Machine Learning Project

Customer Personality Analysis

Mario Avolio
880995

Rocco Gianni Rapisarda
845197

22 gennaio 2022

Sommario

L'apprendimento automatico e la statistica sono discipline strettamente collegate. Secondo Michael I. Jordan, le idee dell'apprendimento automatico, dai principi metodologici agli strumenti teorici, sono stati sviluppati prima in statistica. In questo elaborato si riporta l'attività di sviluppo e sperimentazione di diversi modelli di **Machine Learning** per l'analisi approfondita dei clienti ideali per una generica azienda. La concentrazione è stata focalizzata soprattutto sul **Clustering** mediante l'algoritmo **K-Means**, sebbene nel corso della trattazione si esporranno anche altre metodologie utilizzate per l'analisi dei dati.

1 Descrizione del dominio di riferimento e obiettivi dell'elaborato

Molto spesso lo sviluppo di nuovi prodotti o servizi è attivato dall'imitazione dei concorrenti e da analisi di mercato generiche, mentre al cliente si dedica poca attenzione. L'acquisto è prima di tutto un'esperienza ed è necessario comprendere quali bisogni la guidano: solo così ogni segmento di mercato individuato sarà connesso con la capacità dell'azienda di soddisfare le aspettative dei clienti, comprese quelle inespresse. Progettare, sviluppare e vendere prodotti non connessi con il proprio target rappresenta un costo insostenibile, mentre è necessario progettare uno sviluppo in linea con la *customer satisfaction*. Per questo motivo Customer Personality Analysis riguarda un'analisi dettagliata dei clienti ideali per una generica azienda. Il compito fondamentale è quello di aiutare un'attività commerciale a comprendere meglio i propri compratori al fine di rendere più semplice la modifica e la scelta dei propri prodotti, in relazione alle esigenze richieste dagli acquirenti. L'obiettivo che ha spinto ad analizzare questo insieme di dati è inerente alle **diverse** personalità e comportamenti che gli acquirenti assumono durante il ruolo di potenziali clienti aziendali. Per questo motivo le aziende non possono adottare lo stesso approccio per ogni tipologia di plausibile compratore.

2 Scelte di design

Per gestire al meglio la giusta separazione tra gli elementi del progetto si è deciso di sfruttare un particolare pattern strutturale definito dallo schema sottostante. Il modello, cattura il comportamento dell'applicazione in termini di dominio del problema e gestisce direttamente i dati, la logica e le regole del progetto.

```
Customer Personality Analysis Project
├── ML-Porject.Rproj
├── Data
│   └── marketing_campaign.csv
├── DOC
│   ├── Presentation
│   │   └── ...
│   └── Report
│       └── ...
├── Script
│   ├── D-TREE.R
│   ├── DescriptionOfData.R
│   ├── EDA.R
│   ├── K-MEANS.R
│   ├── PCA.R
│   ├── D-TREE-Model Evaluation.R
│   └── Functions
│       └── Functions.R
├── Output
│   ├── Plots
│   │   └── ...
│   └── Data
│       └── ...
├── Other
│   └── ...
└── README.MD
```

E' bene precisare che all'interno del file **README.MD** sono precisate tutte le librerie utilizzate.

Data

Questa è la sottocartella in cui vengono salvati tutti i file che devono essere letti in R per eseguire l' analisi. Essi potrebbero essere file SPSS (*.sav), file Excel/CSV, file .FST o .RDS. L'idea chiave è quella di trattare, all'interno di questa cartella, file di dati di origine che in nessun momento R dovrebbe salvare o modificare al fine di garantirne la riproducibilità. In particolare si può notare l'esistenza del dataset analizzato all'interno di questa cartella.

Script

All'interno di questa sottocartella è possibile notare tutti gli script per l'analisi dei dati. Ogni script è opportunamente collegato ai precedenti per favoreggiarne l'esecuzione immediata. Si può anche notare la cartella *Functions* dove è possibile trovare una serie di funzioni di utilità per la chiarezza e la compattezza del codice.

Output

All'interno di questa sottocartella è doveroso notare l'insieme dei dati in output dall'esecuzione del codice. In particolare sono stati inseriti numerosi grafici nati dall'analisi del dataset.

3 Descrizione dei Dati

3.1 Attributi

Si fornisce la descrizione originale degli attributi analizzati.

People

- ID: Customer's unique identifier
- Year_Birth: Customer's birth year
- Education: Customer's education level
- Marital_Status: Customer's marital status
- Income: Customer's yearly household income
- Kidhome: Number of children in customer's household
- Teenhome: Number of teenagers in customer's household
- Dt_Customer: Date of customer's enrollment with the company
- Recency: Number of days since customer's last purchase
- Complain: 1 if the customer complained in the last 2 years, 0 otherwise

Products

- MntWines: Amount spent on wine in last 2 years
- MntFruits: Amount spent on fruits in last 2 years
- MntMeatProducts: Amount spent on meat in last 2 years
- MntFishProducts: Amount spent on fish in last 2 years
- MntSweetProducts: Amount spent on sweets in last 2 years
- MntGoldProds: Amount spent on gold in last 2 years

Promotion

- NumDealsPurchases: Number of purchases made with a discount
- AcceptedCmp1: 1 if customer accepted the offer in the 1st campaign, 0 otherwise
- AcceptedCmp2: 1 if customer accepted the offer in the 2nd campaign, 0 otherwise
- AcceptedCmp3: 1 if customer accepted the offer in the 3rd campaign, 0 otherwise
- AcceptedCmp4: 1 if customer accepted the offer in the 4th campaign, 0 otherwise
- AcceptedCmp5: 1 if customer accepted the offer in the 5th campaign, 0 otherwise
- Response: 1 if customer accepted the offer in the last campaign, 0 otherwise

Place

- NumWebPurchases: Number of purchases made through the company's website
- NumCatalogPurchases: Number of purchases made using a catalogue
- NumStorePurchases: Number of purchases made directly in stores
- NumWebVisitsMonth: Number of visits to company's website in the last month

La tabella 1 fornisce un'iniziale descrizione della tipologia di variabili presenti nel dataset.

	sapply(dataSet, class)
ID	integer
Year_Birth	integer
Education	character
Marital_Status	character
Income	integer
Kidhome	integer
Teenhome	integer
Dt_Customer	character
Recency	integer
MntWines	integer
MntFruits	integer
MntMeatProducts	integer
MntFishProducts	integer
MntSweetProducts	integer
MntGoldProds	integer
NumDealsPurchases	integer
NumWebPurchases	integer
NumCatalogPurchases	integer
NumStorePurchases	integer
NumWebVisitsMonth	integer
AcceptedCmp3	integer
AcceptedCmp4	integer
AcceptedCmp5	integer
AcceptedCmp1	integer
AcceptedCmp2	integer
Complain	integer
Z_CostContact	integer
Z_Revenue	integer
Response	integer

Tabella 1: Output funzione *sapply(dataSet, class)*

3.2 Prime analisi

Prima di fornire un'analisi dettagliata degli elementi del dataset si è ritenuto necessario effettuare una prima ispezione di alto livello, senza entrare nel dettaglio di ciascun attributo. Il dataset viene importato mediante la funzione:

```
dataSet <- read.csv(paste(getwd(),"/Data/marketing_campaign.csv",sep = ""), header=TRUE,
    sep="\t", stringsAsFactors=F) # use TAB as separator!
```

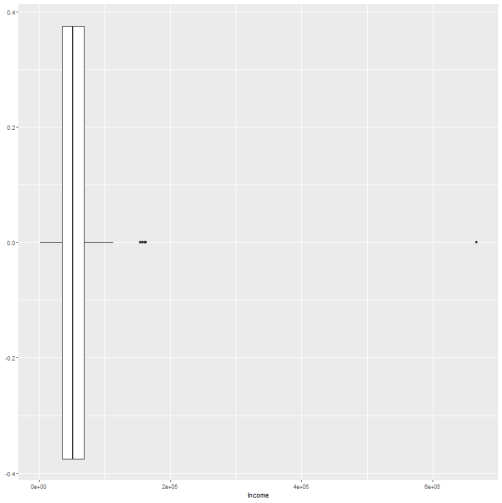


Figura 1: BoxPlot Income

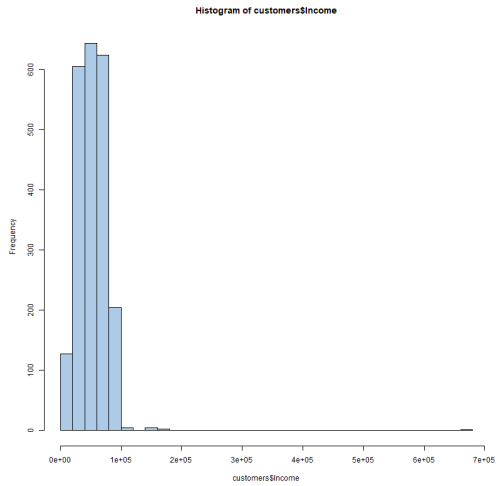


Figura 2: Istogramma di Income

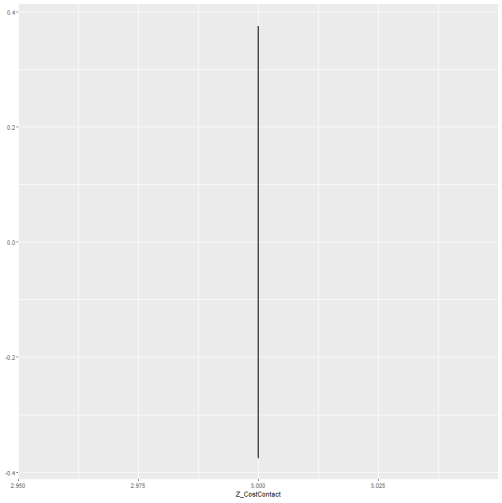


Figura 3: BoxPlot Z_CostContact

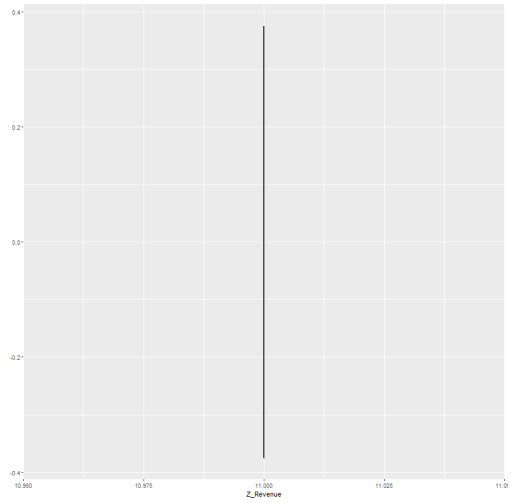


Figura 4: BoxPlot Z_Revenue

Durante questa prima indagine sono stati effettuati controlli di base sulle variabili. In particolare, dalle analisi dei boxplot riportati rispettivamente nelle figure 4 e 3 è doveroso notare la mancanza di **varianza** di tali variabili, questo aspetto sarà successivamente preso in considerazione durante la fase di preprocessing.
Inoltre durante l'analisi delle figure 1 e 2, mediante un apposito *warning* durante l'esecuzione del codice R, si è notata la presenza di alcuni valori mancanti.

```
hist(dataSet$Income,40,col="#adcae6")
ggplot(dataSet, aes(y = Income)) + geom_boxplot()
n_miss(dataSet) # counting the total number of missing values in the data
miss_var_summary(dataSet) # Summarizing missingness in each variable
```

La tabella 2 mostra l'output del comando *miss_var_summary(dataSet)*, dove si possono notare 24 valori mancanti nella variabile *Income*.

	variable	n_miss	pct_miss
1	Income	24	1.07
2	ID	0	0.00
3	Year_Birth	0	0.00
4	Education	0	0.00
5	Marital_Status	0	0.00
6	Kidhome	0	0.00
7	Teenhome	0	0.00
8	Dt_Customer	0	0.00
9	Recency	0	0.00
10	MntWines	0	0.00
11	MntFruits	0	0.00
12	MntMeatProducts	0	0.00
13	MntFishProducts	0	0.00
14	MntSweetProducts	0	0.00
15	MntGoldProds	0	0.00
16	NumDealsPurchases	0	0.00
17	NumWebPurchases	0	0.00
18	NumCatalogPurchases	0	0.00
19	NumStorePurchases	0	0.00
20	NumWebVisitsMonth	0	0.00
21	AcceptedCmp3	0	0.00
22	AcceptedCmp4	0	0.00
23	AcceptedCmp5	0	0.00
24	AcceptedCmp1	0	0.00
25	AcceptedCmp2	0	0.00
26	Complain	0	0.00
27	Z_CostContact	0	0.00
28	Z_Revenue	0	0.00
29	Response	0	0.00

Tabella 2: Output funzione *miss_var_summary(dataSet)*

3.3 Data Preprocessing

La fase di *data preprocessing* è risultata fondamentale per la buona riuscita dell'analisi dei dati preposti. Essa si basa su quattro principali stadi:

- Refactor del dataset
- Risoluzione dei valori mancanti nella variabile *income*
- Splitting del dataset in *trainingSet* e *testSet*
- Feature Scaling

3.3.1 Refactor del Dataset

In questa fase ci si è concentrati su diversi aspetti migliorativi. In primo luogo si è voluto essettuare un'azione di incorporamento di dati. La motivazione è dovuta principalmente alla presenza di dati ridondati all'interno di molte variabili, in particolare si vuole porre l'attenzione su: *Marital_Status* e *Education*. Difatti utilizzando la funzione *unique*, come riportato nelle tabelle 4 e 3, si è potuto rilevare la presenza di troppi elementi superflui.

	unique(dataSet$Marital_Status)
1	Single
2	Together
3	Married
4	Divorced
5	Widow
6	Alone
7	Absurd
8	YOLO

Tabella 3: Output *unique*(*dataSet*$*Marital_Status*)

	unique(dataSet$Education)
1	Graduation
2	PhD
3	Master
4	Basic
5	2n Cycle

Tabella 4: Output *unique*(*dataSet*$*Education*)

Come mostrato dal codice sottostante, l'obiettivo è stato quello di diminuire, per favorire l'analisi mediante i diversi algoritmi utilizzati, in numero delle categorie di valori per le relative variabili.

```
# ------------------------------------ COLLAPSING
#Collapsing marital Status into two categories: Single & Couple
unique(dataSet$Marital_Status)
dataSet <- mutate(dataSet, Marital_Status = replace(Marital_Status, Marital_Status ==
    "Divorced" | Marital_Status == "Widow" | Marital_Status == "Alone" | Marital_Status
    == "Absurd" | Marital_Status == "YOLO", "Single"))
```

```
dataSet <- mutate(dataSet, Marital_Status = replace(Marital_Status, Marital_Status ==
    "Together" | Marital_Status == "Married", "Couple"))

#Collapsing the Education into two Categories: graduate and non-graduate
unique(dataSet$Education)
dataSet <- mutate(dataSet, Education = replace(Education, Education == "Graduation"|
    Education == "PhD" | Education == "Master", "graduate"))
dataSet <- mutate(dataSet, Education = replace(Education, Education == "Basic"|
    Education == "2n Cycle", "non-graduate"))
# ---------------------------------------
```

In particolar modo si è deciso di fornire due possibili valori riassuntivi per ciascuna variabile, come indicato nelle tabelle 5 e 6.

	unique(dataSet$Education)
1	graduate
2	non-graduate

Tabella 5: Output *unique*(*dataSet*$*Education*) dopo la procedura di *collapsing* dei dati

	unique(dataSet$Marital_Status)
1	Single
2	Couple

Tabella 6: Output *unique*(*dataSet*$*Marital_Status*) dopo la procedura di *collapsing* dei dati

