Progetto High Dimensional Data Analysis

Mario Pedol (830296), Marco Maugeri (872873), Nabil Elasri (826040)

Abstract

In questo studio verranno approfondite, grazie al supporto di applicazioni e simulazioni, le motivazioni che portano ad usare le principali tecniche di Feature Selection. Inoltre, si discuterà della metodologia applicativa più adeguata per gestire i casi in cui, una volta selezionate le variabili, il modello presenti problemi di overfitting.

1 Effetto delle Feature irrilevanti

Quanto le variabili inadatte a descrivere un certo fenomeno impattano sulla capacità predittiva del modello?

Generalmente, quando si costruisce un modello si auspica, per questioni di interpretabilità, che solo poche variabili siano davvero importanti nel descrivere il comportamento del fenomeno oggetto di studio. Inoltre, la capacità di previsione oltre a dipendere dal tipo di modello utilizzato, dipende anche dalla natura dei predittori e dal rapporto tra il loro numero e il numero di osservazioni a disposizione.

Al fine di dimostrare ciò, Sapp et al. (2014) simularono l'impatto sull'efficacia previsiva di diversi modelli al variare del numero di feature irrilevanti introdotte, utilizzando come metrica di valutazione l'RMSE. [1]

Nel riprodurre tale esperimento ci si è avvalsi della stessa funzione utilizzata dagli autori sopra citati, generando un data set da una funzione non lineare di 20 variabili:

$$y = x_1 + \sin(x_2) + \log(|x_3|) + x_4^2 + x_5 x_6 + I(x_7 x_8 x_9 < 0) + I(x_{10} > 0) + x_{11} I(x_{11} > 0) + \sqrt{|x_{12}|} + \cos(x_{13}) + 2x_{14} + |x_{15}| + I(x_{16} < -1) + x_{17} I(x_{17} < -1) - 2x_{18} - 2x_{19} x_{20} + \varepsilon$$

Dove ogni variabile è generata casualmente utilizzando delle normali standard indipendenti mentre l'errore è simulato con una normale a media nulla e varianza pari a 3.
Quanto appena descritto, è stato possibile riprodurlo grazie alla funzione "SCL14_1()" della libreria "caret", la quale permette anche l'introduzione di variabili di "noise" mediante un apposito parametro.

I modelli sono:

- KNN: K-Nearest Neighbors è un metodo non parametrico tipicamente utilizzato per problemi di classificazione e regressione, con il vantaggio di non richiedere alcuna assunzione sulla forma funzionale del fenomeno che si vuole studiare; d'altro canto, però, al fine di usufruire positivamente di questa flessibilità, è necessario avere a disposizione un quantitativo elevato di dati.
  Il funzionamento di questo algoritmo può essere così descritto: supponiamo di essere interessati a prevedere il valore y_1^* in corrispondenza di $\boldsymbol{x_1^*}$. All'interno del data set si cercano i *k* punti più vicini a $\boldsymbol{x_1^*}$ che denotiamo con $\boldsymbol{x_i}$ con *"i"* che va da *1* a *k*. Infine, si mediano i *k* valori y_i associati ai punti $\boldsymbol{x_i}$. Con valori bassi dell'iperparametro *k* abbiamo una stima più flessibile sui dati; invece, con grandi valori di *k* si hanno delle stime che meno si adattano al data set di train.

- Bagging Tree: modello che consiste nel generare B campioni di bootstrap dal traning set, e stimare l'albero di regressione per ciascun campione. Infine, si calcolano le previsioni facendo la media delle stime ottenute dai B alberi per la singola osservazione del test set:

$$\bar{f}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(x)$$

- Random Forest: nel random forest, a differenza del bagging tree, prima di effettuare una suddivisione in corrispondenza di un certo nodo dell'albero, si selezionano casualmente "m" predittori come possibili candidati per lo split. Quindi "m" è un parametro di tuning e il bagging tree risulta essere un caso particolare della random forest per m=p, dove p è il numero totale di regressori.

