

Lecture 09

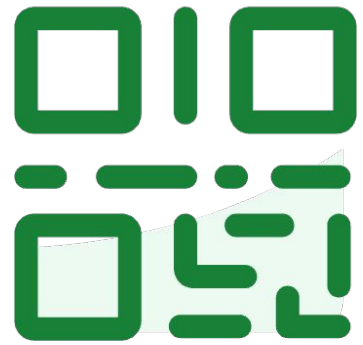
Logistic Regression (2)

Logistic Regression Optimization, Decisions, and Evaluation

EECS 189/289, Fall 2026 @ UC Berkeley

Joseph E. Gonzalez and Narges Norouzi

Do not edit
How to change the
design



**Join at slido.com
#1643396**

① The Slido app must be installed on every computer you're presenting from

slido

Roadmap

- The Logistic Regression Model
- Logistic Regression Objective
- Logistic Regression Optimization
- Regularization with Logistic Regression
- Model Evaluation
- Multi-Class Classification



The Logistic Regression Model

- The Logistic Regression Model
- Logistic Regression Objective
- Logistic Regression Optimization
- Regularization with Logistic Regression
- Model Evaluation
- Multi-Class Classification



1643396

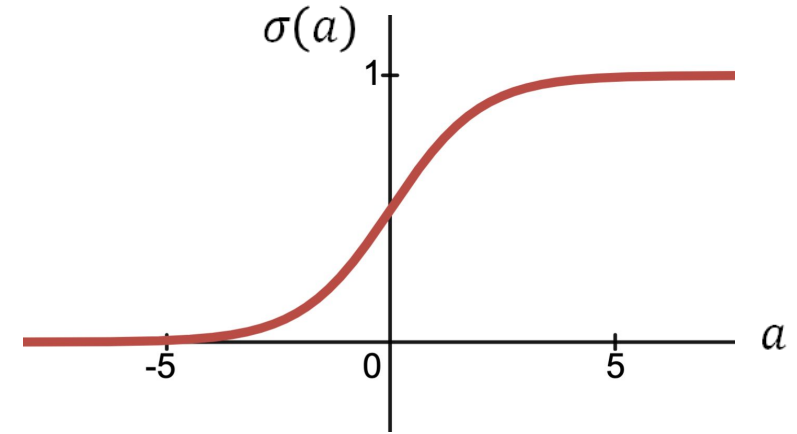


1643396

The Sigmoid (σ) Function

The S-shaped curve is formally known as the **sigmoid function**.

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$



Reflection/
Symmetry

$$1 - \sigma(a) = \frac{e^{-a}}{1 + e^{-a}} = \sigma(-a)$$

Domain
 $-\infty < a < +\infty$

Range
 $0 < \sigma(a) < 1$

Inverse

$$a = \sigma^{-1}(p) = \log\left(\frac{p}{1-p}\right)$$

Logit Function

Derivative

$$\frac{d}{da} \sigma(a) = \sigma(a)(1 - \sigma(a)) = \sigma(a)\sigma(-a)$$



Which of the following are properties of the sigmoid function, $\sigma(\mathbf{x})$?

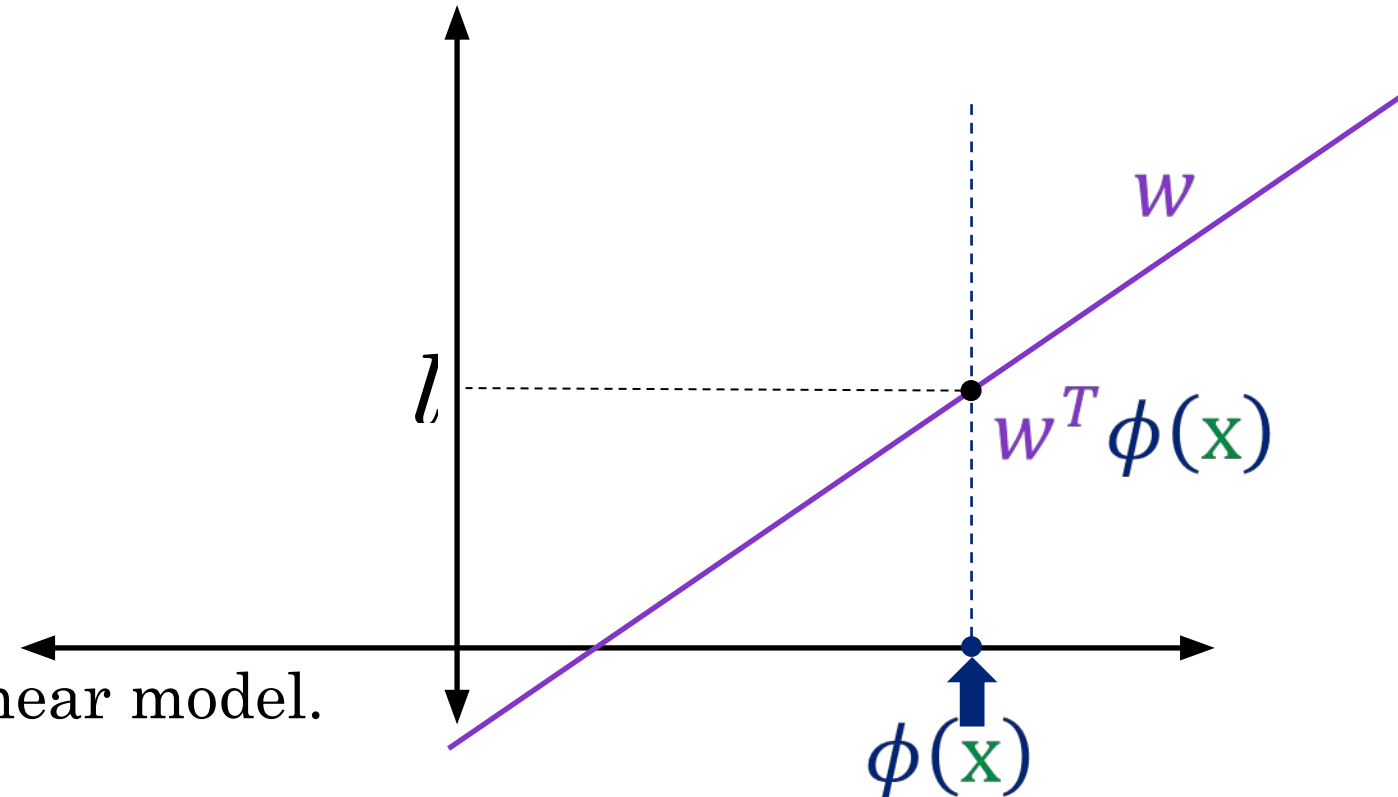
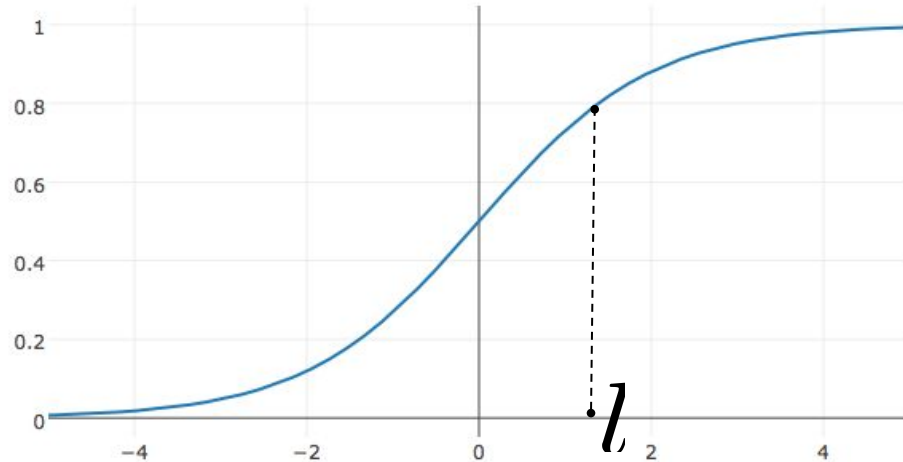
Logistic Regression



1643396

Linear Model

Model: $P(C_1 | \phi(\mathbf{x})) = \underbrace{\sigma}_{\text{Sigmoid}} \left(\underbrace{w^T \phi(\mathbf{x})}_{\text{Linear}} \right) = \frac{1}{1 + \exp(-w^T \phi(\mathbf{x}))}$

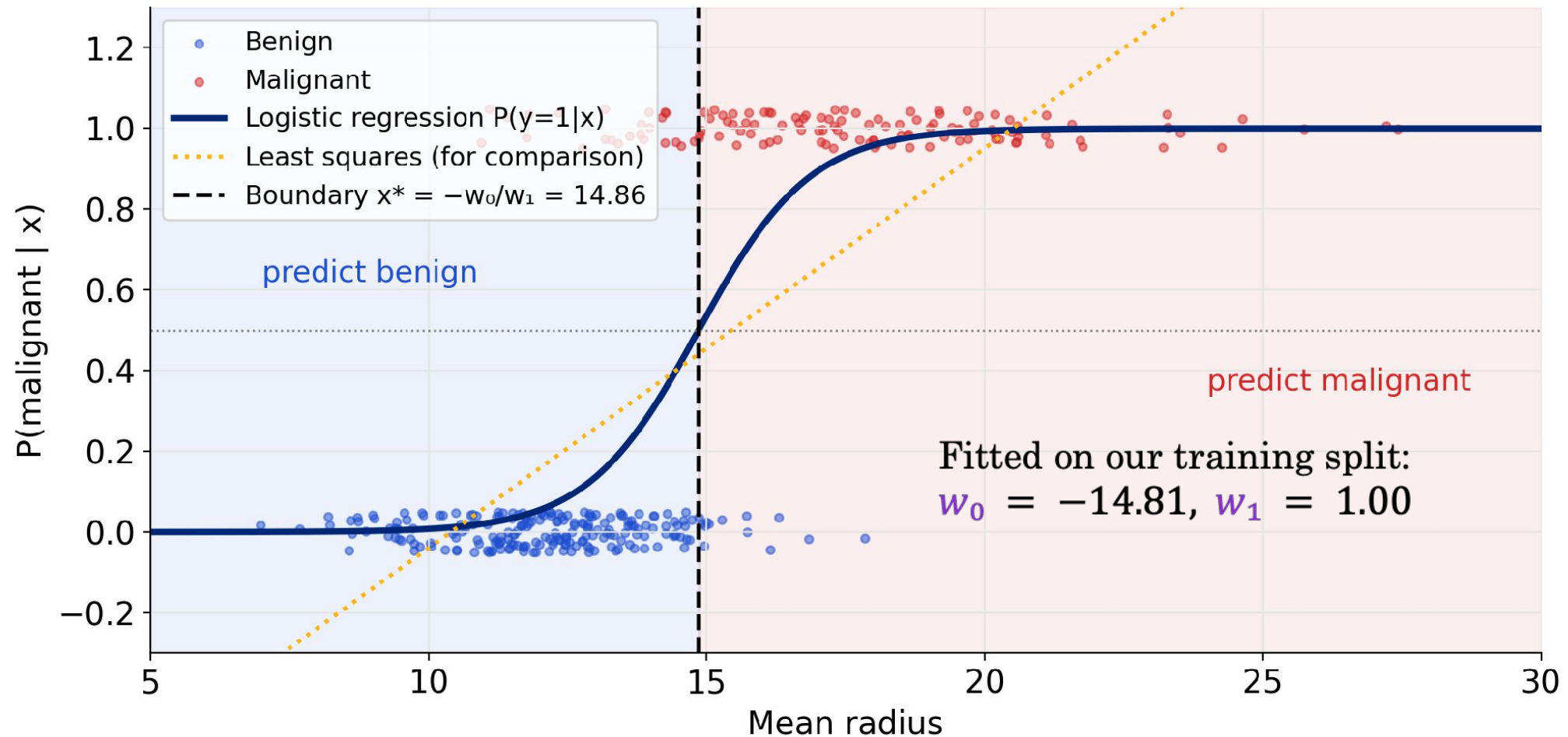


Generalized Linear Model:
Non-linear transformation of a linear model.

