Statistical Modelling

Subhayan Chattopadhyay

04/03/2017

Contents I

Contents II

Contents III

Contents IV

Data

Data import

Importing inbuilt backend data from CRAN repository

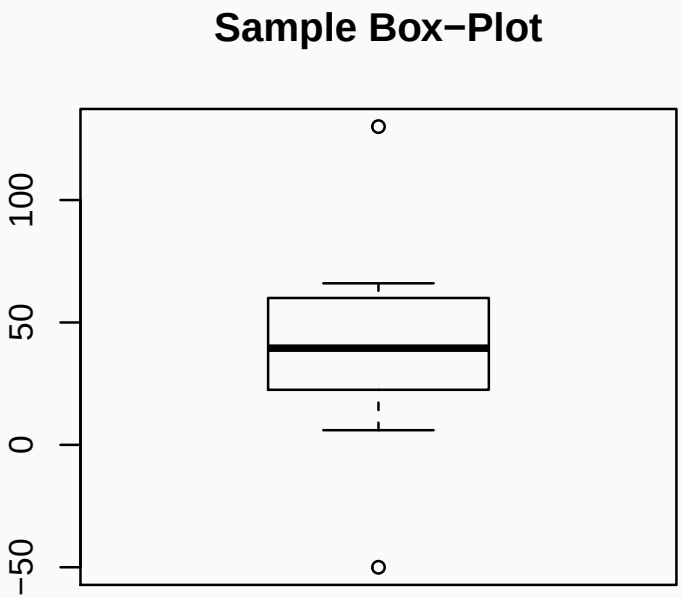
```
#data()
```

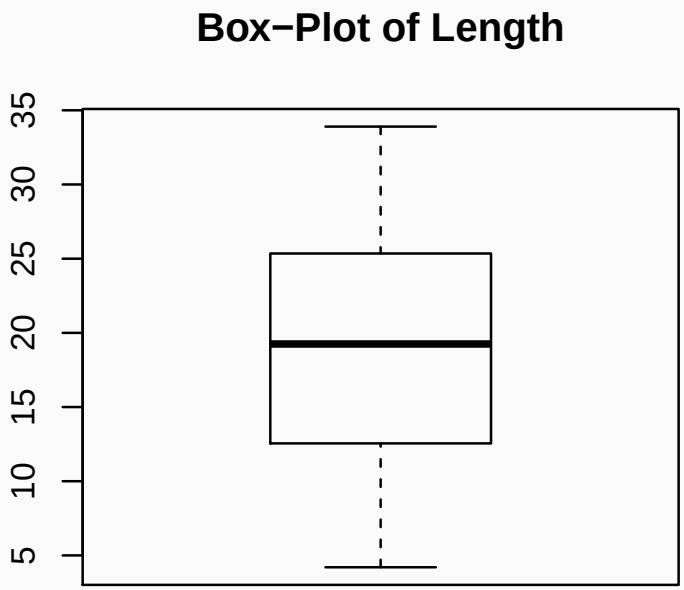
Gives list output of all datasets available in R base package "datasets"
Let's focus on such a data **ToothGrowth**

```
#??ToothGrowth
data(ToothGrowth)
#head(ToothGrowth)
#tail(Toothgrowth)
#dim(ToothGrowth)
#summary(ToothGrowth)
```

Outlier detection : Visual aid

```
par(mfrow=c(2,2))
set.seed(7544)
boxplot(c(sample(1:70,10),-50,130))
title(main ="Sample Box-Plot")
data<-ToothGrowth
boxplot(ToothGrowth$len)
title(main ="Box-Plot of Length")
```





Outlier detection : Statistical aid

Descriptive measures for outlier detection:

- df-fit
- df-beta
- df-hat
- covariance ratio
- Cook's distance
- Influential point

```
#install.packages("outliers",repos='http://cran.us.r-project.org')
require(outliers)
set.seed(013412)
outlier(rnorm(50))

## [1] -3.004867

#influence.measures(lm(mpg~.,data=mtcars)) # Multivariate and robust approach
```

Multicolinearity

- What is Degrees of freedom?

  Wikipedia:
  "In statistics, the number of degrees of freedom is the number of values in the final calculation of a statistic that are free to vary".

- What is Multicolinearity?

  Multicolinearity arises from colinearity between regressor variables.

- How Multicolinearity affects a model?

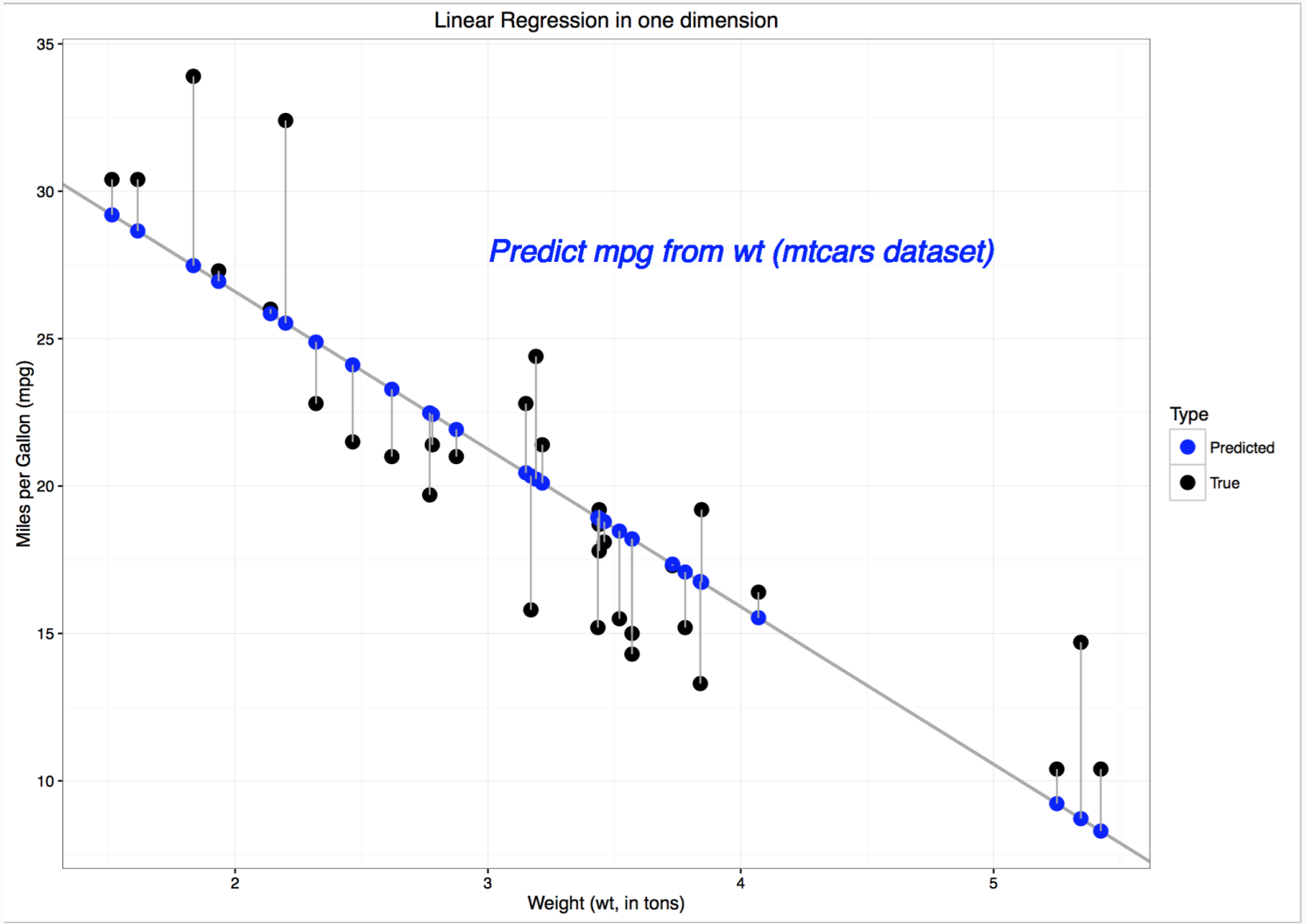
  Colinearity between variables reduces chance of explainability of variables but degrees of freedom of the model increases reducing model power.
  Colinearity between variables distabilizes regressor variance therefore distabilizes the model.

```
#cor(data)
car::vif(lm(mpg~.,data=mtcars))
```

```
##       cyl      disp        hp      drat        wt      qsec        vs
## 15.373833 21.620241  9.832037  3.374620 15.164887  7.527958  4.965873
##        am      gear      carb
##  4.648487  5.357452  7.908747
```

Supervised Learning: Linear Modeling

Linear Models (LM) : Least square optimization



Linear model and its necessary assumptions

The abstract structure of a linear model is

$$Y = bX + e$$

where Y is a vector of response/dependent/observed quantities/variables/characters
X is a matrix of independent/regressor variables
b ais a vector of the true coefficients and
e is a vector of **random errors**

```
str(ToothGrowth)
```

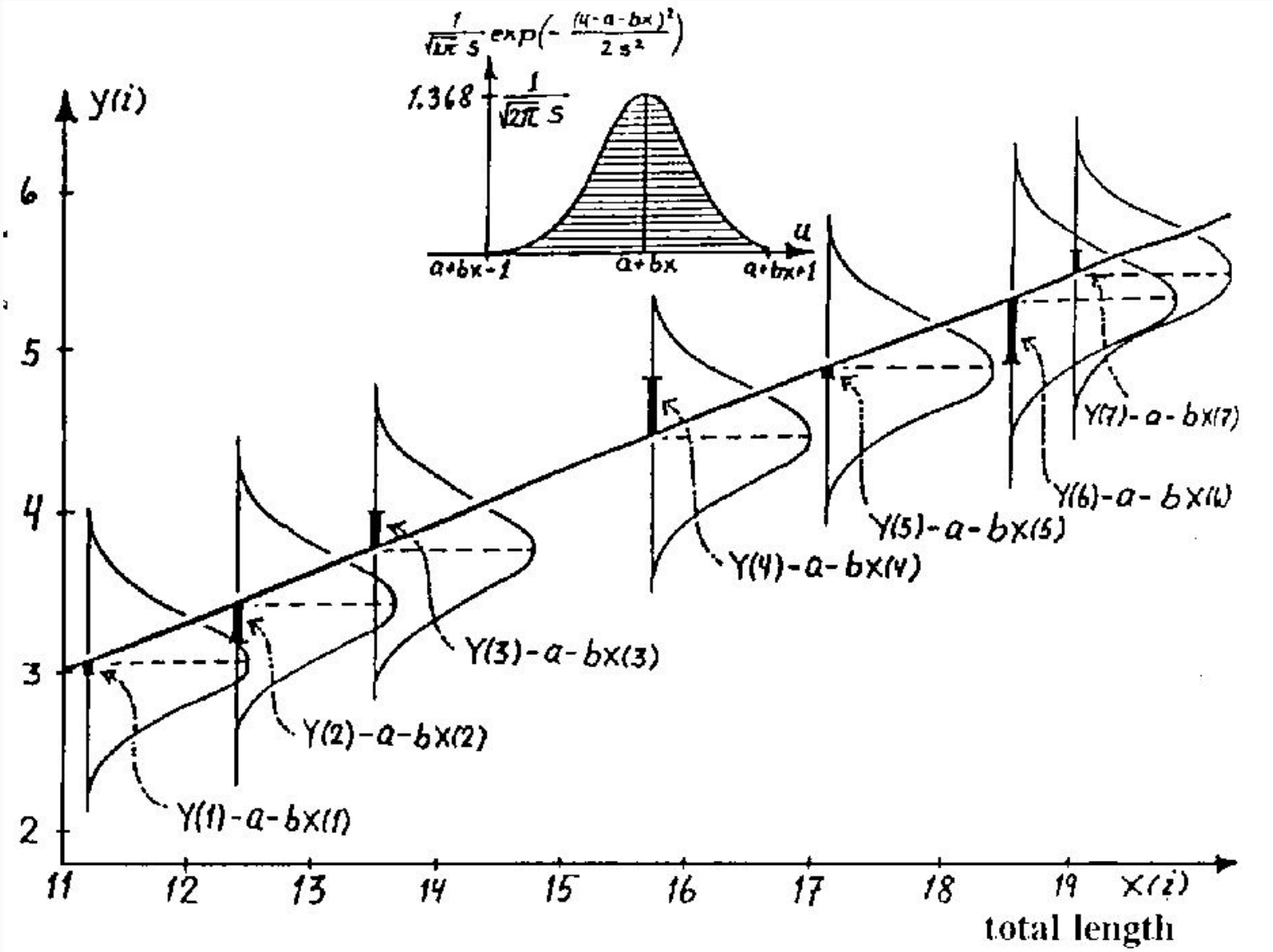
```
## 'data.frame': 60 obs. of  3 variables:
##  $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
##  $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
##  $ dose: num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
```

Assumptions of Linear models

- X are fixed and has a single source of random variance
- Y and X are linearly related
- Random errors are independently Normally distributed

with mean 0 and fixed variance

Normality assumption



Judgement of Normality

Among the very basic tools that we can use to test for normality are

- Student's t-test (statistical test of Normal distribution)
- Bartlett's test (statistical test for constant variance)
- QQ Plot (empirical test of normal distribution)

Student's T-test

```
t.test(ToothGrowth$dose)
```

```
##
##  One Sample t-test
##
## data:  ToothGrowth$dose
## t = 14.37, df = 59, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  1.004212 1.329122
## sample estimates:
## mean of x
##  1.166667
```

