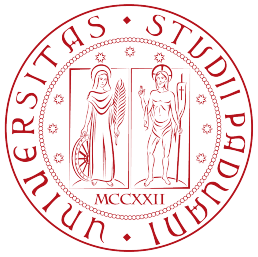


Università degli Studi di Padova

Dipartimento di Fisica e Astronomia Galileo Galilei

Corso di Laurea magistrale in Astronomia

Identificazione di buchi neri di massa intermedia attraverso tecniche di machine learning interpretabile

Relatore
Prof.ssa Michela Mapelli

Laureanda
Erica Greco

Correlatore
Dott. Mario Pasquato

24 Settembre 2020

Anno Accademico 2019-2020

Dedica

Spazio per citazione
autore

Ringraziamenti

Spazio per ringraziamenti

Sommario

I modelli teorici indicano che gli ammassi globulari possano ospitare buchi neri di massa intermedia (*Intermediate-Mass Black Holes*, IMBHs), ma i metodi osservativi usati finora non ne hanno ancora rilevato la presenza.

Rilevarli direttamente, infatti, può risultare difficoltoso con la strumentazione disponibile attualmente. Per questo motivo sono stati sviluppati metodi di identificazione indiretta. Uno di questi sfrutta l'informazione sul moto delle pulsar [**abbate1:paper**] ed in questo lavoro di tesi si è cercato di applicarlo attraverso una tecnica innovativa. Nello specifico, ci si riferisce all'utilizzo di un modello di *Machine Laerning* (ML) interpretabile: gli alberi decisionali di classificazione. In particolare, per costruire i *dataset* forniti al modello ci si è serviti di un campione di simulazioni a N-corpi di ammassi globulari. Quest'ultimo è stato implementato in un codice in grado di classificare gli ammassi secondo la presenza di un IMBH al loro interno sulla base delle caratteristiche dinamiche delle pulsar.

Si ottiene che l'albero decisionale allenato è in grado di classificare gli ammassi con il 70% di accuratezza attraverso l'utilizzo delle proprietà dinamiche di sole 20 pulsar selezionate per ogni ammasso.

Questo è solo un lavoro iniziale, infatti, il metodo può essere affinato e migliorato, ma nonostante questo presenta già dei risultati stimolanti per il futuro, in quanto sarebbe utile e innovativo applicarlo a dei dati reali.

Indice

Introduzione

I buchi neri di massa intermedia (*Intermediate-Mass Black Holes*, IMBHs) sono buchi neri la cui massa è compresa tra 10^2 e $10^6 M_\odot$. Si pensa che essi possano essere l'anello mancante necessario per collegare le informazioni note riguardanti i buchi neri di origine stellare ($\sim 10 M_\odot$) a quelle riguardanti i buchi neri super-massicci ($10^6 - 10^{10} M_\odot$). Infatti, i modelli teorici prevedono l'esistenza degli IMBHs al centro di ammassi globulari (*Globular Cluster*, GC), mentre le osservazioni condotte finora non ne hanno ancora rilevato la presenza a parte il recentissimo evento di detection di onde gravitazionali GW190521, che viene intepretato come la formazione di un buco nero di $\approx$ 150 masse solari [**Abbott2020:paper**], quindi in ogni caso di massa abbastanza vicina all'estremo stellare del range di massa degli IMBH_{MP}.

La motivazione per cui questi oggetti peculiari risultano sfuggenti rispetto ai buchi neri appartenenti alle altre categorie è legata principalmente al fatto che la strumentazione attuale presenti dei limiti in tal senso. Tuttavia, esistono dei metodi indiretti per poter identificare gli IMBHs negli ammassi. Uno fra essi è quello che si basa sulla dinamica delle MSPs nei GCs. In particolare, attraverso misure delle derivate dei periodi delle MSPs sarebbe possibile ottenere i loro valori di accelerazione e delle sue derivate: il *jerk* (derivata prima dell'accelerazione) e lo *snap* (derivata seconda dell'accelerazione). Dallo studio di tali caratteristiche, poi, si potrebbe risalire all'identificazione degli IMBHs [**abbate1:paper**].

A partire da questa proposta, questo lavoro si sviluppa verso tale scopo, ma utilizzando metodi diversi. Infatti, vengono utilizzate tecniche di *Machine Learning* interpretabile per prevedere la presenza di IMBHs al centro di GCs.

Poiché i dati osservativi in questo campo non sono sufficienti, per lo studio si sono utilizzati dati ottenuti da un set di simulazioni a N-corpi. Il modello di *Machine Learning* utilizzato è quello degli alberi decisionali, caratterizzati dalla possibilità di essere interpretati. Questa è un' importante caratteristica per un modello previsionale che riguarda soprattutto la fase di analisi dei risultati.

Gli ammassi simulati sono stati classificati dal modello secondo la presenza di un IMBH o meno sulla base delle caratteristiche dinamiche delle MSPs scelte nelle regioni centrali dei GCs. Nello specifico, le caratteristiche fornite al modello sono: la distanza delle MSPs dal centro di massa del GC proiettata sul piano del cielo e le componenti di velocità, accelerazione, *jerk* e *snap*, considerate lungo la linea di vista.

La ricerca di IMBHs attraverso questo metodo potrebbe essere una nuova strada alternativa da percorrere per ottenere dei risultati soddisfacienti, specie se tali metodi in futuro potranno essere applicati a dati reali.

Il lavoro di tesi può essere suddiviso principalmente in tre parti. La prima e la seconda parte si sviluppano rispettivamente nei capitoli e . La terza parte, invece, contiene i restanti tre capitoli.
Nel primo capitolo viene presentato e contestualizzato il problema astrofisico su cui si basa l'intera tesi: la ricerca degli IMBHs in ammassi globulari. Viene, dunque, fornita un'introduzione sugli IMBHs e su quali sono i metodi utilizzati finora per identificarli, per poi concentrarsi, in particolare, sul metodo delle MSPs.
Nel secondo capitolo, invece, viene descritto lo strumento utilizzato per lo sviluppo del progetto, ovvero gli alberi decisionali. Dopo una breve introduzione al *Machine Learning*, viene descritto l'algoritmo CART [**brei:book**] che è quello utilizzato per la costruzione degli alberi decisionali in questo lavoro. Vengono, infatti, descritte le fasi in cui l'algoritmo si sviluppa e le metodologie che esso adopera, come, ad esempio, la tecnica del *pruning* utilizzata in questo caso per risolvere un problema comune a tutti gli algoritmi di *Machine Learning*: il problema dell'*overfitting*. Inoltre, in questo capitolo, vengono forniti gli strumenti per valutare le prestazioni di un algoritmo di classificazione.
Infine, nell'ultima parte si entra nel merito del lavoro di tesi descrivendo

nello specifico i *dataset* utilizzati, il codice sviluppato per la classificazione degli ammassi (Cap.) e, in ultimo, vengono presentati i risultati ottenuti con la relativa interpretazione (Cap.). Nel quinto ed ultimo capitolo, invece, vengono discusse le conclusioni e presentate le possibili prospettive future per l'identificazione di IMBHs in GCs attraverso tecniche di *Machine Learning*.

A caccia di buchi neri di massa intermedia

In questo Capitolo viene presentato il *background* astrofisico sul quale si sviluppa l'intero lavoro di tesi.
Ci si focalizza sull'importanza di voler ricercare i buchi neri di massa intermedia (*Intermediate-Mass Black Holes*, IMBHs) all'interno degli ammassi globulari (*Globular Clusters*, GCs) e vengono presentate alcune tecniche utilizzate per la loro identificazione. In particolare, viene approfondito il metodo di identificazione basato sulla dinamica delle Pulsar Millisecondo (*Millisecond Pulsars, MSPs*) all'interno dei GCs. Infatti, questo progetto si sviluppa in una direzione diversa, ma complementare all'idea suggerita da Abbate el al.(2019) [**abbate1:paper**] di identificare gli IMBHs nei GCs utilizzando, oltre alle velocità e accelerazioni delle MSPs in ammasso, anche le informazioni dinamiche fornite dalle derivate dell'accelerazione.
Nello specifico, nella mia tesi, ho sviluppato un codice che utilizza un modello di *Machine Learning* (ML) interpretabile in grado di fare previsioni sulla presenza di IMBHs in ammassi sulla bese di tali informazioni dinamiche delle MSPs.

I buchi neri di massa intermedia

I buchi neri vengono classificati in base alla loro massa in: buchi neri di origine stellare, buchi neri di massa intermedia e buchi neri super-massicci. I buchi neri di origine stellare sono oggetti compatti che si formano alla fine della vita di stelle massicce in seguito ad esplosioni di Supernova, mentre

i buchi neri super-massicci normalmente si trovano al centro di galassie. I buchi neri di massa intermedia, invece, (*Intermediate-Mass Black Holes*, IMBHs) sono buchi neri la cui massa (tra 100 e 10^6 $M_\odot$) è significativamente maggiore di quella dei buchi neri di origine stellare e al contempo molto minore di quella dei buchi neri super-massicci.

I buchi neri sono caratterizzati da intensi campi gravitazionali dai quali, escludendo fenomeni quantistici non ancora verificati sperimentalmente, nulla può fuoriuscire, nemmeno la radiazione. Per questo motivo non possono essere osservati direttamente, ma possiamo unicamente misurare gli effetti che essi hanno sugli altri corpi celesti. Ad esempio, è possibile rilevare le emissioni X quando essi, in un sistema binario, accrescono sottraendo massa ad una stella compagna (Fig.1) o le emissioni sottoforma di onde gravitazionali quando si fondono (fenomeno di *merging*) con altri oggetti compatti o anche si possono rilevare gli effetti gravitazionali che hanno sulle orbite di altri oggetti, come nel caso di un buco nero super-massivo nel centro galattico.

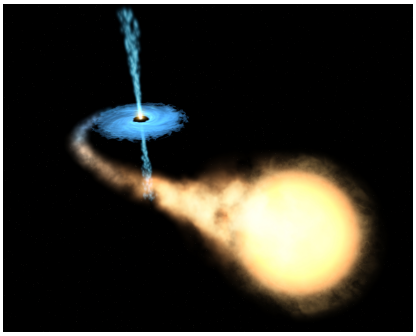


Figura 1: Esempio, in un *artist impression*, di buco nero in un sistema binario che accresce la propria massa a discapito della sua stella compagna [**binaria:online**].

Gli IMBHs, oggetto di studio di questa tesi, dal punto di vista osservativo, potrebbero rappresentare il nesso che collega le informazioni raccolte finora e che riguardano principalmente i buchi neri di origine stellare e quelli super-massici. Infatti, nonostante i modelli teorici suggeriscano fortemente che è possibile trovare gli IMBHs all'interno di ammassi globulari

[**milham:paper**], attualmente non si dispone di prove definitive per confermarlo, in quanto gli indizi che abbiamo sono ancora relativamente controversi e non si è ancora raggiunto il consenso presso la comunità scientifica (Fig.2).

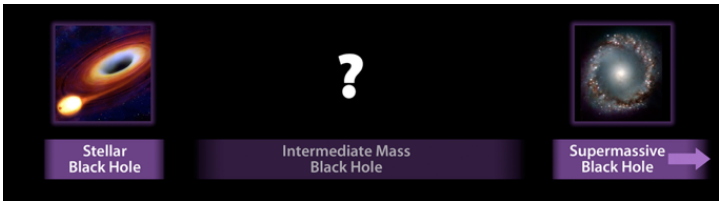


Figura 2: Vuoto di informazioni tra le categoria di buchi neri di origine stellare ed i buchi neri super-massivi: gli IMBHs.

Gli scenari di formazione proposti per questi oggetti peculiari sono molteplici [**jenny:paper**].
Il primo scenario ipotizza che gli IMBHs possono essersi originati in seguito al collasso di stelle di Popolazione III, ovvero ipotetiche stelle formatesi con abbondanza chimica primordiale che possono raggiungere alte masse rispetto alle stelle odierne [**abel:paper**].
La seconda ipotesi è quella del collasso diretto secondo cui la loro massa si sarebbe addensata direttamente dal materiale dell'universo in formazione subito dopo il *Big Bang*, senza passare per le fasi dell'evoluzione stellare [**haehnelt:paper**].
Un ultimo interessante scenario, invece, suggerisce che essi si formino mediante fenomeni di collisione e di *merging*. In questo caso esistono più ipotesi. Secondo *Miller & Hamilton 2002* [**milham:paper**] un buco nero di origine stellare di 50 $M_{\odot}$ potrebbe collocarsi in breve tempo al centro dell'ammasso per l'effetto di segregazione di massa; successivamente, collisioni con altri buchi neri stellari mediate da incontri gravitazionali a tre e quattro corpi e da perdita di energia per emissione di onde gravitazionali, porterebbero al raggiungimento di masse dell'ordine di 10^3 $M_{\odot}$ in un tempo paragonabile a quello di Hubble.
Un altro meccanismo, invece, propone che in un *core* ad alta densità, le stelle

molto massicce ($50 - 100 M_{\odot}$) possono essere soggette ad un'efficiente segregazione di massa che le colloca nel nucleo dell'ammasso mentre si trovano in fase di Sequenza Principale. A questo punto si verificherebbe un numero sempre crescente di collisioni e *merging* tra stelle che porterebbero alla formazione e all'immediato collasso di una stella con massa pari a $\simeq 10^{-3}$ volte la massa dell'ammasso, generando così un IMBH [**portzw:paper**].
I GCs, essendo ambienti stellari densi e dinamicamente attivi, potrebbero essere i luoghi ideali per la formazione di IMBHs attraverso collisioni stellari o incontri gravitazionali tra buchi neri di origine stellare e successive fusioni [**portmcmil:paper**]. L'interesse per la formazione di IMBHs in GCs è anche legato al fatto che gli ammassi globulari più massicci tendono a precipitare rapidamente verso il centro della galassia, dove potrebbero concorrere a formare il *nuclear star cluster* [**arcasedda:paper**]. Pertanto, essi rappresentano un possibile meccanismo di formazione di un buco nero super-massivo attraverso fenomeni di *merging* tra IMBHs.

In condizioni di prossimità ad una binaria di oggetti compatti gli IMBHs possono essere sorgenti di emissione di onde gravitazionali.
Gli interferometri terrestri di onde gravitazionali LIGO e Virgo [**ligo:online**], [**virgo:online**], [**ligovirgo1:paper**] operano in un intervallo di frequenze tra qualche decina e qualche migliaio di Hertz. Quindi possono osservare la coalescenza di buchi neri fino ad una massa di $\sim 500 M_{\odot}$, nel regime degli IMBHs.
La prossima generazione di interferometri nello spazio, come il Laser Interferometer Space Antenna, LISA [**amaro:paper**], sarà ancora più adatta ad osservare gli IMBHs, poiché opererà nel range di frequenza compreso tra $10^{-3} - 10^{-1}$ Hertz.
Inoltre, gli IMBHs sono molto difficili da individuare in quanto, gli ammassi globulari sono ambienti decisamente poveri di gas, poiché sono costituiti principalmente da stelle vecchie e il gas primordiale che era presente è stato utilizzato nelle precedenti fasi di formazione stellare o spazzato via da esplosioni di Supernova. Per tale ragione e per come operano i meccanismi di accrescimento, ovvero tramite la dissipazione di energia e momento angolare verso l'esterno di un disco, risulta difficile rilevare emissioni X, determinate da tali fenomeni.

Un'altra ragione per cui risulta difficoltoso rilevarli è legata alla componente stellare.
Per ogni buco nero di massa M_{BH}, infatti, è possibile stimare il raggio di influenza r_i:

$$r_i = \frac{GM_{BH}}{\sigma^2} \tag{1}$$

con G la costante di gravitazione universale e σ la dispersione di velocità delle stelle subito al di fuori della sfera di influenza del buco nero. Le stelle all'interno di questo raggio risentono principalmente dell'influenza gravitazionale del buco nero stesso.
Il raggio di influenza di un IMBH è dell'ordine di solo qualche secondo d'arco. Ad esempio, un IMBH di $\sim 1000 M_\odot$ all'interno di *47 Tucanae* avrebbe un raggio di influenza di circa 1" [**jenny:paper**], contro la dimensione apparente dell'ammasso di circa 31' [**47tuc:online**]. Pertanto, in generale, la sfera di influenza di un IMBH non è così importante, rispetto alle dimensioni dell'ammasso, da influenzare gravitazionalmente un numero di stelle sufficientemente elevato da determinare una quantità di eventi di accrescimento mareale tale da poter essere osservato.

