

Regressione Logistica Multinomiale

High Dimensional Data Analysis UNIMIB

Giorgio Nardi 819961, Eric Spinelli 799827, Luca Pretini 864014, Giacomo De Gobbi 860913

09/12/2020

Introduzione

La regressione logistica è largamente usata per modellare i risultati di una variabile dipendente categorica. Per le variabili categoriche è inappropriato utilizzare la regressione lineare dato che i valori risposta non sono misurati in rapporti di probabilità e i termini di errore non sono distribuiti normalmente. In aggiunta, il modello di regressione lineare può generare come valori predetti ogni numero reale da meno infinito a più infinito, mentre una variabile categorica può assumere solo un limitato numero di valori discreti in un range specificato. Mentre i modelli di regressione lineari uguaglano al valore atteso della variabile dipendente una combinazione lineare delle variabili dipendenti, i modelli lineari generalizzati equiparano la componente lineare a una certa funzione di probabilità di un dato risultato sulla variabile dipendente. Nella regressione logistica, tale funzione è la trasformazione logit: il logaritmo naturale delle odds che un certo evento accadrà. Nella regressione lineare, i parametri sono stimati usando il metodo dei minimi quadrati minimizzando la somma dei quadrati dei residui dei valori predetti dai valori osservati. Questo coinvolge la risoluzione di un sistema di N equazioni lineari, ognuna avente N variabili sconosciute, che solitamente è una operazione algebricamente semplice. Per la regressione logistica, la stima dei minimi quadrati non è in grado di produrre stimatori non distorti a varianza minima per i parametri. Al suo posto, è usata la stima di massima verosimiglianza per trovare i parametri che fittano nel migliore dei modi i dati.

Regressione logistica binomiale

Si consideri una variabile casuale Z che può assumere uno di due valori possibili. Dato un dataset di dimensione totale M , dove ogni osservazione è indipendente, Z può essere considerato un vettore colonna di M variabili casuali Z_i . Per convenzione, il valore 1 è associato al significato “successo” e un valore di 0 o 2 (ma non entrambi) significa “fallimento”. Per semplificare i dettagli computazionali della stima, è conveniente aggregare i dati in modo tale che ogni riga rappresenti una distinta combinazione delle variabili indipendenti. Queste righe sono spesso chiamate “popolazioni”. Sia N il numero totale delle popolazioni e sia n un vettore colonna con elementi n_i che rappresentano il numero di osservazioni nella popolazione i per $i = 1$ fino a N dove $\sum_{i=1}^N n_i = M$. Inoltre sia Y un vettore colonna di lunghezza N dove ogni elemento Y_i è una variabile casuale che rappresenta il numero di successi di Z per la popolazione i . Sia y il vettore colonna contenente gli elementi y_i che rappresentano il numero di osservato di successi per ogni popolazione. Sia π il vettore colonna, anch'esso di lunghezza N con elementi $\pi_i = P(Z_i = 1|i)$, cioè la probabilità di successo per ogni data osservazione nella i -esima popolazione. La matrice del disegno X è composta da N righe e $K + 1$ colonne, dove K è il numero della variabili indipendenti del modello. Per ogni riga della matrice del disegno il primo elemento $x_{i0=1}$ è l'intercetta. Il vettore dei parametri β , è un vettore colonna di lunghezza $K + 1$. Il modello logistico uguaglia la trasformazione logistica (il logaritmo delle odds della probabilità di successo) alla componente lineare.

Regressione logistica multinomiale

Per poter utilizzare una variabile dipendente multinomiale dobbiamo attuare alcuni aggiustamenti riguardo alle notazioni. Sia J il numero di categorie discrete che la variabile dipendente può assumere, dove $J \geq 2$. Ora consideriamo la variabile casuale Z che può assumere uno dei possibili valori di J . Se ogni osservazione è indipendente allora ogni Z_i è una variabile casuale multinomiale. Ancora una volta, aggreghiamo i dati in popolazioni ognuna delle quali rappresenta un'unica combinazione dei valori delle variabili indipendenti. Come nella regressione logistica binomiale, il vettore colonna n contiene n_i elementi che rappresentano il numero delle osservazioni nella popolazione i e tale che $\sum_{i=1}^N n_i = M$ sia la dimensione totale del campione. Sia y una matrice di N righe (una per ogni popolazione) e $J-1$ colonne. Per ogni popolazione, y_{ij} rappresenta il numero osservato del j -esimo valore di Z_i . Similmente π è una matrice della stessa dimensione di y dove ogni elemento π_{ij} è la probabilità di osservare il j -esimo valore della variabile dipendente per ogni data osservazione nella i -esima popolazione.

