IT342: Pattern Recognition

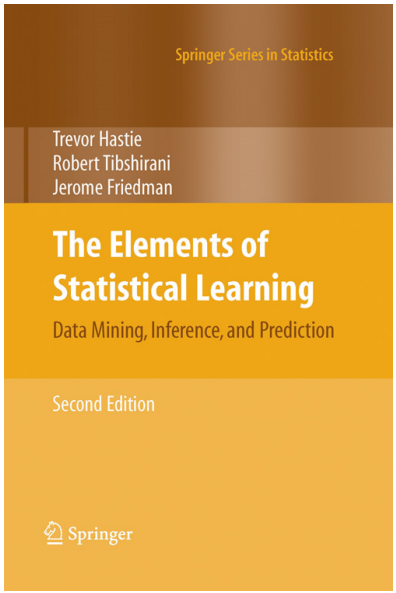
Insight by Mathematics and Intuition for Understanding Pattern Recognition

Waleed A. Yousef, Ph.D.,

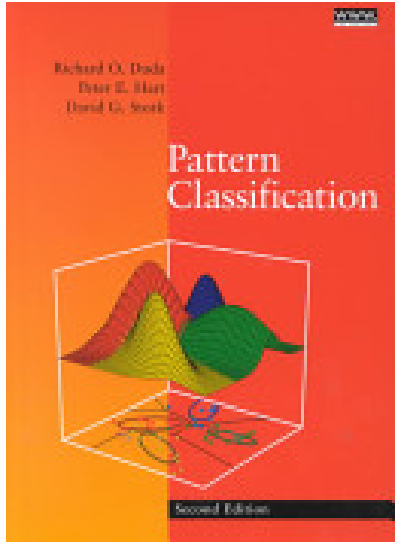
Human Computer Interaction Lab.,
Computer Science Department,
Faculty of Computers and Information,
Helwan University,
Egypt.

December 8, 2018

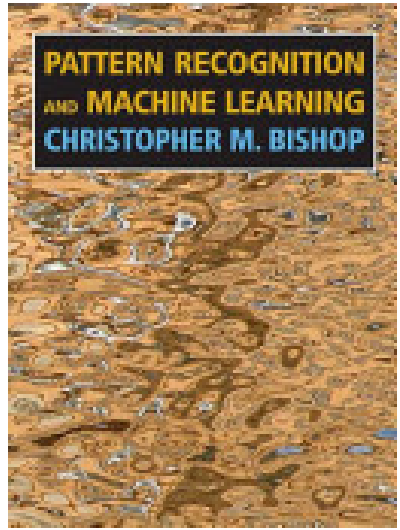
Lectures follow (and some figures are adapted from):



Hastie, Tibshirani, and Friedman, "*The Elements of Statistical Learning*", 2nd edition, Springer.



Duda, Hart, and Stork, "*Pattern Classification*", 2nd edition, Wiley.



Bishop, "*Pattern Recognition and Machine Learning*", Springer.

Course Objectives

- Developing rigorous mathematical treatment.
- Building intuition.
- Developing computer practice to build and assess models on real- and simulated-datasets.

Contents

Chapter 1

Introduction

"**Learning:** is the process of estimating an unknown input-output dependency or structure of a system using a limited number of observations." (Cherkassky and Mulier, 1998).

Statistical learning plays a key role in many areas of science, finance and industry. Here are some examples of learning problems:

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack. The prediction is to be based on demographic, diet and clinical measurements for that patient.
- Predict the price of a stock in 6 months from now, on the basis of company performance measures and economic data.
- Identify the numbers in a handwritten ZIP code, from a digitized image.
- Estimate the amount of glucose in the blood of a diabetic person, from the infrared absorption spectrum of that person's blood.
- Identify the risk factors for prostate cancer, based on clinical and demographic variables.

Supervised Learning we have training set, features (predictors, or input), and outcome (response or output). Build a model to predict future data.

Unsupervised Learning We observe only the features and have no outcome. We need to cluster data or organize it.

Example 1: Email Spam

TABLE 1.1. *Average percentage of words or characters in an email message equal to the indicated word or character. We have chosen the words and characters showing the largest difference between* `spam` *and* `email`.

	george	you	your	hp	free	hpl	!	our	re	edu	remove
spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01	0.28
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29	0.01

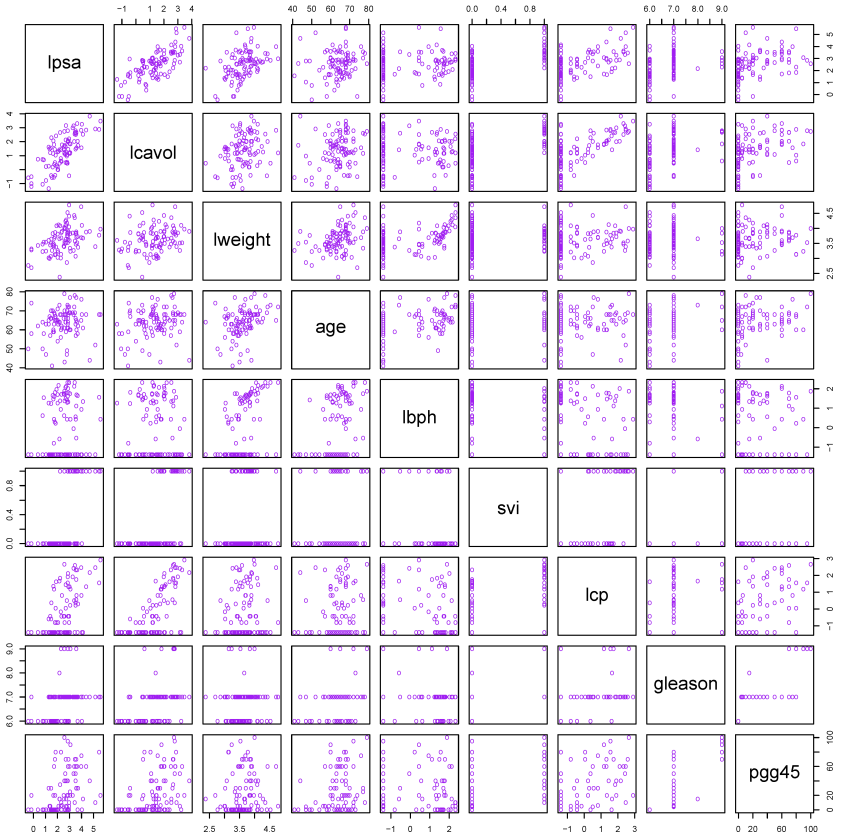
- 4601 email messages to try to predict whether the email was junk or not. The true outcome (`email` or `spam`) is available. This is also called classification problem (as will be explained later).
- The rule could be:

```
if (%george<0.6)&(%you>1.5) then spam
else mail
```

- But is this the "best" rule?
- Not all errors are equal!!

Example 2: Prostate Cancer

- 97 men (observations): predict the log of Prostate Specific Antigen (`lpsa`) from a number of measurements including log-cancer-volume (`lcavol`).
- This is a regression problem (of course supervised).



Example 3: Handwritten Digit Recognition

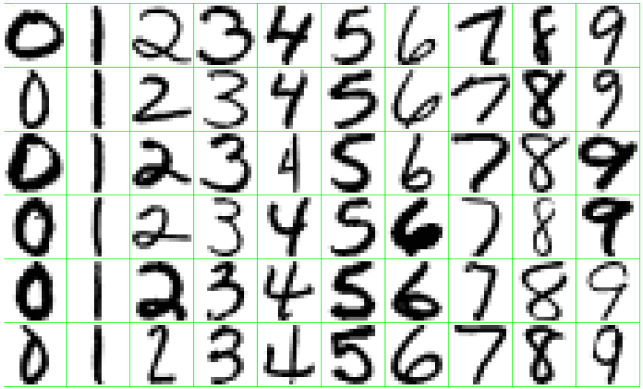


FIGURE 1.2. *Examples of handwritten digits from U.S. postal envelopes.*

- The data comes from the handwritten ZIP codes on envelops from U.S. postal mail.
- The images are 16×16 eight-bit grayscale maps, with each pixel ranging from 0-255.
- The task is to predict (classify) each image from its features (16×16) features to one of the digits.
- I'd like to see one of the projects to study this dataset and apply NN to it.

Example 4: DNA Expression Microarrays

- DNA is the basic material that makes up human chromosomes. On the DNA chip, and through fluoroscopy, the gene-expression is measured.
- It ranges, e.g., between -6 to 6; positive values indicate higher expression (red) and negative indicate lower expression (green).
- Thousands of genes (features) exist; this is always ill-posed problem.
- The figure: experiment of 6830 genes (rows) (only 100 of them are displayed for clarity) and 64 samples (columns). The samples are 64 cancer tumor from different patients.
- Which samples are most similar to each other (across genes)?
- Which genes are most similar to each other (across samples)?
- Do certain genes show very high (or low) expression for certain cancer samples?

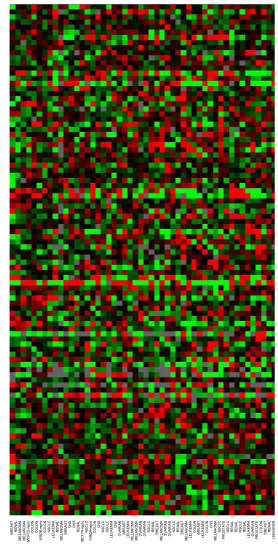


FIGURE 1.3. *DNA microarray data: expression ma-*

This problem can be viewed as:

- unsupervised learning: each sample is an observation in $\mathcal{R}^{6830}$.
- Regression: genes and kind of cancer (sample) are categorical predictors, and gene expression is the response.

Chapter 2

Introduction to Statistical Decision Theory

2.1 Types of Variables and Important Notation

Quantitative, where some measure is given as a value; e.g.,$X = 1, 3, -2.5$.

Qualitative (or Categorical), where no measures or metrics are associated; e.g., $X = Diseased, Nondiseased$.

Ordered Categorical; e.g., $X = small, medium, ...$ The variable $X \in \mathcal{G}$, a set of possible values.

X Random variable (or vector). In general, $X = \left(X_1, \ldots, X_p\right)'$

x_i i^{th}observation from X. Therefore, $x_i = \left(x_{i1}, \ldots, x_{ip}\right)'$

Y To denote a quantitative response

G Qualitative (for group) response; $G \in \mathcal{G}$

$\mathbf{X}$ A data matrix: $\mathbf{X}_{N\times p} = \begin{pmatrix} x_1' \\ \vdots \\ x_N' \end{pmatrix} = \begin{pmatrix} x_{11} & \ldots & x_{1p} \\ \vdots & \ddots & \\ x_{N1} & & x_{Np} \end{pmatrix}$

$\mathbf{x}_j$ All observations of X_j; i.e., j^{th} vector of $\mathbf{X}$: $\mathbf{x}_j = \begin{pmatrix} x_{1j} \\ \vdots \\ x_{Nj} \end{pmatrix}$

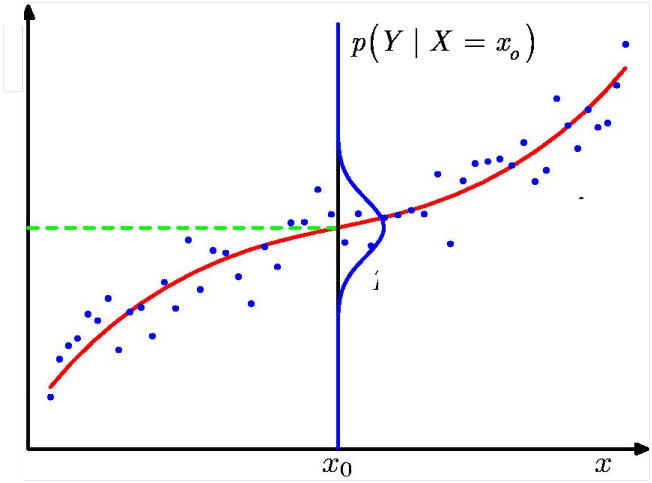
$\widehat{Y}$ The predicted value of Y

$\widehat{G}$ Predicted category (or class); $\widehat{G} \in \mathcal{G}$.

$\mathbf{tr}$ Training set (dataset); $\mathbf{tr} = \left\{t_i | t_i = \left(x_i, y_i\right), i = 1, \ldots, N\right\}$.

2.2 “Best” Decision?

2.2.1 Regression



- Predictor $X \in \mathbb{R}^p$ and the response $Y \in \mathbb{R}$.
- What is the “best” prediction $\widehat{Y} = f(X)$?
- This should be defined in terms of some loss.
- This best will not be the best for another loss! In terms of square error loss.

We have to minimize:

$$
\begin{aligned}
Risk = EPE &= \mathrm{E}\left(Y - f(X)\right)^2 \\
&= \int \left(Y - f(X)\right)^2 f_{XY}(x,y)\,dxdy \\
&= \int \left(Y - f(X)\right)^2 f_{Y|X} f_X\,dxdy \\
&= \int \left[\int \left(Y - f(X)\right)^2 f_{Y|X}\,dy\right] f_X\,dx \\
&= \mathop{\mathrm{E}}_{X}\mathop{\mathrm{E}}_{Y|X}\left(Y - f(X)\right)^2
\end{aligned}
$$

$$
\underbrace{\mathop{\mathrm{E}}_{Y|X}\left(Y - f(x)\right)^2}_{\text{Conditional Risk}} = \qquad \text{(minimize pointwise w.r.t. } X\text{)}
$$

$$
\begin{aligned}
&= \mathop{\mathrm{E}}_{Y|X}\left[(Y - \mathrm{E}[Y|X=x]) + \mathrm{E}\left([Y|X=x] - f(x)\right)\right]^2 \\
&= \mathop{\mathrm{E}}_{Y|X}\{(Y - \mathrm{E}[Y|X=x])^2 + 2\left(Y - \mathrm{E}[Y|X=x]\right)\left(\mathrm{E}[Y|X=x] - f(x)\right) + \left(\mathrm{E}[Y|X=x] - f(x)\right)^2\} \\
&= \mathop{\mathrm{E}}_{Y|X}(Y - \mathrm{E}[Y|X=x])^2 + \left(\mathrm{E}[Y|X=x] - f(x)\right)^2 \\
&= \sigma^2_{Y|X} + \left(\mathrm{E}[Y|X=x] - f(x)\right)^2
\end{aligned}
$$

$$
f^*(x) = \mathop{\arg\min}_{f(X)} \mathop{\mathrm{E}}_{Y|X}\left(Y - f(x)\right)^2 = \mathrm{E}[Y|X=x] \tag{2.1}
$$

$$
Risk_{\min} = \mathop{\mathrm{E}}_{X}\sigma^2_{Y|X} \tag{2.2}
$$

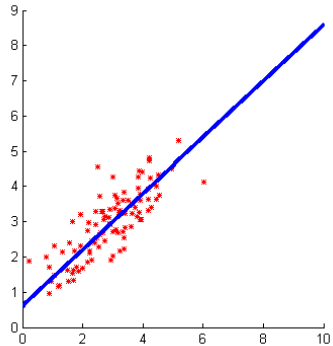
Very Important:

- This is an iff rule: ANY OTHER RULE WILL BE INFERIOR TO f^*.
- So, it is impossible that any other rule (or algorithm) will be the best for all kind of problems.
- We have to try!

Example 1 (Multinormal Distribution) .

Matlab Code 2.1:

```
mu=3+zeros(1, 2); samples=100;
s1=1; s2=1; r=.8; sigma=[s1^2 s1*s2*r; s1*s2*r s2^2];
X=mvnrnd(mu, sigma, samples);
scatter(X(:,1),X(:,2),20,'*r'); hold on

x=0:.1:10; y=mu(1)+sigma(1, 2)/(sigma(2,2)) * (x-mu(2));
plot(x, y, '-', 'LineWidth', 3);

set(gcf, 'Units', 'inches'); set(gcf, 'position', [1, 1, 4, 4]);
```



This complies with the following theorem:

Theorem 2 *Let the components of Z be deviled into two groups composing the sub-vectors Y, X such that*

$$Z \sim \mathcal{N}(\mu_Z, \Sigma_Z),$$
$$\begin{pmatrix} Y \\ X \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_Y \\ \mu_X \end{pmatrix}, \begin{pmatrix} \Sigma_{YY} & \Sigma_{YX} \\ \Sigma_{XY} & \Sigma_{XX} \end{pmatrix}\right),$$

then

$$Y \sim \mathcal{N}(\mu_Y, \Sigma_{YY}),$$
$$Y|X \sim \mathcal{N}\left(\mu_Y + \Sigma_{YX}\Sigma_{XX}^{-1}(x - \mu_X), \Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}\right)$$

Notice:

- $\mathrm{E}[Y|X]$ is a line in p-dimensions.
- for scalar Y and X, $\mathrm{E}[Y|X] = \mu_Y + \frac{\sigma_{YX}}{\sigma_X^2}(x - \mu_X)$.
- the slope $\frac{\sigma_{YX}}{\sigma_X^2} = \frac{\rho\sigma_Y}{\sigma_X}$ of the regression line makes sense.
- proofs and details are in the appendix.

2.2.2 Classification

We have K classes, X may belong to any. The Loss is $L = L\left(G, \widehat{G}(X)\right)$, which is $K \times K$ matrix; sometimes we call it L_{kl}, which is the price paid for classifying an observation belonging to class $\mathcal{G}_k$ as $\mathcal{G}_l$. We have to minimize the risk under this loss:

$$\begin{aligned} R &= \mathrm{E}\, L\left(G, \widehat{G}(X)\right) \\ &= \mathop{\mathrm{E}}_{X} \mathop{\mathrm{E}}_{G|X} L\left(G, \widehat{G}(X)\right) \\ &= \mathop{\mathrm{E}}_{X} \underbrace{\sum_{k=1}^{K} L\left(\mathcal{G}_k, g\right) \Pr\left(\mathcal{G}_k | X\right)}_{\lambda(g)\ \text{Conditional Risk}}. \end{aligned} \tag{2.3}$$

Minimize R pointwise (i.e., minimize the conditional risk). Then, at particular $X = x$, calculate the conditional risk for each decision g and choose the minimum; i.e.,

$$\widehat{G}(x) = \arg\min_{g \in \mathcal{G}} \sum_{k=1}^{K} L\left(\mathcal{G}_k, g\right) \Pr\left(\mathcal{G}_k | X = x\right)$$

2.2.2.1 Two classes

$$\begin{aligned} \lambda\left(\mathcal{G}_1\right) &= L_{11} \Pr\left(\mathcal{G}_1 | X = x\right) + L_{21} \Pr\left(\mathcal{G}_2 | X = x\right) \\ \lambda\left(\mathcal{G}_2\right) &= L_{12} \Pr\left(\mathcal{G}_1 | X = x\right) + L_{22} \Pr\left(\mathcal{G}_2 | X = x\right) \end{aligned}$$

What do you expect for the best rule?

$$\lambda(\mathcal{G}_2) \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \lambda(\mathcal{G}_1)$$

$$L_{12}\Pr(\mathcal{G}_1|X=x) + L_{22}\Pr(\mathcal{G}_2|X=x) \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}}$$
$$L_{11}\Pr(\mathcal{G}_1|X=x) + L_{21}\Pr(\mathcal{G}_2|X=x)$$
$$\frac{\Pr(\mathcal{G}_1|X=x)}{\Pr(\mathcal{G}_2|X=x)} \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \frac{(L_{21}-L_{22})}{(L_{12}-L_{11})}$$

$$\frac{\Pr(\mathcal{G}_1|X=x)}{\Pr(\mathcal{G}_2|X=x)} \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \frac{(L_{21}-L_{22})}{(L_{12}-L_{11})}$$
$$\frac{\Pr(\mathcal{G}_1|X=x)}{\Pr(\mathcal{G}_2|X=x)} \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \frac{L_{21}}{L_{12}}, \qquad (L_{ii}=0 \text{ usually})$$

which makes a lot of sense, as we classify according to the maximum posterior(modified by the loss weights).

$$\frac{\Pr(X|\mathcal{G}_1)\pi_1/P(X=x)}{\Pr(X|\mathcal{G}_2)\pi_2/P(X=x)} \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \frac{L_{21}}{L_{12}}$$
$$\frac{f_1(X)}{f_2(X)} \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \frac{\pi_2 L_{21}}{\pi_1 L_{12}}, \qquad \text{(LR)}$$

which makes another great **sense**. We classify as $\mathcal{G}_1$ if its prior is larger, unless $\mathcal{G}_2$ has higher prevalence (higher probability π_2 or higher loss for misclassification L_{21}) we raise the bar.

Usually, this rule is put in the form

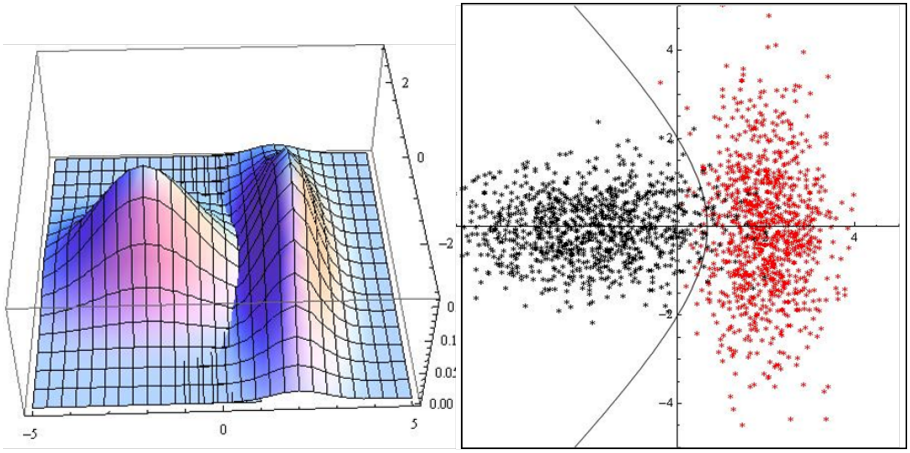
$$\ln\left(\frac{f_1(X)}{f_2(X)}\right) \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \ln\left(\frac{\pi_2 L_{21}}{\pi_1 L_{12}}\right) \tag{LLR}$$

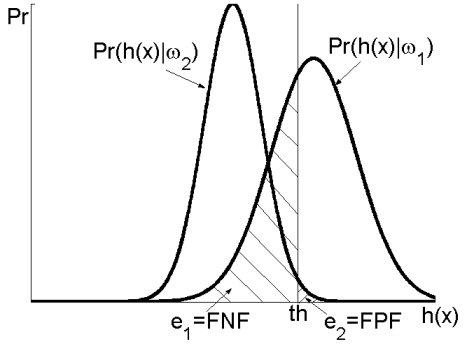
$$h(X) \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} th,$$

$$\eta^*(X) = \begin{cases} \mathcal{G}_1 & h(X) > th \\ \mathcal{G}_2 & h(X) < th \end{cases}$$

The decision surface and the two regions of decision:

$$\begin{aligned} \mathcal{S}^* &= \{x : h(X) = th\}, \\ \mathcal{R}_1^* &= \{x : h(X) > th\}, \\ \mathcal{R}_2^* &= \{x : h(X) < th\} \end{aligned} \tag{2.4}$$


2
0
−5
0
5
0.1
0.0
0.00
4
2
−2
−4
4


Pr
Pr(h(x)|ω2)
Pr(h(x)|ω1)
e1=FNF
th
e2=FPF
h(x)

The risk in general (for any rule η) is given by (2.3) as

$$
\begin{aligned}
R &= \underbrace{\mathop{\mathrm{E}}_{X} \sum_{k=1}^{K} L\left(\mathcal{G}_k, g\right) \Pr\left(\mathcal{G}_k | X\right)}_{\lambda(g) \text{ Conditional Risk}} \\
R^* &= \int_{\mathcal{R}_1} \lambda\left(\mathcal{G}_1\right) P\left(X\right) dX + \int_{\mathcal{R}_2} \lambda\left(\mathcal{G}_2\right) P\left(X\right) dX \\
&= \int_{\mathcal{R}_1} L_{21} \Pr\left(\mathcal{G}_2 | X = x\right)\ P\left(X\right) dX + \int_{\mathcal{R}_2} L_{12} \Pr\left(\mathcal{G}_1 | X = x\right)\ P\left(X\right) dX \\
&= L_{21}\pi_2 \underbrace{\int_{\mathcal{R}_1^*} f\left(X | \mathcal{G}_2\right) dX}_{\text{Error } e_{21}^* = \Pr\left[\mathcal{R}_1^* | \mathcal{G}_2\right]} + L_{12}\pi_1 \underbrace{\int_{\mathcal{R}_2^*} f\left(X | \mathcal{G}_1\right) dX}_{\text{Error } e_{12}^* = \Pr\left[\mathcal{R}_2^* | \mathcal{G}_1\right]}, \\
&= L_{21}\pi_2 \Pr\left[x \in \mathcal{G}_2 \text{ and decision is } \mathcal{G}_1\right] + L_{12}\pi_1 \Pr\left[x \in \mathcal{G}_1 \cap \text{ and decision is } \mathcal{G}_2\right],
\end{aligned}
$$

a lot of sense: each kind of error is an integration of the right class over the wrong decision region, then magnified by the prevalence of that class. Then, From (2.4)

$$
\begin{aligned}
x \in \mathcal{R}_1^* \equiv th < h < \infty &\rightarrow \Pr\left[\mathcal{R}_1^*\right] = \Pr\left[th < h < \infty\right] \\
x \in \mathcal{R}_2^* \equiv -\infty < h < th &\rightarrow \Pr\left[\mathcal{R}_2^*\right] = \Pr\left[-\infty < h < th\right] \\
R^* &= L_{21}\pi_2 \underbrace{\int_{th}^{\infty} f_h\left(h | \mathcal{G}_2\right)\ dh}_{\text{Error } e_{21}^*} + L_{12}\pi_1 \underbrace{\int_{-\infty}^{th} f\left(h | \mathcal{G}_1\right) dh}_{\text{Error } e_{12}^*}.
\end{aligned}
$$

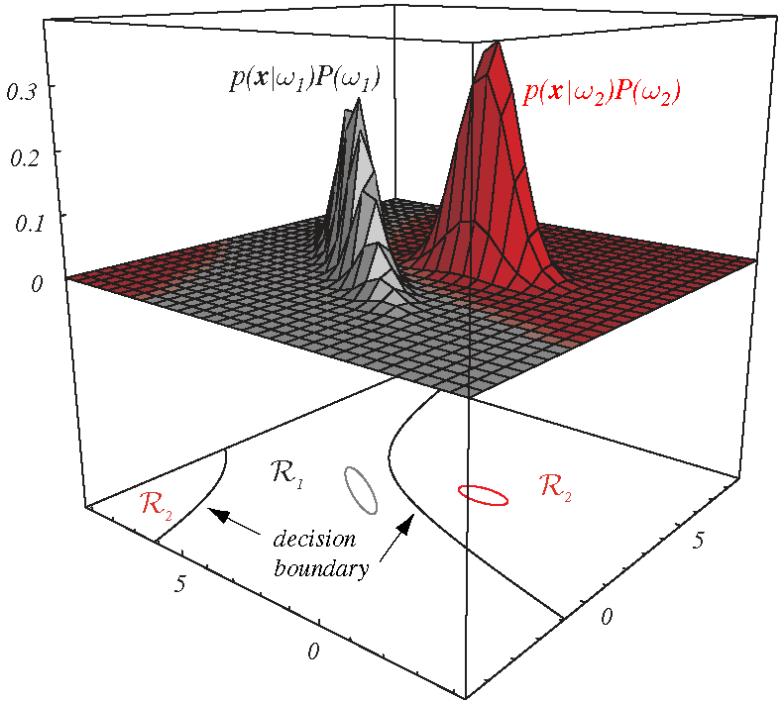
Example 3 (Multinormal Distribution) .

$$f_1(x) = \frac{1}{((2\pi)^p |\Sigma_1|)^{1/2}} e^{-\frac{1}{2}(x-\mu_1)'\Sigma_1^{-1}(x-\mu_1)},$$
$$f_2(x) = \frac{1}{((2\pi)^p |\Sigma_2|)^{1/2}} e^{-\frac{1}{2}(x-\mu_2)'\Sigma_2^{-1}(x-\mu_2)},$$

Then, $\mathcal{S}^*$ is given by:

$$\frac{f_1(x)}{f_2(x)} \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \left(\frac{\pi_2 L_{21}}{\pi_1 L_{12}}\right)$$
$$e^{-\frac{1}{2}(x-\mu_1)'\Sigma_1^{-1}(x-\mu_1)+\frac{1}{2}(x-\mu_2)'\Sigma_2^{-1}(x-\mu_2)} \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \left(\frac{\pi_2 L_{21}}{\pi_1 L_{12}} \frac{|\Sigma_1|^{1/2}}{|\Sigma_2|^{1/2}}\right)$$
$$-(x-\mu_1)'\Sigma_1^{-1}(x-\mu_1) + (x-\mu_2)'\Sigma_2^{-1}(x-\mu_2)$$
$$\underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} 2\ln\left(\frac{\pi_2 L_{21}}{\pi_1 L_{12}} \frac{|\Sigma_1|^{1/2}}{|\Sigma_2|^{1/2}}\right)$$
$$\underbrace{x'\left(\Sigma_2^{-1}-\Sigma_1^{-1}\right)x}_{\textit{Quadratic Term}} - \underbrace{2x'\left(\Sigma_2^{-1}\mu_2 - \Sigma_1^{-1}\mu_1\right)}_{\textit{Linear Term}} + \left(\mu_2'\Sigma_2^{-1}\mu_2 - \mu_1'\Sigma_1^{-1}\mu_1\right)$$
$$\underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} 2th + \ln\left(\frac{|\Sigma_1|}{|\Sigma_2|}\right)$$

The general geometry of S^* will be:



Three special cases of interest are:

Case 1: $\Sigma_1 = \Sigma_2 = \Sigma = \sigma^2\mathbf{I}$; then $\mathcal{S}^*$ is given by:

$$\underbrace{x'\left(\Sigma_2^{-1}-\Sigma_1^{-1}\right)x}_{\text{Quadratic Term}} - \underbrace{2x'\left(\Sigma_2^{-1}\mu_2-\Sigma_1^{-1}\mu_1\right)}_{\text{Linear Term}} + \left(\mu_2'\Sigma_2^{-1}\mu_2-\mu_1'\Sigma_1^{-1}\mu_1\right) = 2th + \ln\left(\frac{|\Sigma_1|}{|\Sigma_2|}\right)$$

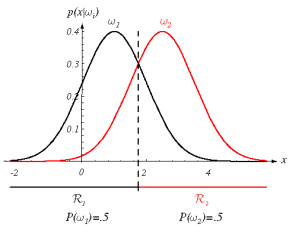
$$x'\Sigma^{-1}\left(\mu_2-\mu_1\right)-\frac{1}{2}\left(\mu_2'\Sigma^{-1}\mu_2-\mu_1'\Sigma^{-1}\mu_1\right)=-th$$

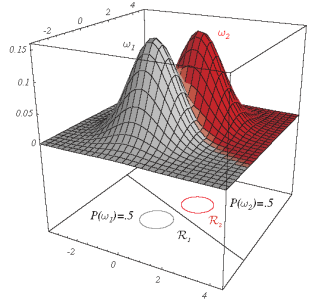
$$\frac{1}{\sigma^2}x'\left(\mu_2-\mu_1\right)-\frac{1}{2\sigma^2}\left(\mu_2'\mu_2-\mu_1'\mu_1\right)=-th$$

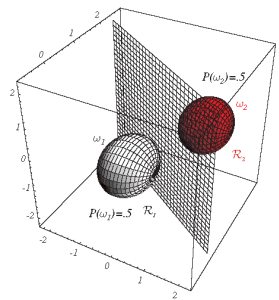
$$\left(\mu_2-\mu_1\right)'x-\frac{1}{2}\left(\mu_2-\mu_1\right)'\left(\mu_2+\mu_1\right)=-\sigma^2 th\frac{\left(\mu_2-\mu_1\right)'\left(\mu_2-\mu_1\right)}{\|\mu_2-\mu_1\|^2}$$

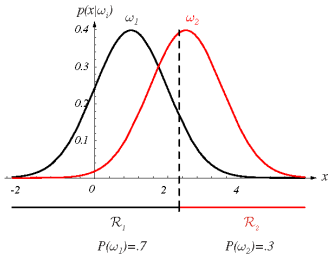
$$\left(\mu_2-\mu_1\right)'\left(x-\left[\frac{1}{2}\left(\mu_2+\mu_1\right)-\frac{\sigma^2 th}{\|\mu_2-\mu_1\|^2}\left(\mu_2-\mu_1\right)\right]\right)=0$$

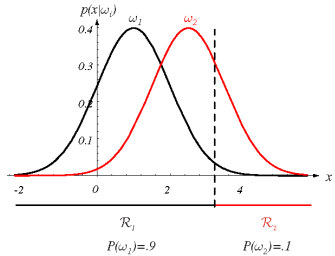
$$w'\left(x-x_0\right)=0$$

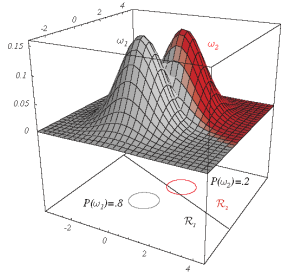


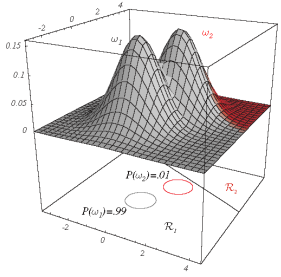


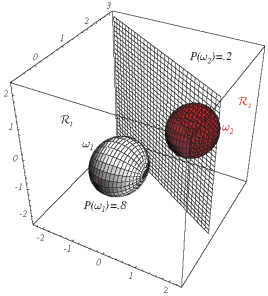


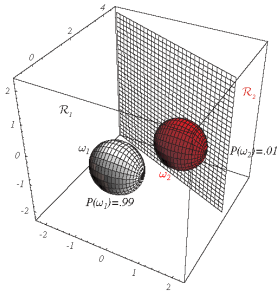

$p(x|\omega_i)$
ω_1
ω_2
0.4
0.3
0.2
0.1
-2
0
2
4
x
$\mathcal{R}_1$
$\mathcal{R}_2$
$P(\omega_1)=.7$
$P(\omega_2)=.3$


