# Data 100, Discussion 15

**Suraj Rampure**

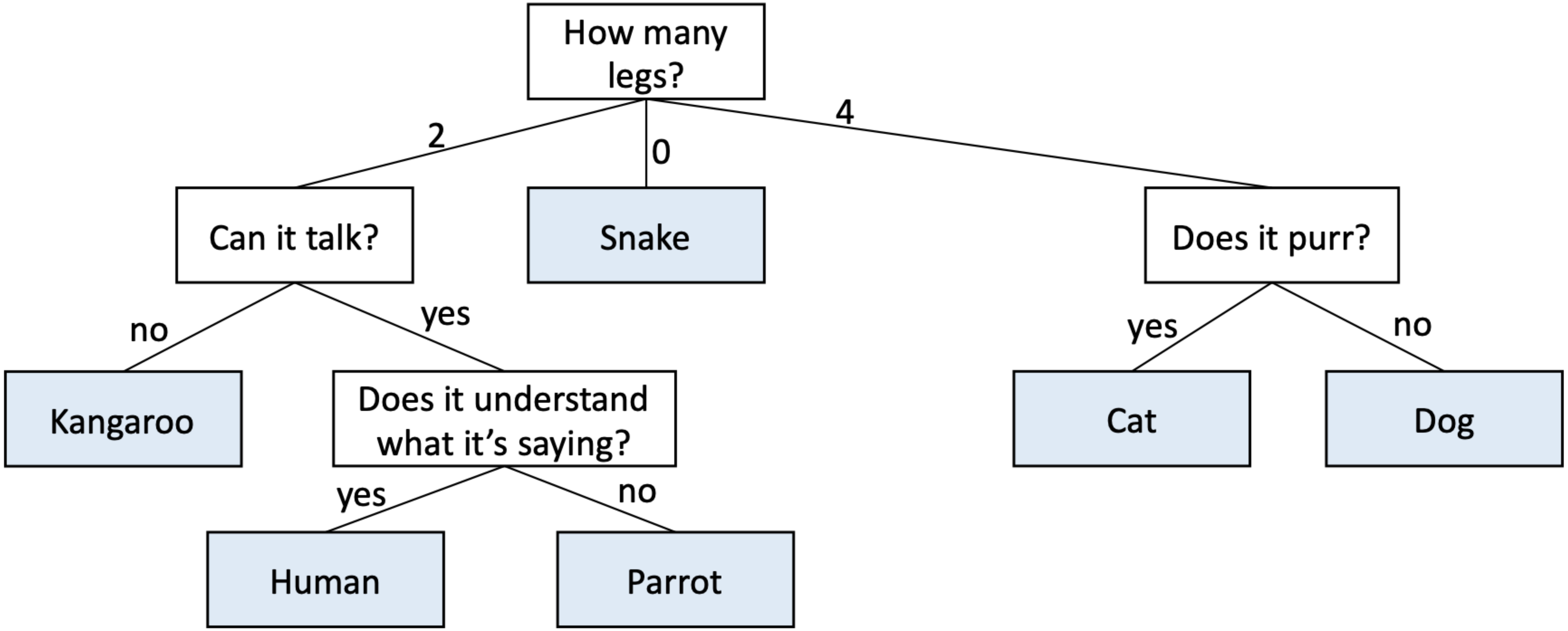
Wednesday, December 4th, 2019

# Agenda

- Decision Trees and Random Forests
- Clustering
- HCE

# Decision Trees

Decision trees are used for **classification**. Instead of using a linear model, decision trees ask a series of yes/no questions, that eventually lead to a classification.



# Training Decision Trees

"best"
entropy

$$= -\sum_c p_c \log p_c$$

When training, we determine the **structure** of our decision tree.

Algorithm:

```
all data starts at root node

repeat until every node is pure or unsplittable:
        pick the best feature x and the best split value β
        split data into two nodes, one where x < β, and one where x ≥ β
```

