On the Path to an Ideal ROC Curve: Considering Cost Asymmetry in Learning Classifiers

Francis Bach David Heckerman Eric Horvitz

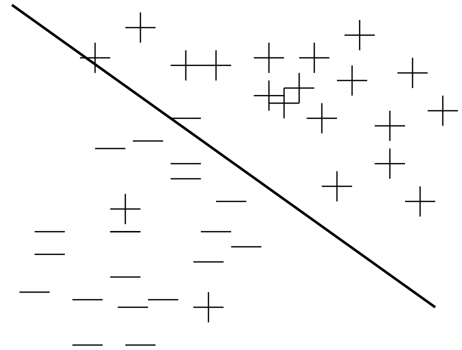
UC Berkeley *Microsoft Research*

January 2005

Outline

- Asymmetric testing cost and ROC analysis
- Training linear classifiers
- Efficient algorithm to vary the training cost asymmetry
- Mismatch between training and testing asymmetries

Linear classification



- Input: $x \in \mathbb{R}^d$
- Output: labels $y \in \{-1, +1\}$
- Linear classifiers - two parameters (w, b): $f(x) = \text{sign}(w^\top x + b)$
  - w : slope
  - b : intercept
- Straightforward extension to **non linear classification** using **kernels**

Asymmetric utility

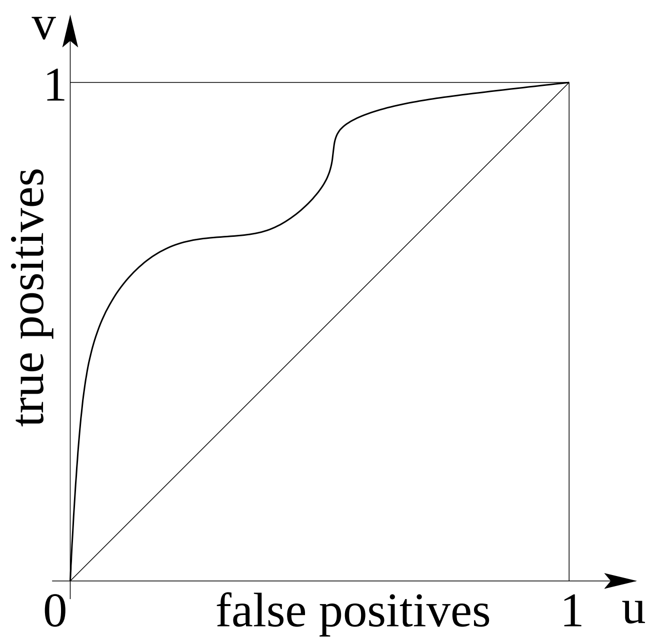
- Two types of errors:
  - false positives: $y = -1$, $f(x) = 1$
  - false negatives: $y = 1$, $f(x) = -1$
- Asymmetric user utility function with two parameters (C_+, C_-):

Correct classification :	0
False positive :	$C_- > 0$
False negative :	$C_+ > 0$

- Definition: **assymetry** $= C_+/(C_+ + C_-)$
- Example: junk mail filtering
- ROC curves: display performance of a set of classifiers for all possible asymmetries

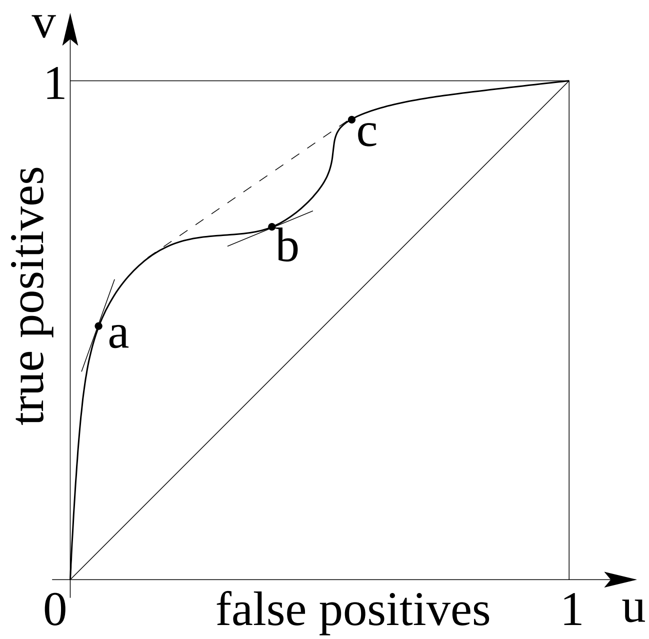
ROC curves

- ROC plane (u, v)
- u = proportion of false positives $= P(f(x) = 1|y = -1)$
- v = proportion of true positives $= P(f(x) = 1|y = 1)$
- Plot a set of classifiers $f_\gamma(x)$ for $\gamma \in \mathbb{R}$