Bartlett's Test for variance Hidden Question.1

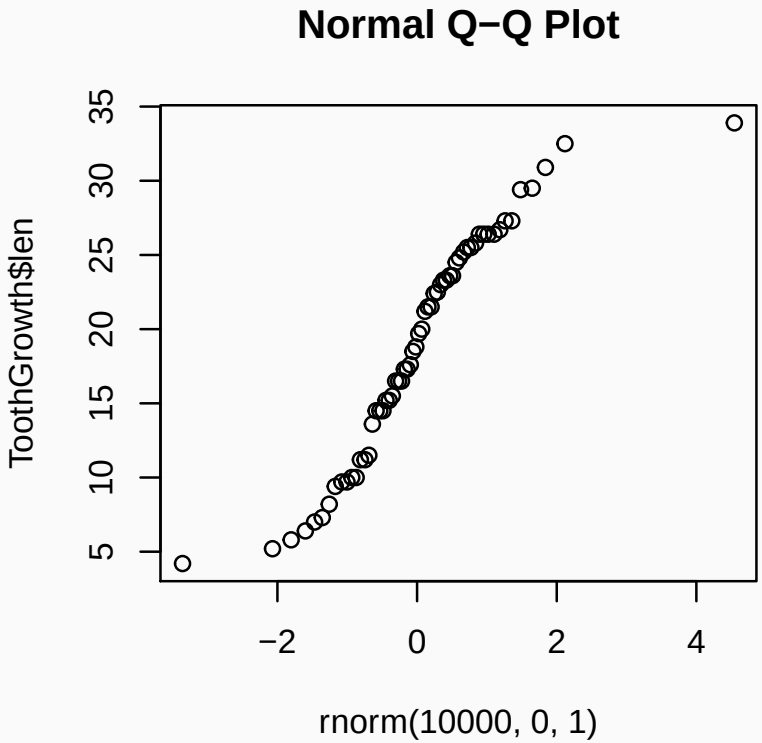
```
set.seed(011127)
bartlett.test(rnorm(60,0,1)~dose,data=ToothGrowth)

##
##  Bartlett test of homogeneity of variances
##
## data:  rnorm(60, 0, 1) by dose
## Bartlett's K-squared = 0.74276, df = 2, p-value = 0.6898

bartlett.test(rnorm(60,0,1)~scale(dose),data=ToothGrowth)

##
##  Bartlett test of homogeneity of variances
##
## data:  rnorm(60, 0, 1) by scale(dose)
## Bartlett's K-squared = 1.3661, df = 2, p-value = 0.5051
```

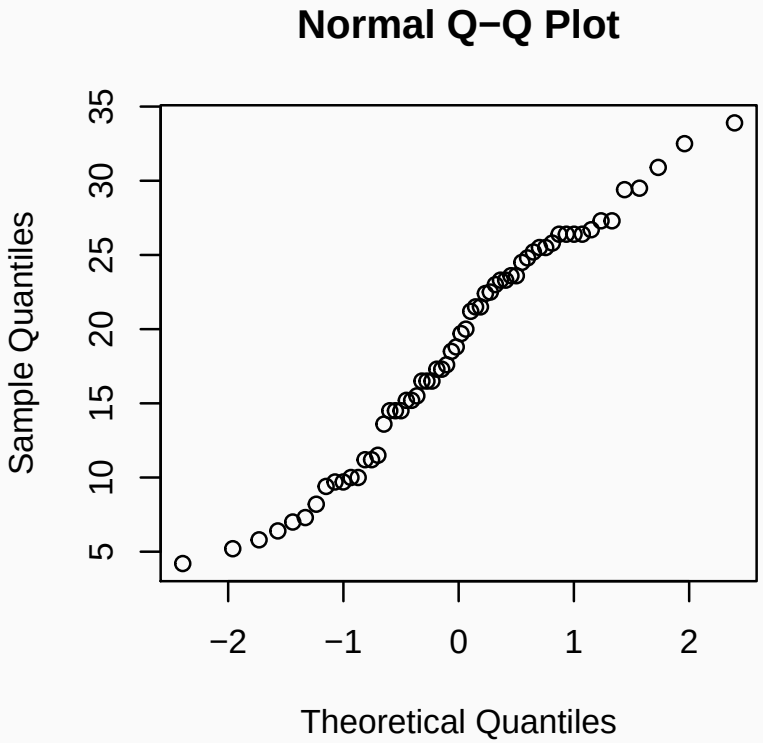
Question: Notice, your results differ than the slides, Why?

QQ Plot from frontend and R-backend

Question: Why is there a slight difference in the plots?

```
par(mfrow=c(2,2))
set.seed(405)
qqplot(rnorm(10000,0,1),ToothGrowth$len,main="Normal Q-Q Plot")
qqnorm(ToothGrowth$len)
```





Fitting LM I

`lm` function to demonstrate linear modelling or linear regression

```
model.1<-lm(len~dose,data=ToothGrowth)
model.1
```

```
##
## Call:
## lm(formula = len ~ dose, data = ToothGrowth)
##
## Coefficients:
## (Intercept)         dose
##       7.423        9.764
```

This creates a linear regression with intercept term **with intercept**.

Fitting LM II

```
summary(model.1)
```

```
##
## Call:
## lm(formula = len ~ dose, data = ToothGrowth)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4496 -2.7406 -0.7452  2.8344 10.1139
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.4225     1.2601    5.89 2.06e-07 ***
## dose          9.7636     0.9525   10.25 1.23e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.601 on 58 degrees of freedom
## Multiple R-squared:  0.6443,Adjusted R-squared:  0.6382
## F-statistic: 105.1 on 1 and 58 DF,  p-value: 1.233e-14
```

Refining the linear model

- Full linear model with all possible variables

```
model.2<-lm(len~.,data=ToothGrowth)
#summary(model.2)
```

- Full linear model with all possible variables without intercept

```
model.3<-lm(len~.-1,data=ToothGrowth)
#summary(model.3)
```

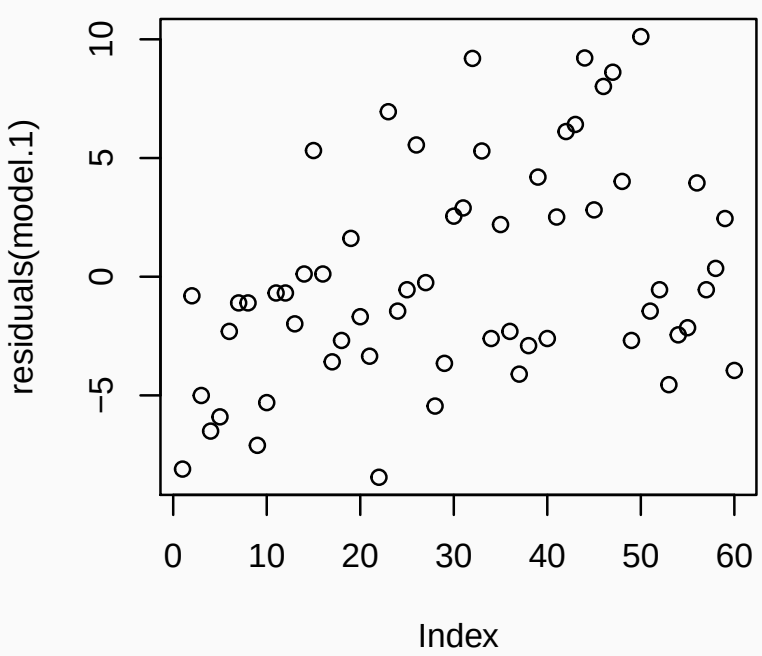
- Full Linear model while treating Supplements as character vector

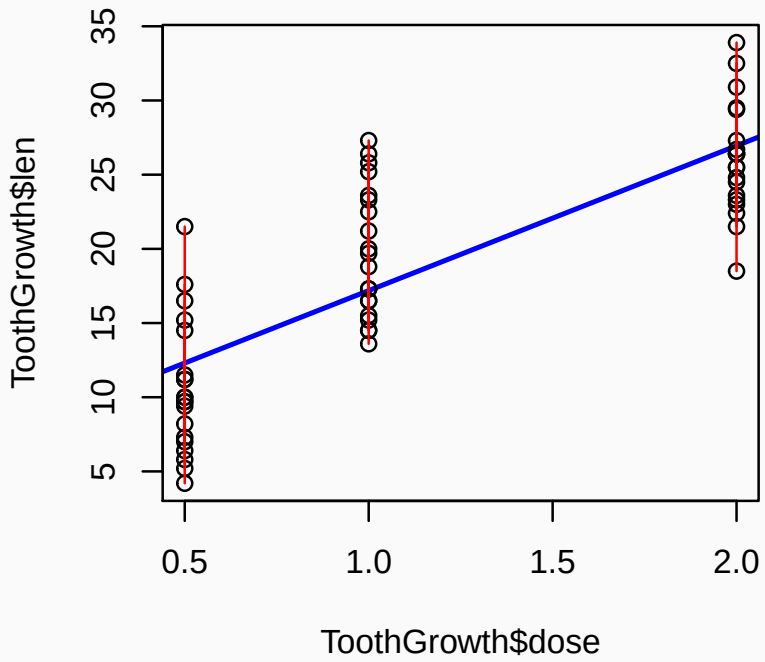
```
model.4<-lm(len~dose+as.character(supp),data=ToothGrowth)
#summary(model.4)
```

Question:
What difference between *model.2* and *model.3* do you observe?
What difference between *model.2* and *model.4* do you observe?

Residuals and Predict Hidden Question.2

```r
resid<-residuals(model.1) #Residuals produced by the model
par(mfrow=c(2,2))
pred<-predict(model.1) #Predicted values based on model.1
#plot(model.1)  #Plot model specification and diagonistics altogether
plot(residuals(model.1))
plot(ToothGrowth$dose,ToothGrowth$len,abline(model.1,lwd=2,col=4))
segments(ToothGrowth$dose, ToothGrowth$len, ToothGrowth$dose, pred, col="red")
```





EXERCISE

Instructions:

- Choose any of the data blindly from the list of datasets already available in R
- Identify a variable/quantity as your response and fit a linear model
- run diagonistics to test conformation of the data to Linear model assumptions
- write down your findings and your comments
- Write down your doubts. If you don't have a doubt, I'll assume you know everything and start asking you questions. Seriously!

- Print the following song lyrics completed in paragraphs with a single line of code in your terminal
  "When you try your best, but you don't succeed
  When you get what you want but.....
  When you feel so ....., but.....
  .....reverse."

General Linear Models (GLM)

When to go for GLM?

- Deviation from normality assumption
- Deviation from simple linearity
- Deviation in structure of the error

$$u = g^{-1}Y = L(bX)$$

Where Y,b,X are the same notations as explained im LM and $g(u) = bX$

- Link function: L demonstrates *Mathematical dependancy* between Y and bX
- L is *Variance function* of Y i.e. $V(Y)$ defined on mean $g(u)$
- **Notice** There's no error term e in the model

Here the error is distributed through the spread/variance structure of the link between Y and X

Link function

- Identity link:

$$g(u) = u$$

  Usage: Normal (Gaussian) and Gamma models because of uniform spread
- Log link:

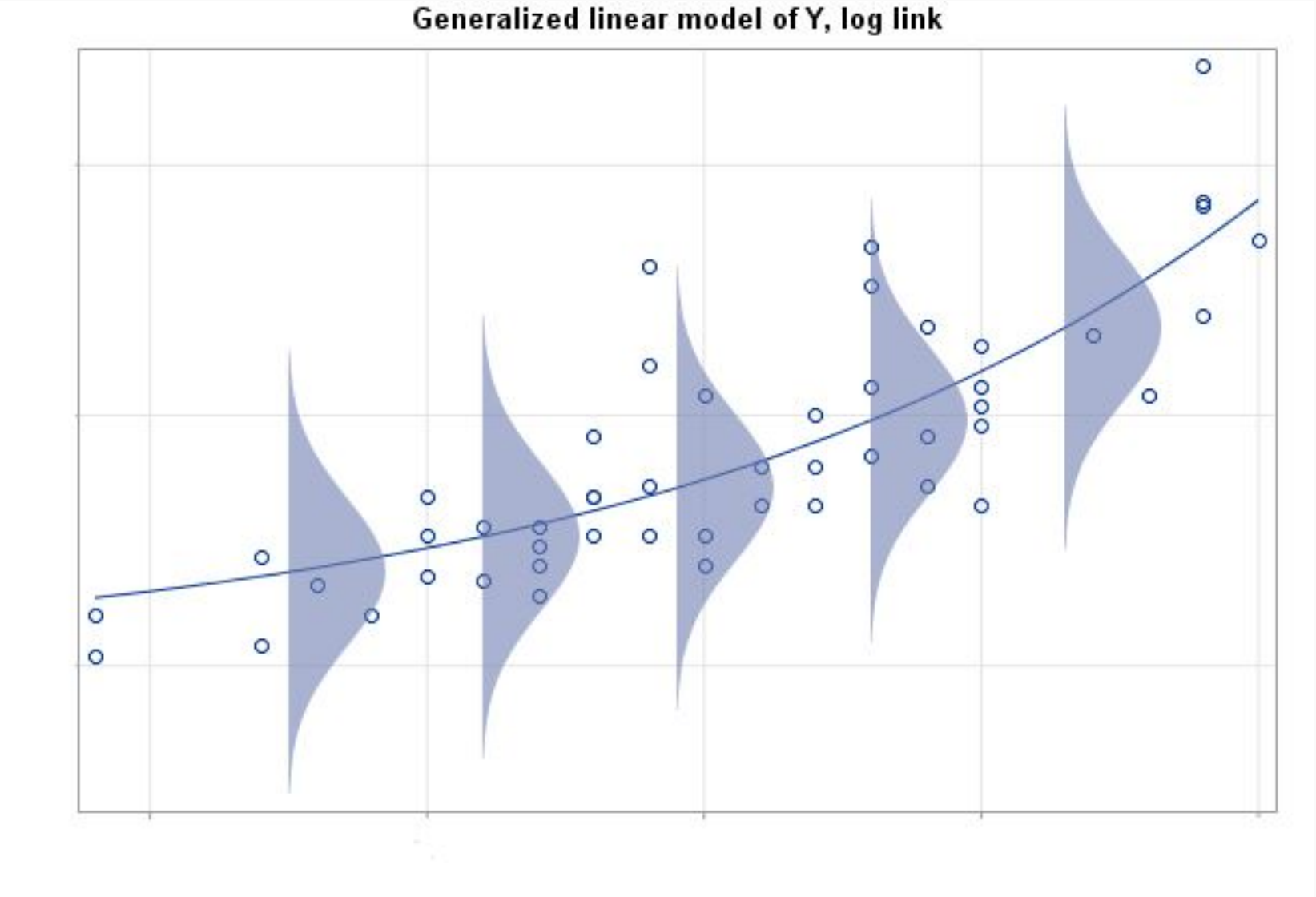
$$g(u) = log(u)$$

  Usage: Count variables (ex. exponential family - Poisson)
- Logit link:

$$g(u) = log(u/(1-u))$$

  Usage: Dichotomous variables/ observation bounded between $[0, 1]$

Variance optimization



Fitting : Identity Link

```
glm(len~dose,data=ToothGrowth,family=gaussian(link = "identity"))

##
## Call:  glm(formula = len ~ dose, family = gaussian(link = "identity"),
##     data = ToothGrowth)
##
## Coefficients:
## (Intercept)         dose
##       7.423        9.764
##
## Degrees of Freedom: 59 Total (i.e. Null);  58 Residual
## Null Deviance:     3452
## Residual Deviance: 1228  AIC: 357.4
```

Observation:
Go back to Slide No.16 and compare the model coefficients

Question: Why is this happening?

