Insegnamento di Analisi dei dati (Data mining)

Prova d'esame del 4 luglio 2016 - parte pratica

Daniele Cugnigni

2023-02-21

Testo d'esame

Nel dataset frodi.csv sono presenti 91557 transazioni finanziarie, effettuate attraverso carte di credito, relative a un periodo di un mese. Oltre ai dati della transazione, sono già stati calcolati alcuni indicatori che esperienze passate hanno mostrato essere utili per identificare eventuali frodi. In particolare sono disponibili degli indicatori di anomalia della transazione, relativamente all'importo speso e a precedenti modi d'uso della medesima carta. Il contenuto specifico di ciascuno di questi indicatori non è disponibile per motivi legati alla proprietà intellettuale degli stessi.

L'obiettivo dell'analisi consiste nello scoprire le operazioni fraudolente in relazione alle caratteristiche delle transazioni, in modo da prevedere le prime in funzione delle seconde.

Il dataset è composto dai seguenti campi:

- Id: Identificativo della transazione
- Id_carta: Identificativo della carta di credito.
- Importo: L'importo della transazione.
- 9 indicatori di anomalia sull'importo della transazione. Esempi: confronto con il mese precedente, confronto con il semestre precedente.
- 8 indicatori di anomalia comportamentale. Esempi: anomalia rispetto ai posti in cui la carta ha operato, anomalia rispetto alla frequenza delle transazioni.
- 8 indicatori di confronto della carta con le carte ad essa simili.
- frode: variabile indicatrice. Assume valore 1 per le transazioni fraudolente; 0 per le transazioni non fraudolente.

```
dati <- read.csv("frodi.csv", stringsAsFactors = TRUE)

str(dati)
```

```
## 'data.frame':    91557 obs. of  30 variables:
##  $ Id                 : int  196339525 196339531 196339546 196339729 196339906 196339898 196339937 19
##  $ Id_carta           : int  54309097 37160392 36562573 33066184 27058062 35516973 47880959 57518253
##  $ ora_GMT            : Factor w/ 28503 levels "2015-09-17 00:57:00",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Importo            : num  85 370 350 68 25 165 18 120 330 50 ...
##  $ Anomaly_importo1   : num  -0.353 -1 -0.434 0.103 -0.2 ...
##  $ Anomaly_importo2   : num  -0.576 -0.182 -0.577 0.272 5.26 ...
##  $ Anomaly_importo3   : num  -0.716 -0.309 -0.611 0.425 3.2 ...
##  $ Anomaly_importo4   : num  -0.754 -0.309 -0.704 0.314 3.188 ...
##  $ Anomaly_importo5   : num  -0.747 -0.359 -0.468 0.168 2.83 ...
```

```
##  $ Anomaly_importo6   : num  0.695 0.451 0.351 -0.525 -2.9 ...
##  $ Anomaly_importo7   : num  -0.294 -1 -0.531 0.535 0.2 ...
##  $ Anomaly_importo8   : num  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ Behaviour_Anomaly1 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Behaviour_Anomaly2 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Behaviour_Anomaly3 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Behaviour_Anomaly4 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Behaviour_Anomaly5 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Behaviour_Anomaly6 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Behaviour_Anomaly7 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Behaviour_Anomaly8 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Anomaly_importo9   : num  0.897 1.49 1.307 0.481 1.092 ...
##  $ Population_Anomaly1: num  -0.687 -1 -0.577 0.525 0.496 ...
##  $ Population_Anomaly2: num  3.79 -0.644 -0.624 0.938 4.27 ...
##  $ Population_Anomaly3: num  3.823 -0.646 -0.625 0.928 4.243 ...
##  $ Population_Anomaly4: num  3.671 -0.66 -0.641 0.849 4.03 ...
##  $ Population_Anomaly5: num  3.693 -0.669 -0.65 0.8 3.896 ...
##  $ Population_Anomaly6: num  3.738 -0.668 -0.65 0.804 3.907 ...
##  $ Population_Anomaly7: num  -0.314 0.429 -0.322 -0.304 0.267 ...
##  $ Population_Anomaly8: num  -0.2353 -0.0411 -0.4811 0.4471 0.16 ...
##  $ Frode              : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
dati$Frode <- as.factor(dati$Frode)
dim(dati)
```

```
## [1] 91557    30
```

```
#summary(dati)
length(unique(dati$Id))
```

```
## [1] 91557
```

Pulizia del dataset

Il file "frodi.csv" è composto da 91557 unità statistiche (le transazioni) sulle quali sono state rilevate complessivamente 30 variabili, con la variabile *Frode* che rappresenta la variabile risposta.
Prima di procedere all'analisi del dataset, è opportuno effettuare delle operazioni di pulizia. In primo luogo si nota la presenza delle variabili *Id* e *Id_carta*, le quali non sono altro che l'identificativo della transazione e l'identificativo della carta di credito e pertanto vengono escluse dall'analisi. Inoltre la variabile *ora_GMT* dà informazione su anno, mese, giorno ed ora della transazione: avendo il dataset composto da transazioni rilevate nell'arco di un mese, non è possibile valutare un eventuale effetto di anno, mese e giorno della transazione, mentre è possibile valutare l'effetto dell'ora. A tal proposito, sembra ragionevole focalizzare l'attenzione non tanto sull'ora esatta ma sul momento della giornata in cui avviene una transazione, di conseguenza si crea la variabile *momento* avente quattro modalità: *mattina* (6-12), *pomeriggio* (13-18), *sera* (19-23) e *notte* (0-5).

```
dati$Id <- NULL
dati$Id_carta <- NULL

#Estrapolazione dell'ora della transazione e trasformazione in "momento della giornata"
dati$ora <- as.POSIXlt(dati$ora_GMT)
```

```r
lista <- unclass(dati$ora)
dati$ora <- as.factor(lista$hour)
momento <- rep(0,nrow(dati))
momento[dati$ora %in% 6:12] <- "mattina"
momento[dati$ora %in% 13:18] <- "pomeriggio"
momento[dati$ora %in% 19:23] <- "sera"
momento[dati$ora %in% 0:5] <- "notte"
momento <- factor(momento)
dati$momento <- momento
dati$ora <- NULL
dati$ora_GMT = NULL
```

Successivamente si analizza l'eventuale presenza di valori mancanti nel dataset e si nota che il 24.74% delle osservazioni non ha un valore relativamente alla variabile *Anomaly_importo9*. Poichè si hanno a disposizione altri 8 indicatori di anomalia sull'importo delle transazioni, si decide di eliminare la variabile *Anomaly_importo9*.

```r
#Controllo della presenza di NA
na_get <- function(data){
  na_vars <- sapply(data, function(col) sum(is.na(col)))
  na_vars <- sort(na_vars[na_vars > 0])
  na_vars <- data.frame(
    variabile = names(na_vars),
    freq_assoluta = as.numeric(na_vars),
    freq_relativa = round(as.numeric(na_vars)/nrow(data), 4)
  )
  na_vars
}
na_tab <- na_get(dati)
na_tab
```

```
##          variabile freq_assoluta freq_relativa
## 1 Anomaly_importo9         22655        0.2474
```

```r
dati$Anomaly_importo9 <- NULL
```

