



Please Sit With Your Group

L-3

Supervised Learning II

MAIB7002
Fundamentals of AI Models and Technologies

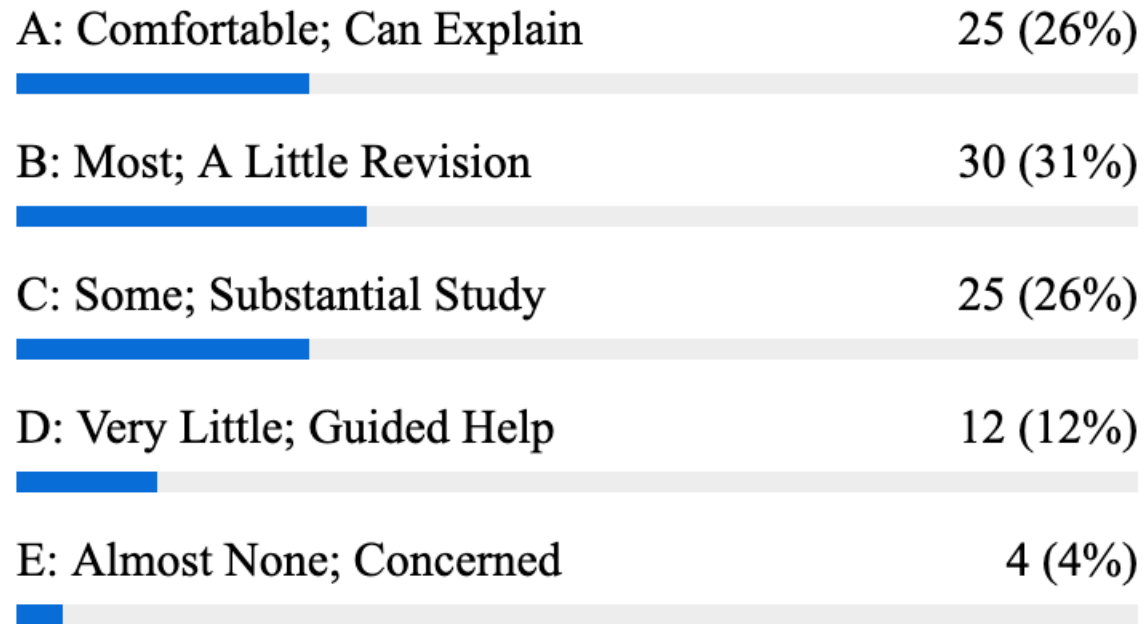
L-2

96 responses to each question · Self-reported experience

L-2

96 responses to each question · Self-reported experience

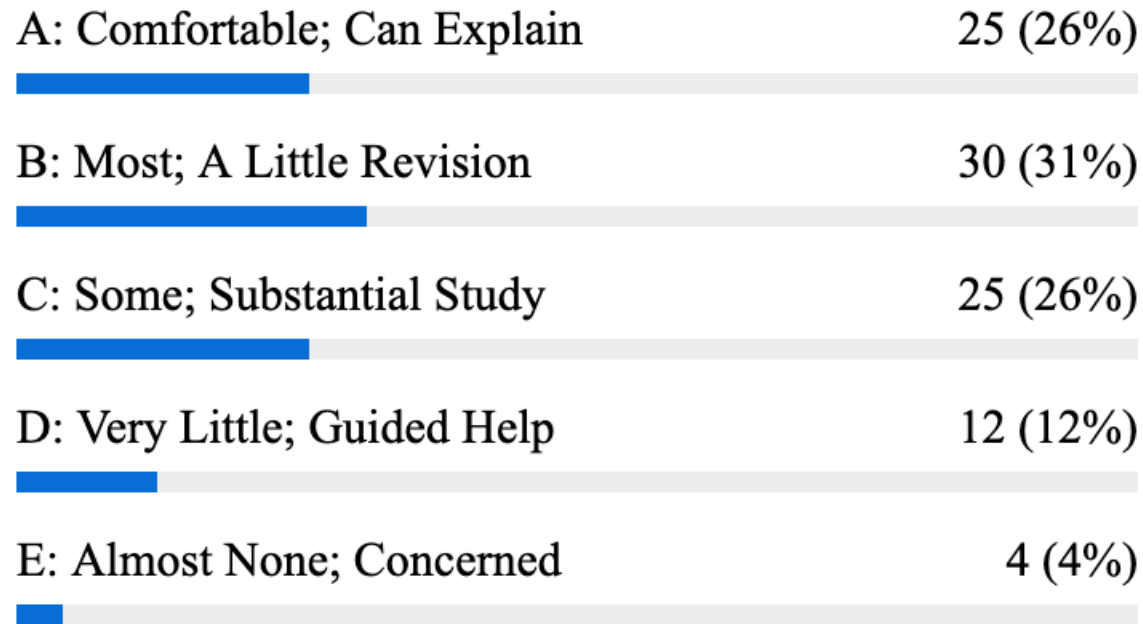
Mathematics



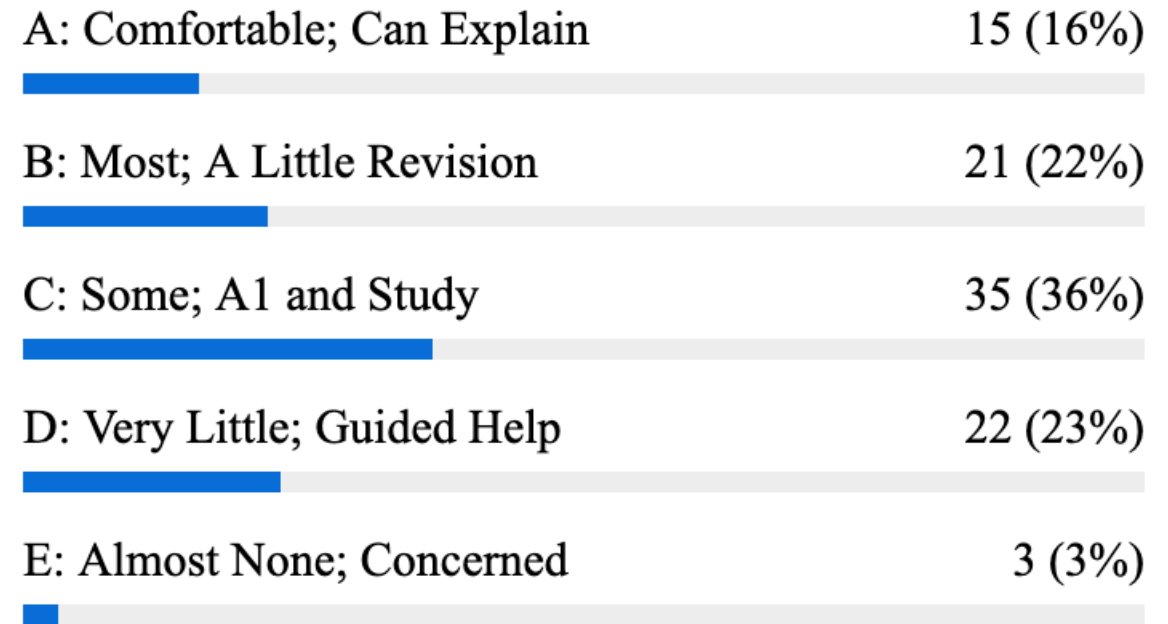
L-2

96 responses to each question · Self-reported experience

Mathematics



Programming



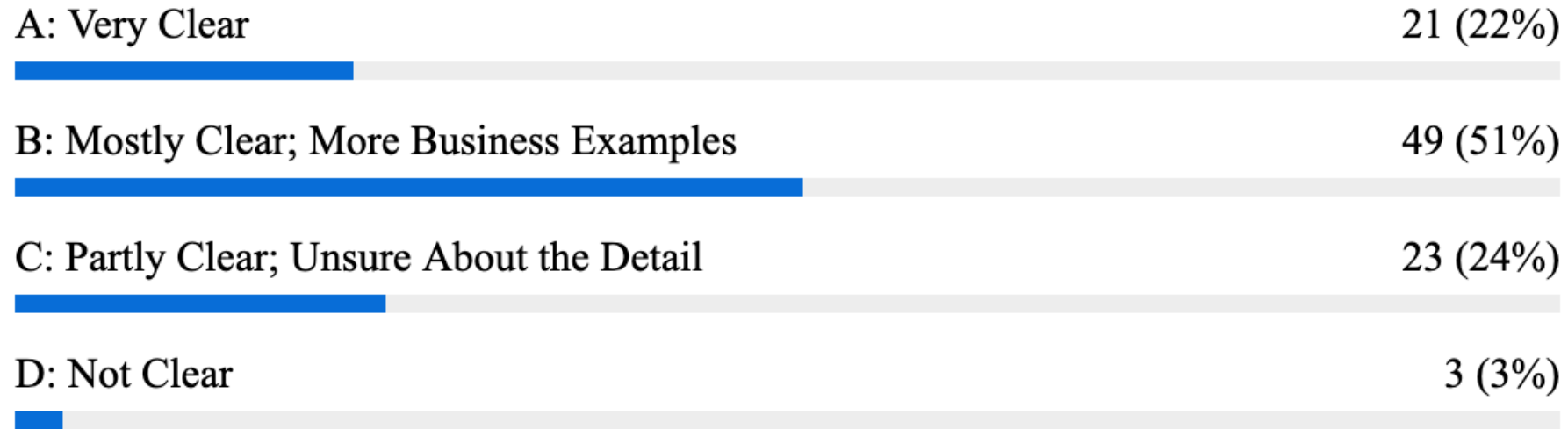
L-2

How clear is the purpose of learning the technical details?

96 responses

L-2

How clear is the purpose of learning the technical details?



96 responses

L-2

L-2

“

I understood the introductory part on ML roadmap, regression, classification examples, but I started becoming really confused about the mathematics parts and how it connects to the big picture of machine learning.

L-2

“

I understood the introductory part on ML roadmap, regression, classification examples, but I started becoming really confused about the mathematics parts and how it connects to the big picture of machine learning.

“

More demos, especially the code stuff

L-2

“

I understood the introductory part on ML roadmap, regression, classification examples, but I started becoming really confused about the mathematics parts and how it connects to the big picture of machine learning.

“

More demos, especially the code stuff

“

It may be better to leave the class more time on trying out the coding ourselves.

L-2

“

I understood the introductory part on ML roadmap, regression, classification examples, but I started becoming really confused about the mathematics parts and how it connects to the big picture of machine learning.

“

Can you upload the slides earlier so that I can preview it before class, thx!

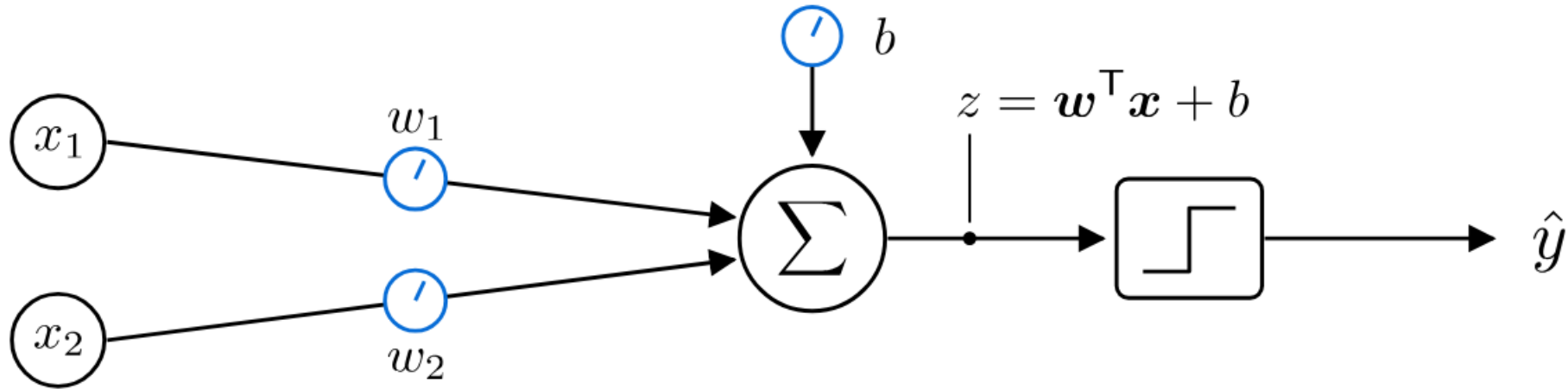
“

More demos, especially the code stuff

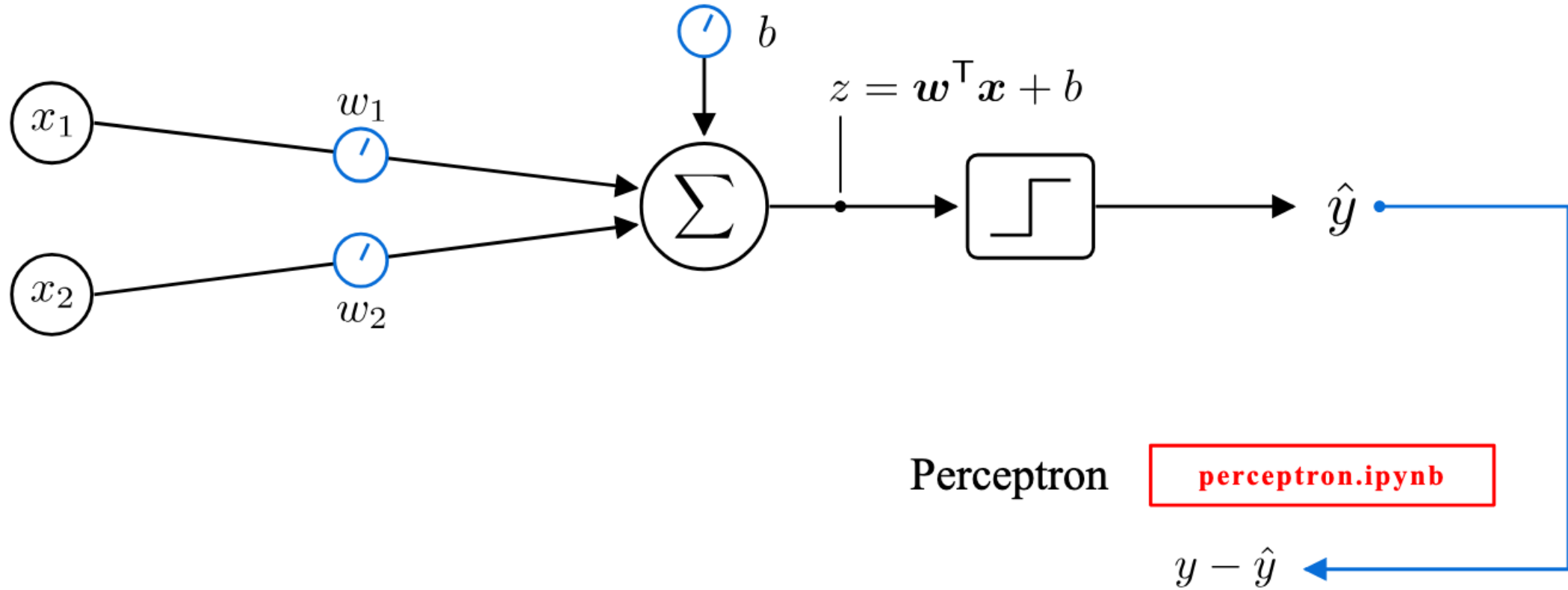
“

It may be better to leave the class more time on trying out the coding ourselves.

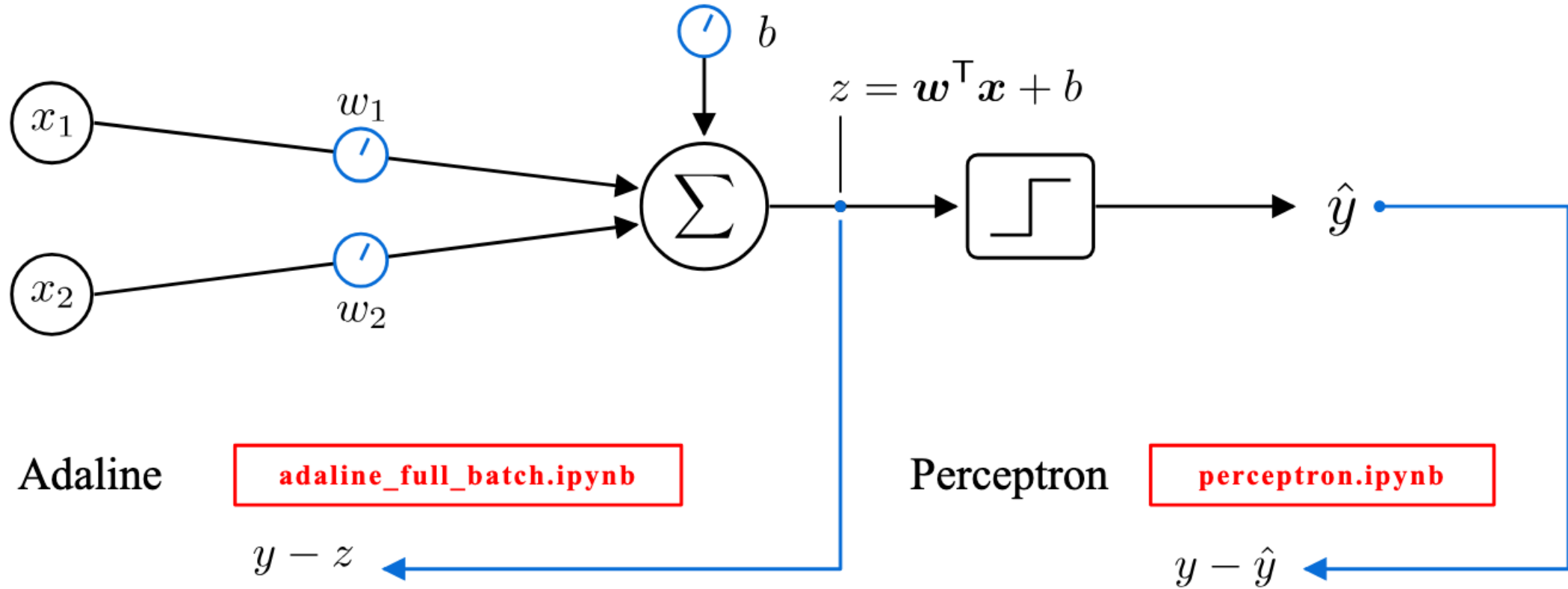
L-2: Two Classifiers from Scratch



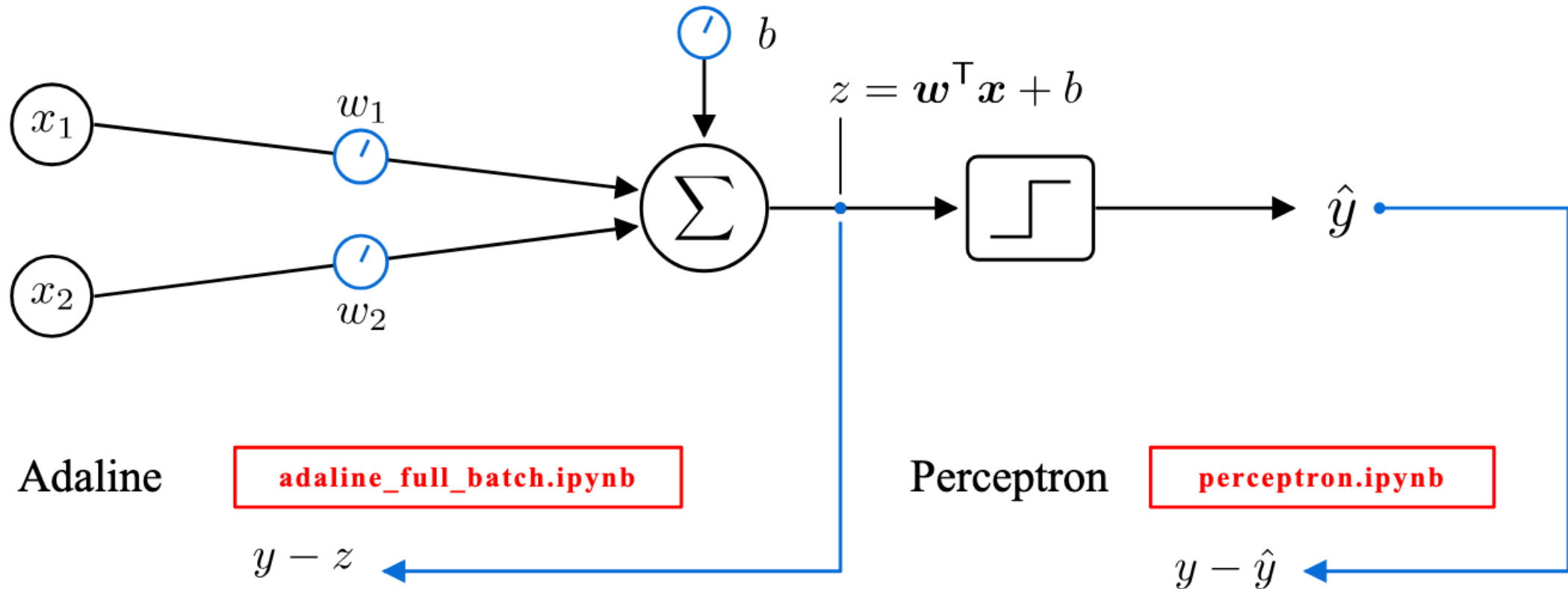
L-2: Two Classifiers from Scratch



L-2: Two Classifiers from Scratch

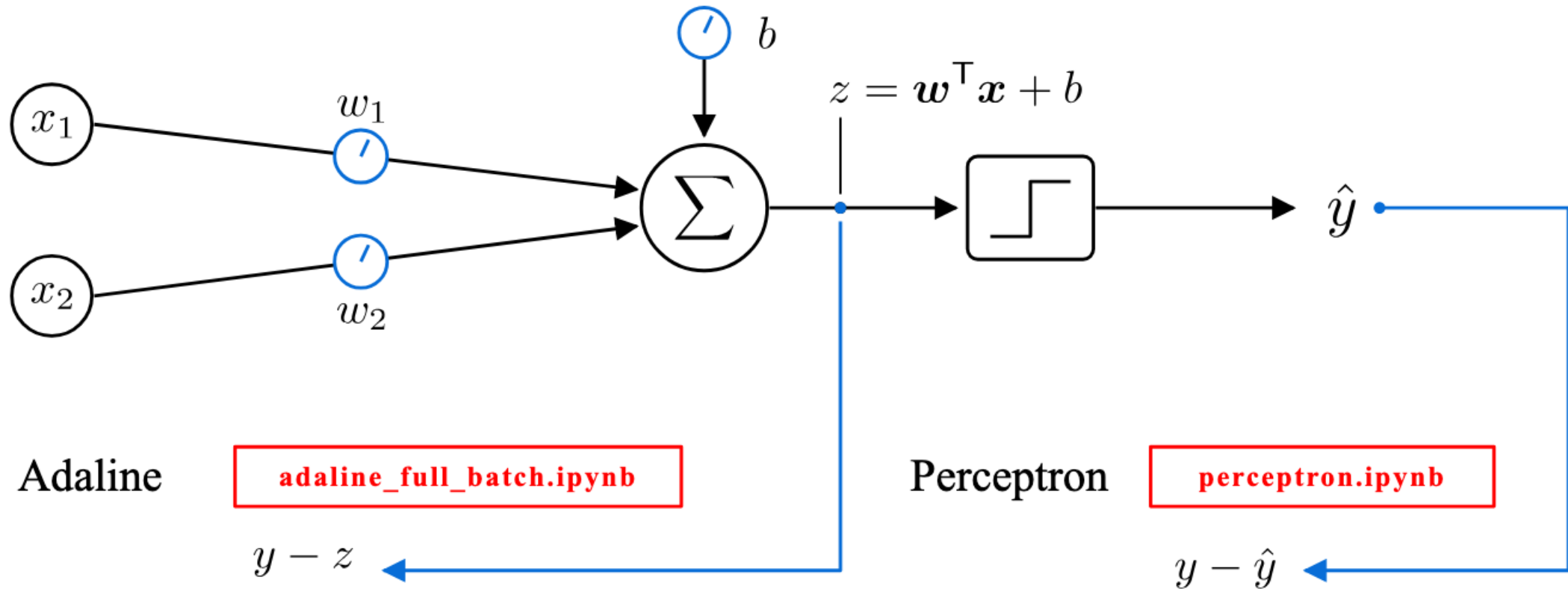


L-2: Two Classifiers from Scratch



NumPy: Calculations · pandas: Data · Matplotlib: Plots

L-2: Two Classifiers from Scratch



NumPy: Calculations · pandas: Data · Matplotlib: Plots

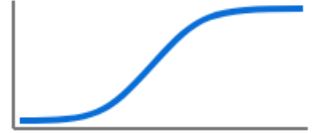
No sklearn

Supervised Learning: L3 and L4

Supervised Learning: L3 and L4

Logistic Regression

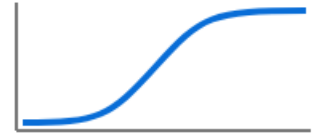
Estimate probabilities from a linear score



Supervised Learning: L3 and L4

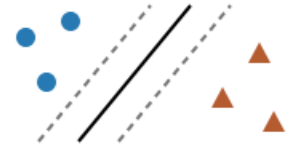
Logistic Regression

Estimate probabilities from a linear score



SVM

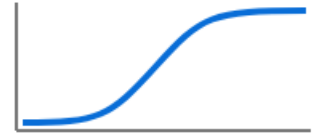
Separate classes with a margin



Supervised Learning: L3 and L4

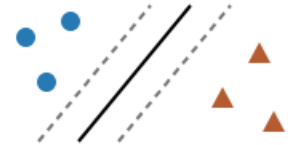
Logistic Regression

Estimate probabilities from a linear score



SVM

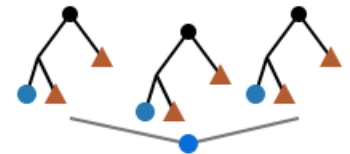
Separate classes with a margin



Random Forest

L4: Decision Trees and
Forests

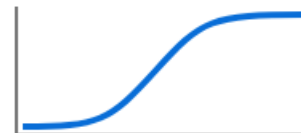
Learn questions, then combine many trees



Supervised Learning: L3 and L4

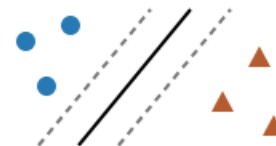
Logistic Regression

Estimate probabilities from a linear score



SVM

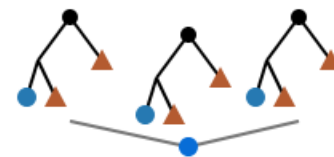
Separate classes with a margin



Random Forest

L4: Decision Trees and
Forests

Learn questions, then combine many trees



kNN

L4

Use the labels of nearby examples



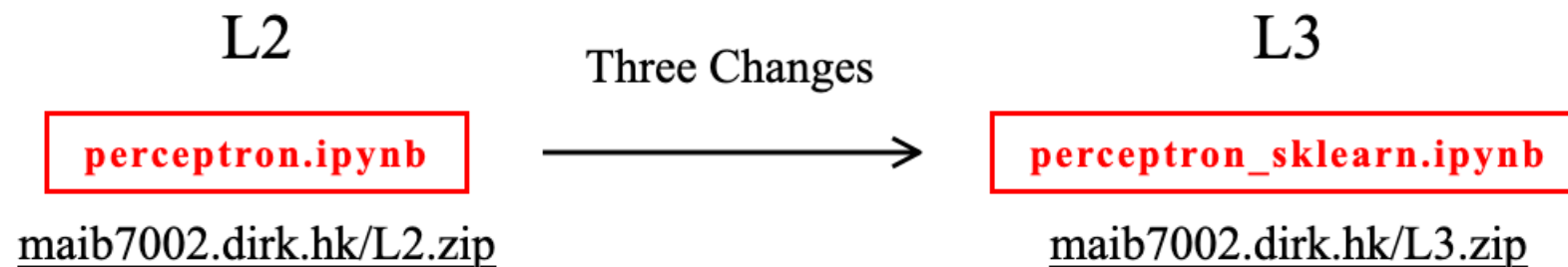
But First, a Familiar Model

L2

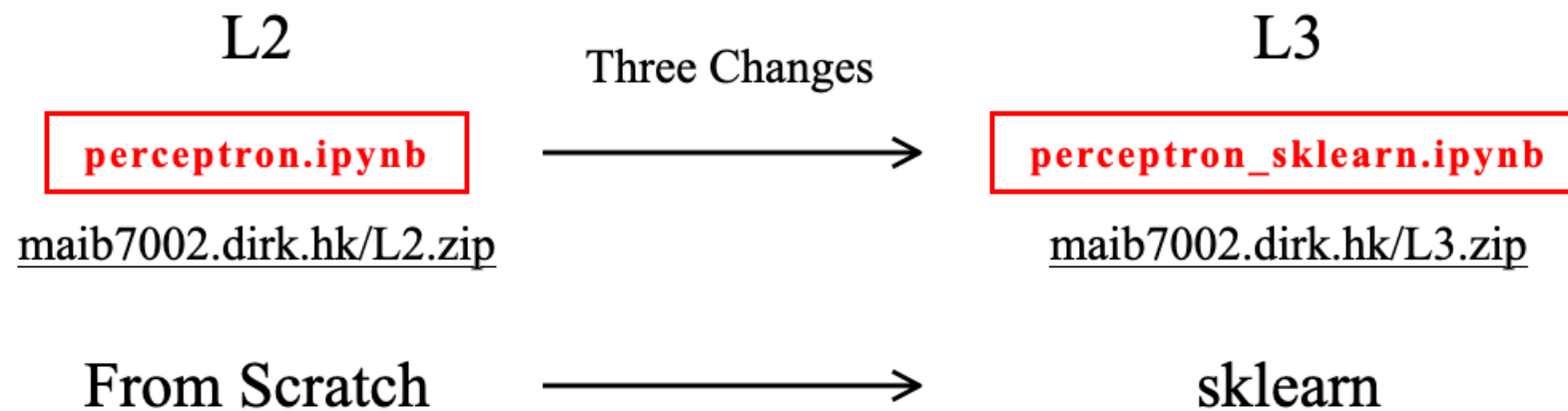
perceptron.ipynb

maib7002.dirk.hk/L2.zip

But First, a Familiar Model



But First, a Familiar Model



1. Implementation

But First, a Familiar Model

L2

perceptron.ipynb

maib7002.dirk.hk/L2.zip

Three Changes



L3

perceptron_sklearn.ipynb

maib7002.dirk.hk/L3.zip

1. Implementation

From Scratch



sklearn

2. Classes

Two



Three

But First, a Familiar Model

L2

perceptron.ipynb

maib7002.dirk.hk/L2.zip

Three Changes



L3

perceptron_sklern.ipynb

maib7002.dirk.hk/L3.zip

1. Implementation

From Scratch



sklearn

2. Classes

Two



Three

3. Tested On

Training Data



Unseen Data

But First, a Familiar Model

L2

perceptron.ipynb

maib7002.dirk.hk/L2.zip

Three Changes



L3

perceptron_sklearn.ipynb

maib7002.dirk.hk/L3.zip

1. Implementation

From Scratch



sklearn

2. Classes

Two



Three

3. Tested On

Training Data



Unseen Data

Same Perceptron

1. Implementation: sklearn

L2: We Implement It

Implementation · Class Excerpt

```
error = target - self.predict(xi)
update = self.eta * error
self.w_ += update * xi
self.b_ += update
```

Calling

```
ppn = Perceptron(eta=0.1, n_iter=10)
ppn.fit(X, y)
pred = ppn.predict(X)
```

1. Implementation: sklearn

L2: We Implement It

Implementation · Class Excerpt

```
error = target - self.predict(xi)
update = self.eta * error
self.w_ += update * xi
self.b_ += update
```

Calling

```
ppn = Perceptron(eta=0.1, n_iter=10)
ppn.fit(X, y)
pred = ppn.predict(X)
```

L3: The Library Implements It

Implementation · Library Import

```
import sklearn.linear_model as lm
```

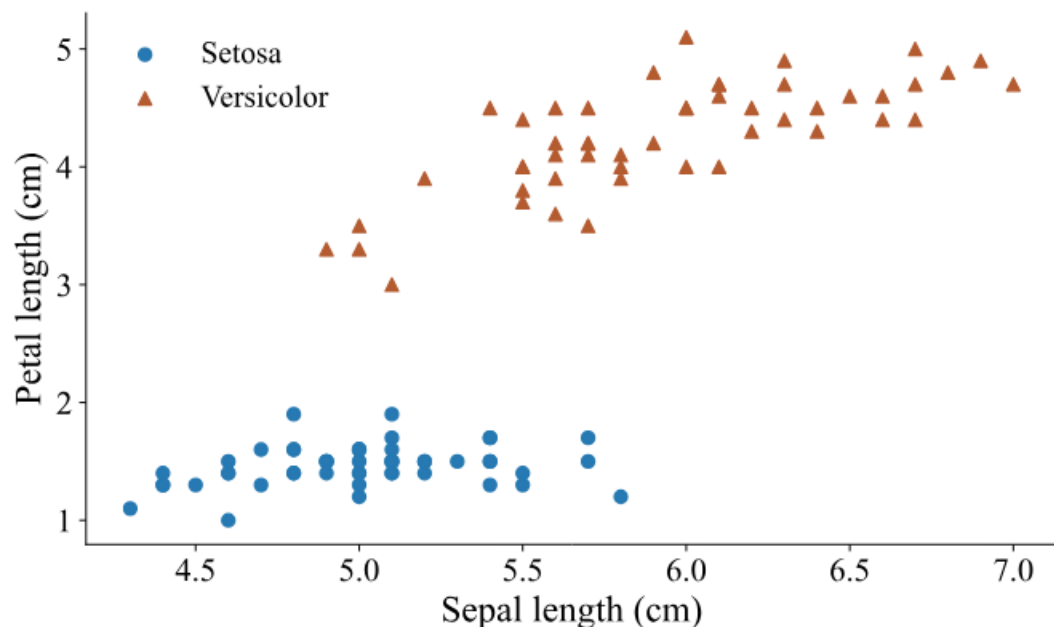
Calling

```
ppn = lm.Perceptron(eta0=0.1, random_state=1)
ppn.fit(X_train_std, y_train)
pred = ppn.predict(X_test_std)
```

