CPSC 340:
Machine Learning and Data Mining

Logistic Regression

Fall 2015

Admin

- Assignment 3 due Friday:
  - Submit as a single PDF file.

Features with Different Scales

- Consider features with different scales:

Egg (#)	Milk (mL)	Fish (g)	Pasta (cups)
0	250	0	1
1	250	200	1
0	0	0	0.5
2	250	150	0

- Should we convert to some standard ‘unit’?
  – For decision trees, it doesn’t matter:
    - (Milk > 100 mL) and (Milk > 0.1 L) will give the same rule.
  – For k-nearest neighbours, it matters:
    - Distance to (Milk = 100, Eggs = 1) is different than distance to (Milk = 0.1, Eggs = 1).

Features with Different Scales

- Consider features with different scales:

Egg (#)	Milk (mL)	Fish (g)	Pasta (cups)
0	250	0	1
1	250	200	1
0	0	0	0.5
2	250	150	0

- Should we convert to some standard ‘unit’?
  - For unregularized linear regression, it doesn’t matter:
    - w_j*(100 mL) gives the same model as w_j*(0.1 L), w_j will just be 1000 times smaller.
  - With regularization, it does matter:
    - Penalization $|w_j|$ means different things if features ‘j’ are on different scales.

Standardizing Features

- To put features on a similar scale, it is common to 'standardize':
  - For each feature:
    - Compute mean and standard deviation:

$$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij} \qquad \sigma_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \mu_j)^2}$$

    - Subtract mean and divide by standard deviation:

$$x_{ij} \leftarrow \frac{x_{ij} - \mu_j}{\sigma_j}$$

  - Change in 'w_j' have similar effect for any feature 'j'.
- Related issue is the 'bias' (y-intercept) variable:
  - Often, we do not regularize the 'bias' (or use small λ).
  - Avoids penalizing global shift up or down.

Standardizing Target

- In regression, we also often standardize the targets y_i.
  - Puts targets on the same standard scale as standardized features:

$$y_i \leftarrow \frac{y_i - \mu_y}{\sigma_y}$$

- With standardized target, choosing no features predicts average y_i:
  - Making features non-zero must then do better than this.

- Another common transformation of y_i is logarithm/exponent:

$$y_i \leftarrow \log(y_i) \qquad y_i \leftarrow \exp(\gamma y)$$

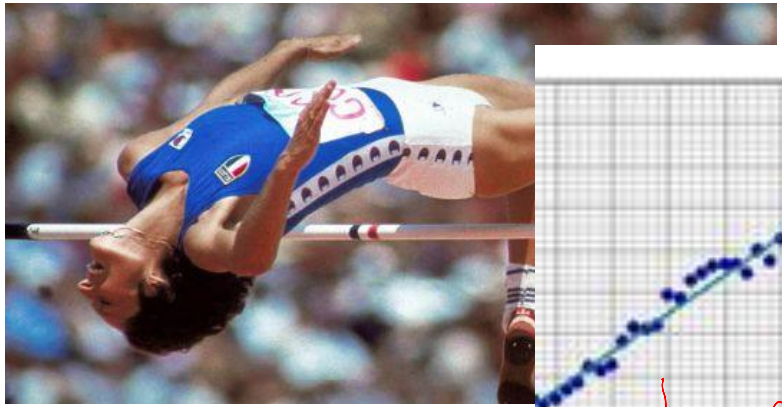
  - Makes sense for geometric/exponential processes.

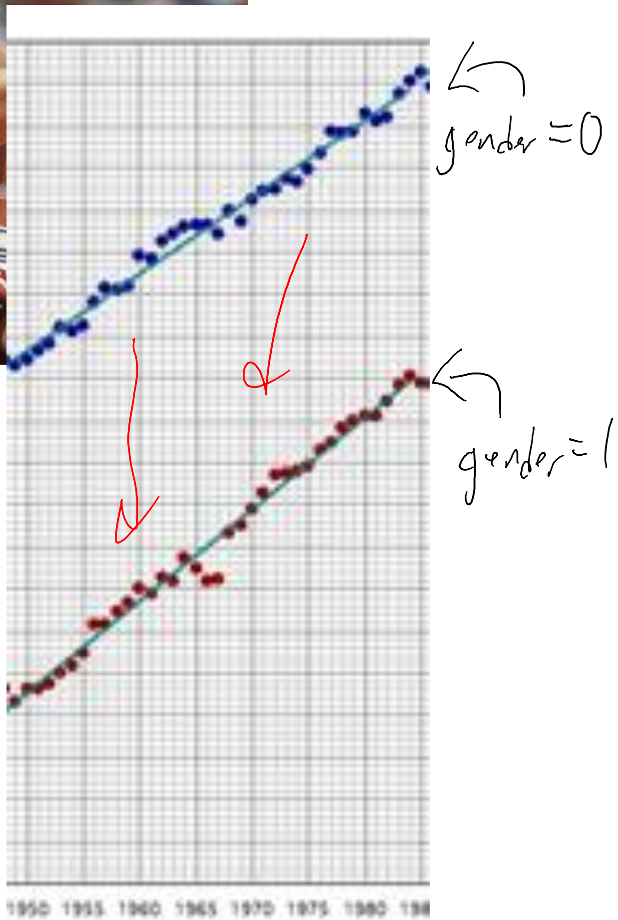
Regression with Binary Features

- What is the effect of a binary feature on linear regression?

Year	Gender
1975	1
1975	0
1980	1
1980	0

Height
1.85
2.25
1.95
2.30





- Adding a bias w_0, with this representation we have:

$$\text{height} = w_0 + w_1(\text{year}) + w_2(\text{gender}).$$

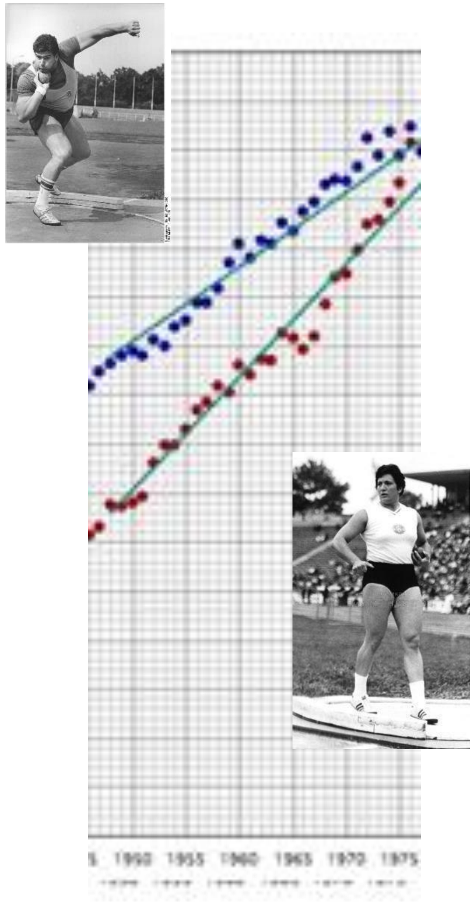
- The 'gender' variable causes a shift up/down:

If gender = 0, $\text{height} = w_0 + w_1(\text{year})$

If gender = 1, $\text{height} = w_0 + w_1(\text{year}) + w_2$

http://www.medalinframe.com/athletes/sara-simeoni/
http://www.at-a-lanta.nl/weia/Progressie.html

Regression with Binary Features



- What if different genders have different slopes?
  - Use a gender-specific slope.

Year	Gender
1975	1
1975	0
1980	1
1980	0

$\Rightarrow$

Bias (gender = 1)	Year (gender = 1)	Bias (gender = 0)	Year (gender = 0)
1	1975	0	0
0	0	1	1975
1	1980	0	0
0	0	1	1980

$$\text{distance} = w_0 + w_1(\text{year}) \quad (\text{if gender} = 1)$$

$$\text{distance} = w_3 + w_4(\text{year}) \quad (\text{if gender} = 0)$$

separate bias

separate slope

http://www.at-a-lanta.nl/weia/Progressie.html
http://www.wikiwand.com/it/Udo_Beyer
http://women-s-rights.blogspot.ca/

Regression with Binary Features

- This trick just fits separate 'local' variable for each gender.
- To share information across genders, include a 'global' version.

