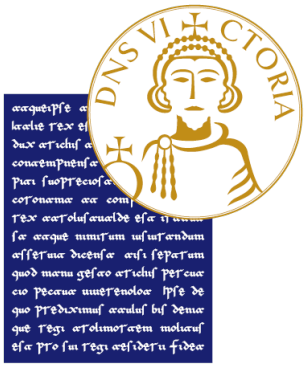


Università degli Studi del Sannio

Dipartimento di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

Data science

Homework 2

Prof:
Pecchia Antonio

Studenti:
Cinelli Jessica, 399000529
Mazzitelli Francesco C., 399000532

Anno Accademico 2022-2023

Indice

Elenco delle figure

1 Introduzione

Il seguente studio[1] ha avuto come obiettivo l'analisi del traffico di rete tra due componenti, ponendo l'enfasi sul riconoscimento del traffico malevolo. Lo studio ha previsto quindi un'analisi delle principali metriche quali lunghezza del pacchetto o durata del flusso, sottoponendole ad algoritmi di clustering per evidenziarne valori anomali. Il clustering è un insieme di tecniche di analisi multivariata il cui scopo è quello di riuscire a raggruppare in insiemi disgiunti tra loro, elementi omogenei il cui elemento rappresentante è detto centroide. Il centroide non per forza deve coincidere con un elemento dell'insieme, l'importante è che sia in qualche modo rappresentativo delle caratteristiche principali degli elementi clusterizzati. Sono state utilizzate le seguenti tecniche di clustering:

- **Clustering gerarchico** (paragrafo 2.1)
- **Clustering point-assigment** (paragrafo 2.2)

Il dataset analizzato è strutturato nelle seguenti dimensioni, ognuna indicante una metrica estrapolata dall'analisi del traffico di rete

- **Total Length of Bwd Packet**
- **Flow Bytes/s**
- **Total TCP Flow Time**
- **Total Fwd Packet**
- **Fwd IAT Std**
- **Bwd Packet Length Std**
- **Flow Duration**
- **Average Packet Size**

[1] È possibile visionare l'intero progetto al link https://github.com/jessicacinelli/Homework2.git.

2 Clustering

Il primo task dell'analisi prevede l'applicazione al dataset di partenza delle seguenti strategie di clustering:

- Clustering gerarchico;
- Clustering di tipo point assignment (k-means).

L'Appendice A riporta gli script R utilizzati per le due strategie utilizzate.

2.1 Clustering gerarchico

Questo approccio ha come obiettivo quello di riuscire ad individuare una gerarchia di cluster a partire da un insieme di partenza. Può essere di due tipi:

1. **Bottom-up o agglomerativo**: si parte dai singoli elementi da aggregare (n gruppi di dimensione 1) per arrivare ad un unico gruppo di dimensioni n;
2. **Top-down o divisivo**: si parte da un unico gruppo per arrivare a n elementi unitari.

La strategia scelta per effettuare il clustering gerarchico è quella agglomerativa. La relativa funzione R è *hclust()*: tale funzione produce una classificazione gerarchica a partire da una matrice delle distanze. Per cui, prima di applicare l'algoritmo sono state calcolate le distanze euclidee tra i punti del dataset. Una volta ottenuta la matrice delle distanze è stato eseguito l'algoritmo di clustering gerachico agglomerativo.

E' stato analizzato il dendrogramma riportato in figura 1, con lo scopo di individuare visivamente il numero ottimale di cluster.

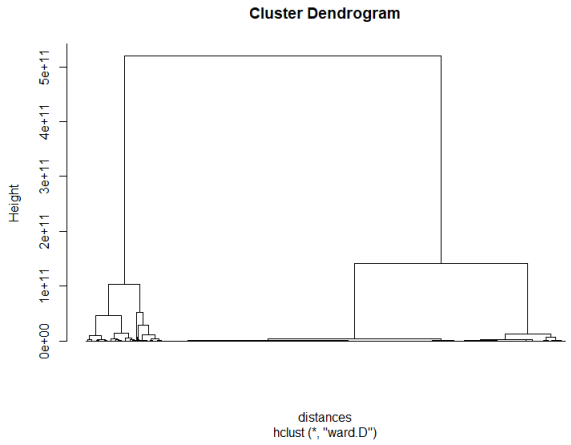


Figura 1: Dendrogramma del clustering gerarchico

Per poter individuare tale numero è stata studiata la distanza tra i gruppi in modo da evitare che questa fosse troppo elevata e scongiurare il rischio di fondere due sub-cluster molto distanti tra di loro all'interno di un unico cluster. Come riportato in figura 2 è stato scelto *k=3* come numero ottimale di cluster.

Successivamente, tramite un istogramma, è stata analizzata la distribuzione dei punti nei cluster individuati come mostrato in figura 3. Dall'analisi visuale del grafico è possibile notare come la maggior parte dei punti è presente nel primo cluster.

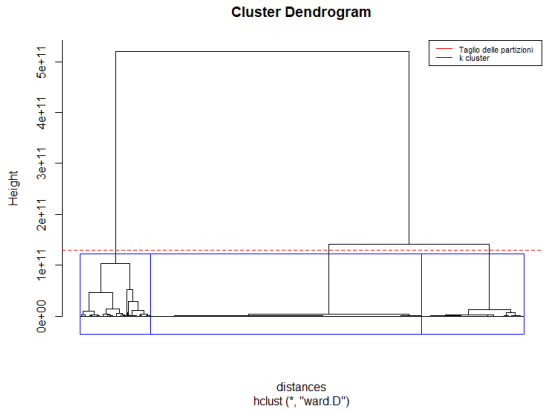


Figura 2: Partizioni individuate

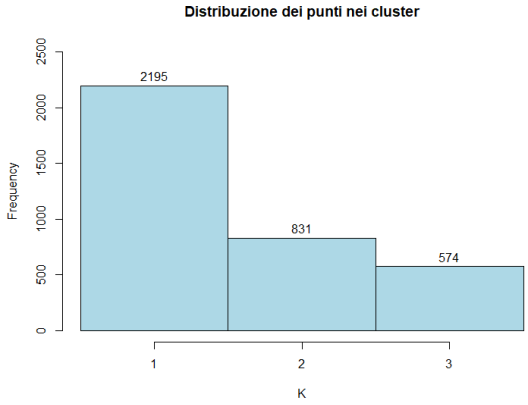


Figura 3: Distribuzione dei punti nei cluster

2.2 Clustering point-assignment

Il Clustering point-assignment definisce l'appartenenza ad un gruppo in relazione alla distanza da un punto rappresentativo del cluster chiamato centroide.

Per effettuare questo tipo di clustering è stato scelto l'algoritmo *k-means* la cui funzione R è *kmeans()*.
La strategia k-means prevede che il numero di cluster debba essere conosciuto in anticipo. Dunque, e stato necessario effettuare un'analisi di sensitività per individuare il numero ottimale di cluster. Per realizzare l'analisi di sensitività, è stata utilizzata la metrica della somma dei quadrati[2]: si tratta di una metrica indicativa per la qualità del clustering; decresce rapidamente in funzione del numero di cluster fino a raggiungere un valore stabile dopo un certo numero di k. Sono stati testati 10 cluster; per ognuno di essi è stata valutata la metrica Total Sum of Square attraverso il campo tot.withinss della funzione kmeans().

Osservando la curva in figura 4 è stato selezionato come numero ottimale di cluster il valore di k in corrispondenza del ginocchio della curva, ossia k=4.
In seguito è stata analizzata al distribuzione dei punti nei cluster individuati.

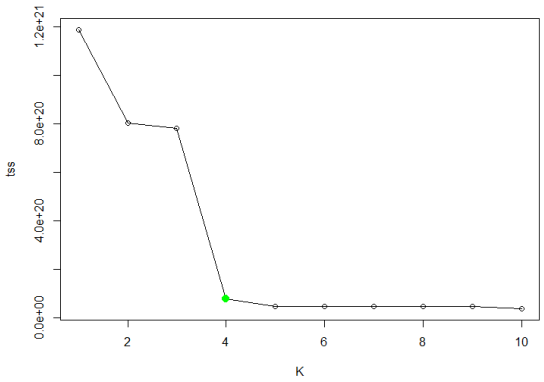


Figura 4: Scelta del valore ottimale di k

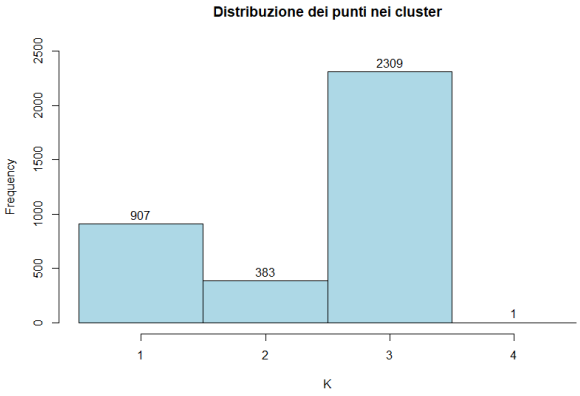


Figura 5: Distribuzione dei punti nei cluster

Come è possibile notare dalla figura 5 è presente un unico elemento all'interno del cluster 4.

