

## Report tecnico progetto Esame Analisi dei Big Data

Il seguente progetto riguarda l'analisi un dataset di biglietti aerei venduti online da una compagnia aerea che effettua collegamenti dalle principali città italiane alle principali città europee.

### Analisi

Il presente progetto contiene i seguenti punti:

#### Analisi descrittiva del dataset:

- a. Descrizione dei dati osservati (indici statistici univariati e/o bivariati, rappresentazioni grafiche, ecc.);
- b. Segmentazione della clientela sulla base delle variabili osservate;
- c. Suddivisione casuale del dataset in campione di apprendimento ( $2/3 \cdot n$ ) e campione test ( $1/3 \cdot n$ ).  
Stima, sul campione di apprendimento, di 3 modelli (Lasso, PCR, PLS) in funzione della variabile “Cfidelity” (-> factor, levels:alta, bassa. Creata tramite ricodifica della Variabile “Fidelity”).

#### Importazione del dataset

```
load("~/Documents/r Wdirectory/Dati_Caria.RData")
summary(dati)

##      Data          Unit_Ticket      Seat      PriorBoard  Luggage
##  Min.   :2010-01-01  Min.   :87.9  No :11765  No :11576  No :13111
##  1st Qu.:2012-06-21  1st Qu.:158.0  Yes:12233  Yes:12422  Yes:10887
##  Median :2014-12-27  Median :173.8
##  Mean   :2014-12-28  Mean   :175.3
##  3rd Qu.:2017-06-30  3rd Qu.:191.5
##  Max.   :2019-12-31  Max.   :274.6
##      Return        Fidelity      BookTime      Taxes      Discount
##  No :12897  Min.   : 0.00  Min.   : 3.00  Min.   : 8.88  Min.   : 0.000
##  Yes:11101  1st Qu.: 33.35  1st Qu.: 9.00  1st Qu.:20.23  1st Qu.: 0.860
##                  Median : 43.06  Median :11.00  Median :23.33  Median : 1.940
##                  Mean   : 45.31  Mean   :11.76  Mean   :23.68  Mean   : 3.493
##                  3rd Qu.: 56.29  3rd Qu.:15.00  3rd Qu.:27.00  3rd Qu.: 6.210
##                  Max.   :100.00  Max.   :22.00  Max.   :41.32  Max.   :15.160
##      NPax        Departure      Arrival      ModPag
##  Min.   : 1.000  Milan   :5481  London   :11212  C_debito : 3095
##  1st Qu.: 4.000  Rome    :9858  Paris    : 192  C_credito:20903
##  Median : 6.000  Naples  : 380  Madrid   :2938
##  Mean   : 5.726  Palermo:2806  Amsterdam: 9656
##  3rd Qu.: 7.000  Bergamo:1656
```

```

##  Max.   :12.000  Venice :3817
##  NBookMonth    NAccWebWeek   NComplaintsYear  NRefundYear
##  Min.   :0.000  Min.   :0.000  Min.   :0.000  Min.   : 0.000
##  1st Qu.:3.000  1st Qu.:2.000  1st Qu.:1.000  1st Qu.: 0.000
##  Median :4.000  Median :3.000  Median :1.000  Median : 0.000
##  Mean   :3.993  Mean   :2.996  Mean   :1.157  Mean   : 5.853
##  3rd Qu.:5.000  3rd Qu.:4.000  3rd Qu.:2.000  3rd Qu.: 0.000
##  Max.   :9.000  Max.   :7.000  Max.   :6.000  Max.   :65.000
##  NCcancelYear
##  Min.   : 1.00
##  1st Qu.: 4.00
##  Median : 5.00
##  Mean   : 5.01
##  3rd Qu.: 6.00
##  Max.   :10.00

str(dati)

## 'data.frame': 23998 obs. of 19 variables:
## $ Data      : Date, format: "2010-01-01" "2010-01-01" ...
## $ Unit_Ticket: num 174 160 173 189 152 ...
## $ Seat       : Factor w/ 2 levels "No","Yes": 2 1 2 2 1 1 2 1 1 2 ...
## $ PriorBoard: Factor w/ 2 levels "No","Yes": 2 2 1 1 2 1 2 2 2 1 ...
## $ Luggage    : Factor w/ 2 levels "No","Yes": 2 1 1 1 2 1 1 2 2 2 ...
## $ Return     : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 1 1 1 1 1 ...
## $ Fidelity   : num 53.5 47.9 39 40.9 30.1 ...
## $ BookTime   : num 18 12 10 14 13 15 14 18 18 16 ...
## $ Taxes      : num 24.6 24.5 18.7 19.6 14.9 ...
## $ Discount   : num 6.97 1.23 0.7 1.01 0.48 1.33 1.06 9.91 8.97 8.02 ...
## $ NPax       : num 4 6 4 9 4 5 5 6 7 7 ...
## $ Departure  : Factor w/ 6 levels "Milan","Rome",...: 2 2 1 6 2 2 2 2 5 4 ...
## $ Arrival    : Factor w/ 4 levels "London","Paris",...: 1 1 4 1 1 1 1 1 1 ...
## $ ModPag     : Factor w/ 2 levels "C_debito","C_credito": 2 2 2 2 2 2 2 1 2 2 ...
## $ NBookMonth : num 5 4 3 5 5 4 5 4 4 4 ...
## $ NAccWebWeek: num 3 3 3 5 3 2 4 3 2 3 ...
## $ NComplaintsYear: num 1 0 2 3 1 1 2 0 0 1 ...
## $ NRefundYear: num 0 0 27 0 0 0 28 0 14 0 ...
## $ NCcancelYear: num 3 4 6 6 4 4 5 5 4 5 ...

sum(is.na(dati)) #non sono presenti valori mancanti

## [1] 0

```

Per prima cosa, è stato importato il dataset. Dal summary si può osservare che non sono presenti valori mancanti. Dalla funzione str() si osserva che ci sono 11 variabili num e 7 factor.

## Analisi esplorativa del dataset: grafici

```

library("tidyverse")

## -- Attaching packages ----- tidyverse 1.3.1 --
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyverse 1.1.3    v stringr 1.4.0
## v readr   1.4.0     vforcats 0.5.1

```

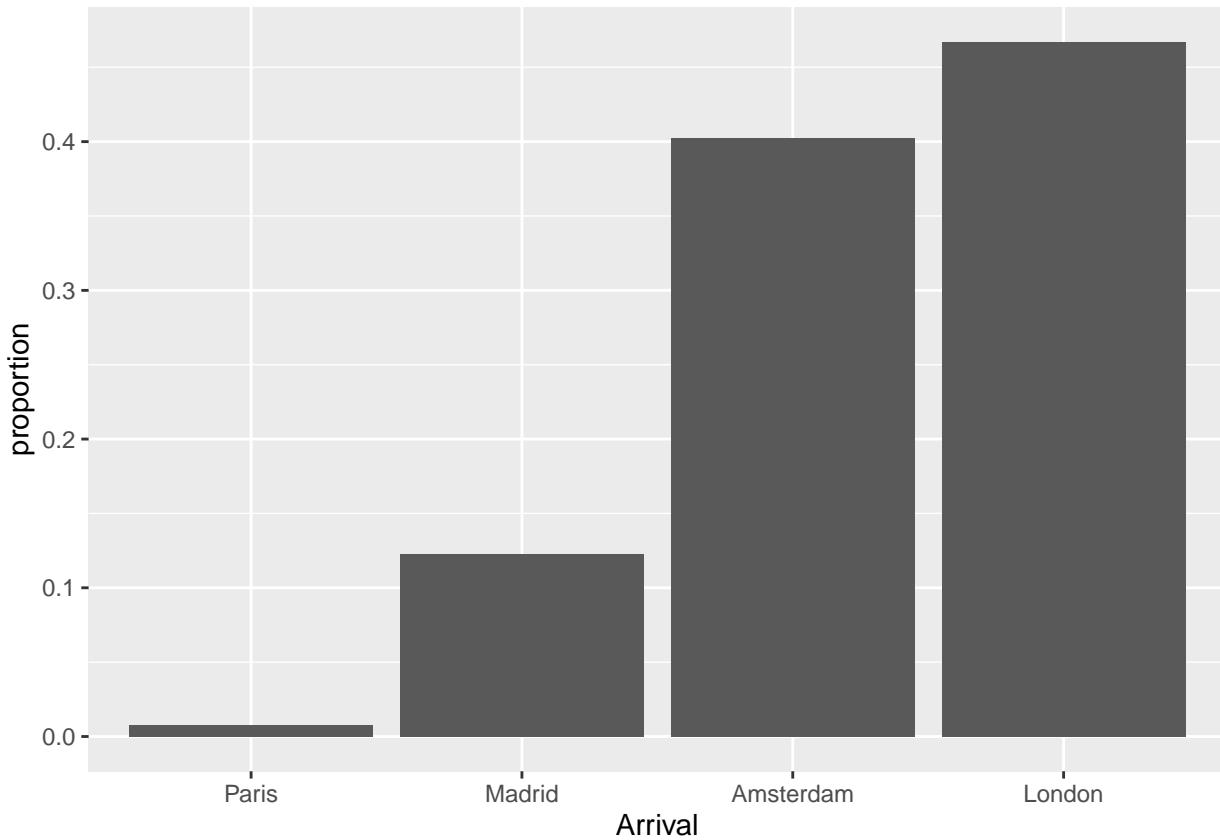
```

## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

#1
tab <-dati %>%
  count(Arrival) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(Arrival= reorder(Arrival, proportion))

tab %>%
  ggplot(aes(Arrival, proportion))+
  geom_bar(stat = "identity")

```



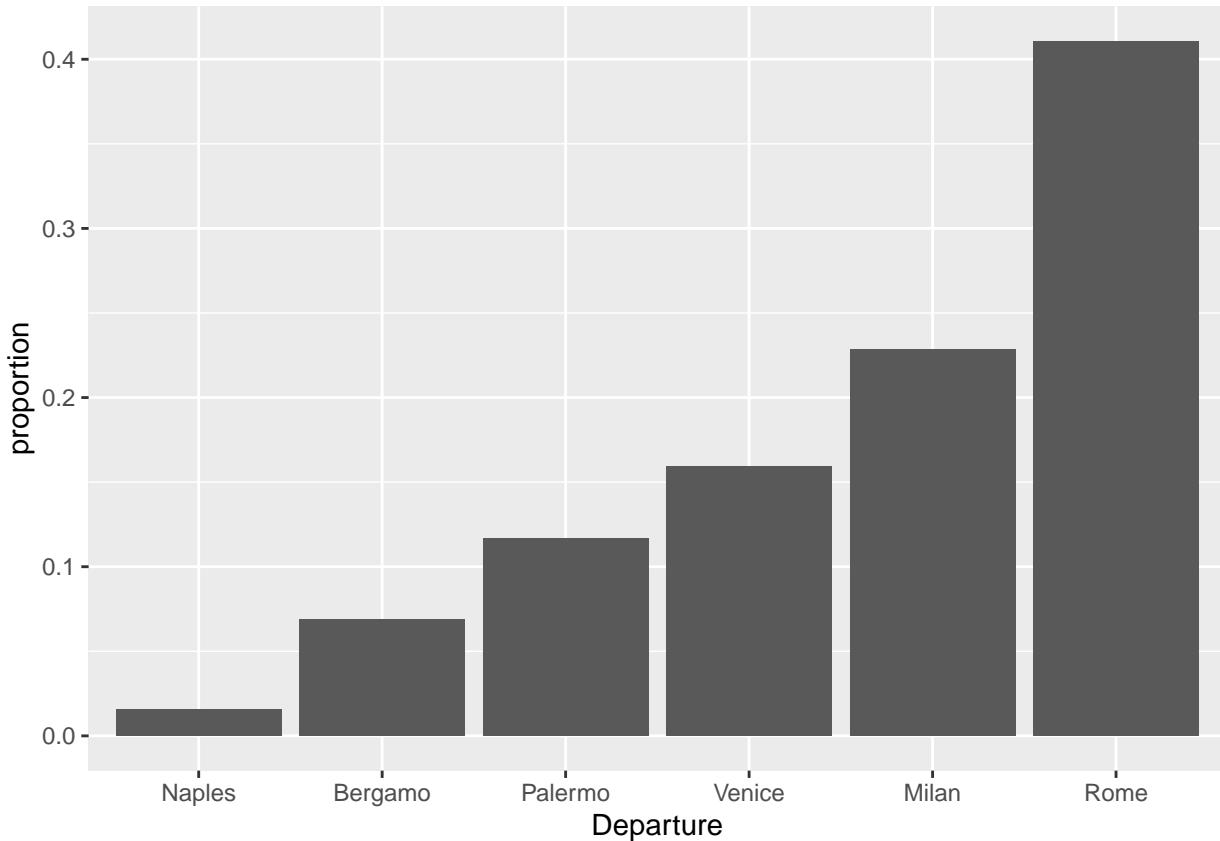
La proporzione di biglietti venduti per le 4 diverse destinazioni

```

tab2 <-dati %>%
  count(Departure) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(Departure= reorder(Departure, proportion))

tab2 %>%
  ggplot(aes(Departure, proportion))+
  geom_bar(stat = "identity")

