CS373 Data Mining and Machine Learning Lecture 6

Jean Honorio

Purdue University

(originally prepared by Tommi Jaakkola, MIT CSAIL)

Today's topics

- Multi-way classification
  - reducing multi-class to binary
  - margin based solution
- Rating (ordinal regression)
  - reduction to binary problems
  - SVM solution, on-line solution
- Ranking
  - ranking SVM

Reducing multi-class to binary

- We train several classifiers. For a test point, we output a *string* (multi way label). We then check which matrix row is closest to the string.

binary tasks

$$
\text{classes}\quad
\begin{array}{c}
1 \\ 2 \\ 3 \\ 4
\end{array}
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 1 & 0 \\
0 & -1 & 0 & -1 & 0 & 1 \\
0 & 0 & -1 & 0 & -1 & -1
\end{bmatrix}
$$

string for class 2

- Properties of good code matrices
  - "binary codes" (rows) should be well-separated: *if minimum Hamming distance between rows is H, we can make at most H/2 mistakes (good error correction)*
    - Which seems better: one-versus-all or all-pairs?
  - binary tasks (columns) should be easy to solve
    - Which seems better: one-versus-all or all-pairs?

Reducing multi-class to binary

- We train several classifiers. For a test point, we output a *string* (multi way label). We then check which matrix row is closest to the string.

binary tasks j

classes y

$$
\begin{array}{c}
1 \\ 2 \\ 3 \\ 4
\end{array}
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 1 & 0 \\
0 & -1 & 0 & -1 & 0 & 1 \\
0 & 0 & -1 & 0 & -1 & -1
\end{bmatrix}
$$

$$
\hat{y} = \operatorname*{argmin}_{y} \sum_{j=1}^{m} \text{Loss}\Big(\overbrace{R(y,j)\, \hat{\underline{\theta}}_j \cdot \phi(\underline{x})}^{\text{agreement}} \Big)
$$

target binary label for the jth classifier if the multi-class label is y

discriminant function value of the jth classifier in response to the new example

Output codes, error correction

- If the loss is the hinge loss loss(z) = max(0,1-z), then

$$\#\ \text{multi-class errors on the training set}$$

$$\leq \frac{1}{H} \sum_{t=1}^{n} \left[\sum_{j=1}^{m} \mathrm{Loss}\Big(R(y_t, j)\, \underline{\hat{\theta}}_j \cdot \underline{\phi}(\underline{x}_t) \Big) \right]$$

small if code words are well-separated

small if each binary task can be solved well

(See Theorem 1 in [2] if interested.)

Today's topics

- Multi-way classification
  - reducing multi-class to binary
  - margin based solution
- Rating (ordinal regression)
  - reduction to binary problems
  - SVM solution, on-line solution
- Ranking
  - ranking SVM

Rating problems

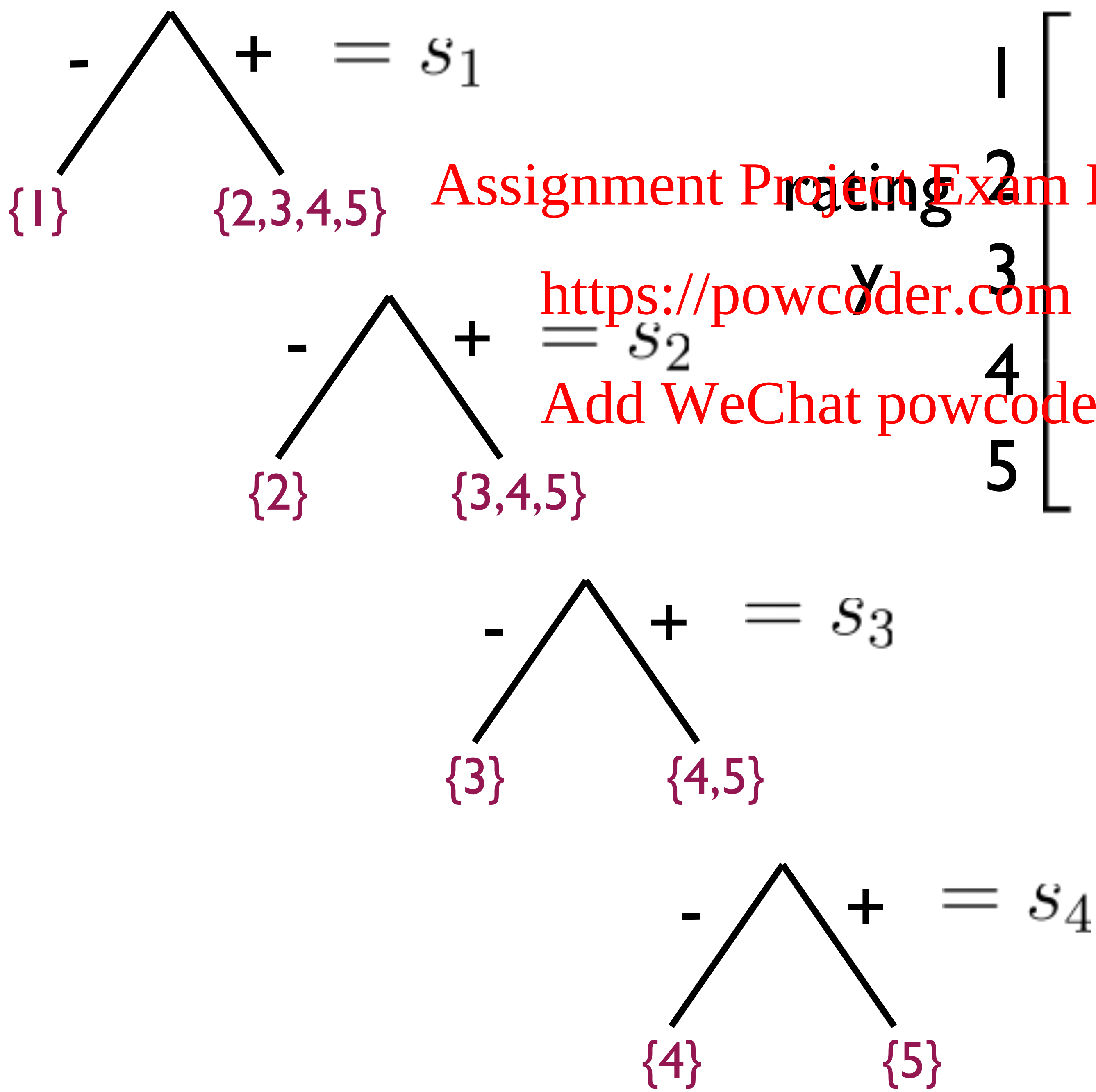
- A common prediction problem in recommender systems involves rating items (movies, products) on the basis some known features about such objects
- The rating scale is often 1-5 stars assigned to the object
- The key difference between rating problems and multi-way classification problems is that the rating scale is ordinal (e.g., $1<2<3<4<5$) while class labels in multi-way classification problems are category symbols