ROC curves and convex envelopes

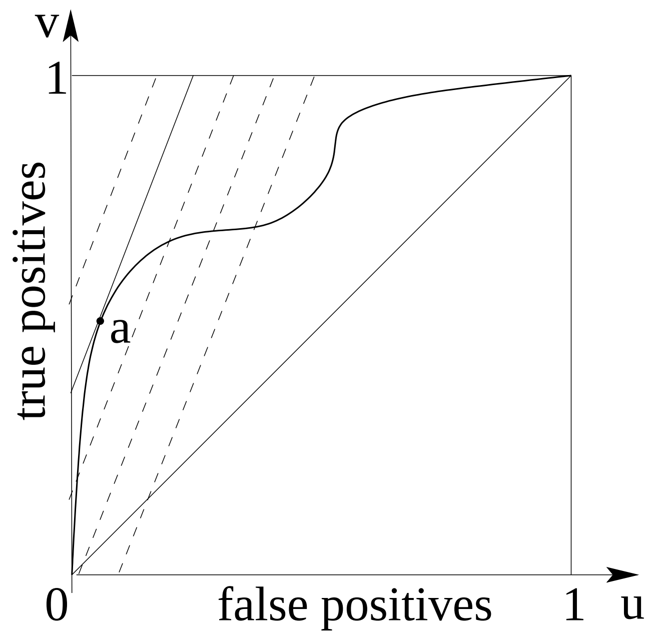
- Any point on the upper convex envelope can be achieved
- Definition: $(u(\gamma), v(\gamma))$ **ROC-consistent** iff it lies on the upper convex envelope of the ROC curve



Reading out performance from ROC curves

- **Given the user (testing) asymmetry** β, find the best γ
  - β defines a direction in the ROC plane
  - finds the most upper left tangent point

- **Given** γ, find the best testing asymmetry β
  - Only relevant for ROC consistent points: $\beta(\gamma) = \frac{1}{1+\frac{p_+}{p_-}\frac{dv}{d\gamma}(\gamma)/\frac{du}{d\gamma}(\gamma)}$



Training linear classifiers

- User cost (testing) : $R(C_+, C_-, w, b)$

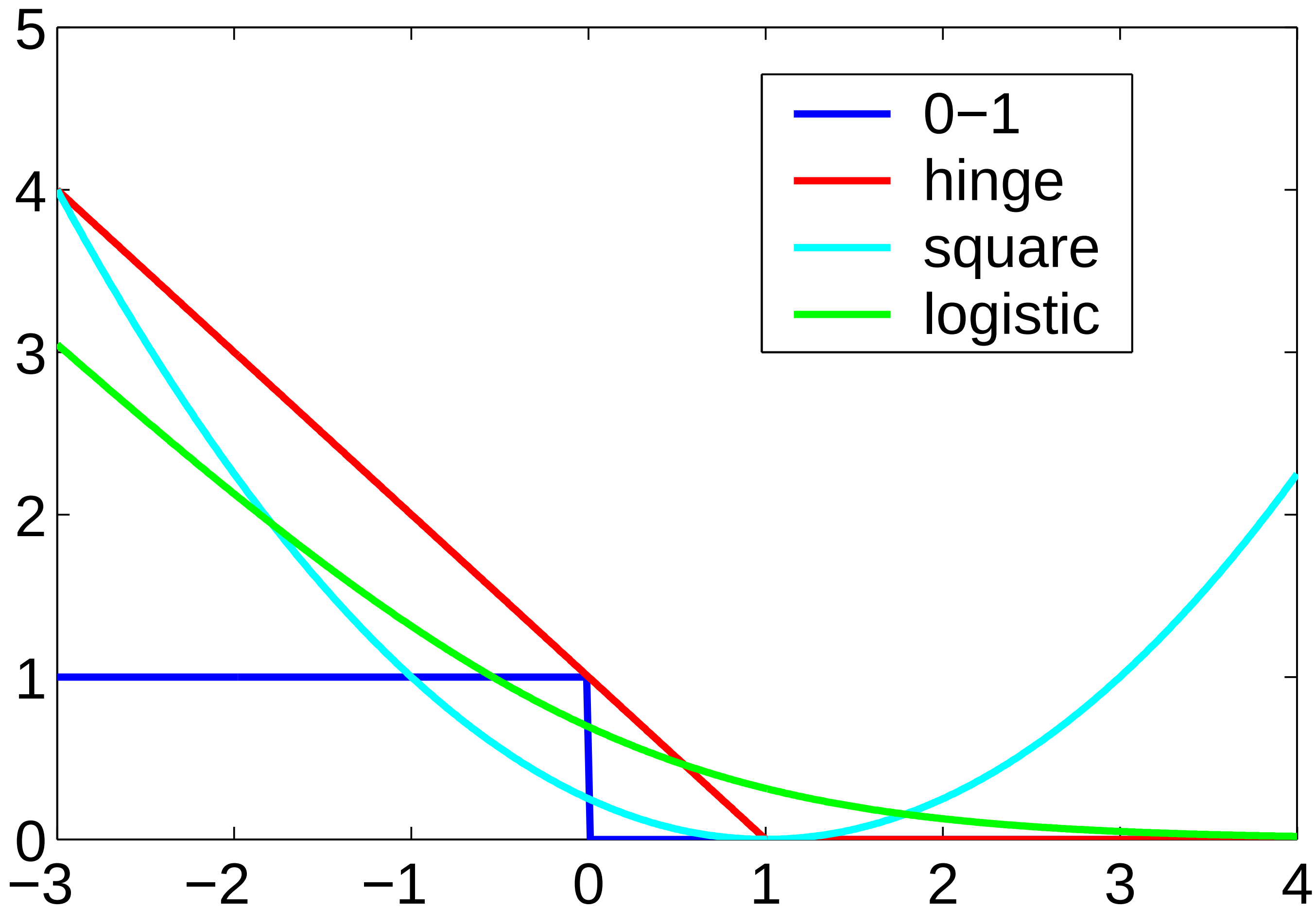
$$
\begin{aligned}
&= C_+ P\{w^\top x + b < 0,\ y = 1\} + C_- P\{w^\top x + b \geqslant 0,\ y = -1\} \\
&= C_+ E\{1_{y=1}\phi_{0-1}(w^\top x + b)\} + C_- E\{1_{y=-1}\phi_{0-1}(-w^\top x - b)\}
\end{aligned}
$$

