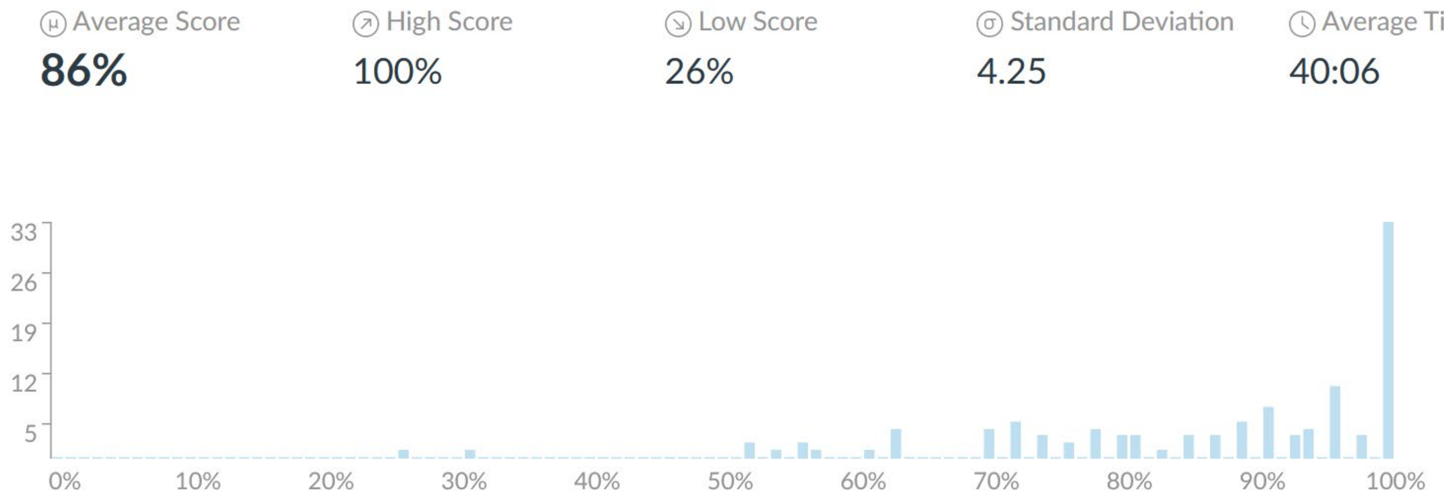
CPSC 340:
Machine Learning and Data Mining

Multi-Class Linear Classifiers

Summer 2021

Admin



Average Score	High Score	Low Score	Standard Deviation	Average Time
86%	100%	26%	4.25	40:06

- Canvas grades released. Good job!
- Assignment 4 due Monday.
  - GO TO TUTORIALS. THEY’RE VERY HELPFUL.
- Assignment 5 out Friday.
- Final exam is Wednesday, June 23, 2021
  - Probably similar format as midterm.

Last Time: SVM and Logistic Regression

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

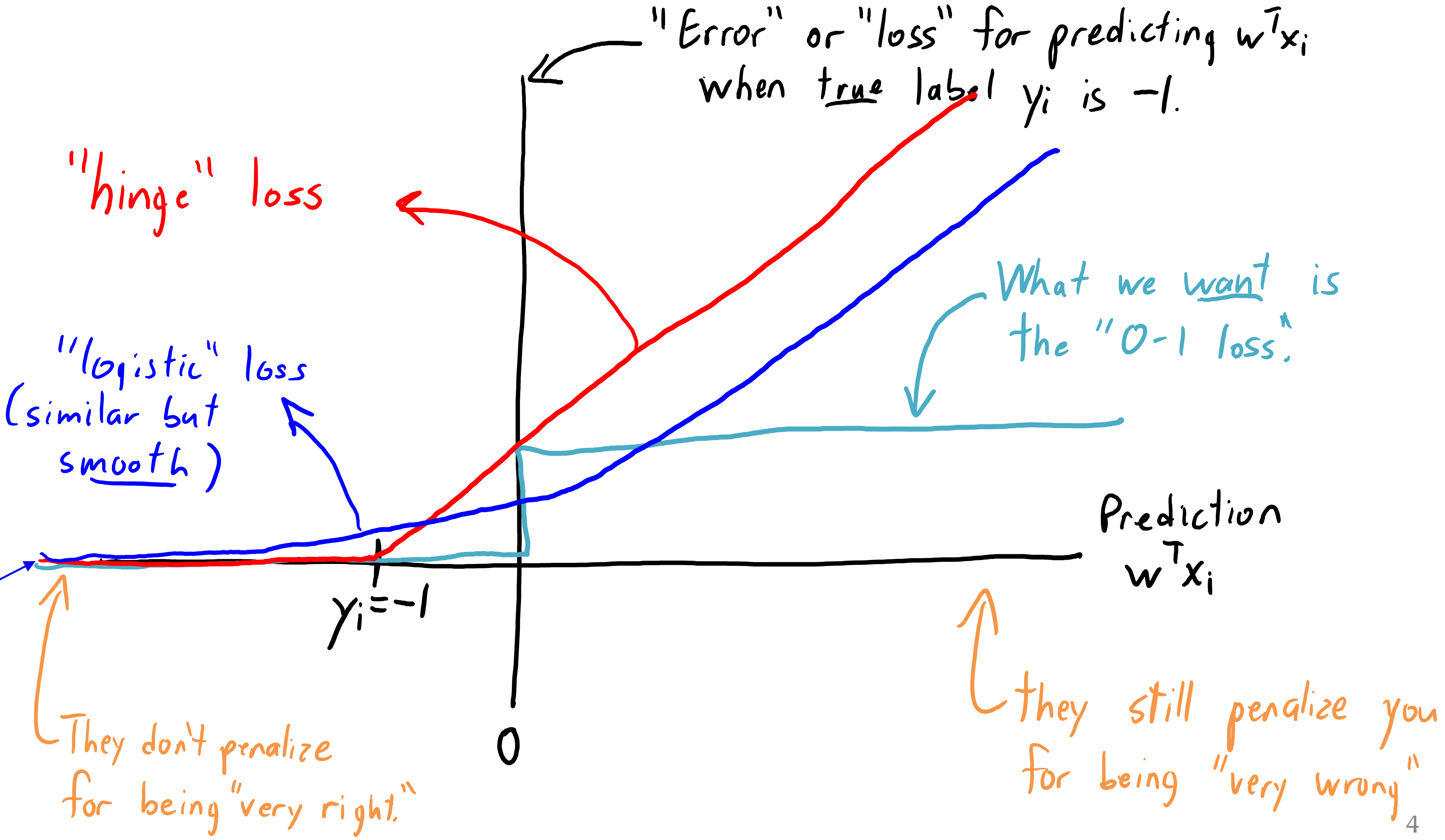
Hinge loss for support vector machine

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

Logistic loss for logistic regression

- We derived these losses step-by-step.
- We will do similar stuff today.

Last Time: SVM and Logistic Regression



They don't penalize for being "very right."

They still penalize you for being "very wrong"


WHY ARE YOU BOOING ME?
I'M RIGHT.

In This Lecture

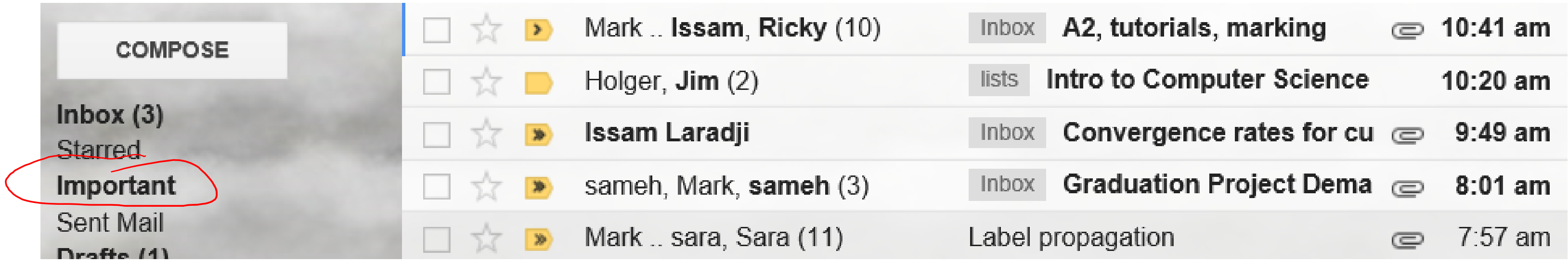
1. Linear Probabilistic Classifiers (10 minutes)
2. Multi-Class Classification Intro (10 minutes)
3. Multi-Class SVM (20 minutes)
4. Multi-Class Logistic Regression (15 minutes)

Coming Up Next

LINEAR PROBABILISTIC CLASSIFIERS

Previously: Identifying Important E-mails

- Recall problem of identifying 'important' e-mails:



Mark .. Issam, Ricky (10)	Inbox	A2, tutorials, marking		10:41 am
Holger, Jim (2)	lists	Intro to Computer Science		10:20 am
Issam Laradji	Inbox	Convergence rates for cu		9:49 am
sameh, Mark, sameh (3)	Inbox	Graduation Project Dema		8:01 am
Mark .. sara, Sara (11)		Label propagation		7:57 am

