CPSC 340: Machine Learning and Data Mining

Logistic Regression

Fall 2016

Admin

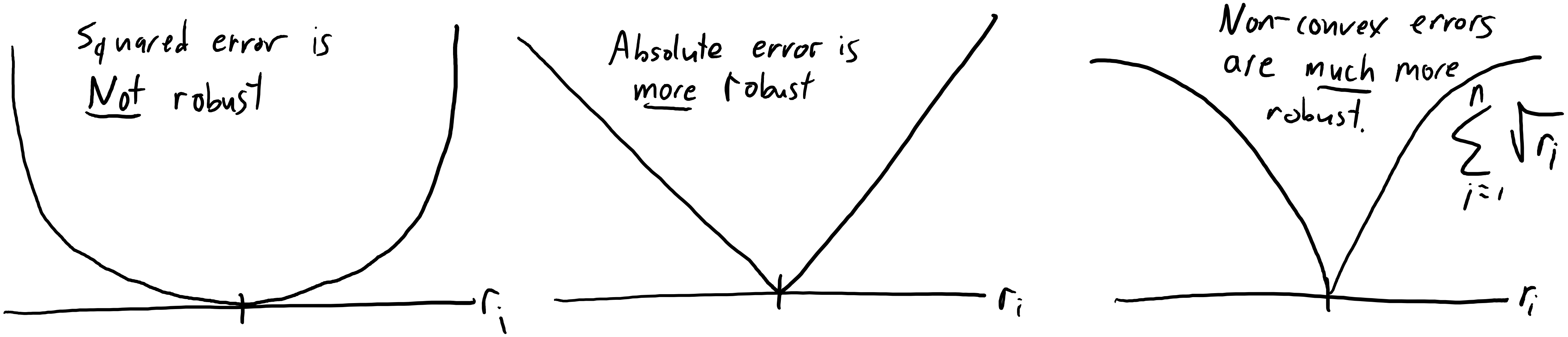
- Assignment 1:
  - Marks visible on UBC Connect.
- Assignment 2:
  - Solution posted after class.
- Assignment 3:
  - Due Wednesday (at any time on Earth).
  - Solutions will be released next Wednesday after class.
- Tutorial room change: T1D (Monday @5pm) moved to DMP 101.
- Midterm on Friday October 28.
  - Practice midterm and list of topics posted (covers Assignments 1-3)
  - In class, 55 minutes, closed-book, cheat sheet: 2-pages each double-sided.

Summary of Last Lecture

1. Error functions:
   - Squared error is sensitive to outliers.
   - Absolute (L_1) error and Huber error are more robust to outliers.
   - Brittle (L_∞) error is more sensitive to outliers.
2. L_1 and L_∞ error functions are non-differentiable:
   - Finding 'w' minimizing these errors is harder.
3. We can approximate these with differentiable functions:
   - L_1 can be approximated with Huber.
   - L_∞ can be approximated with log-sum-exp.
4. Gradient descent finds local minimum of differentiable function.
5. For convex functions, any local minimum is a global minimum.

Very Robust Regression

- Consider data with extreme outliers:

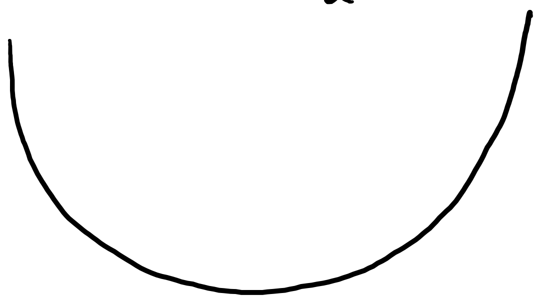


- Non-convex errors can be very robust:
  - Eventually 'give up' on trying to make large errors smaller.
- But with non-convex errors, finding global minimum is hard.
- Absolute value is the most robust convex error function.

How do we know if a function is convex?

- Some useful tricks for showing a function is convex:
  - 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all 'w'.

Consider $f(w) = \frac{1}{2}aw^2$ for $a > 0$.

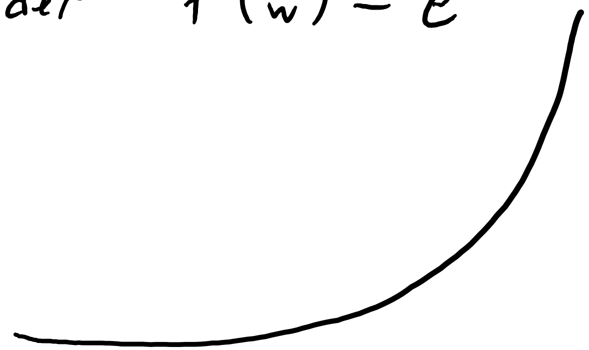


We have $f'(w) = aw$

and $f''(w) = a > 0$

By assumption

Consider $f(w) = e^w$



We have $f'(w) = e^w$

and $f''(w) = e^w > 0$

By definition of exponential function.

How do we know if a function is convex?

- Some useful tricks for showing a function is convex:
  - 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all 'w'.
  - A convex function multiplied by non-negative constant is convex.

We showed that $f(w) = e^w$ is convex, so $f(w) = 10e^w$ is convex.

How do we know if a function is convex?

- Some useful tricks for showing a function is convex:
  - 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all 'w'.
  - A convex function multiplied by non-negative constant is convex.
  - Norms and squared norms are convex.

$\|w\|$, $\|w\|^2$, $\|w\|_1$, $\|w\|_\infty$, $\|w\|_1^2$, and so on are all convex.

How do we know if a function is convex?

- Some useful tricks for showing a function is convex:
  - 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all 'w'.
  - A convex function multiplied by non-negative constant is convex.
  - Norms and squared norms are convex.
  - The sum of convex functions is a convex function.

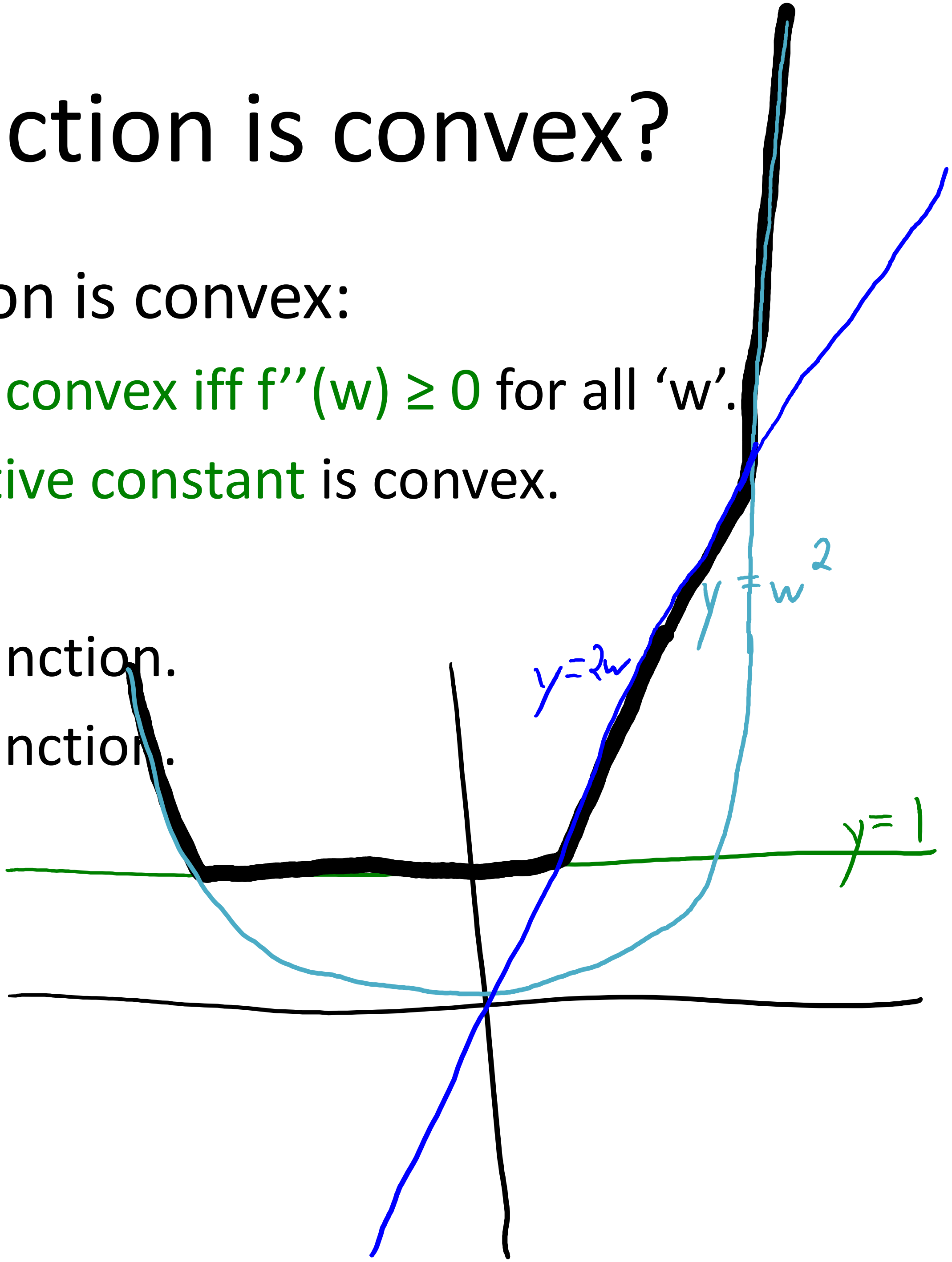
$$f(x) = \underbrace{10e^{w}}_{\text{From earlier}} + \underbrace{\frac{1}{2}}_{\text{constant}}\underbrace{\|w\|^2}_{\text{norm squared}} \quad \text{is convex}$$

How do we know if a function is convex?

- Some useful tricks for showing a function is convex:
  - 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all 'w'.
  - A convex function multiplied by non-negative constant is convex.
  - Norms and squared norms are convex.
  - The sum of convex functions is a convex function.
  - The max of convex functions is a convex function.

$f(w) = \max\{1, 2w, w^2\}$ is convex.

convex



How do we know if a function is convex?

- Some useful tricks for showing a function is convex:
  - 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all ‘w’.
  - A convex function multiplied by non-negative constant is convex.
  - Norms and squared norms are convex.
  - The sum of convex functions is a convex function.
  - The max of convex functions is a convex function.
  - Composition of a convex function and a linear function is convex.

If ‘f’ is convex the $f(Xw - y)$ is convex.

How do we know if a function is convex?

- Some useful tricks for showing a function is convex:
  - 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all 'w'.
  - A convex function multiplied by non-negative constant is convex.
  - Norms and squared norms are convex.
  - The sum of convex functions is a convex function.
  - The max of convex functions is a convex function.
  - Composition of a convex function and a linear function is convex.
- But: not true that composition of convex with convex is convex:

Even if 'f' is convex and 'g' is convex, $f(g(w))$ might not be convex.

