UNIVERSITÀ DI PISA

DIPARTIMENTO DI INFORMATICA

DISTRIBUTED DATA ANALYSIS AND MINING

Asteroid dataset

Michele Papucci (544376)

Chiara Giurdanella (560686)

ANNO ACCADEMICO 2022/2023

INDICE

1. Introduzione

Asteroid dataset[1] è un dataset ufficialmente gestito dal *Jet Propulsion Laboratory* del California Institute of Technology, un'organizzazione della NASA, che contiene informazioni su quasi un milione di asteroidi. I dati disponibili sono relativi ai parametri orbitali, a proprietà fisiche, alla posizione degli oggetti nello spazio, e sono stati resi disponibili per scopi di ricerca scientifica. L'analisi svolta si è concentrata prima di tutto su uno studio del dataset e delle sue caratteristiche, a cui è seguita un'analisi mediante l'uso di algoritmi di clustering per vedere come le classi orbitali si distribuiscono in essi. Successivamente sono stati implementati task di classificazione, al fine di predire se un asteroide è potenzialmente pericoloso per la Terra o meno. Al miglior modello sono state applicate tecniche di *Explainability* per vedere quali sono, effettivamente, i parametri usati dal modello per classificare un asteroide come pericoloso.

2. Data understanding e preparation

Sul dataset originale, che si compone di 958524 righe e 45 colonne, sono state eseguite delle operazioni di pre-processing, mirate all'eliminazione di variabili ridondanti o poco significative per gli obiettivi di analisi prefissati.
Allo scopo di ridurre ulteriormente la dimensionalità, e di evitare informazioni ridondanti, è stata rimossa una variabile per ogni coppia di variabili che hanno mostrato una correlazione molto alta, con un coefficiente di *Pearson* superiore a 0.9.

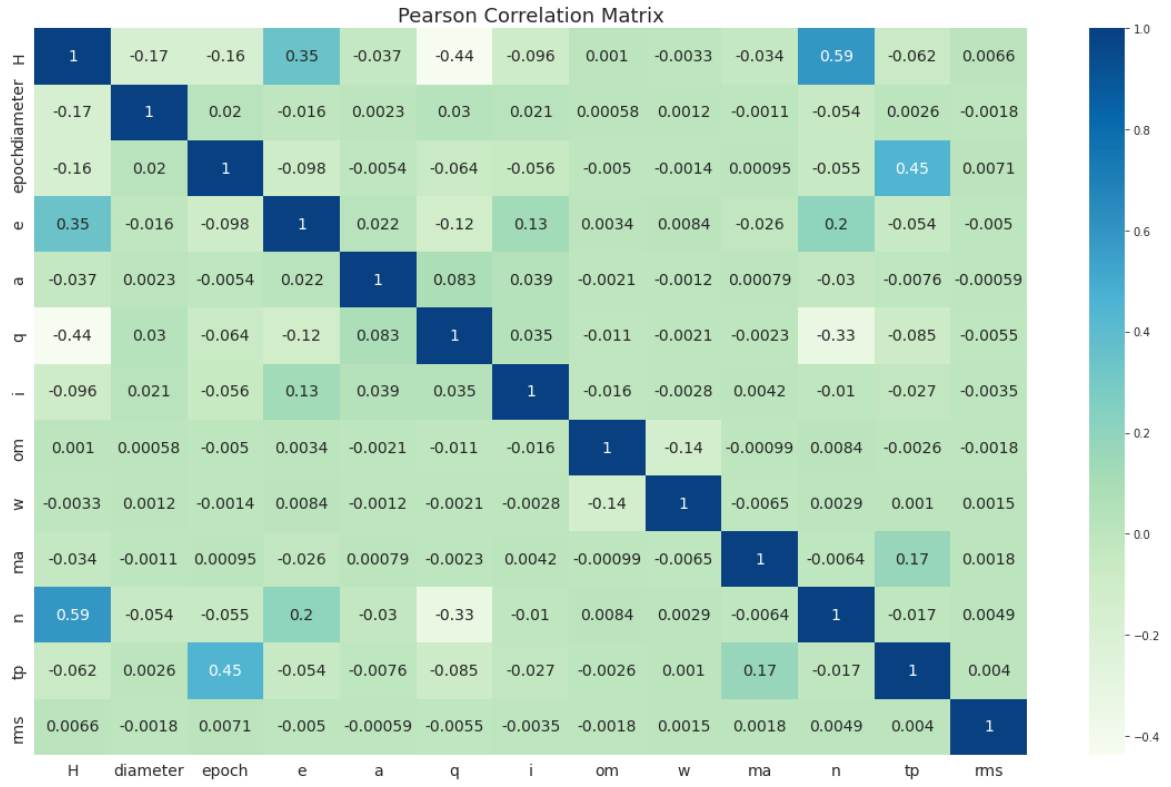


Figura 1 Matrice di correlazione

[1] https://www.kaggle.com/datasets/sakhawat18/asteroid-dataset

In questo modo, alla fine, si è ottenuto un dataset con informazioni poco correlate tra loro, riducendo quindi la dimensione del set di dati ma cercando di mantenere più informazioni possibili.

