Unit 5: Machine Learning & AI

Dave Abel

March 2nd, 2016

Midterm

- In class, Monday, March 21st.
- 50 Minutes long.
- Mostly short answer similar to homeworks. Some multiple choice.
- Covers Unit 0 (Binary) through Unit 6 (Vision/NLP).
- Review Session, Saturday, March 19th, 4:30pm. Room TBD.
- Material from Lecture, Homeworks, Labs.
- I'll release some practice questions and sheets that summarize each unit.

Machine Learning

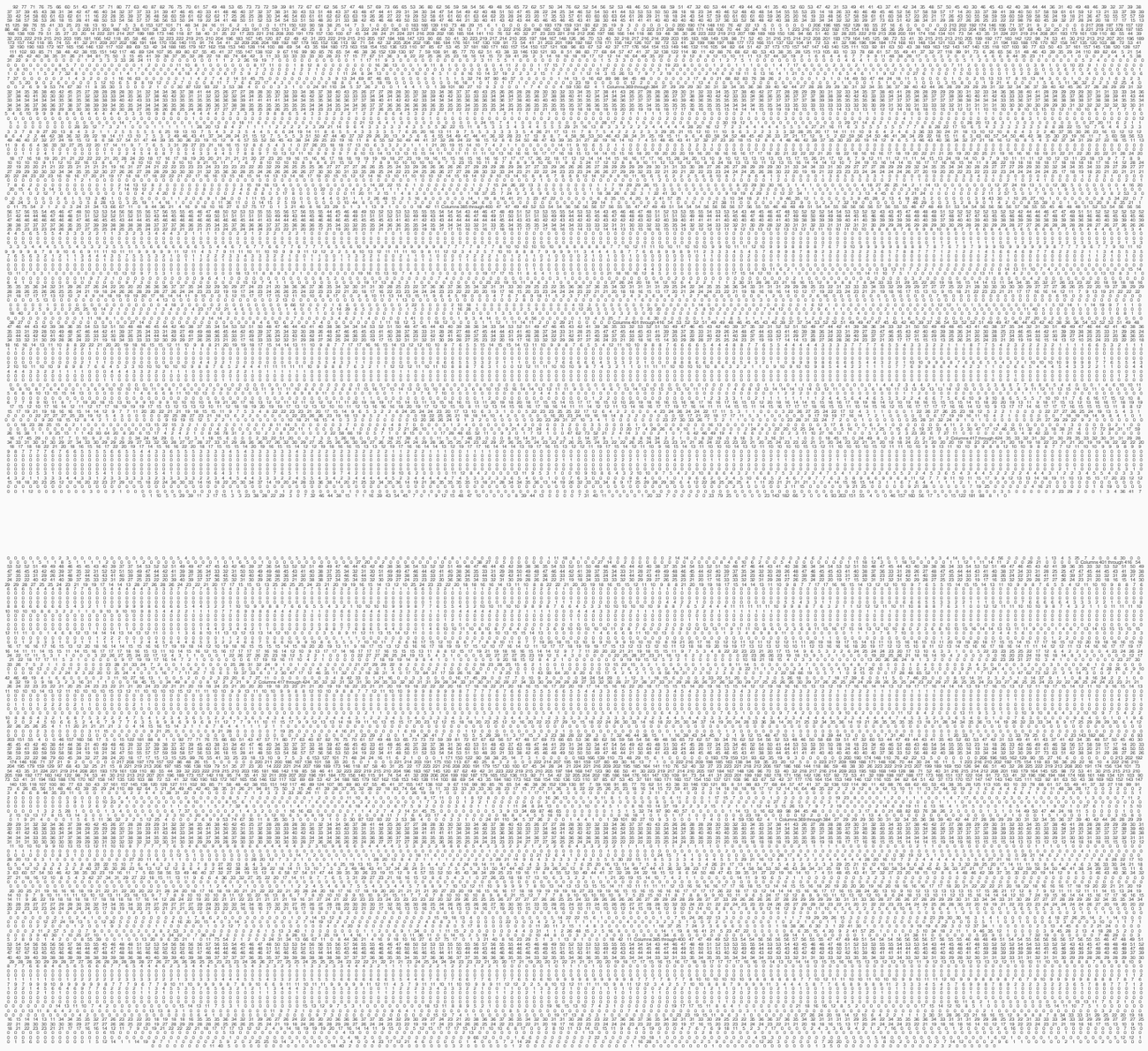
- Problem: Classification
  - INPUT: Some labeled training data, set of labels.
  - OUTPUT: A classifier.
- Algorithm: Memorize and Guess
  - Make a classifier that remembers all of the training data's labels.
  - Given a new item, if it's the same as one of the training data, then report that training data's item. If not, guess randomly.

Machine Learning: Features

Dog

Not a Dog

Machine Learning: Features



???

???

Machine Learning: Features

Learning that “the top left pixel has 57/100 green”, basically says nothing about whether or not there’s a dog in the image.

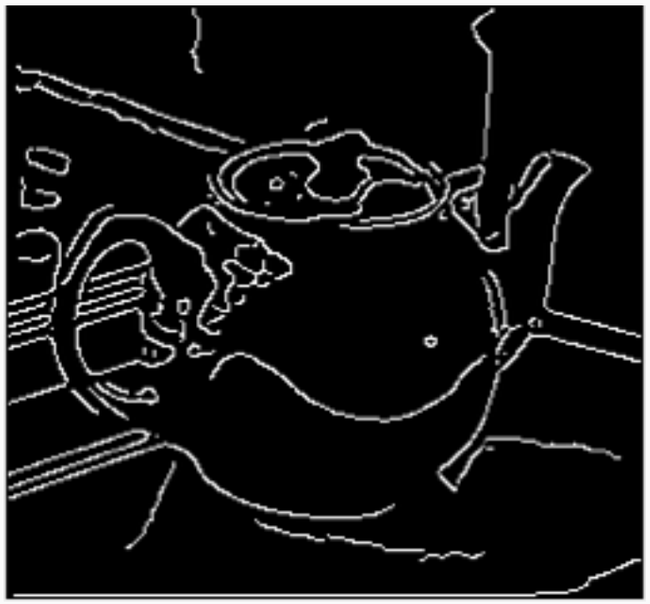
Conclusion: Pixel values are pretty meaningless on their own

Machine Learning: Features

→ Dog

→ Not a Dog

Machine Learning: Features



Dog

Not a Dog

Machine Learning: Features

Location/
Description of
edges

Dog

Location/
Description of
edges

Not a Dog

Machine Learning: Features

Has Eyes → Dog

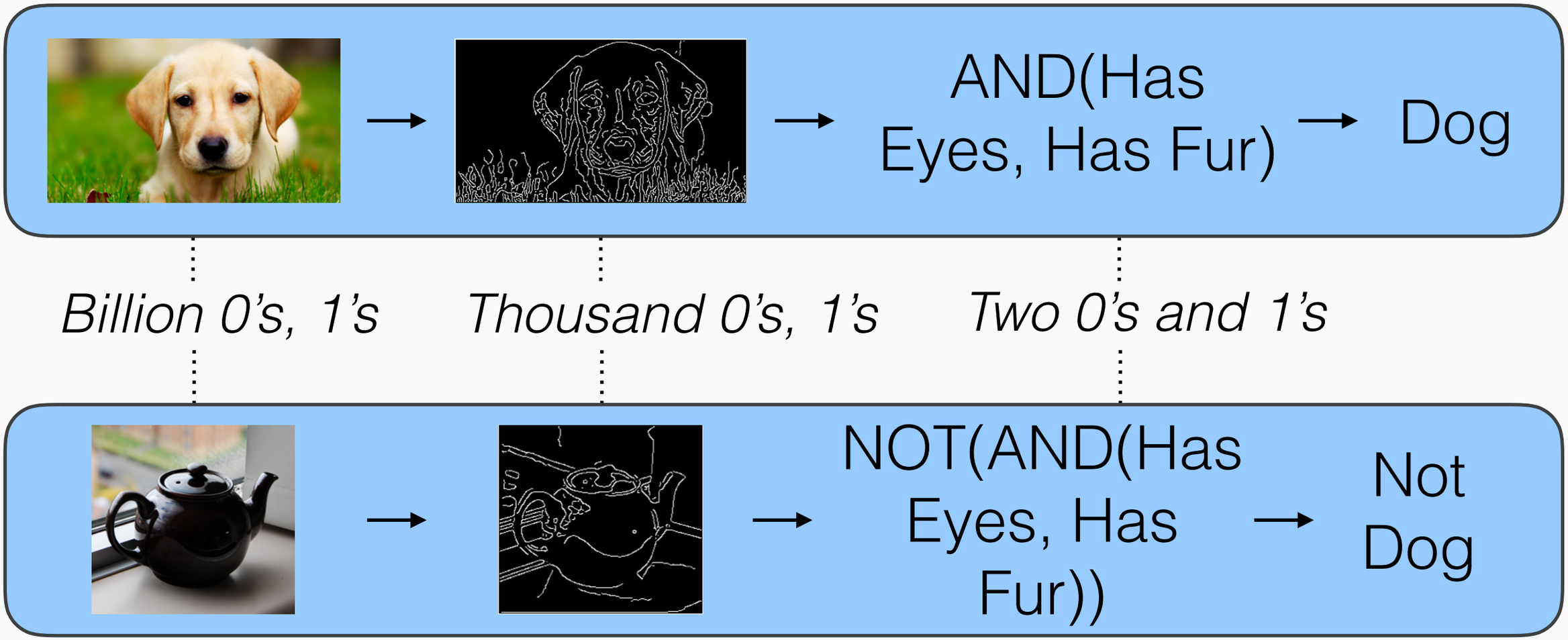
Does Not Have Eyes → Not a Dog

Machine Learning: Features

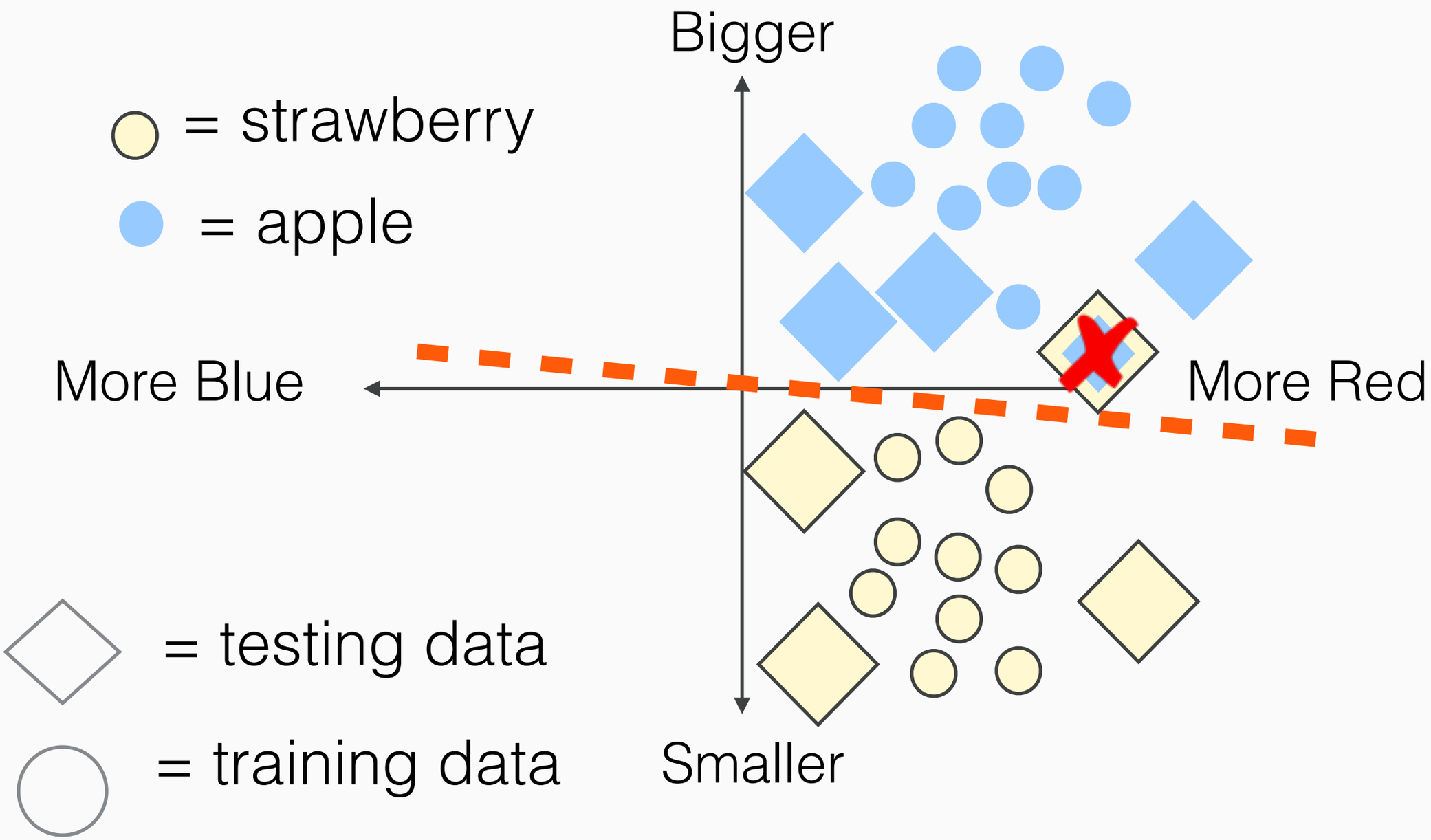
Has Fur → Dog

Does Not Have Fur → Not a Dog

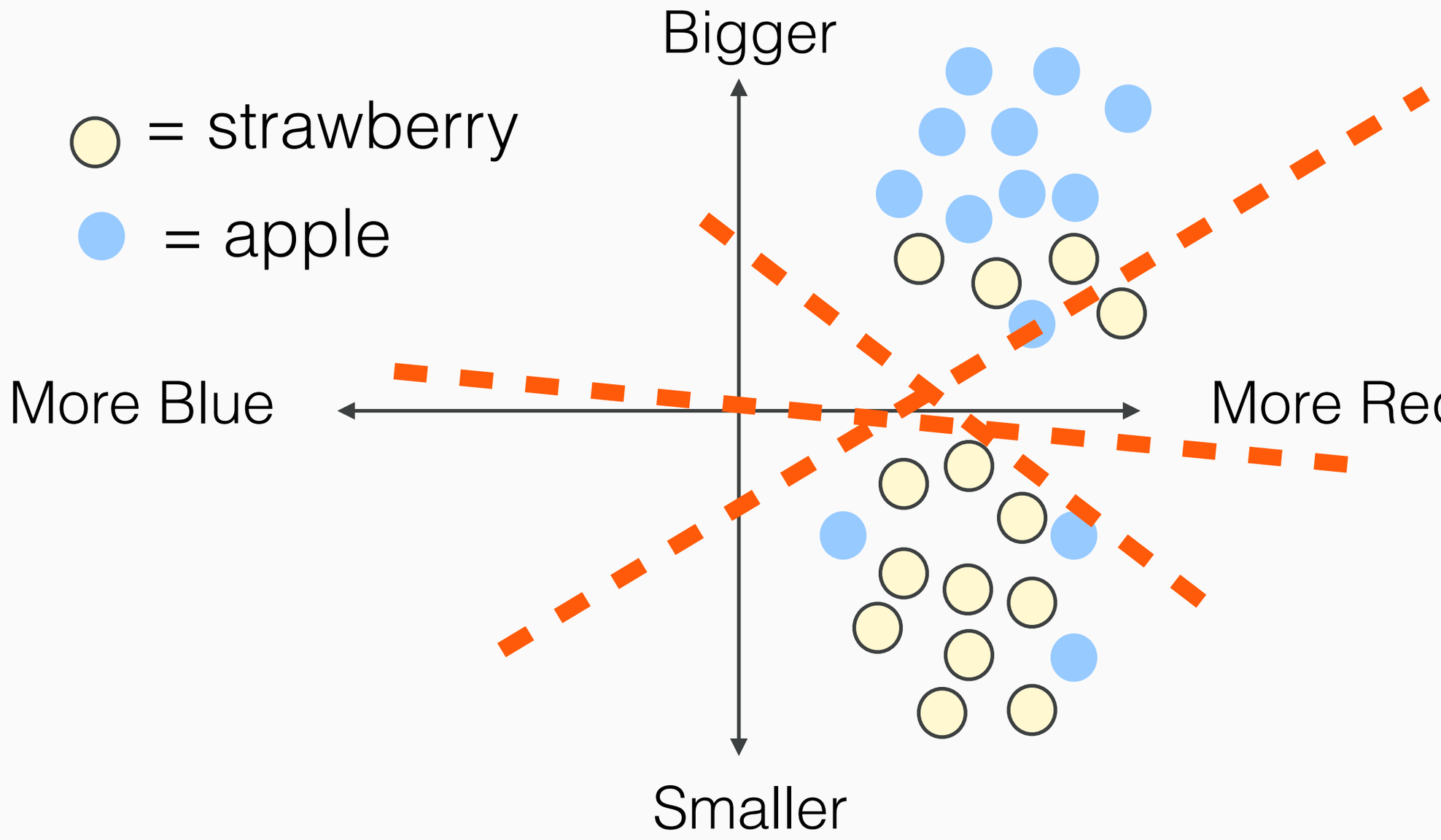
Machine Learning: Features



Classifier


= strawberry
= apple
Bigger
More Blue
More Red
Smaller
= testing data
= training data

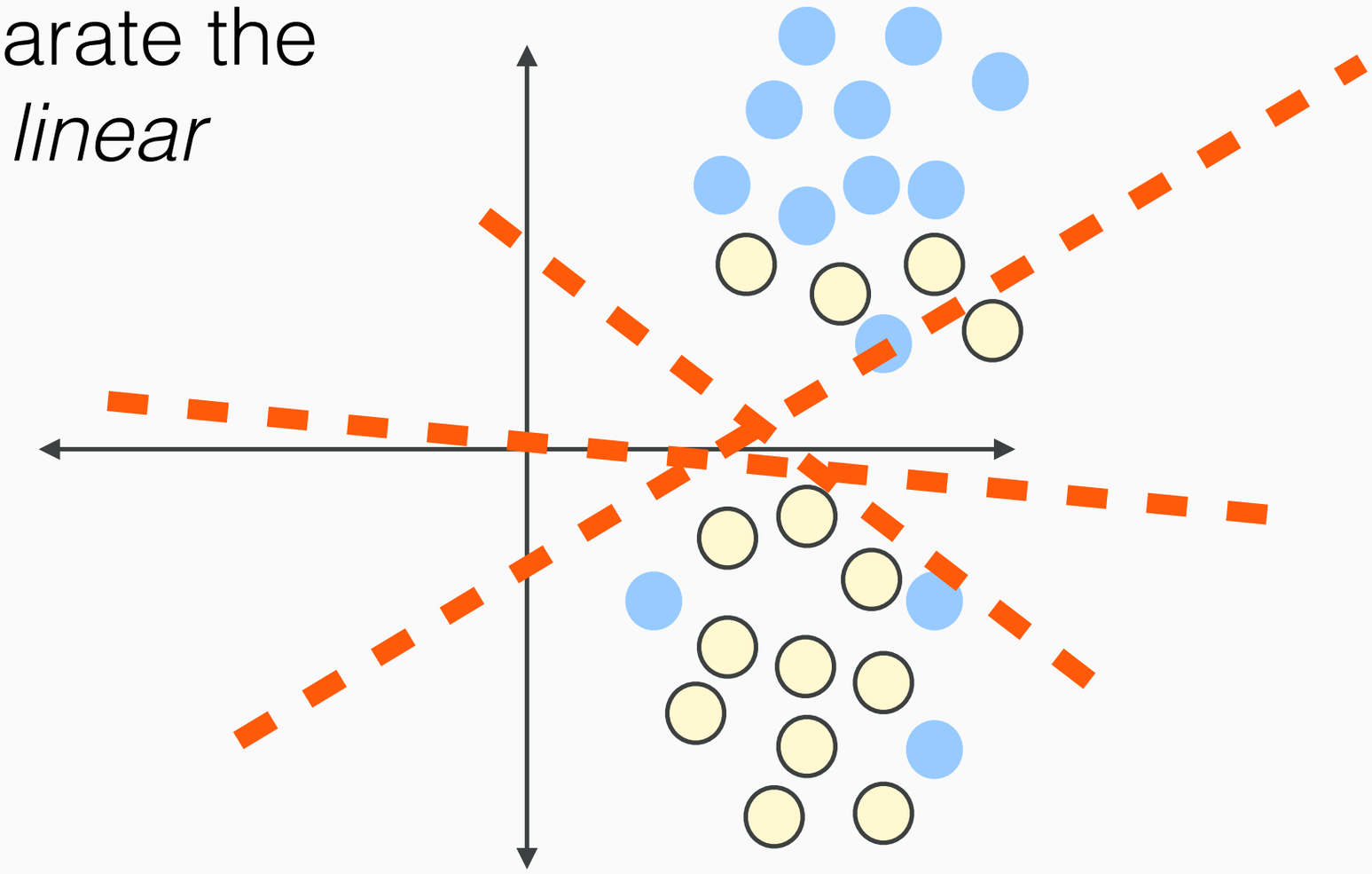
Classifier



Note: There may be huge strawberries! And tiny apples.