- Regressione lineare: nella regressione lineare si ipotizza che la forma funzionale del fenomeno oggetto di studio sia di tipo lineare $y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$ dove gli x_i sono i regressori e i β_i sono dei coefficienti da stimare attraverso la seguente minimizzazione:

$$min_{\beta \in R^p} ||y - X\beta||_2^2$$

- Ridge: nella regressione Ridge si aggiunge un termine di penalità ai minimi quadrati della regressione lineare e quindi si minimizza la quantità:

$$min_{\beta \in R^p} ||y - X\beta||_2^2 + \lambda ||\beta||_2^2$$

  il parametro λ (≥0) è un parametro di tuning. Per $\lambda = 0$ si ritorna al caso della regressione lineare, per $\lambda \to \infty$ si ha un effetto di penalità molto grande, il che comporta che molti coefficienti siano prossimi a zero ma mai esattamente nulli. In generale aumentando il valore del parametro λ si avrà una minore flessibilità del modello, cioè varianza minore e bias maggiore.

- Lasso: la regressione Lasso permette di escludere alcune feature dal modello e quindi di eseguire una feature selection intrinseca. Questo avviene utilizzando una penalità diversa rispetto alla Ridge, data da:

$$min_{\beta \in R^p} ||y - X\beta||_2^2 + \lambda ||\beta||_1$$

  Analogamente a quanto visto nella Ridge, il metodo Lasso costringe i coefficienti stimati ad assumere valori vicini allo zero, e ad annullarsi totalmente laddove la norma sia pari a 1.
  Per valori elevati $\lambda \to \infty$, il modello risultante è il modello nullo.

La simulazione è stata fatta al variare della numerosità dei dati, 500 e 1000 unità statistiche e del numero di variabili di "noise" da 0 a 100 a step di 25. Per ottenere risultati stabili, e senza gravare troppo sul peso computazionale, si è scelto di iterare la procedura per 5 volte mediando gli esiti intermedi su seed diversi.

