Unit 5: AI & Machine Learning

Dave Abel

February 29th, 2016

Field Trip!

Field Trip!

- Three scheduled times in the Yurt! Limited space :(
  - Wednesday 3/9 from 2pm-3pm
  - Wednesday 3/16 3pm-4pm
  - Wednesday 3/23 1pm-2pm
- If you want to go:
  - Send me an email (david_abel@brown.edu) with subject "CS8 Yurt"
  - In the message, list your date/time preferences from 1-3, 1 being top preference, 3 being bottom preference (only list those that you can actually make).

Machine Learning Takeaway

- **Learning can be represented as an algorithm!**
- Several kinds of learning, each requires different teaching/training styles.
    - Classification! (Today, Wednesday)
    - Reinforcement! (Wednesday, Friday)

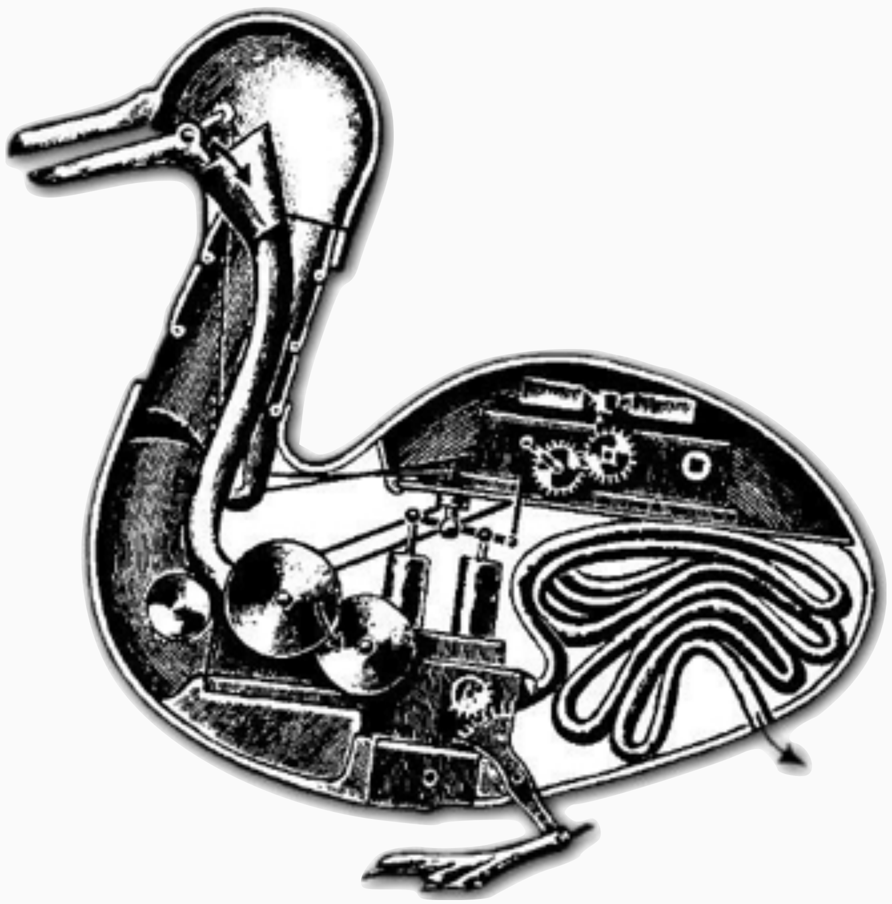
Outline

- Overview of AI
- Some examples of recent success.
- Classification
  - Setup, features
  - Memorize and Guess
  - Nearest Neighbor
- Training Process
- Testing Process
- Overfitting, Occam's Razor

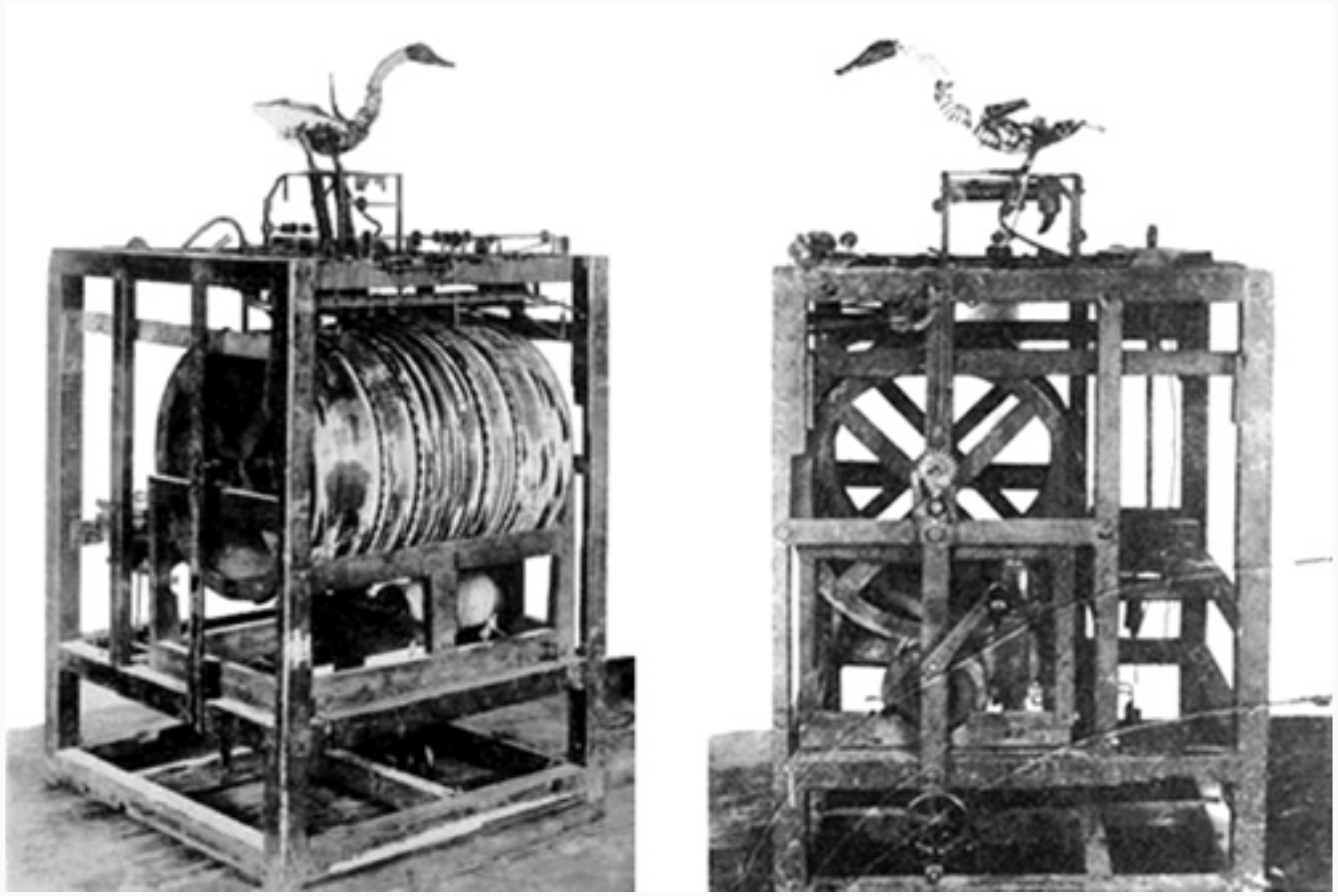
Overview of AI

http://www.atlasobscura.com/places/canard-digerateur-de-vaucanson-vaucansons-digesting-duck

Overview of AI

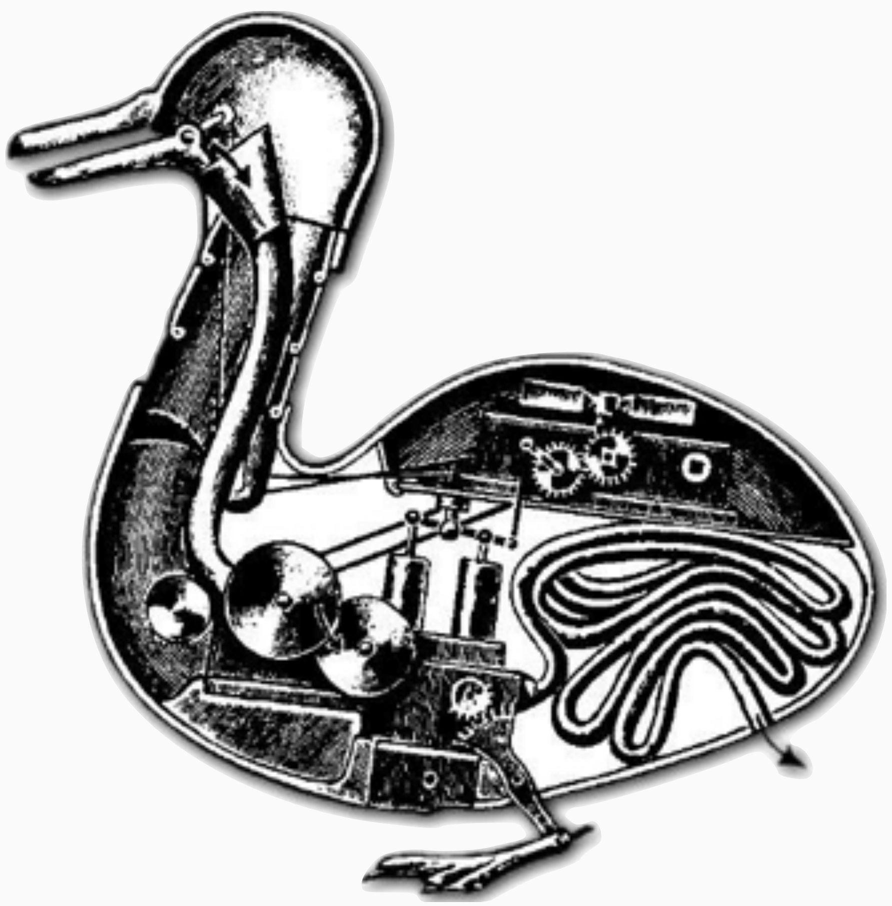


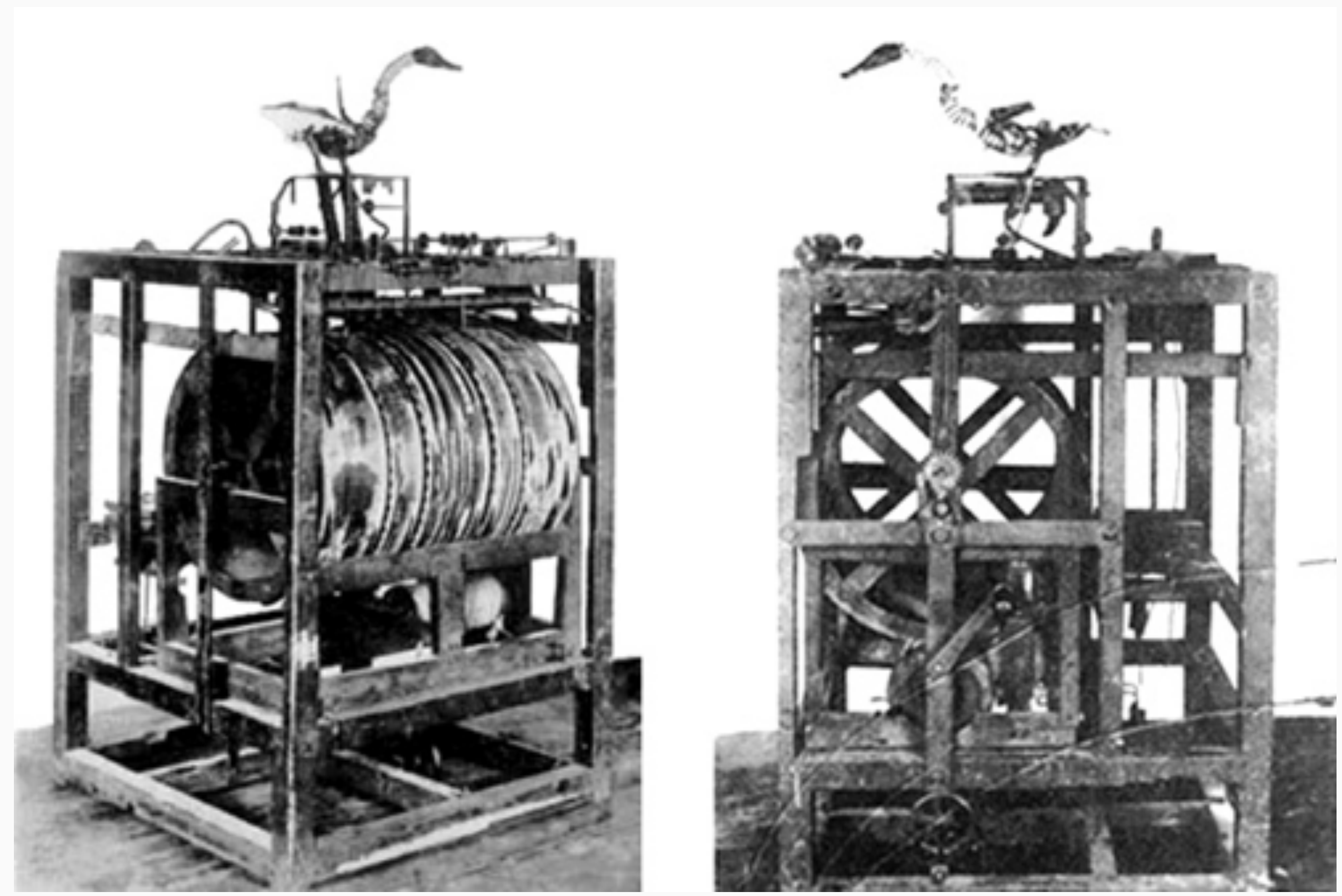
http://www.atlasobscura.com/places/canard-digerateur-de-vaucanson-vaucansons-digesting-duck



1739

Overview of AI





"without...the duck of Vaucanson, you would have nothing to remind you of the glory of France."
- Voltaire

http://www.atlasobscura.com/places/canard-digerateur-de-vaucanson-vaucansons-digesting-duck

Overview of AI

http://dartmouth.edu/sites/default/files/styles/header_image/public/2009-1035500133.jpg?itok=LlpoUNH9

Dartmouth Summer 1956

“We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.”
- John McCarthy, 1955

McCarthy, John; Minsky, Marvin; Rochester, Nathan; Shannon, Claude (1955), A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence retrieved 10:47 (UTC), 9th of April 2006

Dartmouth Summer 1956

- Automatic Computers
- How Can a Computer be Programmed to Use a Language
- Neuron Nets
- Theory of the Size of a Calculation
- Self-Improvement
- Abstractions
- Randomness and Creativity

Dartmouth Summer 1956

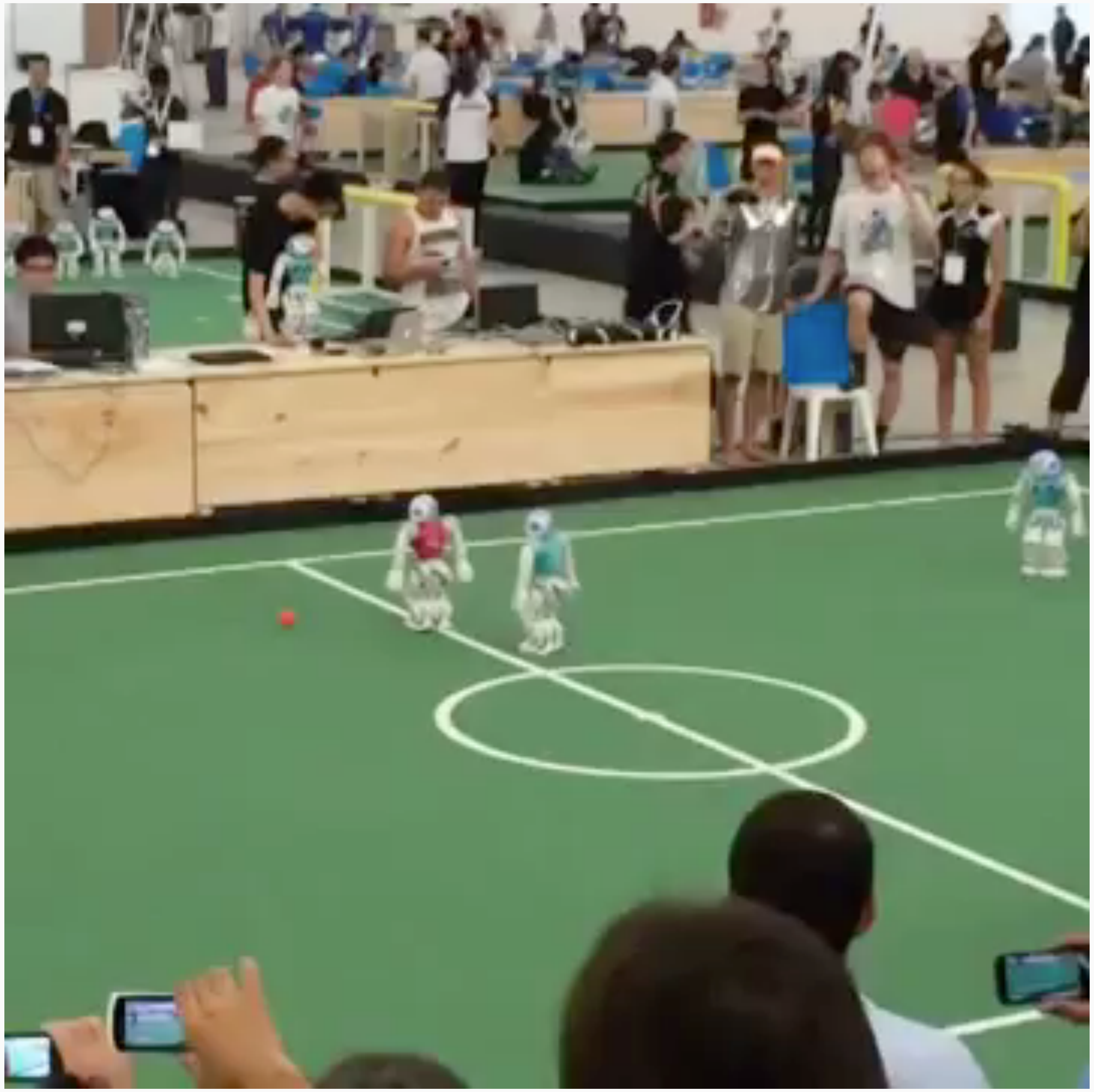
- Automatic Computers
- How Can a Computer be Programmed to Use a Language
- **Neuron Nets** ✓
- **Theory of the Size of a Calculation** ✓
- Self-Improvement
- Abstractions
- Randomness and Creativity

(60 years later…)

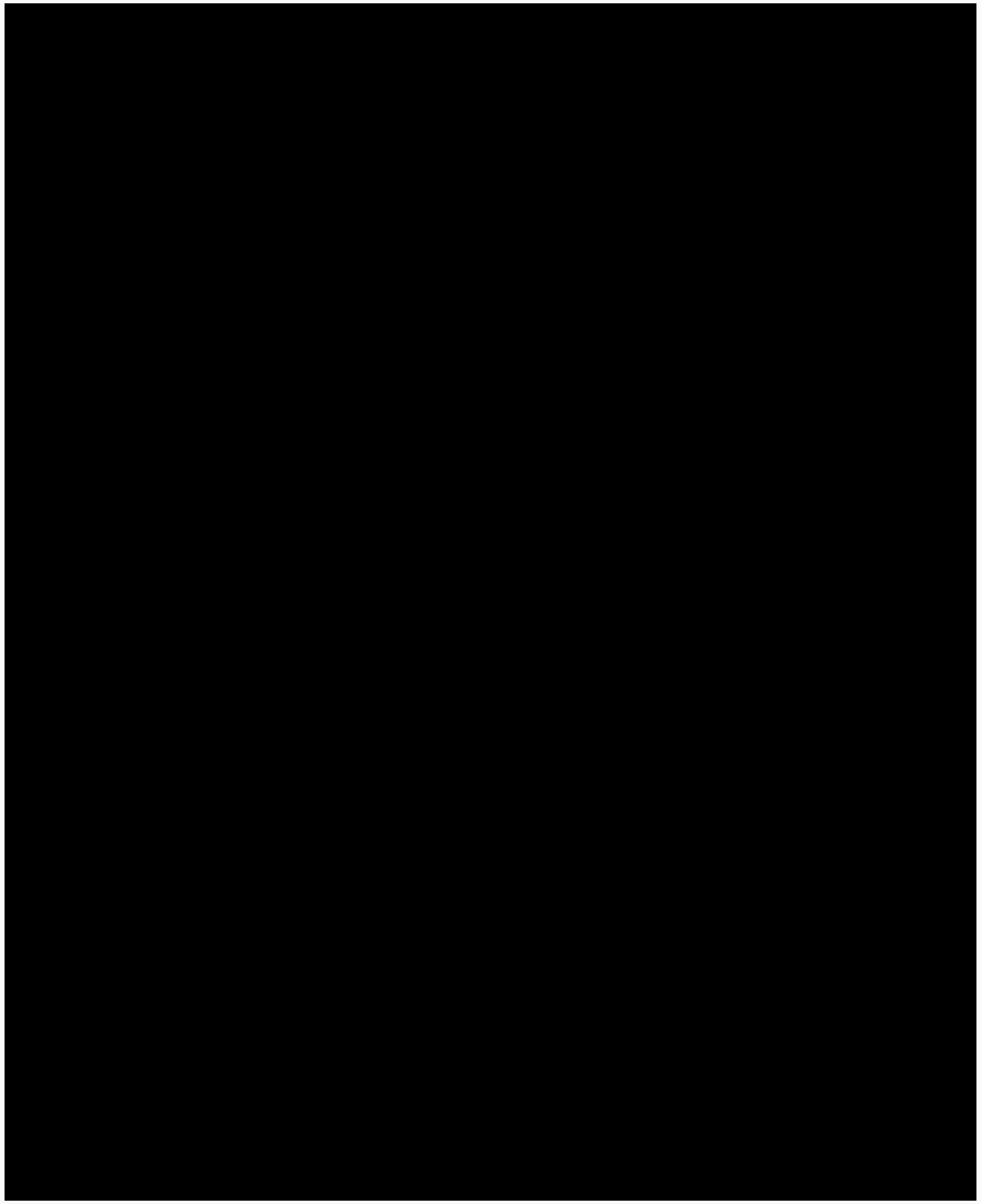
Some Recent Success

Some Recent Success

Some Recent Success



Some Recent Success



The Field of AI

- "The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it."