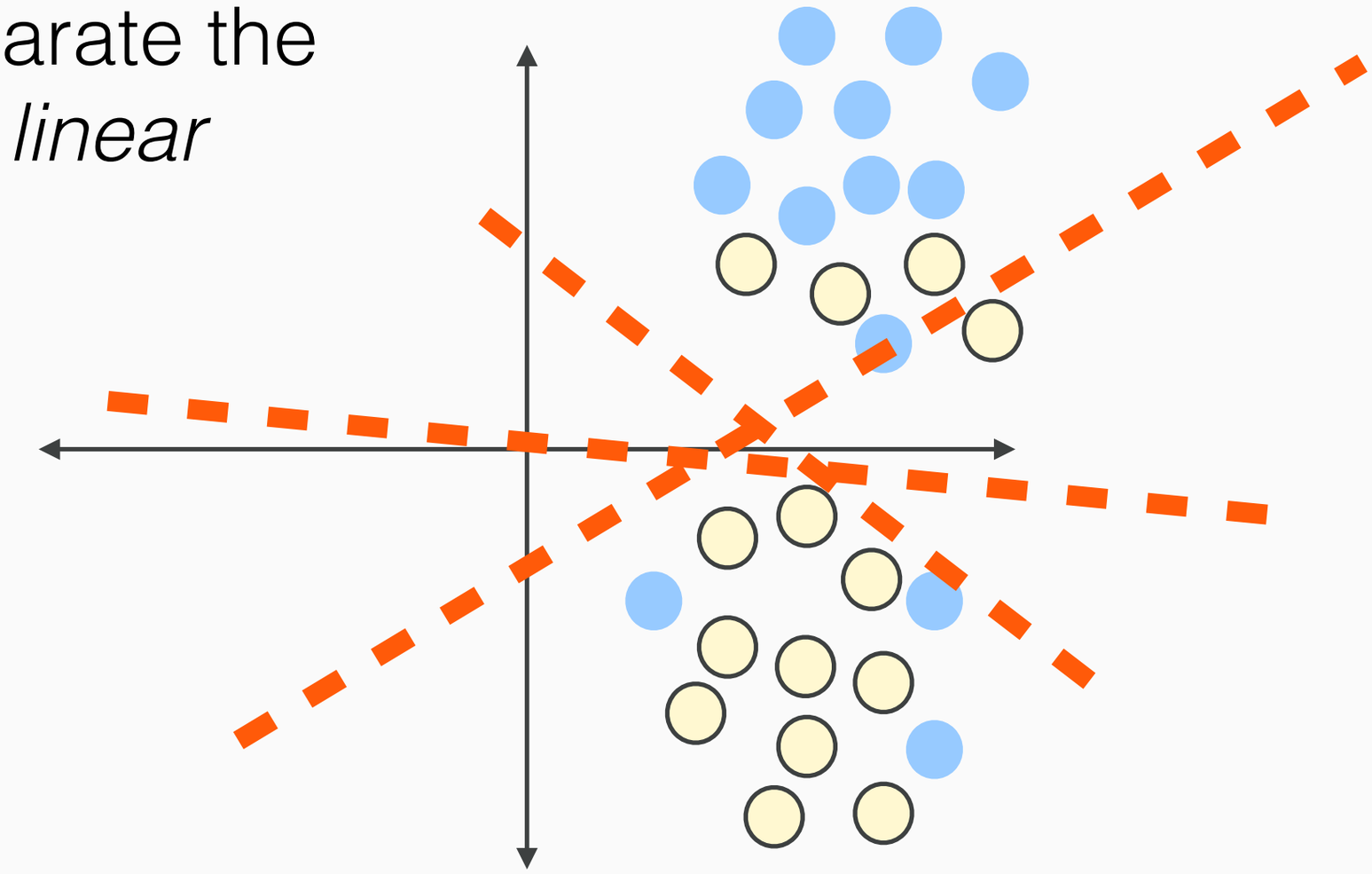
Classifier

Definition: A classifier that draws a line to separate the data is called a *linear classifier*



Classifier

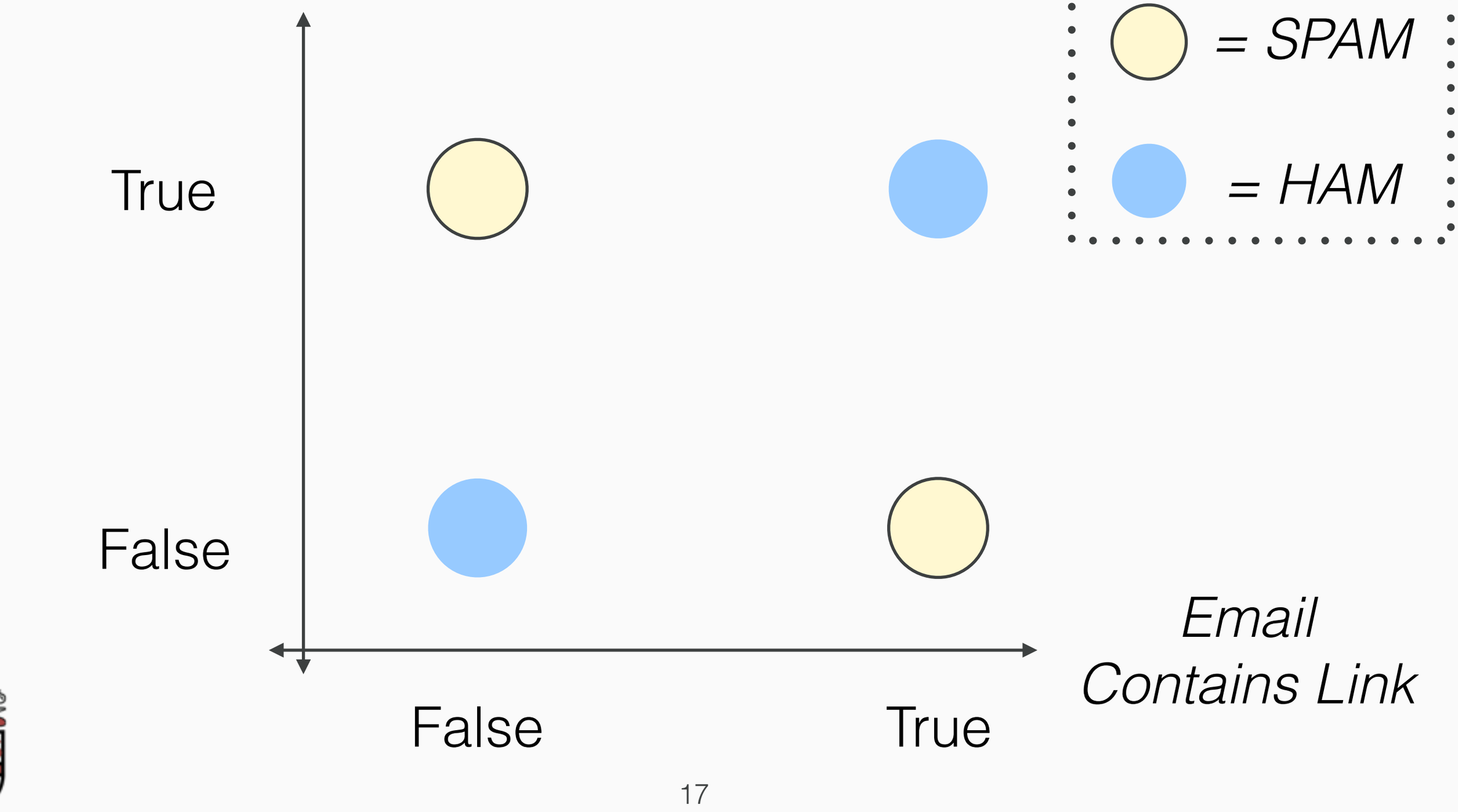
Definition: A classifier that draws a line to separate the data is called a *linear classifier*



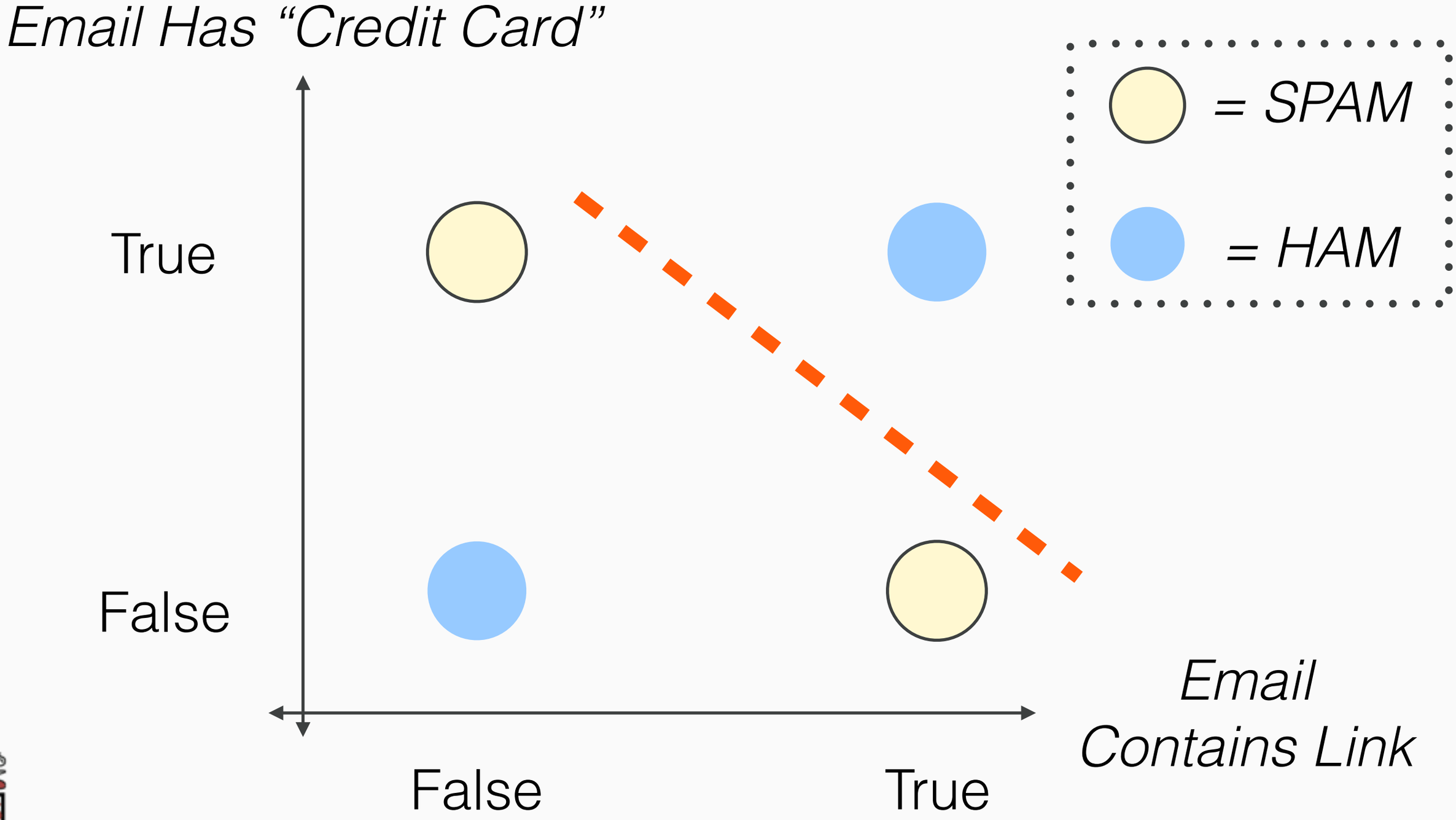
$y = mx+b$

(Learning means finding *m* and *b*!)

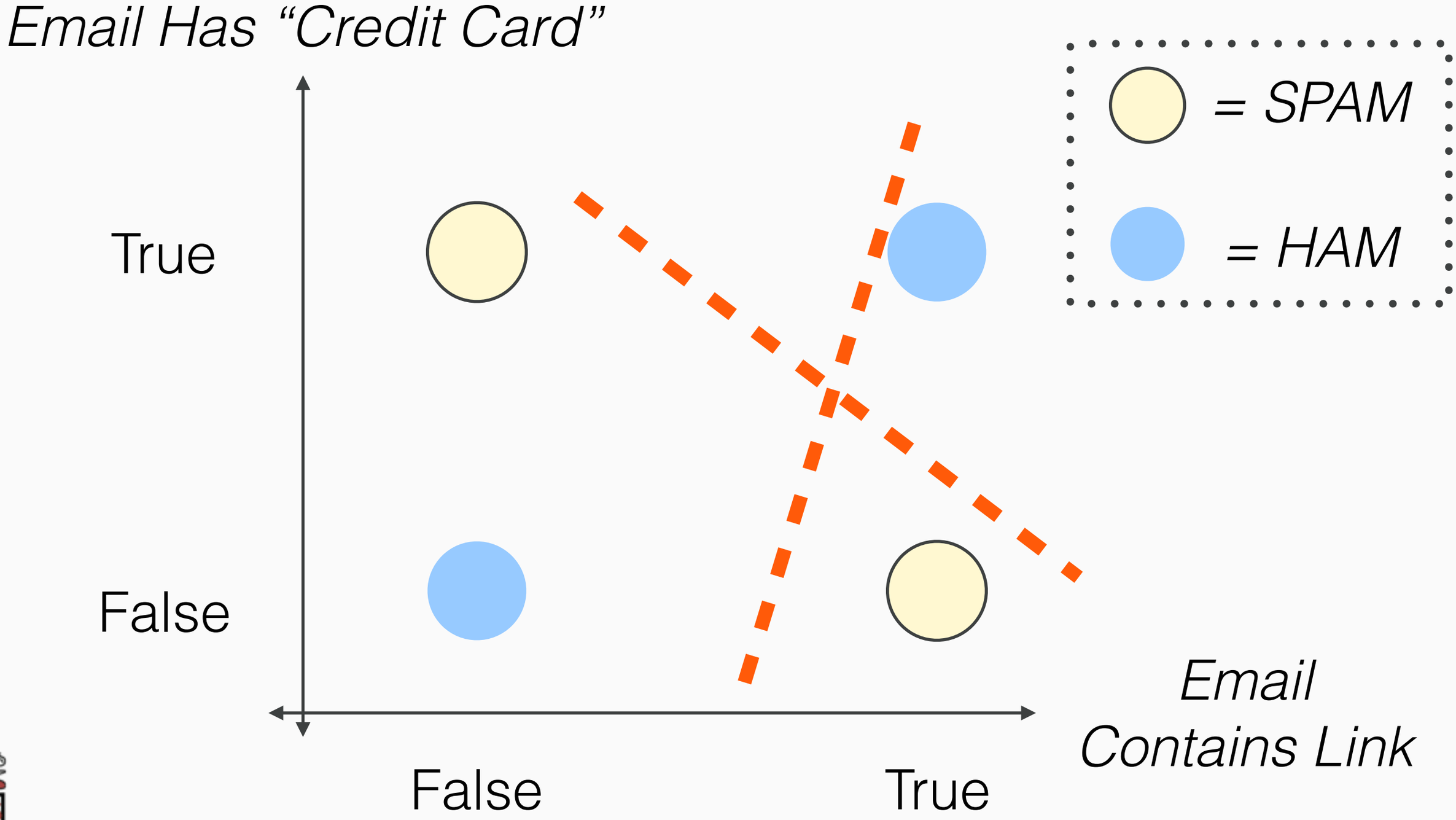
Classifier: SPAM!