Logistic Regression

- Response variable type: Categorical variable
- Distribution of response:
    - Binomial: Two levels
    - Multinomial: More than two levels
- Logit link:

$$g(u) = log(u/(1-u))$$

$$exp(g(u)) = u/(1-u)$$

$$u = exp(g(u))/(1+exp(g(u)))$$

Remember:

$$u = g^{-1}Y = L(bX)$$

So:

$$u = exp(bX)/(1+exp(bX))$$

Interpretation of Coefficients: Odds Ratio

Odds of an event

Wikipedia: *The odds (in favor) of an event or a proposition is the ratio of the probability that the event will happen to the probability that the event will not happen. Mathematically, this is a Bernoulli trial, as it has exactly two outcomes*

$$p/(1-p)$$

where p is the probability of occurrence of the event.

Odds in context of modelling: increment in response with each unit increment in regressor
log odds Ratio:

$$log(odds(x+1)/odds(x)) = b$$

$$odds(x) = u/(1-u)$$

Fitting: Logit link

```
require(MASS)
data(menarche)
#str(menarche)
#summary(menarche)
model.5<-glm(cbind(Menarche,Total-Menarche)~Age,family=binomial,data=menarche)
#summary(model.5)
#exp(coef(model.5)) #Gives Odds Ratios
#confint(model.5) #Gives confidence intervals
```

Remember:

- Logit link is binomial while regressing over one variable
- It's multinomial while regressing over a matrix of variables

Fitting: Exponential link (Poisson Regression) Hidden Qestion.3

Poisson is probability model on *count observation*

```
data<-warpbreaks
#??(warpbreakes)
#str(warpbreaks)
#summary(warpbreakes)
model.6<-glm(breaks ~ wool+tension, data = warpbreaks, family = poisson)
#summary(model.6)
#exp(coef(model.6)) #Gives Relative Risks
```

Model Optimization

Visualization:

```
#plot(model.5)
```

Descriptive tatistics:

- Residuals
- Deviance / Residual Deviance
- Akaike's Information Criterion (AIC)
- Bayesian Information Criterion (BIC)

```
#all similar diagonostics as done in LM
deviance(model.5)
```

```
## [1] 26.70345
```

```
AIC(model.5)
```

```
## [1] 114.7553
```

```
BIC(model.5)
```

```
## [1] 117.193
```

Exercise

Instructions:

- Choose the "`HairEyeColor`" data
- Fit a general linear model and explain why you chose this model
- Run diagonistics on your model and WRITE DOWN your report
- Remember what I said about doubts? I need at least one doubt from you
- Only If you are done, run these codes:

```
#install.packages('fun')
#library(fun)
if (.Platform$OS.type == "windows") x11() else x11(type = "Xlib")
#mine_sweeper()
```

Hiererchical Linear Models (HLM)

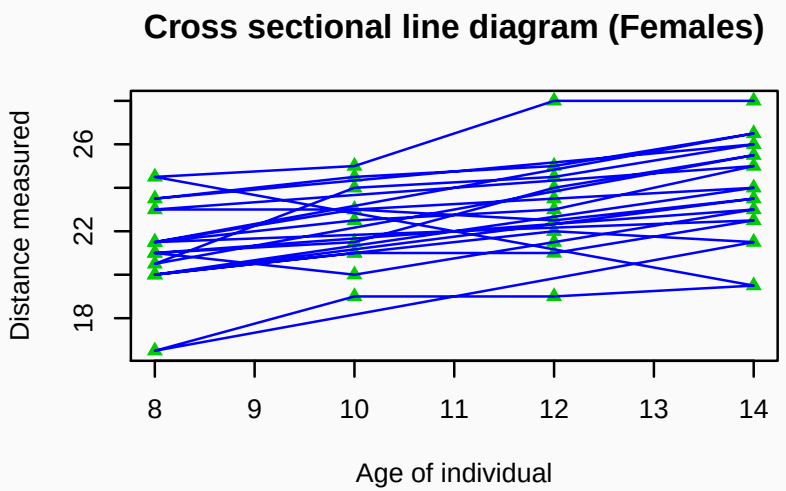
When to use Hiererchical model?
Variance structure of the regressors are individually different.
We'll take `Orthodont` data from `nlme` package

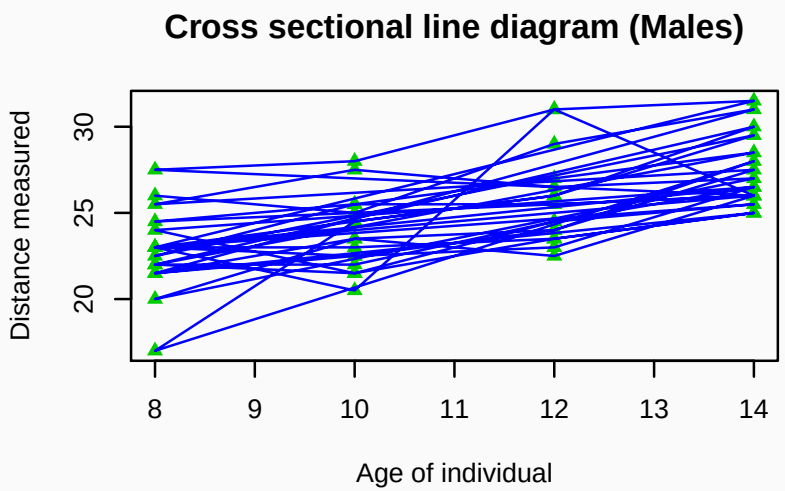
```
library(lme4)

## Loading required package:  Matrix

library(nlme)

##
## Attaching package:  'nlme'

## The following object is masked from 'package:lme4':
##
##     lmList

attach(Orthodont)
```

Exploratory Analysis

What do you observe?

```
par(mfrow=c(3,2))
data<-subset(Orthodont,Orthodont$Sex=="Female")
plot(data$distance~data$age,pch=17,col=11,xlab="Age of individual"
, ylab="Distance measured",main="Cross sectional line diagram (Females)")
lines(data$distance~data$age, col =12, type="l",lwd=1,lty=1)

data<-subset(Orthodont,Orthodont$Sex=="Male")
plot(data$distance~data$age,pch=17,col=11,xlab="Age of individual"
, ylab="Distance measured",main="Cross sectional line diagram (Males)")
lines(data$distance~data$age, col =12, type="l",lwd=1,lty=1)
```





Fitting

```
model.7<-lmer(distance~age+Sex+(1|Subject),data=Orthodont)
#summary(model.7)
```

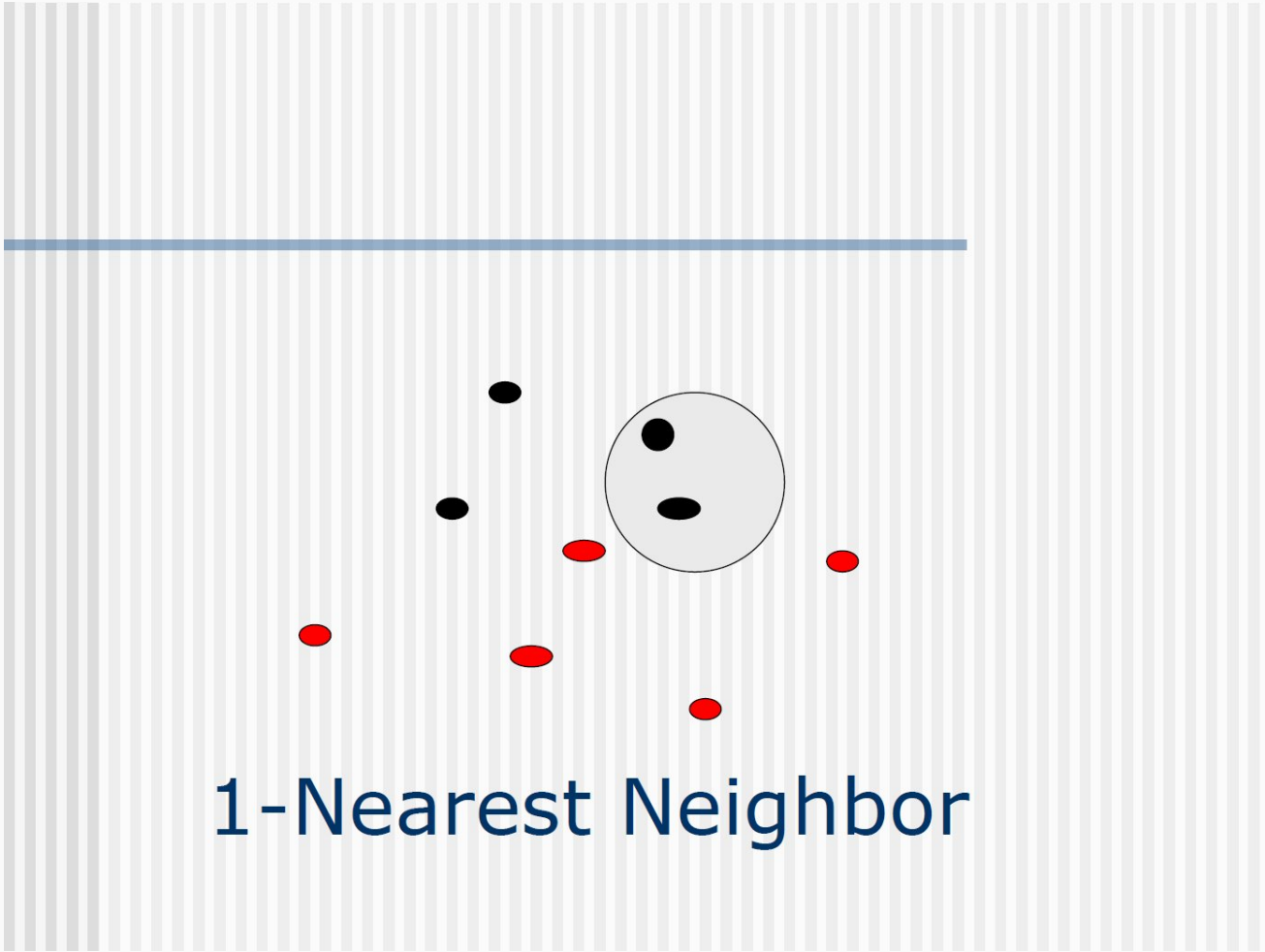
```
Linear mixed model fit by REML ['lmerMod']
Formula: distance ~ age + Sex + (1 | Subject)
   Data: Orthodont

REML criterion at convergence: 437.5

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.7489 -0.5503 -0.0252  0.4534  3.6575

Random effects:
 Groups   Name        Variance Std.Dev.
 Subject  (Intercept) 3.267    1.807
 Residual             2.049    1.432
Number of obs: 108, groups:  Subject, 27

Fixed effects:
            Estimate Std. Error t value
(Intercept) 17.70671    0.83392  21.233
age          0.66019    0.06161  10.716
SexFemale   -2.32102    0.76142  -3.048

Correlation of Fixed Effects:
          (Intr) age
age       -0.813
SexFemale -0.372  0.000
```

EXERCISE

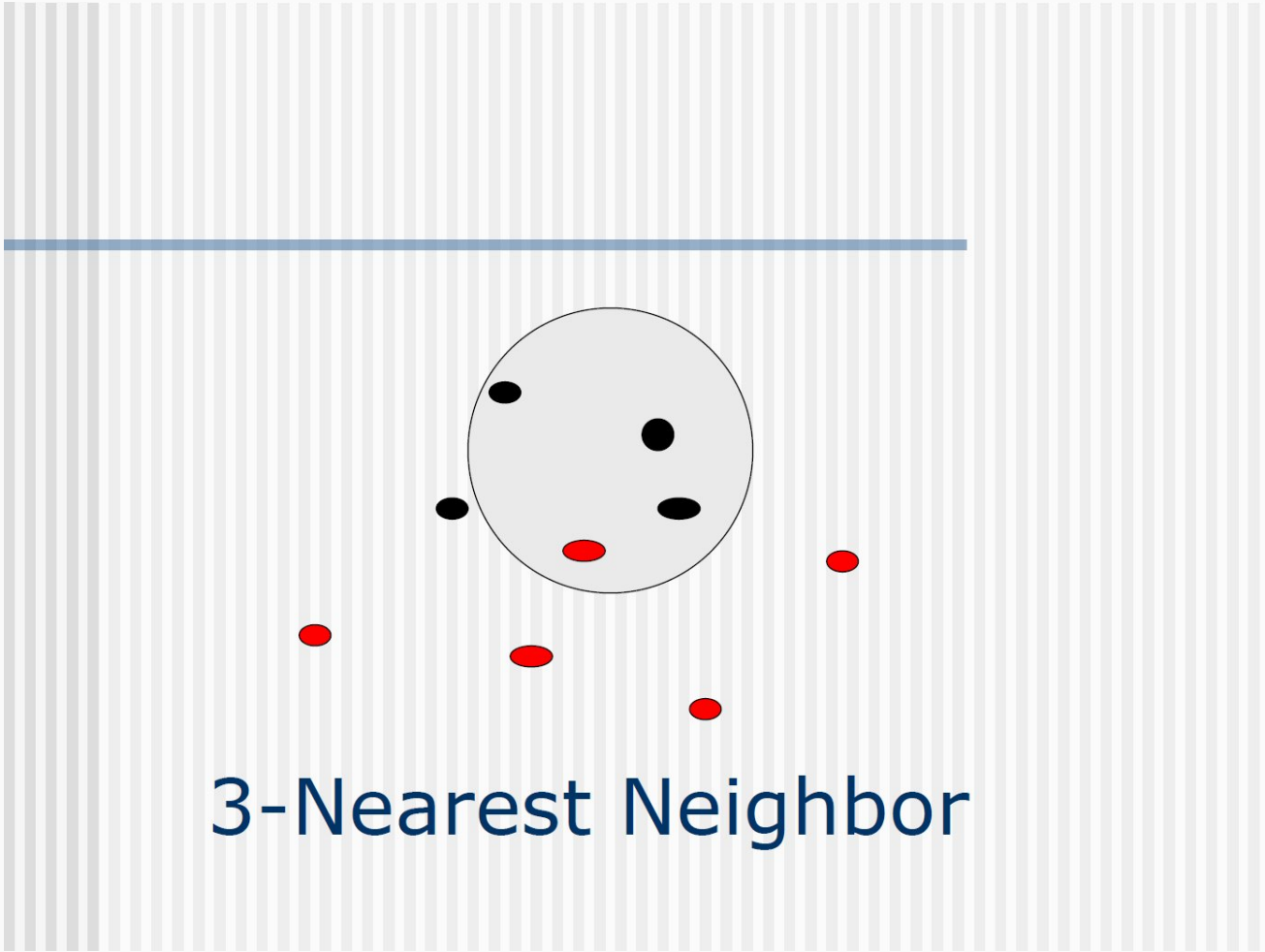
Note down what do you think is different in this summary

