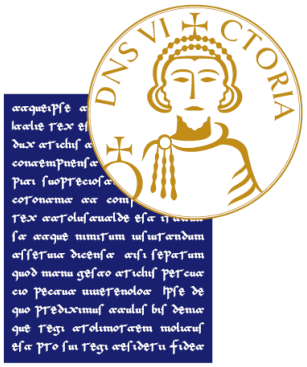


Università degli Studi del Sannio

Dipartimento di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

Data Science

Homework 2

Prof.:
Antonio Pecchia

Studenti:
Stocchetti Federico, matr. 399000543
Fioretti Gabriele, matr. 399000522
Razzano Federica, matr. 399000542

Anno Accademico 2022-2023

Indice

Introduzione

In questo elaborato viene presentato il lavoro svolto per l'esecuzione dell'Homework numero due, per il corso di Data Science. Il lavoro è diviso in quattro tasks e l'obiettivo è quello di analizzare un file di flussi di rete ed applicare correttamente metodi per il clustering, in modo da individuare quali dei record presenti nei dati identifica un attacco DDoS.

Dati di partenza

Tramite il criterio di scelta indicato nella traccia del precedente Homework, e stato selezionato il file `flows1.csv` per l'analisi. Il file è composto da 3600 record che hanno 8 features:

1. **Total Length of Bwd Packet**: Questa metrica rappresenta la dimensione totale (in byte) dei pacchetti di rete inviati dal destinatario al mittente.
2. **Flow Bytes/s**: Questa misura indica la velocità media del flusso di byte nella rete, misurata in byte al secondo.
3. **Total TCP Flow Time**: Si riferisce al tempo totale trascorso per un flusso TCP completo, dal suo inizio alla sua fine.
4. **Total Fwd Packet**: Rappresenta il conteggio totale dei pacchetti inoltrati all'interno di un flusso di dati.
5. **Fwd IAT Std**: Rappresenta la deviazione standard degli interarrivi (IAT) dei pacchetti inoltrati.
6. **Bwd Packet Length Std**: Rappresenta la deviazione standard della lunghezza dei pacchetti inoltrati dal destinatario al mittente.
7. **Flow Duration**: Questa metrica indica la durata totale del flusso di dati in rete.
8. **Average Packet Size**: Questa misura rappresenta la dimensione media dei pacchetti in un flusso di dati.

Da notare che i dati non contengono label tra le features, non permettendoci quindi di avere un approccio supervisionato al problema.

1 Task 1

1.1 Descrizione del task

Per la prima task è richiesto di applicare `hclust` e `kmeans` sui dati senza che siano eseguite operazioni preliminari su di essi. E' richiesto inoltre un corredo di diagnostica appropriato e la discussione dei risultati ottenuti.

1.2 Esecuzione

Come richiesto dalla traccia è stato eseguito il clustering senza ridurre la dimensionalità dei dati di partenza. Tuttavia, prima di eseguire le operazioni finali con Kmeans e hclsut è stata effettuata una analisi di sensitività per determinare il numero migliore di cluster in cui dividere i dati. Questa analisi è stata effettuata sulla proprietà withinss, che sta per "within-cluster sum of squares". Un valore basso di whitinss indica una buona vicinanza dei punti al centroide di ciascun cluster. L'analisi è stata effettuata nel modo seguente:

```
tss<-seq(1,10,1)
for ( i in 1:10) tss[i] <- kmeans(flows,i)$tot.withinss
plot(tss, main = "Sensibility analysis - kmeans tot.withinss", type = "o",
lwd = 1.5, xlab = "k",cex=2, cex.main = 2)
```

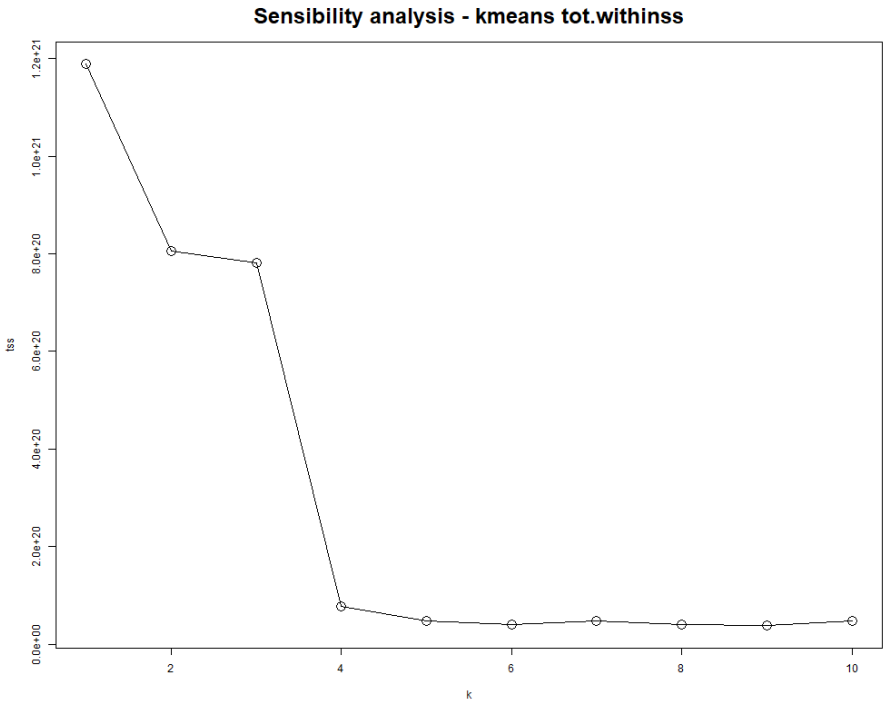


Figura 1: Sensibility analysis with whitinss

Dalla analisi di sensitività risulta evidente che il numero di cluster ottimale è k=4. Eseguiamo quindi hclust e kmeans con questo parametro.

```
d<-dist(flows,method = "euclidean")
fit<-hclust(d,method="ward.D")
plot(fit, labels = FALSE)
groups <- cutree(fit, k=4)
rect.hclust(fit, k=4, border="red")
```

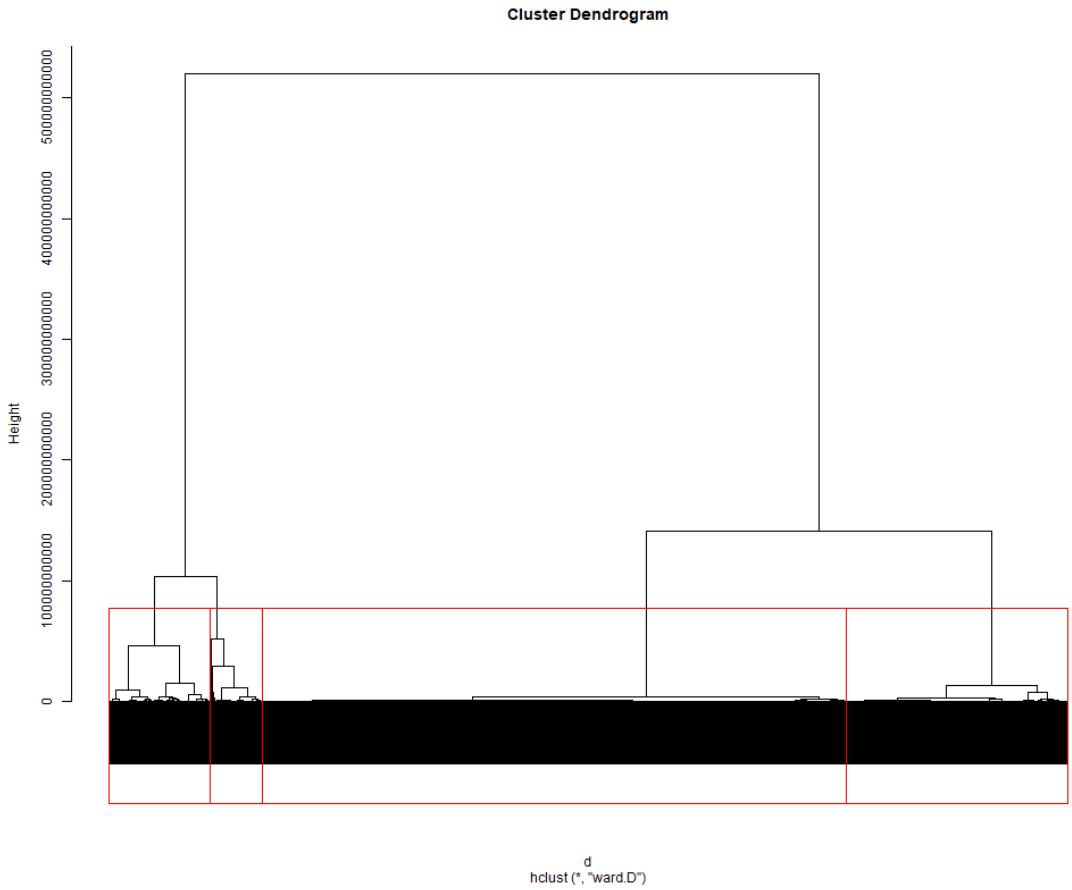


Figura 2: Dendrogram and clusters with k=4

Per visualizzare i cluster prodotti con kmeans è stato prodotto un grafico che mostra lo scatterplot di ogni coppia di feature e come variano i cluster e i centroidi in quella rappresentazione.

```
execute_kmeans <- function(data, k=4) {
    fit <- kmeans(data, k)
    plot(data, col=fit$cluster+1, pch=16)
    points(fit$centers, pch=7, col="black")
}

execute_kmeans(data, 4)
```

Figura 3: Plot of all features combinations and clusters with k=4

Avendo 8 dimensioni nei dati, lo scatterplot con i cluster è generato per coppie di feature. E' interessante vedere come in alcuni degli scatterplot sono visibili dei trend, probabile indice di correlazione tra le dimensioni.