$p(x|\omega_i)$
ω_1
ω_2
0.4
0.3
0.2
0.1
-2
0
2
4
x
$\mathcal{R}_1$
$\mathcal{R}_2$
$P(\omega_1)=.9$
$P(\omega_2)=.1$


0.15
0.1
0.05
0
ω_1
ω_2
$P(\omega_2)=.2$
$P(\omega_1)=.8$
$\mathcal{R}_2$
$\mathcal{R}_1$


0.15
0.1
0.05
0
ω_1
ω_2
$P(\omega_2)=.01$
$P(\omega_1)=.99$
$\mathcal{R}_2$
$\mathcal{R}_1$


$P(\omega_2)=.2$
$\mathcal{R}_2$
$\mathcal{R}_1$
ω_2
ω_1
$P(\omega_1)=.8$


$\mathcal{R}_2$
$\mathcal{R}_1$
$P(\omega_2)=.01$
ω_1
ω_2
$P(\omega_1)=.99$

Case 2: $\Sigma_1 = \Sigma_2 = \Sigma$; then $\mathcal{S}^*$ is given by:

$$\underbrace{x'\left(\Sigma_2^{-1} - \Sigma_1^{-1}\right)x}_{\text{Quadratic Term}} - \underbrace{2x'\left(\Sigma_2^{-1}\mu_2 - \Sigma_1^{-1}\mu_1\right)}_{\text{Linear Term}} + \left(\mu_2'\Sigma_2^{-1}\mu_2 - \mu_1'\Sigma_1^{-1}\mu_1\right) = 2th + \ln\left(\frac{|\Sigma_1|}{|\Sigma_2|}\right)$$

$$x'\Sigma^{-1}\left(\mu_2 - \mu_1\right) - \frac{1}{2}\left(\mu_2'\Sigma^{-1}\mu_2 - \mu_1'\Sigma^{-1}\mu_1\right) = -th$$

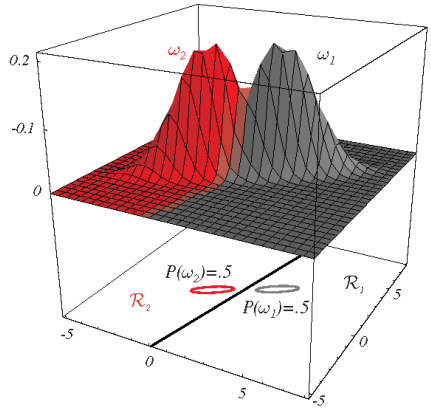
$$\left[\Sigma^{-1}\left(\mu_2 - \mu_1\right)\right]' x - \frac{1}{2}\left(\mu_2 - \mu_1\right)'\Sigma^{-1}\left(\mu_2 + \mu_1\right) =$$

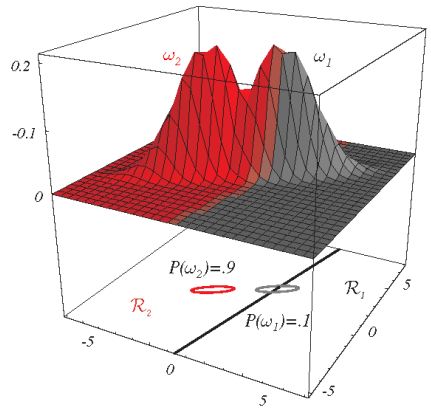
$$-th\frac{\left(\mu_2 - \mu_1\right)'\Sigma^{-1}\left(\mu_2 + \mu_1\right)}{\left(\mu_2 - \mu_1\right)'\Sigma^{-1}\left(\mu_2 + \mu_1\right)}$$

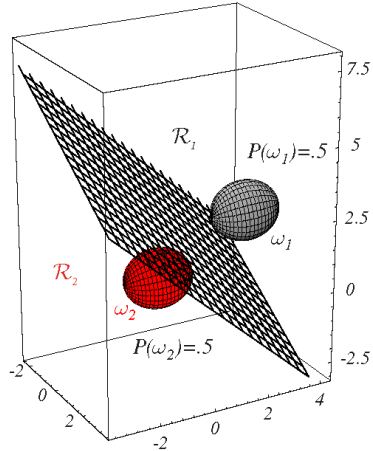
$$\left[\Sigma^{-1}\left(\mu_2 - \mu_1\right)\right]' \cdot$$

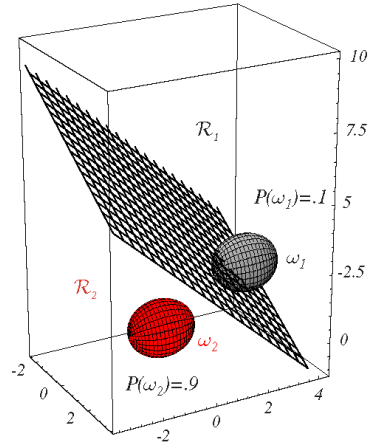
$$\left(x - \left[\frac{1}{2}\left(\mu_2 + \mu_1\right) - \frac{th}{\left(\mu_2 - \mu_1\right)'\Sigma^{-1}\left(\mu_2 + \mu_1\right)}\left(\mu_2 + \mu_1\right)\right]\right) = 0$$

$$w'\left(x - x_0\right) = 0$$


0.2
-0.1
0
ω_2
ω_1
$P(\omega_2)$=.5
$P(\omega_1)$=.5
$\mathcal{R}_1$
$\mathcal{R}_2$
-5
0
5
-5
0
5

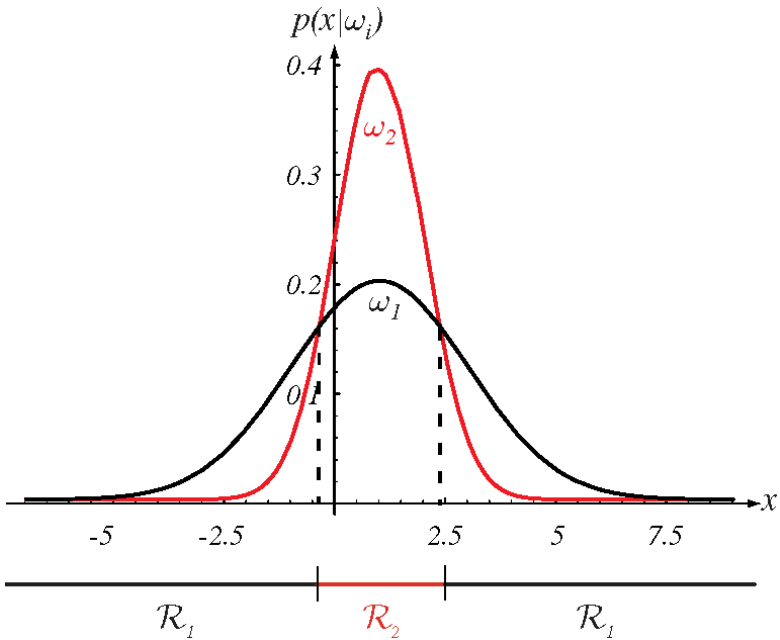

0.2
-0.1
0
ω_2
ω_1
$P(\omega_2)$=.9
$P(\omega_1)$=.1
$\mathcal{R}_1$
$\mathcal{R}_2$
-5
0
5
-5
0
5

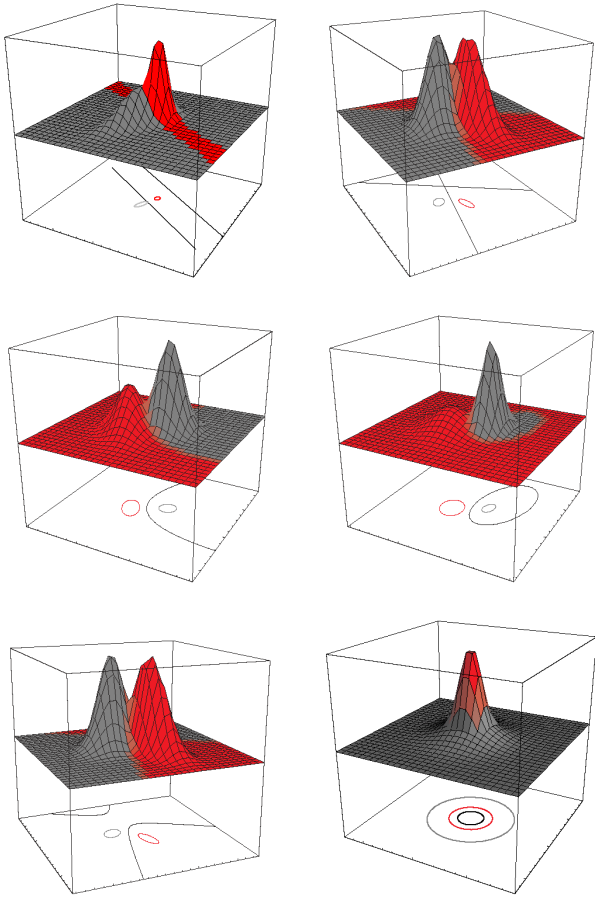

7.5
5
2.5
0
-2.5
$\mathcal{R}_1$
$P(\omega_1)$=.5
ω_1
$\mathcal{R}_2$
ω_2
$P(\omega_2)$=.5
-2
0
2
-2
0
2
4

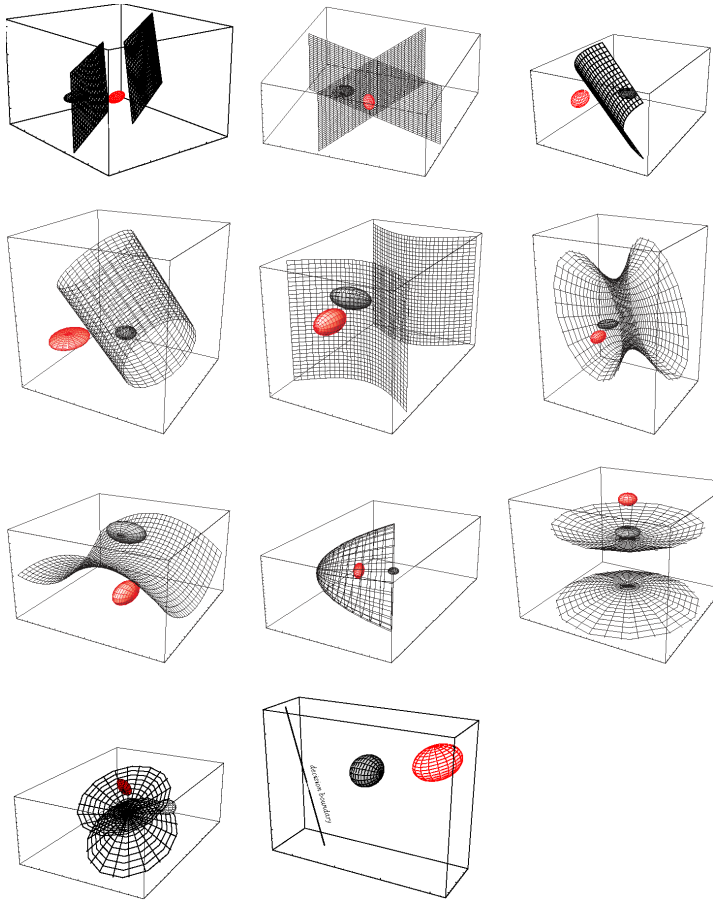

10
7.5
5
-2.5
0
$\mathcal{R}_1$
$P(\omega_1)$=.1
ω_1
$\mathcal{R}_2$
ω_2
$P(\omega_2)$=.9
-2
0
2
-2
0
2
4

Case 3: Σ_1, Σ_2 arbitrary; then $\mathcal{S}^*$ is given by:

$$\underbrace{x'\left(\Sigma_2^{-1}-\Sigma_1^{-1}\right)x}_{\text{Quadratic Term}} \quad - \quad \underbrace{2x'\left(\Sigma_2^{-1}\mu_2-\Sigma_1^{-1}\mu_1\right)}_{\text{Linear Term}} \quad + \quad \left(\mu_2'\Sigma_2^{-1}\mu_2-\mu_1'\Sigma_1^{-1}\mu_1\right) \quad = \quad 2th \quad + \quad \ln\left(\frac{|\Sigma_1|}{|\Sigma_2|}\right)$$



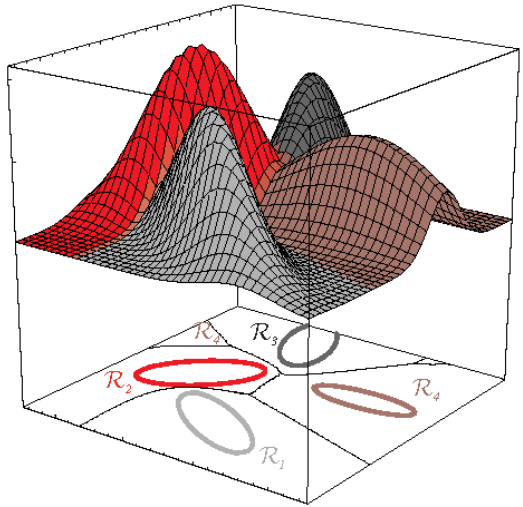



decision boundary

2.2.2.2 Multiclass Problem

$$\widehat{G}(x) = \arg\min_{g \in \mathcal{G}} \sum_{k=1}^{K} L\left(\mathcal{G}_k, g\right) \Pr\left(\mathcal{G}_k | X = x\right)$$

$$\lambda\left(\mathcal{G}_1\right) = \cdots$$
$$\lambda\left(\mathcal{G}_2\right) = \cdots$$
$$\vdots$$
$$\lambda\left(\mathcal{G}_K\right) = \cdots$$



2.3 Getting to “Learning”

- We assume we know the distributions—and hence the “best” rule—but the parameters.
- “Learning” here is nothing but point estimation.
- We consider multinormal case for simplicity.

2.3.1 Regression (Refer to Theorem 2:)

$$
\begin{aligned}
f^*(x) &= \mu_Y + \Sigma_{YX}\Sigma_{XX}^{-1}\left(x-\mu_X\right) && (\mathrm{E}[Y|X]) \\
&= \mu_Y + \left(x-\mu_X\right)'\Sigma_{XX}^{-1}\Sigma_{XY} && (p\text{-dim}) \\
&= \mu_Y + \left(x-\mu_X\right)\frac{\sigma_{XY}}{\sigma_{XX}} && (\text{scalar}) \\
Risk^* &= \mathop{\mathrm{E}}_X\left[\Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}\right] && (\mathrm{E}_X\,\sigma^2_{Y|X}) \\
&= \Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY} && (p\text{-dim}) \\
&= \left(1-\rho^2\right)\sigma_Y^2 && (\text{scalar})
\end{aligned}
$$

Estimate the parameters, then plug in to get $\widehat{f}_{\mathbf{tr}}$ close to f^*

Example 4 (live simulation using Matlab) . *Notice that*

- $\widehat{f}_{\mathbf{tr}}$ *is close to* f^*.
- *The larger the sample size the better the prediction.*
- *This is no longer the best rule nor the risk is min.*
- *New source of variability: training set* **tr**.
- *For this example, or for other difficult distributions, risks can be estimated by simulating a testing set* **ts**

$$
\begin{aligned}
Risk\left(\widehat{f}\right) &= \mathrm{E}\left(Y-\widehat{f}(X)\right)^2 && (\text{page } 13) \\
\widehat{Risk}\left(\widehat{f}_{\mathbf{tr}}\right) &= \frac{1}{n_{\mathbf{ts}}}\sum_{i=1}^{n_{\mathbf{ts}}}\left(y_i-\widehat{f}_{\mathbf{tr}}(x_i)\right)^2
\end{aligned}
$$

Parameter Estimation:

$$\widehat{f}(x) = \widehat{\mu}_Y + \left(x - \widehat{\mu}_X\right)' \widehat{\Sigma}_{XX}^{-1} \widehat{\Sigma}_{XY}$$

$$\widehat{\mu}_Y = \overline{y} = \frac{1}{N}\mathbf{1}'\mathbf{y} = \frac{1}{N}\sum_i y_i,$$

$$\widehat{\mu}_X = \overline{x} = \frac{1}{N}\mathbf{X}'\mathbf{1} = \frac{1}{N}\sum_i x_i,$$

$$\left(\widehat{\Sigma}_{XY}\right)_{p\times 1} = \frac{1}{N-1}\sum_i \left(x_i - \overline{x}\right)_{p\times 1} y_{i_{1\times 1}} = \frac{1}{N-1}\left(\mathbf{X}_c'\right)_{p\times N}\mathbf{y}_{N\times 1}$$

$$\left(\widehat{\Sigma}_{XX}\right)_{p\times p} = \frac{1}{N-1}\sum_i \left(x_i - \overline{x}\right)\left(x_i - \overline{x}\right)' = \frac{1}{N-1}\mathbf{X}_c'\mathbf{X}_c$$

$$\widehat{f}(x) = \overline{y} + x_c'\left(\mathbf{X}_c'\mathbf{X}_c\right)^{-1}\mathbf{X}_c'\mathbf{y} \tag{2.5}$$

$$\widehat{f}(x) = \widehat{\mu}_Y + \left(x - \widehat{\mu}_X\right)\frac{\widehat{\sigma}_{xy}}{\widehat{\sigma}_{xx}} = \overline{y} + x_c\frac{\sum_i \left(x_i - \overline{x}\right) y_i}{\sum_i \left(x_i - \overline{x}\right)^2}. \qquad \text{(for scalar } X\text{)}$$

Hint: *Eq.* (2.5) *will be reached very differently and interestingly next Chapter.*

Mathematical Expression	***Matlab***
$\mathbf{1}_{N\times 1}$	`ones([N, 1])`
$\overline{x}' = \frac{1}{N}\mathbf{1}'\mathbf{X}$	`xbarp = mean(X)`
$(\mathbf{X}_c)_{N\times p} = \mathbf{X} - \mathbf{1}\overline{x}' = \begin{pmatrix} (x_1 - \overline{x})' \\ \vdots \\ (x_p - \overline{x})' \end{pmatrix}$	`X - repmat(xbarp, [N, 1])`

2.3.2 Classification

Quadratic Discriminant Analysis (QDA):

$$\underbrace{x'\left(\Sigma_2^{-1}-\Sigma_1^{-1}\right)x}_{\text{Quadratic Term}} - \underbrace{2x'\left(\Sigma_2^{-1}\mu_2-\Sigma_1^{-1}\mu_1\right)}_{\text{Linear Term}} + \left(\mu_2'\Sigma_2^{-1}\mu_2-\mu_1'\Sigma_1^{-1}\mu_1\right) = 2th + \ln\left(\frac{|\Sigma_1|}{|\Sigma_2|}\right)$$

Get $\widehat{\mu}_1$, $\widehat{\mu}_2$, $\widehat{\Sigma}_1$, and $\widehat{\Sigma}_2$ as before and plug in above.

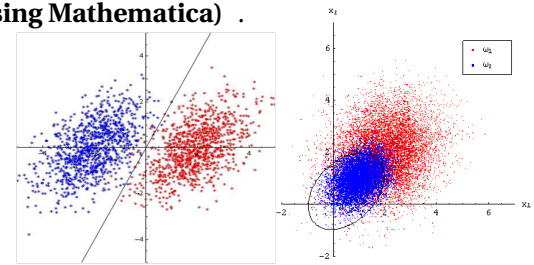
Linear Discriminant Analysis (LDA):

$$-x'\Sigma^{-1}\left(\mu_2-\mu_1\right)+\frac{1}{2}\left(\mu_2'\Sigma^{-1}\mu_2-\mu_1'\Sigma^{-1}\mu_1\right)=th$$

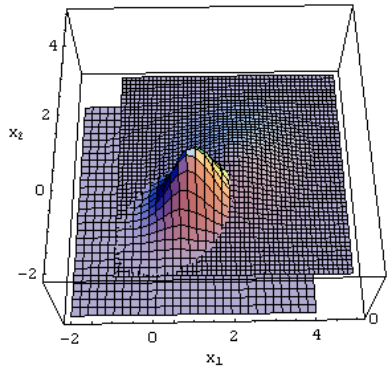
$$\widehat{\Sigma}=\frac{1}{n_{\mathbf{tr}_1}+n_{\mathbf{tr}_2}-2}\left[\left(n_{\mathbf{tr}_1}-1\right)\widehat{\Sigma}_1+\left(n_{\mathbf{tr}_2}-1\right)\widehat{\Sigma}_2\right].$$

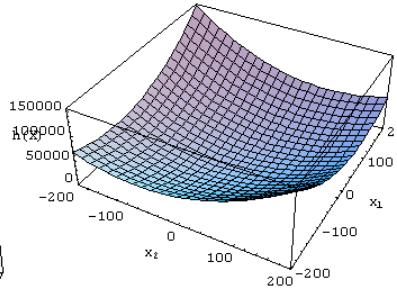
(observe that, if $\widehat{\Sigma}_1$ and $\widehat{\Sigma}_2$ are unbiased $\widehat{\Sigma}$ is so too.)

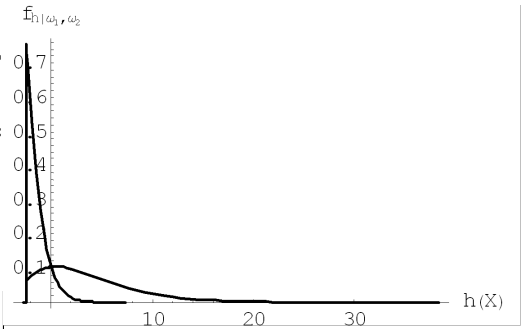
Example 5 (Live simulation using Mathematica) .

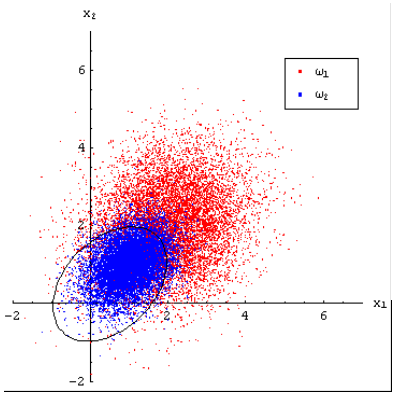


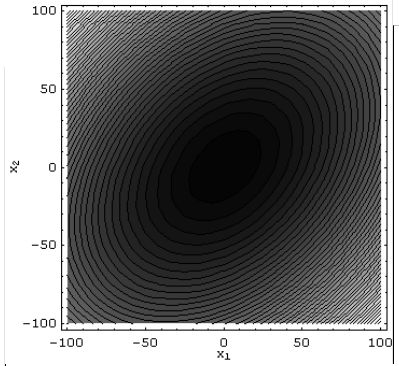
Example 6 (simulating LLR for more understanding) .











Risk Estimation

$$R = L_{21}\pi_2 \underbrace{\int_{\mathcal{R}_1} f\left(X|\mathcal{G}_2\right) dX}_{\text{Error } e_{21}=\Pr[\mathcal{R}_1|\mathcal{G}_2]} + L_{12}\pi_1 \underbrace{\int_{\mathcal{R}_2} f\left(X|\mathcal{G}_1\right) dX}_{\text{Error } e_{12}=\Pr[\mathcal{R}_2|\mathcal{G}_1]},$$

$$L_{21}\pi_2 \underbrace{\int_{th}^{\infty} f_h\left(h|\mathcal{G}_2\right)\, dh}_{\text{Error } e_{21}} + L_{12}\pi_1 \underbrace{\int_{-\infty}^{th} f\left(h|\mathcal{G}_1\right) dX}_{\text{Error } e_{12}},$$

So, simply, after training on **tr** we can test on a very large testing set (MC trial) to get

$$\widehat{e}_{\mathbf{tr}12} = \frac{1}{n_{\mathbf{ts}_1}} \sum_{i=1}^{n_{\mathbf{ts}_1}} I_{\left(\widehat{h}_{\mathbf{tr}}(x_i) < th\right)},$$

$$\widehat{e}_{\mathbf{tr}21} = \frac{1}{n_{\mathbf{ts}_2}} \sum_{i=1}^{n_{\mathbf{ts}_2}} I_{\left(\widehat{h}_{\mathbf{tr}}(x_i) > th\right)}.$$

Under equal priors (0.5) and costs (1), i.e., $th = 0$, we have

$$\widehat{R}_{\mathbf{tr}} = \frac{1}{2}\left(\widehat{e}_{\mathbf{tr}12} + \widehat{e}_{\mathbf{tr}21}\right)$$

2.4 Conclusion and Overview

Usually:

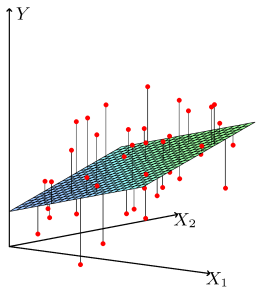
- we do not know the best regression function!
- we do not know the distribution!
- we cannot simulate more data to estimate Risk!

The field is for answering the above questions:

- "**Learning**: is the process of estimating an unknown input-output dependency or structure of a system using a limited number of observations." (Cherkassky and Mulier, 1998). This is the first part of the field, also called **design**
- **Assessment**: how can we assess what we have designed in terms of some measures, e.g., Risk, Error, etc.? This is the second part of the field.

Chapter 3

Linear Models for Regression



3.1 Introduction

We saw that the best regression function is

$$\widehat{Y} = \mathrm{E}[Y|X],$$

In general, we always can write

$$\begin{aligned} Y|X &= \mathrm{E}[Y|X] + \varepsilon \\ &= f(X) + \varepsilon, \end{aligned}$$

where ε is a r.v. and $\mathrm{E}[\varepsilon] = 0$. In linear models, it is assumed that $f(X)$ is linear in X, and the goal is to estimate the coefficients in $f(X)$. Linear models:

- largely developed in statistics community long time ago
- Still are a great tool for prediction and can outperform fancier ones.
- can be applied to transformed features (e.g., if we have $X = \left(X_1, X_2\right)'$, we can make up the feature vector $X = \left(X_1, X_2, X_1^2, X_2^2, X_1 X_2\right)'$, and then assume $f(X)$ is linear in this new X).
- many other methods are generalization to linear models, including Neural Networks and even some methods for classification.

Suppose that the original feature vector is $Z = (Z_1, \ldots, Z_D)'$. In linear models we assume that

$$\begin{aligned} f(X) &= \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p \\ &= \beta' X, \\ X &= (1, X_1, \ldots, X_p)', \\ \beta &= (\beta_0, \ldots, \beta_p), \end{aligned}$$

where 1 accounts for the intercept and $X_1, \ldots, X_p$ can be:

- The components of the original feature vector
- Basis: e.g., $X_1 = Z_1, X_2 = Z_2^2, X_3 = Z_1 Z_2, \ldots$
- Transformation: e.g., $X_1 = \log Z_1, X_2 = \exp[Z_2]$.

The model still is linear in coefficients (or linear in the new features).

Typically, we have N observations; each is (x_i, y_i). So, we have the data matrix and the response values:

$$\mathbf{X}_{N\times p+1} = \begin{pmatrix} (1, x_1)' \\ \vdots \\ (1, x_N)' \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \ldots & x_{1p} \\ \vdots & \vdots & \ddots & \\ 1 & x_{N1} & & x_{Np} \end{pmatrix},$$

$$\mathbf{y}_{N\times 1} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}.$$

3.2 Least Mean Square (LMS)

For any choice β, we have a residual some squares:

$$\begin{aligned} RSS(\beta) &= \sum_{i=1}^{N} (y_i - f(x_i))^2 \\ &= \sum_i (y_i - \beta' x_i)^2 \\ &= \sum_i \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 . \end{aligned}$$

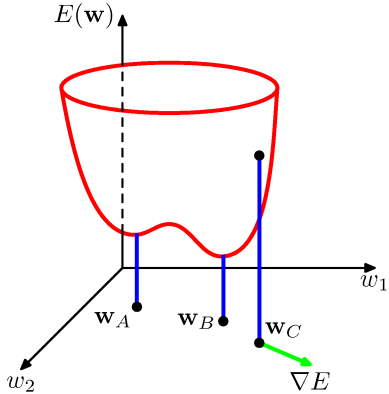
A valid choice of β is to minimize RSS, which can be rewritten in vector form as

$$\begin{aligned} RSS(\beta) &= (\mathbf{y} - \mathbf{X}\beta)' (\mathbf{y} - \mathbf{X}\beta) \\ &= \mathbf{y}'\mathbf{y} - 2\beta'\mathbf{X}'\mathbf{y} + \beta'\mathbf{X}'\mathbf{X}\beta \end{aligned}$$

This is a scalar function of a vector (many variables); how to minimize?

Extremum: Calculus Reminder

- 1st derivative test: $f'(x) = 0$.
- 2nd derivative test: $f''(x) \lesseqgtr 0$.
- saddle points.



In general, to minimize a scalar function E in a vector W, we have to find the point w at which the gradient:

$$\nabla E(W) = \frac{\partial E(W)}{\partial W} = \left(\frac{\partial E(W)}{\partial W_1}, \ldots, \frac{\partial E(W)}{\partial W_p} \right)' = \mathbf{0}'.$$

Then, this is followed by the Hessian matrix test.

Hint: prove, for any matrix A and vector α, that

$$\nabla \alpha' W = \nabla W' \alpha = \alpha \qquad \text{(linear combination)}$$
$$\nabla W' A W = \left[A + A' \right] W. \qquad \text{(quadratic form)}$$

Then, back to minimizing $RSS(\beta)$:

$$\begin{aligned} RSS(\beta) &= \mathbf{y}'\mathbf{y} - 2\beta'\mathbf{X}'\mathbf{y} + \beta'\mathbf{X}'\mathbf{X}\beta \\ \nabla RSS(\beta) &= -2\mathbf{X}'\mathbf{y} + \mathbf{2X}'\mathbf{X}\beta \qquad (\overset{\text{set}}{=} 0) \\ \widehat{\beta} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}. \end{aligned}$$

For a future observation x_0, the prediction $\widehat{y}_0$ is:

$$\widehat{y}_0 = (1, x_0)'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y},$$

and the prediction of the training observation is:

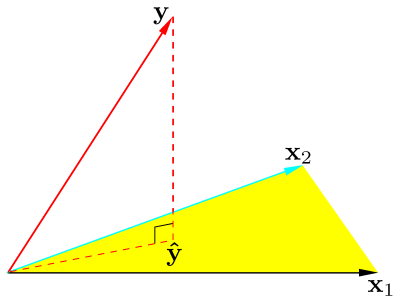
$$\begin{aligned} \widehat{\mathbf{y}} &= \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \\ &= \mathbf{H}\mathbf{y}, \end{aligned}$$

where **H** is called the hat matrix (or the projection matrix). Therefore, the residual error at each observation is:

$$\begin{aligned} \widehat{\varepsilon} &= \widehat{\mathbf{y}} - \mathbf{y} \\ &= \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} - \mathbf{y} \\ &= \left(\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' - \mathbf{I}\right)\mathbf{y}. \end{aligned}$$

Matlab does all of that; look for linear models.

