Computational functional genomics
(Spring 2002: Lecture 12)

David K. Gifford
(Adapted from a lecture by Tommi S. Jaakkola)
MIT LCS and AI Lab

gifford@mit.edu
Classification approaches

- The two main classification approaches
  
  1. Generative approach
     - build a statistical model
       e.g., mixture model
  
  2. Discriminative approach
     - specify a decision rule/boundary directly
       e.g., logistic regression
Discriminative approach: motivation

- One reason for using discriminative approaches is robustness:

<table>
<thead>
<tr>
<th>Class conditional models</th>
<th>type of decision boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian, equal covariances</td>
<td>linear</td>
</tr>
<tr>
<td>Independent exponential</td>
<td>linear</td>
</tr>
<tr>
<td>Independent binomial</td>
<td>linear</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

If we estimate a linear decision boundary directly we are less dependent on what the true class conditional distributions are.

- Examples of discriminative classifiers
  - linear discriminant analysis
  - Logistic regression (generalized linear models generalized additive models)
  - Support vector machines
Discriminative approach to classification

• Simple example: linear decision boundaries

\[ f(x; \theta) = \theta_0 + \theta_1 x_1 + \ldots + \theta_m x_m > 0, \text{ class } = 1 \quad (1) \]
\[ \leq 0, \text{ class } = 0 \quad (2) \]

parameters \( \theta = \{\theta_0, \theta_1, \ldots, \theta_m\} \) and \( x = \{x_1, \ldots, x_m\} \) is a vector of expression levels.

• Similarly to the generative model case, we have to solve the
  1. estimation problem
  2. variable selection problem
Fisher linear discriminant analysis

- Set the parameters $\theta$ so that the two class populations are maximally separated along this direction

- We try to maximize

$$J(\theta) = \frac{(\text{Separation of means along } \theta)^2}{\text{Sum of within population variances along } \theta}$$  \hspace{1cm} (3)
Solution to Fisher linear discriminant analysis

• We try to maximize

\[
J(\theta) = \frac{(\text{Separation of means along } \theta)^2}{\text{Sum of within population variances along } \theta}
\]  

(4)

\[
J(\theta) = \frac{(\theta^T \mu_1 - \theta^T \mu_0)^2}{\theta^T (n_1 \Sigma_1 + n_0 \Sigma_0) \theta}
\]  

(5)

• The solution is

\[
\theta = (n_1 \Sigma_1 + n_0 \Sigma_0)^{-1} (\mu_1 - \mu_0)
\]  

(6)

and is optimal for two Gaussian populations with equal covariances \((\Sigma_1 = \Sigma_0)\)
Logistic regression

• In the logistic regression model, the log-odds of decisions is a linear function of the explanatory variables

\[
\log \frac{P(\text{class} = 1|x, \theta)}{P(\text{class} = 0|x, \theta)} = \theta_0 + \theta_1 x_1 + \ldots + \theta_m x_m
\]  

(7)

Note that the value of the discriminant function (distance away from the boundary) is now interpreted as our confidence in the decisions

• Recall our simple generative Gaussian classifier:

\[
P(\text{class} = 1|x, \theta) = \frac{P(x|\mu_1, \sigma^2)}{P(x|\mu_1, \sigma^2) + P(x|\mu_0, \sigma^2)}
\]

\[
= \frac{1}{1 + \exp \left( -\log \frac{P(x|\mu_1, \sigma^2)}{P(x|\mu_0, \sigma^2)} \right)}
\]

\[
= g(w_0 + w_1 x)
\]

(8) (9) (10)

where \( g(z) = (1 + \exp (-z))^{-1} \) is the logistic function
Logistic regression: estimation

- We can estimate the parameters $\theta$ by maximizing the likelihood of the training labels $\{c^{(1)}, \ldots, c^{(n)}\}$ corresponding to the measurements $\{x^{(1)}, \ldots, x^{(n)}\}$

$$\prod_t P(\text{class} = c^{(t)} | x^{(t)}, \theta)$$

(11)

- Suppose the labels are binary $c^{(t)} \in \{0, 1\}$. We can derive a simple incremental update rule for the parameters

$$\theta_i \leftarrow \theta_i + \epsilon \cdot \left( c^{(t)} - P(\text{class} = 1 | x^{(t)}, \theta) \right) \cdot x_i^{(t)}$$

(12)

where $\epsilon$ is a learning rate and $x_0^{(t)} = 1$ so we can estimate $\theta_0$
Logistic regression: regularization

- In the absence of any data, we’d like the parameters to go to zero (equal probability class predictions)