The Field of AI

- Learning
- Perception
- Language
- Reasoning, Planning
- Motion and Manipulation
- Knowledge Representation

- "The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it."

The Field of AI

Unit 5

- **Learning**
- Perception
- Language
- Reasoning, Planning
- Motion and Manipulation
- Knowledge Representation

- "The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it."

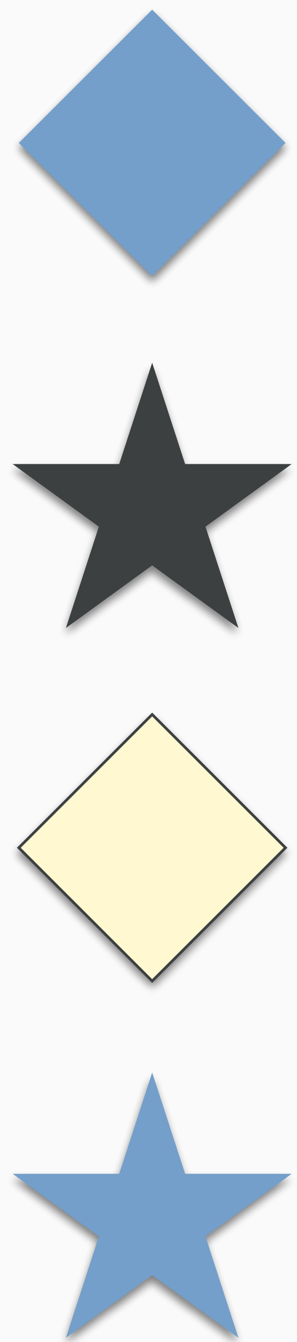
The Field of AI

- Learning
- **Perception**
- **Language**
- Reasoning, Planning
- Motion and Manipulation
- Knowledge Representation

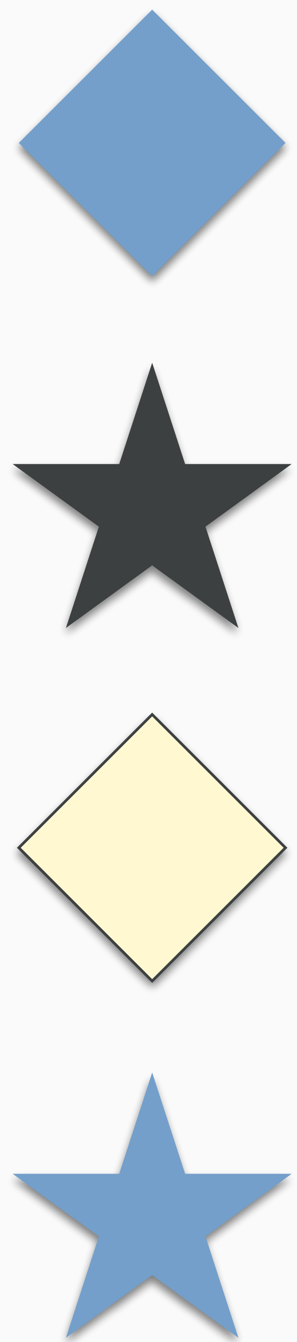
Unit 6

- "The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it."

Machine Learning: Classification

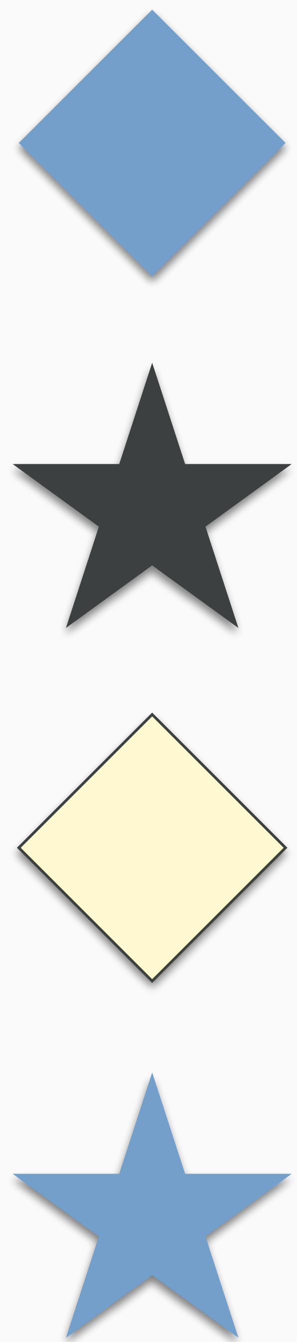


Machine Learning: Classification



Snozzberry

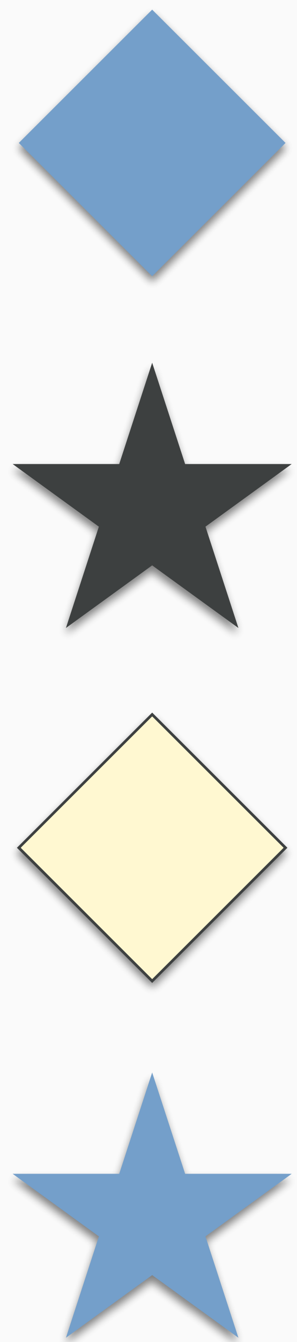
Machine Learning: Classification



Snozzberry

Buzzberry

Machine Learning: Classification

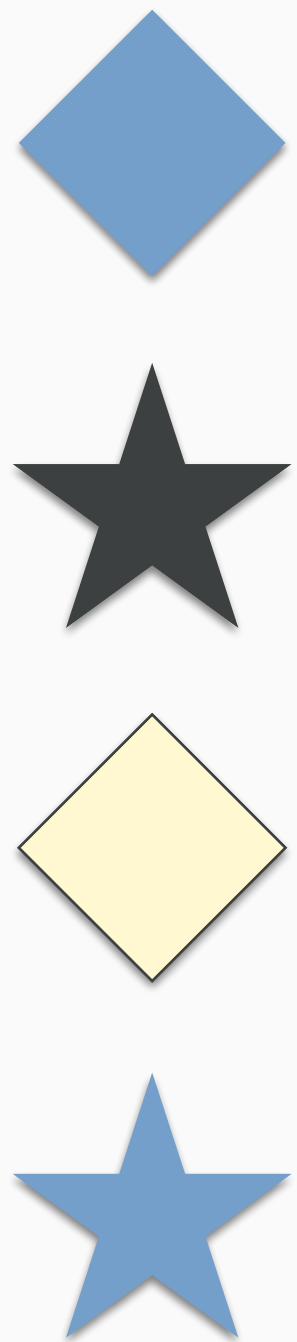


Snozzberry

Buzzberry

Fizzberry

Machine Learning: Classification



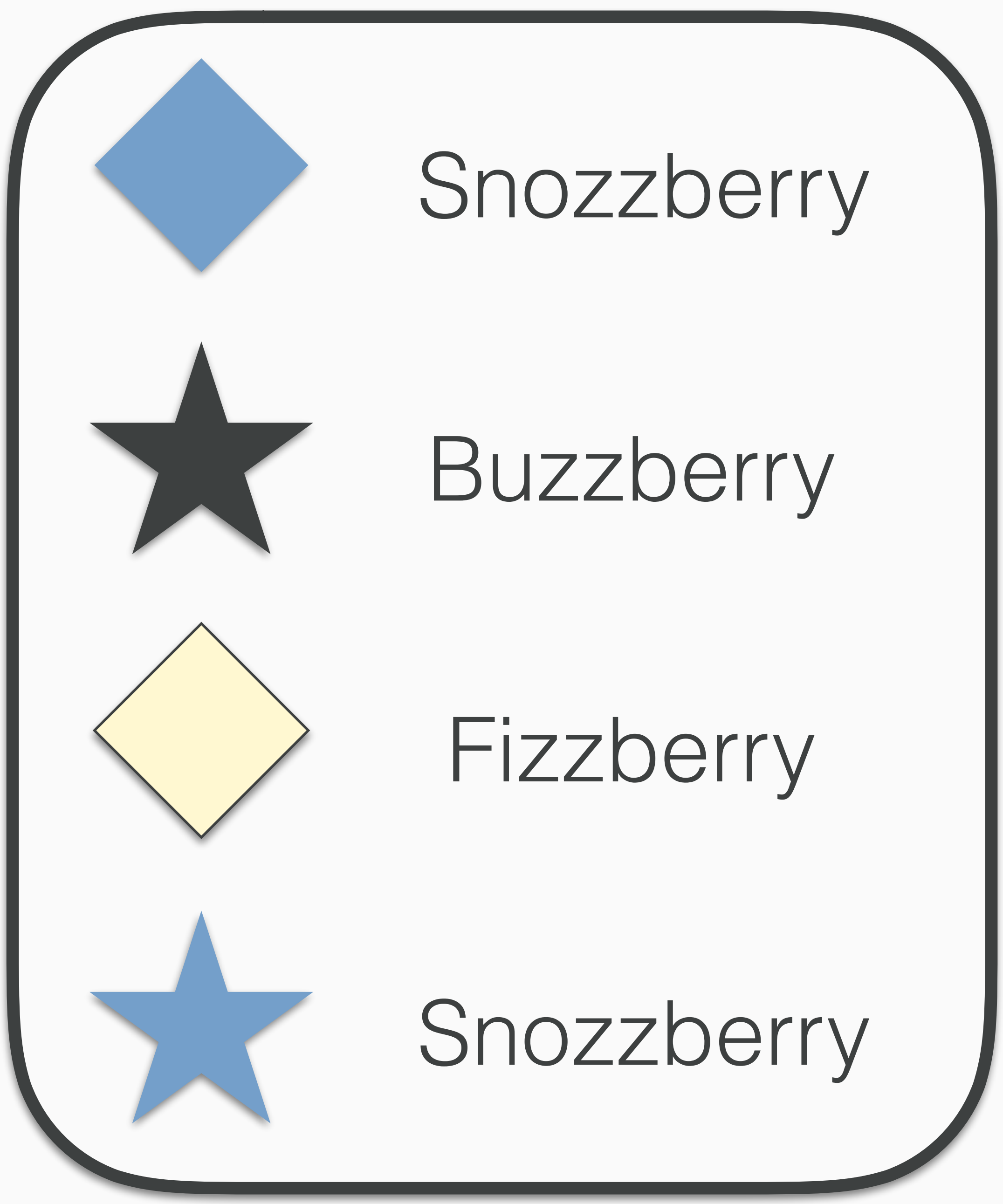
Snozzberry

Buzzberry

Fizzberry

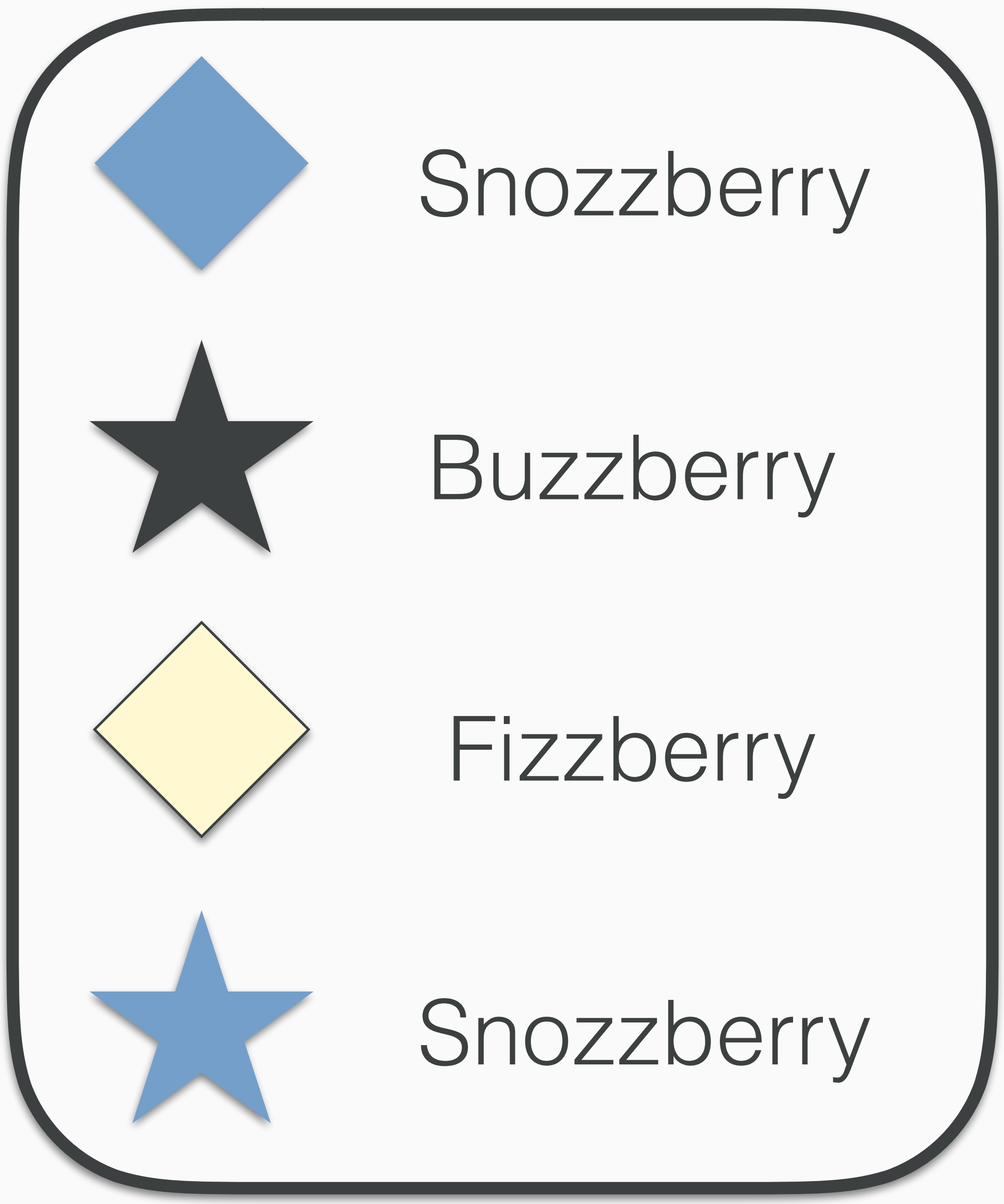
Snozzberry

Machine Learning: Classification



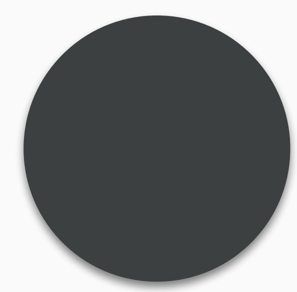
Training!

Machine Learning: Classification

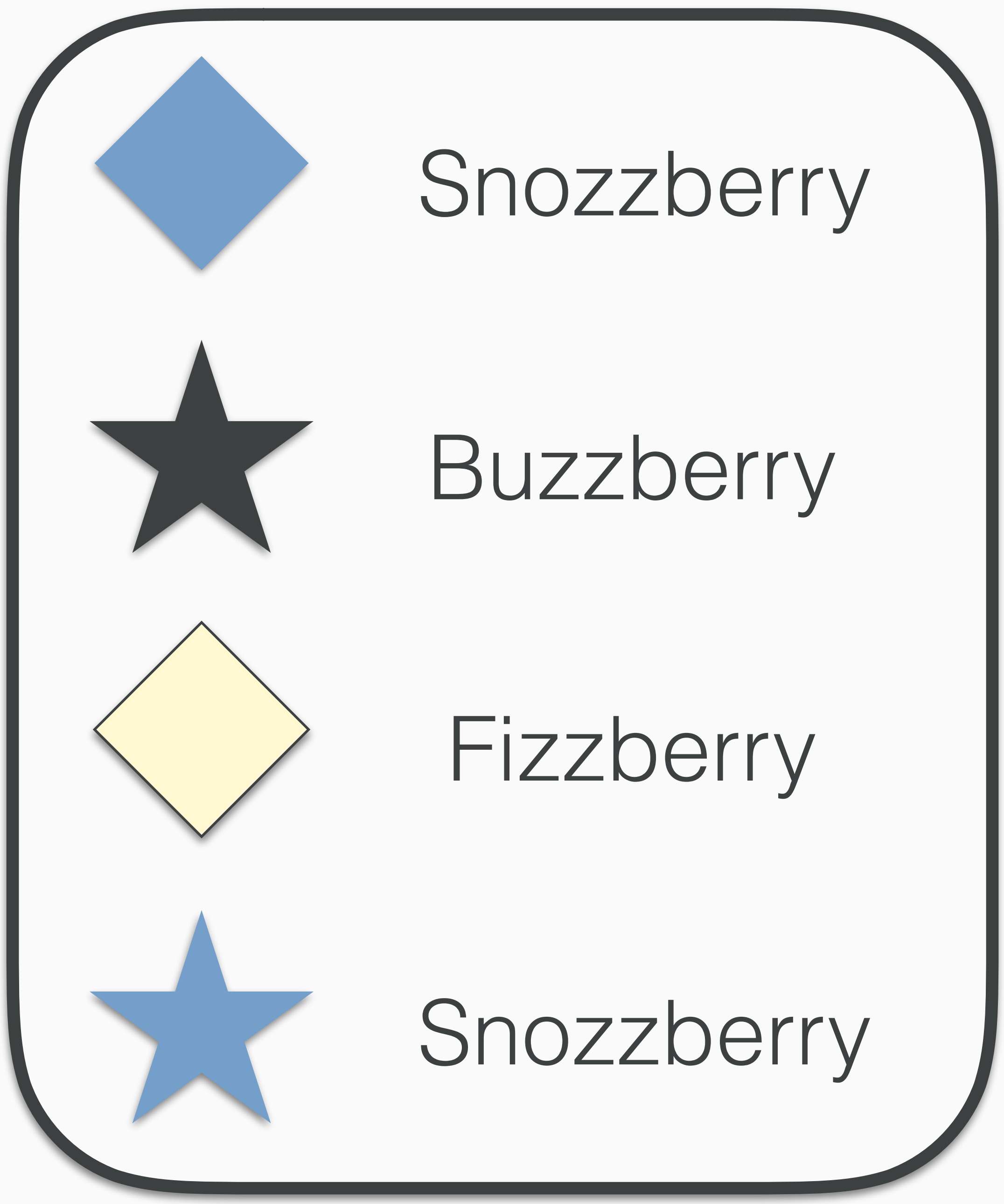


Training!

Q: What should this be called?

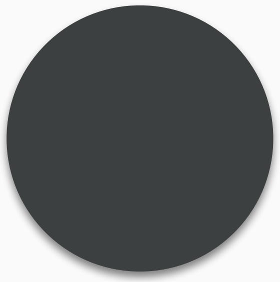


Machine Learning: Classification



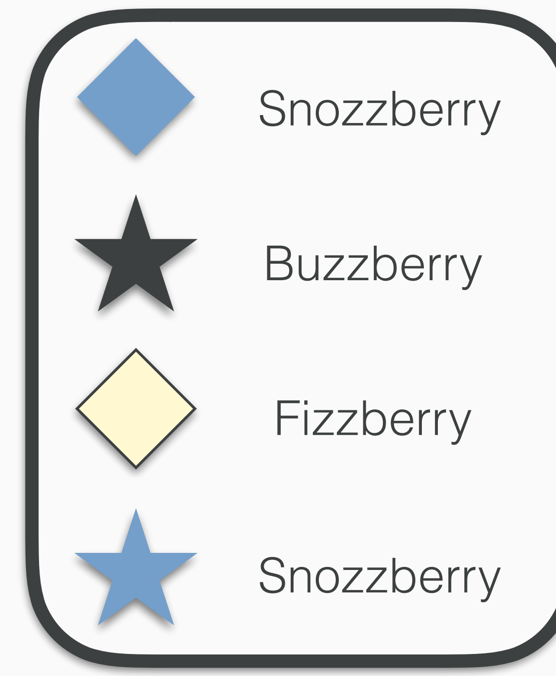
Training!

Q: What should this be called?



A: Buzzberry!