Questa fase ha interessato poi la ricerca di missing values che sono stati riscontrati, in quantità trascurabile, nelle variabili *neo, ma,* rms, per le quali sono stati eliminati i record interessati. La stessa soluzione è stata adottata per la variabile *pha*, nonostante il numero di valori nulli rilevato fosse abbastanza più alto, precisamente 19920 istanze interessate; questa scelta è dovuta al fatto che, essendo *pha* variabile target del task di classificazione implementato, non erano utili record che non potessero addestrare o valutare le performance dei classificatori applicati.
Si è scelto di procedere in modo differente, invece, per gli attributi *H*, *diameter* e *albedo*; in particolar modo, per la variabile *H*, per la quale sono stati identificati 6262 missing values, si è deciso di sfruttare le correlazioni precedentemente calcolate.

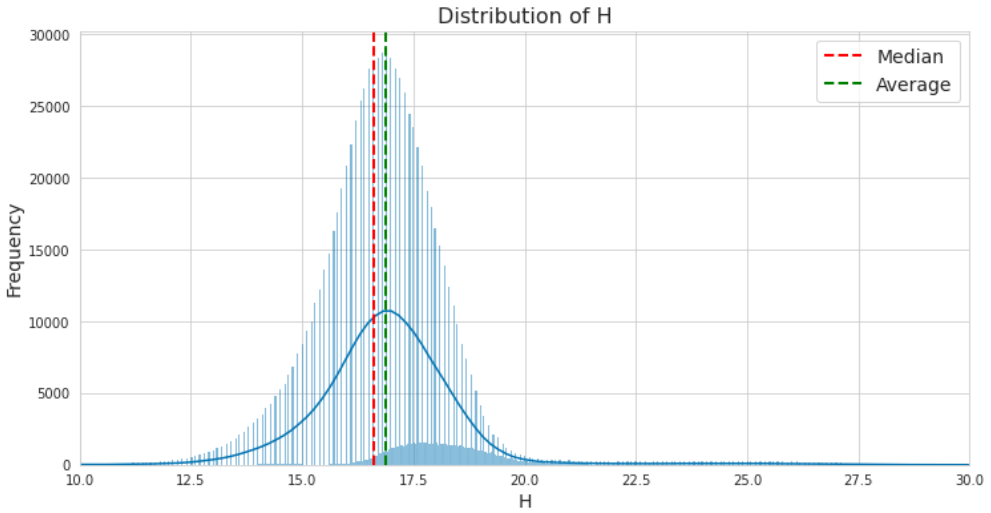


Figura 2 Distribuzione della variabile *H*

Com'è possibile osservare nella Figura 2 la distribuzione di *H* è particolare, con valori di *H* ad altissima frequenza alternati a valori di *H* a bassissima frequenza e, nel suo caso, l'utilizzo di media o mediana sarebbe stato poco significativo. Inoltre, com'è visibile nella matrice di correlazioni in Figura 1, tra *n* e *H* c'è un buon grado di correlazione, che è stato sfruttato per addestrare un regressore lineare alla predizione dei valori di *H*.

Per confermare il lavoro effettuato è stata anche studiata la distribuzione di *H* dopo aver predetto i missing values e non è stato osservato nessun cambiamento significativo, questo è comunque dovuto al fatto che i missing values sono molto pochi rispetto alla grandezza del dataset (lo 0.6%). Per approfondire il lavoro svolto sulla regressione è stato diviso il training set (tutti i record in cui *H* non è mancante) in training e test (10% del training), ed è stato poi utilizzato il test set per valutare le performance del regressore.