In seguito a queste operazioni, il dataset è composto da 91557 unità statistiche e 27 variabili. A questo punto, prima di procedere con la modellazione dei dati:

- per tenere in considerazione il compromesso tra varianza e distorsione, si procede con la divisione del dataset in insieme di stima (75%) e insieme di verifica (25%), ottenendo un insieme di stima con 68668 osservazioni ed un insieme di verifica con 22889 osservazioni;
- si verifica se, nell'insieme di stima, le classi della variabile risposta siano bilanciate. A tal riguardo, si nota come le classi della variabile risposta risultano essere fortemente sbilanciate: 68582 osservazioni (99.87%) sono transazioni non fraudolente mentre 86 (0.13%) risultano essere fraudolente. Una possibile soluzione per tenere in considerazione questo aspetto è sottocampionare (senza reinserimento) le osservazioni che fanno riferimento ad operazioni non fraudolente. In questo caso, poichè il perfetto bilanciamento non è possibile in quanto porterebbe ad una perdita d'informazione troppo elevata, si decide di ricampionare 850 osservazioni relative ad operazioni fraudolente, in modo da avere un insieme di stima di 936 osservazioni e composto per il 10% da operazioni fraudolente e per il 90% da operazioni non fraudolente. E' importante far presente che questa scelta fa perdere molta dell'informazione a disposizione riguardo le operazioni non fraudolente ma, allo stesso tempo, permette di adattare i modelli

su un dataset con classi meno sbilanciate della variabile risposta, portando quindi ad una maggiore attenzione, in fase di stima, verso le operazioni fraudolente;
- si verifica l'assenza di variabili esplicative degeneri (e quindi inutili per l'analisi) nell'insieme di stima meno sbilanciato, rilevando l'assenza della modalità "1" nella variabile dicotomica *Behaviour_Anomaly8*, che pertanto viene eliminata.

```
#Divisione in training e test set
n <- dim(dati)[1]
p <- dim(dati)[2]

set.seed(789)

#Proporzione 3/5 stima e 2/5 verifica
ind <- sample(1:n, round((0.75)*n))
stima <- dati[ind, ]
ver <- dati[-ind, ]

#Divisione variabili quantitative e fattori
tipo_var <- sapply(stima, class)
table(tipo_var)
```

```
## tipo_var
##  factor integer numeric
##       2       8      17
```

```
var_qualitative <- names(stima)[tipo_var == "factor"]
var_quantitative <- setdiff(names(stima), var_qualitative)
var_qualitative
```

```
## [1] "Frode"   "momento"
```

```
var_quantitative
```

```
##  [1] "Importo"             "Anomaly_importo1"    "Anomaly_importo2"
##  [4] "Anomaly_importo3"    "Anomaly_importo4"    "Anomaly_importo5"
##  [7] "Anomaly_importo6"    "Anomaly_importo7"    "Anomaly_importo8"
## [10] "Behaviour_Anomaly1"  "Behaviour_Anomaly2"  "Behaviour_Anomaly3"
## [13] "Behaviour_Anomaly4"  "Behaviour_Anomaly5"  "Behaviour_Anomaly6"
## [16] "Behaviour_Anomaly7"  "Behaviour_Anomaly8"  "Population_Anomaly1"
## [19] "Population_Anomaly2" "Population_Anomaly3" "Population_Anomaly4"
## [22] "Population_Anomaly5" "Population_Anomaly6" "Population_Anomaly7"
## [25] "Population_Anomaly8"
```

```
table(stima$Frode)
```

```
##
##     0     1
## 68582    86
```

```r
prop.table(table(stima$Frode))
```

```
##
##           0           1
## 0.998747597 0.001252403
```

```r
#barplot(prop.table(table(stima$Frode)), xlab = "Frode", ylab = "Frequenza relativa",
#        main = "Distribuzione marginale della risposta", col = 3, names.arg = c("No", "Sì"),
#        ylim = c(0,1))

set.seed(1)
#id_one <- sample(which(stima$Frode == 1), size = 150, replace = T) #sovracampiono 1
id_one <- which(stima$Frode == 1)
id_zer <- sample(which(stima$Frode == 0), size = 850, replace = F) #sottocampiono 0


stima.bal <- stima[c(id_one, id_zer),]
prop.table(table(stima.bal$Frode))
```

```
##
##          0          1
## 0.90811966 0.09188034
```

```r
#Divisione variabili quantitative e fattori
tipo_var <- sapply(stima.bal, class)
table(tipo_var)
```

```
## tipo_var
##  factor integer numeric
##       2       8      17
```

```r
var_qualitative <- names(stima.bal)[tipo_var == "factor"]
var_quantitative <- setdiff(names(stima.bal), var_qualitative)
var_qualitative
```

```
## [1] "Frode"   "momento"
```

```r
var_quantitative
```

```
##  [1] "Importo"             "Anomaly_importo1"    "Anomaly_importo2"
##  [4] "Anomaly_importo3"    "Anomaly_importo4"    "Anomaly_importo5"
##  [7] "Anomaly_importo6"    "Anomaly_importo7"    "Anomaly_importo8"
## [10] "Behaviour_Anomaly1"  "Behaviour_Anomaly2"  "Behaviour_Anomaly3"
## [13] "Behaviour_Anomaly4"  "Behaviour_Anomaly5"  "Behaviour_Anomaly6"
## [16] "Behaviour_Anomaly7"  "Behaviour_Anomaly8"  "Population_Anomaly1"
## [19] "Population_Anomaly2" "Population_Anomaly3" "Population_Anomaly4"
## [22] "Population_Anomaly5" "Population_Anomaly6" "Population_Anomaly7"
## [25] "Population_Anomaly8"
```

```r
#Rimozione delle variabili quantitative degeneri
ids.deg <- which(apply(stima.bal, 2, var) == 0)
ids.deg
```

```
## Behaviour_Anomaly8
##                 17
```

```r
table(stima.bal$Behaviour_Anomaly8)
```

```
##
##   0
## 936
```

```r
stima.bal[,names(ids.deg)] <- NULL
ver[,names(ids.deg)] <- NULL

#Rimozione delle variabili qualitative degeneri
for(col in var_qualitative) cat(col,":", nlevels(stima.bal[,col]), "livelli \n")
```

```
## Frode : 2 livelli
## momento : 4 livelli
```

```r
tipo_var <- sapply(stima.bal, class)
table(tipo_var)
```

```
## tipo_var
##  factor integer numeric
##       2       7      17
```

```r
var_qualitative <- names(stima.bal)[tipo_var == "factor"]
var_quantitative <- setdiff(names(stima.bal), var_qualitative)

const <- apply(stima.bal[,var_quantitative], 2, function(x) length(unique(x)) < 4)
#summary(dati[,var_quantitative][,const])
for(col in names(which(const == T))) {
  stima.bal[,var_quantitative][,col] <- as.factor(stima.bal[,var_quantitative][,col])
  ver[,var_quantitative][,col] <- as.factor(ver[,var_quantitative][,col])
}

#Salvo l'indice della risposta
ids.leak <- which(names(stima.bal) %in% "Frode")
tipo_var <- sapply(stima.bal[, -ids.leak], class)
table(tipo_var)
```

```
## tipo_var
##  factor numeric
##       8      17
```

```r
var_qualitative <- names(stima.bal)[-ids.leak][tipo_var == "factor"]
for(col in var_qualitative) cat(col,":", nlevels(stima.bal[,col]), "livelli \n")
```

```
## Behaviour_Anomaly1 : 2 livelli
## Behaviour_Anomaly2 : 2 livelli
## Behaviour_Anomaly3 : 2 livelli
## Behaviour_Anomaly4 : 2 livelli
## Behaviour_Anomaly5 : 2 livelli
## Behaviour_Anomaly6 : 2 livelli
## Behaviour_Anomaly7 : 2 livelli
## momento : 4 livelli
```

```r
var_quantitative <- setdiff(names(stima.bal)[-ids.leak], var_qualitative)



#Controllo che in verifica non ci siano modalità non presenti in stima
ind.lev = c()
for(col in var_qualitative){
  if(!(all(unique(ver[,col]) %in% unique(stima.bal[,col]))))
  {
    ind.lev = c(ind.lev, col)
    cat(col,"-> in verifica ci sono modalità non presenti in stima.bal\n")
  }
}
for(i in ind.lev){
  cat("Livelli ", i, ": stima = ", sort(unique(stima.bal[,i])),
      " verifica = ", sort(unique(ver[,i])), "\n")
}
```

