COMP 3105 Introduction to Machine Learning
Lecture 2: Logistic Regression

Junfeng Wen

Carleton University

Fall 2025

Binary Classification Examples

Fruit classification

- Shape, weight $\mapsto$ Apple / Orange

Click through rate

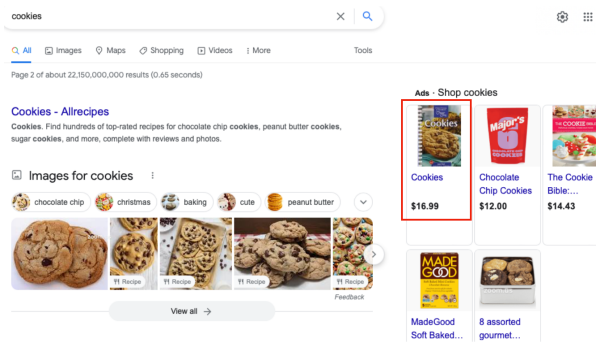
- Query, user info, ad info $\mapsto$ Click / Not click

Email spam filter

- Title, content, links $\mapsto$ Spam / Not spam

Sentiment analysis

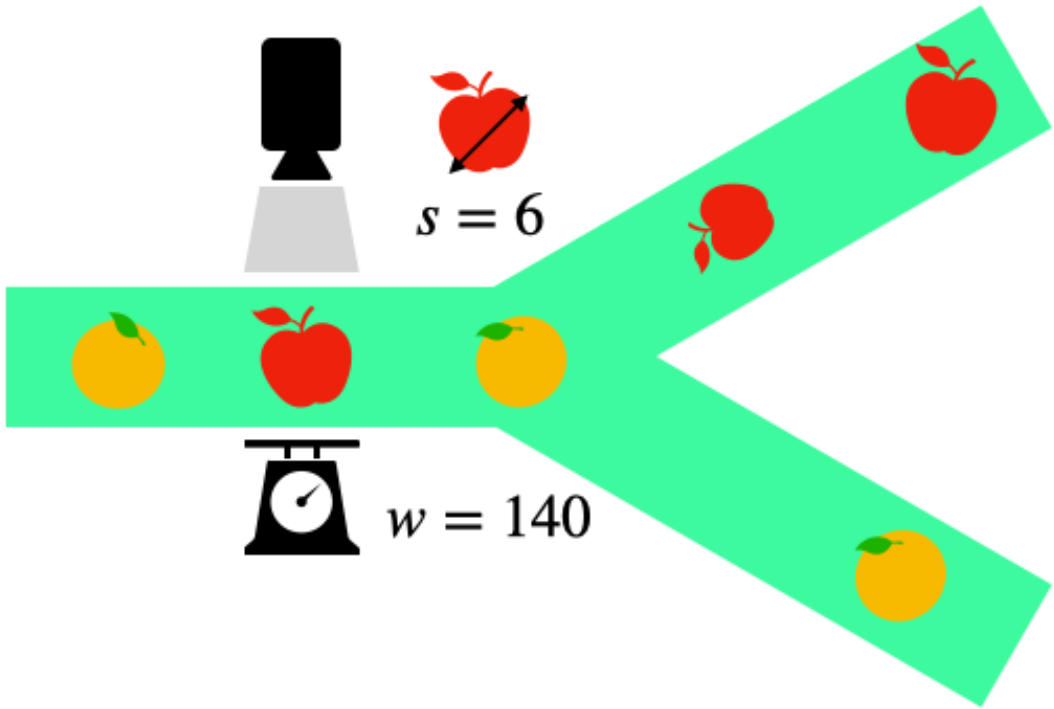
- Text review $\mapsto$ 👍 / 👎



Didn't perform
This product did not work as advertised. I'm disappointed with the purchase

Best product ever!
This is the best product I've ever used. I highly recommend any one to purchase one!

An Example: Apple v.s. Orange



Measured by [size, weight]

- E.g., [6 cm, 140 grams]
- Captured by camera and scale on the conveyor belt

Goal: Decide apple or orange (so we know which way to send)

An Example: Apple v.s. Orange (Cont.)

