Structured learning for sequence labeling
Part 1: Linear Classifiers

Hakan Erdogan
Sabanci University

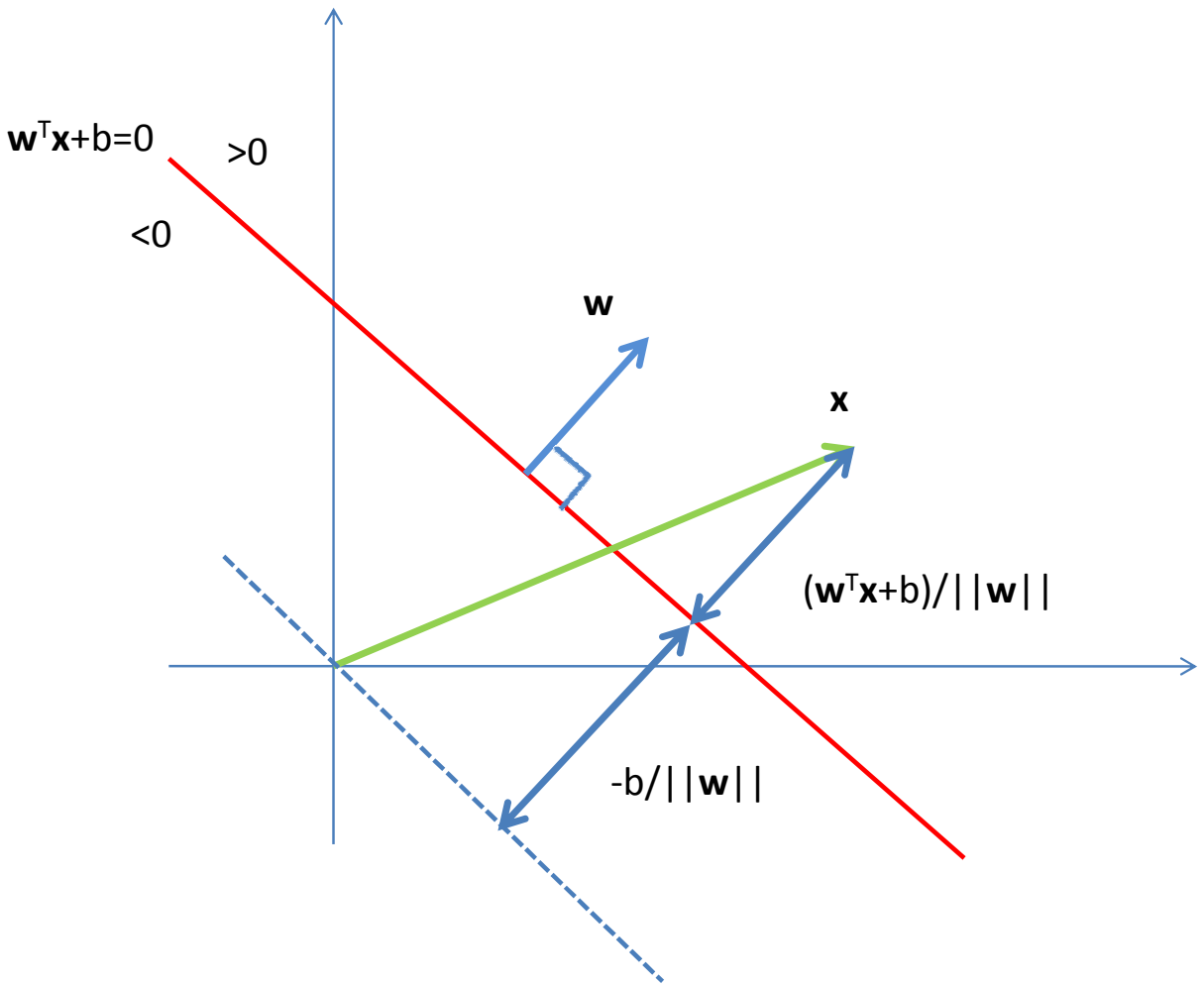
August 2-5, 2010
Enterface'10 workshop, Amsterdam

Outline

Linear binary classification I

- Given training data $\{(\mathbf{x}_i, y_i) : i \in [N]\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathcal{Y} = \{-1, +1\}$
- $[N] = \{1, \ldots, N\}$, $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,d}]^T$ are feature vectors, y_i are class identities
- Find a weight vector $\tilde{\mathbf{w}} \in \mathbb{R}^{d+1}$ such that for test data $\mathbf{x}$, we obtain a score $\tilde{\mathbf{w}}^T \tilde{\mathbf{x}} = \mathbf{w}^T \mathbf{x} + b$ where:
  - $\tilde{\mathbf{x}} = [\mathbf{x}^T, 1]^T$ is the augmented feature vector
  - $\tilde{\mathbf{w}} = [\mathbf{w}^T, b]^T$ is the augmented weight vector, b is called the bias term
- $\tilde{\mathbf{w}}$ represents a hyperplane in $\mathbb{R}^{d+1}$, it is the normal vector to the hyperplane $\{\tilde{\mathbf{x}} : \tilde{\mathbf{w}}^T \tilde{\mathbf{x}} = 0\}$ that passes through the origin

Linear binary classification II



Linear binary classification

- The score can be used to classify a new sample $\mathbf{x}$ into one of two classes by thresholding:

$$\hat{y} = \begin{cases} +1 & \text{if } \tilde{\mathbf{w}}^T \tilde{\mathbf{x}} \geq T \\ -1 & \text{if } \tilde{\mathbf{w}}^T \tilde{\mathbf{x}} < T \end{cases}$$

- We can obtain an ROC curve (or DET curve) by changing the threshold T
- By default, we may assume $T = 0$ since the bias term is included in the model
- One can also obtain posterior probabilities $p(y|\mathbf{x})$ from the score $\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}$
- The problem: how to obtain the "best" weight vector $\tilde{\mathbf{w}}$?
- We consider three different classifiers: Fisher's linear discriminant, logistic regression and support vector machines

Fisher's linear discriminant (FLD) I

- Assume each class' data are multi-variate Gaussian distributed with a common covariance matrix (homoscedastic)
- $p(\mathbf{x}|y) = \mathcal{N}(\mathbf{x}; \mu_y, \Sigma)$ where Σ is the common covariance matrix, μ_y are class means

$$\mathcal{N}(\mathbf{x}; \mu_y, \Sigma) = \frac{\exp\left\{-\frac{1}{2}(x-\mu_y)^T\Sigma^{-1}(x-\mu_y)\right\}}{((2\pi)^d|\Sigma|)^{1/2}}$$