Ordinal regression: setup

- Each item x_i is associated with a feature vector $\phi(x_i)$
  - e.g., product description, movie features, etc.
- We wish to predict an ordinal label $y_i \in \{1, \ldots, k\}$ for each item (reflecting views of one user)
- As in the multi-class setting, we translate each rating into a set of binary labels

Binary translation

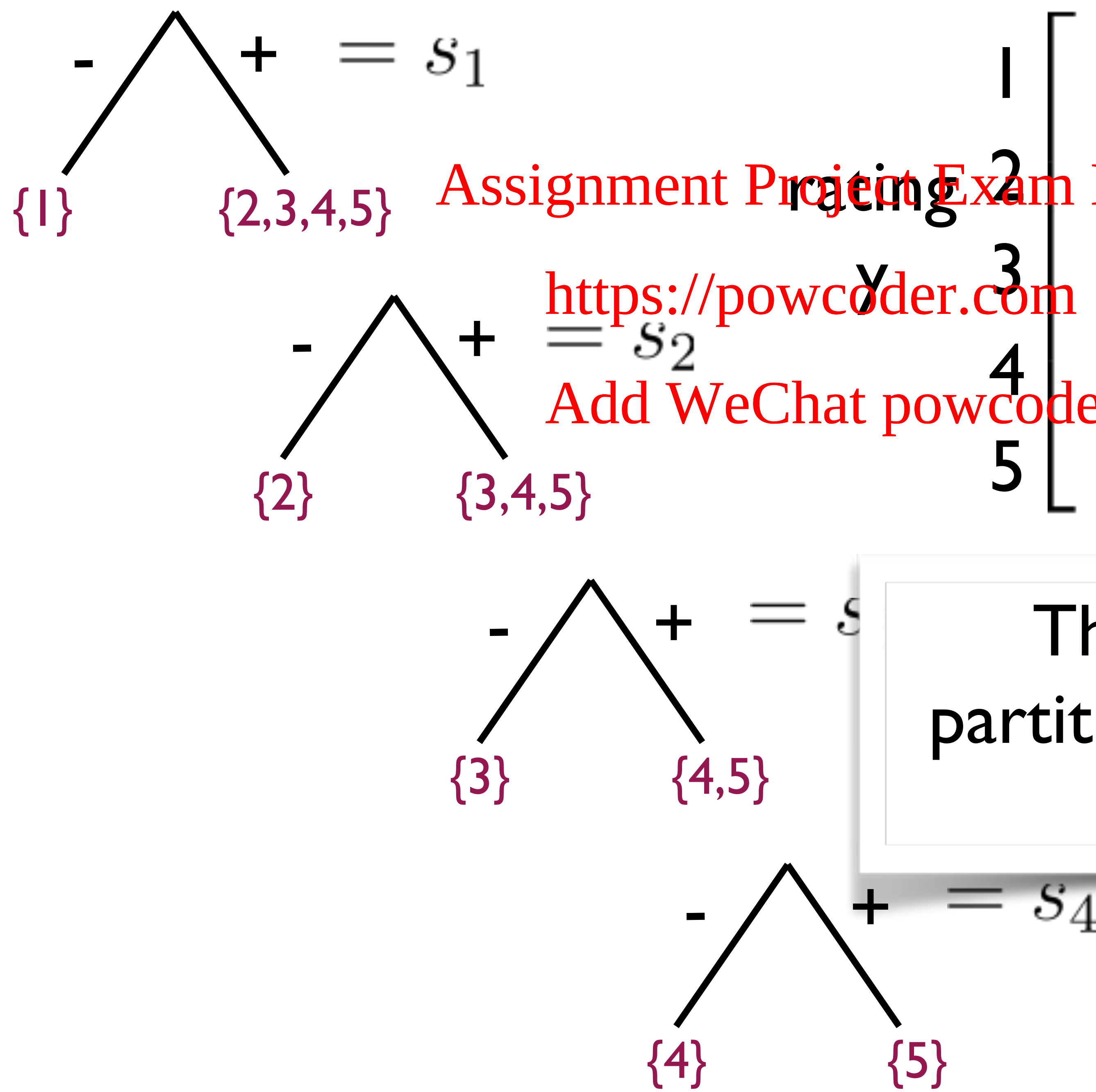
- There are many ways to translate ratings into binary labels...



$$\begin{array}{cc} & \text{binary tasks} \\ \begin{array}{c}\text{rating } y\end{array} \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Binary translation

- There are many ways to translate ratings into binary labels...



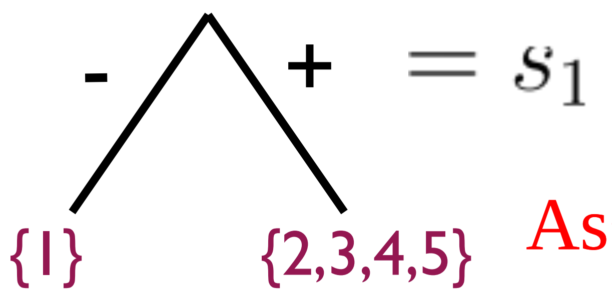
binary tasks

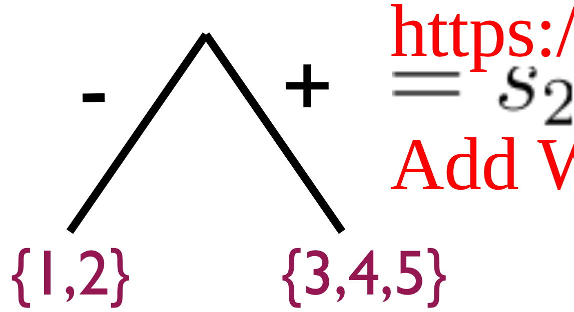
$$\text{rating } y \quad \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{bmatrix} -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

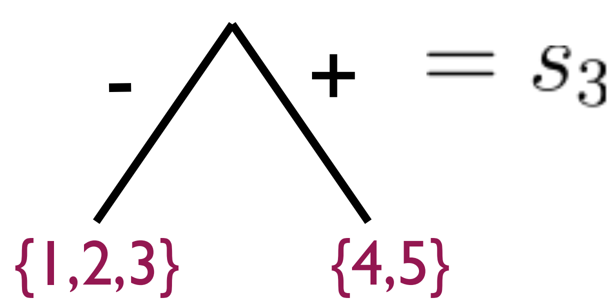
These are independent partitions that do not enforce the ordinal scale