Metodi di identificazione di IMBHs al centro di ammassi globulari

Di fronte alla presenza di un IMBH in un ammasso stellare, ci aspettiamo che si verifichino delle alterazioni nei profili di densità del GC e di dispersione di velocità delle stelle dell'ammasso. Sulla base di questi si costruiscono la maggior parte dei metodi di identificazione degli IMBHs con le relative difficoltà a livello pratico. Inoltre, se ci trovassimo in presenza di un fenomeno di accrescimento, la misurazione delle emissioni X e radio sarebbe un altro canale identificativo di IMBH al centro di un GC [**strader:paper**].

Un IMBH aumenta la profondità della buca di potenziale di un ammasso causando un incremento della densità stellare nelle zone centrali. Queste ultime, pertanto, risultano avere un profilo di densità di una *cuspide* ben descritta dalla legge di potenza $\rho \propto r^{-\alpha}$, con $\alpha \simeq 1.55$ [**baum:paper**].

Le stelle all'interno del raggio di influenza r_i (eq. 1) del IMBH, oltre a risentire principalmente dell'influenza gravitazionale del buco nero, seguono un profilo di dispersione di velocità di tipo *kepleriano* caratterizzato da una ripidità nelle regioni centrali. Tuttavia, la determinazione accurata della dispersione di velocità delle stelle valutata solo nella regione centrale del GC, è al limite delle capacità dell'attuale strumentazione astronomica e richiede un'elevatissima risoluzione spaziale. I risultati di misurazioni effettuate con metodi diversi (spettroscopia in luce integrata, stelle individuali, studio dei moti propri) sono talvolta in disaccordo tra loro, con la conseguenza che la comunità scientifica non ha ancora raggiunto un consenso in merito [**lutz1:paper**].
Il profilo di dispersione di velocità, però si può ottenere teoricamente sulla base del profilo di densità determinato precedentemente e considerando un oggetto puntiforme di una certa massa al centro dell'ammasso [**lutz:paper**]. Come risultato si ottiene una famiglia di profili di dispersione di velocità che, una volta confrontati con le osservazioni, restituiscono la massa del BH centrale (Fig.3).

Tuttavia, la presenza di una *cuspide* nelle regioni centrali di un GC può essere dovuta anche a cause diverse dalla presenza di un IMBH. Ad esempio, a seguito di un collasso del *core*, tanti buchi neri di massa stellare potrebbero trovarsi nelle regioni centrali dell'ammasso e determinare, quindi, i ripidi profili di velocità e densità associati alle cuspidi [**trenti:paper**]. Questo vuol dire, in altre parole, che la presenza di una cuspide non è sufficiente per affermare che al centro dell'ammasso ci sia un IMBH. Ma anche il viceversa non è sufficiente per affermare il contrario. Infatti, la non rilevazione di una cuspide non esclude a priori la presenza di un IMBH centrale. Come abbiamo già visto, infatti, la sfera di influenza di un IMBH potrebbe rivelarsi decisamente troppo piccola per far sì che la cuspide si veda.

Oltre ai metodi che si servono della dinamica stellare in ammasso e dei moti propri delle stelle [**banwolf:paper**], [**gebh:paper**], un altro metodo per l'dentificazione degli IMBH potrebbe essere quello che si basa sul rilevamento dell'emissione X e radio degli IMBHs in fase di accrescimento [**mac1:paper**], [**mac2:paper**].
Tali emissioni, però, risultano difficili da osservare nei GCs poichè si tratta

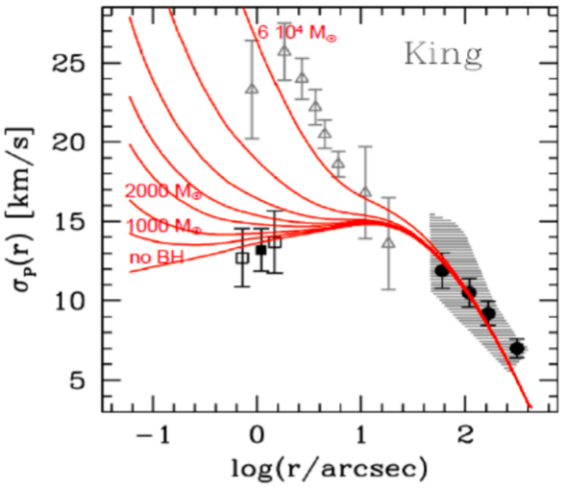


Figura 3: Profilo di dispersione di velocità osservato per l'ammasso globulare NGC 6388 con sovrapposte le linee rosse continue delle famiglie di modelli considerate. Esse sono state ottenute assumendo una diversa massa per il buco nero centrale. Si noti che aumentando la massa del IMBH la pendenza nelle regioni centrali risulta più ripida [**Lanzoni:paper**], [**lutz:paper**].

di sistemi poveri di gas. Tuttavia, recentemente, le prove dell'esistenza di un IMBH in un GC potrebbero essere fornite dall'osservazione di un evento di disturbo di marea in un ammasso extra-galattico [**lin:paper**]. I dati in X sono stati raccolti dai telescopi in orbita *Chandra* e *XMM-Newton* e successivamente confermati dal *Telescopio Spaziale Hubble*. Attualmente si stanno svolgendo ulteriori indagini osservative per identificare la natura dell'oggetto responsabile dell'evento, ma gli autori dello studio, *Lin D. et al. 2020* [**lin_2020:paper**], sostengono si tratti di un IMBH in procinto di inglobare una stella.

Oltre a questi, esistono anche metodi indiretti per rilevare IMBH al centro di ammassi. Uno tra questi è quello che si basa sull'effetto degli IMBHs sulla segregazione di massa [**pasquato:paper**]. Nei GCs, ci si aspetta, infatti, che gli IMBHs per la maggior parte del tempo si trovino in sistemi binari con altri oggetti massicci, come buchi neri di origine stellare. In questa configurazione essi inietterebbero energia nel nucleo del GC, attenuando la

segregazione di massa stellare. Anche le binarie primordiali, però, potrebbero essere responsabili dello stesso fenomeno. Ciò porterebbe ad un problema di degenerazione nell'indicatore di segregazione di massa, che potrebbe essere risolto misurando la frazione di binarie nel *core* in maniera indipendente. Questo, però, ha le sue complicazioni dovute ai problemi di risoluzione spaziale di cui si è già parlato. Anche per questo motivo gli studi portati avanti sulla base di tale metodologia non sono ancora stati in grado di rilevare forti candidati di GCs che ospiterebbero un IMBH.

Tutti i metodi presentati riscontrano delle difficoltà nell'identificazione degli IMBHs all'interno degli ammassi. Queste sono legate soprattuto ai fattori che ne influenzano la rilevazione diretta. Gli IMBHs, infatti, risultano così sfuggenti principalmente a causa dei problemi che comporta la piccola dimensione della loro sfera di influenza e, conseguentemente, dei problemi osservativi relativi alla risoluzione spaziale per i *core* degli ammassi. Per tali ragioni risulta interessante approfondire un metodo alternativo e indiretto per la ricerca di IMBHs all'interno di ammassi globulari: quello che si basa sullo studio delle proprietà dinamiche delle Pulsar Millisecondo (*Millisecond Pulsars, MSPs*) [**pere:paper**], [**abbate:paper**]. I paragrafi successivi, infatti, saranno dedicati all'approfondimento di tale metodo che è proprio quello su cui si basa questo progetto di tesi.

Metodo delle MSPs

Le MSPs sono oggetti molto frequenti nei GCs ed essendo caratterizzate da periodi di rotazione estremamente stabili, possono essere utilizzate come strumento per l'identificazione di IMBHs all'interno di ammassi globulari. Infatti, le misure ottenute per effetto Doppler permettono di avere informazioni sulle loro accelerazioni e sulle derivate temporali di ordine superiore. In particolare, le derivate prima, detta *jerk*, e seconda, detta *jounce* o *snap*, dell'accelerazione delle MSPs in ammassi, sono state valutate in maniera più approfondita nel recente articolo di Abbate et al. (2019) [**abbate1:paper**]. Pero l'articolo di Abbate e puramente simulativo; andrebbe scovata una referenza osservativa su che cosa si puo veramente misurare sulle pulsar reali$_{\text{MP}}$

Profili radiali di Jerk e Snap in ammasso

Analiticamente è possibile derivare le relazioni matematiche che descrivono i profili radiali di *jerk* e *snap* di stelle in ammasso [**abbate1:paper**].

Consideriamo una stella di prova che sperimenta l'attrazione gravitazionale del campo generato da tutte le stelle dell'ammasso, in particolare assumiamo che il GC sia descritto dal profilo di King [**king:paper**].
L'accelerazione in funzione della distanza r dal centro dell'ammasso, nelle regioni centrali del GC, può essere approssimata come:

$$\mathbf{a}(r) = -4\pi G\rho_c r_c^3 \left[\sinh^{-1}\left(\frac{r}{r_c}\right) - \frac{r}{r_c\sqrt{1+(r/r_c)^2}}\right]\frac{\mathbf{r}}{r^3} = -\,|a|\,\frac{\mathbf{r}}{r} \qquad (2)$$

dove ρ_c è la densità centrale dell'ammasso ed r_c il raggio del *core*.
Per calcolare il *jerk* è necessario derivare rispetto al tempo l'equazione 2, ottenendo:

$$\dot{\mathbf{a}}_K(r) = -\frac{d|a(r)|}{dt}\frac{\mathbf{r}}{r} - |a(r)|\frac{\mathbf{v}}{r} + |a(r)|\frac{(\mathbf{v}\cdot\mathbf{r})\mathbf{r}}{r^3} \qquad (3)$$

in cui la derivata temporale della norma dell'accelerazione è data da:

$$\frac{d|a(r)|}{dt} = -2\frac{v|a(r)|}{r} + 4\pi G v\rho_c\left(\frac{1}{1+(r/r_c)^2}\right)^{\frac{3}{2}} \qquad (4)$$

con v la norma della velocità.
Nel caso in cui il GC fosse caratterizzato da forti fenomeni di collisioni stellari, anche i *jerk* potrebbero risultare influenzati dalle stelle vicine. In questo caso, il *jerk* sarebbe caratterizzato dalla seguente distribuzione di probabilità [**prager:paper**]:

$$P(\dot{a}) = \frac{1}{\pi^2}\frac{\dot{a}_0}{\left(\dot{a}^2+\dot{a}_0^2\right)^2} \qquad (5)$$

in cui $\dot{a}_0$ è il *jerk* caratteristico dato da:

$$\dot{a}_0 = \frac{2\pi\xi}{3}G\langle m\rangle\sigma n \qquad (6)$$

dove ξ=3.04 è una costante numerica, $\langle m\rangle$ è la massa media delle stelle, σ è la dispersione di velocità ed n è la densità numerica delle stelle.

La distribuzione dei *jerk* proiettata lungo la linea di vista $\dot{a}_l$ è una distribuzione Lorentziana:

$$P\left(\dot{a}_l\right) = \frac{1}{\pi}\frac{\dot{a}_0}{\dot{a}_l^2 + \dot{a}_0^2} \tag{7}$$

Se all'interno del GC vi è un IMBH, il *jerk* della stella di prova sarà influenzato dalla massa centrale M ed il profilo sarà:

$$\dot{\mathbf{a}}_M = -GM\left(\frac{\mathbf{v}}{r^3} - 3\frac{(\mathbf{v}\cdot\mathbf{r})\mathbf{r}}{r^5}\right) \tag{8}$$

dove $\mathbf{r}$ è la distanza dalla massa M e $\mathbf{v}$ è la relativa velocità.
Inoltre l'IMBH crea una sovra-densità stellare caratterizzata da un profilo radiale con *slope* pari a -1.55 [**baum:paper**]:

$$\dot{\mathbf{a}}_{\rm cusp} = \begin{cases} -\frac{4\pi G}{1.45} r_{\rm i}^{1.55}\rho_{\rm i}\left(\frac{\rm v}{r^{1.55}} - 1.55\frac{({\rm v}\cdot{\rm r}){\rm r}}{r^{3.55}}\right) & \text{for } r < r_{\rm i} \\ -\frac{4\pi G}{1.45} r_{\rm i}^{3}\rho_{\rm i}\left(\frac{\rm v}{r^{3}} - 3\frac{({\rm v}\cdot{\rm r}){\rm r}}{r^{5}}\right) & \text{for } r > r_{\rm i} \end{cases} \tag{9}$$

in cui r_i è il raggio di influenza dell'IMBH e ρ_i è il valore di densità a tale raggio.

Derivando rispetto al tempo le equazioni 3, 8 e 9 è possibile ottenere anche i profili radiali degli *snap* all'interno degli ammassi.
Il contributo del campo gravitazionale dell'intero ammasso è dato dalla:

$$\begin{aligned} \ddot{\mathbf{a}}_K = &-\frac{d^2|a(r)|}{dt^2}\frac{\mathbf{r}}{r} - 2\frac{d|a(r)|}{dt}\frac{\mathbf{v}}{r} + 2\frac{d|a(r)|}{dt}\frac{(\mathbf{v}\cdot\mathbf{r})\mathbf{r}}{r^3} + \\ &+ 5|a(r)|\frac{(\mathbf{v}\cdot\mathbf{r})\mathbf{v}}{r^3} - 3|a(r)|\frac{(\mathbf{v}\cdot\mathbf{r})^2\mathbf{r}}{r^5} \end{aligned} \tag{10}$$

Analogamente a quanto calcolato per i *jerk*, anche per gli *snap* la presenza di un IMBH al centro dell'ammasso porta a due contributi.
Il contributo che dipende direttamente dalla massa centrale M:

$$\ddot{\mathbf{a}}_M = GM\left(-2a\frac{\mathbf{r}}{r^4} - 6\frac{(\mathbf{v}\cdot\mathbf{r})\mathbf{v}}{r^5} - 3\frac{v^2\mathbf{r}}{r^5} + 15\frac{(\mathbf{v}\cdot\mathbf{r})^2\mathbf{r}}{r^7}\right) \tag{11}$$

ed il contributo che determina una sovra-densità. Per quest'ultimo distinguiamo due casi.
Per $r < r_i$:

$$\begin{aligned} \ddot{\mathbf{a}}_{\rm cusp} = &-\tfrac{4\pi G}{1.45} r_1^{1.55}\rho_i\left(-0.45\tfrac{a\mathbf{r}}{r^{2.55}} - 3.1\tfrac{(\mathbf{v}\cdot\mathbf{r})\mathbf{v}}{r^{3.55}} - \right. \\ &\left. -1.55\tfrac{v^2\mathbf{r}}{r^{3.55}} + 5.5\tfrac{(\mathbf{v}\cdot\mathbf{r})^2\mathbf{r}}{r^{5.55}}\right) \end{aligned} \tag{12}$$

e per $r > r_i$:

$$\ddot{\mathbf{a}}_{\rm cusp} = -\frac{4\pi G}{1.45} r_{\rm i}^3 \rho_{\rm i} \left(-2\frac{a\mathbf{r}}{r^4} - 6\frac{(\mathbf{v}\cdot\mathbf{r})\mathbf{v}}{r^5} - 5\frac{v^2\mathbf{r}}{r^5} + \right. \\ \left. + 15\frac{(\mathbf{v}\cdot\mathbf{r})^2\mathbf{r}}{r^7} \right) \tag{13}$$

Metodo osservativo

Un ammasso globulare normalmente è caratterizzato in media da un numero di pulsar molto basso, solo in alcuni casi si raggiunge l'ordine di qualche decina di pulsar [**pulsar:online**]. Esse si localizzano nelle regioni più interne degli ammassi e di solito circa la metà si trova in sistemi binari.

Dal punto di vista pratico, per identificare IMBHs all'interno di GCs basterebbero le informazioni fornite dalle accelerazioni se avessimo a disposizione un numero consistente di pulsar per ogni ammasso. Ma per ovviare al problema delle poche pulsar normalmente presenti in questi sistemi, si potrebbero estrarre più informazioni da ognuna di esse considerando anche *jerk* e *snap*. Per mezzo delle osservazioni, queste quantità si potrebbero ottenere proprio grazie ai periodi prolungati delle MSPs nei GCs.
Infatti, le misure delle derivate seconda e terza del periodo di rotazione di un insieme di MSPs in un GC galattico sono fondamentali perché correlano con la componente lungo la linea di vista di *jerk* e *snap*.
I tempi di osservazione richiesti per ottenere tali informazioni e per raggiungere una precisione adeguata sono molto lunghi, dell'ordine di diversi anni. Gli autori, però, ritengono che portando avanti queste campagne osservative in maniera regolare, le MSPs potrebbero diventare un buon strumento per l'identificazione di IMBHs al centro degli ammassi.