In seguito a queste operazioni, l'insieme di stima è composto da 936 osservazioni e 26 variabili, mentre l'insieme di verifica è composto da 22889 osservazioni e il medesimo numero di variabili.
Concluse le operazioni di pulizia del dataset, si può procedere con l'analisi esplorativa sull'insieme di stima.

Analisi esplorativa

Tenendo in considerazione che la variabile risposta è una variabile categoriale con due modalità e le variabili esplicative risultano essere in parte quantitative e in parte qualitative, un'analisi esplorativa (abbastanza) completa ed adeguata si avrebbe con l'analisi della distribuzione della variabile dipendente al variare delle singole variabili indipendenti. Poichè l'obiettivo primario non è quello di effettuare l'analisi esplorativa ma di adattare i modelli, si valuta la distribuzione della risposta solamente per alcune variabili esogene.

```r
par(mfrow = c(2,3))
#Momento della giornata
condizionata <- prop.table(table(stima.bal$momento, stima.bal$Frode),1)
condizionata <- rbind(condizionata[1,], condizionata[3,], condizionata[4,], condizionata[2,])
rownames(condizionata) <- c("mattina", "pomeriggio", "sera", "notte")
barplot(t(condizionata),beside = T, xlab = "Momento della giornata", ylab = "Frode",
        ylim = c(0,1.15), col = c(3,4),legend.text = c("No", "Sì"), cex.axis = 1.1,
        cex.names = 0.9, cex.lab = 1.2, xlim = c(0,13))
```

```r
#Importo

classi <- cut(stima.bal[,"Importo"], breaks = round(summary(stima.bal[,"Importo"])[-4],2),
              include.lowest = T, dig.lab = 4)
#classi
condizionata <- prop.table(table(classi, stima.bal$Frode),1)
barplot(t(condizionata),beside = T, xlab = "Importo della transazione", ylab = "Frode",
        ylim = c(0,1.15), col = c(3,4),legend.text = c("No", "Si"), cex.axis = 1.1,
        cex.names = 0.9, cex.lab = 1.2)

#Behaviour_Anomaly1

condizionata <- prop.table(table(stima.bal$Behaviour_Anomaly1, stima.bal$Frode),1)
#condizionata
barplot(t(condizionata),beside = T, xlab = "Behaviour_Anomaly1", ylab = "Frode",
        ylim = c(0,1.15), col = c(3,4),legend.text = c("No", "Si"), cex.axis = 1.1,
        cex.names = 0.9, cex.lab = 1.2)

#Behaviour_Anomaly7

condizionata <- prop.table(table(stima.bal$Behaviour_Anomaly7, stima.bal$Frode),1)
#condizionata
barplot(t(condizionata),beside = T, xlab = "Behaviour_Anomaly7", ylab = "Frode",
        ylim = c(0,1.15), col = c(3,4),legend.text = c("No", "Si"), cex.axis = 1.1,
        cex.names = 0.9, cex.lab = 1.2)

#Population_Anomaly1

classi <- cut(stima.bal[,"Population_Anomaly1"],
              breaks = round(summary(stima.bal[,"Population_Anomaly1"])[-4],2),
              include.lowest = T, dig.lab = 4)
condizionata <- prop.table(table(classi, stima.bal$Frode),1)
#condizionata
barplot(t(condizionata),beside = T, xlab = "Population_Anomaly1", ylab = "Frode",
        ylim = c(0,1.15), col = c(3,4),legend.text = c("No", "Si"), cex.axis = 1.1,
        cex.names = 0.8, cex.lab = 1.2, xlim = c(0,13))

#Population_Anomaly7

classi <- cut(stima.bal[,"Population_Anomaly7"],
              breaks = round(summary(stima.bal[,"Population_Anomaly7"])[-4],2),
              include.lowest = T, dig.lab = 4)
condizionata <- prop.table(table(classi, stima.bal$Frode),1)
#condizionata
barplot(t(condizionata),beside = T, xlab = "Population_Anomaly7", ylab = "Frode",
        ylim = c(0,1.15), col = c(3,4), cex.axis = 1.1, legend.text = c("No", "Si"),
        cex.names = 0.8, cex.lab = 1.2, xlim = c(0,13))
```

I barplot in Figura 1 danno indicazione di un possibile effetto significativo del momento della giornata in cui avviene la transazione (*momento*), dell'importo della transazione (*Importo*), del primo indicatore dell'anomalia comportamentale (*Behaviour_Anomaly1*) e del primo (*Population_Anomaly1*) e settimo (*Population_Anomaly7*) indicatore di confronto della carta con le carte ad essa simili. In particolare, si nota che la mattina e il pomeriggio vengono quasi esclusivamente compiute transazioni non fraudolente,