3.2.1 LMS: Geometric Proof



$\mathbf{y} = \mathbf{X}\beta + \varepsilon$:

- can be viewed as linear combinations of vectors in sample space $\mathbb{R}^N$:

$$\mathbf{X}_{N\times(p+1)}\beta = \begin{pmatrix} \mathbf{x}_0 & \dots & \mathbf{x}_p \end{pmatrix}\beta = \mathbf{x}_0\beta_0 + \dots \mathbf{x}_p\beta_p.$$

- $\mathbf{X}$ spans a sub-space of $\mathbb{R}^N$.
- $\widehat{y}$ is in the same space
- We need to minimize the error vector $\widehat{e} = \mathbf{y} - \mathbf{X}\beta$.
- Then, $\widehat{e}$ must be perpendicular on the space $\mathbf{X}$. This nice Geometry can be translated to math as:

$$\mathbf{0} = \mathbf{X}'\left(\mathbf{y} - \mathbf{X}\widehat{\beta}\right)$$
$$\widehat{\beta} = \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{y}$$

3.2.2 LMS: Centered Form

Summary first:

$$
\begin{aligned}
y_1 &= \beta_0 + \beta_1 x_{11} + \cdots + \beta_p x_{1p} + \varepsilon_1, \\
&\vdots \\
y_N &= \beta_0 + \beta_1 x_{N1} + \cdots + \beta_p x_{Np} + \varepsilon_N \\
y_i &= \begin{pmatrix} 1 & x_i' \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p} \end{pmatrix} + \varepsilon_i, \\
\mathbf{y} &= \begin{pmatrix} \mathbf{1}, & \mathbf{X_1} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p} \end{pmatrix} + \varepsilon = \mathbf{X}\beta + \varepsilon \\
RSS &= \varepsilon'\varepsilon = (\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta) = \mathbf{y}'\mathbf{y} - 2\beta'\mathbf{X}'\mathbf{y} + \beta'\mathbf{X}'\mathbf{X}\beta \\
&= \left(\mathbf{y} - \begin{pmatrix} \mathbf{1}, & \mathbf{X}_1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p} \end{pmatrix}\right)' \left(\mathbf{y} - \begin{pmatrix} \mathbf{1}, & \mathbf{X}_1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p} \end{pmatrix}\right) \\
&= \mathbf{y}'\mathbf{y} - 2\begin{pmatrix} \beta_0 & \beta_{1\sim p}' \end{pmatrix} \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}_1'\mathbf{y} \end{pmatrix} + \begin{pmatrix} \beta_0 & \beta_{1\sim p}' \end{pmatrix} \begin{pmatrix} N & \mathbf{1}'\mathbf{X}_1 \\ \mathbf{X}_1'\mathbf{1} & \mathbf{X}_1'\mathbf{X}_1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p} \end{pmatrix} \\
\nabla RSS &= -2\begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}_1'\mathbf{y} \end{pmatrix} + 2\begin{pmatrix} N & \mathbf{1}'\mathbf{X}_1 \\ \mathbf{X}_1'\mathbf{1} & \mathbf{X}_1'\mathbf{X}_1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p} \end{pmatrix} \qquad \text{(not sep.)} \\
\begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_{1\sim p} \end{pmatrix} &= \begin{pmatrix} N & \mathbf{1}'\mathbf{X}_1 \\ \mathbf{X}_1'\mathbf{1} & \mathbf{X}_1'\mathbf{X}_1 \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}_1'\mathbf{y} \end{pmatrix} \\
&= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}
\end{aligned}
$$

Geometric motivation for centered form ($p = 1$):

$$
\begin{aligned}
y_1 &= \alpha + \beta_1 (x_{11} - \overline{x}_1) + \cdots + \beta_p \left(x_{1p} - \overline{x}_p\right) + \varepsilon_1, \\
&\vdots \\
y_N &= \alpha + \beta_1 (x_{N1} - \overline{x}_1) + \cdots + \beta_p \left(x_{Np} - \overline{x}_p\right) + \varepsilon_N, \\
y_i &= \alpha + (x_i - \overline{x})' \beta_{1\sim p} + \varepsilon_i, \\
\alpha &= \beta_0 + \beta_1 \overline{x}_1 + \cdots + \beta_p \overline{x}_p = \beta_0 + \beta'_{1\sim p} \overline{x}.
\end{aligned}
$$

$$
\mathbf{y} = \left(\mathbf{1}, \mathbf{X}_c\right) \begin{pmatrix} \alpha \\ \beta_{1\sim p} \end{pmatrix} + \varepsilon,
$$

$$
\mathbf{X}_c = \mathbf{X}_1 - \mathbf{1}\overline{x}' = \mathbf{X}_1 - \frac{1}{N}\mathbf{1}\mathbf{1}'\mathbf{X}_1 = \left(\mathbf{I} - \frac{1}{N}\mathbf{J}\right)\mathbf{X}_1.
$$

$$
\begin{aligned}
RSS &= \left(\mathbf{y} - \left(\mathbf{1}, \mathbf{X}_c\right) \begin{pmatrix} \alpha \\ \beta_{1\sim p} \end{pmatrix}\right)' \left(\mathbf{y} - \left(\mathbf{1}, \mathbf{X}_c\right) \begin{pmatrix} \alpha \\ \beta_{1\sim p} \end{pmatrix}\right) \\
&= \mathbf{y}'\mathbf{y} - 2\left(\alpha, \beta'_{1\sim p}\right) \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}'_c\mathbf{y} \end{pmatrix} + \left(\alpha, \beta'_{1\sim p}\right) \begin{pmatrix} N & \mathbf{1}'\mathbf{X}_c \\ \mathbf{X}'_c\mathbf{1} & \mathbf{X}'_c\mathbf{X}_c \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_{1\sim p} \end{pmatrix} \\
&= \mathbf{y}'\mathbf{y} - 2\left(\alpha, \beta'_{1\sim p}\right) \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}'_c\mathbf{y} \end{pmatrix} + \left(\alpha, \beta'_{1\sim p}\right) \begin{pmatrix} N & \mathbf{0}' \\ \mathbf{0} & \mathbf{X}'_c\mathbf{X}_c \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_{1\sim p} \end{pmatrix}
\end{aligned}
$$

$$
\nabla RSS = -2 \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}'_c\mathbf{y} \end{pmatrix} + 2 \begin{pmatrix} N & \mathbf{0}' \\ \mathbf{0} & \mathbf{X}'_c\mathbf{X}_c \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_{1\sim p} \end{pmatrix} \tag{3.1}
$$

$$
\begin{pmatrix} \widehat{\alpha} \\ \widehat{\beta}_{1\sim p} \end{pmatrix} = \begin{pmatrix} N & \mathbf{0}' \\ \mathbf{0} & \mathbf{X}'_c\mathbf{X}_c \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}'_c\mathbf{y} \end{pmatrix} = \begin{pmatrix} \frac{1}{N} & \mathbf{0}' \\ \mathbf{0} & \left(\mathbf{X}'_c\mathbf{X}_c\right)^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}'_c\mathbf{y} \end{pmatrix}
$$

Or, simply

$$\mathbf{0} = \begin{pmatrix} \widehat{\alpha} N - \mathbf{1}'\mathbf{y} \\ \mathbf{X}_c'\mathbf{X}_c\widehat{\beta}_{1\sim p} - \mathbf{X}_c'\mathbf{y} \end{pmatrix} \tag{3.2a}$$

$$\widehat{\alpha} = \frac{1}{N}\mathbf{1}'\mathbf{y} = \overline{y}, \tag{3.2b}$$

$$\widehat{\beta}_{1\sim p} = \left(\mathbf{X}_c'\mathbf{X}_c\right)^{-1}\mathbf{X}_c'\mathbf{y} = \left(\frac{\mathbf{X}_c'\mathbf{X}_c}{N-1}\right)^{-1}\frac{\mathbf{X}_c'\mathbf{y}}{N-1} = \widehat{\Sigma}_{XX}^{-1}\widehat{\Sigma}_{XY} \tag{3.2c}$$

$$\widehat{y}_0 = \widehat{\alpha} + (x_0 - \overline{x})'\,\widehat{\beta}_{1\sim p} \tag{3.2d}$$

$$= \overline{y} + (x_0 - \overline{x})'\,\widehat{\Sigma}_{XX}^{-1}\widehat{\Sigma}_{XY}, \tag{3.2e}$$

$$\widehat{\mathbf{y}} = \mathbf{1}\widehat{\alpha} + \mathbf{X}_c\left(\mathbf{X}_c'\mathbf{X}_c\right)^{-1}\mathbf{X}_c'\mathbf{y}. \tag{3.2f}$$

- Eq. (3.2e) is of great surprise; it is the same as Eq. (2.5). **LMS coincides with the best regression function after plug-in parameters estimation in case of multinormal distribution!**
- **This is a source of misconception for some people.**

Example 7 (non-centered & centered) *:*

$$\mathbf{y}=\begin{pmatrix}2\\3\\2\\7\\6\\8\\10\\7\\8\\12\\11\\14\end{pmatrix},\ \mathbf{X}_1=\begin{pmatrix}0&2\\2&6\\2&7\\2&5\\4&9\\4&8\\4&7\\6&10\\6&11\\6&9\\8&15\\8&13\end{pmatrix},\ \mathbf{X}=(\mathbf{1},\mathbf{X}_1)=\begin{pmatrix}1&0&2\\1&2&6\\1&2&7\\1&2&5\\1&4&9\\1&4&8\\1&4&7\\1&6&10\\1&6&11\\1&6&9\\1&8&15\\1&8&13\end{pmatrix}$$

$$\mathbf{X}'\mathbf{X}=\begin{pmatrix}12&52&102\\52&296&536\\102&536&1004\end{pmatrix},$$

$$(\mathbf{X}'\mathbf{X})^{-1}=\begin{pmatrix}0.97476&0.2429&-0.22871\\0.2429&0.16207&-0.11120\\-0.22871&-0.11120&8.3596\times10^{-2}\end{pmatrix},$$

$$\widehat{\beta}=(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}=\begin{pmatrix}5.3711\\3.0123\\-1.2866\end{pmatrix}$$

$$\widehat{y}_0=(1\ x_0')\,\widehat{\beta}=(1\ x_{01}\ x_{02})\,\widehat{\beta}$$
$$=5.3711+3.0123x_{01}-1.2866x_{02}.$$

$$\overline{x}' = \frac{1}{N}\mathbf{1}'\mathbf{X}_1 = \begin{pmatrix} 4.3333 & 8.5 \end{pmatrix}$$

$$\mathbf{X}_c = \mathbf{X}_1 - \mathbf{1}\overline{x}' = \begin{pmatrix} 0 & 2 \\ 2 & 6 \\ 2 & 7 \\ 2 & 5 \\ 4 & 9 \\ 4 & 8 \\ 4 & 7 \\ 6 & 10 \\ 6 & 11 \\ 6 & 9 \\ 8 & 15 \\ 8 & 13 \end{pmatrix} - \begin{pmatrix} 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \\ 4.3333 & 8.5 \end{pmatrix} = \begin{pmatrix} -4.3333 & -6.5 \\ -2.3333 & -2.5 \\ -2.3333 & -1.5 \\ -2.3333 & -3.5 \\ -0.3333 & 0.5 \\ -0.3333 & -0.5 \\ -0.3333 & -1.5 \\ 1.6667 & 1.5 \\ 1.6667 & 2.5 \\ 1.6667 & 0.5 \\ 3.6667 & 6.5 \\ 3.6667 & 4.5 \end{pmatrix}$$

$$\widehat{\alpha} = \overline{y} = 7.5 = \begin{pmatrix} 1 & 4.3333 & 8.5 \end{pmatrix} \begin{pmatrix} 5.3711 \\ 3.0123 \\ -1.2866 \end{pmatrix} = \widehat{\beta}_0 + \widehat{\beta}'_{1\sim 12}\overline{x}$$

$$\widehat{\beta}_{1\sim 12} = \left(\mathbf{X}_c'\mathbf{X}_c\right)^{-1}\mathbf{X}_c'\mathbf{y} = \begin{pmatrix} 70.667 & 94.0 \\ 94.0 & 137.0 \end{pmatrix}^{-1} \begin{pmatrix} 92.003 \\ 107.0 \end{pmatrix} = \begin{pmatrix} 3.0122 \\ -1.2857 \end{pmatrix} = \widehat{\beta}'_{1\sim 12}$$

$$\widehat{y}_0 = \widehat{\alpha} + (x_0 - \overline{x})'\,\widehat{\beta}_{1\sim p}$$

$$= 7.5 + \begin{pmatrix} x_{01} - 4.3333 & x_{02} - 8.5 \end{pmatrix} \begin{pmatrix} 3.0122 \\ -1.2857 \end{pmatrix} = 7.5 + 3.0122\,(x_{01} - 4.3333) - 1.2857\,(x_{02} - 8.5)$$

3.3 Performance

3.3.1 Apparent Performance

also called training or example error (this is what we have minimized):

$$\overline{err} = \frac{1}{N}\sum_i \left(\widehat{y}_i - y_i\right)^2 = \frac{1}{N}\widehat{\varepsilon}'\widehat{\varepsilon} = \frac{1}{N}\left\|\widehat{\varepsilon}\right\|^2$$

3.3.2 Conditional Performance

this is what we really want to minimize:

$$\begin{aligned} err_{\mathbf{tr}} &= \underset{0}{\mathrm{E}}\left(\widehat{y}_0 - y_0\right)^2 && \text{(Risk, MSE, or EPE)} \\ &= \underset{0}{\mathrm{E}}\left[\left(1, x_0\right)'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{y} - y_0\right]^2 \\ &= \underset{x_0}{\mathrm{E}}\,\underset{y_0|x_0}{\mathrm{E}}\left[\left(1, x_0\right)'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{y} - y_0\right]^2 \\ &= \underset{x_0}{\mathrm{E}}\, err_{\mathbf{tr}}\left(x_0\right) \end{aligned}$$

This is conditional on the training set **tr** appearing in the equation as **X** and **y**. In simulation problems, where data comes from a known distribution, we can obtain a very accurate estimate of $err_{\mathbf{tr}}$, using very large **ts** as:

$$err_{\mathbf{tr}} \cong \frac{1}{n_{\mathbf{ts}}}\sum_{i\in\mathbf{ts}}\left(\widehat{y}_i - y_i\right)^2 \qquad \text{(as in Ex. 4)}$$

3.3.3 Unconditional Performance

$$err = \underset{\mathbf{tr}}{\mathrm{E}}\, err_{\mathbf{tr}}$$

We will see how to estimate this from **tr**. This means also that there is

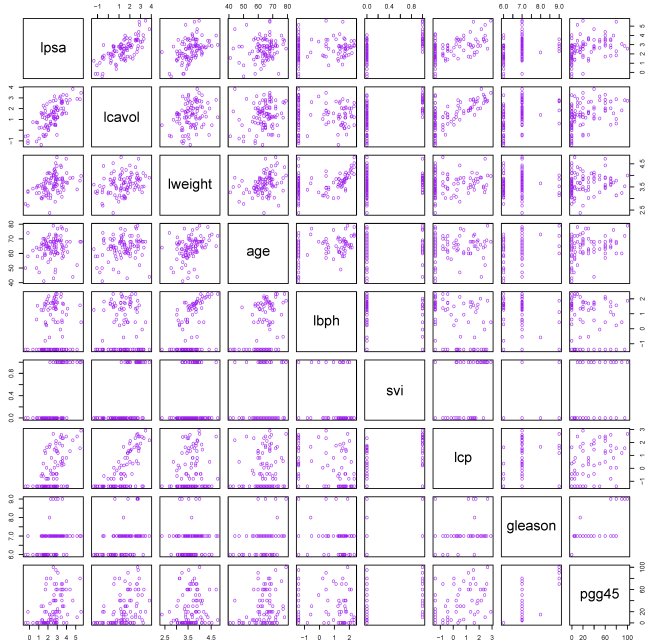
$$\underset{\mathbf{tr}}{\mathrm{Var}}\, err_{\mathbf{tr}},$$

which expresses how stable the regression function is from dataset to another. In simulation problems, we do MC simulation to estimate these quantities as well

$$\underset{\mathbf{tr}}{\mathrm{E}}\, err_{\mathbf{tr}} \cong \frac{1}{M}\sum_{m=1}^{M} err_{\mathbf{tr}_m}$$

$$\mathrm{Var}\, err_{\mathbf{tr}} \cong \frac{1}{M-1}\sum_{m=1}^{M}\left(err_{\mathbf{tr}_m} - \frac{1}{M}\sum_{m=1}^{M} err_{\mathbf{tr}_m}\right)^2$$

Example 8 (prostate cancer) *:*



"The variables are log cancer volume (`lcavol`*), log prostate weight (*`lweight`*), age, log of the amount of benign prostatic hyperplasia (*`lbph`*), seminal vesicle invasion (*`svi`*), log of capsular penetration (*`lcp`*), Gleams score (*`gleason`*), and percent of Gleams scores 4 or 5 (*`pgg45`*)."*

- *First: plot, understand your dataset, and withdraw some measures before any model building.*
- *Eg., `svi` is binary, `gleams` is ordered categorical, both `lcavol` and `lcp` show a strong relationship with the response `lpsa`, and with each other.*
- *Data visualization is a stand alone topic and course.*

TABLE 3.1. *Correlations of predictors in the prostate cancer data.*

	lcavol	lweight	age	lbph	svi	lcp	gleason
lweight	0.300						
age	0.286	0.317					
lbph	0.063	0.437	0.287				
svi	0.593	0.181	0.129	−0.139			
lcp	0.692	0.157	0.173	−0.089	0.671		
gleason	0.426	0.024	0.366	0.033	0.307	0.476	
pgg45	0.483	0.074	0.276	−0.030	0.481	0.663	0.757

TABLE 3.2. *Linear model fit to the prostate cancer data. The Z score is the coefficient divided by its standard error (3.12). Roughly a Z score larger than two in absolute value is significantly nonzero at the $p = 0.05$ level.*

Term	Coefficient	Std. Error	Z Score
Intercept	2.46	0.09	27.60
lcavol	0.68	0.13	5.37
lweight	0.26	0.10	2.75
age	−0.14	0.10	−1.40
lbph	0.21	0.10	2.06
svi	0.31	0.12	2.47
lcp	−0.29	0.15	−1.87
gleason	−0.02	0.15	−0.15
pgg45	0.27	0.15	1.74

- *"We fit a linear model to the log of prostate-specific antigen, `lpsa`, after first standardizing the predictors to have unit variance. We randomly split the dataset into a training set of size 67 and a test set of size 30."*
- *"The mean prediction error on the test data is 0.521. In contrast, prediction using the mean training value of `lpsa` has a test error of 1.057, which is called the base error rate. Hence the linear model reduces the base error rate by about 50%. We will return to this example later to compare various selection and shrinkage methods."*
- *The prediction using the mean training value is:*

$$\widehat{y}_0 = \widehat{\alpha} = \widehat{\beta}_0 = \frac{1}{N}\sum_{i=1}^{N} y_i; \qquad (\beta_{1\sim p} = \mathbf{0})$$

  i.e., using the sample average of the training set as if you do not have any additional information from $\mathbf{X}$.
- *the meaning of* $\widehat{\beta}_i$ *is this: a unit increase in predictor* X_i *results in an increase of* $\widehat{\beta}_i$ *in the response* Y_i.

3.4 Data Preprocessing and Transformation

Standardize your predictors, e.g., to unit variance, so that no variable is more dominant than others. (when testing use of course the inverse scaling). Prove that a linear mapping in the form

$$a_i X_i + b_i,\ i = 1, 2$$

will not deform the correlation between the variables X_1, X_2. **Applying linear transformation to each variable of a data matrix accounts for moving the center of scatter plot then scaling without preserving the aspect ratio.**

One form of linear transformation is mapping all predictors to $[L, H]$ (usually $[-1, 1]$ as `mapminmax` in Matlab (but it assumes $P \times p$ matrix not $N \times p$), or $[0, 1]$ as in parallel cords) by:

$$
\begin{aligned}
X_{new} &= (X - X_{\min})\left(\frac{H-L}{X_{\max}-X_{\min}}\right) + L \\
&= \left(\frac{H-L}{X_{\max}-X_{\min}}\right)X + \frac{LX_{\max} - HX_{\min}}{X_{\max}-X_{\min}}
\end{aligned}
$$

To put it in the form $X_{new} = \frac{X-a}{b}$:

$$
\begin{aligned}
b &= \frac{X_{\max}-X_{\min}}{H-L} \\
\frac{-a}{b} &= \frac{LX_{\max} - HX_{\min}}{X_{\max}-X_{\min}} \\
a &= \frac{HX_{\min} - LX_{\max}}{H-L} \\
X_{new} &= \frac{X - \left(\frac{HX_{\min}-LX_{\max}}{H-L}\right)}{\left(\frac{X_{\max}-X_{\min}}{H-L}\right)}
\end{aligned}
$$

Linear transformation can be done by standardizing to zero sample mean and unit sample variance, by

$$
X_{new} = \frac{X - \overline{X}}{\widehat{\sigma}_X}
$$

Any shifting can be expressed as:

$$\begin{aligned}
\mathbf{X}^{shifted} &= \begin{pmatrix} 1 & x_{11} & \dots & x_{1p} \\ \vdots & \vdots & \ddots & \\ 1 & x_{N1} & & x_{Np} \end{pmatrix} - \begin{pmatrix} 0 & c_1 & \dots & c_p \\ \vdots & \vdots & \ddots & \\ 0 & c_1 & & c_p \end{pmatrix} \\
&= \mathbf{X} - \mathbf{1}c' && (\mathbf{X} - \texttt{repmat}(c', [N, 1])) \\
&= \mathbf{X} - \mathbf{X} \underbrace{\begin{pmatrix} 1 \\ \vdots \\ 0 \end{pmatrix}}_{\mathbf{1}} c' \\
&= \mathbf{X}\left(\mathbf{I} - \begin{pmatrix} 0 & c_1 & \dots & c_p \\ 0 & 0 & & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & & 0 \end{pmatrix}\right) = \mathbf{X}\begin{pmatrix} 1 & -c_1 & \dots & -c_p \\ 0 & 1 & & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & & 1 \end{pmatrix} \\
&= \mathbf{XT}.
\end{aligned}$$

A special case shifting is centering with the sample mean:

$$c' = \left(0, \frac{1}{N}\mathbf{1}'\mathbf{X}_1\right) \qquad (\texttt{mean}(\texttt{X}_1))$$

Also, any linear transformation (without shifting) can be expressed as

$$\begin{aligned}\mathbf{X}^{scaled} &= \left(\mathbf{1}, \mathbf{X}_1\mathbf{S}_1\right) \\ &= \left(\mathbf{1}, \mathbf{X}_1\right)\begin{pmatrix} 1 & \mathbf{0}' \\ \mathbf{0} & \mathbf{S}_1 \end{pmatrix} \\ &= \mathbf{XS}\end{aligned}$$

A special case of this transformation is when we scale ONLY each feature (what is used usually); i.e.,

$$\begin{aligned}\mathbf{S}_1 &= \begin{pmatrix} d_1 & \dots & 0 \\ \vdots & \ddots & \\ 0 & & d_p \end{pmatrix} \\ &= \text{diag}\left(d_1, \dots, d_p\right)\end{aligned}$$

Therefore, a general linear transformation for the features, including shifting and scaling, is given by

$$\begin{aligned}\mathbf{X}^{trans} &= \mathbf{XTS} \\ &= \mathbf{XH},\end{aligned}$$

where we do, first, shifting $\mathbf{T}$, followed by scaling $\mathbf{S}$.

How transformation affects prediction in LM?

$$\widehat{y}_0 = x_0' \left(\mathbf{X}'\mathbf{X}\right)^{-1} \mathbf{X}'\mathbf{y}.$$

With transformation, we replace **X** by **XH**. Therefore,

$$\begin{aligned}
\widehat{\beta}^{trans} &= \left[(\mathbf{XH})'(\mathbf{XH})\right]^{-1}(\mathbf{XH})'\mathbf{y} \\
&= \left[\mathbf{H}'\mathbf{X}'\mathbf{XH}\right]^{-1}\mathbf{H}'\mathbf{X}'\mathbf{y} \\
&= \mathbf{H}^{-1}\left[\mathbf{X}'\mathbf{X}\right]^{-1}\mathbf{H}'^{-1}\mathbf{H}'\mathbf{X}'\mathbf{y} \\
&= \mathbf{H}^{-1}\left[\mathbf{X}'\mathbf{X}\right]^{-1}\mathbf{X}'\mathbf{y}, \\
\widehat{y}_0^{trans} &= \left(x_0'\mathbf{H}\right)\widehat{\beta}^{trans} \\
&= x_0'\mathbf{H}\mathbf{H}^{-1}\left[\mathbf{X}'\mathbf{X}\right]^{-1}\mathbf{X}'\mathbf{y} \\
&= \widehat{y}_0 \qquad \text{(without transformation)}
\end{aligned}$$

3.5 Bias-Variance Decomposition

For any regression function $f(X)$, its conditional risk was given by

$$\mathrm{E}\left[Y - f(x)\right]^2 = \sigma^2_{Y|X} + \left(\mathrm{E}\left[Y|X=x\right] - f(x)\right)^2,$$

where we minimized it by choosing $f(X) = \mathrm{E}[Y|X=x]$. Now, for any learning function $f_{\mathbf{tr}}(x_0) = \widehat{y}_0$, the (conditional) risk are given by (look at Sec. 3.3)

$$err_{\mathbf{tr}}(x_0) = \sigma^2_{Y|X} + \left(\mathrm{E}\, y_0|x_0 - \widehat{y}_0\right)^2$$

$$err_{\mathbf{tr}} = \underbrace{\mathop{\mathrm{E}}_{x_0} \sigma^2_{Y|X=x_0}}_{\text{minimum risk (irreducible)}} + \mathop{\mathrm{E}}_{x_0}\left(\widehat{y}_0 - \mathop{\mathrm{E}}_{y_0|x_0} y_0\right)^2$$

$\mathrm{E}_{\mathbf{tr}}\, err_{\mathbf{tr}} =$

$$= \mathop{\mathrm{E}}_{x_0} \sigma^2_{Y|X=x_0} + \mathop{\mathrm{E}}_{x_0}\mathop{\mathrm{E}}_{\mathbf{tr}}\left(\widehat{y}_0 - \mathop{\mathrm{E}}_{y_0|x_0} y_0\right)^2$$

$$= \mathop{\mathrm{E}}_{x_0} \sigma^2_{Y|X=x_0} + \mathop{\mathrm{E}}_{x_0}\mathop{\mathrm{E}}_{\mathbf{tr}}\left(\left(\widehat{y}_0 - \mathop{\mathrm{E}}_{\mathbf{tr}}\widehat{y}_0\right) + \left(\mathop{\mathrm{E}}_{\mathbf{tr}}\widehat{y}_0 - \mathop{\mathrm{E}}_{y_0|x_0} y_0\right)\right)^2$$

$$= \mathop{\mathrm{E}}_{x_0} \sigma^2_{Y|X=x_0} + \mathop{\mathrm{E}}_{x_0}\left[\mathop{\mathrm{E}}_{\mathbf{tr}}\left(\widehat{y}_0 - \mathop{\mathrm{E}}_{\mathbf{tr}}\widehat{y}_0\right)^2 + \left(\mathop{\mathrm{E}}_{\mathbf{tr}}\widehat{y}_0 - \mathop{\mathrm{E}}_{y_0|x_0} y_0\right)^2 + \underbrace{\mathrm{cov}}_{0}\right]$$

$$= \mathop{\mathrm{E}}_{x_0} \sigma^2_{Y|X=x_0} + \mathop{\mathrm{E}}_{x_0}\left[\mathop{\mathrm{Var}}_{\mathbf{tr}}\widehat{y}_0 + Bias^2_{\mathbf{tr}}\left(\widehat{y}_0\right)\right].$$

In particular for linear models, it can be shown that if the data follows exactly a full linear model, then using the right number of features or more will result in an unbiased model.

3.5.1 Bias for Underfitting

Assume the right model is

$$\begin{aligned}\mathrm{E}\left[y|X\right] &= X'\beta \\ &= \left(X_1', X_2'\right)\begin{pmatrix}\beta_1 \\ \beta_2\end{pmatrix} = X_1'\beta_1 + X_2'\beta_2.\end{aligned}$$

and we used the reduced model (underfitting)

$$\mathrm{E}\left[y|X_1\right] = X_1'\beta_1^*$$

$$\begin{aligned}\underset{\mathbf{tr}}{\mathrm{E}}\,\widehat{y}_0 &= \underset{\mathbf{tr}}{\mathrm{E}}\, x_{01}'\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\mathbf{y} \\ &= x_{01}'\underset{\mathbf{X}}{\mathrm{E}}\,\underset{\mathbf{y}|\mathbf{X}}{\mathrm{E}}\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\mathbf{y} \\ &= x_{01}'\underset{\mathbf{X}}{\mathrm{E}}\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\underset{\mathbf{y}|\mathbf{X}}{\mathrm{E}}\,\mathbf{y} \\ &= x_{01}'\underset{\mathbf{X}}{\mathrm{E}}\left[\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\left(\mathbf{X}_1\beta_1 + \mathbf{X}_2\beta_2\right)\right] \\ &= x_{01}'\left(\beta_1 + \underset{\mathbf{X}}{\mathrm{E}}\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\mathbf{X}_2\beta_2\right) \\ &= x_{01}'\beta_1 + x_{01}'\underset{\mathbf{X}}{\mathrm{E}}\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\mathbf{X}_2\beta_2 \\ &\neq x_{01}'\beta_1 + x_{02}'\beta_2 \\ &= \mathrm{E}\, y_{0.}|x_0\end{aligned}$$

Therefore, there is a bias in the underfitting. **When using the right model, i.e., if $\beta_2 = 0$, the bias is zero.**

3.5.2 Bias for Right model and Overfitting

$$\mathrm{E}\left[y|X\right] = X_1'\beta_1 = X_1'\beta_1 + \underbrace{X_2'\beta_2}_{=0}$$

We use the overfitting model

$$\begin{aligned}
\mathrm{E}\left[y|X\right] &= X'\beta = X_1'\beta_1^* + X_2'\beta_2^* \\
\underset{\mathbf{tr}}{\mathrm{E}}\,\widehat{y}_0 &= \underset{\mathbf{tr}}{\mathrm{E}}\, x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{y} \\
&= \underset{\mathbf{X}}{\mathrm{E}}\, x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}' \underset{\mathbf{y}|\mathbf{X}}{\mathrm{E}}\,\mathbf{y} \\
&= \underset{\mathbf{X}}{\mathrm{E}}\, x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{X}_1\beta_1 \\
&= \underset{\mathbf{X}}{\mathrm{E}}\, x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\left(\mathbf{X}_1\beta_1 + \underbrace{\mathbf{X}_2\beta_2}_{=0}\right) \\
&= \underset{\mathbf{X}}{\mathrm{E}}\, x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{X}\beta \\
&= x_0'\beta \\
&= x_{01}'\beta_1 + \underbrace{x_{02}'\beta_2}_{=0} \\
&= x_{01}'\beta_1 \\
&= \mathrm{E}\, y_0|x_0
\end{aligned}$$

Therefore the overfitted model is unbiased.

3.5.3 Variance

$$\begin{aligned}
\underset{\mathbf{tr}}{\operatorname{Var}}\,\widehat{y}_0 &= \underset{\mathbf{X}}{\operatorname{E}}\underset{\mathbf{y}|\mathbf{X}}{\operatorname{Var}}\, x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{y} + \underset{\mathbf{X}}{\operatorname{Var}}\underset{\mathbf{Y}|\mathbf{X}}{\operatorname{E}}\, x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{y} \\
&= \underset{\mathbf{X}}{\operatorname{E}}\left[\left(x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\right)\left(\sigma^2_{Y|X}\mathbf{I}_{N\times N}\right)\left(\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}x_0\right)\right] \\
&+ \underset{\mathbf{X}}{\operatorname{Var}}\left[x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{X}\beta\right] \\
&= \underset{\mathbf{X}}{\operatorname{E}}\sigma^2_{Y|X}x_0'\left(\mathbf{X}'\mathbf{X}\right)^{-1}x_0 + \underset{\mathbf{X}}{\operatorname{Var}}\, x_0'\beta \\
&= x_0'\underset{\mathbf{X}}{\operatorname{E}}\left[\sigma^2_{Y|X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\right]x_0.
\end{aligned}$$

To simplify things, lets assume for a momonet that $\sigma^2_{Y|X} = \sigma^2$.

$$\underset{\mathbf{tr}}{\operatorname{Var}}\,\widehat{y}_0 = \sigma^2 x_0'\underset{\mathbf{X}}{\operatorname{E}}\left[\left(\mathbf{X}'\mathbf{X}\right)^{-1}\right]x_0.$$

To simplify more, assume that $\mathrm{E}_{\mathbf{X}}\, X = 0$ (so X is centered, which is not a big deal), therefore

$$\underset{\mathbf{X}}{\operatorname{E}}\left(\frac{1}{N-1}\mathbf{X}'\mathbf{X}\right) = \Sigma_X.$$

We can do this ad-hoc approximation (just to get the picture)

$$\underset{\mathbf{X}}{\operatorname{E}}\left(\mathbf{X}'\mathbf{X}\right)^{-1} \approx \frac{1}{N}\Sigma_X^{-1}.$$

Therefore,

$$
\begin{aligned}
\underset{\mathbf{tr}}{\operatorname{Var}}\,\widehat{y}_0 &= \frac{\sigma^2}{N} x_0' \Sigma_X^{-1} x_0, \\
\underset{x_0}{\operatorname{E}}\,\underset{\mathbf{tr}}{\operatorname{Var}}\,\widehat{y}_0 &= \frac{\sigma^2}{N}\,\underset{x_0}{\operatorname{E}}\, x_0' \Sigma_X^{-1} x_0 \\
&= \frac{\sigma^2}{N}\,\underset{x_0}{\operatorname{E}}\,\operatorname{trace}\left[x_0 x_0' \Sigma_X^{-1}\right] \\
&= \frac{\sigma^2}{N}\,\operatorname{trace}\,\underset{x_0}{\operatorname{E}}\left[x_0 x_0' \Sigma_X^{-1}\right] \\
&= \frac{\sigma^2}{N}\,\operatorname{trace}\left[\Sigma_X \Sigma_X^{-1}\right] \\
&= \frac{\sigma^2}{N}\,\operatorname{trace}\,\mathbf{I}_{p\times p} \\
&= \sigma^2 \frac{p}{N}
\end{aligned}
$$

3.5.4 Bias-Variance: illustration, model complexity, and model selection

V. Imp

Apparent (training) error $\overline{err}$, conditional error $err_{\mathbf{tr}}$, mean error err

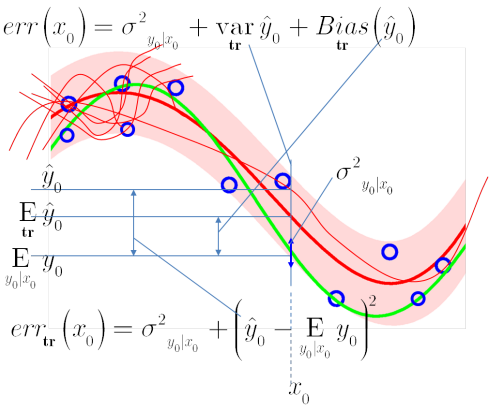
$$\overline{err} = \frac{1}{n_{\mathbf{tr}}} \sum_{i \in \mathbf{tr}} \left(\widehat{y}_i - y_i\right)^2 \cong \frac{1}{n_{\mathbf{ts}}} \sum_{i \in \mathbf{ts}} \left(\widehat{y}_i - y_i\right)^2$$

$$err_{\mathbf{tr}} = \mathop{\mathrm{E}}_{x_0, y_o}\left[\left(\widehat{y}_0 - y_0\right)^2\right] = \mathop{\mathrm{E}}_{x_0}\left[\mathop{\mathrm{E}}_{y_0|x_0}\left(\widehat{y}_0 - y_0\right)^2\right]$$

$$= \mathop{\mathrm{E}}_{x_0}\left[\sigma^2_{Y|X=x_0} + \left(\widehat{y}_0 - \mathop{\mathbf{E}}_{\boldsymbol{y_0|x_0}} \boldsymbol{y_0}\right)^2\right]$$

$$err = \mathop{\mathrm{E}}_{\mathbf{tr}} err_{\mathbf{tr}} \cong \frac{1}{M} \sum_{m=1}^{M} err_{\mathbf{tr}_m}$$

$$= \mathop{\mathrm{E}}_{x_0}\left[\sigma^2_{Y|X=x_0} + \mathop{\mathrm{E}}_{\mathbf{tr}}\left(\widehat{y}_0 - \mathop{\mathbf{E}}_{\mathbf{tr}} \widehat{\boldsymbol{y}}_0\right)^2 + \left(\mathop{\mathbf{E}}_{\mathbf{tr}} \widehat{\boldsymbol{y}}_0 - \mathop{\mathbf{E}}_{\boldsymbol{y_0|x_0}} \boldsymbol{y_0}\right)^2\right]$$



- Performance P. 51.
- Why err not $err_{\mathbf{tr}}$?
- Learning function stability.
- estimators are good for err not $err_{\mathbf{tr}}$.
- Concepts are transferable to other models than LM.

