

Università degli Studi di Napoli "Parthenope"



Dipartimento di Scienze e Tecnologie
Corso di Laurea in Informatica
Tesi di Laurea Sperimentale in Informatica

A Machine Learning Classifier for Vetting Planet Candidates in TESS Data

Relatori

Prof.ssa Laura Inno
Prof. Angelo Ciaramella

Candidato

Attilio Di Vicino
Matr. 0124002347

ANNO ACCADEMICO 2022/2023

Alla mia famiglia

Indice

Elenco delle figure	4
1 Introduzione	9
2 Il mondo degli esopianeti	10
2.1 Cos'è un esopianeta?	10
2.2 Come identificare un esopianeta	10
2.2.1 Il Metodo del Transito	11
2.3 Threshold Crossing Event	12
2.4 Telescopi Spaziali	13
2.4.1 Kepler	13
2.4.2 Transiting Exoplanet Survey Satellite	13
3 Metodologie utilizzate	16
3.1 Intelligenza Artificiale	16
3.2 Il Machine Learning	17
3.2.1 Modello McCulloch-Pitts	18
3.3 Reti Neurali Artificiali	18
3.4 Il Deep Learning	19
3.5 Apprendimento Supervisionato	20
3.5.1 Classificazione	20
3.5.2 Regressione	20
3.5.3 Alberi Decisionali	20
3.5.4 Random Forest	22
3.5.5 K Nearest Neighbors	23
3.5.6 Gradient Boosting	24
3.6 Apprendimento Non Supervisionato	26
3.6.1 Clustering	26
3.6.2 Self-Organizing Maps	26
3.7 Elaborazione dei Dati	29
3.7.1 Gestione dei valori mancanti	29
3.7.2 Selezione delle Caratteristiche	31
3.7.3 Normalizzazione dei Dati	31
3.7.4 Bilanciamento del set di dati	32
4 Risultati sperimentali	34
4.1 Set di dati	34
4.2 Elaborazione dei dati	35
4.2.1 Set di dati: TESS	35
4.2.2 Set di dati: Kepler	36

INDICE

4.2.3	Set di dati: TESS/Kepler	38
4.2.4	Caratteristiche dei set di dati elaborati	39
4.3	Apprendimento Supervisionato	39
4.3.1	Set di dati: TESS	39
4.3.2	Set di dati: Kepler	42
4.3.3	Set di dati: TESS/Kepler	44
4.4	Apprendimento Non Supervisionato	46
4.4.1	SOM: Test 1	46
4.4.2	SOM: Test 2	52
4.4.3	SOM: Test 3	57
4.4.4	SOM: Test 4	62
4.5	Risultati	67
5	Conclusioni	68
5.1	Osservazioni	68
5.2	Sviluppi Futuri	69

Elenco delle figure

2.1	Quando un oggetto (e.g. pianeta, stella, ecc.), rappresentato dai pallini più piccoli, uno rosso e uno blu, transita di fronte alla stella osservata, si verifica un decremento della luminosità della stella.	11
2.2	Nella prima rappresentazione si evidenzia un settore della volta celeste con la direzione di puntamento delle 4 telecamere di TESS, nella seconda rappresentazione invece vengono mostrate le coperture di tutti i settori, che vanno da 27 giorni fino a 351 giorni.	14
3.1	Diagramma di Venn dei concetti e delle classi di Machine Learning (fonte del diagramma [1]).	17
3.2	Neurone Biologico <i>vs</i> Rete Neurale Artificiale.	19
3.3	Rete Neurale Profonda.	19
3.4	Struttura Albero Decisionale.	21
3.5	Architettura Self-Organizing Map.	28
4.1	Matrice di Confusione dalla Random Forest sui dati TESS.	40
4.2	Importanza delle caratteristiche restituite dalla Random Forest sui dati TESS.	41
4.3	Matrice di Confusione della Random Forest sui dati Kepler.	42
4.4	Importanza delle caratteristiche restituite dalla Random Forest sui dati Kepler.	43
4.5	Matrice di Confusione della Random Forest sui dati TESS/Kepler.	44
4.6	Importanza delle caratteristiche restituite dalla Random Forest sui dati TESS/Kepler.	45
4.7	Decadimento del tasso di apprendimento (Test 1 SOM).	48
4.8	Decadimento del raggio del vicinato (Test 1 SOM).	48
4.9	Decadimento del tasso di apprendimento e del raggio del vicinato (Test 1 SOM).	48
4.10	U-Matrix Rettangolare (Test 1 SOM).	49
4.11	Cluster delle singole <i>features</i> nello spazio della SOM (Test 1 SOM).	49
4.12	Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 1 SOM).	50
4.13	Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 1 SOM).	50
4.14	<i>Label Map</i> (Test 1 SOM).	51
4.15	<i>Label Map</i> ricostruita sulla base di un'intorno 3x3 (Test 1 SOM).	51
4.16	Matrice di confusione (Test 1 SOM).	51
4.17	Decadimento del tasso di apprendimento (Test 2 SOM).	53
4.18	Decadimento del raggio del vicinato (Test 2 SOM).	53

4.19	Decadimento del tasso di apprendimento e del raggio del vicinato (Test 2 SOM).	53
4.20	U-Matrix Rettangolare (Test 2 SOM).	54
4.21	Cluster delle singole <i>features</i> nello spazio della SOM (Test 2 SOM).	54
4.22	Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 2 SOM).	55
4.23	Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 2 SOM).	55
4.24	<i>Label Map</i> ricostruita sulla base di un'intorno 3x3 (Test 2 SOM).	56
4.25	Matrice di confusione (Test 2 SOM).	56
4.26	Decadimento del tasso di apprendimento (Test 3 SOM).	58
4.27	Decadimento del raggio del vicinato (Test 3 SOM).	58
4.28	Decadimento del tasso di apprendimento e del raggio del vicinato (Test 3 SOM).	58
4.29	U-Matrix Rettangolare (Test 3 SOM).	59
4.30	Cluster delle singole <i>features</i> nello spazio della SOM (Test 3 SOM).	59
4.31	Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 3 SOM).	60
4.32	Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 3 SOM).	60
4.33	<i>Label Map</i> ricostruita sulla base di un'intorno 3x3 (Test 3 SOM).	61
4.34	Matrice di confusione (Test 3 SOM).	61
4.35	Decadimento del tasso di apprendimento (Test 4 SOM).	63
4.36	Decadimento del raggio del vicinato (Test 4 SOM).	63
4.37	Decadimento del tasso di apprendimento e del raggio del vicinato (Test 4 SOM).	63
4.38	U-Matrix Rettangolare (Test 4 SOM).	64
4.39	Cluster delle singole <i>features</i> nello spazio della SOM (Test 4 SOM).	64
4.40	Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 4 SOM).	65
4.41	Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 4 SOM).	65
4.42	<i>Label Map</i> ricostruita sulla base di un'intorno 3x3 (Test 4 SOM).	66
4.43	Matrice di confusione (Test 4 SOM).	66

Elenco delle tabelle

3.1	Alcune definizioni dell'AI. Organizzate in quattro categorie	16
4.1	Caratteristiche del set di dati TESS dopo l'elaborazione.	36
4.2	Caratteristiche del set di dati Kepler dopo l'elaborazione.	38
4.3	Caratteristiche del set di dati TESS/Kepler dopo l'elaborazione.	39
4.4	Caratteristiche dei set di dati elaborati.	39
4.5	Confronto delle metriche per diversi modelli sfruttando un approccio supervisionato, utilizzando il <i>dataset</i> di TESS.	40
4.6	Confronto delle metriche per diversi modelli sfruttando un approccio supervisionato, utilizzando il <i>dataset</i> di Kepler.	42
4.7	Confronto delle metriche per diversi modelli sfruttando un approccio supervisionato, utilizzando il <i>dataset</i> di TESS/Kepler.	44
4.8	Riassunto dei risultati ottenuti tramite l'algoritmo del Random Forest su vari set di dati.	46
4.9	Risultati ottenuti dai vari test tramite Self-Organizing Map.	67
4.10	Risultati ottenuti mediante approcci sia supervisionati che non, sul <i>dataset</i> combinato di TESS e Kepler denominato TESS/Kepler.	67

Abstract

In this thesis, we explore the application of Machine Learning (ML) techniques to the classification of exoplanets, focusing on data from NASA's TESS telescope. The TESS space telescope, currently in orbit, detects exoplanets by using the transit method, i.e. by measuring the relative decrease in stellar luminosity that are caused by a planet crossing the stellar disk along the line of sight. When a transiting event is detected with a Signal to Noise ratio of at least seven, it is classified as a Threshold Crossing Event (TCE) and analyzed by data reduction centers, such as the Science Processing Operations Center (SPOC), or by automated pipelines like the Quick-Look Pipeline (QLP). Based on the products of this analysis, the most interesting candidates are then reported for follow-up observations, conducted manually by experts, and, in the end, validated. The goal of this work is to automate the identification process of the most promising candidate planets for which follow-up are to be scheduled, using ML techniques. However, the scarcity of labeled TCEs in TESS data (6,977 total observations, including 5,422 planets and 1,555 false planets) and the imbalanced nature of the dataset requires the inclusion of additional data from the Kepler telescope, allowing us to obtain a merged dataset with 13,798 total observations, including 8,396 planets and 5,402 false planets. Based on this dataset, various ML algorithms have been explored, ranging from supervised approaches to unsupervised ones. Among the supervised approaches that we tested, the Random Forest model achieves the best performance, with an accuracy of 83.69%, and it is also easily interpretable. Among the unsupervised approaches, the Self-Organizing Maps are particularly useful for the visualization of the data. We also explored their use for the binary classification and obtained an accuracy of 77.57%, which, although lower than the one achieved by the Random Forest, is very promising and suggests further opportunities for study based on non-linear maps.

Abstract

Nel presente lavoro, esploriamo l'applicazione di tecniche di Machine Learning (ML) nella classificazione di esopianeti, focalizzandoci sui dati provenienti dal telescopio TESS della NASA. Il telescopio spaziale TESS, attualmente in orbita, utilizza il metodo del transito per individuare gli esopianeti, ovvero misura la diminuzione relativa del flusso luminoso proveniente della stella a causa del passaggio di un pianeta lungo la linea di vista. Quando uno di questi eventi è osservato al di sopra di una significatività minima, viene classificato come Threshold Crossing Event (TCE) e analizzato dai centri preposti alla riduzione dei dati, come lo Science Processing Operations Center (SPOC), o da pipeline automatiche come la Quick-Look Pipeline (QLP). Sulla base dei prodotti di questa analisi, i candidati più interessanti vengono quindi segnalati per osservazioni di *follow-up*, condotte manualmente da esperti, e, infine, validati. L'obiettivo di questo lavoro è automatizzare il processo di identificati dei pianeti candidati più promettenti per i quali programmare *follow-up* mediante l'utilizzo di tecniche di ML. Tuttavia, la scarsità di TCEs etichettati nei dati TESS (6,977 osservazioni totali, di cui 5,422 pianeti e 1,555 falsi pianeti) e la natura sbilanciata del *dataset* richiedono l'inclusione di dati dal telescopio Kepler, permettendoci di ottenere così un set di dati composto da 13,798 osservazioni totali, di cui 8,396 pianeti e 5,402 falsi pianeti. Sulla base di questo data-set, sono stati esplorati vari algoritmi di ML, spaziando da quelli supervisionati (Random Forest, Decision Tree, etc.) a quelli non supervisionati. Nell'ambito degli approcci supervisionati, si apprezza l'efficacia della Random Forest nell'ottenere un'accuratezza del 83,69%, supportata dalla sua interpretabilità. Nell'ambito degli approcci non-supervisionati, le Self-Organizing Map sono particolarmente utili a migliorare la comprensione e la visualizzazione della natura dei dati. Utilizzandole per la classificazione, abbiamo ottenuto un'accuratezza del 77,57%, che, pur essendo inferiore alla Random Forest, è molto promettente e suggerisce ulteriori opportunità di studio basate su mappe non lineari.

Capitolo 1

Introduzione

L'esplorazione dell'universo è un affascinante campo di studio che si è evoluto notevolmente grazie alle moderne tecnologie e alle nuove metodologie scientifiche. In particolare, l'avvento dei telescopi spaziali ha aperto nuove frontiere nella scoperta e classificazione di "esopianeti", cioè pianeti al di fuori del nostro sistema solare. In particolare, le agenzie spaziali NASA ed ESA coordinano osservatori orbitanti per studiare la Terra, il Sole, il nostro sistema solare, la nostra Galassia (Via Lattea) e le galassie più lontane, con un'attenzione particolare alla ricerca di esopianeti [2]. In questo ambito, le missioni più importanti che sono state condotte, sono Kepler e TESS, dei telescopi spaziali progettati per individuare pianeti nella Via Lattea. Kepler fu lanciato nel 2009 per osservare continuamente una regione del cielo, e, durante i suoi 10 anni complessivi di vita, è riuscito a identificare oltre 4,000 esopianeti. Quando la missione è terminata, nel 2018 è stato lanciato il telescopio TESS (Transiting Exoplanet Survey Satellite), anch'esso progettato per individuare esopianeti utilizzando il metodo del transito, ma vagliando l'intera volta celeste. Nonostante il successo di questa missione, la mole di dati generata presenta sfide significative nella classificazione accurata dei pianeti rispetto ai falsi positivi. A fronte dell'enormità dei dati disponibili, infatti, la scarsità di dati correttamente *labelati* che possono essere utilizzati per il training rappresenta una delle principali sfide in questa ricerca, specialmente nel caso di TESS, per cui sono disponibili solamente 6,977 potenziali transiti etichettati, e fortemente sbilanciati tra pianeti e falsi positivi. Per affrontare questa limitazione, abbiamo incluso dati provenienti dal telescopio Kepler, ampliando il *dataset* a 13,798 osservazioni totali. L'obiettivo di questo elaborato di tesi è sviluppare una soluzione innovativa e automatica che affronti questa problematica analizzando i dati provenienti da TESS. La proposta si concentra su un approccio che non fa uso delle tradizionali curve di luce, bensì si focalizza sui parametri fisici ottenuti dall'analisi dei transiti. L'obiettivo è automatizzare il processo di identificazione di veri pianeti rispetto a falsi positivi, che possono essere dovuti ad altri fenomeni astrofisici o a problemi strumentali, mediante l'utilizzo di tecniche di Machine Learning. In questo contesto, sono stati esplorati vari approcci, spaziando da quelli supervisionati a quelli non supervisionati.

