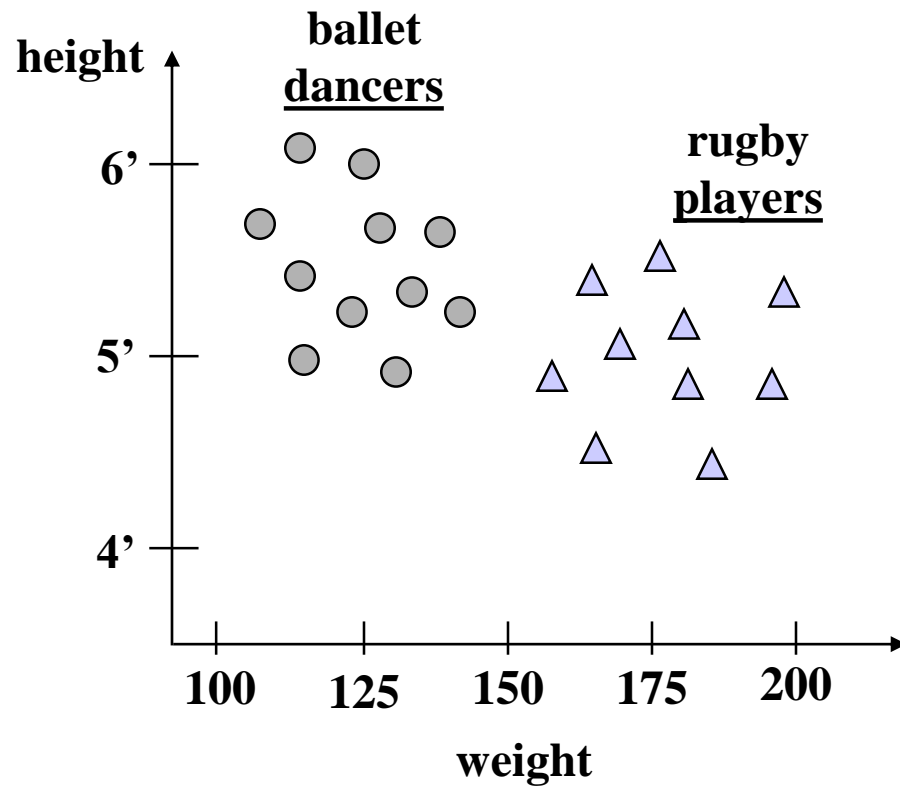


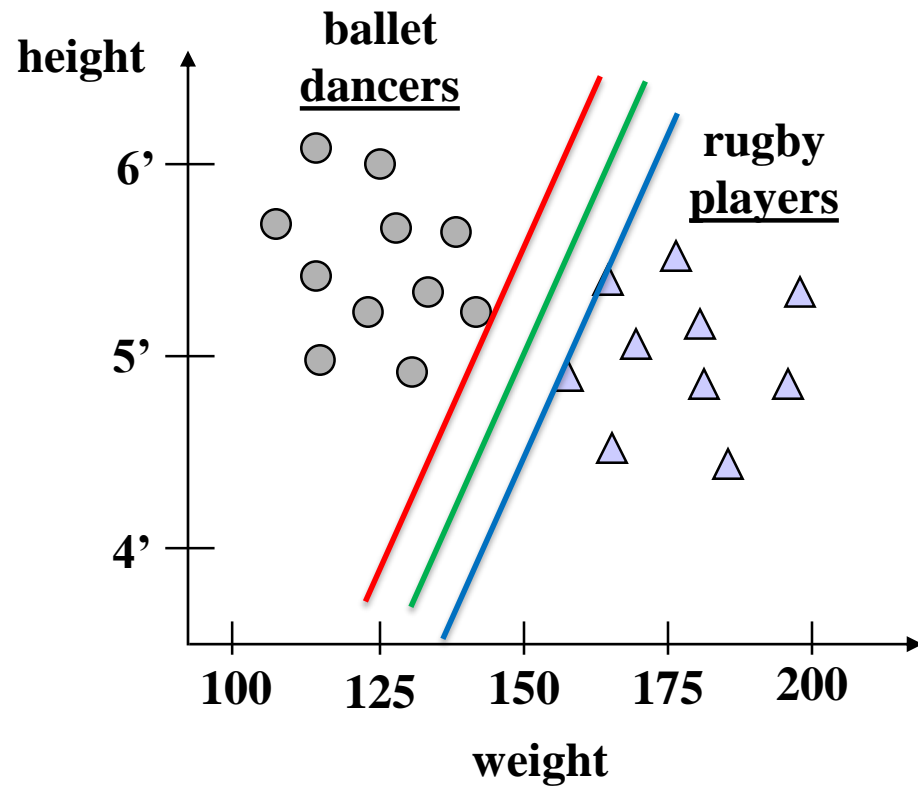
Support Vector Machines

Example training data

