Survival Analysis and Unsupervised Learning

Nathan Taback

06/02/2020

Survival Analysis

- Outcome variable: Time until an event occurs
- Start follow-up $\xrightarrow{\text{Time}}$ Event
- Event examples: death; disease; recovery; purchase.
- Time is often called survival time since it's the time that an individual has survived over some follow-up period.

Censored Data

- Censoring occurs when we have some information about individual survival time, but it's not exactly known.
- Consider a person followed in a medical study until they die. If the study ends while the patient is still alive then the patient's survival time is censored, since the person will die after the study ends.
- Common reasons why censoring may occur:
  - study ends - no event
  - lost to follow-up
  - withdraw from study

Survival Data

The data below is from Pancreas cancer patients in TCGA.

days_to_last_follow_up	days_to_death	vital_status
–	12	Dead
706	–	Alive
–	239	Dead
1794	–	Alive
–	153	Dead
33	–	Alive

How can observed survival time be defined?

Survival Data

days_to_last_follow_up	days_to_death	vital_status
–	12	Dead
706	–	Alive
–	239	Dead
1794	–	Alive
–	153	Dead
33	–	Alive

- If `vital_status = Dead` and `days_to_death` is not missing then $T =$ `days_to_death`.
- If `vital_status = Alive` and `days_to_last_follow_up` is not missing then $T =$ `days_to_last_follow_up`.
- If a person is still alive during the study period then they are censored; if the person died then the event occured.

Survival Data - Define Survival Time

Define survival time:

```
clinical$os_days <- ifelse(clinical$vital_status == "Alive",
                           clinical$days_to_last_follow_up,
                           ifelse(clinical$vital_status == "Dead",
                                  clinical$days_to_death, NA))
```

days_to_last_follow_up	days_to_death	vital_status	os_days
–	12	Dead	12
706	–	Alive	706
–	239	Dead	239
1794	–	Alive	1794
–	153	Dead	153
33	–	Alive	33

Survival Data - Define Censoring

Define event indicator for subject i:

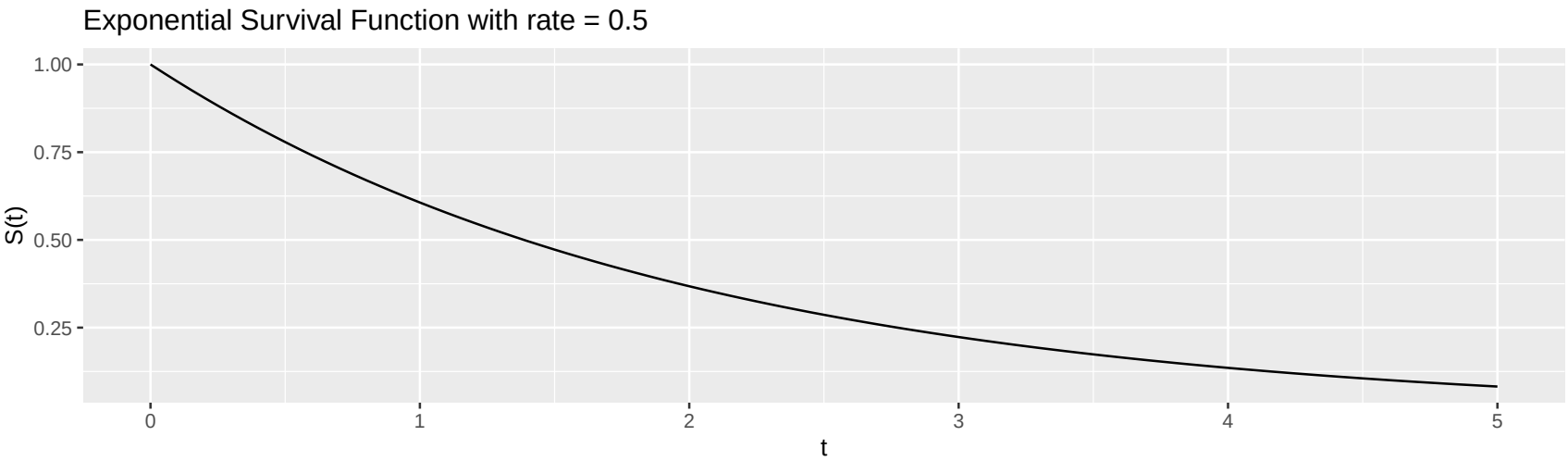
$$\delta_i = \begin{cases} 1 & \text{if death} \\ 0 & \text{if censored} \end{cases}$$

```
clinical$dead <- ifelse(clinical$vital_status == "Alive", 0,
                        ifelse(clinical$vital_status == "Dead", 1, NA))
```

