CPSC 340: Machine Learning and Data Mining

MLE and MAP

Summer 2021

In This Lecture

1. Maximum Likelihood Estimation (35 minutes)
2. Maximum A Posteriori Estimation (20 minutes)

Motivation for Learning about MLE and MAP

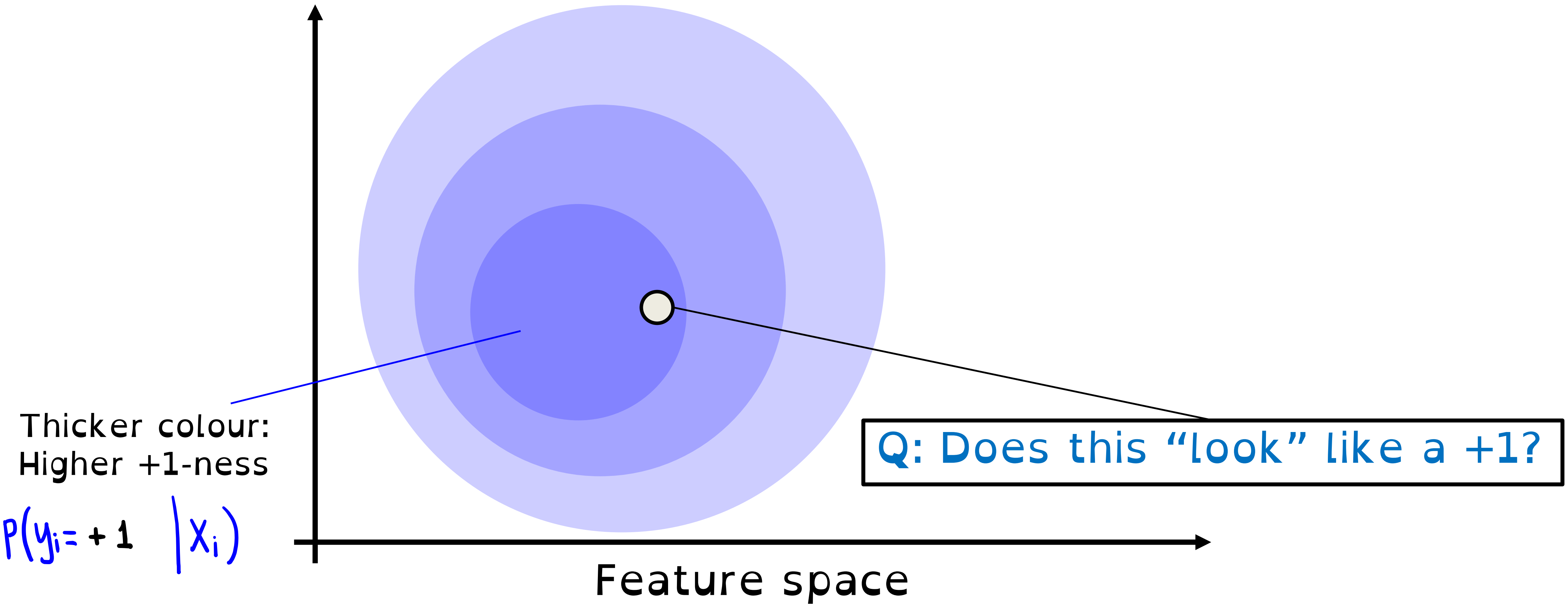
- Next topic: maximum likelihood estimation (MLE) and MAP estimation.
  - Crucial to understanding advanced methods, notation can be difficult at first.

- Why are we learning about these?
  - Justifies the naïve Bayes "counting" estimates for probabilities.
  - Shows the connection between least squares and the normal distribution.
  - Makes connection between "robust regression" and "heavy tailed" probabilities.
  - Shows that regularization and Laplace smoothing are doing the same thing.
  - Justifies using sigmoid function to get probabilities in logistic regression.
  - Gives a way to write complicated ML problems as optimization problems.
    - How do you define a loss for "number of Facebook likes" or "1-5 star rating"?

Coming Up Next

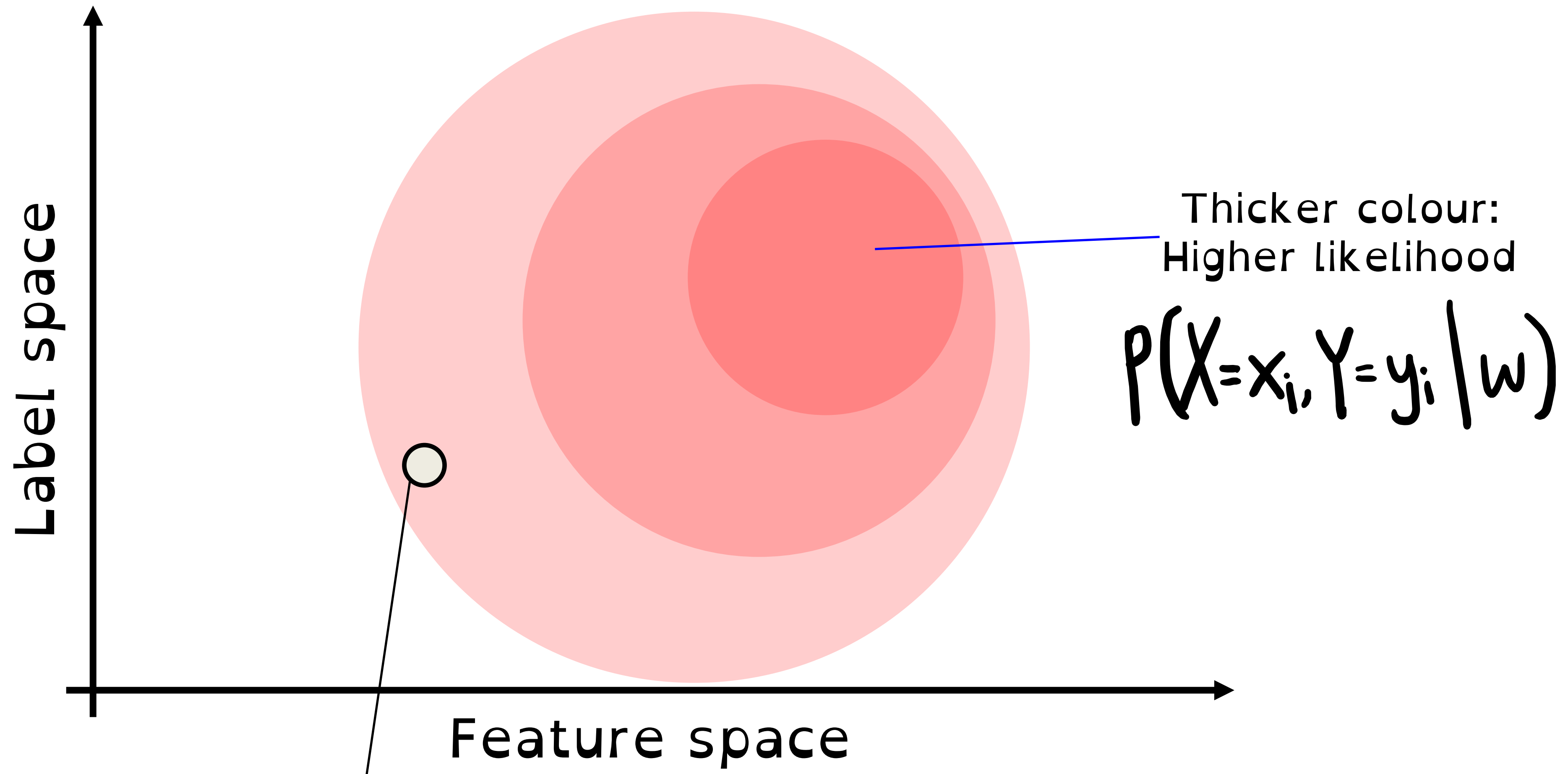
WHAT IS A LIKELIHOOD?

Recall: "+1-ness"



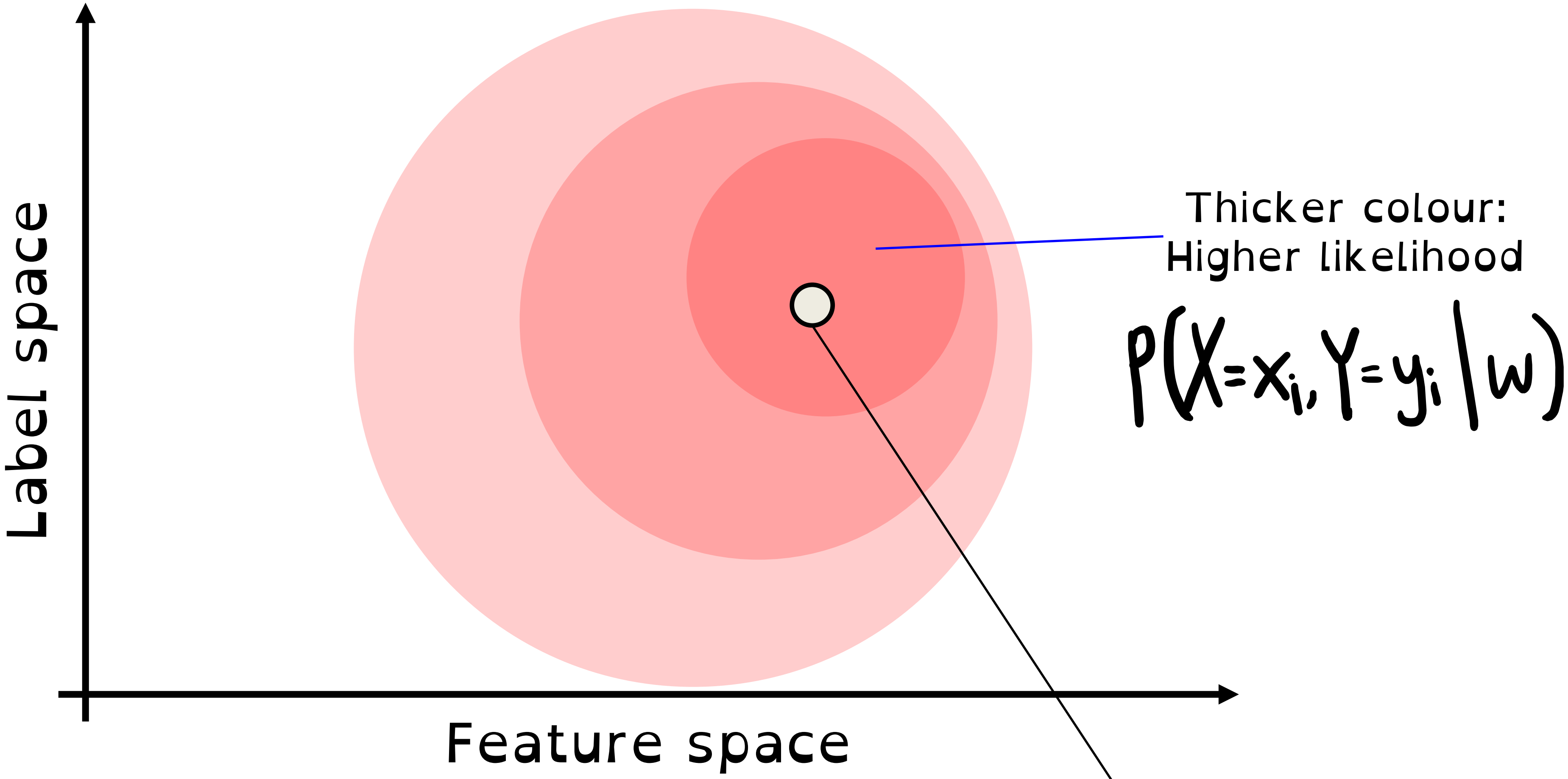
$P(y_i = +1 \mid X_i)$

Now: “Likelihood”



Q: Is this a “likely” example?

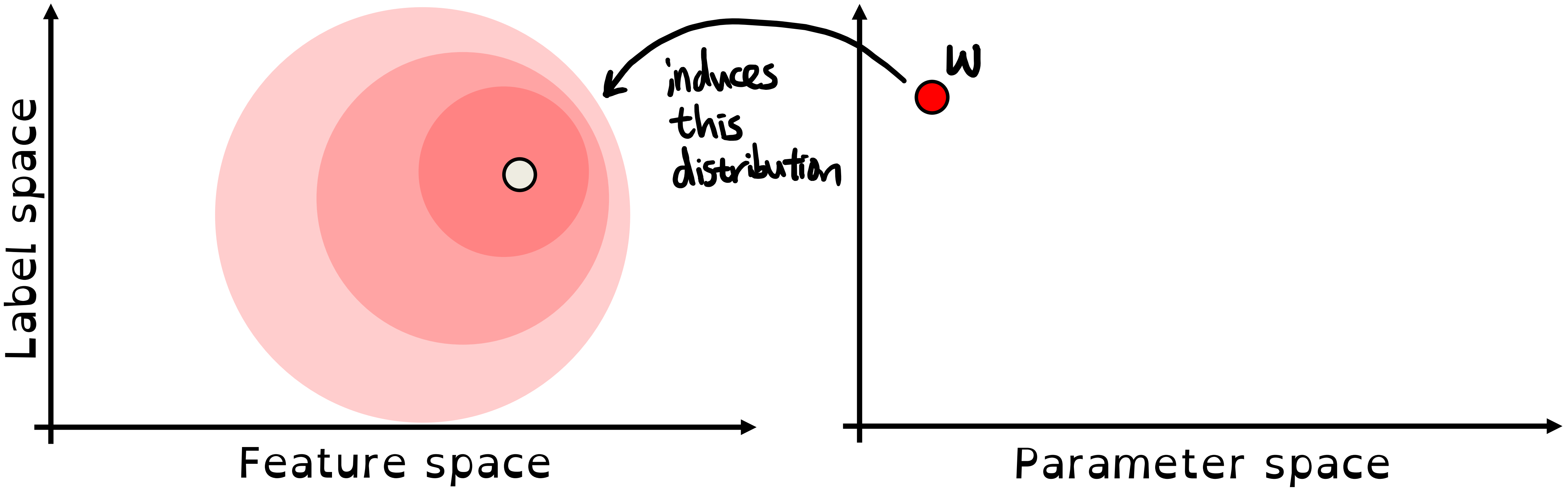
Now: "Likelihood"



Q: Is this a "likely" example?