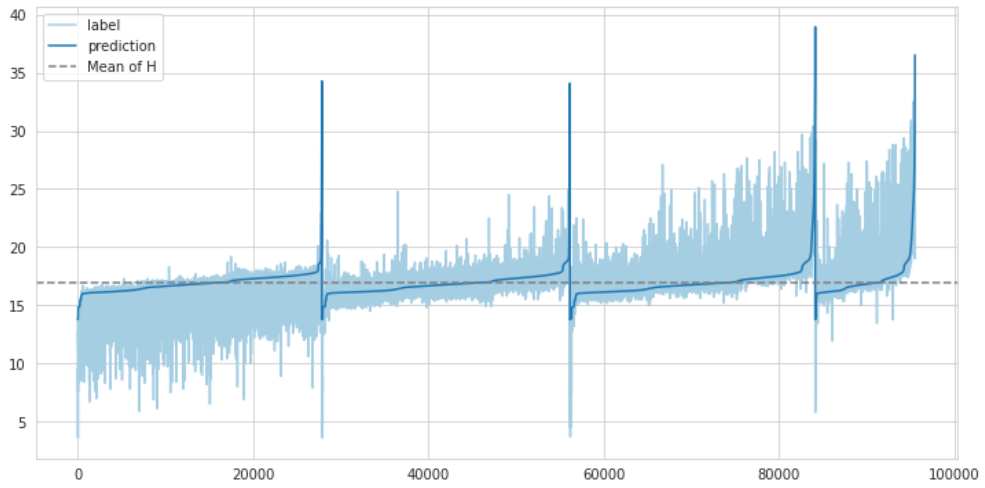


Figura 3 Andamento di *H* rispetto a *n*, valori originali e predetti

La metrica scelta è stata *R2 (coefficiente di determinazione)*, per la quale un dummy regressor, che predice sempre la media, ha ottenuto uno score di -0.0000026225, mentre il regressore lineare, che predice usando *n* come feature, ha ottenuto uno score di 0.32.
Lo score, seppur non altissimo, indica un miglioramento e, come si può vedere anche dalla Figura 3, in generale il regressore riesce ad approssimare meglio il comportamento della variabile *H*.
Per *Diameter* e *albedo*, le cui distribuzioni sono mostrate in Figura 4 e in Figura 5, è stato trovato un numero estremo di valori mancanti, superiore all'85% del numero totale di record; per la prima variabile, che presenta una distribuzione normale abbastanza stretta, si è deciso di trattare questi valori sostituendoli con il valore medio rispettivo, mentre nessuna delle precedenti metodologie è stata ritenuta altrettanto adatta per *albedo*. La variabile presenta una distribuzione *right-skewed* piuttosto marcata, con valore medio e mediana poco significativi; inoltre, non presenta nessuna correlazione rilevante con le altre variabili disponibili e, alla luce di queste osservazioni, si è proseguito con l'eliminazione di tale variabile.

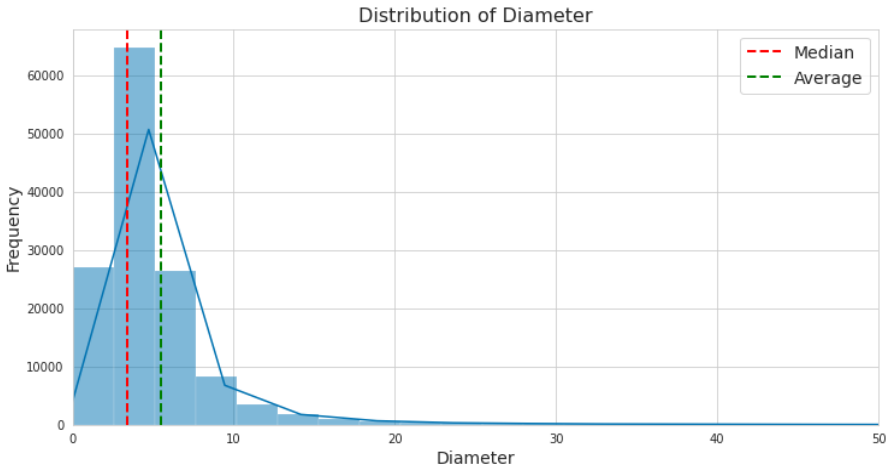


Figura 4 Distribuzione della variabile *diameter*

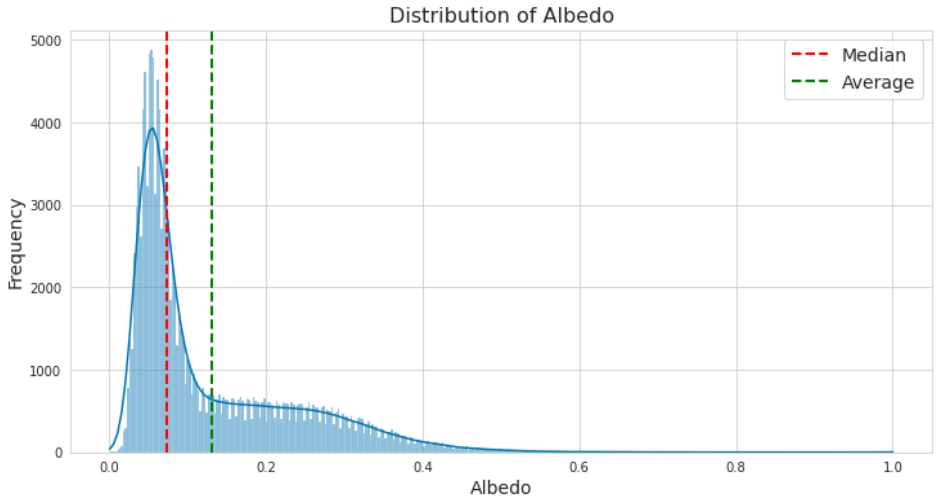


Figura 5 Distribuzione della variabile *albedo*

Il dataset finale su cui si sono svolte le analisi successive si compone di 938597 record e 17 colonne, la cui semantica è mostrata in Tabella 1.

