Introduction to Machine Learning

Eric Grimson

MIT Department Of Electrical Engineering and Computer Science

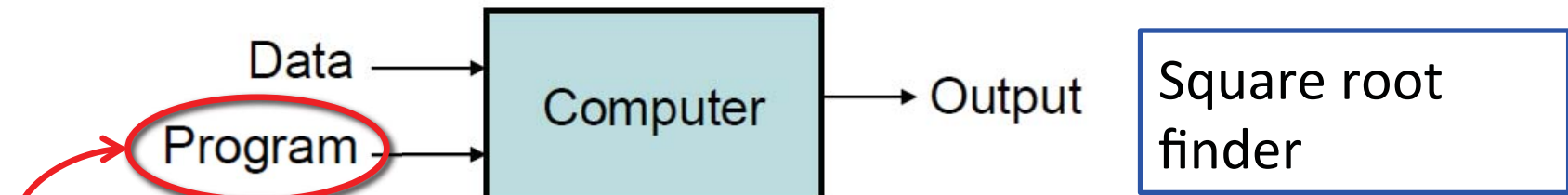# Reading assignment

- Chapter 22

The plan ahead

- Machine learning is a huge topic – with whole courses devoted to it
  - e.g., 6.008, 6.036, 6.860, 6.862, 6.867, and as central part of courses in natural language processing, computational biology, computer vision, robotics, other areas

- In 6.0002, we will
  - Provide an introduction to the basic ideas, including ways to measure distances between examples, and how to group examples based on distance to create models
  - Introduce classification methods, such as "k nearest neighbor" methods
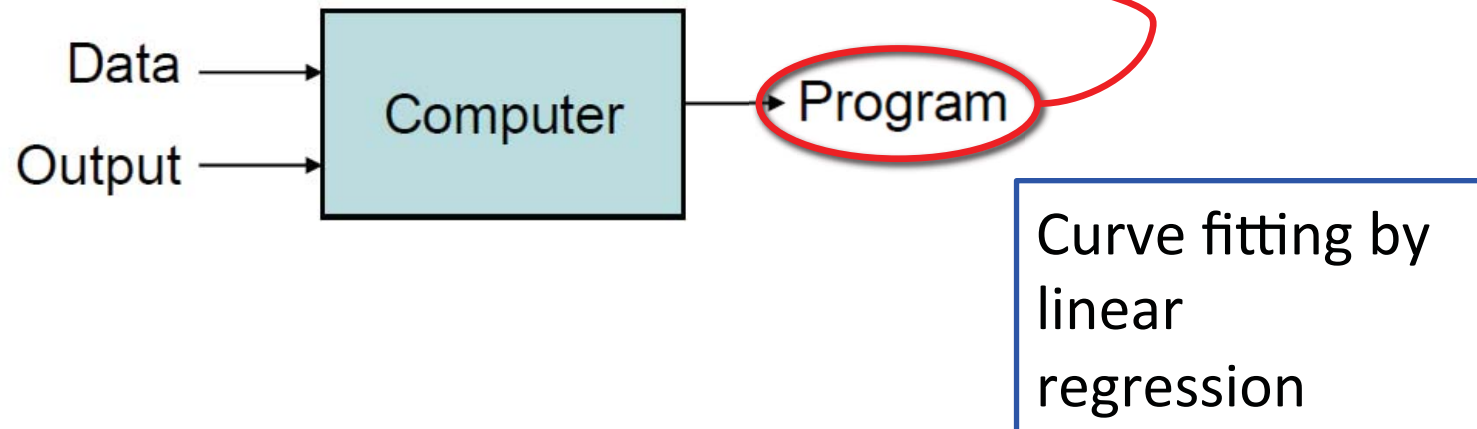 - Introduce clustering methods, such as "k-means"

What Is Machine Learning?

■ All useful programs "learn" something

■ In the first lecture of 6.0001 we looked at an algorithm for finding square roots

■ Last week we looked at using linear regression to find a model of a collection of points

■ Early definition of machine learning:

○ *"Field of study that gives computers the ability to learn without being explicitly programmed."* Arthur Samuel (1959)

○ Computer pioneer who wrote first self-learning program, which played checkers – learned from "experience"

○ Invented alpha-beta pruning – widely used in decision tree searching

What Is Machine Learning?

Traditional Programming

Data ⟶ Computer ⟶ Output

Program ⟶ Computer

Square root finder

Machine Learning

Data ⟶ Computer ⟶ Program

Output ⟶ Computer

Curve fitting by linear regression

How Are Things Learned?

- **Memorization**
  - Accumulation of individual facts
  - Limited by
    - Time to observe facts
    - Memory to store facts

Declarative knowledge

- **Generalization**
  - Deduce new facts from old facts
  - Limited by accuracy of deduction process
    - Essentially a predictive activity
    - Assumes that the past predicts the future

Imperative knowledge

- Interested in extending to programs that can infer useful information from **implicit** patterns in data

Basic Paradigm

■Observe set of examples: training data

> Spatial deviations relative to mass displacements of spring

■Infer something about process that generated that data

> Fit polynomial curve using linear regression

■Use inference to make predictions about previously unseen data: test data

> Predict displacements for other weights

Basic Paradigm

- Observe set of examples: training data

  Football players, labeled by position, with height and weight data

- Infer something about process that generated that data

  Find canonical model of position, by statistics

- Use inference to make predictions about previously unseen data: test data

  Predict position of new players

- Variations on paradigm
  ○ **Supervised**: given a set of feature/label pairs, find a rule that predicts the label associated with a previously unseen input
  ○ **Unsupervised**: given a set of feature vectors (without labels) group them into "natural clusters" (or create labels for groups)

Some Examples of Classifying and Clustering

- Here are some data on the New England Patriots
  - Name, height, weight
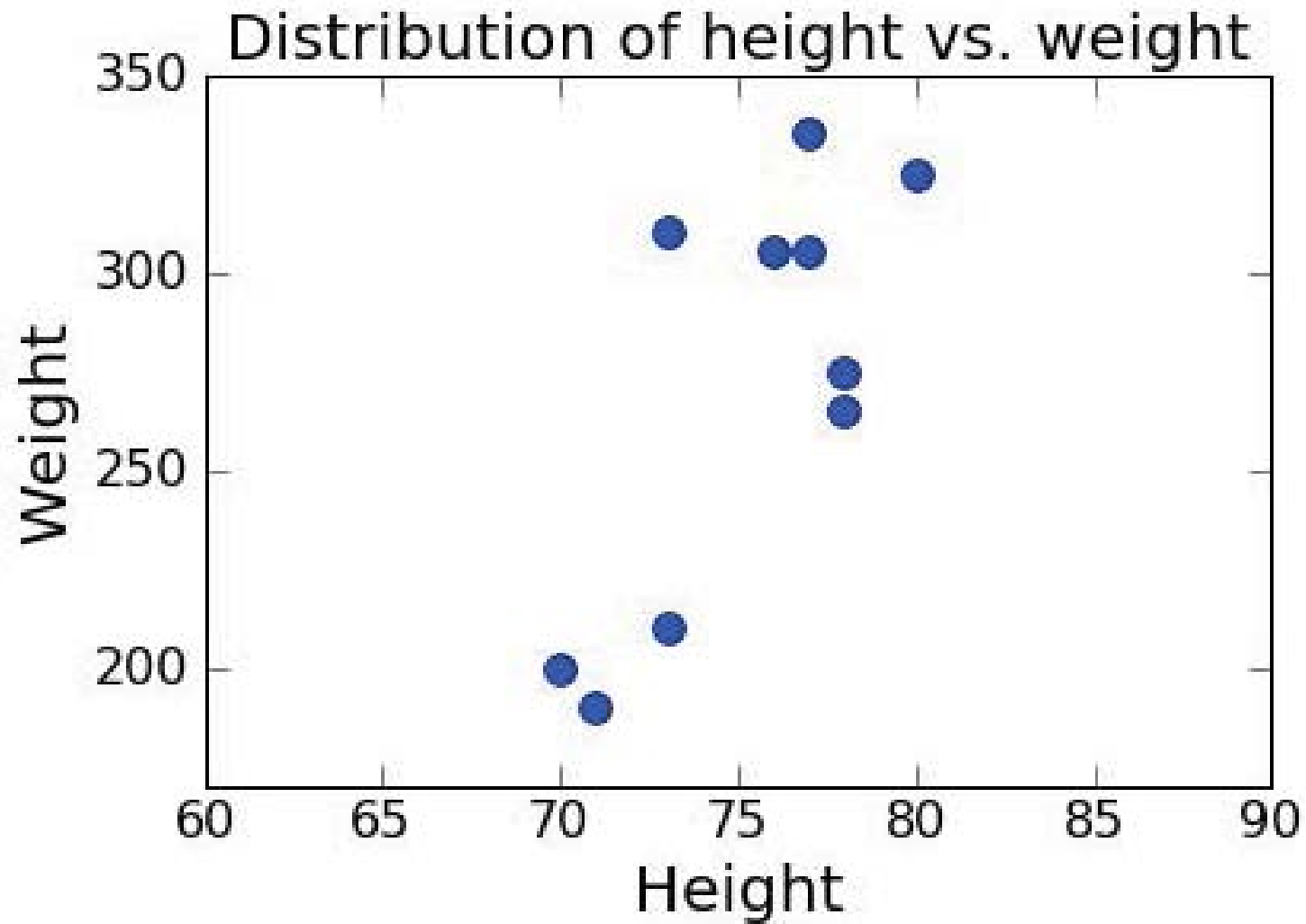 - Labeled by type of position

- Receivers:
  - edelman = ['edelman', 70, 200]
  - hogan = ['hogan', 73, 210]
  - gronkowski = ['gronkowski', 78, 265]
  - amendola = ['amendola', 71, 190]
  - bennett = ['bennett', 78, 275]

