UNIVERSITÀ DI PISA

Dipartimento di Informatica

Corso di Laurea Triennale in Informatica

TESI DI LAUREA

Rank aggregation per la classificazione nell'ambito biomedico: analisi comparativa

Relatrice

Prof.ssa **Alina Sîrbu**

Candidata

Irene Testa

ANNO ACCADEMICO 2021/2022

Riassunto

In questo lavoro viene presentato un nuovo classificatore basato sul concetto di rank aggregation (RAC – Rank Aggregation Classifier), particolarmente adatto per la classificazione di campioni biologici in base ai loro profili di espressione genica. Le prestazioni di RAC sono state confrontate con quelle di altri 5 classificatori allo stato dell'arte, impiegando come benchmark 12 diversi dataset di espressione genica e 3 dataset di immagini. Dai risultati ottenuti è emerso che le prestazioni di RAC sono equiparabili a quelle degli altri classificatori. RAC, inoltre, è facilmente interpretabile e presenta un'intrinseca resistenza al rumore nei dati, caratteristiche che potrebbero renderlo un utile supporto per scopi diagnostici o prognostici in ambito medico-sanitario.

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

Motivazione Negli ultimi anni la comunità scientifica ha mostrato una crescente attenzione nei confronti delle tecniche di apprendimento automatico in ambito medico [1, 2]. Si ritiene, infatti, che il machine learning nel campo sanitario possa diventare a breve un valido supporto all'attività decisionale dei medici per la diagnosi e la prognosi di numerose patologie [3, 4]. In particolare, lo sviluppo di metodi di apprendimento automatico per l'analisi di profili di espressione genica e di dati omici in generale è un problema di grande interesse per le sue potenziali applicazioni investigative, diagnostiche e prognostiche. Tali metodi potrebbero, infatti, favorire la comprensione dei meccanismi molecolari responsabili dell'instaurarsi di stati patologici e consentire l'identificazione di nuovi biomarcatori, nella prospettiva dello sviluppo di una medicina sempre più precisa ed efficace [5].

Obiettivi e contributi In questo lavoro viene presentato un nuovo classificatore basato sul concetto di rank aggregation (RAC – Rank Aggregation Classifier), particolarmente adatto per la classificazione di campioni biologici sulla base dei profili di espressione genica o di altri dati di tipo *omics* (genomica, proteomica e metabolomica). In tale contesto, l'aspetto innovativo di RAC consiste in un approccio rank-based per la costruzione di class signature che includono un vasto numero di geni e che possono essere utilizzate a scopi diagnostici o prognostici. I principi su cui si basa RAC lo rendono resistente al rumore nei dati e facilmente interpretabile.

Partendo da un'implementazione iniziale del classificatore, sono state aggiunte ulteriori caratteristiche, in particolare sono stati implementati: (1) diversi metodi di assegnazione del rango alle feature aventi lo stesso valore, (2) le varianti del metodo di aggregazione di Borda, (3) la possibilità di utilizzare come misura di distanza il

coefficiente di correlazione di Kendall, (4) due modalità di pesatura nel calcolo della distanza di Spearman, (5) l'utilizzo dei centroidi delle classi ai fini della classificazione e (6) il metodo di calcolo delle probabilità di appartenenza delle istanze da classificare alle classi.

Le performance di RAC sono state confrontate con quelle di altri 5 classificatori allo stato dell'arte sia per problemi di classificazione binaria che per problemi di classificazione multiclasse, impiegando come benchmark 12 diversi dataset di espressione genica ricavati da esperimenti di microarray e 3 dataset di immagini.

Struttura della tesi Nel capitolo 1 si introducono le teorie e le tecniche di machine learning alla base del lavoro sperimentale svolto in questa tesi. Il capitolo 2 tratta delle applicazioni del machine learning nell'ambito medico-sanitario e descrive in particolare alcuni metodi computazionali per l'analisi dei profili di espressione genica. Nel capitolo 3 viene presentato il concetto di rank aggregation, principio su cui si fonda il metodo di classificazione proposto in questo lavoro. Il capitolo 4 fornisce una descrizione dettagliata del funzionamento di RAC e della sua implementazione. In tale capitolo vengono, inoltre, illustrati gli aspetti che rendono RAC particolarmente appropriato per la classificazione di profili di espressione genica e di dati omici in generale. Nel capitolo 5 vengono descritti i risultati sperimentali dell'analisi comparativa effettuata per valutare l'efficacia di RAC. Nel capitolo 6 si traggono le conclusioni di questo lavoro e si suggeriscono alcuni possibili sviluppi futuri.

Ulteriori dettagli sui temi trattati sono riportati nelle appendici. Nell'appendice A viene fornito un esempio di applicazione del metodo utilizzato da RAC per stimare le probabilità di appartenenza delle istanze da classificare alle classi e vengono descritte alcune osservazioni sperimentali in relazione alla calibrazione delle probabilità così stimate. L'appendice B raccoglie alcune immagini ottenute applicando RAC ai dataset MNIST, Olivetti faces e CIFAR-10.

Capitolo 1

Machine learning

In questo capitolo si introducono i principi su cui si basa il machine learning e si presentano, in particolare, alcuni metodi di apprendimento supervisionato. Vengono, inoltre, illustrate le misure di prestazione tipicamente utilizzate per valutare le performance dei classificatori. La trattazione procede con la descrizione delle teorie e delle tecniche direttamente correlate con il lavoro sperimentale svolto.

1.1 Introduzione al machine learning

Il *machine learning* [6–8] è una branca dell'informatica che si occupa della progettazione e dello sviluppo di metodi e algoritmi in grado di apprendere autonomamente senza essere programmati in modo esplicito, capaci cioè di acquisire conoscenze e informazioni dai dati osservati e di costruire modelli predittivi che migliorino le proprie prestazioni in modo adattativo all'aumentare della quantità dei dati disponibili. Il machine learning viene impiegato per risolvere problemi per i quali non risulta possibile progettare algoritmi ad hoc in quanto, ad esempio, i dati osservati sono rumorosi e/o incompleti oppure le conoscenze necessarie per la formalizzazione del problema sono insufficienti.

I dati coinvolti nel processo di apprendimento sono detti anche *istanze* o *pattern* e sono descritti da un insieme di *feature*, cioè da attributi che ne specificano le caratteristiche e che spesso assumono valori numerici (in presenza di n feature i dati sono talvolta rappresentati da *punti* o *vettori* di uno spazio n-dimensionale).

A seconda di ciò che desideriamo venga appreso e della tipologia di dati di cui si dispone è possibile distinguere metodi di apprendimento *supervisionato* e metodi di apprendimento *non supervisionato*. Nel primo caso ai dati osservati sono associate delle *etichette*, valori categorici o numerici, che rappresentano ciò che deve essere automaticamente appreso. Sono cioè disponibili degli *esempi* da cui poter appendere al fine di costruire un modello che consenta di predire le etichette da attribuire a nuove osservazioni. I metodi di apprendimento non supervisionato, invece, si propongono di estrarre informazioni di vario genere a partire da una serie di osservazioni non etichettate. L'obiettivo di questi metodi può essere, ad esempio, quello di raggruppare i dati osservati sulla base delle loro somiglianze (*clustering*) oppure quello di proiettare i dati dallo spazio n-dimensionale delle loro feature in uno spazio bidimensionale o tridimensionale in modo da poterli visualizzare.

1.2 Validazione di un modello

L'allenamento di un algoritmo di apprendimento prevede l'utilizzo di un *training set*, cioè di un insieme di esempi necessari per addestrare il modello predittivo. Per i metodi di apprendimento supervisionato, al termine della fase di allenamento, è possibile testare il modello costruito confrontando le etichette da esso predette con quelle attese e valutando l'accuratezza delle predizioni con un'opportuna misura di prestazione. Per stimare la capacità di generalizzazione del modello è opportuno testarlo su dati che non siano stati impiegati nella fase di allenamento; è necessario dunque riservare a questo scopo un insieme separato di dati, chiamato appunto *test set*. Quando, però, i dati di esempio a disposizione sono poco numerosi, riservarne una parte per il test del modello può determinare un'eccessiva riduzione della dimensione del training set. In questo scenario è possibile incorrere nel cosiddetto *overfitting*, cioè in un eccessivo adattamento del modello alle caratteristiche specifiche dei dati usati per allenarlo piuttosto che alle caratteristiche generali del problema.

Per stimare le capacità di generalizzazione di un modello e ridurre il rischio di overfitting è possibile utilizzare una procedura che prende il nome di *k-fold cross validation*. Tale procedura prevede la suddivisione degli n esempi a disposizione in k

insiemi disgiunti chiamati *fold*, ciascuno approssimativamente della stessa dimensione, così da allenare e testare il modello k volte, impiegando per l'allenamento $k-1$ fold e riservando ogni volta un fold distinto per il test. Se il numero di fold in cui viene diviso il dataset è pari alla dimensione stessa del dataset ($k = n$), questa procedura prende il nome di *leave one out cross validation*. La k-fold cross validation consente di testare il modello su tutti i dati a disposizione e di ricavare perciò una stima più accurata delle sue abilità di generalizzazione. Inoltre, con questa tecnica la stima ottenuta è indipendente dalla ripartizione dei dati, non dipende cioè da quali esempi sono stati inseriti negli insiemi di training e di test. La k-fold cross validation può essere anche utilizzata allo scopo di sfruttare al meglio i dati disponibili per individuare gli iperparametri[1] ottimi di un modello e per stimarne le capacità predittive. Una corretta procedura di validazione richiede, in teoria, di suddividere i dati in tre insiemi disgiunti: il *training set*, il *validation set* e il *test set*. Il training set serve per allenare i modelli con diverse configurazioni di iperparametri; il validation set viene usato per testare i modelli allenati e selezionare gli iperparametri con i quali sono state prodotte predizioni più accurate; il test set serve, infine, per testare il modello selezionato[2] e valutarne le capacità di generalizzazione. La ripartizione nei tre insiemi sopra descritti si rende necessaria in quanto utilizzare un solo test set sia per selezionare gli iperparametri del modello che per stimarne le capacità predittive potrebbe condurre a stime eccessivamente ottimistiche. Se i dati sono poco numerosi, è possibile suddividerli in due soli insiemi, il *development set* e il *test set*, impiegando il primo per effettuare una k-fold cross validation in modo da allenare e valutare allo stesso tempo più modelli per selezionare il migliore e utilizzando il secondo per testare il modello selezionato.

Tra le strategie di validazione di un modello merita di essere menzionata anche la *double k-fold cross validation*. Questa tecnica prevede l'impiego di due k-fold cross validation annidate: la più "interna" viene effettuata al fine di selezionare gli iperparametri migliori e la più "esterna" consente di ottenere una stima delle capacità

[1] Gli iperparametri di un modello sono fattori che controllano il processo di apprendimento e che devono essere stabiliti a monte della fase di allenamento.

[2] É eventualmente possibile allenare nuovamente il modello selezionato sia sui dati del training set che su quelli del validation set.

di generalizzazione del modello. In particolare, si suddividono gli n dati a disposizione in k fold disgiunti e per k volte si utilizza un fold diverso come test set e i restanti $k-1$ fold come development set su cui effettuare quindi la k-fold cross validation più interna. Lo scopo di questa procedura non è quello di costruire un modello per la risoluzione di un problema – può accadere, infatti, che vengano selezionate k iperparametrizzazioni diverse nelle cross validation più interne – quanto piuttosto quello di ottenere una stima accurata delle abilità di generalizzazione del modello impiegato, dal momento che la valutazione viene ripetuta più volte su sottoinsiemi distinti ottimizzando l'utilizzo dei dati disponibili.

1.3 Feature selection

Con l'espressione *feature selection* [9] si intende il processo con cui vengono selezionate le feature più utili per la costruzione di un modello di apprendimento. In presenza di feature ridondanti, irrilevanti o rumorose selezionare le feature più significative può risultare vantaggioso per molteplici ragioni: permette la costruzione di modelli più accurati, riduce il tempo di allenamento dei modelli, consente di estrarre informazioni dai dati e rende i modelli più facilmente interpretabili.

Nell'ambito dell'apprendimento supervisionato si distinguono due metodiche per effettuare feature selection: metodi wrapper e metodi filter. Con i *metodi wrapper* la ricerca delle feature più rilevanti viene guidata dalle performance del modello. A questo scopo possono essere utilizzati *algoritmi evolutivi* [10], cioè tecniche di ricerca euristica ispirate ai principi darwiniani della selezione naturale e dell'evoluzione biologica. Con tali metodi, infatti, si generano casualmente sottoinsiemi di feature (*popolazioni*) i quali vengono combinati tra loro (*crossover*) o nei quali una feature può essere sostituita con un'altra (*mutazione*) al fine di individuare il sottoinsieme di feature che permette al modello di raggiungere prestazioni migliori. Con i *metodi filter*, invece, la rilevanza delle feature viene inizialmente stabilita indipendentemente dalle performance del modello di apprendimento, impiegando ad esempio metodi statistici per valutare la correlazione tra le feature e le etichette. In particolare, si stilano liste di feature ordinate in base alla loro rilevanza e per la costruzione del

modello si selezionano le feature in testa alla lista. Al fine di individuare quante feature estrarre dalla lista è possibile allenare e valutare il modello con un numero crescente di feature (*sequential forward generation*) o con un numero decrescente di feature (*sequential backward generation*), arrestandosi una volta individuate le performance ottimali.

Occorre menzionare, inoltre, che per alcuni algoritmi di apprendimento, come ad esempio per gli alberi decisionali, la feature selection è insita nella costruzione del modello, pertanto in questi casi si parla di *embedded* feature selection.

1.4 Classificazione

Nell'ambito dell'apprendimento supervisionato si distinguono problemi di *classificazione* e problemi di *regressione*. Nel primo caso le etichette degli esempi da cui apprendere assumono *valori categorici*, appartengono cioè a un numero finito e predeterminato di *categorie* o *classi*. Gli algoritmi progettati per la risoluzione di questo tipo di problemi prendono il nome di *classificatori*. Nei problemi di regressione, invece, le etichette degli esempi assumono *valori numerici* e gli algoritmi che risolvono problemi di questo tipo prendono il nome di *regressori*.

Di seguito vengono brevemente descritti gli algoritmi di classificazione che sono stati utilizzati in questo lavoro. Alcuni di questi possono essere impiegati anche per la risoluzione di problemi di regressione.

Nearest Centroid *Nearest Centroid* (NC) [8] è uno tra i più semplici metodi di classificazione. Questo classificatore assegna a ogni nuova istanza l'etichetta della classe avente il centroide più vicino all'istanza stessa. Tipicamente, il centroide di una classe si ottiene calcolando la media aritmetica delle feature degli esempi della classe e la distanza tra l'istanza e i centroidi si determina calcolando la distanza euclidea. Esistono, però, diverse varianti di NC che differiscono per la modalità adottata per calcolare i centroidi e le distanze. NC presenta bassi costi computazionali e una sua variante è stata applicata con successo per la classificazione di campioni biologici in base ai profili di espressione genica [11]. Anche il classificatore proposto in questo

lavoro può essere considerato una variante di NC: con RAC i centroidi vengono calcolati mediante metodi di rank aggregation e le distanze vengono determinate utilizzando misure di distanza tra rank.

K-Nearest Neighbors *K-Nearest Neighbor* (KNN) [6] è uno tra i più conosciuti algoritmi per la classificazione. Con tale metodo si assegna a un'istanza non nota l'etichetta della classe a cui appartengono il maggior numero di punti tra i k punti di esempio più vicini a essa. Il valore di k è un iperparametro del classificatore e la metrica di distanza tipicamente utilizzata è la distanza euclidea.

Una variante di questo classificatore prevede l'assegnazione di un peso a ciascuno dei k vicini in modo che i punti più prossimi all'istanza da classificare abbiano un peso maggiore nella classificazione (tale peso può essere determinato calcolando, ad esempio, l'inverso della distanza tra gli esempi e l'istanza da classificare).

KNN posticipa l'intera computazione alla fase di classificazione che risulta, dunque, computazionalmente onerosa. Tale classificatore è largamente impiegato in quanto si è dimostrato efficace su dataset di varia natura.

Support Vector Machine *Support Vector Machine* (SVM) [12] è un metodo di apprendimento che si fonda sui principi della Statistical Learning Theory introdotta da Vapnik et al. La strategia adottata da SVM per trattare problemi di classificazione binaria consiste nella costruzione di un iperpiano che separi gli esempi appartenenti alle due classi in modo da massimizzare la distanza tra i punti più vicini al piano (i *vettori di supporto*) e il piano stesso. Mappando in modo implicito i punti in spazi di dimensioni maggiori (tramite *funzioni kernel*), SVM è in grado di individuare un iperpiano che meglio separi le istanze. Questo classificatore consente, inoltre, di controllare il numero di punti di training che vengono classificati erroneamente dall'iperpiano costruito per mezzo di uno specifico iperparametro.

Nonostante SVM supporti esclusivamente problemi di classificazione binaria, è possibile trattare problemi multiclasse allenando un classificatore SVM per ogni classe e addestrandolo a separare una classe da tutte le altre (strategia *one-vs-rest*) oppure allenando un classificatore per ogni coppia di classi e addestrandolo a separare una classe della coppia dall'altra (strategia *one-vs-one*). Nuove istanze vengono

classificate con l'etichetta della classe predetta dalla maggioranza dei classificatori così allenati.

Grazie alla sua flessibilità SVM trova applicazione in numerosi ambiti ed è uno tra i metodi di apprendimento oggi più utilizzati.

Gaussian Naive Bayes *Gaussian Naive Bayes* (GNB) [6] è un classificatore probabilistico che utilizza il *teorema di Bayes*[3] per individuare la classe più probabile a partire dai dati di esempio e da alcune conoscenze a priori sulle probabilità di appartenenza delle istanze alle classi. Si basa, inoltre, sull'assunzione semplicistica (da cui il termine *naive*) che, date le classi di appartenenza delle istanze, le feature siano indipendenti le une dalle altre e che obbediscano alla *distribuzione gaussiana*.

Nonostante l'assunzione sull'indipendenza delle feature sia irrealistica, GNB si è dimostrato competitivo rispetto ad altre tecniche di classificazione più sofisticate. Il suo basso costo computazionale e la sua efficacia fanno sì che nella pratica trovi numerose applicazioni.

Random Forest Una *Random Forest* (RF) [13] è metodo di apprendimento che prevede l'impiego di un insieme di *alberi decisionali* per effettuare predizioni. Le istanze vengono classificate sulla base dell'etichetta predetta dalla maggioranza degli alberi.

In un albero decisionale ciascun nodo interno rappresenta un test su una feature, gli archi uscenti dai nodi corrispondono ai valori possibili che la feature può assumere (in caso di feature a valori continui si utilizzano intervalli) e le foglie corrispondono a etichette. Per classificare un'istanza occorre dunque sottoporla ai test dei nodi dell'albero e, in base ai valori assunti dalle feature, seguire il cammino che dalla radice conduce alla foglia che corrisponde all'etichetta da assegnare all'istanza. Un albero di decisione viene costruito associando a ogni nodo, a partire dalla radice, la feature che meglio separa gli esempi in base alla classe di appartenenza.

[3] Secondo il teorema di Bayes se A e B sono due eventi con probabilità non nulle la probabilità condizionata di A noto B è data da $P(A \mid B) = \dfrac{P(B \mid A) \cdot P(A)}{P(B)}$.

In una RF gli esempi utilizzati per costruire ciascun albero vengono selezionati randomicamente con rimpiazzamenti dall'insieme di allenamento e le feature da assegnare ai nodi vengono scelte da sottoinsiemi costruiti estraendo casualmente le feature disponibili.

Le RF richiedono tempi di allenamento e di predizione piuttosto alti ma grazie alle loro capacità di generalizzazione sono ampiamente utilizzate.

1.4.1 Misure di prestazione

In un problema di classificazione in cui le etichette possono assumere due soli valori, ovvero in un problema di *classificazione binaria*, senza alcuna perdita di generalità le istanze possono essere *positive* o *negative*. Confrontando le predizioni effettuate da un classificatore con le reali classi di appartenenza possiamo valutare: (1) il numero di istanze che sono state correttamente predette come positive (True positives – TP); (2) il numero di istanze che sono state erroneamente predette come positive (False Positives – FP); (3) il numero di istanze che sono state correttamente predette come negative (True negatives – TN); (4) il numero di istanze che sono state erroneamente predette come negative (False negatives – FN).

Tali valori consentono di definire alcune misure per valutare le prestazioni di un classificatore, di seguito vengono descritte le più comuni [14]. Ciascuna di esse assume valori compresi tra 0 (indice di prestazioni pessime) e 1 (indice di prestazioni ottime).

L'*accuratezza* (Acc) è il rapporto tra le istanze correttamente predette e il numero totale di istanze:

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

Questa misura, banalmente estendibile per valutare anche le prestazioni nella classificazione multiclasse, non è però significativa quando le classi sono sbilanciate, cioè in presenza di un numero di casi positivi e negativi molto diverso. Infatti, in tale circostanza, gli errori di classificazione effettuati sulle istanze appartenenti alla classe sotto-rappresentata hanno una scarsa incidenza nel calcolo della misura, che può assumere, perciò, valori poco indicativi.

La *precisione* (Pr) è il rapporto tra il numero di istanze correttamente predette come positive e il numero totale di istanze predette come positive:

$$Pr = \frac{TP}{TP + FP}.$$

Una precisione elevata è indice di pochi falsi positivi.

La *recall* (Re) o *tasso di veri positivi* (True positive Rate – TPR) è il rapporto tra il numero di istanze correttamente predette come positive e il numero totale di istanze effettivamente positive:

$$Re = \frac{TP}{TP + FN}.$$

Un alto valore di recall indica perciò un basso numero di falsi negativi.

Queste due ultime misure, prese singolarmente, non forniscono, però, informazioni sufficienti per stimare le prestazioni di un classificatore, in quanto non valutano simultaneamente il numero di falsi positivi e il numero di falsi negativi.

La misura f_1 (anche nota come f_1 *score*[4]) si ottiene combinando le due misure sopra descritte, essendo infatti definita come la media armonica della precisione e della recall:

$$f_1 = 2 \cdot \frac{Pr \cdot Re}{Pr + Re}.$$

Un elevato valore di f_1 è pertanto indice di un alto valore di precisione e di un alto valore di recall ma è indipendente dal numero di veri negativi. La misura f_1 può essere estesa per valutare le prestazioni nella classificazione multiclasse calcolando la media aritmetica o la media ponderata dei valori di f_1 ottenuti considerando di volta in volta ciascuna classe come la classe positiva e tutte le altre nel loro insieme come la classe negativa. La media ponderata dei valori di f_1 (f_1 *weighted*), calcolata cioè pesando i valori di f_1 per il numero di istanze di ciascuna classe, consente di effettuare una valutazione più precisa delle prestazioni del classificatore quando le classi sono sbilanciate.

Alcuni classificatori, oltre a classificare le istanze come positive o negative, possono calcolare anche la probabilità di positività, cioè la probabilità che una certa istanza

[4]La misura f_1 deriva dalla più generica misura f_β che attribuisce a precisione e recall un peso diverso tramite il valore del parametro β $\left(f_\beta = (1 + \beta^2) \cdot \frac{Pr \cdot Re}{\beta^2 \cdot Pr + Re}\right)$.

appartenga alla classe positiva. Per valutare in modo più accurato le prestazioni di queste tipologie di classificatori, è possibile costruire la curva ROC (Receiver Operating Characteristic[5]) [15]. Essa mostra la relazione esistente tra il tasso di falsi positivi (False Positive Rate, $FPR = \frac{FP}{FP+TN}$) e il tasso di veri positivi al variare della soglia di probabilità di positività.[6] In particolare, a ogni soglia di probabilità corrisponde un punto della curva le cui coordinate x e y sono rispettivamente il tasso di falsi positivi e il tasso di veri positivi, entrambi calcolati considerando positive solo le istanze che sono state predette tali con una probabilità maggiore o uguale alla soglia.

Due punti che convenzionalmente appartengono a ogni curva ROC sono i punti (1, 1) e (0, 0). Il primo si ottiene con una soglia di probabilità pari a 0, per cui tutte le istanze sono classificate come positive, mentre il secondo si ottiene con una qualsiasi soglia maggiore di 1, per cui tutte le istanze sono classificate come negative. Il punto (0, 1) indica una condizione ottimale, si ottiene infatti quando non vi sono né falsi negativi né falsi positivi. Un classificatore ottimale predice correttamente e con certezza (probabilità pari a 1) tutte le etichette, pertanto la sua curva presenta i soli punti (0, 0), (0, 1) e (1, 1). Invece, la curva ROC di un classificatore che associa a ogni istanza una probabilità di appartenenza alla classe positiva di 0.5 – dichiara cioè di non essere in grado di decidere l'appartenenza delle istanze alle classi – presenta i soli punti (0, 0) e (1, 1) (per soglie minori o uguali di 0.5 si ottiene il punto (0, 0), per soglie maggiori di 0.5 si ottiene il punto (1, 1)). Informalmente, più la curva ROC è "spostata" nella regione del piano prossima al punto (0, 1), migliore è il classificatore. La figura 1.1 mostra un esempio di curva ROC.

Le curve ROC presentano l'interessante proprietà di essere invarianti rispetto alla distribuzione dei dati nelle classi. Inoltre, esse non dipendono dalla calibrazione delle probabilità predette dal classificatore.

[5]Le curve ROC prendono questo nome in quanto furono utilizzate per la prima volta durante la seconda guerra mondiale per studiare l'accuratezza nella rilevazione dei segnali radar degli aerei militari.

