ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Selected topics for revision

Applied Multivariate Statistics – Spring 2012

Review of

- Gaussian Mixture Models
- LDA
- Random Forest

Gaussian Mixture Models (GMMs)

Gaussian Mixture Models (GMM)

- Gaussian Mixture Model:
  $f(x; p, \theta) = \sum_{j=1}^{K} p_j g_j\left(x; \theta_j\right)$
  K populations with different probability distributions

- Find number of classes and parameters p_j and θ_j given data
- Assign observation x to cluster j, where estimated value of

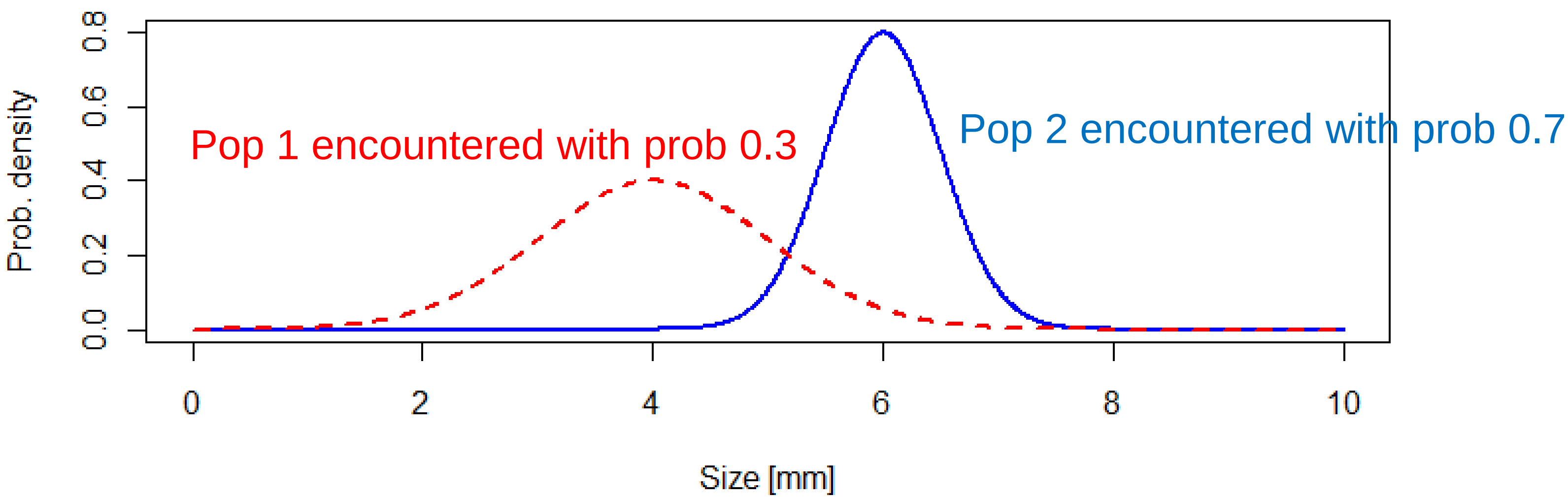
$$P(cluster\ j|x) = \frac{p_j g_j(x; \theta_j)}{f(x; p, \theta)}$$