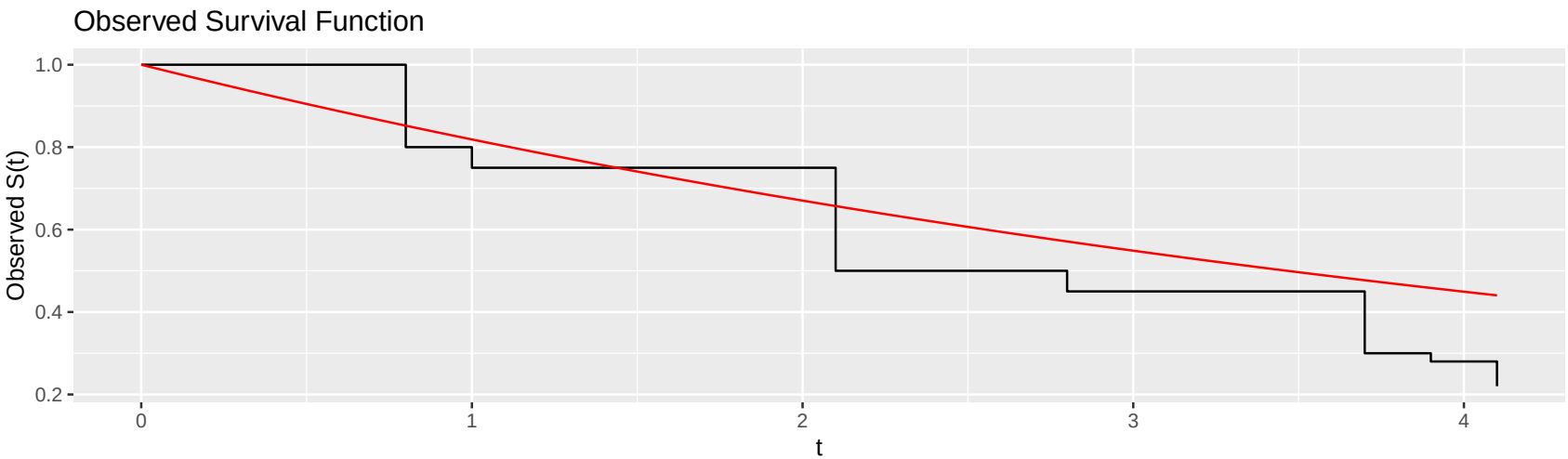
days_to_last_follow_up	days_to_death	vital_status	os_days	dead
–	12	Dead	12	1
706	–	Alive	706	0
–	239	Dead	239	1
1794	–	Alive	1794	0
–	153	Dead	153	1
33	–	Alive	33	0

Survivor Function

- Let $T \geq 0$ be a random variable for a person's survival time.
- $S(t) = P(T > t) = 1 - F(t)$ is the survivor function.
- For example, if $T \sim exp(\lambda)$ then $S(t)$ for $\lambda = 1/2$ is



- Observed survival function and exponential survival with $\lambda = 1/2$.



Hazard Function

The hazard function is defined as:

$$h(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t | T \geq t)}{\Delta t}.$$

- $h(t)\Delta t \approx P(t \leq T < t + \Delta t | T \geq t)$ - the probability of death in $(t, t + \Delta t)$ given survival up to time t.
- For example, if time is measured in days then $h(t)$ is the approximate probability that an individual who is alive on day t, dies in the following day.

Non-parametric Estimation of the Survival Function

$$\hat{S}(t) = \frac{\text{Number of individuals with survival times} \geq t}{\text{Number of individuals in data set}}.$$

- The Kaplan-Meier estimate is an example of a procedure for estimating $\hat{S}(t)$.

