Machine Learning

Università Roma Tre
Dipartimento di Ingegneria
Anno Accademico 2021 - 2022

Dimostrazioni Formali Lasso

Sommario

Dimostrazione della formula di aggiornamento dei coefficienti nell'algoritmo coordinate descent per LASSO

Ottimizzazione Lasso

- Come sappiamo, la Funzione Obiettivo per il Lasso da ottimizzare mediante Coordinate Descent è la seguente:

$$\mathrm{RSS}(\mathbf{w}) + \lambda \cdot \|\mathbf{w}\|_1 = \sum_{i=1}^{N} [y_i - \sum_{j=0}^{D} w_j \phi_j(\mathbf{x}_i)]^2 + \lambda \sum_{j=0}^{D} |w_j|$$

- Vediamo come calcolare le derivate parziali dei due termini presenti nell'espressione rispetto ai pesi w_j.

Derivazione del termine RSS

$$\mathrm{RSS}(\mathbf{w}) + \lambda \cdot \|\mathbf{w}\|_1 = \boxed{\sum_{i=1}^{N} [y_i - \sum_{j=0}^{D} w_j \phi_j(\mathbf{x}_i)]^2} + \lambda \sum_{j=0}^{D} |w_j|$$

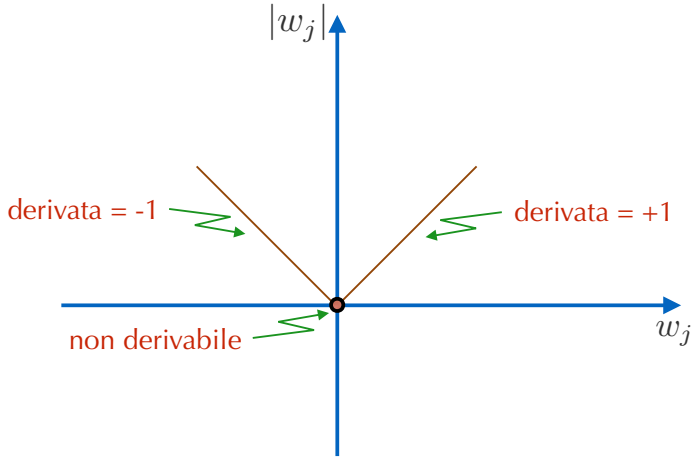
$$\begin{aligned}
\frac{\partial \mathrm{RSS}(\mathbf{w})}{\partial w_j} &= -2 \sum_{i=1}^{N} \phi_j(\mathbf{x}_i)[y_i - \hat{y}_i(\mathbf{w})] = -2 \sum_{i=1}^{N} \phi_j(\mathbf{x}_i)[y_i - \sum_{j=0}^{D} w_j \phi_j(\mathbf{x}_i)] = \\
&= -2 \sum_{i=1}^{N} \phi_j(\mathbf{x}_i)[y_i - \sum_{k \neq j} w_k \phi_k(\mathbf{x}_i) - w_j \phi_j(\mathbf{x}_i)] = \\
&= -2 \overbrace{\sum_{i=1}^{N} \phi_j(\mathbf{x}_i)[y_i - \underbrace{\sum_{k \neq j} w_k \phi_k(\mathbf{x}_i)}_{\text{prediz. senza } \phi_j}]}^{\rho_j} + 2 w_j \overbrace{\sum_{i=1}^{N} \phi_j^2(\mathbf{x}_i)}^{\triangleq z_j} = \\
&= -2\rho_j + 2 w_j z_j
\end{aligned}$$

Derivazione del termine L_1 penalty

- In questo caso c'è il problema del calcolo della derivata parziale:

$$\mathrm{RSS}(\mathbf{w}) + \lambda \cdot \|\mathbf{w}\|_1 = \sum_{i=1}^{N} [y_i - \sum_{j=0}^{D} w_j \phi_j(\mathbf{x}_i)]^2 + \boxed{\lambda \sum_{j=0}^{D} |w_j|}$$

$$\lambda \cdot \frac{\partial |w_j|}{\partial w_j} = ?$$



Subgradiente di Funzioni Convesse

- I metodi che conosciamo (e.g., Gradient Descent, Coordinate Descent) richiedono che la funzione da ottimizzare sia differenziabile.
- E' possibile però generalizzare la discussione andando al di là delle funzioni differenziabili.
- E' possibile ad esempio mostrare come i precedenti algoritmi possano essere applicati anche per funzioni non differenziabili, utilizzando il subgradiente anziché il gradiente.

