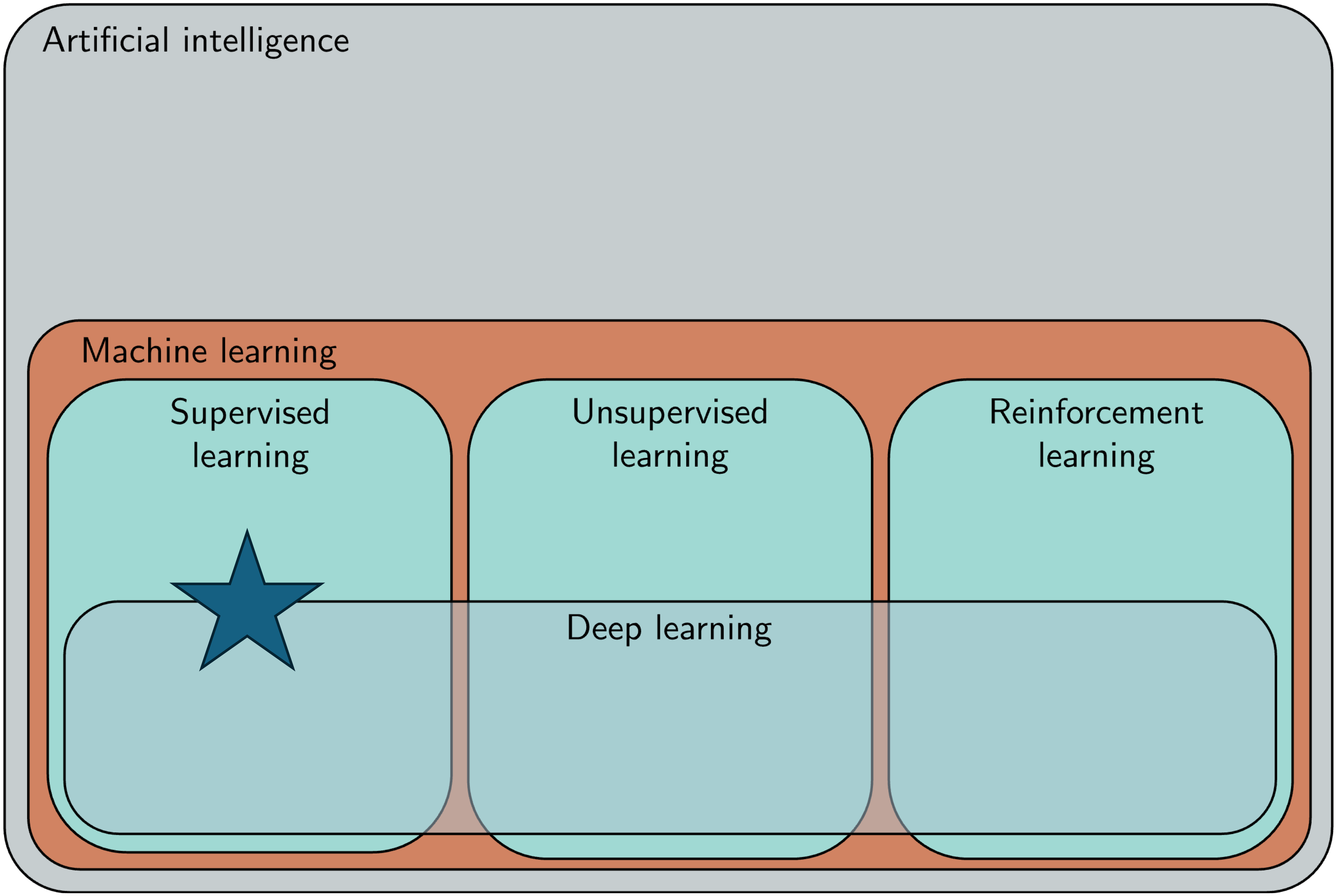
Deep Learning for Data Science
DS 542

https://dl4ds.github.io/sp2026/

Supervised Learning

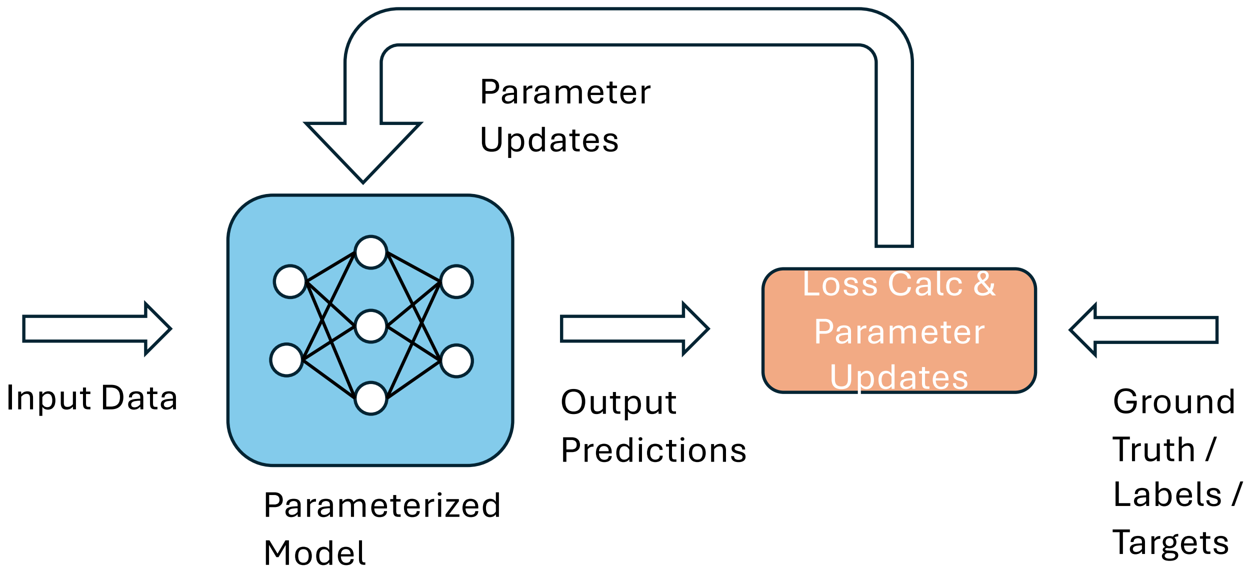
Supervised learning

- Examples
- Terminology
- Notation
  - Model
  - Loss function
  - Training
  - Testing
- 1D Linear regression example
  - Model
  - Loss function
  - Training
  - Testing

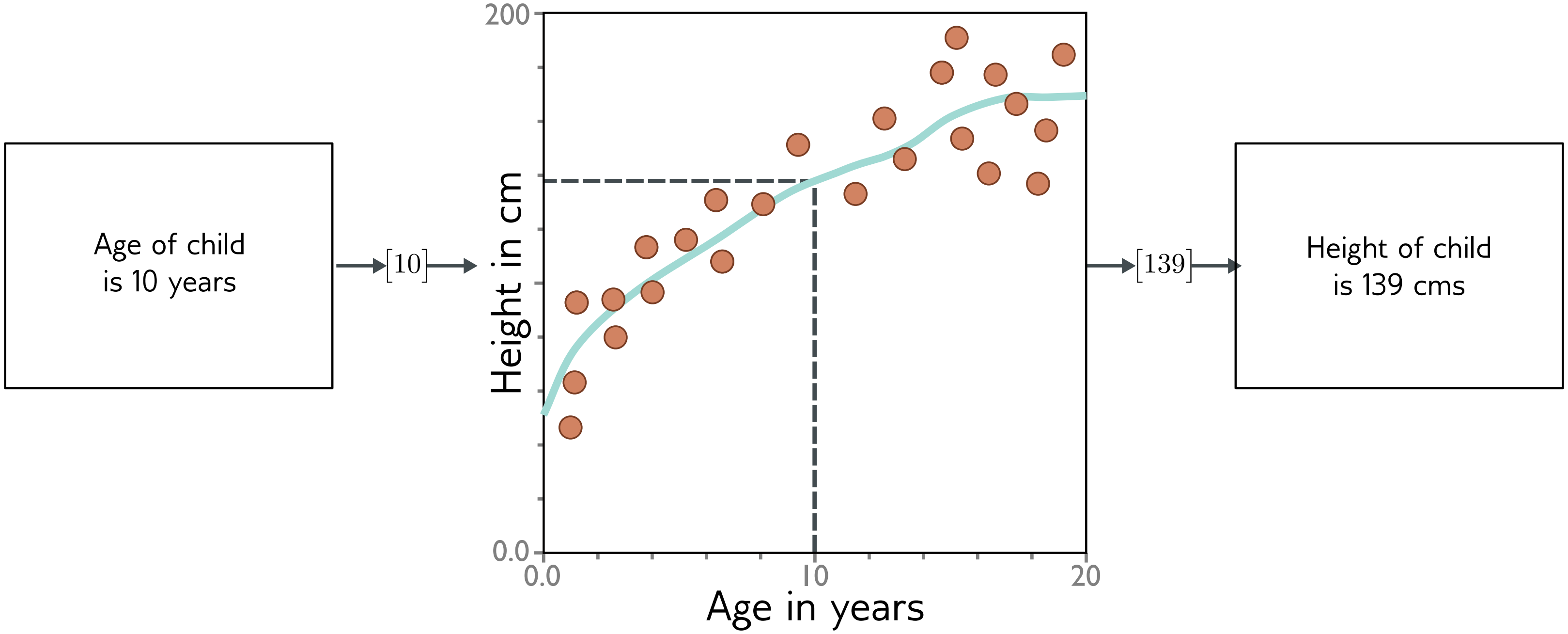

Artificial intelligence
Machine learning
Supervised learning
Unsupervised learning
Reinforcement learning
Deep learning

Supervised learning

- Define a mapping from input to output
- Learn this mapping from paired input/output data examples

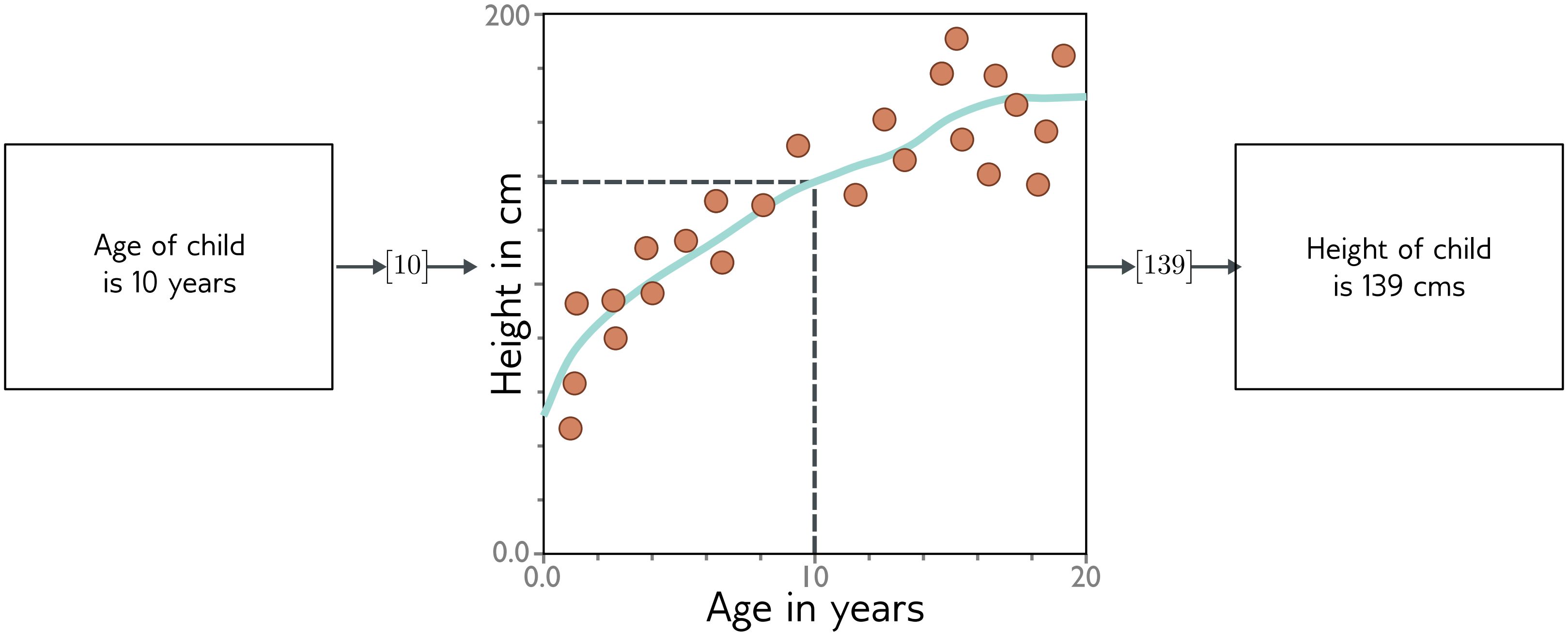


What is a supervised learning model?