La derivata prima del periodo è principalmente influenzata dalla componente dell'accelerazione misurata lungo la linea di vista, mentre le derivate seconda e terza del periodo dipendono in maniera diretta da *jerk* e *snap* rispettivamente.
La relazione tra accelerazione lungo la linea di vista a_c e derivata del periodo $\dot{P}$ della MSPs è data da:

$$\left(\frac{\dot{P}}{P}\right)_{\rm meas} = \left(\frac{\dot{P}}{P}\right)_{\rm int} + \frac{a_{\rm c}}{c} + \frac{a_{\rm g}}{c} + \frac{\mu^2 D}{c} \tag{14}$$

in cui $(\dot{P}/P)_{\rm int}$ è la componente intrinseca dovuta allo *spin-down* della pulsar, a_g è l'accelerazione dovuta al potenziale galattico lungo la linea di vista e l'ultimo addendo rappresenta l'effetto Shklovskii [**shklov:paper**], in cui μ è il moto proprio della pulsar, D è la distanza dell'ammasso dal Sole e c è la velocità della luce.
Gli ultimi due termini generalmente possono essere trascurati rispetto al contributo dato dal termine a_c [**abbate:paper**]. Purtroppo, però, è molto difficile distinguere gli effetti dell'accelerazione dell'ammasso dallo *spin-down* intrinseco. Qualsiasi lavoro focalizzato sulla misurazione dell'accelerazione in un GC a partire da $\dot{P}$, avrà grandi incertezze dovute all'ignoto *spin-down* intrinseco.
Tuttavia, in maniera indipendente è possibile stimare l'accelerazione delle MSPs in ammasso solo se esse appartengono ad un sistema binario utilizzando l'effetto Doppler.

Per quanto riguarda *jerk* e *snap* la situazione cambia.
Le relazioni che legano le derivate seconda e terza del periodo a *jerk*, $\dot{a}_c$, e *snap*, $\ddot{a}_c$, rispettivamente sono date da:

$$\left(\frac{\ddot{P}}{P}\right)_{\rm meas} = \frac{\ddot{\rm a}_{\rm c}}{c} + \left(\frac{\ddot{P}}{P}\right)_{\rm int} \tag{15}$$

$$\left(\frac{\dddot{P}}{P}\right)_{\rm meas} = \frac{\dddot{\rm a}_{\rm c}}{c} + \left(\frac{\dddot{P}}{P}\right)_{\rm int} \tag{16}$$

In questi casi i termini di *spin-down* sono praticamente trascurabili (per ulteriori chiarimenti si rimanda il lettore a [**abbate1:paper**]). Questo vuol dire che misure di derivata seconda e terza del periodo corrispondono direttamente a misure di *jerk* e *snap*.

Simulazioni e sviluppi con modelli di Machine Learning

Abbate et al. (2019) [**abbate1:paper**] sviluppano un set di simulazioni a N-corpi di ammassi stellari in cui calcolano *jerk* e *snap* in maniera autoconistente, trattando le MSPs come particelle di prova.

In primo luogo, dimostrano che la presenza di un IMBH influenza l'andamento di *jerk* e *snap* in funzione della distanza dal centro del GC, specialmente nelle regioni centrali dell'ammasso (Fig. 4).

Successivamente, a seguito di un'analisi condotta utilizzando la statistica Bayesiana, gli autori mostrano che con le derivate delle accelerazioni, in particolare con i *jerk*, per identificare un IMBH di massa dell'ordine di $10^2 M_\odot$ è necessario avere a disposizione un numero di MSPs pari a circa 40, di cui 20 devono trovarsi in sistemi binari.

Sulla base di tale lavoro, nasce poi l'idea nuova e sviluppata in questa tesi di prevedere la presenza degli IMBHs all'interno dei GCs tramite un modello di ML. I dati utilizzati nella tesi sono stati ottenuti dalle simulazioni di ammassi globulari condotte dagli autori.

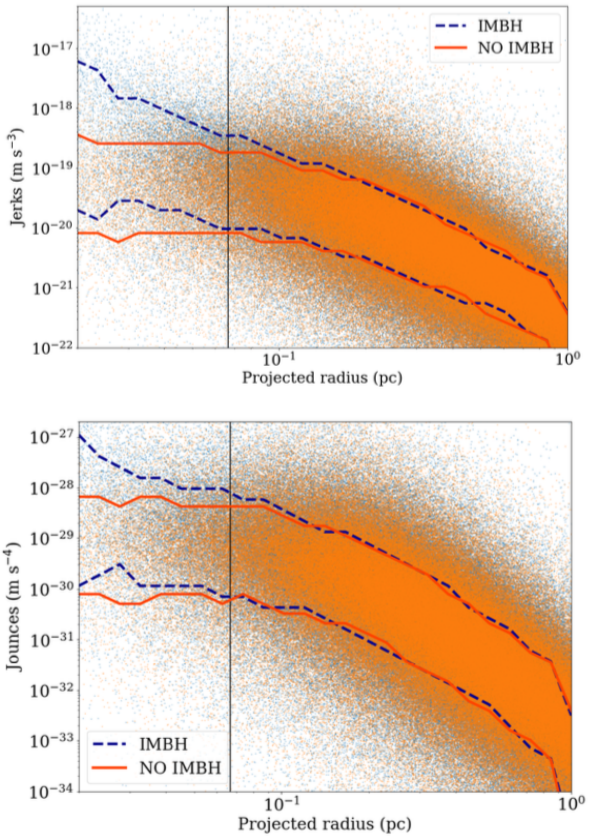


Figura 4: Profili radiali di *jerk* (in alto) e *snap* (in basso) ottenuti dalle simulazioni di Abbate et al.2019.

Tecniche di Machine Learning interpretabile: alberi decisionali

In questo Capitolo, in seguito ad una rapida introduzione generale al *Machine Learning*, viene presentata una delle tecniche di *Machine Learning interpretabile* e cioè gli alberi decisionali.
Vengono descritti i concetti fondamentali necessari per costruire le strutture ad albero, i problemi più importanti che si possono incontrare durante questa fase ed i metodi che si utilizzano per risolverli.
In particolare, viene descritto l'algoritmo CART [**brei:book**], ovvero l'algoritmo utilizzato in questo lavoro, ed i metodi su cui esso si basa.

Introduzione al Machine Learning

Il *Machine Learning, (ML)* è una branca dell'Intelligenza Artificiale e si pone l'obiettivo di far apprendere in modo automatico alle macchine attività svolte da noi esseri umani.
Più precisamente, si dice che un programma impara da una certa esperienza E rispetto ad una classe di compiti T ottenendo una *performance* P, se la sua *performance* P nel realizzare i compiti T, migliora con l'esperienza E [**mitch:book**].
In altre parole, se un programma migliora lo svolgere di un *task* rispetto a un'esperienza passata, si dice che ha imparato.

Questo avviene tramite l'apprendimento automatico. Esso può essere suddiviso in due importanti categorie: apprendimento supervisionato e apprendimento non supervisionato.

Gli algoritmi di apprendimento automatico supervisionato sono sequenze di operazioni che utilizzano i dati di allenamento (*training set*) ricevuti in input per produrre un modello capace di risolvere un problema di classificazione o di regressione su dati di test (*test set*) mai visti in precedenza, con una *performance* che aumenta in funzione della quantità di dati di allenamento ricevuti. In primo luogo, quindi, l'algoritmo lavora su un sottoinsieme del *dataset* chiamato *training set*. Una volta costruito il modello, questo poi viene utilizzato per riconoscere e analizzare dati mai visti (chiamati dati del *test set*).
Durante l'apprendimento supervisionato si hanno a disposizione sia i dati di input (X, matrice delle $features$ composta dalle variabili indipendenti che usiamo per la predizione), sia i dati di output (Y, composto dalle variabili dipendente che vogliamo predire chiamate *labels*). In questo caso si utilizza un algoritmo che apprende la funzione f che dall'input genera l'output: $Y = f(X)$.
L'obiettivo è approssimare la funzione in modo che quando si ha un nuovo dato di input l'algoritmo sia in grado di prevedere il valore di output generato per quel dato.
I problemi di apprendimento supervisionato possono essere distinti in:

- Classificazione: si tratta di un problema discreto, cioè la *label* è una variabile categorica (si/no, vero/falso, $0/1/2\ldots$). Ad esempio, in campo medico, si potrebbe voler determinare in base ai risultati quantitativi di una biopsia se un caso di tumore è benigno o maligno;

- Regressione: si tratta di un problema continuo, cioè la *label* è una variabile numerica. Ad esempio, il prezzo più probabile di una casa che si vuole prevedere sulla base dei metri quadri e della zona di interesse.

Nell'apprendimento non supervisionato, invece, si ha solo la variabile di input X e nessuna variabile di output corrispondente. Pertanto, gli algoritmi di apprendimento non supervisionato cercano di trovare una struttura nel

dataset.
In questo caso i problemi di apprendimento possono essere suddivisi in:

- Raggruppamento: anche detto *clustering*, viene utilizzato quando è necessario raggruppare i dati che presentano caratteristiche simili. Per esempio un assicuratore potrebbe voler individuare clienti che corrispondono a profili di rischio simili tra loro, sulla base di caratteristiche quali: l'età, il genere, indicatori dello stato di salute, ecc...;

- Associazione: è un problema dove si vogliono scoprire regole che descrivono grandi porzioni di dati; si ha come obiettivo quello di trovare schemi frequenti, associazioni, correlazioni o strutture casuali tra un insieme di oggetti in un *dataset* relazionale. Un'applicazione è quella del *market basket analysis*. Si tratta di analisi di transazioni commerciali che possono produrre informazioni per determinare regole ricorrenti che pongono in relazione l'acquisto di uno o più prodotti con altri. Ad esempio, se un cliente compra il latte qual è la probabilità che compri anche i cereali? Sulla base di queste informazioni si possono progettare azioni promozionali o posizionare gli articoli sugli scaffali;

- Riduzione della dimensionalità: è un problema in cui si vuole individuare un numero ridotto di *features* rappresentative delle caratteristiche dei dati all'interno di un grande campione di *features* inizialmente disponibile. Ad esempio, uno psicologo potrebbe utilizzare dati di un campione di studenti delle scuole superiori per costruire un indicatore di abilità linguistica e uno di abilità numerica combinando i voti ricevuti nelle varie materie. Questo consentirebbe la visualizzazione di una pagella come un punto nel piano definito da queste due variabili, invece di trovarsi ad affrontare il problema più complesso di visualizzare contemporaneamente i voti di tutte le materie.

In questo lavoro di tesi ho utilizzato un metodo supervisionato di classificazione: gli alberi decisionali. Qui direi perchè: sono intepretabili, che é il punto cardine della tesi$_{\text{MP}}$ Nelle prossime Sezioni, infatti, saranno approfonditi gli aspetti teorici che li riguardano.

Concetti preliminari: notazione e struttura del dataset

In questa Sezione vengono esposti alcuni concetti preliminari che riguardano gli algoritmi di classificazione, necessari per entrare nel merito dei metodi utilizzati per questo lavoro.
Il *dataset* sul quale avviene la fase di addestramento viene chiamato *training set* e il modo più semplice per descriverlo è mediante una matrice $\mathbf{X} \in \mathbb{R}^{n \times \mathbf{m}}$. Le righe della matrice sono i *records*, ovvero gli esempi o le osservazioni e le colonne sono le $features$, cioè le caratteristiche multiple aventi per ogni esempio.
Per un algoritmo di classificazione, essendo un metodo di addestramento supervisionato, il *dataset* è caratterizzato anche dal vettore Y delle $labels$. Ogni *label*, ovvero ogni etichetta, corrisponde ad una classe e sono le variabili target che si vogliono predire (tab. 1).

	feat. 1 (A_1)	feat. 2 (A_2)	.....	feat. m (A_m)	classe
record 1					
record 2					
record 3					
.....					
.....					
record n					

Tabella 1: Schema della struttura di un generico *dataset* di apprendimento nel caso in cui si stiano usando algoritmi di classificazione.

La classificazione ha come scopo quello di analizzare i dati di input sviluppando un modello in grado di predire la classe di appartenenza dei *records* in base alle *features* presenti nei dati. In genere, partendo dall'utilizzo di insiemi esistenti e già classificati, l'algoritmo cerca di identificare alcune regolarità che caratterizzano le varie classi.

Alberi decisionali: concetti generali

Gli alberi decisionali sono strutture molto conosciute nell'ambito degli algoritmi supervisionati, in quanto permettono di classificare in modo semplice degli oggetti in un numero finito di classi.

L'utilizzo degli alberi decisionali offre numerosi vantaggi:

- sono di facile interpretazione, specie se non sono molto profondi;
- ottengono una buona accuratezza su gran parte dei problemi reali di classificazione su dati tabulari;
- sono robusti rispetto al rumore e alla ridondanza tra le *features*;
- possono essere costruiti efficientemente ed essere visualizzati.

Gli alberi vengono costruiti suddividendo i *records* in sottoinsiemi in base alle relazioni che legano le variabili target, che si cercano di prevedere, alle *features* utilizzate come predittori. In particolare, questo permette di costruire un modello rappresentato da un insieme di regole ottenute ponendo una serie di domande (test). Queste sono mirate sui valori delle *features* e tipicamente consistenti in un confronto tra il valore di una data *feature* e una soglia numerica. Ogni volta che si riceve una risposta viene posta la domanda successiva in modo che sia attinente al risultato ottenuto. Il processo viene iterato fino all'ottenimento della classe di ciascun *record*. La serie di domande, e le relative risposte sono organizzate in una struttura ad albero.
Fondamentalmente si tratta di una struttura semplice composta da nodi, rami e foglie (anche dette nodi terminali) che si sviluppa a partire dal nodo radice. Ogni nodo corrisponde ad una decisione basata sul confronto di una *feature* con una costante. Essi sono collegati dai rami che identificano i livelli di parentela tra i diversi nodi (il nodo genitore rappresenta una decisione presa a monte rispetto al nodo figlio) e che forniscono gli strumenti per la costruzione di regole necessarie per classificare un oggetto. Infine, i risultati sono identificati dalle foglie. Dato un albero decisionale si possono ottenere predizioni rispetto a nuovi dati semplicemente percorrendo l'albero dalla

radice verso le foglie, seguendo di volta in volta il ramo corrispondente al risultato del confronto effettuato in ciascun nodo.

Per fare maggiore chiarezza su come si costruisce un albero decisionale e sulla sua struttura, di seguito viene riportato un semplice esempio.
Supponiamo che una compagnia di assicurazioni voglia identificare il legame che lega le classi di rischio, in cui vengono suddivisi i clienti, con la loro età anagrafica e il tipo di vettura posseduto. Lo studio si deve basare su un gruppo di clienti già classificati nella corrispettiva classe.
Il *training set* T a disposizione è rappresentato in tabella 2 e l'insieme delle classi è $\Gamma = \{A, B\}$, in cui A identifica un Alto rischio e B un Basso rischio.

Rid	Età	Tipo di Auto	Rischio
1	23	Berlina	A
2	18	Sportiva	A
3	43	Sportiva	A
4	68	Berlina	B
5	32	Furgone	B
6	20	Berlina	A

Tabella 2: *Dataset* d'esempio per classificare clienti di una compagnia di assicurazioni in opportune classi di rischio.

In figura 5 viene riportato un possibile albero decisionale per l'esempio in questione. Si noti come i nodi interni corrispondano a test sulle $features$ e come i nodi foglia vengano etichettati con la classe di maggioranza per la rispettiva foglia.

I test sulle *features*, effettuati nei nodi interni, differiscono a seconda del tipo di dati. La suddivisione del *dataset*, dovuta ad esiti diversi nei test, è definita *split*.
In questo esempio gli *split* sono binari, in quanto per ogni nodo si hanno a disposizione due possibilità.

Algoritmo CART

Esistono in letteratura diversi algoritmi di classificazione che fanno uso degli alberi decisionali. Quello che è stato utilizzato in questo lavoro è l'algoritmo

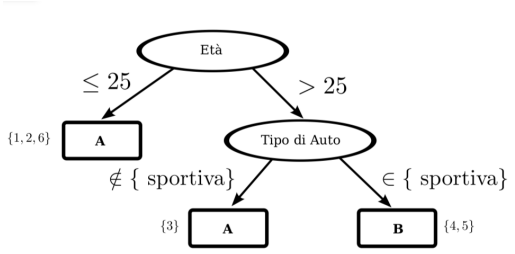


Figura 5: Esempio di albero decisionale per una compagnia di assicurazioni. Il labeling dell'ultimo nodo mi sembra sbagliato, sono quelli che hanno un'auto sportiva ($\in$ sportiva) che sono ad alto rischio nonostante siano vecchi; A e B nelle foglie andrebbero scambiati.$_{\text{MP}}$

CART.

L'algoritmo CART, *Classification and Regression Trees* [**brei:book**], è uno degli algoritmi più conosciuti ed utilizzato per lo sviluppo di alberi decisionali e può lavorare sia come un classificatore che come un regressore.