La matrice del disegno delle variabili indipendenti X contiene N righe e $K + 1$ colonne dove K è il numero delle variabili indipendenti e il primo elemento di ciascuna riga $x_{i0}=1$ è l'intercetta. Sia beta una matrice con $K + 1$ righe e $J - 1$ colonne, in modo che ogni elemento β_{kj} contiene la stima del parametro per la k -esima covariata e il j -esimo valore della variabile dipendente. Per il modello di regressione logistica multinomiale, equipariamo la componente lineare al logaritmo delle odds di una j -esima osservazione comparato alla J -esima osservazione. Cioè considereremmo la J -esima categoria come omessa o come categoria di base, dove le funzioni logit riguardanti le prime $J-1$ categorie sono costruite con la categoria di base al denominatore.

Assunzioni

La regressione logistica multinomiale è spesso scelta per la semplicità delle sue assunzioni, non richiede infatti di verificare le ipotesi di normalità, linearità o omoschedasticità. Si può dunque considerare come un metodo più versatile rispetto ad alcuni concorrenti come ad esempio la Discriminant function analysis, la quale richiede senza dubbio più attenzione alle assunzioni. Tuttavia, per la logistica multinomiale, bisogna tener conto dell'assunzione di indipendenza delle alternative irrilevanti (anche detta IIA, "Independence of irrelevant alternatives"), la quale afferma che la probabilità di preferire una classe rispetto ad un'altra non dipende dall'aggiunta o meno di altre alternative irrilevanti (rispetto alla scelta). Bisogna prendere in considerazione questa ipotesi in ambiti come studi di scienze sociali o economiche: ad esempio, in un contesto di elezioni politiche tra due candidati di ideologie opposte, come possono essere un candidato liberale ed uno conservatore, l'aggiunta di un terzo candidato anch'esso liberale va chiaramente a creare uno scenario diverso rispetto a quello in presenza di due soli candidati: in questo scenario si può parlare di "violazione di IIA", mentre, in uno scenario in cui un terzo candidato non cambia le scelte dei votanti rispetto ai primi due candidati, si parla di "soddisfazione delle condizioni di IIA". La soddisfazione o meno delle condizioni di indipendenza delle alternative irrilevanti può essere verificata tramite alcuni test come il test di Hausman-McFadden o di Small-Hsiao. In aggiunta, bisogna verificare la soddisfazione dell'ipotesi di non-perfect separation. Nello specifico, in condizioni di perfect separation i gruppi della variabile risposta vengono separati perfettamente da uno o più predittori. In tale situazione avviene una stima non realistica dei coefficienti, esagerata in termini di grandezza.

Stima dei parametri

Si può scrivere la funzione di log-verosimiglianza per il modello logistico multinomiale come:

$$l(\beta) = \sum_{i=1}^N \sum_{j=1}^{J-1} (y_{ij} \sum_{k=0}^K x_{ik} \beta_{kj}) - n_i \log [1 + \sum_{j=1}^{J-1} \exp (\sum_{k=0}^K x_{ik} \beta_{kj})] \quad (1)$$

l'obiettivo è trovare quei valori di β che massimizzano l'equazione 1. Lo svolgimento prevede l'utilizzo del metodo di Newton-Raphson, che è un metodo iterativo per calcolare gli zeri di una funzione. Tale algoritmo

esige l'utilizzo di derivata prima e derivata seconda della funzione di log-verosimiglianza. Si procede dunque col calcolo della derivata prima:

$$\frac{\delta l(\beta)}{\delta x} = \sum_{i=1}^N y_{ij} x_{ik} - n_i \cdot \frac{1}{1 + \sum_{j=1}^{J-1} \exp(\sum_{k=0}^K x_{ik} \beta_{kj})} \cdot \frac{\delta}{\delta \beta_{kj}} (1 + \sum_{j=1}^{J-1} \exp(\sum_{k=0}^K x_{ik} \beta_{kj})) = \\ \sum_{i=1}^N y_{ij} x_{ik} - n_i \pi_{ij} x_{ik} \quad (2)$$

