CS 383 – Machine Learning

Nearest Neighbors

Slides adapted from material created by E. Alpaydin
Prof. Mordohai, Prof. Greenstadt, Pattern Classification (2nd Ed.),
Pattern Recognition and Machine Learning

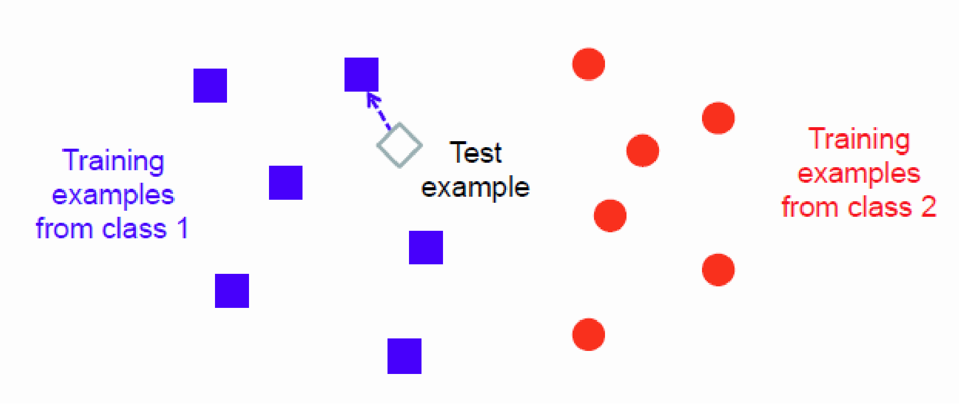
Objectives

- Nearest Neighbor Classifiers

K-Nearest neighbors (KNNs)

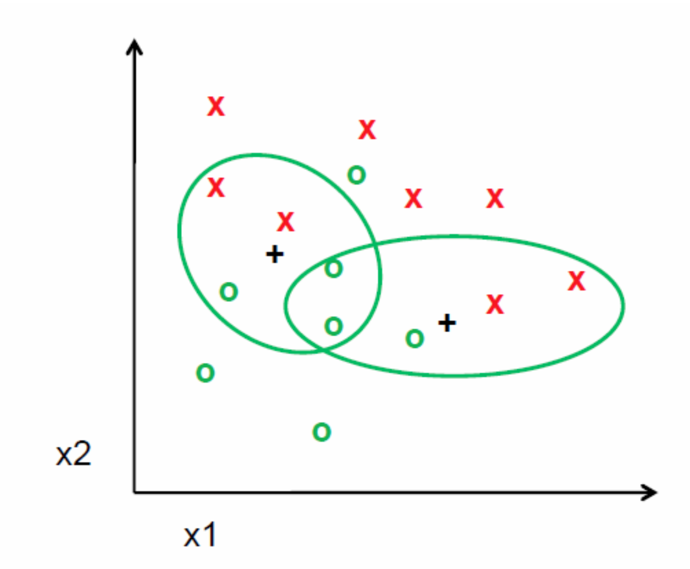
Nearest Neighbor Classifier

- Idea: Assign label to x according to the label of the training example nearest $x' \in trainingset$
- Simple Algorithm
  - All we need is distance/similarity function
    - Or a kernel function
  - No training required!


Training examples from class 1
Test example
Training examples from class 2

k-Nearest Neighbors

- Just using a single nearest neighbor is susceptible to noise.
- So maybe use k-nearest neighbors
  - And choose class that gets the most votes
- Example: 5-nearest neighbors

