LDA, Logistic Regression and Separating Hyperplanes

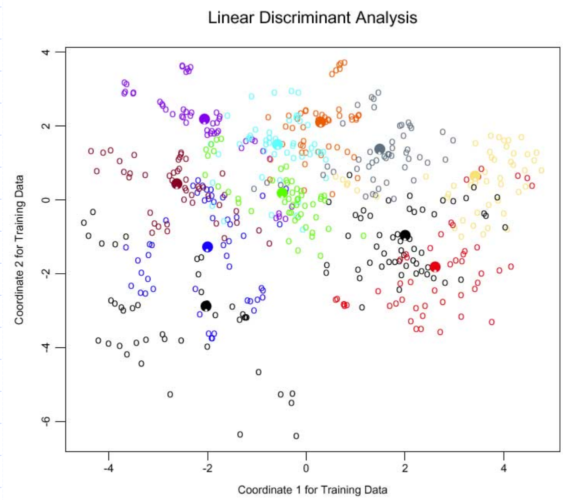
Randy Julian
Lilly Research Laboratories

Why study Logistic Regression

- LDA is not robust to gross outliers
- Logisitic models fit a sigmodial function, not a linear function - outliers are down weighted
- Logistic regression is 'safer' and more 'robust' with only 30% loss in efficiency
- Neural Networks and Support Vector Machines are generalized parallel logistic regression methods

[1]

Example: 11 Classes, 10 Dimensions



[1]

Some performance metrics

Technique	Error Rates	
	Training	Test
Linear Regression	0.48	0.67
LDA	0.32	0.56
Logistic Regression	0.22	0.51

[1]

Another look at LDA

- The decision boundary between Gaussian distributions with identical covariance matrices are linear.
- LDA assumes this underlying form and computes the linear boundaries as if it were true.
- It is parametric: estimating parameters of the Gaussian with a common covariance matrix from the data.

[1]

LDA formula (another form)

$$G(x) = \arg\max_k \delta_k(x)$$

$$\delta_k(x) = \mathbf{x}^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_k - \frac{1}{2} \boldsymbol{\mu}_k{}^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_k + \log \pi_k$$

priors $\hat{\pi}_k = \frac{N_k}{N}$

means $\hat{\mu}_k = \sum_{g_i = k} \frac{x_i}{N_K}$

covariance $\hat{\mathbf{\Sigma}} = \sum_{k=1}^{K} \sum_{g_i = k} \frac{(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T}{N - K}$

[1]

LDA for two classes:

$$\log \frac{p(G=k \mid X=x)}{p(G=l \mid X=x)} = \log \frac{f_k(x)}{f_l(x)} + \log \frac{\pi_k}{\pi_l}$$

$$\log \frac{p(G=k \mid X=x)}{p(G=l \mid X=x)} = \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\boldsymbol{\mu}_k + \boldsymbol{\mu}_l)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_k - \boldsymbol{\mu}_l) + \mathbf{x}^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_k - \boldsymbol{\mu}_l)$$

$$= \alpha_{k0} + \alpha_k{}^T \mathbf{x}$$

Linearity is a consequence of Gaussian assumption for class densities and assumption of a common covariance matrix.

[1]

Logistic Regression for K classes

$$\log \frac{p(G=1 \mid X=x)}{p(G=K \mid X=x)} = \beta_{10} + \beta_1{}^T \mathbf{x}$$

$$\log \frac{p(G=2 \mid X=x)}{p(G=K \mid X=x)} = \beta_{20} + \beta_2{}^T \mathbf{x}$$

$$\vdots$$

$$\log \frac{p(G=K-1 \mid X=x)}{p(G=K \mid X=x)} = \beta_{(K-1)0} + \beta_{(K-1)}{}^T \mathbf{x}$$

[1]

Maximum likelihood fitting

- Fit using the conditional likelihood of G given X.
- p(G|X) completely specifies the conditional distribution.
- Use the multinomial distribution as the model
  - Multidimensional member of "binomial" family

$$p_{g_i}(x_i,\theta) \equiv p(G=k \mid X=x_i;\theta)$$

$$l(\theta)=\sum_{i=1}^{N}\log p_{g_i}(x_i,\theta)$$

[1]