Apparent (training) error $\overline{err}$, conditional error $err_{\mathbf{tr}}$, mean error err

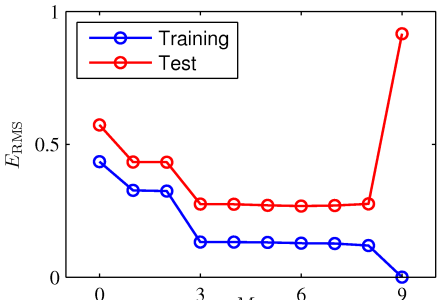
$$\overline{err} = \frac{1}{n_{\mathbf{tr}}} \sum_{i \in \mathbf{tr}} \left(\widehat{y}_i - y_i\right)^2 \cong \frac{1}{n_{\mathbf{ts}}} \sum_{i \in \mathbf{ts}} \left(\widehat{y}_i - y_i\right)^2$$

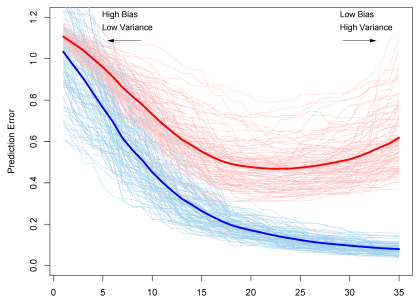
$$err_{\mathbf{tr}} = \mathop{\mathrm{E}}_{x_0, y_o}\left[\left(\widehat{y}_0 - y_0\right)^2\right] = \mathop{\mathrm{E}}_{x_0}\left[\mathop{\mathrm{E}}_{y_0|x_0}\left(\widehat{y}_0 - y_0\right)^2\right]$$

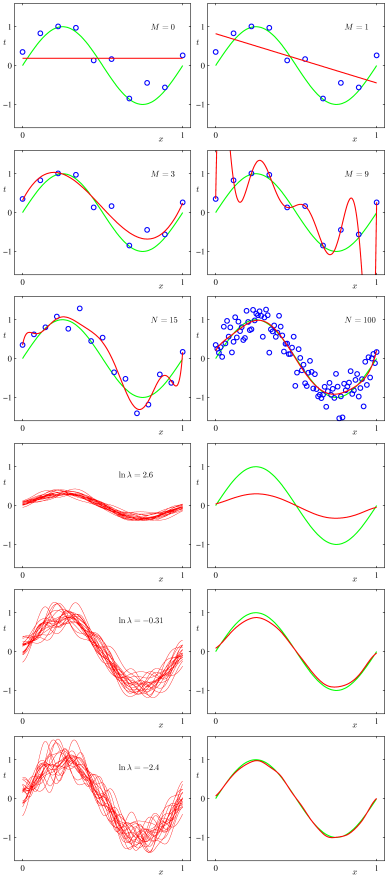
$$= \mathop{\mathrm{E}}_{x_0}\left[\sigma^2_{Y|X=x_0} + \left(\widehat{y}_0 - \mathop{\mathbf{E}}_{\mathbf{y_0|x_0}} \mathbf{y_0}\right)^2\right]$$

$$err = \mathop{\mathrm{E}}_{\mathbf{tr}} err_{\mathbf{tr}} \cong \frac{1}{M} \sum_{m=1}^{M} err_{\mathbf{tr}_m}$$

$$= \mathop{\mathrm{E}}_{x_0}\left[\sigma^2_{Y|X=x_0} + \mathop{\mathrm{E}}_{\mathbf{tr}}\left(\widehat{y}_0 - \mathop{\mathbf{E}}_{\mathbf{tr}} \widehat{\boldsymbol{y}}_0\right)^2 + \left(\mathop{\mathbf{E}}_{\mathbf{tr}} \widehat{\boldsymbol{y}}_0 - \mathop{\mathbf{E}}_{\mathbf{y_0|x_0}} \mathbf{y_0}\right)^2\right]$$

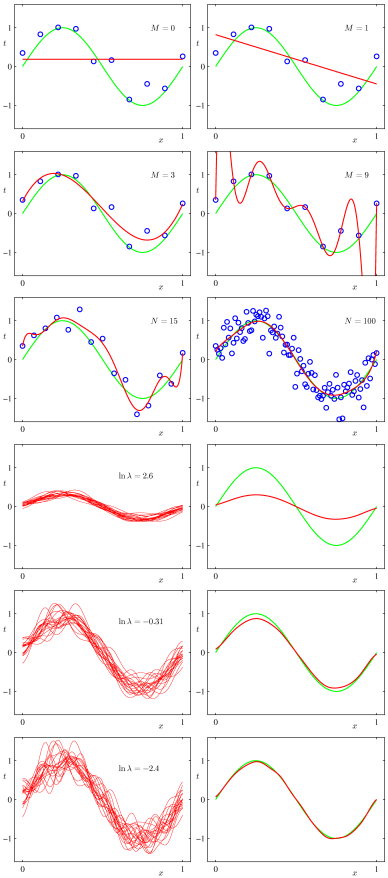


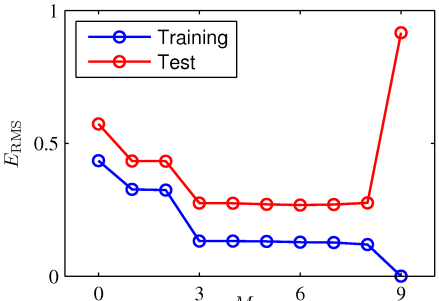


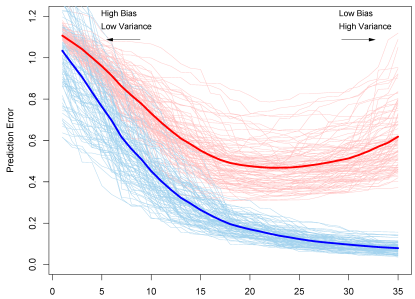


Notes:

- All methods have complexity parameter(s) to tune (p in LM). The right complexity is chosen via Cross-Validation (next chapters).
- **Overfitting:** is increasing model complexity to adapt **too much** to the training dataset. It causes the behavior of single $err_{\mathbf{tr}}$ curve. NOT the bias-variance decomposition. Rather, the bias-variance decomposition and the behavior of err are caused as well by the same effect of increasing complexity.
- But with increasing the training set size we get better performance; **So it depends on the ratio** p/N.
- HW. prove that $\mathrm{E}_{\mathbf{tr}}\, err_{\mathbf{tr}} \neq$ error of $\mathrm{E}_{\mathbf{tr}}\, \widehat{y}_0$.
- $\overline{err}$ must $\searrow$ with complexity; and $\mathrm{Var}_{\mathbf{tr}}\, \overline{err}$ is "in general" $\searrow$; why? However, $err_{\mathbf{tr}}$ is "in general" has a minimum and $\mathrm{Var}_{\mathbf{tr}}\, err_{\mathbf{tr}}$ is "in general" $\nearrow$.







3.6 Reducing Complexity by Subset Selection

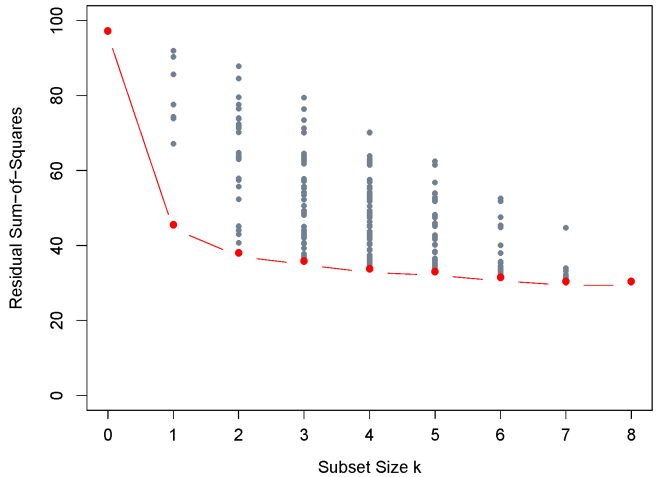
There are two reasons why we are often not satisfied with the least squares estimates:

- "The first is prediction accuracy: the least squares estimates often have low bias but large variance. Prediction accuracy can sometimes be improved by shrinking or setting some coefficients to zero. By doing so we sacrifice a little bit of bias to reduce the variance of the predicted values, and hence may improve the overall prediction accuracy."
- "The second reason is interpretation. With a large number of predictors, we often would like to determine a smaller subset that exhibit the strongest effects. In order to get the "big picture", we are willing to sacrifice some of the small details."

Common methods for subset selection:

- Best subset selection.
- Forward- and Backward-Stepwise regression.
- Forward-Stagewise regression.

Example 9 (Best subset selection (prostate cancer)) .



- *Best subset regression finds for each* $k \in 0, 1, 2, \ldots, p$ *the subset of size* k *that gives smallest RSS.*
- *The best subset of size 2, e.g., needs not include the variable that was in the best subset of size 1. Therefore, the red lower boundary is necessarily decreasing.*
- *The question of how to choose* k *involves the tradeoff between bias and variance, usually done by CV.*

3.7 Reducing Complexity by Regularization (Shrinkage Methods)

"*By retaining a subset of the predictors and discarding the rest, subset selection produces a model that is interpretable and has possibly lower prediction error than the full model. However, because it is a discrete process variables are either retained or discarded—it often exhibits high variance, and so does not reduce the prediction error of the full model. Shrinkage methods are more continuous, and do not suffer as much from high variability.*"

However, another very important factor is time complexity. Subset selection is very time consuming w.r.t. regularization; it can even be untractable in many real life problems with high dimensions.

- Ridge regression
- Lasso regression (Tibshirani)
- Least Angle Regression (LAR) (Efron)

Hint: Great picture can be seen by studying the connection of these methods with each other; deferred to the advanced course!

3.7.1 Ridge Regression

$$\widehat{\beta}^{\text{ridge}} = \underset{\beta}{\arg\min} \underbrace{\left\{ \underbrace{\sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2}_{\text{RSS}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\text{penalty term}} \right\}}_{\text{new loss to be minimized}} \tag{3.3}$$

- "*Here $\lambda \geq 0$ is a complexity parameter that controls the amount of shrinkage: the larger the value of λ, the greater the amount of shrinkage. The coefficients are shrunk toward zero (and each other). The idea of penalizing by the sum-of-squares of the parameters is also used in neural networks, where it is known as weight decay (Ch. 11).*"

- "*The ridge solutions are not equivariant under scaling of the inputs, and so one normally standardizes the inputs before solving (3.3).*"

- "*In addition, notice that the intercept β_0 has been left out of the penalty term. Penalization of the intercept would make the procedure depend on the origin chosen for Y; that is, adding a constant c to each of the targets y_i would not simply result in a shift of the predictions by the same amount c.*"

- Eq. way to write (3.3) is:

$$\widehat{\beta}^{\text{ridge}} = \underset{\beta}{\arg\min} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \tag{3.4a}$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 \leq t. \tag{3.4b}$$

Now, minimizing the loss in (3.3) as done in centered LM (it could be just LM as we will see, but centered LM gives us great insight):

$$\widehat{\beta}^{\text{ridge}} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}.$$

$$RSS^{\text{ridge}} = RSS + \lambda \sum_{j=1}^{p} \beta_j^2.$$

$$\nabla RSS^{\text{ridge}} = \nabla RSS + \nabla \left(\lambda \sum_{j=1}^{p} \beta_j^2 \right).$$

Substituting from (3.1), and $\nabla RSS^{\text{ridge}} \overset{\text{set}}{=} \mathbf{0}$, we get

$$\begin{aligned}
\mathbf{0} &= -2 \begin{pmatrix} \mathbf{1}'\mathbf{y} \\ \mathbf{X}_c'\mathbf{y} \end{pmatrix} + 2 \begin{pmatrix} N & \mathbf{0}' \\ \mathbf{0} & \mathbf{X}_c'\mathbf{X}_c \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_{1\sim p} \end{pmatrix} + 2\lambda \begin{pmatrix} 0 \\ \beta_{1\sim p} \end{pmatrix} \\
&= \begin{pmatrix} N\alpha - \mathbf{1}'\mathbf{y} \\ \left(\mathbf{X}_c'\mathbf{X}_c + \lambda \mathbf{I}\right)\beta_{1\sim p} - \mathbf{X}_c'\mathbf{y} \end{pmatrix} \\
\widehat{\alpha} &= \frac{1}{N}\mathbf{1}'\mathbf{y} = \bar{\mathbf{y}}, \\
\widehat{\beta}_{1\sim p}^{\text{ridge}} &= \left(\mathbf{X}_c'\mathbf{X}_c + \lambda \mathbf{I}\right)^{-1} \mathbf{X}_c'\mathbf{y}.
\end{aligned}$$

- We keep varying λ and choose the value that minimizes the total estimated error.
- Notice that, when $\lambda = 0$, this is the case of ordinary LM obtained in (3.2).
- Notice also that, we had to use the centered model to find a closed form solution. However, in Neural Networks, we will use the same regularization formula but find β_0 numerically. We will not need to center the data, but of course we need to standardize, e.g., to $[-1, 1]$, as mentioned above.
- Matlab does ridge regression: `ridge`.
- Python (scikit-learn): `Ridge`.

Example 10 (prostate cancer using regularization) .

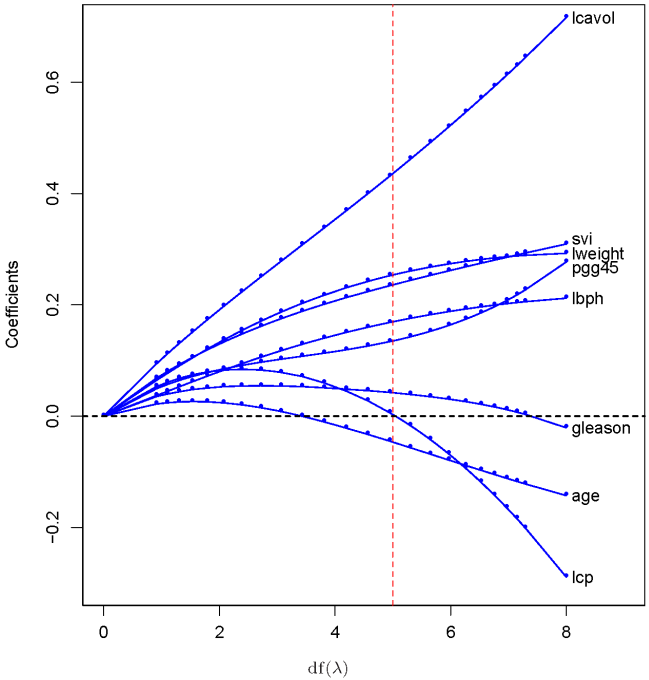


FIGURE 3.8. *Profiles of ridge coefficients for the prostate cancer example, as the tuning parameter* λ *is varied. Coefficients are plotted versus* $\mathrm{df}(\lambda)$*, the effective degrees of freedom. A vertical line is drawn at* $\mathrm{df} = 5.0$*, the value chosen by cross-validation.*

- Imagine we plot vs. $1/\lambda$ NOT λ. This is to have the same trend with complexity p.
- In the book, they plot vs. $\texttt{df}(\lambda)$, which involves understating PCA; hence, deferred:

$$\widehat{\mathbf{y}} = \mathbf{1}\widehat{\alpha} + \mathbf{X}_c\widehat{\beta}^{\text{ridge}} \tag{3.5}$$

$$= \mathbf{1}\widehat{\alpha} + \mathbf{X}_c\left(\mathbf{X}_c'\mathbf{X}_c + \lambda\mathbf{I}\right)^{-1}\mathbf{X}_c'\mathbf{y} \tag{3.6}$$

$$\mathrm{df}(\lambda) = \mathrm{tr}\left[\mathbf{X}_c\left(\mathbf{X}_c'\mathbf{X}_c + \lambda\mathbf{I}\right)^{-1}\mathbf{X}_c'\right]. \tag{3.7}$$

- After PCA, it can be shown that:

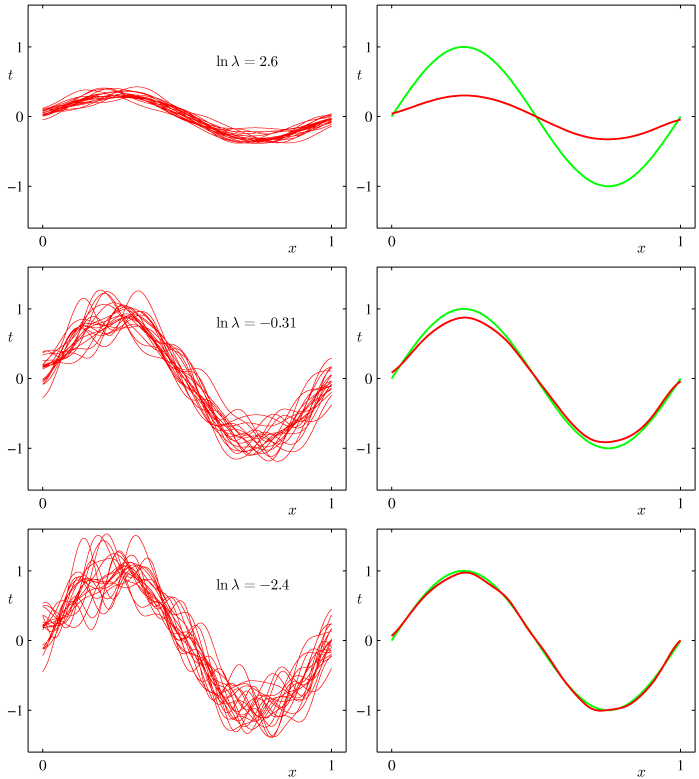
$$\mathrm{df}(\lambda) \searrow \lambda, \quad \mathrm{df}(0) = p, \quad \mathrm{df}(\infty) = 0.$$

Hint: We could have proceeded without centered model as:

$$
\begin{aligned}
\mathbf{0} &= -\mathbf{X}'y + \begin{pmatrix} N & \mathbf{1}'\mathbf{X}_1 \\ \mathbf{X}_1'\mathbf{1} & \mathbf{X}_1'\mathbf{X}_1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p+1} \end{pmatrix} + \lambda \begin{pmatrix} 0 \\ \beta_{1\sim p+1} \end{pmatrix} \\
&= -\mathbf{X}'y + \begin{pmatrix} N & \mathbf{1}'\mathbf{X}_1 \\ \mathbf{X}_1'\mathbf{1} & \mathbf{X}_1'\mathbf{X}_1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p+1} \end{pmatrix} + \begin{pmatrix} 0 & \mathbf{0}' \\ \mathbf{0} & \lambda\mathbf{I} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p+1} \end{pmatrix} \\
&= -\mathbf{X}'y + \begin{pmatrix} N & \mathbf{1}'\mathbf{X}_1 \\ \mathbf{X}_1'\mathbf{1} & \mathbf{X}_1'\mathbf{X}_1 + \lambda\mathbf{I} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_{1\sim p+1} \end{pmatrix} \\
\widehat{\beta} &= \begin{pmatrix} N & \mathbf{1}'\mathbf{X}_1 \\ \mathbf{X}_1'\mathbf{1} & \mathbf{X}_1'\mathbf{X}_1 + \lambda\mathbf{I} \end{pmatrix}^{-1} \mathbf{X}'\mathbf{y}.
\end{aligned}
$$

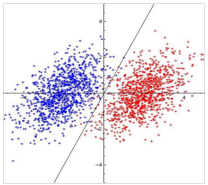
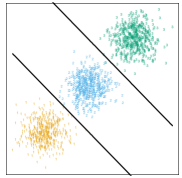
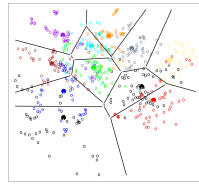
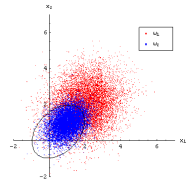
However: we get big insight about the effect of λ from the centered model => df concept.

Example 11 (Regularization for Example on P. 68) *: The model complexity $(p+1)$is set to 25.*

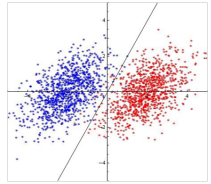


Chapter 4

Linear Models for Classification

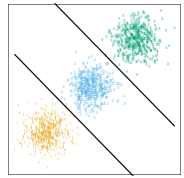





Introduction



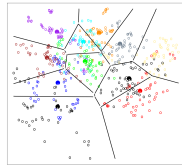
- Recall regression, when we assumed that the regression function is linear $Y = \beta' X + \epsilon$. Similarly, we mean by "*linear methods of classification*" those methods that produce a "linear decision boundary" between pairs of classes, $\mathcal{G}_i$, $\mathcal{G}_j$:

$$S_{ij}(X) = \left\{ x | \, \beta'_{ij} X = 0 \right\}.$$



Hint: This does not mean that there has to be a boundary between each two classes!

- The problem is reduced to estimating βs, as it was in LM.
- **Notice**: Nonlinear decision boundaries (surfaces) in the original feature space are linear in the expanded feature space. But, how to find the right expansion/transformation. For example, in the case of $p = 2$, we can do:

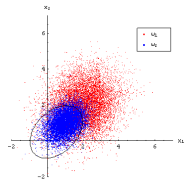


$$\mathcal{T} : \mathcal{R}^2 \mapsto \mathcal{R}^5$$

$$X^{\texttt{new}} = \left(X_1^{\texttt{new}}, X_2^{\texttt{new}}, X_3^{\texttt{new}}, X_4^{\texttt{new}}, X_5^{\texttt{new}}\right) = \mathcal{T}(X) = \left(X_1, X_2, X_1 X_2, X_1^2, X_2^2\right)$$

$$h(X) = X_1 + X_2 + X_1 X_2 + X_1^2 + X_2^2 = \sum_{i=1}^{i=5} X_i^{\texttt{new}} = h^{\texttt{new}}(X^{\texttt{new}}) = h^{\texttt{new}}(\mathcal{T}(X)) = h^{\texttt{new}} \cdot \mathcal{T}(X)$$

$$h = h^{\texttt{new}} \cdot \mathcal{T}$$



- First, let's revisit the best decision boundaries (Sec. 2.2.2), and see a very simple 3-class problem with no equal costs!

Example 12 (3-class Multinormal) *: Notebook*

$f_i \sim \mathcal{N}(\mu_i, \Sigma_i)$, $\Sigma_i = I$, $\pi_i = \frac{1}{3}$, $\quad i = 1,2,3$
$\mu_1 = (-a, 0)$, $\mu_2 = (a, 0)$, $\mu_3 = (0, \sqrt{3}a)$ $\quad$ ***(symmetry?)***
$L_{ii} = 0$, $L_{12} = L_{21} = L_{23} = L_{32} = 1$, $L_{13} = L_{31} = L$.

$$\widehat{G}(x) = \arg\min_{g \in \mathcal{G}} \sum_{k=1}^{K} L\left(\mathcal{G}_k, g\right) \Pr\left(\mathcal{G}_k | X = x\right)$$

$$f(X)\lambda(\mathcal{G}_1) = L_{11}\pi_1 f_1(X) + L_{21}\pi_2 f_2(X) + L_{31}\pi_3 f_3(X)$$
$$f(X)\lambda(\mathcal{G}_2) = L_{12}\pi_1 f_1(X) + L_{22}\pi_2 f_2(X) + L_{32}\pi_3 f_3(X)$$
$$f(X)\lambda(\mathcal{G}_3) = L_{13}\pi_1 f_1(X) + L_{23}\pi_2 f_2(X) + L_{33}\pi_3 f_3(X)$$

$$f_i(X) = \frac{1}{2\pi} e^{\frac{-1}{2}((x-\mu_i)'(x-\mu_i))}$$

We will solve only for S_{12} and the rest is HW.

First: *consider the fully symmetric case:* $L = 1$.
Step 1: $D_{12} = \left\{x |\ \lambda(\mathcal{G}_1) = \lambda(\mathcal{G}_2)\right\}$.

$$0 = \left(f_2(X) + L f_3(X)\right) - \left(f_1(X) + f_3(X)\right) = f_2(X) - f_1(X$$
$$= \frac{1}{2\pi} e^{\frac{-1}{2}((x_1-a)^2 + x_2^2)} - \frac{1}{2\pi} e^{\frac{-1}{2}((x_1+a)^2 + x_2^2)} \quad \Rightarrow x_1 = 0.$$

Step 2: $R_{12} = \left\{x |\ \left(\lambda(\mathcal{G}_1) = \lambda(\mathcal{G}_2)\right)|_{D_{12}} < \lambda(\mathcal{G}_3)|_{D_{12}}\right\}$.

$$f_2(X) + f_3(X) < f_1(X) + f_2(X) \qquad (|_{D_{12}})$$
$$\frac{1}{2\pi} e^{\frac{-1}{2}((x_1-0)^2 + (x_2-\sqrt{3}a)^2)} < \frac{1}{2\pi} e^{\frac{-1}{2}((x_1-a)^2 + x_2^2)} \qquad (|_{x_1=0})$$
$$x_2 < a/\sqrt{3}.$$

Step 3: $S_{12} = D_{12} \cap R_{12} = \left\{x |\ x_1 = 0 \ \&\ x_2 < a/\sqrt{3}\right\}$.

Second, *general L: if S_{12} exists it will be:*

$$0 = e^{\frac{-1}{2}((x_1-a)^2 + x_2^2)} - e^{\frac{-1}{2}((x_1+a)^2 + x_2^2)} + (L-1)e^{\frac{-1}{2}\left(x_1^2 + (x_2-\sqrt{3}a)^2\right)}$$
$$0 = e^{\frac{-1}{2}(x_1^2 + x_2^2 + a^2)}\left(e^{ax_1} - e^{-ax_1} + (L-1)e^{(\sqrt{(3)}ax_2 - a^2)}\right)$$
$$x_2 = \frac{1}{\sqrt{3}}(a - x_1) - \frac{1}{\sqrt{3}a}\log\left(\frac{1-L}{e^{2ax_1} - 1}\right).$$

@@@ This page still needs elaboration from Bishop P.180 and connection to Hastie and Tibs.

- Linear decision boundaries can be established in two different paths:
  1. Modeling "discriminant functions" $(1-\lambda)$:

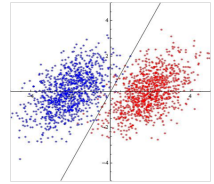
$$1-\lambda(\mathcal{G}_i)=\delta_i(x)=\beta_i' X,\ i=1,\cdots,K,\ X=(1,\cdots)'.$$
$$\widehat{G}(x)=\arg\min\lambda(\mathcal{G}_i)=\arg\max\delta_i(x).$$
$$S_{ij}=\left\{x\mid \beta_i' X-\beta_j' X=0\right\}\cap R_{ij}$$

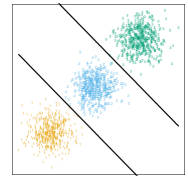
     Here come: regression of indicators, LDA, QDA, RDA, LR, etc.

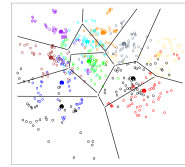
  2. Modeling decision boundaries $h_{ij}(X)$ directly as a hyperplane; i.e., a normal vector r and a cut point x_0.

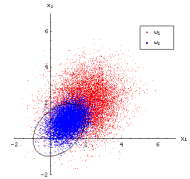
     Here come: perceptron, optimal separating hyper plane, SVM, etc.

- Same concepts studied in LM apply here: bias-variance trade-off, regularization, etc.
- Which method is better? Same question and always same answer!









4.1 Discriminant Functions

Theorem 13 *General Linear discriminant (score) functions, with any monotone transformation $\mathcal{T}$:*

$$\delta_i(x) = \mathcal{T}(\beta_i' X),\ i = 1, \cdots, K,$$
$$\widehat{G}(x) = \arg\max \delta_i(x),$$

where, in general the intercept is absorbed in the first feature vector $X = (1, \cdots)'$, have the following properties:

1. *all decision surfaces are linear.*
2. *all decision regions are singly connected and convex.*

Proof.

1. If a decision surface S_{ij} exists, then $S_{ij} = \{x |\ \delta_i(x) = \delta_j(x)\}$. Then,

$$\begin{aligned} \mathcal{T}(\beta_i' X) = \mathcal{T}(\beta_j' X) &\equiv \beta_i' X = \beta_j' X \quad \text{(monotonecity)} \\ &\equiv (\beta_i' - \beta_j') X = 0. \quad \text{(linear)} \end{aligned}$$

2. Consider the decision region $\mathcal{R}_k$, and the two points $x_a, x_b \in \mathcal{R}_k$, then consider the generic point x_c on the line segment between these two points

$$\begin{aligned} x_c &= \lambda x_a + (1-\lambda) x_b \\ \delta_k(x_c) &= \beta_k' x_c \\ &= \lambda \beta_k' x_a + (1-\lambda)\beta_k' x_b \\ &= \lambda \delta_k(x_a) + (1-\lambda)\delta_k(x_b) \\ &> \lambda \delta_i(x_a) + (1-\lambda)\delta_i(x_b)\ \forall i \qquad (x_a,\ x_b \in \mathcal{R}_k) \\ &= \lambda \beta_i' x_a + (1-\lambda)\beta_i' x_b \\ &= \beta_i' x_c \\ &= \delta_i(x_c). \end{aligned}$$

Hence, $x_c \in \mathcal{R}_k$, which proves convexity and singly connected in one step and completes the proof. ∎

4.1.1 Linear Regression of an Indicator Matrix

First, we introduce multiple output regression

Theorem 14 (Multiple Output Regression) *has an LMS solution that is equivalent to the LMS solution for each individual output problem (the solutions decouple): $Y = X'\beta + \varepsilon$ I.e., for K outputs, $k = 1, \ldots, K$:*

$$\begin{aligned} Y_k &= \beta_{0k} + X_1\beta_k + \ldots + X_p\beta_{pk} + \varepsilon_k \\ &= \beta_{0k} + \sum_{j=1}^{p} X_j\beta_{jk} + \varepsilon_k \\ &= f_k(X) + \varepsilon_k \\ (Y_1 \cdots Y_k) &= \left(X'\mathbf{B}_1 \cdots X'\mathbf{B}_k\right) + (\varepsilon_1\ \varepsilon_2 \cdots \varepsilon_k) \\ &= X'\mathbf{B} + (\varepsilon_1\ \varepsilon_2 \cdots \varepsilon_k)\,. \end{aligned}$$

and for N observations:

$$\mathbf{Y}_{N\times k} = \mathbf{X}_{N\times(p+1)}\mathbf{B}_{(p+1)\times k} + \mathbf{E}_{N\times k}.$$

The LMS solution will be

$$\begin{aligned} \mathbf{B}_k &= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}_k \\ \widehat{\mathbf{B}} &= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}. \end{aligned}$$

Proof.

$$\begin{aligned} \mathrm{RSS}(\mathbf{B}) &= \sum_{k=1}^{K}\sum_{i=1}^{N}\left(y_{ik} - f_k(x_i)\right)^2 && (4.1)\\ &= \sum_{k=1}^{K}\mathbf{E}_k'\mathbf{E}_k && (4.2)\\ &= \mathrm{tr}\,\mathbf{E}'\mathbf{E} && (4.3)\\ &= \mathrm{tr}\left[(\mathbf{Y}-\mathbf{X}\mathbf{B})'(\mathbf{Y}-\mathbf{X}\mathbf{B})\right]. && (4.4) \end{aligned}$$

But, (4.2) is sum minimized if each term $\mathbf{E}_k'\mathbf{E}_k$ is minimized separately since each is positive. Hence, the problem is reduced to K individual LMS problems. The proof is complete. ∎

Lemma 15 (Properties of Regression of Indicator) *: Formalizing the classification problem as multiple out regression problem of indicator discriminant functions $\delta_k(x) = I_{\mathcal{G}(x)=k}$ as:*

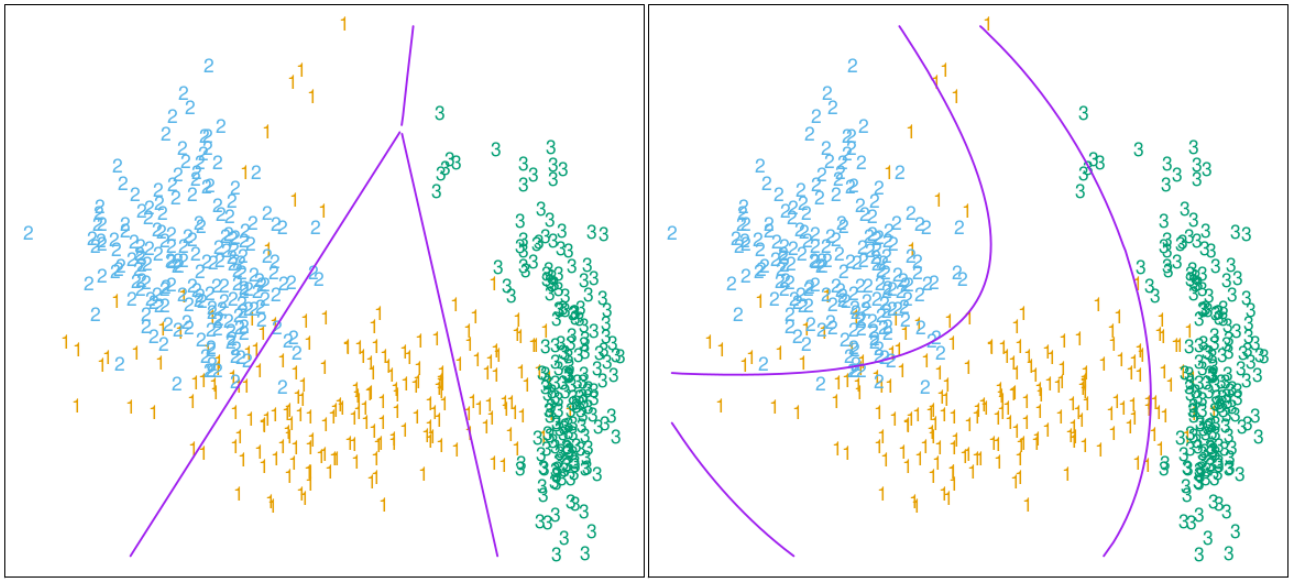
$$\begin{aligned}
\delta_k(x) &= X'\mathbf{B}_k + \varepsilon_k, \\
\mathbf{Y} &= \mathbf{X}'\mathbf{B} + \mathbf{E}, \\
\widehat{\mathbf{B}} &= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}, \\
\widehat{y}_{1\times K} &= x'_{1\times(p+1)}\mathbf{B}_{(p+1)\times K} \\
\widehat{\mathcal{G}}(x) &= \arg\max_k y_k,
\end{aligned}$$

where each row of the response is $\mathbf{Y}_{N\times K}$, where each row has only single 1. I.e., $\mathbf{Y}_{N\times k}1_{k\times 1} = 1_{N\times 1}$., has the following properties:

1. *The expectation of the response is the posterior probability.*
2. *For any observation x_i, $\sum_k \widehat{y}_k(x_i) = 1$ as well as $\sum_k y_k(x_i)$.*

Proof. trivial ∎

Example 16 (Indicator regression in original and expanded space) *:*
$(X_1, X_2) \mapsto (X_1, X_2, X_1X_2, X_1^2, X_2^2)$



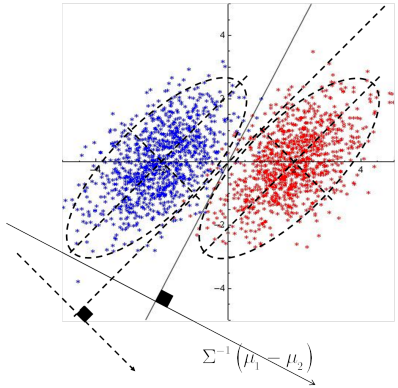
4.1.1.1 Multicollinearity Problem

Effect on regression (“support” argument); projecting on largest PC.
Effect on regression (separation argument); projecting on smallest PC (Q and O image example from Duda and Hart.)

