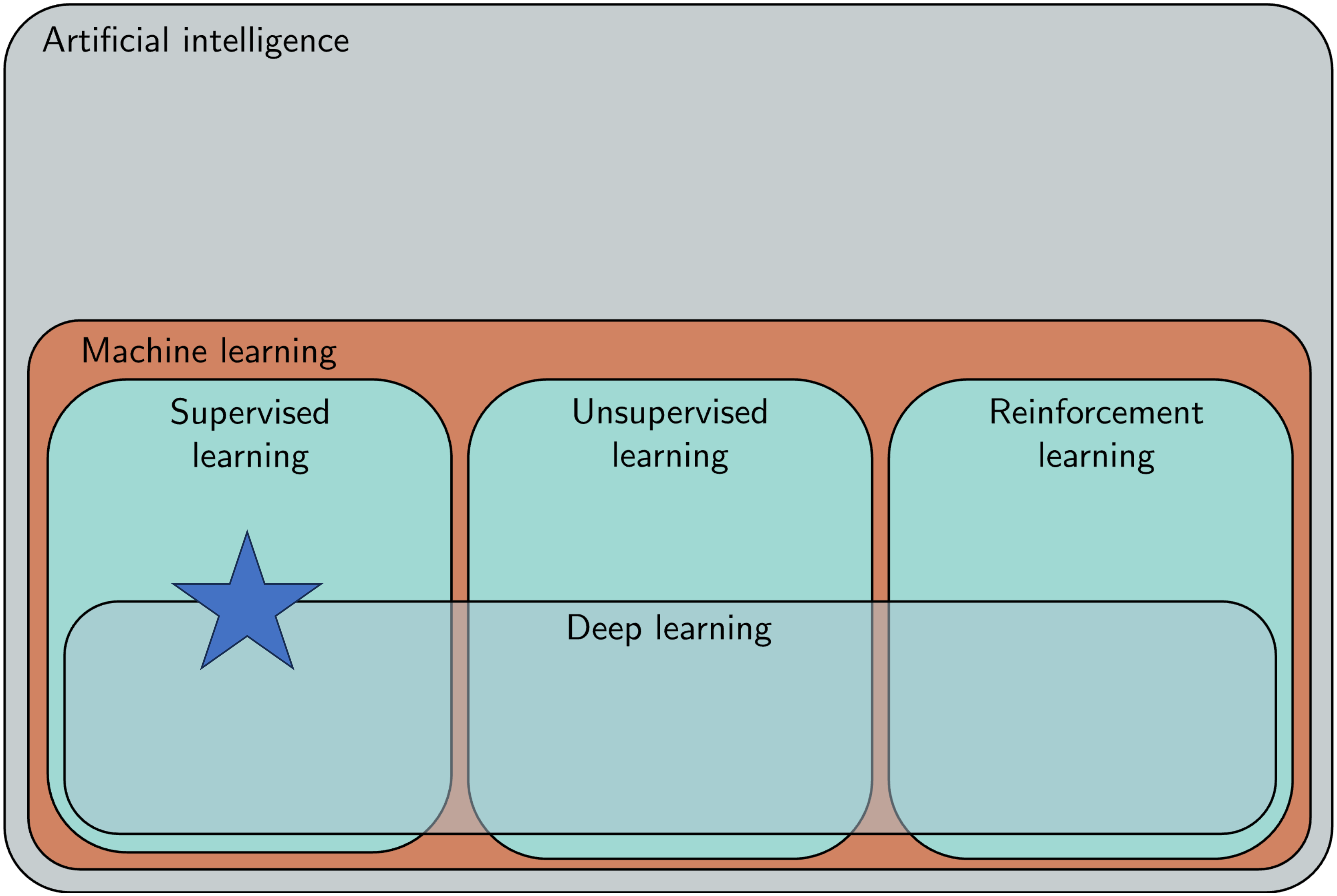
Supervised Learning Terminology and Concepts

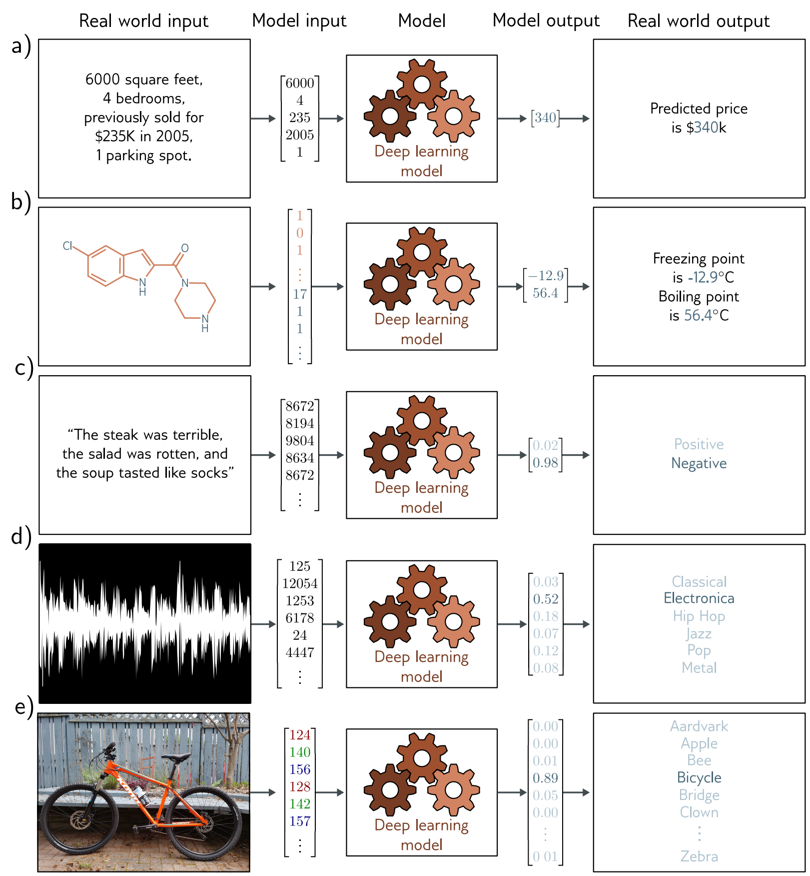
DL4DS Spring 2025

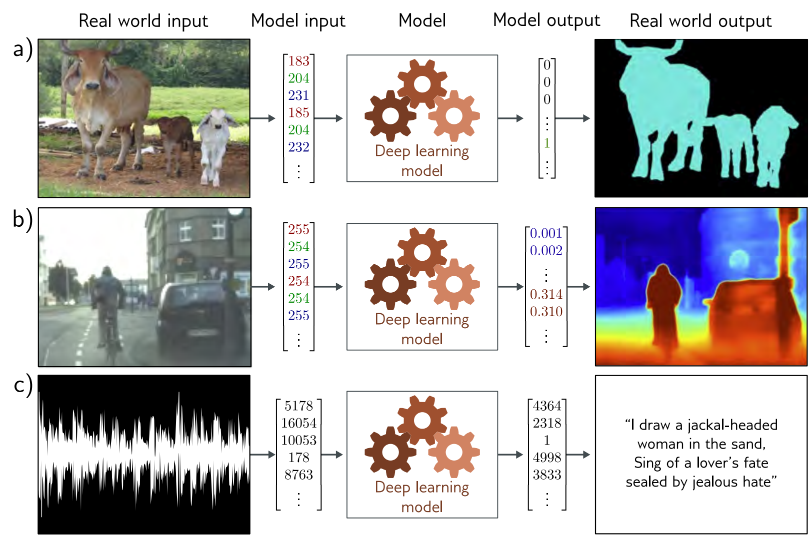
Lecture Outline

- Homeworks and Jupyter Notebooks plan
- Supervised Learning
- More on Projects

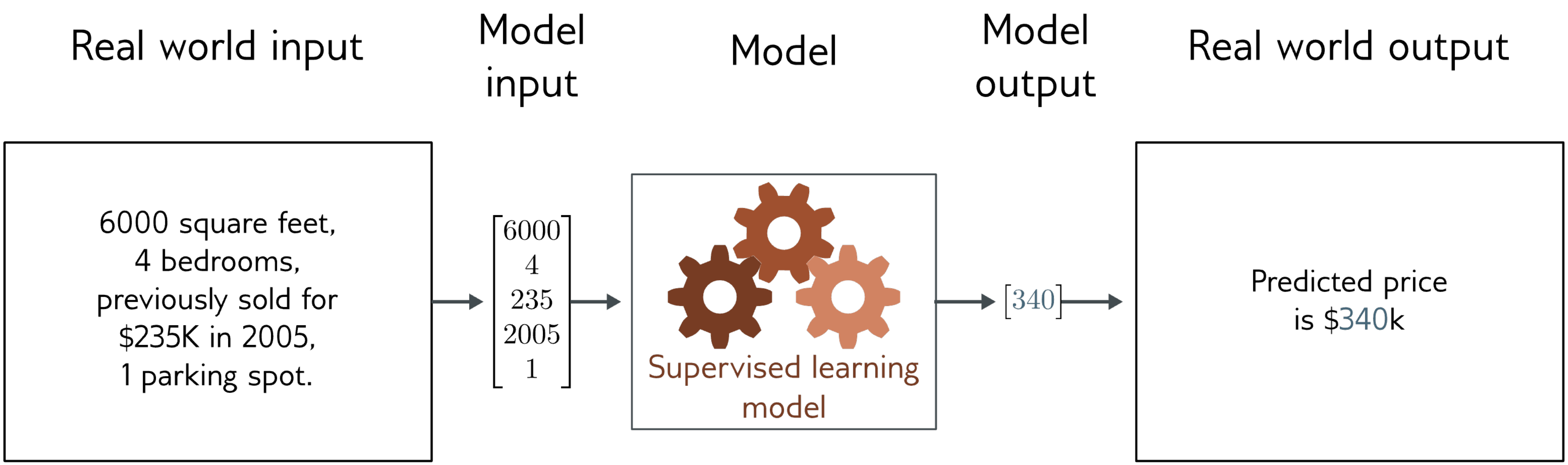

Artificial intelligence
Machine learning
Supervised learning
Unsupervised learning
Reinforcement learning
Deep learning

Supervised Learning Classification and Regression Applications





Regression



- Univariate regression problem (one output, real value)

Supervised learning

- Overview
- Notation
    - Model
    - Loss function
    - Training
    - Testing
- 1D Linear regression example
    - Model
    - Loss function
    - Training
    - Testing
- Where are we going?