Year	Gender
1975	1
1975	0
1980	1
1980	0

Year	Year (if gender = 1)	Year (if gender = 0)
1975	1975	0
1975	0	1975
1980	1980	0
1980	0	1980

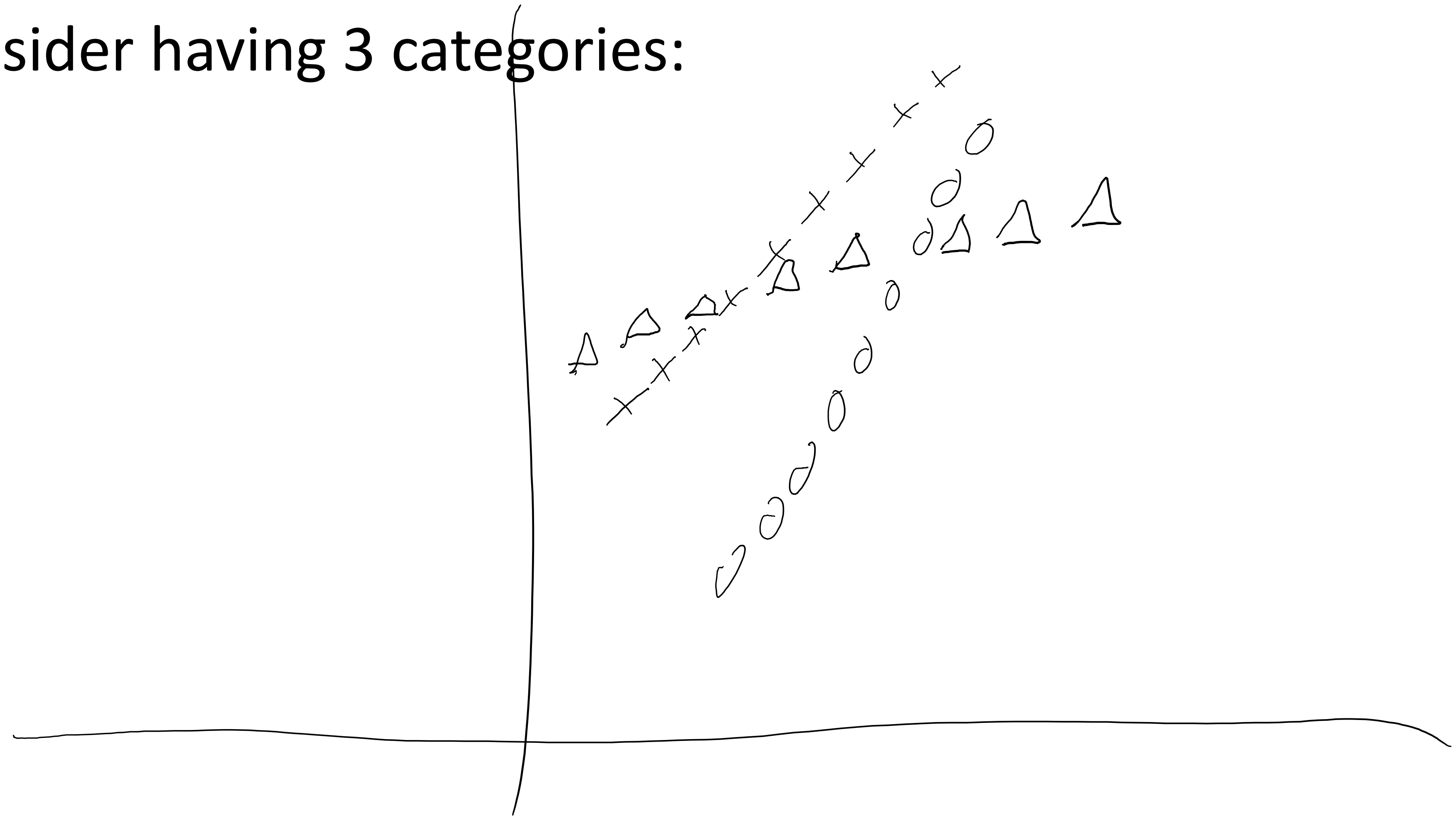
- 'Global' year feature: influence of time on both genders.
  - E.g., improvements in technique.
- 'Local' year feature: gender-specific deviation from global trend.
  - E.g., different effects of performance-enhancing drugs.

'global' 'local'

$$y_i = w_0 + w_g(\text{year}) + w_\ell(\text{year})$$

Regression with Binary Features

- Consider having 3 categories:

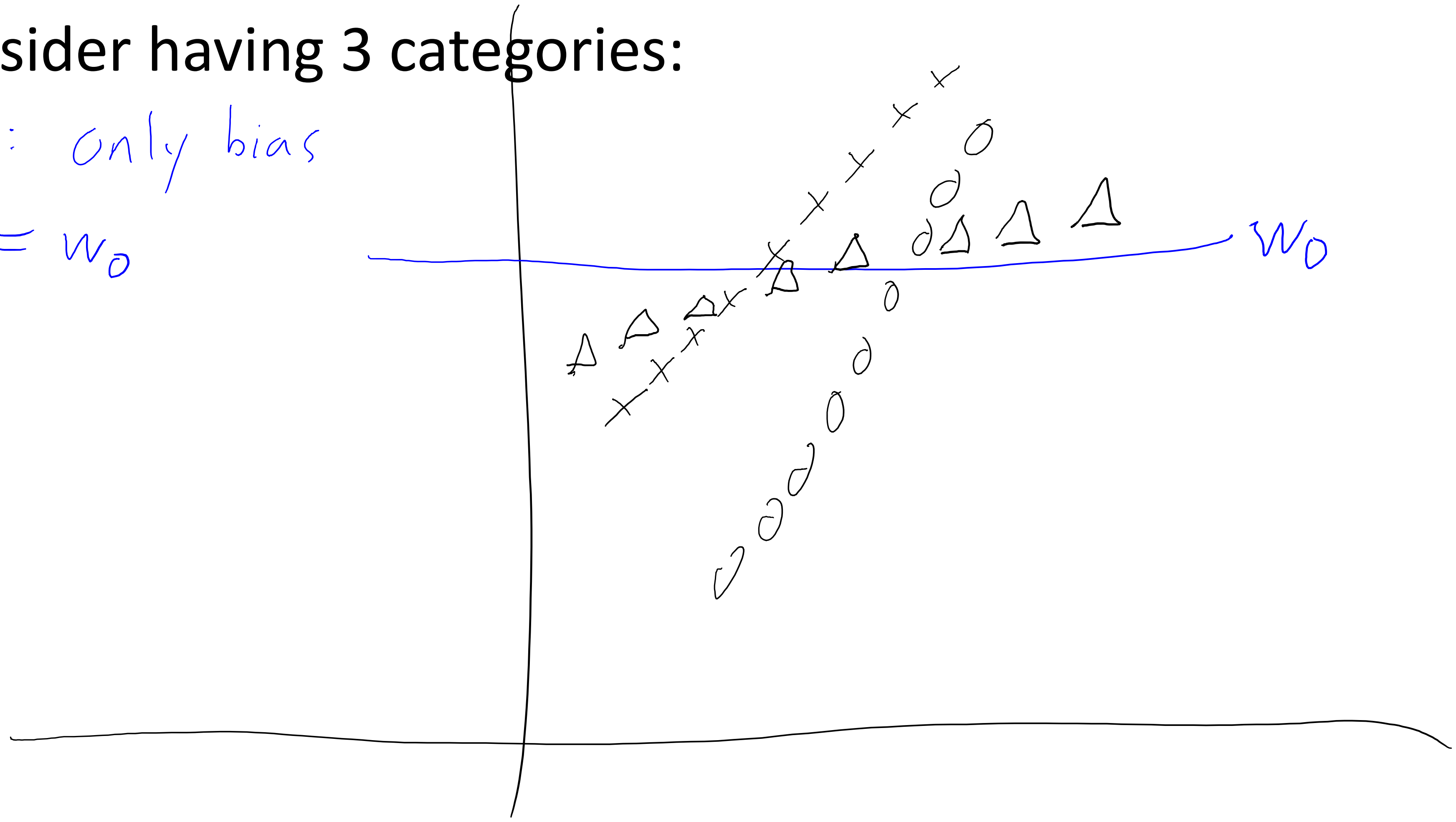


Regression with Binary Features

- Consider having 3 categories:

Model 1: only bias

$y = w_0$

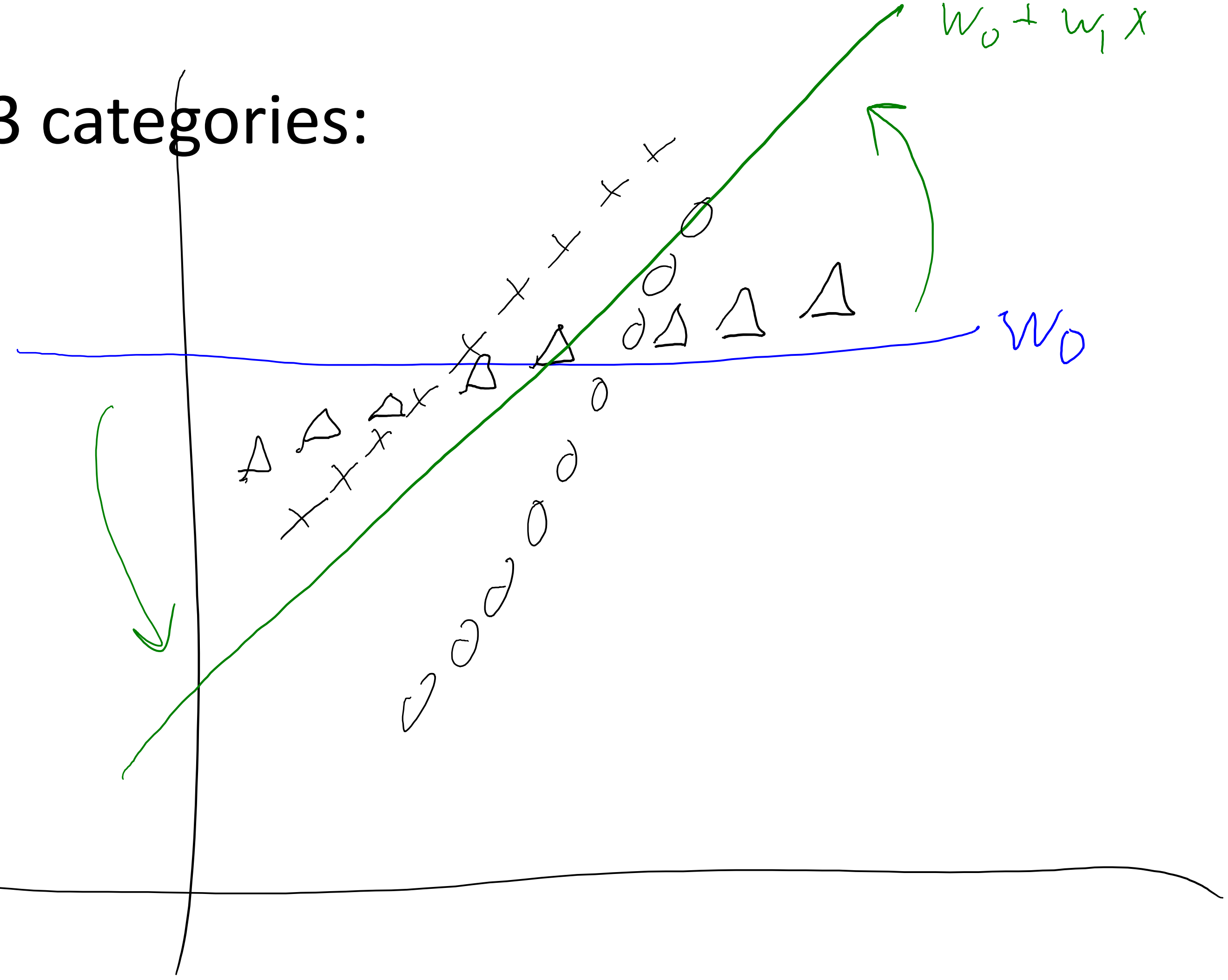


Regression with Binary Features

- Consider having 3 categories:

Model 2: bias + slope

$y = w_0 + w_1 x$



Regression with Binary Features

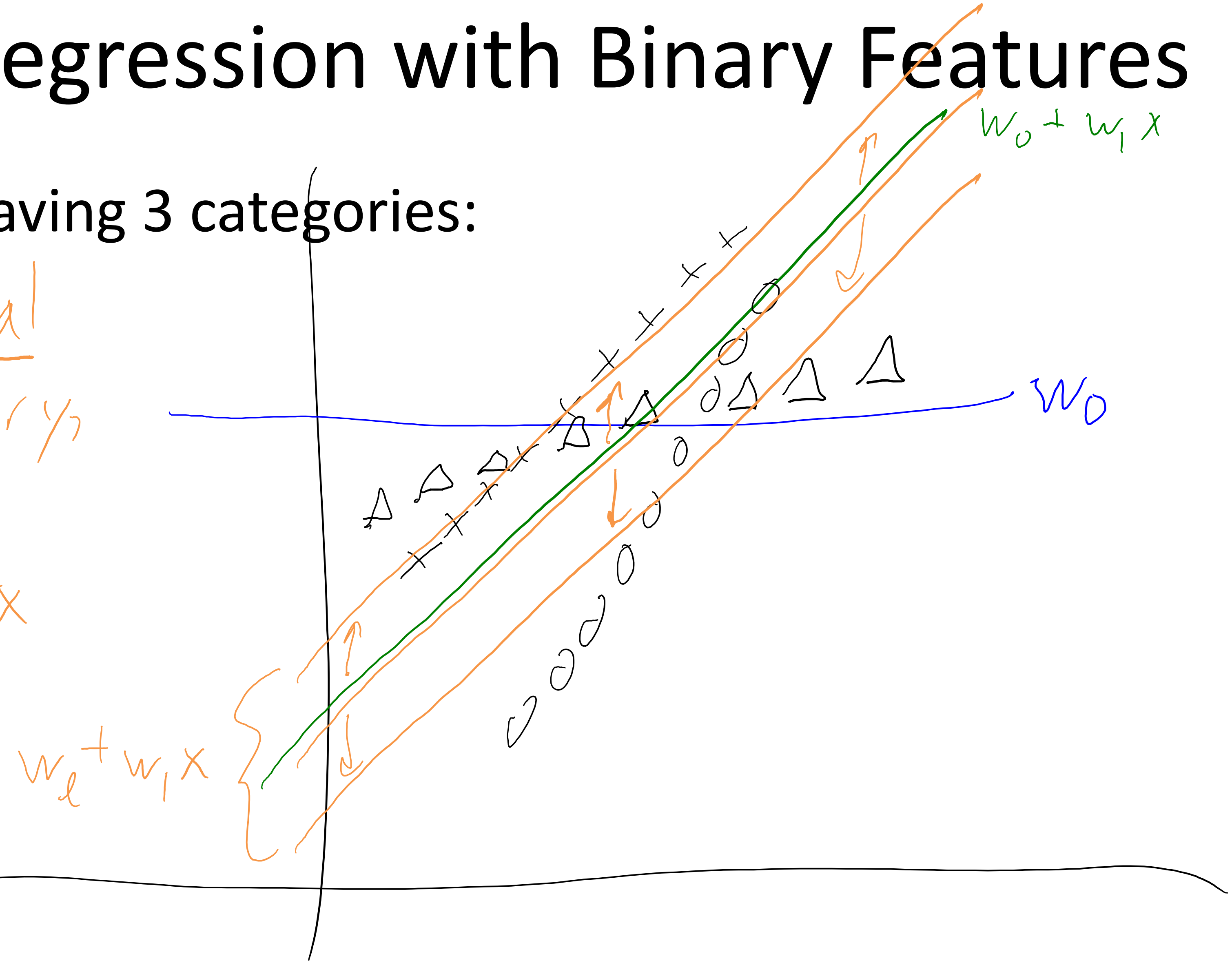
- Consider having 3 categories:

Model 3: <u>local</u>

bias for category, global slope.

$y = w_l + w_1 x$

"local"

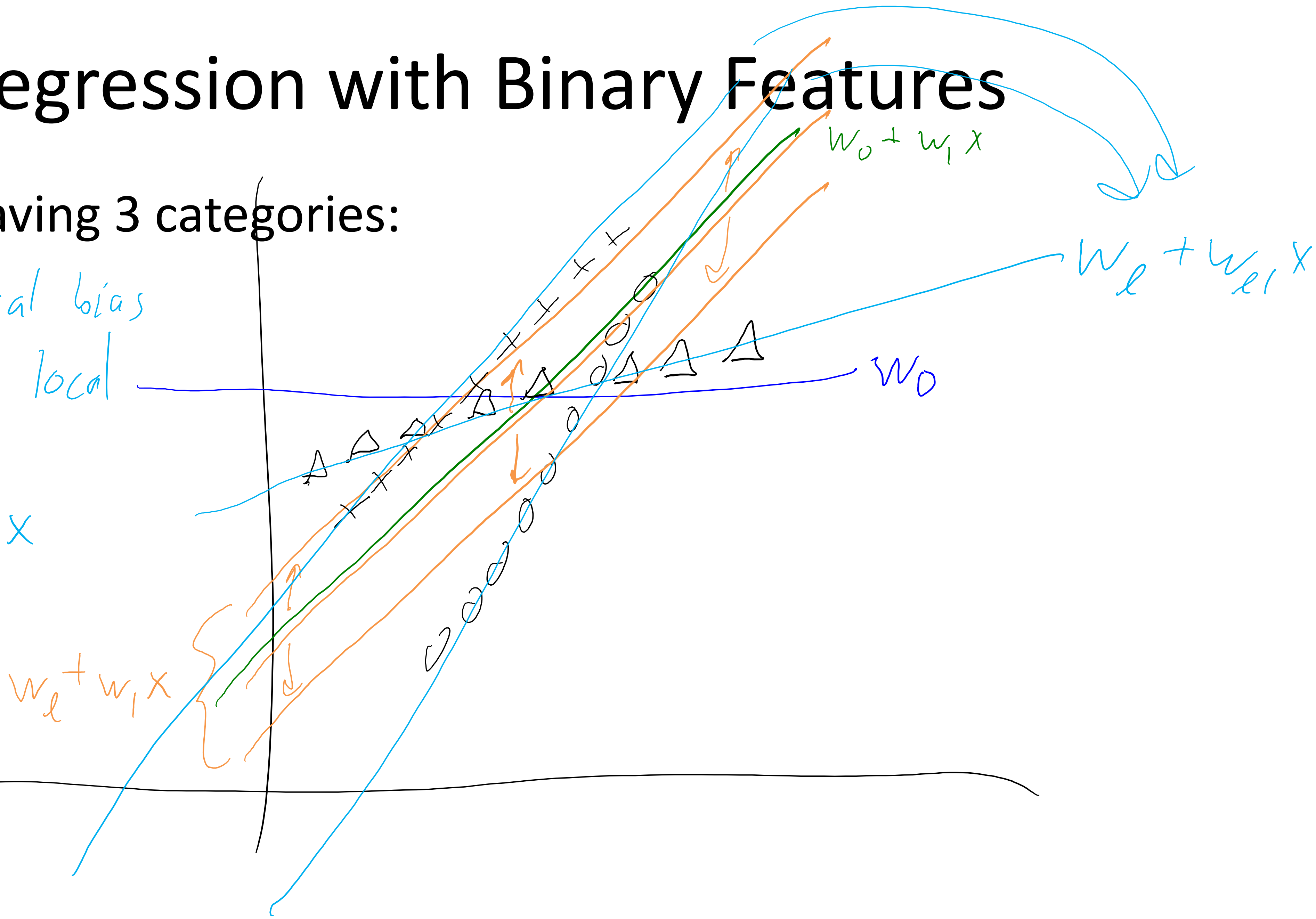


Regression with Binary Features

- Consider having 3 categories:

Model 4: local bias for category, local slope

$y = w_l + w_{l1} x$

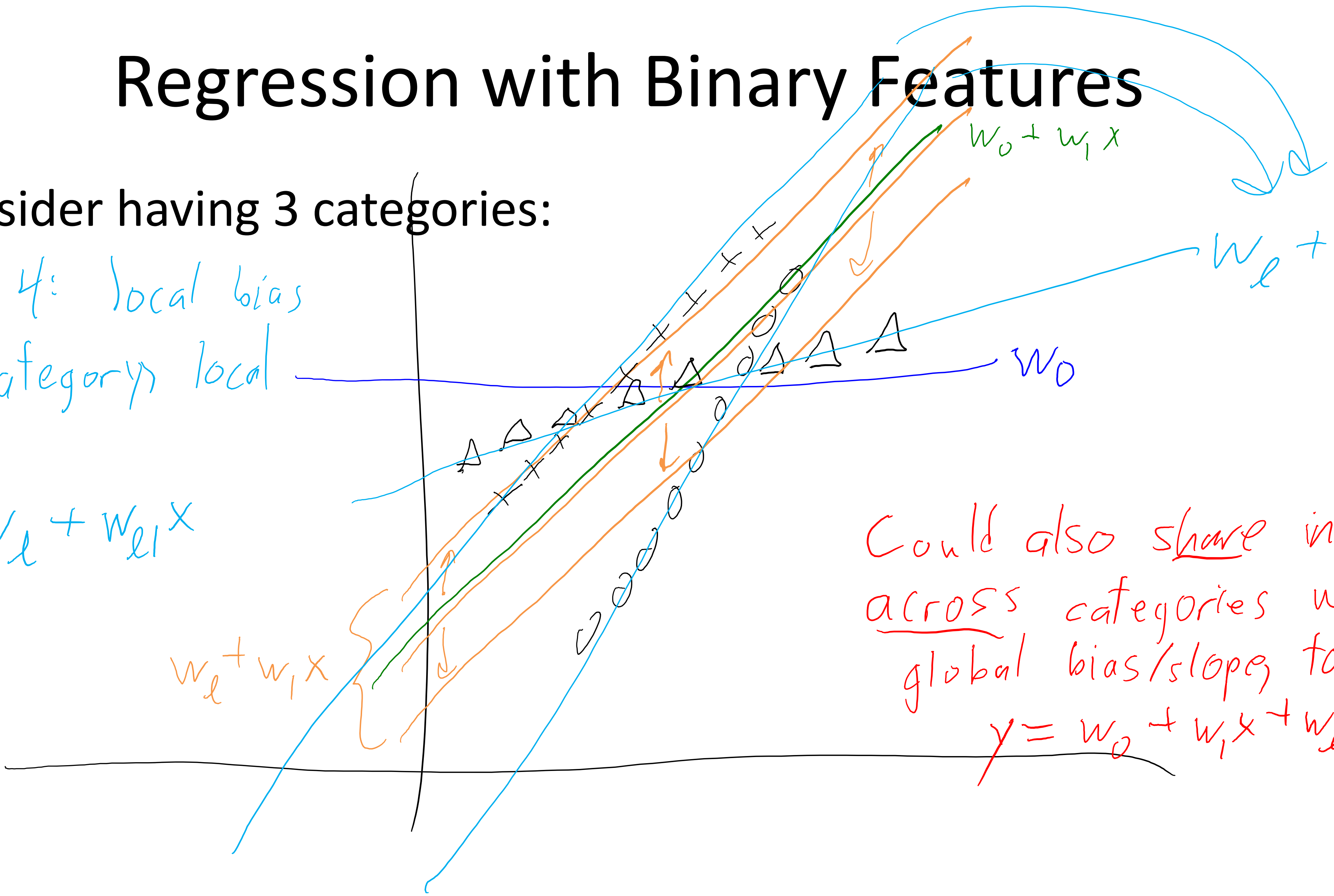


Regression with Binary Features

- Consider having 3 categories:

Model 4: local bias for category, local slope

$y = w_\ell + w_{\ell 1} x$

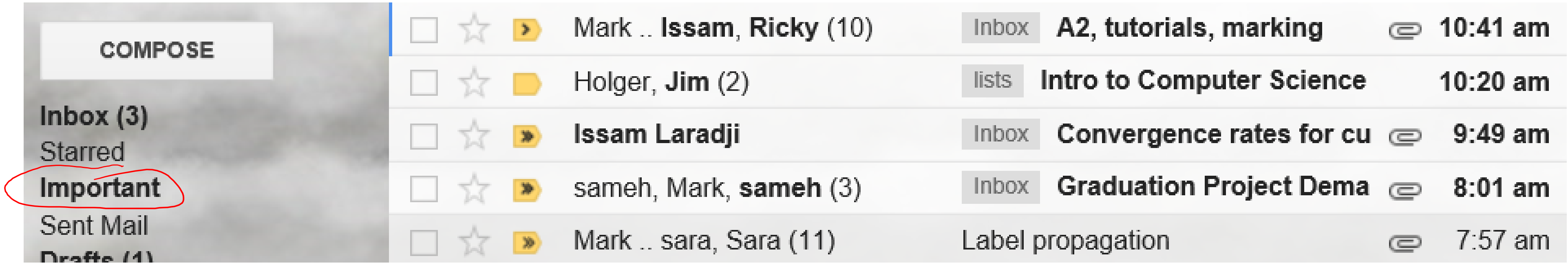


Could also share information across categories with global bias/slope too:

$y = w_0 + w_1 x + w_\ell + w_{\ell 1} x$

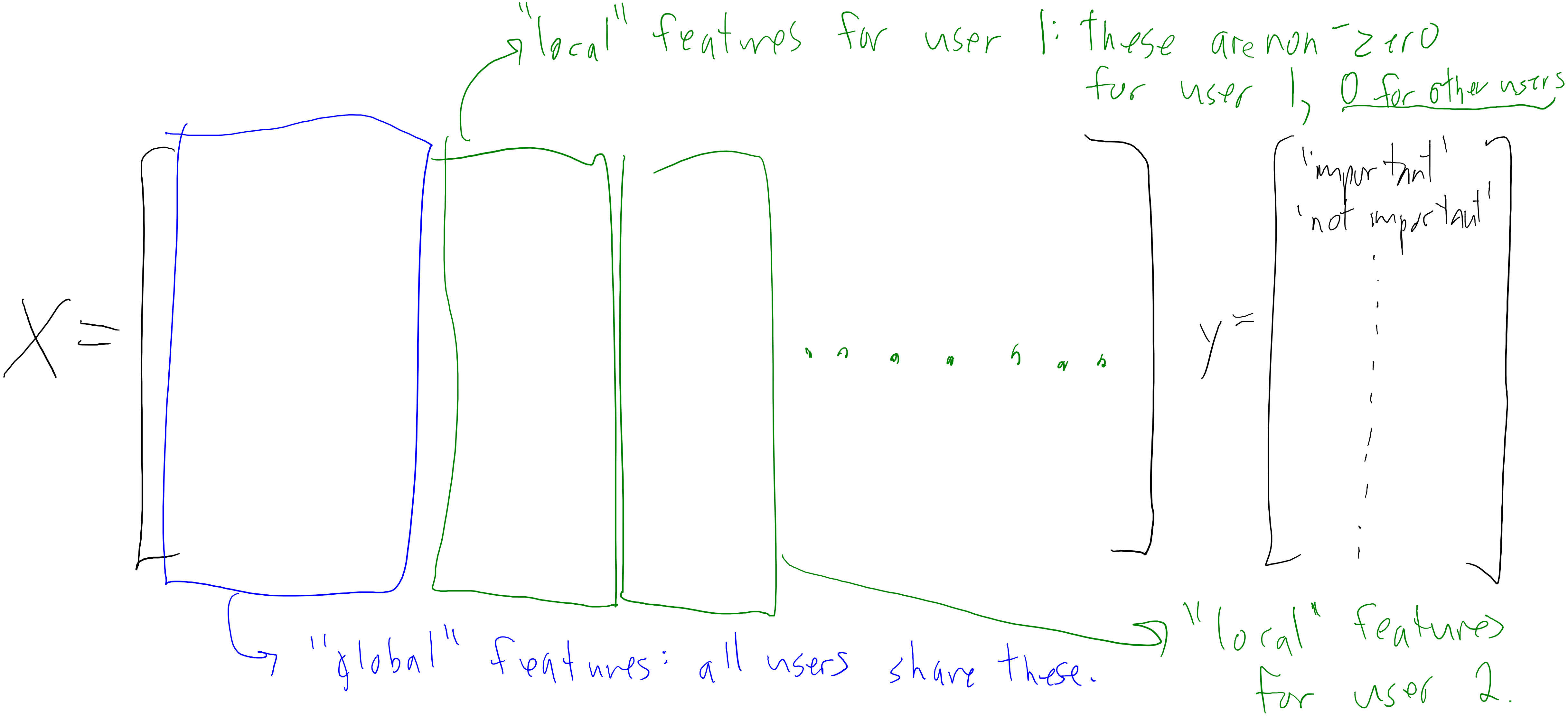
Motivation: Identifying Important E-mails

- How can we automatically identify 'important' e-mails?



- We have a big collection of e-mails:
  - Mark as 'important' if user takes some action based on them.
- There might be some 'universally' important messages:
  - "This is your mother, something terrible happened, give me a call ASAP."
- But your "important" message may be unimportant to others.
  - Similar for spam: "spam" for one user could be "not spam" for another.

The Big Global/Local Feature Table



Predicting Importance of E-mail For New User

- Consider a new user:
  - Start out with no information about them.
  - Use global features to predict what is important to generic user.
- As we collect data about the user, we update local features:
  - The local features let us give *personalized prediction* of importance.
  - User might not agree with global importance, or have specialized interests.

- Classification with logistic regression (variant of linear regression):
  - With large datasets, almost always better than naïve Bayes.

Classification Using Regression?

