testing multivariate analyses on cities data

Land use proportional area data for 40 cities on Earth

load packages

load data

Data are from: Hu, J., Wang, Y., Taubenböck, H., Zhu, X.X., 2021. Land consumption in cities: A comparative study across the globe. *Cities*, **113**: 103163, https://doi.org/10.1016/j.cities.2021.103163. (numeric from Fig. 11)

```
cities <- read.csv("cities_Hu_etal_2021.csv", stringsAsFactors = TRUE)
# str(cities)
row.names(cities) <- as.character(cities$City)
cities$sType <- as.character(cities$Type)
cities$sType <- gsub("Compact-Open","CO",cities$sType)
cities$sType <- gsub("Open-Lightweight","OL",cities$sType)
cities$sType <- gsub("Compact","C",cities$sType)
cities$sType <- gsub("Open","O",cities$sType)
cities$sType <- gsub("Industrial","I",cities$sType)
cities$sType <- as.factor(cities$sType)
```

make CLR-transformed dataset

```
cities_clr <- cities
cities_clr[,c("Compact","Open","Lightweight","Industry")] <-
  clr(cities_clr[,c("Compact","Open","Lightweight","Industry")])
```

```
##   ** Are the data/parts all in the same measurement units? **
```

```
names(cities);names(cities_clr)
```

```
## [1] "City"        "Compact"     "Open"        "Lightweight" "Industry"
## [6] "Type"        "Global"      "Region"      "sType"
```

```
## [1] "City"        "Compact"     "Open"        "Lightweight" "Industry"
## [6] "Type"        "Global"      "Region"      "sType"
```

correlation matrices

```
require(DataExplorer)
plot_correlation(cities[,c("Compact","Open","Lightweight","Industry")],
                 cor_args = list("use" = "pairwise.complete.obs"))
```

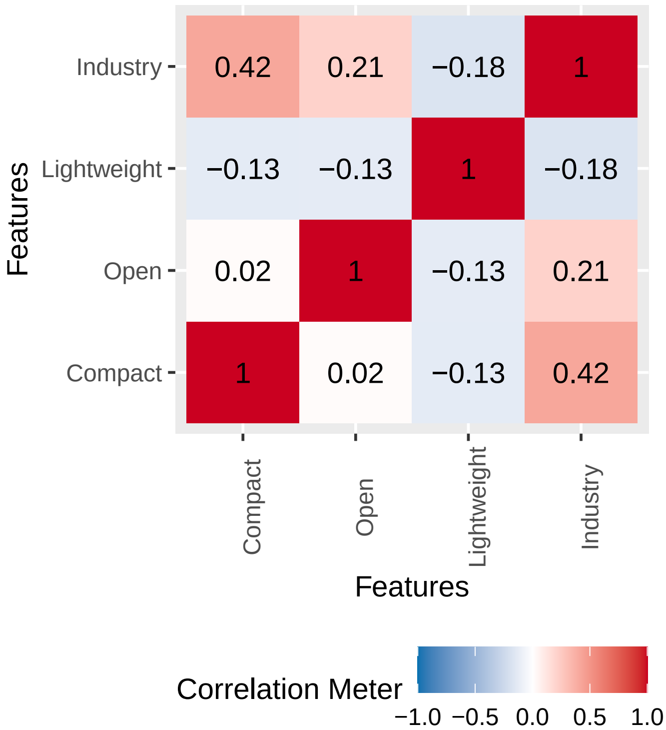


Figure 1: Correlation matrix for closed variables.

correlation matrices (CLR)

```
require(DataExplorer)
plot_correlation(cities_clr[,c("Compact","Open","Lightweight","Industry")],
                 cor_args = list("use" = "pairwise.complete.obs"))
```

principal components analysis

```
data0 <- na.omit(cities[,c("Compact","Open","Lightweight","Industry")])
pca_cities_clos <- prcomp(data0, scale. = TRUE)
pca_cities_clos$rot
```

```
##                    PC1        PC2        PC3         PC4
## Compact     -0.5582052  0.5356252 0.05639404 -0.63113567
## Open        -0.3409659 -0.7576682 0.46899761 -0.29953715
## Lightweight  0.3992211  0.3417785 0.85067121  0.01297763
## Industry    -0.6424731  0.1491041 0.23069343  0.71538580
```

```
cat("...\n\nComponent Variances\n")
```

```
## ...
##
## Component Variances
```

```
pca_cities_clos$sdev^2
```

```
## [1] 1.5860528 1.0026876 0.8705986 0.5406610
```

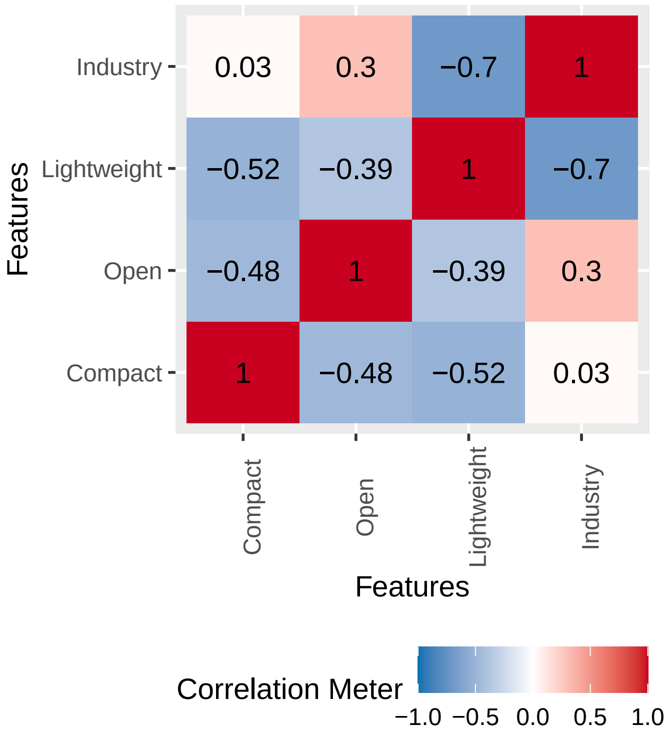


Figure 2: Correlation matrix for open (CLR-transformed) variables.

```
cat("\n--------------------\n")
```

```
##
## --------------------
```

```
data0 <- na.omit(cities_clr[,c("Compact","Open","Lightweight","Industry")])
pca_cities_open <- prcomp(data0, scale. = TRUE)
pca_cities_open$rot
```

```
##                    PC1         PC2        PC3        PC4
## Compact     -0.2081793  0.75498116  0.3214239 -0.5323077
## Open        -0.3535490 -0.62668193  0.5577101 -0.4138022
## Lightweight  0.6833728 -0.17189496 -0.2520621 -0.6632635
## Industry    -0.6038759 -0.08789385 -0.7225723 -0.3248042
```

```
cat("...\n\nComponent Variances\n")
```

```
## ...
##
## Component Variances
```

```
pca_cities_open$sdev^2
```

```
## [1] 1.975945e+00 1.511204e+00 5.128514e-01 1.470492e-31
```

```
cat("\n--------------------\n")
```

```
##
## --------------------
```

```r
rm(data0)
```