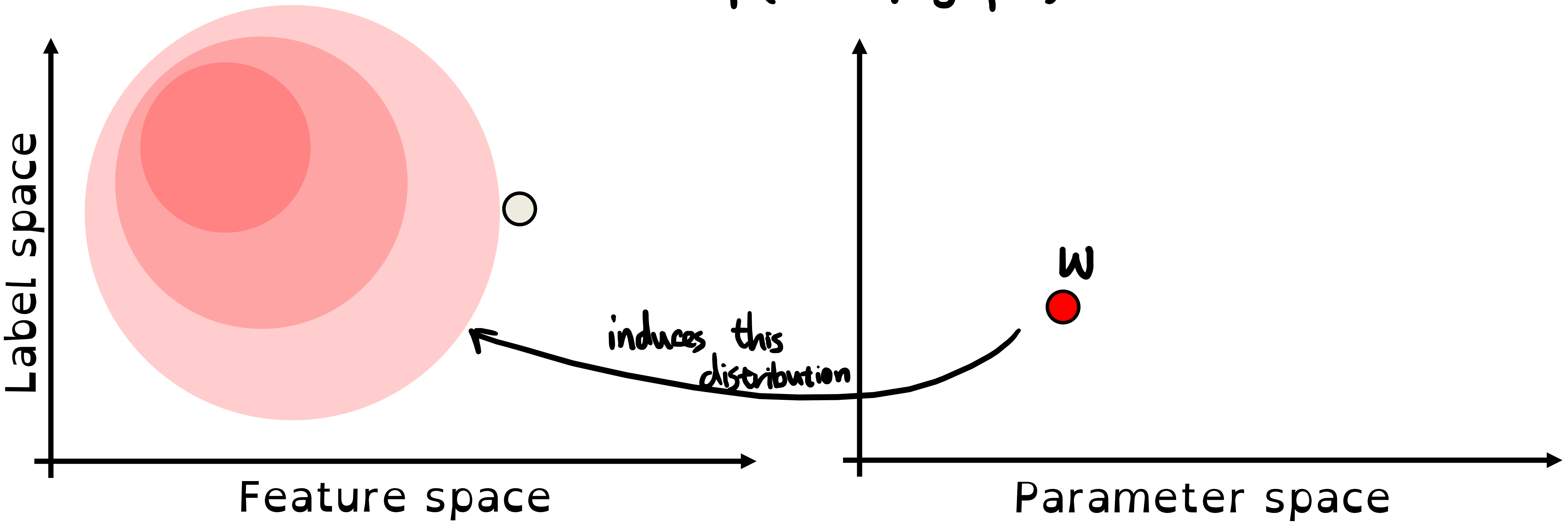
Now: "Likelihood"

$P(X=x_i, Y=y_i \mid w)$



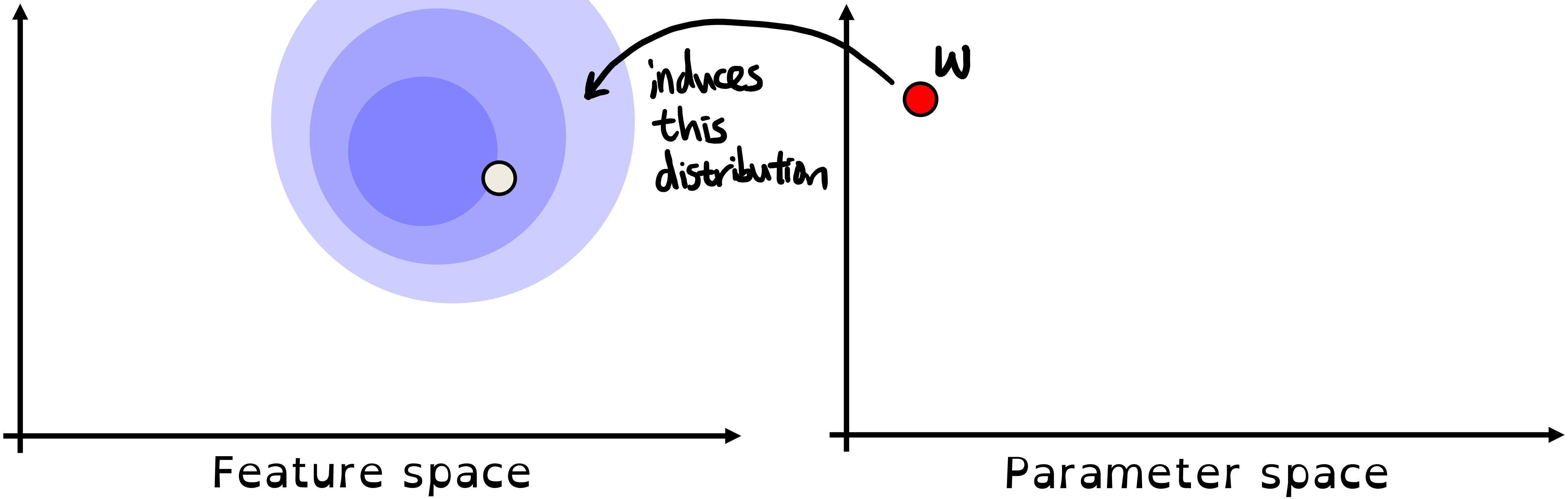
Now: “Likelihood”

$$P(X=x_i, Y=y_i \mid w)$$



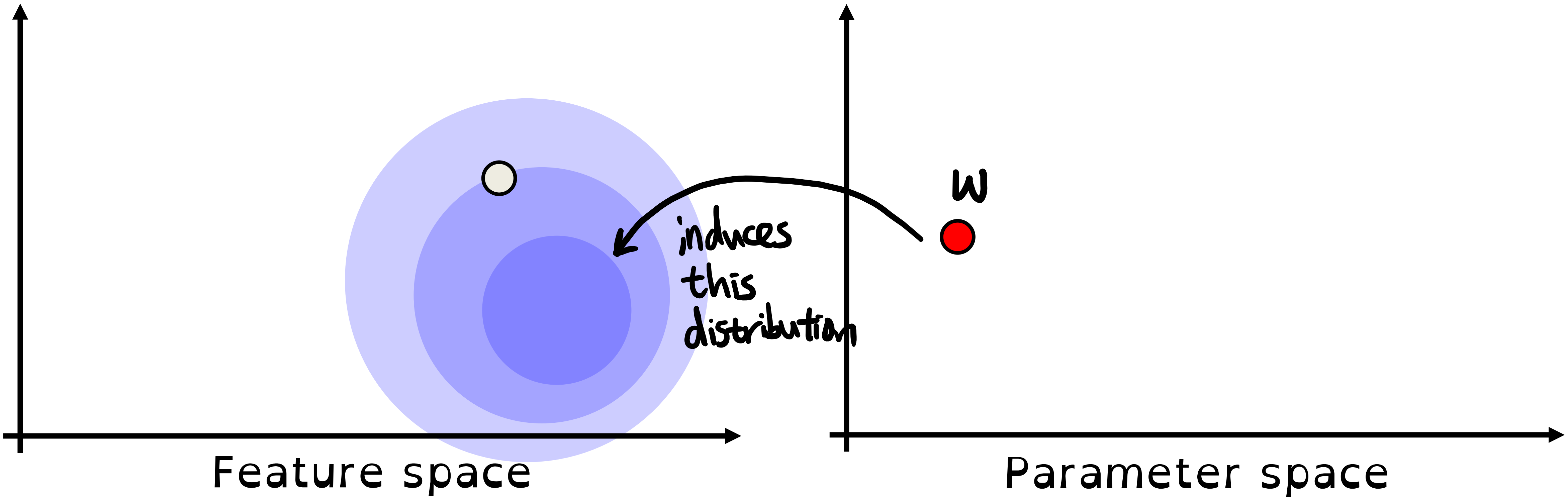
+1-ness Also Depends on Parameters

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



+1-ness Also Depends on Parameters

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



Model = Assumption

$$X = {}_{n}\begin{bmatrix} & & \end{bmatrix}_{d} \rightarrow y = {}_{n}\begin{bmatrix} \; \end{bmatrix}_{1}$$

$$w^* = {}_{d}\begin{bmatrix} \; \end{bmatrix}_{1}$$

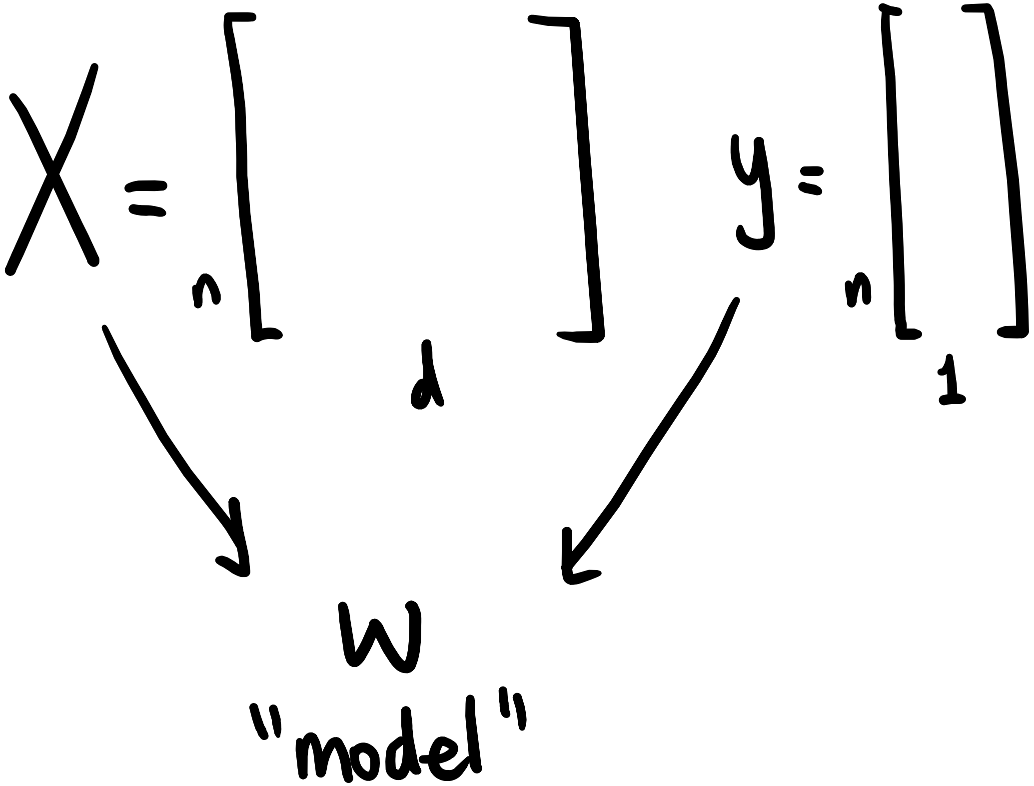
"true parameters"

e.g. $y = Xw^* + \varepsilon$

$$\varepsilon \sim \mathcal{N}(0, \sigma^2)$$

- Linear models assume that the data was generated according to some linear combination

Learning = Finding Parameters



- Given data, "learning a model" means approximating the parameters of the data generating process
  - e.g. feature coefficients for a linear model

"Likelihood"

$$P(D \mid w)$$

Probability of seeing dataset D given parameters w

$$P(X, y \mid w)$$

Probability of seeing dataset X,y given parameters w

$$P(y \mid X, w)$$

Probability of seeing labels y given parameters w and features X

$$= P((y \mid X) \mid w)$$

Probability of seeing labeled dataset X,y given parameters w

- A parameter value induces a distribution called "likelihood"
  - "Probability of seeing the given data"
  - Corresponds to the assumption about how data is generated

Signature of Likelihood Function

$$P(\cdot \mid w) : \mathbb{R}^{n \times d} \times \mathbb{R}^{n} \to [0, 1]$$

fixed w, varying data
(unsupervised)

$$P(y \mid X, \cdot) : \mathbb{R}^{d} \to [0, 1]$$

fixed data, varying w

- Be careful about what's being varied:
  - Given same w, we can vary examples
    - We are usually doing this during prediction
  - Given same examples, we can vary w
    - Varying w changes the induced distribution
    - We are usually doing this during training

Coming Up Next

MAXIMUM LIKELIHOOD ESTIMATION

"argmin" and "argmax"

- We've repeatedly used the min and max functions:

$$\min_w \{w^2\} = 0 \qquad \max_w \{\cos(w)\} = 1$$

  - Minimum (or maximum) value achieved by a function.

- A related set of functions are the argmin and argmax:
  - The set of parameter values achieving the minimum (or maximum).

$$\min_w \{(w-1)^2\} = 0$$

$$\underset{w}{\operatorname{argmin}} \{(w-1)^2\} = 1$$

'1' is the 'w' value that gives the min.

$$\underset{w}{\operatorname{argmin}} \left\{\frac{1}{2}\|Xw-y\|^2 + \frac{\lambda}{2}\|w\|^2\right\} = (X^TX+\lambda I)^{-1}(X^Ty)$$

$$\underset{w}{\operatorname{argmax}} \{\cos(w)\} = 0, 2\pi, 4\pi, \ldots$$

"argmin" and "argmax"

- The last slide is a little sloppy for the following reason:
  - There may be multiple values achieving the min and/or max.
  - So the argmin and argmax return sets.

$$\underset{w}{\operatorname{argmin}}\{(w-1)^2\} \equiv \{1\}$$

"set containing the element '1'"

"sets are equivalent"

$$\underset{w}{\operatorname{argmax}}\{\cos(w)\} \equiv \{\dots, -4\pi, -2\pi, 0, 2\pi, 4\pi, \dots\}$$