Problem: Classification

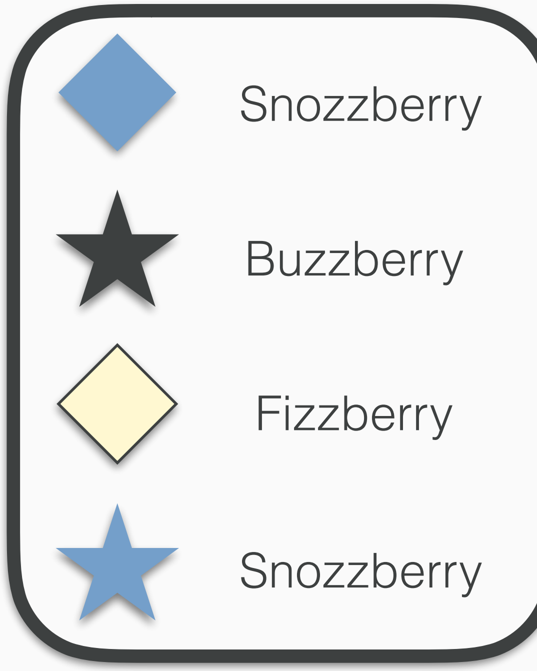


Training!

- INPUT: A bunch of labeled training data, a discrete set of possible labels.
- OUTPUT: A classifier! (that best classifies all objects in the space).

Problem: Classification

- INPUT: A bunch of labeled training data, a discrete set of possible labels.



Training!

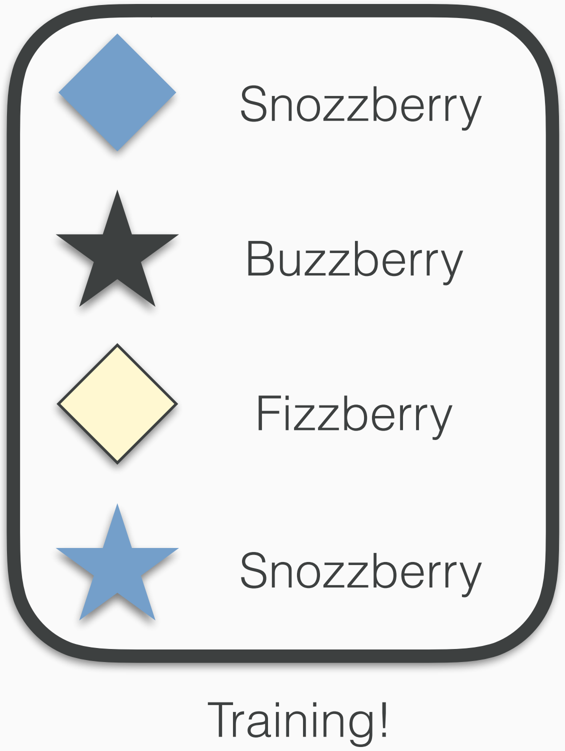
- OUTPUT: A classifier! (that best classifies all objects in the space).

A Classifier

- Takes an object from the space of interest (pieces of fruit, shapes, images, words, etc.).
- Reports a *label*, from the discrete set of possible labels.
- Could do this according to a simple rule! (If blue, snozzberry!)
- Could be a complicated logical rule.
- **In general: Could be any (Scratch) program!**

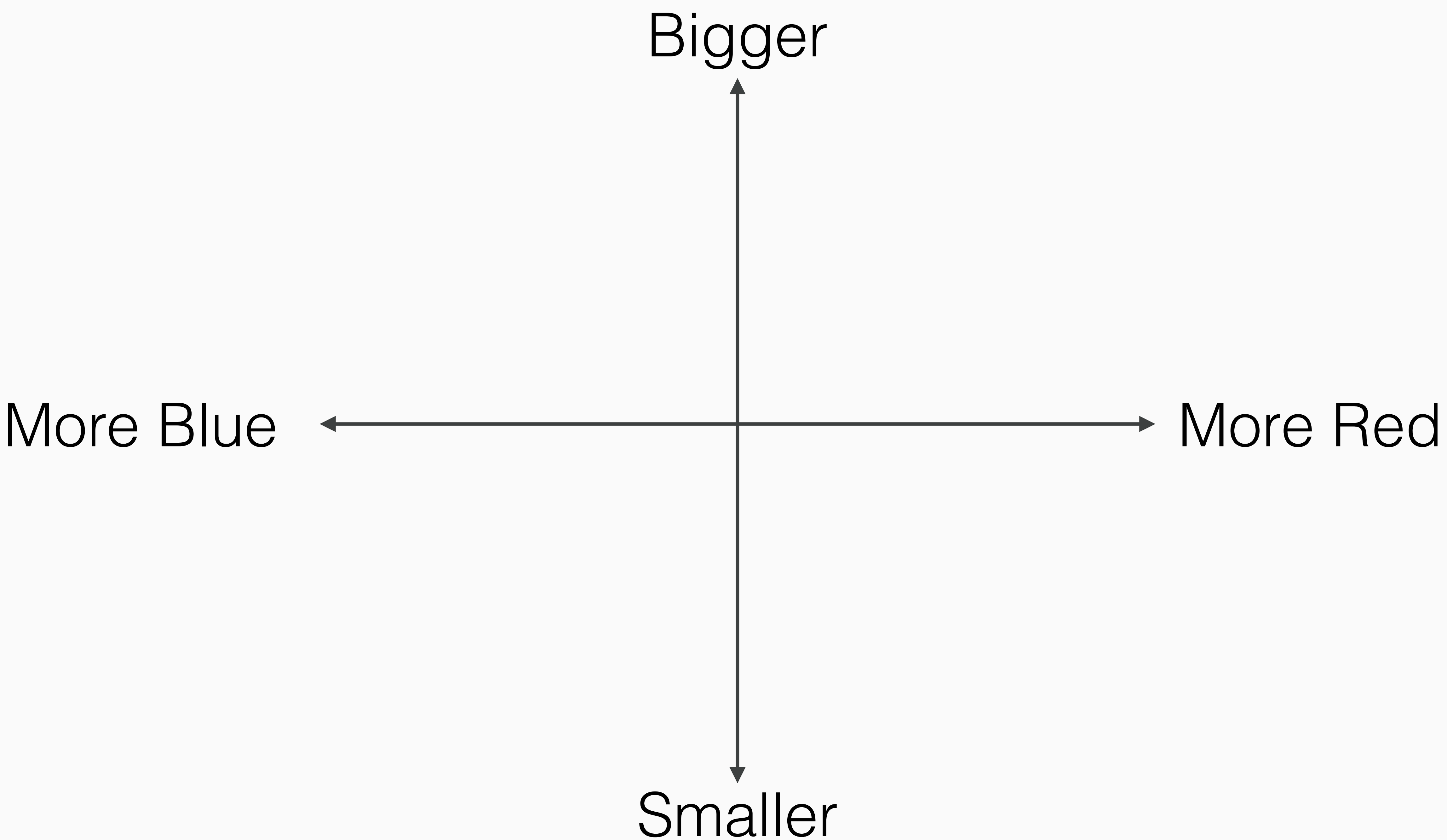
Problem: Classification

- INPUT: A bunch of labeled training data, a discrete set of possible labels.

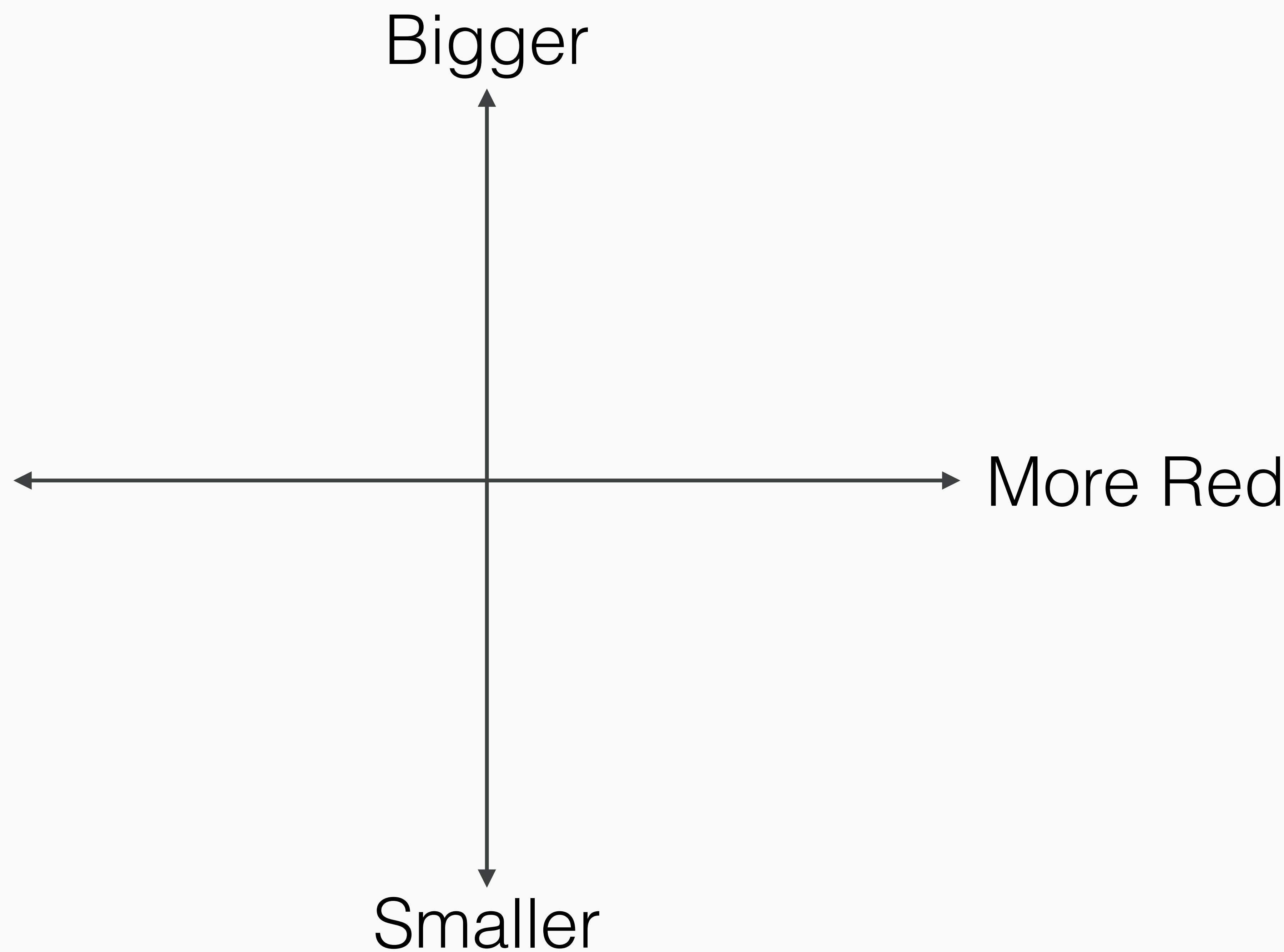


- OUTPUT: A classifier! (that best classifies all objects in the space).