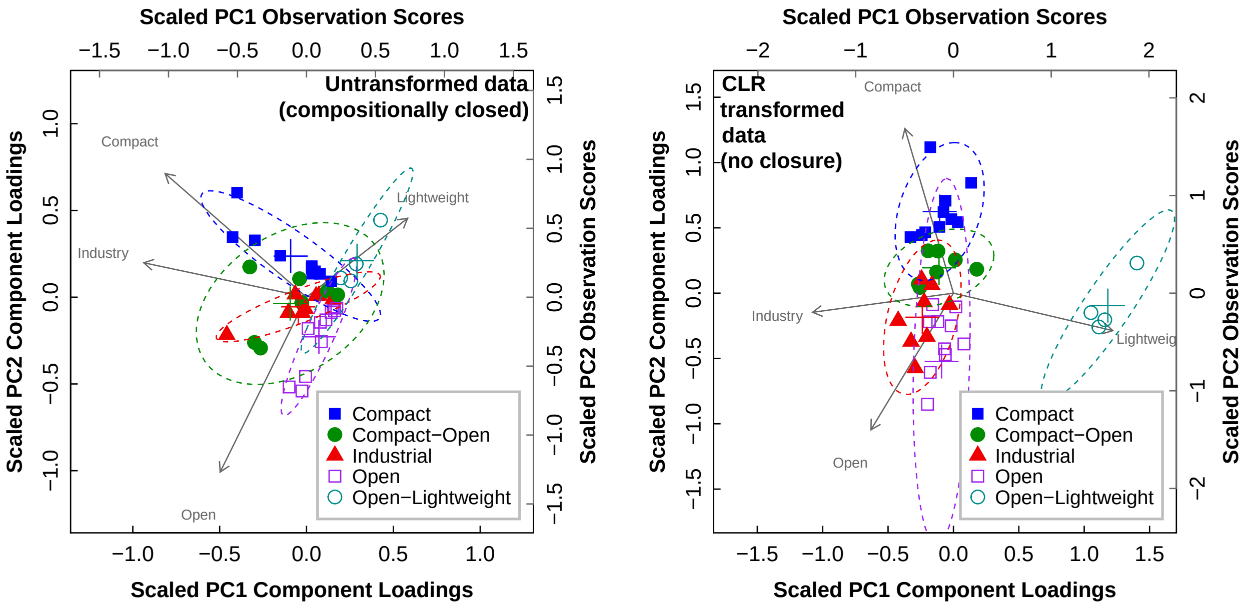
```r
require(car)
palette(c("black","blue","green4","red2","purple",
          "darkcyan","firebrick","grey","grey40","white",
          "transparent"))
par(mfrow=c(1,2), mar = c(3.5,3.5,3.5,3.5), oma = c(0,0,0,0),
    mgp=c(1.7,0.3,0), tcl = 0.25, font.lab=2,
    lend = "square", ljoin = "mitre")
# choose components and set scaling factor (sf)
v1 <- 1
v2 <- 2
sf <- 0.2

biplot(pca_cities_clos, choices = c(v1,v2), col = c(11,9), cex=c(1,0.65),
       pc.biplot = FALSE, scale = 0.4, arrow.len = 0.08,
       xlab = paste0("Scaled PC",v1," Component Loadings"),
       ylab = paste0("Scaled PC",v2," Component Loadings"))
mtext(paste0("Scaled PC",v1," Observation Scores"), 3, 1.6, font = 2)
mtext(paste0("Scaled PC",v2," Observation Scores"), 4, 1.6, font = 2)
mtext("Untransformed data\n(compositionally closed)",
      side = 3, line = -2, font = 2, adj = 0.98)
data0 <- na.omit(cities[,c("Type","Global","Region","Compact","Open","Lightweight","Industry")])
points(pca_cities_clos$x[,v1]*sf, pca_cities_clos$x[,v2]*sf*1.5,
       pch = c(22,21,24,0,1)[data0$Type],
       bg = c(2,3,4,5,6)[data0$Type],
       col = c(2,3,4,5,6)[data0$Type],
       cex = c(1.2,1.4,1.2,1.2,1.4)[data0$Type])
dataEllipse(x=pca_cities_clos$x[,v1]*sf, y=pca_cities_clos$x[,v2]*sf*1.5,
            groups = data0$Type, add = TRUE,
            plot.points = FALSE, levels = c(0.9),
            center.pch = 3, col = c(2,3,4,5,6,7),
            lty = 2, lwd = 1, center.cex = 2.4, group.labels = "")
legend("bottomright", bty = "o", inset = 0.03,
       box.col = "gray", box.lwd = 2, bg = 10,
       legend = levels(data0$Type),
       pch = c(22,21,24,0,1),
       col = c(2,3,4,5,6), pt.bg = c(2,3,4,5,6),
       pt.cex = c(1.2, 1.4, 1.2,1.2,1.4),
       cex = 0.9, y.intersp = 0.9)

# v1 <- 1
# v2 <- 3
sf <- 0.4

biplot(pca_cities_open, choices = c(v1,v2), col = c(11,9), cex=c(1,0.65),
       pc.biplot = FALSE, scale = 0.5, arrow.len = 0.08,
xlab = paste0("Scaled PC",v1," Component Loadings"),
       ylab = paste0("Scaled PC",v2," Component Loadings"))
mtext(paste0("Scaled PC",v1," Observation Scores"), 3, 1.6, font = 2)
mtext(paste0("Scaled PC",v2," Observation Scores"), 4, 1.6, font = 2)
mtext("CLR\ntransformed\ndata\n(no closure)",
      side = 3, line = -4, font = 2, adj = 0.02)
data0 <- na.omit(cities_clr[,c("Type","Compact","Open","Lightweight","Industry")])
points(pca_cities_open$x[,v1]*sf, pca_cities_open$x[,v2]*sf*1.5,
       pch = c(22,21,24,0,1)[data0$Type],
       bg = c(2,3,4,5,6)[data0$Type],
       col = c(2,3,4,5,6)[data0$Type],
       cex = c(1.2,1.4,1.2,1.2,1.4)[data0$Type])
dataEllipse(x=pca_cities_open$x[,v1]*sf, y=pca_cities_open$x[,v2]*sf*1.5,
```

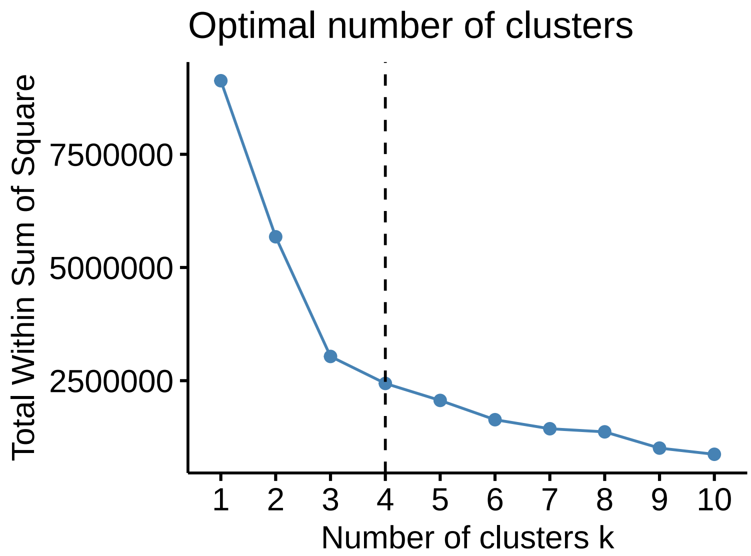
```
              groups = data0$Type, add = TRUE,
              plot.points = FALSE, levels = c(0.9),
              center.pch = 3, col = c(2,3,4,5,6),
              lty = 2, lwd = 1, center.cex = 2.4, group.labels = "")
legend("bottomright", bty = "o", inset = 0.03,
       box.col = "gray", box.lwd = 2, bg = 10,
       legend = levels(data0$Type),
       pch = c(22,21,24,0,1),
       col = c(2,3,4,5,6), pt.bg = c(2,3,4,5,6),
       pt.cex = c(1.2, 1.4, 1.2,1.2,1.4),
       cex = 0.9, y.intersp = 0.9)
```



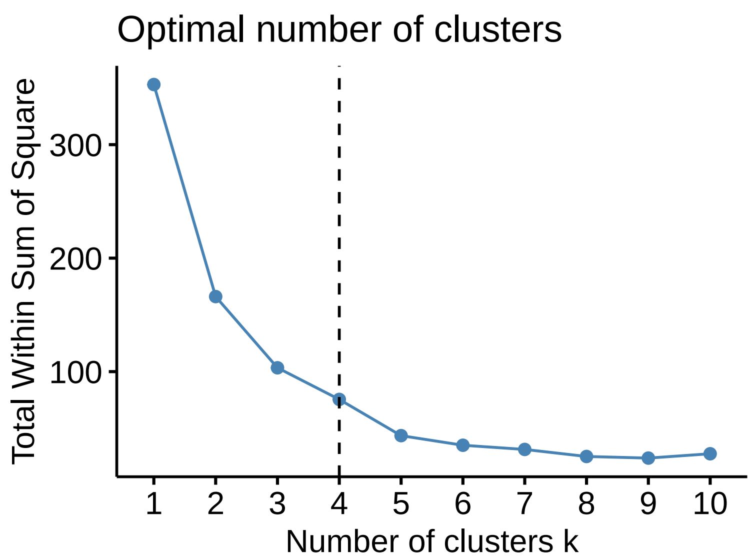

```
rm(list = c("v1","v2","sf","data0"))
```

K-means clustering

```
require(factoextra)
data0 <- na.omit(cities[,c("Compact","Open","Lightweight","Industry")])
fviz_nbclust(data0, kmeans, method = "wss") +
  geom_vline(xintercept = 4, linetype = 2)
```




```
require(factoextra)
data0 <- na.omit(cities_clr[,c("Compact","Open","Lightweight","Industry")])
fviz_nbclust(data0, kmeans, method = "wss") +
  geom_vline(xintercept = 4, linetype = 2)
```



compute K-means closed

```
data0 <- na.omit(cities[,c("sType","Compact","Open","Lightweight","Industry")])
data0[,c("Compact","Open","Lightweight","Industry")] <- scale(data0[,c("Compact","Open","Lightweight","
set.seed(123)
cities_clos_kmeans <- kmeans(data0[,2:NCOL(data0)], 4, nstart = 25)
cat("K-means clustering with",length(cities_clos_kmeans$size),"clusters of sizes",cities_clos_kmeans$si
```

```
## K-means clustering with 4 clusters of sizes 4 5 25 6
```

```
cat("\ncomponents of output object are:\n")
```