E.g. x^2 is convex and $-\log(x)$ is convex but $-\log(x^2)$ is not convex.

Example: Convexity of Linear Regression

- Consider linear regression objective with error function 'g':

$$f(w) = \sum_{i=1}^{n} g(w^T x_i - y_i)$$

- Sufficient for 'g' to be convex for 'f' to be convex:
  - Then each term is composition of convex with linear.
  - And sum of convex is convex.
- Examples:

For squared error $g(r_i) = \frac{1}{2} r_i^2$ so $g''(r_i) = 1$ and 'f' is convex.

For absolute error $g(r_i) = |r_i|$ which is a <u>norm</u> so 'f' is convex.

Example: Convexity of Linear Regression

- Consider linear regression objective with error function 'g':

$$f(w) = \sum_{i=1}^{n} g(w^T x_i - y_i) + \frac{\lambda}{2}\|w\|^2$$

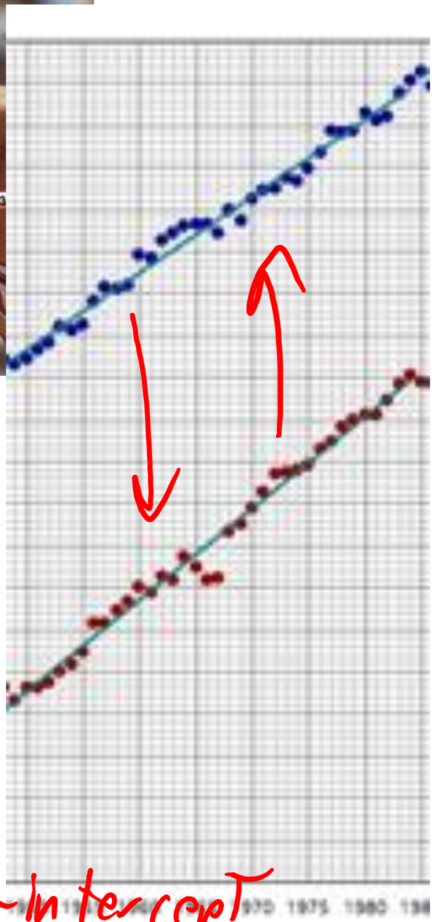
- Sufficient for 'g' to be convex for 'f' to be convex:
  - Then each term is composition of convex with linear.
  - And sum of convex is convex.
- Same condition applies with L_2-regularization.

Linear Models 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, our linear model is:

$$height = w_0 + w_1 * year + w_2 * gender$$

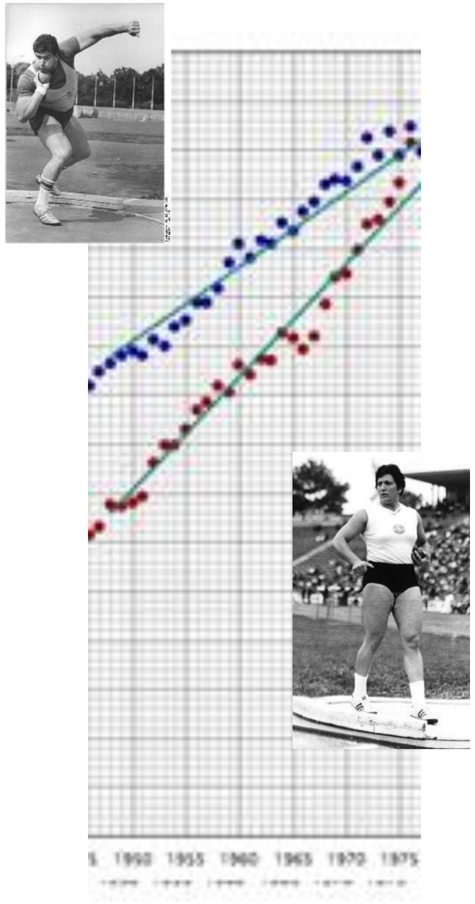
- The 'gender' variable causes a change in y-intercept:

If $gender == 0$ then $height = w_0 + w_1 * year$

If $gender == 1$ then $height = w_0 + w_1 * year + w_2$

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

Linear Models with Binary Features



- What if different genders have different slopes?
  - You can use gender-specific feature.

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

$distance = w_0 + w_1 * year$ (if gender = 1)

$distance = w_3 + w_4 * year$ (if gender = 0)

separate bias

separate slope

Linear Models with Binary Features

- That trick 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

$\Longrightarrow$

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" across genders

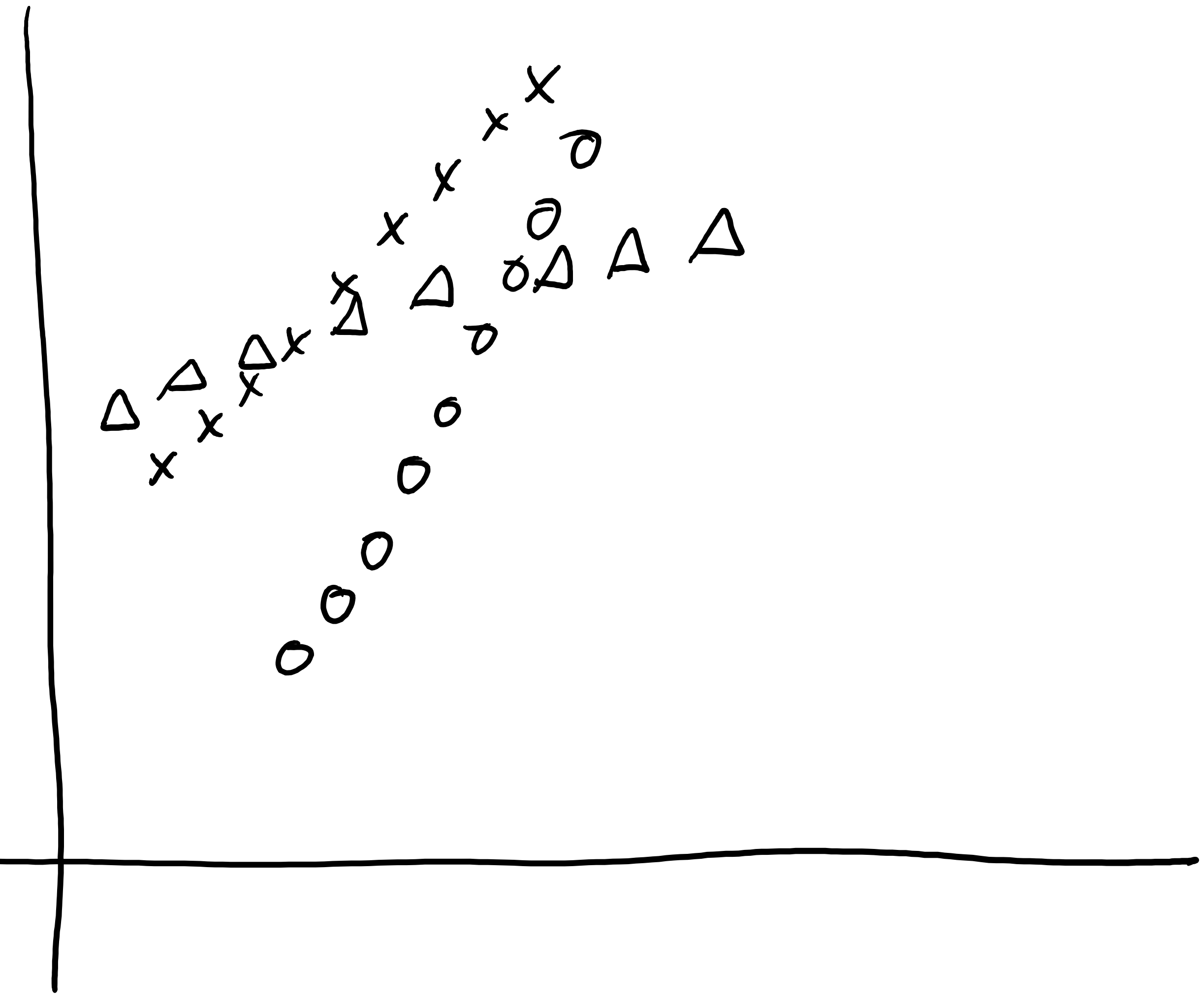
"local" to gender

$y_i = w_0 + w_1 * year + w_3 * year$

Linear Models with Binary Features

$X =$

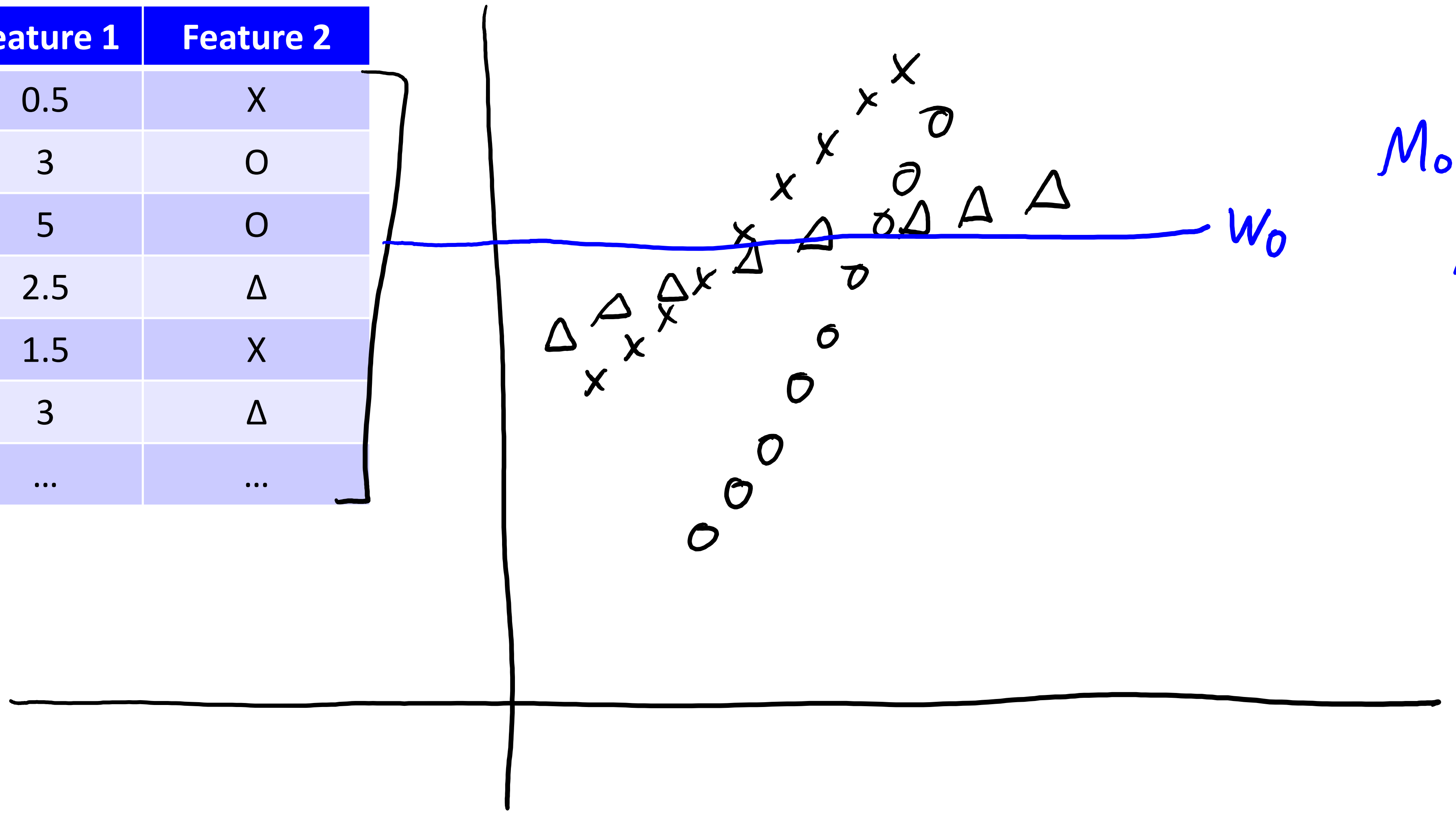
Feature 1	Feature 2
0.5	X
3	O
5	O
2.5	Δ
1.5	X
3	Δ
…	…