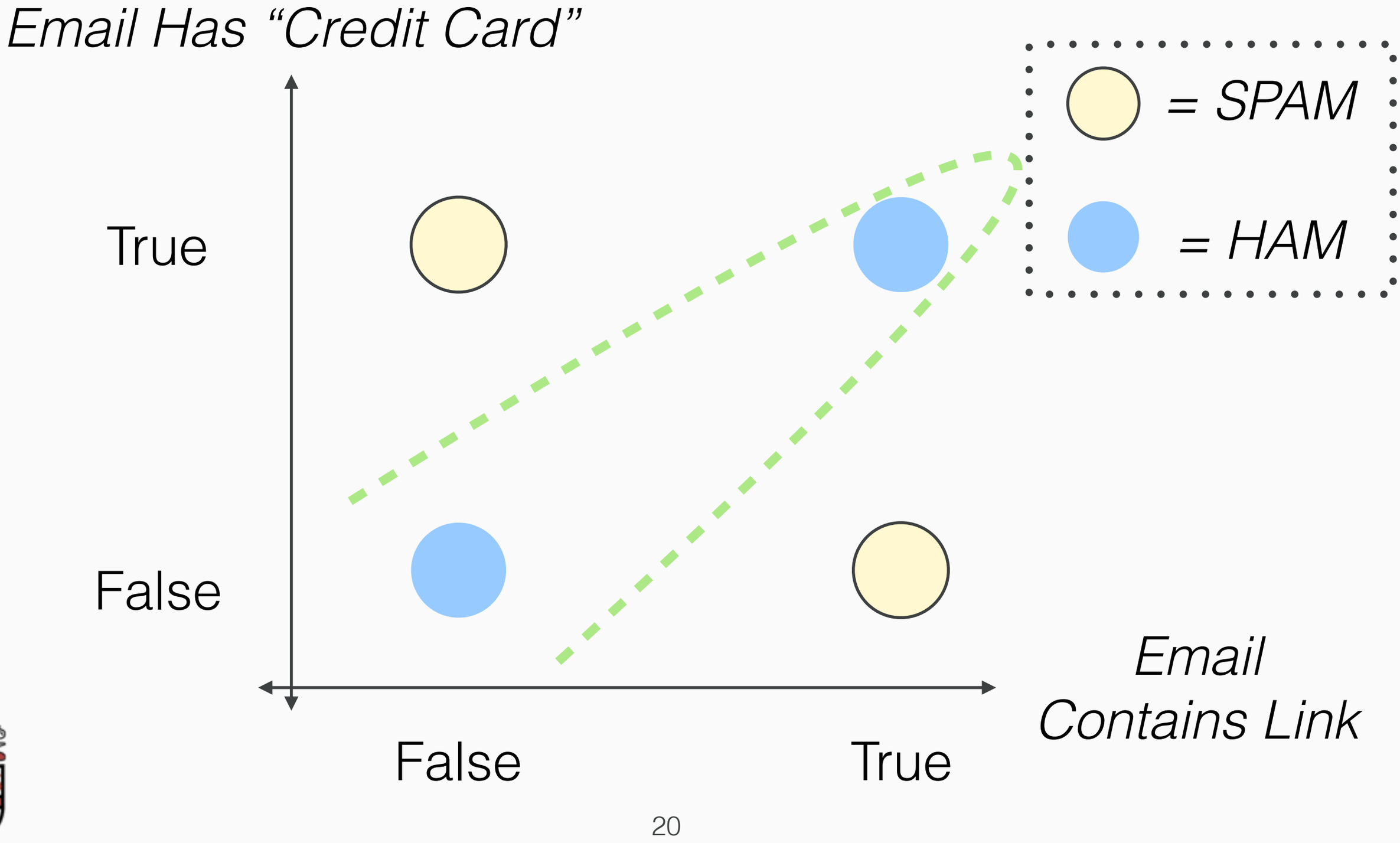
Classifier: SPAM!



Classifier: SPAM!



Classifier: SPAM!



Classifier Shapes

Takeaway: Not all data can be divided into groups by all shapes

Our First Classification Algorithm: Memorize And Guess

1. Memorize every training data-label pair we see.

2. Create a classifier that, when given any item seen in our training data, reports exactly that item's label. If it gets an item it's never seen before, guess the label randomly.

Our First Classification Algorithm: Memorize And Guess

- So, is this Memorize and Guess thing a good idea?
- Sure! If you're guaranteed to see just about every possible data point during training.
- Well that's crazy…
- Conclusion: no. Memorize and Guess is a bad idea.
- Q: Can we do better?
- A: Of course!

Our Second Classification Algorithm: Nearest Neighbor

- **Idea:** instead of randomly guessing when we haven't seen an item before, guess the label of the thing that is most similar to it!

Our Second Classification Algorithm: Nearest Neighbor

- **Idea:** instead of randomly guessing when we haven't seen an item before, guess the label of the thing that is most similar to it!

Strawberry
(known from training)

Our Second Classification Algorithm: Nearest Neighbor

- **Idea:** instead of randomly guessing when we haven't seen an item before, guess the label of the thing that is most similar to it!

Strawberry
(known from training)

Given this item to classify

Our Second Classification Algorithm: Nearest Neighbor

- **Idea:** instead of randomly guessing when we haven't seen an item before, guess the label of the thing that is most similar to it!

Strawberry
(known from training)

Given this item to classify

They're super similar!

Our Second Classification Algorithm: Nearest Neighbor

- **Idea:** instead of randomly guessing when we haven't seen an item before, guess the label of the thing that is most similar to it!

1. Memorize the label of every training data we get.

2. Create a classifier that, for a given item X, will do the following:

   I. Find the item most similar to X.

   II. Report this similar item's training label.

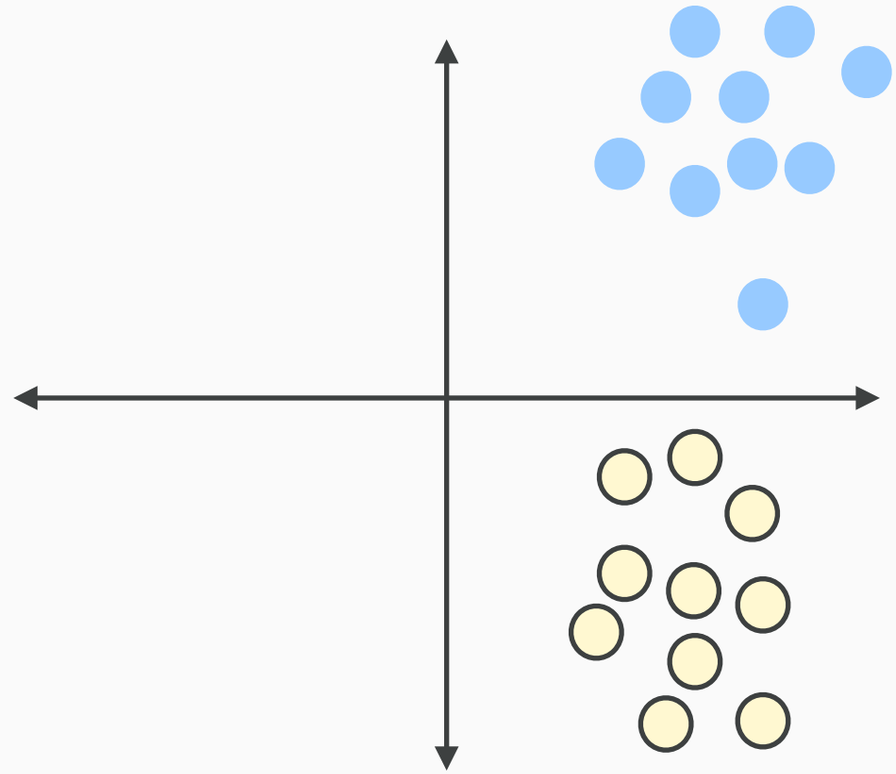
Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.



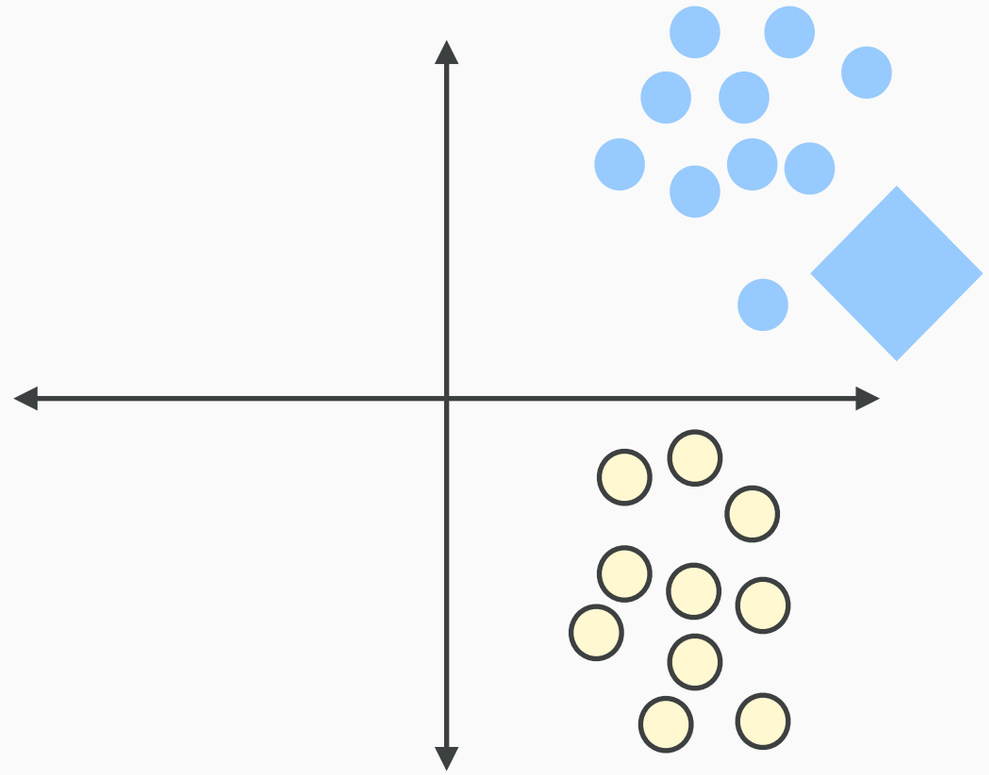
Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.



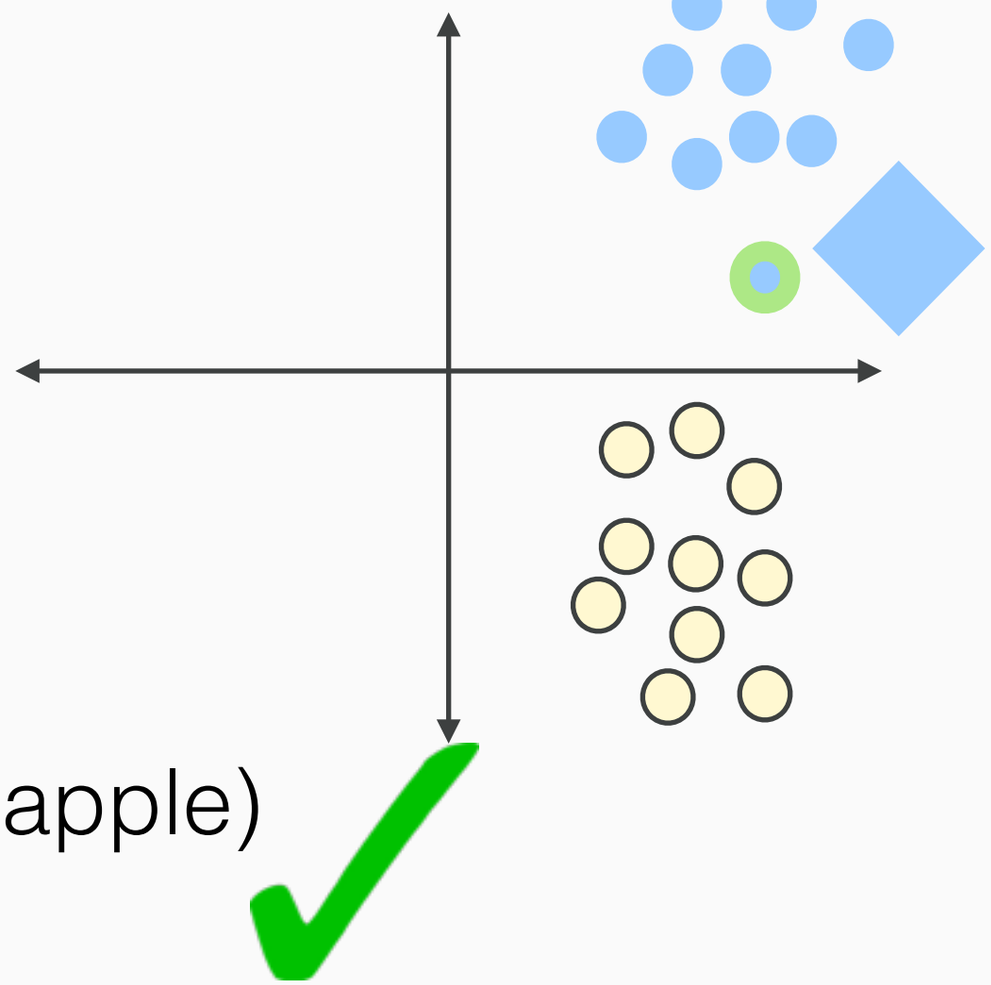
Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.



Label guess: blue (apple)

Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

Q1: Halts?

Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

Q1: Halts?

A: Yes!

Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

Q2: Correct?

A: Again, correctness in learning is tricky. If we ask for the *perfect* classifier, we're setting ourselves up for a loss.

Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

Q3: Growth Rate?

A: Memorize each item.

Nearest Neighbor

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

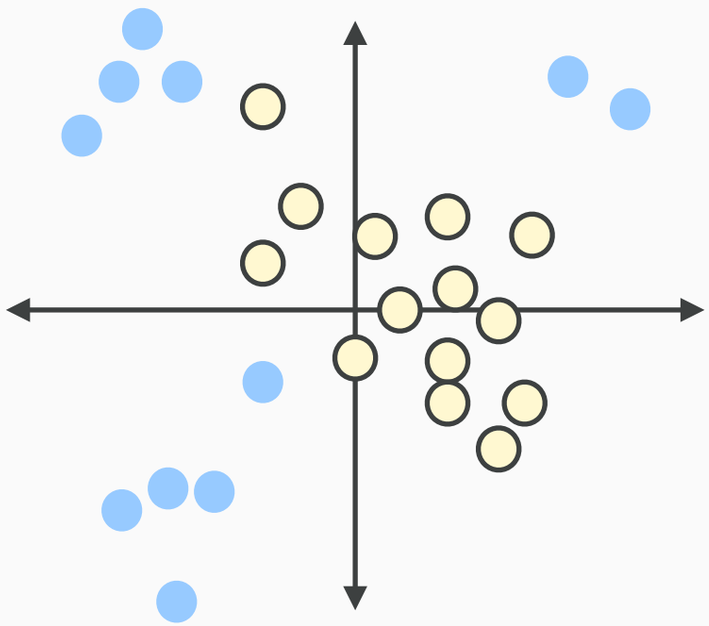
Q3: Growth Rate?

A: Memorize each item.

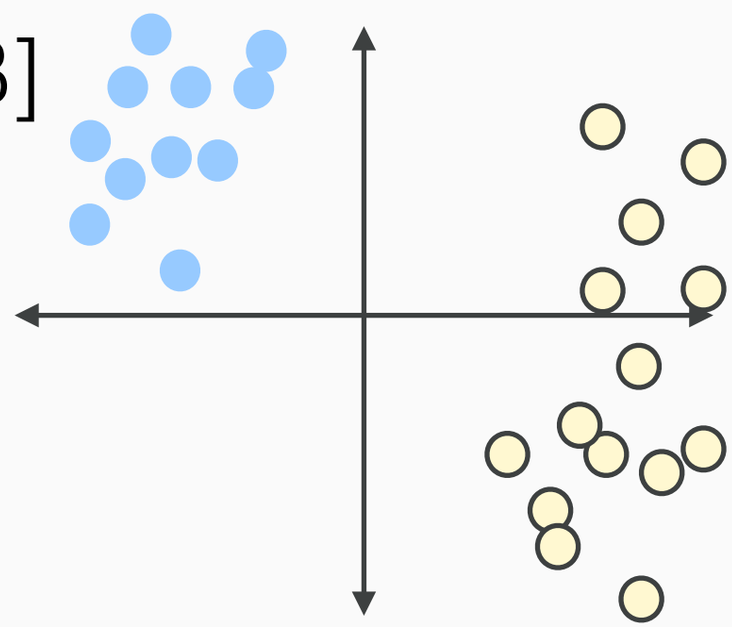
Note: do we care about the growth rate of the classifier, too?

Clicker Question!

[A]



[B]



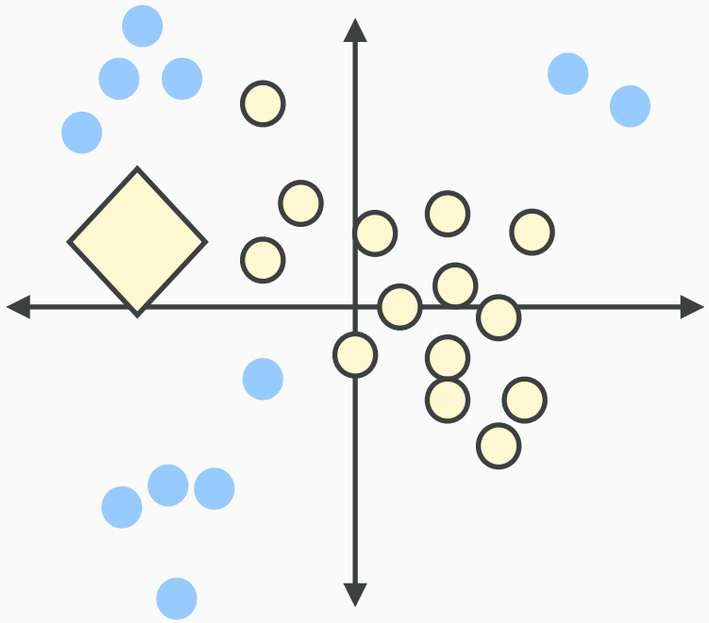
= class one

= class two

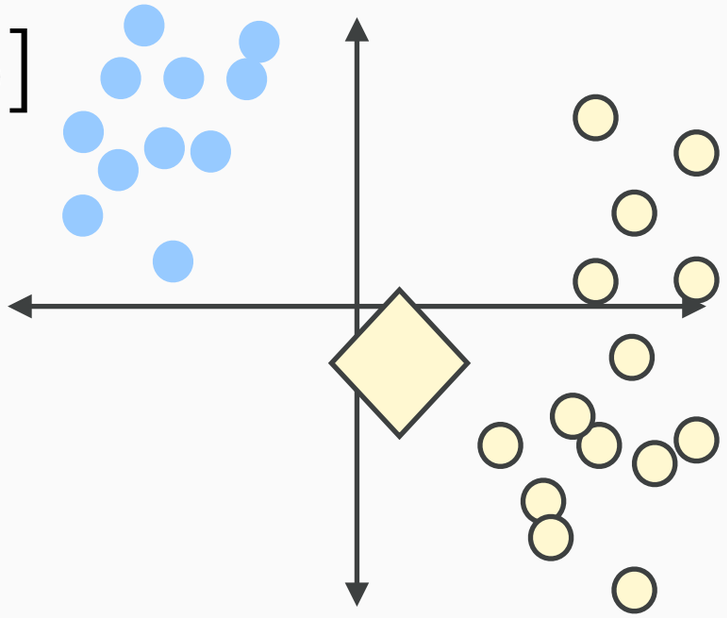
Q: Which dataset would Nearest Neighbors likely do better on?