Capitolo 2

Il mondo degli esopianeti

Ormai sono trascorsi circa tre decenni dal primo esopianeta scoperto negli anni 90', il pianeta *51 Pegasi b*. Questo evento ha suscitato un forte interesse tra gli scienziati, inaugurando una nuova era dedicata allo studio delle caratteristiche di questi corpi celesti, come temperatura, grandezza, atmosfera e molto altro. Considerato l'importante ruolo che riveste lo studio di pianeti diversi dal nostro, è necessario capire come individuarli e caratterizzarli [3].

2.1 Cos'è un esopianeta?

Gli "esopianeti" sono pianeti che orbitano attorno a una stella diversa dalla nostra, il Sole. Nella nostra galassia, ci sono oltre 400 miliardi di stelle, di cui una è appunto il Sole. Se ipotizziamo che almeno un esopianeta orbiti attorno a ciascuna di queste stelle, possiamo immaginare che ci siano molti pianeti. Anzi, grazie alle scoperte del telescopio spaziale Kepler, sappiamo che potrebbero addirittura esserci più pianeti che stelle. Infatti, si è visto che attorno a una singola stella possono orbitare più pianeti, organizzati in sistemi planetari simili o anche più complessi del nostro. Nella nostra galassia, quindi, il numero di pianeti potrebbe essere superiore a quello delle stelle, e raggiungere quantità anche nell'ordine dei trilioni. Ad oggi, non tutte le stelle della Galassia - e i loro pianeti - sono state osservate, ma solo una parte. Tra quelli scoperti, si sono trovati pianeti di vario tipo, dai giganti gassosi più grandi di Giove ai piccoli pianeti rocciosi delle dimensioni di Terra o Marte. Si sono individuati pianeti con temperature estreme, dai più caldi ai più freddi, alcuni che orbitano così vicino a una stella da avere un anno di durata di soli pochi giorni, altri che ruotano attorno a due stelle contemporaneamente, e altri ancora che possono vagare nella completa oscurità, senza alcuna stella [4].

2.2 Come identificare un esopianeta

Le tecniche impiegate nella ricerca di esopianeti sono cinque, il metodo del *transito*, quello della *velocità radiale*, l'*imaging diretto*, il *Gravitational microlensing* e l'*astrometria*. L'astrometria ha portato alla scoperta di 3 esopianeti, l'*imaging diretto* ne ha individuati 69, il *microlensing* 204, la velocità radiale 1.071, e infine il metodo del transito ha validato ben 4.146 esopianeti. Ognuna di queste tecniche presenta caratteristiche diverse che consentono la validazione di pianeti in modi distinti. Ad esempio, il metodo del *microlensing* ci permette di individuare esopianeti che con altri metodi potremmo non riuscire a trovare. I dati evidenziano chiaramente che il metodo

più utilizzato è quello del transito, che si verifica quando il pianeta, o altro corpo celeste, passa di fronte all’osservatore (nel nostro caso, il telescopio) e una stella. Durante questo evento, osservando la stella per un certo periodo e monitorando le variazioni di luminosità in funzione del tempo, il passaggio di un pianeta di fronte ad essa provoca un leggero abbassamento della luminosità, definendo così l’evento di transito [3].

2.2.1 Il Metodo del Transito

Il transito, come già definito, è un evento che si verifica quando un pianeta passa di fronte a una stella rispetto a un osservatore, producendo una diminuzione relativa del flusso di luce osservato. In questo contesto, gli osservatori sono i telescopi spaziali (e.g., TESS, Kepler, ecc.), dato che le variazioni di luminosità sono piccole ed è richiesta una grande precisione fotometrica. La Figura 2.1 illustra una curva di luce (cioè la luminosità ricevuta in funzione del tempo) durante un evento di transito, evidenziando la riduzione della luminosità nel tempo. Il pallino più grande in giallo rappresenta la stella, mentre i due pallini più piccoli, uno rosso e uno blu, raffigurano i pianeti. Quando questi pianeti passano tra il telescopio e la stella, si verifica un abbassamento della luminosità percepita dall’osservatore, come visibile nel grafico. Da questo fenomeno è possibile estrarre una serie di informazioni, come il periodo, che può essere determinato dal tempo che intercorre tra due transiti, o la dimensione del pianeta, misurata in base alla quantità di luce bloccata durante l’evento [5].

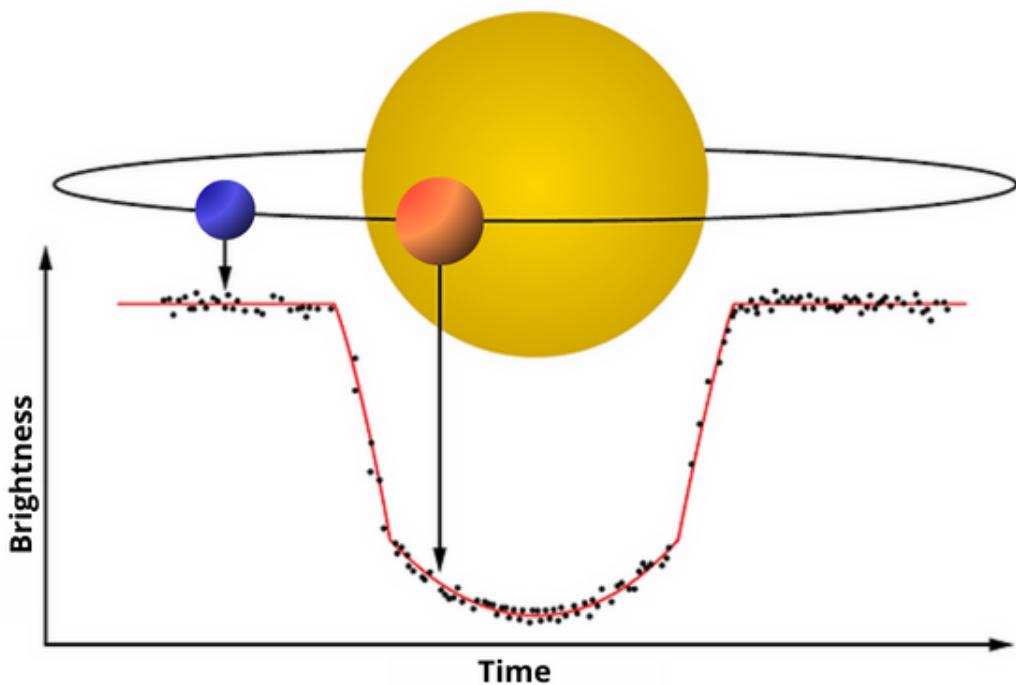


Figura 2.1. Quando un oggetto (e.g. pianeta, stella, ecc.), rappresentato dai pallini più piccoli, uno rosso e uno blu, transita di fronte alla stella osservata, si verifica un decremento della luminosità della stella.

2.3 Threshold Crossing Event

Una volta definito un transito si può caratterizzare un TCE (dall’inglese Threshold Crossing Event): se un transito si verifica in modo periodico e supera una certa soglia di significatività, viene classificato come TCE [6]. Per ogni TCE individuato, viene calcolata una sequenza di caratteristiche, che saranno analizzate al fine di determinare la sua natura. Nello specifico, la domanda che ci si pone in questo contesto è se il TCE in analisi è dovuto al passaggio di un pianeta intorno alla propria stella, oppure se esso è generato da un qualsiasi altro fenomeno di natura astrofisica (una stella che orbita intorno a un’altra, variabilità stellare) o meno (rumore generato dalla strumentazione del telescopio). La grande quantità di TCE che non sono dovuti al passaggio di un pianeta (definiti come falsi positivi), rende indispensabile l’analisi di questi dati con tecniche automatiche e affidabili al fine di supportare le analisi manuali condotte dagli astronomi.

Ottenimento dei Parametri Stellari e Planetari

Il processo di ottenimento dei parametri stellari e planetari inizia dopo l’identificazione dei Threshold Crossing Events (TCEs). Una volta individuati, si procede all’estrazione delle caratteristiche del pianeta candidato mediante un *fitting* della forma del transito. In aggiunta, è possibile utilizzare cataloghi complementari per ottenere i parametri stellari, come quelli prodotti dalla missione spaziale ESA Gaia [7]. La necessità dei parametri stellari deriva dal fatto che le misure ottenute attraverso i transiti forniscono informazioni relative al rapporto tra il raggio della stella e quello del pianeta. Pertanto, conoscere il raggio della stella è cruciale per determinare il raggio del pianeta in modo accurato. Va sottolineato che la misurazione diretta dei raggi stellari è un compito complesso.

2.4 Telescopi Spaziali

Compreso appieno cos’è un esopianeta, è fondamentale disporre della giusta strumentazione che ci permetta di individuare i fenomeni che lo caratterizzano, nel nostro caso il transito, come spiegato in sezione 2.2.1. Diversi strumenti, con tecnologie e architetture diverse, sono impiegati per la cattura di dati relativi agli esopianeti. Tra questi, due telescopi spaziali della NASA, basati sul metodo del transito, sono Kepler, che ha completato la sua missione, e TESS, attualmente ancora operativo.

2.4.1 Kepler

Da diversi anni, l’umanità specula sull’esistenza di pianeti nella nostra galassia, sia che siano abitabili o meno. Tuttavia, fino a qualche decennio fa, mancavano le tecnologie adatte per la caccia agli esopianeti. Nel 2009 fu lanciato il primo telescopio spaziale interamente dedicato a questa ricerca: Kepler. Le osservazioni di Kepler hanno confermato le ipotesi preesistenti, validando migliaia di pianeti e fornendo per la prima volta una stima statistica del numero di pianeti nella galassia, confermando la presenza di trilioni di essi e quindi di veri e propri sistemi planetari. La missione terminò nel 2013 per via di un guasto tecnico, ma grazie all’ingegno dei tecnici della NASA, si riuscì a far rimanere in vita il telescopio proseguendo con le attività scientifiche, che presero il nome della missione K2. Tuttavia le aspettative di vita della missione rimasero basse, data la scarsità di liquido refrigerante necessario al funzionamento degli strumenti, per questo si continuò a lavorare alla preparazione del suo successore TESS, un telescopio spaziale innovativo e tecnologico [8].

2.4.2 Transiting Exoplanet Survey Satellite

Grazie all’esperienza maturata con Kepler, la progettazione di TESS non si basò soltanto sull’obiettivo di prolungare la missione, ma di innovarla. Motivo per cui TESS, acronimo di Transiting Exoplanet Survey Satellite, non fu soltanto il successore, bensì un’evoluzione. Questa missione è gestita dal Massachusetts Institute of Technology (MIT), e si concentra principalmente sull’individuazione di esopianeti tramite l’osservazione di stelle nane più luminose e vicine, consentendo così la rilevazione di piccoli pianeti e agevolando ulteriori studi per valutare la loro abitabilità, compresa l’analisi delle caratteristiche atmosferiche [9].

Caratteristiche della Missione TESS

La missione TESS è stata lanciata il 18 aprile 2018, equipaggiata con quattro telecamere che osservano un settore alla volta. Ciò garantisce una copertura continua minima di circa 27 giorni per ogni settore, estendendosi fino a un massimo di 351 giorni [10]. Nella figura 2.2 sono mostrati due esempi di come TESS monitora la volta celeste. Nella prima, viene evidenziato un singolo settore e la direzione del puntamento delle sue quattro telecamere. È interessante notare che la quarta telecamera, puntata verso il polo dell’eclittica, osserva costantemente la stessa zona, consentendo una copertura fino a 351 giorni. La seconda rappresentazione evidenzia tutti i settori della volta celeste e la distribuzione della copertura delle telecamere, che varia da 27 giorni fino a 351 giorni. La missione prevede osservazioni di 26 settori totali, suddivisi in 13 per l’emisfero dell’eclittica meridionale nel primo anno e altri 13 per l’emisfero dell’eclittica settentrionale nel secondo anno. TESS utilizza il metodo della fotometria di

transito (spiegato in sezione 2.2.1) per il rilevamento di esopianeti, un approccio che si è dimostrato estremamente efficace. È proprio per tutte queste caratteristiche che risulta essere un'evoluzione rispetto a Kepler [11].

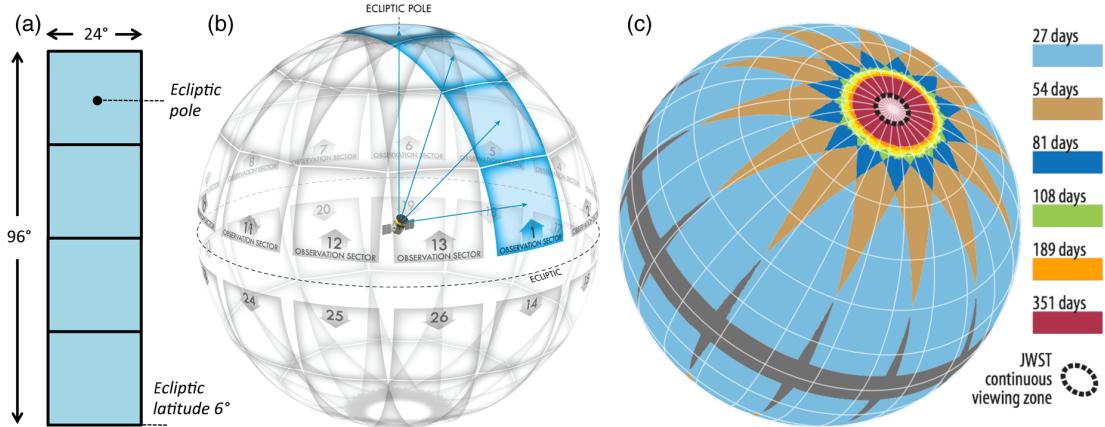


Figura 2.2. Nella prima rappresentazione si evidenzia un settore della volta celeste con la direzione di puntamento delle 4 telecamere di TESS, nella seconda rappresentazione invece vengono mostrate le coperture di tutti i settori, che vanno da 27 giorni fino a 351 giorni.

TESS Objects of Interest

Grazie alle sue 4 fotocamere, TESS, è in grado di catturare immagini full-frame (FFI) raccolte ogni trenta minuti. Seguendo la pipeline che parte dalla cattura fino alla validazione, questi immagini catturate, una volta arrivati a terra, vengono inviate al Science Processing Operations Center (SPOC) della NASA/Ames e al Quick-Look Pipeline (QLP) presso il TESS Science Office del MIT, i quali si occupano dell’elaborazione e della ricerca di esopianeti e oggetti di interesse (TOI). Di conseguenza l’obiettivo di entrambi è quello di selezionare i migliori candidati, per future osservazioni di *follow-up* [12].