2 Task 2

2.1 Descrizione del task

Per il secondo task è richiesto di effettuare una PCA sui dati, sia con iil parametro scale=TRUE che scale=FALSE, e produrre screeplot e biplot. Inoltre è richiesti di rispondere alle seguenti domande:

1. Quanta varianza è spiegata dalle due top PC?
2. Quante PC sono necessarie per catturare almeno il 90% della varianza totale?

2.2 Esecuzione

Per l'esecuzione di questo task e dei successivi è stato fatto uso del package `factoextra`. Questo package è stato usato per produrre le visualizzazioni rilevanti, nel caso dello script che segue

per biplot. Inoltre per rispondere alle domande relative alla varianza è stata definita una funzione apposita, che produce un plot dell'aumento della varianza all'aggiunta delle varie principal components.

```
plot_variance_explained <- function(pc, percentage_line = 90) {
  #questa funzione mostra quante PC servono per raggiungere
  #il 90% della varianza spiegata

  # Calcolo della varianza percentuale spiegata da ogni PC
  var_exp <- (pc$sdev^2 / sum(pc$sdev^2)) * 100
  print("Percentuale di varianza spiegata da ogni PC:")
  print(var_exp)

  # Calcolo della varianza cumulativa
  cum_var_exp <- cumsum(var_exp)

  # Creazione del grafico
  plot(cum_var_exp, type = "b", xlab = "number of PCs",
    ylab = "% of variance explained",
    main = "Percentage of variance explained by PCs",
    ylim = c(0, 100), cex.main = 2)

  # Tracciamento della linea al percentuale specificata
  abline(h = percentage_line, col = "red", lwd = 2, lty = 2)

  # Aggiunta dell'annotazione
  text(x = length(cum_var_exp), y = percentage_line-4,
       labels = paste(percentage_line, "% of the variance"),
       pos = 2, col = "red")
}
```

Il resto dello script per questo task è molto semplice, si occupa solo di produrre i plot necessari dopo aver calcolato la PCA e salvarli tramite la funzione png. Il processo viene ripetuto per scale=TRUE e scale=FALSE.

```
PC <- prcomp(data, scale=FALSE)

#Screeplot
png(filename = paste0(plots, "Task2 - unscaled screeplot.png"),
    width = 1000, height = 800)
screeplot(PC, main = "PC with scale=FALSE")
dev.off()

# Biplot
png(filename = paste0(plots, "Task2 - unscaled biplot.png"),
    width = 1000, height = 800)
fviz_pca_var(PC)
dev.off()

# 90% variance explained
png(filename = paste0(plots, "Task2 - unscaled PCs variance.png"),
    width = 1000, height = 800)
plot_variance_explained(PC,90)
dev.off()
```

Al termine di questa fase abbiamo quindi in output la varianza spiegata da ogni pc, lo screeplot, il biplot e un ulteriore visualizzazione della varianza necessaria per raggiungere il target del 90%, per entrambi i valori di scale.

2.3 Analisi dei risultati

Come è evidente da tutti gli output prodotti, lo scaling dei dati crea una variazione importante nella varianza spiegata da ogni PC. In particolare in assenza di scaling la PC1 diventa la variabile

predominante, con circa il 99% della varianza e meno dell'1% per le rimanenti 7 PC. Questo è dovuto principalmente alla natura dei dati, che sono rappresentati su una scala più ampia in una delle 8 dimensioni rispetto alle altre e questo influisce negativamente sulla fase di PCA, creando una sorta di bias verso una delle dimensioni. Lo scaling quindi non serve altro che a riportare tutti i dati in una scala di rappresentazione standard e unica per tutte le dimensioni e viene fatto sottraendo ad ogni valore la media del campione dividendo il risultato per la deviazione standard.

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

PC	**% Varianza spiegata (scale=FALSE)**	**% Varianza spiegata (scale=TRUE)**
PC1	99.32418	31.9454414
PC2	0.6506203	24.2796545
PC3	0.02201085	13.4219435
PC4	0.003187339	11.8890142
PC5	4.460643e-07	11.0679474
PC6	3.933665e-10	5.3059265
PC7	5.977741e-12	1.1063661
PC8	1.260364e-14	0.9837064

Tabella 1: Percentuale di varianza spiegata da ogni PC

L'applicazione dello scaling ai dati mostra una distribuzione molto più equilibrata della varianza spiegata da ogni PC. A questo punto siamo pronti a rispondere alla prima domanda del task, "Quanta varianza è spiegata dalle due top PC?". Nel caso scale=FALSE la varianza spiegata è il 99.97%, mentre nel caso scale=TRUE questa è pari al 56.22%.

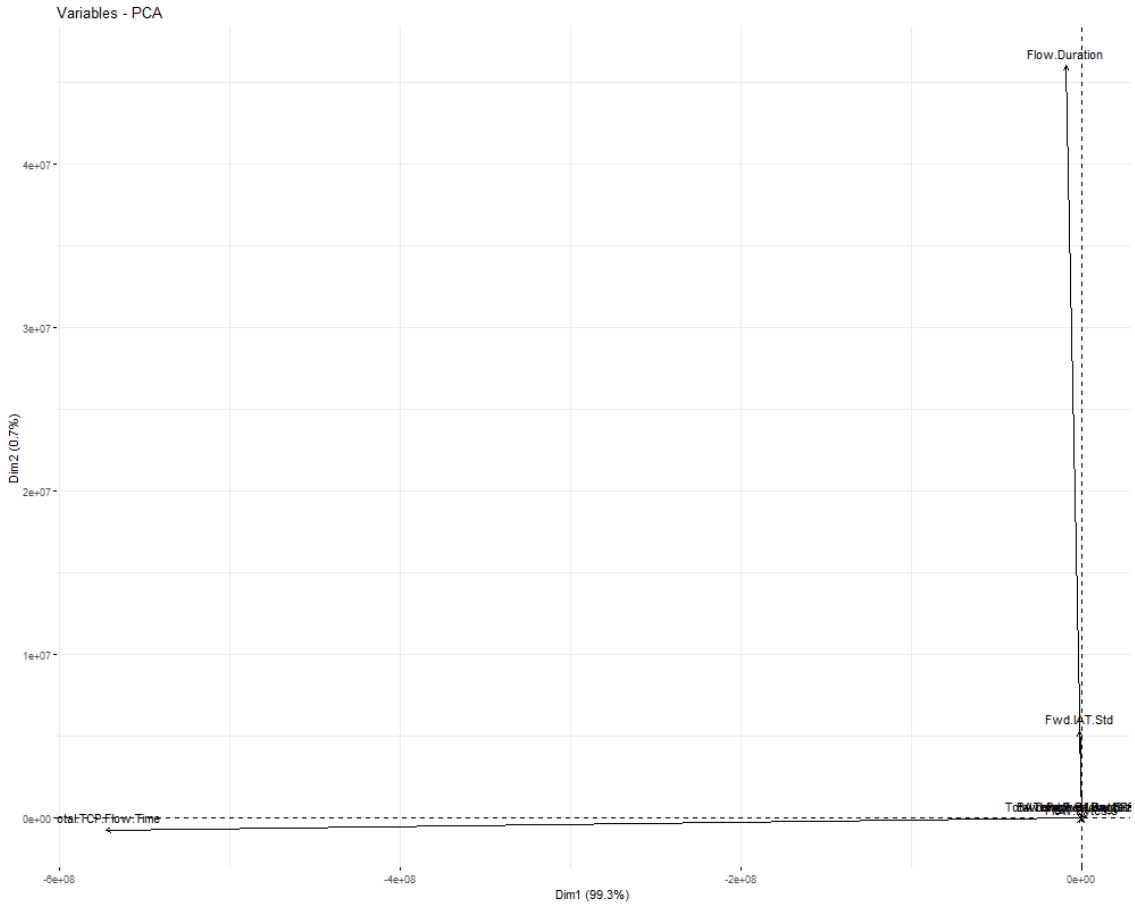


Figura 4: Unscaled biplot

L'importanza dell'effetto dello scaling dei dati è visibile anche a colpo d'occhio confrontando i biplot prodotti. Mentre nel biplot relativo al caso scaled è possibile individuare relazioni tra le

variabili in termini di correlazione, nel caso unscaled si possono dedurre informazioni significative solo dalle due variabili "Total TCP Flow time" e "Flow duration", che risultano essere non correlate. Bisogna ricordare che il biplot è una rappresentazione delle variabili su uno spazio fatto dalle due top PC, in cui ogni variabile è rappresentata da un vettore che parte da [0,0]. La proiezione della lunghezza del vettore sugli assi indica quanto quella variabile "contribuisce" alla dimensione, mentre l'angolo tra i vari vettori indica quanto e se questi siano correlati.

Alla luce di queste informazioni, osservando il biplot unscaled possiamo dedurre che le due variabili "Total TCP Flow time" e "Flow duration" siano quasi corrispondenti alle due dimensioni, poiché sono parallele ad esse e non ci sono altri contributi significativi da parte di altre variabili a quegli assi.