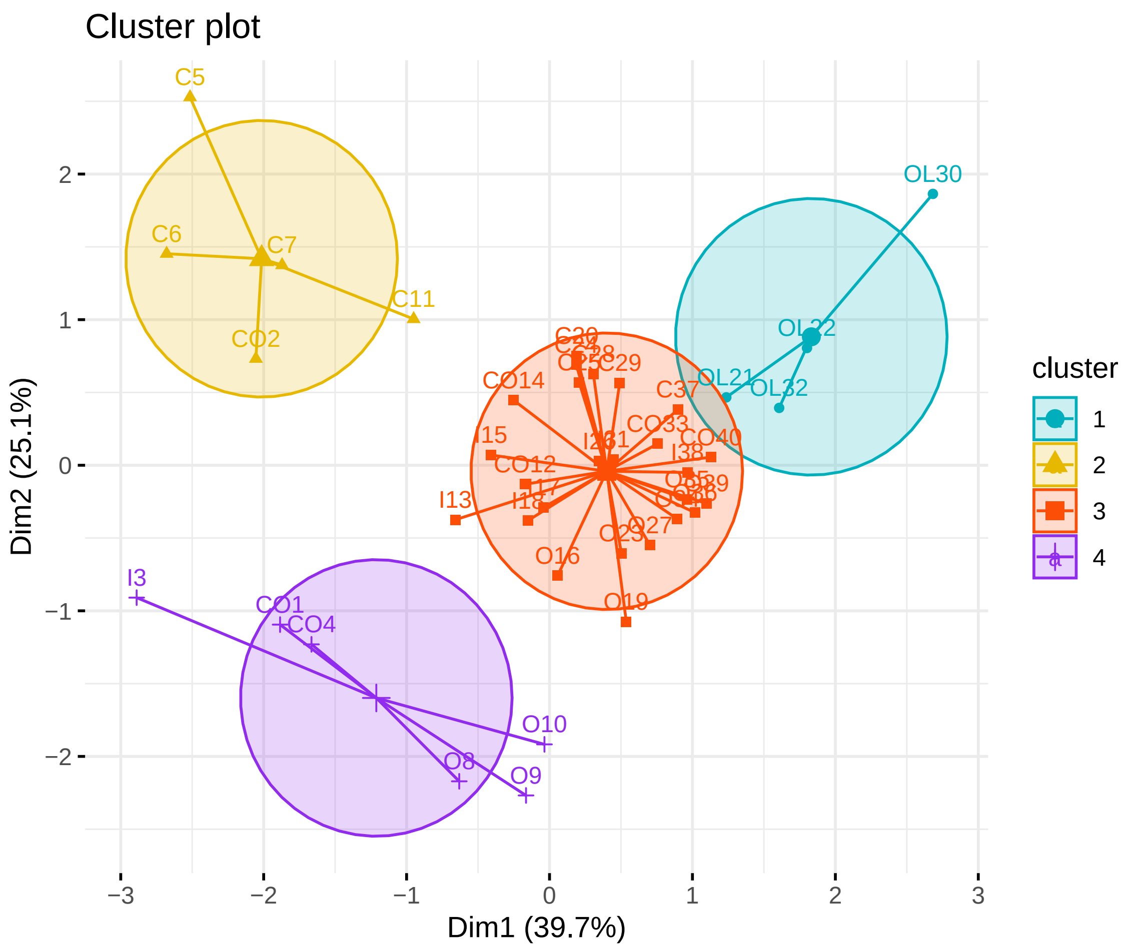
```
##
## components of output object are:
```

```r
ls(cities_clos_kmeans)
```

```
## [1] "betweenss"    "centers"      "cluster"      "ifault"       "iter"
## [6] "size"         "tot.withinss" "totss"        "withinss"
```

plot kmeans clusters closed

```r
row.names(data0) <- paste0(data0$sType,seq(1,NROW(data0)))
fviz_cluster(cities_clos_kmeans, data = data0[,2:NCOL(data0)],
             palette = c("#00AFBB", "#E7B800", "#FC4E07","purple2"),
             labelsize=10,
             ellipse.type = "euclid", # Concentration ellipse
             star.plot = TRUE, # Add segments from centroids to items
             repel = F, # if true avoids label overplotting (slow)
             ggtheme = theme_minimal()
             )
```



compute K-means open

```r
data0 <- na.omit(cities_clr[,c("sType","Compact","Open","Lightweight","Industry")])
data0[,c("Compact","Open","Lightweight","Industry")] <- scale(data0[,c("Compact","Open","Lightweight","
set.seed(123)
cities_open_kmeans <- kmeans(data0[,2:NCOL(data0)], 4, nstart = 25)
cat("K-means clustering with",length(cities_open_kmeans$size),"clusters of sizes",cities_open_kmeans$si
```

```
## K-means clustering with 4 clusters of sizes 1 10 4 25
```

```r
cat("\ncomponents of output object are:\n")
```

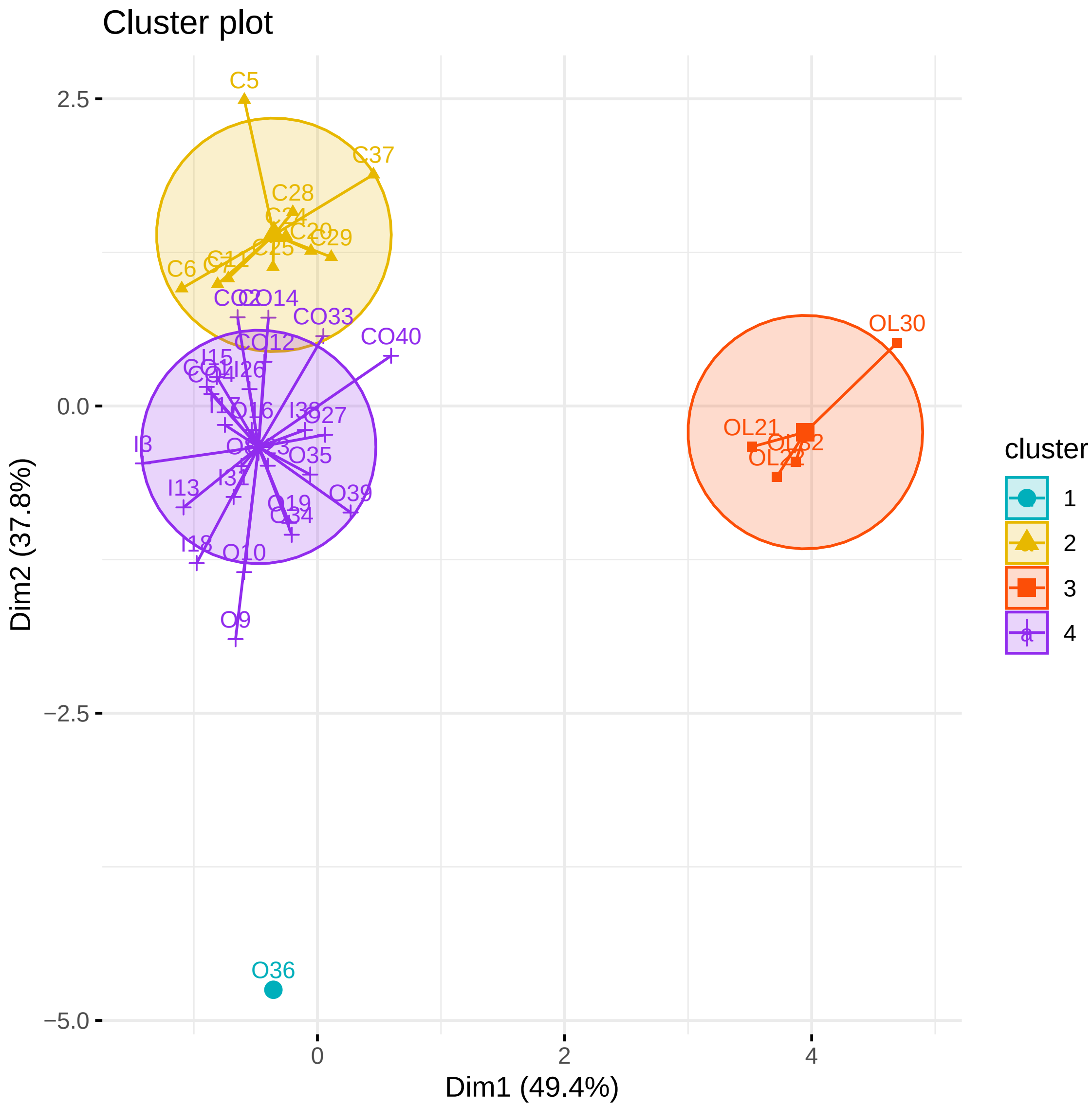
```
##
## components of output object are:
```

```r
ls(cities_open_kmeans)
```

```
## [1] "betweenss"    "centers"      "cluster"      "ifault"       "iter"
## [6] "size"         "tot.withinss" "totss"        "withinss"
```

plot kmeans clusters open

```r
row.names(data0) <- paste0(data0$sType,seq(1,NROW(data0)))
fviz_cluster(cities_open_kmeans, data = data0[,2:NCOL(data0)],
             palette = c("#00AFBB", "#E7B800", "#FC4E07","purple2"),
             labelsize=10,
             ellipse.type = "euclid", # Concentration ellipse
             star.plot = TRUE, # Add segments from centroids to items
             repel = F, # if true avoids label overplotting (slow)
             ggtheme = theme_minimal()
             )
```



hierarchical clustering

create dissimilarity (distance) matrix

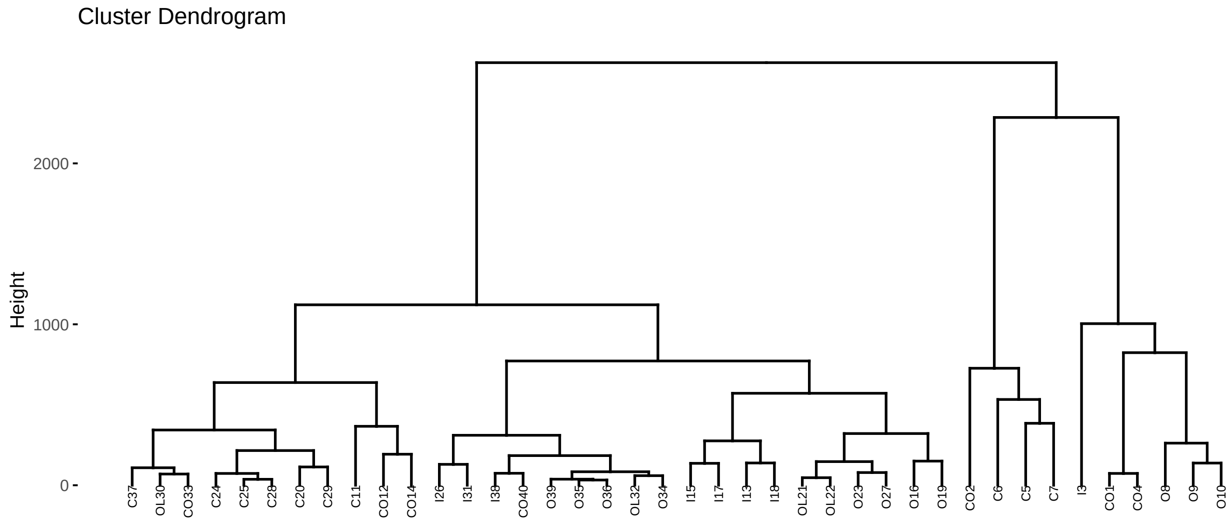
```
dataHC <- na.omit(cities[,c("sType","Compact","Open","Lightweight","Industry")])
row.names(dataHC) <- paste0(dataHC$sType, seq(1,NROW(dataHC)))
dataHC$sType <- NULL
cities_clos_diss <- get_dist(dataHC, method = "euclidean")
as.matrix(cities_clos_diss)[1:8, 1:8]
```

```
##             CO1       CO2        I3        CO4        C5        C6        C7
## CO1     0.00000  670.3172  798.3005   73.45455 1200.6492  958.6454  845.4384
## CO2   670.31720    0.0000 1220.9442  718.57846  681.8654  788.5676  431.4067
## I3    798.30049 1220.9442    0.0000  786.22008 1384.2567  868.7446 1113.4086
## CO4    73.45455  718.5785  786.2201    0.00000 1238.2498  981.7671  874.6789
## C5   1200.64924  681.8654 1384.2567 1238.24981    0.0000  554.8134  384.9671
## C6    958.64545  788.5676  868.7446  981.76710  554.8134    0.0000  438.4932
## C7    845.43844  431.4067 1113.4086  874.67891  384.9671  438.4932    0.0000
## O8    469.61856 1070.0129  850.0454  397.95442 1549.2668 1239.3503 1168.6747
##              O8
```

```
## CO1  469.6186
## CO2 1070.0129
## I3   850.0454
## CO4  397.9544
## C5  1549.2668
## C6  1239.3503
## C7  1168.6747
## O8     0.0000
```