2. Classes: Two to Three

L2: 100 Samples, Two Species

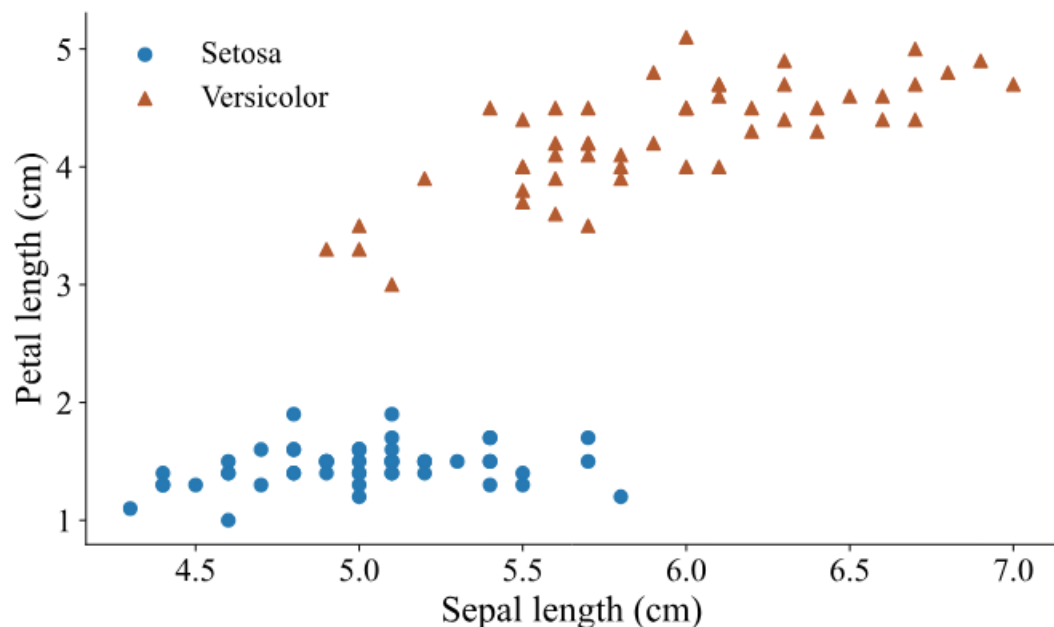
```
import pandas as pd
file = "iris.csv"
df = pd.read_csv(file, header=None)
X = df.iloc[0:100, [0, 2]].values
y = df.iloc[0:100, 4].values
```



2. Classes: Two to Three

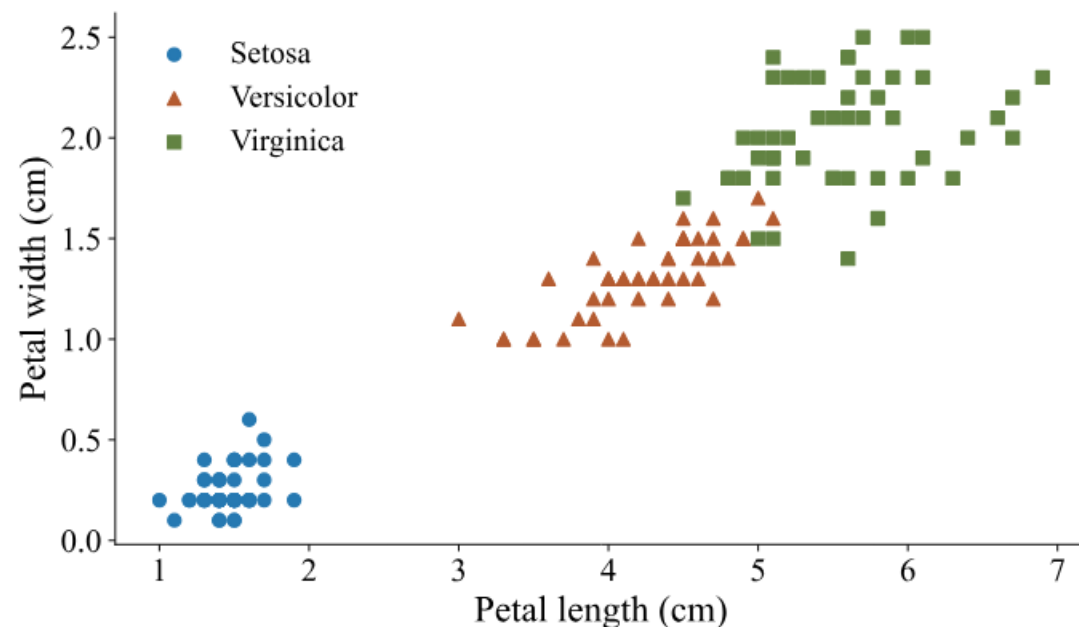
L2: 100 Samples, Two Species

```
import pandas as pd
file = "iris.csv"
df = pd.read_csv(file, header=None)
X = df.iloc[0:100, [0, 2]].values
y = df.iloc[0:100, 4].values
```

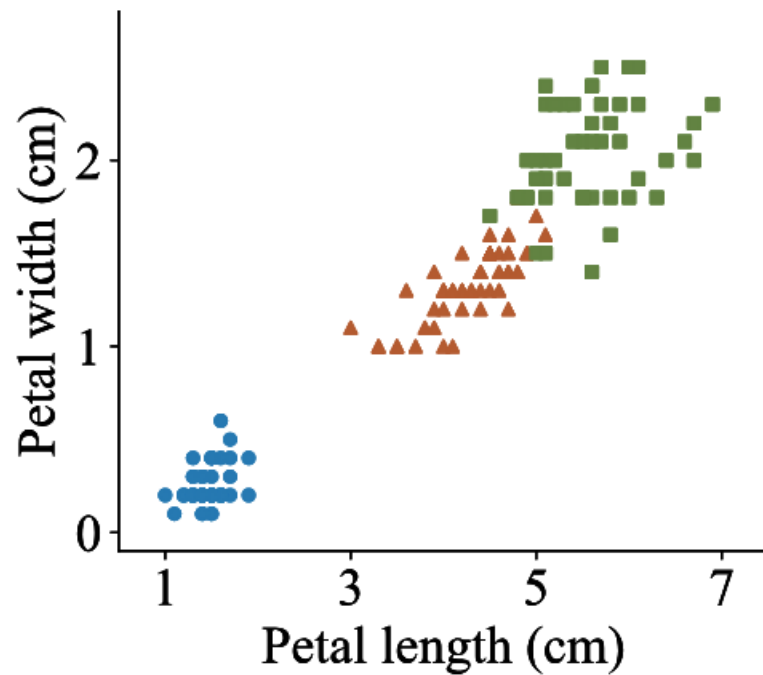


L3: 150 Samples, Three Species

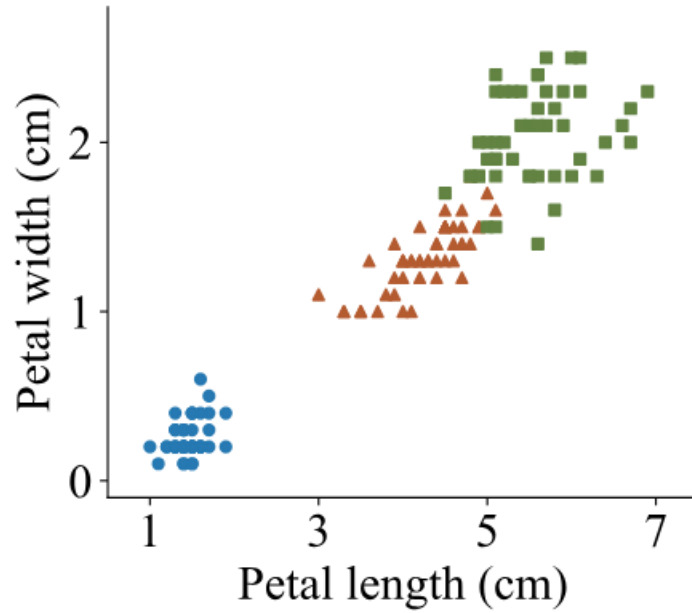
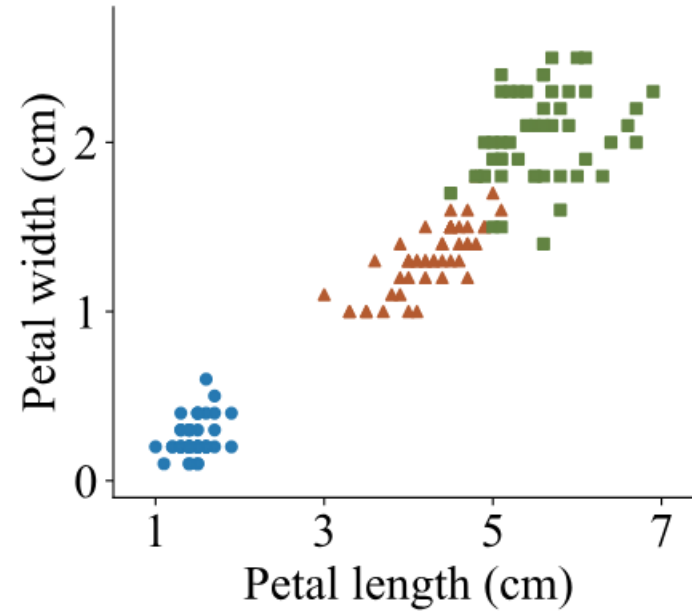
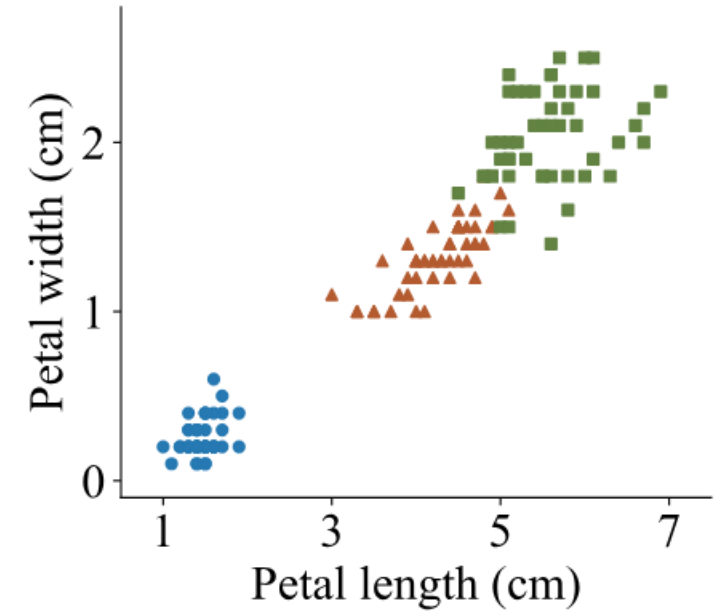
```
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data[:, [2, 3]]
y = iris.target
```



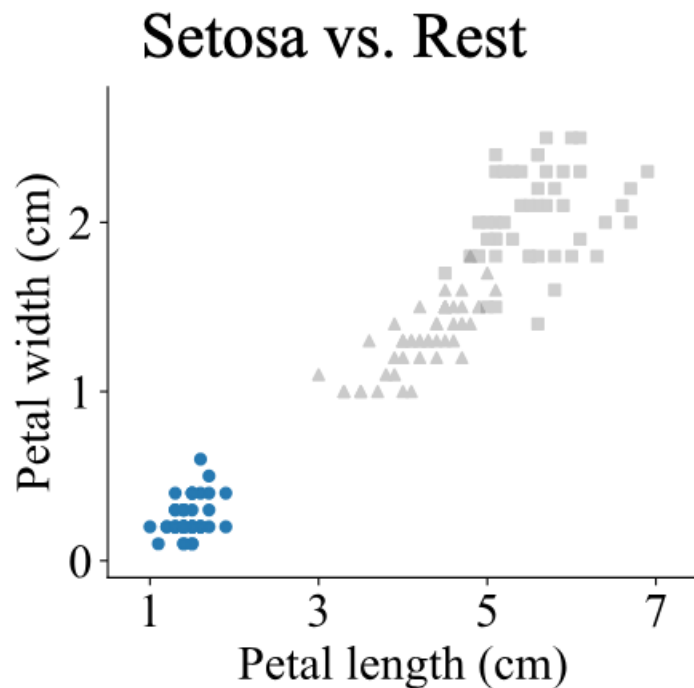
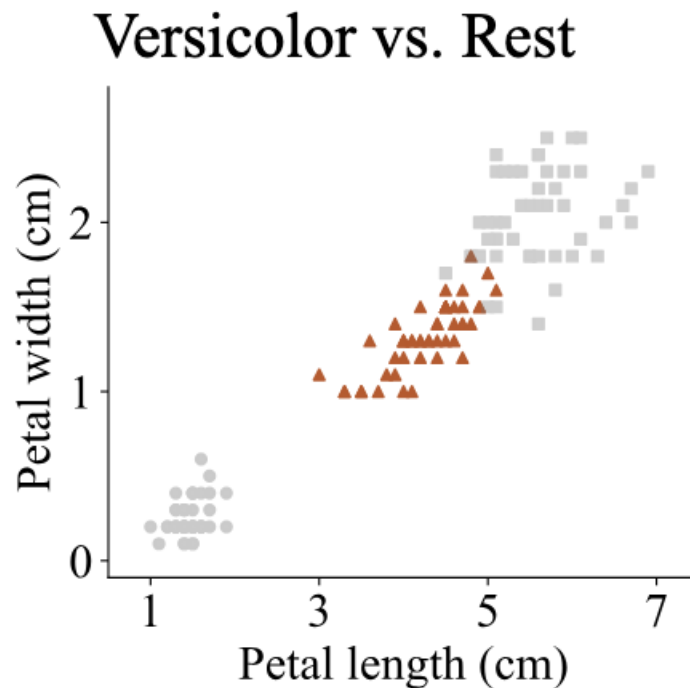
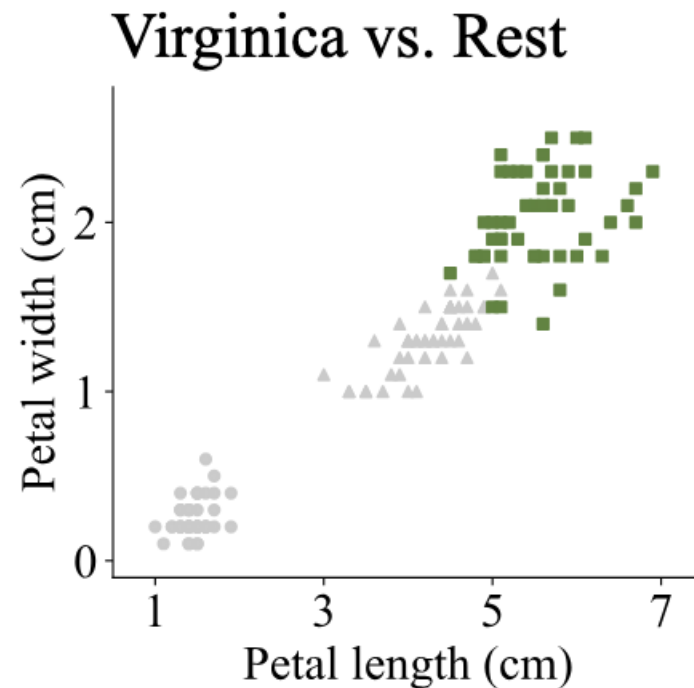
Three Classes, Three Perceptrons



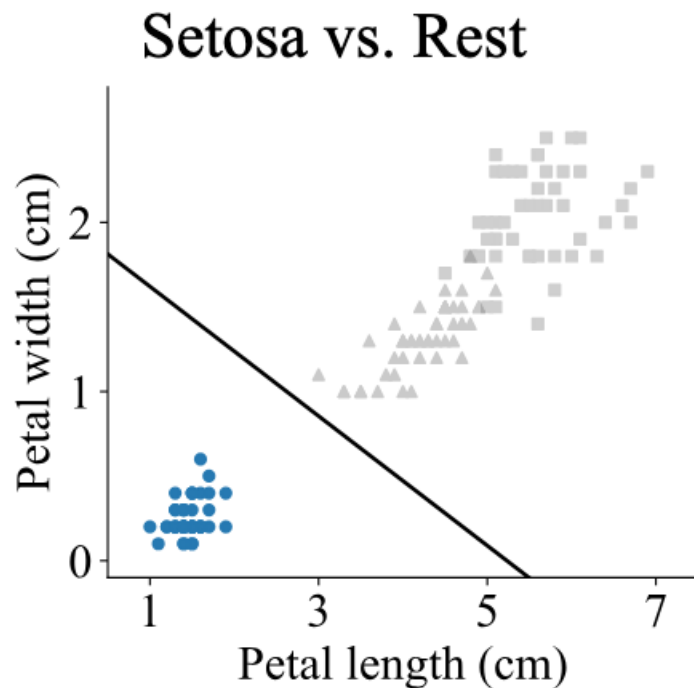
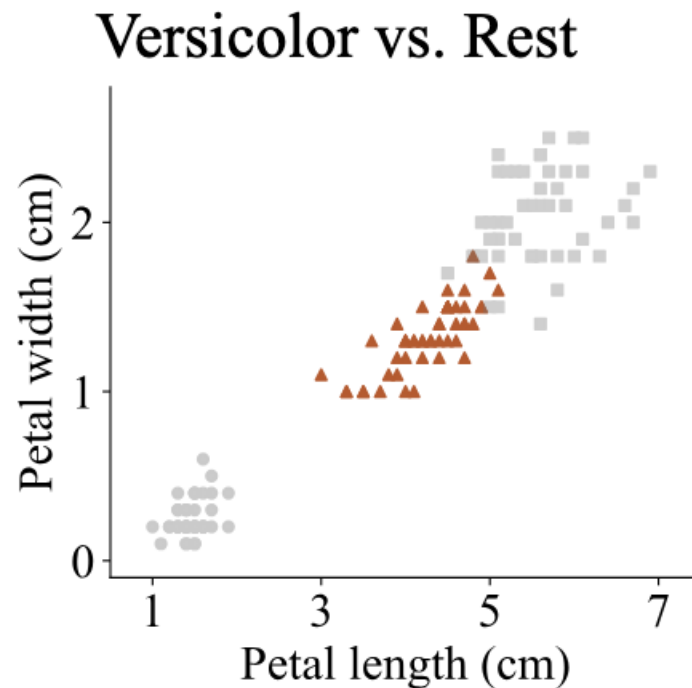
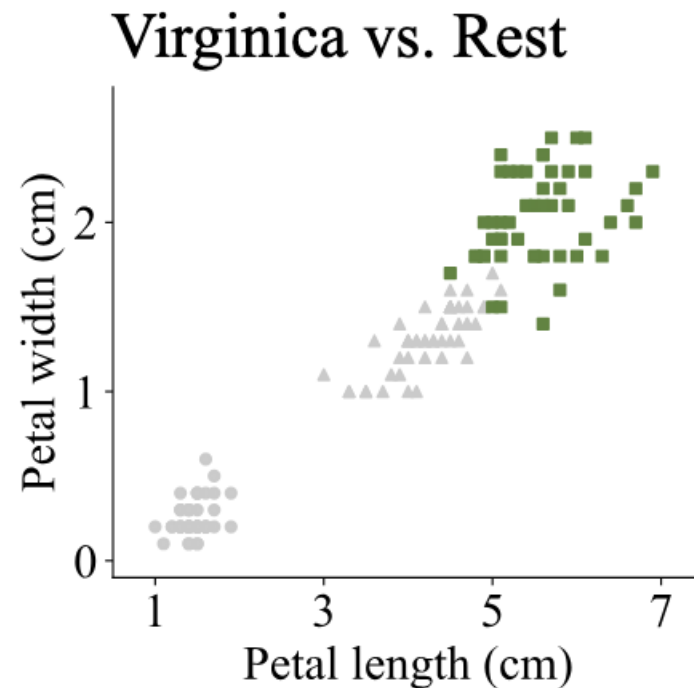
Three Classes, Three Perceptrons



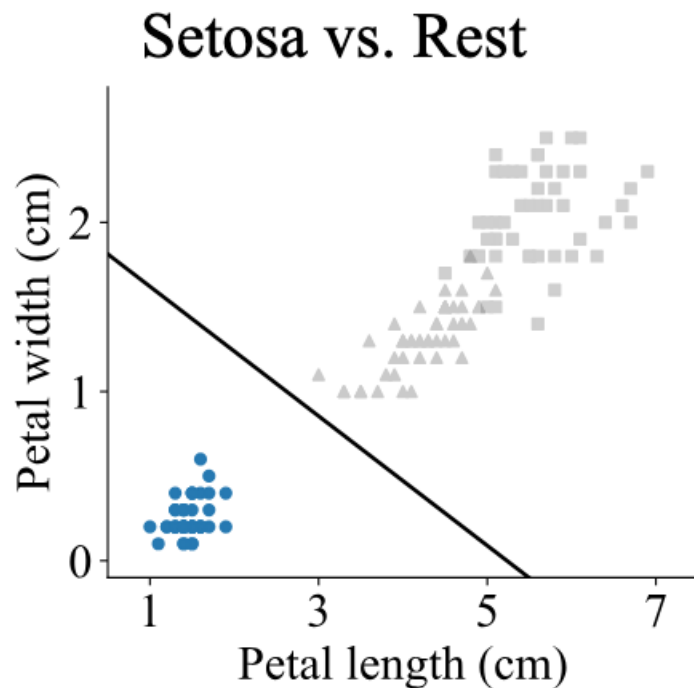
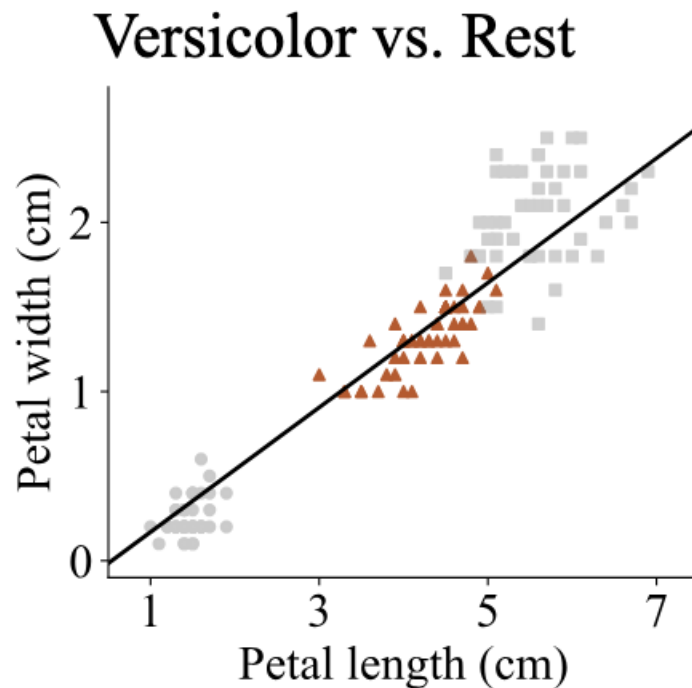
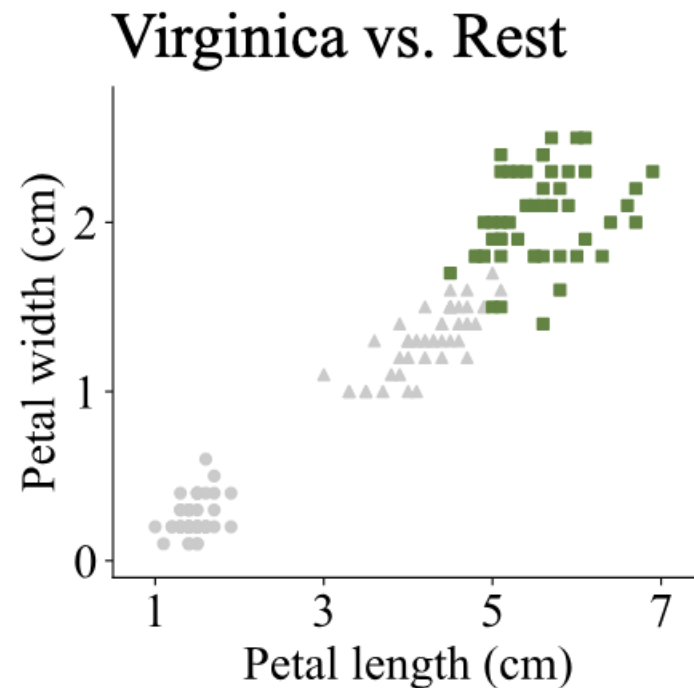
Three Classes, Three Perceptrons



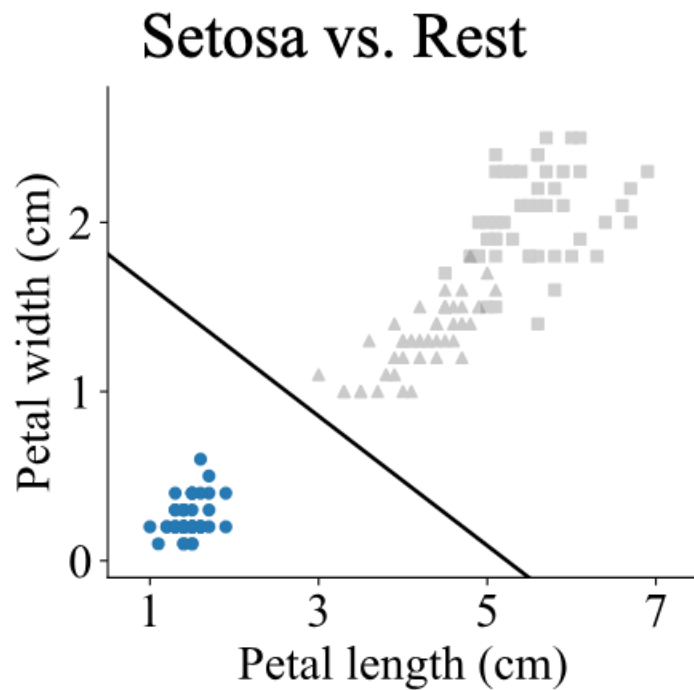
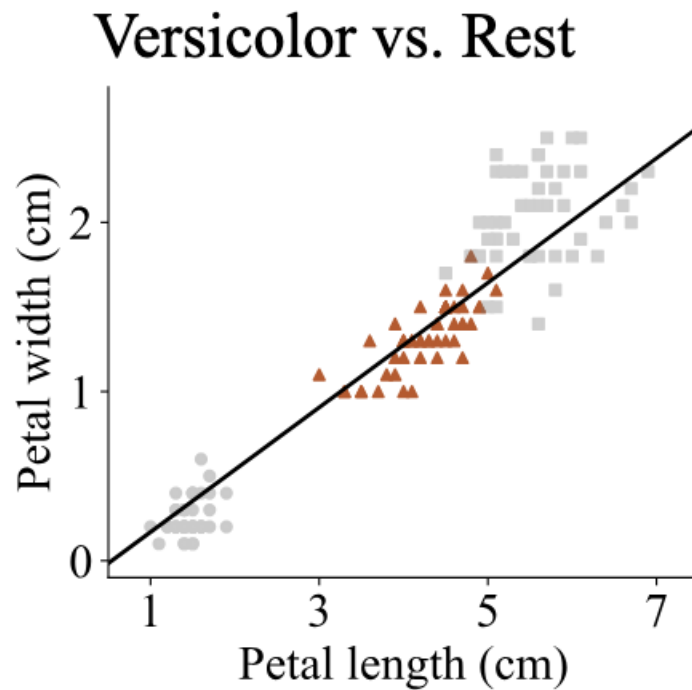
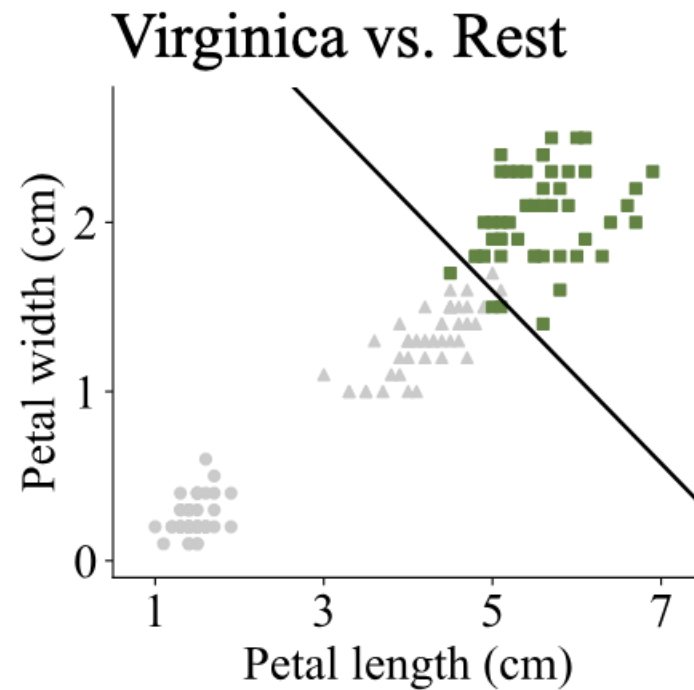
Three Classes, Three Perceptrons



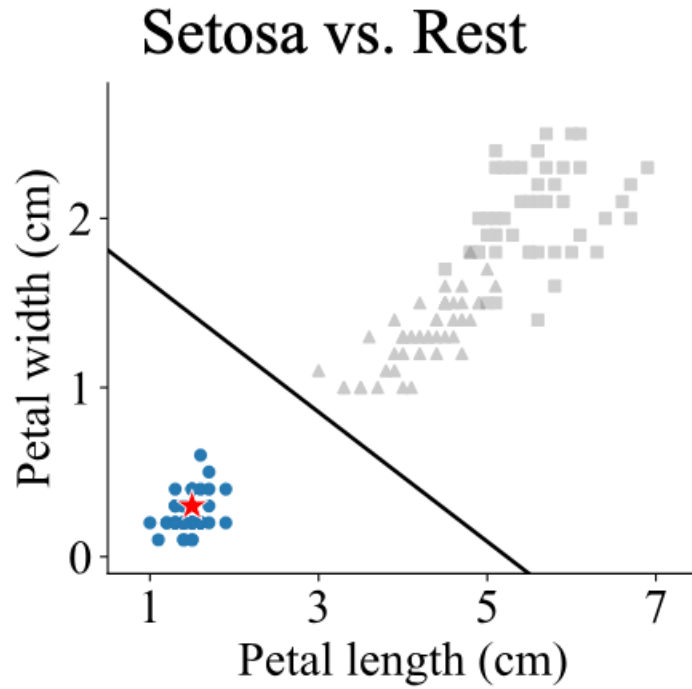
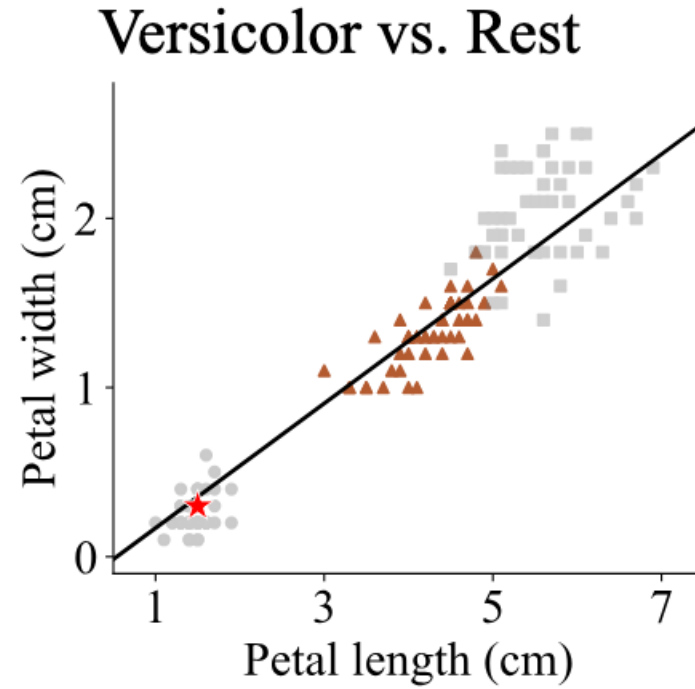
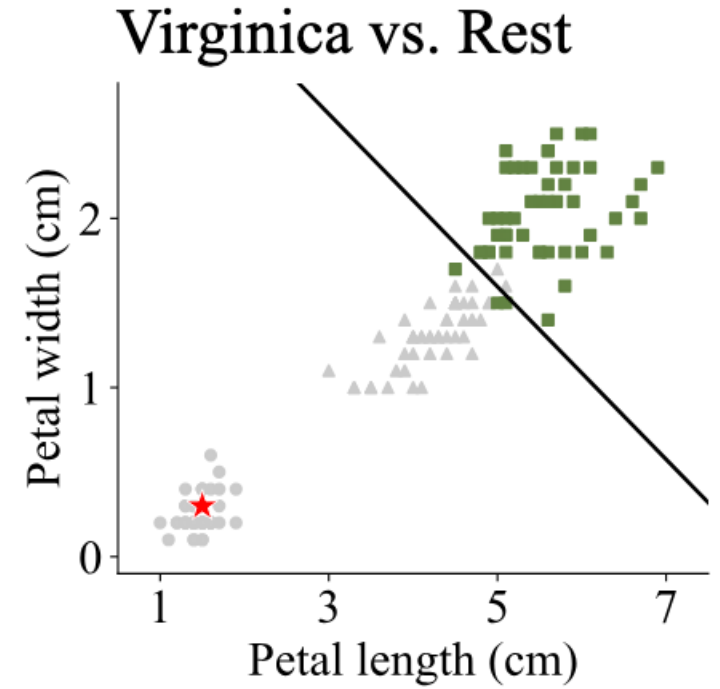
Three Classes, Three Perceptrons