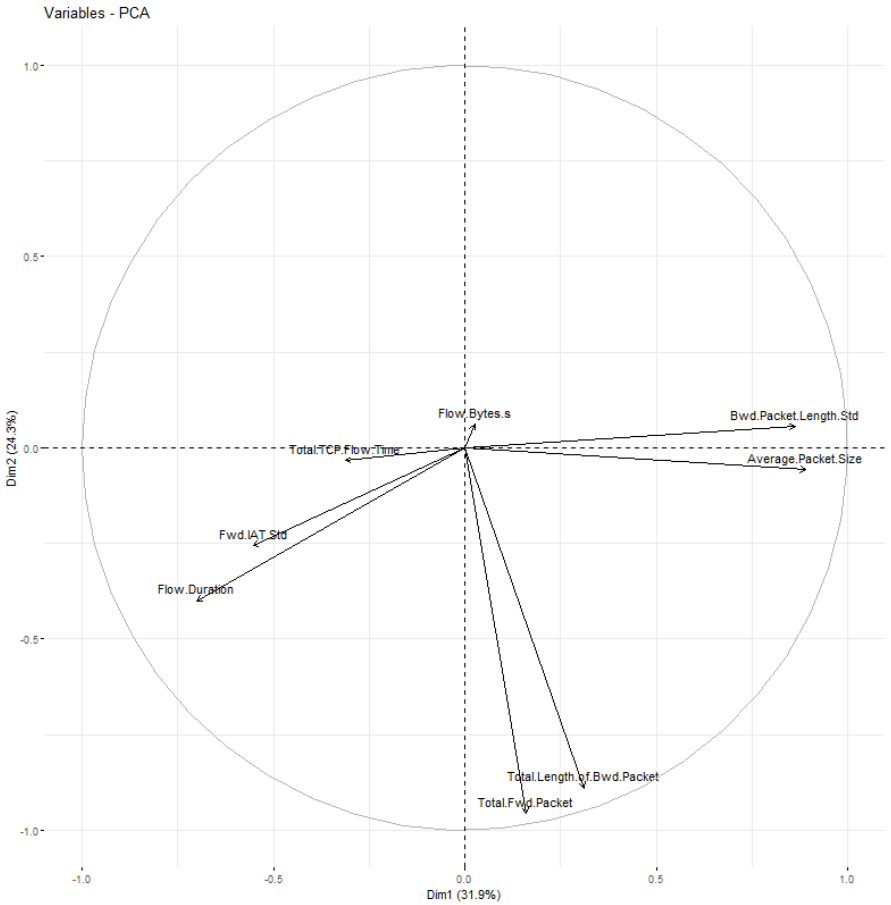
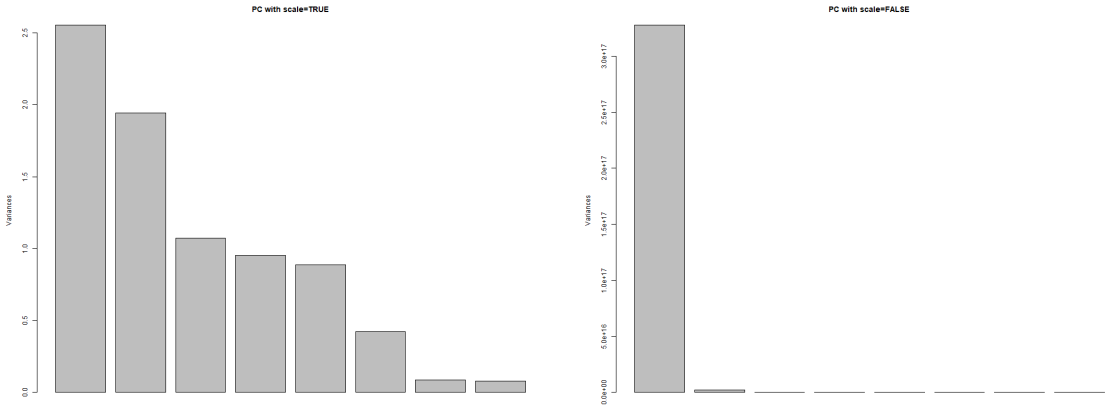


Figura 5: Scaled biplot

Nel biplot scaled è possibile individuare relazioni tra le variabili, che verranno però esplorate nel Task 3 di questo Homework.



(a) Scaled screeplot (b) Unscaled screeplot

Figura 6: screeplots

Gli screeplot riportati confermano quanto analizzato finora e forniscono una rappresentazione visiva dei valori della tabella precedente. Per rispondere alla seconda domanda del task "Quante PC sono necessarie per catturare almeno il 90% della varianza totale?" consideriamo i valori della varianza spiegata e li sommiamo semplicemente. Per il caso scaled servono almeno 5 PC per spiegare il 90% della varianza, per il caso unscaled serve soltanto una PC.