ϕ_{0-1} = "0-1 loss" (step function): 1 for negative values, 0 otherwise

- Training cost using convex surrogate: $R_\phi(C_+, C_-, w, b)$

$$
= C_+ E\{1_{y=1}\phi(w^\top x + b)\} + C_- E\{1_{y=-1}\phi(-w^\top x - b)\}
$$

Loss functions



Building ROC curves for linear classifiers

- Usual method:
  - train *once* with a given asymmetry $\gamma \in (0,1) \to w, b$
  - hold the slope w fixed
  - vary the intercept b from $-\infty$ to $+\infty$

- Proposed method:
  - train for all possible asymmetries $\gamma \in (0,1) \to w(\gamma), b(\gamma)$
  - should perform better than not optimizing w
  - if also varying b, it strictly includes the usual one $\Rightarrow$ must perform better

Building ROC curves for linear classifiers

- Usual method:
  - train *once* with a given asymmetry $\gamma \in (0, 1) \rightarrow w, b$
  - hold the slope w fixed
  - vary the intercept b from $-\infty$ to $+\infty$
- Proposed method:
  - train for all possible asymmetries $\gamma \in (0, 1) \rightarrow w(\gamma), b(\gamma)$
  - should perform better than not optimizing w
  - if also varying b, it strictly includes the usual one $\Rightarrow$ must perform better
- Computational feasibility ?
- Links between training asymmetry and testing asymmetry ?

Training data and regularization

- Regularized empirical training cost $\widehat{R}_\phi(C_+, C_-, w, b)$

$$= \frac{C_+}{n} \sum_{i \in \mathcal{I}_+} \phi(y_i(w^\top x_i + b)) + \frac{C_-}{n} \sum_{i \in \mathcal{I}_-} \phi(y_i(w^\top x_i + b)) + \frac{1}{2n} ||w||^2$$

  $\mathcal{I}_+$ positive examples, $\mathcal{I}_-$ negative examples,

- Two different effects in training:
  - Asymmetry $C_+/(C_- + C_+)$
  - Total amount of regularization $1/(C_+ + C_-)$

- Simplification: $(C_+ + C_-)$ held fixed to the best value for a particular asymmetry

Building paths of linear classifiers for the SVM

- SVM corresponds to hinge loss $\phi(u) = \max\{0, 1-u\}$
- Usual formulation:

$$\min_{w,b,\xi} \; C_+ \sum_{i \in \mathcal{I}_+} \xi_i + C_- \sum_{i \in \mathcal{I}_-} \xi_i + \frac{1}{2}||w||^2 \;\; \text{s.t.} \;\; \forall i, \; \xi_i \geqslant 0, \qquad \forall i, \; \xi_i \geqslant 1 - y_i(w^\top x_i + b)$$

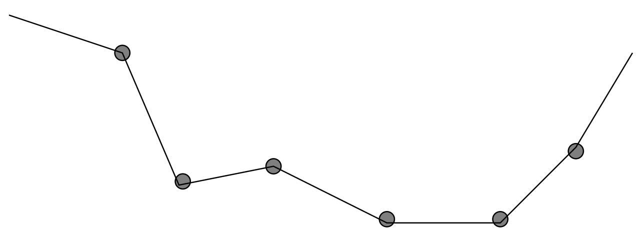
- **Goal** : follow optimal solution along lines in the (C_+, C_-)-plane
- Path following method:

1. Find (C_+, C_-) for which the solution is trivial to find
2. Use efficient path following technique

Path following for the SVM

- **Proposition** — extension of recent result by Hastie et al (2004): $(C_+, C_-) \mapsto (w, b)$ is piecewise linear

- **Corollary**: following paths of solutions along straight lines in the (C_+, C_-)-plane is computationally feasible.