Logistic fit for two classes

$$l(\beta)=\sum_{i=1}^{N}\{y_i \log p(x_i,\beta)+(1-y_i)\log(1-p(x_i,\beta))\}$$

$$=\sum_{i=1}^{N}\left\{y_i\beta^T-\log\left(1+e^{\beta^T x_i}\right)\right\}$$

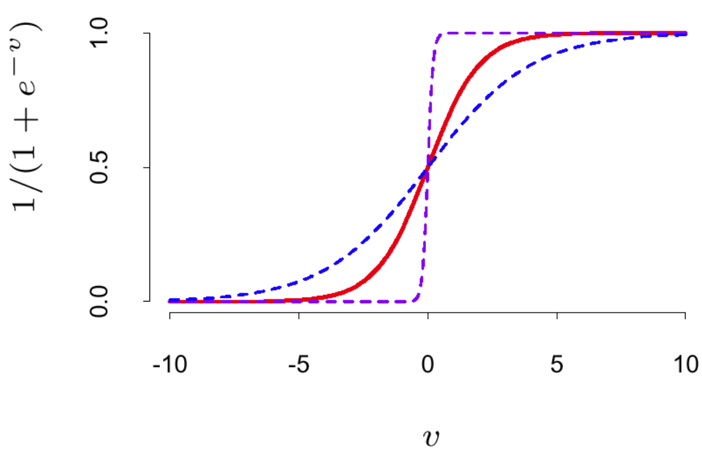
Note that this is non-linear in the parameter β. That means to fit this needs a non-linear regression.

R: glm()

```
method:     the method to be used in fitting the model.
            The default (and presently only) method
            glm.fit' uses iteratively reweighted least
            squares (IWLS).
```

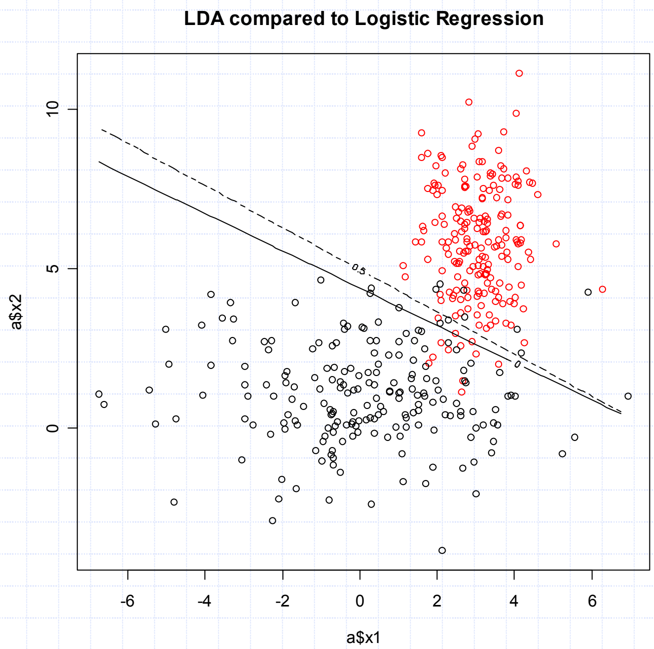
[1]