Clicker Question!

[A]



[B]



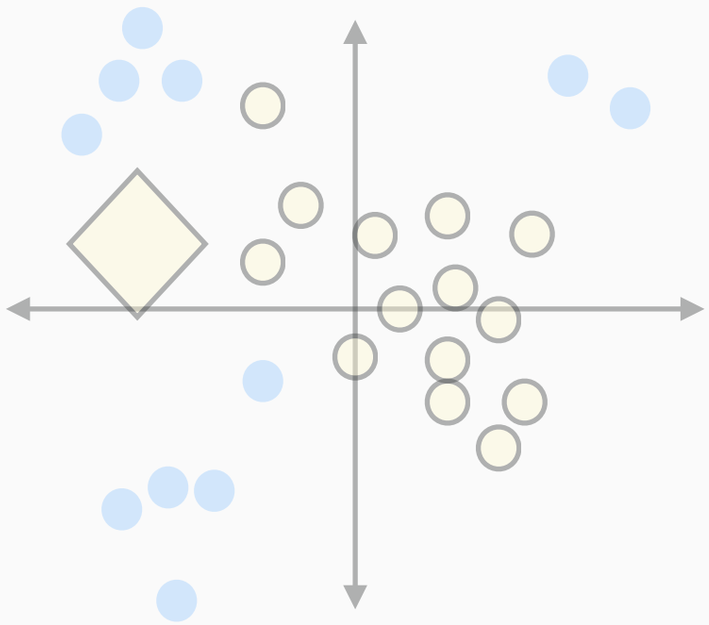
(hint)

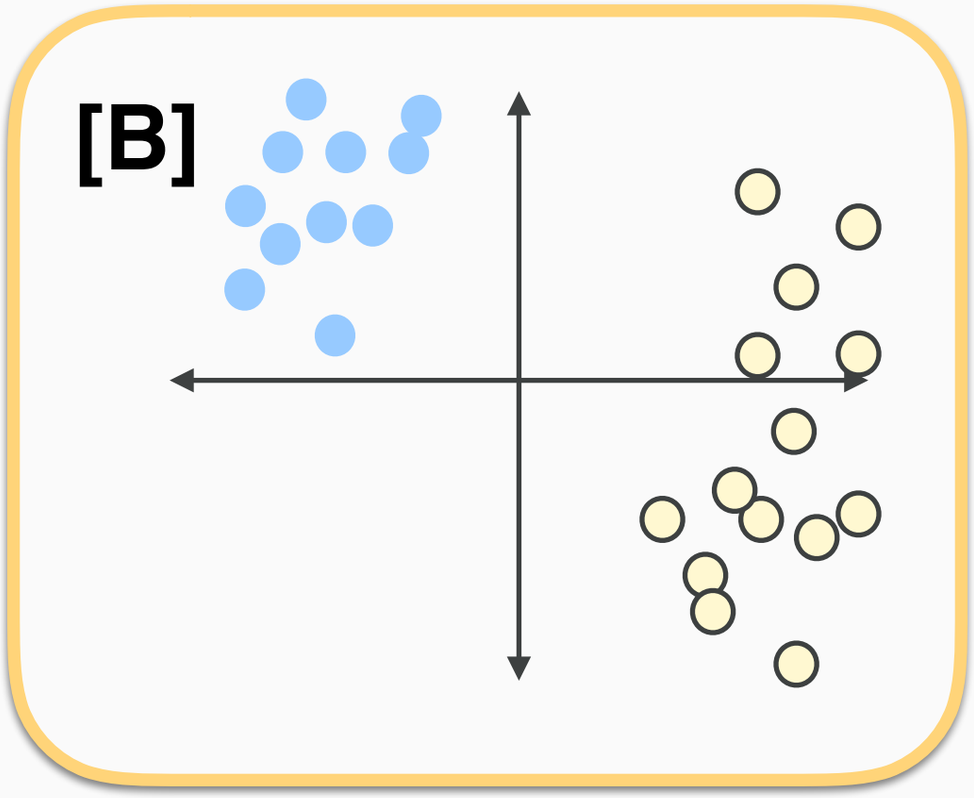
= class one

= class two

Q: Which dataset would Nearest Neighbors likely do better on?

Clicker Answer!





= class one

= class two

Q: Which dataset would Nearest Neighbors likely do better on?

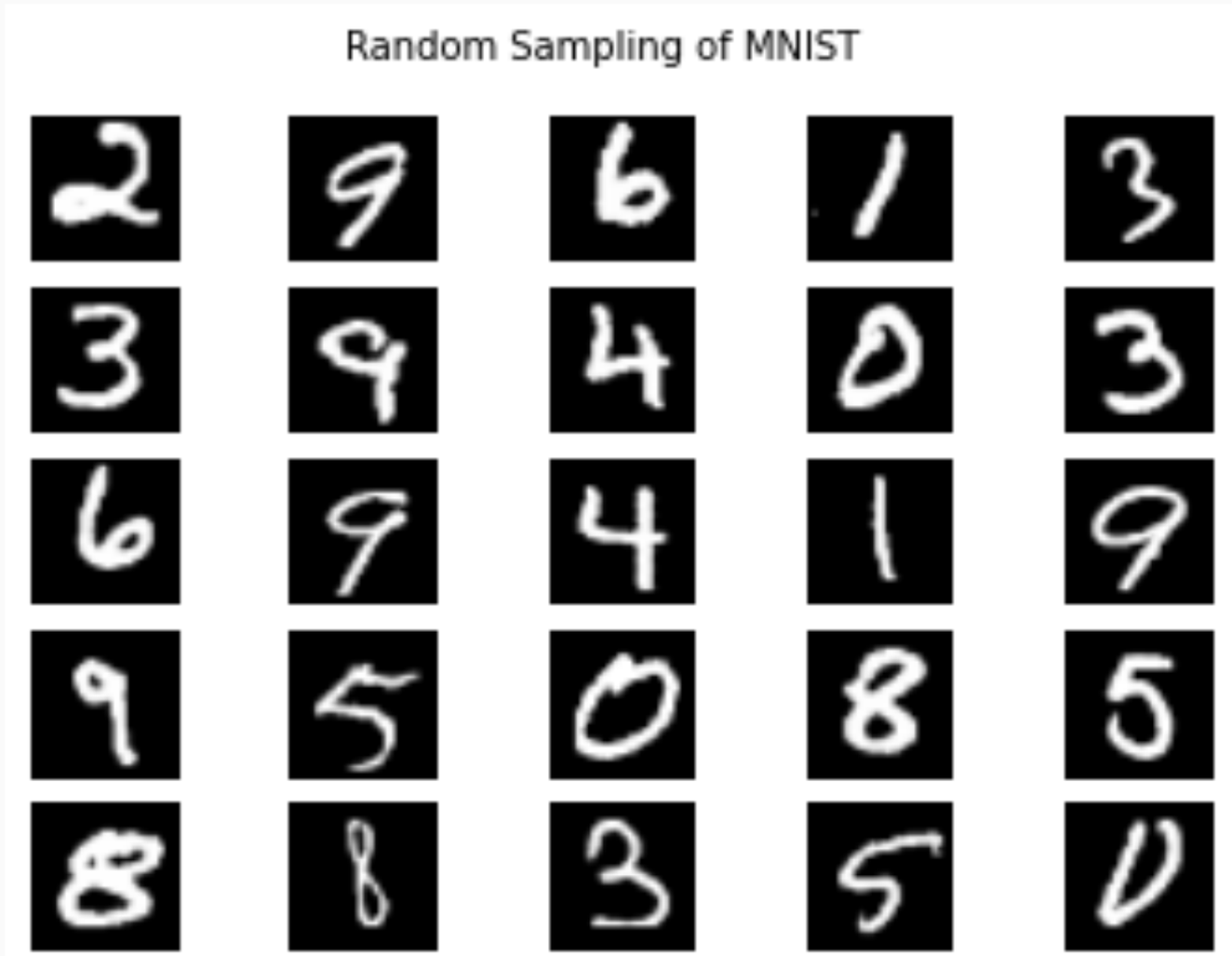
Multiple Classes

- So far we've always been talking about choosing between class one and class two.

- We can extend *all of this* to many classes!

- Consider MNIST:

MNIST



Each object is a 20x20, black and white image.
0 = black, 1 = white.

MNIST

- 70,000 images total.
- Even slightly sophisticated learning algorithms can achieve a nearly perfect score on this dataset (train on 60,000, test on 10,000).
- E.g. an extension of nearest neighbors called K Nearest Neighbors gets 99.5 percent correct.
- So how does it work with multiple classes?

MNIST

: “five”

INPUT:

: “four”

: “two”

MNIST

: “five”

INPUT:

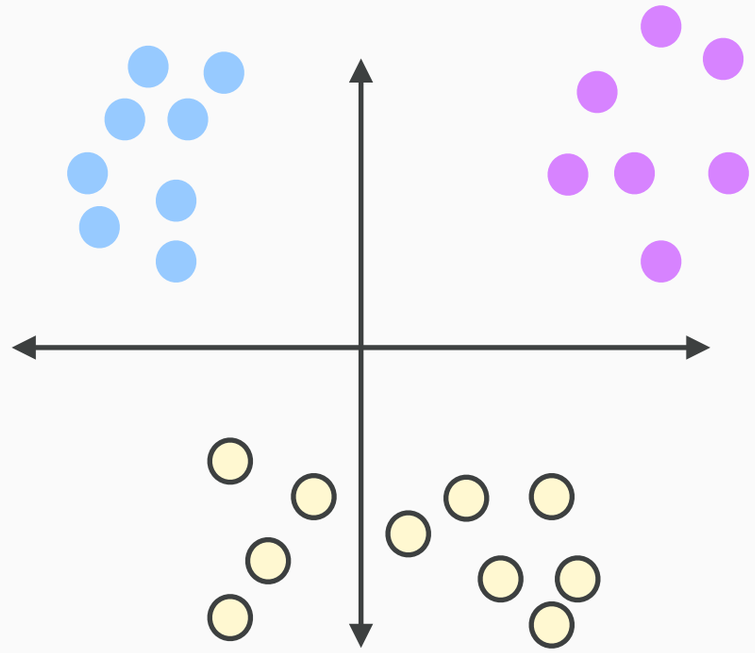
: “four”

: “two”

= class one

= class two

= class three



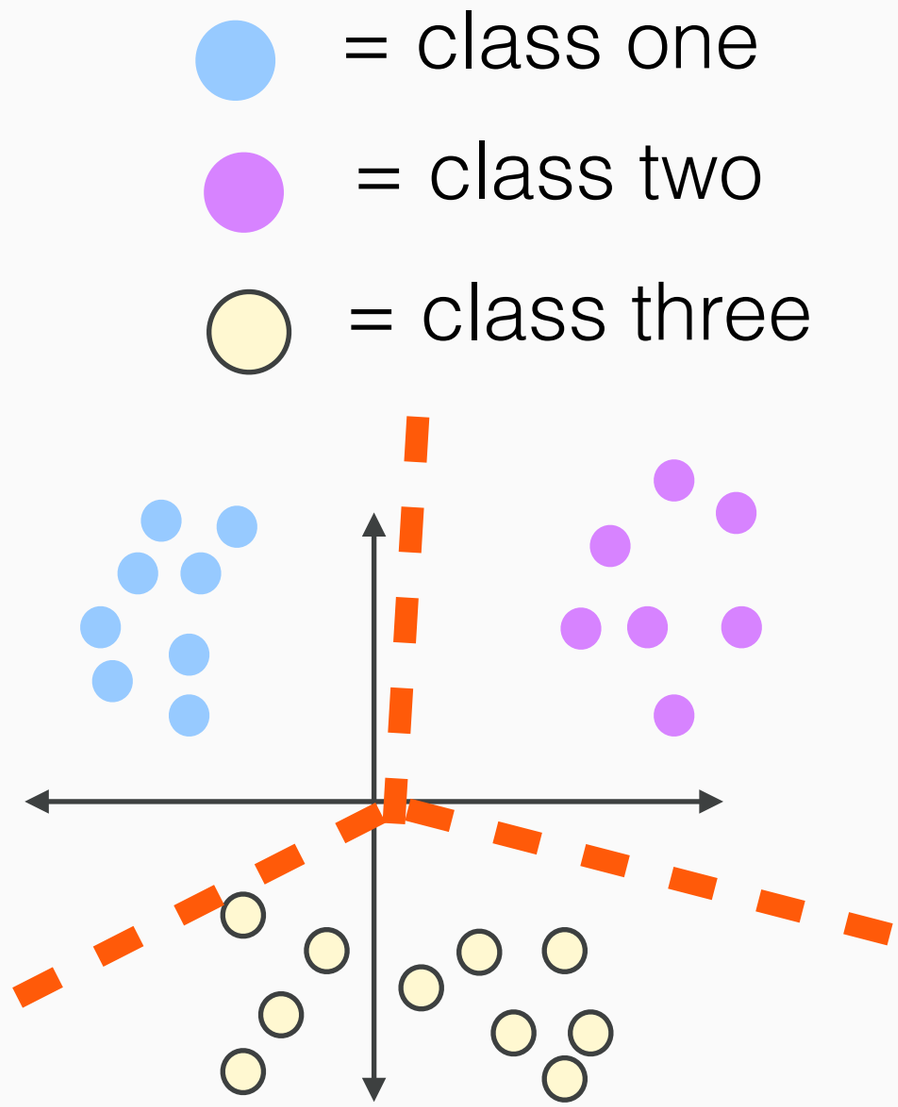
MNIST

: “five”

INPUT:

: “four”

: “two”



Clicker Question!

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

Nearest Neighbor

Q: Does Nearest Neighbor need to change at all to account for more classes (instead of just class one vs class two)?

Clicker Question!

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

[A] Yes!

[B] No!

Nearest Neighbor

Q: Does Nearest Neighbor need to change at all to account for more classes (instead of just class one vs class two)?

Clicker Answer!

1. Memorize the label of every training data we get.

2. Create a classifier that will report, for a given item X, do the following:

I. Find the item most similar to X.

II. Report this similar item's training label.

[A] Yes!

[B] No!

Nearest Neighbor

Q: Does Nearest Neighbor need to change at all to account for more classes (instead of just class one vs class two)?

Overfitting

- **Generalization is the goal of learning.**

- If all we cared about was how well you did on the training data, then the Memorize and Guess algorithm would be amazing!

- We can test how well our learner does by how well its classifier does on the test data relative to the training data (hopefully just as well!).

- In the algorithms we saw, 100% on training. ??% on Test.

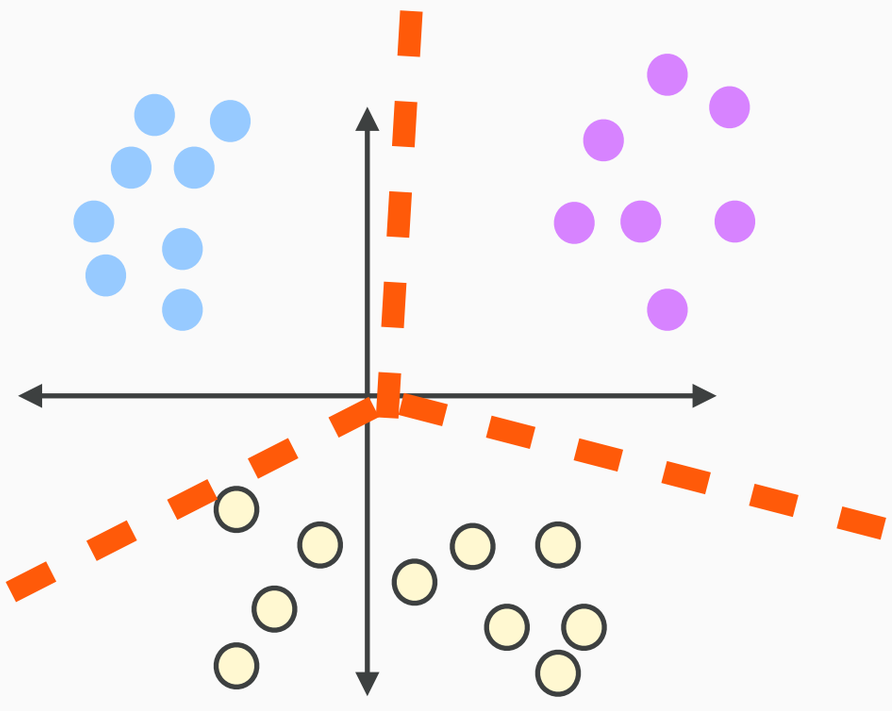
Clicker Question!

= class one

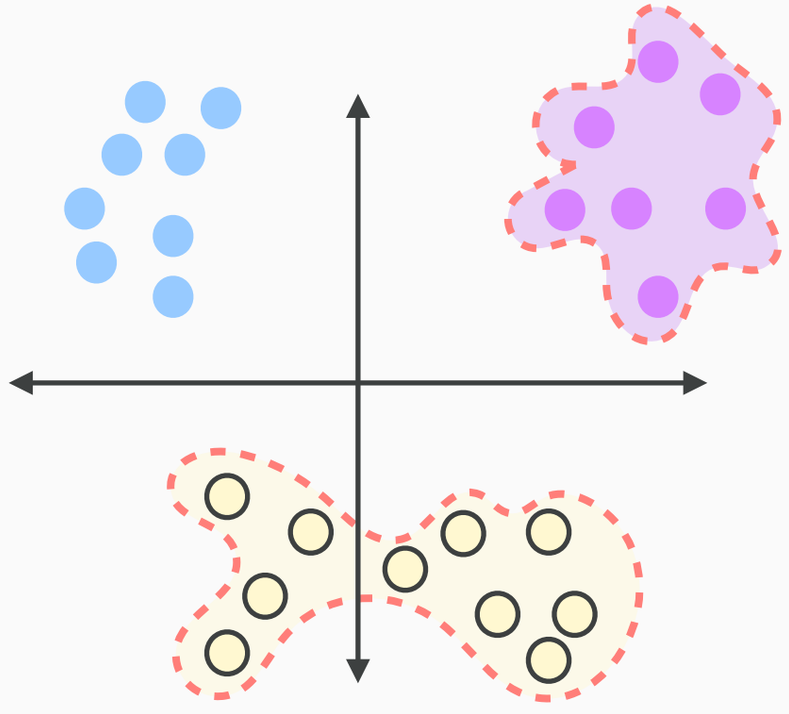
= class two

= class three

Q: Which classifier is overfitting more?