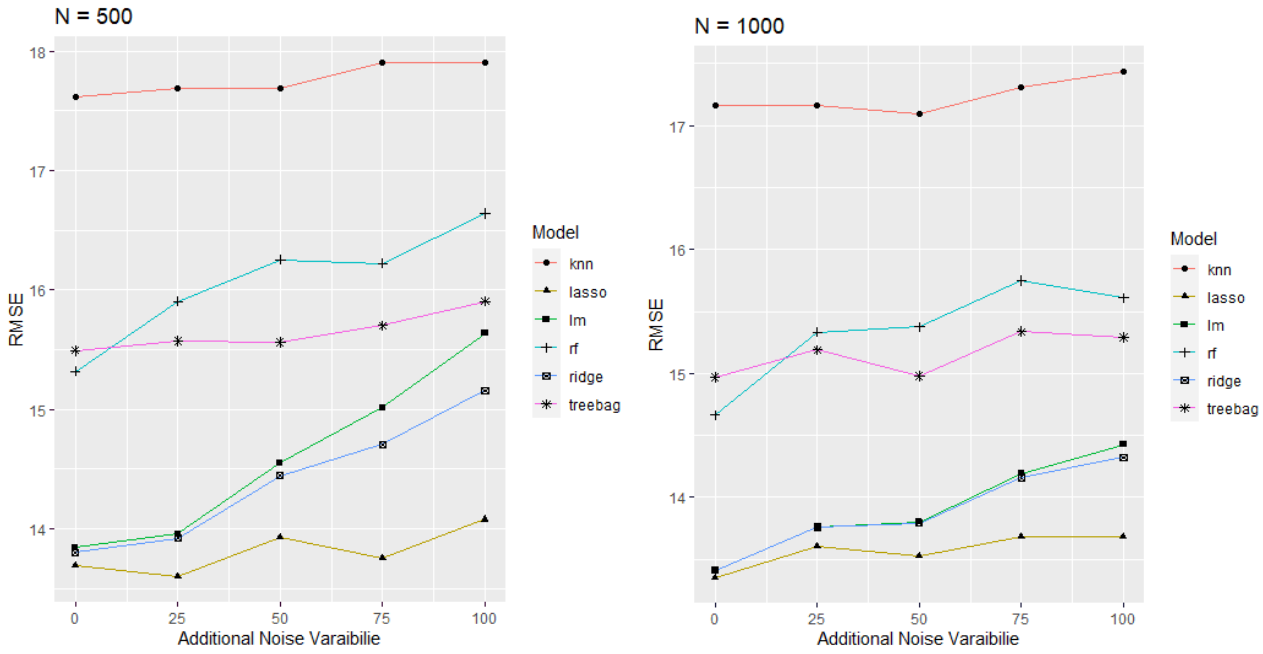

Figura 1: *andamento dell'RMSE mediato all'aumentare del noise.*

Possiamo notare dalla Figura 1 che la regressione Lasso è la migliore, più robusta della regressione Ridge, la quale non si discosta molto dalla regressione lineare soffrendo molto la presenza di variabili "noise". Il modello peggiore risulta essere il KNN, ciò potrebbe essere dovuto alla dimensione ridotta del dataset. Inoltre, osserviamo che la Random Forest è più performante del Bagging Tree solo nel caso in cui non ci sono variabili di "noise". All'aumentare della dimensione del data set si hanno risultati migliori e l'effetto delle variabili di "noise" appare essere meno importante.

2 Tecniche principali di Feature Selection

Le tecniche di feature selection supervisionate si possono dividere in tre principali gruppi:

1. Filter Feature Selection;
2. Wrapper Feature Selection;
3. Intrinsic Feature Selection.

Al fine di comprenderne meglio il funzionamento, si è deciso di approfondire un metodo per ognuna delle precedenti tecniche, utilizzando il data set Breast Cancer contenuto nella libreria mlbench di R.
Questo è composto da 11 variabili e 699 osservazioni così definite:

1. *ID:* codice identificativo;
2. *Cl.thickness:* Spessore del ciuffo;
3. *Cell.size:* Uniformità delle dimensioni delle cellule;
4. *Cell.shape:* Uniformità della forma delle cellule;

5. *Marg.adhesion:* Adesione marginale;

6. *Epith.c.size:* Dimensione della singola cellula epiteliale;

7. *Bare.nuclei:* Nuclei nudi;

8. *Bl.cromatin:* Cromatina insipida;

9. *Normal.nucleoli:* Nuclei normali;

10. *Mitoses:* Mitosi;

11. *Class:* Variabile dipendente che indica se il cancro è benigno o maligno.

Procedendo con una rapida analisi esplorativa, si nota che tutte le variabili sono di tipo numerico, con un range che varia da 0 a 10, ad eccezione di "ID", che viene eliminata prima di qualsiasi analisi avendo solo uno scopo identificativo, e "Class" variabile di interesse.
Inoltre, è opportuno notare la presenza di 16 valori mancanti che ai fini del nostro scopo verranno rimossi, in quanto questa operazione non va a causare una perdita di informazioni rilevante.
Il data set risultante, quindi, sarà composto da 683 osservazioni, 9 predittori e la variabile dipendente.

2.1 Filter Feature Selection

Questi metodi consistono nel definire una qualche statistica, detta filtro, con l'obiettivo di determinare una relazione tra uno o più predittori e la variabile dipendente. In generale, si scartano tutte le feature che non soddisfano questa regola, mantenendo le altre.

Il metodo che si è scelto di approfondire è "ChiSquared" della funzione "filterEvaluator()" contenuto nella libreria "FSinR", il quale si serve di un Test Chi-quadro per verificare l'indipendenza tra due variabili.
L'ipotesi nulla che si vuole testare è:

H0: (x e y sono indipendenti) *H1: (x e y non sono indipendenti)*

Per procedere con il test si calcola la tabella di contingenza delle due variabili e si ottengono i gradi di libertà della statistica test:

$$df=(r-1)*(c-1)$$

Dove:

r = numero di righe;
c = numero di colonne.

Si prosegue calcolando i valori attesi della tabella:

$$E = N*p$$

Con

$$p = (Total_i/N)* (Total_j/N)$$

Infine, si calcola il valore della chi quadro come:

$$\frac{(O-E)^2}{E}$$

Dove

O = valore osservato,
E = valore atteso.

E si verifica se il valore ottenuto, calcolando il p-value, cade o meno nella soglia di rifiuto o accettazione della regione critica. Se si rifiuta H0 allora la variabile rimane nel training set altrimenti viene scartata. [2]

Applicando quanto appena detto al data set Breast Cancer mediante la funzione implementata "fselection_filter()" si ottiene la Figura 2 riportante i valori del Chi-Quadro.