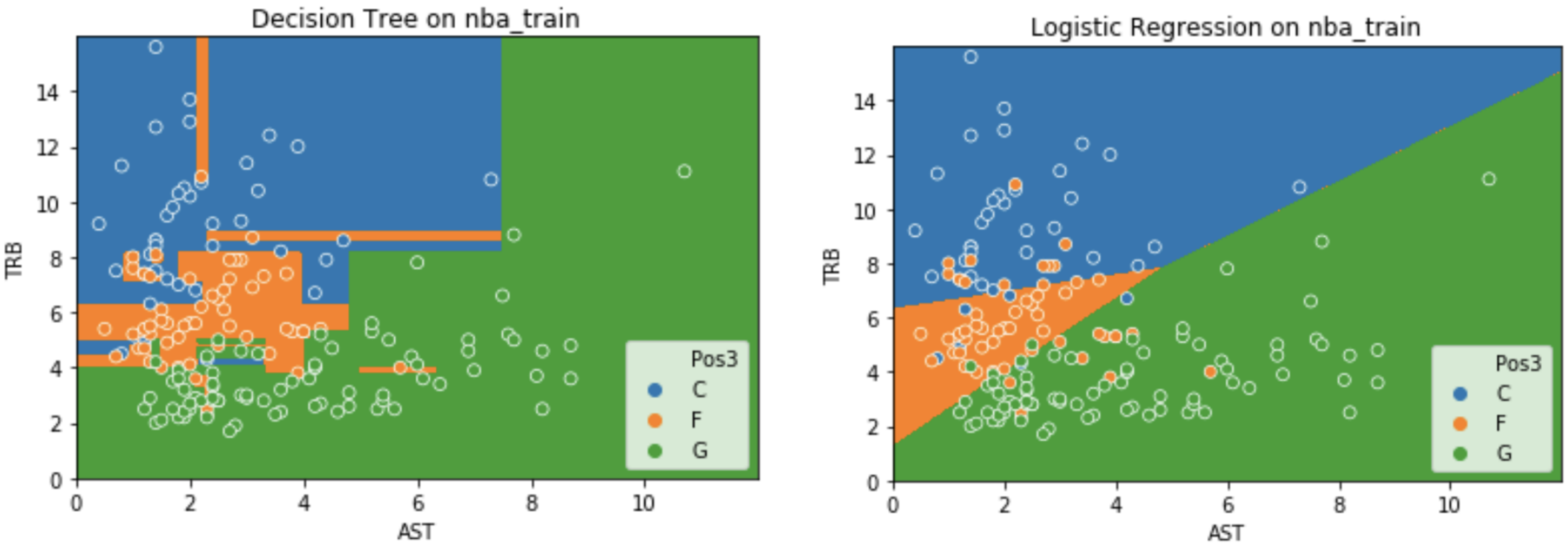
Our decision tree reflects our training data.

- If, in our training data, there are multiple points that belong to the same class (e.g. $(2, 3, cat)$ and $(2, 3, dog)$), we will have nodes that are **unsplittable**.

$$\vec{x} = [2 \quad 3]^T$$

# Overfitting

Since our decision trees are non-linear, they can severely overfit to our training data. For instance, the decision tree from lab (vs logistic regression on the same data):



In fact, if our training data doesn't have any overlapping classes, we will always by default reach 100% training accuracy. This is not good in practice, since it indicates our model will fail to generalize.

## Combatting Overfitting

1. Prevent Growth

- Examples:
  - Set a maximum tree depth
  - Don’t split nodes that have less than 1% of samples

2. Build full trees, then prune branches

- One way to do this: set aside a validation set. If replacing a node by its most common prediction doesn’t change the validation error, then don’t split that node.

3. Use random forests

# Random Forests and Ensemble Methods

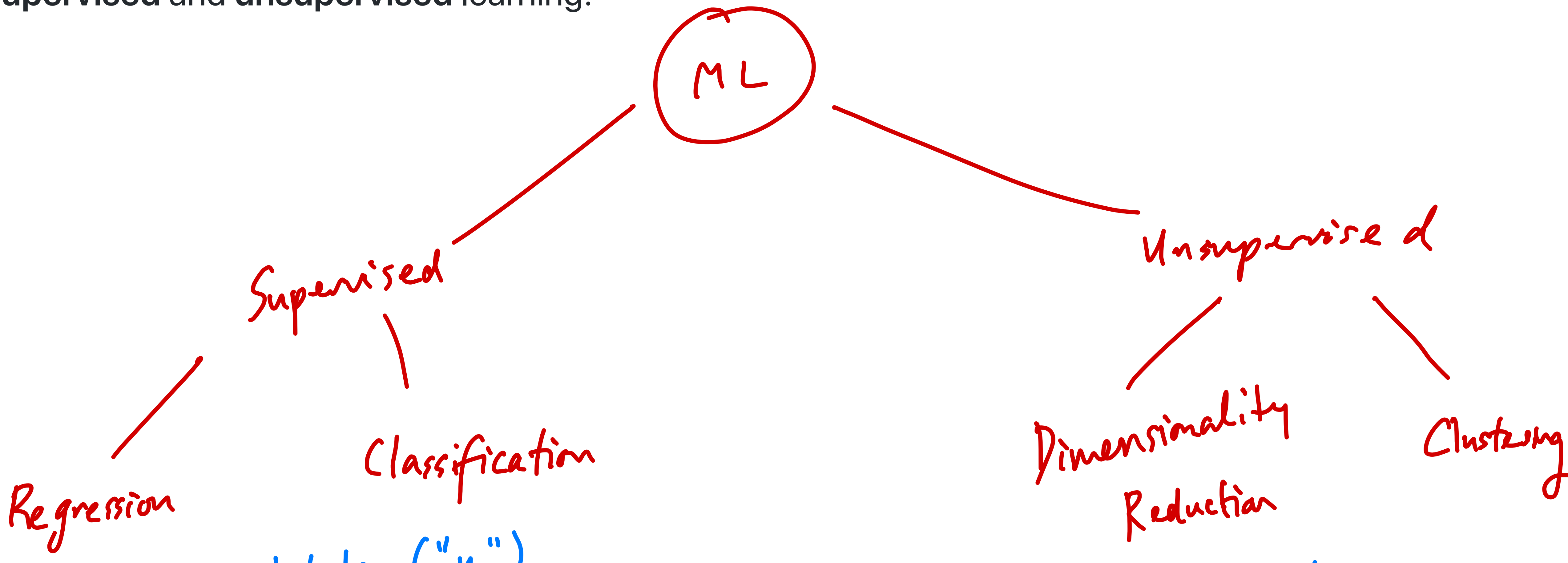
Random forests are a collection of decision trees.