Logistic Regression and Decision Boundary



1643396



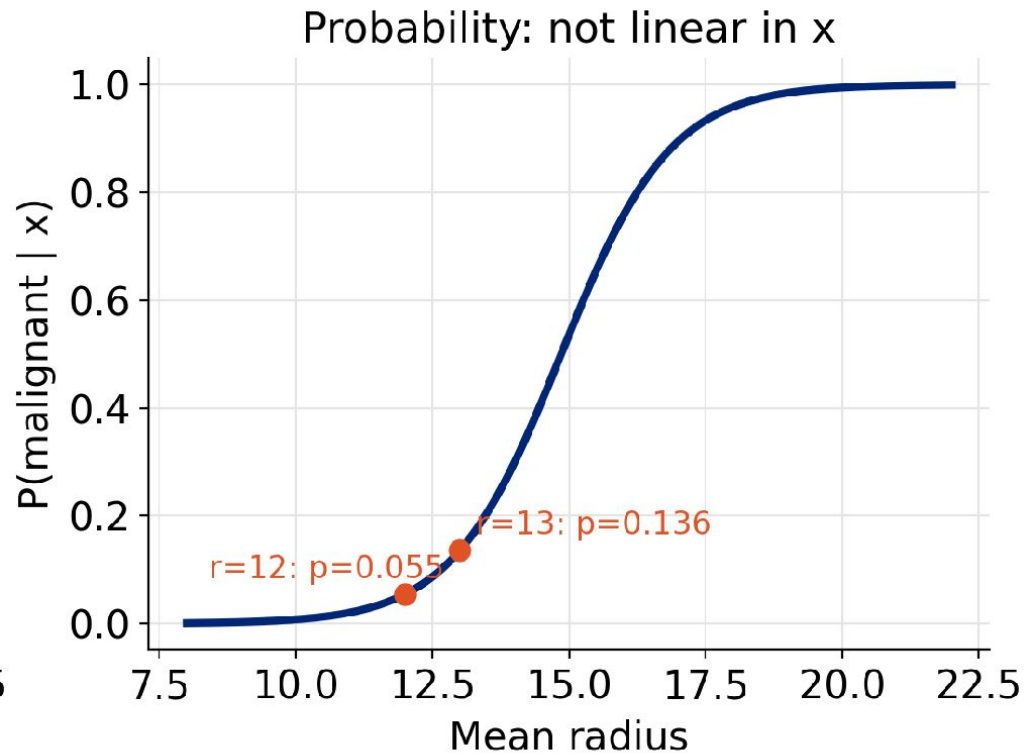
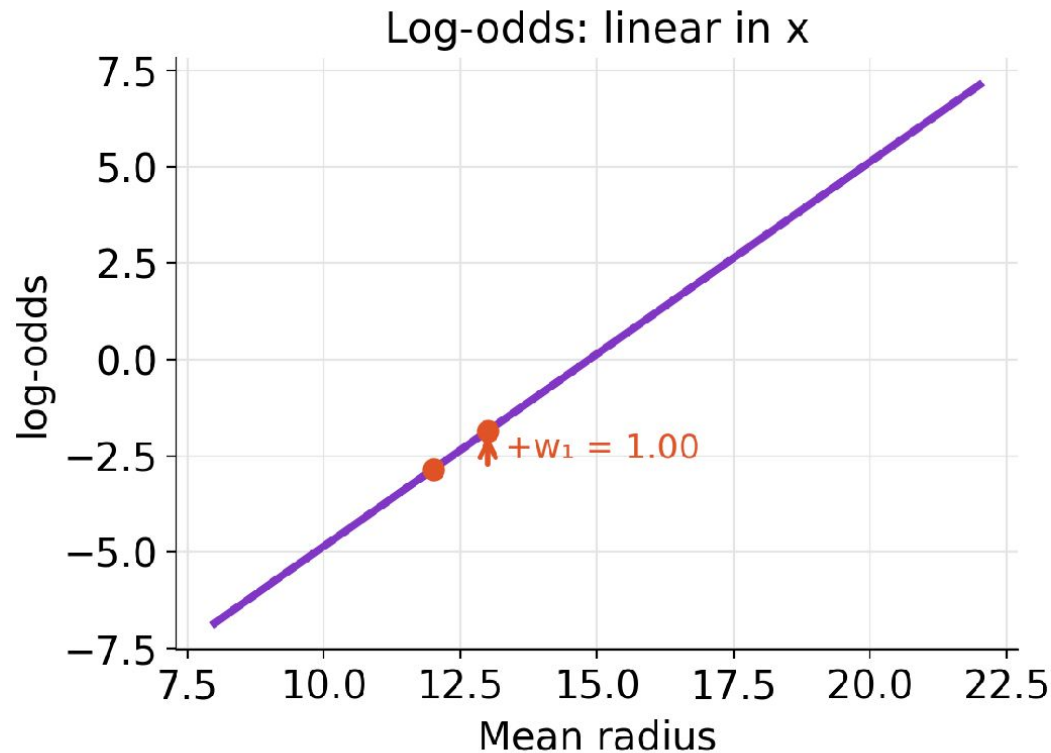
$$\sigma(w_0 + w_1 x) = 0.5 \Leftrightarrow w_0 + w_1 x = 0 \Leftrightarrow x^* = -\frac{w_0}{w_1} = 14.86$$



1643396

Reading the Weights: Odds

- The model says $\log \frac{p}{1-p} = \mathbf{w_0} + \mathbf{w_1}x$. One more unit of x adds $\mathbf{w_1}$ to the log-odds, and it **multiplies the odds** by $\exp(\mathbf{w_1}) = \exp(\mathbf{1.00}) = \mathbf{2.71}$. It does not multiply the probability.
- Radius 12 $\rightarrow$ 13: $p = 0.055 \rightarrow 0.136 (\times 2.48)$

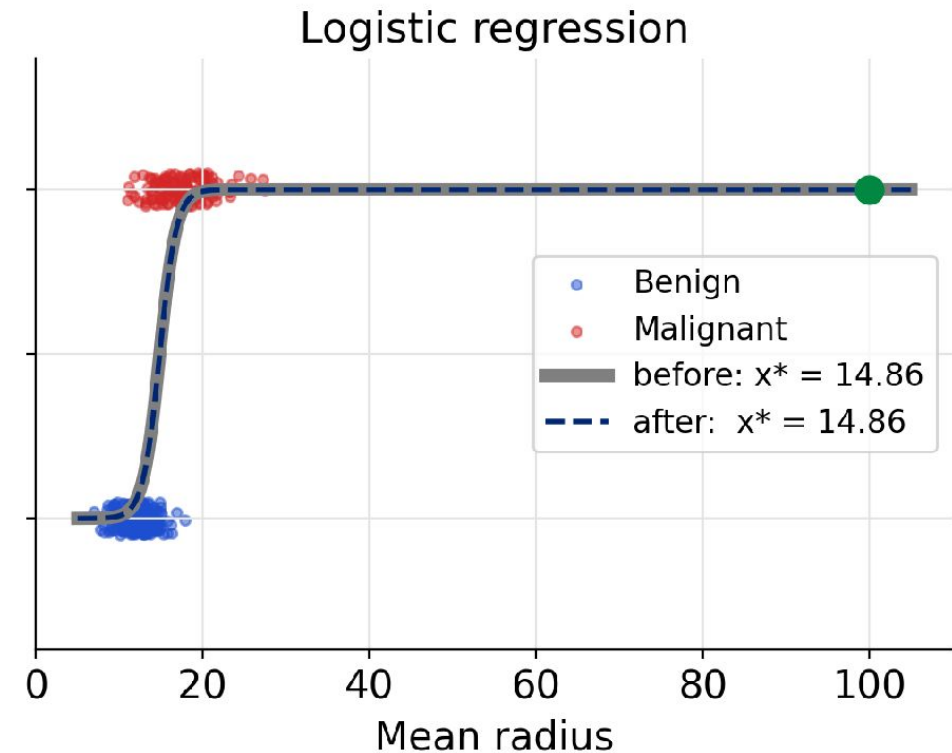
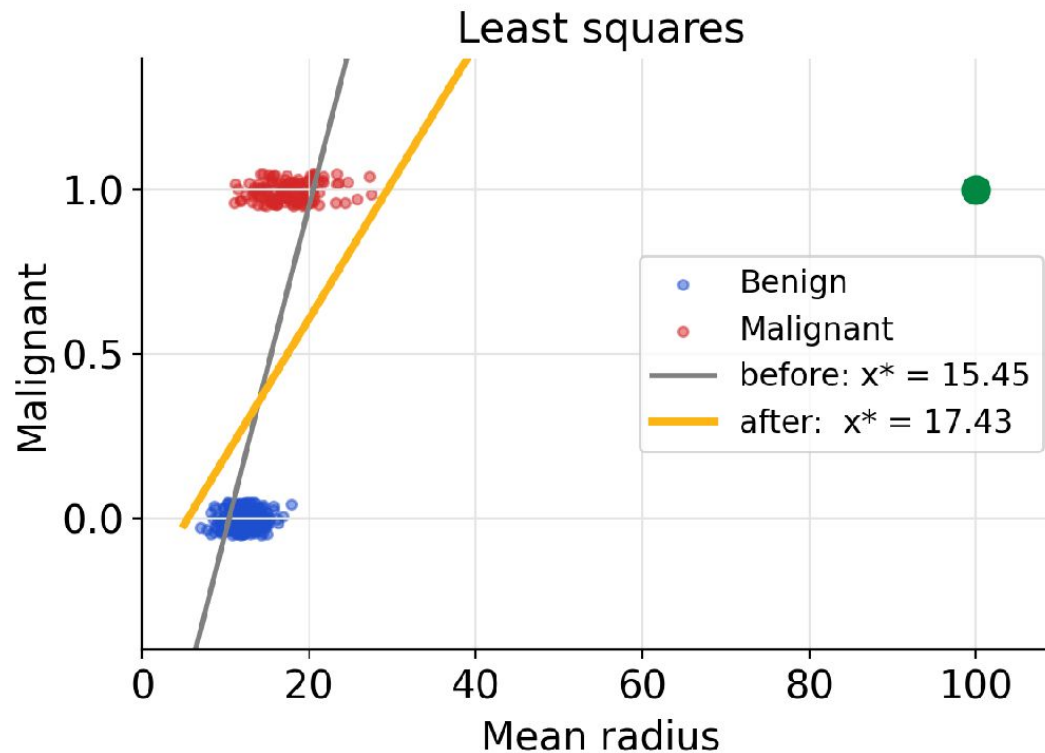


Logistic Regression Ignores Easy Points



1643396

- Same extra tumor at radius 100 as before.
- Least squares boundary: $15.45 \rightarrow 17.43$. Logistic regression boundary: $14.86 \rightarrow 14.86$.
- A confident, correct point has $\sigma(\cdot) \approx 1$, so it adds almost nothing to the loss we are about to define.