Subgradiente di Funzioni Convesse

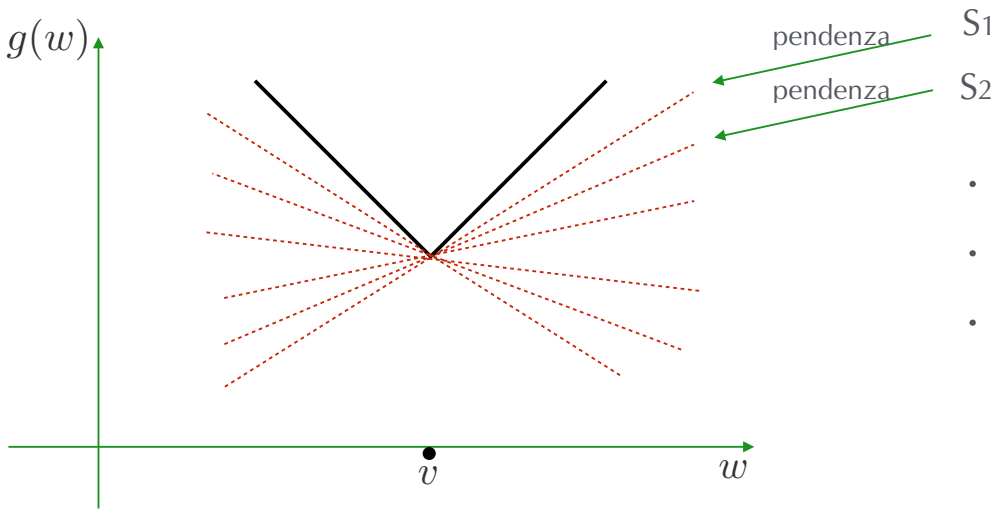
Un vettore ***S*** che soddisfa la:

$$g(\mathbf{w}) \geq g(\mathbf{v}) + \mathbf{S}^T(\mathbf{w} - \mathbf{v})$$

è detto subgradiente di g in **v**.

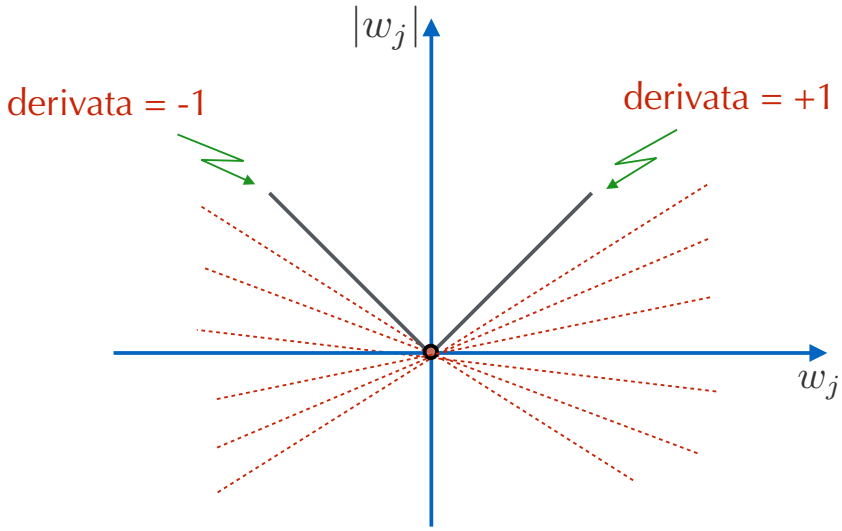
L'insieme dei subgradienti di g in **v** è chiamato "differential set" e indicato: $\partial g(\mathbf{v})$

$g : \mathbb{R} \rightarrow \mathbb{R}$



Subgradiente della funzione Valore Assoluto

- Nel punto non derivabile della funzione "valore assoluto" i subgradienti variano da -1 a +1:



Subgradiente della funzione Valore Assoluto

- Il “differential set” è dunque il seguente per i vari punti:

$$\partial_{w_j} |w_j| = \begin{cases} \{-1\} & \text{se} \quad w_j < 0 \\ [-1, 1] & \text{se} \quad w_j = 0 \\ \{1\} & \text{se} \quad w_j > 0 \end{cases}$$

Subgradiente di L_1 term

- Nel caso del Lasso abbiamo:

$$\lambda \cdot \partial_{w_j} |w_j| = \begin{cases} -\lambda & \text{se} \quad w_j < 0 \\ [-\lambda, \lambda] & \text{se} \quad w_j = 0 \\ \lambda & \text{se} \quad w_j > 0 \end{cases}$$