- Using Bayes' therorem, we can show that

$$p(y=1|\mathbf{x}) = \frac{p(\mathbf{x}|y=1)p(y=1)}{\sum_{y'\in\mathcal{Y}} p(\mathbf{x}|y')p(y')} = \sigma(\mathbf{w}^T\mathbf{x}+b)$$

where

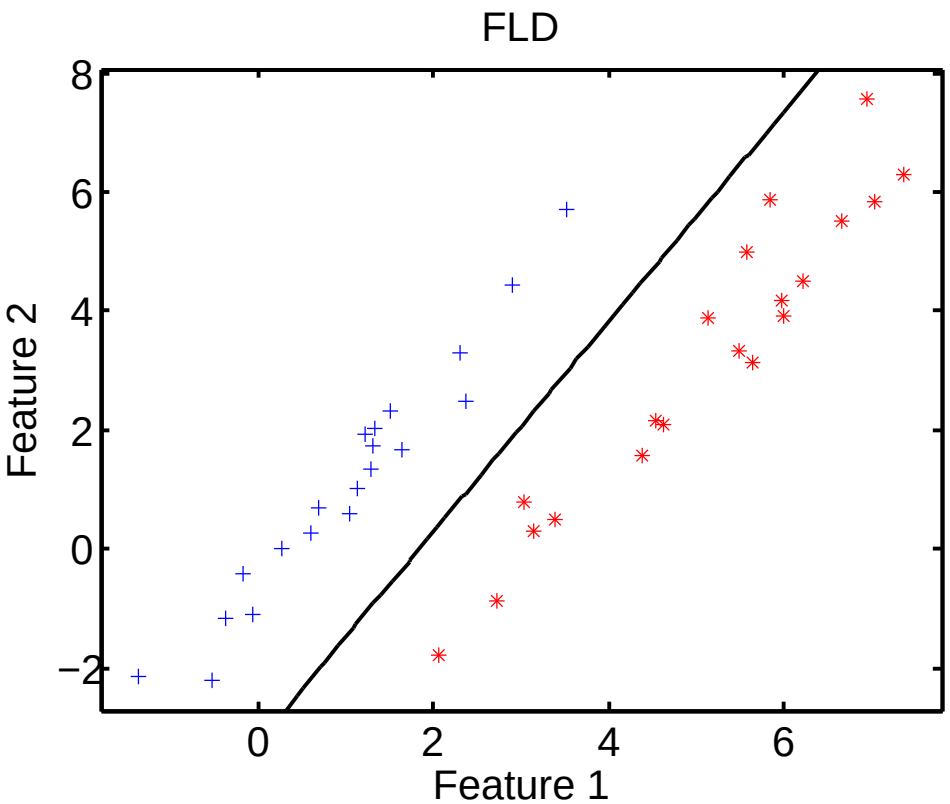
$$\sigma(t) = (1+\exp(-t))^{-1}$$

Fisher's linear discriminant (FLD) II

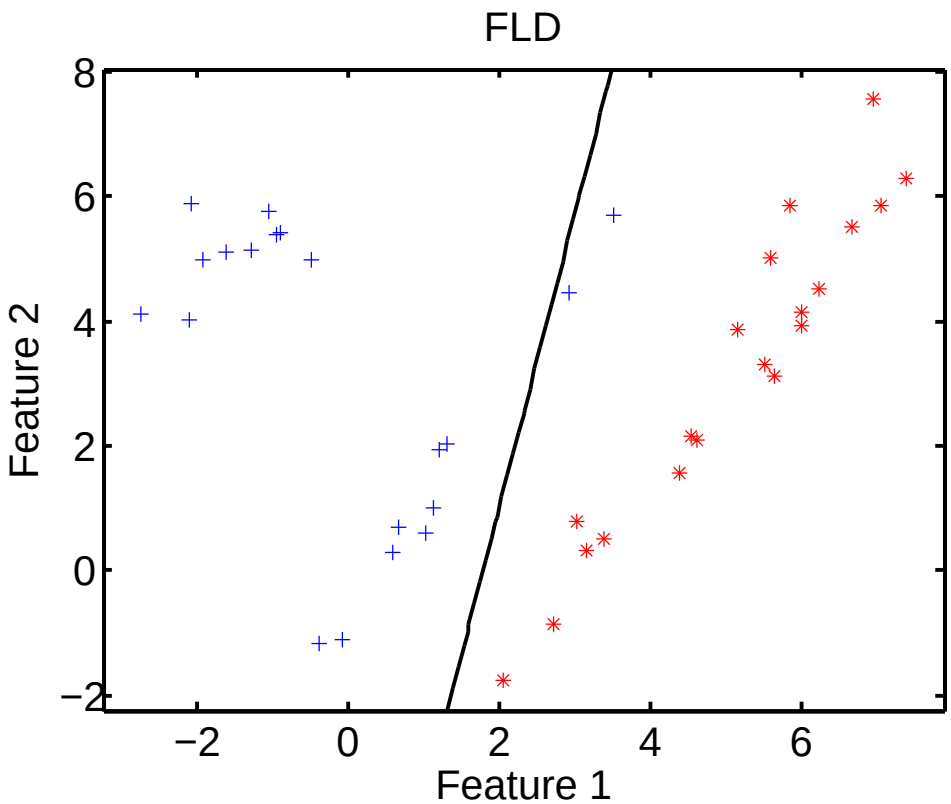
is the logistic sigmoid function, and

$$\begin{aligned} \mathbf{w} &= \Sigma^{-1}(\mu_1 - \mu_{-1}) \\ b &= -\frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_{-1}^T\Sigma^{-1}\mu_{-1} + \log(\frac{p(y=1)}{p(y=-1)}) \end{aligned} \tag{1}$$

- Fisher's linear discriminant yields a linear boundary for classification
- Fisher's linear discriminant is a **generative model** for $\mathbf{x}$ conditioned on y
- It seems a waste to find two class means (of size $2d$) and a common covariance matrix (size $d(d+1)/2$) where in the end what matters for classification is the weight vector $\mathbf{w}$ and the bias b (size $d+1$ only)



Modeling assumption valid



Modeling assumption invalid

Logistic regression I

- Discriminative linear model as opposed to FLD which is a generative model
- Assume $p(y = 1|\mathbf{x}) = \sigma(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})$ where $\sigma(.)$ is the logistic sigmoid [Bishop, 2006]
- Find parameters $\tilde{\mathbf{w}}$ by maximizing the log-likelihood of the class identities y, $p(y|\mathbf{x})$ (instead of $p(\mathbf{x}|y)$ in FLD, a generative model) using the training data
- Define $\pi_i(\tilde{\mathbf{w}}) = \sigma(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)$

$$\hat{\tilde{\mathbf{w}}} = \arg\max_{\tilde{\mathbf{w}}} \sum_{y_i=1} \log(\pi_i(\tilde{\mathbf{w}})) + \sum_{y_i=-1} \log(1 - \pi_i(\tilde{\mathbf{w}}))$$