[6]Più in generale la curva ROC può essere tracciata anche per i classificatori che non forniscono una probabilità in senso stretto ma uno "score", non necessariamente compreso tra 0 e 1, che rappresenta il grado con cui, secondo il classificatore, un'istanza appartiene a una classe.

Per riassumere in unica misura le prestazioni di un classificatore osservate attraverso la curva ROC possiamo considerare l'area sottesa alla curva (Area Under ROC curve – AUROC). Anche questa misura può essere estesa per valutare la performance di un classificatore in un task di classificazione multiclasse con una procedura analoga a quella descritta precedentemente per f_1.

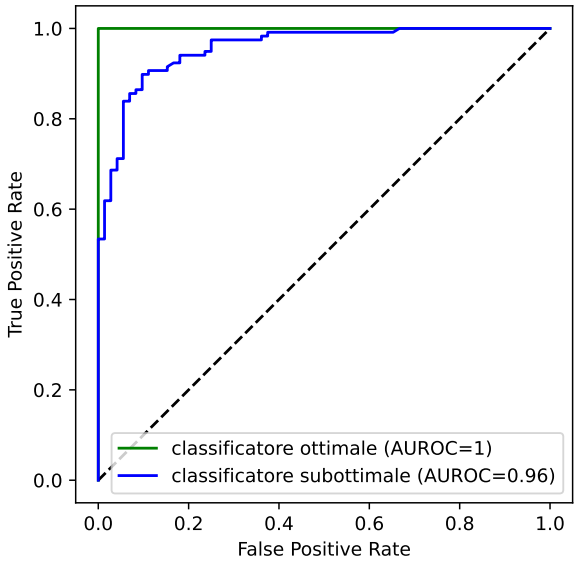

Figura 1.1: Curva ROC del classificatore ottimale e di un classificatore subottimale.

Capitolo 2

Machine learning e medicina

In questo capitolo si descrivono le possibili applicazioni del machine learning in medicina e si analizzano le difficoltà tecniche che insorgono nella progettazione e nell'utilizzo di queste metodiche. Successivamente, si presentano i fondamenti biologici alla base dei meccanismi di regolazione dell'espressione genica e si forniscono alcuni esempi di modelli di apprendimento automatico, proposti in letteratura, per l'analisi dei profili di espressione genica.

2.1 Il machine learning come supporto per la diagnosi e la prognosi

Con il termine *diagnosi*, dal greco "riconoscere attraverso", si indica il processo per mezzo del quale il medico riconosce una condizione patologica in un paziente sulla base dell'esame clinico del malato e dei referti degli esami strumentali e di laboratorio. La diagnosi di alcune malattie può risultare complicata in quanto i reperti diagnostici possono essere di difficile interpretazione e in certi casi solo il personale medico con molti anni di formazione ed esperienza alle spalle può giungere a una loro corretta valutazione. Basti pensare che l'Istituto di Medicina statunitense (IoM) dell'Accademia Nazionale delle Scienze, dell'Ingegneria e della Medicina in un report del 2015 giunge alla conclusione che la maggior parte delle persone sarà vittima di un errore diagnostico almeno una volta nel corso della propria vita [16].

In questo contesto l'utilizzo di sistemi di apprendimento automatico può rivelarsi molto utile per supportare i medici al fine di migliorare l'accuratezza diagnostica, evitare la prescrizione di esami inutili e l'esecuzione di interventi invasivi o erronei. Ciò può avere ricadute positive in termini economici, ma soprattutto può consentire di individuare in tempi rapidi terapie farmacologiche mirate ed efficaci [3, 4].

La *prognosi*, dal greco "conoscere prima" è, invece, il processo con cui il medico prevede il decorso e l'esito di una malattia, valutando la possibilità di guarigione, di eventuali recidive o ricadute e le relative tempistiche. Predire il decorso della malattia può permette di differenziare il trattamento farmacologico, riservando le terapie ad alto impatto o con elevati effetti collaterali avversi esclusivamente ai pazienti a maggior rischio di progressione della patologia. Nell'ottica di una personalizzazione delle cure è, quindi, necessario stratificare i pazienti, ovvero raggrupparli in sottogruppi in base alle loro caratteristiche peculiari, integrando dati genetici e clinici [17]. Anche a questi scopi il machine learning può rivelarsi uno strumento molto promettente [18].

Tra i metodi di apprendimento automatico, le strategie di clustering sono particolarmente idonee per l'individuazione di nuovi biomarcatori[1] o per la stratificazione dei pazienti. I classificatori, invece, possono supportare il medico nella predizione di esiti discreti come la presenza o l'assenza di una patologia, l'individuazione del sottotipo di una malattia o la predizione di una prognosi fausta o infausta. I regressori, infine, possono essere utilizzati per stimare valori numerici come il rischio di sviluppare una certa malattia o il tempo necessario alla guarigione [1].

Sebbene i metodi di apprendimento automatico appaiano teoricamente come un potente e preciso strumento al servizio della medicina, essi non sono stati ancora applicati su larga scala nella pratica clinica a causa di una serie di problematiche che si presentano nella loro progettazione e nel loro utilizzo [20–22].

[1] Un biomarcatore, secondo il Biomarkers Definitions Working Group [19], è una "caratteristica che viene oggettivamente misurata e valutata come indicatore di processi biologici normali o patogeni, oppure di risposte farmacologiche a interventi terapeutici".

2.2 Criticità

Diverse sono le criticità che si presentano nell'applicazione del machine learning in ambito medico. Una di queste è la limitata disponibilità dei dati dei pazienti. Infatti, spesso alcune informazioni non vengono digitalizzate, i pazienti possono negare il consenso all'utilizzo dei propri dati sanitari oppure il numero dei dati disponibili è oggettivamente limitato perchè i soggetti affetti dalla patologia in analisi sono effettivamente pochi [23]. Questa scarsa numerosità dei dati può facilmente condurre al fenomeno dell'overfitting: i modelli tendono ad apprendere le caratteristiche tipiche dei dati su cui sono stati allenati senza cogliere le caratteristiche più generali della patologia in esame. Tale rischio può essere ridotto validando i modelli con metodi statistici rigorosi e con il coinvolgimento e l'approvazione dei medici [22].

Per avere una visione più ampia di fenomeni biologici complessi è opportuno integrare dati eterogenei, come quelli provenienti da più coorti di pazienti,[2] quelli ottenuti con diversi protocolli di laboratorio o raccolti con l'utilizzo di tecnologie differenti. L'integrazione dei dati può però rivelarsi difficile a causa della loro natura non omogenea e dell'intrinseco rumore dei dati biologici [24]. Negli ultimi anni sono stati pubblicati diversi studi che propongono specifiche strategie per affrontare questo problema [25]. Tipicamente viene adottata una delle seguenti modalità: si normalizzano i dati disponibili e si utilizzano per costruire un solo modello (*early integration*) oppure si allena un modello per ogni dataset e si adoperano tutti modelli così costruiti per effettuare predizioni (*late integration* con modelli *ensemble*) [26].

Un altro aspetto critico che si può riscontrare è il problema dei cosiddetti *valori mancanti*, cioè il fatto che i valori di alcune feature potrebbero essere assenti. Sono molteplici e talvolta banali le ragioni per cui i dataset possono risultare incompleti: i valori di alcuni esami potrebbero non essere stati registrati o potrebbero essere stati smarriti nel trasferimento di un paziente da una clinica all'altra, oppure alcuni pazienti potrebbero non essere stati sottoposti agli esami più invasivi. La rimozione dal dataset dei pazienti con valori mancanti o delle feature che non sono state misurate per tutti i pazienti riduce la quantità di informazioni disponibili. È quindi

[2] Gruppi di pazienti accomunati da una o più caratteristiche che vengono monitorate nel tempo.

utile applicare specifici algoritmi per dedurre i valori delle feature mancanti [27].

Un'ulteriore criticità, nota in letteratura con l'espressione "small n, large p problem", riguarda l'alta dimensionalità dei dataset, il fatto cioè che il numero di pazienti n è molto minore del numero di feature p. Si rischia, infatti, di incorrere nella cosiddetta *curse of dimensionality* (la maledizione della dimensionalità), espressione coniata da Richard E. Bellman [28] con cui si indica la problematica che scaturisce dalla sparsità dei dati che limita la possibilità di individuare pattern in essi. Si devono, quindi, adottare tecniche di dimensionality reduction in modo da rendere i modelli maggiormente efficaci [29].

Infine, occorre evidenziare che spesso il responso fornito da alcuni sofisticati modelli di machine learning appare inesplicabile. Sarebbe invece opportuno che i medici potessero prendere decisioni avendo anche la possibilità di consultare le correlazioni individuate dagli algoritmi, senza doversi affidare esclusivamente all'esito da essi predetto [30].

2.3 L'espressione genica: dal DNA alle proteine

2.3.1 Contesto biologico

Il *DNA* (acido desossiribonucleico) [31] è una molecola costituita da due filamenti di *nucleotidi* appaiati e avvolti a spirale, presente nelle cellule di tutti gli esseri viventi. I nucleotidi sono costituiti da un gruppo fosfato, da una molecola di zucchero a 5 atomi di carbonio (il desossiribosio) e da una base azotata (adenina, guanina, citosina o timina). Alcune sequenze nucleotidiche del DNA, i *geni*, contengono le informazioni per la sintesi delle *proteine*, sostanze organiche necessarie per il funzionamento delle cellule. Il processo di sintesi delle proteine prevede la *trascrizione* delle informazioni contenute nei geni nell'*mRNA* (acido ribonucleico messaggero), una molecola a singolo filamento complementare al DNA. Le informazioni così trascritte nell'mRNA sono utilizzate nel processo di *traduzione*, che conduce alla produzione di proteine. Esistono diverse altre tipologie di molecole di RNA coinvolte nel processo di sintesi proteica e nella sua regolazione (mRNA, tRNA, rRNA, miRNA, ecc.).

Un gene è *attivo* o *espresso* in una cellula se viene tradotto nella proteina corrispondente. I geni sono gli stessi in tutte le cellule di un individuo ma, a seguito del processo di *differenziamento cellulare* che attiva o inibisce alcuni geni, tessuti diversi di un organismo esprimono geni distinti, dunque producono specifiche proteine che ne caratterizzano il funzionamento. Un gene può attivarsi o disattivarsi in una cellula anche a causa di una *mutazione*, cioè di un'alterazione della sequenza dei nucleotidi di cui è costituito, che può compromettere il corretto funzionamento cellulare. Le mutazioni possono essere ereditate ma è possibile che si verifichino anche nel corso della vita di un individuo a causa di molteplici fattori quali l'esposizione ad agenti nocivi, l'invecchiamento o lo stile di vita.

Per valutare quali geni sono attivi in un dato momento in una cellula è possibile studiare il *profilo di espressione genica*, ricercare cioè le proteine o le specifiche molecole di RNA coinvolte nel processo di sintesi proteica. Un'analisi di questo tipo consente di comprendere i ruoli funzionali dei geni e la loro partecipazione ai processi cellulari, permettendo di indagare i meccanismi alla base dei processi patologici e favorendo l'individuazione di trattamenti terapeutici.

Per misurare il livello di espressione genica sono disponibili diverse tecnologie, alcune delle più utilizzate sono i microarray e l'RNA-seq. La tecnologia dei *microarray di DNA* [32, 33] permette di analizzare contemporaneamente il livello di espressione di migliaia di geni in modo rapido ed economico. Un microarray è costituito da un supporto solido dotato di piccoli incavi, o pozzetti, su cui vengono depositati frammenti di DNA a singolo filamento, detti sonde o probe. Il principio su cui si basa questa tecnologia è quello dell'*ibridazione*, ovvero dell'appaiamento tra molecole complementari. I campioni da analizzare, marcati con coloranti fluorescenti e contenenti acidi nucleici a singolo filamento coinvolti nel processo di sintesi proteica, si legano alle sonde deposte sul microarray. Illuminando i pozzetti con un fascio di luce laser, in base all'intensità del segnale luminoso emesso dalle sonde, è possibile misurare la concentrazione delle sequenze geniche del campione e quindi stabilire i geni che in quel dato momento sono attivi. La più recente tecnologia *RNA-seq* [34] si basa invece sul sequenziamento dell'RNA, cioè sull'individuazione della sequenza con cui si succedono i nucleotidi. Tale tecnica ha diversi vantaggi rispetto ai microarray

in quanto permette di ottenere misure meno rumorose e non richiede la conoscenza a priori delle sequenze geniche necessarie per la produzione delle sonde, ha però costi sperimentali più alti.

2.3.2 Modelli computazionali per l'analisi dei profili di espressione genica

La letteratura registra un'elevata quantità di pubblicazioni relative all'applicazione di modelli di apprendimento automatico per l'analisi dei profili di espressione genica. Di seguito si riassumono i risultati di alcuni degli studi più rappresentativi.

Uno dei primi approcci alla classificazione di campioni biologici in base ai dati di espressione genica è quello di Golub et al. [35] e riguarda la distinzione tra la leucemia mieloide acuta e la leucemia linfocitica acuta. Queste due tipologie di leucemie hanno decorsi clinici differenti e rispondono in modo diverso alle terapie, ma risultano difficili da diagnosticare unicamente sulla base delle loro caratteristiche morfologiche perché presentano elevate similarità istopatologiche. Il metodo proposto in questo studio prevede l'individuazione dei "geni più informativi", ovvero quelli che mostrano una maggiore correlazione con le classi in analisi. In particolare ogni gene informativo fornisce un voto per una classe con un peso diverso a seconda del livello di espressione del gene nel campione e del grado di correlazione del gene con la classe. Se la somma dei voti ricevuti da una classe supera una determinata soglia, si classifica il campione con l'etichetta di tale classe.

Lo studio di Furey et al. [36] dimostra l'efficacia di Support Vector Machine nel distinguere tessuti tumorali da quelli sani sulla base dei profili di espressione genica e suggerisce che tale classificatore potrebbe essere utilizzato anche per identificare campioni precedentemente classificati in modo erroneo dai medici.

Brown et al. [37], invece, hanno utilizzato Support Vector Machine per identificare i geni con funzioni simili: allenando il modello con un insieme di geni di cui è noto se appartengono o meno a una certa classe funzionale, SVM apprende come distinguere i geni in base al loro livello di espressione e può predire la classe funzionale di appartenenza di geni per cui essa non è nota. Anche in questo caso SVM può essere

applicata nuovamente ai geni forniti come esempio per individuare eventuali geni associati alla classe funzionale sbagliata.

Van't Veer et al. [38] hanno applicato un algoritmo di classificazione per identificare una signature genica[3] capace di predire lo sviluppo di metastasi in un gruppo di giovani pazienti affette da tumore al seno con linfonodi negativi. Da questo studio è emerso che l'algoritmo di classificazione utilizzato fornisce esiti più accurati di quelli stimabili osservando esclusivamente i parametri clinici e permette quindi di selezionare le pazienti che potrebbero beneficiare di una terapia adiuvante (chemioterapia o terapia ormonale).

Ye et al. [39], con un algoritmo di apprendimento supervisionato, hanno analizzato il profilo di espressione genica in campioni di carcinoma epatocellulare ottenuti da pazienti positivi al virus dell'epatite B e hanno identificato una signature genica che consente di riconoscere i pazienti con metastasi intraepatiche. Questo studio ha permesso di individuare i geni coinvolti nel processo metastatico che possono dunque essere utilizzati come marcatori diagnostici e che costituiscono target terapeutici.

Tecniche di clustering sono, invece, tipicamente utilizzate per raggruppare i geni in base alle loro somiglianze nei pattern di espressione [40], ma si sono dimostrate utili anche per individuare nuovi sottotipi di patologie. Ad esempio, Perou et al. [41] hanno identificato nuove tipologie di carcinoma mammario, mentre lo studio di Alizadeh A.A. et al. [42] ha consentito di individuare una stratificazione dei pazienti affetti da linfoma diffuso a grandi cellule che riflette il differente decorso clinico della malattia.

[3]Insieme di geni con uno specifico pattern di espressione che caratterizza uno stato patologico.

Capitolo 3

Rank Aggregation

Questo capitolo presenta il concetto di rank aggregation. Dopo un breve excursus storico sulle origini del problema, vengono descritte alcune sue applicazioni e ne viene illustrata la formalizzazione matematica.

3.1 Le origini del problema

Il problema del *rank aggregation* è stato ampiamente trattato nell'ambito degli studi relativi alla *teoria della scelta sociale* e scaturisce dall'esigenza di aggregare le scelte degli individui, espresse sotto forma di liste preferenziali, in un'unica lista che meglio rappresenti l'opinione collettiva [43]. I metodi di rank aggregation sono tipicamente studiati per sviluppare sistemi di voto per le elezioni ma possono essere utilizzati in qualsiasi scenario in cui si presenta la necessità di prendere una decisione collettiva. A questo scopo sono stati proposti diversi metodi. Uno dei primi, che risale alla seconda metà del diciottesimo secolo, deriva dal sistema di voto per le elezioni politiche proposto da Jean-Charles de Borda come alternativa al sistema elettorale maggioritario [44]. Borda aveva infatti osservato che, in presenza di molti candidati, il sistema maggioritario può condurre alla vittoria di un candidato votato soltanto da un'esigua minoranza dell'elettorato. Con il sistema di voto ideato da Borda, invece, ciascun elettore è chiamato a esprimere le proprie preferenze sui candidati ordinandoli in una lista e viene eletto colui che mediamente si è posizionato al primo posto nelle liste delle preferenze degli elettori. Operativamente, con tale

metodo si sommano, per ciascun candidato, le posizioni che esso ha assunto nelle liste delle preferenze degli elettori. La lista di candidati che riflette le preferenze della collettività si ottiene, dunque, disponendo i candidati in ordine crescente in base ai valori delle somme delle posizioni assunte da ciascuno. Viene decretato vincitore delle elezioni il primo candidato di tale lista.

3.2 Alcune applicazioni

Disporre di più giudizi o criteri con cui ordinare un insieme di alternative e avere la necessità di combinare le liste ottenute al fine di produrre un'unica lista ordinata che sia più attendibile è un problema che si presenta in svariati contesti. Di seguito si descrivono alcune applicazioni proposte in letteratura.

Nello studio di Dwork et al. [45] vengono illustrati alcuni metodi per aggregare le liste di pagine web prodotte da motori di ricerca diversi o ordinate in base a differenti criteri, al fine di combattere il problema dello *spam* e di produrre risultati quanto più soddisfacenti per gli utenti utilizzatori.

Il lavoro di Kold et al. [46], che si colloca nell'ambito della bioinformatica, descrive e valuta l'impiego del rank aggregation per integrare liste di geni ordinate in base a differenti criteri o i cui livelli di espressione sono stati misurati con tecnologie diverse in un'unica lista ordinata da cui poter trarre conclusioni più attendibili circa la correlazione tra i geni e la patologia in analisi.

Numerosi studi nel campo del machine learning hanno suggerito l'utilizzo di metodi di rank aggregation finalizzati alla costruzione di liste di feature più robuste per effettuare feature selection con approcci filter; si vedano ad esempio i lavori di Prati [47] e di Onan et al. [48].

3.3 Formalizzazione

Il problema del rank aggregation può essere formalizzato come un problema di ottimizzazione: stabilito un criterio per misurare la distanza d tra due liste ordinate di alternative, l'obiettivo è quello di individuare, tra tutte le possibili liste ordinate σ,

la lista σ' per cui è minima la somma delle distanze tra σ' e le liste ordinate $\pi_1, ..., \pi_k$ che intendiamo aggregare, cioè

$$\sigma' = \text{argmin}_\sigma \sum_{i=1}^{k} d(\sigma, \pi_i). \tag{3.1}$$

Per determinare la distanza tra due liste ordinate di alternative tipicamente si utilizza la distanza di Spearman [49] o la distanza di Kendall [50], misure utilizzate anche in statistica per determinare la correlazione tra due variabili. Le definizioni di tali misure sono fornite di seguito insieme alla notazione necessaria per la loro definizione.

Dato un insieme di alternative $A = \{a_1, \ldots, a_n\}$, una lista ordinata delle alternative di A è una permutazione $\pi = \langle a'_1, \ldots, a'_n \rangle$ degli elementi di A. Indichiamo con $\pi(a_i)$ la posizione assunta dall'alternativa a_i nella permutazione π.

La *distanza di Spearman* tra le liste ordinate π e ρ di un insieme di n alternative misura la differenza delle posizioni assunte dalle alternative nelle liste, dunque

$$\mathcal{F}(\pi, \rho) = \sum_{i}^{n} |\pi(i) - \rho(i)|. \tag{3.2}$$

La distanza di Spearman può essere calcolata in tempo lineare e sono noti algoritmi polinomiali per individuare la lista aggregata ottima secondo tale distanza [45].

La *distanza di Kendall*, invece, conta le coppie di alternative che nelle due liste ordinate assumono posizioni discordanti, ovvero

$$\mathcal{K}(\pi, \rho) = |\{(i, j) \mid (\pi(i) < \pi(j) \wedge \rho(i) > \rho(j)) \text{ per } i, j = 1, ..., n \text{ con } i \neq j\}|. \tag{3.3}$$

La distanza di Kendall viene anche chiamata *distanza bubble sort*, in quanto coincide con il minimo numero di coppie di elementi adiacenti che devono essere scambiati per trasformare una lista nell'altra. Utilizzando semplici strutture dati, con un algoritmo basato sul *merge sort*, è possibile calcolare questa misura in tempo computazionale $O(n \log n)$. La lista aggregata ottima secondo la distanza di Kendall viene indicata in letteratura con l'espressione *aggregazione ottima di Kemeny* [51] ed è stato dimostrato che calcolare tale lista è un problema NP-hard [45]. Per individuare soluzioni approssimate al problema sono stati proposti numerosi algoritmi euristici [45, 52–54].

Capitolo 4

Classificazione tramite rank aggregation: Rank Aggregation Classifier (RAC)

In questo capitolo viene descritto RAC, il classificatore basato sul concetto di rank aggregation oggetto di studio di questo lavoro di tesi. Oltre a fornire alcuni dettagli dell'implementazione del classificatore si discutono le caratteristiche che lo rendono appropriato per la classificazione di profili di espressione genica e si analizzano gli aspetti che lo distinguono da altri classificatori proposti nella letteratura.

4.1 Descrizione del classificatore

Il principio su cui si basa RAC è analogo a quello adottato dal classificatore Nearest Centroid: le istanze non note vengono confrontate con gli esemplari rappresentativi delle classi, costruiti a partire dagli esempi nella fase di allenamento. RAC, però, non opera sugli effettivi valori assunti dalle feature, ma sui loro ranghi. La fase di allenamento del classificatore (algoritmo 1) prevede, infatti, la *trasformazione in ranghi* delle feature e l'*aggregazione* dei vettori di ranghi degli esempi appartenenti alla stessa classe in un unico vettore di ranghi rappresentativo (*class signature*). Per classificare nuove istanze (algoritmo 2) esse vengono trasformate in ranghi, confrontate con le class signature ed etichettate con l'etichetta della class signature

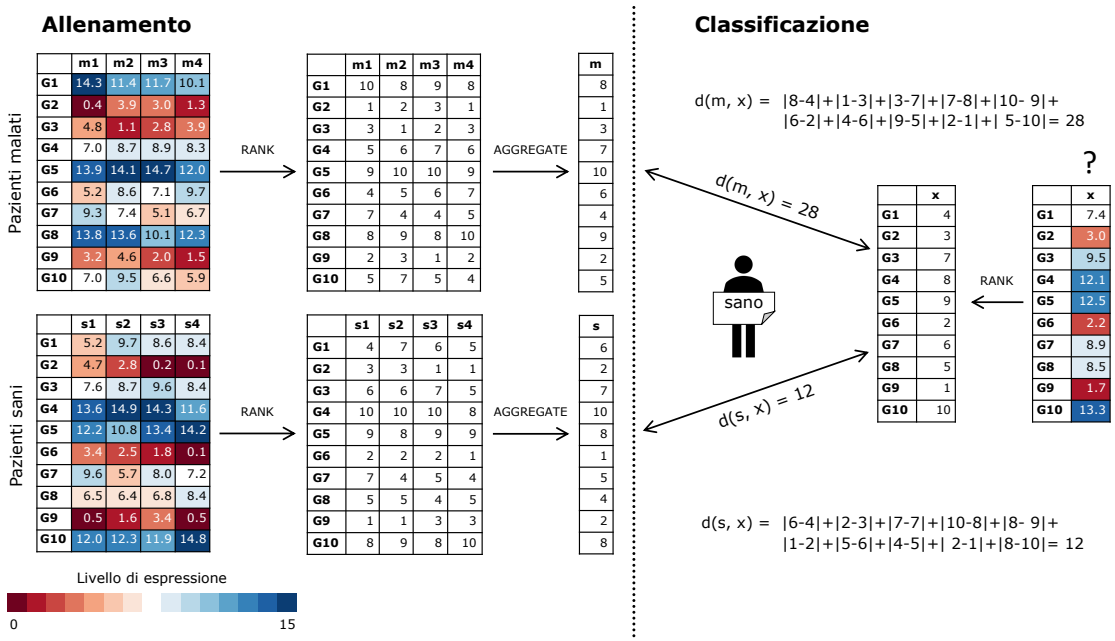

Figura 4.1: Esempio "giocattolo" di applicazione di RAC su un dataset di espressione genica con 10 feature (geni) e 8 campioni (4 di pazienti malati e 4 di pazienti sani). Le matrici a sinistra sono heat map: il livello di espressione rilevato per ciascun gene nell'esperimento di microarray è associato a un colore che varia dal rosso (basso livello di espressione) al blu (alto livello di espressione). In questo esempio RAC utilizza il metodo di assegnazione dei ranghi min (sezione 4.1.1), il metodo di aggregazione Borda (sezione 4.1.2) e la distanza di Spearman (equazione 3.2). Dal momento che il vettore dei ranghi del paziente da classificare risulta essere più vicino alla signature della classe dei pazienti sani, RAC classifica il paziente come sano.

a cui risultano essere più "simili". Le seguenti sezioni chiarificano la notazione appena adottata per descrivere il funzionamento di RAC e forniscono ulteriori dettagli sull'implementazione del classificatore. La figura 4.1 illustra un esempio di applicazione di RAC.