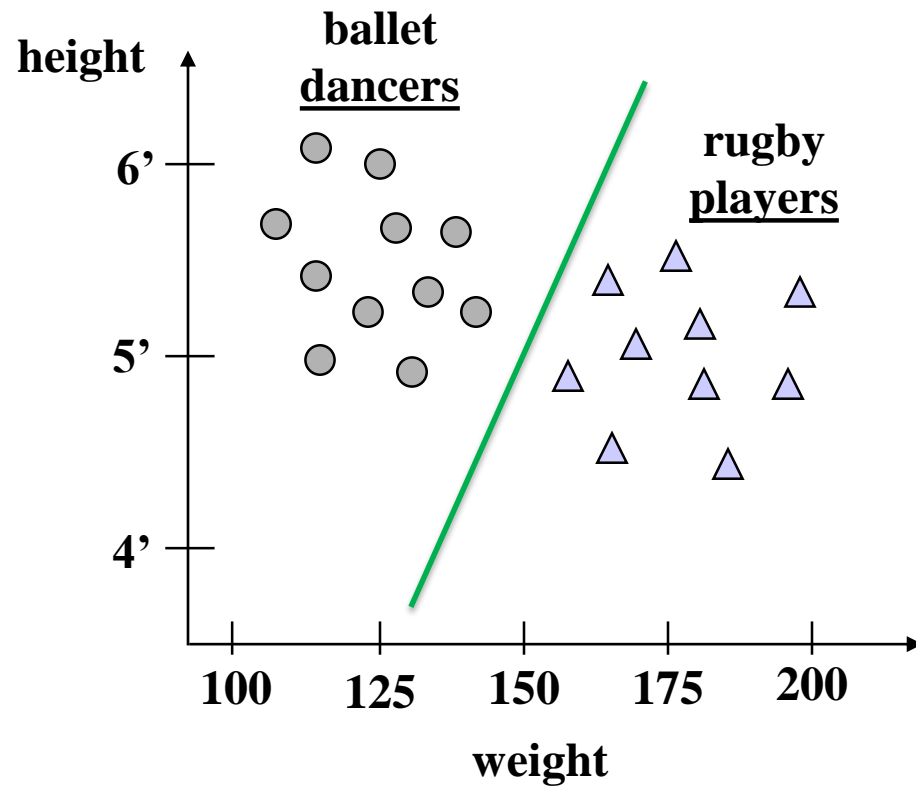


Training Data	
input	output
125, 5'3"	ballet
175, 5'6"	rugby
115, 6'1"	ballet
...	