Logistic Regression Objective

- The Logistic Regression Model
- Logistic Regression Objective
- Logistic Regression Optimization
- Regularization with Logistic Regression
- Model Evaluation
- Multi-Class Classification



1643396



1643396

The Logistic Regression Objective (MLE)

- We can adopt the same **maximum likelihood estimation**.
- Define target variable $t \in \{0, 1\}$, with $t = 1$ representing class C_1 and $t = 0$ for class C_2 .
- The likelihood of t given $\mathbf{x}$ under our model $p(C_1 | \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x})$:

$$p(t | \mathbf{w}, \mathbf{x}) = \underbrace{\sigma(\mathbf{w}^\top \mathbf{x})}_t [1 - \underbrace{\sigma(\mathbf{w}^\top \mathbf{x})}_{1-t}]^{1-t}$$

for $t_n = 1$, only
this term stays

for $t_n = 0$, only
this term stays



The Logistic Regression Objective (MLE)

- We can adopt the same **maximum likelihood estimation**.
- Define target variable $t \in \{0, 1\}$, with $t = 1$ representing class C_1 and $t = 0$ for class C_2 .
- The likelihood of t given $\mathbf{x}$ under our model $p(C_1|\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x})$:

$$p(t|\mathbf{w}, \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x})^t [1 - \sigma(\mathbf{w}^\top \mathbf{x})]^{1-t}$$

- For a dataset $\mathcal{D} = \{\mathbf{x}_n, t_n\}$, where $y_n = \sigma(\mathbf{w}^\top \mathbf{x})$ the likelihood function can be written as:

$$p(t|\mathbf{x}, \mathbf{w}, \mathcal{D}) = \prod_{n=1}^N y_n^{t_n} (1 - y_n)^{1 - t_n}$$

The Logistic Regression Objective (MLE)



1643396

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \mathcal{D}) = \prod_{n=1}^N y_n^{t_n} (1 - y_n)^{1 - t_n}$$

We can define an error function by taking the **negative logarithm** of the **likelihood**, which gives the **cross-entropy error function**.

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}, \mathcal{D}) = -\sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$

Cross-Entropy: Two Error Functions In One!



1643396

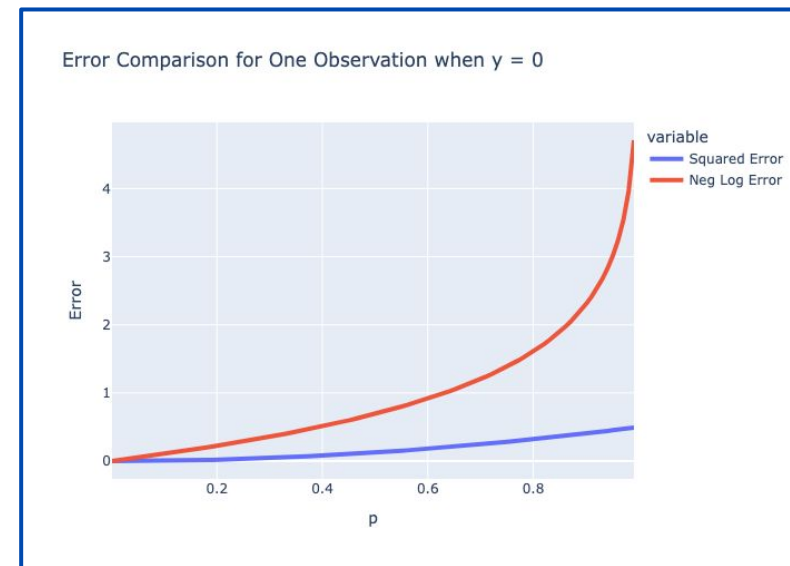
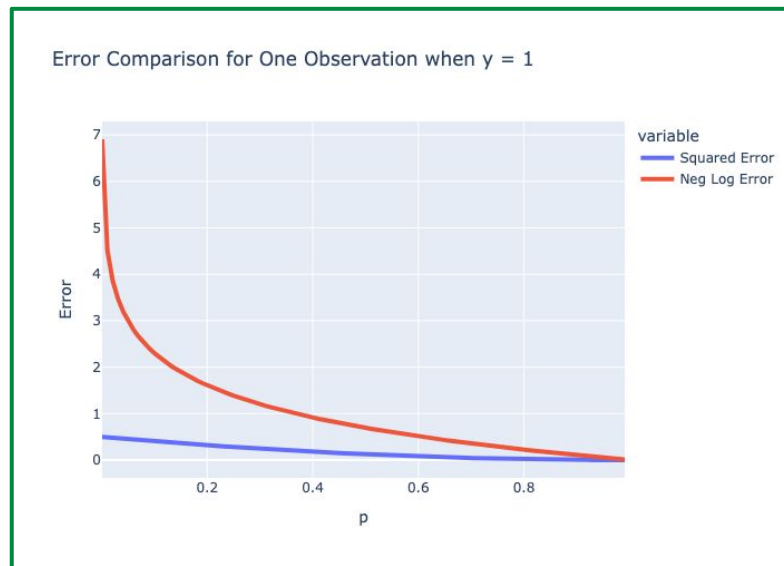
The Cross-Entropy (CE) function is often written more compactly as the **product of the label** and **the log probability**, summed over both classes.

$$\text{CE error} = \begin{cases} -\ln y_n, & \text{if } t_n = 1 \\ -\ln 1 - y_n & \text{if } t_n = 0 \end{cases}$$

$$-\underbrace{[t_n \ln y_n]}_{\text{for } t_n = 1, \text{ only this term stays}} + \underbrace{(1 - t_n) \ln(1 - y_n)}_{\text{for } t_n = 0, \text{ only this term stays}}$$

for $t_n = 1$, only this term stays

for $t_n = 0$, only this term stays



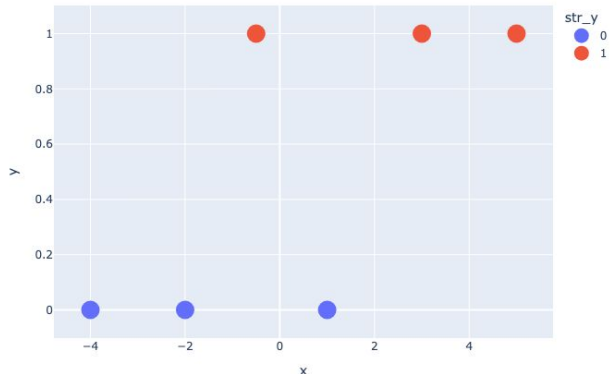
Unbounded for wrong-way certainty

Convexity By Picture



1643396

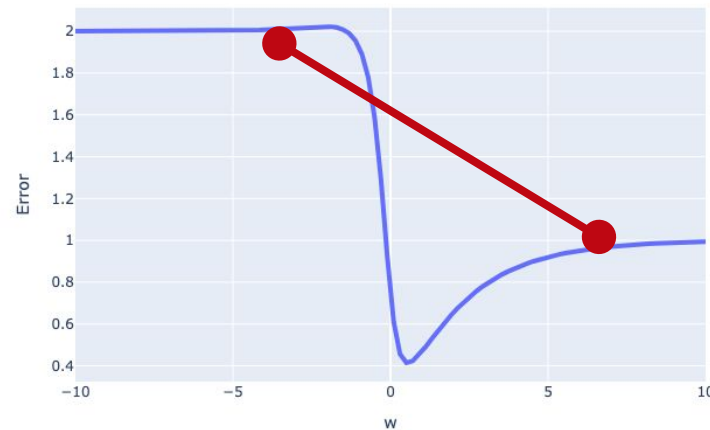
$$w = \underset{w}{\operatorname{argmin}} - \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$



	x	y
0	-4.0	0
1	-2.0	0
2	-0.5	1
3	1.0	0
4	3.0	1
5	5.0	1

Squared Error Surface

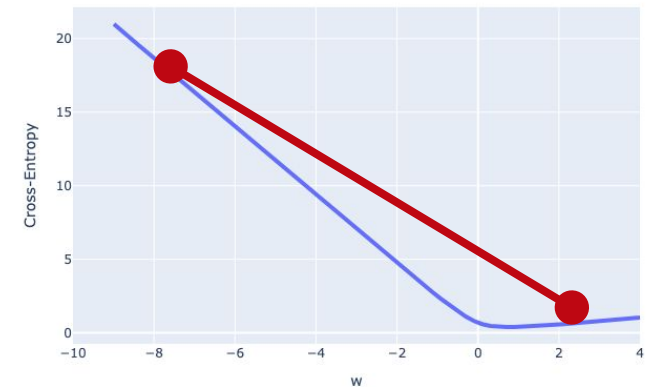
Error on Toy Classification Data



A straight line crosses the curve Non-convex

Cross-Entropy Error Surface

Cross-Entropy on Toy Classification Data



Convex!

Logistic Regression Optimization

- The Logistic Regression Model
- Logistic Regression Objective
- Logistic Regression Optimization
- Regularization with Logistic Regression
- Model Evaluation
- Multi-Class Classification



1643396



1643396

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^N \left[\underbrace{t_n \ln y_n}_{\frac{t_n}{y_n} \frac{\partial y_n}{\partial \mathbf{w}}} + \underbrace{(1 - t_n) \ln(1 - y_n)}_{\frac{(1 - t_n)}{1 - y_n} \frac{\partial(1 - y_n)}{\partial \mathbf{w}}} \right]$$

Derivative wrt $\mathbf{w}$
using chain rule:

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = -\sum_{n=1}^N \left[\frac{t_n}{y_n} \frac{\partial y_n}{\partial \mathbf{w}} + \frac{(1 - t_n)}{1 - y_n} \frac{\partial(1 - y_n)}{\partial \mathbf{w}} \right]$$

$$= -\sum_{n=1}^N \left[\frac{t_n}{y_n} \frac{\partial y_n}{\partial \mathbf{w}} - \frac{(1 - t_n)}{1 - y_n} \frac{\partial y_n}{\partial \mathbf{w}} \right]$$

$$= -\sum_{n=1}^N \left[\frac{t_n}{y_n} - \frac{(1 - t_n)}{1 - y_n} \right] \frac{\partial y_n}{\partial \mathbf{w}}$$

$$\frac{\partial(1 - y_n)}{\partial \mathbf{w}} = -\frac{\partial y_n}{\partial \mathbf{w}}$$



1643396

$$\begin{aligned} E(\mathbf{w}) &= -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)] \\ \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} &= -\sum_{n=1}^N \left[\frac{t_n}{y_n} \frac{\partial y_n}{\partial \mathbf{w}} + \frac{(1 - t_n)}{1 - y_n} \frac{\partial (1 - y_n)}{\partial \mathbf{w}} \right] \\ &= -\sum_{n=1}^N \left[\frac{t_n}{y_n} \frac{\partial y_n}{\partial \mathbf{w}} - \frac{(1 - t_n)}{1 - y_n} \frac{\partial y_n}{\partial \mathbf{w}} \right] \\ &= -\sum_{n=1}^N \left[\frac{t_n}{y_n} - \frac{(1 - t_n)}{1 - y_n} \right] \frac{\partial y_n}{\partial \mathbf{w}} \end{aligned}$$

$$\frac{\partial (1 - y_n)}{\partial \mathbf{w}} = -\frac{\partial y_n}{\partial \mathbf{w}}$$

$y_n = \sigma(\mathbf{w}^T \phi(\mathbf{x}_n))$
Need to apply chain rule

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = -\sum_{n=1}^N \left[\frac{t_n}{y_n} - \frac{(1 - t_n)}{1 - y_n} \right] y_n(1 - y_n) \phi(\mathbf{x}_n)$$
$$\frac{\partial y_n}{\partial \mathbf{w}} = \frac{\partial \sigma(\mathbf{w}^T \phi(\mathbf{x}_n))}{\partial \mathbf{w}} = \sigma(\mathbf{w}^T \phi(\mathbf{x}_n)) (1 - \sigma(\mathbf{w}^T \phi(\mathbf{x}_n))) \phi(\mathbf{x}_n) = y_n(1 - y_n) \phi(\mathbf{x}_n)$$



1643396

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = -\sum_{n=1}^N \left[\frac{t_n}{y_n} - \frac{(1 - t_n)}{1 - y_n} \right] y_n(1 - y_n) \phi(\mathbf{x}_n) \quad \text{Separating the terms}$$

$$= -\sum_{n=1}^N [t_n(1 - y_n) \phi(\mathbf{x}_n) - (1 - t_n)y_n \phi(\mathbf{x}_n)]$$

$$= -\sum_{n=1}^N [t_n \phi(\mathbf{x}_n) - \cancel{t_n y_n \phi(\mathbf{x}_n)} - y_n \phi(\mathbf{x}_n) + \cancel{t_n y_n \phi(\mathbf{x}_n)}]$$

$$= -\sum_{n=1}^N [t_n \phi(\mathbf{x}_n) - y_n \phi(\mathbf{x}_n)] = \sum_{n=1}^N [y_n - t_n] \phi(\mathbf{x}_n)$$



1643396

We Cannot Solve Directly

$$\begin{aligned} E(\mathbf{w}) &= -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)] \\ \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} &= \sum_{n=1}^N [y_n - t_n] \phi(\mathbf{x}_n) \\ &= \sum_{n=1}^N [\sigma(\mathbf{w}^T \phi(\mathbf{x}_n)) - t_n] \phi(\mathbf{x}_n) \end{aligned}$$

Setting $\nabla E = 0$ gives

$$\sum_{n=1}^N \sigma(\mathbf{w}^T \phi(\mathbf{x}_n)) \phi(\mathbf{x}_n) = \sum_{n=1}^N t_n \phi(\mathbf{x}_n)$$

Nonlinear in $\mathbf{w}$: **no closed form solution.**

We minimize $E(\mathbf{w})$ numerically instead.