```



percentuale di biglietti venduti in base ai 6 diversi aeroporti di provenienza

```
library("gridExtra")

##
## Attaching package: 'gridExtra'
## The following object is masked from 'package:dplyr':
##      combine

tab3 <- dati %>%
  count(ModPag) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(ModPag = reorder(ModPag, proportion))

tab3<-tab3 %>%
  ggplot(aes(ModPag, proportion))+
  geom_bar(stat = "identity") # quasi il 90% (0.87) dei clienti ha utilizzato
#la carta di credito per il pagamento

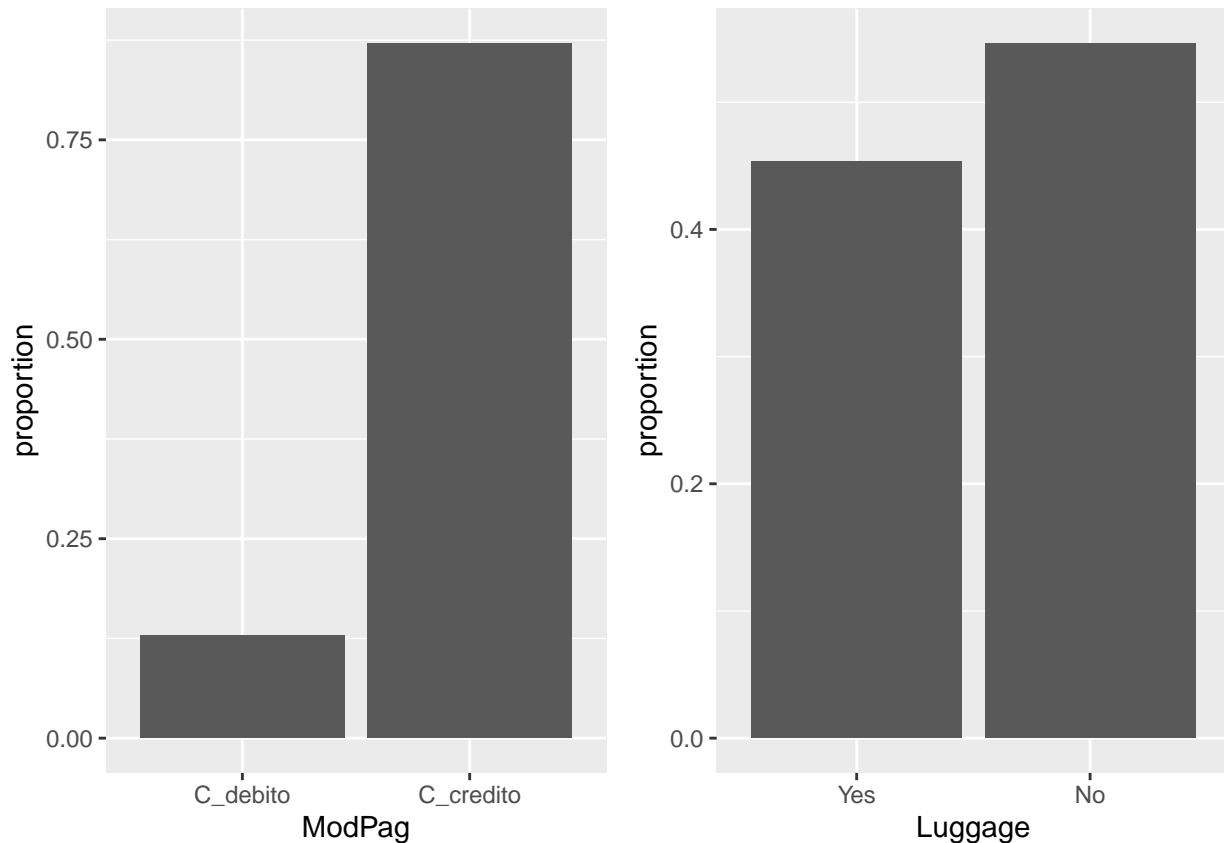
#4
tab4 <- dati %>%
  count(Luggage) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(Luggage = reorder(Luggage, proportion))

tab4<- tab4 %>%
  ggplot(aes(Luggage, proportion))+
```

```

geom_bar(stat = "identity") # il 54% dei clienti ha effettuato una
#prenotazione senza imbarcare un bagaglio aggiuntivo
grid.arrange(tab3, tab4, ncol=2)

```



Quasi il 90% (0.87) dei clienti ha utilizzato la carta di credito per il pagamento, mentre dal grafico a sinistra osserviamo che il 54% dei clienti ha effettuato una prenotazione senza imbarcare un bagaglio aggiuntivo

```

#5
tab5 <- dati %>%
  count(PriorBoard) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(PriorBoard = reorder(PriorBoard, proportion))

tab5<- tab5 %>%
  ggplot(aes(PriorBoard, proportion))+ 
  geom_bar(stat = "identity") # il 52% (0.517) dei clienti ha effettuato una
#prenotazione con imbarco prioritario

#6
tab6 <- dati %>%
  count(Seat) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(Seat = reorder(Seat, proportion))

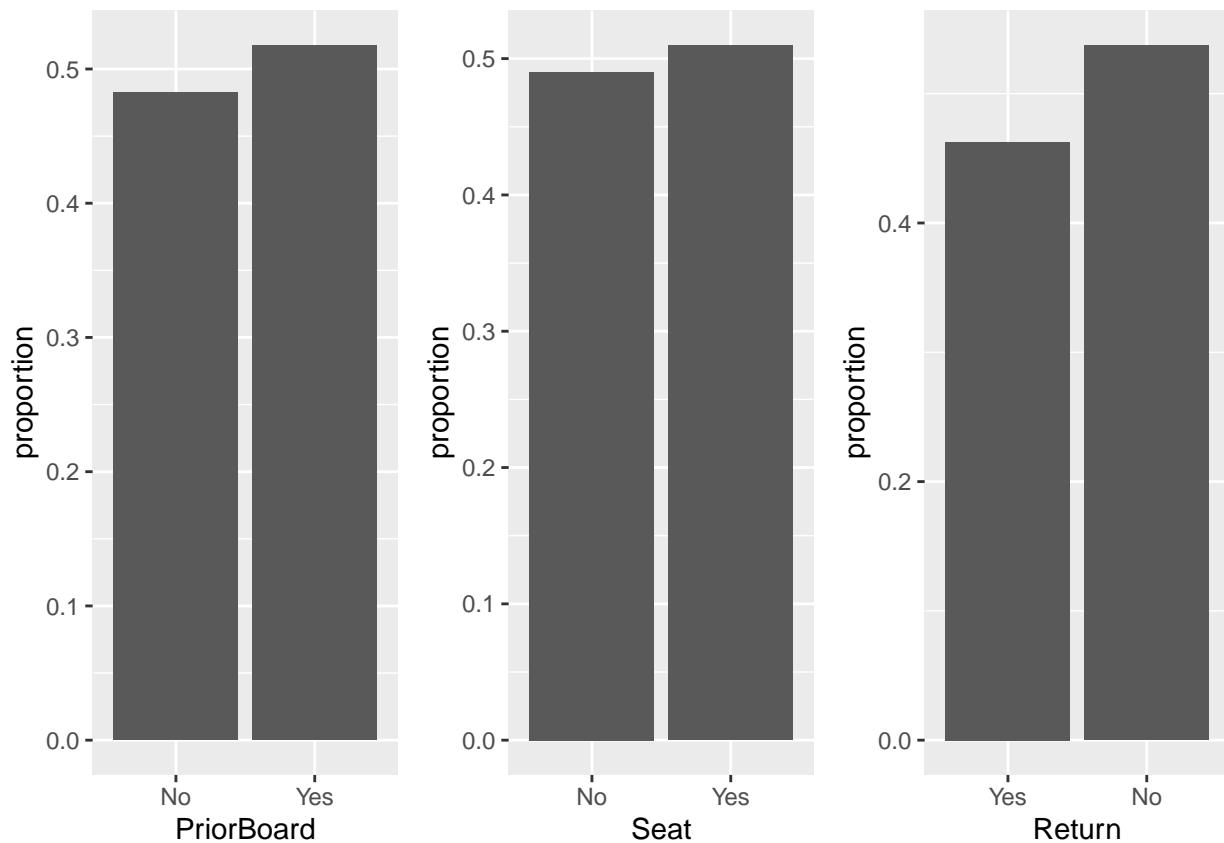
tab6<- tab6 %>%
  ggplot(aes(Seat, proportion))+ 
  geom_bar(stat = "identity") # il 51% (0.509) dei clienti ha effettuato una

```

```
#prenotazione del posto a sedere
```

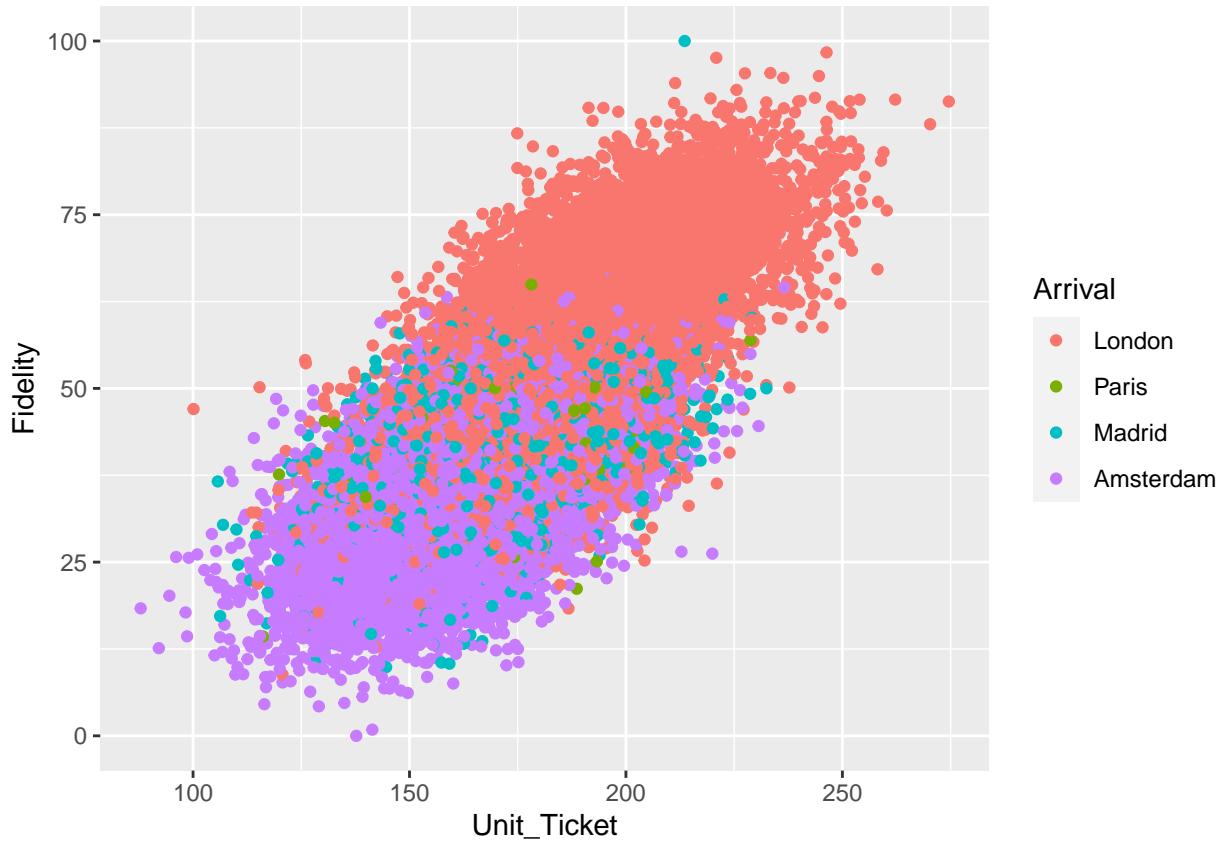
```
#7
tab7 <- dati %>%
  count(Return) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(Return = reorder(Return, proportion))

tab7<-tab7 %>%
  ggplot(aes(Return, proportion))+
  geom_bar(stat = "identity") # il 54% (0.537) dei clienti ha effettuato una
#prenotazione solo andata
grid.arrange(tab5, tab6, tab7, ncol=3)
```