Si noti che nella equazione 2 vi sono $(J-1) \cdot (K+1)$ equazioni che vanno poste uguali a 0 e risolte per ciascun β_{kj} . A questo punto, si può formare la matrice delle derivate seconde parziali. Per ogni β_{kj} si procede calcolando la derivata prima dell'equazione 2 rispetto a ogni altro β_{kj} :

$$\frac{\delta^2 l(\beta)}{\delta \beta_{kj} \delta \beta_{k'j'}} = \frac{\delta}{\delta \beta_{k'j'}} \sum_{i=1}^N y_{ij} x_{ik} - n_i \pi_{ij} x_{ik} = - \sum_{i=1}^N n_i x_{ik} \frac{\delta}{\delta \beta_{k'j'}} \left(\frac{\exp(\sum_{k=0}^K x_{ik} \beta_{kj})}{1 + \sum_{j=1}^{J-1} \exp(\sum_{k=0}^K x_{ik} \beta_{kj})} \right) \quad (3)$$

A questo punto, è importante considerare che le derivate di numeratore e denominatore differiscono dipendentemente dal fatto che $j' = j$ o $j' \neq j$. Infatti, indicando con $f'(a)$ la derivata del numeratore e con $g'(a)$ la derivata del denominatore, si ha:

$$f'(a) = g'(a) = \exp(\sum_{k=0}^K x_{ik} \beta_{kj}) \cdot x_{ik'} \quad j = j' \quad (4)$$

$$f'(a) = 0 \quad g'(a) = \exp(\sum_{k=0}^K x_{ik} \beta_{kj'} \cdot x_{ik'}) \quad j' \neq j \quad (5)$$

Dunque, svolgendo la derivata del quoziente, quando $j' = j$ si ottiene:

$$\pi_{ij} x_{ik'} (1 - \pi_{ij}) \quad (6)$$

mentre quando $j' \neq j$:

$$-\pi_{ij} x_{ik'} \pi_{ij'} \quad (7)$$

In aggiunta, si può esprimere la matrice delle derivate seconde parziali come:

$$\frac{\delta^2 l(\beta)}{\delta \beta_{kj} \delta \beta_{k'j'}} = - \sum_{i=1}^N n_i x_{ik} \pi_{ij} (1 - \pi_{ij}) x_{ik'} \quad j' = j \quad (8)$$

$$\frac{\delta^2 l(\beta)}{\delta \beta_{kj} \delta \beta_{k'j'}} = \sum_{i=1}^N n_i x_{ik} \pi_{ij} p_{ij'} x_{ik'} \quad j' \neq j$$

Per illustrare la procedura iterativa del metodo di Newton-Raphson si parte dall'equazione:

$$\beta^{(1)} = \beta^{(0)} + [-l''(\beta^{(0)})]^{-1} \cdot l'(\beta^{(0)}) \quad (9)$$

Sia μ una matrice con N righe e $J-1$ colonne formata dagli elementi $n_i \pi_{ij}$, allora si può scrivere:

$$l'(\beta) = X^T(y - \mu) \quad (10)$$

A questo punto si consideri W come una matrice quadrata di dimensione $(J-1) \times (J-1)$ in cui ogni sottomatrice $W_{jj'}$ è una matrice diagonale $N \times N$. Per $j = j'$ l'elemento diagonale in W_{jj} è $n_i \pi_{ij} (1 - \pi_{ij})$. Mentre quando $j \neq j'$ l'elemento diagonale in $W_{jj'}$ è $-n_i \pi_{ij} \pi_{ik}$. Servendosi di tale matrice, si può percorrere ogni passo del metodo di Newton-Raphson come nella regressione logistica binomiale. Utilizzando:

$$\beta^{(1)} = \beta^{(0)} + [X^T W X]^{-1} \cdot X^T(y - \mu)$$

Regolarizzazione

Si è precedentemente illustrato il metodo di stima senza nessun tipo di penalità, tuttavia, oggigiorno molti pacchetti di software statistici eseguono la procedura aggiungendo una regolarizzazione alla funzione di log-verosimiglianza da minimizzare.