La fase di *refactoring* si è anche occupata della convesione in *factor* degli elementi *character* all'interno dell'insieme di dati. Il codice sottostante mostra la procedura seguita. Le tabelle 8 e 7 mostrano il risultato di tale procedura.

```
# ------------------------------------ CONVERSION
#Converting them to factors
dataSet <- mutate(dataSet, Marital_Status = as.factor(Marital_Status), Education =
    as.factor(Education))

# Encoding the categorical features to numeric
dataSet <- mutate(dataSet, Education = case_when(Education == "graduate" ~ 1,
                                                 Education == "non-graduate" ~ 0))
dataSet <- mutate(dataSet, Marital_Status = case_when(Marital_Status == "Couple" ~ 1,
                                                      Marital_Status == "Single" ~ 0))
```

	head(dataSet$Marital_Status)
1	0.00
2	0.00
3	1.00
4	1.00
5	1.00
6	1.00

Tabella 7: Output *head*(*dataSet*$*Marital_Status*)

	head(dataSet$Education)
1	1.00
2	1.00
3	1.00
4	1.00
5	1.00
6	1.00

Tabella 8: Output *head(dataSet$Education)*

Questa fase si è anche occupata della creazione di nuove variabili partendo da quelle già presenti all'interno da quelle già esistenti. Si ponga l'attenzione in particolar modo alle **categorie** di variabili *Mnt*, *Accepted*, *KidHome*, *TeenHome*. Tali categorie possono essere sommate per creare nuove variabili riassuntive. Il codice seguente e la tabella 9 ne riportano un esempio:

```
# ------------------------------------ TOTAL
#Creating a new variable:Total_spent
dataSet <- mutate(dataSet, Total_spent = MntWines + MntFruits + MntMeatProducts +
    MntFishProducts + MntSweetProducts + MntGoldProds)

# Details about previous campains also combined. Creating a new variable:Total_Campains
dataSet <- mutate(dataSet, Total_Campains = AcceptedCmp1 + AcceptedCmp2 + AcceptedCmp3 +
    AcceptedCmp4 + AcceptedCmp5)

# These variables can be combined and we can get the no of children for the dataSet.
    Creating a new variable:Total_Childs
dataSet <- mutate(dataSet, Total_Childs = Kidhome + Teenhome)
# ------------------------------------
```

	Total_spent	Total_Campains	Total_Childs
1	1617	0	0
2	27	0	2
3	776	0	0
4	53	0	1
5	422	0	1
6	716	0	1

Tabella 9: Primi valori delle variabili Total_spent & Total_Campains & Total_Childs

Per finire è doveroso sottolineare che in questa sezioni ci si è anche occupati dell'eliminazione delle variabili superflue, come *Z_CostContact* e *Z_Revenue* che non hanno varianza, e della sostituzione della varibile *Year_Birth* con *Age*.

```
# we can calculate customer age from the birth year. It will be more usefull to our
    analysis.
# creating a new variable Age from Year of Birth
thisYear <- as.numeric(format(as.Date(Sys.Date(), format="%d-%m-%Y"),"%Y"))
thisYear
dataSet <- mutate(dataSet, Age = thisYear - Year_Birth)
#Dropping some redundant features
dataSet <- select(dataSet, - ID, - Year_Birth, - Z_CostContact, - Z_Revenue,
    -Dt_Customer)
```

3.3.2 Risoluzione dei valori mancanti

Come mostrato nel codice sottostante, non si è deciso di eliminare completamente i valori mancanti all'interno della variabile *income*, bensì si è adottata un'altra strategia: la sostituzione con il valore medio della variabile stessa.

```
dataSet$Income <- ifelse(is.na(dataSet$Income), # is.na check is a value is not available
                         ave(dataSet$Income, FUN = function(x) mean(x, na.rm = TRUE)), #
                             if is not available change with average
                         dataSet$Income # else )
```

3.3.3 Splitting in TrainingSet e TestSet

Al fine di una buona analisi dei dati, si è deciso di suddividere il dataset iniziale in due più ridotti: l'insieme di allenamento e quello di testing. Il codice sottostante mostra la procedura effettuata.

```
set.seed(17538)
split <- sample.split(dataSet$Response, SplitRatio = 0.8)
trainingSet <- subset(dataSet, split == TRUE)
testSet <- subset(dataSet, split == FALSE)
```

3.3.4 Feature Scaling

E' opportuno sottolineare che si è deciso di normalizzare i valori delle variabili al fine di poter utilizzare al meglio gli algoritmi di machine learning che verranno descritti nel corso di questa trattazione. Il codice seguente mostra un esempio:

```
trainingSet_scaled <- as.data.frame(scale(trainingSet[, getIndipendentNumbersOfCol()]))
testSet_scaled <- as.data.frame(scale(testSet[, getIndipendentNumbersOfCol()]))
dataSet_scaled <- as.data.frame(scale(dataSet[, getIndipendentNumbersOfCol()]))
```

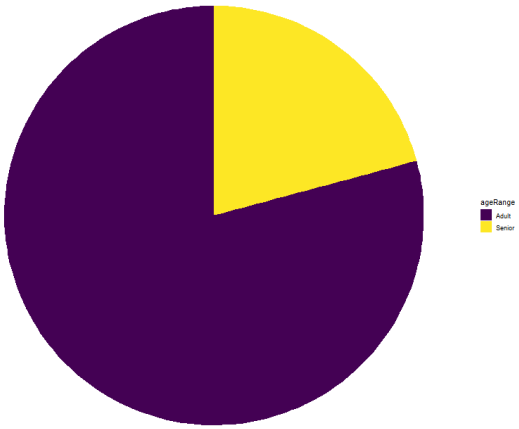
3.4 EDA

Dopo aver eseguito il preprocessing dei dati si è passati ad un'analisi esplorativa dei dati. Le prime variabili che sono state analizzate sono state *Age, Income* e *Total_Spent* rappresentando i grafici a torta.

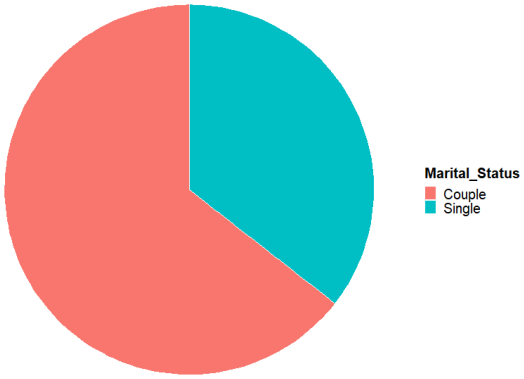
```
# Age Range
ageRange <- cut(dataSet$Age, breaks = c(24, 64, Inf), include.lowest = T,
                ordered_result = T, labels = c("Adult", "Senior"))
dataSet <- mutate(dataSet, Age_range = ageRange)
# Income Range
incomeRange <- cut(dataSet$Income,
                   calculateBreaksFromSummary(dataSet$Income),
                   labels = c("low", "low medium", "medium high", "high"))
dataSet <- mutate(dataSet, Income_range = incomeRange)
# Spent Range
spentRange <- cut(dataSet$Total_spent,
                  calculateBreaksFromSummary(dataSet$Total_spent),
                  labels = c("low", "low medium", "medium high", "high"))
dataSet <- mutate(dataSet, Spent_range = spentRange)
```

Il primo grafico a torta 5(a) è relativo alla variabile Age, il colore viola rappresenta il valore *Adult* mentre il colore giallo *Senior*. Dal grafico si ricava che la maggior parte degli individui

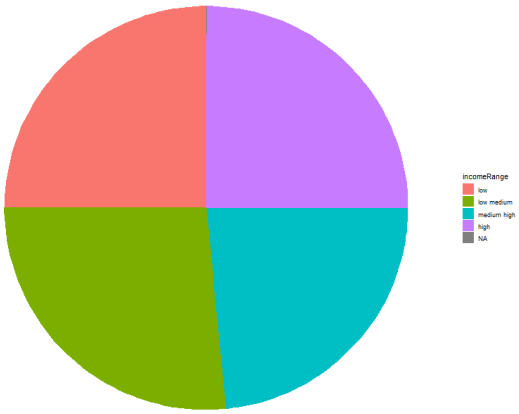
è *Adult*. Il secondo, il grafico 5(c) rappresenta la variabile *income*, il dataset è equi-distribuito in questo caso. La distribuizione dei valori che assume la variabile *Total spent* è raffigurata nella Figura 5(c). Da essa si ricava che low e high sono simili mentre il valore più frequente è *low medium*.



(a) PiePlot Age



(b) PiePlot Marital_Status



(c) PiePlot di Income

Inoltre si può notare facendo l'istogramma, figura 5 della variabile *Age* che l'età media degli individui presenti nel dataset è superiore ai 50 anni e dalla figura 7 si ha che la maggioranza ha un titolo di studio maggiore o uguale alla laurea e che hanno uno stipendio medio pari a medio alto, *medium-high*, mentre i non laureati hanno uno stipendio medio pari a medio basso, *low-medium*

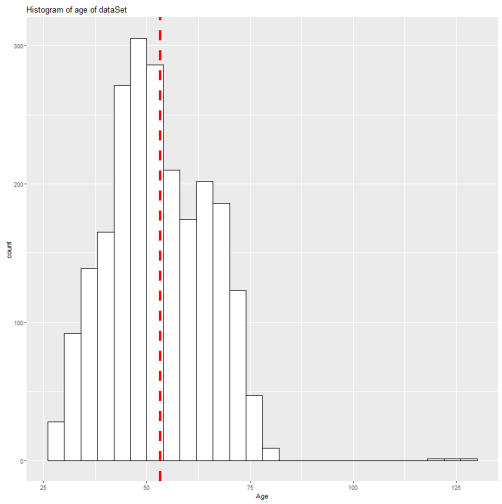


Figura 5: Istogramma di Age.

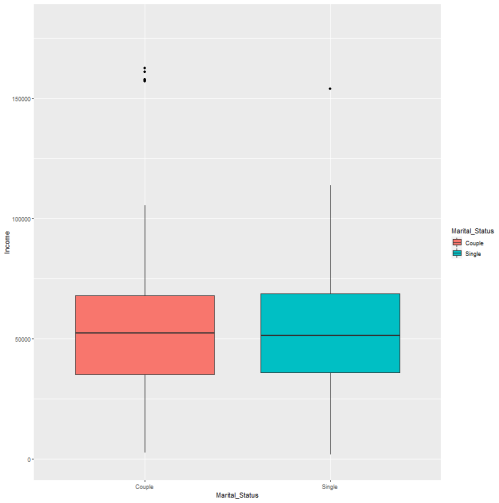


Figura 6: BoxPlot di Income e Marital_Status.

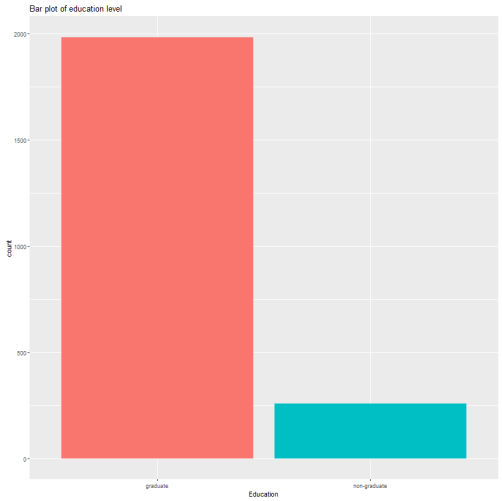


Figura 7: Istogramma di Education.

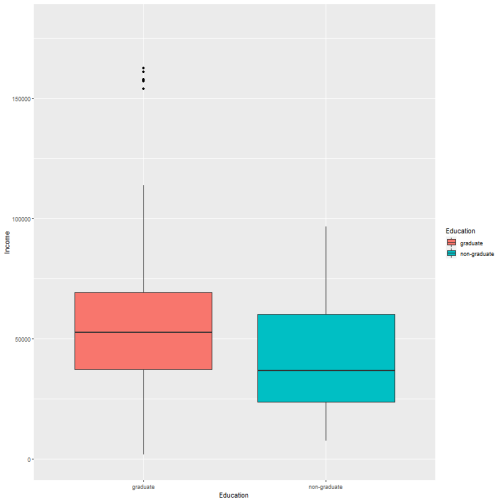


Figura 8: BoxPlot di Income ed Education.

3.4.1 Total Children

La maggior parte delle istanze del dataset ha 1 figlio, la variabile Total_Children è la somma tra la variabile KidHome e TeenHome. Questa informazione si è ricavata eseguendo il codice riportato di seguito.

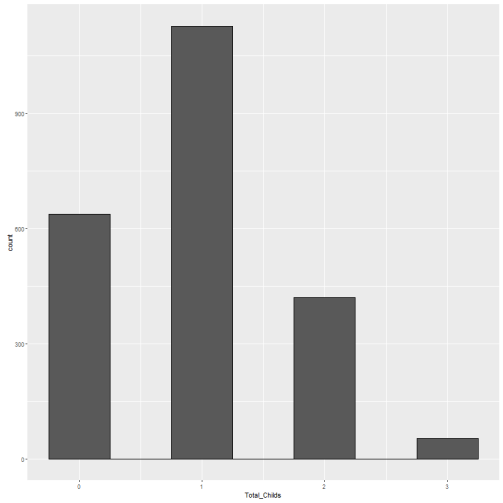


Figura 9: Istogramma di Total_children

3.4.1.1 Total_Childen e Age

Si è analizzata anche la relazione tra il numero totale di figli, *Total_children* ed *Age* ed è risultato che tra gli *Adult* il numero di figli più frequente è 1 e che rispetto ai *Senior* hanno più figli.

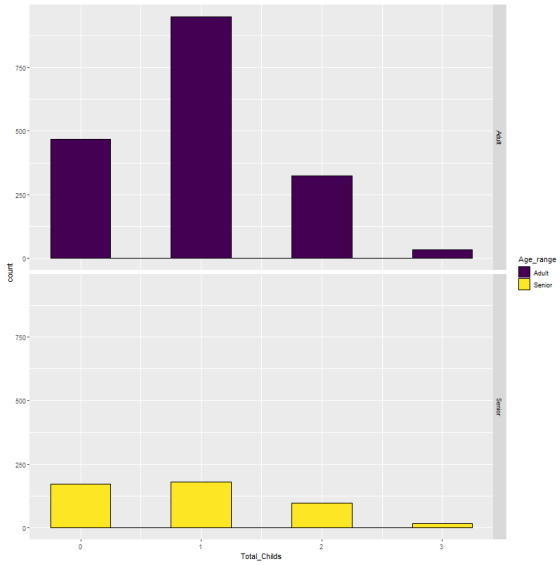


Figura 10: Istogramma di Total_Children e Age

3.4.1.2 Total_Children e Marital_Status

Nel pre processing i valori che può assumere la variabile *Marital_Status* sono stati collassati in due possibili valori. *Marital_Status* è single se la variabile *Marital_Status* è *single*, *widow* oppure *divorced*. E' uguale a 1 se il valore assunto da *Marital_Status* è *couple* oppure *together*. Rappresentando il diagramma a barre considerando *Total_Childrene Marital_Status* si ricava che la maggior parte delle istanze ha un figlio. Il numero di istanze con *Marital_Status* pari a 0 è minore rispetto a quelle con valore uguale a 1 per i casi di zero e due figli, mentre per il caso di tre figli sono molto simili.

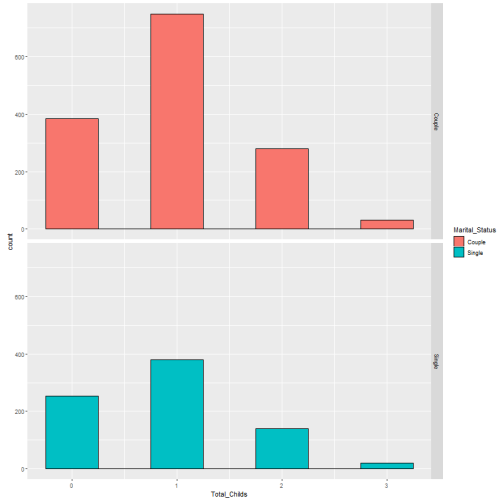


Figura 11: Istogramma di Total_Children e Marital_Status

Il grafico prendendo in considerazione la variabile *Education* è simile.