Possible separators

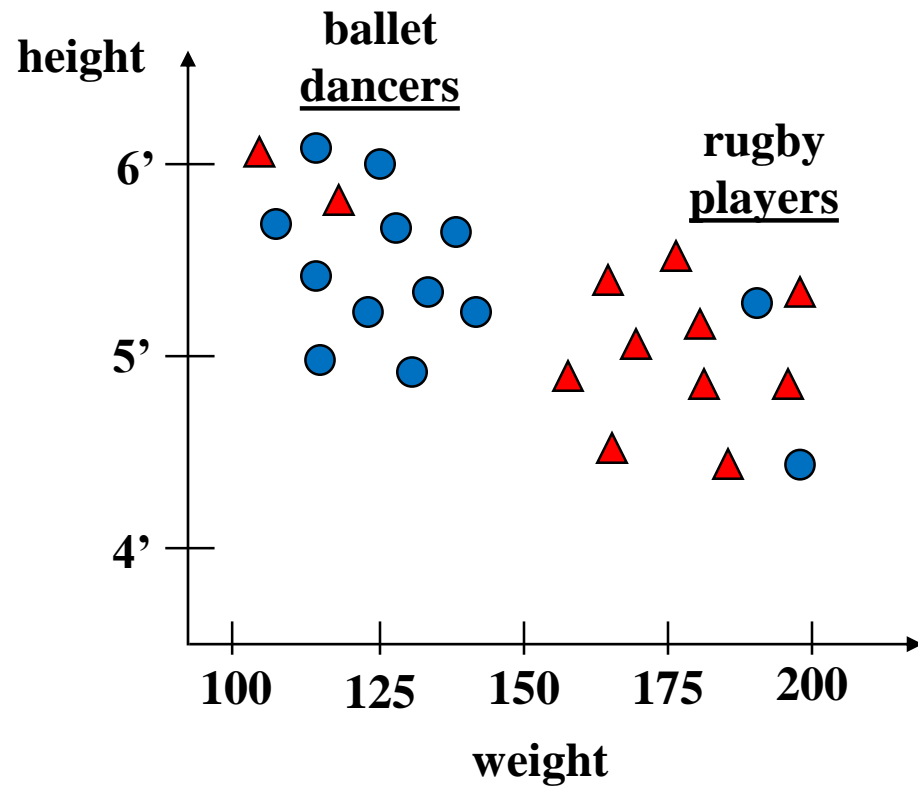


Maximum margin separator



Decision boundary with largest possible distance to example points.

What if the data are not linearly separable?



No line(s) can be drawn that separate the two classes.

Kernel functions

- Data that are not linearly separable in the original space are separable (almost always) in a higher dimensional (non-linear) space

Example transformation of training data

Original training data		
input		output
<i>weight</i>	<i>height</i>	
125	5'3"	ballet
175	5'6"	rugby
115	6'1"	ballet
...		

Transformed training data			
input			output
<i>weight</i> ²	<i>height</i> ²	$\sqrt{2} \times \textit{weight} \times \textit{height}$	
15,625	3,969"	11,136.9	ballet
30,625	4,356"	16,334.2	rugby
13,225	5,329"	11,872.3	ballet
...			

Ensembles of Classifiers

The Basic Idea

The accuracy of supervised learning can be improved by learning an ensemble of classifiers: a set of classifiers whose individual decisions are combined in some way (typically by voting) to classify new examples.

Improved accuracy

- An ensemble can be more accurate than its component classifiers if:
 - individual classifiers disagree with each other (errors made by classifiers are uncorrelated)
 - error rate less than $1/2$
- Example:
 - 21 classifiers, each with error rate of 0.3, majority vote
 - probability 11 or more of the classifiers are simultaneously wrong is 0.026

Constructing ensembles: Bagging

- Learning algorithm is run multiple times, each time with a different subset of training examples
 - each run uses a training set of m examples drawn randomly with replacement from original training set of m examples
 - on average: 63.2% of original training set, with some training examples appearing multiple times

Constructing ensembles: Cross-validated committees

- Learning algorithm is run multiple times, each time with a different subset of training examples
 - divide original training set of m examples into k disjoint subsets
 - construct k overlapping training sets by dropping out a different one of the k subsets