Osservazioni di *follow-up*

Una volta individuati i TOI, il team di Follow-up Observing Program (TFOP) di TESS si impegna principalmente a effettuare analisi per soddisfare i requisiti scientifici chiave, che vengono definite come osservazioni di *follow-up*. Parallelamente, l’obiettivo secondario del TFOP è di promuovere la comunicazione e la cooperazione, sia all’interno del TESS Science Team che con l’intera comunità scientifica. Questo si traduce in una riduzione delle duplicazioni di osservazioni e analisi. Per cui con un utilizzo efficiente delle risorse e una comunità sempre più vasta, possono migliorare significativamente la qualità e la quantità delle produzioni scientifiche. Ci sono cinque aree generali di funzionalità, che sono:

- SG1: Fotometria a visione limitata.
- SG2: Spettroscopia di riconoscione.
- SG3: *Imaging* ad alta risoluzione con ottica adattiva.
- SG4: *Precise Radial Velocity*.
- SG5: Fotometria spaziale.

Queste osservazioni quindi sono date da un lavoro da parte di esperti nel settore, e considerando il fatto che il numero di esempi provenienti dai vari telescopi è in costante aumento, la selezione dei migliori candidati e le osservazioni di *follow-up* diventano sempre più onerose [13].

Capitolo 3

Metodologie utilizzate

Spesso, nel contesto odierno, si ritrova soluzione a specifici problemi con sistemi intelligenti che offrono funzionalità di intelligenza artificiale, e molte volte si affidano a tecniche di apprendimento automatico (Machine Learning o ML). Questo capitolo offre una panoramica sull'intelligenza artificiale, con particolare attenzione alle sue ramificazioni come Machine Learning, Artificial Neural Networks e Deep Learning.

3.1 Intelligenza Artificiale

L'intelligenza artificiale (AI) è una disciplina che abbraccia diverse aree, tra cui filosofia, matematica, economia, neuroscienze, psicologia, scienze cognitive, linguistica e informatica [14]. Il concetto di intelligenza ha suscitato un forte interesse sin dagli albori della storia, con l'obiettivo di attribuire una definizione ad esso. Il risultato fu un insieme di definizioni, le quali sono state spesso basate su due dimensioni principali: i processi del pensiero e il ragionamento, nonché il comportamento [15], descritte dalla Tabella 3.1. Alan Turing, uno dei più grandi matematici del XX secolo, propose il "gioco dell'imitazione" come un test per determinare se una macchina può essere considerata "*intelligente*". La sfida consiste nel valutare se essa può simulare il comportamento umano al punto da essere indistinguibile da un terzo attore inteso come "*interlocutore*" o "*giudice*". Tuttavia, definire l'intelligenza di una macchina rimane una sfida, dato che richiederebbe una comprensione approfondita dell'ambiente circostante, del linguaggio naturale, dovrebbe avere la capacità di ragionamento, e poi dovrrebbe apprendere l'individuazione di nuovi pattern, ossia informazioni che derivano dallo specifico contesto. L'obiettivo dell'AI è creare macchine intelligenti che possano pensare e agire in modo simile al cervello umano. Si tratta di una tematica molto complessa, ma uno degli aspetti più interessanti è sicuramente quello dell'apprendimento automatico. Alcune ramificazioni dell'AI sono proprio il Machine Learning (ML), le Artificial Neural Networks (ANNs) e il Deep Learning (DL). Nella figura 3.1 viene mostrata questa gerarchia tramite un diagramma di Venn [1].

Sistemi che pensano come gli esseri umani	Sistemi che pensano razionalmente
Sistemi che operano come gli esseri umani	Sistemi che agiscono razionalmente

Tabella 3.1. Alcune definizioni dell'AI. Organizzate in quattro categorie

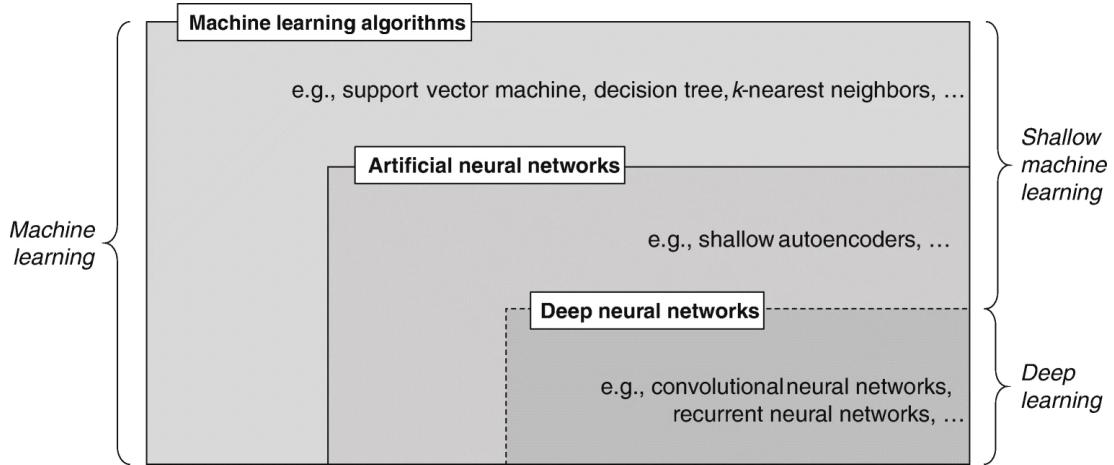


Figura 3.1. Diagramma di Venn dei concetti e delle classi di Machine Learning (fonte del diagramma [1]).

3.2 Il Machine Learning

L'evoluzione dell'intelligenza artificiale ha portato all'emergere del Machine Learning (ML). Questo campo si occupa dello sviluppo di algoritmi e modelli capaci di apprendere da dati e migliorare le prestazioni in compiti specifici. L'obiettivo del ML è rendere i calcolatori capaci di apprendere dall'esperienza umana, senza seguire un modello prefissato. Gli algoritmi di ML, in modo adattivo, migliorano man mano che vengono esposti a nuovi esempi. Inoltre sono usati per individuare dei modelli comportamentali all'interno dei dati per fare delle predizioni. Queste tecniche si sono dimostrate efficaci in diversi settori, come ad esempio mercati azionari, la visione artificiale, l'ingegneria aerospaziale e molti altri [1]. Le tecniche di ML si suddividono in diverse categorie, tra cui:

- Apprendimento Supervisionato: Il modello viene addestrato su un insieme di dati etichettati, cioè dati associati a risultati noti. Le sue principali applicazioni sono la classificazione e la regressione.
- Apprendimento Non Supervisionato: Il modello viene addestrato su dati non etichettati, cercando modelli e relazioni intrinseche nei dati. Include tecniche come il *clustering*.
- Apprendimento Semi-Supervisionato: Utilizza un mix di dati etichettati e non etichettati per l'addestramento, cercando di combinare i vantaggi di entrambi gli approcci.
- Apprendimento per Rinforzo: Un agente impara a compiere azioni in un ambiente, ricevendo feedback positivo o negativo in base alle azioni intraprese [16].

Nella sezione 3.5 esploreremo più nel dettaglio l'apprendimento supervisionato con l'ausilio di alcuni algoritmi. Dall'altro lato, nella sezione 3.6, esploreremo le sfumature dell'apprendimento non supervisionato.

3.2.1 Modello McCulloch-Pitts

Un punto chiave nella storia del ML risale al 1943, quando McCulloch e Pitts proposero il primo modello di neuroni (Equazione 3.1). Il neurone è una cellula che riceve degli stimoli (segnali elettrici) provenienti da altri neuroni, sulla base di questi stimoli, alcuni di essi, si attivano (in gergo si dice che un neurone *"spari"*) è viene prodotto un output. Il funzionamento del modello McCulloch-Pitts trae ispirazione dai questi sistemi nervosi biologici, per questo motivo una singola unità è chiamata neurone, esso produce degli stimoli e fa una somma pesata degli input e se supera una certa somma (definita mediante una funzione di attivazione) si attiva altrimenti no.

$$y = f \left(\sum_{i=1}^n x_i w_i - b \right) \quad (3.1)$$

L'equazione 3.1 descrive il modello McCulloch-Pitts dove x_i è il dato in ingresso del neurone i , w_i è il peso della connessione del neurone i , b è la soglia di questo neurone, f è la funzione di attivazione e y è il dato in uscita di questo neurone [17]. Questo modello ha fornito le basi per lo sviluppo delle reti neurali artificiali.

3.3 Reti Neurali Artificiali

Le Reti Neurali Artificiali (ANNs) sono modelli computazionali ispirati al funzionamento dei neuroni biologici. Sono composte da unità di elaborazione connesse chiamate neuroni, ogni connessione tra neuroni trasmette segnali la cui forza può essere amplificata o attenuata da un peso che viene continuamente modificato durante il processo di apprendimento. I segnali vengono elaborati dai neuroni successivi solo se viene superata una certa soglia, attraverso una funzione di attivazione, a livello biologico si dice che un neurone *"spari"* [1]. Le ANNs sono caratterizzate dalla presenza di tre gruppi di neuroni che sono: *unità di ingresso (input)*, *unità nascoste (hidden)* e *unità di uscita (output)*, meglio descritte dalla figura 3.2. Il neurone artificiale è composto da diverse componenti chiave:

1. Pesi sinaptici (w): Rappresentano la forza della connessione tra i neuroni. Ogni input è moltiplicato per il suo peso corrispondente.

$$\text{Somma pesata} = \sum_{i=1}^n w_i \cdot x_i \quad (3.2)$$

2. *Bias (b)*: Aggiunto alla somma pesata prima di applicare la funzione di attivazione. Contribuisce a introdurre un certo grado di flessibilità nel modello.

$$\text{Somma pesata} = \sum_{i=1}^n w_i \cdot x_i + b \quad (3.3)$$

3. Funzione di attivazione (f): Dopo la somma pesata, il risultato viene passato attraverso una funzione di attivazione che determina l'output del neurone. Una comune funzione di attivazione è la funzione *sigmoid* o la funzione *ReLU*.

$$\text{Output} = f(\text{Somma pesata}) \quad (3.4)$$

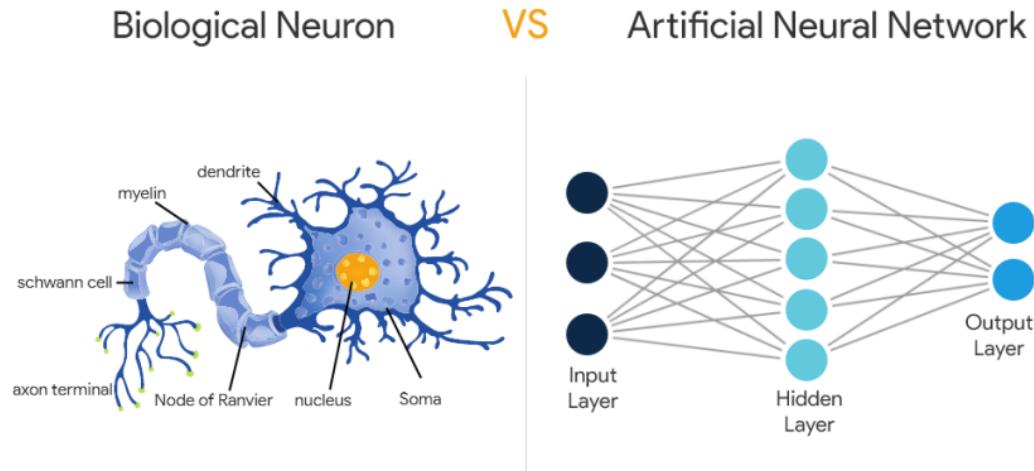


Figura 3.2. Neurone Biologico vs Rete Neurale Artificiale.

3.4 Il Deep Learning

Oltre al Machine Learning e alle Reti Neurali Artificiali, una delle sotto-discipline più avanzate dell’AI è il Deep Learning. Esso è una branca del Machine Learning che si concentra sull’uso di reti neurali artificiali profonde (DNNs) per apprendere e rappresentare dati complessi. Le reti neurali profonde sono caratterizzate dalla presenza di strati multipli (profondità), ognuno dei quali contribuisce alla comprensione sempre più avanzata dei dati. Le unità input ricevono informazioni dall’ambiente esterno, le unità output emettono risposte all’ambiente esterno mentre le unità *hidden* si occupano di comunicare con le unità all’interno della rete. L’ascesa del Deep Learning è stata favorita dai progressi nella capacità computazionale, dalla disponibilità di grandi quantità di dati e dagli algoritmi di apprendimento più efficienti. Questa tecnologia ha dimostrato eccezionali risultati in una serie di applicazioni, inclusi il riconoscimento di immagini, la traduzione automatica, l’analisi del linguaggio naturale e altri specifici compiti complessi [18]. Nella figura 3.3 viene mostrata l’architettura di una rete neurale profonda, e si evidenziano gli n strati di unità nascoste (*hidden*).

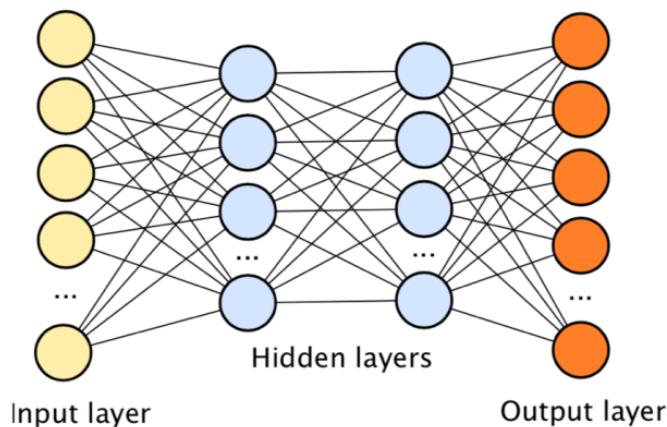


Figura 3.3. Rete Neurale Profonda.

3.5 Apprendimento Supervisionato

L'apprendimento supervisionato rappresenta una metodologia fondamentale nell'ambito dell'apprendimento automatico, dove un modello viene addestrato su dati etichettati per effettuare previsioni o classificare nuovi dati. Quando ci riferiamo a dati etichettati, intendiamo che si conosce a priori la natura dell'input. Il modello apprende la relazione tra un insieme di input e i relativi output attraverso l'addestramento su esempi noti. Questa fase di apprendimento è cruciale, poiché abilita il modello a fornire previsioni accurate quando gli vengono presentati solo nuovi input, senza output. Le tecniche di apprendimento supervisionato sono comunemente impiegate per risolvere problemi di classificazione e regressione. Il tipo di variabile di output può variare nei diversi tipi di problemi del mondo reale. La distinzione nel tipo di output determina lo specifico compito di predizione, con classificazione quando si desidera predire valori qualitativi o regressione quando invece sono quantitativi. Questo capitolo esplorera alcuni algoritmi in questo contesto, offrendo una comprensione delle dinamiche di classificazione e regressione.