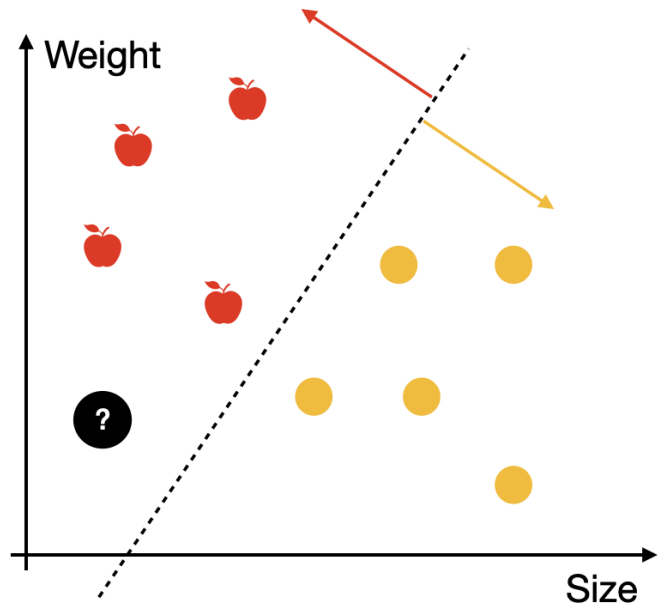
One possible formulation

- Fruit as a *vector* $\mathbf{x} = \begin{bmatrix} 1 \\ 6 \\ 140 \end{bmatrix}$ (augmented for bias/intercept)
- Plot what we know
- Find a "line" (decision boundary):

$$w_0 + w_1 x_1 + w_2 x_2 = 0 \Leftrightarrow \mathbf{x}^\top \mathbf{w} = 0$$

- To predict:

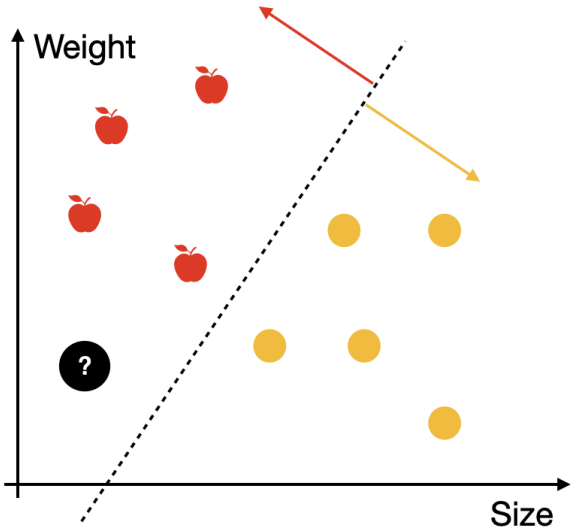
$$h(\mathbf{x}) = \begin{cases} \textit{Apple} & \text{if} \quad \mathbf{x}^\top \mathbf{w} \geq 0 \\ \textit{Orange} & \text{if} \quad \mathbf{x}^\top \mathbf{w} < 0 \end{cases}$$

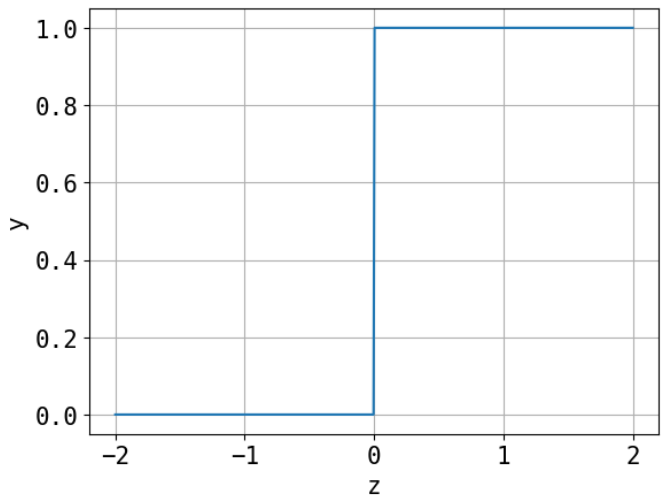


Binary Classification Formulation

Suppose

- The labels: 1 (for Apple) or 0 (for Orange)
- Training dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, e.g., $\mathbf{x} = \begin{bmatrix} 1 \\ 6 \\ 140 \end{bmatrix}$ and $y = 1$
- $\widehat{z} = \mathbf{x}^\top \mathbf{w}$ linear prediction (boundary $\mathbf{x}^\top \mathbf{w} = 0$)
- Actual prediction $\widehat{y} = f(\widehat{z}) = \begin{cases} 1 & \text{if} \quad \widehat{z} \geq 0 \\ 0 & \text{if} \quad \widehat{z} < 0 \end{cases}$