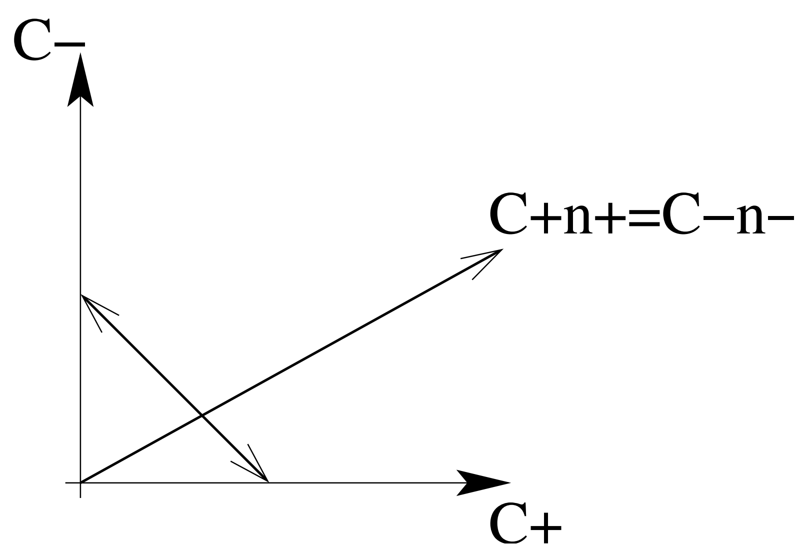


- Path following algorithm:
  - Follow a straight line in the (w, b)-space until a kink
  - Once at a kink, compute the new direction

Building paths of linear classifiers for the SVM

- Initialization:
  - Original method of Hastie et al requires “balanced data” ($C_+ n_+ = C_- n_-$) for simple initialization
  - We allow the ratio C_+/C_- to vary $\Rightarrow$ always possible

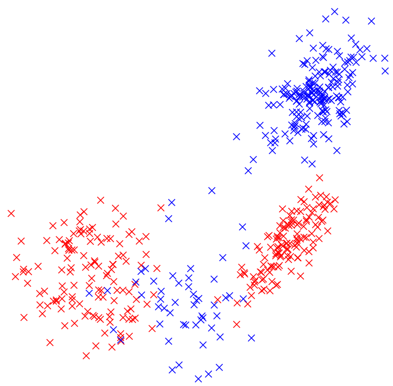
- Exploring the (C_+, C_-)-plane

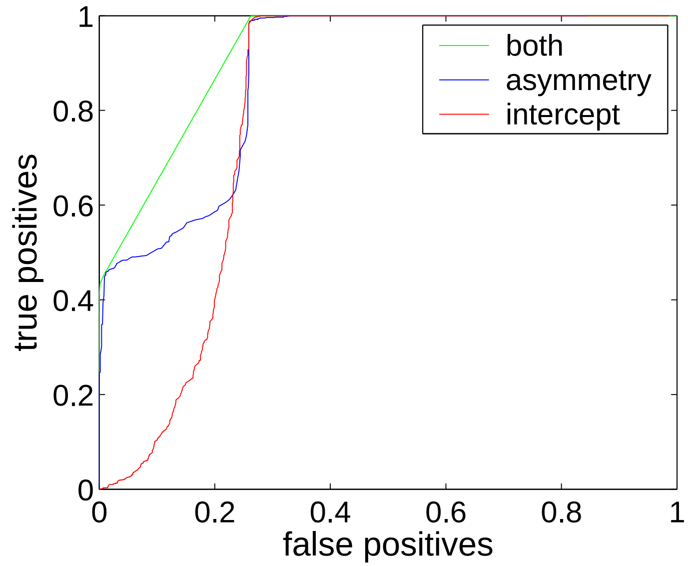


Computational complexity

- n number of data points, m number of support vectors
- Complexity of each step $O(mn + m^2)$
- Number of kinks along a straight line empirically $O(n)$
- Total empirical complexity is $O(mn^2 + m^2n)$ for the entire path
- Similar to SMO for a single point