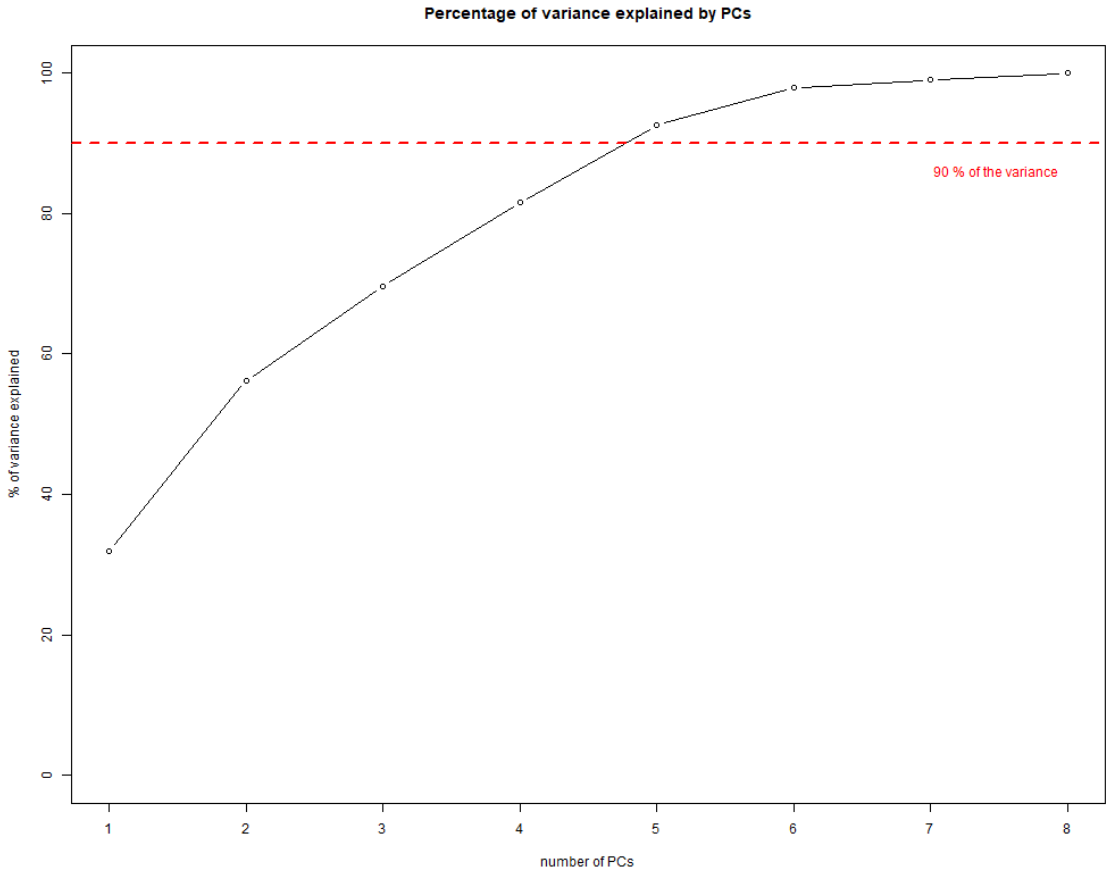


Figura 7: Scaled PCs variance

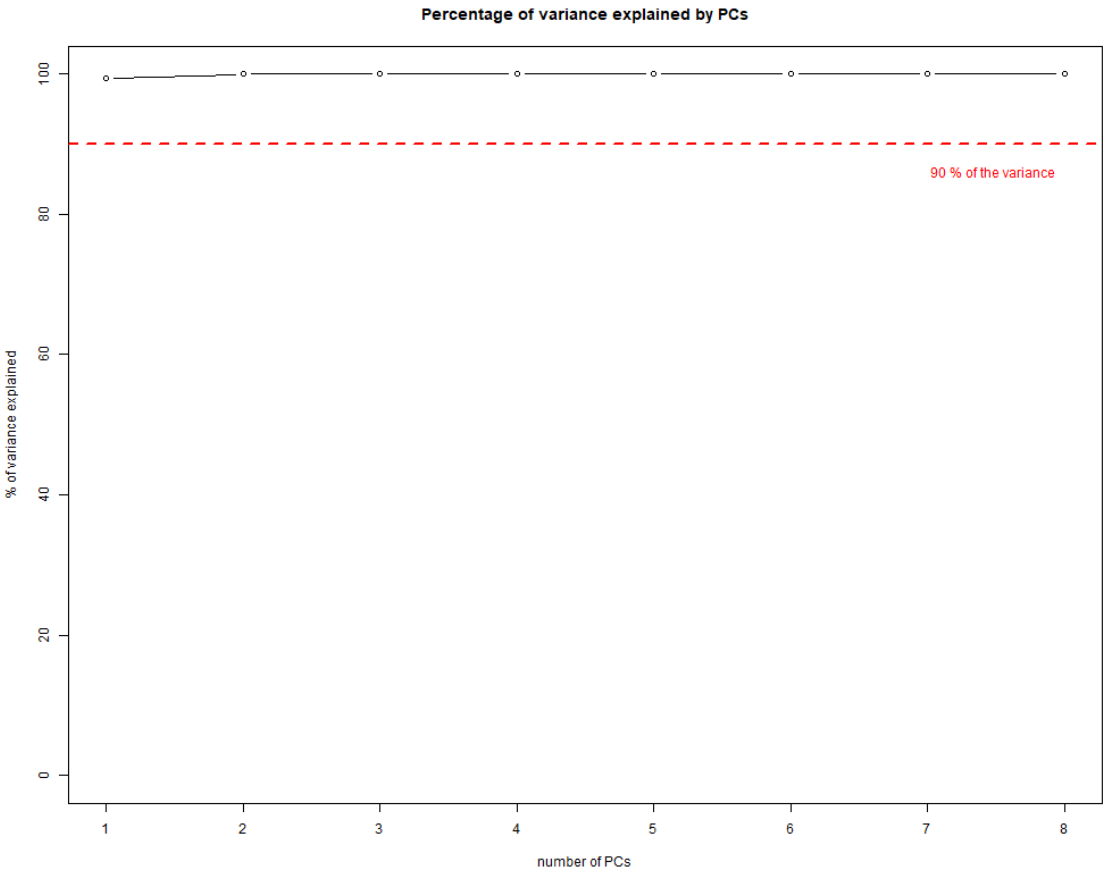


Figura 8: Unscaled PCs variance

3 Task 3

3.1 Descrizione del task

Nel terzo task si richiede di identificare tutte le coppie di variabili correlate, anticorrelate e non correlate, basandosi sul biplot prodotto con il parametro `scale=TRUE`. Per ognuna di queste coppie bisogna:

1. Produrre lo scatterplot $x - y$;
2. Calcolare la regressione lineare (lm);
3. Calcolare R^2;
4. Generare il grafico con il 95% di confidenza sulla regressione e gli intervalli di predizione.

3.2 Esecuzione

Per l'esecuzione di questo task è stata definita una funzione che effettua tutte le analisi menzionate nella descrizione del task sulle due variabili passate alla stessa come input.

```
analyze_var<-function(data,xname,yname,corr,vis_path){
  #Questa funzione produce uno scatterplot, un modello lineare
  #e un grafico con intervalli di confidenza e predizione

  x=data[[xname]]
  y=data[[yname]]

  #scatterplot
  png(filename = paste0(vis_path, "Task3 - Scatterplot of ",xname," - ",yname
    ," (",corr,").png"), width = 1000, height = 800)
```

```
  plot(x,y,main=paste0("Scatterplot of ",xname," - ",yname," (",corr,")"),
    xlab=xname,ylab=yname, cex.main = 2)
  dev.off()

  #model fitting
  if (corr != "uncorrelated") {
    png(filename = paste0(vis_path, "Task3 - Linear regression model for ",
        xname," - ",yname," (",corr,").png"), width = 1000, height = 800)
    plot(x,y,main=paste0("Linear regression model for ",xname," - ",yname,
        " (",corr,")"),xlab=xname,ylab=yname, cex.main = 2)

    model<-lm(y ~ x)
    abline(model,col="blue")

    print(paste0("R squared for ",xname," - ",yname
        ," regression model: ",summary(model)$r.squared))
    cint<-predict(model, level = 0.95, interval = "confidence")
    pint<-predict(model, level = 0.95, interval = "prediction")

    lines(x, cint[,2], type="o", lty=2, col="magenta")
    lines(x, cint[,3], type="o", lty=2, col="magenta")
    lines(x, pint[,2], type="o", lty=2)
    lines(x, pint[,3], type="o", lty=2)
    dev.off()
  }
}
```

Per ogni coppia di variabili individuate nel biplot è stata quindi eseguita la funzione. La funzione effettua il fitting di un modello lineare solo se si tratta di variabili indicate come anticorrelate o correlate.

```
# Variabili anticorrelate
analyze_var(flows, "Total.TCP.Flow.Time", "Average.Packet.Size", "anticorrelated",plots)
analyze_var(flows, "Bwd.Packet.Length.Std", "Total.TCP.Flow.Time", "anticorrelated",plots)

# Variabili non correlate
analyze_var(flows, "Fwd.IAT.Std", "Total.Length.of.Bwd.Packet", "uncorrelated",plots)
analyze_var(flows, "Flow.Duration", "Total.Length.of.Bwd.Packet", "uncorrelated",plots)
analyze_var(flows, "Fwd.IAT.Std", "Total.Fwd.Packet", "uncorrelated",plots)
analyze_var(flows, "Flow.Duration", "Total.Fwd.Packet", "uncorrelated",plots)
analyze_var(flows, "Total.Fwd.Packet", "Total.TCP.Flow.Time", "uncorrelated",plots)
analyze_var(flows, "Flow.Bytes.s", "Average.Packet.Size", "uncorrelated",plots)

# Variabili correlate
analyze_var(flows, "Bwd.Packet.Length.Std", "Average.Packet.Size", "correlated",plots)
analyze_var(flows, "Fwd.IAT.Std", "Flow.Duration", "correlated",plots)
analyze_var(flows, "Total.Length.of.Bwd.Packet", "Total.Fwd.Packet", "correlated",plots)
```

3.3 Analisi dei risultati

Tramite l'analisi sul biplot sono state individuate le seguenti coppie di variabili correlate:

- Bwd Packet Length Std - Average Packet Size;
- Fwd IAT Std - Flow Duration;
- Total Length of Bwd Packet - Total Fwd Packet;

Queste variabili sono state individuate in base all'angolo che le rispettive rappresentazioni vettoriali formano sul biplot.

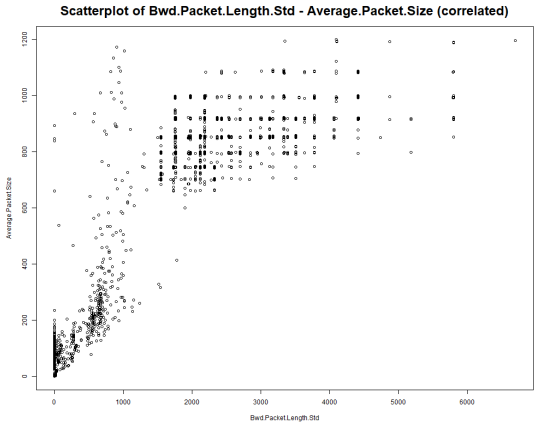


Figura 9: Scatterplot of Bwd.Packet.Length.Std - Average.Packet.Size

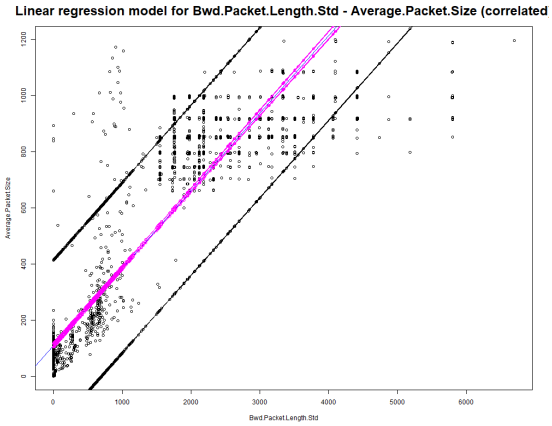


Figura 10: Linear regression model for Bwd.Packet.Length.Std - Average.Packet.Size