Uno dei suoi punti di forza è la sua semplicità: esso opera mediante degli *split* binari (ad ogni nodo corrispondono due soli rami) su una singola variabile in modo ricorsivo, quindi, classificare un campione può richiedere solo pochi semplici passi.
Nonostante la sua semplicità, è comunque in grado di ottenere risultati migliori di diversi altri metodi, su *dataset* complessi, non lineari e composti da molte variabili.
Questo è senz'altro un valore aggiunto al fatto che gli alberi decisionali siano di così facile ed intuitiva interpretazione, per questo motivo vengono utilizzati negli ambiti più svariati. Quelle appena discusse sono anche le motivazioni primarie che hanno spinto all'utilizzo di tale metodologia per lo sviluppo di questo progetto di tesi. Trattandosi di un *dataset* non ancora mai sottoposto ad uno studio con metodi di ML, gli alberi decisionali con l'algoritmo CART hanno fornito una prima analisi esplorativa dei dati fornendo già dei buoni risultati .

Il metodo CART è molto efficiente anche perché non richiede in ambito applicativo di sperimentare trasformazioni delle variabili indipendenti (logaritmi, radici quadrate, elevamento a potenza, ecc). Tali trasformazioni, in

quanto monotóne, non modificano i risultati a meno che lo *split* sia basato su combinazioni lineari delle variabili. Questo in genere non succede in un albero convenzionale, dove le *features* vengono valutate indipendentemente in ciascun nodo. Inoltre, esso può utilizzare la stessa variabile in punti differenti dell'albero.
Il CART è stato creato con lo scopo di trattare dati dotati di una struttura complessa. E' estremamente robusto all'effetto degli *outliers* e può utilizzare congiuntamente variabili di tipo categorico e continue.
L'algoritmo CART produce alberi ottimali, utilizzando strumenti sofisticati per stabilirne l'accuratezza (Sez.). E, infine, oltre ad essere alla base di altri algoritmi che generano alberi più complessi, esso può essere utilizzato a supporto di altri tipi di modelli.

L'algoritmo si sviluppa in due fasi:

- Fase di generazione dell'albero;
- Fase di riduzione della complessità.

Vedremo ora quali sono in generale i criteri per la costruzione di alberi decisionali e su quali in particolare il CART si basa.

Criteri per la costruzione di alberi decisionali

Per costruire una struttura ad albero efficace occorre seguire tre *step*:

- Selezionare una regola per lo *split* per ogni nodo, ciò significa determinare le variabili indipendenti ed i rispettivi valori soglia, che saranno usati per partizionare il *training set*;
- Determinare quali nodi sono terminali, quindi decidere quando "fermarsi";
- Assegnare una classe ad ogni nodo terminale.

Regole di Splitting

Definire le regole di *splitting* vuol dire identificare, per ogni nodo, una variabile sulla quale effettuare il test sulla base del rispettivo valore soglia

utilizzato per il partizionamento del *training set*.
Il punto è quello di scegliere in ogni nodo la migliore variabile di *split*, che garantisca la suddivisione dei *records* presenti nel nodo in sottogruppi il più possibili omogenei al loro interno ed eterogenei tra loro in termini dei valori delle rispettive *labels*.
Occorre, quindi, effettuare una sorta di valutazione della bontà degli *split*.
A tal proposito gli autori dell'algoritmo CART hanno sviluppato un $framework$ metodologico, introducendo il concetto generico di *impurità*.
Intuitivamente, quando dividiamo i dati, vorremmo che la regione corrispondente a ciascun nodo foglia sia "pura", ovvero che la maggior parte dei dati in questa regione provenga dalla stessa classe e quindi che ci sia una classe dominante.

Consideriamo il seguente esempio [**split:online**] mostrato in figura 6. Qui abbiamo due classi: la classe **x** e la classe **o**. Le variabili di input sono

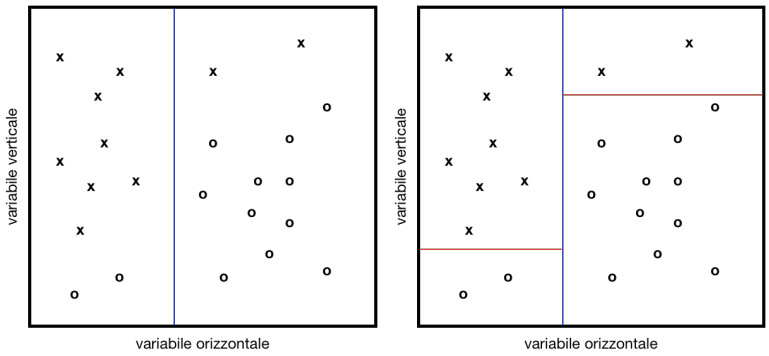


Figura 6: Un semplice esempio di *splitting*. La figura a sinistra mostra il primo *split* (linea blu), mentre quella a destra mostra anche il secondo (linea rossa).

la variabile orizzontale e quella verticale. Il primo *split* viene fatto controllando se la variabile orizzontale è al di sopra o al di sotto di una soglia (la divisione è indicata dalla linea blu).
Potremmo considerare questo *split* una buona divisione perché il lato sinistro è quasi puro in quanto la maggior parte dei punti appartiene alla classe **x** e solo due punti appartengono alla classe **o**. Il viceversa vale per il lato destro.

Proseguendo ad un livello più in basso nell'albero, vediamo che sono state create altre due divisioni (linee rosse). La regione in alto a sinistra (o il nodo foglia) contiene solo la classe **x**, così come quella in alto a destra. Invece le regioni in basso a sinistra ed in basso a destra contengono solo la classe **o**. A questo punto non è necessario effettuare altri *split* perché tutte le foglie sono pure al 100%.

La funzione di impurità

La misura di impurità può essere ricavata a partire dalla cosidetta funzione di impurità ϕ.
Essa misura l'entità della purezza di una regione contenente osservazioni appartenenti a classi possibilmente diverse.

Supponiamo che il numero di classi sia k. Quindi la funzione di impurità è definita sul set di *k-tuple* $p_1, p_2, ..., p_k$, che sono le probabilità di ogni osservazione di appartenere ad una certa classe. Vale che $p_j \in [0,1] \ \forall \ j = 1, ..., k$ e $\sum_j p_j = 1$.
Durante l'addestramento non si conoscono le reali probabilità; si utilizzano, infatti, le percentuali di osservazioni in classe 1, classe 2, classe 3 e così via, in base al set di dati di addestramento.
La funzione di impurità può essere pensata come una funzione avente valori tra 0 e 1 che fornisce una misura di quanto le osservazioni siano correttamente distribuite nelle classi.

Si definisce, per un generico nodo t, la misura di *impurità* $i(t)$ come segue:

$$i(t) = \phi(p(1 \mid t), p(2 \mid t), \ldots, p(k \mid t)) \tag{17}$$

ove $p(j \mid t)$, con $j = 1, ..., k$, è la probabilità stimata a posteriori di ottenere la classe j per un'osservazione nel nodo t.
L'impurità di un nodo è massima quando tutte le classi sono presenti nella stessa proporzione, mentre è minima quando il nodo contiene casi appartenenti ad un'unica classe. La misura di impurità viene usata per decidere quale *split* fare in un dato nodo, determinando in sostanza in che modo si fa crescere l'albero.

Misure di impurità più comuni sono:

- L'indice di eterogeneità di Gini: $\text{Gini}(t) = 1 - \sum_j p_j^2(t)$
- L'entropia: $H(t) = -\sum_j p_j \log_2 p_j(t)$
- Il tasso di errata classificazione: $\text{r}(t) = 1 - \max\{p_j(t) : 1 \leq j \leq k\}$

Generalmente, in particolare negli algoritmi che lavorano con *splitting* binari come CART, viene utilizzato l'indice di Gini.
Tale misura può essere interpretata come la stima della probabilità che un'osservazione scelta casualmente nel nodo t sia assegnata alla classe errata.
Essa tende a dare luogo a suddivisioni bilanciate dal punto di vista del numero di casi inviati dallo *split* nei due nodi figli, ovvero tende ad evitare i cosiddetti *small splits*.

Dichiarazione dei nodi terminali

A questo punto, per completare la costruzione dell'albero è necessario passare per gli ultimi due *step*.
Bisogna capire, in primo luogo, quando arrestare la "crescita" dell'albero e quindi identificare i nodi terminali. Infine è necessario assegnare una classe ad ogni nodo terminale.

Criteri di arresto

Nella maggior parte dei casi, se si lasciasse sviluppare un albero fino alle foglie estreme, si avrebbe una struttura molto grande e caratterizzata dal concatenarsi di numerose condizioni. Così facendo, il vantaggio della semplicità interpretativa, caratteristica degli alberi decisionali, verrebbe perso e si andrebbe incontro al problema dell'*overfitting* (approfondito nella Sezione).
La dimensione eccessiva sarebbe dovuta al numero di nodi terminali o, equivalentemente, al numero di suddivisioni che essi rappresentano. Per questa ragione, per ovviare al problema, si può decidere di interrompere l'espansione dell'albero sulla base di determinati criteri.
Alcuni criteri comuni sono:

- continuare la crescita finché si raggiunge una certa profondità predefinita;

- continuare finché il numero di osservazioni in ciascun nodo terminale non superi una certa soglia;

- continuare finché tutti i nodi terminali non sono puri, cioè contengono solo una classe Questo è un po' come non fermarsi però$_{\text{MP}}$.

Queste soglie vanno decise a priori e influenzano direttamente la dimensione dell'albero. Per questo motivo, sebbene sia questo il metodo più semplice, risulta piuttosto inefficiente perché rischia di sfoltire l'albero troppo o troppo poco rispetto al necessario, in base a quanto sia restrittivo il criterio d'arresto.

Assegnazione della classe al nodo terminale

Nella fase di assegnazione di una delle classi ai nodi terminali si possono presentare tre situazioni diverse:

- il nodo contiene solo osservazioni appartenenti alla stessa classe;

- il nodo contiene osservazioni appartenenti a classi diverse, ma una di queste presenta una proporzione di osservazioni maggiore delle altre;

- Il nodo contiene osservazioni appartenenti a classi diverse e nella stessa proporzione.

Nel primo caso l'assegnazione viene effettuata precisamente senza alcun tipo di indecisione e nel secondo caso il nodo viene assegnato alla classe che presenta più osservazioni. Nell'ultimo caso, invece, si presenta una situazione di massima incertezza, in quanto la probabilità delle classi risulta identica per ciascuna di esse. Pertanto si ricorre ad un tipo di assegnazione casuale salvo l'intervento del ricercatore che può effettuare un'assegnazione diversa sulla base delle sue conoscenze.

Problema dell'overfitting

Alla fase di costruzione dell'albero decisionale, per far sì che esso si comporti come un buon classificatore, segue la fase di riduzione della complessità dell'albero. Ma prima di approfondire la descrizione che riguarda questa seconda fase, bisogna introdurre un problema con il quale, quasi sempre, tutti gli algoritmi di ML si confrontano: il problema dell'*overfitting*.

Il metodo di classificazione è un processo che si sviluppa in due fasi: la fase di apprendimento e la fase di predizione.
Nella fase di apprendimento, il modello viene sviluppato sulla base di dati di allenamento conosciuti, che abbiamo chiamato *training set*.
Successivamente, la fase di predizione viene eseguita su un insieme di dati totalmente nuovi per il modello, il cosiddetto *test set*.
In pratica, quindi, il modello apprende informazioni, relazioni e collegamenti durante la prima fase e poi applica tutto ciò che ha acquisito su un nuovo insieme di dati per fornire una predizione.
Quello che normalmente accade dipende... se si allena un modello rigido, per es. una regressione lineare, questo non succede$_{\text{MP}}$ nelle prime prove di addestramento è che il modello porta a delle prestazioni molto buone sul *trainig set* e alquanto scadenti sul *test set*. Vuol dire che il modello 'impara troppo bene' gli schemi dal *training set*, così tanto da non essere poi in grado di generalizzare quanto appreso su un nuovo gruppo di dati. Più che 'imparare troppo bene', il modello impara caratteristiche del training set che sono proprie solo del training set e non si applicano altrove. Per esempio nel training set, per puro caso, tutti i fumatori erano pisani. Nella realtà anche persone di altre città fumano con la stessa probabilità e quindi la città di provenienza è pressoché inutile per prevedere se una persona fuma o meno. Ma, specialmente in training set piccoli, correlazioni spurie di questo tipo possono verificarsi abbastanza di frequente.$_{\text{MP}}$

Per 'generalizzazione', quindi, si intende l'abilità di una macchina di portare a termine in maniera accurata esempi o compiti nuovi, che non ha mai affrontato, dopo aver fatto esperienza su un insieme di dati di apprendimento [**generalizz:online**].
Questo problema è noto con il termine *overfitting*, ovvero si tratta di un

problema di sovradattamento da parte del modello ai dati *trainig set*. Si verifica quando il modello viene addestrato eccessivamente su un set di dati rumorosi.
Pertanto, a causa di questo problema, quando si sviluppa un albero decisionale, è importante capire quando fermarsi. Se un albero crescesse troppo, fino alle foglie più estreme, oltre a perdere la sua importante caratteristica di essere interpretabile, si andrebbe incontro al problema dell'*overfitting*. Questo è evidente se, ad esempio, avessimo un albero con una sola osservazione per foglia, raggiungendo quindi una purezza perfetta in ciascuna foglia sul *training set*. Se si estendesse al massimo un albero, infatti, tale obbiettivo si potrebbe sempre raggiungere per qualunque scelta dei dati di *training*, ma negli ultimi *split* l'albero non avrebbe imparato alcuna informazione utile dai dati, adattandosi solo alle idiosincrasie del *training set*. Pertanto, risulterebbe probabile che la *performance* predittiva su dati non visti sia molto inferiore, nonostante la predizione nominalmente perfetta in *training*.

Una tecnica per risolvere tale problema per gli alberi decisionali è la tecnica del *pruning*. In breve, essa consiste nell'ottenere da un albero il più piccolo "sottoalbero" che, di fatto, non comprometta l'accuratezza della classificazione. In particolare, nell'algoritmo CART, la tecnica utilizzata è quella del *Cost-Complexity Pruning* che verrà approfondita nella prossima Sezione.

Cost-Complexity Pruning

Per ridurre la complessità dell'albero affinchè possa essere effettivamente utile nel classificare *records* e nel fornire regole efficaci, è necessario sfoltire la ridondanza dell'albero, ovvero eliminanare i rami meno significativi.

La tecnica che l'algoritmo CART utilizza prende il nome di *Cost-Complexity Pruning*. Consiste nel far sviluppare l'intero albero e poi estarre da esso il più piccolo "sottoalbero". Quest'ultimo porterà a delle migliori previsioni perché vi sono stati "tagliati" i rami corrispondenti a *splitting* eccessivamente specifici sul *training set* e, quindi, superflui. Infatti *pruning* sta proprio per "potatura".

Questo processo, appunto, non peggiora l'accuratezza delle previsioni, ma bensì, in genere, le migliora, in quanto, poi, il modello sarà capace di classificare efficientemente anche dati totalmente nuovi.

Notazione e metodo

Ora viene introdotta la notazione necessaria per la trattazione della tecnica del *Cost-Complexity Pruning*.
Si definiscono:

- T_{max} dimensione di massima crescita dell'albero T;
- t i nodi genitori;
- t' i nodi figli, cioè un nodo collegato tramite un percorso al nodo t;
- T_t un ramo dell'albero T con nodo radice t.

Un metodo di potatura efficiente dovrebbe garantire che la ricerca della sottostruttura ottimale possa essere trattabile computazionalmente.

L'algoritmo del *Cost-Complexity Pruning* è parametrizzato dal parametro $\alpha \in \mathbb{R}$, $\alpha \in [0,1]$, noto come *parametro di complessità*. Questo parametro viene utilizzato per definire la misura *cost-complexity* $R_\alpha(T)$ per ogni sottostruttura $T < T_{max}$ come:

$$R_\alpha(T) = R(T) + \alpha|\tilde{T}| \tag{18}$$

dove $R(T)$ è il tasso di errata classificazione totale dei nodi terminali (definito) e $|\tilde{T}|$ è il numero di nodi terminali (o nodi foglia) che definisce la complessità dell'albero T. Infatti, maggiore è il numero di nodi foglia che l'albero contiene, maggiore è la complessità dell'albero perché vi è maggiore flessibilità nel partizionare lo spazio in parti più piccole e quindi maggiori possibilità di adattare i dati di addestramento.
Durante la potatura dell'abero bisogna minimizzare la funzione $R_\alpha(T)$ e la sottostruttura che alla fine verrà selezionata dipende dal parametro α. Infatti, se $\alpha = 0$ verrà scelto l'albero più grande perché il termine di complessità viene essenzialmente eliminato. Per α tendente all'infinito, invece, verrà selezionato l'albero di dimensione 1, cioè un singolo nodo radice.