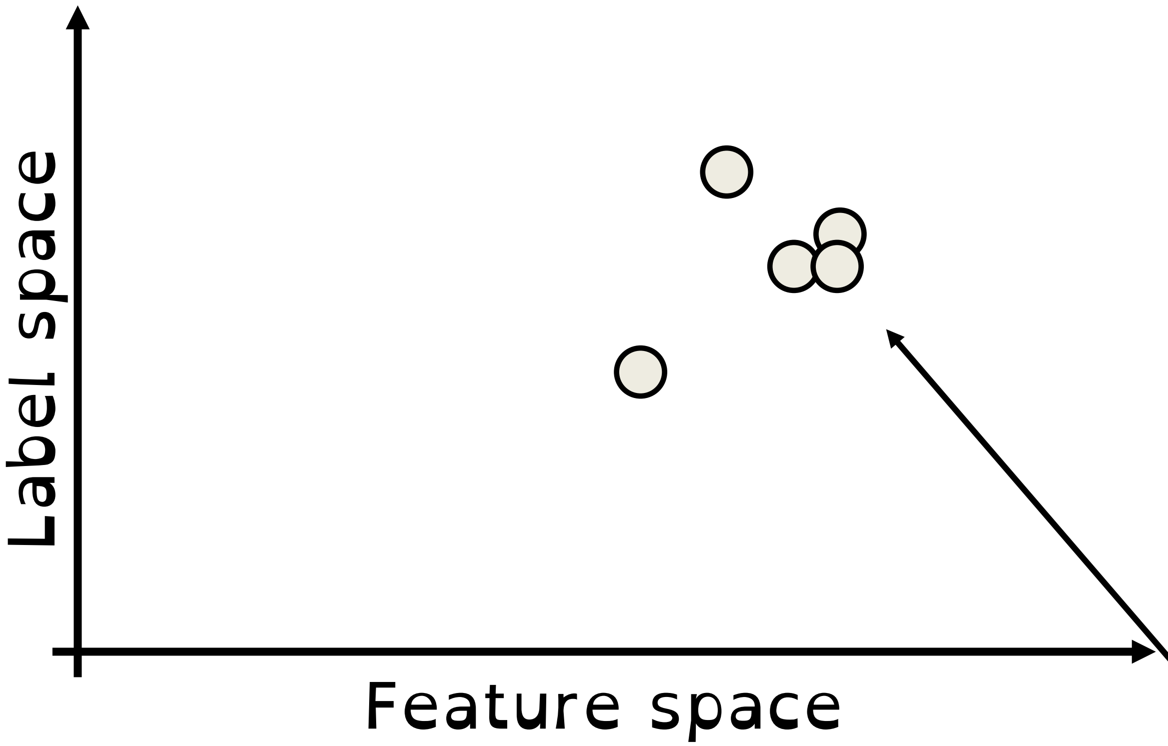
$$\underset{w}{\operatorname{argmax}}\left\{\frac{1}{2}\|Xw-y\|^2\right\} \equiv \{w \mid X^TXw = X^Ty\}$$

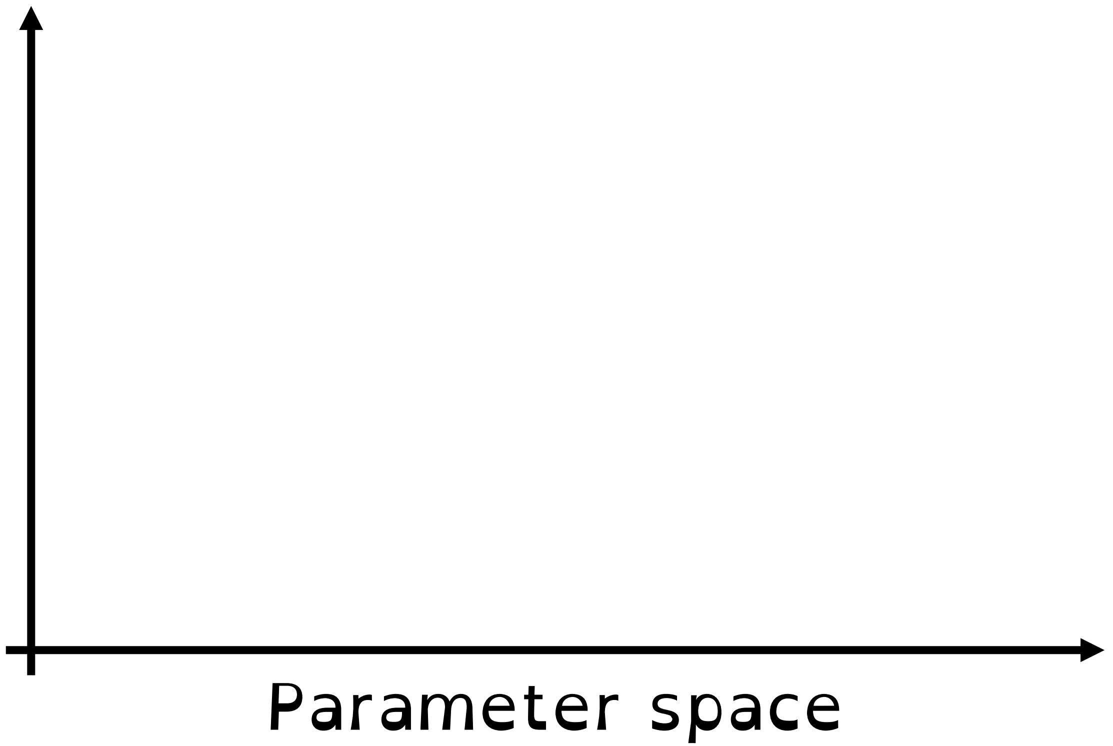
  - And we don't say a variable "is" the argmax, but that it "is in" the argmax.

$$2\pi \in \underset{w}{\operatorname{argmax}}\{\cos(w)\} \qquad (X^TX+\lambda I)^{-1}(X^Ty) \in \underset{w}{\operatorname{argmin}}\left\{\frac{1}{2}\|Xw-y\|^2 + \frac{\lambda}{2}\|w\|^2\right\}$$

Maximum Likelihood Estimation

$p(X=x_i, Y=y_i \mid w)$

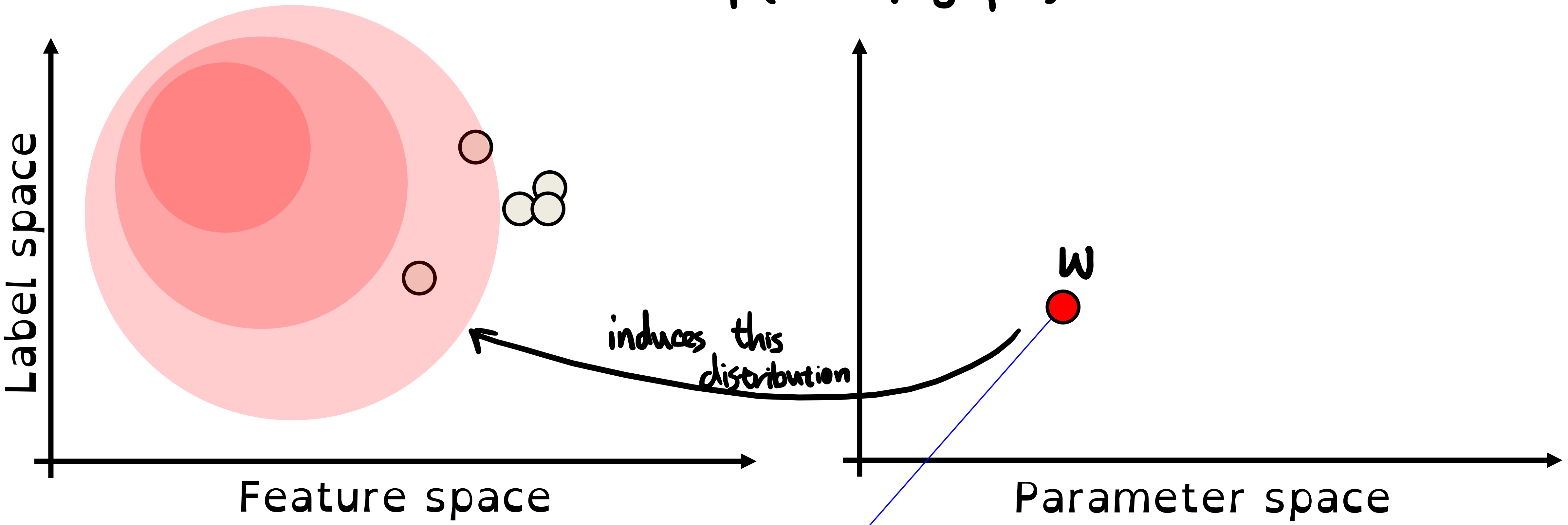




Q: What is the parameter value that makes these examples **most likely**?

Maximum Likelihood Estimation

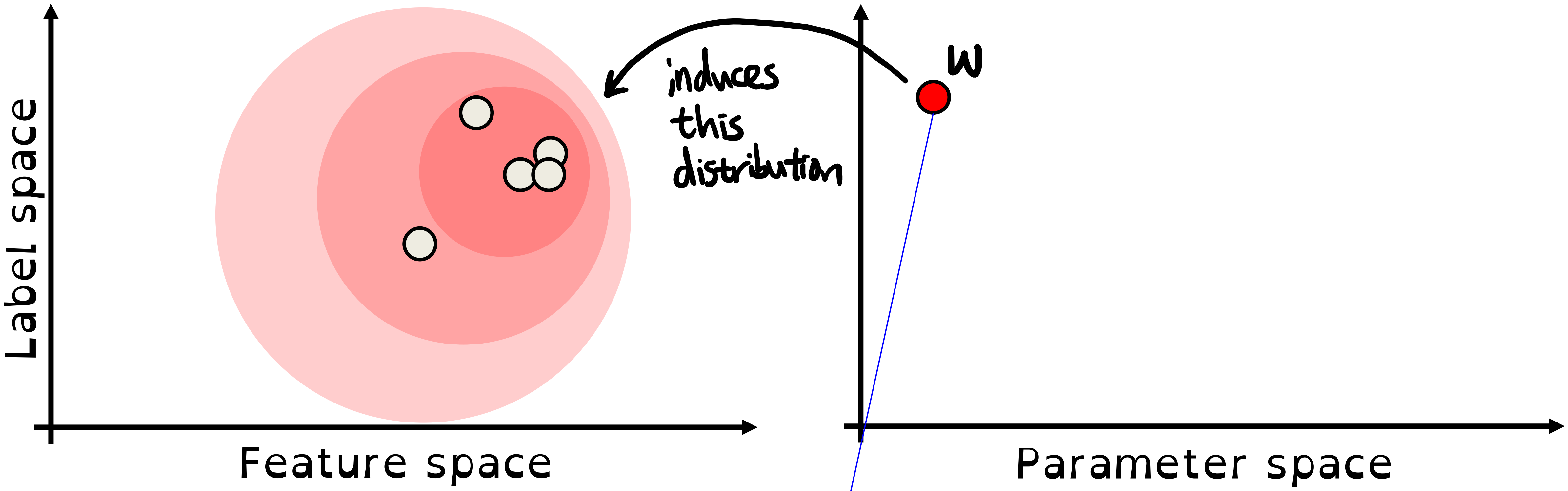
$P(X=x_i, Y=y_i \mid w)$



Q: Is this a good parameter value?

Maximum Likelihood Estimation

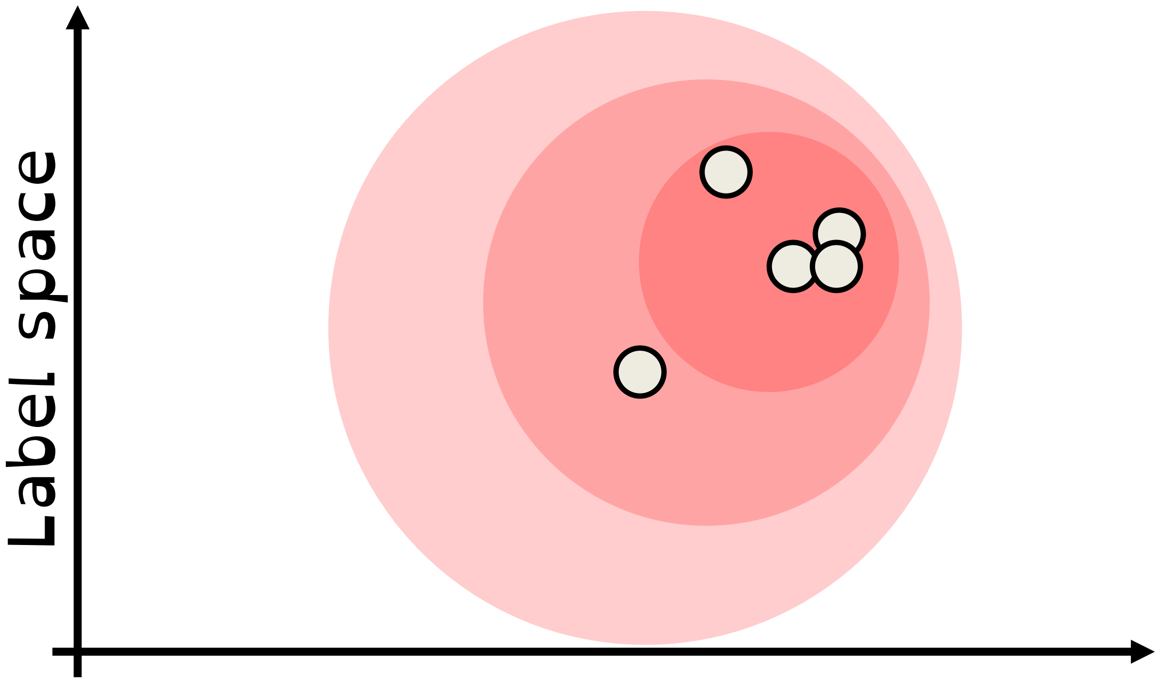
$P(X = x_i, Y = y_i \mid w)$

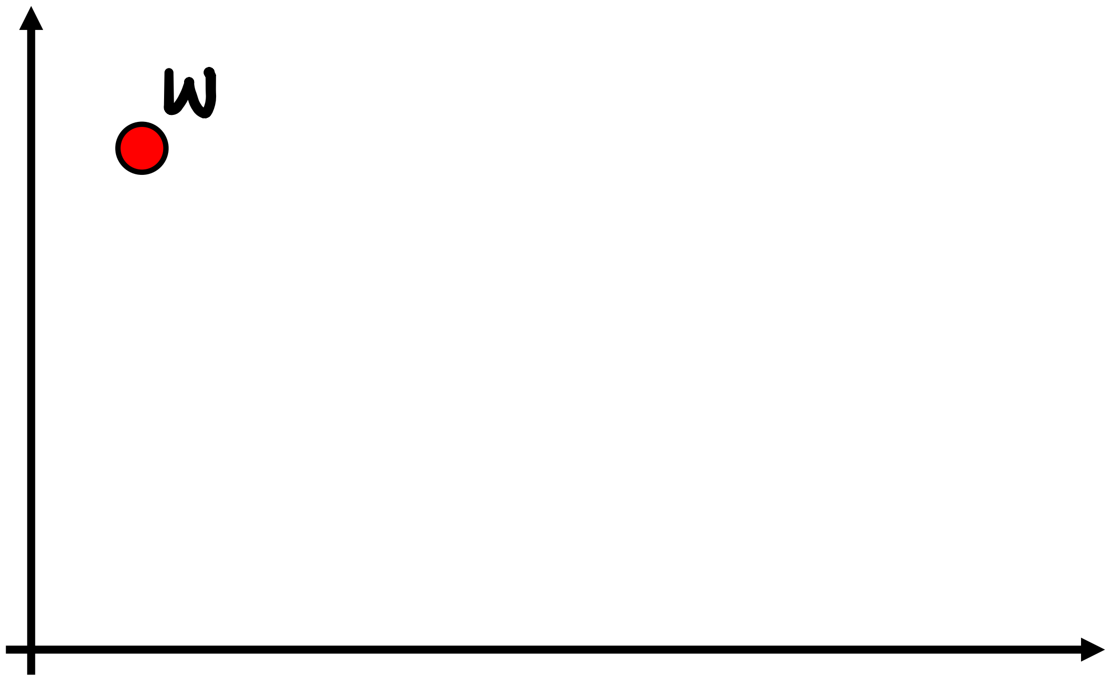


Q: Is this a good parameter value?