$$l_p(\beta) = l(\beta) - \lambda F(\beta) \quad (11)$$

dove λ è il parametro di tuning e $F(\beta)$ è una funzione che solitamente contrae i coefficienti. Generalmente, nei modelli lineari, si utilizza la penalità lasso per effettuare lo shrinkage con variable selection al fine di migliorare l'interpretabilità del modello. Tuttavia, nel caso della regressione logistica multinomiale, dove si ha una matrice di coefficienti β , il lasso selezionerà differenti variabili per differenti classi. Ciò significa che sebbene il vettore di ciascun coefficiente può essere sparso non vuol dire che il modello sia anch'esso sparso. Un approccio alternativo e molto usato nella pratica consiste nell'utilizzo della penalità grouped-lasso. La funzione da minimizzare risulta essere:

$$l_p(\beta) = l(\beta) - \lambda \sum_{k=1}^K \|\beta_k\|_2 \quad (12)$$

E' importante sottolineare che tale criterio coinvolge la somma delle norme ordinarie $l_2 \|\cdot\|_2$, e non le norme l_2 al quadrato. L'effetto di questa penalità è quello di selezionare tutti i coefficienti di una particolare variabile in modo che siano incluse o escluse dal modello.

Bontà del modello

Come già anticipato nell'introduzione, la classe della regressione logistica si differenzia in vari aspetti dalla regressione lineare. A differenza di quest'ultima non si può utilizzare l'indice R^2 standard per misurare la bontà di adattamento ai dati. Per la regressione logistica binomiale sono stati studiati e implementati numerosi indici *pseudo*- R^2 come quello di McFadden, quello di Cox & Snell e quello di Tjur solo per citarne alcuni. Per quanto riguarda la regressione logistica multinomiale, si hanno pochissimo esempi di tali misure e quelle poche a disposizione non riscontrano l'adesione unanime dei ricercatori. Una delle più studiate e utilizzate è l'adattamento del test di Hosmer-Lemeshow per il caso multinomiale proposto da Fagerland, Hosmer & Bofin(2008). Tale test è implementato nella libreria *nnet* utilizzando il comando **logitgof**, mentre non è disponibile per la libreria *glmnet*. E' stata fatta una prova con il dataset usato per l'applicazione, ma data la dimensione e dato che il modello predice per la quasi totalità delle osservazioni la classe di appartenenza con probabilità prossima a uno, il test ha dato problemi di esecuzione e di interpretabilità.

Pacchetto Glmnet

Glmnet è un pacchetto R usato per stimare modelli lineari generalizzati attraverso la minimizzazione della massima verosimiglianza penalizzata. In particolare, per mezzo della corretta impostazione del parametro family permette di elaborare modelli logistici multiclass. Di base, *glmnet* considera la penalità elastic net:

$$\lambda \cdot [(1-\alpha)\|\beta\|_2^2/2 + \alpha\|\beta\|_1] \quad (13)$$

Il tipo di regolarizzazione dipende dalla scelta del parametro α , il quale assume valori compresi tra 0 (ridge) e 1 (lasso). Nello specifico, si sceglie di settare il parametro uguale a 1 nell'applicazione presa in considerazione.

Applicazione

Si è deciso di prendere in considerazione un dataset contenente le cifre da 0 a 9 scritte manualmente come variabile risposta. Ciascuna delle variabili esplicative del set di dati corrisponde a un singolo pixel dell'immagine