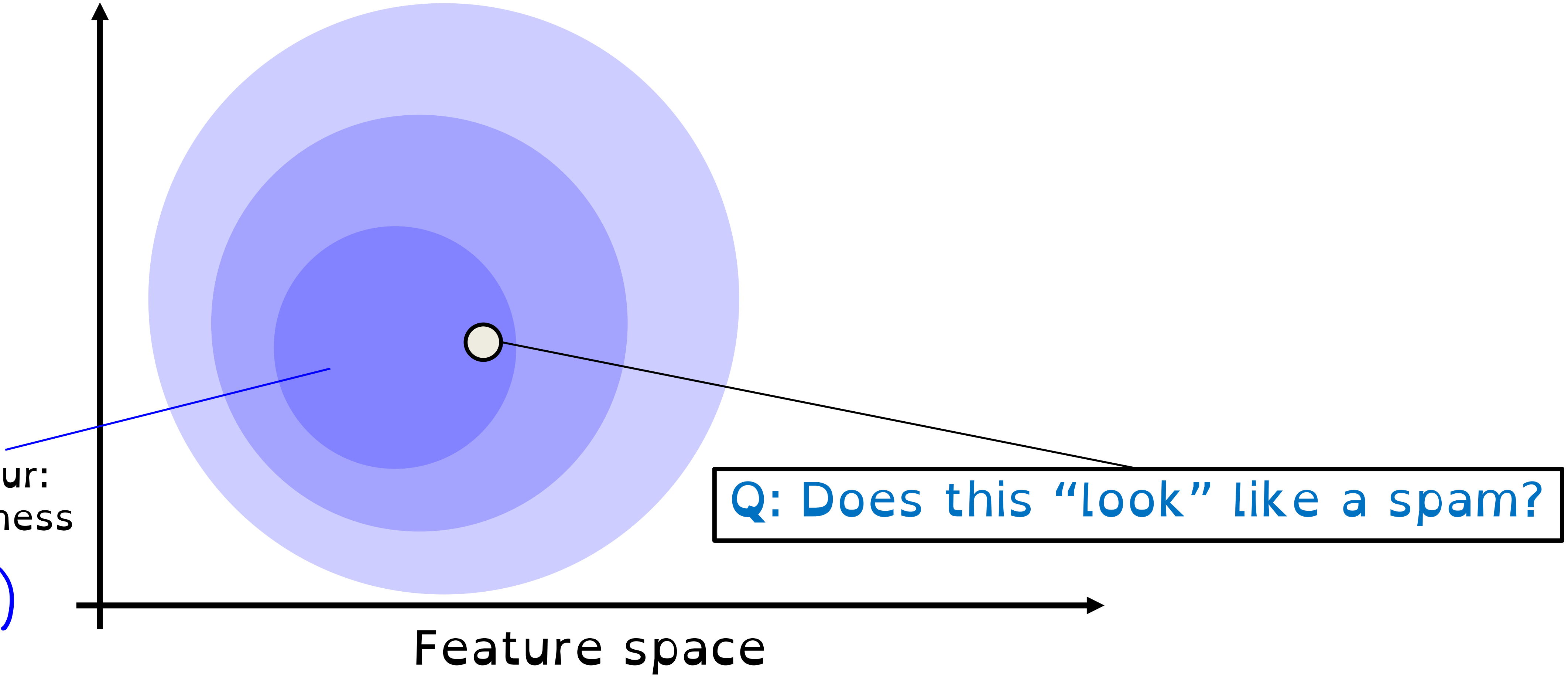
- We can do binary classification by taking sign of linear model:

$$\hat{y}_i = \text{sign}(w^T x_i)$$

  - Convex loss functions (hinge/logistic loss) let us find an appropriate 'w'.

- But what if we want a probabilistic classifier?
  - Want a model of $p(y_i = \text{"important"} \mid x_i)$ for use in decision theory.

Recall: “Spam-ness”

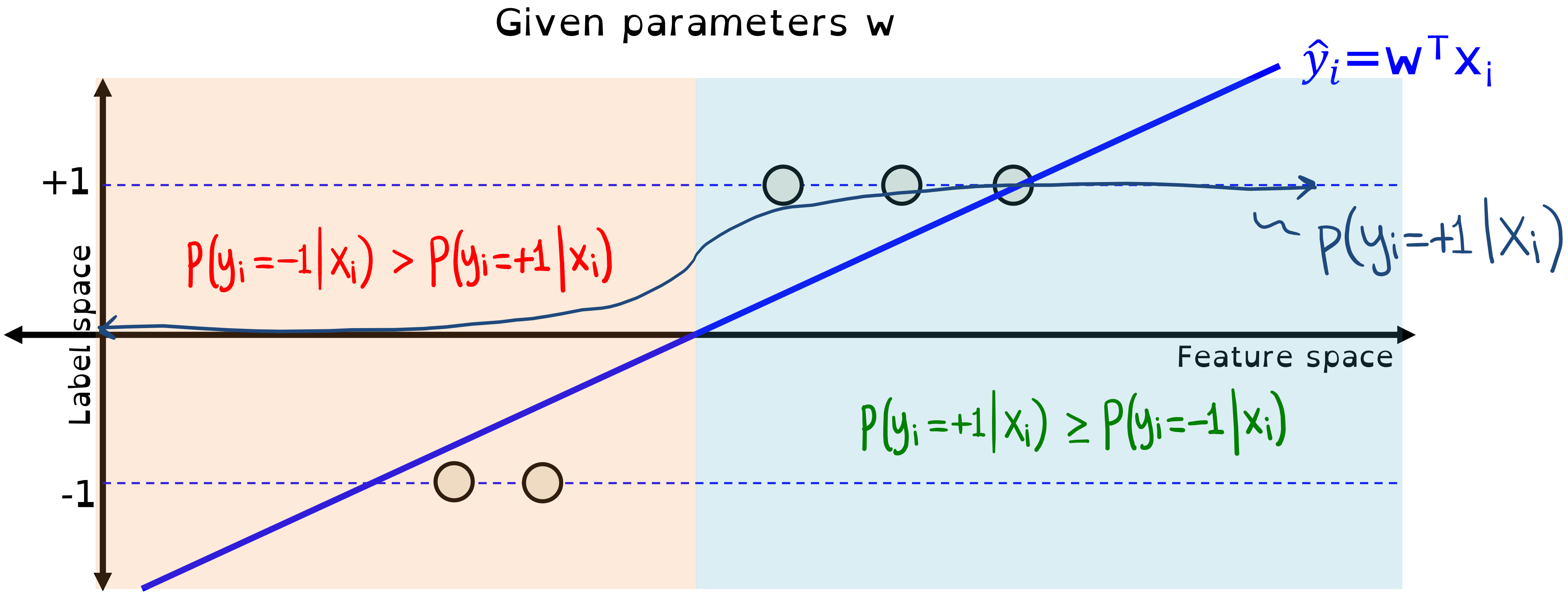


Thicker colour:
Higher spam-ness

$P(y_i = \text{"spam"} \mid X_i)$

Q: Does this “look” like a spam?

Linear Prediction of “+1-ness”



Q: How should $p(y_i=+1 \mid x_i)$ behave?