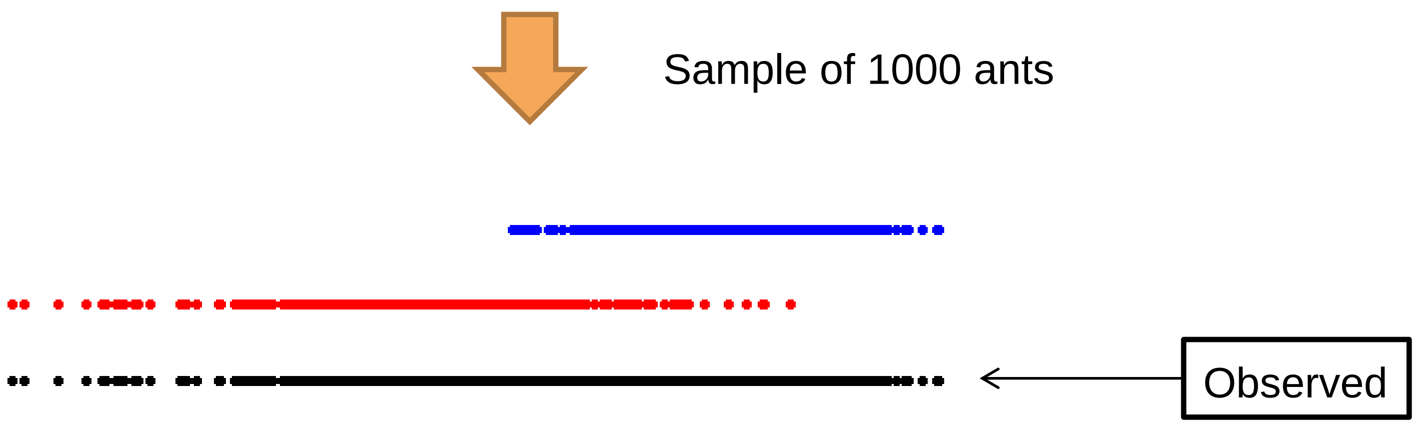
  is largest

Example (1/6): Size of ants in two populations

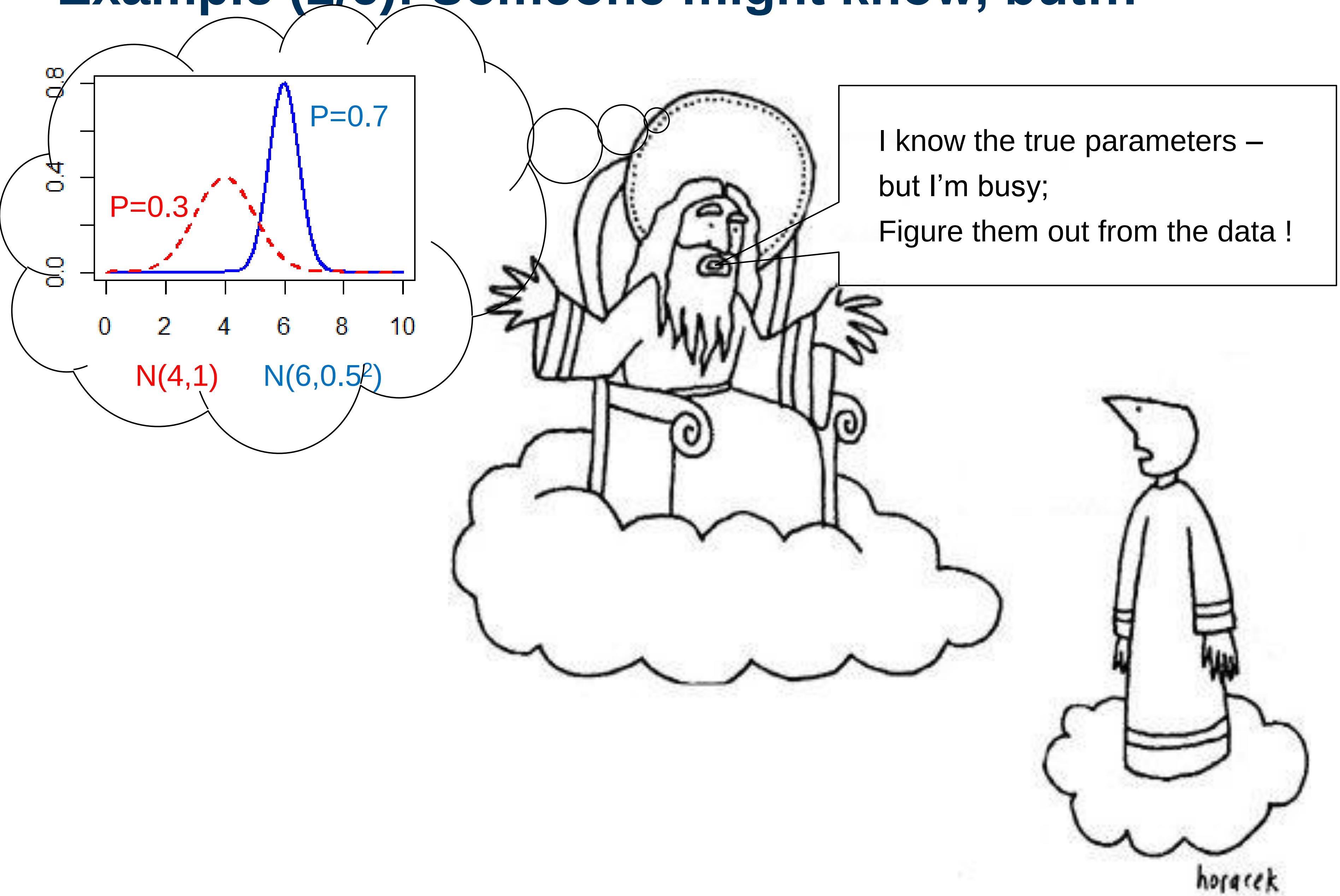
Suppose ants ***look the same apart from size***:

How can we learn about the two populations, if we can only observe a mixture of them ?

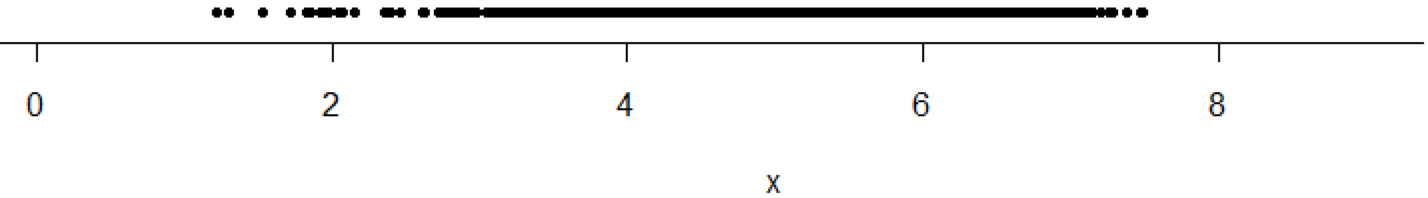




Example (2/6): Someone might know, but…



Example (3/6): We just see this



and we guess that there are two Normal populations involved

Example (4/6): How likely is the observation?

- Likelihood function for one observation x:

$$\begin{aligned} f(x;p,\theta) &= p \cdot \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp(-(x-\mu_1)^2/2\sigma_1^2) + \\ &+ (1-p) \cdot \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp(-(x-\mu_2)^2/2\sigma_2^2) \end{aligned}$$

  Parameters to estimate: p, μ_1, μ_2, σ_1, σ_2

- Likelihood function for n (independent) observations $x_1,\ldots,x_n$:

$$\tilde{f}(x_1, ..., x_n; p, \theta) = \prod_{i=1}^{n} f(x_i; p; \theta)$$

- For numerical reasons, compute log-Likelihood function:

$$l(x_1, ..., x_n; p, \theta) = \log(\tilde{f}(x_1, ..., x_n; p, \theta))$$

Example (5/6): Find the set of parameters under which the observation is most likely

Guessing the parameters:

p	μ_1	μ_2	σ_1	σ_2	Log-Likelihood
0.5	3	5	2	1	-1891
0.4	3.5	5.5	1	0.5	-1723
0.7	5	7	1	1	-1678
Etc.					

Using some numerical optimization technique:

p	μ_1	μ_2	σ_1	σ_2	Log-Likelihood
0.35	4.18	6.03	1.05	0.47	-1365

True parameters:

p	μ_1	μ_2	σ_1	σ_2	Log-Likelihood
0.3	4	6	1	0.5	-1366

Example (6/6): Doing it with R

```
> res <- Mclust(xobs)
> str(res)
List of 11
 $ modelName     : chr "V"
 $ n             : int 1000
 $ d             : num 1
 $ G             : int 2
 $ BIC           : num [1:9, 1:2] -3120 -2826 -2840 -2854 -2812 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:9] "1" "2" "3" "4" ...
  .. ..$ : chr [1:2] "E" "V"
  ..- attr(*, "G")= num [1:9] 1 2 3 4 5 6 7 8 9
  ..- attr(*, "modelNames")= chr [1:2] "E" "V"
  ..- attr(*, "oneD")= logi TRUE
 $ bic           : num -2765
 $ loglik        : num -1365
 $ parameters    :List of 4
  ..$ Vinv    : NULL
  ..$ pro     : num [1:2] 0.347 0.653
  ..$ mean    : Named num [1:2] 4.18 6.03
  .. ..- attr(*, "names")= chr [1:2] "1" "2"
  ..$ variance:List of 5
  .. ..$ modelName: chr "V"
  .. ..$ d        : num 1
  .. ..$ G        : int 2
  .. ..$ sigmasq  : num [1:2] 1.113 0.223
  .. ..$ scale    : num [1:2] 1.113 0.223
 $ classification: num [1:1000] 1 1 1 1 1 1 1 1 1 1 ...
 $ uncertainty   : num [1:1000] 2.06e-01 6.30e-09 7.56e-02 1.57e-02 1.15e-02 ...
 $ z             : num [1:1000, 1:2] 0.794 1 0.924 0.984 0.988 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : NULL
 - attr(*, "class")= chr "Mclust"
```