Logistic regression II

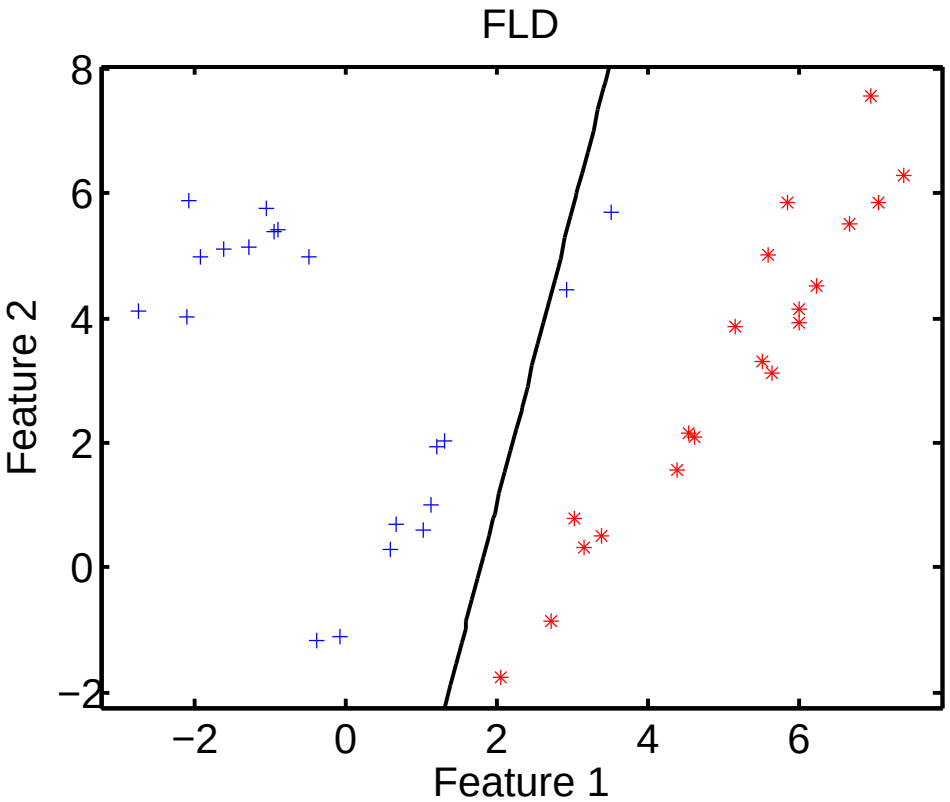
- Gradient of the log-likelihood wrt to the parameters is [Bishop, 2006]

$$\nabla L(\tilde{\mathbf{w}}) = \sum_{y_i=1} x_i(\pi_i(\tilde{\mathbf{w}}) - 1) + \sum_{y_i=-1} x_i \pi_i(\tilde{\mathbf{w}})$$

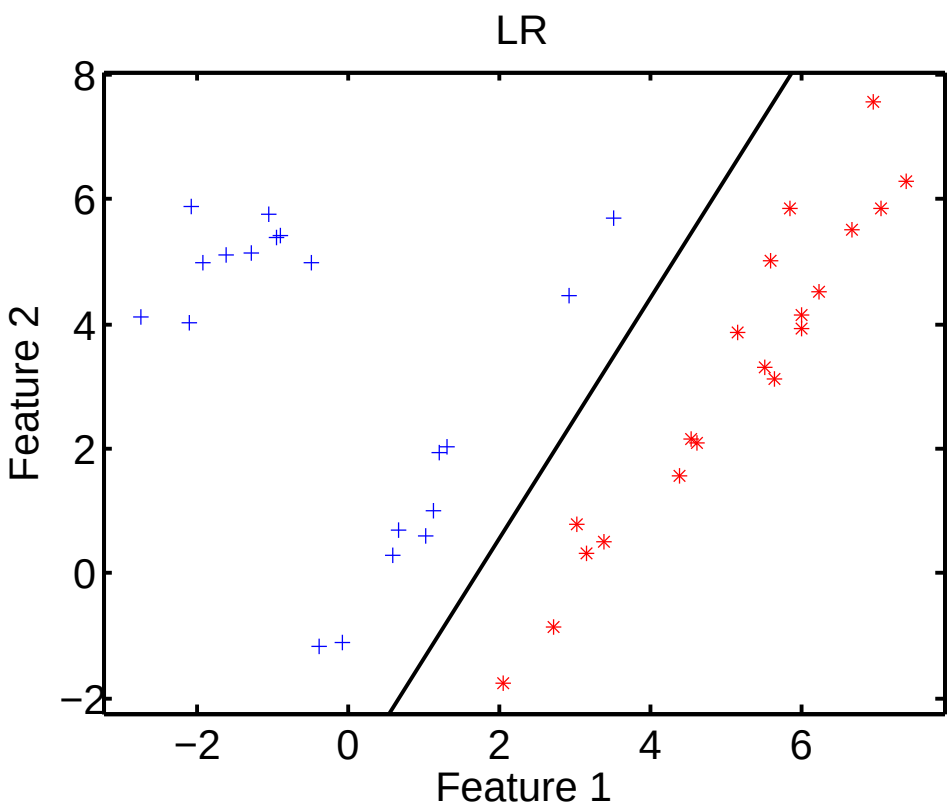
- There is an iterative reweighted least squares (IRLS) algorithm to solve for $\tilde{\mathbf{w}}$ given in [Bishop, 2006]
- We only estimate $d + 1$ parameters from data directly
- There is no need to assume a distribution $p(\mathbf{x}|y)$ since $\mathbf{x}$'s are given to us in a training scenario
- So, we only model $p(y|\mathbf{x})$ which becomes "discriminative modeling"
- The assumed parametric form of $p(y|\mathbf{x})$ comes from the generative FLD model though!
- Question: Can I get back a discriminatively trained conditional Gaussian distributions from logistic regression result $\tilde{\mathbf{w}}$?

Logistic regression III

- Answer is yes, choosing μ_y and Σ consistent with $\tilde{\mathbf{w}}$ obtained through logistic regression using equation 2 will give discriminatively trained parameters for Gaussians (note that they will not be ML trained parameters)