Supervised learning

- Overview
- Notation
    - Model
    - Loss function
    - Training
    - Testing
- 1D Linear regression example
    - Model
    - Loss function
    - Training
    - Testing
- Where are we going?

Supervised learning overview

- Supervised learning model = mapping from one or more inputs to one or more outputs
- Model is a family of equations → "inductive bias"
- Computing the outputs from the inputs → inference
- Model also includes parameters
- Parameters affect outcome of equation
- Training a model = finding parameters that predict outputs "well" from inputs for training and evaluation datasets of input/output pairs

Supervised learning

- Overview
- Notation
  - Model
  - Loss function
  - Training
  - Testing
- 1D Linear regression example
  - Model
  - Loss function
  - Training
  - Testing
- Where are we going?

Also Appendix A of the book.

Notation:

- Input:

$$\mathbf{x}$$

Variables always Roman letters

Normal lower case = scalar
Bold lower case = vector
Capital Bold = matrix

- Output:

$$\mathbf{y}$$

- Model:

$$\mathbf{y} = \mathbf{f}[\mathbf{x}]$$

Functions always square brackets

Normal lower case = returns scalar
Bold lower case = returns vector
Capital Bold = returns matrix

Notation example:

- Input:

$$\mathbf{x} = \begin{bmatrix} \text{age} \\ \text{mileage} \end{bmatrix}$$

Vector: Structured or tabular data

- Output:

$$y = \begin{bmatrix} \text{price} \end{bmatrix}$$

Scalar output

- Model:

$$y = \mathrm{f}[\mathbf{x}]$$

Scalar output function (with vector input)

Model

- Parameters:

$$\phi$$

Parameters always Greek letters

- Model :

$$\mathbf{y} = \mathbf{f}[\mathbf{x}, \phi]$$

Data Set and Loss function

- Training dataset of I pairs of input/output examples:

$$\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}$$

Data Set and Loss function

- Training dataset of I pairs of input/output examples:

$$\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}$$

- Loss function or cost function measures how bad model is:

$$L\Big[\boldsymbol{\phi}, \underbrace{\mathrm{f}[\mathbf{x}, \boldsymbol{\phi}]}_{\text{model}}, \underbrace{\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}}_{\text{train data}}\Big]$$

Dataset and Loss function

- Training dataset of I pairs of input/output examples:

$$\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}$$

- Loss function or cost function measures how bad model is:

$$L\Big[\boldsymbol{\phi}, \underbrace{\mathrm{f}[\mathbf{x}, \boldsymbol{\phi}]}_{\text{model}}, \underbrace{\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}}_{\text{train data}}\Big]$$

or for short:

$$L[\boldsymbol{\phi}]$$

Returns a scalar that is smaller when model maps inputs to outputs better

Training

- Loss function:

$$L\left[\phi\right]$$

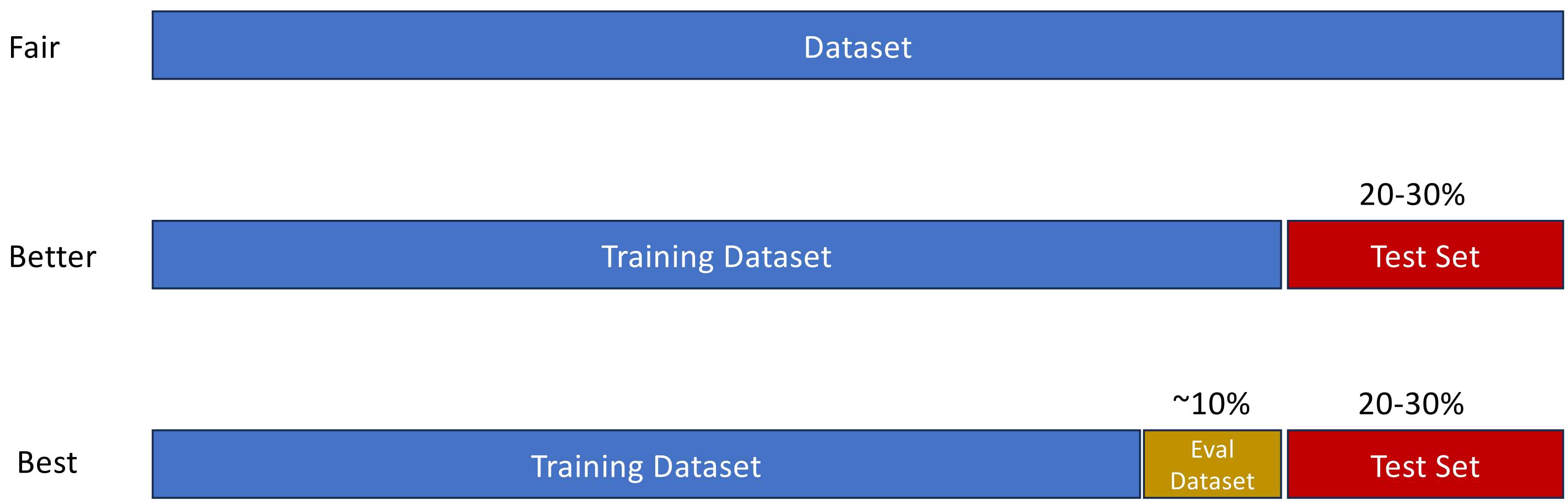
Returns a scalar that is smaller when model maps inputs to outputs better

- Find the parameters that minimize the loss:

$$\hat{\phi} = \underset{\phi}{\operatorname{argmin}}\Big[\mathrm{L}\left[\phi\right]\Big]$$

Testing (and evaluating)

- To test the model, run on a separate test dataset of input / output pairs
- See how well it generalizes to new data



Supervised learning

- Overview
- Notation
  - Model
  - Loss function
  - Training
  - Testing
- 1D Linear regression example
  - Model
  - Loss function
  - Training
  - Testing
- Where are we going?

Example: 1D Linear regression model

- Model:

$$
\begin{aligned}
y &= \mathrm{f}[x, \boldsymbol{\phi}] \\
&= \phi_0 + \phi_1 x
\end{aligned}
$$

- Parameters

$$
\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix}
\begin{array}{l} \longleftarrow \text{y-offset} \\ \longleftarrow \text{slope} \end{array}
$$

Example: 1D Linear regression model

- Model:

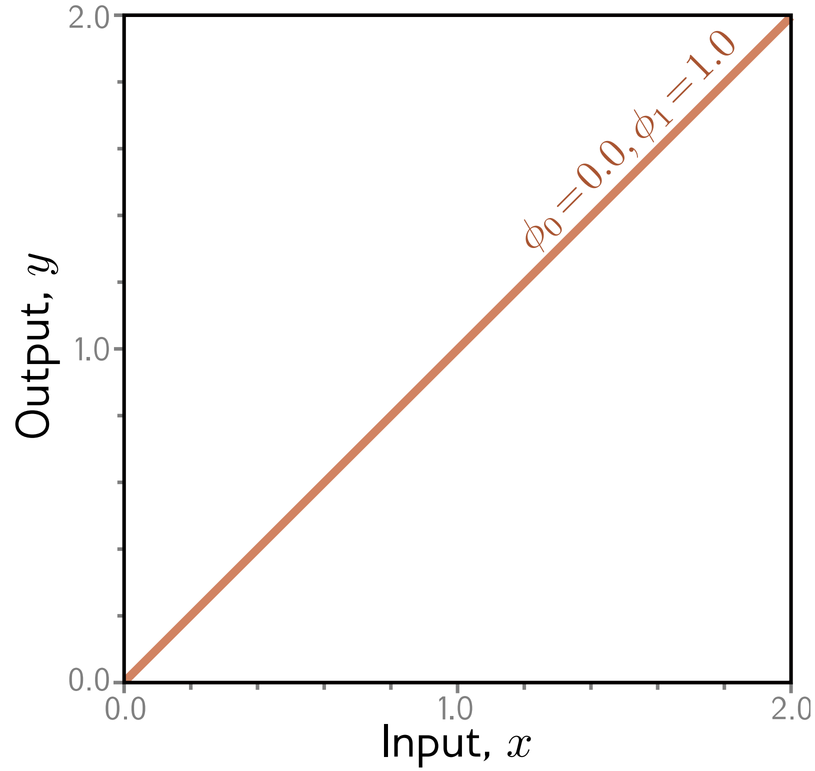
$$y = \mathrm{f}[x, \boldsymbol{\phi}]$$
$$= \phi_0 + \phi_1 x$$

- Parameters

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix}$$

← y-offset

← slope



Example: 1D Linear regression model

- Model:

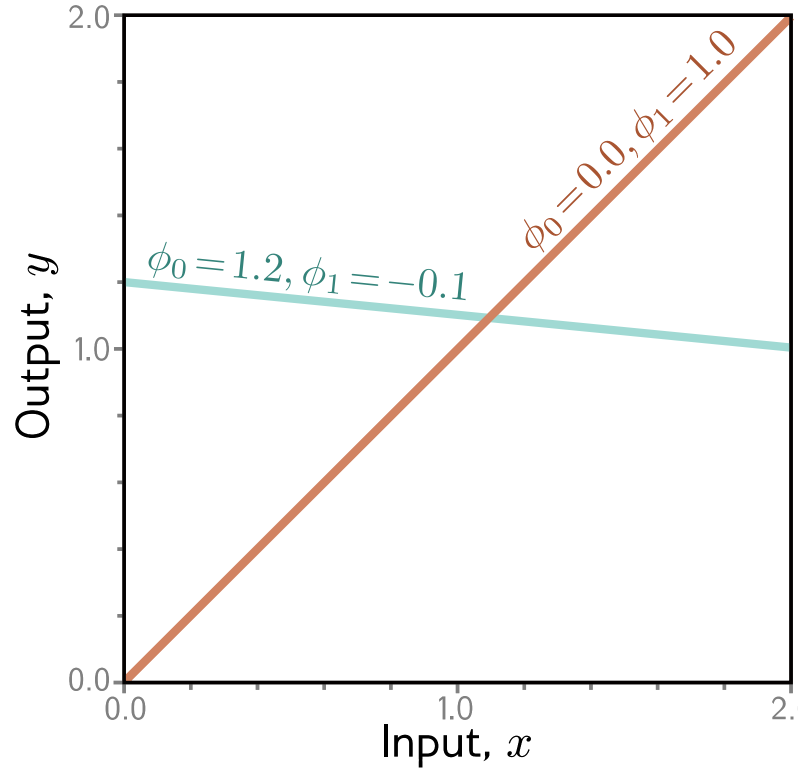
$$y = \mathrm{f}[x, \boldsymbol{\phi}]$$
$$= \phi_0 + \phi_1 x$$

- Parameters

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix}$$

y-offset

slope



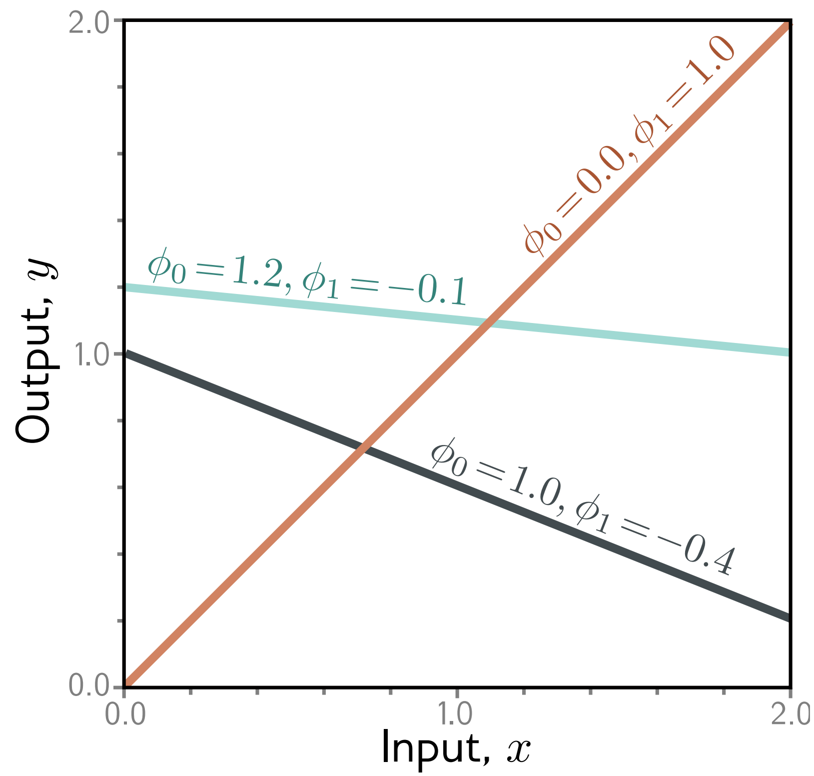
Example: 1D Linear regression model

- Model:

$$y = \mathrm{f}[x, \boldsymbol{\phi}]$$
$$= \phi_0 + \phi_1 x$$

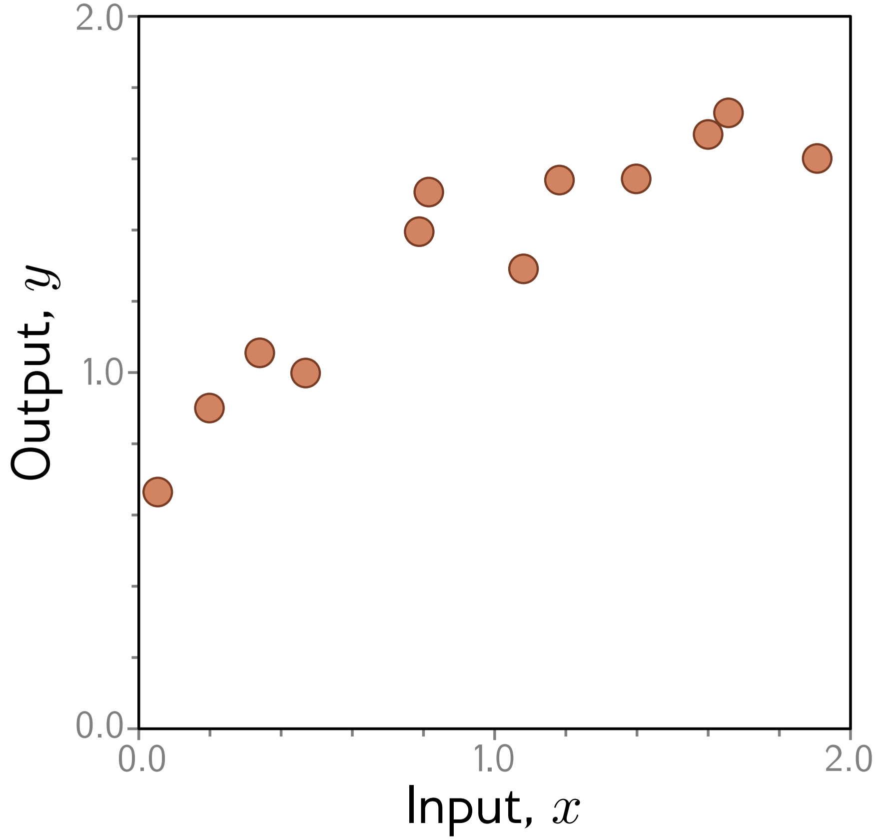
- Parameters

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix} \begin{matrix} \leftarrow \text{y-offset} \\ \leftarrow \text{slope} \end{matrix}$$

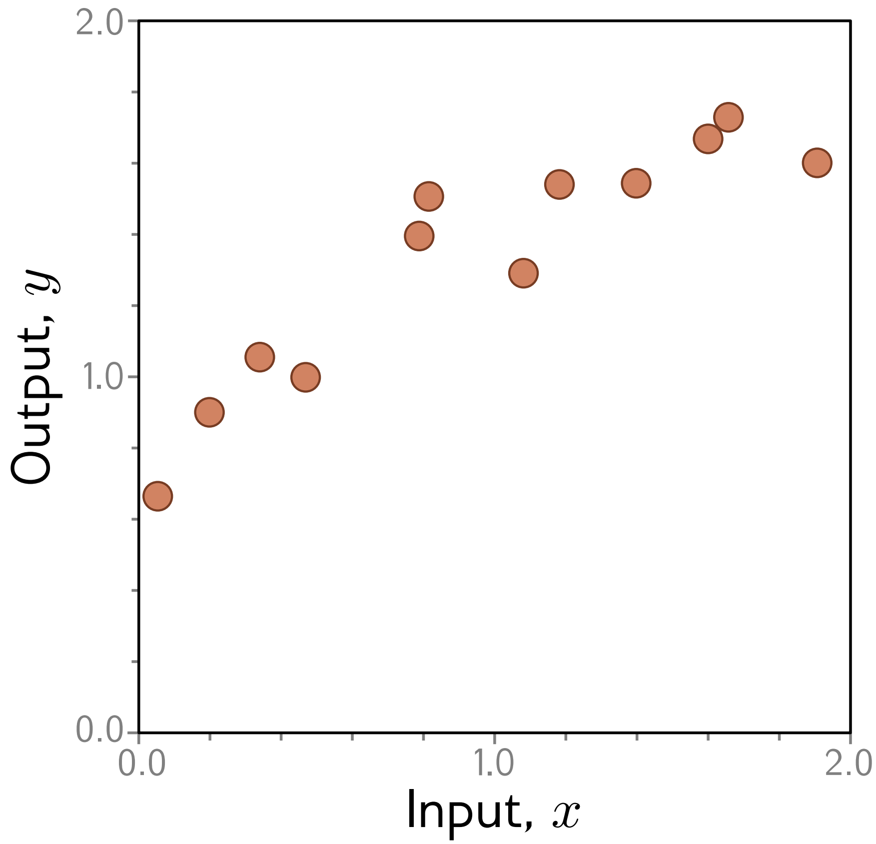


[Interactive Figure 2.1]