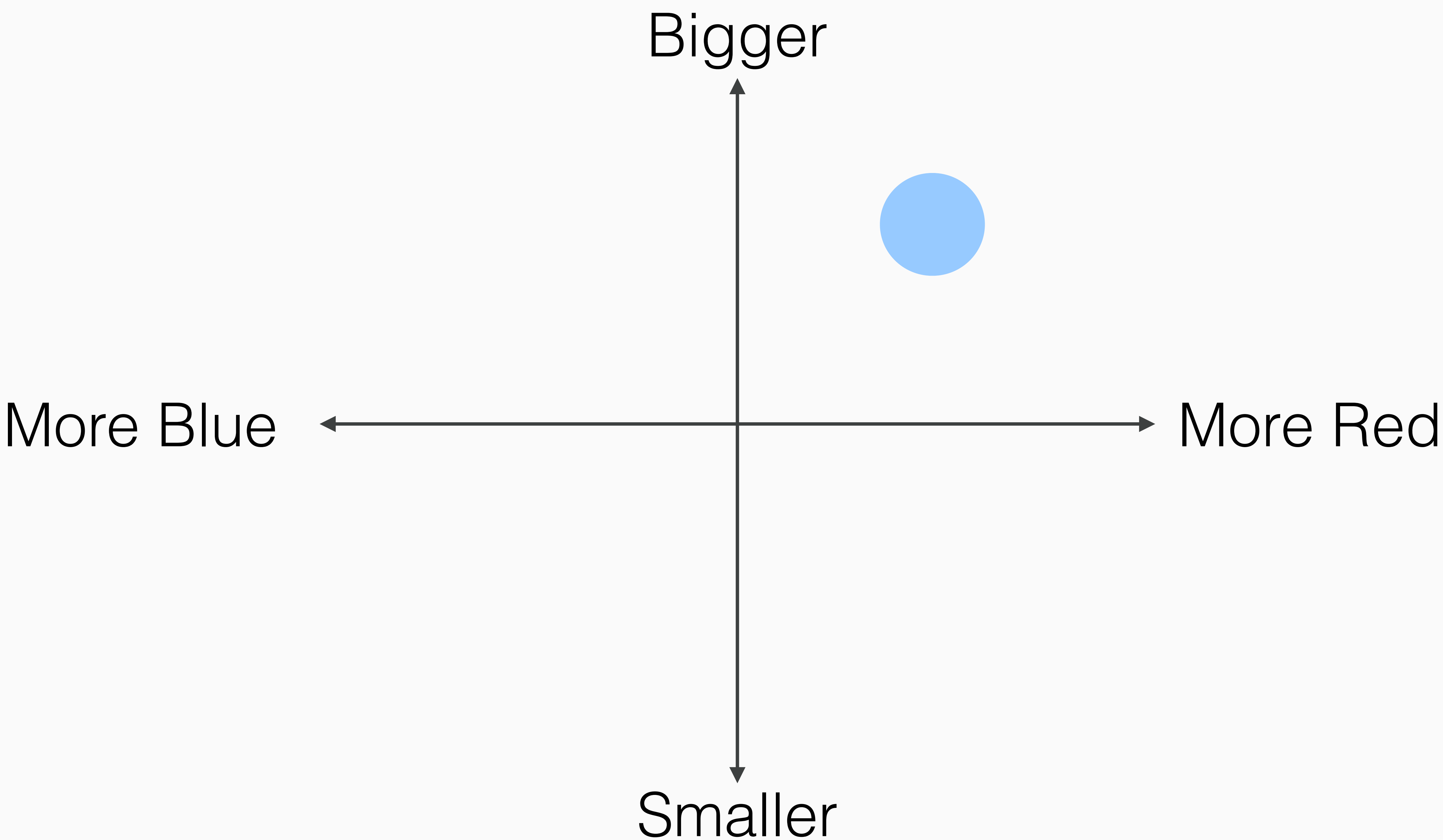
A Classifier: Fruit Example



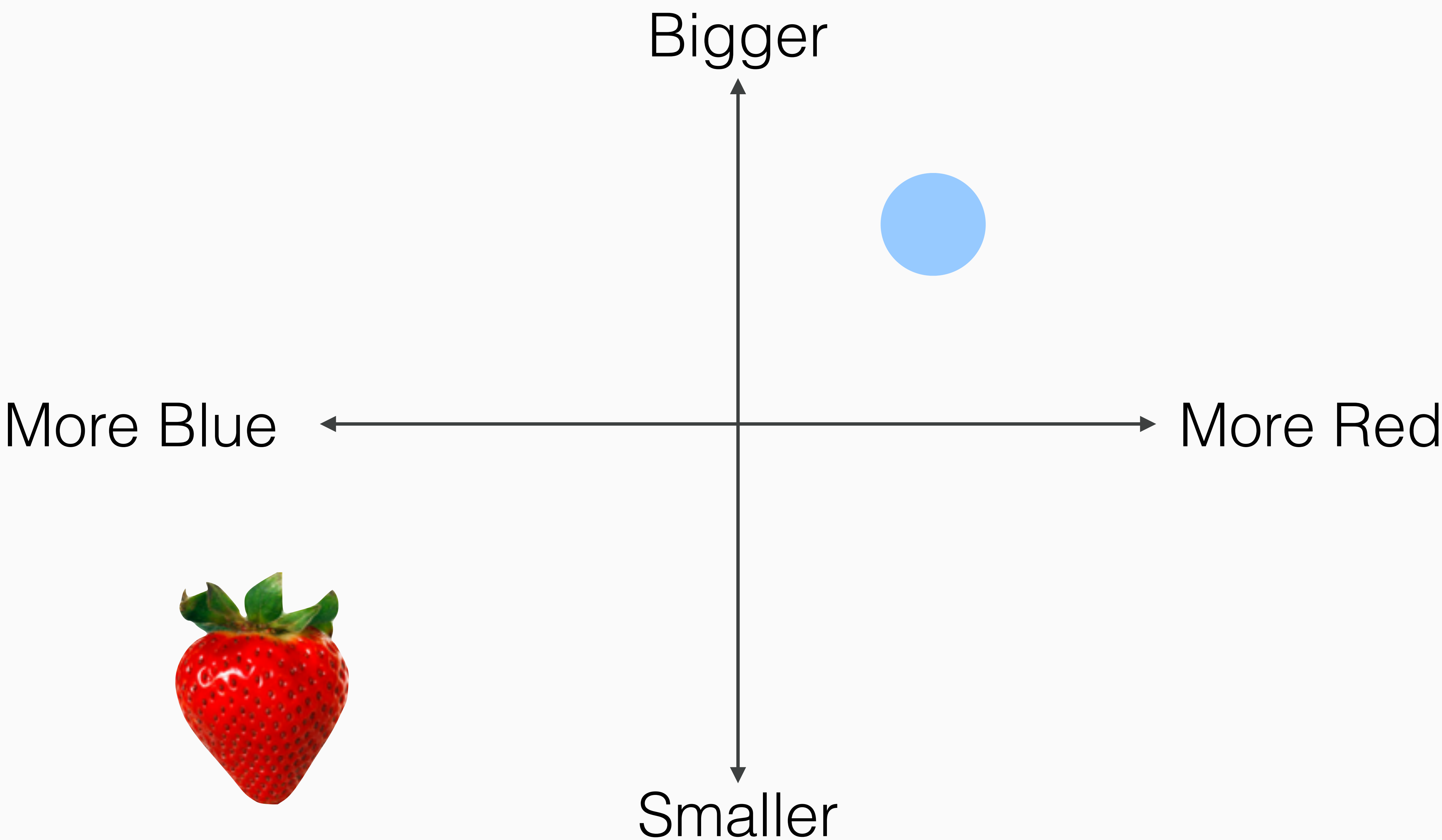
A Classifier: Fruit Example



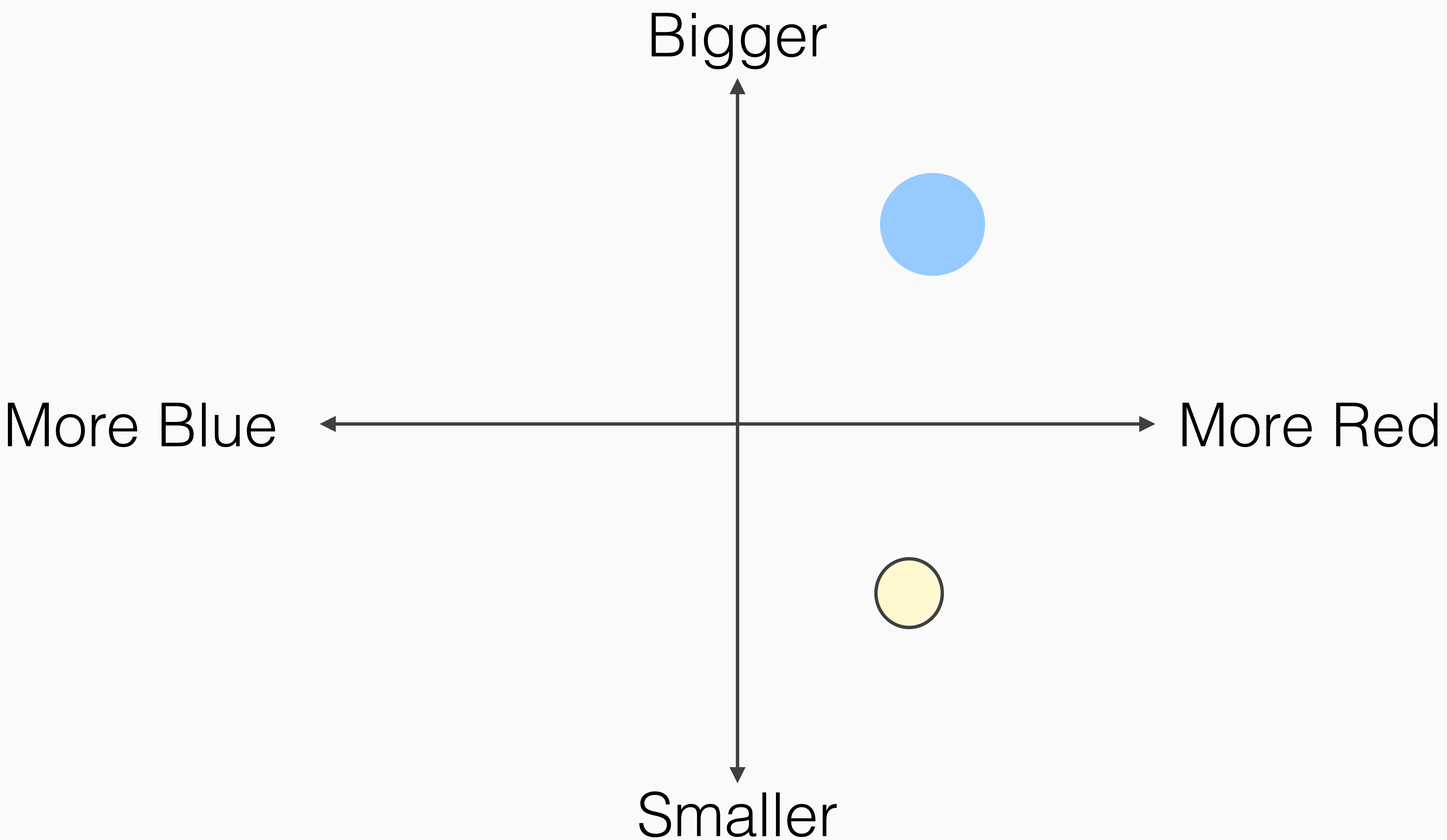
A Classifier: Fruit Example



A Classifier: Fruit Example

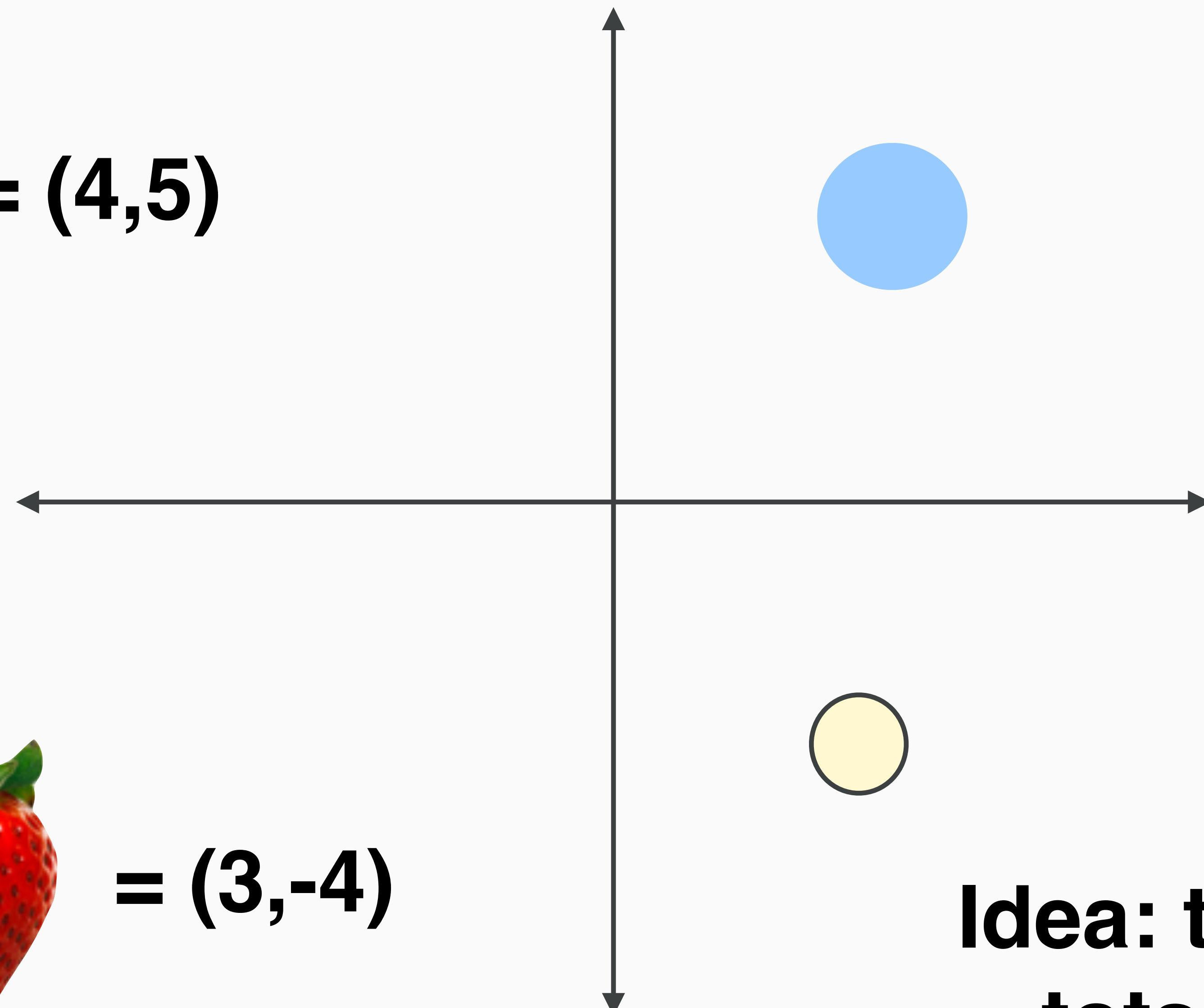


A Classifier: Fruit Example



Features

= (4,5)



= (3,-4)

Idea: these numbers totally represent each fruit!

Clicker Question!

Q: Which of the following might be reasonable features for classifying a recipe's cuisines?

Clicker Question!

[A] A list of cooking tools required to make the food

[B] A list of spices in the recipe

[C] The name of the recipe

[D] The country the recipe comes from

Q: Which of the following might be reasonable features for classifying a recipe’s cuisine?

Clicker Answer!

[A] A list of cooking tools required to make the food

[B] A list of spices in the recipe

[C] The name of the recipe

[D] The country the recipe comes from

Q: Which of the following might be reasonable features for classifying a recipe's cuisine?

Clicker Answer!

[A] A list of cooking tools required to make the food

[B] A lis

[C] The

Features just need to contain some information that bears on the class of the object!

[D] The country the recipe comes from

Q: Which of the following might be reasonable features for classifying a recipe's cuisines?

Clicker Question!

Q: Which feature above will be most useful for classifying SPAM vs. HAM emails?

Clicker Question!

[A] Is the email longer than 500 words?

[B] Is .edu in the email address?

[C] Does the email contain a link?

[D] Does the email contain the word “credit card”?

Q: Which feature above will be most useful for classifying SPAM vs. HAM emails?

Clicker Answer!

[A] Is the email longer than 500 words?

[B] Is .edu in the email address?

[C] Does the email contain a link?

[D] Does the email contain the word “credit card”?

Q: Which feature above will be most useful for classifying SPAM vs. HAM emails?

Clicker Answer!

[A] Is the email longer than 500 words?

[B] Is .edu in the email address?

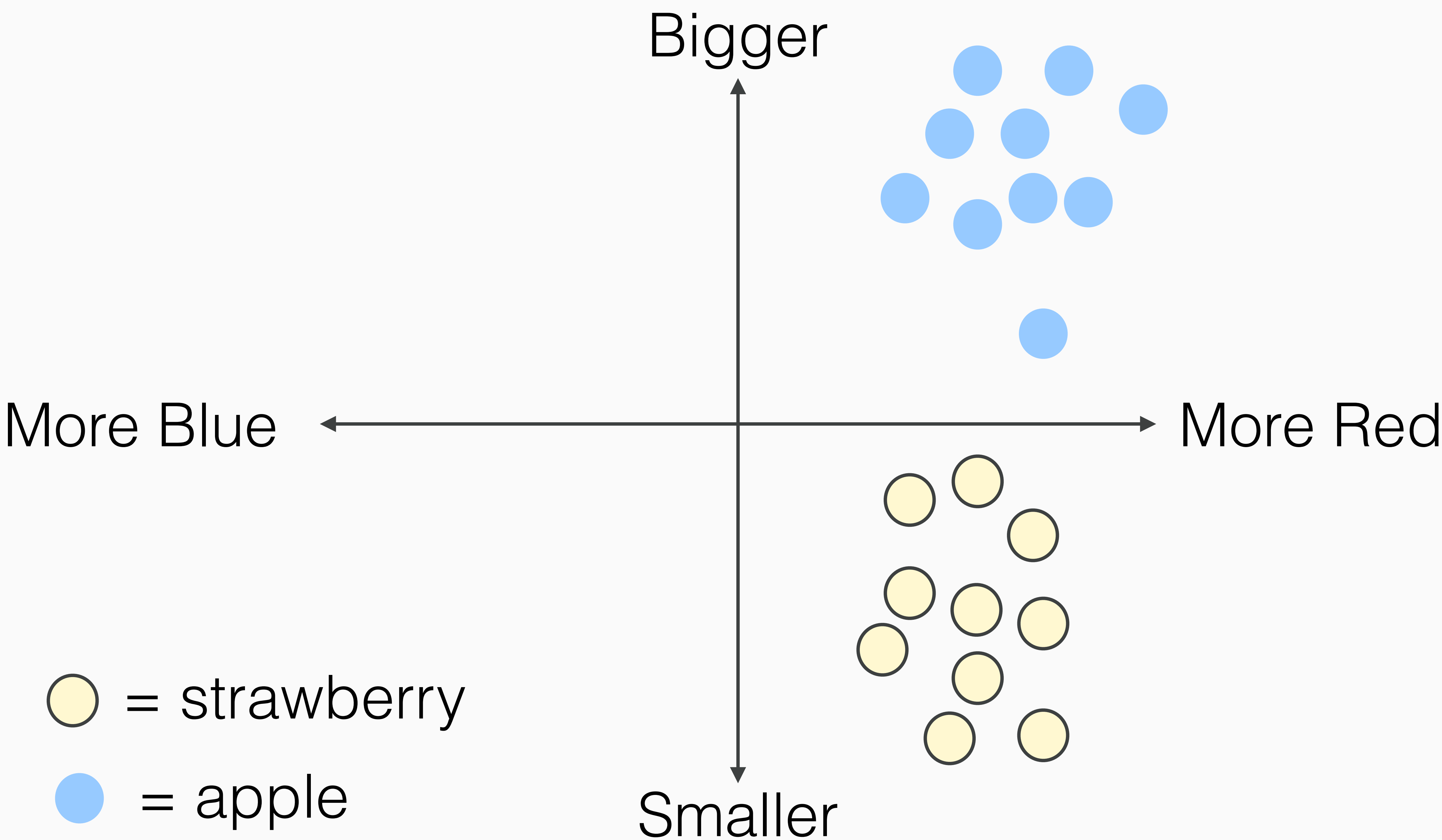
[C] Does the email contain a link?

[D] Does the email contain the word "credit card"?

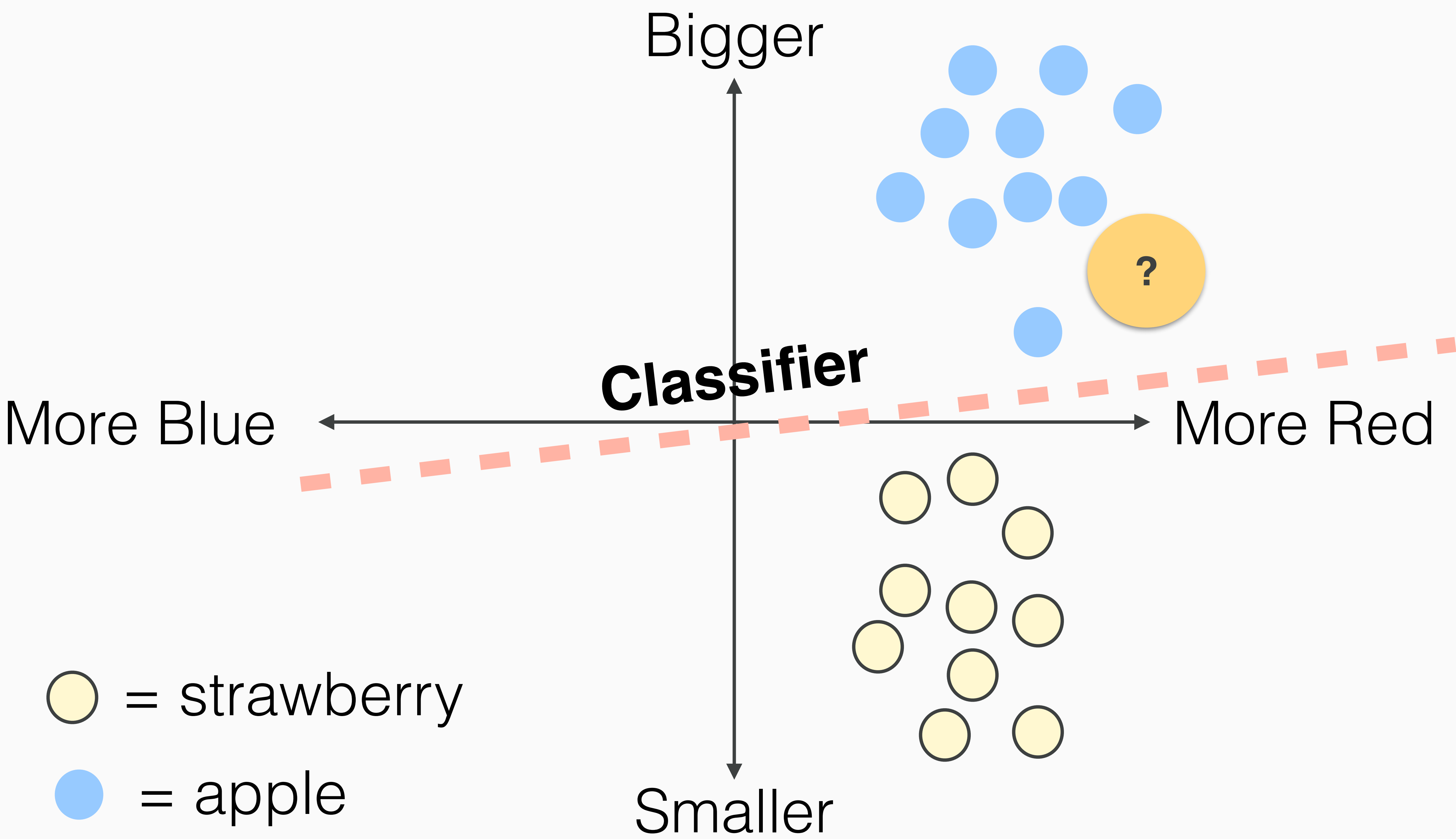
Note: could also imagine the AND of D and C would be really effective as a feature!