perform hierarchical clustering

```
cities_clos_hc <- hclust(cities_clos_diss, method = "ward.D2")
require(factoextra)
fviz_dend(cities_clos_hc, cex = 0.5)
```



assess cluster tree

```
cities_clos_coph <- cophenetic(cities_clos_hc)
cor(cities_clos_diss,cities_clos_coph)
```

```
## [1] 0.8871279
```

```
cat("\nRule-of-thumb: cluster tree represents actual distance matrix accurately enough if r>0.75\n")
```

```
##
## Rule-of-thumb: cluster tree represents actual distance matrix accurately enough if r>0.75
```

cut dendrogram into different groups

```
cities_clos_grp <- cutree(cities_clos_hc, k = 5)
cities_clos_grp ; cat("\n")
```

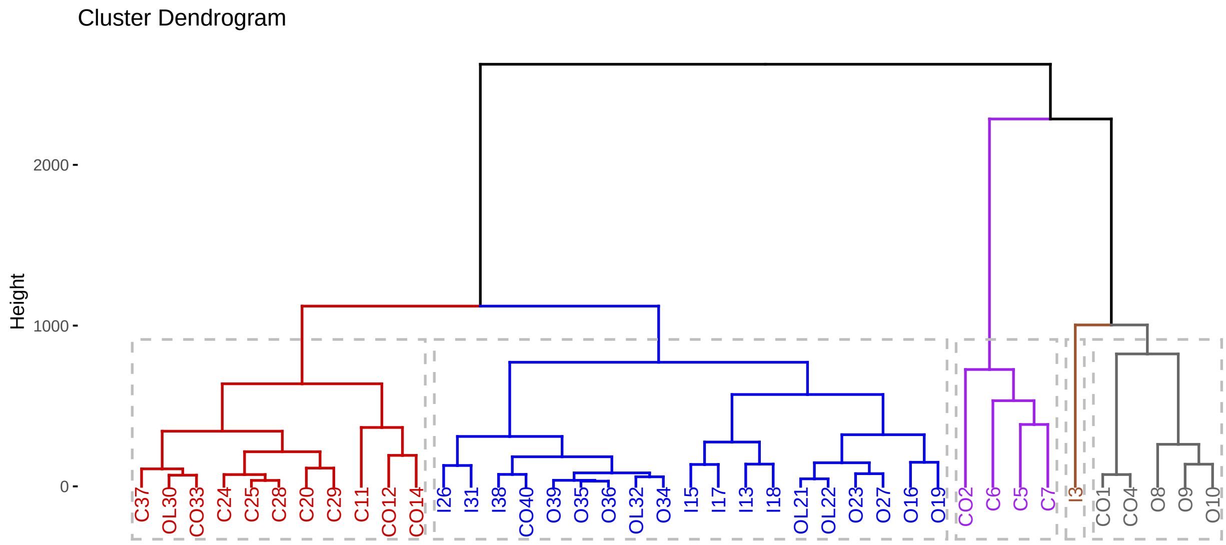
```
##   CO1   CO2    I3   CO4    C5    C6    C7    O8    O9   O10   C11  CO12   I13  CO14   I15   O16
##     1     2     3     1     2     2     2     1     1     1     4     4     5     4     5     5
##   I17   I18   O19   C20  OL21  OL22   O23   C24   C25   I26   O27   C28   C29  OL30   I31  OL32
##     5     5     5     4     5     5     5     4     4     5     5     4     4     4     5     5
##  CO33   O34   O35   O36   C37   I38   O39  CO40
##     4     5     5     5     4     5     5     5
```

```
table(cities_clos_grp)
```

```
## cities_clos_grp
##  1  2  3  4  5
##  5  4  1 11 19
```

plot dendrogram with cuts

```
fviz_dend(cities_clos_hc, k = 5, # Cut in five groups
          cex = 0.75, # label size
          k_colors = c("red3", "blue2", "purple","sienna","grey40"),
          color_labels_by_k = TRUE, # color labels by groups
          rect = TRUE # Add rectangle around groups
          )
```



hierarchical clustering (CLR-open data)

create dissimilarity (distance) matrix

```
dataHC <- na.omit(cities_clr[,c("sType","Compact","Open","Lightweight","Industry")])
row.names(dataHC) <- paste0(dataHC$sType, seq(1,NROW(dataHC)))
dataHC$Type <- NULL
cities_open_diss <- get_dist(dataHC, method = "euclidean")
as.matrix(cities_open_diss)[1:8, 1:8]
```

```
##           CO1       CO2       I3       CO4       C5        C6        C7
## CO1 0.0000000 0.9738808 1.901676 0.1189296 4.704880 2.0601631 1.5618557
## CO2 0.9738808 0.0000000 2.711537 1.0428290 4.164463 1.9260999 1.1239095
```

```
## I3  1.9016758 2.7115365 0.000000 1.8884099 5.000641 2.3041675 2.5280750
## CO4 0.1189296 1.0428290 1.888410 0.0000000 4.783363 2.1425696 1.6486186
## C5  4.7048796 4.1644632 5.000641 4.7833625 0.000000 2.8330977 3.1589439
## C6  2.0601631 1.9260999 2.304168 2.1425696 2.833098 0.0000000 0.8677997
## C7  1.5618557 1.1239095 2.528075 1.6486186 3.158944 0.8677997 0.0000000
## O8  1.2082891 1.9973093 1.935856 1.0912449 5.596906 2.9898355 2.5929279
##           O8
## CO1 1.208289
## CO2 1.997309
## I3  1.935856
## CO4 1.091245
## C5  5.596906
## C6  2.989835
## C7  2.592928
## O8  0.000000
```

perform hierarchical clustering

```r
cities_open_hc <- hclust(cities_open_diss, method = "ward.D2")
require(factoextra)
fviz_dend(cities_open_hc, cex = 0.5)
```

assess cluster tree

```r
cities_open_coph <- cophenetic(cities_open_hc)
cor(cities_open_diss,cities_open_coph)
```

```
## [1] 0.7820011
```

```r
cat("\nRule-of-thumb: cluster tree represents actual distance matrix accurately enough if r>0.75\n")
```

```
##
## Rule-of-thumb: cluster tree represents actual distance matrix accurately enough if r>0.75
```

cut dendrogram into different groups

```
cities_open_grp <- cutree(cities_open_hc, k = 5)
cities_open_grp ; cat("\n")
```

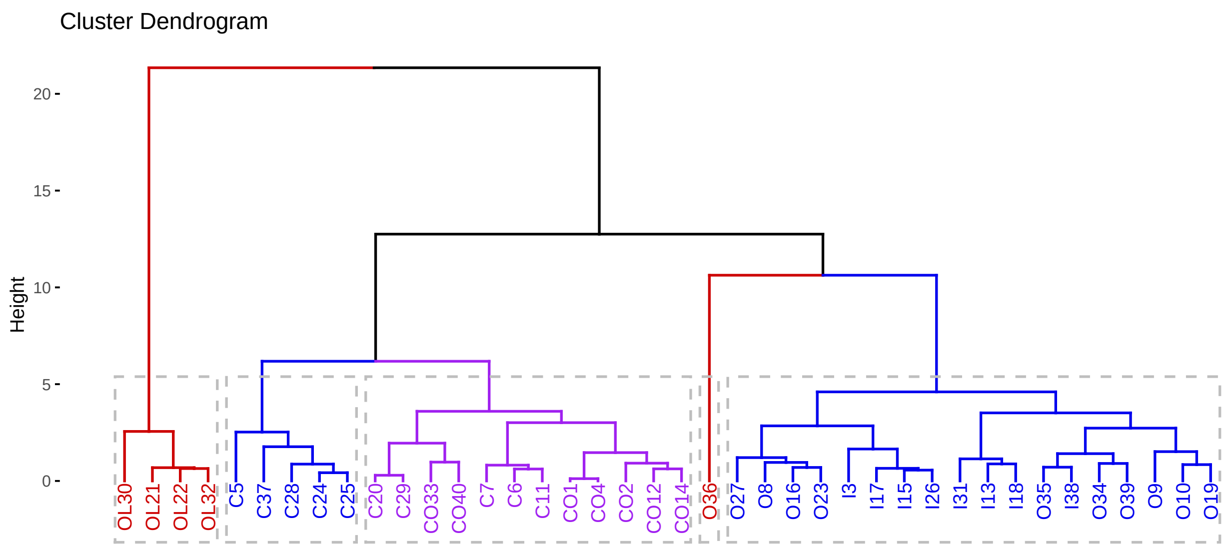
```
##   CO1  CO2   I3  CO4   C5   C6   C7   O8   O9  O10  C11 CO12  I13 CO14  I15  O16
##     1    1    2    1    3    1    1    2    2    2    1    1    2    1    2    2
##   I17  I18  O19  C20 OL21 OL22  O23  C24  C25  I26  O27  C28  C29 OL30  I31 OL32
##     2    2    2    1    4    4    2    3    3    2    2    3    1    4    2    4
## CO33  O34  O35  O36  C37  I38  O39 CO40
##     1    2    2    5    3    2    2    1
```

```
table(cities_open_grp)
```

```
## cities_open_grp
##  1  2  3  4  5
## 12 18  5  4  1
```

plot dendrogram with cuts

```
fviz_dend(cities_open_hc, k = 5, # Cut in five groups
          cex = 0.75, # label size
          k_colors = c("red3", "blue2", "purple"),
          color_labels_by_k = TRUE, # color labels by groups
          rect = TRUE # Add rectangle around groups
)
```