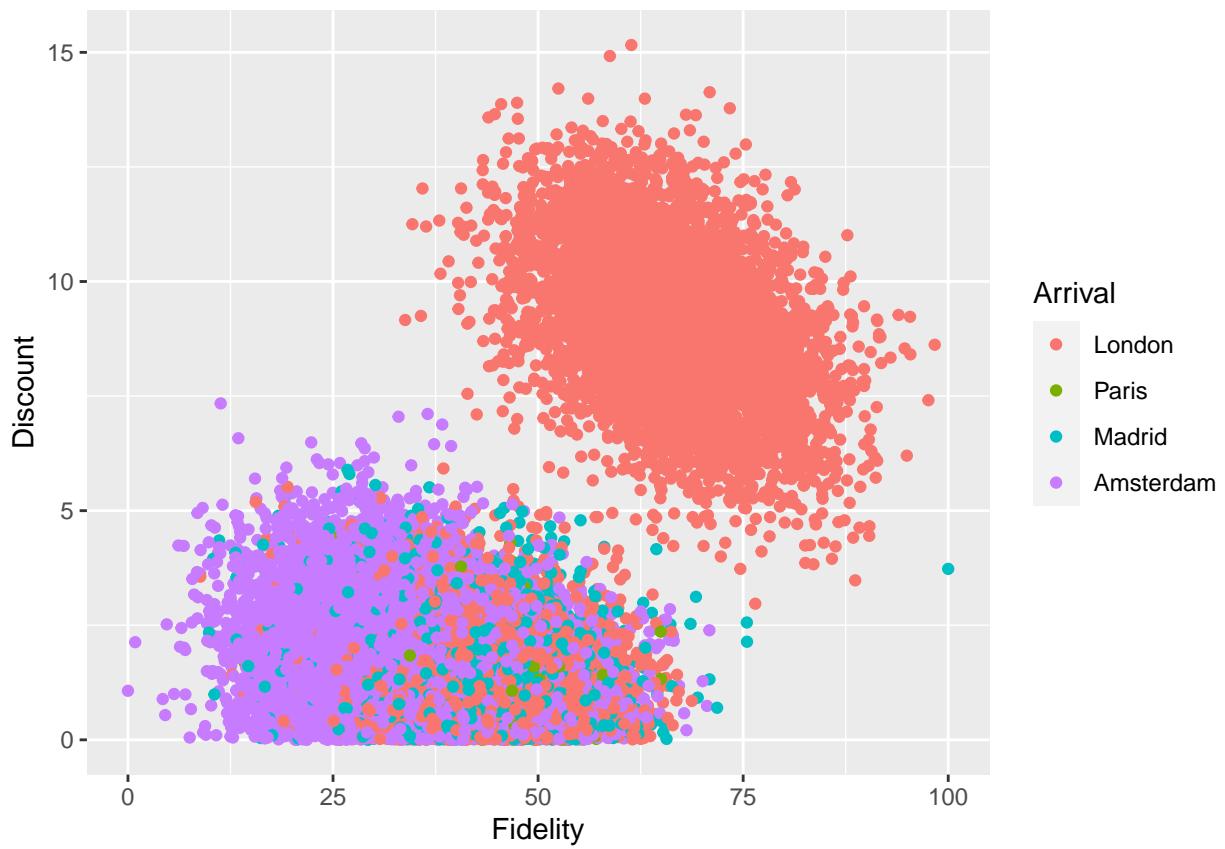
Il primo grafico a sinistra mostra che il 52% (0.517) dei clienti ha effettuato una prenotazione con imbarco prioritario. Dal secondo grafico si osserva che il 51% (0.509) dei clienti ha effettuato una prenotazione del posto a sedere. Dal terzo grafico invece, si osserva che il 54% (0.537) dei clienti ha effettuato una prenotazione solo andata.

```
filter(dati) %>%
  ggplot(aes(Unit_Ticket, Fidelity, color= Arrival)) +
  geom_point()
```



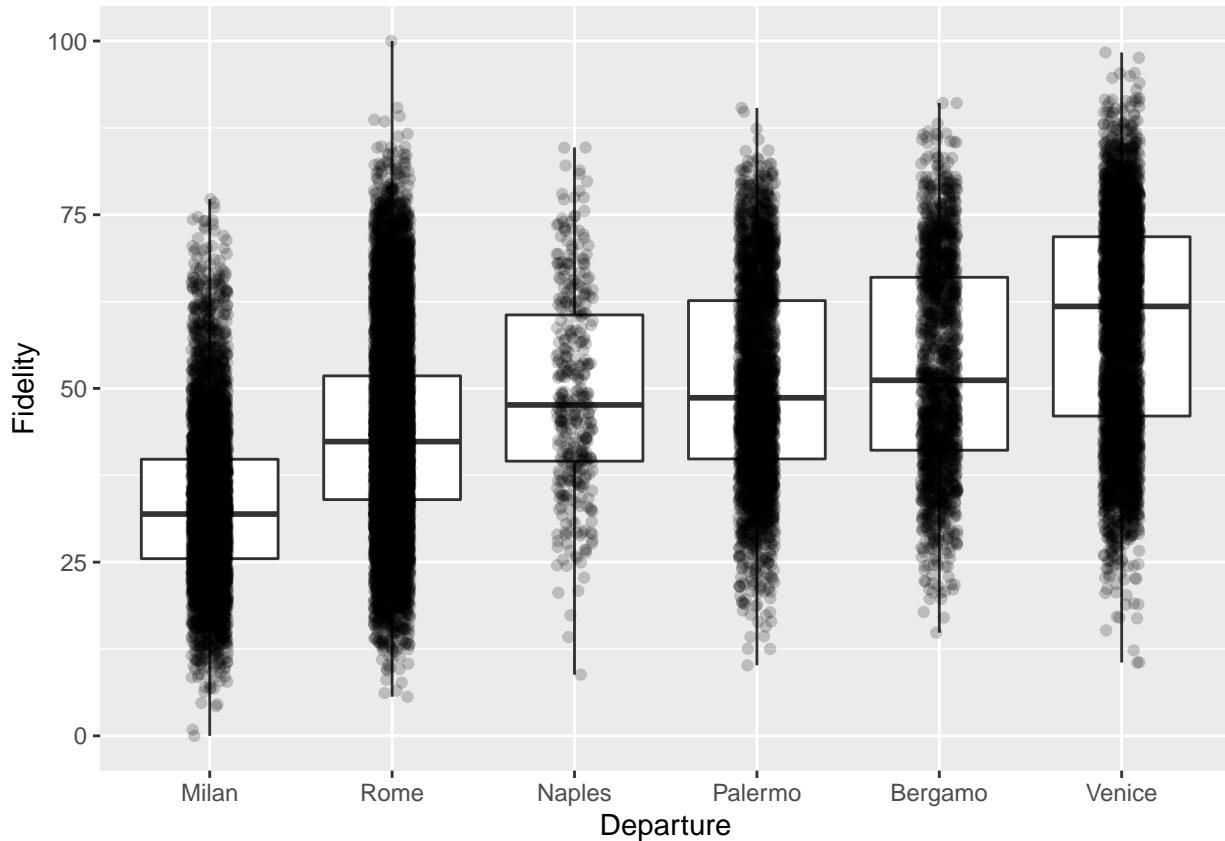
Il prezzo dei biglietti di Londra è più alto rispetto a quello di Amsterdam e Madrid. Il costo inferiore sembra essere dovuto al fatto che l'indice di fidelizzazione del cliente è basso per quest'ultime, mentre assume valori alti per Londra.

```
filter(dati) %>%
  ggplot(aes(Fidelity, Discount, color= Arrival)) +
  geom_point()
```



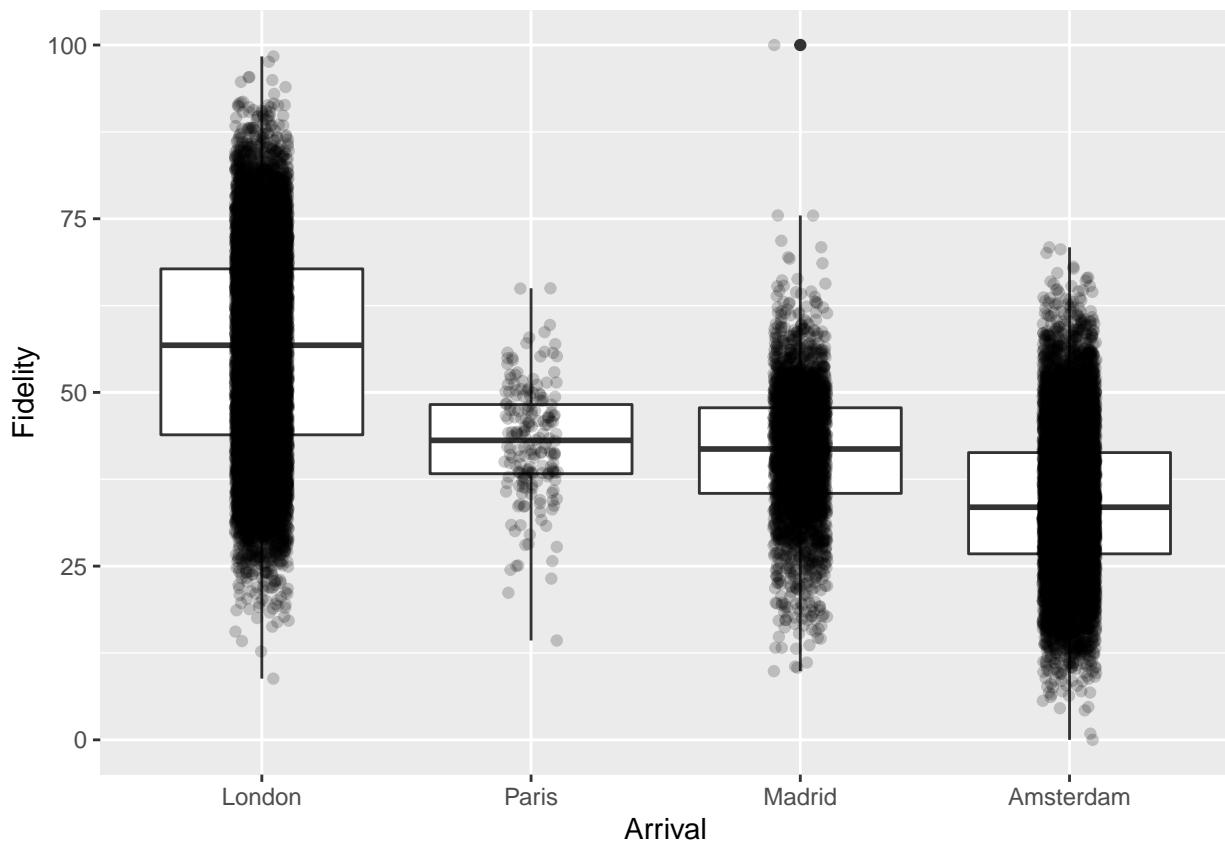
I clienti che acquistano i biglietti per Londra risultano essere più fedeli rispetto a quelli di Amsterdam e Madrid e per questo godono di uno sconto maggiore a dispetto di quest'ultimi

```
dati %>%
  ggplot(aes( y=Fidelity, Departure)) +
  geom_boxplot(coef=3) +
  geom_jitter(width= 0.1, alpha =0.2)
```



l'indice di fedeltà risulta mediamente più alto per le prenotazioni con partenza da Venezia

```
dati %>%
  ggplot(aes( y=Fidelity, Arrival)) +
  geom_boxplot(coef=3) +
  geom_jitter(width= 0.1, alpha =0.2)
```



