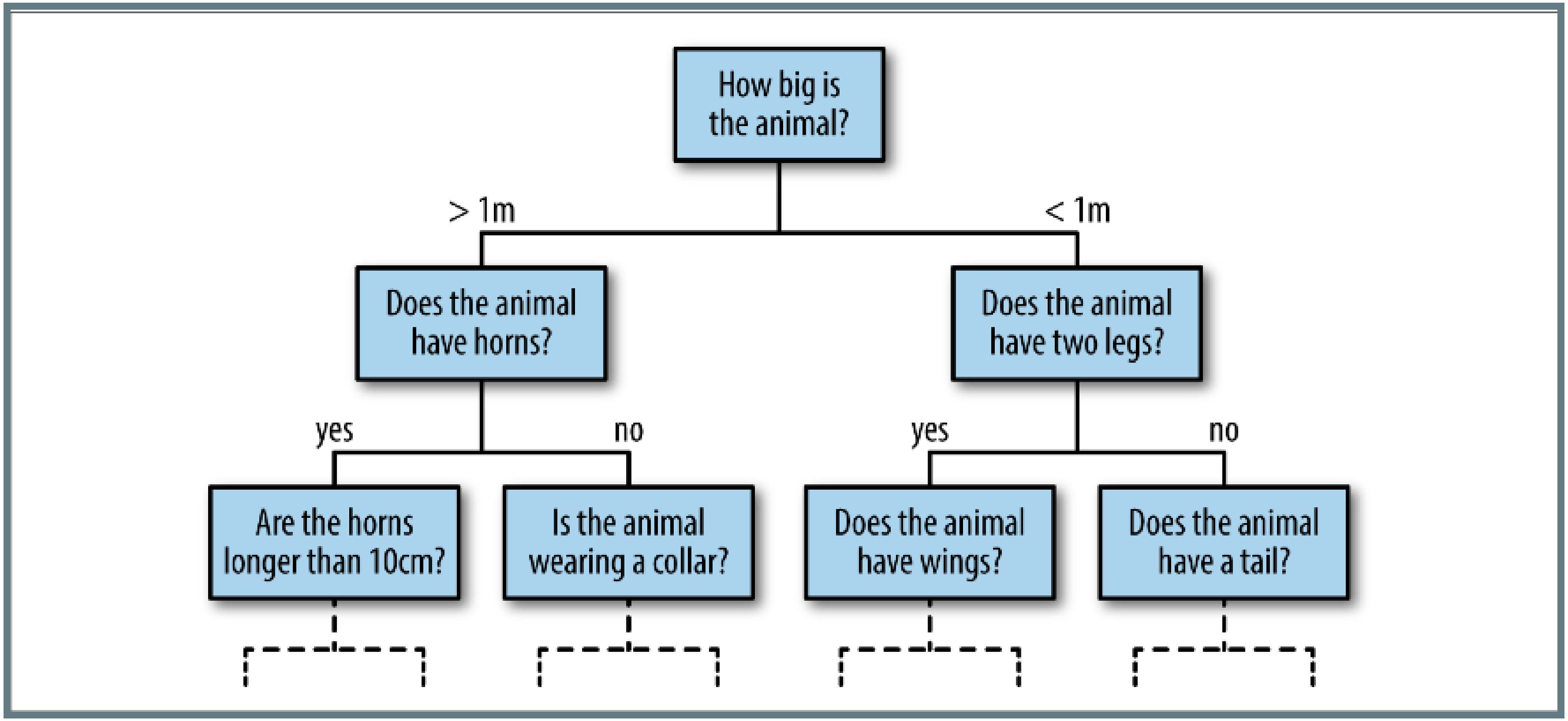
DECISION TREES AS A MACHINE LEARNING MODEL

ABHIJIT DASGUPTA

DECISION TREES

DECISION TREES

- Decision trees (nominally) are flowcharts of a decision-making process



At the bottom nodes are the predicted outcomes

DECISION TREES IN MACHINE LEARNING

- Decision Trees are a form of supervised learning
- Data is in the form of predictors and a target
- The target can be continuous (e.g., credit risk) or categorical (e.g. risk categories)
- Objective is to predict the target from the predictors

WHAT DO WE WANT A LEARNING MACHINE TO LEARN?