Linear Models with Binary Features

$X =$

Feature 1	Feature 2
0.5	X
3	O
5	O
2.5	Δ
1.5	X
3	Δ
…	…



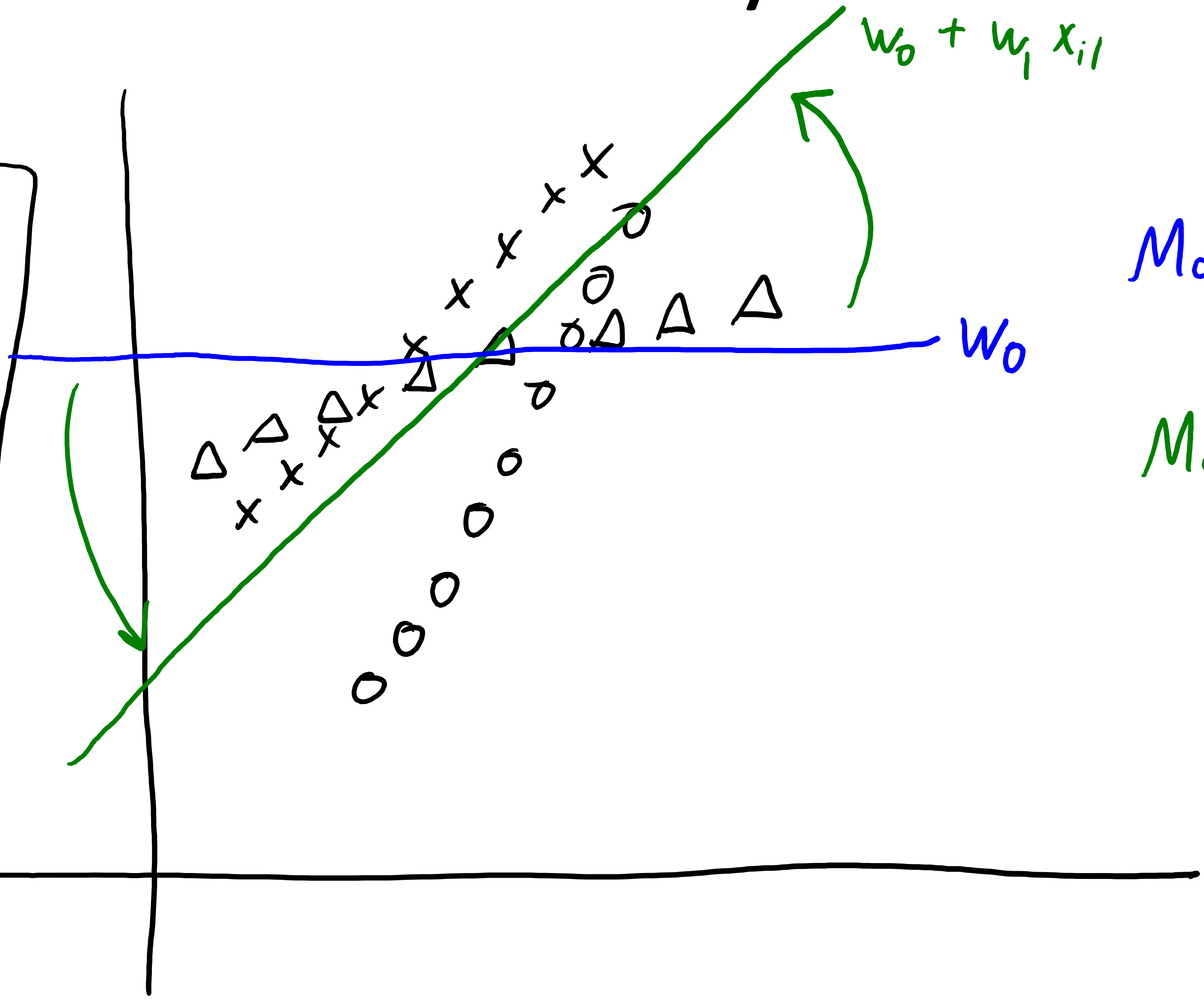
Model 1: only bias

$y_i = w_0$

Linear Models with Binary Features

$X =$

Feature 1	Feature 2
0.5	X
3	O
5	O
2.5	Δ
1.5	X
3	Δ
...	...