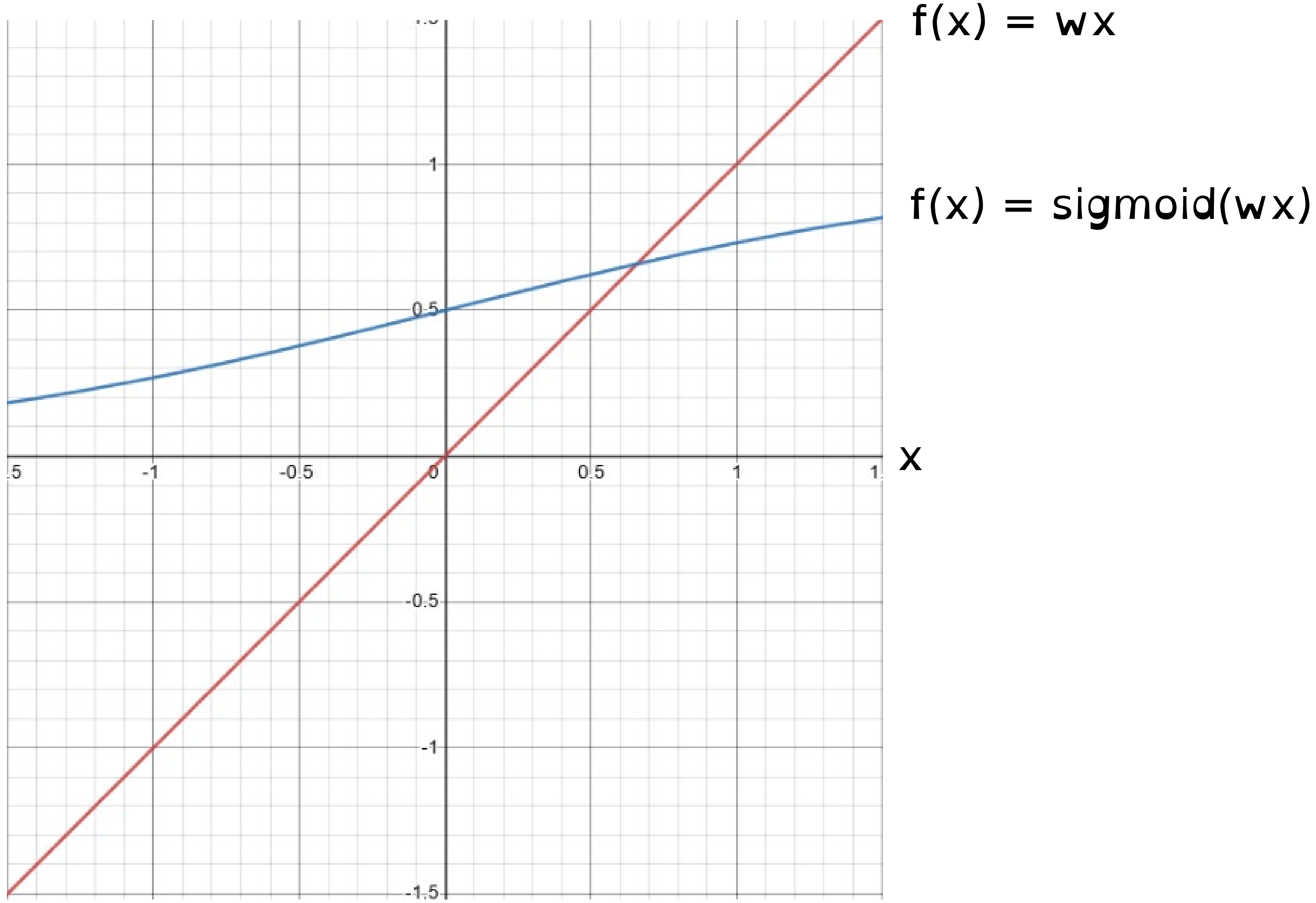
Sigmoid Function

$\text{Sigmoid}: \mathbb{R} \rightarrow (0, 1)$

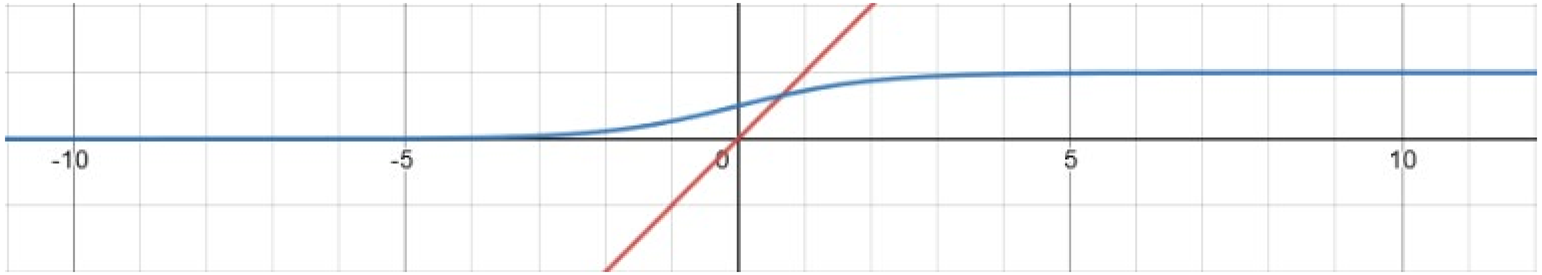
$$\text{Sigmoid}(z) = \frac{1}{1+\exp(-z)}$$

Q: What is sigmoid(z) when z is negative?

What is sigmoid(z) when z is positive?



+1-ness with Sigmoid



f(x) = sigmoid(wx)

- Idea: let’s compute +1-ness with sigmoid.
- Given parameters **w**:
  1. Compute $z_i = \mathbf{w}^T x_i$
  2. Compute $p(y_i = +1 \mid \mathbf{w}, x_i) = \text{sigmoid}(z_i)$

$$P(y_i=+1 \mid w, \cdot) : \mathbb{R}^d \longrightarrow (0,1)$$

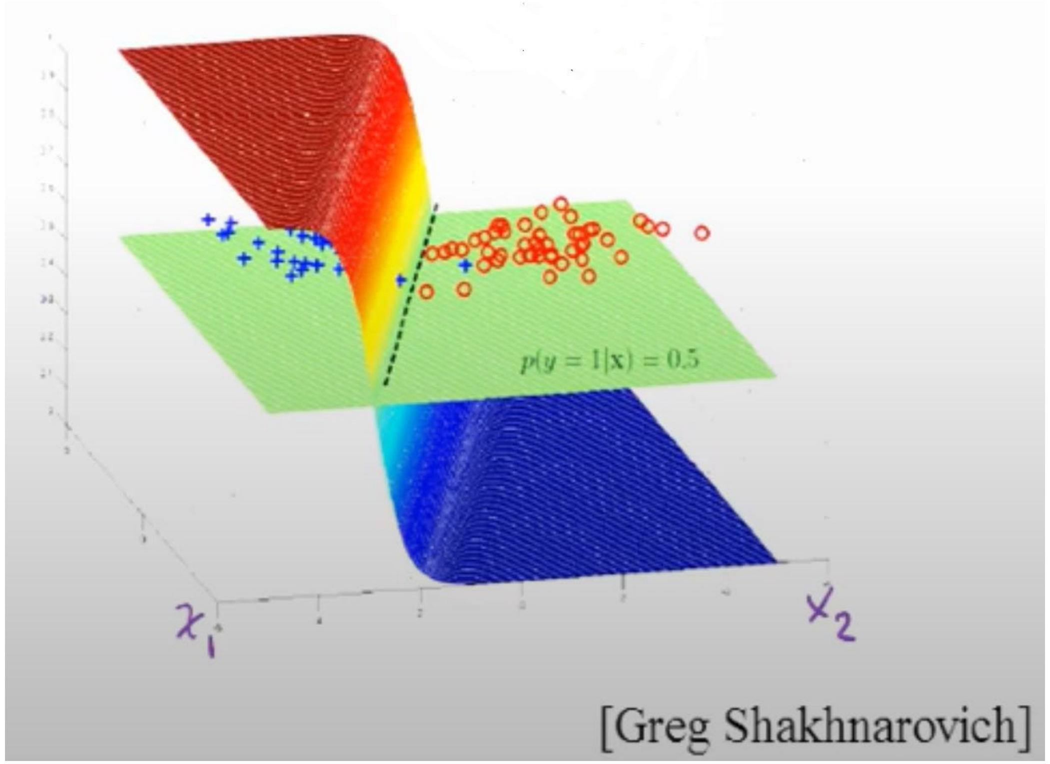
$$p(y_i=+1 \mid w, x_i) = \text{sigmoid}(w^T x_i)$$

Probabilities for Linear Classifiers using Sigmoid

- Using sigmoid function, we output probabilities for linear models using:

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

- Visualization for 2 features:



[Greg Shakhnarovich]

https://www.youtube.com/watch?v=Zc7ouSD0DEQ

What About "-1-ness"?

- Using sigmoid function, we output probabilities for linear models using:

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

- By rules of probability:

$$p(y_i = -1 \mid w, x_i) = 1 - p(y_i = 1 \mid w, x_i)$$

$$= \frac{1}{1 + \exp(w^T x_i)}$$

(with some effort)

- We then use these for "probability that an email x_i is important".
- This may seem heuristic, but later we'll see that:
  - minimizing logistic loss does "maximum likelihood estimation" in this model.

Coming Up Next

MULTI-CLASS CLASSIFICATION INTRO

Multi-Class Linear Classification

- We've been considering linear models for binary classification:

$$X = \begin{bmatrix} \ \end{bmatrix} \quad y = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

- E.g., is there a cat in this image or not?

https://www.youtube.com/watch?v=tntOCGkgt98

Multi-Class Linear Classification

- Today we'll discuss linear models for multi-class classification:

$$X = \begin{bmatrix} \quad \\ \quad \\ \quad \end{bmatrix} \qquad y = \begin{bmatrix} 27 \\ 16 \\ 8 \\ 7 \\ 21 \\ 5 \end{bmatrix}$$

- For example, classify image as "cat", "dog", or "person".
  - This was natural for methods of Part 1 (decision trees, naïve Bayes, KNN).
  - For linear models, we need some new notation.

Q3,2

Q: Can we use binary classifiers for multi-class?

"One vs All" Classification

- Training phase:
  - For each class 'c', train binary classifier to predict whether example is a 'c'.
    - For example, train a "cat detector", a "dog detector", and a "human detector".
    - If we have 'k' classes, this gives 'k' binary classifiers .

$$X = {}_{n}\begin{bmatrix} \quad & \quad & \quad \\ \\ \\ \end{bmatrix}_{d} \qquad y = {}_{n}\begin{bmatrix}\text{"cat"}\\ \text{"dog"}\\ \text{"human"}\\ \text{"cat"}\end{bmatrix}_{1} \longrightarrow y_{cat} = \begin{bmatrix}+1\\ -1\\ -1\\ +1\end{bmatrix}$$

$$(X, y_{cat}) \rightarrow w_{cat} \qquad (X, y_{dog}) \rightarrow w_{dog} \qquad (X, y_{human}) \rightarrow w_{human}$$

"cat detector"

"One vs All" Classification

$$W = {}_{k}\underset{d}{\begin{bmatrix} — \; w_{cat} \; — \\ — \; w_{dog} \; — \\ \\ — \; w_{human} \; — \end{bmatrix}}$$

#classes

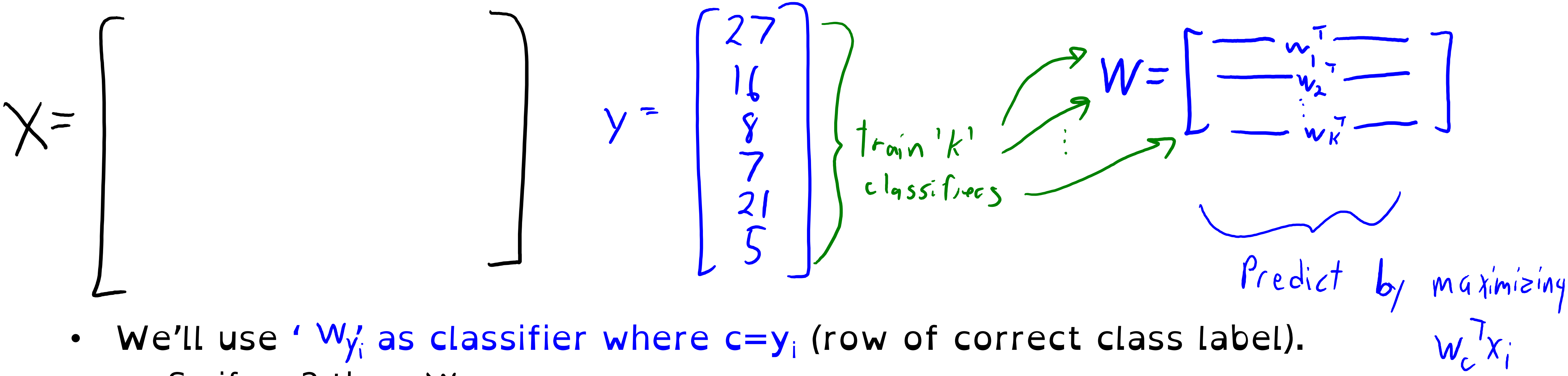
- Prediction phase:
  - Apply the 'k' binary classifiers to get a "score" for each class 'c'.
  - Predict the 'c' with the highest score.