Si può dimostrare [**split:online**] che per ogni α esiste ed è unica una sottostruttura che minimizza $R_\alpha(T)$. Inoltre, si può dimostrare anche che le sottostrutture che minimizzano $R_\alpha(T)$ al crescere di α sono annidate. Ovvero, il sottoalbero dell'α successivo è compreso in quello precedente e, quindi, vale che $T_1 > T_2 > \cdots > t_1$, dove i pedici rappresentano il numero progressivo degli α. Praticamente, a partire dalle foglie l'algoritmo procede spostandosi verso la radice valutando in ogni nodo la *cost-complexity* al variare di α.

Per fare maggiore chiarezza sul metodo di taglio, in primo luogo, è necessario estendere la definizione 18 a un nodo e ad un singolo ramo fuoriuscente dal nodo:

- per qualsiasi nodo t appartenente ad una sottostruttura la 18 diventa:

$$R_\alpha(t) = R(t) + \alpha$$

  in quanto ci si sta riferendo proprio ad un nodo, in questo caso non c'è il termine di complessità $|\tilde{T}|$ non è che non ci sia, ma è uguale a 1_{MP};

- per qualsiasi ramo T_t, invece la 18 diventa:

$$R_\alpha\left(T_t\right) = R\left(T_t\right) + \alpha\left|\tilde{T}_t\right|$$

  in cui il termine $R\left(T_t\right)$ è calcolato considerando tutti i nodi terminali che discendono dal nodo t attraverso il ramo T_t (Fig. 7).

In ogni nodo, facendo variare α tra 0 e 1 in maniera crescente, l'algoritmo confronta il *cost-complexity* calcolato nel nodo t con quello calcolato per il ramo T_t. Affinchè l'algoritmo decida di non tagliare un ramo, lo *split* che avviene per mezzo di esso, deve essere efficiente. Questo si verifica quando lo *split* riduce l'errore sulla classificazione. Pertanto, finchè $R_\alpha\left(T_t\right) < R_\alpha(t)$ il ramo viene tenuto, ma nonappena i due valori si eguagliano esso viene tagliato. Per cui, la condizione di taglio è data da $R_\alpha\left(T_t\right) = R_\alpha(t)$, da cui:

$$\alpha = \frac{R(t) - R\left(T_t\right)}{\left|\tilde{T}_t\right| - 1} \tag{19}$$

L'algoritmo lavora percorrendo a ritroso l'albero, dalle foglie verso la radice. Inizialmente, quindi, considera tutti i nodi che precedono le foglie e, facendo

variare α, il primo nodo trovato che soddisfa la condizione 19 viene tagliato. A questo punto, l'algoritmo salva in memoria l'α appena ricavato e poi procede al passo successivo considerando l'albero appena ottenuto come albero iniziale.

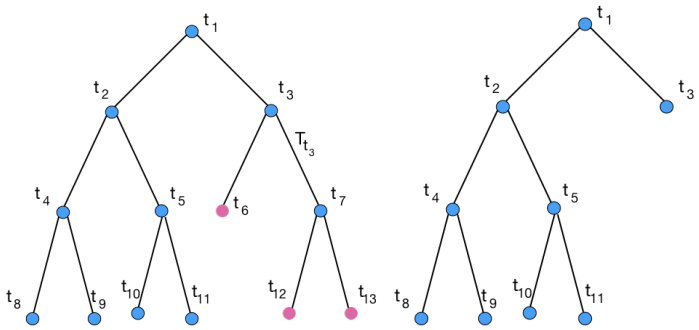


Figura 7: Un semplice schema di una struttura ad albero. I nodi terminali del ramo T_{t_3} sono t_6, t_{12} e t_{13} (a sinistra). In t_3 si verifica la condizione 19, per cui l'intero ramo T_{t_3} viene tagliato (a destra).

nella sezione appena conclusa sottolineerei che α è un parametro che l'utente sceglie a mano e che quindi va determinato in base a conoscenze specifiche legate al problema oppure sperimentalmente, tramite validation. Così si transisce in modo naturale alla prossima sezione.$_{\text{MP}}$

Training, validation e test sets

Per identificare i parametri da fornire ai modelli di ML è di uso comune suddividere il *dataset* in tre sottogruppi: *training set, validation set* e *test set*.

In questo caso particolare, per effettuare un *pruning* efficiente, dal punto di vista pratico, è necessario identificare il parametro di complessità α al quale corrisponderebbe la miglior sottostruttura dell'albero T. Infatti, come spiegato nel paragrafo precedente, ad ogni passo l'algoritmo *Cost-Complexity Pruning* salva in memoria il parametro α che soddisfa la relazione 19 con il relativo modello ad albero. Sarà l'utente, in una fase successiva, a scegliere tra questi la miglior sottostruttura, cioè quella per cui si ottiene la miglior

accuratezza nella classificazione. Per questo motivo si divide il *dataset* in tre sottogruppi.

Il *training set* è l'insieme di dati con i *records* e le relative *labels* e serve al classificatore durante la fasse di allenamento per apprendere gli schemi che possono essere utilizzati successivamente per prevedere le *labels* dei nuovi dati. In questa fase è importante che non venga effettuata nessuna scelta sul modello.

Il *validation set*, invece, è utile per determinare con quali parametri l'algoritmo di classifcazione avrebbe le migliori prestazioni. Nel caso degli alberi decisionali la fase di validazione è necessaria per scegliere il miglior parametro di complessità α.

Una volta ottenuto il miglior classificatore bisogna valutare le sue prestazioni su un set di dati completamente nuovo per lui: il *test set*.

La scelta su come effettuare la divisione del *dataset* è individuale, ma generalmente il *test set* rappresenta circa il $20 - 30\%$ dei dati. La restante parte del *dataset* viene suddiviso nel seguente modo: circa il $70-80\%$ dei dati rimasti costituirà il *training set* e circa il $20 - 30\%$ farà parte del *validation set*.

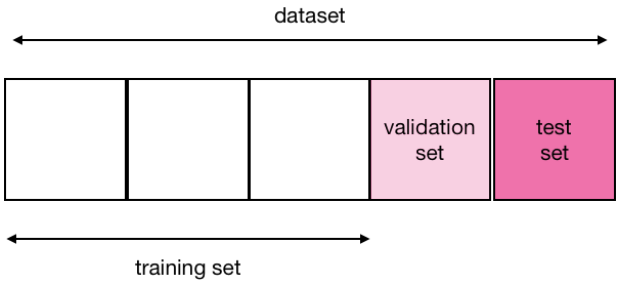


Figura 8: Schema per visualizzare la divisione dei dati nei tre sottogruppi: *training set, validation set* e *test set*.

Qualità delle previsioni

Per qualsiasi algoritmo di ML, dopo aver ottenuto le previsioni, bisogna valutarne la qualità. Per quantificare la bontà delle *performance* possono

essere utilizzate diverse metriche.
In questo lavoro, per valutare le prestazioni della classificazione da parte degli alberi decisionali, in primo luogo, è stata utilizzata l'*accuratezza* e, successivamente, altre metriche più adatte in che senso? Avevamo notato che la sola acciratezza su un dataset sbilanciato dava risultati pessimi, ma poi hai bilanciato il dataset... direi semplicemente altre metriche, che misurano aspetti specifici delle prestazioni di classificazione$_{\text{MP}}$, come *precisione, richiamo* ed *F-score (o F-1)*. Peraltro poi queste metriche vanno spiegate bene, se non qui nel seguito (nel qual caso indicherei che esse sono spiegate nel seguito).$_{\text{MP}}$

Accuratezza degli alberi decisionali

Per definire l'accuratezza di un albero decisionale, in primo luogo, introduciamo una notazione. Indichiamo con:

- $\bar{n}_t$ il numero totale dei *records* del *test set* che finiscono nel nodo terminale t;
- n_t il numero di *records* classificati correttamente in t.

Ad ogni nodo terminale t è possibile associare un'accuratezza a_t data dalla frazione dei *records* classificati correttamente nella data foglia t:

$$a_t = \frac{n_t}{\bar{n}_t} \tag{20}$$

L'accuratezza nella classificazione associato all'intero albero è data dalla somma delle accuratezze calcolate su tutti i t, pesate rispetto alla probabilità che un *record* possa finire su ciascuna foglia.
Se il numero totale di *records* è N , allora, la probabilità che un certo *record* finisca nella foglia t è data da $\frac{\bar{n}_i}{N}$. Pertanto l'accuratezza A dell'albero risulta:

$$A = \sum_t \frac{\bar{n}_t}{N} a_t \tag{21}$$

La situazione ideale si ha quando $A = 1$, in cui il modello è in grado di classificare perfettamente tutti i *records*.
Questa metrica di solito lavora bene su un set di dati bilanciato, ovvero

quando si ha un numero simile di campioni appartenenti a ogni classe.
L'accuratezza, però, non è il miglior metodo per valutare le prestazioni di un classificatore.

Matrice di confusione: precisione, richiamo e F-score

Uno strumento intuitivo per valutare le *performance* di un modello di classificazione è la *matrice di confusione*. Per analizzare le informazioni che essa fornisce bisogna definire le metriche che utilizza e cioè la *precisione*, il *richiamo* e la *F-score (o F-1)*.

Supponiamo di avere due classi: la classe P dei positivi e la classe N dei negativi.
Allora, le previsioni ottenute da un modello su un *test set* possono essere definite nei seguenti modi:

- **Veri Positivi** (TP), sono i dati test realmente$_{\text{MP}}$ etichettati come P e che il modello ha previsto correttamente come P;
- **Falsi Positivi** (FP), sono i dati test etichettati come N, ma per i quali il modello ha predetto essere P;
- **Falsi Negativi** (FN), sono i dati test appartenenti alla classe P, ma per cui il modello ha predetto come N;
- **Veri Negativi** (TN), sono i dati test della classe N per i quali il modello ha predetto correttamente collocandoli nella classe N.

A partire da queste definizioni è possibile introdurre le metriche:

- **Precisione**, $P = \frac{\text{TP}}{TP+FP}$ il comando mathrm ti permette di far apparire il testo come testo normale dentro ad una espressione matematica; forse puoi provare a usarlo se ti piace il risultato$_{\text{MP}}$, è il numero di TP diviso per il numero di classificazioni positive, cioè TP e FP darei degli esempi: il numero di persone veramente malate su quelle che un dato test ritiene malate$_{\text{MP}}$;
- **Richiamo**, $R = \frac{TP}{TP+FN}$, è il numero di TP diviso per il numero di tutti i *records* positivi, cioè TP e FN il numero di persone malate individuate da un test sul totale delle persone veramente malate$_{\text{MP}}$;

- **F-score**, $F_1 = 2 \cdot \frac{P \cdot R}{P+R}$, è una media armonica tra precisione e richiamo indicherei qui che la media armonica è più vicina al minimo di quanto lo sia la media aritmetica, quindi per ottenere una buona F score bisogna avere sia un buon richiamo che una buona precisione (come dici dopo)$_{\text{MP}}$.

Un valore vicino a 1 della *F-score* segnala la capacità del modello di classificare correttamente la maggior parte dei dati, ottenendo sia una buona precisione che un buon richiamo (Fig. 9). D'altra parte ottenere un buon richiamo con una pessima precisione è semplice: basta dichiarare tutti malati, riducendo così a zero il numero di falsi negativi al costo di avere molti falsi positivi. Ma il classificatore che ne risulta è inutile. Per questo il richiamo da solo non è una buona metrica. Per ragioni analoghe (potrei dichiarare che nessuno è malato, portando i falsi positivi a zero assieme ai veri positivi) la precisione da sola è a sua volta una metrica non buona. Combinandole, la F-score ci garantisce una misura più bilanciata tra questi due aspetti.$_{\text{MP}}$

Un metodo molto utile per visualizzare queste metriche è la matrice di confusione. Si tratta di uno strumento che mostra la frequenza di errori di classificazione e la corretta classificazione dei dati.
Nelle celle di una matrice di confusione sono riportati valori numerici per indicare i vari casi (TP, FP, TN e FN). Pertanto la precisione, il richiamo e, di conseguenza, l' *F-score*, possono essere calcolati direttamente dai valori delle celle di una matrice di confusione.
In figura 10 è riportato uno schema d'esempio di matrice di confusione per un problema di classificazione con due classi.

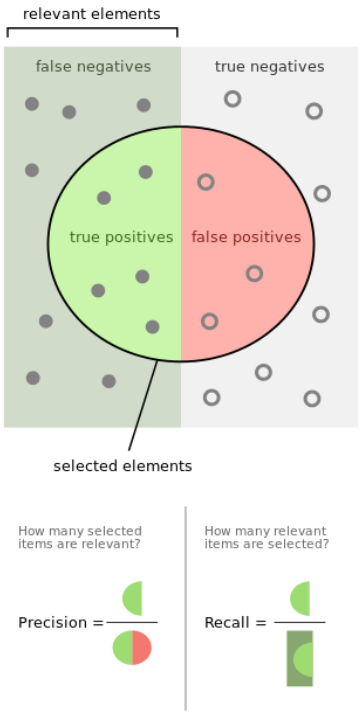


Figura 9: Schematizzazione visiva di TP, FP, TN, FN e delle metriche di precisione e richiamo [**prec_ric:online**].

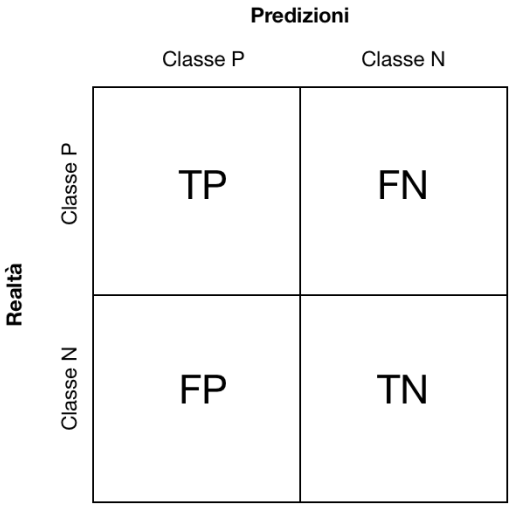


Figura 10: Struttura della matrice di confusione nel caso di un problema di classificazione con due classi: P e N.

Alberi decisionali per l'identificazione di IMBHs

Dopo aver introdotto gli strumenti teorici necessari, in questo Capitolo viene descritta nello specifico l'applicazione di tali concetti al problema astrofisico in questione: l'identificazione di IMBHs al centro di GCs. In particolare, in questo lavoro di tesi, si vuole predire la presenza di IMBHs al centro dei GCs utilizzando un metodo di ML interpretabile sulla base delle caratteristiche dinamiche delle MSPs presenti nelle regioni centrali degli ammassi. I modelli di ML utilizzati sono gli alberi decisionali costruiti con l'algoritmo CART (Cap.).

In primo luogo verrà brevemente descritto il codice HiGPUs utilizzato per le simulazioni a N-corpi da cui provengono i *dataset* forniti all'albero decisionale. Successivamente verranno, quindi, presentati i *dataset* e poi si passerà ad una descrizione generale del codice sviluppato per la predizione.

Motivazioni

Nel Capitolo si è parlato dell'importanza di identificare IMBHs al centro di ammassi e delle problematiche che si riscontrano utilizzando i metodi diretti. Abbiamo visto che esse sono legate principalmente alle difficoltà osservative, in quanto misurare le dispersioni di velocità all'interno della sfera di influenza del GC richiede elevatissime risoluzioni spaziali. Inoltre, l'identificazione di una cuspide nei profili di densità e di dispersione di velocità non sempre è indice della presenza di un IMBH. Per lo sviluppo della tesi, invece, ci si è basati su un metodo di identificazione indiretto, ovvero quello delle MSPs

[**abbate1:paper**]. Dallo studio delle informazioni dinamiche delle MSPs all'interno dei GCs, infatti, è possibile identificare indirettamente la presenza di IMBHs in ammassi.