Regularization with Logistic Regression

- The Logistic Regression Model
- Logistic Regression Objective
- Logistic Regression Optimization
- Regularization with Logistic Regression
- Model Evaluation
- Multi-Class Classification



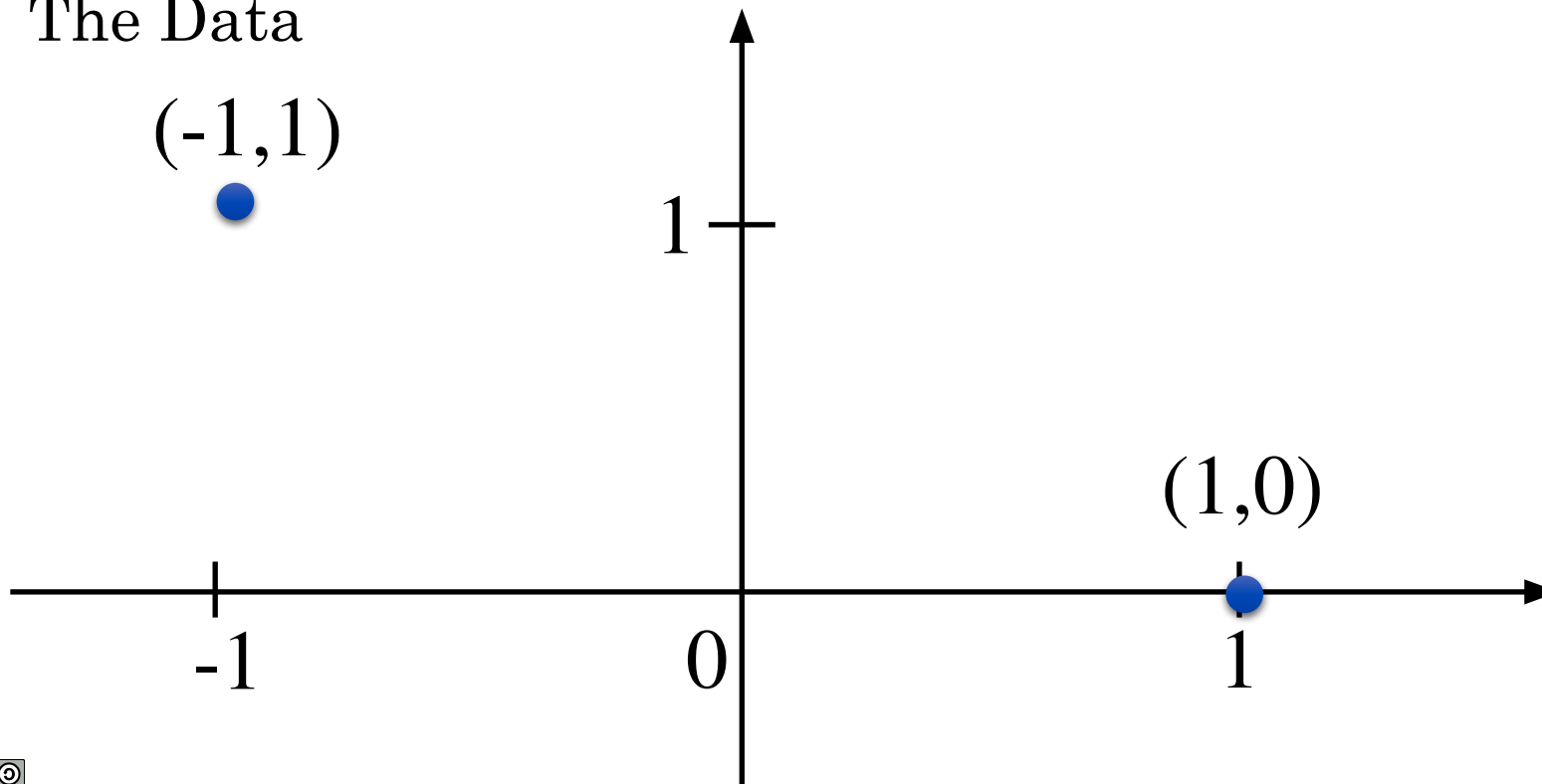
1643396

What Is the Value of w ?

$$w = \underset{w}{\operatorname{argmin}} - \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$

Assume $\phi(\mathbf{x}) = \mathbf{x}$

The Data

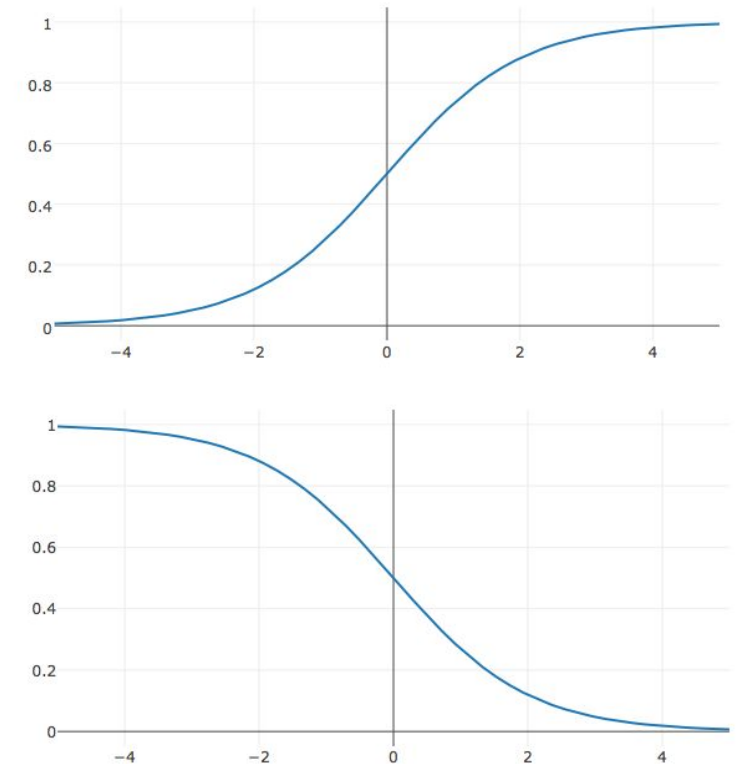


A. $w = -1$

B. $w = 1$

C. $w = -\infty$

D. $w = +\infty$



1643396



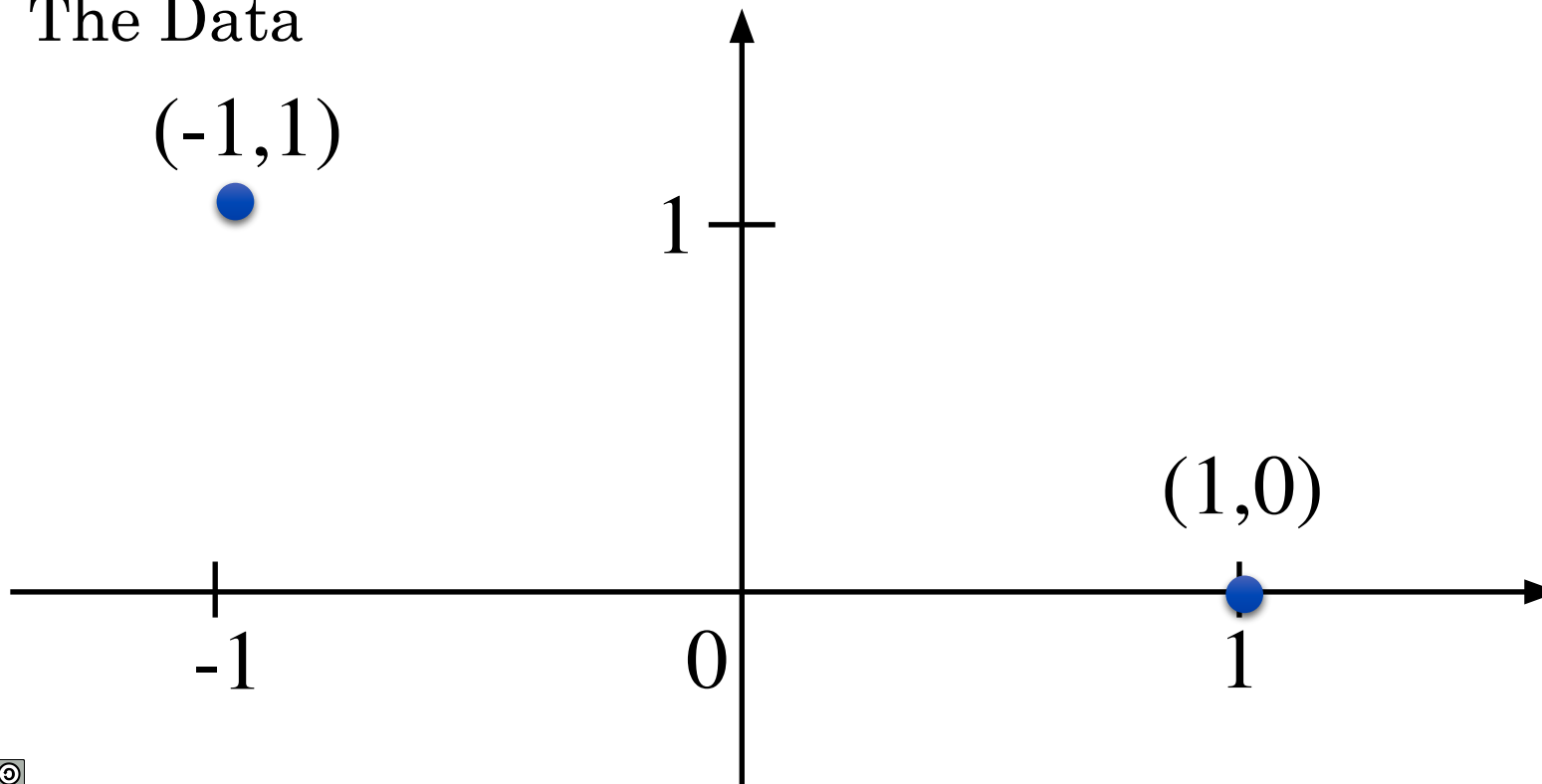
What is the best value for w ?