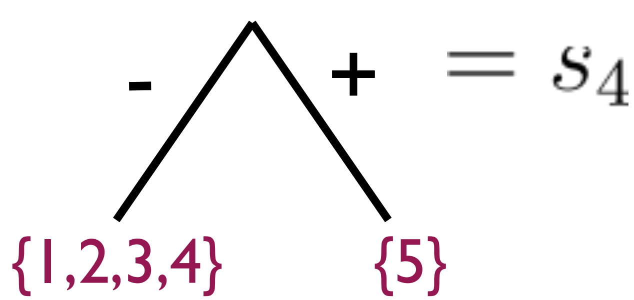
Binary translation

- We can create more relevant partitions by "sliding" across the ordinal scale







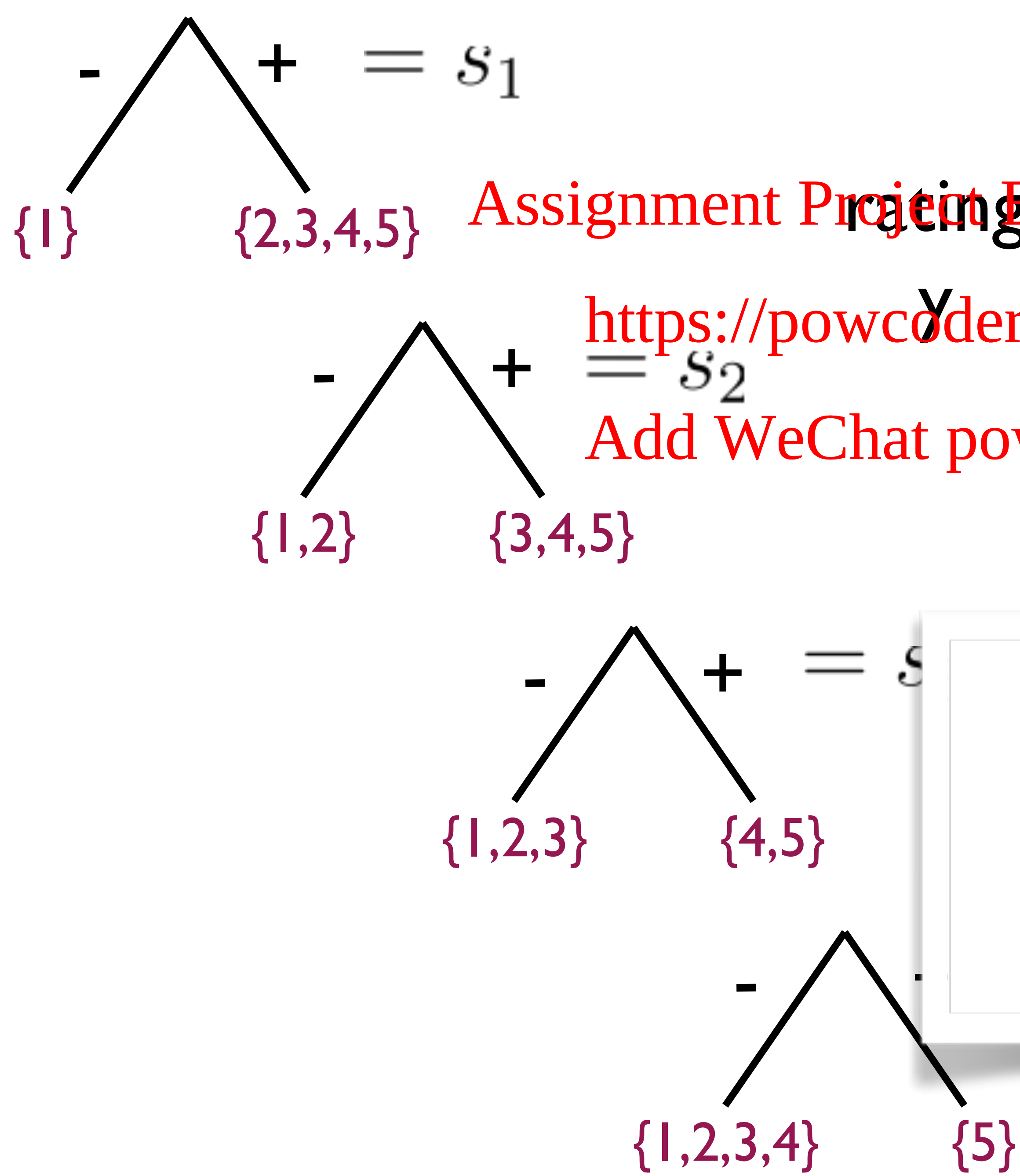


binary tasks

$$\text{rating } y \quad \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{bmatrix} -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Binary translation