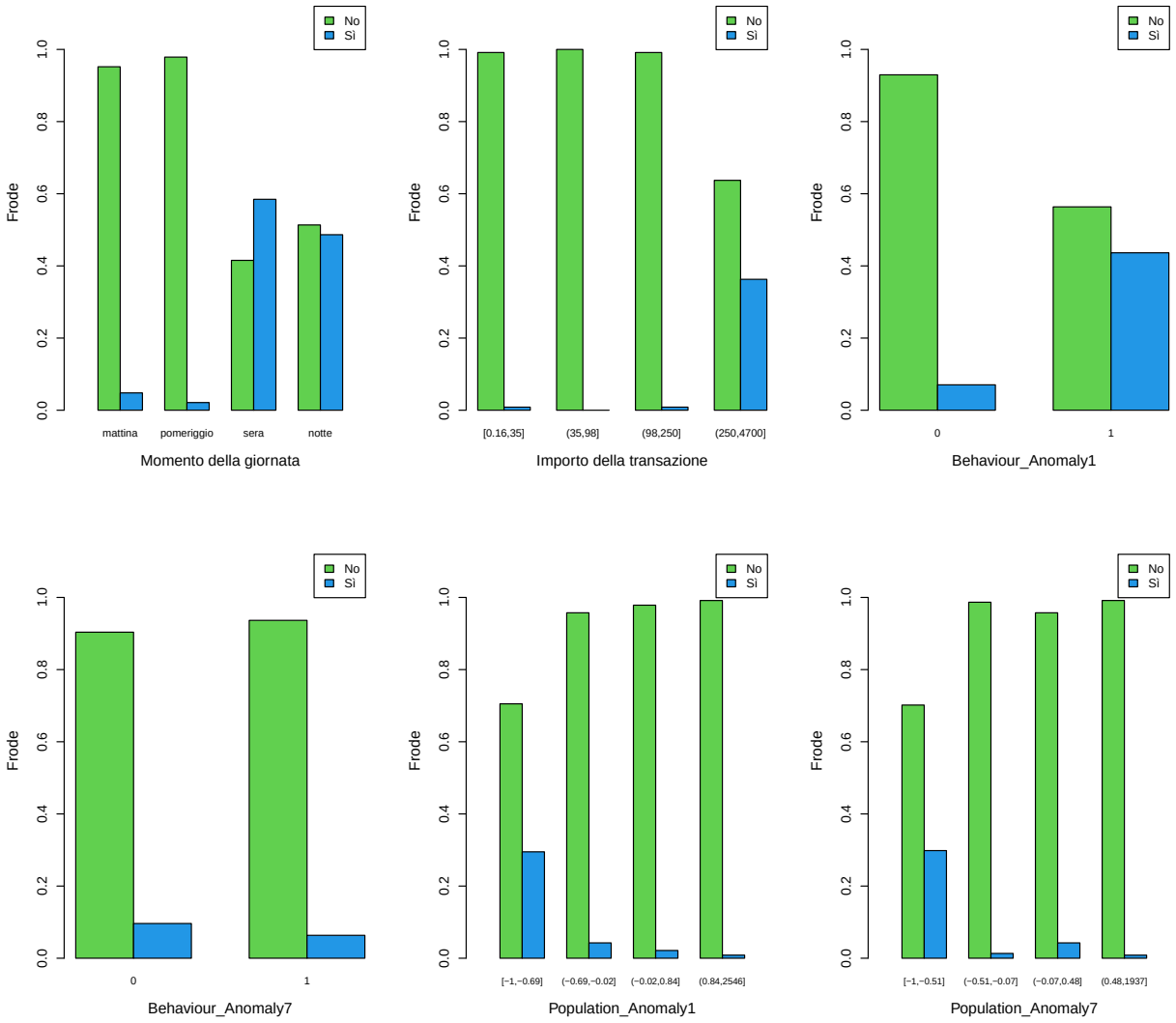

Figure 1: Barplot della variabile risposta rispetto ad alcune variabili esplicative

mentre la sera e la notte la percentuale di operazioni fraudolente e non fraudolente è praticamente la stessa. Per quanto riguarda l'importo della transazione, emerge la quasi assenza di operazioni fraudolente per importi inferiori a €250. Infine, si nota un andamento decrescente della proporzione di operazioni fraudolente all'aumentare dei valori assunti dal primo e settimo indicatore di confronto della carta con le altre carte.

Conclusa l'analisi esplorativa nell'insieme di stima, si può procedere alla modellazione dei dati.

Modellazione dei dati

In questo contesto, è importante tenere conto del differente peso degli errori di previsione, in quanto è molto più grave prevedere come non fraudolenta un'operazione che lo è piuttosto che prevedere come fraudolenta un'operazione che non lo è. In altri termini, è più importante minimizzare il numero di falsi negativi rispetto al numero di falsi positivi, pertanto si fissa il valore della soglia pari a 0.10, ovvero la proporzione di operazioni fraudolente presenti nell'insieme di stima, e si valuterà la performance dei modelli sia in termini di tasso di errata classificazione sia in termini di percentuale di falsi negativi.

```
#Formula del modello completo ()
nomi <- names(stima.bal)
form <- as.formula(paste("Frode ~ ", paste(nomi[-ids.leak],collapse ="+")))

#Funzione che calcola matrice di confusione, tasso di errata classificazione, falsi positivi e falsi ne
tabella.sommario <- function(previsti, osservati){
  n <-  table(previsti,osservati)
  err.tot <- 1-sum(diag(n))/sum(n)
  fn <- n[1,2]/(n[1,2]+n[2,2])
  fp <- n[2,1]/(n[1,1]+n[2,1])
  print(n)
  cat("errore totale: ", format(err.tot),"\n")
  cat("falsi positivi & falsi negativi: ",format(c(fp, fn)),"\n")
  invisible(n)
}

s <- prop.table(table(stima.bal$Frode))[2]
tab <- list()
```

Modello logistico stepwise

Il primo modello che si adatta è il modello di regressione logistica stepwise basato sulla minimizzazione dell'AIC, con ricerca in entrambe le direzioni e a partire dal modello con la sola intercetta.

```
mlog1 <- glm(Frode ~ 1, weights = NULL, data = stima.bal, family = binomial)
mlog2 <- step(mlog1, scope = form, direction = "both", trace = F)
summary(mlog2)
```

```
##
## Call:
## glm(formula = Frode ~ Importo + Anomaly_importo2 + Anomaly_importo7 +
##     momento + Behaviour_Anomaly1 + Anomaly_importo5 + Population_Anomaly6 +
##     Population_Anomaly4 + Population_Anomaly3 + Anomaly_importo8 +
##     Anomaly_importo6 + Population_Anomaly5 + Behaviour_Anomaly7 +
##     Behaviour_Anomaly3, family = binomial, data = stima.bal,
```

```
##     weights = NULL)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.992   0.000   0.000   0.000   2.248
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -7.123e+01  2.260e+01  -3.152 0.001620 **
## Importo              1.023e-03  7.669e-04   1.334 0.182089
## Anomaly_importo2    -3.877e+00  2.877e+00  -1.348 0.177763
## Anomaly_importo7    -9.630e+00  4.860e+00  -1.981 0.047551 *
## momentonotte         8.196e+00  3.287e+00   2.494 0.012647 *
## momentopomeriggio    2.692e+00  1.384e+00   1.945 0.051753 .
## momentosera          8.133e+00  2.259e+00   3.600 0.000319 ***
## Behaviour_Anomaly11 -4.155e+00  2.002e+00  -2.075 0.037960 *
## Anomaly_importo5    -1.411e+01  5.375e+00  -2.625 0.008653 **
## Population_Anomaly6  3.089e+03  1.122e+03   2.752 0.005930 **
## Population_Anomaly4 -1.083e+03  3.454e+02  -3.137 0.001707 **
## Population_Anomaly3  7.948e+02  2.825e+02   2.814 0.004900 **
## Anomaly_importo8     4.459e+00  2.200e+00   2.027 0.042641 *
## Anomaly_importo6    -1.210e+01  4.870e+00  -2.484 0.012974 *
## Population_Anomaly5 -2.860e+03  1.102e+03  -2.595 0.009462 **
## Behaviour_Anomaly71 -1.093e+01  4.537e+00  -2.410 0.015971 *
## Behaviour_Anomaly31  8.682e+00  4.136e+00   2.099 0.035787 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 574.455  on 935  degrees of freedom
## Residual deviance:  34.478  on 919  degrees of freedom
## AIC: 68.478
##
## Number of Fisher Scoring iterations: 17
```

```
logist.step.var <- names(mlog2$model)[-1]
length(logist.step.var)
```

```
## [1] 14
```

```
mlog2.pred <- predict(mlog2, newdata = ver, type = "response")
mlog2.tab <- tabella.sommario(mlog2.pred > s, ver$Frode)
```

```
##         osservati
## previsti     0     1
##    FALSE 22404     0
##    TRUE    457    28
## errore totale:  0.01996592
## falsi positivi & falsi negativi:  0.01999038 0.00000000
```

```
tab <- c(tab, list(Logistico.stepwise = mlog2.tab))
names(tab)[1] <- "Logistico stepwise"
```