LDA linear discriminant analysis

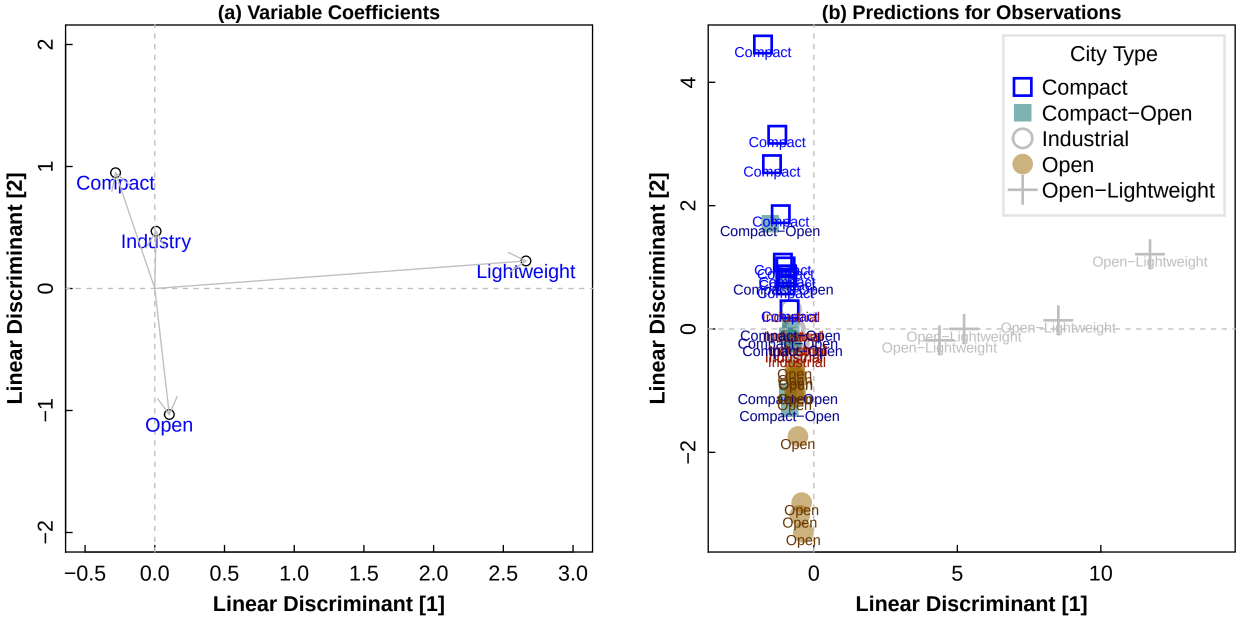
```
data0 <- cities
data0[,c("Compact", "Open", "Lightweight", "Industry")] <- scale(data0[,c("Compact", "Open", "Lightweig
lda_cities_clos <- lda(formula = Type ~ Compact + Open + Lightweight + Industry,
                       data=data0,
                       prior=as.numeric(summary(cities$Type))/
                         nrow(cities))
print(lda_cities_clos)
```

```
## Call:
## lda(Type ~ Compact + Open + Lightweight + Industry, data = data0,
##     prior = as.numeric(summary(cities$Type))/nrow(cities))
##
## Prior probabilities of groups:
##          Compact     Compact-Open       Industrial             Open
##            0.250            0.175            0.200            0.275
## Open-Lightweight
##            0.100
##
## Group means:
##                     Compact        Open Lightweight   Industry
## Compact           0.8681624 -0.76749804   -0.307629  0.3544503
## Compact-Open      0.7215955  0.53760420   -0.307629 -0.1829727
## Industrial       -0.5610637 -0.07062059   -0.307629  0.7992610
## Open             -0.6703995  0.53491974   -0.307629 -0.5914479
## Open-Lightweight -0.4674720 -0.35185037    2.768661 -0.5379637
##
## Coefficients of linear discriminants:
##                      LD1        LD2         LD3       LD4
## Compact     -0.281363312  0.9489128 -1.22823850 0.2266807
## Open         0.104056051 -1.0325030 -0.58447411 0.6372860
## Lightweight  2.662403231  0.2259667 -0.06228637 0.2228803
## Industry     0.009172738  0.4693317  1.32610142 0.5290885
##
## Proportion of trace:
##    LD1    LD2    LD3    LD4
## 0.7160 0.1602 0.1186 0.0051
```

```r
par(mfrow = c(1,2), mar = c(3.5,3.5,1,1), mgp = c(1.5,0.3,0), tcl = 0.25,
    lend = "square", ljoin = "mitre", cex.main = 0.9, font.lab=2)
plot(lda_cities_clos$scaling[,1], lda_cities_clos$scaling[,2],
     xlim = c(-0.5,3), ylim=c(-2,2),
     xlab="Linear Discriminant [1]", ylab="Linear Discriminant [2]", main="(a) Variable Coefficients")
abline(v=0,col="grey",lty=2)
abline(h=0,col="grey",lty=2)
text(lda_cities_clos$scaling[,1],lda_cities_clos$scaling[,2],
     labels=c("Compact", "Open", "Lightweight","Industry"),
     pos=1,cex=0.95,col="blue2",offset=0.2)

ldaPred_cities_clos <- predict(lda_cities_clos)
for(i in 1:NROW(lda_cities_clos$scaling)){
  arrows(0,0,lda_cities_clos$scaling[i,1],lda_cities_clos$scaling[i,2],
         length = 0.15, col = 8)
}
plot(ldaPred_cities_clos$x[,1], ldaPred_cities_clos$x[,2],
     col=c(2,"#00666680",8,"#99660080","grey")[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-3,14),
  lwd=c(2,1,2,1,2)[cities$Type],
  cex=c(1.8,1.8,2,2,2)[cities$Type],
  xlab="Linear Discriminant [1]", ylab="Linear Discriminant [2]",
  main="(b) Predictions for Observations")
abline(v=0,col="grey",lty=2)
abline(h=0,col="grey",lty=2)
text(ldaPred_cities_clos$x[,1], ldaPred_cities_clos$x[,2],
     labels=cities$Type, col=c(2,"blue4","#991100","#663300",8)[cities$Type],
     pos=1, offset=0.15, cex=0.65)
legend("topright",legend=levels(cities$Type),
       col=c(2,"#00666680",8,"#99660080","grey"),
       pch=c(0,15,1,19,3), pt.lwd=c(2,1,2,1,2),
       title="City Type",bty="o", box.col="grey90",
```

```
        box.lwd=2, inset=0.02,
        pt.cex=c(1.8,1.8,2,2,2), cex=1.)
```