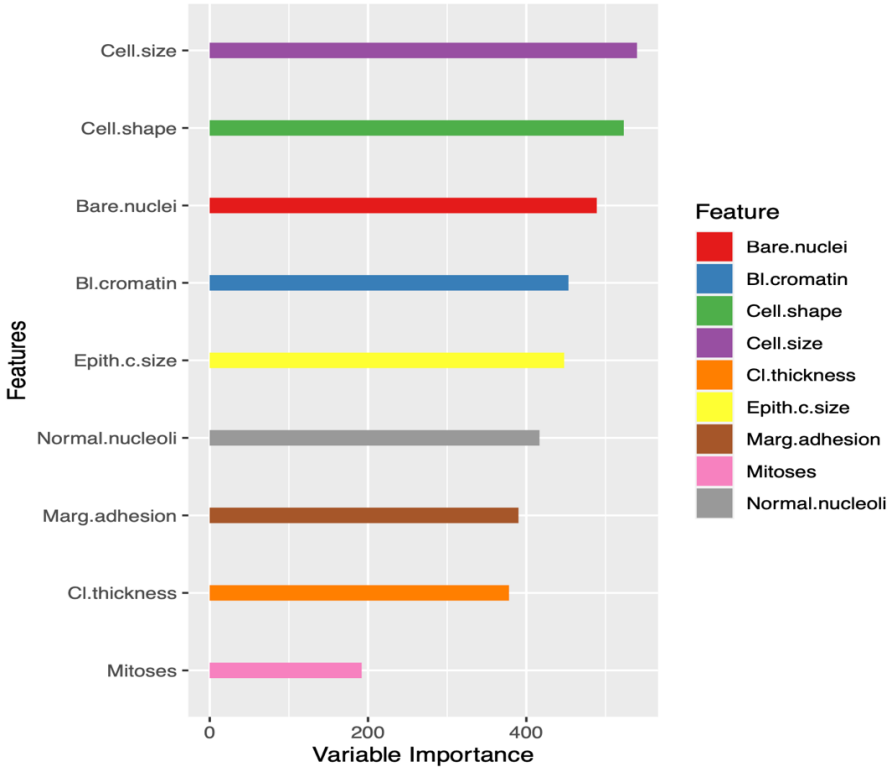

Figura 2: *Valore Chi-Quadro per ogni variabile.*

Questi possono essere intesi come misure d'importanza. Infatti, più sarà basso il valore del Chi-Quadro più sarà basso il p-value, di conseguenza la variabile sarà meno significativa.

2.2 Wrapper Feature Selection

I metodi wrapper sono una procedura di ricerca iterativa delle variabili, forniscono sottoinsiemi di predittori, valutando passo a passo un modello scelto con l'obiettivo di selezionare il sottoinsieme che porta alle performance migliori.
Tali metodi possono adottare un approccio greedy o non greedy. La ricerca greedy è quella che sceglie il percorso di ricerca in base alla direzione che sembra essere migliore in quel momento, senza più tornare indietro e riconsiderare le altre direzioni. Al contrario, la ricerca non greedy è in grado di rivalutare le precedenti combinazioni di feature e di muoversi in direzioni inizialmente sfavorevoli per raggiungere potenziali benefici nell'iterazione successiva.

Per capirne meglio il funzionamento si è voluto approfondire un metodo greedy. In particolare, si è applicata la funzione rfe(), Recursive Feature Elimination, della libreria Caret di R. La funzione richiede quattro parametri:

- **x** : una matrice di dati delle feature;
- **y:** la variabile target da prevedere;
- **size**: il numero di feature che dovrebbero essere mantenute nel processo di selezione;
- **rfeControl**: un elenco di opzioni di controllo per l'algoritmo di selezione delle feature.