Nel modello finale sono incluse 14 delle 26 variabili esplicative: l'importo della transazione, il momento della giornata in cui avviene la transazione, cinque indicatori relativi all'anomalia dell'importo, tre indicatori relativi all'anomalia comportamentale e quattro indicatori di confronto della carta con carte ad essa simili. Ad un livello di significatività del 10%, gli effetti dell'importo della transazione e del secondo indicatore di anomalia sull'importo della transazione risultano non essere significativi.
Il tasso di errata classificazione e la percentuale di falsi negativi nell'insieme di verifica sono pari rispettivamente al **2.00%** e allo **0.00%**.

Albero di classificazione

Si prosegue la fase di modellazione con l'adattamento di un albero di classificazione, con l'entropia come funzione da minimizzare. Poichè questo modello prevede la selezione del numero di foglie ottimale, si fa crescere l'albero nell'insieme di stima e si effettua la fase di potatura tramite convalida incrociata con 5 fold. Nella fase di crescita dell'albero viene impostata una numerosità minima di osservazioni per foglia pari a 2 e una diminuzione dell'entropia per consentire uno split pari a 0.0000005, in modo da far diventare l'albero il più profondo possibile. Nella fase di potatura viene valutata la devianza in convalida incrociata al variare del numero di foglie dell'albero.
Il grafico in Figura 2 mostra come il minimo si ottenga con un albero con 11 foglie.

```
library(tree)
set.seed(12)
m.tree.cv <- tree(Frode ~., weights = NULL, data = stima.bal,
                  control = tree.control(nobs =nrow(stima.bal), minsize = 2,
                                         mindev = 0.0000005))
set.seed(23)
m.tree.prune.cv <- cv.tree(m.tree.cv, FUN = prune.tree,
                           K = 5, method = "deviance")
plot(m.tree.prune.cv)
j = m.tree.prune.cv$size[which.min(m.tree.prune.cv$dev)]
abline(v = j, col = 2, lty = "dashed")
```

Uno dei pregi di questo modello è la facile interpretabilità nel caso in cui l'albero sia poco profondo. A tal riguardo, il grafico in Figura 3 mostra che l'importo, il momento della giornata e il quinto indicatore di anomalia sull'importo della transazione sono le variabili esplicative che determinano i primi split. E' comunque importante ricordare che con questo modello la misura di importanza delle variabili è condizionata alle suddivisioni effettuate in precedenza.

```
m.tree.best.cv <- prune.tree(m.tree.cv, best = j)
plot(m.tree.best.cv, type = "uniform")
text(m.tree.best.cv, cex = 0.6, pretty = 5)
```

```
p.tree.cv <- predict(m.tree.best.cv, newdata = ver,type = "vector")[,2]

tree.tab <- tabella.sommario(p.tree.cv > s, ver$Frode)
```

```
##         osservati
## previsti     0     1
```

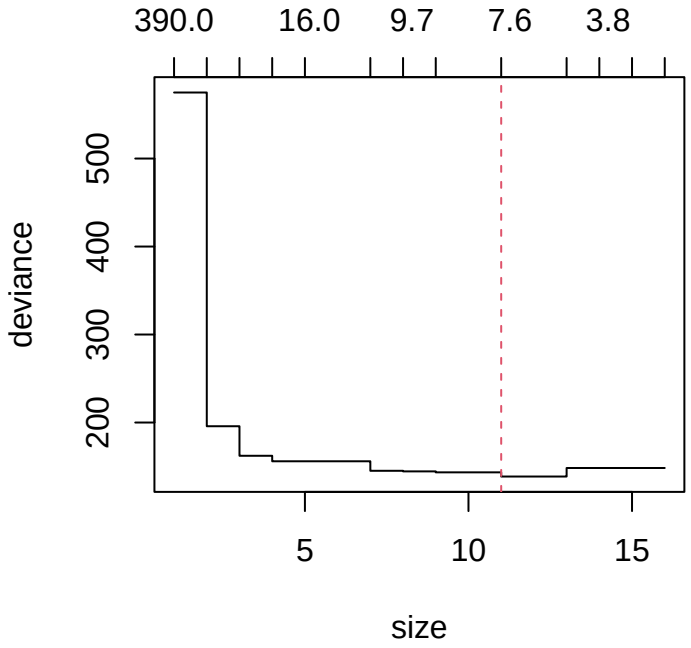

Figure 2: Errore in convalida incrociata in funzione del numero di foglie

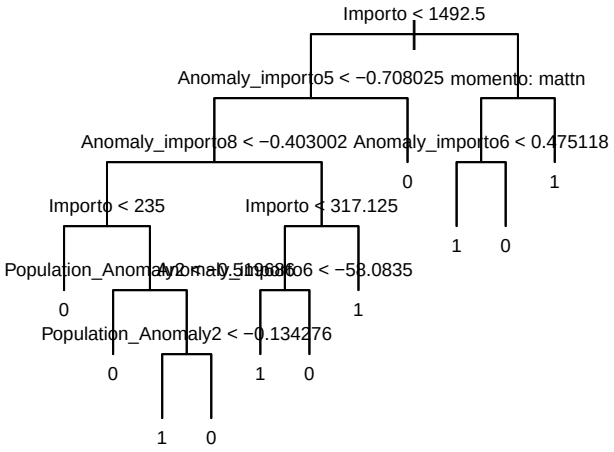

Figure 3: Albero di classificazione selezionato

```
##    FALSE 22359     3
##    TRUE    502    25
## errore totale:  0.022063
## falsi positivi & falsi negativi:  0.02195879 0.10714286
```

```
tab <- c(tab, list(Albero = tree.tab))
```

Nell'insieme di verifica il tasso di errata classificazione e la percentuale di falsi negativi risultano essere pari rispettivamente al **2.21%** e al **10.71%**.

