GRUPPO LASSO

Erisa Dajcaj

Abstract

La crescita tecnologica negli ultimi anni ha dato frutto all'evoluzione più profonda e pervasiva del mondo digitale: il fenomeno Big Data.

Figure 1: Big data

Basterebbe formulare una semplice query "Quanti dati vengono prodotti ogni giorno" su Google e le statistiche sono sorprendenti:

> - **1.7MB of data** is created every second by every person during 2020.
> - In the last two years alone, the astonishing **90%** of the world's data has been created.
> - **2.5 quintillion bytes** of data are produced by humans every day.
> - **463 exabytes** of data will be generated each day by humans as of 2025.
> - **95 million** photos and videos are shared every day on Instagram.
> - By the end of 2020, **44 zettabytes** will make up the entire digital universe.
> - Every day, **306.4 billion emails** are sent, and **500 million Tweets** are made.

Figure 2: Dati prodotti ogni giorni.

Avere tutte queste informazioni ibride a disposizione, è senz'altro una ottima opportunità, ma riuscire ad estrarre e "scegliere i dati buoni" in una dimensione di dati così grande, risulta essere uno dei task più difficili. Richard Bellman, nel 1961, ha chiamato questo fenomeno come "La maledizione della dimensionalità".

Le statistiche ad alta dimensione si riferiscono all'inferenza statistica quando il numero di parametri sconosciuti p è molto più grande della dimensione del campione n: p»n.

In questo lavoro viene descritta in dettaglio la tecnica di regolarizzazione Gruppo Lasso, che è una estensione del Lasso, usata per effettuare selezione di variabili su gruppi di variabili.

- Obiettivi:
  - descrivere in dettaglio la tecnica e i relativi algoritmi di stima;
  - analizzare le principali implementazioni software disponibili in R;
  - presentare un esempio realistico in R relativo alla regressione logistica.

Concetti base

In seguito vengono forniti alcuni concetti necessari per comprendere al meglio le tecniche e gli algoritmi trattati.

Variabili categoriali
Siano $x_1, ..., x_n$ n variabili rapprentati da codici opportuni all'interno di un dataset. Esse vengono definite come variabili categoriali. Spesso si usano come codici dei numeri, ma il loro significato non è numerico e nessuna operazione aritmetica ha significato, a parte il contare le unità in ciascuna categoria. Le variabili categoriali si possono distinguere in variabili ordinali e nominali.

Gradi di libertà
I gradi di libertà di una variabile aleatoria esprimono il numero minimo di dati sufficienti a valutare la quantità d'informazione contenuta nella variabile.

Funzione convessa
Sia $f : \mathbb{R}$ una funzione definita su un intervallo. f si dice convessa se il segmento che congiunge due qualsiasi punti del suo grafico si trova al di sopra del grafico stesso. Un esempio di una funzione convessa è la funzione quadratica $f(x) = x^2$.

Subdifferenziale
* Sia $f : \mathbb{R}^p \to \mathbb{R}$ una funzione convessa. Sia $x \in \mathbb{R}$, un vettore $d \in \mathbb{R}^p$ è chiamato *subgradiente* di f in x se:

$$f(y) \geq f(x) + (y - x)^T d.$$

L'insieme di tutti i subgradienti della funzione f in x è chiamato **"Subdifferenziale di f in x"** e viene denotato con $\partial f(x)$. Una condizione sufficiente e necessaria che x sia un punto di minimo per f è: $0 \in \partial f(x)$.

Base ortogonale
Sia V uno spazio vettoriale di dimensione finita sul campo S, nel quale sia definito un prodotto scalare. Una base ortogonale per V è una base composta da vettori $v_1, ..., v_n$ a due a due ortogonali, ovvero tale che il loro prodotto sia pari a 0: $\langle v_1, v_j \rangle = 0, i \neq j$

Base ortonormale
Una base ortonormale è una base ortogonale in cui ogni vettore ha norma 1: $\langle v_1, v_j \rangle = \delta_{ij}$, con $delta_{ij}$ delta di Kronecker.

Matrice definita positiva
Sia $A_{n \times n}$ una matrice quadrata e sia $x \in \mathbb{R}^n$. A si dice una matrice definita positiva se $xAx^T > 0, \forall x \in \mathbb{R}^n, x \neq 0$.

Modello lineare
Sia $Y \in \mathbb{R}^n$ una variabile di risposta continua e X una matrice di $n \times p$. Siano $\epsilon_i, i = 1, \dots n$ variabili i.i.d di X_i avente $\mathbb{E}[\epsilon_i] = 0$ e sia $\beta \in \mathbb{R}^n$ un vettore parametro. Un modello lineare ad alta dimensione è data da:

$$Y_i = \sum_{j=1}^{p} \beta_j X_i^{(j)} + \epsilon_i$$

Modello logit o Regressione Logistica
Sia Y una variabile dicotomica che assume valori in $\{0, 1\}$. Il modello logit si presenta come:

$$logit(\mathbb{P}(Y = 1|X)) = \mu + \sum_{i=1}^{p} X_i \beta_i + \sum_{i<j} X_{i,j} \beta_{i,j}$$

CROSS VALIDATION
Cross-validation è una procedura che viene utilizzata per selezionare un valore ottimale per il parametro di restringimento / regolazione, λ, suddividendo in modo casuale il dataset in training set e validation set.

LASSO

Lasso sta per "Least Absolute Shrinkage and Selection Operator". E' una tecnica di regolarizzazione introdotta da Tibshirani nel 1996, inizialmente per modelli di regressione lineare, che effettua una selezione di variabili riducendo alcuni coefficienti $\hat{\beta}_j(\lambda)$ esattamente a 0, per λ molto grande. Gli altri coefficienti, diversi da zero, rappresentano variabili rilevanti per il modello.

I parametri del modello lineare definito precedentemente, vengono stimati con la penalizzazione ℓ_1 del Lasso come:

$$\hat{\beta}(\lambda) = argmin_\beta \left(\frac{1}{2} ||\mathbf{Y} - \mathbf{X}\beta||_\mathbf{2}^\mathbf{2} + \lambda ||\beta||_\mathbf{1} \right)$$

dove $||\mathbf{Y} - \mathbf{X}\beta||_\mathbf{2}^\mathbf{2} = \sum_{i=1}^{n} (Y_i - (\mathbf{X}\ \beta)_i)^2, ||\beta||_1 = \sum_{i=1}^{p} |\beta|_1; \lambda \geq 0$ è il parametro di penalizzazione dal quale dipende la selezione delle variabili.

Osservazione: Lasso si presta ad una ampia varietà di modelli proprio grazie al suo approccio di probabilità penalizzato.