Supervised Learning: Lazy Learning

k-Nearest Neighbour Classification



k-Nearest Neighbour Classification



k-Nearest Neighbour Classification

```
require(class)

## Loading required package:  class

data<-Orthodont
smp_size <- floor(0.5 * nrow(data))
set.seed(123)
train_prop<-sample(seq_len(nrow(data)), size = smp_size)
train<-data[train_prop, ]
test<-data[-train_prop, ]
train1<-train[,-c(3,4)]
test1<-test[,-c(3,4)]
train1$sex<-with(train,ifelse(Sex=="Male",1,0))
test1$sex<-with(test,ifelse(Sex=="Male",1,0))
knn.1<-knn(train1[,-3],test1[-3],train1$sex,k=2)
```

Supervised Learning: Decision Trees

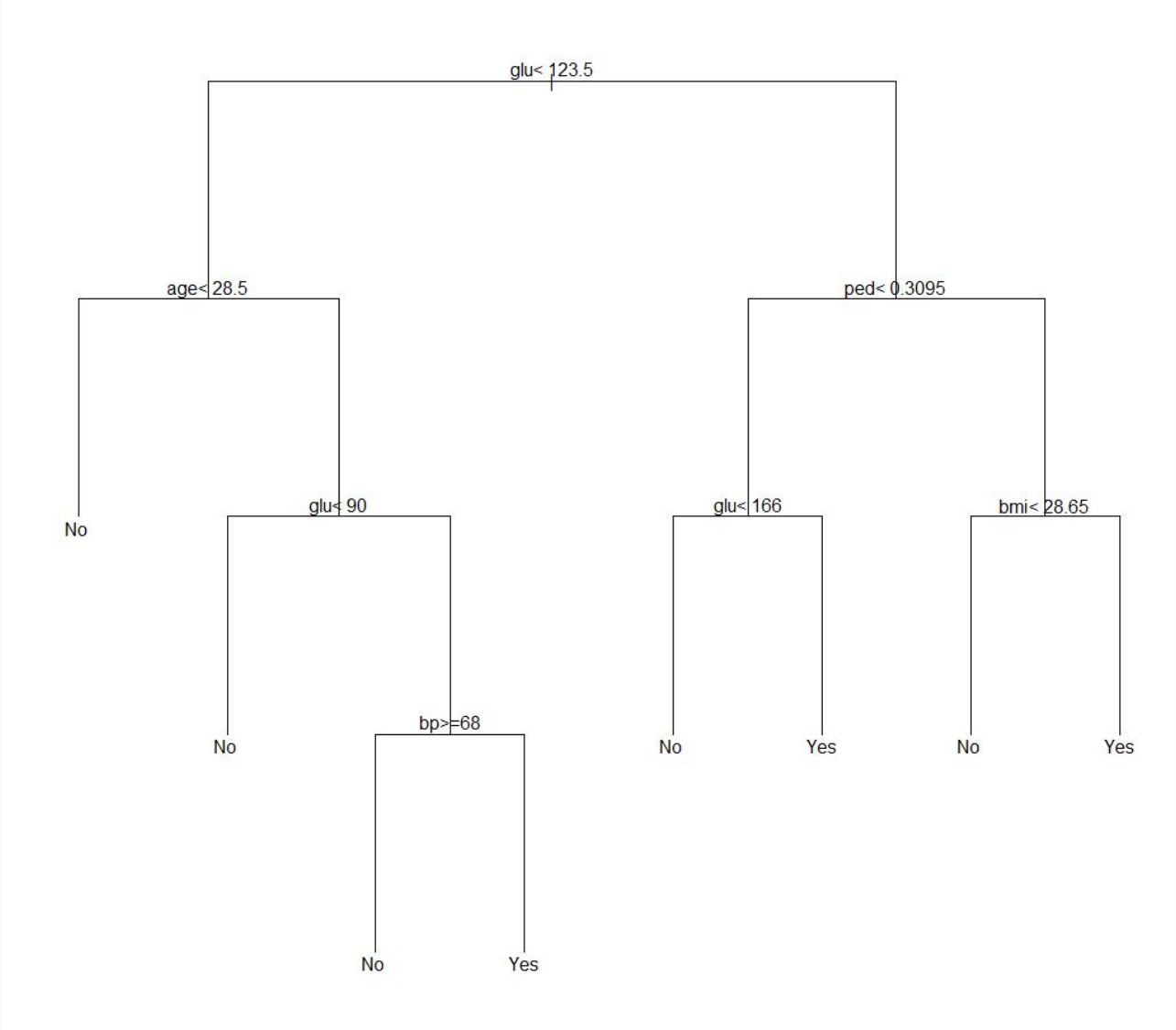
Recursive Partitioning

Recursive partitioning is built on a very basic mathematical idea:

- Association statics between dependant variable Y and independent variable X
- Subset Y with closest association or stop subsetting
- repeat the procedure in each of the subsequent subgroups created

```
require(MASS)
require(rpart)

## Loading required package:  rpart

data<-Pima.tr
set.seed(20161125)
model.8 <- rpart(type ~ ., data = data)
#plot(model.8,uniform=T)
#text(model.8)
```

Recursive Partitioning



Bagging

Bagging is introduced to reduce dependency on single tree from `Recursive Partitioning` and chance finding produced there of. It uses **B**ootstrap **Agg**regation by employing re-sampling technique and combine multiple trees.
Mathematics behind:

- Draw n bootstrap samples
- Learn trees with each of the n samples
- Aggregate the tree node outcomes with majority voting

Random Forest

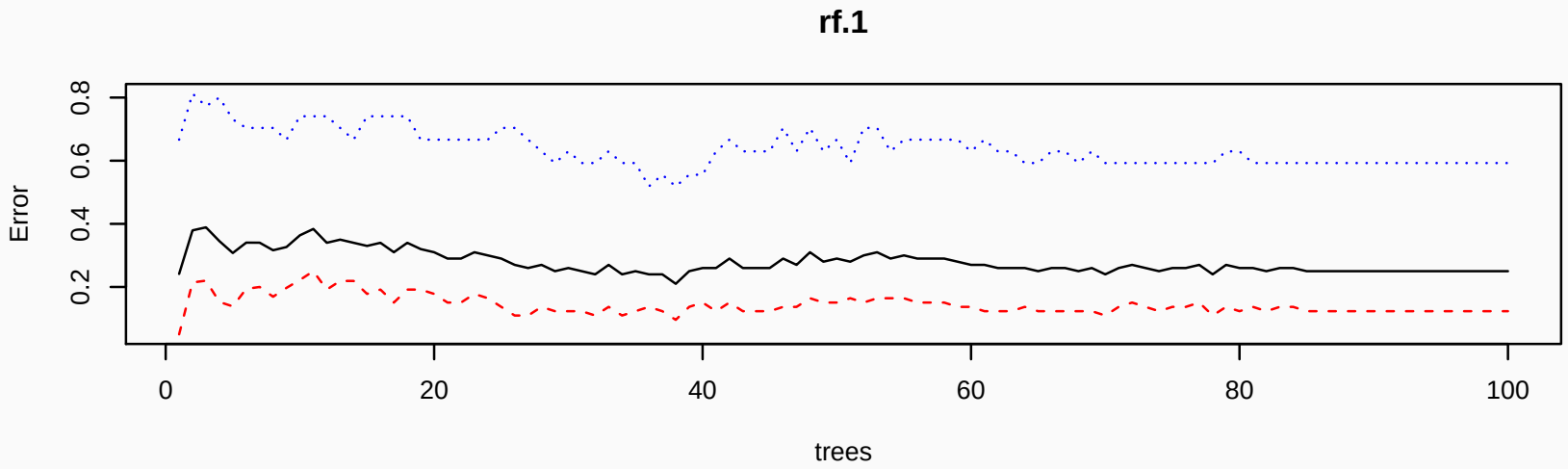
- Random Forest comes from the coined term "Random Decision Forest"
- Random Forest(s) is a ensemble classifier consisting of many decisioin trees
- It is a combination of `bagging` and `random selection`

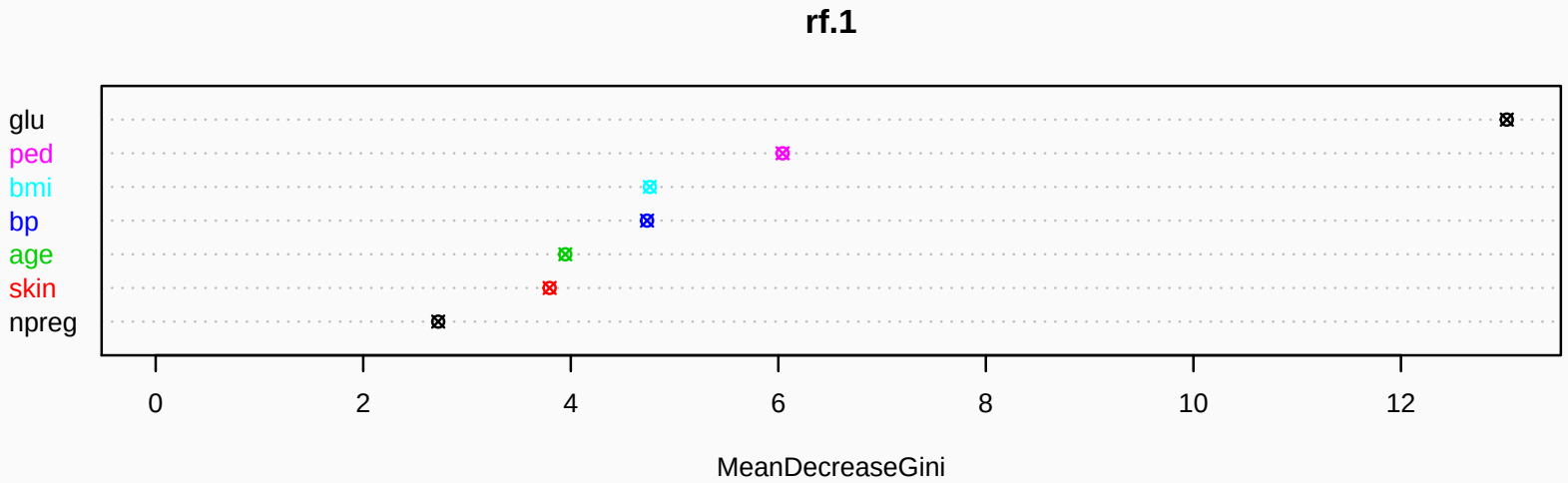
Random Forest

```
library(randomForest)

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package:  'randomForest'

## The following object is masked from 'package:outliers':
##
##    outlier

smp_size <- floor(0.5 * nrow(Pima.tr))
data<-Pima.tr
smp_size <- floor(0.5 * nrow(data))
set.seed(127)
train_prop<-sample(seq_len(nrow(data)), size = smp_size)
train<-data[train_prop, ]
test<-data[-train_prop, ]
rf.1<-randomForest(type~.,data=train,mtry=4,ntree=100,replace=TRUE)
pred_rf.1<- predict(rf.1, newdata=test)
```

Random Forest

```
par(mfrow=c(3,1))
plot(rf.1,col=c(1,2,4))
varImpPlot(rf.1,pch=13,col=seq(2:7))
```

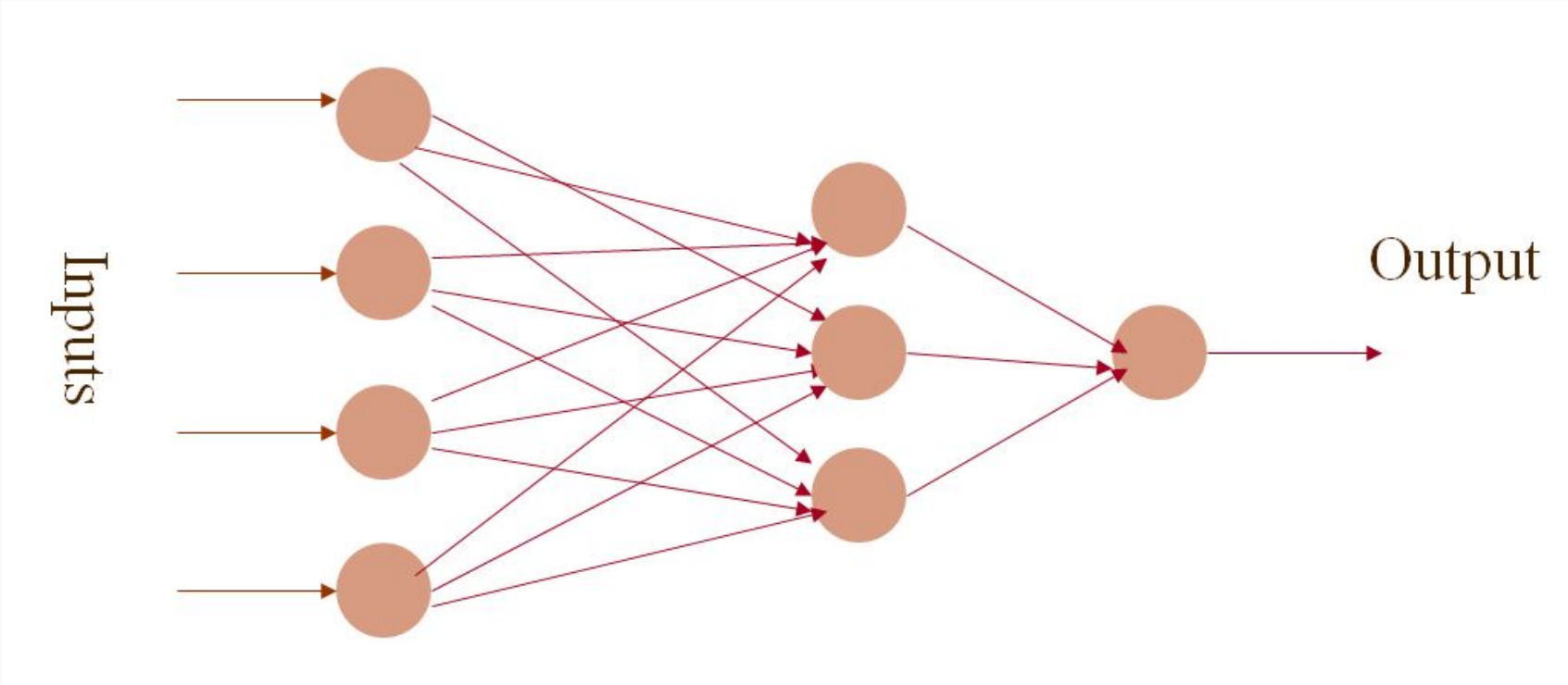




Supervised Learning: Eager Learning

Artificial Neural network

An artificial neural network is composed of many artificial neurons that are linked together according to a specific network architecture. The objective of the neural network is to transform the inputs into meaningful outputs.



