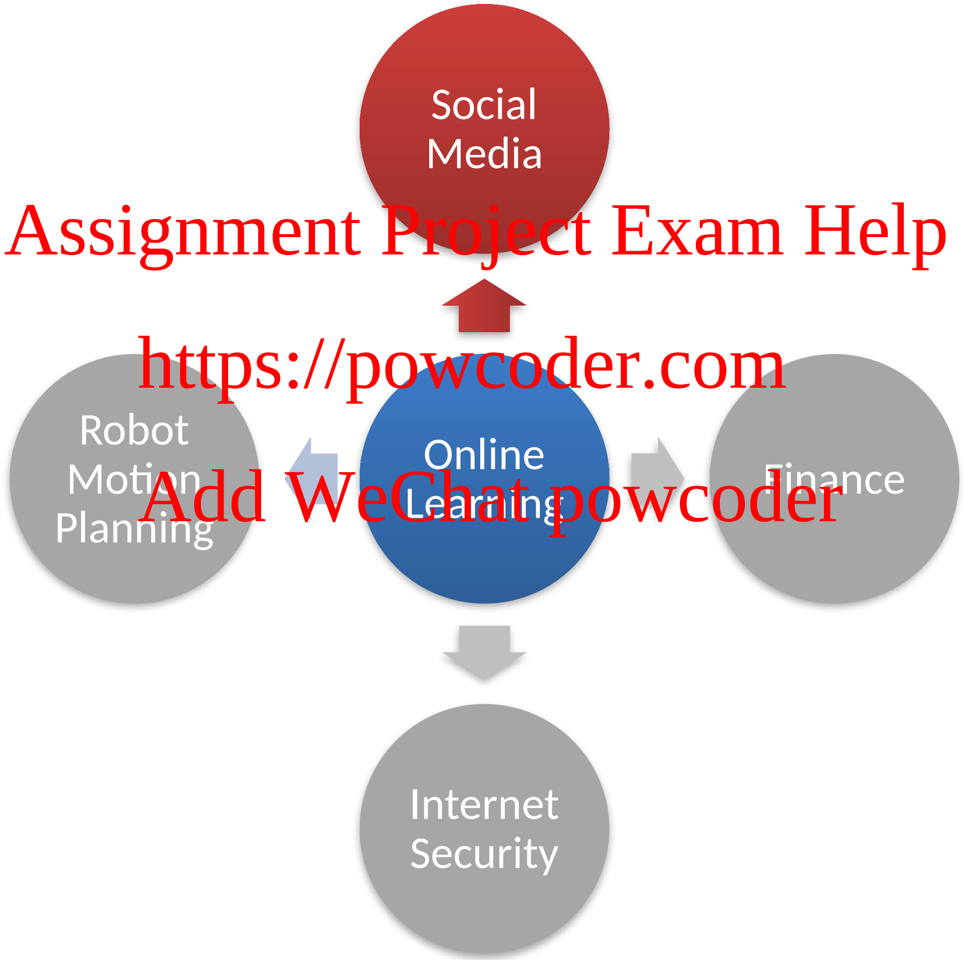
$$\text{regret} = \frac{1}{T}\sum_{t=1}^{T}\left[l(f_{t-1}(\mathbf{x}_t), y_t) - l(f_*(\mathbf{x}_t), y_t)\right]$$

- Advantages
  - We do not lose much from not knowing future events
  - We can perform almost as well as someone who observes the entire sequence and picks the best prediction strategy in hindsight
  - We can also compete with changing environment

Advantages of Online Learning

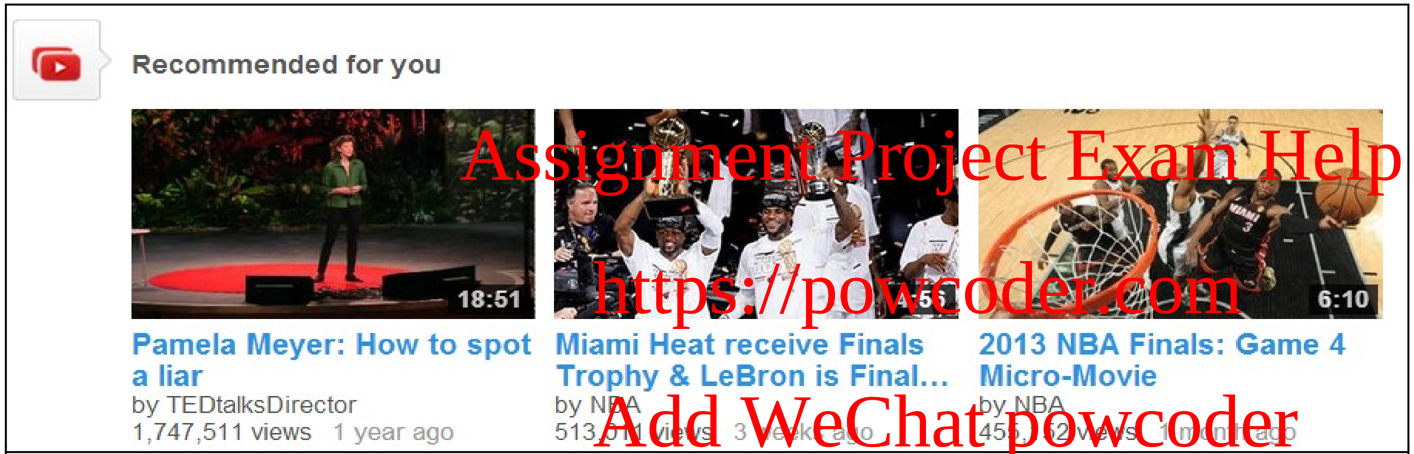
- Meet many applications for data arriving sequentially while predictions are required on-the-fly
  - Avoid re-training when adding new data
- Applicable in adversarial and competitive environment
- Strong adaptability to changing environment
- High efficiency and excellent scalability
- Simple to understand and easy to implement
- Easy to be parallelized
- Theoretical guarantees

Where to Apply Online Learning?



Online Learning for Social Media

- Recommendation, sentiment/emotion analysis

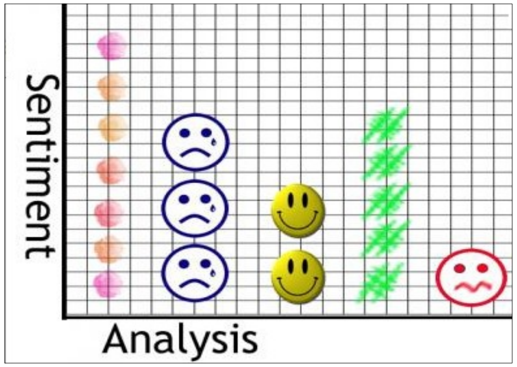


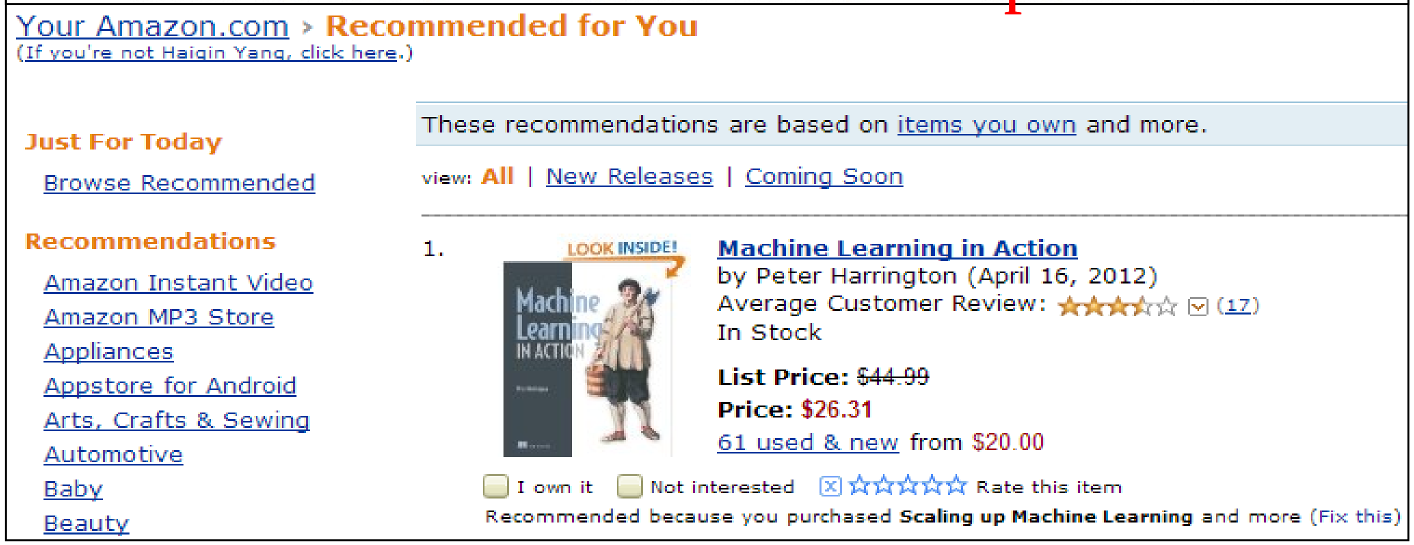
Recommended for you

18:51 Pamela Meyer: How to spot a liar
by TEDtalksDirector
1,747,511 views 1 year ago

Miami Heat receive Finals Trophy & LeBron is Final...
by NBA
513,511 views 3 weeks ago

6:10 2013 NBA Finals: Game 4 Micro-Movie
by NBA
455,562 views 1 month ago





Your Amazon.com › Recommended for You
(If you're not Haiqin Yang, click here.)

Just For Today
Browse Recommended

Recommendations
Amazon Instant Video
Amazon MP3 Store
Appliances
Appstore for Android
Arts, Crafts & Sewing
Automotive
Baby
Beauty

These recommendations are based on items you own and more.

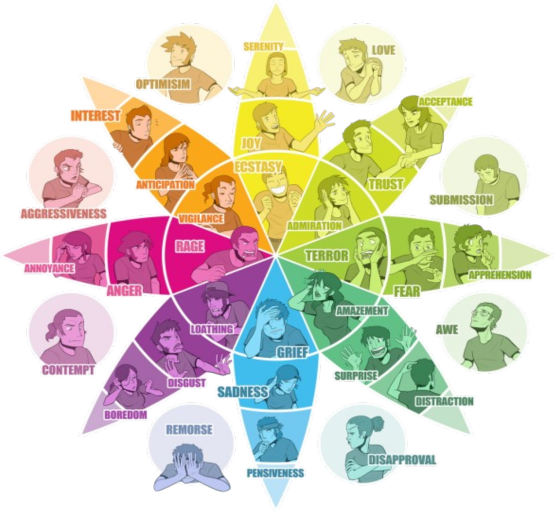
view: All | New Releases | Coming Soon

1. Machine Learning in Action
by Peter Harrington (April 16, 2012)
Average Customer Review: (17)
In Stock

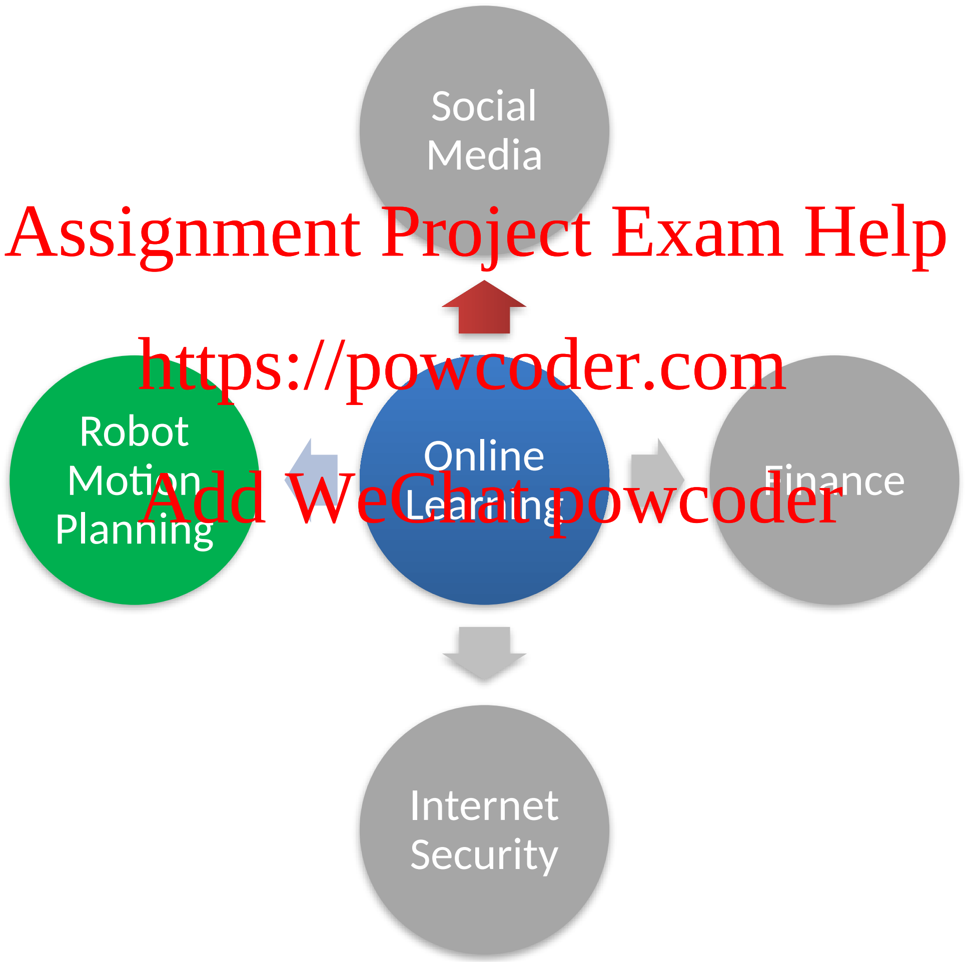
List Price: ~~$44.99~~
Price: $26.31
61 used & new from $20.00

I own it | Not interested | Rate this item

Recommended because you purchased **Scaling up Machine Learning** and more (Fix this)



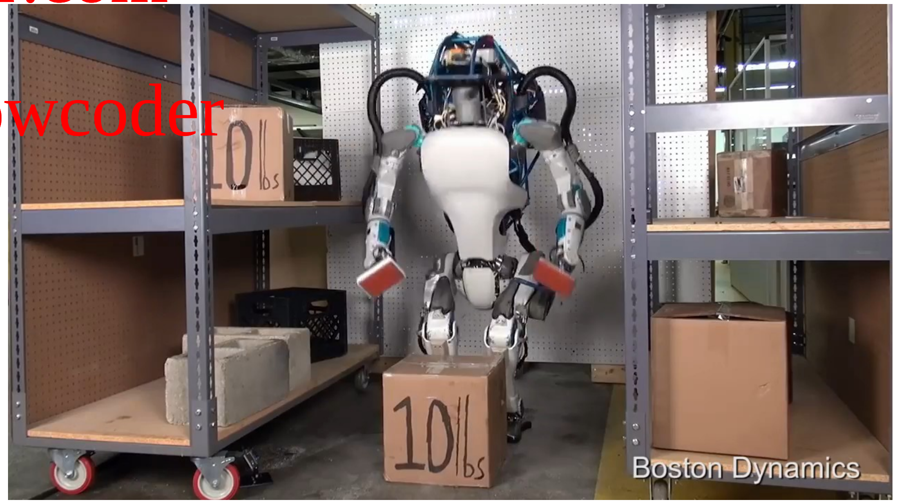
Where to Apply Online Learning?