Per quanto riguarda l'areoporto di Arrivo, l'indice di fedeltà risulta essere mediamente più alto per le prenotazioni con arrivo a Londra

```
library("lubridate")

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##   date, intersect, setdiff, union
dates<-ymd(dati>Data)

month_day_year<-tibble(date = dates,
                       month = month(dates, label = TRUE),
                       day = day(dates),
                       year = year(dates))

head(month_day_year)

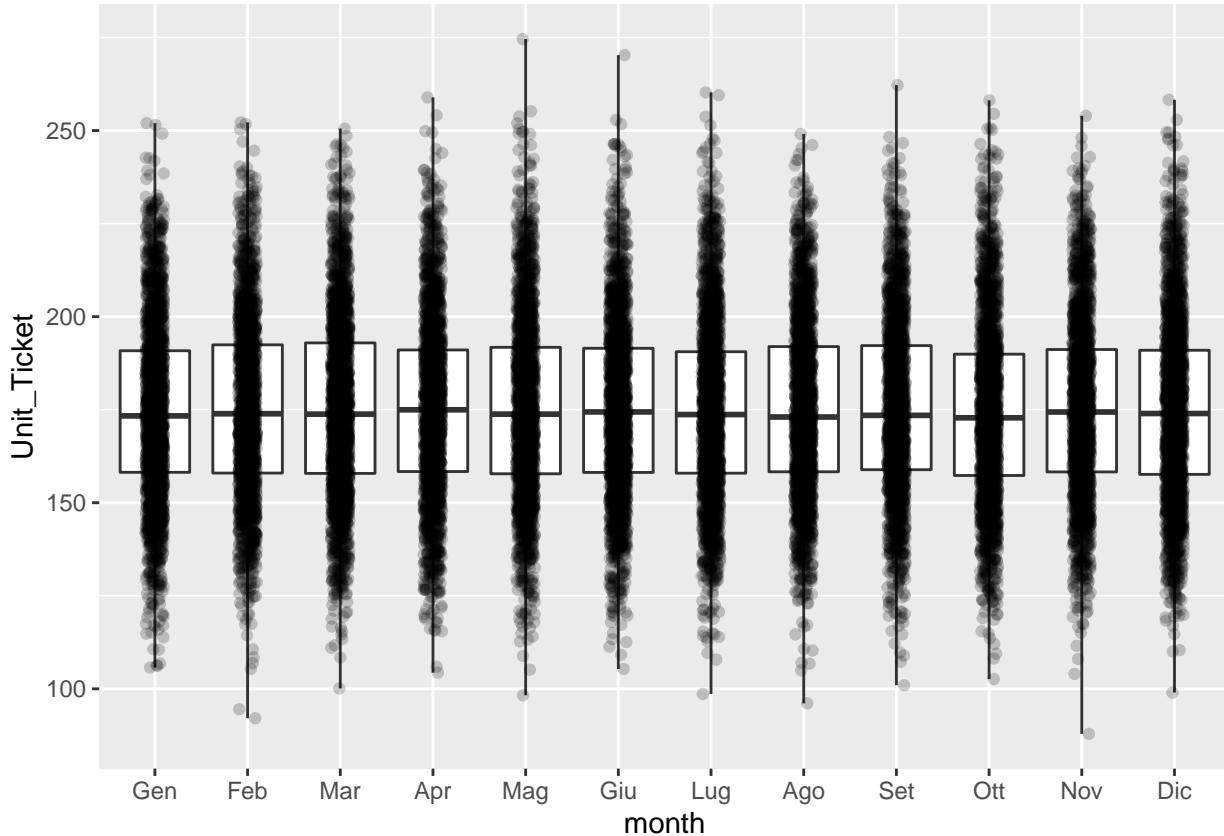
## # A tibble: 6 x 4
##   date      month   day   year
##   <date>    <ord> <int> <dbl>
## 1 2010-01-01 Gen     1  2010
## 2 2010-01-01 Gen     1  2010
## 3 2010-01-01 Gen     1  2010
## 4 2010-01-01 Gen     1  2010
```

```

## 5 2010-01-01 Gen      1 2010
## 6 2010-01-01 Gen      1 2010

#11  graf
dati2= dati
dati2<- mutate(dati2, month= month_day_year$month ) #inserisco una nuova colonna
dati2 %%
  ggplot(aes(month, Unit_Ticket)) +
  geom_boxplot(coef=3) +
  geom_jitter(width= 0.1, alpha =0.2)

```



*#le distribuzioni del prezzo del biglietto nel corso dell'anno è molto simile. Il prezzo del biglietto ;*

Dopo aver creato un oggetto tibble per dividere giorno, mese e anno, inserisco una nuova colonna in una copia del dataset (dati2). tramite ggplot, ottengo dei boxplot riguardanti le distribuzioni del prezzo del biglietto nel corso dell'anno. tali distribuzioni risultano essere molto simili. Il prezzo del biglietto più basso si è registrato a Novembre, il più prezzo più alto è stato registrato a maggio.

```

#ricodifica dell'indice di fedeltà
dati2$clientFidelity[dati2$Fidelity<=50] = "basso"
dati2$clientFidelity[dati2$Fidelity>50] = "alto"

fedeltà <-as.factor(dati2$clientFidelity)
class(fedeltà)

## [1] "factor"

dati2<- dati2 %>% mutate(Cfidelity= fedeltà)

#eliminazione colonne rindondanti dopo la ricodifica

```

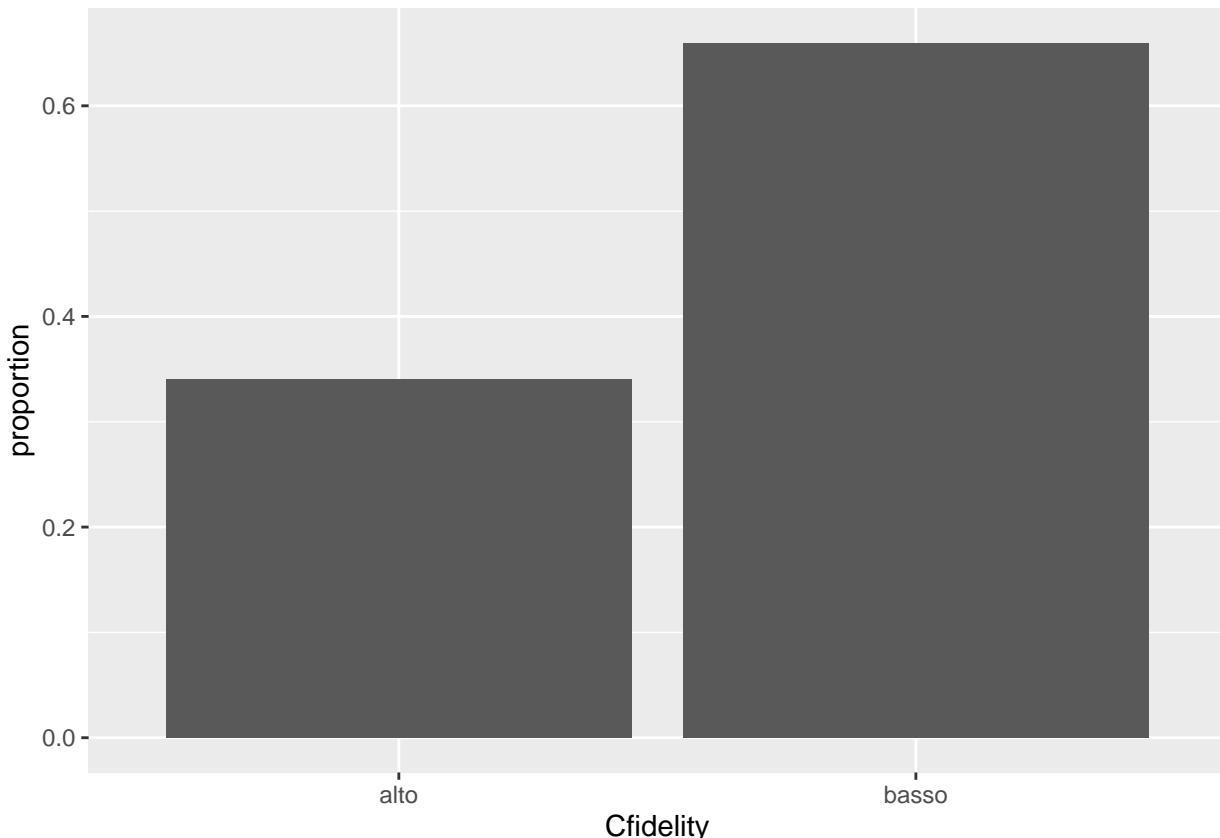
```

dati2= dati2[-21] #elimino ClientFidelity
dati2= dati2[-7] #elimino la vecchia colonna Fidelity
dati2= dati2[-19]

#12
tab12 <- dati2 %>%
  count(Cfidelity) %>%
  mutate(proportion= n/sum(n)) %>%
  mutate(Cfidelity = reorder(Cfidelity, proportion))

tab12 %>%
  ggplot(aes(Cfidelity, proportion))+
  geom_bar(stat = "identity")

```



è stato ricodificato l'indice di fedelizzazione: è stata creata una nuova variabile Cfidelity) in cui se l'indice di fedelizzazione (tra 0 e 100) è maggiore di 50 verrà assegnato il valore “alto”, se invece sarà minore di 50 “basso”. Il 66% (0.659) dei clienti ha un indice di fedelizzazione basso (minore al 50) mentre il 34 % dei clienti ha un valore alto per questo indice (maggiore di 50)

```
library(dslabs)
```

```

arrival_departures <- count(dati, Departure, Arrival) #creo un nuovo oggetto
#tibble in cui conto i viaggi in base a aeroporto di partenza e aeroporto
#di arrivo

```

```

arrival_departures

##      Departure   Arrival     n
## 1      Milan    London 1022
## 2      Milan    Paris   32
## 3      Milan   Madrid  490
## 4      Milan Amsterdam 3937
## 5      Rome     London 4448
## 6      Rome     Paris   89
## 7      Rome     Madrid 1432
## 8      Rome Amsterdam 3889
## 9      Naples   London 229
## 10     Naples   Paris    2
## 11     Naples   Madrid  57
## 12     Naples Amsterdam 92
## 13     Palermo  London 1644
## 14     Palermo  Paris   28
## 15     Palermo  Madrid  405
## 16     Palermo Amsterdam 729
## 17     Bergamo  London 1057
## 18     Bergamo  Paris   19
## 19     Bergamo  Madrid  210
## 20     Bergamo Amsterdam 370
## 21     Venice   London 2812
## 22     Venice   Paris   22
## 23     Venice   Madrid  344
## 24     Venice Amsterdam 639

summary(arrival_departures)

##      Departure       Arrival        n
##  Milan :4    London :6 Min.   : 2.0
##  Rome  :4    Paris  :6 1st Qu.: 81.0
##  Naples :4   Madrid :6 Median :387.5
##  Palermo:4 Amsterdam:6 Mean   :999.9
##  Bergamo:4                      3rd Qu.:1150.8
##  Venice :4                           Max.   :4448.0

#13
p1<-arrival_departures %>% unite(flights, Departure, Arrival) #unisco le colonne
#Unite tramite la funzione Unite della libreria Tidyr
p1

##           flights     n
## 1      Milan_London 1022
## 2      Milan_Paris   32
## 3      Milan_Madrid  490
## 4      Milan_Amsterdam 3937
## 5      Rome_London 4448
## 6      Rome_Paris   89
## 7      Rome_Madrid 1432
## 8      Rome_Amsterdam 3889
## 9      Naples_London 229
## 10     Naples_Paris    2
## 11     Naples_Madrid  57

