Guadagnerai più di $50.000?

Lorenzo Caimi, Valerio Calà, Lorenzo Caruso

Università degli Studi di Milano - Bicocca

Abstract

Gran parte della letteratura microeconomica applicata al mercato del lavoro è incentrata sullo studio delle determinanti del salario: in particolare c'è un grande interesse nel capire se esiste o meno una relazione causale tra l'educazione e il salario. Il focus sulla causalità solleva diversi problemi a livello di ricerca empirica, che per essere risolti richiedono di formulare teorie e di adottare specifiche tecniche che assicurano validità interna. Noi invece, grazie ai *tools* imparati con il corso di Machine Learning, vogliamo sviluppare un modello predittivo (che non implica quindi relazioni causali tra le variabili studiate) al fine di classificare correttamente nuove osservazioni sulla base di caratteristiche osservabili. I modelli predittivi, a differenza di quelli causali, vogliono massimizzare la probabilità di previsione *out-of-sample* senza prestare troppa attenzione alla non distorsione delle stime. A titoli esemplificativo, pensiamo ad una situazione in cui il *policy maker* vuole fare una politica assistenziale che avvantaggerà solo un certo gruppo di individui, ad esempio i più poveri: in questo contesto è fondamentale riuscire a creare un modello in grado di prevedere lo scaglione di reddito dell'aspirante beneficiario e confrontarlo con quanto dichiarato. Per questo abbiamo sfruttato i dati relativi ad un censimento americano, disponibili su Kaggle[1], per costruire un modello predittivo in grado di classificare gli individui in due scaglioni di reddito.

Sommario

Introduzione

Prima di cominciare la fase di *model building* vera e propria, ispezioniamo il contenuto del dataset originario, composto da 32.561 osservazioni e 15 variabili:

- ***Income:*** variabile target binaria, rappresenta il reddito degli individui suddiviso in >50K oppure <=50K
- ***Age:*** variabile quantitativa che rappresenta l'età degli individui espressa in anni
- ***Workclass:*** variabile nominale che rappresenta il tipo di lavoro che svolge l'individuo
- ***Fnlwgt:*** variabile quantitativa continua dall'interpretazione poco chiara
- ***Education:*** variabile nominale che rappresenta il più alto livello di istruzione raggiunto da quell'individuo

- ***Education.num:*** conversione numerica della precedente variabile
- ***Marital.status:*** variabile categoriale rappresentante lo stato civile dell'individuo
- ***Occupation:*** l'occupazione dell'individuo
- ***Relationship:*** contiene i valori relativi alle relazioni familiari
- ***Race:*** variabile nominale con modalità all'etnia dell'individuo
- ***Sex:*** il sesso (uomo/donna)
- ***Capital gain:*** variabile numerica, rappresenta la plusvalenza degli investimenti effettuati dall'individuo
- ***Capital loss:*** la minusvalenza degli investimenti effettuati dall'individuo
- ***Hours per week:*** variabile numerica, rappresenta le ore lavorate a settimana
- ***Native country:*** il paese di origine dell'individuo

La variabile dipendente della nostra analisi è *income*. Negli studi di classificazione si definisce come classe positiva quella la cui modalità è più rara, nel nostro caso: reddito superiore a $50.000 annui. Sulla base della classe attuali e di quelle previste dal modello di classificazione, costruiremo le principali misure di performance che ci porteranno a scegliere il miglior modello in termini di capacità previsionali.

La relazione è suddivisa nel seguente modo:

- Nel primo paragrafo viene affrontata un'analisi esplorativa sul dataset originario, finalizzata a capire quali variabili vanno escluse a priori e quali necessitano di una preliminare trasformazione per essere utilizzate.
- Nel secondo paragrafo presentiamo i diversi cinque algoritmi di classificazione utilizzati durante l'analisi.
- Nel terzo paragrafo addestriamo e validiamo i classificatori con metodologia *hold-out*.
- Nel quarto paragrafo discutiamo la procedura di *feature selection* secondo gli approcci *filter* e *wrapper* per individuare la migliore combinazione di attributi per ogni tecnica di classificazione.
- Nel quinto paragrafo addestriamo e validiamo i classificatori con metodologia *cross validation* con e senza gli attributi pre-selezionati.
- Nel sesto e conclusivo paragrafo, confrontiamo le misure di performance di tutti i modelli ottenuti per decidere quale tra questi è il migliore.