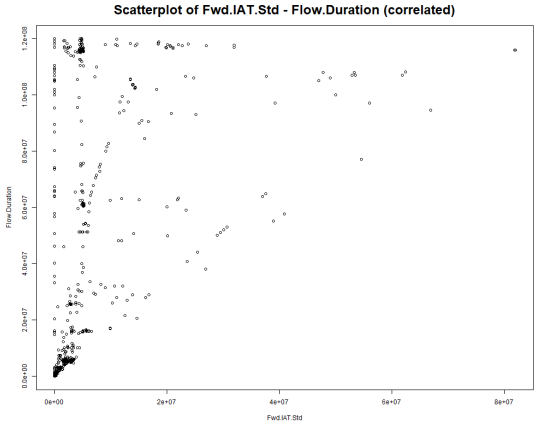


Figura 11: Scatterplot of Fwd.IAT.Std - Flow.Duration

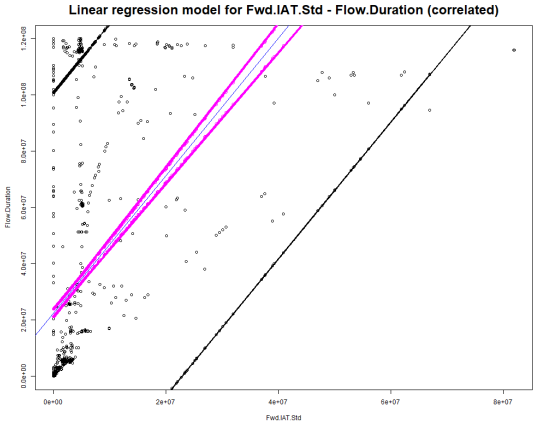


Figura 12: Linear regression model for Fwd.IAT.Std - Flow.Duration

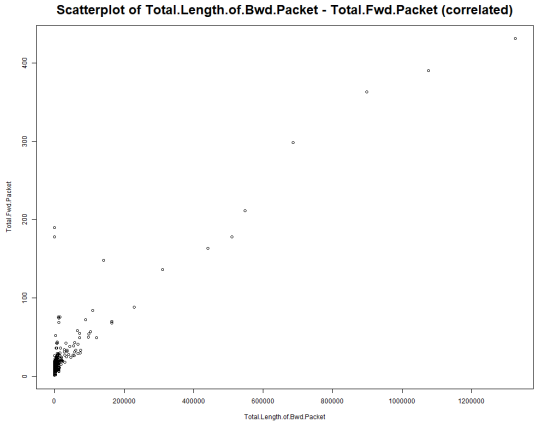


Figura 13: Scatterplot of Total.Length.of.Bwd.Packet - Total.Fwd.Packet

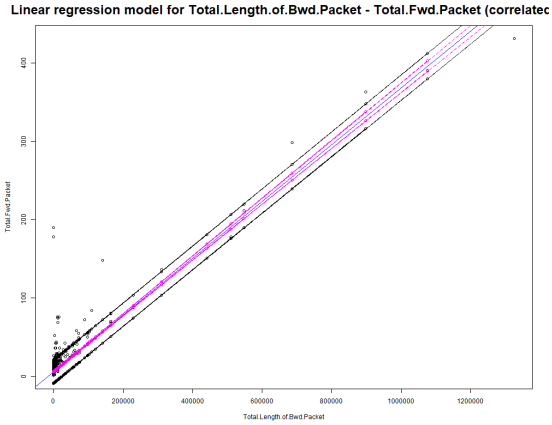


Figura 14: Linear regression model for Total.Length.of.Bwd.Packet - Total.Fwd.Packet

Da una veloce analisi visiva si può già individuare il peggior performer tra i modelli generati, ovvero quello sulle variabili Fwd IAT Std e Flow Duration, per via delle ampissime fasce del prediction interval. In generale gli scatterplot mostrano delle relazioni tra le variabili che non sono facilmente assimilabili a modelli lineari, se non per la coppia Total Length of Bwd Packet - Total Fwd Packet. Si può inferire però una correlazione tra le variabili, poiché all'aumentare di una variabile sembra aumentare anche l'altra.

Le variabili anticorrelate invece sono:

- Bwd Packet Length Std - Total TCP Flow Time;
- Total TCP Flow Time - Average Packet Size;

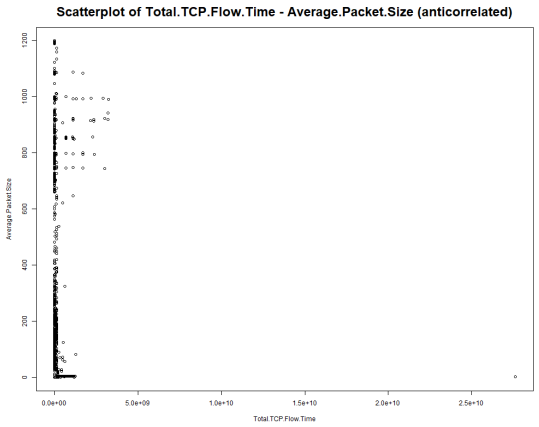


Figura 15: Scatterplot of Total.TCP.Flow.Time - Average.Packet.Size

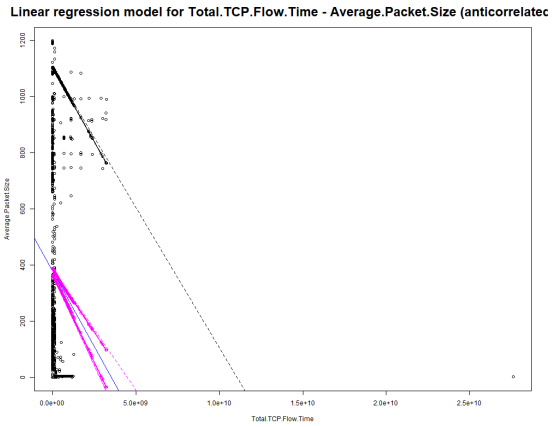


Figura 16: Linear regression model for Total.TCP.Flow.Time - Average.Packet.Size

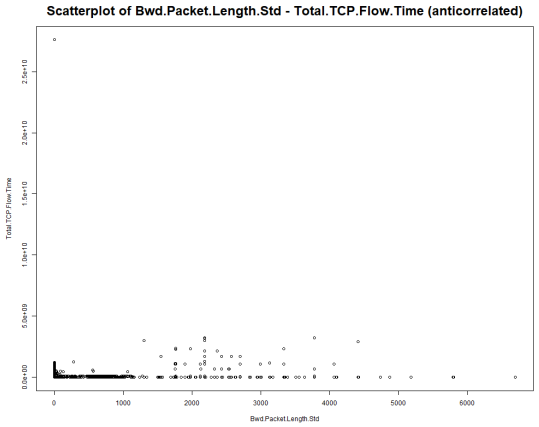


Figura 17: Scatterplot of Bwd.Packet.Length.Std - Total.TCP.Flow.Time

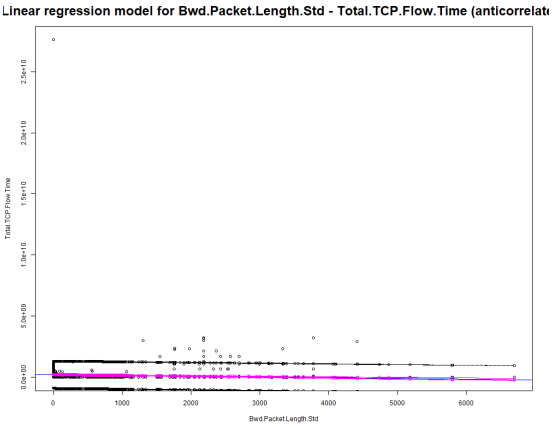


Figura 18: Linear regression model for Bwd.Packet.Length.Std - Total.TCP.Flow.Time

Come per le variabili correlate è difficile definire un modello lineare adeguato a come si dispongono i datapoints sullo scatterplot, ma è possibile individuare per TCP flow time e Average packet size una anticorrelazione, ovvero all'aumentare del valore di una variabile l'altra diminuisce. Per l'altra coppia individuata invece risulta molto difficile trarre la stessa conclusione.

La valutazione dei modelli attraverso una misura statistica viene spesso eseguita utilizzando il parametro R^2, che varia da 0 a 1. Un R^2 vicino a 1 indica che una grande percentuale della variabilità nella variabile dipendente è spiegata dal modello di regressione, ovvero le variabili indipendenti nel modello spiegano bene la variazione della variabile dipendente. Al contrario, un R^2 vicino a 0 indica che il modello di regressione non spiega efficacemente la variabilità della variabile dipendente.

x	y	Correlation	R^2
Total.TCP.Flow.Time	Average.Packet.Size	Anticorrelated	0.027
Bwd.Packet.Length.Std	Total.TCP.Flow.Time	Anticorrelated	0.016
Bwd.Packet.Length.Std	Average.Packet.Size	Correlated	0.832
Fwd.IAT.Std	Flow.Duration	Correlated	0.276
Total.Length.of.Bwd.Packet	Total.Fwd.Packet	Correlated	0.774

Tabella 2: Correlation and R^2 values for selected variable pairs

4 Task 4

4.1 Descrizione del task

In questo ultimo task viene richiesto di utilizzare le due top PC (sia per `scale=TRUE` che `scale=FALSE`) per ottenere un dataset a due dimensioni. Su questo dataset verrà ripetuta l'analisi di sensitività e `kmeans` per confrontare i risultati con il dataset iniziale. Sarà poi prodotto uno scatterplot del dataset, dei cluster e dei centroidi e saranno discussi i risultati ottenuti.