Vector with observations

Two groups were found

Optimized log-likelihood

Probability of group 1

Probability of group 2

Mean of group 2

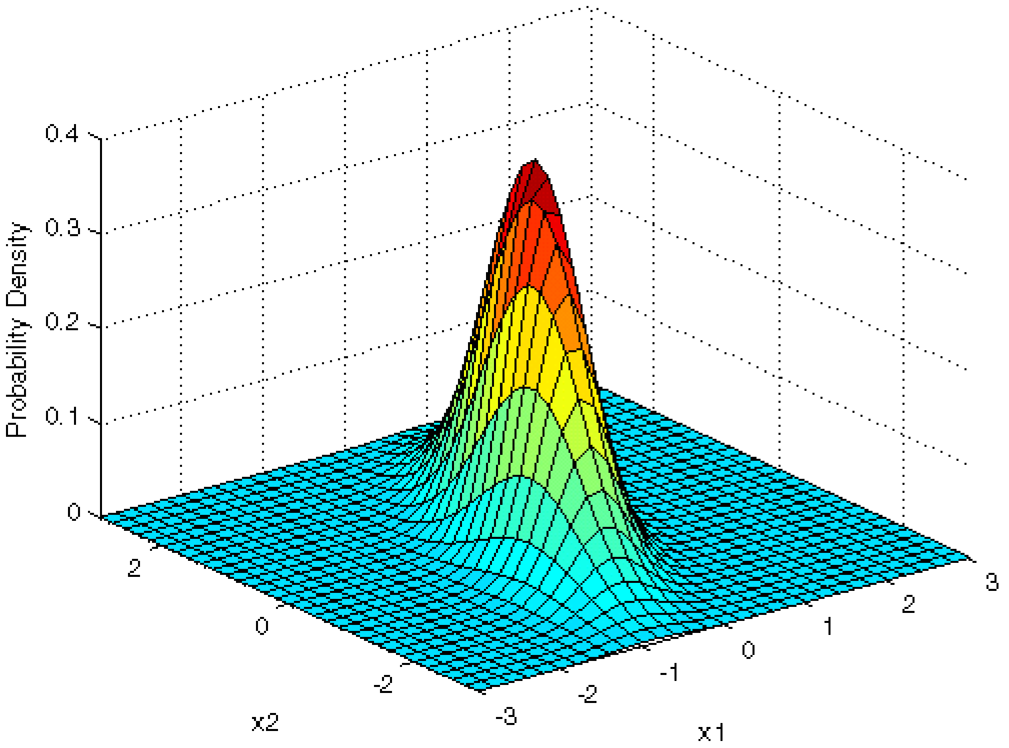
Mean of group 1

Variance of group 2

Variance of group 1

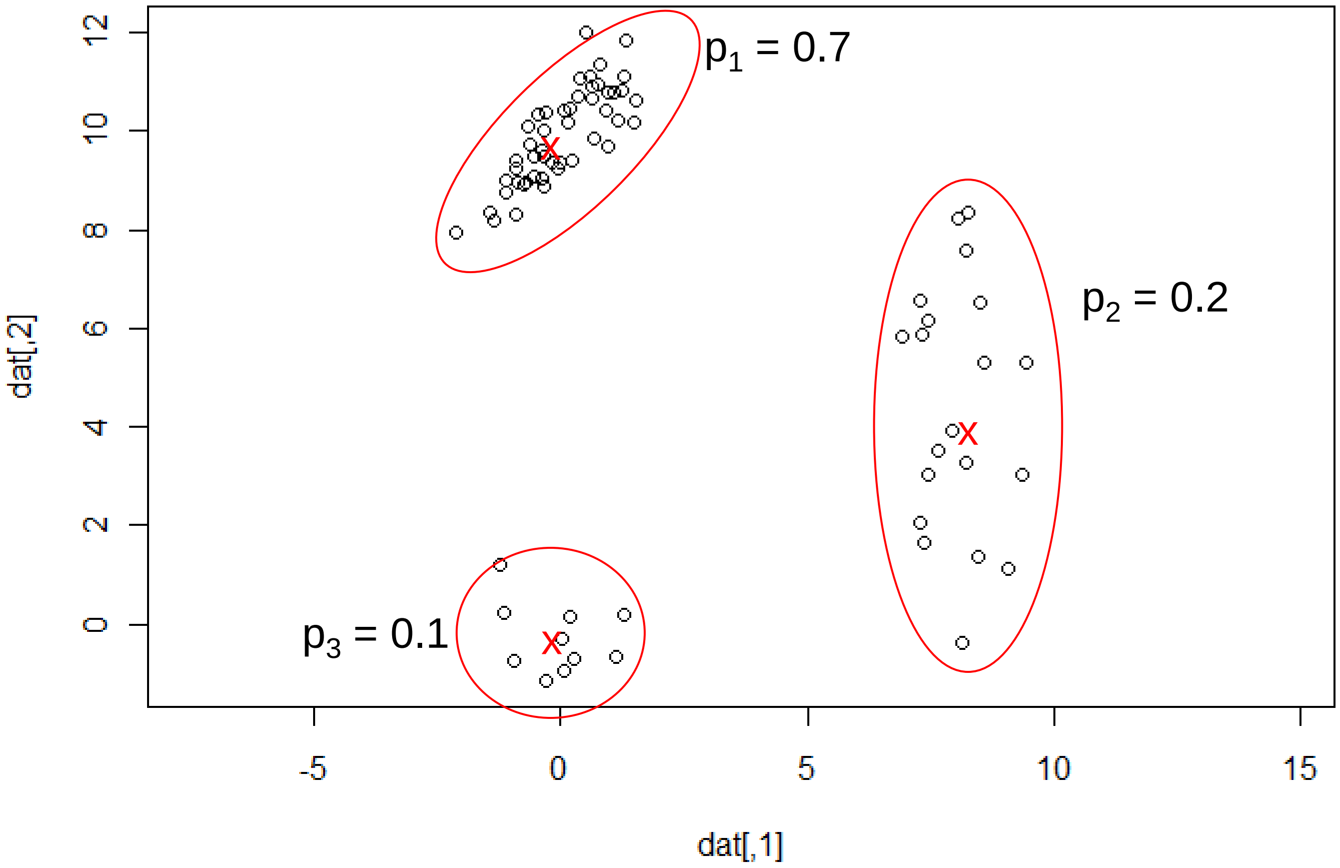
Revision: Multivariate Normal Distribution