4.1.1 Trasformazione in ranghi

Per trasformare in ranghi gli esempi si attribuisce a ciascuna feature la posizione che essa assume nella lista ottenuta disponendo le feature in ordine crescente in base al loro valore. Più formalmente, dato un vettore $x = [x_1, \ldots, x_n]$ privo di elementi uguali, ovvero tale che $x_i \neq x_j$ per $i, j = 1, \ldots, n$ con $i \neq j$ e dato $o = [x'_1, x'_2, \ldots, x'_n]$ con $x'_1 < x'_2 < \cdots < x'_n$ il vettore ottenuto disponendo gli elementi di x in ordine crescente, si definisce *rango* di un elemento x_i di x la posizione occupata da x_i nel

vettore o, dunque

$$\text{rango}(x_i) = j \Leftrightarrow o_j = x_i \text{ per } i, j = 1, \ldots, n. \tag{4.1}$$

Nel caso in cui il vettore x presenti elementi uguali, il loro rango può essere definito come la media aritmetica delle posizioni occupate da essi nel vettore o (metodo *average*), oppure come la minima o la massima posizione occupata da essi nel vettore o (metodo *min* e metodo *max*). Stabilita la modalità di definizione del rango per gli elementi uguali, è possibile definire il *vettore dei ranghi* di x come il vettore r tale che

$$r_i = \text{rango}(x_i) \text{ per } i = 1, \ldots, n. \tag{4.2}$$

Se, ad esempio, $x = [0, 2, 3, 2]$, il vettore dei ranghi di x calcolato con il metodo *average* è $r_{avg} = [1, 2.5, 4, 2.5]$, il vettore dei ranghi di x calcolato con il metodo *min* è $r_{min} = [1, 2, 4, 2]$ e il vettore dei ranghi di x calcolato con il metodo *max* è $r_{max} = [1, 3, 4, 3]$.

Nonostante la trasformazione in ranghi delle istanze attuata dal classificatore determini una perdita di informazione circa il valore quantitativo di ciascuna feature, essa può essere considerata una pratica di trattamento dei dati resistente al rumore. Infatti, utilizzando l'informazione relativa all'ordinamento tra le feature di un esempio, eventuali variabilità nei dati hanno un impatto minore sia nella fase di allenamento che in quella di classificazione.

Una tale trasformazione restringe, però, l'ambito di applicazione del classificatore. Questa procedura è, infatti, applicabile solo se le feature del dataset a cui il classificatore viene applicato sono tra loro uniformi, sono cioè misurate con la stessa unità di misura. Il classificatore, inoltre, è applicabile nei contesti in cui l'ordine tra le feature cambia negli esempi appartenenti a classi diverse.

4.1.2 Metodi di rank aggregation

Per costruire le class signature i vettori dei ranghi degli esempi appartenenti alla stessa classe devono essere aggregati in un unico vettore di ranghi. A questo scopo sono stati implementati quattro metodi *posizionali* [55]: il metodo di Borda e le tre sue varianti Borda *median*, Borda *gmean* e Borda *l2*. Questi metodi utilizzano

una funzione di aggregazione – rispettivamente la somma, la mediana, la media geometrica $\left(\sqrt[n]{x_1 \cdots x_n}\right)$ o la norma euclidea $\left(\sqrt{x_1^2 + \cdots + x_n^2}\right)$ – per ottenere un valore aggregato delle posizioni assunte da ciascuna feature nelle liste ordinate degli esempi. In particolare, se $r_1, ..., r_k$ sono i vettori di ranghi di k esempi di n feature appartenenti alla stessa classe e ψ è la funzione di aggregazione propria del metodo di rank aggregation, la class signature s si calcola come

$$s_i = \text{rango}(\psi(r_{1i}, \ldots, r_{ki})) \text{ per } i = 1, \ldots, n. \tag{4.3}$$

Questi metodi producono approssimazioni dell'aggregazione ottima di Kemeny [53], sono semplici da implementare, hanno un basso costo computazionale in tempo e la loro efficacia è stata provata in diversi studi [56, 57].

4.1.3 Misure di distanza

Per classificare un esempio è necessario calcolare la distanza tra il suo vettore di ranghi e le class signature, in modo da individuare la signature della classe più simile. Per misurare queste distanze il classificatore può essere configurato affinché utilizzi: (1) la distanza di Spearman (equazione 3.2), (2) una versione pesata della distanza di Spearman (equazione 4.4) o (3) il coefficiente di correlazione di Kendall (equazione 4.7).

La distanza pesata di Spearman

La versione pesata della distanza di Spearman consente di attribuire maggior rilievo alle feature che si posizionano in testa e/o in coda alla lista aggregata rappresentativa di ogni classe, per mezzo di un vettore dei pesi w associato a ciascuna class signature. In particolare, se r è il vettore dei ranghi dell'esempio da classificare e s una class signature, entrambi di dimensione n, la distanza pesata di Spearman tra r e s è definita come

$$\mathcal{F}_w(r, s) = \sum_{i}^{n} |r_i - s_i| \cdot w_i. \tag{4.4}$$

Sono stati implementati due metodi di pesatura con cui definire i vettori dei pesi da associare a ciascuna class signature, che possono essere specificati attraverso i

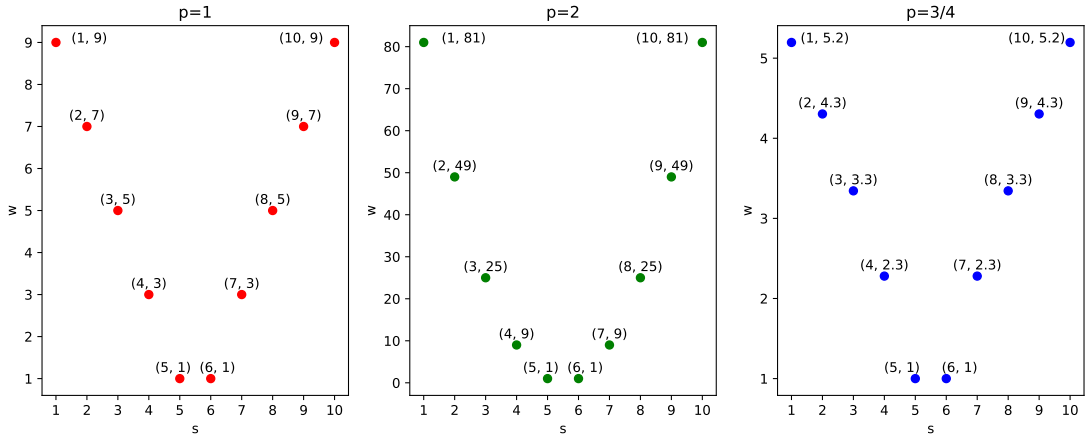

Figura 4.2: Un esempio dei valori dei pesi attribuiti ad una class signature che presenta 10 feature a cui è stato applicato il metodo di pesatura descritto dall'equazione 4.5 con tre diversi valori del parametro p.

parametri del classificatore. Con il primo metodo di pesatura, indicando con s una class signature, il vettore dei pesi w ad essa associato è definito come

$$w_i = |\max(s) + 1 - 2s_i|^p \text{ per } i = 1, \ldots, n, \tag{4.5}$$

dove p è un numero reale positivo anch'esso configurabile. In tal modo, il peso associato alle feature poste agli estremi della lista è massimo e diminuisce spostandosi verso le posizioni centrali, dove assume valore minimo (figura 4.2). Si noti che il massimo elemento del vettore s non è necessariamente n, ma dipende dal metodo di definizione del rango e dalla presenza di feature con lo stesso rango in s.

Alternativamente, è possibile configurare il classificatore in modo che il vettore dei pesi sia definito come

$$w_i = \begin{cases} 1 & \text{se } s_i \leq n_1 \vee s_i > \max(s) - n_2 \\ 0 & \text{altrimenti} \end{cases} \quad \text{per } i = 1, \ldots, n \tag{4.6}$$

dove n_1 e n_2 sono interi positivi anch'essi configurabili come parametri del classificatore. Con questo metodo di pesatura è quindi possibile specificare, attraverso il valore dei parametri n_1 e n_2, quante feature dovranno essere utilizzate in testa e/o in coda alla lista per calcolare le distanze.

I metodi di pesatura implementati consentono di effettuare una forma di *embedded feature selection* e sono stati pensati per essere applicati principalmente a dataset di

espressione genica. In tale contesto, permettono di attribuire maggior rilievo ai geni che hanno un alto o un basso livello di espressione.

Il coefficiente di correlazione di Kendall

Il coefficiente di correlazione di Kendall, indicato tipicamente con la lettera dell'alfabeto greco τ, è una misura che deriva dalla distanza di Kendall. Esso, infatti, assume valori compresi tra -1 (massima discordanza tra i vettori di ranghi) e 1 (massima concordanza tra i vettori di ranghi). La definizione del coefficiente di correlazione di Kendall [58] che è stata utilizzata nell'implementazione del classificatore è la seguente

$$\tau(r,s) = \frac{|P| - |Q|}{\sqrt{(|P| + |Q| + |T|) \cdot (|P| + |Q| + |U|)}}, \tag{4.7}$$

dove, se r e s sono i due vettori di ranghi di n elementi di cui si intende calcolare il coefficiente di correlazione, si ha che

$$P = \{(i,j) \mid (r_i < r_j \wedge s_i < s_j) \vee (r_i > r_j \wedge s_i > s_j) \text{ per } i,j = 1,\ldots,n \text{ con } i < j\}$$

è l'insieme delle coppie concordanti in r e s,

$$Q = \{(i,j) \mid (r_i < r_j \wedge s_i > s_j) \vee (r_i > r_j \wedge s_i < s_j) \text{ per } i,j = 1,\ldots,n \text{ con } i < j\}$$

è l'insieme delle coppie discordanti in r e s (la cardinalità di Q equivale alla distanza di Kendall, $|Q| = \mathcal{K}(r,s)$),

$$T = \{(i,j) \mid r_i = r_j \wedge s_i \neq s_j \text{ per } i,j = 1,\ldots,n \text{ con } i < j\}$$

è l'insieme delle coppie che hanno stesso rango solo in r e

$$U = \{(i,j) \mid s_i = s_j \wedge r_i \neq r_j \text{ per } i,j = 1,\ldots,n \text{ con } i < j\}$$

è l'insieme delle coppie che hanno stesso rango solo in s. Il coefficiente di Kendall così definito consente di trattare opportunamente anche la presenza di elementi con stesso rango nei vettori da confrontare.

4.1.4 Utilizzo dei centroidi

Nel caso in cui le class signature di classi distinte coincidano e la distanza tra il vettore dei ranghi dell'istanza da classificare e le class signature coincidenti è minima, l'istanza verrebbe classificata in modo arbitrario con una delle etichette delle classi le cui signature coincidono. Per ovviare a questa problematica, la fase di allenamento del classificatore prevede anche il calcolo dei centroidi di ciascuna classe e nell'eventualità sopra descritta si calcolano le distanze euclidee tra l'istanza da classificare e i centroidi delle sole classi equidistanti, in modo da attribuire all'istanza l'etichetta della classe il cui centroide è il più vicino.

Più formalmente, data la matrice x di dimensioni $m \times n$ (m esempi ciascuno di n feature), in presenza di k classi, indicando con C_j l'insieme degli indici di x che corrispondono agli m_j esempi della classe j, il centroide della classe j è definito come

$$\overline{x}_{ji} = \frac{1}{m_j} \sum_{l \in C_j} x_{li} \text{ per } i = 1, \ldots, n. \tag{4.8}$$

Se $d = [d_1, \ldots, d_k]$ è un vettore in cui ciascun d_j rappresenta la distanza tra la signature della classe j e il vettore di ranghi r dell'esempio da classificare s e I è l'insieme degli indici degli elementi minimi in d ($I = \{j \mid d_j = \min(d)\}$), quando $|I| > 1$ per ogni $u \in I$ si calcola la distanza euclidea tra $\overline{x}_u$ e s e si attribuisce a s l'etichetta della classe v la cui distanza euclidea è minima, ovvero

$$v = \text{argmin}_{u \in I} \left(\sqrt{\sum_{i=1}^{n} (\overline{x}_{ui} - s_i)^2} \right). \tag{4.9}$$

4.1.5 Probabilità di appartenenza delle istanze alle classi

Avere una misura della "sicurezza" con cui un classificatore predice la classe di appartenenza di un'istanza consente di valutare in modo più accurato le prestazioni del classificatore. Utilizzando RAC, per stimare la probabilità di appartenenza di un'istanza da classificare alle classi del problema possiamo confrontare tra loro le distanze tra il vettore di ranghi dell'istanza e le signature delle classi: la classe più probabile è quella la cui signature è la meno distante dal vettore di ranghi dell'istanza da classificare.

Algoritmo 1 Allenamento di RAC

Input:

Una matrice X di esempi di dimensione $n \times m$ (n esempi con m feature)
Un vettore y di dimensione n con le etichette associate agli esempi di X
Un metodo `rank` di assegnazione del rango
Un metodo `aggregate` di rank aggregation
Un metodo `weight` di pesatura

Output:

Il modello M allenato

// Individua le etichette distinte in y
$M.y \leftarrow \texttt{unique}(y)$
// Memorizza le modalità di configurazione
$M.\texttt{rank} \leftarrow \texttt{rank}$
$M.\texttt{aggregate} \leftarrow \texttt{aggregate}$
$M.\texttt{weight} \leftarrow \texttt{weight}$
// Per ogni etichetta
for $i \leftarrow 1, |M.y|$ **do**
    // Raccogli nella matrice C_i gli esempi con la stessa etichetta
    $C_i \leftarrow$ gli esempi di X con etichetta $M.y_i$
    // Calcola la class signature
    $S_i \leftarrow M.\texttt{aggregate}(M.\texttt{rank}(C_i))$
    // Calcola il centroide
    $\overline{X}_i \leftarrow \texttt{mean}(C_i)$
    // Calcola i pesi associati alla class signature
    $W_i \leftarrow M.\texttt{weight}(S_i)$
end for
return M
// $M.y$ è il vettore delle etichette delle classi che il classificatore è in grado di riconoscere
// $M.S$ è una matrice di dimensione $c \times m$ (ciascuna $M.S_i$ è una class signature)
// $M.W$ è una matrice di dimensione $c \times m$
// (ciascun $M.W_i$ è il vettore dei pesi associati ad una class signature)
// $M.\overline{X}$ è una matrice di dimensione $c \times m$ (ciascun $M.\overline{X}_i$ è il centroide di una classe)

Algoritmo 2 Classificazione di RAC

Input:

Il modello allenato M

Una misura di distanza `rank_distance` per confrontare vettori di ranghi

Una matrice X di istanze da classificare di dimensione $n \times m$ (n istanze con m feature)

Output:

Un vettore y di dimensione n con le etichette predette per le istanze in X

// Per ogni istanza da classificare
for $i \leftarrow 1, n$ **do**
  // Calcola il vettore dei ranghi dell'istanza da classificare X_i
  $r \leftarrow M.\texttt{rank}(X_i)$
  for $j \leftarrow 1, |M.y|$ **do**
    // Calcola la distanza tra la class signature e r
    $d_j \leftarrow \texttt{rank\_distance}(M.S_j, r)$
  end for
  // Individua le classi le cui signature hanno distanza minima da X_i
  $I \leftarrow \text{argmin}_j(d_j)$
  if $|I| > 1$ **then**
    // Individua tra le classi le cui signature hanno distanza minima da r
    // quella che ha il centroide più vicino a X_i usando la distanza euclidea
    $k \leftarrow \text{argmin}_{u \in I} \left(\sqrt{\sum_{j=1}^{n} (M.\overline{X}_{uj} - M.S_{uj})^2} \right)$
  else
    $k \leftarrow$ l'unico elemento di I
  end if
  $y_i \leftarrow M.y_k$
end for
return y

Date le classi $1, \ldots, k$, e la distanza d_i tra il vettore dei ranghi dell'istanza da classificare e la signature della classe i, la probabilità p_i che l'istanza da classificare appartenga alla classe i può essere calcolata nel modo seguente

$$p_i = \begin{cases} \dfrac{1 - \dfrac{d_i}{\sum_{j=1}^{k} d_j}}{\sum_{l=1}^{k}\left(1 - \dfrac{d_l}{\sum_{m=1}^{k} d_m}\right)} = \dfrac{1 - \dfrac{d_i}{\sum_{j=1}^{k} d_j}}{k-1} & \text{se } \sum_{j=1}^{k} d_j \neq 0 \\ \dfrac{1}{k} & \text{altrimenti} \end{cases} \tag{4.10}$$

Nel caso in cui, però, l'istanza da classificare sia equidistante da più class signature e la distanza da esse sia minima, per determinare la classe a cui essa appartiene si fa uso anche delle distanze dai centroidi delle classi da cui l'istanza è equidistante. Pertanto, in tal caso, utilizzando la notazione della sezione 4.1.4, per ogni $u \in I$, note le distanze euclidee e_u tra i centroidi $\overline{x}_u$ e l'esempio da classificare, è possibile utilizzare ancora l'equazione 4.10 per trasformare tali distanze in probabilità p'_u e, data la massima probabilità $p_l = \max\{p_i \mid \text{ per } i = 1, \ldots, k\}$ e la seconda probabilità più grande $p_{2l} = \max\{p_i | \text{ per } i \notin I\}$, la probabilità p''_i che l'esempio appartenga alla classe i può essere definita nel modo seguente

$$p''_i = \begin{cases} p_{2l} + p'_i(p_l|I| - p_{2l}|I|) = p_{2l} + |I|p'_i(p_l - p_{2l}) & \text{se } i \in I \\ p_i & \text{se } i \notin I \end{cases} \tag{4.11}$$

Nell'appendice A viene mostrato un esempio di applicazione del metodo di calcolo appena presentato e vengono descritti alcuni risultati sperimentali ottenuti con esso.

Si deve osservare, però, che le probabilità così calcolate difficilmente si avvicinano agli estremi 0 e 1. Per esempio, nell'eventualità in cui un'istanza da classificare coincida con la class signature di una classe (dunque la distanza da essa sia 0) e $\sum_{j=1}^{k} d_j \neq 0$, la probabilità che l'istanza appartenga a quella classe è pari a $\frac{1}{k-1}$, quando invece sarebbe auspicabile che fosse 1. Potrebbe rivelarsi utile, in uno studio futuro, affinare tali stime di probabilità per meglio apprezzare la separazione tra le classi.

```python
import numpy as np
from sklearn.base import BaseEstimator, ClassifierMixin
from sklearn.utils.validation import check_X_y, check_array, check_is_fitted
from sklearn.utils.multiclass import unique_labels
from scipy.stats import rankdata, gmean, kendalltau
from numpy.linalg import norm

class RAClassifier(BaseEstimator, ClassifierMixin):

    def __init__(
    self,
    r_method='min',
    ra_method='borda',
    metric='spearman',
    weighted=False,
    p=1):
        self.r_method = r_method
        self.ra_method = ra_method
        self.metric = metric
        self.weighted = weighted
        self.p = p

    def fit(self, X, y):
        ⋮


    ⋮
```

Listato 1: La classe che implementa RAC.

4.2 Implementazione del classificatore

L'implementazione del classificatore è compatibile con la API della libreria Python scikit-learn[1] [59]. Il codice sorgente di RAC è reperibile su GitHub al seguente link https://github.com/iretes/RAC.

Il classificatore è implementato dalla classe `RAClassifier` (listato 1). Il costruttore di tale classe presenta cinque parametri opzionali che rappresentano gli iperparametri del classificatore:

- Il parametro `r_method` consente di specificare il metodo di definizione dei ranghi per le feature che hanno lo stesso valore. Può assumere una delle seguenti stringhe: `'average'`, `'min'` o `'max'` (vedi sezione 4.1.1).
- Il parametro `ra_method` consente di specificare il metodo di rank aggregation da utilizzare per costruire le class signature. Può assumere una delle seguen-

[1] Scikit-learn è una libreria open source di machine learning per il linguaggio di programmazione Python che offre l'implementazione dei più comuni algoritmi di apprendimento automatico e che mette a disposizione strumenti per il preprocessing e la validazione di un modello.

ti stringhe: `'borda'`, `'borda_median'`, `'borda_gmean'` o `'borda_l2'` (vedi sezione 4.1.2).

- Il parametro `metric` consente di specificare la misura di distanza da utilizzare per calcolare le distanze tra il vettore di ranghi dell'istanza da classificare e le class signature. Può assumere valore `'spearman'` o `'kendall'` (vedi sezione 4.1.3).

- Il parametro `weighted` consente di specificare la modalità di pesatura da utilizzare nel calcolo delle distanze tra il vettore di ranghi dell'istanza da classificare e le class signature (vedi sezione 4.1.3). Può essere un booleano o una tupla di due interi. Se `metric='spearman'` e `weighted=False` viene utilizzata la distanza di Spearman (equazione 3.2), se `weighted=True` viene utilizzata la distanza di Spearman con la modalità di pesatura descritta dall'equazione 4.5; se, invece, `metric='spearman'` e `weighted` è una tupla di interi, viene utilizzato il metodo di pesatura descritto dall'equazione 4.6 con i valori di n_1 e n_2 specificati nella tupla. Quando `metric='kendall'` il valore del parametro viene ignorato.

- Il parametro `p`, significativo solo quando `weighted=True` e `metric='spearman'`, consente di specificare l'esponente dell'equazione 4.5.

I principali metodi implementati dalla classe `RAClassifier` sono descritti di seguito:

- Il metodo `fit` (listato 2) consente di allenare il classificatore. Esso prende in input una matrice di esempi e un array con le etichette associate agli esempi. Questo metodo calcola per ciascuna classe le signatures, i centroidi e i pesi, restituendo il classificatore allenato. Il calcolo delle signature di ciascuna classe viene effettuato nel metodo `aggregate` (listato 3) il quale trasforma gli esempi appartenenti alla stessa classe in vettori di ranghi con il metodo di assegnazione del rango definito dal parametro `r_method` e aggrega le liste così ottenute in base al metodo di rank aggregation definito dal parametro `ra_method`, secondo l'equazione 4.3. Il calcolo dei centroidi di ciascuna classe viene effettuato nel metodo `compute_centroid` (listato 3) in accordo all'equazione 4.8. Il calcolo

```python
def fit(self, X, y):
    # Check that X and y have correct shape
    X, y = check_X_y(X, y)
    # Store the features seen during fit
    self.n_features_in_ = X.shape[1]
    # Store the classes seen during fit
    self.classes_ = unique_labels(y)
    # Store samples and targets
    self.X_ = X
    self.y_ = y

    # Parameters validation

    ⋮

    # Compute signature, centroid and weights for each class
    self.class_signatures_ = np.empty((len(self.classes_), self.n_features_in_))
    self.class_centroids_ = np.empty((len(self.classes_), self.n_features_in_))
    self.weights_ = np.ones((len(self.classes_), self.n_features_in_))
    for i in range(len(self.classes_)):
        self.class_signatures_[i] = self.aggregate(self.X_[self.y_ == self.classes_[i]])
        self.class_centroids_[i]= self.compute_centroid(self.X_[self.y_ == self.classes_[i]])
        if self.weighted:
            self.weights_[i] = self.compute_weights(self.class_signatures_[i])

    # Return the classifier
    return self
```

Listato 2: Il metodo per allenare RAC.

dei pesi viene effettuato nel metodo `compute_weights` (listato 3) in base al valore del parametro `weighted`, con una delle modalità descritte dalle equazioni 4.5 (listato 3, linea 26) e 4.6 (listato 3, linee 23–24).

- Il metodo `predict` (listato 4) consente di ottenere le classi di appartenenza predette dal classificatore per istanze non note. Esso prende in input una matrice di istanze da classificare e restituisce un array con le etichette predette per ogni istanza. Internamente invoca il metodo `distances_to_signatures` (listato 5), che trasforma le istanze da classificare in vettori di ranghi e calcola le distanze di queste dalle class signature, facendo uso della distanza di Spearman e dei pesi calcolati nel metodo `fit`, in accordo con l'equazione 4.4 (listato 5, linee 11–14), oppure facendo uso del coefficiente di correlazione di Kendall (equazione 4.7) cambiato di segno e incrementato di 1 (listato 5, linea 16). Se ci sono più classi a distanza minima dall'istanza da classificare, questa viene classificata calcolando le distanze euclidee dai centroidi, come descritto dall'equazione 4.9.

```python
def aggregate(self, X):
    # Rank features
    r = rankdata(X, self.r_method, axis=1)

    # Aggregate ranks
    a = np.empty(X.shape[1])
    if self.ra_method == 'borda':
        a = rankdata(np.sum(r, axis=0), method=self.r_method)
    elif self.ra_method == 'borda_median':
        a = rankdata(np.median(r, axis=0), method=self.r_method)
    elif self.ra_method == 'borda_gmean':
        a = rankdata(gmean(r, axis=0), method=self.r_method)
    else: # self.ra_method == 'borda_l2'
        a = rankdata(norm(r, 2, axis=0), method=self.r_method)
    return a

def compute_centroid(self, X):
    return X.mean(axis=0)

def compute_weights(self, signature):
    max_rank = np.max(signature)
    if isinstance(self.weighted, tuple):
        return ((signature <= self.weighted[0]) | \
                (signature > max_rank-self.weighted[1])).astype(int)
    else:
        return np.power(np.abs(max_rank+1-2*signature), self.p)
```

Listato 3: I metodi di RAC per il calcolo della signature, del centroide e dei pesi di una classe.