- Linemen:
  - cannon = ['cannon', 77, 335]
  - solder = ['solder', 80, 325]
  - mason = ['mason', 73, 310]
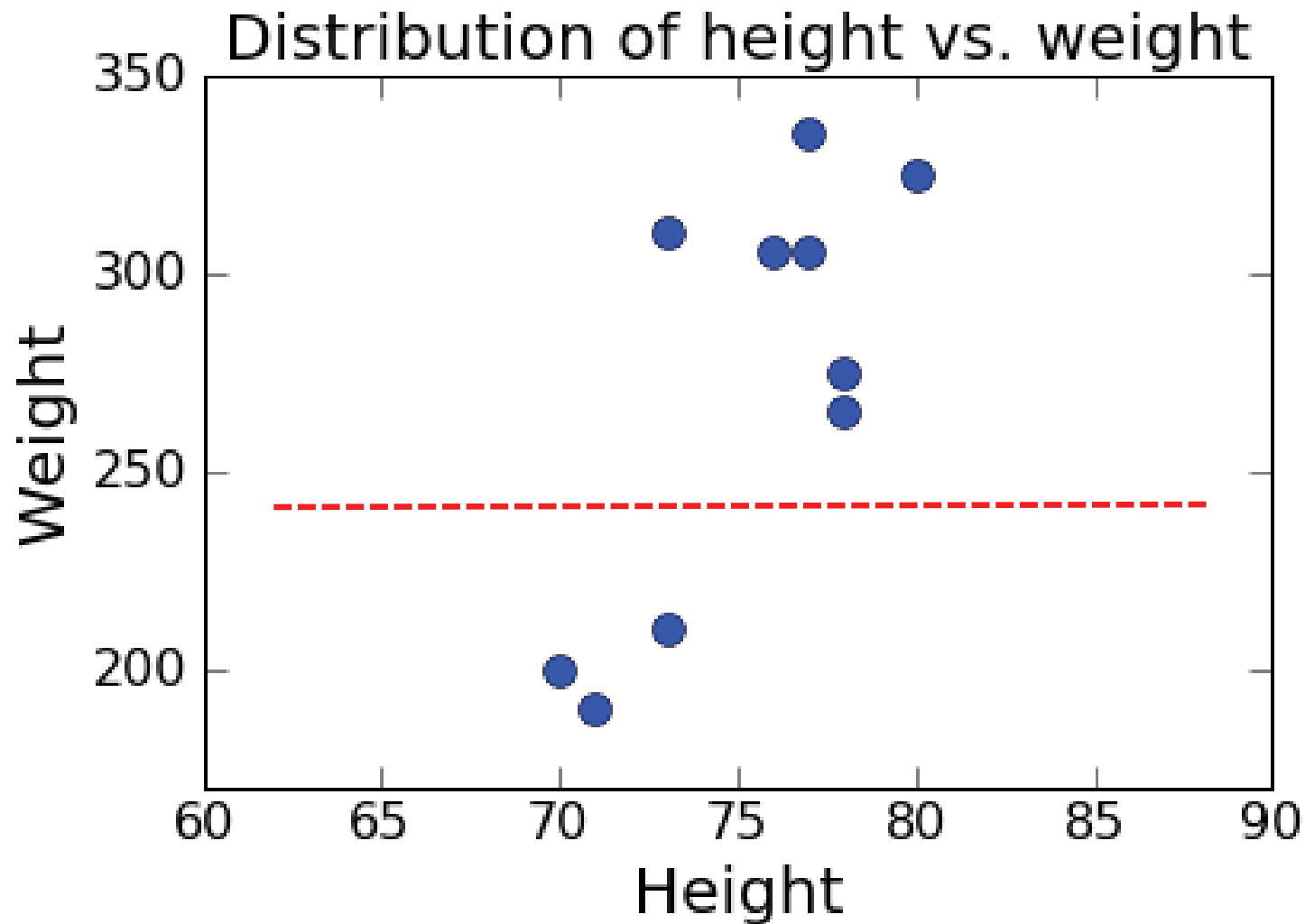 - thuney = ['thuney', 77, 305]
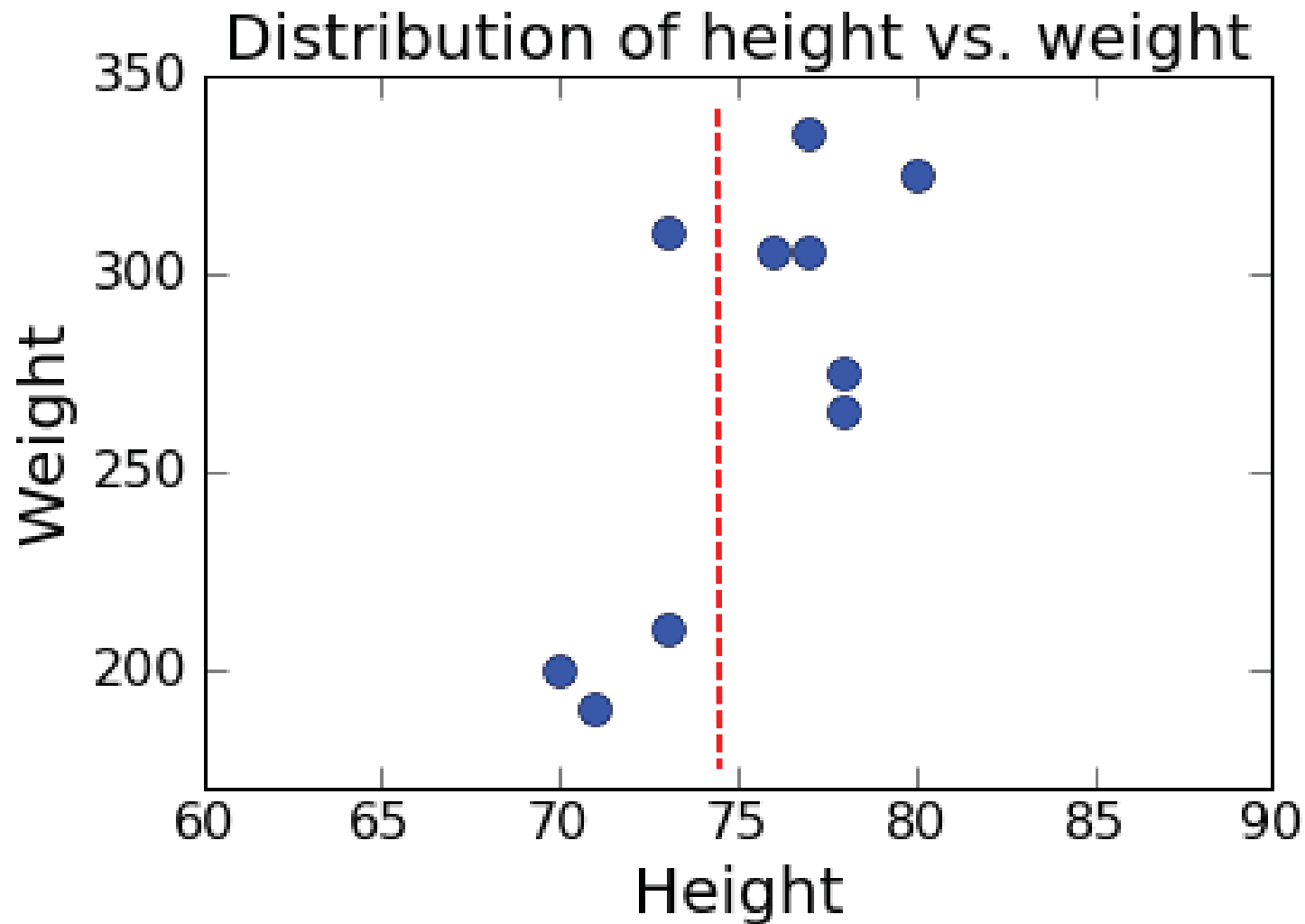 - karras = ['karras', 76, 305]

Unlabeled Data

Distribution of height vs. weight

Clustering examples into groups

- Want to decide on "similarity" of examples, with goal of separating into distinct, "natural", groups
  - Similarity is a distance measure

- Suppose we know that there are k different groups in our training data, but don't know labels (here k = 2)
  - Pick k samples (at random?) as exemplars
  - Cluster remaining samples by minimizing distance between samples in same cluster (objective function) – put sample in group with closest exemplar
  - Find median example in each cluster as new exemplar
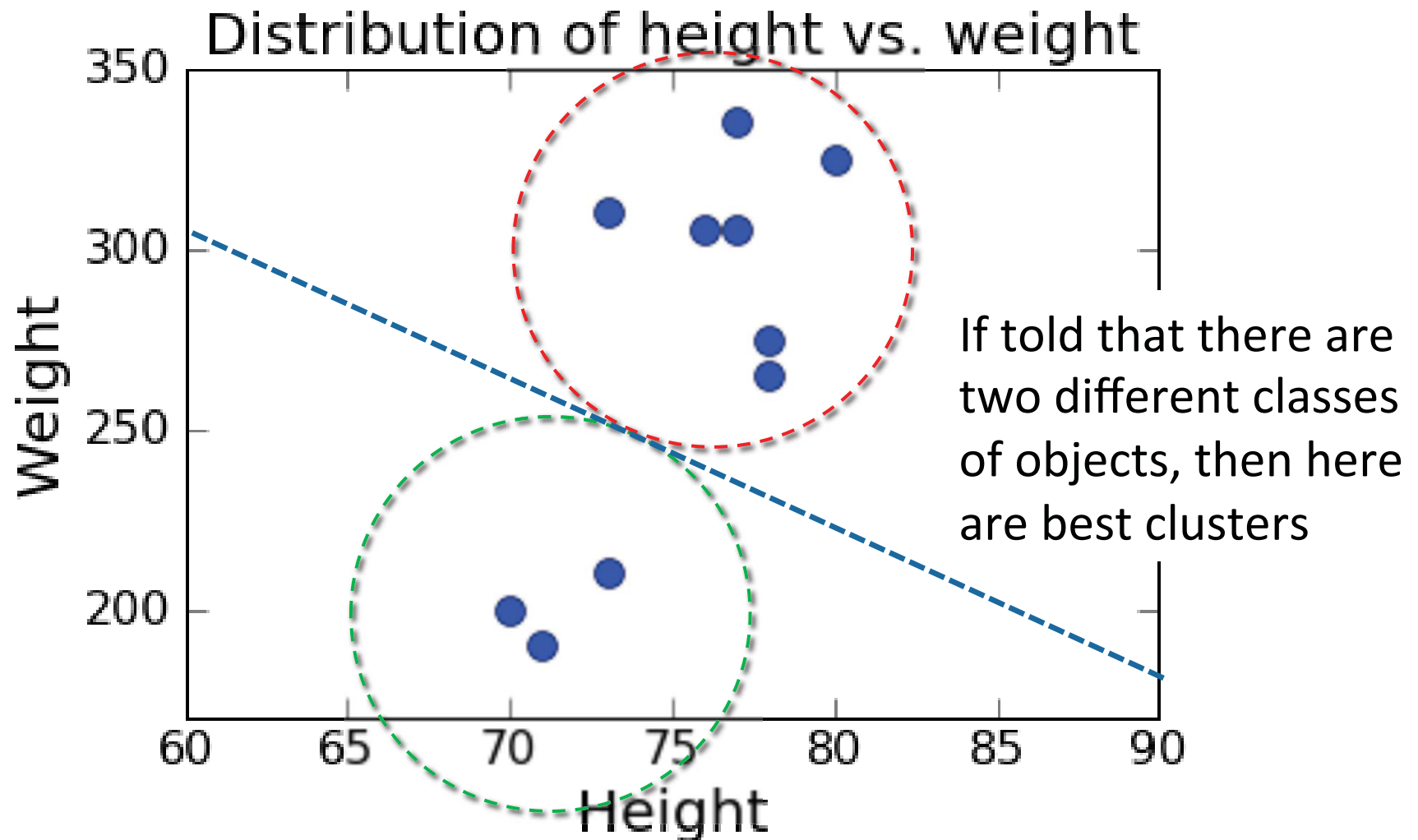 - Repeat until no change