Online Learning for Robot Motion Planning

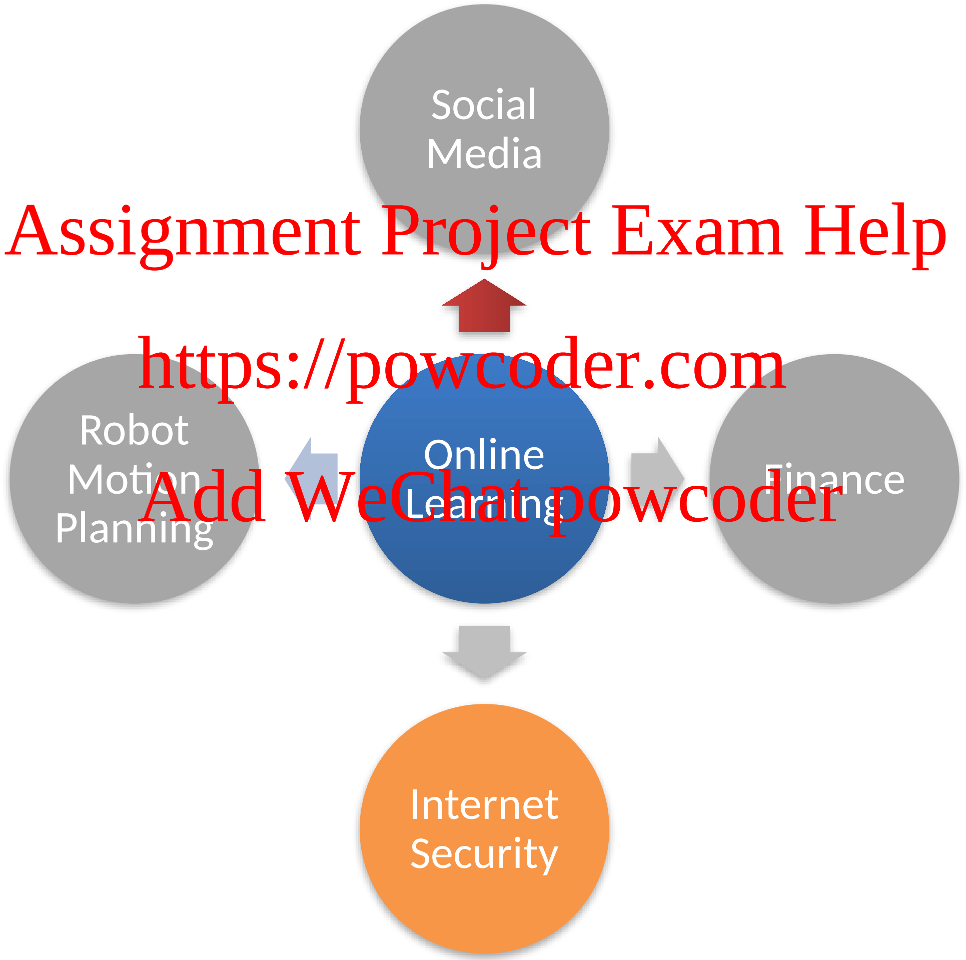
- Tasks
  - Exploring an unknown terrain
  - Finding a destination

Rock-Paper-Scissors: You vs. the Computer



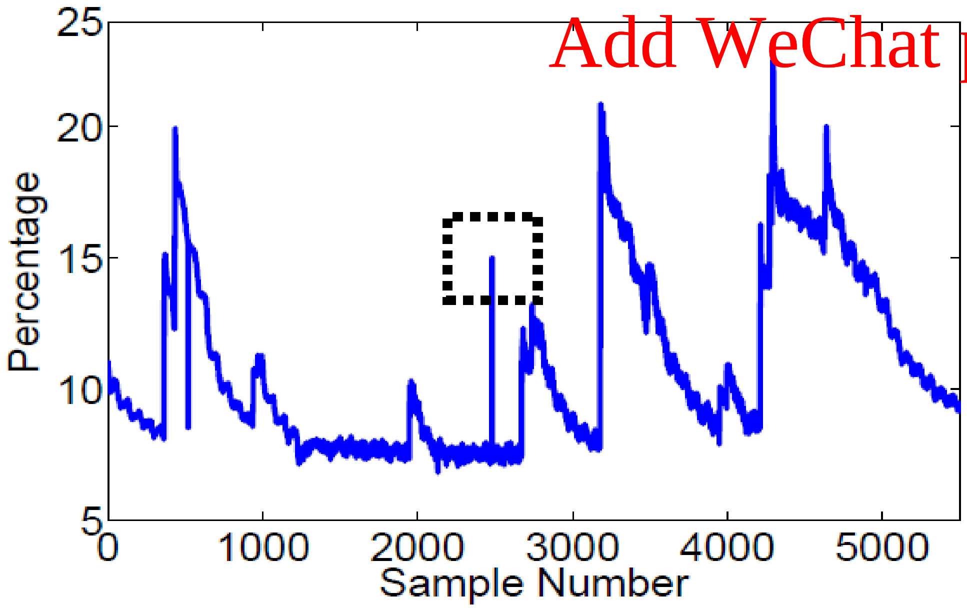
Robot Dog

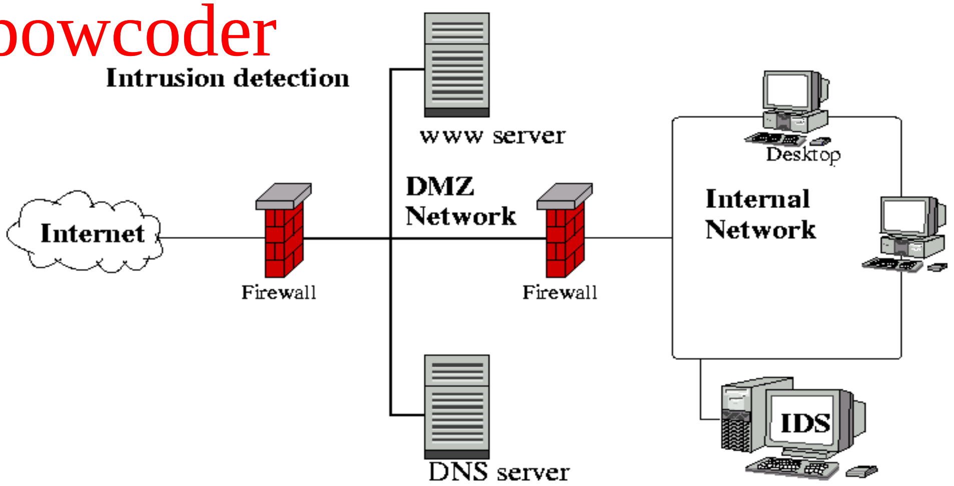
Where to Apply Online Learning?



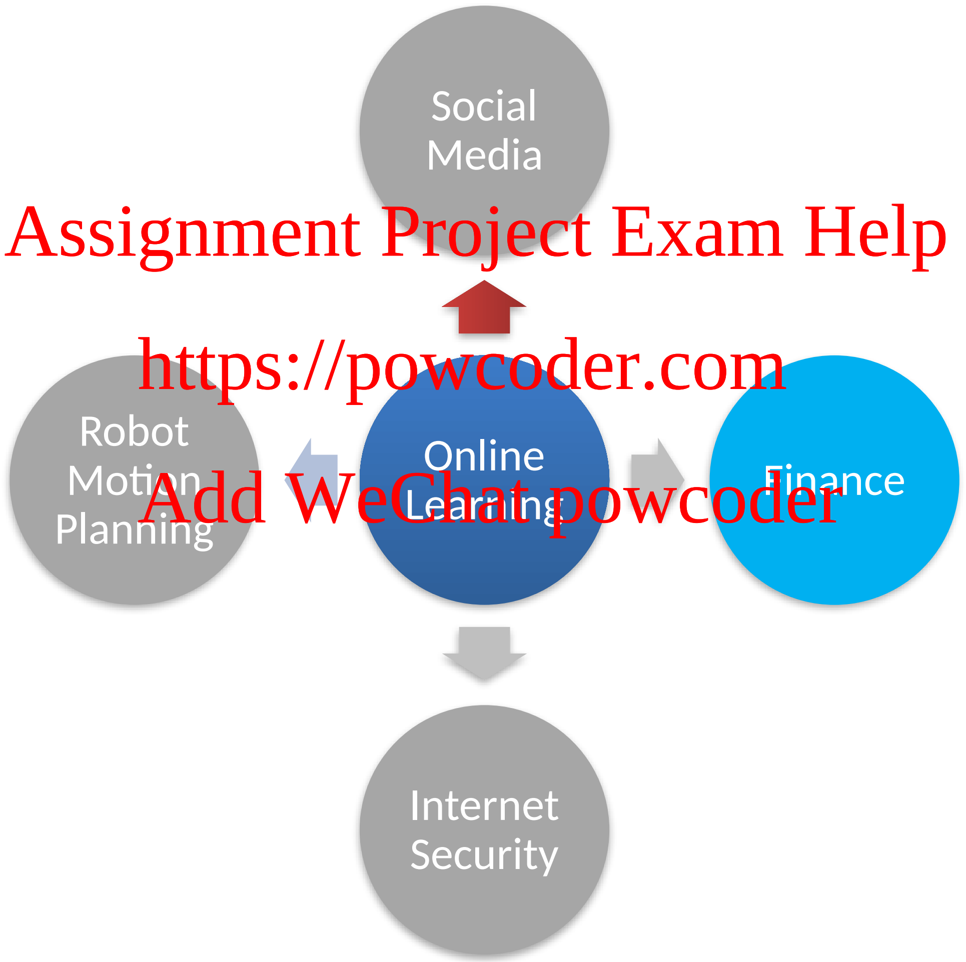
Online Learning for Internet Security

- Electronic business sectors
  - **Spam email** filtering
  - **Fraud** credit card transaction detection
  - Network **intrusion detection** system, etc.



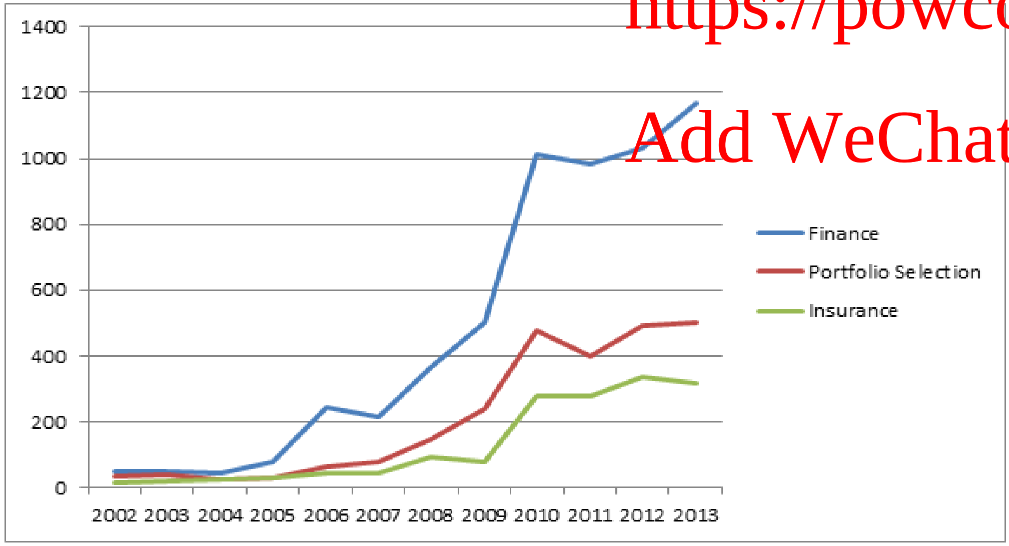


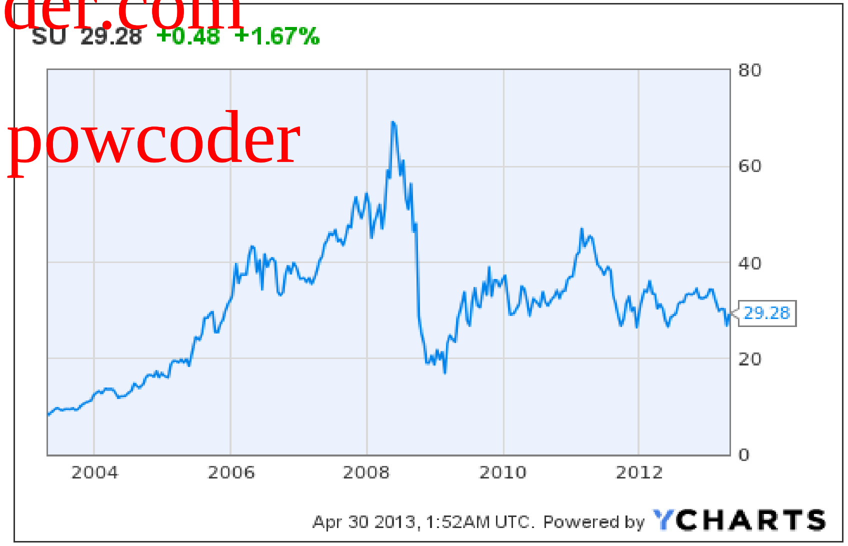
Where to Apply Online Learning?



Online Learning for Financial Decision

- Financial decision
  - Online portfolio selection
  - Sequential investment, etc.