ROC curves

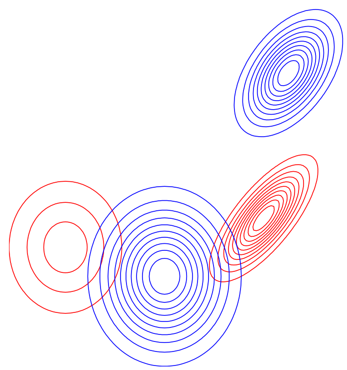


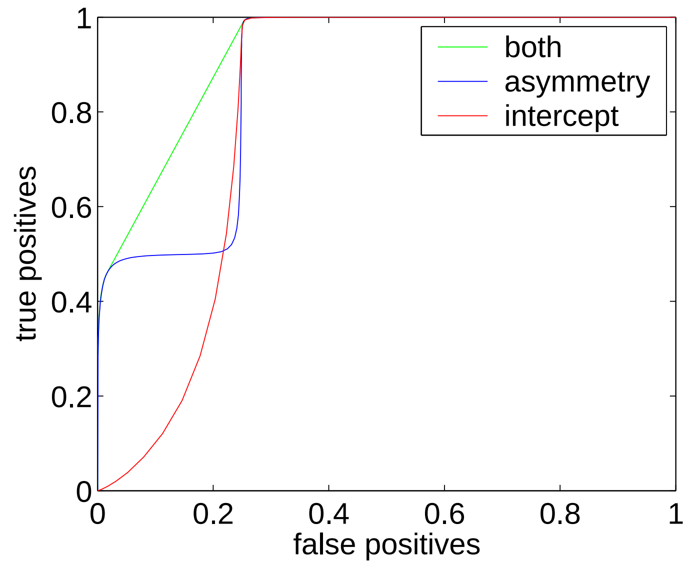


- Varying the asymmetry does not always perform better than varying the intercept

- Some points are ROC inconsistent when varying the asymmetry

ROC curves - population densities





- ⇒ empirical mismatch between training and testing asymmetries
  - Not a small sample effect
  - Due to the use of a convex surrogate to the 0-1 loss

Training and optimal testing asymmetries

- Population case (infinite sample) $\Rightarrow$ no need for regularization
- One-dimensional ROC curve $u(\gamma), v(\gamma)$ parameterized by training asymmetry γ
- For each γ, there exists one optimal testing asymmetry

$$\beta(\gamma) = \frac{1}{1 + \frac{p_+}{p_-}\frac{dv}{d\gamma}(\gamma) / \frac{du}{d\gamma}(\gamma)}$$

- $\beta(\gamma)$ **is different from** γ
  - Characterization around extreme asymmetries $\gamma = 0$ or 1

Characterization around extreme asymmetries

- Requires asymptotic expansion of $\beta(\gamma)$ around $\gamma = 0$

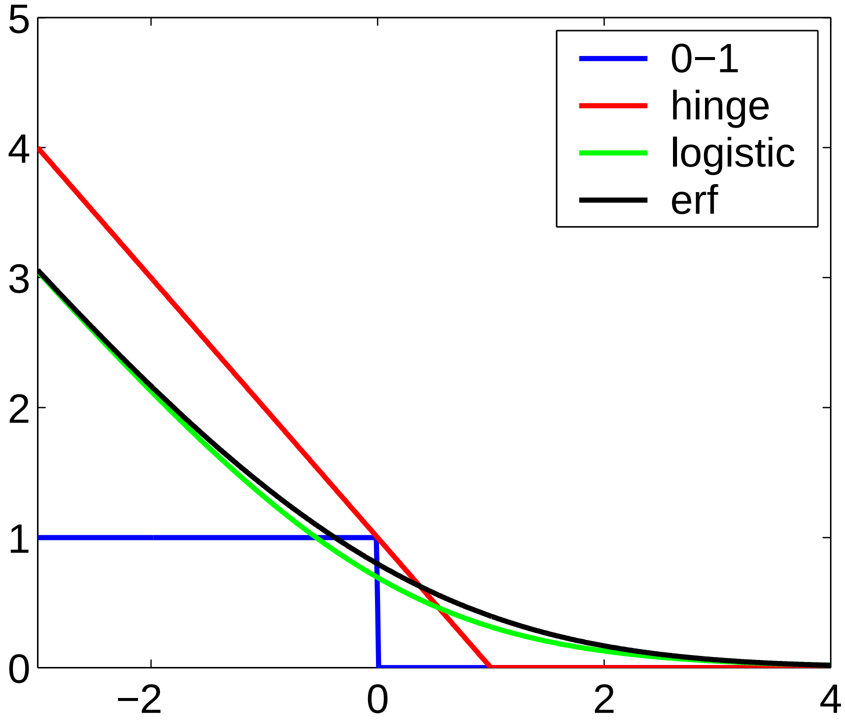
- Expansion can be done in semi-closed form when
  - class-conditional densities are mixtures of Gaussians
  - the loss functions are the square loss and the erf loss

- **erf loss**: $\phi_{erf}(u) = 2\left[\frac{u}{2}\psi\left(\frac{u}{2}\right) - \frac{u}{2} + \psi'\left(\frac{u}{2}\right)\right]$, where ψ is the cumulative distribution of the standard normal distribution, a.k.a the erf function.