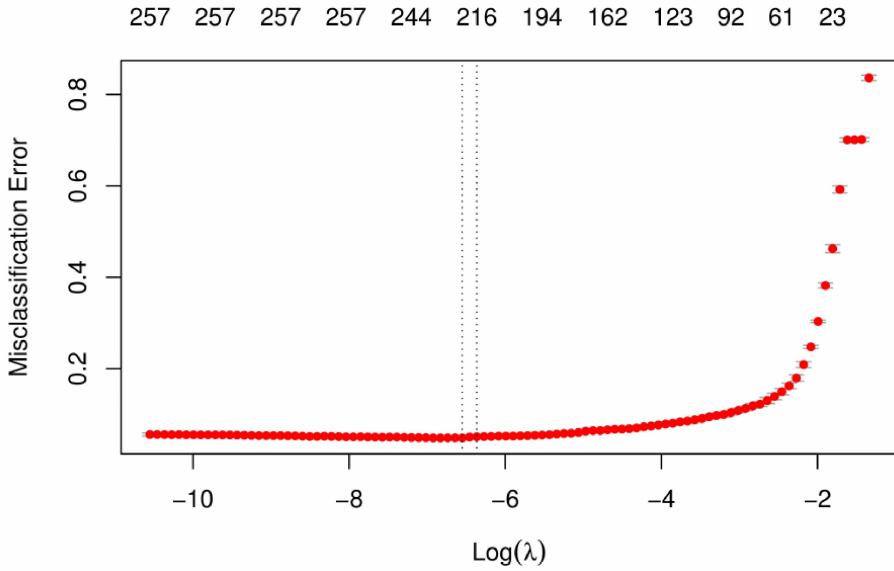


Figure : Grafico 1

che rappresenta ciascuna cifra. Il training set è composto da $N = 7291$ immagini di cifre. Il test set si compone di 2007 osservazioni. La variabile target è costituita da 10 categorie. Si procede effettuando una stima sul training set del modello di regressione logistica multinomiale con penalità grouped lasso.

Il Grafico 1 rappresenta la procedura di 5-folds cross validation effettuata per selezionare il valore di lambda che minimizza il misclassification error. Nella parte superiore del grafico si può notare come l'effetto di variable selection aumenta al crescere di λ , il parametro di tuning. Si decide di effettuare la stima dei coefficienti considerando un $\lambda = 0.00172$. Tale valore corrisponde al parametro più estremo entro una deviazione standard dal valore di λ minimo. Questa scelta, suggerita dalla one-standard error rule, permette di incrementare l'interpretabilità del modello a fronte di un piccolo aumento di misclassification error. Il modello stimato è caratterizzato da 216 coefficienti per ciascuna classe.

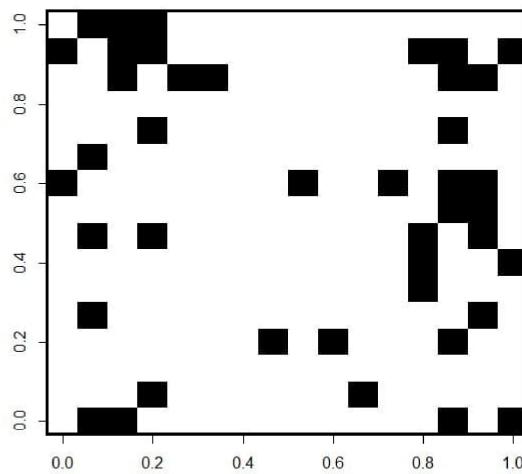


Figure : Grafico 2

Nel Grafico 2 si può apprezzare l'effetto di variable selection. L'eliminazione delle features in gruppo si

riflette sullo spazio di pixel comunque a tutte le classi, quindi i pixel neri corrispondono alle variabili eliminate dall'intero dataset di partenza. La feature selection implica che il modello non considererà i pixel esclusi nella predizione di una nuova immagine, perchè i rispettivi coefficienti sono posti uguali a zero.