In seguito un esempio pratico dell'utilizzo del Lasso sul dataset Boston

```
#Carico il dataset da leggere
data = read.csv('Boston_Housing.csv')
set.seed(123)

#Preprocessione dei data: fase importantissima cui scopo è togliere dal modello
#tutti i valori indicati con NA
data <- na.omit(data)

#Standardizzazione dei dati viene effettuata come segue:
#Si considerano le osservazioni; si sottrae per la media delle colonne;
#poi si divide per la deviazione standard della colonna
data_scaled <- cbind(scale(data[,1:13]),data[,14])

# TrainSet 80/20
size <- floor(0.8 *  nrow(data_scaled))

#Si recuperano tutti i dati normalizzati in modo random.
train_ind <- sample(seq_len(nrow(data_scaled)), size = size)

train <- data_scaled[train_ind, ]
xtrain <- train[,1:13] #considero 13 feature
ytrain <- train[,14]

# Creiamo tutte le variabili che non sono state scelte.
# Test values
test <- data_scaled[-train_ind,]
xtest <- test[,1:13]
ytest <- test[,14]

lambda.array <- seq(from = 0.01, to = 100, by = 0.01)

#Libreria che permette di utilizzare il lasso
library(glmnet)
```

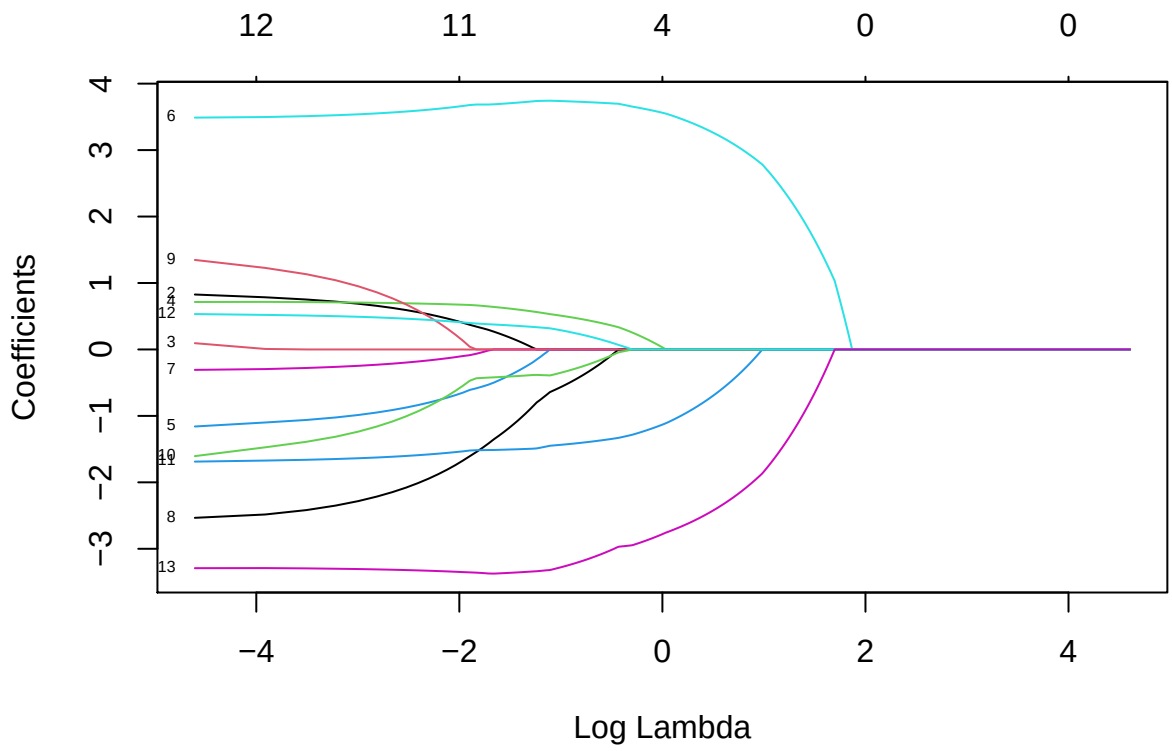
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-1
```

```
# alpha=1 indica che si vuole eseguire lasso regression
lassoFit <- glmnet(xtrain,ytrain, alpha=1, lambda=lambda.array)
summary(lassoFit)
```

```
##           Length Class     Mode
## a0         10000 -none-    numeric
## beta      130000 dgCMatrix S4
## df         10000 -none-    numeric
## dim            2 -none-    numeric
## lambda     10000 -none-    numeric
## dev.ratio  10000 -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
```

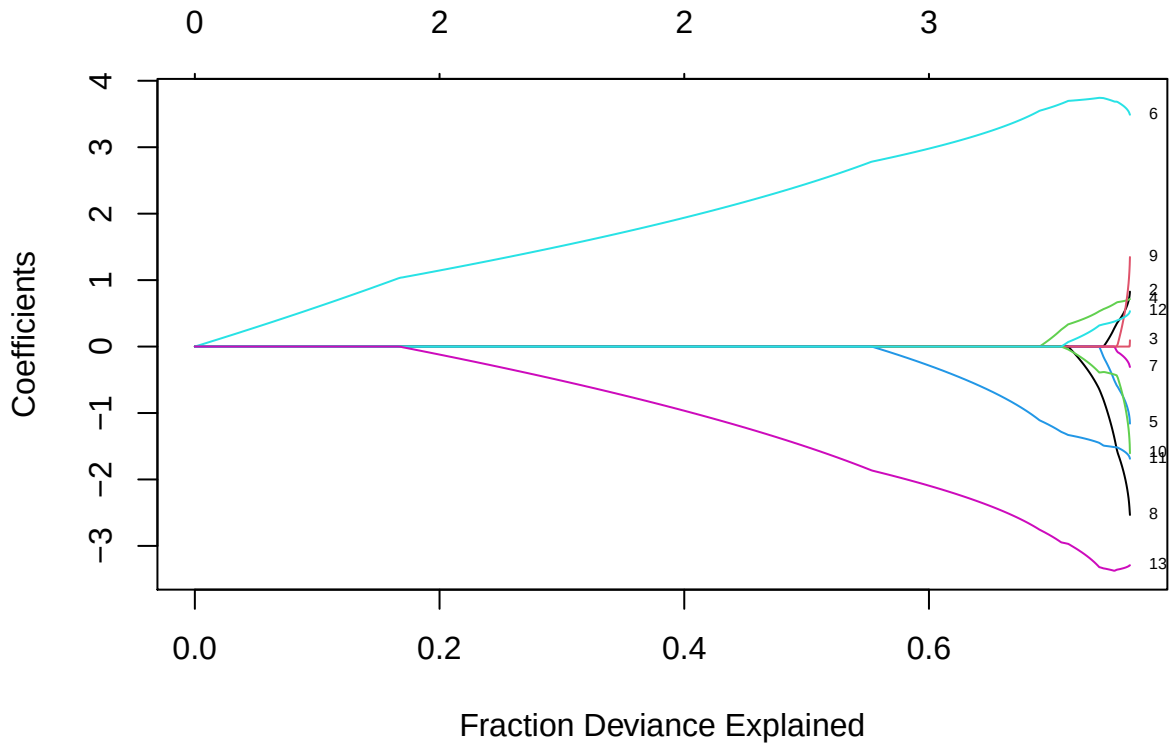
```
# Lambda in relazione con i coefficienti
plot(lassoFit, xvar = 'lambda', label=T)
```



Si osserva dal grafico che le variabili 6 e 13 vanno verso 0 più lentamente rispetto alle variabili 5 o 7. Ciò significa che la 6 e la 13 hanno più peso nel nostro modello e sono estremamente importanti, ma una volta che raggiungono lo 0 verranno rimosse dal modello.