FLD when assumption invalid



LR with the same data

Support vector machines I

- A discriminative classifier that tries to maximize the soft margin between classes to increase generalizibility

$$\min \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i$$

subject to:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \quad (2)$$

$$\xi_i \geq 0 \quad (3)$$

Support vector machines II

- The constraint optimization is simply equivalent to minimizing the following objective function wrt $\mathbf{w}$ and b without constraints (called the primal formulation)

$$\min \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}(1 - y_i(\mathbf{w}^T\mathbf{x} + b))_+$$

where $(x)_+ = \max(x, 0)$ and C is a fixed parameter to be determined

Support vector machines III

- Usually, the dual formulation is used to solve the SVM problem since it appears to be easier, results in sparsity due to support vectors and enables using kernel functions for nonlinear separability

$$\max \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

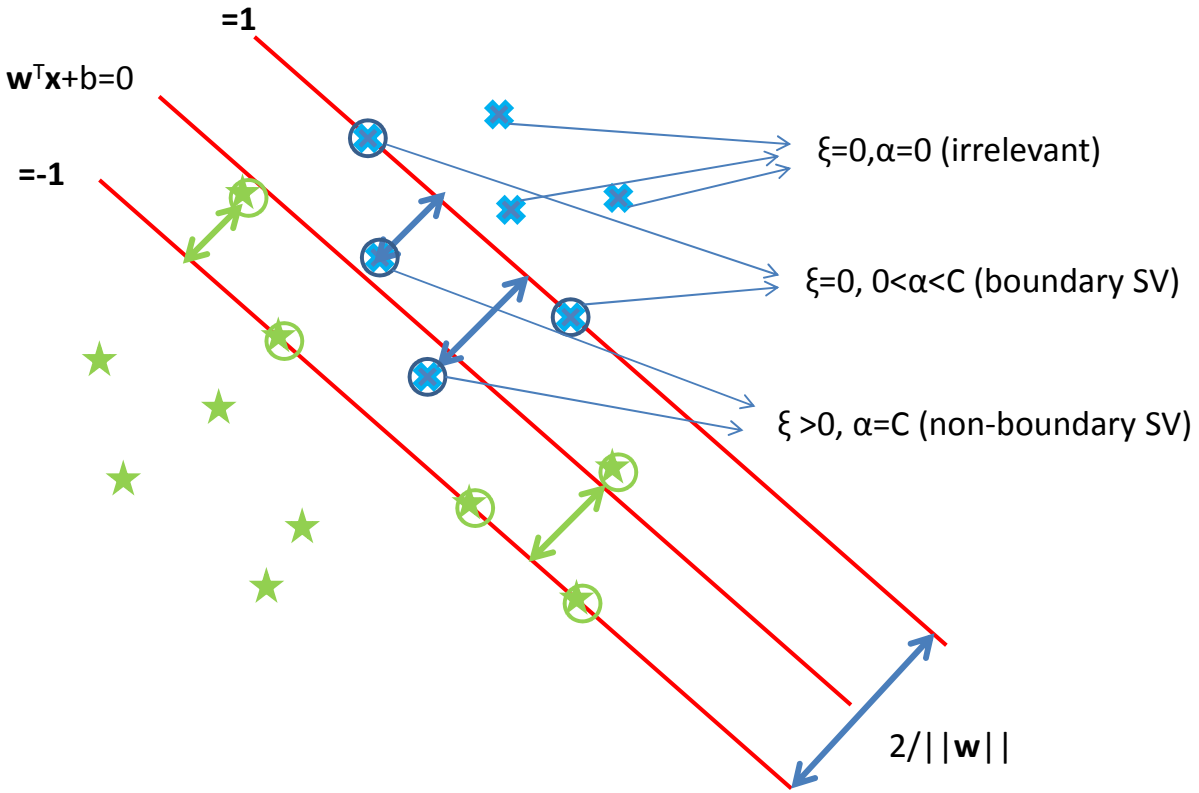
subject to

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \quad (4)$$

$$0 \geq \alpha_i \geq C \quad (5)$$

where α_i are the dual variables (Lagrange multipliers for each constraint, that is training sample). Optimal weight vector can be found by $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$

Support vector machines IV



- There is sparsity in α_i and most α_i turn out to be zero
- The $\mathbf{x}_i$ that correspond to nonzero α_i are called support vectors
- If $\alpha_i = C$, then $\mathbf{x}_i$ are non-boundary support vectors

- The support vectors are the only training samples that matter in determining the optimal weights (hence the separating hyperplane)

Support vector machines V

- To find optimal b, take any nonzero α_i and use the equation $\alpha_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1\right) = 0$ (comes from KKT conditions) or average values from all nonzero α_i's.

Regularized empirical risk (RER) minimization I

- Unifying framework for different linear classifiers
- Consider a predictor function $f_\theta(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^m$ that can be used to predict output labels y from features $\mathbf{x}$, m is the number of prediction values (discriminants)
  - For the linear binary classification problem, $m = 1$, $\theta = \tilde{\mathbf{w}}$ and $f_{\tilde{\mathbf{w}}}(\mathbf{x}) = \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}$
- Define a "loss function" $\mathcal{L} : \mathbb{R}^m \times \mathcal{Y} \to \mathbb{R}^+$ that measures the cost of prediction
- We define the expected risk as follows:

$$\mathcal{R}(\theta) = \int \mathcal{L}\left(f_\theta(\mathbf{x}), y\right) p(\mathbf{x}, y) d\mathbf{x} dy$$

Regularized empirical risk (RER) minimization II

- Since the joint distribution $p(\mathbf{x}, y)$ is usually unknown, we can find an estimate of the risk from the training data called the "empirical risk" which needs to be minimized wrt the parameters θ

$$\hat{\mathcal{R}}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(f_\theta(\mathbf{x}_i), y_i\right)$$

- Usually we add a penalty term $\Omega(\theta)$ for the parameters which penalizes complex (or large) predictor functions to arrive at what is called the "regularized empirical risk"

$$\hat{\mathcal{R}}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(f_\theta(\mathbf{x}_i), y_i\right) + \Omega(\theta)$$

Regularized empirical risk (RER) minimization III

- Note that "structured risk" minimization is similar where the regularization function is replaced by the Vapnik-Chervonenkis (VC) confidence of the predictor function (which is also a measure of complexity of the predictor)
- All previous methods (FLD, LR and SVM) can be seen as variants of regularized empirical risk minimization by utilization of a different loss function as we elaborate in the following slides

Loss functions I

- RER for linear binary classification is as follows:

$$\hat{\mathcal{R}}(\tilde{\mathbf{w}}) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}\left(\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}_i, y_i\right) + \Omega(\tilde{\mathbf{w}})$$

- Using different loss functions yield different methods for linear binary classification

LS loss

$$\mathcal{L}(\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}, y) = (\tilde{\mathbf{w}}^T\tilde{\mathbf{x}} - t_y)^2$$

- Here t_y are regression targets for each class in least squares regression. Using LS loss with no regularization is equivalent to FLD when $t_y = y\frac{N}{N_y}$ where N_y is the number of samples in class y [Bishop, 2006]
- LS-SVM uses $t_y = y$ and a quadratic regularizer [Suykens and Vandewalle, 1999]. Without regularization, LS-SVM is similar to FLD
- Regularized LDA [Friedman, 1989] is equivalent to using a regularization function in the RER framework

BNLL loss

$$\mathcal{L}(\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}, y) = \log\left(1 + \exp\left\{-y\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}\right\}\right)$$

- Using binomial negative log-likelihood (BNLL) loss function with no regularization is equivalent to performing a logistic regression classification