Outline

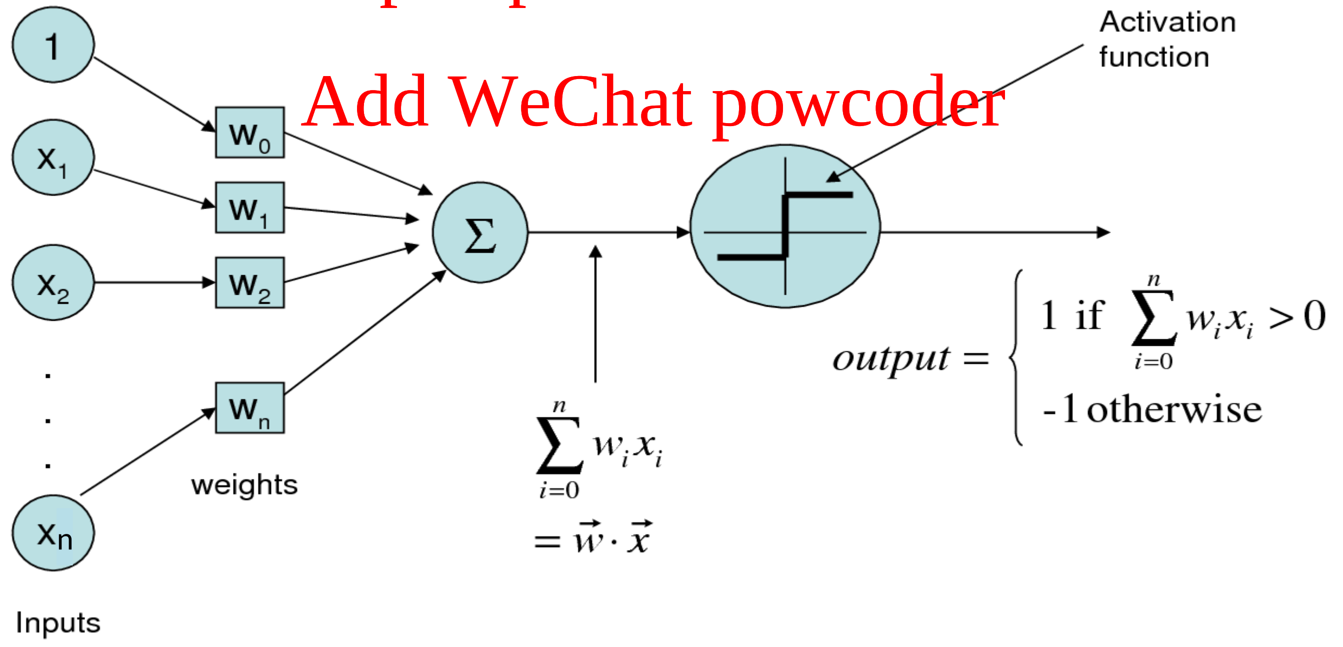
- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

Outline

- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

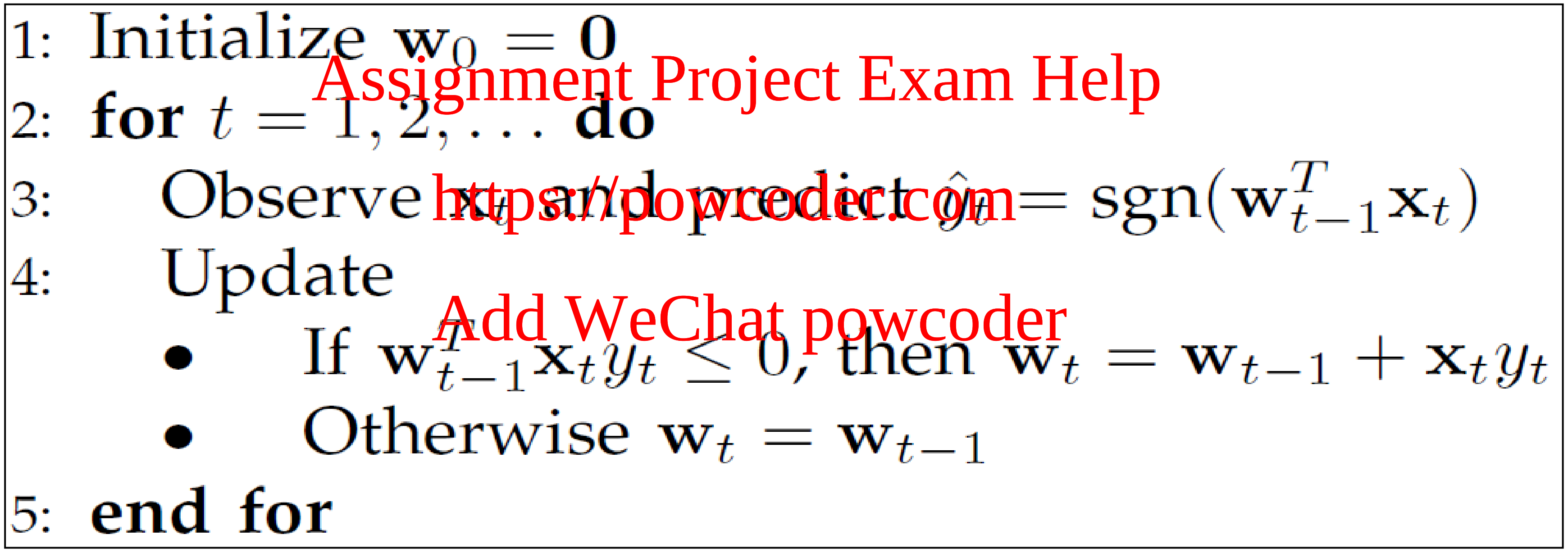
Perceptron Algorithm (F. Rosenblatt, 1958)

- One of the oldest machine learning algorithm
- Online algorithm for learning a linear threshold function with small error


1
x_1
x_2
x_n
Inputs
w_0
w_1
w_2
w_n
weights
Σ
Activation function
$\sum_{i=0}^{n} w_i x_i = \vec{w} \cdot \vec{x}$
$output = \begin{cases} 1 \text{ if } \sum_{i=0}^{n} w_i x_i > 0 \\ -1 \text{ otherwise} \end{cases}$

Perceptron Algorithm (F. Rosenblatt, 1958)

- Goal: find a linear classifier with small error



```
1: Initialize $\mathbf{w}_0 = \mathbf{0}$
2: for $t = 1, 2, \ldots$ do
3:     Observe $\mathbf{x}_t$ and predict $\hat{y}_t = \mathrm{sgn}(\mathbf{w}_{t-1}^T \mathbf{x}_t)$
4:     Update
       •   If $\mathbf{w}_{t-1}^T \mathbf{x}_t y_t \le 0$, then $\mathbf{w}_t = \mathbf{w}_{t-1} + \mathbf{x}_t y_t$
       •   Otherwise $\mathbf{w}_t = \mathbf{w}_{t-1}$
5: end for
```

If no error, keeping the same;
otherwise, update.

Intuition Explanation

- Want positive margin:

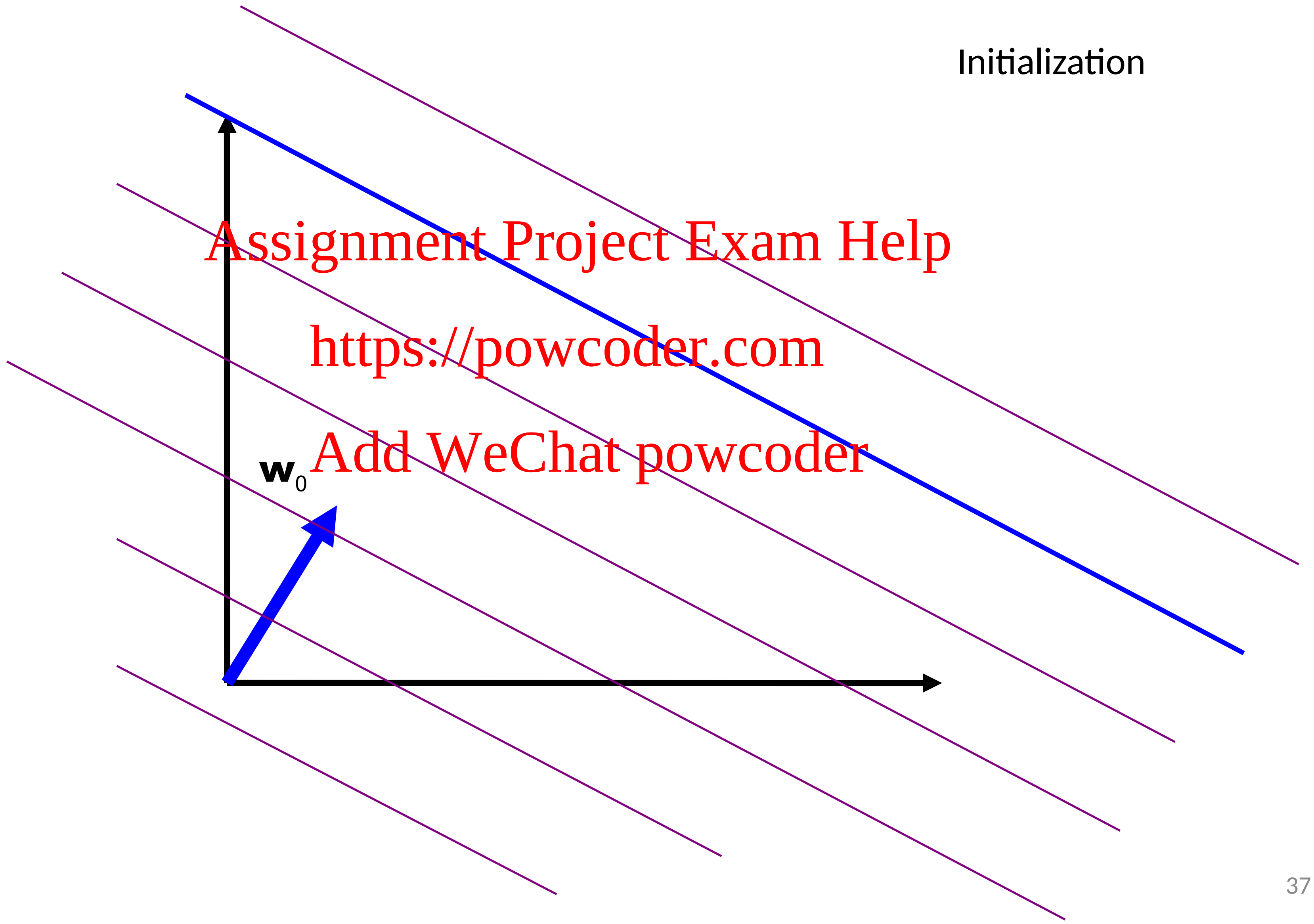
$$\hat{y}_t \neq y_t \quad \text{iff} \quad \underbrace{y_t \mathbf{w}_{t-1}^T \mathbf{x}_t}_{\text{margin}} \leq 0$$

- Effect of Perceptron update on margin:

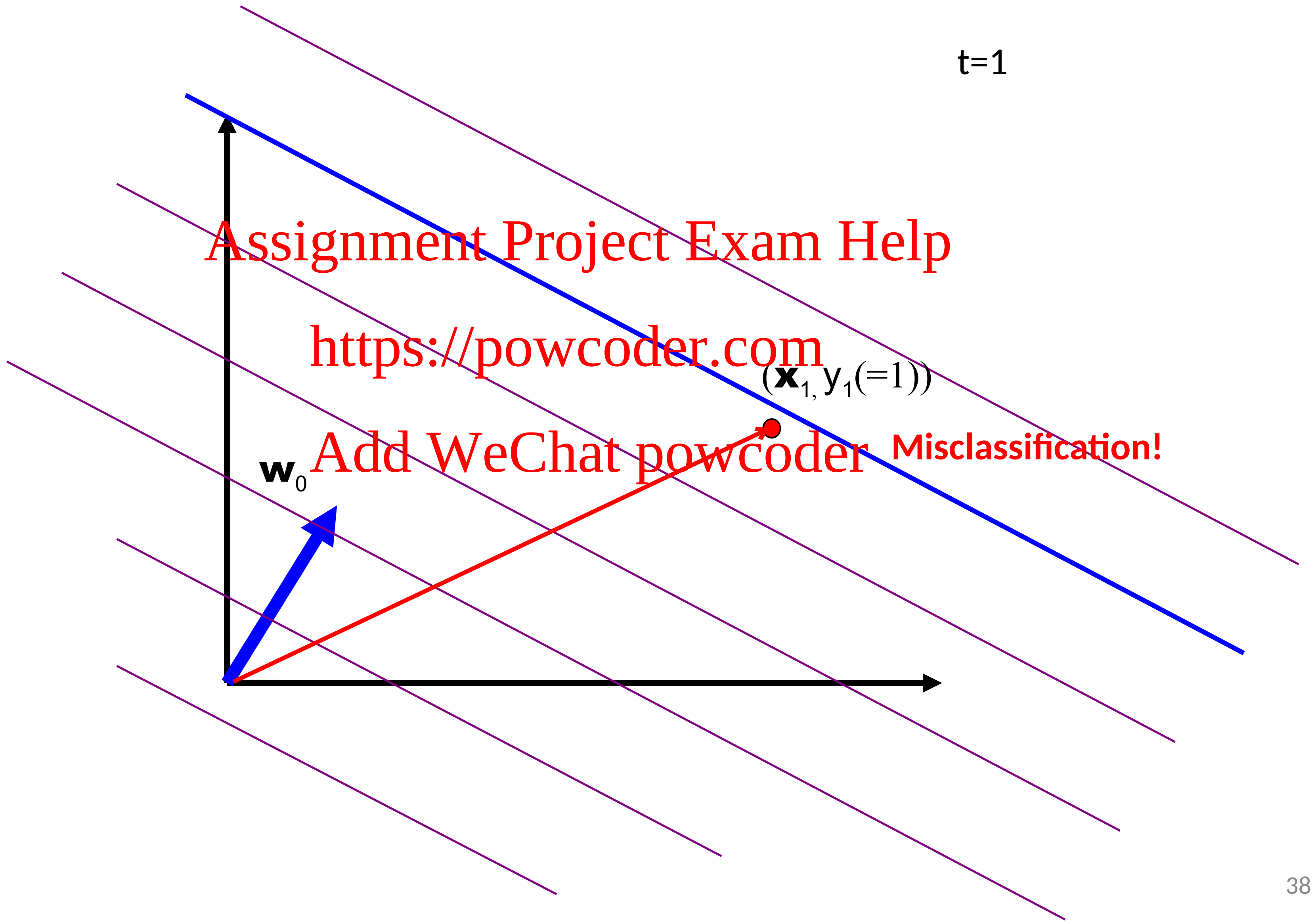
$$y_t \mathbf{w}_t^T \mathbf{x}_t = y_t \left(\mathbf{w}_{t-1} + y_t \mathbf{x}_t\right)^T \mathbf{x}_t = y_t \mathbf{w}_{t-1}^T \mathbf{x}_t + \|\mathbf{x}_t\|^2$$