What Is the Value of w ?

$$w = \underset{w}{\operatorname{argmin}} - \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$

Assume $\phi(\mathbf{x}) = \mathbf{x}$

The Data

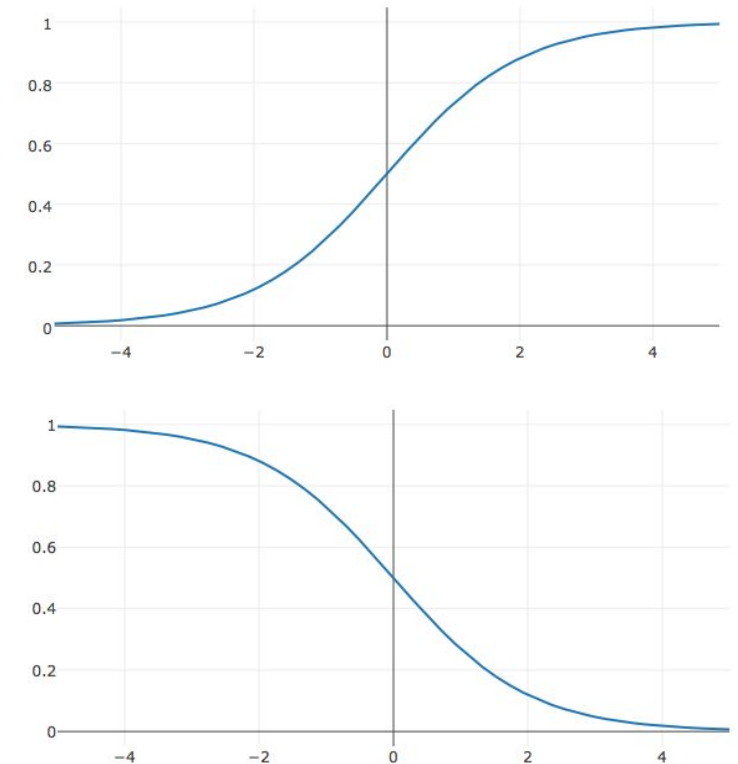


A. $w = -1$

B. $w = 1$

C. $w = -\infty$

D. $w = +\infty$



1643396

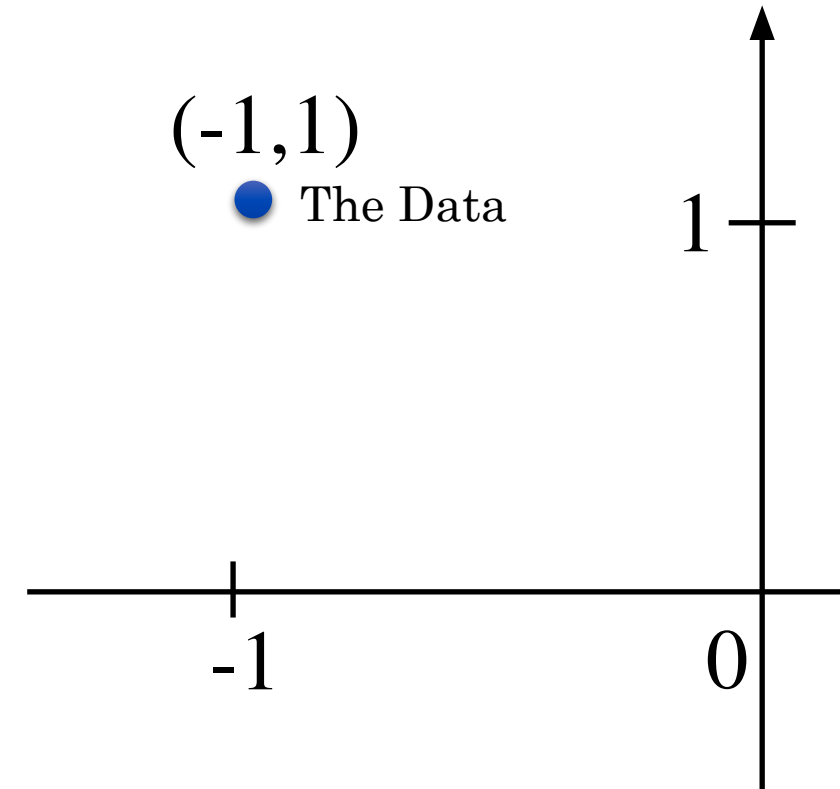
What Is the Value of w ?

$$w = \underset{w}{\operatorname{argmin}} - \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$

For the point $(-1, 1)$:

Objective:

$$-\ln \sigma(w^T \mathbf{x}) = -\ln \sigma(-w) \Rightarrow w \rightarrow -\infty$$



1643396

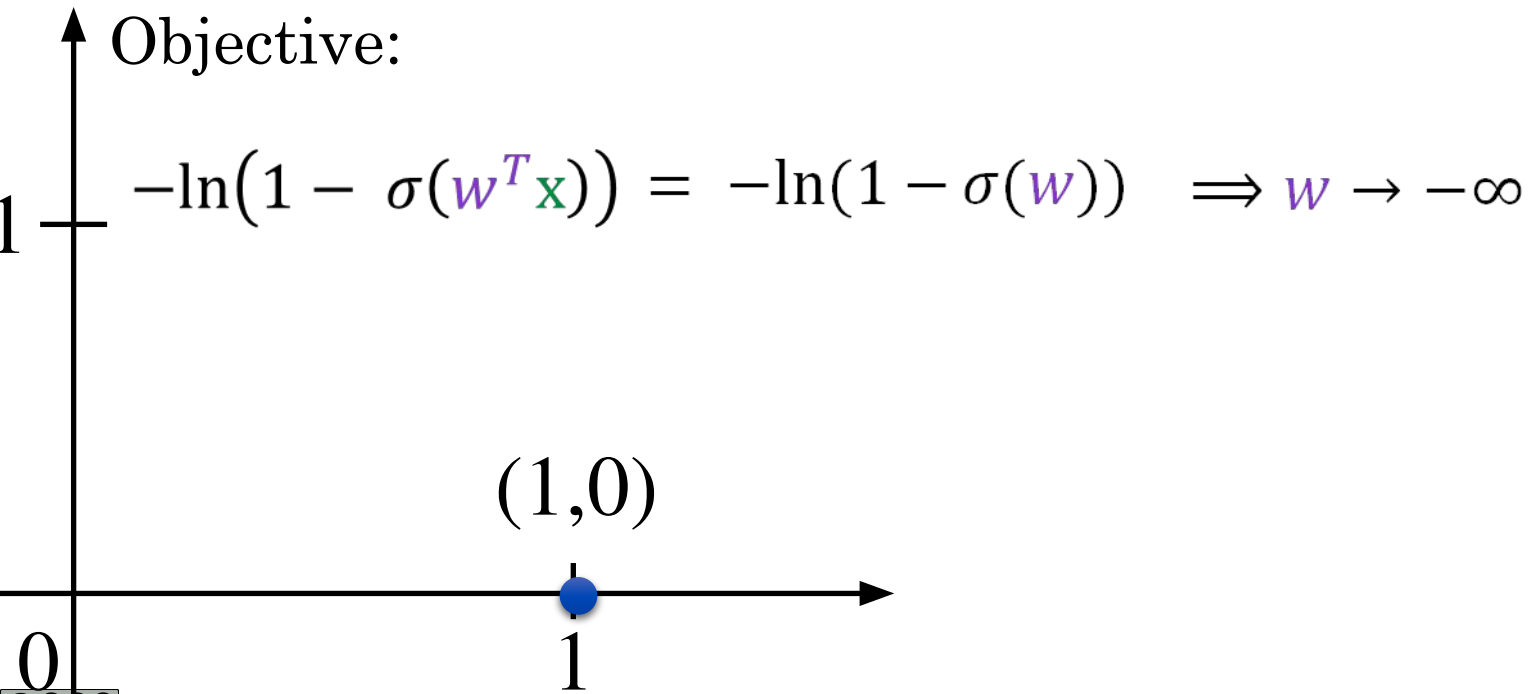


1643396

What Is the Value of w ?

$$w = \underset{w}{\operatorname{argmin}} - \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$

For the point (1,0):



What Is the Value of w ?



1643396

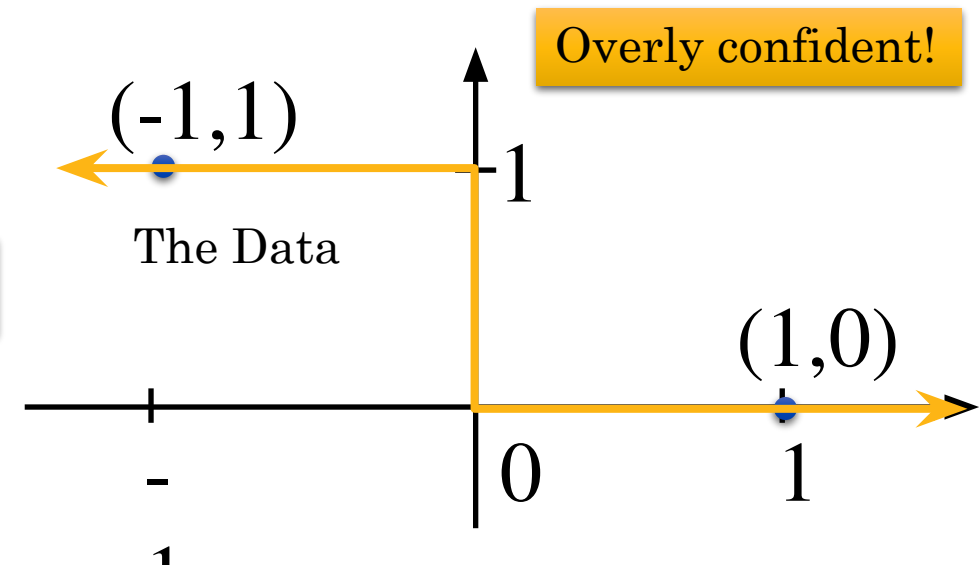
For the point $(-1,1)$:

$$-\ln \sigma(w^T \mathbf{x}) \Rightarrow w \rightarrow -\infty$$

For the point $(1,0)$:

$$-\ln(1 - \sigma(w^T \mathbf{x})) \Rightarrow w \rightarrow -\infty$$

Degenerate Solution!



Adding Regularization to Logistic Regression

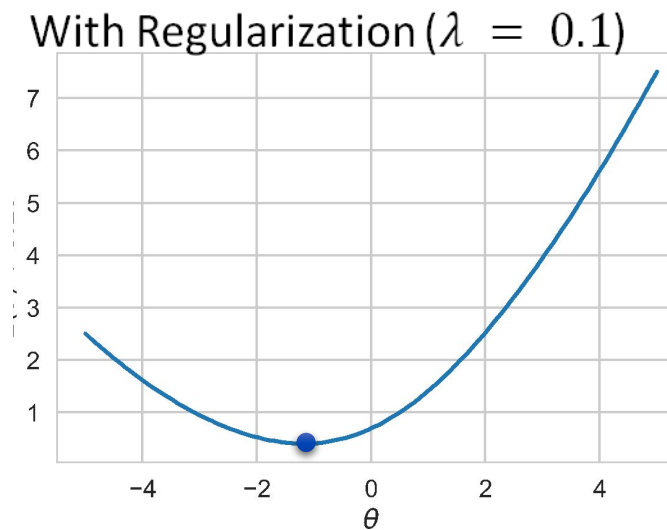
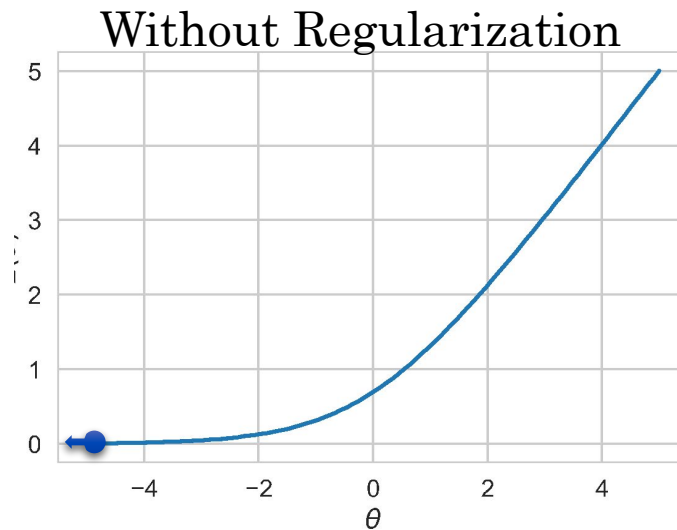


1643396

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} - \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)] + \frac{\lambda}{2} \sum_{d=1}^D w^2$$

Prevents weights from diverging on linearly separable data.