- We can create more relevant partitions by “sliding” across the ordinal scale



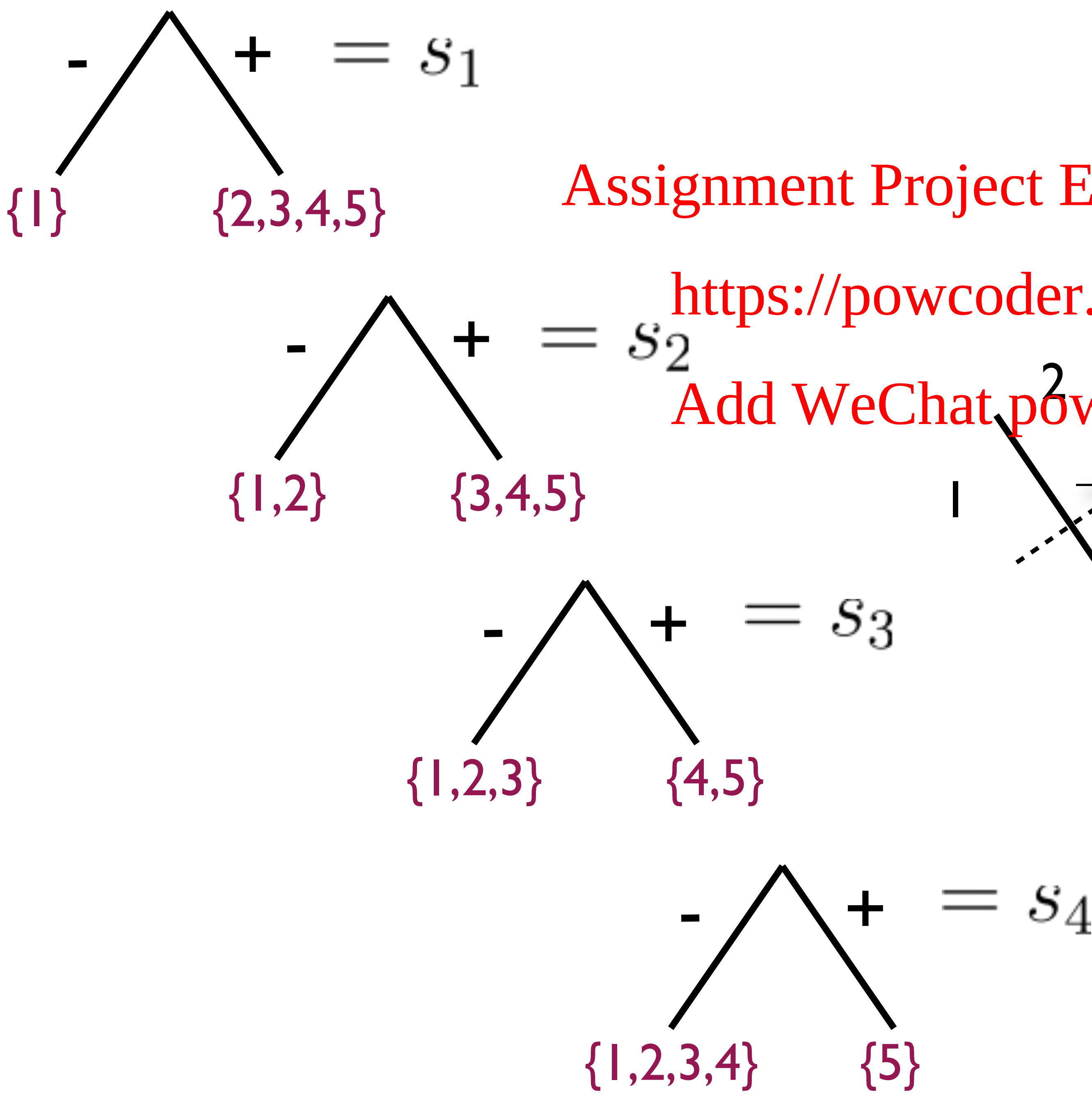
binary tasks

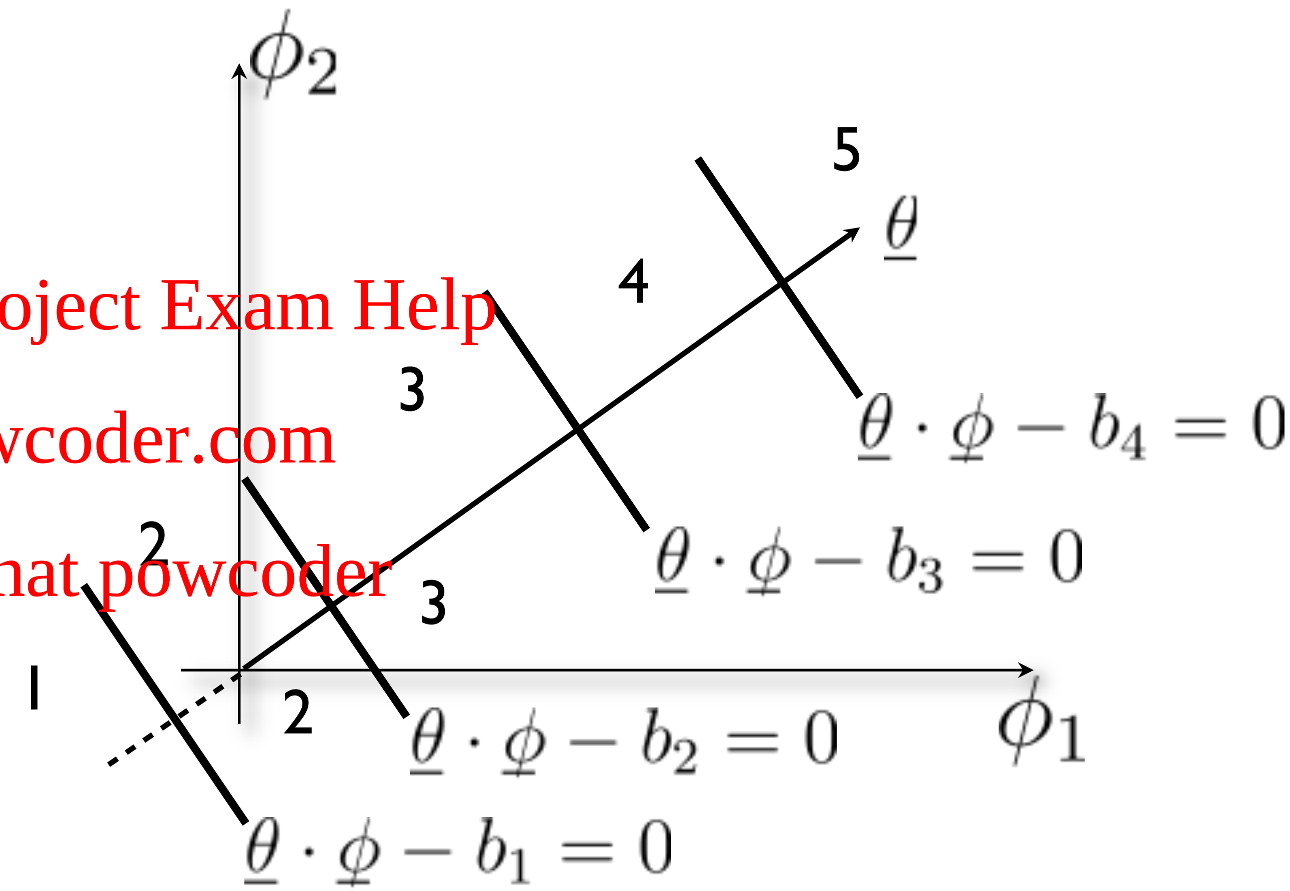
$$
\begin{array}{c c}
 & \text{binary tasks} \\
\begin{array}{c} \text{rating } y \end{array}
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} &
\begin{bmatrix}
-1 & -1 & -1 & -1 \\
1 & -1 & -1 & -1 \\
1 & 1 & -1 & -1 \\
1 & 1 & 1 & -1 \\
1 & 1 & 1 & 1
\end{bmatrix}
\end{array}
$$

The partitions are now dependent ... need to enforce consistency of the binary labels across partitions

Ordinal regression

- We can specify a set of classifiers with shared parameters that always produce consistent binary labels