```r
#Goodness of fit: utile per vedere la distribuzione della varianca in
#corrispondenza con le nostre feature
plot(lassoFit, xvar = 'dev', label = T)
```

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




```r
# Predicted Values
y_predicted_lasso <- predict(lassoFit, s=min(lambda.array), newx = xtest)

# SSE(sum of squared error), SST (sum of squared total)
sst <- sum((ytest - mean(ytest))^2)
sse <- sum((y_predicted_lasso - ytest)^2)


rsquare_lasso <- 1 - (sse/sst)
#rsquare_lasso indica la varianza del nostro modello che è circa 0.6.
```

Notiamo che più i coefficienti lambda aumentano, più feature decrementano verso la fine, ovvero vengono penalizzati dal lasso.

GROUP LASSO

Consideriamo un modello lineare con più predittori, alcuni dei quali categoriali. Un predittore categoriale con ℓ livelli sarà rappresentato nel modello da $\ell - 1$ variabili. Lasso ha solo la capacità di ridurre a zero i coefficienti di regressione individuali. Nel caso del predittore categoriale, questo ha poca interpretazione. Se il predittore categoriale non è rilevante per la risposta, tutte le variabili $\ell - 1$ devono essere rimosse dal modello.

Yuan e Lin hanno sviluppato il metodo Gruppo Lasso come estensione del Lasso per risolvere questo problema. Questa penalizzazione effettua selezione delle variabili considerando ciascuno degli gruppi di variabili per l'inclusione o l'esclusione nel modello. Spesso quando viene stimato un modello con una struttura di gruppo per il vettore parametro, l'obiettivo è quello di raggiungere sparsità a livello di gruppo e le entrate di $\hat{\beta}_{G_j}$ devono essere tutte zero o tutte non-zero. Questo obiettivo può essere raggiunto con la penalizzazione del gruppo lasso:

$$\lambda \sum_{j=1}^{q} m_j ||\beta_{G_j}||_2.$$

Il moltiplicatore m_j è un bilanciatore, usato quando ci sono gruppi con dimensioni veramente diverse. Tipicamente si sceglie:

$$m_j = \sqrt{T_j},$$

dove T_j denota la cardinalità di $|G_j|$.

Lo stimatore del Gruppo Lasso in un modello lineare o in un modello lineare generalizzato viene definito rispettivamente come:

$$\hat{\beta}(\lambda) = argmin_\beta Q_\lambda(\beta),$$

$$Q_\lambda(\beta) = n^{-1} \sum_{i=1}^{n} \rho_\beta(X_i, Y_i) + \lambda \sum_{j=1}^{q} m_j ||\beta_{G_j}||_2,$$

dove $\rho_\beta(x, y)$ è una funzione obiettivo convesso in β. Alcuni esempi di funzione obiettivo sono: $\rho_\beta(x, y) = |y - x\beta|^2$; $\rho_\beta(x, y) = -log\beta(p(y|x))$ con $p(\cdot|x)$ che denota la densità di $[Y|X = x]$. Di solito, quando si utilizza Gruppo Lasso, viene incluso un termine di intercetta non penalizzata facendo diventare lo stimatore rispettivamente:

$$\hat{\mu}(\lambda), \hat{\beta}(\lambda) = argmin_{\mu,\beta} Q_\lambda(\mu, \beta),$$

$$Q_\lambda(\mu, \beta) = n^{-1} \sum_{i=1}^{n} \rho_{\mu,\beta}(X_i, Y_i) + \lambda \sum_{i=1}^{q} m_j ||\beta_{G_j}||_2.$$

Lemma 1 Supponiamo che $\rho_\beta(X_i, Y_i) \geq C > -\infty \quad \forall \beta, X_i, Y_i (i = 1, \ldots, n)$ e che $\rho_\beta(X, Y)$ sia una funzione convessa in $\beta \ \ \forall X_i, Y_i (i = 1, \ldots, n)$. Allora, per $\lambda > 0$ e per $m_j > 0 \forall j$, si raggiunge il minimo nel problema di ottimizzazione.

Dimostrazione

Siccome $Q_\lambda(\beta)$ è continua e $Q_\lambda(\beta) \to \infty$ come $||(\beta_{G_1}, \ldots, \beta_{G_q})||_2 \to \infty$, viene raggiunto il minimo. □

L'assunzione di limitatezza nel lemma 1 è banale e vale per le funzioni di penalizzazione comunemente usate per la regressione o la classificazione (generalizzata).

Proprietà

Lo stimatore del gruppo Lasso ha le seguenti proprieta:

- in base al valore del parametro di regolarizzazione λ, i coefficienti stimati all'interno di un gruppo G_j soddisfano la seguente condizione: $(\hat{\beta_{G_j}})_r \equiv 0$ per tutti i componenti $r = 1, \ldots, T_j$ oppure, $(\hat{\beta_{G_j}})_r \neq 0$ per tutti i componenti $r = 1, \ldots, T_j$. Questa è una conseguenza della non-differenziabilità della funzione $\sqrt{\cdot}$ in 0. Inoltre, nei casi di gruppi semplici, costituiti da singleton $G_j = j \quad \forall j = 1, \ldots, q = p$, e dove $m_j = T_j \equiv 1$, la funzione di penalizzazione coincide con la penalizzazione standard del Lasso.
- La penalizzazione del Group Lasso è **invariante sotto trasformazioni ortonormali** all'interno dei gruppi.

Spesso viene scelto qualsiasi base ortonormale per la parametrizzazione che porta a sotto-matrici ortonormali $\mathbf{X_{G_j}^T X_{G_j}}$ per ogni gruppo G_j, dove $\mathbf{X_{G_j}}$ è la sottomatrice $n \times T_j$ di $\mathbf{X}$, le colonne della quale corrispondono a G_j. Quest'ultima fornisce dei vantaggi computazionali, ma bisogna sempre considerare che in generale, lo stimatore dipende dagli eventuali parametrizzazioni non-ortonormali.

Lo stimatore del gruppo lasso ha delle proprietà qualitative simili al Lasso:

Mostra una buona precisione per la previsione e la stima dei parametri, grazie alla proprietà di selezionare variabili a livello di gruppo. Ovvero, tutti i gruppi rilevanti con vettore parametro $\beta_G \neq 0$ vengono stimati come gruppi attivi con il corrispondente vettore parametro $\hat{\beta}_G \neq 0$.

La penalizzazione del Gruppo Lasso generalizzato

La penalizzazione del Gruppo Lasso è definita:

$$\lambda \sum_{j=1}^{q} m_j ||\beta_{G_j}||_2 = \lambda \sum_{j=1}^{q} m_j \sqrt{\beta_{G_j}^T \beta_{G_j}}.$$

In alcune applicazioni però, si richiede una penalizzazione della forma:

$$\lambda \sum_{j=1}^{q} m_j \sqrt{\beta_{G_j}^T A_j \beta_{G_j}},$$

dove A_j sono delle matrici $T_j \times T_j$ definite positive. Grazie al fatto che A_j è definita positiva, si può parametrizzare ancora: $\tilde{\beta_{G_j}} = A_j^{1/2} \beta_{G_j}$, e quindi, sorge una penalità di gruppo Lasso normale della forma:

$$\lambda \sum_{j=1}^{q} m_j ||\tilde{\beta}_{G_j}||_2.$$

La matrice $A_j^{1/2}$ può essere ottenuta usando ad esempio la decomposizione di Cholesky $A_j = R_j^T R_j$, dove R_j sono matrice quadrate ed $A_j^{1/2} = R_j$. Occorre ri-parametrizzare anche la parte del modello lineare (generalizzato):

$$\mathbf{X}\beta = \sum_{\mathbf{j=1}}^{\mathbf{q}} \mathbf{X_{G_j}} \beta_{\mathbf{G_j}}.$$

Lo stimatore del Gruppo Lasso generalizzato in un modello lineare e dunque definito come:

$$\hat{\beta} = argmin_\beta (||\mathbf{Y} - \mathbf{X}\beta||_{\mathbf{2}}^{\mathbf{2}}/\mathbf{n} + \lambda \sum_{\mathbf{j=1}}^{\mathbf{q}} \mathbf{m_j} \sqrt{\beta_{\mathbf{G_j}}^{\mathbf{T}} \mathbf{A_j} \beta_{\mathbf{G_j}}})$$

Equivalentemente si ha:

$$\tilde{\beta_{G_j}} = A_j^{-1/2}\hat{\tilde{\beta}}_{G_j},$$

$$\hat{\tilde{\beta}} = argmin_{\tilde{\beta}}(|\mathbf{Y} - \sum_{\mathbf{j=1}}^{\mathbf{q}} \tilde{\mathbf{X}}_{\mathbf{G_j}}\tilde{\beta}_{\mathbf{G_j}}||_{\mathbf{2}}^{\mathbf{2}}/\mathbf{n} + \lambda \sum_{\mathbf{j=1}}^{\mathbf{q}} \mathbf{m_j}||\tilde{\beta}_{\mathbf{G_j}}||_{\mathbf{2}}).$$

In seguito un esempio di utilizzo della penalizzazione gruppo lasso

Dataset bardet

Bardet è un dataset che contiene 120 campioni con 100 predittori (espansi da 20 geni utilizzando 5 basi B-spline, come descritto in Yang, Y. e Zou, H. (2015)).

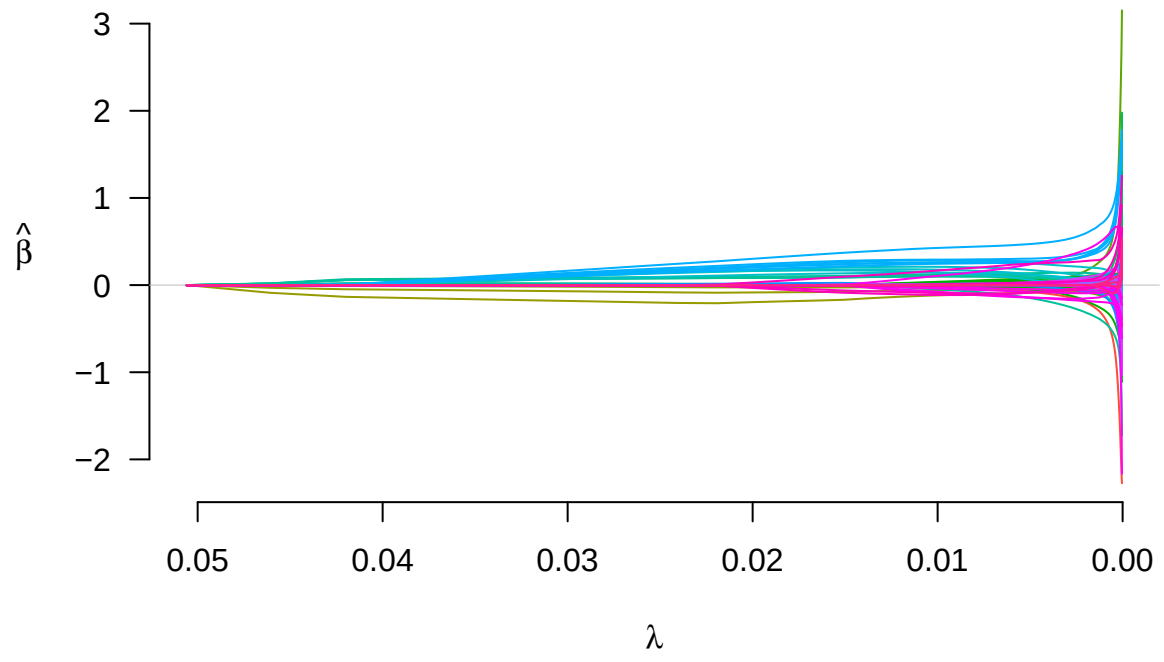
```
set.seed(123)
library(grpreg)
```