Three Classes, Three Perceptrons

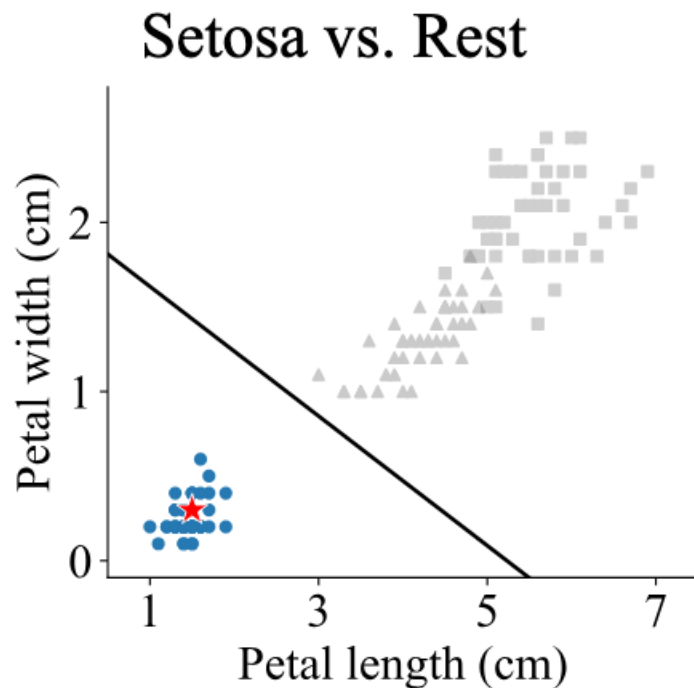


Three Classes, Three Perceptrons

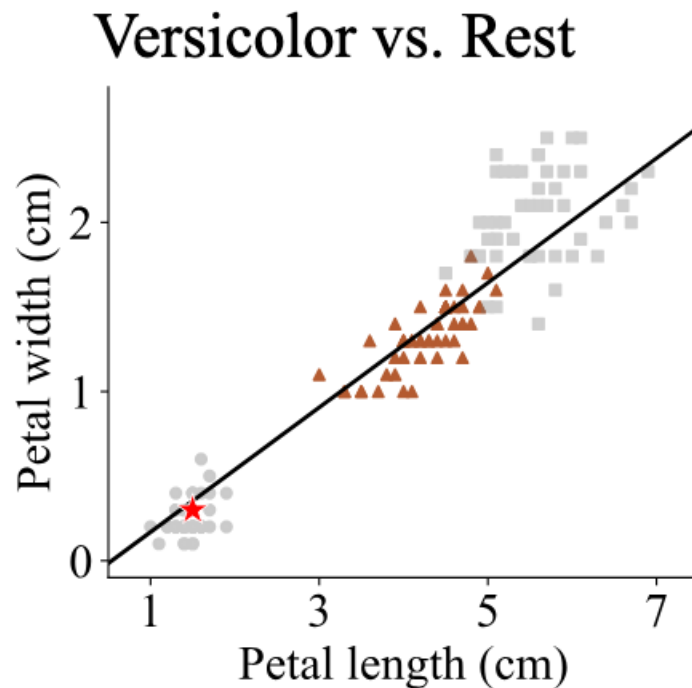


New Sample: Petal Length 1.5 cm · Petal Width 0.3 cm

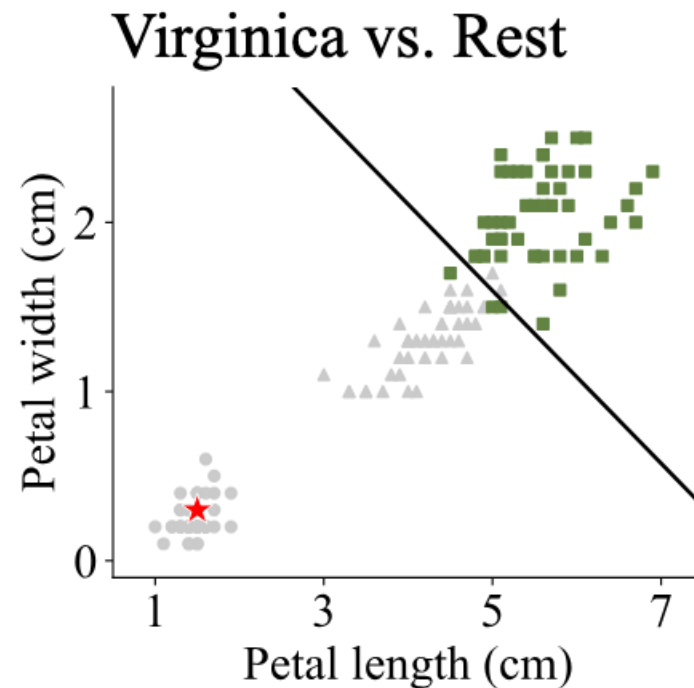
Three Classes, Three Perceptrons



$$z_{\text{setosa}} = +0.175$$



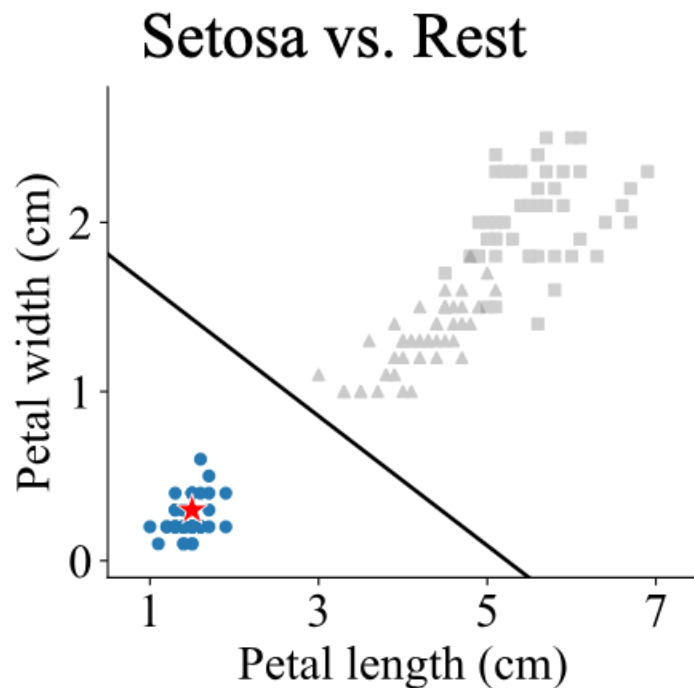
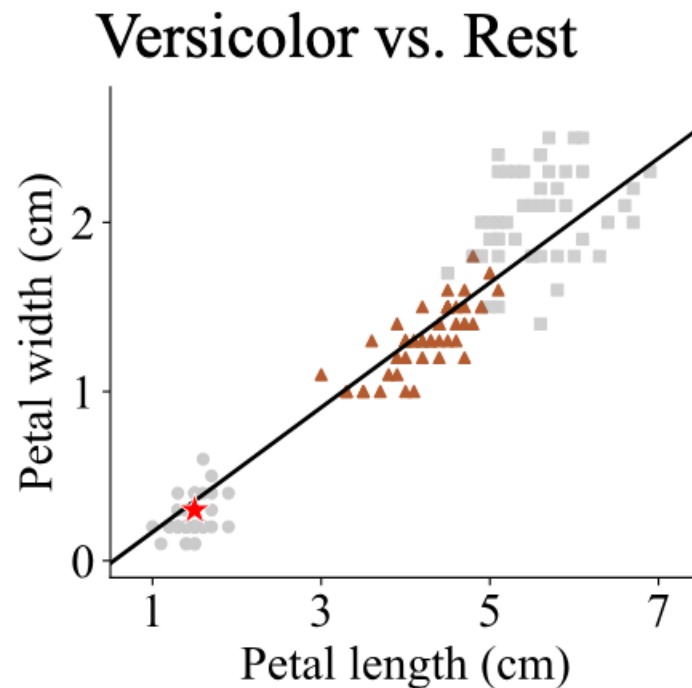
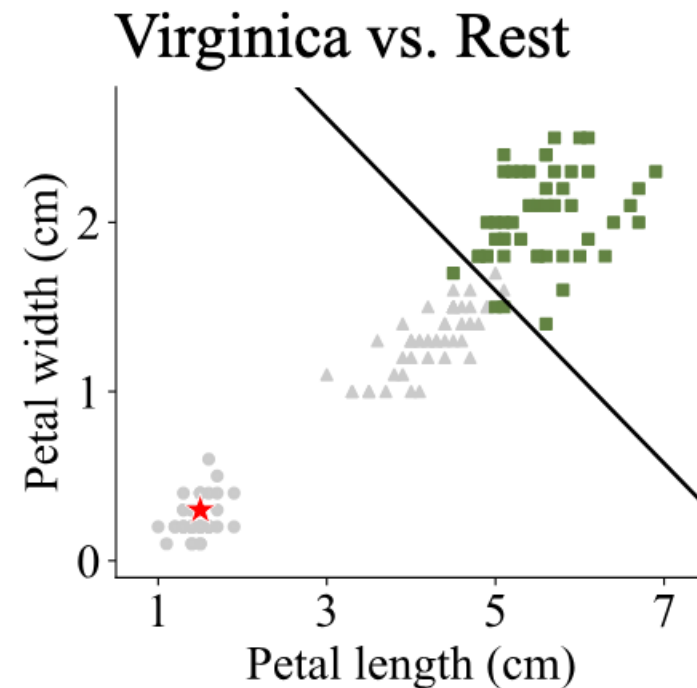
$$z_{\text{versicolor}} = +0.026$$



$$z_{\text{virginica}} = -1.213$$

New Sample: Petal Length 1.5 cm · Petal Width 0.3 cm

Three Classes, Three Perceptrons



$$z_{\text{setosa}} = +0.175$$

$$z_{\text{versicolor}} = +0.026$$

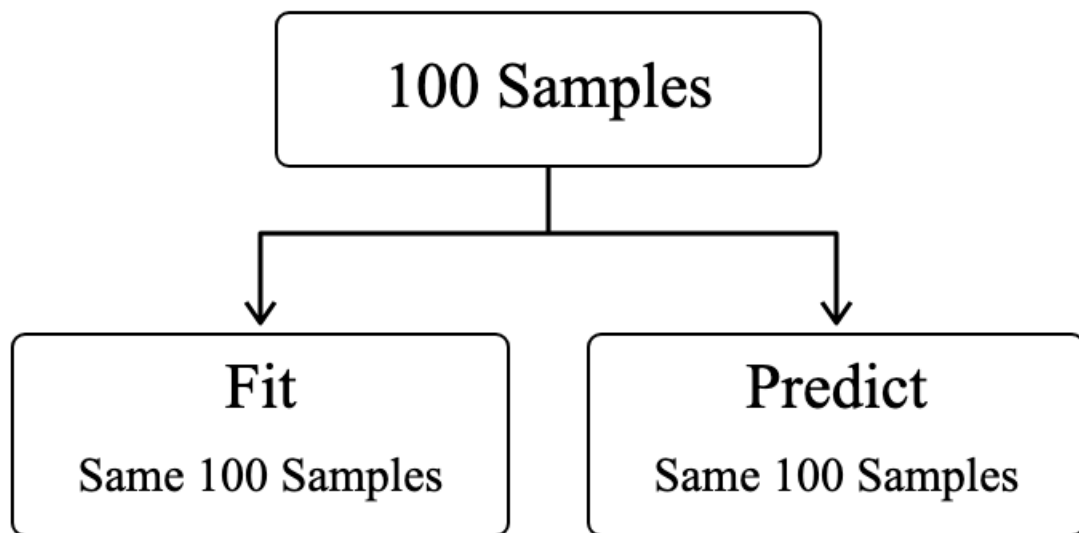
$$z_{\text{virginica}} = -1.213$$

New Sample: Petal Length 1.5 cm · Petal Width 0.3 cm

$$\hat{y} = \arg \max_{k \in \{\text{Setosa}, \text{Versicolor}, \text{Virginica}\}} z_k = \text{Setosa}$$

3. Tested On: Unseen Data

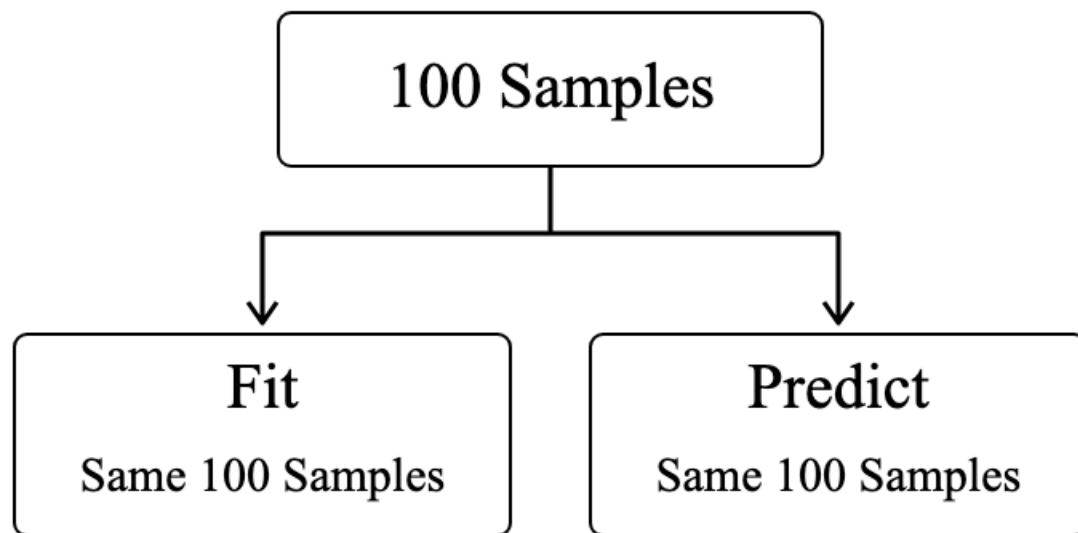
L2: Same Data



```
ppn.fit(X, y)  
ppn.predict(X)
```

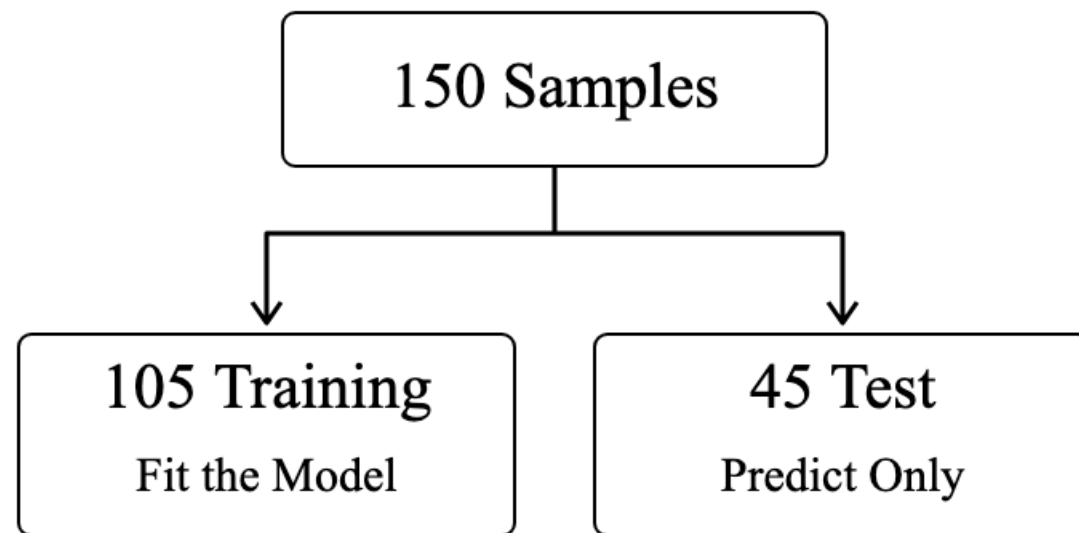
3. Tested On: Unseen Data

L2: Same Data



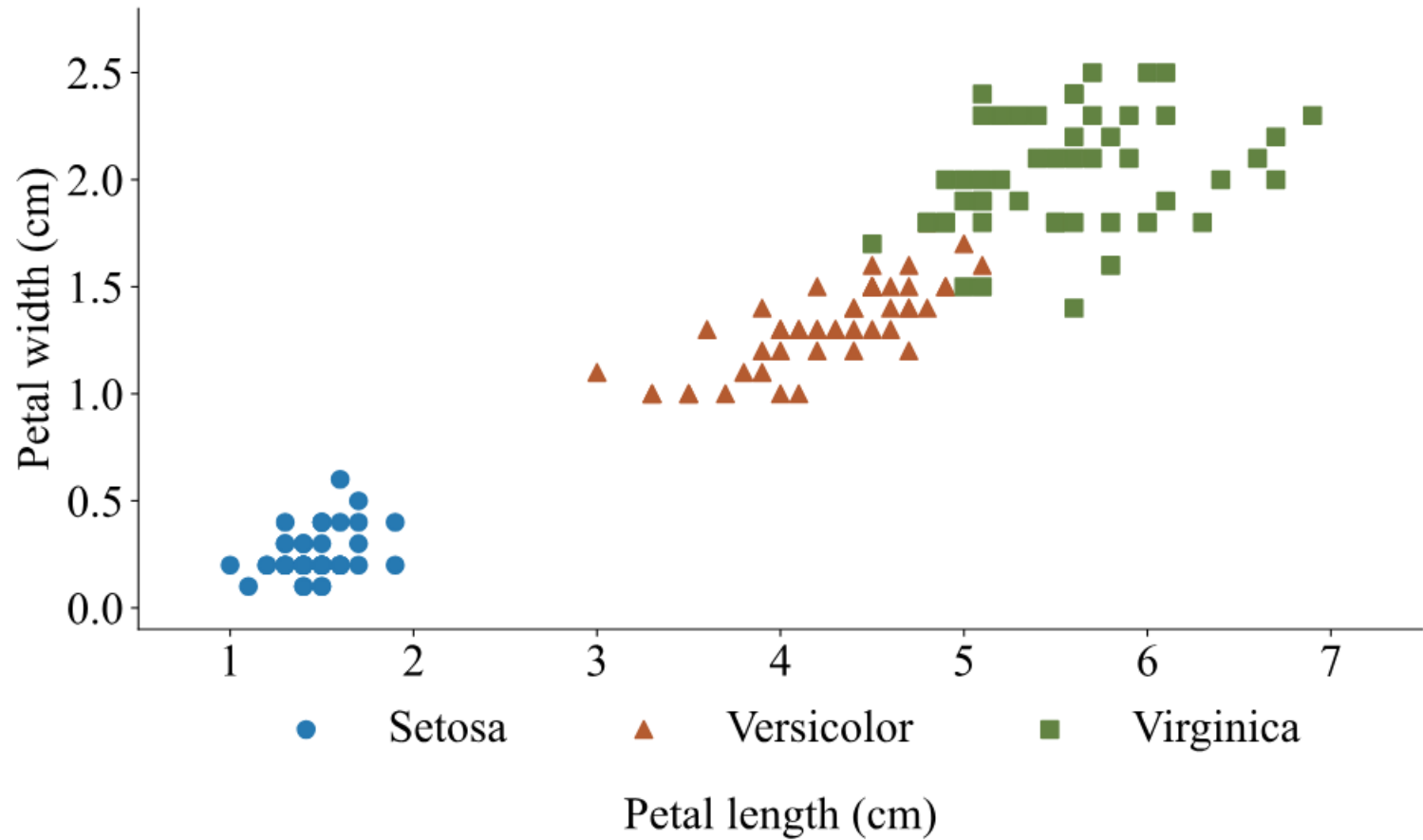
```
ppn.fit(X, y)  
ppn.predict(X)
```

L3: Unseen Data

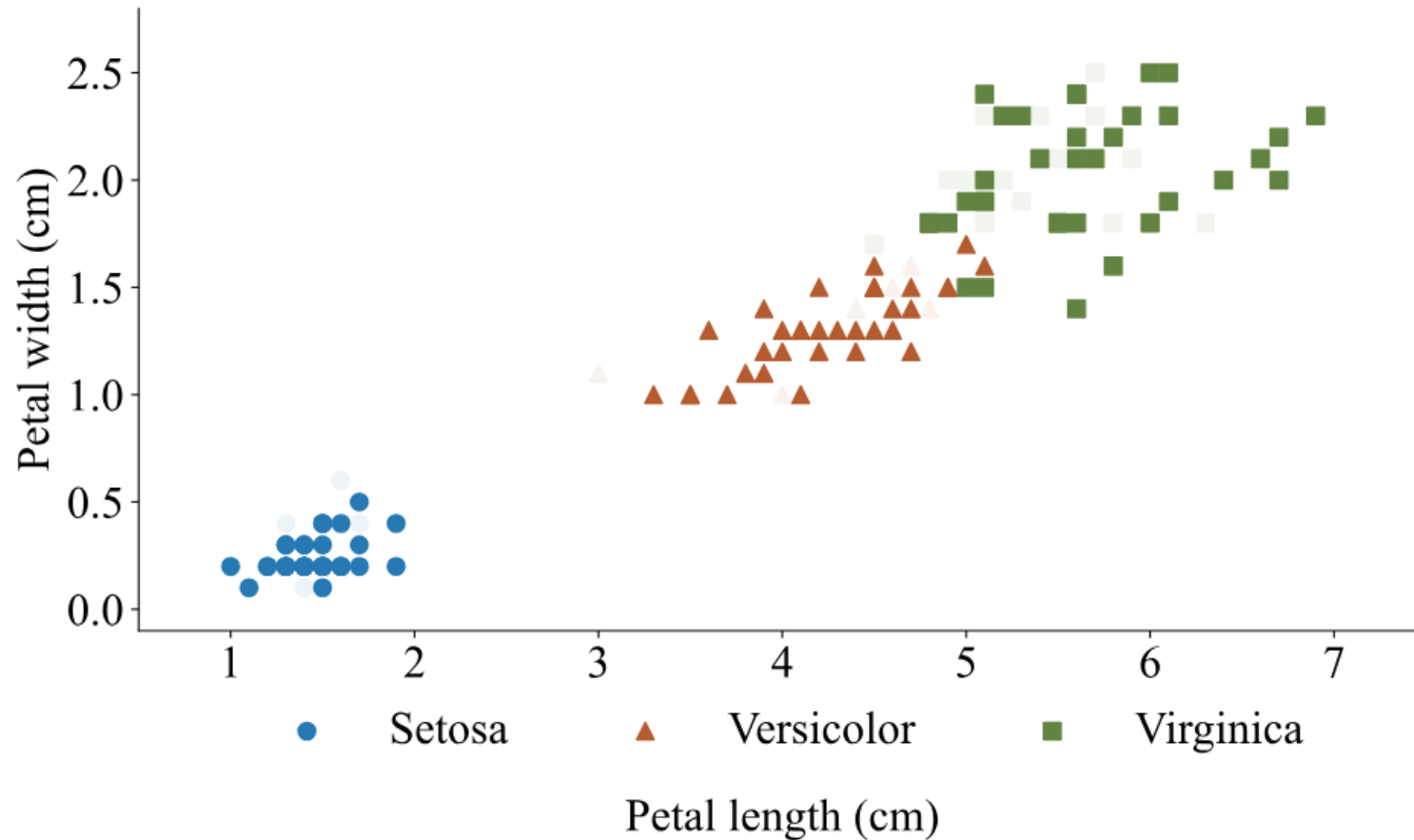


```
ppn.fit(X_train_std, y_train)  
ppn.predict(X_test_std)
```

Training and Test Samples

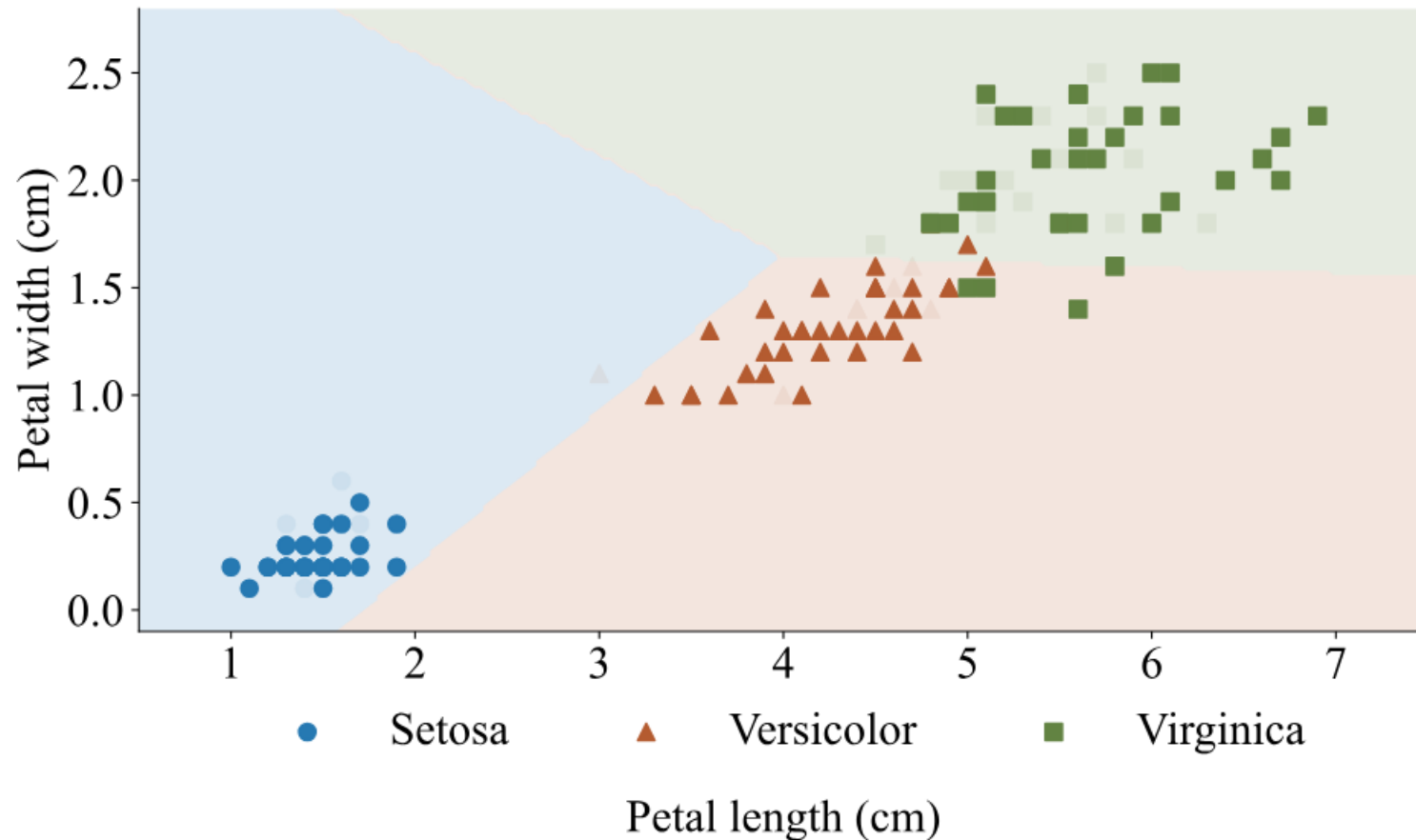


Training and Test Samples



1. Hold Out 45 Samples
Keep Them Aside for Testing

Training and Test Samples



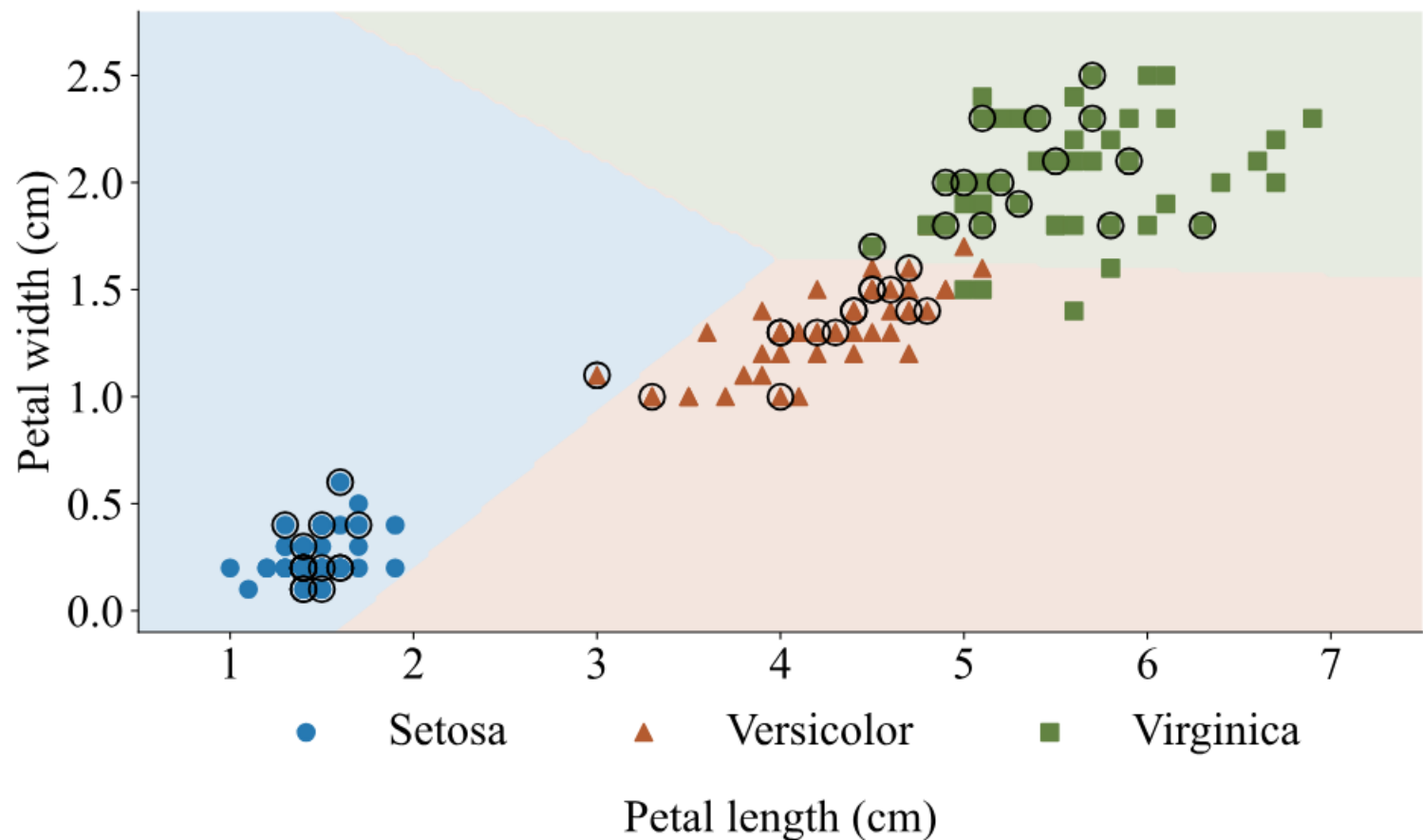
1. Hold Out 45 Samples

Keep Them Aside for Testing

2. Fit Using 105 Samples

```
ppn.fit(X_train_std, y_train)
```

Training and Test Samples



1. Hold Out 45 Samples

Keep Them Aside for Testing

2. Fit Using 105 Samples

```
ppn.fit(X_train_std, y_train)
```

3. Predict the Test Samples

```
ppn.predict(X_test_std)
```

Compare With `y_test`

maib7002.dirk.hk/L3.zip

Demo: `perceptron_sklearn.ipynb`

Group Exercise: Perceptron

1. maib7002.dirk.hk/L3.zip, extract it, and open `l3` in JupyterLab
 2. Duplicate `perceptron_sklearn.ipynb`
 3. Save as `group_XX_perceptron.ipynb` and run all cells
-
1. Use code to print the number of training samples and test samples.
 2. Use code to count and print the incorrect test predictions.
 3. Use code to predict and print the species of a sample with **petal length 4.8 cm** and **petal width 1.5 cm**.

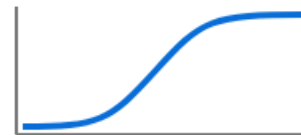
Submit Your Code and Output for All Three Tasks

One Submission per Group on Ed

Supervised Learning: L3 and L4

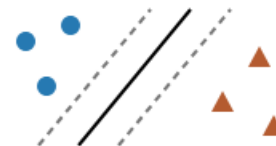
Logistic Regression

Estimate probabilities from a linear score



SVM

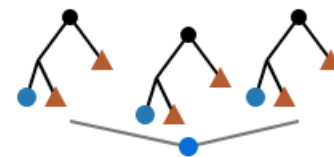
Separate classes with a margin



Random Forest

L4: Decision Trees and
Forests

Learn questions, then combine many trees



kNN

L4

Use the labels of nearby examples



Logistic Regression

Perceptron

Will This Customer Leave?

Leave

A Class Prediction

Logistic Regression

How Likely Are They to Leave?