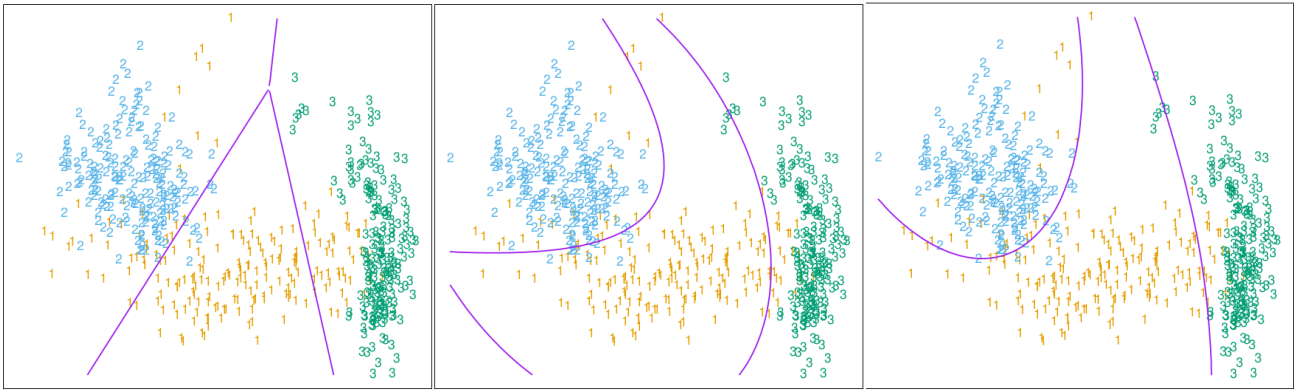
4.1.2 LDA: revisiting, emphasizing, and more insight

$$\left[\Sigma^{-1}\left(\mu_2-\mu_1\right)\right]'\cdot\left(x-\left[\frac{1}{2}\left(\mu_2+\mu_1\right)-\frac{th}{\left(\mu_2-\mu_1\right)'\Sigma^{-1}\left(\mu_2+\mu_1\right)}\left(\mu_2-\mu_1\right)\right]\right)=0$$

$$w'\left(x-x_0\right)=0.$$



4.1.3 LDA in Extended Space vs. QDA



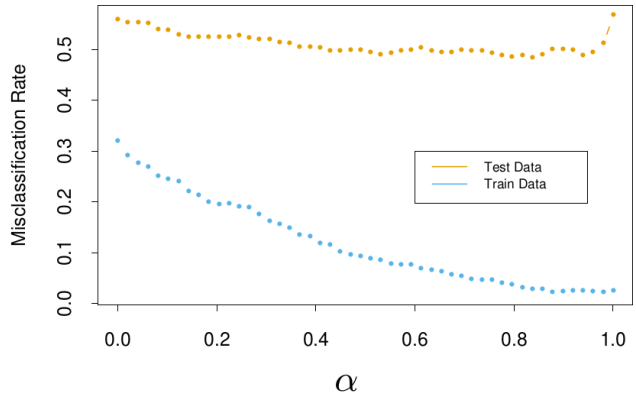
- "*For this figure and many similar figures in the book we compute the decision boundaries by and exhaustive contouring method. We compute the decision rule on a fine lattice of points, and then use contouring algorithms to compute the boundaries.*"
-

4.1.4 Regularized Discriminant Analysis (RDA)

$$\widehat{\Sigma}_k(\alpha) = \alpha\widehat{\Sigma}_k + (1-\alpha)\widehat{\Sigma}, \tag{4.5}$$

$$\widehat{\Sigma}(\gamma) = \gamma\widehat{\Sigma} + (1-\gamma)\sigma^2\mathbf{I}. \tag{4.6}$$

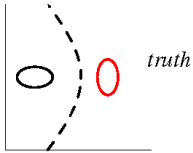
Example 17 (RDA on Vowel dataset) .

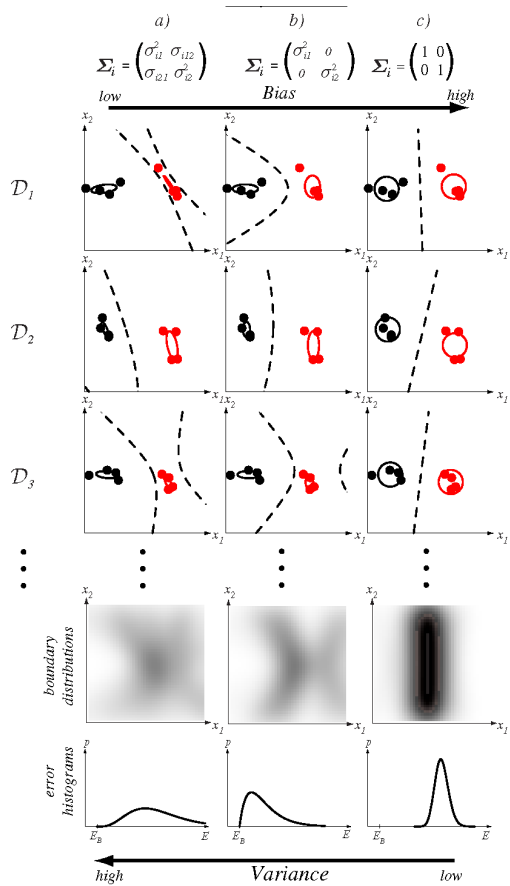


- *The optimum occurs around* $\alpha = 0.9$*, closer to QDA.*
- *This*

Revisiting bias-variance trade-off

- Remember case 1, case 2, case 3 (P. 24).
- 3 different complexities:
  - *a*) is case 3 (QDA).
  - *b*) is case 1, less complex than LDA (how to estimate only diagonals; trivial!)
  - *c*) is very naive.





4.1.5 Principal Component Analysis (PCA)

PCA is:

- for dimension reduction and **linear** data summary.
- NOT a discrimination rule.
- but has strong connection to discrimination.

PCA: two different questions with the same answer:

- what is the best linear representation of data?
- what are directions of largest spread?

4.1.6 LDA vs. PCA

One may conjecture that the decision boundary has the direction of the largest principal component! Sounds reasonable; let's investigate. If so then, the perpendicular direction to the decision boundary ($\Sigma^{-1}\left(\mu_2-\mu_1\right)$) has the direction of the smallest principal component. Now

$$\begin{aligned}\Sigma &= V\Lambda V', \quad V'V=\mathbf{I}\\ &= v_1v_1'\lambda_1+v_2v_2'\lambda_2,\\ \Sigma v_2 &= v_2\lambda_2\end{aligned}$$

Our conjecture leads to

$$\begin{aligned}\Sigma^{-1}\left(\mu_2-\mu_1\right) &= v_2c\\ \mu_2-\mu_1 &= \Sigma v_2c\\ &= v_2\lambda_2c,\end{aligned}$$

which means that the line connecting the two means (which is arbitrary) has the direction of the smallest principal component; which is not mandatory of course.

Let's see Mathematica Notebook:

4.1.7 Mahalanobis' Distance Classifier

The covariance matrix (population or sample) is positive semidefinite (prove as homework):

$$\begin{aligned}\Sigma &= U\Lambda U' \\ &= \left(u_1 \cdots u_p\right)\begin{pmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_p\end{pmatrix}\begin{pmatrix}u_1' \\ \vdots \\ u_p'\end{pmatrix} \\ &= \lambda_1 u_1 u_1' + \cdots + \lambda_p u_p u_p',\end{aligned}$$

It is straight forward to show that (for full rank Σ):

$$\begin{aligned}\lambda_i &= \sigma_i^2 \text{ in the direction of } u_i \\ \Sigma^{-1} &= U\Lambda^{-1}U'\end{aligned}$$

Distance to class centroid may weighted by inverse of spread (small spread in one direction makes a data point far from the class centroid in that direction): $\left(x-\mu\right)' u_i/\sigma_i$. So, the whole distance should be

$$\begin{aligned}\delta^2\left(x,\mu\right) &= \sum_{i=1}^{p}\left(\left(x-\mu\right)' u_i/\sigma_i\right)^2 \\ &= \sum_{i=1}^{p}\left(\left(x-\mu\right)' u_i/\sigma_i\right)\left(u_i'\left(x-\mu\right)/\sigma_i\right) \\ &= \left(x-\mu\right)'\left(\frac{1}{\sigma_1^2}u_1u_1' + \cdots + \frac{1}{\sigma_p^2}u_pu_p'\right)\left(x-\mu\right) \\ &= \left(x-\mu\right)'\Sigma^{-1}\left(x-\mu\right),\end{aligned}$$

which is the Mahalanobis' distance between x and μ (the class centroid) wrt to the matrix Σ (the covariance matrix of the data).

Then it is natural to classify based on the closest class centroid to the testing point x @@@

$$\widehat{G}(x) = \omega_k | k = \arg\min_k \left(x - \mu_k\right)' \Sigma_k^{-1} \left(x - \mu_k\right)$$

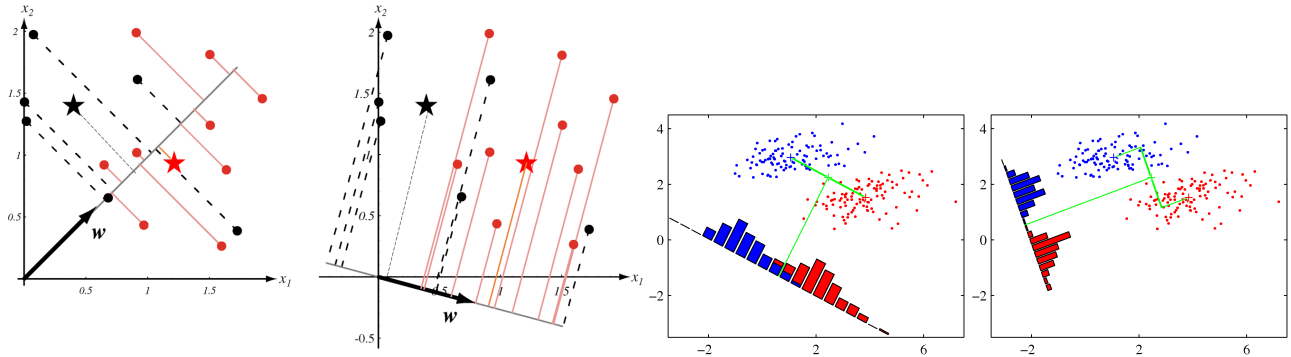
For $K = 2$, and $\Sigma_1 = \Sigma_2 = \Sigma$ this reduces to

$$\left(x - \mu_2\right)' \Sigma^{-1} \left(x - \mu_2\right) - \left(x - \mu_1\right)' \Sigma^{-1} \left(x - \mu_1\right) \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} th,$$

which is the same as LDA.

4.1.8 Fisher Discriminant Analysis (FDA)

4.1.8.1 2-Class Problem



Projected data is $y = w'x$. Let's maximize

$$J(w) = \frac{(\Delta\text{means})^2}{\text{spread}} = \frac{\left(\overline{y}_2 - \overline{y}_1\right)^2}{\sum_{y_i \epsilon \omega_1}\left(y_i - \overline{y}_1\right)^2 + \sum_{y_i \epsilon \omega_2}\left(y_i - \overline{y}_2\right)^2}$$

$$\overline{y}_1 = \frac{1}{n_1}\Sigma_{y_i \epsilon \omega_1} y_i = \frac{1}{n_1}\Sigma_{i=1}^{n_1} w' x_i = w' \overline{x}_1$$

$$\left(\overline{y}_2 - \overline{y}_1\right)^2 = \left(w'\left(\overline{x}_2 - \overline{x}_1\right)\right)^2 = w' \underbrace{\left(\overline{x}_2 - \overline{x}_1\right)\left(\overline{x}_2 - \overline{x}_1\right)'}_{S_B} w$$

Of course S_B is singular with rank 1, and called Between-Class.

$$\sum_{y_i \epsilon \omega_1} \left(y_i - \overline{y}_1\right)^2 = \sum_{i=1}^{n_1} \left(w' x_i - w' \overline{x}_1\right)^2$$

$$= \sum_{i=1}^{n_1} w' \left(x_i - \overline{x}_1\right) \left(x_i - \overline{x}_1\right)' w$$

$$= w' \sum_{i=1}^{n_1} \left(x_i - \overline{x}_1\right) \left(x_i - \overline{x}_1\right)' w = w' S_1 w$$

$$w' S_1 w + w' S_2 w = w' S_W w$$

$$J(w) = \frac{w' S_B w}{w' S_W w}.$$

$$\nabla j(w) = \frac{\left(2 S_B w\right)\left(w' S_W w\right) - \left(w' S_B w\right)\left(2 S_W w\right)}{\left(w' S_W w\right)^2}$$

$$\left(S_B w\right)\left(w' S_W w\right) = \left(w' S_B w\right)\left(S_W w\right).$$

$S_B w$ has $(\overline{x}_2 - \overline{x}_1)$ direction and hence $(S_W w)$. Then

$$S_W w \propto \left(\overline{x}_2 - \overline{x}_1\right)$$

$$w \propto S_W^{-1}\left(\overline{x}_2 - \overline{x}_1\right)$$

Then the decision rule is

$$\left(S_W^{-1}\left(\overline{x}_2 - \overline{x}_1\right)\right)' x \underset{G_2}{\overset{G_1}{\gtrless}} \text{some value}.$$

Recall LDA (with Normal assumption with $\Sigma_1 = \Sigma_2 = \Sigma$):

$$\left[\Sigma^{-1}\left(\mu_2 - \mu_1\right)\right]'\left(x - x_0\right) \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} 0$$

$$\left[\Sigma^{-1}\left(\mu_2 - \mu_1\right)\right]' x \underset{\mathcal{G}_2}{\overset{\mathcal{G}_1}{\gtrless}} \text{some value, where}$$

$$\begin{aligned}
\widehat{\Sigma} &= \left(\frac{1}{n_1 + n_2 - 2}\right) \sum_{k=1}^{2} \sum_{i} \left(x_i - \overline{x}_k\right)\left(x_i - \overline{x}_k\right)' \\
&= \left(\frac{1}{n_1 + n_2 - 2}\right) S_W, \\
\widehat{\mu}_1 &= \overline{x}_1, \\
\widehat{\mu}_2 &= \overline{x}_2
\end{aligned}$$

This explains why LDA works even if the data is neither linearly separable nor following Normal distribution. **The LDA is always optimal in Fisher's sense**.

Let's see same Mathematica Notebook again for connection between PCA, LDA, and FDA (for Fisher (not Flexible) Discriminant Analysis).

4.1.8.2 K-Class Problem

FDA also maximizes:

$$J(w) = \frac{w' S_B w}{w' S_W w},$$

This is a well known mathematical problem called generalized eigenvalue problem, whose solution is

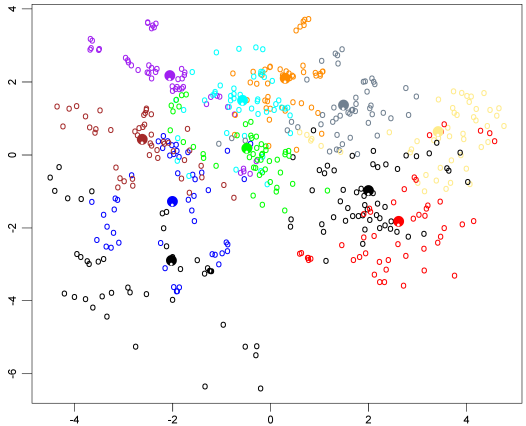
$$S_W^{-1} S_B w = \lambda w,$$

so w is the first eigenvector for the matrix $S_W^{-1} S_B$. This is called generalized eigenvalue problem.
S_B is called between-class covariance matrix,
S_W is called within-class covariance matrix.
In case of 2-class problem

$$\begin{aligned}
S_B &= n_1 (\overline{x}_1 - \overline{x}) (\overline{x}_1 - \overline{x})' + n_2 (\overline{x}_2 - \overline{x}) (\overline{x}_2 - \overline{x})' \\
&= n_1 \left(\overline{x}_1 - \frac{n_1 \overline{x}_1 + n_2 \overline{x}_2}{n} \right) \left(\overline{x}_1 - \frac{n_1 \overline{x}_1 + n_2 \overline{x}_2}{n} \right)' + \cdots \\
&= \frac{n_1}{n^2} (n_2 \overline{x}_1 - n_2 \overline{x}_2) (n_2 \overline{x}_1 - n_2 \overline{x}_2)' + \cdots \\
&= \frac{n_1 n_2^2}{n^2} (\overline{x}_1 - \overline{x}_2) (\overline{x}_1 - \overline{x}_2)' + \cdots \\
&= \left(\frac{n_1 n_2^2}{n} + \frac{n_1^2 n_2}{n} \right) (\overline{x}_1 - \overline{x}_2) (\overline{x}_1 - \overline{x}_2)' \\
&= n_1 n_2 (\overline{x}_1 - \overline{x}_2) (\overline{x}_1 - \overline{x}_2)',
\end{aligned}$$

different from the S_B defined above. However, the maximizer is the same, of course, since it is just a factor $(n_1 n_2)$.

More on S_W and S_B for K-class problem:



$$S_T = \sum_i (x_i - \overline{x})(x_i - \overline{x})'$$

$$= \sum_i (x_i - \overline{x})(x_i - \overline{x})'$$

$$S_B = \sum_{k=1}^{K} n_k (\overline{x}_k - \overline{x})(\overline{x}_k - \overline{x})'$$

$$S_W = \sum$$

Connection to AUC:
Show that this projection maximizes the AUC (for normal dist).

4.1.9 Logistic Regression

4.1.9.1 Generalized Linear Models (GLM)

4.1.9.2 Fitting Logistic Regression

4.1.9.3 Logistic Regression vs. LDA

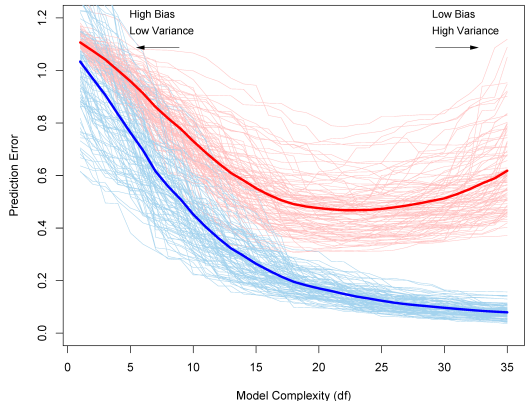
4.2 Separating Hyperplanes

4.2.1 Rosenblatt's Perceptron Learning Algorithm

4.2.2 Optimal Separating Hyperplane

Chapter 7

Model Assessment and Selection



7.1 “Dimensionality” from Different Perspectives

7.1.1 Curse of Dimensionality

Example 18 *:*
A point is chosen randomly in a disk of radius 1. Then

$$f_{XY}(x,y) = \begin{cases} \frac{1}{\pi} & x^2+y^2 \le 1 \\ 0 & otherwise. \end{cases}$$

What is $P(R \le r)$? In general, if $X = (X_1, \ldots, X_p)$ is uniformly distributed over an area A, then

$$\begin{aligned} f_X(x_1, \ldots, x_p) &= \frac{1}{|A|}, \\ P(X \in B) &= \int_B f_X \, dx \\ &= \frac{|B|}{|A|} \end{aligned}$$

$$\begin{aligned} P(R \le r) &= P\left(x^2+y^2 \le r\right) \\ &= \frac{\pi r^2}{\pi} \\ &= r^2. \end{aligned}$$

Later, we can do it differently: $r = x^2+y^2$, which is a function of 2 r.v. (transformation).

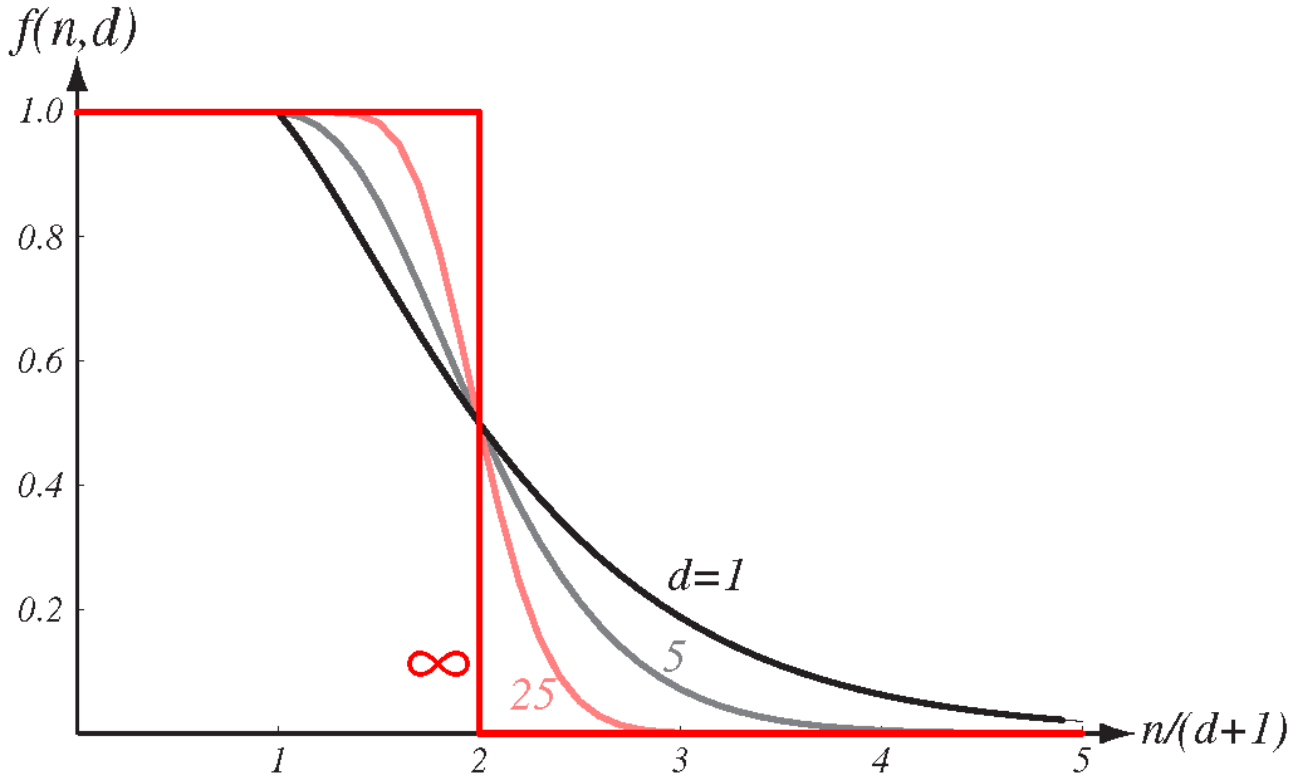
What is f_X?

$$\begin{aligned} f_X(x) &= \frac{1}{\pi} \int_{-\sqrt{1-x^2}}^{\sqrt{1-x^2}} dy \\ &= \frac{2}{\pi}\sqrt{1-x^2}, \ -1 \le x \le 1. \end{aligned}$$

For p-dimensional spheres, $P(R \le r) = r^p$, which means that data will be on the surface!!

7.1.2 Vapnik-Chervonenkis (VC) Dimension

7.1.3 Cover's Theorem

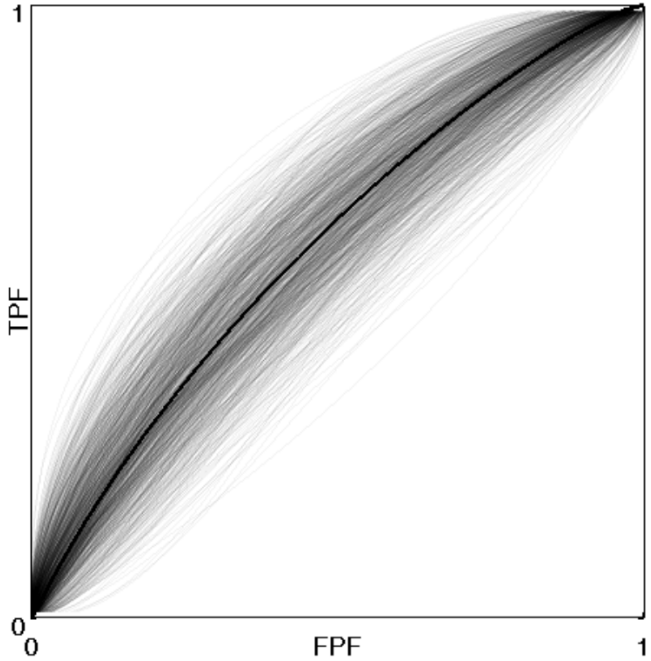


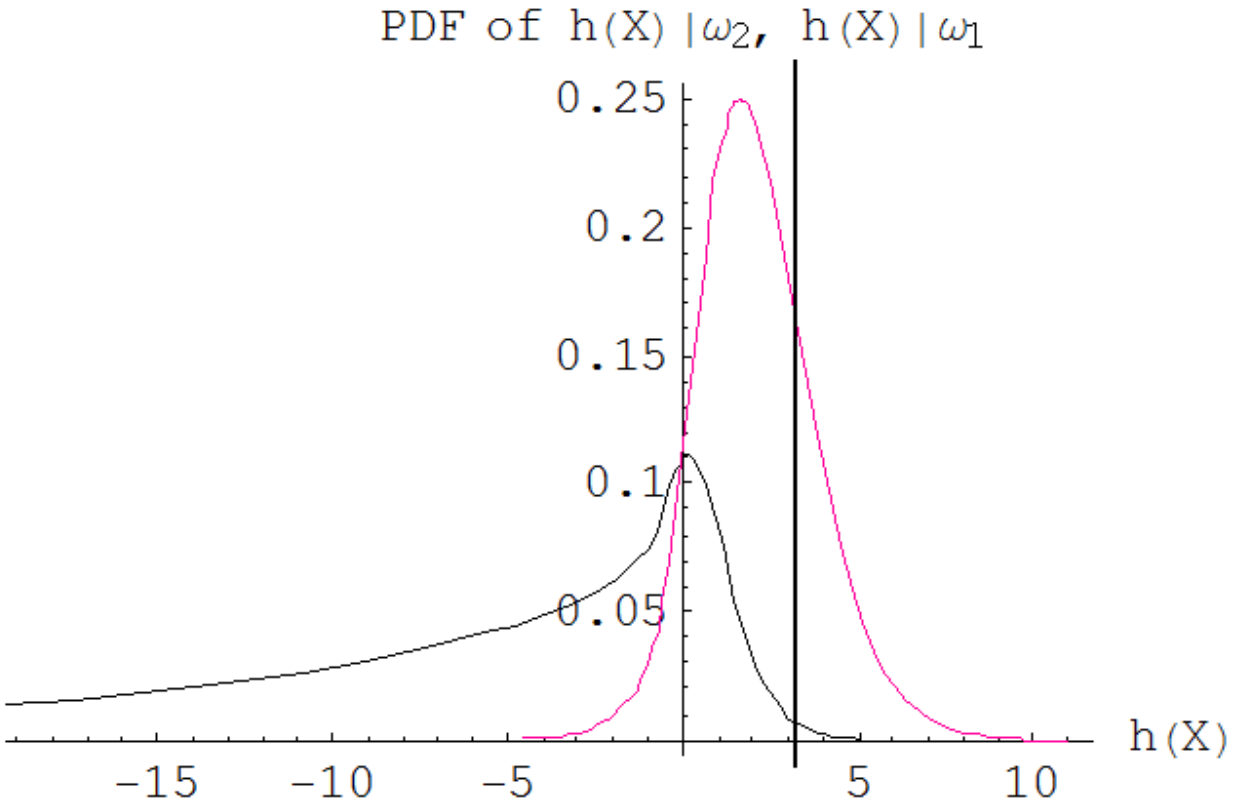
7.2 Assessing Regression Functions

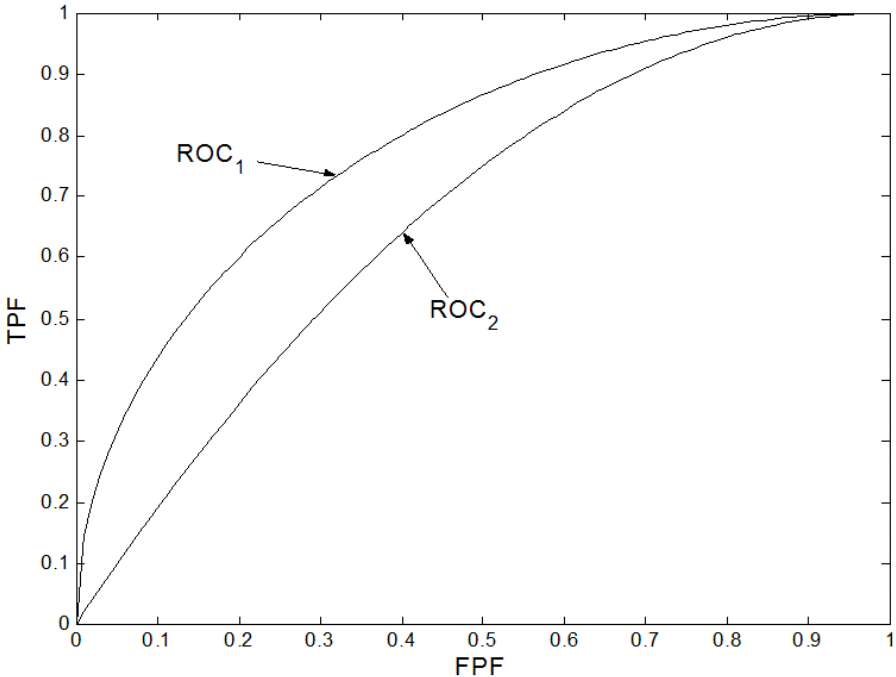
7.3 Assessing Classification Functions

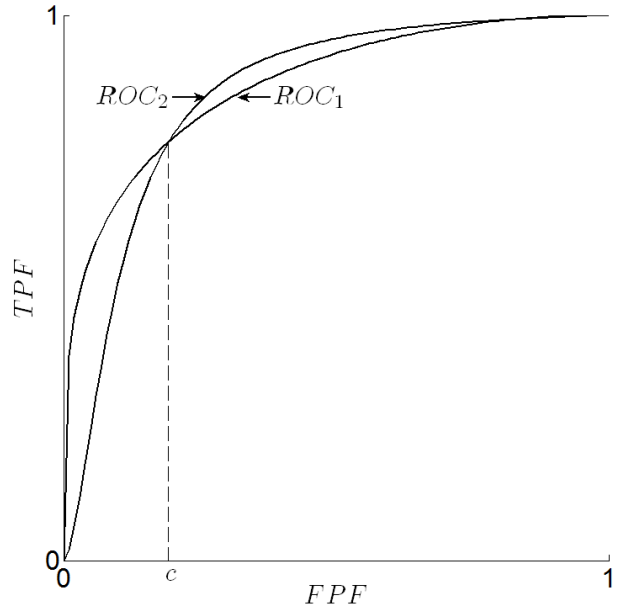
7.3.1 Any Discrimination Rule Can Have an Arbitrarily Bad Probability of Error for Finite Sample Size (Devroye, 1982)

7.3.2 Receiver Operating Characteristics (ROC)




PDF of h(X) | ω2, h(X) | ω1
0.25
0.2
0.15
0.1
0.05
h(X)
-15
-10
-5
5
10


ROC1
ROC2
TPF
FPF
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


1
ROC_2
ROC_1
TPF
0
0
c
1
FPF

7.3.3 Area Under the Curve (AUC)

7.3.4 Important Mesures for Binary Classification Problem

- 1-FNF, $1-e_2$, TPF, TPR, Sensitivity, Recall, Probability of Detection/Positive alaram, Hit Rate, per-class accuracy:

$$\frac{TP}{P} = \frac{TP}{TP+FN}$$

- 1-FPF, $1-e_2$, TNF, TNR, Specificity:

$$\frac{TN}{N}$$

- Positive Predictive Value (PPV), Precision (obtainable as well from the ROC. It is the conditional probability that a positive decision is true.):

$$\frac{TP}{TP+FP} = \frac{1}{1+FP/TP} = \frac{1}{1+FPF/TPF}$$

- Accuracy:

$$\frac{TP+TN}{P+N}$$

- F-score:

$$2\frac{Precision \cdot Recall}{Precision + Recal}$$

- F_β-score:

$$(1+\beta^2)\frac{Precision \cdot Recall}{\beta^2 Precision + Recal}$$

7.4 Resampling and Variability

7.4.1 p-values

7.4.2 Cross Validation (CV)

@@@ about the CV error bars:
Hastie: It is a muccky area but we do it anyway.
Tibs: it is a good approximation.