Hinge loss

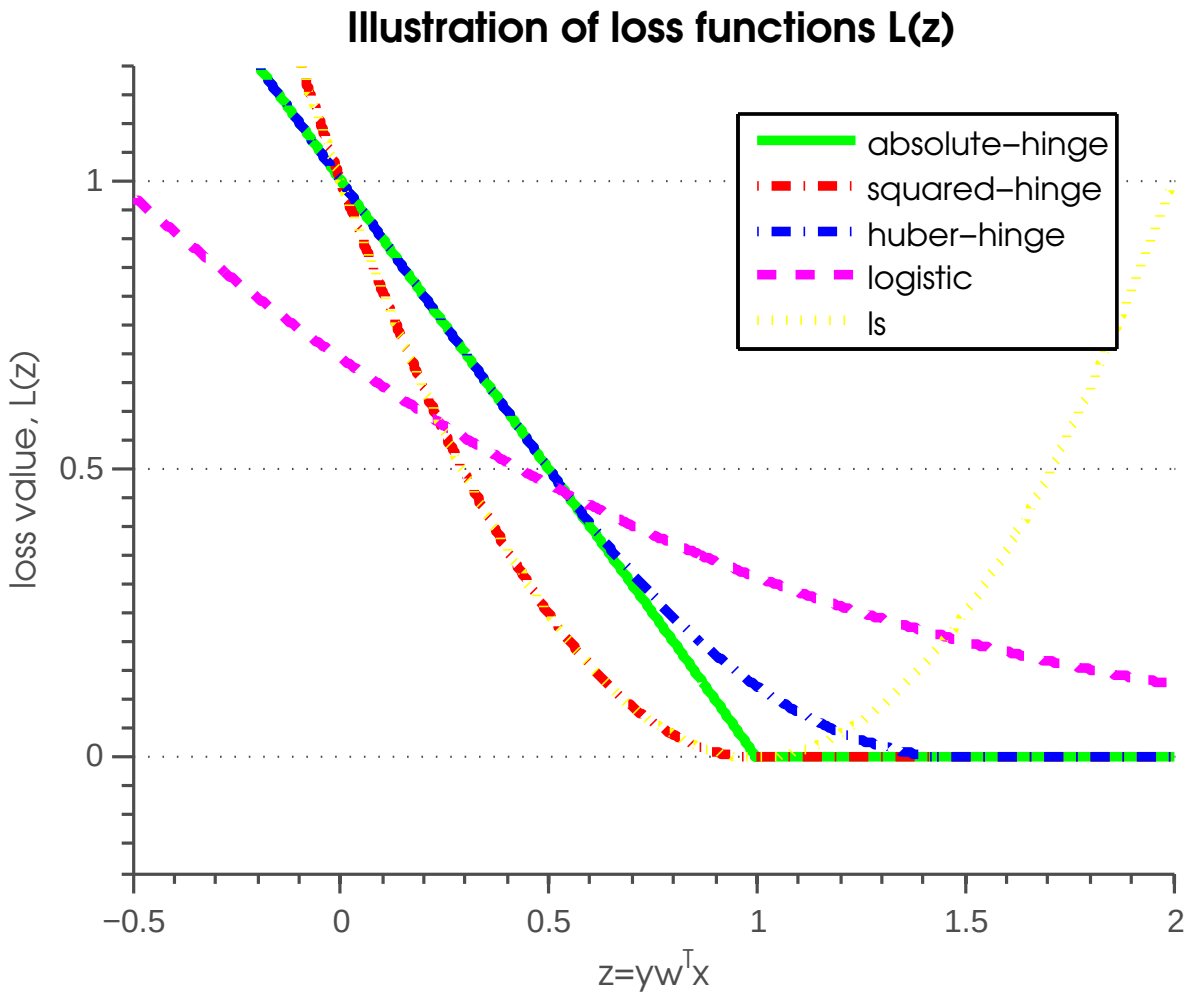
$$\mathcal{L}(\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}, y) = (1 - y\tilde{\mathbf{w}}^T\tilde{\mathbf{x}})_+$$

- Using hinge loss function and a regularization function $\Omega(\tilde{\mathbf{w}}) = \lambda\tilde{\mathbf{w}}^T\mathbf{D}\tilde{\mathbf{w}}$ (where $\mathbf{D}$ is a diagonal matrix with all ones in the diagonal except for the last entry which is zero) is equivalent to performing an SVM classification
- Note that $\Omega(\tilde{\mathbf{w}}) = \lambda\tilde{\mathbf{w}}^T\tilde{\mathbf{w}}$ is also used which is slightly different (in liblinear [Fan et al., 2008])

Other loss functions

- There are many other loss functions defined in the literature [LeCun et al., 2006]
- Huber-hinge loss is one which smooths the edge of the hinge loss so that we end up with a differentiable loss function
- Squared-hinge loss is the square of the hinge loss which is also differentiable (called L_2-SVM)

Loss function plots



Regularization functions I

- Although originally FLD and LR do not use regularization, it was shown to be beneficial in all cases
- Typically an L_2-norm regularization $\Omega(\tilde{\mathbf{w}}) = \lambda||\tilde{\mathbf{w}}||^2$ is used
- λ is a hyper-parameter that needs to be determined (next slide)
- Regularization helps in all classifiers arguably due to
  - improving test accuracy due to penalizing the complexity of the classifier
  - avoiding overtraining/overfitting
  - avoiding numerical problems in implementation.
- L_1-norm is used when sparse parameter vectors are desired
  - For example L_1-regularized L_1-SVM is considered in [Mangasarian, 2006].