- If I get values for a set of predictors
- I want to know a prediction of the target for that multivariate predictor value
- I want this prediction to be as accurate as possible

Petal Length = 2 cm, Petal Width = 0.4 cm
Prediction: setosa

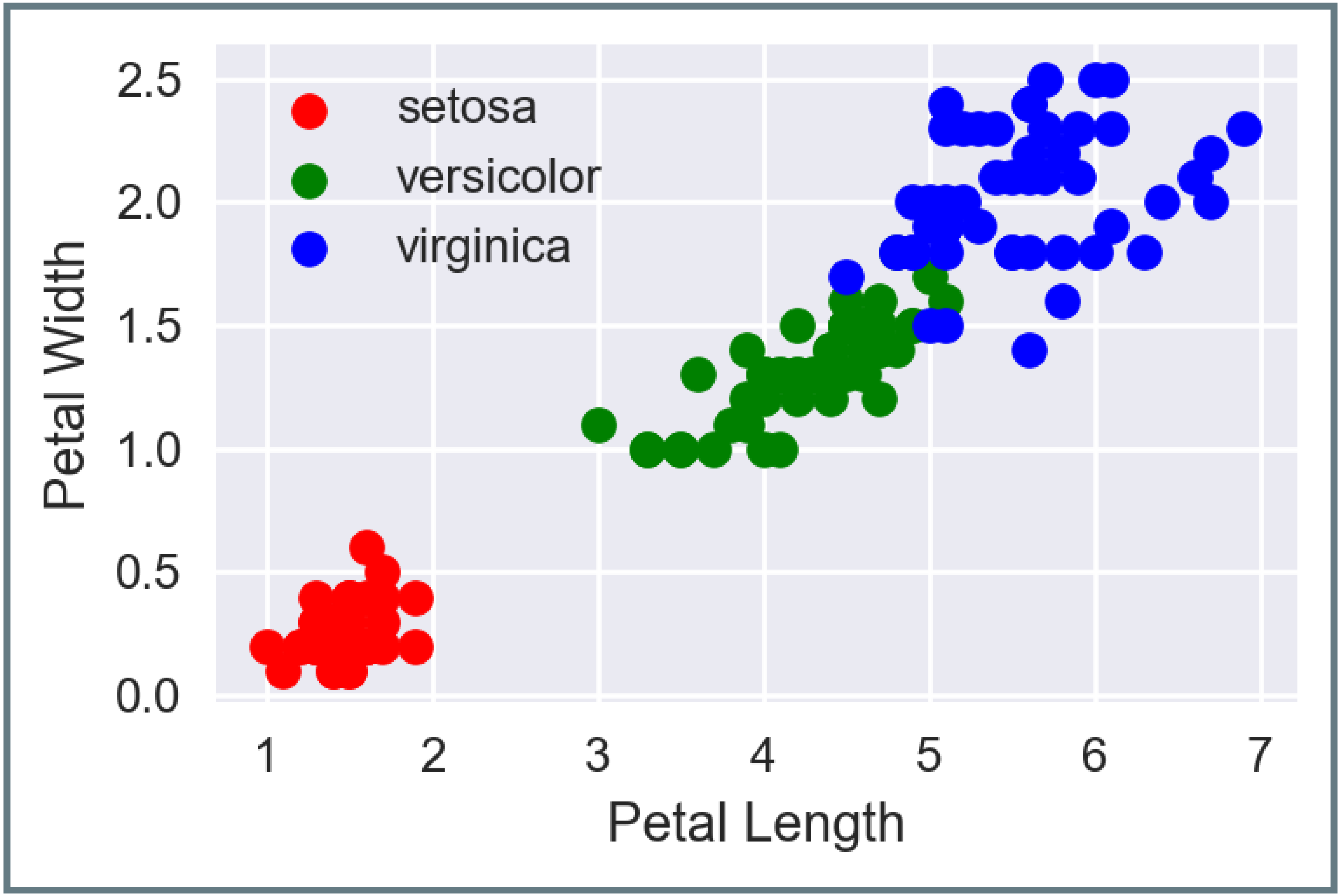
Petal Length = 4 cm, Petal Width = 1 cm
Prediction: versicolor

WHAT DO WE WANT A LEARNING MACHINE TO LEARN?