Example: 1D Linear regression training data



Example: 1D Linear regression training data

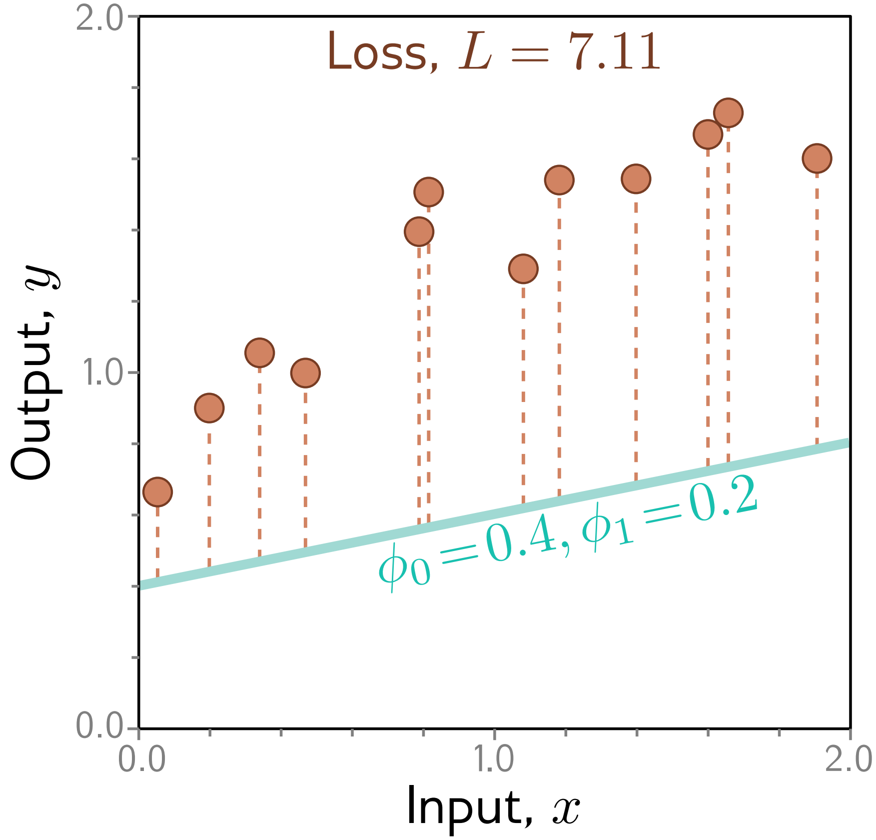


Loss function:

$$L[\phi] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \phi] - y_i)^2$$
$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Example: 1D Linear regression loss function



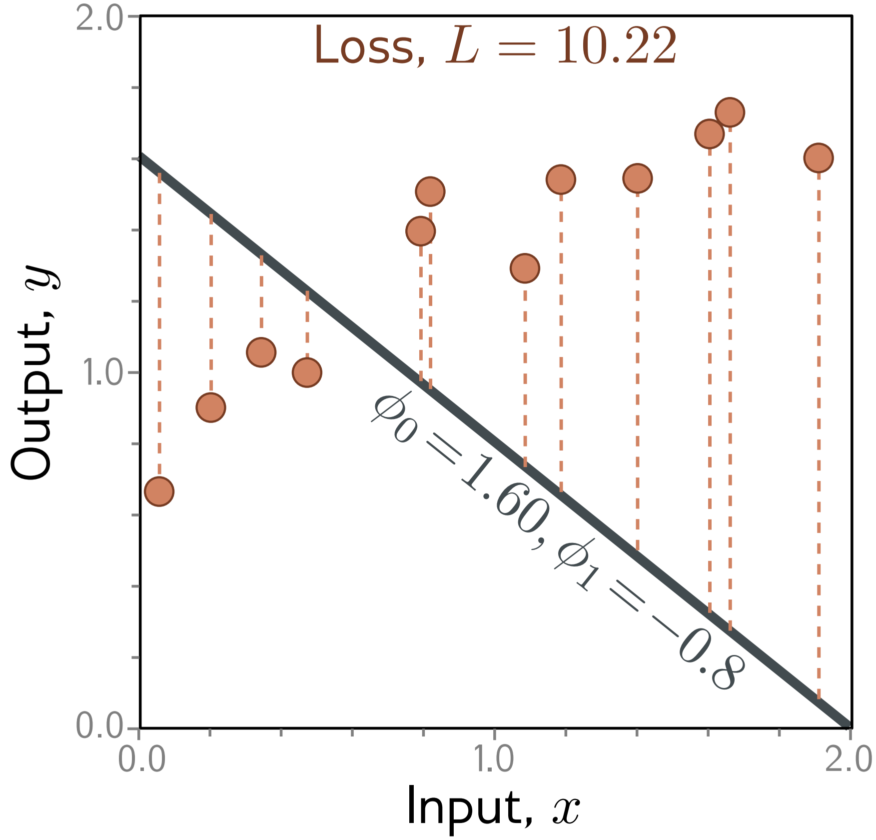
Loss function:

$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \boldsymbol{\phi}] - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Example: 1D Linear regression loss function



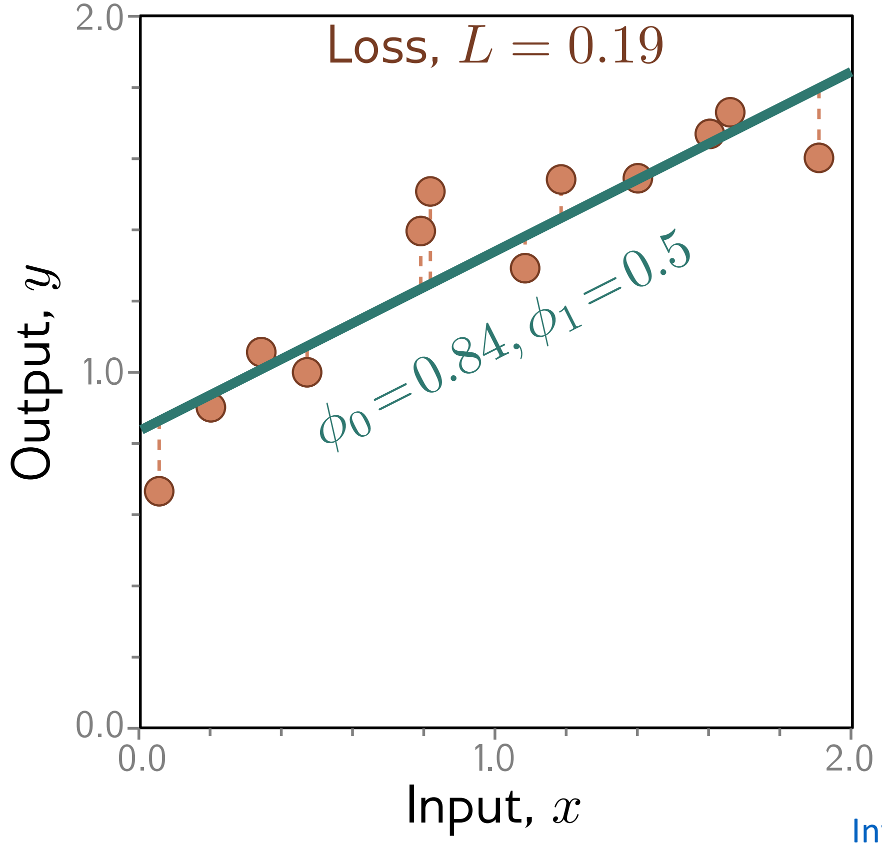
Loss function:

$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \boldsymbol{\phi}] - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Example: 1D Linear regression loss function



Loss function:

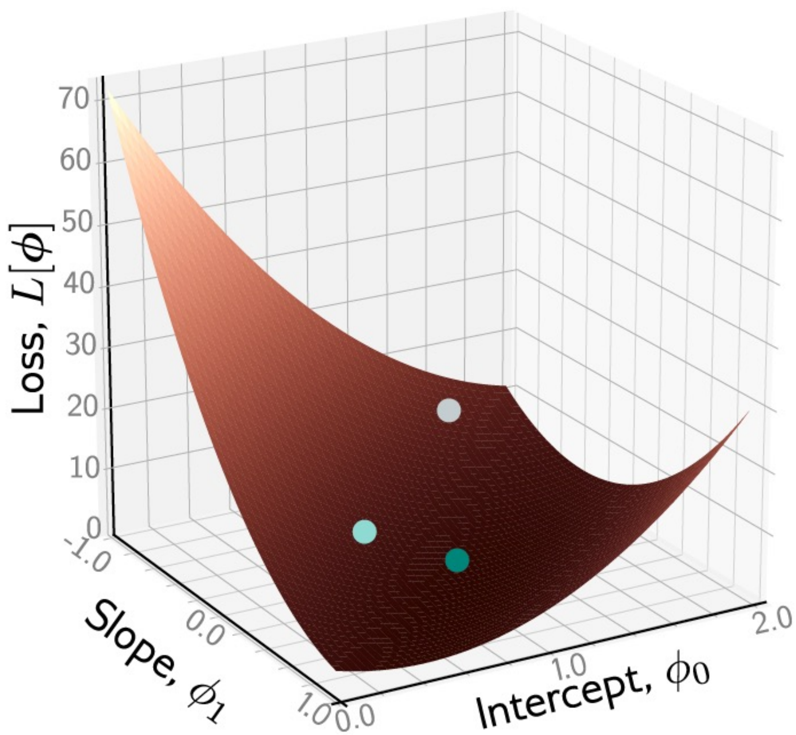
$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \boldsymbol{\phi}] - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Interactive Figure 2.2