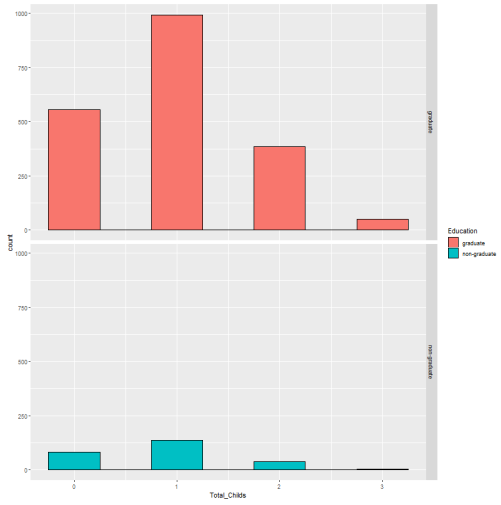


Figura 12: Istogramma di Total_Children e Education

3.4.1.3 Total_Children e Income

Sia dall' hist plot che dal jitte rplot si nota che lee istanze che hanno uno stipendio basso hanno più figli rispetto a chi ha uno stipendio alto. Nel caso di due figli ci sono più casi di persone con stipendio *low medium* rispetto a chi ha uno stipendio *low*. Nel caso di istanze con stipendio *high* è comune che si abbiano figli.

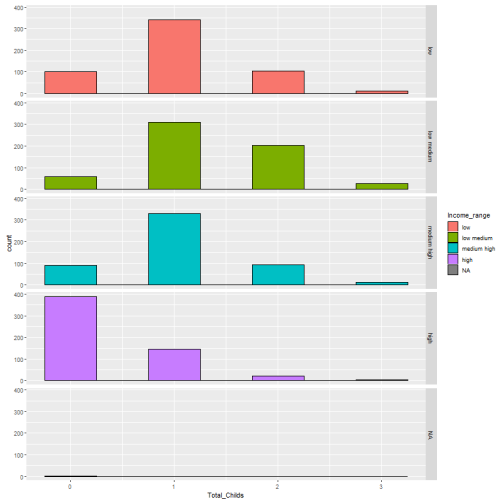


Figura 13: Istogramma Income e Total_Children

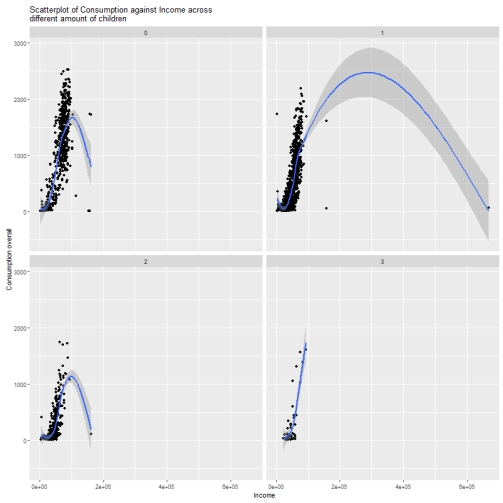


Figura 14: ScatterPlot Income e Total_Children

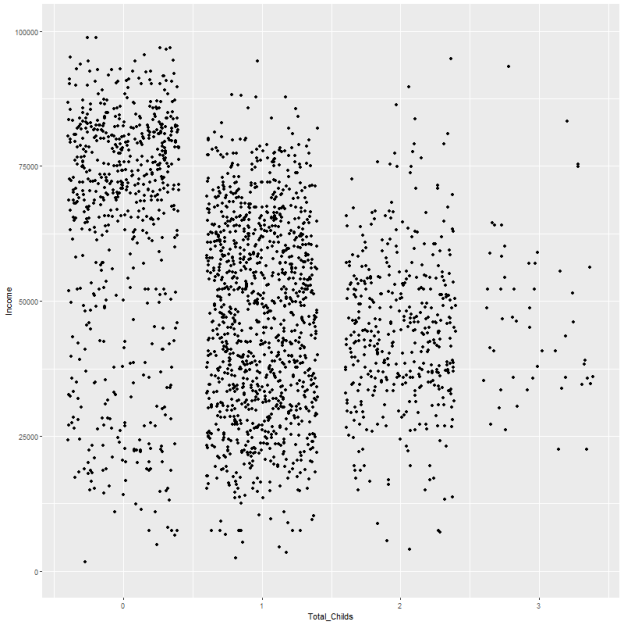


Figura 15: JitterPlot di Total_Children e Income

3.4.2 Total Spent

La maggior parte degli individui spende meno di 500$. Facendo il bar-plot della variabile *Total_Spent* si nota che le istanze presenti nel dataset spendono più per il vino e per la carne.
Per poter produrre il grafico della Figura 16 sono stati sommati i totali che ogni istanza ha speso per un certo prodotto.

```
# Plot histogram of total_spent
ggplot(dataSet, aes(x=Total_spent)) + geom_histogram(binwidth = 15, colour = "Black")

# Create dataAcceptedCmp
dataTotalSpent <- data.frame(
  name = c("MntWines", "MntFruits", "MntMeatProducts", "MntFishProducts",
      "MntSweetProducts", "MntGoldProds"),
  value = c(sum(dataSet$MntWines), sum(dataSet$MntFruits), sum(dataSet$MntMeatProducts),
            sum(dataSet$MntFishProducts), sum(dataSet$MntSweetProducts),
                sum(dataSet$MntGoldProds)))

# Barplot of total_spent for each type
ggplot(dataTotalSpent, aes(x=name, y=value)) +
  geom_bar(stat = "identity", color = "Black") + xlab("Total Spent for each type")
```

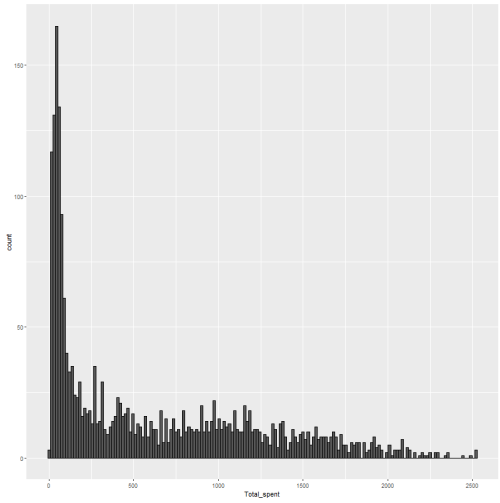


Figura 16: Istogramma Total_Spent.

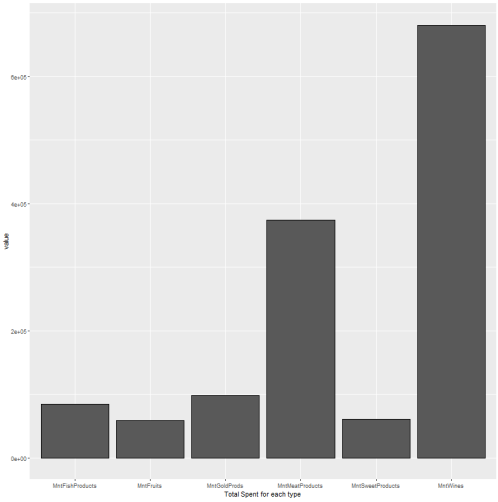


Figura 17: Istogramma di Total_Spent distinguendo i tipi di prodotto.

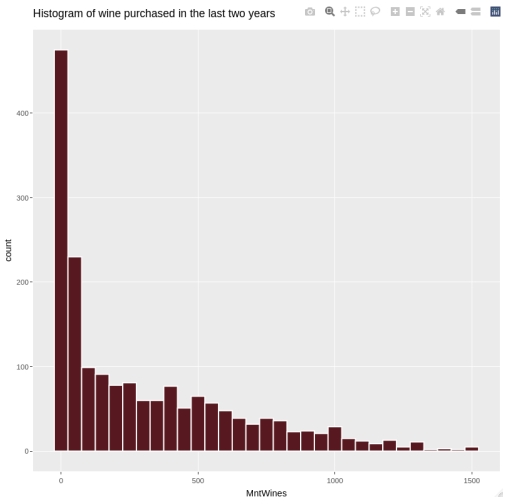


Figura 18: Istogramma Wine

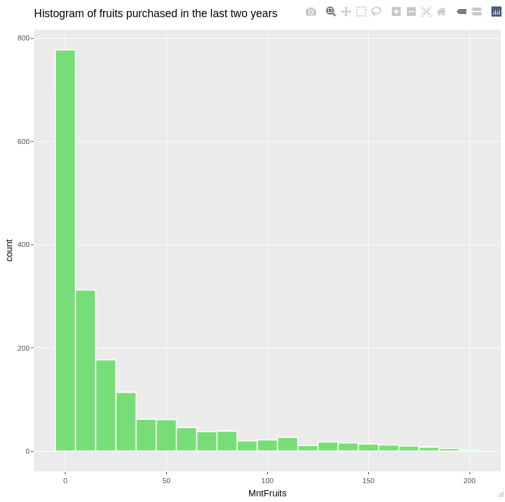


Figura 19: Istogramma Fruit

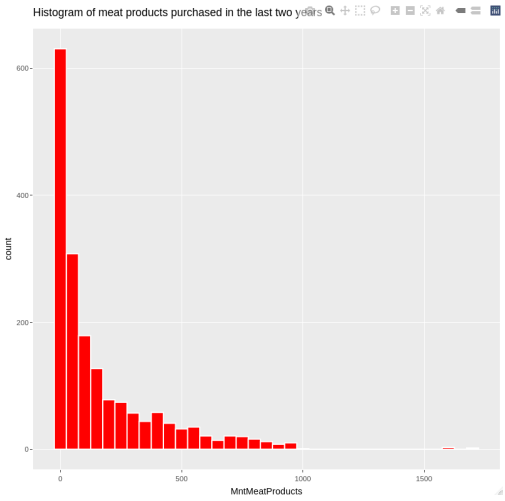


Figura 20: Istogramma Meat

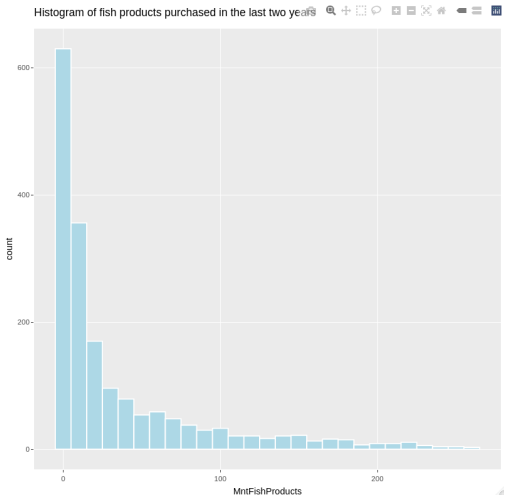


Figura 21: Istogramma Fish

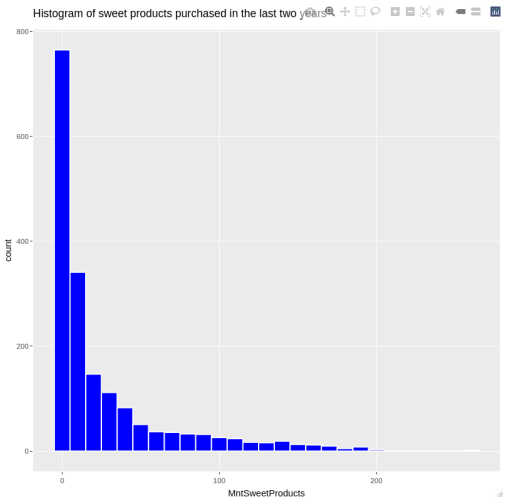


Figura 22: Istogramma Sweet

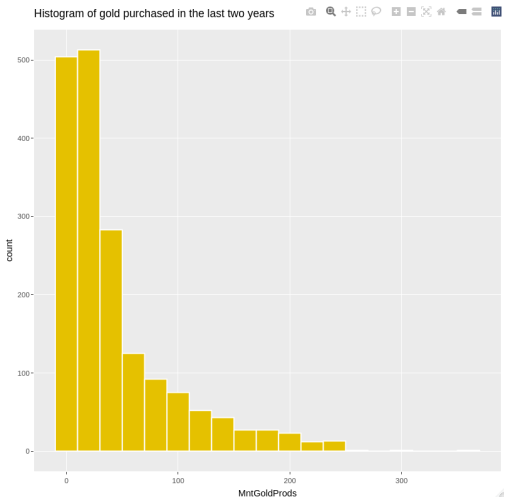


Figura 23: Istogramma Gold

3.4.2.1 Total Spent e Age

Osservando la variabile *Total_Spent* considerando *Age* si ha che la maggior parte degli individui spende meno di 500$.

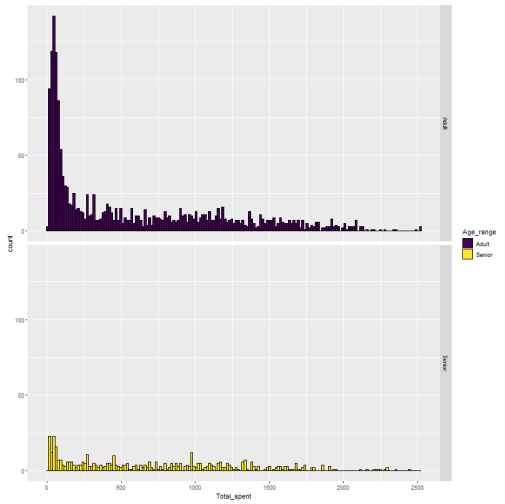


Figura 24: Istogramma di Total_Spent e Age

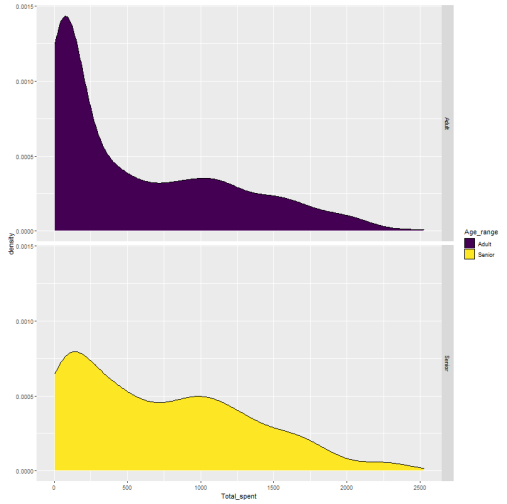


Figura 25: Diagramma di Densità di Total_Spent e Age

3.4.2.2 Total Spent e Marital_Status

In questo caso sembrano simili. la maggior parte delle coppie spende un totale inferiore a 500$. La situazione in single è un po' più rilassata che può essere dovuto al fatto che la maggior parte degli individui nel dataset sono coppie.

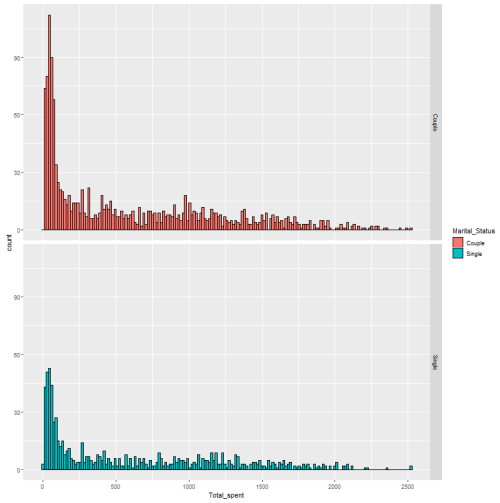


Figura 26: Istogramma di Total_Spent e Marital_Status

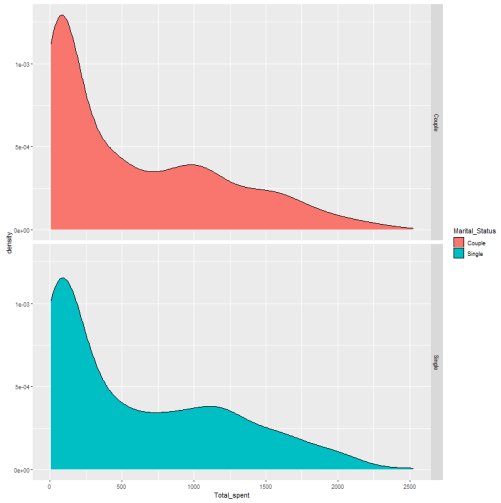


Figura 27: Diagramma di Densità di Total_Spent e Marital_Status

3.4.2.3 Total Spent ed Education

I laureati in genere spendono più dei non laureati. la maggior parte dei non laureati spende da 0 a 1500. da 1500 ci sono più casi di laureati che non laureati. Il grafico della densità è molto simile a quello della variabile *Marital_Status*.