```
## Warning: package 'grpreg' was built under R version 4.0.5
```

```
bardet <- read.csv("bardet.csv")
group1 <- rep(1:20, each=5)
dim(bardet)
```

```
## [1] 120 101
```

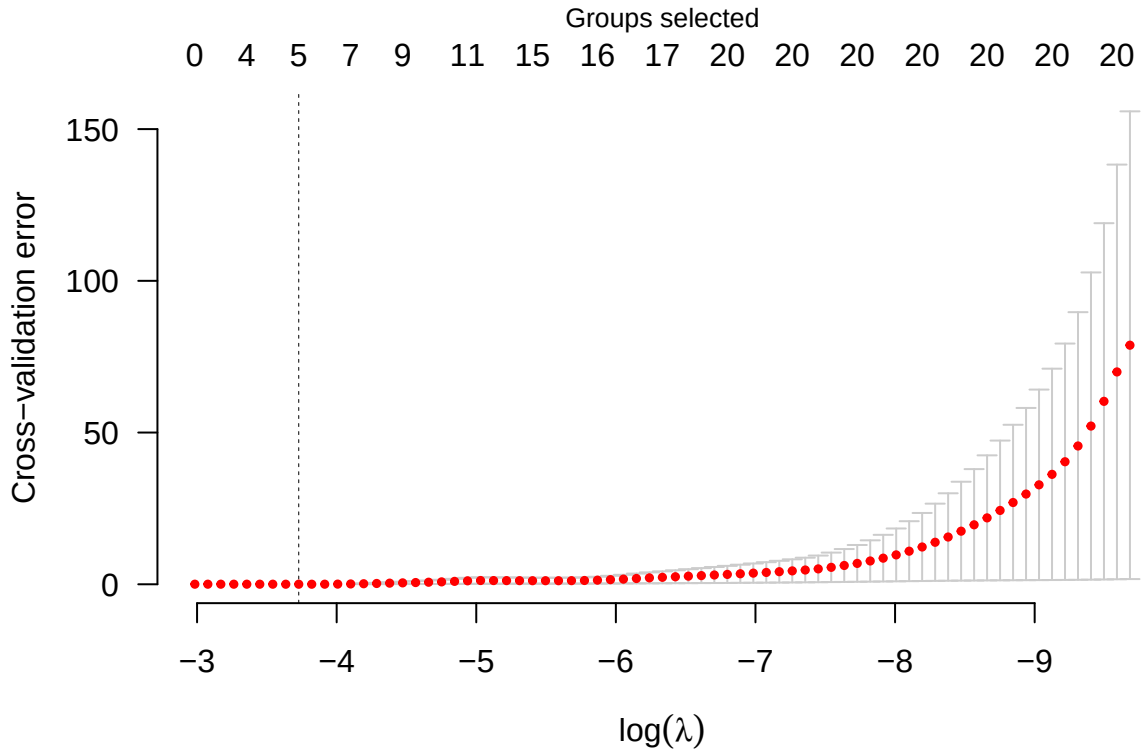
```
fit <- grpreg(bardet[,-1], bardet$Y, group1, penalty ="grLasso")
plot(fit)
```