Analisi discriminante lineare

Si adatta un modello di analisi discriminante lineare, utilizzando per la stima tutte le variabili esplicative disponibili. Il modello ottiene, nell'insieme di verifica, un tasso di errata classificazione pari all' **1.00%** e una percentuale di falsi negativi pari al **10.71%**.

```
m.lda <- MASS::lda(form, data = stima.bal)

p.lda <- predict(m.lda, newdata = ver)$posterior[,2]
lda.tab <- tabella.sommario(p.lda > s, ver$Frode)
```

```
##         osservati
## previsti     0     1
##    FALSE 22636     3
##    TRUE    225    25
## errore totale:  0.009961117
## falsi positivi & falsi negativi:  0.009842089 0.107142857
```

```
tab <- c(tab, list(lda = lda.tab))
names(tab)[3] <- "Analisi discriminante lineare"
```

Data la presenza di variabili qualitative, non risulta particolarmente sensato stimare un modello di analisi discriminante quadratica, in quanto quest'ultima si appoggia sull'ipotesi di normalità delle covariate.

Random forest

Si procede con l'adattamento del *random forest*. Il parametro di regolazione del modello è il numero di covariate da considerare ad ogni suddivisione dell'albero. A tal riguardo, l'insieme di stima viene diviso in un insieme di stima ridotto e uno di convalida e viene adattato il *random forest* con 500 alberi in corrispondenza di ognuno dei possibili valori del numero di covariate considerate. Il numero di covariate selezionato è il valore corrispondente al modello con tasso di errata classificazione minore nell'insieme di convalida. La Figura 4 mostra che, con tale procedura, si sceglie un numero di colonne da campionare per ogni albero pari a 4.

```
set.seed(589)
ind <- sample(1:nrow(stima.bal),
              3/4*nrow(stima.bal))
stima.rid <- stima.bal[ind,]
conv <- stima.bal[-ind,]
```

```
library(randomForest)
mtries <- 1:(ncol(stima.rid)-1)
err <- rep(NA, length(mtries))
set.seed(123)
for(i in 1:length(mtries)){
  rf <- randomForest(x = stima.rid[, -ids.leak], y = stima.rid$Frode,
                     xtest = conv[, -ids.leak], ytest = conv$Frode,
                     ntree = 500, mtry = mtries[i],
                     nodesize = 2, weights = NULL)
  err[i] <- rf$test$err.rate[500,1]
  cat(i, "")
}
```

```
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

```
plot(mtries, err, type = "l", xlab = "Numero di covariate campionate",
     ylab = "Tasso di errata classificazione", main = "")
mtry.opt <- mtries[which.min(err)]
abline(v = mtry.opt, col = 2, lty = "dashed")
```

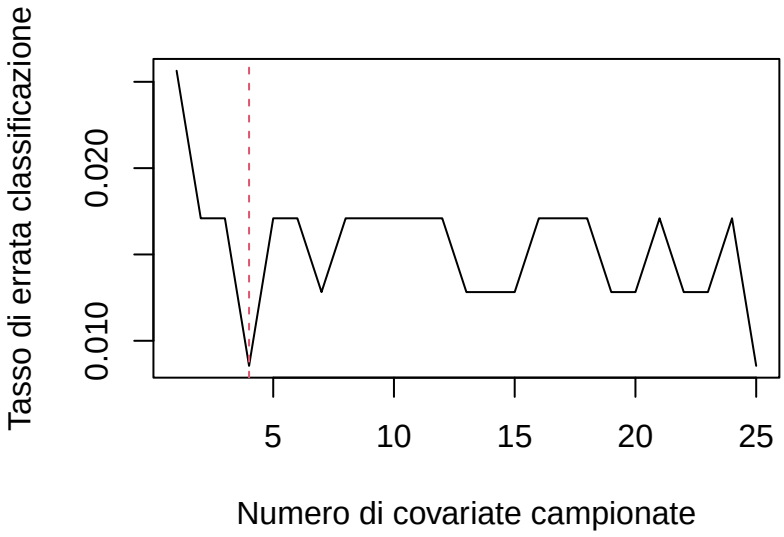

Figure 4: Errore nell'insieme di convalida in funzione del numero di covariate campionate

Successivamente il modello selezionato (4 variabili esplicative considerate in ogni albero) è adattato sull'intero insieme di stima e permette di ottenere un tasso di errata classificazione e una percentuale di falsi negativi nell'insieme di verifica pari rispettivamente al **3.36%** e allo **0.00%**.
Questo modello permette di ottenere una misura di importanza delle variabili esplicative, senza però avere indicazione sulla direzione dell'effetto di esse sulla risposta. In queso caso, la Figura 5 mette in luce che le variabili più importanti in termini di diminuzione dell'errore di previsione risultano essere l'importo della transazione, il secondo, il terzo, il quarto, il quinto e il sesto indicatore di confronto della carta con carte ad essa simili.

```
set.seed(2222)
rf <- randomForest(x = stima.bal[, -ids.leak], y = stima.bal$Frode, ntree = 500,
                   mtry = mtry.opt, importance = TRUE, weights = NULL)
rf.pred.prob <- predict(rf, newdata = ver, type = "prob")[,2]
rf.tab <- tabella.sommario(rf.pred.prob > s, ver$Frode)
```

```
##         osservati
## previsti     0     1
##    FALSE 22024     0
##    TRUE    837    28
## errore totale:  0.03656778
## falsi positivi & falsi negativi:  0.03661257 0.00000000
```

```
tab <- c(tab, list(randomforest = rf.tab))
names(tab)[4] <- "Random Forest"
varImpPlot(rf, type = "1", main = "", n.var = 15, cex = 0.8)
```

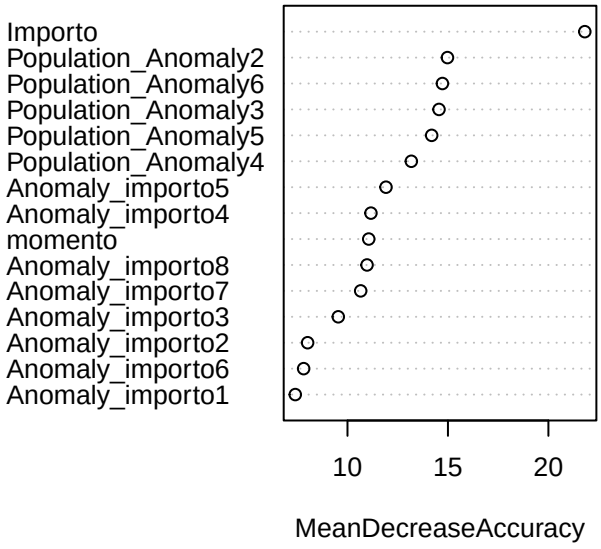

Figure 5: Importanza delle variabili nel random forest

Bagging

Si adatta un *bagging* con alberi di classificazione. Viene calcolato l'errore OOB per diversi valori del numero di campioni boostrap (e quindi di alberi) utilizzato dal modello, scegliendo il valore per cui l'errore OOB è minore. In questo caso è pari a 230, come si evince dalla Figura 6, in cui si riporta il grafico dell'errore OOB in funzione del numero di campioni bootstrap.