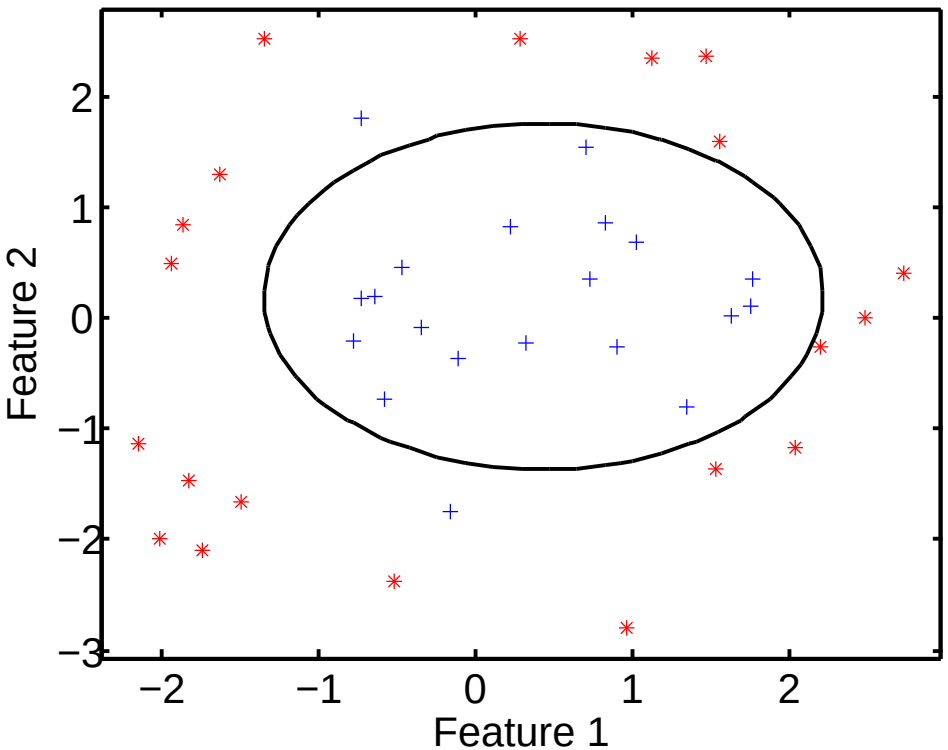
Estimating the hyperparameters I

- The hyperparameter λ can be found by grid search on validation data, that is by evaluating the performance of the trained classifier on the validation data
- Multiple fold cross-validation on the training data can also be performed for this purpose
- Bayesian "evidence framework" can be used to find the hyperparameters, this requires re-interpreting the empirical risk optimization as a maximum aposteriori probability (MAP) estimation problem [Hoffmann, 2007] - Bayesian LDA
- Perturbation analysis can be used [Zheng et al., 2009]
- Other ad-hoc methods exist as well

RER optimization methods

- LS and logistic losses are differentiable, so primal algorithms such as Newton's method work very well
- LS loss has closed form solution
- For logistic loss, Newton's method results in iterative re-weighted least squares (IRLS) formulation
- For hinge loss, one needs to go to the dual QP problem and solve it in the dual
- For squared-hinge or huber-hinge losses, one can solve it in the primal domain as well
- libsvm and liblinear are good libraries for solving RER formulations

How to obtain nonlinear classifiers? I



- Sometimes, class data are not separable by a line!
- Replace $\mathbf{x}$ with $\phi(\mathbf{x})$ which is a nonlinear mapping into a higher dimensional space
- If $\phi(\mathbf{x})$ is known explicity, you may use it instead of $\mathbf{x}$ to solve the problem

- Use the kernel trick to replace inner products with the kernel function: $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$

How to obtain nonlinear classifiers? II

- No need to know what $\phi(.)$ is, if we use the kernel trick (ϕ can be infinite dimensional) $\rightarrow$ just replace inner products with the kernel function
- Kernel versions usually solved in the dual, but it was shown it is possible to solve in the primal as well [Chapelle, 2007]
- We will not focus on kernel versions of these classifiers in this talk!
- linear classifiers are as good as or better than kernel versions for large scale learning [Fan et al., 2008]

Outline

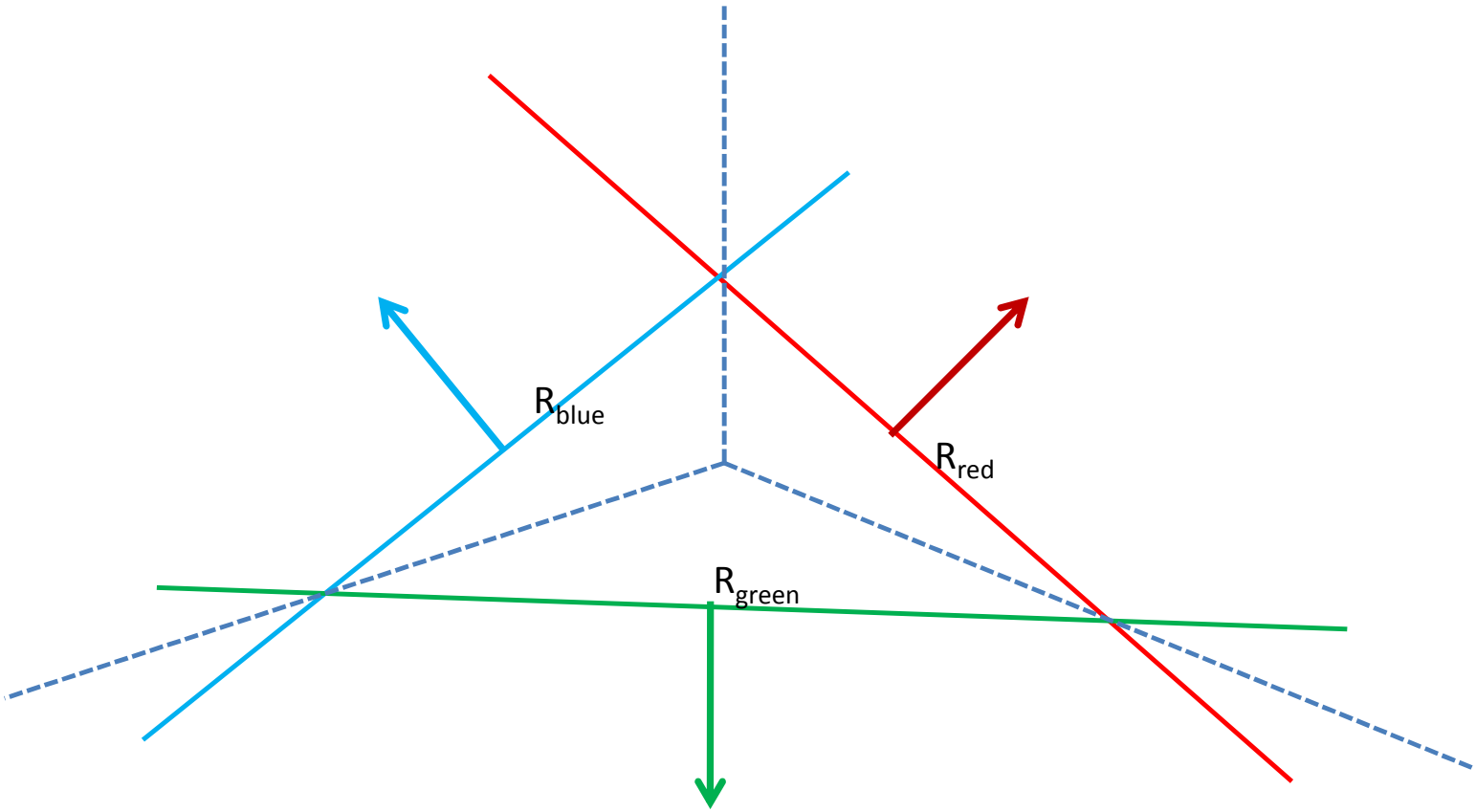
Multi-class classification

- Using multiple binary classifiers and combining them (multiple machine)
  - One-vs-all
  - One-vs-one
  - Error correcting output codes (ECOC)
- Direct multi-class classification (single machine) (explanation next slide)
- A paper compared these approaches and found that one-vs-one gave the best result (used in libsvm) [Hsu and Lin, 2002]
- For structured classification, we need direct multi-class classification since almost impossible to enumerate all possible output labels (topic of future lectures)