Similarity Based on Weight

Distribution of height vs. weight
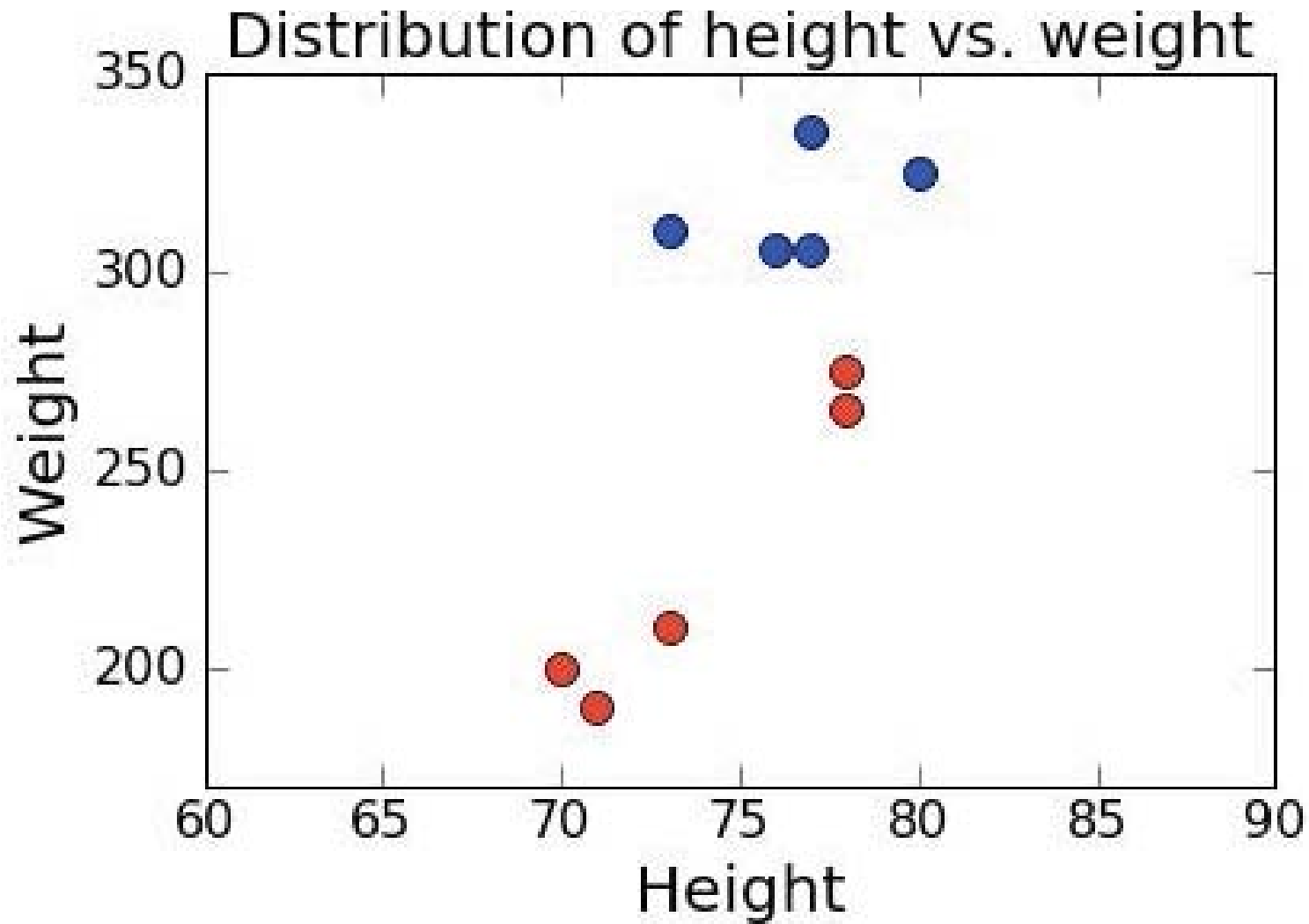
Similarity Based on Height

Distribution of height vs. weight

Cluster into Two Groups Using Both Attributes

Distribution of height vs. weight

If told that there are two different classes of objects, then here are best clusters

Suppose Data Was Labeled

Distribution of height vs. weight
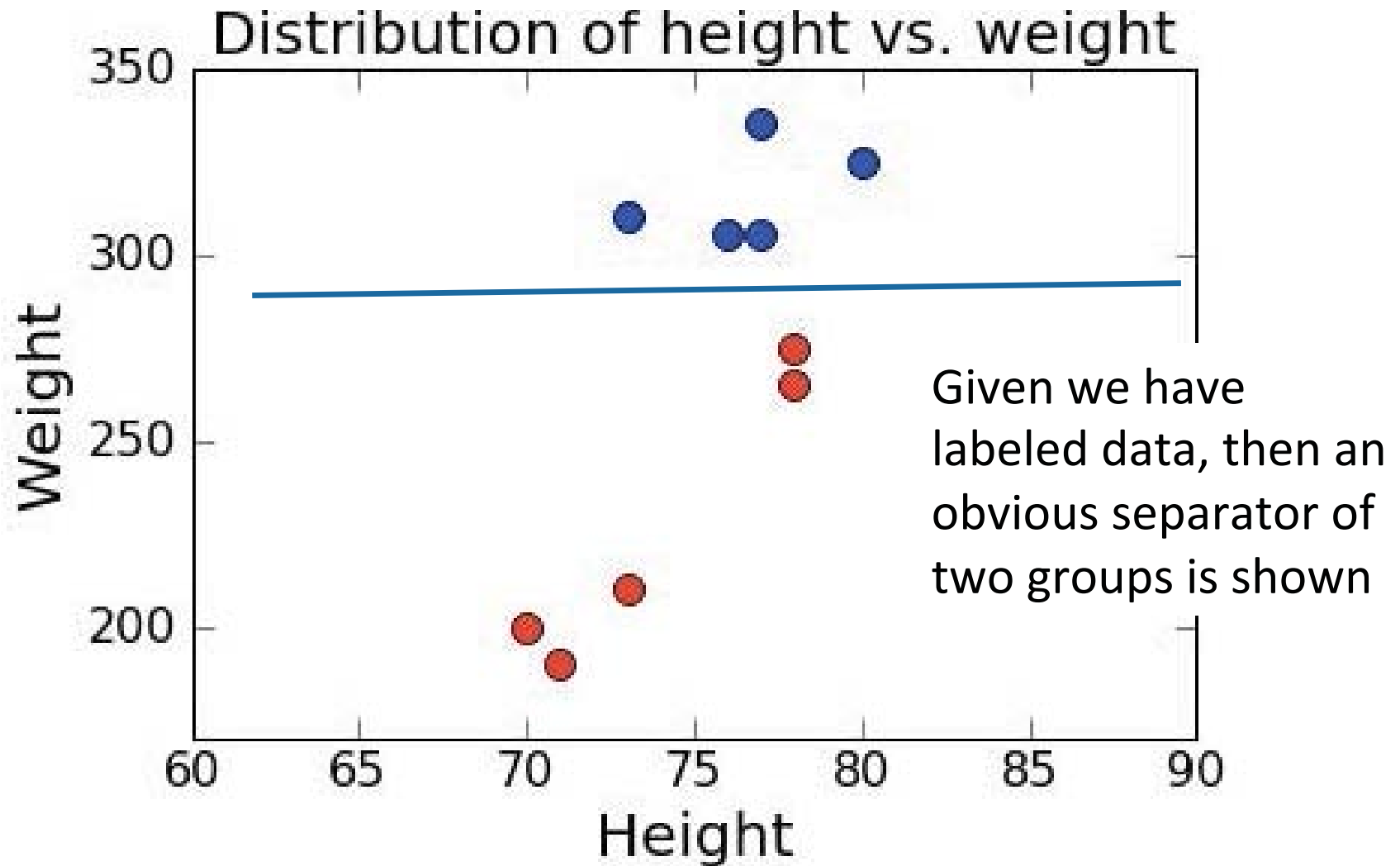
Finding Classifier Surfaces

- Given labeled groups in feature space, want to find subsurface in that space that separates the groups
  - Subject to constraints on complexity of subsurface

- In this example, have 2D space, so find line (or connected set of line segments) that best separates the two groups