Maximum Likelihood Estimation

- Maximum likelihood estimation (MLE):
  - Choose parameters that maximize the likelihood:





$$\hat{w} \in \underset{w}{\operatorname{argmax}} \{ P(D|w) \}$$

MLE for Binary Variables (General Case)

- Consider a binary feature:

$$X = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

- Using 'w' as "probability of 1", the maximum likelihood estimate is:

$$\hat{w} = \frac{\#\text{ of ones}}{\#\text{ of examples}}$$

- This is the "estimate" for the probabilities we used in naïve Bayes.
  - The conditional probabilities we used in naïve Bayes are also MLEs.
    - The derivation is tedious, but if you're interested I put it here.

Coming Up Next

MAXIMUM LIKELIHOOD ESTIMATION AND NEGATIVE LOG LIKELIHOOD

Maximum Likelihood Estimation (MLE)

- Maximum likelihood estimation (MLE) for fitting probabilistic models.
  - We have a dataset D.
  - We want to pick parameters 'w'.
  - We define the likelihood as a probability mass/density function p(D | w).
  - We choose the model $\hat{w}$ that maximizes the likelihood:

$$\hat{w} \in \underset{w}{\operatorname{argmax}} \{ p(D \mid w) \}$$

- Appealing "consistency" properties as n goes to infinity (take STAT 4XX).
  - "This is a reasonable thing to do for large data sets".

Least Squares is Gaussian MLE

- It turns out that most objectives have an MLE interpretation:
  - For example, consider minimizing the squared error:

$$f(w) = \frac{1}{2}\|Xw - y\|^2$$

  - This gives MLE of a linear model with IID noise from a normal distribution:

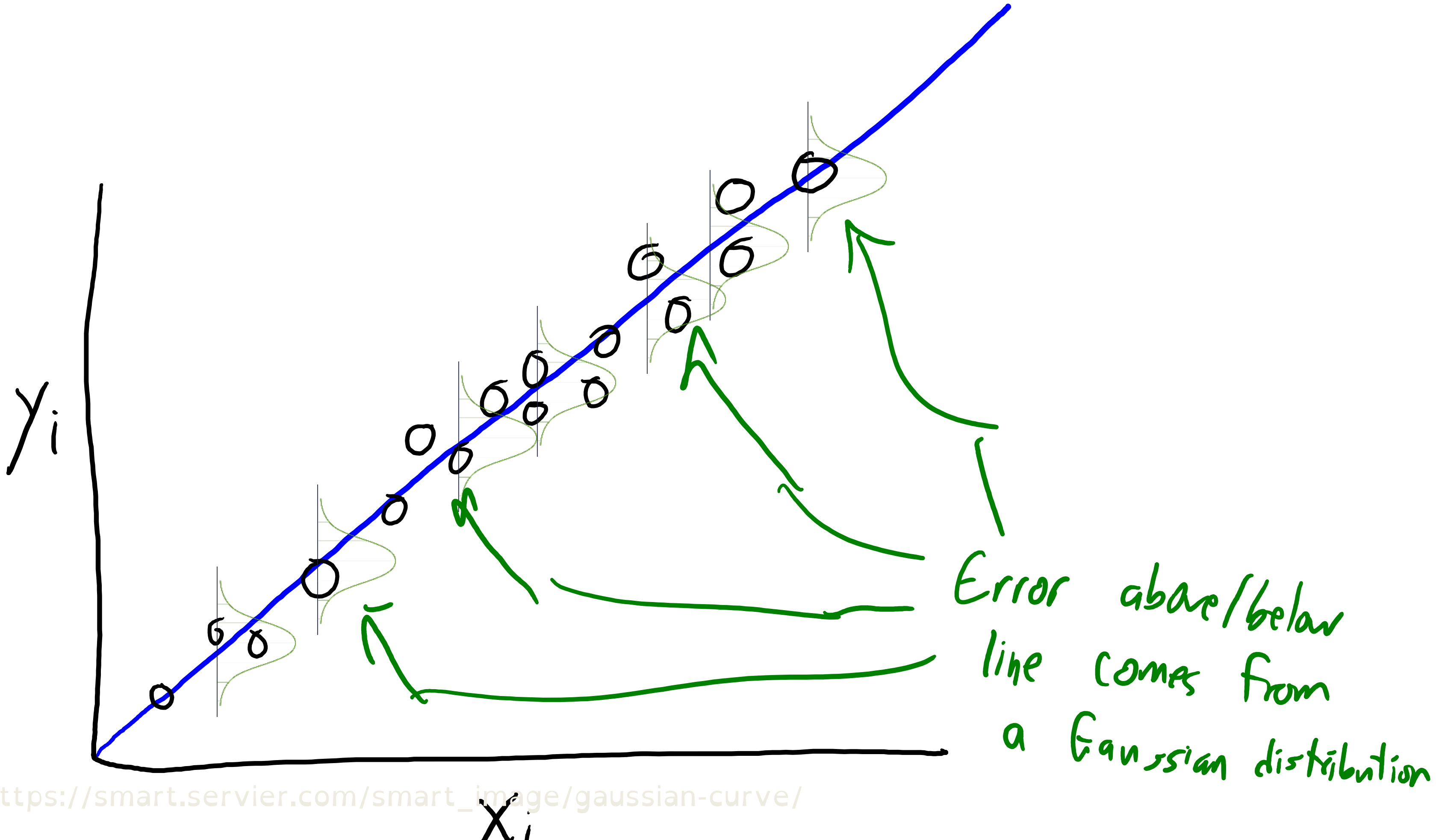
$$y_i = w^T x_i + \varepsilon_i$$

where each ε_i is sampled independently from standard normal

    - "Gaussian" is another name for the "normal" distribution.
  - Remember that least squares solution is called the "normal equations".

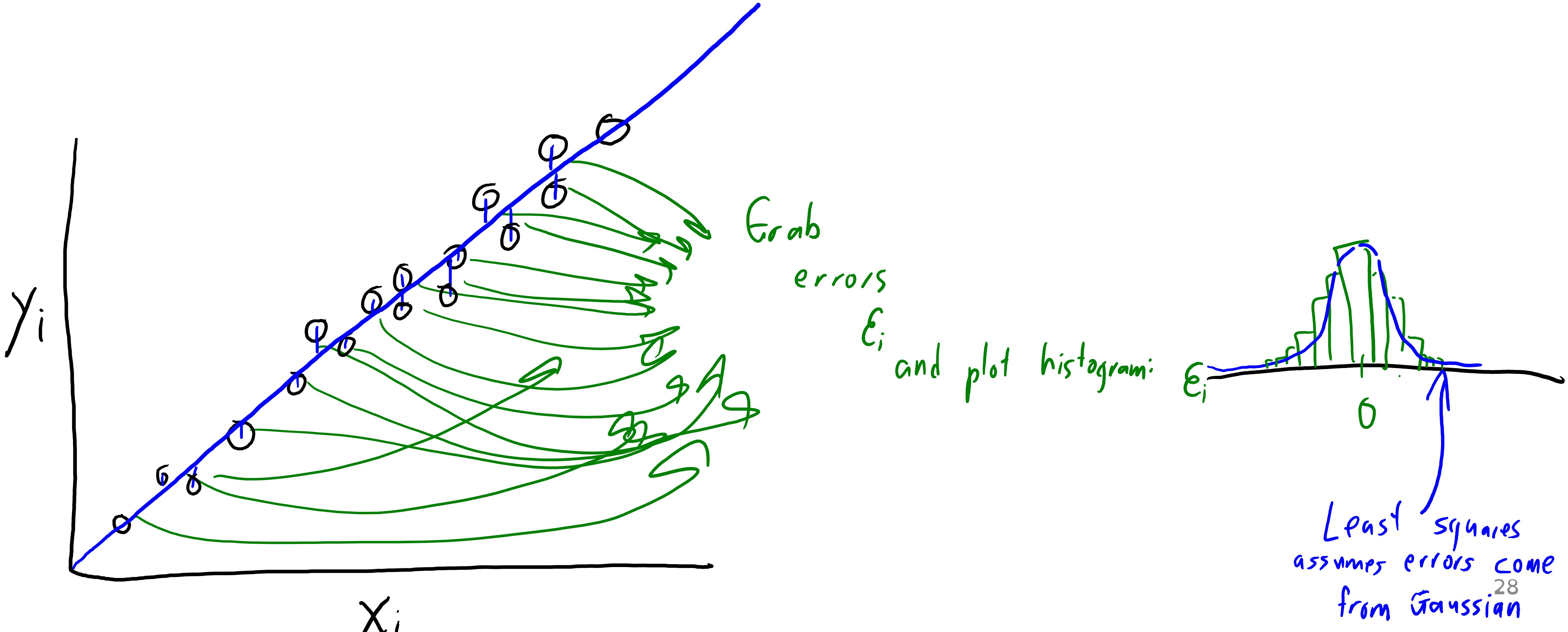
Least Squares is Gaussian MLE

- It turns out that most objectives have an MLE interpretation:
  - For example, consider minimizing the squared error:



Least Squares is Gaussian MLE

- It turns out that most objectives have an MLE interpretation:
  - For example, consider minimizing the squared error:



Minimizing the Negative Log-Likelihood (NLL)

- To compute MLE, usually we equivalently minimize the negative "log-likelihood" (NLL):
  - "Log-likelihood" is short for "logarithm of the likelihood".

$$\hat{w} \in \underset{w}{\operatorname{argmax}}\{p(D \mid w)\} \underset{\text{"equivalent"}}{\equiv} \underset{w}{\operatorname{argmin}}\{-\log(p(D \mid w))\}$$

- Why are these equivalent?
  - Logarithm is strictly monotonic: if $\alpha > \beta$, then $\log(\alpha) > \log(\beta)$.
    - So location of maximum doesn't change if we take logarithm.
  - Changing sign flips max to min.

- See Max and Argmax notes if this seems strange.

Minimizing the Negative Log-Likelihood (NLL)

- We use log-likelihood because it turns multiplication into addition:

$$\log(\alpha\beta) = \log(\alpha) + \log(\beta)$$

- More generally:

$$\log\left(\prod_{i=1}^{n} a_i\right) = \sum_{i=1}^{n} \log(a_i)$$

- If data is 'n' IID samples then $p(D \mid w) = \prod_{i=1}^{n} \underbrace{p(D_i \mid w)}_{\text{likelihood of example 'i'}}$

and our MLE is $\hat{w} \in \underset{w}{\operatorname{argmax}} \left\{ \prod_{i=1}^{n} p(D_i \mid w) \right\} \equiv \underset{w}{\operatorname{argmin}} \left\{ -\sum_{i=1}^{n} \log(p(D_i \mid w)) \right\}$

Least Squares is Gaussian MLE (Gory Details)

- Let's assume that $y_i = w^T x_i + \varepsilon_i$, with ε_i following standard normal:

$$p(\varepsilon_i) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{\varepsilon_i^2}{2}\right)$$

also known as "Gaussian" distribution

- This leads to a Gaussian likelihood for example 'i' of the form:

$$p(y_i \mid x_i, w) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)$$

- Finding MLE (minimizing NLL) is least squares:

[1] $f(w) = -\sum_{i=1}^{n} \log(p(y_i \mid w, x_i))$

[2] $= -\sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)\right)$

[3] $= -\sum_{i=1}^{n} \left[\log\left(\frac{1}{\sqrt{2\pi}}\right) + \log\left(\exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)\right)\right]$

constant in 'w'

operations cancel

[4] $= -\sum_{i=1}^{n} \left[(\text{constant}) - \frac{1}{2}(w^T x_i - y_i)^2\right]$

[5] $= (\text{constant}) + \frac{1}{2}\sum_{i=1}^{n} (w^T x_i - y_i)^2$

[6] $= (\text{constant}) + \frac{1}{2}\|Xw - y\|^2$

Coming Up Next

MORE DETAILS ON MAXIMUM LIKELIHOOD ESTIMATES

Digression: "Generative" vs. "Discriminative"

- Discriminative model:
  - Optimize parameters to maximize "+1-ness" for +1 examples, etc.

$$\hat{w} \in \underset{w}{\operatorname{argmax}} \{ p(\underbrace{y \mid X}_{D}, w) \}$$

$$p((y|x) \mid w) = p(y \mid X, w)$$

- Generative model:
  - Optimize parameters to maximize "data likelihood"