Basic shape of the sigmoid function



In practice

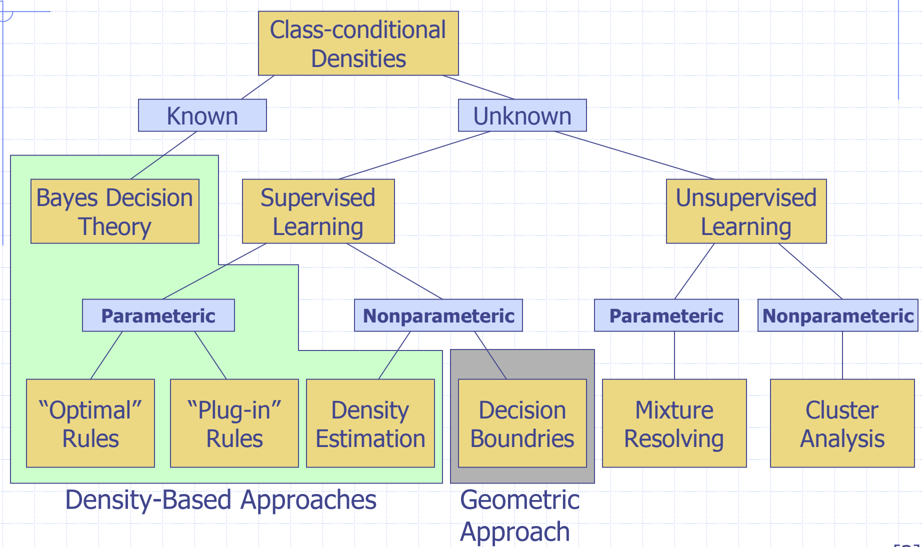
```
g <- lda( y ~ x1 + x2 , data = a)
Z <- predict(g, Xcon)
zp <- Z$post[,1] - Z$post[,2]
contour(x,y,matrix(zp,length(x),length(y)), add=T,
levels=0, labex = 0)
title("LDA compared to Logistic Regression")


g.g <- glm( y ~ x1 + x2, data = a, family=binomial(logit))
Z.g <- predict(g.g, Xcon)
contour(x,y,matrix(Z.g,length(x),length(y)), add=T, lty=2,
       levels=0.5, labex = 0)
```



LDA/Logistic Regression Summary

- Logistic Regression is considered more robust than LDA for classification
  - It makes fewer assumptions
  - Not as sensitive out gross outliers
- glm() is the recommended approach for standard linear classifiers
  - Recommended over lm() and lda()
- Canonical variates still a very important exploratory concept
- glm() uses an iterative non-linear regression: might not converge, is slower.

Various approaches

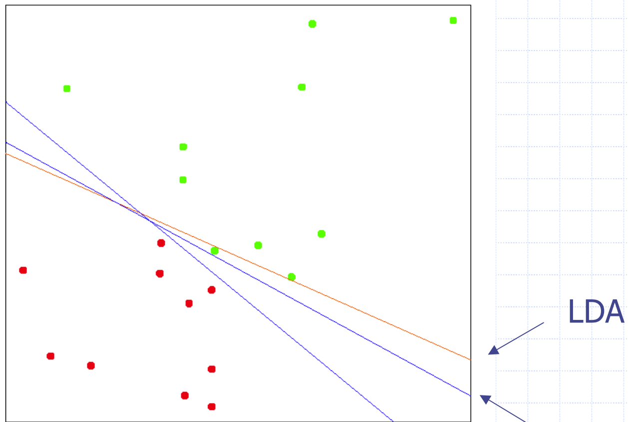


[3]

Separating hyperplanes

- Explicit attempt to construct a boundary
- Starting point for multilayer networks
- Basis for SVM
- Iterative linear method