Linear multi-class classification

- Given training data $\{(\mathbf{x}_i, y_i) : i \in [N]\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathcal{Y} = [M]$
- $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,d}]^T$ are feature vectors, y_i are class identities
- Find a set of weight vectors $\tilde{\mathbf{w}}_y \in \mathbb{R}^{d+1}$ such that for test data $\mathbf{x}$, we obtain a score for class y as $\tilde{\mathbf{w}}_y^T \tilde{\mathbf{x}} = \mathbf{w}_y^T \mathbf{x} + b_y$
- Classification is done by $\hat{y} = \arg\max_y \tilde{\mathbf{w}}_y^T \tilde{\mathbf{x}}$
- Each $\tilde{\mathbf{w}}_y$ represents a hyperplane in $\mathbb{R}^{d+1}$

Geometry of linear multi-class discriminants



Generative multi-class classification I

- Given class conditional probability distributions $p(\mathbf{x}|y)$
- Obtain class posteriors as:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{\sum_{y'} p(\mathbf{x}|y')p(y')}$$

- If class conditional probability distributions are from the exponential family

$$p(\mathbf{x}|y;\boldsymbol{\theta}_y) = h(\mathbf{x}) \exp\left\{\boldsymbol{\theta}_y^T \mathbf{t}(\mathbf{x}) - A(\boldsymbol{\theta}_y)\right\}$$

where $\boldsymbol{\theta}$ are the parameters (or transformed parameters) of the pdf, $\mathbf{t}(\mathbf{x})$ is the sufficient statistics vector, $A(\boldsymbol{\theta})$ is the log-partition function and $h(\mathbf{x})$ is a base measure independent of the parameters.

Generative multi-class classification II

- Then, the conditional likelihood has the form:

$$p(y|\mathbf{x}) = \frac{\exp\{\tilde{\mathbf{w}}_y^T \phi(\mathbf{x})\}}{\sum_{y'} \exp\{\tilde{\mathbf{w}}_{y'}^T \phi(\mathbf{x})\}}$$

  where $\tilde{\mathbf{w}}_y = \left[\boldsymbol{\theta}_y^T, -A(\boldsymbol{\theta}_y) + log(p(y))\right]^T$ and $\phi(\mathbf{x}) = \left[\mathbf{t}(\mathbf{x})^T, 1\right]^T$
- This form of the conditional likelihood is called normalized exponential or *softmax* function.
- Hence, if we map the features $\mathbf{x}$ using the sufficient statistics and add a constant feature, conditional likelihood will have a log-linear form

Generative multi-class classification III

- For example, for the Gaussian setup with means $\boldsymbol{\mu}_y$ and equal covariances (known Σ) (multi-class FLD), we get the following

$$\begin{aligned} \boldsymbol{\theta}_y &= \boldsymbol{\mu}_y \\ \mathbf{t}(\mathbf{x}) &= \mathbf{x} \\ \mathbf{w}_y &= \Sigma^{-1}\mu_y \\ b_y &= -\frac{1}{2}\mu_y^T\Sigma^{-1}\mu_y + \log(p(y)) \end{aligned} \tag{6}$$

- Exercise: Perform the same analysis when both $\boldsymbol{\mu}_y$ and $\boldsymbol{\Sigma}_y$ are different for each class. What are the sufficient statistics and parameters then?

Multi-class (multinomial) logistic regression I

- Assuming $p(y|\mathbf{x}) = \frac{\exp\{\tilde{\mathbf{w}}_y^T \phi(\mathbf{x})\}}{\sum_{y'} \exp\{\tilde{\mathbf{w}}_{y'}^T \phi(\mathbf{x})\}}$ we can maximize the conditional log-likelihood of the training data

$$\log p(y_1, \ldots, y_N | \mathbf{x}_1, \ldots, \mathbf{x}_N) = \sum_{i=1}^{N} \log p(y_i | \mathbf{x}_i)$$

- This yields the following negative log likelihood (NLL) objective function to minimize

$$\sum_{i=1}^{N} -\tilde{\mathbf{w}}_{y_i}^T \phi(\mathbf{x}_i) + \log \left(\sum_{y'} \exp\{\tilde{\mathbf{w}}_{y'}^T \phi(\mathbf{x}_i)\} \right)$$

- This objective can be minimized using gradient-based techniques.

Multiclass SVM I

- Multiclass SVM was formulated in [Crammer and Singer, 2001] as follows
- First define $\mathbf{w} = [\mathbf{w}_1^T, \mathbf{w}_2^T, \ldots, \mathbf{w}_M^T]^T$ as concatenation of weight vectors.

$$\min \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i$$

subject to:

$$\begin{aligned} \tilde{\mathbf{w}}_{y_i}^T\tilde{\mathbf{x}}_i - \tilde{\mathbf{w}}_{y'}^T\tilde{\mathbf{x}}_i &\geq 1-\xi_i, \quad \forall i, y' \neq y_i \\ \xi_i &\geq 0 \end{aligned} \tag{7}$$

- Let $\bar{y}_i = \arg\max_{y' \neq y_i} \tilde{\mathbf{w}}_{y'}^T\tilde{\mathbf{x}}_i$

Multiclass SVM II

- The first set of inequalities can also be written as:

$$\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i - \tilde{\mathbf{w}}_{\bar{y}_i}^T \tilde{\mathbf{x}}_i \geq 1 - \xi_i, \quad \forall i$$

- The problem can be formulated as an unconstrained minimization problem as follows:

$$\min \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{N} (1 - \tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i + \max_{y' \neq y_i} \tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i)_+$$

- In multi-class classification, some class pairs may be less costly to be confused, so we can define a *label-loss* function $\Delta(y, y')$ which gives a cost to replacing y with y' during classification, which can be incorporated in the SVM constraints formulation [Tsochantaridis et al., 2005]

Multiclass SVM III

- "Margin rescaling" yields the set of constraints

$$\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i - \tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i \geq \Delta(y_i, y') - \xi_i, \quad \forall i, y' \neq y_i$$

- whereas "slack rescaling" yields the following set of constraints

$$\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i - \tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i \geq 1 - \frac{\xi_i}{\Delta(y_i, y')}, \quad \forall i, y' \neq y_i$$

- There is a cutting-plane algorithm in [Tsochantaridis et al., 2005] which solves a series of constrained optimization algorithms to solve these problems
- Furthermore, [Joachims et al., 2009] introduced 1-slack constraints, 1-slack formulation with margin rescaling uses the constraints

$$\frac{1}{N} \sum_{i=1}^{N} (\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i - \tilde{\mathbf{w}}_{y_i'}^T \tilde{\mathbf{x}}_i) \geq \frac{1}{N} \sum_{i=1}^{N} \Delta(y_i, y_i') - \xi, \quad \forall (y_i')_{i=1}^N$$