- So margin increases

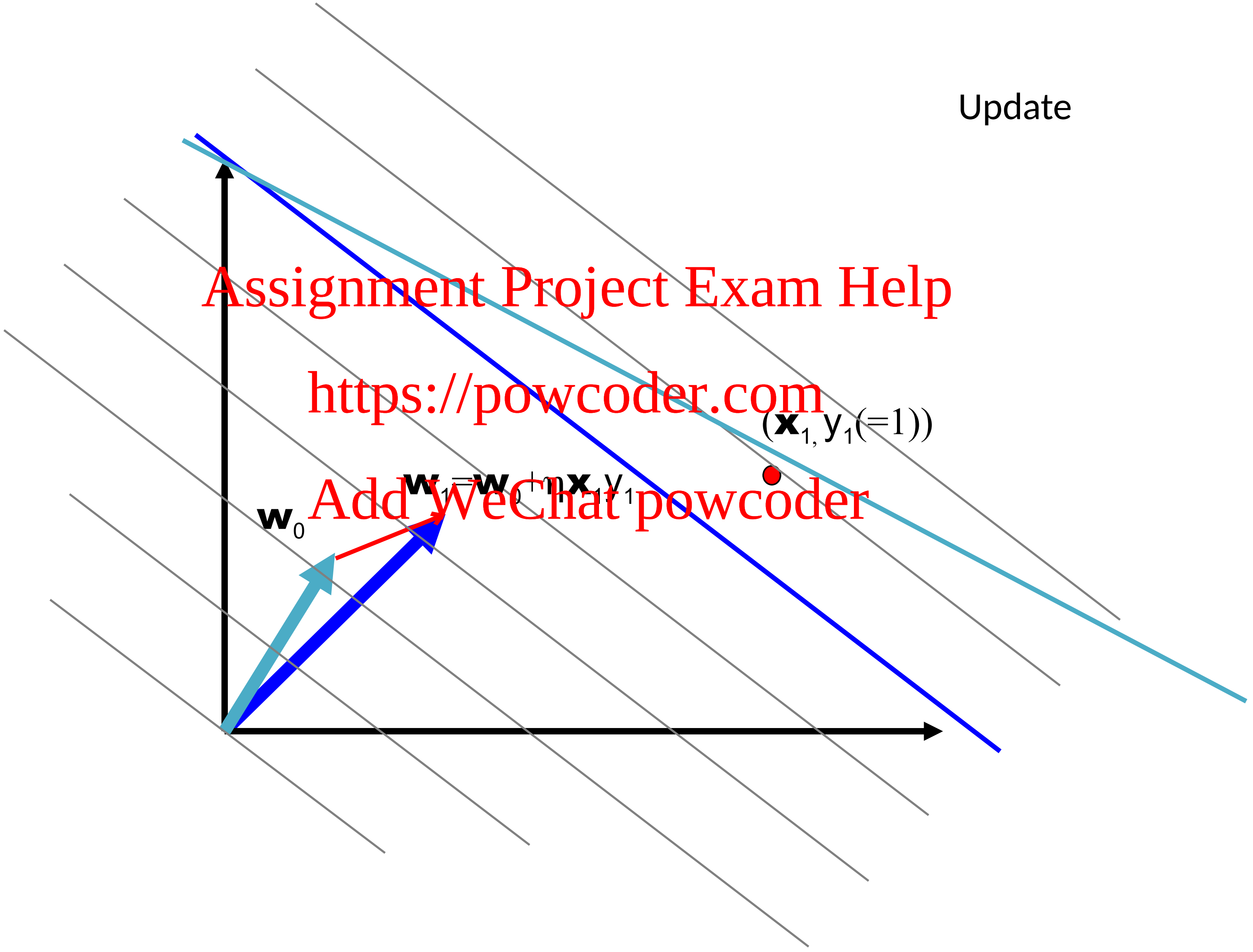
Geometric View



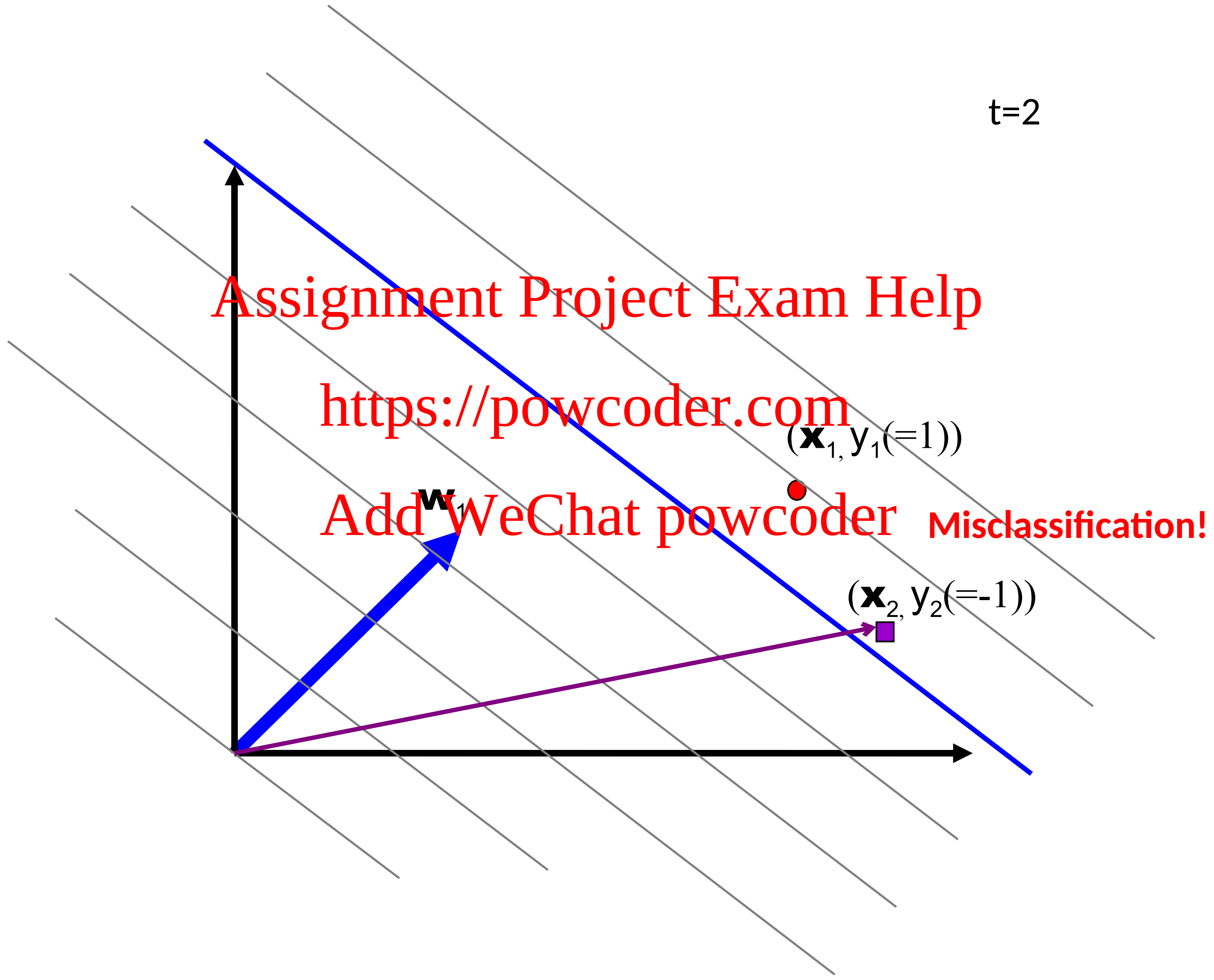
Geometric View



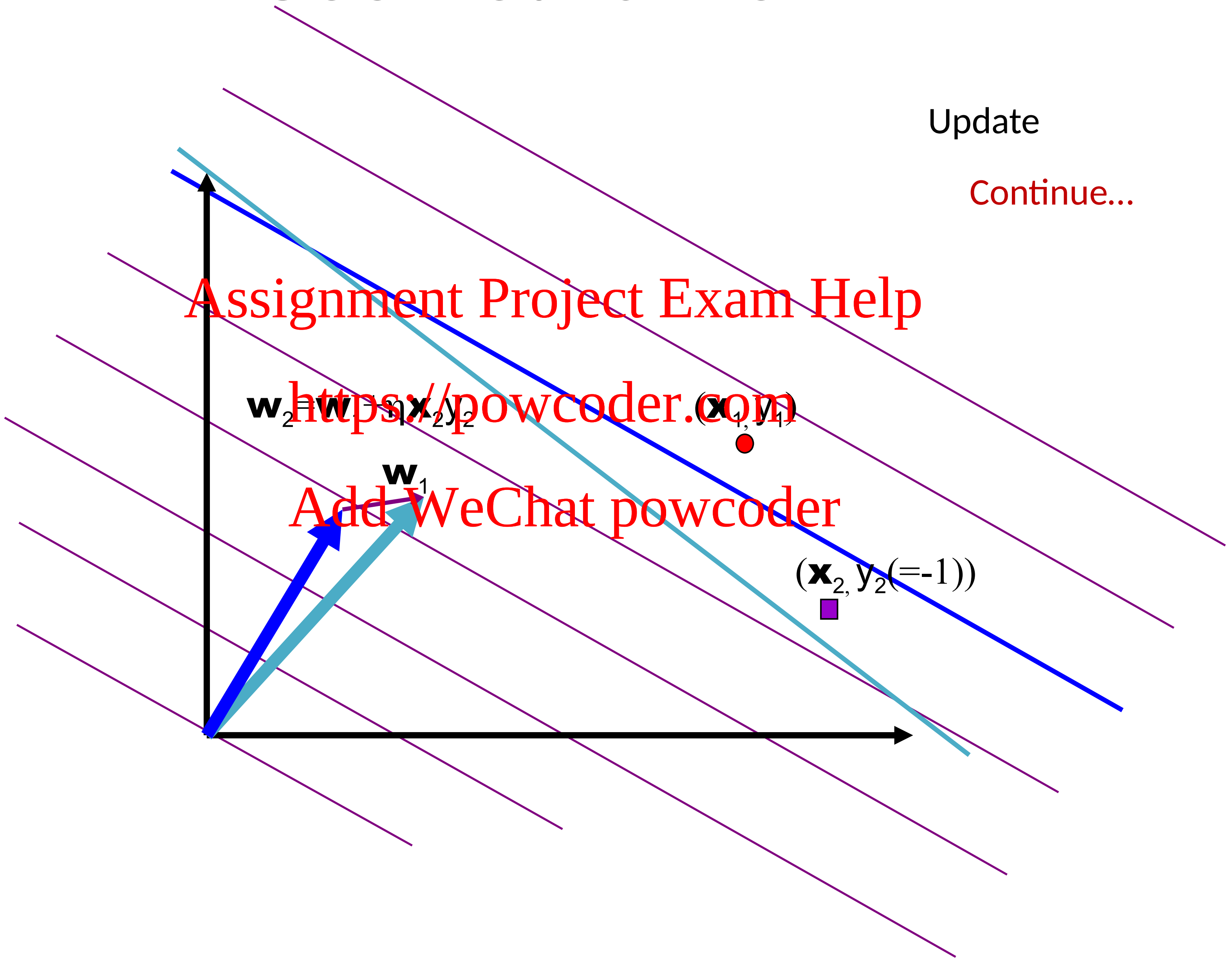
Geometric View


Update
$(\mathbf{x}_{1,} y_1(=1))$
$\mathbf{w}_1 = \mathbf{w}_0 + \eta \mathbf{x}_1 y_1$
$\mathbf{w}_0$

Geometric View



Geometric View


Update
Continue...
$\mathbf{w}_2 = \mathbf{w}_1 + \eta \mathbf{x}_2 y_2$
$(\mathbf{x}_1, y_1)$
$\mathbf{w}_1$
$(\mathbf{x}_2, y_2(=-1))$

In-class Practice

- Go to practice

Perceptron Mistake Bound

- Consider w_* separate the data: $\mathbf{w}_*^T \mathbf{x}_i y_i > 0$
- Define margin

$$\gamma = \frac{\min_i |\mathbf{w}_*^T \mathbf{x}_i|}{\|\mathbf{w}_*\|_2 \sup_i \|\mathbf{x}_i\|_2}$$

The larger, the more confidence

Norm of $\mathbf{x}$: the larger, the larger mistake bound

- The number of mistakes perceptron makes is at most γ^{-2}

Proof of Perceptron Mistake Bound [Novikoff, 1963]

Proof: Let $\mathbf{v}_k$ be the hypothesis before the k-th mistake. Assume that the k-th mistake occurs on the input example $(\mathbf{x}_i, y_i)$.

$$\gamma = \frac{\min_i |\mathbf{w}_*^T \mathbf{x}_i|}{\|\mathbf{w}_*\|_2 \sup_i \|\mathbf{x}_i\|_2}$$

First,

$$\begin{aligned} \|\mathbf{v}_{k+1}\|^2 &= \|\mathbf{v}_k + y_i \mathbf{x}_i\|^2 \\ &= \|\mathbf{v}_k\|^2 + 2y_i(\mathbf{v}_k^T \mathbf{x}_i) \\ &\quad + \|\mathbf{x}_i\|^2 \\ &\leq \|\mathbf{v}_k\|^2 + R^2 \\ &\leq kR^2 (R := \sup_i \|\mathbf{x}\|_2) \end{aligned}$$

Second,

$$\begin{aligned} \mathbf{v}_{k+1} &= \mathbf{v}_k + y_i \mathbf{x}_i \\ \mathbf{v}_{k+1}^T \mathbf{u} &= \mathbf{v}_k^T \mathbf{u} + y_i \mathbf{x}_i^T \mathbf{u} \\ &\geq \mathbf{v}_k^T \mathbf{u} + \gamma R \\ \mathbf{v}_{k+1}^T \mathbf{u} &\geq k\gamma R. \end{aligned}$$