80%

An Estimated Probability

Illustrative Customer Churn Example

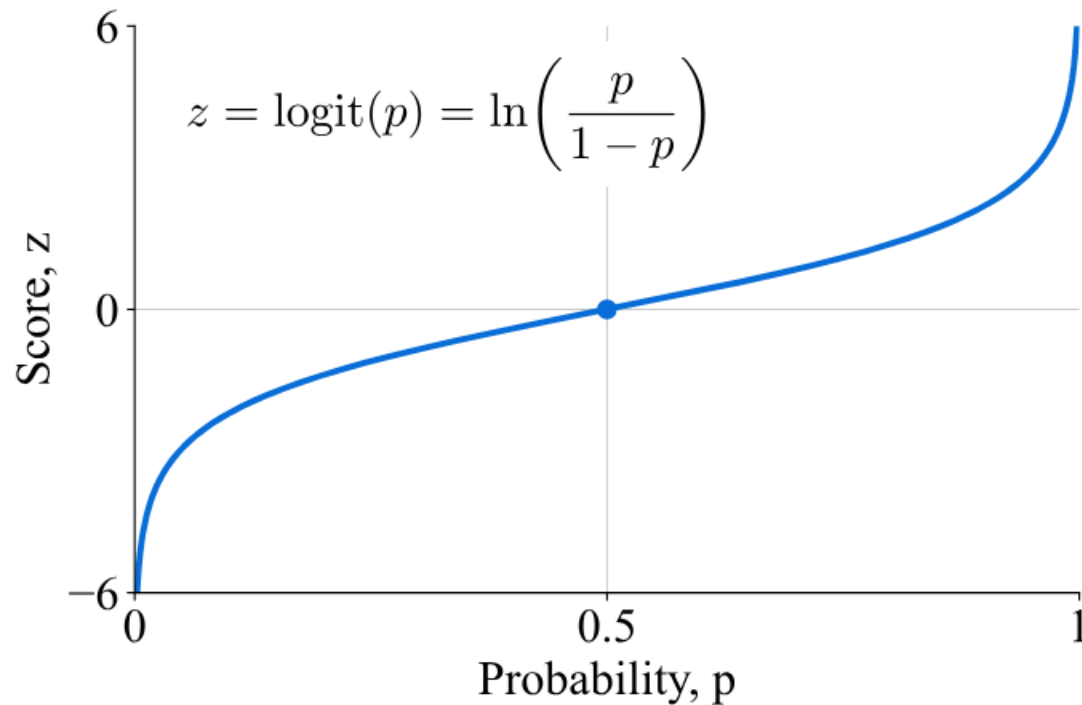
Used for Classification, Despite the Name “Regression”

1944 · Joseph Berkson

Applied the Logistic Function to Biological Response Data

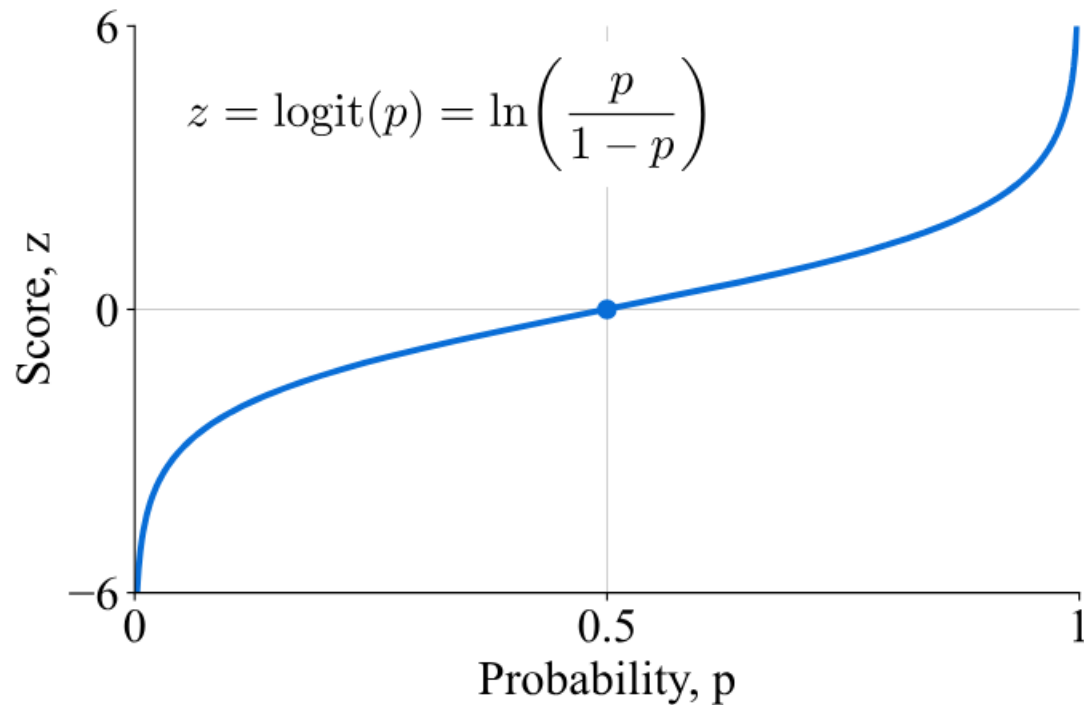
Logit and Its Inverse: Sigmoid

Logit: Probability $\rightarrow$ Score

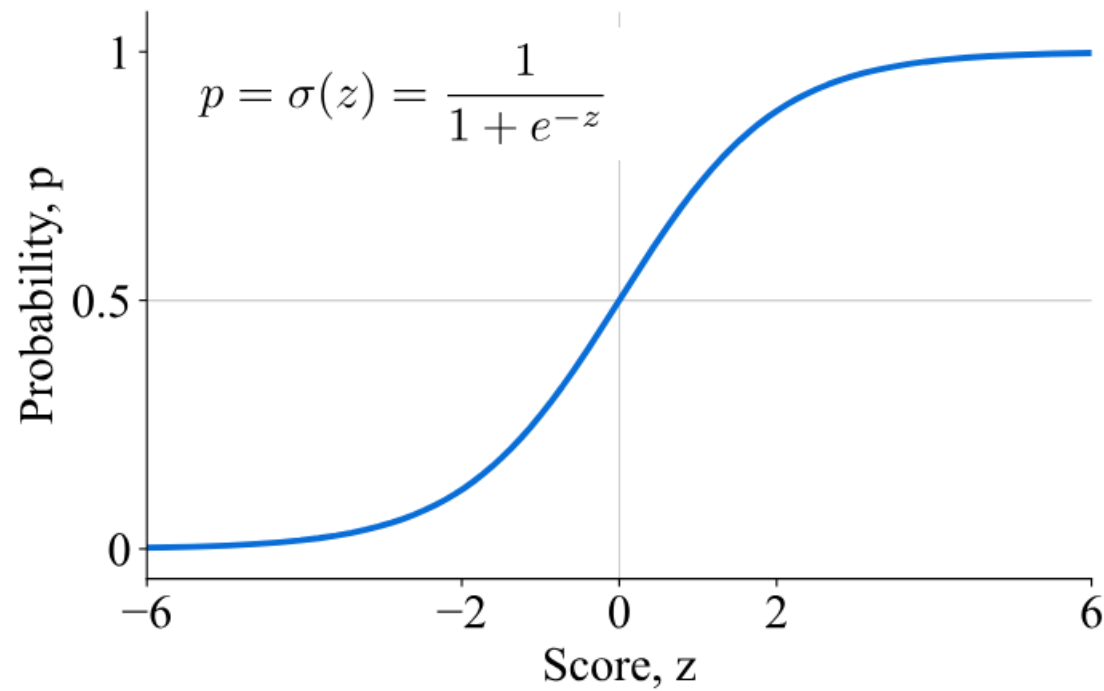


Logit and Its Inverse: Sigmoid

Logit: Probability $\rightarrow$ Score

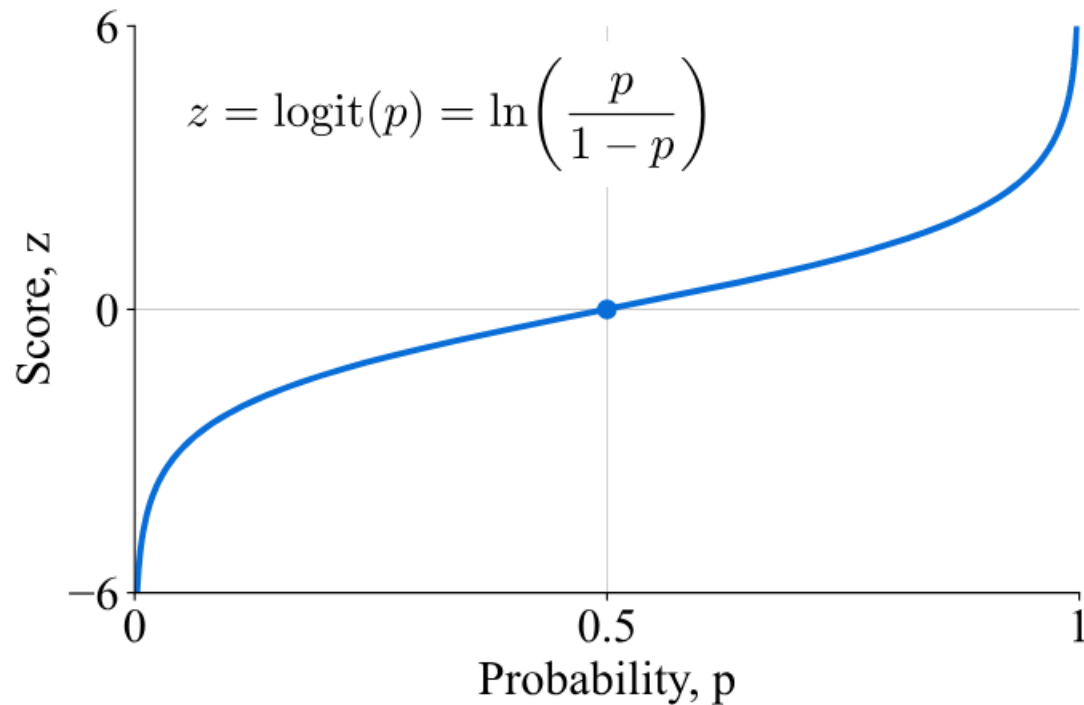


Sigmoid: Score $\rightarrow$ Probability

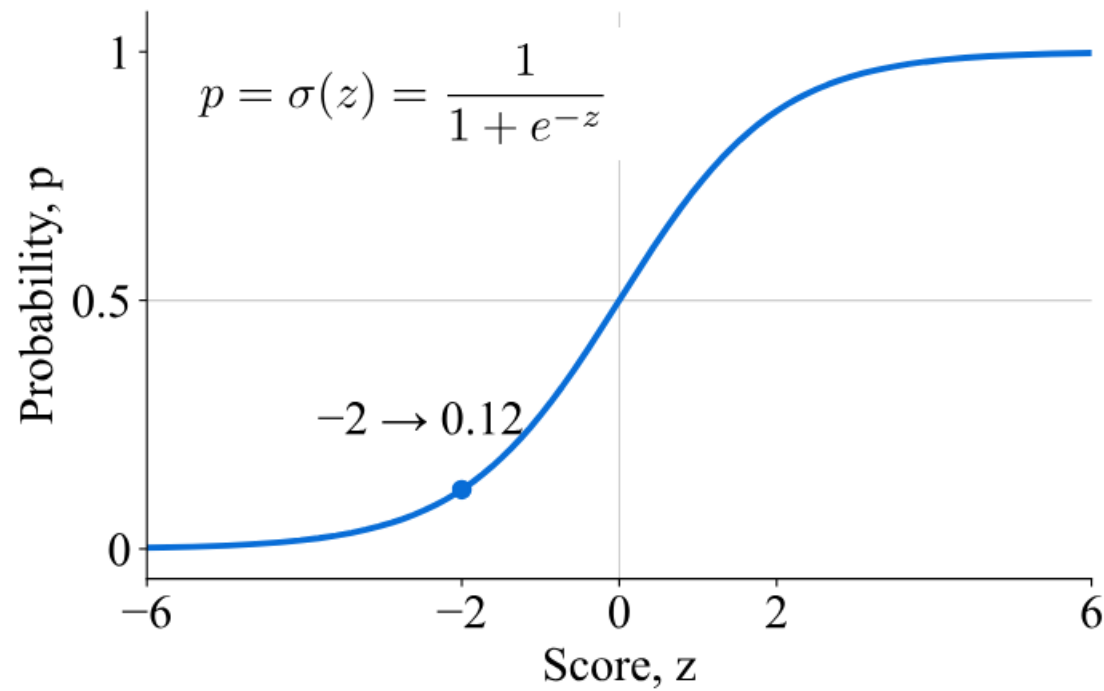


Logit and Its Inverse: Sigmoid

Logit: Probability $\rightarrow$ Score

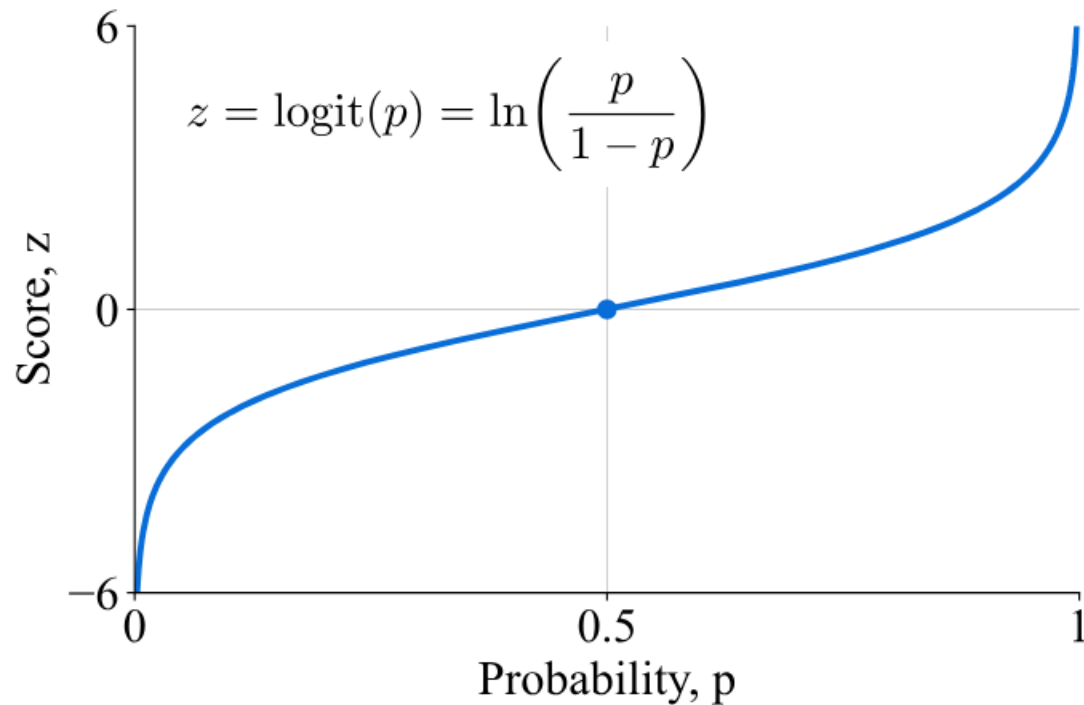


Sigmoid: Score $\rightarrow$ Probability

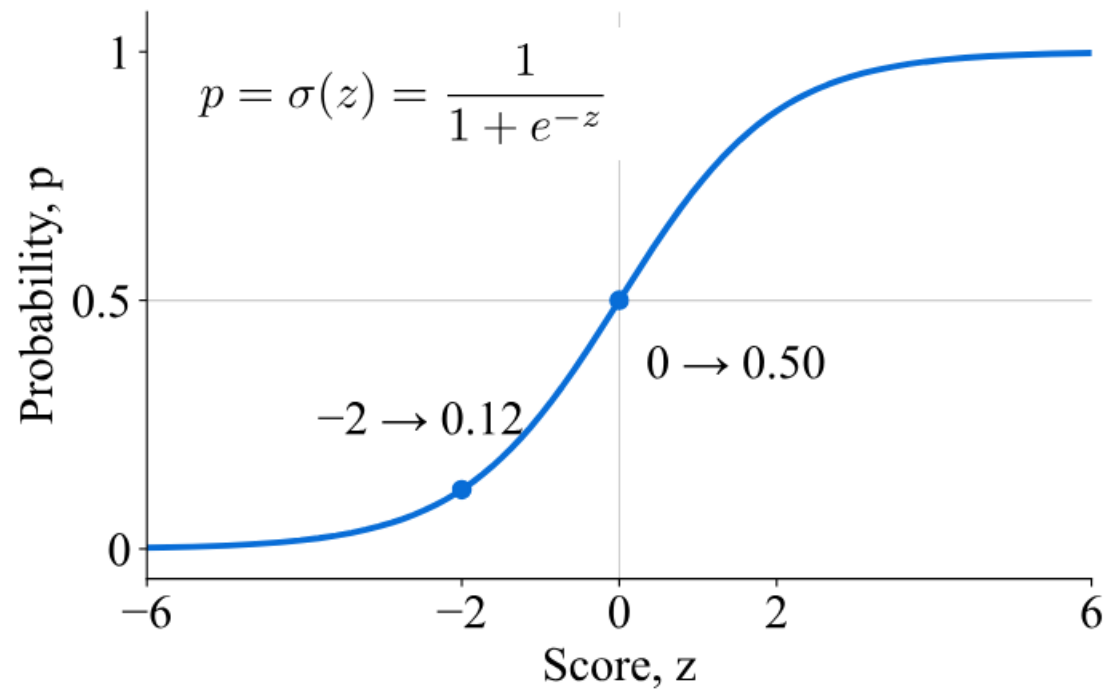


Logit and Its Inverse: Sigmoid

Logit: Probability $\rightarrow$ Score

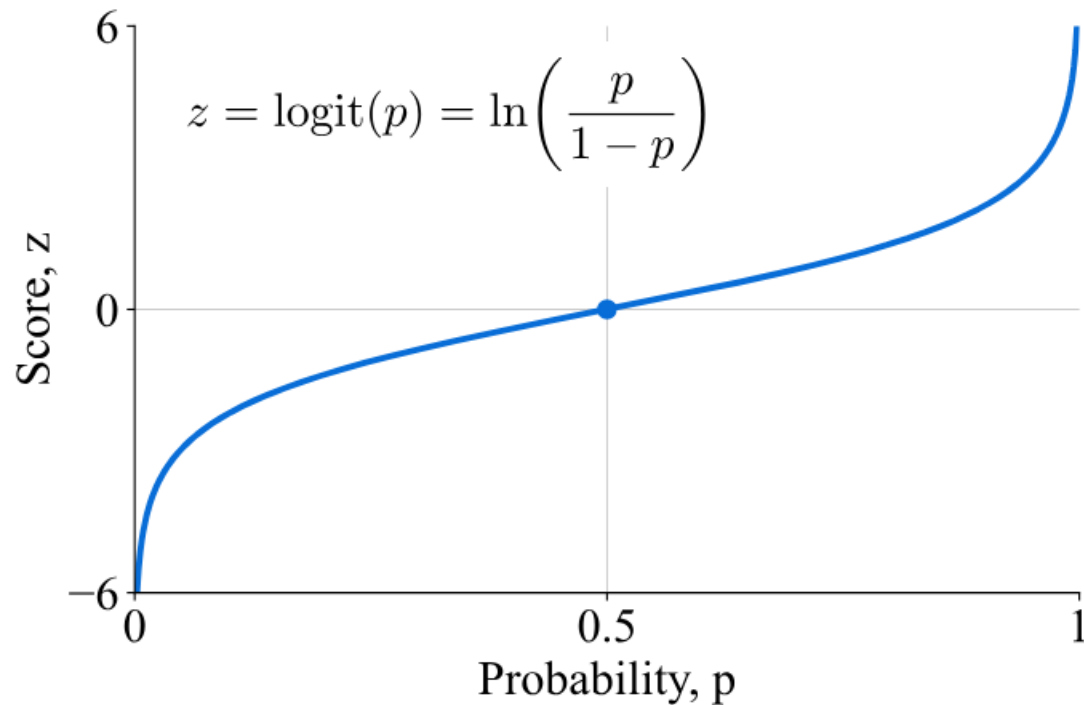


Sigmoid: Score $\rightarrow$ Probability

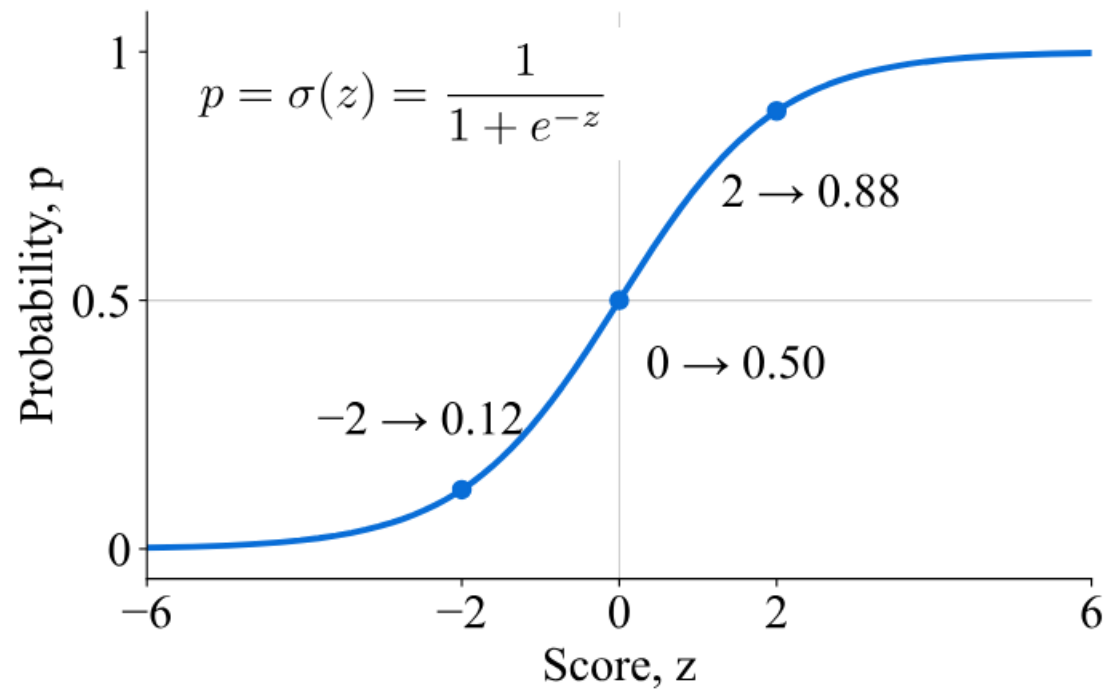


Logit and Its Inverse: Sigmoid

Logit: Probability $\rightarrow$ Score

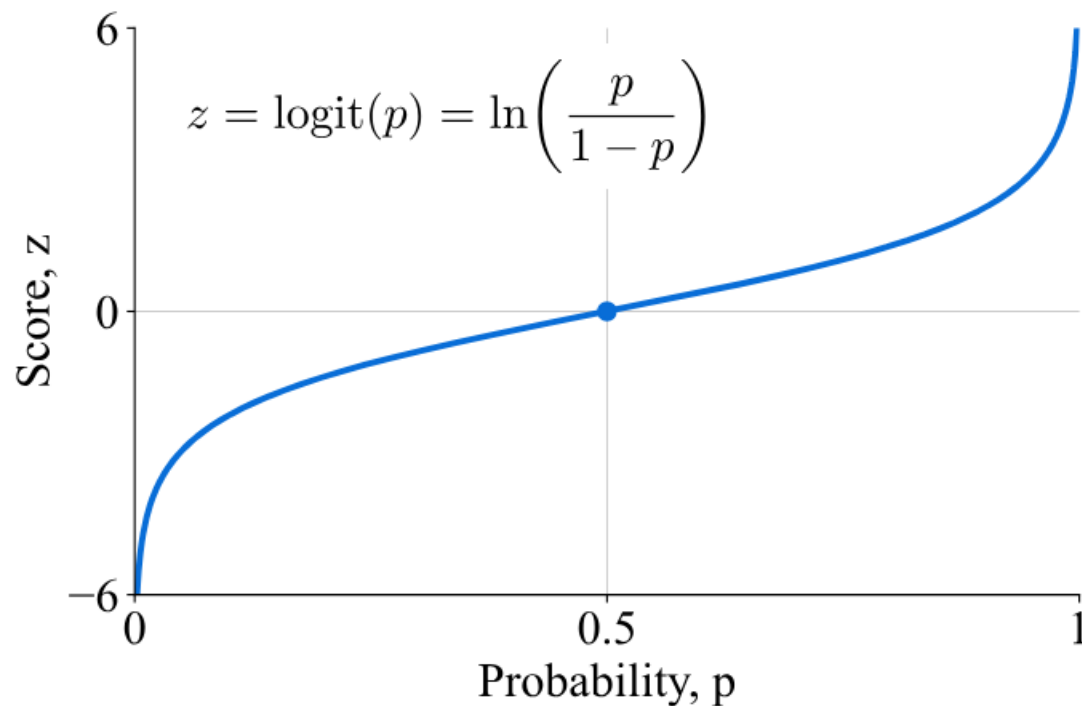


Sigmoid: Score $\rightarrow$ Probability

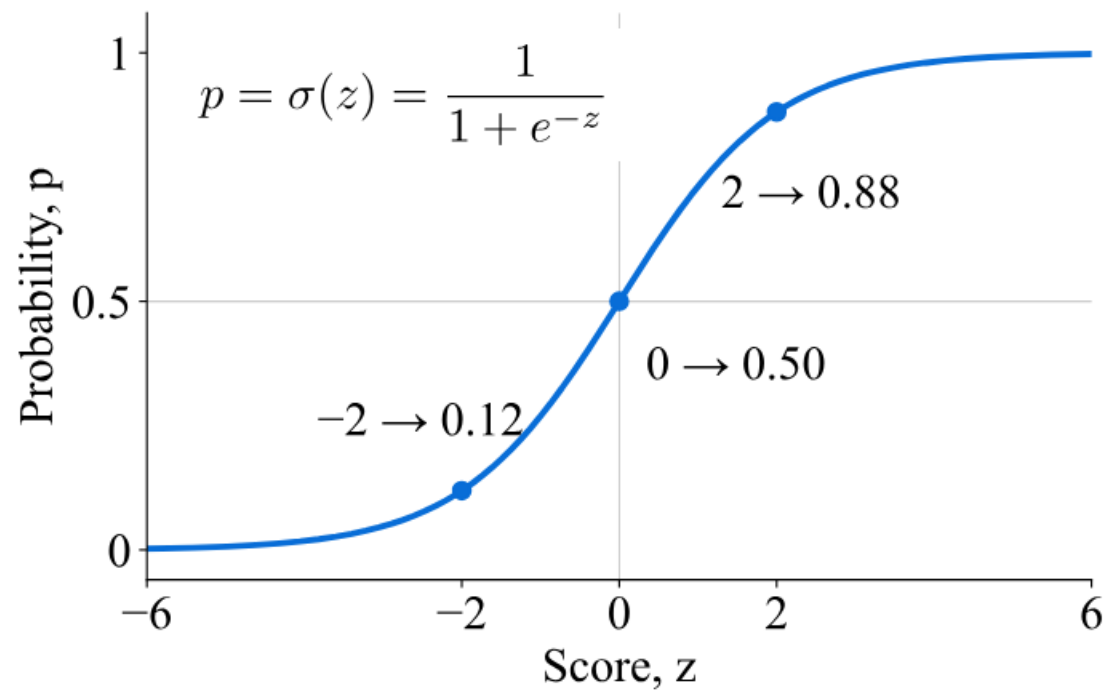


Logit and Its Inverse: Sigmoid

Logit: Probability $\rightarrow$ Score



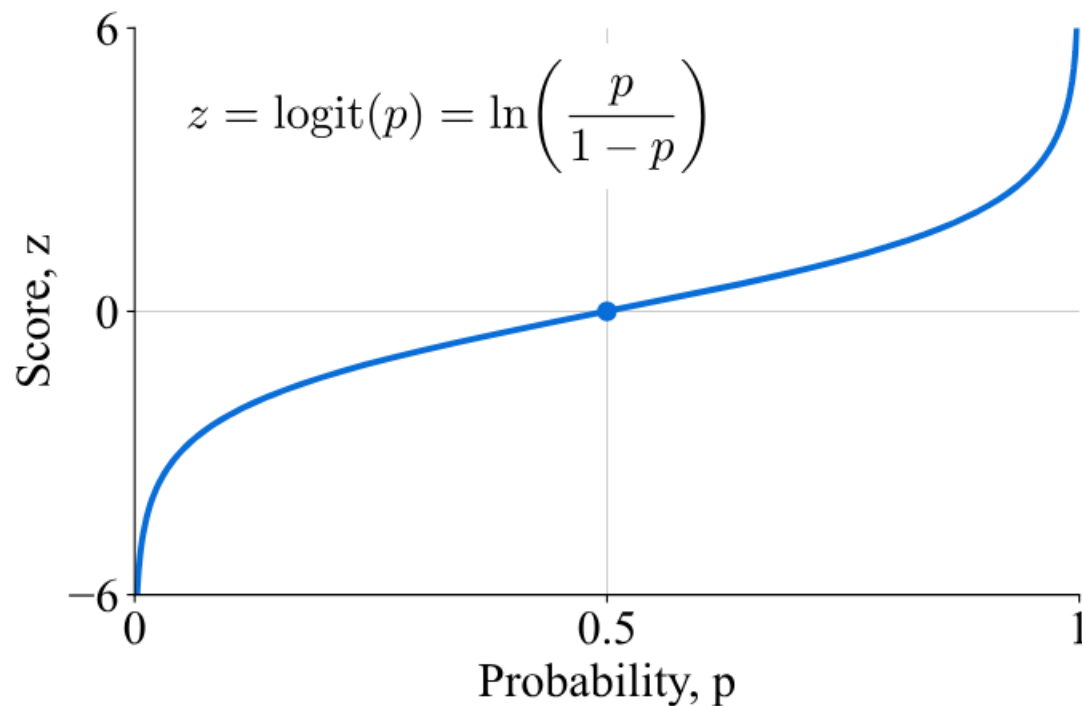
Sigmoid: Score $\rightarrow$ Probability



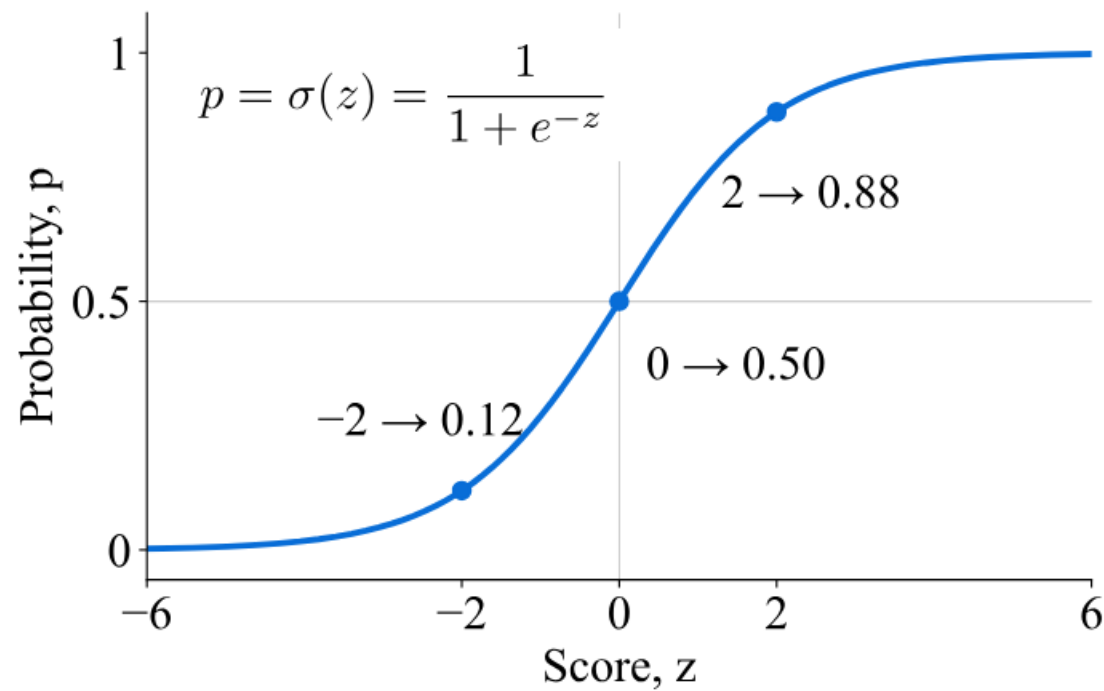
Logistic Regression Uses a Linear Score $z = \mathbf{w}^T \mathbf{x} + b$

Logit and Its Inverse: Sigmoid

Logit: Probability $\rightarrow$ Score



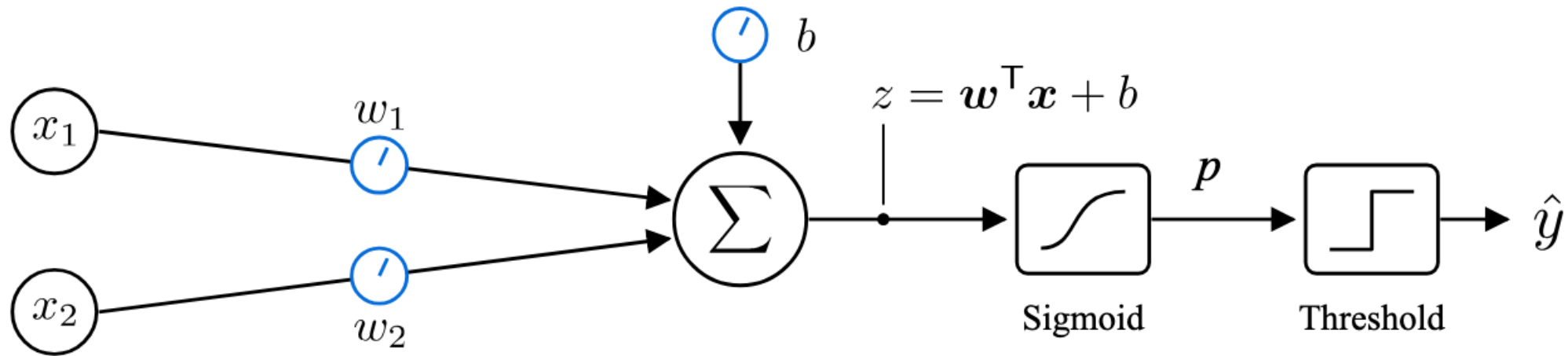
Sigmoid: Score $\rightarrow$ Probability



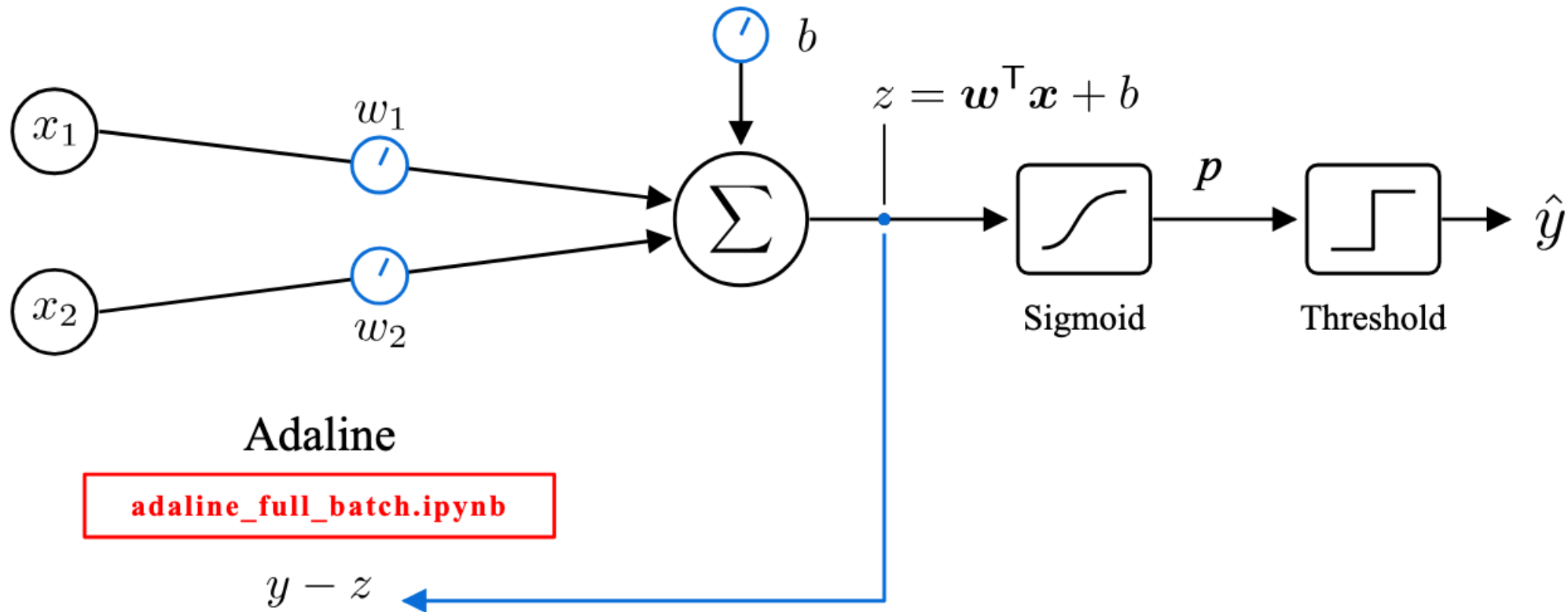
Logistic Regression Uses a Linear Score $z = \mathbf{w}^T \mathbf{x} + b$

If the Score Increases, What Happens to the Predicted Probability?