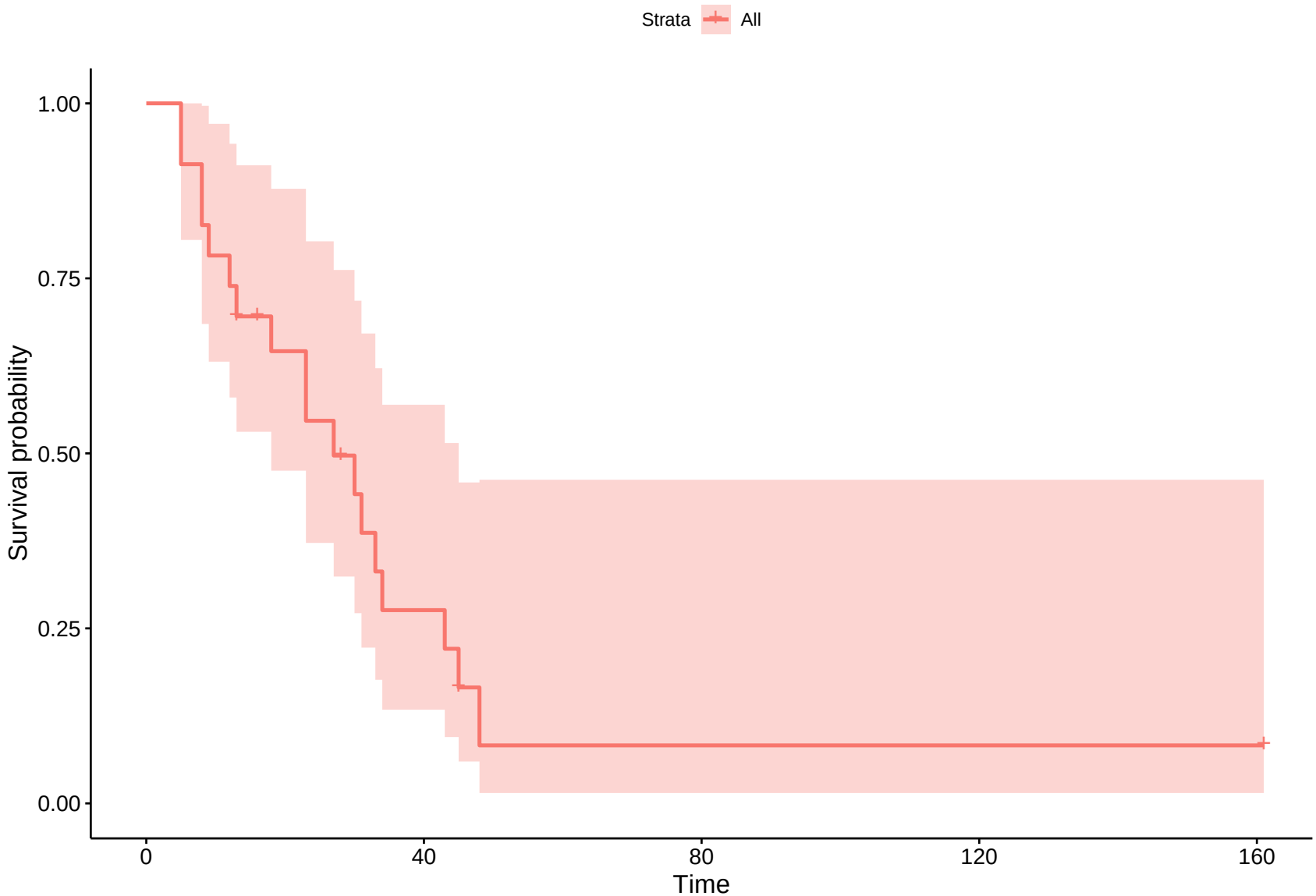
Kaplan-Meier

```
km <- survfit(Surv(time, status) ~ 1, data = leukemia)
summary(km)
```

```
## Call: survfit(formula = Surv(time, status) ~ 1, data = leukemia)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     5     23       2   0.9130  0.0588        0.8049        1.000
##     8     21       2   0.8261  0.0790        0.6848        0.996
##     9     19       1   0.7826  0.0860        0.6310        0.971
##    12     18       1   0.7391  0.0916        0.5798        0.942
##    13     17       1   0.6957  0.0959        0.5309        0.912
##    18     14       1   0.6460  0.1011        0.4753        0.878
##    23     13       2   0.5466  0.1073        0.3721        0.803
##    27     11       1   0.4969  0.1084        0.3240        0.762
##    30      9       1   0.4417  0.1095        0.2717        0.718
##    31      8       1   0.3865  0.1089        0.2225        0.671
##    33      7       1   0.3313  0.1064        0.1765        0.622
##    34      6       1   0.2761  0.1020        0.1338        0.569
##    43      5       1   0.2208  0.0954        0.0947        0.515
##    45      4       1   0.1656  0.0860        0.0598        0.458
##    48      2       1   0.0828  0.0727        0.0148        0.462
```

Kaplan-Meier

```
library(survminer)
ggsurvplot(km)
```



Proportional Hazards Model

- Suppose we have two groups of patients: A and B.
- Assume the hazard at time t for a patient in B is proportional to the hazard at time t for a patient in group A. That is,

$$h_B = \psi h_A(t),$$

where $t \geq 0$ and ψ is a constant.