- An equation relating input (age) to output (height)
- Search through family of possible equations to find one that fits training data well

What is a supervised learning model?



- Deep neural networks are just a very flexible family of equations
- Fitting deep neural networks = "Deep Learning"

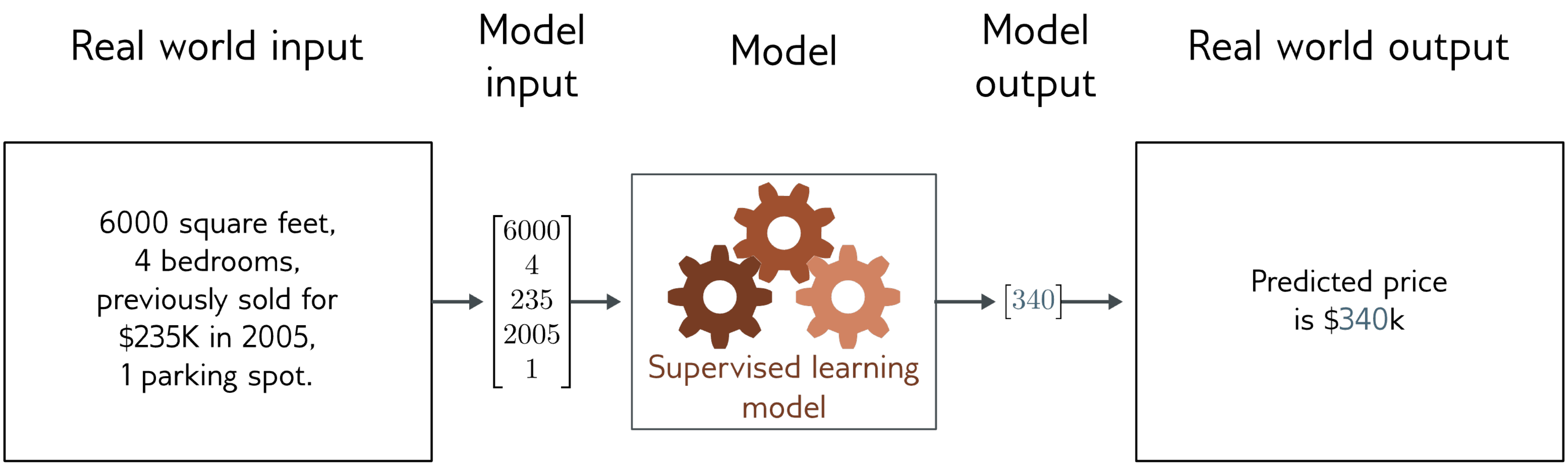
Prediction Types

- Regression
  - Prediction a continuous valued output

- Classification
  - Assigning input to one of a finite number of classes or categories
  - Two classes are a special case

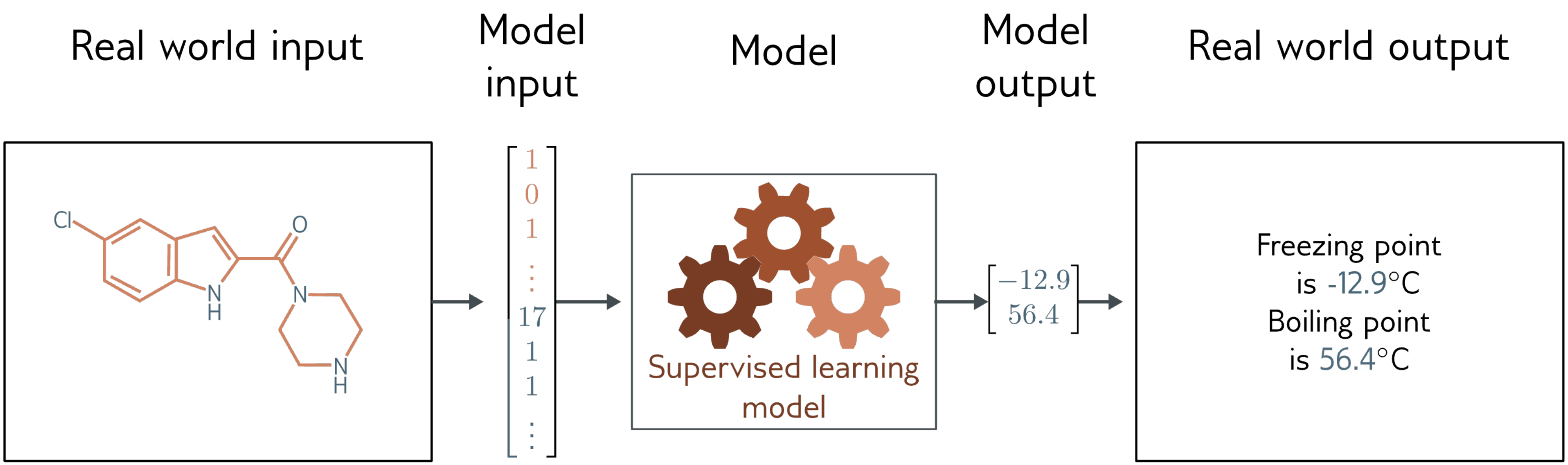
Can be univariate (one output) or multivariate (more than one output)

Regression



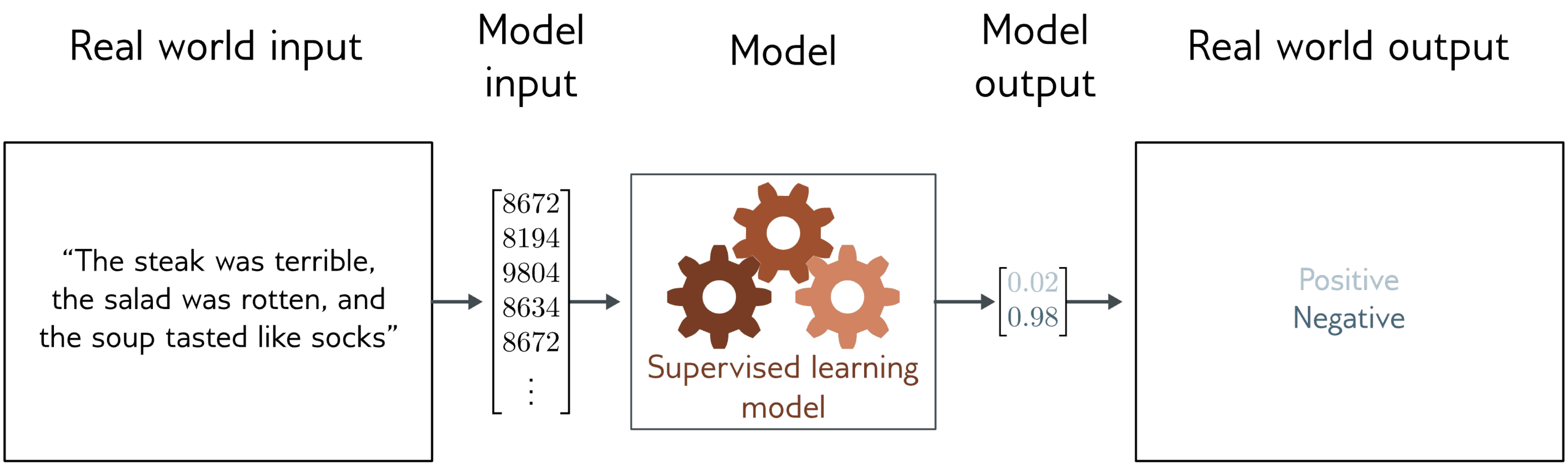
- Univariate regression problem (one output, real value)
- Fully connected network

Graph regression



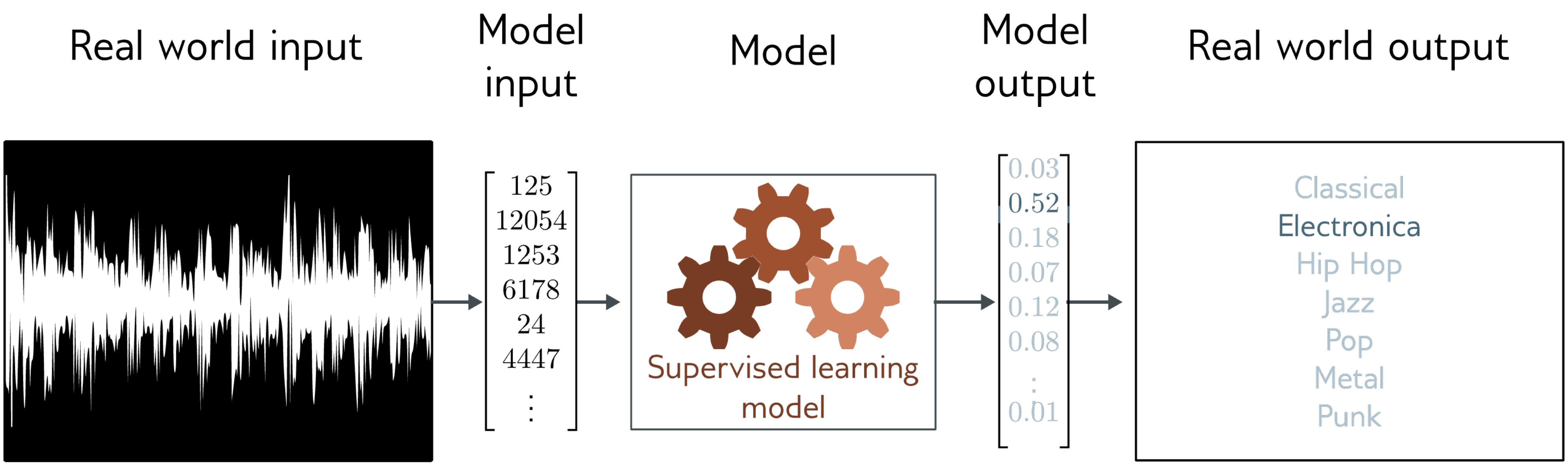
- Multivariate regression problem (>1 output, real value)
- Graph neural network

Text classification



- Binary classification problem (two discrete classes)
- Transformer network

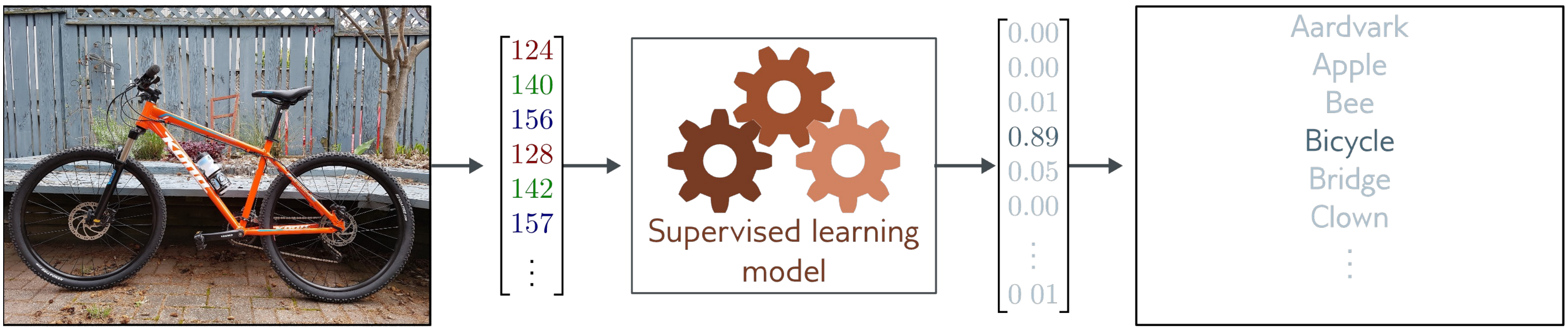
Music genre classification



- Multiclass classification problem (discrete classes, >2 possible values)
- Recurrent neural network (RNN)