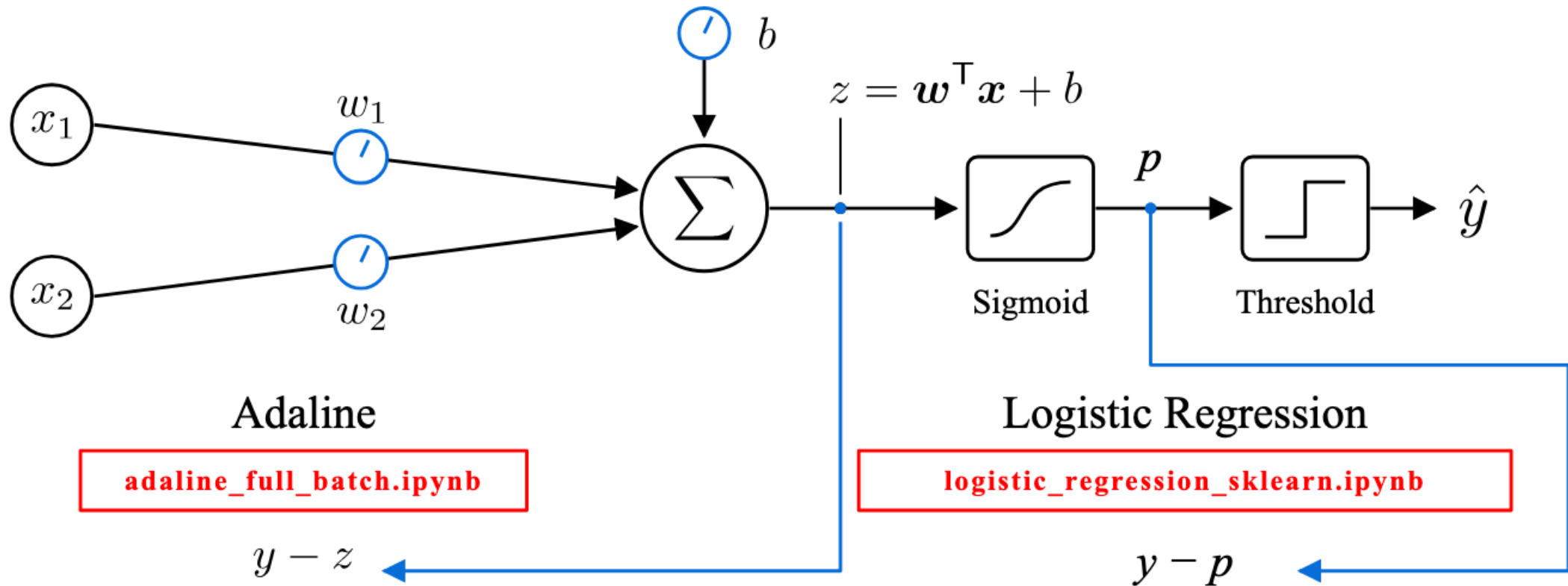
Adaline vs. Logistic Regression



Adaline vs. Logistic Regression



Adaline vs. Logistic Regression



maib7002.dirk.hk/L3.zip

Demo: `logistic_regression_sklearn.ipynb`

Group Exercise: Logistic Regression

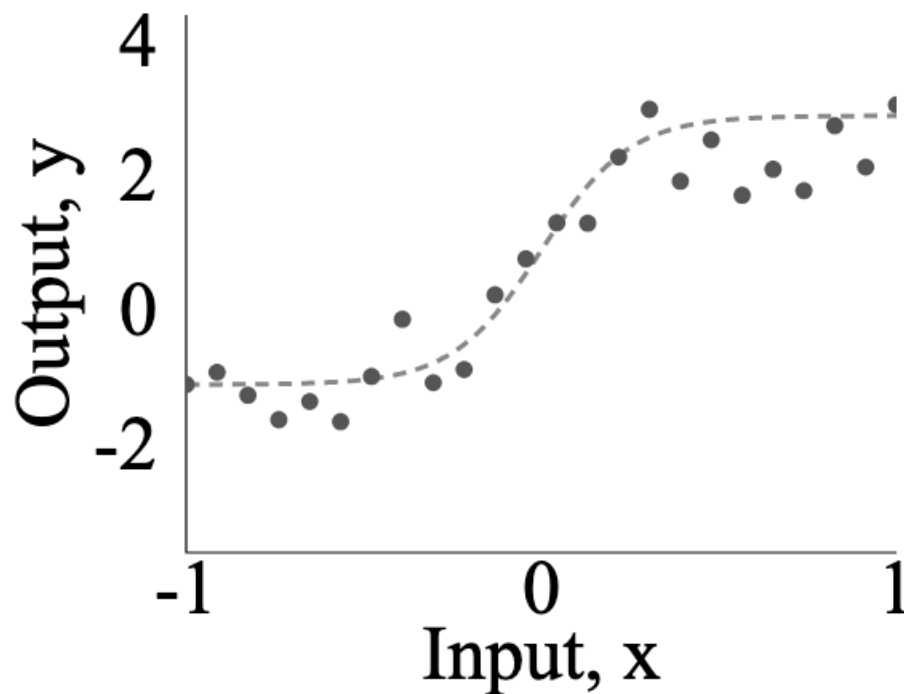
1. Duplicate `logistic_regression_sklearn.ipynb`
 2. Save as `group_XX_logistic.ipynb`; run through the prediction cells
-
1. Read `predict_proba(X_test_std[:3, :])`. For the first sample, explain the three numbers, identify its predicted species, and verify using `predict`.
 2. Find `fit` and `predict_proba`. Why does one use `X_train_std` and the other `X_test_std`? What would we measure if both used `X_train_std`?

Submit Two Explanations, Supported by the Code and Output

One Submission per Group on Ed

Overfitting: Perfect Training Fits

Original Training Data · 24 Samples

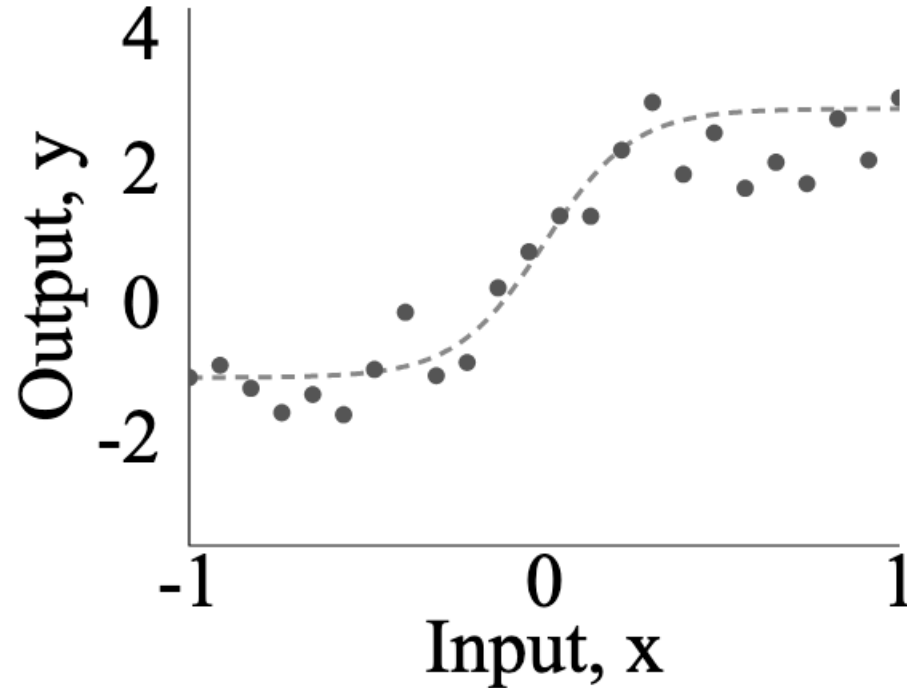


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

Illustrative Regression Example · Dashed Grey: Underlying Pattern

Overfitting: Perfect Training Fits

Original Training Data · 24 Samples



14 11 24 22 13 2 1 3 7 5 8 20 17 19 4 23 15 9 18 10 16 21 6 12

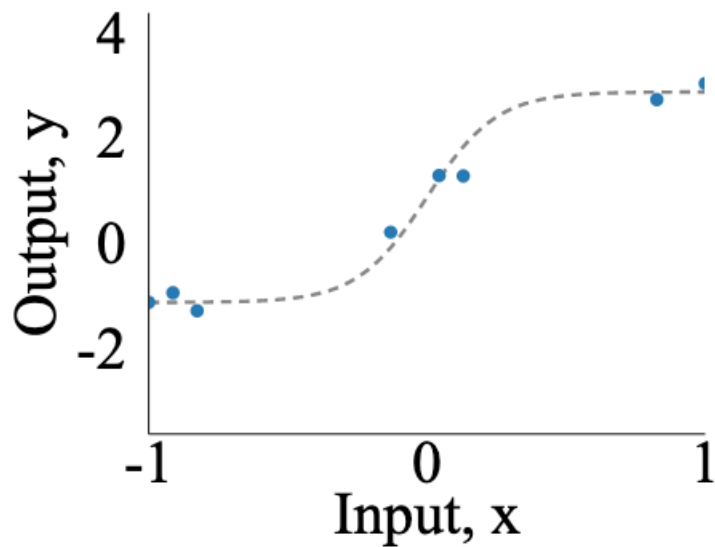
Shuffle the Samples, Then Split Into Three Equal Groups

Illustrative Regression Example · Dashed Grey: Underlying Pattern

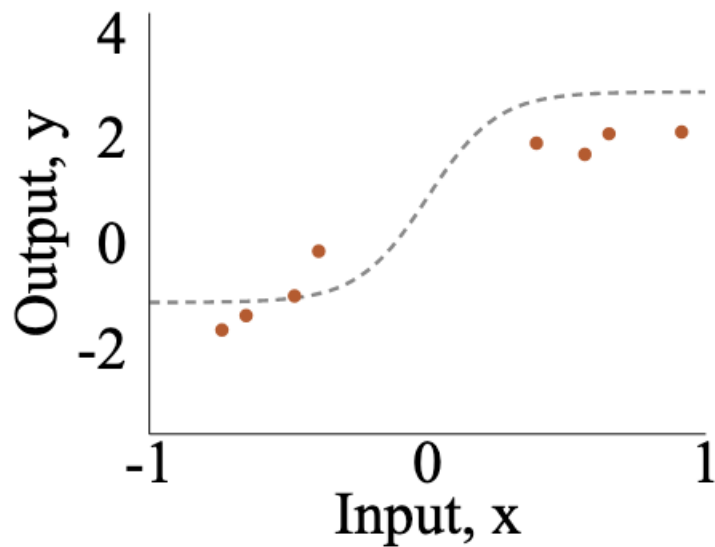
Overfitting: Perfect Training Fits

14 11 24 22 13 2 1 3 7 5 8 20 17 19 4 23 15 9 18 10 16 21 6 12

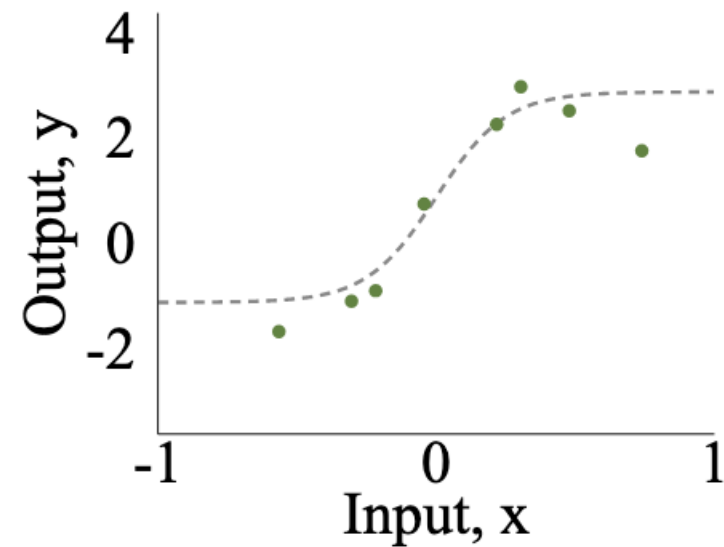
T1 · 8 Samples



T2 · 8 Samples



T3 · 8 Samples

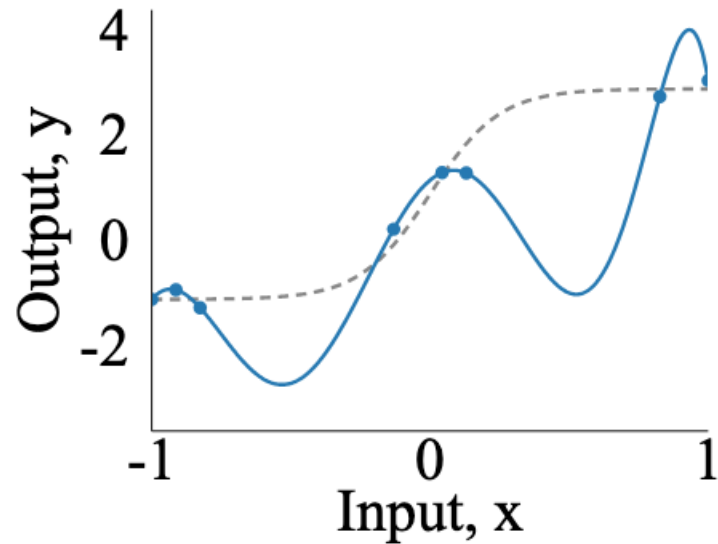


Illustrative Regression Example · Dashed Grey: Underlying Pattern

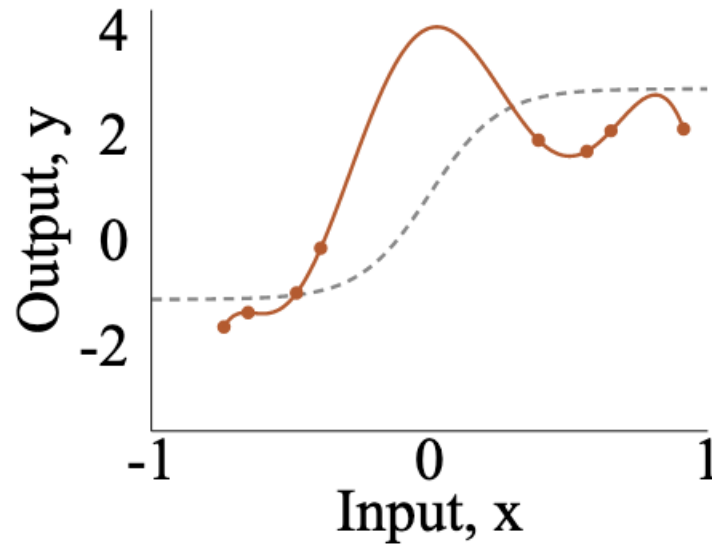
Overfitting: Perfect Training Fits

14 11 24 22 13 2 1 3 7 5 8 20 17 19 4 23 15 9 18 10 16 21 6 12

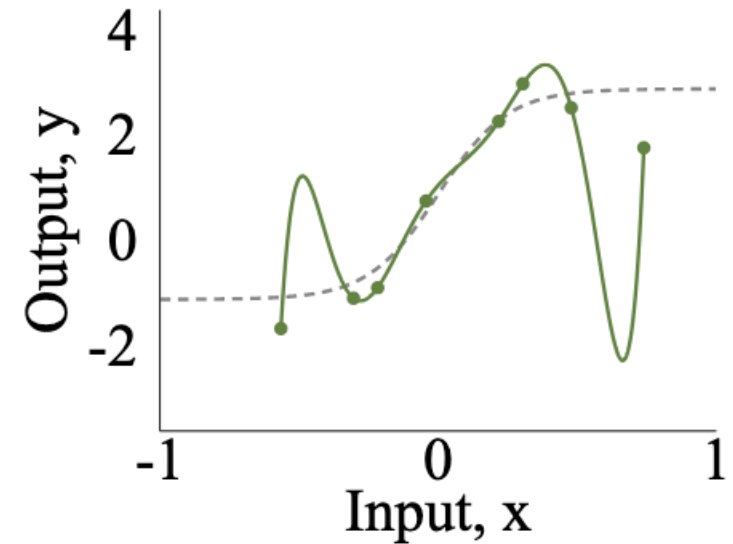
T1 · 8 Samples



T2 · 8 Samples

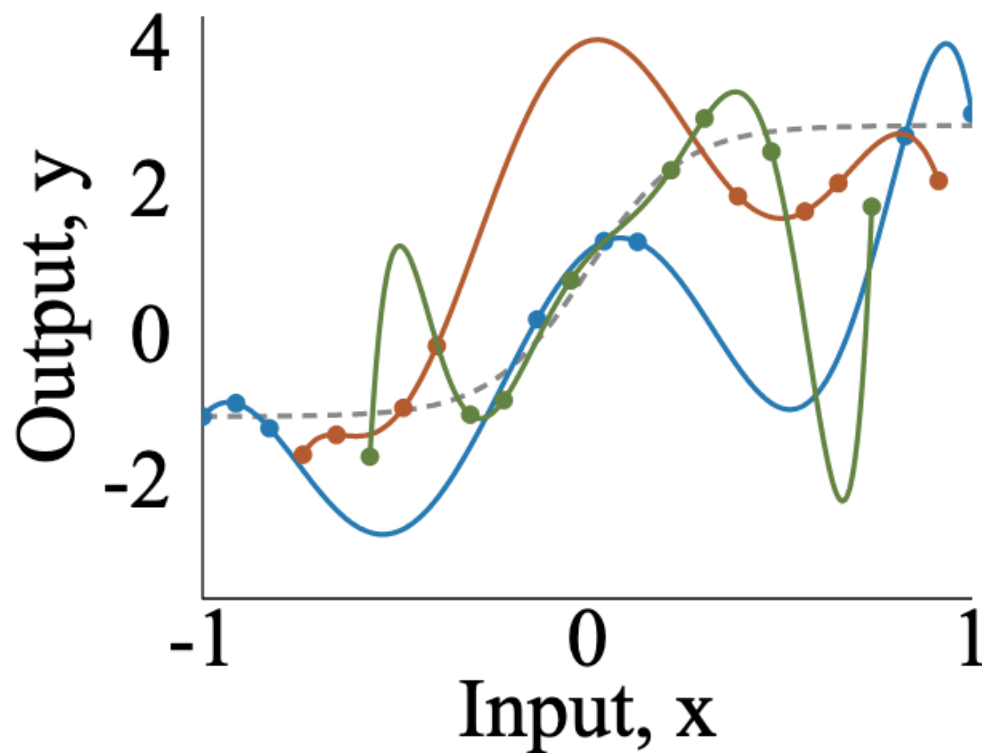


T3 · 8 Samples



Illustrative Regression Example · Dashed Grey: Underlying Pattern

Learn the Pattern, Not the Noise

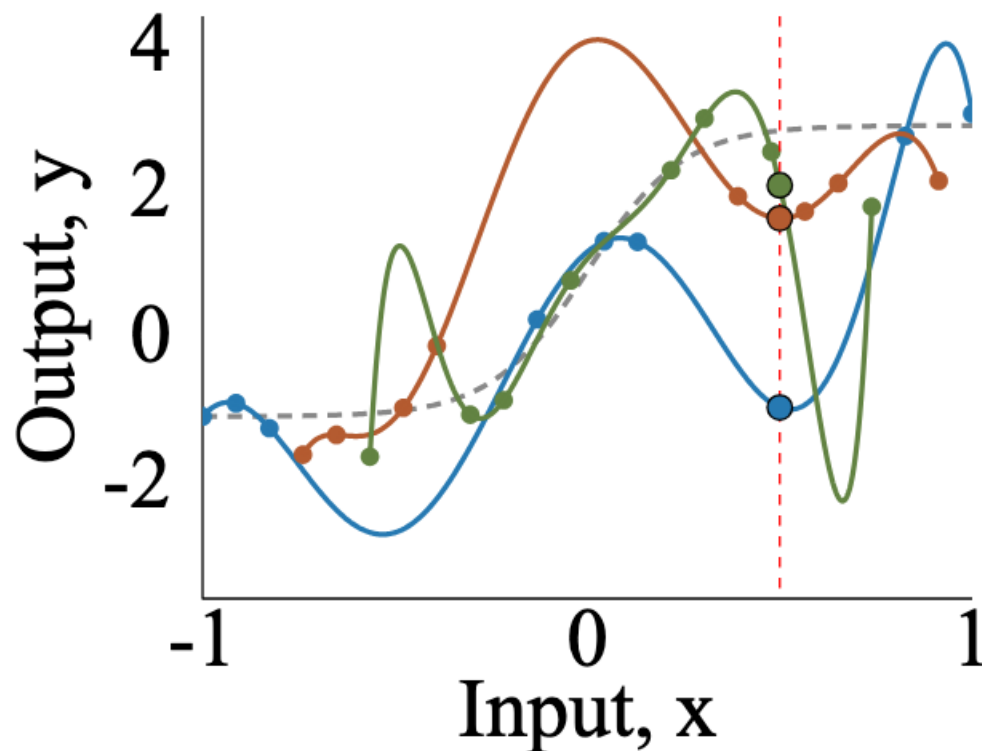


$$\hat{y} = w_0 + w_1x + w_2x^2 + \cdots + w_7x^7$$

Same Model Form
Different Learned Parameters

Illustrative Regression Example · Dashed Grey: Underlying Pattern

Learn the Pattern, Not the Noise



$$\hat{y} = w_0 + w_1x + w_2x^2 + \dots + w_7x^7$$

Same Model Form
Different Learned Parameters

Unseen Input: $x = 0.5$

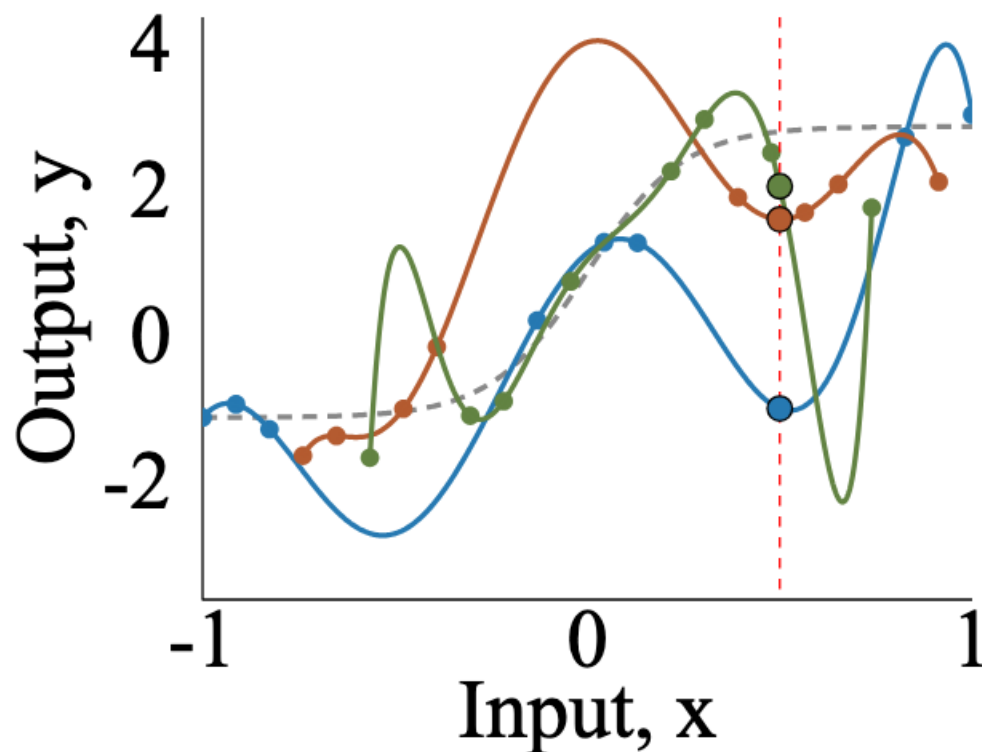
T1 Prediction: -0.88

T2 Prediction: 1.72

T3 Prediction: 2.18

Illustrative Regression Example · Dashed Grey: Underlying Pattern

Learn the Pattern, Not the Noise



$$\hat{y} = w_0 + w_1x + w_2x^2 + \dots + w_7x^7$$

Same Model Form
Different Learned Parameters

Unseen Input: $x = 0.5$

T1 Prediction: -0.88

T2 Prediction: 1.72

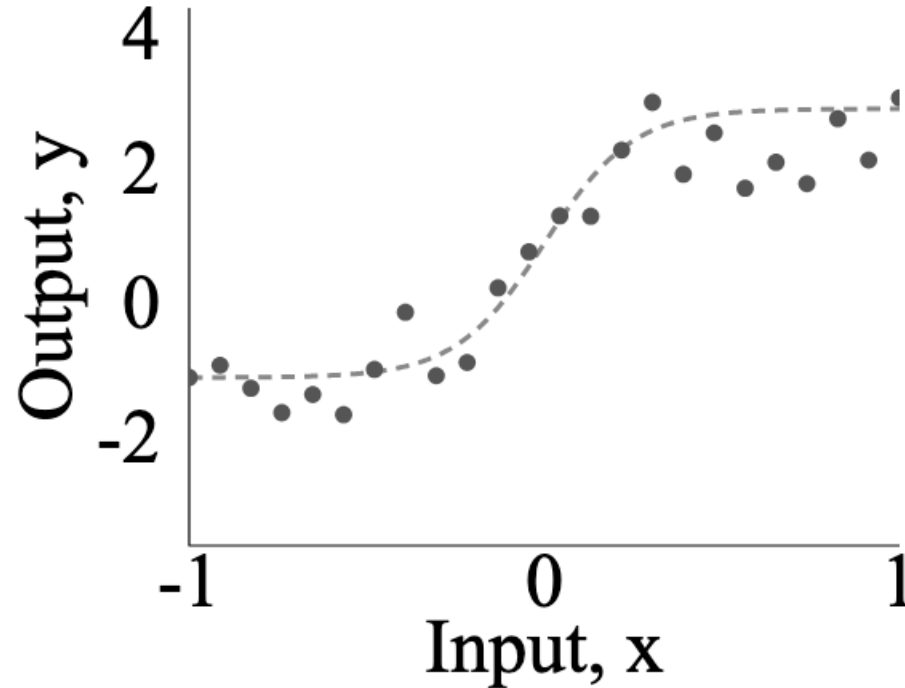
T3 Prediction: 2.18

Illustrative Regression Example · Dashed Grey: Underlying Pattern

All Three Fit Their Training Data Perfectly
Which Will Predict Well?

Underfitting: Missing the Pattern

Original Training Data · 24 Samples

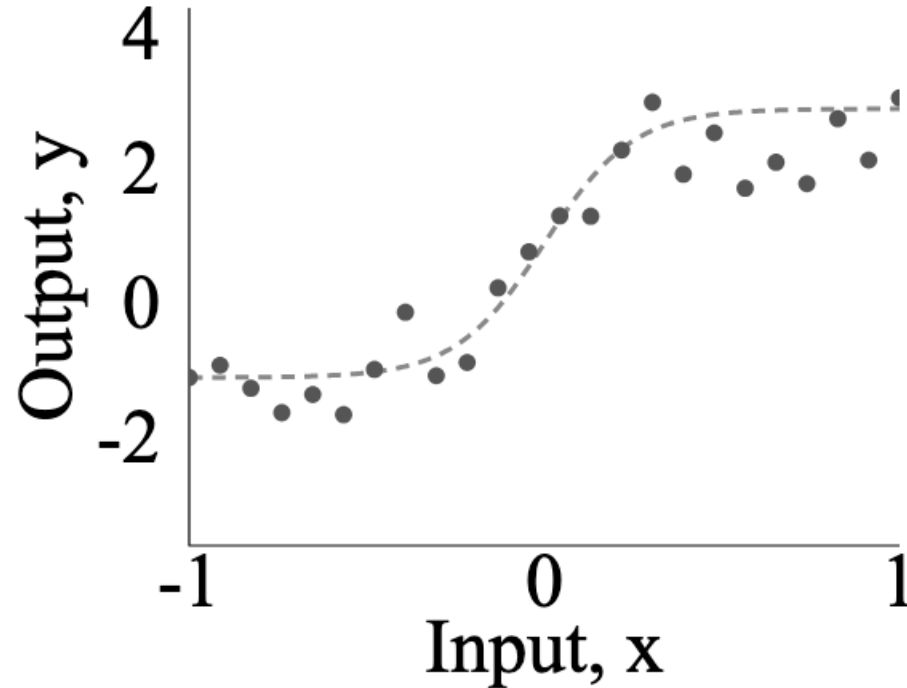


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

Illustrative Regression Example · Dashed Grey: Underlying Pattern

Underfitting: Missing the Pattern

Original Training Data · 24 Samples



14 11 24 22 13 2 1 3 7 5 8 20 17 19 4 23 15 9 18 10 16 21 6 12

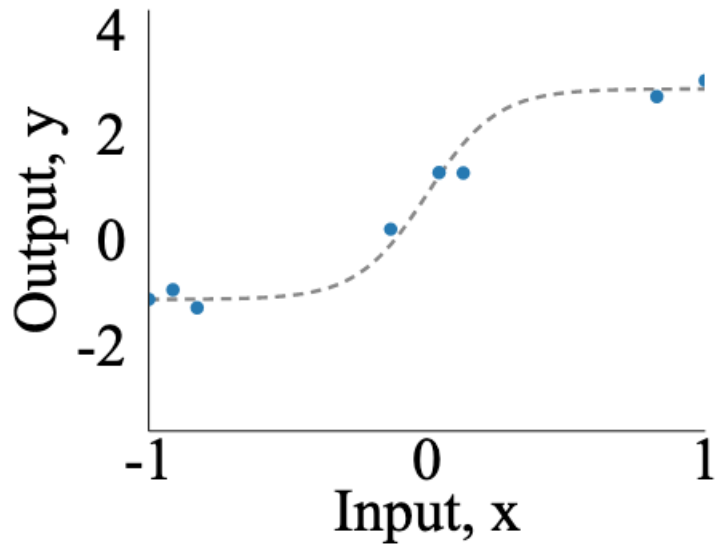
Shuffle the Samples, Then Split Into Three Equal Groups