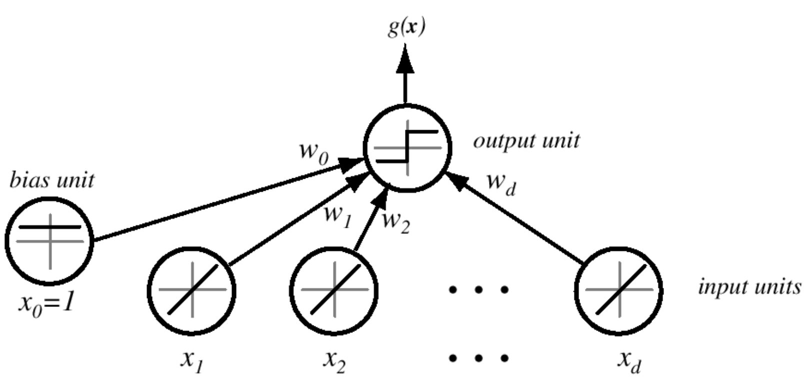
Hyperplanes compared to LDA



$$\{x : \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 = 0\}$$

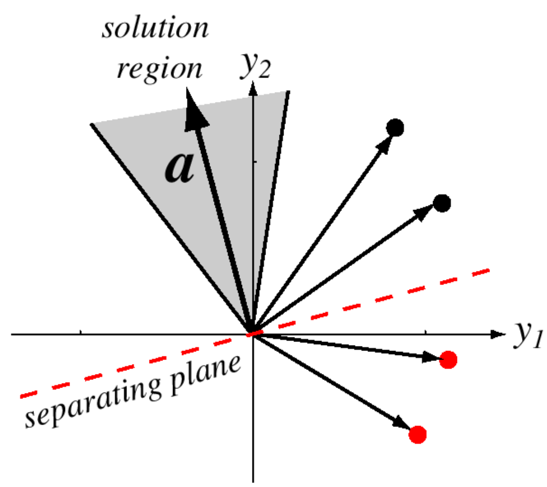
[1]

Basic perceptron Design



[4]

Starting point



[4]

Rosenblatt's Perceptron Learning Algorithm

Function to be minimized:

$$D(\beta, \beta_0) = -\sum_{i \in M} y_i (x_i^T \beta + \beta_0)$$

M = index of misclassified points

Gradient:

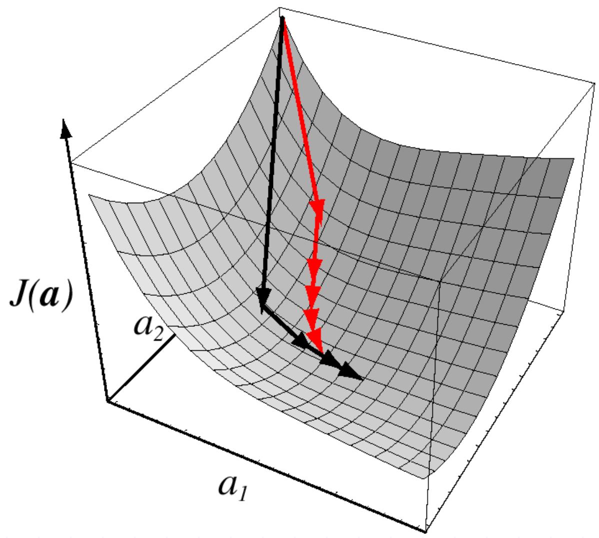
$$\partial \frac{D(\beta, \beta_0)}{\partial \beta} = -\sum_{i \in M} y_i x_i$$

$$\partial \frac{D(\beta, \beta_0)}{\partial \beta_0} = -\sum_{i \in M} y_i$$

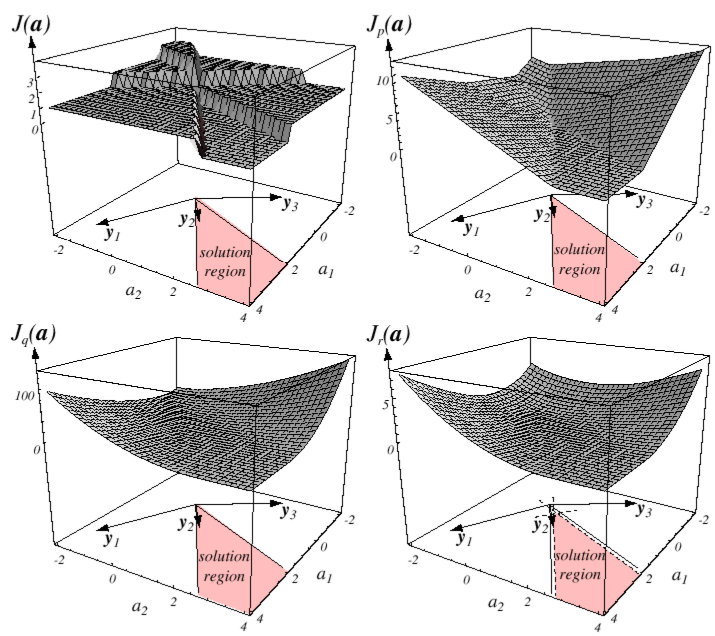
Rosenblatt used 'stocastic gradient descent' to minimize this peicewise linear criterion