- Il metodo `predict_proba` (listato 6) consente di ottenere le probabilità di appartenenza delle istanze alle classi. Esso prende in input una matrice di istanze da classificare e restituisce una matrice con le probabilità di appartenenza alle classi predette per ciascuna istanza. Questo metodo utilizza internamente il metodo `convert_distances_to_probas` (listato 6), che calcola le probabilità in accordo con l'equazione 4.10. Se ci sono più classi a distanza minima dall'istanza da classificare, le probabilità vengono calcolate facendo uso delle distanze euclidee tra il campione e i centroidi, come descritto dall'equazione 4.11.

Nel listato 7 viene mostrato un esempio di utilizzo del classificatore applicato al noto dataset degli Iris[2], che evidenzia la sua completa compatibilità con scikit-learn e il suo facile utilizzo.

[2]Tale dataset è costituito da 150 esempi del fiore iris definiti da 4 feature (lunghezza e larghezza in centimetri del petalo e del sepalo), suddivisi in 3 classi (le specie *Setosa*, *Versicolor* e *Virginica*). Le misure dei fiori furono effettuate dal botanico Edgar Anderson nel 1936 [60] e uno dei primi studi statistici su questi dati fu condotto da Ronald Fisher [61].

```python
def predict(self, X):
    # Check if fit had been called
    check_is_fitted(self, ['X_', 'y_'])

    # Input validation
    X = check_array(X, ensure_min_features=self.n_features_in_)

    # Predict label of samples according to the distances to signatures
    d = self.distances_to_signatures(X)
    closest = np.empty((X.shape[0]), dtype=int)
    for i in range(X.shape[0]):
        closers = np.where(d[i]==d[i].min())[0]
        closest[i] = closers[0]
        if closers.size > 1: # Determine nearest centroid (using euclidean distance)
            closest[i] = closers[
                np.argmin(np.linalg.norm(X[i]-self.class_centroids_[closers], axis=1))
            ]
    return self.classes_[closest]
```

Listato 4: Il metodo di RAC per classificare istanze.

```python
def distances_to_signatures(self, X):
    d = np.zeros((X.shape[0], len(self.classes_)))
    if X.shape[1] == 1:
        return d

    # Compute distances to signatures according to the metric
    for i in range(X.shape[0]):
        ranked_sample = rankdata(X[i], method=self.r_method)
        for j in range(len(self.classes_)):
            if self.metric == 'spearman':
                d[i][j] = np.sum(np.multiply(
                    np.abs(ranked_sample-self.class_signatures_[j]),
                    self.weights_[j]
                ))
            elif self.metric == 'kendall':
                d[i][j] = 1-kendalltau(ranked_sample, self.class_signatures_[j])[0]

    return d
```

Listato 5: Il metodo di RAC per calcolare la distanza tra le istanze e le class signature.

```
def predict_proba(self, X):
    # Check if fit had been called
    check_is_fitted(self, ['X_', 'y_'])

    # Input validation
    X = check_array(X, ensure_min_features=self.n_features_in_)

    p = np.ones((X.shape[0], len(self.classes_)))
    if len(self.classes_) == 1:
        return p

    d = self.distances_to_signatures(X)

    # Convert distances to probabilities
    for i in range(X.shape[0]):
        p[i] = self.convert_distances_to_probas(d[i])
        closers = np.where(d[i]==d[i].min())[0]
        if closers.size > 1: # Compute euclidean distances to centroids
            pl = 0
            p2l = 0
            for j in range(p.shape[1]):
                if p[i][j] > pl:
                    p2l = pl
                    pl = p[i][j]
            leftp = closers.size*(pl-p2l)
            dc = np.linalg.norm(X[i]-self.class_centroids_[closers], axis=1)
            pc = self.convert_distances_to_probas(dc)
            for j in range(closers.size): # Adjust probabilities
                p[i][closers[j]] = p2l + pc[j]*leftp

    return p

def convert_distances_to_probas(self, d):
    p = np.empty((d.shape[0]))
    sumd = np.sum(d)
    if sumd == 0:
        p.fill(1/d.shape[0])
        return p
    for i in range(d.shape[0]):
        p[i] = (1-(d[i]/sumd))/(d.shape[0]-1)
    return p
```

Listato 6: I metodi di RAC per calcolare le probabilità di appartenenza delle istanze alle classi.

```
from rac import RAClassifier
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

data = load_iris(return_X_y=True)
X, y = data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1/3)

clf = RAClassifier(metric='kendall')
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
prob_pred = clf.predict_proba(X_test)
```

Listato 7: Un esempio di utilizzo di RAC.

4.3 Applicazione in ambito biomedico

La classificazione di campioni biologici in base ai loro profili di espressione genica costituisce una delle più adeguate applicazioni di RAC. Infatti, la trasformazione in ranghi rende il classificatore più robusto al rumore nei dati e ciò può risultare particolarmente efficace nell'analisi di dati di trascrittomica, tipicamente rumorosi sia perché il processo di espressione genica è per sua natura stocastico [24], sia perché le tecnologie utilizzate per questo tipo di analisi possono condurre a misurazioni imprecise [62]. Inoltre, l'approccio rank-based, essendo invariante a trasformazioni monotòne, potrebbe favorire una più facile integrazione dei dati di esperimenti diversi, ottenuti con protocolli di laboratorio distinti o con tecniche di preprocessing differenti.

Occorre anche evidenziare che, per classificare i campioni, RAC confronta un vasto numero di sequenze geniche. Questo modo di operare si contrappone a quello dei metodi statistici convenzionali che mirano a individuare un piccolo sottoinsieme di geni con un alto potere discriminante e che nella pratica si sono, però, dimostrati fallaci in termini di affidabilità e robustezza [63]. L'approccio di RAC può, quindi, rivelarsi più efficace dal momento che patologie come il cancro sono fenomeni complessi che coinvolgono molti meccanismi regolatori e metabolici, per cui il confronto di un ampio numero di trascritti può fornire informazioni più adeguate per cogliere tale complessità.

4.3.1 Lavori correlati

L'utilizzo di metodi di rank aggregation in bioinformatica non è nuovo. Vari sono gli studi in letteratura che impiegano tali metodi per integrare liste di geni ordinate con criteri differenti o provenienti da esperimenti diversi [46, 64–66]. Questi approcci hanno però un'impostazione diversa da quella di RAC in quanto prevedono l'integrazione di liste di geni ordinate in base al grado di correlazione con la patologia studiata e non hanno come scopo la classificazione.

L'impiego di algoritmi rank-based per la classificazione di profili di espressione genica è già stato proposto in letteratura. I metodi di classificazione adottati negli studi consultati, però, differiscono da RAC sotto molteplici aspetti. Ad esempio,

Lauria [67] presenta un metodo che prevede la definizione di una signature per ciascun campione mediante la selezione degli n_1 geni più espressi e degli n_2 geni meno espressi (con n_1 e n_2 parametri configurabili) e, successivamente, la costruzione di una rete nella quale i nodi rappresentano i campioni e la lunghezza degli archi riflette il grado di similarità tra le signature dei campioni. Dalla rete così ottenuta tipicamente emerge che i pazienti con fenotipi simili costituiscono cluster, per cui è possibile classificare nuovi campioni individuando i cluster a cui appartengono. Tale metodo si è dimostrato molto efficace: nella competizione scientifica internazionale sbv IMPROVER [68], a cui hanno partecipato 54 team, il gruppo di lavoro che lo ha presentato si è posizionato al secondo posto.

Un altro approccio rank–based che prevede l'ordinamento dei geni in base al loro livello di espressione è quello adottato dal classificatore *TSP* (Top Scoring Pair), proposto da Geman et al [69]. Per compiti di classificazione binaria questo metodo individua la coppia di geni (i, j) tale che la differenza tra le probabilità dell'evento $\{R_i < R_j\}$ calcolata per le due classi è massima, dove con R_i e R_j si indica il rango dei geni i e j. Tale coppia di geni fornisce una semplice regola decisionale per classificare nuovi campioni. Tan et al. [70] hanno riproposto questo classificatore suggerendo di individuare le k coppie di geni con la proprietà sopra descritta (dove k è un parametro configurabile) e di impiegare tutte le regole decisionali da esse derivate. Le prestazioni di questo metodo sono risultate pressoché equivalenti a quelle di altri metodi di classificazione, ma le regole decisionali costruite da esso coinvolgono un ridotto numero di geni e sono facilmente interpretabili.

Capitolo 5

Risultati sperimentali

Questo capitolo illustra il processo di validazione di RAC e i risultati sperimentali dell'analisi comparativa condotta. Le prestazioni di RAC sono state state confrontate con quelle di altri 5 classificatori allo stato dell'arte: Nearest Centroid, K-Nearest Neighbors, Support Vector Machine, Gaussian Naive Bayes e Random Forest. RAC è stato valutato per la classificazione binaria e multiclasse su 12 dataset di espressione genica ricavati da esperimenti di microarray e su 3 dataset di immagini. La valutazione è stata effettuata impiegando le implementazioni dei classificatori e gli strumenti per il preprocessing e la validazione offerti dalla libreria Python scikit-learn. La macchina utilizzata per effettuare la valutazione dei modelli è dotata di una memoria RAM di 8 GB e di un processore Intel Core i5 dual-core con una frequenza di clock di 1.6 GHz.

5.1 Dataset di espressione genica

I dataset di espressione genica utilizzati per valutare il classificatore sono stati scaricati da Gene Expression Omnibus (GEO) [71], un vasto database di profili di espressione genica gestito dal Centro Nazionale per le Informazioni Biotecnologiche (NCBI) degli Stati Uniti. Tutti i dataset impiegati derivano da esperimenti realizzati con la tecnologia dei microarray. La tabella 5.1 descrive le specificità dei dataset, riportando per ciascuno di essi il nome, il codice di accesso nel database GEO

(GEO Accession), la descrizione dello studio condotto, le classi a cui appartengono i campioni e il numero di feature di ciascun campione.

Per meglio valutare il classificatore sono stati scelti dataset molto eterogenei, provenienti da studi di trascrittomica condotti con metodi statistici tradizionali e aventi finalità diverse, che differiscono anche per il numero di campioni esaminati (da un minimo di 3 campioni per classe a un massimo di 809) e per il numero di feature (il minimo numero di geni è 1 134 e il massimo è 54 675). Due dataset (ImmuneCells1 e ImmuneCells2) derivano da studi aventi lo scopo di individuare i profili di espressione genica caratteristici di differenti tipologie di cellule immunitarie; gli altri derivano da esperimenti condotti allo scopo di individuare i profili di espressione genica di pazienti affetti da patologie di vario tipo (cancro ai polmoni, al seno, alla vescica, alla prostata, al fegato e al pancreas, psoriasi e demenza). Dal dataset Dementia, che originariamente presentava un forte sbilanciamento delle classi, sono stati estratti randomicamente alcuni campioni appartenenti alle classi di interesse.

5.1.1 Procedura di valutazione

Come primo step del processo di valutazione è stata applicata una trasformazione logaritmica ai dataset che presentavano distribuzioni con asimmetria positiva.[1] I dati sono stati successivamente standardizzati[2] sottraendo a ciascuna feature il suo valore medio e dividendo la differenza ottenuta per la sua deviazione standard. La media e la deviazione standard sono state calcolate soltanto per le feature dei campioni nei fold di allenamento in modo da tenere separati i dati di test in ogni fase del processo di validazione. Sui dataset ImmuneCells1 e ImmuneCells2, dato il ridotto numero di campioni disponibili, è stata effettuata una double leave one out cross validation. Sugli altri dataset è stata, invece, effettuata una double 5-fold cross validation, suddividendo i dati nei fold in modo da preservare la percentuale di esempi di ciascuna classe. I valori degli iperparametri testati per ciascun modello sono mostrati

[1] Tale procedura consente di ridurre la variabilità dei dati e di rendere la distribuzione più simile alla distribuzione normale [83].

[2] La standardizzazione dei dati è una pratica tipicamente utilizzata per migliorare le prestazioni degli algoritmi distance-based come NC, KNN e SVM [9].

Tabella 5.1: Specificità dei dataset di espressione genica.

Nome	**GEO Accession**	**Descrizione**	**Classi (#istanze)**	**#Feature**
ImmuneCells1	GSE28489	miRNA expression profiling of human immune cell subsets (HUG) [72]	B-cells (5), Eosinophils (3)	7 815
ImmuneCells2	GE28487	miRNA expression profiling of human immune cell subsets (Roche) [72]	B-cells (5), Eosinophils (4)	7 815
LungCancer1	GSE19804	Genome-wide screening of transcriptional modulation in non-smoking female lung cancer in Taiwan [73]	Normal (60), Lung Cancer (60)	54 675
LungCancer2	GSE43346	Gene repression with H3K27me3 modification in human small cell lung cancer [74]	Normal (42), SCLC[1] (23)	54 675
BreastCancer1	GSE27562	Expression data from human PBMCs from breast cancer patients and controls [75]	Normal (31), Malignant (51), Benign (37), Ectopic (22)	54 675
BreastCancer2	GSE22981	Circulating miRNAs as biomarkers and potential functional mediators of early stage breast cancer [76]	Control (20), Case (20)	1 134
BreastCancer3	GSE41526	CWRU Circulating miRNAs in Breast Cancer [77]	Control (20), Pre-resection (20), Post-resection (20)	1 145
BladderCancer	GSE113486	Circulating miRNA panels for specific and early detection in bladder cancer [78]	Non-cancer (100), Bladder Cancer (392)	2 565
ProstateCancer	GSE112264	Large-scale and high-confidence serum circulating miRNA biomarker discovery in prostate cancer [79]	Non-cancer (41), NPBx[2] (241), PCa[3] (809), HCC[4] (50), PC[5] (50)	2 565
Psoriasis1	GSE14905	Type I Interferon: Potential Therapeutic Target for Psoriasis? [80]	Normal (21), Psoriasis (61)	54 675
Psoriasis2	GSE13355	Gene expression data of skin from psoriatic patients and normal controls [81]	Normal (64), Psoriasis (58)	54 675
Dementia	GSE120584	Risk prediction models for dementia constructed by supervised principal component analysis using miRNA expression data [82]	AD[6] (91), VaD[7] (91), DLB[8] (91)	2 562

[1] Small Cell Lung Cancer
[2] Negative Prostate Biopsy
[3] Prostate Cancer
[4] Hepatocellular Carcinoma
[5] Pancreatic Cancer
[6] Alzheimer's Disease
[7] Vascular Dementia
[8] Dementia with Lewy Bodies

Tabella 5.2: Valori degli iperparametri dei classificatori valutati. Nella tabella viene utilizzata la notazione adottata da scikit-learn. Degli altri parametri sono utilizzati i valori di default.

Classificatore	**Iperparametro**	**Valori**
RAC	`r_method`	`'average'`, `'min'`, `'max'`
	`ra_method`	`'borda'`, `'borda_median'`, `'borda_gmean'`, `'borda_l2'`
	`metric`	`'spearman'`, `'kendall'`
	`weighted`	`True`, `False`, `(1/3*n_feature, 1/3*n_feature)`[1]
	`p`	1, 2, 3/4
NC	`metric`	`'euclidean'`, `'manhattan'`
KNN	`weights`	`'uniform'`, `'distance'`
	`n_neighbors`	`range(3, max_neighbors+1, 2)`[2]
SVM	`C`	0.001, 0.1, 1, 100
	`kernel`	`'linear'`, `'poly'`, `'rbf'`, `'sigmoid'`
	`degree`	2, 3, 4
	`gamma`	`'scale'`, `'auto'`, 0.001, 0.1, 1, 100
GNB	`var_smoothing`	1e-9, 1e-3, 1e-1, 0.5
RF	`n_estimators`	50, 100, 500
	`min_samples_split`	0.1, 0.25, 0.33, 2

[1] `n_feature` indica il numero di feature del dataset

[2] `max_neighbors` è pari alla dimensione del training fold divisa per il numero di classi se questa quantità è minore di 20, altrimenti è pari a 20

nella tabella (5.2). La selezione degli iperparametri è stata effettuata valutando lo score f_1 weighted calcolato sulla concatenazione delle predizioni effettuate sui test fold interni e scegliendo, quindi, gli iperparametri con cui i modelli raggiungevano i valori più alti dello score. Dopo aver allenato nuovamente i classificatori configurati con gli iperparametri selezionati sui training fold esterni, sono stati valutati gli score f_1 weighted e AUROC, calcolati sulla concatenazione delle predizioni effettuate sui test fold esterni. Lo score f_1 weighted è stato impiegato anche nei task di classificazione binaria al fine di poter meglio valutare le prestazioni dei classificatori in presenza di classi sbilanciate.

Dal momento che l'implementazione di scikit-learn di NC non supporta il calcolo della probabilità di appartenenza alle classi, tale stima, necessaria per la valutazione dello score AUROC, è stata ricavata facendo uso della stessa procedura implementata per RAC e descritta nella sezione 4.1.5.

5.1.2 Analisi dei risultati

Misure di prestazione

I risultati della valutazione, riportati nelle tabelle 5.3–5.6, mostrano che le performance di RAC sono molto simili a quelle degli altri classificatori. In particolare, confrontando le medie degli score f_1 weighted e AUROC, RAC riporta rispettivamente il quarto e il secondo risultato migliore nei task di classificazione binaria e il quinto e il quarto risultato migliore nei task di classificazione multiclasse. Tali classifiche non sono però particolarmente significative dal momento che le differenze tra le medie degli score dei classificatori sono relativamente basse.

Sul dataset Dementia (AD vs VaD) RAC ha raggiunto il valore più alto dello score f_1 weighted e sul dataset BreastCancer3 (Control vs Post-resection) RAC ha raggiunto il valore più alto dello score AUROC. Sul dataset BreastCancer1 (Ectopic vs Malignant), BladderCancer, ProstateCancer (Non-cancer vs HCC) e Psoriasis1 RAC ha ottenuto il valore più basso dello score f_1 weighted, mentre sul dataset BreastCancer1 (Benign vs Malignant) RAC ha ottenuto il valore più basso dello score AUROC.

Non sono emerse caratteristiche dei dataset (in termini di numero di campioni e numero di feature) che influiscono in modo sistematico sulle performance di RAC.

Tabella 5.3: Performance dei classificatori su dataset di espressione genica binari valutate con la misura f_1 weighted. Il valore migliore e il valore peggiore della misura per ciascun dataset sono evidenziati rispettivamente in grassetto e in corsivo.

Dataset (#feature)	**Classi (#istanze)**	**RAC**	**NC**	**KNN**	**SVM**	**GNB**	**RF**
ImmuneCells1 (7 815)	B-cells (5), Eosinophils (3)	**1.000**	**1.000**	**1.000**	**1.000**	*0.481*	0.708
ImmuneCells2 (7 815)	B-cells (5), Eosinophils (4)	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
LungCancer1 (54 675)	Normal (60), Lung Cancer (60)	0.917	0.908	*0.786*	**0.958**	0.925	**0.958**
LungCancer2 (54 675)	Normal (42), SCLC (23)	**1.000**	*0.985*	**1.000**	**1.000**	*0.985*	**1.000**
BreastCancer1 (54 675)	Normal (31), Malignant (51)	0.904	0.940	0.926	*0.902*	0.903	**0.963**
BreastCancer1 (54 675)	Benign (37), Malignant (51)	0.419	0.424	0.521	*0.407*	0.435	**0.522**
BreastCancer1 (54 675)	Ectopic (22), Malignant (51)	*0.841*	0.893	**0.973**	**0.973**	0.906	0.905
BreastCancer2 (1 134)	Control (20), Case (20)	0.472	0.499	0.495	**0.549**	*0.458*	0.475
BreastCancer3 (1 145)	Control (20), Pre-resection (20)	0.763	*0.605*	0.646	0.665	**0.768**	0.750
BreastCancer3 (1 145)	Control (20), Post-resection (20)	0.499	0.522	*0.449*	0.475	0.520	**0.575**
BladderCancer (2 565)	Non-cancer (100), Bladder Cancer (392)	*0.960*	0.992	0.998	**1.000**	0.988	0.992
ProstateCancer (2 565)	NPBx (241), PCa (809)	0.893	*0.865*	0.894	**1.000**	0.880	0.998
ProstateCancer (2 565)	Non-cancer (41), HCC (50)	*0.956*	*0.956*	**0.989**	**0.989**	*0.956*	0.967
ProstateCancer (2 565)	Non-cancer (41), PC (50)	0.978	*0.934*	0.978	**0.989**	0.967	0.967
Psoriasis1 (54 675)	Normal (21), Psoriasis (61)	*0.805*	0.881	0.924	0.951	0.889	**0.962**
Psoriasis2 (54 675)	Normal (64), Psoriasis (58)	**1.000**	*0.992*	*0.992*	**1.000**	**1.000**	**1.000**
Dementia (2 562)	AD (91), DLB (91)	0.553	0.500	*0.472*	0.497	0.514	**0.574**
Dementia (2 562)	AD (91), VaD (91)	**0.555**	0.520	0.478	*0.461*	0.509	0.516
Dementia (2 562)	DLB (91), VaD (91)	0.458	**0.505**	0.499	*0.445*	0.503	0.450
Media		0.788	0.785	0.791	0.803	*0.768*	**0.804**
Rank		4	5	3	2	6	1

Tabella 5.4: Performance dei classificatori su dataset di espressione genica binari valutate con la misura AUROC. Il valore migliore e il valore peggiore della misura per ciascun dataset sono evidenziati rispettivamente in grassetto e in corsivo.

Dataset (#feature)	**Classi (#istanze)**	**RAC**	**NC**	**KNN**	**SVM**	**GNB**	**RF**
ImmuneCells1 (7 815)	B-cells (5), Eosinophils (3)	**1.000**	**1.000**	**1.000**	**1.000**	*0.500*	**1.000**
ImmuneCells2 (7 815)	B-cells (5), Eosinophils (4)	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
LungCancer1 (54 675)	Normal (60), Lung Cancer (60)	0.964	0.969	*0.897*	0.980	0.924	**0.984**
LungCancer2 (54 675)	Normal (42), SCLC (23)	**1.000**	**1.000**	**1.000**	**1.000**	*0.978*	**1.000**
BreastCancer1 (54 675)	Normal (31), Malignant (51)	0.988	**0.994**	0.966	0.954	*0.908*	0.985
BreastCancer1 (54 675)	Benign (37), Malignant (51)	*0.404*	0.414	**0.485**	0.431	0.427	0.480
BreastCancer1 (54 675)	Ectopic (22), Malignant (51)	0.955	0.961	0.967	**0.993**	*0.906*	0.977
BreastCancer2 (1 134)	Control (20), Case (20)	0.413	**0.480**	0.423	*0.368*	0.468	0.468
BreastCancer3 (1 145)	Control (20), Pre-resection (20)	0.760	0.733	*0.710*	0.740	0.773	**0.853**
BreastCancer3 (1 145)	Control (20), Post-resection (20)	**0.575**	0.548	*0.490*	0.521	0.555	0.540
BladderCancer (2 565)	Non-cancer (100), Bladder Cancer (392)	0.995	0.999	**1.000**	**1.000**	*0.989*	**1.000**
ProstateCancer (2 565)	NPBx (241), PCa (809)	0.955	0.939	0.939	**1.000**	*0.868*	**1.000**
ProstateCancer (2 565)	Non-cancer (41), HCC (50)	0.984	0.994	**1.000**	**1.000**	*0.962*	0.999
ProstateCancer (2 565)	Non-cancer (41), PC (50)	0.987	0.986	0.988	**1.000**	*0.962*	0.999
Psoriasis1 (54 675)	Normal (21), Psoriasis (61)	0.952	0.961	0.945	0.978	*0.848*	**0.989**
Psoriasis2 (54 675)	Normal (64), Psoriasis (58)	**1.000**	**1.000**	*0.991*	**1.000**	**1.000**	**1.000**
Dementia (2 562)	AD (91), DLB (91)	0.545	0.503	0.498	*0.480*	0.505	**0.607**
Dementia (2 562)	AD (91), VaD (91)	0.555	0.531	*0.489*	0.521	0.520	**0.558**
Dementia (2 562)	DLB (91), VaD (91)	0.454	0.462	0.489	*0.436*	0.469	**0.498**
Media		0.815	0.814	0.804	0.811	*0.766*	**0.839**
Rank		2	3	5	4	6	1

Tabella 5.5: Performance dei classificatori su dataset di espressione genica multiclasse valutate con la misura f_1 weighted. Il valore migliore e il valore peggiore della misura per ciascun dataset sono evidenziati rispettivamente in grassetto e in corsivo.