Differential set della funzione di costo Lasso

- Il differential set rispetto al generico peso w_j è pertanto il seguente:

$$\partial_{w_j}[\text{costo\_lasso}] = \overbrace{2z_j w_j - 2\rho_j}^{\text{da RSS}} + \overbrace{\lambda \cdot \partial_{w_j}|w_j|}^{\text{da L1 penalty}}$$

$$\partial_{w_j}[\text{costo\_lasso}] = 2z_j w_j - 2\rho_j + \begin{cases} -\lambda & \text{se } w_j < 0 \\ [-\lambda, \lambda] & \text{se } w_j = 0 \\ \lambda & \text{se } w_j > 0 \end{cases}$$

Differential Set della funzione di costo Lasso

- Abbiamo pertanto la seguente espressione finale:

$$\partial_{w_j}[\text{costo\_lasso}] = \begin{cases} 2z_j w_j - 2\rho_j - \lambda & \text{se} \quad w_j < 0 \\ [-2\rho_j - \lambda, -2\rho_j + \lambda] & \text{se} \quad w_j = 0 \\ 2z_j w_j - 2\rho_j + \lambda & \text{se} \quad w_j > 0 \end{cases}$$

Soluzione ottima

Se uguagliamo a zero la precedente espressione, abbiamo tre casi:

- caso 1 ($w_j < 0$): $2z_j\hat{w}_j - 2\rho_j - \lambda = 0$

  da cui otteniamo:

$$\hat{w}_j = \frac{2\rho_j + \lambda}{2z_j} = \frac{\rho_j + \frac{\lambda}{2}}{z_j}$$

  Poiché $\hat{w}_j < 0$, abbiamo:

$$\hat{w}_j = \frac{\rho_j + \frac{\lambda}{2}}{z_j} < 0 \quad \Rightarrow \quad \rho_j + \frac{\lambda}{2} < 0 \quad \Rightarrow \quad \rho_j < -\frac{\lambda}{2}$$

Soluzione ottima

- caso 2 ($w_j = 0$): l'intervallo $[-2\rho_j - \lambda, -2\rho_j + \lambda]$ deve contenere 0

Abbiamo dunque:

$$-2\rho_j - \lambda \leq 0 \quad \Rightarrow \quad \rho_j \geq -\frac{\lambda}{2}$$

$$-2\rho_j + \lambda \geq 0 \quad \Rightarrow \quad \rho_j \leq \frac{\lambda}{2}$$

In definitiva:

$$-\frac{\lambda}{2} \leq \rho_j \leq \frac{\lambda}{2}$$

Soluzione ottima

- caso 3 ($w_j > 0$): $2z_j\hat{w}_j - 2\rho_j + \lambda = 0$

da cui otteniamo:

$$\hat{w}_j = \frac{2\rho_j - \lambda}{2z_j} = \frac{\rho_j - \frac{\lambda}{2}}{z_j}$$

Poiché $\hat{w}_j > 0$, abbiamo:

$$\hat{w}_j = \frac{\rho_j - \frac{\lambda}{2}}{z_j} > 0 \Rightarrow \rho_j - \frac{\lambda}{2} > 0 \Rightarrow \rho_j > \frac{\lambda}{2}$$

Soluzione ottima

- In conclusione:

$$\partial_{w_j}[\text{costo\_lasso}] = \begin{cases} 2z_j w_j - 2\rho_j - \lambda & \text{se} \quad w_j < 0 \\ [-2\rho_j - \lambda, -2\rho_j + \lambda] & \text{se} \quad w_j = 0 \\ 2z_j w_j - 2\rho_j + \lambda & \text{se} \quad w_j > 0 \end{cases}$$

$$\hat{w}_j = \begin{cases} \frac{\rho_j + \frac{\lambda}{2}}{z_j} & \text{se} \quad \rho_j < -\frac{\lambda}{2} \\ 0 & \text{se} \quad \rho_j \in [-\frac{\lambda}{2}, \frac{\lambda}{2}] \\ \frac{\rho_j - \frac{\lambda}{2}}{z_j} & \text{se} \quad \rho_j > \frac{\lambda}{2} \end{cases}$$

Algoritmo Coordinate Descent per Lasso

[versione con feature non normalizzate]

- calcola: $z_j = \sum_{i=1}^{N} \phi_j(\mathbf{x}_i)^2$
- Inizializza $\hat{\mathbf{w}} = 0$ (o in altro modo)
- **while** not converged:

  for j = 0, 1, …, D:

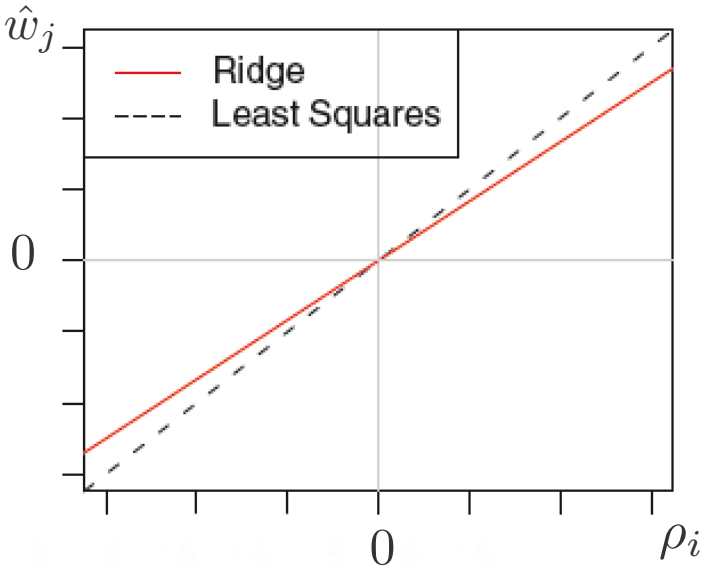
calcola: $\rho_j = \sum_{i=1}^{N} \phi_j(\mathbf{x}_i)[y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j})]$

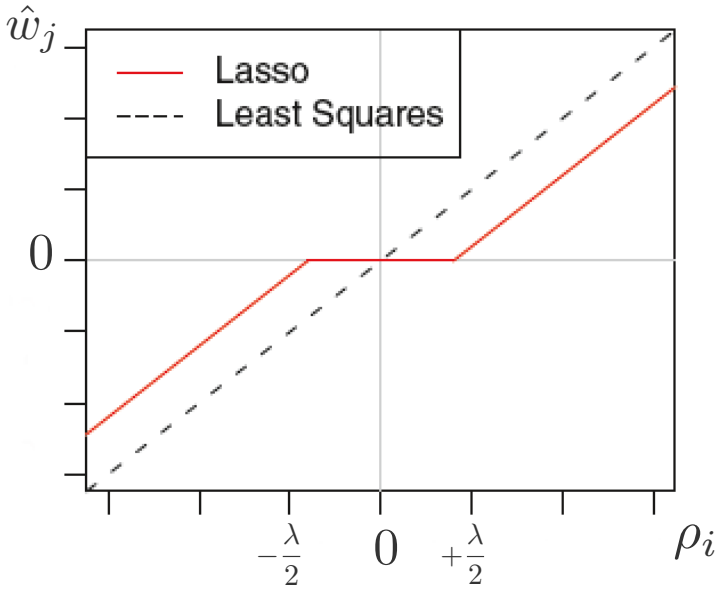
set: $\hat{w}_j = \begin{cases} \frac{\rho_j + \frac{\lambda}{2}}{z_j} & \text{se } \rho_j < -\frac{\lambda}{2} \\ 0 & \text{se } \rho_j \in [-\frac{\lambda}{2}, \frac{\lambda}{2}] \\ \frac{\rho_j - \frac{\lambda}{2}}{z_j} & \text{se } \rho_j > \frac{\lambda}{2} \end{cases}$

Coefficienti per LS, Ridge e Lasso

$$\hat{w}_j = \begin{cases} \frac{\rho_j + \frac{\lambda}{2}}{z_j} & \text{se} \quad \rho_j < -\frac{\lambda}{2} \\ 0 & \text{se} \quad \rho_j \in [-\frac{\lambda}{2}, \frac{\lambda}{2}] \\ \frac{\rho_j - \frac{\lambda}{2}}{z_j} & \text{se} \quad \rho_j > \frac{\lambda}{2} \end{cases}$$

soft thresholding





Riferimenti

- Watt, J., Borhani, R., Katsaggelos, A.K. *Machine Learning Refined*, 2nd edition, Cambridge University Press, 2020.
- James, G., Witten, D., Hastie, T., Tibishirani, R. *An Introduction to Statistical Learning*, Springer, 2013.
- Ross, S.M. *Probabilità e Statistica per l'Ingegneria e le Scienze*, Apogeo, 2015.
- *Machine Learning: Regression*, University of Washington - Coursera, 2015.
- Flach, P. *Machine Learning - The Art and Science of Algorithms that Make Sense of Data*, Cambridge University Press, 2012.
- Murphy, K.P. *Machine Learning - A Probabilistic Approach*, The MIT Press, 2012.