[1]

Finding the separator



[4]



[4]

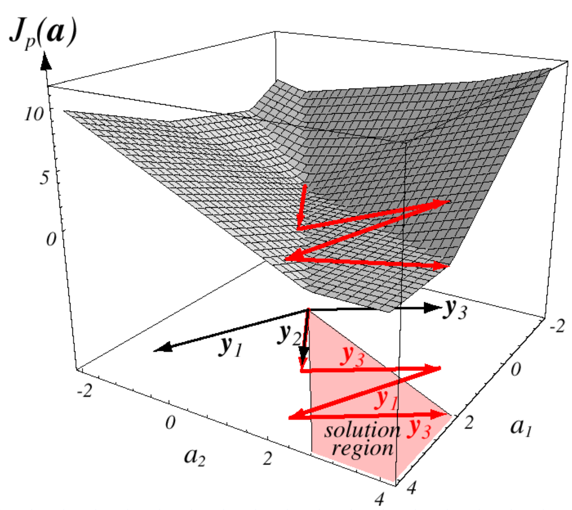
Piecewise update:

$$\begin{pmatrix}\beta \\ \beta_0\end{pmatrix} \leftarrow \begin{pmatrix}\beta \\ \beta_0\end{pmatrix} + \rho \begin{pmatrix}y_i x_i \\ y_i\end{pmatrix}$$

Here ρ is the 'learning rate' this can be shown to eventually converge on a separating hyperplane in a finite number of steps.

[1]

Fitting a perceptron



[4]

25

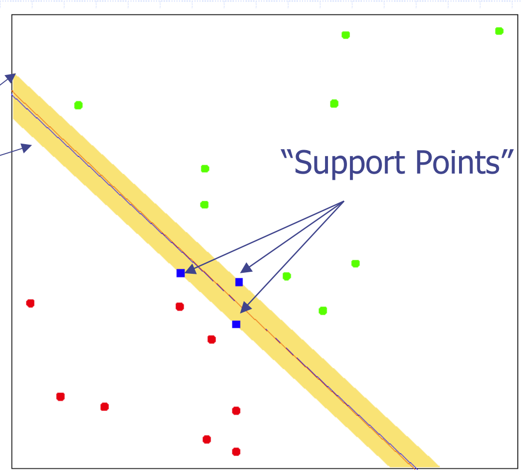
Problems with Rosenblatt

1. When the data are separable, there are many solutions, which one found depends on the starting values.
2. The 'finite number of steps' can be very large, the smaller the gap, the hard it is to find it.
3. When the data are not separable, cycles develop. The cycles can be long and hard to detect.
4. **Linear method only**

[5]

26

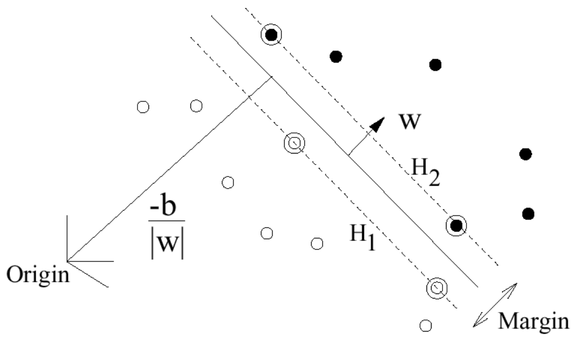
Finding an optimal single solution (1,2,3): Support Vector Machines



[1]