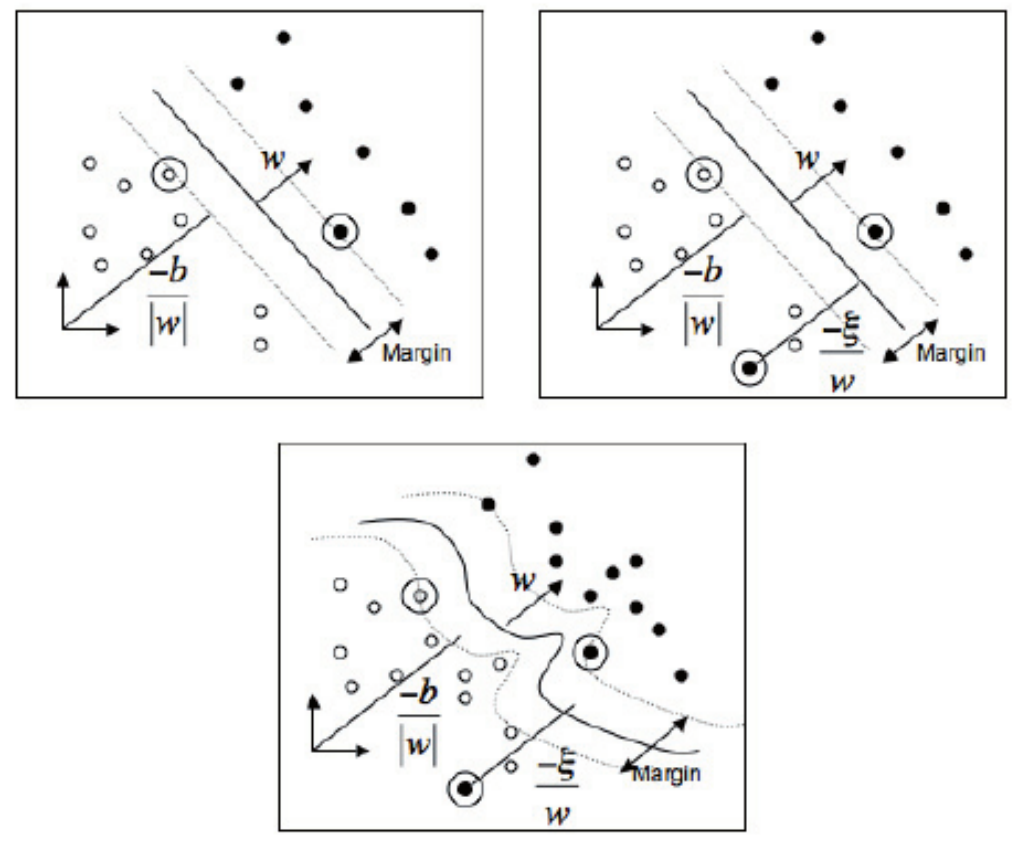
Artificial Neural network

```
require(nnet)

## Loading required package:  nnet

set.seed(7544)
nnet.1<-nnet(type~.,data=train,size=5)

## # weights:  46
## initial  value 67.797539
## iter  10 value 54.777267
## iter  20 value 50.342170
## iter  30 value 49.604673
## iter  40 value 48.885616
## iter  50 value 48.078264
## iter  60 value 48.063161
## final  value 48.063145
## converged

pred<-predict(nnet.1,test,type="class")
#table((noquote(pred)),test["type"])
```

Supervised Learning: Support Vector machine (SVM)

Support Vector machine (SVM)

SVM is an extension of Linear classifier to a multidimensional non-linear classifier. Observe:



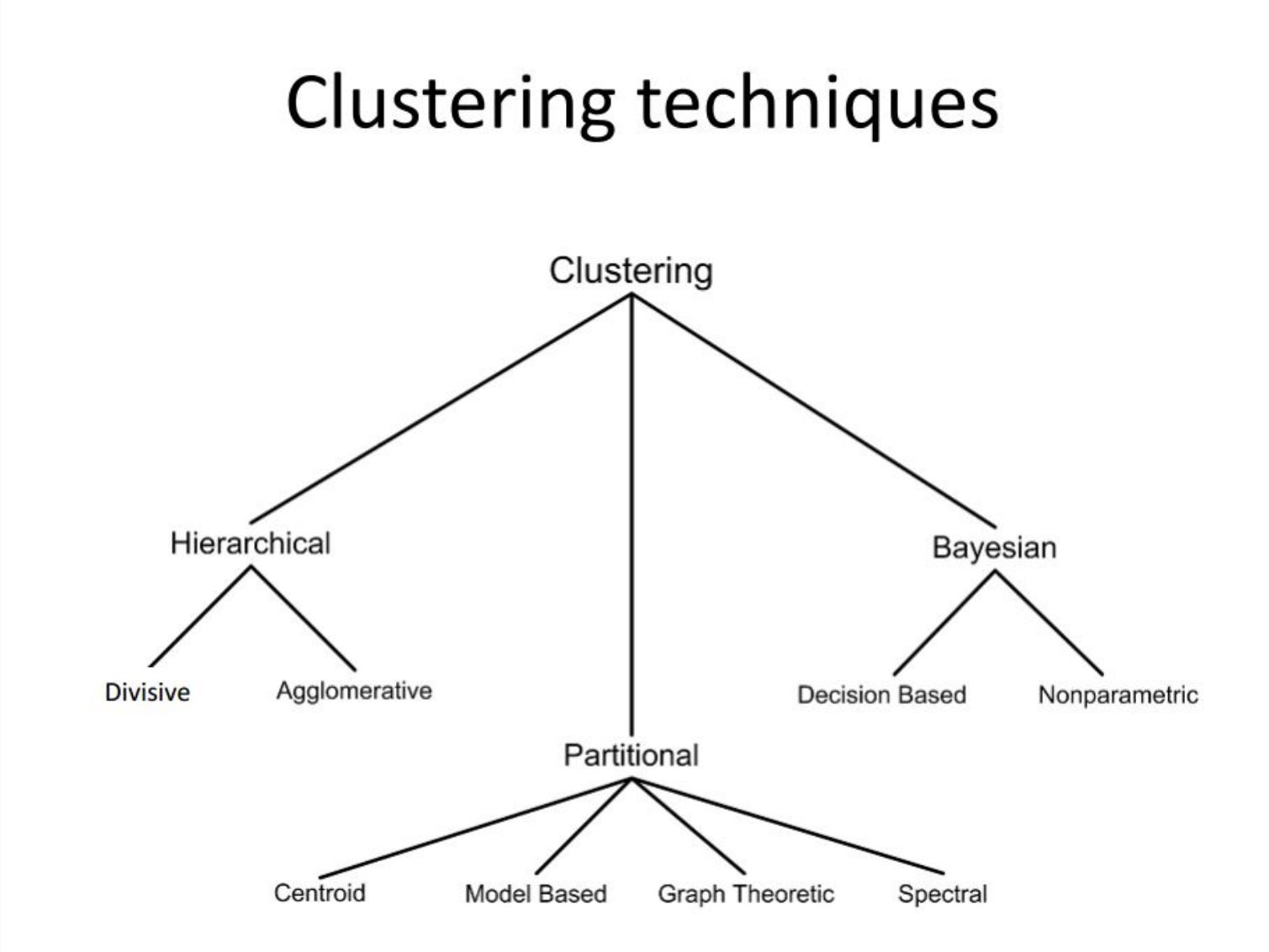
Support Vector machine (SVM)

```
require(e1071)

## Loading required package:  e1071

set.seed(7544)
tune.svm(type~.,data=test,gamma=2^(-7:-2), cost = 2^(-1:1))

##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##      gamma cost
##  0.0078125    1
##
## - best performance: 0.23
```

Support Vector machine (SVM) Hidden Question.4

```
set.seed(7544)
svm.1<-svm(type~.,data=test,gamma = 2^-7, cost = 1)
pred<-predict(svm.1,train,type="class")
table((noquote(pred)),train[,"type"])

##
##        No Yes
##    No  58  10
##    Yes 15  17

table(test$type)

##
##  No Yes
##  59  41
```

Unsupervised Learning: Clustering


Clustering techniques
Clustering
Hierarchical
Bayesian
Divisive
Agglomerative
Decision Based
Nonparametric
Partitional
Centroid
Model Based
Graph Theoretic
Spectral

K-means Clustering

Wikipedia: "k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster."

- K-means is a **partitional clustering** algorithm
- The k-means algorithm partitions the given data into `k` clusters

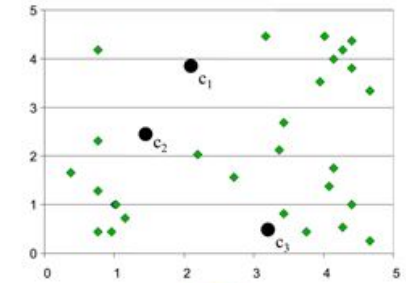
Workflow:

- Choose k (random) data points (seeds) to be the initial centroids, cluster centers
- Assign each data point to the closest centroid
- Re-compute the centroids using the current cluster memberships
- Repeat re-comutation until convergence criterion is met

K-means Clustering

K-means clustering example: step 1

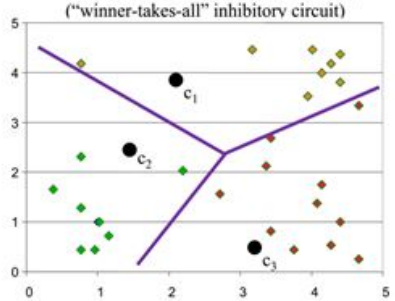
Randomly initialize the cluster centers (synaptic weights)



1

K-means clustering example – step 2

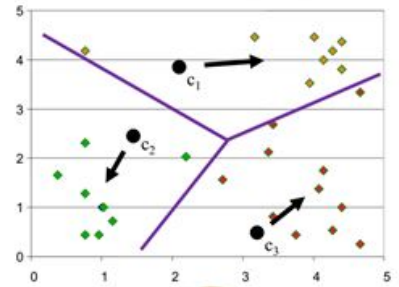
Determine cluster membership for each input ("winner-takes-all" inhibitory circuit)



2

K-means clustering example – step 3

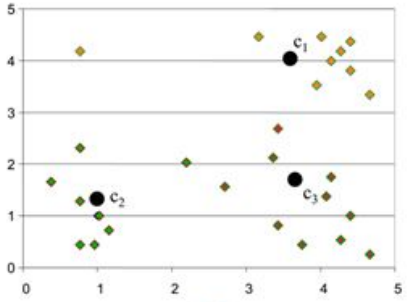
Re-estimate cluster centers (adapt synaptic weights)



3

K-means clustering example

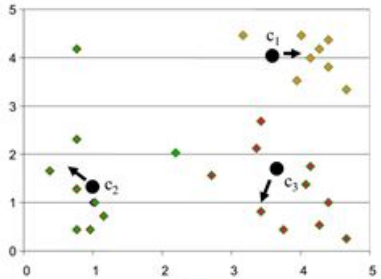
Result of first iteration



4

K-means clustering example

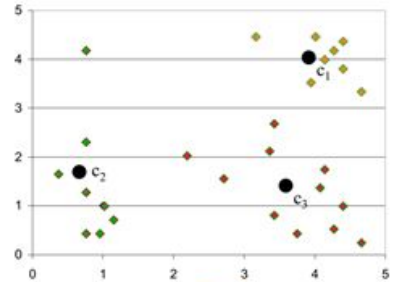
Second iteration



5

K-means clustering example

Result of second iteration

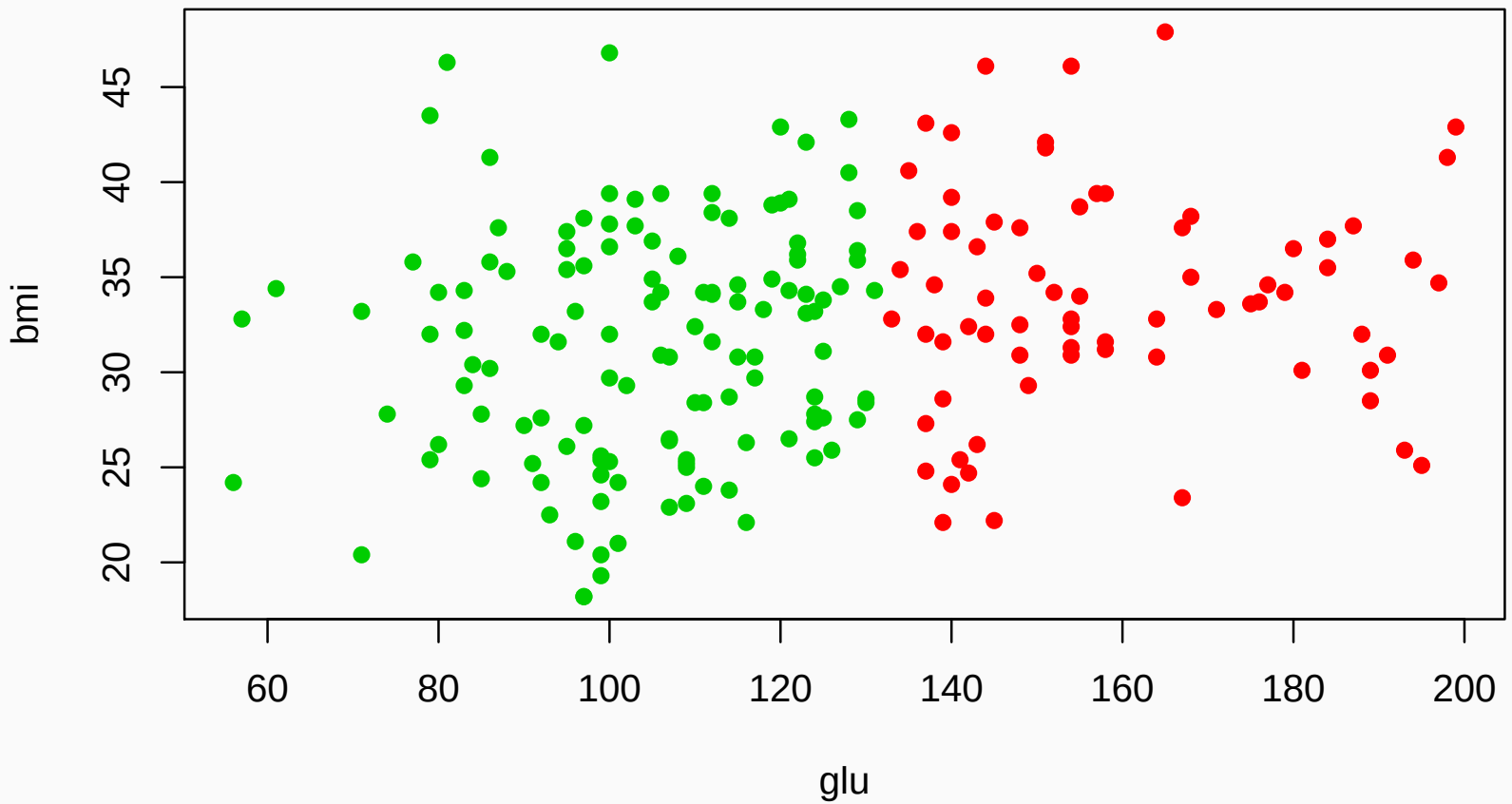


6

K-means Clustering: Fitting

We'll take two most classifying variables from The `Pima.tr` data.

```
km1 = kmeans(Pima.tr[,c(2,5)], 2, nstart=100)
plot(Pima.tr[,c(2,5)], col =(km1$cluster +1), pch=16, cex=1)
```