- When examples well separated, this is straightforward

- When examples in labeled groups overlap, may have to trade off false positives and false negatives
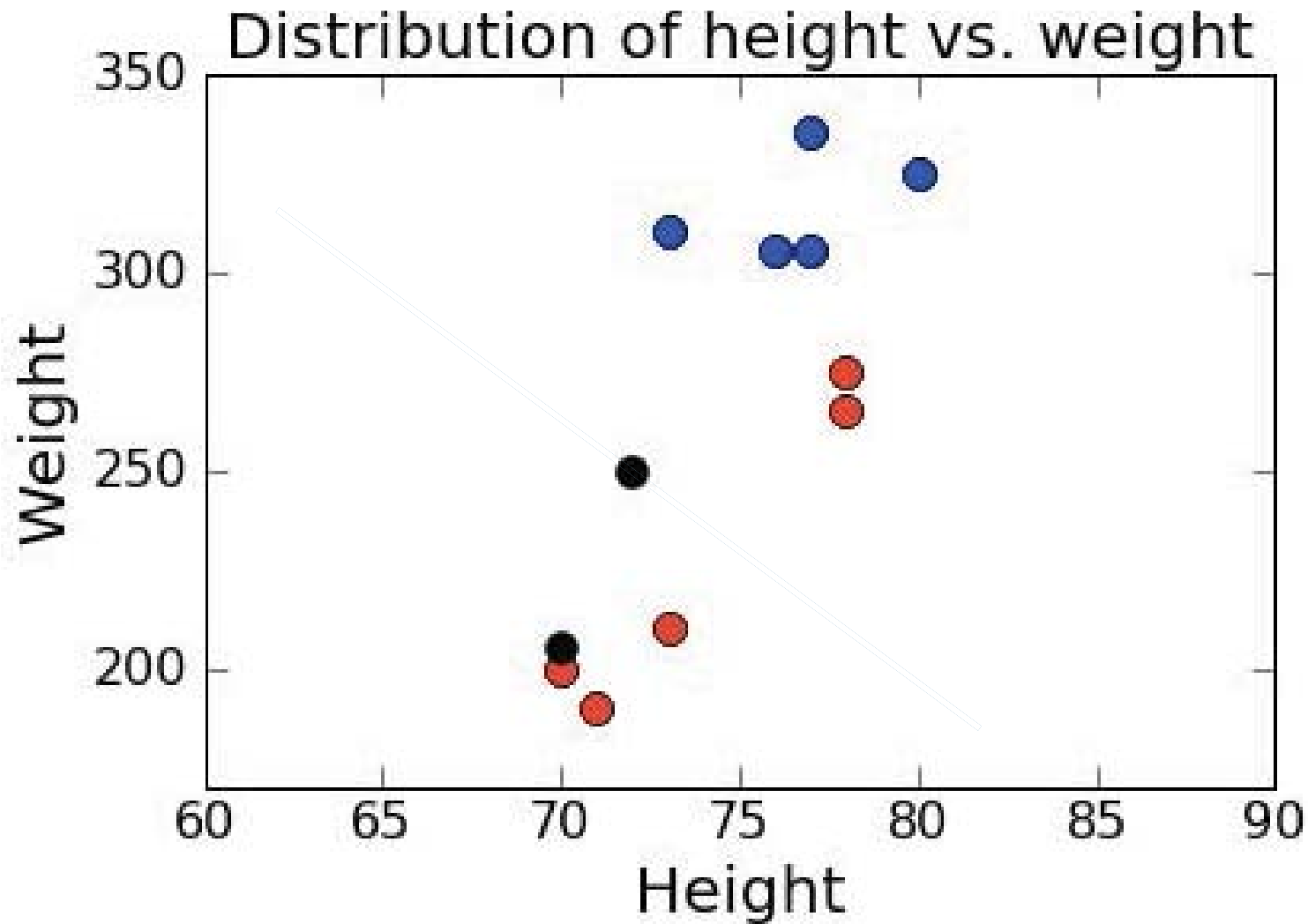
Suppose Data Was Labeled

Distribution of height vs. weight

Given we have labeled data, then an obvious separator of two groups is shown
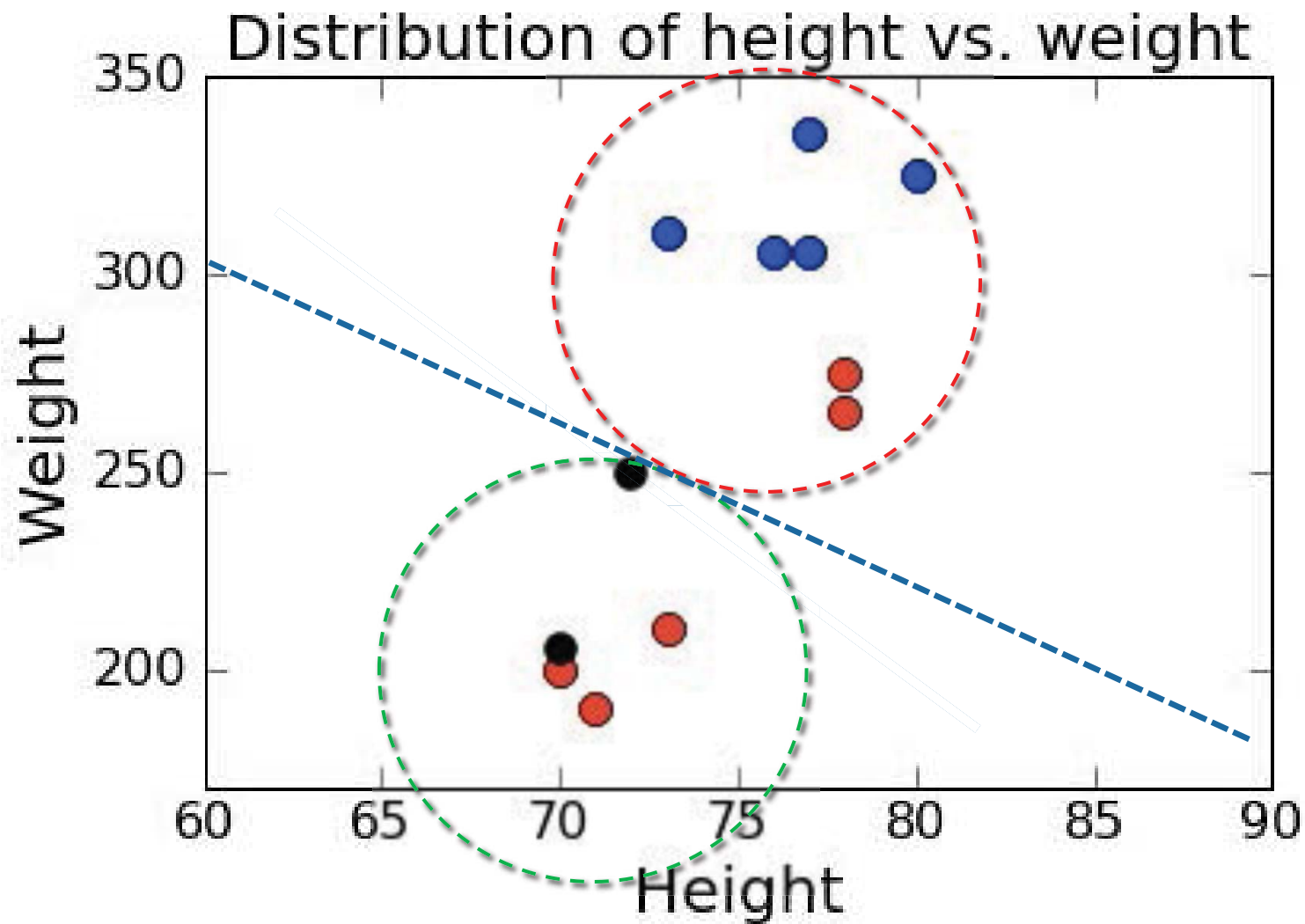
Adding Some New Data

- Suppose we have learned to separate receivers versus linemen

- Now we are given some running backs, and want to use model to decide if they are more like receivers or linemen
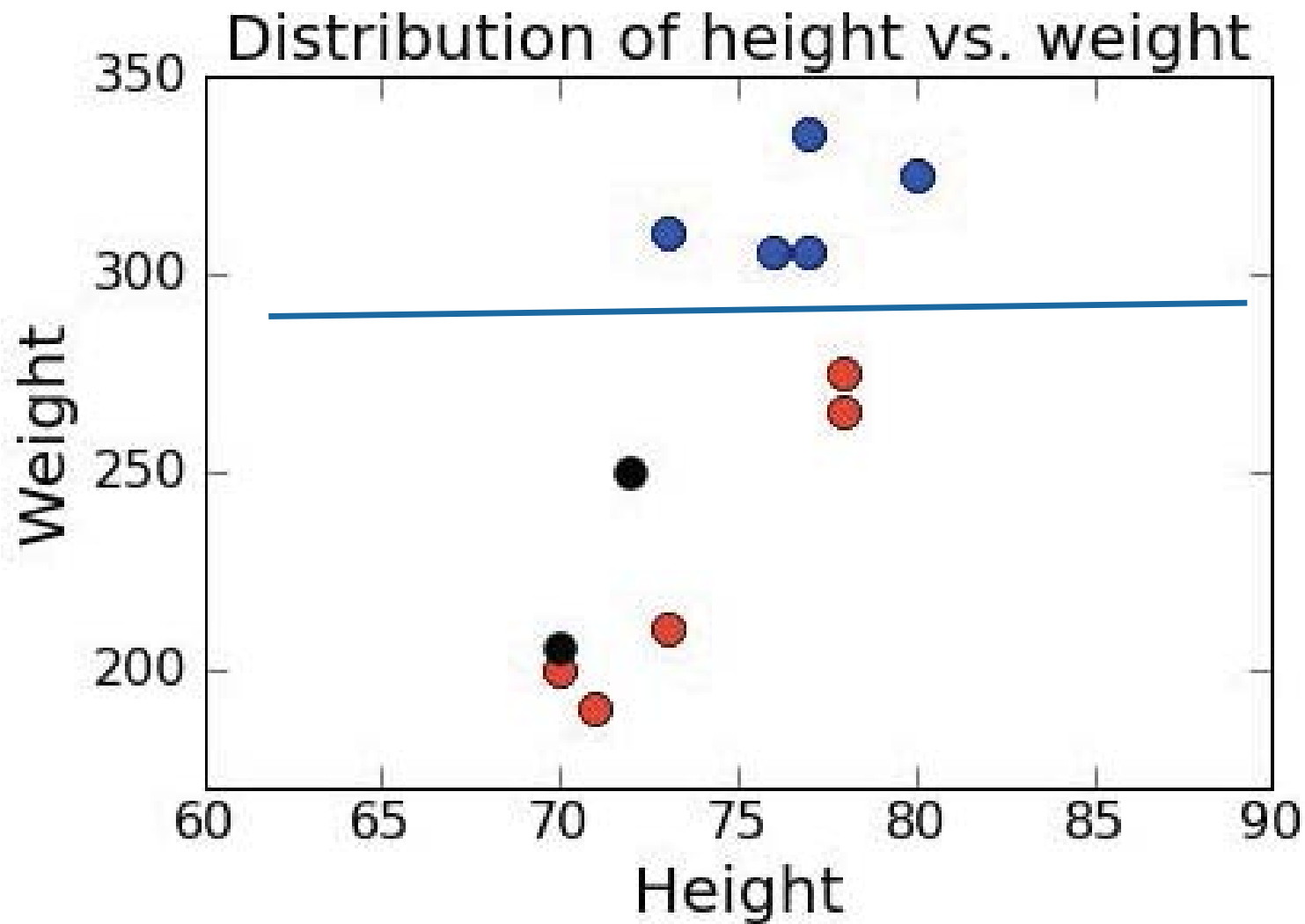 - `blount = ['blount', 72, 250]`
  - `white = ['white', 70, 205]`