Illustrative Regression Example · Dashed Grey: Underlying Pattern

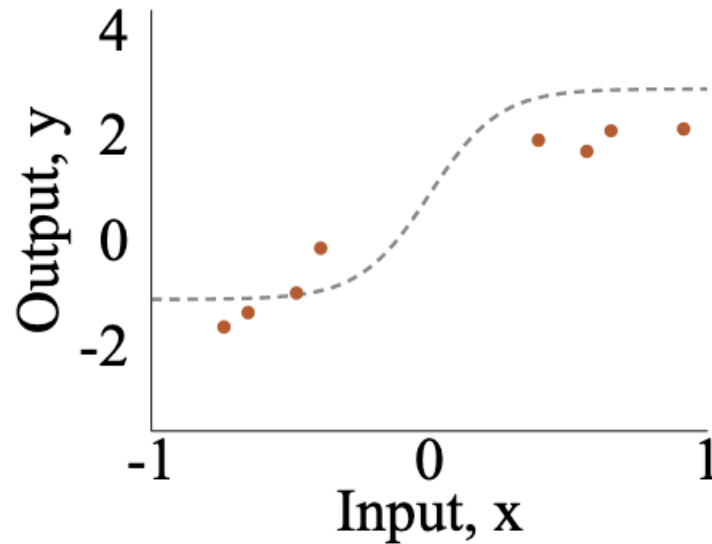
Underfitting: Missing the Pattern

14 11 24 22 13 2 1 3 7 5 8 20 17 19 4 23 15 9 18 10 16 21 6 12

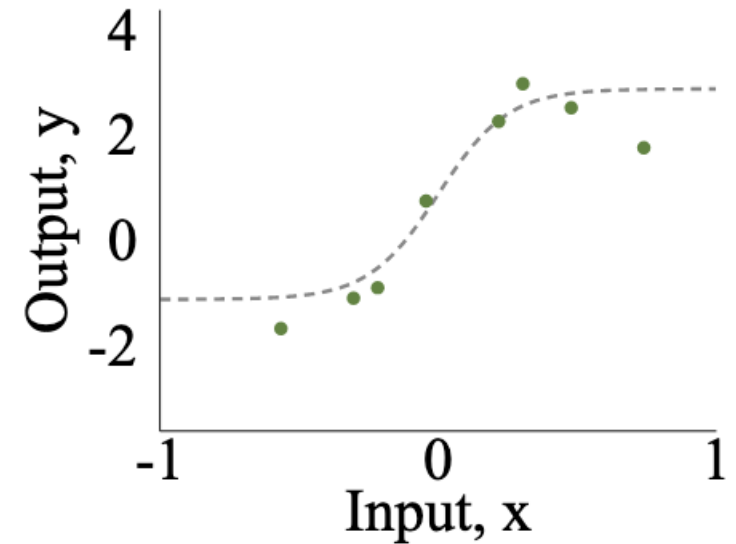
T1 · 8 Samples



T2 · 8 Samples



T3 · 8 Samples

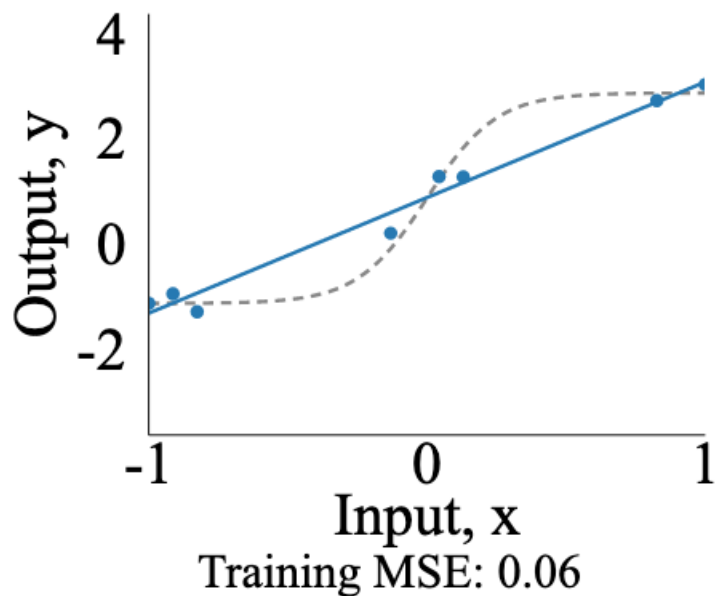


Illustrative Regression Example · Dashed Grey: Underlying Pattern

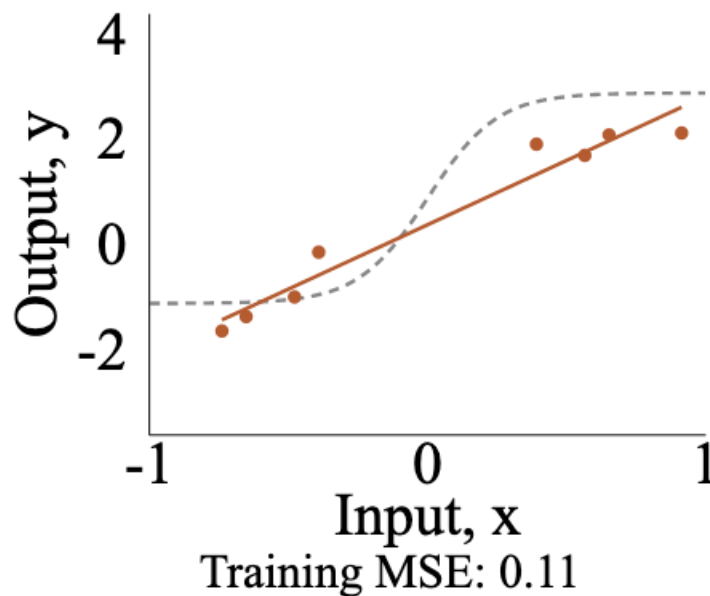
Underfitting: Missing the Pattern

14 11 24 22 13 2 1 3 7 5 8 20 17 19 4 23 15 9 18 10 16 21 6 12

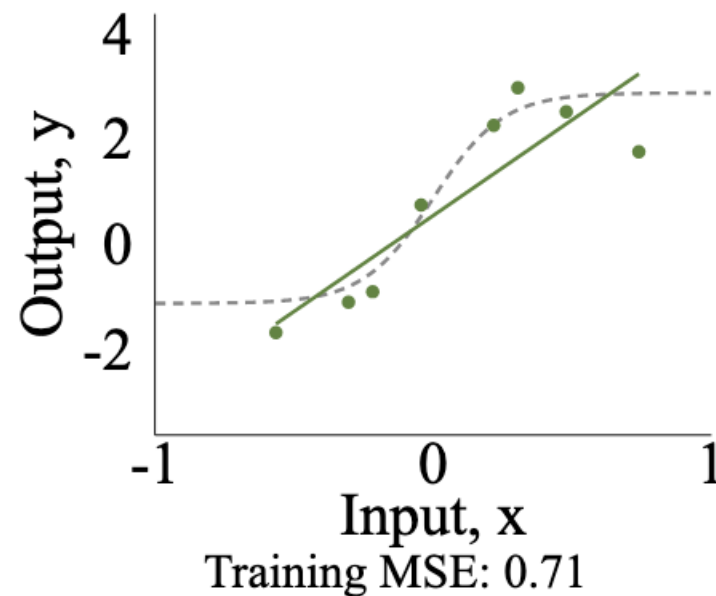
T1 · 8 Samples



T2 · 8 Samples

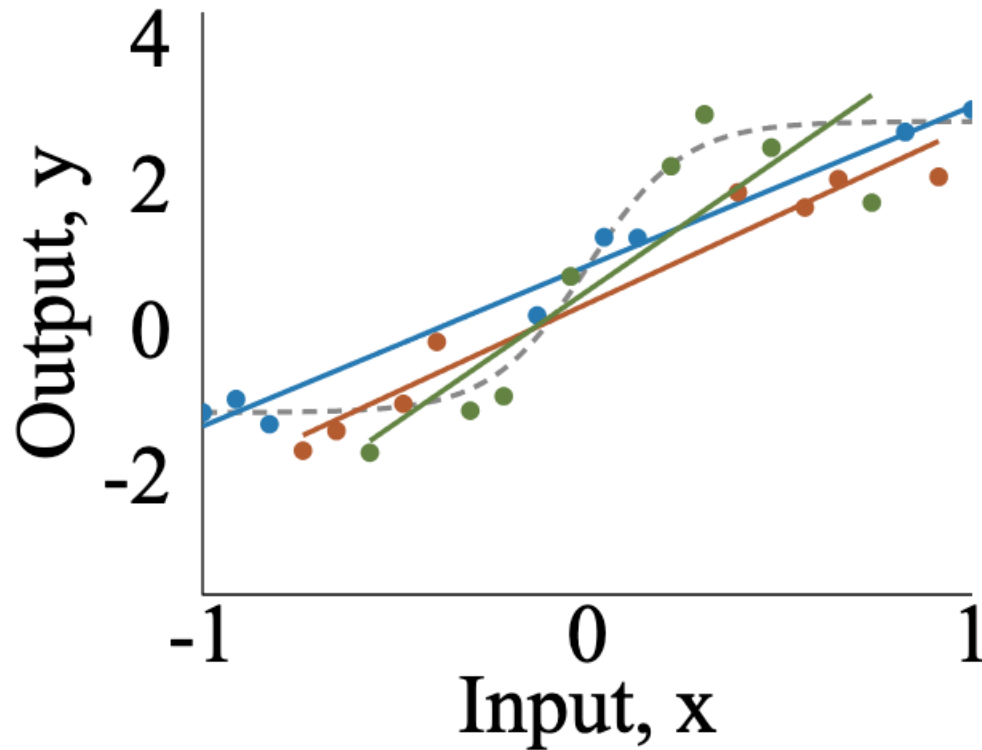


T3 · 8 Samples



Illustrative Regression Example · Dashed Grey: Underlying Pattern

Underfitting: Missing the Pattern

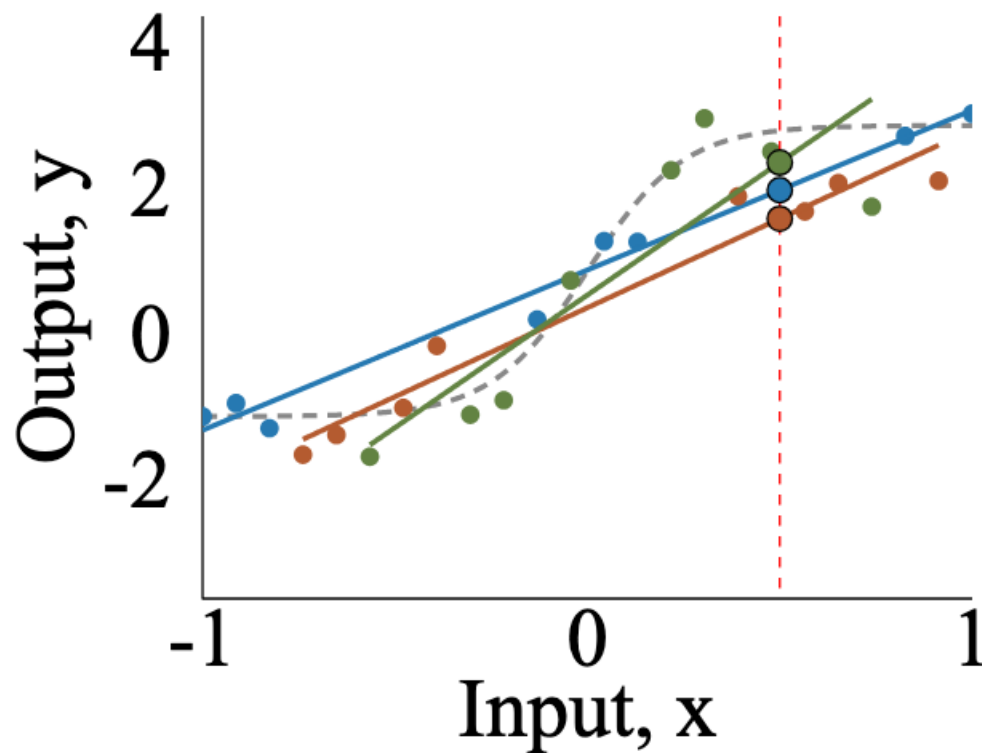


Three Straight-Line Models

All Three Miss the Curved Pattern

Illustrative Regression Example · Dashed Grey: Underlying Pattern

Underfitting: Missing the Pattern



Three Straight-Line Models

All Three Miss the Curved Pattern

Unseen Input: $x = 0.5$

T1 Prediction: 2.11

T2 Prediction: 1.72

T3 Prediction: 2.49

Illustrative Regression Example · Dashed Grey: Underlying Pattern

Similar Fits Can Still Be Poor Fits

The Model Is Too Simple

Group Exercise: Diagnose the Fit

Same Data Distribution · Achievable Error: About 1%

Error	A	B	C	D
Training	1%	15%	15%	1%
Validation	15%	16%	30%	2%

For Each Case: Underfitting, Overfitting, Both, or Neither?

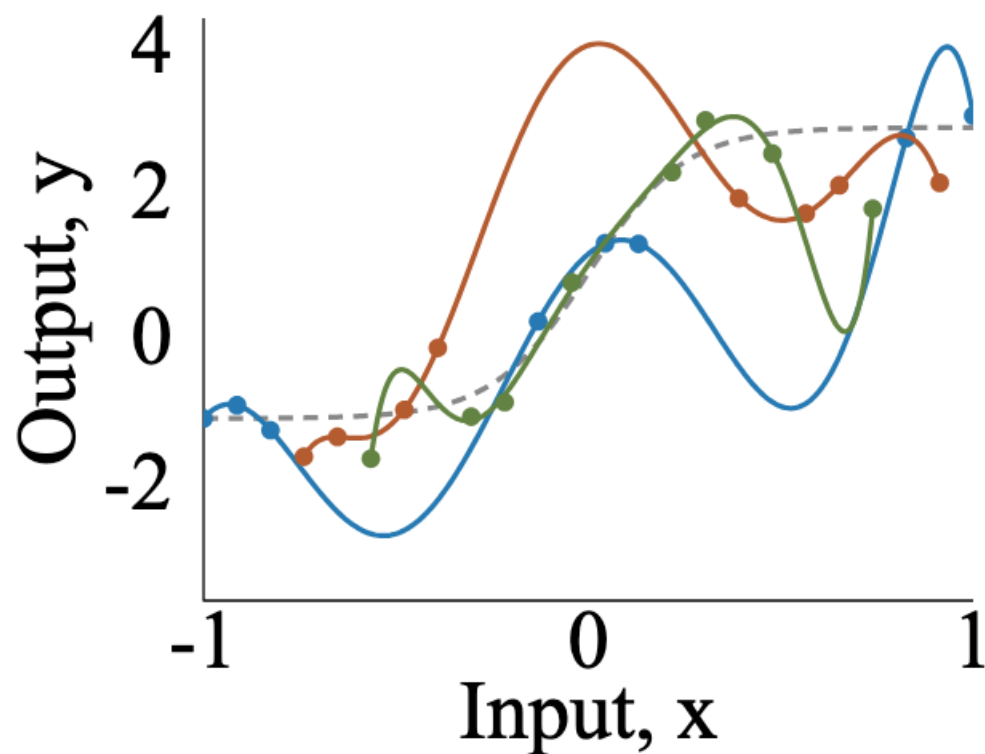
Explain Using the Training Error and the Validation Gap

One Submission per Group on Ed

Regularization Controls the Fit

$$\text{Objective} = \text{Training Loss} + \underbrace{\lambda \sum_{j=1}^7 w_j^2}_{\text{Penalty}}$$

T1 T2 T3 Same Data · Same Degree-7 Model



Vary the Penalty Strength



$\lambda = 1.00\text{e-}7$

$$\sum_{j=1}^7 w_j^2 = 130,603.14$$

$$\text{Penalty} = 1.00\text{e-}7 \times 130,603.14 \approx 0.01$$

Stronger Penalty $\rightarrow$ Smaller Weights

Pause

Weight Size and Penalty: Averages Across T1, T2, T3

Stronger Regularization Encourages Smaller Weights

Group Exercise: Propose Your Project

Read Pages 1–3 of *project_brief.pdf*
Brainstorm Ideas, Then Choose One to Submit

1. Business Question

Who would use your project, and what decision would it help them make?

2. Make the Idea Concrete

Give one example of the input and desired output

Where could you obtain the data, and would the inputs be available when the decision is made?

3. What Would You Like to Find Out?

What would you investigate with your project,
and why would the answer matter to the intended user?

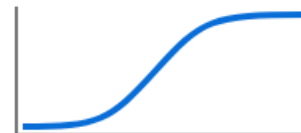
One Submission per Group on Ed

Group ID · Working Title · Three Short Answers

Supervised Learning: L3 and L4

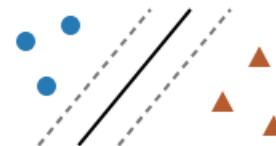
Logistic Regression

Estimate probabilities from a linear score



SVM

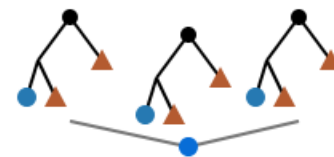
Separate classes with a margin



Random Forest

L4: Decision Trees and Forests

Learn questions, then combine many trees



kNN

L4

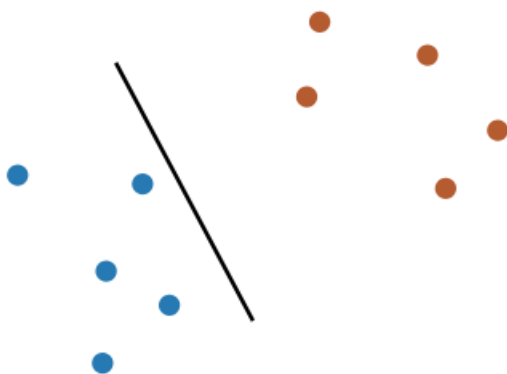
Use the labels of nearby examples



Maximum-Margin Classifiers

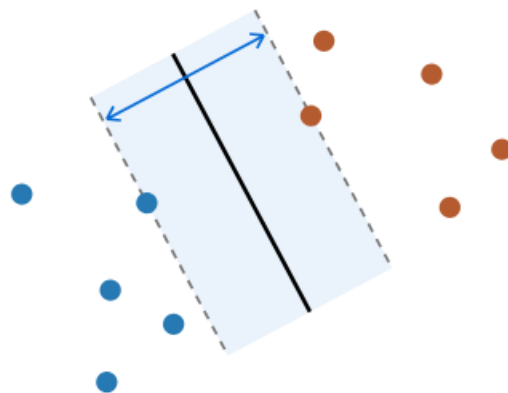
Perceptron / Adaline

Learn a Linear Decision Boundary



Maximum-Margin Classifier

Choose the Widest Margin

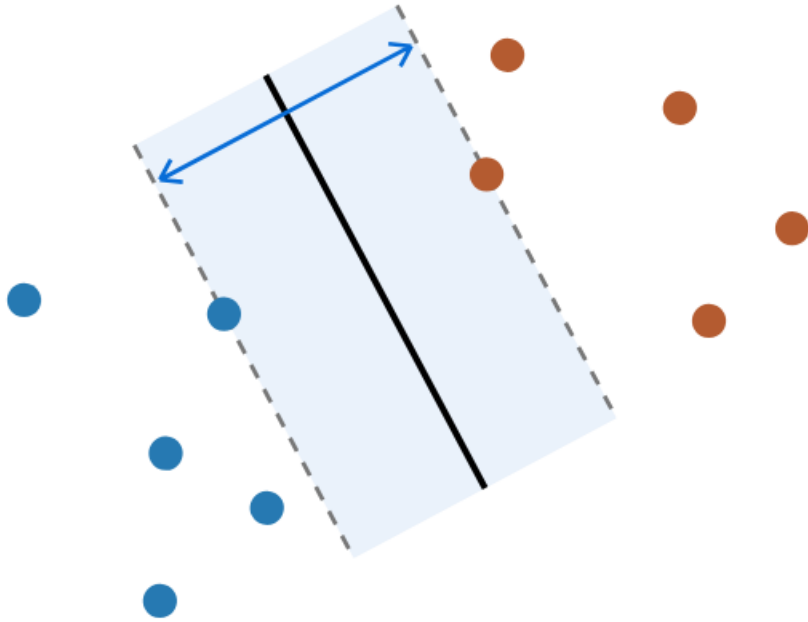


Choose the Boundary with the Widest Margin

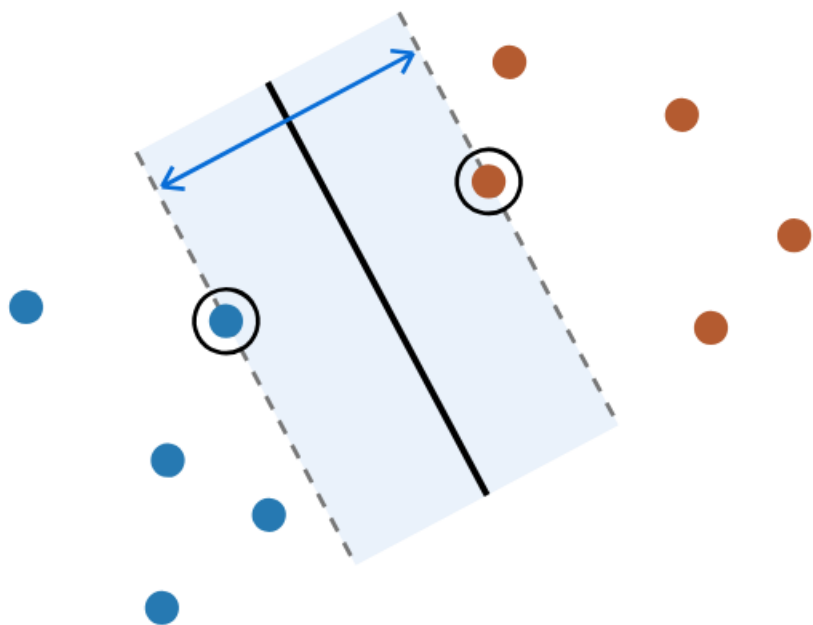
1963 · Vapnik & Lerner: Generalized-Portrait Foundations

1992 · Kernel Extension · 1995 · Soft-Margin Formulation

Support Vectors



Support Vectors

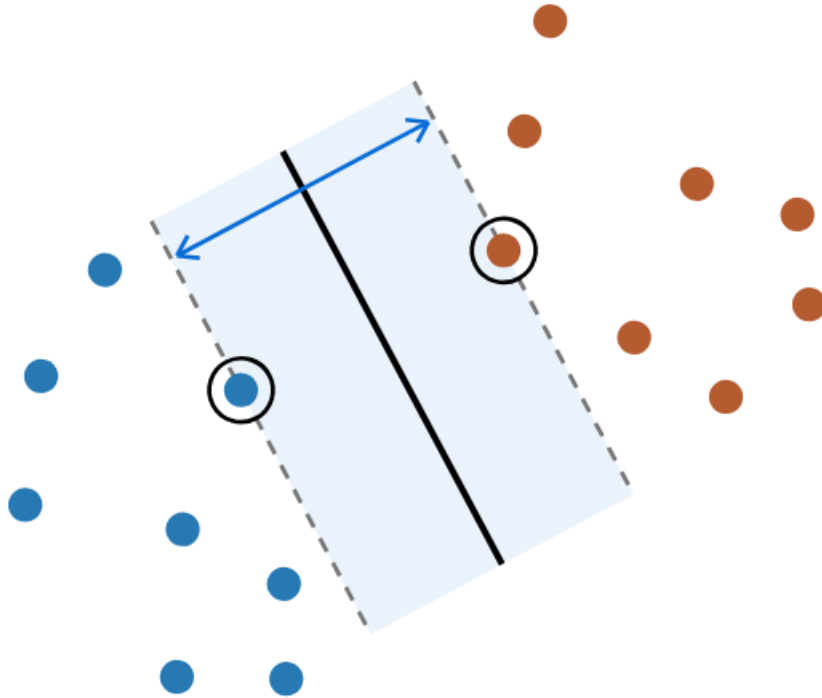


Support Vectors

The samples touching the edges of the margin

They determine the boundary

Support Vectors



Support Vectors

The samples touching the edges of the margin

They determine the boundary

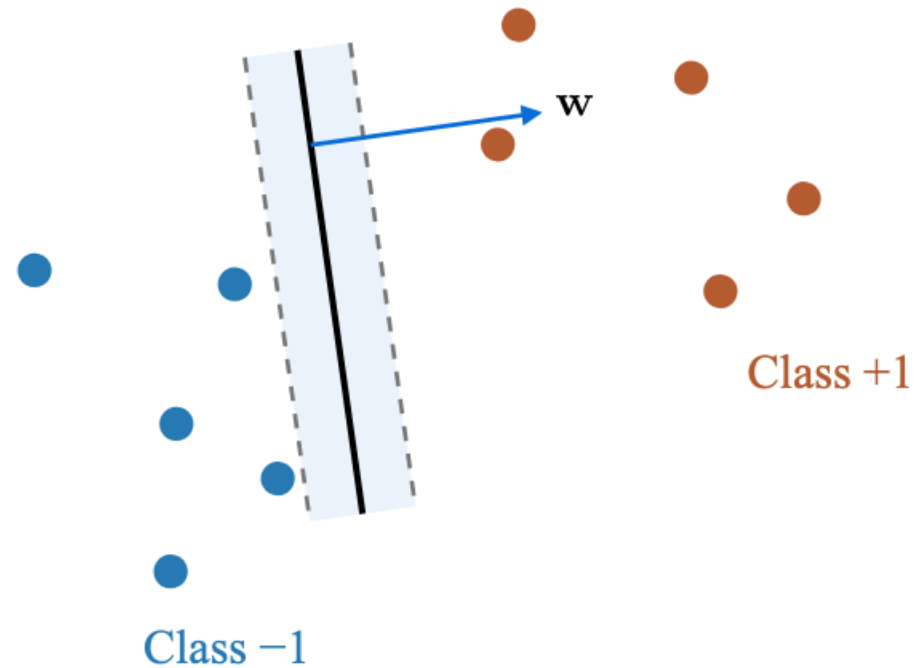
More Samples, Same Boundary

Add samples outside the margin, on the correct side

The boundary and margin stay fixed

The Closest Samples Determine the Widest Margin

Which Street Should We Choose?



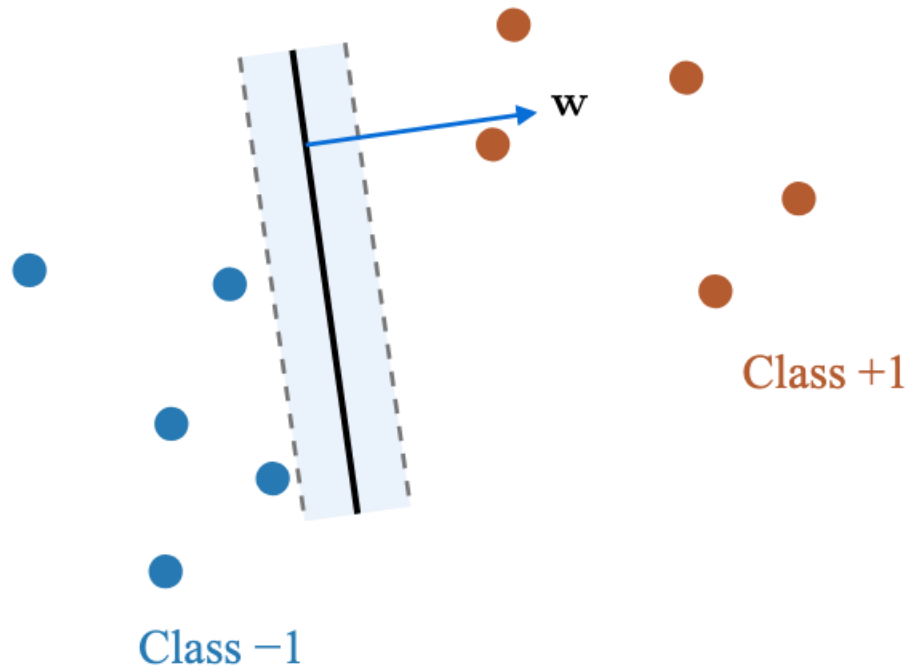
Change w , Keep b Fixed

$$z = \mathbf{w}^\top \mathbf{x} + b$$

w Points Across the Street

b Sets Its Position

Which Street Should We Choose?



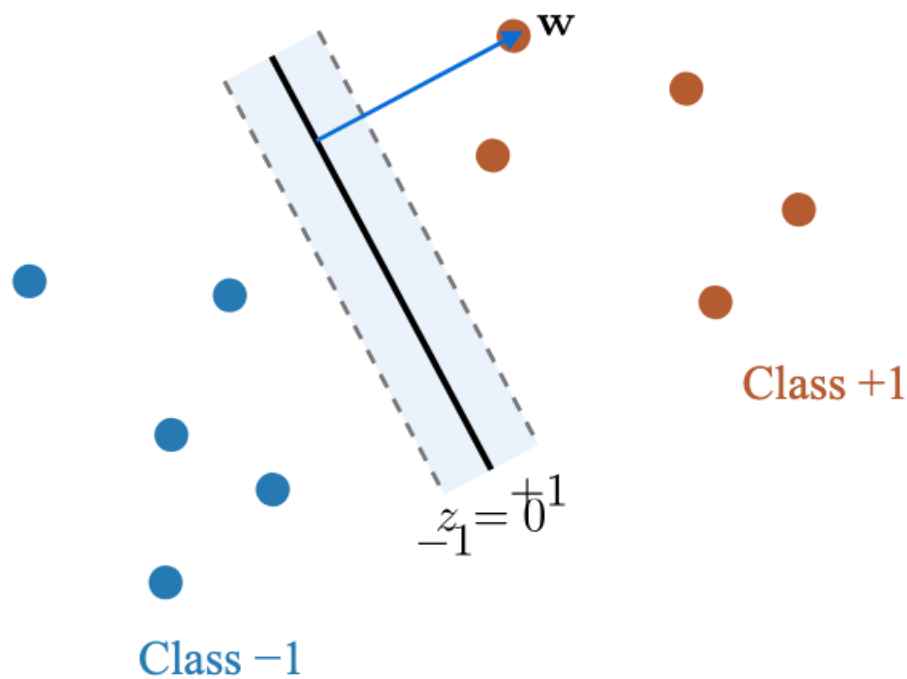
Change b , Keep w Fixed

$$z = \mathbf{w}^\top \mathbf{x} + b$$

w Points Across the Street

b Sets Its Position

Which Street Should We Choose?



Choose Its Orientation and Position

$$z = \mathbf{w}^\top \mathbf{x} + b$$

$\mathbf{w}$ Points Across the Street

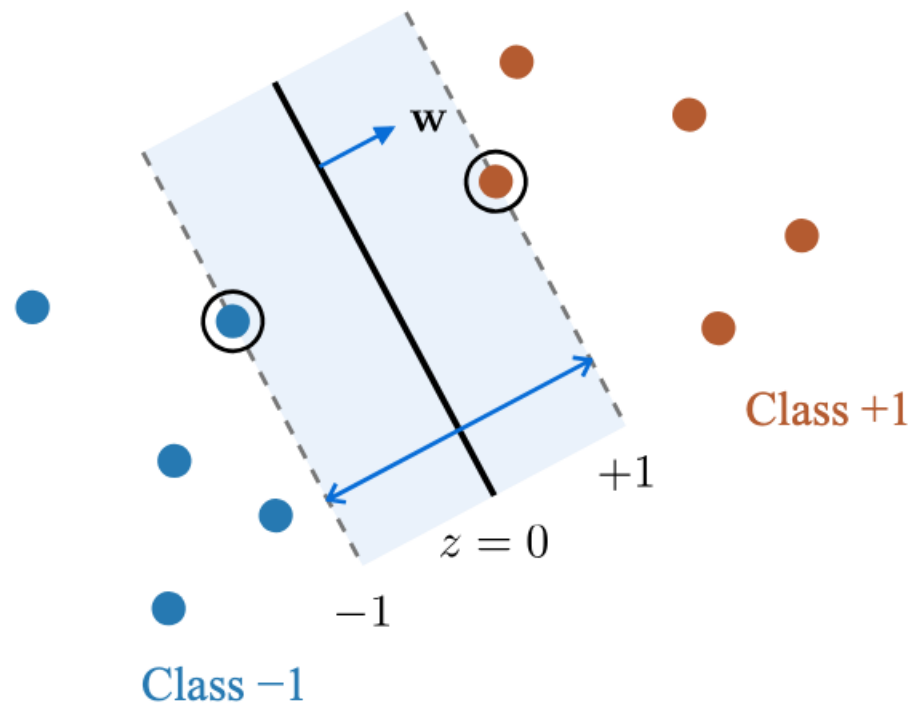
b Sets Its Position

Keep Samples Outside Their Edges

$$y_i z_i \geq 1 \quad (y_i \in \{-1, +1\})$$

Fix the Edge Scores at -1 and $+1$

Which Street Should We Choose?



Choose Its Orientation and Position

$$z = \mathbf{w}^\top \mathbf{x} + b$$

w Points Across the Street

b Sets Its Position

Keep Samples Outside Their Edges

$$y_i z_i \geq 1 \quad (y_i \in \{-1, +1\})$$

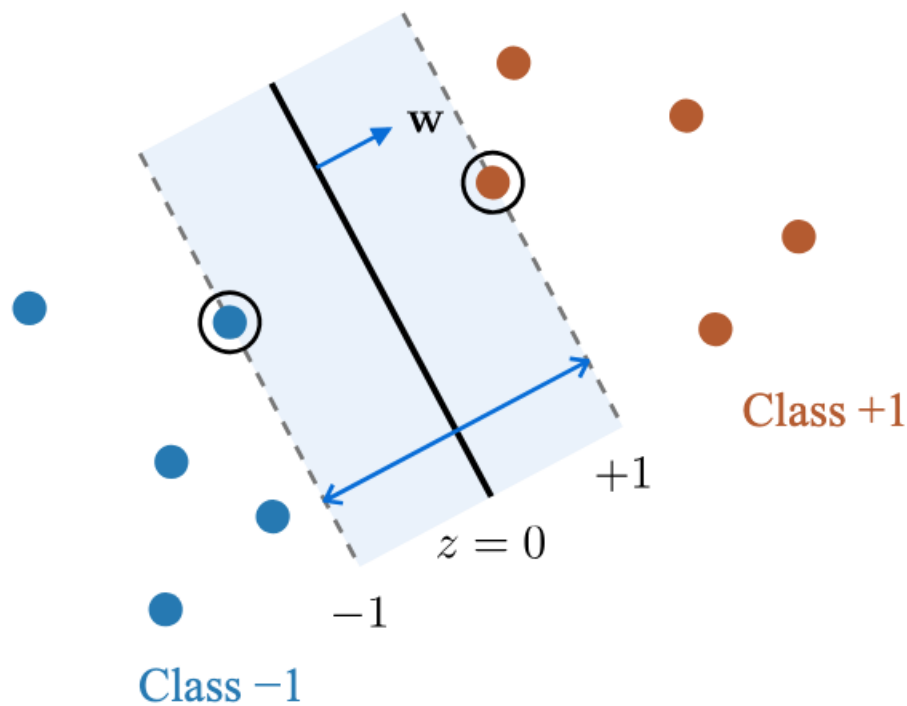
Fix the Edge Scores at -1 and $+1$

Find the Widest Valid Street

$$\text{Width} = \frac{2}{\|\mathbf{w}\|}$$

Smaller Weight Norm, Wider Margin

Which Street Should We Choose?



Choose Its Orientation and Position

$$z = \mathbf{w}^\top \mathbf{x} + b$$

$\mathbf{w}$ Points Across the Street

b Sets Its Position

Keep Samples Outside Their Edges

$$y_i z_i \geq 1 \quad (y_i \in \{-1, +1\})$$

Fix the Edge Scores at -1 and $+1$

Find the Widest Valid Street

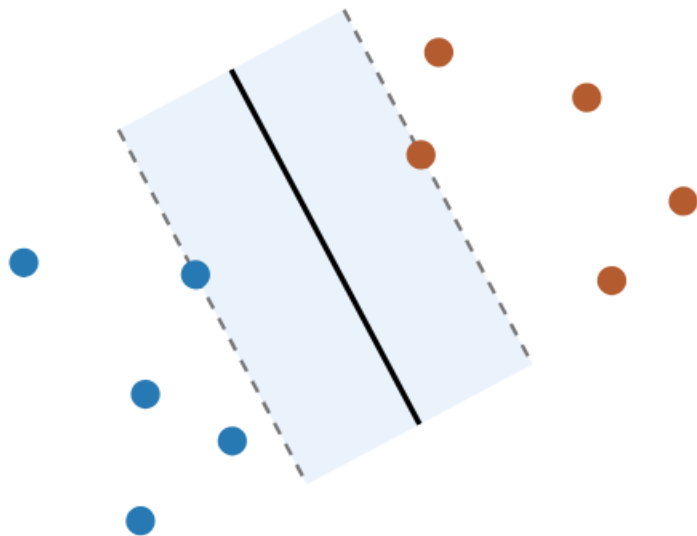
$$\text{Width} = \frac{2}{\|\mathbf{w}\|}$$

Smaller Weight Norm, Wider Margin

The Hard-Margin Objective

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \text{for all } i$$

When One Sample Squeezes the Margin



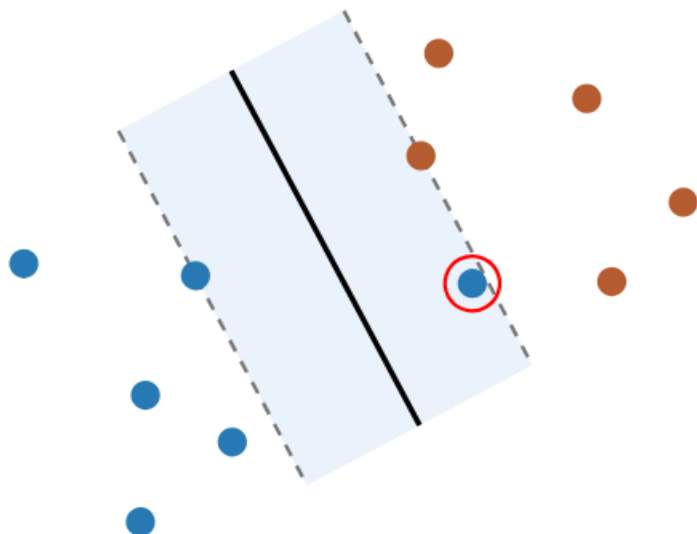
Hard Margin: Every Constraint Must Hold

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \text{for all } i$$

When One Sample Squeezes the Margin

One Unusual Sample

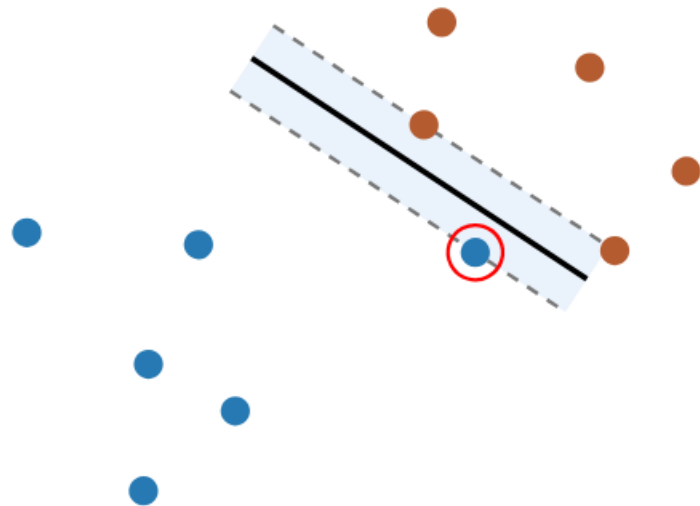
A Blue Sample Near the Orange Class



Hard Margin: Every Constraint Must Hold

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \text{for all } i$$

When One Sample Squeezes the Margin



One Unusual Sample

A Blue Sample Near the Orange Class

Insist on Perfect Separation

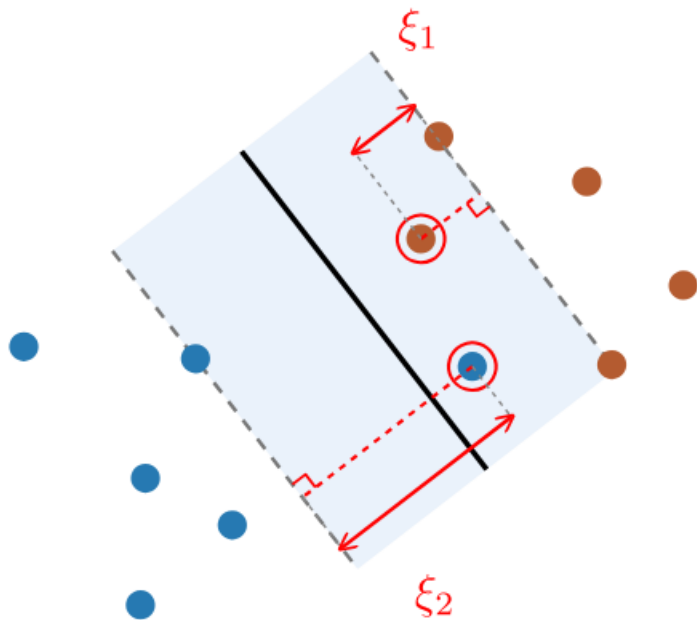
The Boundary Shifts

The Margin Becomes Much Narrower

Hard Margin: Every Constraint Must Hold

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \text{for all } i$$

When One Sample Squeezes the Margin



One Unusual Sample

A Blue Sample Near the Orange Class

Insist on Perfect Separation

The Boundary Shifts

The Margin Becomes Much Narrower

Allow Some Margin Violations

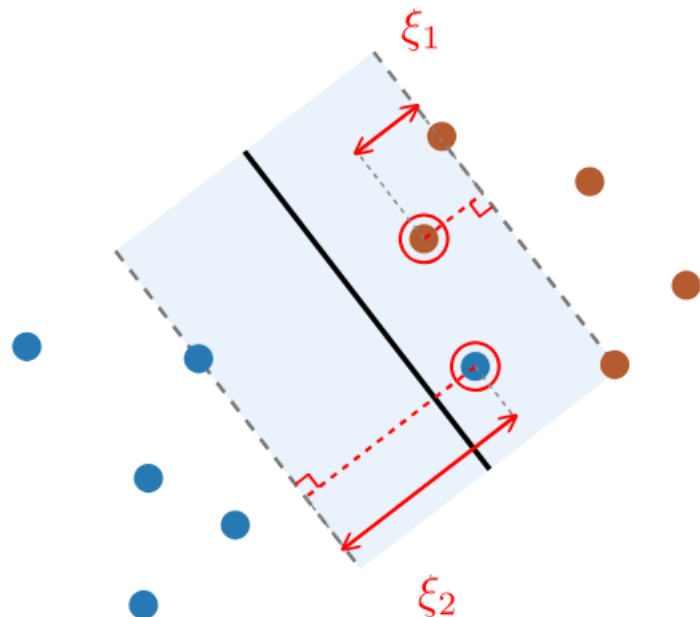
ξ_i : Slack Variable

Measures the Margin Violation

Soft Margin: Allow Violations, Add a Penalty

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i \quad \text{subject to} \quad y_i z_i \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i$$