- General idea: Instead of fitting just one decision tree to our training data, we fit several. Our prediction is then just the most common prediction that our sub-trees make.

You can bootstrap your data to generate different samples to train your decision tree on. However, this often isn’t enough to get enough variability, especially because decision trees tend to overfit. Additionally, we should train each tree on a subset of the total features.

# Taxonomy of Machine Learning

Before talking about clustering, let's talk about the two different branches of machine learning – **supervised** and **unsupervised** learning.



→ given labels ("y")

→ learn the relationship between X and y

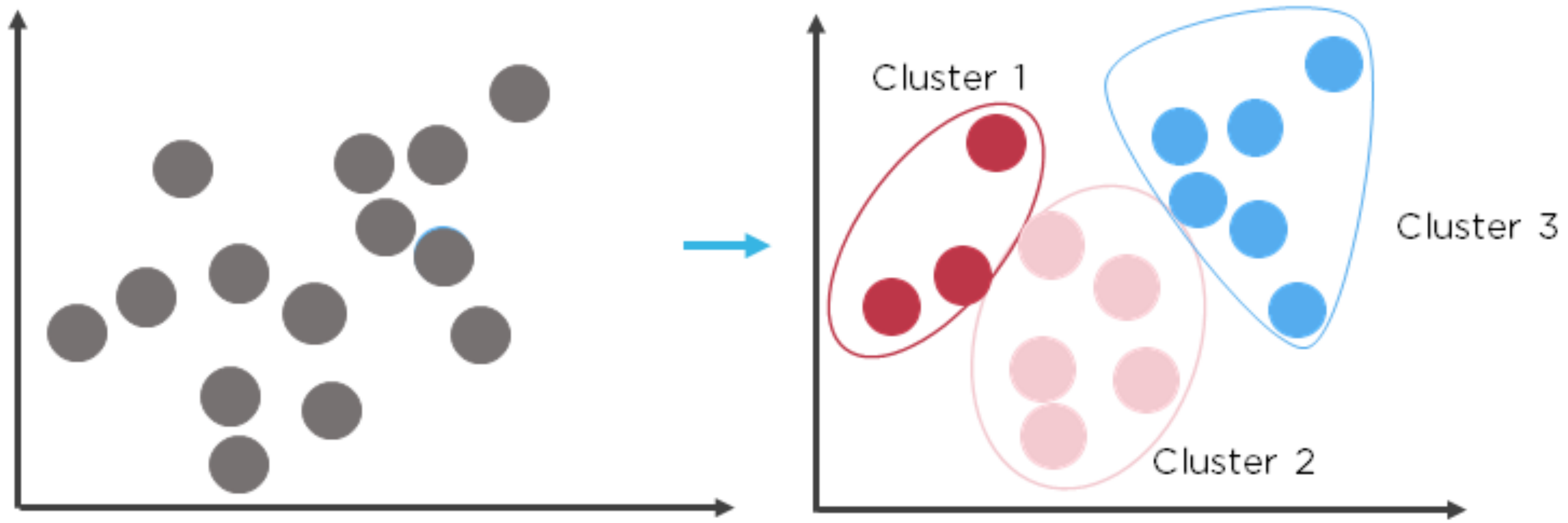
→ no labels

→ want to learn the structure of the data

# Clustering

Clustering is an **unsupervised** learning task. We're given raw data, and we want to try and find clusters in it. This can be useful in performing EDA (understanding the data we're given), or even in perhaps performing classification.