Per quest'ultimo parametro si è deciso di utilizzare l'algoritmo Random Forest ("rfFuncs") perché ha un meccanismo robusto per calcolare l'importanza delle feature. In particolare, per ogni albero, viene registrata l'accuratezza della previsione sulla parte dei dati out-of-bag (inutilizzati). Quindi lo stesso viene fatto per ogni combinazione delle variabili pari a size, mediando l'accuratezza ottenuta su tutti gli alberi. L'importanza delle variabili predittive viene così determinata sulla base delle differenze di performance che si ottengono permutando le variabili stesse. [3]
Applicando quanto appena detto al data set descritto nei paragrafi precedenti, la RFE restituisce un sottoinsieme ottimale (sulla base dell'accuracy) di 8 feature; la funzione rfe() permette anche di esaminare visivamente, con un semplice barplot riportato nella Figura 3, l'importanza delle variabili predittive selezionate.

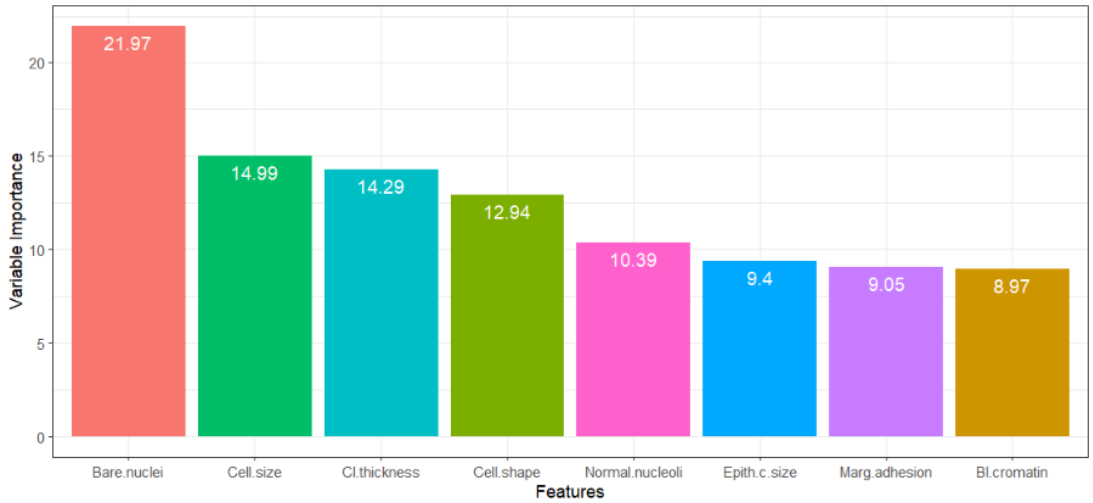

Figura 3: *Importanza Variabili RFE.*

Si nota come la variabile Bare.nuclei sia molto più importante delle altre, seguita da Cell.size e Cl.thickness. Il valore dell'asse delle Y deve essere interpretato come un semplice punteggio d'ordinamento assegnato dall'algoritmo.

2.3 Intrinsic Feature Selection

I metodi intrinsic hanno la selezione delle variabili incorporata nel processo di modellazione; pertanto, risultano essere veloci e permettono di fare una scelta informata tra il numero delle variabili e la performance predittiva. Un esempio di modello che include la selezione interna è la regressione Lasso.

Questo ha un approccio che aggiunge una penalità che cerca di tener conto della complessità del modello per ridurre il grado di overfitting o la varianza, aggiungendo più bias.
Poiché il termine di penalità cresce con il valore dei parametri dei pesi, possiamo indurre la sparsità attraverso questa norma del vettore L1:

$$L1\colon \lambda \sum_i \beta_i = \lambda ||\beta||_1$$

che può essere considerata come un metodo intrinseco di selezione delle variabili che fa parte della fase di modellazione.

Questo fenomeno si può osservare anche dalla Figura 1 dove si nota che l'RMSE del modello Lasso rimane pressoché costante anche dopo l'introduzione di variabili "noise".