Q: Wh for

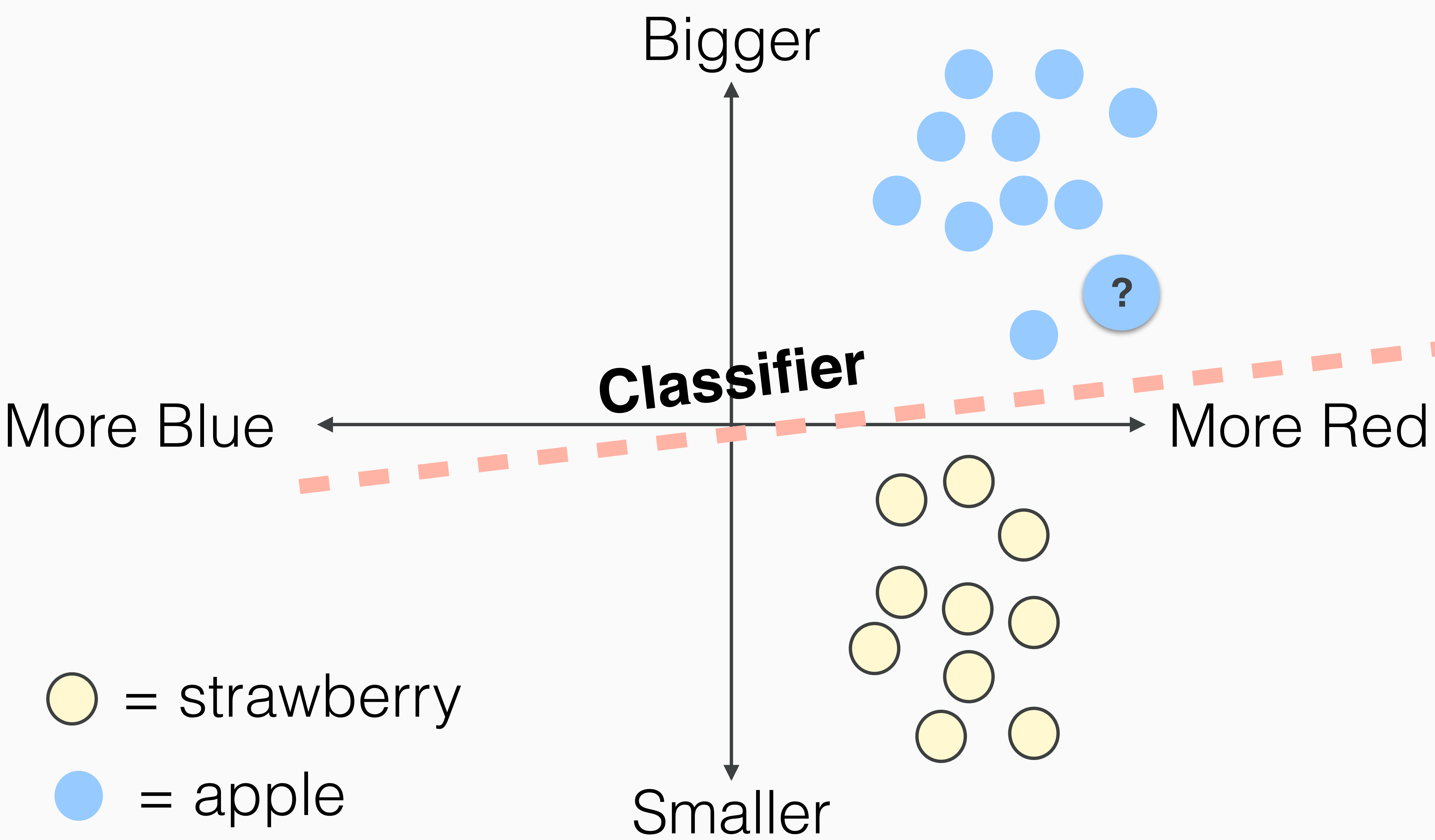
A Classifier: Fruit Example



A Classifier: Fruit Example



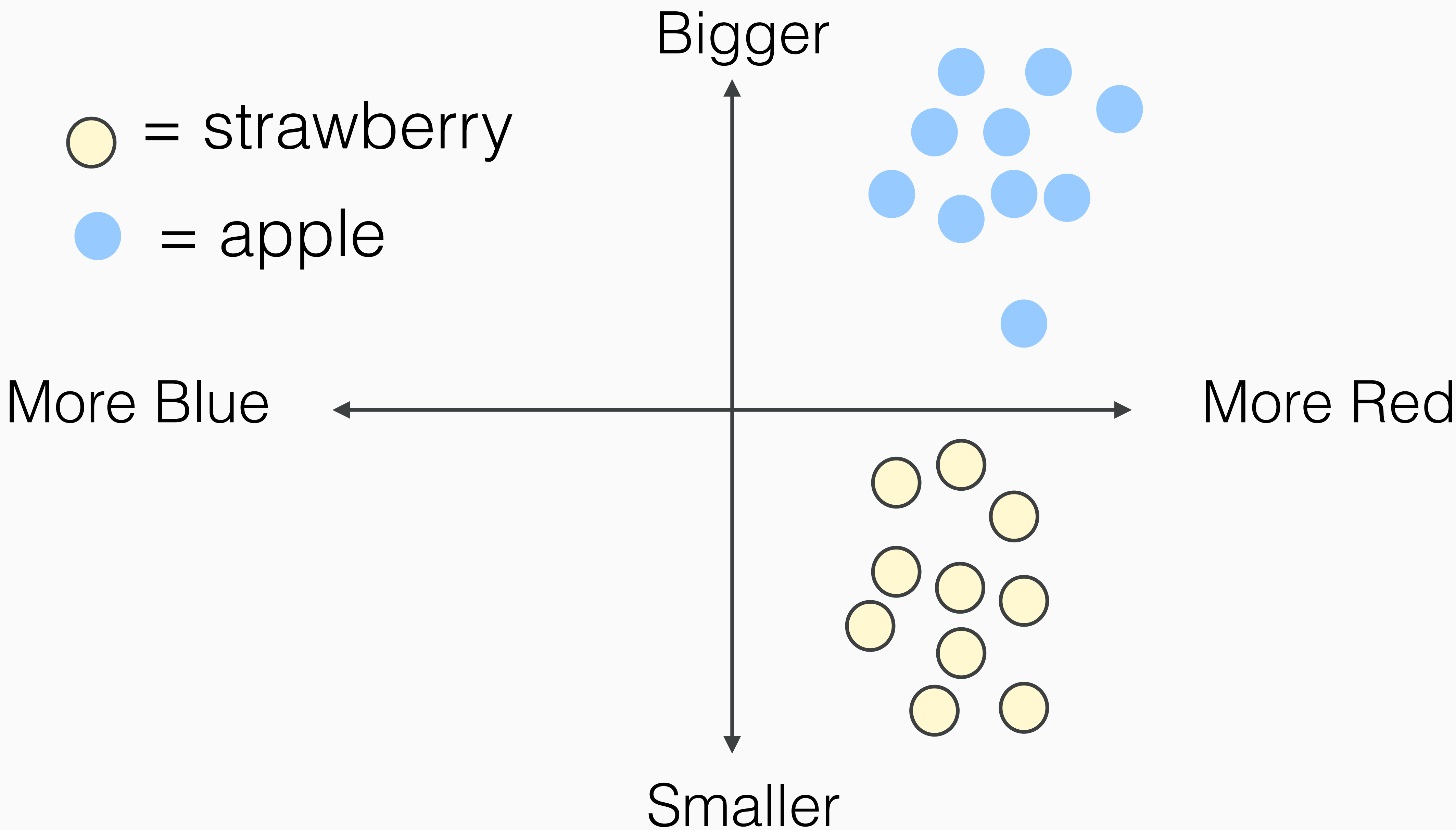
A Classifier: Fruit Example



Machine Learning

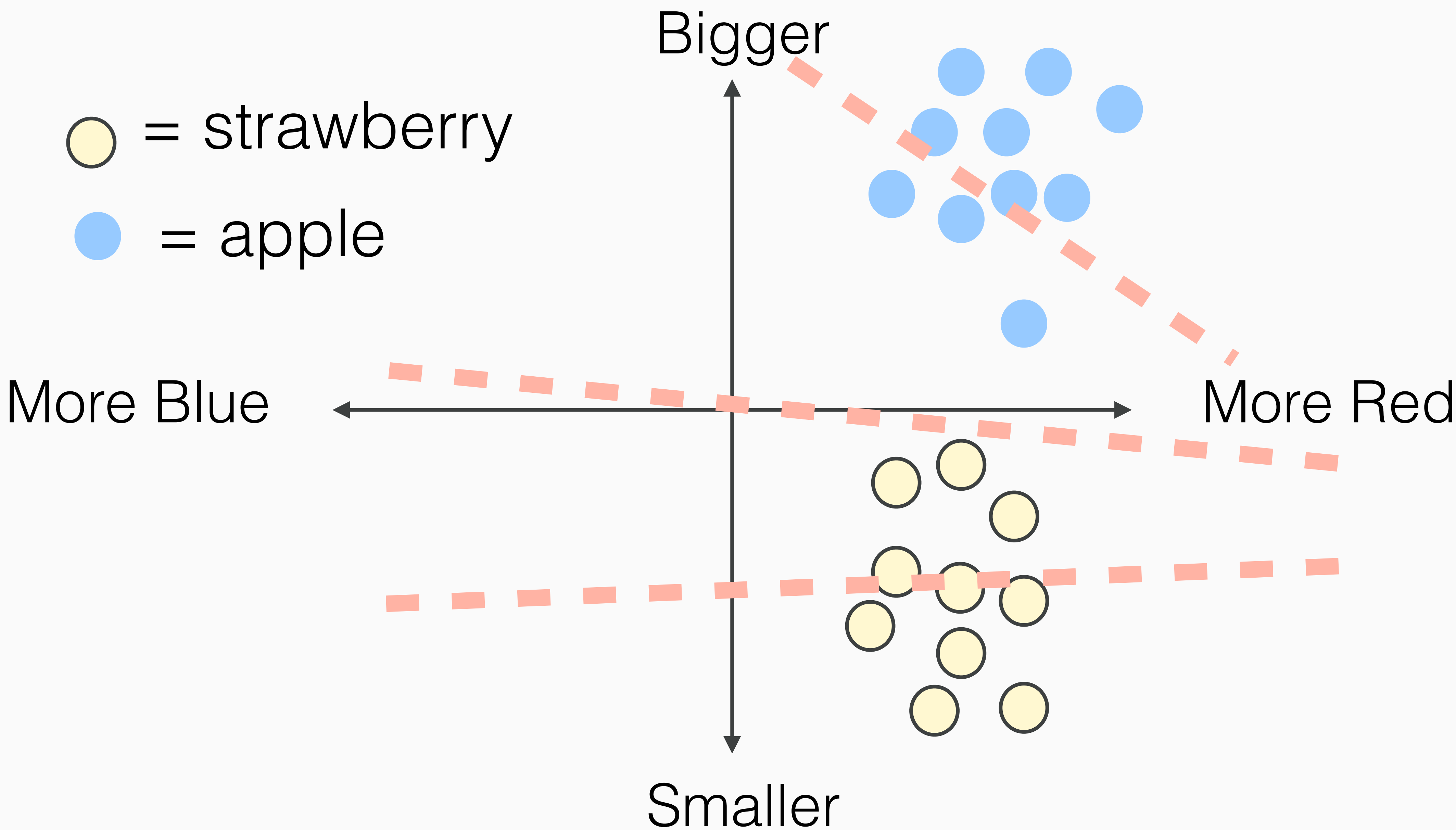
- INPUT: Some labeled training data (some fruits, with their name)
- OUTPUT: A classifier.
- **Goal:** Spit out the classifier that will classify the most things *correctly.*
- **Tools:** What do you do with the training data you receive to inform your classifier?

Machine Learning



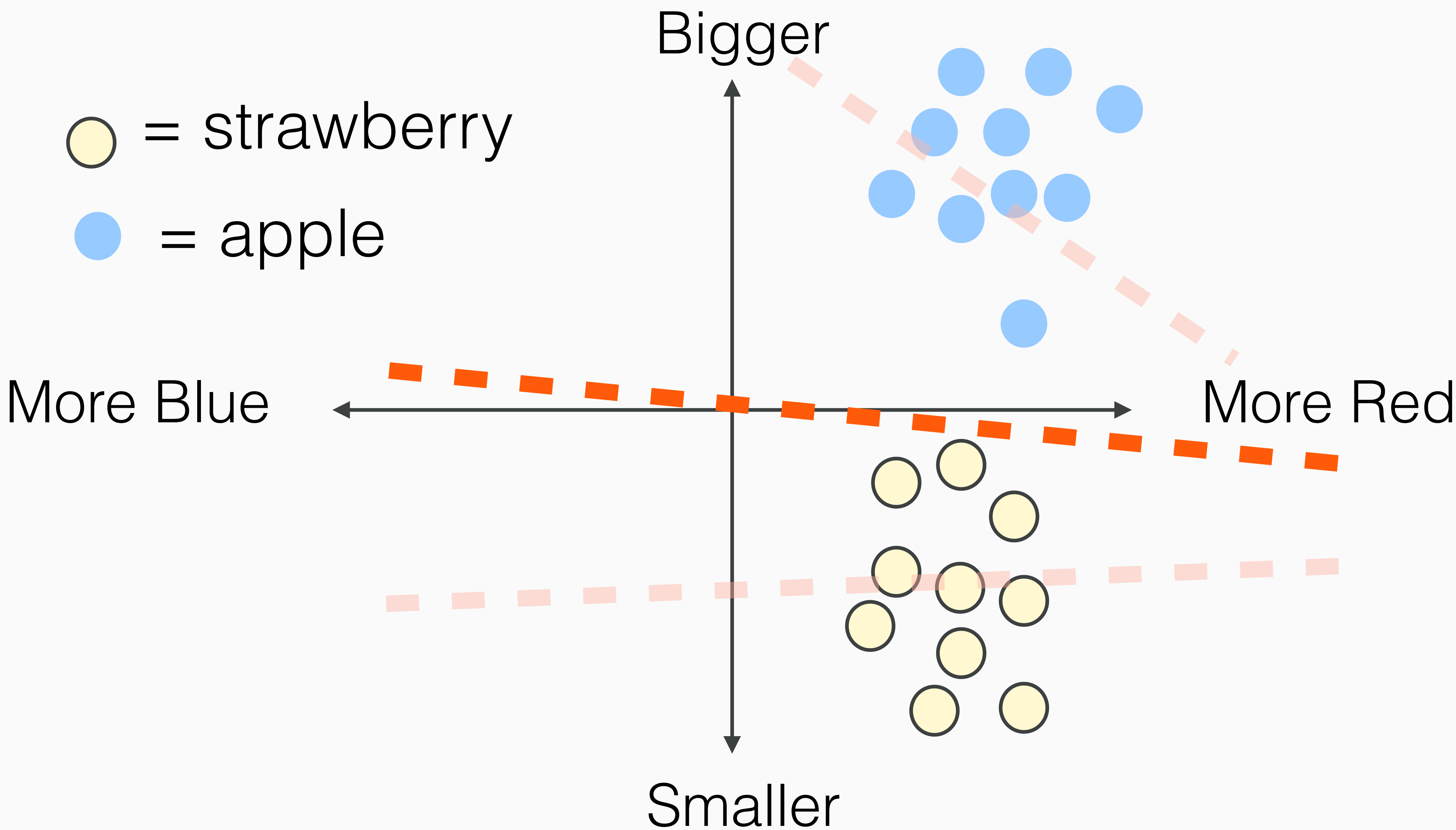
- **Goal:** Spit out the classifier that will classify the most things *correctly*.

Machine Learning



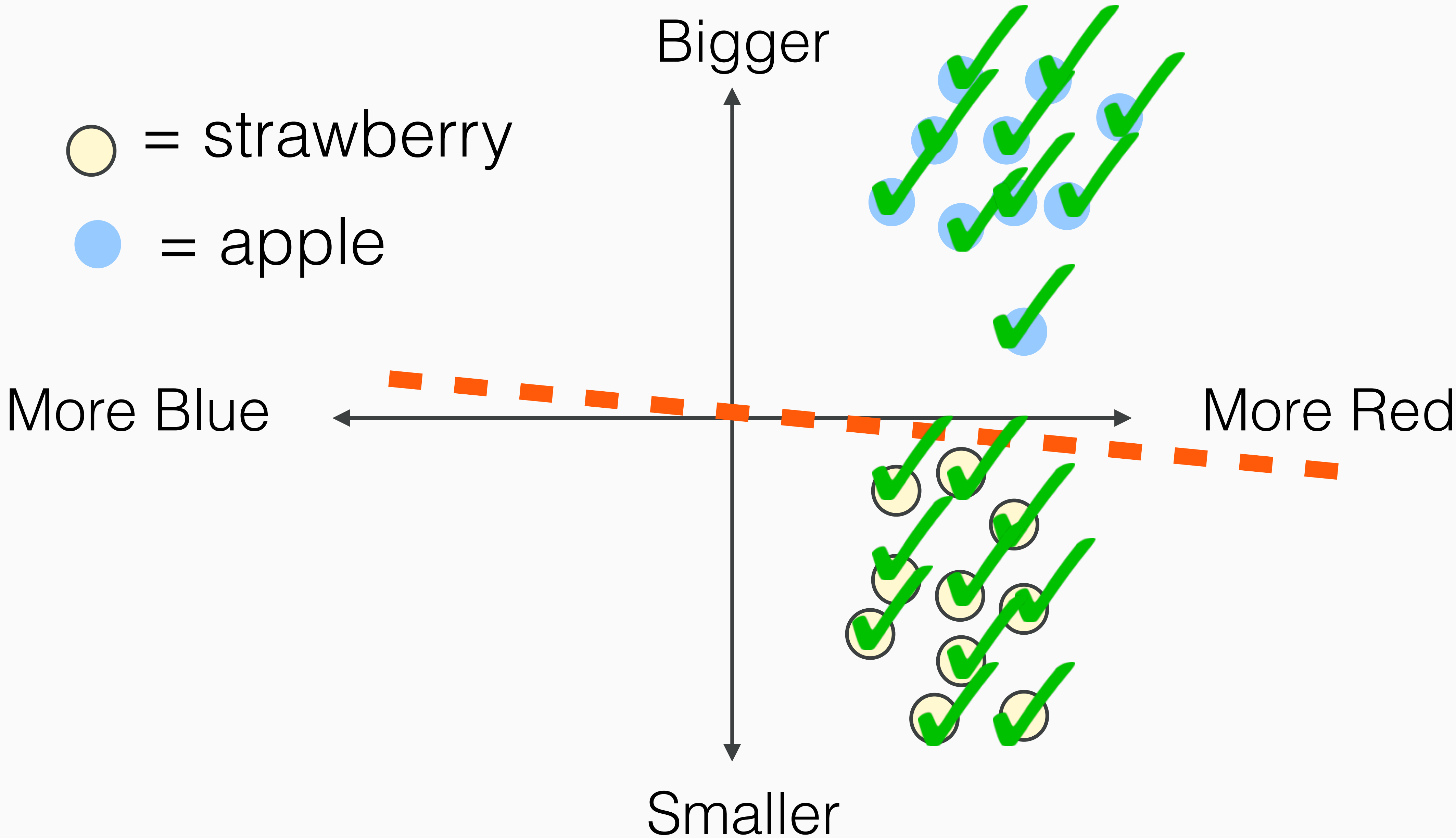
- **Goal:** Spit out the classifier that will classify the most things *correctly*.

Machine Learning



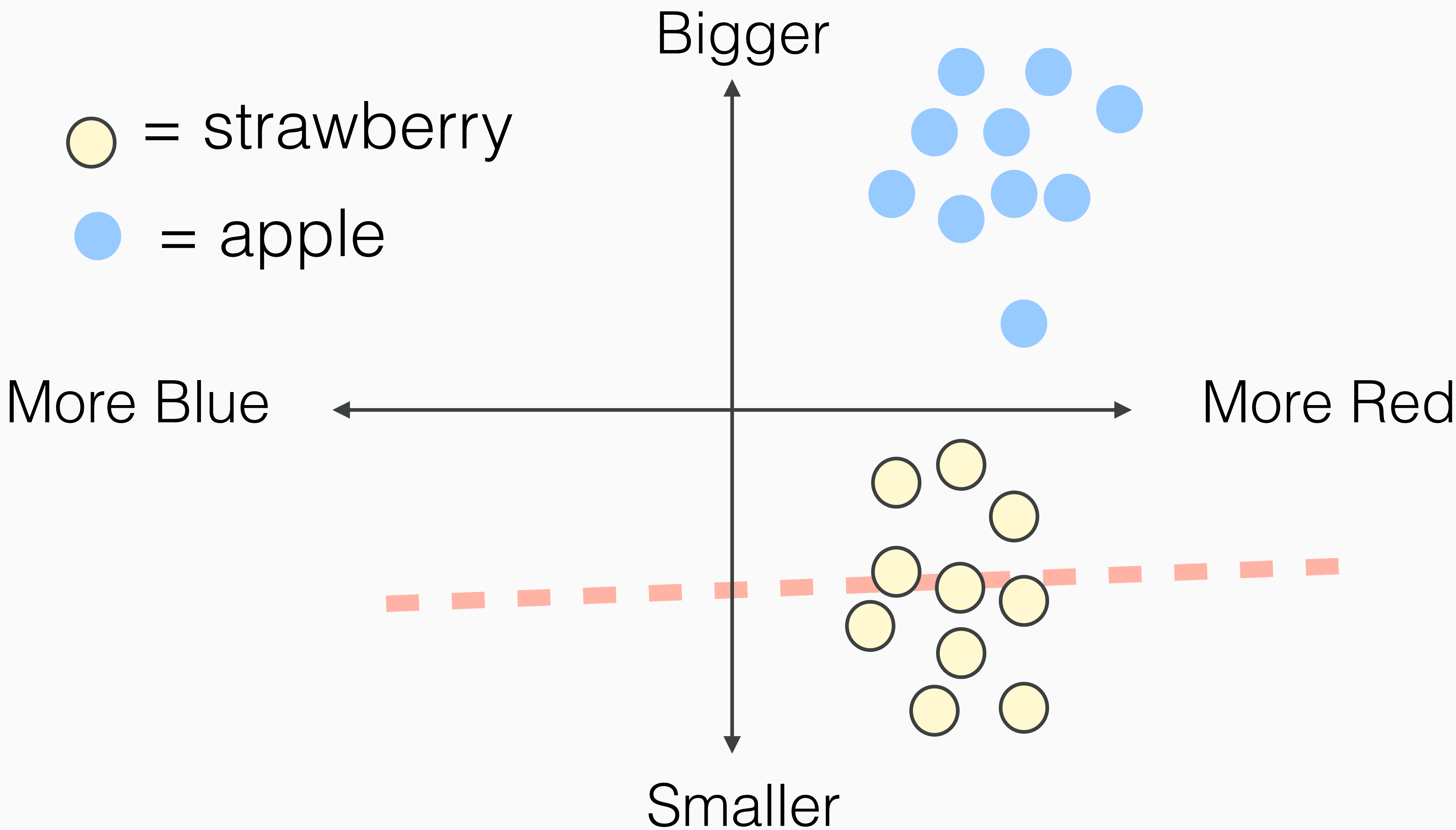
- **Goal:** Spit out the classifier that will classify the most things *correctly*.

Machine Learning



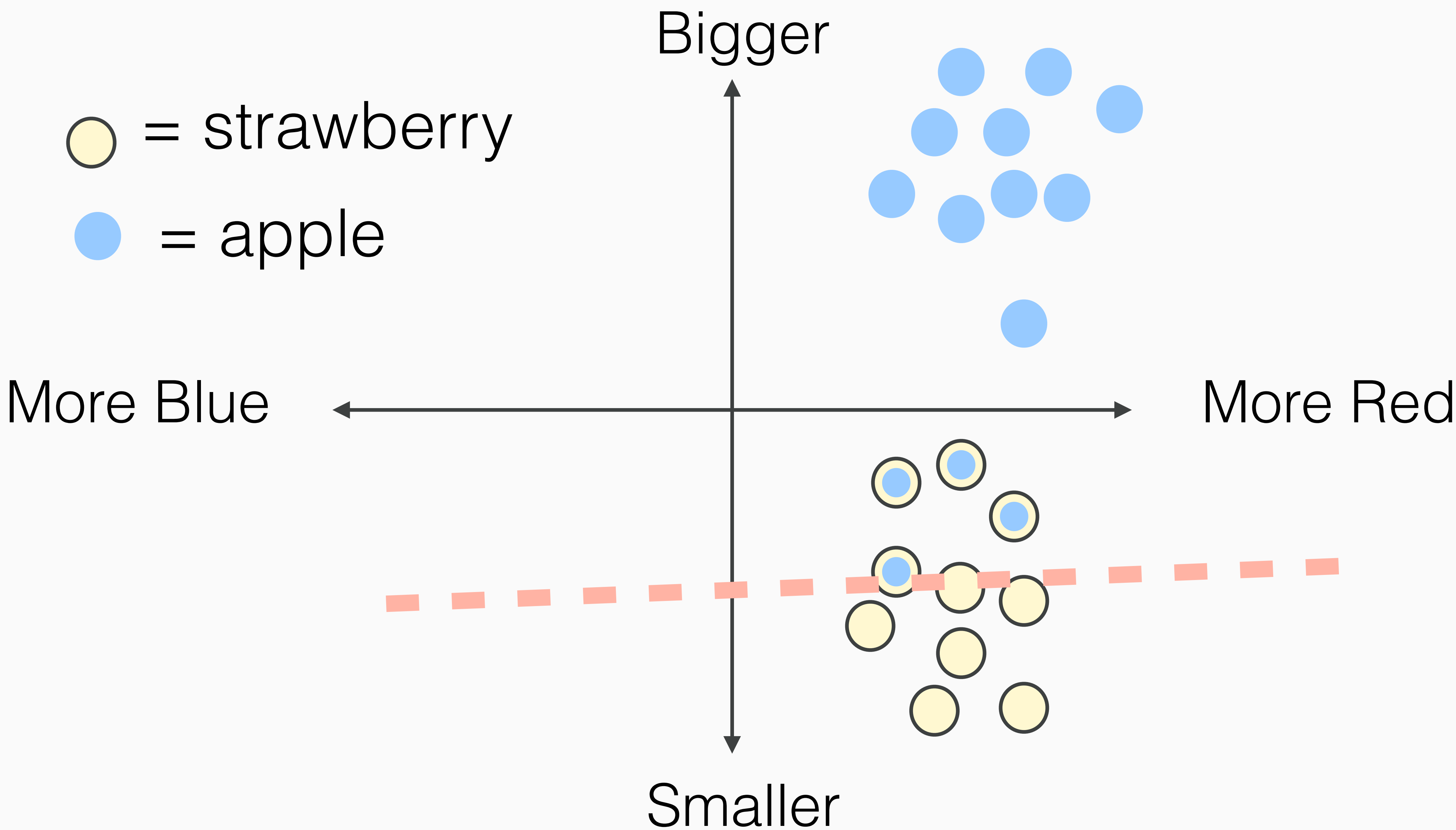
- **Goal:** Spit out the classifier that will classify the most things *correctly*.