$$f(x; \mu, \Sigma) = \frac{1}{\sqrt{2\pi|\Sigma|}} \exp\Big(-\frac{1}{2} \cdot (x-\mu)^T \Sigma^{-1} (x-\mu) \Big)$$



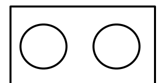
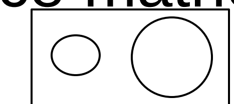
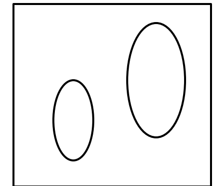
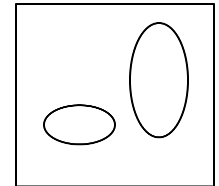
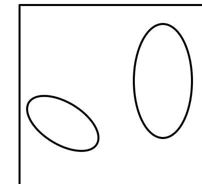
GMM: Example estimated manually

- 3 clusters
- $p_1 = 0.7$, $p_2 = 0.2$, $p_3 = 0.1$
- Mean vector and cov. Matrix per cluster



Fitting GMMs 1/2

- Maximum Likelihood Method
  Hard optimization problem
- Simplification: Restrict Covariance matrices to certain patterns (e.g. diagonal)

identifier	Model	HC	EM	Distribution	Volume	Shape	Orientation
E		•	•	(univariate)	equal		
V		•	•	(univariate)	variable		
EII	λI	•	•	Spherical	equal	equal	NA
VII	$\lambda_k I$	•	•	Spherical	variable	equal	NA
EEI	λA		•	Diagonal	equal	equal	coordinate axes
VEI	$\lambda_k A$		•	Diagonal	variable	equal	coordinate axes
EVI	λA_k		•	Diagonal	equal	variable	coordinate axes
VVI	$\lambda_k A_k$		•	Diagonal	variable	variable	coordinate axes
EEE	$\lambda D A D^T$	•	•	Ellipsoidal	equal	equal	equal
EEV	$\lambda D_k A D_k^T$		•	Ellipsoidal	equal	equal	variable
VEV	$\lambda_k D_k A D_k^T$		•	Ellipsoidal	variable	equal	variable
VVV	$\lambda_k D_k A_k D_k^T$	•	•	Ellipsoidal	variable	variable	variable

Fitting GMMs 2/2

- Problem: Fit will never get worse if you use more cluster or allow more complex covariance matrices
  → How to choose optimal model ?
- Solution: Trade-off between model fit and model complexity

  BIC = log-likelihood – log(n)/2*(number of parameters)

  Find solution with maximal BIC

GMMs in R

- Function “Mclust” in package “mclust”

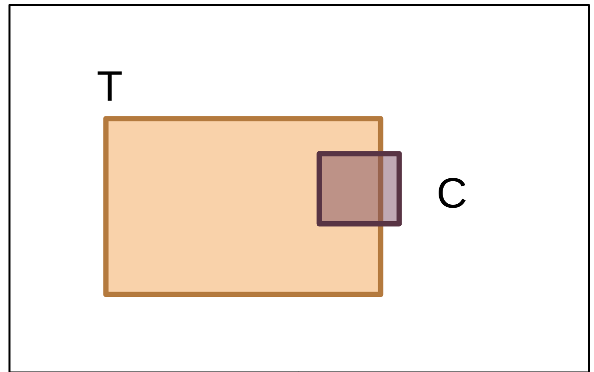
Linear Discriminant Analysis (LDA)

Conditional Probability

Sample space

T: Med. Test positive

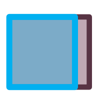
C: Patient has cancer



(Marginal) Probability:
P(T), P(C)

New sample space:
People with cancer

P(T|C)



large

Conditional Probability:
P(T|C), P(C|T)

New sample space:
People with pos. test

P(C|T)

small

Bayes Theorem:

posterior

$$\boxed{P(C|T)} = \frac{\boxed{P(T|C)}\boxed{P(C)}}{P(T)}$$

Class conditional probability

prior

One approach to supervised learning

$$P(C|X) = \frac{P(C)P(X|C)}{P(X)} \sim P(C)P(X|C)$$

Find some estimate

Prior / prevalence:
Fraction of samples
in that class

Assume:
$$X|C \sim N(\mu_c, \Sigma_c)$$

Bayes rule:

Choose class where P(C|X) is maximal

(rule is “optimal” if all types of error are equally costly)

Special case: Two classes (0/1)

- choose c=1 if P(C=1|X) > 0.5 or
- choose c=1 if posterior odds P(C=1|X)/P(C=0|X) > 1

In Practice: Estimate $P(C), \mu_C, \Sigma_C$

QDA: Doing the math… $\frac{1}{\sqrt{(2\pi)^d|\Sigma_C|}}\exp\left(-\frac{1}{2}(x-\mu_c)^T\Sigma_C^{-1}(x-\mu_c)\right)$

- $P(C|X) \sim P(C)P(X|C)$
- Use the fact: $\max P(C|X) \Leftrightarrow \max(\log(P(C|X)))$
- $\delta_c(x) = \log(P(C|X)) = \log(P(C)) + \log(P(X|C)) =$

$$= \underbrace{\log(P(C))}_{\text{Prior}} - \frac{1}{2}\underbrace{\log(|\Sigma_C|)}_{\text{Additional term}} - \frac{1}{2}\underbrace{(x-\mu_C)^T\Sigma_C^{-1}(x-\mu_C)}_{\text{Sq. Mahalanobis distance}} + c$$