3.5.1 Classificazione

La classificazione è una tecnica di apprendimento supervisionato che coinvolge l'assegnazione di un'etichetta di classe a un'istanza in base alle sue caratteristiche. L'obiettivo è creare un modello in grado di generalizzare correttamente su nuovi dati, classificando in modo accurato le istanze in classi precedentemente osservate durante l'addestramento. Gli algoritmi di classificazione variano in complessità e adattabilità. Alcuni dei più comuni includono Alberi Decisionali, Random Forest, K Nearest Neighbors e Gradient Boosting. La scelta dell'algoritmo dipende spesso dalla natura dei dati e dalla complessità del problema di classificazione.

3.5.2 Regressione

La regressione è un'altra forma di apprendimento supervisionato che si concentra sulla previsione di un valore numerico anziché sull'assegnazione di classi. In altre parole, l'obiettivo della regressione è modellare la relazione tra le variabili di input e un valore di output continuo. Questo può includere la previsione di prezzi, temperatura, o qualsiasi altra variabile numerica. Gli algoritmi di regressione comprendono la Regressione Lineare, Regressione Logistica, Support Vector Regression, e Regressione Polinomiale. La scelta dell'algoritmo dipende dalla natura del problema e dalla complessità della relazione tra le variabili.

3.5.3 Alberi Decisionali

Gli alberi decisionali sono modelli di apprendimento supervisionato utilizzati per problemi di classificazione e regressione. L'idea principale è quella di suddividere iterativamente il *dataset*. Nelle sotto-sezioni successive discuteremo dell'idea di base dell'algoritmo e alcuni pro e contro.

Algoritmo

Per la creazione di un albero decisionale, possono essere impiegati diversi algoritmi di apprendimento automatico, tra i più comuni, ci sono *ID3* e *CART* (Classification and

Regression Trees) [19]. Prendendo come esempio il *CART*, Matematicamente, seleziona la soglia decisionale ottimale per una variabile in base al partizionamento ricorsivo, ovvero testando il valore potenziale di diverse soglie per ciascun predittore e implementando quello più prezioso finché un’ulteriore suddivisione non migliora più la discriminazione. In generale, un algoritmo per la creazione di un albero decisionale analizza un set di dati di addestramento e cerca di trovare la migliore sequenza di decisioni per classificare i dati. L’algoritmo utilizza una struttura ad albero per rappresentare una serie di decisioni che portano a una classificazione. Ogni nodo dell’albero rappresenta una decisione basata su una caratteristica dei dati, mentre le foglie rappresentano le possibili etichette a quella decisione. Nella Figura 3.4 è mostrato un esempio di albero decisionale generico, con il nodo radice e i suoi sotto-alberi. Ogni sotto-albero procede fino a raggiungere un nodo foglia, il quale rappresenta un’etichetta di classificazione.

Pro e Contro

Gli alberi decisionali sono ampiamente utilizzati per costruire modelli di classificazione poiché sono facili da interpretare e forniscono una maggiore trasparenza nel processo decisionale rispetto ai modelli *“black-box”* come le reti neurali artificiali. Questo li rende adatti per situazioni in cui è importante comprendere il ragionamento alla base delle decisioni del modello [20]. Tuttavia, gli alberi decisionali possono presentare problemi come l’*overfitting* e hanno una scarsa capacità di generalizzazione [19]. Alcuni di questi problemi possono essere affrontati utilizzando le random forest, un tipo di modello di apprendimento automatico basato sugli alberi decisionali, che verranno approfondite nella sezione successiva (Sezione 3.5.4).

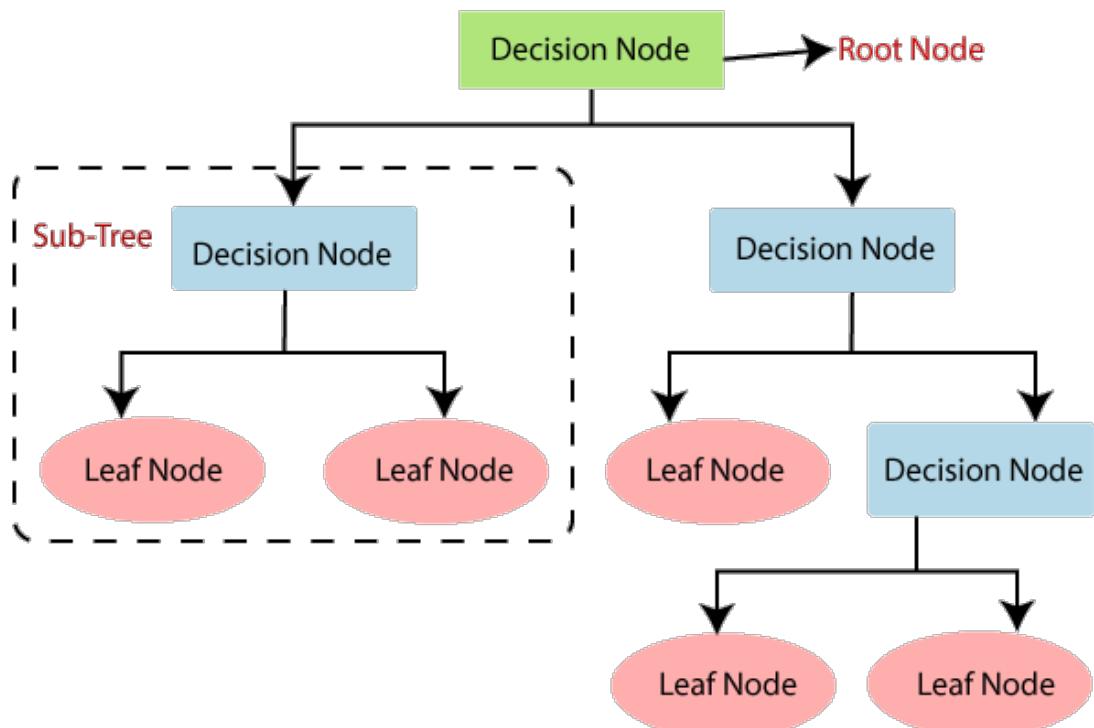


Figura 3.4. Struttura Albero Decisionale.

3.5.4 Random Forest

Dopo aver introdotto gli Alberi Decisionali (Sezione 3.5.3) e averne apprezzato la facile interpretabilità, è importante notare che il partizionamento ricorsivo può risultare instabile, specialmente con set di dati di dimensioni ridotte. Per contrastare questo problema, si sono sviluppati approcci come il *bagging* e il *boosting*. Il *bagging* è un metodo che crea insiemi utilizzando diversi sottoinsiemi di dati di addestramento, con un singolo algoritmo di apprendimento.

La Foresta

Le Random Forest rappresentano un'estensione degli alberi decisionali, progettata per la classificazione o la regressione. Per affrontare l'*overfitting* e la sensibilità ai piccoli cambiamenti nei dati di addestramento, utilizzano il "bagging" (*bootstrap aggregating*). In pratica, vengono addestrati diversi alberi decisionali su sottoinsiemi casuali del set di dati, formando così una foresta di alberi (la Figura 3.4 mostra un singolo albero). Le previsioni di questi alberi vengono combinate per ottenere una previsione finale. Questo approccio favorisce la diversità tra gli alberi, riducendo la dipendenza da singole caratteristiche e migliorando la capacità di generalizzazione del modello [20]. Nel caso della classificazione, ogni albero è un classificatore separato, e l'output finale è la classe maggioritaria (tutti gli alberi effettuano una classificazione, di queste la classe che si presenta più volte è l'output). Viceversa, per i problemi di regressione, si calcola la media delle singole predizioni per ottenere una stima statistica equivalente all'output finale.

Randomizzazione

Il concetto di "foresta" è soltanto uno degli aspetti evolutivi rispetto agli Alberi Decisionali. Un'ulteriore caratteristica risiede nell'utilizzo della randomizzazione per campionare il set di dati, implementata in due modalità principali. In primo luogo, il set di dati viene campionato con sostituzione, ciò significa che vengono estratti campioni casuali dai dati, e il processo di estrazione viene fatto più volte. Poiché il campionamento avviene con sostituzione, alcuni soggetti potrebbero comparire più volte, mentre invece altri potrebbero essere esclusi. Per quanto concerne la seconda implementazione, durante la costruzione di ciascun albero decisionale nella foresta, viene effettuata una randomizzazione nei nodi decisionali. In ogni nodo, un certo numero di variabili viene selezionato in modo casuale. Tipicamente, il numero di variabili scelti è la radice quadrata arrotondata del numero totale di *features* $\lfloor \sqrt{\# \text{features} + 1} \rfloor$ [21], o il logaritmo $\lfloor \log_2(\# \text{features} + 1) \rfloor$ [22], nel set di dati. L'algoritmo testa quindi tutte le possibili soglie per tutte le variabili selezionate, scegliendo la combinazione variabile-soglia che produce la migliore suddivisione dei dati.

Interpretabilità

Dato che un modello di Random Forest può essere visto come un modello complesso di Alberi Decisionali, ciò incide sull'interpretazione. Poiché la foresta casuale è composta da molti alberi, risulta più difficile individuare quali variabili influenzano maggiormente le previsioni del modello. Tuttavia, risultano più interpretabili rispetto a modelli "black-box" come le reti neurali artificiali [20]. Per agevolare la comprensione del modello, esistono diversi costrutti matematici utilizzati per misurare l'impatto delle variabili nel modello, chiamate "misure di importanza delle variabili". Queste

misure di importanza delle variabili sono strumenti che possono essere utilizzati per valutare e quantificare l'influenza di ciascuna variabile nel processo decisionale del modello Random Forest. L'uso di queste misure può fornire considerazioni sull'importanza relativa delle variabili nel contribuire alle predizioni del modello [21]. Ad esempio nella Figura 4.6 sono rappresentate le importanze delle *features* restituite dal modello Random Forest dopo l'addestramento. Questo grafico potrebbe fornire strumenti per studi approfonditi da parte degli astronomi. Permette di capire quali siano le caratteristiche che pesano di più nel processo decisionale del modello.

3.5.5 K Nearest Neighbors

L'algoritmo dei K Nearest Neighbors è una tecnica di apprendimento supervisionato ampiamente utilizzata per problemi di classificazione e regressione. L'idea di base è assegnare una classe ad un'istanza basandosi sulla maggioranza delle etichette delle sue "vicine" più prossime, dove la vicinanza è definita da una metrica di distanza, e il processo di classificazione coinvolge l'analisi delle etichette delle istanze più vicine.

Classificatore KNN

Per la classificazione, l'algoritmo KNN determina la classe di un'istanza d_i tramite l'Equazione 3.5:

$$y(d_i) = \operatorname{argmax}_k \sum_{x_j \in \text{KNN}} y(x_j, c_k) \quad (3.5)$$

Dove d_i rappresenta il campione da classificare, x_j è uno dei suoi k vicini più prossimi, e $y(x_j, c_k)$ indica se x_j appartiene alla classe c_k . In altre parole, la previsione sarà la classe che ha il maggior numero di membri nei k vicini più prossimi. Ad esempio se fissiamo $k = 3$ e uno dei vicini più prossimi di un'osservazione appartiene alla classe *Zero* e gli altri due appartengono alla classe *Uno*, allora si può concludere che il campione di test appartiene alla classe *Uno* [23].

Metrica di distanza

La distanza tra le osservazioni, fondamentale per il KNN, è spesso definita dalla metrica di distanza Minkowski, mostrata nell'Equazione 3.6:

$$\text{dminkowski}(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^q |x_i - x_j|^p \right)^{\frac{1}{p}} \quad (3.6)$$

Dove se $p = 2$ essa corrisponde alla distanza euclidea [24]. Nella sezione 3.7.1 vengono esplorate nel dettaglio anche la distanza euclidea (Equazione 3.16) e di Manhattan (Equazione 3.17)

Parametro K

Un altro concetto importante è il parametro k che rappresenta il numero di vicini considerati. La scelta appropriata di k ha un impatto significativo sulle prestazioni dell'algoritmo. La chiave per scegliere un valore k appropriato è trovare un equilibrio tra *overfitting* e *underfitting* [25]. Secondo [26], è stato proposto un limite superiore approssimato per K pari al vicino più prossimo di \sqrt{N} , dove N è il numero totale di campioni nel set di dati. Questa approssimazione è descritta dettagliatamente nella sezione 3.7.1 e rappresentata dall'equazione 3.15.

3.5.6 Gradient Boosting

I metodi *ensemble* basati su alberi hanno raggiunto ottimi risultati nel campo del Machine Learning, come ad esempio quelli che sfruttano la tecnica di *bagging* (e.g. Random Forest sezione 3.5.4) e di *boosting*, il quale genera modelli base in sequenza. La differenza principale tra i metodi di *bagging* e di *boosting* è che mentre il *bagging* aggrega diversi modelli utilizzando campionamenti casuali indipendenti, il metodo di *boosting* ricampiona strategicamente i dati focalizzandosi sulle informazioni più utili per ogni modello successivo [27]. Il Gradient Boosting è una tecnica di apprendimento supervisionato che costruisce un modello predittivo attraverso l'insieme sequenziale di modelli più deboli, in cui ogni modello successivo mira a correggere gli errori del modello precedente. Questo approccio è particolarmente efficace per migliorare le prestazioni dei modelli in termini di accuratezza.

Algoritmo

L'idea che c'è alla base dell'algoritmo risiede nel trovare una funzione che approssimi la relazione tra i dati di input e gli output del modello durante la fase di addestramento, minimizzando al contempo la funzione di perdita [22]. Considerando un insieme di campioni indipendenti e identicamente distribuiti, rappresentati da un set di dati etichettato, l'algoritmo mira a costruire un predittore F che assegna una risposta a ciascun possibile valore della variabile casuale indipendente X , che può essere discreta nel caso della classificazione o continua nel caso della regressione. L'approccio generale prevede l'utilizzo di un insieme di funzioni elementari H , spesso chiamate deboli (*base learners*), che vengono selezionate per minimizzare una specifica funzione di rischio empirico. La soluzione finale è ottenuta attraverso una combinazione lineare di queste funzioni, espressa come:

$$F(x) = \sum_{m=1}^M f_m(x) \quad (3.7)$$

dove f_m è scelto dall'insieme di funzioni elementari H [27]. È importante sottolineare che l'obiettivo non è solo trovare $F(x)$, ma anche minimizzare la funzione di costo associata.