Adding Some New Data

Distribution of height vs. weight

Clustering using Unlabeled Data

Distribution of height vs. weight

Classified using Labeled Data

Machine Learning Methods

- We will see some examples of machine learning methods:

- Learn models based on unlabeled data, by clustering training data into groups of nearby points
  - Resulting clusters can assign labels to new data

- Learn models that separate labeled groups of similar data from other groups
  - May not be possible to perfectly separate groups, without "over fitting"
  - But can make decisions with respect to trading off "false positives" versus "false negatives"
  - Resulting classifiers can assign labels to new data

All ML Methods Require:

- Choosing training data and evaluation method

- Representation of the features

- Distance metric for feature vectors

  Today

- Objective function and constraints

- Optimization method for learning the model

  Next week

Feature Representation

- Features never fully describe the situation
  - "All models are wrong, but some are useful." – George Box

- Feature engineering
  - Represent examples by feature vectors that will facilitate generalization
  - Suppose I want to use 100 examples from past to predict, at the start of the subject, which students will get an A in 6.0002
  - Some features surely helpful, e.g., GPA, prior programming experience (not a perfect predictor)
  - Others might cause me to overfit, e.g., birth month, eye color

- Want to maximize ratio of useful input to irrelevant input
  - Signal-to-Noise Ratio (SNR)

An Example

<div align="center">Features</div>   **Label**

Name	Egg-laying	Scales	Poisonous	Cold-blooded	# legs	Reptile
Cobra	True	True	True	True	0	Yes

Initial model:

- Not enough information to generalize

An Example

	Features					**Label**
Name	**Egg-laying**	**Scales**	**Poisonous**	**Cold-blooded**	**# legs**	**Reptile**
Cobra	True	True	True	True	0	Yes
Rattlesnake	True	True	True	True	0	Yes

Initial model:
- Egg laying
- Has scales
- Is poisonous
- Cold blooded
- No legs

An Example

	Features					**Label**
Name	**Egg-laying**	**Scales**	**Poisonous**	**Cold-blooded**	**# legs**	**Reptile**
Cobra	True	True	True	True	0	Yes
Rattlesnake	True	True	True	True	0	Yes
Boa constrictor	False	True	False	True	0	Yes

Current model:
- Has scales
- Cold blooded
- No legs

Boa doesn't fit model, but is labeled as reptile.
Need to refine model

An Example

		Features				**Label**
Name	**Egg-laying**	**Scales**	**Poisonous**	**Cold-blooded**	**# legs**	**Reptile**
Cobra	True	True	True	True	0	Yes
Rattlesnake	True	True	True	True	0	Yes
Boa constrictor	False	True	False	True	0	Yes
Chicken	True	True	False	False	2	No

Current model:
- Has scales
- Cold blooded
- No legs

An Example

	Features					Label
Name	**Egg-laying**	**Scales**	**Poisonous**	**Cold-blooded**	**# legs**	**Reptile**
Cobra	True	True	True	True	0	Yes
Rattlesnake	True	True	True	True	0	Yes
Boa constrictor	False	True	False	True	0	Yes
Chicken	True	True	False	False	2	No
Alligator	True	True	False	True	4	Yes

Current model:
- Has scales
- Cold blooded
- Has 0 or 4 legs

Alligator doesn't fit model, but is labeled as reptile.
Need to refine model

An Example

		Features				Label
Name	**Egg-laying**	**Scales**	**Poisonous**	**Cold-blooded**	**# legs**	**Reptile**
Cobra	True	True	True	True	0	Yes
Rattlesnake	True	True	True	True	0	Yes
Boa constrictor	False	True	False	True	0	Yes
Chicken	True	True	False	False	2	No
Alligator	True	True	False	True	4	Yes
Dart frog	True	False	True	False	4	No

Current model:
- Has scales
- Cold blooded
- Has 0 or 4 legs

An Example

	Features					Label
Name	**Egg-laying**	**Scales**	**Poisonous**	**Cold-blooded**	**# legs**	**Reptile**
Cobra	True	True	True	True	0	Yes
Rattlesnake	True	True	True	True	0	Yes
Boa constrictor	False	True	False	True	0	Yes
Chicken	True	True	False	False	2	No
Alligator	True	True	False	True	4	Yes
Dart frog	True	False	True	False	4	No
Salmon	True	True	False	True	0	No
Python	True	True	False	True	0	Yes

Current model:
- Has scales
- Cold blooded
- Has 0 or 4 legs

No (easy) way to add to rule that will correctly classify salmon and python (since identical feature values)

An Example

| | | **Features** | | | | **Label** |
Name	**Egg-laying**	**Scales**	**Poisonous**	**Cold-blooded**	**# legs**	**Reptile**
Cobra	True	True	True	True	0	Yes
Rattlesnake	True	True	True	True	0	Yes
Boa constrictor	False	True	False	True	0	Yes
Chicken	True	True	False	False	2	No
Alligator	True	True	False	True	4	Yes
Dart frog	True	False	True	False	4	No
Salmon	True	True	False	True	0	No
Python	True	True	False	True	0	Yes

Good model:
- Has scales
- Cold blooded

Not perfect, but no false negatives (anything classified as not reptile is correctly labeled); some false positives (may incorrectly label some animals as reptile)

Need to Measure Distances between Features

- Feature engineering:
  - Deciding which features to include and which are merely adding noise to classifier
  - Defining how to measure distances between training examples (and ultimately between classifiers and new instances)
  - Deciding how to weight relative importance of different dimensions of feature vector, which impacts definitionof distance

Measuring Distance Between Animals

- We can think of our animal examples as consisting of four binary features and one integer feature

- One way to learn to separate reptiles from non-reptiles is to measure the distance between pairs of examples, and use that:

  - To cluster nearby examples into a common class (unlabeled data), or

  - To find a classifier surface in space of examples that optimally separates different (labeled) collections of examples from other collections

```
rattlesnake = [1,1,1,1,0]
boa constrictor = [0,1,0,1,0]
dart Frog = [1,0,1,0,4]
```

Can convert examples into feature vectors

Minkowski Metric

$$dist(X1, X2, p) = \left(\sum_{k=1}^{len} abs(X1_k - X2_k)^p \right)^{1/p}$$

p = 1: Manhattan Distance
p = 2: Euclidean Distance

Need to measure distances between feature vectors

Typically use Euclidean metric; Manhattan may be appropriate if different dimensions are not comparable
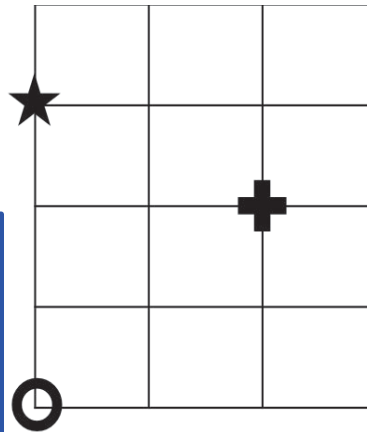
Is circle closer to star or cross?
- Euclidean distance
  - Cross – 2.8
  - Star – 3
- Manhattan Distance
  - Cross – 4
  - Star - 3

Euclidean Distance Between Animals

```
rattlesnake = [1,1,1,1,0]
boa constrictor = [0,1,0,1,0]
dartFrog = [1,0,1,0,4]
```

Euclidean Distance Between Animals

```
rattlesnake = [1,1,1,1,0]
boa constrictor = [0,1,0,1,0]
dartFrog = [1,0,1,0,4]
```

	rattlesnake	boa constrictor	dart frog
rattlesnake	--	1.414	4.243
boa constrictor	1.414	--	4.472
dart frog	4.243	4.472	--

Using Euclidean distance, rattlesnake and boa constrictor are much closer to each other, than they are to the dart frog

Add an Alligator

- `alligator = Animal('alligator', [1,1,0,1,4])`
- `animals.append(alligator)`
- `compareAnimals(animals, 3)`

Add an Alligator

```
alligator = Animal('alligator', [1,1,0,1,4])
animals.append(alligator)
compareAnimals(animals, 3)
```

	rattlesnake	boa constrictor	dart frog	alligator
rattlesnake	--	1.414	4.243	4.123
boa constrictor	1.414	--	4.472	4.123
dart frog	4.243	4.472	--	1.732
alligator	4.123	4.123	1.732	--

Alligator is closer to dart frog than to snakes – why?
- Alligator differs from frog in 3 features, from boa in only 2 features
- But scale on "legs" is from 0 to 4, on other features is 0 to 1
- "legs" dimension is disproportionately large

Using Binary Features

```
rattlesnake = [1,1,1,1,0]
boa constrictor = [0,1,0,1,0]
dartFrog = [1,0,1,0,1]
Alligator = [1,1,0,1,1]
```

	rattlesnake	boa constrictor	dart frog	alligator
rattlesnake	--	1.414	1.732	1.414
boa constrictor	1.414	--	2.236	1.414
dart frog	1.732	2.236	--	1.732
alligator	1.414	1.414	1.732	--

Now alligator is closer to snakes than it is to dart frog – makes more sense

Feature Engineering Matters

Supervised versus Unsupervised Learning

- In the next few lectures, we will see examples of learning algorithms:

- When given unlabeled data, try to find clusters of examples near each other
  - Use centroids of clusters as definition of each learned class
  - New data assigned to closest cluster

- When given labeled data, learn mathematical surface that "best" separates labeled examples, subject to constraints on complexity of surface (don't over fit)
  - New data assigned to class based on portion of feature space carved out by classifier surface in which it lies

Issues of Concern When Learning Models

Learned models will depend on:

- Distance metric between examples

- Choice of feature vectors

- Constraints on complexity of model
  - Specified number of clusters
  - Complexity of separating surface
  - Want to avoid over fitting problem (each example is its own cluster, or a complex separating surface)

Clustering approaches

- Suppose we know that there are k different groups in our training data, but don't know labels
  - Pick k samples (at random?) as exemplars
  - Cluster remaining samples by minimizing distance between samples in same cluster (objective function) – put sample in group with closest exemplar
  - Find median example in each cluster as new exemplar
  - Repeat until no change

- Issues:
  - How do we decide on the best number of clusters?
  - How do we select the best features, the best distance metric?