Dataset (#feature)	**Classi (#istanze)**	**RAC**	**NC**	**KNN**	**SVM**	**GNB**	**RF**
BreastCancer1 (54 675)	Normal (31), Benign (37), Ectopic (22), Malignant (51)	*0.518*	0.575	**0.642**	0.632	0.528	0.608
BreastCancer3 (1 145)	Control (20), Pre-resection (20) Post-resection (20)	0.474	0.406	*0.404*	**0.514**	0.405	0.466
Dementia (2 562)	AD (91), VaD (91), DLB (91)	*0.325*	0.353	0.359	**0.363**	0.329	0.329
Media		0.439	0.445	0.468	**0.503**	*0.421*	0.468
Rank		5	4	2	1	6	2

Tabella 5.6: Performance dei classificatori su dataset di espressione genica multiclasse valutate con la misura AUROC. Il valore migliore e il valore peggiore della misura per ciascun dataset sono evidenziati rispettivamente in grassetto e in corsivo.

Dataset (#feature)	**Classi (#istanze)**	**RAC**	**NC**	**KNN**	**SVM**	**GNB**	**RF**
BreastCancer1 (54 675)	Normal (31), Benign (37), Ectopic (22), Malignant (51)	0.794	0.780	0.834	**0.851**	*0.698*	0.833
BreastCancer3 (1 145)	Control (20), Pre-resection (20), Post-resection (20)	0.591	0.570	*0.569*	0.587	0.571	**0.625**
Dementia (2 562)	AD (91), VaD (91), DLB (91)	0.494	*0.477*	0.518	**0.522**	0.499	0.502
Media		0.627	0.609	0.640	**0.653**	*0.590*	**0.653**
Rank		4	5	3	1	6	1

Curve di apprendimento

Per valutare l'andamento delle prestazioni dei classificatori al variare delle dimensioni dei training set sono state realizzate le curve di apprendimento. I classificatori con i migliori iperparametri precedentemente individuati sono stati quindi allenati su training set di dimensioni diverse, calcolando il valore dello score f_1 weighted raggiunto nel test. In particolare, per ogni dimensione del training set si è ripetuto l'allenamento e il test dei modelli 5 volte, utilizzando test fold disgiunti e riportando infine la media aritmetica degli score ottenuti. Si osservi che lo scopo di questa valutazione è quello di confrontare le prestazioni dei classificatori al variare della dimensione del training set e non quello di stimare l'errore di generalizzazione dei modelli. Pertanto, anche se i migliori iperparametri sono stati selezionati allenando i modelli sugli stessi dati, tale pratica in questo contesto non è da considerarsi errata.

Di seguito sono riportate le curve di apprendimento relative ai dataset su cui i classificatori hanno raggiunto buone prestazioni nella double k-fold cross validation (figure 5.1–5.10). Per i dataset ImmuneCells1 e ImmuneCells2 le curve di apprendimento non sono state realizzate data la scarsa numerosità dei campioni. Si può osservare che l'andamento delle curve di apprendimento di RAC è simile a quello delle curve degli altri classificatori. Per alcuni dataset RAC sembra ottenere buoni risultati di classificazione anche in presenza di pochi esempi (figure 5.3, 5.8, 5.10). In alcuni casi la curva di RAC giace al di sotto di quella degli altri classificatori (figure 5.4, 5.5, 5.9).

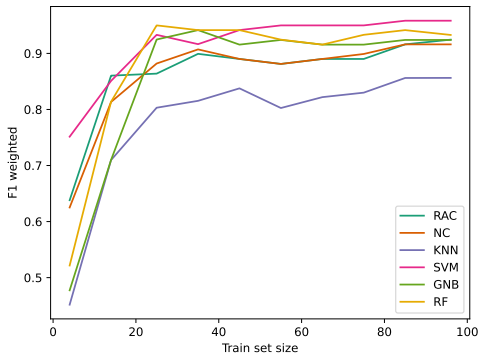

Figura 5.1: Curve di apprendimento dei classificatori relative al dataset LungCancer1.

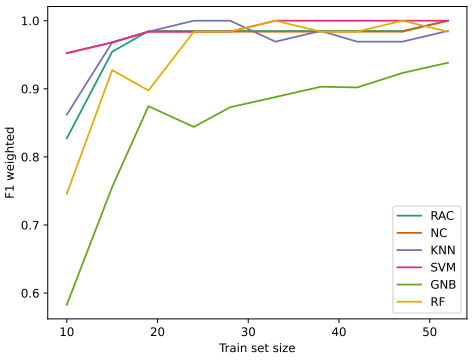

Figura 5.2: Curve di apprendimento dei classificatori relative al dataset LungCancer2.

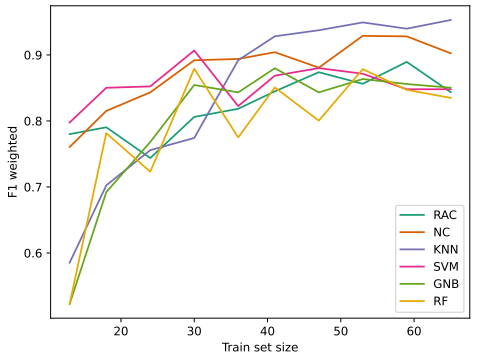

Figura 5.3: Curve di apprendimento dei classificatori relative al dataset BreastCancer1 (Normal vs Malignant).

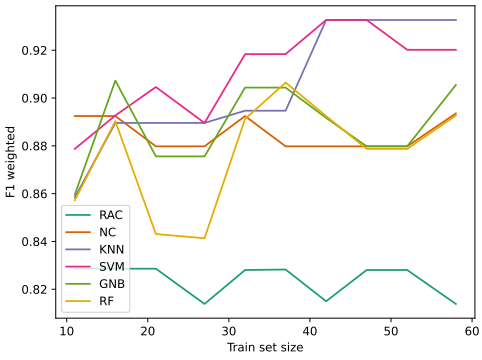

Figura 5.4: Curve di apprendimento dei classificatori relative al dataset BreastCancer1 (Ectopic vs Malignant).

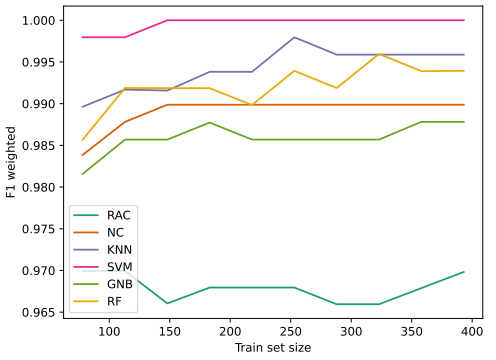

Figura 5.5: Curve di apprendimento dei classificatori relative al dataset BladderCancer.

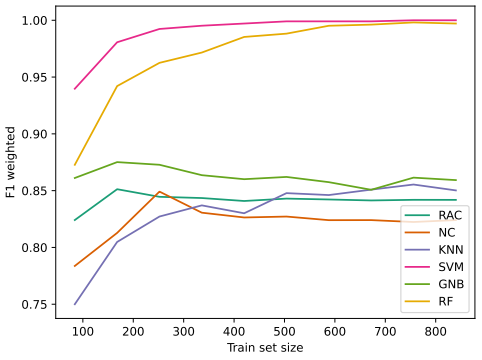

Figura 5.6: Curve di apprendimento dei classificatori relative al dataset ProstateCancer (NPBx vs PCa).

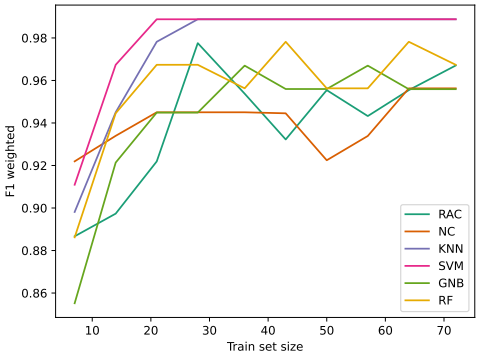

Figura 5.7: Curve di apprendimento dei classificatori relative al dataset ProstateCancer (Non-cancer vs HCC).

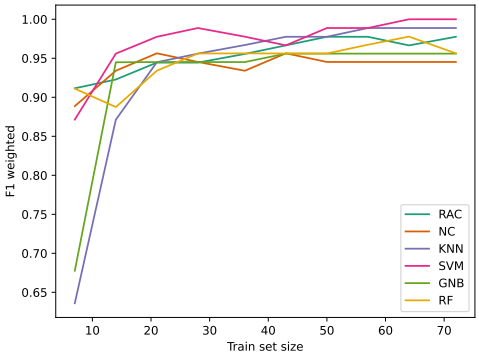

Figura 5.8: Curve di apprendimento dei classificatori relative al dataset ProstateCancer (Non-cancer vs PC).

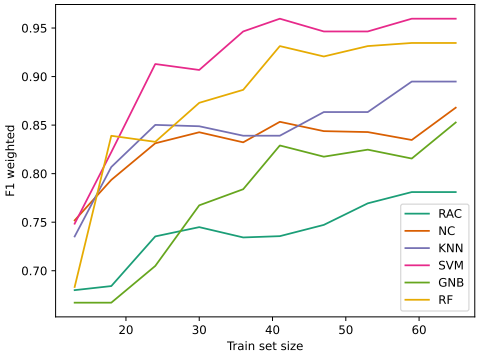

Figura 5.9: Curve di apprendimento dei classificatori relative al dataset Psoriasis1.

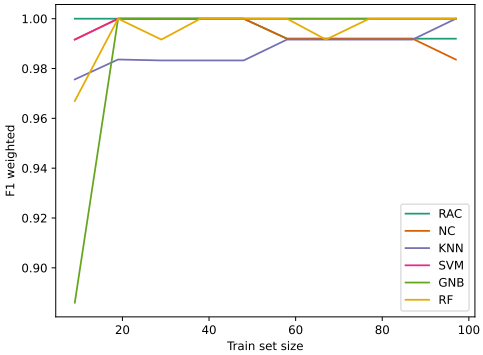

Figura 5.10: Curve di apprendimento dei classificatori relative al dataset Psoriasis2.

Tempi di computazione

Per confrontare i tempi di computazione dei classificatori sono stati realizzati diagrammi a scatola con i tempi di CPU in secondi richiesti per l'allenamento e la predizione nei fold delle cross validation più interne (figure 5.11–5.26). Dal momento che il calcolo della distanza di Kendall richiede tempi di computazione maggiori di quelli necessari per il calcolo della distanza di Spearman, i tempi di predizione di RAC sono stati analizzati separatamente in base alla misura di distanza impiegata.

Generalmente i tempi di allenamento di RAC sono minori o confrontabili con quelli di RF, talvolta sono simili anche a quelli di SVM. Rispetto ai tempi di allenamento degli altri classificatori, però, quelli richiesti da RAC sono maggiori. I tempi di predizione di RAC configurato con la distanza di Spearman dipendono dalle caratteristiche del dataset: sono talvolta maggiori di quelli degli altri classificatori e talvolta paragonabili ai tempi richiesti da alcuni di essi. I tempi di predizione di RAC configurato con la distanza di Kendall sono generalmente maggiori di quelli di tutti gli altri classificatori.

Con un numero di campioni molto basso sia i tempi di allenamento che quelli di predizione di RAC sono nettamente inferiori a quelli di RF e si avvicinano di più a quelli degli altri classificatori (figure 5.11 e 5.12).

Dal confronto delle figure 5.17 e 5.18 con le figure 5.23 e 5.24 è possibile valutare come variano i tempi di computazione dei classificatori al crescere del numero di feature. Infatti, esse si riferiscono a dataset con un numero di campioni pressoché uguale (88 e 91) ma con un numero di feature molto diverso (54 675 e 2 565). Come si osserva, al crescere del numero di feature i tempi di allenamento di RAC tendono ad avvicinarsi a quelli di RF (rispetto ai quali precedentemente erano inferiori) e i tempi di predizione, che prima erano simili a quelli di SVM e RF, diventano più alti di quelli di tutti gli altri classificatori.

Quando il numero di feature è alto, indipendentemente dal numero di campioni, i tempi di allenamento di RAC sono paragonabili a quelli di RF e quelli di predizione sono maggiori di quelli degli altri classificatori (si vedano le figure 5.13–5.18, tutte relative a dataset con 54 675 feature).

Quando, invece, il numero di feature è più basso, come per i dataset a cui si

riferiscono le figure 5.19–5.26, risulta che i tempi di allenamento di RAC sono sempre inferiori a quelli di RF e con un numero più alto di campioni sono paragonabili a quelli di SVM. In questo caso i tempi di predizione di RAC configurato con la distanza di Spearman sono sempre confrontabili con quelli di KNN e SVM, mentre i tempi di predizione di RAC configurato con la distanza di Kendall sono maggiori di quelli di tutti gli altri classificatori.

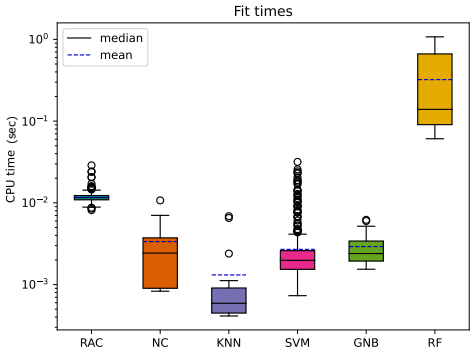

Figura 5.11: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset ImmuneCells1.

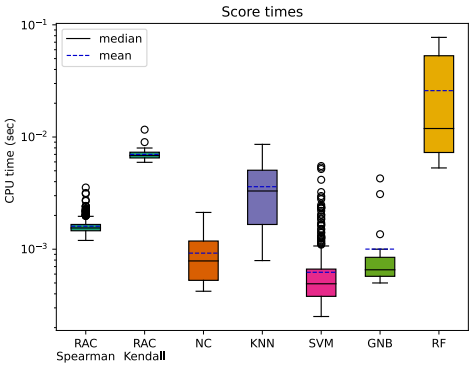

Figura 5.12: Diagramma a scatola dei tempi di predizione dei classificatori relativo al dataset ImmuneCells1.

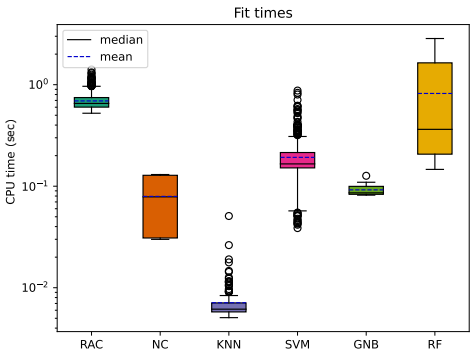

Figura 5.13: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset LungCancer1.

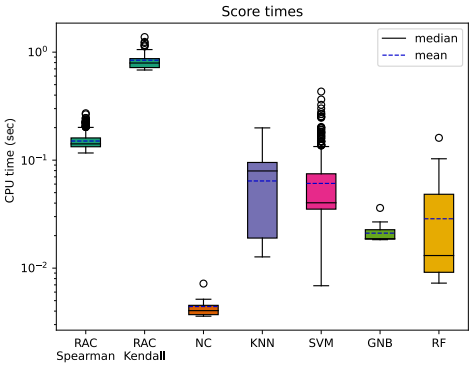

Figura 5.14: Diagramma a scatola dei tempo di predizione dei classificatori relativo al dataset LungCancer1.

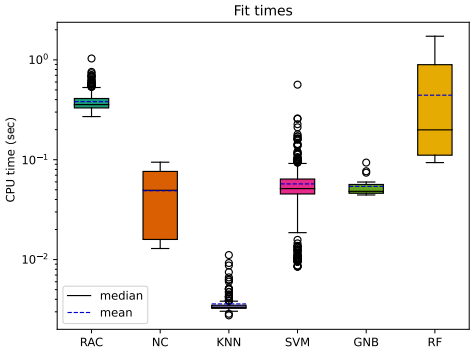

Figura 5.15: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset LungCancer2.

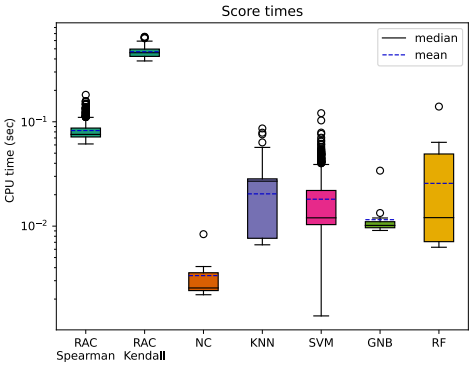

Figura 5.16: Diagramma a scatola dei tempo di predizione dei classificatori relativo al dataset LungCancer2.

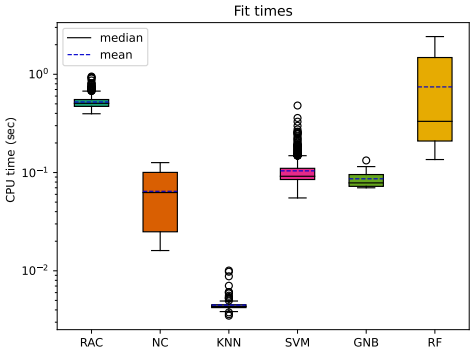

Figura 5.17: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset BreastCancer1 (Benign vs Malignant).

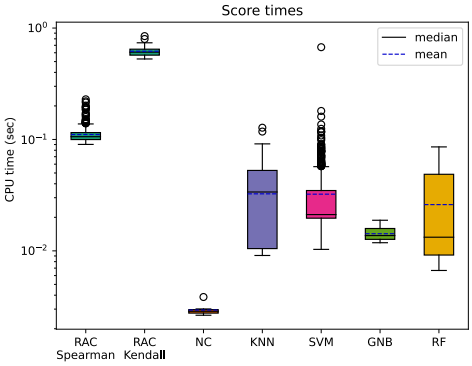

Figura 5.18: Diagramma a scatola dei tempi di predizione dei classificatori relativo al dataset BreastCancer1 (Benign vs Malignant).

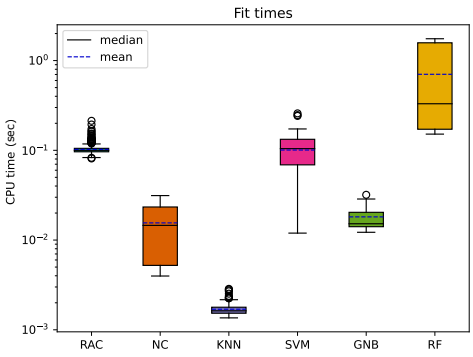

Figura 5.19: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset BladderCancer.

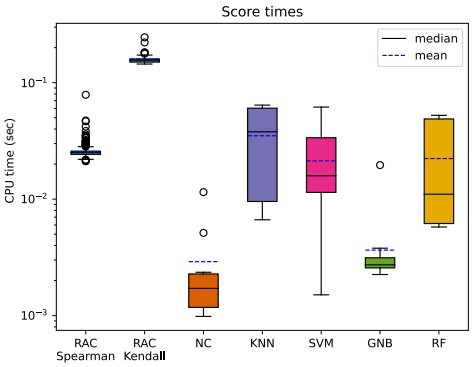

Figura 5.20: Diagramma a scatola dei tempi di predizione dei classificatori relativo al dataset BladderCancer.

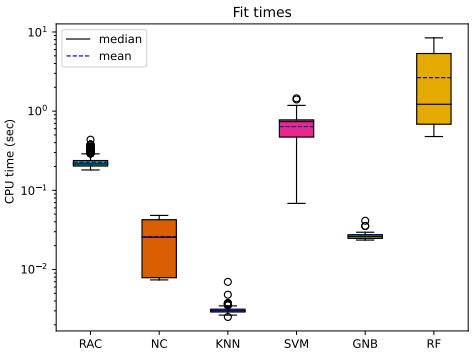

Figura 5.21: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset ProstateCancer (NPBx vs PCa).

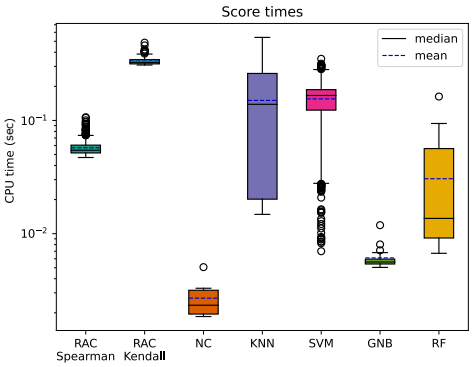

Figura 5.22: Digramma a scatola dei tempi di predizione dei classificatori relativo al dataset ProstateCancer (NPBx vs PCa).

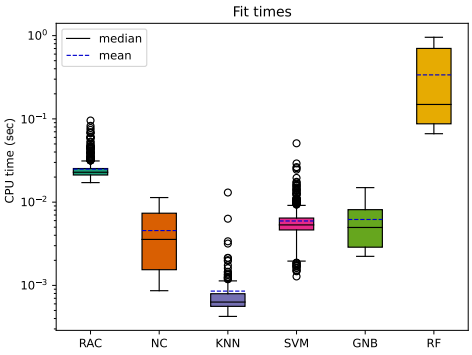

Figura 5.23: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset ProstateCancer (Non-cancer vs HCC).

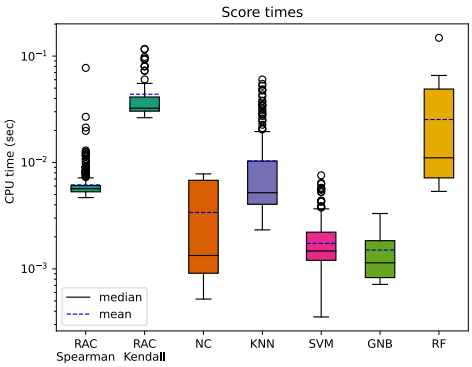

Figura 5.24: Digramma a scatola dei tempi di predizione dei classificatori relativo al dataset ProstateCancer (Non-cancer vs HCC).

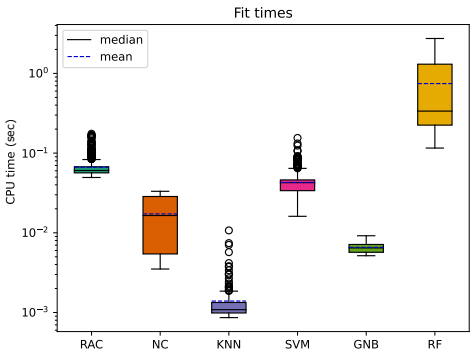

Figura 5.25: Diagramma a scatola dei tempi di allenamento dei classificatori relativo al dataset Dementia (AD vs DLB).

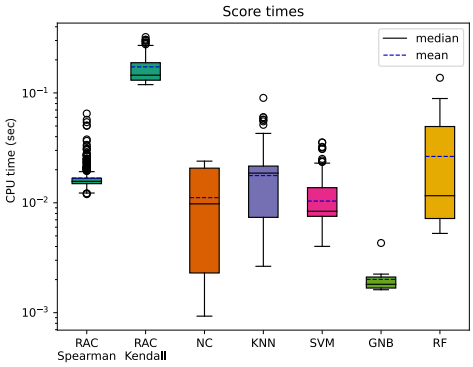

Figura 5.26: Digramma a scatola dei tempi di predizione dei classificatori relativo al dataset Dementia (AD vs DLB).

Influenza degli iperparametri

Per indagare l'influenza degli iperparametri di RAC sulle performance del classificatore sono state realizzate le heatmap dello score f_1 weighted. In particolare, è stata utilizzata la media dei k valori di tale score calcolato sulla concatenazione dei test fold delle cross validation più interne nella double k-fold cross validation. Per ciascun dataset si è scelto di realizzare 4 heatmap, in modo da valutare tutte le combinazioni significative degli iperparametri. Nella figura 5.27 sono illustrate, a titolo d'esempio, le heatmap relative al dataset ProstateCancer (Non-cancer vs PCa).