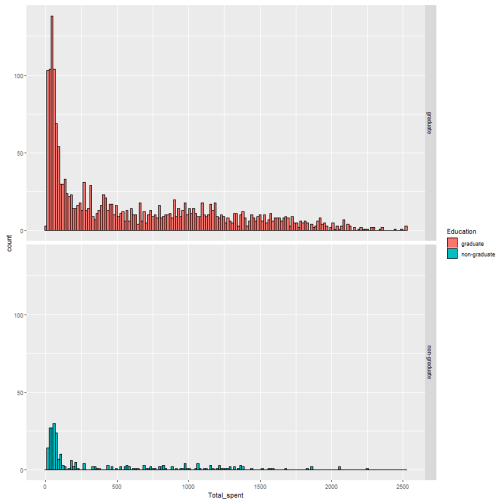


Figura 28: Istogramma di Total_Spent e Education

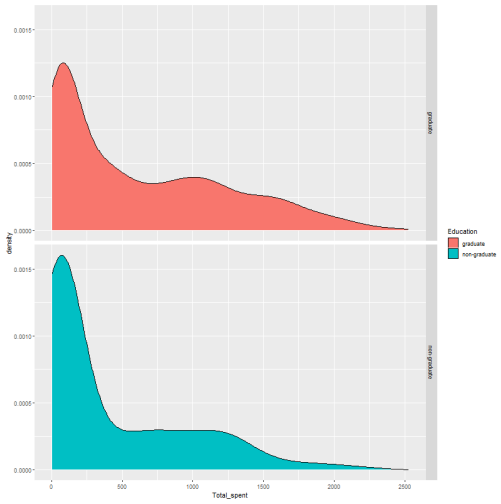


Figura 29: Diagramma di Densità di Total_Spent e Education

3.4.2.4 Total Spent e Total_Children

Si è studiata la variabile *Total_Spent* anche per *Total_Children* e si nota che le istanze che non hanno figli spendono somme maggiori rispetto a quelle in cui il numero di figli è superiore a 0. Questo fatto è più evidente osservando il diagramma di densità rappresentato nella Figura 31

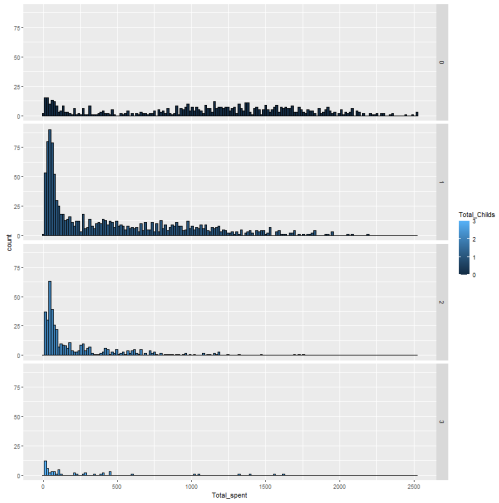


Figura 30: Istogramma di Total_Spent e Total_Children

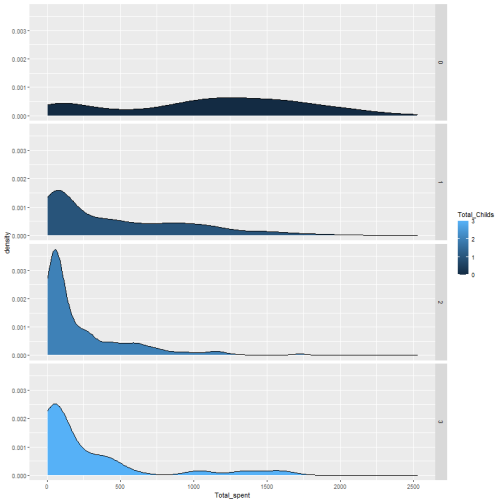


Figura 31: Diagramma di Densità di Total_Spent e Total_Children

3.4.2.5 Total_Spent e Income

Inoltre analizzando l'istogramma della variabile *Total_Spent* in relazione ad *Income* che sono direttamente proporzionali infatti nel caso di un *Income* pari a *low* il *Total_Spent* è basso. Per completezza sono stati prodotti anche gli *ScatterPlot* della variabile *Income* e la somma spesa per ogni tipologia di prodotto. Questi grafici sono molto simili tranne quello nella Figura 36.

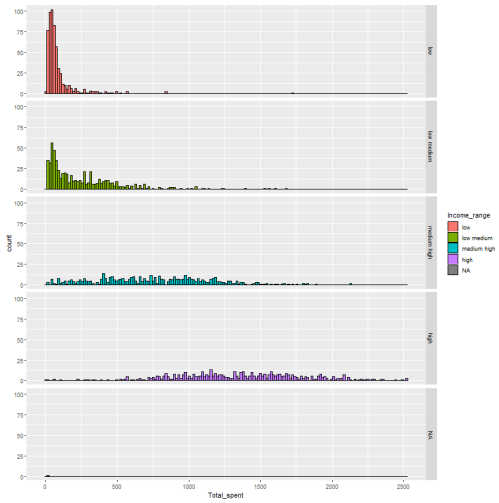


Figura 32: Istogramma di Total_Spent e Income

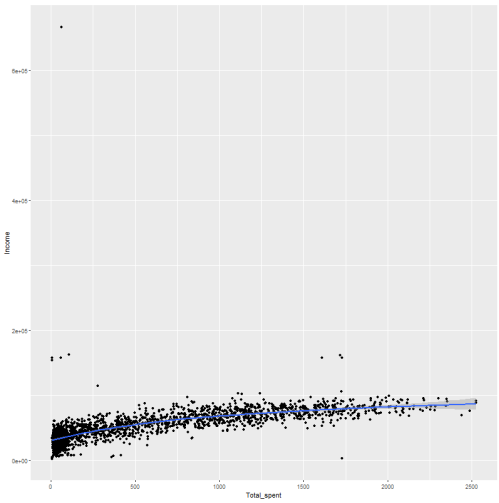


Figura 33: Diagramma di Densità di Total_Spent e Income

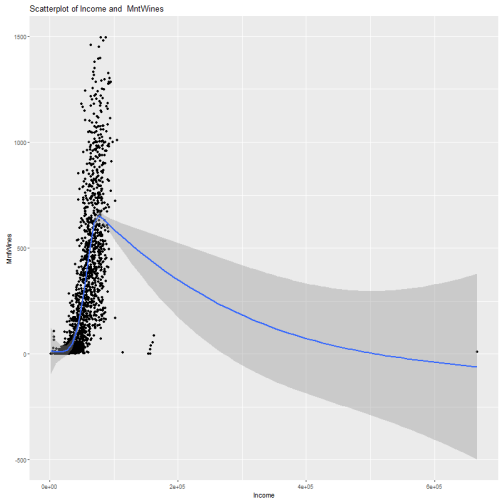


Figura 34: ScatterPlot Income e MntWines

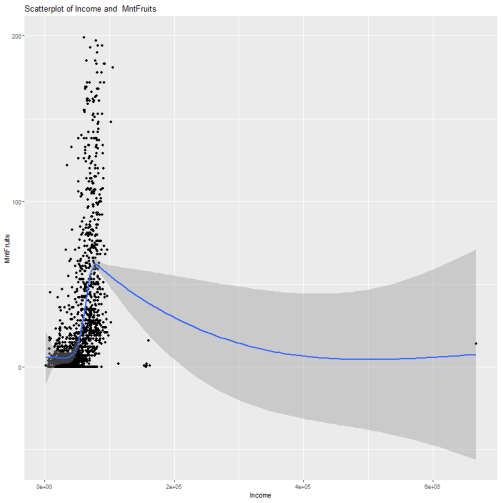


Figura 35: ScatterPlot Income e MntFruits

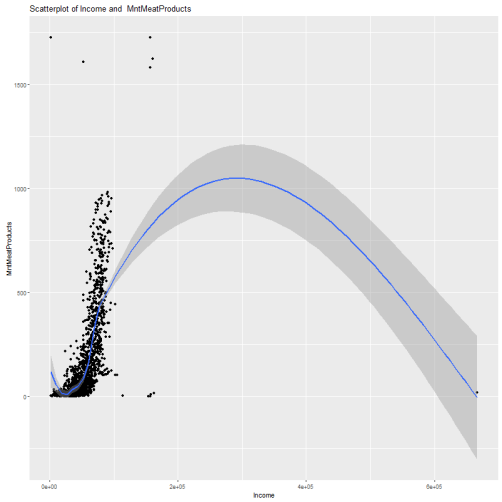


Figura 36: ScatterPlot Income e MntMeatProducts

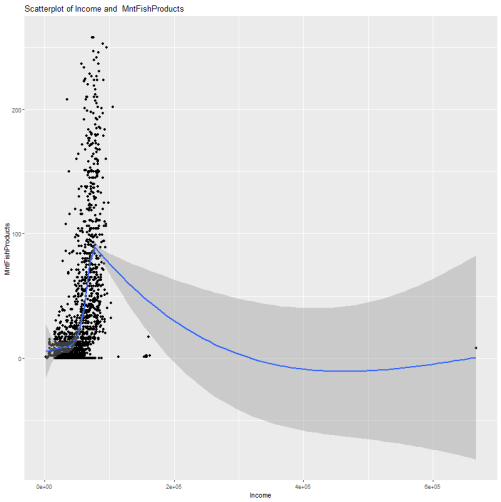
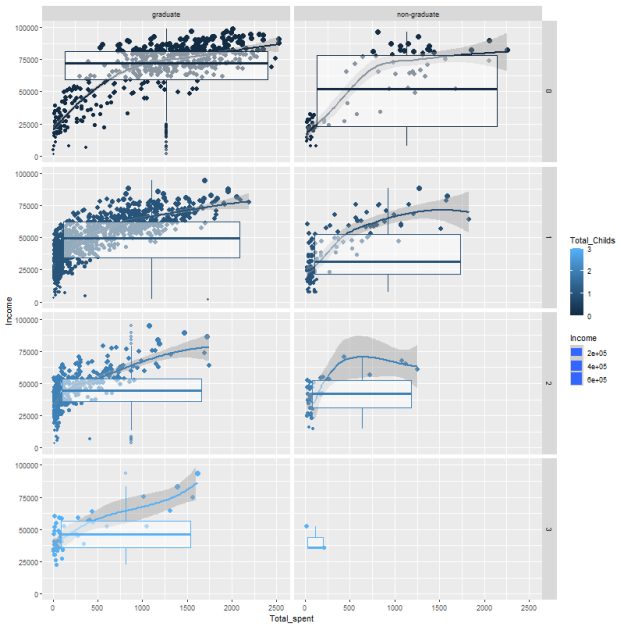
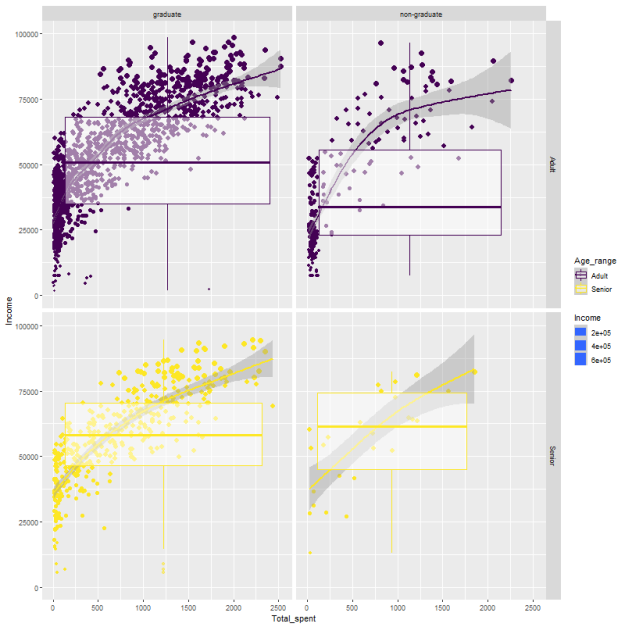


Figura 37: ScatterPlot Income e MntFishProducts

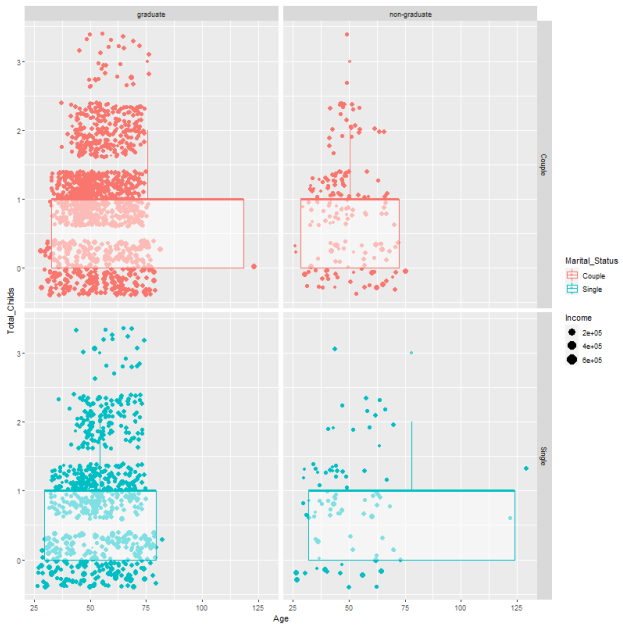
3.4.2.6 Total_Spent JitterPlot+BoxPlot



(a) JitterPlot+BoxPlot di Total_Spent, Income e Marital_Status.



(b) JitterPlot+BoxPlot di Total_Spent, Income e Total_Children.



(c) JitterPlot+BoxPlot di Total_Spent, Income ed Age.

3.4.3 Campaign Analysis

Si è eseguita un'analisi anche sulle campagne accettate da ogni individuo. Per ogni istanza del dataset si sa se ha accettato o meno una campagna. In questo caso le campagne considerate sono cinque e come è possibile osservare dai barplot più di mille istanze non hanno accettato alcuna campagna, mentre chi ha accettato una o più campagne ha scelto la campagna 4.

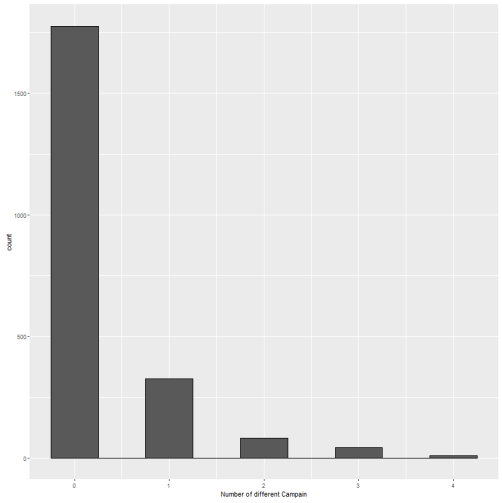


Figura 38: Istogramma del numero totale di campagne accettate da un'istanza. Total_Campaign

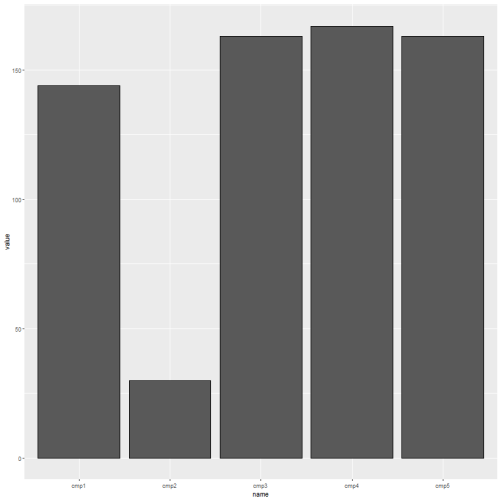


Figura 39: Istogramma del numero di istanze che ha accettato la campagna i-esima.