features x_i

$$\left.\begin{array}{l} w_{cat}^T x_i = -0.1 \\ w_{dog}^T x_i = -0.8 \\ w_{human}^T x_i = 0.9 \end{array}\right\} \hat{y}_i = \text{human}$$

Multi-Class Linear Classification (MEMORIZE)

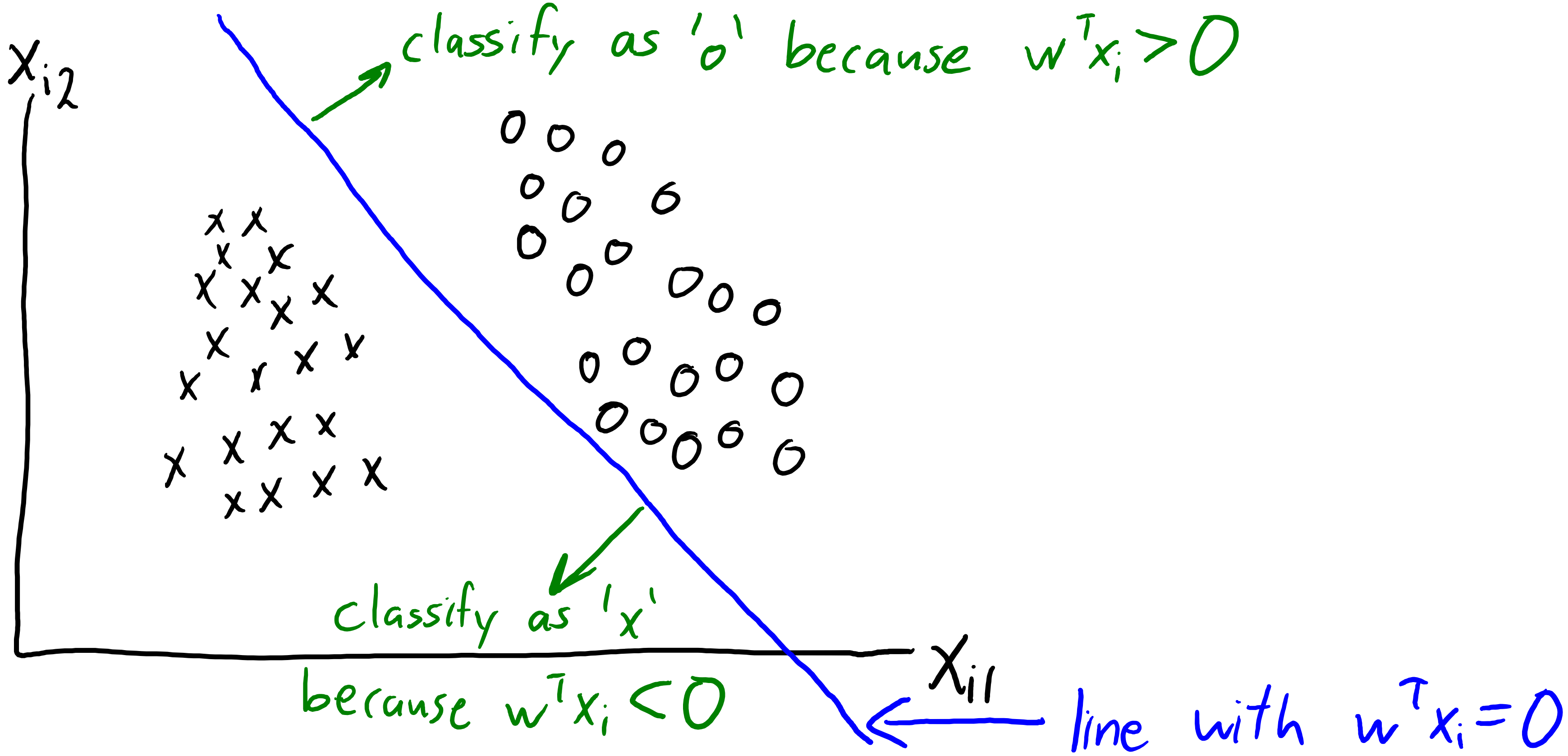
- Back to multi-class classification where we have 1 “correct” label:



- We’ll use ‘ w_{y_i}’ as classifier where $c=y_i$ (row of correct class label).
  - So if $y_i=2$ then $w_{y_i} = w_2$.

Shape of Decision Boundaries

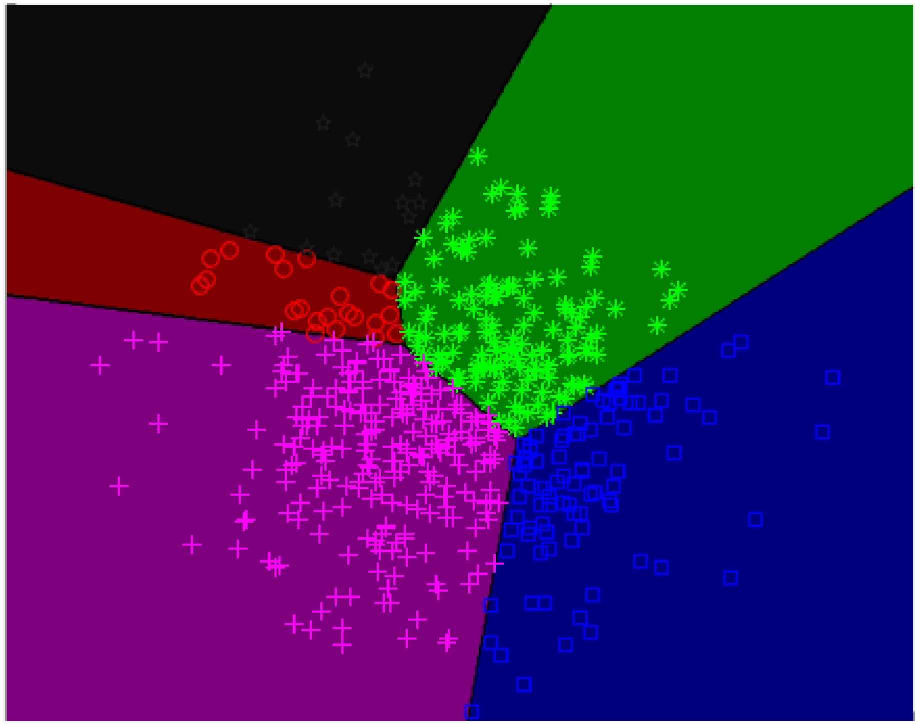
- Recall that a binary linear classifier splits space using a hyper-plane:



- Divides x_i space into 2 "half-spaces".

Shape of Decision Boundaries

- Multi-class linear classifier is intersection of these "half-spaces":
  - This divides the space into convex regions (like k-means):



"Blue" region is region where we have:

$w_{blue}^T x_i \geq w_{green}^T x_i$

$w_{blue}^T x_i \geq w_{magenta}^T x_i$

$w_{blue}^T x_i \geq w_{red}^T x_i$

$w_{blue}^T x_i \geq w_{black}^T x_i$

  - Could be non-convex with change of basis.

Digression: Multi-Label Classification

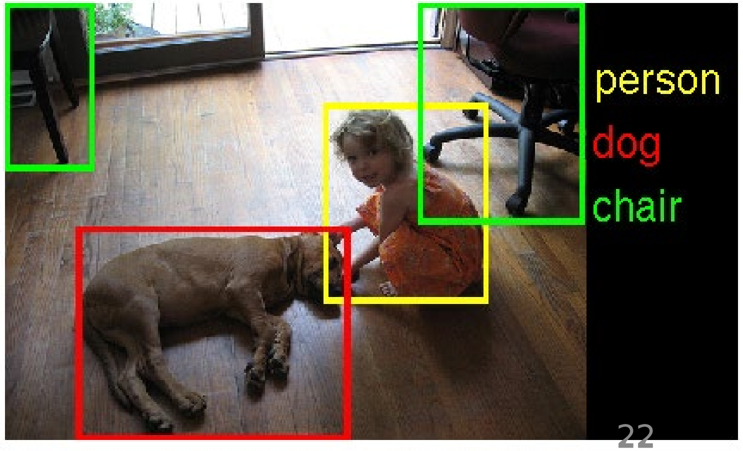
- A related problem is multi-label classification:
-

$$X = \underbrace{\left.\left[\begin{matrix} \quad & \quad & \quad \\ & & \\ & & \\ & & \end{matrix}\right]\right\} n}_{d}$$

$$Y = \underbrace{\left.\left[\begin{array}{ccccc} \text{cat} & \text{dog} & \text{person} & \text{chair} & \text{mouse} \\ 1 & -1 & 1 & 1 & 1 \\ -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 \\ 1 & 1 & -1 & 1 & -1 \\ -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 \end{array}\right]\right\} n}_{k}$$

$$W = \underbrace{\left.\left[\begin{matrix} — w_1^T — \\ — w_2^T — \\ \vdots \\ — w_k^T — \end{matrix}\right]\right\} k}_{d}$$

- Which of the ‘k’ objects are in this image?
  - There may be more than one “correct” class label.
  - Here we can also fit ‘k’ binary classifiers.
    - But we would take all the $\text{sign}(w_c^T x_i) = +1$ as the labels.



http://image-net.org/challenges/LSVRC/2013/

“One vs All” Multi-Class Linear Classification

- Problem: We didn’t train the w_c so that the largest $w_c^T x_i$ would be $w_{y_i}^T x_i$.
  - Each classifier is just trying to get the sign right.