Example: 1D Linear regression loss function



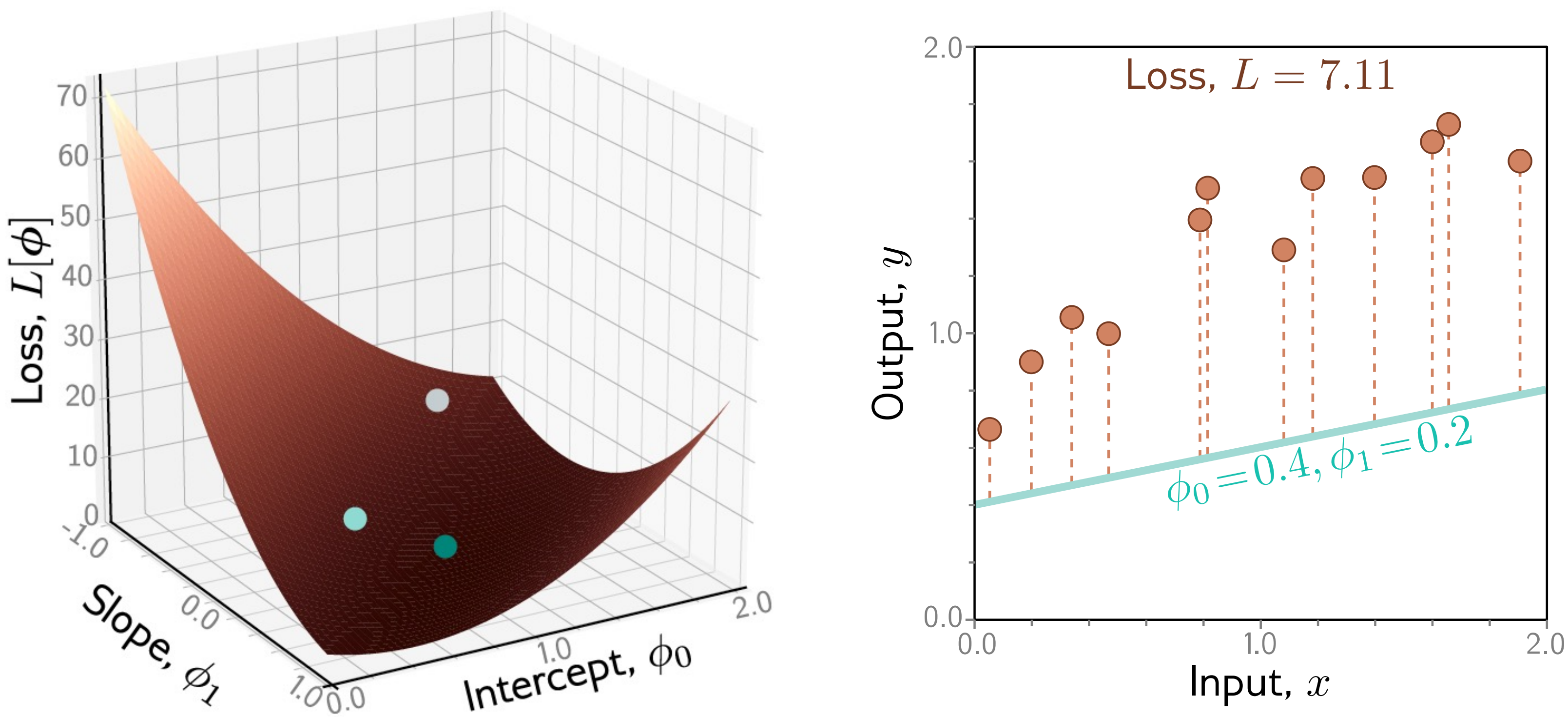
Loss function:

$$L[\phi] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \phi] - y_i)^2$$

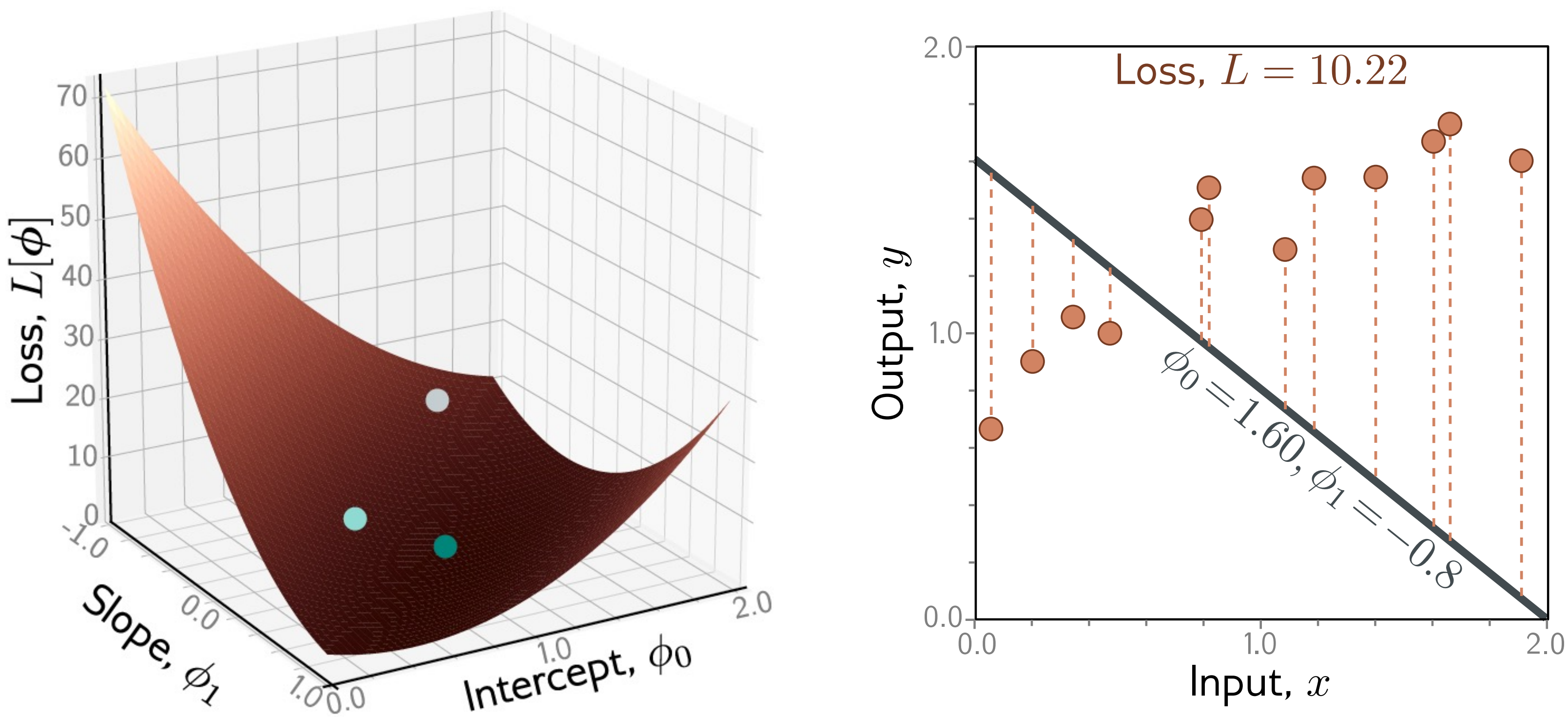
$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

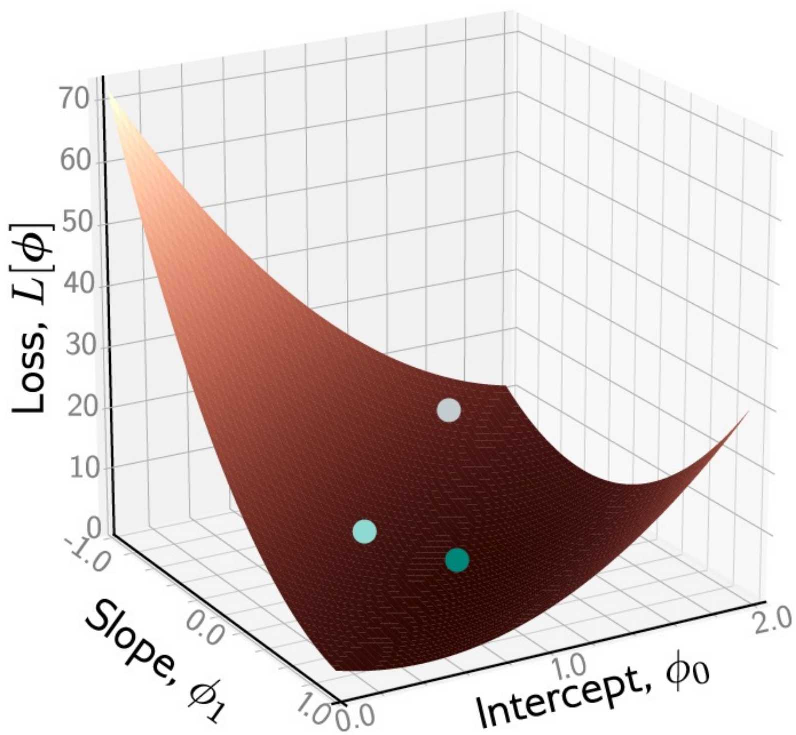
Example: 1D Linear regression loss function