Hence,

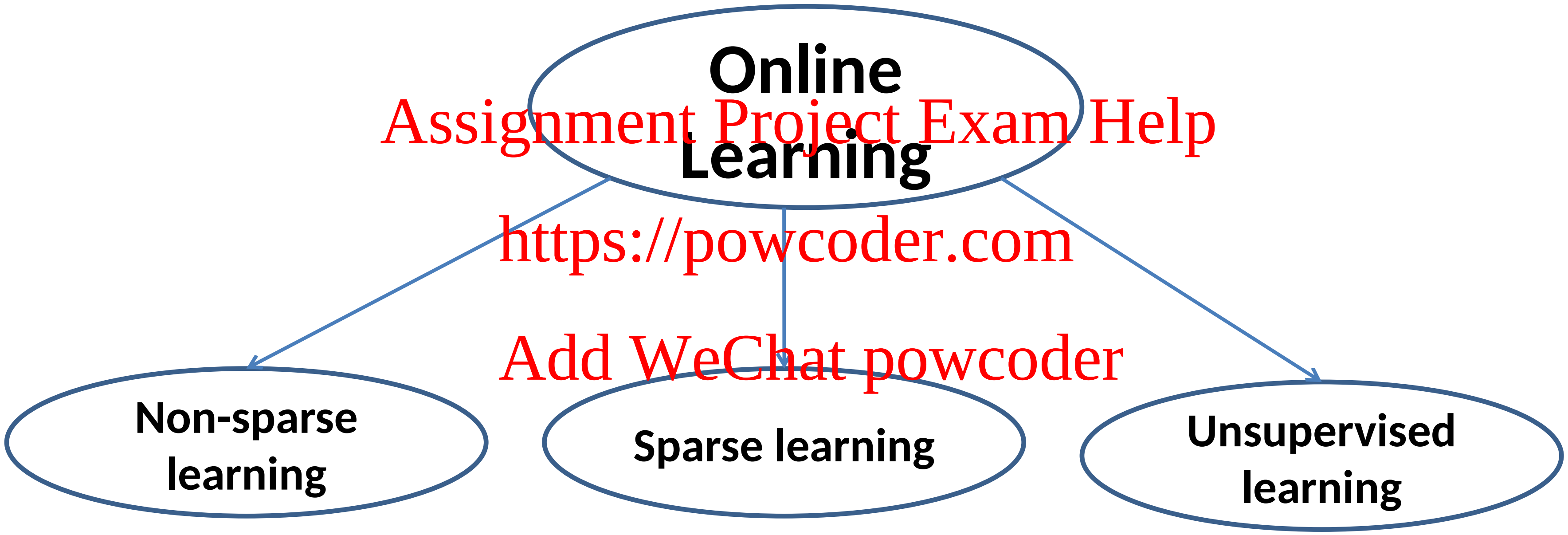
$$\sqrt{k} R \geq \|\mathbf{v}_{k+1}\| \geq \mathbf{v}_{k+1}^T \mathbf{u} \geq k\gamma R$$

$$k \leq \gamma^{-2}$$

Outline

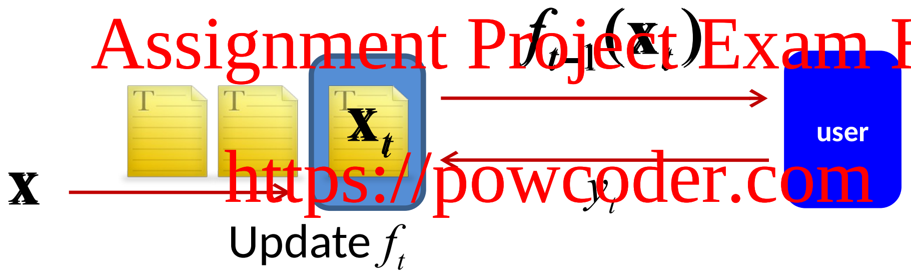
- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

Overview

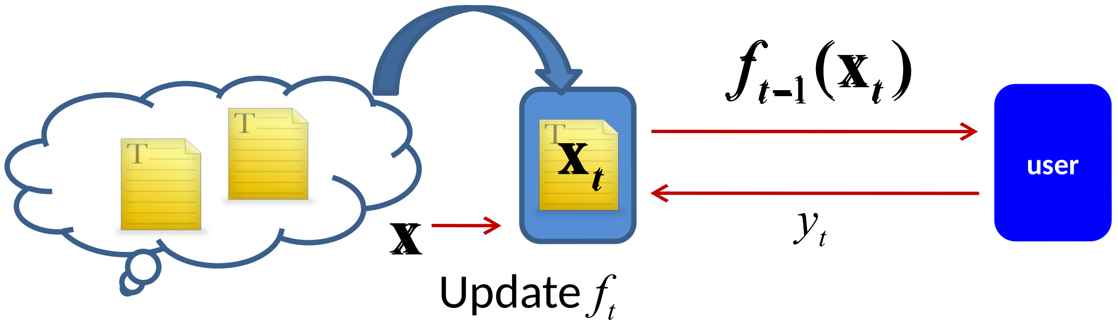

Online Learning
Non-sparse learning
Sparse learning
Unsupervised learning

Online/Stochastic Gradient Descent

- Online gradient descent



- Stochastic gradient descent



Outline

- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

Online Non-Sparse Learning

- Decision function can be linear and non-linear as

- **First order** learning methods
  - **Online gradient descent** (Zinkevich, 2003)
  - **Passive aggressive learning** (Crammer et al., 2006)
  - Others (including but not limited)
    - ALMA: A New Approximate Maximal Margin Classification Algorithm (Gentile, 2001)
    - ROMMA: Relaxed Online Maximum Margin Algorithm (Li and Long, 2002)
    - MIRA: Margin Infused Relaxed Algorithm (Crammer and Singer, 2003)
    - DUOL: A Double Updating Approach for Online Learning (Zhao et al., 2009)

Online Gradient Descent (OGD)
(Zinkevich, 2003)

- Online convex optimization
  - Consider a convex objective function

$$f : S \to \mathbb{R}$$

where $S \subset \mathbb{R}^n$ is a bounded convex set

  - Update by Online Gradient Descent (OGD) or Stochastic Gradient Descent (SGD)

$$\mathbf{w}_{t+1} \leftarrow \underbrace{\prod_S}_{\text{projection}} \big(\underbrace{\boxed{\mathbf{w}_t - \eta \nabla f(\mathbf{w}_t)}}_{\text{gradient descent}}\big)$$

where η is a learning rate

Provide a framework to prove regret bound for online convex optimization

Online Gradient Descent (OGD) (Zinkevich, 2003)

- For t = 1, 2, …
  - An unlabeled sample $\mathbf{x}_t$ arrives
  - Make a prediction based on existing weights

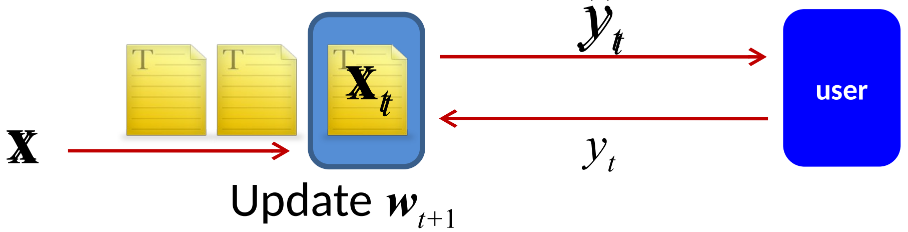
$$\hat{y}_t = \mathrm{sgn}(\mathbf{w}_t^T \mathbf{x}_t)$$

  - Observe the true class label $y_t \in \{-1,+1\}$
  - Update the weights by

$$\mathbf{w}_{t+1} \leftarrow \prod_S(\mathbf{w}_t - \eta \nabla f(\mathbf{w}_t))$$

where η is a learning rate

regret bound is established.

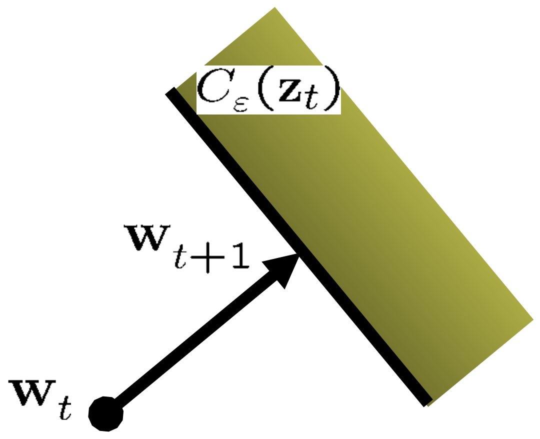


Passive-Aggressive Online Learning (Crammer et al., 2006)

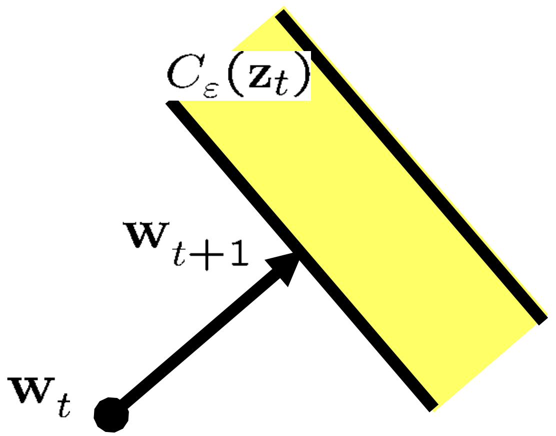
- Each example defines a set of consistent hypotheses: $C_\varepsilon(\mathbf{z}_t) = \{\mathbf{w} \mid \delta(\mathbf{w}; \mathbf{z}_t) \leq \varepsilon\}$
- The new vector $\mathbf{w}_{t+1}$ is set to be the projection of $\mathbf{w}_t$ ont $C_\varepsilon(\mathbf{z}_t)$

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \|\mathbf{w} - \mathbf{w}_t\| \text{ s.t. } \mathbf{w} \in C_\varepsilon(\mathbf{z}_t)$$

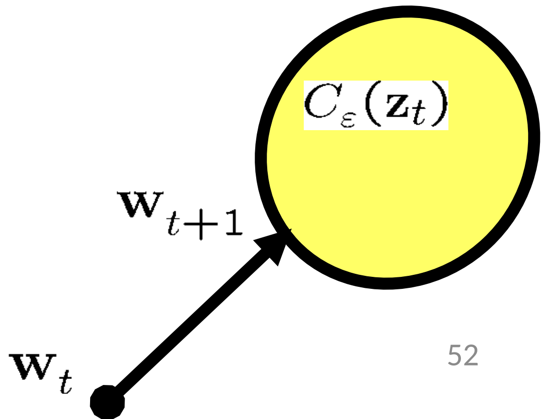
Classification



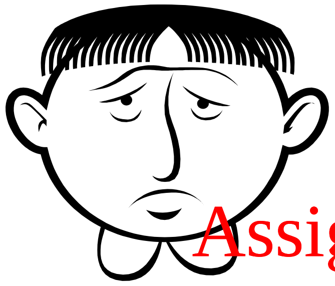
Regression

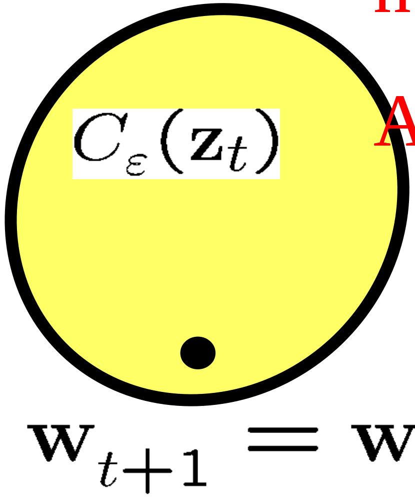


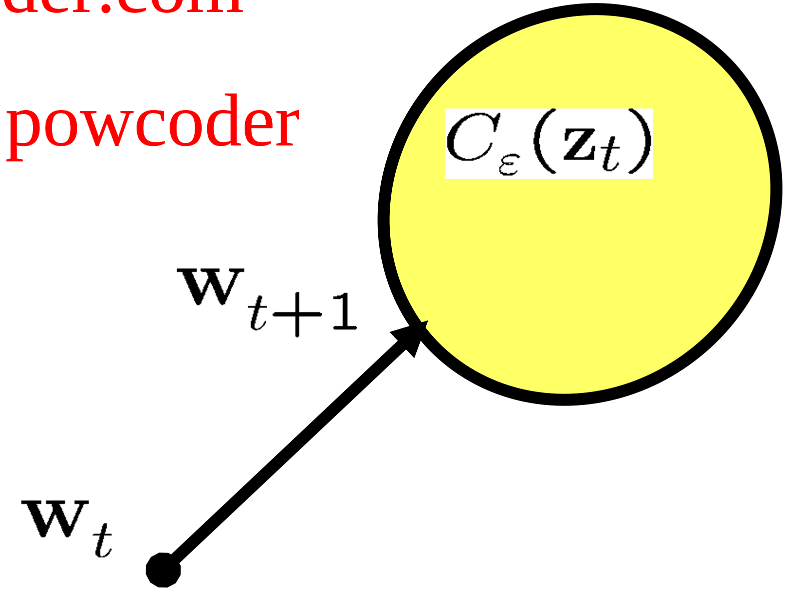
Uniclass



Passive-Aggressive




$C_\varepsilon(\mathbf{z}_t)$
$\mathbf{w}_{t+1} = \mathbf{w}_t$



Passive Aggressive Online Learning
(Crammer et al., 2006)

- PA (Binary classification)

$$\mathbf{w}_{t+1} = \underset{\mathbf{w}\in\mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}-\mathbf{w}_t\|^2$$
$$\text{s.t.} \quad \ell(\mathbf{w};(\mathbf{x}_t,y_t)) = 0.$$

- PA-I (C-SVM)

$$\mathbf{w}_{t+1} = \underset{\mathbf{w}\in\mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}-\mathbf{w}_t\|^2 + C\xi$$
$$\text{s.t.} \quad \ell(\mathbf{w};(\mathbf{x}_t,y_t)) \le \xi \text{ and } \xi \ge 0.$$

- PA-II (Relaxed C-SVM)

$$\mathbf{w}_{t+1} = \underset{\mathbf{w}\in\mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}-\mathbf{w}_t\|^2 + C\xi^2$$
$$\text{s.t.} \quad \ell(\mathbf{w};(\mathbf{x}_t,y_t)) \le \xi.$$