Earlier
Example

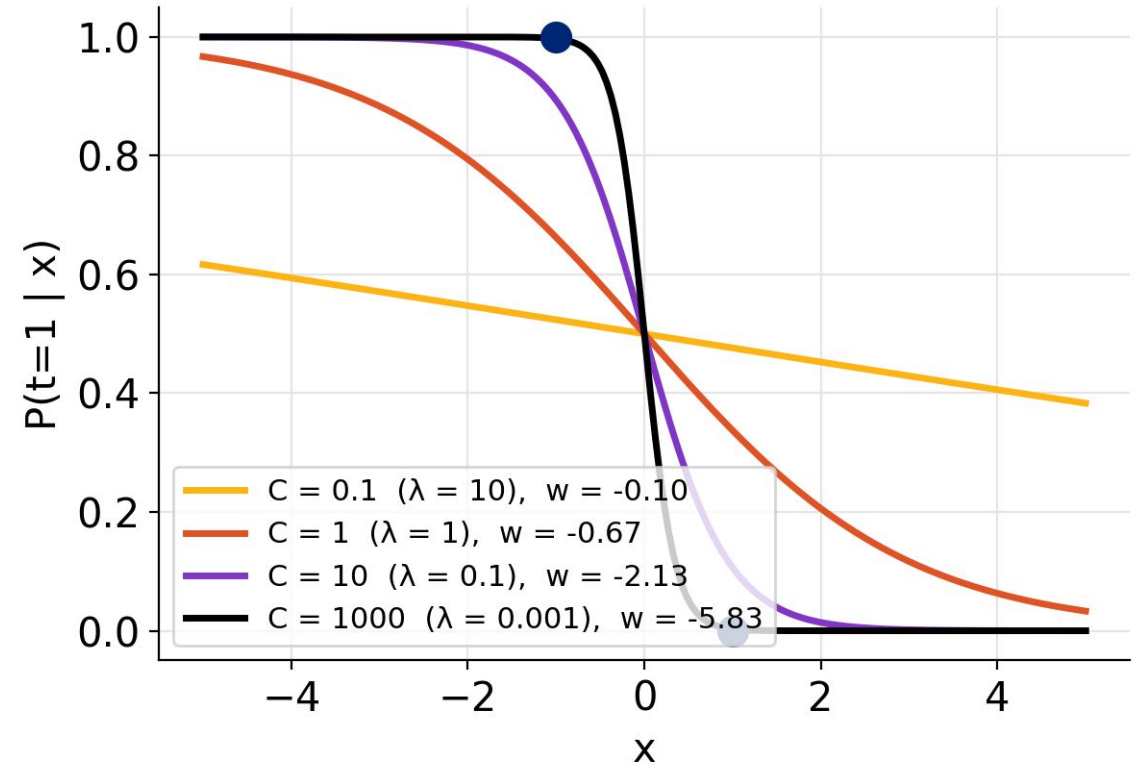


In scikit-learn: $C = \frac{1}{\lambda}$

- scikit-learn minimizes

$$C \times \Sigma_n CE_n + \frac{1}{2} \|w\|^2$$

- **Small C = strong regularization.**
- Default is $C = 1$:
`LogisticRegression()` is always regularized. Use `C=np.inf` for none.
- Check on the toy data (no intercept):
 - $\lambda = 0.1 \rightarrow w = -2.128$ ($C = 10$)
 - $\lambda = 1 \rightarrow w = -0.675$ ($C = 1$)



1643396

Model Evaluation

- The Logistic Regression Model
- Logistic Regression Objective
- Regularization with Logistic Regression
- Model Evaluation
- Multi-Class Classification



1643396

Classifier Performance



1643396

“**Positive**” means a **prediction** of **1**.
“**Negative**” means a **prediction** of **0**.

“**True**” means **correct** prediction.
“**False**” means **incorrect** prediction.

		Prediction	
		0	1
Actual	0	True Negative (TN)	False Positive (FP)
	1	False Negative (FN)	True Positive (TP)

Classifier Performance



1643396

A **confusion matrix** plots TP, TN, FP, FN for a particular classifier, dataset, and **threshold (!)**.

Our **0/1 predictions** depend on our choice of probability threshold, so the confusion matrix can also **change with a new threshold!**

		Predicted	
		0	1
Actual	0	TN 65	FP 6
	1	FN 9	TP 34

Classifier Accuracy



1643396

$$\text{accuracy} = \frac{\# \text{ of points classified correctly}}{\# \text{ Total points}}$$

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{TP + TN}{N}$$

The most basic evaluation metric for a classifier is **accuracy**.



1643396

Pitfalls of Accuracy: A Case Study

Suppose we're trying to build a classifier to filter spam emails.

Each email is **spam** (1) or **ham** (0).

We have 100 emails: **5** of them are **spam**, and the remaining **95** are real (i.e., **ham**).

Classifier 1:

Classify every email as **ham** (0).

What is the accuracy of Classifier 1?



1643396

Pitfalls of Accuracy: A Case Study

Suppose we're trying to build a classifier to filter spam emails.

Each email is **spam** (1) or **ham** (0).

We have 100 emails: **5** of them are **spam**, and the remaining **95** are real (i.e., **ham**).

Classifier 1:

Classify every email as **ham** (0).

$$accuracy_1 = \frac{95}{100} = 0.95$$

High accuracy...

...but we detected **none** ⚠ of the spam!!!

Classifier 2:

Classify every email as **spam** (1).

$$accuracy_2 = \frac{5}{100} = 0.05$$

Low ⚠ accuracy...

...but we detected **all** of the spam!!!

While widely used, the accuracy metric is **is problematic** if there is **class imbalance**.



Suppose I increase the probability threshold for converting my estimated probabilities from logistic regression into 0/1 predictions.

I do not change my fitted model or test data.

What can happen to the number of true positives (TP) in the test data? Select all that apply.

Effect of changing the probability threshold on TP



1643396

I **increase** the probability threshold. What happens to the count of **true positives (TP)**?

Estimated probabilities	Threshold=0.6			
	Threshold=0.3		Threshold=0.8	
	0.21	0.63	0.85	
Prediction (T=0.3)	0 (TN)	1 (TP)	1 (TP)	TP = 2
Prediction (T=0.6)	0 (TN)	1 (TP)	1 (TP)	No change!
Prediction (T=0.8)	0 (TN)	0 (FN)	1 (TP)	TP = 1

If we **increase** the threshold, it becomes **harder** to make a positive prediction. The number of **positives (P)** will **stay the same or decrease**. So, TP and FP will either go down or stay the same, too.

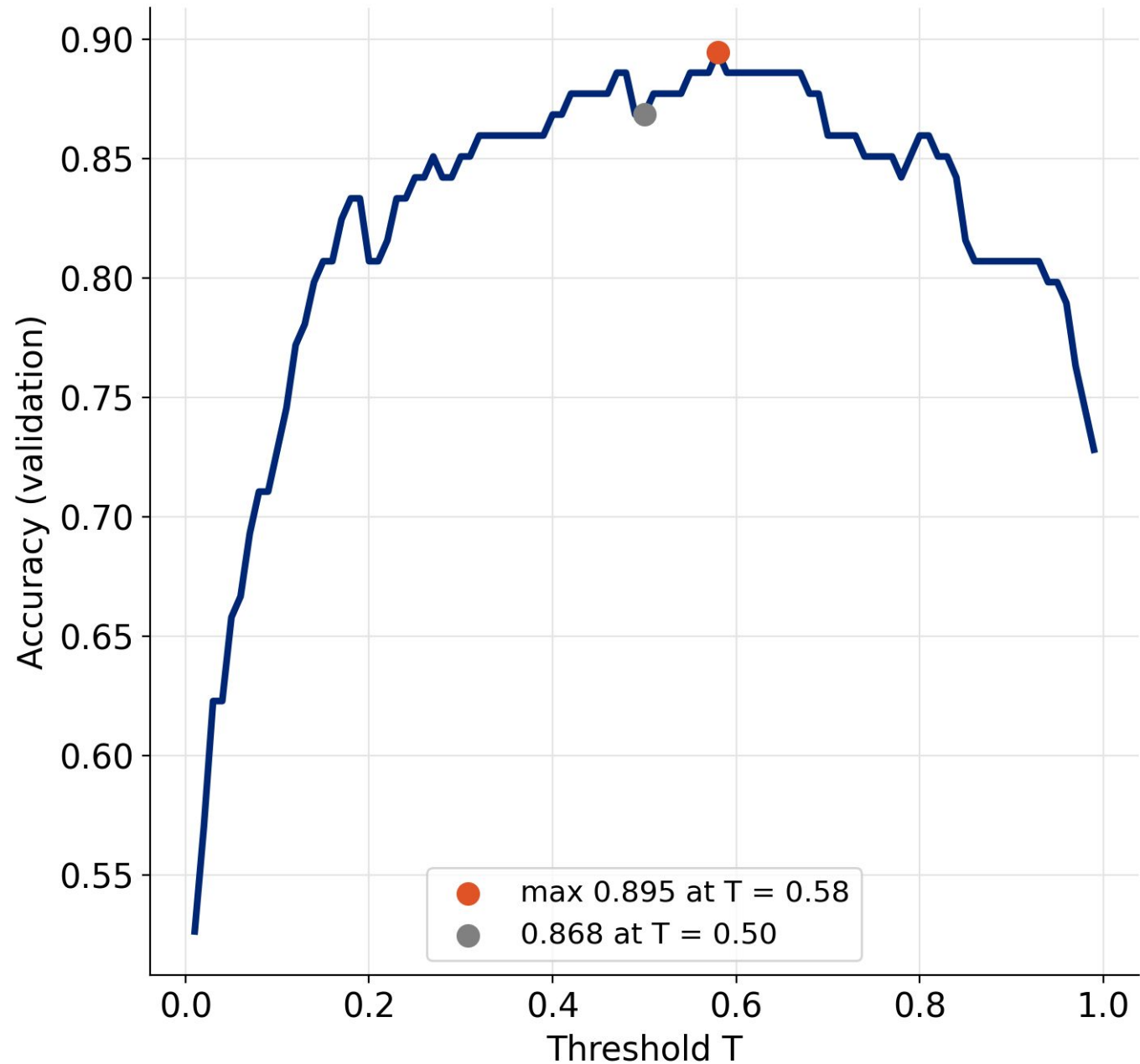
Choosing an Accuracy Threshold

The choice of threshold T impacts classification performance.

- High T : Most predictions are 0. Lots of FNs.
- Low T : Most predictions are 1. Lots of FPs.

Do we get max accuracy when $T = 0.5$? Not always the case...

Here (validation): 0.868 at $T = 0.5$, max 0.895 at $T = 0.58$.