Per avere una visione d'insieme, le miniature di tutte le heatmap prodotte sono state raccolte nella figura 5.28. Si può osservare che in alcuni dataset i valori degli iperparametri non sembrano influire in modo significativo sulle prestazioni; in altri invece sembrano avere maggior influenza come, ad esempio, nel dataset BladderCancer, in cui tra il migliore e il peggior valore di f_1 weighted si ha una perdita di efficienza del 17.6%. Ad eccezione del caso in cui `metric='spearman'` e `weighted=True`, nelle heatmap emergono strisce orizzontali di colore che dimostrano che l'iperparametro che influisce maggiormente sulle prestazioni è il metodo di rank aggregation. Quando invece `metric='spearman'` e `weighted=True` le performance variano anche a seconda del valore di `p`. Sui dataset analizzati risulta, inoltre, che il

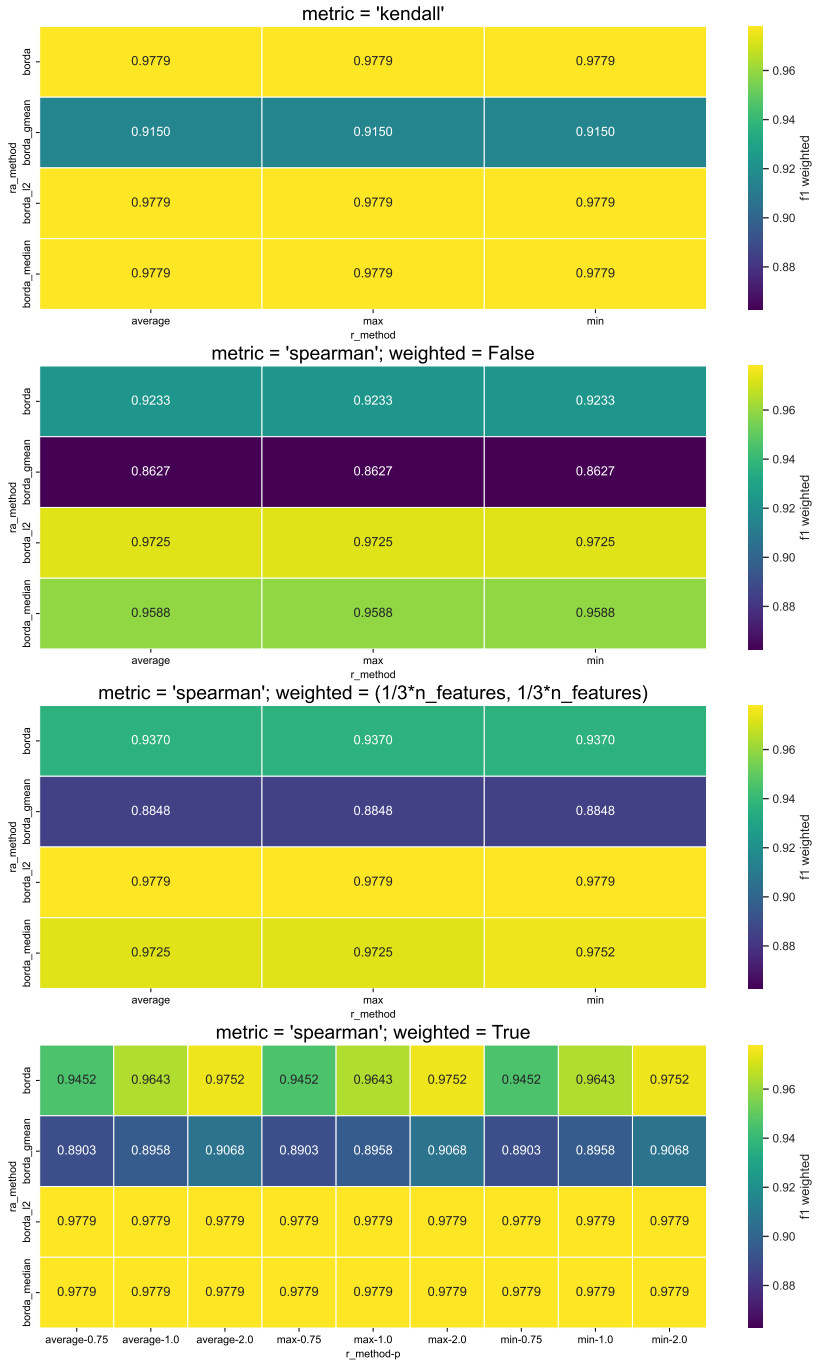

Figura 5.27: Heatmap dello score f_1 weighted di RAC con diverse combinazioni di iperparametri sul dataset ProstateCancer (Non-cancer vs PCa).

valore del parametro `r_method` non influisce mai sulle prestazioni del classificatore.Dal momento che non sembra esserci un metodo di rank aggregation, un valore di `p` o una misura di distanza che consentano di ottenere performance più alte su tutti i dataset, la ricerca dei valori migliori di questi iperparametri può rivelarsi utile al fine di ottimizzare le performance del classificatore.

5.2 Dataset di immagini

É stato possibile valutare RAC anche su dataset di immagini in quanto per loro natura presentano feature uniformi (i colori dei pixel).

A questo scopo sono stati selezionati tre dataset (tabella 5.7):

- Il dataset *MNIST* (Modified National Institute of Standards and Technology database) colleziona immagini in bianco e nero di cifre scritte a mano [84]. Prende questo nome in quanto raccoglie immagini provenienti da due dataset del NIST: un dataset di cifre scritte a mano da studenti delle scuole superiori (NIST's Special Database 1) e un dataset di cifre scritte da impiegati di un ufficio di censimento (NIST's Special Database 2). Fu utilizzato per la prima volta da Lecun et al. [85] nel 1998 come benchmark per valutare le performance di reti neurali convoluzionali. Il dataset presenta immagini di cifre scritte da 500 persone diverse ed è stato suddiviso in un insieme di training e uno di test in modo che cifre scritte dalla stessa persona non siano presenti in entrambi. Per la valutazione del classificatore sono state selezionate randomicamente 6 000 immagini dall'insieme di allenamento e 1 000 immagini da quello di test in modo da mantenere bilanciato il numero di esempi per classe.

- Il dataset *Olivetti faces* raccoglie fotografie in bianco e nero scattate in varie condizioni di illuminazione e con angolazioni differenti ai volti di 40 soggetti (10 immagini per ciascuno) che mostrano diverse espressioni facciali. Le fotografie sono state scattate a impiegati e studenti universitari tra il 1992 e il 1994 presso i laboratori di ricerca Olivetti di Cambridge (AT&T Laboratories Cambridge). É stata utilizzata la versione del dataset che può essere scaricata tramite la API

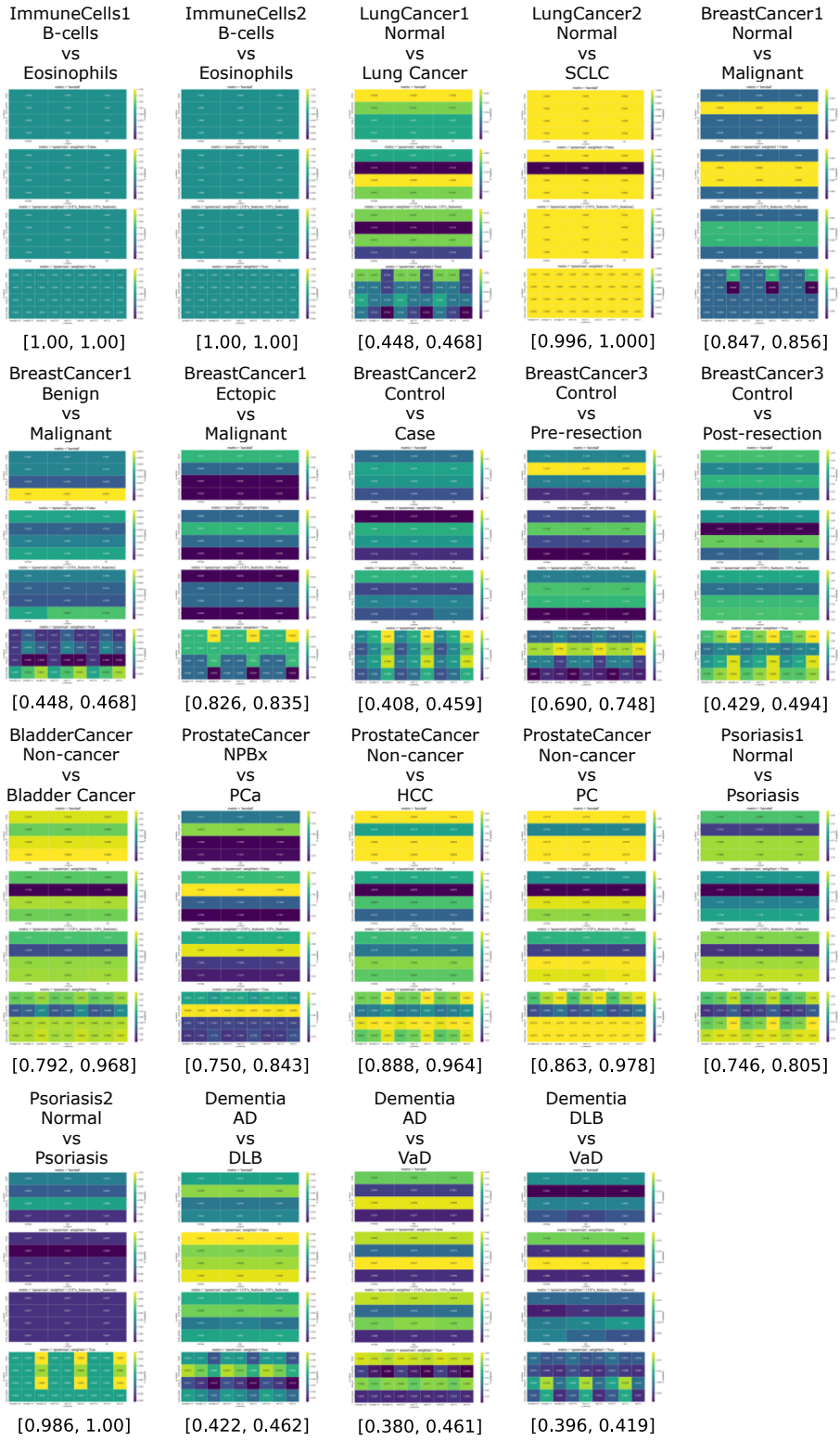

Figura 5.28: Miniature delle heatmap dello score f_1 weighted di RAC sui dataset di espressione genica binari. Sotto ciascuna miniatura è specificato l'intervallo dei valori assunti dallo score nella heatmap.

di scikit-learn [86] impiegando 7 immagini di ciascun soggetto per l'allenamento e 3 immagini di ciascun soggetto per il testing.

- Il dataset *CIFAR-10* (Canadian Institute For Advanced Research) colleziona immagini a colori in formato RGB di 10 soggetti (aeroplani, automobili, uccelli, gatti, cervi, cani, rane, cavalli, navi e camion) [87]. Le immagini sono state raccolte dal web dai ricercatori del MIT (Massachusetts Institute of Technology) e dell'NYU (New York University) [88]. Il dataset è suddiviso in 5 insiemi di allenamento e uno di test. Per la valutazione sono state selezionate randomicamente 100 immagini di ciascun soggetto da uno degli insiemi di allenamento e 25 immagini dall'insieme di test, tali immagini sono state utilizzate rispettivamente per allenare e testare i modelli.

Tabella 5.7: Specificità dei dataset di immagini utilizzati.

Nome	**Descrizione**	**#Immagini di train**	**#Immagini di test**	**#Classi**	**#Feature**
MNIST	Immagini in bianco e nero di cifre scritte a mano	6 000	1 000	10	784 (28×28)
Olivetti faces	Immagini in bianco e nero di volti	280	120	40	4 096 (64×64)
CIFAR-10	Immagini a colori di 10 soggetti (RGB)	1 000	250	10	3 072 ($3 \times 32 \times 32$)

5.2.1 Procedura di valutazione

Per valutare le performance dei classificatori, dopo aver standardizzato i dati, è stata effettuata una 5-fold cross validation sul training set e, una volta selezionati i migliori iperparametri[3] in base al valore dello score f_1 weighted, sono stati testati i classificatori sul test set, valutando i valori degli score f_1 weighted e AUROC. Dal momento che i tempi di computazione per l'allenamento e per la predizione di SVM risultavano troppo lunghi a causa del mancato supporto della classificazione

[3]Sono stati valutati gli stessi valori degli iperparametri dei classificatori utilizzati con i dataset di espressione genica.

multiclasse e della necessità di adottare una delle due tecniche one-versus-one o one-versus-rest, si è deciso di non impiegare questo classificatore per il confronto.

5.2.2 Analisi dei risultati

Dai risultati ottenuti (tabelle 5.8 e 5.9) è emerso che le prestazioni di RAC sono simili a quelle degli altri classificatori. Confrontando le medie degli score f_1 weighted e AUROC raggiunti dai classificatori su ciascun dataset, risulta che RAC consegue, rispettivamente, il terzo e il secondo valore migliore.

Tabella 5.8: Performance dei classificatori su dataset di immagini valutate con la misura f_1 weighted. Il valore migliore e il valore peggiore della misura per ciascun dataset sono evidenziati rispettivamente in grassetto e in corsivo.

Dataset	**RAC**	**NC**	**KNN**	**GNB**	**RF**
MNIST	0.756	0.769	0.898	*0.691*	**0.929**
Olivetti faces	0.832	*0.800*	0.891	**0.900**	0.898
CIFAR-10	0.256	0.261	*0.213*	0.239	**0.296**
Media	0.615	*0.610*	0.667	*0.610*	**0.708**
Rank	3	4	2	4	1

Tabella 5.9: Performance dei classificatori su dataset di immagini valutate con la misura AUROC. Il valore migliore e il valore peggiore della misura per ciascun dataset sono evidenziati rispettivamente in grassetto e in corsivo.

Dataset	**RAC**	**NC**	**KNN**	**GNB**	**RF**
MNIST	0.958	0.949	0.971	*0.945*	**0.997**
Olivetti faces	0.978	0.975	*0.944*	0.965	**0.996**
CIFAR-10	0.718	0.702	0.703	*0.700*	**0.788**
Media	0.885	0.875	0.873	*0.870*	**0.927**
Rank	2	3	4	5	1

Per valutare l'influenza degli iperparametri sulle prestazioni di RAC sono state realizzate le heatmap degli score f_1 weighted. Nella figura 5.29 viene mostrata la heatmap relativa al dataset MNIST. Da essa emerge che oltre al metodo di rank aggregation anche il metodo di assegnazione dei ranghi sembra incidere sulle prestazioni del classificatore, a differenza di quanto evidenziato nei dataset di espressione genica. In questo caso, dal momento che le feature hanno valori discreti è più probabile che

ci siano feature che assumono lo stesso valore, per cui il metodo di assegnazione dei ranghi che opera su tali feature potrebbe influire sui risultati della classificazione.

Si può osservare, inoltre, che la distanza di Spearman pesata, nonostante sia stata introdotta per essere applicata a dataset di espressione genica, anche su questo dataset ha un impatto positivo sulla classificazione. Ciò è dovuto al fatto che essa consente di attribuire un peso maggiore ai pixel più chiari e più scuri nelle signature delle classi, dunque ai pixel più rilevanti per l'identificazione delle cifre. Le parti più significative delle signature sono, infatti, costituite da pixel molto chiari o molto scuri, solo lo sfondo è di una tonalità di grigio intermedia, come è possibile osservare nelle immagini riportate nell'appendice B. In tale appendice sono raccolte anche alcune figure prodotte durante l'analisi degli altri dataset di immagini.

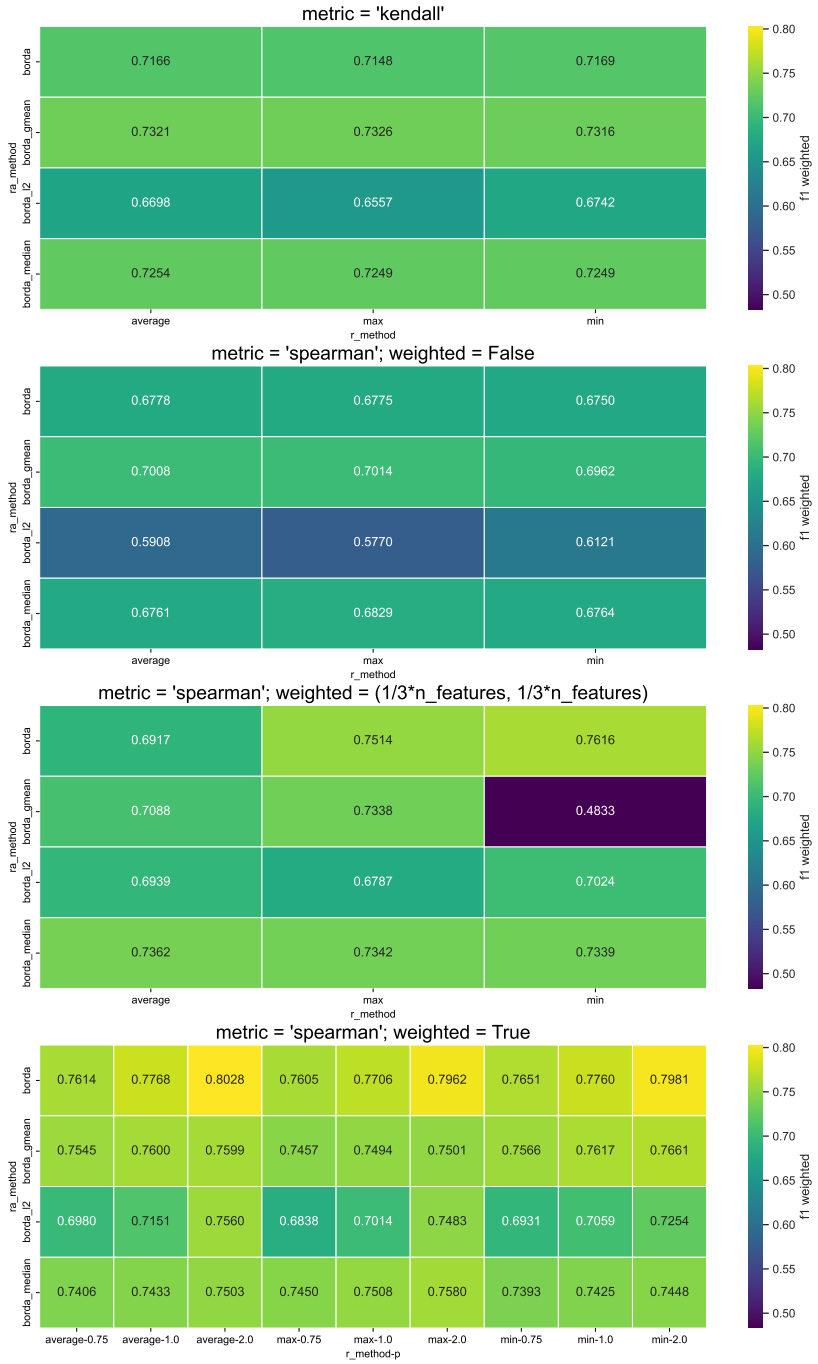

Figura 5.29: Heatmap dello score f_1 weighted di RAC con diverse combinazioni di iperparametri sul dataset MNIST.

Capitolo 6

Conclusioni

In questo lavoro è stato presentato un classificatore basato sul concetto di rank aggregation (RAC) che può essere applicato in contesti in cui le feature sono uniformi. Questo requisito scaturisce dal fatto che RAC trasforma in ranghi le feature, operazione che conduce a una perdita di informazioni, ma che può rivelarsi vantaggiosa perché determina una maggiore robustezza del classificare al rumore nei dati. Infatti, RAC si basa esclusivamente sull'ordinamento relativo delle feature e non sui valori che esse assumono, per cui può essere applicato a dati che presentano diversi range di misura senza dover ricorrere a una loro preventiva normalizzazione.

La classificazione dei campioni biologici sulla base dei rispettivi profili di espressione genica – problema di grande interesse in medicina per le sue potenzialità investigative, diagnostiche e prognostiche – costituisce una delle applicazioni di RAC più adeguate. L'ordinamento dei geni sulla base del loro livello di espressione può fornire, infatti, valide indicazioni per la classificazione dei campioni. Inoltre, RAC non richiede una selezione preliminare dei geni più significativi, ma identifica delle signature che dipendono dai trascritti di un vasto numero di sequenze geniche. Questo modo di operare potrebbe permettere di cogliere meglio la complessità dei meccanismi molecolari responsabili dell'instaurarsi di stati patologici. Oltre a ciò, a differenza di più sofisticati algoritmi per la classificazione, RAC è molto trasparente e facilmente interpretabile, caratteristiche queste che lo rendono adatto a essere applicato in ambito biomedico.

RAC è stato applicato a 12 diversi dataset di espressione genica ricavati da

esperimenti di microarray e a 3 dataset di immagini ed è stato valutato sia per problemi di classificazione binaria che multiclasse. Le prestazioni di RAC sono state confrontate con quelle di altri 5 classificatori allo stato dell'arte (Nearest Centroid, K-Nearest Neighbors, Support Vector Machine, Gaussian Naive Bayes, Random Forest). Su tutti i dataset è stata effettuata una double k-fold cross validation, misurando le prestazioni con le metriche f_1 weighted e AUROC. Dai risultati ottenuti risulta che le performance di RAC sono simili a quelle degli altri classificatori e non sono emerse caratteristiche dei dataset (in termini di numero di campioni e numero di feature) che influiscono in modo significativo sulle sue prestazioni. Il fatto che RAC abbia mostrato performance equiparabili a quelli degli altri classificatori su dataset di natura diversa – di immagini e di espressione genica – è indice della generalità e della flessibilità di tale metodo di classificazione.

L'implementazione di RAC in Python e la sua compatibilità con la libreria scikit-learn lo rendono uno strumento facilmente fruibile. Questo potrebbe favorire ulteriori sperimentazioni con altri tipi di dati al fine di individuarne nuove applicazioni.

6.1 Sviluppi futuri

Il lavoro svolto fornisce alcuni spunti per ulteriori sperimentazioni. Sarebbe interessante valutare la robustezza del classificatore applicandolo a dataset perturbati dall'introduzione di rumore generato artificialmente oppure applicandolo a dataset di espressione genica ottenuti integrando campioni di esperimenti diversi. RAC potrebbe infatti dimostrarsi resistente alle variazioni nei dati dovute ai diversi protocolli di laboratorio o a differenti tecniche di preprocessing e potrebbe favorire un'integrazione più rapida in quanto non richiede la normalizzazione dei dati. Per valutare la robustezza del classificatore in presenza di dataset rumorosi, oltre a esaminare come variano le sue prestazioni, tale indagine potrebbe essere condotta anche osservando le variazioni nelle signature delle classi. Altri sviluppi futuri potrebbero essere indirizzati all'implementazione di nuovi metodi per aggregare le liste e per misurare le distanze tra i vettori di ranghi delle istanze e le class signature. Sarebbe, inoltre, opportuno sviluppare un metodo per stimare la probabilità di appartenenza delle

istanze alle classi che consenta di ottenere delle stime meglio calibrate di quelle prodotte dal metodo implementato in questo lavoro. Infine, ulteriori studi potrebbero essere rivolti verso la possibilità di individuare nuovi ambiti di applicazione di RAC.

Appendice A

Dettagli sul calcolo delle probabilità delle classi

A.1 Esempio di calcolo

Di seguito viene mostrato un esempio di applicazione del metodo utilizzato da RAC per stimare la probabilità di appartenenza alle classi per istanze non note. La notazione utilizzata è quella introdotta nelle sezioni 4.1.4 e 4.1.5.

In presenza di 5 classi ($k = 5$), se le distanze tra il vettore dei ranghi dell'istanza da classificare e le signature delle classi sono $d_1 = d_2 = d_3 = 2$, $d_4 = 6$, $d_5 = 8$ e le distanze euclidee dalle classi in $I = \{1, 2, 3\}$ sono $e1 = 2$, $e2 = 4$, $e3 = 6$, conseguentemente si ha che

$$p_1 = p_2 = p_3 = p_l = \frac{1 - \frac{2}{2+2+2+6+8}}{5-1} = \frac{9}{40},$$

$$p_4 = p_{2ndl} = \frac{1 - \frac{6}{2+2+2+6+8}}{5-1} = \frac{7}{40},$$

$$p_5 = \frac{1 - \frac{8}{2+2+2+6+8}}{5-1} = \frac{6}{40},$$

$$p'_1 = \frac{1 - \frac{2}{2+4+6}}{3-1} = \frac{5}{12},$$

$$p'_2 = \frac{1 - \dfrac{4}{2+4+6}}{3-1} = \frac{4}{12},$$

$$p'_3 = \frac{1 - \dfrac{6}{2+4+6}}{3-1} = \frac{3}{12}.$$

Pertanto, le probabilità di appartenenza dell'istanza a ciascuna classe sono

$$p''_1 = \frac{7}{40} + 3 \cdot \frac{5}{12}\left(\frac{9}{40} - \frac{7}{40}\right) = \frac{19}{80} = 0.2375,$$

$$p''_2 = \frac{7}{40} + 3 \cdot \frac{4}{12}\left(\frac{9}{40} - \frac{7}{40}\right) = \frac{18}{80} = 0.225,$$

$$p''_3 = \frac{7}{40} + 3 \cdot \frac{3}{12}\left(\frac{9}{40} - \frac{7}{40}\right) = \frac{17}{80} = 0.2125,$$

$$p''_4 = p_4 = \frac{14}{80} = 0.175,$$

$$p''_5 = p_5 = \frac{12}{80} = 0.15.$$

A.2 Osservazioni sperimentali

Per valutare il metodo di calcolo della probabilità introdotto nella sezione 4.1.5 sono state realizzate le curve di calibrazione dei classificatori.

Una *curva di calibrazione* (o *diagramma di affidabilità*) [89] consente di valutare la sicurezza con cui un classificatore predice l'appartenenza delle istanze alle classi. In un problema di classificazione binaria, suddividendo in intervalli i valori delle probabilità che le istanze siano positive, la media delle probabilità appartenenti a ciascun intervallo corrisponde alle coordinate x dei punti della curva, mentre la frazione di istanze effettivamente positive che sono state predette tali con una probabilità appartenente a quell'intervallo corrisponde alle coordinate y dei punti della curva. Un classificatore è ben calibrato se la sua curva di calibrazione non si

discosta dalla bisettrice del quadrante. Quando si verifica ciò, infatti, la probabilità media di appartenenza alla classe positiva, predetta dal classificatore per un insieme di istanze, equivale all'effettiva percentuale di istanze positive in quell'insieme.

Le figure A.1–A.10 mostrano le curve di calibrazione dei classificatori e gli istogrammi delle frequenze delle probabilità da essi predette per le istanze di alcuni dataset di espressione genica. Le curve di calibrazione sono state realizzate suddividendo le probabilità predette in 5 intervalli di ampiezza tale da contenere lo stesso numero di istanze. Da questi grafici risulta che le probabilità calcolate con il metodo utilizzato da RAC e da NC sono mal calibrate. Sembra, inoltre, che NC effettui predizioni con un livello di sicurezza minore di quello di RAC.