[2]La metrica Total Sum of Square è data dalla somma delle distanze al quadrato tra i punti appartenenti al cluster k e il corrispondente centroide.

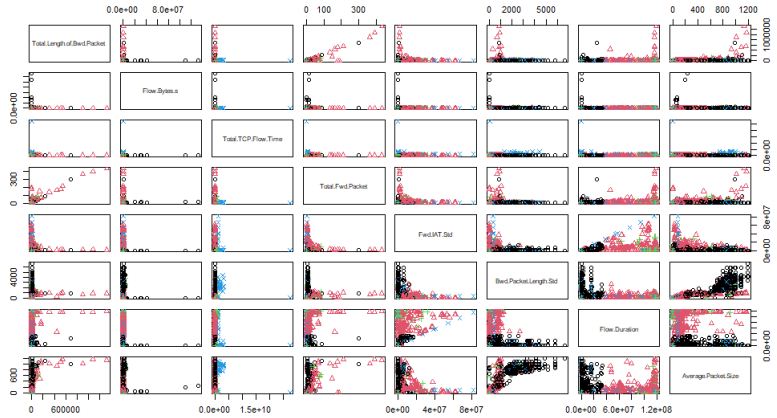
2.3 Analisi dei risultati e visualizzazione dei cluster

Dopo aver analizzato singolarmente i risultati ottenuti dall'applicazione delle due tecniche di clustering è stato ritenuto opportuno effettuare un confronto. In particolare, è possibile dedurre che:

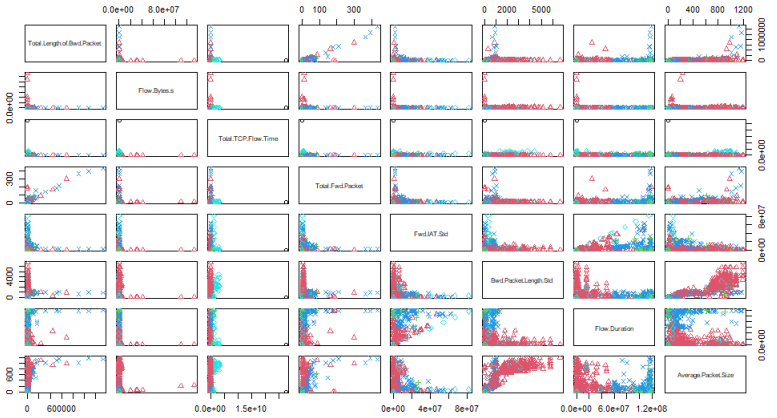
- Il numero di cluster ottimale per il clustering gerarichico è k=3;
- Il numero di cluster ottimale per il clustering point-assignment è k=4.

Per comprendere le distribuzioni delle osservazioni tra i cluster è stato utilizzato un pairwise scatterplot, cioè uno scatter plot a due dimensioni per tutte le possibili coppie di attributi. In particolare, la figura 6(a) riporta la clusterizzazione eseguita con la tecnica gerarchica, invece la figura 6(b) riporta la clusterizzazione eseguita con l'algoritmo k-means. La leggibilità risulta molto limitata quindi non è stato possibile individuare una relazione direttamente visibile tra tutti gli attributi.

Per il clustering point-assignment è stato individuato un insieme contenente un solo elemento. Tale elemento potrebbe essere considerato un outlier; poiché il dataset prevede 8 dimensioni e non è possibile visualizzare chiaramente il punto, non è stata effettuata alcuna operazione di post-processing ai risultati del k-means. Pertanto, si rimanda lo studio di eventuali outlier all'analisi delle componenti principali (sezione 5).



((a)) Clustering gerarchico



((b)) Clustering k-means

Figura 6: Selezione del punto di finestratura ottimale

3 Principal Component Analysis - PCA

Per ridurre la dimensionalità del dataset è stata utilizzata la Principal Component Analysis (PCA), una tecnica di tipo lineare che consente di analizzare le componenti principali, cioè gli attributi in grado di spiegare la maggior parte della variabilità dei dati.

La funzione R utilizzata per eseguire la PCA e *prcomp()*. Essa prevede un parametro di input *scale* settato a TRUE o FALSE che consente di specificare se le variabili devono o meno essere normalizzate. L'analisi è stata effettuata per entrambi i valori del parametro scale.

La funzione prcomp() produce in output il nuovo sistema di riferimento e le nuove coordinate delle osservazioni. Le nuove variabili sono rappresentate in ordine decrescente di significatività, cioè da quella che spiega la maggior parte della variabilità dei dati, a quella che invece ha un impatto meno significativo sulla variabilità.

Attraverso uno screeplot e stato possibile visualizzare l'incidenza di ogni dimensione sulla variabilità.
Per poter individuare quanta variabilità spiegano le n componenti selezionate è possibile effettuare il seguente calcolo:

$$\frac{\sum_{i=1}^{n}(sdev[i])^2}{\sum_{i=1}^{c}(sdev[i])^2} \tag{1}$$

dove n rappresenta il numero di componenti del dataset ridotto, c rappresenta il numero di componenti del dataset di partenza.

Una soluzione alternativa consiste nell'invocazione della funzione *get_eigenvalue()*.

Dal punto di vista grafico è possibile determinare il numero di componenti principali osservando uno *ScreePlot*, cioè il grafico degli autovalori ordinati dal più grande al più piccolo. Il numero di componenti è determinato nel punto oltre il quale gli autovalori rimanenti sono tutti relativamente piccoli e di dimensioni comparabili.

Per analizzare i risultati per le variabili, a partire da un output PCA, viene invocata la funzione *get_pca_var()*. Questa fornisce un elenco di matrici contenenti tutti i risultati per le variabili attive (coordinate, correlazione tra variabili e assi, coseno quadrato e contributi).
Nello specifico, per poter analizzare le relazioni tra tutte le variabili si utilizza il biplot (*fviz_pca_var()*).
All'interno di un biplot

- Le variabili correlate positivamente sono raggruppate;
- Le variabili correlate negativamente sono posizionate sui lati opposti dell'origine del grafico (quadranti opposti);

- La lunghezza del vettore misura quanta varianza di quella variabile è spiegata dalle prime due componenti principali. Più è lungo e più la varianza è colta da quella dimensione.

Per determinare la qualità di rappresentazione delle variabili si utlizza la metrica *cos2* che corrisponde al quadrato del coseno, ed è calcolata usando la funzione *fviz_cos2*. Un valore basso indica che la variabile non è perfettamente rappresentata da quella componente. Un valore alto, invece, indica una buona rappresentazione della variabile su quella componente.
L'Appendice B riporta gli script realizzati per condurre l'analisi.

3.1 Variabili non scalate ("scale = FALSE")

La figura 7 riporta lo screeplot della PCA eseguita con il parametro *scale = FALSE*.

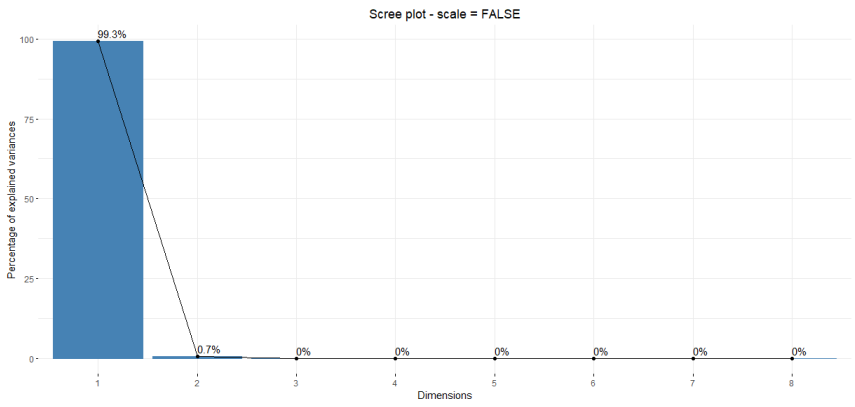


Figura 7: Screeplot della PCA - scale = FALSE

Dalla figura 7 è possibile osservare che

- La prima componente spiega il 99.3% della varianza;
- La seconda componente spiega lo 0.7% della varianza.