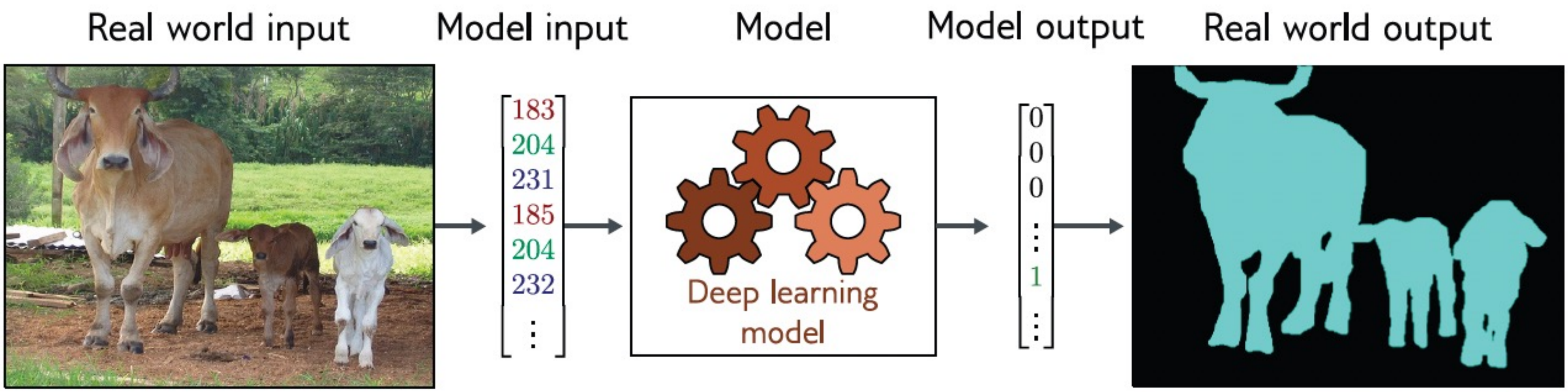
Image classification

Real world input | Model input | Model | Model output | Real world output



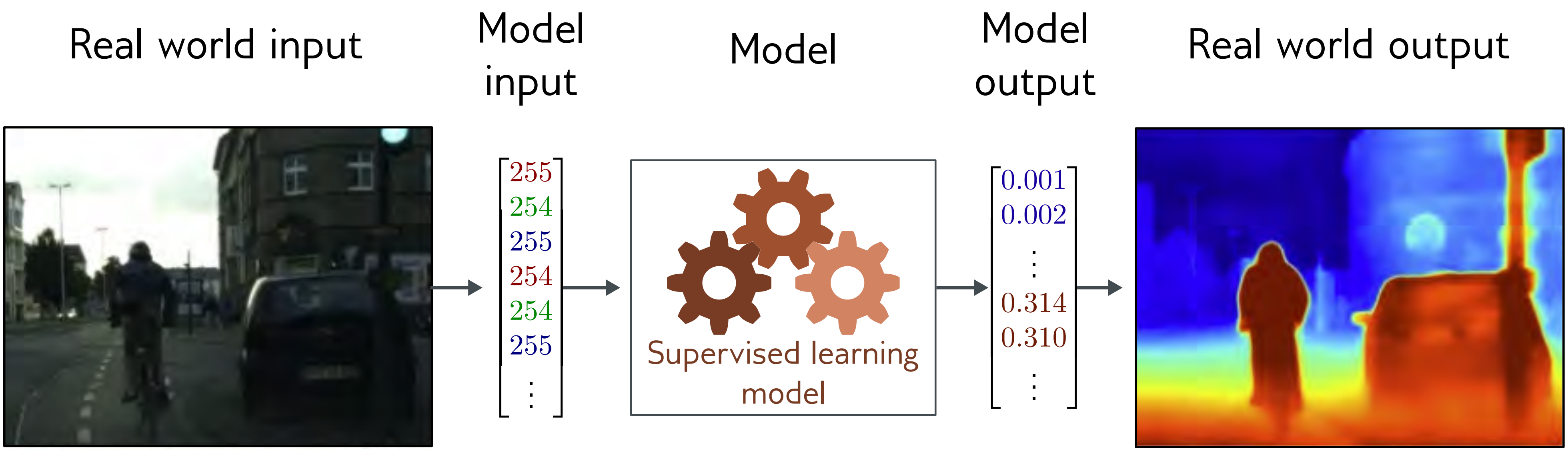
- Multiclass classification problem (discrete classes, >2 possible classes)
- Convolutional network

Image segmentation



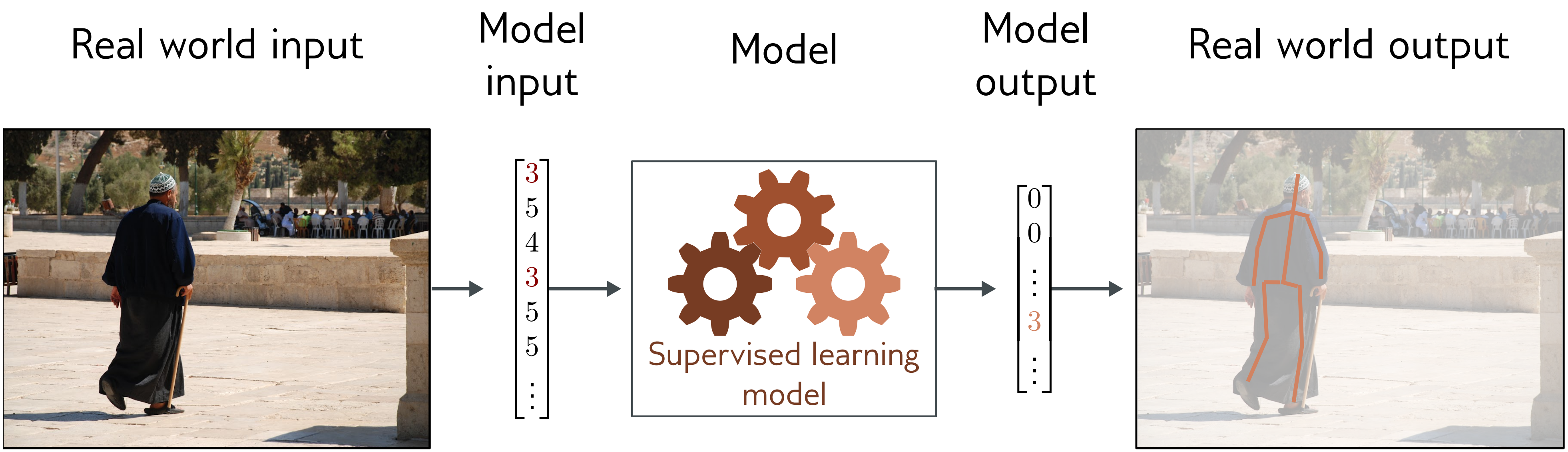
- Multivariate binary classification problem (many outputs, two discrete classes)
- Convolutional encoder-decoder network

Depth estimation



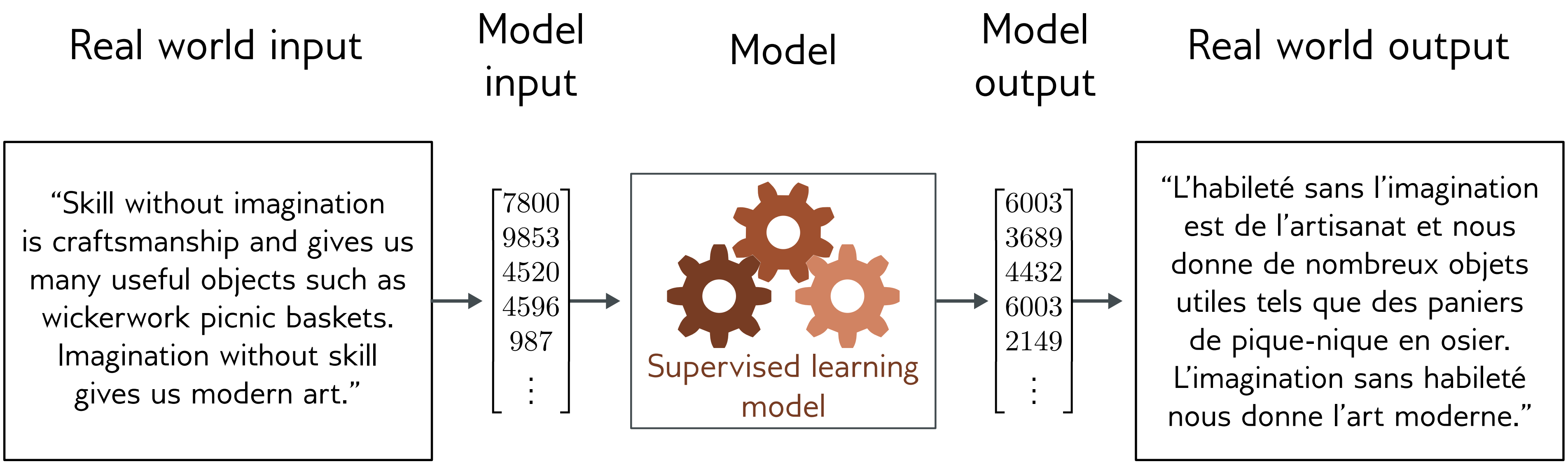
- Multivariate regression problem (many outputs, continuous)
- Convolutional encoder-decoder network

Pose estimation



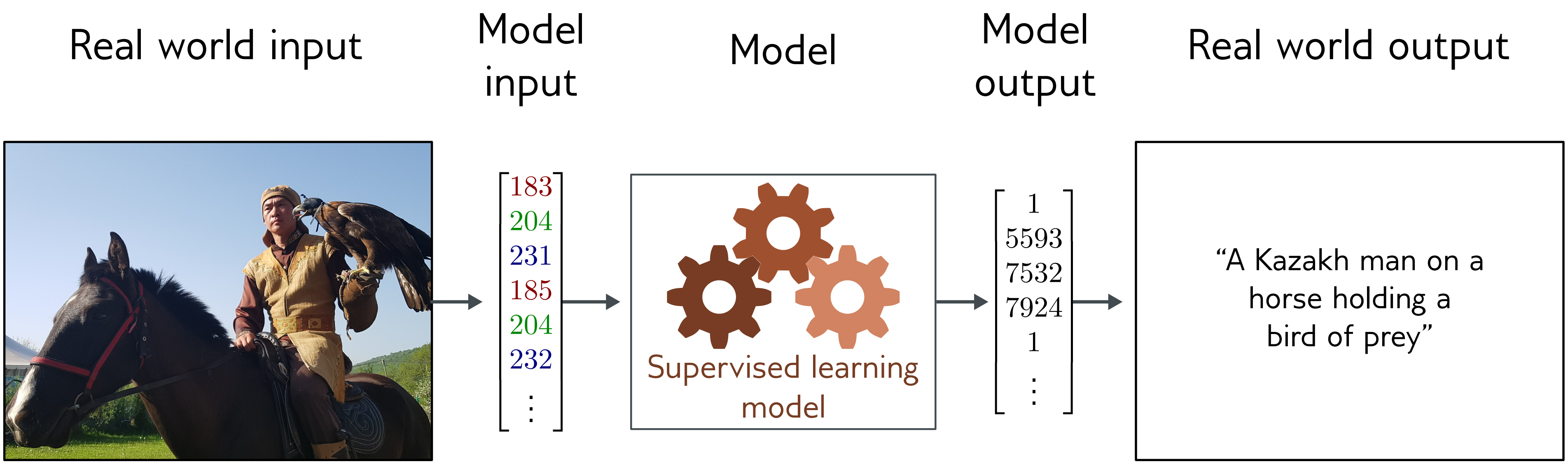
- Multivariate regression problem (many outputs, continuous)
- Convolutional encoder-decoder network

Translation



- Encoder-Decoder Transformer Networks

Image captioning



- E.g. CNN-RNN, LSTM, Transformers

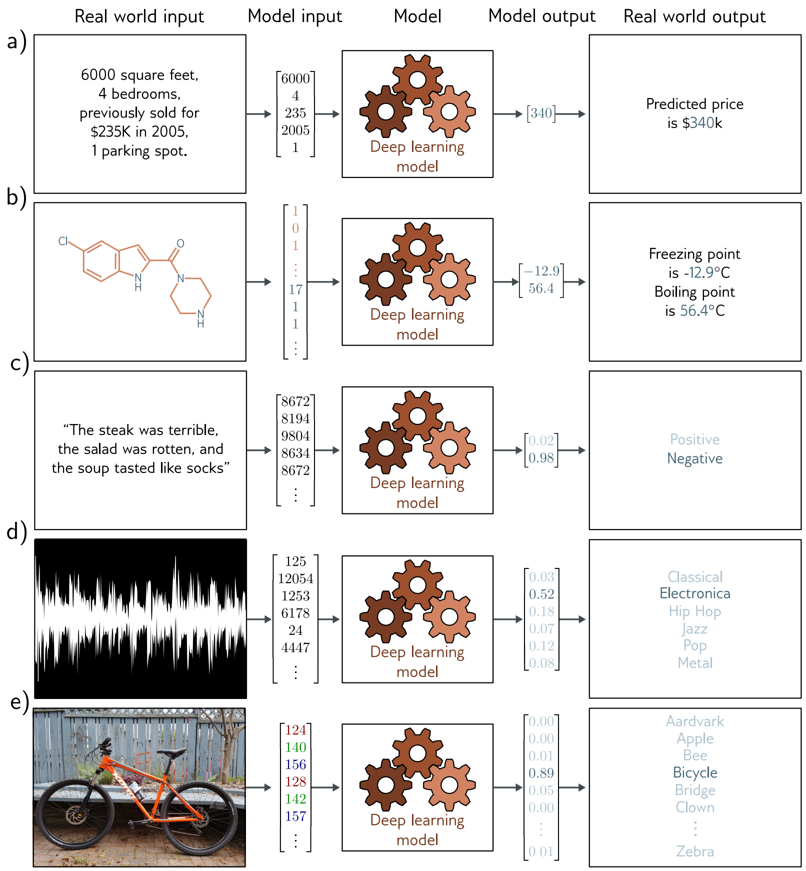
Image generation from text

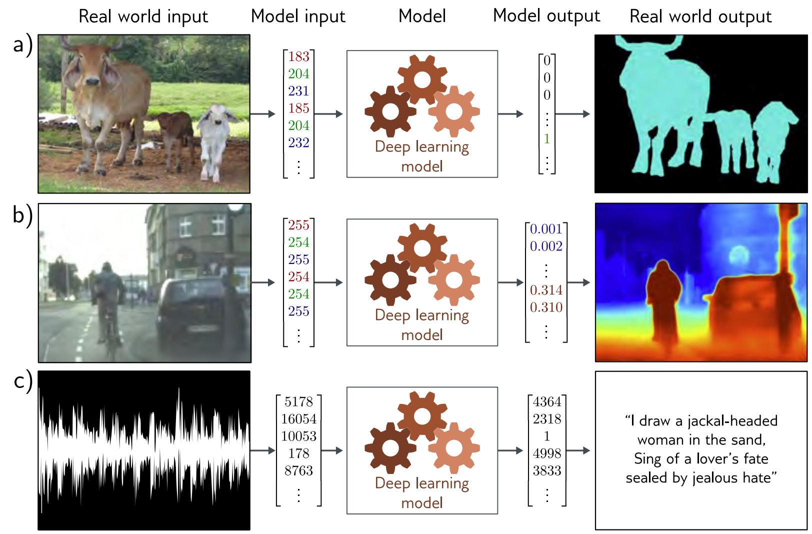
Real world input | Model input | Model | Model output | Real world output

"Teddy bears mixing sparkling chemicals as mad scientists, in a steampunk style."