7.4.2.1 K-Fold CV

7.4.2.2 Model Assessment & Selection Using K-Fold CV

$$\widehat{err}_{\mathbf{tr}}^{(KCV)}\left(\widehat{f},\lambda\right)=\frac{1}{N}\sum_{i}L\left(y_i,\widehat{f}^{-\mathcal{K}(i)}\left(x_i,\lambda\right)\right)$$

```
KCV ALGORITHM for model m{
    Divide the N-observation dataset to...
        K partitions, each has N/K;

    for k = 1 : K
        Train on all data except partition k;
        Test on partition k;
        Save the N/K predictions;
    end

    Collect the K * N/K predictions;

    Estimate your error;
}
```

HW: Prove that estimating the error rate from each fold then averaging over the folds gives the same estimate as if we pool the scores from folds and obtain one estimate from the n pooled scores.

```
KCV ALGORITHM for model selection{
    for m = 1 : M
        Err[m] = error of model m...
             estimated using KCV;
    end

    Find the minimum Err and its model;
}
```

7.4.3 Bootstrap

7.5 Discussion in Hyperspace: Pitfalls and Advices

Lee (2008)
Ambroise and McLachlan (2002)
Simon et al. (2003)

7.5.1 Good Features vs. Good Classifiers

All classifiers perform similarly Lim et al. (2000); focus on selecting good features if you have the physics of the problem.

7.5.2 Pitfall of CV

Golub et al. (1999)

7.5.3 Pitfall of Reusing the Testing set

@@@This section is not rigorous and the idea is not consolidated yet!!
Good Practice: one use of testing (or called validation) set.

$$A = \mathrm{E}_{\mathbf{ts}}\, L(y_{\mathbf{ts}}, \eta_{\mathbf{tr}}) \tag{7.1}$$

$$\eta^* = \underset{i}{\operatorname{argmin}}\ \mathrm{E}_{\mathbf{ts}}\ L(y_{\mathbf{ts}}, \eta_{\mathbf{tr}}^{(i)}) \tag{7.2}$$

Bad Practice: the pitfall of reusing the testing set. E.g., selecting the best model η^*, then reusing the same testing set to select the optimal threshold for classification!
when do you know you have an over-optimistic measure A of your model (or over trained)?
This happens, e.g., when training model**s** on **tr** then testing on **ts** to choose the best model in terms of some loss; e.g., MSE. Then this chosen best model is used on the same test set to choose the best threshold value. In this case, the threshold value is a tuning parameter selected using **ts** and tested on **ts** as well.

"Suppose instead that we use the test-set repeatedly, choosing the model with smallest test-set error. Then the test set error of the final chosen model will underestimate the true test error, sometimes substantially." (Hastie et al., 2009, P. 222)
Incrementally converting the testing set to training set Gur et al. (2004)

7.5.4 One Independent Test set is not Sufficient.

Dave et al. (2004) and Tibshirani (2005)

Chapter 11

Neural Networks

@@@ The connection between Trees and NN:
root-firstlevel-leafs tree, you can get the same performance using input-twolayers-output NN. Deeper tree is similar to wider network (more neurons on the two layers). Deeper networks are much powerful than trees therefore. (https://www.youtube.com/watch?v=2_Jv11VpOF4) imperial college talk.
(what about many trees; i.e., random forests) Proof everything.

β for logistic regression can start with 0, For NN we have to start randomly.

Linear classification vs. logistic regression (complexity) Bishop page 205.

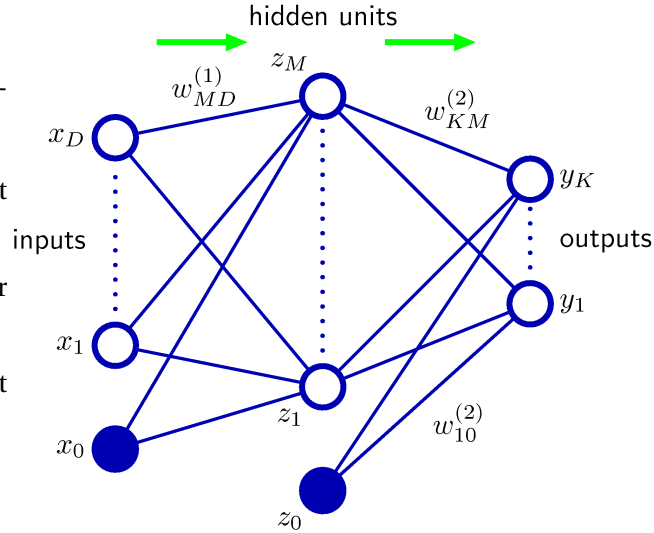
Difference between numerical solution of NN and Logistic regression.

11.1 Basic and Definitions

There is some confusion in layer counting in the literature. This network is

- three layer by counting all layers including input and output.
- two layer by counting all but the input. (our convention)
- single layer by counting all but both the input and output; (only the hidden layers).

Neuron: each element in a layer.
Activation Function of layer i: $\sigma^{(i)}$
Bias element x_0 and z_0 : accounts for intercept.



$$\begin{aligned}
\widehat{Y}_k &= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}\sigma^{(1)}\left(w^{(1)}_{m0} + \sum_{i=1}^{D} w^{(1)}_{mi}X_i\right)\right) \\
&= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}\sigma^{(1)}\left(w^{(1)}_{m0} + \mathbf{W}_m\mathbf{X}\right)\right) \\
&= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}Z_m\right)
\end{aligned}$$

11.2 Connection to Linear Model

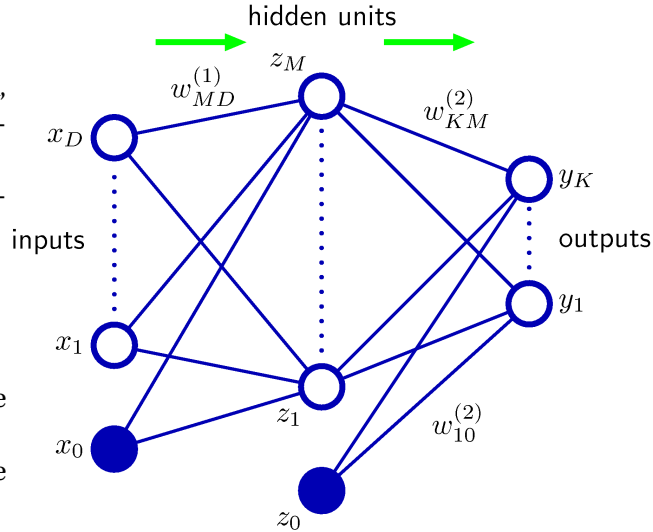
this means: Projection in a linear subspace $\mathbf{W}_{M\times D}\mathbf{X}$, followed by non-linear feature generation (to add flexibility) $Z = \sigma^{(1)}(\mathbf{WX})$.
Usually, for regression, $\sigma^{(2)}$ of the second layer is chosen to be identity $\sigma^{(2)}(T) = T$. Therefore

$$\widehat{Y}_k = w_{0k}^{(2)} + \sum_{m=1}^{M} w_{mk}^{(2)} Z_m,$$

which is nothing but a linear model in the new feature space.
And for binary classification, $\sigma^{(2)}$ is chosen to be the softmax function

$$\sigma^{(2)}(T_k) = \frac{\exp(T_k)}{\sum_k \exp(T_k)},$$

which means that $\widehat{Y}_k = \widehat{\Pr}(\mathcal{C}_k|X) = \sigma^{(2)}(T_k)$. This is exactly a multi-logistic regression problem in the transformed features Z.
If we start with zero weights the algorithm will never move.



$$\begin{aligned}\widehat{Y}_k &= \sigma^{(2)}\left(w_{0k}^{(2)} + \sum_{m=1}^{M} w_{mk}^{(2)}\sigma^{(1)}\left(w_{m0}^{(1)} + \sum_{i=1}^{D} w_{mi}^{(1)} X_i\right)\right)\\ &= \sigma^{(2)}\left(w_{0k}^{(2)} + \sum_{m=1}^{M} w_{mk}^{(2)}\sigma^{(1)}\left(w_{m0}^{(1)} + \mathbf{W}_m\mathbf{X}\right)\right)\\ &= \sigma^{(2)}\left(w_{0k}^{(2)} + \sum_{m=1}^{M} w_{mk}^{(2)} Z_m\right)\end{aligned}$$

Cont.

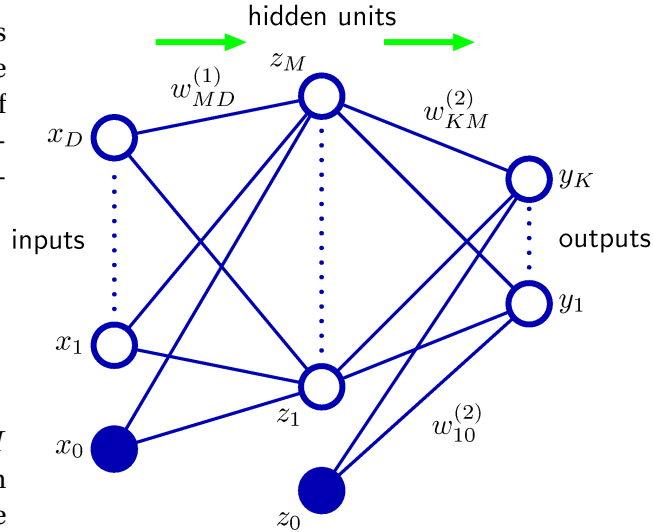
The advantage of transformation from X to Z is adding more predictive power. However, this is on the expense of model interpretability. In linear models of regression or classification, the output is related directly to the input and every coefficient has a meaning.

Usually $\sigma^{(1)}$ is chosen to be sigmoid

$$\sigma(\mu) = \frac{1}{1+\exp(-\mu)}$$

or tan-sigmoid function.

It can be proven that a 2-layer NN with sufficient M can approximate any input function of finite domain with finite discontinuities. This result hold for a wide range of activation functions $\sigma^{(1)}$ excluding polynomials.



$$\begin{aligned}
\widehat{Y}_k &= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}\sigma^{(1)}\left(w^{(1)}_{m0} + \sum_{i=1}^{D} w^{(1)}_{mi}X_i\right)\right) \\
&= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}\sigma^{(1)}\left(w^{(1)}_{m0} + \mathbf{W}_m\mathbf{X}\right)\right) \\
&= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}Z_m\right)
\end{aligned}$$

11.3 Connection to PCA

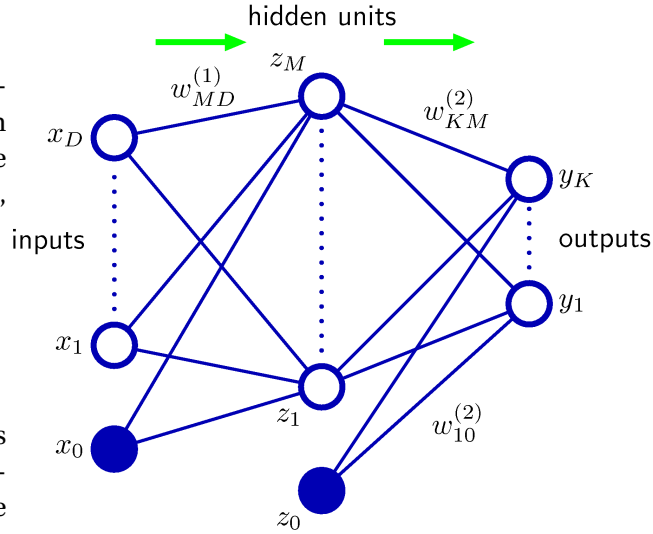
The network in the figure can We can design the network without having a response variable, but with number of outputs K equals to dimensions p. We keep changing the weights and observe the output, then measure the error:

$$E(w) = \sum_{i=1}^{n} \left\| \widehat{y}(x_i, w) - x_i \right\|^2$$
$$= \|\mathbf{WX} - \mathbf{X}\|$$

With one hidden layer with link function it is obvious that the output is linear in the inputs. We can design the network without having a response to achieve minimum error.

In this case the NN will be nothing but PCA, because both use square loss function as minimization criterion. The network will find the largest M principal components. But of course, PCA is better since the solution is found at once using SVD rather than training.

Adding a nonlinear hidden layer will project to the PCA also (there is a proof for that)



$$\widehat{Y}_k = \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}\sigma^{(1)}\left(w^{(1)}_{m0} + \sum_{i=1}^{D} w^{(1)}_{mi} X_i\right)\right)$$
$$= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk}\sigma^{(1)}\left(w^{(1)}_{m0} + \mathbf{W}_m\mathbf{X}\right)\right)$$
$$= \sigma^{(2)}\left(w^{(2)}_{0k} + \sum_{m=1}^{M} w^{(2)}_{mk} Z_m\right)$$

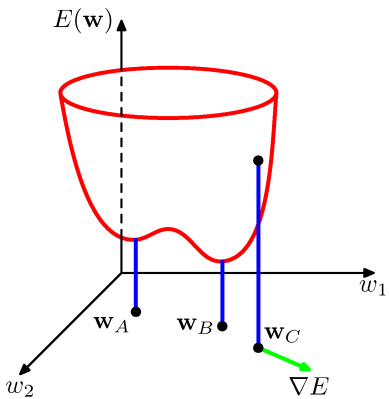
11.4 Solution vs. Optimization

We have to find the weights w such that error

$$E(w) = \sum_{i=1}^{n} \left\| \widehat{y}(x_i, w) - y_i \right\|^2$$

is minimized. Notice that, x_i is the i^{th} realization of the random vector X. So, $x_i = (x_{i1}, \ldots, x_{iD})'$; and:

$$E(w) = \sum_{i=1}^{n} \left\| \sigma^{(2)} \left(w_{0k}^{(2)} + \sum_{m=1}^{M} w_{mk}^{(2)} \sigma^{(1)} (\mathbf{W}_m x_i) \right) - y_i \right\|^2.$$



Problem 1: This function is complicated in ws, and no closed form solution for its minimum. Any 2-layer feed forward network, with M hidden units has $(D+1)M + (M+1)K$ weights.
The simplest network has 1 input, 1 output, and 1 hidden unit. This gives 4 weights! Solution has to be numerical.

$$E(w) = \sum_{i=1}^{n} \left\| \sigma^{(2)} \left(w_{0k}^{(2)} + \sum_{m=1}^{M} w_{mk}^{(2)} \sigma^{(1)} (\mathbf{W}_m x_i) \right) - y_i \right\|^2$$
$$= \sum_{i=1}^{n} \left\| \underbrace{w_4 + w_3 \sigma (w_1 + w_2 x_i)}_{\widehat{y}_i} - y_i \right\|^2$$

Problem 2: It is non-convex function, this means local minima exist; however, we seek the global one. (how do we know?)

11.5 NN for Regression

Consider training a network on a 100-observation sample from $Y = \sin(2X) + \varepsilon$, where $X \sim \mathcal{N}(0,1)$ and $\varepsilon \sim \mathcal{N}(0,\sigma^2)$ so that the signal to noise ratio (SNR) is about 4. Remember

$$SNR = \frac{\text{Var}(Y|X)}{\text{Var}(\varepsilon)}.$$

In this case, it will be achieved for $\sigma = 0.35$.

Testing set (never seen by NN, this is just for understanding what's going on) is 10,000.

Let's train each network until it reaches the maximum accuracy for the gradient (or 10^{-5} is enough).

Try several M (from 1 to 5); for each do the training 10 times with different initialization vector (see the code).

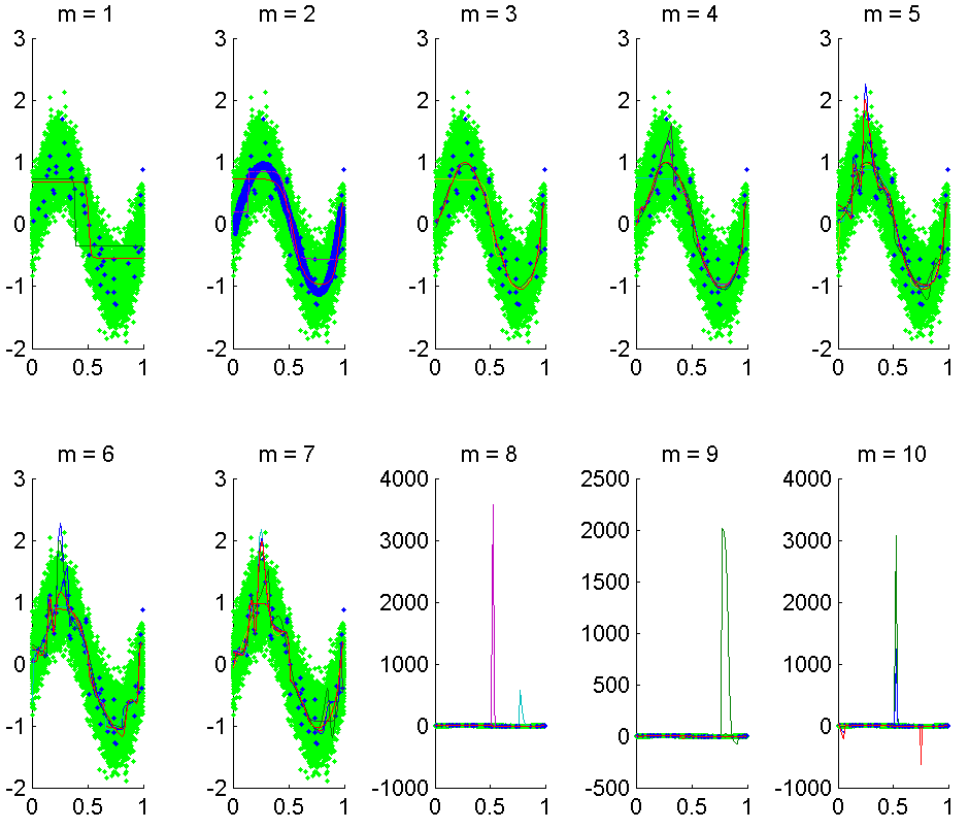
Matlab Code 11.1:

```
trainsize=100; testsize=10000; sig=0.3;
x=rand(trainsize+testsize, 1)'; fx=sin(2*pi*x);
y=fx+ sig*mvnrnd(0, 1, trainsize+testsize)'; %4 to 1 SNR; SNR = var(E(Y|X))/var(eps).
xtrain=x(1:trainsize); ytrain=y(1:trainsize);
xtest=x(trainsize+1:end); ytest=y(trainsize+1:end);

%% This to fix the initialization over the simulation below.
M=10; I=10;
InitWeightMat=cell(I, M);
for m=1:M for i=1:I net = newfit(xtrain, ytrain, m); InitWeightMat{i, m}={net.IW, net.LW, net.b
    }; end; end;
%% This cell trains M nets, each for I initializations and test each solution on the same
    training set and infinite testing set
ApparPerfMat=zeros(I, M); PerfMat=zeros(I, M); WeightMat=cell(I, M);
for m=1:M
    net = newfit(xtrain, ytrain, m);
    net.divideFcn=''; net.trainParam.showWindow=0;
    net.trainParam.min_grad = 1e-2;% saving some time and prohibits reaching global minima.
    for i=1:I
        tmp=InitWeightMat{i, m};    net.IW=tmp{1, 1}; net.LW=tmp{1, 2}; net.b=tmp{1, 3};
        [net,tr,Y,E,Pf,Af]=train(net, xtrain, ytrain);
        ApparPerfMat(i, m)=sum(E.^2)/length(xtrain);
        [Y2,Xf,Af,E2,perf] = sim(net, xtest, [], [], ytest);
        PerfMat(i, m)=perf;
        WeightMat{i, m}={net.IW, net.LW, net.b};
    end;
end;

%%  With parameter regularization
Lamdalist=0:.01:0.1;    net = newfit(xtrain, ytrain, M);    net.divideFcn='';
ApparPerfMat=zeros(I, length(Lamdalist)); PerfMat=zeros(I, length(Lamdalist)); WeightMat=cell(I
    , length(Lamdalist));
net.performFcn='msereg';   net.trainParam.showWindow=0;
```

```
for m=1:length(Lamdalist)%m here is the regularization parameter.
    net.performParam.ratio=Lamdalist(m);
    for i=1:I
        tmp=InitWeightMat{i, M};
        net.IW=tmp{1, 1}; net.LW=tmp{1, 2}; net.b=tmp{1, 3};
        [net,tr,Y,E,Pf,Af]=train(net, xtrain, ytrain);
        ApparPerfMat(i, m)=sum(E.^2)/length(xtrain);
        [Y2,Xf,Af,E2,perf] = sim(net, xtest, [], [], ytest);
        PerfMat(i, m)=sum(E2.^2)/length(xtest);
        WeightMat{i, m}={net.IW, net.LW, net.b};
    end;
end;
```

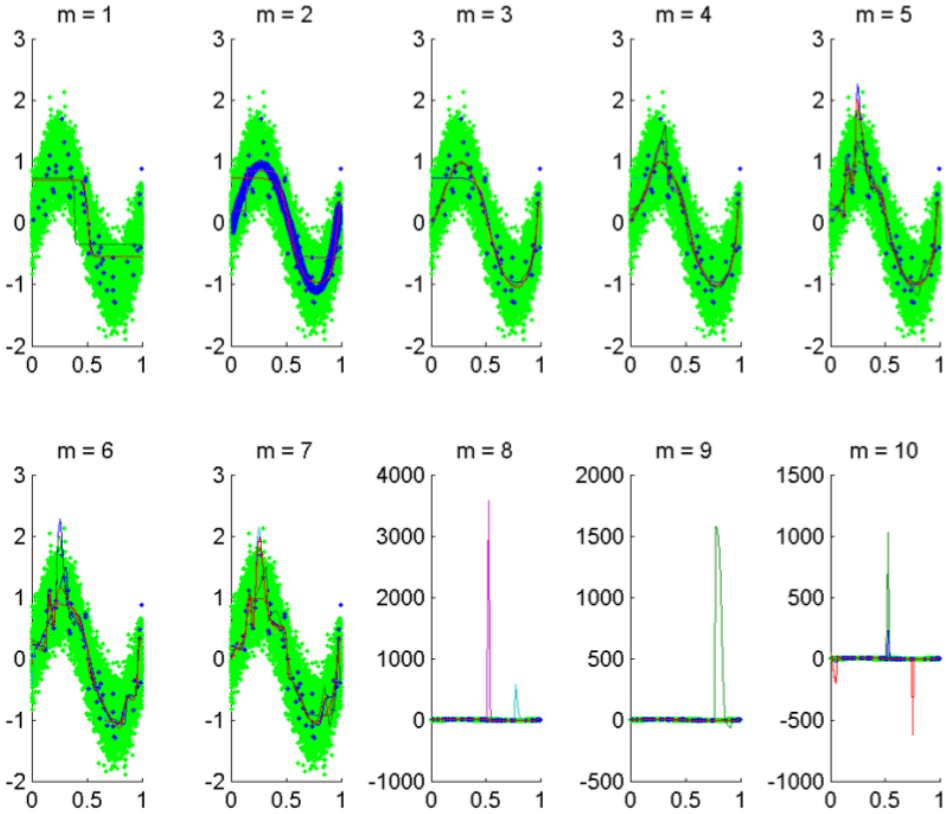
Same 10 initializations per m:

- Small m implies few minima
- Max. Gradient is set to 10^{-10}; this allows for very large unrealistic weight vector searching for an asymptotic minima.
- Best solution is roughly at $(D+1)m+(m+1)K = n/10$. That is, $3m+1 = 10$, i.e., $m = 3$.



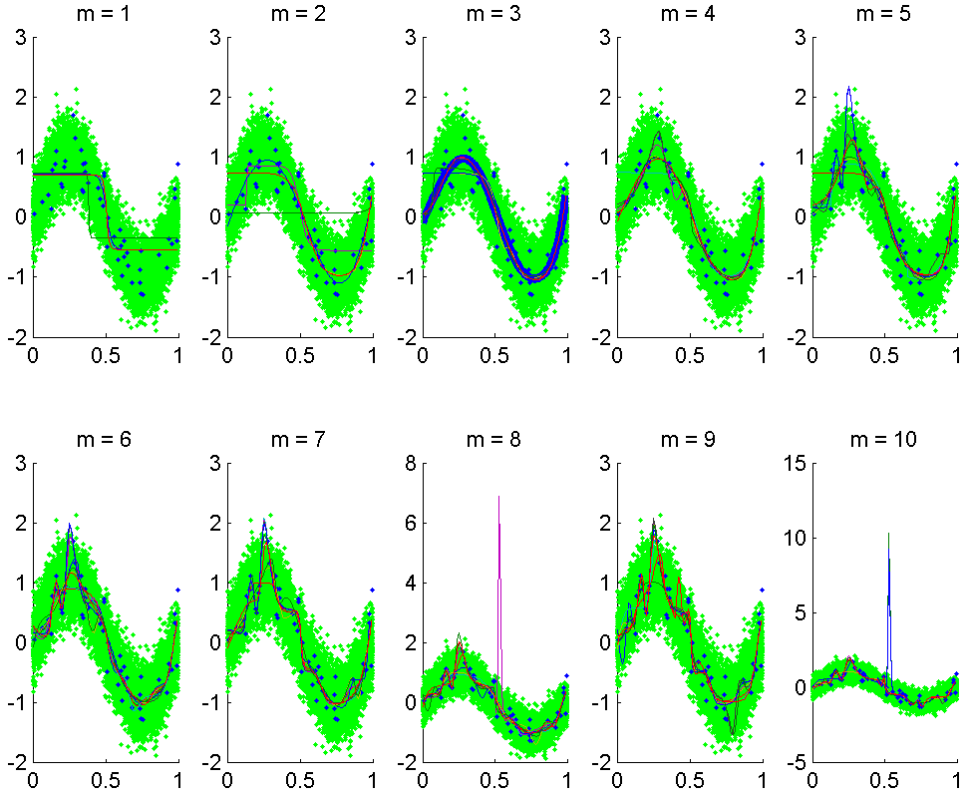
Same 10 initializations per m:

- Small m implies few minima
- Max. Gradient is set to 10^{-4}, which we consider as 0. This sacrifices a little decrease in MSE on the training set for getting reasonably small weight vectors.
- Best solution is roughly at $(D+1)m+(m+1)K = n/10$. That is, $3m+1 = 10$, i.e., $m = 3$.



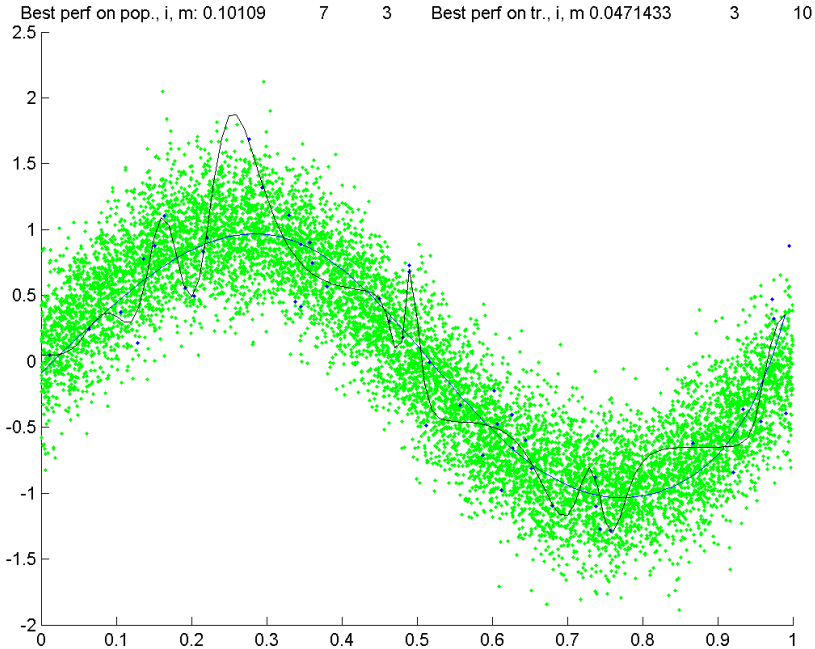
Same 10 initializations per m:

- Small m implies few minima
- Max. Gradient is set to 10^{-3}; Same comment as above.
- Best solution is roughly at $(D+1)m+(m+1)K = n/10$. That is, $3m+1 = 10$, i.e., $m = 3$.



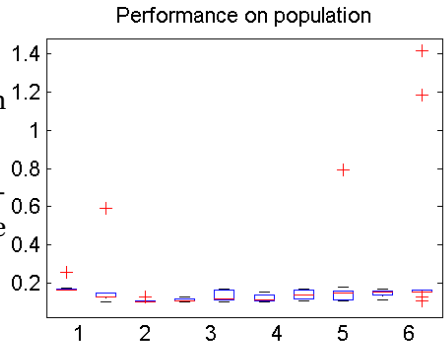
Best solution on population and best on training set for min. grad. $10^{-}3$**:**

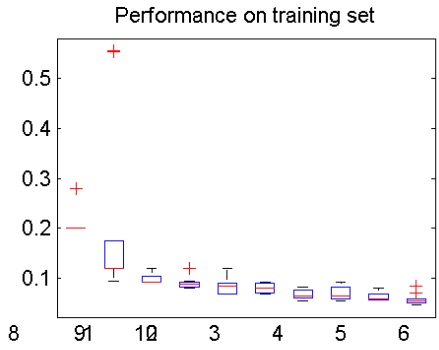
- @@@ i think this was for min-grad 10^{-2} NOT 10^{-3}. Therefore, the curve is not one of those of the previous subplot where the best had a spiky overshoot. I have to rerun the Matlab code.

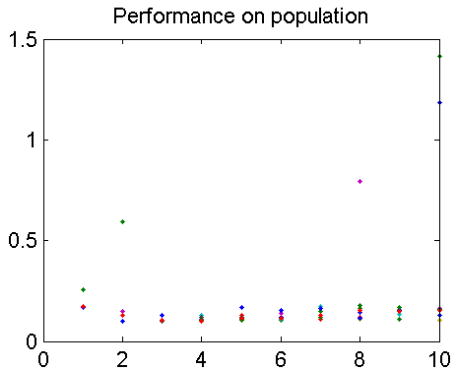


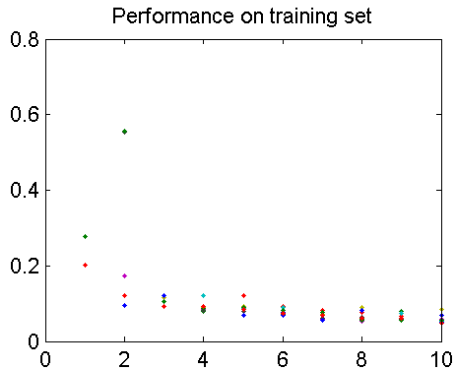
Performance of all solutions for min. grad. 10^{-3}:

- Monotonic decrease in training MSE.
- There is an optimal solution for minimizing the MSE on the population.