In questo lavoro di tesi, però, si vuole applicare un nuovo metodo in questo contesto, ovvero servirsi di un algoritmo di ML interpretabile (gli alberi decisionali) per prevedere la presenza di IMBHs al centro di GCs. La sfida, infatti, sta proprio nel voler mettere a punto un metodo identificativo nuovo per cercare di avanzare nella risoluzione di un problema astrofisico ancora così aperto. Le informazioni delle MSPs che si vogliono fornire all'algoritmo per la predizione di IMBHs sono l'accelerazione e le sue derivate rispetto al tempo: *jerks* e *snaps (o jounce)*. Queste quantità sono legate alle derivate dei periodi delle MSPs nell'ammasso e, di conseguenza, sarebbe possibile stimarle a partire da tali misure. Le campagne osservative, tuttavia, richiedono tempistiche di diversi anni per ottenere *jerks* e *snaps* con una precisione adeguata, oltre a richiedere una certa regolarità e puntualità nelle osservazioni. Pertanto, i dati osservativi necessari non sono ancora a disposizione e, per questo lavoro, ci si è serviti dei dati provenienti da un set di simulazioni a N-corpi. Inoltre, avere dati di simulazioni in tal senso è di fondamentale importanza per un futuro confronto tra dati osservati e dati teorici. NB che il training va comunque fatto su simulazioni perché anche avendo dati di jerk e snap in ammassi reali non avremmo le etichette, visto che non sappiamo quali ammassi reali siano IMBH host e quali non lo siano.$_{\text{MP}}$

HiGPUs

Le simulazioni a N-corpi da cui derivano i dati forniti all'algoritmo di ML per la previsione di IMBHs nei GCs sono state ottenute mediante la nuova versione del codice *HermiteIntegratorGPUs (HiGPUs)* [**capdolspe:paper**]. Si tratta di un codice ad alta precisione, adatto a simulare l'evoluzione nel tempo di sistemi di masse puntiformi che interagiscono tramite la forza newtoniana classica.

Il codice HiGPUs (disponibile gratuitamente in [**HiGPUs:online**]) implementa un metodo di integrazione di Hermite fino al 6°ordine e utilizza

accelerazioni, *jerks* e *snaps* per far avanzare le posizioni e le velocità delle stelle nel tempo.

Lo schema che segue è diviso in tre fasi: predizione, valutazione e correzione. Di seguito verranno indicate le derivate dell'accelerazione rispetto al tempo con la notazione con i punti e chiameremo indistintamente gli elementi *stelle* o *particelle*. Consideriamo, quindi, un sistema composto da N stelle e assumiamo che l'*i-esima* particella al tempo iniziale $t_{c,0}$ sia caratterizzata da posizione $\mathbf{r}_{i,0}$, velocità $\mathbf{v}_{i,0}$, accelerazione $\mathbf{a}_{i,0}$, *jerk* $\dot{\mathbf{a}}_{i,0}$, *snap* $\ddot{\mathbf{a}}_{i,0}$, *crackle* $\dddot{\mathbf{a}}_{i,0}$ ed un *time step* $\Delta t_{i,0}$. Inoltre, chiamiamo m il numero di particelle appartenenti allo stesso blocco temporale e che devono essere evolute allo stesso tempo $t_{c,0} + \Delta t_{i,0}$. I passi da seguire sono, dunque, i seguenti:

- *Predizione*: consiste nel calcolo di posizione, velocità e accelerazione di tutte le stelle a partire dai valori iniziali:

$$\begin{aligned}
\mathbf{r}_{i,pred} &= \mathbf{r}_{i,0} + \mathbf{v}_{i,0}\Delta t_{i,0} + \frac{1}{2}\mathbf{a}_{i,0}\Delta t_{i,0}^2 + \frac{1}{6}\dot{\mathbf{a}}_{i,0}\Delta t_{i,0}^3 + \\
&\quad + \frac{1}{24}\ddot{\mathbf{a}}_{i,0}\Delta t_{i,0}^4 + \frac{1}{120}\dddot{\mathbf{a}}_{i,0}\Delta t_{i,0}^5 \\
\mathbf{v}_{i,pred} &= \mathbf{v}_{i,0} + \mathbf{a}_{i,0}\Delta t_{i,0} + \frac{1}{2}\dot{\mathbf{a}}_{i,0}\Delta t_{i,0}^2 + \frac{1}{6}\ddot{\mathbf{a}}_{i,0}\Delta t_{i,0}^3 + \\
&\quad + \frac{1}{24}\dddot{\mathbf{a}}_{i,0}\Delta t_{i,0}^4 \\
\mathbf{a}_{i,pred} &= \mathbf{a}_{i,0} + \dot{\mathbf{a}}_{i,0}\Delta t_{i,0} + \frac{1}{2}\ddot{\mathbf{a}}_{i,0}\Delta t_{i,0}^2 + \frac{1}{6}\dddot{\mathbf{a}}_{i,0}\Delta t_{i,0}^3
\end{aligned} \tag{22}$$

- *Valutazione*: calcolo dell'accelerazione e delle sue derivate prima e seconda per tutte le particelle $m < N$ sulla base delle posizioni e velocità predette. Le mutue interazioni tra l'*i-esima* particella e le $N-1$ rimanenti sono descritte dalle seguenti relazioni:

$$\begin{aligned}
\mathbf{a}_{i,1} &= \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{a}_{ij,1} = \sum_{\substack{j=1 \\ j\neq i}}^{N} m_j \frac{\mathbf{r}_{ij}}{r_{ij}^3} \\
\dot{\mathbf{a}}_{i,1} &= \sum_{\substack{j=1 \\ j\neq i}}^{N} \dot{\mathbf{a}}_{ij,1} = \sum_{\substack{j=1 \\ j\neq i}}^{N} \left(m_j \frac{\mathbf{v}_{ij}}{r_{ij}^3} - 3\alpha_{ij}\mathbf{a}_{ij,1} \right) \\
\ddot{\mathbf{a}}_{i,1} &= \sum_{\substack{j=1 \\ j\neq i}}^{N} \ddot{\mathbf{a}}_{ij,1} = \sum_{\substack{j=1 \\ j\neq i}}^{N} \left(m_j \frac{\mathbf{a}_{ij}}{r_{ij}^3} - 6\alpha\dot{\mathbf{a}}_{ij,1} - 3\beta_{ij}\mathbf{a}_{ij,1} \right)
\end{aligned} \tag{23}$$

dove $\mathbf{r}_{ij} \equiv \mathbf{r}_{j,pred} - \mathbf{r}_{i,\mathrm{pred}}, \mathbf{v}_{ij} \equiv \mathbf{v}_{j,\mathrm{pred}} - \mathbf{v}_{i,\mathrm{pred}}, \mathbf{a}_{ij} \equiv \mathbf{a}_{j,\mathrm{pred}} - \mathbf{a}_{i,\mathrm{pred}}$, $\alpha_{ij} r_{ij}^2 \equiv \mathbf{r}_{ij} \cdot \mathbf{v}_{ij}, \beta_{ij} r_{ij}^2 \equiv v_{ij}^2 + \mathbf{r}_{ij} \cdot \mathbf{a}_{ij} + \alpha_{ij}^2 r_{ij}^2$

- *Correzione*: le posizioni e le velocità delle m particelle vengono corrette per mezzo delle accelerazioni e relative derivate secondo le relazioni:

$$\begin{aligned} \mathbf{v}_{i,\mathrm{corr}} &= \mathbf{v}_{i,0} + \frac{\Delta t_{i,0}}{2}\left(\mathbf{a}_{i,1} + \mathbf{a}_{i,0}\right) - \frac{\Delta t_{i,0}^2}{10}\left(\dot{\mathbf{a}}_{i,1} - \dot{\mathbf{a}}_{i,0}\right) + \\ &+ \frac{\Delta t_{i,0}^3}{120}\left(\ddot{\mathbf{a}}_{i,1} + \ddot{\mathbf{a}}_{i,0}\right) \\ \mathbf{r}_{i,corr} &= \mathbf{r}_{i,0} + \frac{\Delta t_{i,0}}{2}\left(\mathbf{v}_{i,\mathrm{corr}} + \mathbf{v}_{i,0}\right) - \frac{\Delta t_{i,0}^2}{10}\left(\mathbf{a}_{i,1} - \mathbf{a}_{i,0}\right) + \\ &+ \frac{\Delta t_{i,0}^3}{120}\left(\dot{\mathbf{a}}_{i,1} + \dot{\mathbf{a}}_{i,0}\right) \end{aligned} \tag{24}$$

HiGPUs è un codice ad alta precisione perché oltre ad utilizzare l'integrazione di Hermite fino al 6° ordine, calcola sempre le interazioni mutue tra tutte le stelle e definisce un *time step* variabile per ciascuna stella sulla base delle loro accelerazioni. Questo garantisce un buon compromesso tra precisione e velocità di calcolo.

Tuttavia, utilizzando intervalli temporali differenti durante l'integrazione, nel caso di incontri ravvicinati tra stelle, potrebbe accadere che l'intervallo temporale diventi così piccolo da bloccare l'integrazione come è il caso in cui si voglia simulare un ammasso stellare per la tipica vita media di uno di questi oggetti, dell'ordine dell'età dell'universo in cui si formano binarie dinamicamente importanti con periodi dell'ordine di un anno$_{\mathrm{MP}}$. Per ovviare al problema si può utilizzare il *softening* ϵ: una costante che fa in modo che le forze di interazione ed i rispettivi potenziali vengano smorzati per piccoli $r_{i,j}$. In questo modo, il potenziale di interazione tra due particelle di massa m_i ed m_j diventerà:

$$U_{ij} = \frac{G m_i m_j}{\sqrt{r_{ij}^2 + \epsilon^2}} \tag{25}$$

Simulazioni e condizioni iniziali

Le simulazioni da cui provengono i *dataset* utilizzati, sono state effettute sul supercomputer del CINECA [**cineca:online**] mediante l'utilizzo del codice

HiGPUs.

E' stato fatto evolvere un set di ammassi stellari; nello specifico sono state utilizzate circa 200 196$_{\text{MP}}$ simulazioni. Ogni ammasso è composto da un numero di stelle variabile fino a $N = 100000$ ed è rappresentato da una funzione di distribuzione di Plummer [**plummer:paper**]. Si tratta del modello di ammasso stellare più semplice che si possa avere Dire così è un po' eccessivo, ce ne sono altri come la sfera isoterma$_{\text{MP}}$. Esso è simmetrico nello spazio delle fasi ed è caratterizzato dalla distribuzione di densità:

$$\rho(r) = \frac{3M}{4\pi a^3}\left(1 + \frac{r^2}{a^2}\right)^{-\frac{5}{2}} \tag{26}$$

e potenziale corrispondente:

$$\Phi_P(r) = -\frac{GM}{\sqrt{r^2 + a^2}} \tag{27}$$

con G costante di gravitazione universale, M massa totale dell'ammasso, r la cordinata radiale ed a lunghezza di scala (**parametri usati?**).

Le singole particelle sono state generate seguendo una funzione di massa iniziale (*Initial Mass Function, IMF*) di Kroupa 2001 [**kroupa:paper**] definita come:

$$N(m)dm \propto m^{-\alpha}dm \tag{28}$$

dove

$$\alpha = \begin{cases} 0.3 & \text{se } m < 0.08M_\odot \\ 1.3 & \text{se } 0.08 < m < 0.5M_\odot \\ 2.3 & \text{se } m > 0.5M_\odot \end{cases} \tag{29}$$

Le masse delle singole stelle variano tra un minimo fissato a $0.1M_\odot$ ed un massimo variabile nell'intervallo $10 - 100M_\odot$ a causa degli effetti dovuti alla presenza di un IMBH. Anche la massa centrale dell'IMBH è stata fatta variare, in questo caso tra 0 e $10^4 M_\odot$. Inoltre, ogni sistema è stato simulato fino ad un tempo variabile tra **?** e **?**. Infine, le simulazioni non includono nè binarie primordiali nè evoluzione stellare.

Di seguito vengono mostrati i risultati per uno degli ammassi simulati con queste condizioni iniziali.

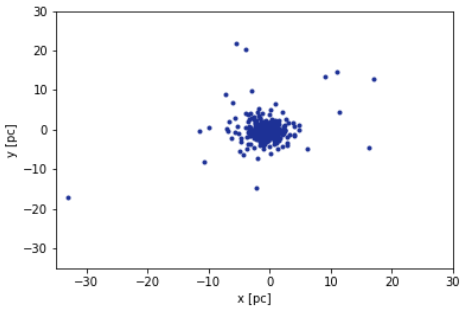
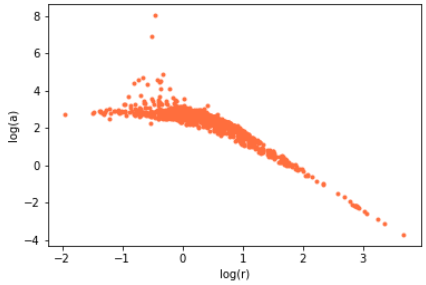
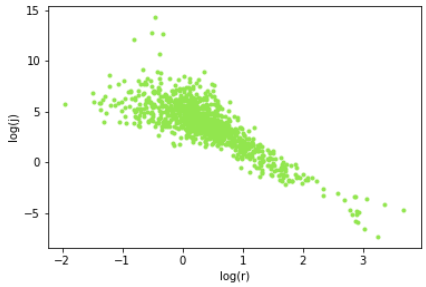


Figura 11: Proiezione di un ammasso campione sul piano del cielo.

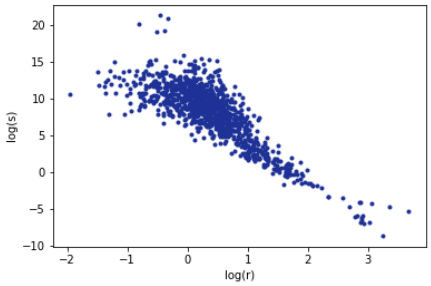
In figura 11 vi è la proiezione sul piano del cielo dell'ammasso. In figura 12 sono riportati gli andamenti dell'accelerazione, del *jerk* e dello *snap* in funzione del raggio dell'ammasso. Per questi ultimi, si nota nelle regioni centrali una dispersione: probabilmente si tratta di particelle interagenti, in quanto presentano valori di accelerazione, *jerk* e *snap* maggiori rispetto agli andamenti generali.



(a) accelerazione VS raggio



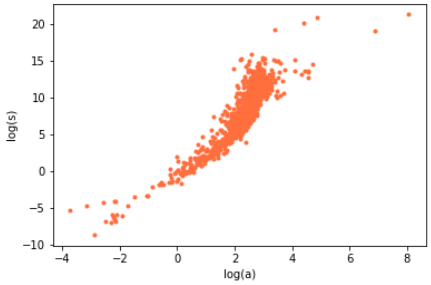
(b) *jerk* VS raggio



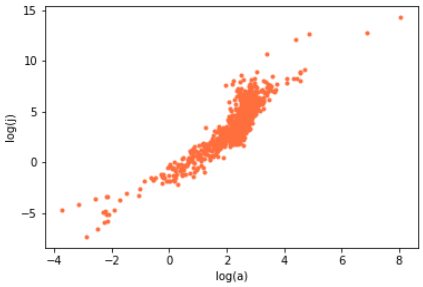
(c) *snap* VS raggio

Figura 12: Andamenti di accelerazione, *jerk* e *snap* in funzione del raggio delle particelle nell'ammasso campione in scala logaritmica.

In figura 13, invece, vi sono i *jerks* e gli *snaps* in funzione delle accelerazioni. Infine in figura 14 è riportato il profilo radiale di densità di massa dell'ammasso. Tutti i grafici sono in scala logaritmica.



(a) *jerk* VS accelerazione



(b) *snap* VS accelerazione

Figura 13: *Jerk* e *snap* in funzione dell'accelerazione delle stelle dell'ammasso in scala logaritmica.

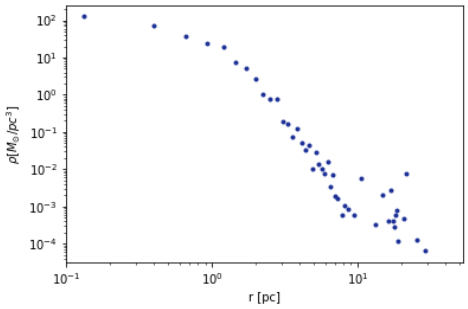


Figura 14: Profilo di densità dell'ammasso in scala logaritmica.

Dataset

Tra i risultati delle simulazioni sono stati considerati solo quelli relativi alle particelle delle zone più interne degli ammassi, in quanto, normalmente le MSPs si trovano proprio in tali regioni.

Si è deciso di definire le zone centrali degli ammassi per mezzo del raggio che contenesse il 20% delle stelle totali. Successivamente, in queste regioni, sono state eseguite 6 estrazioni casuali per ogni ammasso di un numero di MSPs variabile tra 1 e 40. Trattandosi di simulazioni che non tengono conto dell'evoluzione stellare, le particelle risultano tutte uguali, per questo motivo la scelta di estrarle casualmente è ragionevole. Tuttavia, si è scelto di selezionarle nelle regioni centrali poiché è lì che normalmente si trovano.

I *dataset* forniti all'algoritmo di ML sono in totale sei ed essi differiscono tra loro in base al diverso numero di MSPs estratte per ciascun ammasso, ovvero 1, 5, 10, 20, 30 o 40. Pertanto, ogni *record* dei diversi *dataset* corrisponde ad un ammasso e, come già detto, gli ammassi simulati sono circa 200. Ciascun ammasso può contenere o meno un IMBH ed è caratterizzato da un certo numero di particelle da 1 a 40 che sono state estratte casualmente come MSPs.