Machine Learning



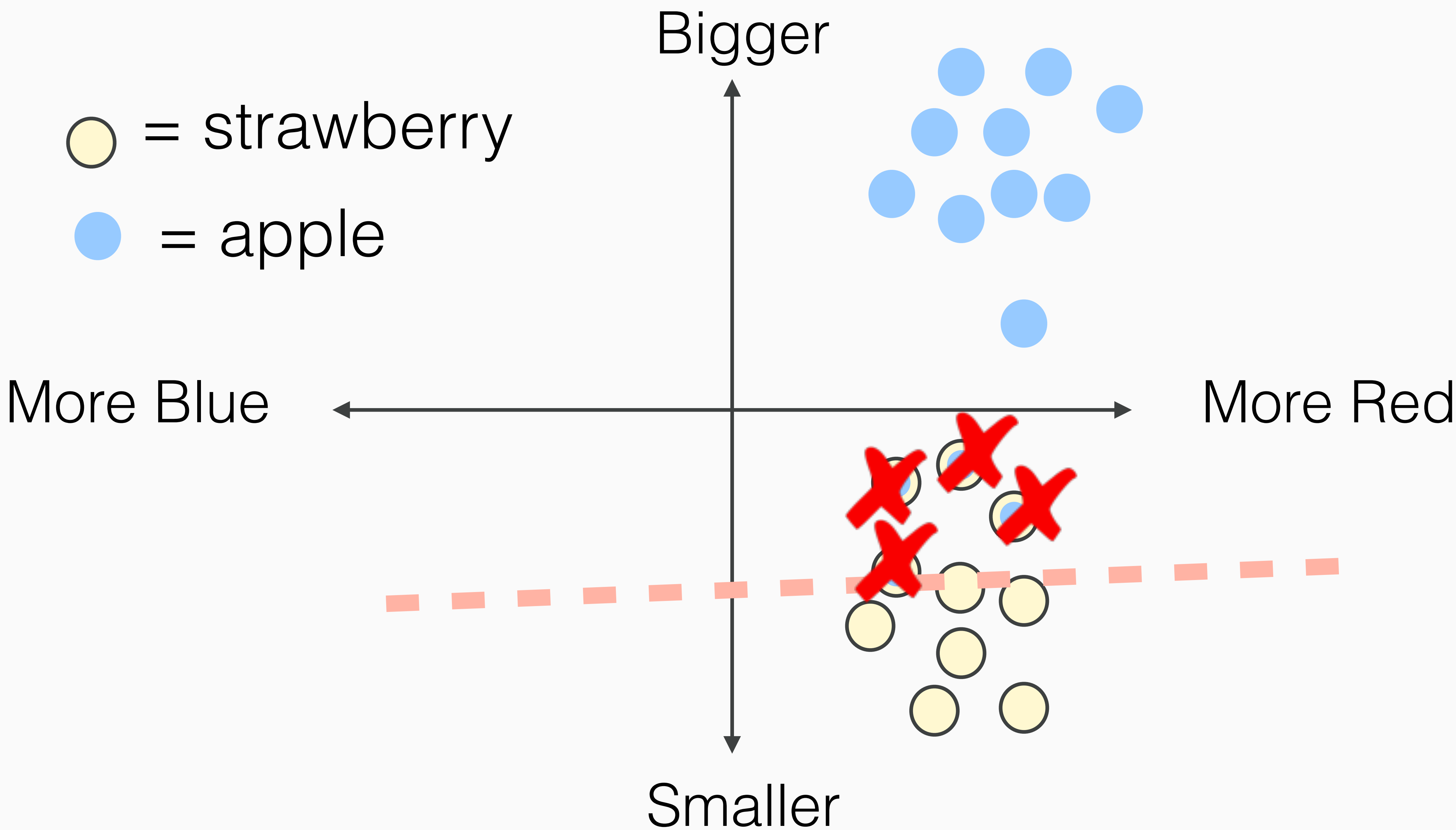
- **Goal:** Spit out the classifier that will classify the most things *correctly*.

Machine Learning



- **Goal:** Spit out the classifier that will classify the most things *correctly*.

Machine Learning

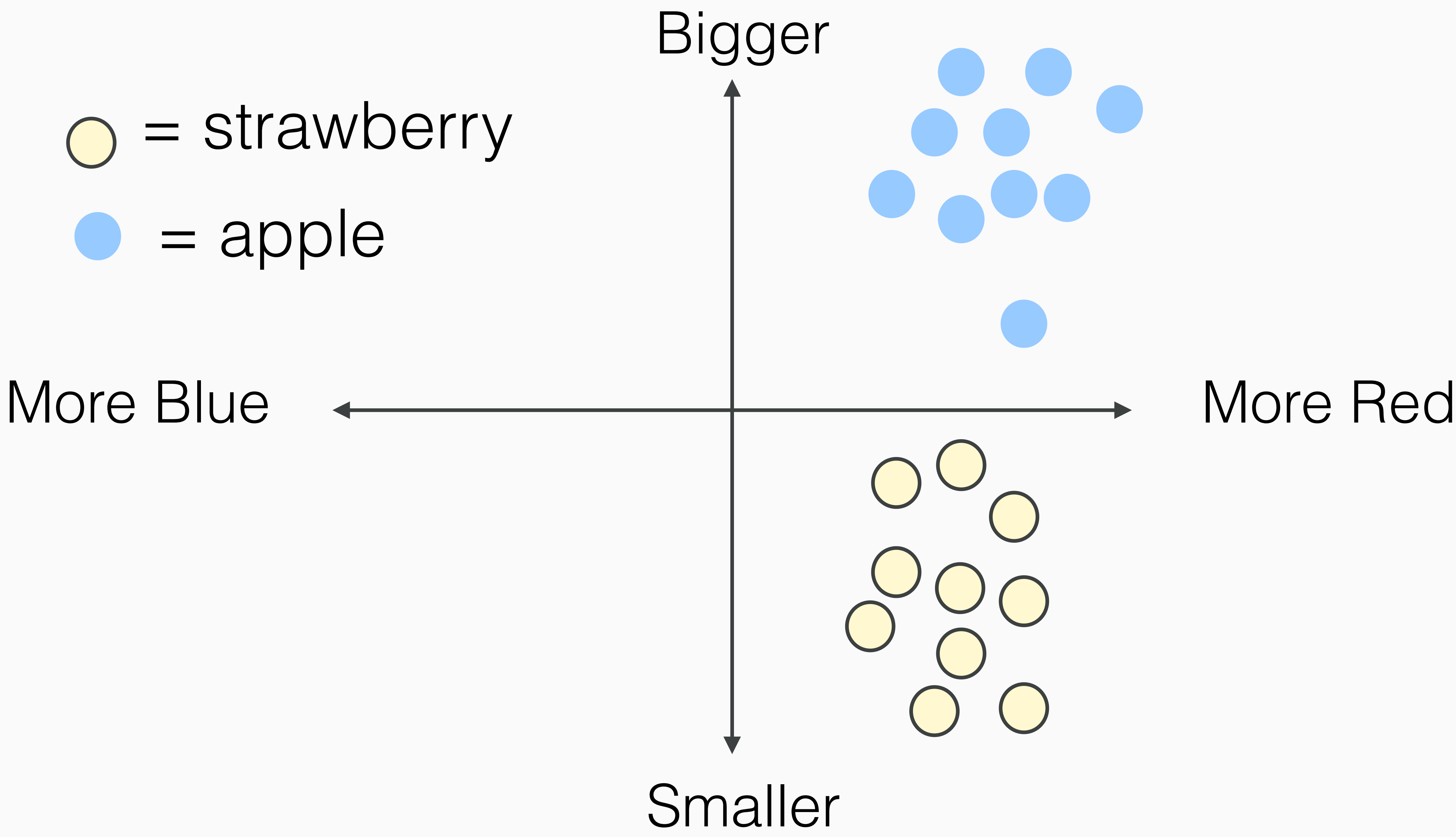


- **Goal:** Spit out the classifier that will classify the most things *correctly*.

Machine Learning

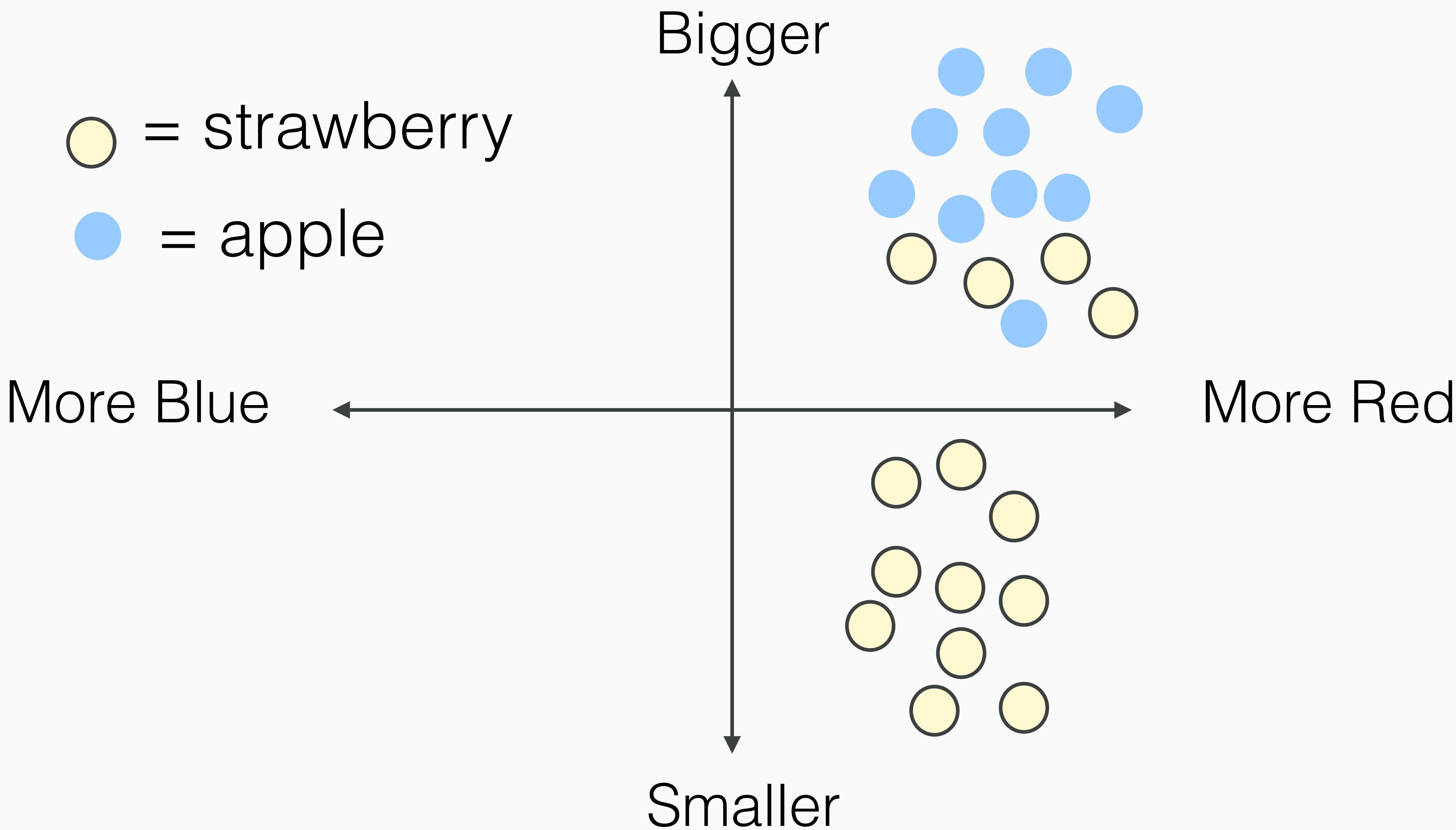
- INPUT: Some labeled *training data* (some fruits, with their name)
- OUTPUT: A classifier.
- **Goal:** Spit out the classifier that will classify the most things *correctly*.
- *Training Data:* finite! Only represents a small space of the thing we're trying to learn.

Training Data



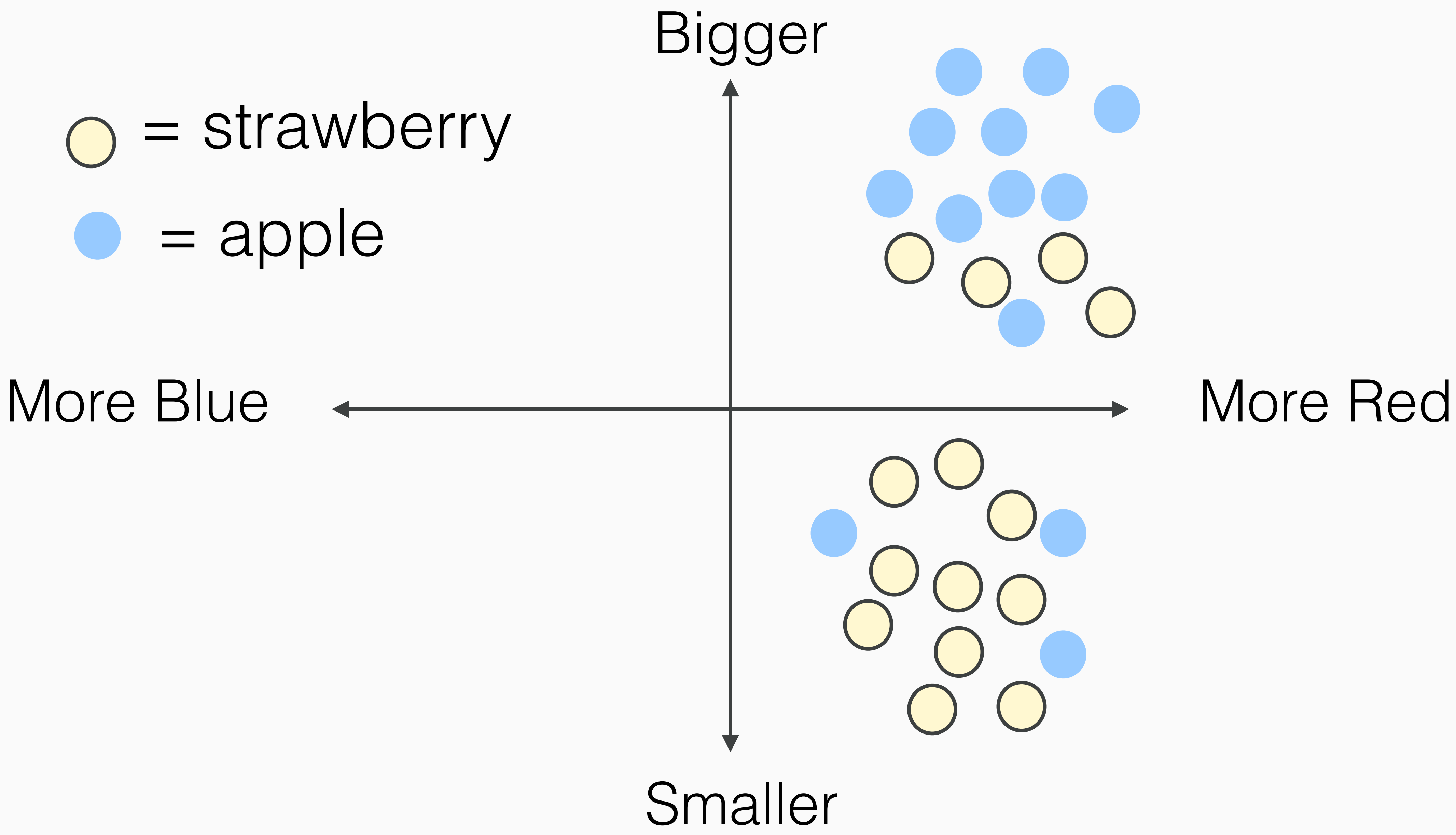
Note: There are *way more than 10 apples in the world*

Training Data



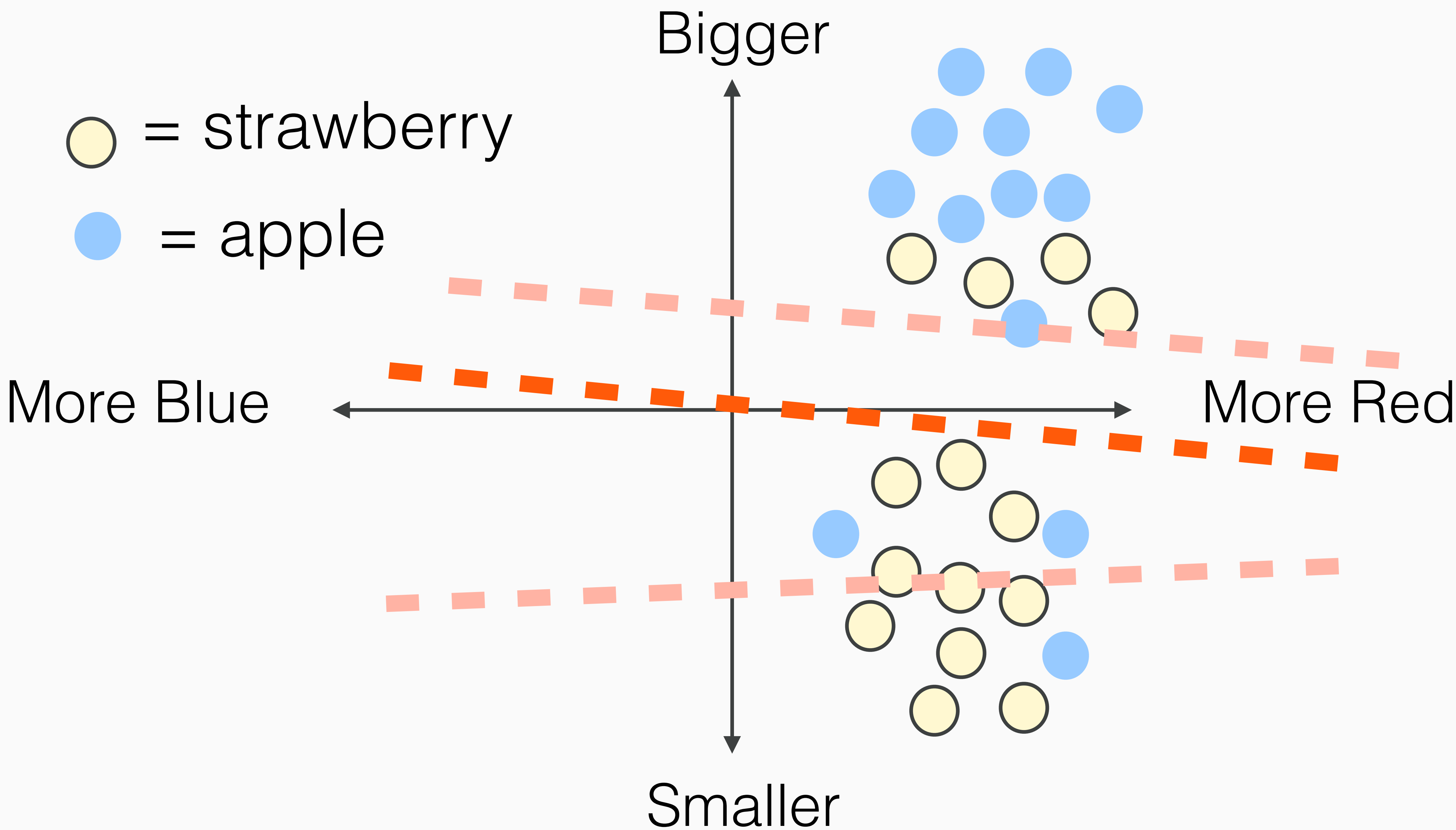
Note: There may be huge strawberries!

Training Data



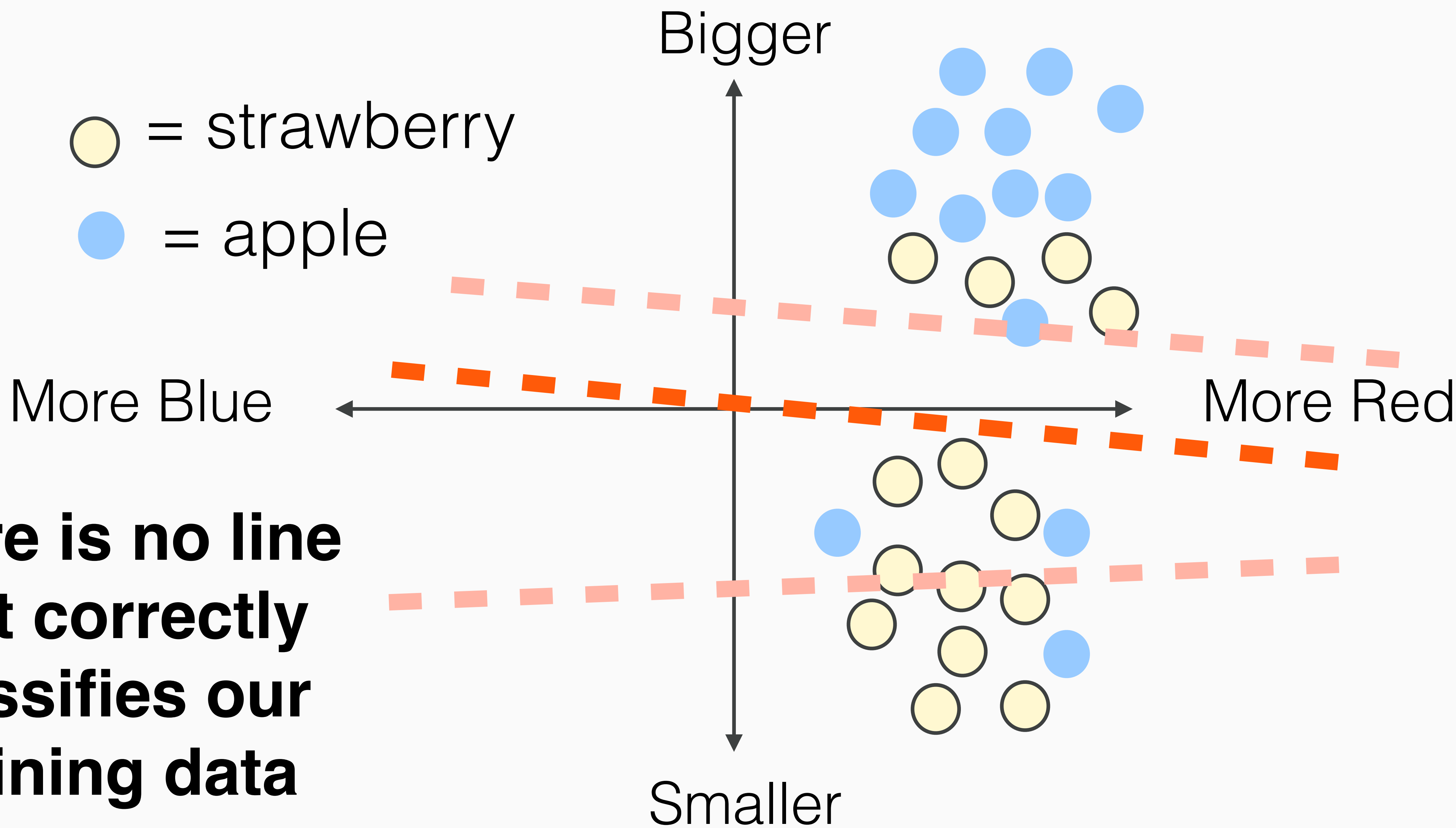
Note: There may be huge strawberries! And tiny apples.