[A]



[B]

Clicker Question!

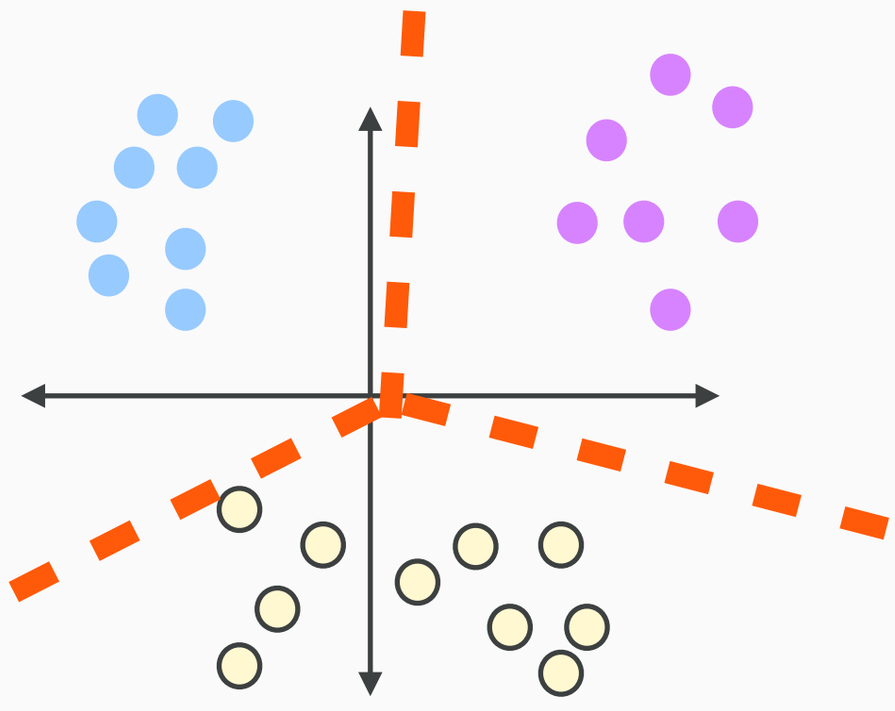
= class one

= class two

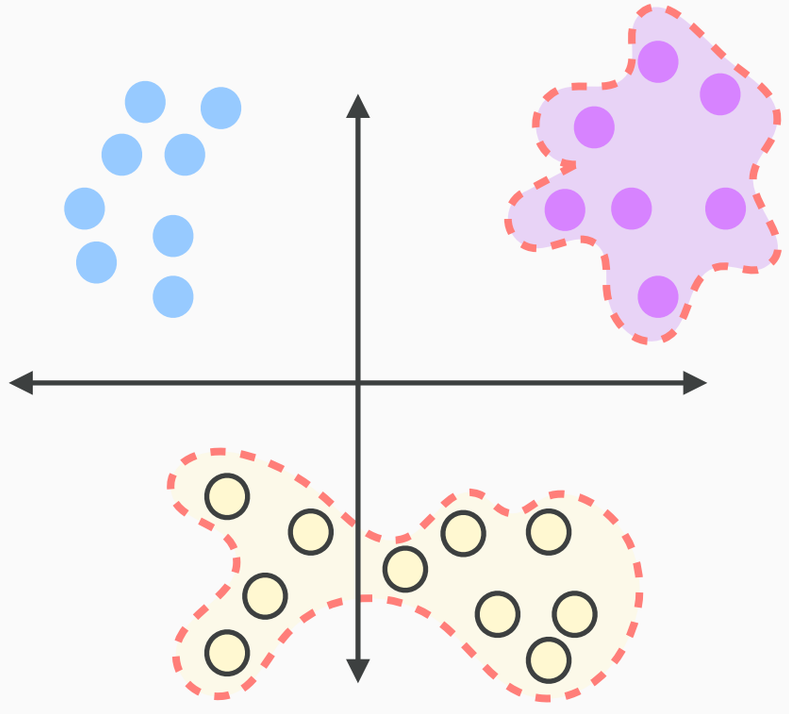
= class three

Q: Which classifier is overfitting more?

Hint: Which one would we expect to perform worse during testing?



[A]



[B]

Clicker Answer!

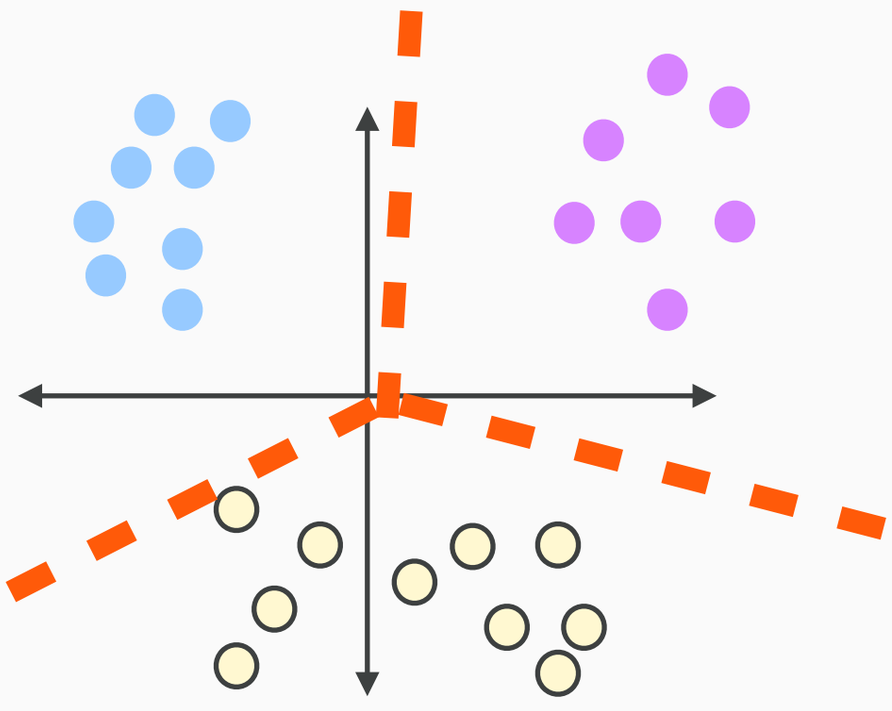
= class one

= class two

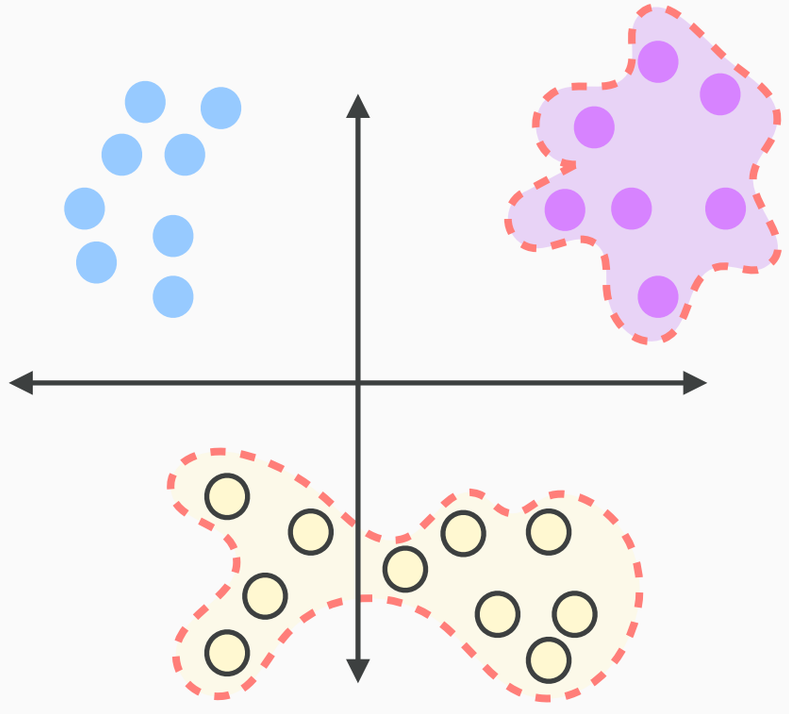
= class three

Q: Which classifier is overfitting more?

Hint: Which one would we expect to perform worse during testing?



[A]



[B]

Clicker Answer!

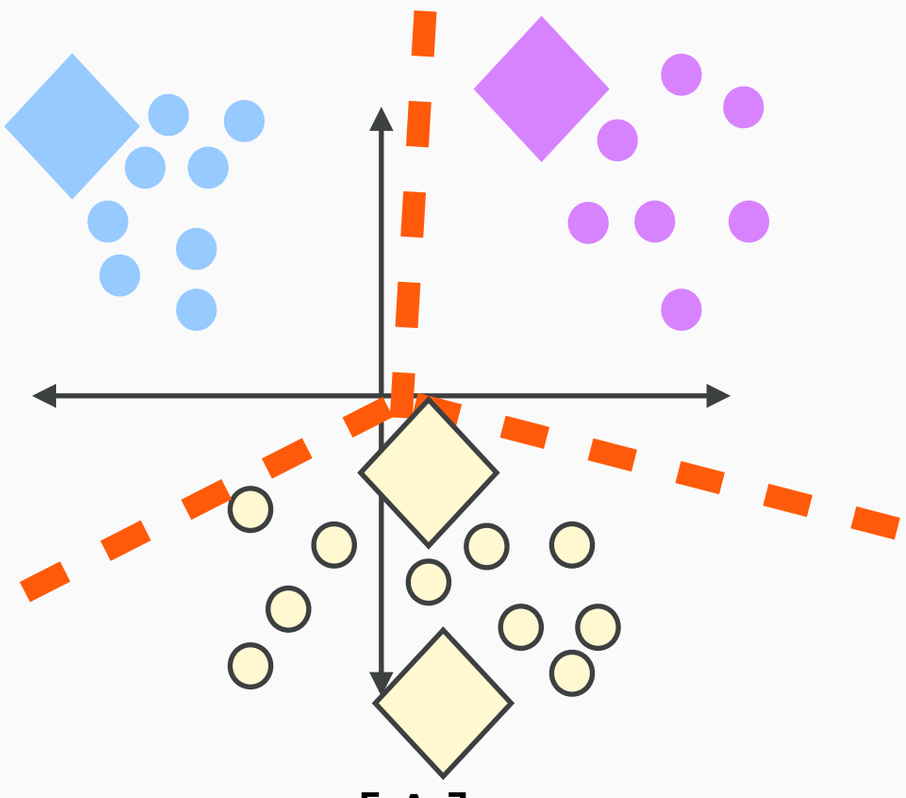
= class one

= class two

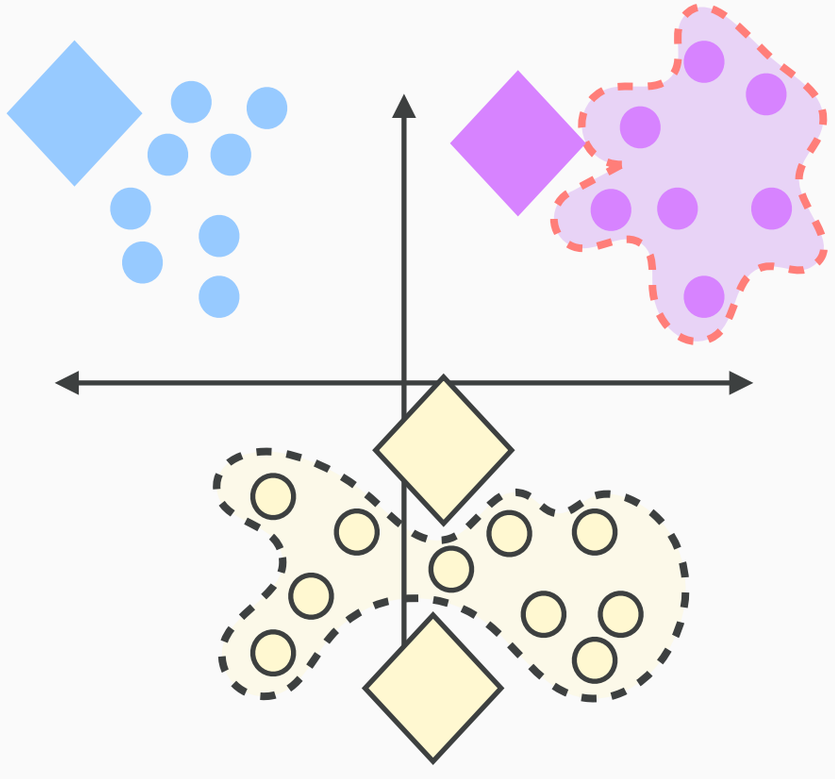
= class three

Q: Which classifier is overfitting more?

Hint: Which one would we expect to perform worse during testing?



[A]



[B]

Clicker Answer!

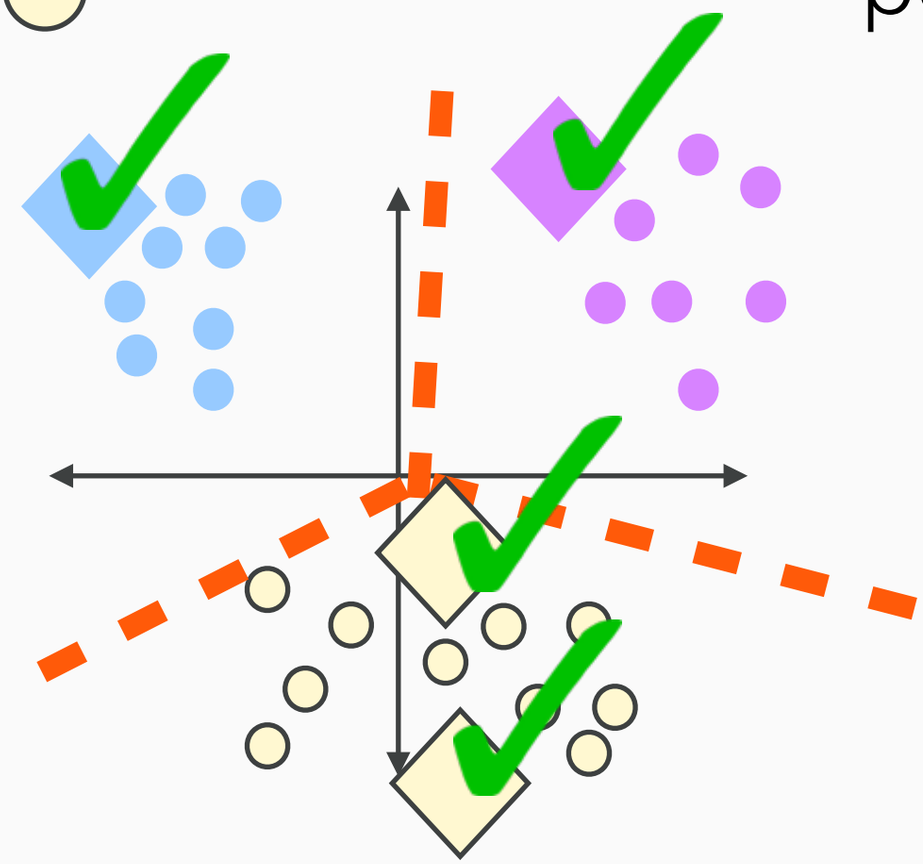
= class one

= class two

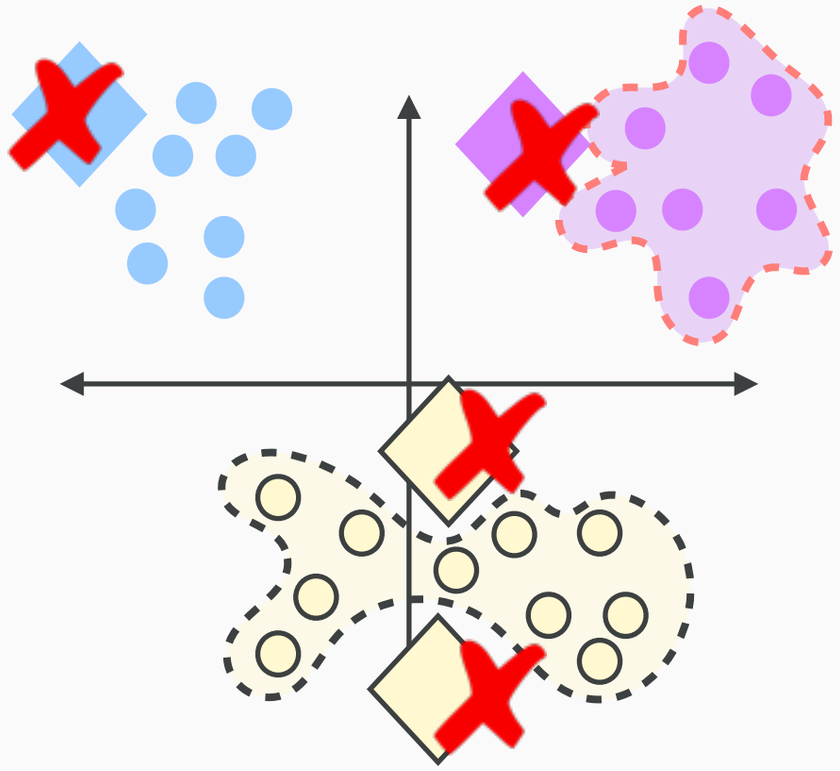
= class three

Q: Which classifier is overfitting more?

Hint: Which one would we expect to perform worse during testing?