3.5 PCA

La PCA è stata prevalentemente sfruttata al fine di ridurre il numero elevato di variabili che descrivono l'insieme di dati a un numero minore di variabili latenti, limitando il più possibile la perdita di informazioni. Il codice seguente mostra parte del codice riportato durante l'analisi dei dati:

```
pca <- PCA(dataSet_scaled, graph = FALSE)

#Getting the variance of the first 9 new dimensions
pca$eig[,2][1:9]

#Getting the cummulative variance
pca$eig[,3][1:5]

#Getting the most correlated variables
dimdesc(pca, axes = 1:2)

# get eigenvalue
get_eigenvalue(pca)

# visualize pca
fviz_eig(pca, addlabels = TRUE, ylim = c(0, 50))
fviz_contrib(pca, choice = "var", axes = 1, top = 5)
fviz_pca_biplot(pca)

#Creating a factor map for the variable contributions
fviz_pca_var(pca, col.var = "contrib", repel = TRUE)

fviz_pca_var(pca, select.var = list(contrib = 5), col.var = "contrib", repel = TRUE)

# Extract the principal components
pcaTraining <- PCA(trainingSet_scaled, graph = FALSE)
trainingSet_input <- data.frame(get_pca_ind(pcaTraining)$coord)
```

Da esso si vuole dare particolare attenzione alla funzione *get_eigenvalue(pca)* che fornire le informazioni rappresentate nella tabella 10. Si vuole anche fornire un riferimento grafico a quest'ultima tramite l'output della funzione *fviz_eig(pca, addlabels = TRUE, ylim = c(0, 50))* descritto dalla figura 40. Da essa si può notare che le prime 5 dimensioni fornite dalla PCA forniscono il 70% della varianza cumulativa, per questo motivo si è deciso di prendere in considerazione tali dimensioni. Inoltre si vuole sottolineare l'importanza della prima dimensione che riesce a spiegare più del 40% della varianza dei dati.

	eigenvalue	variance.percent	cumulative.variance.percent
Dim.1	7.00	41.15	41.15
Dim.2	1.75	10.31	51.46
Dim.3	1.14	6.69	58.15
Dim.4	1.08	6.34	64.49
Dim.5	1.00	5.86	70.34
Dim.6	0.77	4.54	74.88
Dim.7	0.66	3.86	78.74
Dim.8	0.62	3.64	82.38
Dim.9	0.58	3.38	85.76
Dim.10	0.46	2.73	88.49
Dim.11	0.43	2.52	91.00
Dim.12	0.39	2.31	93.31
Dim.13	0.35	2.04	95.35
Dim.14	0.31	1.83	97.18
Dim.15	0.26	1.52	98.70
Dim.16	0.22	1.30	100.00
Dim.17	0.00	0.00	100.00

Tabella 10: Output funzione *get_eigenvalue(pca)*

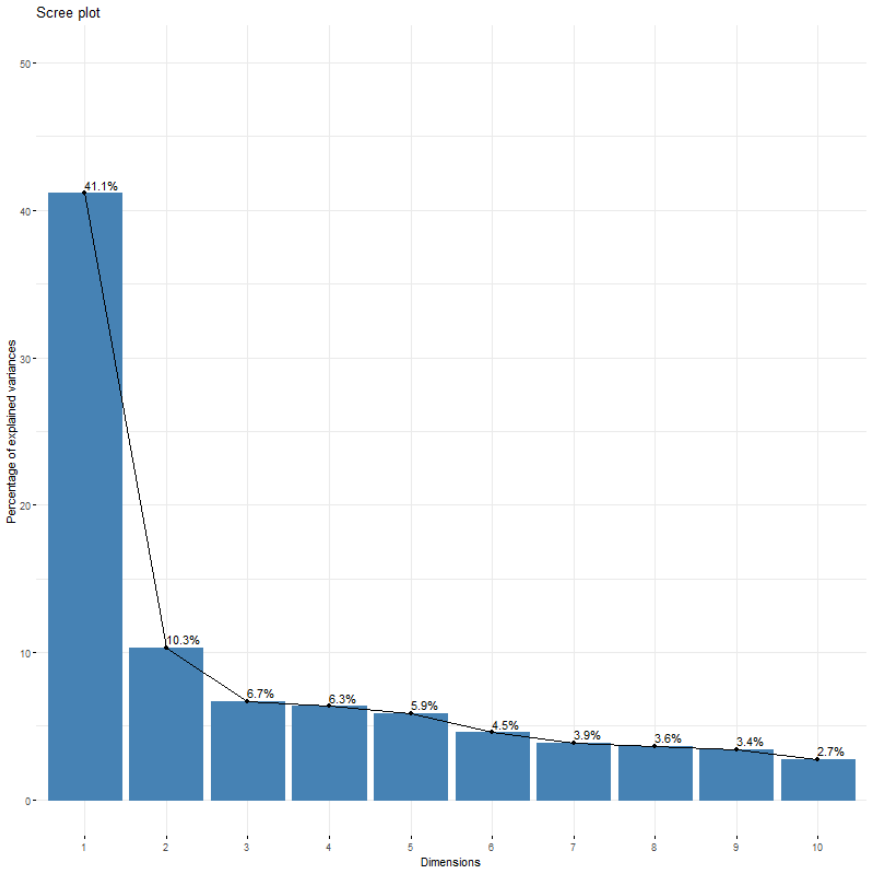


Figura 40: Output funzione *fviz_eig(pca, addlabels = TRUE, ylim = c(0, 50))*

In particolare la tabella 11 vuole far notare il contributo di ciascun attributo del dataset nella creazione delle dimensioni della pca. Da essa possiamo notare le cinque principali variabili che hanno contribuito maggiormente nella creazione della prima dimensione della *principal component analysis*: Total_spent, MntMeatProducts, NumCatalogPurchases, MntWines e Income. La figura 41 ne mostra un grafico più esplicativo.

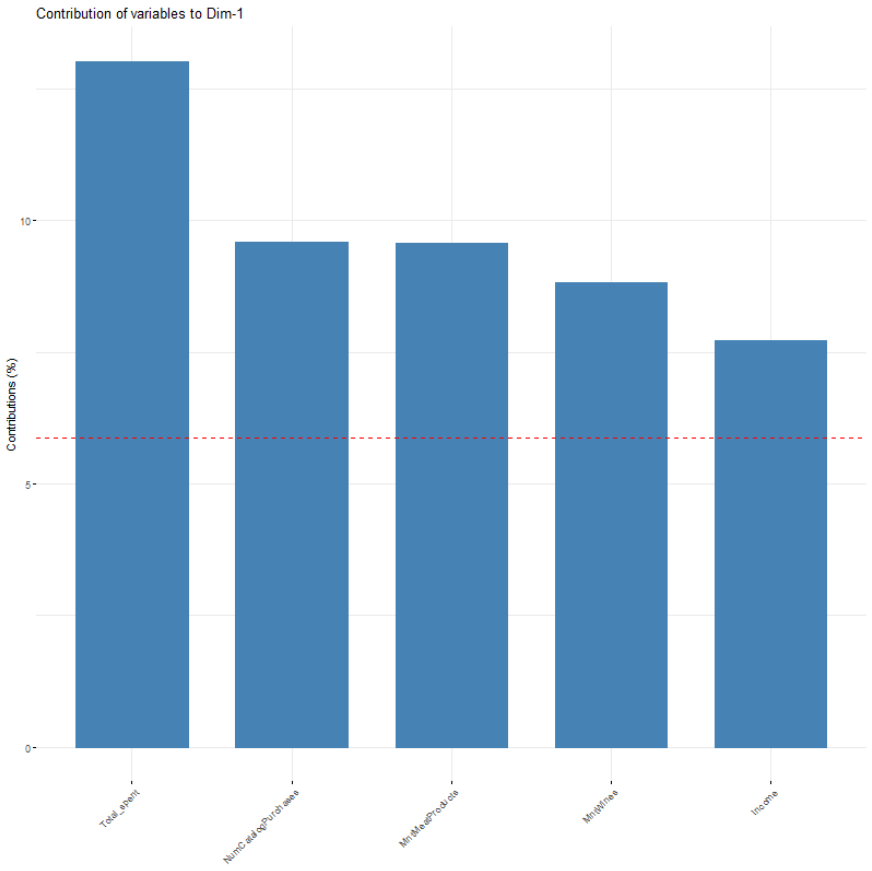


Figura 41: Output funzione *fviz_contrib(pca, choice = "var", axes = 1, top = 5)*

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
Income	7.73	0.16	1.54	4.61	0.91
Recency	0.00	0.00	0.25	10.43	88.32
MntWines	8.82	5.02	11.32	0.32	0.19
MntFruits	6.94	1.20	11.53	0.33	0.06
MntMeatProducts	9.58	1.08	0.06	0.00	0.13
MntFishProducts	7.46	1.42	9.96	0.10	0.04
MntSweetProducts	6.89	0.75	9.12	0.32	0.05
MntGoldProds	4.66	2.42	5.33	2.54	0.40
NumDealsPurchases	0.26	36.20	4.32	0.71	0.11
NumWebPurchases	4.10	18.79	0.53	2.07	0.00
NumCatalogPurchases	9.60	0.11	0.43	0.25	0.05
NumStorePurchases	7.55	3.44	0.43	0.29	0.31
NumWebVisitsMonth	5.66	9.62	0.01	12.58	1.15
Total_spent	13.03	0.57	1.12	0.30	0.17
Total_Campains	2.71	0.06	42.35	6.21	1.26
Total_Childs	4.78	13.94	0.53	2.94	0.05
Age	0.22	5.20	1.15	55.99	6.81

Tabella 11: Output *pcavarcontrib*

Le funzioni *fviz_pca_var* hanno permesso di analizzare graficamente le dimensioni che spiegano il contributo delle variabili considerate nelle prime due dimensioni (fig. 42 e fig. 43).

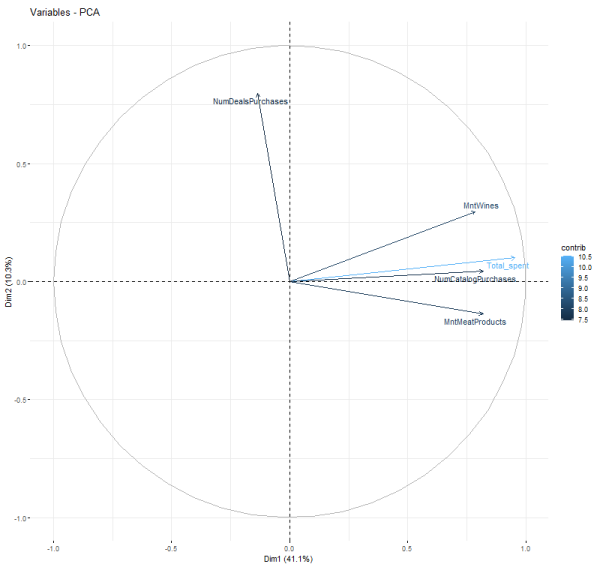


Figura 42: Contributo di tutte la variabili sulle prime due dimensioni.

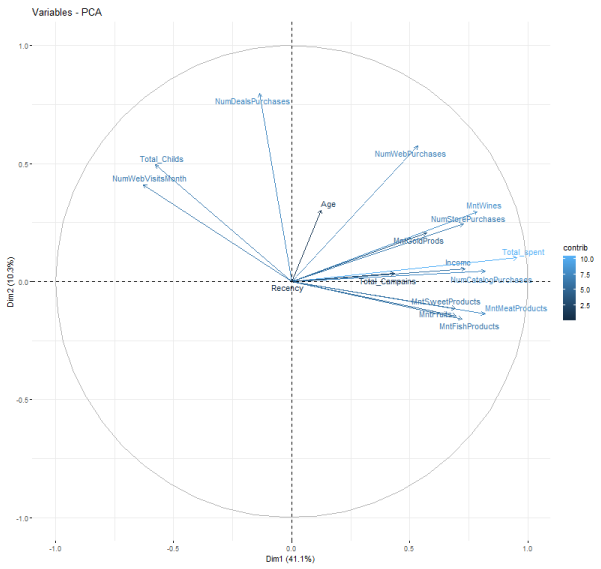


Figura 43: Contributo delle cinque variabili più significative sulle prime due dimensioni.

In fine si è preso in considerazione un nuovo dataset, nato dalle dimensioni fornite dalla PCA, descritto dalla tabella 12. Esso verrà sfruttato durante l'analisi del dataset tramite i alcuni degli algoritmi utilizzati.

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
1	4.15	0.41	1.40	0.12	0.27
2	-2.56	-0.14	-0.62	1.48	-0.78
3	1.81	-0.22	0.42	0.13	-1.15
4	-2.46	-0.91	0.29	-0.99	-0.58
5	-0.24	0.50	1.19	0.05	1.42
6	0.65	0.73	-0.16	-0.19	-1.22

Tabella 12: *Head* del Dataset fornito dalla PCA

4 Modelli utilizzati

Il dataset non presenta una variabile target specifica, per questo motivo di è ritenuto opportuno analizzare il tutto mediante algoritmi non supervisionati come quelli di **clustering**. In particolare si è deciso di utilizzare principalmente l'algoritmo **K-Means**. Oltre a ciò si è anche cercato di trovare un target su cui poter fare predizioni mediante alberi decisionali.

4.1 K-Means

La natura stessa dei dati ha comportato l'obbligo di analizzare il tutto mediante un algoritmo come **K-Means**.

4.1.1 Silhouette e Elbow Method

Si è ritenuto opportuno utilizzare metodi come **Elbow Method** e **Silhouette** al fine di determinare il numero ottimale di **clusters** da utilizzare.

```
set.seed(6)
wcss <- vector()
for (i in 1:10) {
  wcss[i] <- sum(kmeans(dataSet_scaled, i)$withinss)
}
plot(1:10, wcss, type="b", main = paste('Clusters'), xlab='Number of clusters',
    ylab="WCSS")

# OR

fviz_nbclust(dataSet_scaled,kmeans,method="wss")+geom_vline(xintercept=2,linetype=2)
```

Il codice mostra una prima analisi mediante *elbow method*, il cui output si può notare dalle figure 44 e 45. Esse mostrano un'inclinazione tra un numero di clusters compreso tra 2 e 3.

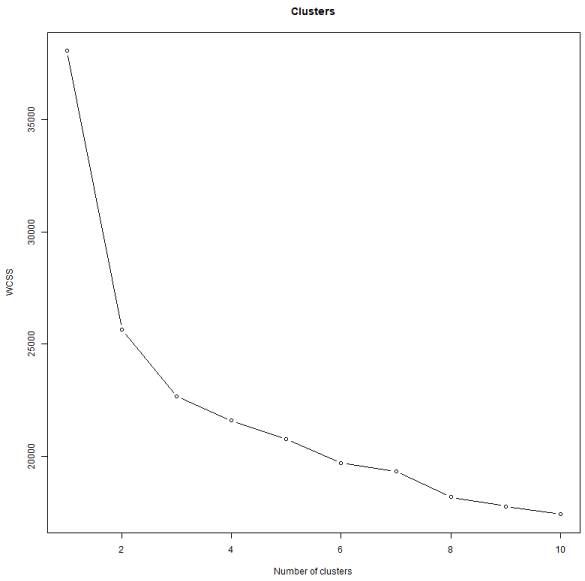


Figura 44: Elbow Method effettuato manualmente

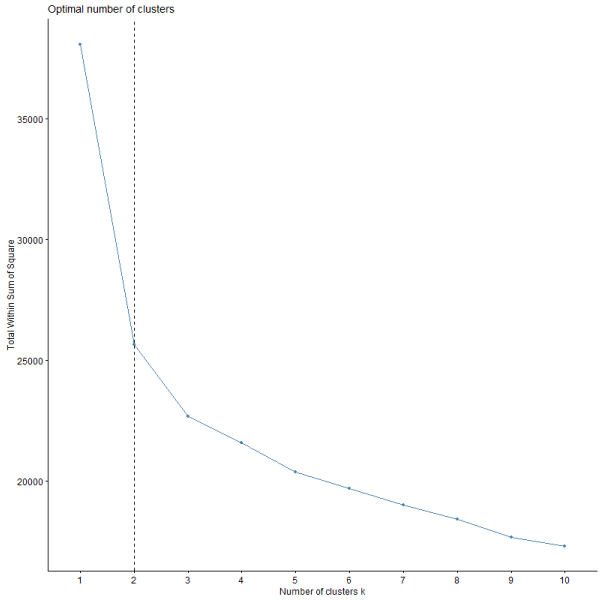


Figura 45: Elbow Method effettuato automaticamente dal metodo *fviz_nbclust*

Per maggior sicurezza si è quindi deciso di sfruttare anche il metodo *Silhouette* mediante il codice sottostante.

```
k <- 2:10
avg_sil <- sapply(k, silhouette_score)
plot(k, type='b', avg_sil, xlab='Number of clusters', ylab='Average Silhouette Scores',
    frame=FALSE)
avg_sil # <<<- important

# OR
fviz_nbclust(dataSet_scaled, kmeans, method="silhouette")
```

Il codice presenta in output i grafici espressi della figura 46, da cui si è potuto constatare con maggior sicurezza l'utilizzo di un numero di clusters pari a 2.