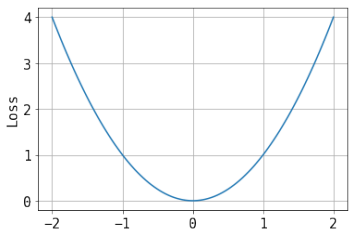




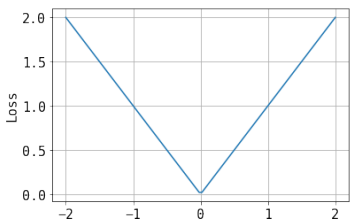
Loss Function?

$$\text{0/1 loss} \quad \mathbb{I}(\widehat{y} \neq y) = \begin{cases} 0 & \text{if} \quad \widehat{y} = y \\ 1 & \text{if} \quad \widehat{y} \neq y \end{cases}$$

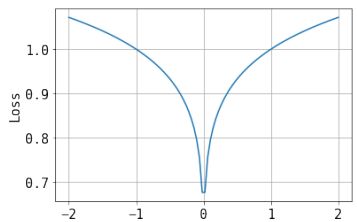
- $\mathbb{I}$ is the *indicator function*
- A.k.a. L_0 loss (L_p when $p \to 0$): $L_0(\widehat{y}, y) \doteq \mathbb{I}(\widehat{y} \neq y)$



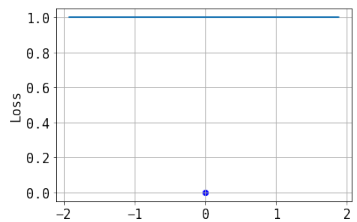
$p = 2$



$p = 1$



$p = 0.1$



$p = 0$

Objective function: *Misclassification rate*

$$J(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} L_0(\widehat{y}_i, y_i) \quad \in \quad [0, 1]$$

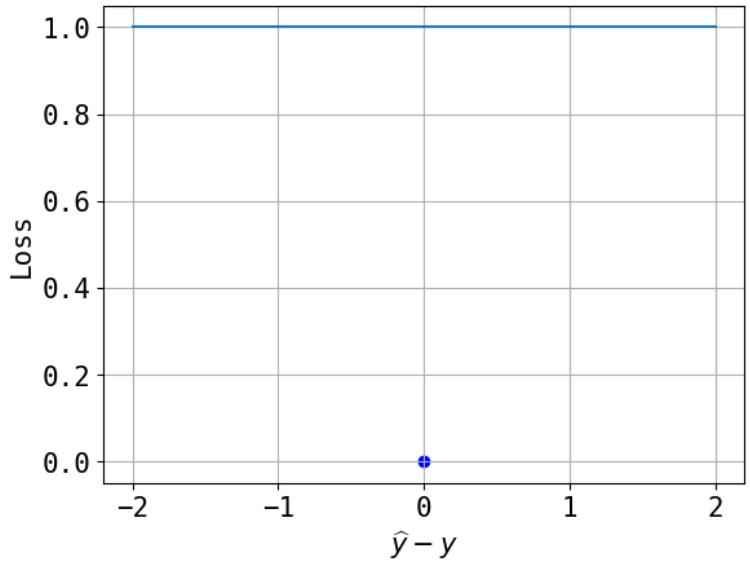
Then *accuracy* $= 1 -$ misclassification rate $\in [0, 1]$

L_0 Problem?

Recall $p < 1$ is not convex nor smooth

$$\nabla J(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \nabla L_0(\widehat{y}_i, y_i)$$

$$\nabla L_0 = \frac{\partial L_0}{\partial \mathbf{w}} = \frac{\partial L_0}{\partial \widehat{y}} \cdot \frac{\partial \widehat{y}}{\partial \widehat{z}} \cdot \frac{\partial \widehat{z}}{\partial \mathbf{w}}$$



- $\frac{\partial L_0}{\partial \widehat{y}}$ zero gradient everywhere, *except*
- When $\widehat{y} = y$, which is not even defined!
- $\partial \widehat{y} / \partial \widehat{z} = f'(\widehat{z})$: similar issue (step function)
- Even GD doesn't work
- Even approximately minimize $J(\mathbf{w})$ is NP-hard when not linear separable (Höffgen and Simon, 1992)