Dunque, è necessaria una sola componente per spiegare almeno il 90% della variabilità dei dati, per cui sarebbe ottimale effettuare il clustering usando 1 componente su 8.

La figura 8 mostra i contributi (in percentuale) delle variabili alle componenti principali. È possibile osservare che l'attributo **Total.TCP.Flow.Time** contribuisce maggiormente alla componente principale. Poiché per correlazione si intende il coefficiente di correlazione di Pearson, che ha un valore compreso tra 1 (perfetta correlazione) e -1 (perfetta anti-correlazione), non è possibile individuare eventuali correlazioni o anticorrelazioni tra gli attributi.

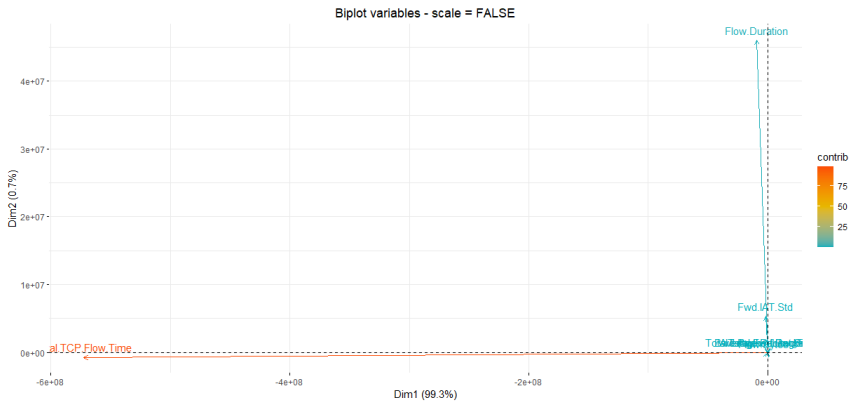


Figura 8: Biplot della PCA: contrib - scale = FALSE

Infine, è stato realizzato il grafico in figura 9 per determinare la qualità di rappresentazione delle variabili. L'attributo **Total.TCP.Flow.Time** prensenta un valore molto elevato, ciò pobrebbe significare che è ben rappresentato dalle due componenti principali. Tuttavia, non è possibile effettuare alcuna considerazione in quanto il cos^2 può assumere valori soltanto tra 0 e 1 e i valori riportati in figura sono molto maggiori di 1 (maggiore di 3×10^{17}).

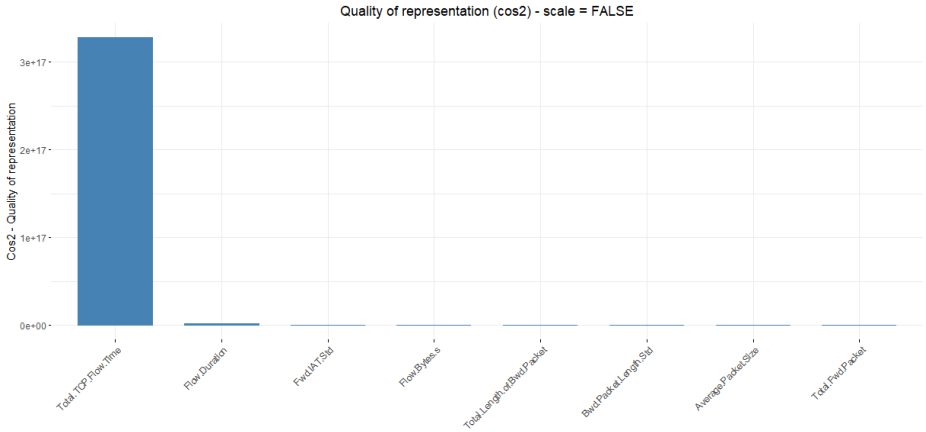


Figura 9: Biplot della PCA: cos2 - scale = FALSE

3.2 Variabili scalate ("scale = TRUE")

La figura 10 riporta lo screeplot della PCA eseguita con il parametro *scale = TRUE*.

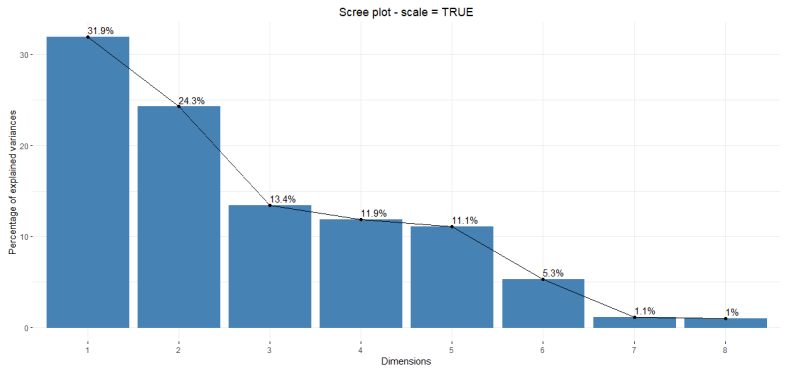


Figura 10: Screeplot della PCA - scale = TRUE

Dalla figura 10 è possibile osservare che

- La prima componente spiega il 31.9% della varianza;
- La seconda componente spiega il 24.3% della varianza.

Dunque, sono necessarie almeno cinque componenti per spiegare almeno il 90% della variabilità dei dati; infatti, la varianza spiegata dalle cinque componenti è pari al 92.6%.

La figura 11 mostra i contributi (in percentuale) delle variabili alle componenti principali. È possibile osservare che gli attributi che contribuiscono maggiornmente sono **Total.Fwd.Packet** e **Total.Length.of.Bwd.Packet**; invece gli attributi che contribuiscono di meno sono **Total.TCP.Flow.Time** e **Flow.Bytes.s**.

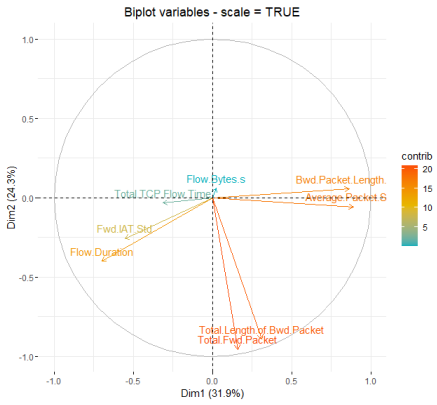


Figura 11: Biplot della PCA: contrib - scale = TRUE

Infine, è stato realizzato il grafico in figura 12 per determinare la qualità di rappresentazione delle variabili. È possibile osservare che gli attributi

- **Total.Fwd.Packet**
- **Total.Length.of.Bwd.Packet**
- **Average.Packet.Size**
- **Bwd.Packet.Length.Std**
- **Flow.Duration**

prensentano ognuno un valore molto elevato; ciò significa che sono ben rappresentati dalle due componenti principali.

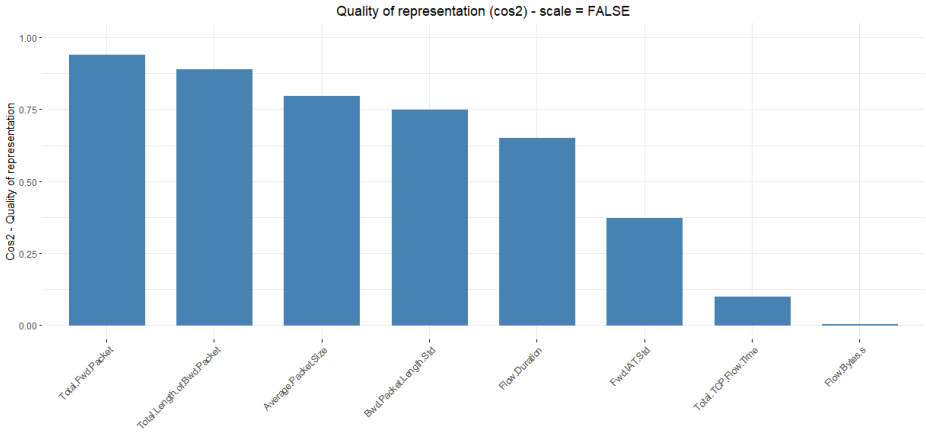


Figura 12: Biplot della PCA: cos2 - scale = TRUE

4 Analisi delle componenti correlate

Analizzando il biplot riportato in figura 13 sono state individuate le coppie di variabili correlate e anticorrelate.