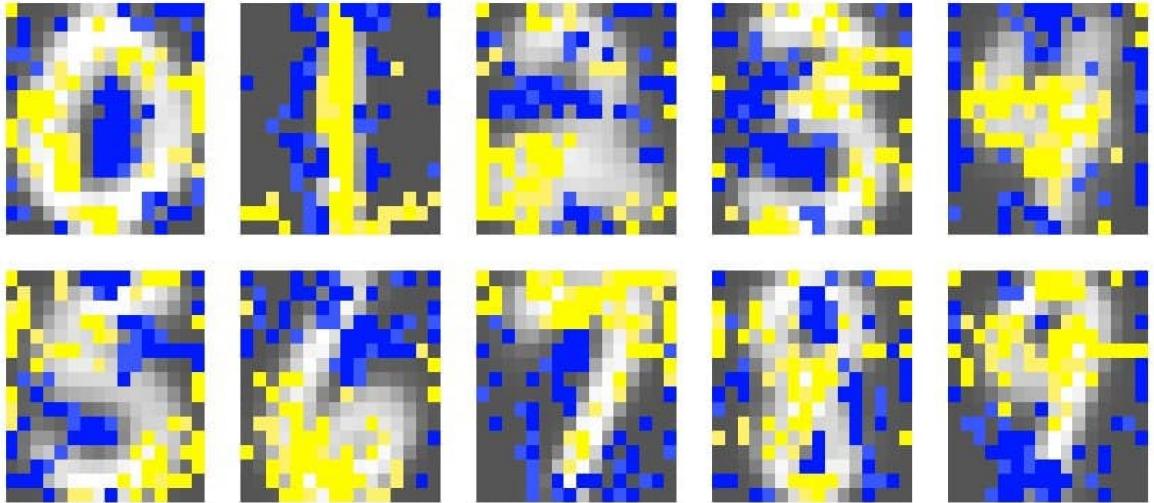


Figure : Grafico 3

Nel Grafico 3 si rappresenta l'effetto dei coefficienti per ciascuna classe. Per facilitare l'interpretabilità del risultato si è scelto di raffigurare solo i coefficienti più estremi: rispetto alla loro distribuzione si è selezionato il 20% più negativo in blu e il 20% più positivo in giallo. Rispetto a ciascun colore si è scelta una sfumatura in modo da dare risalto ai coefficienti più estremi in assoluto. Dal grafico si evince che tendenzialmente i coefficienti soddisfano l'aspettativa rispetto alla forma delle relative classi. Infatti si noti come in generale i coefficienti positivi ricalcano gli spazi interessati dalla forma della cifra, mentre quelli negativi tendono a distribuirsi in zone non interessate da essa.

	0	1	2	3	4	5	6	7	8	9
0	350	0	4	2	2	4	1	0	7	0
1	0	252	0	0	2	0	0	1	0	2
2	1	1	172	4	7	0	3	2	3	1
3	3	3	4	146	0	8	0	2	3	0
4	3	3	7	1	175	2	3	5	1	4
5	0	0	1	8	1	141	3	0	5	1
6	0	2	3	0	2	0	159	0	1	0
7	0	0	1	2	1	1	0	131	0	1
8	1	1	6	2	3	1	1	1	143	1
9	1	2	0	1	7	3	0	5	3	167

Figure : Tabella 1

Il modello elaborato raggiunge un'accuracy di 91,57% sul test set. Infine, si è deciso di stimare un modello con la penalità lasso, per mostrare la differenza con Grouped Lasso nell'effetto di shrinkage. In particolare, nella tabella sottostante si può notare come Grouped Lasso annulli interi vettori in blocco all'interno della matrice dei coefficienti. D'altra parte, la penalità lasso effettua una massiccia variable selection, ma nella maggior parte dei casi si ha l'annullamento solo di alcuni elementi del singolo vettore.

cifra	nzerosLasso	nzerosGroupedL
1	200	41
2	220	41
3	182	41
4	182	41
5	188	41
6	175	41
7	185	41
8	205	41
9	202	41