features x_i

$w_1^T x_i = -5$ ("cat" score)

$w_2^T x_i = -0.1$ ("dog" score)

$w_3^T x_i = -0.2$ ("human" score)

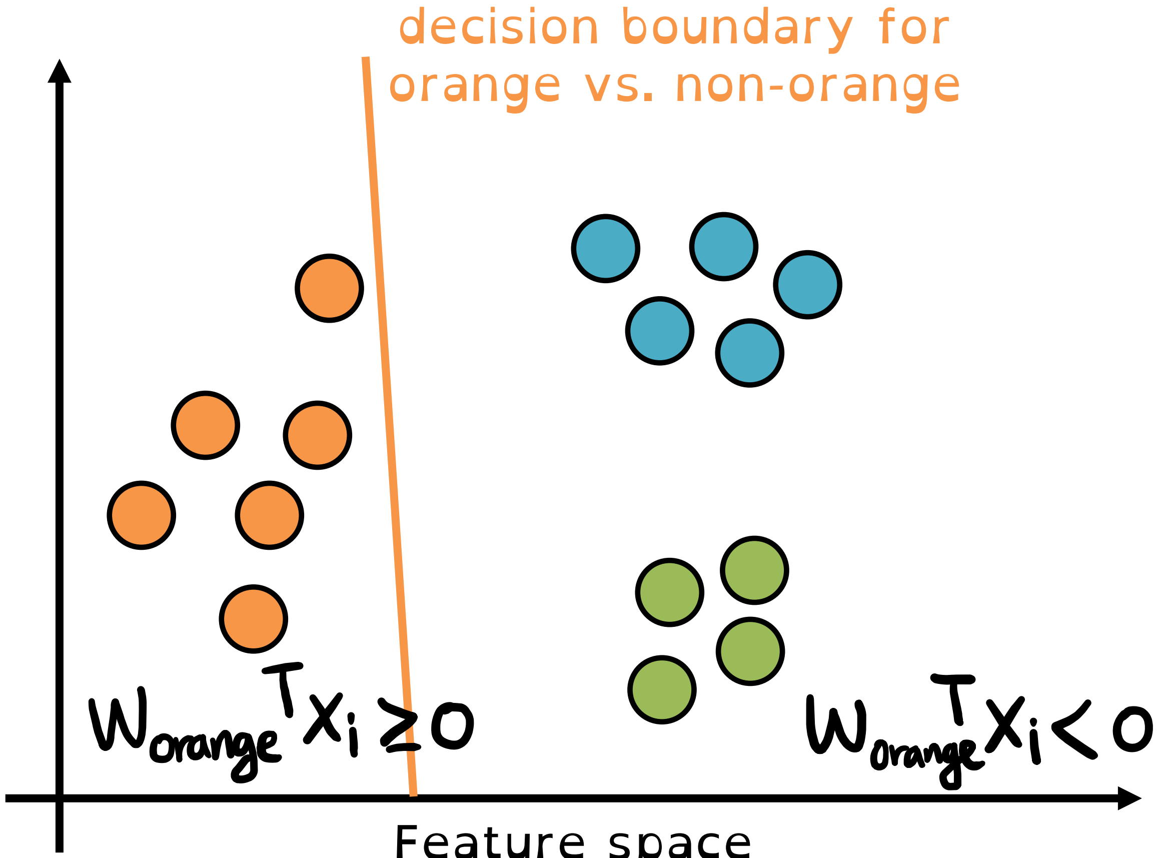
  - Here the classifier incorrectly predicts “dog”.
    - “One vs All” doesn’t try to put $w_2^T x_i$ and $w_3^T x_i$ on same scale for decisions like this.
    - We should try to make $w_3^T x_i$ positive and $w_2^T x_i$ negative relative to each other.
    - The multi-class hinge losses and the multi-class logistic loss do this.

https://laughingsquid.com/pug-mask-a-latex-mask-so-you-can-look-like-a-dog/

Coming Up Next

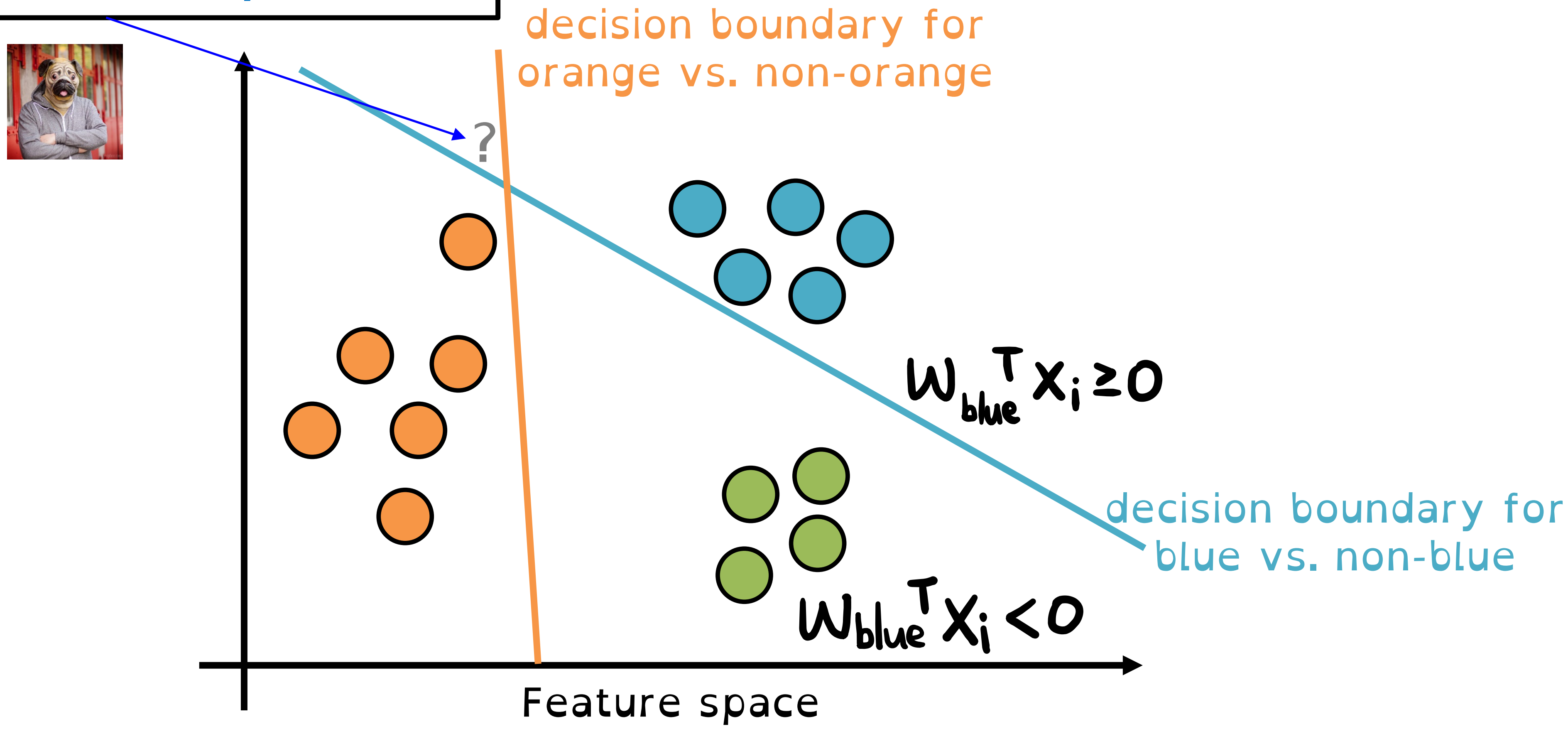
MULTI-CLASS SVM

Binary Classifiers are “Under-constrained”



Binary Classifiers are “Under-constrained”

Q: What’s the prediction for this example?



decision boundary for orange vs. non-orange

?

$W_{blue}^T x_i \geq 0$

$W_{blue}^T x_i < 0$

decision boundary for blue vs. non-blue

Feature space

What Do We Want for Multi-Class Classifiers?

- Idea: additional constraints on slopes
  - Will “force” classifier to find better boundaries

- Think of $w_{cat}^T x_i$, $w_{dog}^T x_i$, $w_{human}^T x_i$ as scores.
  - Previously, we only wanted $w_{cat}^T x_i > 0$ (underconstrained!)
  - New constraint: “cat” example x_i should have higher $w_{cat}^T x_i$ than $w_{dog}^T x_i$, $w_{human}^T x_i$
  - Now, we want: $w_{cat}^T x_i > w_{dog}^T x_i$ and $w_{cat}^T x_i > w_{human}^T x_i$

Q: How should we design the error here?

Multi-Class Loss Function

Now, we want: $w_{cat}^T x_i > w_{dog}^T x_i$ and $w_{cat}^T x_i > w_{human}^T x_i$

Let's count # times $W_{cat}^T x_i \leq W_{dog}^T x_i$ $+$ # $W_{cat}^T x_i \leq W^T_{human} x_i$

[1] $$f_{cat}(W) = \sum_{i \in \text{cat examples}} I\left(W_{cat}^T x_i \leq W_{dog}^T x_i\right) + I\left(W_{cat}^T x_i \leq W_{human}^T x_i\right)$$

[2] $$= \sum_{i \in \text{cat examples}} I\left(0 \leq -W_{cat}^T x_i + W_{dog}^T x_i\right) + I\left(0 \leq -W_{cat}^T x_i + W_{human}^T x_i\right)$$

[3] $$\approx \sum_{i \in \text{cat examples}} \max\left\{0, -W_{cat}^T x_i + W_{dog}^T x_i\right\} + \max\left\{0, -W_{cat}^T x_i + W_{human}^T x_i\right\}$$

Multi-Class Loss Function

- Let's generalize this!