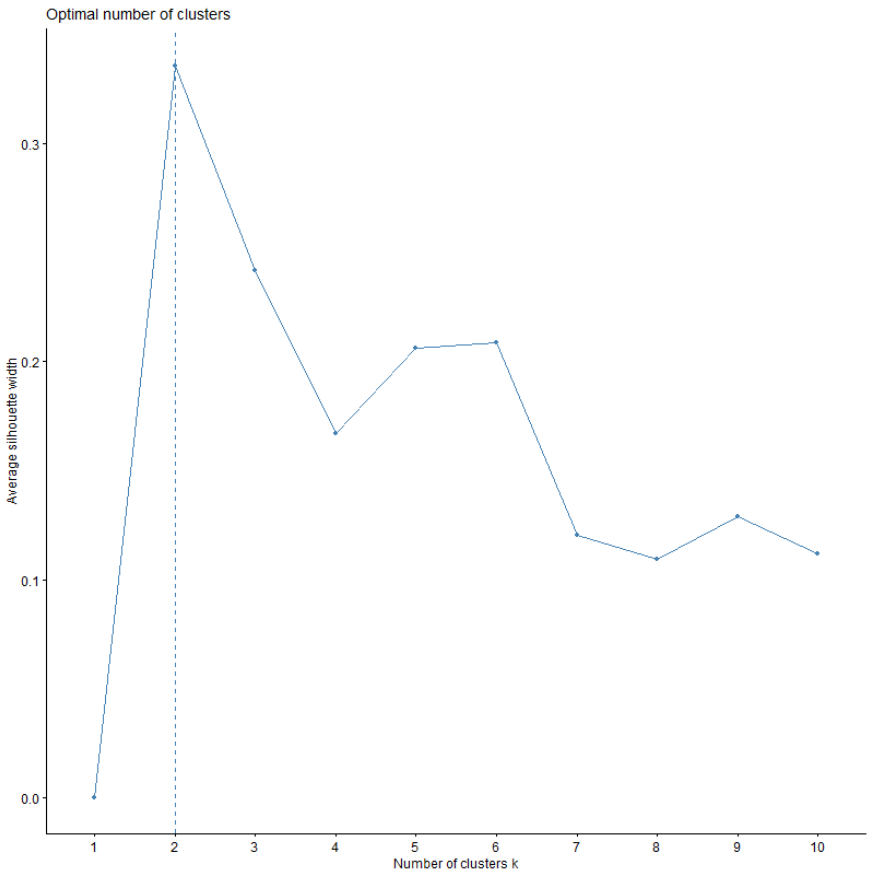


Figura 46: Silhouette effettuata automaticamente dal metodo *fviz_nbclust*

4.1.2 Algoritmo e Analisi

Il codice sottostante viene espresso mediante le figure 47 e 48 che rappresentano rispettivamente il **partizionamento per ogni cluster** e la **Matrice di dissimilarità**. E' doveroso notare che entrambe le figure mostrano un buon partizionamento degli elementi del cluster, sebbene ci sia qualche elemento di intersezione tra i due.

```
set.seed(29)
km <- kmeans(dataSet_scaled, 2, nstart = 10)
print(km$centers)
fviz_cluster(km, dataSet_scaled, geom = "point",ellipse.type = "norm",repel = TRUE)
cl <- km$cluster
dissplot(dist(dataSet_scaled), labels=cl,options=list(main="Kmeans Clustering With k=2"))
```

Figura 47: Partizionamento in clusters dei dati

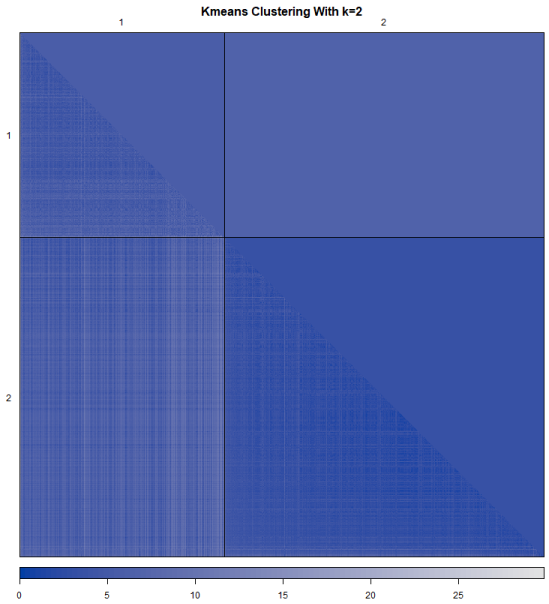


Figura 48: Dissimilarity matrix

In particolare la figura 47 presenta gli elementi all'interno di ogni cluster. La tabella 13 mostra il numero preciso di elementi per ogni cluster.

	cluster	n
1	1	874
2	2	1366

Tabella 13: Numero di elementi per ogni cluster

Si è ritenuto necessario analizzare i risultati ottenuti mediante le variabili più significative, sorte durante la *Principal Component Analisys*.

```
wines <- ggplot(dataSet, aes(MntWines)) +
  facet_grid(cluster~.)

wines + geom_histogram(color = "black", fill = "red")
wines + geom_density(fill="red", position = "Stack")
ggplot(dataSet,
    aes(x=cluster,y=MntWines,fill=cluster))+geom_boxplot(outlier.colour="black")
```

Il codice soprastante è descritto dalle figure 51, 50 e 49. Dall'analisi dei grafici si nota una spesa maggiore di vini per i *customers* all'interno del primo cluster. In particolare la maggior parte dei clienti all'interno del secondo cluster non ha acquistato vini negli ultimi due anni.

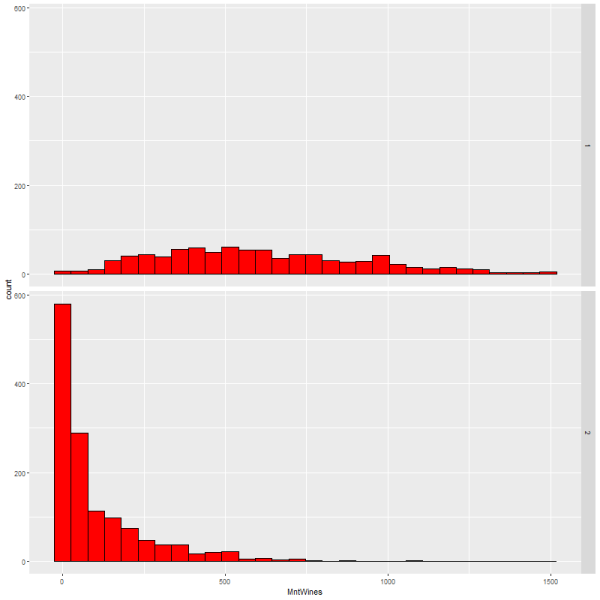


Figura 49: Istogramma della variabile Wines in relazione al numero di cluster

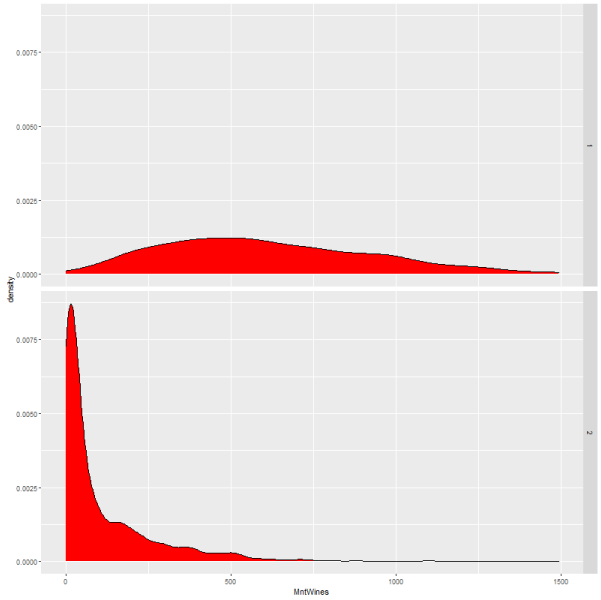


Figura 50: Diagramma di densità della variabile Wines in relazione al numero di cluster

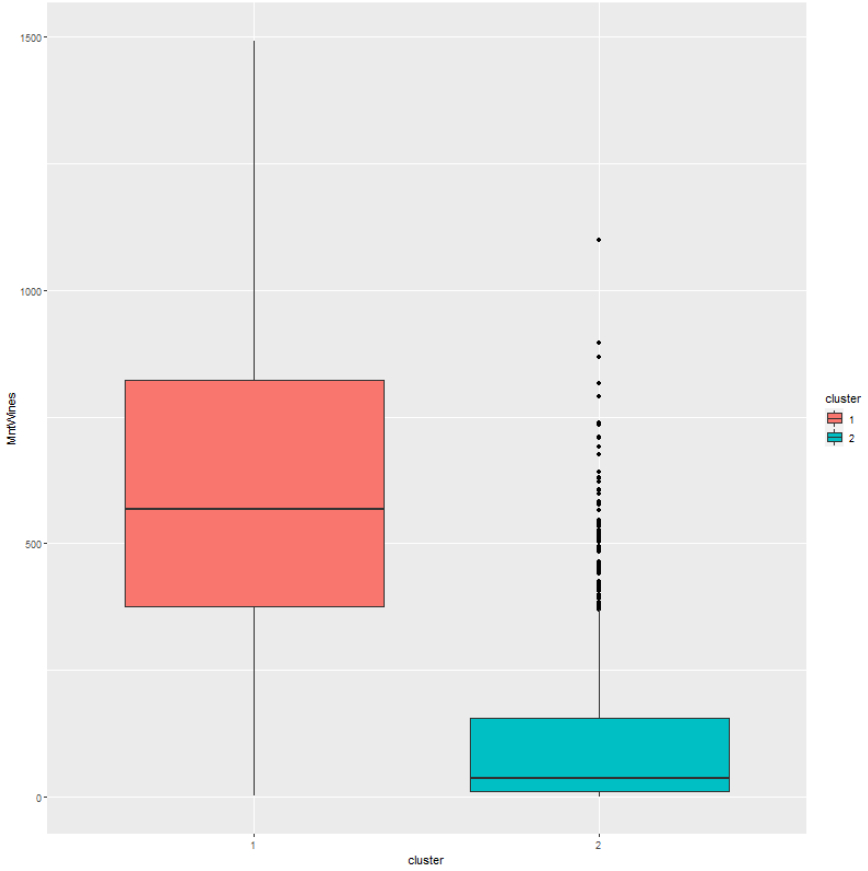


Figura 51: BoxPlot della variabile Wines in relazione al numero di cluster

```
income <- ggplot(dataSet, aes(Income))+
  facet_grid(cluster~.) +
  xlim(0,200000)

income + geom_histogram(color = "black", fill = "green")
income + geom_density(fill="green", position = "Stack")
ggplot(dataSet,
    aes(x=cluster,y=Income,fill=cluster))+geom_boxplot(outlier.colour="black") +
    ylim(0,200000)
```

Il codice appena descritto ha il compito di analizzare gli elementi dei cluster in relazione al reddito di ciascun utente, ovvero in relazione alla variabile *income*. Le figure 53, 52 e 54 ne mostrano la rappresentazione dei grafici. Da esse è doverso notare che nel secondo cluster la maggior parte dei clienti possiede un reddito generalmente più basso rispetto ai *customers* facenti parte del primo. Calcolando il reddito medio dei compratori, pari a 52247 dollari, si può anche notare che la maggior parte degli elementi nel secondo cluster possiedono un il reddito inferiore alla media. E' opportuno notare anche che, la maggior parte degli utenti del primo raggruppamento, contrariamente ai primi, possiedono un reddito superiore alla media.

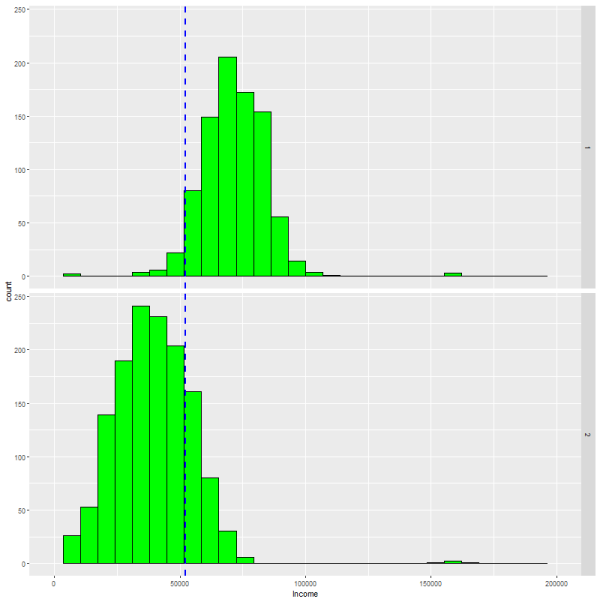


Figura 52: Istogramma della variabile Income in relazione al numero di cluster

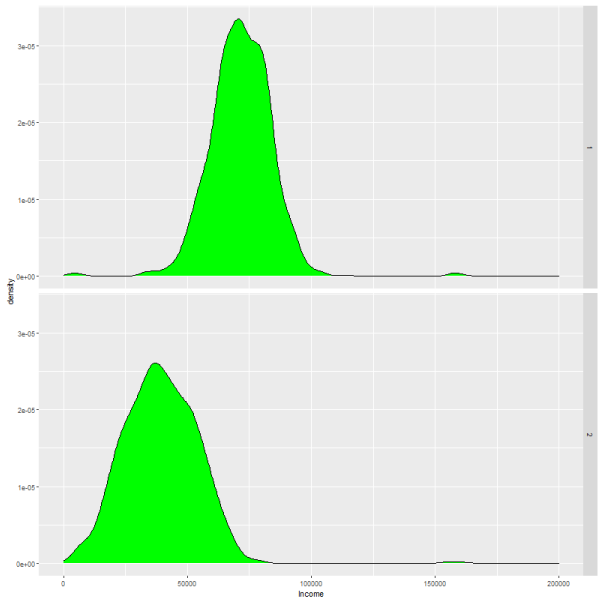


Figura 53: Diagramma di densità della variabile Income in relazione al numero di cluster