Figure : Tabella 2

Appendice: codice R

```
Training<- read.table(gzfile('\zip.train.gz'), header=F)
Test=read.table(gzfile('\zip.test.gz'), header=F)
library(data.table)
setnames(Training, "V1", "Response")
setnames(Test,'V1','Response')
Training$Response <- factor(Training$Response)
Test$Response <- factor(Test$Response)

# costruisco matrice del disegno per training e test,
# vettore delle risposte per training e test
mx <- as.matrix(Training[2:257])
my <- as.matrix(Training[1])

mx.star=as.matrix(Test[2:257])
my.star<- as.matrix(Test[1])

#numero di osservazioni in training e test set
n=nrow(Training)
m=nrow(Test)

#Si procede utilizzando la funzione cv.glmnet sul training set
#per trovare il valore di lambda ottimale
library(glmnet)
set.seed(12345)
nFolds <- 5
foldid <- sample(1:nFolds,n,replace=T)
cvlambda=cv.glmnet(mx,my,type.measure='class',alpha=1,family='multinomial',
                    nfolds=5,type.multinomial='grouped',foldid = foldid)
plot(cvlambda)

####grafici coefficienti grouped-lasso
#si crea un dataset in cui le variabili sono rispettive a ciascuna classe
d<-coef(cvlambda,s='lambda.1se')
coeffs<-data.frame(v0=as.numeric(d$'0')[2:257],v1=as.numeric(d$'1')[2:257],
                     v2=as.numeric(d$'2')[2:257],v3=as.numeric(d$'3')[2:257],
                     v4=as.numeric(d$'4')[2:257],v5=as.numeric(d$'5')[2:257],
                     v6=as.numeric(d$'6')[2:257],v7=as.numeric(d$'7')[2:257],
                     v8=as.numeric(d$'8')[2:257],v9=as.numeric(d$'9')[2:257])
#si individuano le righe che in gruppo sommano a zero,
#corrispondenti ai coefficienti annullati
zero_coeffs<-which(apply(coeffs,1,sum)==0)
#si dispone una matrice di zeri, in cui la cifra 1 è presente
#rispetto alla posizione dei coefficienti annullati.
mat_image=as.numeric(rep(0,256))
for(i in 1:256){
  mat_image[i]=0
  if(i %in% zero_coeffs){
    mat_image[i]=1
  }
}
#questa funzione consente di ruotare una qualunque immagine bidimensionale
```

```

rotate <- function(x) t(apply(x, 2, rev))
image(rotate(matrix(mat_image, ncol=16, byrow=T)), col=c('#FFFFFF', '#000000'), axes=FALSE)

library(fields)
#Questa funzione computa la media di tutte le immagini rispetto a ciascuna classe
#e sovrascrive i pixel interessati dai coefficienti più rilevanti
#con colori corrispondenti alla loro intensità ed al loro effetto

#parametri:
#coeff_variable --> singola variabile rispetto alla matrice dei coefficienti
#class --> valore corrispondente alla classe presa in considerazione
#quant_min --> percentuale dei coefficienti più piccoli che si vogliono considerare,
#rispetto alla distribuzione di tutti i coefficienti
#quant_max --> percentuale dei coefficienti più grandi che si vogliono considerare,
#rispetto alla distribuzione di tutti i coefficienti

coeffsOverNum<-function(coeff_variable,class,quant_min,quant_max){

  x<-coeff_variable
  k<-x
  z<-class
  quantile_min<- quant_min
  quantile_max<-quant_max

  #si selezionano i coefficienti rispetto ai perfentili scelti
  coefs_min<-x[which(x<quantile(x,quantile_min))]
  coefs_max<-x[which(x>quantile(x,quantile_max))]

  #si distinguono ulteriori classi per far risaltare graficamente
  #i coefficienti più significativi rispetto a ciascun estremo
  coefs_min1<-coefs_min[coefs_min<quantile(coefs_min,0.7)] #i più piccoli
  coefs_min2<-coefs_min[coefs_min>quantile(coefs_min,0.7)] #i meno piccoli
  coefs_max1<-coefs_max[coefs_max<quantile(coefs_max,0.3)] #i meno grandi
  coefs_max2<-coefs_max[coefs_max>quantile(coefs_max,0.3)] #i più grandi
  #si crea l'immagine del numero in background
  num<-matrix(as.vector(colMeans(mx[which(my==z),])),ncol=16,byrow=T)
  num_t<-t(num)

  #nelle posizioni dei coefficienti estremi si sostituiscono numeri alti,
  #in modo che i colori di questi pixel si distinguano dagli altri.
  #Sono inseriti numeri intermedi per creare una sfumatura rispetto ai coefficienti
  #meno rilevanti rispetto a quelli considerati

  min1=which(k %in% coefs_min1)
  for(i in min1){num_t[i]=4}

  min2=which(k %in% coefs_min2)
  for(i in min2){num_t[i]=3}

  max1=which(k %in% coefs_max1)
  for(i in max1){num_t[i]=5}

  max2=which(k %in% coefs_max2)
}