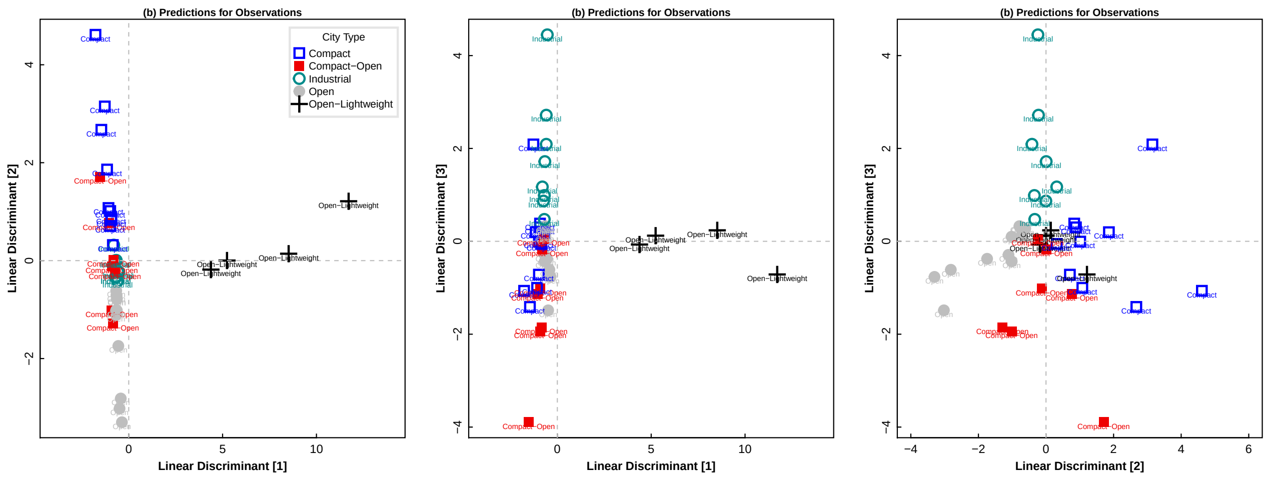



```
par(mfrow = c(1,3), mar = c(3.5,3.5,1,1), mgp = c(1.5,0.3,0), tcl = 0.25,
    lend = "square", ljoin = "mitre", cex.main = 0.9, font.lab=2)

plot(ldaPred_cities_clos$x[,1], ldaPred_cities_clos$x[,2], col=c(2,4,6,8,1)[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-4,14),lwd=c(2,1,2,1,2)[cities$Type],
     cex=c(1.8,1.8,2,2,2)[cities$Type], main="(b) Predictions for Observations",
     xlab="Linear Discriminant [1]", ylab="Linear Discriminant [2]")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(ldaPred_cities_clos$x[,1], ldaPred_cities_clos$x[,2], labels=cities$Type,
     col=c(2,4,6,8,1)[cities$Type], pos=1, offset=0.15, cex=0.65)
legend("topright", legend=levels(cities$Type), col=c(2,4,6,8,1),
       pch=c(0,15,1,19,3), pt.lwd=c(2,1,2,1,2),
       title="City Type",bty="o", box.col="grey90",
       box.lwd=2, inset=0.02, pt.cex=c(1.8,1.8,2,2,2), cex=0.9)

plot(ldaPred_cities_clos$x[,1], ldaPred_cities_clos$x[,3], col=c(2,4,6,8,1)[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-4,14),lwd=c(2,1,2,1,2)[cities$Type],
     cex=c(1.8,1.8,2,2,2)[cities$Type], main="(b) Predictions for Observations",
     xlab="Linear Discriminant [1]", ylab="Linear Discriminant [3]")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(ldaPred_cities_clos$x[,1], ldaPred_cities_clos$x[,3], labels=cities$Type,
     col=c(2,4,6,8,1)[cities$Type], pos=1, offset=0.15, cex=0.65)

plot(ldaPred_cities_clos$x[,2], ldaPred_cities_clos$x[,3], col=c(2,4,6,8,1)[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-4,6),lwd=c(2,1,2,1,2)[cities$Type],
     cex=c(1.8,1.8,2,2,2)[cities$Type], main="(b) Predictions for Observations",
     xlab="Linear Discriminant [2]", ylab="Linear Discriminant [3]")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(ldaPred_cities_clos$x[,2], ldaPred_cities_clos$x[,3], labels=cities$Type,
     col=c(2,4,6,8,1)[cities$Type], pos=1, offset=0.15, cex=0.65)
```




```r
par(mfrow=c(1,1))
```

```r
data0 <- cities_clr
data0[,c("Compact", "Open", "Lightweight", "Industry")] <- scale(data0[,c("Compact", "Open", "Lightweig
lda_cities_open <- lda(formula = Type ~ Compact + Open + Lightweight + Industry,
                       data=data0,
                       prior=as.numeric(summary(data0$Type))/nrow(data0))
```

```
## Warning in lda.default(x, grouping, ...): variables are collinear
```

```r
print(lda_cities_open)
```

```
## Call:
## lda(Type ~ Compact + Open + Lightweight + Industry, data = data0,
##     prior = as.numeric(summary(data0$Type))/nrow(data0))
##
## Prior probabilities of groups:
##           Compact      Compact-Open        Industrial              Open
##             0.250             0.175             0.200             0.275
## Open-Lightweight
##             0.100
##
## Group means:
##                     Compact       Open Lightweight      Industry
## Compact           0.9635293 -1.0288909  -0.3534227  0.453428934
## Compact-Open      0.6373470  0.2644211  -0.5097742 -0.340414826
## Industrial       -0.2963007  0.2754019  -0.3542026  0.858027234
## Open             -0.6914142  1.0562017  -0.1049318  0.001797854
## Open-Lightweight -1.0301901 -1.3458683   2.7726291 -2.258844954
##
## Coefficients of linear discriminants:
##                     LD1        LD2        LD3
## Compact      0.27186685 -0.2987298 -0.5686536
## Open        -0.08394227  2.0140469 -0.1492896
## Lightweight -2.32468223 -0.3122192  0.6871651
## Industry     1.28825453 -0.9427297  1.1514474
##
## Proportion of trace:
##    LD1    LD2    LD3
## 0.7283 0.2448 0.0269
```

```r
cormat <- rcorr.adjust(data0[,c("Compact", "Open", "Lightweight", "Industry")], type="pearson")
cat("\nCorrelation matrix for predictors:\n"); print(cormat$R$r, digits = 3)
```

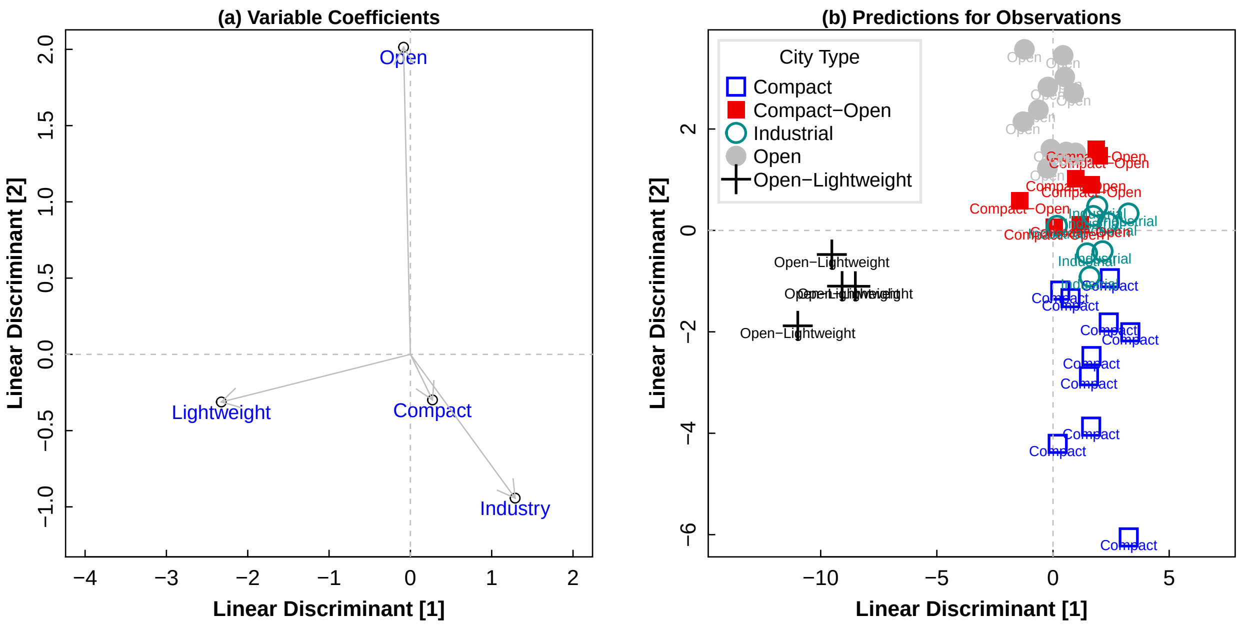
```
##
## Correlation matrix for predictors:

##             Compact   Open Lightweight Industry
## Compact       1.000 -0.478      -0.519    0.029
## Open         -0.478  1.000      -0.387    0.298
## Lightweight  -0.519 -0.387       1.000   -0.699
## Industry      0.029  0.298      -0.699    1.000
```

```r
rm(cormat)
```

```r
par(mfrow = c(1,2), mar = c(3.5,3.5,1,1), mgp = c(1.5,0.3,0), tcl = 0.25,
    lend = "square", ljoin = "mitre", cex.main = 0.9, font.lab=2)
plot(lda_cities_open$scaling[,1], lda_cities_open$scaling[,2], xlim = c(-4,2), ylim = c(-1.2,2),
     xlab="Linear Discriminant [1]", ylab="Linear Discriminant [2]", main="(a) Variable Coefficients")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(lda_cities_open$scaling[,1], lda_cities_open$scaling[,2], labels=names(cities)[2:5],
     pos=1,cex=0.95,col="blue2",offset=0.2)
for(i in 1:NROW(lda_cities_open$scaling)){
  arrows(0,0,lda_cities_open$scaling[i,1],lda_cities_open$scaling[i,2],
         length = 0.15, col = 8)
}
ldaPred_cities_open <- predict(lda_cities_open)

plot(ldaPred_cities_open$x[,1], ldaPred_cities_open$x[,2], col=c(2,4,6,8,1)[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-14,7),lwd=c(2,1,2,1,2)[cities$Type],
     cex=c(1.8,1.8,2,2,2)[cities$Type], main="(b) Predictions for Observations",
     xlab="Linear Discriminant [1]", ylab="Linear Discriminant [2]")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(ldaPred_cities_open$x[,1], ldaPred_cities_open$x[,2], labels=cities$Type,
     col=c(2,4,6,8,1)[cities$Type], pos=1, offset=0.15, cex=0.65)
legend("topleft", legend=levels(cities$Type), col=c(2,4,6,8,1),
       pch=c(0,15,1,19,3), pt.lwd=c(2,1,2,1,2),
       title="City Type",bty="o", box.col="grey90",
       box.lwd=2, inset=0.02, pt.cex=c(1.8,1.8,2,2,2), cex=0.9)
```