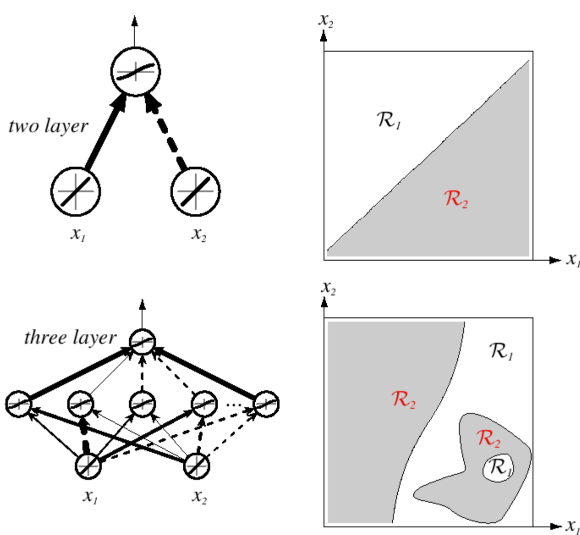
Support Vector Machines

margin: $$d(\mathbf{x}, H(\mathbf{w}, b)) = \frac{\mathbf{w}^\top \mathbf{x} + b}{\|\mathbf{w}\|}$$

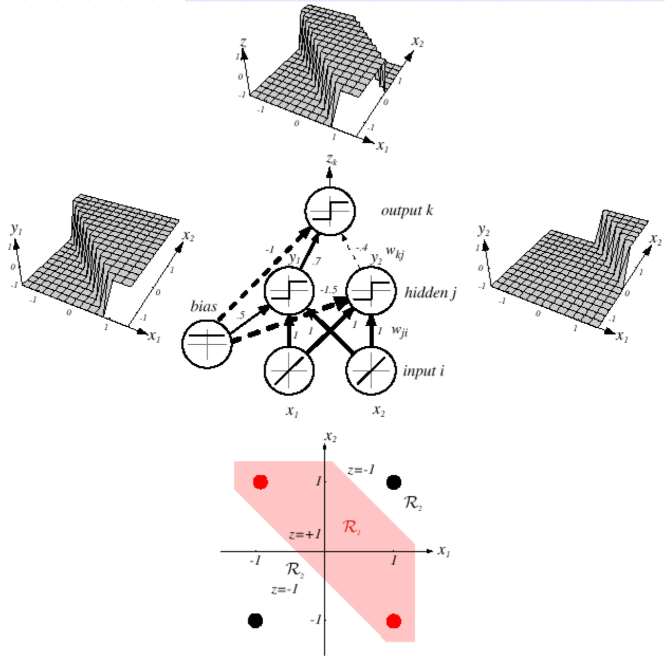


[6]

Non-Linear Systems (4): multiple layers

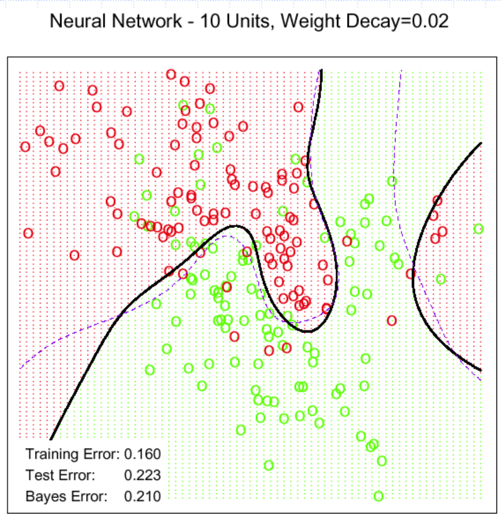


[4]



[4]

Next steps for non-linear boundaries: Multilayer Neural Networks



[1]

References

[1] "The Elements of Statistical Learning: Data Mining, Inference and Prediction", Hastie, Tibshirani and Friedman, Springer-Verlag, (2001).

[2] "Modern Applied Statistics in S", W.N. Venables, B.D. Ripley - 4th Ed, Springer-Verlag, 2002

[3] "Statistical Pattern Recognition: A Review", A.K. Jain, R.P.W. Duin, and J. Mao, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 22, NO. 1, JANUARY 2000

[4] "Pattern Classification", Duda, Hart, Stork, John Wiley Sons, (2001).

[5] Pattern Recognition and Neural Networks, B.D. Ripley, Cambridge University Press, 1996

[6] "A Tutorial on Support Vector Machines for Pattern Recognition", C. Burges, In: Data Mining and Knowledge Discovery 2 p 121-167, Kluwer 1998