Attributo	Data type	Descrizione	Range di valori
id	String	Id assegnato a un asteroide.	958524 valori distinti
neo	String	Indica se l'oggetto è vicino alla Terra.	Due valori distinti: N e Y
pha	String	Indica se l'asteroide è potenzialmente pericoloso.	Due valori distinti: N e Y
H	Float	La magnitudine visiva di un asteroide è misurata dal punto in cui l'osservazione è stata posta a distanze eliocentriche e geocentriche unitarie ad angolo di fase zero.	-1.1 - 33.2
diameter	Float	Diametro di un asteroide in unità di chilometro.	0 - 939
epoch	Float	Epoca dell'osculazione nella forma modificata del giorno giuliano.	2.43m - 2.46m
e	Float	L'eccentricità è un parametro orbitale che descrive la struttura dell'orbita. È un rapporto tra fuochi e semiasse maggiore.	0 - 1.86
a	Float	È il semiasse maggiore di un'orbita ellittica, che è la metà dell'asse maggiore.	-14.7k - 33.5k
q	Float	La distanza del perielio, indicata con q, è la distanza più vicina tra il corpo orbitante e il sole.	0.07 - 80.4
i	Float	L'inclinazione è l'angolo tra un vettore normale al piano orbitale dei corpi orbitanti e il piano dell'eclittica come piano di riferimento.	0.01 - 175
om	Float	La longitudine del nodo ascendente, indicata con om, è l'angolo tra il sistema inerziale dell'equinozio di primavera e il punto di passaggio attraverso il sistema di riferimento.	0 - 360

w	Float	L'argomento del perielio, indicato con w, è l'angolo tra la linea del nodo ascendente e il punto del perielio nella direzione dell'orbita.	0 - 360
ma	Float	L'anomalia media è il prodotto del moto medio del corpo orbitante e del passaggio al perielio passato.	-70.7 - 492
n	Float	Il moto medio, indicato con n, è la velocità angolare in gradi unitari al giorno per completare un'orbita attorno a un'ellisse ideale.	0 - 2.38
tp	Float	Tempo di passaggio al perielio.	2.28m - 2.55m
class	String	Classe dell'asteroide.	12 valori distinti
rms	Float	Valore quadratico medio del segnale.	0 - 2.69k

Tabella 1 Semantica degli attributi

3. Clustering

Questa fase di analisi ha interessato l'osservazione della distribuzione delle classi orbitali, attraverso una tecnica di clustering di tipo *center-based*, ovvero l'algoritmo *K-means.* Il primo passo è stato rendere i dati adatti a essere processati adeguatamente dall'algoritmo, escludendo le variabili *rms, id, pha, neo* e *class*, e applicando delle operazioni di vettorizzazione e normalizzazione alle restanti.

La variabile *class*, che indica appunto la classe degli asteroidi, si compone di 12 valori distinti, che corrispondono ai tipi differenti di classi orbitali, e la cui distribuzione, con la frequenza rappresentata in scala logaritmica, all'interno del dataset è mostrata nel grafico in Figura 6. La loro semantica è osservabile nella Tabella 2.

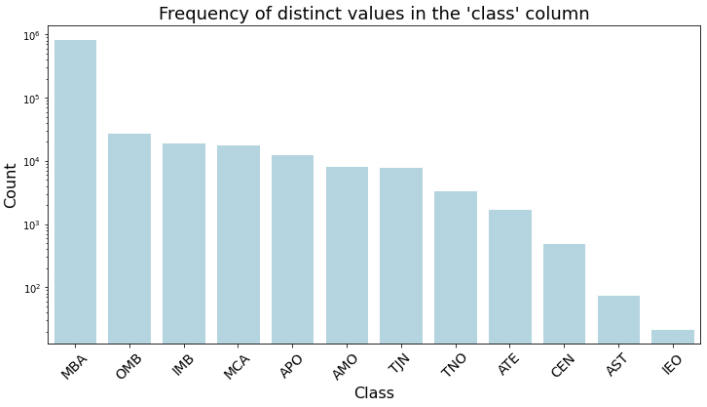


Figura 6 Distribuzione di frequenza dei valori della variabile *class*

Classe	Descrizione
OMB	Oggetti che orbitano tra Marte e Giove, nei tratti esterni della fascia principale degli asteroidi.
IEO	Oggetti con un'orbita interamente confinata all'interno dell'orbita terrestre.
ATE	Oggetti la cui orbita potrebbe portarli in prossimità della Terra.
TJN	Oggetti che condividono l'orbita di Giove intorno al Sole.