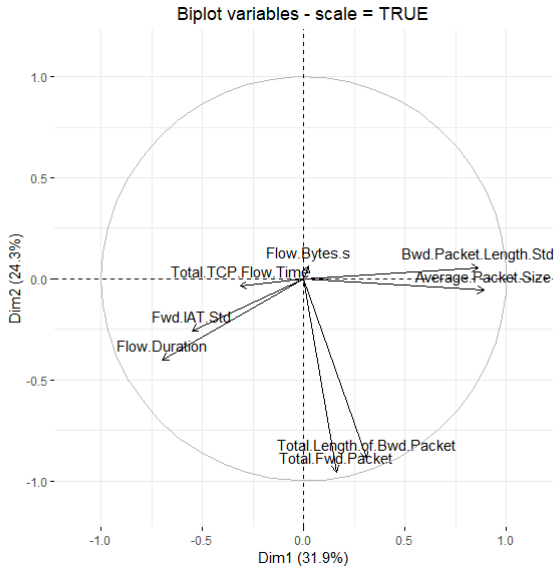


Figura 13: Biplot della PCA: scale = TRUE

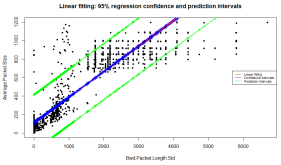
Il biplot fviz_pca_var(), della libreria factoextra, e un tipo di grafico che consente di analizzare le dipendenze tra attributi in base agli angoli formati dai vettori. Le variabili di interesse sono state individuate e selezionate utilizzando la seguente notazione:

- **Correlazione**: l'ampiezza dell'angolo compreso tra le variabili è un valore inferiore a 45°
- **Incorrelazione**: l'ampiezza dell'angolo compreso tra le variabili è un valore superiore a 45°
- **Anticorrelazione**: l'ampiezza dell'angolo compreso tra le variabili è tendente a 180°

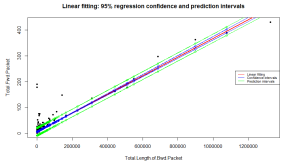
Per ogni coppia di variabili analizzate, è stata realizzato uno scatterplot e un'analisi di regressione per controllare se ci fosse un rapporto di proporzionalità lineare. La stessa operazione è stata svolta anche considerando degli intervalli di confidenza al 95% e degli intervalli di predizione. Infine è stato calcolato il coefficiente di determinazione R^2, come riportato nelle tabelle 1 e 2, per determinare la bontà della regressione: maggiore è il coefficiente di determinazione, maggiore è la bontà della regressione. Il coefficiente di determinazione è il rapporto della variabilità dovuta dalla regressione diviso quella totale ed è un valore che varia tra 0 e 1:

$$R^2 = \frac{SSR}{SST} = \frac{(SST - SSE)}{SST} \tag{2}$$

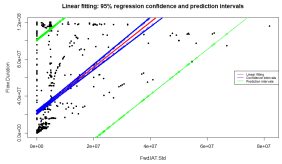
Nell'appendice C si riporta il codice .R utilizzato per effettuare l'analisi.



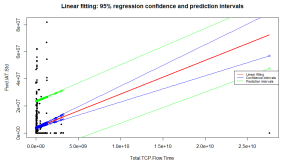
((a)) 1



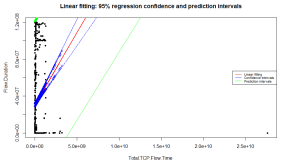
((b)) 2



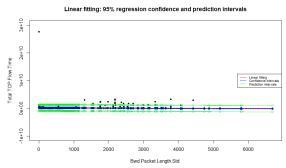
((c)) 3



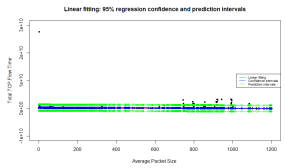
((d)) 4



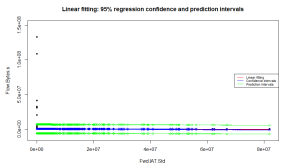
((e)) 5



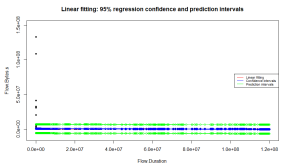
((f)) 6



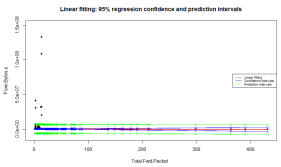
((g)) 7



((h)) 8



((i)) 9



((j)) 10

Figura 14: Regressione lineare

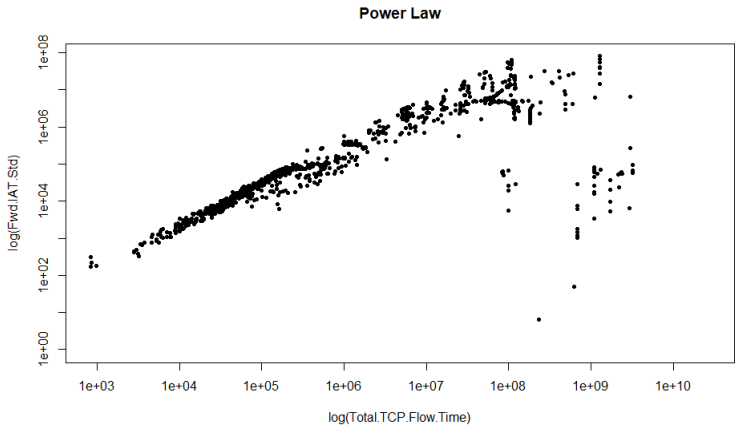
Id figura	Coppia di variabili	R2
14(a)	Bwd.Packet.Length.Std - Average.Packet.Size	0.8317416
14(b)	Total.Length.of.Bwd.Packet - Total.Fwd.Packet	0.7744556
14(c)	Fwd.IAT.Std - Flow.Duration	0.2763812
14(d)	Total.TCP.Flow.Time - Fwd.IAT.Std	0.01995415
14(e)	Total.TCP.Flow.Time - Flow.Duration	0.03653087

Tabella 1: R2 Variabili correlate

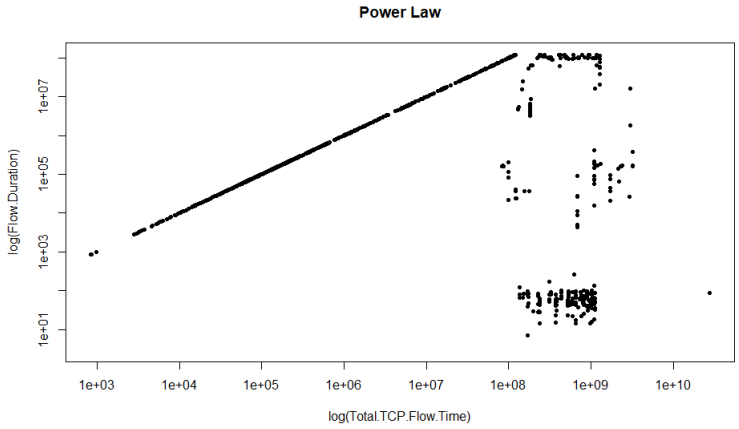
Id figura	Coppia di variabili	R2
14(f)	Bwd.Packet.Length.Std - Total.TCP.Flow.Time	0.01604434
14(g)	Average.Packet.Size - Total.TCP.Flow.Time	0.02660251
14(h)	Fwd.IAT.Std - Flow.Bytes.s	0.001760144
14(i)	Flow.Duration - Flow.Bytes.s	0.004827193
14(j)	Total.Fwd.Packet - Flow.Bytes.s	0.0001041532

Tabella 2: R2 Variabili anti-correlate

Dall'analisi grafica delle figure 14(d) e 14(e) si è potuto ricondurre le stesse a distribuzioni a legge di potenza (*Power Law*). Quindi è stato ritenuto opportuno riportare i grafici in scala logaritmica per visualizzare meglio eventuali rapporti di proporzionalità lineare (Appendice C.1).



((a)) 4 - Power Law



((b)) 5 - Power Law

Figura 15: Correlazioni di tipo Power Law

Dall'analisi grafica delle figure rappresentanti le variabili anticorrelate (figure 14(f), 14(g), 14(h), 14(i), 14(j)) è possibile notare un rapporto di proporzionalità lineare tra tutte le coppie evidenziate.

