CSDS 440: Machine Learning

Soumya Ray (he/him, sray@case.edu)

Olin 516

Office hours T, Th 11:15-11:45 or by appointment

Recap

- In the dual form of the SVM, examples appear as a d____ p____.
- We define the k_____ function K(x,y) as ____ dot _____.
- In many cases, K can be computed more efficiently than the dot product of the feature maps. This is called the k____ t____.
- Intuitively, a kernel function measures the s______ between examples.
- To be a valid kernel function, a function must satisfy M_____ conditions. These say that the kernel matrix must be s______ p_____ s____ d_______.
- Kernels can be applied to many other problems using the R_____ t_____.
- This says that any optimization program of the form $\min_f$ (A) +(B) has a solution which is f=sum_i(a)(b)(c).
- The SVM uses the H_____ loss function whereas LR uses the L_____ loss function.
- What are some of the pros of SVMs? Cons?
- An ensemble is a c____ of c____ combined with v__.

Today

- Part 2: Ensemble Methods

Single vs. multiple classifiers

- Suppose for some problem we have k classifiers $h_1, \ldots, h_k$ that:
  - Each has error less than chance: $\varepsilon_i < ½$
  - Make *uncorrelated* errors on new examples
- Suppose we combine their predictions on a new example via majority vote
- What is the error rate of the combined system?

Single vs. multiple classifiers

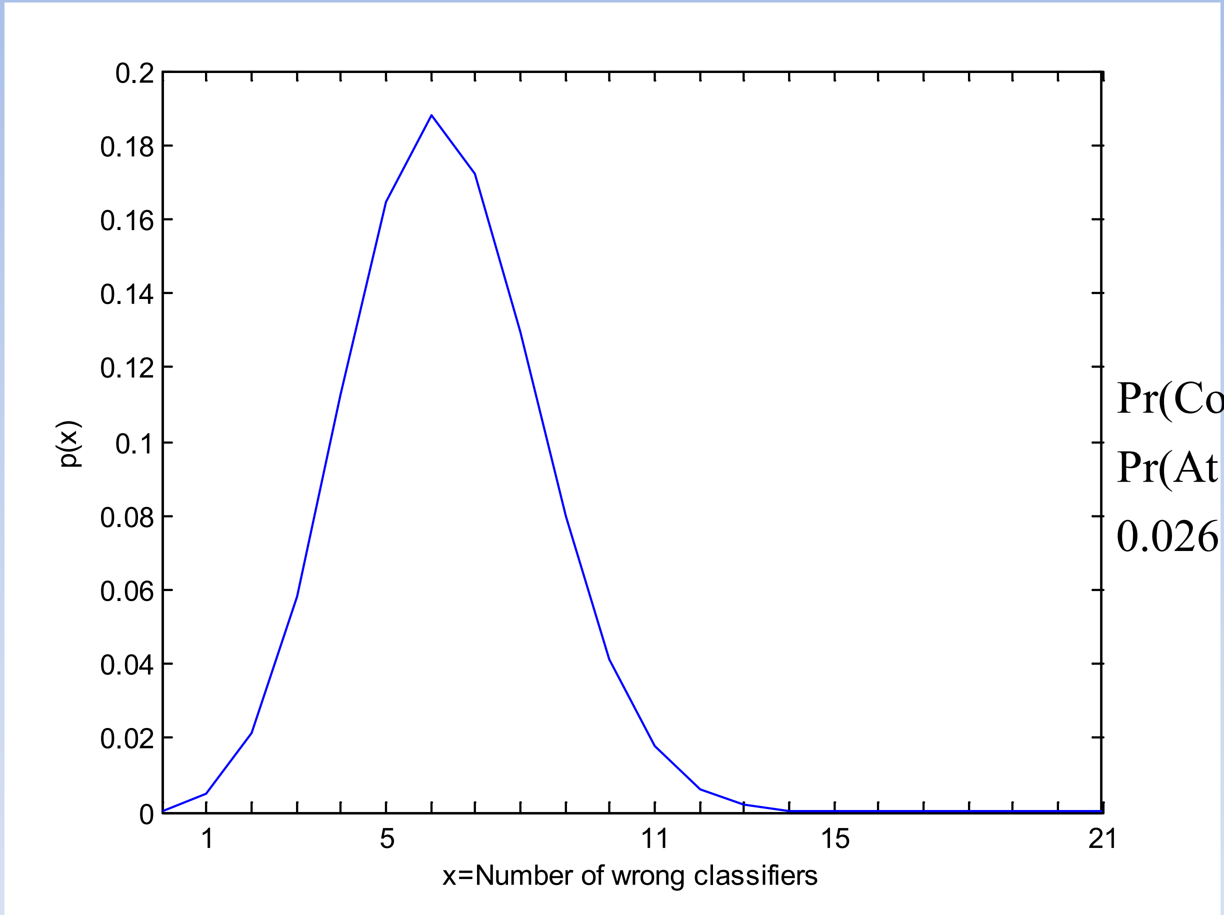
- Since we combine using majority voting, to get an example wrong, at least half the classifiers must be wrong

- Since the errors of each classifier is uncorrelated, we can treat each classifier as an "independent trial" for the new example

Single vs. multiple classifiers

- Assume all the classifiers have the same error rate ε
  - No loss of generality—can choose max error over all classifiers to get upper bound on error rate of combined system
- Then the number of wrong classifiers follows a Binomial distribution with parameters ε, k