- Choose class where $\delta_c(x)$ is maximal
- Special case: Two classes
  Decision boundary: Values of x where $\delta_0(x) = \delta_1(x)$ is quadratic in x

- **Quadratic Discriminant Analysis (QDA)**

Simplification

- Assume same covariance matrix in all classes, i.e.

$$X|C \sim N(\mu_c, \Sigma)$$

Fix for all classes

- $$\delta_c(x) = \log\big(P(C)\big) - \frac{1}{2}\log(|\Sigma|) - \frac{1}{2}(x-\mu_C)^T\Sigma^{-1}(x-\mu_C) + c =$$

Prior

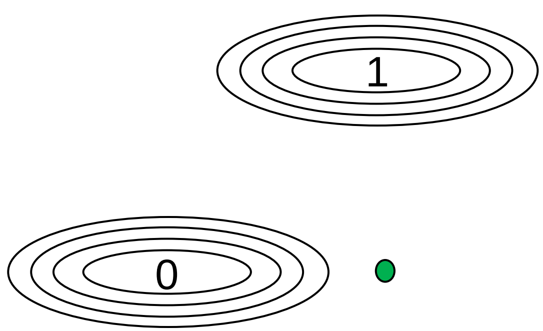
$$= \log\big(P(C)\big) - \frac{1}{2}(x-\mu_C)^T\Sigma^{-1}(x-\mu_C) + d =$$

Sq. Mahalanobis distance

$$(= \log\big(P(C)\big) + x^T\Sigma^{-1}\mu_C - \frac{1}{2}\,\mu_C^T\,\Sigma^{-1}\mu_C)$$

Decision boundary is linear in x

- **Linear Discriminant Analysis (LDA)**



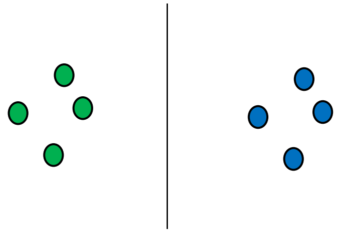
Classify to which class (assume equal prior)?

- Physical distance in space is equal
- Classify to class 0, since Mahal. Dist. is smaller

LDA vs. QDA

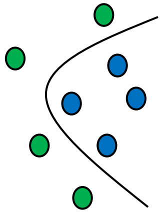
+ Only few parameters to estimate; accurate estimates

- Inflexible (linear decision boundary)



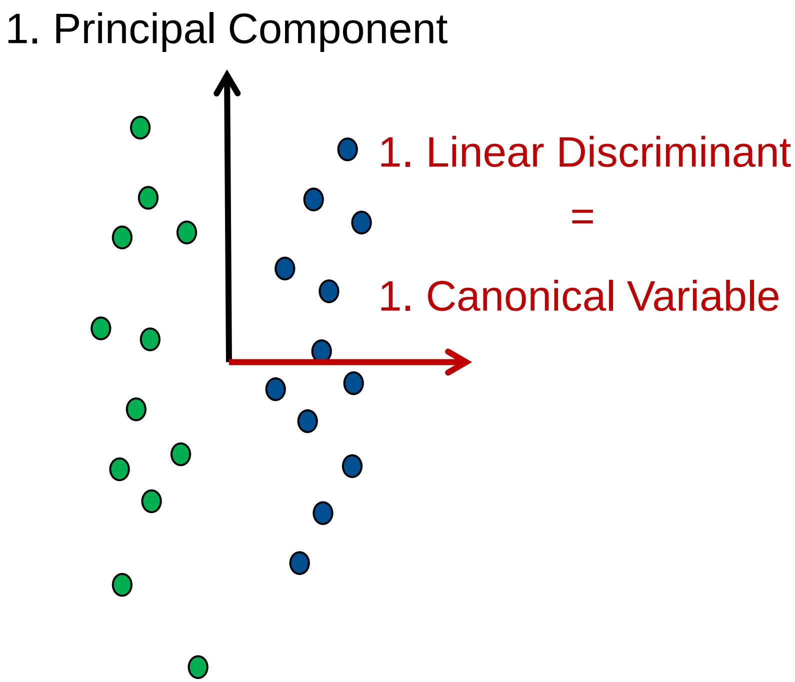
- Many parameters to estimate; less accurate

+ More flexible (quadratic decision boundary)



Fisher's Discriminant Analysis: Idea

Find direction(s) in which groups are separated best

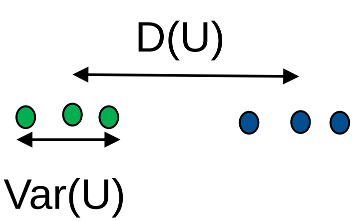


- Class Y, predictors $X = (X_1, \dots, X_d)$
  $\rightarrow U = w^T X$
- Find w so that groups are separated along U best
- Measure of separation: Rayleigh coefficient

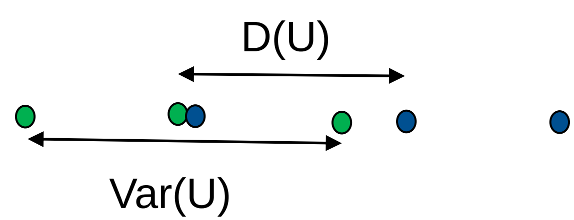
$$J(w) = \frac{D(U)}{Var(U)}$$

where $D(U) = \big(E(U|Y=0) - E(U|Y=1)\big)^2$

- $E[X|Y=j] = \mu_j, Var(X|Y=j) = \Sigma$
  $\Rightarrow E[U|Y=j] = w^T\mu_j, V(U) = w^T\Sigma \mathrm{w}$
- Concept extendable to many groups



$J(w)$ large



$J(w)$ small

LDA and Linear Discriminants

- - Direction with largest J(w): 1. Linear Discriminant (LD 1)
  - orthogonal to LD1, again largest J(w): LD 2
  - etc.
- At most: min(Nmb. dimensions, Nmb. Groups -1) LD's
  e.g.: 3 groups in 10 dimensions – need 2 LD's