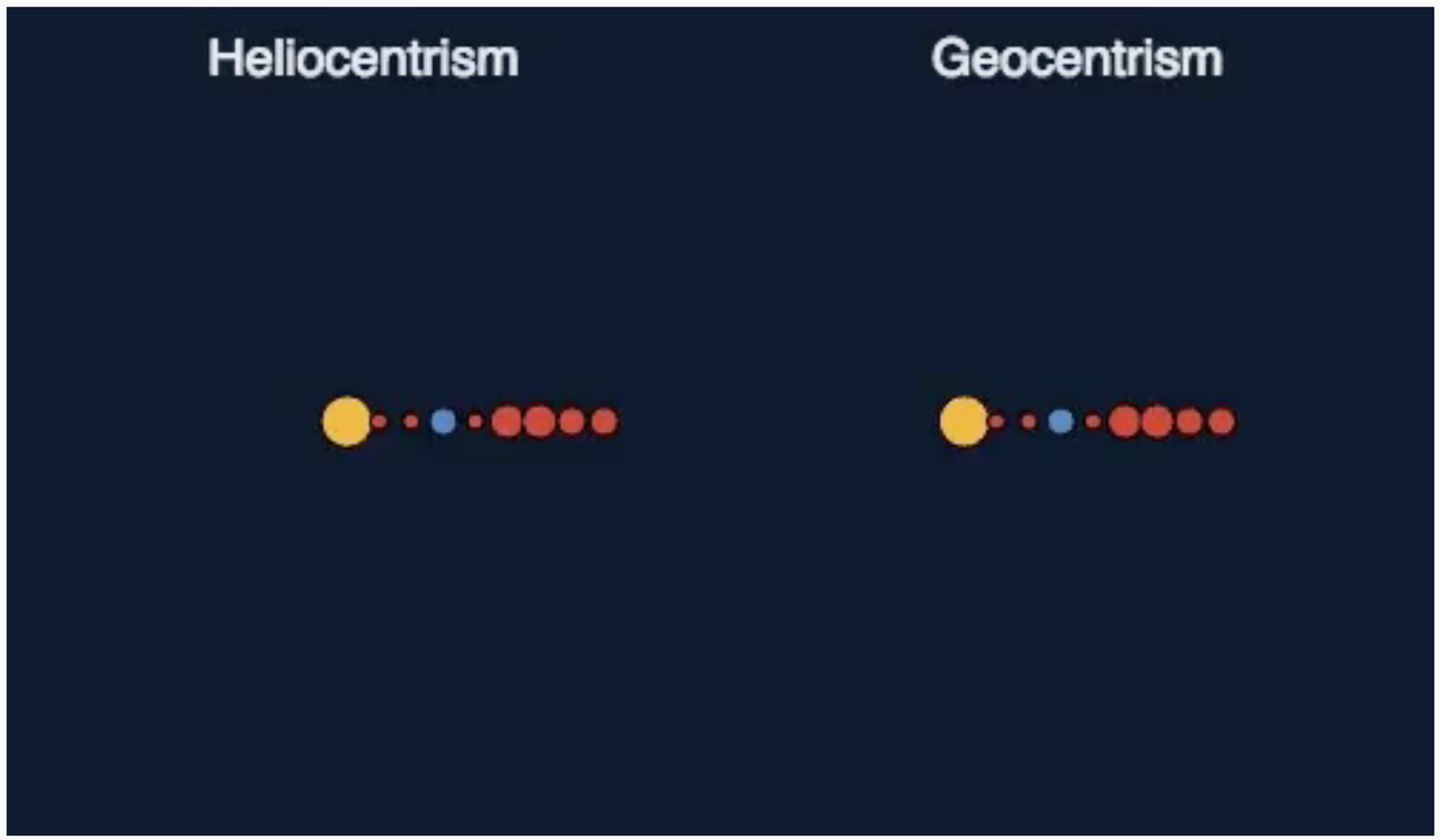


[A]



[B]

Occam's Razor



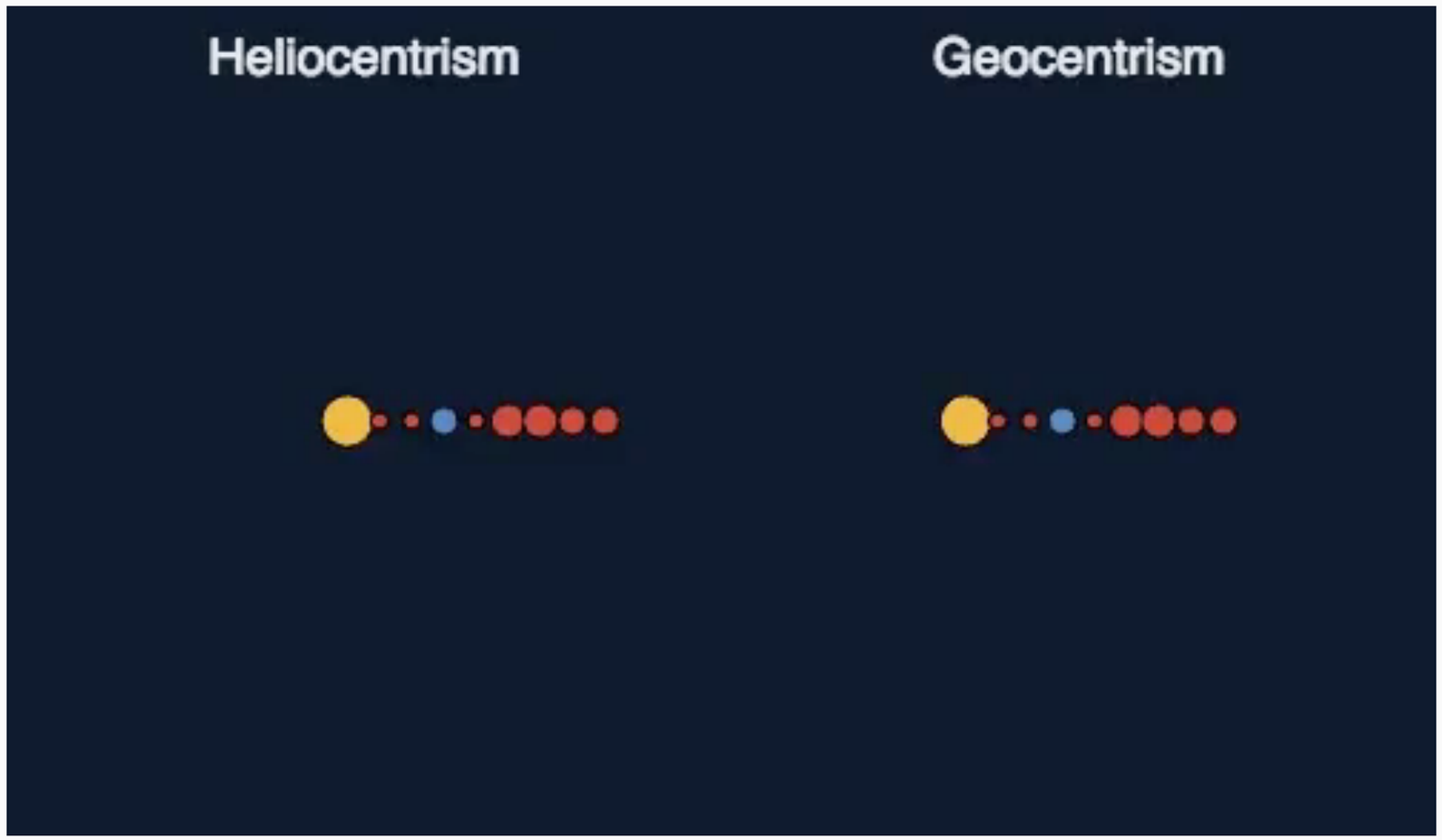
We ought to prefer simpler explanations

Occam's Razor

Isaac Newton

Ptolemy

Bertrand Russell



Ray Solomonoff

Aristotle

William of Ockham

We ought to prefer simpler explanations

Our Final Classification Algorithm: K-Nearest Neighbor

- **Idea:** instead of randomly guessing when we haven't seen an item before, guess the label of some things that are most similar to it!

1. Memorize the label of every training data we get.

2. Create a classifier that, for a given item X, do the following:

   I. Find the K items most similar to X.

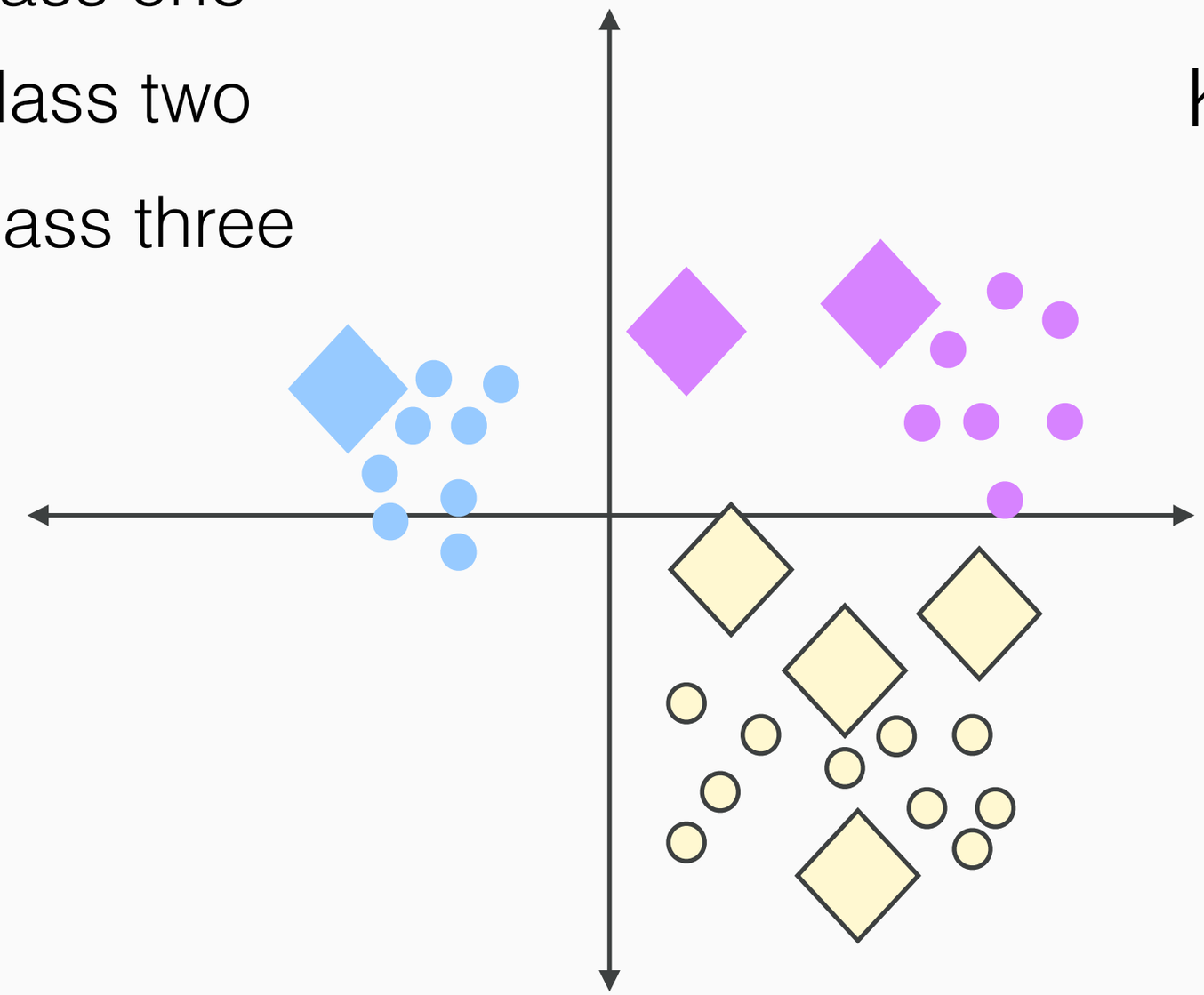
   II. Each of these K items vote on the label for the new item