Multiclass SVM IV

- 1-slack formulation with slack rescaling is also provided in the same paper
- There are efficient cutting-plane algorithms in [Joachims et al., 2009] for solving 1-slack problems in the dual space
- These solvers are provided in the svm-struct package

Regularized empirical risk for multi-class I

- Remembering RER

$$\hat{\mathcal{R}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}\left(f_\theta(\mathbf{x}_i), y_i\right) + \Omega(\theta)$$

- For multi-class, the parameters are $\tilde{\mathbf{w}} = [\tilde{\mathbf{w}}_1^T, \ldots, \tilde{\mathbf{w}}_M^T]^T$ and the loss functions for each type are as follows:
- multi-class FLD:

$$\mathcal{L}\left(f_{\tilde{\mathbf{w}}}(\mathbf{x}_i), y_i\right) = \sum_{y=1}^{M}(\tilde{\mathbf{w}}_y^T\tilde{\mathbf{x}}_i - t(y, y_i))^2$$

where $t(y, y_i)$ are appropriate targets for class y for least squares regression when the true class is y_i

Regularized empirical risk for multi-class II

- [Ye, 2007] shows that using $t(y, y_i) = \sqrt{N/N_y} - \sqrt{N_y/N}$ when $y = y_i$ and $t(y, y_i) = -\sqrt{N_{y_i}/N}$ otherwise, is equivalent to linear discriminant analysis
- FLD is equivalent to using a least squares loss function
- multi-class LR:

$$\mathcal{L}\left(f_{\tilde{\mathbf{W}}}(\mathbf{x}_i), y_i\right) = -\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i + \log\left(\sum_{y'} \exp\{\tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i\}\right)$$

- multi-class SVM:

$$\mathcal{L}\left(f_{\tilde{\mathbf{W}}}(\mathbf{x}_i), y_i\right) = (1 - \tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i + \max_{y' \neq y_i} \tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i)_+$$

Regularized empirical risk for multi-class III

- Re-writing the SVM loss function is possible as follows:

$$\mathcal{L}\left(f_{\tilde{\mathbf{W}}}(\mathbf{x}_i), y_i\right) = -\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i + \max_{y'}(\tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i + 1 - \delta_{y,y_i})$$

- This form is equivalent to the one before since this is guaranteed to be nonnegative
- We can generalize using label-loss $\Delta(y, y')$ and "margin rescaling" introduced before to get

$$\mathcal{L}\left(f_{\tilde{\mathbf{W}}}(\mathbf{x}_i), y_i\right) = -\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i + \max_{y'}(\tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i + \Delta(y_i, y'))$$

- We can replace the max with a softmax (log-sum-exp) to get an "approximation" SVM-softmax

$$\mathcal{L}\left(f_{\tilde{\mathbf{W}}}(\mathbf{x}_i), y_i\right) = -\tilde{\mathbf{w}}_{y_i}^T \tilde{\mathbf{x}}_i + \log\left(\sum_{y'} \exp\{\tilde{\mathbf{w}}_{y'}^T \tilde{\mathbf{x}}_i + \Delta(y_i, y')\}\right)$$

Regularized empirical risk for multi-class IV

- Comparing the LR loss function with the SVM-softmax one, we see that they are similar except for the addition of the label-loss function to provide additional margin for confusable classes in the SVM-softmax
- It is possible to get the RER expression for "slack rescaling" and 1-slack versions of margin and slack rescaling as well
- For other possible loss functions, for example see [LeCun et al., 2006]

Optimization algorithms I

- Multi-class FLD has closed from solution
- Multi-class LR can be solved using Newton's method in the primal yielding an iterative re-weighted least squares algorithm
- Multi-class SVM can be directly solved from the dual problem [Crammer and Singer, 2001, Fan et al., 2008] especially if number of training samples N is small
- Cutting-plane algorithms are attractive alternatives [Tsochantaridis et al., 2005, Joachims et al., 2009]
- Multi-class SVM-softmax can be solved using Newton's method in the primal
- If Newton algorithm's Hessian is too big to be computed efficiently, L-BFGS can be used
- Stochastic gradient algorithms are fast and applicable, but need to be careful with convergence and choosing step sizes [Bottou, 2004]

Optimization algorithms II

- The optimal choice of the algorithm depends on the values of feature dimension d, number of classes M and number of training samples N

Log-linear models

- A general log-linear model is given by:

$$p(y|\mathbf{x};\mathbf{w}) = \frac{1}{Z(x,\mathbf{w})} \exp\left\{\sum_j w_j F_j(\mathbf{x},y)\right\}$$

where

$$Z(x,\mathbf{w}) = \sum_{y'} \exp\left\{\sum_j w_j F_j(\mathbf{x},y')\right\}$$

is called the partition function.

- Note that multi-class LR is a log-linear model where we define a concatenated weight vector $\mathbf{w} = [\tilde{\mathbf{w}}_1^T, \ldots, \tilde{\mathbf{w}}_M^T]^T$ and where the feature vector $F(\mathbf{x},y) = \tilde{\mathbf{x}} \otimes \mathbf{e}_y$ where $\mathbf{e}_y$ is the unit vector with one in position y and zeros elsewhere and $\otimes$ denotes the Kronecker product.

References I

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Leon Bottou. Stochastic learning. In Olivier Bousquet and Ulrike von Luxburg, editors, *Advanced Lectures on Machine Learning*. LNAI 3176, 2004.

Olivier Chapelle. Training a Support Vector Machine in the Primal. *Neural Computation*, 19(5):1155–1178, 2007.

K Crammer and Y Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–92, 2001.

R E Fan, K W Chang, C J Hsieh, X R Wang, and C J Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–74, 2008.

References II

J H Friedman. Regularized discriminant analysis. *J. Amer. Statistical Assoc.*, 84:165–175, 1989.

Ulrich Hoffmann. *Bayesian machine learning applied in a brain-computer interface for disabled users*. PhD thesis, EPFL, 2007.

C W Hsu and C J Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–25, 2002.

T Joachims, T Finley, and C N J Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, October 2009.

Yann LeCun, Sumit Chopra, Raia Hadsell, Ranzato Marc'Aurelio, and Fu-Jie Huang. A tutorial on Energy based learning. In G. Bakir, T. Hofman, B. Schlkopf, A. Smola, and B. Taskar, editors, *Predicting Structured Data*. MIT Press, 2006.

References III

Olvi L Mangasarian. Exact 1-Norm Support Vector Machines Via Unconstrained Convex Differentiable Minimization. *Journal of Machine Learning Research*, 7:1517–30, 2006.

J A K Suykens and J Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, June 1999.

I Tsochantaridis, T Joachims, T Hoffman, and Y Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–84, December 2005.

Jieping Ye. Least squares linear discriminant analysis. In *International Conference on Machine Learning*, volume 227, pages 1083–97, 2007.

W S Zheng, J H Lai, and P C Yuen. Perturbation LDA: learning the difference between the class empirical mean and its expectation. *Pattern Recognition*, 42(5):764–779, 2009.