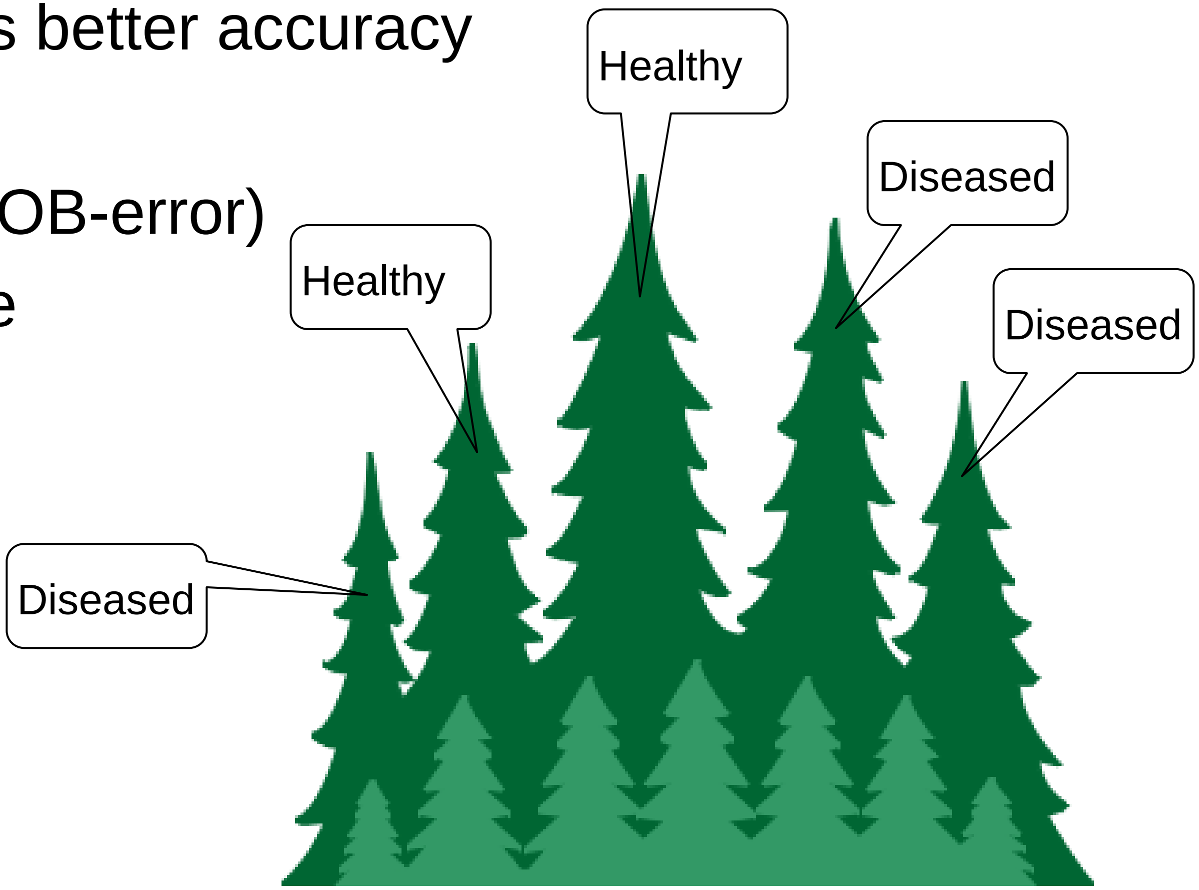
- R: Function «lda» in package MASS does LDA and computes linear discriminants (also «qda» available)

Random Forest

Random Forest

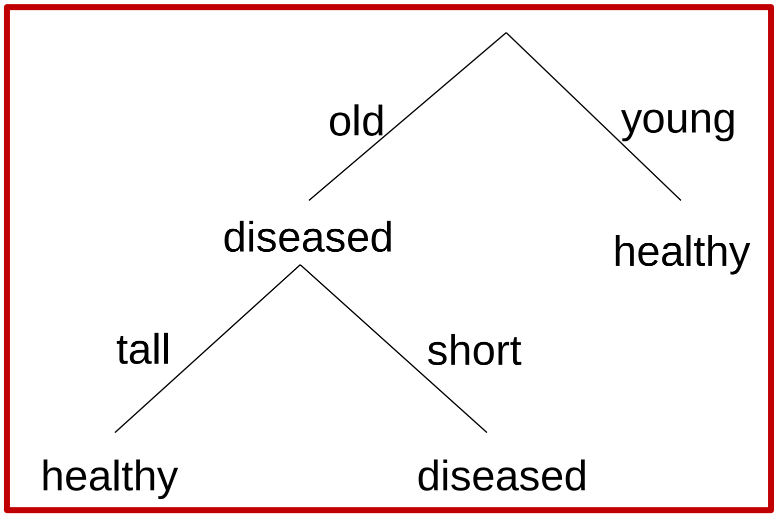
- Intuition of Random Forest
- The Random Forest Algorithm
- De-correlation gives better accuracy