Funzione di costo

La funzione di costo (*loss function*), descritta da $L(y, F(x))$ misura l'errore associato tra la predizione $F(x_i)$ e all'effettiva etichetta y_i . L'obiettivo principale è minimizzare questo errore, considerando possibili problematiche come l'*overfitting*. Nel caso della

regressione, una scelta comune per la funzione di costo è l'errore quadratico medio (Mean Squared Error, MSE), il cui calcolo è definito dall'equazione:

$$L(y, F(x)) = (y - F(x))^2 \quad (3.8)$$

Nel caso della classificazione binaria, una scelta comune per la funzione di costo è la *logit loss*, la cui equazione è espressa come:

$$L(y, F(x)) = \log(1 + e^{-yF(x)}) \quad (3.9)$$

dove log è il logaritmo in base due [27].

Estensioni

Il Gradient Boosting ha dato origine a diverse estensioni, le più famose sono il XGBoost LightGBM e CatBoost, che si concentrano sia sulla velocità che sulla precisione. Questi modelli sono ampiamente utilizzati grazie alla loro potenza predittiva. Tuttavia, è fondamentale gestire attentamente la complessità del modello per evitare l'*overfitting* [22].

3.6 Apprendimento Non Supervisionato

Nella sezione 3.5 abbiamo esaminato l'apprendimento supervisionato. Tuttavia, spesso non disponiamo di un dataset già etichettato, e l'etichettatura manuale può diventare un processo dispendioso. L'apprendimento non supervisionato rappresenta una branca fondamentale dell'apprendimento automatico in cui il modello è chiamato a scoprire modelli e relazioni intrinseche nei dati senza il supporto di etichette o obiettivi specifici. In questo caso, il problema non prevede un output noto iniziale. Il sistema analizza i dati in input e cerca autonomamente similitudini e regolarità tra i pattern, organizzando i dati in modo coerente. La tecnica più diffusa nell'apprendimento non supervisionato è il *clustering*, che riceve dati in input e li raggruppa in base a similitudini, utilizzando un'euristica definita a priori. Questo capitolo esplorerà alcune delle tecniche più comuni dell'apprendimento non supervisionato.

3.6.1 Clustering

Il *clustering* rappresenta la tecnica più comune nell'apprendimento non supervisionato, con diversi approcci e algoritmi disponibili. Una di queste è il clustering gerarchico, che organizza i dati in una struttura ad albero. Questa struttura può essere visualizzata tramite un dendrogramma, che mostra la gerarchia dei cluster e le relazioni di similarità tra di essi. Ci sono due approcci principali nel clustering gerarchico: agglomerativo, in cui i dati vengono combinati per formare cluster più grandi, e divisivo, in cui un cluster viene suddiviso in insiemi più piccoli. Un'alternativa è rappresentata dal K-means, che raggruppa i dati in cluster omogenei, basandosi su un criterio di similarità. In questo approccio, il *dataset* viene suddiviso in un numero predeterminato di cluster (K), e ogni dato è assegnato al cluster che meglio lo rappresenta. L'algoritmo cerca di minimizzare la varianza intra-cluster e massimizzare la varianza inter-cluster. Uno dei vantaggi del K-Means è la sua semplicità concettuale e la velocità di convergenza. Tuttavia, è importante notare che la scelta del numero ottimale di cluster (il parametro K) può influire sulle prestazioni del modello [28].

3.6.2 Self-Organizing Maps

Le mappe auto-organizzanti, o Self-Organizing Maps (SOM), rappresentano una forma particolare di rete neurale artificiale che adotta un approccio di apprendimento non supervisionato per organizzare i dati in una mappa bidimensionale, facilitando così la formazione di cluster. Differiscono dalle reti neurali artificiali (ANNs) tradizionali sia in termini di architettura che di proprietà algoritmiche, poiché si basano sull'apprendimento competitivo anziché sull'apprendimento correttivo.

Apprendimento

Durante la fase di addestramento di una SOM, viene attivato un nodo vincitore noto come Best Matching Unit (BMU), determinato in base a una misura di distanza come quella euclidea o di Manhattan. I pesi del BMU vengono poi aggiornati utilizzando un tasso di apprendimento e i dati dell'osservazione corrente. È importante notare che l'aggiornamento dei pesi coinvolge sia il BMU che un vicinato di neuroni. Se un neurone è nel vicinato del BMU, anche i suoi pesi saranno aggiornati. Questo processo consente alla mappa di auto-organizzarsi, posizionando i neuroni simili in regioni vicine e quelli più distanti in regioni separate. Ad esempio, se consideriamo l'utilizzo della

distanza euclidea per calcolare la distanza tra la i -esima osservazione e tutti i neuroni, il BMU sarà colui che ha la distanza minore, e l'equazione sarebbe la seguente:

$$\text{BMU} = \underset{d}{\operatorname{argmin}} \{ d_{\text{euclidea}}(\vec{x}_i, \vec{w}_{i,j}) \} \quad (3.10)$$

Trovato il BMU, si aggiornano i pesi sia di esso che del suo *vicinato*, seguendo la seguente equazione:

$$\vec{w}_{i,j}(\tau + 1) = \vec{w}_{i,j}(\tau) + \sigma \times (\vec{x}_i - \vec{w}_{i,j}(\tau)), \quad \text{dove } \tau = 1, 2, 3, \dots, n \quad (3.11)$$

Infine, ad ogni iterazione, il tasso di apprendimento e il raggio del vicinato decadono secondo le seguenti equazioni:

$$\sigma = \sigma_0 \times \exp\left(-\frac{\tau}{\lambda}\right), \quad \text{dove } \tau = 1, 2, 3, \dots, n \quad (3.12)$$

$$\alpha = \alpha_0 \times \exp\left(-\frac{\tau}{\lambda}\right), \quad \text{dove } \tau = 1, 2, 3, \dots, n \quad (3.13)$$

Dove \vec{x}_i rappresenta la i -esima osservazione cioè un vettore di n -features, $\vec{w}_{i,j}$ il i, j -esimo neurone che compone la mappa ed è anch'esso un vettore n -dimensionale, τ il passo dell'iterazione corrente (*step*), σ il tasso di apprendimento e α il raggio del vicinato.

Architettura

A differenza delle normali reti neurali artificiali (ANNs) (consulta sezione 3.3), i neuroni in una SOM non sono interconnessi. La struttura di una SOM è costituita da una mappa bidimensionale in cui ciascun neurone rappresenta un vettore n -dimensionale, con n che corrisponde alle *features* dei dati in input. In pratica, una SOM accetta in input un'osservazione caratterizzata da n *features*, e ciascun neurone nella mappa è un vettore n -dimensionale. La Figura 3.5 mostra l'architettura classica della SOM, con due dimensioni X e Y . Le diverse colorazioni dei neuroni evidenziano la creazione dei cluster. Lo strato di input è definito come *input vector*, il quale rappresenta il set di dati composto da n campioni (Equazione 3.14). Infine $\vec{w}_{i,j}$ rappresentano il vettore dei pesi che connettono il neurone i, j -esimo ad un campione.

$$\text{Input vector} = X_1, X_2, \dots, X_n \quad (3.14)$$

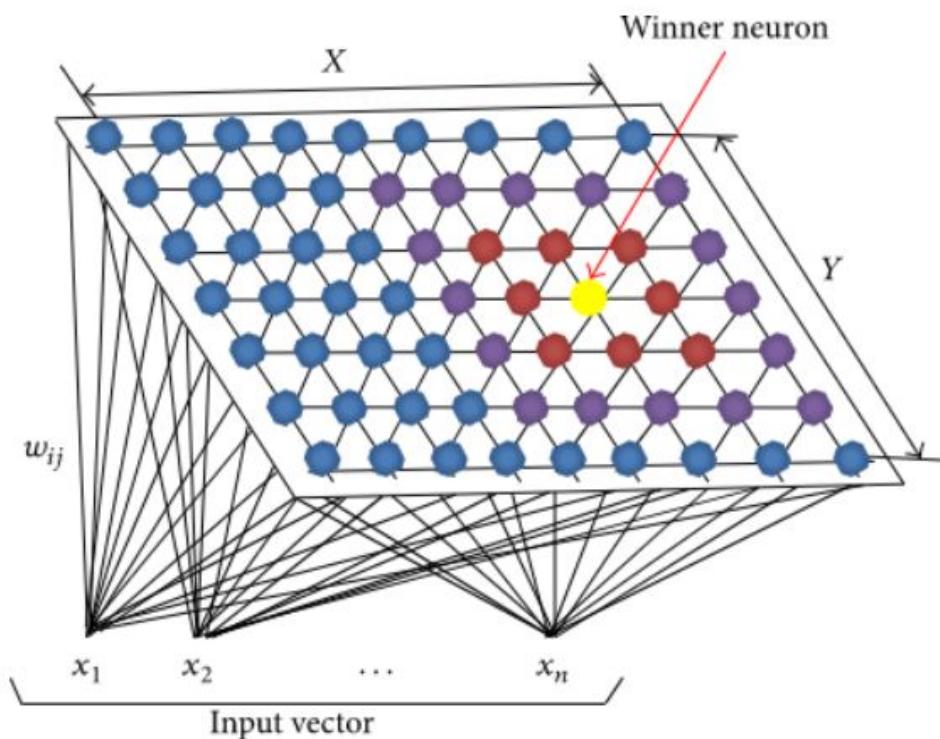


Figura 3.5. Architettura Self-Organizing Map.

In conclusione le SOM sono spesso utilizzate per visualizzare la struttura di dati ad alta dimensionalità in una mappa bidimensionale. Sono particolarmente efficaci per la riduzione della dimensionalità e la visualizzazione dei dati. L'auto-organizzazione in cluster della mappa costruisce una sorta di diagramma di similarità dei modelli, rendendo visibili delle relazioni topografiche dei dati, difatti essa preserva la topologia dei dati. Inoltre se i dati con il quale si è addestrata la mappa appartengono a determinate classi, i neuroni che la compongono possono essere calibrati in tali classi. Un elemento di input sconosciuto viene quindi classificato in base a quel neurone, il cui nodo è più simile ad esso in alcune metriche (e.g. distanza euclidea) utilizzate nella costruzione [29].

3.7 Elaborazione dei Dati

Spesso nell’ambito del Machine Learning ci si ritrova a lavorare con un numero di dati quantitativamente molto grande, i quali si possono presentare in svariate forme, come immagini, testo naturale, o anche numeri. La fase della pre-elaborazione (o *pre-processing*) dei dati, che precede la fase di addestramento, è molto importante e permette di ovviare a possibili problematiche di varia natura, come ad esempio *overfitting* o tempi di addestramento molto elevati. L’obiettivo principale di questa fase è ottenere un *dataset* appropriato ed efficiente per uno specifico caso d’uso. La pre-elaborazione dei dati non si limita solo a questo. I ricercatori prestano sempre più attenzioni alla pre-elaborazione dei dati come strumento per migliorare i loro modelli. La maggior parte delle tecniche di apprendimento automatico si basano su un set di dati presumibilmente completo o privo di rumore. Tuttavia, i dati del mondo reale sono spesso lontani dall’essere puliti o completi. Nella pre-elaborazione dei dati è comune impiegare tecniche per rimuovere i dati rumorosi o per imputare (riempire) i dati mancanti. Quando i *dataset* diventano grandi in termini di numero di caratteristiche, possono sorgere problemi come l’*overfitting*. Inoltre, le elevate dimensioni possono ostacolare il funzionamento di molti algoritmi di ML. Tuttavia, esistono tecniche come la selezione delle caratteristiche o l’estrazione delle caratteristiche per affrontare queste sfide. Nei modelli di Machine Learning che producono output qualitativi, come i classificatori, è cruciale avere un *dataset* con classi bilanciate, altrimenti se si considera un *dataset* sbilanciato si può avere un tasso di classificazione errato, che è più elevato per le istanze delle classi minoritarie. La disparità tra le classi può essere gestita mediante strategie come il sotto-campionamento o il pesaggio delle classi, esistono anche tecniche più avanzate di *data augmentation*, come il sovra-campionamento, le Generative Adversarial Networks (GAN) o modelli di diffusione [30].

3.7.1 Gestione dei valori mancanti

Uno dei problemi principali nell’elaborazione di *dataset* risiede nella gestione dei valori mancanti di cui le osservazioni sono affette. Questo problema riguarda soprattutto i *dataset* contenenti osservazioni aventi un elevato numero di caratteristiche (centinaia o migliaia). In questo caso, la presenza di valori mancanti in almeno una di queste caratteristiche è diventato un problema inevitabile [31]. Tipicamente, si usa rimuovere queste osservazioni in fase di *pre-processing* del *dataset*, oppure utilizzare la tecnica dello zero-*filling* per sostituire i valori mancanti. Altri approcci prevedono l’utilizzo di modelli complessi, e.g., neural networks [32] random forests [33], decision trees [34], low-rank matrix factorization [35]. Lo scopo di questo lavoro è quello di preservare la maggior parte delle osservazioni cercando al contempo di non minarne l’affidabilità rimpiazzando i dati mancanti con valori non consistenti. Per raggiungere questo obiettivo, proponiamo i seguenti metodi per gestire i casi in cui la generica osservazione presenta dei valori mancanti in alcune delle sue *features*.

Metodo 1: K-Nearest Neighbors

Dato l’elemento i -esimo per il quale la j -esima caratteristica è mancante, si vuole utilizzare il metodo dei K-Nearest Neighbors (KNN) per attribuire un valore all’elemento i, j della matrice dei dati D . L’applicazione di questo metodo richiede la determinazione di tre iperparametri: il valore di K , il metodo per calcolare la distanza tra le osservazioni e il metodo di attribuzione del valore.

Fissare il valore di K . Il primo *step* consiste nel fissare sperimentalmente il valore dell’iperparametro K , che rappresenta il numero dei K elementi di D più vicini all’osservazione i . [36, 37] hanno fissato il limite superiore di K come la radice quadrata del numero di osservazioni nel *dataset*. Sulla base di questi lavori, [26] hanno approssimato il limite superiore di K al vicino più prossimo di \sqrt{N} e successivamente fissato il seguente range di valori entro cui cercare il valore ottimale di K :

$$\{2q+1 \mid q \in \mathbb{N}, 0 \leq q \leq \frac{\sqrt{N}-1}{2}\} \quad (3.15)$$

Questo range contiene tutti i numeri dispari da 0 a \sqrt{N} .