1) Preprocessing

Colleghiamo il dataset originario ad un nodo *statistics* ed ispezioniamo gli istogrammi ed i diagrammi a barre di tutti gli attributi sia quantitativi che qualitativi: ci rendiamo subito conto che la maggior parte dei *missing values* non `e casuale, in quanto presente congiuntamente nelle variabili *workclass* ed *occupation*, e che molte variabili qualitative presentano un alto numero di modalità rendendo quindi gli algoritmi computazionalmente più onerosi senza una preliminare fase di aggregazione. Delle 15 variabili disponibili nel dataset originario, decidiamo di rimuoverne da subito 6:

- ***fnwgt*** perchè non è chiara la sua interpretazione
- ***capital.loss*** e ***capital.gain*** dato che più dell'80% delle osservazioni ha congiuntamente valore zero, contro i valori molti altidelle restanti osservazioni.
- ***native.country*** poiché il 90% delle osservazioni ha come modalità United-States su un totale molto alto di diverse modalità
- ***relationships*** poiché di scarsa utilità senza un'opportuna aggregazione
- ***education*** perché è già disponibile nel dataset la stessa informazione in forma numerica

A questo punto risolviamo il problema dei *missing values* adottando prima una strategia di *missing replacement* ed eliminando poi i records che presentavano ancora valori mancanti in una delle due voci. La strategia adottata consiste nel creare una nuova voce per la variabile *workclass: inactive.* L'etichetta viene assegnata sfruttando una terza variabile, *age*, ed individua il gruppo di individui esclusi dalla forza lavoro o perché troppo giovani (quindi con *occupation* verosimilmente *student*) o perché troppo anziani (quindi con *occupation* verosimilmente *retired*).
Vengono poi effettuate delle trasformazioni sulle seguenti variabili:

- ***workclass*** viene aggregata in *Inactive, Private, Public, Without-pay*
- ***marital.status*** viene aggregata in *Married, Not Married*
- ***occupation*** viene esclusa per semplificare il problema, dato l'elevato numero di modalità che può assumere
- ***income*** viene rinominata in income >50K: *yes* or *no*

Colleghiamo a questo punto la tabella contenente i dati preprocessati con un nodo *partitioning* per suddividere il dataset ed evitare di incorrere in un alto errore di generalizzazione addestrando il modello sulla totalità dei dati. Con il nodo di partizione è stato estratto il *validation set* (10%) e il restante 90% delle osservazioni è stato

diviso in *training set* (67%) e *test set* (33%). Questa divisione multipla è necessaria per ottenere misure di accuratezza non distorte e per non incorrere in *overfitting*. Dato che la variabile *income* presenta classi sbilanciate, per sviluppare modelli che prevedano bene soggetti con income >50K è opportuno campionare adeguatamente il dataset: scegliamo una tecnica di campionamento stratificato per mantenere costante e pari a circa 1/3, nelle varie partizioni di dati, il rapporto tra individui con classe positiva su quelli aventi classe negativa.

2) Modelli & Misure di Valutazione

2.1 Modelli

Per costruire il nostro modello predittivo, abbiamo utilizzato 5 tecniche di classificazione differenti: ***Random Forest, J48, NaiveBayes, MultilayerPerceptron*** e ***LogisticRegression***.

Gli attributi considerati nei nostri modelli sono i seguenti: *age, education.num, hour.per.week, marital.status, workclass, sex, race, income.*

Random Forest: il Random Forest[2] è un classificatore *ensemble* composto da molti alberi di decisione: combina le singole classificazioni, ottenute da alberi addestrati parallelamente su porzioni ridotte del dataset. Gli alberi di decisione che compongono il Random Forest sono un metodo di apprendimento supervisionato non parametrico, in quanto non ipotizza una sottesa forma funzionale, utilizzato sia per la classificazione di variabili qualitative che per la regressione di variabili quantitative. Esso utilizza tecniche per analizzare la relazione tra una variabile dipendente (target) ed altre variabili indipendenti (covariate di qualsiasi tipo). I vantaggi di questo metodo sono quelli dei metodi cosiddetti ”euristici”: la sua flessibilità, visto che non richiede che i dati soddisfino una particolare relazione matematica, ed il fatto che è capace di lavorare con attributi misti; inoltre riporta di solito buone performance, anche se le sue ipotesi sono in qualche modo violate dal vero processo generatore dei dati. Lo svantaggio è il rischio di far crescere un insieme di alberi troppo complesso, che manca di interpretabilità e con un alto errore di generalizzazione.