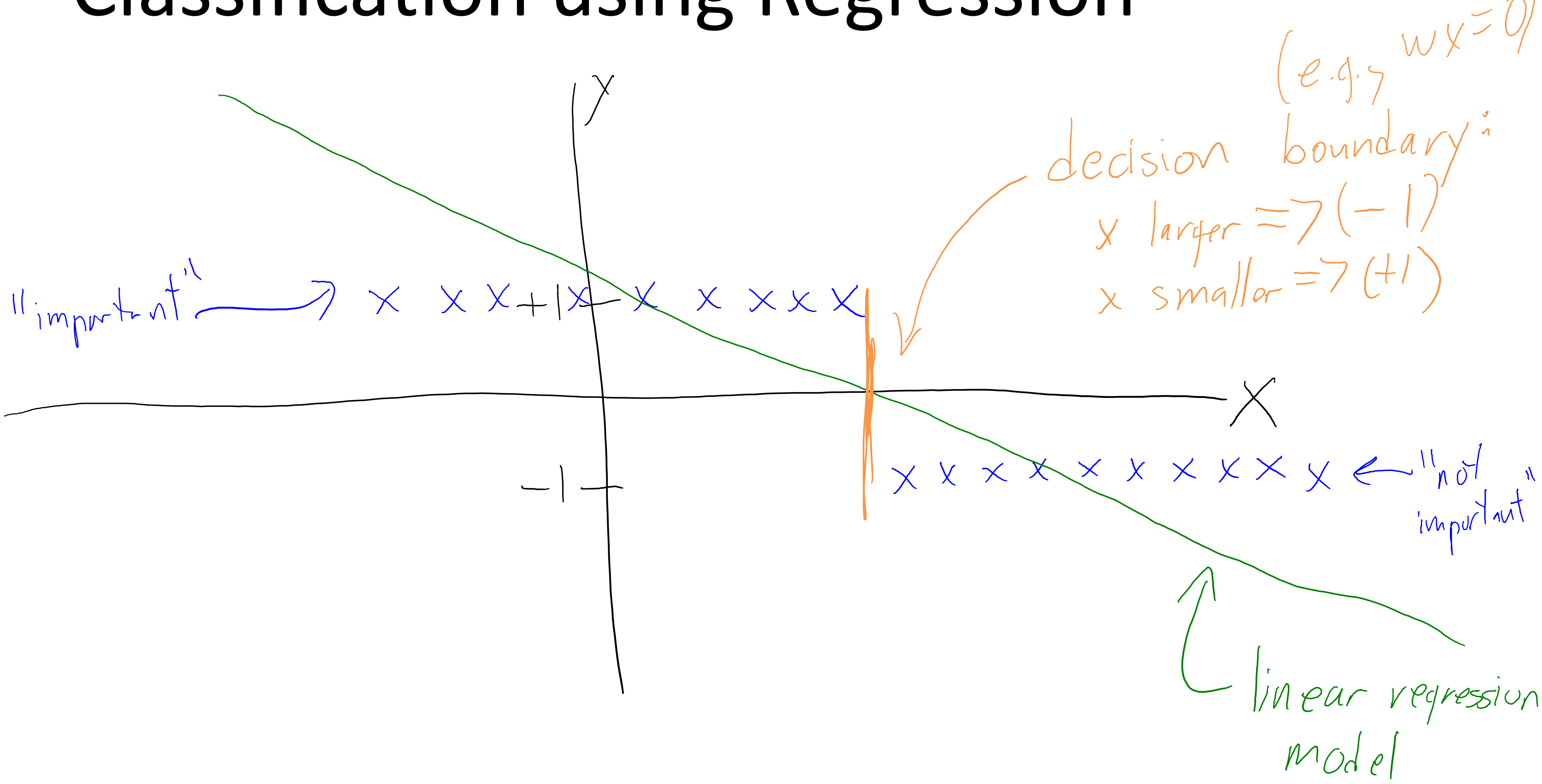
- Usual approach to do classification with regression:
  - Code y_i as '-1' for one class and '+1' for the other class.
  - E.g., '+1' means 'important' and '-1' means 'not important'.
- Fit a linear regression model:

$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id}$$
$$= w^T x_i$$

- To classify, take the sign (i.e., closer '-1' or '+1'?):