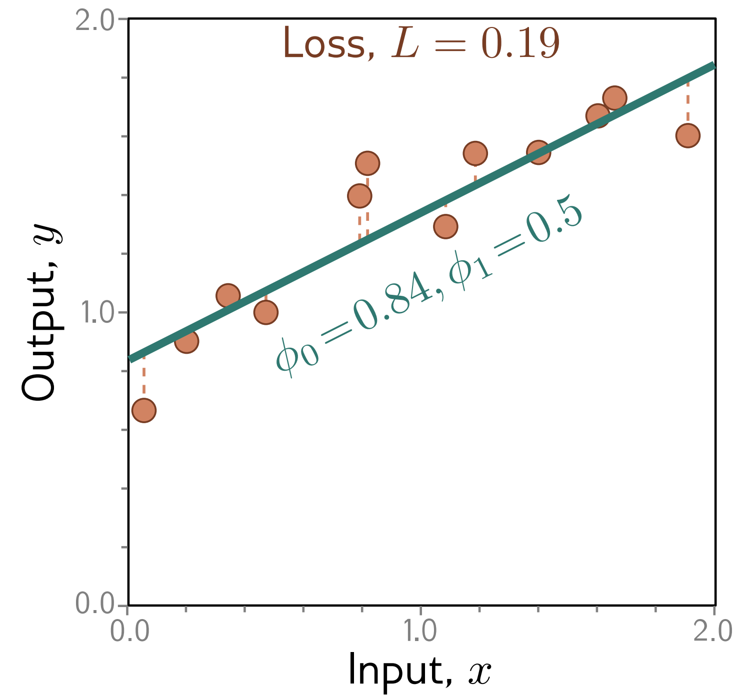


Example: 1D Linear regression loss function

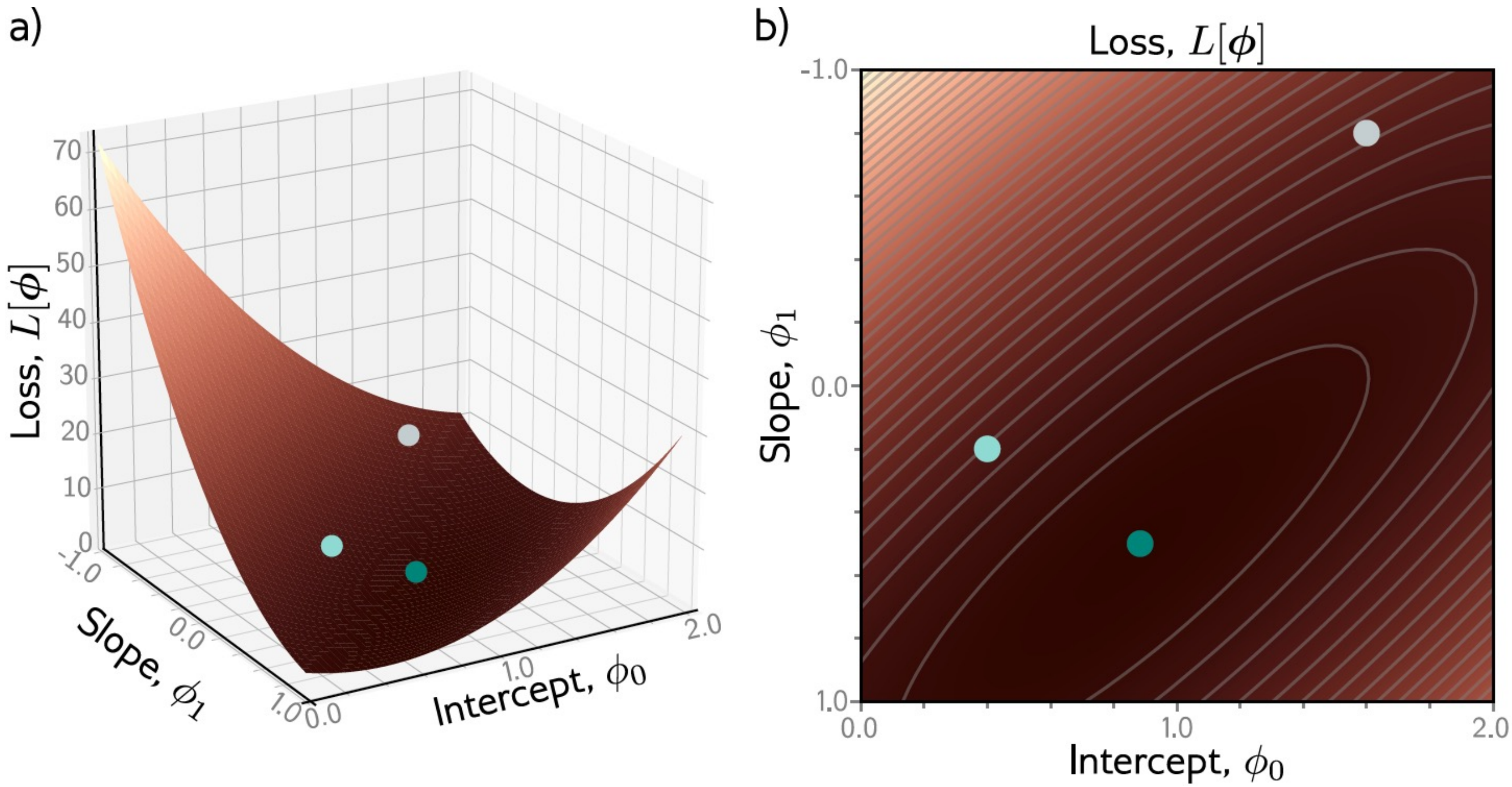


Example: 1D Linear regression loss function

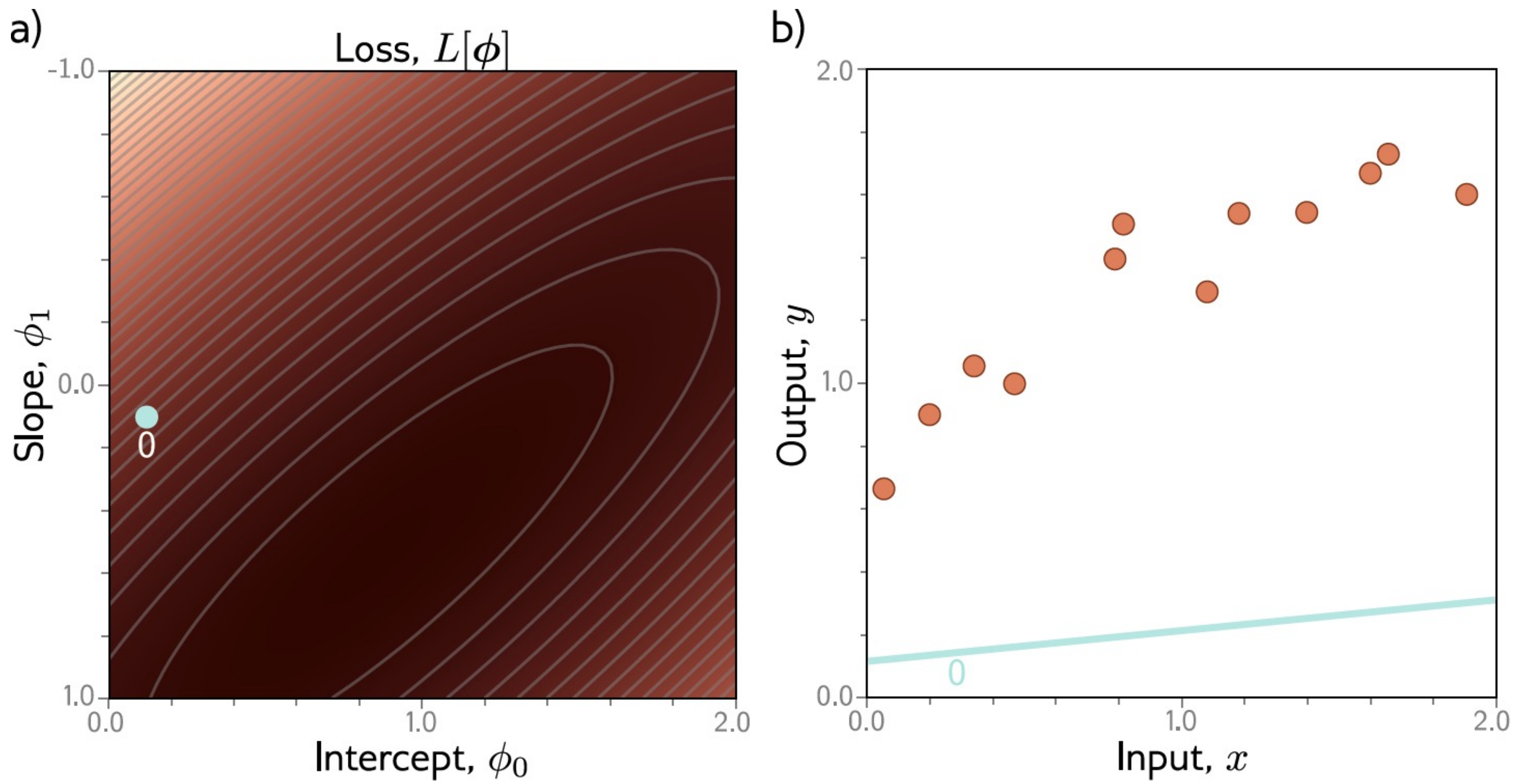




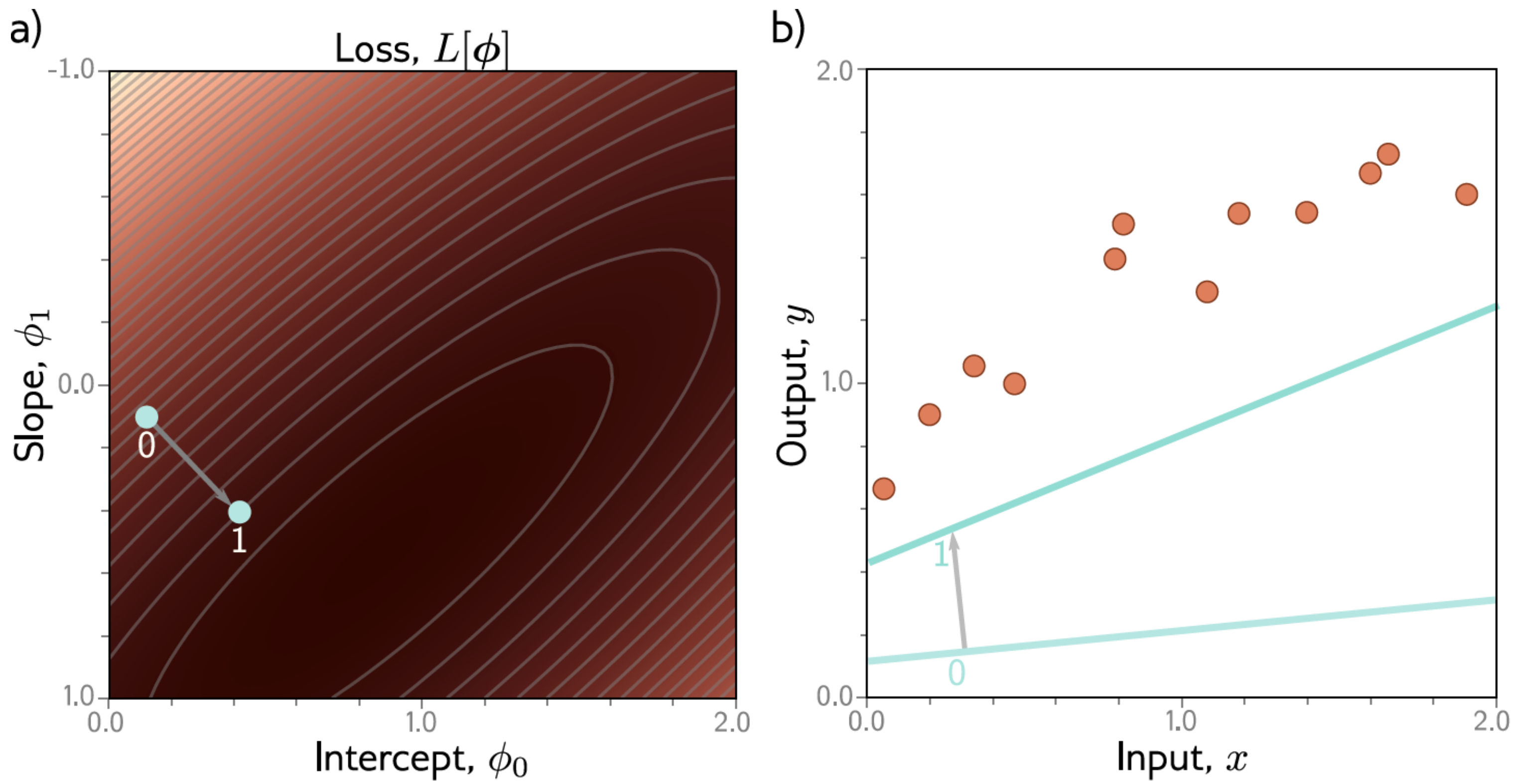
Example: 1D Linear regression loss function