[4] $$f_{cat}(W) = \sum_{i \in \text{cat examples}} \max\{0, -W_{cat}^T x_i + W_{dog}^T x_i\} + \max\{0, -W_{cat}^T x_i + W_{human}^T x_i\}$$

[5] $$f_c(W) = \sum_{i \in \text{"c" examples}} \sum_{c' \neq c} \max\{0, -W_c^T x_i + W_{c'} x_i\}$$

[6] $$f(W) = \sum_{c=1}^{k} f_c(W) = \sum_{c=1}^{k} \sum_{i \in \text{"c" examples}} \sum_{c' \neq c} \max\{0, -W_c^T x_i + W_{c'} x_i\}$$

[7] $$= \sum_{i=1}^{n} \sum_{c' \neq y_i} \max\{0, -W_{y_i}^T x_i + W_{c'} x_i\}$$

Multi-Class Loss Function

$$f(W) = \sum_{i=1}^{n} \sum_{c' \neq y_i} \max\{0, -W_{y_i}^T X_i + W_{c'} X_i\}$$

$$X = {}_{n}\underset{d}{\left[\begin{array}{c} \\ \\ \\ - x_i - \end{array}\right]} \leftarrow i \qquad y = {}_{n}\underset{1}{\left[\begin{array}{c} \text{"cat"} \\ \text{"dog"} \\ \text{"human"} \\ \text{"cat"} \end{array}\right]} \leftarrow i$$

$$W = {}_{k}\underset{d}{\left[\begin{array}{c} - W_{cat} - \\ - W_{dog} - \\ \\ - W_{human} - \end{array}\right]} \qquad W_{y_i} \rightarrow W_{cat}$$

$$= \max\{0, -W_{cat}^T X_i + W_{dog}^T X_i\} + \max\{0, -W_{cat}^T X_i + W_{human}^T X_i\}$$

Multi-Class Hinge Loss

$$f(W) = \sum_{i=1}^{n} \sum_{c' \neq y_i} \max\{0, \underbrace{-w_{y_i}^T x_i}_{0} + \underbrace{w_{c'} x_i}_{0}\}$$

- This function is degenerate: $f(0) = \underline{\;0\;}$.
- As with binary SVM, we introduce an offset

$$f(W) = \sum_{i=1}^{n} \sum_{c' \neq y_i} \max\{0, 1 - w_{y_i}^T x_i + w_{c'} x_i\}$$

"sum"-rule multi-class hinge loss

$$f(W) = \sum_{i=1}^{n} \max_{c' \neq y_i}\left[\max\{0, 1 - w_{y_i}^T x_i + w_{c'} x_i\}\right]$$

"max"-rule multi-class hinge loss

Multi-Class SVMs

- Idea: for a cat example, we want: $w_{cat}^T x_i > w_{dog}^T x_i$ and $w_{cat}^T x_i > w_{human}^T x_i$

$$f(W) = \sum_{i=1}^{n} \sum_{c' \neq y_i} \max\{0, 1 - w_{y_i}^T x_i + w_{c'} x_i\}$$

$$f(W) = \sum_{i=1}^{n} \max_{c' \neq y_i} \{\max\{0, 1 - w_{y_i}^T x_i + w_{c'} x_i\}\}$$

- For each training example 'i':
  - "Sum" rule penalizes for each 'c' that violates the constraint.
  - "Max" rule penalizes for one 'c' that violates the constraint the most.
    - "Sum" gives a penalty of 'k-1' for W=0, "max" gives a penalty of '1'.
- If we add L2-regularization, both are called multi-class SVMs:
  - "Max" rule is more popular, "sum" rule usually works better.
  - Both are convex upper bounds on the 0-1 loss.

Coming Up Next

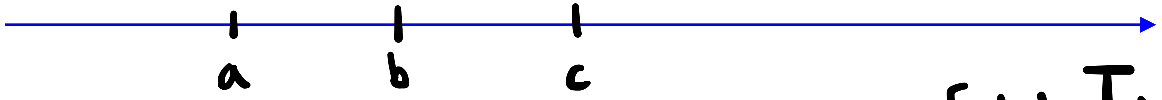
MULTI-CLASS LOGISTIC REGRESSION

Multi-Class Logistic Regression

- Idea: for a cat example, we want: $\left(w_{cat}^T x_i > w_{dog}^T x_i \text{ and } w_{cat}^T x_i > w_{human}^T x_i\right)$
  - In other words: **we want** $w_{cat}^T x_i$ **to be max** $w_c^T x_i$

$f(W) =$ # times this is not true across all "cat" examples.

$c > a \quad c > b \quad c = \max\{a, b, c\}$



$$\left.\begin{matrix} W_{cat}^T X_i > W_{dog}^T X_i \\ W_{cat}^T X_i > W_{human}^T X_i \end{matrix}\right\} \Leftrightarrow W_{cat}^T X_i = \max\left\{\begin{matrix} W_{cat}^T X_i \\ W_{dog}^T X_i \\ W_{human}^T X_i \end{matrix}\right\} = \sum_{i \in \text{"cat" examples}} I(\quad)$$

$$\text{Count \#times } W_{cat}^T X_i < \max\left\{\begin{matrix} W_{cat}^T X_i \\ W_{dog}^T X_i \\ W_{human}^T X_i \end{matrix}\right\} \text{ or } 0 < \left[\underbrace{-W_{cat}^T X_i}_{} + \max\left\{\begin{matrix} w_{cat}^T x_i \\ w_{dog}^T x_i \\ w_{human}^T x_i \end{matrix}\right\}\right]$$

$\boxed{\text{Q: What happens when } W=0\text{?}}$

Multi-Class Logistic Regression

$$-w_{cat}^T x_i + \max\begin{Bmatrix} w_{cat}^T x_i \\ w_{dog}^T x_i \\ w_{human}^T x_i \end{Bmatrix} \approx -w_{cat}^T x_i + \log\begin{pmatrix} \exp(w_{cat}^T x_i) \\ + \\ \exp(w_{dog}^T x_i) \\ + \\ \exp(w_{human}^T x_i) \end{pmatrix}$$

log-sum-exp

$$\underbrace{\nabla f(W)}_{d \times 1} = \sum_{i=1}^{n} (\cdot)$$

$f(W) =$

$$\sum_{i \in \text{"cat" examples}} (\cdot)$$

- Ideas:
  1. use log-sum-exp to approximate max
  2. instead of counting number of times this quantity is positive, use this quantity as objective function

Q: What happens when W=0?

$f(0) = \log(k)$

Multi-Class Logistic Regression

$$f(W) = \sum_{i=1}^{n} -w_{y_i}^T x_i + \log\left(\sum_{c=1}^{k} \exp(w_c^T x_i)\right)$$

"Softmax loss"

$$f(W) = \sum_{i=1}^{n} -w_{y_i}^T x_i + \log\left(\sum_{c=1}^{k} \exp(w_c^T x_i)\right) + \frac{\lambda}{2}\sum_{c=1}^{k}\sum_{j=1}^{d} w_{cj}^2$$

L2-regularized softmax loss

gradient is associated with softmax probability.

- Multi-class (multinomial) logistic regression: optimize W with L2-regularized softmax loss

Multi-Class Logistic Regression

$$f(W) = \sum_{i=1}^{n}\Big[\underbrace{-w_{y_i}^T x_i}_{\text{Tries to make } w_c^T x_i \text{ big for the correct label}} + \underbrace{\log\Big(\sum_{c=1}^{k} \exp(w_c^T x_i)\Big)}_{\text{Approximates } \max_c \{w_c^T x_i\} \text{ so tries to make } w_c^T x_i \text{ small for all labels.}}\Big] + \underbrace{\frac{\lambda}{2}\sum_{c=1}^{k}\sum_{j=1}^{d} w_{cj}^2}_{\text{Usual } L_2\text{-regularizer on elements of 'W'}}$$

- This objective is convex (should be clear for 1st and 3rd terms).
  - It’s differentiable so you can use gradient descent.
- When k=2, equivalent to using binary logistic loss.
  - Not obvious at the moment.

Softmax Function

$$\mathrm{Softmax} : \mathbb{R} \times \mathbb{R}^{K} \to (0, 1)$$

$$\mathrm{Softmax}\left(\underbrace{z, \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_K \end{bmatrix}}_{z \in \begin{bmatrix} z_1 \\ \vdots \\ z_K \end{bmatrix}}\right) = \frac{\exp(z)}{\sum_{c=1}^{K} \exp(z_c)}$$

- Evaluates “max-ness” of z compared to group
  - if z is big compared to every other z_c, softmax is close to 1

"Max-ness" to Find "Cat-ness"

$$P(y_i = \text{"cat"} \mid W, \cdot) : \mathbb{R}^d \longrightarrow (0, 1)$$

$$P(y_i = \text{"cat"} \mid W, x_i) = \text{Softmax}\left(w_{cat}^T x_i, \begin{bmatrix} w_{cat}^T x_i \\ w_{dog}^T x_i \\ w_{human}^T x_i \end{bmatrix}\right)$$

"cat-ness"

"How big is my cat score compared to my dog score and human score?"

$$\left.\begin{array}{l} \nearrow w_{cat}^T x_i = 1.83 \\ \rightarrow w_{dog}^T x_i = -1.17 \\ \searrow w_{human}^T x_i = -2.20 \end{array}\right\} \begin{array}{l} \text{cat-ness} = \sim 93.6\% \\ \text{dog-ness} = \sim 0.04\% \\ \text{human-ness} = \sim 0.02\% \end{array}$$

Multi-Class Linear Prediction in Matrix Notation

- In multi-class linear classifiers our weights are:

$$\underset{k\times d}{W} = \underbrace{\begin{bmatrix} — & w_1^T & — \\ — & w_2^T & — \\ & \vdots & \\ — & w_k^T & — \end{bmatrix}}_{d} \left.\vphantom{\begin{bmatrix} a \\ a \\ a \\ a \end{bmatrix}}\right\} k$$

- To predict on all training examples, we first compute all $w_c^T x_i$.
  - Or in matrix notation:

$$n\left\{\vphantom{\begin{bmatrix} a \\ a \\ a \\ a \end{bmatrix}}\right. \underbrace{\begin{bmatrix} w_1^Tx_1 & w_2^Tx_1 & \cdots & w_k^Tx_1 \\ w_1^Tx_2 & w_2^Tx_2 & \cdots & w_k^Tx_2 \\ & \vdots & & \\ w_1^Tx_n & w_2^Tx_2 & \cdots & w_k^Tx_n \end{bmatrix}}_{k \quad XW^T} = \underbrace{\begin{bmatrix} — & x_1^T & — \\ — & x_2^T & — \\ & \vdots & \\ — & x_n^T & — \end{bmatrix}}_{\underset{n\times d}{X}} \underbrace{\begin{bmatrix} | & | & & | \\ w_1 & w_2 & \cdots & w_k \\ | & | & & | \end{bmatrix}}_{\underset{d\times k}{W^T}}$$

  - So predictions are maximum column indices of XW^T (which is 'n' by 'k').