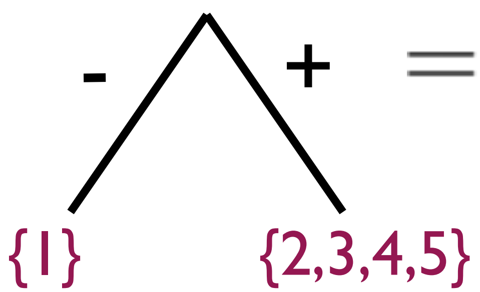
$$b_1 \leq b_2 \leq b_3 \leq b_4$$

Ordinal regression

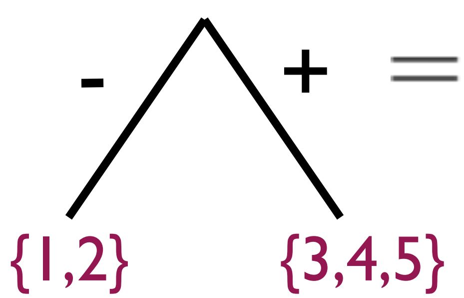
- We can specify a set of classifiers with shared parameters that always produce consistent binary labels

$$b_1 \leq b_2 \leq b_3 \leq b_4$$

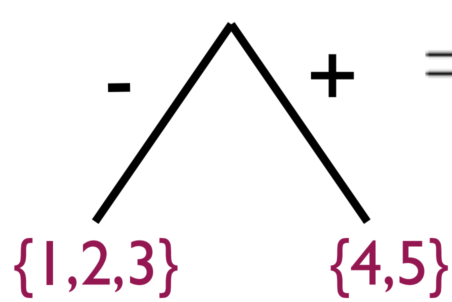
thresholds are different but ordered



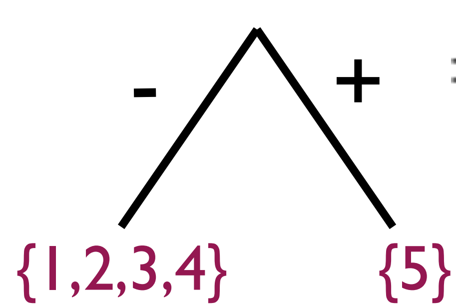
$$= s_1 = \mathrm{sign}(\,\underbrace{\underline{\theta} \cdot \underline{\phi}(x_i)}_{\text{common prediction}} - b_1\,)$$



$$= s_2 = \mathrm{sign}(\,\underline{\underline{\theta} \cdot \underline{\phi}(x_i)} - b_2\,)$$



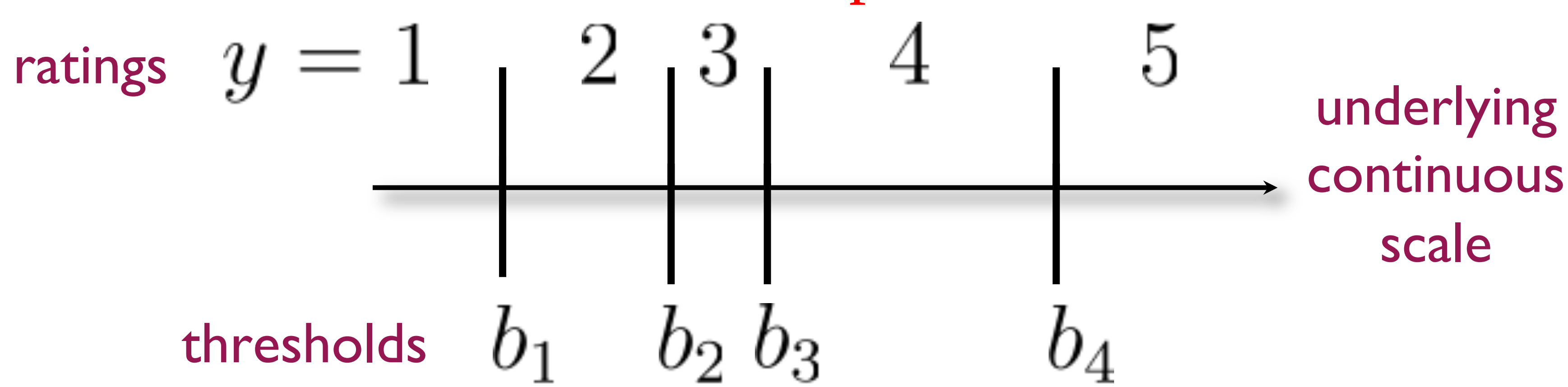
$$= s_3 = \mathrm{sign}(\,\underline{\underline{\theta} \cdot \underline{\phi}(x_i)} - b_3\,)$$



$$= s_4 = \mathrm{sign}(\,\underline{\underline{\theta} \cdot \underline{\phi}(x_i)} - b_4\,)$$

Ordinal regression, 2nd view

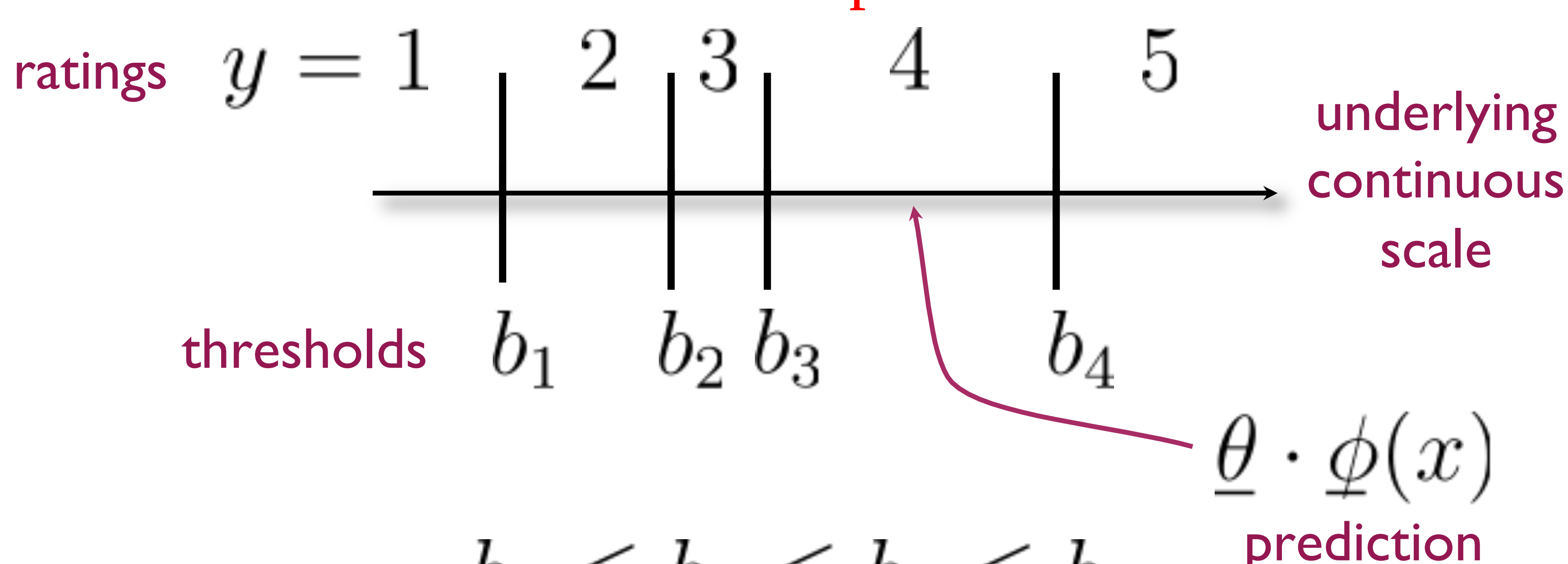
- Each item x_i is associated with a feature vector $\phi(x_i)$
  - e.g., product description, movie features, etc.
- We wish to predict an ordinal label $y_i \in \{1, \ldots, k\}$ for each item (reflecting views of one user)
- We assume that there exists an underlying continuous scale from which ratings are obtained via thresholding