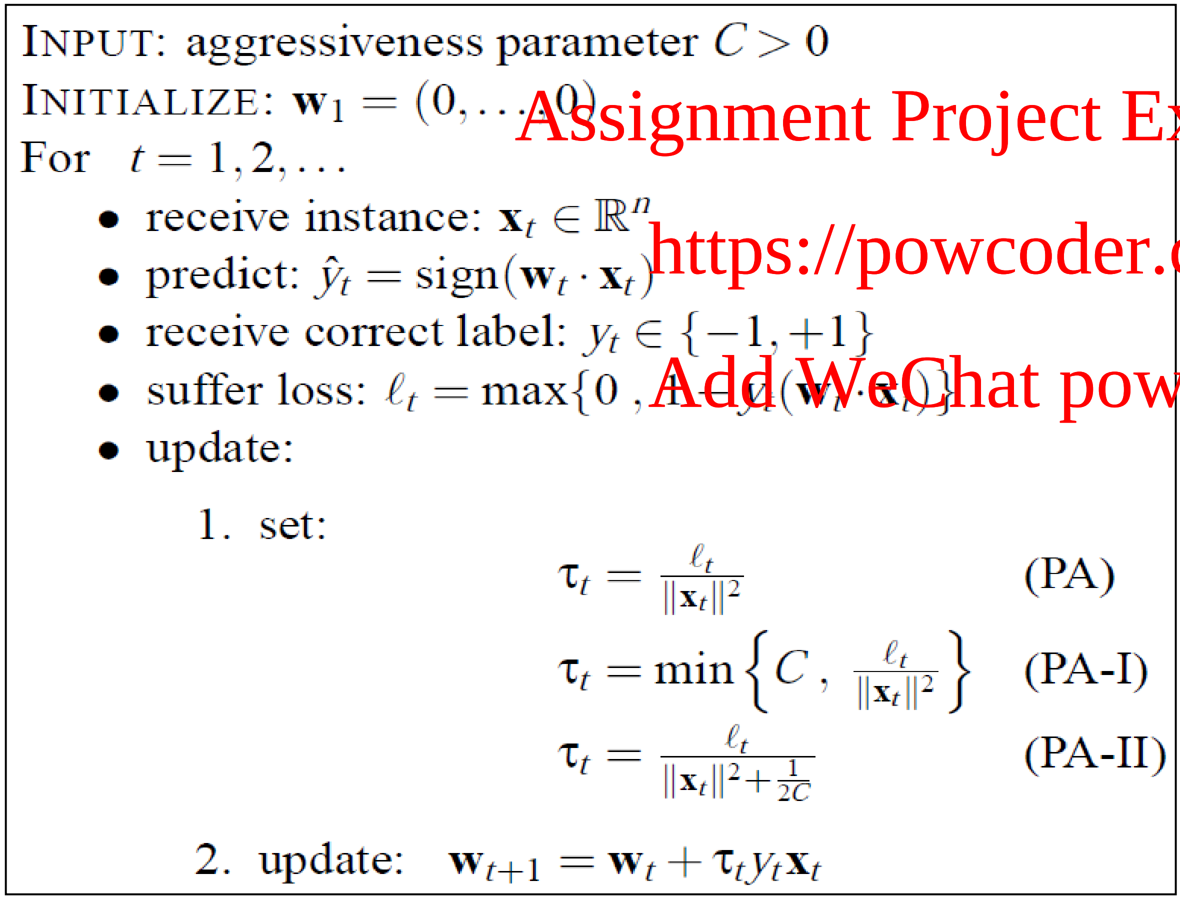
- Closed-form solution

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t y_t \mathbf{x}_t$$

$$\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2} \quad \text{(PA)}$$
$$\tau_t = \min\left\{C, \frac{\ell_t}{\|\mathbf{x}_t\|^2}\right\} \quad \text{(PA-I)}$$
$$\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}} \quad \text{(PA-II)}$$

Passive Aggressive Online Learning
(Crammer et al., 2006)

- Algorithm



```
INPUT: aggressiveness parameter $C > 0$
INITIALIZE: $\mathbf{w}_1 = (0, \ldots, 0)$
For $t = 1, 2, \ldots$
    • receive instance: $\mathbf{x}_t \in \mathbb{R}^n$
    • predict: $\hat{y}_t = \text{sign}(\mathbf{w}_t \cdot \mathbf{x}_t)$
    • receive correct label: $y_t \in \{-1, +1\}$
    • suffer loss: $\ell_t = \max\{0, 1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t)\}$
    • update:
        1. set:
            $\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2}$   (PA)
            $\tau_t = \min\left\{C, \frac{\ell_t}{\|\mathbf{x}_t\|^2}\right\}$   (PA-I)
            $\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}}$   (PA-II)
        2. update: $\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t y_t \mathbf{x}_t$
```

- Objective

$$\mathbf{w}_{t+1} = \underset{\mathbf{w} \in \mathbb{R}^n}{\text{argmin}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|^2$$

$$\text{s.t.} \quad \ell(\mathbf{w}; (\mathbf{x}_t, y_t)) = 0$$

- Closed-form solutions

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t y_t \mathbf{x}_t$$

$$\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2} \quad \text{(PA)}$$

$$\tau_t = \min\left\{C, \frac{\ell_t}{\|\mathbf{x}_t\|^2}\right\} \quad \text{(PA-I)}$$

$$\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}} \quad \text{(PA-II)}$$

Online Non-Sparse Learning

- **First order** methods
  - Learn a **linear** weight vector (first order) of model

- Pros and Cons

😆 Simple and easy to implement

😆 Efficient and scalable for high-dimensional data

😫 Relatively slow convergence rate

Online Non-Sparse Learning

- **Second order** online learning methods
  - Update the weight vector **w** by maintaining and exploring both first-order and second-order information
- Some representative methods, but not limited
  - SOP: Second Order Perceptron (Cesa-Bianchi et al., 2005)
  - CW: Confidence Weighted learning (Dredze et al., 2008)
  - AROW: Adaptive Regularization of Weights (Crammer et al., 2009)
  - IELLIP: Online Learning by Ellipsoid Method (Yang et al., 2009)
  - NHERD: Gaussian Herding (Crammer & Lee 2010)
  - NAROW: New variant of AROW algorithm (Orabona & Crammer 2010)
  - SCW: Soft Confidence Weighted (SCW) (Hoi et al., 2012)
- Pros and Cons

  😆 Faster convergence rate

  😢 Expensive for high-dimensional data

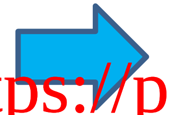
  😢 Relatively sensitive to noise

Outline

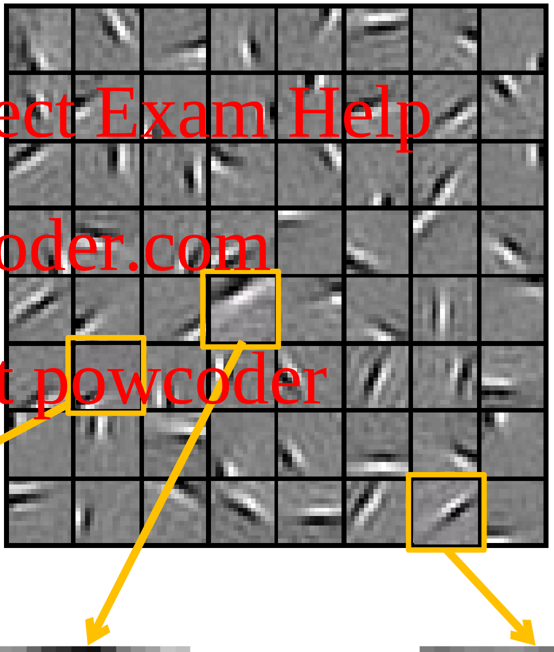
- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

Sparse Learning

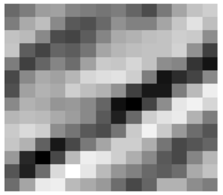
Natural Images



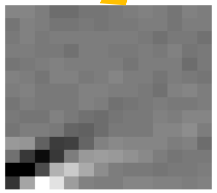
Learned bases ($\phi_{1}, ..., \phi_{64}$): “Edges”



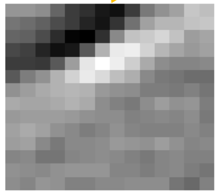
Test example



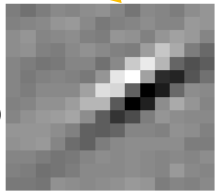
≈ 0.8 *



$+ 0.3$ *



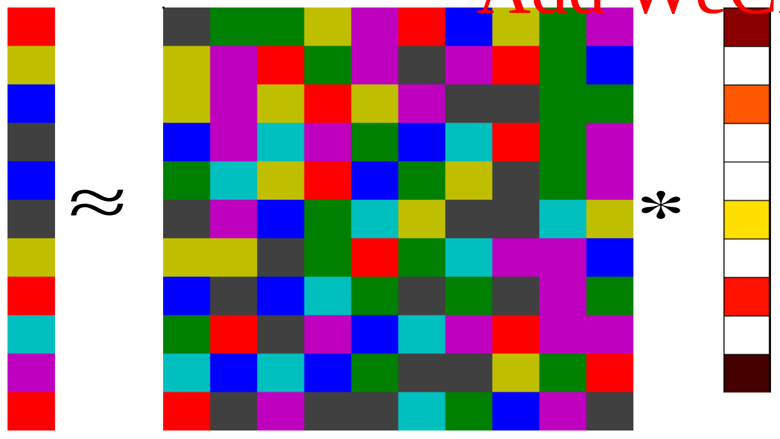
$+ 0.5$



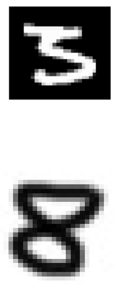
$x \quad \approx 0.8 \quad * \quad \phi_{36} \quad + \ 0.3 \quad * \quad \phi_{42} \quad + \quad 0.5 \quad * \quad \phi_{63}$

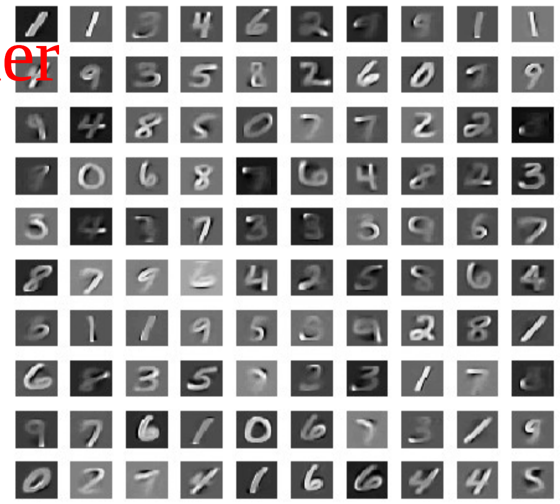
Online Sparse Learning

- Motivation
  - Space constraint: RAM overflow
  - Test-time constraint
  - How to induce Sparsity in the weights of online learning algorithms?



$$Y = \mathbf{X} \cdot \mathbf{w}$$

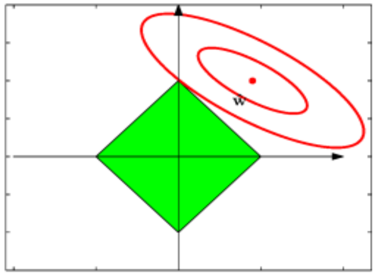




Online Sparse Learning

- Objective function

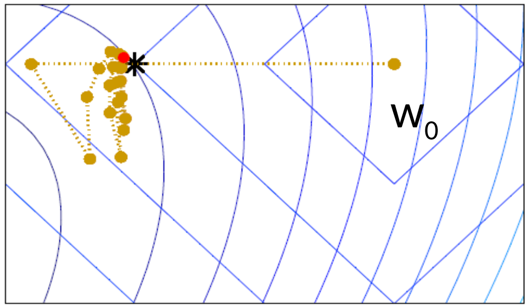
$$\hat{w} = \arg\min_{w} \sum_{i=1}^{n} L(w, z_i) + g\|w\|_1$$



- Problem in online learning
  - Standard stochastic gradient descent

$$f(w_i) = w_i - \eta \nabla_1 L(w_i, z_i)$$

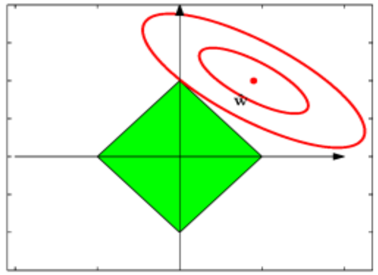
  - It does not yield sparse solution



Subgradient

- Some representative work
  - **Truncated gradient** (Langford et al., 2009)
  - FOBOS: Forward Looking Subgradients (Duchi and Singer, 2009)
  - **Dual averaging** (Xiao, 2009)
  - etc.