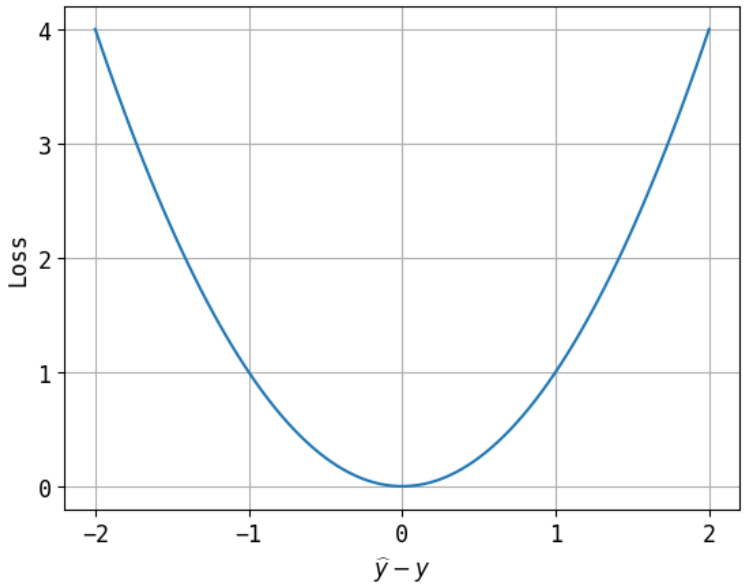
Another Attempt?

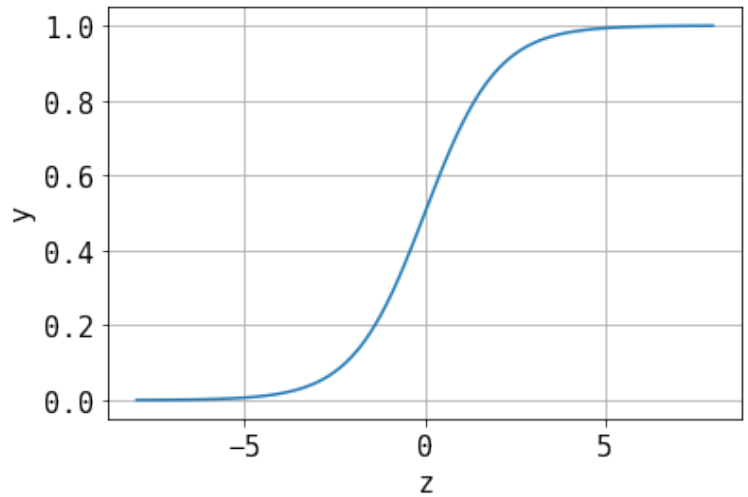
L_0 is hard because of discontinuites in $\frac{\partial L_0}{\partial \widehat{y}}$ and $\frac{\partial \widehat{y}}{\partial \widehat{z}}$

- Let's find some alternatives!
- Replace L_0 with $L_2 : \frac{1}{2}(\widehat{y} - y)^2$
- Choose a different f (Recall that $\widehat{y} = f(\widehat{z})$)

The *sigmoid transformation* $y = f(z) = \sigma(z) \doteq \frac{1}{1+e^{-z}}$

- Range (0, 1), smooth, invertible





Problem solved?

Problem using Sigmoid with L_2 Loss

$$\nabla L_2 = \frac{\partial L_2}{\partial \mathbf{w}} = \frac{\partial L_2}{\partial \widehat{y}} \cdot \frac{\partial \widehat{y}}{\partial \widehat{z}} \cdot \frac{\partial \widehat{z}}{\partial \mathbf{w}}$$
$$= (\widehat{y} - y) \cdot \frac{\partial \widehat{y}}{\partial \widehat{z}} \cdot \mathbf{x}$$

Recall $L_2 = \frac{1}{2}(\widehat{y} - y)^2$

$$\widehat{y} = \sigma(\widehat{z}) = 1/(1 + e^{-\widehat{z}})$$
$$\widehat{z} = \mathbf{x}^\top \mathbf{w}$$

Exercise: The gradient of the sigmoid function $\sigma'(\widehat{z})$

$$\frac{\partial \widehat{y}}{\partial \widehat{z}} = \frac{e^{-z} + 1 - 1}{(1 + e^{-z})^2} = \widehat{y}(1 - \widehat{y})$$
$$\nabla L_2 = (\widehat{y} - y) \cdot \widehat{y}(1 - \widehat{y}) \cdot \mathbf{x}$$

If $\widehat{y}$ is very wrong ($|\widehat{y} - y| \approx 1$), the gradient $\nabla L_2 \approx \mathbf{0}_d$