We add a penalty term (prior) to the likelihood criterion that encourages small parameter values

$$\max_{\theta} \left\{ \log\text{-likelihood}(\theta) + \log\text{-prior}(\theta) \right\}$$

(13)

- With such regularization the learning/update rule changes only slightly

$$\theta_i \leftarrow \theta_i + \epsilon \cdot \left[ (c(t) - P(class = 1|x^{(t)}, \theta)) \cdot x_i^{(t)} - C \cdot \theta_i \right]$$

(14)

where $C > 0$ is a regularization parameter.
Logistic regression: example

- Golub et al. leukemia classification problem
  - 7130 ORFs
  - 38 labeled training examples,
  - 34 test examples

- If we apply the regularized logistic regression model without variable selection, we get 1 test error (out of 34).
The figure shows the values of the discriminant function

$$f(x; \theta) = \theta_0 + \theta_1 x_1 + \ldots + \theta_m x_m$$  \hspace{1cm} (15)

across the test examples

- Variable selection?
Support vector machines

• Basics of support vector machines
  – optimal hyperplane
  – finding the optimal hyperplane
  – kernel function
  – complexity, control of complexity
  – how many training examples?

• Examples
“Optimal” hyperplane

• Let’s assume for simplicity that the classification problem is linearly separable

• Maximum margin hyperplane is maximally removed from all the training examples

• This hyperplane can be defined on the basis of only a few training examples called support vectors
“Optimal” hyperplane cont’d

• We are estimating a linear classifier:

\[
f(x; \Theta) = \theta_0 + x_1 \theta_1 + \ldots + x_d \theta_d = \theta_0 + \theta \cdot x
\]
where \( \Theta = \{\theta_0, \theta\} \)

• We can try to find the “optimal” hyperplane by requiring that the sign of the decision boundary (clearly) agrees with all the training labels

\[
y^{(t)} \left[ \theta_0 + \theta \cdot x^{(t)} \right] \geq 1, \quad t = 1, \ldots, n
\]
where \( f(x; \Theta) \)

where the labels \( y^{(t)} \) are \( \pm 1 \).

BUT...
- this is actually an alternative definition of linear separability
- there are multiple answers
- larger values of \( \theta_0, \theta \) would yield “larger” separation.
Support vector machine

- We find the smallest parameter values that still satisfy the classification constraints
- We minimize

\[ \|\theta\|^2 / 2 = \sum_{j=1}^{d} \frac{\theta_i^2}{2} \]

subject to the classification constraints

\[ y^{(t)} [\theta_0 + \theta \cdot x^{(t)}] \geq 1, \quad t = 1, \ldots, n \]

- Only a few of the classification constraints are relevant
  \[ \Rightarrow \text{support vectors} \]
Minimizing $\|\theta\|^2$ with constraints

- We introduce Lagrange multipliers $\alpha_i$ for each constraint (example) and pick $\theta_0$ and $\theta$ to minimize

$$J(\theta, \theta_0, \alpha) = \|\theta\|^2/2 - \sum_{i=1}^{n} \alpha_i (y^{(i)}(\theta_0 + \theta^T x^{(i)}) - 1)$$  \hspace{1cm} (17)

For fixed $\alpha_i$ we find

$$\frac{\partial}{\partial \theta} J(\theta, \theta_0, \alpha) = \theta - \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)} = 0$$  \hspace{1cm} (18)

$$\frac{\partial}{\partial \theta_0} J(\theta, \theta_0, \alpha) = - \sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$  \hspace{1cm} (19)
Solution

- Each classification constraint
  \[ y^{(t)} [\theta_0 + \theta \cdot x^{(t)}] \geq 1, \quad t = 1, \ldots, n \]
  has a coefficient \( \alpha_t \) (Lagrange multiplier) that tries to enforce the constraint
  \[ \alpha_t = 0 \Rightarrow \text{constraint not relevant} \]
  \[ \alpha_t > 0 \Rightarrow \text{relevant constraint (larger values, more relevant)} \]

- The Lagrange multipliers try to “push” the boundary in different directions
Solution

- Solving for $\{\theta_0, \theta\}$ leaves us with the following (dual) optimization problem over Lagrange multipliers associated with the constraints:

We maximize

$$J(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)} \cdot x^{(j)})$$  \hspace{1cm} (20)

subject to the constraints

$$\alpha_i \geq 0, \quad i = 1, \ldots, n, \quad \sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$  \hspace{1cm} (21)

(For non-separable problems we simply limit $\alpha_i \leq C$ for some positive constant $C$)

- This is a quadratic programming problem
Interpretation of support vector machines

• Before:
  – example vectors $x^{(t)}$ of dimension $d$ (e.g., the number of genes)
  – parameters $\theta_1, \ldots, \theta_d$ which multiply each component of $x$ (e.g., genes)