K Nearest Neighbor

= class one

= class two

= class three

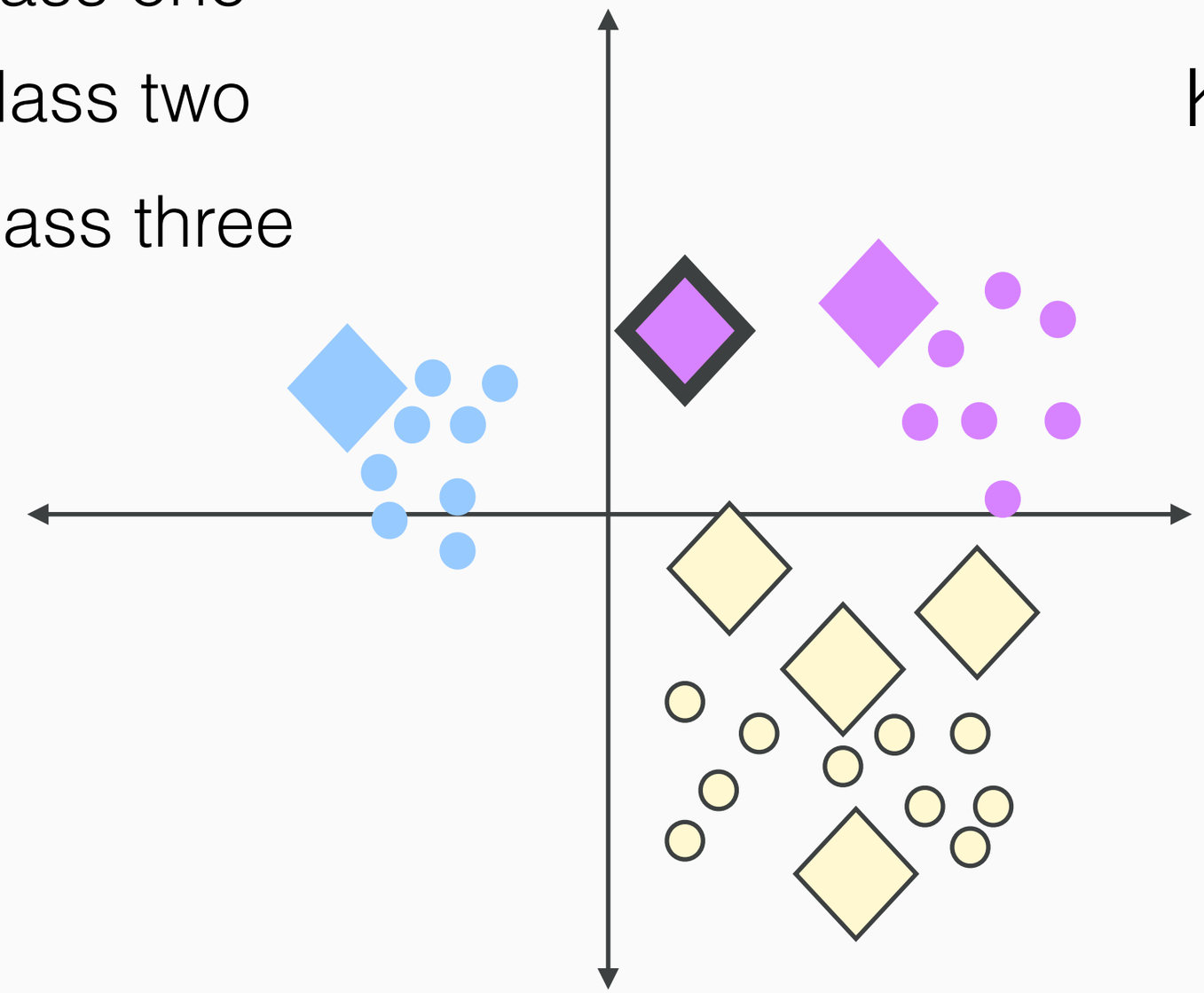


K Nearest Neighbor

= class one

= class two

= class three

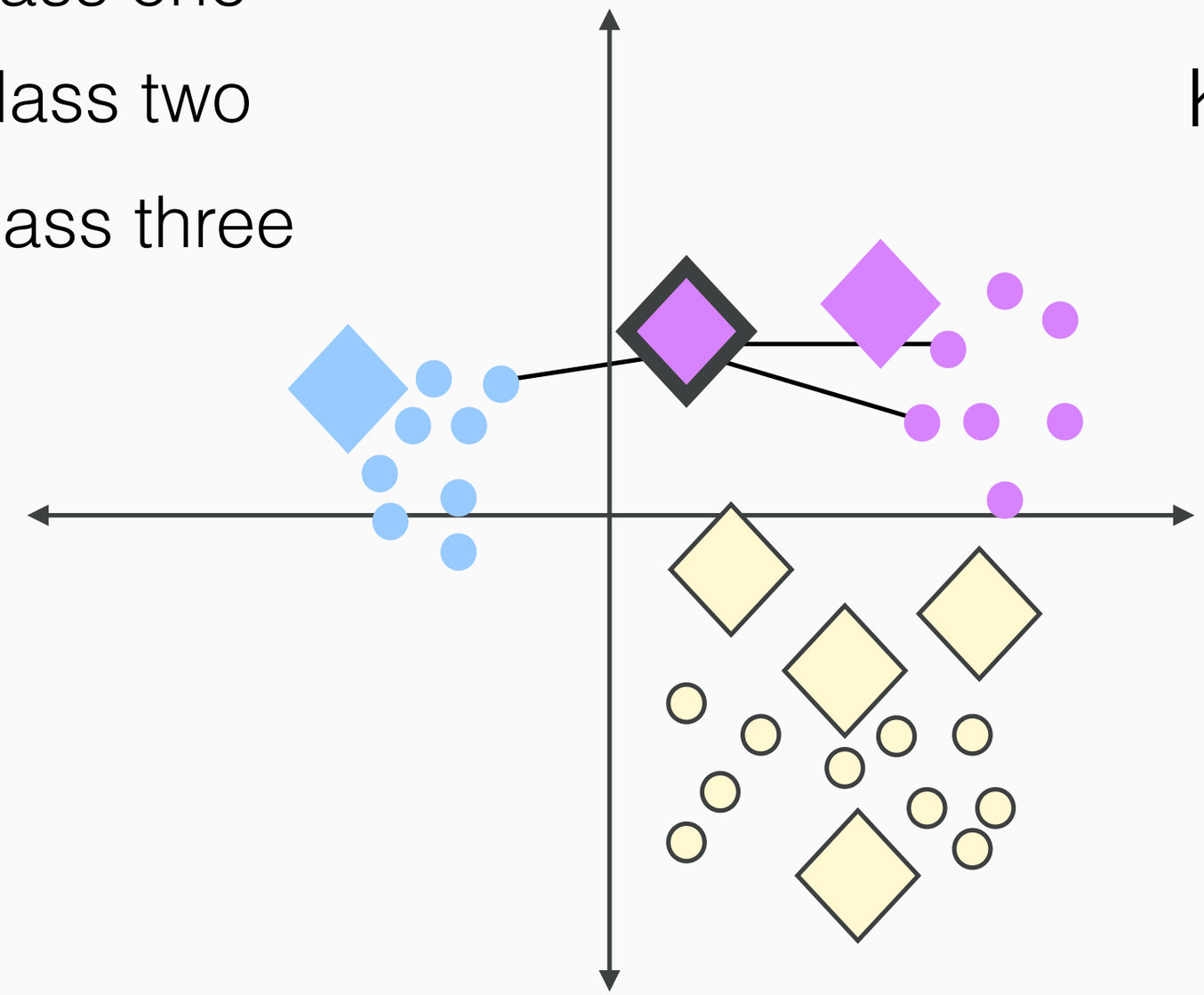


K Nearest Neighbor

= class one

= class two

= class three

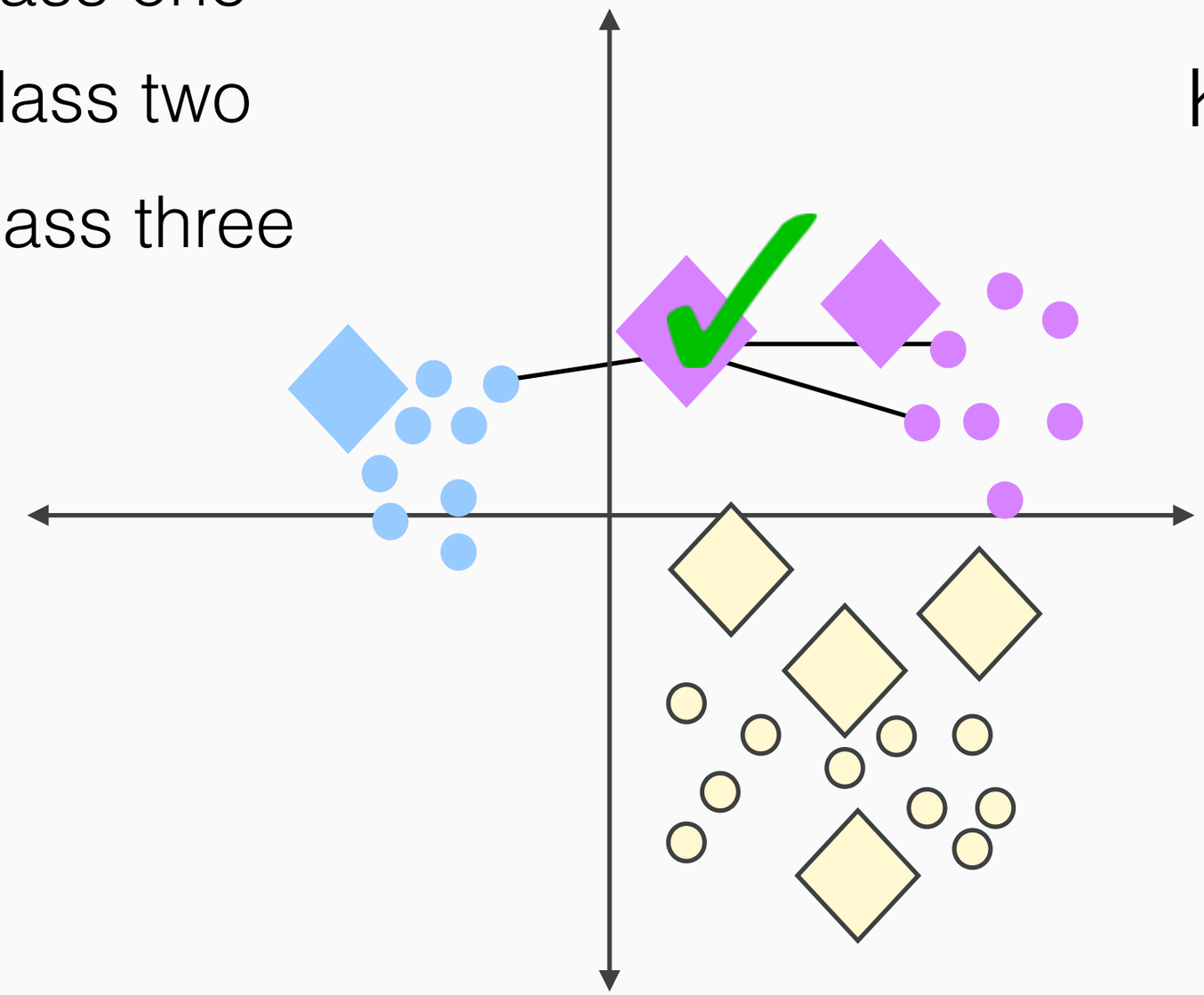


K Nearest Neighbor

= class one

= class two

= class three

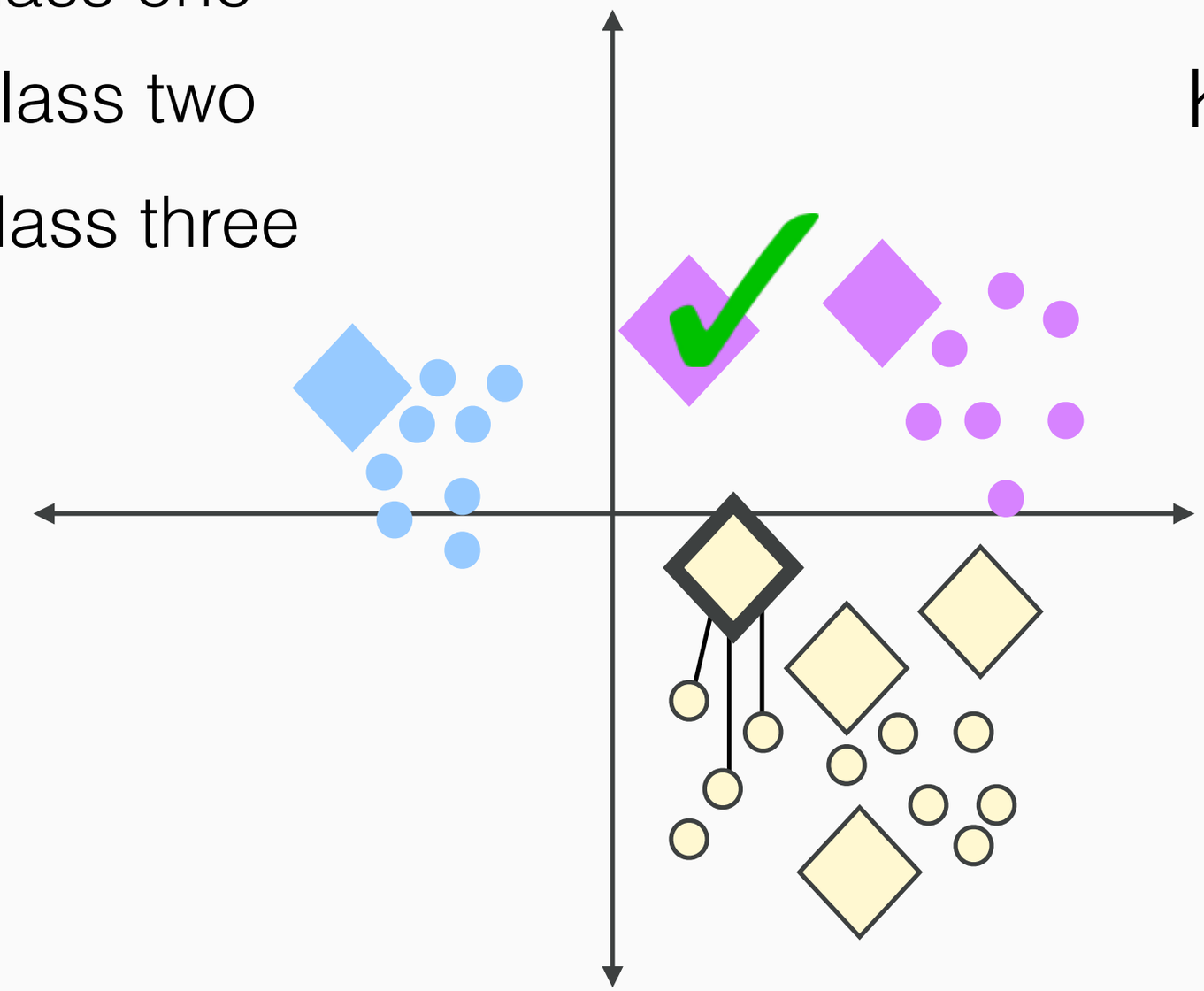


K Nearest Neighbor

= class one

= class two

= class three

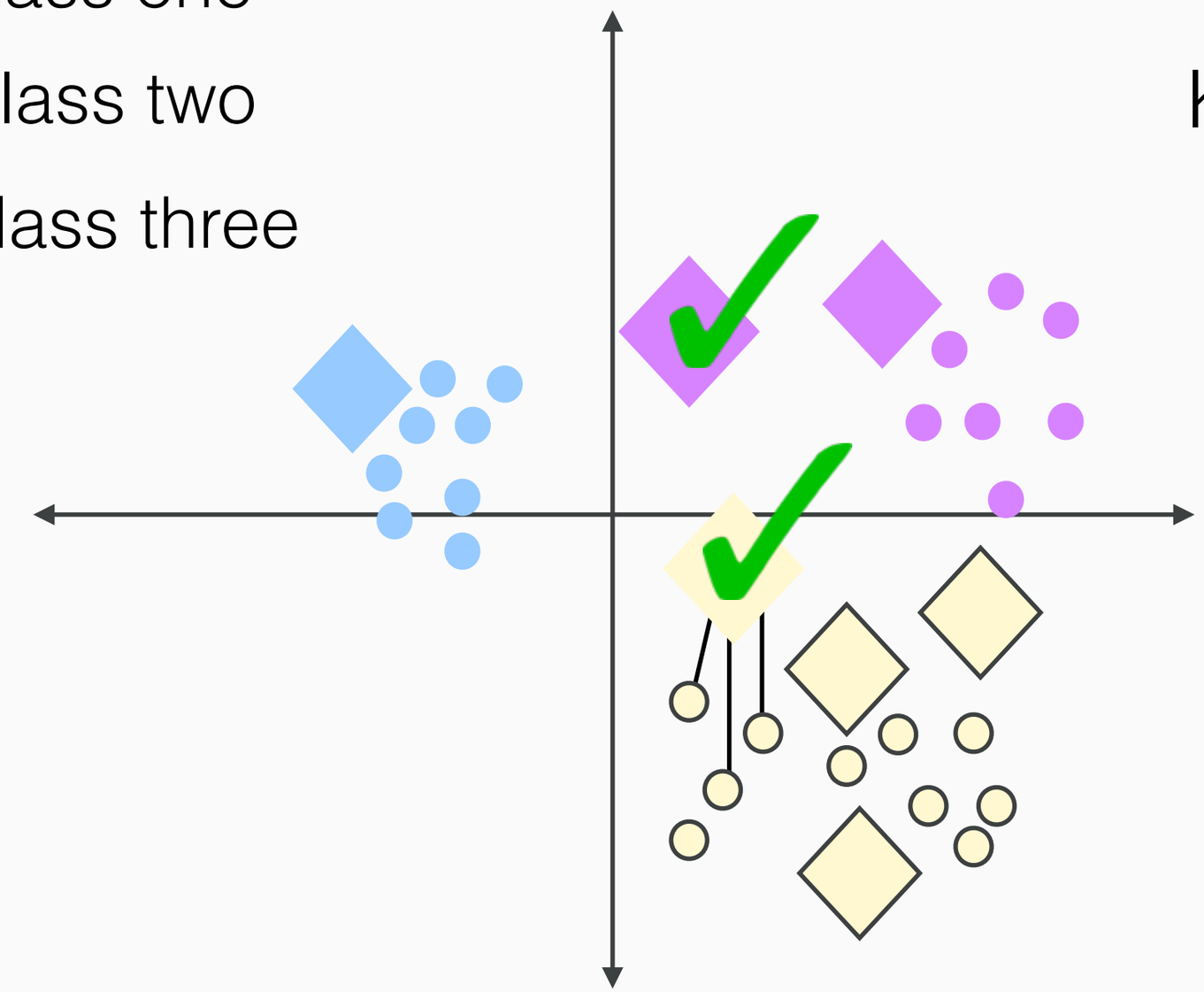


K Nearest Neighbor

= class one

= class two

= class three



K Nearest Neighbor

For both K-NN and Nearest Neighbor, we need a notion of *distance*

Reflection on Classification

- **Problem**: Come up with a classifier that can distinguish objects from different classes.
  - Algorithms: Memorize and Guess, NN, K-NN
- **Train**: think really hard, based on what objects you've seen so far, about how to make the classifier.
- **Test**: evaluate how well your classifier does.
- **Overfitting**: your way better on training than on testing (don't generalize well)
- **Features**: the relevant pieces of information an algorithm gets access to in order to learn (i.e. an image wouldn't be pixel values, it would be location of edges, presence of eyes, nose, fur, etc.).

One Final Philosophical Note…

The classification problem can be about *any* concept…

One Final Philosophical Note…

Q: What *is* language?

The classification problem can be about *any* concept…

Q: What *is* art?

Q: What *is* a sport?

Q: What *is* a chair?

One Final Philosophical Note…

The classification problem can be about *any* concept…

Q: What *is* language?

Q: What *is* a sport?

Q: What *is* art?

Q: What *is* a chair?

In some sense, our ability to define algorithms to learn these concepts paints a picture of how effective *we can be* at defining these concepts.

One Final Philosophical Note…

Q: What *is* a sport?

What would a classifier for this look like?

In some sense, our ability to define algorithms to learn these concepts paints a picture of how effective *we can be* at defining these concepts.

One Final Philosophical Note…

Q: What *is* a sport?

What would a classifier for this look like?

Can it be learned by pointing to positive and negative examples?

In some sense, our ability to define algorithms to learn these concepts paints a picture of how effective *we can be* at defining these concepts.

Clicker Question!

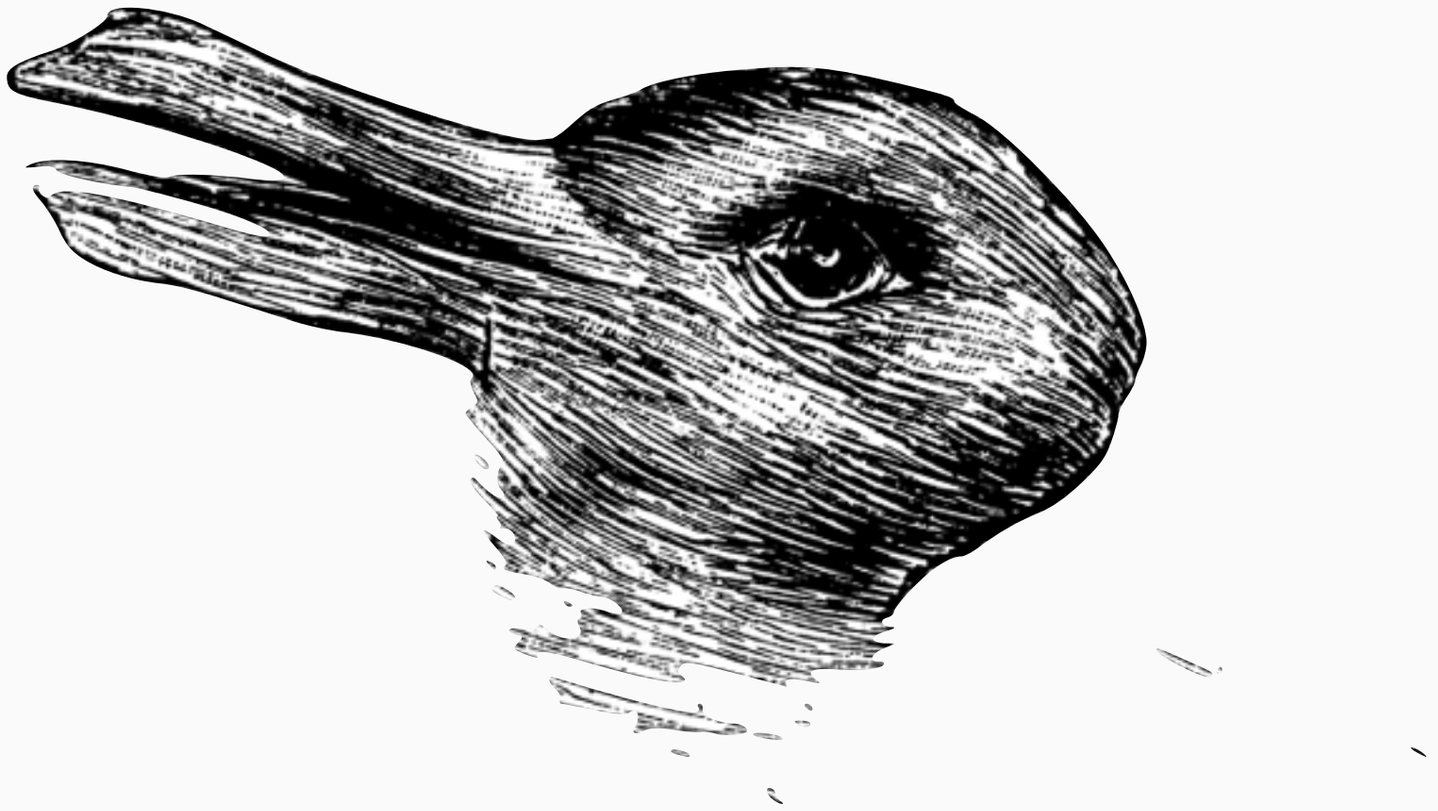
[A] That's a 1

[B] That's a 7

[C] That's a crowbar

Clicker Answer!

Welche Thiere gleichen ein-
ander am meiſten?



Kaninchen und Ente.

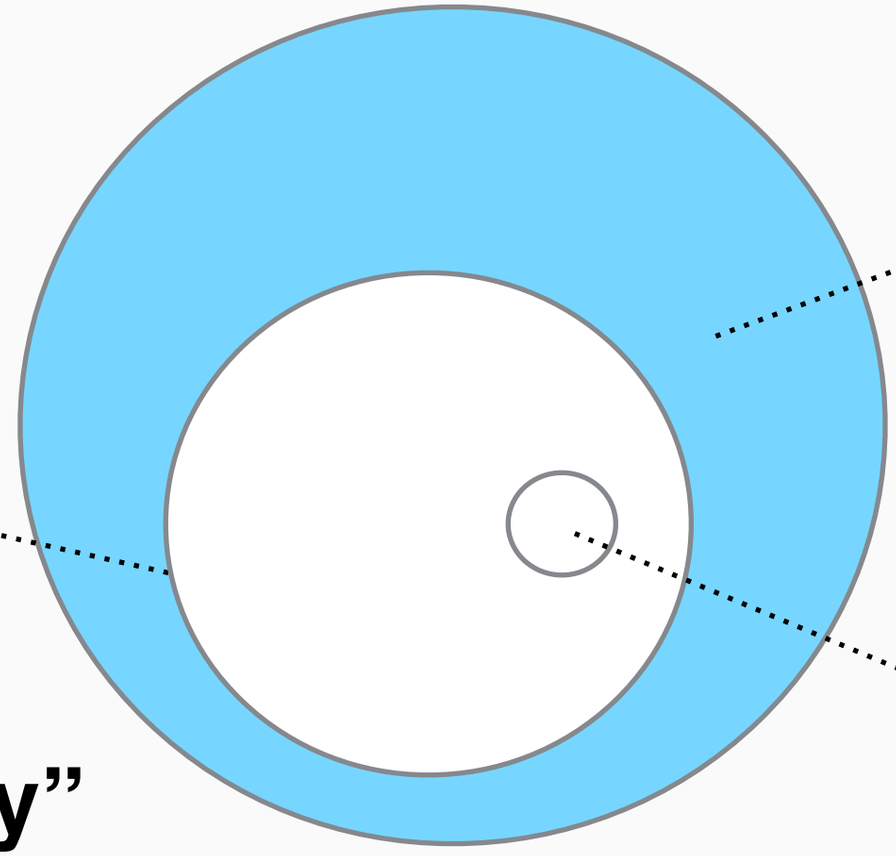
"Kaninchen und Ente" ("Rabbit and Duck") from the 23 October 1892 issue of Fliegende Blätter

One Final Philosophical Note…

Our concepts have meaning in so far as they can be learned (algorithmically?)