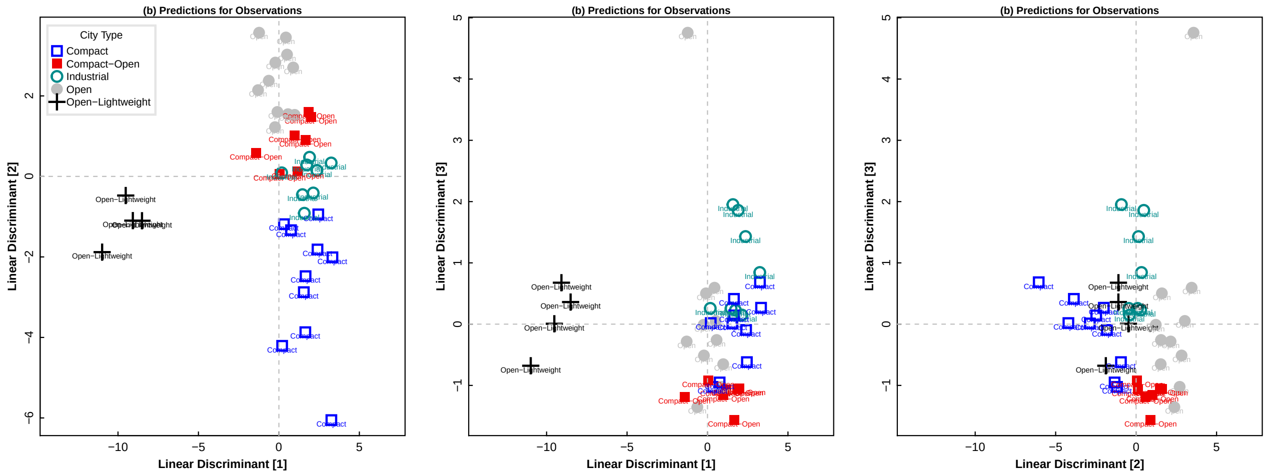
```r
par(mfrow=c(1,1))
```

```r
par(mfrow = c(1,3), mar = c(3.5,3.5,1,1), mgp = c(1.5,0.3,0), tcl = 0.25,
    lend = "square", ljoin = "mitre", cex.main = 0.9, font.lab=2)

plot(ldaPred_cities_open$x[,1], ldaPred_cities_open$x[,2], col=c(2,4,6,8,1)[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-14,7),lwd=c(2,1,2,1,2)[cities$Type],
     cex=c(1.8,1.8,2,2,2)[cities$Type], main="(b) Predictions for Observations",
     xlab="Linear Discriminant [1]", ylab="Linear Discriminant [2]")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(ldaPred_cities_open$x[,1], ldaPred_cities_open$x[,2], labels=cities$Type,
     col=c(2,4,6,8,1)[cities$Type], pos=1, offset=0.15, cex=0.65)
legend("topleft", legend=levels(cities$Type), col=c(2,4,6,8,1),
       pch=c(0,15,1,19,3), pt.lwd=c(2,1,2,1,2),
       title="City Type",bty="o", box.col="grey90",
       box.lwd=2, inset=0.02, pt.cex=c(1.8,1.8,2,2,2), cex=0.9)

plot(ldaPred_cities_open$x[,1], ldaPred_cities_open$x[,3], col=c(2,4,6,8,1)[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-14,7),lwd=c(2,1,2,1,2)[cities$Type],
     cex=c(1.8,1.8,2,2,2)[cities$Type], main="(b) Predictions for Observations",
     xlab="Linear Discriminant [1]", ylab="Linear Discriminant [3]")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(ldaPred_cities_open$x[,1], ldaPred_cities_open$x[,3], labels=cities$Type,
     col=c(2,4,6,8,1)[cities$Type], pos=1, offset=0.15, cex=0.65)

plot(ldaPred_cities_open$x[,2], ldaPred_cities_open$x[,3], col=c(2,4,6,8,1)[cities$Type],
     pch=c(0,15,1,19,3)[cities$Type], xlim = c(-14,7),lwd=c(2,1,2,1,2)[cities$Type],
     cex=c(1.8,1.8,2,2,2)[cities$Type], main="(b) Predictions for Observations",
     xlab="Linear Discriminant [2]", ylab="Linear Discriminant [3]")
abline(v=0,col="grey",lty=2); abline(h=0,col="grey",lty=2)
text(ldaPred_cities_open$x[,2], ldaPred_cities_open$x[,3], labels=cities$Type,
     col=c(2,4,6,8,1)[cities$Type], pos=1, offset=0.15, cex=0.65)
```




```r
par(mfrow=c(1,1))
```

```r
closComp <- as.data.frame(cbind(as.character(cities_clr$Type),
                                as.character(ldaPred_cities_clos$class)))
colnames(closComp) <- c("=-Actual-=","=-Predicted-=")
closComp$test <- as.character(cities_clr$Type) == as.character(ldaPred_cities_clos$class)
k = length(which(closComp$test == TRUE))
cat("Predictions by LDA using closed data:",k,"out of",NROW(cities_clr),"=",paste0(100*k/NROW(cities_cl
```

```
## Predictions by LDA using closed data: 37 out of 40 = 92.5% correct
```

```r
closComp
```

```
##          =-Actual-=    =-Predicted-=  test
## 1      Compact-Open     Compact-Open  TRUE
## 2      Compact-Open     Compact-Open  TRUE
## 3        Industrial       Industrial  TRUE
## 4      Compact-Open     Compact-Open  TRUE
## 5           Compact          Compact  TRUE
## 6           Compact          Compact  TRUE
## 7           Compact          Compact  TRUE
## 8              Open             Open  TRUE
## 9              Open             Open  TRUE
## 10             Open             Open  TRUE
## 11          Compact          Compact  TRUE
## 12     Compact-Open     Compact-Open  TRUE
## 13       Industrial       Industrial  TRUE
## 14     Compact-Open     Compact-Open  TRUE
## 15       Industrial       Industrial  TRUE
## 16             Open             Open  TRUE
## 17       Industrial       Industrial  TRUE
## 18       Industrial       Industrial  TRUE
## 19             Open             Open  TRUE
## 20          Compact          Compact  TRUE
## 21 Open-Lightweight Open-Lightweight  TRUE
## 22 Open-Lightweight Open-Lightweight  TRUE
## 23             Open             Open  TRUE
## 24          Compact          Compact  TRUE
## 25          Compact          Compact  TRUE
## 26       Industrial       Industrial  TRUE
## 27             Open             Open  TRUE
## 28          Compact          Compact  TRUE
## 29          Compact          Compact  TRUE
## 30 Open-Lightweight Open-Lightweight  TRUE
## 31       Industrial       Industrial  TRUE
## 32 Open-Lightweight Open-Lightweight  TRUE
## 33     Compact-Open             Open FALSE
## 34             Open             Open  TRUE
## 35             Open             Open  TRUE
## 36             Open             Open  TRUE
## 37          Compact          Compact  TRUE
## 38       Industrial             Open FALSE
## 39             Open             Open  TRUE
## 40     Compact-Open             Open FALSE
```

```r
openComp <- as.data.frame(cbind(as.character(cities_clr$Type), as.character(ldaPred_cities_open$class)))
colnames(openComp) <- c("=-Actual-=","=-Predicted-=")
openComp$test <- as.character(cities_clr$Type) == as.character(ldaPred_cities_open$class)
k = length(which(openComp$test == TRUE))
cat("\nPredictions by LDA using open data:",k,"out of",NROW(cities_clr),"=",paste0(100*k/NROW(cities_cl
```

```
##
## Predictions by LDA using open data: 38 out of 40 = 95% correct
```

```r
openComp
```

```
##          =-Actual-=    =-Predicted-=  test
## 1      Compact-Open     Compact-Open  TRUE
## 2      Compact-Open     Compact-Open  TRUE
```

```
## 3        Industrial       Industrial  TRUE
## 4      Compact-Open     Compact-Open  TRUE
## 5           Compact          Compact  TRUE
## 6           Compact          Compact  TRUE
## 7           Compact          Compact  TRUE
## 8              Open             Open  TRUE
## 9              Open             Open  TRUE
## 10             Open             Open  TRUE
## 11          Compact          Compact  TRUE
## 12     Compact-Open     Compact-Open  TRUE
## 13       Industrial       Industrial  TRUE
## 14     Compact-Open     Compact-Open  TRUE
## 15       Industrial       Industrial  TRUE
## 16             Open     Compact-Open FALSE
## 17       Industrial       Industrial  TRUE
## 18       Industrial       Industrial  TRUE
## 19             Open             Open  TRUE
## 20          Compact          Compact  TRUE
## 21 Open-Lightweight Open-Lightweight  TRUE
## 22 Open-Lightweight Open-Lightweight  TRUE
## 23             Open             Open  TRUE
## 24          Compact          Compact  TRUE
## 25          Compact          Compact  TRUE
## 26       Industrial       Industrial  TRUE
## 27             Open             Open  TRUE
## 28          Compact          Compact  TRUE
## 29          Compact     Compact-Open FALSE
## 30 Open-Lightweight Open-Lightweight  TRUE
## 31       Industrial       Industrial  TRUE
## 32 Open-Lightweight Open-Lightweight  TRUE
## 33     Compact-Open     Compact-Open  TRUE
## 34             Open             Open  TRUE
## 35             Open             Open  TRUE
## 36             Open             Open  TRUE
## 37          Compact          Compact  TRUE
## 38       Industrial       Industrial  TRUE
## 39             Open             Open  TRUE
## 40     Compact-Open     Compact-Open  TRUE
```

```
library(MASS)
library(RcmdrMisc)
library(knitr)
library(beepr)
### MATCH this to read file below
# sink(file="cities_type_pred.csv", type="output")
n0 <- 1000 # number of iterations
ftrain <- 0.75 # proportion of observations in training set
results <- data.frame(
    Rep = rep(NA, n0),
    matches = rep(NA, n0),
    non_matches = rep(NA, n0),
    success = rep(NA, n0))
# train <- sample(1:NROW(cities), round(NROW(cities)-5,0))
train <- sample(1:NROW(cities), round(NROW(cities) * ftrain,0))
lda.cities.train <- lda(formula = Type ~  Compact + Open + Lightweight + Industry,
                   data = cities[train,],
                   prior = as.numeric(summary(cities$Type[train]))/
                     nrow(cities[train,]))
lda.cities.pred <- predict(lda.cities.train, cities[-train,])
# make vector of individual category non-matches
matchByClass <- data.frame(Matches = rep(0,nlevels(cities$Type)))
```

```r
rownames(matchByClass) <- levels(lda.cities.pred$class)
e0 <- 0
# colnames(matchByClass) <- c("Matches")
for (i in 1:n0) {
  # train <- sample(1:NROW(cities), round(NROW(cities)-5,0))
  train <- sample(1:NROW(cities), round(NROW(cities) * ftrain,0))
  if (is.na(match(NA,tapply(cities[train,]$Open, cities[train,]$Type, sd, na.rm=T))) == TRUE) {
    lda.cities.train <- lda(formula = Type ~ Compact + Open + Lightweight + Industry,