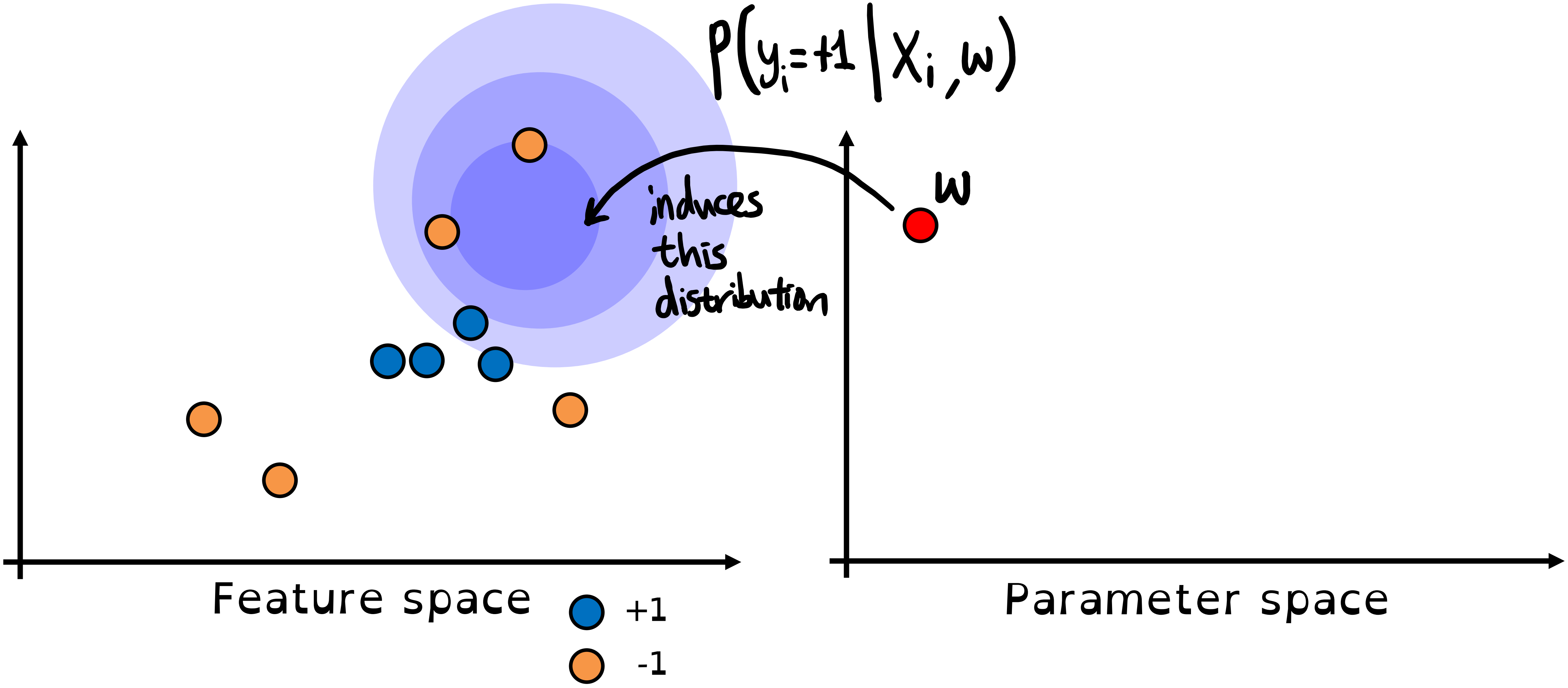
$$\hat{w} \in \underset{w}{\operatorname{argmax}} \{ p(\underbrace{y, X}_{D} \mid w) \}$$

- Prediction time:
  - both discriminative and generative models aim to predict "+1-ness"

Digression: “Generative” vs. “Discriminative”

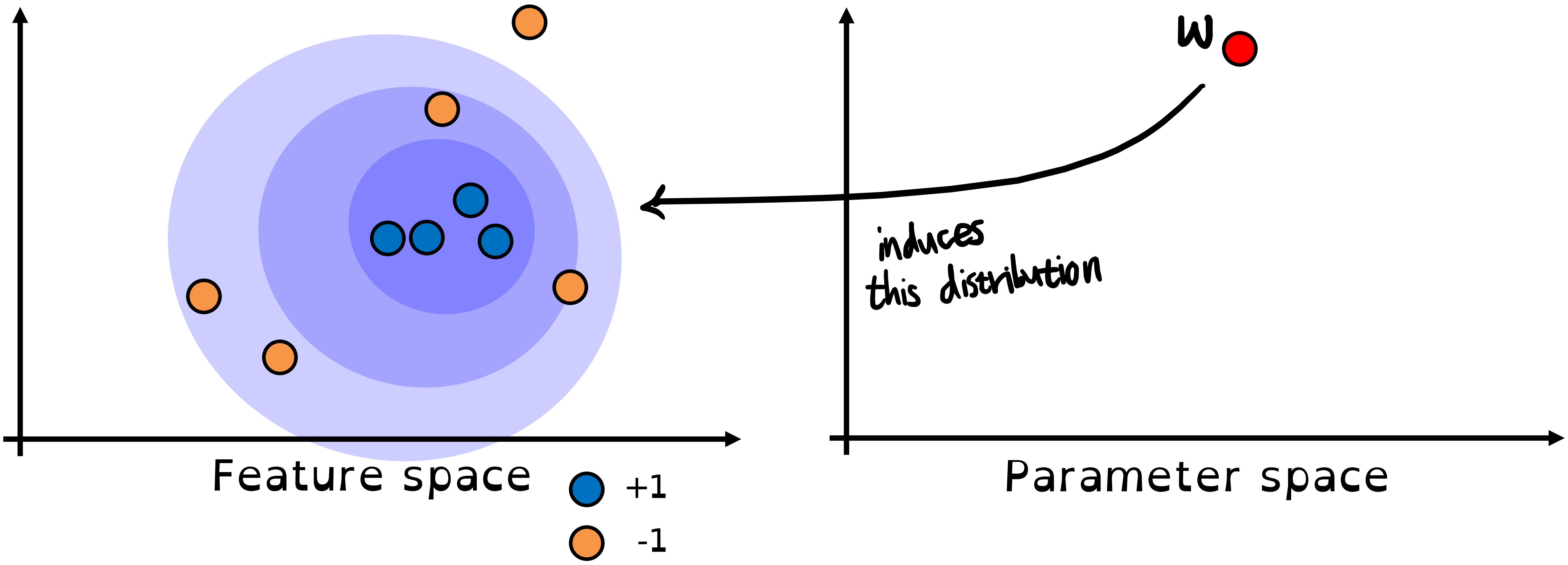
- For least squares, maximize conditional p(y | X, w), not the likelihood p(y, X | w).
  - We did MLE “conditioned” on the features ‘X’ being fixed (no “likelihood of X”).
  - This is called a “discriminative” supervised learning model.
  - A “generative” model (like naïve Bayes) would optimize p(y, X | w).
- Discriminative probabilistic models:
  - Least squares, robust regression, logistic regression.
  - Can use complicated features because you don’t model ‘X’.
- Example of generative probabilistic models:
  - Naïve Bayes, linear discriminant analysis (makes Gaussian assumption).
  - Often need strong assumption because they model ‘X’.
- “Folk” belief: generative models are often better with small ‘n’.

Discriminative MLE



Discriminative MLE

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



Loss Functions and Maximum Likelihood Estimation

- So least squares is MLE under Gaussian likelihood.

If $p(y_i | x_i, w) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)$

then MLE of 'w' is minimum of $f(w) = \frac{1}{2}\|Xw - y\|^2$

- With a Laplace likelihood you would get absolute error.

If $p(y_i | x_i, w) = \frac{1}{2} \exp(-|w^T x_i - y_i|)$

then MLE is minimum of $f(w) = \|Xw - y\|_1$

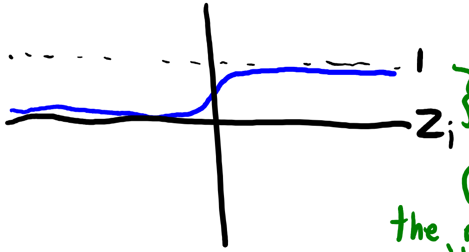
- Other likelihoods lead to different errors ("sigmoid" -> logistic loss).

Sigmoid: transforming $w^T x_i$ to a Probability

- Recall we got probabilities from binary linear models with sigmoid:

$$\text{sigmoid}: \mathbb{R}^d \rightarrow (0, 1)$$

$$h(z_i) = \frac{1}{1 + \exp(-z_i)}$$



Gives values in the range (0,1) that we'll interpret as probabilities

1. The linear model $w^T x_i$ gives us a number z_i in $(-\infty, \infty)$.
2. We'll map $z_i = w^T x_i$ to a probability with the sigmoid function.

- We can show that MLE with this model gives logistic loss.

Sigmoid: transforming w^Tx_i to a Probability

- We'll define $p(y_i = +1 \mid z_i) = h(z_i)$, where 'h' is the sigmoid function.

[1] So $p(y_i = -1 \mid z_i) = 1 - p(y_i = +1 \mid z_i)$

[2] $= 1 - h(z_i)$

[3] $= h(-z_i)$ can show from definition of 'h'

- With y_i in {-1,+1}, we can write both cases as $p(y_i \mid z_i) = h(y_iz_i)$.
- So we convert $z_i=w^Tx_i$ into "probability of y_i" using:

[4] $$p(y_i \mid w, x_i) = h(y_i \underbrace{w^Tx_i}_{z_i})$$

[5] $$= \frac{1}{1+\exp(-y_iw^Tx_i)}$$

- MLE with this likelihood is equivalent to minimizing logistic loss.

MLE Interpretation of Logistic Regression

- For IID regression problems the conditional NLL can be written:

[1] $$\underbrace{-\log(p(y \mid X, w))}_{\text{NLL}} = -\log\left(\underbrace{\prod_{i=1}^{n} p(y_i \mid x_i, w)}_{\text{IID assumption}}\right) = -\underbrace{\sum_{i=1}^{n}}_{\substack{\text{log turns}\\ \text{product into sum}}} \log(p(y_i \mid x_i, w))$$

- Logistic regression assumes sigmoid($w^T x_i$) conditional likelihood:

[2] $$p(y_i \mid x_i, w) = h(y_i w^T x_i) \quad \text{where} \quad h(z_i) = \frac{1}{1+\exp(-z_i)}$$

- Plugging in the sigmoid likelihood, the NLL is the logistic loss:

[3] $$NLL(w) = -\sum_{i=1}^{n} \log\left(\frac{1}{1+\exp(-y_i w^T x_i)}\right) = \sum_{i=1}^{n} \log(1+\exp(-y_i w^T x_i))$$

(since $\log(1) = 0$)

MLE Interpretation of Logistic Regression

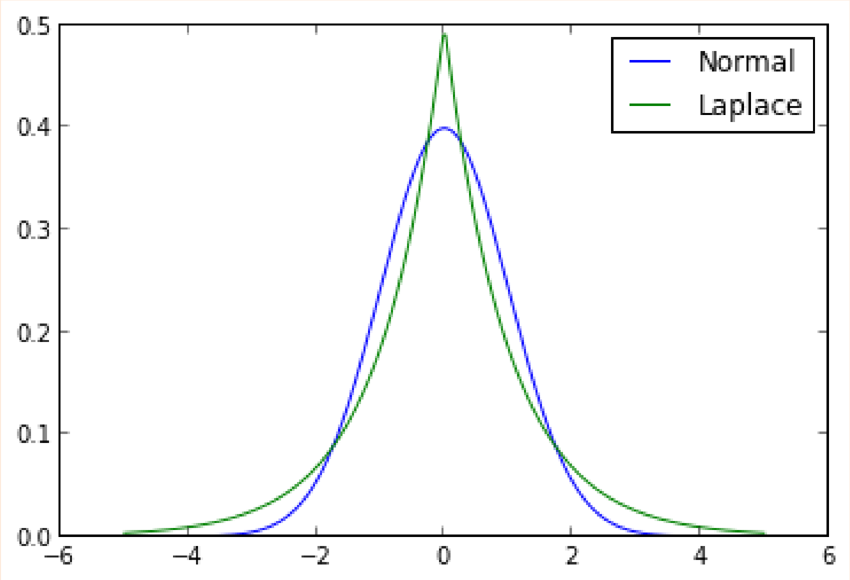
- Instead of "smooth convex approximation of 0-1 loss", we now have that logistic regression is doing MLE in a probabilistic model.
  - "Maximize +1-ness of +1 examples and -1-ness of -1 examples"
  - The training and prediction would be the same as before.
    - We still minimize the logistic loss in terms of 'w'.
  - But MLE justifies using sigmoid with learned w to get +1-ness:

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

  - Softmax function and softmax loss are also connected via NLL
    - See Piazza for derivations

"Heavy" Tails vs. "Light" Tails

- We know that L1-norm is more robust than L2-norm.
  - What does this mean in terms of probabilities?



Here "tail" means "mass of the distribution away from the mean."

  - Gaussian has "light tails": assumes everything is close to mean.
  - Laplace has "heavy tails": assumes some data is far from mean.
  - Student 't' is even more heavy-tailed/robust, but NLL is non-convex.

http://austinrochford.com/posts/2013-09-02-prior-distributions-for-bayesian-regression-using-pymc.html

Coming Up Next

MAXIMUM A POSTERIORI ESTIMATION

Maximum Likelihood Estimation and Overfitting

- In our abstract setting with data D the MLE is:

$$\hat{w} \in \underset{w}{\operatorname{argmax}} \{p(D|w)\}$$

$$P(D|\cdot): \mathbb{R}^d \to [0,1]$$

- But conceptually MLE is a bit weird:
  - "Find the 'w' that makes 'D' have the highest probability given 'w'."
- And MLE often leads to overfitting:
  - Data could be very likely for some very unlikely 'w'.
  - For example, a complex model that overfits by memorizing the data.
- What we really want: $P(\cdot|D): \mathbb{R}^d \to [0,1]$
  - "Find the 'w' that has the highest probability given the data D."

Maximum a Posteriori (MAP) Estimation

- Maximum a posteriori (MAP) estimate maximizes the reverse probability:

$$\hat{w} \in \underset{w}{\operatorname{argmax}} \{ p(w | D) \}$$

  - This is what we want: the probability of 'w' given our data.