APO	Oggetti la cui orbita attraversa l'orbita della Terra.
AST	L'orbita dell'asteroide non corrisponde a nessuna classe di orbita definita.
TNO	Oggetti la cui orbita si estende oltre l'orbita di Nettuno.
IMB	Oggetti in orbita tra Marte e Giove, all'interno della porzione interna della fascia degli asteroidi.
MCA	Oggetti con un'orbita che attraversa l'orbita di Marte.
AMO	Oggetti la cui orbita si avvicina all'orbita della Terra ma non la attraversa.
CEN	Oggetti con un'orbita compresa tra Giove e Nettuno.
MBA	Oggetti in orbita tra Marte e Giove, nella parte principale della fascia degli asteroidi.

Tabella 2 Semantica delle classi orbitali

3.1 K-means

Come anticipato, per l'esecuzione di *K-means* sono stati normalizzati i dati. Empiricamente si è scelto di usare uno *Standard Scaler*. È stato provato anche il *MinMax* ma portava inaspettatamente a dei valori di *Silhouette* più bassa e ad una distribuzione delle classi all'interno dei cluster ancora più omogenea.
La prima operazione, per l'applicazione dell'algoritmo *K-means* al set di dati, richiede di selezionare il numero *k* di cluster desiderati. Per determinare il numero ottimale del parametro *k* è stato implementato un approccio iterativo utilizzando un ciclo for. I risultati ottenuti sono stati valutati mediante i punteggi di *SSE (Sum of Squared Error)* e *Silhouette*, per valori incrementali di *k*, impostati da 4 a 30.

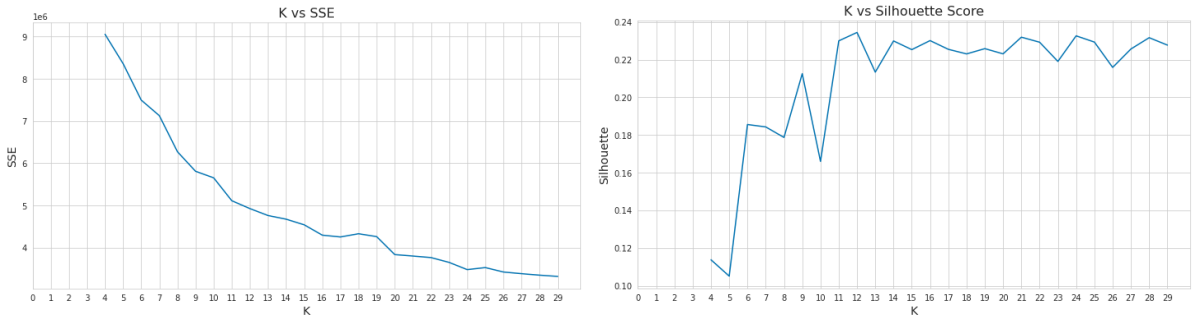


Figura 7 Valori di *SSE* e *Silhouette* per valori crescenti (da 4 a 30) del parametro *k* per l'algoritmo *K-means*

Dal grafico del *Silhouette score* in Figura 7, è possibile osservare un picco in corrispondenza di *k* uguale a 12. Allo stesso k, nel grafico del *SSE* ci troviamo nell'area di gomito; con questa configurazione è stato ottenuto un *Silhouette score* pari a 0.230.

Allo scopo di indagare la composizione dei cluster, è stata analizzata la distribuzione delle classi al loro interno, mostrata in Figura 8; per ragioni di visualizzazione di nuovo la frequenza delle classi è stata rappresentata in scala logaritmica.

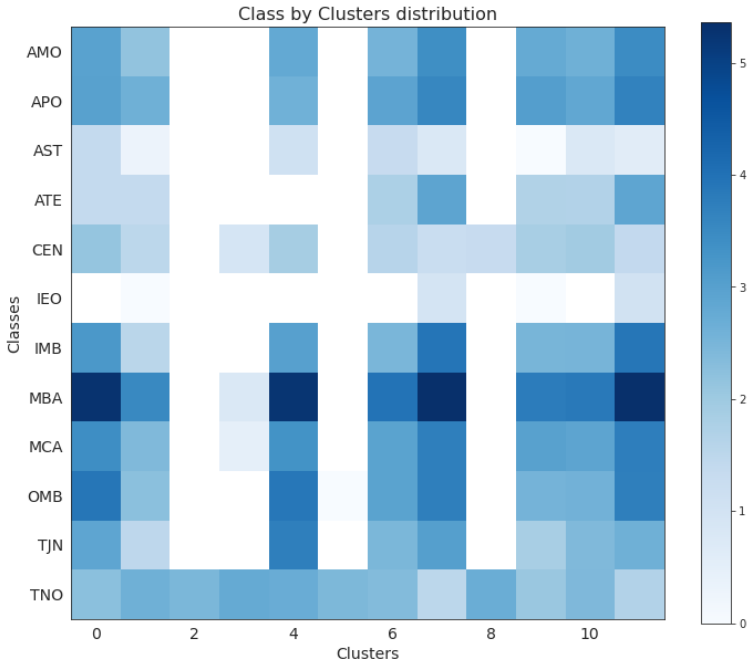


Figura 8 Distribuzione delle classi all'interno dei cluster, con frequenza in scala logaritmica