```

```

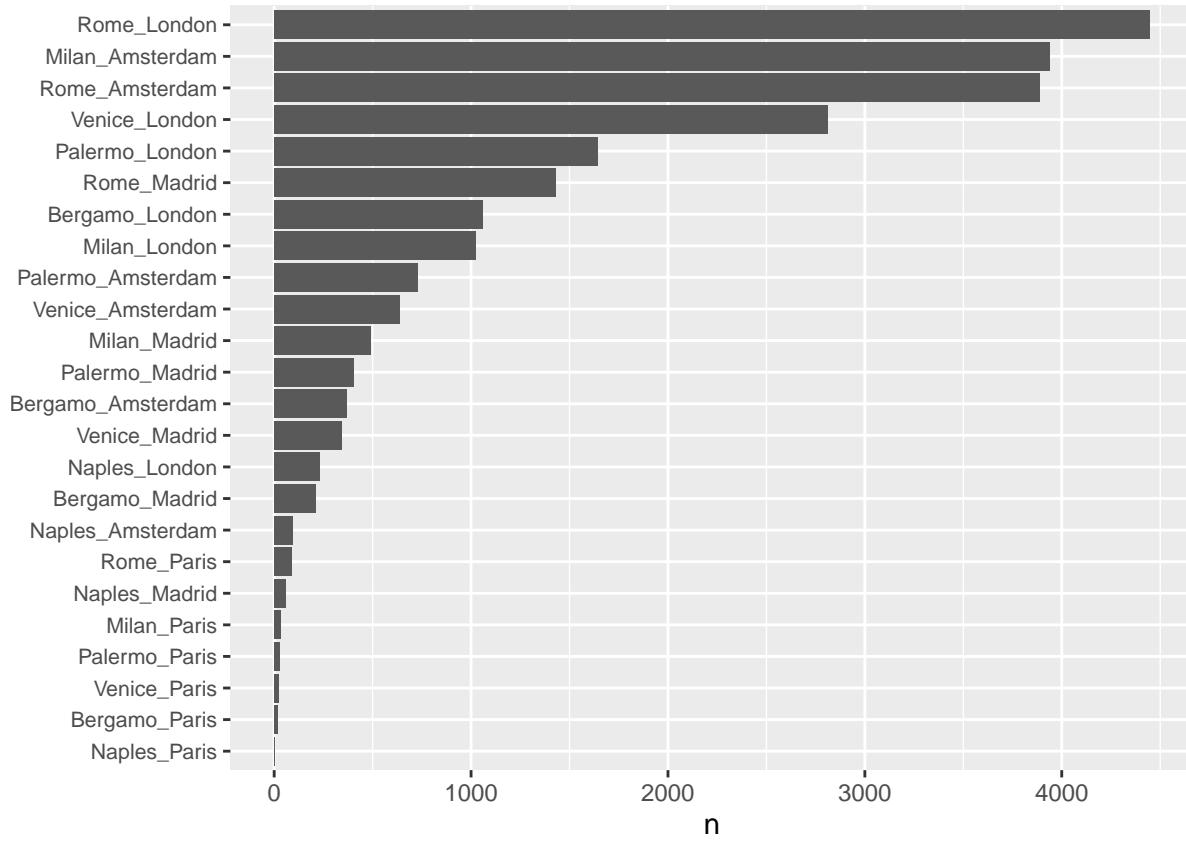
## 12 Naples_Amsterdam 92
## 13 Palermo_London 1644
## 14 Palermo_Paris 28
## 15 Palermo_Madrid 405
## 16 Palermo_Amsterdam 729
## 17 Bergamo_London 1057
## 18 Bergamo_Paris 19
## 19 Bergamo_Madrid 210
## 20 Bergamo_Amsterdam 370
## 21 Venice_London 2812
## 22 Venice_Paris 22
## 23 Venice_Madrid 344
## 24 Venice_Amsterdam 639

```

```

p1 %>%
  mutate( flights= reorder(flights, n)) %>%
  ggplot(aes(flights, n)) +
  geom_bar(stat= "identity")+
  coord_flip()+
  theme(axis.text.y = element_text(size = 8)) +
  xlab("")

```



Si osserva che i biglietti maggiormente venduti nel sito sono per il volo Roma-Londra, seguito da Milano-Amsterdam e Roma-Amsterdam

```

index<- dati2$Unit_Ticket; x<- dati2$Unit_Ticket[index]
m <-mean(x) ; s<- sd(x)
c(average=m, sd= s )

```

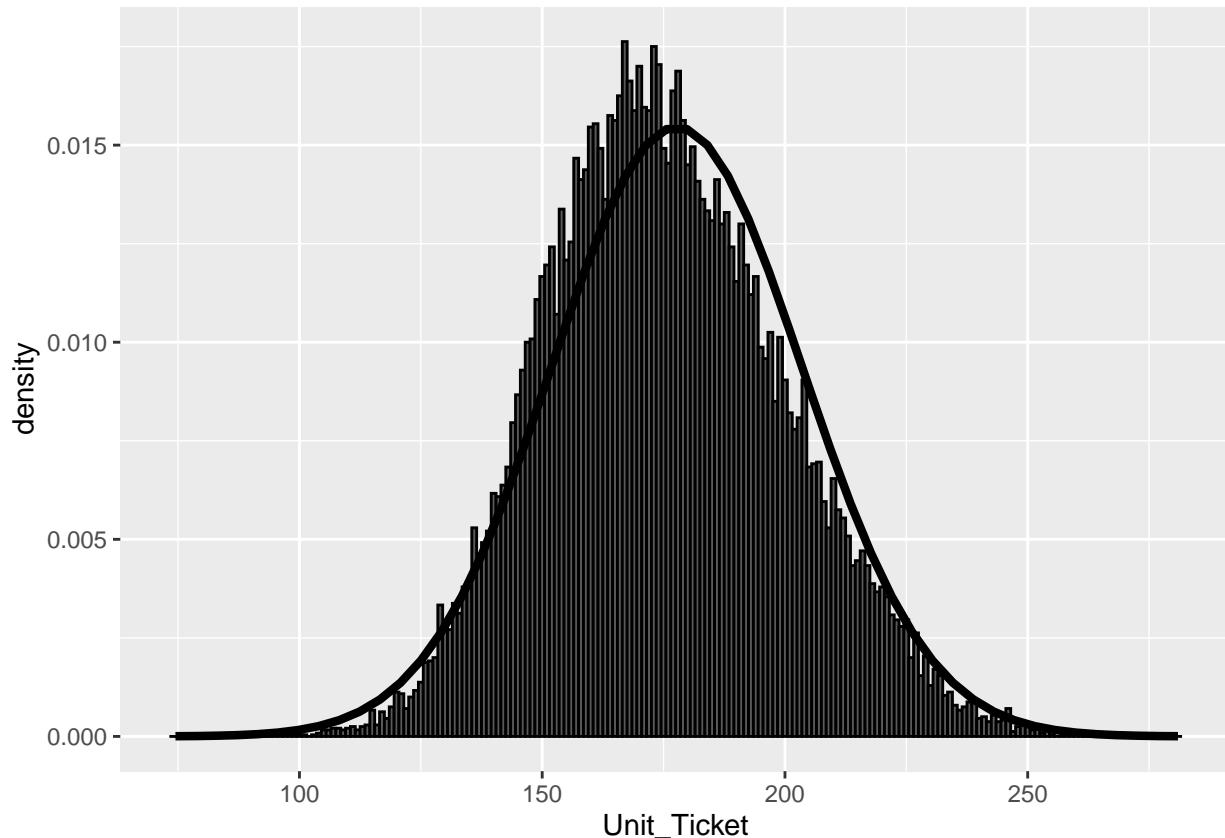
```

##   average      sd
## 177.70689 25.81076

norm_dist<- data.frame(x= seq(-4, 4, len=50)*s + m) %>%
  mutate(density= dnorm(x, m, s))

#14
dati2 %>% ggplot(aes(Unit_Ticket))+
  geom_histogram(aes(y=..density..), binwidth = 1, color = "black")+
  geom_line(aes(x, density), data= norm_dist, lwd=1.5)

```



La distribuzione del prezzo dei biglietti venduti sembra essere approssimabile a una normale.

## PCA

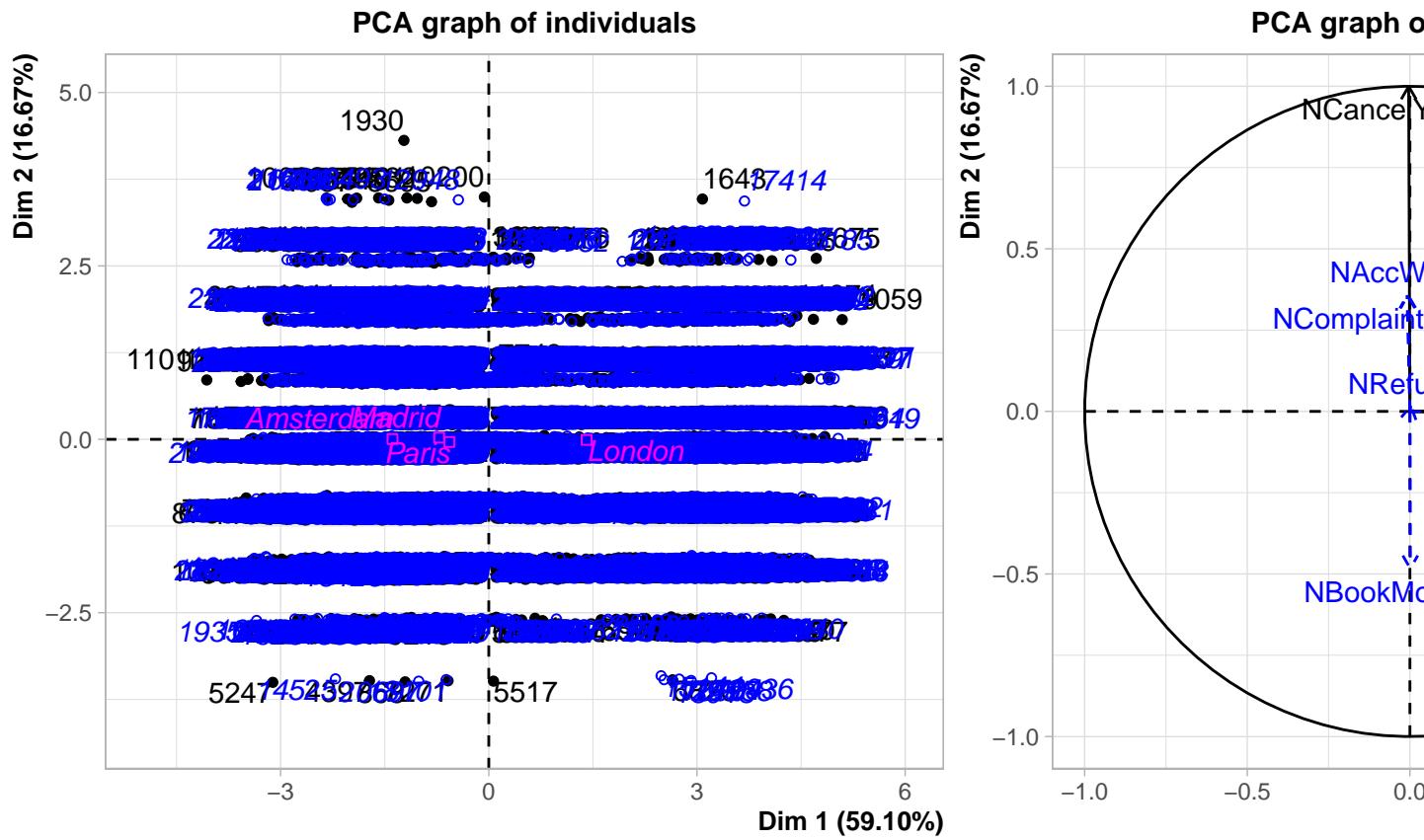
```

library(FactoMineR)

attach(dati)
new_dati<- tibble(Unit_Ticket, Fidelity, BookTime, Taxes, Discount, NPax, NBookMonth,NAccWebWeek, NComp)

new_dati2 <- tibble(Unit_Ticket, Fidelity, BookTime, Taxes, Discount, NPax, NBookMonth,NAccWebWeek, NComp)
res <- PCA(new_dati2,
           scale.unit = TRUE,
           ind.sup = 12000: 23998,
           quanti.sup = 6:10,
           quali.sup = 12,
           ncp=8)

```



```
summary(res)

##
## Call:
## PCA(X = new_dati2, scale.unit = TRUE, ncp = 8, ind.sup = 12000:23998,
##      quanti.sup = 6:10, quali.sup = 12)
##
## 
## Eigenvalues
##              Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6
## Variance     3.546   1.000   0.585   0.470   0.229   0.169
## % of var.  59.097  16.671   9.758   7.839   3.811   2.824
## Cumulative % of var. 59.097  75.768  85.525  93.365  97.176 100.000
##
## Individuals (the 10 first)
##          Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3
## 1        | 2.690 | 1.566  0.006  0.339 | -1.739  0.025  0.418 | -0.612
## 2        | 1.273 | -0.372  0.000  0.085 | -0.864  0.006  0.461 | -0.264
## 3        | 1.685 | -1.231  0.004  0.534 |  0.840  0.006  0.248 |  0.731
## 4        | 1.658 | -0.261  0.000  0.025 |  0.837  0.006  0.255 |  0.952
## 5        | 2.605 | -1.827  0.008  0.492 | -0.898  0.007  0.119 |  0.214
## 6        | 2.499 | 1.208  0.003  0.234 | -0.902  0.007  0.130 |  1.827
## 7        | 1.666 | -0.867  0.002  0.271 | -0.024  0.000  0.000 |  0.173
## 8        | 3.457 | 3.270  0.025  0.895 | -0.016  0.000  0.000 |  0.395
## 9        | 3.018 | 2.754  0.018  0.833 | -0.881  0.006  0.085 |  0.127
## 10       | 1.935 | 1.632  0.006  0.711 | -0.017  0.000  0.000 | -0.133
##          ctr   cos2