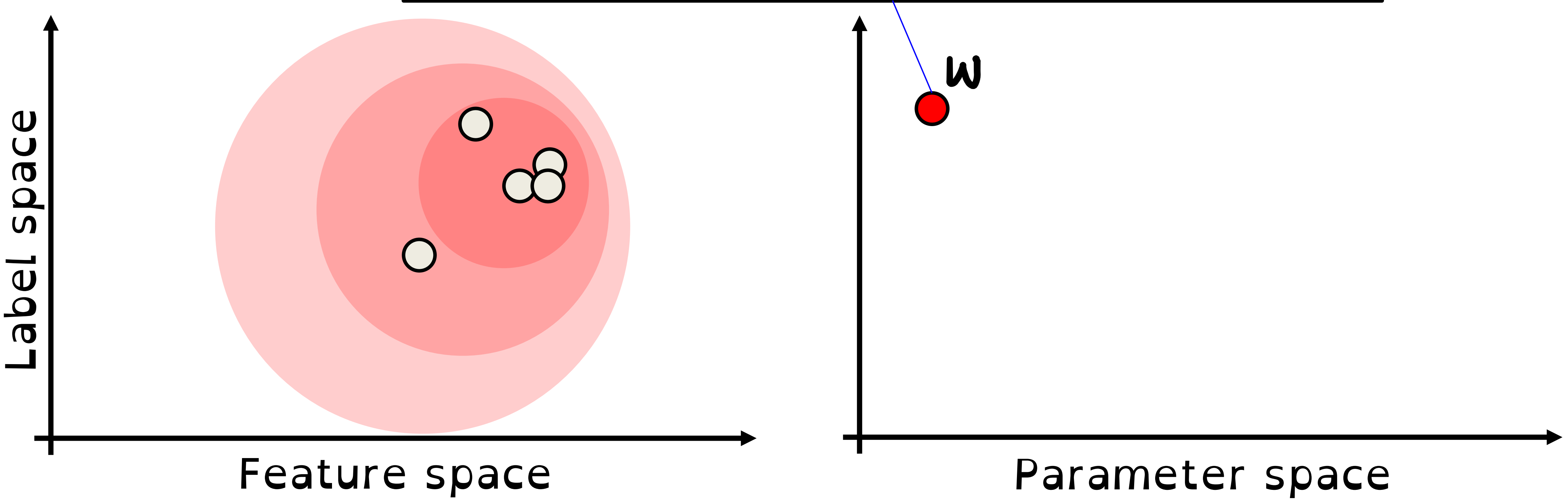
- MLE and MAP are connected by Bayes rule:

$$\underbrace{p(w|D)}_{\text{posterior}} = \frac{p(D|w)p(w)}{p(D)} \propto \underbrace{p(D|w)}_{\text{likelihood}}\underbrace{p(w)}_{\text{prior}}$$

- So MAP maximizes the likelihood p(D|w) times the prior p(w):
  - Prior is our "belief" that 'w' is correct before seeing data.
  - Prior can reflect that complex models are likely to overfit.

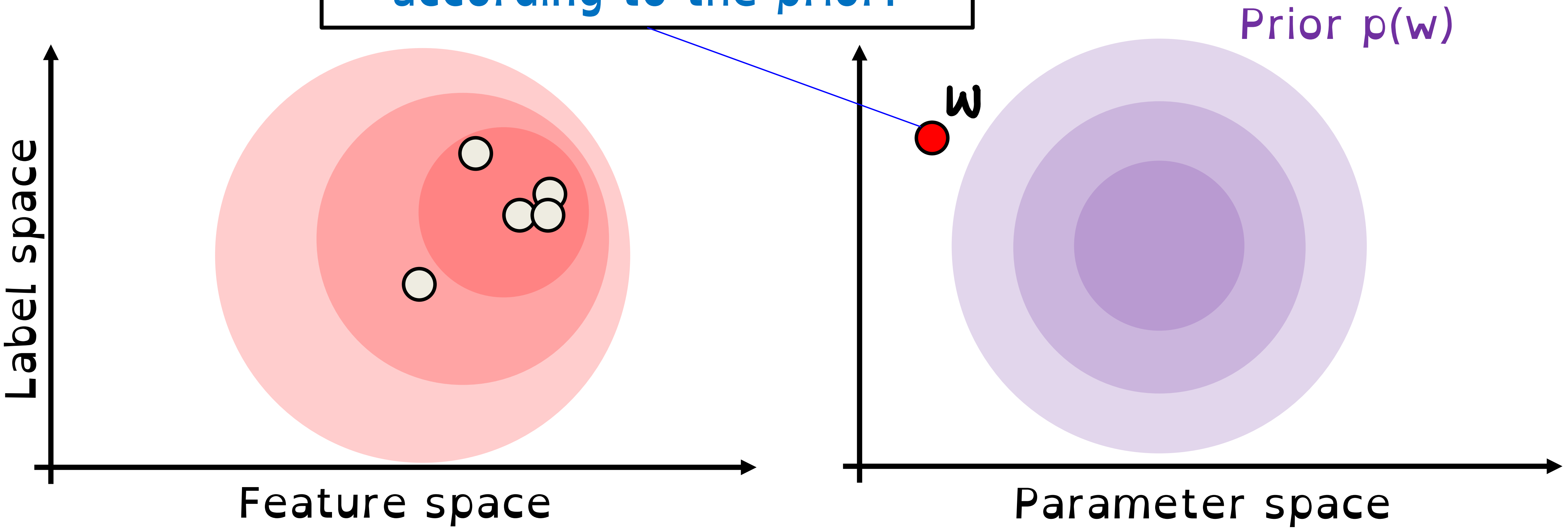
“Prior”

Q: What if this is overfitting?
How do we discourage w from going here?



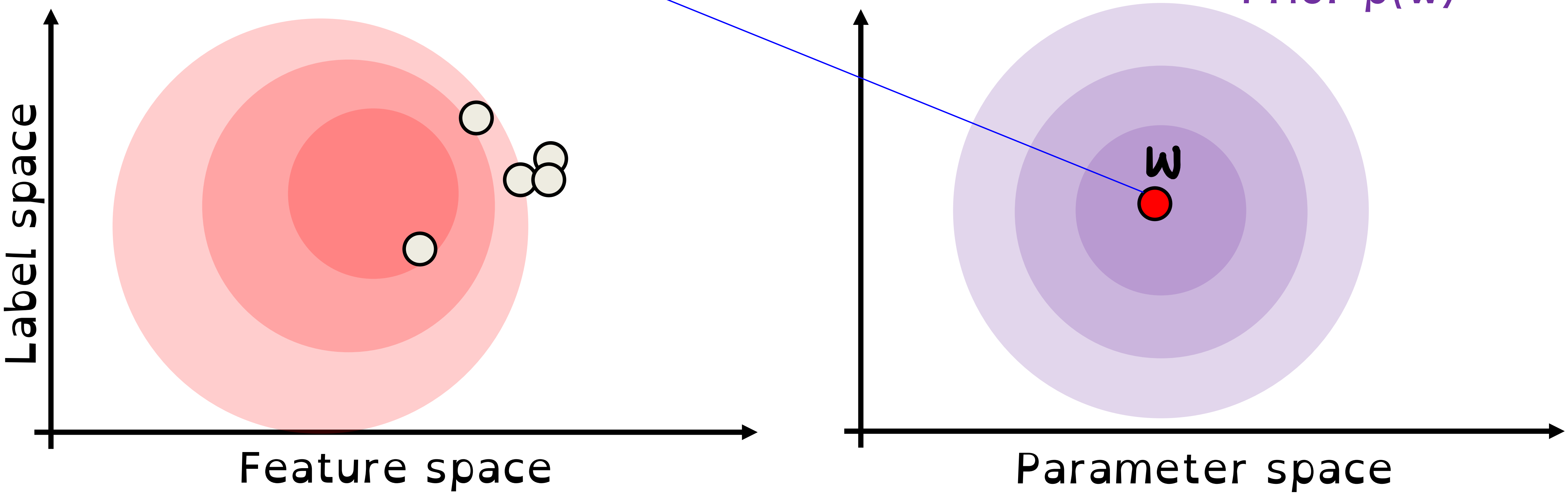
“Prior”

Q: Is this a good value according to the prior?



"Prior"

Q: Is this a good value according to the prior?



Q: Haven't we seen a similar concept before?

MAP Estimation and Regularization

- From Bayes rule, the MAP estimate with IID examples D_i is:

$$\hat{w} \in \underset{w}{\text{argmax}}\left\{p(w \mid D)\right\} \equiv \underset{w}{\text{argmax}}\left\{\prod_{i=1}^{n}\left[p(D_i \mid w)\right]p(w)\right\}$$

- By again taking the negative of the logarithm as before we get:

$$\hat{w} \in \underset{w}{\text{argmin}}\left\{-\sum_{i=1}^{n}\left[\underbrace{\log(p(D_i \mid w))}_{\text{loss}}\right] - \underbrace{\log(p(w))}_{\text{regularizer}}\right\}$$

- So we can view the negative log-prior as a regularizer:
  - Many regularizers are equivalent to negative log-priors.

L2-Regularization and MAP Estimation

- We obtain L2-regularization under an independent Gaussian assumption:

[1] Assume each w_j comes from a Gaussian with mean 0 and variance $1/\lambda$

- This implies that:

Gaussian assumption

[2] $$\underbrace{p(w) = \prod_{j=1}^{d} p(w_j)}_{\text{independence}} \propto \prod_{j=1}^{d} \exp\left(-\frac{\lambda}{2} w_j^2\right) = \exp\left(-\frac{\lambda}{2} \sum_{j=1}^{d} w_j^2\right)$$

$e^\alpha e^\beta = e^{\alpha+\beta}$

- So we have that:

[3] $$-\log(p(w)) = -\log\left(\exp\left(-\frac{\lambda}{2}\|w\|^2\right)\right) + (\text{constant}) = \frac{\lambda}{2}\|w\|^2 + (\text{constant})$$

- With this prior, the MAP estimate with IID training examples would be

[4] $$\hat{w} \in \operatorname{argmin}\{-\log(p(y|X,w)) - \log(p(w))\} \equiv \underset{w}{\operatorname{argmin}}\left\{-\sum_{i=1}^{n}[\log(p(y_i|x_i,w))] + \frac{\lambda}{2}\|w\|^2\right\}$$

MAP Estimation and Regularization

- MAP estimation gives link between probabilities and loss functions.
  - Gaussian likelihood (σ = 1) + Gaussian prior gives L2-regularized least squares.

If $p(y_i \mid x_i, w) \propto \exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)$ $\quad p(w_j) \propto \exp\left(-\frac{\lambda}{2} w_j^2\right)$

then MAP estimation is equivalent to minimizing $f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$

  - Laplace likelihood (σ = 1) + Gaussian prior give L2-regularized robust regression:

If $p(y_i \mid x_i, w) \propto \exp(-|w^T x_i - y_i|)$ $\quad p(w) \propto \exp\left(-\frac{\lambda}{2} w_j^2\right)$

then MAP estimation is equivalent to minimizing $f(w) = \|Xw - y\|_1 + \frac{\lambda}{2}\|w\|^2$

  - As 'n' goes to infinity, effect of prior/regularizer goes to zero.
  - Unlike with MLE, the choice of σ changes the MAP solution for these models.

Summarizing the past few slides

- Many of our loss functions and regularizers have probabilistic interpretations.
  - Laplace likelihood leads to absolute error.
  - Laplace prior leads to L1-regularization.

- The choice of likelihood corresponds to the choice of loss.
  - Our assumptions about how the y_i-values can come from the x_i and 'w'.

- The choice of prior corresponds to the choice of regularizer.
  - Our assumptions about which 'w' values are plausible.

Regularizing Other Models

- We can view priors in other models as regularizers.
- Remember the problem with MLE for naïve Bayes:
  - The MLE of p('lactase' = 1| 'spam') is: count(spam,lactase)/count(spam).
  - But this caused problems if count(spam,lactase) = 0.
- Our solution was Laplace smoothing:
  - Add "+1" to our estimates: (count(spam,lactase)+1)/(counts(spam)+2).
  - This corresponds to a "Beta" prior so Laplace smoothing is a regularizer.

Why do we care about MLE and MAP?

- Unified way of thinking about many of our tricks?
  - Probabilitic interpretation of logistic loss.
  - Laplace smoothing and L2-regularization are doing the same thing.

- Remember our two ways to reduce overfitting in complicated models:
  - Model averaging (ensemble methods).
  - Regularization (linear models).

- "Fully"-Bayesian methods (CPSC 440) combine both of these.
  - Average over all models, weighted by posterior (including regularizer).
  - Can use extremely-complicated models without overfitting.

Losses for Other Discrete Labels

- MLE/MAP gives loss for classification with basic labels:
  - Least squares and absolute loss for regression.
  - Logistic regression for binary labels {“spam”, “not spam”}.
  - Softmax regression for multi-class {“spam”, “not spam”, “important”}.

- But MLE/MAP lead to losses with other discrete labels (bonus):
  - Ordinal: {1 star, 2 stars, 3 stars, 4 stars, 5 stars}.
  - Counts: 602 ‘likes’.
  - Survival rate: 60% of patients were still alive after 3 years.
  - Unbalanced classes: 99.9% of examples are classified as +1.
- Define likelihood of labels, and use NLL as the loss function.

- We can also use ratios of probabilities to define more losses (bonus):
  - Binary SVMs, multi-class SVMs, and “pairwise preferences” (ranking) models.

Coming Up Next

SUMMARY OF PART 3

End of Part 3: Key Concepts

- Linear models predict based on linear combination(s) of features:

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

- We model non-linear effects using a change of basis:
  - Replace d-dimensional x_i with k-dimensional z_i and use $v^T z_i$.
  - Examples include polynomial basis and (non-parametric) RBFs.

- Regression is supervised learning with continuous labels.
  - Logical error measure for regression is squared error:
  - Can be solved as a system of linear equations.

$$f(w) = \frac{1}{2}\|Xw - y\|^2$$

End of Part 3: Key Concepts

- Gradient descent finds local minimum of smooth objectives.
  - Converges to a global optimum for convex functions.
  - Can use smooth approximations (Huber, log-sum-exp)

- Stochastic gradient methods allow huge/infinite 'n'.
  - Though very sensitive to the step-size.

- Kernels let us use similarity between examples, instead of features.
  - Lets us use some exponential- or infinite-dimensional features.

- Feature selection is a messy topic.
  - Classic method is forward selection based on L0-norm.
  - L1-regularization simultaneously regularizes and selects features.

End of Part 3: Key Concepts

- We can reduce over-fitting by using regularization:

$$f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$$

- Squared error is not always right measure:
  - Absolute error is less sensitive to outliers.
  - Logistic loss and hinge loss are better for binary y_i.
  - Softmax loss is better for multi-class y_i.
- MLE/MAP perspective:
  - We can view loss as log-likelihood and regularizer as log-prior.
  - Allows us to define losses based on probabilities.

The Story So Far…

- Part 1: Supervised Learning.
  - Methods based on counting and distances.
- Part 2: Unsupervised Learning.
  - Methods based on counting and distances.
- Part 3: Supervised Learning (just finished).
  - Methods based on linear models and gradient descent.
- Part 4: Unsupervised Learning (next time).
  - Methods based on linear models and gradient descent.

Summary

- Maximum likelihood estimate viewpoint of common models.
  - Objective functions are equivalent to maximizing p(y, X | w) or p(y | X, w).
- MAP estimation directly models p(w | X, y).
  - Gives probabilistic interpretation to regularization.
- Losses for weird scenarios are possible using MLE/MAP:
  - Ordinal labels, count labels, censored labels, unbalanced labels.

- Next time:
  - What 'parts' is your personality made of?