- GD update $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla L_2 \approx \mathbf{w}^{(t)}$
- Almost no incentive to update $\mathbf{w}$ even though we should correct such misclassification errors

Logistic Regression: Choose the "Right" Loss

Cross entropy loss (a.k.a. log loss)

$$L_{\text{CE}}(\widehat{y}, y) \doteq -y \log(\widehat{y}) - (1 - y) \log(1 - \widehat{y})$$

Its gradient

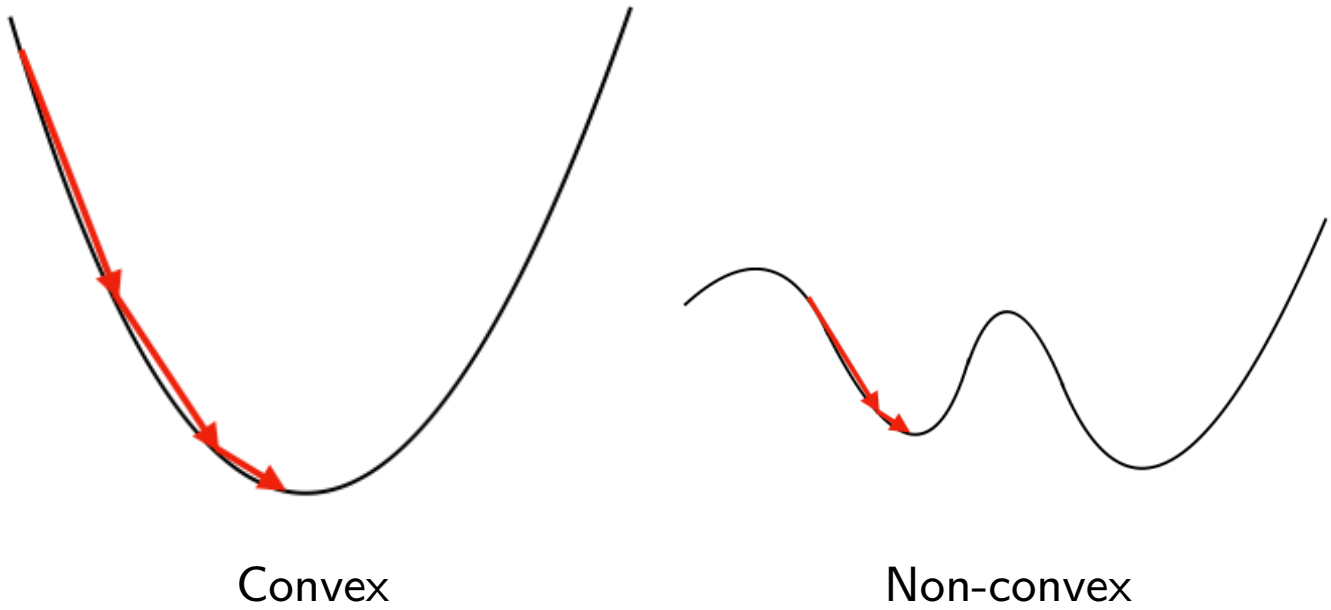
$$\begin{aligned}
\nabla L_{\text{CE}} &= \frac{\partial L_{\text{CE}}}{\partial \widehat{y}} \cdot \frac{\partial \widehat{y}}{\partial \widehat{z}} \cdot \frac{\partial \widehat{z}}{\partial \mathbf{w}} \\
&= \left(-\frac{y}{\widehat{y}} + \frac{1 - y}{1 - \widehat{y}} \right) \cdot \widehat{y}(1 - \widehat{y}) \cdot \mathbf{x} \\
&= [-y(1 - \widehat{y}) + (1 - y)\widehat{y}] \cdot \mathbf{x} \\
&= (\widehat{y} - y) \cdot \mathbf{x}
\end{aligned}$$

- If $\widehat{y}$ is very wrong (e.g., $y = 1, \widehat{y} \approx 0$), the gradient won't be close to zero
- The reciprocal ($1/\widehat{y}$ or $1/(1 - \widehat{y})$) resolves the issue in $\partial \widehat{y} / \partial \widehat{z}$

Convexity

L_{CE} is convex w.r.t. $\mathbf{w}$

- Indirect approach (GD) finding global optimum



Convex Non-convex

Yet direct solution is not available

- Setting $\nabla L_{\text{CE}} = \mathbf{0}_d$ does not give analytic solution

Checking Gradient

Many implementation errors come from incorrect gradient!