- Out-of-bag error (OOB-error)
- Variable importance

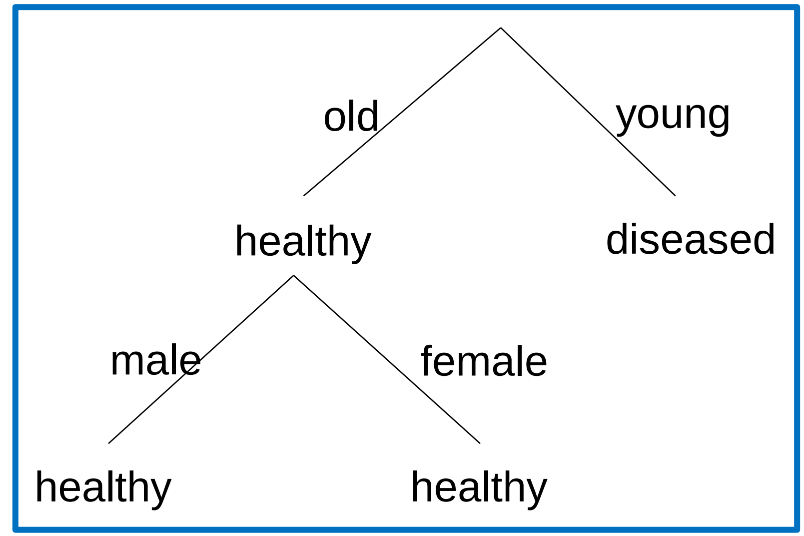


Intuition of Random Forest

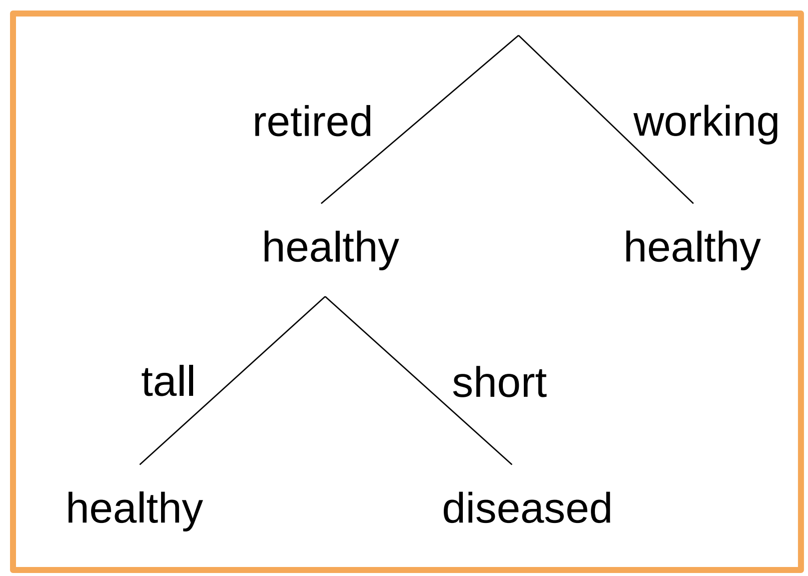
Tree 1



Tree 2



Tree 3



New sample:

old, retired, male, short

Tree predictions:

diseased, healthy, diseased

Majority rule:

diseased

The Random Forest Algorithm

1. For $b = 1$ to B:

   (a) Draw a $\boxed{\text{bootstrap sample}}$ $\mathbf{Z}^*$ of size N from the training data.

   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

      i. Select $\boxed{m \text{ variables at random}}$ from the p variables.

      ii. Pick the best variable/split-point among the m.

      iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

Differences to standard tree

- Train each tree on bootstrap resample of data
  (Bootstrap resample of data set with N samples:
  Make new data set by drawing **with replacement** N samples; i.e., some samples will probably occur multiple times in new data set)
- For each split, consider only m randomly selected variables
- Don’t prune
- Fit B trees in such a way and use average or majority voting to aggregate results

Why Random Forest works 1/2

- Mean Squared Error = Variance + $Bias^2$
- If trees are sufficiently deep, they have very small bias

- How could we improve the variance over that of a single tree?

Why Random Forest works 2/2

$$Var\left(\frac{1}{B}\sum_{i=1}^{B}T_i(c)\right) = \frac{1}{B^2}\sum_{i=1}^{B}\sum_{j=1}^{B}Cov(T_i(x),T_j(x))$$

i=j

$$= \frac{1}{B^2}\sum_{i=1}^{B}\left(\sum_{j\neq i}^{B}Cov(T_i(x),T_j(x)) + Var(T_i(x))\right)$$

$$= \frac{1}{B^2}\sum_{i=1}^{B}\left((B-1)\sigma^2\cdot\rho + \sigma^2\right)$$

$$= \frac{B(B-1)\rho\sigma^2 + B\sigma^2}{B^2}$$

De-correlation gives better accuracy

$$= \frac{(B-1)\rho\sigma^2}{B} + \frac{\sigma^2}{B}$$

$$= \rho\sigma^2 - \frac{\rho\sigma^2}{B} + \frac{\sigma^2}{B}$$

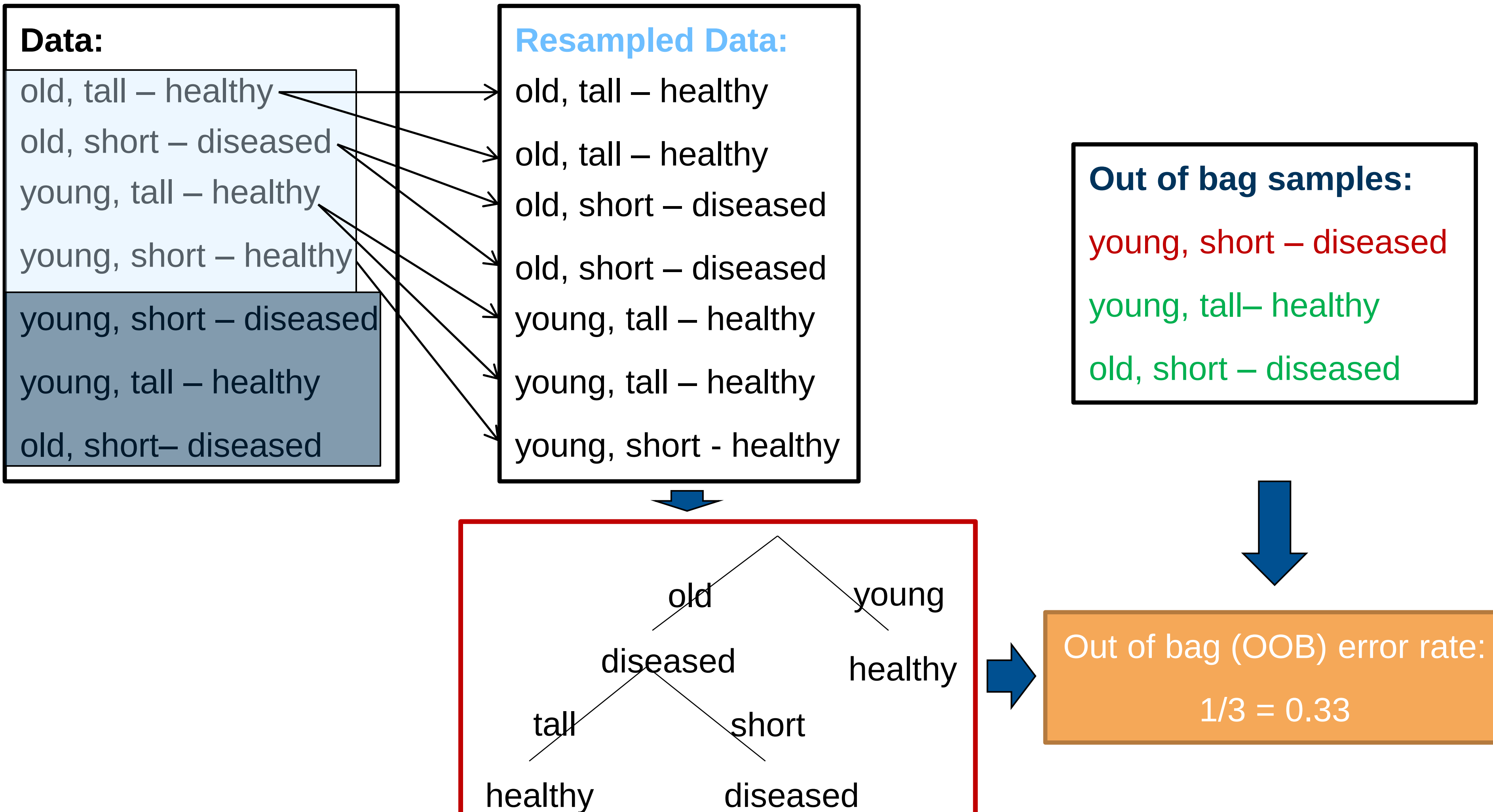
$$= \boxed{\rho\sigma^2} + \boxed{\sigma^2\frac{1-\rho}{B}}$$