When One Sample Squeezes the Margin



Recall Regularization

Larger λ : Stronger Weight Penalty

For the SVM

Smaller C : Stronger Regularization

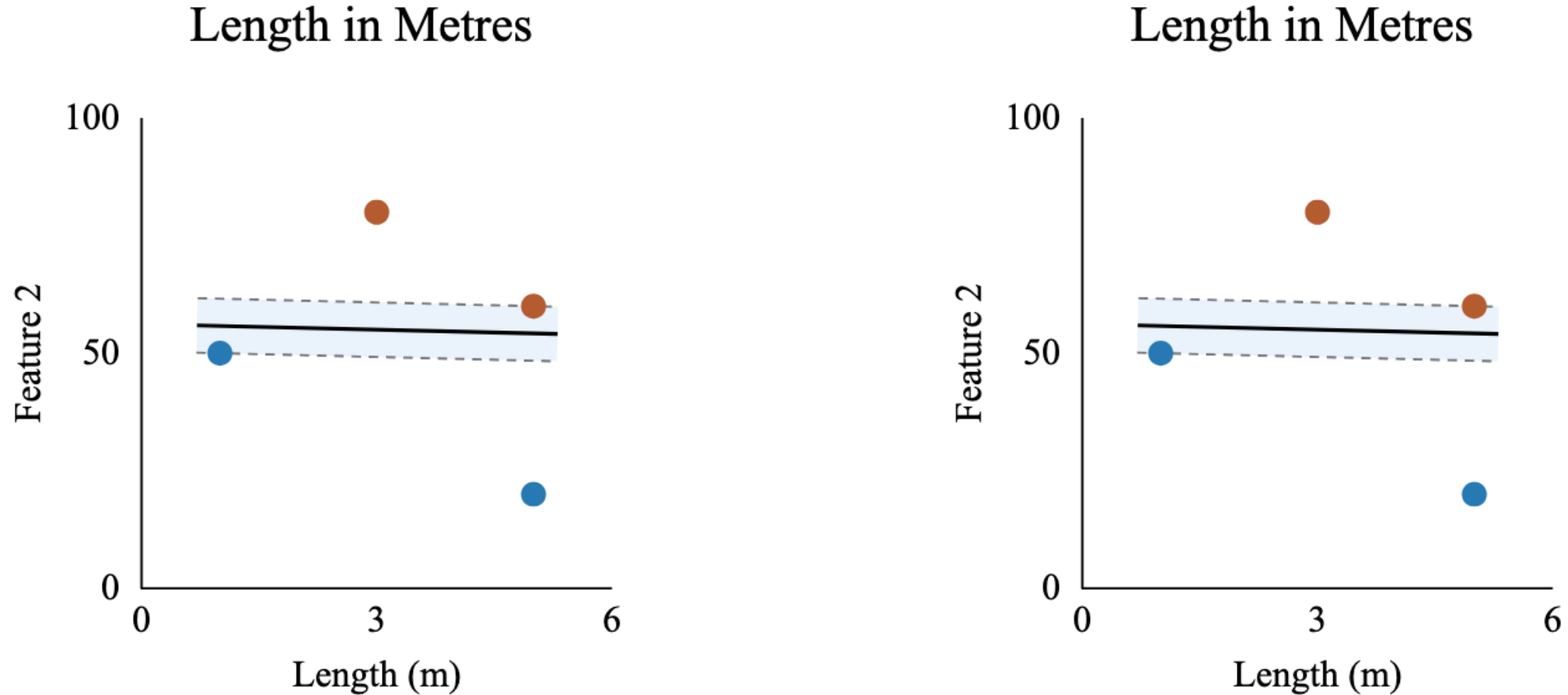
$\lambda \uparrow \iff C \downarrow$

Same Trade-off, Opposite Directions

Divide the Objective by C : Same Minimizer

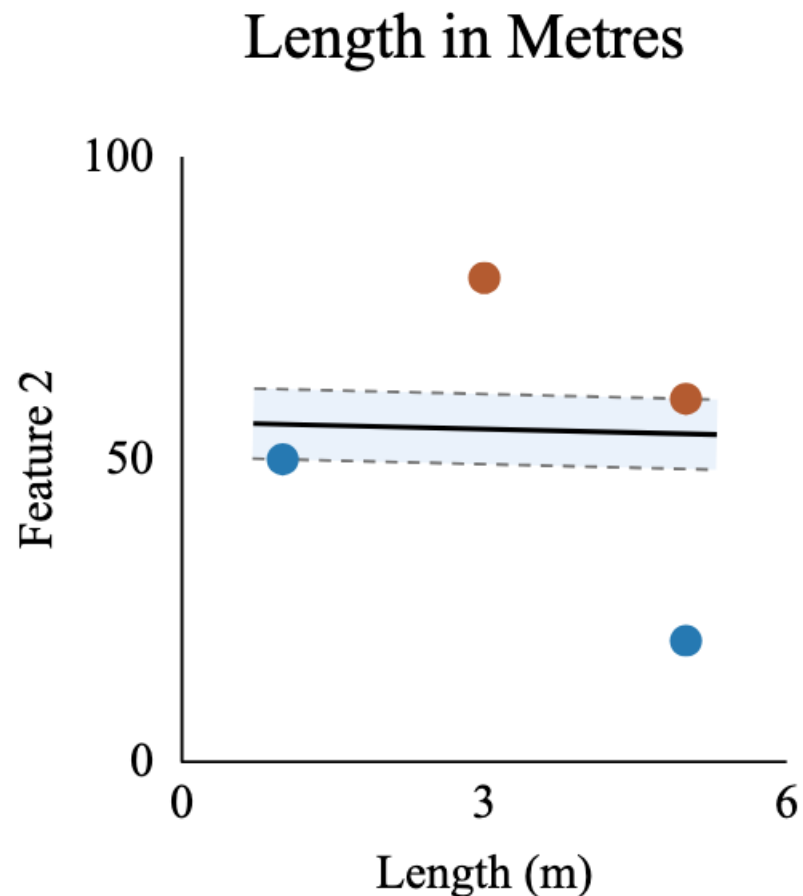
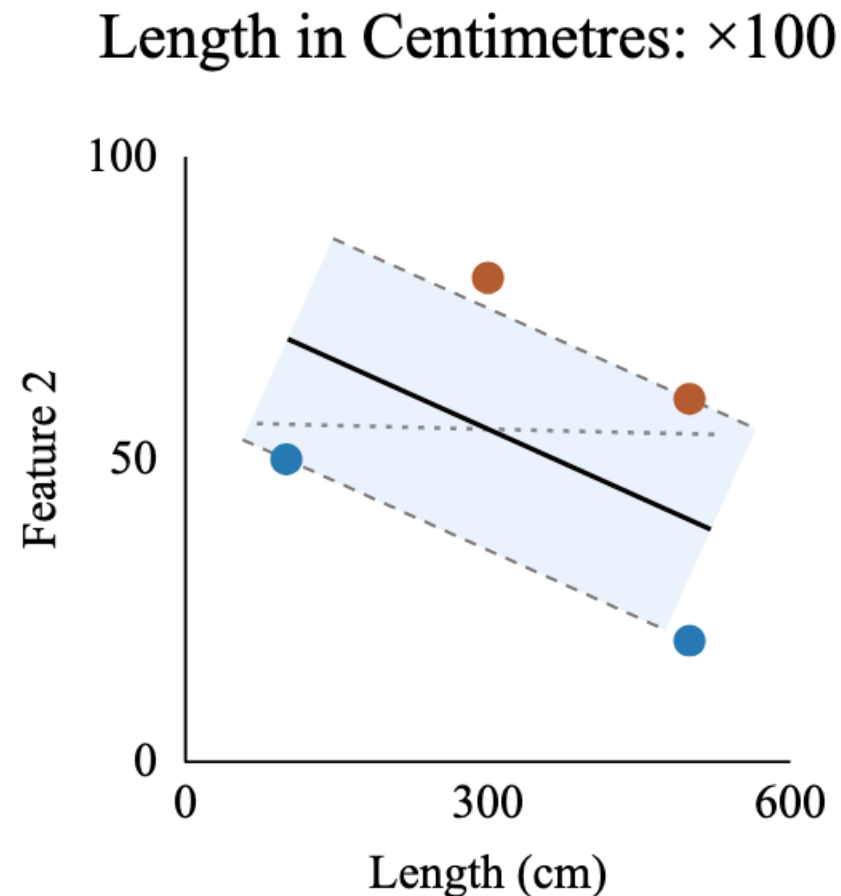
$$\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i \xrightarrow{\div C} \sum_i \xi_i + \frac{1}{2C} \|\mathbf{w}\|^2$$

Why Scale the Features?



Same Samples · Same Labels · Only the Units Change

Why Scale the Features?

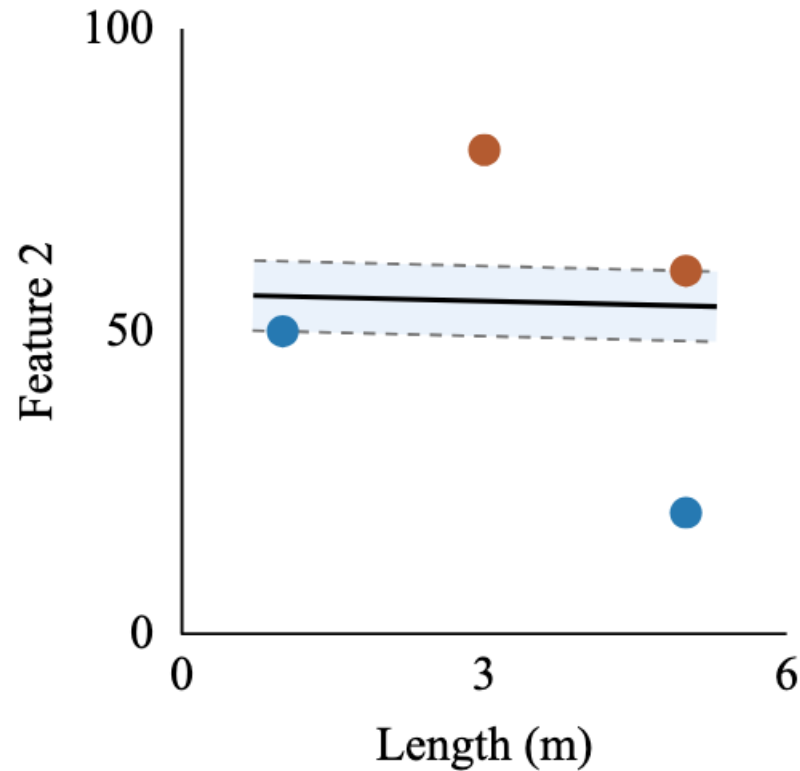


Same Samples · Same Labels · Only the Units Change

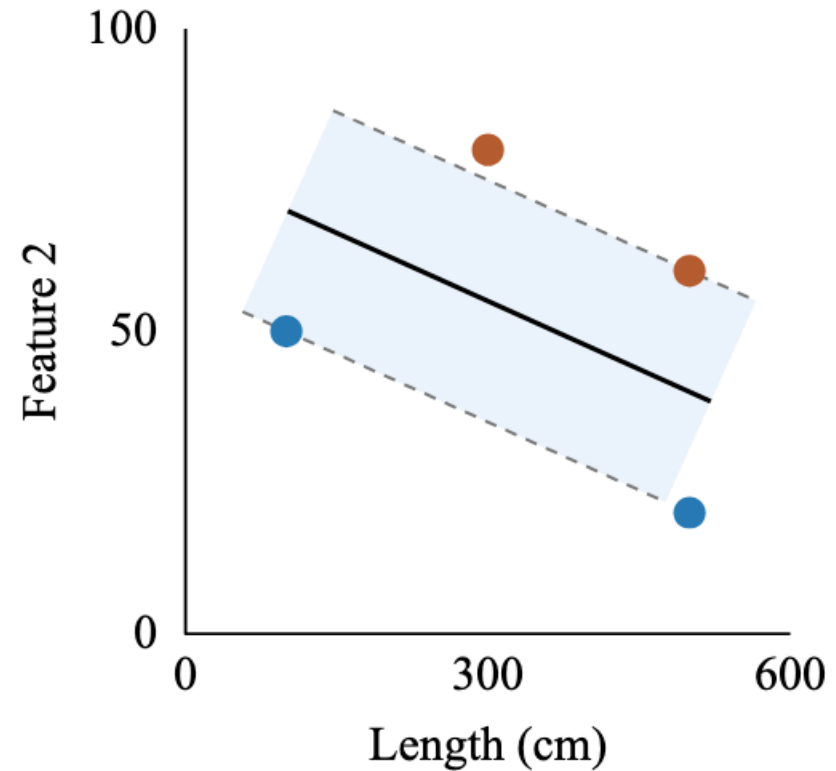
Changing Units Can Change the Maximum-Margin Boundary

Standardize Before Training

Starting from Metres

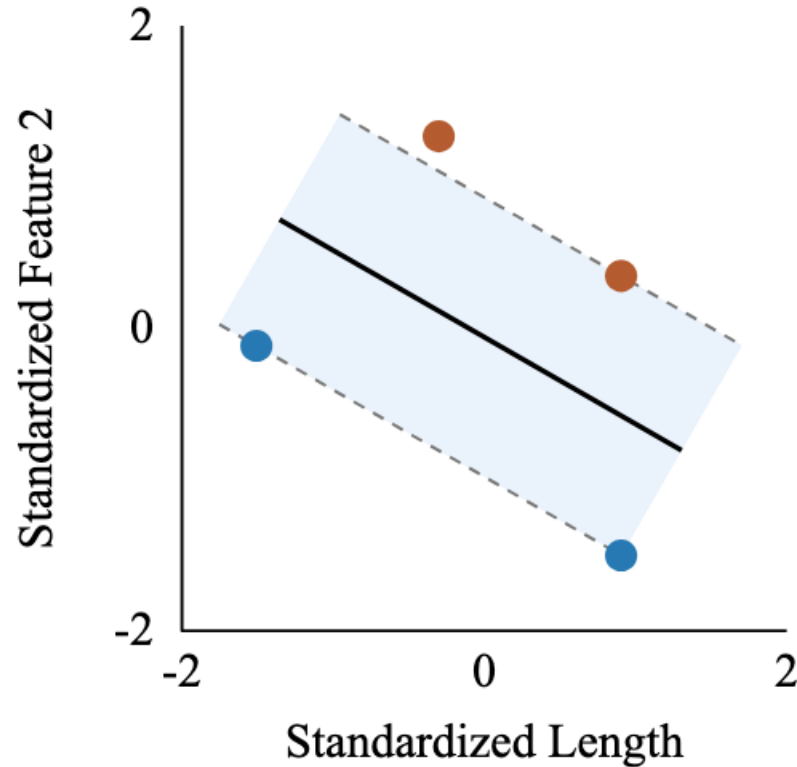


Starting from Centimetres

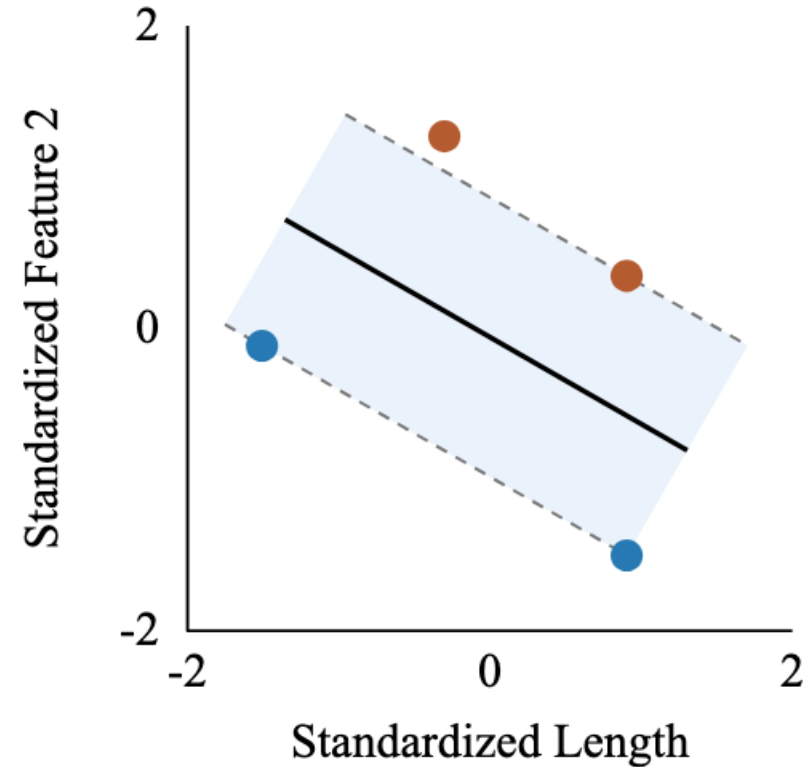


Standardize Before Training

Standardized from Metres



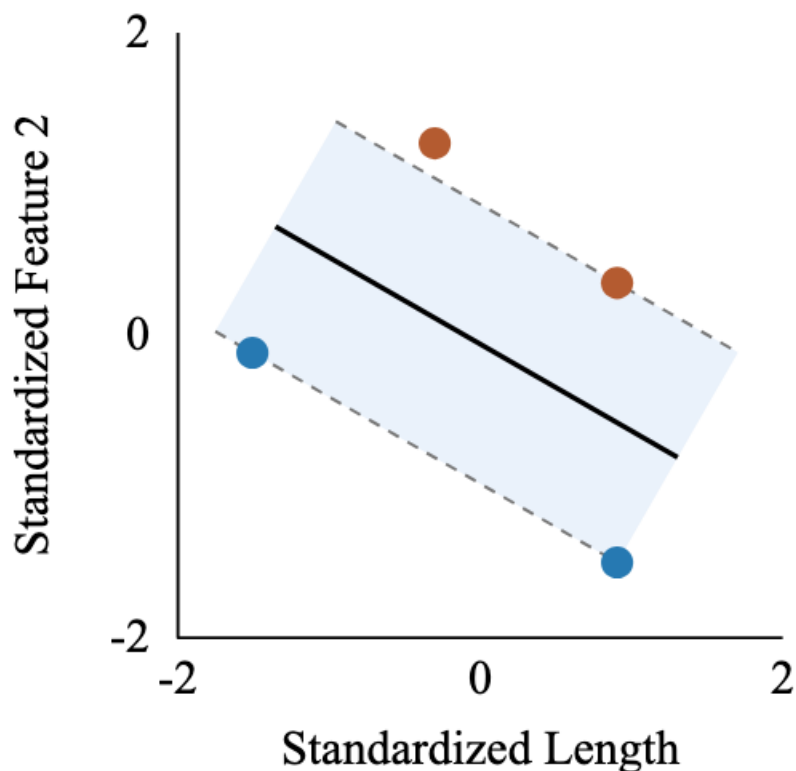
Standardized from Centimetres



Standardizing Either Unit System Gives the Same Result

Standardize Before Training

Standardized from Metres



Standardize Each Feature

$$x_j^{\text{std}} = \frac{x_j - \mu_{\text{train},j}}{\sigma_{\text{train},j}}$$

```
from sklearn.preprocessing import StandardScaler

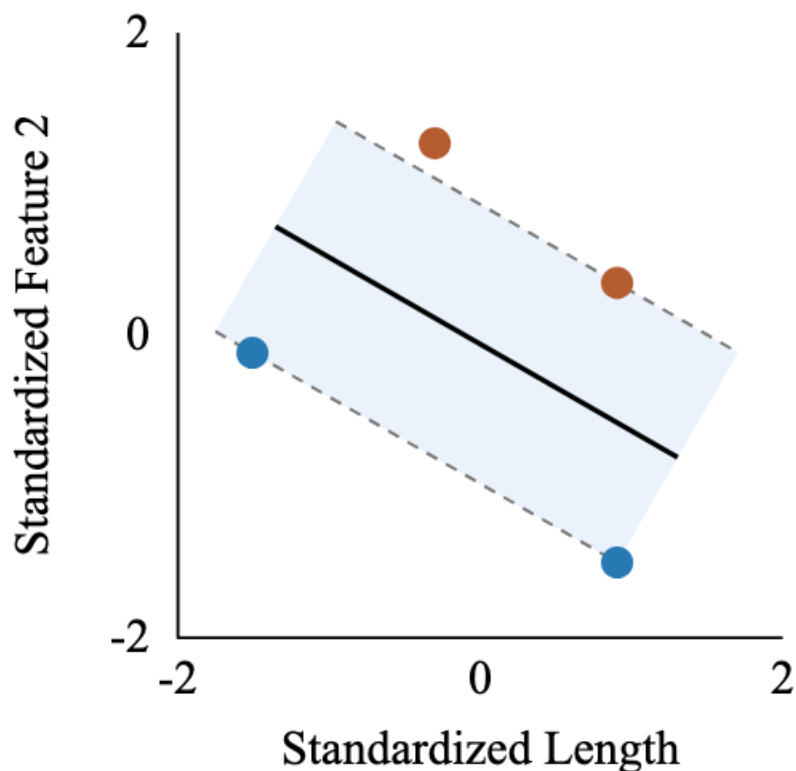
sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
X_test_std = sc.transform(X_test)
```

Standardizing Either Unit System Gives the Same Result

After the Train/Test Split: [support_vector_machines_sklearn.ipynb](#)

Standardize Before Training

Standardized from Metres



Standardize Each Feature

$$x_j^{\text{std}} = \frac{x_j - \mu_{\text{train},j}}{\sigma_{\text{train},j}}$$

```
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
X_test_std = sc.transform(X_test)
```

Fit on Training Data Only

Reuse the Scaler for Test Data

Standardizing Either Unit System Gives the Same Result

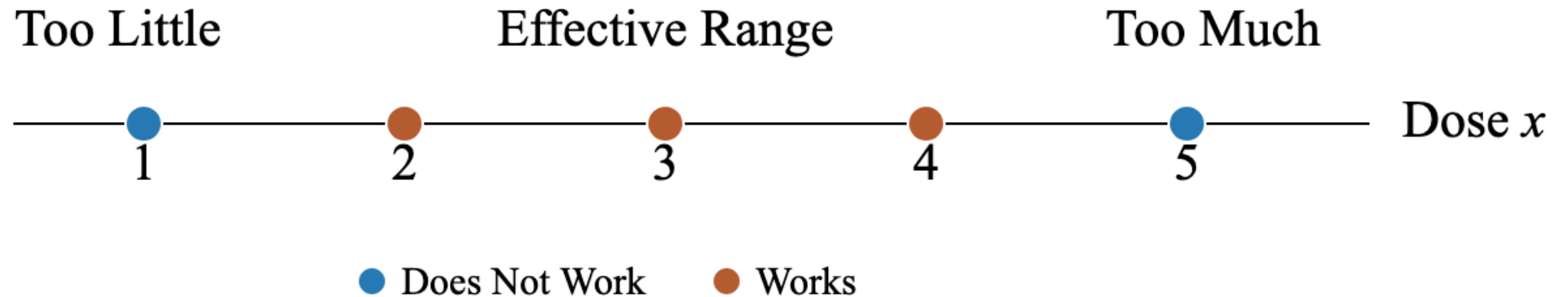
After the Train/Test Split: [support_vector_machines_sklearn.ipynb](#)

maib7002.dirk.hk/L3.zip

Demo: support_vector_machines_sklearn.ipynb

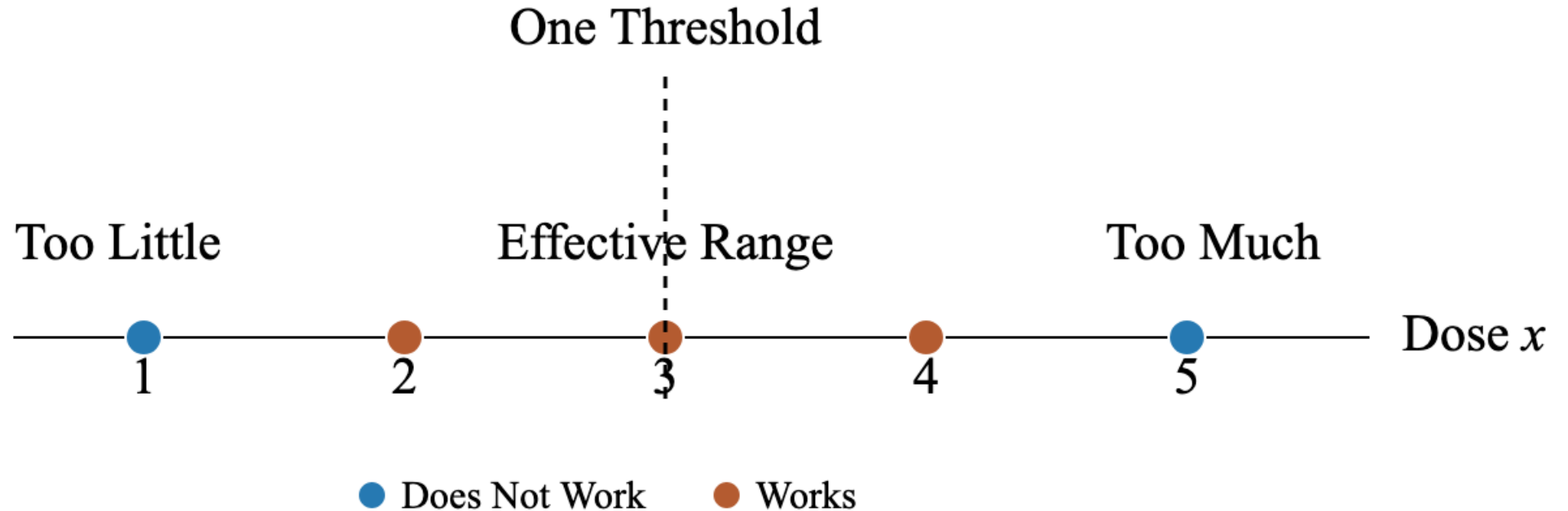
Can One Threshold Separate These Samples?

A Hypothetical Medicine



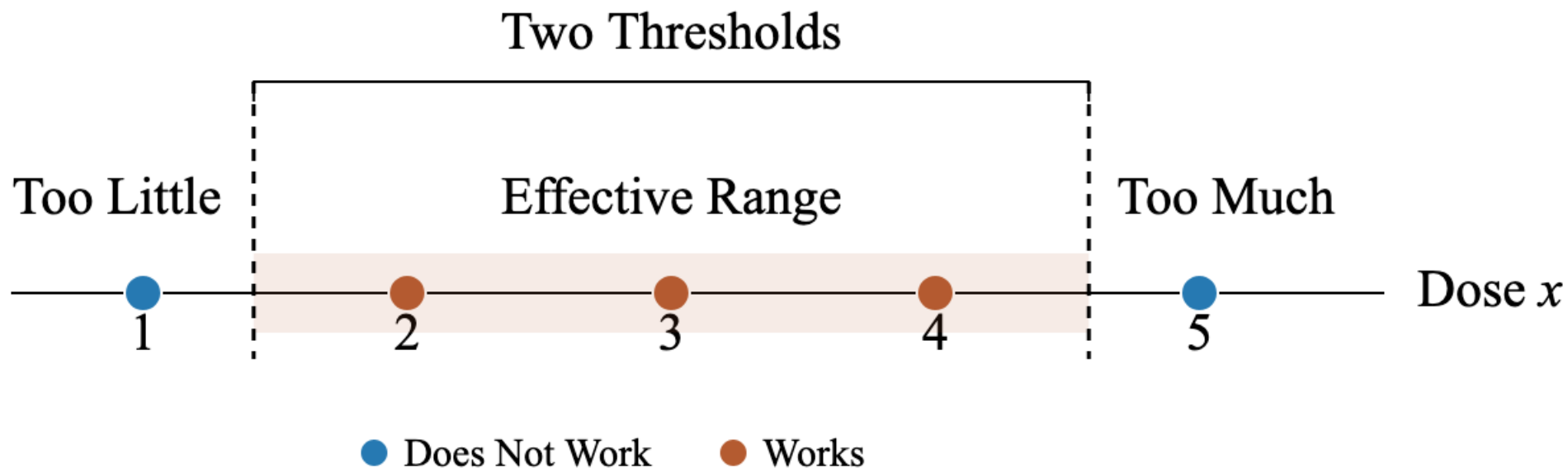
Can One Threshold Separate These Samples?

A Hypothetical Medicine



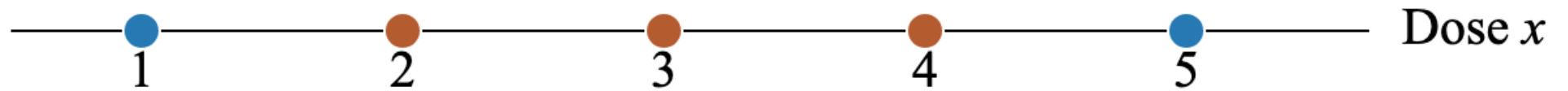
Can One Threshold Separate These Samples?

A Hypothetical Medicine



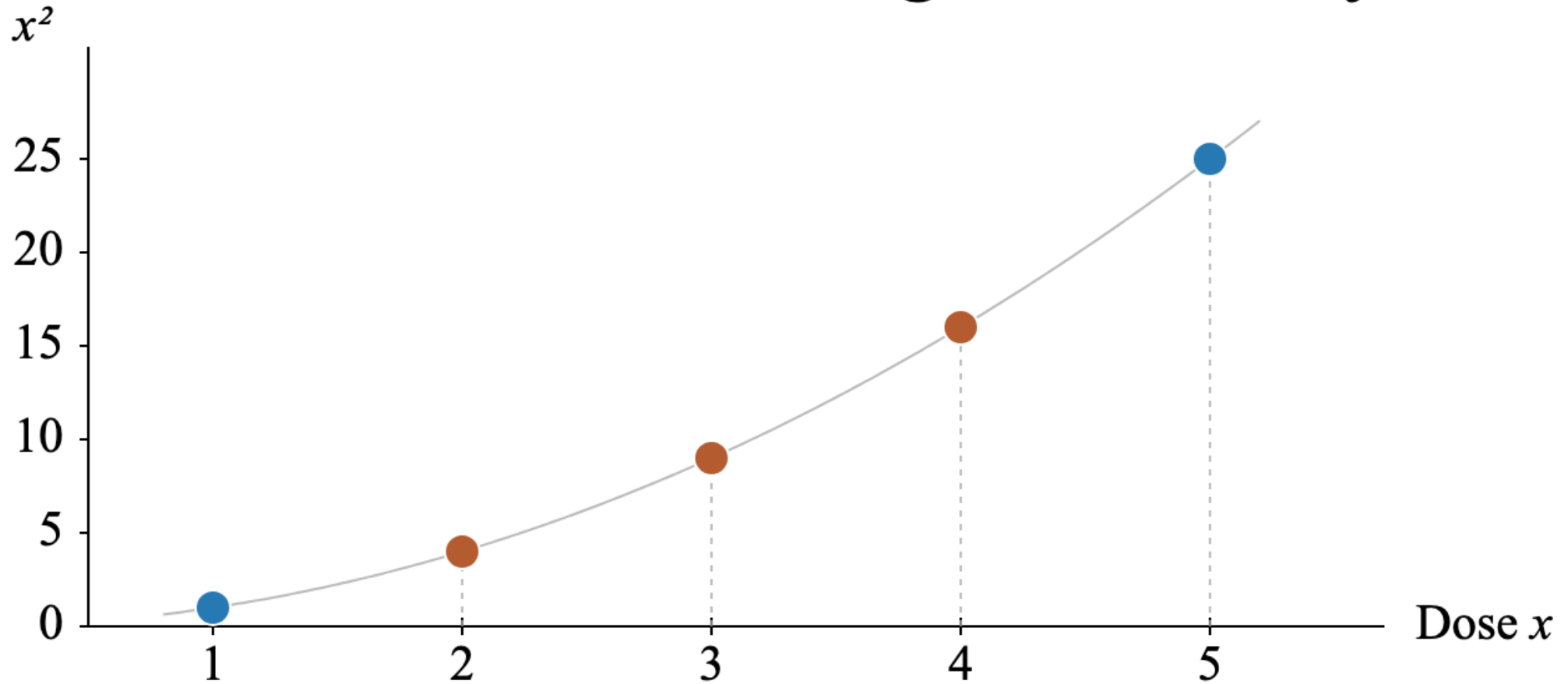
In One Dimension, a Linear Boundary Is One Threshold