```
coef(fit, lambda = 0.03)
```

```
##   (Intercept)            X1            X2            X3            X4
##  8.1438890477  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##            X5            X6            X7            X8            X9
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X10           X11           X12           X13           X14
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X15           X16           X17           X18           X19
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X20           X21           X22           X23           X24
##  0.0000000000 -0.0184176366 -0.0096718123 -0.0707926448 -0.0066209284
##           X25           X26           X27           X28           X29
## -0.1819705554  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X30           X31           X32           X33           X34
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X35           X36           X37           X38           X39
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X40           X41           X42           X43           X44
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X45           X46           X47           X48           X49
##  0.0000000000  0.0203444929  0.0742885085  0.0709400111  0.0848818199
##           X50           X51           X52           X53           X54
##  0.0718342645  0.1102619926  0.0686899478  0.1064836267  0.1173297411
##           X55           X56           X57           X58           X59
##  0.1002532155  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X60           X61           X62           X63           X64
##  0.0000000000  0.1392457931  0.0998493873  0.1169540317  0.1252844903
##           X65           X66           X67           X68           X69
##  0.1640337431  0.0004270203  0.0015217266  0.0015146601  0.0018883848
##           X70           X71           X72           X73           X74
##  0.0014696660  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X75           X76           X77           X78           X79
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X80           X81           X82           X83           X84
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X85           X86           X87           X88           X89
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X90           X91           X92           X93           X94
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##           X95           X96           X97           X98           X99
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000  0.0000000000
##          X100
##  0.0000000000
```

```
cvfit <- cv.grpreg(bardet[,-1], bardet$Y, group1, penalty ="grLasso")
plot(cvfit)
```




```
cvfit$lambda.min
```

```
## [1] 0.02403177
```

```
predict(fit, as.matrix(bardet[1,-1]), type = "response", lambda = cvfit$lambda.min)
```