```
library(ipred)
nbag <- seq(10, 300, by = 10)
err <- rep(NA, length(nbag))
set.seed(909)
for(i in 1:length(nbag)){
  bag <- bagging(stima.bal$Frode ~., data = stima.bal,
                 nbagg = nbag[i], coob = TRUE)
  err[i] <- bag$err
  cat(i, "")
}
```

```
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
```

```
plot(nbag, err, xlab = "Numero di campioni bootstrap", ylab = "Errore OOB", type = "l",
     main = "")
nbag.opt <- nbag[which.min(err)]
abline(v = nbag.opt, col = 2, lty = "dashed")
```

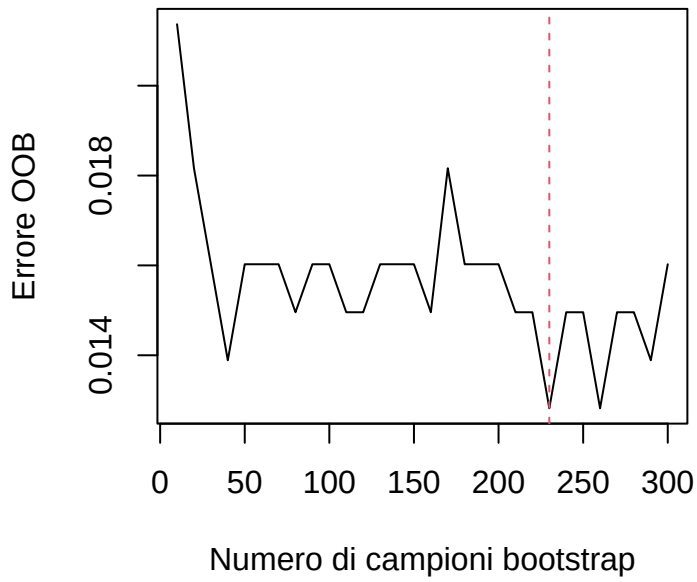

Figure 6: Errore OOB (Out-Of-Bag) nell'insieme di stima in funzione del numero di campioni bootstrap

```
set.seed(567)
bag <- bagging(stima.bal$Frode ~., data = stima.bal,
               nbagg = nbag.opt, coob = TRUE)
bag.pred.prob <- predict(bag, newdata = ver, type = "prob")[,2]
bag.tab <- tabella.sommario(bag.pred.prob > s, ver$Frode)
```

```
##         osservati
```

```
## previsti     0     1
##    FALSE 22093     0
##    TRUE    768    28
## errore totale:  0.03355324
## falsi positivi & falsi negativi:  0.03359433 0.00000000
```

```
tab <- c(tab, list(Bagging = bag.tab))
```

Il modello selezionato ottiene sull'insieme di verifica un tasso di errata classificazione pari al **3.36%** e una percentuale di falsi negativi pari allo **0.00%**.

Boosting

Si adatta un *boosting* con alberi di classificazione. Per individuare il numero di alberi necessari a stabilizzare l'errore di previsione, si divide l'insieme di stima in un insieme di stima ridotto e uno di convalida. La Figura 7 mostra l'errore di previsione nell'insieme di convalida in funzione del numero di iterazioni dell'algoritmo, facendo notare che l'errore è minimo e costante dopo 130 iterazioni.

```
library(ada)
set.seed(3)
boost <- ada(stima.rid$Frode ~ ., data = stima.rid,
             test.x = conv[, -ids.leak], test.y = conv$Frode, iter = 300)
plot(boost, test = TRUE)
```

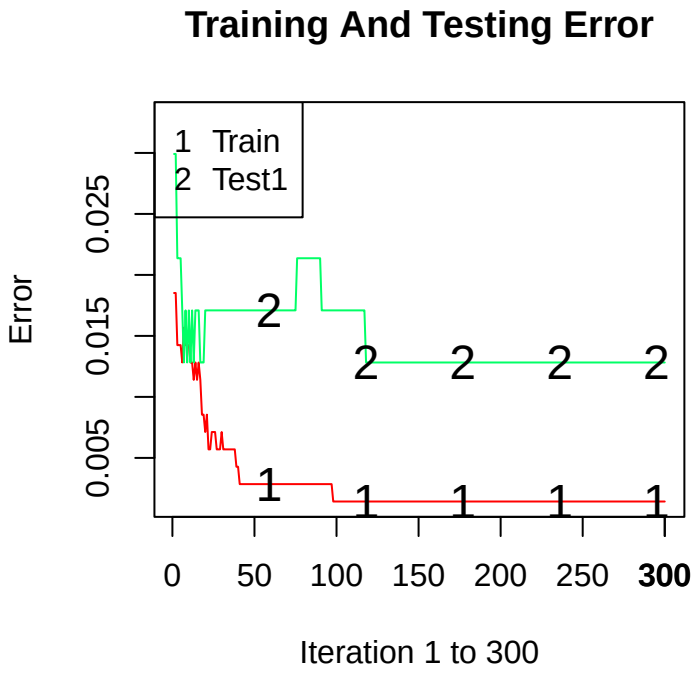

Figure 7: Errore di previsione nell'insieme di convalida in funzione del numero di iterazioni

Il modello selezionato, e riadattato sull'intero insieme di stima, ottiene un tasso di errata classificazione e una percentuale di falsi negativi nell'insieme di verifica pari rispettivamente allo **0.83%** e allo **0.00%**.
Anche questo modello ha il pregio di portare informazione sull'importanza delle variabili esplicative. La Figura 8 permette di far notare che le variabili maggiormente presenti negli stumps risultano essere il secondo, il terzo, il quarto, il quinto e il sesto indicatore di confronto della carta con carte ad essa simili.