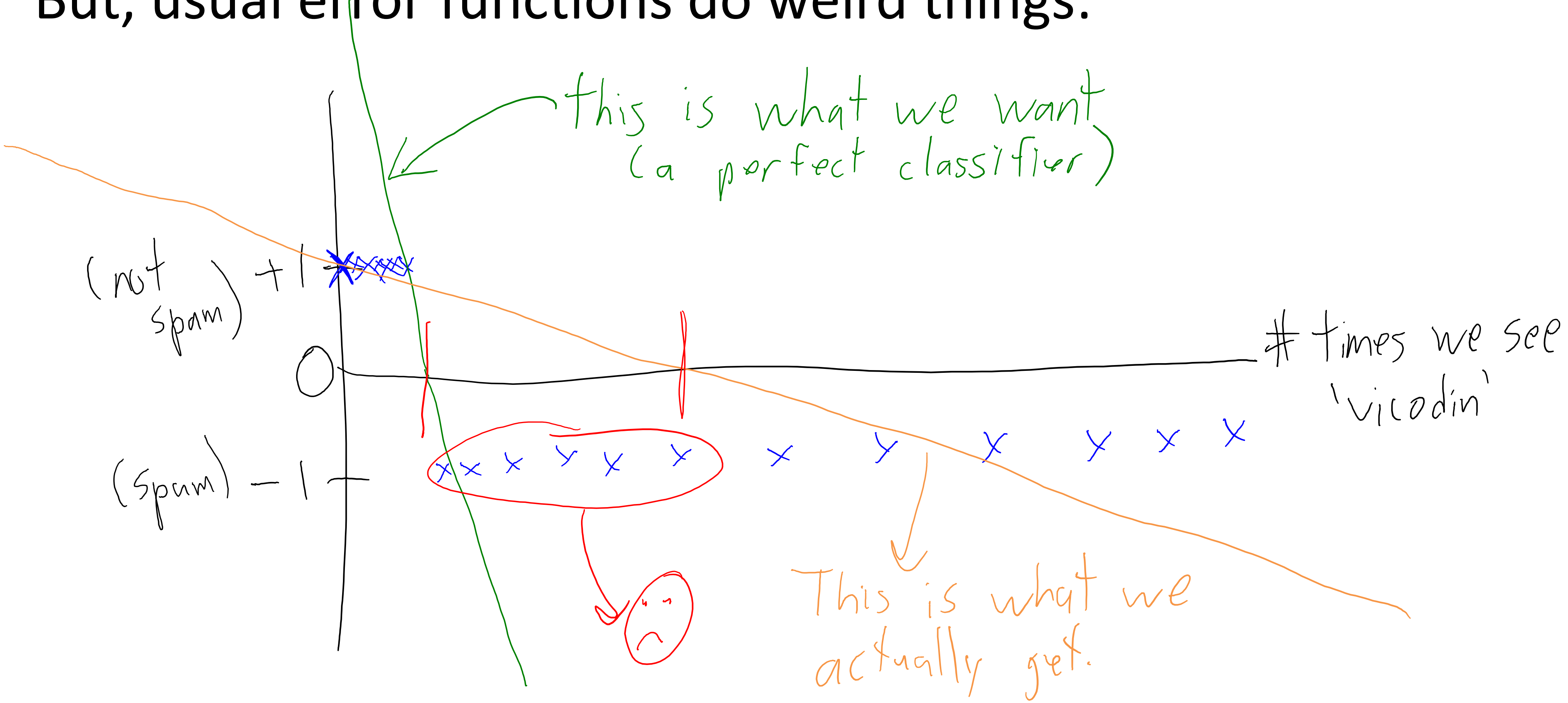
$$y_i = \text{sign}(w^T x).$$

Classification using Regression



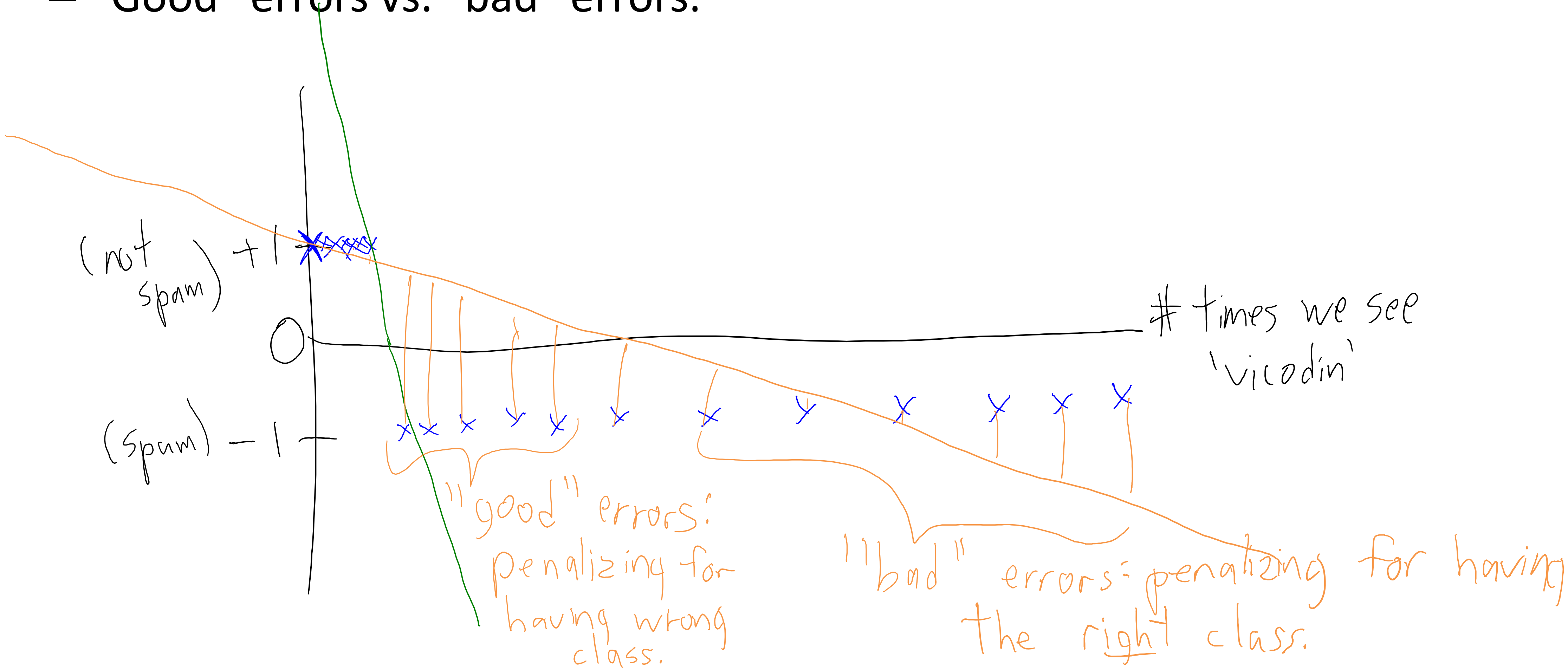
Classification Using Regression

- Can use our regression tricks (e.g., regularization) for classification.
- But, usual error functions do weird things:



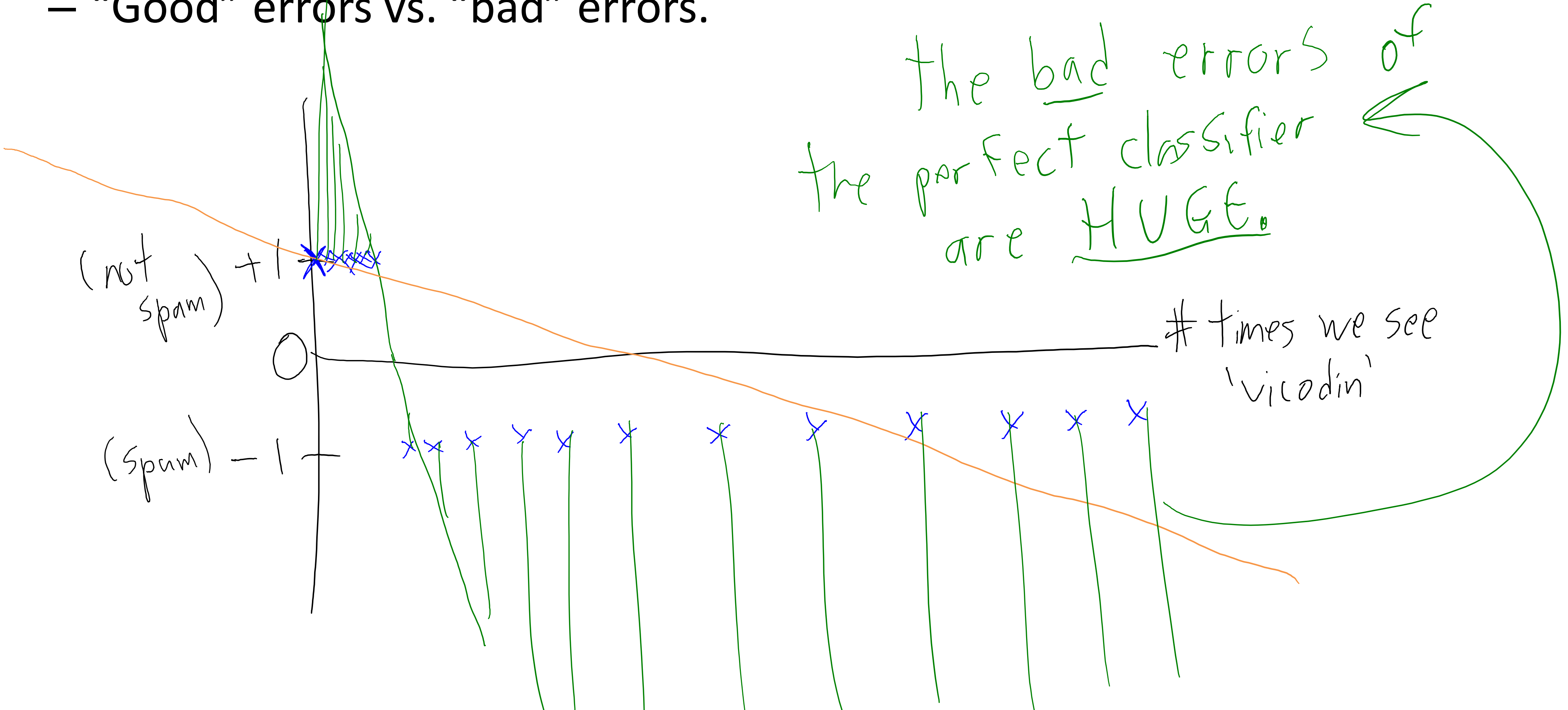
Classification Using Regression

- What went wrong?
  - "Good" errors vs. "bad" errors.