Model 1: only bias

$y_i = w_0$

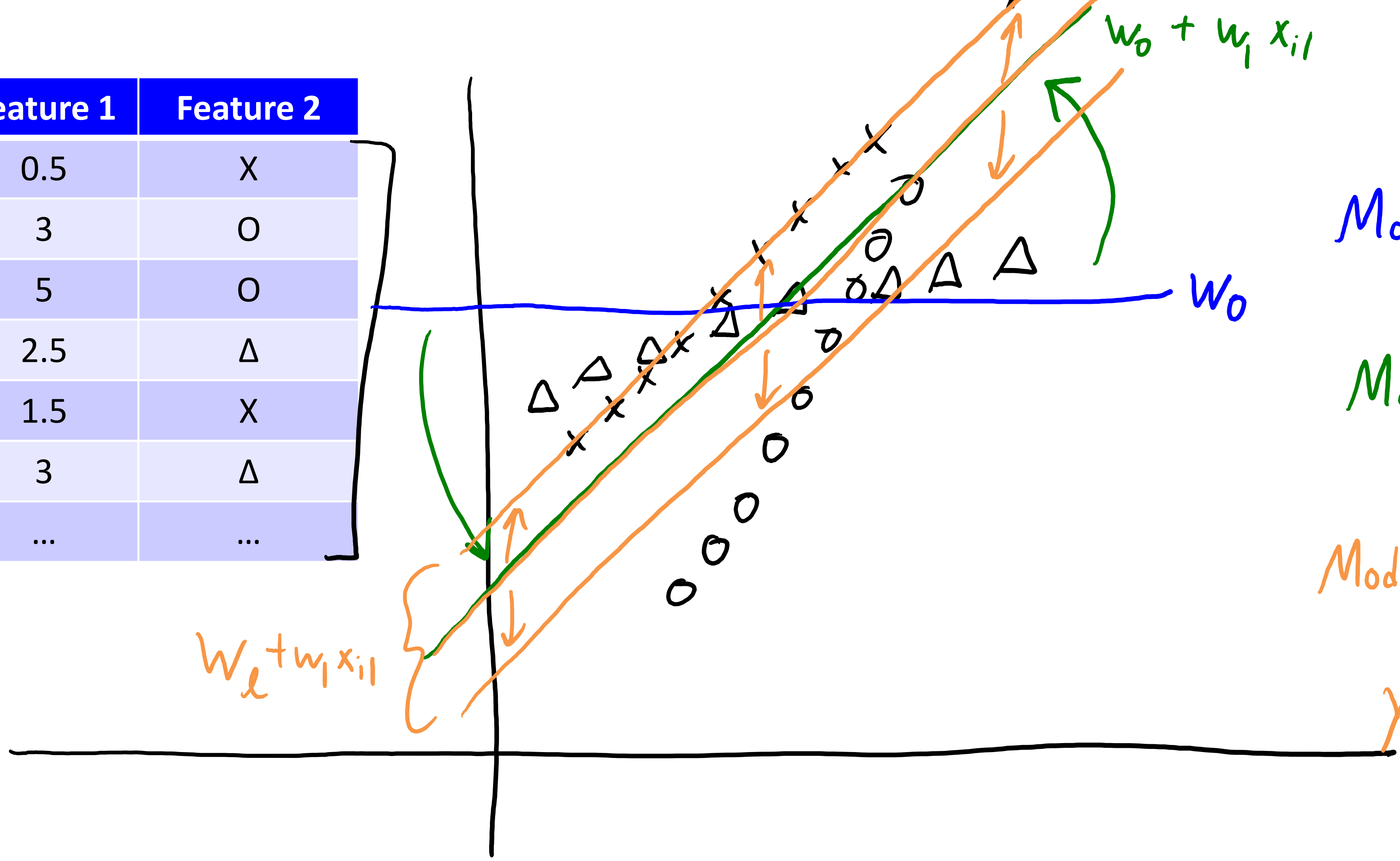
Model 2: bias + feature 1

$y_i = w_0 + w_1 x_{i1}$

Linear Models with Binary Features

$X =$

Feature 1	Feature 2
0.5	X
3	O
5	O
2.5	Δ
1.5	X
3	Δ
…	…



Model 1: only bias

$y_i = w_0$

Model 2: bias + feature 1

$y_i = w_0 + w_1 x_{i1}$

Model 3: "local" bias + feature 1

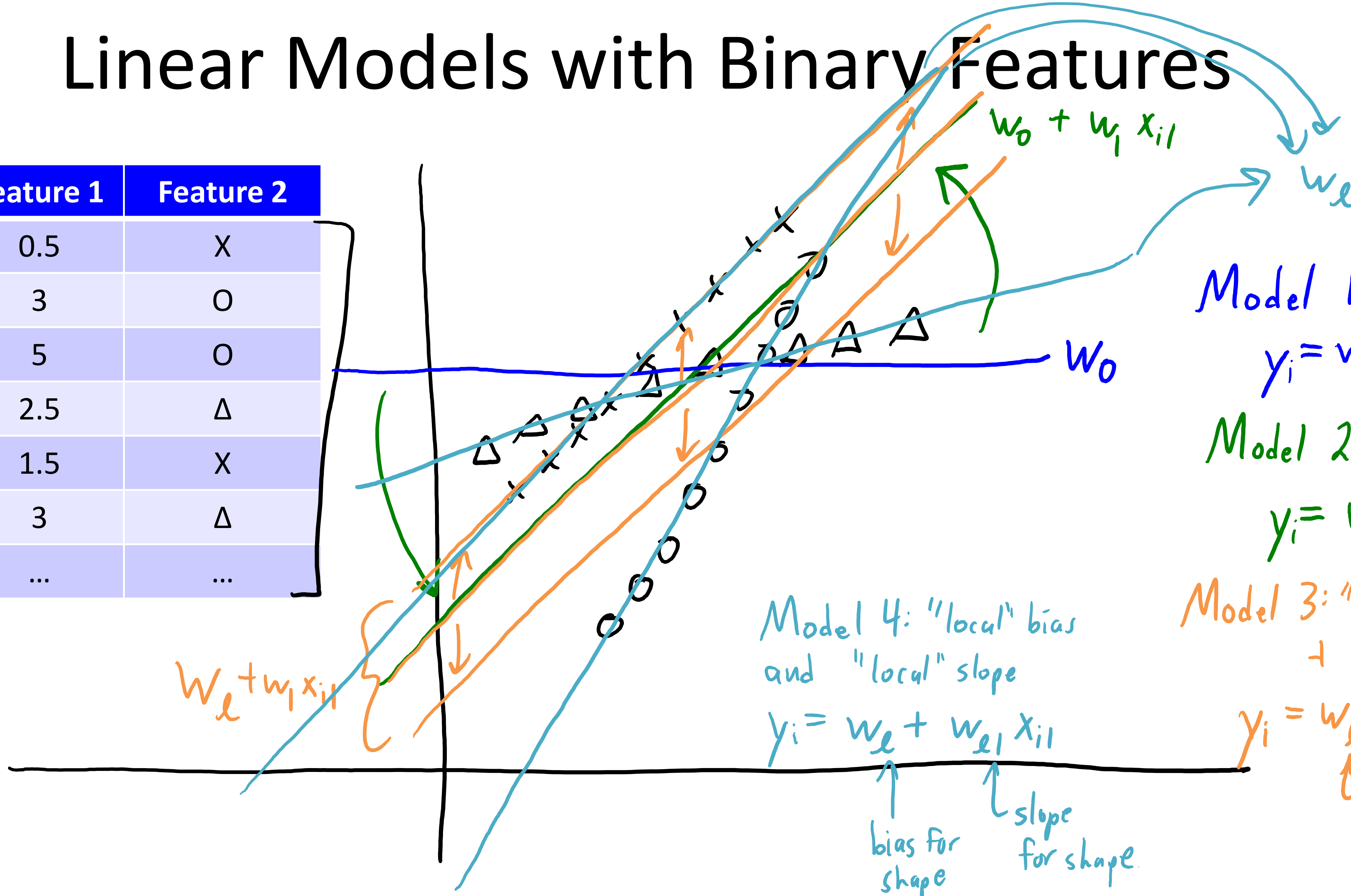
$y_i = w_\ell + w_1 x_{i1}$

shape

Linear Models with Binary Features

$X =$

Feature 1	Feature 2
0.5	X
3	O
5	O
2.5	Δ
1.5	X
3	Δ
…	…



Model 1: only bias

$y_i = w_0$

Model 2: bias + feature 1

$y_i = w_0 + w_1 x_{i1}$

Model 3: "local" bias + feature 1

$y_i = w_\ell + w_1 x_{i1}$

shape

Model 4: "local" bias and "local" slope

$y_i = w_\ell + w_{\ell 1} x_{i1}$

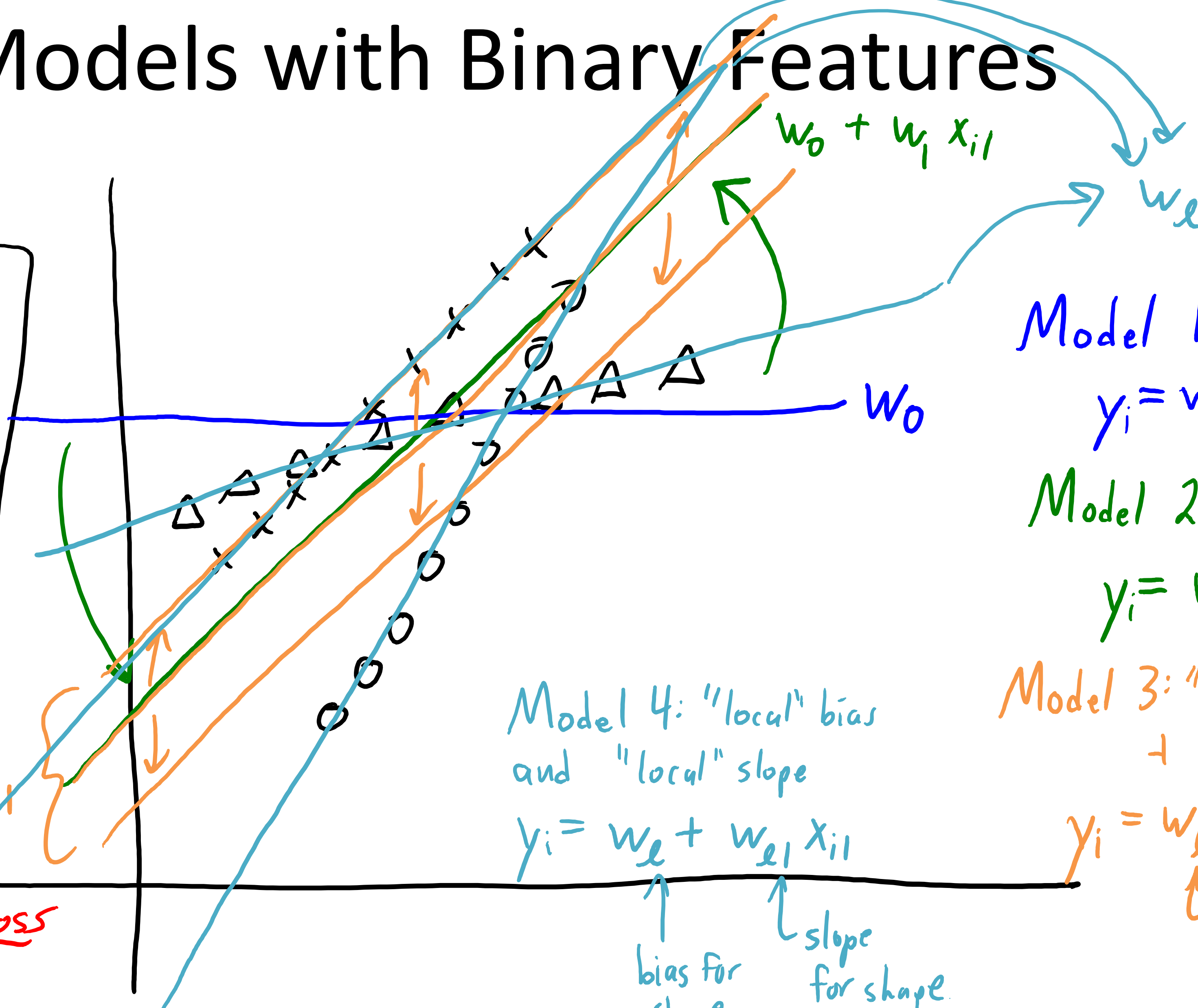
bias for shape

slope for shape

Linear Models with Binary Features

$X =$

Feature 1	Feature 2
0.5	X
3	O
5	O
2.5	Δ
1.5	X
3	Δ
…	…



Model 1: only bias

$y_i = w_0$

Model 2: bias + feature 1

$y_i = w_0 + w_1 x_{i1}$

Model 3: "local" bias + feature 1

$y_i = w_\ell + w_1 x_{i1}$

w_ℓ ← shape

Model 4: "local" bias and "local" slope

$y_i = w_\ell + w_{\ell 1} x_{i1}$

w_ℓ ← bias for shape

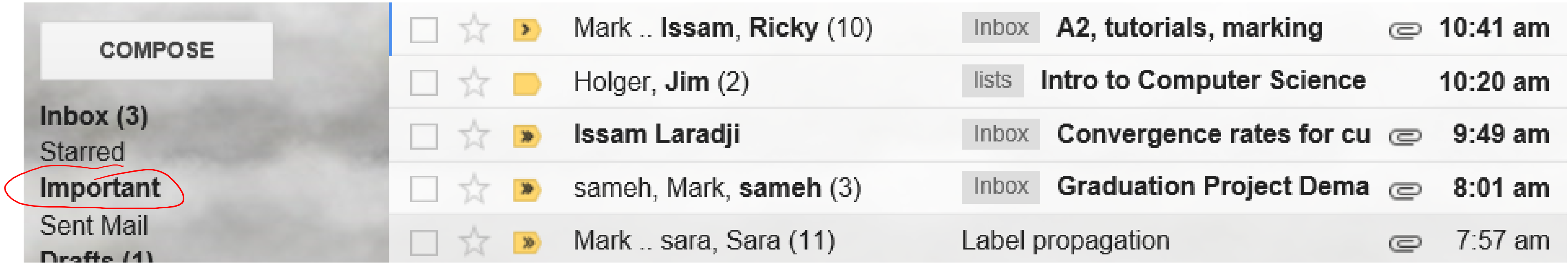
$w_{\ell 1}$ ← slope for shape

Could also share information across categories with global bias slope:

$y_i = w_0 + w_1 x_{i1} + w_\ell + w_{\ell 1} x_{i\ell}$

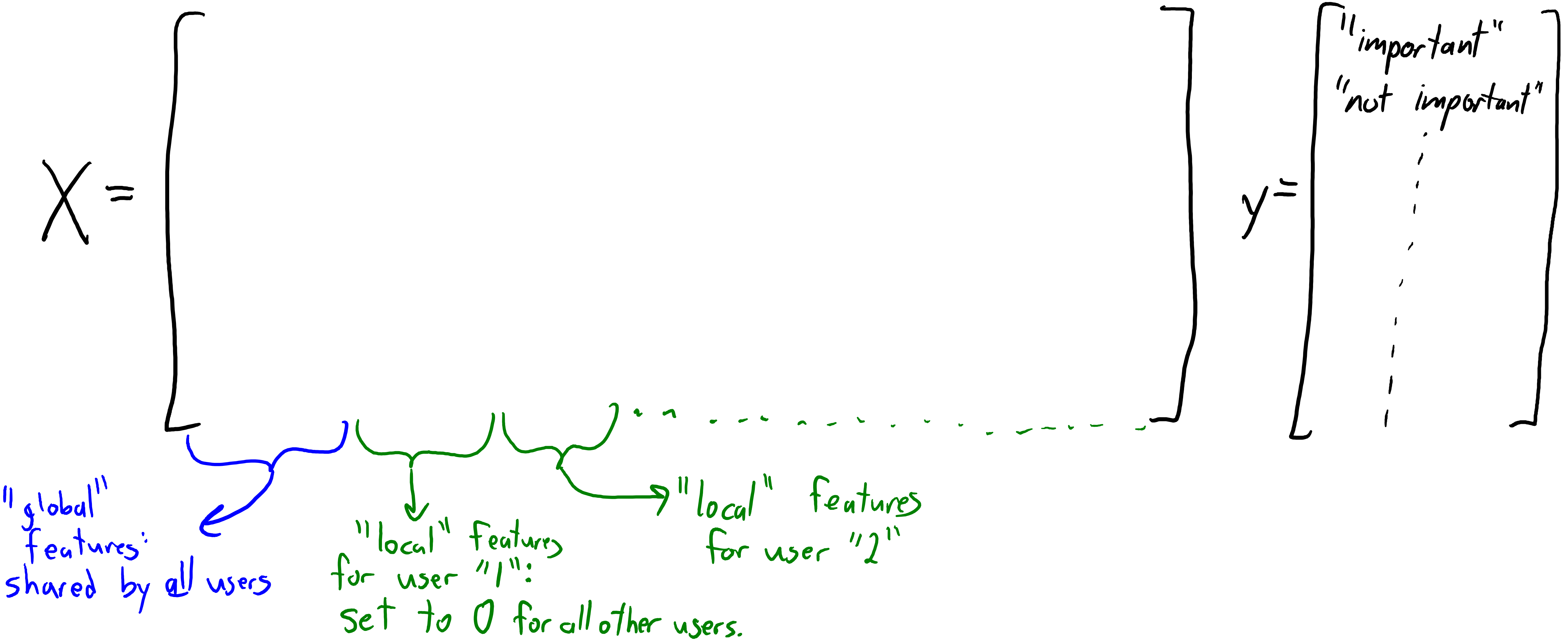
Motivation: Identifying Important E-mails

- 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.

$$y_i = \text{sign}(\underbrace{w_g^T x_g}_{\text{features/weights shared across users.}})$$

- With more data, update global features and user's local features:
  - Local features make prediction *personalized*.

$$y_i = \text{sign}(w_g^T x_g + \underbrace{w_u^T x_u}_{\text{features/weights specific to user.}})$$

  - What is important to *this* user?
- G-mails system: classification with logistic regression.

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'.
- At training time, fit a linear regression model:

$$\begin{aligned} y_i &= w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id} \\ &= w^T x_i \end{aligned}$$

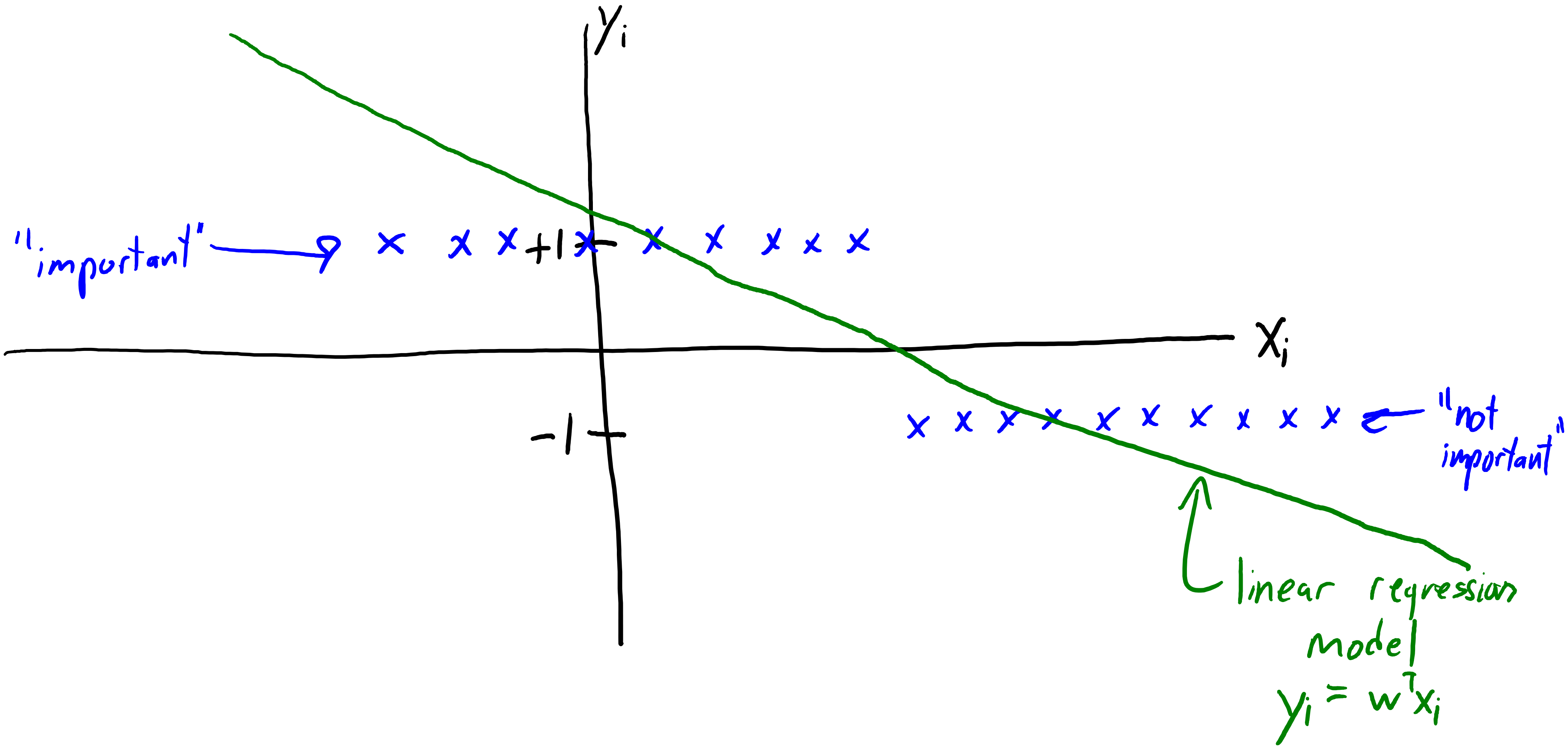
- To predict, we take the sign (i.e., closer '-1' or '+1'?):