```
## [1] 8.384639
```

Group Lasso adattivo

Come ribadito precedentemente, il gruppo lasso è un'estensione del lasso e quindi, ci si aspetta che il gruppo lasso soffra dell'inefficienza di stima e dell'incongruenza di selezione allo stesso modo del lasso. Come rimedio, è stato proposto il metodo del gruppo lasso addattivo.

È simile al lasso addattivo ma ha la capacità di selezionare le variabili in modo raggruppato. E' stato provato teoricamente che questo stimatore è in grado di identificare il vero modello in modo coerente ed efficiente.

Modello

Siano $(x_1, y_1), ..., (x_n, y_n)$ n vettori casuali indipendenti e identicamente distribuiti, cioè tutte le variabili casuali hanno la stessa distribuzione di probabilità e sono tutte indipendenti. Sia $y_i \in \mathbb{R}^1$ il vettore risposta e sia $x_i \in \mathbb{R}^d$ il predittore d-dimensionale associato. Inoltre, si assume che x_i può essere raggrupato in p fattori come $x_i = (x_{i1}^T, ..., x_{ip}^T)^T)$, dove $x_{ij} = (x_{ij1}, ..., x_{ijd_j})^T) \in \mathbb{R}^{d_j}$ rappresenta un gruppo di d_j variabili.

In una tale situazione, è praticamente più significativo identificare fattori importanti invece di variabili individuali (Yuan and Lin, 2006). In questa sezione, si utilizzano i termini fattore e gruppo in modo intercambiabile per indicare il raggruppamento delle variabili.Ad esempio, una variabile categoriale può essere rappresentata da alcune variabili indicatore le quali formano un fattore.

Si consideri un modello di regressione lineare:

$$y_i = \sum_{j=1}^{p} \beta_j x_{ij}^T + \epsilon_i = x_i^T \beta + \epsilon_i,$$

dove $\beta_j = (\beta_{j1}, ..., \beta_{jd_j})^T \in \mathbb{R}^{d_j}$ è il vettore del coefficiente di regressione associato al fattore j-esimo e β viene definito come $\beta_j = (\beta_1^T, ..., \beta_p^T)^T$. Senza perdita di generalità, si assume che solo i primi fattori $p_0 \leq p$ sono rilenvanti. Ciò significa che si assume $||\beta_j|| = 0$ per $j > p_0$.

Consideriamo la funzione obiettivo dei minimi quadrati penalizzati con il gruppo lasso:

$$\sum_{j=1}^{n} \frac{1}{2}(y - \sum_{j=1}^{p} \beta_j x_{ij}^T)^2 + n\lambda \sum_{j=1}^{p} ||\beta_j||.$$

Si fa notare che se il numero di variabili contenute in ogni fattore è effettivamente uno ($d_j = 1$), la funzione obiettivo del gruppo lasso si riduce al lasso normale. Tuttavia, se esistono alcuni fattori contenenti più di una variabile, lo stimatore gLasso descritto sopra, ha la capacità di selezionare quelle variabili in modo raggruppato. Come si può vedere, gLasso penalizza ogni fattore in modo molto simile al solito lasso. In altre parole, lo stesso parametro di regolarizzazione viene utilizzato per ogni fattore senza valutare la loro importanza relativa. In una tipica impostazione di regressione lineare, è stato dimostrato che una penalità così eccessiva applicata alle variabili rilevanti può degradare l'efficienza della stima (Fan and Li, 2001) e può influire sulla coerenza della selezione. Per superare tale limitazione, è stao pensato di introdurre il gruppo lasso adattivo.

$$Q(\beta) = \sum_{j=1}^{n} \frac{1}{2}(y - \sum_{j=1}^{p} \beta_j x_{ij}^T)^2 + n \sum_{j=1}^{p} \lambda ||\beta_j||.$$

Quindi, ridurre al minimo la funzione obiettivo produce lo stimatore $\hat{\beta}$ del gruppo lasso adattivo. Come si può vedere, la differenza fondamentale tra il gLasso adattivo ed il gLasso è che il gLasso adattivo consente l'utilizzo dei diversi parametri di regolarizzazione per diversi fattori. Una tale flessibilità a sua volta produce diverse quantità di restringimento per diversi fattori. Intuitivamente, se una quantità relativamente maggiore di restringimento viene applicata ai coefficienti pari a zero e una quantità relativamente minore viene utilizzata per i coefficienti diversi da zero, è possibile ottenere uno stimatore con una migliore efficienza. Gli studi hanno dimostrato che lo stimatore del gruppo lasso adattivo può effettivamente identificare il vero modello in modo coerente e lo stimatore risultante è efficiente.

Algoritmi per il GROUP LASSO

Lo stimatore $\hat{\beta}(\lambda)$ del Gruppo Lasso è dato dalla minimizzazione della funzione obiettivo convessa:

$$Q_\lambda(\beta) = n^{-1} \sum_{i=1}^{n} \rho_\beta(X_i, Y_i) + \lambda \sum_{j=1}^{q} m_j ||\beta_{G_j}||_2,$$

dove $\rho_\beta(X_i, Y_i)$ indica una funzione obiettivo convessa in β.

- Per lo scarto quadratico, si considera:

$$\rho_\beta(x, y) = |y - x\beta|^2, (y \in \mathbb{R}, x \in \mathbb{R}^p),$$

- Per la regressione logistica in un problema di classificazione binaria, si considera:

$$\rho_\beta(x, y) = -y f_\beta(x) + log(1 + exp(f_\beta(x))), (y \in \{0, 1\}, x \in \mathbb{R}^p),$$

$$f_\beta(x) = x\beta$$

In entrambi questi esempi, la fuzione obiettivo è nella forma: $\rho_\beta(x, y) = \rho(f_\beta(x), y)$ come composizione di una funzione lineare su β ed una funzione convessa su f : $f \mapsto \rho(f, y)\ \forall y$.

Indichiamo nel seguito il rischio empirico come:

$$\rho(\beta) = n^{-1} \sum_{i=1}^{n} \rho_\beta(X_i, Y_i).$$

La versione penalizzata si decompone quindi come:

$$Q_\lambda(\beta) = \rho(\beta) + \lambda \sum_{j=1}^{q} m_j ||\beta_{G_j}||_2.$$

Block co-ordinate descent

E' un algoritmo iterativo proposto da Yuan e Lin (2006), noto anche come metodo di tipo Gauss-Seidel, che risolve in sequenza un sistema di equazioni non lineari che corrispondono a una minimizzazione a gruppi della somma dei quadrati residua penalizzata.

L'algoritmo inizia assumendo un vettore parametro iniziale: $\beta^{[0]}$. Ad ogni passo m, per ogni gruppo di variabili β_{G_j}, l'algoritmo ottimizza la funzione obiettivo rispetto al corrispondente gruppo G_j mantenendo tutti i parametri tranne quelli correnti corrispondenti ad un gruppo fisso. Denotiamo con β_{-G_j} il vettore β le cui componenti in G_j sono impostate a zero:

$$(\beta_{-G_j})_k = \begin{cases} \beta_k, & k \notin G_j \\ 0, & k \in G_j \end{cases}$$

Algoritmo **Block Coordinate Descent**

- 1: Sia $\beta^{[0]} \in \mathbb{R}^p$ il vettore parametro iniziale. Poniamo $m = 0$.
- 2: Ripetere
- 3: $m \leftarrow m + 1$.
- 4: $se \quad ||(-\nabla_\rho(\beta^{[m-1]}_{-G_j})_{G_j})||_2 \leq \lambda m_j$
  $\beta^{[m]}_{G_j} = 0$
  $altrimenti$
  $\beta^{[m]}_{G_j} = argmin_{\beta_{G_j}} Q_\lambda(\beta^{[m-1]}_{+G_j})$
  $fine$
- 5: Ripetere il passo 2 finchè non si incontra un criterio di convergenza

Figure 3: Algoritmo del Group Lasso: Block coordinate descent usato per effettuare selezione di variabili.

Nella fig.2 j rappresenta l'indice itarativo nelle le coordinate del blocco $\{1, \ldots, q\}$.

Denotiamo la matrice $\mathbf{X_{G_j}}$ $n \times T_j$ costituito dalle colonne della matrice $\mathbf{X}$ corrispondenti ai predittori del gruppo G_j. Per semplicità notazionale, verrà denotata con $\hat{\beta}$.

La norma ℓ_2 del gradiente negativo e la corrispondente disuguaglianza sono:

- per lo scarto quadratico:

$$||2n^{-1}\mathbf{X^T_{G_j}}(\mathbf{Y} - \beta^{[\mathbf{m-1}]}_{-\mathbf{G_J}})||_\mathbf{2} \leq \lambda \mathbf{m_j}$$

- per la regressione logistica:

$$||n^{-1}\mathbf{X^T_{G_j}}(\mathbf{Y} - \pi_{\beta^{[\mathbf{m-1}]}_{-\mathbf{G_J}}})||_\mathbf{2} \leq \lambda \mathbf{m_j}$$

Il passo 4 di questo algoritmo è un controllo esplicito se il minimo si trova nel punto non-differenziabile con $\beta_{G_j} \equiv 0$. In caso contrario, possiamo utilizzare un minimizzatore numerico standard, ad esempio, un algoritmo di tipo gradiente, per trovare la soluzione ottimale rispetto a β_{G_j}.

Osservazioni sull'algoritmo

Quando si cicla tra i blocchi di coordinate (o gruppi), ci si limita al set attivo corrente e si visitano solo "raramente" i blocchi (o gruppi) rimanenti, come ad esempio, dopo 10 iterazioni, aggiornare il set attivo. Ciò è particolarmente utile per le impostazioni ad altissime dimensioni riducendo notevolmente il tempo di calcolo.