Naive Bayes: il Naive Bayes [3] è un classificatore probabilistico che si basa sull’applicazione del teorema di Bayes e l’ipotesi di indipendenza condizionata tra tutti gli attributi esplicativi. Sebbene spesso questa ipotesi sia violata, questi algoritmi si dimostrano particolarmente robusti. Ciò che viene richiesto per implementare l’algoritmo è la conoscenza delle distribuzioni condizionate (ad *income*) degli attributi esplicativi, nonché la prior della variabile target. Viene poi calcolata la distribuzione a posteriori della classe, condizionata ai valori assunti dalle esplicative, e si etichetta il record con la classe con probabilità massima.

J48: l’algoritmoJ48[4] è l’implementazione Weka dell’albero di decisione C4.5. Uno dei vantaggi di questo algoritmo è che si possono classificare sia dati numerici che nominali, ma l’attributo di output deve essere categoriale; inoltre, non sono necessarie assunzioni a priori sulla natura dei dati. Resta il fatto che, purtroppo, molteplici attributi output non sono consentiti e che se troppo complessi gli alberi di decisione possono dare performance diverse se addestrati su dati diversi.

LogisticRegression: la regressione logistica [5] è un caso particolare di modello lineare generalizzato avente come specificazione funzionale la funzione *logit*. Si tratta di un modello di regressione applicato nei casi in cui la variabile dipendente è binaria. Come in tutti i modelli di regressione, si utilizza la variabilità nei dati per stimare i parametri che entrano nella funzione e che catturano gli effetti marginali delle esplicative sulla probabilità di avere un reddito maggiore di $50K. In particolare implementiamo la versione *ridge estimator* che permette di trovare il miglior compromesso tra distorsione e varianza del classificatore, modificando la funzione obiettivo con un regolarizzatore che penalizza la scelta di funzioni complesse (con tanti attributi).

Multilayer Perceptron: il Multilayer Perceptron [6] è un classificatore che utilizza neuroni artificiali di diverse tipologie, in comunicazione tra loro in modo unidirezionale. In particolare ci sono:

- Neuroni di input associati alle variabili esplicative
- Neuroni nascosti corrispondenti alle diverse trasformazioni che collegano le variabili input all’output
- Neurone di output associato alla classe degli attributi etichetta

Ogni neurone di input viene collegato ad ogni neurone presente nello strato nascosto attraverso un link diretto in un’unica direzione: dal neurone di input al neurone dello strato nascosto (direzione del segnale). Una volta che ogni neurone di input ha apportato il segnale, i neuroni dello strato nascosto si attivano. Da questi il segnale verrà inviato ai neuroni di output.
Non esiste una regola di decisione riguardo al numero di strati nascosti e al numero di neuroni in ognuno di questi strati: più aumentano, più è complessa la struttura e computazionalmente oneroso l'algoritmo. Si crea una completa connessione tra i neuroni dei diversi strati nascosti: il segnale infatti deve passare dai neuroni di ogni

strato, mantenendo la condizione che le informazioni vengano propagate solamente in un'unica direzione e sotto la condizione di non collegare i neuroni dello stesso strato.

2.2 Misure di Valutazione

Avendo definito come classe positiva quella degli individui con un reddito maggiore di $50.000, possiamo costruire la seguente matrice di confusione (Tabella 1) che verrà calcolata per ogni algoritmo di classificazione implementato nella nostra analisi.