$$b_1 \leq b_2 \leq b_3 \leq b_4$$

Ordinal regression, 2nd view

- Each item x_i is associated with a feature vector $\phi(x_i)$
  - e.g., product description, movie features, etc.
- We wish to predict an ordinal label $y_i \in \{1, \ldots, k\}$ for each item (reflecting views of one user)
- We assume that there exists an underlying continuous scale from which ratings are obtained via thresholding



$$b_1 \leq b_2 \leq b_3 \leq b_4$$

Ordinal regression, SVM style

- Given a training set $D = \{(x_i, y_i)\}_{i=1,\ldots,n}$

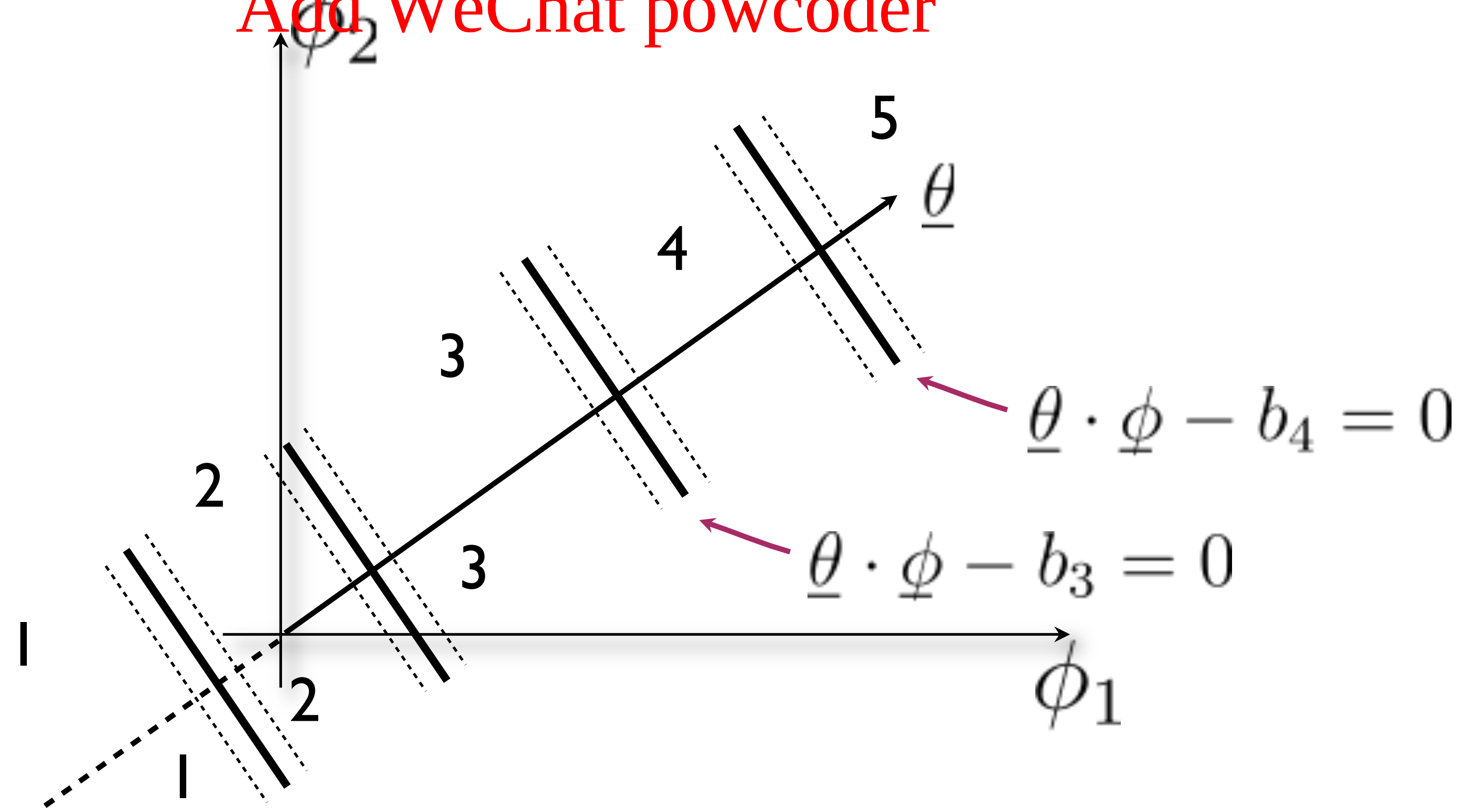
$$\text{minimize} \quad \frac{1}{2}\|\underline{\theta}\|^2 \quad \text{with respect to } \underline{\theta},\, b_1, \ldots, b_{k-1}$$

$$\text{such that } b_1 \leq b_2 \leq \ldots \leq b_{k-1} \text{ and}$$

$$\underbrace{s_{il}(\underline{\theta} \cdot \underline{\phi}(x_i) - b_l) \geq 1}_{\text{binary classification constraint}}, \quad l = 1, \ldots, k-1, \quad i = 1, \ldots, n$$

binary classification constraint

k-1 binary labels obtained from each observed rating



Ordinal regression, PRank

- We can also define a mistake driven perceptron algorithm for solving ordinal regression problems
- The updates are modified slightly due to shared parameters

cycle through the training set $i = 1, \ldots, n$

for each example i

$$E_i = \{ l : \; s_{il}(\underline{\theta} \cdot \underline{\phi}(x_i) - b_l) \leq 0 \}$$

identify all binary mistakes

$$\underline{\theta} \leftarrow \underline{\theta} + \Big(\sum_{l \in E_i} s_{il} \Big) \, \underline{\phi}(x_i)$$

perform a collective update based on the mistakes

$$b_l \leftarrow b_l - s_{il}, \; l \in E_i$$

update thresholds of each classifier

Note: having a threshold is equivalent to having an extra feature, in which all samples have -1. Thus, the update rule for b_l is not surprising.

Ordinal regression, PRank

- We can also define a mistake driven perceptron algorithm for solving ordinal regression problems
- The updates are modified slightly due to shared parameters

cycle through the training set $i = 1, \ldots, n$

for each example i

$$E_i = \{l : \; s_{il}(\underline{\theta} \cdot \underline{\phi}(x_i) - b_l) \leq 0\}$$

identify all binary mistakes

$$\underline{\theta} \leftarrow \underline{\theta} + \Big(\sum_{l \in E_i} s_{il}\Big)\, \underline{\phi}(x_i)$$

perform a collective update based on the mistakes

$$b_l \leftarrow b_l - s_{il}, \; l \in E_i$$

update thresholds of each classifier

- **Lemma**: if the thresholds are set to zero initially, they will maintain the correct ordering in the course of the algorithm

(See Lemma 1 in [1] if interested in the proof.)