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

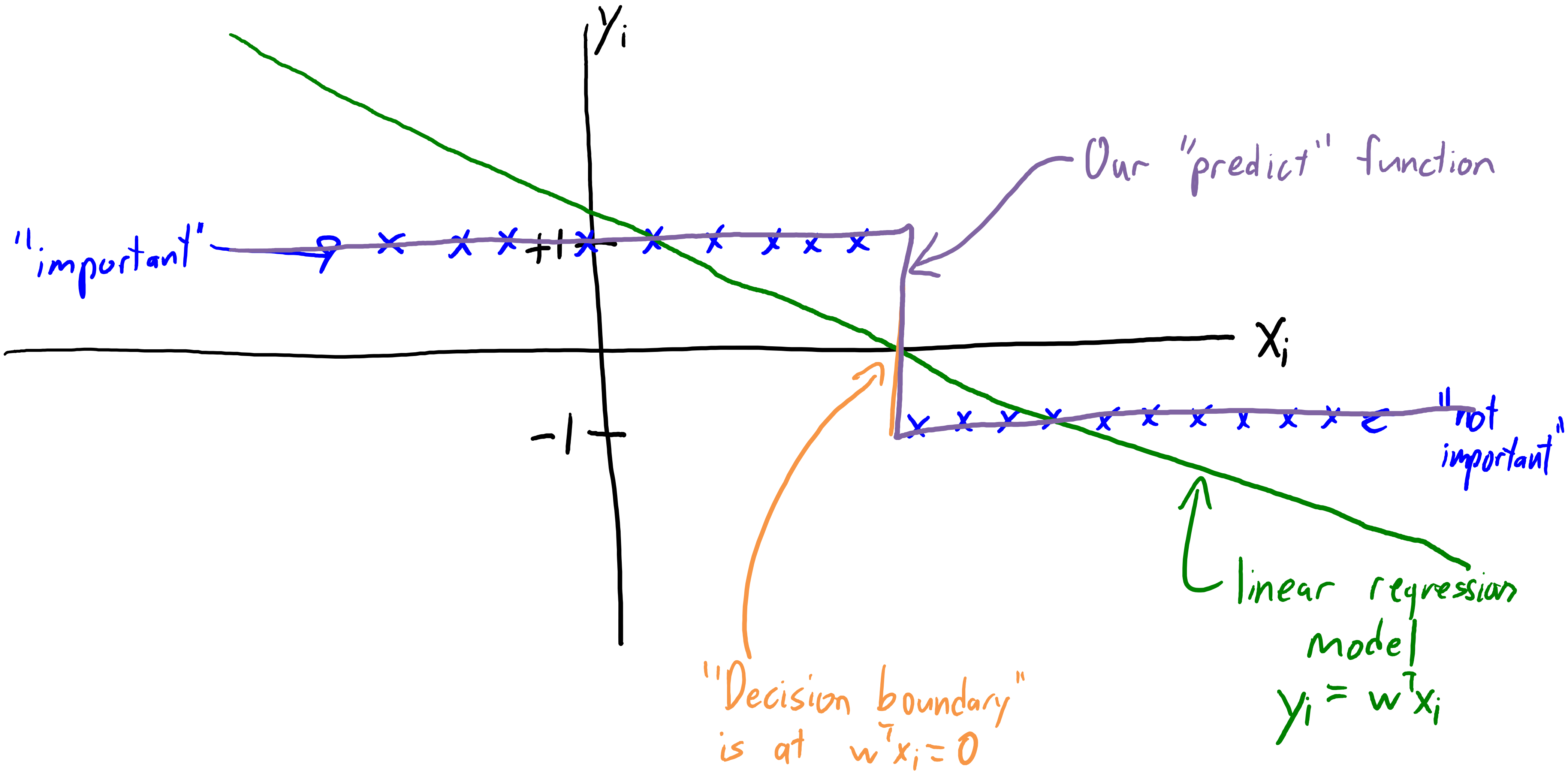
Set $y_i = +1$ if $w^T x_i > 0$

Set $y_i = -1$ if $w^T x_i < 0$

Classification using Regression

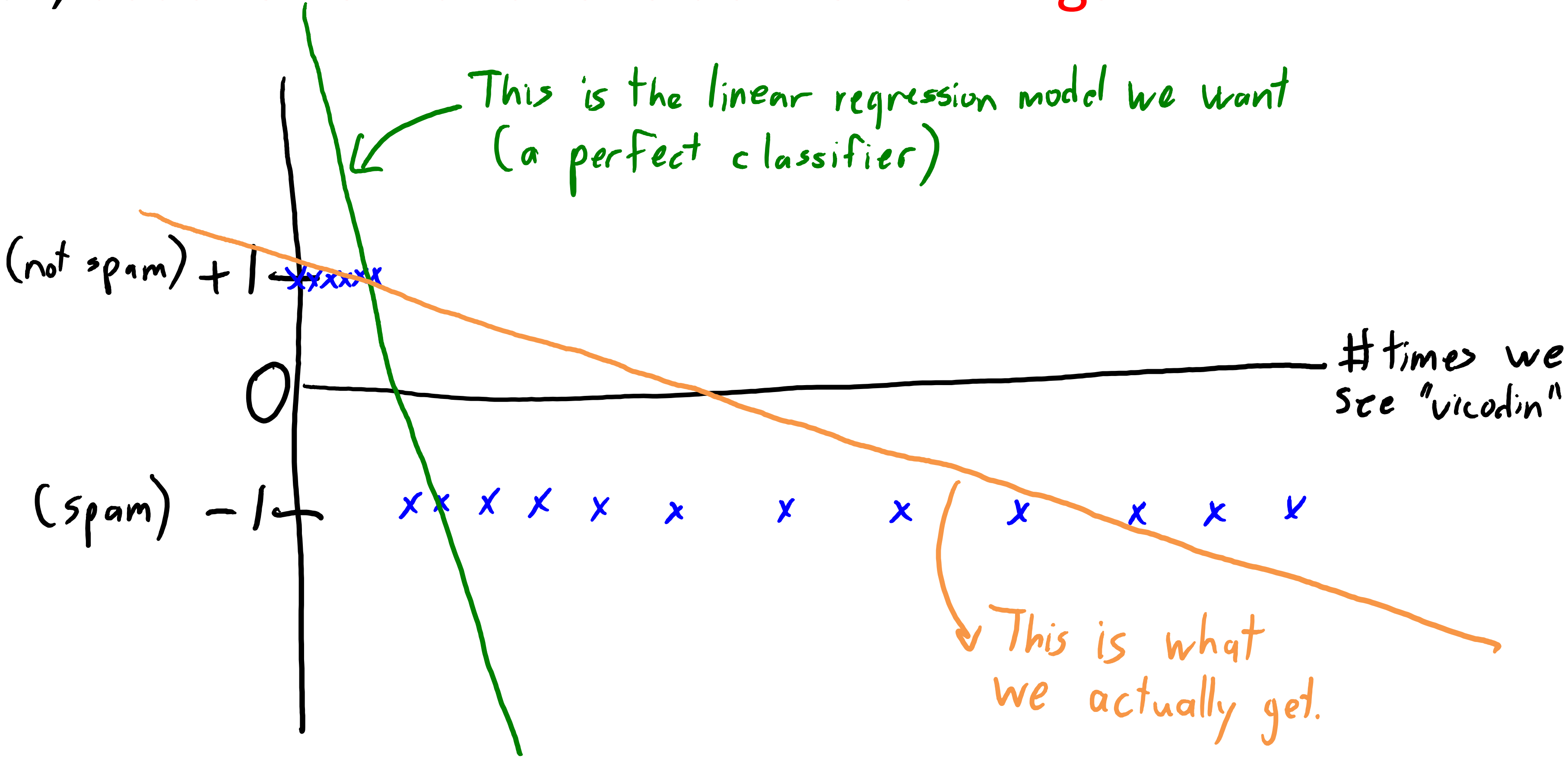


Classification using Regression



Classification Using Regression

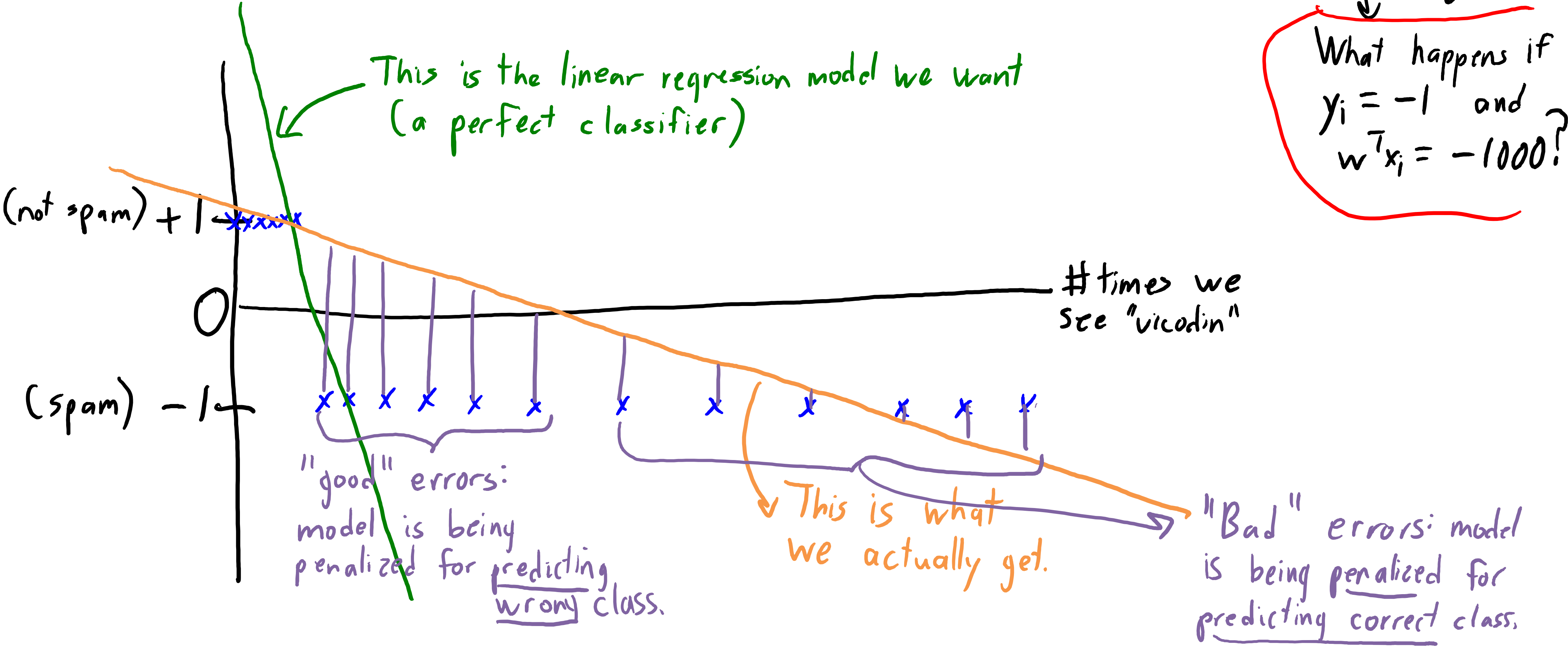
- Can use regression tricks (basis, regularization) for classification.
- But, usual error functions do weird things:



Classification Using Regression

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

$$f(w) = \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

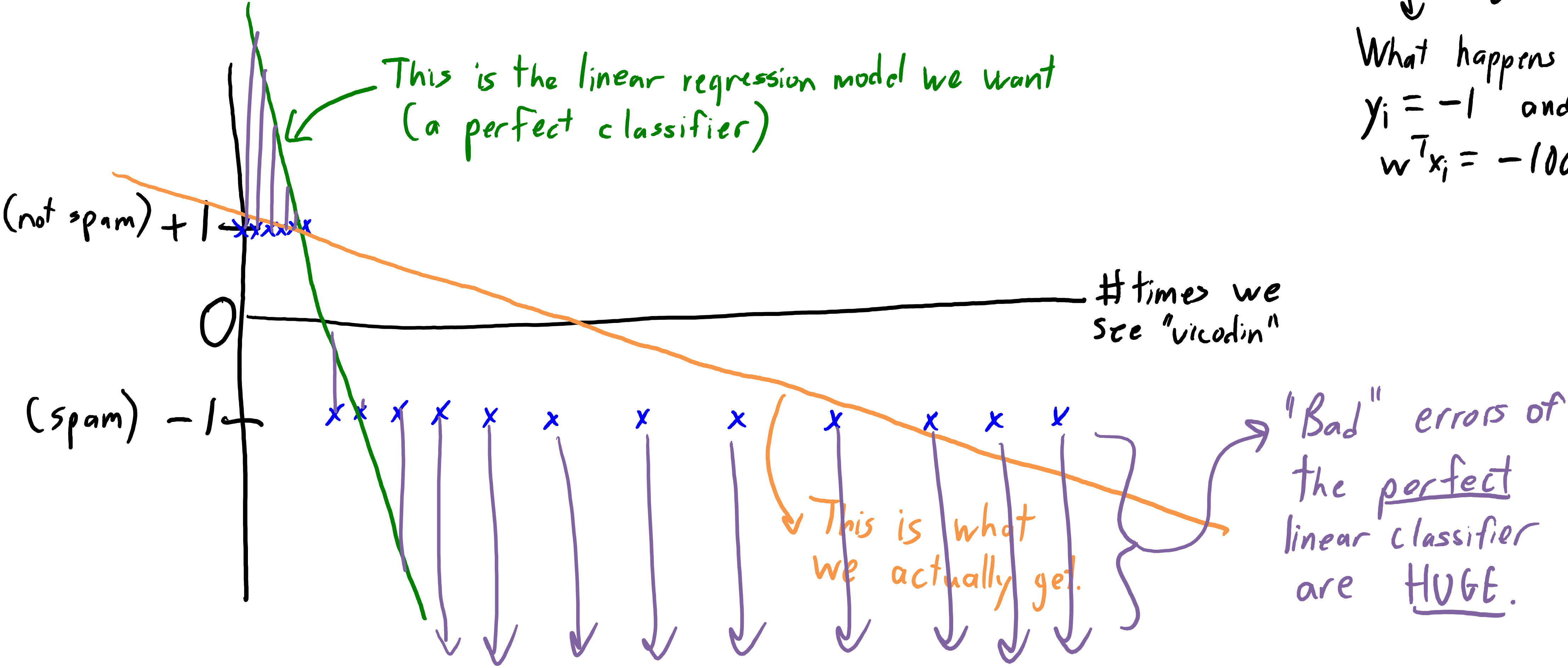


Classification Using Regression

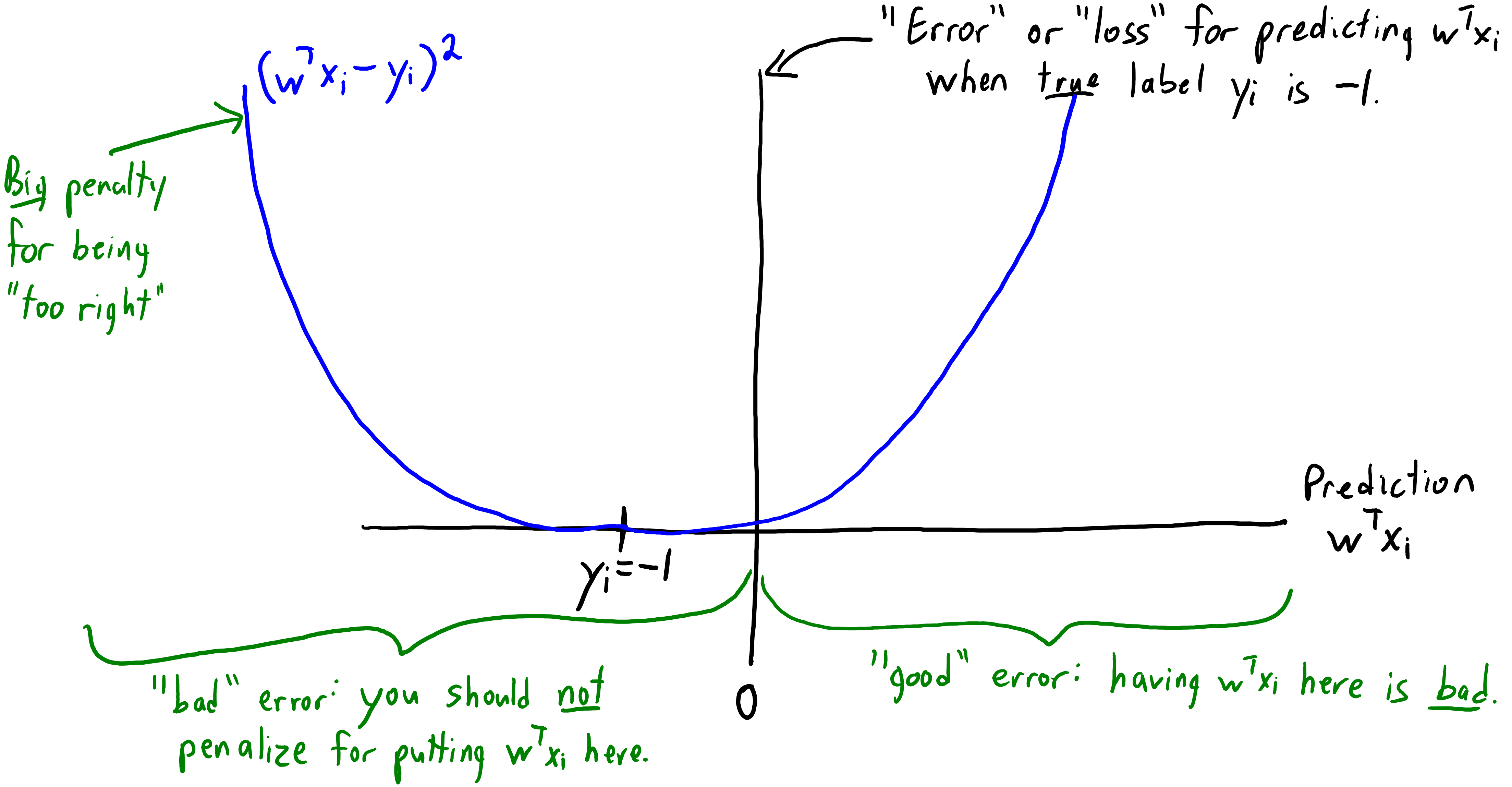
- What went wrong?
  - "Good" errors vs. "bad" errors.

$$f(w) = \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

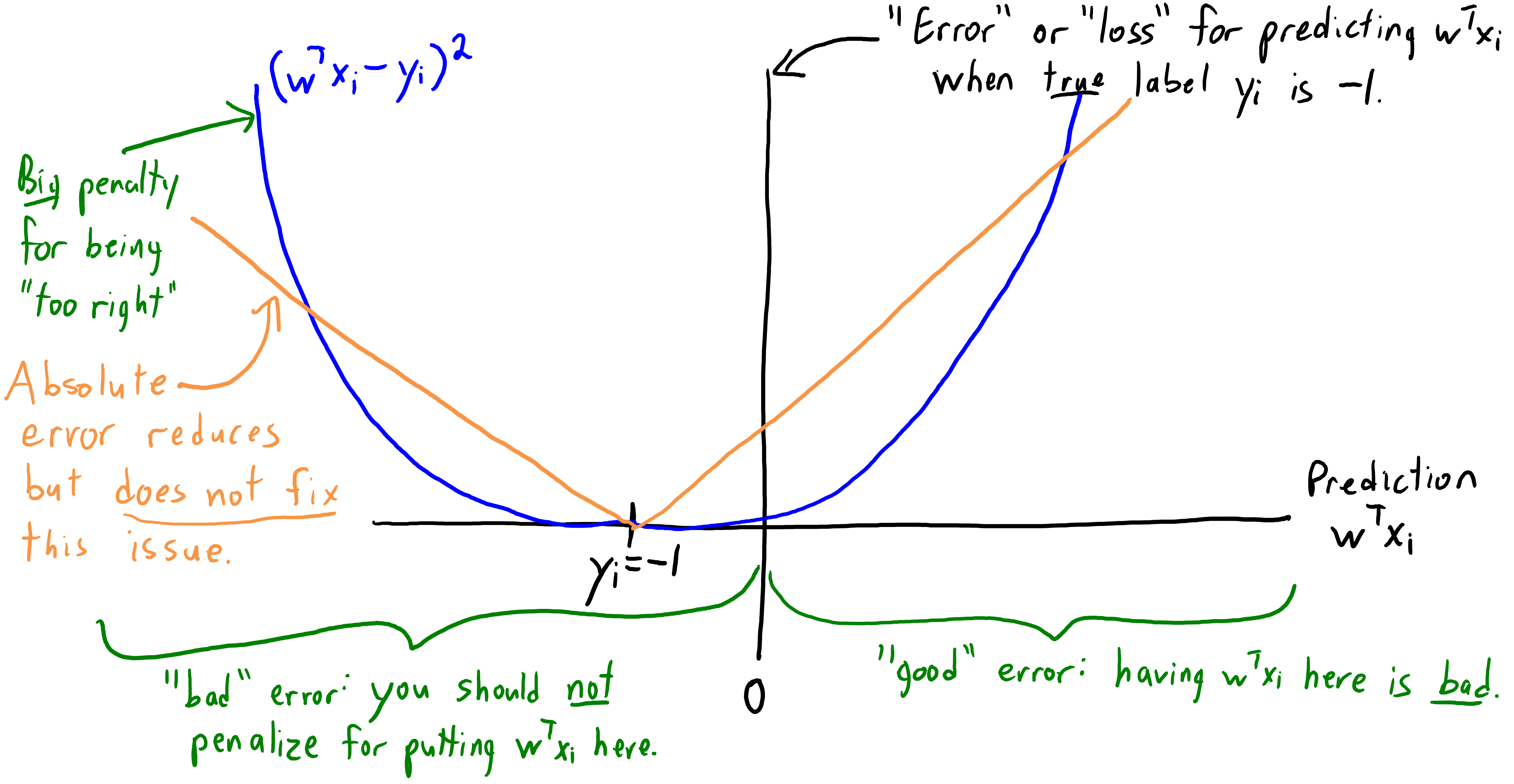
What happens if $y_i = -1$ and $w^T x_i = -1000$?