• After:
  – real valued inner products $(x^{(t)} \cdot x^{(t')})$ measuring how similar the training examples are to each other
  – weights $\alpha_i$ on the examples indicating how important each training example is to the classification task
Interpretation of support vector machines cont’d

- To use support vector machines we
  1. specify similarities between the examples (i.e., \((x \cdot x')\))
  2. set the example weights \(\{\alpha_i\}\) by enforcing the classification constraints (the quadratic programming problem)

- We make decisions by comparing each new sample \(x\) with only the \(k\) support vectors \(x^{(t_1)}, \ldots, x^{(t_k)}\)

\[
\hat{y} = \text{sign}\left( \sum_{i=1}^{k} \tilde{\alpha}_i \ y^{(t_i)} \ (x^{(t_i)} \cdot x) + \theta_0 \right)
\]

(22)

- Non-linear classifiers? Similarity?
Non-linear classifier

- So far the SVM classifier is able to separate our sample populations only linearly.
- We can easily obtain a non-linear classifier by mapping our samples $x = [x_1 \ x_2]$ (involving two genes) into longer feature vectors $\phi(x)$

$$
\phi(x) = [x_1^2 \ x_2^2 \ \sqrt{2}x_1x_2 \ \sqrt{2}x_1 \ \sqrt{2}x_2 \ 1]
$$

(23)

and applying the linear classifier to the new feature vectors $\phi(x)$ instead.

- This way we can for example take into account dependencies among the genes to better classify tissue samples (or genes into functional categories).
Non-linear classifier

Linear separator in the **feature space**

Non-linear separator in the **original space**
Kernel function and feature mapping

- Let’s look at the previous example in a bit more detail

\[ x \rightarrow \phi(x) = [x_1^2 \ x_2^2 \ \sqrt{2}x_1x_2 \ \sqrt{2}x_1 \ \sqrt{2}x_2 \ 1] \]  
(24)

- If we try to find the “optimal” hyperplane in the feature space, i.e., using \( \phi(x) \) as the observed examples, we have to deal with (only) the inner products (kernels) between such feature vectors

\[
\phi(x) \cdot \phi(x') = x_1^2x_1'^2 + x_2^2x_2'^2 + 2x_1x_2x_1'x_2' + 2x_1x_1' + 2x_2x_2' + 1 \\
= (1 + x_1x_1' + x_2x_2')^2 \\
= \left(1 + (x \cdot x')\right)^2
\]  
(25)

But these can be evaluated without ever explicitly constructing the feature vectors \( \phi(x) \)!
Other examples of kernel functions

• There are a number of feature mappings that behave in nicely in this way

• Linear kernel

\[ K(x, x') = (x \cdot x') \] (26)

• Polynomial kernel

\[ K(x, x') = \left(1 + (x \cdot x')\right)^p \] (27)

where \( p = 2, 3, \ldots \). To get the feature vectors we concatenate all \( p^{th} \) order polynomial terms of the components of \( x \) (weighted appropriately)

• Radial basis kernel

\[ K(x, x') = \exp \left( -\frac{1}{2} \|x - x'\|^2 \right) \] (28)

In this case the feature space consists of functions and results in a non-parametric classifier.
SVM examples

- Linear
- 2nd order polynomial
- 4th order polynomial
- 8th order polynomial
Support vector machine results

- Golub et al. leukemia classification problem
  - 7130 ORFs
  - 38 labeled training examples,
  - 34 test examples

- Let’s blindly apply SVMs to this problem using polynomial kernels of degree $p = 1, 2, 4, 8$.

  We get 1 test error for all classifiers regardless of their complexity.
  There doesn’t seem to be much overfitting...
The figure shows the discriminant function values for the test samples resulting from polynomial kernels of degree $p = 1, 2, 4, 8$. 
Support vector machine results cont’d

• Recall that the cross-validation error for the support vector machine is related to the number of support vectors

\[
CV \text{ error } \leq \frac{\text{number of support vectors}}{\text{number of training examples}}
\]  

(29)

• In our classification problem we have

<table>
<thead>
<tr>
<th>kernel degree</th>
<th>training error</th>
<th># of support vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>28</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>29</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>33</td>
</tr>
</tbody>
</table>

These bounds are rather loose ... **use cross-validation**
We would like to characterize when we can expect the classifier to generalize well, i.e., when

Low training error ⇒ low test error

For this we need to have some notion of complexity of the classifier and relate this to the number of training examples that we need

\[
\text{Test error} - \text{Training error} \leq O \left( \sqrt{\frac{d}{n}} \right)
\]
Complexity and the size of the margin