4.2 Esecuzione

Per questo task vengono selezionate solo due PC dalla PCA con scale=TRUE e FALSE.

```
scale=FALSE
PC <- prcomp(data, scale=scale)

# Get the first two principal components
PC1 <- PC$x[,1]
PC2 <- PC$x[,2]
pc_df <- data.frame(PC1 = PC1, PC2 = PC2)
```

Per produrre l'analisi di sensibilità è stato effettuato il kmeans clustering per 10 volte ed è stato valutato l'andamento della whitinss al variare del numero di cluster.

```
tss<-seq(1,10,1)
for ( i in 1:10) tss[i] <- kmeans(pc_df,i)$tot.withinss

png(filename = paste0(plots, "Task4 scale=",scale," - Sensibility analysis kmeans.png"),

    width = 1000, height = 800)
plot(tss, main = "Sensibility analysis - kmeans tot.withinss", type = "o",
    lwd = 1.5, xlab = "k",cex=2, cex.main = 2)

dev.off()
```

Lo scatterplot con cluster e centroidi è stato prodotto sia completo che zoomato sulla maggior parte dei punti, limitando l'asse x e y nel plot.

```
png(filename = paste0(plots,"Task4 scale=",scale,"- Kmeans with k=",k,".png"),
    width = 1000, height = 800)

fit <- kmeans(pc_df, k)
plot(main=paste0("Kmeans with k=",k," and scale=",scale),pc_df, col=fit$cluster+1,
    pch=16, cex.main = 2)
points(fit$centers, pch=7, col="black")

dev.off()

png(filename = paste0(plots,"Task4 scale=",scale,"- Kmeans with k=",k," (zoomed).png"),
    width = 1000, height = 800)

fit <- kmeans(pc_df, k)
plot(main=paste0("Kmeans with k=",k," (zoomed, scale=",scale,")"),pc_df, col=fit$cluster+1,
```

```
    pch=16, xlim = c(-4,4), ylim = c(-5,2), cex.main = 2)
points(fit$centers, pch=7, col="black")

dev.off()
```

E' stato prodotto anche il grafico del clustering gerarchico per k=6 e k=4.

```
d<-dist(pc_df,method = "euclidean")
fit<-hclust(d,method="ward.D")

png(filename = paste0(plots,"Task4 scale=",scale,"- hclust plot with k=4.png"),
    width = 1000, height = 800)
plot(main=paste0("Kmeans with k=4 and scale=",scale),fit, labels = FALSE,
    cex.main = 2)
groups <- cutree(fit, k=4)
rect.hclust(fit, k=4, border="red")
dev.off()

png(filename = paste0(plots,"Task4 scale=",scale,"- hclust plot with k=6.png"),
    width = 1000, height = 800)
plot(main=paste0("Kmeans with k=6 and scale=",scale),fit, labels = FALSE,
    cex.main = 2)
groups <- cutree(fit, k=6)
rect.hclust(fit, k=6, border="red")
dev.off()
```

4.3 Analisi dei risultati

Il primo passaggio è quello di ripetere l'analisi di sensitività con entrambi i valori di scale, sulle due top PC. E' facile notare come per i dati unscaled l'analisi di sensitività non cambia rispetto al dataset iniziale. E' facile immaginare sia per via del fatto che i dati della colonna "Total TCP Flow time" che influenzano pesantemente sia i dati originali che unscaled.

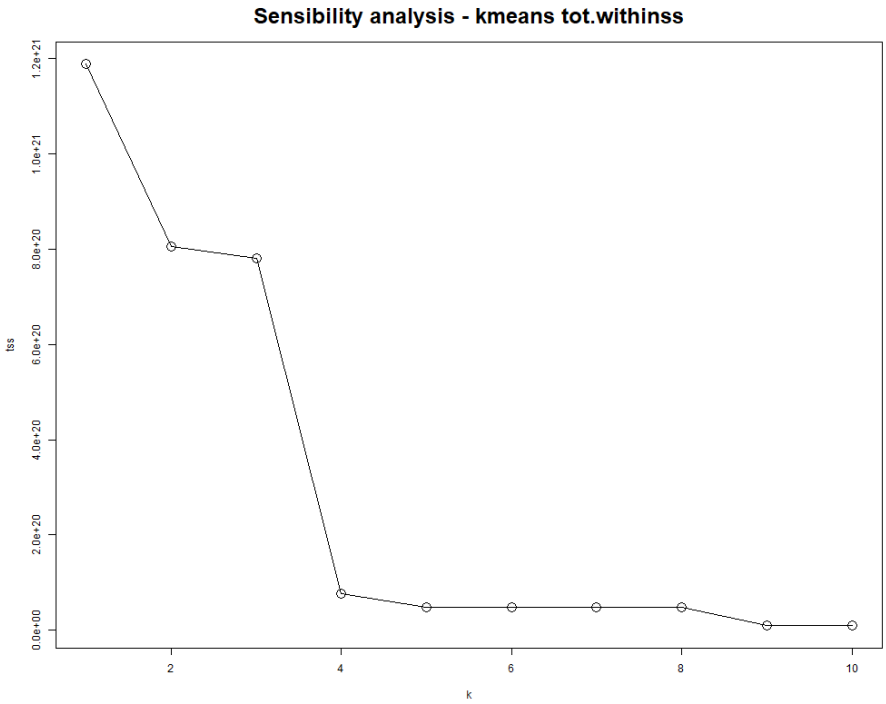


Figura 19: Sensibility analysis with whitinss,scale=FALSE

Al contrario il caso scaled presenta una notevole differenza con il dataset originale, portando un nuovo punto di k ottimale tra 6 e 7, in base alle iterazioni.

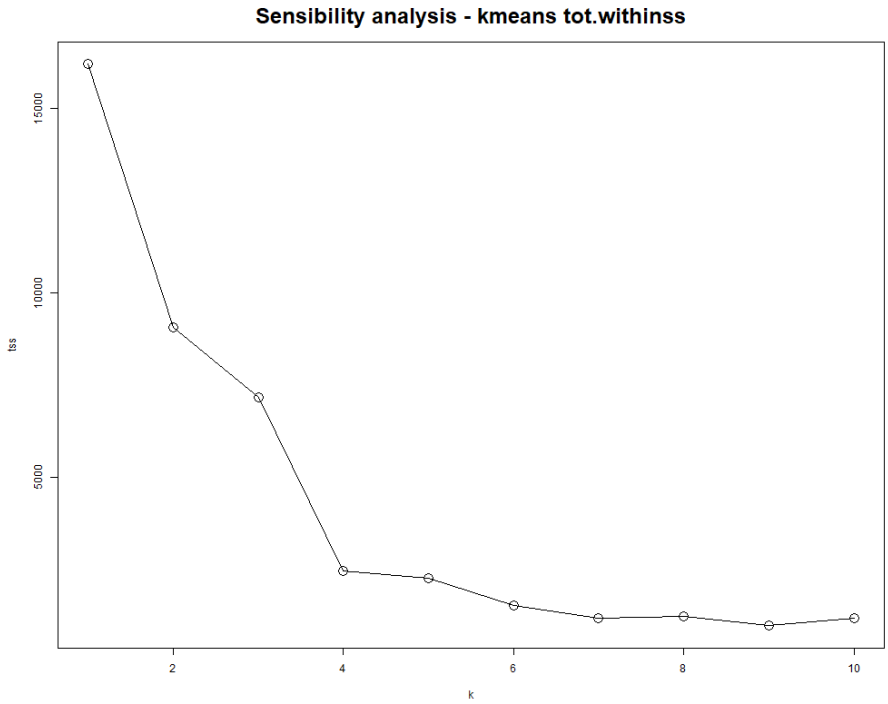


Figura 20: Sensibility analysis with whitinss,scale=TRUE

4.3.1 Kmeans con scale=FALSE

La rappresentazione nello spazio dei punti con scale=FALSE produce una divisione poco netta in cluster e in generale una disposizione non soddisfacente dei datapoints nello spazio ai fini dell'analisi da svolgere.