Clustering using Unlabeled Data

Distribution of height vs. weight
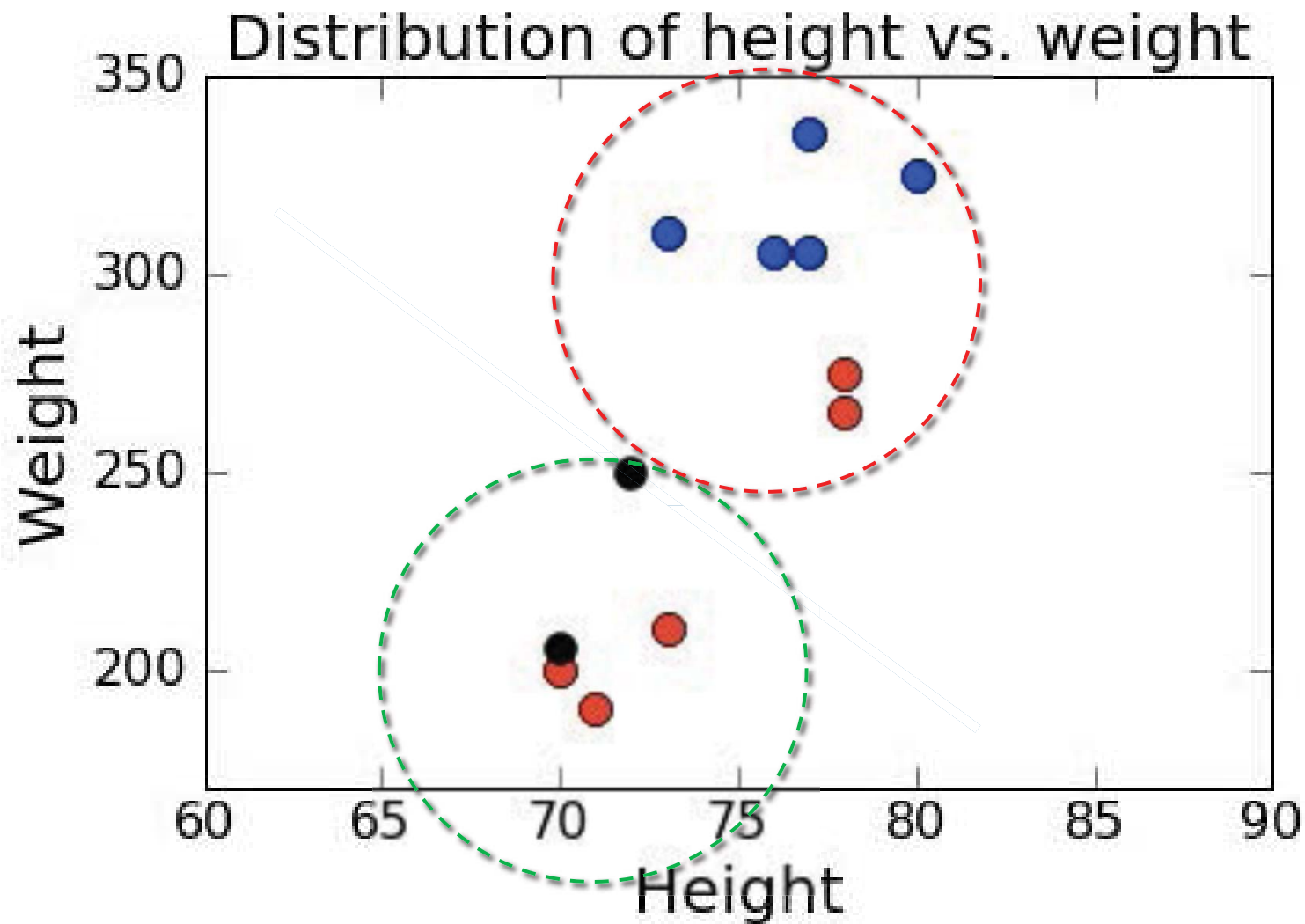
Fitting Three Clusters Unsupervised
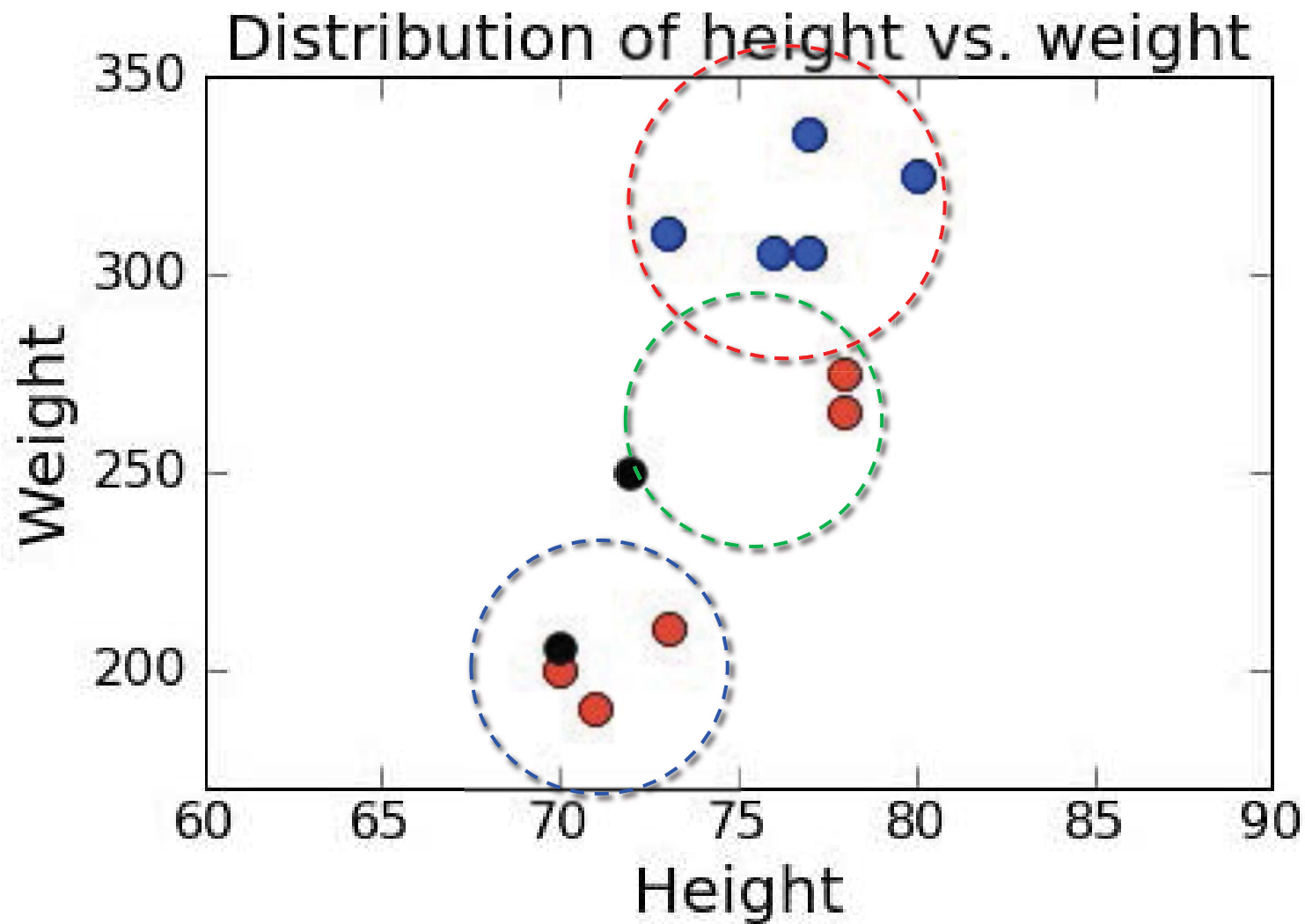
Classification approaches

- Want to find boundaries in feature space that separate different classes of labeled examples
  - Look for simple surface (e.g. best line or plane) that separates classes
  - Look for more complex surfaces (subject to constraints) that separate classes
  - Use voting schemes
    - Find k nearest training examples, use majority vote to select label

- Issues:
  - How do we avoid over-fitting to data?
  - How do we measure performance?
  - How do we select best features?