1643396

Accuracy, Precision, and Recall



1643396

		Prediction	
		0	1
Actual	0	TN	FP
	1	FN	TP

		Prediction	
		0	1
Actual	0	TN	FP
	1	FN	TP

		Prediction	
		0	1
Actual	0	TN	FP
	1	FN	TP

$$accuracy = \frac{TP + TN}{N}$$

What proportion of **all points** were correctly classified?

"What % of emails did I classify correctly?"

$$precision = \frac{TP}{(TP + FP)}$$

Of all **predicted positives**, what proportion were correct **(TP)**?

"What % of spam predictions were correct?"

$$recall = \frac{TP}{Actual\ 1s} = \frac{TP}{TP + FN}$$

Of all **actual 1s**, what proportion did our model **detect**?

"What % of spam emails were detected?"

These metrics are **threshold-dependent**.

Recall is also known as sensitivity.

Back to the Spam

	0	1
0	TN: 0	FP: 95
1	FN: 0	TP: 5

$$\begin{aligned} \text{accuracy} &= \frac{TP + TN}{N} \\ \text{precision} &= \frac{TP}{TP + FP} \\ \text{recall} &= \frac{TP}{TP + FN} \end{aligned}$$



1643396

Suppose we're trying to build a classifier to filter spam emails.

Each email is **spam** (1) or **ham** (0).

Let's say we have 100 emails, of which only **5** are truly **spam**, and the remaining **95** are **ham**.

Classifier 1:

Classify every email as **ham** (0).

$$\text{accuracy}_1 = \frac{95}{100} = 0.95$$

$$\text{precision}_1 = \frac{0}{0 + 0} = \text{undefined}$$

$$\text{recall}_1 = \frac{0}{0 + 5} = 0$$

Classifier 2:

Classify every email as **spam** (1).

$$\text{accuracy}_2 = \frac{5}{100} = 0.05$$

$$\text{precision}_2 = \frac{5}{5 + 95} = 0.05 \quad \left. \vphantom{\text{precision}_2} \right\} \text{Many false positives!}$$

$$\text{recall}_2 = \frac{5}{5 + 0} = 1 \quad \left. \vphantom{\text{recall}_2} \right\} \text{No false negatives!}$$



1643396

Precision vs. Recall

$$\text{precision} = \frac{TP}{TP + FP}$$

Precision penalizes FPs.

$$\text{recall} = \frac{TP}{TP + FN}$$

Recall penalizes FNs.

There is a **tradeoff** between precision and recall; they are often **inversely related**.

FPs and FNs can have **different costs** based on the Loss matrix.

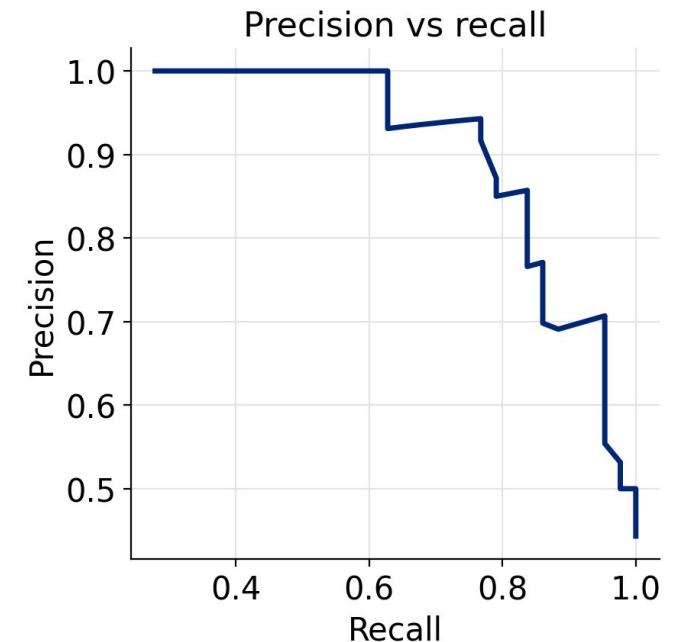
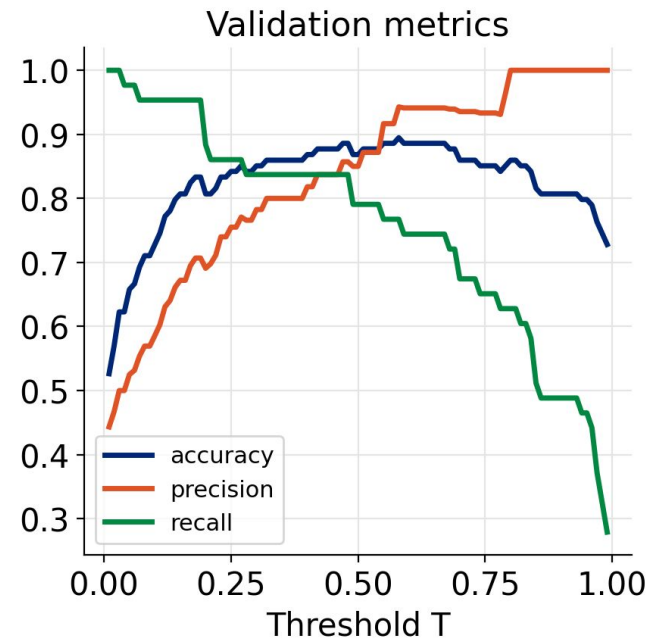
- You should **adjust** your threshold to optimize for **the expected Loss**.

Precision-Recall Curves



1643396

- To construct a precision-recall curve, we:
- 1) Generate predictions for many different thresholds.
 - 2) For each threshold, compute precision and recall.



We often choose a threshold that keeps both precision and recall high. But, if the cost of FPs and FNs differs a lot, we may not.



1643396

Decisions = Posteriors + Loss

- We separate *inference* from *decision*.

Inference gives posteriors $p(C_k|\mathbf{x})$.

The task gives a **loss matrix** L_{kj} .

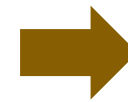
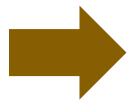
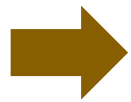
L_{kj} : Loss if we predict j
when the true class is k .

The right action is the one with **minimum expected loss at this $\mathbf{x}$** .

Expected loss at $\mathbf{x}$ if
we choose class j

$$= \sum_k L_{kj} p(C_k|\mathbf{x})$$

Input
 $\mathbf{x}$



we separate *inference* from *decision*.
Inference gives posteriors $p(C_k|\mathbf{x})$.
The task gives a **loss matrix** L_{kj} .
The right action is the one with **minimum expected loss at this $\mathbf{x}$** .



1643396

Decisions = Posteriors + Loss

The right action is the one with **minimum expected loss at this $\mathbf{x}$** .

$$\text{Expected loss at } \mathbf{x} \text{ if we choose class } j = \sum_k L_{kj} p(C_k | \mathbf{x})$$



Change the **costs** $L \rightarrow$ decisions change immediately;
No retraining. That's why we prefer probabilities over hard labels.



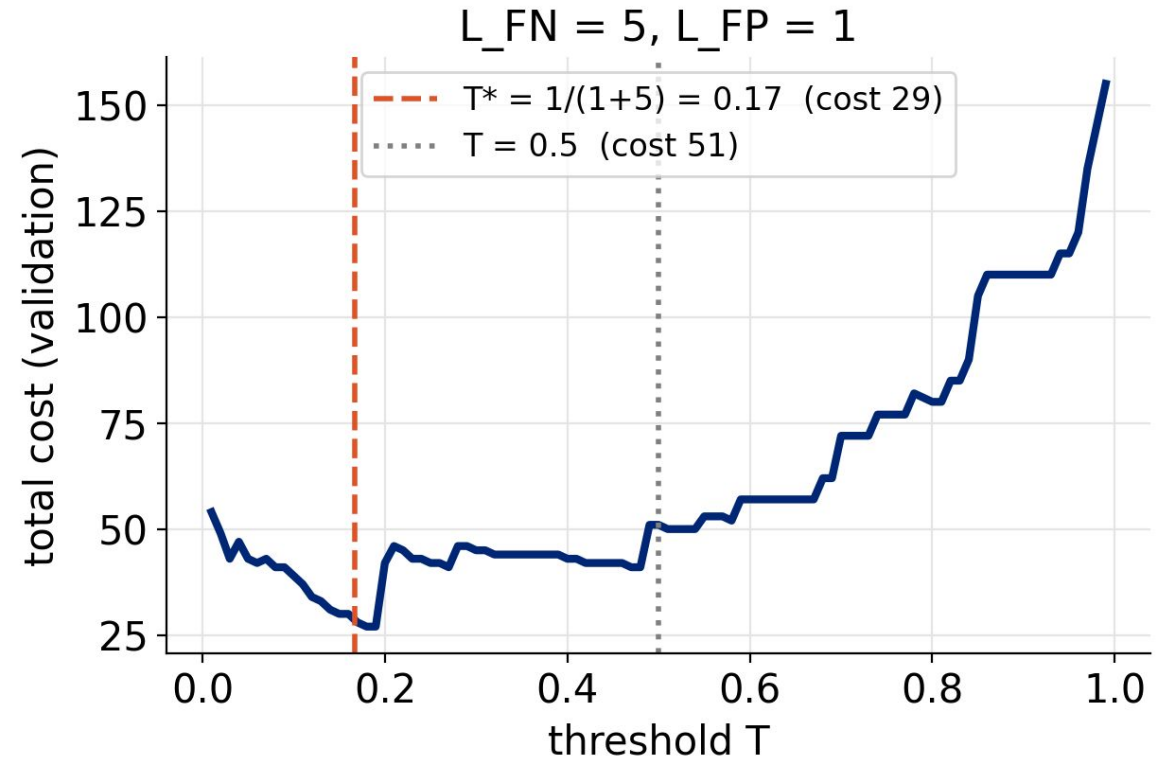
1643396

From a Loss Matrix to a Threshold

- Let $p = p(C_1|\mathbf{x})$. Expected loss of each action:
 - predict 0: $L_{FN} \times p$
 - predict 1: $L_{FP} \times (1 - p)$
- Predict 1 when $L_{FP} \times (1 - p) < L_{FN} \times p$, i.e.

$$p > T^* = \frac{L_{FP}}{(L_{FP} + L_{FN})}$$

- In our previous case, assume $L_{FN} = 5$, $L_{FP} = 1$, then $T^* = 0.17$.
- **No retraining. Only the threshold moves.**



Computing an F1 Score



1643396

One way to **balance** precision and recall is to maximize the F_1 **Score**:

$$\text{F1 Score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- The harmonic mean of precision and recall
- Often used when there is a **large class imbalance**