Calcolo della distanza. Una volta fissato K , si procede con la determinazione dei K elementi più vicini. La “vicinanza” tra due elementi è misurata sulla base della distanza calcolata tra essi. Tra le varie metodologie, le due più comuni sono la distanza euclidea (Equazione (3.16)) e la distanza di Manhattan (Equazione (3.17)). Date due osservazioni \mathbf{x}_i e \mathbf{x}_j , le due distanze sono definite come segue:

$$\text{deuclidean}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{p=1}^M w_p (\mathbf{x}_{i,p} - \mathbf{x}_{j,p})^2} \quad (3.16)$$

$$\text{dmanhattan}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{p=1}^M w_p |\mathbf{x}_{i,p} - \mathbf{x}_{j,p}| \quad (3.17)$$

dove w_p è il peso normalizzato della p -esima *feature*.

Attribuzione del valore. Dopo aver calcolato le distanze, è possibile definire i K vicini all’osservazione i -esima per cui si desidera attribuire un valore alla j -esima *feature*. Per attribuire un valore, oltre ai classici metodi di calcolo della media o del mediano, si possono implementare metodi come inverse distance weighted mean (ID-WM) [38] e inverse rank weighted mean (IRWM) [39, 40]. L’IDWM (Equazione 3.18) pesa maggiormente l’influenza dei vicini più prossimi secondo la seguente formula:

$$\hat{f}' = \frac{\sum_{k=1}^K \frac{1}{\delta + d(\mathbf{x}_k, \mathbf{x}')} f_k}{\sum_{k=1}^K \frac{1}{\delta + d(\mathbf{x}_k, \mathbf{x}')}} \quad (3.18)$$

dove \hat{f}' è il valore stimato (elemento i, j), $d(\mathbf{x}_k, \mathbf{x}')$ è la distanza pesata tra \mathbf{x}' e il vicino \mathbf{x}_k , e δ è una costante settata a 10^{-6} [26]. Inoltre, \mathbf{x}' è l’osservazione per cui il valore mancante deve essere calcolato, f_k è il valore della *feature* corrispondente ad \mathbf{x}_k . Analogamente, l’IRWM (Equazione 3.19) è formulato in modo che i vicini più prossimi siano più influenti. In questo caso, f_k è basato sulla corrispondente distanza $d(\mathbf{x}_k, \mathbf{x}')$ in ordine ascendente. La stima finale del valore mancante secondo IRWM è definita come segue:

$$\hat{f}' = \frac{\sum_{k=1}^K (K - k + 1) f_k}{\sum_{k=1}^K k} \quad (3.19)$$

Metodo 2: Pesare l'importanza delle *features*

Questo metodo tratta in modo diverso i valori mancanti (identificati da valori *not-a-number*), attribuendo maggiore peso alle caratteristiche più rilevanti per la classificazione. L'obiettivo è quello di eliminare la generica riga se il valore mancante non si presenta in una delle *features* altamente rilevanti in termini di classificazione. In caso contrario, il valore mancante sarà sostituito utilizzando la stessa strategia definita in metodo 3.7.1. Il calcolo della rilevanza di ogni *feature* in termini di classificazione richiede i seguenti passi. In *primis*, è necessario rimuovere tutte le righe aventi almeno un *not-a-number*, ottenendo la matrice dei dati senza valori inconsistenti. Si lancia un *grid search* sulla matrice ripulita dai *not-a-number* al fine di ottenere la lista dei valori sub-ottimali su cui addestrare il Random Forest (RF). Terminato il *grid search*, si addestra il RF. Il processo di training genererà in output uno score di importanza associato ad ogni *feature* di input. L' i -esimo score rappresenta il potere discriminativo della *feature* f_i . In altre parole, questo score consiste nella rilevanza della i -esima *feature* sulla classificazione dei campioni. Ottenuto lo score per ogni *feature*, si analizzano tutte le colonne in cerca di valori mancanti. Per ogni valore mancante individuato nella colonna j , si rimuove tutta la riga della matrice se la colonna j non è nelle top T *features*. Altrimenti, se la colonna $j > T$ presenta valori mancanti, questi si sostituiscono con il metodo definito in sezione 3.7.1. Il valore di T è un iperparametro e determina il numero delle colonne che vogliamo considerare più rilevanti in funzione della classificazione.

3.7.2 Selezione delle Caratteristiche

La tecnica di *Features Selection* ha l'obiettivo di selezionare le caratteristiche che hanno maggiore rilevanza nella fasi di apprendimento e predizioni e scartare quelle meno importanti senza degradare le prestazioni [41]. Per cui, una volta attribuita un'importanza ad ogni caratteristica (e.g con l'utilizzo dell'algoritmo Random Forest), sulla base di una soglia T vengono eliminate alcune di esse.

3.7.3 Normalizzazione dei Dati

Nonostante non costituisca un prerequisito necessario nei modelli di Machine Learning, il processo di normalizzare i dati è tipicamente impiegato con l'obiettivo di:

1. standardizzare l'intervallo di valori di tutte le caratteristiche del *dataset*;
2. migliorare la robustezza numerica degli algoritmi impiegati.

Ad esempio, nell'insieme di dati in analisi (Cumulative KOI Table) è definita una colonna corrispondente alla temperatura di equilibrio della stella, espressa in gradi Kelvin. Si nota che il valore minimo riscontrato in questa colonna è di 25° K, mentre il valore massimo supera i 14.000° K. Analogamente, una seconda colonna presente nel *dataset* rappresenta la gravità superficiale stellare, con un valore minimo registrato pari a 0,047 e un massimo di 5,364. La disparità significativa tra le scale di queste variabili potrebbe condurre a imprecisioni in termini di calcolo di distanza tra le osservazioni. Pertanto, la procedura di normalizzazione dei dati verrebbe impiegata per uniformare la scala di tutte le caratteristiche, spesso attraverso la trasformazione degli intervalli di valori in un intervallo compreso tra zero e uno. Nel suo lavoro, Armstrong [42] ha scelto di normalizzare i dati affinché essi abbiano una distribuzione normale con media 0 e varianza unitaria. Vengono quindi proposti due metodi che sono:

Metodo 1: Z-Score

Il metodo Z-Score è una tecnica di normalizzazione che misura il numero di deviazioni standard di un dato rispetto alla media del set di dati. Questo metodo è utile per ridurre l'impatto di valori estremi o *outliner*, garantendo che i dati siano rappresentati in una scala comune. Il punteggio Z di un dato x nel contesto della distribuzione può essere calcolato usando la seguente formula:

$$Z = \frac{x - \mu}{\sigma} \quad (3.20)$$

dove:

- x è il valore specifico dei dati;
- μ rappresenta la media del set di dati;
- σ è la deviazione standard del set di dati.

Metodo 2: Normalizzazione [0, 1]

La normalizzazione [0, 1] è un processo mediante il quale i dati vengono trasformati in un intervallo compreso tra 0 e 1. Questo metodo è particolarmente utile quando le *feature* del *dataset* hanno scale diverse. Normalizzare i dati in questo intervallo può migliorare la stabilità degli algoritmi di Machine Learning, specialmente quelli sensibili alle differenze di scala tra le *feature*. La normalizzazione [0, 1] può essere eseguita utilizzando la seguente formula:

$$X_{\text{normalizzato}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \quad (3.21)$$

dove:

- x è il valore specifico dei dati;
- X_{\min} è il valore minimo del set di dati;
- X_{\max} è il valore massimo del set di dati.

3.7.4 Bilanciamento del set di dati

Il bilanciamento delle classi è importante per il buon funzionamento dei modelli di Machine Learning. Se ci sono meno esempi di una categoria rispetto all'altra, la macchina può imparare meno su quella minoritaria. Non è sempre un problema, ma diventa critico se ci sono troppo pochi esempi della categoria meno comune. Una tecnica semplice ed efficace è il *sovra campionamento casuale* [43]. Esistono anche tecniche che non generano dati sintetici, ma rimuovono, o assegnano alle classi un peso diverso, noti come il *sotto campionamento*, o il *Pesaggio delle classi* [44]. Per mantenere una certa qualità del *dataset* proponiamo questi ultimi due metodi.

Metodo 1: Sotto Campionamento

Il sotto campionamento è una tecnica utilizzata per bilanciare un *dataset* riducendo il numero di istanze della classe maggioritaria. Questo approccio è spesso adottato quando ci sono disparità significative tra le dimensioni delle classi nel *dataset*. Il sotto campionamento può essere implementato rimuovendo casualmente alcune istanze della classe maggioritaria. Questo processo mira a migliorare le prestazioni del modello riducendo il rischio di favorire la classe maggioritaria durante l'addestramento.

Metodo 2: Pesaggio delle Classi

Il pesaggio delle classi è un'alternativa al sotto campionamento e sovra campionamento, che assegna pesi diversi alle classi durante l'addestramento del modello. Questo approccio è utile quando non si desidera modificare la dimensione effettiva del *dataset*, ma si vuole influenzare la percezione del modello rispetto alle classi. Considerando solo due classi, di solito, si assegna un peso maggiore alla classe minoritaria e un peso minore alla classe maggioritaria, lo stesso vale nel caso in cui ci siano più di due classi. La formula generale per il peso delle classi è la seguente:

$$W_i = \frac{\text{Numero totale di campioni}}{\text{Numero di classi} \times \text{Numero di campioni della classe specifica}} \quad (3.22)$$

dove W_i è il peso da attribuire alla i -esima classe.

W_i quindi viene utilizzato durante l'addestramento per penalizzare maggiormente gli errori sulla classe minoritaria, contribuendo così a mantenere un equilibrio tra le classi nel processo di apprendimento.

Capitolo 4

Risultati sperimentali

Durante la fase sperimentale, sono stati condotti diversi test esplorando vari approcci, sia supervisionati che non supervisionati. I test sono stati eseguiti considerando i *dataset* di Kepler e TESS separatamente, e i principali test sono stati condotti sul *dataset* creato dall’intersezione dei due cataloghi. Le analisi effettuate su ciascun *dataset* sono state fondamentali per comprendere la natura dei dati e valutare la capacità dei modelli di apprendere in base alle etichette associate a ciascun campione. Questo processo ci ha consentito di valutare l’importanza delle caratteristiche e di ottimizzare i dati per migliorare le metriche del modello. Questo capitolo illustra la fase di pre-elaborazione dei dati, quella di addestramento e dei risultati ottenuti.

4.1 Set di dati

Definito l’obiettivo principale di automatizzare il processo di validazione dei dati provenienti dal telescopio spaziale TESS, si è analizzato il catalogo di dati provenienti dall’archivio di ExoFOP [45]. Questi dati sono classificati come oggetti di interesse in quanto vengono elaborati inizialmente da centri come SPOC o QLP, notificati per osservazioni di *follow-up* e resi successivamente disponibili. Il *dataset* risultante è composto da 6,977 osservazioni totali, di cui 5,422 Veri Pianeti e 1,555 Falsi Pianeti, rendendolo fortemente sbilanciato a favore dei Veri Pianeti. Questo squilibrio ha motivato l’utilizzo di un ulteriore catalogo proveniente dal telescopio Kepler [46], composto da 9,564 osservazioni totali, equamente distribuite tra 4,725 Veri Pianeti e 4,839 Falsi Pianeti, e quindi favorevole al nostro scopo. I primi test sono stati condotti considerando i set di dati singolarmente per valutare le risposte dei modelli. Successivamente, il secondo obiettivo consiste nella creazione di un *dataset* contenente TCEs individuati sia nei dati Kepler che TESS. Per fare ciò, sono stati considerati i cataloghi di Kepler TCEs e di TESS TCEs.

Siano:

$$\mathcal{D}K = Xk_1, \dots, Xk_n \quad (4.1)$$

il generico catalogo di n TCEs individuati nei dati Kepler, dove

$$Xk_{i=1, \dots, n} = xk_1, \dots, xk_\ell \quad (4.2)$$

rappresenta l’ i -esimo TCE Kepler caratterizzato da ℓ *features*.

Dato un catalogo di m TESS TCEs:

$$\mathcal{D}T = XT_1, \dots, XT_m \quad (4.3)$$

dove

$$XT_{i=1,\dots,m} = xt_1, \dots, xt_d \quad (4.4)$$

È stato creato un catalogo $\mathcal{D}_{KT} = \mathcal{D}K \cup \mathcal{D}T$ contenente gli $n + m$ TCEs. Tuttavia, nel processo di unione, è necessario considerare solo le *features* comuni tra $\mathcal{D}K$ e $\mathcal{D}T$ per ogni TCE, sia esso appartenente al catalogo Kepler o TESS. Per esempio, considerando una *feature* del catalogo Kepler, xk_j = Orbital Period [days], questa *feature* sarà inclusa per ogni TCE del nuovo catalogo ($XKT_p \in DKT$) solo se è una *feature* condivisa anche dai TCEs del catalogo TESS, ottenendo così un set di dati unito tra i due cataloghi composto da 16,541 osservazioni totali, di cui 10,147 Veri Pianeti e 6,394 Falsi Pianeti. Alla fine di questo processo, si ottengono tre *dataset* distinti: uno relativo esclusivamente ai dati del telescopio TESS, un secondo relativo solo ai dati del telescopio Kepler e un terzo, chiamato TESS/Kepler, ottenuto dall’intersezione dei due cataloghi.

4.2 Elaborazione dei dati

Ogni TCE che costituisce il *dataset* è categorizzato come ”oggetto di interesse”, poiché centri come SPOC e QLP convalidano questi dati etichettandoli come tali (consultare la sezione 2.4.2). Questa convalida da parte di tali centri ci assicura una certa qualità del dataset. Tuttavia, è necessario sottoporli a procedure specifiche di elaborazione, le quali saranno dettagliate in questo capitolo.