Ogni MSP è caratterizzata da posizione nell'ammasso, velocità, accelerazione, *jerk* e *snap*. Ma le caratteristiche delle MSPs che si è deciso di fornire come *features* all'albero decisionale sono:

- il raggio proiettato sul piano del cielo, cioè la distanza di ogni particella dal centro di massa dell'ammasso: $r_p = \sqrt{x^2 + y^2}$;
- la componente z della velocità, v_z;
- la componente z dell'accelerazione, a_z;
- la componente z del *jerk*, che ora indicheremo con j_z;
- la componente z dello *snap*, che indicheremo con s_z.

Sulla base di tali *features* per ciascun *dataset*, l'algoritmo di ML sarà in grado di predire la presenza di un IMBH negli ammassi con una certa accuratezza ed *F-score*.

Codice

Il codice sviluppato per la predizione degli IMBHs al centro di GCs mediante l'utilizzo di alberi decisionali è stato scritto in Python 3.7.0 utilizzando i moduli **NumPy** [**numpy:online**] e **scikit-learn** [**sklearn:online**]. Esso è consultabile al seguente link di *jupyter notebook*: **inserire**.

Il codice si sviluppa principalmente in quattro parti:

- Fase di preparazione del *dataset*;
- Fase di addestramento sul *training set*;
- Fase di validazione;
- Valutazione delle prestazioni sul *test set*.

Durante la prima fase, il *dataset* viene diviso nei tre sottogruppi: *training set*, *validation set* e *test set* (Sez.). Per questo lavoro si è scelto di considerare nel *test set* il 20% dei dati e il restante 80% viene suddiviso nel seguente modo: il 30% di questi dati costituisce il *validation set* ed il rimanente 70% fa parte del *training set*. Pertanto, il *dataset* è diviso in: 56% in *training set*, 24% in *validation set* e 20% in *test set*.

Nella fase successiva, l'albero decisionale di classificazione viene addestrato sul *training set*. Trattandosi di un algoritmo supervisionato, quindi, gli viene fornito in input anche il vettore delle *labels* in cui le due classi considerate vengono indicate con:

- **classe 0 o "no"**: "non c'è un IMBH nell'ammasso";
- **classe 1 o "yes"**: "c'è un IMBH nell'ammasso".

Per distinguere le classi si è fissato a priori un valore di soglia per la massa. In particolare, si è decisiso di inserire nella **classe "yes"** tutti gli ammassi per cui

$$\frac{M_{BH}}{M_{clus}} > 0.14$$

dove con M_{BH} si indica la massa del buco nero centrale e con M_{clus} la massa dell'intero ammasso. Quando, invece, la massa del buco nero centrale risulta

inferiore o uguale al 14% della massa dell'intero ammasso, consideriamo che non ci sia un IMBH al centro dell'ammasso in questione.

Durante la fase di validazione, poi, avviene il processo di *pruning* dell'albero (Sez.). In questa fase si vuole scegliere il parametro di complessità α relativo al modello allenato per cui non si abbia *overfitting*, ma che permetta di migliorare l'accuratezza delle previsioni. Come già visto in maggiore dettaglio nelle Sezioni e , per ogni valore del parametro α che l'algoritmo fa variare in maniera crescente tra 0 e 1, esiste un'unica sottostruttura dell'albero T che minimizzi la 18. Ad ognuno degli α selezionati e salvati in memoria dall'algoritmo corrisponde un modello allenato. Si vuole ora scegliere il migliore tra essi in modo da ottenere un classificatore in grado di predire in maniera accurata la presenza degli IMBHs al centro dei GCs. Per fare ciò si utilizza il *validation set*. In particolare, in seguito alla fase di addestramento, il codice identifica il modello classificatore corrispondente al relativo α in grado di fare una predizione sul *validation set* con la miglior accuratezza possibile.
In figura 15 si riporta l'andamento delle accuratezze dei modelli allenati per ciascun valore del parametro di complessità α in funzione del numero progressivo degli α stessi. Questo grafico è un esempio ottenuto a partire dal *dataset* con l'estrazione di 20 MSPs per ogni ammasso, ma si è svolto lo stesso procedimento per tutti i *dataset*.
Il modello scelto, quindi, è quello per cui si ha il massimo valore di accuratezza nelle previsioni sul *validation set*.

Una volta scelto il modello, le sue prestazioni vengono valutate su un set di dati totalmente nuovi per lui: il *test set*. Dopo il *pruning*, infatti, il modello è in grado di classificare nuovi ammassi con una certa accuratezza senza produrre *overfitting*.

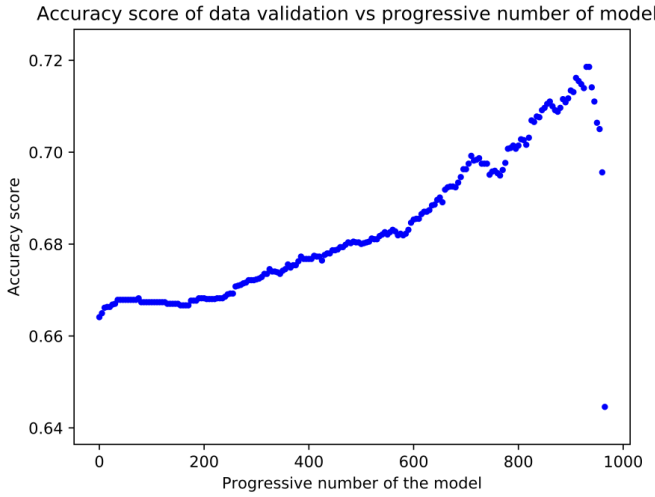


Figura 15: Il plot è stato ottenuto a partire dal *dataset* contenente 20 MSPs per ogni ammasso. Si riporta l'andamento delle accuratezze ottenute con ciascun modello allenato in funzione del numero progressivo di α. Il modello scelto è quello che restituisce il massimo valore di accuratezza sul *validation set*.

Risultati

Lo scopo principale e, se vogliamo la sfida, di ogni problema di ML consiste nell'ottenere delle buone *performance* dell'algoritmo su dati nuovi e mai visti. Come descritto nel capitolo precedente, dopo aver scelto il miglior modello di albero decisionale mediante la tecnica del *pruning*, bisogna valutare le sue prestazioni sul *test set*. In questo Capitolo, dunque, vengono presentati i risultati ottenuti dal classificatore su ognuno dei *dataset* considerati. Gli strumenti utilizzati per la valutazione sono le matrici di confusione (Sez.) e le metriche di precisione, richiamo ed *F-score*. Infine, proprio grazie alle caratteristiche degli alberi decisionali, è possibile fornire un'interpretazione dei risultati sulla base dell'*importanza* delle *features*.

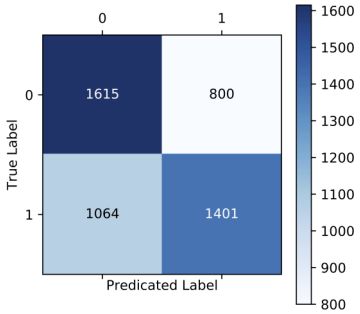
Valutazione delle performance

Sono stati utilizzati sei *dataset* diversi per la previsione della presenza di IMBHs al centro di GCs a partire dalle caratteristiche dinamiche delle MSPs che si trovano nelle regioni centrali degli ammassi. I *dataset* differiscono tra loro per il numero di MSPs selezionate casualmente in queste zone per ogni ammasso: 1, 5, 10, 20, 30 o 40.

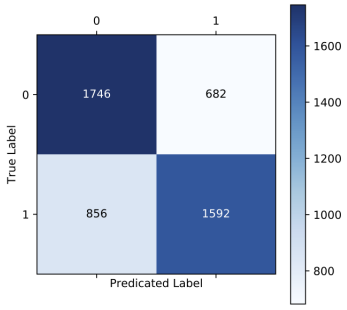
Le prestazioni degli alberi decisionali vengono valutate sui relativi *test set* in base alla loro capacità nel classificare gli ammassi nella **classe 0** o nella **classe 1**.
In figura 16 sono riportate le matrici di confusione ottenute considerando il miglior modello classificatore per ogni *dataset*. Esse sono un utile strumento per visualizzare la qualità delle classificazioni. Infatti, mettono in relazione quelli che sono i reali valori delle *labels* (*"True label"*) con i valori che,

invece, sono stati predetti (*"Predicted label"*) per ciascuna classe. Nelle celle sulle diagonali vi è il numero di risposte corrette date dal classificatore, mentre nelle altre celle vi è il numero di risposte sbagliate. Si vede che, con l'aumentare del numero di MSPs selezionate nel centro di ciascun ammasso, aumenta anche il numero di risposte corrette da parte dell'albero decisionale. In particolare, i corrispondenti valori delle metriche di precisione, richiamo ed *F-score* sono riportati nelle tabelle 3. Per maggiore chiarezza, infine, in figura 17, vi è l'andamento della *F-score* per ciascuna classe, in funzione del numero di MSPs considerato per ogni ammasso. Si vede che, per entrambe le classi, i valori di *F-score* aumentano notevolmente (di circa il 10%) con l'aumentare del numero di MSPs, fino a 10 MSPs per ammasso. Da 20 MSPs in poi, le *F-score* si stabilizzano al 72% e 73%.

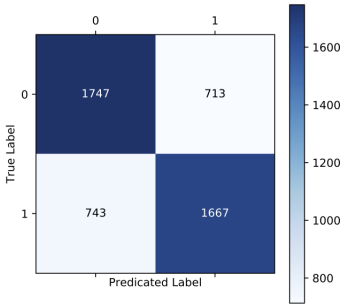
Discussione: risultati buoni? perchè? perchè ad un certo punto la f1 si stabilizza?



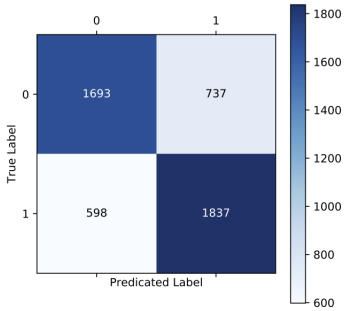
(a) 1 MSP per ammasso.



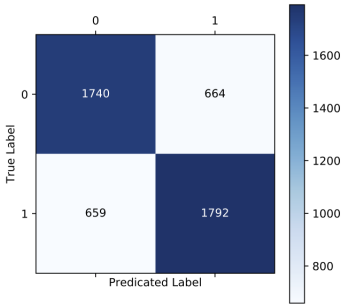
(b) 5 MSPs per ammasso.



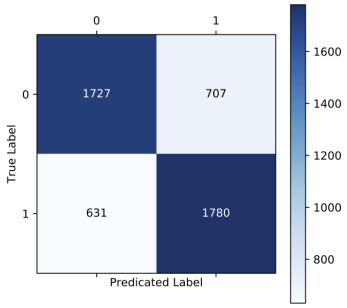
(c) 10 MSPs per ammasso.



(d) 20 MSPs per ammasso.



(e) 30 MSPs per ammasso.



(f) 40 MSPs per ammasso.

Figura 16: Matrici di confusione ottenute sui *test set* per i sei diversi *dataset*.

classe	precisione	richiamo	F-score
0	0.60	0.67	0.63
1	0.64	0.57	0.60

(a) 1 MSP per ammasso.

classe	precisione	richiamo	F-score
0	0.67	0.72	0.69
1	0.70	0.65	0.67

(b) 5 MSPs per ammasso.

classe	precisione	richiamo	F-score
0	0.70	0.71	0.71
1	0.70	0.69	0.70

(c) 10 MSPs per ammasso.

classe	precisione	richiamo	F-score
0	0.74	0.70	0.72
1	0.71	0.75	0.73

(d) 20 MSPs per ammasso.

classe	precisione	richiamo	F-score
0	0.73	0.72	0.72
1	0.73	0.73	0.73

(e) 30 MSPs per ammasso.

classe	precisione	richiamo	F-score
0	0.73	0.71	0.72
1	0.72	0.74	0.73

(f) 40 MSPs per ammasso.

Tabella 3: Risultati di precisione, richiamo ed F-score per ciascuna classe, ottenuti sui *test set* per i sei diversi *dataset*.

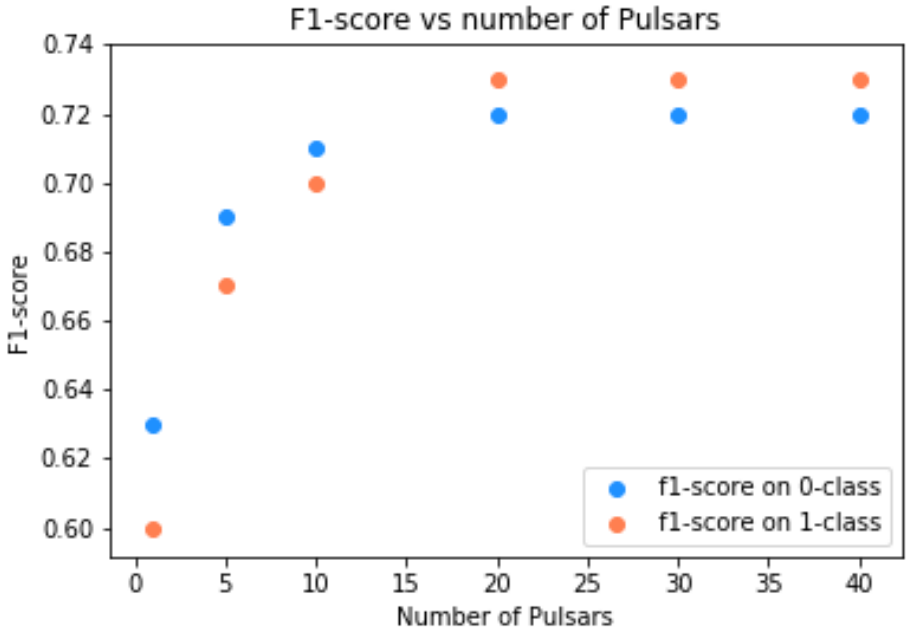


Figura 17: Andamento della *F-score* per le due distinte classi in funzione del numero di MSPs considerate per ogni ammasso.

Interprtazione dei risultati

Gli alberi decisionali, soprattutto se non sono molto profondi, sono semplici da interpretare, in quanto essi possono essere visualizzati.
In figura 18 è riportato un esempio di albero decisionale ottenuto con il *dataset* relativo ad una MSP estratta per ogni ammasso.

L'interpretazione è semplice: partendo dal nodo radice si percorre l'albero passando per i nodi successivi per mezzo dei rami. Questi ultimi suggeriscono quale sottoinsime andremo a guardare. Abbiamo già visto che gli *split*, per ogni sottoinsieme, vengono decisi in base ad una misura di impurità fatta sulla valutazione dell'indice di Gini.
L'algoritmo sceglie come nodo radice quello con impurità più bassa, ossia quello con il valore di indice di Gini minore.
Una volta deciso il nodo radice, l'albero viene ingrandito ad una profondità pari ad uno. Lo stesso processo viene ripetuto per gli altri nodi nell'albero

fino a raggiungere la massima profondità. In questo caso, stiamo interpretando un albero con profondità pari a 7, ottenuta in seguito alla fase di potatura.