Review Questions

- Q1: How is the likelihood different between supervised and unsupervised learning?
- Q2: How is maximizing +1-ness for +1 examples related to the logistic loss?
- Q3: Why is the argmin of negative log likelihood the same as the argmax of likelihood?
- Q4: How does Bayes' rule connect likelihood and prior?

Discussion: Least Squares and Gaussian Assumption

- Classic justifications for the Gaussian assumption underlying least squares:
  - Your noise might really be Gaussian. (It probably isn't, but maybe it's a good enough approximation.)
  - The central limit theorem (CLT) from probability theory. (If you add up enough IID random variables, the estimate of their mean converges to a Gaussian distribution.)

- I think the CLT justification is wrong as we've never assumed that the x_{ij} are IID across 'j' values. We only assumed that the examples x_i are IID across 'i' values, so the CLT implies that our estimate of 'w' would be a Gaussian distribution under different samplings of the data, but this says nothing about the distribution of y_i given w^Tx_i.

- On the other hand, there are reasons *not* to use a Gaussian assumption, like it's sensitivity to outliers. This was (apparently) what lead Laplace to propose the Laplace distribution as a more robust model of the noise.
- The "student t" distribution (published anonymously by Gosset while working at the Guiness beer company) is even more robust, but doesn't lead to a convex objective.

Binary vs. Multi-Class Logistic

- How does multi-class logistic generalize the binary logistic model?
- We can re-parameterize softmax in terms of (k-1) values of z_c:

$$p(y \mid z_1, z_2, \dots, z_{k-1}) = \frac{\exp(z_y)}{1 + \sum_{c=1}^{k-1} \exp(z_c)} \text{ if } y \neq k \quad \text{and} \quad p(y \mid z_1, z_2, \dots, z_{k-1}) = \frac{1}{1 + \sum_{c=1}^{k-1} \exp(z_c)} \text{ if } y = k$$

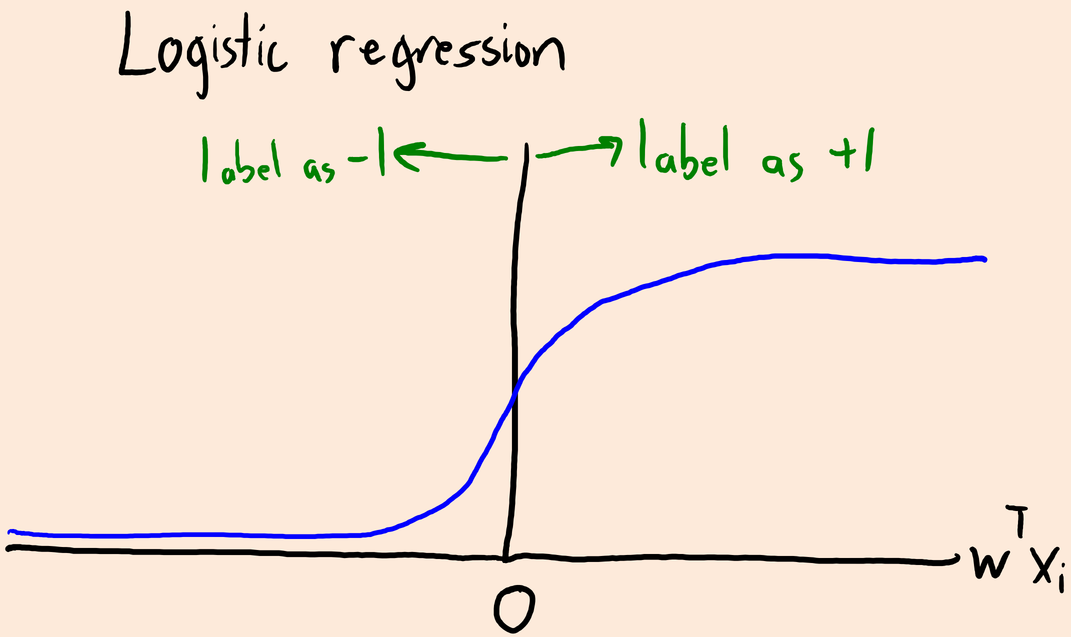
  - This is due to the "sum to 1" property (one of the z_c values is redundant).
  - So if k=2, we don't need a z_2 and only need a single 'z'.
  - Further, when k=2 the probabilities can be written as:

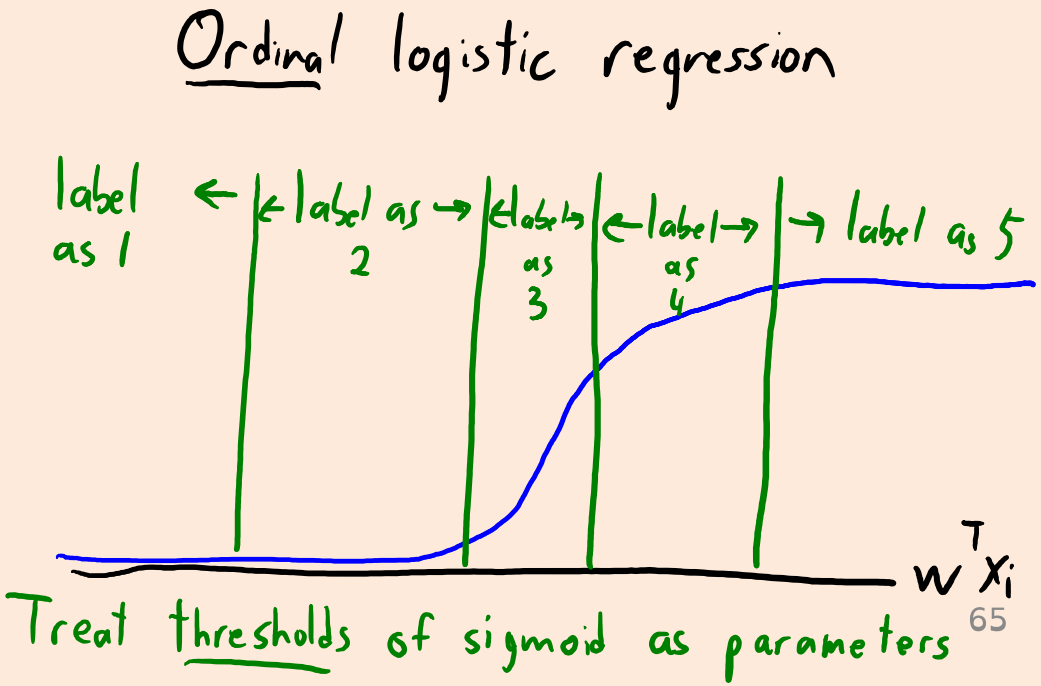
$$p(y=1 \mid z) = \frac{\exp(z)}{1+\exp(z)} = \frac{1}{1+\exp(-z)} \qquad p(y=2 \mid z) = \frac{1}{1+\exp(z)}$$

  - Renaming '2' as '-1', we get the binary logistic regression probabilities.

Ordinal Labels

- Ordinal data: categorical data where the order matters:
  - Rating hotels as {‘1 star’, ‘2 stars’, ‘3 stars’, ‘4 stars’, ‘5 stars’}.
  - Softmax would ignore order.
- Can use ‘ordinal logistic regression’.





Count Labels

- Count data: predict the number of times something happens.
  - For example, y_i = "602" Facebook likes.
- Softmax requires finite number of possible labels.
- We probably don't want separate parameter for '654' and '655'.
- Poisson regression: use probability from Poisson count distribution.
  - Many variations exist, a lot of people think this isn't the best likelihood.

Censored Survival Analysis (Cox Partial Likelihood)

- Censored survival analysis:
  - Target y_i is last time at which we know person is alive.
    - But some people are still alive (so they have the same y_i values).
    - The y_i values (time at which they die) are "censored".
  - We use $v_i=0$ is they are still alive and otherwise we set $v_i = 1$.

- Cox partial likelihood assumes "instantaneous" rate of dying depends on x_i but not on total time they've been alive (not that realistic). Leads to likelihood of the "censored" data of the form:

$$p(y_i, v_i \mid x_i, w) = \exp(v_i w^T x_i)\exp(-y_i \exp(w^T x_i))$$

- There are many extensions and alternative likelihoods.

Other Parsimonious Parameterizations

- Sigmoid isn't the way to model a binary $p(y_i \mid x_i, w)$:
  - Probit (uses CDF of normal distribution, very similar to logistic).
  - Noisy-Or (simpler to specify probabilities by hand).
  - Extreme-value loss (good with class imbalance).
  - Cauchit, Gosset, and many others exist…

Unbalanced Training Sets

- Consider the case of binary classification where your training set has 99% class -1 and only 1% class +1.
  - This is called an "unbalanced" training set
- Question: is this a problem?
- Answer: it depends!
  - If these proportions are representative of the test set proportions, and you care about both types of errors equally, then "no" it's not a problem.
    - You can get 99% accuracy by just always predicting -1, so ML can only help with the 1%.
  - But it's a problem if the test set is not like the training set (e.g. your data collection process was biased because it was easier to get -1's)
  - It's also a problem if you care more about one type of error, e.g. if mislabeling a +1 as a -1 is much more of a problem than the opposite
    - For example if +1 represents "tumor" and -1 is "no tumor"

Unbalanced Training Sets

- This issue comes up a lot in practice!
- How to fix the problem of unbalanced training sets?
  - Common strategy is to build a "weighted" model:
    - Put higher weight on the training examples with y_i=+1.

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

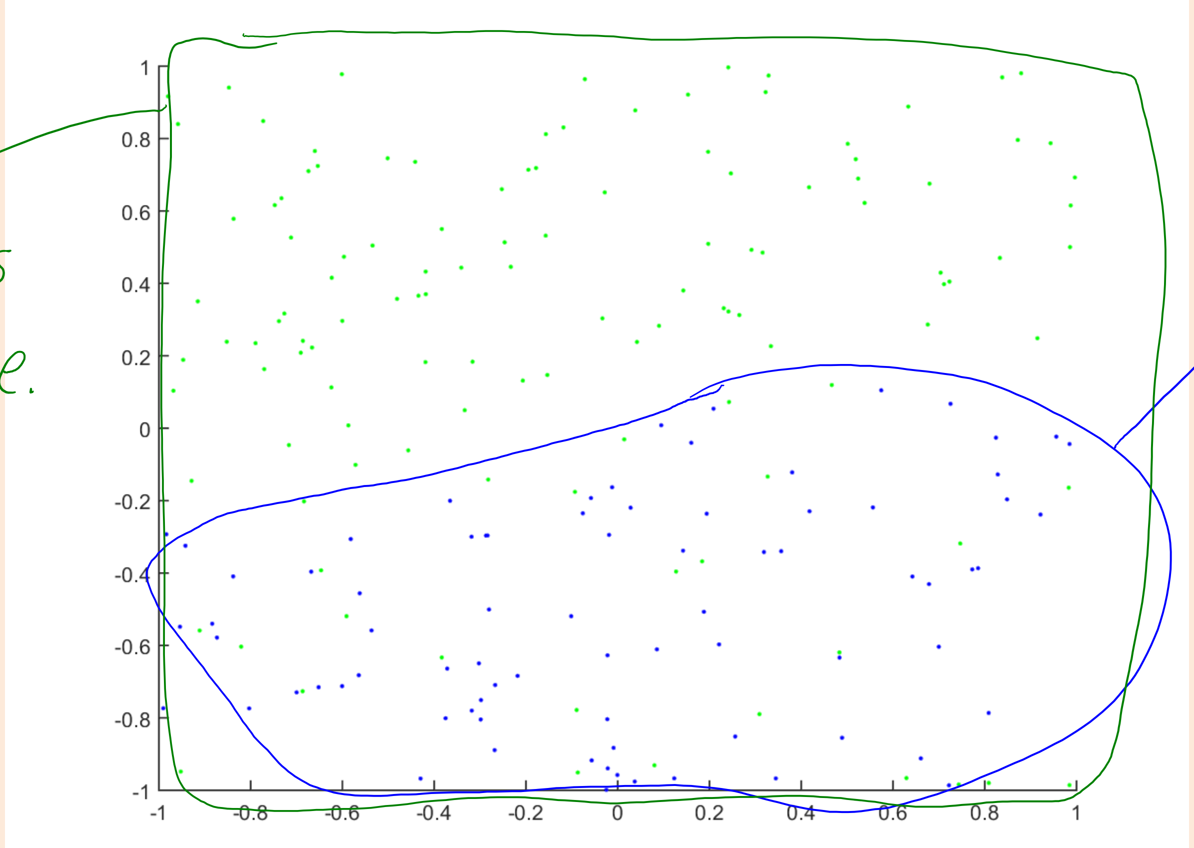
Make this weight bigger for under-represented class

    - This is equivalent to replicating those examples in the training set.
    - You could also subsample the majority class to make things more balanced.
    - Boostrap: create a dataset of size 'n' where n/2 are sampled from +1, n/2 from -1.
  - Another approach is to try to make "fake" data to fill in minority class.
  - Another option is to change to an asymmetric loss function (next slides) that penalizes one type of error more than the other.
  - Some discussion of different methods here.

Unbalanced Data and Extreme-Value Loss

- Consider binary case where:
  - One class overwhelms the other class ('unbalanced' data).
  - Really important to find the minority class (e.g., minority class is tumor).