3 Overfitting nella Feature Selection

Un problema che si può manifestare quando si esegue la selezione delle variabili è l'overfitting. Infatti, è possibile individuare un sottoinsieme di variabili che si adatta bene sul training set ma non sul test set.
Come viene fatto in altre situazioni simili che conducono all'overfitting, si può adottare una soluzione per "sfuggire" a tale problema, ossia utilizzare un processo di ricampionamento per selezionare le variabili.
Tuttavia questa soluzione, se non applicata adeguatamente, porta a frequenti errori. In seguito, vengono elencate due procedure:

1. Condurre il ricampionamento solo all'interno dell'iter di selezione delle variabili.

   Con lo schema che segue, si può descrivere un'applicazione di tale procedura:

   a. Il data set viene suddiviso in training e test set;
   b. Si applica una misura di importanza per determinare il rank delle variabili;
   c. Si rimuove la variabile meno importante;
   d. Si ricampiona il training set, con le variabili selezionate, dividendolo in Analysis e Assessment set;
   e. Si fitta il modello sull'Analysis set e si misura la performance sull'Assessment set;
   f. Si ripete lo schema dal punto c. finché tutte le variabili non sono rimosse;
   g. Si prende il set di variabili che risulta avere la performance più elevata.

Mettendo in pratica questa procedura al data set Breast Cancer si può notare come questo metodo induca a overfitting.
Infatti, scegliendo come misura del rank il valore del Chi-Quadro, come metodo di ricampionamento un campionamento casuale senza reinserimento e come modello per misurare la performance migliore il KNN si ottiene che il migliore subset di variabili individuato all'interno della procedura è:

"Cell.size" | "Cell.shape" | "Epith.c.size" | "Bare.nuclei" | "Class"

Addestrando il modello KNN con le variabili selezionate sull'intero training set si ottiene una buona performance sugli stessi dati ma leggermente inferiore sul test set; questo è un segnale di overfitting, sebbene non sia così evidente a causa dello scarso numero di osservazioni.

Accuracy Training	0.95
Accuracy Test	0.94

Le ragioni che ci portano a concludere che tale procedura è sbagliata sono le seguenti:

1) Poiché la valutazione dell'importanza delle variabili è esterna al ricampionamento, esso non è in grado di quantificare efficacemente l'impatto del processo di selezione.

2) Vengono utilizzati gli stessi dati per misurare le performance e guidare la direzione della selezione.

2. Selezione delle variabili come componente del processo di modellazione (metodo migliore del precedente). In altre parole, un modo appropriato per fare feature selection è quello di farlo all'interno del processo di ricampionamento.

Con lo schema che segue, si può descrivere un'applicazione di tale procedura:

a. Il data set viene suddiviso in training e test set;
b. Si ricampiona il training set, che viene diviso in Analysis e Assessment set;
c. Si applica una misura di importanza alle variabili dell'Analysis set per determinarne il rank;
d. Si rimuove la variabile meno importante;
e. Si addestra il modello sull'Analysis set e si misura la performance sull'Assessment set;
f. Si ripete lo schema dal punto d. finché tutte le variabili non sono rimosse.
g. Si determina il migliore sottoinsieme prendendo le migliori performance del modello, se esistono modelli che per diversi riacampionamenti sono uguali, allora si media la loro performance in termini di accuracy;
h. Si ripete lo schema per ogni ricampionamento (punto b).
i. Si prende il set di variabili che risulta avere la performance più elevata.

Come già fatto per il punto precedente si replica quanto descritto al data set Breast Cancer. Anche qui in modo analogo si applica come misura del rank il valore del Chi-Quadro, come metodo di ricampionamento, un campionamento casuale semplice senza reinserimento e come modello il KNN.
Prendendo poi l'intero training set con le variabili selezionate:

"Cl.thickness" | "Cell.size" | "Cell.shape" | "Epith.c.size" | "Bare.nuclei" | "Bl.cromatin" | "Normal.nucleoli" | "Class"

E addestrando un modello KNN si ottiene che la performance sul test set è maggiore a quella misurata sul training set, segno di una buona performance complessiva del modello:

Accuracy Training	0.955
Accuracy Test	0.971

Le differenze riscontrate rispetto al primo metodo possono essere così riassunte:

a) Facendo feature selection all'interno del processo di ricampionamento si ottiene una stima più accurata della perfomance complessiva.

b) Problema dell'aumento del carico computazionale: per i modelli che sono sensibili alla scelta dei parametri di tuning, può esserci bisogno di fare nuovamente un tuning del modello quando ogni sottoinsieme di predittori è valutato (i parametri ottimali cambiano ogni volta).

4 Bibliografia

[1] Kuhn, Johnson (2019). Feature Engineering and Selection. Chapman and Hall/CRC.

[2] https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223

[3] https://topepo.github.io/caret/variable-importance.html