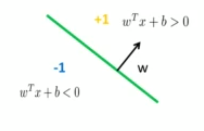
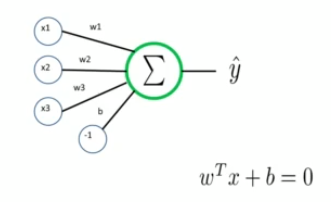
# SVM and Evaluation

## Separating Hyperplane

* X: data point
* Y: label
* W: weight vector
  + The orientation of the hyperplane
* B: bias
* 

## Perceptron

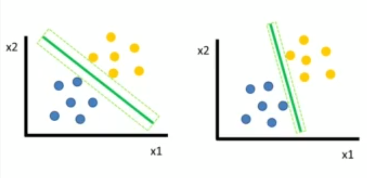


* The basis of neural networks

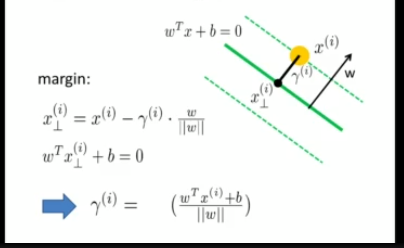
## Support Vector Machine

* Widely used for all sorts of classification problems
* Some people say it is the best off the shelf classifier out there

### Maximum Margin Classification

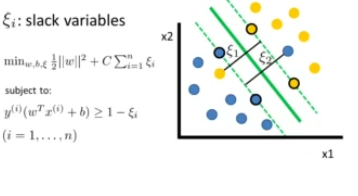


* In SVMs, the most important points are the support vectors
* The solution depends only on the support vectors

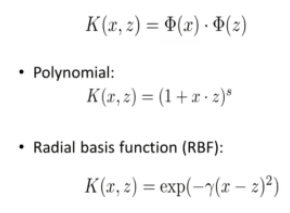


* Gamma is what we care about

### What about outliers?



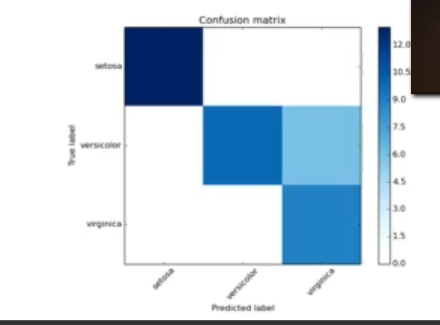
### Kernel Functions



### Tips and Tricks

* SVMs are not scale invariant
* Check if your library normalizes by default
* Normalize your data
  + Mean: 0, std: 1
  + Map to [0, 1] or [-1, 1]
* Normalize test set in same way
* Rbf kernel is a good default
* For parameters try exponential sequences
* Read:
  + A Practical Guide to Support Vector Classification by Hsu

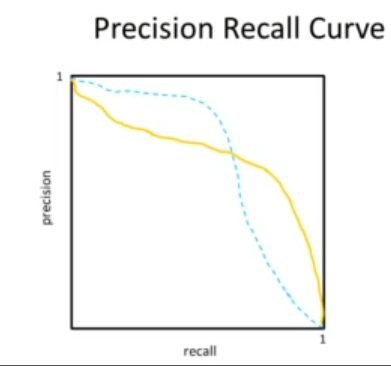
### Multi Class

* One vs All
  + Train n classifier for n classes
  + Take classification with greatest margin
  + Slow training
* One vs one
  + Train n(n-1)/2 classifiers
  + Take majority vote
  + Fast training
* Confusion matrix
  + You want the diagonal to be very high, and everything else as low as possible
  + 

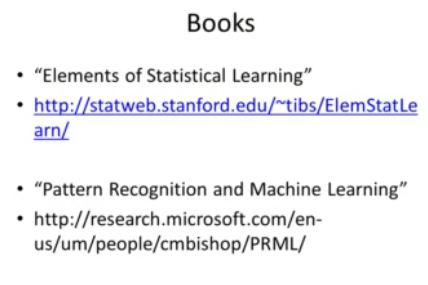
## Parameter Tuning

* Given a classification task
  + Which kernel?
  + Which kernel parameter value
  + Which value for c?

## Precision Recall Curve

* Precision recall is better than ROC for unbalanced data
* Recall: If I pick a random positive example, what is the probability of making the correct prediction?
* Precision: If I take a positive prediction example, what is the probability that it is indeed a positive example?
* 
* We want to be in the upper right corner.

# Decision Trees and Random Forests



## Decision Trees

* Fast training
* Fast prediction
* Easy to understand
* Easy to interpret

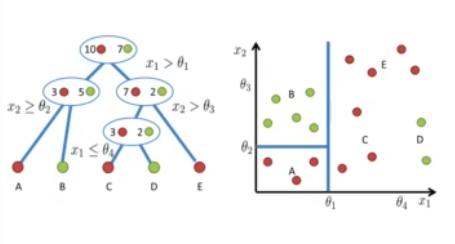
### Training

* Learn the tree structure
  + Which feature to query
  + Which threshold to choose

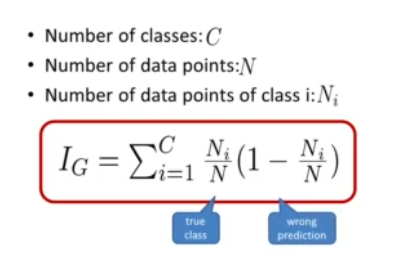
#### When to stop

* Node contains only one class
* Node contains less than x data points
* Max depth is reached
* Node purity is sufficient
* You start to overfit => cross-validation

### Node Purity

* Come up with splits that results in nodes that are pure.
* 

#### Gini Impurity

* Expected error of if you randomly choose a sample and predict the class of the entire node based on it
* 

#### Node Purity Gain

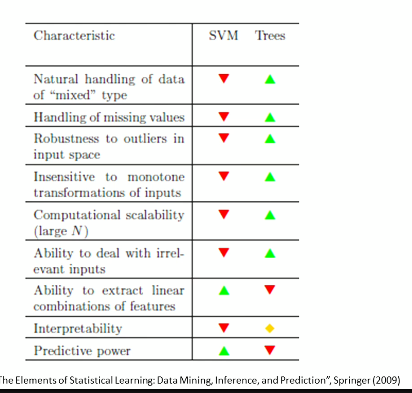
* Compare:
  + Gini impurity of parent node
  + Gini impurity of child nodes

### Tree Pruning

### Decision Trees Disadvantages

* Sensitive to small changes in the data
* Overfitting
* Only axis aligned splits

### Decision Trees vs SM

* 

## Ensemble Methods

### Boostrap

* Resampling method from statistics
* Useful to get error bars on estimates
* You can not do cross validation on this because there is overlap

### Boostrap vs Cross-Valdiation

* Bootstrap has overlap in data sets
* Do not use simple bootstrap to generate train and test data from the same data set

### Bagging

* Bootstrap aggregating
* Characteristics
  + Sample with replacement from your data set
  + Learn a classifier for each bootstrap sample
  + Average the results
* It wants to reduce the variance without introducing more bias
* It reduces overfitting (variance)
* Normally uses one type of classifier
* Not helpful with linear models
* Easy to parallelize

## Random Forests

* Builds upon idea of bagging
* Each tree build from bootstrap sample
* Node splits calculated form random feature subsets