		Inducer Prediction	
		-1	+1
Actual Class	-1	TN	FP
	+1	FN	TP

Tabella 1. Confusion Matrix

Dove con TN (TP) indichiamo i *true negative (true positive)* ovvero la porzione di classe negativa (positiva) predetta correttamente e con FN (FP) indichiamo i *false negative (false positive)* ovvero la porzione di classe negativa (positiva) predetta erroneamente. Basandoci su queste quantità abbiamo calcolato gli indici [7] che ci consentono di valutare l'effettiva bontà dei modelli utilizzati. Tra questi abbiamo:

Accuracy:

$$A = \frac{TP + TN}{TP + FP + TN + FN}$$

Indica il rapporto tra le osservazioni positive e negative predette correttamente.

Recall:

$$R = \frac{TP}{TP + FN}$$

Indica il rapporto tra il numero di osservazioni correttamente classificate come positive e il numero di valori appartenenti alla classe positiva.

Precision:

$$P = \frac{TP}{TP + FP}$$

Indica il rapporto tra il numero di osservazioni correttamente classificate come positive e il numero di valori previsti, classificati come classe positiva.

F-measure:

$$F = \frac{2 \cdot R \cdot P}{R + P}$$

Media armonica tra Recall e Precision.

Queste sono le misure che generalmente vengono utilizzate come indici di bontà del modello.
In particolare guardiamo al modello con l'Accuracy più alta se diamo lo stesso peso ad entrambi i tipi di errore (falsi positivi, cioé individui per cui prevediamo un reddito superiore ai $50K ma che in realtà hanno un reddito inferiore, e falsi negativi, il viceversa). In molti contesti però, come quello di una politica assistenziale, può avere più senso voler minimizzare il numero di falsi negativi: in questi casi, sarà la Recall la nostra misura di riferimento.
Noi calcoliamo e confrontiamo tutte le misure, mostrando come in questo caso il miglior classificatore è robusto rispetto alle diverse metriche proposte.

3) Hold out

Il primo approccio adottato per addestrare e validare i modelli di classificazione è l'*hold out*: partizioniamo il dataset in due sottoinsiemi disgiunti, con un campionamento stratificato per mantenere invariate le proporzioni tra classi positive e negative. Il primo sottoinsieme, detto *training set*, ha il 67% delle osservazioni e verrà utilizzato per addestrare l'algoritmo di classificazione. Il secondo sottoinsieme, invece, si chiama *test set* e verrà utilizzato per validare il modello attraverso un *inducer* (che applica il modello ai nuovi records) ed uno *scorer* (per calcolare le misure di performance). Il metodo *hold out* è stato applicato a tutti e cinque i modelli di classificazione precedentemente descritti, in modo da poterne calcolare gli indici di performance e confrontarli tra loro (tabella 2).

	RF	NB	J48	Logistic	MLP
Accuracy	0.798	0.817	0.826	0.819	0.825
Recall	0.527	0.656	0.503	0.493	0.571
Precision	0.594	0.616	0.695	0.675	0.663
F-Measure	0.869	0.878	0.890	0.886	0.887

Tabella 2. Confronto indici di performance in Hold Out

Dalla tabella e relativi box plots consultabili nel *workflow* possiamo notare che, secondo la misura dell'accuratezza, il Random Forest performa peggio degli altri classificatori, mentre secondo la *recall* il miglior classificatore è il Naive Bayes con una differenza di performance significativa rispetto alle altre tecniche.
Ad ogni modo l'*hold out* rappresenta una tecnica di partizione molto elementare e soggetta a pericoli di distorsione e *overfitting,* dovuti dal fatto che non possiamo controllare quali records finiranno nel *test set*.

4) Feature Selection & Validation

Il fine della Feature Selection [8] è quello di ridurre il numero degli attributi per migliorare l'interpretabilità del classificatore. Nel modello abbiamo deciso di utilizzare due approcci differenti:

1. ***Filter*:** gli attributi sono selezionati prima di apprendere il classificatore.
   Abbiamo utilizzato due tipi di Filter:
   - *Univariate*
   - *Multivariate*

2. ***Wrapper***: il classificatore è usato per cercare il sottoinsieme ottimale degli attributi disponibili.
   Abbiamo utilizzato due tipi di Wrapper:
   - *Backward Elimination*
   - *Forward Inclusion*

Per selezionare le variabili rilevanti abbiamo utilizzato il nodo Weka *AttributeSelectedClassifier* con i seguenti parametri:

- ***Filter Univariate:***
  - *Evaluator:* InfoGainAttributEval
  - *Search:* Ranker
- ***Filter Multivariate:***
  - *Evaluator:* CfSubsetEval
  - *Search:* BestFirst
- ***Wrapper Forward Inclusion:***
  - *Evaluator:* WrapperSubsetEval
  - *Search*: GreedyStepwise Forward
- ***Wrapper Backward Elimination:***
  - *Evaluator*: WrapperSubsetEval
  - *Search:* GreedyStepwise Forward

Successivamente abbiamo effettuato un *Validation Test* per ogni approccio utilizzato di Feature Selection per ogni tecnica di classificazione, selezionando come attributi quelli scelti dal nodo AttributeSelectedClassifier ed effettuando poi un confronto sull'accuratezza prima tra i due approcci dello stesso tipo: Filter Univariate vs Filter Multivariate (Tabella 3), Wrapper Forward Inclusion vs Wrapper Backward elimination (Tabella 4).

Row ID	Univariate	Multivariate	Winner	WinnerAccuracy
RF	0.806	0.923	Multivariate	0.923
NB	0.82	0.83	Multivariate	0.83
J48	0.83	0.833	Multivariate	0.833
Logistic	0.829	0.825	Univariate	0.829
MLP	0.827	0.816	Univariate	0.827

Tabella 3. Confronto tra Filters

Row ID	Forward	Backward	Winner	WinnerAccuracy
RF	0.821	0.821	Same Accuracy	0.821
NB	0.83	0.83	Same Accuracy	0.83
J48	0.83	0.83	Same Accuracy	0.83
Logistic	0.825	0.828	Backward	0.828
MLP	0.827	0.827	Forward	0.827

Tabella 4. Confronto tra Wrappers

Abbiamo poi confrontato i vincitori congiunti di ogni approccio: Vincitori Filter vs Vincitori Wrapper (Tabella 5). Possiamo notare dalla figura che non esiste una regola di selezione degli attributi univoca, ma che la selezione ottimale dipende dal tipo di classificatore adottato.

Row ID	Filter	Wrapper	Winner	Winner Type	Winner Accuracy
RF	0.923	0.821	Filter	Multivariate	0.923
NB	0.83	0.83	Wrapper	Same Accuracy	0.83
J48	0.833	0.83	Filter	Multivariate	0.833
Logistic	0.829	0.828	Filter	Univariate	0.829
MLP	0.827	0.827	Wrapper	Forward	0.827

Tabella 5. Confronto tra i vincitori

5) Cross Validation

Per ovviare ai problemi dell'*hold out* adottiamo una tecnica di partizione leggermente più sofisticata: con la *cross validation* dividiamo in K *folds* il nostro dataset e utilizziamo (K-1) *folds* come *training set* e il K-esimo escluso come *test set*. In particolare, dopo varie prove, abbiamo scelto K=5 e fatte 5 iterazioni in cui, di volta in volta, un quinto dei dati veniva utilizzato come insieme di validazione. In questo modo ci aspettiamo delle performance leggermente peggiori, ma costanti al variare dei dati su cui viene addestrato l'algoritmo.
Abbiamo addestrato i cinque classificatori utilizzando prima tutte e sette le variabili esplicative, poi utilizzando solo quelle suggerite dalla procedura di *feature selection*.

5.1 Without Feature Selection

Per quanto riguarda i modelli senza preselezione degli attributi, riportiamo qui in Figura 1 e Figura 2 i box plots relativi alle misure di *accuracy* e *recall* e rimandiamo al *workflow* per un'analisi estesa alle altre metriche.