Truncated Gradient (Langford et al., 2009)

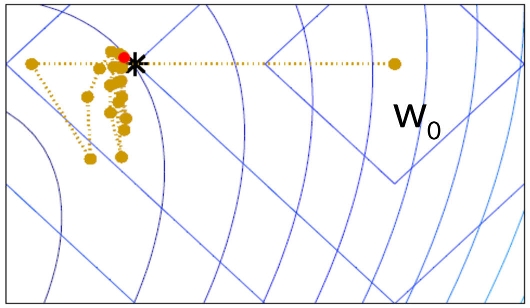


- Objective function

$$\hat{w} = \arg\min_{w} \sum_{i=1}^{n} L(w, z_i) + g\|w\|_1$$

- Stochastic gradient descent

$$f(w_i) = w_i - \eta \nabla_1 L(w_i, z_i)$$



Subgradient

- Simple coefficient rounding

$$f(w_i) = T_0(w_i - \eta \nabla_1 L(w_i, z_i), \theta)$$

when the coefficient is small

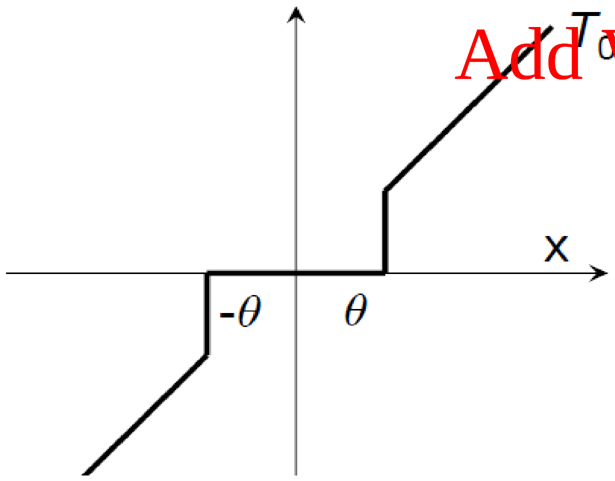
Truncated gradient: impose sparsity by modifying the stochastic gradient descent

Truncated Gradient (Langford et al., 2009)

Simple Coefficient Rounding vs. Less aggressive truncation

$$T_0(v_j,\theta) = \begin{cases} 0 & \text{if } |v_j| \le \theta \\ v_j & \text{otherwise} \end{cases}$$

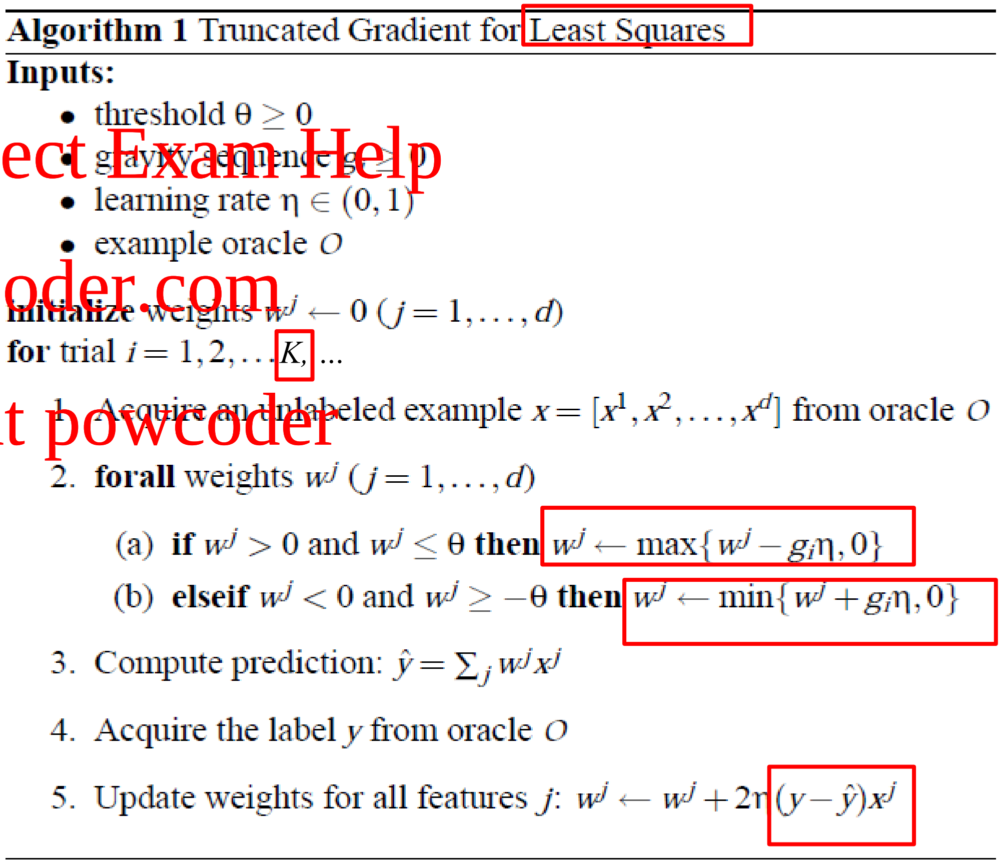
$$T_1(v_j,\alpha,\theta) = \begin{cases} \max(0, v_j - \alpha) & \text{if } v_j \in [0,\theta] \\ \min(0, v_j + \alpha) & \text{if } v_j \in [-\theta, 0] \\ v_j & \text{otherwise} \end{cases}$$



Truncated Gradient (Langford et al., 2009)

$$f(w_i) = T_1(w_i - \eta \nabla_1 L(w_i, z_i), \eta g_i, \theta)$$

- The amount of shrinkage is measured by a *gravity* parameter
- When , the update rule is identical to the standard SGD
- The truncation can be performed every *K* online steps
- Loss functions: $L(w, z) = \phi(w^T x, y)$
  - Logistic $\phi(p, y) = \ln(1 + \exp(-py))$
  - SVM (hinge) $\phi(p, y) = \max(0, 1 - py)$
  - Least square $\phi(p, y) = \boxed{(p - y)^2}$



```
Algorithm 1 Truncated Gradient for Least Squares
Inputs:
  • threshold $\theta \ge 0$
  • gravity sequence $g_i \ge 0$
  • learning rate $\eta \in (0, 1)$
  • example oracle $O$
initialize weights $w^j \leftarrow 0$ $(j = 1, \ldots, d)$
for trial $i = 1, 2, \ldots$ $\boxed{K,}$ ...
  1. Acquire an unlabeled example $x = [x^1, x^2, \ldots, x^d]$ from oracle $O$
  2. forall weights $w^j$ $(j = 1, \ldots, d)$
      (a) if $w^j > 0$ and $w^j \le \theta$ then $\boxed{w^j \leftarrow \max\{w^j - g_i\eta, 0\}}$
      (b) elseif $w^j < 0$ and $w^j \ge -\theta$ then $\boxed{w^j \leftarrow \min\{w^j + g_i\eta, 0\}}$
  3. Compute prediction: $\hat{y} = \sum_j w^j x^j$
  4. Acquire the label $y$ from oracle $O$
  5. Update weights for all features $j$: $w^j \leftarrow w^j + 2\eta\boxed{(y - \hat{y})x^j}$
```

Truncated Gradient (Langford et al., 2009)

- Theoretical result (*T*: No. of samples)

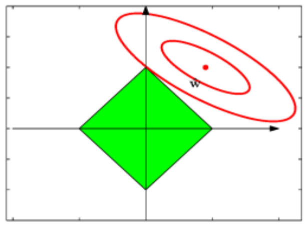
$$\frac{1-0.5A\eta}{T}\sum_{i=1}^{T}\left[L(w_i,z_i)+\frac{g_i}{1-0.5A\eta}\|w_{i+1}\cdot I(w_{i+1}\leq\theta)\|_1\right]$$
$$\leq\frac{\eta}{2}B+\frac{\|\bar{w}\|^2}{2\eta T}+\frac{1}{T}\sum_{i=1}^{T}[L(\bar{w},z_i)+g_i\|\bar{w}\cdot I(w_{i+1}\leq\theta)\|_1],$$

- Let , the regret is

$$\sum_{i=1}^{T}(L(w_i,z_i)+g\|w_i\|_1)-\sum_{i=1}^{T}(L(\bar{w},z_i)+g\|\bar{w}\|_1)$$
$$\leq\frac{\sqrt{T}}{2}(B+\|\bar{w}\|^2)\left(1+\frac{A}{2\sqrt{T}}\right)+\frac{A}{2\sqrt{T}}\left(\sum_{i=1}^{T}L(\bar{w},z_i)+g\sum_{i=1}^{T}(\|\bar{w}\|_1-\|w_{i+1}\|_1)\right)+o(\sqrt{T})$$

regret bound is established.

Dual Averaging (Xiao, 2010)



- Objective function

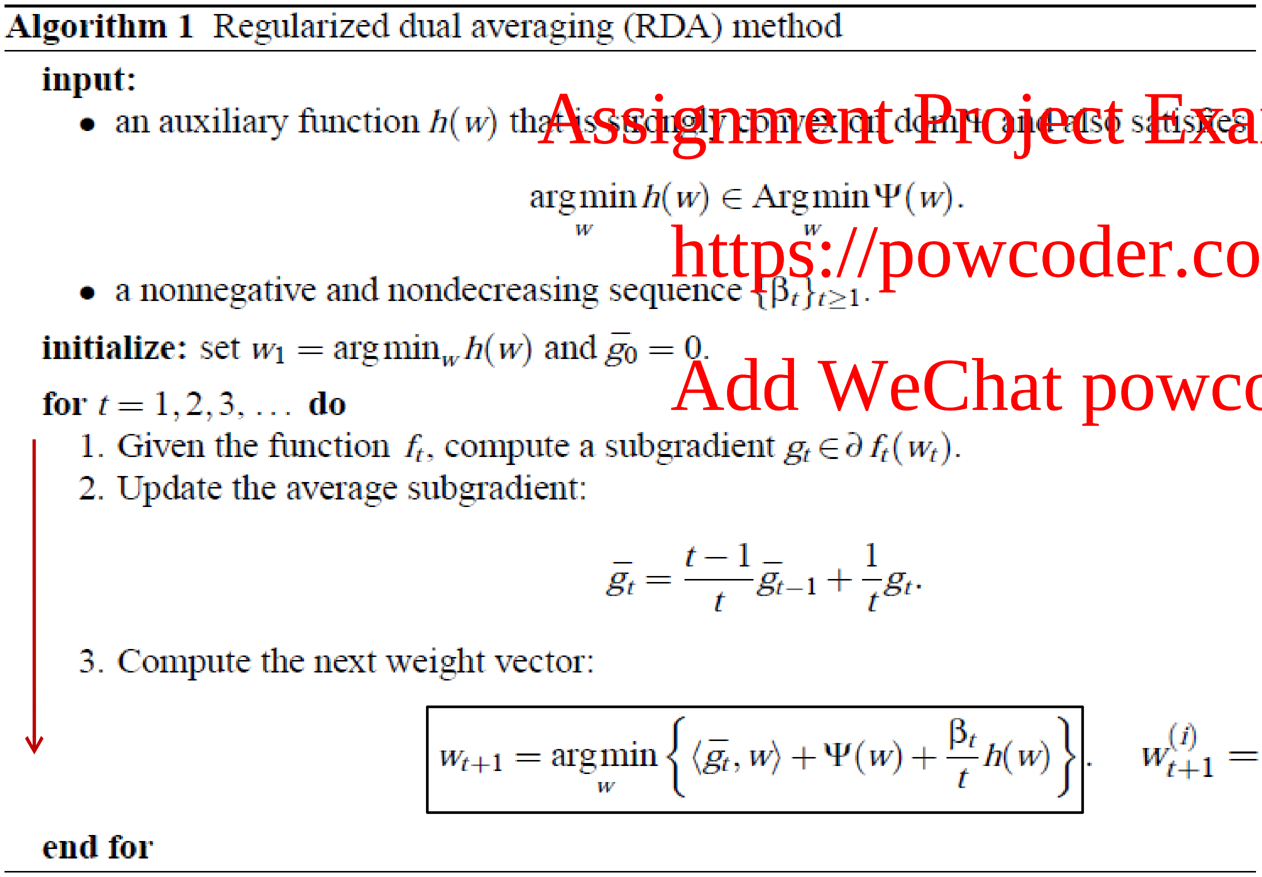
$$\underset{w}{\text{minimize}} \quad \left\{ \phi(w) \triangleq \mathbf{E}_z f(w,z) + \Psi(w) \right\} \quad \Psi(w) = \lambda \|w\|_1 \text{ with } \lambda > 0$$

- Problem: truncated gradient doesn't produce truly sparse weight due to small learning rate

- Fix: dual averaging which keeps two state representations:
  - parameter w_t
  - average gradient vector $\bar{g}_t = \frac{1}{t}\sum_{i=1}^{t} f_i(w_i)$

Dual Averaging (Xiao, 2010)

- Algorithm



Algorithm 1 Regularized dual averaging (RDA) method

```
input:
  • an auxiliary function $h(w)$ that is strongly convex on dom Ψ and also satisfies
        $\arg\min_w h(w) \in \operatorname{Arg\,min}_w \Psi(w)$.
  • a nonnegative and nondecreasing sequence $\{\beta_t\}_{t\geq 1}$.
initialize: set $w_1 = \arg\min_w h(w)$ and $\bar{g}_0 = 0$.
for $t = 1, 2, 3, \ldots$ do
  1. Given the function $f_t$, compute a subgradient $g_t \in \partial f_t(w_t)$.
  2. Update the average subgradient:
        $\bar{g}_t = \frac{t-1}{t}\bar{g}_{t-1} + \frac{1}{t} g_t$.
  3. Compute the next weight vector:
        $\boxed{w_{t+1} = \arg\min_w \left\{ \langle \bar{g}_t, w\rangle + \Psi(w) + \frac{\beta_t}{t} h(w) \right\}}$.
end for
```

- has entry-wise closed-form solution
- Advantage: sparse on the weight
- Disadvantage: keep a non-sparse subgradient

$$w_{t+1}^{(i)} = \begin{cases} 0 & \text{if } \left|\bar{g}_t^{(i)}\right| \leq \lambda, \\ -\frac{\sqrt{t}}{\gamma}\left(\bar{g}_t^{(i)} - \lambda \operatorname{sgn}\left(\bar{g}_t^{(i)}\right)\right) & \text{otherwise,} \end{cases}$$

Convergence and Regret

- Average regret

$$\bar{R}_T(w) \triangleq \frac{1}{T}\sum_{t=1}^{T}\left(f_t(w_t)+\Psi(w_t)\right) - S_T(w)$$

$$S_T(w) \triangleq \frac{1}{T}\sum_{t=1}^{T}\left(f_t(w)+\Psi(w)\right)$$

- Theoretical bound: similar to gradient descent

$$\bar{R}_T \sim \mathcal{O}(1/\sqrt{T})$$
$$\bar{R}_T \sim \mathcal{O}(\log(T)/T), \quad \text{if } h(\cdot) \text{ is strongly convex}$$

average regret bound is established.