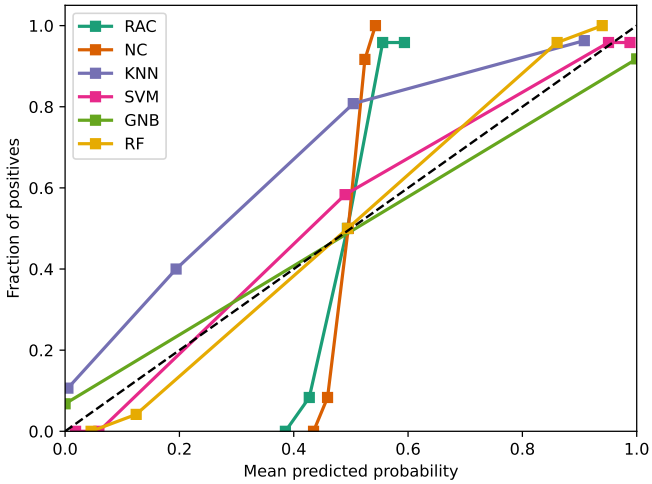

Figura A.1: Curve di calibrazione dei classificatori relative al dataset LungCancer1. Nel grafico viene considerata positiva la classe "Lung Cancer".

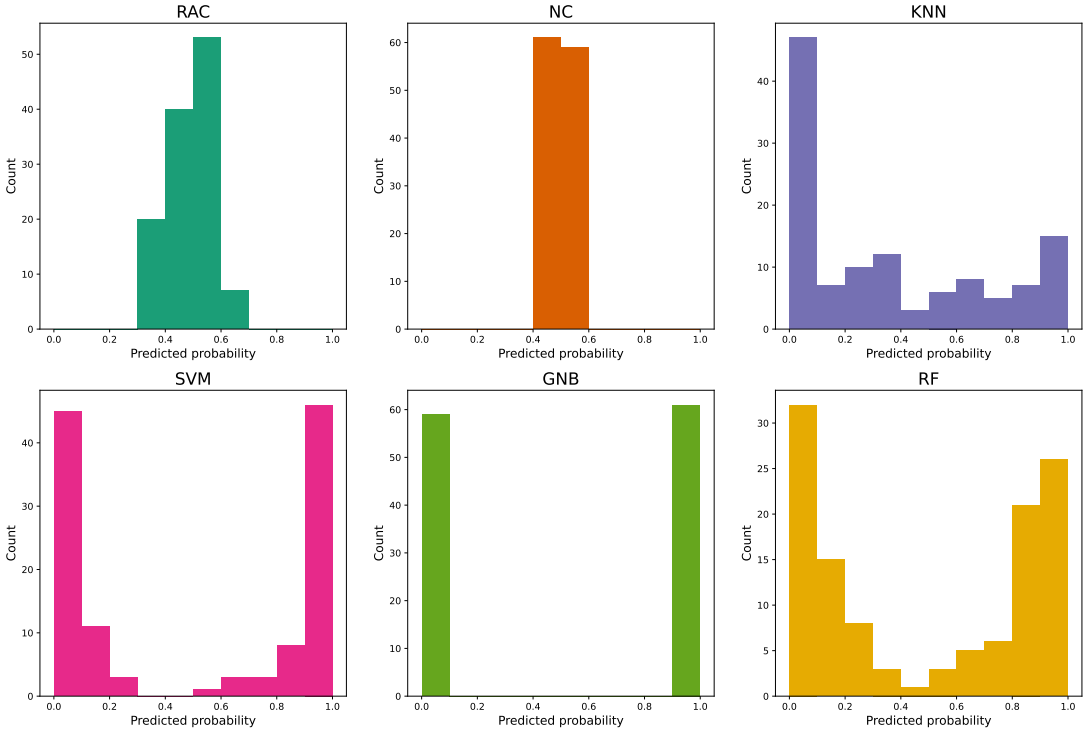

Figura A.2: Istogrammi delle probabilità delle predizioni dei classificatori relativi al dataset LungCancer1. Nel grafico viene considerata positiva la classe "Lung Cancer".

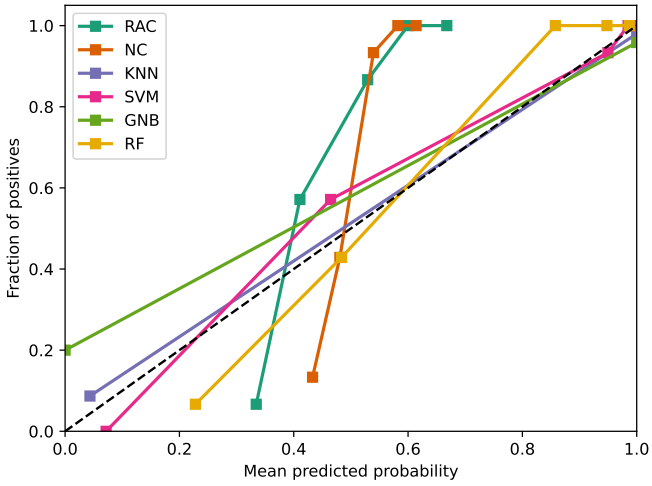

Figura A.3: Curve di calibrazione dei classificatori relative al dataset BreastCancer1 (Ectopic vs Malignant). Nel grafico viene considerata positiva la classe "Malignant".

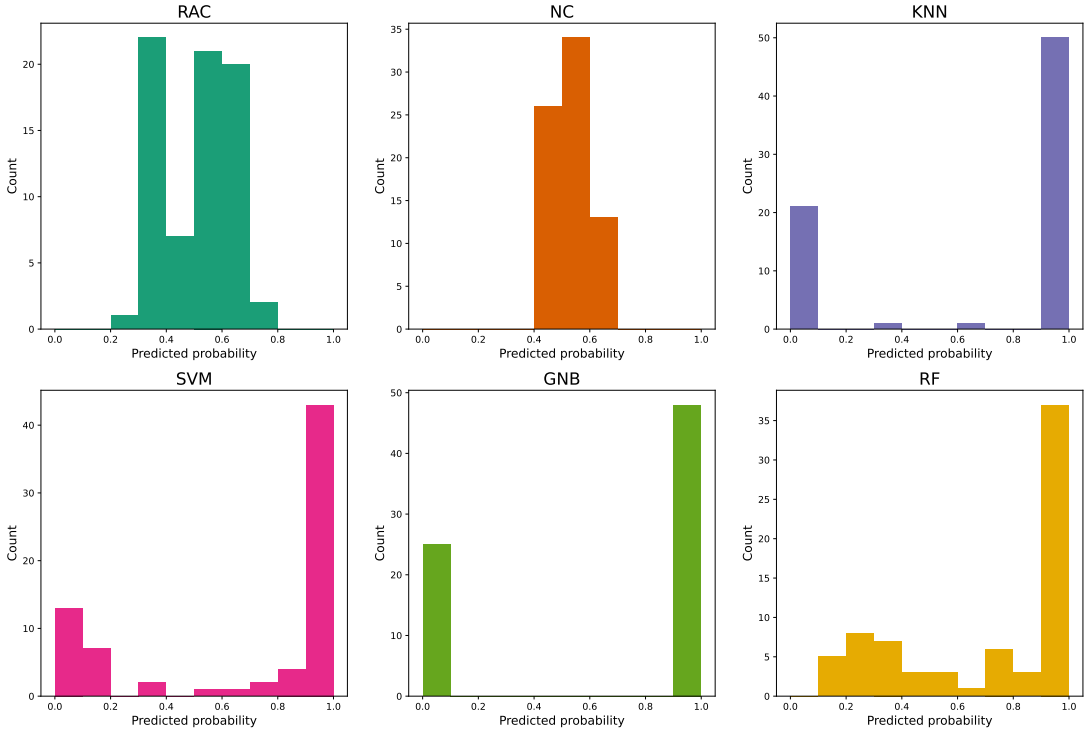

Figura A.4: Istogrammi delle probabilità delle predizioni dei classificatori relativi al dataset BreastCancer1 (Ectopic vs Malignant). Nel grafico viene considerata positiva la classe "Malignant".

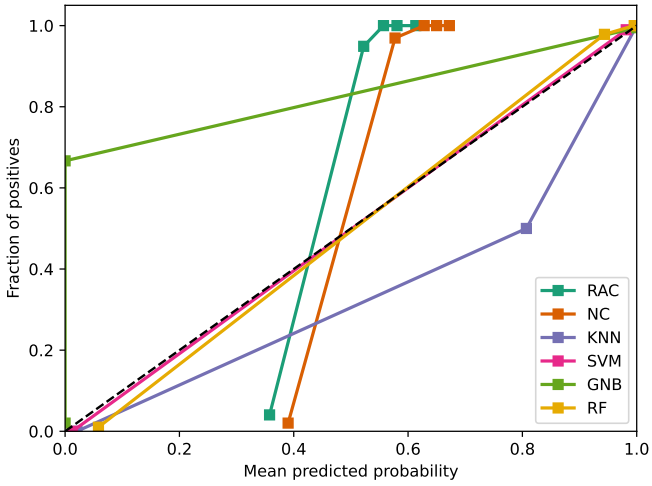

Figura A.5: Curve di calibrazione dei classificatori relative al dataset BladderCancer. Nel grafico viene considerata positiva la classe "Bladder Cancer".

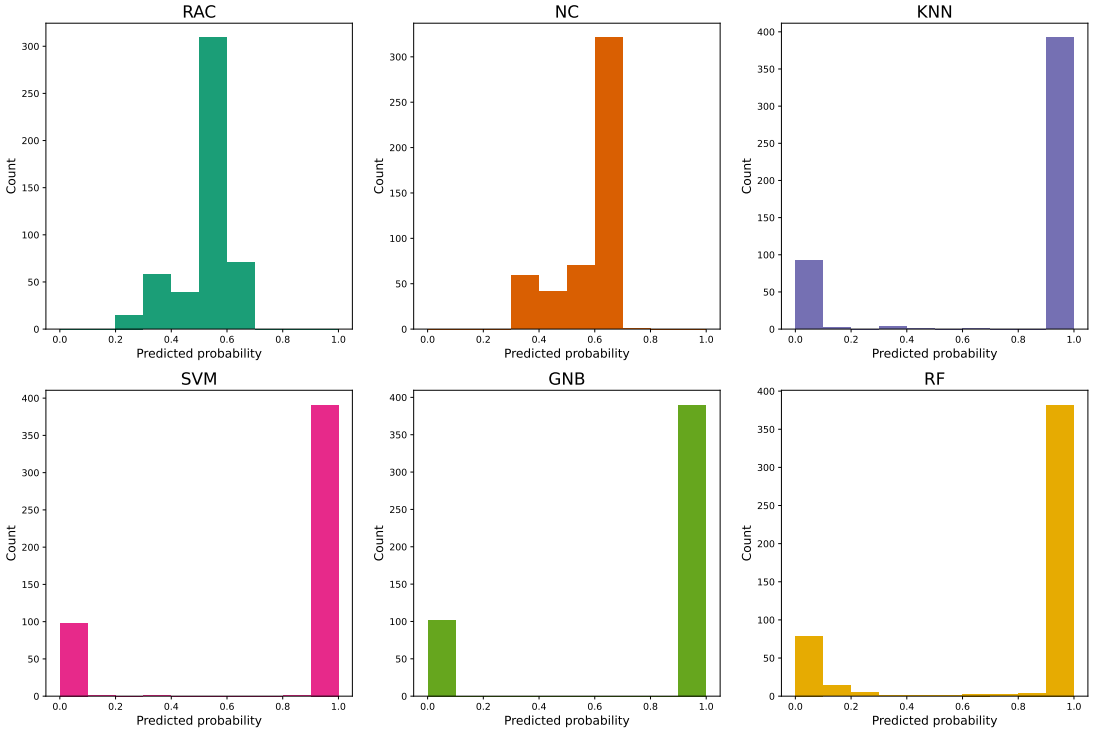

Figura A.6: Istogrammi delle probabilità delle predizioni dei classificatori relativi al dataset BladderCancer. Nel grafico viene considerata positiva la classe "Bladder Cancer".

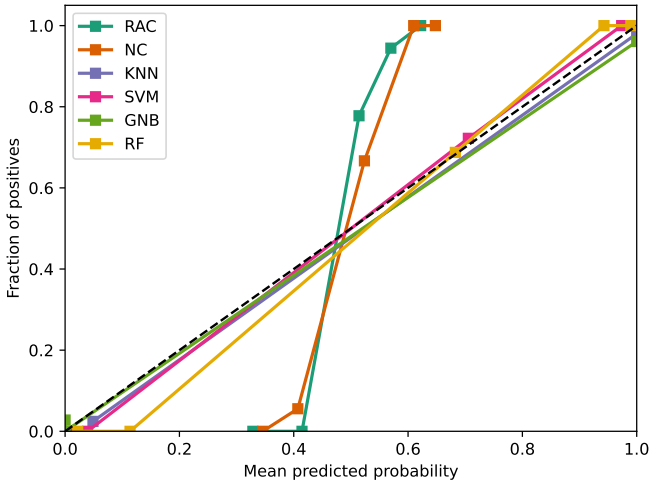

Figura A.7: Curve di calibrazione dei classificatori relative al dataset ProstateCancer (Non-cancer vs PC). Nel grafico viene considerata positiva la classe "PC".

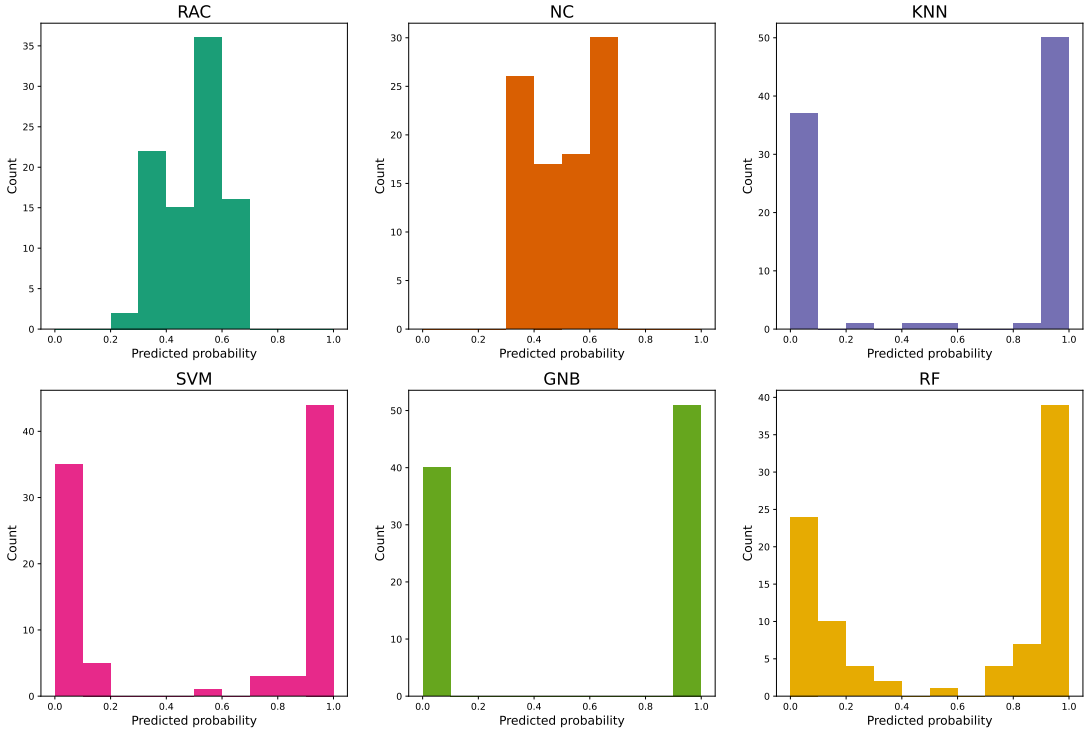

Figura A.8: Istogrammi delle probabilità delle predizioni dei classificatori relativi al dataset ProstateCancer (Non-cancer vs PC). Nel grafico viene considerata positiva la classe "PC".

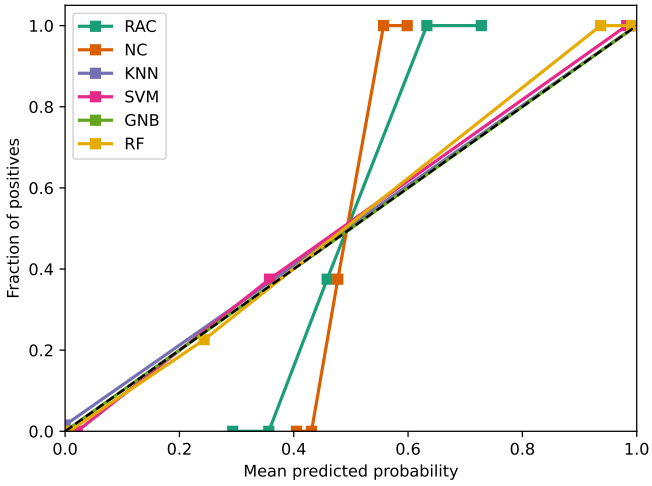

Figura A.9: Curve di calibrazione dei classificatori relative al dataset Psoriasis2. Nel grafico viene considerata positiva la classe "Psoriasis".

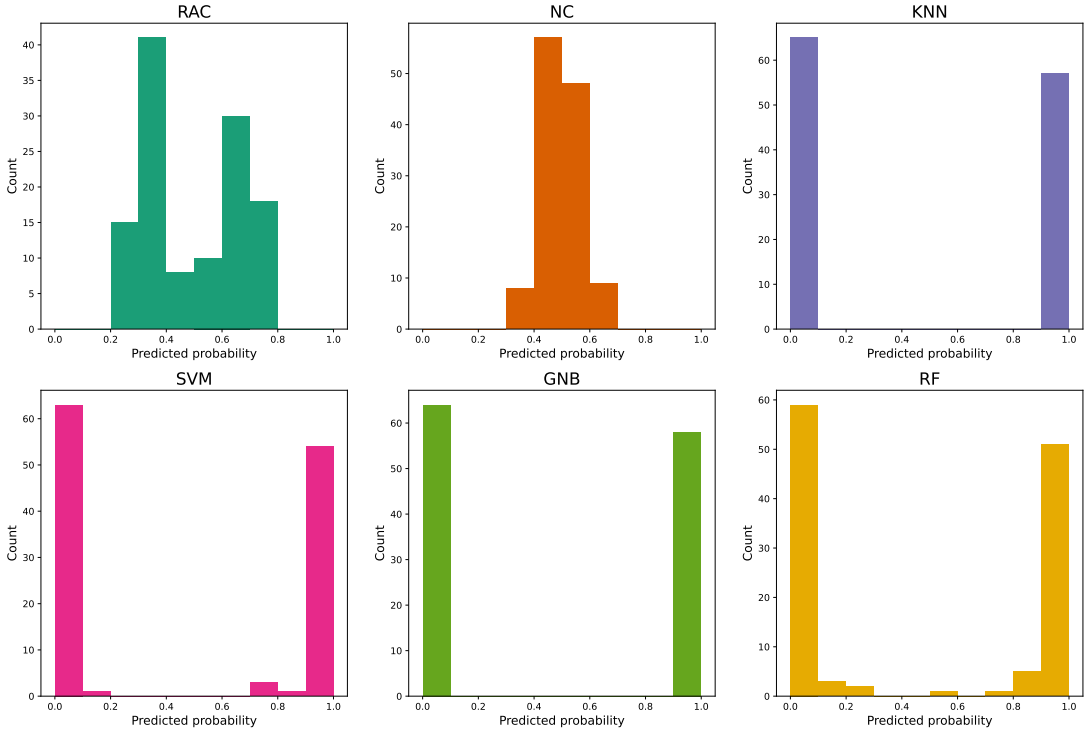

Figura A.10: Istogrammi delle probabilità delle predizioni dei classificatori relativi al dataset Psoriasis2. Nel grafico viene considerata positiva la classe "Psoriasis".

Appendice B

Applicazioni del classificatore a immagini

Le figure presenti in questa appendice mostrano alcune immagini ottenute applicando RAC ai dataset MNIST, Olivetti faces e CIFAR-10.

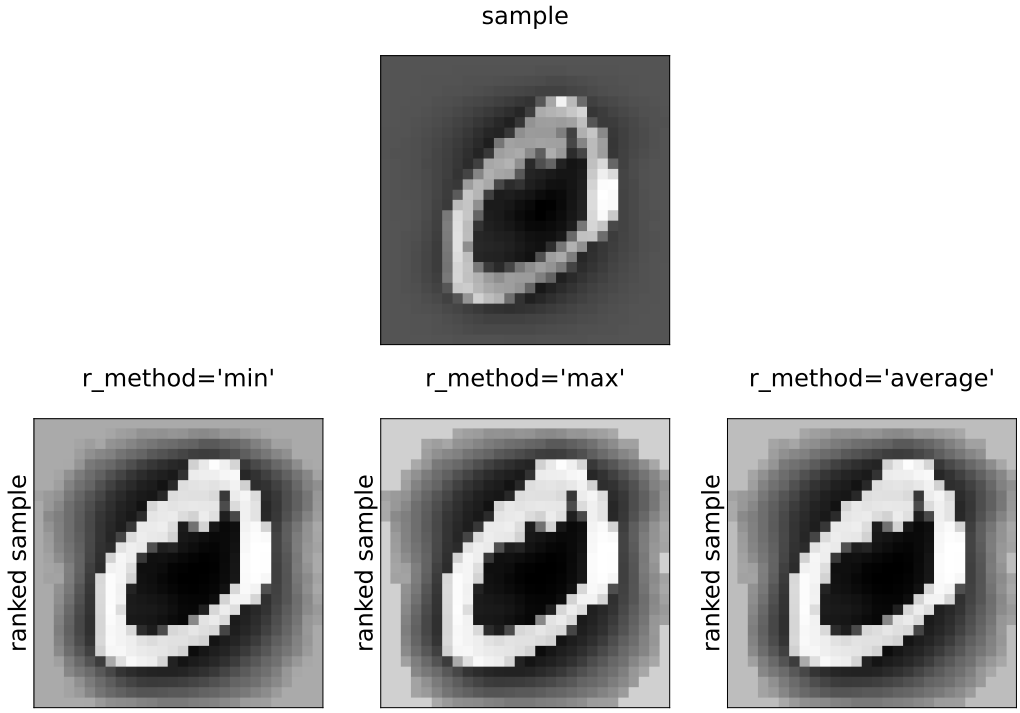

Figura B.1: Un'immagine della cifra 0 del dataset MNIST e di come appare dopo la trasformazione in ranghi con diversi metodi di assegnazione del rango.

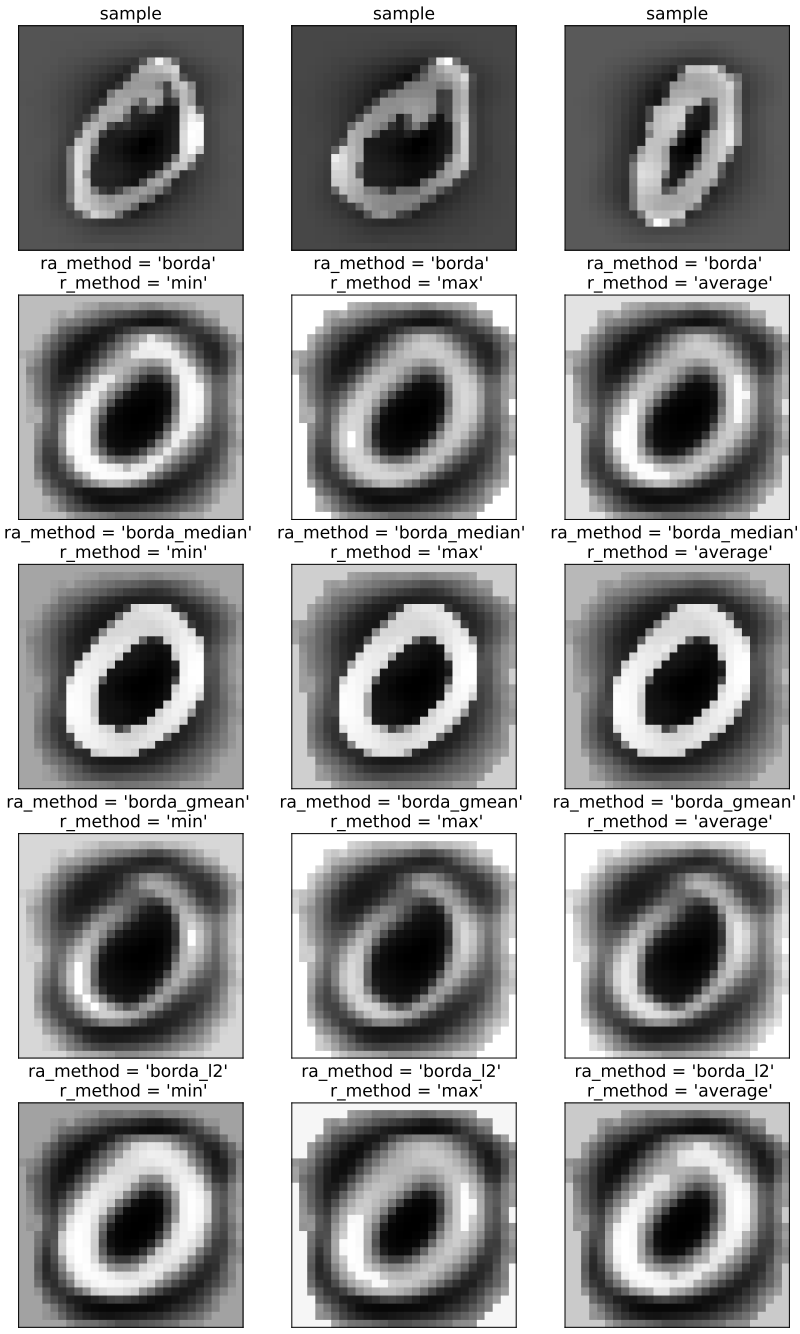

Figura B.2: Alcune immagini della cifra 0 del dataset MNIST e le signature della classe relativa prodotte da RAC con diversi metodi di assegnazione del rango e di rank aggregation.