One Final Philosophical Note…

Our concepts have meaning in so far as they can be learned (algorithmically?)

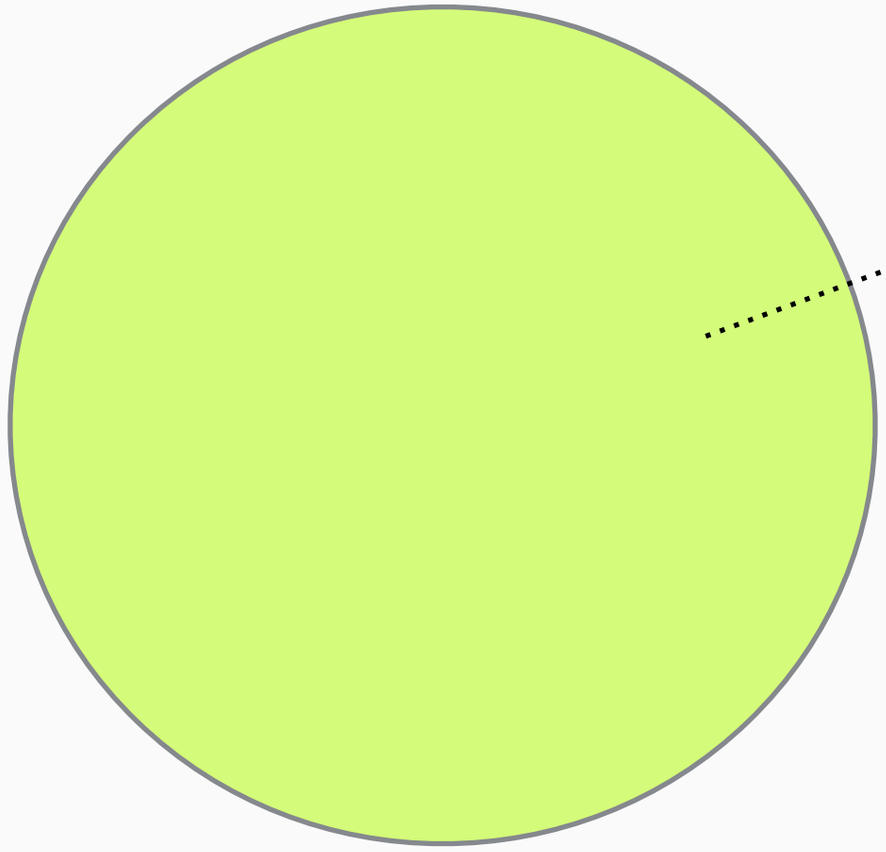


“Computability”

One Final Philosophical Note…

Our concepts have meaning in so far as they can be learned (algorithmically?)

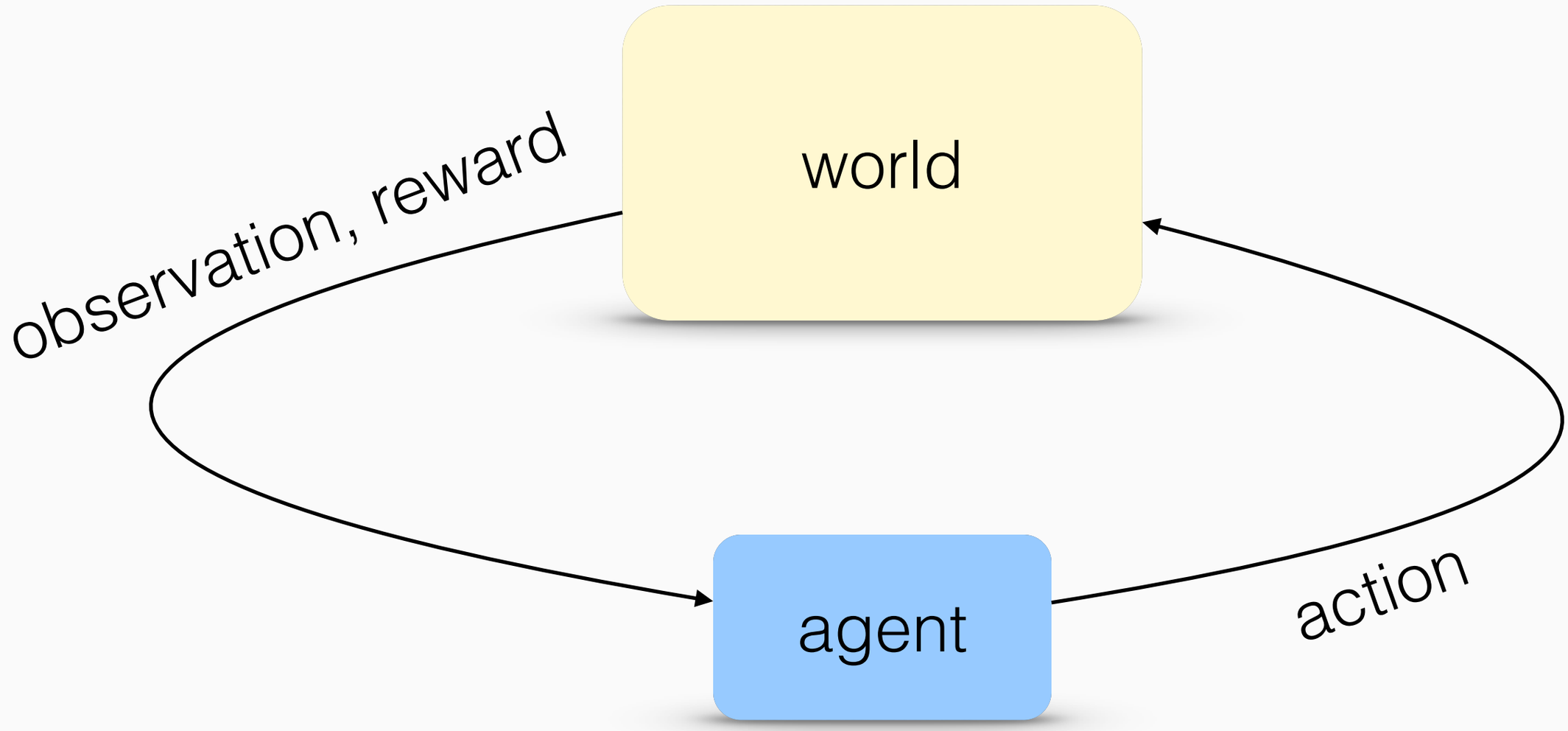
"A Theory of the Learnable"
Les Valiant, 1984



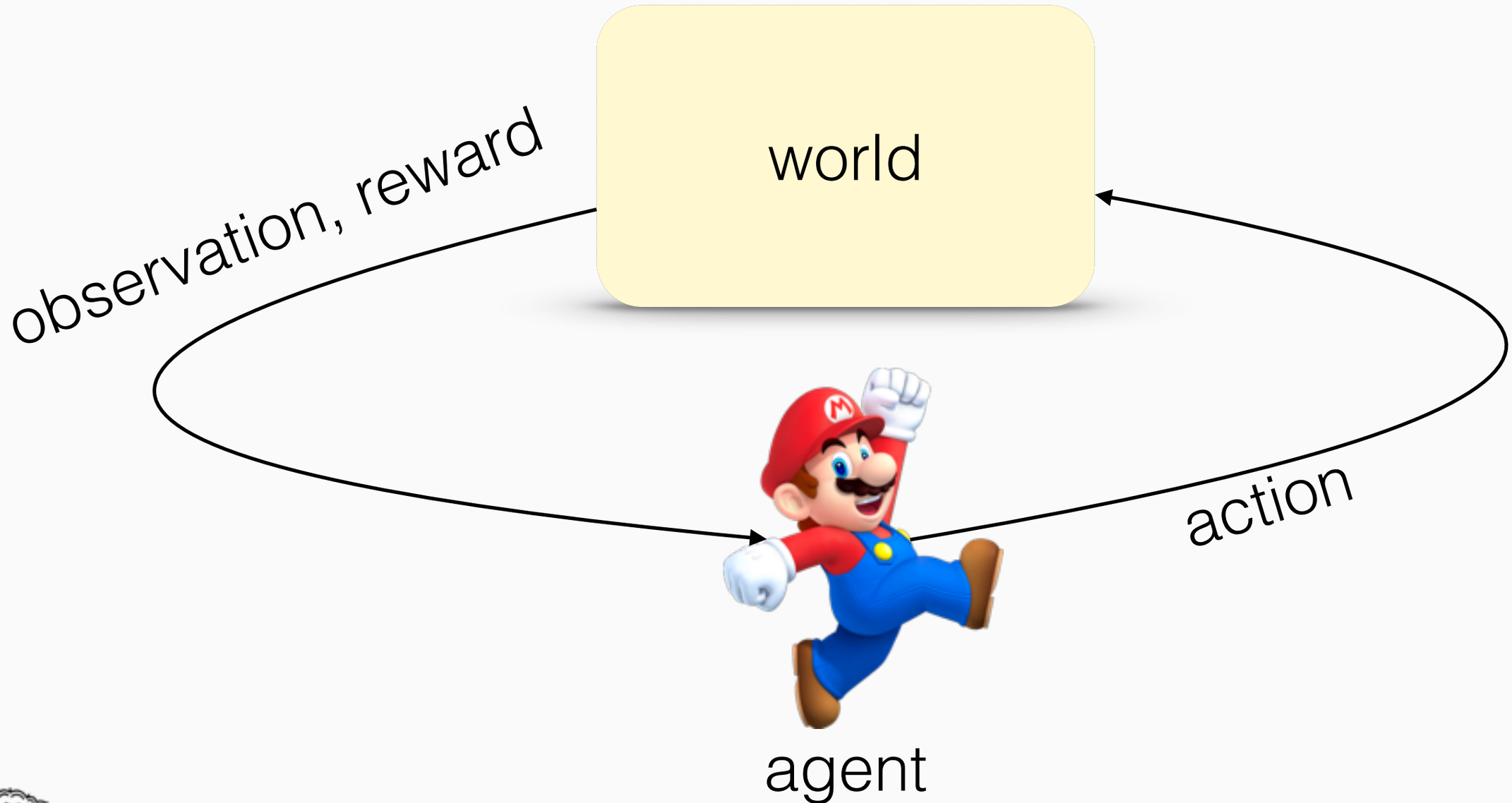
Reinforcement Learning

Demo time!

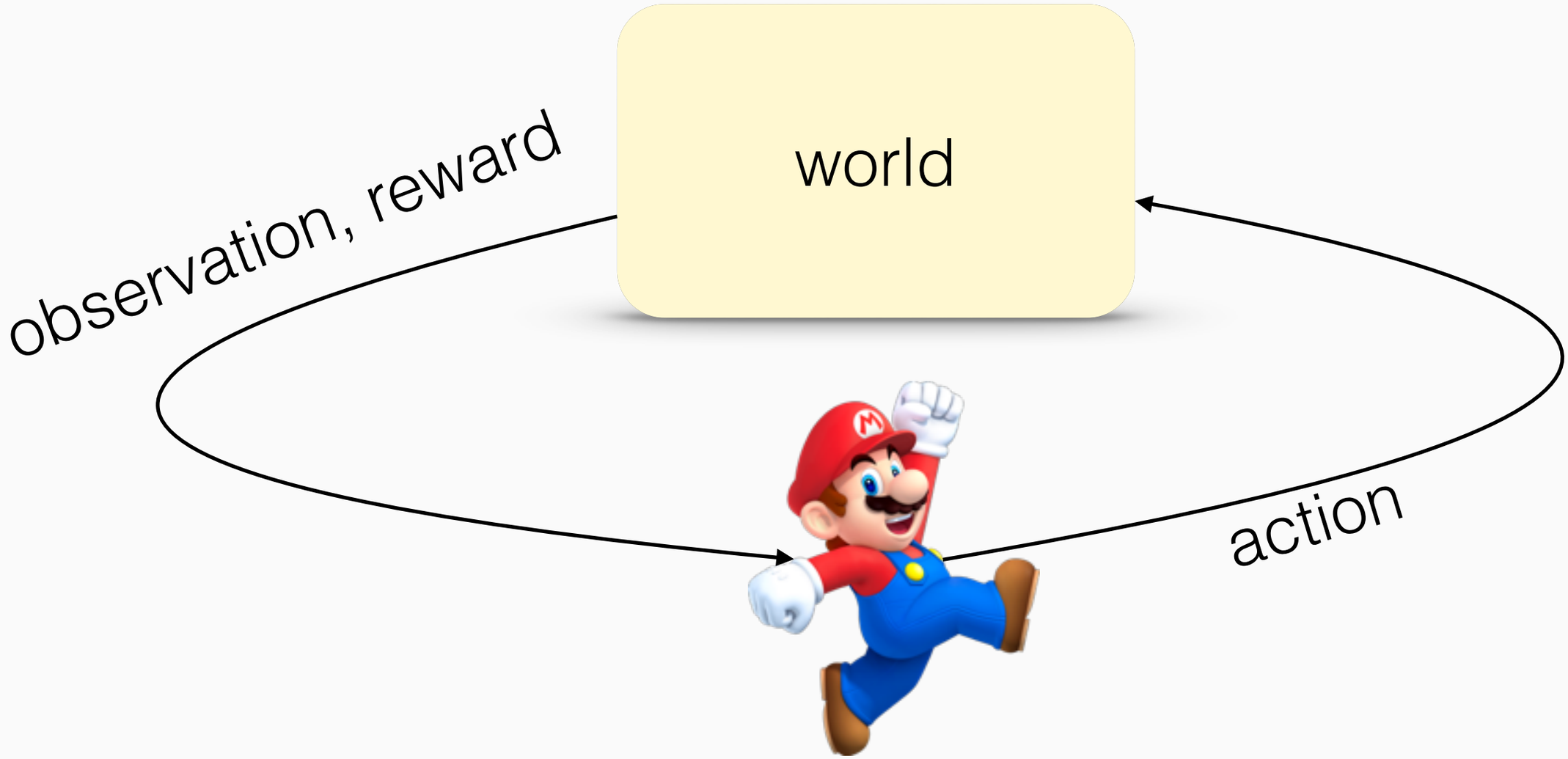
Reinforcement Learning



Reinforcement Learning

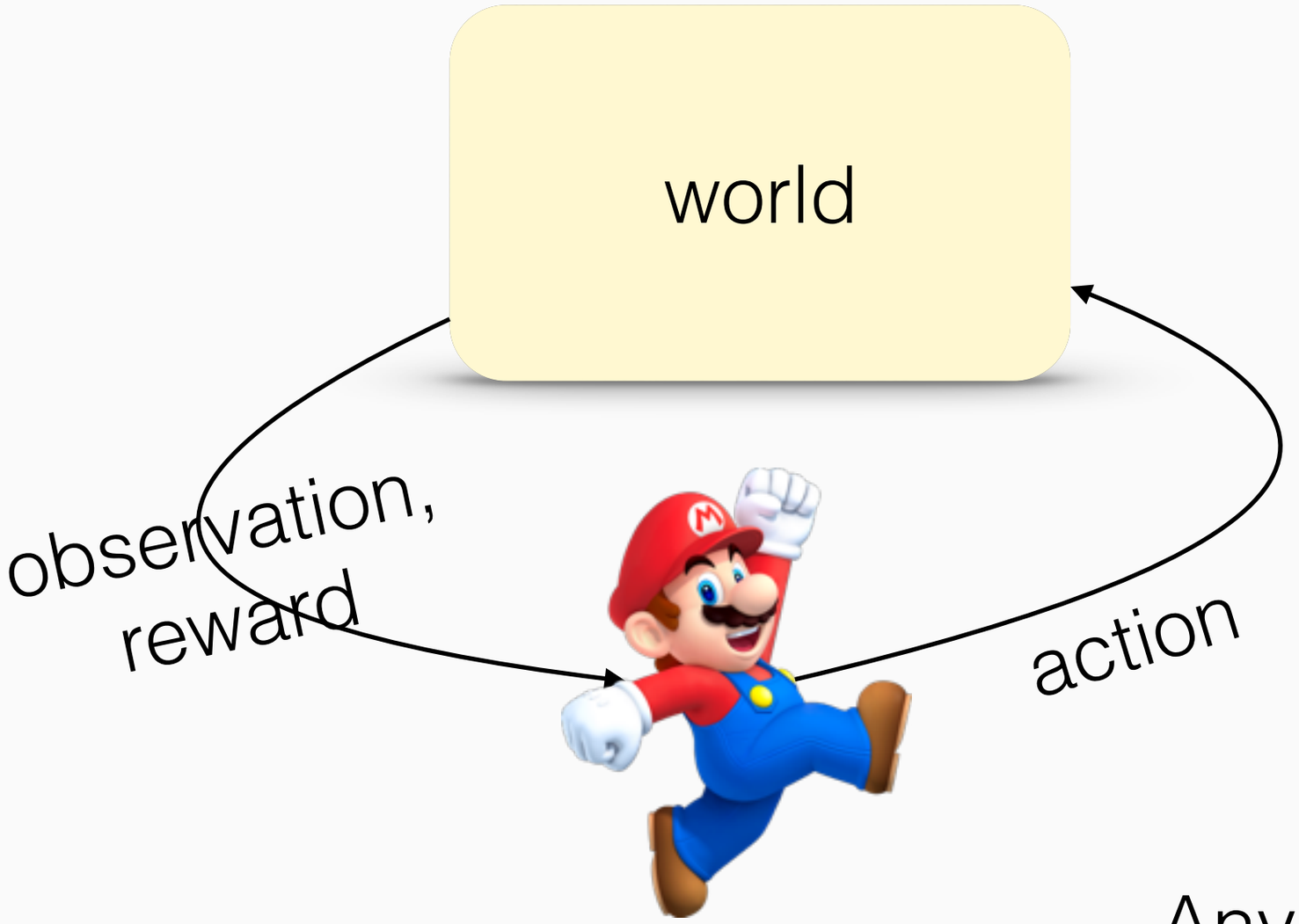


Reinforcement Learning



Goal: Maximize reward!

Reinforcement Learning



Any problem with a goal…

Goal: Maximize reward!