- It's convenient to set $\psi = \exp(\beta)$, since the ratio is always positive. Let $x_i = 1$, if subject is in B and $x_i = 0$, if subject is in A then

$$h_i(t) = \exp(\beta x_i) h_0(t),$$

This is the proportional hazards model for comparing two groups, with $\psi = \exp(\beta x_i)$.

General Proportional Hazards Model

- Suppose that we would like to model the hazard of death at particular time as a linear function of p explanatory variables: $x_1, \ldots, x_p$.
- In this case,

$$\psi_i = \beta_1 x_{1i} + \cdots + \beta_p x_{pi}.$$

So, $h_i(t) = \exp(\beta_1 x_{1i} + \cdots + \beta_p x_{pi}) h_0(t)$, or,

$$\log\left\{\frac{h_i(t)}{h_0(t)}\right\} = \beta_1 x_{1i} + \cdots + \beta_p x_{pi}.$$

- $h_0(t)$ is called the baseline hazard - the hazard with no x's in the model.
- A linear model for the logarithm of the hazard ratio.

Fitting the proportional Hazards model

- Cox(1972) derived the appropriate likelihood for this model.
- MLE of the β parameters can be found by maximizing the log-likelihood function using numerical methods such as Newton-Raphson.

Fitting the proportional Hazards model

```
cox.mod <- coxph(Surv(time, status) ~ x, data = leukemia)
summary(cox.mod)
```

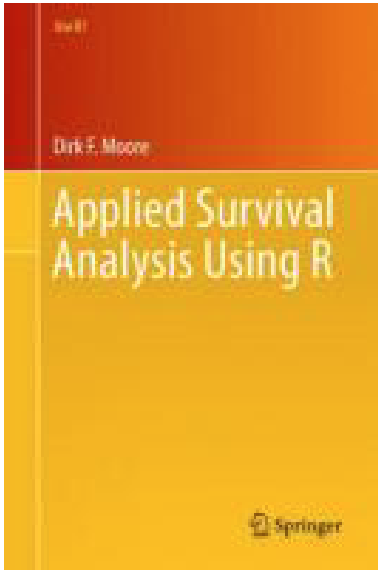
```
## Call:
## coxph(formula = Surv(time, status) ~ x, data = leukemia)
##
##   n= 23, number of events= 18
##
##                    coef exp(coef) se(coef)     z Pr(>|z|)
## xNonmaintained 0.9155    2.4981   0.5119 1.788   0.0737 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                exp(coef) exp(-coef) lower .95 upper .95
## xNonmaintained     2.498     0.4003    0.9159     6.813
##
## Concordance= 0.619  (se = 0.063 )
## Likelihood ratio test= 3.38  on 1 df,   p=0.07
## Wald test            = 3.2  on 1 df,   p=0.07
## Score (logrank) test = 3.42  on 1 df,   p=0.06
```

- The estimated hazard ratio of patients not given maintenance chemotherapy vs. patients given maintenance chemotherapy is 2.4981.

Cox Model Diagnostics

- Graphical approaches such as log-log plots.
- Goodness of fit tests such as Schoenfield residual plot

Book Reference



UofT lib link to Applied Survival Analysis using R

Unsupervised Learning

- Priciple Component Analysis
- k-means
- Hierarchical Clustering

Principal Components Analysis (PCA)

- Suppose we have n measurements on p features/covariates, $X_1, \ldots, X_p$.
- PCA finds a low-dimensional represen- tation of a data set that contains as much as possible of the variation.
- The first principal component is the linear combination of the features

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \cdots + \phi_{p1} X_p$$

that has the largest variance such that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.

- $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ are called the loadings of the first principal component.

Computing the first PC

- To compute the first principal component centre each of $X_1, \ldots, X_p$ (i.e., so that they have mean 0).
- Then find the linear combination such that:

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \cdots + \phi_{p1} x_{ip}$$

that has the largest variance, subject to $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.