Exercise

Instructions

- Choose any `Three` variables from the `Pima.tr` data
- Fit a clustering with three clusters
- Plot the clusters
- Note down your observations

Hierarchical Agglomerative Clustering

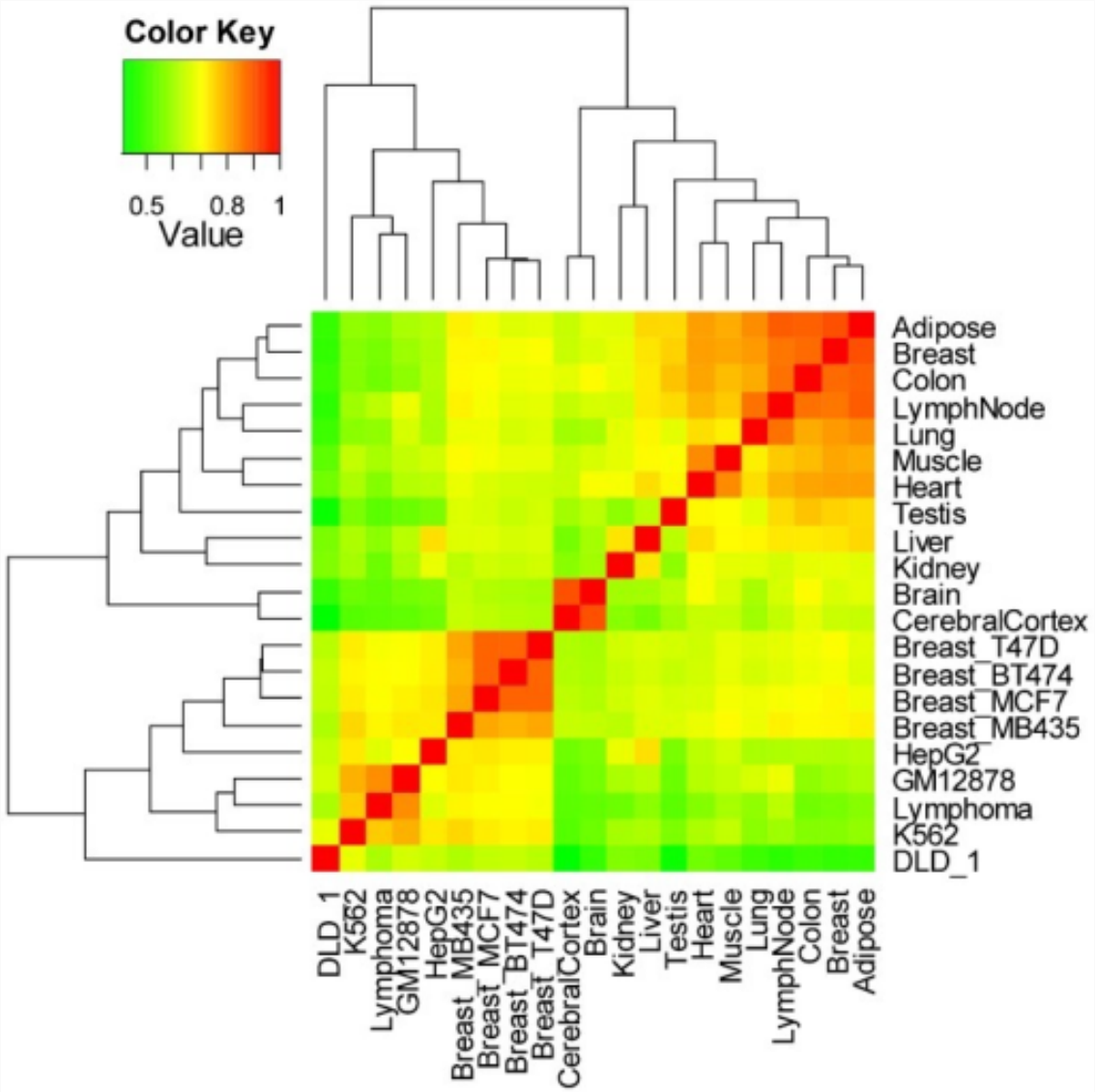
Agglomerative clustering is bottom-up approach of tree-like classification algorithm.

- Start with each document being a single cluster
- Eventually all documents belong to the same cluster

Workflow:

- Assumes a similarity function for determining the similarity of two instances
- Starts with all instances in a separate cluster and then repeatedly joins the two clusters that are most similar until there is only one cluster
- The history of merging forms a binary tree or hierarchy

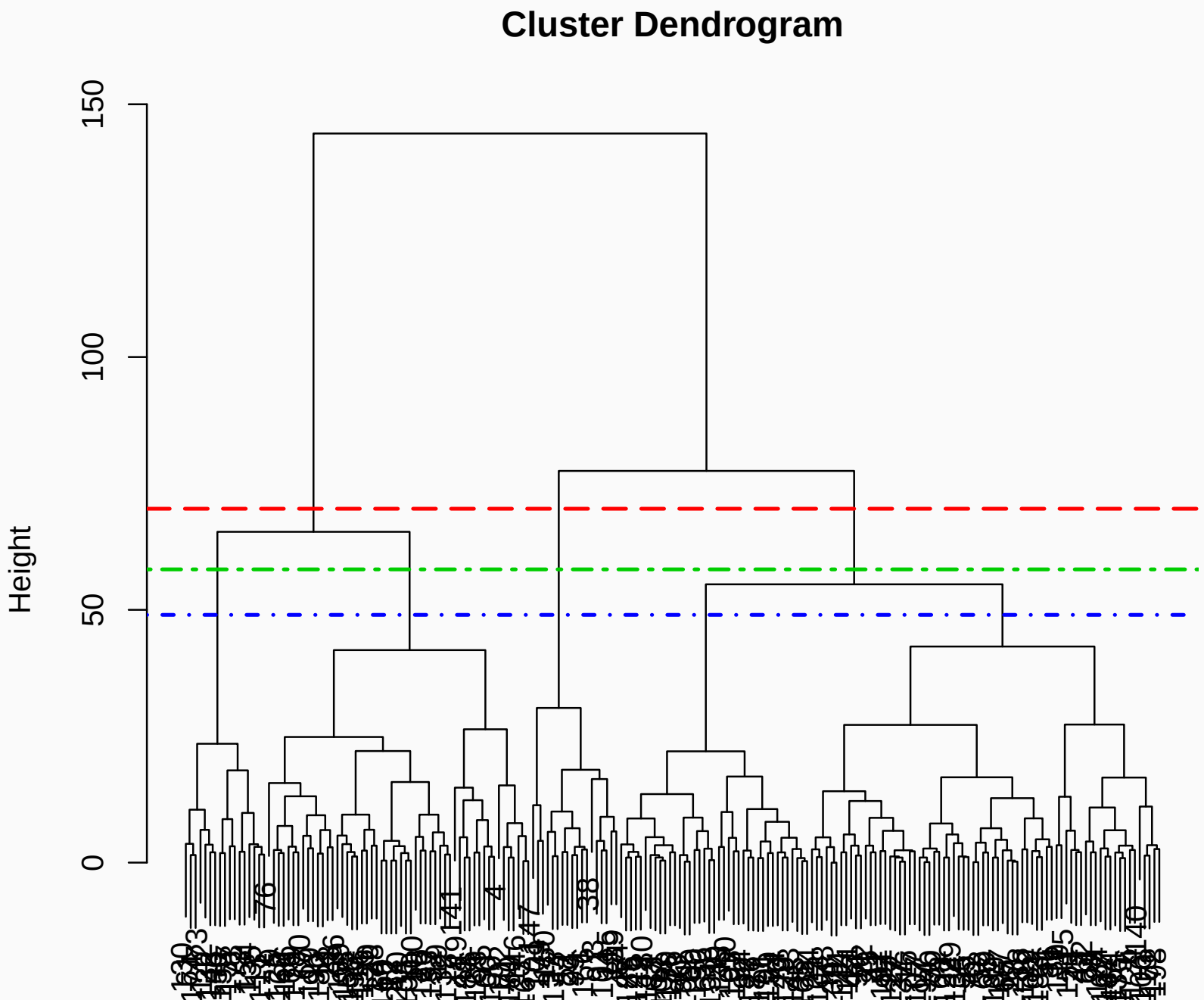
Hierarchical Clustering: Example of Dendogram and cluster matrix



Hierarchical Clustering: Fitting I

```
require(fastcluster)

## Loading required package:  fastcluster

##
## Attaching package:  'fastcluster'

## The following object is masked from 'package:stats':
##
##    hclust

clust1<-hclust(dist(Pima.tr[,c(2,5)]))
plot(clust1)
abline(58,0,col=3,lwd=2,lty=6)
abline(70,0,col=2,lwd=2,lty=5)
abline(49,0,col=4,lwd=2,lty=4)
```

Hierarchical Clustering: Fitting II



Exercise

Instructions

- Choose any `Four` variables from the `Pima.tr` data
- Fit an agglomerative clustering
- Plot the cluster dendogram
- Note down significant levels to get two and three clusters

Survival Analysis

Kaplan-Meyer statistics I

Let t_1, t_{12}, t_3,... denote the actual times of death of the n individuals in the cohort. Also let d_1, d_2, d_3,... denote the number of deaths that occur at each of these times, and let n_1, n_2, n_3,... be the corresponding number of patients remaining in the cohort. Note that $n_2 = n_1 - d_1$, $n_3 = n_2 - d_2$, etc.

Then, loosely speaking, $S(t_2) = P(T > t_2) =$ "Probability of surviving beyond time t_2" depends conditionally on $S(t_1) = P(T > t_1) =$ "Probability of surviving beyond time t_1." Likewise, $S(t_3) = P(T > t_3) =$ "Probability of surviving beyond time t_3" depends conditionally on $S(t_2) = P(T > t_2) =$ "Probability of surviving beyond time t_2," etc. By using this recursive idea, we can iteratively build a numerical estimate $S_E(t)$ of the true survival function $S(t)$

Kaplan-Meyer statistics II

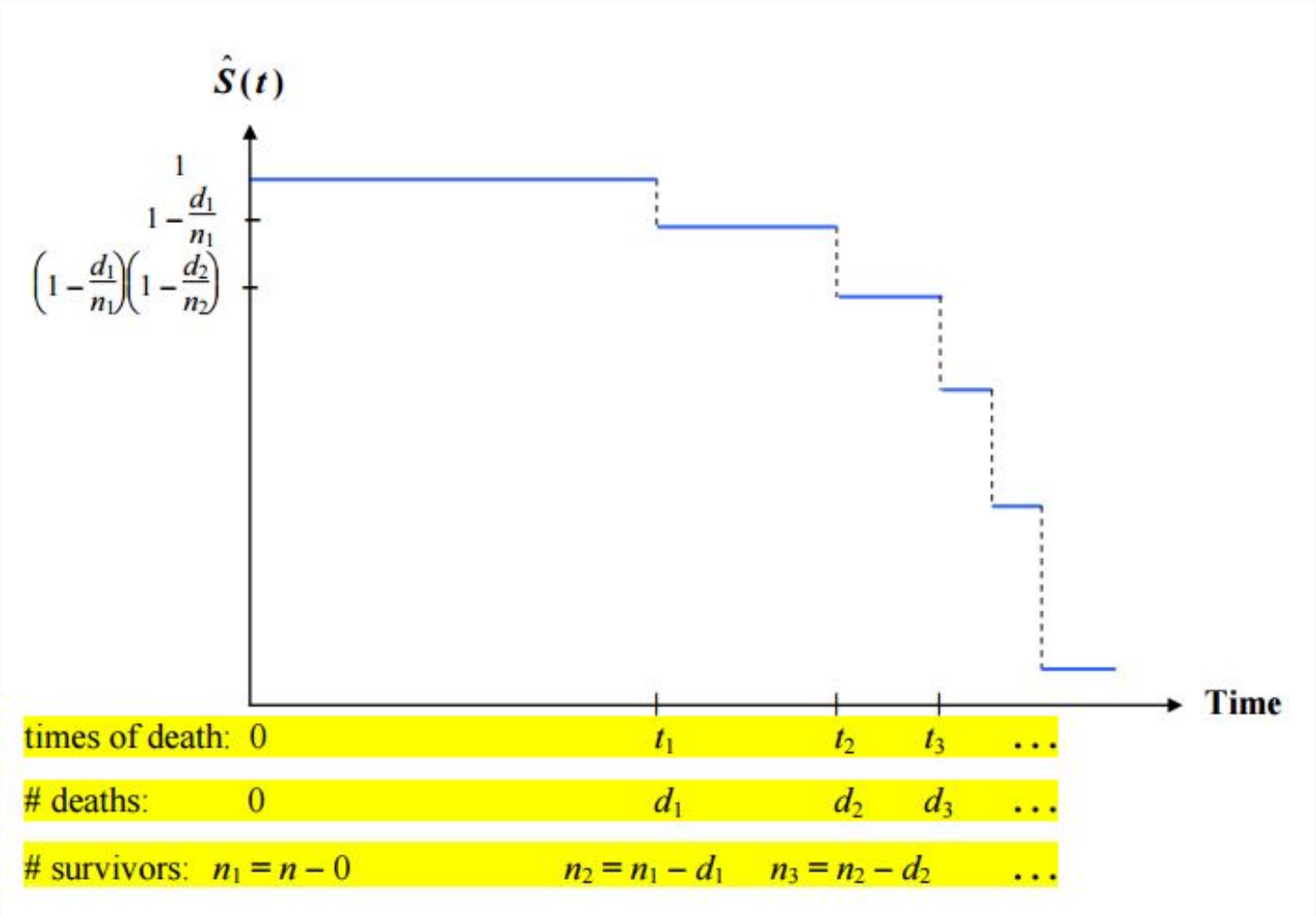
- For any time t in $[t_1, t_2)$, we have...
  $S(t) = P(T > t)$ $P((\text{survive in}[0, t_1)$ X $P(\text{survive in } [t_1, t] \text{ — survive in } [0, t_1))$
  so, $S_E(t) = 1 - (d_1/n_1)$