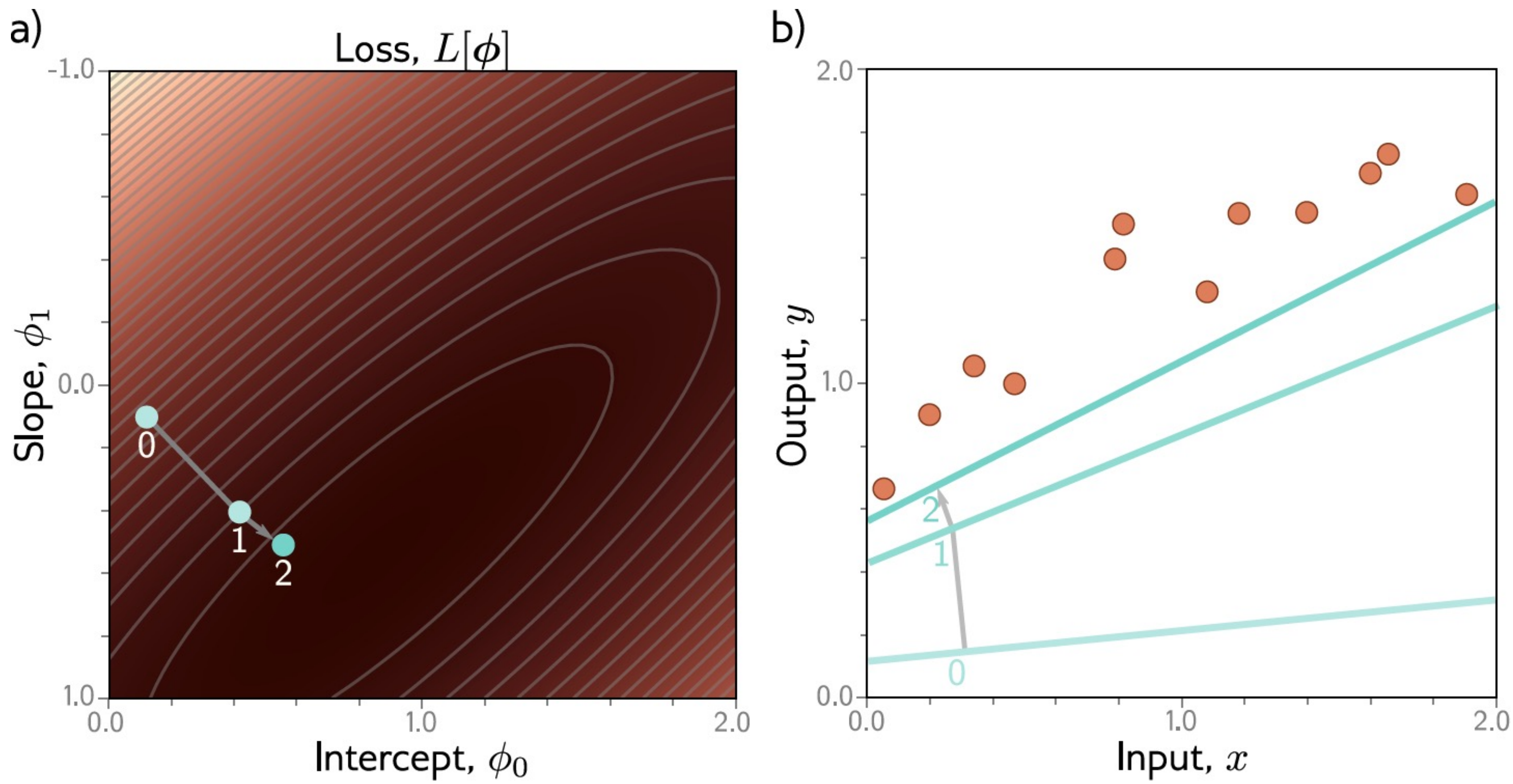
Example: 1D Linear regression training



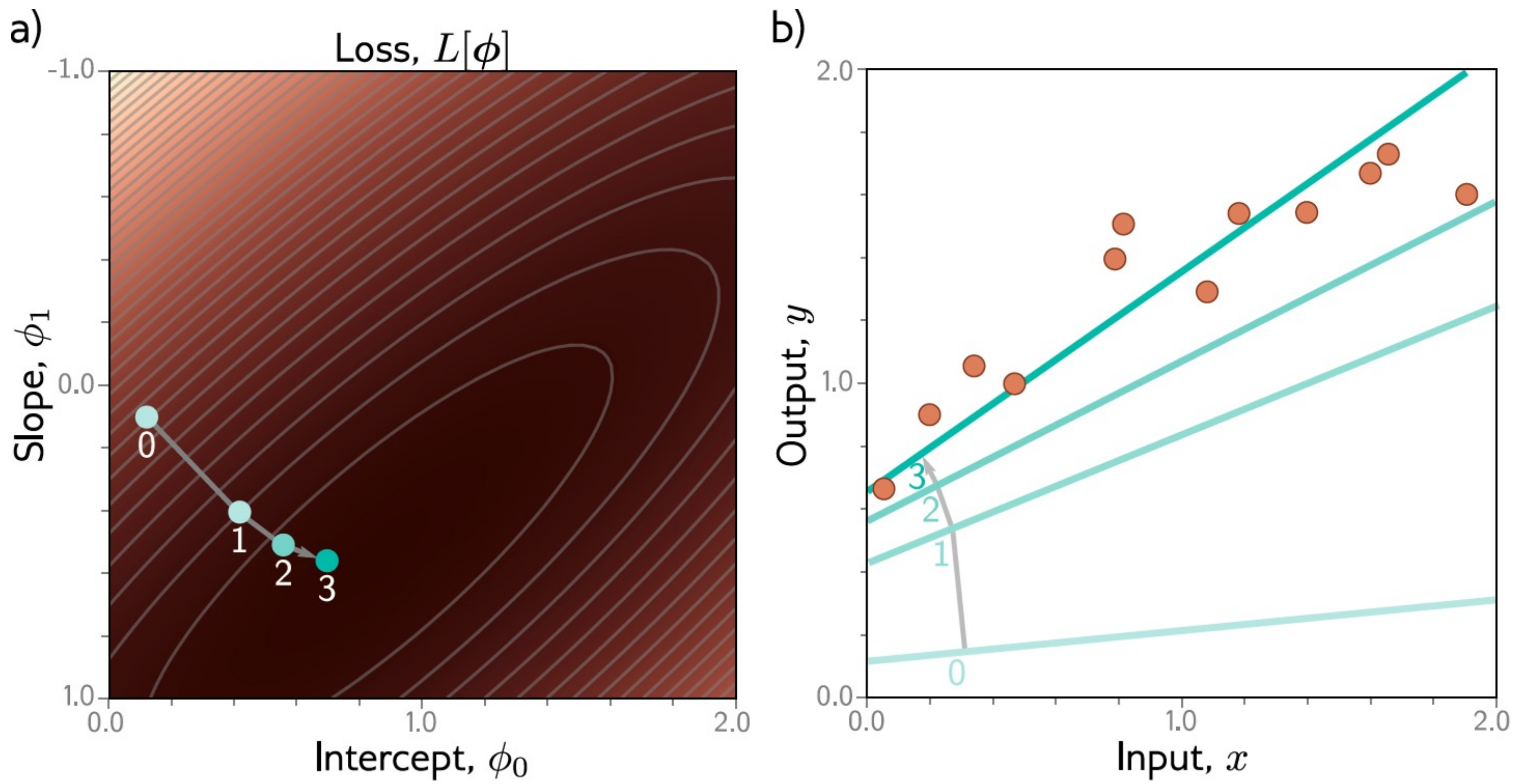
Example: 1D Linear regression training



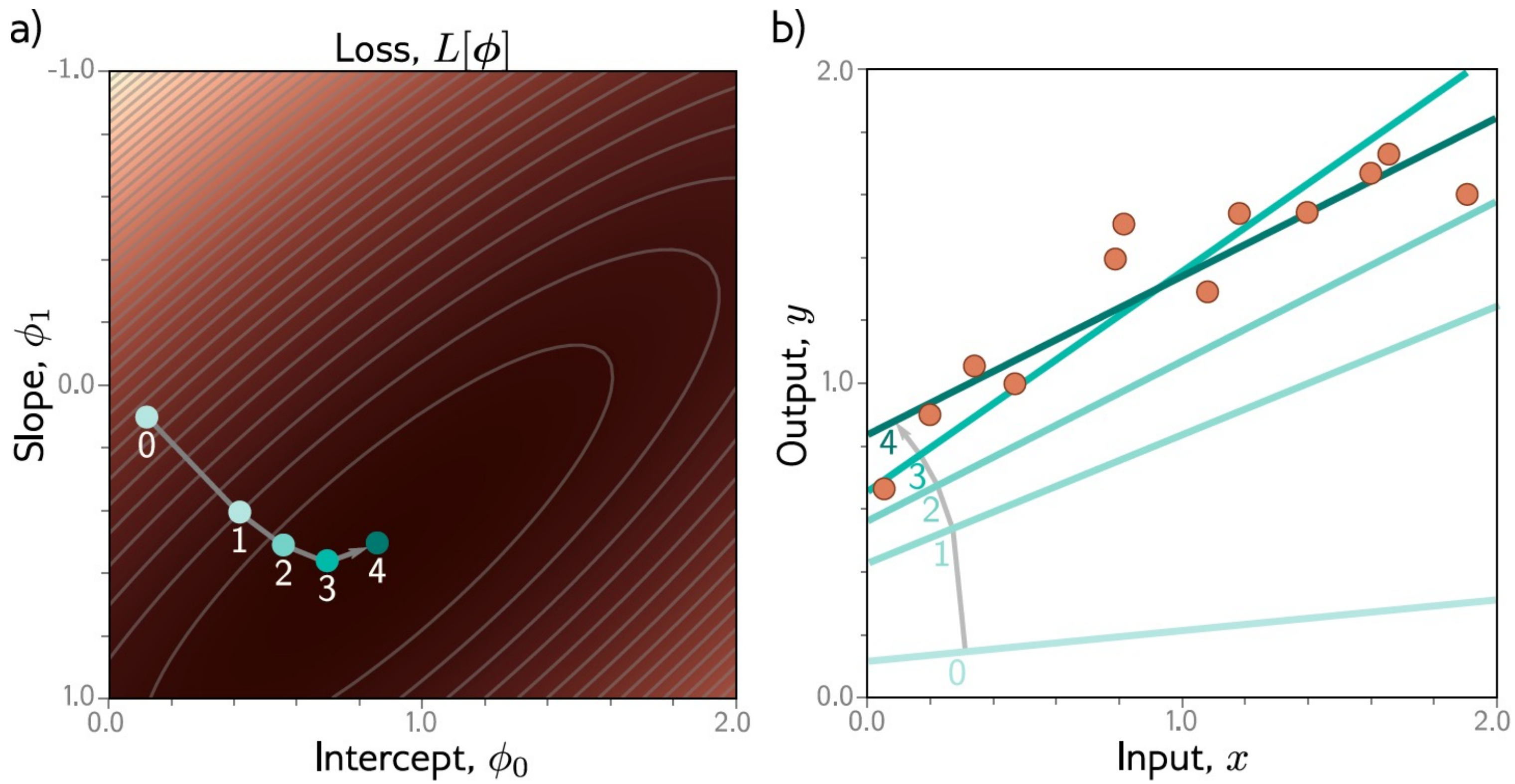
Example: 1D Linear regression training



Example: 1D Linear regression training



Example: 1D Linear regression training

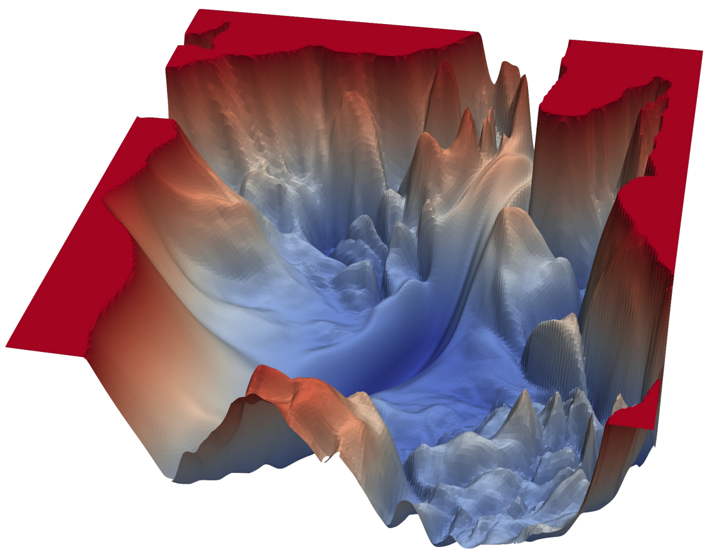


This technique is known as gradient descent

Interactive Figure 2.3

Possible objections

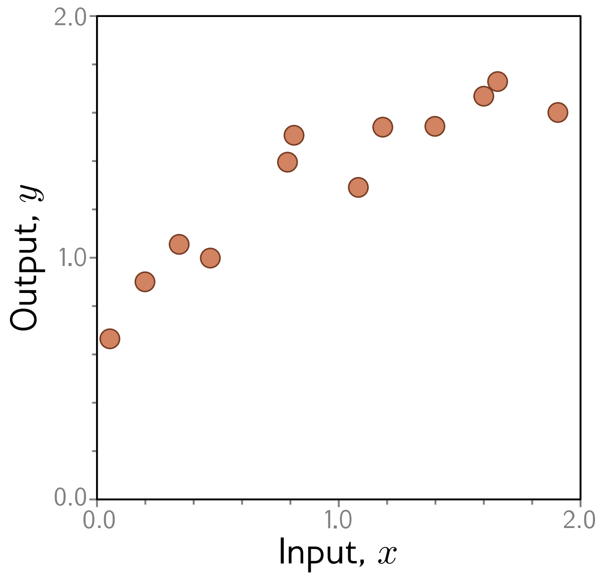
- But you can fit the line model in closed form!
    - Yes – but we won't be able to do this for more complex models
- But we could exhaustively try every slope and intercept combo!
    - Yes – but we won't be able to do this when there are a million parameters



Here's a visualization of the loss surface for the 56-layer neural network VGG-56(from Visualizing the Loss Landscape of Neural Networks -- https://losslandscape.com/explorer

Example: 1D Linear regression testing

- Test with different set of paired input/output data (Test Set)
  - Measure performance
  - Degree to which *Loss* is same as training = generalization
- Might not generalize well because
  - Model too simple: underfitting
  - Model too complex
    - fits to statistical peculiarities of data
    - this is known as overfitting



Piazza Poll

- https://piazza.com/class/m5v834h9pcatx/post/12

Supervised learning

- Overview
- Notation
    - Model
    - Loss function
    - Training
    - Testing
- 1D Linear regression example
    - Model
    - Loss function
    - Training
    - Testing
- Where are we going?

Where are we going? Next lectures…

- Shallow neural networks (a more flexible model)
- Deep neural networks (even more flexible with fewer parameters)
- Loss functions (where did least squares come from?)