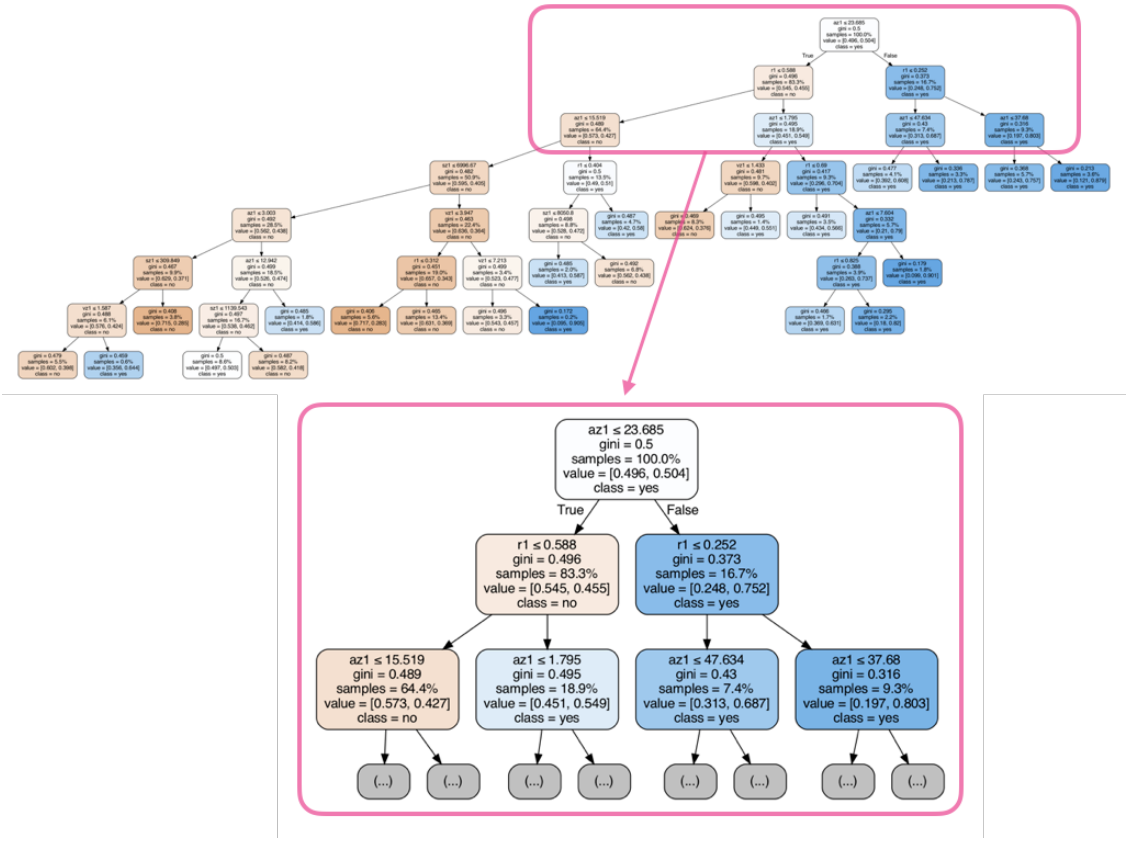


Figura 18: Un esempio di albero decisionale ottenuto a partire dal *dataset* di 1 MSP selezionata nella regione centrale di ogni ammasso considerato. Per chiarezza è stato eseguito uno zoom nei primi *split* dell'albero.

Per maggior chiarezza, in figura 18 è stato fatto uno zoom sui primi *split*, in particolare per il primo nodo abbiamo che:

- $\mathbf{a_{z1} \leq 23.685}$: il primo *split* viene fatto sulla *feature* a_{z1}, ovvero ci si chiede se essa è minore o uguale di 23.685 e, in base al risultato, si segue il percorso *True* (freccia di sinistra) o *False* (freccia di destra);
- $\mathbf{gini = 0.5}$: è l'indice di Gini per la *feature* a_{z1}. Un valore dell'indice di Gini pari a 0 significa che il nodo è puro, ossia che esiste solo una

classe (quindi che gli ammassi appartengono tutti alla **classe "yes"** o alla **classe "no"**). Al contrario, un valore maggiore di 0, come nel nodo radice, ci indica che i campioni all'interno del nodo appartengono ad entrambe le classi;

- **samples** = **100**%: indica la percentuale di ammassi da classificare appartenenti al *training set*. Nel nodo radice è corretto aspettarsi che ancora tutti gli ammassi devono essere classificati;

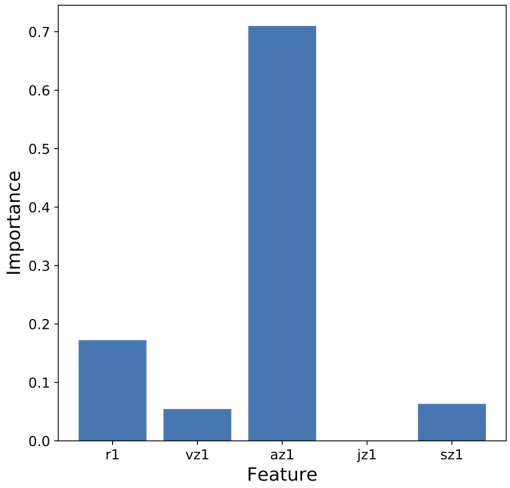
- **value** = [**0.496**, **0.504**]: indica la frazione di ammassi di quel nodo che ricadono in ciascuna classe. Ciò significa che nel nodo radice il 49.6% degli ammassi appartiene alla **classe "no"** (che abbiamo anche chiamato **classe 0**) ed il restante 50.4% fa parte della **classe "yes"** (chiamata anche **classe 1**);

- **class** = **yes**: rappresenta la previsione su quel determinato nodo e si ricava da **value**. Infatti, dal punto sopra, capiamo che nel nodo c'è una percentuale maggiore di campioni appartenenti alla **classe "yes"**.

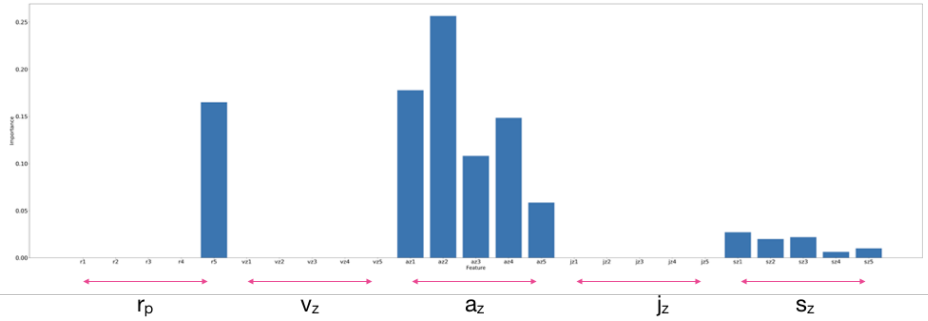
Queste denominazioni vengono ripetute per i vari nodi successivi dell'albero. Nello specifico, si può notare che la somma delle *samples* dei nodi figli restituisce il valore *samples* dei nodi padri.
Inoltre, la **classe "yes"** è identificata con l'azzurro, mentre la **classe "no"** con l'arancione. L'intensità della colorazione di ogni nodo varia in base all'accuratezza della previsione: vediamo che nei nodi di azzurro più intenso una percentuale maggiore di ammassi viene classificata in **"yes"** e in quelli di arancione più intenso, invece, una percentuale maggiore di ammassi viene classificata in **"no"**.
Infine, nei nodi foglia non appaiono più le *features*, in quanto si è raggiunta la massima profondità e tutti gli ammassi in quel nodo, sono stati classificati.

In ultimo, è possibile valutare l'importanza delle diverse *features* per un albero decisionale. L'importanza complessiva di una singola *feature* può essere calcolata nel modo seguente: bisogna esaminare tutti gli *split* per i quali la *feature* è stata utilizzata e misurare quanto essa ha ridotto l'indice di Gini rispetto al nodo padre. La somma di tutte le importanza viene scalata a 100, pertanto, ogni importanza può essere interpretata come una quota

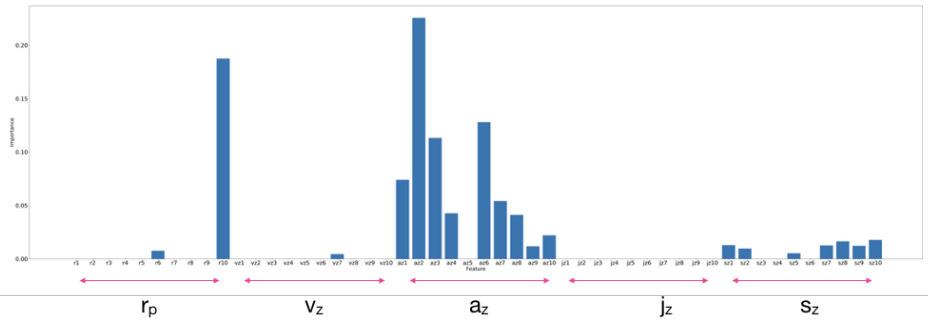
dell'importanza complessiva per il modello [**molnar2019:book**]. Nelle figure 19 e 20 vengono riportati gli istogrammi relativi alle importanze delle diverse *features* per ciascuno dei *dataset* utilizzati. Le MSPs sono ordinate per raggi r_p proiettati sul piano del cielo crescenti. E, le altre *features* sono i rispettivi valori di velocità v_z, accelerazione a_z, *jerk* j_z e *snap* s_z delle MSPs lungo la linea di vista. Si può notare che, in tutti i casi, le *features* con importanza maggiore sono: i raggi, in particolare quello dell'ultima MSP, le accelerazioni e gli *snap*.
Che le accelerazioni fossero caratterizzate da un'alta importanza era un risultato atteso. Una possibile spiegazione, invece, per l'importanza del raggio dell'ultima MSP potrebbe essere che il modello lo interpreti come una misura della dimensione dell'ammasso, o meglio, come la distanza che delimita lo spazio in cui si trovano le MSPs considerate.
Per quanto riguarda, infine, velocità e *jerks*, ossia le derivate dispari delle posizioni, essi sembra che non abbiano importanza per il modello **perchè?**



(a) 1 MSP per ammasso.

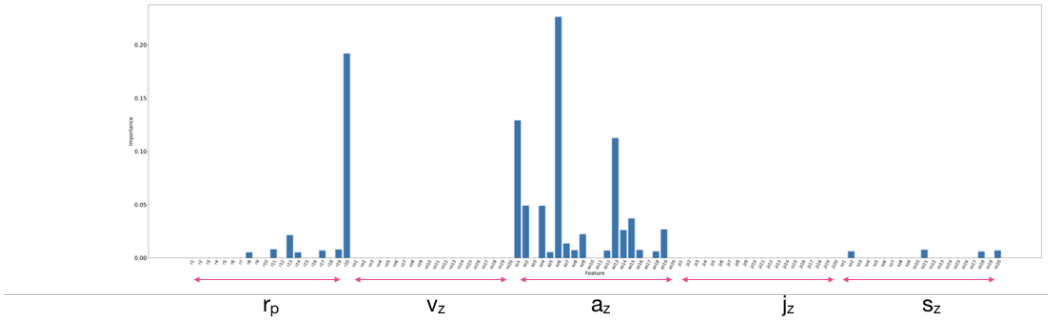


(b) 5 MSPs per ammasso.

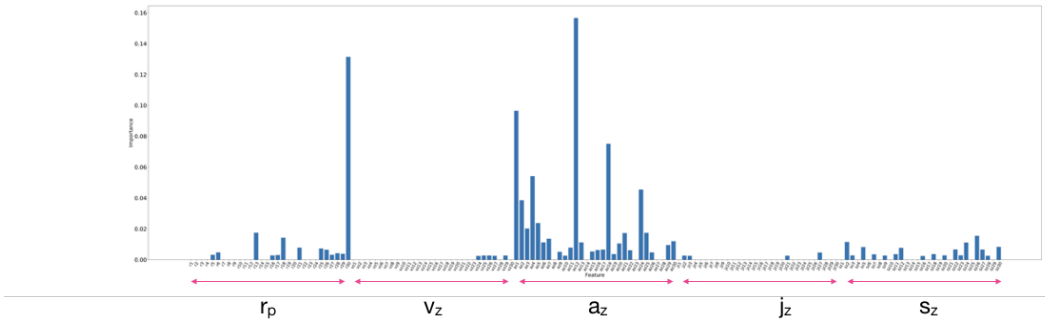


(c) 10 MSPs per ammasso.

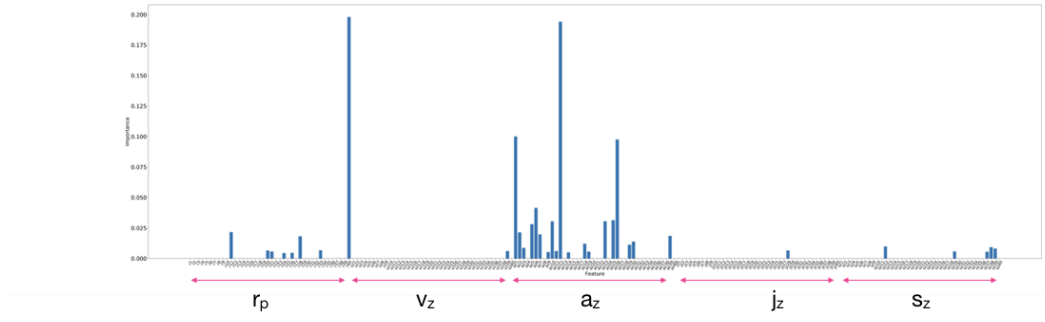
Figura 19: Grafici delle importanze delle *features* per il modello calcolate per i *dataset* relativi a 1, 5 e 10 MSPs per ammasso (dall'alto verso il basso).



(a) 20 MSPs per ammasso.



(b) 30 MSPs per ammasso.



(c) 40 MSPs per ammasso.

Figura 20: Grafici delle importanze delle *features* per il modello calcolate per i *dataset* relativi a 20, 30 e 40 MSPs per ammasso (dall'alto verso il basso).

Conclusioni e sviluppi futuri

Gli IMBHs, la loro esistenza ed il fatto di poterli trovare al centro dei GCs sono stati per lungo tempo oggetto di dibattito nel mondo scientifico. I modelli teorici attuali indicano che è possibile identificarli negli ammassi globulari, mentre i metodi osservativi usati finora non ne hanno ancora verificato la presenza. Un interessante metodo proposto per identificarli è quello che si basa sullo studio dinamico delle MSPs in ammasso. Da qui nasce l'idea di questo lavoro di tesi: voler identificare gli IMBHs in ammassi globulari attraverso una tecnica di ML interpretabile utilizzando le caratteristiche dinamiche delle MSPs. Si tratta, infatti, di voler applicare un metodo nuovo ad un problema ben noto.

A partire dai risultati di un set di simulazioni a N-corpi di ammassi globulari sono stati creati i *dataset* forniti all'algoritmo di ML. Lo scopo è stato quello di voler prevedere la presenza di IMBHs in questi ammassi sulla base della posizione nel piano del cielo delle MSPs e delle loro componenti lungo la linea di vista di velocità, accelerazione, *jerk* e *snap*. Per farlo ci si è serviti degli alberi decisionali.
Sono stati considerati circa 200 ammassi, per ognuno dei quali sono state effettuate 6 estrazioni casuali di un numero variabile tra 1 e 40 di MSPs selezionate casualmente dalle regioni centrali degli ammassi stessi.

Nonostante gli alberi decisionali non siano tra i modelli di ML più sofisticati, si ottengono dei buoni risultati. In particolare, si riescono a classificare gli ammassi in base alla presenza di un IMBH con valori di accuratezza pari a circa il 70%.

Inoltre, la caratteristica fondamentale degli alberi decisionali è che essi possono essere facilmente interpretati, specie quando si tratta di alberi non molto profondi. Per questo motivo, questi modelli vengono spesso preferiti rispetto ad altri.
Dall'interpretazione delle scelte che il modello compie in ogni nodo dell'albero è possibile capire quali sono state le caratteristiche dinamiche delle MSPs che hanno avuto più importanza nella fase di allenamento del modello. Per tutti i casi analizzati risulta che, come atteso, l'importanza maggiore viene attribuita alle accelerazioni delle MSPs negli ammassi. Molto importante risulta anche il raggio proietatto sul piano del cielo della MSP più distante. Il resto delle importanze è distribuito tra gli *snaps* ed i raggi proiettati delle altre MSPs.

Nella pratica, dopo aver monitorato un certo numero di MSPs in un ammasso misurandone le caratteristiche dinamiche, quello che si vuole capire è se, in base ai loro valori, l'ammasso possa ospitare un IMBH o meno. Se in questa fase avessimo già allenato un modello di ML a partire da dati di sumilazioni, allora si potrebbe prevedere nella realtà la presenza di tali oggetti tanto ricercati.
L'importanza di questo lavoro sta proprio nel fatto che sarebbe possibile ottenere un riscontro futuro su dati veri. Per tale ragione, sarebbe fondamentale affinare questo modello e svilupparne altri a partire da questo, per ottenere previsioni più accurate.
Un primo passo verso il miglioramento delle *performance* dell'algoritmo è quello di rendere le simulazioni da cui provengono i *dataset* il più verosimili possibile, ad esempio considerando anche l'evoluzione stellare. In secondo luogo, invece, si potrebbero migliorare alcuni punti importanti del codice di previsione. Infatti, il criterio considerato per determinare la presenza di un IMBH in un ammasso si basa su un valore di soglia di massa relativa scelto a priori. Questo passaggio è cruciale per la classificazione degli ammassi nelle due distinte classi, per questo sarebbe importante svolgere, in futuro, uno studio più approfondito sulla scelta di tale criterio (**suggerimenti su un possibile metodo?**).

Infine, è lecito sostenere che l'originalità di questo studio stia proprio nei metodi applicati ad una questione astrofisica ormai nota. Per questo

sarebbe importante concentrarsi sui suoi possibili sviluppi e miglioramenti, con la prospettiva che possa essere utile per ricerche future.