I valori presenti nelle tabelle 1 e 2, per la valutazione della bontà della regressione lineare, in alcuni casi risultano essere molto bassi a causa della presenza di outlier molto distanti dalla funzione di regressione.

5 Clustering del dataset trasformato

A seguito della riduzione del dataset è stato applicato il clustering k-means sul nuovo dataset avente le sole due dimensioni principali (Appendice D). L'esecuzione del clustering sul dataset trasformato offre diversi vantaggi:

- Consente di ridurre la dimensionalità dei dati, conservando le informazioni più significative. Di conseguenza, il numero di variabili sulle quali viene eseguito il clustering sarà ridotto, semplificando la computazione del clustering e migliorando la visualizzazione dei risultati.
- Può rimuovere la correlazione tra le variabili, rendendo i cluster più omogenei e separati. Ciò significa che il clustering eseguito sui dati ridotti dalla PCA potrebbe produrre cluster più distinti e significativi rispetto al clustering eseguito sui dati originali.
- Può ridurre il tempo necessario per l'esecuzione del clustering, in quanto si riduce il numero di variabili da considerare.
- Può aiutare a identificare le variabili più importanti per la formazione dei cluster, semplificando la comprensione dei risultati e la loro interpretazione.

Il clustering è stato applicato ai dataset ridotti ottenuti non scalando le variabili e scalando le variabili, quindi per scale=FALSE e scale=TRUE.

5.1 Clustering del dataset ridotto con variabili non scalate

La figura 16 mostra la distribuzione dei nuovi punti del dataset 2D.

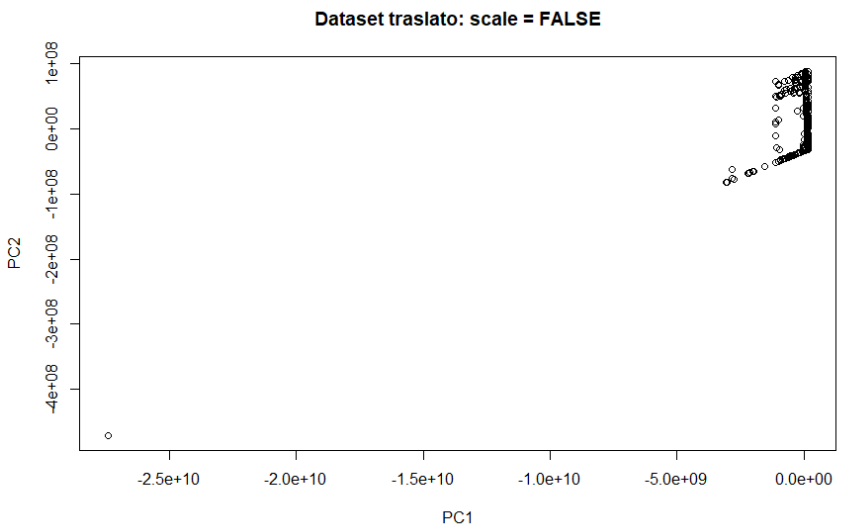


Figura 16: Scatterplot del dataset 2D - scale = FALSE

Per poter effettuare il clustering k-means, come primo step è stata svolta l'analisi di sensitività, ripetendo gli step già effettuati per il dataset di partenza (sezione 2.2). Osservando la curva in figura 17 è stato selezionato come numero ottimale di cluster il valore di k in corrispondenza del ginocchio della curva, ossia k=4. In seguito è stata analizzata al distribuzione dei punti nei cluster individuati(figura 18). Infine, sono stati raffigurati i risultati ottenuti dal clustering (figura 19).

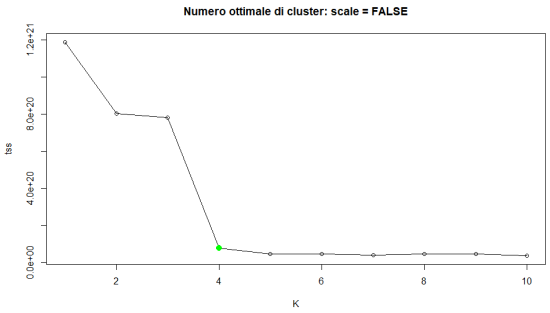


Figura 17: Scelta del valore di k

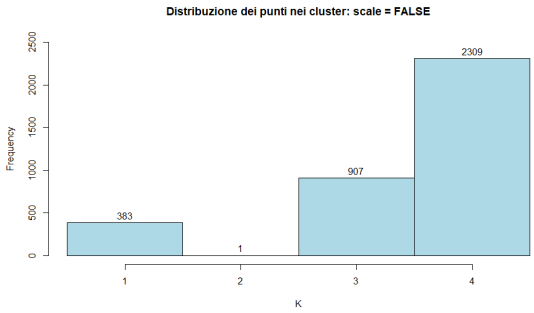


Figura 18: Distribuzione dei punti nei cluster

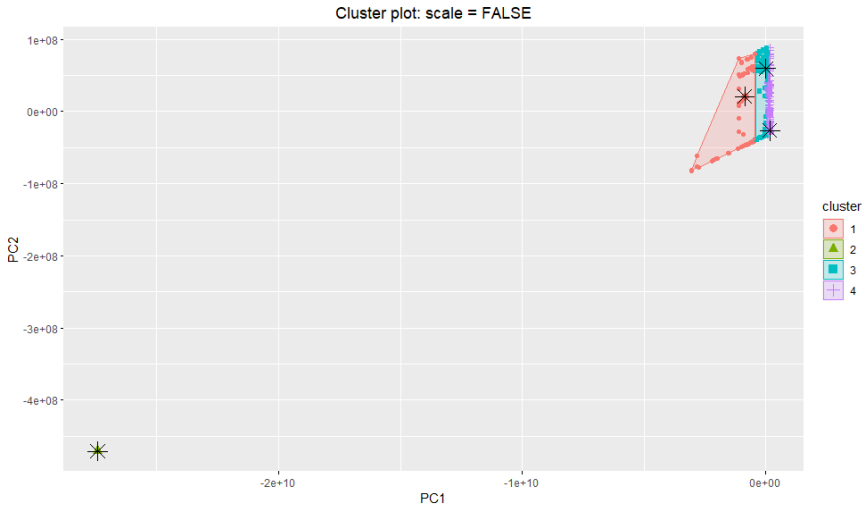


Figura 19: Risutlati del clustering

Osservando le figure 18 e 19 è possibile notare che il cluster 2 contiene al suo interno un solo elemento del dataset traslato. Questo cluster è considerato un outlier, di conseguenza l'elemento contenuto al suo interno è stato rimosso dal dataset ridotto ed è stato applicato nuovamento l'algorimo k-means. Anche in questo caso come numero ottimale di cluster è stato scelto k=4 (figura 20).

Si riportano la nuova distribuzione dei punti nei cluster (figura 21) e i nuovi risultati del clustering (figura 22).