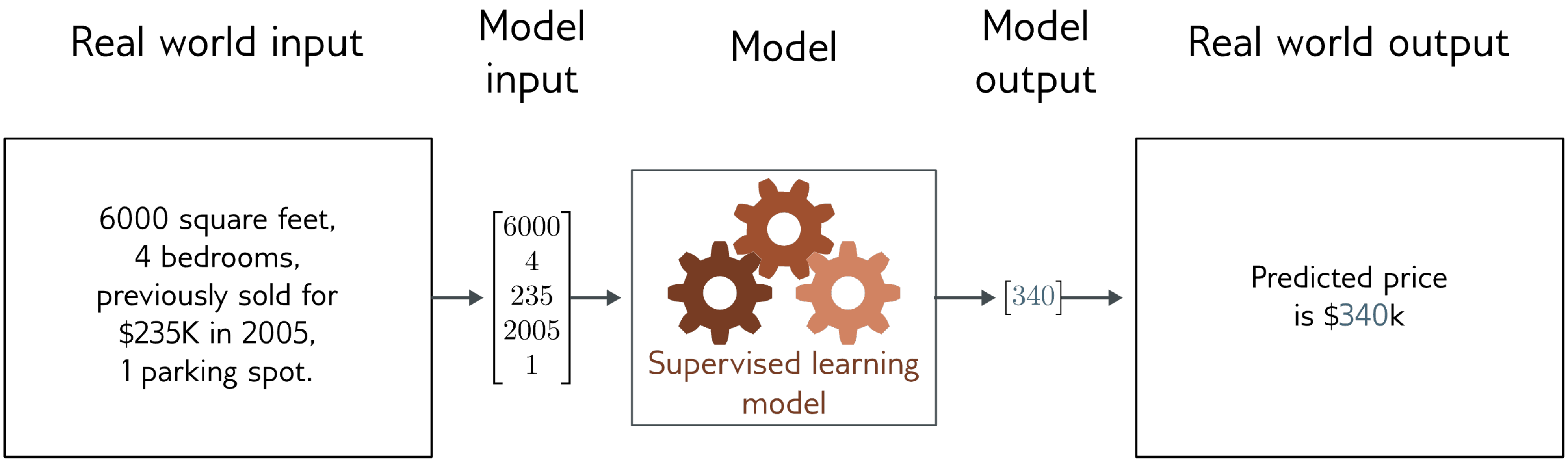
$$\begin{bmatrix} 8300 \\ 532 \\ 7676 \\ 7898 \\ 883 \\ \vdots \end{bmatrix} \rightarrow \text{Supervised learning model} \rightarrow \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 1 \\ \vdots \end{bmatrix} \rightarrow$$

Supervised Learning Classification and Regression Applications





Regression



- Univariate regression problem (one output, real value)

What are the model properties?

- Sentiment Analysis

- ❑ Univariate output
- ❑ Multivariate output

- ❑ Binary Class
- ❑ Multi-Class

- ❑ Classification
- ❑ Regression

What kind of model should we use?

- Image classification

❑ Univariate output
❑ Multivariate output

❑ Binary Class
❑ Multi-Class

❑ Classification
❑ Regression

What kind of model should we use?

- Image Semantic Segmentation

❑ Univariate output
❑ Multivariate output

❑ Binary Class
❑ Multi-Class

❑ Classification
❑ Regression

What kind of model should we use?

- Monocular Depth Estimation

❑ Univariate output
❑ Multivariate output

❑ Binary Class
❑ Multi-Class

❑ Classification
❑ Regression

What kind of model should we use?

- Next word prediction in a transformer language model

- ❑ Univariate output
- ❑ Multivariate output

- ❑ Binary Class
- ❑ Multi-Class

- ❑ Classification
- ❑ Regression

Supervised learning

- Examples
- Terminology
- Notation
  - Model
  - Loss function
  - Training
  - Testing
- 1D Linear regression example
  - Model
  - Loss function
  - Training
  - Testing

Supervised learning terminology

- 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

- Examples
- Terminology
- Notation
    - Model
    - Loss function
    - Training
    - Testing
- 1D Linear regression example
    - Model
    - Loss function
    - Training
    - Testing

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:

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

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:

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

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:

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

or for short:

$$L[\phi] \leftarrow$$

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:

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

Select all that are True

Supervised learning

- Examples
- Terminology
- Notation
    - Model
    - Loss function
    - Training
    - Testing
- 1D Linear regression example
    - Model
    - Loss function
    - Training
    - Testing

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:

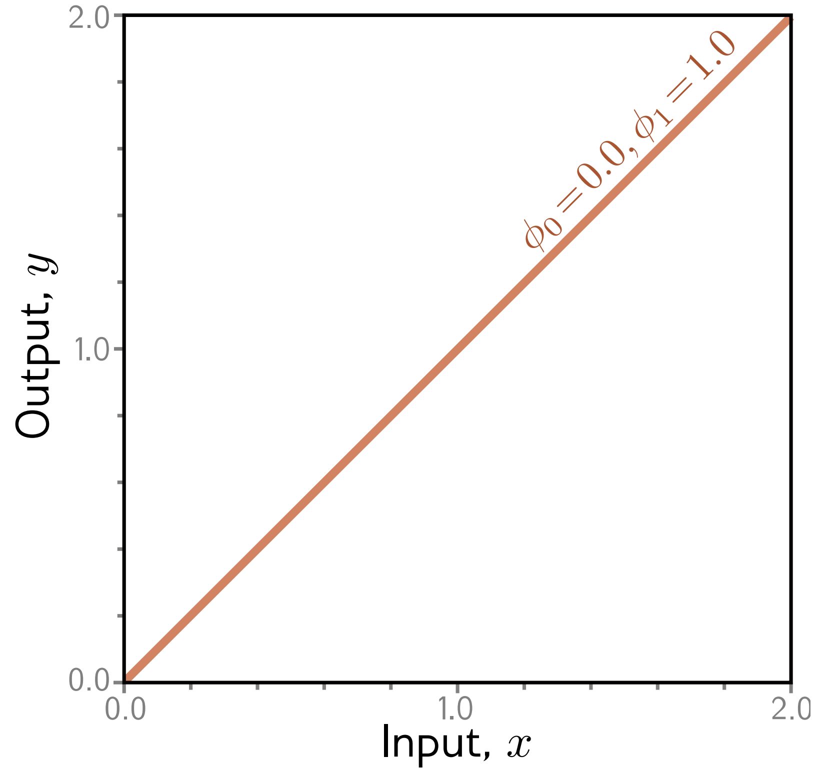
$$y = \mathrm{f}[x, \boldsymbol{\phi}]$$
$$= \phi_0 + \phi_1 x$$

- Parameters

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

y-offset

slope



Example: 1D Linear regression model

- Model:

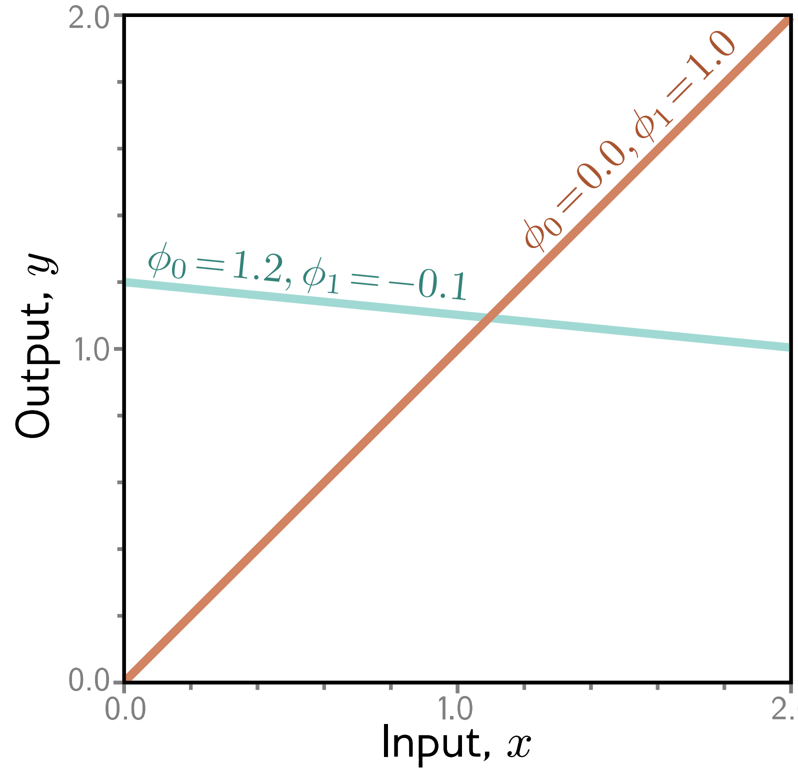
$$y = \mathrm{f}[x, \boldsymbol{\phi}]$$
$$= \phi_0 + \phi_1 x$$

- Parameters

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

y-offset

slope



Example: 1D Linear regression model

- Model:

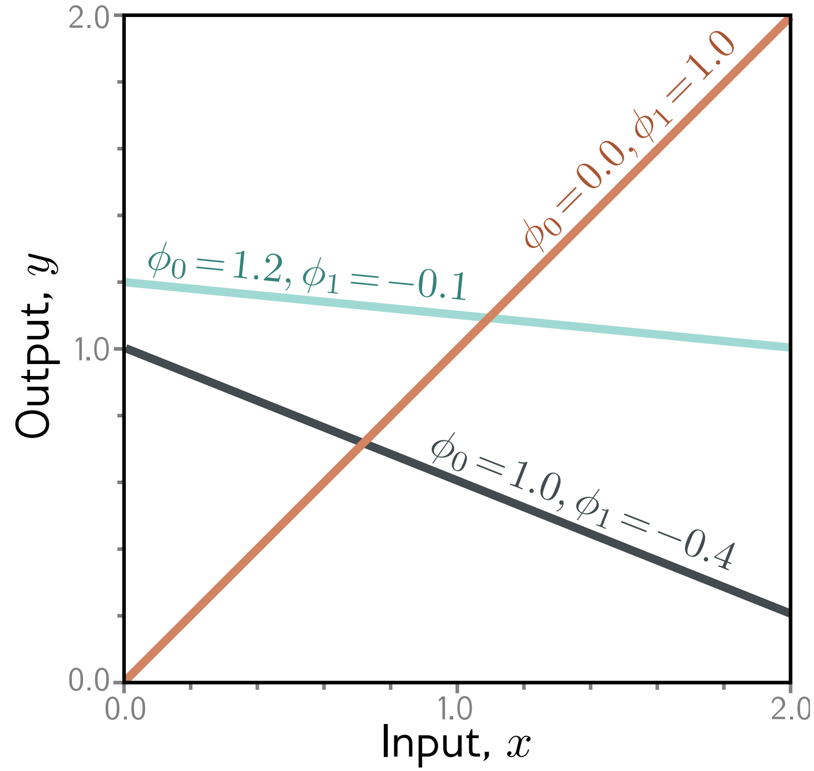
$$
\begin{aligned}
y &= \mathrm{f}[x, \boldsymbol{\phi}] \\
&= \phi_0 + \phi_1 x
\end{aligned}
$$