Dal grafico è evidente come non vi sia un'associazione forte tra cluster e classi. Si può notare come la maggior parte delle classi siano distribuite più o meno equamente attraverso i cluster trovati. Più interessanti sono i cluster 2, 3, 5 e 8 che hanno al loro interno, quasi esclusivamente, elementi della classe *TNO* che sono però presenti anche all'interno di tutti gli altri cluster in egual misura. In linea generale è possibile dire che, tramite il clustering, non è possibile dividere gli asteroidi nelle loro classi orbitali; probabilmente i cluster trovati rappresentano qualcosa di diverso.

4. Classificazione

Come anticipato, uno dei task su cui si è concentrata l'analisi è la classificazione, con l'obiettivo di costruire un modello in grado di predire correttamente la pericolosità di un asteroide per la Terra.

Il dataset è stato splittato in maniera casuale in training set, impiegato per l'addestramento dei modelli di classificazione, pari all'80% del set originale, e test set, corrispondente al 20% e utilizzato per testarne le capacità predittive. Allo scopo di valutare le prestazioni dei classificatori sono state adottate le seguenti metriche: *Precision*, *Recall*, *F1 score*, *AUC (area under the curve) score* e *Accuracy*; per ciascuno dei modelli sono, inoltre, state calcolate le matrici di confusione, riportate in ogni sezione specifica. Sono stati confrontati due modelli di classificazione: *Support Vector Machine* e *Gradient Boosting*.

Prima di passare all'esposizione dei risultati ottenuti, è necessario introdurre un'analisi della variabile target *pha*, composta dai valori *N* e *Y*; le due etichette, che stanno per *No* e *Yes*, e che in fase di pre-processing sono state convertite in 0 e 1, indicano se un asteroide è pericoloso o meno. È stato riscontrato uno sbilanciamento piuttosto significativo tra le classi dell'attributo che influisce sulla capacità dei modelli di apprendere correttamente dai dati di addestramento; è, pertanto, importante tenere in considerazione questo fattore durante la valutazione e l'interpretazione dei risultati mostrati nelle sezioni seguenti. In particolare, la classe principale *(N)* rappresenta il 99.77% delle istanze.

Dalle variabili utilizzate per la classificazione sono state escluse *rms, id, pha, neo* e *class*, o per la loro natura categoriale o perché ritenute poco utili ai fini della predizione. La fase di pre-processing è consistita nel trasformare le features in un unico vettore, e nella loro normalizzazione, utilizzando gli strumenti *VectorAssembler* e *StandardScaler* della libreria *pySpark*.

4.1 Support Vector Machine

Il primo modello di classificazione che è stato impiegato è *Support Vector Machine*, abbreviato in *SVM*, un algoritmo di apprendimento automatico supervisionato che cerca di trovare un confine ottimale, noto come iperpiano, che separi i dati in diverse classi, utilizzando un sottoinsieme degli esempi di training, chiamati vettori di supporto.
L'algoritmo *SVM* utilizza un insieme di funzioni matematiche, definite come *kernel*, che prendono i dati come input e li trasformano nella forma richiesta.
L'implementazione dell'algoritmo *SVM* scelto è *LinearSVC* della libreria *pyspark* ed è stato utilizzato con i parametri di default, che corrispondono a un kernel lineare e a un numero massimo di iterazioni per la convergenza pari a 100. Le prestazioni del modello sono state valutate attraverso le metriche precedentemente elencate e i risultati sono riportati in Tabella 3 e in Figura 9.

	Support Vector Machine				
	Precision	*Recall*	*Weighted F1*	*AUC*	*Accuracy*
N	0.997	1.0	0.996	0.991	0.997
Y	0.0	0.0			

Tabella 3 Metriche di valutazione per *SVM*

TN	FP
187238	0
FN	TP
422	0

Com'è possibile vedere sia dalle metriche misurate dalla matrice di confusione, sia dalla matrice stessa, l'*SVM* non sembra aver imparato a separare le due classi ma ha adottato la strategia di classificare tutto il test set come la classe di maggioranze.