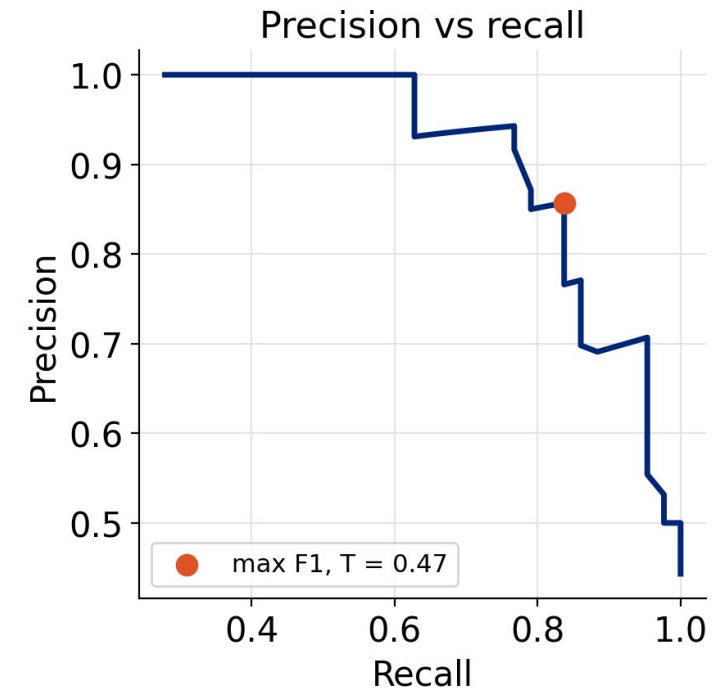
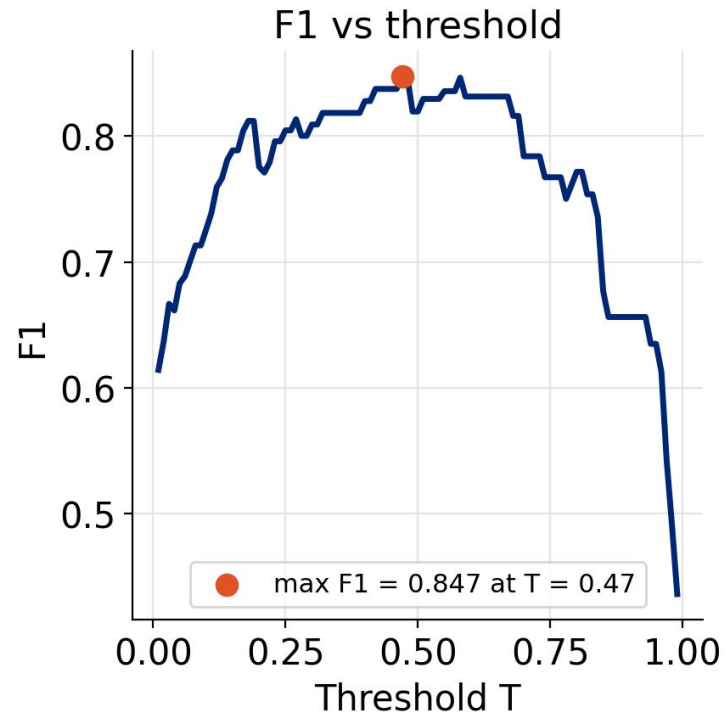
How do you use it?

- Pick the threshold that **maximizes the F_1 score**

Note: If FPs and FNs have different costs, we may not want to balance precision + recall.

Maximizing F1 Score

We can maximize the F1 score by evaluating the F1 score for many threshold values.



1643396



1643396

True and False Positive Rates

Two more performance metrics worth knowing!

$$FPR = \frac{FP}{TN + FP}$$

False Positive Rate (FPR): Out of all actual 0s, how many did we classify incorrectly?

$$TPR = \frac{TP}{TP + FN}$$

True Positive Rate (TPR): Out of all actual 1s, how many did we classify correctly? **Same as recall.**

	Prediction	
	0	1
Actual	0	FP
	1	TP

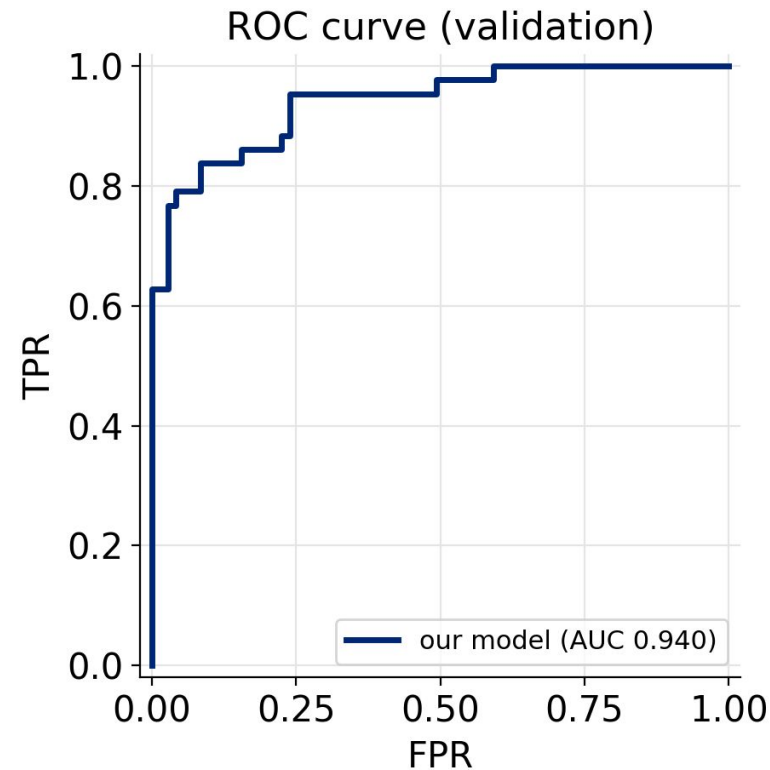
Lots of classification metrics out there! All based on TP, TN, FP, FN.

ROC Curves

We can perform a similar process with FPR and TPR.

- 1) Try many thresholds
- 2) Compute the FPR and TPR for each threshold
- 3) Choose a threshold that keeps FPR *low* and TPR *high*

ROC = “receiver operating characteristic” (comes from radar in WWII)



Threshold is high:
FPR low, TPR low
(few positive predictions)

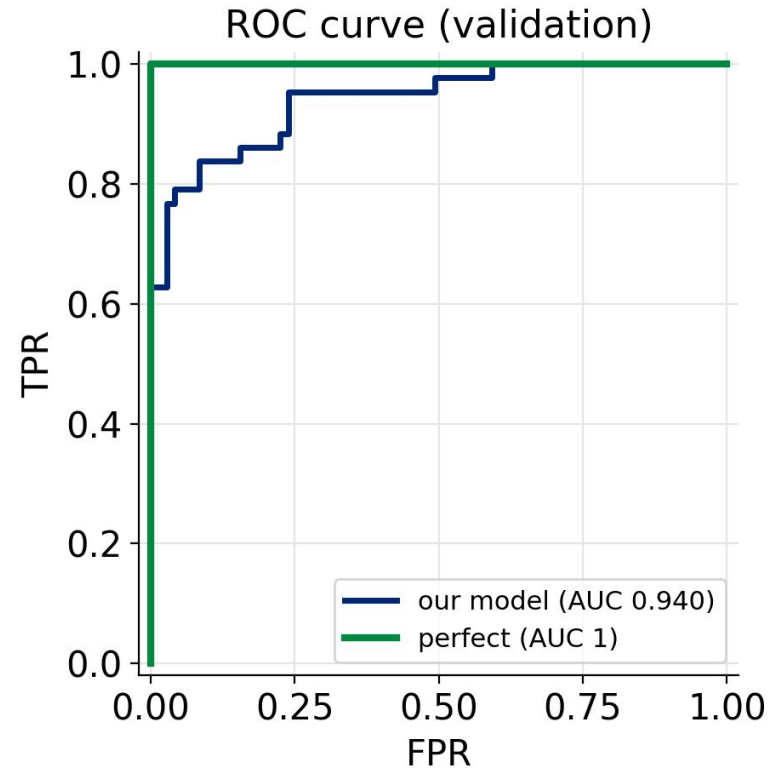
Threshold is low:
FPR high, TPR high
(many positive predictions)



1643396

ROC Curves

A **perfect predictor** has $\text{TPR} = 1$ and $\text{FPR} = 0$



The best **Area Under the ROC Curve (AUC-ROC)** is 1.

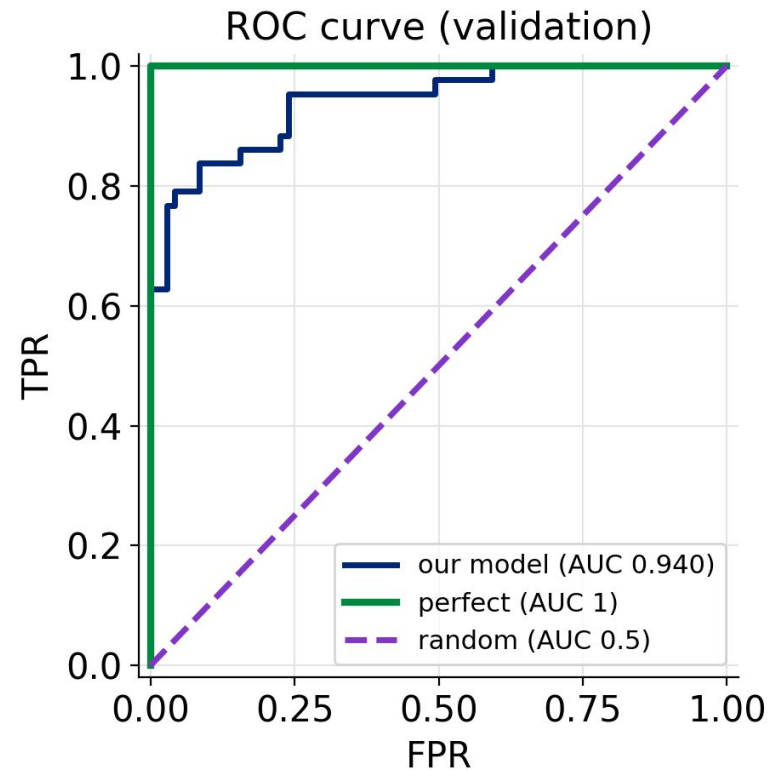
Because we want our classifier to be as close as possible to the perfect predictor, we aim to maximize the AUC.



1643396

ROC Curves

A predictor that **predicts randomly** has an AUC-ROC of 0.5



Useful classifiers have AUC between 0.5 and 1. Ours: 0.940 on validation. Below 0.5 means worse than random.



1643396

Multi-Class Classification

- The Logistic Regression Model
- Logistic Regression Objective
- Regularization with Logistic Regression
- Model Evaluation
- Multi-Class Classification



1643396



1643396

Multiple Classes

One-versus-the-rest classifier

- A model with $K - 1$ classifiers, each of which solves a two-class problem of separating points in a particular class C_k from points not in that class.

One-versus-one classifier

- Introduce $\frac{K(K-1)}{2}$ binary discriminant functions, one for every possible pair of classes.

Better Option: Multi-class Discriminant Solution



1643396

- Single K-class discriminant with K linear functions:

$$y_k(x) = w_k^T x + w_{k0}$$

- Assign x to C_k if $y_k(x) > y_j(x)$ for all $j \neq k$.
- Boundary between C_k and C_j :

$$(w_k - w_j)^T x + (w_{k0} - w_{j0}) = 0$$

- The decision **regions** are always singly connected and convex.



1643396

$K > 2$ Classes: Softmax Function

- Let the model output **scores** (logits) $a_1, \dots, a_K \in R$.
- We want to turn these scores into $p(C_k|\mathbf{x})$.
- We generalize binary logit:

For every two class i, j : $\ln \frac{p_i}{p_j} = a_i - a_j$

- Now we solve to find p'_k s, knowing that $\sum_{k=1}^K p_k = 1$.

$$1 = \sum_{k=1}^K p_k = p_j \sum_{k=1}^K e^{a_k - a_j} \rightarrow p_j = \frac{1}{\sum_{k=1}^K e^{a_k - a_j}} = \frac{e^{a_j}}{\sum_{k=1}^K e^{a_k}}$$

Softmax Function

$$p_j = \frac{e^{a_j}}{\sum_{k=1}^K e^{a_k}}$$

Writing the equation
based on one class j

How? $\ln \frac{p_i}{p_j} = a_i - a_j \rightarrow p_i = e^{a_i - a_j} p_j$



Which of the following are true about the softmax function

Lecture 09

Logistic Regression (2)

Credit: Joseph E. Gonzalez and Narges Norouzi

Reference Book Chapters: Chapter 5 Sections 5.1.2, 5.1.3, 5.2