Computing the second PC

The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance out of all linear combinations that are uncorrelated with Z_1.

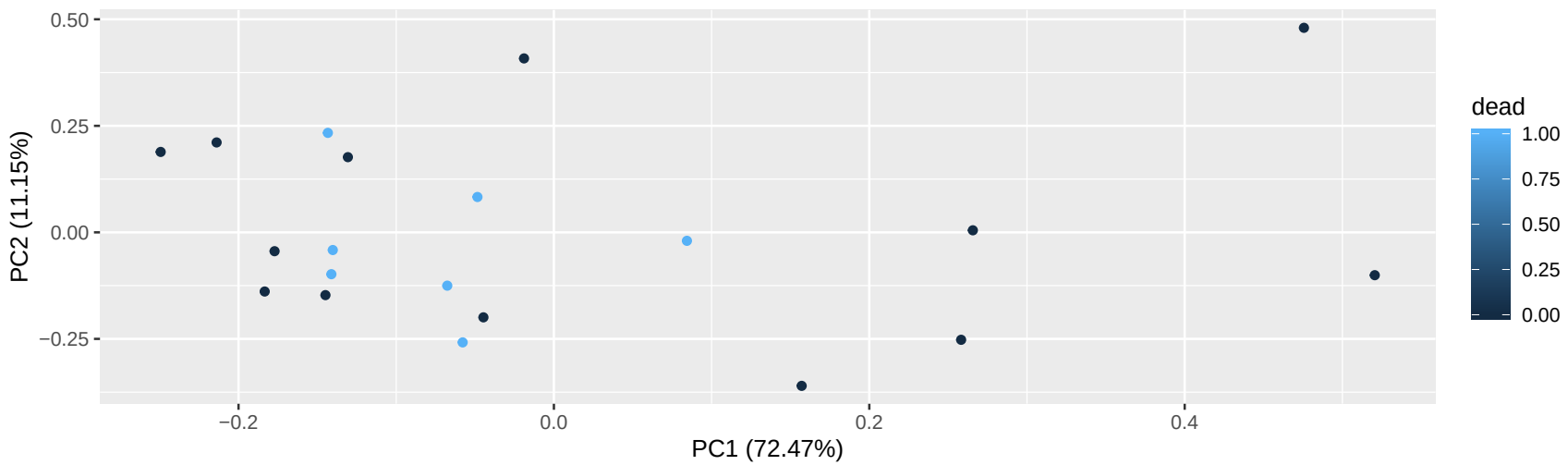
Computing PC

Plot of the score vectors for PC1 and PC2

```
library(ggfortify)
# Random sample of patients and genes from pancreas
pca.out <- prcomp(pancreas_gene_sample[, 9:38], center = T)
as.data.frame(pca.out$rotation[,c(1,2)]) %>% rownames_to_column() %>%
  arrange(desc(PC1)) %>% head(n=3)
```

```
##                rowname        PC1         PC2
## 1 ENSG00000152969.15 0.15331976 -0.09474669
## 2  ENSG00000183580.9 0.08169159 -0.59653953
## 3 ENSG00000169696.14 0.04970751  0.35904737
```

```
autoplot(pca.out, data = pancreas_gene_sample,colour = "dead")
```



More about principal components

- How many principal components? Scree plot.
- Proportion of variance explained.

PCA using sklearn

- In Python use sklearn.decomposition module

```
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
import numpy as np
from sklearn.decomposition import PCA
import pandas as pd

data = load_iris() # load iris data
df = pd.DataFrame(data = data.data,columns = data.feature_names)
x = StandardScaler().fit_transform(df) # standardize data

pca = PCA(n_components=2) # 2 components
pcs = pca.fit_transform(x) # compute pc scores

principalDf = pd.DataFrame(data = pcs,
              columns = ['principal component 1',
              'principal component 2'])
principalDf.head(n=2)
```

```
##    principal component 1  principal component 2
## 0              -2.264703               0.480027
## 1              -2.080961              -0.674134
```

k means

- Try to define k disjoint subsets such that each observation belongs such that the within cluster variation is as small as possible.
- Typically use Euclidean distance to define within cluster variation.