- For any time t in $[t_2, t_3)$, we have...
  $S(t) = P(T > t)$ $P((\text{survive in}[t_1, t_2)$ X $P(\text{survive in } [t_2, t] \text{ — survive in } [t_1, t_2))$
  so, $S_E(t) = (1 - (d_1/n_1))X(1 - (d_2/n_2))$

- In general , for t in $[t_j, t_{j+1}), j = 1, 2, 3, ...,$ we have...
  $S_E(t) = (1 - (d_1/n_1))X(1 - (d_2/n_2))X...X(1 - (d_j/n_j))$

Kaplan-Meyer statistics III

Kaplan-Meyer Point estimation of survival function:



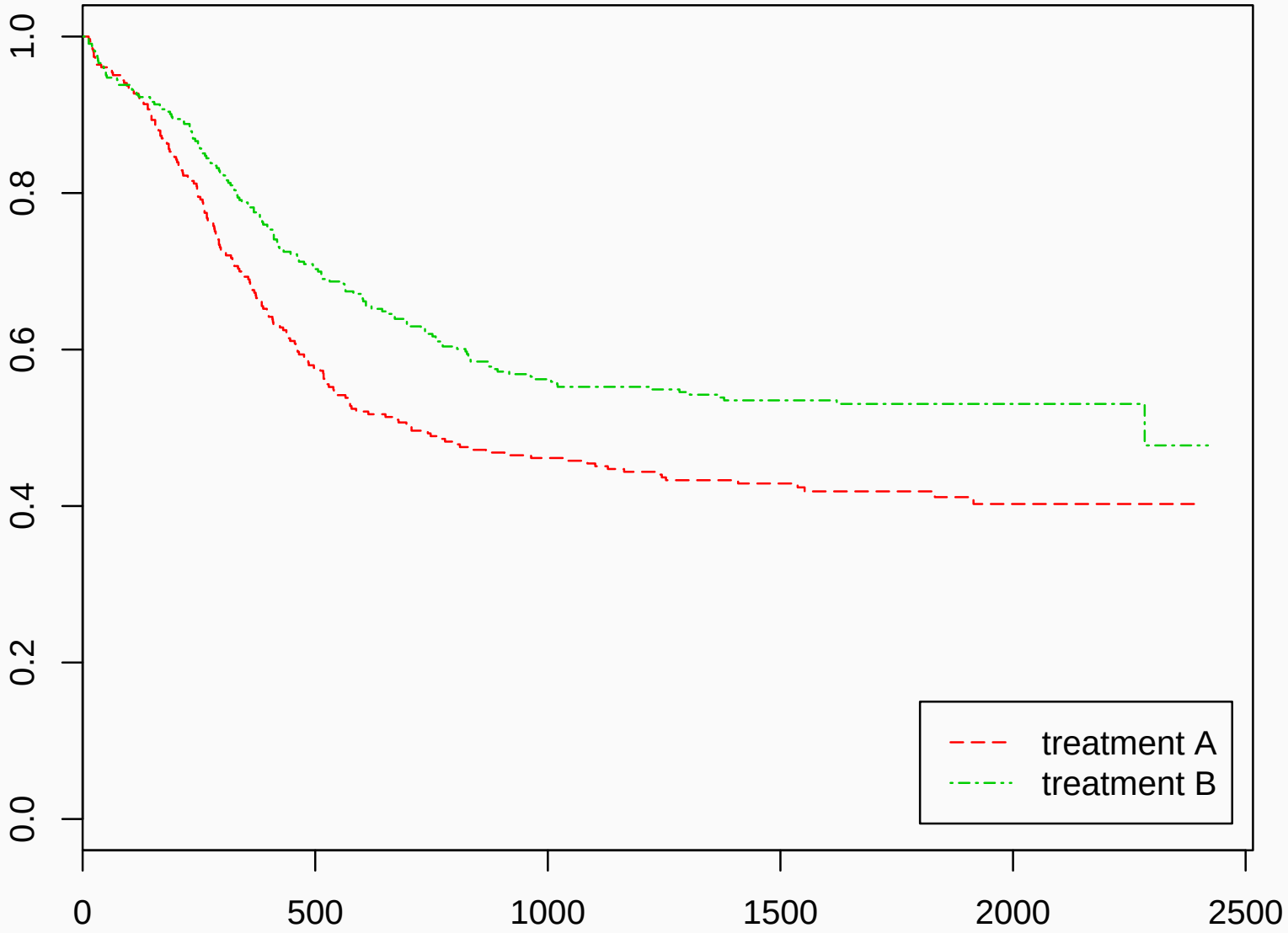
Kaplan-Meyer: Fitting I

we will use `myeloid` data of a simulated RCT for Acute Myeloid Leukemia patients in the "survival" library

```
require(survival)

## Loading required package:  survival

survobj<-with(myeloid,Surv(futime, death==1))
```

```
plot(survfit(survobj trt,data=myeloid),col=c(2,3),lty=c(5,6))
legend(1800,0.15,c("treatment A","treatment B"),lty=c(5,6),col=c(2,3))
```

Kaplan-Meyer: Fitting II



Hazard, Hazard ratio and Proportional hazard model

Hazard function/ Loss function/ Failure rate is defined by : **Ratio of probability that a fail will occur in next t small time increment from T to the probability that the fail has not occurred until time T**

$$h(t) = \frac{f(t)}{1 - F(t)}$$

Hazard ratio is as the term explains, ratio of hazards of two different competing parameters in a common set up.
`Example:` In an RCT say the cases are dying 1.2 times as fast as the controls. This means hazard ratio for the control to the cases is 1.2

Proportional Hazard models are a class of survival objects that are modeled with hazard ratio rather than using the hazard itself. Rationale behind this set up is to have perspective of magnitude in difference in multiplicative hazard comparison

Cox-ph characteristic Requirements

Assumptions:

- First and foremost is the issue of *non-informative censoring*

  To satisfy this assumption, the design of the underlying study must ensure that the mechanisms giving rise to censoring of individual subjects are not related to the probability of an event occurring. For example, in clinical studies, care must be taken that continuation of follow-up not depend on a participants medical condition. Violation of this assumption can invalidates just about any sort of survival analysis

- The second key assumption in the Cox model is that of *proportional hazards*

  In a regression type setting this means that the survival curves for two strata (determined by the particular choices of values for the x-variables) must have hazard functions that are proportional over time (i.e. constant relative hazard)

Fitting I

```
cox.1<-coxph(Surv(futime, death) ~ trt, data=myeloid)
```

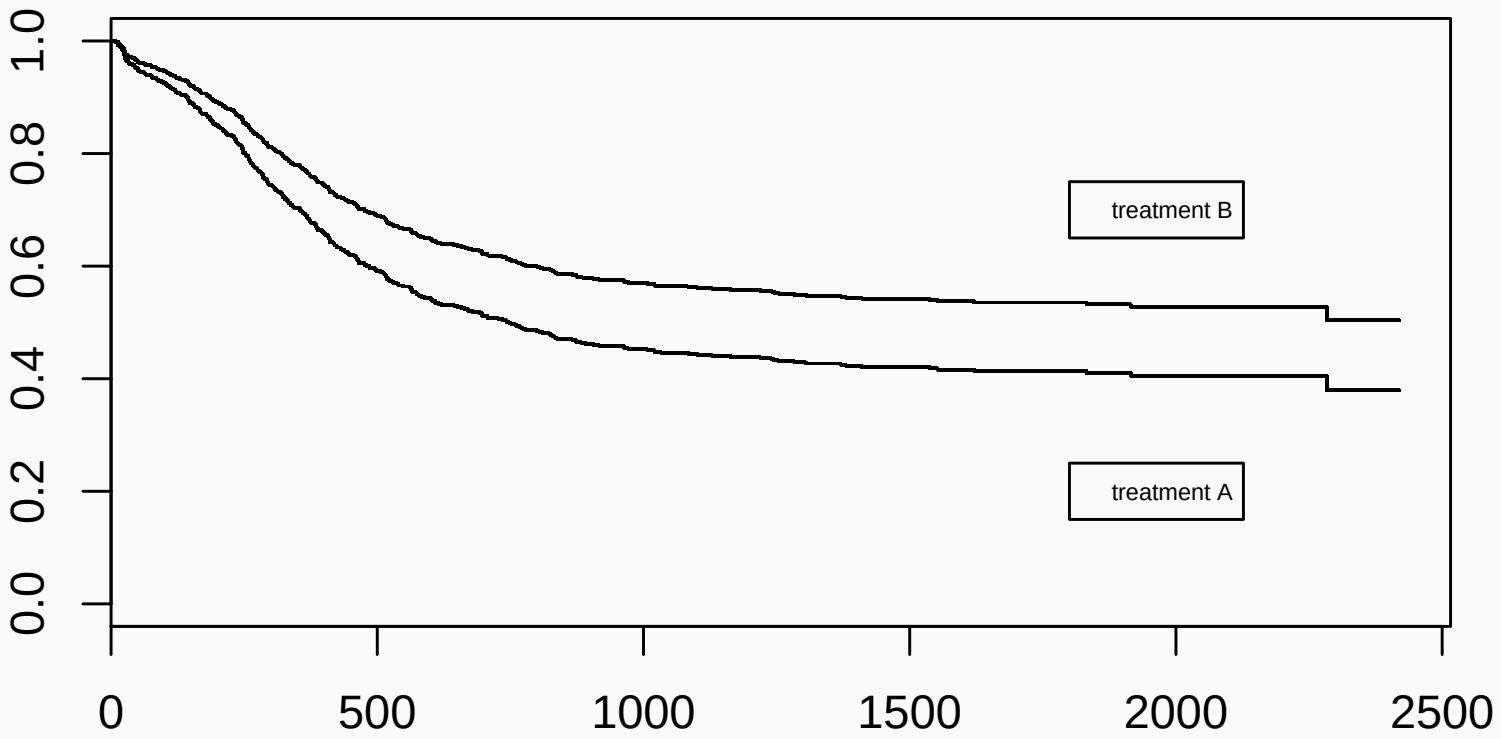
Check for hazard proportionality assumption conformation

```
cox.zph(cox.1)
```

```
##         rho chisq     p
## trtB 0.0412 0.542 0.462
```

Fitting II

```
plot(survfit(cox.1,myeloid,lty=c(4,6)))
legend(1800,0.25,"treatment A",cex=0.5)
legend(1800,0.75,"treatment B",cex=0.5)
```



Exercise

Instructions

- Choose the `kidney` data from package survival
- Fit a proportional hazard model on time against status, explained by age and sex
- Note down your observations
- plot the survival curves

High-Dimensional Data Handling

High Dimensional Data I

Problem with high dimensional data?

- Over Filtting
- Dependecy among variables: multicollinearity
- Loss in degress of Freedom
- Variable Selection

```
library(devtools)
install_github('datamicroarray','ramey')
library(datamicroarray)
data('chin', package = 'datamicroarray')
table(chiny)
chinx[1 : 10, 1 : 5]
```

Principal Component Analysis (PCA) I

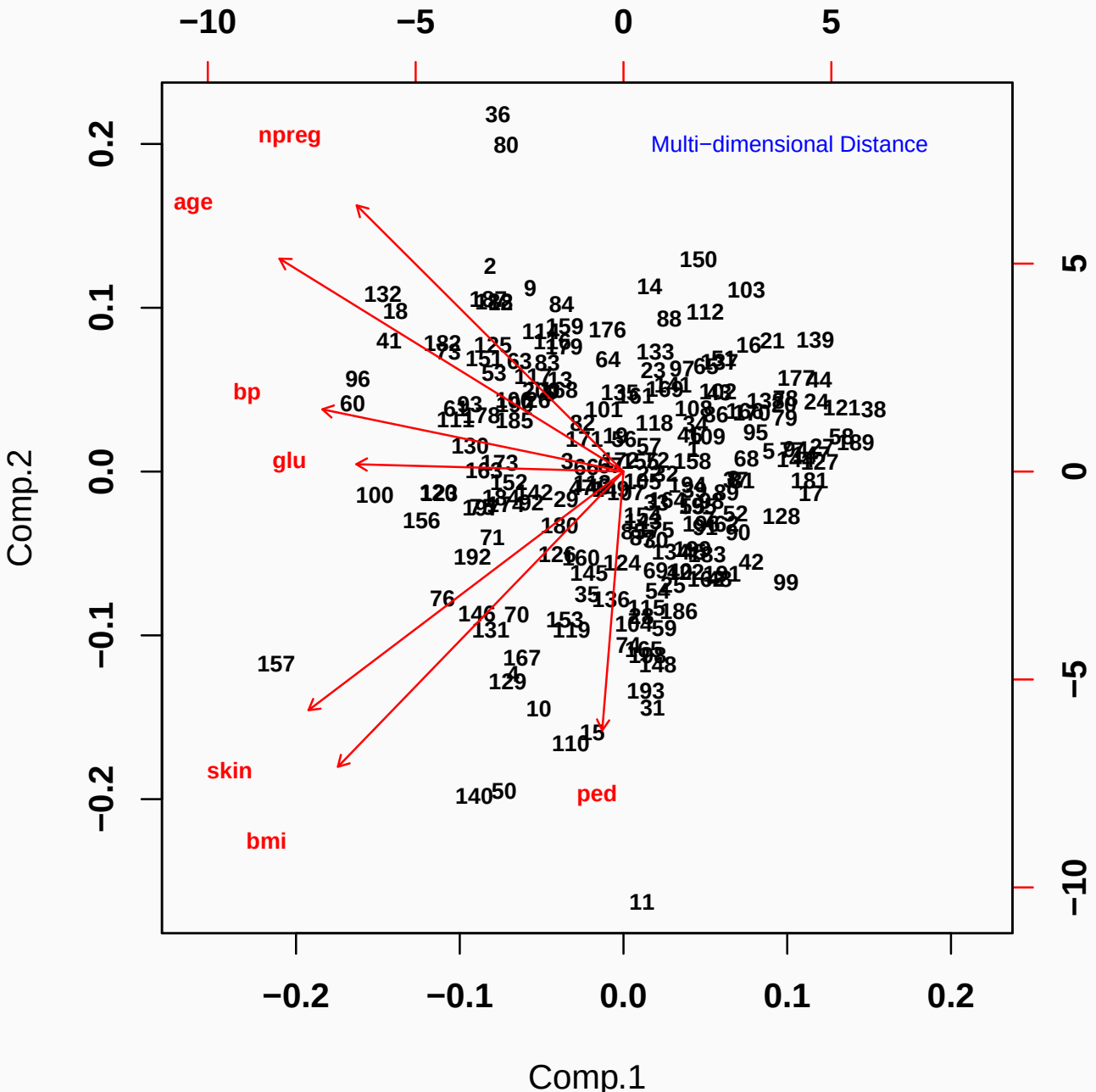
Workflow:

- PCA takes a data matrix of n objects by p variables, which may be correlated, and summarizes it by uncorrelated axes (principal components or principal axes) that are linear combinations of the original p variables
- Euclidean Distance calculated from the p variables as the measure of dissimilarity among the n objects
- The first k components display as much as possible of the variation among objects

```
prcm.1 <- princomp(Pima.tr[,-8], cor=TRUE)
prcm.1
```

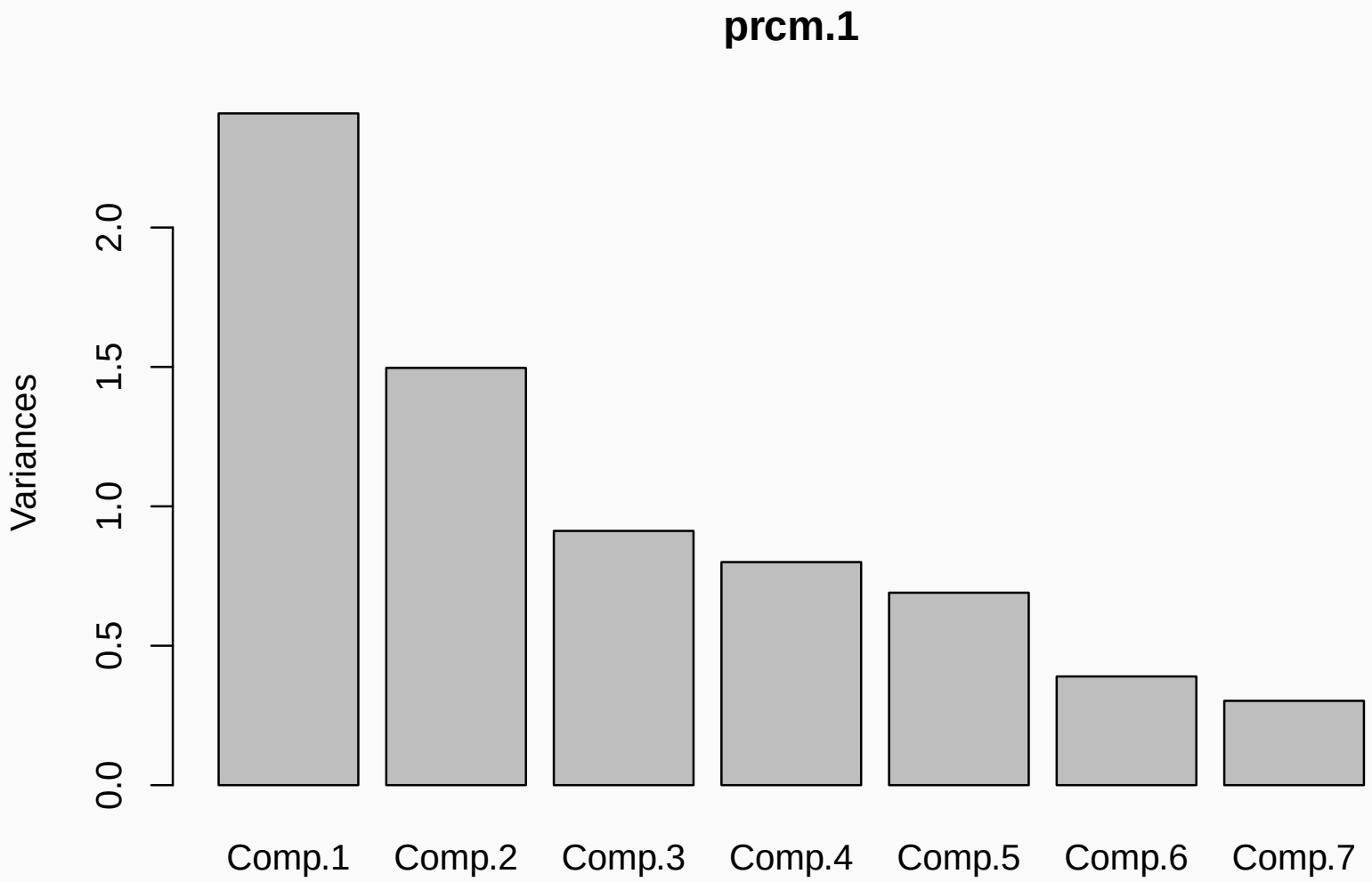
```
biplot(prcm.1,cex=0.7,arrow.len=0.06,font=2)
text(4,8.5,"Multi-dimensional Distance",pos=1,col=4,cex=0.7)
```

Principal Component Analysis (PCA) II



Principal Component Analysis (PCA) III

```
plot(prcm.1)
```



Exploratory Factor Analysis (EFA)

Factor analysis is a useful tool for investigating variable relationships for complex concepts

- The key concept of factor analysis is that multiple observed variables have similar patterns of responses because they are all associated with a latent variable
- In every factor analysis, there are the same number of factors as there are variables. Each factor captures a certain amount of the overall variance in the observed variables, and the factors are always listed in order of how much variation they explain
- The eigenvalue is a measure of how much of the variance of the observed variables a factor explains. Any factor with an eigenvalue greater than 1 explains more variance than a single observed variable

```
#set.seed(011127)
#a<-sample(seq(1:4),100,replace=TRUE)
#b<-data.frame(matrix(a,nrow=10,ncol=10))
#require(psych)
#require(GPArotation)
#corMat<-cor(b)
#solution<-fa(r=corMat,nfactors=6,rotate="oblimin",fm ="miners")
#solution
```

Fisher's Linear Discriminant Analysis (LDA) I

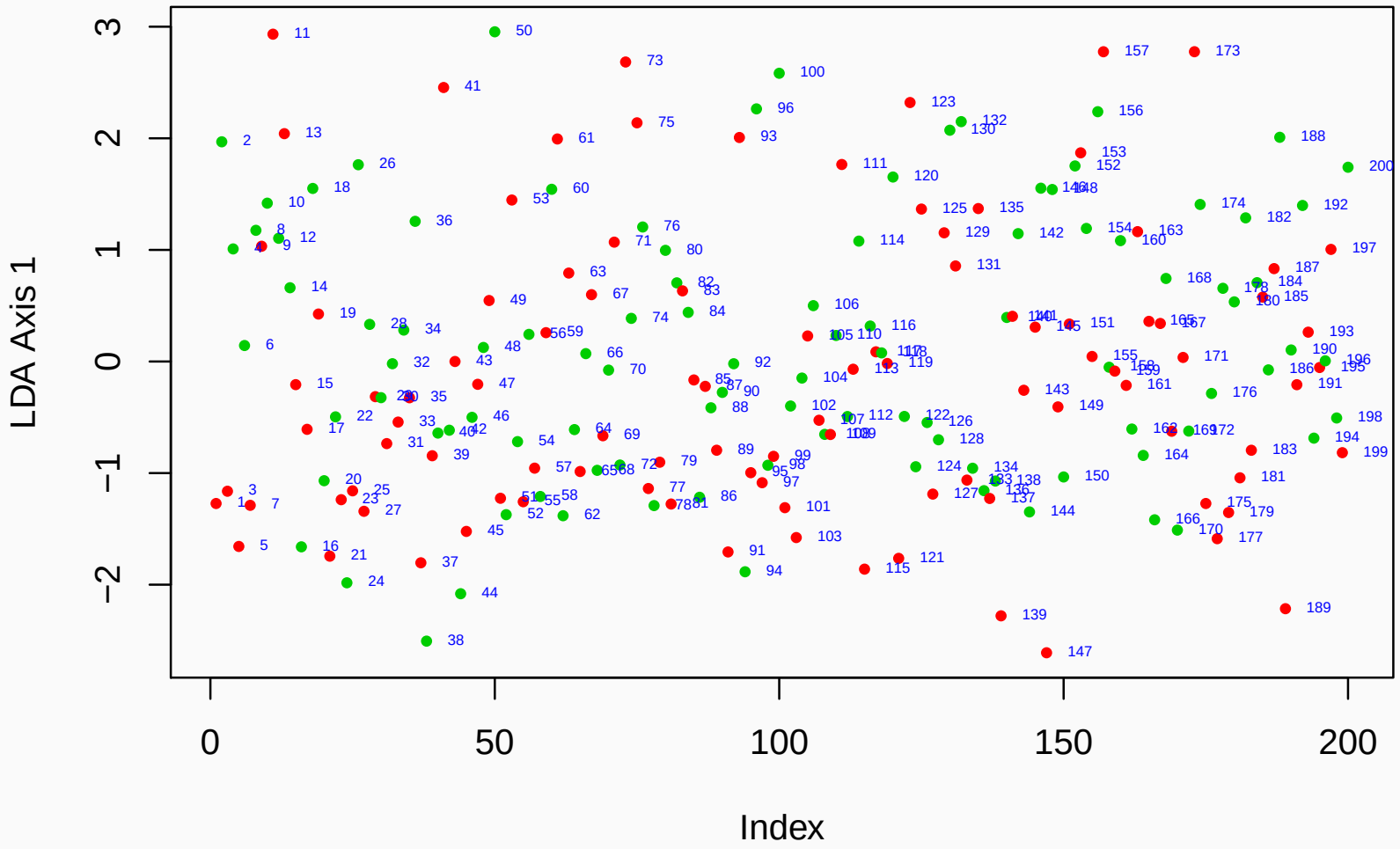
The idea is to find the line that best separates the two classes. LDA finds a linear combination of features which characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier, or, more commonly, for dimensionality reduction before later classification.

- it is optimal if and only if the classes are Gaussian and have equal covariance
- better than PCA, but not necessarily good enough

```
require(MASS)
ld.1<-lda(type .,data=Pima.tr)
pred<-predict(ld.1,Pima.tr)
```

```
plot(predx[,1],pch=16,ylab=c("LDAAxis1"),col=c(2,3),cex=0.7)
text(predx[,1],offset=0.5,pos=4,cex=0.4,col=4)
```

Fisher's Linear Discriminant Analysis (LDA) II



Missing value and Imputation

Missing value and Imputation

Missing values in high dimensional data is a very common difficulty to come accross. Problems thus created are:

- Unstable data matrix, hence unstable variance matrix
- Loss in predictive information
- Relative deflation/inflation in test statistic

There are some standard packages in R to deal with missing values, some of the mostly used are :

- mice (Multivariate Imputation via Chained Equations)
- Amelia (Amelia Earhart)
- missForest
- Hmisc (Harrell Miscellaneous)
- mi (Missing data Imputation and model Checking)

Algorithms of Imputation Algorithms

mice

- PMM (Predictive Mean Matching) - For numeric variables
- logreg(Logistic Regression) - For Binary Variables
- polyreg(Bayesian polytomous regression) - For Factor Variables
- Proportional odds model

Amelia

- Bootstrapping and EMB opitimization

missForest

- Bootstrapped randomforest

Hmisc

- It assumes linearity in the variables being predicted
- Fisher's optimum scoring method is used for predicting categorical variables

mi

- Predictive mean matching model

mice

```
data<-missForest::prodNA(Pima.tr[,-8], noNA = 0.1)#Create Missing data
require(mice)

## Loading required package:  mice

system.time(imputed_mice<-mice(data,m=2,maxit=2,method ='pmm',seed=500))

##
##  iter imp variable
##   1   1  npreg  glu  bp  skin  bmi  ped  age
##   1   2  npreg  glu  bp  skin  bmi  ped  age
##   2   1  npreg  glu  bp  skin  bmi  ped  age
##   2   2  npreg  glu  bp  skin  bmi  ped  age
##    user  system elapsed
##    0.09    0.00    0.09
```

Amelia

```
set.seed(7544)
system.time(imputed_amelia<-Amelia::amelia(data,m=4))

## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8
##    user  system elapsed
##    0.08    0.00    0.08

#imputed_amelia$imputations
```

missForest

```r
set.seed(7544)
#require(missForest)
system.time(imputed_forest<-missForest::missForest(data,maxiter=4,ntree=50))

##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
##    user  system elapsed
##    0.63    0.00    0.64

#imputed_forest$ximp
```

Hmisc

```
set.seed(7544)
system.time(impute_hmisc<-Hmisc::aregImpute(~npreg+glu+bp+
skin+bmi+ped+age,data,n.impute=4))

## Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7

##    user  system elapsed
##    1.18    0.05    1.22

#head(impute_hmisc$imputed$npreg)
```

```r
set.seed(7544)
system.time(imputed_mi<-mi::mi(data,n.iter=4))

##    user  system elapsed
##    0.33    0.03    4.62

imputed<-mi::complete(imputed_mi)
#head(imputed)
```

Appendix

Packages

- outliers
- MASS
- car
- lme4
- nlme
- class
- rpart
- randomForest
- nnet
- e1071
- fastcluster
- survival
- devtools
- GitHub: datamicroarray
- GitHub: ramey
- mice
- Amelia
- missForest
- Hmisc
- mi