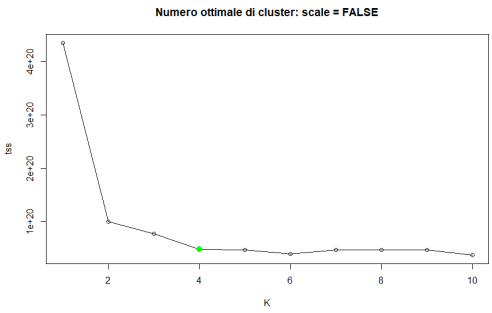


Figura 20: Scelta del valore di k senza outlier

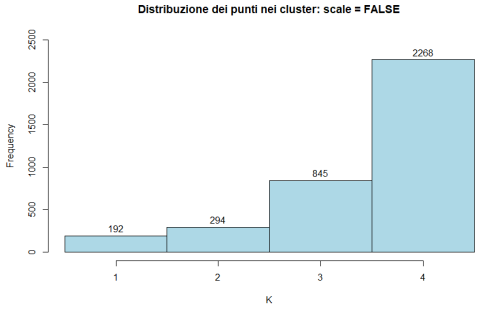


Figura 21: Distribuzione dei punti nei cluster senza outlier

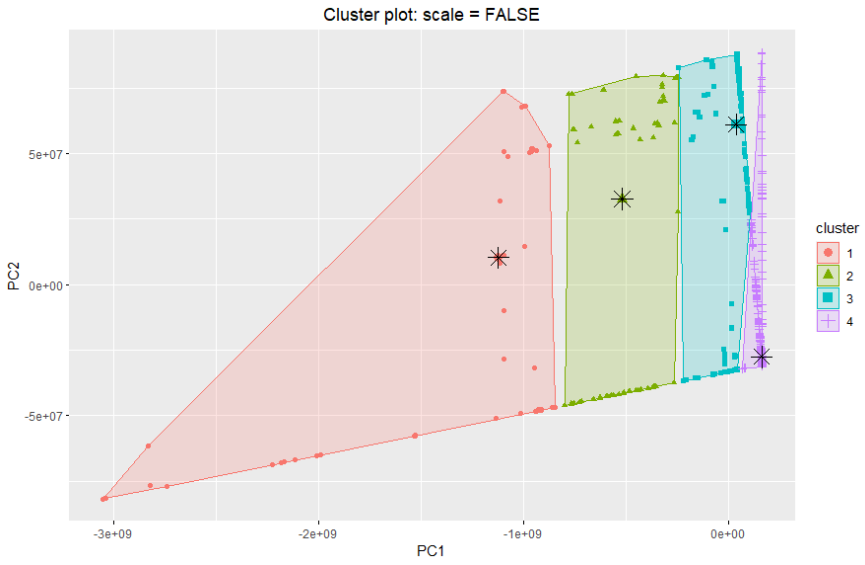


Figura 22: Risultati del clustering per il dataset non scalato e senza outlier

5.2 Clustering del dataset ridotto con variabili scalate

La figura 23 mostra la distribuzione dei nuovi punti del dataset 2D.

Figura 23: Scatterplot del dataset 2D - scale = TRUE

Per poter effettuare il clustering k-means, come primo step è stata svolta l'analisi di sensitività, ripetendo gli step già effettuti per il dataset di partenza (sezione 2.2). Osservando la curva in figura 24 è stato selezionato come numero ottimale di cluster il valore di k in corrispondenza del ginocchio della curva, ossia k=4. In seguito è stata analizzata al distribuzione dei punti nei cluster individuati(figura 25). Infine, sono stati raffigurati i risultati ottenuti dal clustering (figura 26).

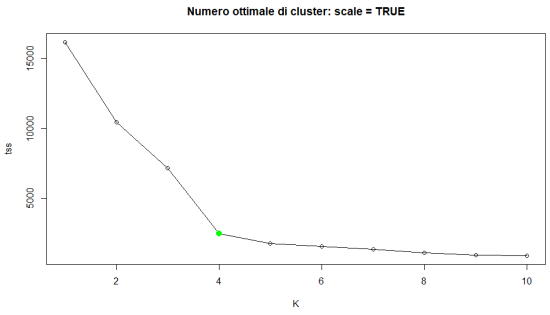


Figura 24: Scelta del valore di k

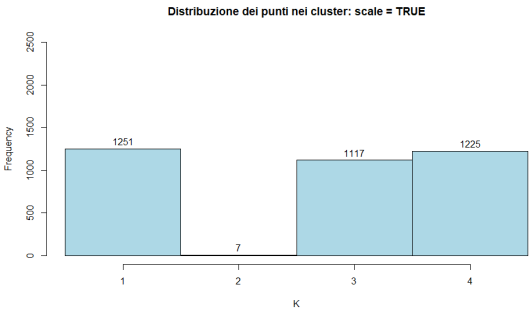


Figura 25: Distribuzione dei punti nei cluster

Osservando le figure 25 e 26 è possibile notare che il cluster 2 contiene al suo interno solo 7 elementi del dataset traslato. I cluster piccoli possono essere outlier; di conseguenza gli elementi appartenenti al cluster 2 sono stati rimossi dal dataset ridotto ed è stato applicato nuovamente l'algorimo k-means. Anche in questo caso come numero ottimale di cluster è stato scelto k=4 (27).

Figura 26: Risutlati del clustering

Si riportano la nuova distribuzione dei punti nei cluster (figura 28) e i nuovi risultati del clustering (figura 29).

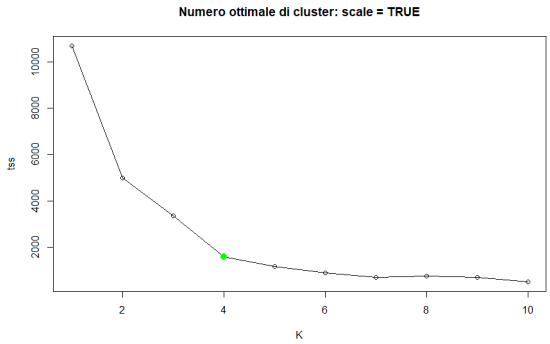


Figura 27: Scelta del valore di k per il dataset scalato e senza outlier

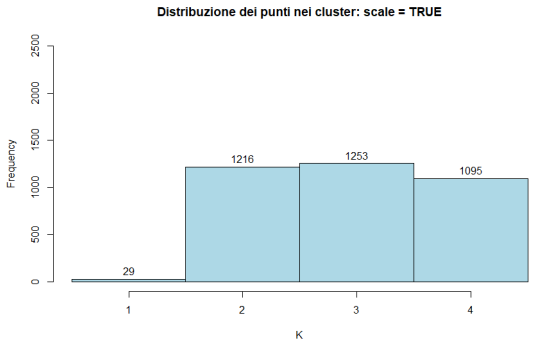


Figura 28: Distribuzione dei punti nei cluster per il dataset scalato e senza outlier

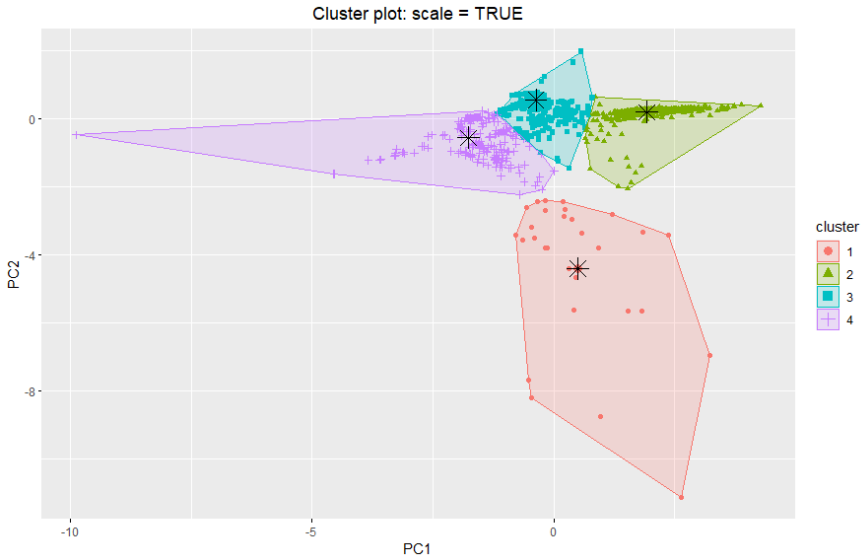


Figura 29: Risultati del clustering per il dataset scalato e senza outlier

6 Analisi dei risultati e conclusioni

La tabella 3 riporta nel dettaglio i risultati ottenuti dall'applicazione dell'algoritmo nei diversi casi considerati.

Clustering	**Configurazione**	K_1	K_2	K_3	K_4
Gerarchico	Dataset di partenza	2195	831	537	-
K-means	Dataset di partenza	907	383	2309	1
K-means	PC=2, scale=FALSE	383	1	907	2309
K-means	PC=2, scale=FALSE, outlier rimosso	192	294	845	2266
K-means	PC=2, scale=TRUE	1251	7	1117	1225
K-means	PC=2, scale=TRUE, outlier rimosso	29	1216	1253	1095

Tabella 3: Risultati del clustering nei diversi casi considerati

L'applicazione dell'algoritmo di clustering k-means prima e dopo la riduzione del dataset ha prodotto risultati simili:

- Il numero di cluster ottimale individuato nella maggior parte dei casi è risultato k=4;
- Per il dataset di partenza e per il dataset ridotto con variabili non scalate è stata individuata la stessa distribuzione. Inoltre, in entrmabi i casi è stato individuato un cluster contenente un solo elemento al suo interno. L'analisi del dataset ridotto ha consentito di classificare tale cluster come un outlier;
- Per il dataset ridotto con variabili non scalate è stato individuato un cluster contenente solo sette elementi al suo interno. Anche in questo caso l'analisi del dataset ridotto ha consentito di classificare tale cluster come un outlier;
- La rimozione degli outlier ha consentito un raggruppamento più preciso dei punti.