PRank, mistake bound

- **Theorem:** Assume that there exists $\underline{\theta}^*, b_1^*, \dots, b_{k-1}^*$

$$\|\underline{\theta}^*\|^2 + \sum_{l=1}^{k-1} b_l^{*2} = 1$$

such that

$$s_{il}(\underline{\theta}^* \cdot \underline{\phi}(x_i) - b_l^*) \geq \gamma, \quad l = 1, \dots, k-1, \quad i = 1, \dots, n$$

then the algorithm makes at most

$$(k-1)\frac{R^2+1}{\gamma^2}$$

binary mistakes on the training set.

(See Theorem 2 in [1] if interested in the proof.)

Today's topics

- Multi-way classification
  - reducing multi-class to binary
  - margin based solution
- Rating (ordinal regression)
  - reduction to binary problems
  - SVM solution, on-line solution
- Ranking
  - ranking SVM

Ranking

- Rating products, movies, etc. using a few values (e.g., 1-5 stars) results in a partial ranking of the items
- Many rating / classification problems are better viewed as ranking problems
  - suggest movies in the order of user interest in them,
  - rank websites to display in response to a query,
  - suggest genes relevant to a particular disease condition, etc.
- By casting the learning problem as a ranking problem we can also incorporate other types of data / feedback
  - e.g., click through data from users

Ranking example

- We would like to rank n websites (find top sites to display) in response to a few query words

$$x = \text{context (set of query words)}$$

$$y = \text{website}$$

$$\begin{array}{ll}
(x_1, y_2) & (x_2, y_7) \\
(x_1, y_{10}) & (x_2, y_2) \\
(x_1, y_3) & (x_2, y_1) \\
\hline
\cdots & \cdots \\
(x_1, y_n) & (x_2, y_4)
\end{array}$$

$$x_1 = \{\text{ ranking applications }\} \qquad x_2 = \{\text{ ranking SVM code }\}$$

Ranking example

- We would like to rank n websites (find top sites to display) in response to a few query words

$$x = \text{context (set of query words)}$$
$$y = \text{website}$$

(x_1, y_2)		(x_2, y_7)	x
(x_1, y_{10})		(x_2, y_2)	
(x_1, y_3)	x	(x_2, y_1)	
$\cdots$		$\cdots$	
(x_1, y_n)		(x_2, y_4)	

$x_1 = \{ \text{ ranking applications } \}$ $\qquad$ $x_2 = \{ \text{ ranking SVM code } \}$

- The available data contain user selections (clicks) of websites out of those displayed to them

From selections to preferences

- We can interpret a user click as a statement that they prefer the selected link over others displayed in the context of the query

(x_1, y_2)	(x_2, y_7) **x**
(x_1, y_{10})	(x_2, y_2)
(x_1, y_3) **x**	(x_2, y_1)
$\cdots$	$\cdots$
(x_1, y_n)	(x_2, y_4)

$x_1 = \{ \text{ ranking applications } \}$ $\qquad$ $x_2 = \{ \text{ ranking SVM code } \}$

$\Downarrow$ $\qquad$ $\Downarrow$

$(x_1, y_3) > \{(x_1, y_{10}), (x_1, y_2)\}$ $\qquad$ $(x_2, y_7) > \{(x_2, y_2), (x_2, y_1)\}$

Ranking function

- Our goal is to estimate a ranking function over pairs f(x,y) such that its values are consistent with the observed preferences.

$$(x_2, y_7) > \{(x_2, y_2), (x_2, y_1)\}$$

$$\Rightarrow \quad f(x_2, y_7) > f(x_2, y_2), \quad f(x_2, y_7) > f(x_2, y_1)$$

- We can parameterize this function in terms of feature vectors extracted from each pair (context,website)

$$f(x, y; \underline{\theta}) = \underline{\theta} \cdot \underline{\phi}(x, y)$$

Ranking function

- Our goal is to estimate a ranking function over pairs f(x,y) such that its values are consistent with the observed preferences.

$$(x_2, y_7) > \{(x_2, y_2), (x_2, y_1)\}$$

$$\Rightarrow \quad f(x_2, y_7) > f(x_2, y_2), \quad f(x_2, y_7) > f(x_2, y_1)$$

- We can parameterize this function in terms of feature vectors extracted from each pair (context,website)

$$f(x, y; \underline{\theta}) = \underline{\theta} \cdot \underline{\phi}(x, y)$$