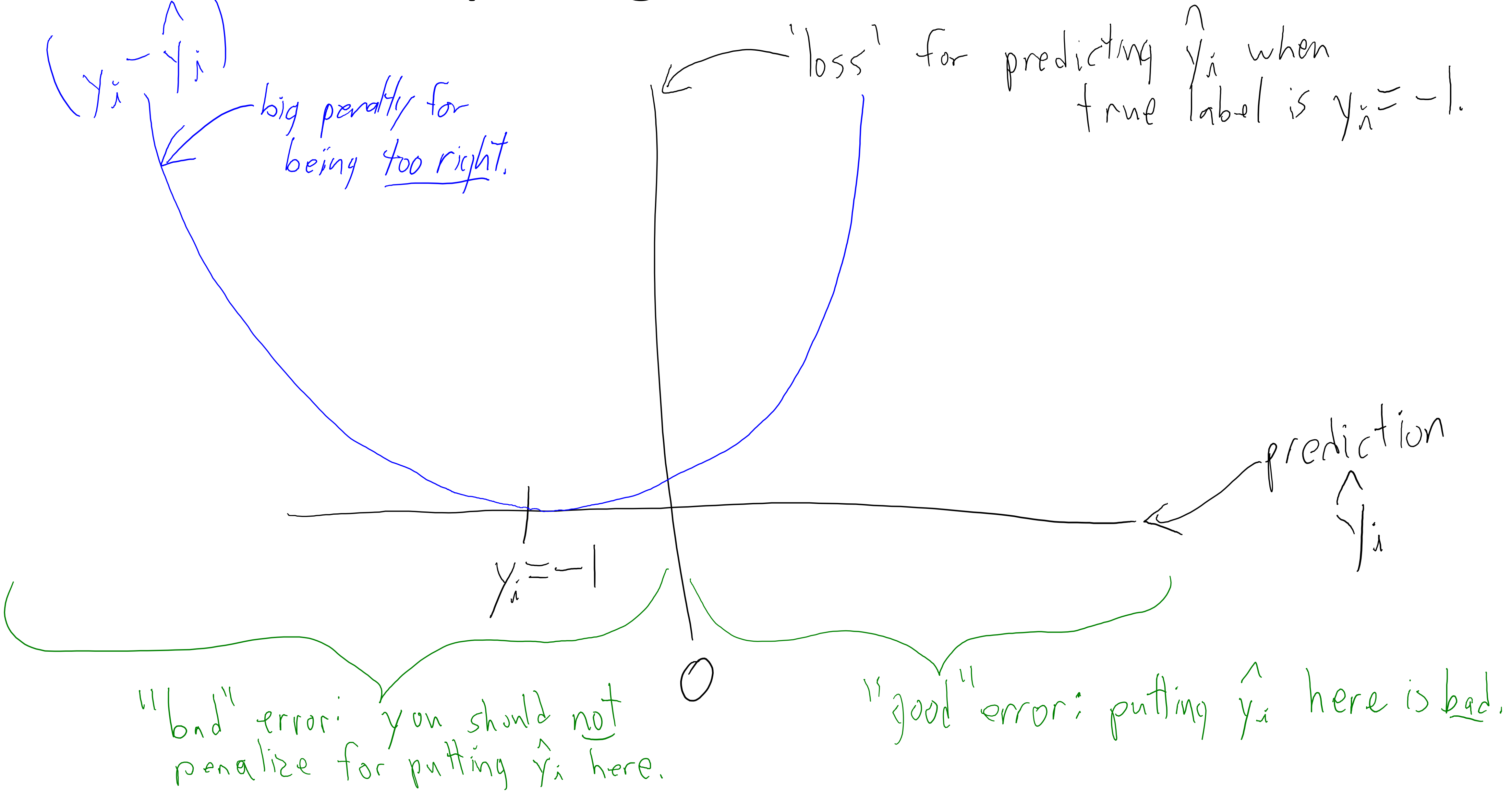


Classification Using Regression

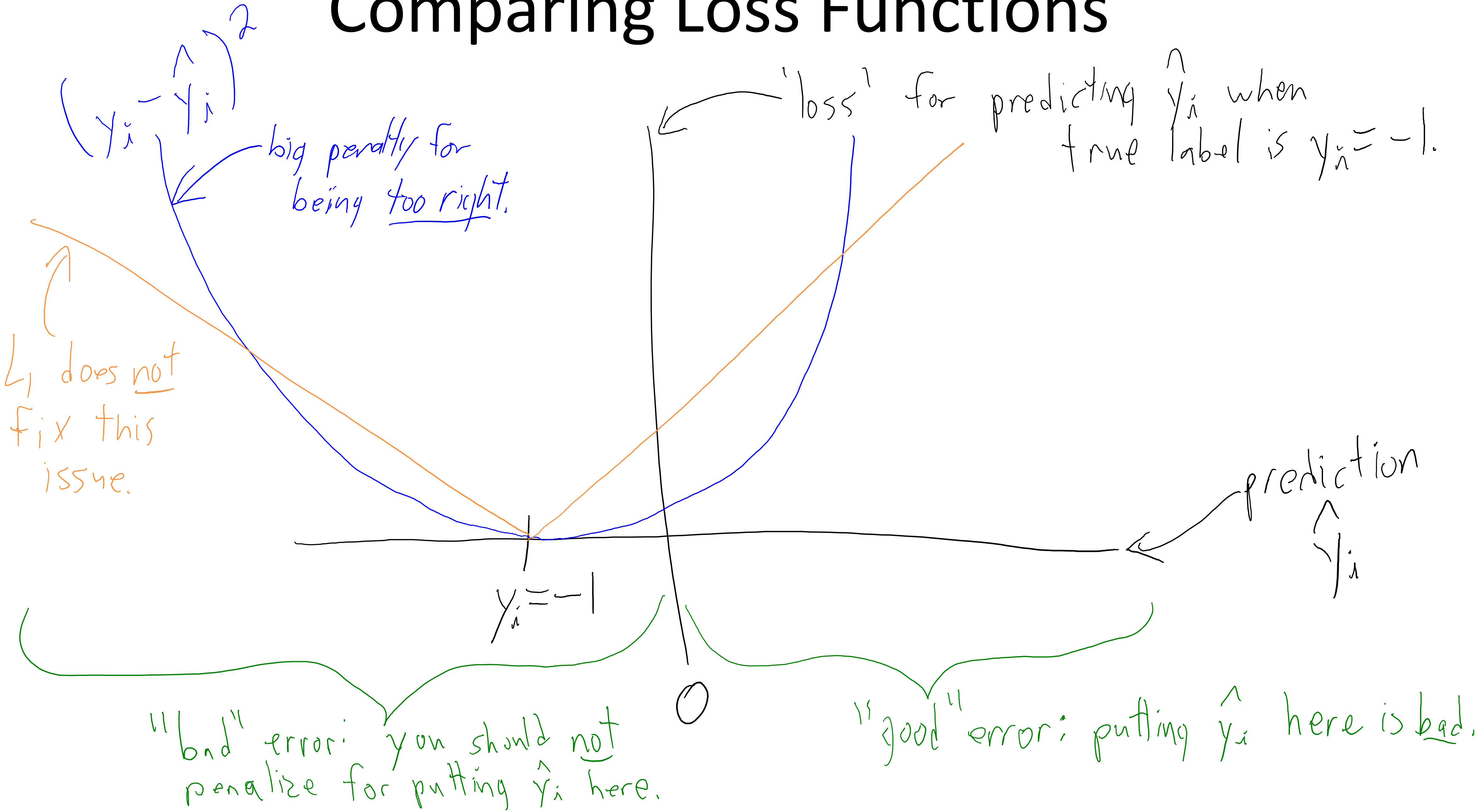
- What went wrong?
  - "Good" errors vs. "bad" errors.



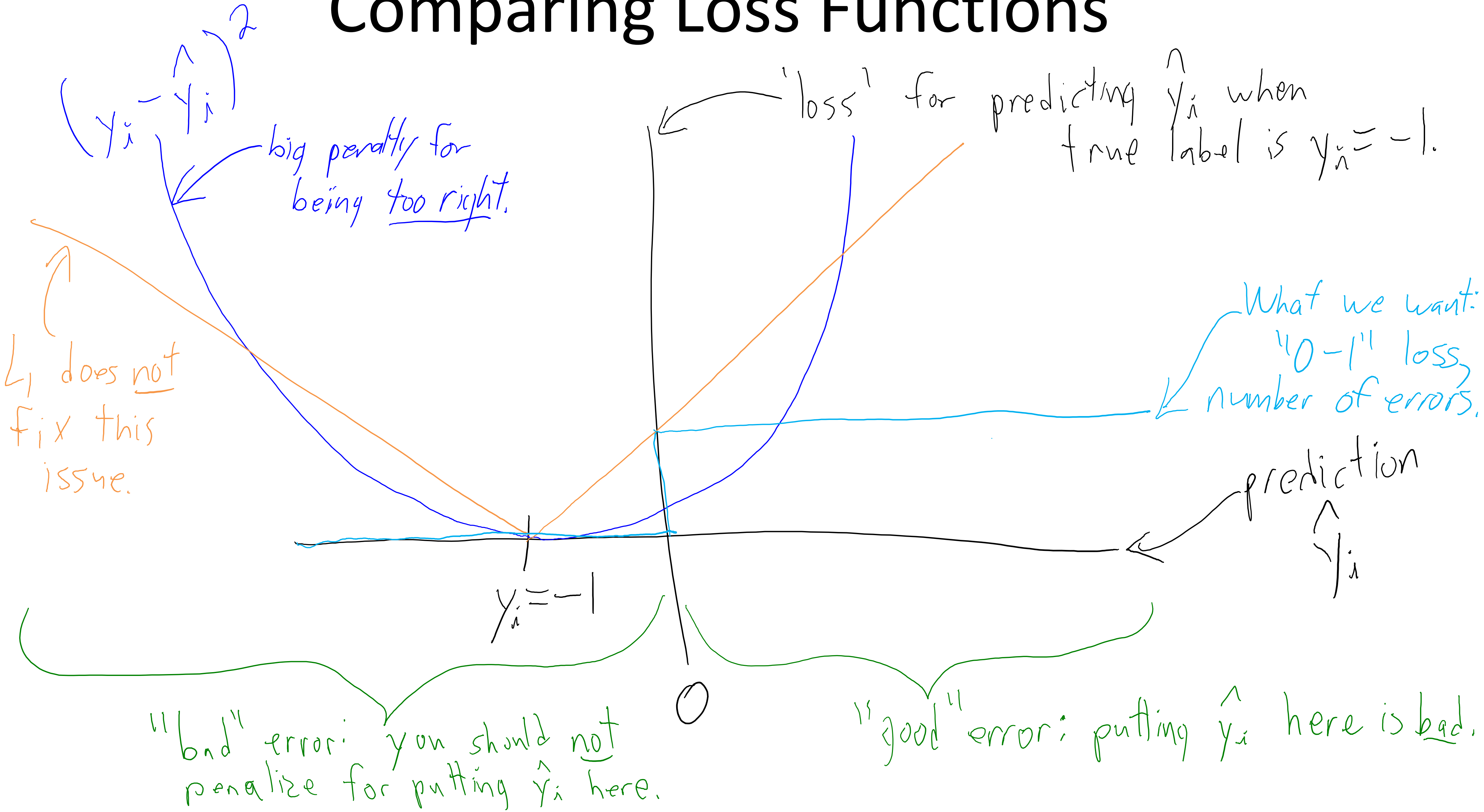
Comparing Loss Functions



Comparing Loss Functions



Comparing Loss Functions



0-1 Loss Function

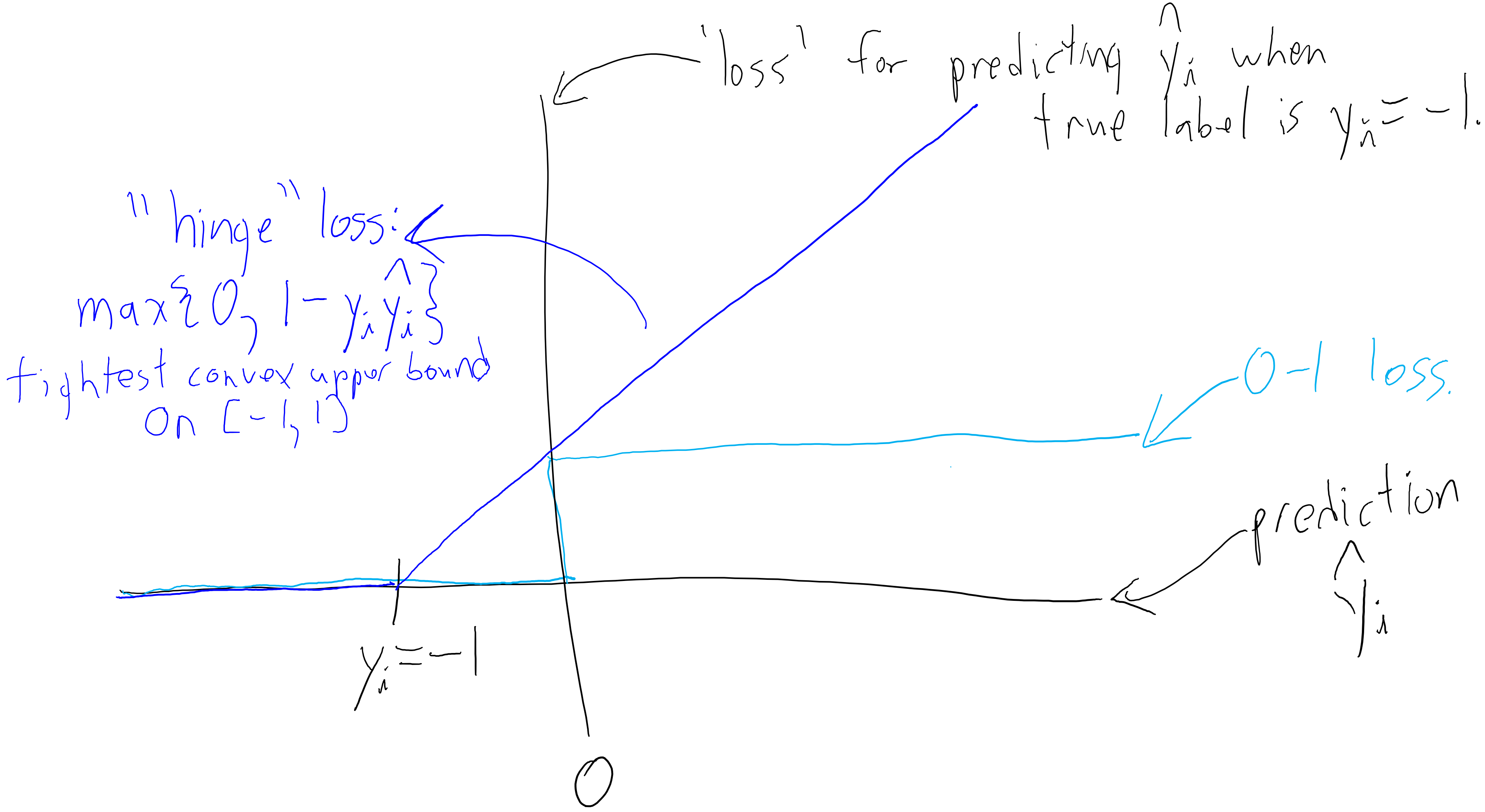
- Using the 0-1 loss function:

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{n} \underbrace{I[y_i \neq \operatorname{sign}(\overbrace{w^T x_i}^{\hat{y}_i})]}_{\text{number of errors}}$$

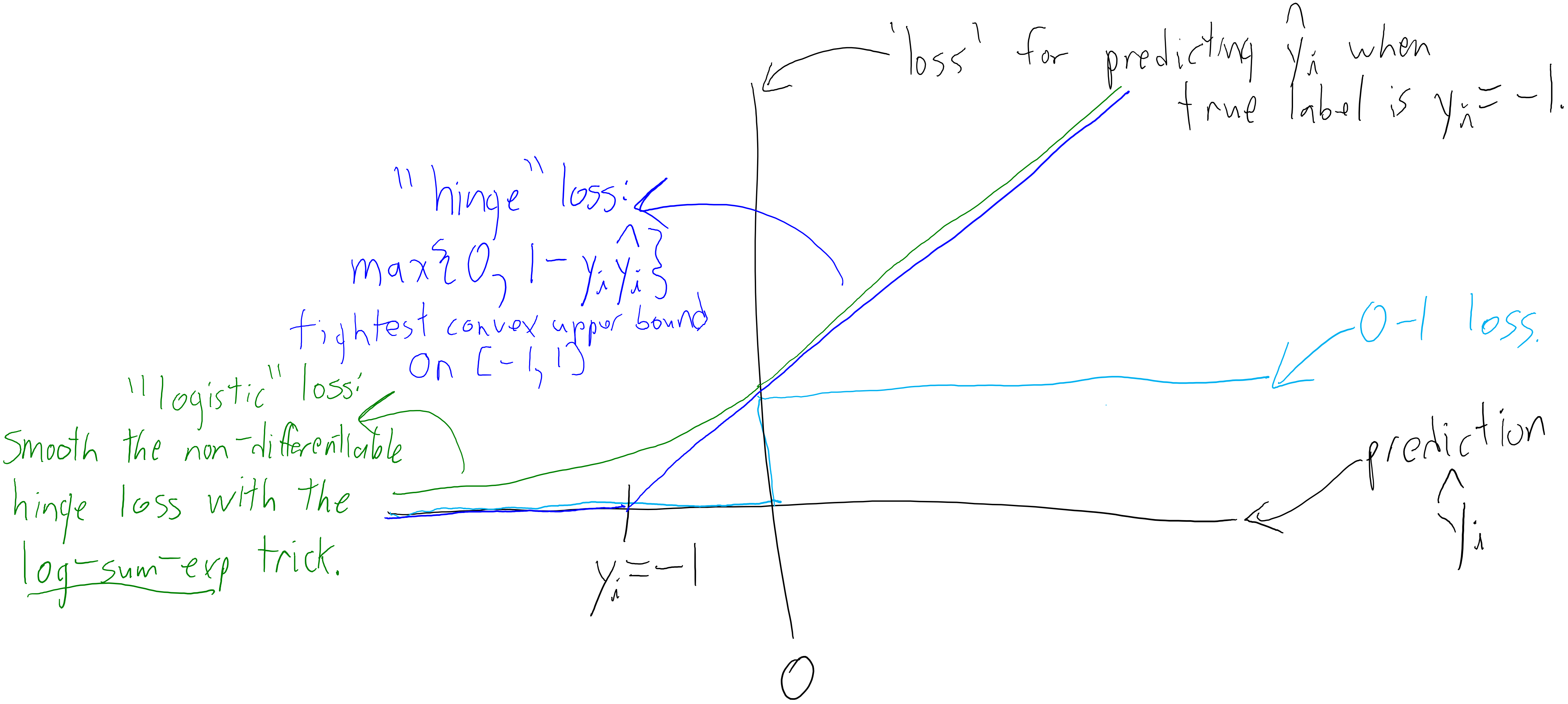
"Indicator" function: $I(\text{expr}) = \begin{cases} 1 & \text{if 'expr' is true} \\ 0 & \text{if 'expr' is false} \end{cases}$

- Can we solve this non-convex problem?
- If there exists a perfect classifier:
  - Yes, 'perceptron' algorithm returns a solution.
- If there does not exist a perfect classifier:
  - Finding the 'w' minimizing 0-1 loss is a hard problem.

Convex Approximations to 0-1 Loss



Convex Approximations to 0-1 Loss



Convex Approximations to 0-1 Loss

- Convex upper-bound on 0-1 loss is using hinge loss:

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{n} \max\{0, 1 - y_i w^T x_i\}$$

- Solution will be a perfect classifier, if one exists.
- But it is non-differentiable.
- We can smooth 'max' function with 'log-sum-exp':

$$\max\{0, 1 - y_i w^T x_i\} \approx \log(\exp(0) + \exp(-y_i w^T x_i))$$

- Using this approximation, we obtain logistic regression:

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i))$$

Logistic Regression

- Fit (convex/smooth) logistic regression using gradient descent.
- You should add an L2- or L1-regularizer, too.
- Hinge loss and logistic regression are used EVERYWHERE!
  - Training and testing are both fast.
  - It is easy to understand what the weights 'w_j' mean.
  - With high-dimensional features and regularization, often good test error.
  - Otherwise, often good test error with RBF basis and regularization.
  - Smoother predictions than random forests.
  - Predictions have probabilistic interpretation.

Generative vs. Discriminative Models

- In supervised learning part 1, we discussed generative models:

$$p(y_i | x_i) \propto p(x_i | y_i) p(y_i)$$

  - For example, naïve Bayes.
- The other type of probabilistic classifiers is discriminative models:

Directly model $p(y_i | x_i)$

  - Logistic regression is equivalent to using:

$$p(y_i | x_i) = \frac{1}{1 + \exp(-y_i w^T x_i)}$$

- Theory and practice indicate:
  - Generative models work better when we don't have much data.
  - Discriminative models work better when we have a lot of data.
  - Usually, logistic regression works much better than naïve Bayes.
- Probabilistic perspective also suggests multi-class generalization ('multinomial' logistic)

Other Motivations for Logistic Regression

- We motivated logistic loss as smooth/convex approximation to 0-1.
- We arrive at same model from several different perspectives:
  - Maximum likelihood estimate with logistic likelihood:

$$p(y_i \mid x_i) = \frac{1}{1+\exp(-y_i w^T x_i)}$$

(logarithm gives logistic loss)

maximizing $p(y_i \mid x_i) \Longleftrightarrow$ minimizing $\log p(y_i \mid x_i)$

  - Linear model of 'log-odds':

$$\log\left(\frac{p(y_i = 1 \mid x_i)}{p(y_i = -1 \mid x_i)}\right) = w^T x_i$$

  - Linear parameterization of Bernoulli ('coin flipping') distribution.
  - 'Maximum entropy' subject to 'moment constraints':
    - Distribution that makes fewest assumptions, subject to fitting data.

Multinomial Logistic Regression

- For non-binary classification, we have weight 'w_c' for class 'c'.
- Classify by maximizing inner product:

$$y_i = \max_c \{ w_c^T x_i \}$$

- Probabilistic model and corresponding error:

$$p(y_i = c \mid x_i) = \frac{\exp(w_c^T x)}{\sum_{c'} \exp(w_{c'}^T x)}$$

$$\underset{w_1, w_2, \ldots, w_C \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{n} \left\{ w_{y_i}^T x_i - \log\left(\sum_c \exp(w_c^T x_i) \right) \right\}$$

(minimize negative logarithm of probabities)

- Binary logistic regression is special case where $w_2 = 0$.

Summary

- Standardizing features puts features on the same scale.
- Global vs. local features allows ‘personalized’ predictions.
- Classification using regression works if done right.
- 0-1 loss is the ideal loss, but is non-smooth and non-convex.
- Logistic regression uses a convex and smooth approximation to 0-1.

- Next time:
  - One more reason to use regularization, and how to find gold.