- How to train neural networks (gradient descent and variants)
- How to measure performance of neural networks (generalization)

Course Project

- Work in teams of 2-3
- Can be application, algorithmic, theoretical or combination thereof
- Project proposal due ~Feb. 16
- Deliverables:
  - Code in GitHub repo
  - Report/paper
  - 3-4 minute video
- More info later, but feel free to brainstorm with me now

https://dl4ds.github.io/sp2024/miniconf.html

Look at Kaggle, Conferences, Workshops, Datasets….

- Application workshops at major conferences can be good sources of ideas. Often times they are associated with new and interesting datasets. Some potential conferences include:
  - NeurIps,
  - CVPR,
  - ICML,
  - ICMLA,
  - SPIE
- Kaggle and other competition websites can be a source of ideas.
- You might find some interesting datasets at Papers with Code
- Lot of applications are posted on X/Twitter, Reddit, LinkedIn, etc.

Project Grading (45% of course grade)

% of Project Grade	Category	Criteria
20%	Project Report	Conference style paper with complete sections (per template), well written, no typos or formatting issues.
20%	Project Repo/Software	Repo is well documented. Code is reproducible. Top level readme giving project overview, roadmap to directories/files, summary of results.
20%	Final Presentation and Video	Video/presentation is clear and concise, adheres to time limits. Introduces the problem/project, approach, dataset, conclusions, etc.
30%	Individual contribution	Is there clear evidence of project contributions such as commit history or co-authored commits, document revisions. Leave bread crumbs!!
10%	Individual contribution to collaboration and teamwork	Is there indication, e.g. from peer surveys, of collaboration and constructive teamwork?

Project proposal and mid-point check-in will count as homework.

Feedback?

[Link]