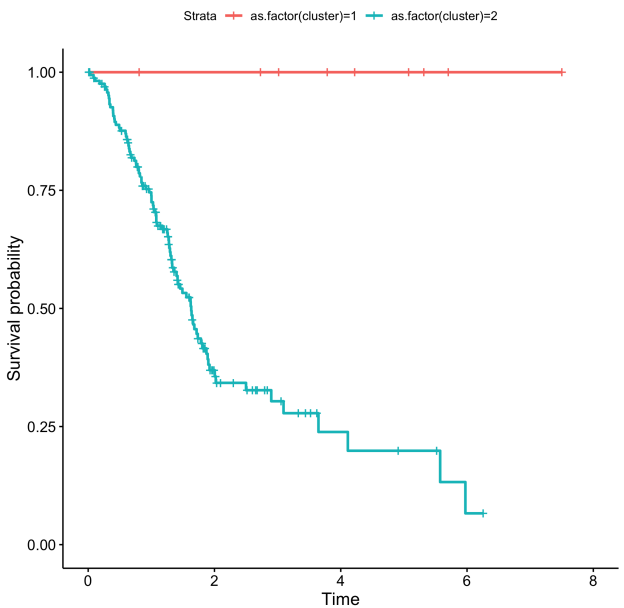
1. Randomly assign a number, from 1 to k, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:

(a) For each of the k clusters, compute the cluster centroid. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.

(b) Assign each observation to the cluster whose centroid is closest (where closest is defined using Euclidean distance).

k means

```
set.seed(416)
k <- kmeans(x = alldat[,c(9:60491)], centers = 2, nstart = 25)
alldat$cluster <- k$cluster
fit <- survfit(Surv(os_days/365.25, dead) ~ as.factor(cluster),
               data = alldat)
ggsurvplot(fit, risk.table = T)
```



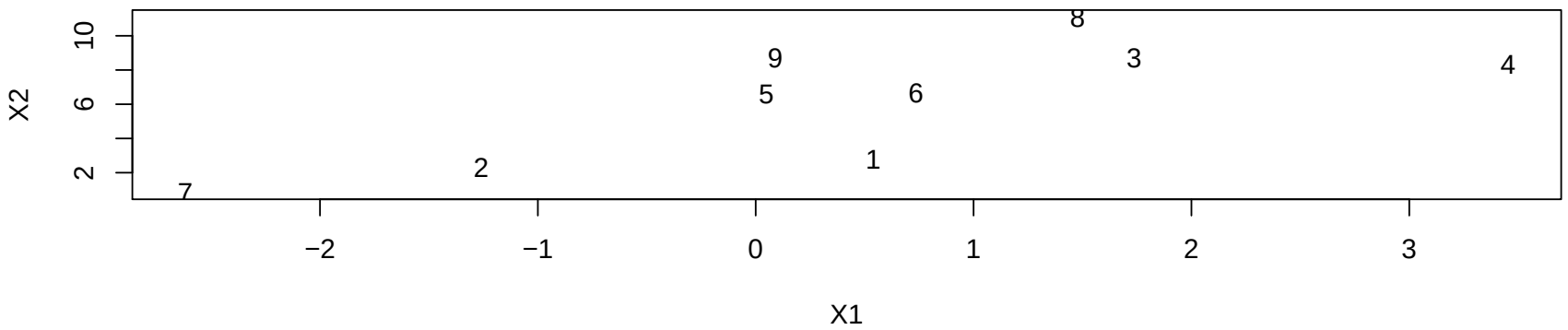
k means

```
table(alldat$cluster,alldat$dead)
```