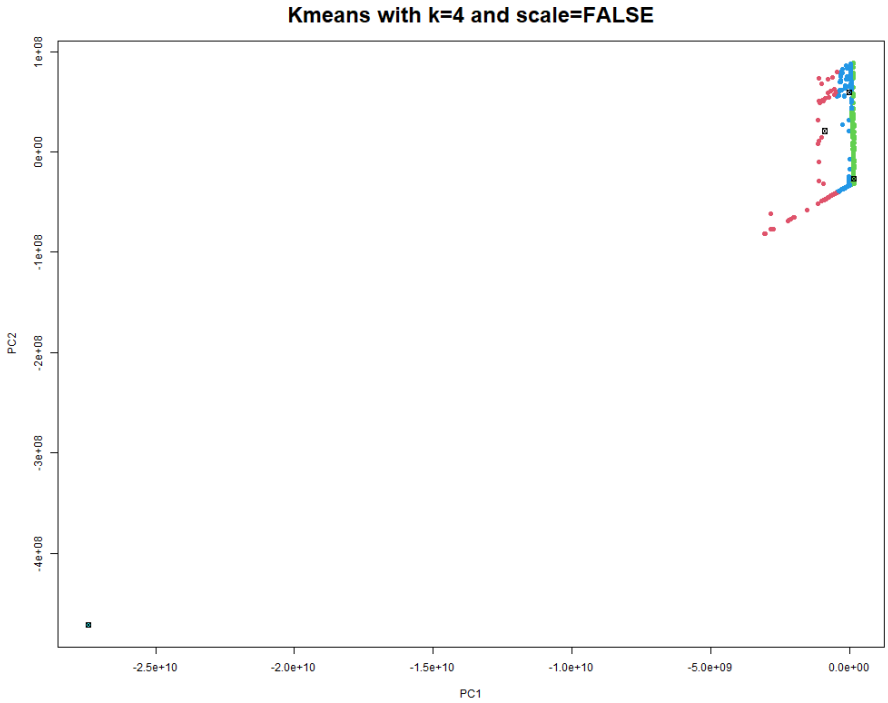


Figura 21: Kmeans,scale=FALSE,k=4

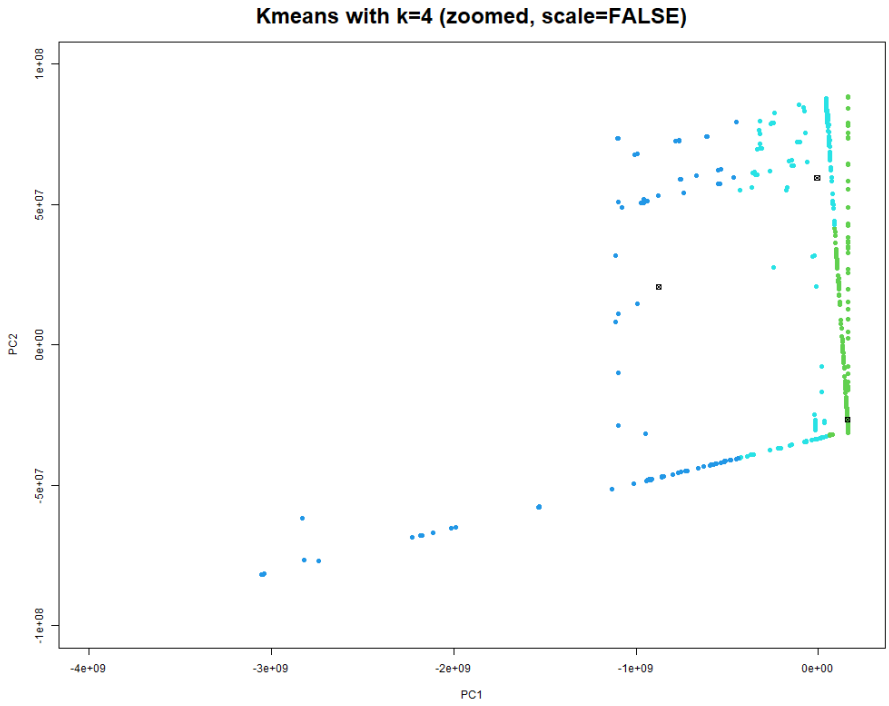


Figura 22: Kmeans,scale=FALSE,k=4,zoomed

4.3.2 Kmeans con scale=TRUE

Il parametro scale=TRUE ha un impatto molto positivo sulla disposizione dei datapoint nello spazio di rappresentazione scelto, in base alle due top PC. Nello scatterplot i punti nono visibilmente divisi e a differenza del caso precedente anche gli outliers si dispongono in maniera distribuita nel piano.

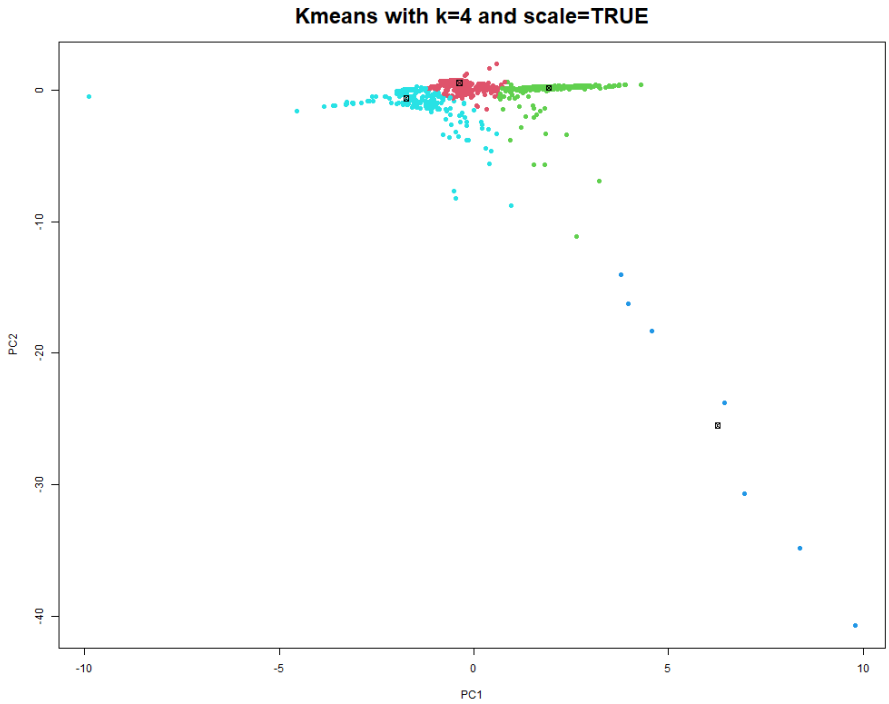


Figura 23: Kmeans,scale=TRUE,k=4

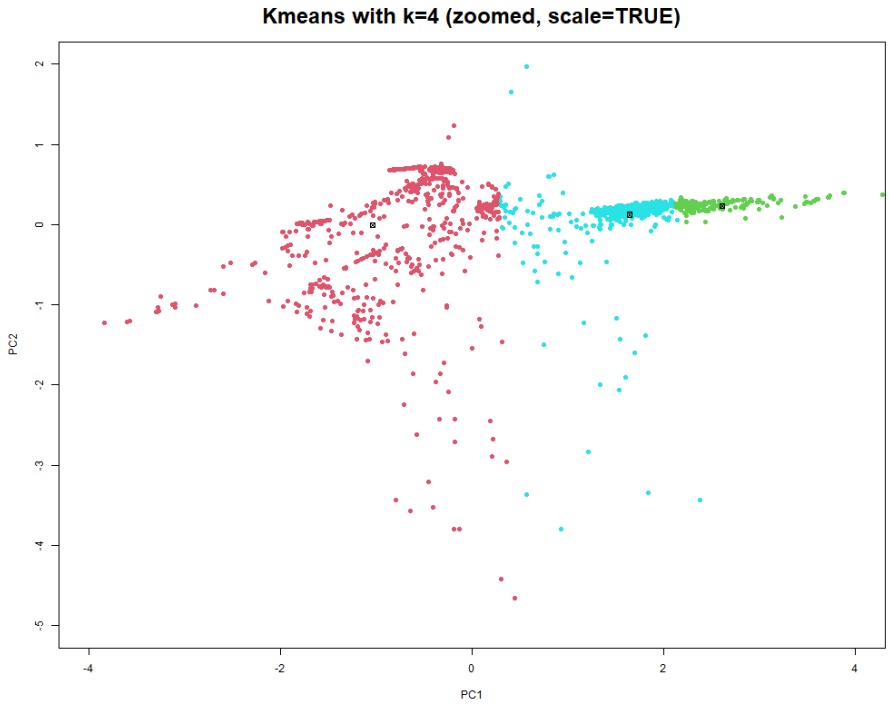


Figura 24: Kmeans,scale=TRUE,k=4,zoomed

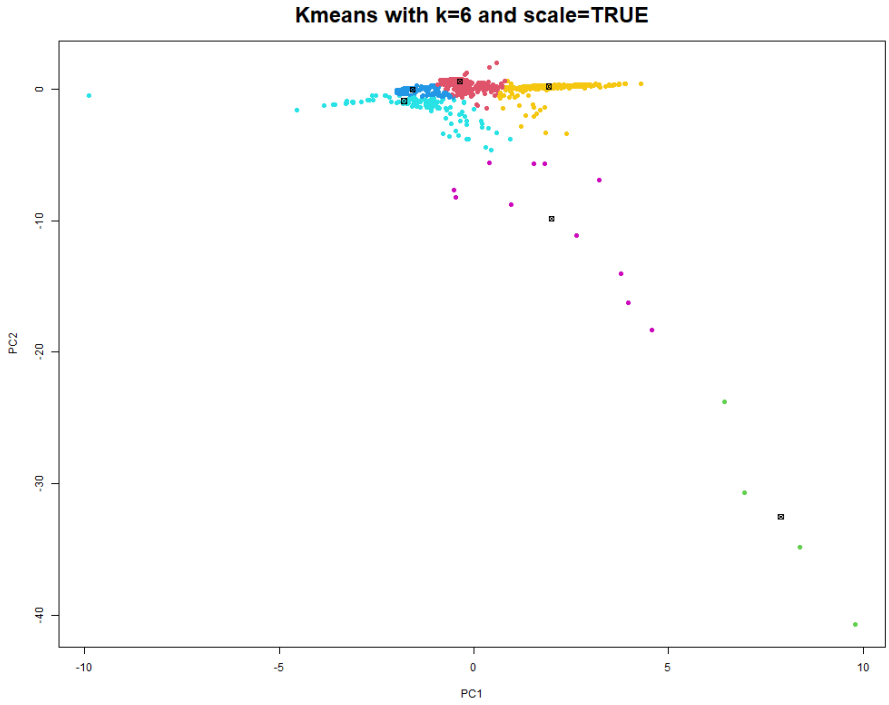


Figura 25: Kmeans,scale=TRUE,k=6

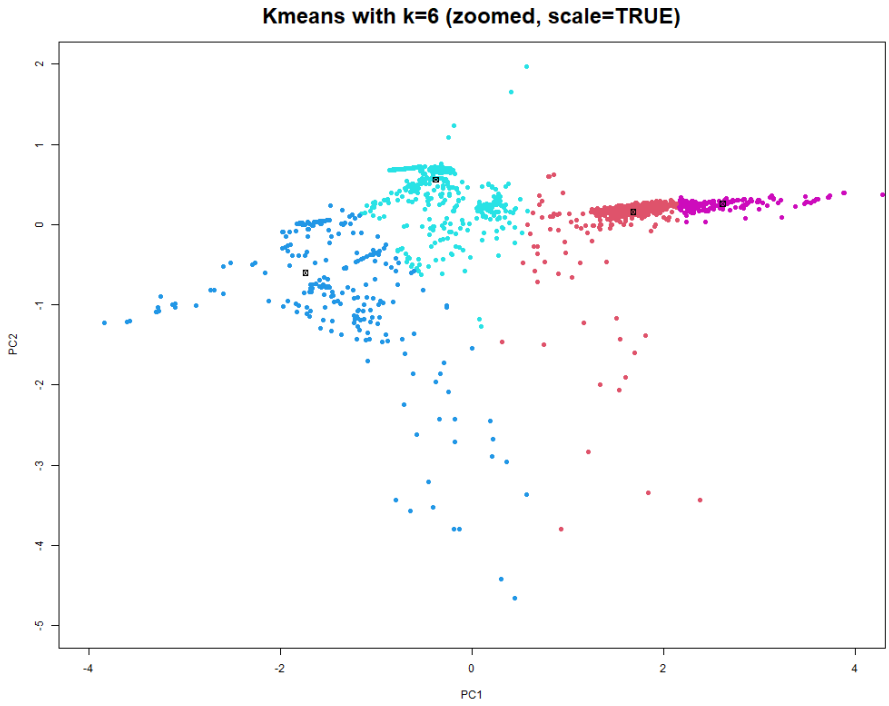


Figura 26: Kmeans,scale=TRUE,k=6,zoomed

4.4 Hclust

Al termine di questo task è stato ripetuto anche il clustering di tipo gerarchico e la generazione del dendrogramma.