La riduzione della dimensionalità ha consentito di individuare gli outlier; infatti, la visualizzazione e molto piu efficace data la possibilità di visualizzare i valori delle osservazioni sulle sole due dimensioni individuate.
Tramite la clusterizzazione degli elementi è stato possibile individuare in modo agevole il traffico dati malevolo evidenziato nell'ultimo gruppo. Quest'ultimo è infatti riconoscibile proprio dagli elevati valori di "Total Length of Bwd Packet". E' possibile quindi affermare, sulla base dei risultati ottenuti, come l'algoritmo di clustering k-means riesca meglio a raggruppare e ad evidenziare valori anomali del traffico.

Appendice A Clustering

A.1 Clustering gerarchico

```
#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')

#Calcolo delle distanze euclidee:
#restituisce la matrice di distanze euclidee, cio tutte le possibili distanze tra coppie di punti.
distances<- dist(data, method="euclidean")

# algoritmo standard di clustering gerarchico:
#"ward.D" "single", "complete", "average" (= UPGMA), "mcquitty" (= WPGMA), "median" (= WPGMC) or
    "centroid" (= UPGMC).
fit <- hclust(distances, method="ward.D")

#plot del dendrogramma
plot(fit)

# utilizziamo la funzione rect.hclust per mettere i rettangoli
#k    il numero di cluster che individuiamo guardando il dendrogramma
rect.hclust(fit, k=3, border = "blue")

#Taglio dell'albero:
# luscita     fornita come identificativo del cluster di appartenenza per ogni punto.
groups<- cutree(fit, k=3)

#creazione dataframe
clusters<-groups
df<-clusters
df
#plot(data, col=g, pch=g, cex=2)

#distribuzione dei punti nei cluster
#hist(g, breaks=5, xlab = "k", main=paste("Distribuzione dei punti nei cluster" ), labels = TRUE,
    ylim=c(0,2500))
h<-hist(df, col="lightblue", labels = TRUE,
       breaks=seq(min(df)-1,max(df)),
       axes=F,
       ylim=c(0,2500),
       xlab="K",
       main="Distribuzione dei punti nei cluster")
axis(2)
#stampa il cluster di ogni bin)
axis(1,at=h$mids,seq(min(df),max(df)))
```

A.2 Clustering point-assignment

```
#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')
#definiamo un vettore di 10 elementi con un id crescente (tss=total squares sum):
#ipotizziamo di testare fino a 10 cluster (k=10)
#tss   la metrica che vogliamo misurare in base al numero di cluster.
tss<-seq(1,10,1)

# Test dei valori di k da 1 a 10.
for (i in 1:10) tss[i] <- kmeans(data,i)$tot.withinss
tss  #->   una metrica del totale della somma dei quadrati intra-cluster. Varia molto velocemente
     quando varia il numero di cluster.
plot(tss, type='o', xlab="K")

#si dovrebbe scegliere qualcosa che si trova in corripondenza del ginocchio
points( c(4), tss[which.min(tss>2e20)], pch=20, col="green", cex=2)
points( c(3), tss[which.min(tss>8e20)], pch=20, col="blue", cex=2)

res<-kmeans(data,3)

#restituisce gli id del cluster per i vari punti
res$cluster

#restituisce i centroidi
res$centers

#restituisce restituisce la somma dei quadrati delle distanze dei punti rispetto al centroide
     (intra-cluster). Se faccio tot.withinss ottengo la somma di questi valori.
res$withinss

#creazione dataframe
clusters_kmeans<-res$cluster
df_km<-clusters_kmeans
df_km

#distribuzione dei punti nei cluster
hist(res$cluster, xlab = "k", main="Distribuzione dei punti nei cluster")

h2<-hist(df_km, col="lightblue", labels = TRUE,
       breaks=seq(min(df_km)-1,max(df_km)),
       axes=F, main="Distribuzione dei punti nei cluster", xlab="K", ylim=c(0,2500))
axis(2)
axis(1,at=h2$mids,seq(min(df_km),max(df_km)))
```

Appendice B Principal Component Analysis

B.1 scale=FALSE

```
#librerie
library("FactoMineR")
library("factoextra")

#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')

data.pca <- prcomp(data, scale=FALSE)

eig.val <- get_eigenvalue(data.pca)
eig.val
#screeplot: i seguenti metodi sono identici
fviz_screeplot(data.pca, addlabels=TRUE,
               title="Scree plot - scale = FALSE") +
                theme(plot.title = element_text(hjust = 0.5))
fviz_eig(data.pca, addlabels = TRUE,
         title="Scree plot - scale = FALSE") +
          theme(plot.title = element_text(hjust = 0.5))

#biplot
fviz_pca_var(data.pca, col.var = "black",
             xlim=c(-6e8,75000), ylim=c(0,5e7),
             title="Biplot variables - scale = FALSE")+
               theme(plot.title = element_text(hjust = 0.5))
fviz_pca_var(data.pca, col.var = "black",
             xlim=c(-6e8,6e8), ylim=c(-6e8,6e8),
             title="Biplot variables - scale = FALSE")+
  theme(plot.title = element_text(hjust = 0.5))

#var$contrib: contains the contributions (in percentage) of the variables to the principal
     components.
fviz_pca_var(data.pca, col.var = "contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             title="Biplot variables - scale = FALSE") +
                theme(plot.title = element_text(hjust = 0.5))

#Total cos2 of variables on Dim.1 and Dim.2
fviz_cos2(data.pca, choice = "var", axes = 1:2,
          title="Quality of representation (cos2) - scale = FALSE")+
            theme(plot.title = element_text(hjust = 0.5))
```

B.2 scale=TRUE

```
#librerie
library("FactoMineR")
library("factoextra")

#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')

data.pca <- prcomp(data, scale=TRUE)

eig.val <- get_eigenvalue(data.pca)
eig.val
#screeplot: i seguenti metodi sono identici
fviz_screeplot(data.pca, addlabels=TRUE,
               title="Scree plot - scale = TRUE") +
  theme(plot.title = element_text(hjust = 0.5))
fviz_eig(data.pca, addlabels = TRUE,
         title="Scree plot - scale = TRUE") +
  theme(plot.title = element_text(hjust = 0.5))

#calcolo della varianza
sum(data.pca$sdev[1:5]^2) / sum (data.pca$sdev^2)

#biplot
fviz_pca_var(data.pca, col.var = "black",
             title="Biplot variables - scale = TRUE")+
  theme(plot.title = element_text(hjust = 0.5))


#var$contrib: contains the contributions (in percentage) of the variables to the principal
     components.
fviz_pca_var(data.pca, col.var = "contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             title="Biplot variables - scale = TRUE") +
  theme(plot.title = element_text(hjust = 0.5))

#Total cos2 of variables on Dim.1 and Dim.2
fviz_cos2(data.pca, choice = "var", axes = 1:2,
          title="Quality of representation (cos2) - scale = TRUE")+
  theme(plot.title = element_text(hjust = 0.5))
```

Appendice C Analisi delle componenti correlate

Per analizzare le componenti correlate, effettuare il fitting lineare, calcolare il coefficiente R^2 e individuare gli intervalli di confidenza e predizione è stato utilizzato un unico script R nel quale sono stati cambiati di volta in volta i valori di x e y che rappresentano i valori delle coppie considerate. Per semplicità si riporta il codice corrispondente alla prima coppia di variabili correlate.

```
#librerie
library("FactoMineR")
library("factoextra")

#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')

data.pca <- prcomp(data, scale=TRUE)

#biplot
fviz_pca_var(data.pca, col.var = "black",
             title="Biplot variables - scale = TRUE")+

  theme(plot.title = element_text(hjust = 0.5),
        aspect.ratio = 4/4,) +
  expand_limits(x=c(-1.12,1.12), y=c(-1.12,1.12))
       # coord_cartesian(xlim = c(-1.1,1.1))

# 1
# VARIABILI CORRELATE: Bwd.Packet.Length.Std - Average.Packet.Size
x<-data$Bwd.Packet.Length.Std
y<-data$Average.Packet.Size
plot(x, y, pch=20, xlab = ("Bwd.Packet.Length.Std"), ylab=("Average.Packet.Size"),
     main=("Linear fitting: 95% regression confidence and prediction intervals"))

l_mod<-lm(y~x)

lines(x,predict(l_mod), col="red", lwd=2)

# Coefficiente di determinazione
R2<-summary(l_mod)$r.squared
R2

c_int<- predict(l_mod, level = 0.95, interval="confidence")
p_int<- predict(l_mod, level = 0.95, interval="prediction")

lines(x, c_int[,2], type="o", lty=2, col="blue")
lines(x, c_int[,3], type="o", lty=2, col="blue")
lines(x, p_int[,2], type="o", lty=2, col="green")
lines(x, p_int[,3], type="o", lty=2, col="green")
legend( x="right",
        legend=c("Linear fitting", "Confidence intervals", "Prediction intervals"),
        col=c("red","blue", "green"), lwd=1, cex=0.7)
```