Figura 9 Matrice di confusione *SVM*

Questo potrebbe essere dovuto al fatto che in un set di dati così pesantemente sbilanciato, l'algoritmo potrebbe avere difficoltà a trovare un iperpiano che separi accuratamente le classi.

4.2 Gradient Boosting

Sulla base dei risultati ottenuti e dei dati disponibili, si è scelto poi di testare le performance di un altro modello: *Gradient Boosting*. In particolare, la scelta è dovuta alla composizione della variabile target *pha* la quale, come anticipato, presenta un forte sbilanciamento tra le classi, per cui solo 2066 asteroidi sono stati etichettati come pericolosi.

Con dati altamente sbilanciati, questo modello potrebbe avere delle performance migliori degli *SVM*, essendo un metodo di apprendimento automatico di tipo ensemble, che crea un modello predittivo più robusto, combinando insieme dei modelli di apprendimento più deboli. Ad ogni iterazione, l'algoritmo cerca di correggere gli errori dell'iterazione precedente, con l'obiettivo di ridurre al minimo l'errore complessivo o la differenza tra il valore effettivo della classe dell'esempio di addestramento e il valore della classe predetto. L'implementazione scelta è *GBTClassifier* della libreria *pyspark,* della quale sono stati mantenuti gli iperparametri di default.

I risultati ottenuti dal modello sono stati valutati utilizzando le misure di cui sopra e sono presentati in Tabella 4 e in Figura 10.

	Gradient Boosting				
	Precision	*Recall*	*Weighted F1*	*AUC*	*Accuracy*
N	0.997	0.997	0.996	0.993	0.997
Y	0.846	0.025			

Tabella 4 Metriche di valutazione per *Gboost*

TN 187236	FP 2
FN 415	TP 11

Il classificatore riesce, rispetto a *SVM*, ad imparare qualcosa e a predire correttamente qualche dato come classe *'Y'*, ma la *recall* rimane vicina a zero e, con questi tipo di dati, la *recall* è la metrica più significativa: è preferibile classificare più asteroidi come pericolosi e scoprirli innocui, che il contrario.

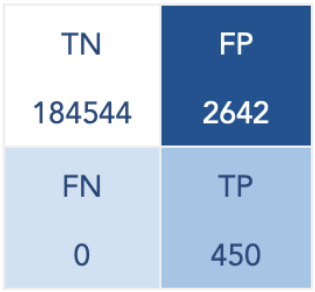
Figura 10 Matrice di confusione *GBoost*

4.3 Classificazione su dati bilanciati

È importante tenere a mente che sull'efficacia di ciascun algoritmo influiscono diversi fattori, in primis la quantità e la qualità dei dati a disposizione. Dopo aver eseguito l'addestramento e i test sull'intero set di dati disponibili, è stata osservata una bassa capacità di predizione della classe minoritaria. Ciò ha condotto alla decisione di provare a riequilibrare i dati mediante tecniche di bilanciamento. Nello specifico è stato applicato un *undersampling* di tipo casuale al dataset. E, con i nuovi dati, si è proceduto a ri-addestrare il miglior modello che avevamo fino a questo punto, ovvero il *Gradient Boost Tree*, testandone poi le prestazioni sul test set originale.

	Gradient Boosting con *Random Undersampling*				
	Precision	*Recall*	*Weighted F1*	*AUC*	*Accuracy*
N	1.0	0.985	0.991	0.996	0.985
Y	0.145	1.0			

Tabella 5 Metriche di valutazione *Gboost*, su dataset bilanciato



Com'è visibile dai risultati, osservabili in Tabella 5 e in Figura 11, grazie a questo undersampling adesso il modello riesce a classificare correttamente tutti i record della classe *'Y'*, e la *recall*, di conseguenza, è salita al 100%.

Figura 11 Matrice di confusione *GBoost* su dataset bilanciato

In conclusione, l'implementazione dell'*undersampling* si è rivelata una scelta efficace per gestire l'alto sbilanciamento della variabile target, consentendo di ottenere risultati di classificazione più precisi e affidabili.
Sarebbe interessante fare esperimenti usando anche tecniche di *oversampling* o dove si utilizzano entrambe le tecniche di *sampling*. Al momento ciò non è possibile farlo, usando *pyspark,* senza implementare manualmente queste tecniche.

5. Explainability

Questa sezione è incentrata sull'applicazione di tecniche di *Explainability* globale, allo scopo di fornire delle spiegazioni per la classificazione ottenuta utilizzando l'algoritmo *Gradient Boosting* sui dati bilanciati, presentata nella *Sezione 4.3*.