                            data = cities[train,],
                            prior=as.numeric(summary(cities$Type[train]))/
                          nrow(cities[train,]))
    lda.cities.pred <- predict(lda.cities.train, cities[-train,])
    e0 <- e0 + 1
  }

  k=0 # number of non-matches
  m0 <- as.matrix(rep(0,5)) # vector of individual category non-matches
  rownames(m0) <- levels(lda.cities.pred$class)
  rownames(matchByClass) <- levels(lda.cities.pred$class)
  colnames(matchByClass) <- c("Matches")
  for (jM in 1:NROW(cities[-train,])) {
    # cat("big loop #",jM,"\n")
    for (jS in 1:nlevels(lda.cities.pred$class)) {
      if((lda.cities.pred$class[jM] == levels(lda.cities.pred$class)[jS]) &
         (cities$Type[-train][jM] == levels(lda.cities.pred$class)[jS]) )
        m0[jS] = m0[jS] + 1
      else  m0[jS] = m0[jS]
      # cat("small loop iteration #",jS,"; matching",
      #     levels(lda.cities.pred$class)[jS],
      #     "; matches =",m0,"\n")
    }
    k = sum(m0)
    # cat("medium loop iteration ",jM,"; matches = ",k,"\n", sep="")
    # if(jM==NROW(cities[-train,]))
  }
  # cat("GIANT LOOP #",i,"\n")
  matchByClass <- matchByClass + m0
  # output to results data frame: iteration, matches, non-matches, proportion matched
  results[i,] <- c(i, k, NROW(cities[-train,])-k, signif(k/NROW(cities[-train,]),3))
  # cbind(lda.cities.pred$class,cities$Type[-train])
}
matchByClass$Actual <- round(1000*as.numeric(summary(cities$Type))*
                               (NROW(cities[-train,])/NROW(cities)),0)
matchByClass$Proportion <- matchByClass$Matches/matchByClass$Actual

# sink() # close output file
### make sure read file is SAME AS SINK
beep(sound = 10)

# results <- read.csv("cities_type_pred.csv")
{cat("[Based on", n0, "random subsets of dataset to",
    "train LDA model to\npredict remaining observations]\n")
cat("Number of obs. in random subsets =",NROW(train),
    " (predicting",NROW(cities)-NROW(train),"samples)\n")
print(numSummary(results[,2:4], statistics=c("mean","sd"))$table)
ns0 <- numSummary(results$success)
t0 <- t.test(results$success)
cat(rep("-\u2013-",24),
    "\nStat. summary for 'success':\nMean = ",round(ns0$table[1],4),
    ", sd = ",round(ns0$table[2],4),", 95% confidence interval = (",
```

```r
    signif(t0$conf.int[1],3),", ",signif(t0$conf.int[2],4),") (after ",i," reps)\n", sep="")
cat(n0-e0,"iterations failed due to random sampling missing a group\n\n")
print(matchByClass, digits=3)}
```

```
## [Based on 1000 random subsets of dataset to train LDA model to
## predict remaining observations]
## Number of obs. in random subsets = 30  (predicting 10 samples)
##               mean        sd
## matches     7.7530 1.6949525
## non_matches 2.2470 1.6949525
## success     0.7753 0.1694952
## ---------------------------------------------------------------------------
## Stat. summary for 'success':
## Mean = 0.7753, sd = 0.1695, 95% confidence interval = (0.765, 0.7858) (after 1000 reps)
## 34 iterations failed due to random sampling missing a group
##
##                   Matches Actual Proportion
## Compact              1972   2500      0.789
## Compact-Open          698   1750      0.399
## Industrial           1653   2000      0.827
## Open                 2559   2750      0.931
## Open-Lightweight      871   1000      0.871
```

```r
rm(list = c("n0","ftrain","i","e0","jS","jM","k","m0","matchByClass","results","train"))
```

```r
n0 <- 1000 # number of iterations
ftrain <- 0.75 # proportion of observations in training set
results <- data.frame(
  Rep = rep(NA, n0),
  matches = rep(NA, n0),
  non_matches = rep(NA, n0),
  success = rep(NA, n0))
# train <- sample(1:NROW(cities_clr), round(NROW(cities_clr)-5,0))
train <- sample(1:NROW(cities_clr), round(NROW(cities_clr) * ftrain,0))
lda.cities_clr.train <- lda(formula = Type ~  Compact + Open + Lightweight + Industry,
                        data = cities_clr[train,],
                        prior = as.numeric(summary(cities_clr$Type[train]))/
                          nrow(cities_clr[train,]))
lda.cities_clr.pred <- predict(lda.cities_clr.train, cities_clr[-train,])
# make vector of individual category non-matches
matchByClass <- data.frame(Matches = rep(0,nlevels(cities_clr$Type)))
rownames(matchByClass) <- levels(lda.cities_clr.pred$class)
e0 <- 0
# colnames(matchByClass) <- c("Matches")
for (i in 1:n0) {
  # train <- sample(1:NROW(cities_clr), round(NROW(cities_clr)-5,0))
  train <- sample(1:NROW(cities_clr), round(NROW(cities_clr) * ftrain,0))
  if (is.na(match(NA,tapply(cities_clr[train,]$Open, cities_clr[train,]$Type, sd, na.rm=T))) == TRUE) {
    lda.cities_clr.train <- lda(formula = Type ~ Compact + Open + Lightweight + Industry,
                            data = cities_clr[train,],
                            prior=as.numeric(summary(cities_clr$Type[train]))/
                              nrow(cities_clr[train,]))
    lda.cities_clr.pred <- predict(lda.cities_clr.train, cities_clr[-train,])
    e0 <- e0 + 1
  }

  k=0 # number of non-matches
  m0 <- as.matrix(rep(0,5)) # vector of individual category non-matches
  rownames(m0) <- levels(lda.cities_clr.pred$class)
```

```r
  rownames(matchByClass) <- levels(lda.cities_clr.pred$class)
  colnames(matchByClass) <- c("Matches")
  for (jM in 1:NROW(cities_clr[-train,])) {
    # cat("big loop #",jM,"\n")
    for (jS in 1:nlevels(lda.cities_clr.pred$class)) {
      if((lda.cities_clr.pred$class[jM] == levels(lda.cities_clr.pred$class)[jS]) &
         (cities_clr$Type[-train][jM] == levels(lda.cities_clr.pred$class)[jS]) )
        m0[jS] = m0[jS] + 1
      else  m0[jS] = m0[jS]
      # cat("small loop iteration #",jS,"; matching",
      #     levels(lda.cities_clr.pred$class)[jS],
      #     "; matches =",m0,"\n")
    }
    k = sum(m0)
    # cat("medium loop iteration ",jM,"; matches = ",k,"\n", sep="")
    # if(jM==NROW(cities_clr[-train,]))
  }
  # cat("GIANT LOOP #",i,"\n")
  matchByClass <- matchByClass + m0
  # output to results data frame: iteration, matches, non-matches, proportion matched
  results[i,] <- c(i, k, NROW(cities_clr[-train,])-k, signif(k/NROW(cities_clr[-train,]),3))
  # cbind(lda.cities_clr.pred$class,cities_clr$Type[-train])
}
matchByClass$Actual <- round(n0*as.numeric(summary(cities_clr$Type))*
                              (NROW(cities_clr[-train,])/NROW(cities_clr)),0)
matchByClass$Proportion <- matchByClass$Matches/matchByClass$Actual

beep(sound = 10)

{cat("[Based on", n0, "random subsets of dataset to",
     "train LDA model to\npredict remaining observations]\n")
  cat("Number of obs. in random subsets =",NROW(train),
      " (predicting",NROW(cities_clr)-NROW(train),"samples)\n")
  print(numSummary(results[,2:4], statistics=c("mean","sd"))$table)
  ns0 <- numSummary(results$success)
  t0 <- t.test(results$success)
  cat(rep("-\u2013-",24),
      "\nStat. summary for 'success':\nMean = ",round(ns0$table[1],4),
      ", sd = ",round(ns0$table[2],4),", 95% confidence interval = (",
      signif(t0$conf.int[1],3),", ",signif(t0$conf.int[2],4),") (after ",i," reps)\n", sep="")
  cat(n0-e0,"iterations failed due to random sampling missing a group\n\n")
  print(matchByClass, digits=3)}
```

```
## [Based on 1000 random subsets of dataset to train LDA model to
## predict remaining observations]
## Number of obs. in random subsets = 30  (predicting 10 samples)
##                 mean        sd
## matches     8.0980 1.7656482
## non_matches 1.9020 1.7656482
## success     0.8098 0.1765648
## -------------------------------------------------------------------------
## Stat. summary for 'success':
## Mean = 0.8098, sd = 0.1766, 95% confidence interval = (0.799, 0.8208) (after 1000 reps)
## 49 iterations failed due to random sampling missing a group
##
##                   Matches Actual Proportion
## Compact              1950   2500      0.780
## Compact-Open         1416   1750      0.809
## Industrial           1677   2000      0.839
## Open                 2198   2750      0.799
```

```
## Open-Lightweight     857   1000      0.857
```

```r
rm(list = c("n0","ftrain","i","e0","jS","jM","k","m0","matchByClass","results","train"))
```