```

```

## 1      0.005  0.052 |
## 2      0.001  0.043 |
## 3      0.008  0.188 |
## 4      0.013  0.330 |
## 5      0.001  0.007 |
## 6      0.047  0.534 |
## 7      0.000  0.011 |
## 8      0.002  0.013 |
## 9      0.000  0.002 |
## 10     0.000  0.005 |
##
## Supplementary individuals (the 10 first)
##          Dist   Dim.1   cos2   Dim.2   cos2   Dim.3   cos2
## 12000    | 1.403 | -0.515  0.135 | 0.006  0.000 | -0.310  0.049 |
## 12001    | 1.325 | -0.638  0.232 | -0.028  0.000 |  0.647  0.239 |
## 12002    | 0.913 | -0.207  0.051 | -0.014  0.000 |  0.331  0.131 |
## 12003    | 2.435 | -2.004  0.678 | -0.017  0.000 | -0.388  0.025 |
## 12004    | 2.964 |  2.523  0.725 |  0.891  0.090 | -1.105  0.139 |
## 12005    | 1.726 | -1.112  0.415 | -0.853  0.244 | -0.672  0.152 |
## 12006    | 1.462 | -0.825  0.318 |  0.847  0.336 |  0.441  0.091 |
## 12007    | 2.831 |  2.742  0.938 | -0.003  0.000 | -0.158  0.003 |
## 12008    | 2.336 | -1.759  0.567 | -0.898  0.148 |  0.698  0.089 |
## 12009    | 3.771 |  3.465  0.844 |  0.855  0.051 |  1.005  0.071 |
##
## Variables
##          Dim.1   ctr   cos2   Dim.2   ctr   cos2   Dim.3   ctr
## Unit_Ticket | 0.793 17.726  0.629 | -0.010  0.009  0.000 |  0.544 50.632
## Fidelity    | 0.902 22.948  0.814 | -0.005  0.002  0.000 |  0.223  8.479
## BookTime    | 0.889 22.283  0.790 |  0.001  0.000  0.000 | -0.109  2.041
## Taxes       | 0.765 16.490  0.585 |  0.022  0.047  0.000 | -0.382 24.968
## Discount    | 0.854 20.553  0.729 | -0.001  0.000  0.000 | -0.285 13.845
## NCcancelYear| -0.004  0.000  0.000 |  1.000 99.942  1.000 |  0.014  0.034
##
##          cos2
## Unit_Ticket | 0.296 |
## Fidelity    | 0.050 |
## BookTime    | 0.012 |
## Taxes       | 0.146 |
## Discount    | 0.081 |
## NCcancelYear| 0.000 |
##
## Supplementary continuous variables
##          Dim.1   cos2   Dim.2   cos2   Dim.3   cos2
## NPax        | 0.558  0.311 | -0.002  0.000 |  0.198  0.039 |
## NBookMonth  | 0.001  0.000 | -0.483  0.233 | -0.018  0.000 |
## NAccWebWeek | -0.006  0.000 |  0.358  0.128 |  0.004  0.000 |
## NComplaintsYear | -0.009  0.000 |  0.357  0.128 |  0.001  0.000 |
## NRefundYear  | 0.003  0.000 |  0.017  0.000 |  0.007  0.000 |
##
## Supplementary categories
##          Dist   Dim.1   cos2   v.test   Dim.2   cos2   v.test
## London     | 1.429 |  1.409  0.973 76.366 | -0.008  0.000 -0.846 |
## Paris      | 0.730 | -0.567  0.605 -2.949 | -0.038  0.003 -0.376 |
## Madrid     | 0.796 | -0.715  0.807 -15.489 |  0.025  0.001  1.022 |
## Amsterdam  | 1.421 | -1.388  0.954 -66.715 |  0.003  0.000  0.247 |

```

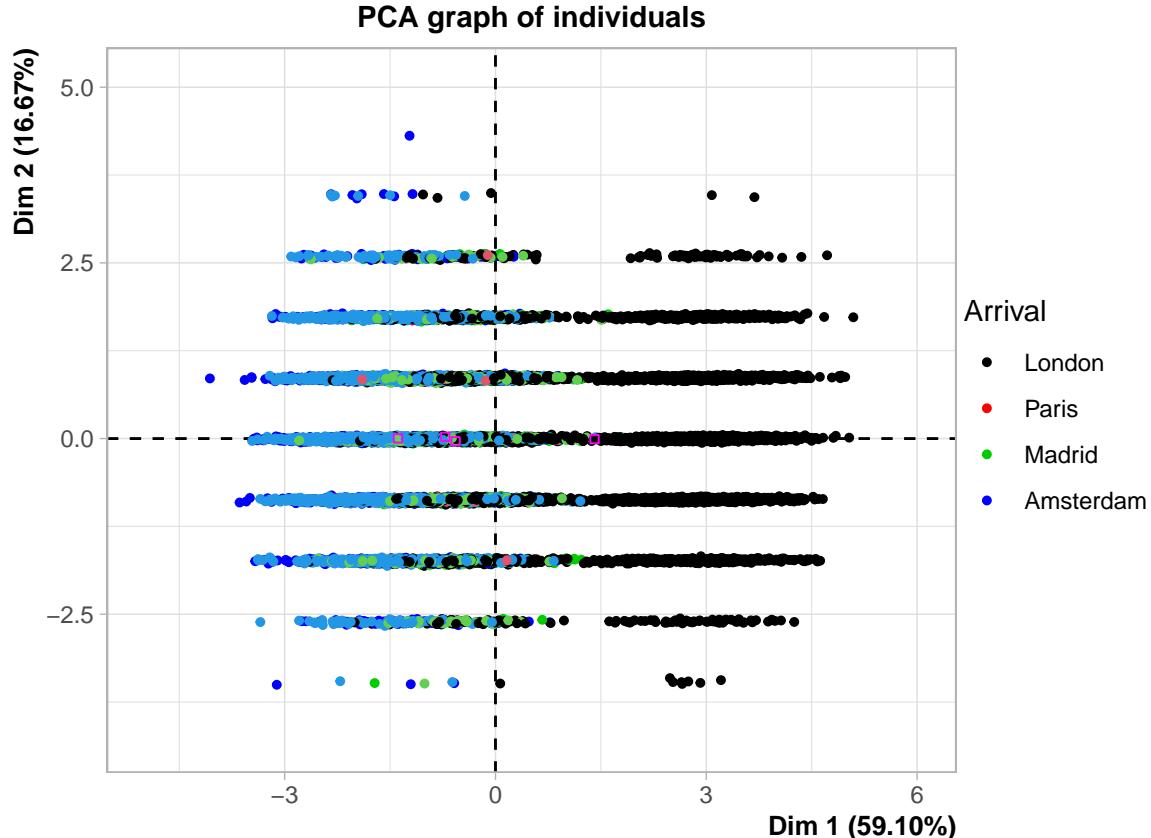
```

##          Dim.3    cos2 v.test
## London      -0.027  0.000 -3.548 |
## Paris       0.302  0.171  3.864 |
## Madrid      0.246  0.096 13.140 |
## Amsterdam   -0.049  0.001 -5.850 |

```

La PCA può essere utile per la segmentazione della clientela. Si osserva che il primo asse fattoriale è correlato positivamente con le variabili : Unit\_Tickets, Fidelity, BookTime, Taxes, Discount. Il secondo asse fattoriale è correlato positivamente con NCcancelYear. Inoltre sono state proiettate le variabili illustrate.

```
plot(res, cex=0.8, habillage="Arrival", label = "none")
```



Si osserva che i punti sul primo quadrante riguardano la tipologia di clienti con un alto indice di fidelizzazione per le quali prenotazioni si caratterizzano per il costo del biglietto elevato, per uno sconto alto sulla prenotazione, e per un alto numero di giorni di anticipo sulla prenotazione rispetto alla partenza.

## Suddivisione casuale del dataset in training set e test set

```

set.seed(123)
x=model.matrix(Cfidelity ~.,dati2) [,-1]

train = sample(nrow(x), nrow(x)*2/3)
test=(-train)
dati_test = dati2[-train, ] #creo il test set
dati_train = dati2[train,]

y=as.numeric(dati2$Cfidelity)
y.test=as.numeric(y[test])

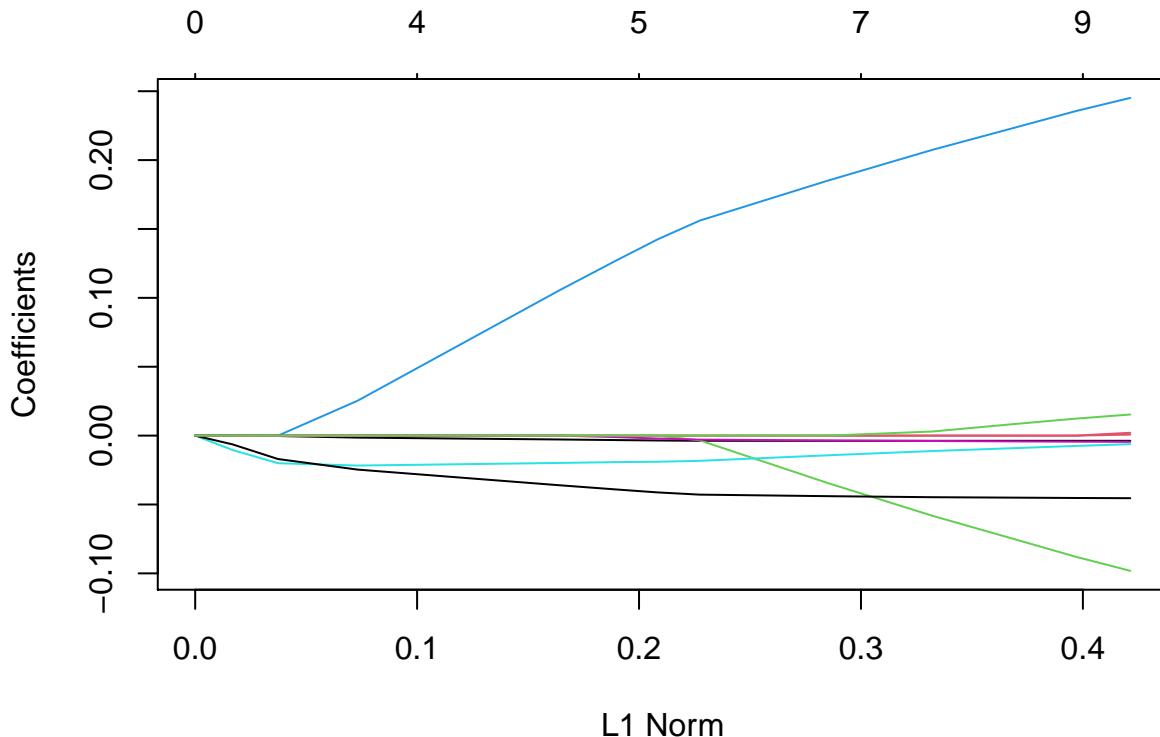
```