C.1 Analisi delle distribuzioni a legge di potenza

```
#librerie
library("FactoMineR")
library("factoextra")

#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')

data.pca <- prcomp(data, scale=TRUE)
data.pca
#biplot
fviz_pca_var(data.pca, col.var = "black",
             title="Biplot variables - scale = TRUE")+

  theme(plot.title = element_text(hjust = 0.5),
        aspect.ratio = 4/4,) +
  expand_limits(x=c(-1.12,1.12), y=c(-1.12,1.12))
       # coord_cartesian(xlim = c(-1.1,1.1))

# 4
# VARIABILI CORRELATE: Total.TCP.Flow.Time - Fwd.IAT.Std
x<-data$Total.TCP.Flow.Time
y<-data$Fwd.IAT.Std
plot(x, y, pch=20, xlab = ("log(Total.TCP.Flow.Time)"), ylab=("log(Fwd.IAT.Std)"),
     log="xy",
     main=("Power Law"))

# 5
# VARIABILI CORRELATE: Total.TCP.Flow.Time - Flow.Duration
x<-data$Total.TCP.Flow.Time
y<-data$Flow.Duration
plot(x, y, pch=20, xlab = ("log(Total.TCP.Flow.Time)"), ylab=("log(Flow.Duration)"),
     log="xy",
      main=("Power Law"))
```

Appendice D Clustering del dataset trasformato

D.1 Clustering del dataset ridotto con variabili non scalate

```
#librerie
library("FactoMineR")
library("factoextra")
library("ggplot2")

#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')

data.pca <- prcomp(data, scale=FALSE)

#dataset traslato
data.translated<- data.pca$x[,1:2]
plot(data.translated, main = "Dataset traslato: scale = FALSE")

############ k means
tss<-seq(1,10,1)

# Analisi di sensitivit.
for (i in 1:10) tss[i] <- kmeans(data.translated,i)$tot.withinss
plot(tss, type='o', xlab="K", main="Numero ottimale di cluster: scale = FALSE")

points( c(4), tss[which.min(tss>2e20)], pch=20, col="green", cex=2) #scale=FALSE

res<-kmeans(data.translated,4)

clusters_pca<-res$cluster

#distribuzione dei punti nei cluster
h2<-hist(clusters_pca, col="lightblue", labels = TRUE,
        breaks=seq(min(clusters_pca)-1,max(clusters_pca)),
        axes=F, main="Distribuzione dei punti nei cluster: scale = FALSE", xlab="K", ylim=c(0,2500))
axis(2)
axis(1,at=h2$mids,seq(min(clusters_pca),max(clusters_pca)))

fviz_cluster(res, data.translated, geom = c("point"),
             repel=TRUE, stand=FALSE,
             title="Cluster plot: scale = FALSE") +
             theme(plot.title = element_text(hjust = 0.5)) +
             geom_point(data = as.data.frame(res$centers),
             aes(x = res$centers[,1], y =res$centers[,2] ),
             size = 5, shape = 8)

#Rimozione degli outlier dal dataset
clusters<-table(res$cluster)<10

# Get the indices where the logical vector is TRUE
cluster_index <- which(clusters)
```

```
#rimozione del cluster con indice cluster_index dal dataset trasformato
data.translated2<-subset(data.translated, (res$cluster) != cluster_index)

#data.translated2 <- data.translated[-rows_to_remove,]
dim(data.translated2)

plot(data.translated2, main = "Dataset traslato: scale = False")

############ k means sul dataset senza outlier
tss<-seq(1,10,1)

# Test dei valori di k da 1 a 10.
for (i in 1:10) tss[i] <- kmeans(data.translated2,i)$tot.withinss
plot(tss, type='o', xlab="K", main="Numero ottimale di cluster: scale = FALSE")

res<-kmeans(data.translated2,4) #non va bene perch restituisce un cluster costituito da un unico
    punto.

clusters_kmeans<-res$cluster

#distribuzione dei punti nei cluster
h2<-hist(clusters_kmeans, col="lightblue", labels = TRUE,
        breaks=seq(min(clusters_kmeans)-1,max(clusters_kmeans)),
        axes=F, main="Distribuzione dei punti nei cluster: scale = FALSE", xlab="K", ylim=c(0,2500))
axis(2)
axis(1,at=h2$mids,seq(min(clusters_kmeans),max(clusters_kmeans)))

# Visualize clusters
fviz_cluster(res, data.translated2, geom = c("point"),
           repel=TRUE, stand=FALSE,
           #xlim=c(-5,5), ylim=c(-12,3),
           title="Cluster plot: scale = FALSE") +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_point(data = as.data.frame(res$centers),
           aes(x = res$centers[,1], y =res$centers[,2] ),
           size = 5, shape = 8)
```

D.2 Clustering del dataset ridotto con variabili scalate

```
#librerie
library("FactoMineR")
library("factoextra")

#lettura dei dati flows.csv
data<- read.csv('flows/flows1.csv')

data.pca <- prcomp(data, scale=TRUE)

#dataset traslato
data.translated<- data.pca$x[,1:2]
plot(data.translated, main = "Dataset traslato: scale = TRUE")

############ k means
tss<-seq(1,10,1)

# Test dei valori di k da 1 a 10.
for (i in 1:10) tss[i] <- kmeans(data.translated,i)$tot.withinss
plot(tss, type='o', xlab="K", main="Numero ottimale di cluster: scale = TRUE")

points( c(4), tss[which.max(tss<4000)], pch=20, col="green", cex=2) #scale=TRUE

res<-kmeans(data.translated,4)

clusters_kmeans<-res$cluster

#distribuzione dei punti nei cluster
h2<-hist(clusters_kmeans, col="lightblue", labels = TRUE,
        breaks=seq(min(clusters_kmeans)-1,max(clusters_kmeans)),
        axes=F, main="Distribuzione dei punti nei cluster: scale = TRUE", xlab="K", ylim=c(0,2500))
axis(2)
axis(1,at=h2$mids,seq(min(clusters_kmeans),max(clusters_kmeans)))

# Visualize clusters
fviz_cluster(res, data.translated, geom = c("point"),
             repel=TRUE, stand=FALSE,
             #xlim=c(-5,5), ylim=c(-12,3),
             title="Cluster plot: scale = TRUE") +
             theme(plot.title = element_text(hjust = 0.5)) +
             geom_point(data = as.data.frame(res$centers),
             aes(x = res$centers[,1], y =res$centers[,2] ),
             size = 5, shape = 8)

#rimozione degli outlier dal dataset ridotto
clusters<-table(res$cluster)<10

# Get the indices where the logical vector is TRUE
cluster_index <- which(clusters)

#rimozione del cluster con indice cluster_index dal dataset trasformato e dai risultati del kmeans
```

```
data.translated2<-subset(data.translated, (res$cluster) != cluster_index)
dim(data.translated2)

################################################

plot(data.translated2, main = "Dataset traslato: scale = TRUE")

############ k means
tss<-seq(1,10,1)

# Test dei valori di k da 1 a 10.
for (i in 1:10) tss[i] <- kmeans(data.translated2,i)$tot.withinss
plot(tss, type='o', xlab="K", main="Numero ottimale di cluster: scale = TRUE")

points( c(4), tss[which.max(tss<2000)], pch=20, col="green", cex=2) #scale=TRUE

res<-kmeans(data.translated2,4) #non va bene perch restituisce un cluster costituito da un unico
    punto.

clusters_kmeans<-res$cluster

#distribuzione dei punti nei cluster
h2<-hist(clusters_kmeans, col="lightblue", labels = TRUE,
        breaks=seq(min(clusters_kmeans)-1,max(clusters_kmeans)),
        axes=F, main="Distribuzione dei punti nei cluster: scale = TRUE", xlab="K", ylim=c(0,2500))
axis(2)
axis(1,at=h2$mids,seq(min(clusters_kmeans),max(clusters_kmeans)))

# Visualize clusters
fviz_cluster(res, data.translated2, geom = c("point"),
           repel=TRUE, stand=FALSE,
           #xlim=c(-5,5), ylim=c(-12,3),
           title="Cluster plot: scale = TRUE") +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_point(data = as.data.frame(res$centers),
           aes(x = res$centers[,1], y =res$centers[,2] ),
           size = 5, shape = 8)
```