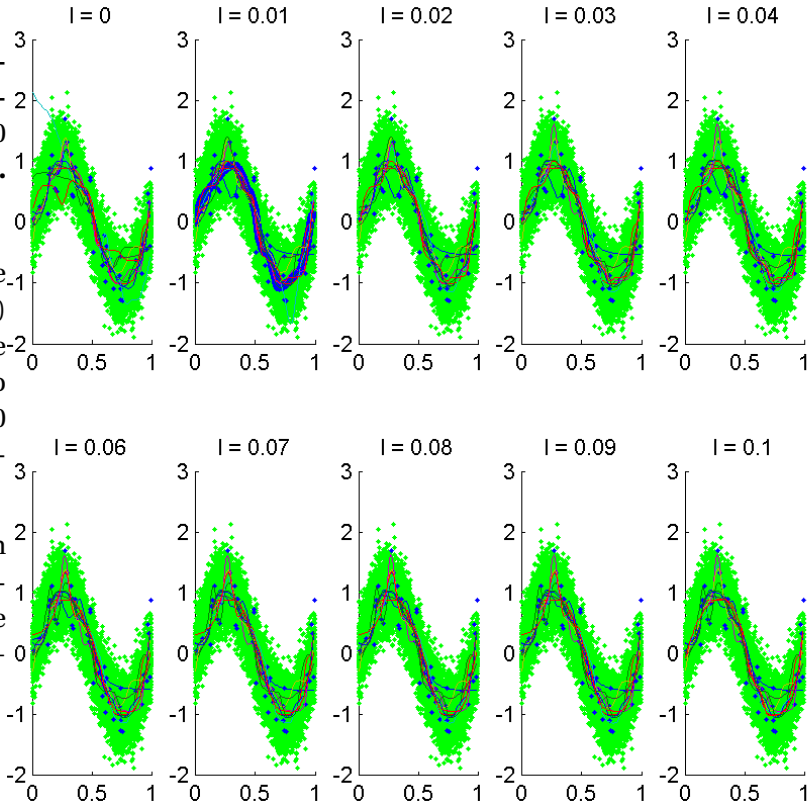






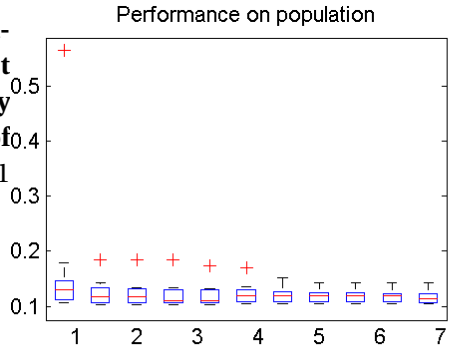


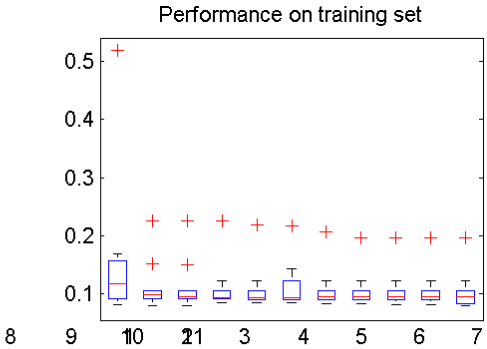
Regularization λ instead of varying m:
Only one net with 10 neurons, with same 10 initializations of the case $m = 10$ previously for min. grad. 10^{-10}.

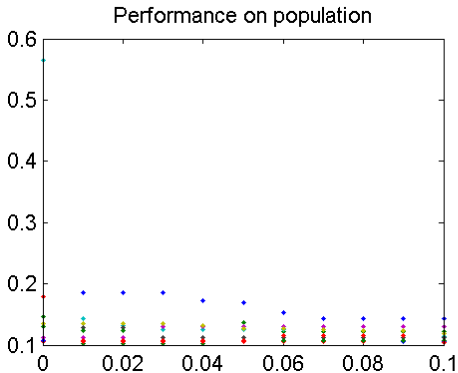


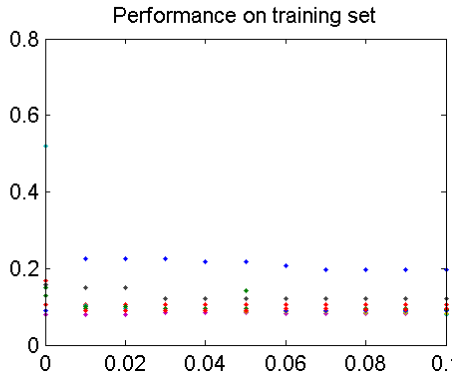
- λ ranges from 0 (the most constrained model) to 0.1. Of course, for case of $\lambda = 1$ this reduces to the full model $M = 10$ without using any regularization.
- training stops when MSEREG starts increasing, where $MSEREG = \lambda\ MSE + (1-\lambda)\sum_{allweights} w^2$

Performance of the all solutions using regularization. Notice that this best solution would be exactly as those for the case of $m = 10$ had we set $\lambda = 1$ (no regularization).

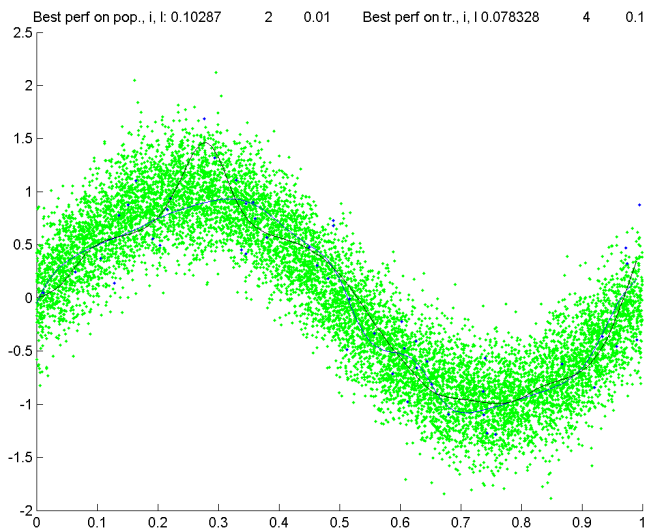




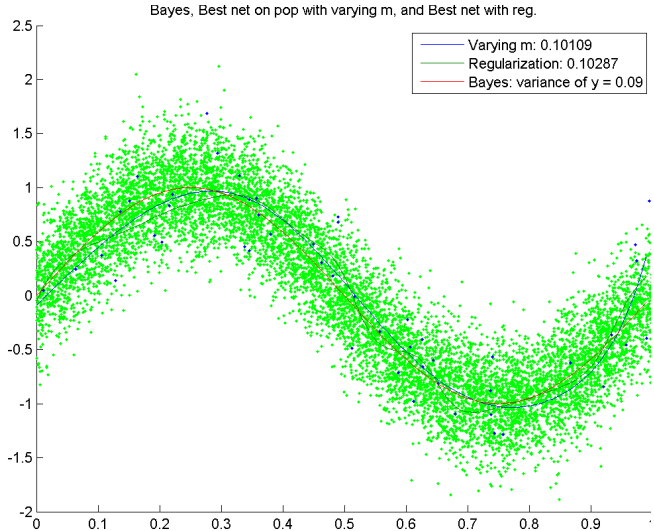




Best solution on the population and best on the training set using regularization.



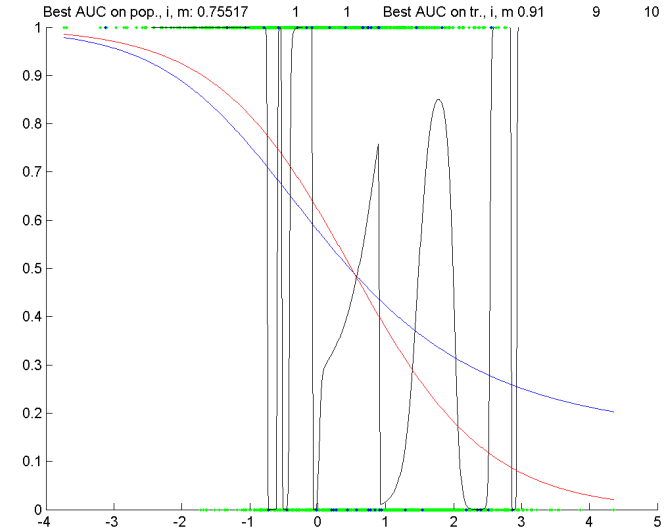
Comparison between: varying m, regularization, and Bayes.

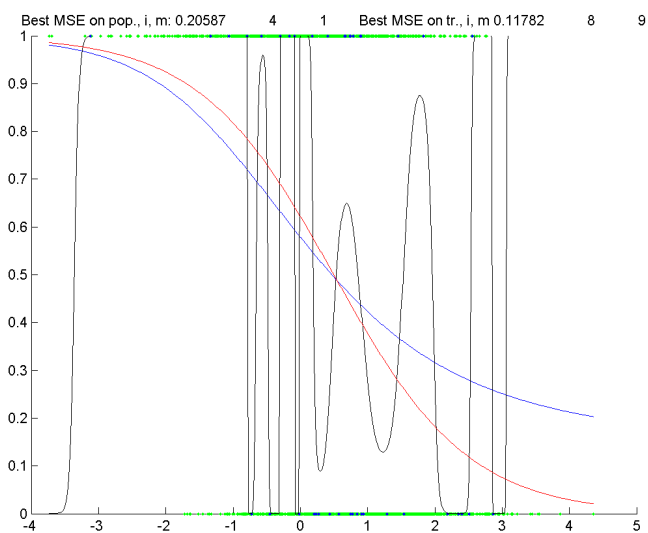


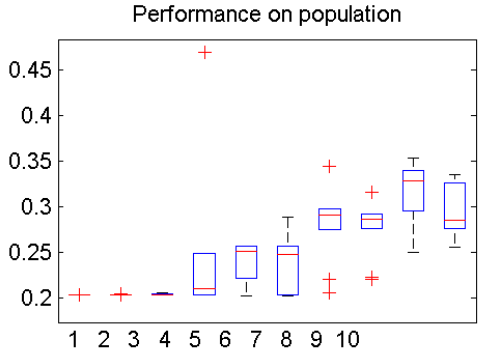
11.6 NN for Classification

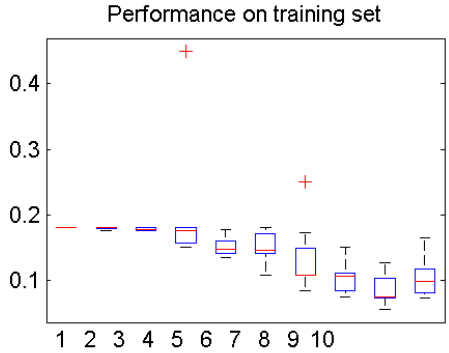
$\mathcal{C}_1 \sim \mathcal{N}(1,1)$
$\mathcal{C}_0 \sim \mathcal{N}(0,1)$
Training: 10 observations/class (blue)
Testing: 1000 observations/class (green).
complicated network tries to estimate $\Pr(\mathcal{C}_1|x)$ closely to 1 or zero. Decision surface is complicated disconnected regions.
Sometimes the best solution is for the net with, say, $m = 4$; however its MSE is very close to the one with $m = 1$.
Red curve is the Bayes' posterior (calculated below)
small network looks like simple logistic regression, decision surface at each threshold is simply a point as it should be.
The true AUC for this problem is 0.76.

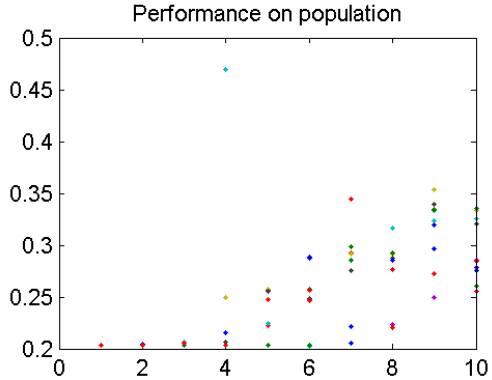
$$f(C_1|x) = \frac{f(x|C_1)\Pr(C_1)}{f(x)} = \frac{f(x|C_1)\Pr(C_1)}{f(x|C_1)\Pr(C_1) + f(x|C_0)\Pr(C_0)}$$
$$= \frac{1}{1 + \frac{1}{LR}\frac{\Pr(C_0)}{\Pr(C_1)}} = \frac{1}{1 + \frac{\sqrt{2\pi}\sigma_0}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2}\left[\left(\frac{x-\mu_0}{\sigma_1}\right)^2 - \left(\frac{x-\mu_1}{\sigma_2}\right)^2\right]}}$$
$$= \frac{1}{1 + e^{x - \frac{1}{2}}}$$

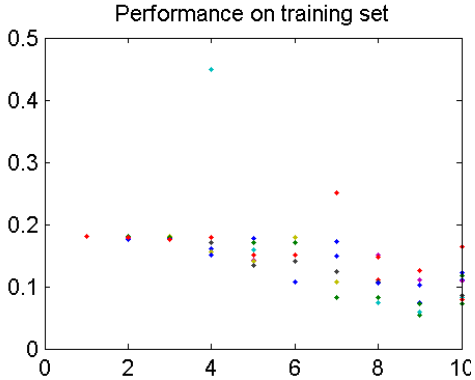








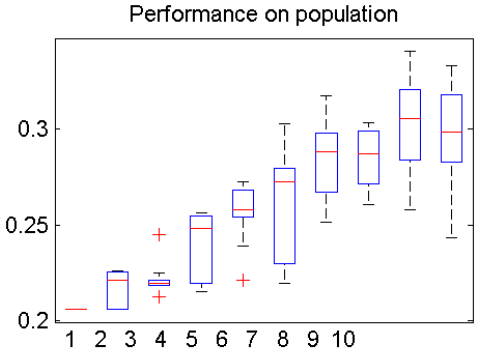


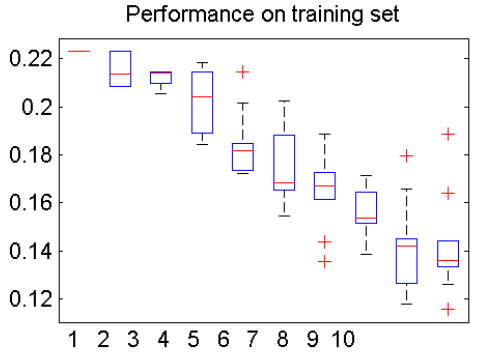


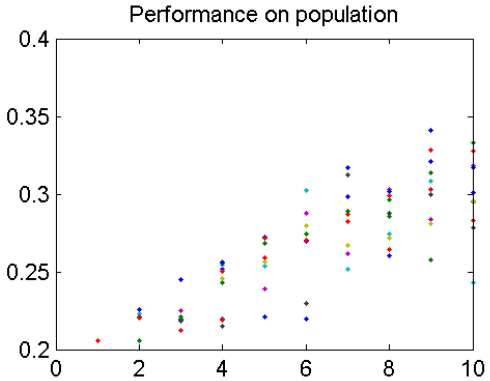
Performance (MSE):

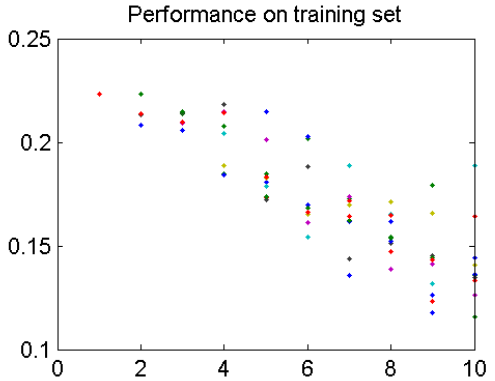
MSE (measured on population) increases with m.

MSE (measured on training set) decreases with m.





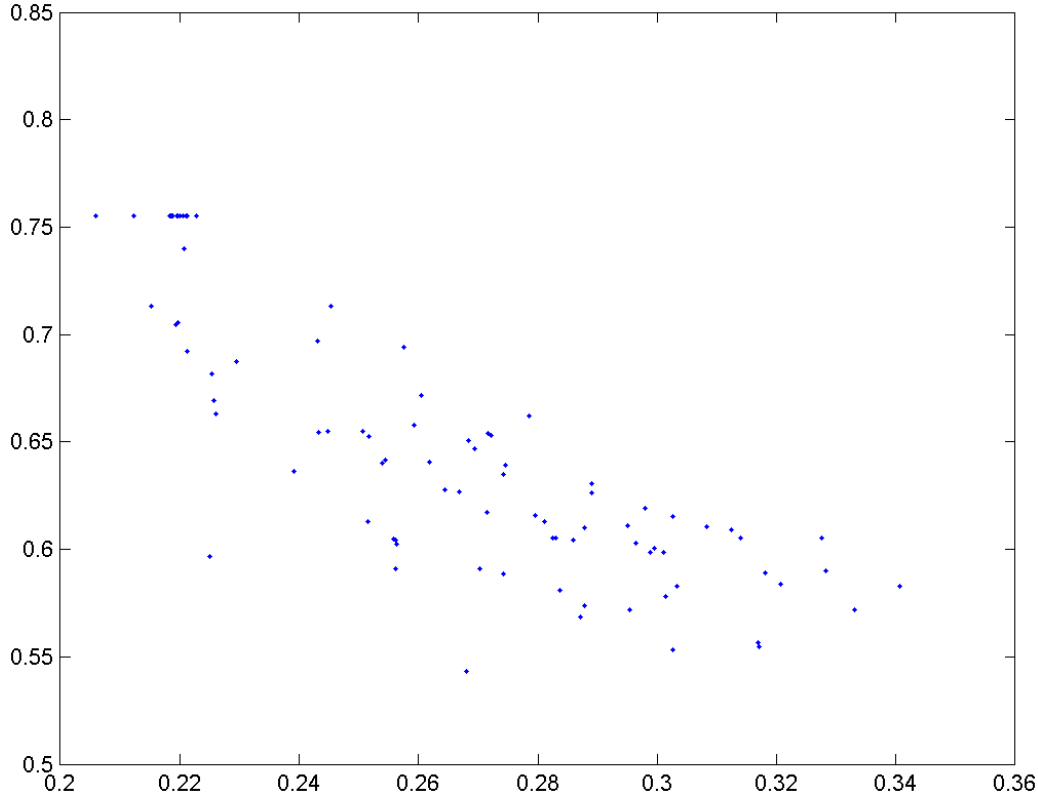




Performance (AUC):

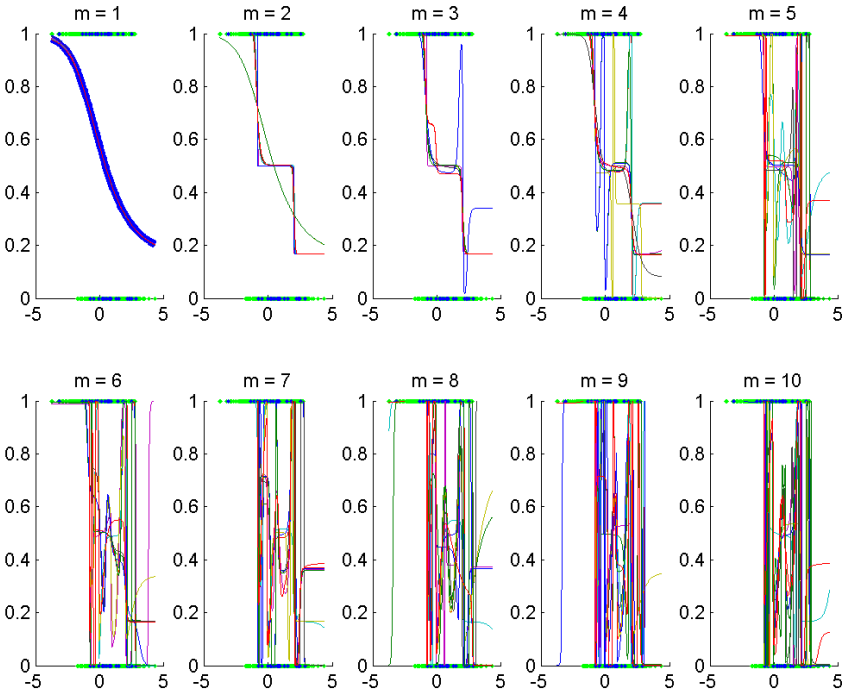
AUC (measured on population) decreases with m.

AUC (measured on training set) increases with m.



It is not necessarily that the solution with best MSE is the same as the solution with the best AUC.

However, **sometimes** there is a general trend.



All solutions ($1 \leq m \leq 10$, 10 different initializations per each m)

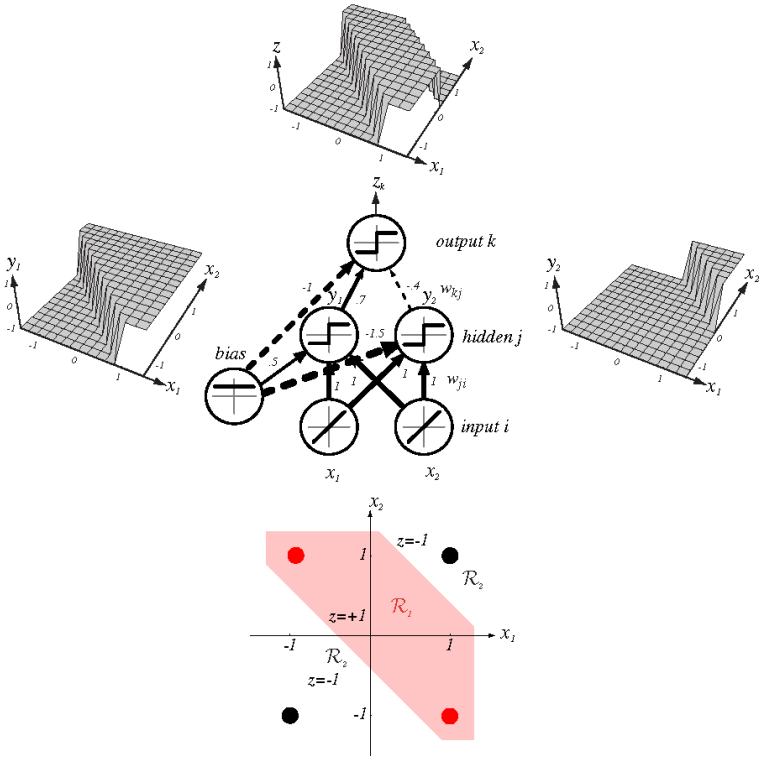
Practical Issues:

Matlab Code 11.2:

```
%%%%%% Important Bug %%%%%%%
% Seting the output transfer function in newpr does not work: and example is:
load simpleclass_dataset
net = newpr(simpleclassInputs,simpleclassTargets,2, {'tansig', 'softmax'});
net.layers{2}.transferFcn
%will give 'tansig', which is the default; not 'softmax' as initialized.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

testsamplesno=1000; n1=10; n2=10;
p=1; MDIST=2;
mu1=zeros(1, p);       mu2=sqrt( (MDIST/p) * ones(1, p) );
sigma1= eye(p);        sigma2=eye(p);
% sigma1= eye(p);         roh=.5; sigma2=repmat(roh, [p p])+(1-roh)*eye(p);

infinitetestsamples1=mvnrnd(mu1, sigma1, testsamplesno)';    infinitetestsamples2=mvnrnd(mu2,
    sigma2, testsamplesno)';
trainers1=mvnrnd(mu1, sigma1, n1)';
trainers2=mvnrnd(mu2, sigma2, n2)';

xtrain=[trainers1 trainers2]; ytrain=[ones(1, n1), zeros(1, n2); zeros(1, n1), ones(1, n2)];
xtest=[infinitetestsamples1 infinitetestsamples2]; ytest=[ones(1, testsamplesno), zeros(1,
    testsamplesno); zeros(1, testsamplesno), ones(1,
testsamplesno)];
%% This to fix the initialization over the simulation below.
M=10; I=10;
InitWeightMat=cell(I, M);
for m=1:M for i=1:I net = newfit(xtrain, ytrain, m); InitWeightMat{i, m}={net.IW, net.LW, net.b
    }; end; end;
%% This cell trains M nets, each for I initializations and test each solution on the same
    training set and infinite testing set
ApparPerfMat=zeros(I, M); PerfMat=zeros(I, M); ApparAUCMat=zeros(I, m); AUCMat=zeros(I, M);
```

```
    WeightMat=cell(I, M);
for m=1:M
    net = newfit(xtrain, ytrain, m);
    net.divideFcn=''; net.trainParam.showWindow=0;
    net.layers{2}.transferFcn='tansig';
    net.trainParam.min_grad = 1e-4;% saving some time and prohibits reaching global minima.
    for i=1:I
        tmp=InitWeightMat{i, m};     net.IW=tmp{1, 1}; net.LW=tmp{1, 2}; net.b=tmp{1, 3};
        [net,tr,Y,E,Pf,Af]=train(net, xtrain, ytrain);
        ApparPerfMat(i, m)=sum(E(:).^2)/(2*(n1+n2));
        ApparAUCMat(i, m)=MannWhitney(Y(1, 1:n1), Y(1, n1+1:n1+n2), -inf);
        [Y2,Xf,Af,E2,perf] = sim(net, xtest, [], [], ytest);
        PerfMat(i, m)=perf;
        AUCMat(i, m)=MannWhitney(Y2(1, 1:testsamplesno), Y2(1, testsamplesno+1:2*testsamplesno)
            , -inf);
        WeightMat{i, m}={net.IW, net.LW, net.b};
    end;
end;
```

with `softmax`, set `net.outputs2.processFcns=` as long as the classes are coded by 0 and 1.

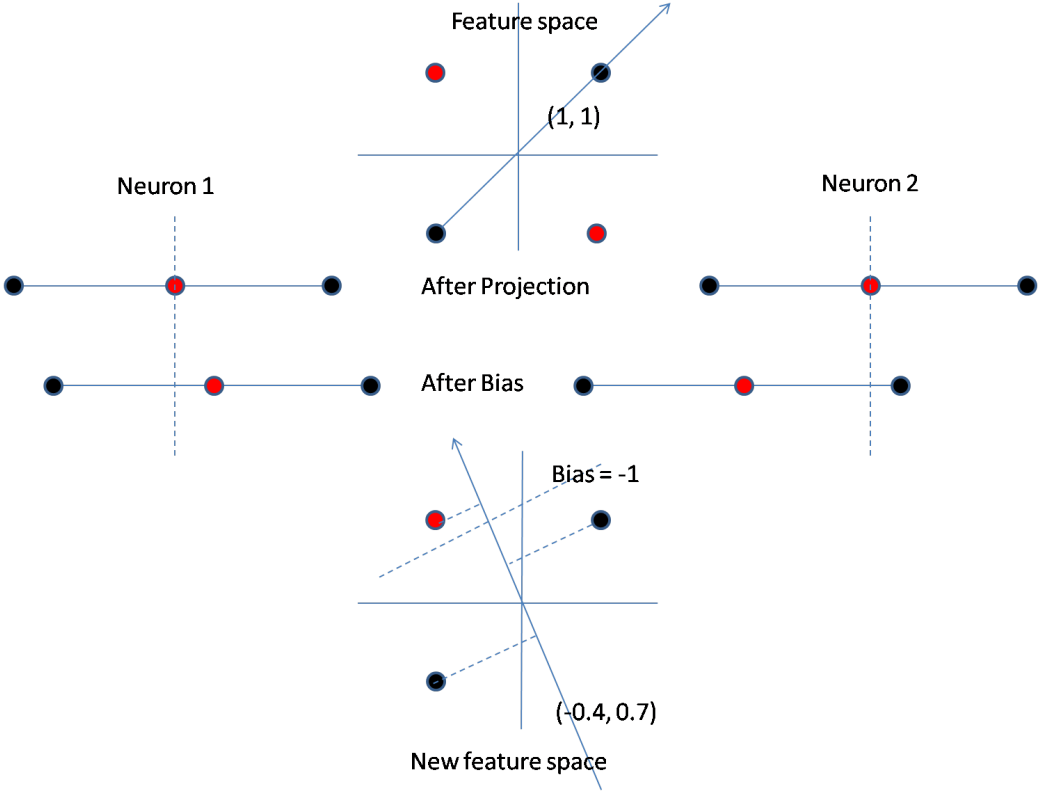


The XOR classification:

$$
\begin{aligned}
y_1 &= \sigma^{(1)}\left(0.5 + 1 \cdot x_1 + 1 \cdot x_2\right), \\
y_2 &= \sigma^{(1)}\left(-1.5 + 1 \cdot x_1 + 1 \cdot x_2\right), \\
z &= \sigma^{(2)}\left(-1 + 0.7 \cdot y_1 - 0.4 \cdot y_2\right),
\end{aligned}
$$

where
the activation functions in the two layers are the hard limits given by:

$$
\sigma^{(1)}\left(\mu\right) = \sigma^{(2)}\left(\mu\right) = \begin{cases} 1 & \mu > 0 \\ 0 & \mu = 0 \\ -1 & \mu < 0 \end{cases}
$$


Feature space
(1, 1)
Neuron 1
Neuron 2
After Projection
After Bias
Bias = -1
(-0.4, 0.7)
New feature space

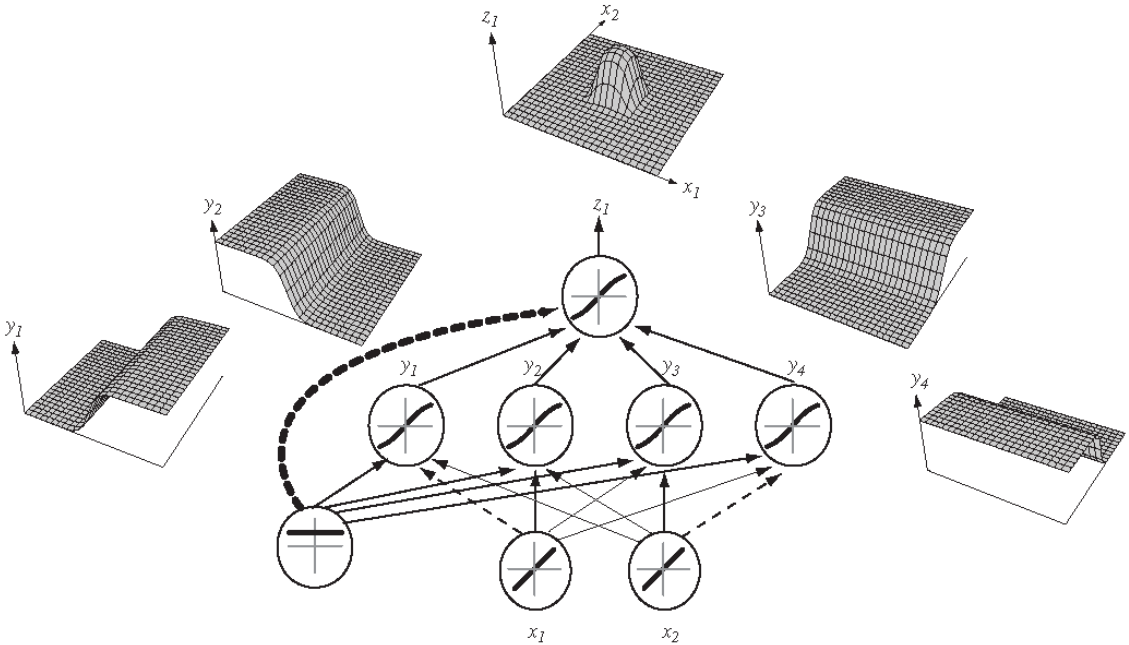


FIGURE 6.2. A 2-4-1 network (with bias) along with the response functions at different units; each hidden output unit has sigmoidal activation function $f(\cdot)$. In the case shown, the hidden unit outputs are paired in opposition thereby producing a "bump" at the output unit. Given a sufficiently large number of hidden units, any continuous function from input to output can be approximated arbitrarily well by such a network. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Chapter 14

Unsupervised Learning

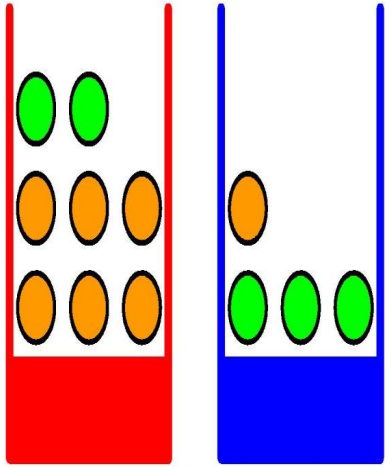
Appendix A

Fast Revision on Probability

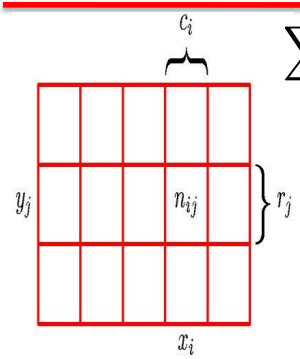
This introduction is adapted from Bishop's slides