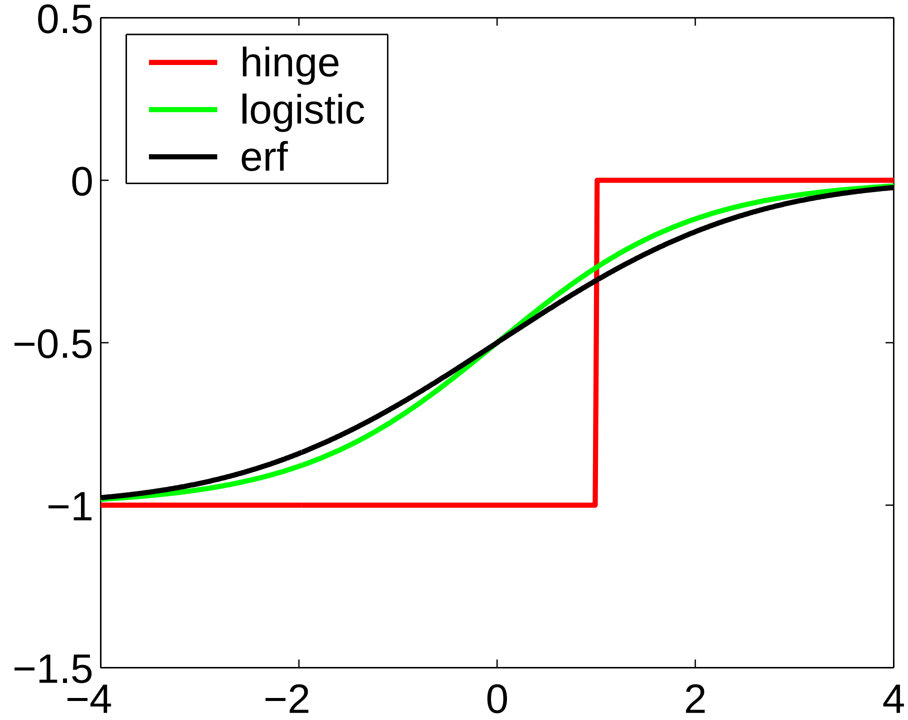
$$\psi(v) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{v} e^{-t^2/2} dt$$

- the erf loss is a close approximation to the *logistic loss* $\log(1 + e^{-u})$

erf loss



Loss functions



Derivatives

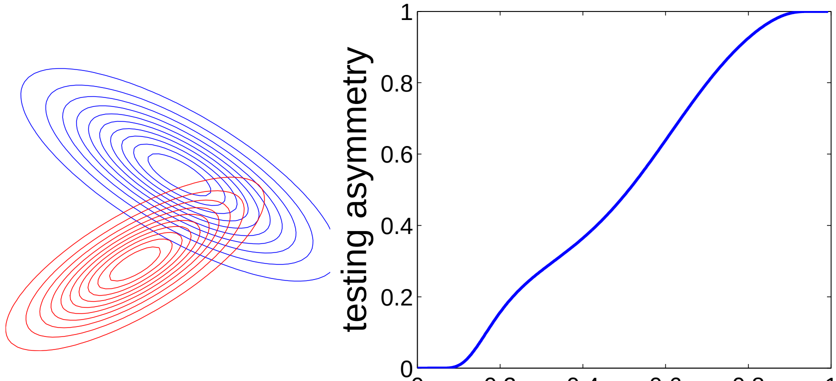
Gaussian densities - square loss

- Notations:
  - $P(y = \pm 1) = p_\pm$,
  - Given $y = \pm 1$, x is normal with mean $\mu_\pm$ and covariance $\Sigma_\pm$
- Proof: square loss $\Leftrightarrow$ linear regression
- Expansion:

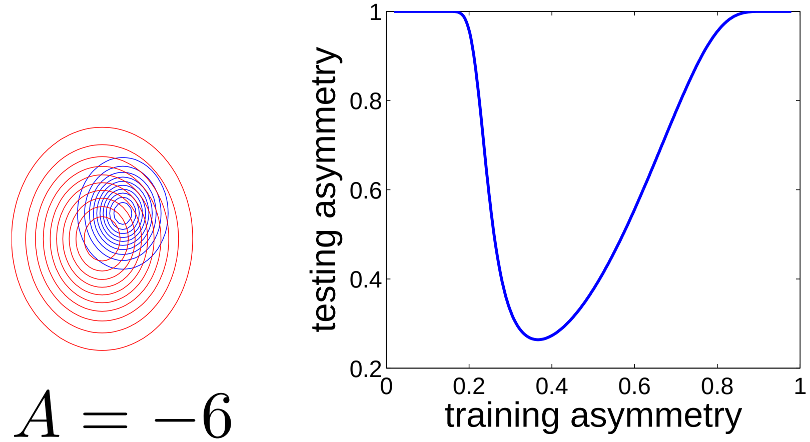
$$\log\left(\frac{p_-}{p+}(\beta(\gamma)^{-1}-1)\right) \approx \frac{p_-^2}{8p_+^2\gamma^2}\left(\frac{1}{m^\top\Sigma_-^{-1}m} - \frac{1}{m^\top\Sigma_-^{-1}\Sigma_+\Sigma_-^{-1}m}\right)$$

- Behavior depends on sign of $A = \left(\frac{1}{m^\top\Sigma_-^{-1}m} - \frac{1}{m^\top\Sigma_-^{-1}\Sigma_+\Sigma_-^{-1}m}\right)$