- Parameters

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

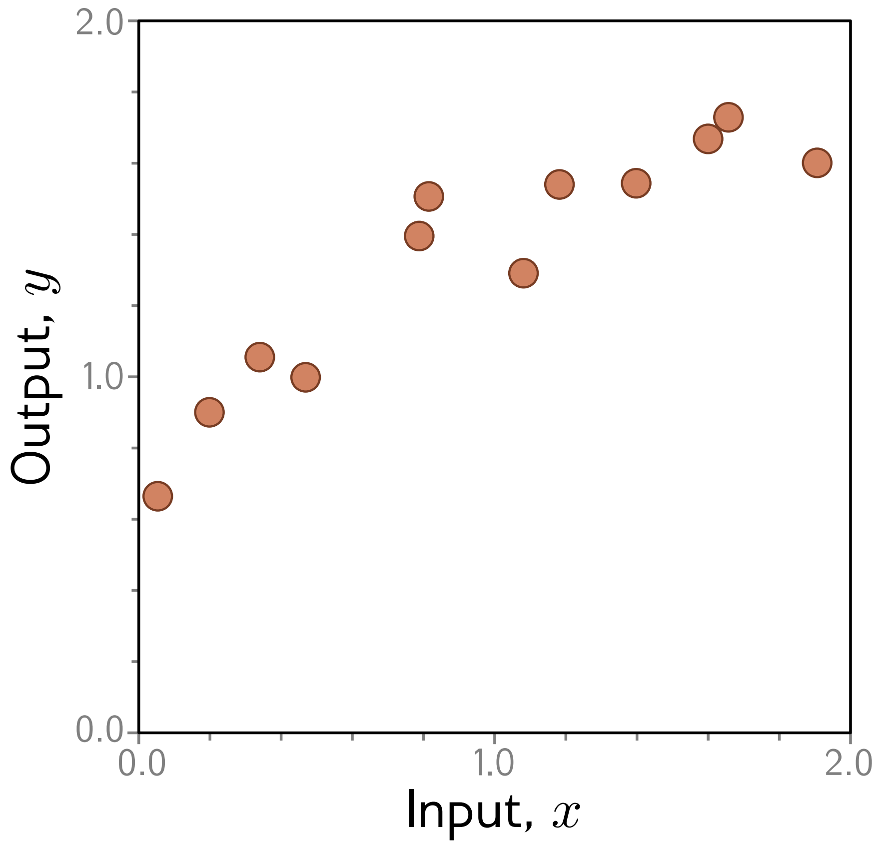
y-offset

slope

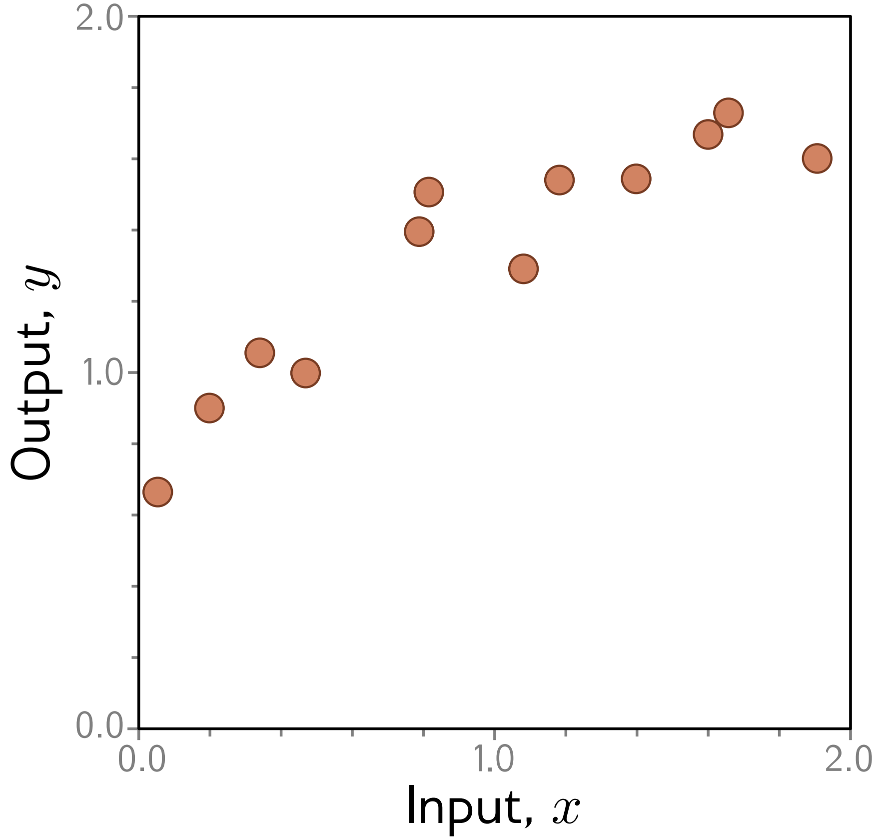


Interactive Figure 2.1

Example: 1D Linear regression training data



Example: 1D Linear regression training data

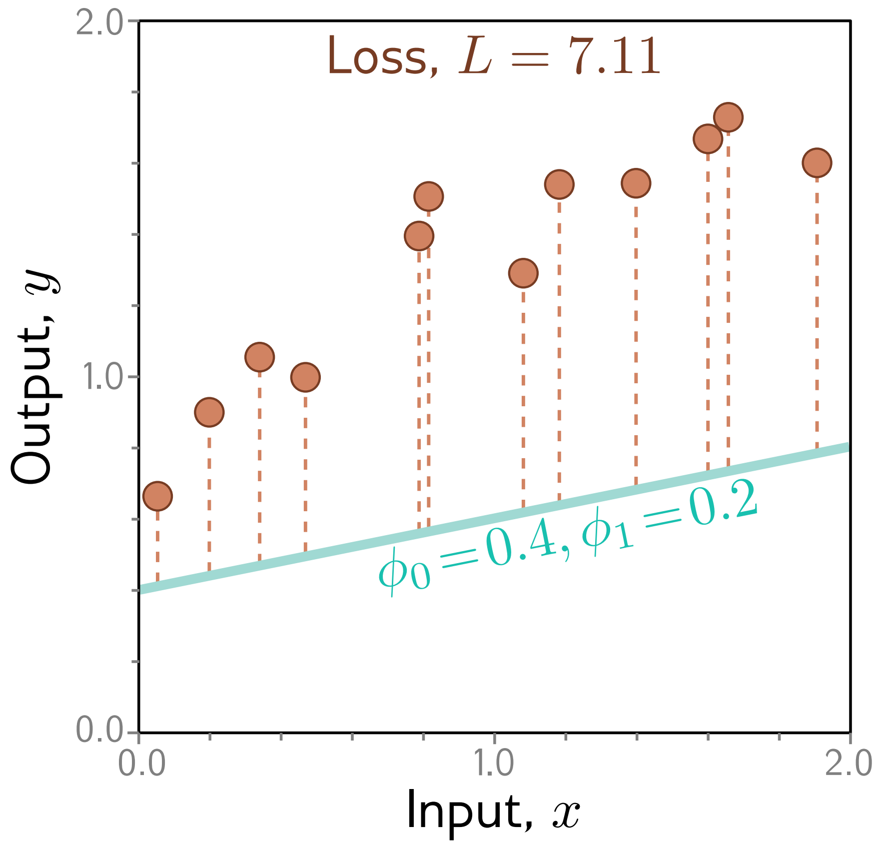


Loss function:

$$
\begin{aligned}
L[\boldsymbol{\phi}] &= \sum_{i=1}^{I} \left(\mathrm{f}[x_i, \boldsymbol{\phi}] - y_i\right)^2 \\
&= \sum_{i=1}^{I} \left(\phi_0 + \phi_1 x_i - y_i\right)^2
\end{aligned}
$$

"Least squares loss function"

Example: 1D Linear regression loss function



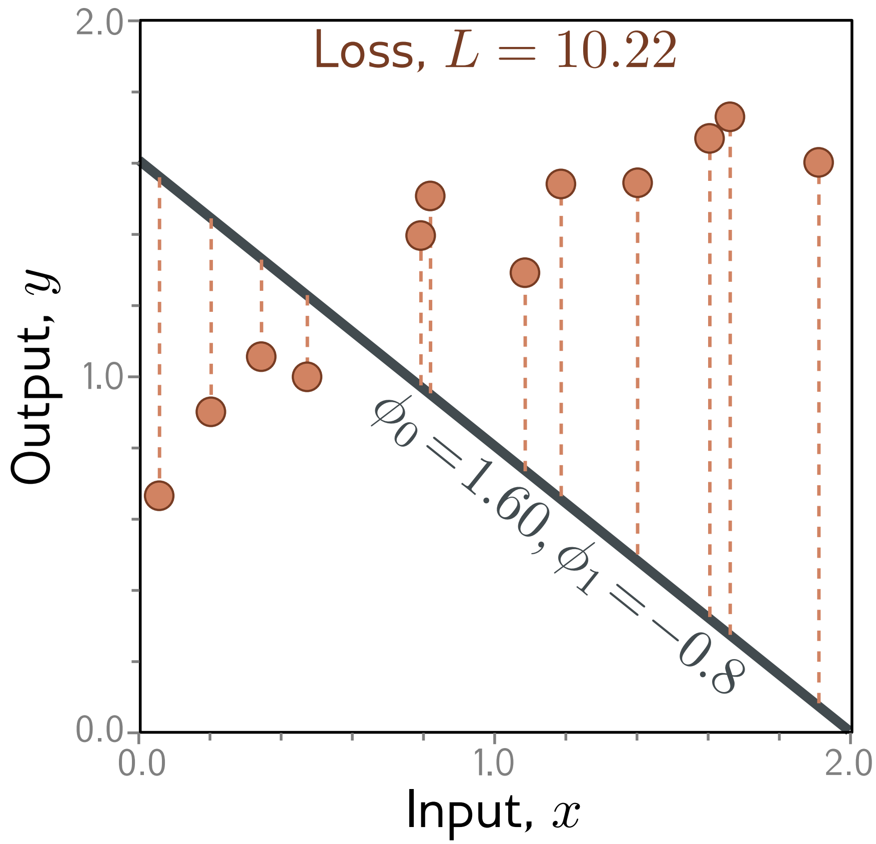
Loss function:

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

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

"Least squares loss function"

Example: 1D Linear regression loss function



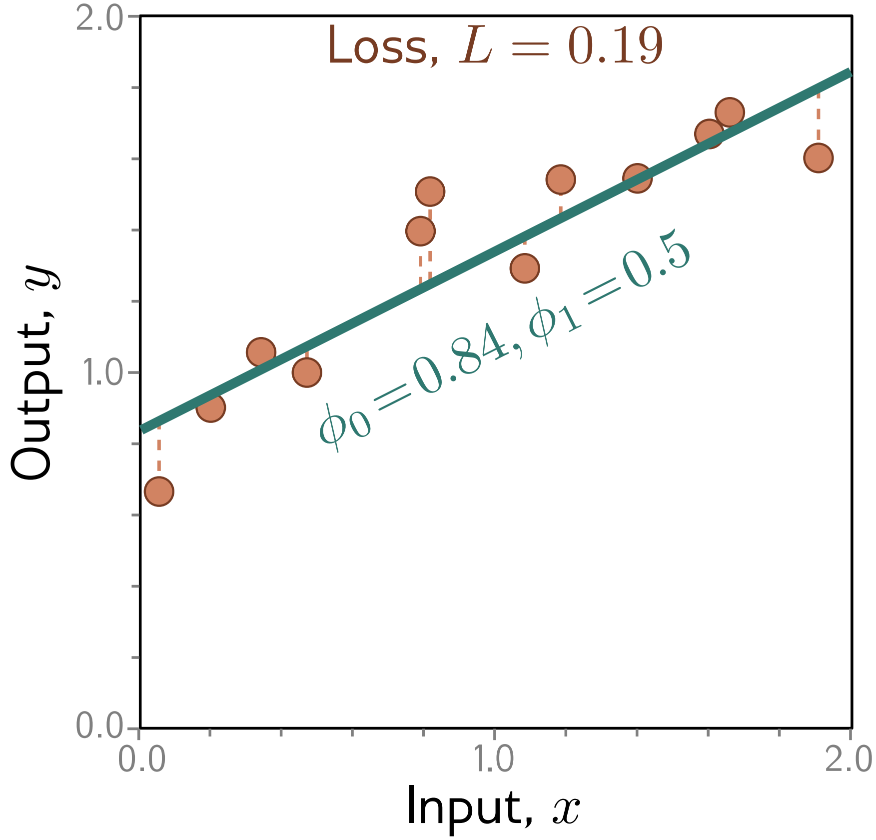
Loss function:

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

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

"Least squares loss function"

Example: 1D Linear regression loss function



Loss function:

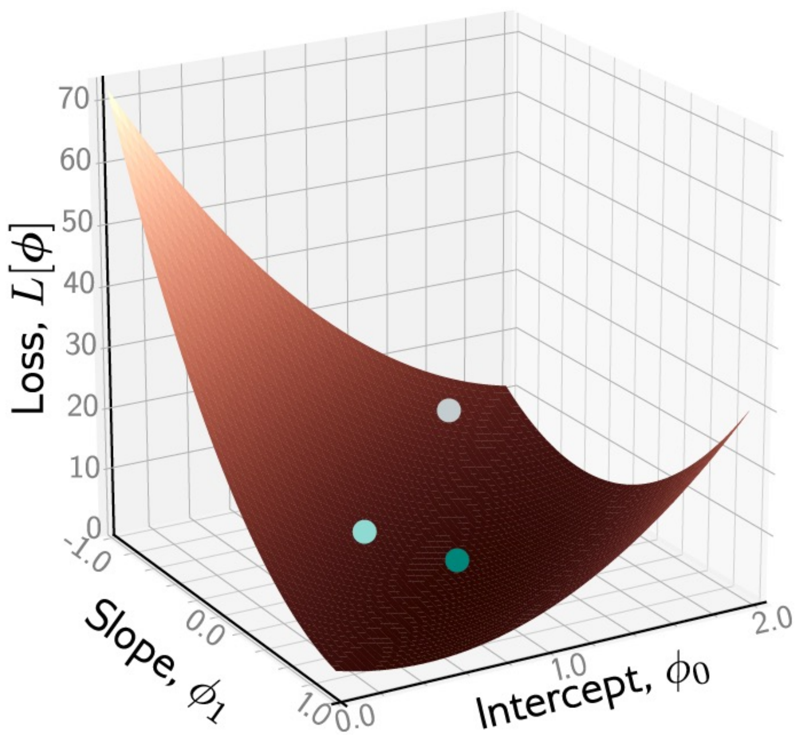
$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \boldsymbol{\phi}] - y_i)^2$$

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

"Least squares loss function"

Interactive Figure 2.2

Example: 1D Linear regression loss function



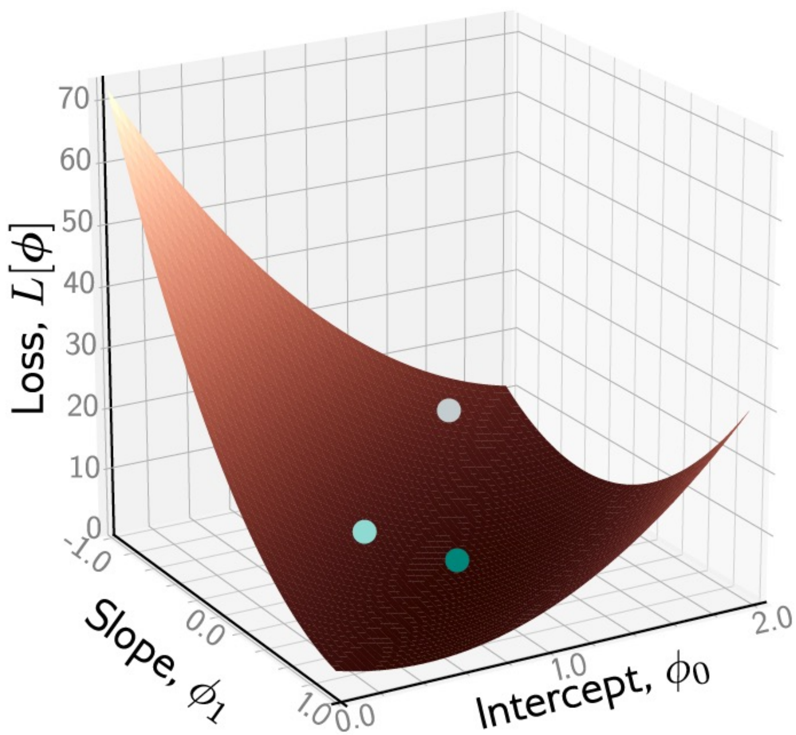
Loss function:

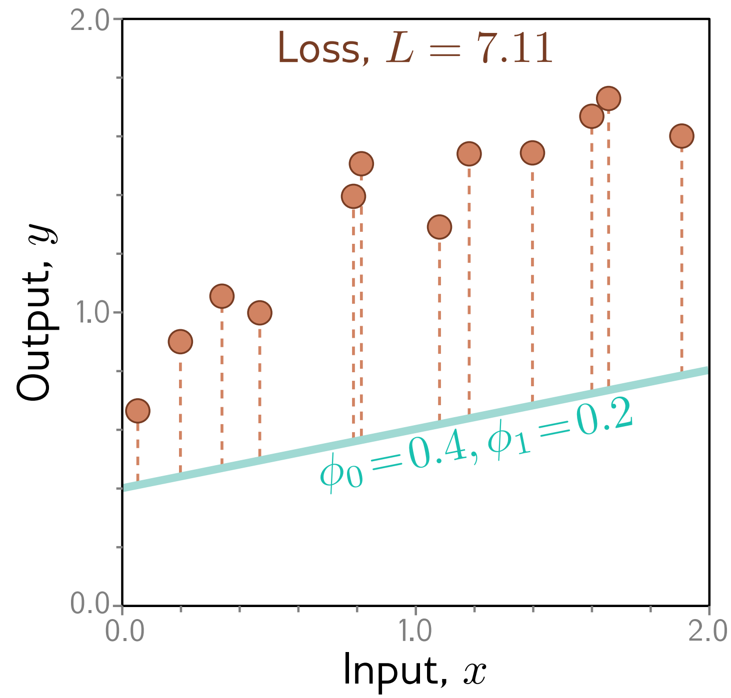
$$L[\phi] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \phi] - y_i)^2$$

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

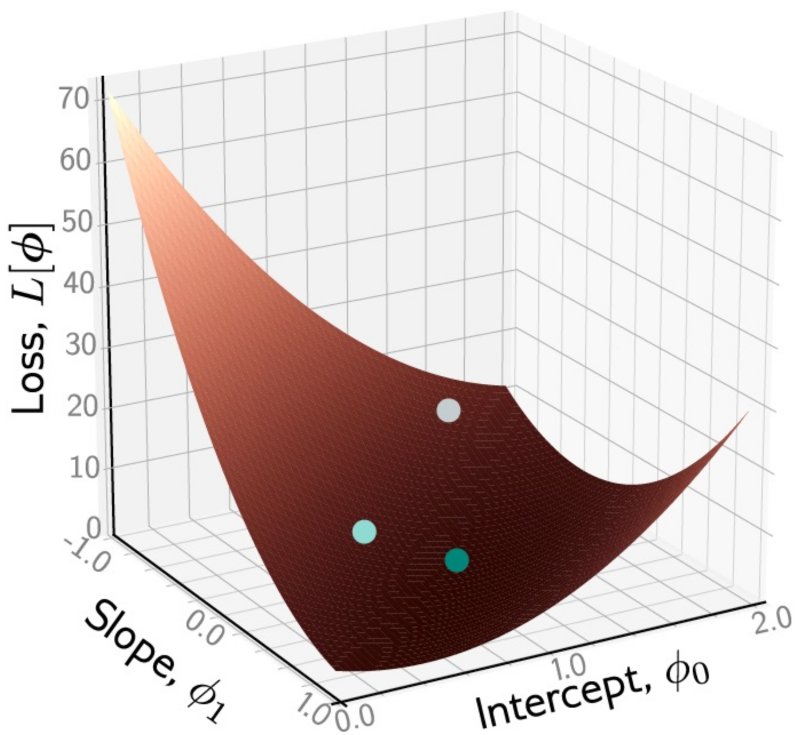
"Least squares loss function"

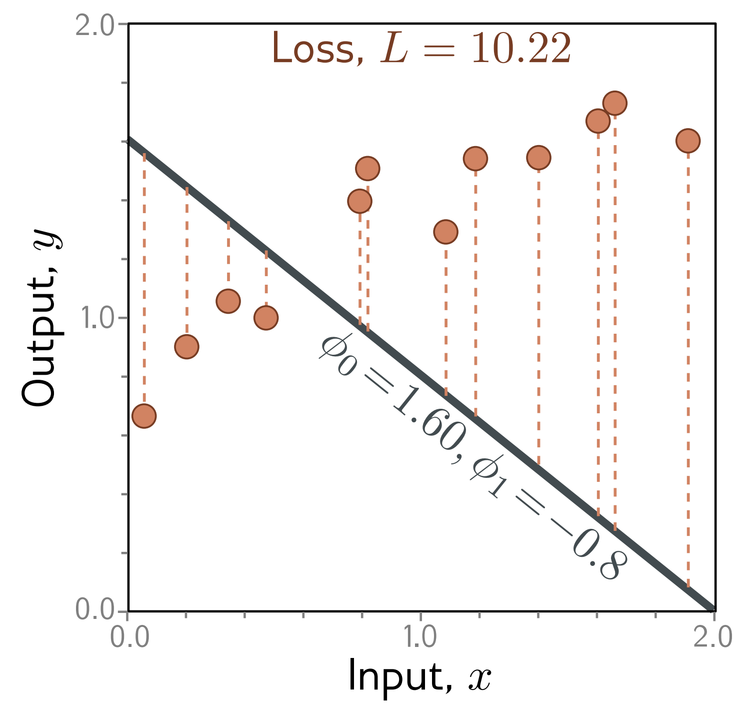
Example: 1D Linear regression loss function



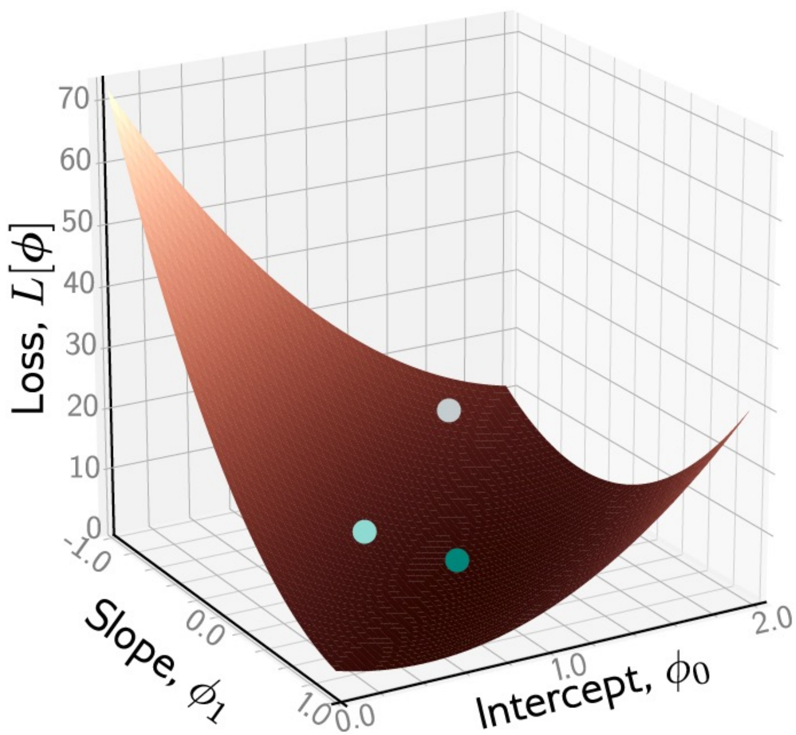


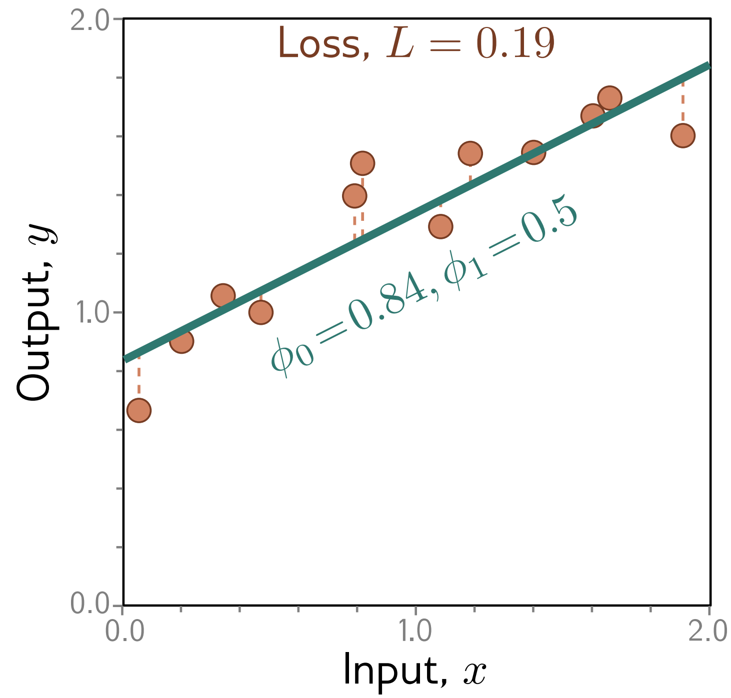
Example: 1D Linear regression loss function