In clustering, we want to assign our data points to one of some number of clusters.



# K-Means Clustering

In $k$-means clustering, we **first** pick $k$, the number of clusters that we're trying to find.

- The principle that $k$-means clustering follows is that points in a given cluster should be close to the **centroid** of that cluster.

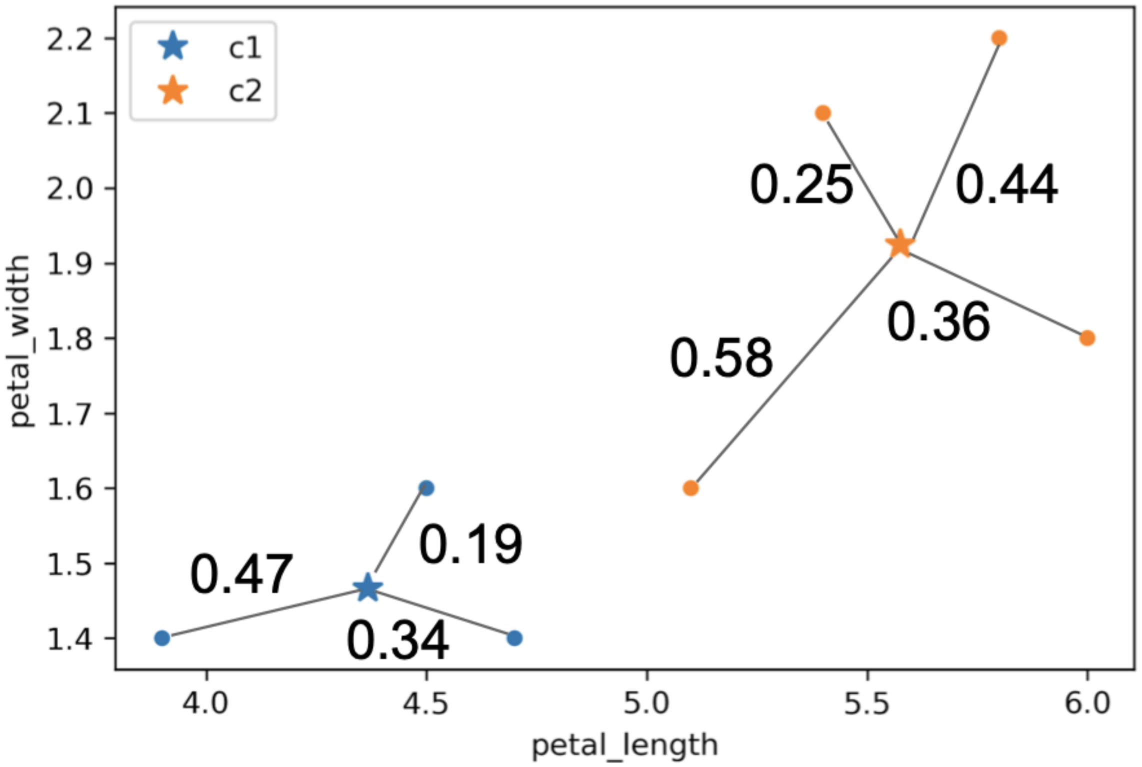
Algorithm:

```
initialize k centroids randomly

repeat until convergence (no change in clusters):
        assign each point to the closest centroid
        update centroids based on new clusters
```

# Distortion

**Distortion** is one of the possible loss functions we can use for clustering, and it is the one that $k$-means clustering minimizes.



Distortion is the **sum of the average of the squared distance to the centroid for each cluster**:

$$D = \frac{0.47^2 + 0.19^2 + 0.34^2}{3} + \frac{0.58^2 + 0.25^2 + 0.44^2 + 0.36^2}{4}$$

# Agglomerative Clustering

Agglomerative clustering is another clustering technique. Similarly to $k$-means, we also start off with a fixed number $k$ of clusters that we want to find.

Algorithm:

```
assign each point to its own cluster

while there are more than k clusters left:
        join the two closest clusters
```

**Question:** What does it mean for two clusters to be the "closest" together?

# Silhouette Score

The silhouette score is a metric for how good the cluster assignment **for a single point** is (whereas distortion was for an entire dataset).

Given the following definitions,

$$A = \text{average distance to points in its own cluster}$$

$$B = \text{average distance to points in the next nearest cluster}$$

We define $S$ as

neg: $A > B$

$$S = \frac{B - A}{\max(A, B)}$$