- How to check if the gradient is correct? Numerical comparison

Finite difference

$$\frac{\partial}{\partial w_i} J(\mathbf{w}) = \lim_{\delta \to 0} \frac{J(w_1, \ldots, w_i + \delta, \ldots, w_d) - J(w_1, \ldots, w_i, \ldots, w_d)}{\delta}$$

- Pick a small δ, say 10^{-10}
- Pick a random location $\mathbf{w}^{(0)}$ (& random $X, \mathbf{y}$)
- Compare analytic gradient with numeric gradient

```
np.allclose(analytic_grad, numeric_grad)
```

Checking Gradient (Cont.)

Another way to check gradient: Automatic differentiation

- Compute $\nabla J(\mathbf{w}^{(0)})$ at some location $\mathbf{w}^{(0)}$

```
from jax import grad
from autograd import grad  # older

w0 = # where you want to compute the gradient
analytic_grad = # your analytic gradient at w0
J_function = lambda w: # define a function of w
autodiff_grad = grad(J_function(w0))

np.allclose(analytic_grad, autodiff_grad)
```

Relaxed Formulation for Classification

$$\begin{array}{cc} \text{Step function} & L_0 \text{ loss} \\ \downarrow & \downarrow \\ \text{Sigmoid transform} & \text{CE loss} \end{array}$$

- Known as *relaxation*
- No longer solve the (original) objective we care the most
- Small CE loss $\neq$ small misclassification error
- For test point $\mathbf{x}_0$, still need to decide 0 or 1:

$$\widehat{y}_0 = \mathbb{I}(\sigma(\mathbf{x}_0^\top \mathbf{w}) \geq 0.5) \qquad \text{(Is 0.5 the best option?)}$$

- Beyond accuracy: precision, recall & F_1 score...

General ML Design Process

General ML design process

- Define input/output (i.e., $\mathbf{x}, y$) and model
- Define objective: what we want to achieve
- Find the best model
  - Direct approach (analytic solution)
  - Indirect approach (gradient descent, LP)
- If doesn't work, identify the key issues and redesign

Later in the course

- Principled way to find "suitable" loss function

References

Bishop and Nasrabadi (2006, Sec.4.1 & Sec.4.3) Hastie et al. (2009, Sec.4.4)

Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Höffgen, K.-U. and Simon, H. U. (1992). Robust trainability of single neurons. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 428–439.

Appendix: Why Logistic Regression is Convex?

Claim: A loss L is convex iff its Hessian $\mathbf{H}(L) \in \mathbb{R}^{d \times d}$ is *positive semi-definite* (PSD).

- Hessian: gradient of gradient
- PSD means $\mathbf{u}^\top \mathbf{H}(L)\mathbf{u} \geq 0, \forall \mathbf{u} \in \mathbb{R}^d$
- Scalar counterpart example: $f(x) = \frac{1}{2}ax^2$ is convex iff $f''(x) = a \geq 0$

Appendix: Why Logistic Regression is Convex? (Cont.)

Consider one training point for simplicity

$$\begin{aligned}
\nabla L_{\text{CE}} &= (\widehat{y} - y) \cdot \mathbf{x} \qquad \text{(previous slides)} \\
\mathbf{H}(L_{\text{CE}}) &= \frac{\partial \widehat{y}}{\partial \mathbf{w}} \times_? \mathbf{x} \\
&= \left(\frac{\partial \widehat{y}}{\partial \widehat{z}} \cdot \frac{\partial \widehat{z}}{\partial \mathbf{w}} \right) \times_? \mathbf{x} \\
&= \left(\widehat{y}(1-\widehat{y}) \cdot \underset{d \times 1}{\mathbf{x}} \right) \times_? \underset{d \times 1}{\mathbf{x}} \\
&= \widehat{y}(1-\widehat{y}) \cdot (\underset{d \times d}{\mathbf{x}\mathbf{x}^\top}) \\
\text{Then } \forall \mathbf{u} \in \mathbb{R}^d, \mathbf{u}^\top \mathbf{H}(L_{\text{CE}})\mathbf{u} &= \widehat{y}(1-\widehat{y}) \cdot \mathbf{u}^\top \mathbf{x} \cdot \mathbf{x}^\top \mathbf{u} \\
&= \widehat{y}(1-\widehat{y}) \cdot (\mathbf{u}^\top \mathbf{x})^2 \geq 0
\end{aligned}$$

Actual objective has multiple training points

- Sum of convex functions is still convex