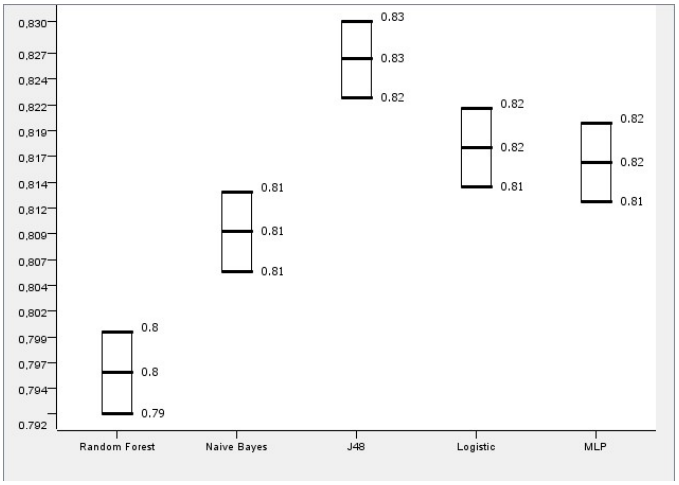

Figura 1. Accuracy nella Cross Validation senza *feature selection.*

Da questo box plot possiamo notare una differenza statisticamente significativa tra la performance del Random Forest (peggiore) e tutti gli altri. In particolare il J48 sembrerebbe il modello più performante secondo questo indice di performance.

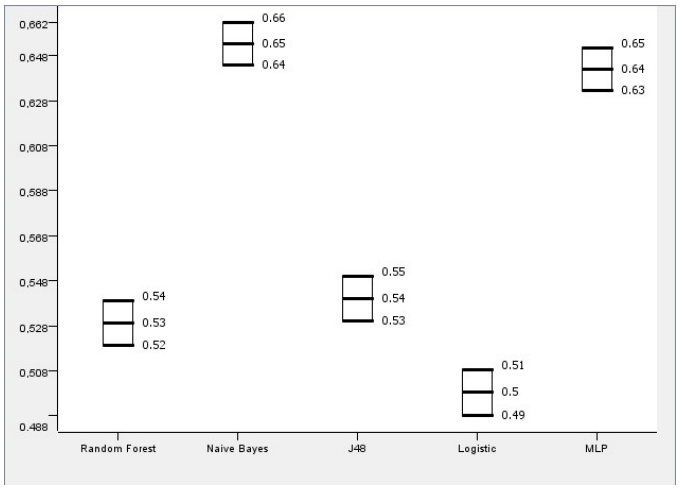

Figura 2. *Recall* nella Cross Validation senza *feature selection.*

Da questo box plot vediamo una differenza statisticamente significativa tra le performance della coppia di modelli Naive Bayes e MLP rispetto alle performance degli altri tre modelli.

5.2 With Feature Selection

In questo caso, prima di addestrare il modello di classificazione sul *training set*, filtriamo le variabili che non sono state giudicate rilevanti secondo la procedura di *feature selection* già discussa.
Rimandiamo al *workflow* per i dettagli riguardo alle variabili selezionate, in quanto ciascun classificatore utilizza attributi differenti, selezionati con approcci (filter o wrapper) e criteri (uni/multivariate e for/backward).
In Figura 3 e Figura 4 riportiamo i box plots relativi alle due misure di performance discusse (*accuracy* e *recall*) e dall'ispezione di questi possiamo concludere che, ancora una volta, il Random Forest performa sensibilmente peggio degli altri classificatori in termini di accuratezza una volta controllati i rischi di *overfitting*.

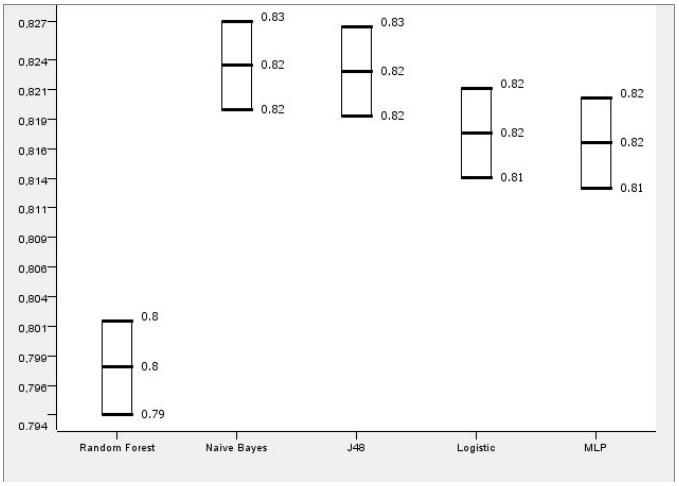

Figura 3. Box Plot dell'accuracy con feature selection

Secondo la Figura 4, invece, vediamo che basandoci sulla metrica *recall* i migliori modelli sono quelli probabilistici e di separazione: rispettivamente Naive Bayes e Multilayer Perceptron.