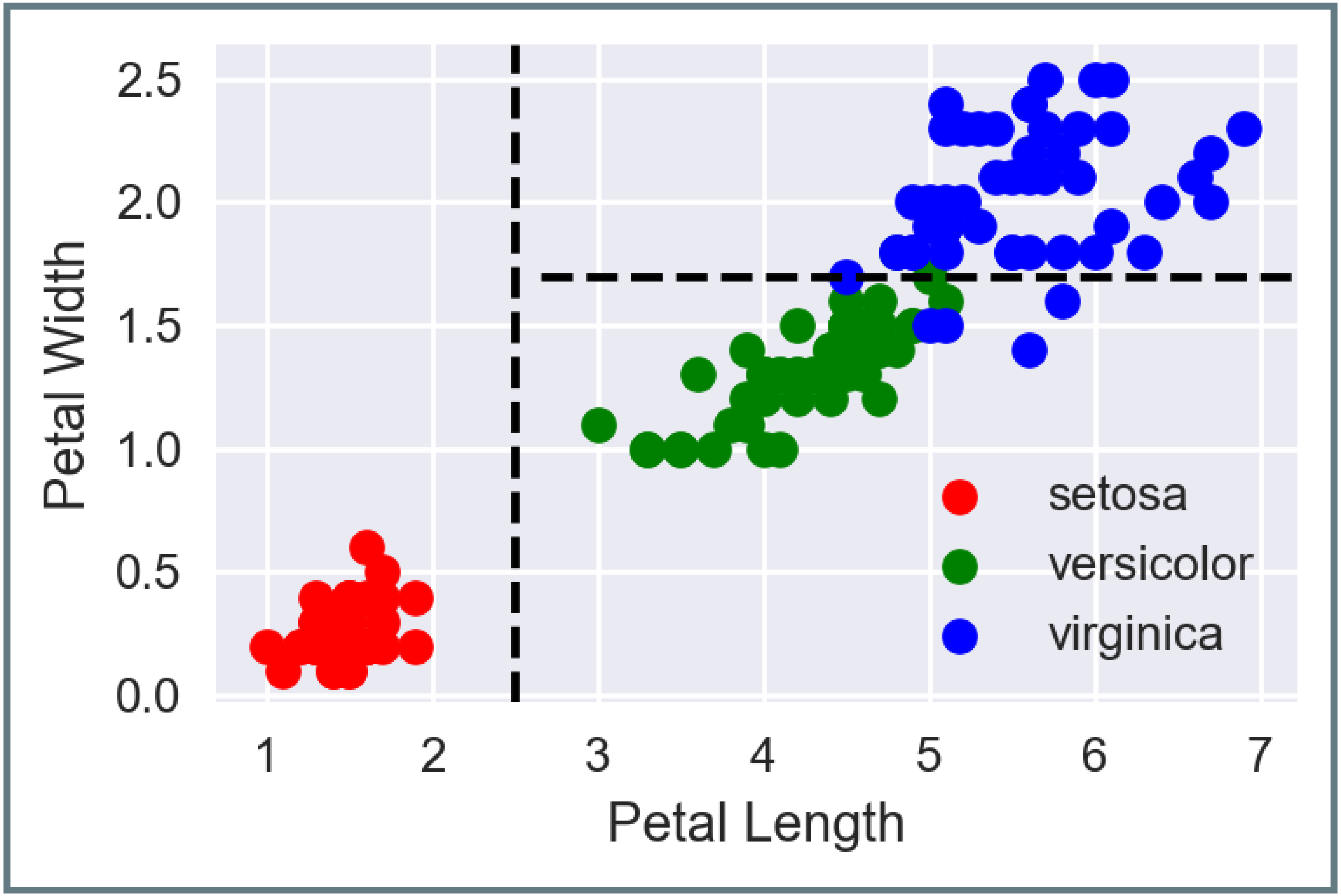
- So I want some kind of rule for prediction
- I want the learning machine to derive/estimate this rule from data
- I want the best possible (most accurate) rule