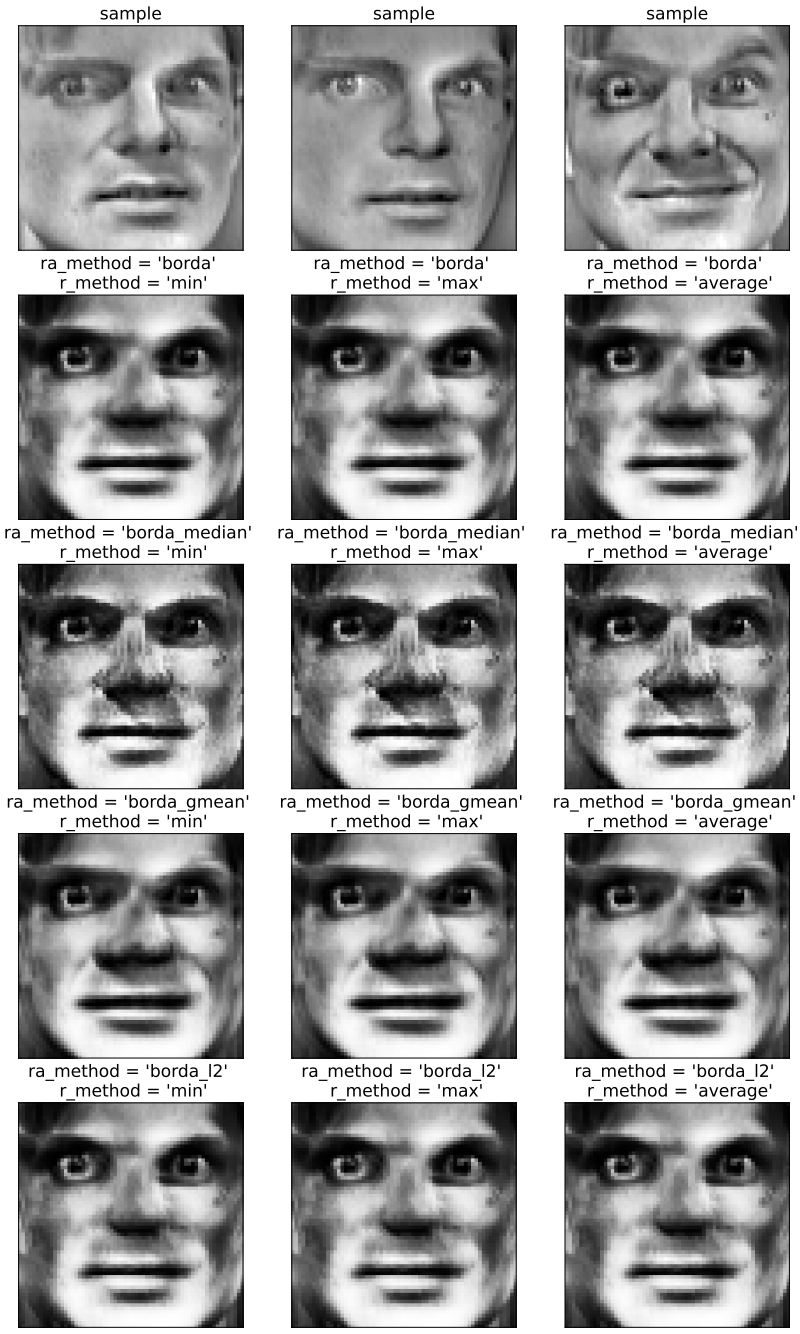

Figura B.3: Alcune immagini di un volto del dataset Olivetti faces e le signature della classe relativa prodotte da RAC con diversi metodi assegnazione del rango e di rank aggregation.

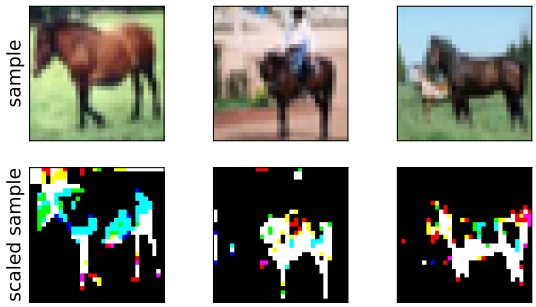

Figura B.4: Alcune immagini di cavalli del dataset CIFAR-10 e di come appaiono dopo aver standardizzato i valori dei pixel.

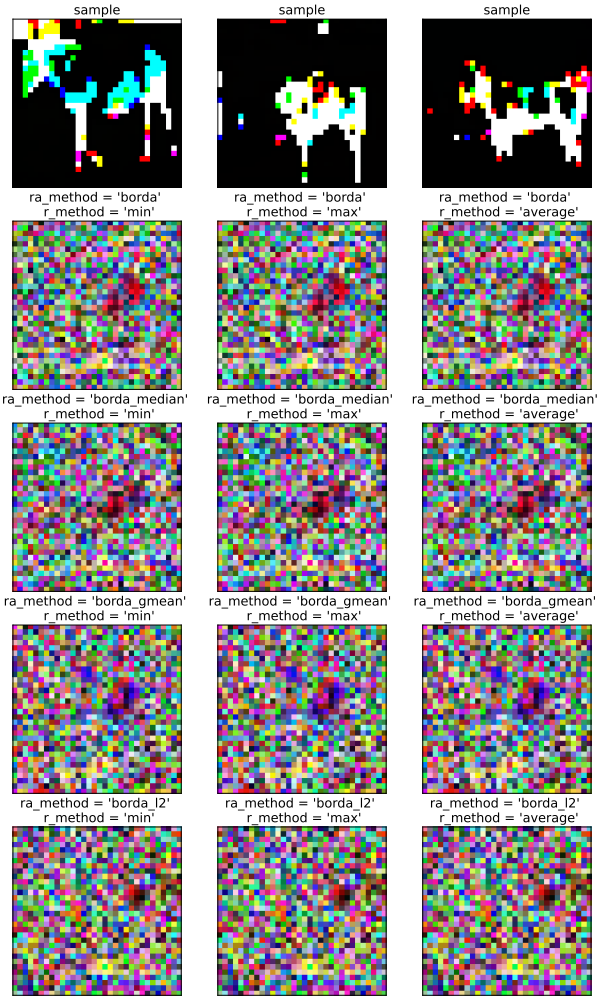

Figura B.5: Alcune immagini di cavalli del dataset CIFAR-10 e le signature della classe relativa prodotte da RAC con diversi metodi di assegnazione del rango e di rank aggregation.

Bibliografia

[1] G. Handelman, H. Kok, R. Chandra, A. Razavi, M. Lee, and H. Asadi, "ed octor: machine learning and the future of medicine," *Journal of internal medicine*, vol. 284, no. 6, pp. 603–619, 2018.

[2] F. Jiang, Y. Jiang, H. Zhi, Y. Dong, H. Li, S. Ma, Y. Wang, Q. Dong, H. Shen, and Y. Wang, "Artificial intelligence in healthcare: past, present and future," *Stroke and vascular neurology*, vol. 2, no. 4, pp. 230–243, 2017.

[3] Z. Obermeyer and E. J. Emanuel, "Predicting the future — big data, machine learning, and clinical medicine," *The New England journal of medicine*, vol. 375, no. 13, p. 1216, 2016.

[4] A. Rajkomar, J. Dean, and I. Kohane, "Machine learning in medicine," *New England Journal of Medicine*, vol. 380, no. 14, pp. 1347–1358, 2019.

[5] W. P. Kuo, E. Y. Kim, J. Trimarchi, T. K. Jenssen, S. A. Vinterbo, and L. Ohno-Machado, "A primer on gene expression and microarrays for machine learning researchers," *Journal of Biomedical Informatics*, vol. 37, no. 4, pp. 293–303, 2004.

[6] T. M. Mitchell, *Machine learning.* McGraw-Hill, 1997.

[7] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning.* Springer, 2006.

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data Mining, Inference, and Prediction.* Springer, 2009.

[9] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015.

[10] A. E. Eiben and J. E. Smith, *Introduction to evolutionary computing*. Springer, 2003.

[11] R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu, "Diagnosis of multiple cancer types by shrunken centroids of gene expression," *Proceedings of the National Academy of Sciences*, vol. 99, no. 10, pp. 6567–6572, 2002.

[12] V. Vapnik, *The nature of statistical learning theory*. Springer, 1999.

[13] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[14] J. Lever, "Classification evaluation: It is important to understand both what a classification metric expresses and what it hides," *Nature methods*, vol. 13, no. 8, pp. 603–605, 2016.

[15] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[16] E. P. Balogh, B. T. Miller, and J. R. Ball, "Improving diagnosis in health care," *Rapporto tecnico*, 2015.

[17] M. R. Trusheim, E. R. Berndt, and F. L. Douglas, "Stratified medicine: strategic and economic implications of combining drugs and clinical biomarkers," *Nature reviews Drug discovery*, vol. 6, no. 4, pp. 287–293, 2007.

[18] R. C. Deo, "Machine learning in medicine," *Circulation*, vol. 132, no. 20, pp. 1920–1930, 2015.

[19] A. J. Atkinson, W. A. Colburn, V. G. DeGruttola, D. L. DeMets, G. J. Downing, D. F. Hoth, J. A. Oates, C. C. Peck, R. T. Schooley, B. A. Spilker, J. Woodcock, and S. L. Zeger, "Biomarkers and surrogate endpoints: preferred definitions and conceptual framework," *Clinical pharmacology & therapeutics*, vol. 69, no. 3, pp. 89–95, 2001.

[20] P. Rajpurkar, E. Chen, O. Banerjee, and E. J. Topol, "Ai in health and medicine," *Nature Medicine*, vol. 28, no. 1, pp. 31–38, 2022.

[21] A. A. H. de Hond, A. M. Leeuwenberg, L. Hooft, I. M. J. Kant, S. W. J. Nijman, H. J. A. van Os, J. J. Aardoom, T. P. A. Debray, E. Schuit, M. van Smeden, J. B. Reitsma, E. W. Steyerberg, N. H. Chavannes, and K. G. M. Moons, "Guidelines and quality criteria for artificial intelligence-based prediction models in healthcare: a scoping review," *npj Digital Medicine*, vol. 5, no. 1, pp. 1–13, 2022.

[22] E. J. Topol, "High-performance medicine: the convergence of human and artificial intelligence," *Nature medicine*, vol. 25, no. 1, pp. 44–56, 2019.

[23] P. H. C. Chen, Y. Liu, and L. Peng, "How to develop machine learning models for healthcare," *Nature materials*, vol. 18, no. 5, pp. 410–414, 2019.

[24] J. M. Raser and E. K. O'shea, "Noise in gene expression: origins, consequences, and control," *Science*, vol. 309, no. 5743, pp. 2010–2013, 2005.

[25] D. Gomez-Cabrero, I. Abugessaisa, D. Maier, A. Teschendorff, M. Merkenschlager, A. Gisel, E. Ballestar, E. Bongcam-Rudloff, A. Conesa, and J. Tegnér, "Data integration in the era of omics: current and future challenges," *BMC systems biology*, vol. 8, no. 2, pp. 1–10, 2014.

[26] M. Zitnik, F. Nguyen, B. Wang, J. Leskovec, A. Goldenberg, and M. M. Hoffman, "Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities," *Information Fusion*, vol. 50, pp. 71–91, 2019.

[27] A. K. Waljee, A. Mukherjee, A. G. Singal, Y. Zhang, J. Warren, U. Balis, J. Marrero, J. Zhu, and P. D. Higgins, "Comparison of imputation methods for missing laboratory data in medicine," *BMJ open*, vol. 3, no. 8, p. e002847, 2013.

[28] R. E. Bellman, *Adaptive Control Processes*. Princeton University Press, 1961.

[29] R. Clarke, H. W. Ressom, A. Wang, J. Xuan, M. C. Liu, E. A. Gehan, and Y. Wang, "The properties of high-dimensional data spaces: implications for

exploring gene and protein expression data," *Nature reviews cancer*, vol. 8, no. 1, pp. 37–49, 2008.

[30] A. Vellido, "The importance of interpretability and visualization in machine learning for applications in medicine and health care," *Neural computing and applications*, vol. 32, no. 24, pp. 18069–18083, 2020.

[31] T. A. Brown, *Genomes*. BIOS Scientific Publishers Ltd, 2002.

[32] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown, "Quantitative monitoring of gene expression patterns with a complementary dna microarray," *Science*, vol. 270, no. 5235, pp. 467–470, 1995.

[33] M. Schena, D. Shalon, R. Heller, A. Chai, P. O. Brown, and R. W. Davis, "Parallel human genome analysis: microarray-based expression monitoring of 1000 genes," *Proceedings of the National Academy of Sciences*, vol. 93, no. 20, pp. 10614–10619, 1996.

[34] Z. Wang, M. Gerstein, and M. Snyder, "Rna-seq: a revolutionary tool for transcriptomics," *Nature reviews genetics*, vol. 10, no. 1, pp. 57–63, 2009.

[35] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.

[36] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, vol. 16, no. 10, pp. 906–914, 2000.

[37] M. P. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, M. Ares Jr, and D. Haussler, "Knowledge-based analysis of microarray gene expression data by using support vector machines," *Proceedings of the National Academy of Sciences*, vol. 97, no. 1, pp. 262–267, 2000.

[38] L. J. Van't Veer, H. Dai, M. J. Van De Vijver, Y. D. He, A. A. Hart, M. Mao, H. L. Peterse, K. Van Der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend, "Gene expression profiling predicts clinical outcome of breast cancer," *Nature*, vol. 415, no. 6871, pp. 530–536, 2002.

[39] Q. H. Ye, L. X. Qin, M. Forgues, P. He, J. W. Kim, A. C. Peng, R. Simon, Y. Li, A. I. Robles, Y. Chen, Z. C. Ma, Z. Q. Wu, S. L. Ye, Y. K. Liu, Z. Y. Tang, and X. W. Wang, "Predicting hepatitis b virus–positive metastatic hepatocellular carcinomas using gene expression profiling and supervised machine learning," *Nature medicine*, vol. 9, no. 4, pp. 416–423, 2003.

[40] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proceedings of the National Academy of Sciences*, vol. 95, no. 25, pp. 14863–14868, 1998.

[41] C. M. Perou, T. Sørlie, M. B. Eisen, M. Van De Rijn, S. S. Jeffrey, C. A. Rees, J. R. Pollack, D. T. Ross, H. Johnsen, L. A. Akslen, Ø. Fluge, A. Pergamenschikov, C. Williams, S. X. Zhu, P. E. Lønning, A.-L. Børresen-Dale, P. O. Brown, and D. Botstein, "Molecular portraits of human breast tumours," *Nature*, vol. 406, no. 6797, pp. 747–752, 2000.

[42] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. Hudson Jr, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, and L. M. Staudt, "Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling," *Nature*, vol. 403, no. 6769, pp. 503–511, 2000.

[43] C. List, "Social Choice Theory," in *The Stanford Encyclopedia of Philosophy*, Metaphysics Research Lab, Stanford University, 2022.

[44] J. C. Borda, "Mémoire sur les élections au scrutin," *Histoire de l'Academie Royale des Sciences*, 1781.

[45] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation methods for the web," in *Proceedings of the 10th international conference on World Wide Web*, pp. 613–622, ACM, 2001.

[46] R. Kolde, S. Laur, P. Adler, and J. Vilo, "Robust rank aggregation for gene list integration and meta-analysis," *Bioinformatics*, vol. 28, no. 4, pp. 573–580, 2012.

[47] R. C. Prati, "Combining feature ranking algorithms through rank aggregation," in *The 2012 international joint conference on neural networks (IJCNN)*, pp. 1–8, IEEE, 2012.

[48] A. Onan and S. Korukoğlu, "A feature selection model based on genetic rank aggregation for text sentiment classification," *Journal of Information Science*, vol. 43, no. 1, pp. 25–38, 2017.

[49] C. Spearman, "The proof and measurement of association between two things," *The American journal of psychology*, vol. 15, no. 1, pp. 72–101, 1904.

[50] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1/2, pp. 81–93, 1938.

[51] J. G. Kemeny, "Mathematics without numbers," *Daedalus*, vol. 88, no. 4, pp. 577–591, 1959.

[52] N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: ranking and clustering," *Journal of the ACM (JACM)*, vol. 55, no. 5, pp. 1–27, 2008.

[53] D. Coppersmith, L. Fleischer, and A. Rudra, "Ordering by weighted number of wins gives a good ranking for weighted tournaments," in *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pp. 776–782, SIGACT, 2006.

[54] A. V. Zuylen and D. P. Williamson, "Deterministic algorithms for rank aggregation and other ranking and clustering problems," in *International Workshop on Approximation and Online Algorithms*, pp. 260–273, Springer, 2007.

[55] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation revisited," *Manoscritto*, 2001.

[56] F. Schalekamp and A. V. Zuylen, "Rank aggregation: Together we're strong," in *2009 Proceedings of the Eleventh Workshop on Algorithm Engineering and Experiments (ALENEX)*, pp. 38–51, SIAM, 2009.

[57] X. Li, X. Wang, and G. Xiao, "A comparative study of rank aggregation methods for partial and top ranked lists in genomic applications," *Briefings in bioinformatics*, vol. 20, no. 1, pp. 178–189, 2019.

[58] M. G. Kendall, "The treatment of ties in ranking problems," *Biometrika*, vol. 33, no. 3, pp. 239–251, 1945.

[59] Scikit-learn. URL: https://scikit-learn.org.

[60] E. Anderson, "The species problem in iris," *Annals of the Missouri Botanical Garden*, vol. 23, no. 3, pp. 457–509, 1936.

[61] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[62] Y. Tu, G. Stolovitzky, and U. Klein, "Quantitative noise analysis for gene expression microarray experiments," *Proceedings of the National Academy of Sciences*, vol. 99, no. 22, pp. 14031–14036, 2002.

[63] L. Ein-Dor, O. Zuk, and E. Domany, "Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer," *Proceedings of the National Academy of Sciences*, vol. 103, no. 15, pp. 5923–5928, 2006.

[64] R. P. DeConde, S. Hawley, S. Falcon, N. Clegg, B. Knudsen, and R. Etzioni, "Combining results of microarray experiments: a rank aggregation approach," *Statistical applications in genetics and molecular biology*, vol. 5, no. 1, Articolo 15, 2006.

[65] S. Lin and J. Ding, "Integration of ranked lists via cross entropy monte carlo with applications to mrna and microrna studies," *Biometrics*, vol. 65, no. 1, pp. 9–18, 2009.

[66] V. Pihur, S. Datta, and S. Datta, "Rankaggreg, an r package for weighted rank aggregation," *BMC bioinformatics*, vol. 10, no. 1, pp. 1–10, 2009.

[67] M. Lauria, "Rank-based transcriptional signatures: a novel approach to diagnostic biomarker definition and analysis," *Systems Biomedicine*, vol. 1, no. 4, pp. 228–239, 2013.

[68] K. Rhrissorrakrai, J. J. Rice, S. Boue, M. Talikka, E. Bilal, F. Martin, P. Meyer, R. Norel, Y. Xiang, G. Stolovitzky, J. Hoeng, and M. C. Peitsch, "sbv improver diagnostic signature challenge: design and results," *Systems Biomedicine*, vol. 1, no. 4, pp. 196–207, 2013.

[69] D. Geman, C. d'Avignon, D. Q. Naiman, and R. L. Winslow, "Classifying gene expression profiles from pairwise mrna comparisons," *Statistical applications in genetics and molecular biology*, vol. 3, no. 1, pp. 1–19, 2004.

[70] A. C. Tan, D. Q. Naiman, L. Xu, R. L. Winslow, and D. Geman, "Simple decision rules for classifying human cancers from gene expression profiles," *Bioinformatics*, vol. 21, no. 20, pp. 3896–3904, 2005.

[71] Gene Expression Omnibus. URL: https://www.ncbi.nlm.nih.gov/geo/.

[72] F. Allantaz, D. T. Cheng, T. Bergauer, P. Ravindran, M. F. Rossier, M. Ebeling, L. Badi, B. Reis, H. Bitter, M. D'Asaro, A. Chiappe, S. Sridhar, G. D. Pacheco, M. E. Burczynski, D. Hochstrasser, J. Vonderscher, and T. Matthes, "Expression profiling of human immune cell subsets identifies mirna-mrna regulatory relationships correlated with cell type specific expression," *PloS one*, vol. 7, no. 1, p. e29979, 2012.

[73] T. P. Lu, M. H. Tsai, J. M. Lee, C. P. Hsu, P. C. Chen, C. W. Lin, J. Y. Shih, P. C. Yang, C. K. Hsiao, L. C. Lai, and E. Y. Chuang, "Identification of a novel biomarker, sema5a, for non–small cell lung carcinoma in nonsmoking women,"

Cancer Epidemiology and Prevention Biomarkers, vol. 19, no. 10, pp. 2590–2597, 2010.

[74] T. Sato, A. Kaneda, S. Tsuji, T. Isagawa, S. Yamamoto, T. Fujita, R. Yamanaka, Y. Tanaka, T. Nukiwa, V. E. Marquez, Y. Ishikawa, M. Ichinose, and H. Aburatani, "Prc2 overexpression and prc2-target gene repression relating to poorer prognosis in small cell lung cancer," *Scientific reports*, vol. 3, no. 1, pp. 1–9, 2013.

[75] H. G. LaBreche, J. R. Nevins, and E. Huang, "Integrating factor analysis and a transgenic mouse model to reveal a peripheral blood predictor of breast tumors," *BMC medical genomics*, vol. 4, no. 1, pp. 1–14, 2011.

[76] H. Zhao, J. Shen, L. Medico, D. Wang, C. B. Ambrosone, and S. Liu, "A pilot study of circulating mirnas as potential biomarkers of early stage breast cancer," *PloS one*, vol. 5, no. 10, p. e13735, 2010.

[77] R. S. Leidner, L. Li, and C. L. Thompson, "Dampening enthusiasm for circulating microrna in breast cancer," *PloS one*, vol. 8, no. 3, p. e57841, 2013.

[78] W. Usuba, F. Urabe, Y. Yamamoto, J. Matsuzaki, H. Sasaki, M. Ichikawa, S. Takizawa, Y. Aoki, S. Niida, K. Kato, S. Egawa, T. Chikaraishi, H. Fujimoto, and T. Ochiya, "Circulating mirna panels for specific and early detection in bladder cancer," *Cancer science*, vol. 110, no. 1, pp. 408–419, 2019.

[79] F. Urabe, J. Matsuzaki, Y. Yamamoto, T. Kimura, T. Hara, M. Ichikawa, S. Takizawa, Y. Aoki, S. Niida, H. Sakamoto, K. Kato, S. Egawa, H. Fujimoto, and T. Ochiya, "Large-scale circulating microrna profiling for the liquid biopsy of prostate cancer," *Clinical Cancer Research*, vol. 25, no. 10, pp. 3016–3025, 2019.

[80] Y. Yao, L. Richman, C. Morehouse, M. De Los Reyes, B. W. Higgs, A. Boutrin, B. White, A. Coyle, J. Krueger, P. A. Kiener, and B. Jallal, "Type I interferon: potential therapeutic target for psoriasis?," *PloS one*, vol. 3, no. 7, p. e2737, 2008.

[81] R. P. Nair, K. C. Duffin, C. Helms, J. Ding, P. E. Stuart, D. Goldgar, J. E. Gudjonsson, Y. Li, T. Tejasvi, B. J. Feng, A. Ruether, S. Schreiber, M. Weichenthal, D. Gladman, P. Rahman, S. J. Schrodi, S. Prahalad, S. L. Guthery, J. Fischer, W. Liao, P. Y. Kwok, A. Menter, G. M. Lathrop, C. A. Wise, A. B. Begovich, J. J. Voorhees, J. T. Elder, G. G. Krueger, A. M. Bowcock, and G. R. Abecasis, "Genome-wide scan reveals association of psoriasis with il-23 and nf-κb pathways," *Nature genetics*, vol. 41, no. 2, pp. 199–204, 2009.

[82] D. Shigemizu, S. Akiyama, Y. Asanomi, K. A. Boroevich, A. Sharma, T. Tsunoda, K. Matsukuma, M. Ichikawa, H. Sudo, S. Takizawa, T. Sakurai, K. Ozaki, T. Ochiya, and S. Niida, "Risk prediction models for dementia constructed by supervised principal component analysis using mirna expression data," *Communications biology*, vol. 2, no. 1, pp. 1–8, 2019.

[83] C. Feng, H. Wang, N. Lu, and X. M. Tu, "Log transformation: application and interpretation in biomedical research," *Statistics in medicine*, vol. 32, no. 2, pp. 230–239, 2013.

[84] MNIST dataset. URL: http://yann.lecun.com/exdb/mnist/.

[85] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[86] Olivetti faces dataset. URL: https://scikit-learn.org/stable/datasets/real_world.html#the-olivetti-faces-dataset.

[87] CIFAR-10 dataset. URL: https://www.cs.toronto.edu/~kriz/cifar.html.

[88] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Rapporto tecnico*, 2009.

[89] J. Bröcker and L. A. Smith, "Increasing the reliability of reliability diagrams," *Weather and forecasting*, vol. 22, no. 3, pp. 651–661, 2007.