```

```

for(i in max2){num_t[i]=6}

num_t1<-num_t+1
surface <- list(x=seq(0,16), y=seq(0,16), z=rotate(t(num_t)))

library(fields)
colorTable<- designer.colors(500, c("#555555", "#747474", "#BCBCBC",
                                    "#DCDCDC", "#FFFFFF", "#FFFFFF",
                                    "#3A57FF", "#001AFF", "#FFF16B", "#FFFD00"),
                                    x = c(0,0.3,0.7,1.1,1.4,1.8,4,5,6,7) / 7)
image(surface,col=colorTable,axes=FALSE)
}

#si applica la funzione coeffsOverNum rispetto alle variabili del dataframe
#dei coefficienti rispetto a ciascuna classe

l<-c('v0','v1','v2','v3','v4','v5','v6','v7','v8','v9')
v<-seq(0,9)
par(mfrow = c(2,5))
for(i in 1:10){
  x<-l[i]
  coeffsOverNum(coeffs[,x],v[i],0.2,0.8)
}
par(mfrow = c(1,1))

#previsione classi modello con lambda minimo
prediction=predict(cvlambda,newx=mx.star,type='class',s='lambda.1se')
#confusion matrix
confm=table(prediction,my.star)
library(knitr)
kable(confm)
#accuracy
acc=sum(diag(confm))/m

####confronto con lasso
cvlambda2=cv.glmnet(mx,my,type.measure='class',alpha=1,family='multinomial',
                     nfolds=5,foldid = foldid)
coef_cv2=coef(cvlambda2, s = "lambda.1se")
d2=data.frame(as.matrix(coef_cv2$'0'),as.matrix(coef_cv2$'1'),as.matrix(coef_cv2$'3'),
               as.matrix(coef_cv2$'4'),as.matrix(coef_cv2$'5'),as.matrix(coef_cv2$'6'),
               as.matrix(coef_cv2$'7'),as.matrix(coef_cv2$'8'),as.matrix(coef_cv2$'9'))
a2=which(apply(d2,1,sum)==0)
nzerosLasso=c()
for(i in 1:9){
  a=sum(d2[,i]==0)
  nzerosLasso[i]=a
}
nzerosGroupedL=c()
for(i in 1:9{
  b=sum(coeffs[,i]==0)
  nzerosGroupedL[i]=b
}
cifra=c(1:9)

```

```
cont=cbind(cifra,nzerosLasso,nzerosGroupedL)
t=kable(cont)
t
```

Bibliografia

- Maximum Likelihood Estimation of Logistic Regression Models: Theory and Implementation, Scott A. Czepiel, <https://czep.net/stat/mlplr.pdf>
- Statistical Learning with Sparsity, Hastie,Tibshirani,Wainwright,https://web.stanford.edu/~hastie/StatLearnSparsity_files/SLS.pdf
- Glmnet Vignette, Trevor Hastie and Junyang Qian, https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html#:~:text=Glmnet%20is%20a%20package%20that,in%20the%20input%20matrix%20x%20.
- Multinomial Logistic Regression, Dr. Jon Starkweather and Dr. Amanda Kay Moske, https://it.unt.edu/sites/default/files/mlr_jds_aug2011.pdf
- Multinomial goodness-of-fit tests for logistic regression models, Morten W. Fagerland1, David W. Hosmer2 and Anna M. Bofin3,[file:///C:/Users/preinstalled/Downloads/Multinomial_goodness_of_fit_tests_for_logistic_regression_models%20\(1\).pdf](file:///C:/Users/preinstalled/Downloads/Multinomial_goodness_of_fit_tests_for_logistic_regression_models%20(1).pdf)
- Logistic Regression: From Binary to Multi-Class, Shuiwang Ji and Yaochen Xie, <http://people.tamu.edu/~sji/classes/LR.pdf>
- The Elements of Statistical Learning, Trevor Hastie, Robert Tibshirani, Jerome Friedman, <https://web.stanford.edu/~hastie/Papers/ESLII.pdf>