Training Data



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

Training Data



There is no line that correctly classifies our training data

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

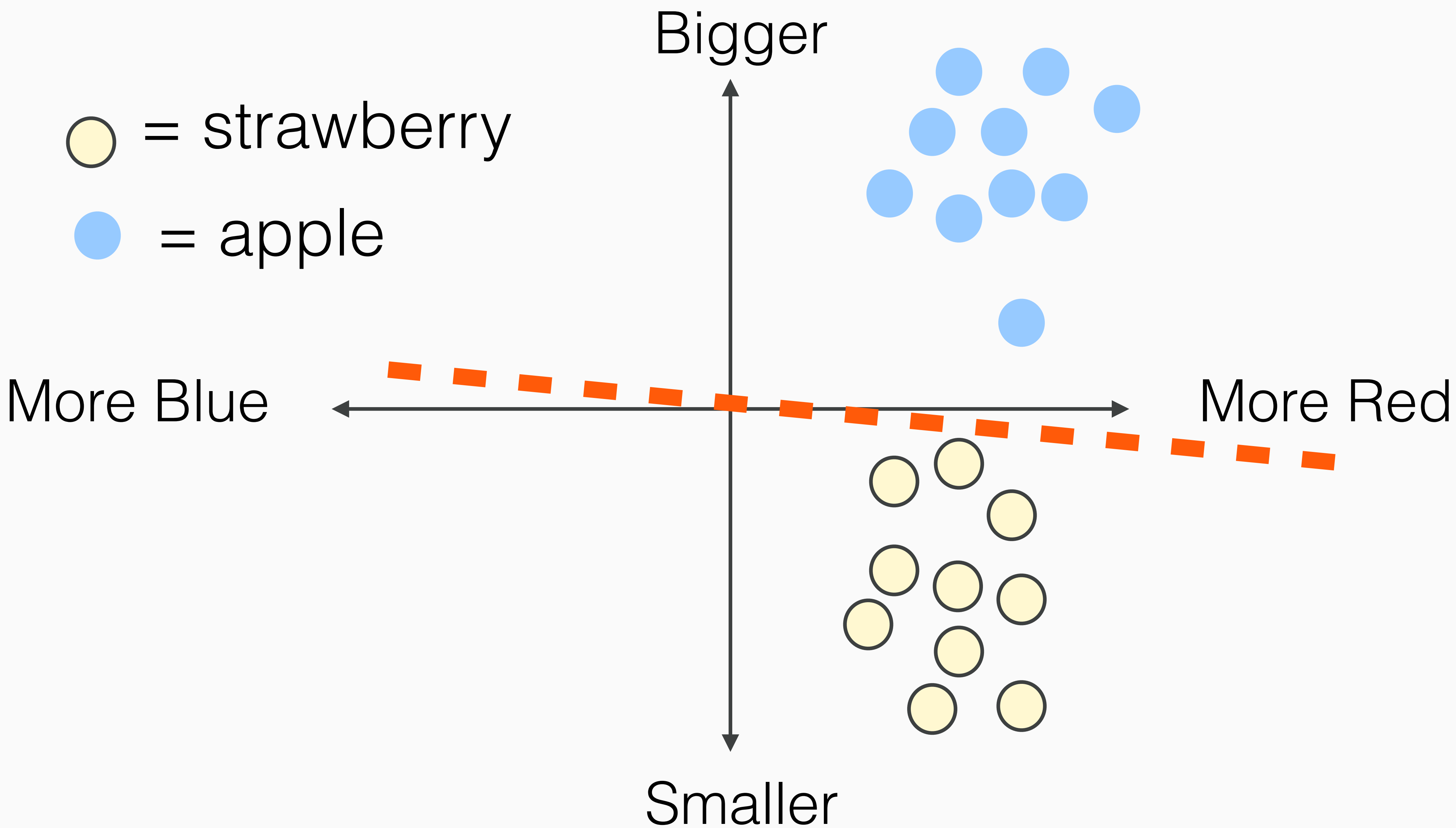
Classification

1. The algorithm receives a bunch of labeled training data.

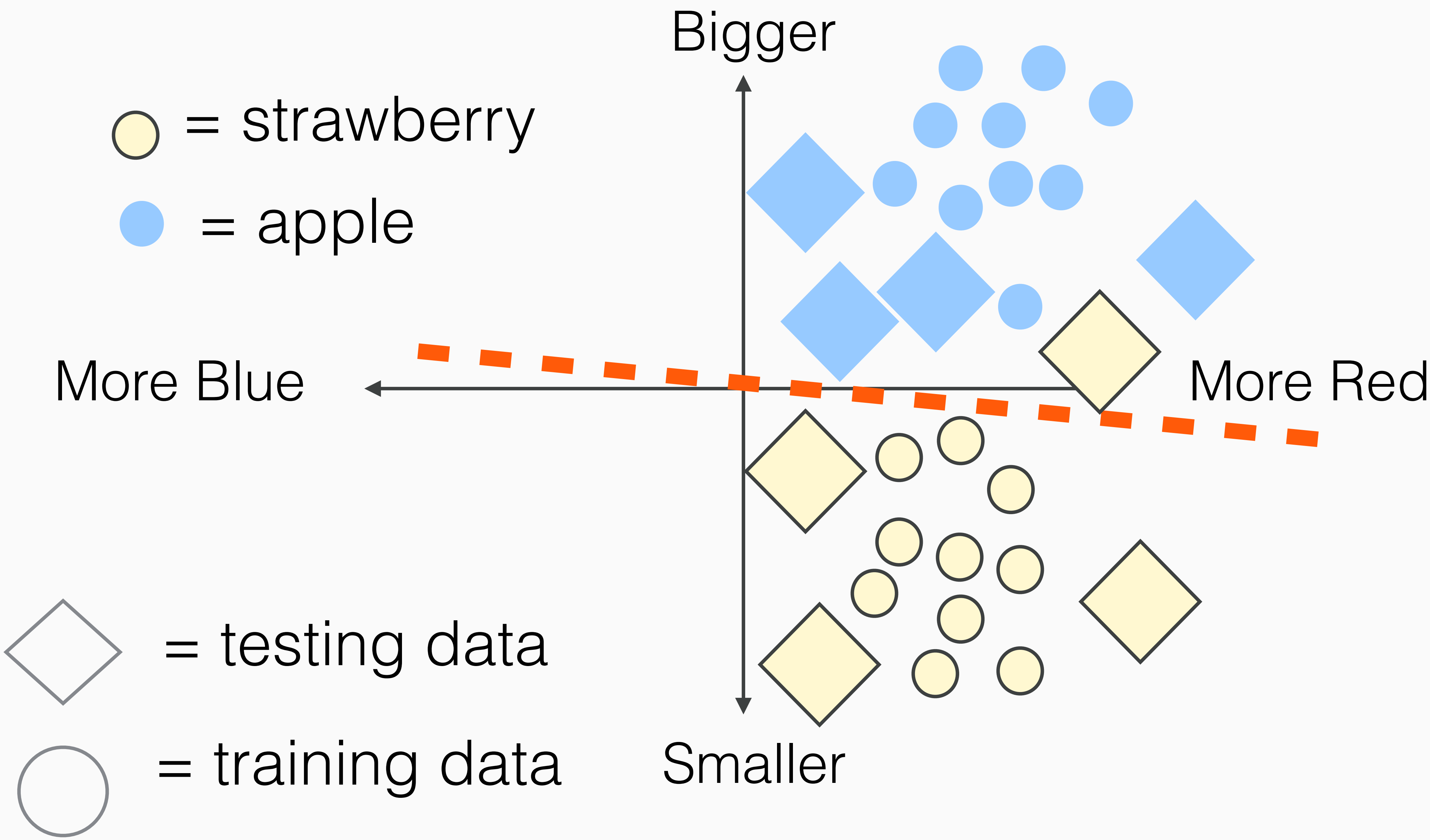
2. The algorithm tries to use the labels to learn the best classifier it can (rule for distinguishing between classes).

3. We **evaluate** learning algorithms according to how well their classifier is able to classify things more generally. This round is called, “testing” (so the data used to test is called “testing data”).

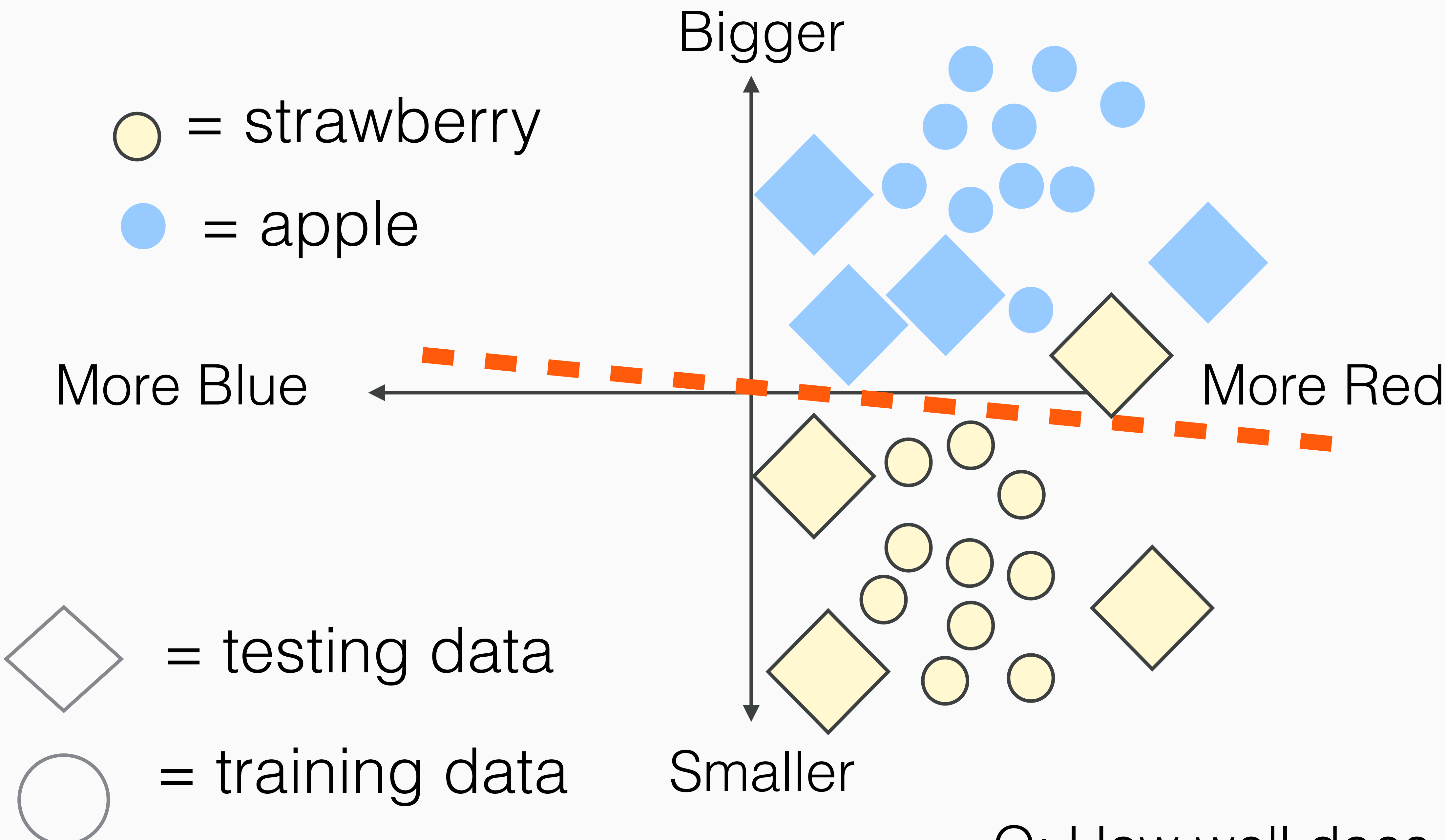
Testing Data



Testing Data

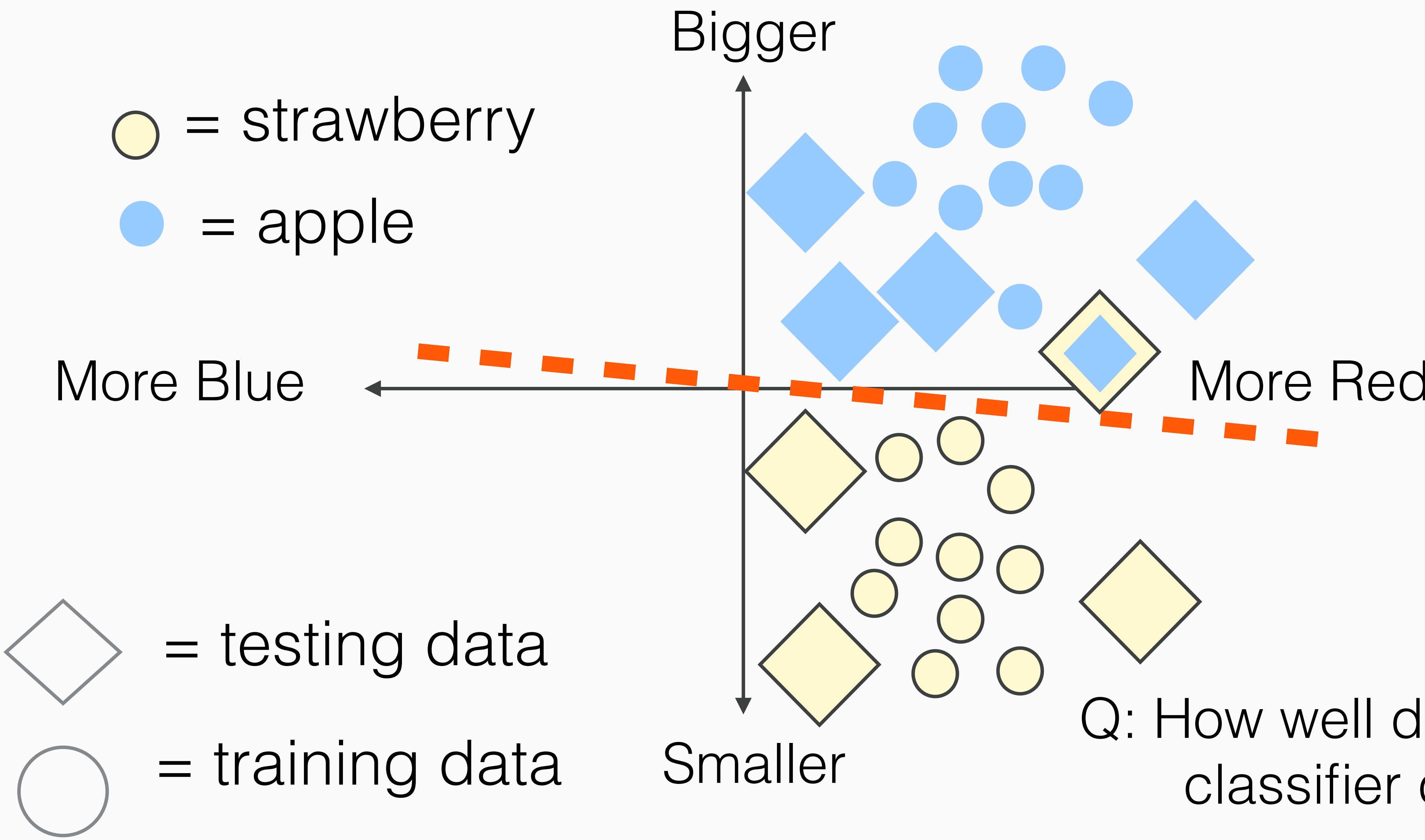


Testing Data



Q: How well does this classifier do?

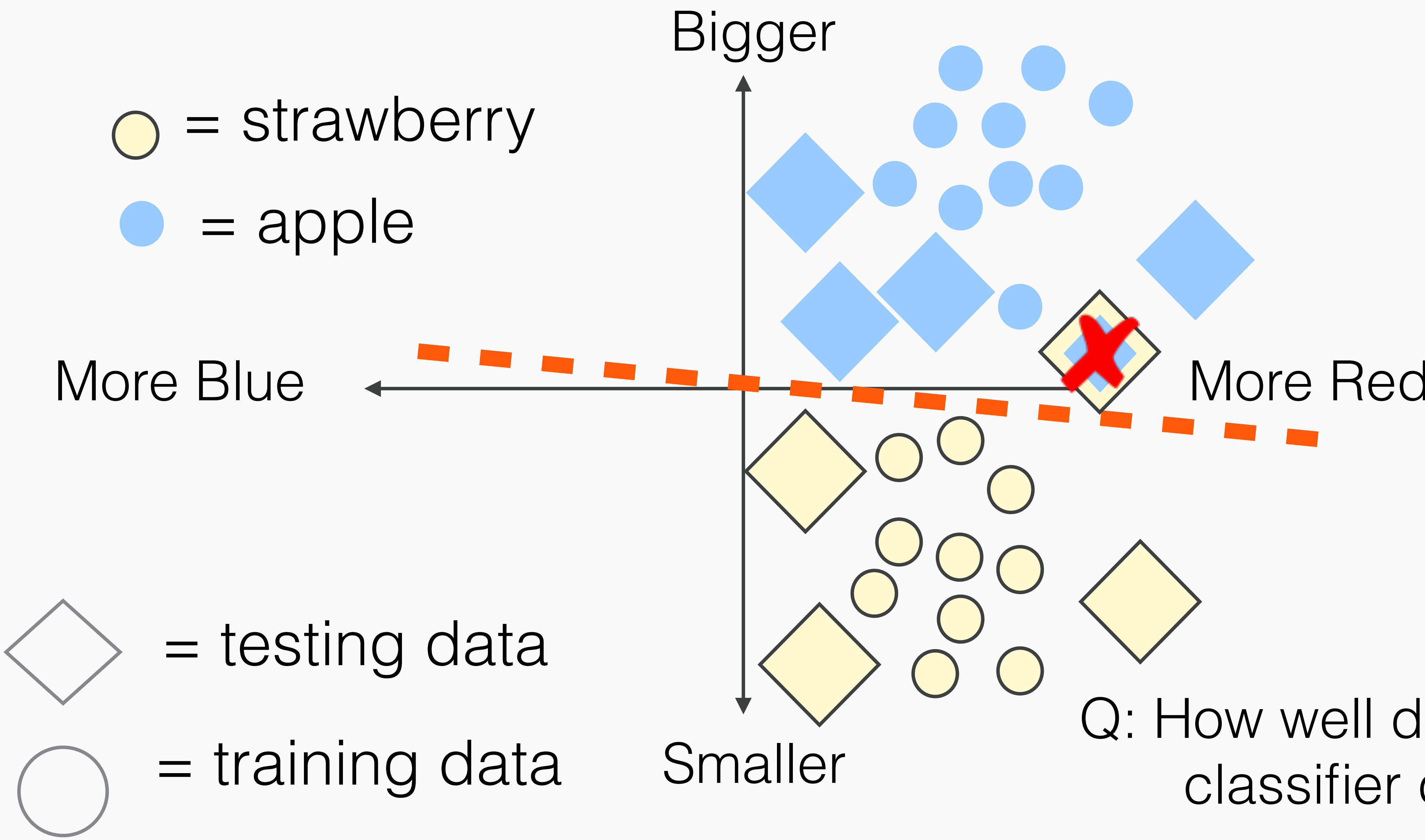
Testing Data



Q: How well does this classifier do?

A: Super well! Only misses one

Testing Data



Q: How well does this classifier do?

A: Super well! Only misses one

Our First Classification Algorithm: Memorize And Guess

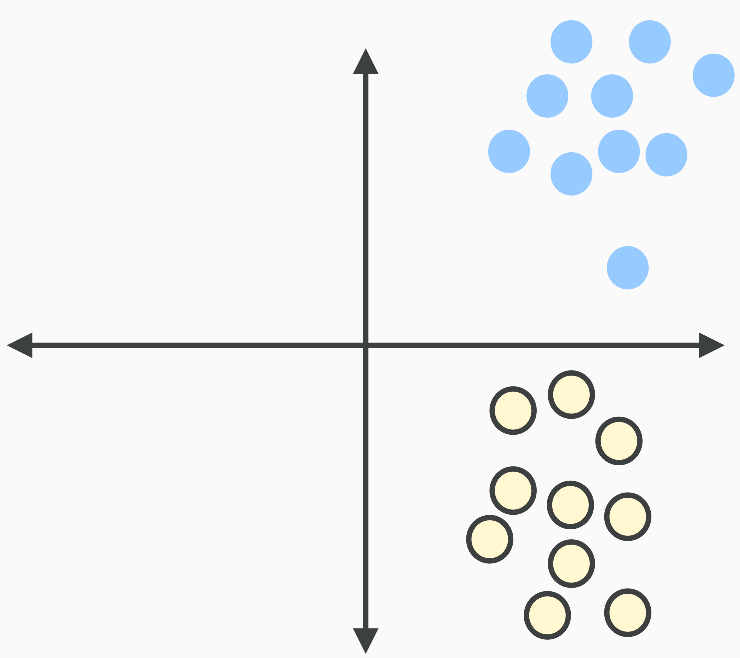
1. Memorize every training data-label pair we see.