4.2.1 Set di dati: TESS

Nel catalogo di TESS, composto da 6,977 osservazioni, ognuna con 62 caratteristiche, si evidenzia il problema dei ”*not-a-number*” (NaN). Infatti, si contano 38,201 NaN su un totale di 432,574, ovvero $\approx 8,83\%$

Sostituzione etichette

Di queste 62 caratteristiche, una di esse rappresenta una disposizione che assegna una classificazione all’ i -esima osservazione [47], le quali sono: *APC*(Candidato Planetario Ambiguo), *CP*(Pianeta Confermato), *FA*(Falso Allarme), *FP*(Falso Positivo), *KP*(Pianeta Conosciuto) e *PC*(Candidato Planetario). Con l’obiettivo preposto si mappano nel seguente modo:

- KP, CP, PC → 1
- APC, FA, FP → 0

Rimozione delle caratteristiche

Dopo la mappatura delle etichette, sono state analizzate tutte le caratteristiche non numeriche, tra cui: *Previous CTOI*, *TESS Disposition*, *Source*, *Detection*, *RA*, *Dec*, *Sectors*, *Date TOI Alerted (UTC)*, *Date TOI Updated (UTC)*, *Date Modified* e *Comments*. Queste colonne, insieme alle colonne di identificazione (*TOI*, *TIC ID*, *Pipeline Signal ID*), alle osservazioni di follow-up che sono: *Imaging Observations*, *Spectroscopy Observations*, *Time Series Observations*, *SG3*, *SG5*, *SG2*, *SG4*, *Master*, *SG1B* e *SG1A* (vedi sezione 2.4.2) e alcune features risultate anomale (*Epoch (BJD)* e *Epoch*

(*BJD err*), sono state eliminate. Complessivamente, sono state rimosse 26 colonne, riducendo il numero di caratteristiche a 36, di cui una è la disposizione.

Normalizzazione dei dati

Nonostante non costituisca un prerequisito necessario nei modelli di Machine Learning, il processo di normalizzare i dati è tipicamente impiegato quando entrano in gioco le distanze (e.g. euclidea, Manhattan ecc.), questo poiché esse risultano essere sensibili a grandi scali di dati. Per fare ciò viene utilizzato il metodo di normalizzazione in un range prefissato, che in questo caso è [0, 1] (vedi sezione 3.7.3).

Gestione dei valori mancanti

Nonostante le operazioni precedenti abbiano ridotto il numero di NaN, è fondamentale affrontare con precisione questo problema, dato l'importanza della qualità dei dati durante la fase di addestramento. Inizialmente, sono stati eliminati 98 NaN presenti nelle etichette, comportando la rimozione di 98 campioni. Successivamente, sono state esaminate le colonne composte esclusivamente da valori nulli; è emerso che solo la colonna *Planet Name* presentava questa caratteristica e quindi è stata eliminata. In seguito, sono state analizzate le colonne per il numero di valori mancanti, applicando una soglia di 5,000 ($T=5,000$). Ad esempio, se la colonna j -esima presentava più di T NaN, questa è stata eliminata. Sono state quindi rimosse *Stellar Metallicity* e *Stellar Metallicity err*. Questo processo è stato iterato anche sulle righe, con una soglia di 10 ($T=10$), portando all'eliminazione di 325 campioni su un totale di 6,879. Alla fine di questa fase, sono rimaste 6,554 osservazioni in totale, con un significativo miglioramento della percentuale di NaN, ridotta da $\approx 12,86\%$ a $\approx 4,10\%$.

K Nearest Neighbors

Per garantire che non ci siano NaN durante l'addestramento, è stata utilizzata la tecnica K Nearest Neighbors (vedi sezione 3.7.1) per l'imputazione dei valori mancanti, risolvendo definitivamente il problema dei *not-a-number*.

La Tabella 4.1 riporta in sintesi le caratteristiche del set di dati TESS dopo l'elaborazione.

	Totale Osservazioni	Veri Pianeti	Falsi Pianeti	Caratteristiche
TESS	6,554	5,199	1,355	32

Tabella 4.1. Caratteristiche del set di dati TESS dopo l'elaborazione.

4.2.2 Set di dati: Kepler

Nel catalogo di Kepler, composto da 9,564 osservazioni, ognuna con 141 caratteristiche, si evidenzia, anche qui, il problema dei *"not-a-number"* (NaN). Infatti, si contano 237,116 NaN su un totale di 1,348,524, ovvero $\approx 17,58\%$

Sostituzione etichette

Di queste 141 caratteristiche, una di esse rappresenta una disposizione che assegna una classificazione all’ i -esima osservazione [48], le quali sono: *CONFIRMED*, *CANDIDATE*, *FALSE POSITIVE*. Con l’obbiettivo preposto si mappano nel seguente modo:

- CONFIRMED, CANDIDATE → 1
- FALSE POSITIVE → 0

Rimozione delle caratteristiche

Dopo la mappatura delle etichette, sono state analizzate tutte le caratteristiche non numeriche, tra cui: *kepoi_name*, *kepler_name*, *koi_vet_stat*, *koi_vet_date*, *koi_pdisposition*, *koi_disp_prov*, *koi_comment*, *koi_fittype*, *koi_limbdark_mod*, *koi_parm_prov*, *koi_tce_delivname*, *koi_quarters*, *koi_trans_mod*, *koi_datalink_dvr*, *koi_datalink_dvs* e *koi_sparprov*. Queste colonne, insieme alle colonne di identificazione (*rowid* e *kepid*) e alle osservazioni di follow-up (*koi_score*, *koi_fpflag_nt*, *koi_fpflag_co*, *koi_fpflag_ss*, *koi_fpflag_ec*), sono state eliminate. Complessivamente, sono state rimosse 23 colonne, riducendo il numero di caratteristiche a 118, di cui una è la disposizione.

Normalizzazione dei dati

Anche in questo caso viene utilizzato il metodo di normalizzazione in un range prefissato, che è [0, 1] (vedi sezione 3.7.3).

Gestione dei valori mancanti

Utilizzando le stesse procedure che con TESS, sono state esaminate le colonne composte esclusivamente da valori nulli, riscontrando che *koi_eccen_err1*, *koi_eccen_err2*, *koi_longp*, *koi_longp_err1*, *koi_longp_err2*, *koi_ingress*, *koi_ingress_err1*, *koi_ingress_err2*, *koi_sma_err1*, *koi_sma_err2*, *koi_incl_err1*, *koi_incl_err2*, *koi_teq_err1*, *koi_teq_err2*, *koi_model_dof*, *koi_model_chisq*, *koi_sage*, *koi_sage_err1* e *koi_sage_err2* presentano tale caratteristica e quindi sono state rimosse. Successivamente, sono state analizzate le colonne per il numero di *not-a-number*, applicando una soglia di 600 (T=600) (vedi esempio in sezione 4.2.1), e sono state eliminate *koi_bin_oedp_sig*, *koi_max_sngle_ev*, *koi_max_mult_ev*, *koi_num_transits*, *koi_fwm_stat_sig*, *koi_fwm_prao*, *koi_fwm_prao_err*, *koi_fwm_pdeco*, *koi_fwm_pdeco_err* e *koi_zmag*. Questo processo è stato iterato anche per le righe, con una soglia di 10 (T=10), portando all’eliminazione di 956 campioni su un totale di 9,564. Alla fine di questa fase, sono rimaste 8,608 osservazioni in totale, con una significativa miglioramento della percentuale di *not-a-number*, ridotta da $\approx 17,58\%$ a $\approx 0,21\%$.

K Nearest Neighbors

Per garantire che non ci siano NaN durante l’addestramento, è stata utilizzata la tecnica K Nearest Neighbors (vedi sezione 3.7.1) per l’imputazione dei valori mancanti, risolvendo definitivamente il problema dei *not-a-number*.

Selezione delle caratteristiche

Infine, considerando la complessità del set di dati di Kepler, che, anche dopo le ultime operazioni, è composto da 88 *features*, per cui è stato adottato il metodo descritto nella sezione 3.7.2 per la selezione delle caratteristiche. È stata calcolata una soglia pari a 0 (T=0), portando all'eliminazione di *koi_eccen*, *koi_ldm_coeff4*, e *koi_ldm_coeff3*, riducendo così il numero di caratteristiche a 85, oltre a una per le etichette.

La Tabella 4.2 riporta in sintesi le caratteristiche del set di dati Kepler dopo l'elaborazione.

	Totale Osservazioni	Veri Pianeti	Falsi Pianeti	Caratteristiche
Kepler	8,608	4,527	4,081	85

Tabella 4.2. Caratteristiche del set di dati Kepler dopo l'elaborazione.

4.2.3 Set di dati: TESS/Kepler

Infine, anche il *dataset* combinato (TESS/Kepler) è stato sottoposto a un'elaborazione dei dati seguendo le stesse procedure. Dopo aver sostituito le etichette nei singoli *dataset* (consultare le sezioni 4.2.1 e 4.2.1), si è proceduto con l'unione, considerando solo le caratteristiche in comune come spiegato nella sezione 4.1. Queste caratteristiche sono 27, di cui una rappresenta l'etichetta, ottenendo così 16,541 campioni in totale. Anche se in percentuale ridotta, emerge il problema dei "not-a-number" (NaN). Infatti, si contano 48,685 NaN su un totale di 430,066, corrispondenti a $\approx 11,32\%$.

Pre-normalizzazione

Prima di procedere con la normalizzazione dei dati, sono state eliminate 98 osservazioni dal *dataset* TESS che presentavano NaN nelle etichette. Successivamente, sono state escluse le caratteristiche non numeriche, come *RA* e *Dec*, e alcune *features* che, durante la fase di sviluppo, sono risultate anomale, ovvero *Epoch (BJD)* e *Epoch (BJD) err*. Ciò ha portato a un totale di 22 caratteristiche, oltre a una per le etichette. È importante notare che in questo caso non sono state eliminate osservazioni di *follow-up* o caratteristiche di identificazione. Ciò è dovuto al fatto che, durante l'operazione di intersezione, rimangono solo le caratteristiche comuni, escludendo quelle non condivise come le osservazioni di *follow-up* o le caratteristiche di identificazione.

Normalizzazione dei dati

Anche in questo caso viene utilizzato il metodo di normalizzazione in un range prefissato, che è [0, 1] (vedi sezione 3.7.3).

Gestione dei valori mancanti

Applicando le medesime procedure utilizzate con TESS e Kepler, sono state esaminate le colonne composte esclusivamente da valori nulli, riscontrandone nessuna. Successivamente, sono state analizzate le colonne per il numero di *not-a-number*, applicando

una soglia di 2,400 ($T=2,400$) (vedi esempio in sezione 4.2.1), eliminando così *Stellar Radius (R_Sun) err*, *Stellar Radius (R_Sun)*, *Stellar Metallicity err*, *Stellar Metallicity* e *Stellar log(g) (cm/s 2) err*. Questo processo è stato iterato anche per le righe, con una soglia di 1 ($T=1$), portando all’eliminazione di 2,645 campioni su un totale di 16,443. Alla fine di questa fase, sono rimaste 13,798 osservazioni in totale, e risolvendo definitivamente il problema dei *not-a-number*, senza l’ausilio di tecniche come K Nearest Neighbors, preservando la qualità del set di dati.

La Tabella 4.3 riporta in sintesi le caratteristiche del set di dati TESS/Kepler dopo l’elaborazione.

	Total Osservazioni	Veri Pianeti	Falsi Pianeti	Caratteristiche
TESS/Kepler	13,798	8,396	5,402	17

Tabella 4.3. Caratteristiche del set di dati TESS/Kepler dopo l’elaborazione.

4.2.4 Caratteristiche dei set di dati elaborati

La tabella 4.4 riporta in sintesi le caratteristiche dei set di dati ottenute dopo la fase di elaborazione.

	Total Osservazioni	Veri Pianeti	Falsi Pianeti	Caratteristiche
TESS	6,554	5,199	1,355	32
Kepler	8,608	4,527	4,081	85
TESS/Kepler	13,798	8,396	5,402	17

Tabella 4.4. Caratteristiche dei set di dati elaborati.

4.3 Apprendimento Supervisionato

Dopo aver ottenuto i *dataset*, inizialmente, sono stati esplorati vari algoritmi basati su un approccio supervisionato, tra cui Alberi decisionali, Random Forest, K-Nearest Neighbors e Gradient Boosting (consultare la sezione 3.5), i quali sono stati testati su tutti e tre i set di dati.

4.3.1 Set di dati: TESS

Il *dataset* di TESS mostra un significativo sbilanciamento a favore dei Veri Pianeti, con un numero limitato di Falsi Pianeti, pari a 1,355 come indicato nella tabella 4.4. Tuttavia, è stato preso in considerazione il metodo del pesaggio delle classi, descritto nella sezione 3.7.4, calcolando un peso per la classe *Veri Pianeti* pari a 0,63 e per la classe *Falsi Pianeti* pari a 2,41. Questa disparità tra i due valori è dovuta dal forte sbilanciamento presente. Successivamente, il *dataset* è stato suddiviso, destinando l’80% all’addestramento e il 20% al test.

Selezione del modello

In generale, ci sono diversi modi per migliorare le prestazioni di un modello di Machine Learning. Uno di questi è l'ottimizzazione dei parametri, basata sulla ricerca dei valori migliori per affrontare problemi specifici. Un approccio comune è l'uso di una griglia di ricerca, che determina i parametri ottimali valutando le metriche del modello [49]. In questo specifico caso, sono stati ottimizzati i parametri più importanti seguendo un lavoro precedente [50]. I parametri ottimizzati includono il numero di alberi nella foresta, il numero di caratteristiche da considerare, la profondità massima dell'albero e il numero minimo di campioni richiesti per dividere un nodo interno per Random Forest, la profondità massima dell'albero e il numero minimo di campioni richiesti per dividere un nodo interno per l'albero decisionale, il numero dei vicini più vicini da considerare per K-Nearest Neighbors, e il numero di fasi di potenziamento e il tasso di apprendimento per Gradient Boosting.

Risultati

Dopo l'ottimizzazione dei parametri del modello, prosegue la fase di addestramento. Nella Tabella 4.5 sono riportati i risultati ottenuti dai vari modelli. La matrice di confusione della Random Forest è visualizzata nella Figura 4.1. Analizzando la matrice di confusione, emerge che sono stati erroneamente classificati 188 campioni come Veri Pianeti, quando in realtà sono Falsi Pianeti. Inoltre, questo numero è più elevato rispetto ai campioni correttamente classificati come Falsi Pianeti. Infine, le importanze delle caratteristiche restituite dal modello Random Forest sono illustrate nella Figura 4.2.

	Accuratezza (%)	F1 Score (%)	Precisione (%)	Recall (%)
Albero Decisionale	79,32	87,05	86,02	88,10
Random Forest	83,06	90,00	84,17	96,71
K Nearest Neighbors	80,93	88,97	81,76	97,58
Gradient Boosting	83,98	90,46	85,27	96,32

Tabella 4.5. Confronto delle metriche per diversi modelli sfruttando un approccio supervisionato, utilizzando il *dataset* di TESS.