Probability Theory

Apples and Oranges



Probability Theory



$$\sum_i \sum_j p\ x_i, y_j\ = 1$$

Marginal Probability

$$p(X = x_i) = \frac{c_i}{N}.$$

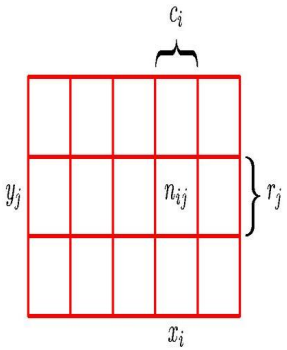
Joint Probability

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N}$$

Conditional Probability

$$p(Y = y_j | X = x_i) = \frac{n_{ij}}{c_i}$$

Probability Theory



Sum Rule

$$p(X = x_i) = \frac{c_i}{N} = \frac{1}{N}\sum_{j=1}^{L} n_{ij}$$

$$= \sum_{j=1}^{L} p(X = x_i, Y = y_j)$$

Product Rule

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N} = \frac{n_{ij}}{c_i} \cdot \frac{c_i}{N}$$

$$= p(Y = y_j | X = x_i) p(X = x_i)$$

The Rules of Probability

Sum Rule

$$p(X) = \sum_{Y} p(X, Y)$$

Product Rule

$$p(X, Y) = p(Y|X)p(X)$$

Independence

$$p\ X,Y\ = p\ X\ p\ Y$$

$$p\ Y \mid X\ = p\ Y$$

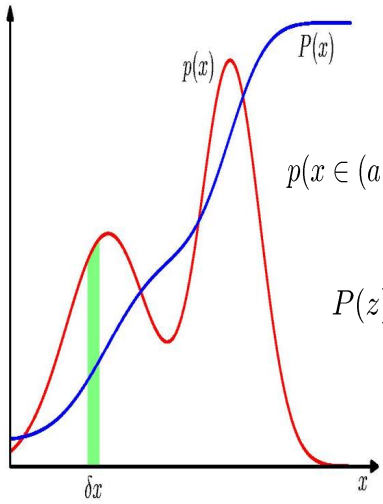
Bayes' Theorem

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

$$p(X) = \sum_{Y} p(X|Y)p(Y)$$

posterior $\propto$ likelihood × prior

Probability Densities



$$p(x \in (a,b)) = \int_a^b p(x)\,\mathrm{d}x$$

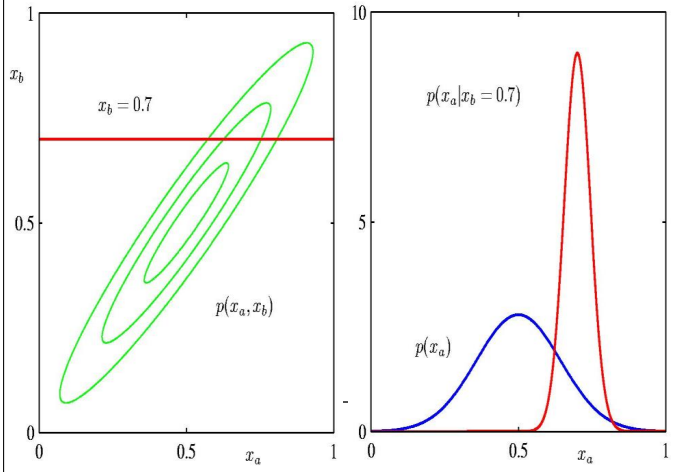
$$P(z) = \int_{-\infty}^{z} p(x)\,\mathrm{d}x$$

$$p(x) \geqslant 0 \qquad \int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1$$

Continuous variables

$$f\ \ Y\ \ = \int f\ \ X,Y\ \ dX$$

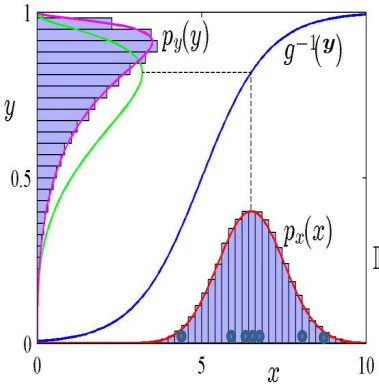
$$f\ \ X \mid Y = y\ \ = \frac{f\ \ X,Y = y}{f\ \ Y = y}$$



Transformed Densities and Expectation

$$p_y(y) = p_x(x)\left|\frac{dx}{dy}\right| = p_x(g(y))\,|g'(y)|$$

$$\mu_X = E[X] = \sum p\ x\ xdx \qquad \int p\ x\ xdx$$



$$\mathbb{E}[f] = \int p(x) f(x)\,\mathrm{d}x$$

$$\mathbb{E}[f] = \sum_x p(x) f(x)$$

Conditional Expectation (discrete)

$$\mathbb{E}_x[f|y] = \sum_x p(x|y) f(x)$$

Approximate Expectation

$$\mathbb{E}[f] \simeq \frac{1}{N}\sum_{n=1}^{N} f(x_n)$$

$$\sigma^2{}_X = \mathrm{var}[X] = E\ \ X - \mu_X\ \ ^2 = E\ \ X^2\ \ - \mu^2{}_X$$

$$\mathrm{var}[f] = \mathbb{E}\left[(f(x) - \mathbb{E}[f(x)])^2\right] = \mathbb{E}[f(x)^2] - \mathbb{E}[f(x)]^2$$

$$E\left[Y \mid X = x_o\right] = \mu_{Y|X=x_o}$$

$$\text{var}\left[Y \mid X = x_o\right] = \sigma^2_{Y|X=x_o} = E\left[\ Y - \mu_{Y|X=x_o}\ ^2 \mid X = x_o\right]$$

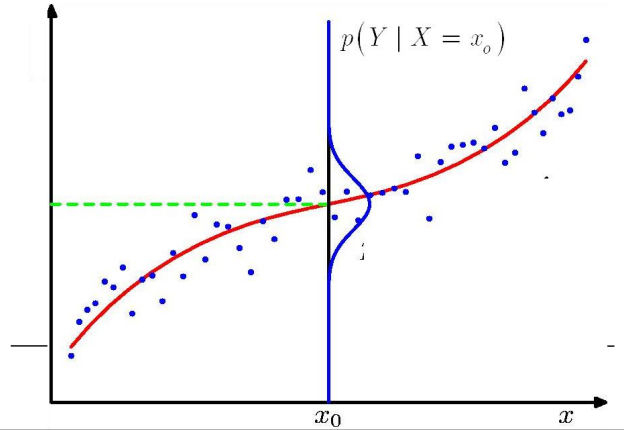
$$\text{var}[Y] = E_X \text{var}\left[Y \mid X\right] + \text{var}_X E\left[Y \mid X\right]$$

$$\begin{aligned}\text{cov}[x, y] &= \mathbb{E}_{x,y}\left[\{x - \mathbb{E}[x]\}\{y - \mathbb{E}[y]\}\right] \\ &= \mathbb{E}_{x,y}[xy] - \mathbb{E}[x]\mathbb{E}[y]\end{aligned}$$

$$\rho = \frac{\text{cov}\ \ X, Y}{\sigma_X \sigma_Y}$$

$$-1 \leq \rho \leq 1$$

Observed correlation does NOT imply causation



Prove the follwing

$$\mathrm{var}\left[X+Y\right] = \mathrm{var}\,X + \mathrm{var}\,Y + 2\,\mathrm{cov}\;\;X,Y$$

$$\mathrm{cov}\;\;X,Y\;\; = 0 \;\; if \; X \perp Y$$

$$\mathrm{var}\left[aX\right] = a^2\,\mathrm{var}\left[X\right]$$

Intuition:

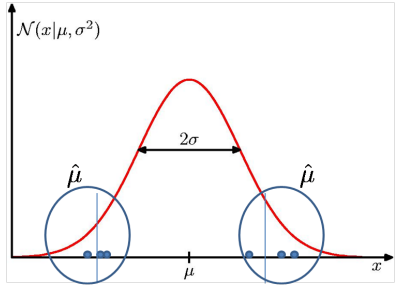
the variance of each affects the total variance, but if one decreases with the increase of the other this should affect the total variance

If independent, of course there is not association.

Scaling a r.v. increases or decreases the spread depending on whether the scale “a” is larger or smaller than one.

Appendix B

Fast Revision on Statistics



A random variable (or vector) is denoted by upper case letters, e.g., X.

Independent observations (realizations) of this r.v. are called independent and identically distributed (i.i.d.), e.g., $x_1, \ldots, x_n$.

Estimator is a real-valued function that tries to be "close" in some sense to a population quantity.
How "close"? Define a loss function, e.g., the Mean Square Error (MSE): $L\left(\widehat{\mu},\mu\right) = \left(\widehat{\mu}-\mu\right)^2$ .And, define the Risk to be the Expected loss: $\mathrm{E}\left(\widehat{\mu}-\mu\right)^2$.

Important Decomposition for any estimator $\widehat{\mu}$:

$$\begin{aligned}
\mathrm{E}\left(\widehat{\mu}-\mu\right)^2 &= \mathrm{E}\left(\left(\widehat{\mu}-\mathrm{E}\,\widehat{\mu}\right)+\left(\mathrm{E}\,\widehat{\mu}-\mu\right)\right)^2 \\
&= \mathrm{E}\left(\widehat{\mu}-\mathrm{E}\,\widehat{\mu}\right)^2 + \mathrm{E}\left(\mathrm{E}\,\widehat{\mu}-\mu\right)^2 + 2\,\mathrm{E}\left[\left(\widehat{\mu}-\mathrm{E}\,\widehat{\mu}\right)\left(\mathrm{E}\,\widehat{\mu}-\mu\right)\right] \\
&= \mathrm{Var}\,\widehat{\mu} + Bias^2\left(\widehat{\mu}\right)
\end{aligned}$$

We could have defined other loss functions, e.g., the absolute deviance loss:

$$L\left(\widehat{\mu},\mu\right)=|\widehat{\mu}-\mu|$$

One estimator may be better for one loss and not better for another loss.

Estimation of μ_X

Sample mean $\overline{X}$ as an estimator of μ_X : $\widehat{\mu}_X = \frac{1}{n}\sum_{i=1}^{n} x_i$.

$$\begin{aligned}
\mathrm{E}\,\overline{X} &= \mathrm{E}\,\frac{1}{n}\sum_{i=1}^{n} x_i = \mathrm{E}\,X \;\; (=\mu) \\
Bias\left(\widehat{\mu}\right) &= \mathrm{E}\,\widehat{\mu} - \mu = 0 \\
\mathrm{Var}\,\widehat{\mu} &= \frac{1}{n^2}\left[\sum_i \sigma^2 + \sum_i \sum_j \mathrm{Cov}\left(X_i, X_j\right)\right] = \frac{1}{n}\sigma^2
\end{aligned}$$

This means that from sample to sample it will vary with this variance.

An estimator with zero bias is called “unbiased”. This means that on average it will be exactly as what we want.

Estimation of σ^2

$$\begin{aligned}\sigma^2 &= \mathrm{E}\left(X-\mu\right)^2 \\ &= \mathrm{E}\,X^2-\mu^2\end{aligned}$$

$$\begin{aligned}\widehat{\sigma^2} &= \frac{1}{n-1}\sum_i\left(x_i-\overline{X}\right)^2 \\ &= \frac{1}{n-1}\left(\sum_i x_i^2-n\overline{X}^2\right) \\ \mathrm{E}\,\widehat{\sigma^2} &= \mathrm{E}\left[\frac{1}{n-1}\left(\sum_i x_i^2-n\overline{X}^2\right)\right] \\ &= \frac{1}{n-1}\left(n\,\mathrm{E}\,X^2-n\,\mathrm{E}\,\overline{X}^2\right) \\ &= \frac{1}{n-1}\left(n\left(\sigma^2+\mu^2\right)-n\left(\frac{\sigma^2}{n}+\mu^2\right)\right) \\ &= \sigma^2,\end{aligned}$$

therefore, $\widehat{\sigma^2}$ is unbiased for σ^2.

Estimation of $\mathrm{Cov}(X,Y)$

$$\begin{aligned}\mathrm{Cov}(X,Y) &= \mathrm{E}(X-\mu_X)(Y-\mu_Y)\\ &= \mathrm{E}XY-\mu_X\mu_Y\end{aligned}$$

$$\widehat{\mathrm{Cov}}(X,Y) = \frac{1}{n-1}\sum_i\left(x_i-\overline{X}\right)\left(y_i-\overline{Y}\right)$$

$$= \frac{1}{n-1}\left(\sum_i x_iy_i - n\overline{XY}\right)$$

$$\mathrm{E}\widehat{\mathrm{Cov}}(X,Y) = \frac{1}{n-1}\left(n\,\mathrm{E}XY - n\,\mathrm{E}\overline{XY}\right)$$

$$\left(n\,\mathrm{E}XY - n\,\mathrm{E}\overline{XY}\right) =$$

$$= n\left(\mathrm{Cov}(X,Y)+\mu_X\mu_Y\right) - n\,\mathrm{E}\frac{1}{n^2}\sum_i\sum_j x_iy_j$$

$$= n\left(\mathrm{Cov}(X,Y)+\mu_X\mu_Y\right) - \frac{1}{n}\,\mathrm{E}\left(\sum_i x_iy_i + \sum_{i\neq j}\sum x_iy_j\right)$$

$$= n\left(\mathrm{Cov}(X,Y)+\mu_X\mu_Y\right) - \frac{1}{n}\left(n\,\mathrm{E}XY + n(n-1)\,\mathrm{E}x_iy_j\right)$$

$$= n\left(\mathrm{Cov}(X,Y)+\mu_X\mu_Y\right) - \left(\mathrm{Cov}(X,Y)+\mu_X\mu_Y+(n-1)\,\mu_X\mu_Y\right)$$

$$= n\,\mathrm{Cov}(X,Y) + n\mu_X\mu_Y - \left(\mathrm{Cov}(X,Y)+n\mu_X\mu_Y\right)$$

Therefore, $\widehat{\mathrm{Cov}}(X,Y)$ is unbiased as well for $\mathrm{Cov}(X,Y)$

Random Vectors and Multivariate Statistics

A p-dimensional random vector X is $X = (X_1, \ldots, X_p)'$ has joint pdf

$$f_X = f_{X_1,\ldots,X_p}$$

Mean:

$$\begin{aligned}\mu &= \mathrm{E}\, X \\ &= (\mathrm{E}\, X_1, \ldots, \mathrm{E}\, X_p)'\end{aligned}$$

Covariance Matrix Σ $(= \mathrm{Cov}(X))$:

$$
\begin{aligned}
\Sigma &= \mathrm{E}\left(X-\mu\right)\left(X-\mu\right)' \\
&= \mathrm{E}\left[\begin{pmatrix} X_1-\mu_1 \\ \vdots \\ X_p-\mu_p \end{pmatrix}\left(X_1-\mu_1,\ldots,X_p-\mu_p\right)\right] \\
&= \mathrm{E}\begin{pmatrix} \left(X_1-\mu_1\right)^2 & \cdots & \left(X_1-\mu_1\right)\left(X_p-\mu_p\right) \\ \vdots & \ddots & \\ \left(X_p-\mu_p\right)\left(X_1-\mu_1\right) & & \left(X_p-\mu_p\right)^2 \end{pmatrix} \\
&= \begin{pmatrix} \sigma_1^2 & \cdots & \sigma_{1p} \\ \vdots & \ddots & \\ \sigma_{p1} & & \sigma_p^2 \end{pmatrix} \\
&= \begin{pmatrix} \sigma_1^2 & \cdots & \rho_{1p}\sigma_1\sigma_p \\ \vdots & \ddots & \\ \rho_{1p}\sigma_1\sigma_p & & \sigma_p^2 \end{pmatrix}.
\end{aligned}
$$

Prove, for any random vector X, that:
Σ is positive definite (p.d.) matrix; i.e., $\alpha'\Sigma\alpha > 0\ \forall\alpha$. (See next how to generate Σ)

Σ is symmetric; i.e., $\sigma_{ij} = \mathrm{Cov}\left(X_i, X_j\right) = \mathrm{Cov}\left(X_j, X_i\right) = \sigma_{j_i}$

Estimation:

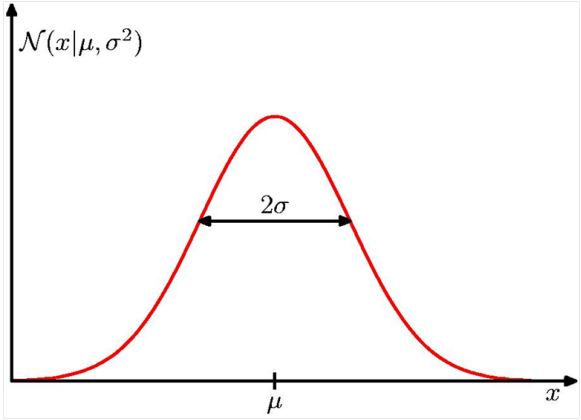
Estimation of μ and Σ is nothing but estimation of their components, which is discussed above. In vector form, it is written as:

$$\widehat{\mu} = \overline{X} = \frac{1}{n}\sum_i x_i,$$

$$= \frac{1}{n}\left[\begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix} + \ldots \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}\right] = \begin{pmatrix} \frac{1}{n}\sum_i x_{i1} \\ \vdots \\ \frac{1}{n}\sum_i x_{ip} \end{pmatrix}$$

$$\widehat{\Sigma} = \frac{1}{n-1}\sum_i \left(x_i - \overline{X}\right)\left(x_i - \overline{X}\right)' =$$

$$\begin{pmatrix} \frac{1}{n-1}\sum_i \left(x_{i1} - \overline{X_1}\right)^2 & \cdots & \frac{1}{n-1}\sum_i \left(x_{i1} - \overline{X_1}\right)\left(x_{ip} - \overline{X_p}\right) \\ \vdots & \ddots & \\ \frac{1}{n-1}\sum_i \left(x_{ip} - \overline{X_p}\right)\left(x_{i1} - \overline{X_1}\right) & & \frac{1}{n-1}\sum_i \left(x_{ip} - \overline{X_p}\right)^2 \end{pmatrix}$$



We know that X is normally distributed if

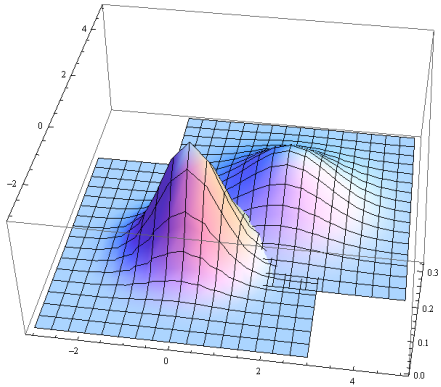
$$f_X(x) = \frac{1}{\left(2\pi\sigma^2\right)^{1/2}} \exp^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}.$$

Prove that:

$$\mathrm{E}\, X = \mu,$$
$$\mathrm{Var}\, X = \sigma^2.$$

So, the population parameters appear explicitly in the pdf. We say, $X \sim \mathcal{N}\left(\mu,\sigma^2\right)$. The figure shows the geometry of the normal distribution.

Multinormal Distribution:

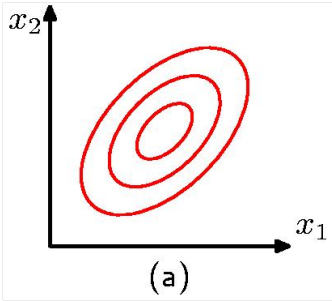


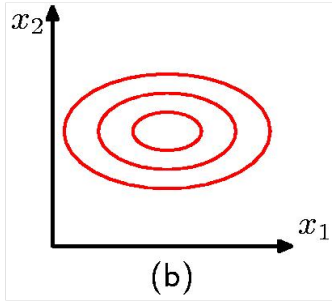
X is said to have a multinormal distribution if

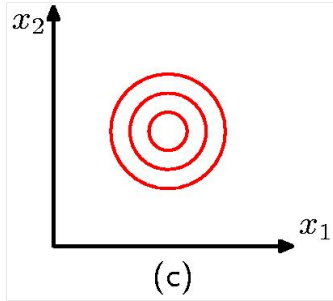
$$f_X(x) = \frac{1}{((2\pi)^p |\Sigma|)^{1/2}} e^{-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)}.$$

It is obvious that for $p = 1$, the pdf is the same as univariate normal.
Prove that:

$$\mathrm{E}\, X = \mu,$$
$$\mathrm{Cov}(X) = \Sigma.$$







You can generate a covariance matrix by choosing $\sigma_i^2, i = 1, \ldots, p$; then choose ρ_{ij}, where $-1 \le \rho_{ij} \le 1$. In the case of $p = 2$, we have many cases:
(a) $\rho > 0$
(b) $\rho = 0$, and $\sigma_2 < \sigma_1$.
(c) $\rho = 0$, and $\sigma_2 = \sigma_1$.

For diagonal Σ (uncorrelated components),

$$f_X(x) = \frac{1}{\prod_{i=1}^{p} \left(2\pi\sigma_i^2\right)^{1/2}} \exp\left[\sum_{i=1}^{p} -\frac{1}{2}\frac{(x-\mu_i)^2}{\sigma_i^2}\right]$$
$$= \prod_i \frac{1}{\left(2\pi\sigma_i^2\right)^{1/2}} \exp\left[-\frac{1}{2}\frac{(x-\mu_i)^2}{\sigma_i^2}\right],$$

which is joint of p independent normals. This is not the case for other distributions.

Transformation by Projection

For any random vector $X_{p\times 1}$, any matrix of m column vectors $A_{p\times m}$ can be seen as a set of m projections from $\mathcal{R}^p$ to $\mathcal{R}^m$: $A'_{m\times p} = \begin{pmatrix} \alpha'_1 \\ \vdots \\ \alpha'_m \end{pmatrix}$

$\mathrm{E}\, A'_{m\times p} X_{p\times 1} = A'\,\mathrm{E}\, X = A'\mu.$

$\mathrm{Cov}\left(A'_{m\times p} X_{p\times 1}\right) = A'\,\mathrm{Cov}\,(X)\,A = A'\Sigma A.$

Theorem

If $X \sim \mathcal{N}(\mu, \Sigma)$ then $A'X \sim \mathcal{N}(A'\mu, A'\Sigma A)$.

Simple example (a single projection): $A'_{1\times p} = \alpha'$

if $\alpha' = (1, 0, \ldots, 0)$, we simply get the first component X_1.

if $\alpha' = (1/\sqrt{2}, 1/\sqrt{2})$, we project on the $\pi/4$ direction; this will be the value of the new r.v. in the direction of α.

$$\begin{aligned} \alpha' X &= \|\alpha\|\,\|X\| \cos(x, \alpha) \\ &= \|\alpha\| \times \text{ Projected Length} \end{aligned}$$

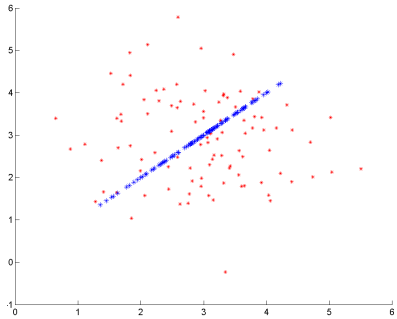
If we need the projected length only, then project on a unit vector $\frac{\alpha'}{\|\alpha\|}$, then

$$\frac{\alpha'}{\|\alpha\|} X = \|X\| \cos(x, \alpha).$$

Multiply this scalar in the direction of the projection to get the new feature Z in the direction α

$$\begin{aligned} Z^{(\alpha)} &= \left(\frac{\alpha}{\|\alpha\|}\right)_{p\times 1}\left(\frac{\alpha'}{\|\alpha\|}X\right)_{1\times 1} \\ &= \frac{\alpha\alpha'}{(\alpha'\alpha)}X \\ &= P^{(\alpha)}_{p\times p}X_{p\times 1}, \end{aligned}$$

where we call P the projection matrix of the direction α.



Matlab Code B.1:

```
mu=3+zeros(1, 2); sigma= eye(2); samples=100;
X=mvnrnd(mu, sigma, samples);
scatter(X(:,1),X(:,2),20,'*r'); hold on

alpha=[1/sqrt(2); 1/sqrt(2)];
proj=X*alpha;

Xnew=repmat(proj, [1 2]).*repmat(alpha', [samples 1]);
scatter(Xnew(:, 1), Xnew(:, 2), '*b');


%the following should be equivalent
var(proj)
alpha'*cov(X)*alpha
```

Estimation of scalar quantities can be obtained from multivariate versions by:
$Z = (x_1, \ldots, x_n)$ has mean and covariance matrix $\mu\mathbf{1}$ and $\sigma^2\mathbf{I}$ respectively, (since $x_i, i = 1, \ldots n,$ are i.i.d), where $\mathbf{1} = (1, \ldots, 1)'$. Then,

$$\overline{X} = \frac{1}{n}\sum_i x_i = \frac{1}{n}\mathbf{1}'Z,$$

$$\mathrm{E}\,\overline{X} = \left(\frac{1}{n}\mathbf{1}'\right)(\mu\mathbf{1}) = \frac{1}{n}\mu\mathbf{1}'\mathbf{1} = \mu$$

$$\mathrm{Var}\,\overline{X} = \left(\frac{1}{n}\mathbf{1}'\right)(\sigma^2\mathbf{I})\left(\frac{1}{n}\mathbf{1}'\right)' = \frac{\sigma^2}{n^2}\mathbf{1}'\mathbf{1} = \frac{\sigma^2}{n}$$

Appendix C

Fast Revision on Geometry, Linear Algebra, and Matrix Theory

Inner product and orthogonalization

For vectors in p dimensional space, the inner product $\langle x, y\rangle$ is the dot product $x'y$,

$$\begin{aligned}\langle x, y\rangle &= x'y \\ &= \begin{pmatrix} x_1 & \dots & x_p \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ \vdots \\ y_p \end{pmatrix} \\ &= \sum_{i=1}^{p} x_i y_i.\end{aligned}$$

when $x'y$ is zero we say they are orthogonal. It can be shown that, for $p \leq 3$,

$$\begin{aligned}\cos\theta &= \frac{x'y}{\|x\| \cdot \|y\|} \\ &= \frac{x'y}{\sqrt{x'x} \cdot \sqrt{y'y}}\end{aligned}$$

Then, we can generalize this definition in higher dimensions, and define the angle between two vectors for $p > 3$.

Simple example (a single projection): $\mathbf{X}_{p\times 1} = X$
if $X' = (1, 0, \dots, 0)$, we simply get the first component Y_1.

if $X' = (1/\sqrt{2}, 1/\sqrt{2})$, we project on the $\pi/4$ direction; this will be the value of the new vector in the direction of X.

$$\begin{aligned} X'Y &= \|X\|\,\|Y\|\cos(Y,X) \\ &= \|X\| \times \text{ Projected Length} \end{aligned}$$

If we need the projected length only, then project on a unit vector $\frac{X'}{\|X\|}$, then

$$\frac{X'}{\|X\|}Y = \|Y\|\cos(Y,X).$$

Multiply this scalar in the direction of the projection to get the new component in the direction X

$$\begin{aligned} \widehat{Y} &= \left(\frac{X}{\|X\|}\right)\left(\frac{X'}{\|X\|}Y\right) \\ &= X\widehat{\beta} \\ &= \frac{XX'}{(X'X)}Y \\ &= P^{(X)}_{p\times p}Y_{p\times 1}, \end{aligned}$$

where we call P the projection matrix of the direction X.

For decomposition (not projection yet) on set of vectors constituting the columns of $\mathbf{X} = (X_1, \ldots, X_n)$,

$$\begin{aligned} &= X_1\widehat{\beta}_1 + \ldots + X_n\widehat{\beta}_n \\ &= \mathbf{X}\widehat{\beta} \end{aligned}$$

We need to minimize the remaining error

$$
\begin{aligned}
e &= Y - \mathbf{X}\widehat{\beta}, \\
\|e\|^2 &= \left\langle \left(Y - \mathbf{X}\widehat{\beta}\right), \left(Y - \mathbf{X}\widehat{\beta}\right) \right\rangle \\
&= \langle Y, Y \rangle - 2\beta' \begin{pmatrix} \langle X_1, Y \rangle \\ \vdots \\ \langle X_n, Y \rangle \end{pmatrix} + \beta' \begin{pmatrix} \langle X_1, X_1 \rangle & \dots & \langle X_1, X_n \rangle \\ \vdots & \ddots & \vdots \\ \langle X_n, X_1 \rangle & \dots & \langle X_n, X_n \rangle \end{pmatrix} \beta \\
&= Y'Y - 2\beta'\mathbf{X}'Y + \beta'\mathbf{X}'\mathbf{X}\beta
\end{aligned}
$$

$$
\begin{aligned}
\nabla \|e\|^2 &= -2\mathbf{X}'\mathbf{y} + \mathbf{2}\mathbf{X}'X\beta \\
&= -2\mathbf{X}'\left(\mathbf{y} - \mathbf{X}\beta\right),
\end{aligned}
$$

which means that the error is perpendicular to each component X_i; i.e.,

$$
\begin{aligned}
\langle X_i, e \rangle &= 0, \text{ or} \\
X_i' e &= 0
\end{aligned}
$$

The solution will be

$$
\widehat{\beta} = \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'Y
$$

Therefore, the projection $\widehat{Y}$ is

$$
\begin{aligned}
\widehat{Y} &= \mathbf{X}\widehat{\beta} \\
&= \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'Y
\end{aligned}
$$

and the projection matrix $\mathbf{P}$ is

$$\mathbf{P} = \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'$$

The very interesting thing is

$$\begin{aligned}\mathbf{X}'\mathbf{X} &= \begin{pmatrix} X_1' \\ \vdots \\ X_n' \end{pmatrix} \begin{pmatrix} X_1 & \dots & X_n \end{pmatrix} \\ &= \begin{pmatrix} X_1'X_1 & X_1'X_2 & \dots & X_1'X_n \\ X_2'X_1 & X_2'X_2 & \dots & X_2'X_n \\ \vdots & \vdots & \vdots & \vdots \\ X_n'X_1 & X_n'X_2 & \dots & X_n'X_n \end{pmatrix}\end{aligned}$$

if we choose the basis X_i orthogonal

$$\begin{aligned}\mathbf{X}'\mathbf{X} &= \begin{pmatrix} X_1'X_1 & & & \\ & X_2'X_2 & & \\ & & \ddots & \\ & & & X_n'X_n \end{pmatrix} \\ &= \operatorname{diag}\left(\|X_1\|^2, \dots, \|X_n\|^2\right),\end{aligned}$$

$$
\begin{aligned}
\mathbf{P} &= \begin{pmatrix} X_1 & \dots & X_n \end{pmatrix} \begin{pmatrix} \frac{1}{\|X_1\|^2} & & \\ & \ddots & \\ & & \frac{1}{\|X_n\|^2} \end{pmatrix} \begin{pmatrix} X_1' \\ \vdots \\ X_n' \end{pmatrix} \\
&= \frac{X_1 X_1'}{X_1' X_1} + \dots + \frac{X_n X_n'}{X_n' X_n} \\
&= \frac{X_1}{\|X_1\|} \frac{X_1'}{\|X_1\|} + \dots + \frac{X_n}{\|X_n\|} \frac{X_n'}{\|X_n\|} \\
&= \mathbf{P}_1 + \dots + \mathbf{P}_n
\end{aligned}
$$

The projection will be

$$\widehat{Y} = \mathbf{P}_1 Y + \dots \mathbf{P}_n Y$$

The orthogonality of the basis made it possible to project on each and sum up the projections. If the set of basis X_i span the whole space, then they are complete and $\widehat{Y} = Y$ and the error is zero; which means we can express Y in terms of the new subspace **X**. And for **orthonormal** basis, where $\|X_i\| = 1$

$$
\begin{aligned}
\mathbf{P}_i &= X_i X_i', \\
\widehat{\beta}_i &= X_i' Y
\end{aligned}
$$

Appendix D

Fast Revision on Multivariate Statistics

Bibliography

Ambroise, C., McLachlan, G. J., 2002. Selection Bias in Gene Extraction on the Basis of Microarray Gene-Expression data. PNAS 99 (10), 6562–6566.

Cherkassky, V. S., Mulier, F., 1998. Learning from data : concepts, theory, and methods. Wiley, New York.

Dave, S. S., Wright, G., Tan, B., Rosenwald, A., Gascoyne, R. D., Chan, W. C., Fisher, R. I., Braziel, R. M., Rimsza, L. M., Grogan, T. M., Miller, T. P., LeBlanc, M., Greiner, T. C., Weisenburger, D. D., Lynch, J. C., Vose, J., Armitage, J. O., Smeland, E. B., Kvaloy, S., Holte, H., Delabie, J., Connors, J. M., Lansdorp, P. M., Ouyang, Q., Lister, T. A., Davies, A. J., Norton, A. J., Muller-Hermelink, H. K., Ott, G., Campo, E., Montserrat, E., Wilson, W. H., Jaffe, E. S., Simon, R., Yang, L., Powell, J., Zhao, H., Goldschmidt, N., Chiorazzi, M., Staudt, L. M., 2004. Prediction of Survival in Follicular Lymphoma Based on Molecular Features of Tumor-Infiltrating Immune cells. New England Journal of Medicine November 351 (21), 2159–2169.

Devroye, L., 1982. Any Discrimination Rule Can Have an Arbitrarily Bad Probability of Error for Finite Sample Size. Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-4 (2), 154–157.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., Lander, E. S., 1999. Molecular Classification of Cancer: Class Discovery and Class Prediction By Gene Expression Monitoring. Science 286 (5439), 531–537.

Gur, D., Wagner, R. F., Chan, H. P., 2004. On the Repeated Use of Databases for Testing Incremental Improvement of Computer-Aided Detection schemes. Acad Radiol 11 (1), 103–105.

Hastie, T., Tibshirani, R., Friedman, J. H., 2009. The elements of statistical learning: data mining, inference, and prediction, 2nd Edition. Springer, New York.

Lee, S., 2008. Mistakes in Validating the Accuracy of a Prediction Classifier in High-Dimensional But Small-Sample Microarray data. Statistical Methods in Medical Research 17 (6), 635–642.
URL https://doi.org/Doi10.1177/0962280207084839

Lim, T. S., Loh, W. Y., Shih, Y. S., 2000. A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification algorithms. Machine Learning 40 (3), 203–228.

Simon, R., Radmacher, M. D., Dobbin, K., McShane, L. M., 2003. Pitfalls in the Use of Dna Microarray Data for Diagnostic and Prognostic classification. Journal of the National Cancer Institute 95 (1), 14–18.

Tibshirani, R., 2005. Immune Signatures in Follicular lymphoma. N Engl J Med 352 (14), 1496–1497.
URL https://doi.org/352/14/1496[pii]10.1056/NEJM200504073521422