Variants of Online Sparse Learning Models

- Online feature selection (OFS)
  - A variant of sparse online learning
  - The key difference is that OFS focuses on selecting a fixed subset of features in online learning process
  - Could be used as an alternative tool for batch feature selection when dealing with big data
- Other existing work
  - Online learning for Group Lasso (Yang et al., 2010) and online learning for multi-task feature selection (Yang et al. 2013) to select features in group manner or features among similar tasks

Online Sparse Learning

- Objective
  - Induce sparsity in the weights of online learning algorithms

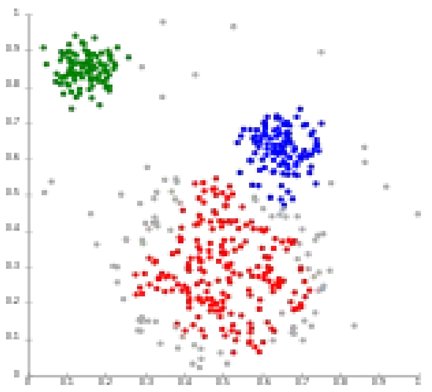
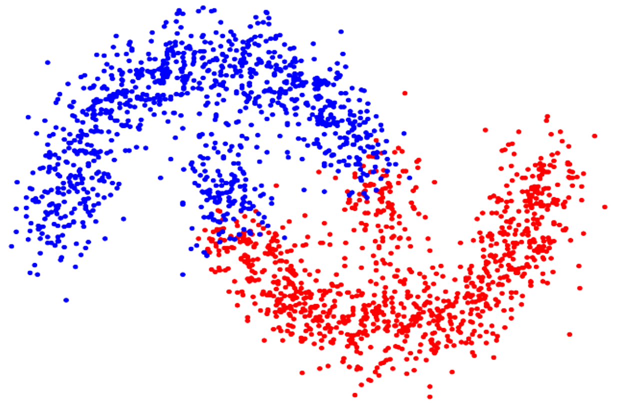
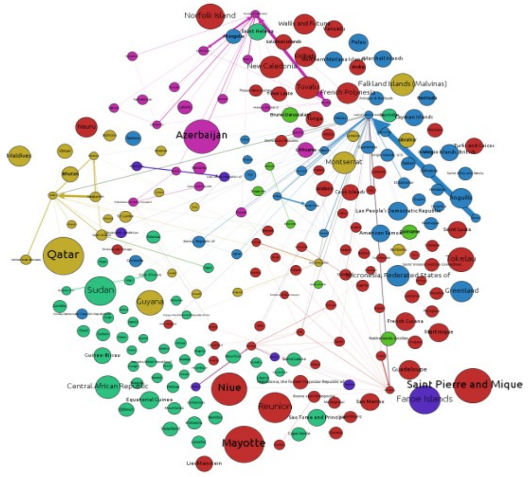
- Pros and Cons
  - 😆 Simple and easy to implement
  - 😆 Efficient and scalable for high-dimensional data
  - 😫 Relatively slow convergence rate
  - 😫 No perfect way to attain sparsity solution yet

Outline

- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

Online Unsupervised Learning

- Assumption: data generated from some underlying parametric probabilistic density function
- Goal: estimate the parameters of the density to give a suitable compact representation

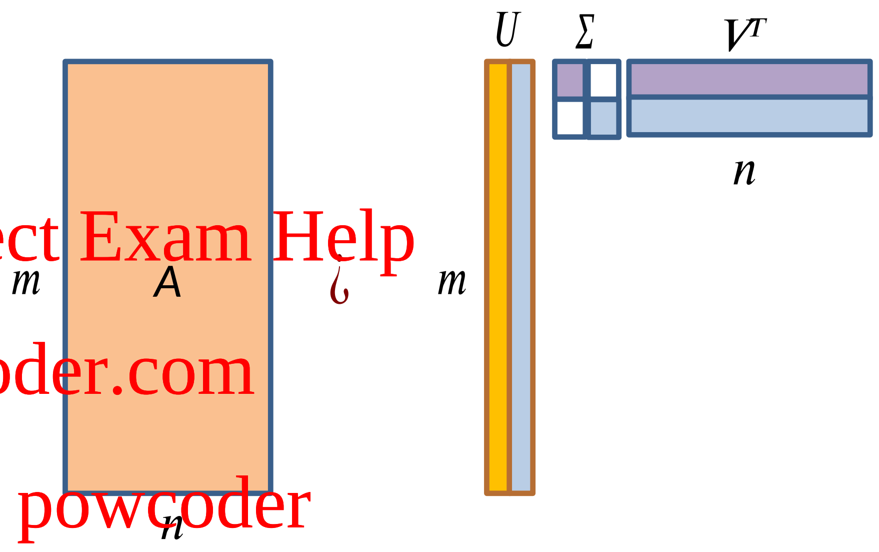




Online Unsupervised Learning

- Some representative work
  - **Online singular value decomposition** (SVD) (Brand, 2003)
  - Online principal component analysis (PCA) (Warmuth and Kuzmin, 2006)
  - Online dictionary learning for sparse coding (Mairal et al. 2009)
  - Online learning for latent Dirichlet allocation (LDA) (Hoffman et al., 2010)
  - Online variational inference for the hierarchical Dirichlet process (HDP) (Wang et al. 2011)
  - Online Learning for Collaborative Filtering (Ling et al. 2012)
  - …

SVD: Definition

- : input data matrix
  - matrix (e.g. documents, terms)
- : left singular vectors
  - matrix (documents, topics)
- : singular values
  - diagonal matrix (strength of each "topic")
  - rank of matrix
  - Nonnegative and sorted
- : right singular vectors
  - matrix (terms, topics)



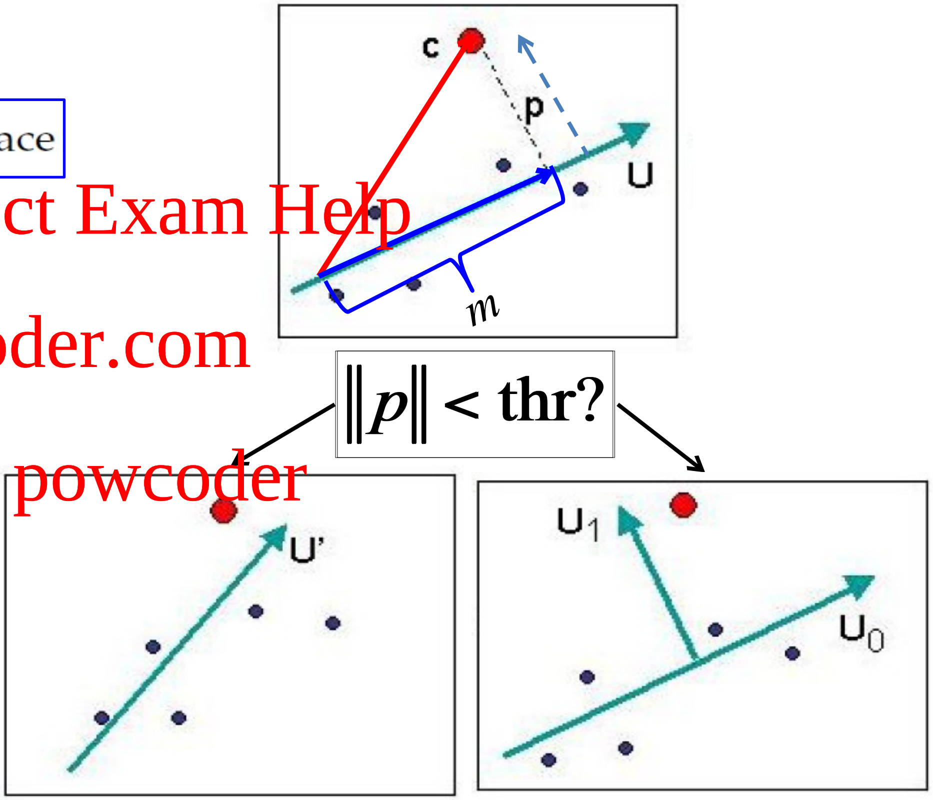
- : scalar
- : vector
- : vector

Online SVD (Brand, 2003)

- Challenges: storage and computation
- Idea: an incremental algorithm computes the principal eigenvectors of a matrix without storing the entire matrix in memory

Online SVD (Brand, 2003)

```
1: Existing rank-$r$ PCA
       $$A = U\Sigma V^T$$
2: A new sample $c$ arrives, project it onto eigenspace
       $$m = U^T c$$
3: Compute the orthogonal component
       $$p = c - Um$$
4: if $\|p\| < thr$ then
5:     Incorporate the new sample by rotating $U$
           $$U = UR_u, \quad V = VR_v$$
6: else
7:     increase a rank
           $$U' = [U; \overset{\downarrow}{m}]R_u, \quad V' = VR_v$$
8: end if
9: Rotation by re-diagonalizing the matrix
       $$\begin{pmatrix} \text{diag}(S) & m \\ 0 & \|p\| \end{pmatrix} \longrightarrow [R_u, R_v]$$
```



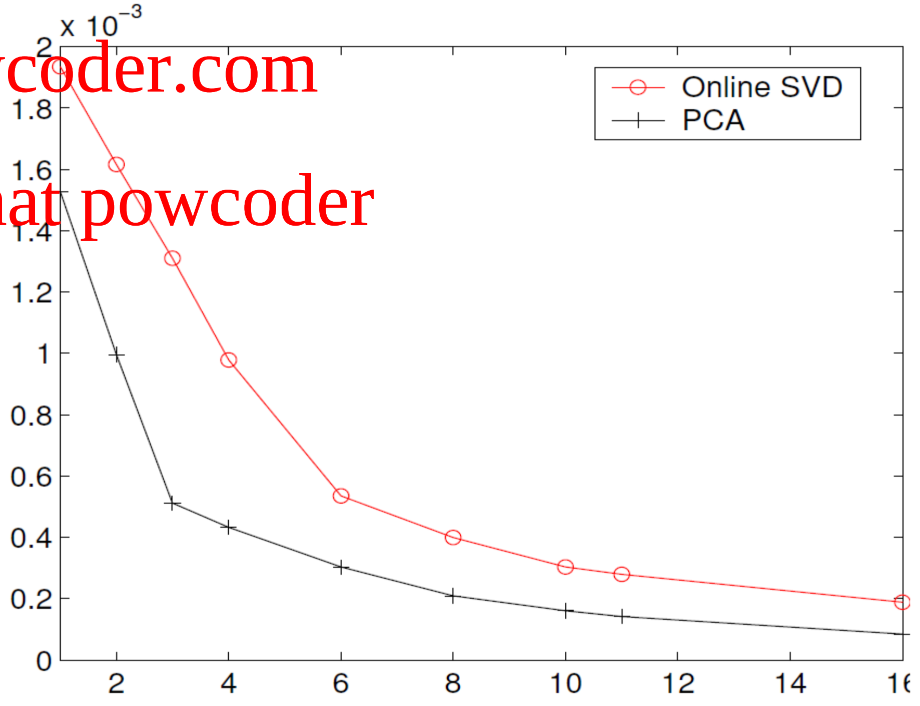
Online SVD (Brand, 2003)

- Complexity

$$O(r^2)$$

- Store
  -

- The online SVD has more error, but it is comparable to PCA (SVD)



Online SVD

- Unsupervised learning: minimizing the reconstruction errors
- Each update will increase the rank by at most one, until a user-specified ceiling is reached
- Pros and Cons
  - 😆 Simple to implement
  - 😆 Fast computation
  - 😆 Comparable performance
  - 😰 Lack of theoretical guarantee

Outline

- Introduction
  - Learning paradigms
  - Online learning and its applications
- Online learning algorithms
  - Perceptron
  - Online non-sparse learning
  - Online sparse learning
  - Online unsupervised learning
- Conclusion

One-slide Takeaway

- Basic concepts
  - What is online learning?
  - What is regret analysis?
- Online learning algorithms
  - Perceptron
  - Online gradient descent
  - Passive aggressive
  - Truncated gradient
  - Dual averaging
  - Online SVD

Resources

- Book and Video:
  - Prediction Learning and Games. N. Cesa-Bianchi and G. Lugosi. Cambridge university press, 2006.
  - [Shal11] Online Learning and Online Convex Optimization. Shai Shalev-Shwartz. Foundations and Trends in Machine Learning, Vol. 4, No. 2, 2011, 107-194. DOI: 10.1561/2200000018
  - http://videolectures.net/site/search/?q=online+learning
- Software:
  - Pegasos: http://www.cs.huji.ac.il/~shais/code/index.html
  - VW: hunch.net/~vw/
  - SGD by Leon Bottou: http://leon.bottou.org/projects/sgd

References

- Cesa-Bianchi, Nicol`o, Conconi, Alex, and Gentile, Claudio. A second-order perceptron algorithm. SIAM J. Comput., 34(3):640–668, 2005.
- M. Brand. Fast online svd revisions for lightweight recommender systems. In SDM, 2003.
- N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order perceptron algorithm. SIAM J. Comput., 34(3):640–668, 2005.
- K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. Journal of Machine Learning Research, 7:551–585, 2006.
- K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. In NIPS, pages 414–422, 2009.
- K. Crammer and D. D. Lee. Learning via gaussian herding. In NIPS, pages 451–459, 2010.
- K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. Journal of Machine Learning Research, 3:951–991, 2003.
- M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. In ICML, pages 264–271, 2008.
- C. Gentile. A new approximate maximal margin classification algorithm. Journal of Machine Learning Research, 2:213–242, 2001.
- M. D. Hoffman, D. M. Blei, and F. R. Bach. Online learning for latent dirichlet allocation. In NIPS, pages 856–864, 2010.
- S. C. H. Hoi, J. Wang, and P. Zhao. Exact soft confidence-weighted learning. In ICML, 2012.

References

- J. Langford, L. Li, and T. Zhang. Sparse online learning via truncated gradient. Journal of Machine Learning Research, 10:777–801, 2009.
- Y. Li and P. M. Long. The relaxed online maximum margin algorithm. Machine Learning, 46(1-3):361–387, 2002.
- Guang Ling, Haiqin Yang, Irwin King and M.R. Lyu. Online Learning for Collaborative Filtering, IJCNN, Brisbane, Australia, 2012.
- P. L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. SIAM J. Numer. Anal., 16(6):964–979, 1979.
- J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In ICML, page 87, 2009.
- Y. Nesterov. Gradient methods for minimizing composite objective function. CORE Discussion Paper 2007/76, Catholic University of Louvain, Center for Operations Research and Econometrics, 2007.
- F. Orabona and K. Crammer. New adaptive algorithms for online classification. In NIPS, pages 1840–1848, 2010.
- F. Rosenblatt. The Perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65:386–408, 1958.

References

- C. Wang, J. W. Paisley, and D. M. Blei. Online variational inference for the hierarchical dirichlet process. Journal of Machine Learning Research - Proceedings Track, 15:752–760, 2011.
- M. K. Warmuth and D. Kuzmin. Randomized pca algorithms with regret bounds that are logarithmic in the dimension. In NIPS, pages 1481–1488, 2006.
- S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation. IEEE Transactions on Signal Processing, 57(7):2479–2493, 2009.
- L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. Journal of Machine Learning Research, 11:2543–2596, 2010.
- H. Yang, M. R. Lyu, and I. King. Efficient online learning for multi-task feature selection. ACM Transactions on Knowledge Discovery from Data, 2013.
- H. Yang, Z. Xu, I. King, and M. R. Lyu. Online learning for group lasso. In ICML, pages 1191–1198, Haifa, Israel, 2010.
- L. Yang, R. Jin, and J. Ye. Online learning by ellipsoid method. In ICML, page 145, 2009.
- P. Zhao, S. C. H. Hoi, and R. Jin. Duol: A double updating approach for online learning. In NIPS, pages 2259–2267, 2009.
- M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In ICML, pages 928–936, 2003.

In-class Practice

- We have two data and , how to get a classifier by Perceptron learning rule?
- Assume
  - is in class (the first data)
  - is in class
- Data points are linearly separable and can be applied repeatedly (for validation).