Il dataset è stato diviso casualmente per 2/3 in train e per 1/3 test set

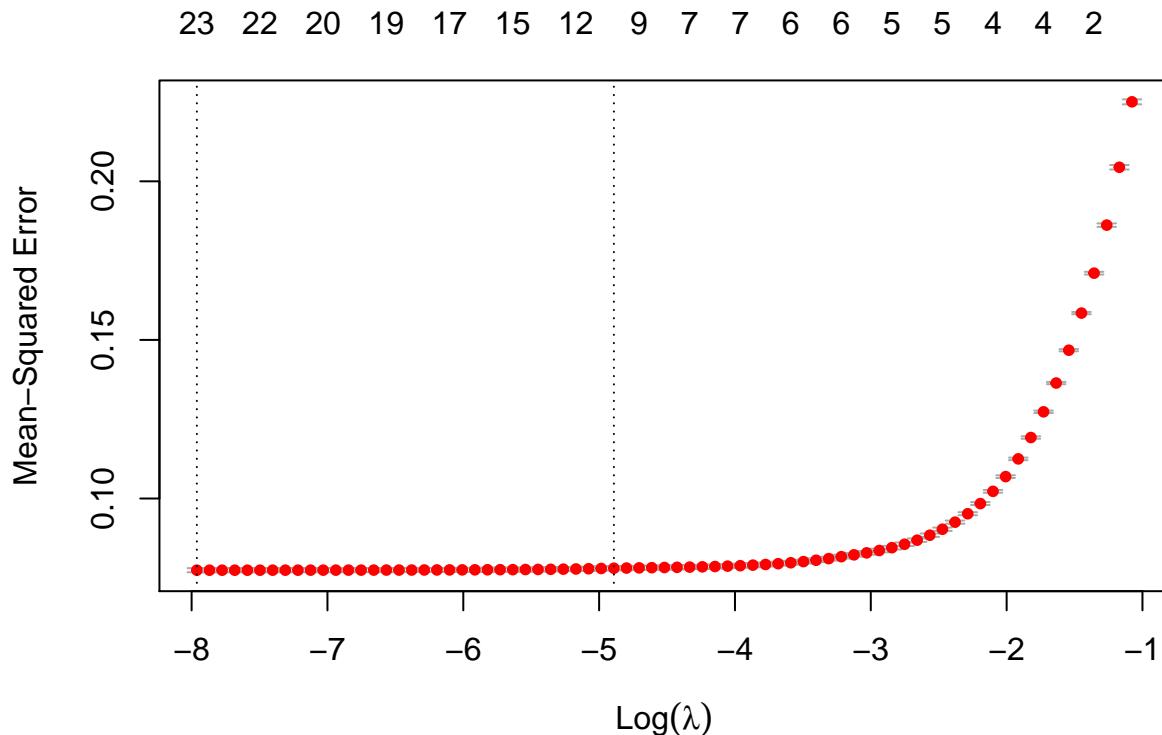
```
#lasso
library(glmnet)

## Loading required package: Matrix
##
## Attaching package: 'Matrix'
## The following objects are masked from 'package:tidyverse':
##      expand, pack, unpack
## Loaded glmnet 4.0-2
grid=10^seq(10,-2,length=100)
lasso.mod=glmnet(x[train ,],as.numeric(y[train]),alpha=1,lambda=grid)
plot(lasso.mod)

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



```
cv.out=cv.glmnet(x[train ,],as.numeric(y[train]),alpha=1)
plot(cv.out)
```



```
bestlam=cv.out$lambda.min
lasso.pred=predict(lasso.mod,s=bestlam ,newx=x[test,])
table(y.test, round(lasso.pred,0))
```

```
##
## y.test      1     2
##   1 1984    702
##   2    76 5238
```

778 malclassificati su 8000 -> 0.0971 -> 9,7%

```
out=glmnet(x,y,alpha=1,lambda=grid)
lasso.coef=predict(out,type="coefficients",s=bestlam)[1:19,]
lasso.coef
```

```
##      (Intercept)          Data       Unit_Ticket       SeatYes
## 2.573904903  0.000000000 -0.003758809  0.001333084
## PriorBoardYes LuggageYes      ReturnYes      BookTime
## 0.000000000 -0.100771968  0.246232407 -0.006281931
##      Taxes      Discount       NPax DepartureRome
## -0.004634191 -0.044389692  0.000000000  0.000000000
## DepartureNaples DeparturePalermo DepartureBergamo DepartureVenice
## 0.000000000  0.000000000  0.000000000  0.000000000
## ArrivalParis   ArrivalMadrid  ArrivalAmsterdam
## 0.000000000  0.010621467  0.000000000
```

```
lasso.coef[lasso.coef!=0]
```

```
##      (Intercept)  Unit_Ticket       SeatYes       LuggageYes      ReturnYes
## 2.573904903 -0.003758809  0.001333084 -0.100771968  0.246232407
##      BookTime      Taxes      Discount ArrivalMadrid
## -0.006281931 -0.004634191 -0.044389692  0.010621467
```

solamente 8 dei coefficienti stimati hanno valore diverso da 0. Per cui il lasso model scelto dalla cross-validation coniene solo 8 variabili

```
summary(lasso.mod)

##          Length Class     Mode
## a0           100  -none-   numeric
## beta        2400 dgCMatrix S4
## df            100  -none-   numeric
## dim             2  -none-   numeric
## lambda       100  -none-   numeric
## dev.ratio    100  -none-   numeric
## nulldev         1  -none-   numeric
## npasses         1  -none-   numeric
## jerr            1  -none-   numeric
## offset           1  -none- logical
## call             5  -none-  call
## nobs            1  -none-   numeric

#PCR
library(pls)

##
## Attaching package: 'pls'
## The following object is masked from 'package:stats':
## 
##      loadings

pcr_model <- pcr(as.numeric(Cfidelity) ~ ., data = dati2,
                  scale = TRUE,
                  validation = "CV")

summary(pcr_model)

## Data:    X dimension: 23998 24
## Y dimension: 23998 1
## Fit method: svdpc
## Number of components considered: 24
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##              (Intercept) 1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            0.4738   0.3038   0.3038   0.3038   0.3038   0.2961   0.2956
## adjCV         0.4738   0.3038   0.3038   0.3038   0.3038   0.2961   0.2956
##              7 comps  8 comps  9 comps 10 comps 11 comps 12 comps 13 comps
## CV            0.2925   0.2924   0.2921   0.2921   0.2919   0.2917   0.2916
## adjCV         0.2925   0.2924   0.2921   0.2921   0.2919   0.2916   0.2916
##              14 comps 15 comps 16 comps 17 comps 18 comps 19 comps 20 comps
## CV            0.2916   0.2916   0.2916   0.2913   0.2867   0.2866   0.2809
## adjCV         0.2916   0.2916   0.2916   0.2913   0.2867   0.2866   0.2809
##              21 comps 22 comps 23 comps 24 comps
## CV            0.28      0.28      0.2792   0.2792
## adjCV         0.28      0.28      0.2792   0.2792
##
## TRAINING: % variance explained
```

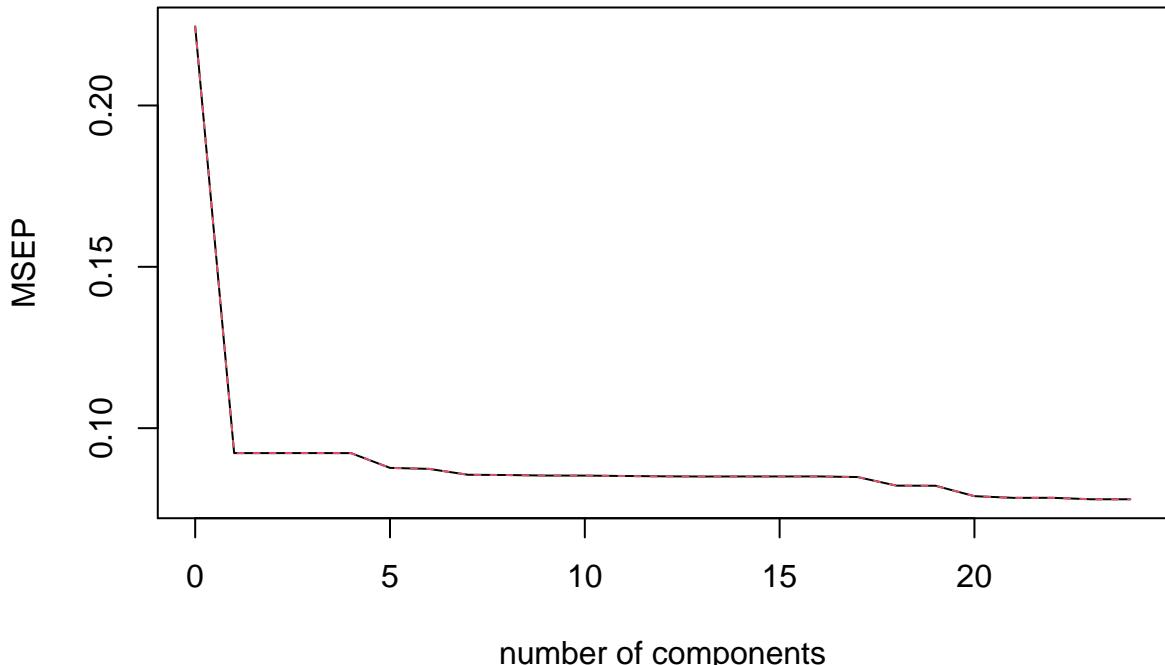
```

##                               1 comps   2 comps   3 comps   4 comps   5 comps   6 comps
## X                           22.04    28.77    35.01    40.94    46.53    51.43
## as.numeric(Cfidelity)      58.90    58.90    58.91    58.92    60.96    61.08
##                               7 comps   8 comps   9 comps   10 comps  11 comps  12 comps
## X                           56.24    60.79    65.09    69.34    73.55    77.69
## as.numeric(Cfidelity)      61.91    61.95    62.03    62.03    62.10    62.16
##                               13 comps  14 comps  15 comps  16 comps  17 comps
## X                           81.82    85.38    88.70    91.69    94.06
## as.numeric(Cfidelity)      62.17    62.17    62.17    62.17    62.26
##                               18 comps  19 comps  20 comps  21 comps  22 comps
## X                           95.85    97.45    98.55    99.15    99.56
## as.numeric(Cfidelity)      63.45    63.46    64.90    65.13    65.13
##                               23 comps  24 comps
## X                           99.81    100.00
## as.numeric(Cfidelity)      65.34    65.34

# Plot the cross validation MSE
validationplot(pcr_model, val.type="MSEP")

```

### as.numeric(Cfidelity)

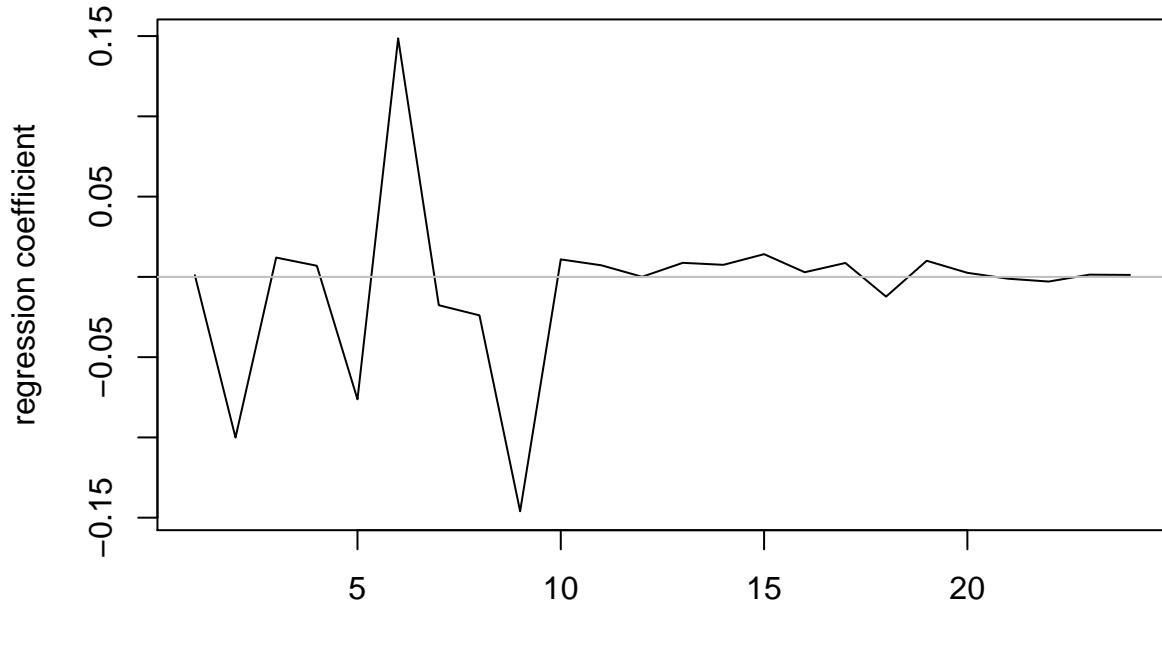


si appiattisce già per un numero di componenti principali pari a 5.

coefplot(pcr\_model) #il modello con 7 componenti principali ha un coefficiente di regressione negativo

Il MSE

## as.numeric(Cfidelity)



modello con 7 componenti principali ha un coefficiente di regressione positivo alto

```
pcr_model <- pcr(as.numeric(Cfidelity) ~ ., ncomp = 7,
                   data = dati2,
                   subset = train,
                   scale = TRUE,
                   validation = "CV")

summary(pcr_model)