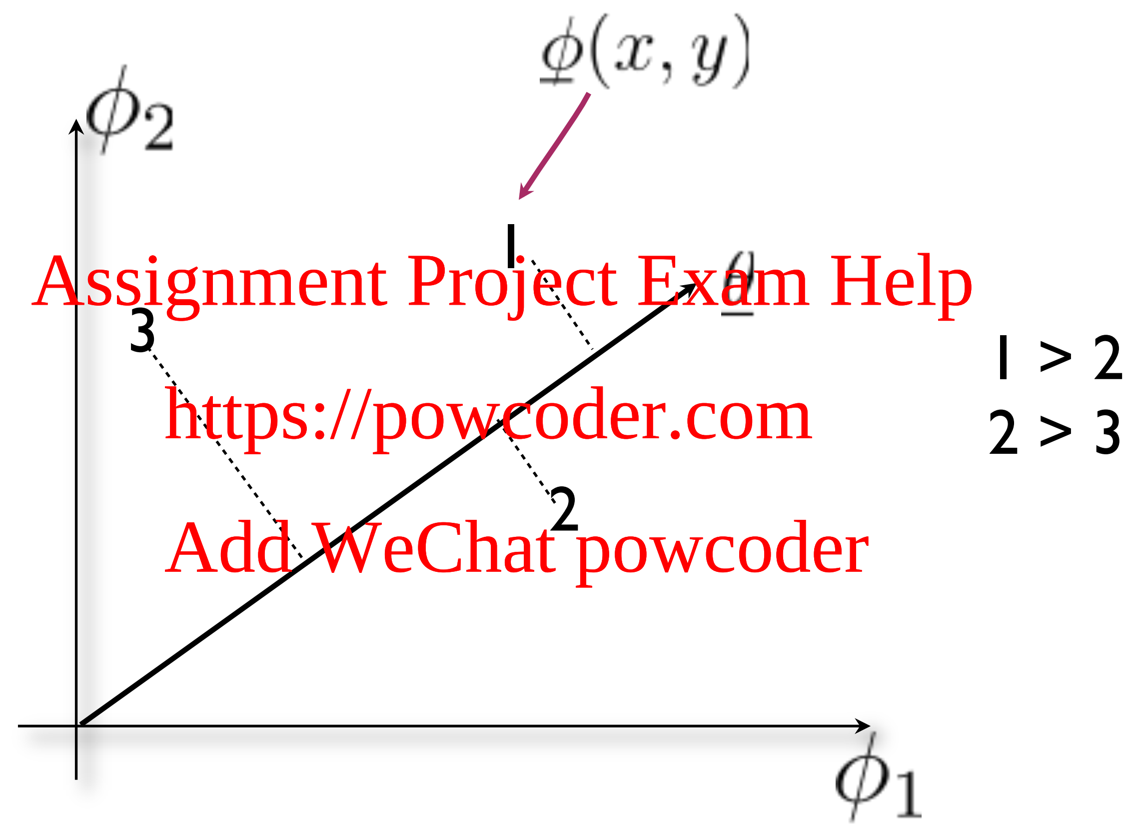
where the features could be, e.g.,

$$\phi_w(x, y) = \left\{ \begin{array}{ll} 1, & \text{if word } w \text{ appears in } x \text{ and } y \\ 0, & \text{otherwise} \end{array} \right\}$$

for all $w \in \mathcal{W}$

Ranking function

- The ranking function gives rise to a total ordering of the pairs via projection to the parameter vector



$$f(x, y; \underline{\theta}) = \underline{\theta} \cdot \underline{\phi}(x, y)$$

SVM rank

- A training set of order relations between pairs

$$D = \{ \{(x_i, y_j) > (x_k, y_l)\} \}$$

- An SVM style algorithm for finding a consistent ranking function

$$\text{minimize} \quad \frac{1}{2}||\underline{\theta}||^2 \quad \text{with respect to } \underline{\theta} \text{ such that}$$

$$\underline{\theta} \cdot \underline{\phi}(x_i, y_j) \geq \underline{\theta} \cdot \underline{\phi}(x_k, y_l) + 1,$$
$$\forall \{(x_i, y_j) > (x_k, y_l)\} \text{ in } D$$

SVM rank

- A training set of order relations between pairs

$$D = \{\, \{(x_i, y_j) > (x_k, y_l)\} \,\}$$

- An SVM style algorithm for finding a consistent ranking function

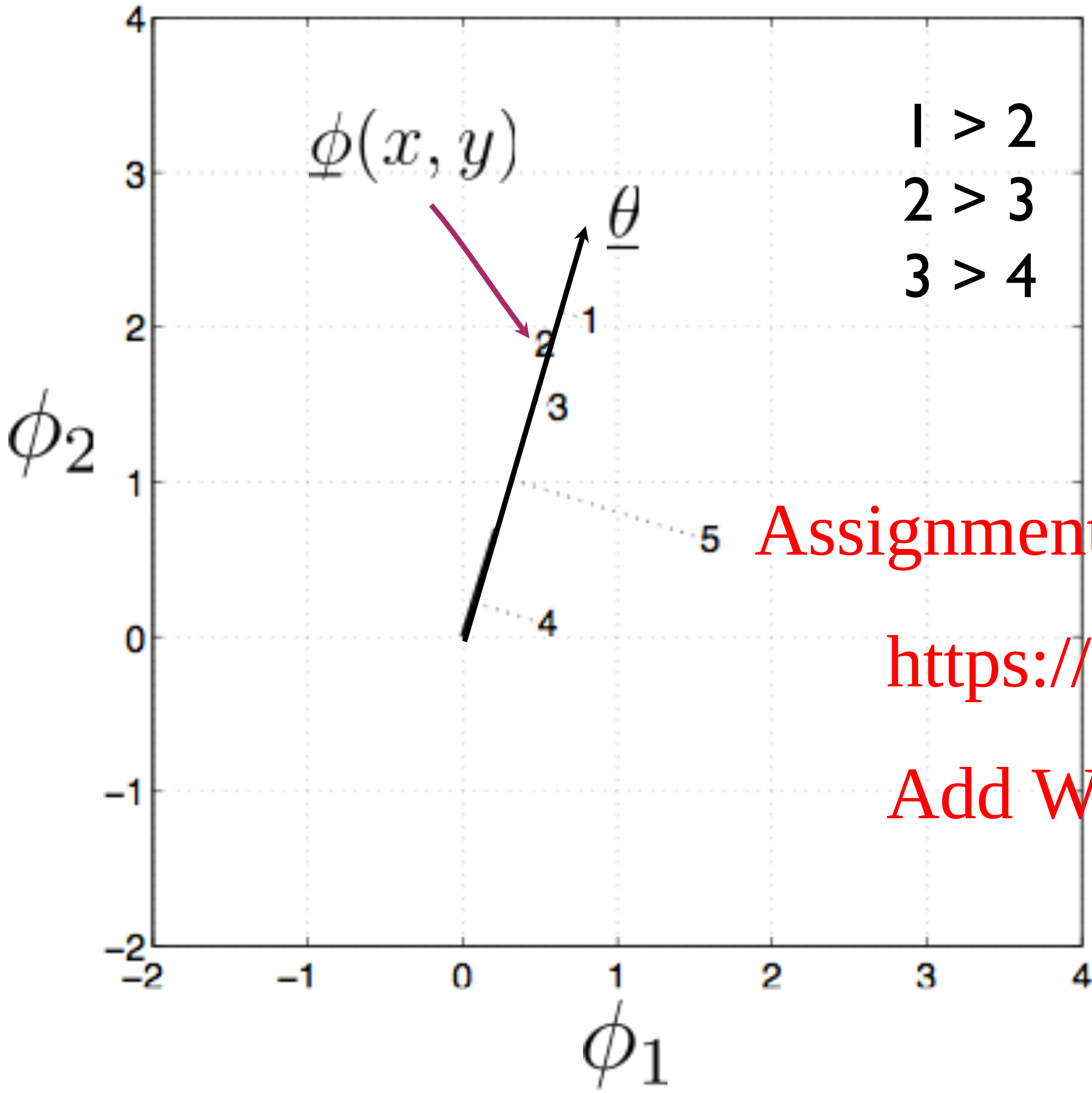
$$\text{minimize} \quad \frac{1}{2}\|\underline{\theta}\|^2 + C \sum_{ij;kl} \xi_{ij;kl} \quad \text{subject to}$$

$$\underline{\theta} \cdot \underline{\phi}(x_i, y_j) \geq \underline{\theta} \cdot \underline{\phi}(x_k, y_l) + 1 - \xi_{ij;kl}, \quad \xi_{ij;kl} \geq 0$$

$$\forall \{(x_i, y_j) > (x_k, y_l)\} \text{ in } D$$

- It is important to appropriately weight or choose which constraints to include

The effect of ranking constraints



adding a single constraint can have a large effect on the ranking solution