How Do I Regularize W?

- The Frobenius norm of a ('k' by 'd') matrix 'W' is defined by:

$$\|W\|_F = \sqrt{\sum_{c=1}^{k}\sum_{j=1}^{d} w_{jc}^2}$$

$$\underset{k \times d}{W} \rightarrow \underset{kd \times 1}{w} \rightarrow \underset{w^Tw}{\|w\|_2} = \|W\|_F$$

(L$_2$-norm if you "stack" elements into one big vector)

- We can use this to write regularizer in matrix notation:

$$\frac{\lambda}{2}\sum_{c=1}^{k}\sum_{j=1}^{d} w_{cj}^2 = \frac{\lambda}{2}\sum_{c=1}^{k}\|w_c\|^2$$

("L$_2$-regularizer on each vector")

$$= \frac{\lambda}{2}\|W\|_F^2$$

("Frobenius-regularizer on matrix")

Summary

- Sigmoid function: turn linear predictions into probabilities.
- One vs all: turn binary classifiers into a multi-class classifier.
- Multi-class SVM: measure violation of classification constraints.
- Multi-class logistic regression: log-sum-exp approximation of “correct=max” constraint violations
- Next time: feature engineering and how to represent text data

Review Questions

- Q1: How does the sigmoid function satisfy the definition of probability?
- Q2: What makes the one-vs-all classifier under-constrained?
- Q3: Mathematically, what leads to the “sum”-rule and “max”-rule variants of multi-class SVM?
- Q4: How do we know that the objective functions for multi-class SVM and multi-class logistic regression are convex?

“All-Pairs” and ECOC Classification

- Alternative to “one vs. all” to convert binary classifier to multi-class is “all pairs”.
  - For each pair of labels ‘c’ and ‘d’, fit a classifier that predicts +1 for examples of class ‘c’ and -1 for examples of class ‘d’ (so each classifier only trains on examples from two classes).
  - To make prediction, take a vote of how many of the (k-1) classifiers for class ‘c’ predict +1.
  - Often works better than “one vs. all”, but not so fun for large ‘k’.

- A variation on this is using “error correcting output codes” from information theory (see Math 342).
  - Each classifier trains to predict +1 for some of the classes and -1 for others.
  - You setup the +1/-1 code so that it has an “error correcting” property.
    - It will make the right decision even if some of the classifiers are wrong.

Motivation: Dog Image Classification

- Suppose we're classifying images of dogs into breeds:

- What if we have images where class label isn't obvious?
  - Siberian husky vs. Inuit dog?

https://www.slideshare.net/angjoo/dog-breed-classification-using-part-localization
https://ischlag.github.io/2016/04/05/important-ILSVRC-achievements

Learning with Preferences

- Do we need to throw out images where label is ambiguous?
  - We don't have the y_i.

  - We want classifier to prefer Siberian husky over bulldog, Chihuahua, etc.
    - Even though we don't know if these are Siberian huskies or Inuit dogs.

  - Can we design a loss that enforces preferences rather than "true" labels?

https://ischlag.github.io/2016/04/05/important-ILSVRC-achievements

Learning with Pairwise Preferences (Ranking)

- Instead of y_i, we're given list of (c_1,c_2) preferences for each 'i':

We want $w_{c_1}^T x_i > w_{c_2}^T x_i$ for these <u>particular</u> (c_1,c_2) values

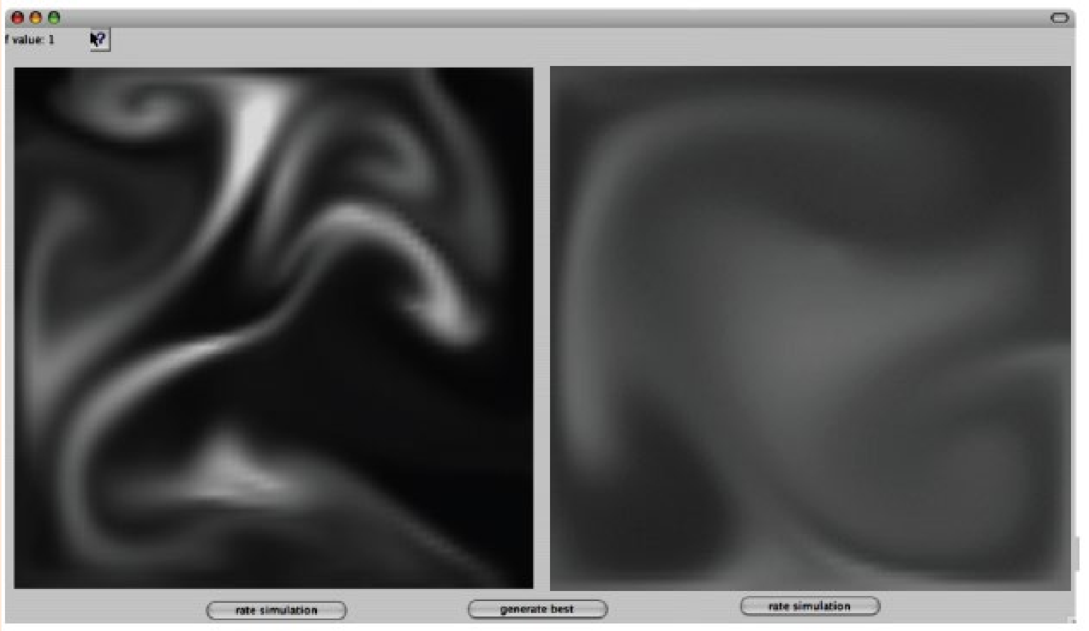
- Multi-class classification is special case of choosing (y_i,c) for all 'c'.
- By following the earlier steps, we can get objectives for this setting:

$$\sum_{i=1}^{n} \underbrace{\sum_{(c_1,c_2)} \max\{0, 1 - w_{c_1}^T x_i + w_{c_2}^T x_i\}}_{\text{"sum" version of multi-class SVM}} + \frac{\lambda}{2}\|W\|_F^2$$

https://lschlag.github.io/2016/04/05/important-ILSVRC-achievements

Learning with Pairwise Preferences (Ranking)

- Pairwise preferences for computer graphics:
  - We have a smoke simulator, with several parameters:

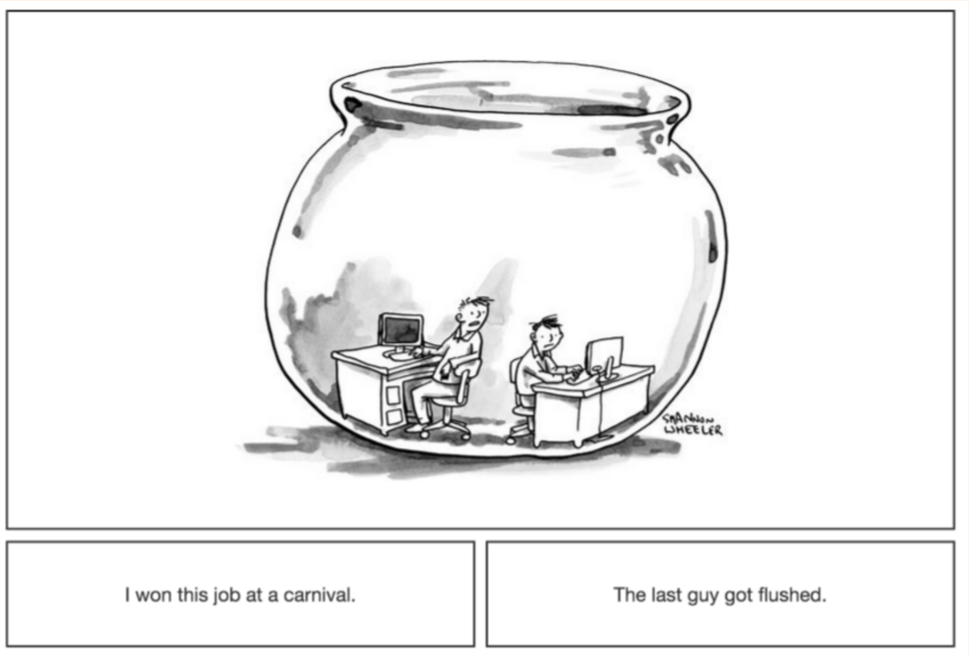


  - Don't know what the optimal parameters are, but we can ask the artist:
    - "Which one looks more like smoke"?

https://circle.ubc.ca/bitstream/handle/2429/30519/ubc_2011_spring_brochu_eric.pdf?sequence=3

Learning with Pairwise Preferences (Ranking)

- Pairwise preferences for humour:
  - New Yorker caption contest:



I won this job at a carnival.