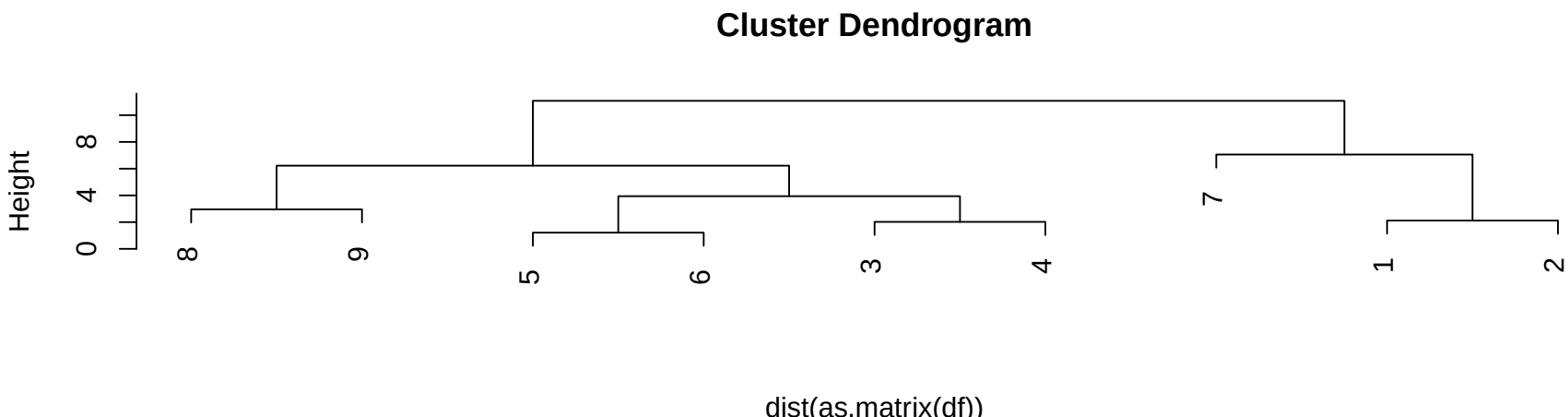
```
   0  1
1  9  0
2 77 92
```

Hierarchical Clustering

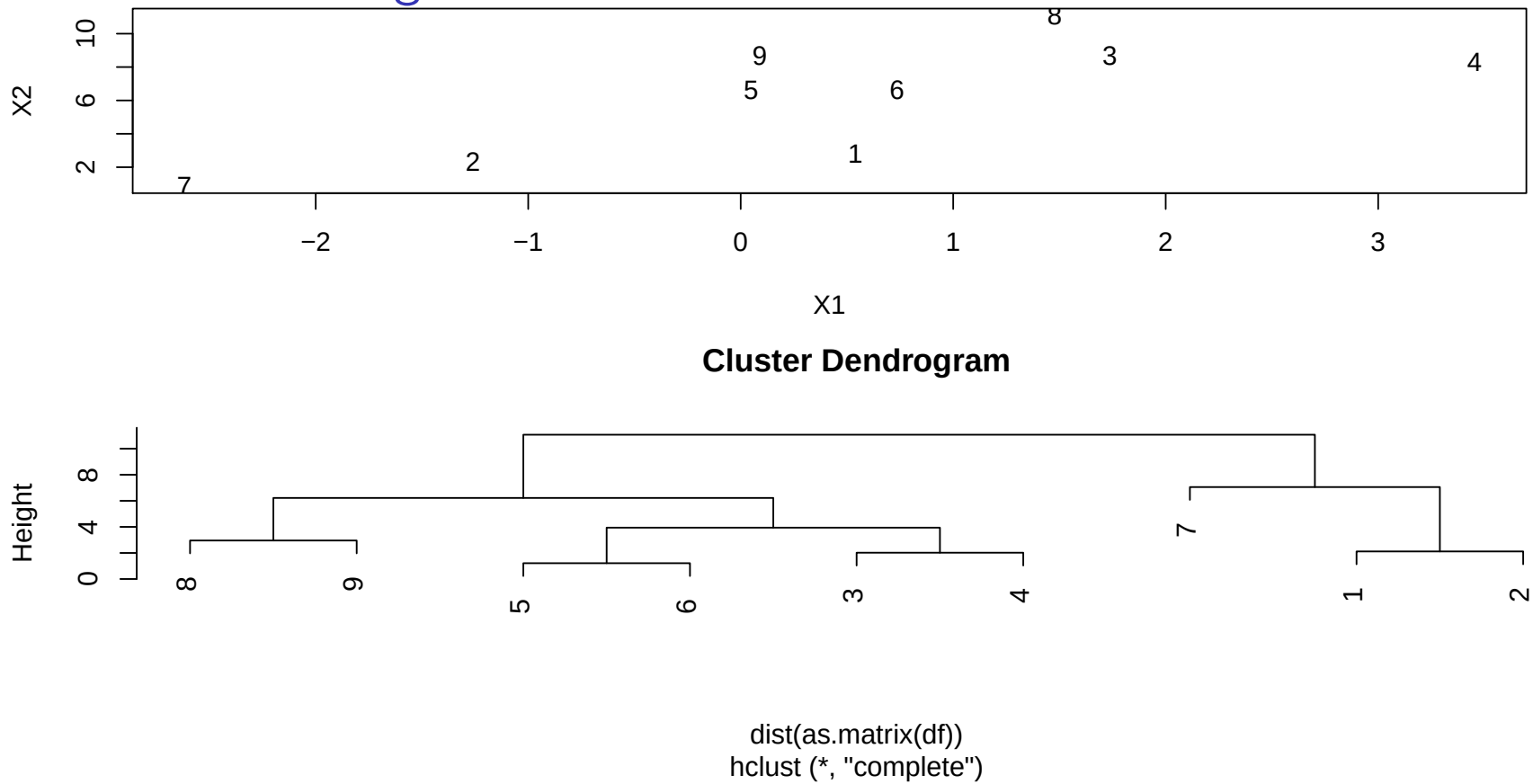
```
set.seed(6)
X1 <- rnorm(9, 0, 2)
X2 <- rnorm(9,0,1) + runif(9, 0, 10)
df <- data.frame(X1,X2)
df$rownum <- 1:nrow(df)
plot(X1,X2, type ="n");text(X1,X2,labels = rownames(df))
```




```
hc <- hclust(dist(as.matrix(df)))
plot(hc)
```