## Data:    X dimension: 15998 24
##  Y dimension: 15998 1
## Fit method: svdpc
## Number of components considered: 7
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##              (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps
## CV          0.4746   0.3032   0.3032   0.3031   0.3031   0.2958   0.2953
## adjCV       0.4746   0.3032   0.3032   0.3031   0.3031   0.2958   0.2953
##             7 comps
## CV          0.2920
## adjCV       0.2919
##
## TRAINING: % variance explained
##                      1 comps 2 comps 3 comps 4 comps 5 comps 6 comps
## X                  22.09    28.86   35.09   40.99   46.54   51.45
## as.numeric(Cfidelity) 59.19    59.20   59.20   59.24   61.19   61.31
##                      7 comps
## X                  56.26
## as.numeric(Cfidelity) 62.18
```

I valori 59.19 59.20 59.20 59.24 61.19 61.31 62.18 rappresentano la variabilità percentuale spiegata per la variabile y

```
pcr.pred= predict(pcr_model ,x[test,] ,ncomp=7)
mean((pcr.pred-y.test)^2) # MSE 0.08625655
```

```
## [1] 0.08625655
```

```
table(as.numeric(dati_test$Cfidelity), round(pcr.pred,0)) #
```

```
##
```

```
##      1     2
## 1 1981 705
## 2    72 5242
```

777 casi su 8000 sono malclassificati, quindi il modello in un'ottica di previsione funziona bene con un errore di malclassificazione di  $794/8000 = 0.099 \rightarrow 9.9\%$

```
#PLS
```

```
pls_model <- plsr(as.numeric(Cfidelity)~., data = dati2 ,
subset = train,
scale = TRUE,
validation = "CV")
```

```
summary(pls_model)
```

```
## Data: X dimension: 15998 24
## Y dimension: 15998 1
## Fit method: kernelpls
## Number of components considered: 24
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##          (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps
## CV          0.4746  0.2967  0.2846  0.2796  0.2789  0.2785  0.2785
## adjCV       0.4746  0.2967  0.2845  0.2796  0.2789  0.2785  0.2784
##          7 comps 8 comps 9 comps 10 comps 11 comps 12 comps 13 comps
## CV          0.2784  0.2783  0.2783  0.2783  0.2783  0.2782  0.2782
## adjCV       0.2783  0.2783  0.2782  0.2782  0.2782  0.2782  0.2782
##          14 comps 15 comps 16 comps 17 comps 18 comps 19 comps 20 comps
## CV          0.2782  0.2782  0.2782  0.2782  0.2782  0.2782  0.2782
## adjCV       0.2782  0.2782  0.2782  0.2782  0.2782  0.2782  0.2782
##          21 comps 22 comps 23 comps 24 comps
## CV          0.2782  0.2782  0.2782  0.2782
## adjCV       0.2782  0.2782  0.2782  0.2782
##
## TRAINING: % variance explained
##          1 comps 2 comps 3 comps 4 comps 5 comps 6 comps
## X          22.04   27.03   29.35   33.60   36.67   41.46
## as.numeric(Cfidelity) 60.93   64.15   65.40   65.56   65.65   65.68
##          7 comps 8 comps 9 comps 10 comps 11 comps 12 comps
## X          44.03   47.06   50.97   54.49   56.55   60.54
## as.numeric(Cfidelity) 65.71   65.73   65.73   65.74   65.74   65.74
```

```

##                               13 comps   14 comps   15 comps   16 comps   17 comps
## X                           64.95     68.55    70.78     73.15    75.63
## as.numeric(Cfidelity)      65.74     65.74    65.74     65.74    65.74
##                               18 comps   19 comps   20 comps   21 comps   22 comps
## X                           79.14     80.57    83.70     87.47    91.60
## as.numeric(Cfidelity)      65.74     65.74    65.74     65.74    65.74
##                               23 comps   24 comps
## X                           95.79    100.00
## as.numeric(Cfidelity)      65.74     65.74

```

I valori 60.93 64.15 65.40 65.56 65.65 rappresentano la percentuale di variabilità spiegata per la variabile y. Rispetto al modello PCR risulta leggermente più alta.

```

lm2<-lm(as.numeric(dati_train$Cfidelity)~plsr_model$scores[,1:10])
summary(lm2)

```

```

##
## Call:
## lm(formula = as.numeric(dati_train$Cfidelity) ~ plsr_model$scores[,,
##       1:10])
##
## Residuals:
##      Min        1Q        Median        3Q        Max
## -1.10447 -0.09221  0.01968  0.16968  1.18481
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)               1.6575197  0.0021969 754.491 < 2e-16 ***
## plsr_model$scores[, 1:10]Comp 1  0.1619042  0.0009602 168.607 < 2e-16 ***
## plsr_model$scores[, 1:10]Comp 2  0.0841267  0.0021710  38.750 < 2e-16 ***
## plsr_model$scores[, 1:10]Comp 3  0.0790540  0.0032728  24.155 < 2e-16 ***
## plsr_model$scores[, 1:10]Comp 4  0.0219615  0.0024833  8.844 < 2e-16 ***
## plsr_model$scores[, 1:10]Comp 5  0.0208429  0.0032570  6.399 1.6e-10 ***
## plsr_model$scores[, 1:10]Comp 6  0.0087906  0.0023858  3.685 0.000230 ***
## plsr_model$scores[, 1:10]Comp 7  0.0154078  0.0040857  3.771 0.000163 ***
## plsr_model$scores[, 1:10]Comp 8  0.0084104  0.0029281  2.872 0.004080 **
## plsr_model$scores[, 1:10]Comp 9  0.0041540  0.0027964  1.485 0.137441
## plsr_model$scores[, 1:10]Comp 10 0.0034422  0.0029972  1.148 0.250791
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2779 on 15987 degrees of freedom
## Multiple R-squared:  0.6574, Adjusted R-squared:  0.6572
## F-statistic:  3067 on 10 and 15987 DF,  p-value: < 2.2e-16

```

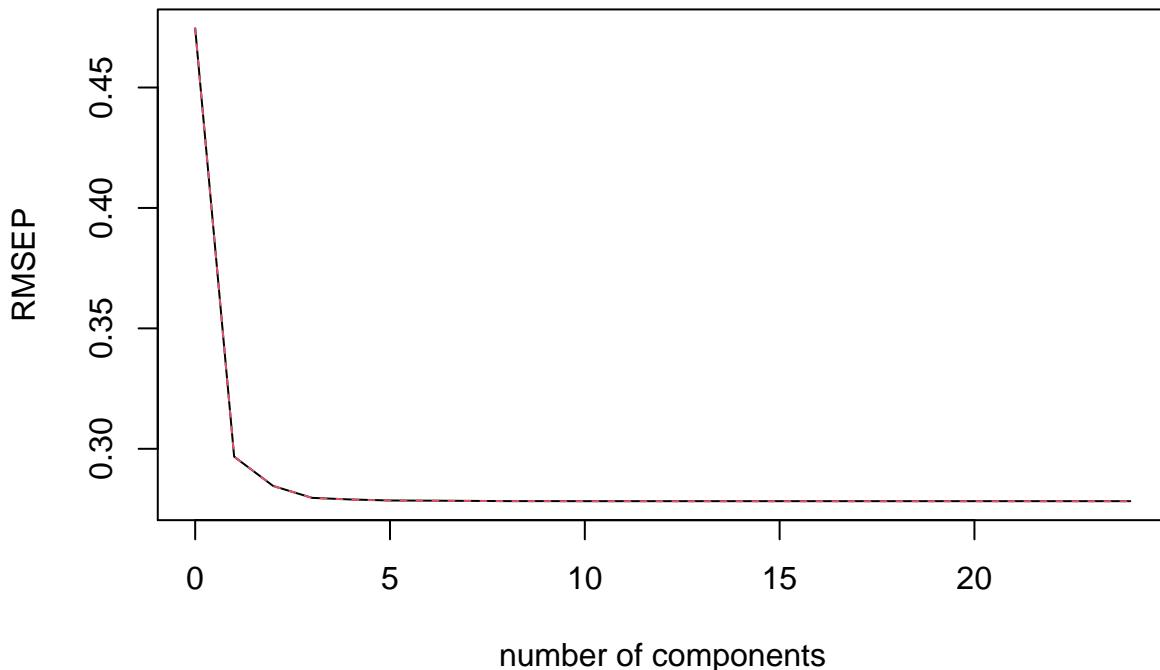
il valore dell' Adjusted R-squared: 0.657.

```

validationplot(plsr_model)

```

## as.numeric(Cfidelity)



```
# considerando solo le prime 7 componenti...
```

```
plsr_model <- plsr(as.numeric(Cfidelity)~., ncomp = 7,
                     data = dati2,
                     subset= train,
                     scale = TRUE,
                     validation = "CV")
```

```
summary(plsr_model)
```

```
## Data:      X dimension: 15998 24
##   Y dimension: 15998 1
## Fit method: kernelpls
## Number of components considered: 7
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##           (Intercept) 1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV          0.4746  0.2967  0.2846  0.2797  0.2789  0.2785  0.2785
## adjCV       0.4746  0.2967  0.2846  0.2797  0.2789  0.2785  0.2784
##           7 comps
## CV          0.2784
## adjCV       0.2784
##
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## X            22.04    27.03    29.35    33.60    36.67    41.46
## as.numeric(Cfidelity) 60.93    64.15    65.40    65.56    65.65    65.68
##           7 comps
```

```

## X          44.03
## as.numeric(Cfidelity)    65.71

```

la percentuale di varianza in fedeltà con 7 componenti 65.71. Risulta più alta della percentuale spiegata col modello finale dal modello PCR a parità di componenti principali (7)

```

lm2<-lm(as.numeric(dati_train$Cfidelity)~plsr_model$scores[,1:7])
summary(lm2)

```

```

##
## Call:
## lm(formula = as.numeric(dati_train$Cfidelity) ~ plsr_model$scores[,,
##       1:7])
##
## Residuals:
##      Min      1Q      Median      3Q      Max
## -1.111150 -0.09145  0.02034  0.16783  1.19204
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1.6575197  0.0021975 754.284 < 2e-16 ***
## plsr_model$scores[, 1:7]Comp 1 0.1619042  0.0009605 168.561 < 2e-16 ***
## plsr_model$scores[, 1:7]Comp 2 0.0841267  0.0021716 38.739 < 2e-16 ***
## plsr_model$scores[, 1:7]Comp 3 0.0790540  0.0032737 24.148 < 2e-16 ***
## plsr_model$scores[, 1:7]Comp 4 0.0219615  0.0024840  8.841 < 2e-16 ***
## plsr_model$scores[, 1:7]Comp 5 0.0208429  0.0032579  6.398 1.62e-10 ***
## plsr_model$scores[, 1:7]Comp 6 0.0087906  0.0023865  3.684 0.000231 ***
## plsr_model$scores[, 1:7]Comp 7 0.0154078  0.0040868  3.770 0.000164 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2779 on 15990 degrees of freedom
## Multiple R-squared:  0.6571, Adjusted R-squared:  0.657
## F-statistic:  4378 on 7 and 15990 DF,  p-value: < 2.2e-16

```

Il valore di Adjusted R-squared: 0.657 rimane quasi invariato ma questa volta sono state utilizzate solamente le prime 7 componenti principali

```

pls.pred=predict(plsr_model,x[test,],ncomp=7)

```

```

mean((pls.pred-y.test)^2) # MSE = 0.07939584

```

```

## [1] 0.07930442
table(as.numeric(dati_test$Cfidelity), round(pls.pred,0)) # 775 malclassificati su 8000. 775/8000= 0.0968
##
##           1     2
##     1 1986   700
##     2   75 5239

```

Il MSE = 0.07939584. Ci sono stati 775 malclassificati su 8000. 775/8000= 0.0968 -> 9,68%

## **Conclusioni**

Successivamente all'analisi esplorativa del dataset, si è proceduto alla creazione della variabile Cfidelity, ottenuta dalla variabile originale "Fidelity". In seguito, si è suddiviso il dataset casualmente per 2/3 in train e per 1/3 test set. Sono stati applicati 3 modelli sul training set con i quali si è prevista la variabile Cfidelity. Il modello Lasso ha classificato erroneamente 778 osservazioni sulle 8000 del test set. Il modello creato con PCR ha classificato erroneamente 777 osservazioni. Infine il modello PLS ha classificato erroneamente 775 osservazioni del test set.