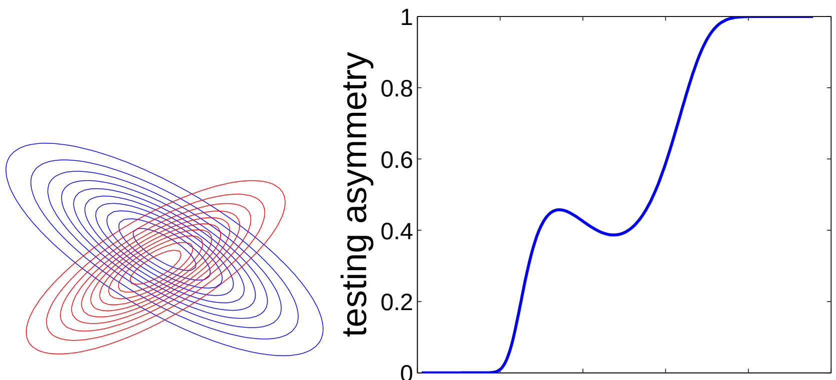
Square loss - Gaussian densities



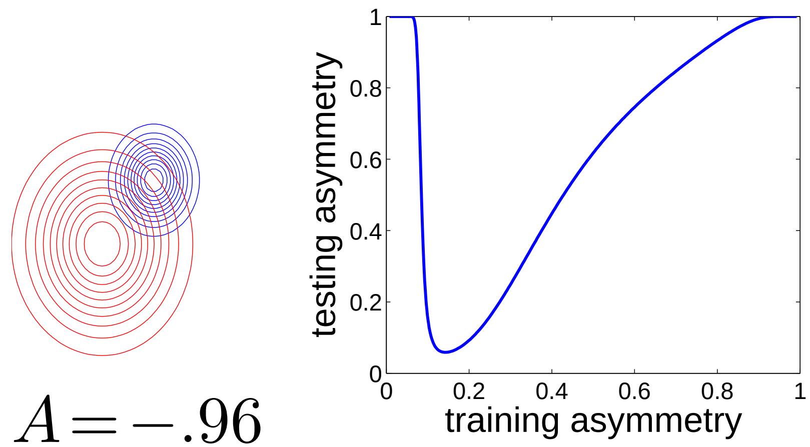
$A = .12$



$A = -6$



$A = 3$



$A = -.96$

$$\log\left(\frac{p_-}{p+}(\beta(\gamma)^{-1}-1)\right) \approx A\frac{p_-^2}{8p_+^2} \times \frac{1}{\gamma^2}$$

(training asymmetry $= \gamma$, testing asymmetry $= \beta$)

Gaussian densities - erf loss

- Notations:
  - $P(y = \pm 1) = p_\pm$,
  - Given $y = \pm 1$, x is normal with mean $\mu_\pm$ and covariance $\Sigma_\pm$

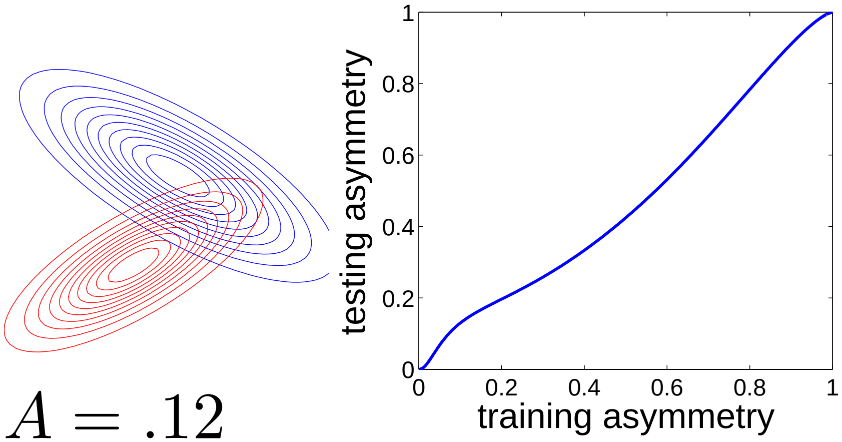
- Proof: write down the optimality conditions and compute...

- Expansion:

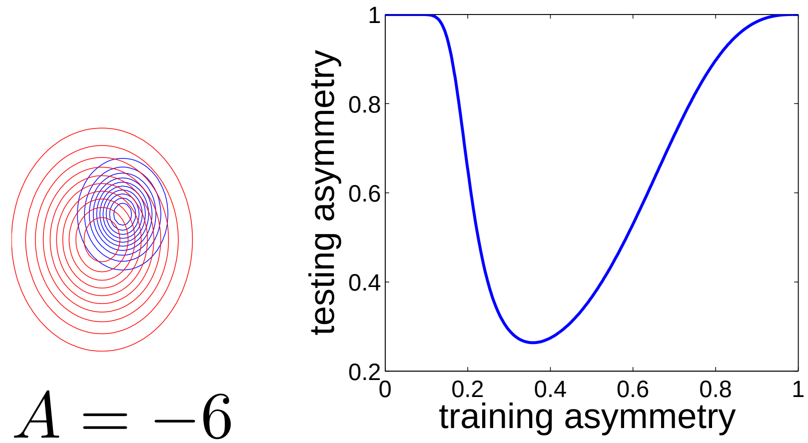
$$\log\left(\frac{p_-}{p+}(\beta(\gamma)^{-1} - 1)\right) \approx 2\log(1/\gamma)\left(\frac{1}{m^\top \Sigma_-^{-1} m} - \frac{1}{m^\top \Sigma_-^{-1}\Sigma_+\Sigma_-^{-1} m}\right)$$