Example: $\varepsilon=0.3, k=21$



Pr(Combined System is wrong)=
Pr(At least 11 classifiers are wrong)=
0.026

“Ensembles” of Classifiers

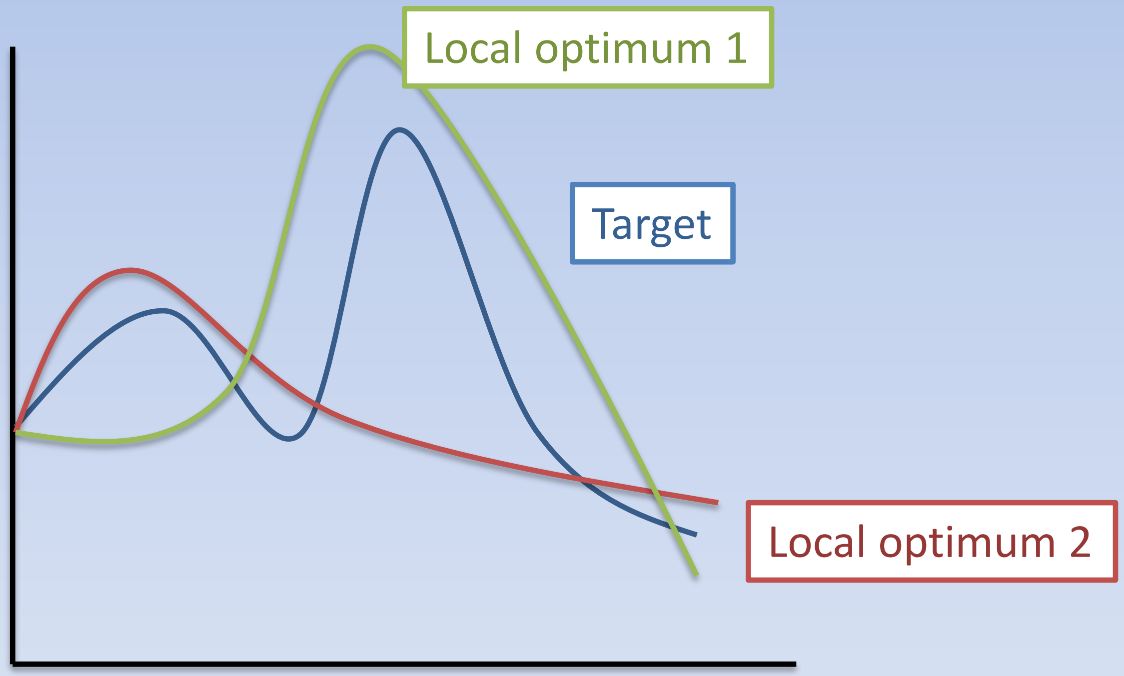
- A collection of classifiers combined using some sort of voting scheme is called an “ensemble”
- Getting classifiers that have error less than chance is (usually) easy
  - Note that this is *generalization* error
- Getting classifiers that make uncorrelated errors is usually not
  - But even if not, ensembles can outperform single classifiers in practice

Why ensembles do well in practice (1)

- Many classifiers are a result of some search procedure (e.g. gradient descent) which can get stuck in local optima

- Averaging these “local optimum” classifiers can provide a better approximation to the target function

Picture



Why ensembles do well in practice (2)

- An ensemble of classifiers may have a more complex decision boundary than any single classifier
  - E.g. an ensemble of voting linear classifiers is not (generally) a linear classifier

Why ensembles do well in practice (3)

- Consider probabilistic classification

- Given a training sample D, what is the most probable classification for a new instance $\mathbf{x}_{new}$?
  - i.e. what is $\Pr(Y_{new} = y \mid D, \mathbf{x}_{new})$?
  - Assume you are investigating a hypothesis class H

Bayesian Model Averaging

$$\Pr(Y_{new} = y \mid D, \mathbf{x}_{new})$$
$$= \sum_{h \in H} \Pr(Y_{new} = y \mid D, h, \mathbf{x}_{new}) \Pr(h \mid D, \mathbf{x}_{new})$$
$$= \sum_{h \in H} \Pr(Y_{new} = y \mid h, \mathbf{x}_{new}) \Pr(h \mid D)$$

Classification according to h

Posterior probability of h given training sample

- Also called Bayes optimal classification
- Cannot be outperformed on average by any single hypothesis in H

The downsides

- How large an ensemble to use is not well understood
  - Too small—no effect, too large—tends to overfit

- An ensemble is usually much harder to interpret than a single classifier
  - E.g. a decision tree vs a set of trees

- Computation time, memory etc. all increase
  - (sometimes we can parallelize)

General Ensemble Construction

- Can construct ensembles in several ways
  - Modifying the training set
  - Modifying the set of attributes
  - Modifying the outputs
  - Randomizing the learning algorithm

Modifying the Training Set

- General idea:
  - Create multiple training sets, each different from the others in some way
  - Apply learning algorithm to each set
  - Resulting classifiers vote on new examples
- Works best for “unstable” algorithms
  - Small change to data can lead to large change in solution
- Two important methods
  - Bagging
  - Boosting

Bagging (Breiman 96)

- "**B**ootstrap **Agg**regation"
- Each training sample is a bootstrap replicate of the initial set
  - If the set has size m, sample m examples *uniformly with replacement* from it
- To classify a new example, use majority voting