```
boost <- ada(stima.bal$Frode ~., data = stima.bal, iter = 130)
boost.pred.prob <- predict(boost, newdata = ver, type = "prob")[,2]
boost.tab <- tabella.sommario(boost.pred.prob > s, ver$Frode)
```

```
##         osservati
## previsti     0     1
##    FALSE 22670     0
##    TRUE    191    28
## errore totale:  0.00834462
## falsi positivi & falsi negativi:  0.00835484 0.00000000
```

```
tab <- c(tab, list(Boosting = boost.tab))
varplot(boost, max.var.show = 10)
```

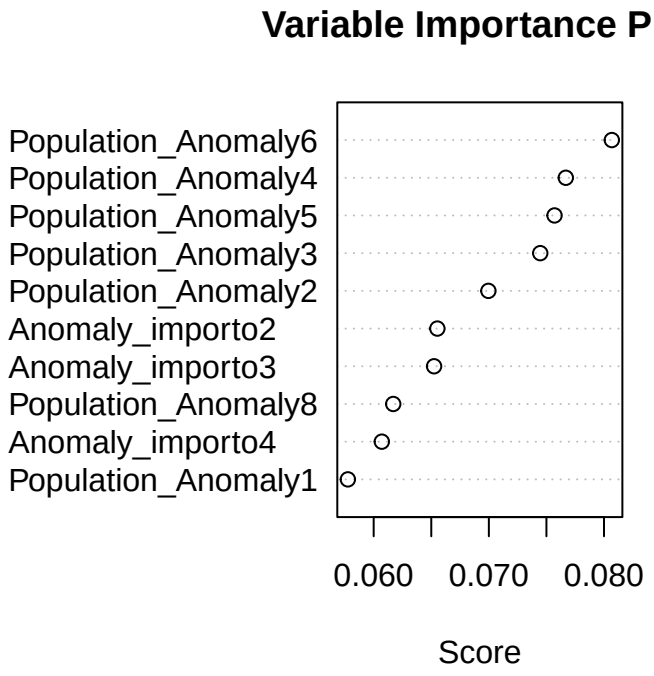

Figure 8: Importanza delle variabili nel boosting

Support Vector Machine

L'ultimo modello adattato è una *Support Vector Machine* con nucleo radiale. Il parametro di regolazione è il costo relativo alle errate classificazioni e, per selezionarlo, si suddivide l'insieme di stima in un insieme di

stima ridotto e uno di convalida e si considera una griglia di valori interi da 1 a 30. Il valore scelto è quello che corrisponde al modello con minor tasso di errata classificazione nell'insieme di convalida ed è pari a 27, come si può vedere dalla Figura 9. Il modello selezionato è adattato su tutto l'insieme di stima e ottiene, nell'insieme di verifica, un tasso di errata classificazione pari all' **1.74%** ed una percentuale di falsi negativi pari allo **0.00%**.

```
library(e1071)

ranges <- 1:30
err_svm <- matrix(NA,nrow = length(ranges),ncol =2)
colnames(err_svm) <- c("costo","errore")
set.seed(454)

for (i in 1:length(ranges)){
  cat("indice:",i,"  misura di costo:", ranges[i],"\n")
  s1 = svm(Frode ~ .,
           data = stima.rid, kernel = "radial", cost = ranges[i])
  pr.svm = predict(s1, newdata = conv)
  svm.tab = tabella.sommario(pr.svm, conv$Frode)
  err_svm[i,] = c(ranges[i], 1-sum(diag(svm.tab))/sum(svm.tab))
}

plot(err_svm, type="b")
```

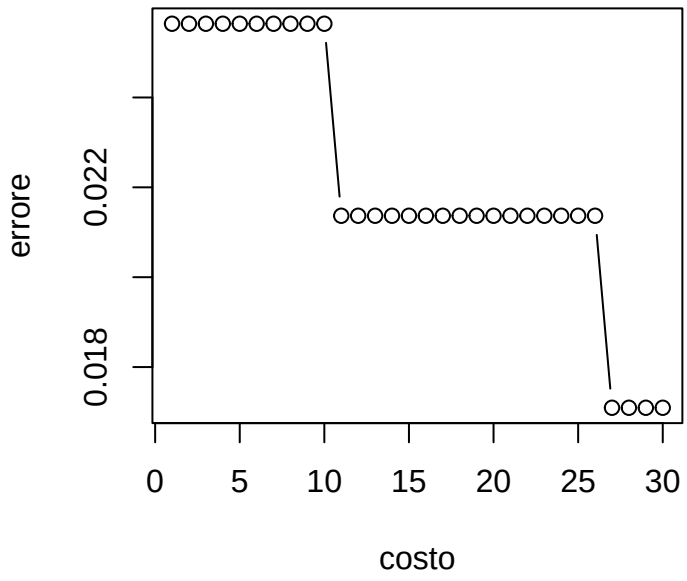

Figure 9: Tasso di errata classificazione nell'insieme di convalida in funzione del costo dell'errore

```
bestcost <- ranges[which.min(err_svm[,2])]

m.svm <- svm(Frode ~., data = stima.bal, cost = bestcost, probability = T)
p.svm <- predict(m.svm, newdata = ver, probability = T)
p.svm <- attr(p.svm,"probabilities")[,1]
svm.tab <- tabella.sommario(p.svm > s, ver$Frode)
tab <- c(tab, list(svm = svm.tab))
names(tab)[length(names(tab))] <- "Support Vector Machine"
```

Risultati

Nella Tabella 1 si riportano i risultati ottenuti coi diversi modelli adattati in termini di tasso di accuratezza (ovvero il complemento ad 1 del tasso di errata classificazione) e di proporzione di falsi negativi.

```
metriche.class = function(lista){
  n.mod = length(lista)
  nomi = names(lista)
  nomi.num = rep(NA, n.mod)
  for(i in 1:n.mod) nomi.num[i] = nomi[i]
  mat = matrix(NA, n.mod, 2)
  rownames(mat) = nomi.num
  colnames(mat) = c("Falsi negativi", "Accuratezza")
  for(i in 1:n.mod){
    mat[i,1] = fn = lista[[i]][1,2]/sum(lista[[i]][,2])
    mat[i,2] = acc = sum(diag(lista[[i]]))/sum(lista[[i]])
}
  return(mat)
}


knitr::kable(metriche.class(tab)[order(metriche.class(tab)[,2] ,decreasing = T),],
             caption = "Misure di performance dei modelli adattati",
             col.names = c("Falsi negativi", "Accuratezza"), align = "c",
             digits = 4 ,format = "simple")
```

Table 1: Misure di performance dei modelli adattati

	Falsi negativi	Accuratezza
Boosting	0.0000	0.9917
Analisi discriminante lineare	0.1071	0.9900
Support Vector Machine	0.0000	0.9826
Logistico stepwise	0.0000	0.9800
Albero	0.1071	0.9779
Bagging	0.0000	0.9664
Random Forest	0.0000	0.9634

Si nota che, ad eccezione dell'*albero di classificazione* e dell'*analisi discriminante lineare*, tutti i modelli non prevedono mai un'operazione fraudolenta come non fraudolenta. Alla luce di questa considerazione, sembra ragionevole scegliere il modello che permette di ottenere il tasso di accuratezza più elevato tra tutti i modelli che non prevedono falsi negativi. La Tabella 1 mostra come questa misura sia più elevata nel

boosting (99.17%), con a seguire la *Support Vector Machine* (98.26%), il modello *logistico stepwise* (98.00%), il *bagging* (96.64%) e il *random forest* (96.34%).
Focalizzando l'attenzione sul *boosting*, come già è stato detto in precedenza, questo modello permette di avere una misura di importanza delle variabili esplicative, senza però avere indicazione sulla direzione dell'effetto di queste variabili sulla risposta. In questo caso, le variabili maggiormente importanti risultano essere il secondo, il terzo, il quarto, il quinto e il sesto indicatore di confronto della carta con carte ad essa simili.