Lo svantaggio principale dell'algoritmo è per i casi diversi dallo scarto quadratico in cui le minimizzazioni a blocchi dei gruppi attivi del passo 4 devono essere eseguite numericamente. Tuttavia, per problemi di piccole e moderate dimensioni ciò risulta essere sufficientemente veloce. È possibile migliorare l'efficienza computazionale sostituendo un'esatta minimizzazione a livello di gruppo nel passo 4 con un'approssimazione appropriata il cui calcolo è esplicito.

Block co-ordinate gradient descent (BCGD)

E' un algoritmo proposto da Tseng and Yun (2009) che ha come idea principale la combinazione tra l'approssimazione quadratica del log di verosimiglianza con una ricerca lineare aggiuntiva.

Usando lo sviluppo di Taylor del secondo ordine al $\beta^{[m]}$, la stima nella m-esima iterazione, e sostituendo la Hessiana del rischio empirico $\rho(\beta)$ con una matrice adatta $H^{[m]}$, si definisce:

$$\begin{aligned} M_\lambda^{[m]}(d) &= \rho(\beta^{[m]}) + d^T \nabla\rho(\beta^{[m]}) + \frac{1}{2} d^T H^{[m]} d + \lambda \sum_{j=1}^{q} m_j ||\beta_{G_j}^{[m]} + d_{G_j}||_2 \\ &\approx Q_\lambda(\beta^{[m]} + d), \end{aligned}$$

dove $d \in \mathbb{R}^p$.

Consideriamo ora la minimizzazione di $M_\lambda^{[m]}(\cdot)$ rispetto al j-esimo gruppo di parametri. Ciò significa che ci limitiamo a considerare i vettori d con $d_k = 0, \quad k \notin G_j$. Inoltre, assumiamo che la sottomatrice quadrata corrispondente $H_{G_j,G_j}^{[m]}$, $T_j \times T_j$ sia uguale a $h_j^{[m]} \cdot I_{T_j}, \quad h_j^{[m]} \in \mathbb{R}$.

- Se $||\nabla_\rho(\beta^{[m]})_{G_j} - h_j^{[m]} \beta_{G_j}^{[m]}||_2 \leq \lambda m_j$, la minimizzazione di $M_\lambda^{[m]}(d)$ rispetto al gruppo G_j è: $d_{G_j}^{[m]} = -\beta_{G_j}^{[m]}$
- Altrimenti la minimizzazione di $M_\lambda^{[m]}(d)$ rispetto al gruppo G_j sarà:

$$d_{G_j}^{[m]} = -\frac{1}{h_{G_j}^{[m]}} \left\{ \nabla\rho(\beta^{[m]})_{G_j} - \lambda m_j \frac{\nabla\rho(\beta^{[m]})_{G_j} - h_{G_j}^{[m]} \beta_{G_j}^{[m]}}{||\nabla\rho(\beta^{[m]})_{G_j} - h_{G_j}^{[m]} \beta_{G_j}^{[m]}||_2} \right\}$$

Se $d_{G_j}^{[m]} \neq 0$, occorre aggiornare il vettore parametro usando la regola di Armijo, ottenendo: $\beta^{[m+1]} = \beta^{[m]} + \alpha_j^{[m]} d_{G_j}^{[m]}$.

Quando si minimizza $M_\lambda^{[m]}(\cdot)$ rispetto ad un gruppo con una penalizzazione, bisogna prima assicurarsi che il minimo non sia in un punto non-differenziabile. Per una intercetta β_0 non penalizzante, non c'è bisogno di tutto l'algoritmo e la soluzione può essere data direttamente da:

$$d_0^{[m]} 0 - \frac{1}{h_0^{[m]}} \nabla_\rho(\beta^{[m]})_0.$$

Algoritmo **Block Coordinate Gradient Descent**

- 1: Sia $\beta^{[0]} \in \mathbb{R}^p$ il vettore parametro iniziale. Poniamo $m = 0$.
- 2: Ripetere
- 3: $m \leftarrow m + 1$.
- 4: $H^{[m-1]}_{G_j,G_j} = h_j^{[m-1]} \cdot I_{T_j} = h_j(\beta^{[m-1]}) \cdot I_{T_j}$
  $d^{[m-1]} \leftarrow argmin_{d,d_{G_k}=0} M_\lambda^{[m-1]}(d)$
  $d^{[m-1]}_{G_j} = (d^{[m-1]})_{G_j}$
  $se \;\; d^{[m-1]}_{G_j} \neq 0$
  $\alpha_j^{[m-1]} \leftarrow \;\; riga\ di\ ricerca,$
  $\beta^{[m]} \leftarrow \beta^{[m-1]} + \alpha_j^{[m-1]} d^{[m-1]}_{G_j}$
- 5: Ripetere il passo 2 finchè non si incontra un criterio di convergenza

Figure 4: Algoritmo del Group Lasso: Block coordinate gradient descent.

Nella fig.3 j rappresenta l'indice itarativo nelle le coordinate del blocco $\{1, \ldots, q\}$.

Proposizone

Assumiamo che la funzione obiettivo $\rho_\beta(X_i, Y_i) \geq C > -\infty, \; \forall \beta, X_i, Y_i, i = 1, \ldots, n$ sia continua e differenziabile rispetto a β e che il rischio empirico $\rho(\beta)$ sia convesso. Denotiamo con $\hat{\beta}^{[m]}$ il vettore parametro ottenuto dall'algoritmo BCGD dopo m iterazioni.Se $H^{[m]}_{G_j,G_j}$ viene considerata nello stesso modo come abbiamo fatto nell'impostare il BCGD, allora ogni punto della sequenza $\{\hat{\beta}^{[m]}\}_{m\geq 0}$ è un punto di minimo di $Q_\lambda(\cdot)$. (Meier et al. 2008, Proposizione 2).

L'algoritmo BCGD può essere applicato al gruppo lasso anche in altri modelli lineari generalizzati in cui la risposta Y ha una distribuzione dalla famiglia esponenziale.

R PACKAGE

CRAN - Package 'grplasso'

L'autore di questo package è Lukas Meier, il quale fornisce la soluzione di un problema di gruppo lasso per un modello di tipo grpl.model.

Quando si utilizza grplasso.formula, il raggruppamento delle variabili è derivato dal tipo delle variabili: Le variabili fittizie di un fattore verranno automaticamente trattate come un gruppo. Il processo di ottimizzazione inizia utilizzando il primo componente di lambda come parametro di penalità λ e con i valori iniziali definiti in coef.init per il vettore del parametro. Una volta adattato, il componente successivo di lambda è considerato come parametro di penalità con valori iniziali definiti come il vettore del coefficiente (adattato) basato sulla componente precedente di lambda.

Utilizzo in R

grplasso(x, . . .)

Metodo per la classe 'formula'

grplasso(formula, nonpen = ~ 1, data, weights, subset, na.action, lambda, coef.init, penscale = sqrt, model = LogReg(), center = TRUE, standardize = TRUE,control = grpl.control(), contrasts = NULL, . . .)

Metodo di default

grplasso(x, y, index, weights = rep(1, length(y)), offset = rep(0, length(y)), lambda, coef.init = rep(0, ncol(x)), penscale = sqrt, model = LogReg(), center = TRUE, standardize = TRUE, control = grpl.control(), . . .)

Parametri di input

- **X**: Matrice di progetto (include l'intercetta)
- **y**: Vettore risposta
- **formula**: Formula delle variabili di penalizzazione. La risposta si deve trovare a sinistra del
- **nonpen**: Formula delle variabili di non penalizzazione. Viene aggiunto al argomento formula descritto precedentemente e non necessita di avere la risposta a sinistra del
- **data**: data.frame contiene le variabili del modello
- **index**: Vettore che definisce il raggruppamento delle variabili. Le componenti che condividono lo stesso numero costruiscono un gruppo. I coefficienti non-penalizzati sono segnati con NA
- **weights**: Vettore dei pesi di osservazione
- **subset**: Vettore opzionale che specifica un sottoinsieme di osservazioni da utilizzare nel processo di adattamento.
- **subset**: Vettore dei pesi di osservazione
- **na.action**: Funzione che indica cosa dovrebbe accadere quando i dati contengono variabili "NA".
- **offset**: Vettore di valori di offset; deve avere la stessa lunghezza del vettore di risposta.
- **lambda**: Vettore dei parametri di penalizzazione. L'ottimizzazione inizia con la prima componente.
- **coef.init**: Vettore iniziale delle stime dei parametri corrispondenti al primo componente nel vettore lambda.
- **penscale**: Funzione di riscalaggio per adattare il valore del parametro di penalizzazione ai gradi di libertà del gruppo di parametri
- **model**: Oggetto della classe grpl.model che implementa log-verosimiglianza negativa, gradiente, la Hessiana etc..
- **center**: Variabile booleana. Se true, le colonne della matrice di progetto saranno centrate (tranne una possibile colonna di intercettazione).

- **standardize**: Variabile booleana. Se vero, la matrice di progetto sarà ortonormalizzata a blocchi in modo tale che per ogni blocco $X^T X = n_1$ (*dopo* un possibile centraggio).
- **control**: Opzioni per l'algoritmo di adattamento
- **contrasts**: Una lista opzionale.
- **...**: Argomenti aggiuntivi da passare alle funzioni definite nel modello.

Output

Viene restituito un oggetto grplasso, per il quale esistono metodi *coef*, *print*, *plot* e *predict*.

- **coefficients**: coefficienti rispetto alle variabili di input *originali* (anche se standardize = TRUE viene utilizzato per l'adattamento).
- **lambda**: Vettore dei valori lambda dove sono stati calcolati i coefficienti
- **index**: Vettore indice di raggruppamento

Implementazione in R

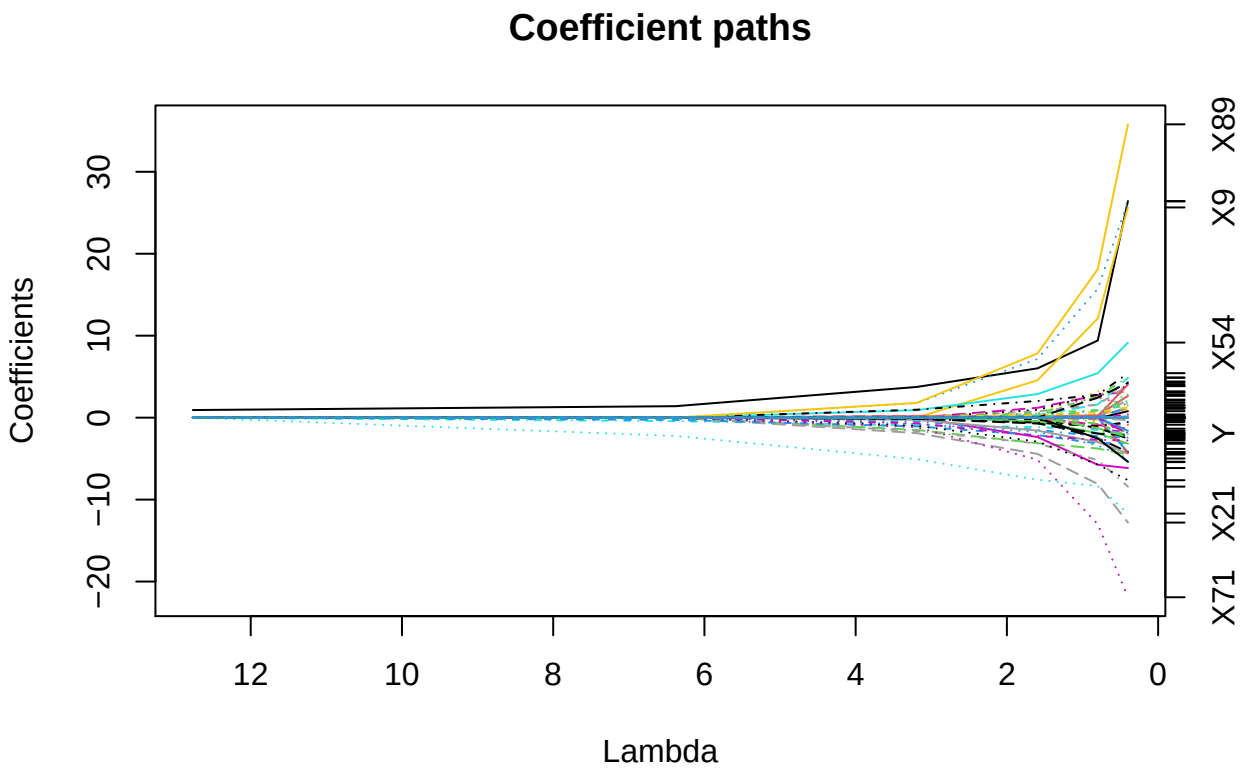
Nel seguito verrà utilizzato nuovamente il dataset Bardet, applicando direttamente funzione grplasso.