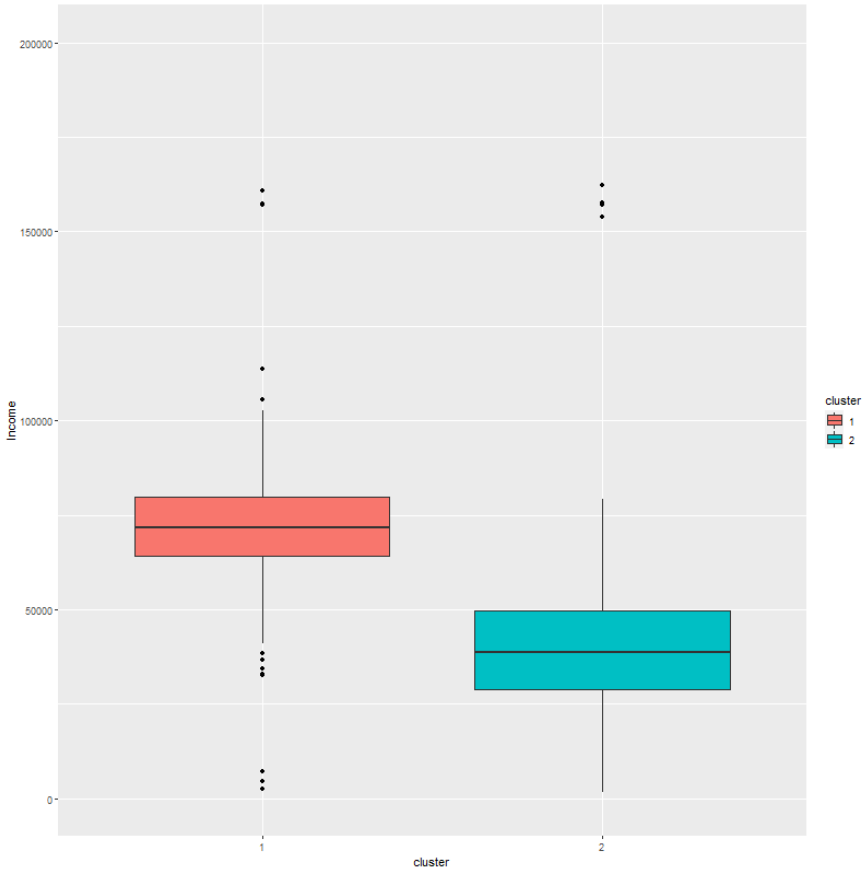


Figura 54: BoxPlot della variabile income in relazione al numero di cluster

Un'altra variabile analizzata è Total_spent, che spiega la gran parte della varianza della prima dimensione della PCA. Il codice sottostante speiga le figure 57, 56 e 55. In particolare, dall'analisi di esse è emerso che i compratori del secondo cluster generalmente spendono molto meno denaro rispetto a quelli del primo. Oltre a ciò si può anche notare che quest'ultimi versano mediamente più di mille dollari ogni due anni per prodotti come: vino, frutta, pesce e carne.

```
ts <- ggplot(dataSet, aes(Total_spent), colour=cluster) + facet_grid(cluster~.)
ts + geom_histogram(color = "black", fill = "purple")
ts + geom_density(fill="purple", position = "Stack")
ggplot(dataSet,
    aes(x=cluster,y=Total_spent,fill=cluster))+geom_boxplot(outlier.colour="black")
```

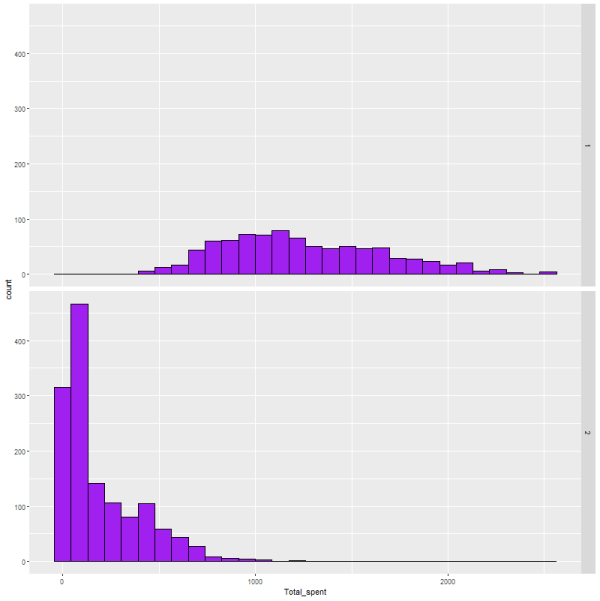


Figura 55: Istogramma della variabile Total_spent in relazione al numero di cluster

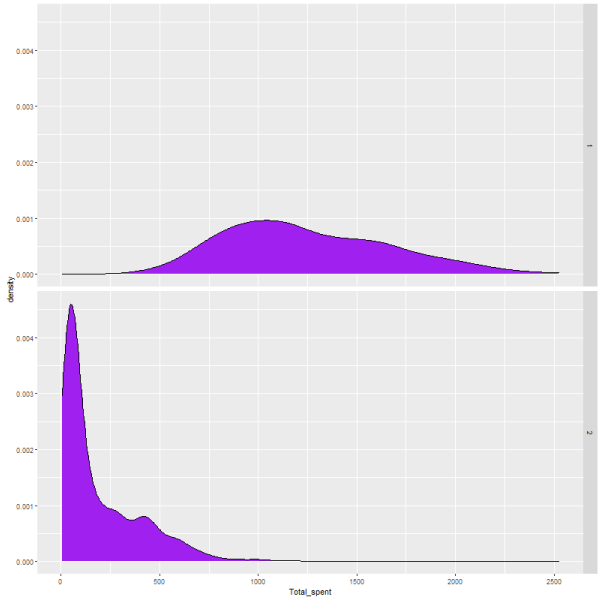


Figura 56: Diagramma di Densità della variabile Total_spent in relazione al numero di cluster

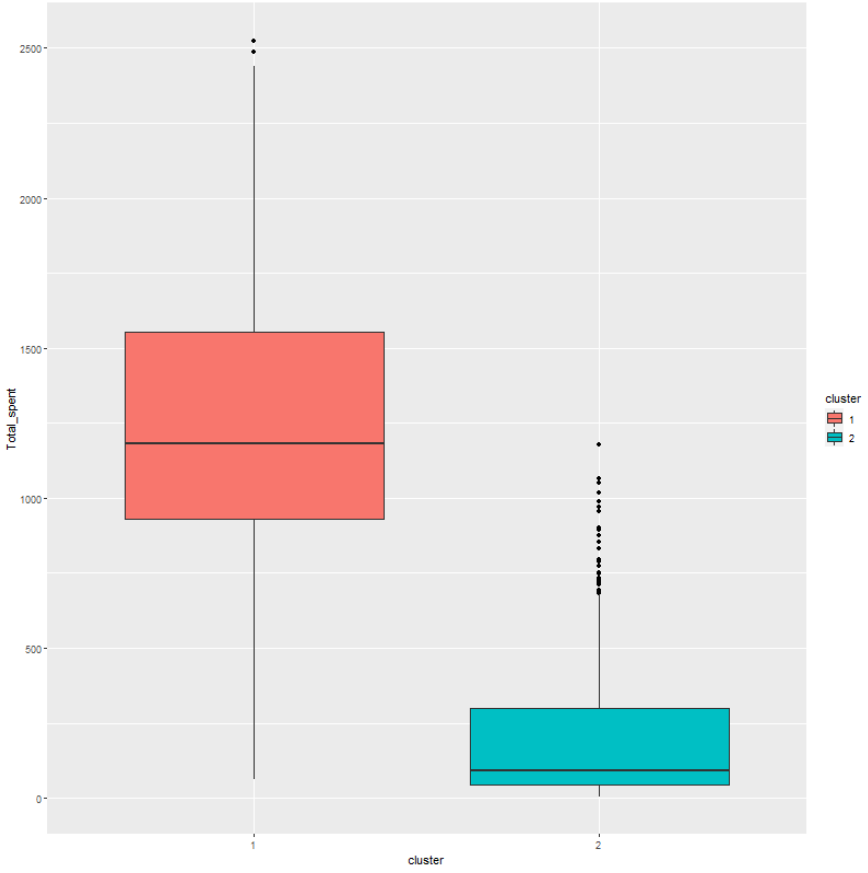


Figura 57: BoxPlot della variabile Total_spent in relazione al numero di cluster

Le figure 60, 59 e 58 mostrano che i clienti del secondo cluster effettuano compere sul catalogo generalmente in quantità minore rispetto a quelli del primo. Difatti acquistano mediamente 5 prodotti dal catalogo.

```
numCatalogPurchases <- ggplot(dataSet, aes(NumCatalogPurchases)) + facet_grid(cluster~.)
numCatalogPurchases + geom_histogram(color = "black", fill = "blue")
numCatalogPurchases + geom_density(fill="blue", position = "Stack")
ggplot(dataSet,
    aes(x=cluster,y=NumCatalogPurchases,fill=cluster))+geom_boxplot(outlier.colour="black")
    + ylim(0,10)
```

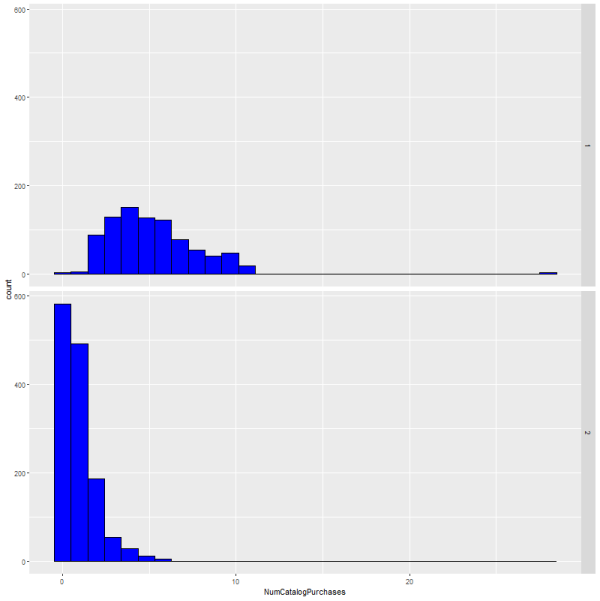


Figura 58: Istogramma della variabile NumCatalogPurchases in relazione al numero di cluster

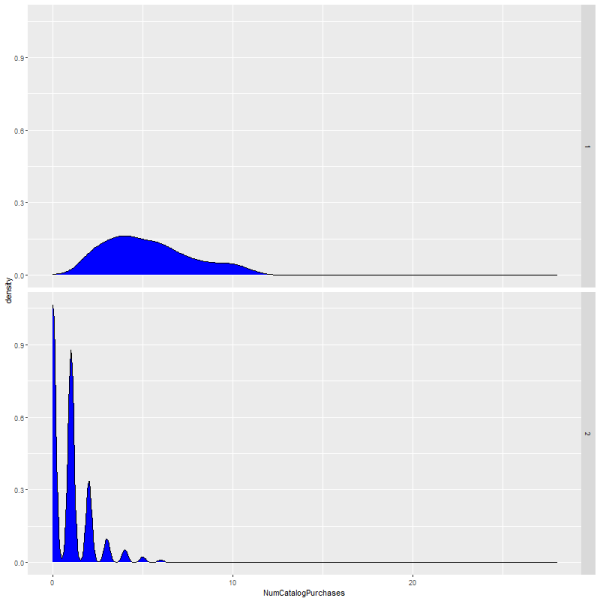


Figura 59: Diagramma di Densità della variabile NumCatalogPurchases in relazione al numero di cluster

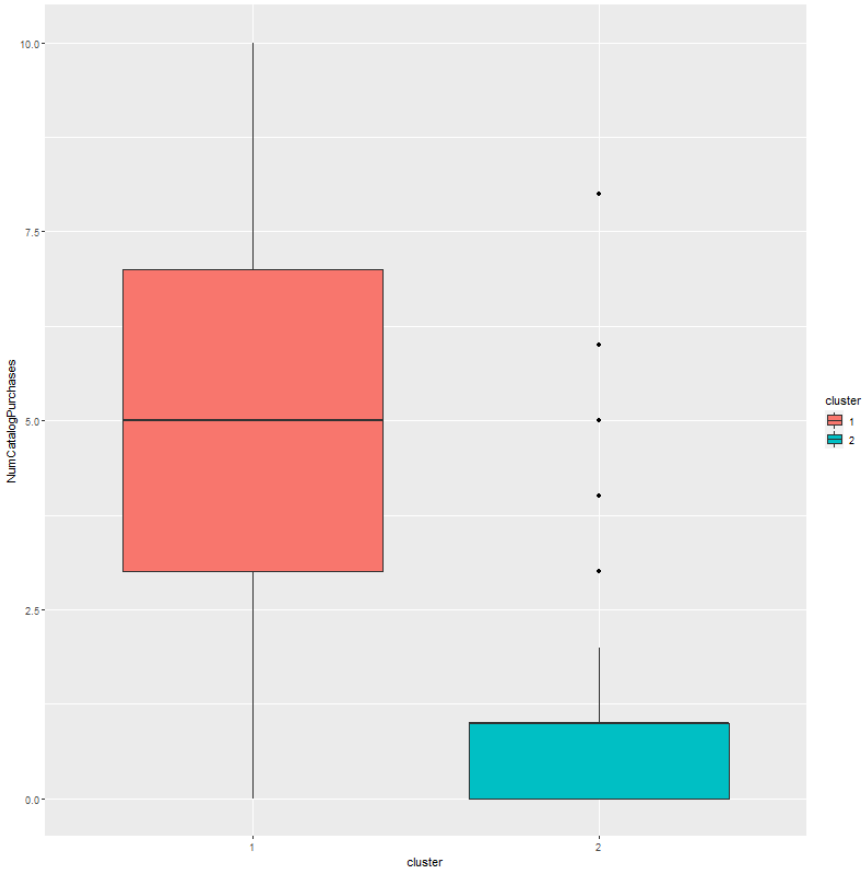


Figura 60: BoxPlot della variabile NumCatalogPurchases in relazione al numero di cluster

Dall'analisi della variabile *MntMeatProducts*, inerente all'acquisto di prodotti di carne negli ultimi due anni, si possono evincere alcune importanti informazioni correlate anche alla precedente analisi di *Total_spent*. In particolare le figure 63, 62 e 61 mostrano come gli acquirenti del secondo raggruppamento tendano a spendere generalmente di meno rispetto a quelli del primo. Difatti i secondi acquistano sicuramente molto meno rispetto alla media. In particolare si noti la differenza tra i due, di circa 170 dollari (secondo statistiche analitiche), spesi per prodotti di carne (in un arco temporale di due anni).

```
meat <- ggplot(dataSet, aes(MntMeatProducts)) + facet_grid(cluster~.)
meat + geom_histogram(color = "black", fill = "brown")
meat + geom_density(fill="brown", position = "Stack")
ggplot(dataSet,
    aes(x=cluster,y=MntMeatProducts,fill=cluster))+geom_boxplot(outlier.colour="black")
```

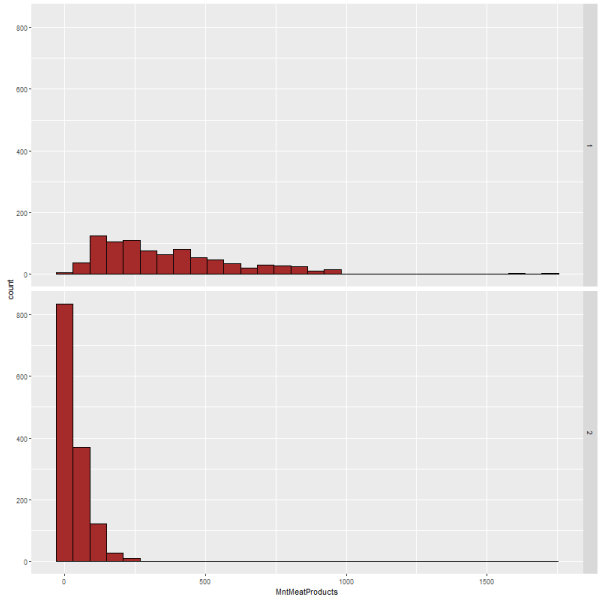


Figura 61: Istogramma della variabile MntMeatProducts in relazione al numero di cluster