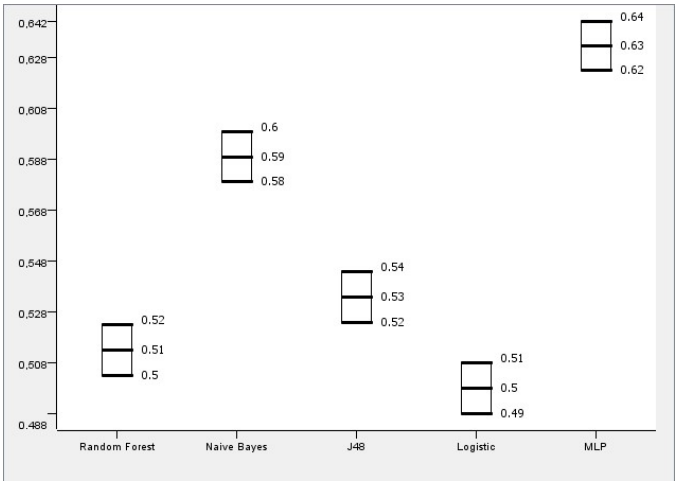

Figura 4. Box plot della recall con feature selection

6) Conclusioni

Per finire facciamo un confronto, secondo tutte le principali metriche, di tutti gli algoritmi di classificazione sviluppati con cross validation con e senza feature selection.
Qui riportiamo solo il confronto relativo alle misure di *accuracy* e *recall* con le relativi conclusioni:

Row ID	Without...	With...	Winner	Winner ...
RF	0.796	0.798	With feature selection	0.798
NB	0.81	0.824	With feature selection	0.824
J48	0.827	0.823	Without feature sel...	0.827
Logistic	0.818	0.818	No difference	0.818
MLP	0.817	0.817	With feature selection	0.817

Tabella 6. Comparazione finale su Accuracy

Row ID	Without...	With...	Winner	Winner ...
RF	0.529	0.513	Without feature selection	0.529
NB	0.653	0.589	Without feature selection	0.653
J48	0.54	0.534	Without feature selection	0.54
Logistic	0.498	0.498	No difference	0.498
MLP	0.642	0.632	Without feature selection	0.642

Tabella 7. Comparazione finale su Recall

Nella Tabella 6 possiamo vedere che, in termini di *accuracy*, il miglior modello di classificazione è il J48 senza feature selection, con un valore pari a 0.827. Tuttavia anche il NB con feature selection presenta un valore simile; la differenza tra le due performance non è statisticamente significativa.
Secondo la misura della *recall* invece, vediamo nella Tabella 7 che il NB senza feature selection presenta il valore più alto e con una differenza, rispetto a quello ottenuto con gli altri metodi, significativa.
Per concludere, se vogliamo classificare minimizzando entrambi i tipi di errore (falsi positivi e falsi negativi) implementeremo o un modello J48 senza feature selection, oppure un Naive Bayes rimuovendo gli attributi *sex* e *race,* aspettandoci circa l'83% delle osservazioni classificate correttamente in una delle due classi; se invece l'obiettivo è classificare minimizzando il numero di falsi negativi, allora utilizzeremo come modello predittivo un Naive Bayes senza feature selection, utilizzando tutte e sette le variabili esplicative disponibili, e ci aspetteremmo che circa il 65% delle assegnazioni alla classe con reddito maggiore di $50.000 sono corrette.

Bibliografia

1. https://www.kaggle.com/uciml/adult-census-income
2. https://en.wikipedia.org/wiki/Random_forest
3. https://en.wikipedia.org/wiki/Naive_Bayes_classifier
4. Ross Quinlan (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA.
5. https://en.wikipedia.org/wiki/Logistic_regression
6. https://en.wikipedia.org/wiki/Multilayer_perceptron
7. Pang-Ning Tan, Michael Steinbach and Vipin Kumar (2006). Introduction to Data Mining, Pearson International.
8. https://elearning.unimib.it/mod/book/view.php?id=236870