```
library(grplasso)
set.seed(999)
bardet <- read.csv("bardet.csv")

X = cbind(1,as.matrix(bardet))

y = X[,1] + runif(10)*0.1

prob <- 1 / (1 + exp(-X))
mean(pmin(prob, 1 - prob)) ## Rischio Bayesiano
```

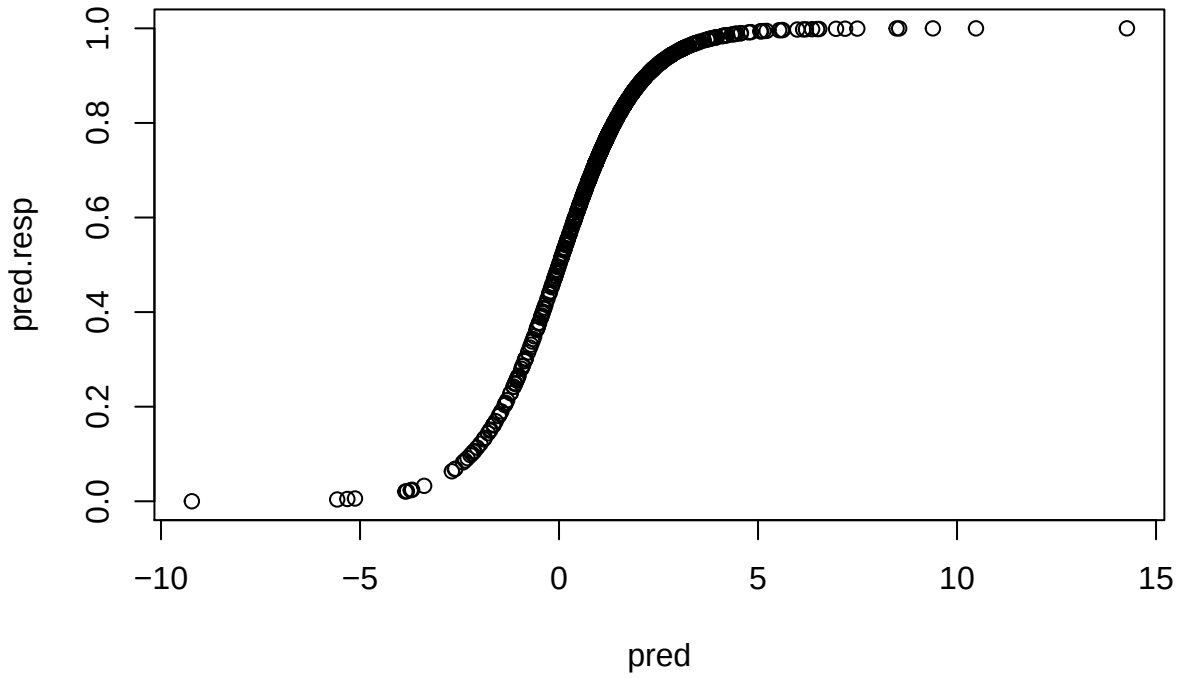
```
## [1] 0.4463134
```

```
y <- rbinom(nrow(X), size = 1, prob = prob) ## Vettore di risposta binario

## Creazione dei gruppi
index <- c()
numberOfIter <- (ncol(X)/2)-1
for(i in 1:numberOfIter) {
  index <- c( index, i, i)
}

index<-c(NA, 0, index)
##Valorizzazione del parametro lambda da utilizzare
lambda <- lambdamax(X, y = y, index = index, penscale = sqrt,
                    model = LogReg()) * 0.5^(0:5)

## Si applica il metodo grplasso al modello
fit <- grplasso(X, y = y, index = index, lambda = lambda, model = LogReg(),
                penscale = sqrt,
                control = grpl.control(update.hess = "lambda", trace = 0))
```

```
plot(fit)
```




```
# pred è una matrice le cui colonne corrispondono ai diversi valori del parametro
# di penalità lambda dell'oggetto grplasso.
pred <- predict(fit)
pred.resp <- predict(fit, type = "response")

## I seguenti punti dovrebbero trovarsi sulla curva sigmoidea
plot(pred, pred.resp)
```


pred.resp
0.0
0.2
0.4
0.6
0.8
1.0
−10
−5
0
5
10
15
pred

REFERENZE

(1) Lukas Meier, Sara van de Geer and Peter Buhlmann (2008), The Group Lasso for Logistic Regression, Journal of the Royal Statistical Society, 70 (1), 53 - 71

(2) CRAN - Package 'grplasso'

- https://cran.r-project.org/web/packages/grplasso/grplasso.pdf

(3) http://www.columbia.edu/~my2550/papers/glasso.final.pdf

(4) Statistics for High-Dimensional Data, Methods, Theory and Applications, Peter Buhlmann, Sara van de Geer.

(5) Computational Statistics & Data Analysis, Volume 52, Issue 12, 15 August 2008

(6) https://github.com/raoy/data/blob/master/bardet.rda