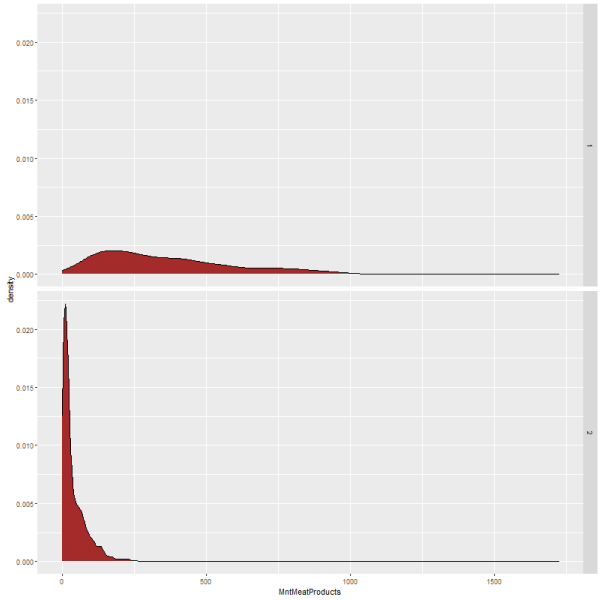


Figura 62: Diagramma di Densità della variabile MntMeatProducts in relazione al numero di cluster

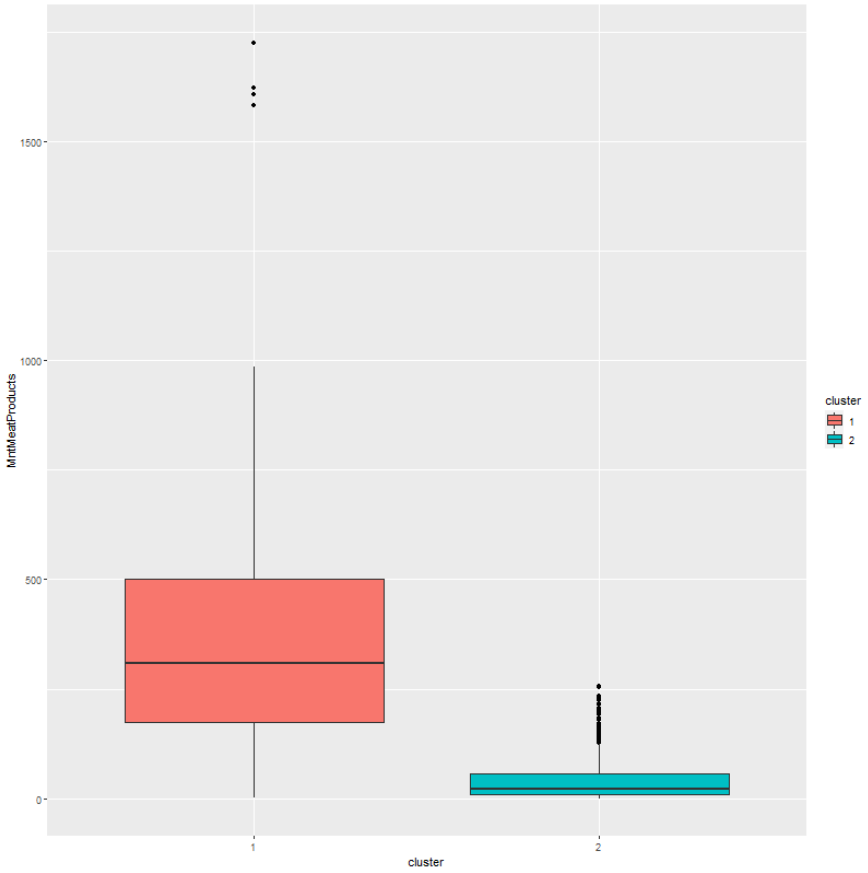


Figura 63: BoxPlot della variabile MntMeatProducts in relazione al numero di cluster

4.2 Decision Tree

Usando il dataset ricavato dalla PCA ovvero *trainingSet_input* e *testSet_input* si è cercato di fare una previsione sul valore che assume la variabile *Response*. Per farlo è stata aggiunta una variabile al dataset.

```
trainingSet_input$Response <- trainingSet$Response
testSet_input$Response <- testSet$Response
```

Successivamente è stata usata la funzione *rpart* per costruire un albero di decisione indicando come formula la variabile *Response*.

```
classifier <- rpart(Response ~ ., data = trainingSet_input, method = "class")
```

Per visualizzarlo si è eseguita la funzione prp del *package, rpart.plot* indicando per i campi richiesti un parametro come per esempio *type* che assegna a tutti i nodi un etichetta, non solo le foglie. *extra*, che se uguale ad 1 visualizza il numero di osservazioni che cadono nel nodo.

```
prp(classifier,
    type = 1, extra = 1, varlen = -10,
    box.col = ifelse(classifier$frame$var == "<leaf>", 'gray', 'white'))
```

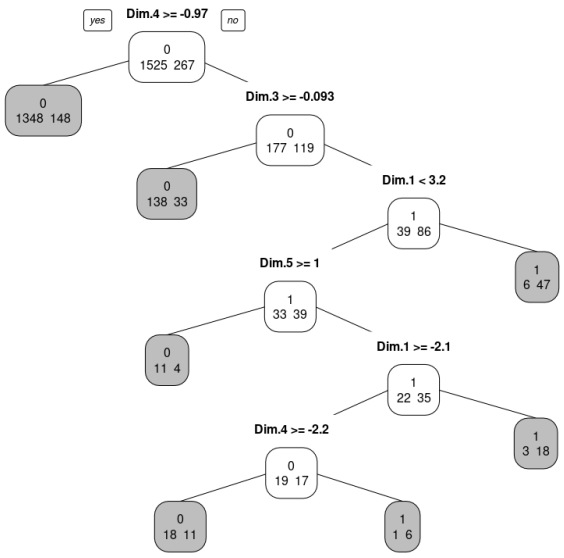


Figura 64: DecisionTree dataset PCA

Successivamente si è eseguita una previsione sulla variabile *Response* tramite la funzione *predict* passando come parametro anche l'albero *classifier*.

```
y_pred <- predict(classifier, testSet_input, type = "class")
```

Per confrontare il valore effetivo della variabile *Response*, colonna 6 del *testSet_input* e della previsione si è fatta la matrice di confusione.

```
cm = table(testSet_input[,6],y_pred)
```

Dalla matrice di confusione sommando i valori presenti nella diagonale e facendo il rapporto con la somma di tutti i valori della matrice si ricava che l'accuratezza è del 83.48%. Di seguito sono riportati altri dati statistici che è possibile ricavare.

		Prediction	
	_	0	1
Reference	0	366	15
	1	59	8

Positive Class: 1
Accuracy: 0.8348 **Precision:** 0.1194
Recall: 0.3478 **F-Measure:** 0.1777

Per capire meglio l'albero decisionale si usa la funzione printcp().

```
cv.ct <- rpart( Response ~ ., data = trainingSet_input, method ="class", cp = 0,
    minsplit = 2, xval = 10)
printcp(cv.ct)
```

	CP	nsplit	rel error	xerror	xstd
1	0.0880150	0	0.00000	1.00000	0.056456
2	0.0131086	2	0.823970	0.84270	0.052535
3	0.0074906	6	0.771536	0.85393	0.052833
4	0.0056180	9	0.749064	0.91386	0.054375
5	0.0049938	32	0.554307	0.91386	0.054375
6	0.0037453	38	0.524345	0.91386	0.054375
7	0.0028090	98	0.299625	0.99625	0.056369
8	0.0024969	114	0.250936	1.04494	0.057483
9	0.0022472	123	0.228464	1.04494	0.057483
10	0.0018727	128	0.217228	1.11236	0.058955
11	0.0012484	202	0.052434	1.11236	0.058955
12	0.000000	205	0.048689	1.11985	0.059113

Succesivamente si è cercato di evitare *overfitting* semplificando l'albero di decisione cercando prendendo prima il minimo tra una matrice di informazioni sulle strutture ottimali in base ad un parametro di complessità, cp dalla*cptable* che è pari a 0.84644. Poi si è cercato l'errore standard corrispondente al minimo errore che è 0.0526.

```
# min error
minerror <- min(cv.ct$cptable[ , 4])
minerrorstd <- cv.ct$cptable[cv.ct$cptable[,4] == minerror, 5]

simplertrees <- cv.ct$cptable[cv.ct$cptable[,4] < minerror + minerrorstd, ]

bestcp <- simplertrees[1, 1]
```

Si ricava l'insieme di alberi in cui xerror e minore della somma tra minerror e minerrorstd e da esso si trova che il parametro di complessità, cp è 0.013.

```
bestcp <- simplertrees[1, 1]
```

L'albero più semplice si può visualizzare eseguendo le istruzioni successive.

```
response.best.tree <- prune( cv.ct, cp = bestcp )
prp(response.best.tree, type = 1, extra = 1, varlen = -15, cex = 0.5,
    box.col = ifelse(response.best.tree$frame$var == "<leaf>", 'gray', 'white' ))
```

	CP	nsplit	rel error	xerror	xstd
2	0.013108614	2	0.8239700	0.8464419	0.05263442
3	0.007490637	6	0.7715356	0.8838951	0.05361429

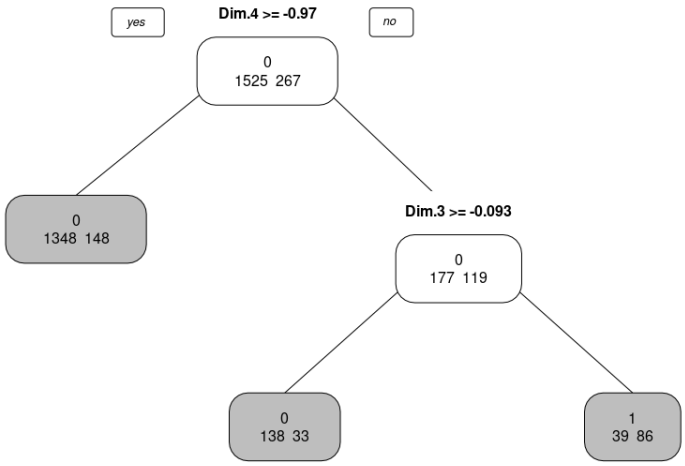


Figura 65: Prune Decision Tree cp = 0.13

Come per l'albero precedente prima di visualizzare la matrice di confusione ed altri dati statistici si è fatta una previsione.

```
response.best.tree.pred <- predict(response.best.tree, testSet_input, type = "class")
```

Ed in fine per confrontare il valore effetivo della variabile *Response* e della previsione si è fatta la matrice di confusione.

```
confusionMatrix<-confusionMatrix(response.best.tree.pred,
    as.factor(testSet_input$Response), positive = "1", mode = "prec_recall")
```

		Prediction	
	_	0	1
Reference	0	355	57
	1	26	10

Positive Class: 1
Accuracy: 0.8147 **Precision:** 0.27778
Recall: 0.14925 **F-Measure:** 0.19417

5 Esperimenti

Per valutare l'efficacia del modello supervisionato *Decision-Tree* sul dataset si sono creati 10 *fold* tramite la funzione *ceateFolds* successivamente sui fold creati si è eseguita una funzione che si ocuppa di trovare la matrice di ricavare la matrice di confusione per ogni *fold*.

```
# applying k-fold cross validation
folds= createFolds(trainingSet_input$Response, k=10)
cv = lapply(folds, function(x){

  training_fold= trainingSet_input[-x,]
  test_fold= trainingSet_input[x,]
  # building the classifier
  response.default.tree <- rpart(Response ~ ., data = training_fold, method = "class")

  # predicting values
  response.default.tree.pred <- predict(response.default.tree, newdata = test_fold, type
      = "class")

  cm = table(test_fold[,6], response.default.tree.pred)
  return (cm)
})
```

Per ricavare la matrice di confusione complessiva si riduce la lista di matrici rappresentate da *cv*in unica matrice tramite l'istruzione *Reduce()* l'insieme di matrici.

```
complex_cf =Reduce('+', cv)
complex_cf
```

		Prediction	
	_	0	1
Reference	0	1488	37
	1	200	67

Successivamente si sono ricavati i valori di *Accuracy*, *Precision*, *Recall* e *F-measure*.

```
accuracy = sum(diag(complex_cf))/sum(complex_cf)
precision = (complex_cf[2,2] /sum(complex_cf[2,1], complex_cf[2,2]))
recall = (complex_cf[2,2] /sum(complex_cf[2,2], complex_cf[1,2]))
fmeasure = (2*precision*recall)/(precision+recall)
```

Accuracy: 0.8677 **Precision:** 0.2509
Recall: 0.6442 **F-Measure:** 0.3611

Essendo un problema in cui la classificazione è binaria si è definita la curva di ROC, che rappresenta il tasso dei veri positivi (TPR) in funzione del tasso dei falsi positivi (FPR) e si è ricavato che AUC, l'area sotto la curva è pari al 0.5457 cioè la probabilità che il modello classifichi un esempio positivo casuale in modo più elevato rispetto a un esempio negativo casuale.

```
treefit= rpart(Response ~ ., data = trainingSet_input, method = "class")
predTree = predict(dtreefit,testSet_input[, !names(testSet_input) %in% c("Response")],
    probability=TRUE)
```

```
predTReeToRoc = pred[, 2]
pred.to.roc
predRocr = prediction(predTReeToRoc, testSet_input$Response)
perfRocr = performance(predRocr, measure = "auc", x.measure = "cutoff")
perfTprRocr = performance(predRocr, "tpr","fpr")
plot(perfTprRocr, colorize=T,main=paste("AUC:",(perfRocr@y.values)))
abline(a=0, b=1)
```

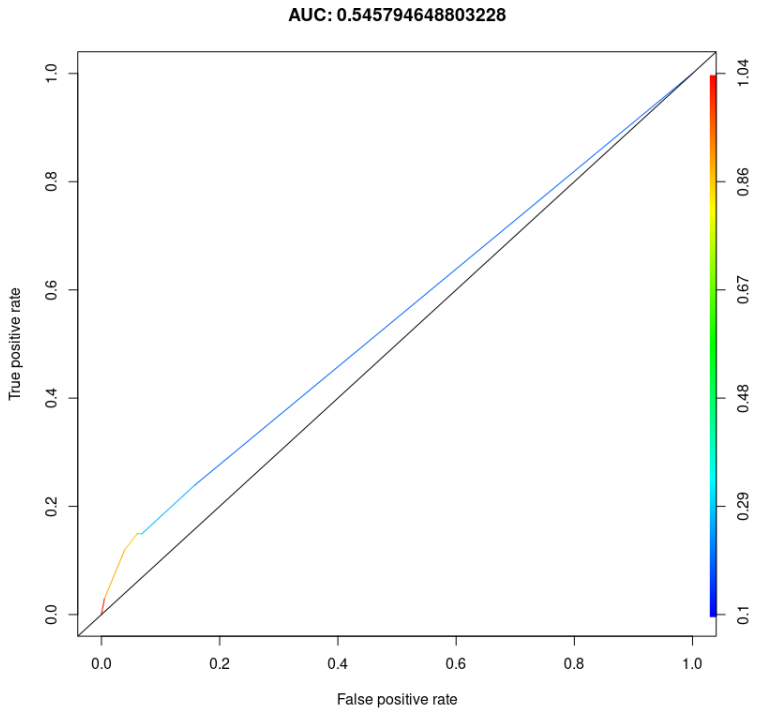


Figura 66: Grafico rappresentante la curva ROC

Conclusioni

Questa sperimentazione ha avuto l'obiettivo di analizzare un insieme di dati mediante tecniche di machine learning differenti, in particolare mediante un algoritmo di tipo supervisionato, ***Decision Tree*** ed un algoritmo non supervisionato cioè ***K-Means***. Per il modello Decision Tree si ha che l'albero più semplice, che riduce l'overfitting, ha una *Precision* e *F-measure* maggiore rispetto a quello che si ha dal classificatore e predittore ricavato usando seplicemente la funzione *rpart* e *predict* mentre i valori di *Recall* e *Accuracy* sono diminuiti. Risulta che sul dataset scelto l'algoritmo che si comporta meglio sia *K-Means*. Dall'analisi dei dati riscontrati si può giungere alla conclusione che una buona suddivisione dei dati riportati può avvenire mediante l'utilizzo di due cluster. In particolare il secondo cluster presenta clienti con un reddito generalmente al di sotto della media e sicuramente minore rispetto alla maggior parte dei compratori facenti parte della prima divisione. Secondo i dati analizzati ciò ha comportato indubbiamente una riduzione delle spese totali da parte degli elementi all'interno del secondo gruppo. La riduzione del numero di acquisti di prodotti ha generalmente toccato elementi che variano dai vini fino alla carne.