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

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

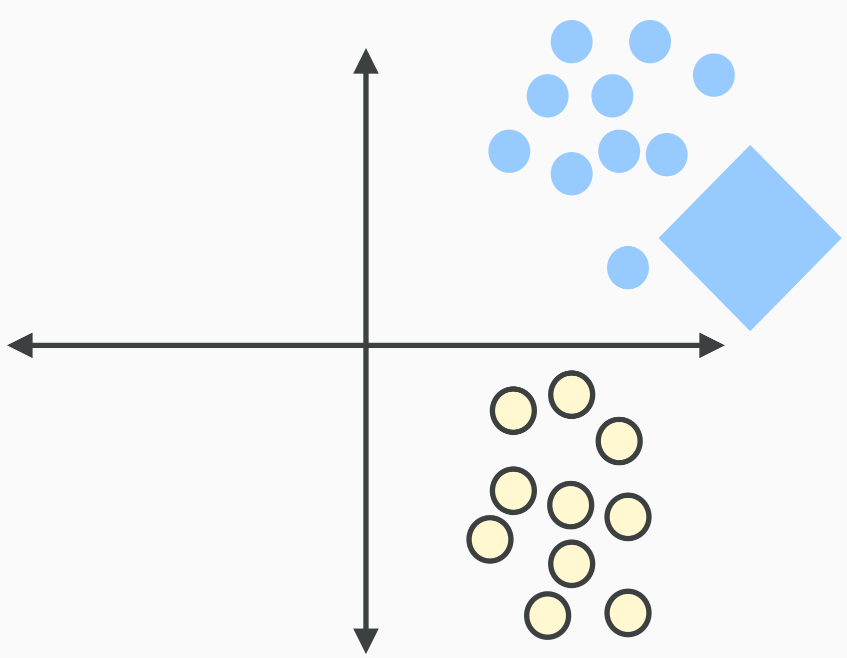
2. Create a classifier that, when given any item seen in our training data, reports exactly that item's label. If it gets an item it's never seen before, guess the label randomly.



Our First Classification Algorithm: Memorize And Guess

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

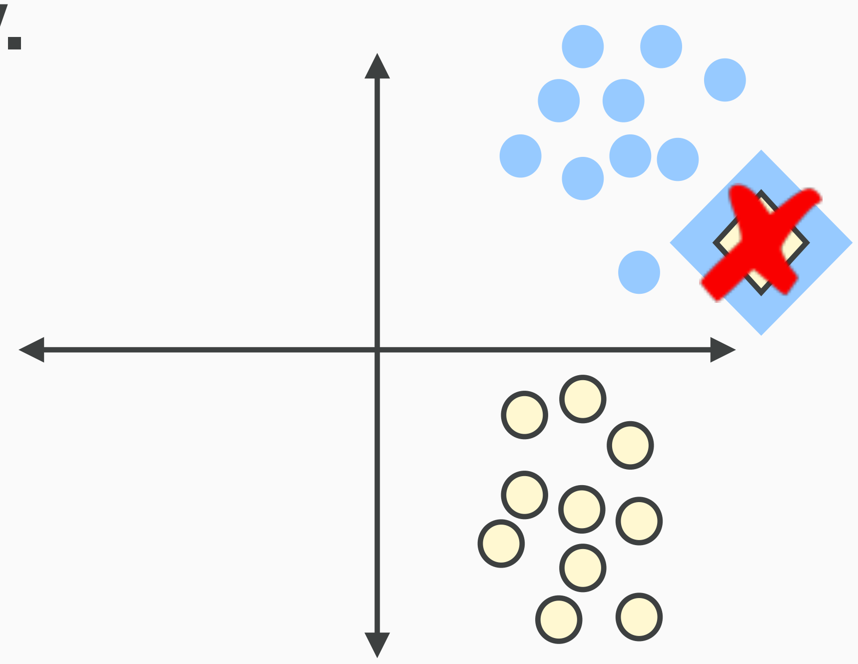
2. Create a classifier that, when given any item seen in our training data, reports exactly that item's label. If it gets an item it's never seen before, guess the label randomly.



Our First Classification Algorithm: Memorize And Guess

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

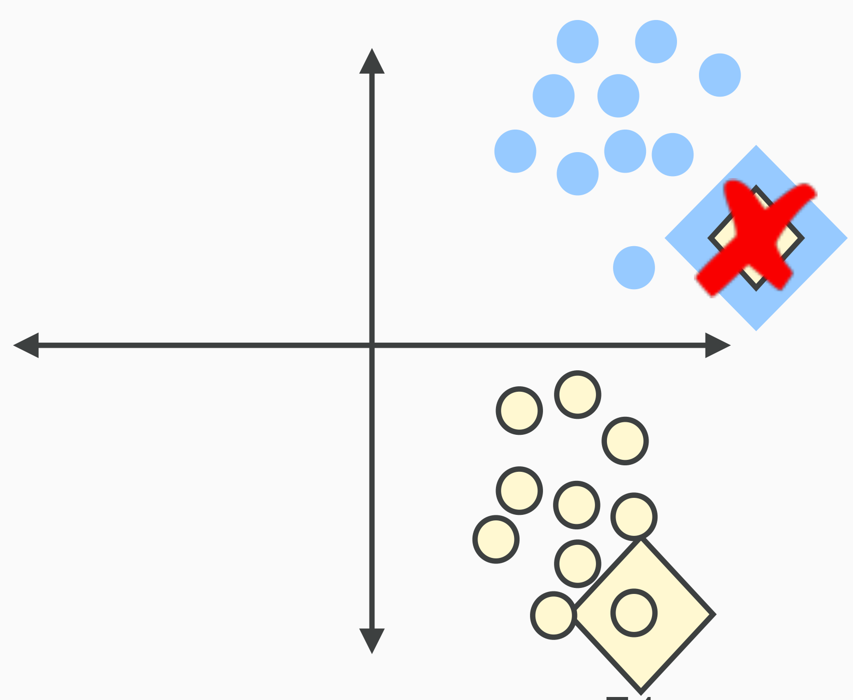
2. Create a classifier that, when given any item seen in our training data, reports exactly that item's label. If it gets an item it's never seen before, **guess the label randomly.**



Our First Classification Algorithm: Memorize And Guess

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

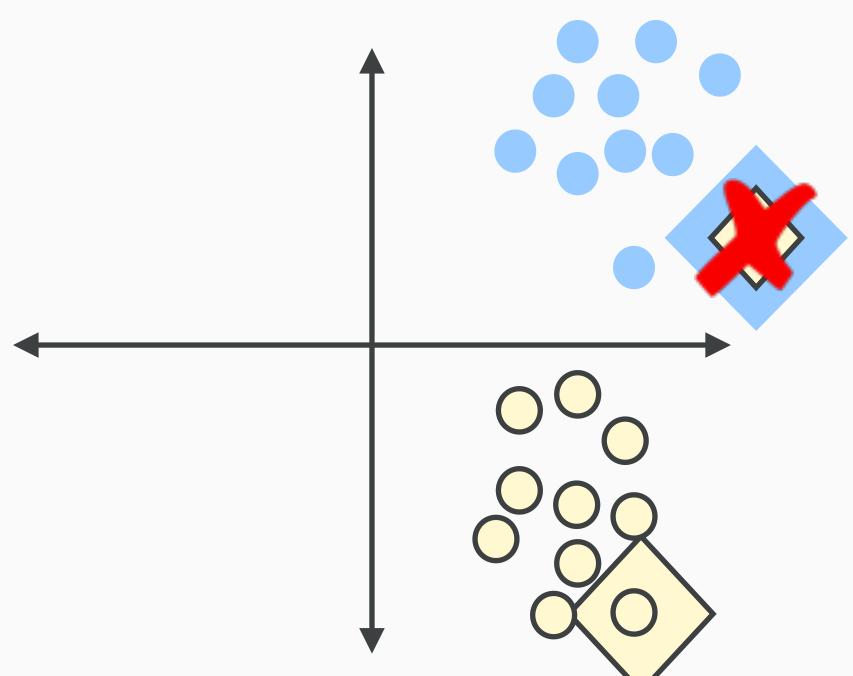
2. Create a classifier that, when given any item seen in our training data, reports exactly that item's label. If it gets an item it's never seen before, guess the label randomly.



Our First Classification Algorithm: Memorize And Guess

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

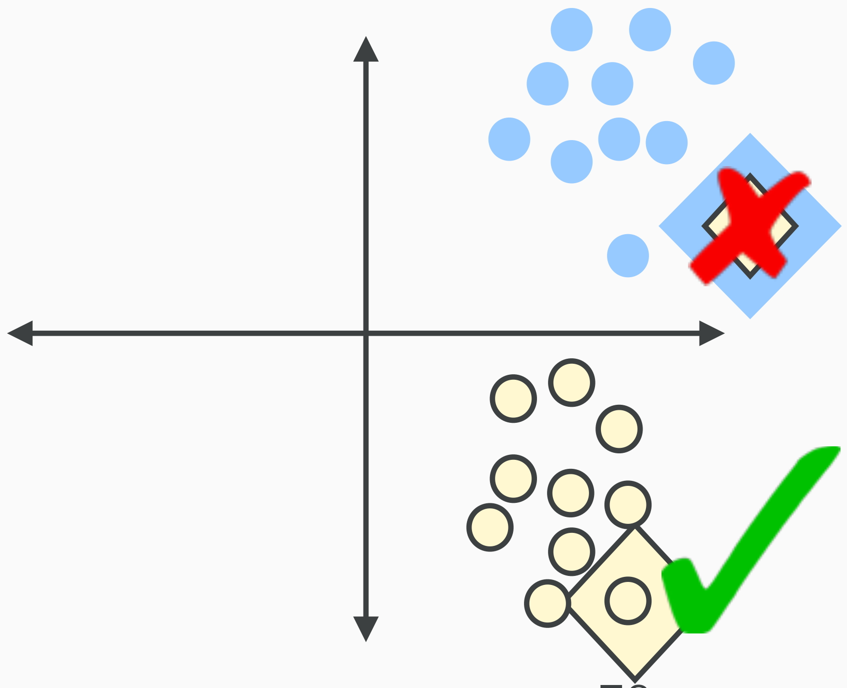
2. Create a classifier that, when **given any item seen in our training data**, reports exactly that item's label. If it gets an item it's never seen before, guess the label randomly.



Our First Classification Algorithm: Memorize And Guess

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

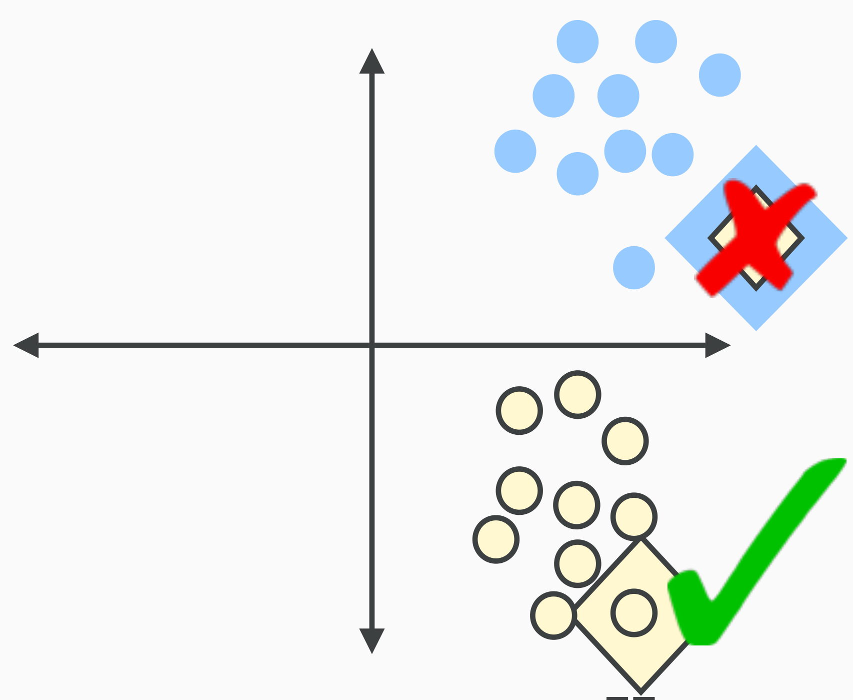
2. Create a classifier that, when **given any item seen in our training data**, **reports exactly that item's label**. If it gets an item it's never seen before, guess the label randomly.



Our First Classification Algorithm: Memorize And Guess

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.

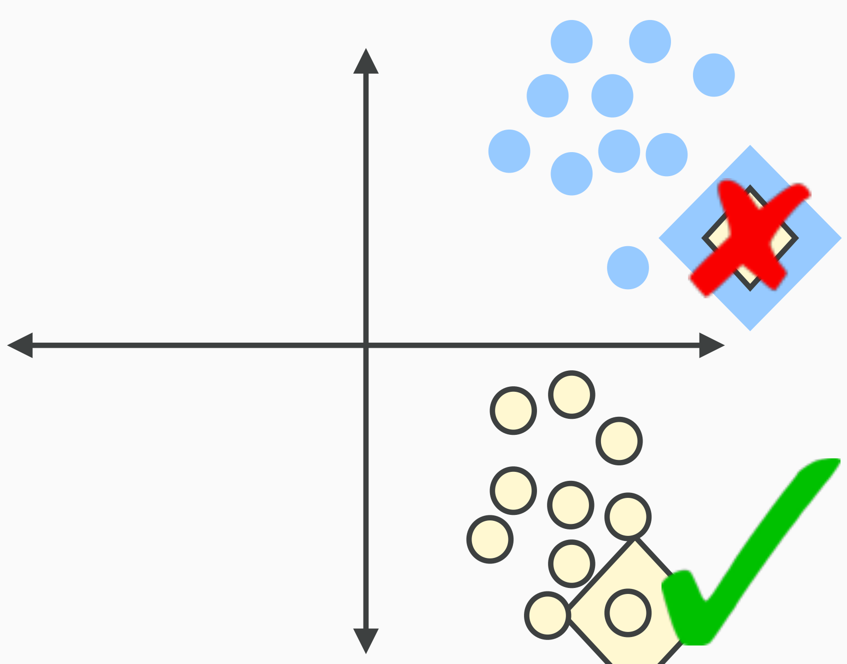


Q1: Halt?

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.



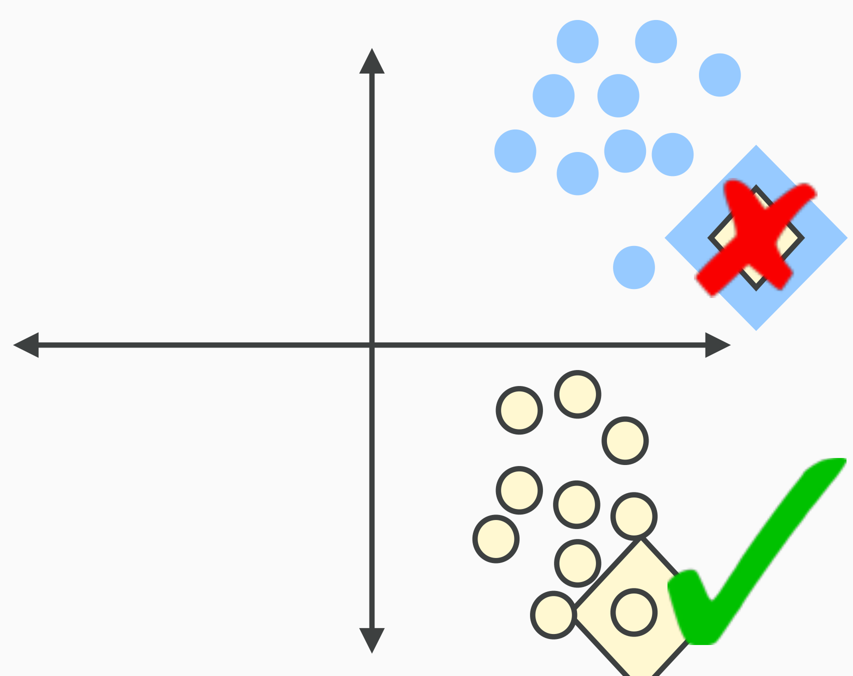
Q1: Halt?

A: Yep!

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.

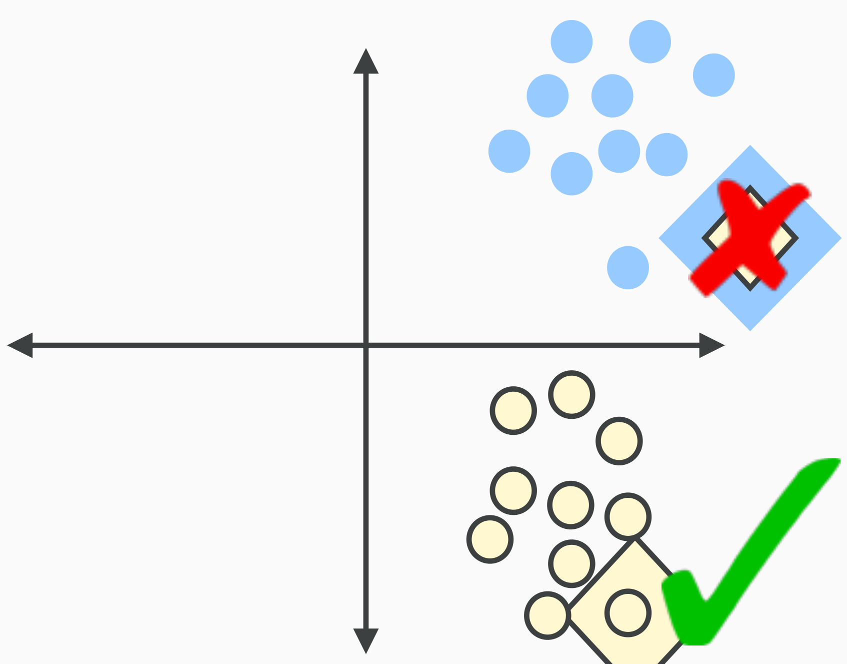


Q2: Correct?

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.



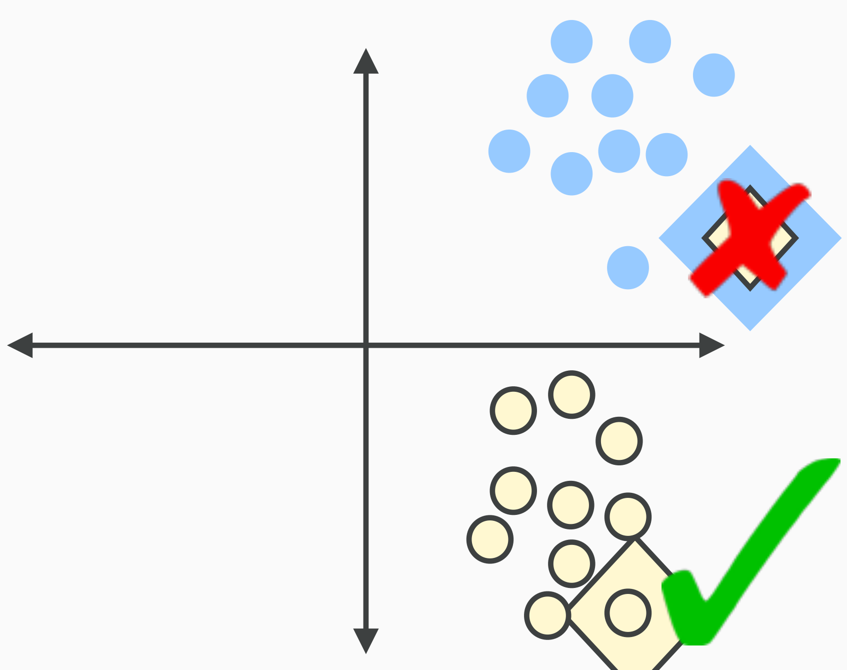
Q2: Correct?

A: We return a classifier, but it might be impossible to create a perfect classifier

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.

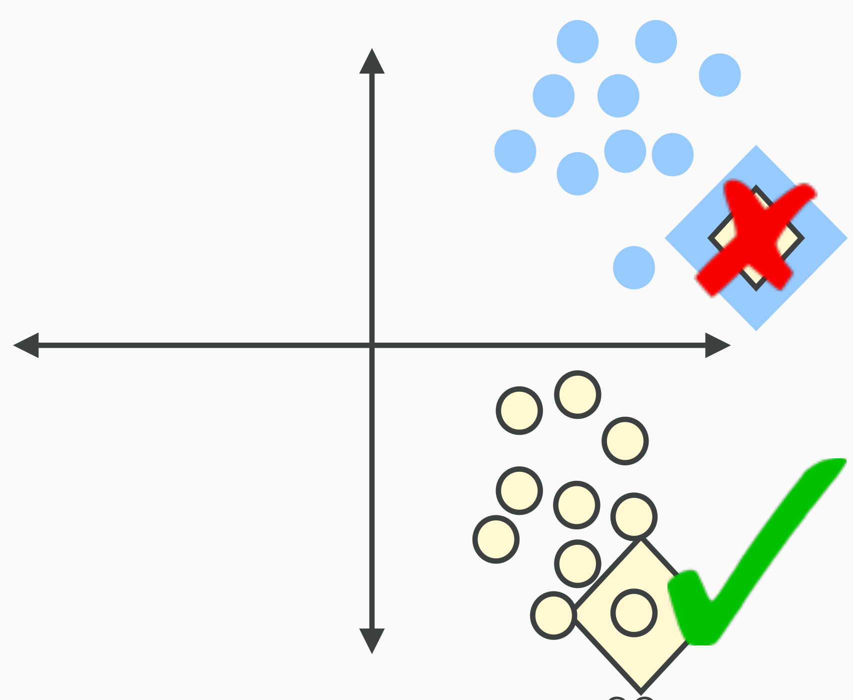


Q3: Growth Rate?

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.



Q3: Growth Rate?

A: Have to memorize N things. So let's say N.

Our First Classification Algorithm: Memorize And Guess

Side note: in learning there's a different metric that roughly translates to "how many experiences do I need to perform well?"

We'll revisit this in reinforcement learning

Q3: Growth Rate?

A: Have to memorize N things. So let's say N.

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!