Constructing ensembles: Boosting

- Learning algorithm is run multiple times, each time with a different probability distribution $p_l(x)$ over training examples
 - in iteration l , training set of m examples drawn randomly with replacement from original training set of m examples according to probability distribution $p_l(x)$
 - learning algorithm applied to construct classifier h_l
 - error rate on training set is used to determine $p_{l+1}(x)$
 - effect is to place more weight on misclassified examples
 - at subsequent iterations: progressively more difficult learning problems

Constructing ensembles: Injecting randomness

- Neural networks: multiple runs based on different initial random weights
- Decision trees: when choosing a feature, randomly choose from among the top few

Combining classifiers

- Unweighted voting
- Weighted voting
 - e.g., weight proportional to how well a classifier does on validation set
- Stacking
 - given classifiers h_1, \dots, h_L
 - learn a classifier h which takes as input the outputs of h_1, \dots, h_L

Steps in Supervised Machine Learning for Classification

1. Choose features

- To use supervised learning techniques, problem first has to be phrased as a classification problem
- Choice of distinguishing features is critical to successfully learning a classifier
- Begin with many features that are felt to be promising
- Possibly synthesize new features from existing features.
 - Examples: comparison of two features, max of two features, average of some set of features

2. Collect data

- Data must be representative of what will be seen in practice
- Let S be the set of correctly labeled data, and let $n = |S|$

3. Filter features

- Goal of filtering is to select most important features for constructing a good classifier
- Only selected features are then used in learning; irrelevant and redundant features are removed
- Motivation:
 - efficiency
 - many learning methods do poorly if there are redundant or irrelevant features
- Example: prune feature if...
 - single feature decision tree classifier using this feature is not much better than random guessing, *and*
 - all two feature classifiers using this feature are not much better than random guessing

4. Select classifier (model selection)

- Given S , the set of correctly labeled data, select (learn) classifier h from hypothesis space H
 - if lots of data available:
 - split data into data used for training and data used for testing
 - learn hypothesis h (select classifier) using training data, evaluate h on test data, where there is enough test data for the results to be significant
 - if not lots of data available:
 - use cross-validation or boot-strapping
 - repeatedly learn hypothesis h (select classifier) using different training data, evaluate h on different test data

Select classifier:

Setting parameters

for each choice of parameter settings
 evaluate classifier h learned from training data
 with current choice of parameter settings
return parameter settings that give best performance

- Examples:
 - setting number of hidden units in neural network
 - setting minimum examples at a leaf in decision tree
- Learning may involve splitting data into training and validation sets

5. Evaluate classifier (model assessment)

- Estimate classifier performance by:
 - applying classifier to test data,
 - use estimate from cross-validation, or
 - use estimate from bootstrapping
- If using cross-validation or bootstrapping, after estimate is obtained, combine all of the data into one large training set to obtain final classifier

Evaluate classifier: Cross-validation

- When amount of data is limited

partition data S into k disjoint subsets S_1, \dots, S_k
for $i = 1$ to k
 train on $S - S_i$ to obtain h_i (select h_i using $S - S_i$)
 test h_i on S_i
average the k performance estimates to give an
overall performance estimate

- Common value for k

$k = 10$ ten-fold cross-validation

Evaluate classifier: Bootstrapping

- When amount of data is limited

for $i = 1$ to m

sample with replacement n times from S to give S_i

train on S_i to obtain h_i (select h_i using S_i)

test h_i on $S - S_i$ (unpicked instances)

average the m performance estimates to give an
overall performance estimate


- On average, training set will contain 63% of
instances


Evaluate classifier:

Cost of classification errors

- Cost of false negative and false positive may be unequal

	$y_i = \text{no}$	$y_i = \text{yes}$
$h(x_i) = \text{no}$	0	cost fn
$h(x_i) = \text{yes}$	cost fp	0

false negative 

false positive 

- For example: spam? cost fp = 10, cost fn = 1