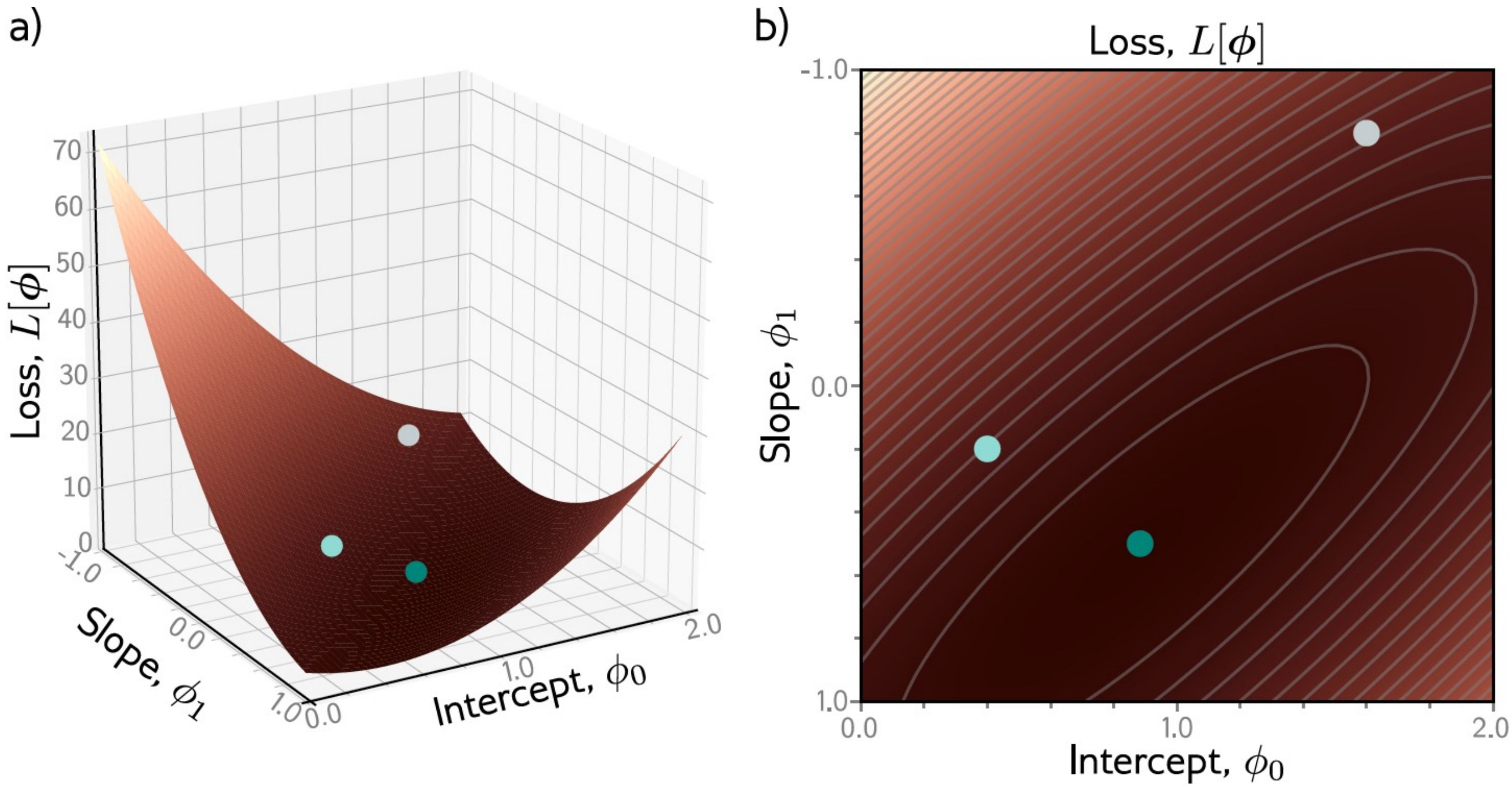


Example: 1D Linear regression loss function

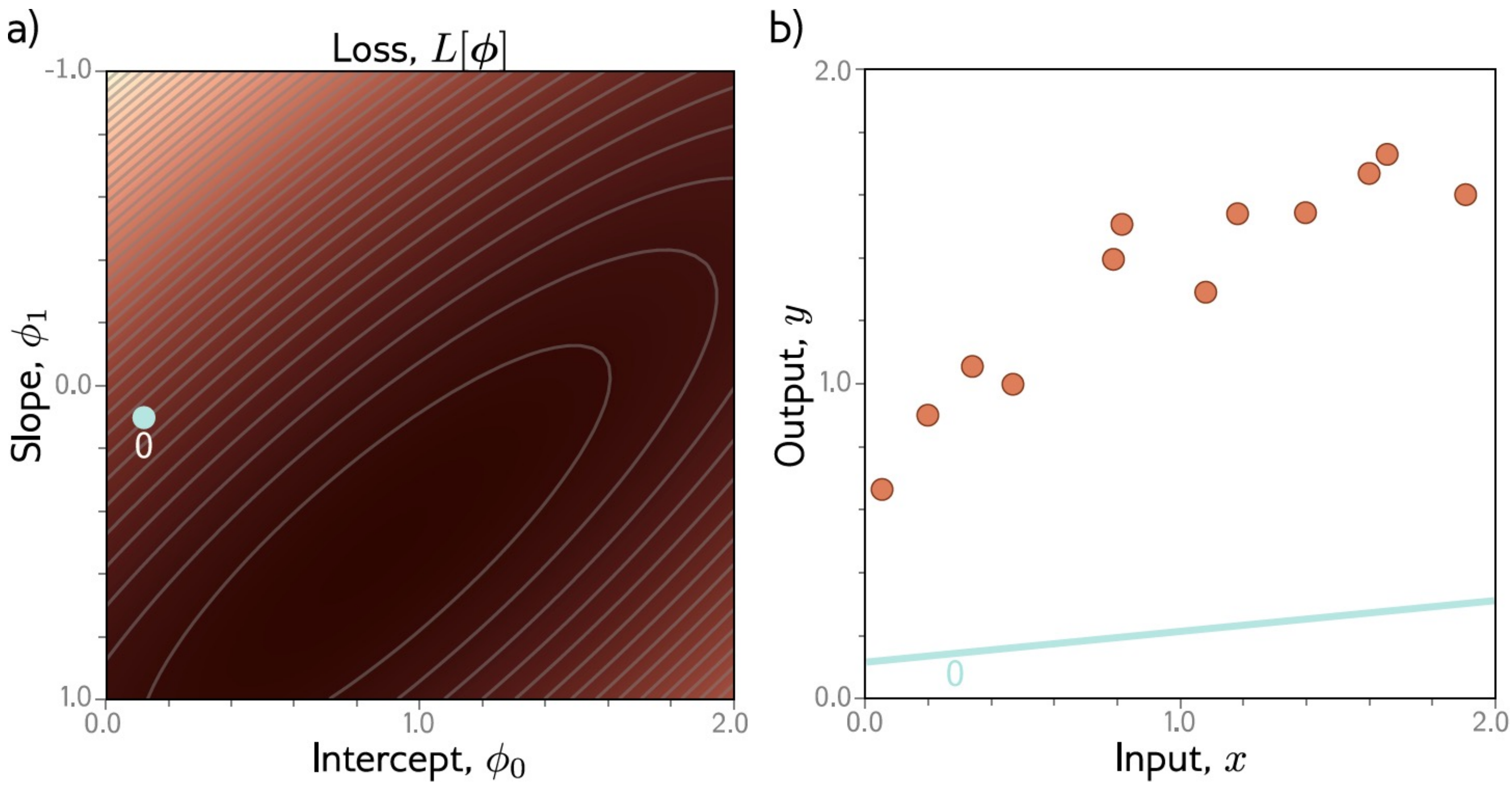




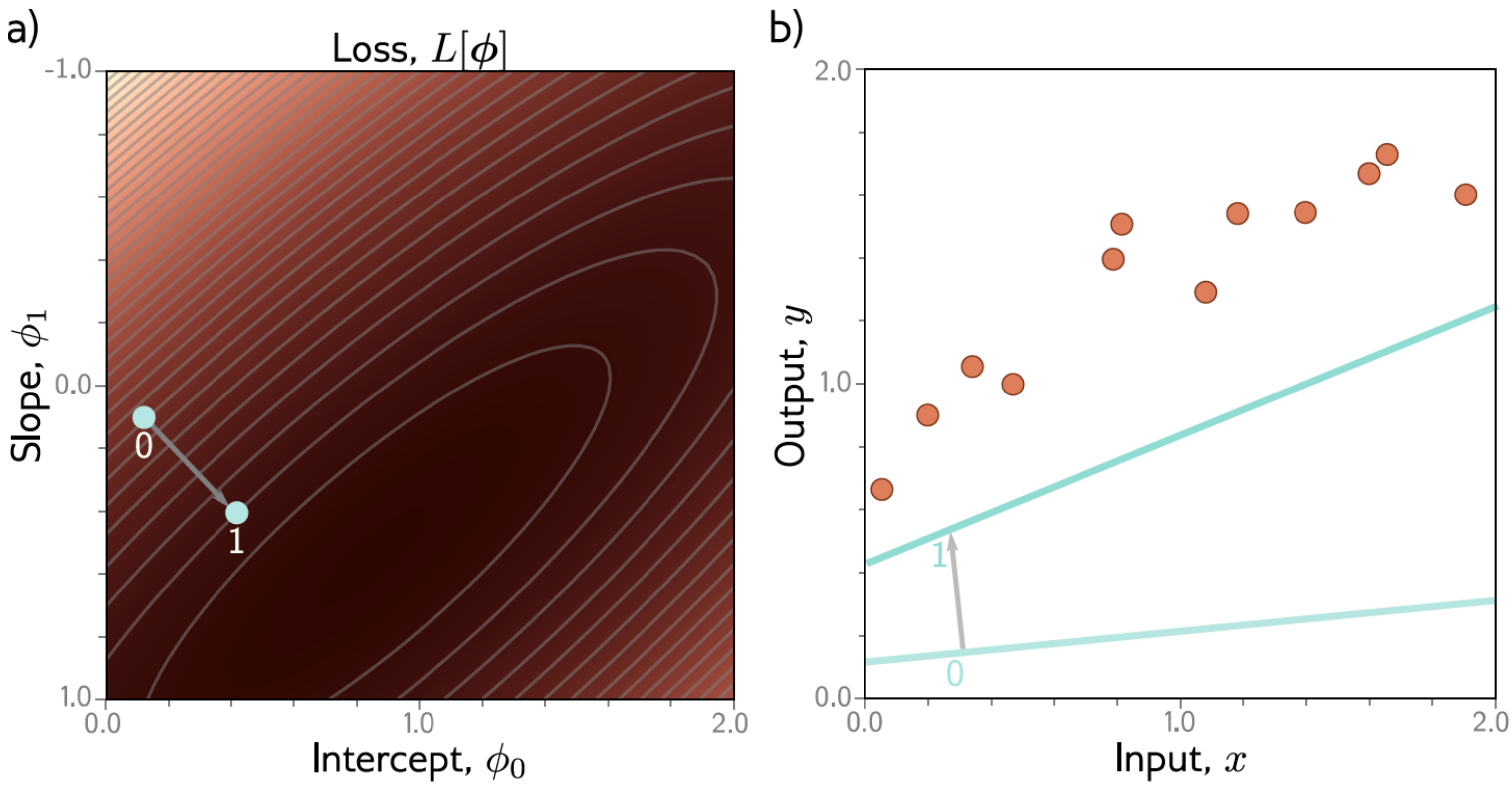
Example: 1D Linear regression loss function