"majority" class is everywhere.



important "minority" class

Unbalanced Data and Extreme-Value Loss

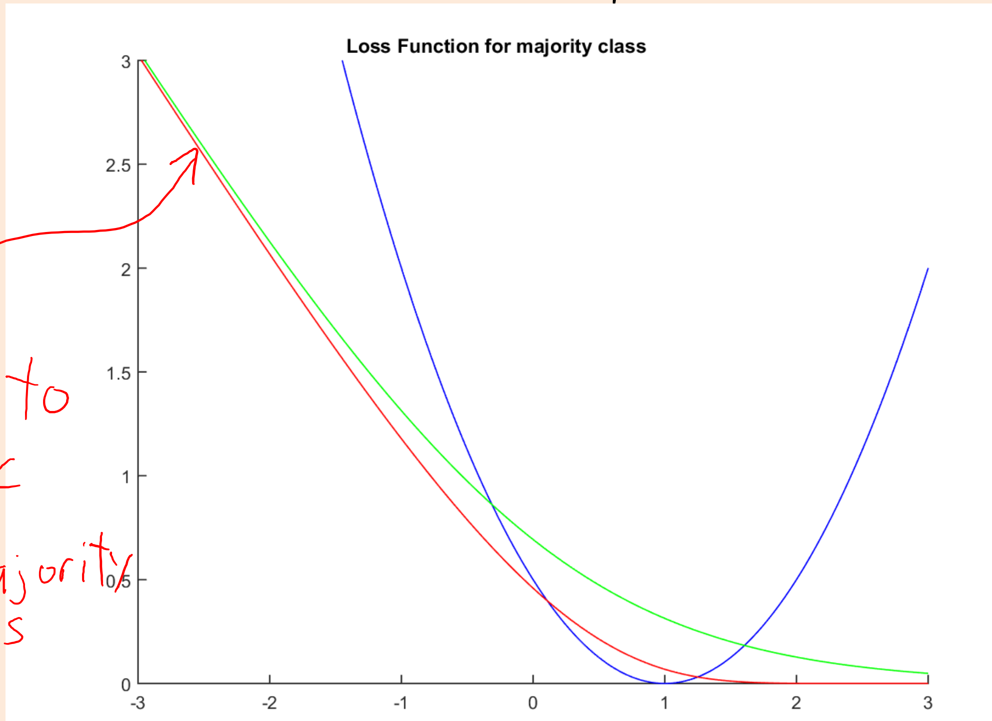
- Extreme-value distribution:

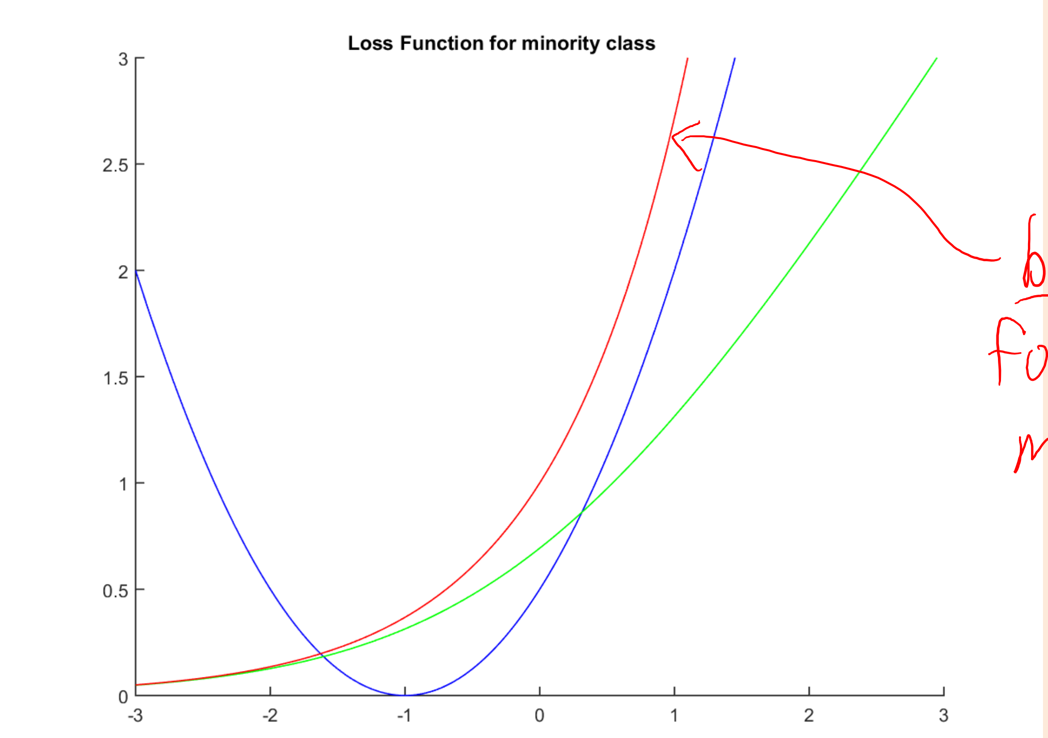
$p(y_i = +1 \mid \hat{y}_i) = 1 - \exp(-\exp(\hat{y}_i))$ [+1 is majority class]

asymmetric

To make it a probability, $p(y_i = -1 \mid \hat{y}_i) = \exp(-\exp(\hat{y}_i))$

Similar to logistic for majority class





big penalty for getting minority class wrong.

Unbalanced Data and Extreme-Value Loss

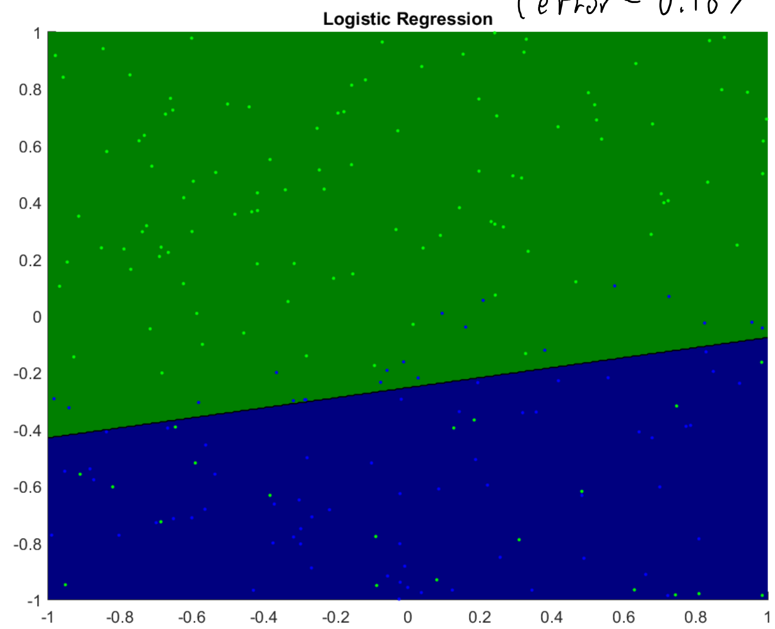
- Extreme-value distribution:

$$p(y_i = +1 \mid \hat{y}_i) = 1 - \exp(-\exp(\hat{y}_i)) \quad [+1 \text{ is majority class}]$$

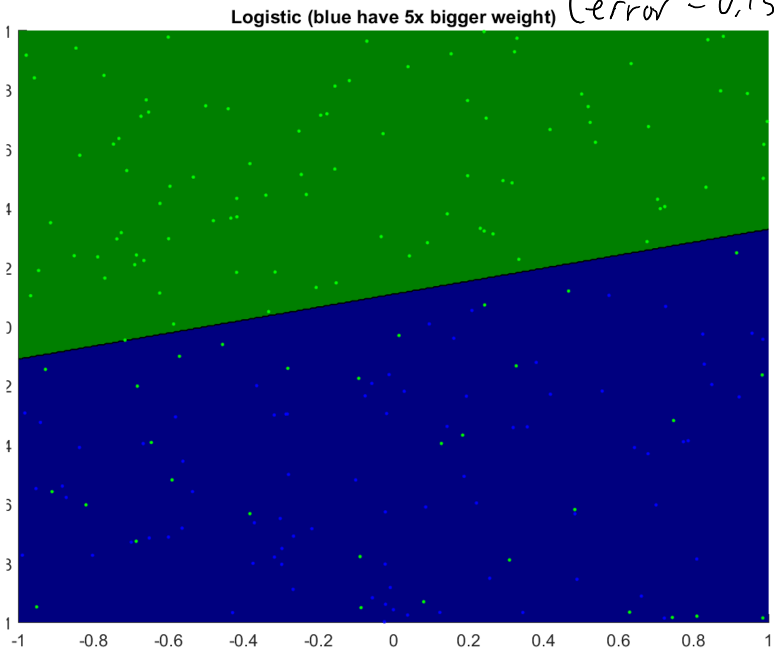
asymmetric

To make it a probability, $p(y_i = -1 \mid \hat{y}_i) = \exp(-\exp(\hat{y}_i))$

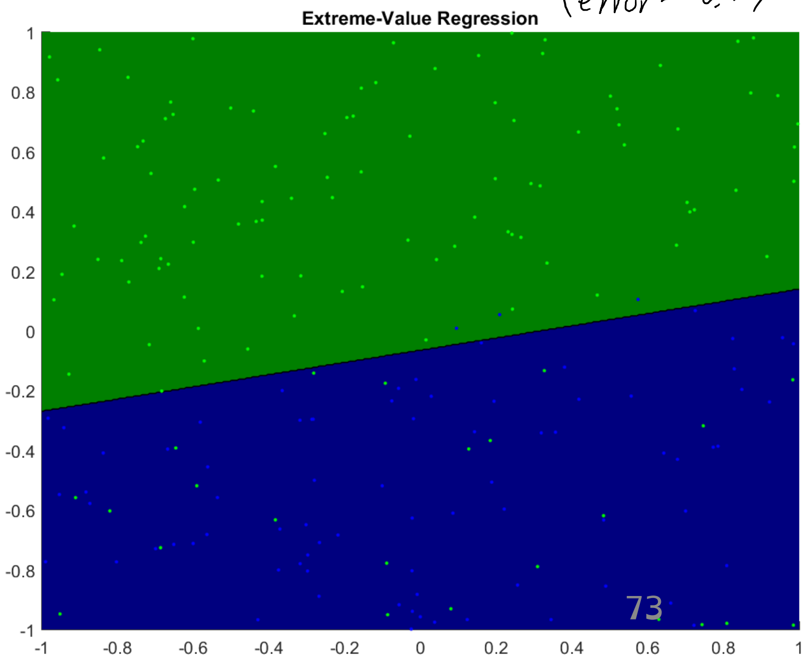
(error = 0.18)



(error = 0.15)



(error = 0.13)



Loss Functions from Probability Ratios

- We've seen that loss functions can come from probabilities:
  - Gaussian => squared loss, Laplace => absolute loss, sigmoid => logistic.
- Most other loss functions can be derived from probability ratios.
  - Example: sigmoid => hinge.

$$p(y_i \mid x_i, w) = \frac{1}{1 + \exp(-y_i w^T x_i)} = \frac{\exp(\frac{1}{2} y_i w^T x_i)}{\underbrace{\exp(\frac{1}{2} y_i w^T x_i) + \exp(-\frac{1}{2} y_i w^T x_i)}_{\text{Same normalizing constant for } y_i = +1 \text{ and } x_i = -1}} \propto \exp(\tfrac{1}{2} y_i w^T x_i)$$

Loss Functions from Probability Ratios

- We've seen that loss functions can come from probabilities:
  - Gaussian => squared loss, Laplace => absolute loss, sigmoid => logistic.
- Most other loss functions can be derived from probability ratios.
  - Example: sigmoid => hinge.

$p(y_i \mid x_i, w) \propto \exp(\frac{1}{2} y_i w^T x_i)$

To classify y_i correctly, it's sufficient to have $\underbrace{\frac{p(y_i \mid x_i, w)}{p(-y_i \mid x_i, w)}} \geq \beta$ for some 'β' > 1

Notice that normalizing constant doesn't matter:

$$\frac{\exp(\frac{1}{2} y_i w^T x_i)}{\exp(-\frac{1}{2} y_i w^T x_i)} \geq \beta$$

Loss Functions from Probability Ratios

- We've seen that loss functions can come from probabilities:
  - Gaussian => squared loss, Laplace => absolute loss, sigmoid => logistic.
- Most other loss functions can be derived from probability ratios.
  - Example: sigmoid => hinge.

$p(y_i \mid x_i, w) \propto \exp(\frac{1}{2} y_i w^T x_i)$

We need: $\frac{\exp(\frac{1}{2} y_i w^T x_i)}{\exp(-\frac{1}{2} y_i w^T x_i)} \geq \beta$

Take log:

$\log\left(\frac{\exp(\frac{1}{2} y_i w^T x_i)}{\exp(-\frac{1}{2} y_i w^T x_i)}\right) \geq \log(\beta) \iff \frac{1}{2} y_i w^T x_i + \frac{1}{2} y_i w^T x_i \geq \log(\beta)$

$\Updownarrow$

$y_i w^T x_i \geq 1$ (if we choose $\log(\beta) = 1$)

Loss Functions from Probability Ratios

- We've seen that loss functions can come from probabilities:
  - Gaussian => squared loss, Laplace => absolute loss, sigmoid => logistic.
- Most other loss functions can be derived from probability ratios.
  - Example: sigmoid => hinge.

$$p(y_i \mid x_i, w) \propto \exp\left(\tfrac{1}{2} y_i w^T x_i\right)$$

We need: $\dfrac{\exp\left(\frac{1}{2} y_i w^T x_i\right)}{\exp\left(-\frac{1}{2} y_i w^T x_i\right)} \geq \beta$

Or equivalently:

$y_i w^T x_i \geq 1$ (for $\beta = \exp(1)$)

Define a loss function by amount of constraint violation:

$$\max\{\underbrace{0}_{\text{when } 1 - y_i w^T x_i \leq 0},\ \underbrace{1 - y_i w^T x_i}_{\text{when } 1 - y_i w^T x_i \geq 0}\}$$

We get SVMs by looking at regularized average loss:

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

Loss Functions from Probability Ratios

- General approach for defining losses using probability ratios:
  1. Define constraint based on probability ratios.
  2. Minimize violation of logarithm of constraint.
- Example: softmax => multi-class SVMs.

Assume: $p(y_i = c \mid x_i, w) \propto \exp(w_c^T x_i)$

Want: $\frac{p(y_i \mid x_i, w)}{p(y_i = c' \mid x_i, w)} \geq \beta$ for all c' and some $\beta > 1$

For $\beta = \exp(1)$ equivalent to

$$w_{y_i}^T x_i - w_{c'}^T x_i \geq 1$$

for all $c' \neq y_i$

Option 1: penalize all violations:

$$\sum_{c'=1}^{K} \max\{0, 1 - w_{y_i}^T x_i + w_{c'}^T x_i\}$$

Option 2: penalize only max violation:

$$\max_{c' \neq c} \{\max\{0, 1 - w_{y_i}^T x_i + w_{c'}^T x_i\}\}$$

Supervised Ranking with Pairwise Preferences

- Ranking with pairwise preferences:
  - We aren't given any explicit y_i values.
  - Instead we're given list of objects (i,j) where $y_i > y_j$.

Assume $p(y_i \mid X, w) \propto \exp(w^T x_i)$ is probability that object 'i' has highest rank.

Want: $\frac{p(y_i \mid X, w)}{p(y_j \mid X, w)} \geq \beta$ for all preferences (i,j)

For $\beta = \exp(1)$ equivalent to

$$w^T x_i - w^T x_j \geq 1$$

for preferences (i,j)

We can use $f(w) = \sum_{(i,j) \in R} \max\{0, 1 - w^T x_i + w^T x_j\}$

This approach can also be used to define losses for total/partial orderings. (but this information is hard to get)