Classification

- Attempt to minimize error on training data
  - Similar to fitting a curve to data

- Evaluate on training data

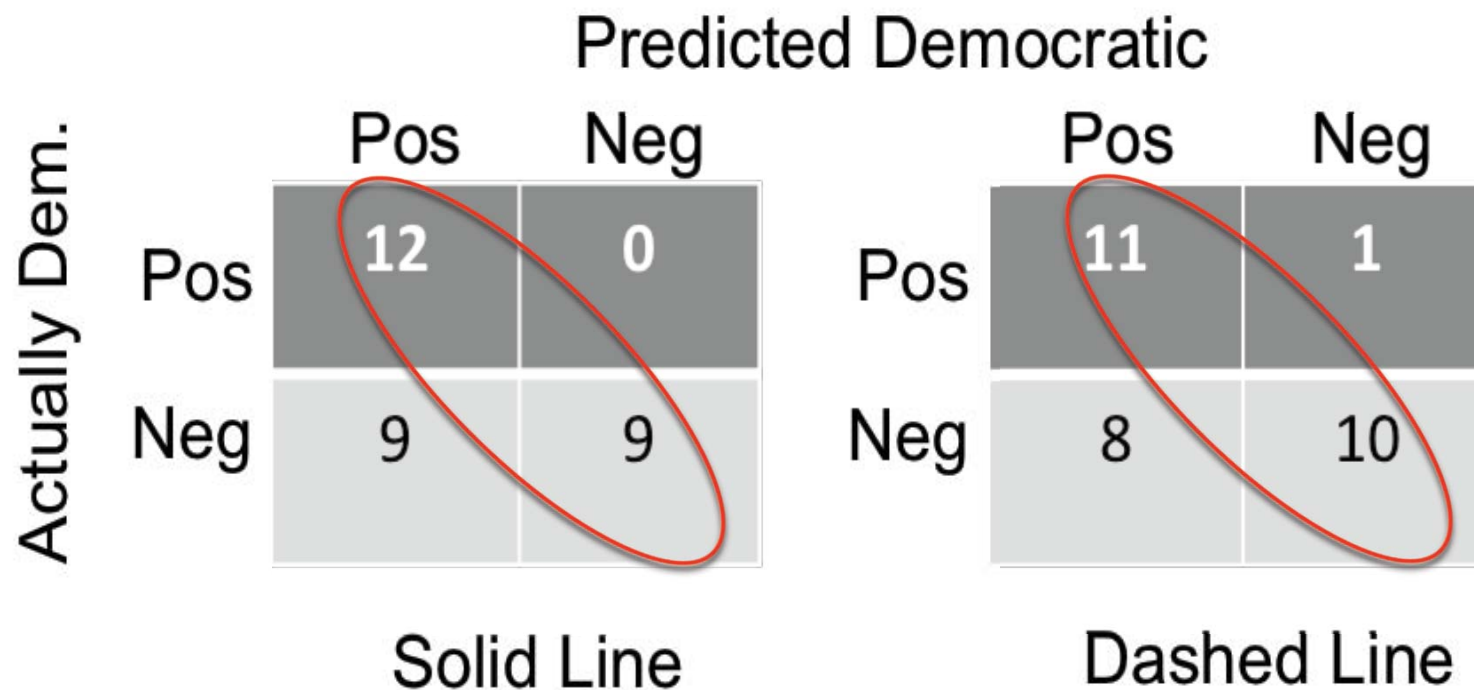Voter preference, by age and distance from Boston

Voter Preference
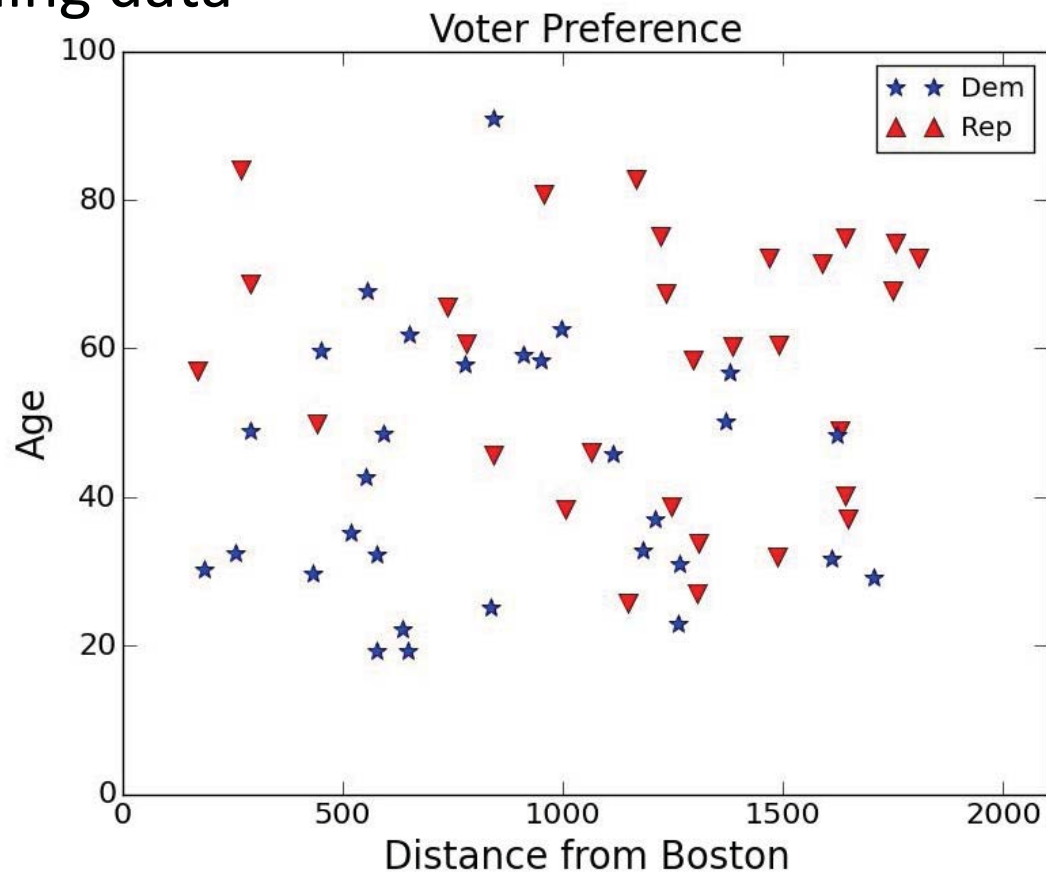
Randomly Divide Data into Training and Test Set
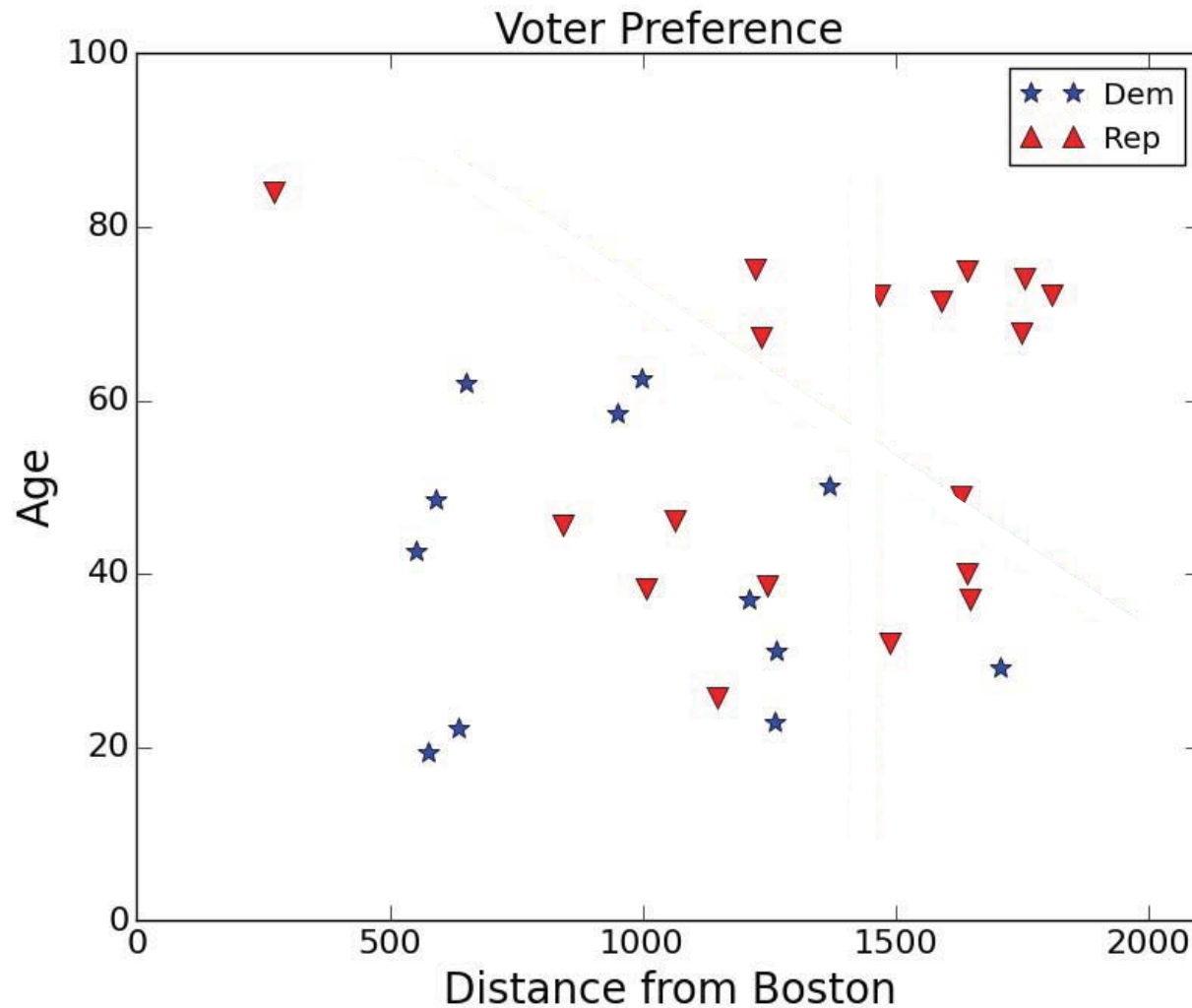
Two Possible Models for a Training Set
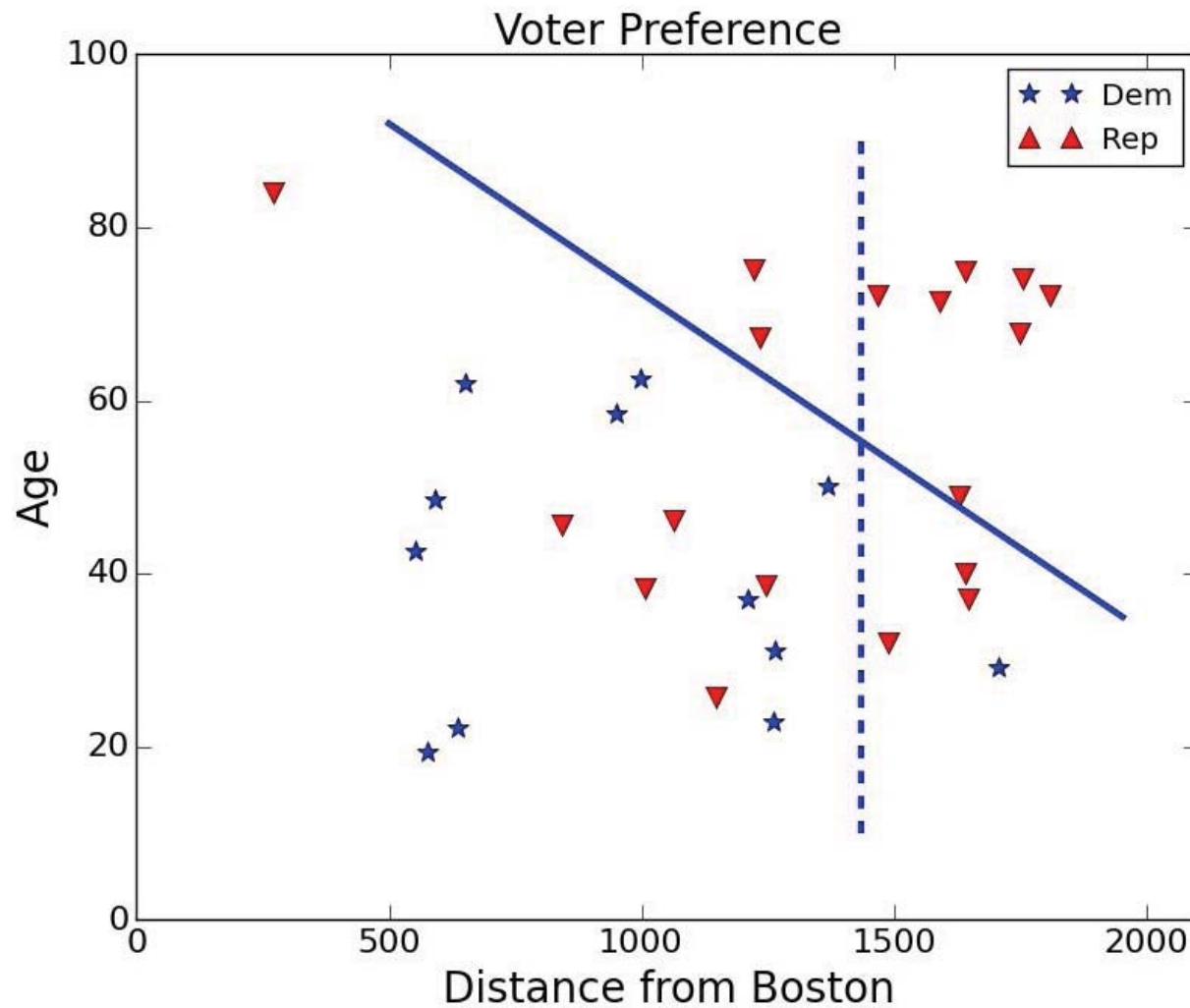
Confusion Matrices (Training Error)