Add a Feature, Find a Straight Boundary



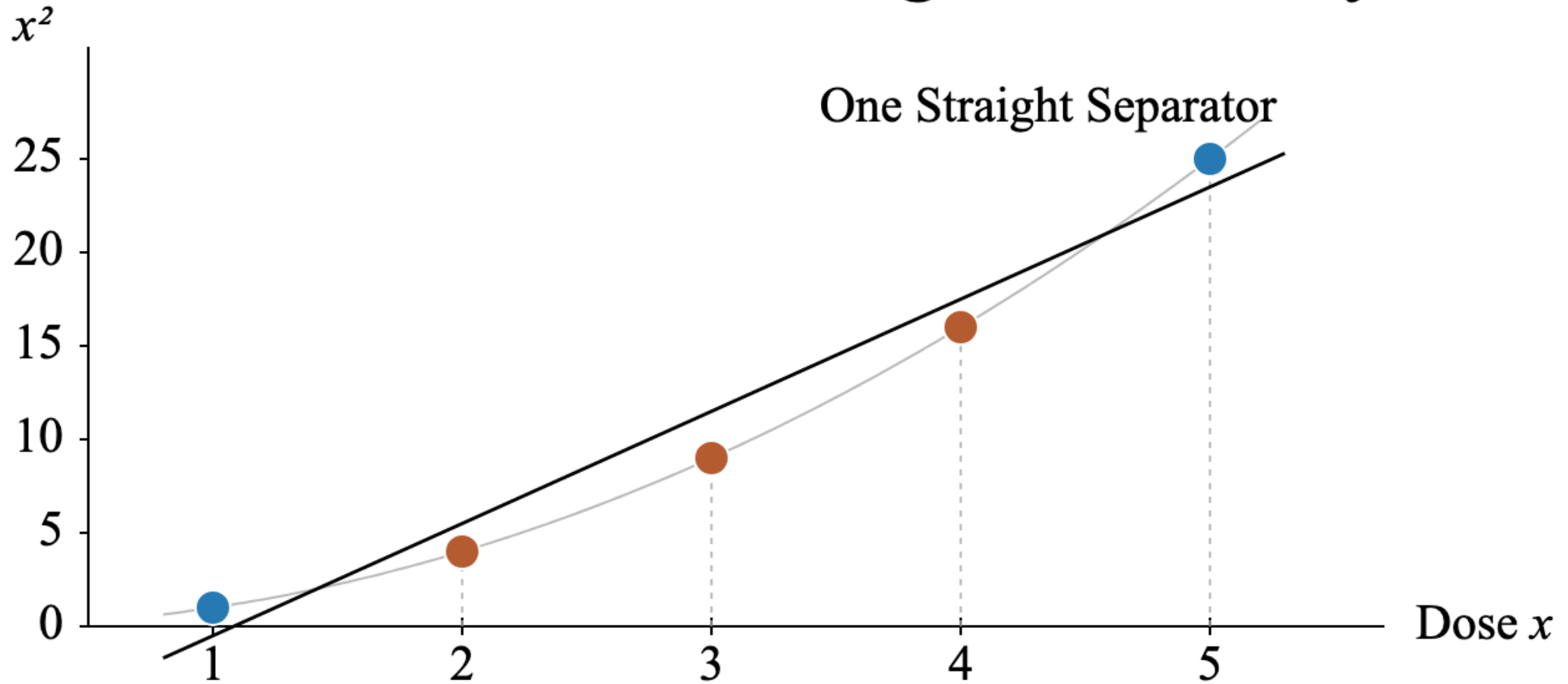
Keep x , Add x^2 as a Second Feature

Add a Feature, Find a Straight Boundary



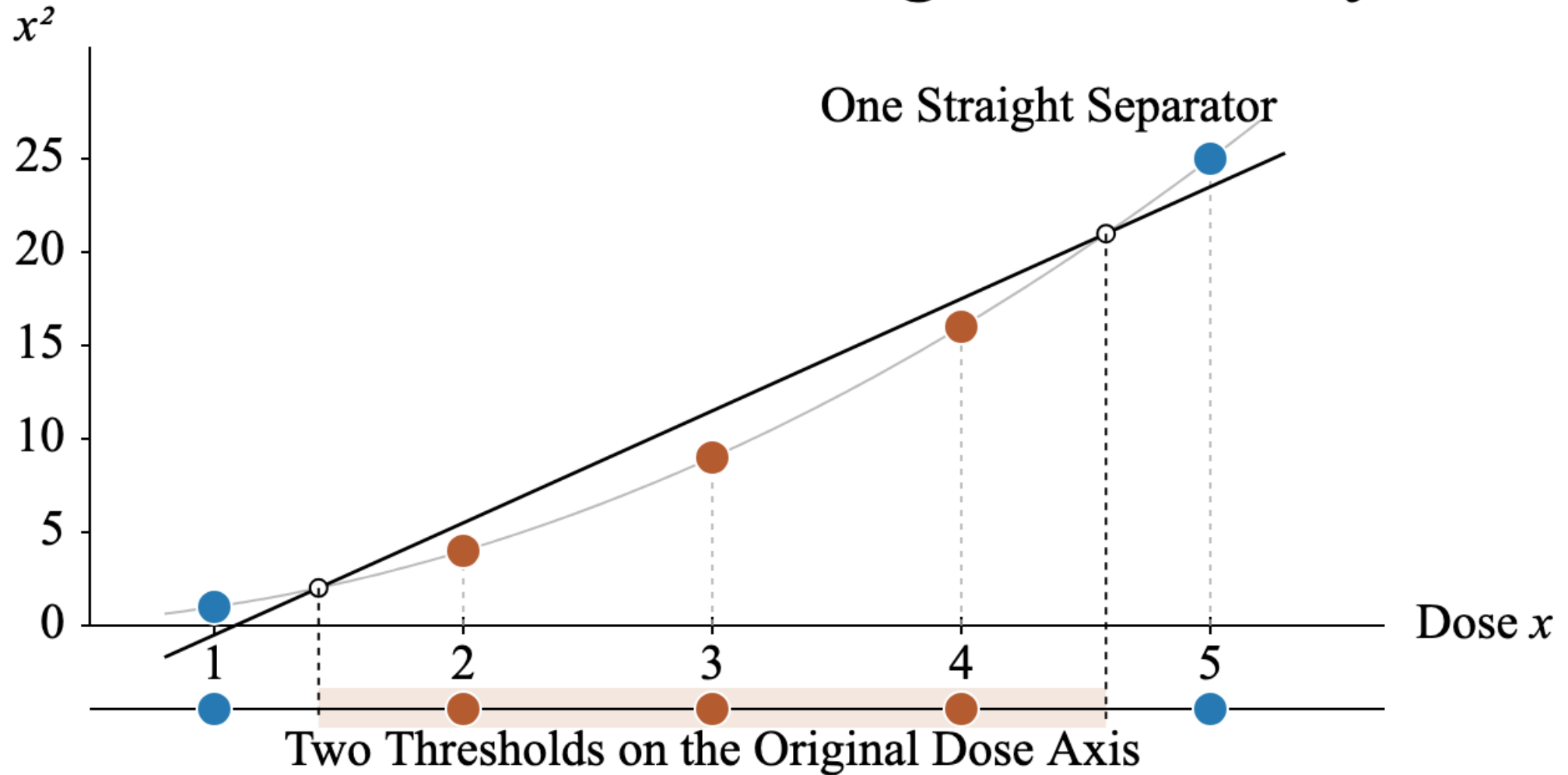
Keep x , Add x^2 as a Second Feature

Add a Feature, Find a Straight Boundary



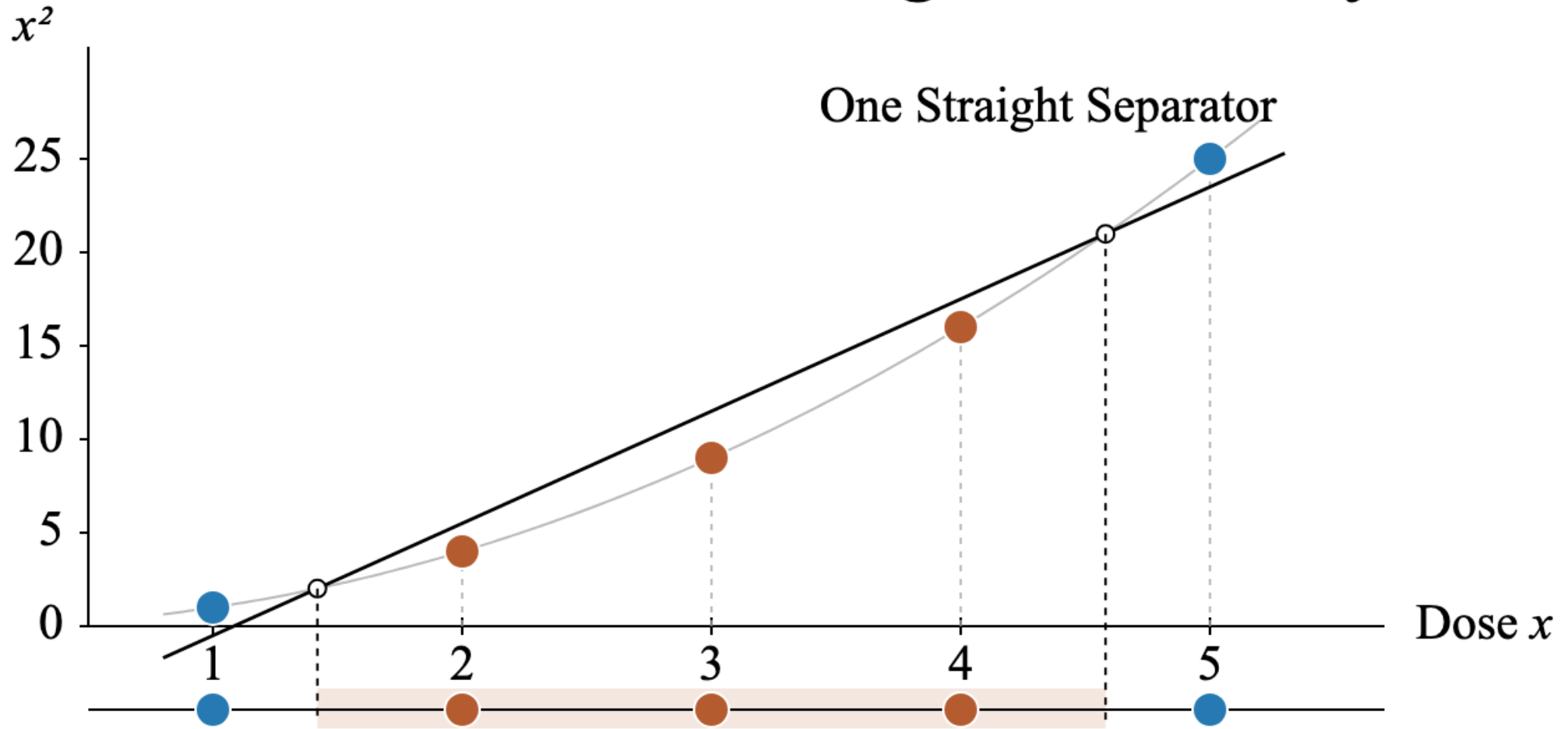
Keep x , Add x^2 as a Second Feature

Add a Feature, Find a Straight Boundary



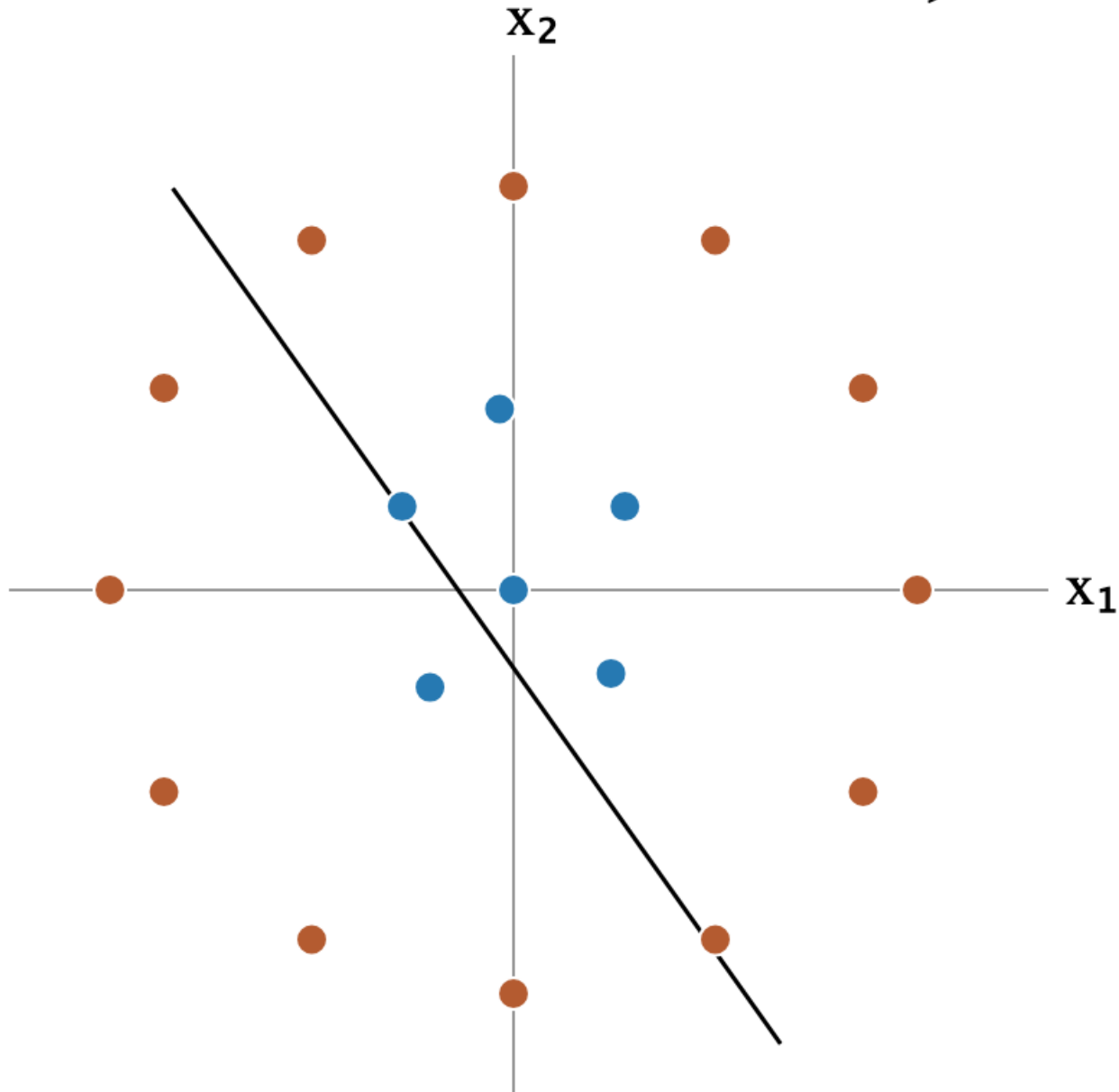
A Straight Boundary in Feature Space Can Be Nonlinear in the Original Space

Add a Feature, Find a Straight Boundary



A Linear Model in the New Features
A Nonlinear Decision Boundary in the Original Input

A Plane in 3D, a Circle in 2D

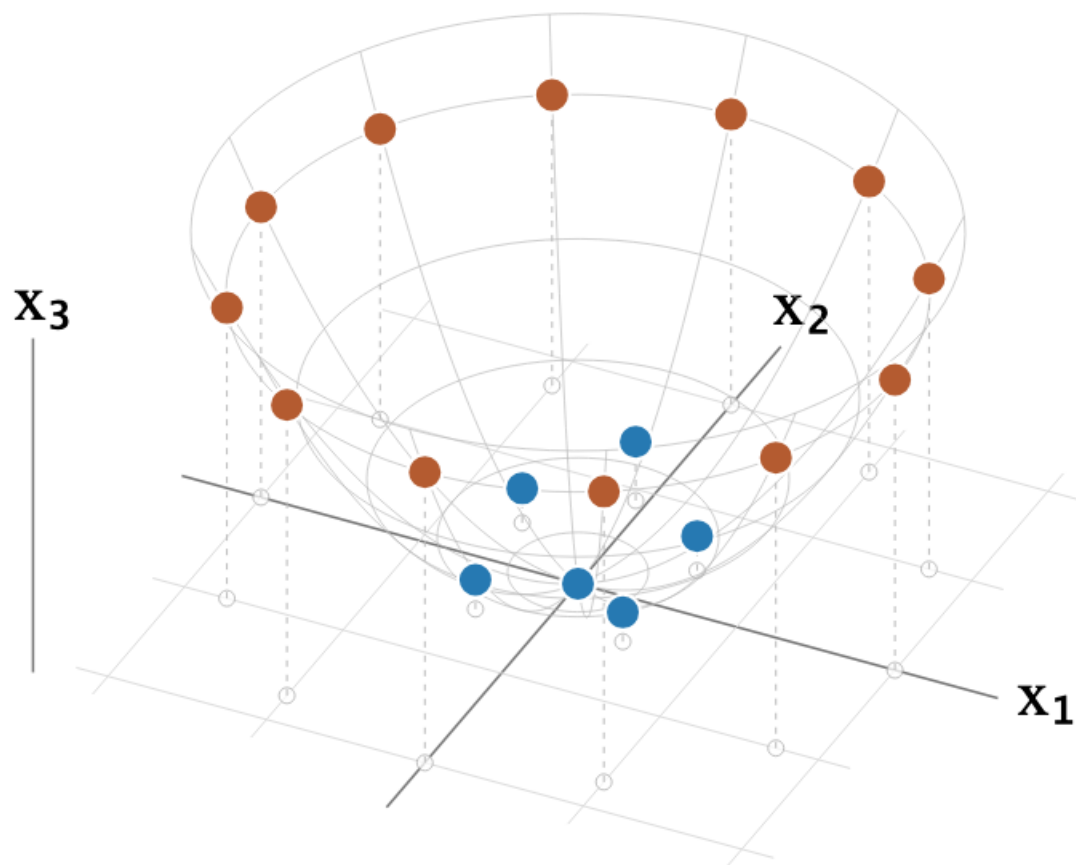


Start in 2D

No Straight Line

Separates These Classes

A Plane in 3D, a Circle in 2D



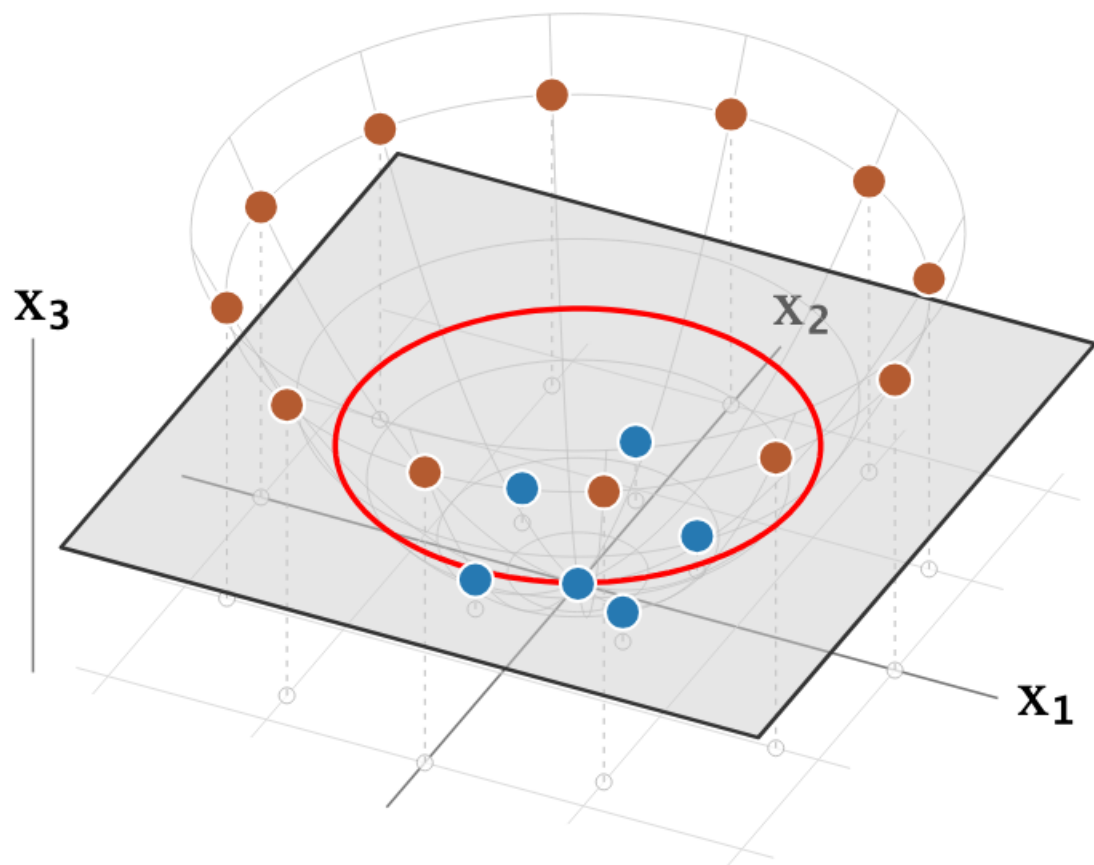
Add a Third Feature

$$x_3 = x_1^2 + x_2^2$$

Farther from the Centre
Means Higher Up

Drag to Rotate · Double-click to Reset

A Plane in 3D, a Circle in 2D



Add a Third Feature

$$x_3 = x_1^2 + x_2^2$$

Farther from the Centre
Means Higher Up

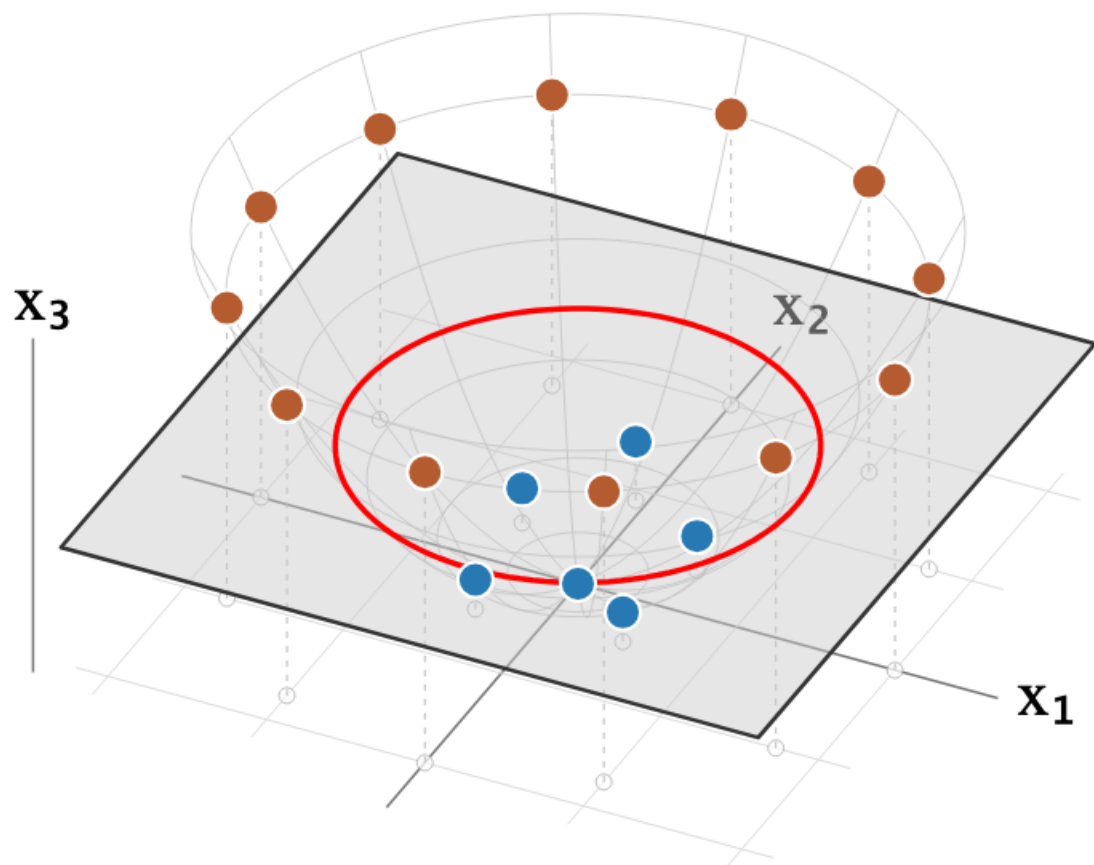
Separate with a Plane

$$x_3 = c$$

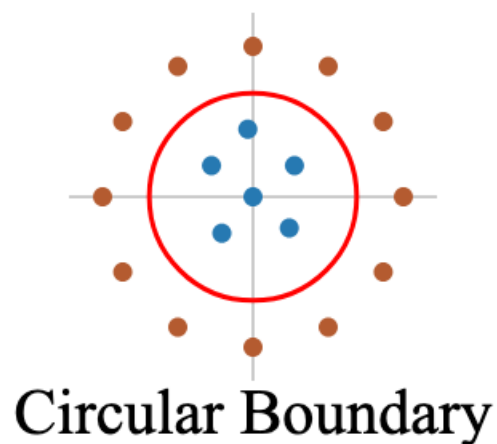
Blue Below · Orange Above

Drag to Rotate · Double-click to Reset

A Plane in 3D, a Circle in 2D



Original 2D Space



Add a Third Feature

$$x_3 = x_1^2 + x_2^2$$

Farther from the Centre
Means Higher Up

Separate with a Plane

$$x_3 = c$$

Blue Below · Orange Above

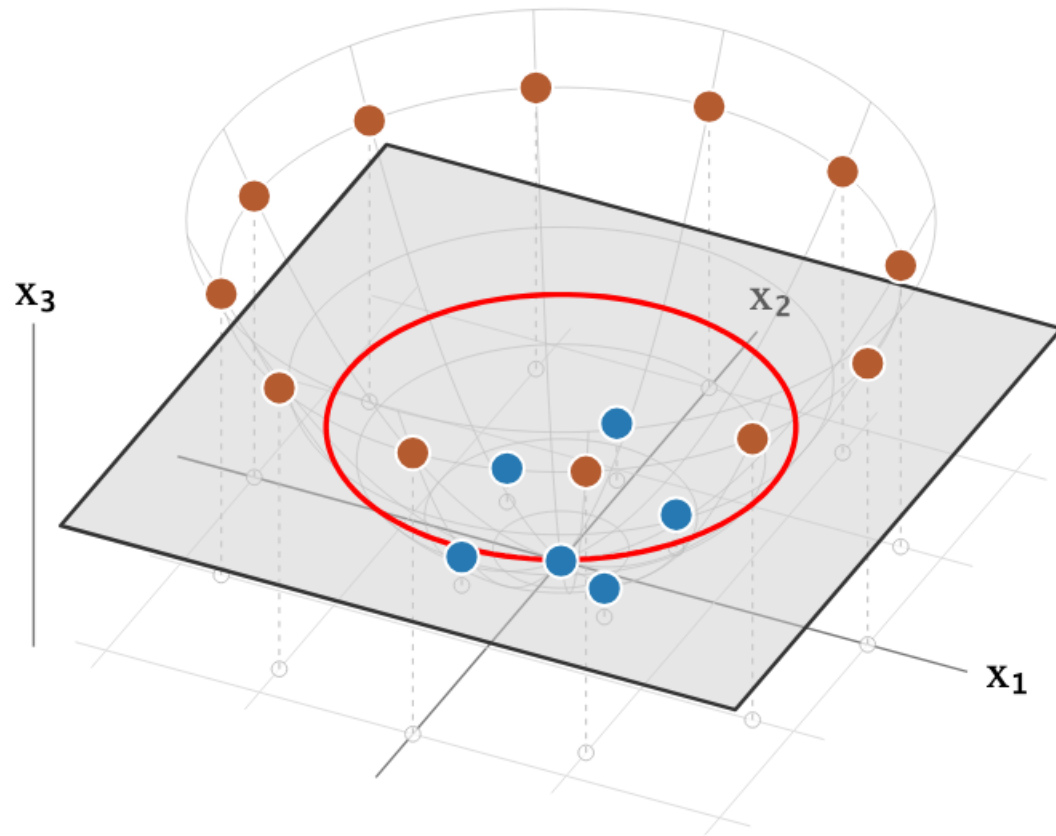
Project the Intersection

$$x_1^2 + x_2^2 = c$$

A Circle in the Original Space

Drag to Rotate · Double-click to Reset

From Feature Transformations to Kernels



Last Slide: We Added a Feature

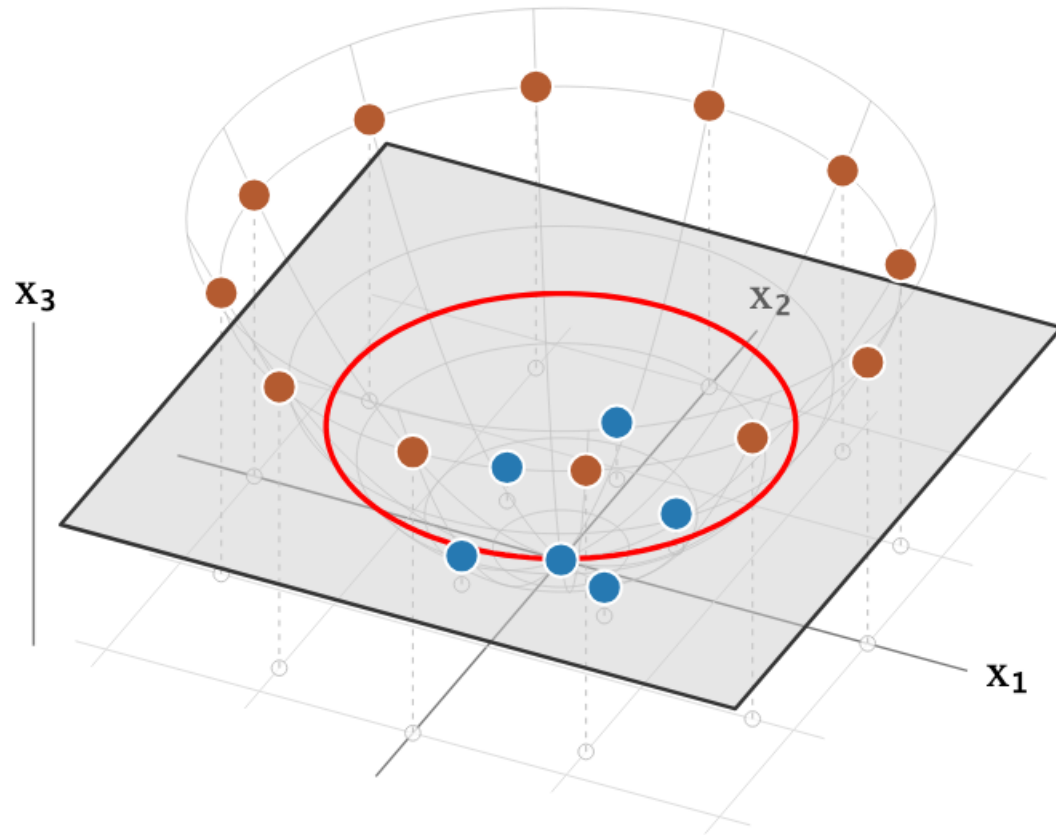
$$x_3 = x_1^2 + x_2^2$$

Lift the Samples into 3D

Separate Them with a Plane

We Added Squared Distance as a Feature

From Feature Transformations to Kernels



Without Building
the Extra Features?

A **Kernel** Compares Two Samples
in a Transformed Feature Space

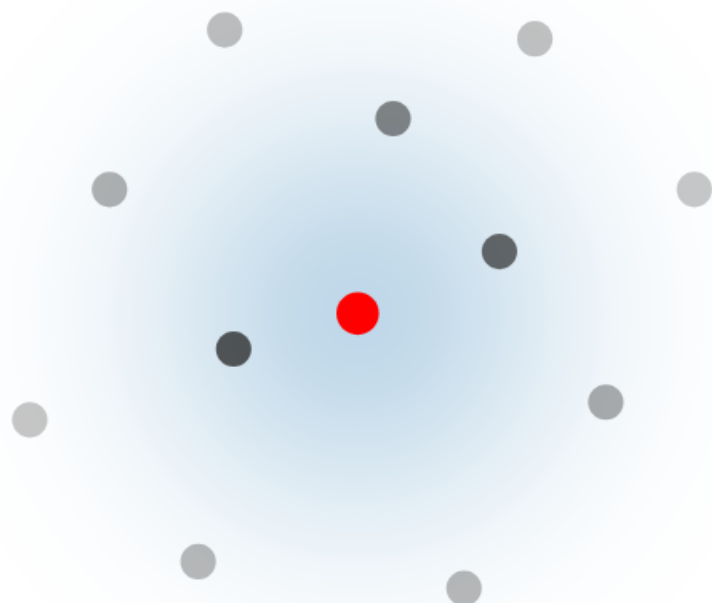
It Computes This Directly from
Their Original Features

The SVM Can Then Find a
Separating Plane in That Space

New Feature Space, Same Original Inputs

From Feature Transformations to Kernels

RBF Similarity to the Marked Sample



Darker Samples = More Similar

RBF Is One Such Kernel

RBF: Radial Basis Function

Nearby Samples Are More Similar

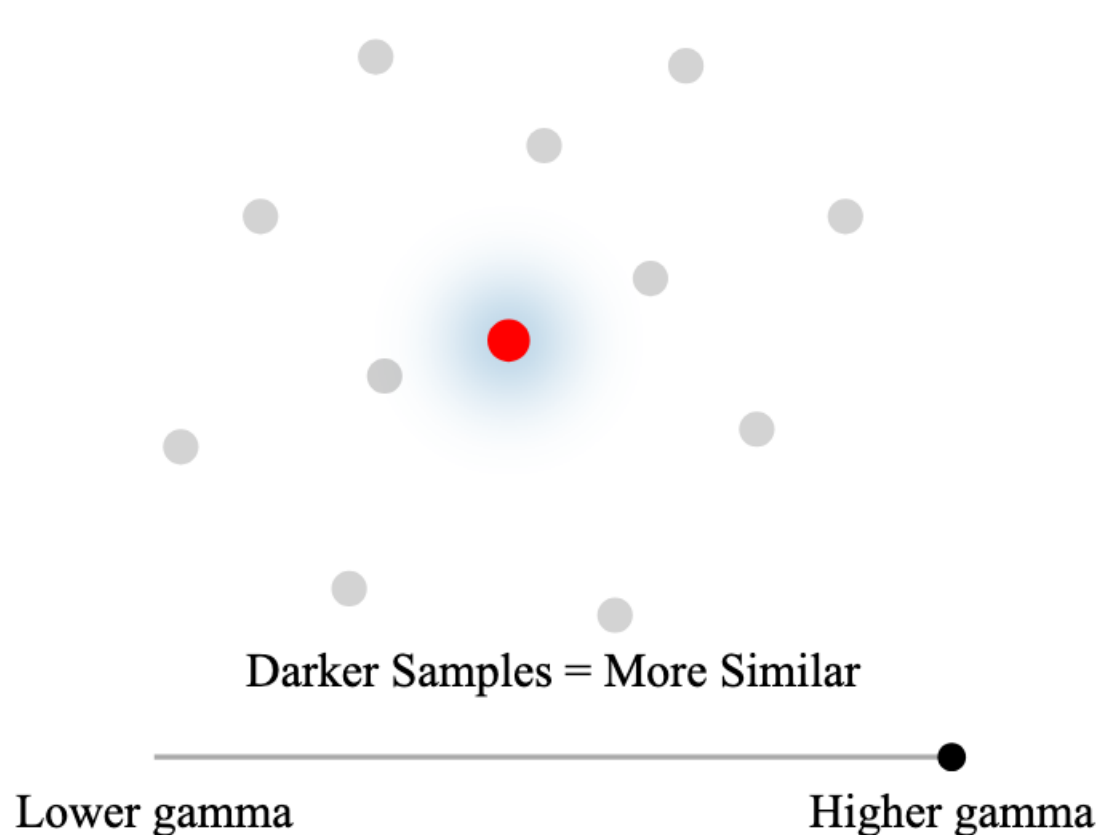
Distant Samples Are Less Similar

A Plane in Its Feature Space Can Give
a Curved Boundary in Our Input Space

New Feature Space, Same Original Inputs

From Feature Transformations to Kernels

RBF Similarity to the Marked Sample



gamma Controls How Local

Higher gamma:

A Narrower Neighbourhood

More Intricate Boundaries

Are Possible

C Controls Regularization

Smaller C: Stronger Regularization

New Feature Space, Same Original Inputs

maib7002.dirk.hk/L3.zip

Demo: support_vector_machines_sklearn.ipynb

References and Suggested Reading

Core Reading

Machine Learning with PyTorch and Scikit-Learn

Raschka, Liu and Mirjalili · 2022

Chapter 3: A Tour of Machine Learning
Classifiers Using Scikit-Learn

Perceptron, Logistic Regression and SVMs

scikit-learn Documentation

[Logistic Regression](#)

[Support Vector Machines and Kernels](#)

[Feature Scaling](#)

Next Lecture: L4

Supervised Learning II (Continued)

Model Evaluation and Interpretation

Exit Ticket

Complete Individually Before You Leave
Submit Your Answers on Ed