4.4.1 hclust con scale=FALSE

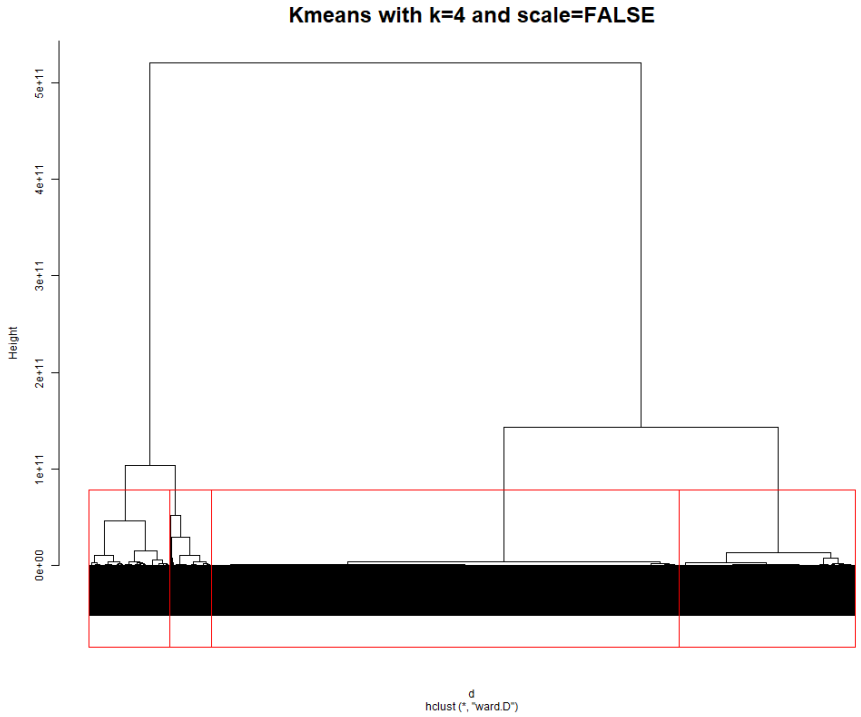


Figura 27: hclust, scale=FALSE, k=4

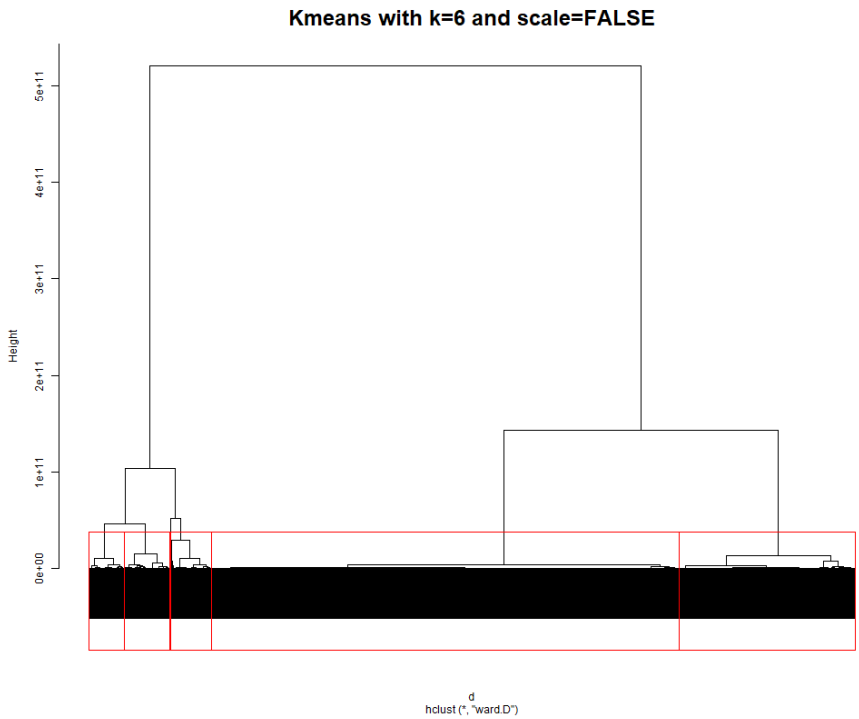


Figura 28: hclust, scale=FALSE, k=6

4.4.2 hclust con scale=TRUE

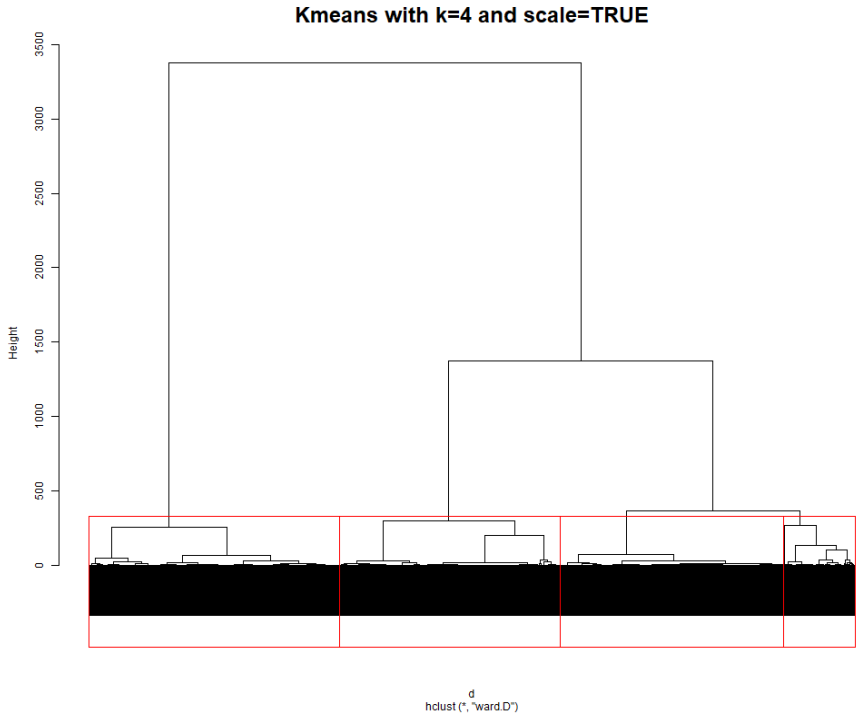


Figura 29: hclust,scale=TRUE,k=4

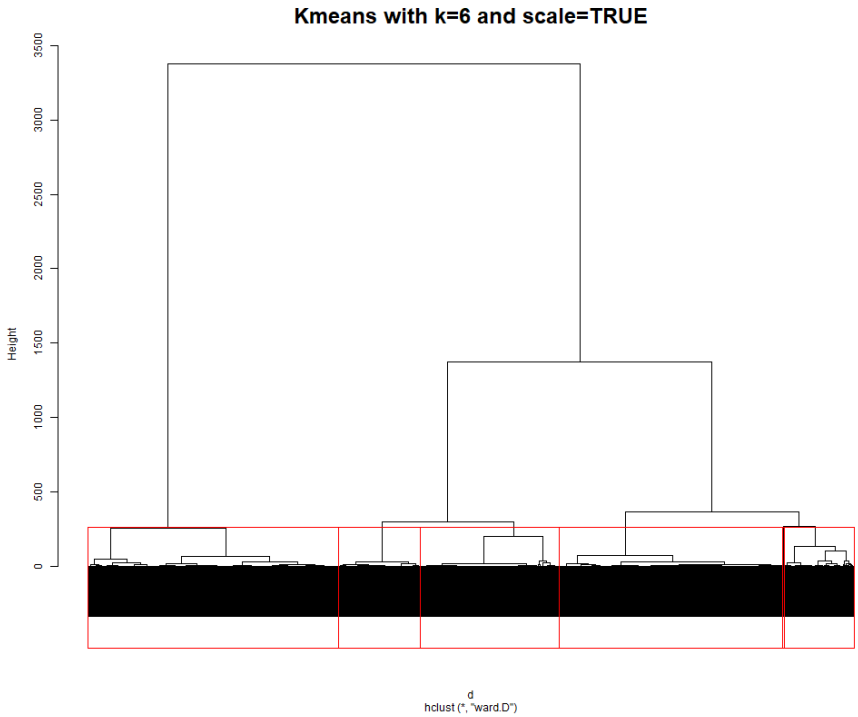


Figura 30: hclust,scale=TRUE,k=6