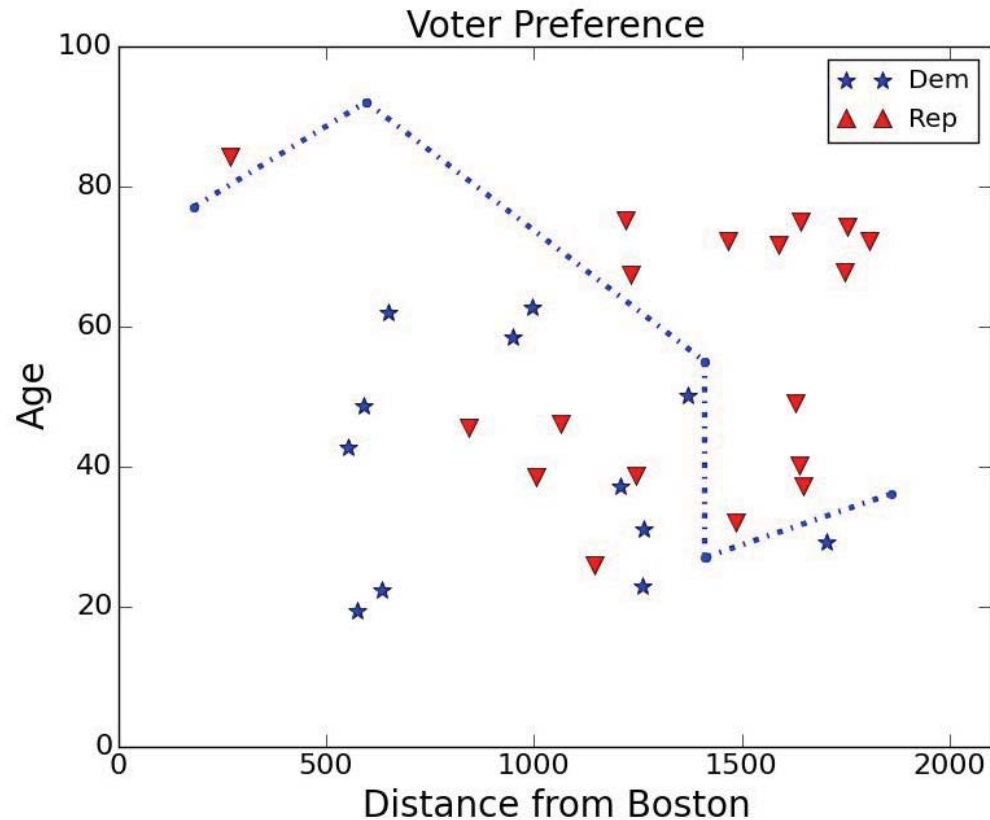# Training Accuracy of Models

$$accuracy = \frac{true\ positive + true\ negative}{true\ positive + true\ negative + false\ positive + false\ negative}$$

- 0.7 for both models
  - Which is better?

- Can we find a model with less training error?

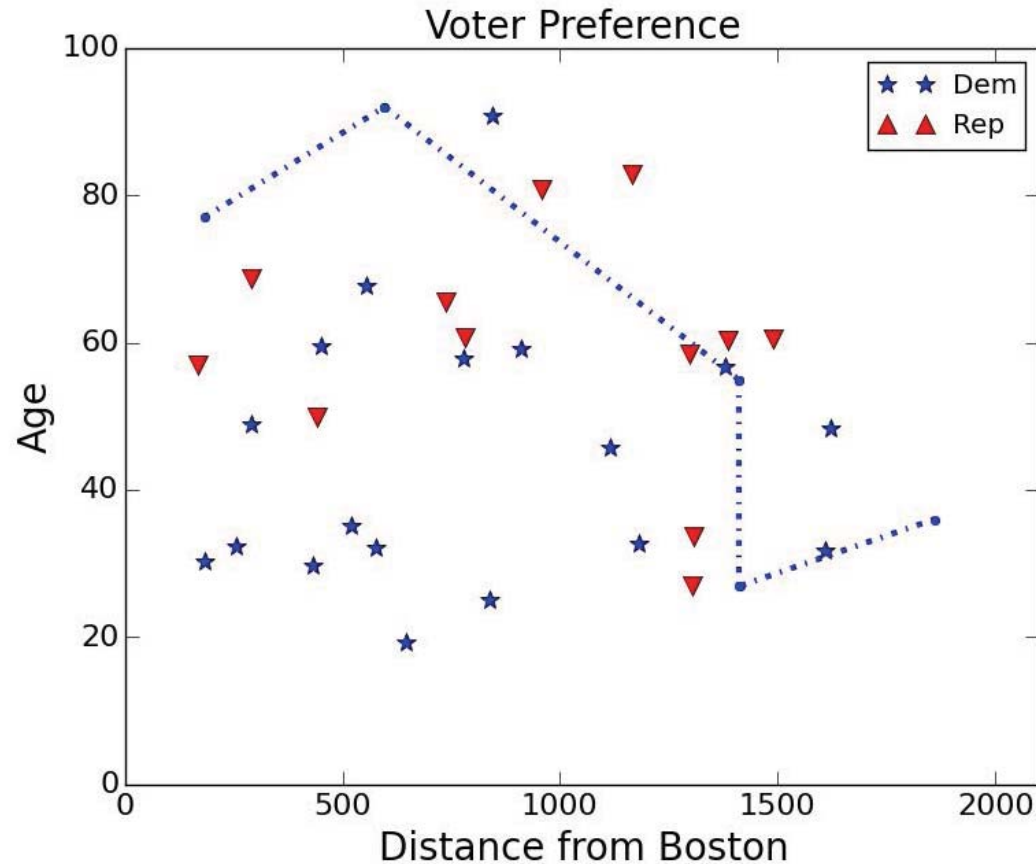# A More Complex Model

TP = 12, FP = 5, TN = 13, FN = 0
Accuracy = 25/30 = 0.833

Applying Model to Test Data

TP = 14, FP = 4, TN = 4, FN = 8
Accuracy = 18/30 = 0.6

Other Statistical Measures

$$positive\ predictive\ value = \frac{true\ positive}{true\ positive + false\ positive}$$

- Solid line model: .57

- Dashed line model: .58

- Complex model, training: .71

- Complex model, testing: .78

- You will also see "sensitivity" versus "specificity"

$$sensitivity = \frac{true\ positive}{true\ positive + false\ negative}$$

$$specificity = \frac{true\ negative}{true\ negative + false\ positive}$$

Percentage correctly found	
Percentage correctly rejected	

Summary

- Machine learning methods provide a way of building models of processes from data sets
  - Supervised learning uses labeled data, and creates classifiers that optimally separate data into known classes
  - Unsupervised learning tries to infer latent variables by clustering training examples into nearby groups

- Choice of features influences results

- Choice of distance measurement between examples influences results

- We will see some examples of clustering methods, such as k-means

- We will see some examples of classifiers, such as k nearest neighbor methods

6.0002 Introduction to Computational Thinking and Data Science

Fall 2016