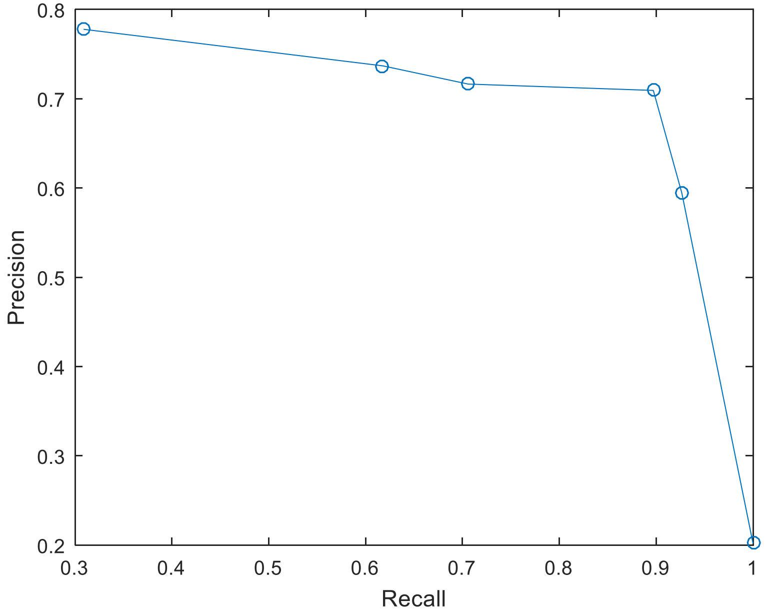


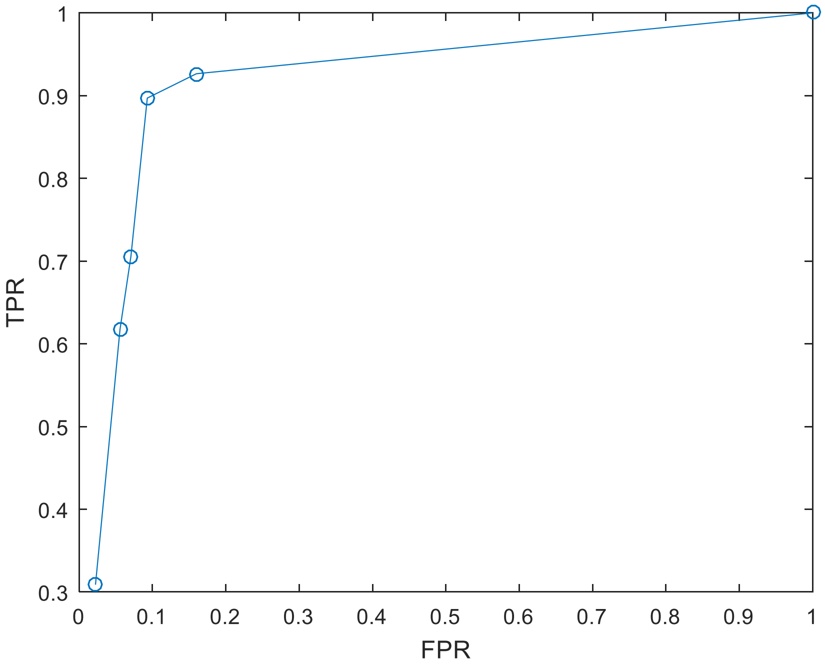
Example

- Intuitively we’d assign the label of the mode of the neighbors’ labels.
    - What should we do if k is even?
- We could also trace out a precision recall (and/or ROC) graph by varying how many neighbors must have a positive label in order for the testing sample to have a positive label.
- So if $k = 5$ then we could vary the number of required votes to be between 0 and 5

Example

- From the PR-Graph it looks like choosing threshold $t = 2$ is best (since $t = 0$ would be the rightmost point)
- We could also plot the ROC graph
  - Here it looks like choosing $t =$1or $t = 2$ would be best (again since $t = 0$ would be the rightmost point)





Final Observations

- Let’s think about this algorithm
  - Supervised or non-supervised?
  - Classification or regression?
  - Model-based or instance-based?
    - When it comes time to test/use, are we using the original data?
  - Linear vs Non-Linear?
  - Can this work on categorical data?
  - Can this work on continuous valued data?
  - Training Complexity?
  - Testing Complexity?
  - How to deal with overfitting?
  - Directly handles multi-class?