Hierarchical Clustering

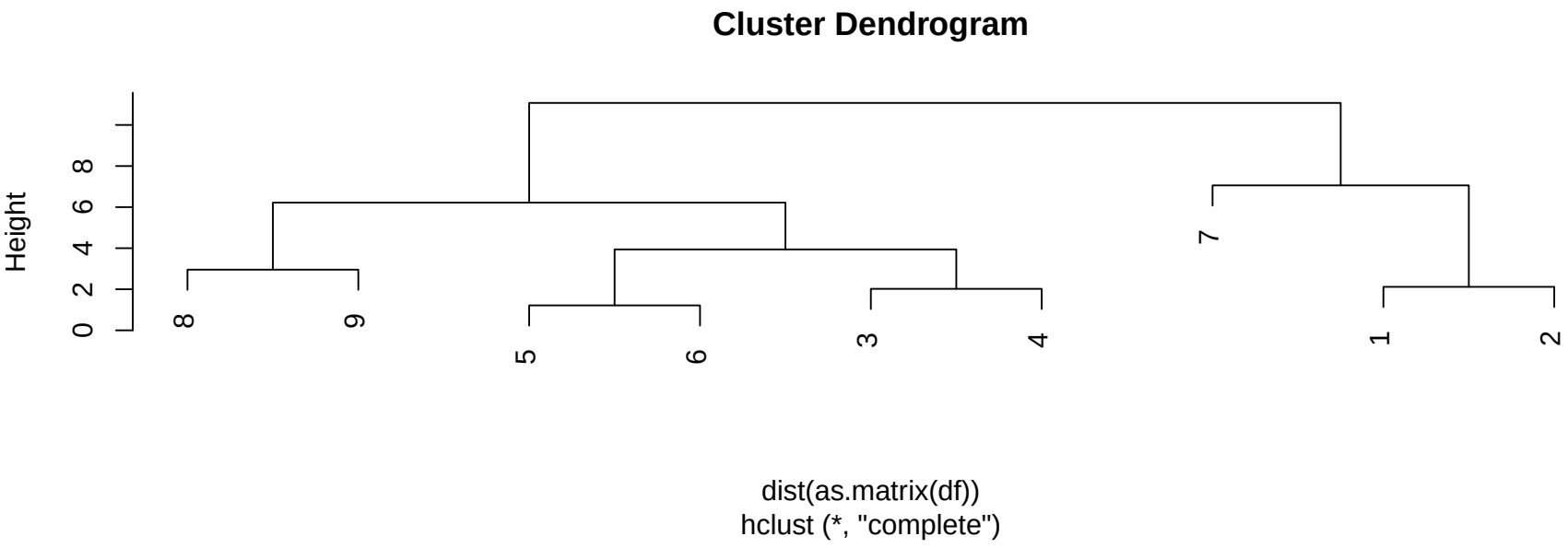



```
hc$merge
```

```
##      [,1] [,2]
## [1,]   -5   -6
## [2,]   -3   -4
## [3,]   -1   -2
## [4,]   -8   -9
## [5,]    1    2
```

Hierarchical Clustering

```
plot(hc)
```




```
cutree(hc, h = 4) # cut tree at height 4
```

```
## [1] 1 1 2 2 2 2 3 4 4
```

```
cutree(hc, h = 7)
```

```
## [1] 1 1 2 2 2 2 3 2 2
```