- Behavior depends on sign of $A = \left(\frac{1}{m^\top \Sigma_-^{-1} m} - \frac{1}{m^\top \Sigma_-^{-1}\Sigma_+\Sigma_-^{-1} m}\right)$

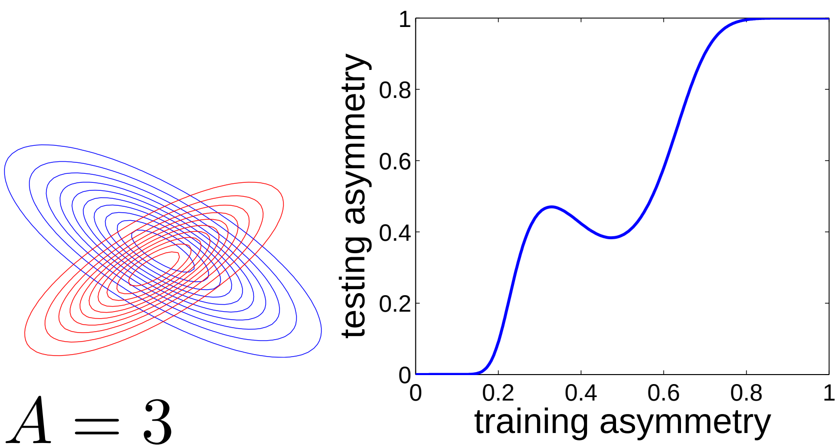
Erf loss - Gaussian densities



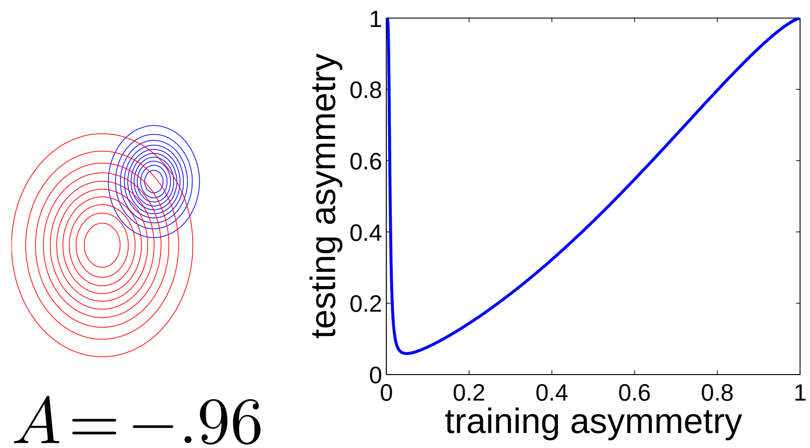
$A = .12$



$A = -6$



$A = 3$



$A = -.96$

$$\log\left(\frac{p_-}{p+}(\beta(\gamma)^{-1}-1)\right) \approx 2A\log(1/\gamma)$$

(training asymmetry $= \gamma$, testing asymmetry $= \beta$)

Results for mixtures of Gaussians

- Qualitatively similar:
  - to the first order, phase transition
  - test available given the class-conditional densities
- For details see the paper and the technical report

Empirical study of the mismatch

- Mismatch between training and testing asymmetries
  - quantifiable for extreme asymmetries

- Given one desired testing cost asymmetry, which training asymmetry?
  - currently no rule of thumb, but ...
  - ... one can try all of them (if it is efficient)

Maximal discrepancies

- For each dataset, compute the asymmetry γ for which performance is most different
- Performance measured by 10 fold cross validation

Dataset	γ	one asym.	all asym.
Pima	0.68	41 ± 0.4	22 ± 1
Breast	0.99	0.9 ± 0.03	0.09 ± 0.04
Ionosphere	0.82	10 ± 0.5	4 ± 0.8
Liver	0.32	27 ± 1.8	23.8 ± 0.02
Ringnorm	0.94	6.3 ± 0.06	4.3 ± 0.1
Twonorm	0.16	15 ± 0.2	1.2 ± 0.2
Adult	0.70	12.8 ± 0.8	11.5 ± 0.3

Conclusion

- Efficient algorithm to compute the solutions of the SVM for many cost asymmetries
- Allow to build better ROC curves
- Mismatch between training and testing asymmetries due to convex surrogate to the 0-1 loss
- Future work:
  - Theoretical analysis: extend to other losses
  - Algorithm: path following extended to multiple kernel learning