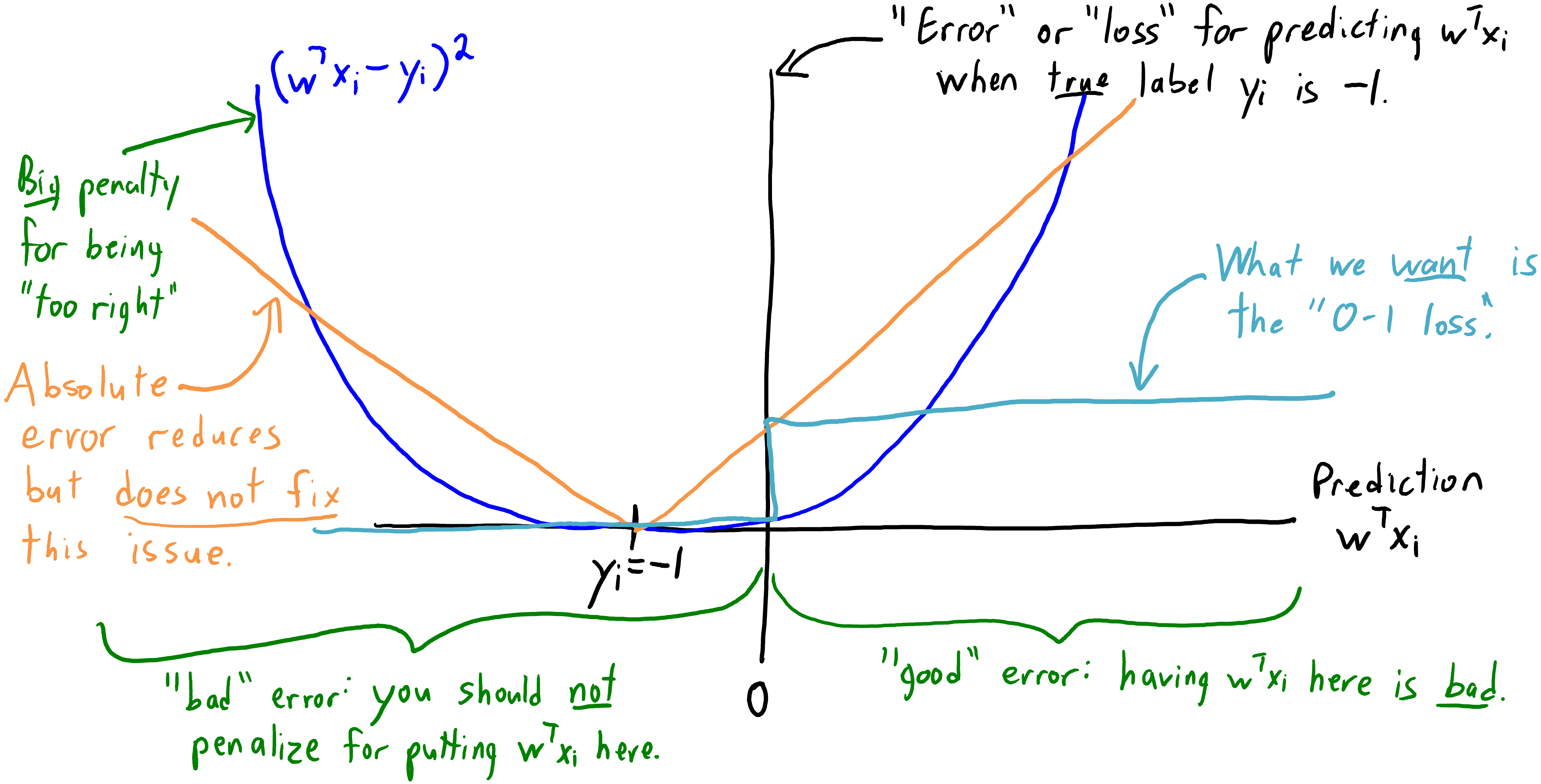
Comparing Loss Functions



Comparing Loss Functions



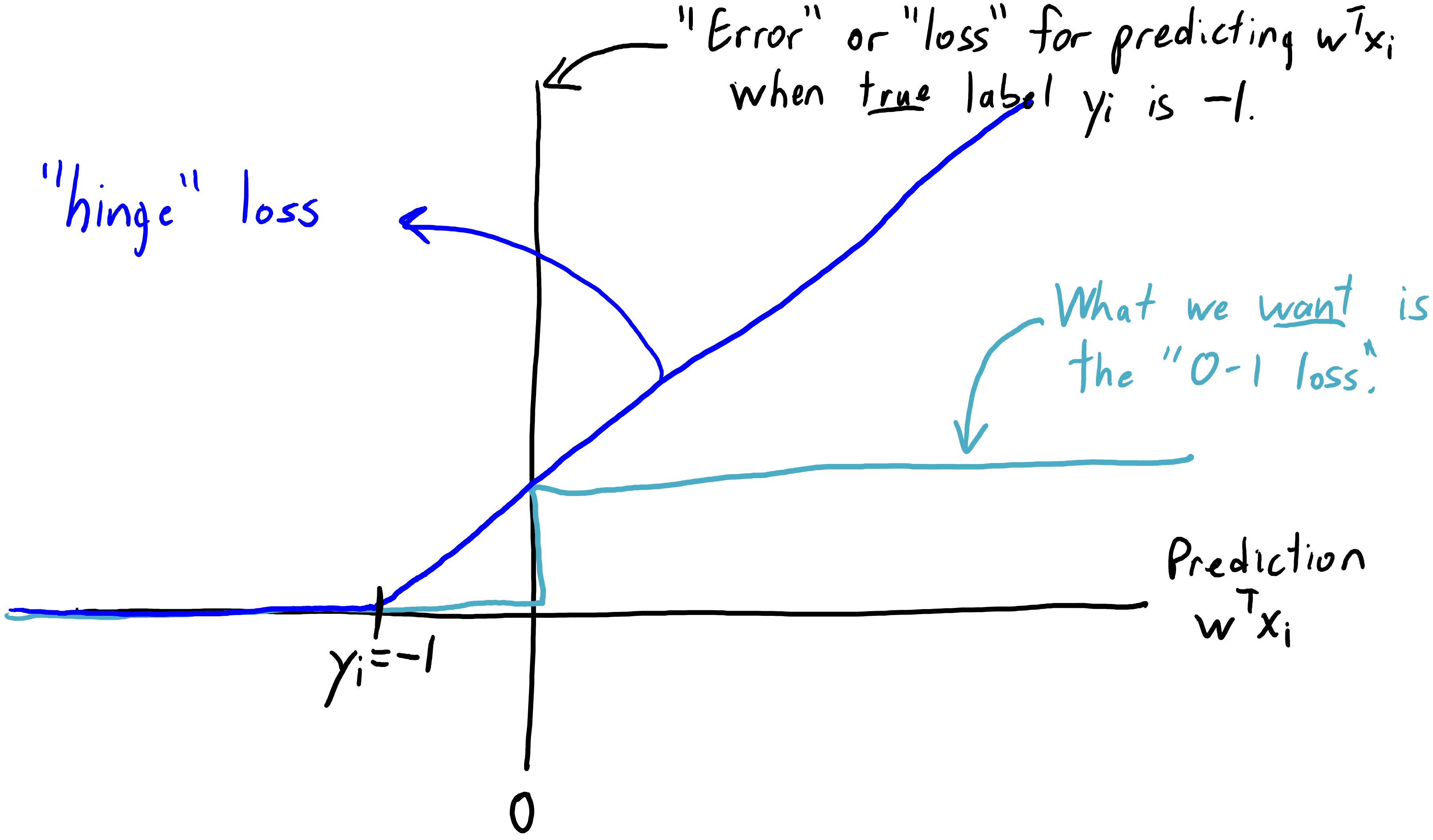
Comparing Loss Functions



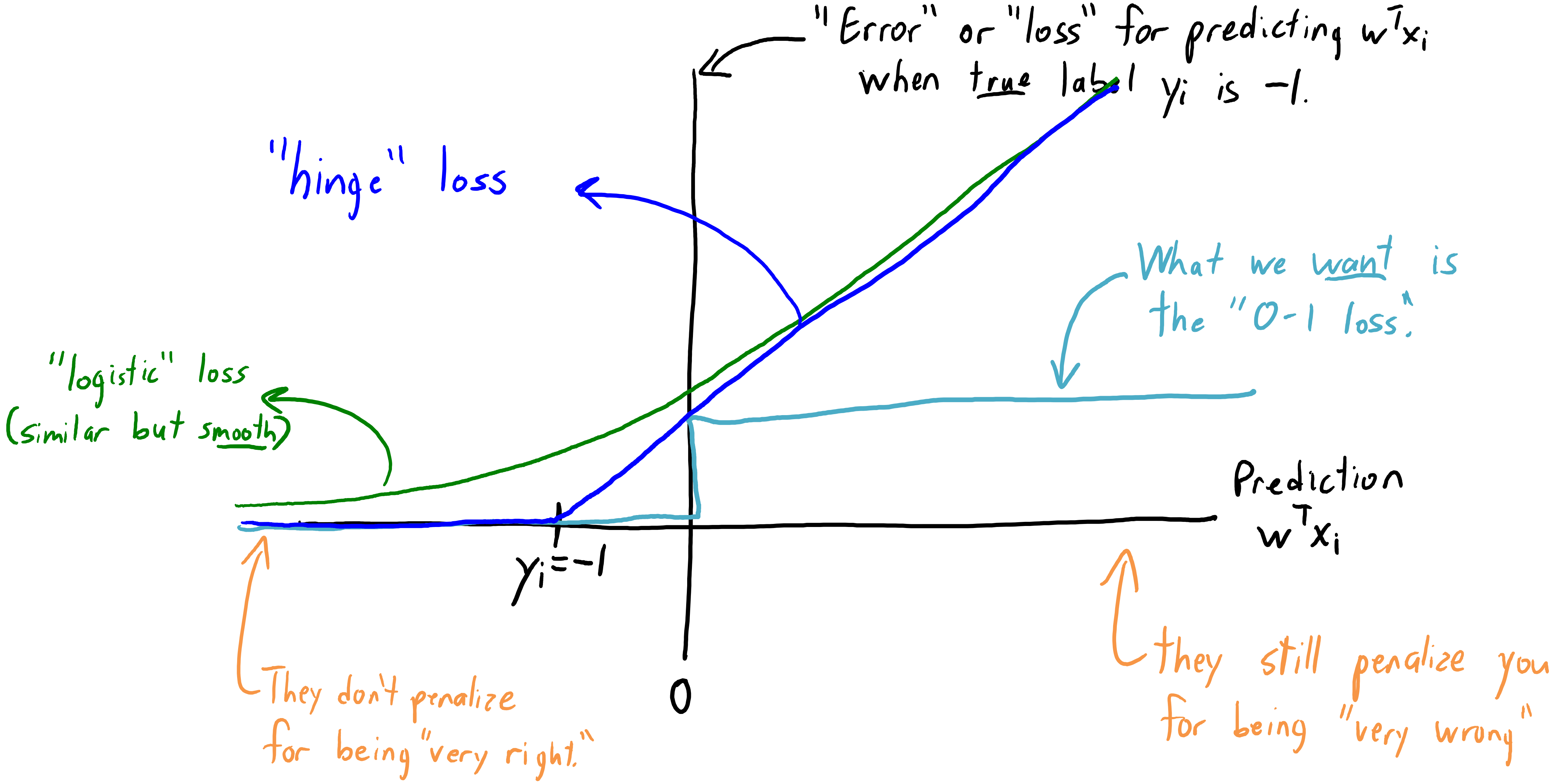
0-1 Loss Function

- The 0-1 loss function is the number of classification errors:
  - Unlike regression, in classification it's reasonable that $\text{sign}(w^T x_i) = y_i$.
- Unfortunately the 0-1 loss is non-convex in 'w'.
  - It's easy to minimize if a perfect classifier exists.
  - Otherwise, finding the 'w' minimizing 0-1 loss is a hard problem.
- Convex approximations to 0-1 loss:
  - Hinge loss (non-smooth) and logistic loss (smooth).

Convex Approximations to 0-1 Loss



Convex Approximations to 0-1 Loss



Hinge Loss and Support Vector Machines

- Hinge loss is given by:

$$f(w) = \sum_{i=1}^{n} \max\{0, 1 - y_i w^T x_i\}$$

  - Convex upper bound on number of classification errors.
  - Solution will be a perfect classifier, if one exists.
- Support vector machine (SVM) is hinge loss with L2-regularization.

$$f(w) = \sum_{i=1}^{n} \max\{0, 1 - y_i w^T x_i\} + \frac{1}{2}\|w\|^2$$

  - Next time we'll see that it "maximizes the margin".

Logistic Regression

- Logistic regression minimizes logistic loss:

$$f(w) = \sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i))$$

- You can/should also add regularization:

$$f(w) = \sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i)) + \frac{\lambda}{2}\|w\|^2$$

- Convex and differentiable: minimize this with gradient descent.

Logistic Regression and SVMs

- Logistic regression and SVMs are used EVERYWHERE!

- Why?
  - 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.

Summary

- Convex functions an be identified using a few simple rules.
- 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.

Bonus Slide: Perceptron Algorithm

- One of the first "learning" is the perceptron algorithm.
  - Searches for a 'w' such that $w^Tx_i > 0$ when $y_i = +1$, $w^Tx_i < 0$ for $y_i=-1$.
- Perceptron Algorithm:
  - Start with $w^0 = 0$.
  - Go through examples in any order until you make a mistake predicting y^i.
    - Set $w^{t+1} = w^t + y_ix_i$.
  - Keep going through examples until you make no errors on training data.
- If a perfect classifier exists, this algorithm converges to one.
  - In fact, "perceptron mistaked bound" result says that number of mistakes is finite.