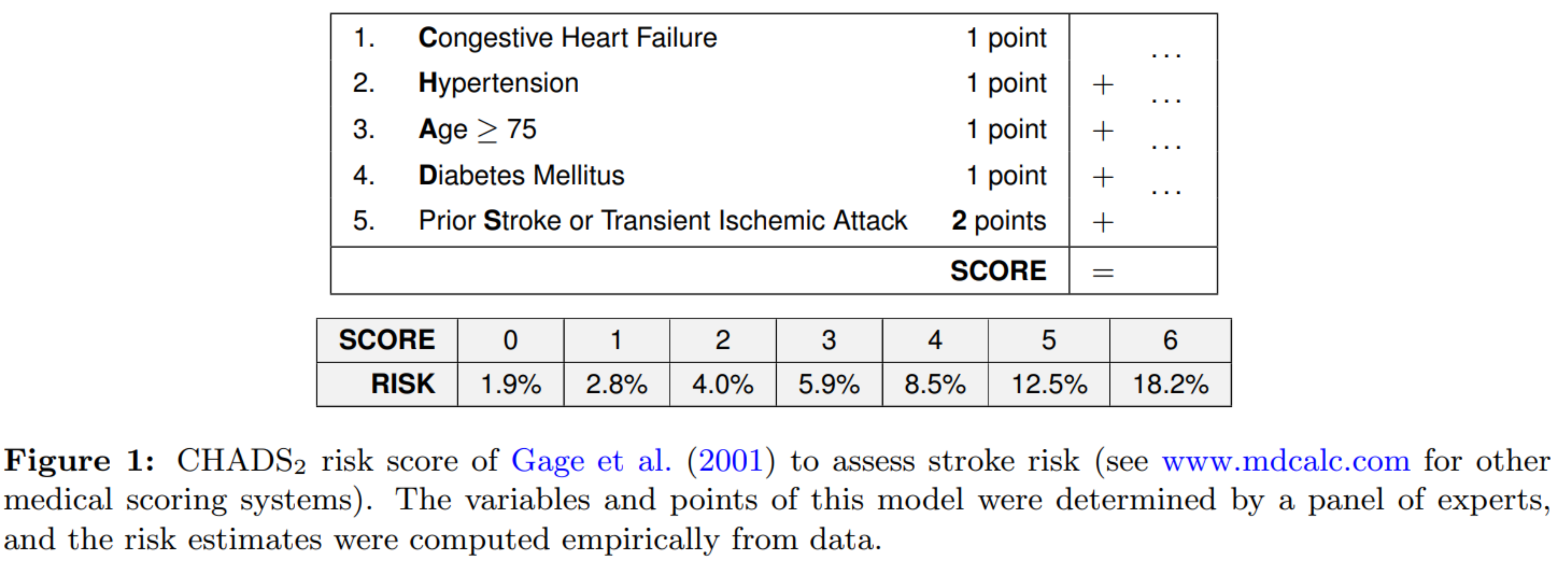
The last guy got flushed.

  - "Which one is funnier"?

https://homes.cs.washington.edu/~jamieson/resources/next.pdf

Risk Scores

- In medicine/law/finance, risk scores are sometimes used to give probabilities:



1.	**C**ongestive Heart Failure	1 point	. . .
2.	**H**ypertension	1 point	+ . . .
3.	**A**ge ≥ 75	1 point	+ . . .
4.	**D**iabetes Mellitus	1 point	+ . . .
5.	Prior **S**troke or Transient Ischemic Attack	**2** points	+
		SCORE	=

SCORE	0	1	2	3	4	5	6
RISK	1.9%	2.8%	4.0%	5.9%	8.5%	12.5%	18.2%

Figure 1: CHADS$_2$ risk score of Gage et al. (2001) to assess stroke risk (see www.mdcalc.com for other medical scoring systems). The variables and points of this model were determined by a panel of experts, and the risk estimates were computed empirically from data.

- Get integer-valued "points" for each "risk factor", and probability is computed from data based on people with same number of points.
- Less accurate than fancy models, but interpretable and can be done by hand.
  - Some work on trying to "learn" the whole thing (like doing feature selection then rounding).

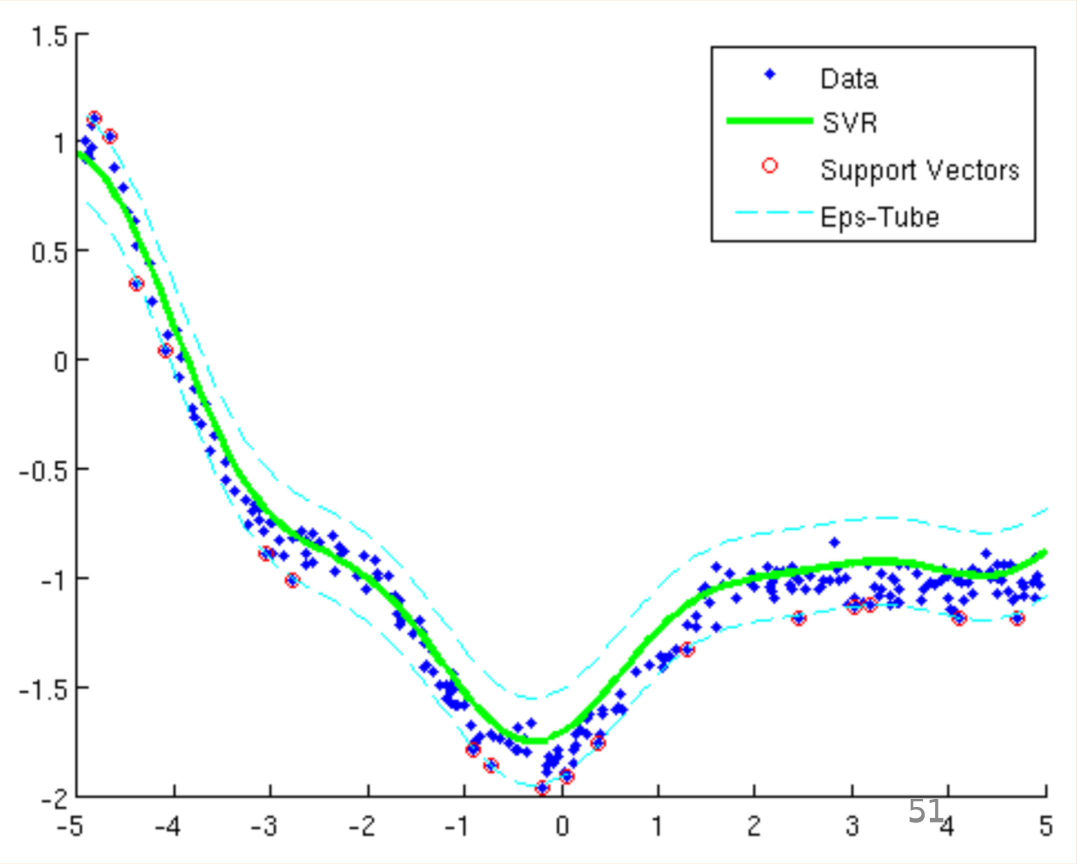
https://arxiv.org/pdf/1610.00168.pdf

Support Vector Regression

- Support vector regression objective (with hyper-parameter ϵ):

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

  - Looks like L2-regularized robust regression with the L1-loss.
  - But have loss of 0 if $\hat{y}_i$ within ϵ of $\tilde{y}_i$.
    - So doesn’t try to fit data exactly.
      - This can help fight overfitting.
  - Support vectors are points with loss>0.
    - Points outside the “epsilon-tube”.
  - Example with Gaussian-RBFs as features:

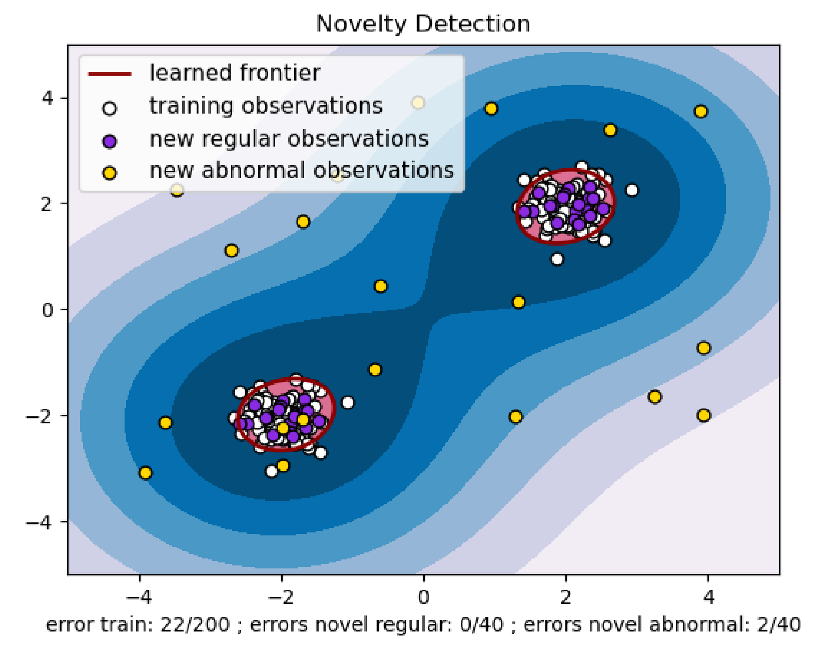


1-Class SVMs

- 1-class SVMs for outlier detection.

$$f(w, w_0) = \sum_{i=1}^{N} [\max\{0, w_0 - w^T x_i\} - w_0] + \frac{\lambda}{2}\|w\|_2^2$$

  - Variables are ‘w’ (vector) and ‘w_0’ (scalar).
  - Only trains on “inliers”.
    - Tries to make $w^T x_i$ bigger than w_0 for inliers.
    - At test time: says “outlier” if $w^T x_i < w_0$.
    - Usually used with RBFs.



https://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html