Decreaes, if ρ decreases, i.e., if m decreases

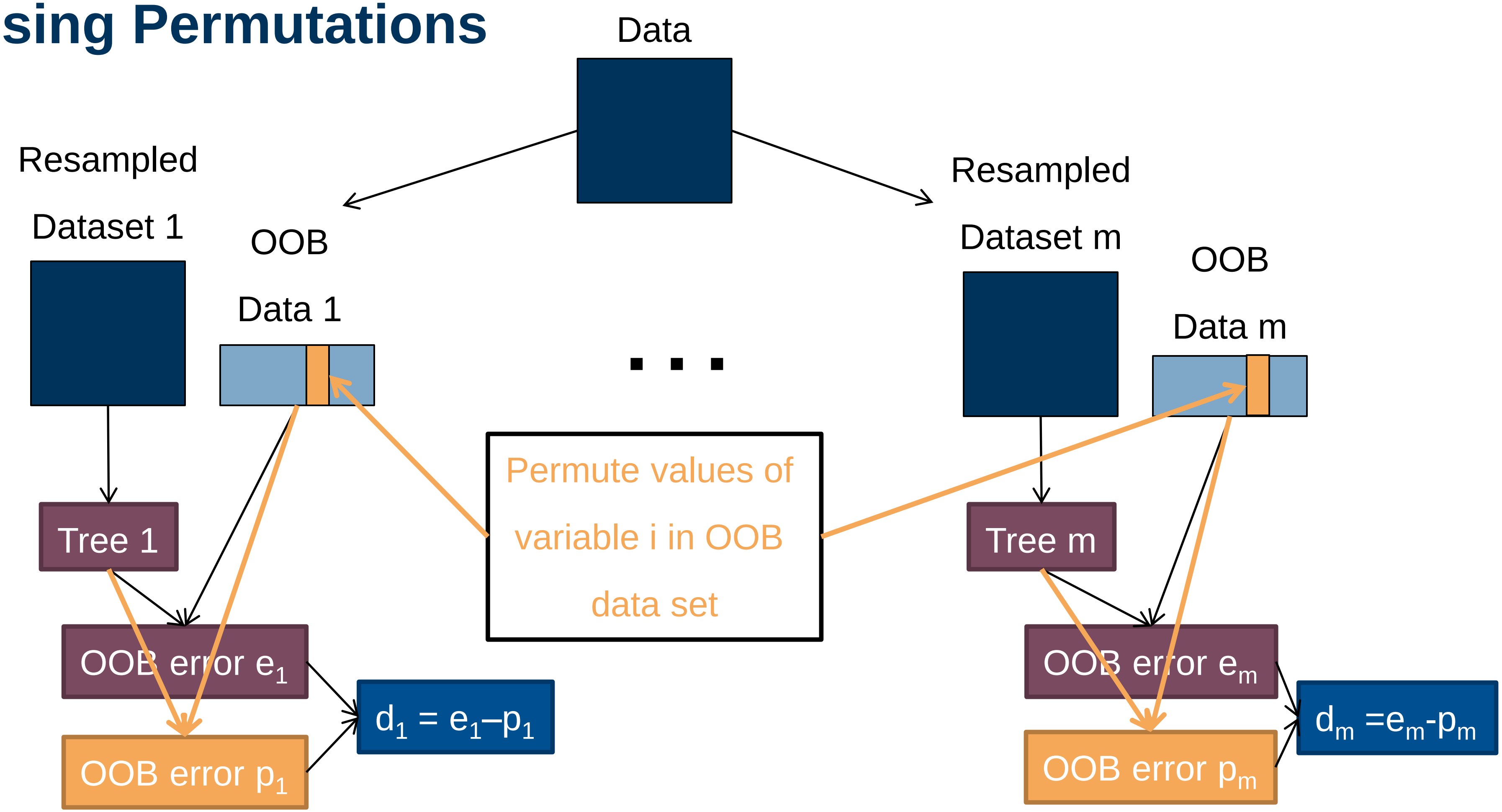
Decreases, if number of trees B increases (irrespective of ρ)

Estimating generalization error: Out-of bag (OOB) error

- Similar to leave-one-out cross-validation, but almost without any additional computational burden



Variable Importance for variable i using Permutations



$$\left.\begin{aligned} \overline{d} &= \tfrac{1}{m}\textstyle\sum_{i=1}^{m} d_i \\ s_d^2 &= \tfrac{1}{m-1}\textstyle\sum_{i=1}^{m}(d_i-\overline{d})^2 \end{aligned}\right\} \quad v_i = \frac{\overline{d}}{s_d}$$

Thank you for your attention
and
all the best for the exams!