Allo scopo di aumentare la comprensione del modello e fornire un'interpretazione dei risultati è stata, anzitutto, calcolata la rilevanza di ciascuna variabile, in modo da quantificare quanto ciascuna di esse ha influito sulle predizioni del modello di classificazione. La *feature importance*, infatti, rappresenta un metodo comunemente utilizzato per interpretare i modelli di apprendimento automatico, misurando l'importanza relativa di ciascuna variabile di input rispetto alla previsione del modello. Nella Tabella 6 sono riportati i risultati estratti.

Feature	Importanza
H	0.006836358853359786
diameter	4.864755220984198e-16
epoch	0.002816553181794318
e	0.01031717569656276
a	0.008641881628611995
q	0.9244261568218762
i	0.013280163653434703
om	0.006728031495115179
w	0.009162886242976093
ma	0.0063694942107942135
n	0.0063012414303633985
tp	0.0051200567851108405

Tabella 6 Features importance

Si è deciso, poi, di confrontare le feature importance con i primi split di un albero decisionale costruito con *TREPAN,* un explainer globale di tipo *model-agnostic* che approssima un black box model utilizzando un albero decisionale. Quindi, si è addestrato un *Decision Tree* sul test set del modello da spiegare, utilizzando le label predette dal modello per l'addestramento. L'algoritmo così ci restituisce un *Decision Tree* che è un modello surrogato dell'originale e, in quanto albero binario, è intrinsecamente spiegabile. Dato che l'interesse era posto principalmente sui primi split, dove avvengono le decisioni con il massimo guadagno, è stata impostata una *max_depth=4*. L'albero ottenuto è mostrato in Figura 12, la quale è stata generata a partire dai dot data estratti dall'oggetto *DecisionTreeClassifier* e poi disegnata tramite un tool online[2].

[2] Graphviz online

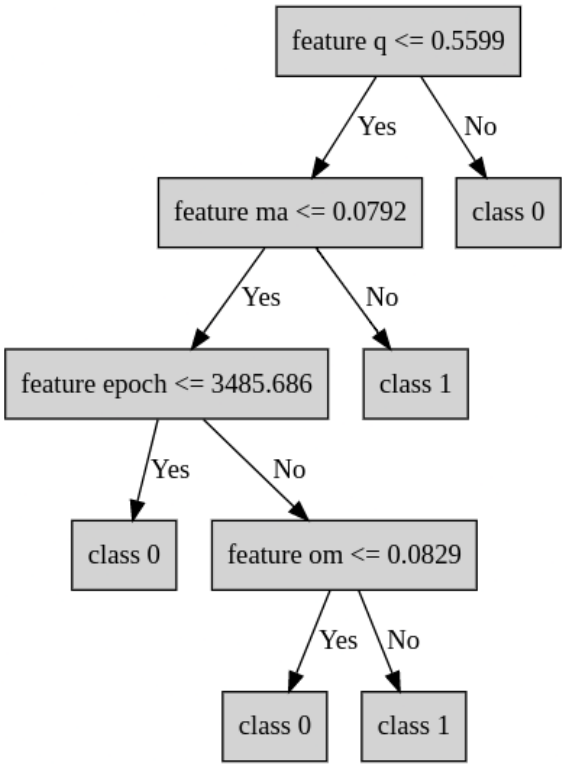


Figura 12 *Decision Tree* che approssima il comportamento di *GBoost* sul dataset bilanciato

Da un primo confronto, ciò che emerge è una coerenza per quanto riguarda la feature più importante, la quale combacia con quello che è il primo split dell'albero, ovvero *q*. Questa variabile risulta essere molto significativa per il tipo di analisi svolta, con una rilevanza pari a 0.92.
La definizione con la quale la NASA classifica gli asteroidi[3] è *"A potentially hazardous asteroid is one with an orbit that comes within 0.05 AU (about 4,650,000 miles or 7,480,000 km) of Earth's orbit and has an absolute magnitude, a measure of brightness, of 22 or less*". Da questo si evince che *H* dovrebbe essere una feature molto importante, poiché è uno dei due parametri, insieme alla distanza dalla Terra, utilizzato per classificare gli asteroidi pericolosi. Nonostante questo, *H* non appare nei primi split dell'albero e ha una *feature importance* molto bassa.
Non avendo tra le feature scelte una misura diretta della distanza tra la Terra e il corpo, è possibile che il classificatore riesca ad avere informazioni implicite sulla posizione del corpo rispetto alla Terra utilizzando *q*, che da definizione è la distanza più vicina tra il corpo orbitante e il Sole. Quindi, è possibile che oggetti che siano distanti dal Sole più o meno quanto lo è la Terra vengano identificati come pericolosi. Una possibile alternativa per meglio rappresentare questo concetto potrebbe essere l'introduzione della variabile categoriale *neo* indicizzata come variabile binaria per rappresentare meglio questo concetto.

[3] https://www.jpl.nasa.gov/edu/news/2017/4/18/how-nasa-studies-and-tracks-asteroids-near-and-far/