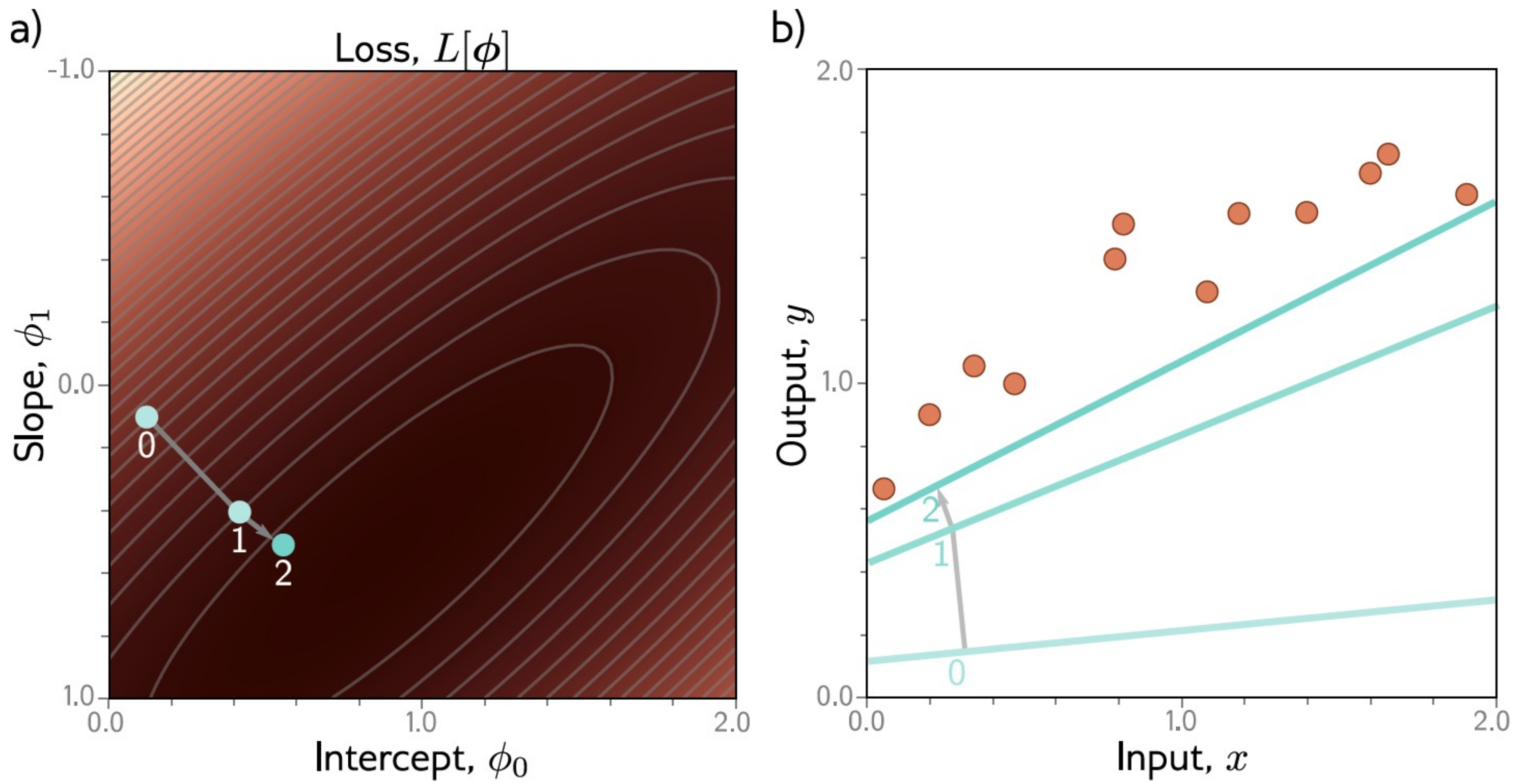
Example: 1D Linear regression training



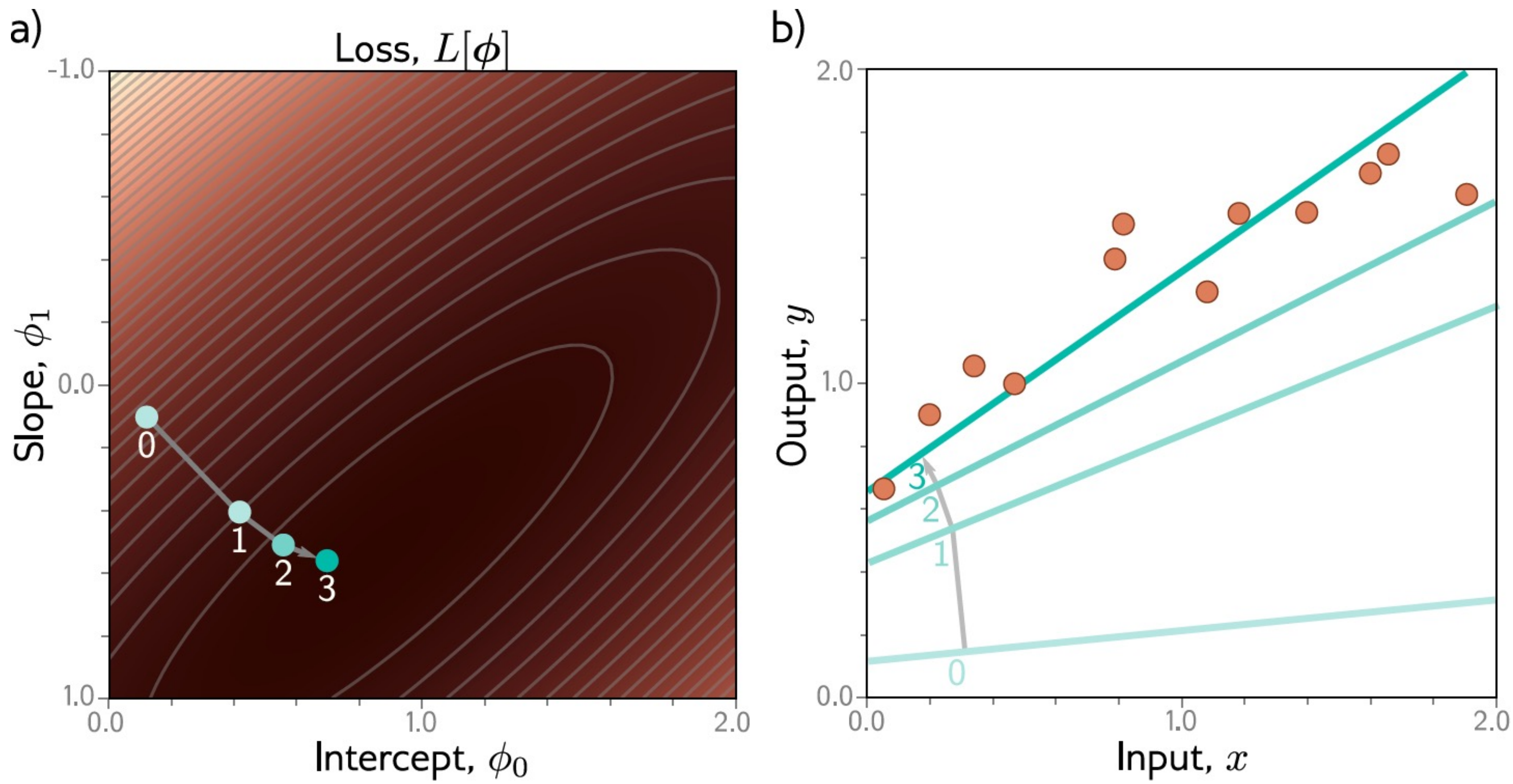
Example: 1D Linear regression training



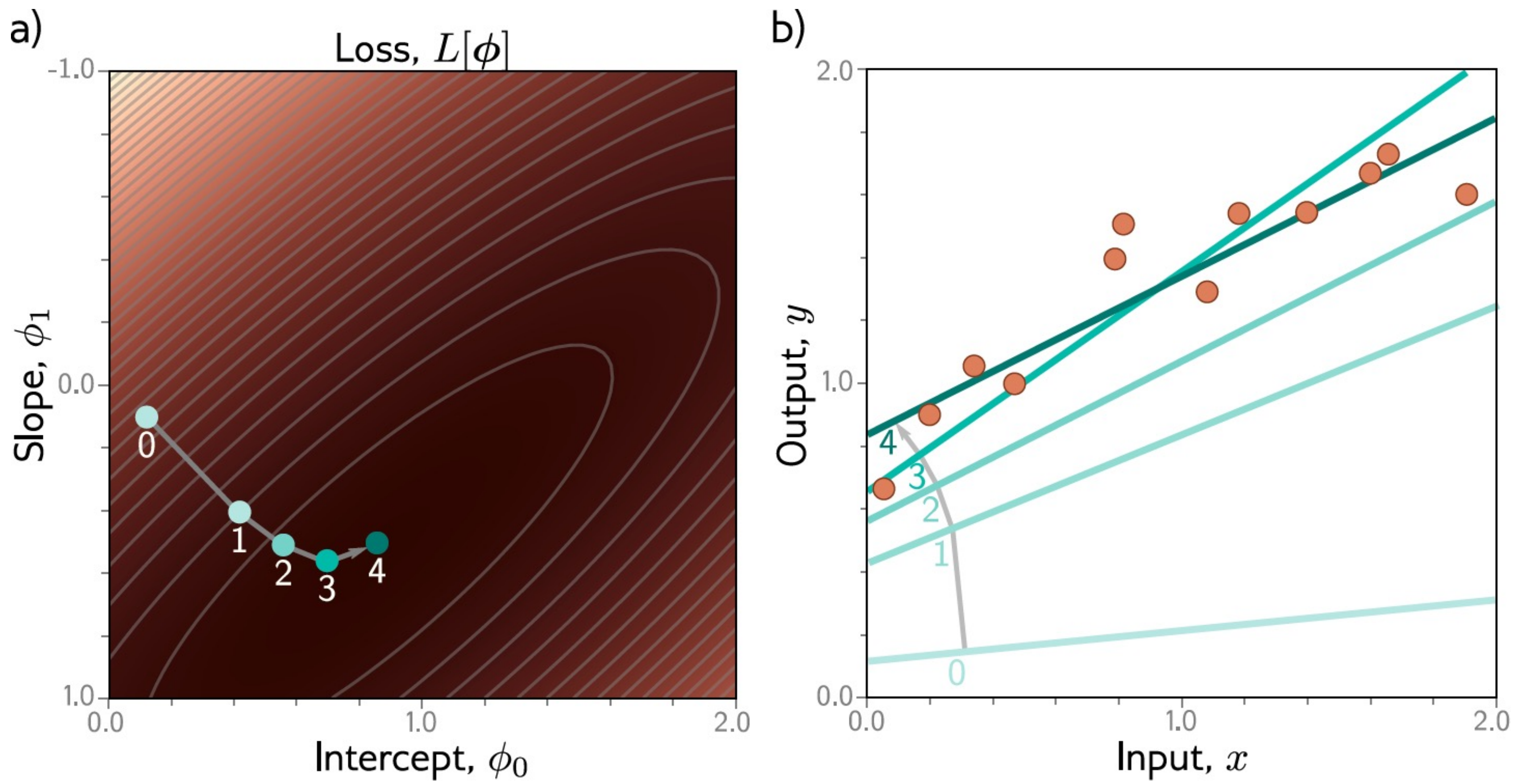
Example: 1D Linear regression training



Example: 1D Linear regression training



Example: 1D Linear regression training

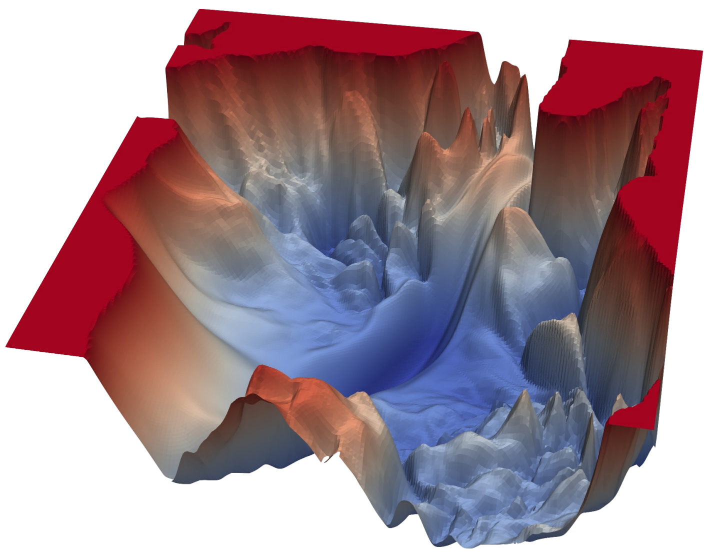


This technique is known as gradient descent

Interactive Figure 2.3

Possible objections

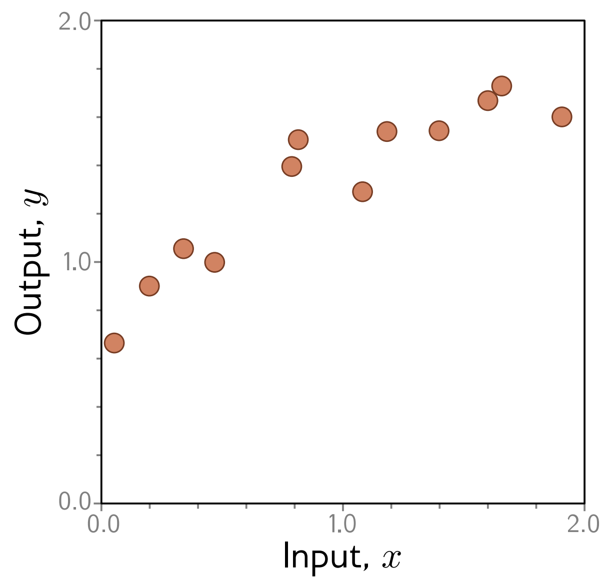
- But you can fit the line model in closed form!
  - Yes – but we won't be able to do this for more complex models
- But we could exhaustively try every slope and intercept combo!
  - Yes – but we won't be able to do this when there are a million parameters



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



Any Questions?

Next Lecture

- How do we choose a loss function in a principled way?

Lecture Feedback

https://forms.gle/pXHM5nx1Ti9aFmpw6