A MOTIVATING EXAMPLE

LET'S MOTIVATE THE IDEA



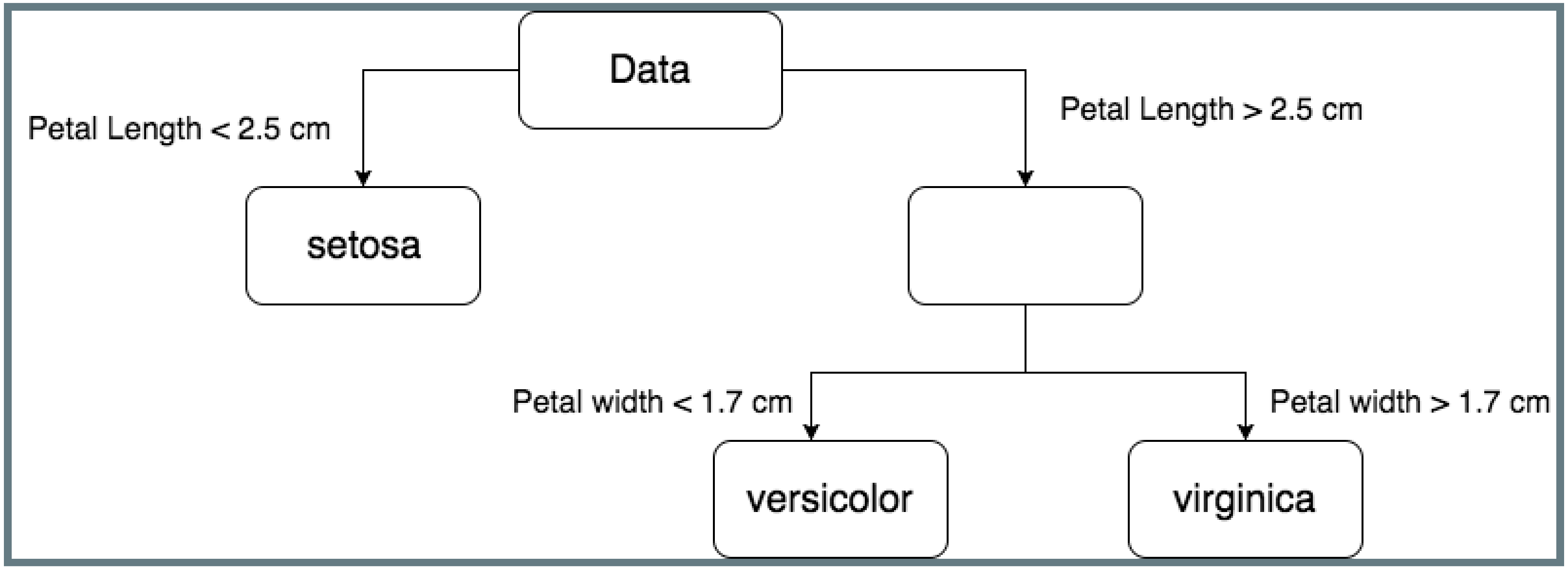
LET'S MOTIVATE THE IDEA



A RULE

Petal Length	Petal Width	Prediction
< 2.5 cm		setosa
> 2.5 cm	< 1.7 cm	versicolor
> 2.5 cm	> 1.7 cm	virginica

A DECISION TREE



PYTHON

```
iris = sns.load_dataset('iris')
# iris = pd.read_csv('iris.csv')
iris['Prediction'] = ''
iris.loc[iris['petal_length'] < 2.5, 'Prediction'] = 'setosa'
iris.loc[(iris['petal_length'] > 2.5) & (iris['petal_width'] < 1.7), 'Pre
iris.loc[(iris['petal_length'] > 2.5) & (iris['petal_width'] > 1.7),'Pred
pd.crosstab(iris['species'], iris['Prediction'])

from sklearn.metrics import accuracy_score
accuracy_score(iris['species'],iris['Prediction'])
```

WHAT WOULD A MACHINE NEED?

THE BASIC PROCESS

Let's agree to only binary splits for one variable each time.

Let's also agree that we are classifying (i.e. we have discrete label)

We would need:

1. A criterion for splitting
2. Determining when not to split any more (terminal node)
3. Splitting recursively to build a tree
4. A way to make a prediction from new data

A CRITERION FOR SPLITTING

THE GINI INDEX

The Gini index is a measure of the class-purity of a group.

How homogeneous is the group?

If it’s homogeneous (all have the same label) we don’t need to split it further.

THE GINI INDEX

- If our data has *k* labels {1,2, ..., K}, let $P(label = k) = p_k$.

$$GI = \sum_{k=1}^{K} p_k(1 - p_k)$$

$$= 1 - \sum_{k=1}^{K} p_k^2$$

GINI INDEX

$$GI = 1 - \sum_k p_k^2$$

Group 1 = {red, red}, Group 2 = {blue, blue} -> GI_1 = ?, GI_2 = ?

Group 1 = {red, blue}, Group 2 = {blue, red} -> GI_1 = ?, GI_2 = ?

WHEN DO WE DECIDE TO SPLIT THE DATA

We split the data where the **frequency-weighted sum of Gini indices** for the split is smallest.

WHEN DO WE DECIDE TO SPLIT THE DATA?

If parent node has n rows, and the split groups have n_1 and n_2 rows then

$$GI = GI_1 \times \frac{n_1}{n} + GI_2 \times \frac{n_2}{n}$$

SPLITTING THE DATA

SPLITTING THE DATA

For each value of each predictor, we

1. Split the data into a left and a right group at that value
2. Compute the Gini index for the split
3. Choose that combination of predictor and value that has the smallest Gini index

BUILD A TREE

HOW TO BUILD A TREE

1. Calling terminal nodes
2. Recursive splitting
3. Grow the tree

TERMINAL NODES

- Node is homogeneous
- It is below the minimum number of records required for a split (user-supplied)
- The tree has grown to maximum depth (user-supplied)

At a terminal node, the prediction will be the most popular class in the node

RECURSIVE SPLITTING

1. We call our splitting algorithm over and over from the top
2. At each stage we will call a node either terminal or split-worthy
3. If a node is split-worthy, we split it again
4. We stop when all nodes are terminal

BUILDING THE TREE

1. Start at the root node
2. Call our splitting function recursively until the whole tree is built

MAKE A PREDICTION

PREDICTING A CLASS LABEL

1. Determine which terminal node the new datum falls in
2. Find the prediction for that terminal node