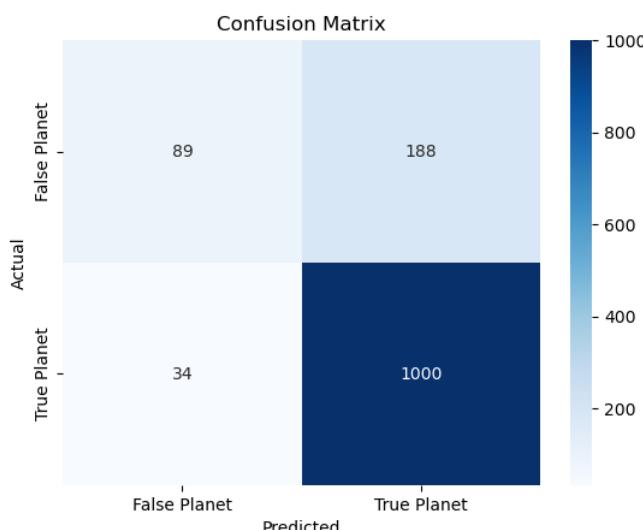


Figura 4.1. Matrice di Confusione dalla Random Forest sui dati TESS.

Capitolo 4 – Risultati sperimentali

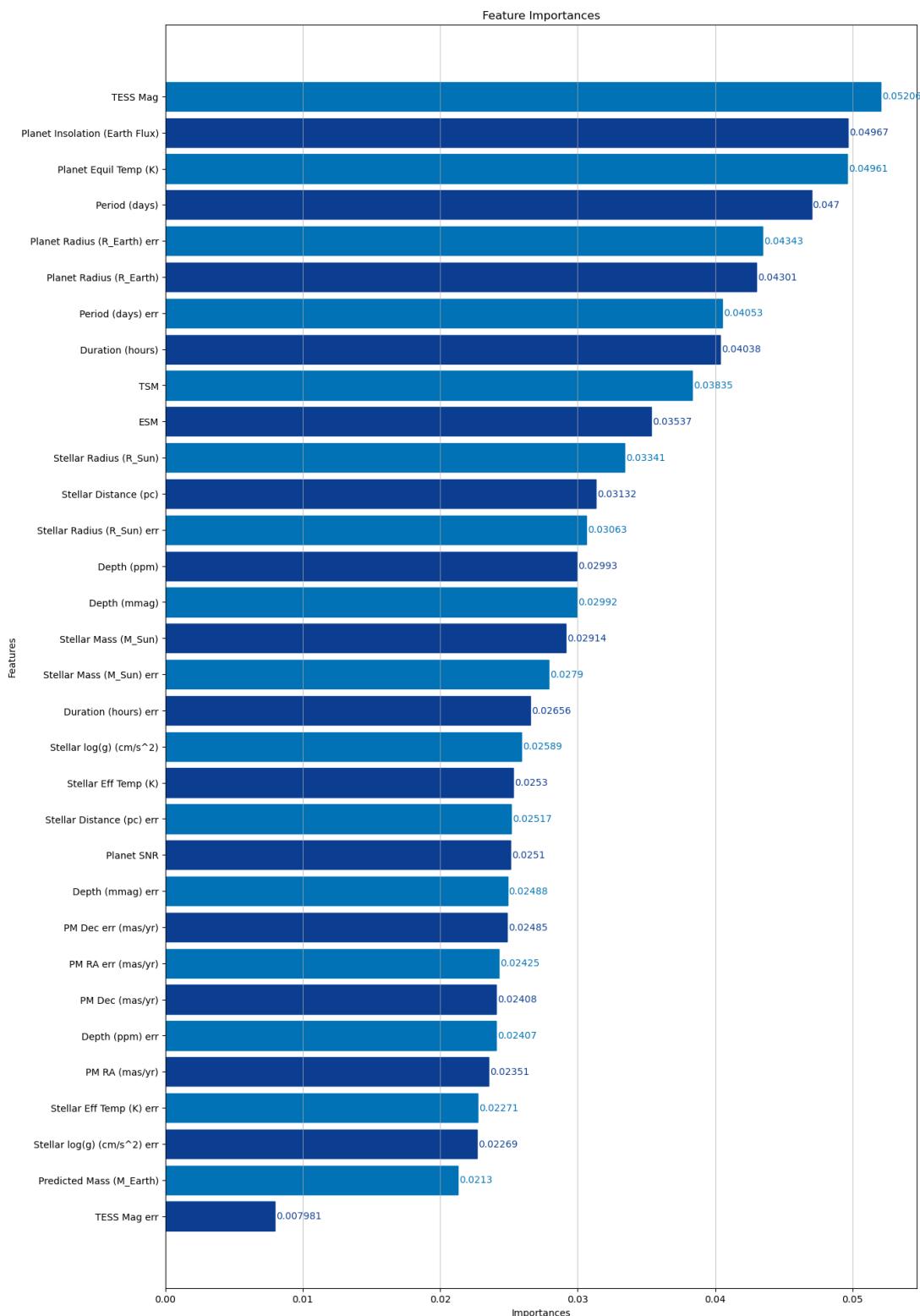


Figura 4.2. Importanza delle caratteristiche restituite dalla Random Forest sui dati TESS.

4.3.2 Set di dati: Kepler

A differenza del *dataset* di TESS, quello di Kepler risulta essere quasi bilanciato, come indicato nella tabella 4.4. Per questo motivo, non è stato applicato alcun metodo di bilanciamento. Successivamente, il *dataset* è stato suddiviso, destinando l’80% all’addestramento e il 20% al test. La selezione del modello ha seguito la stessa griglia considerata per TESS, come descritto nella sezione 4.3.1.

Risultati

Nella tabella 4.6 sono riportati i risultati ottenuti dai vari modelli. La matrice di confusione della Random Forest è visualizzata nella Figura 4.3. In questo caso i risultati sono più affidabili, grazie al fatto che il *dataset* è bilanciato, è c’è un quantitativo sufficiente di campioni per ogni classe. Infine, vengono presentate le importanze delle caratteristiche restituite dal modello *Random Forest* nella Figura 4.4.

	Accuratezza (%)	F1 Score (%)	Precisione (%)	Recall (%)
Albero Decisionale	87,10	87,93	89,49	86,43
Random Forest	90,47	91,07	92,79	89,42
K Nearest Neighbors	79,79	82,42	78,16	87,17
Gradient Boosting	90,12	90,87	91,36	90,38

Tabella 4.6. Confronto delle metriche per diversi modelli sfruttando un approccio supervisionato, utilizzando il *dataset* di Kepler.

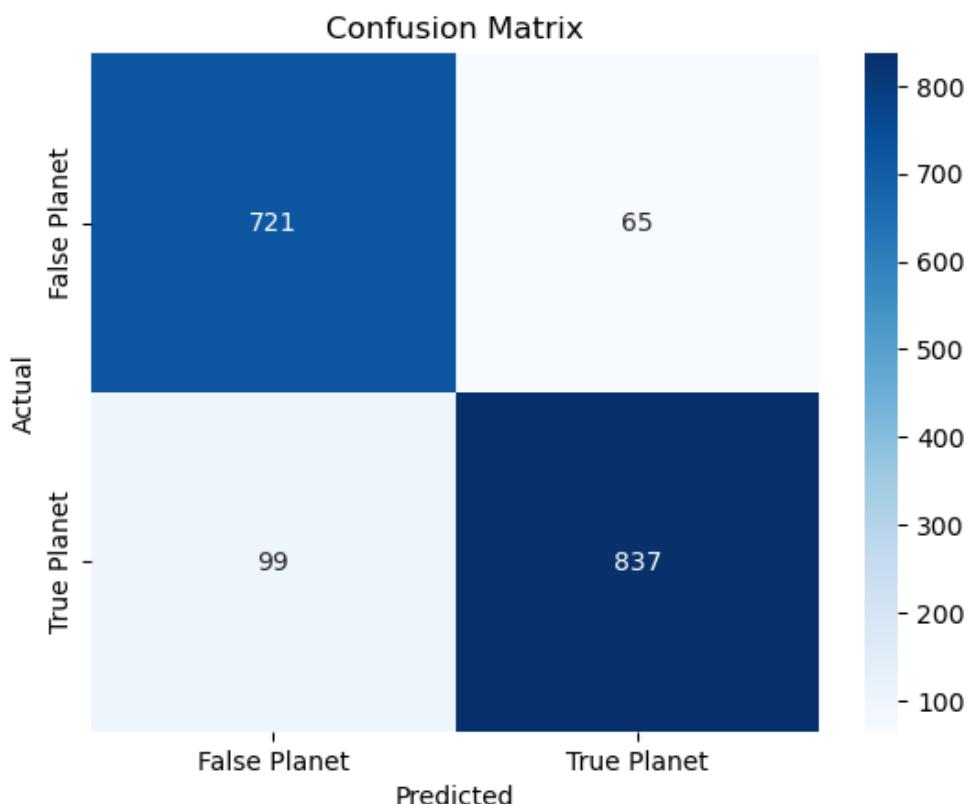


Figura 4.3. Matrice di Confusione della Random Forest sui dati Kepler.

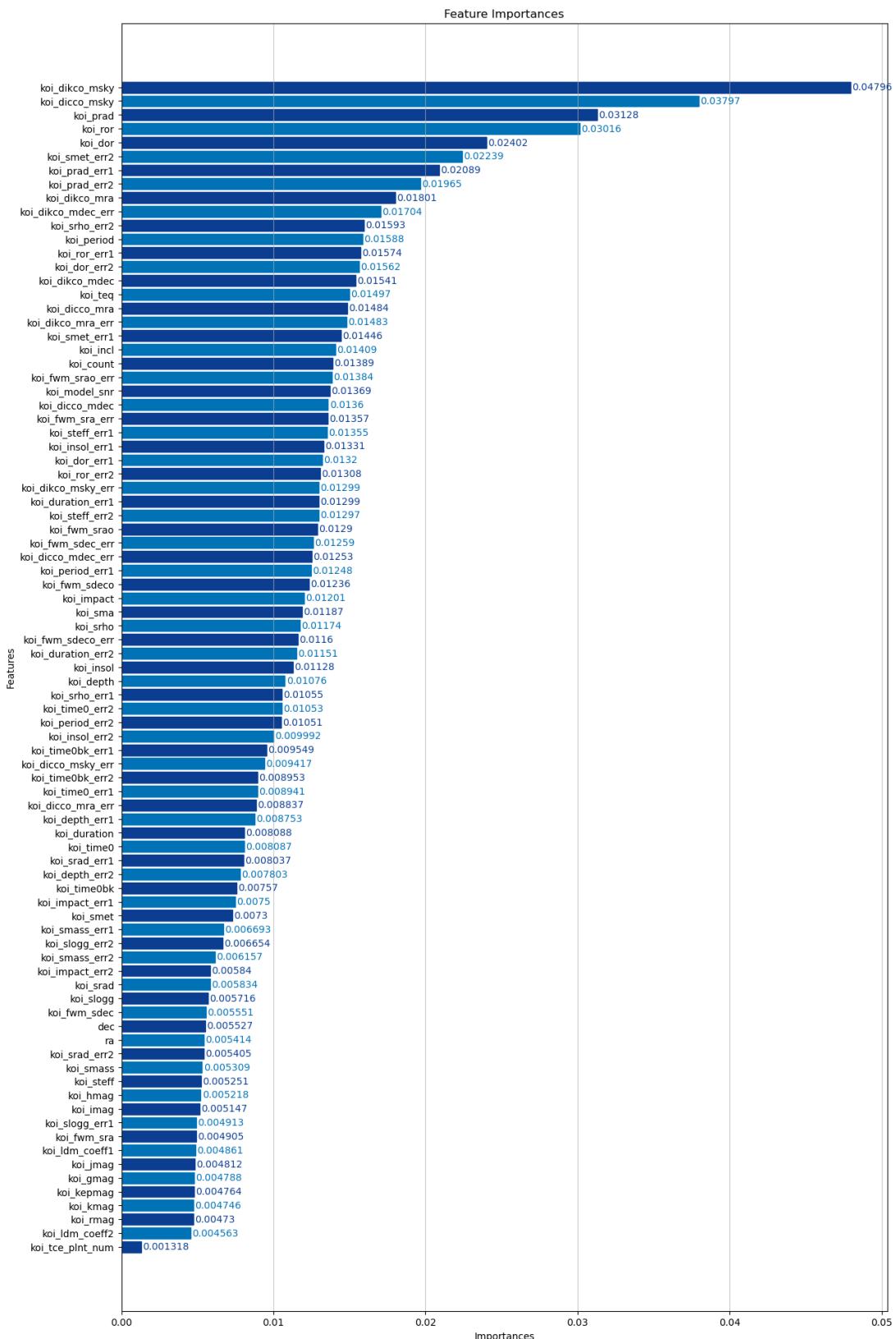


Figura 4.4. Importanza delle caratteristiche restituite dalla Random Forest sui dati Kepler.

4.3.3 Set di dati: TESS/Kepler

Il *dataset* TESS/Kepler mostra uno sbilanciamento a favore dei Veri Pianeti, evidenziato nella tabella 4.4. Pertanto, è stato considerato il metodo descritto nella sezione 3.7.4, calcolando un peso per la classe *Vero Pianeta* pari a 0,82 e per la classe *Falso Pianeta* pari a 1,27. Successivamente, il *dataset* è stato suddiviso, destinando l'80% all'addestramento e il 20% al test. Per la selezione del modello è stata presa in considerazione la stessa griglia utilizzata per TESS, come descritto nella sezione 4.3.1.

Risultati

I risultati ottenuti dai vari modelli sono riportati nella Tabella 4.7. La matrice di confusione della Random Forest è visualizzata nella Figura 4.5. In questo caso, si osserva un lieve calo di accuratezza rispetto ai test eseguiti sul *dataset* di Kepler, come descritto nella Tabella 4.6. Tuttavia, confrontando i risultati con quelli ottenuti esclusivamente sul *dataset* di TESS (vedi Tabella 4.5), emergono risultati affidabili. Infine, le importanze delle caratteristiche restituite dal modello Random Forest sono presentate nella Figura 4.6.

	Accuratezza (%)	F1 Score (%)	Precisione (%)	Recall (%)
Albero Decisionale	79,63	83,35	82,42	84,30
Random Forest	83,69	86,92	84,37	89,63
K Nearest Neighbors	79,49	84,09	79,19	89,63
Gradient Boosting	82,97	86,40	83,51	89,51

Tabella 4.7. Confronto delle metriche per diversi modelli sfruttando un approccio supervisionato, utilizzando il *dataset* di TESS/Kepler.

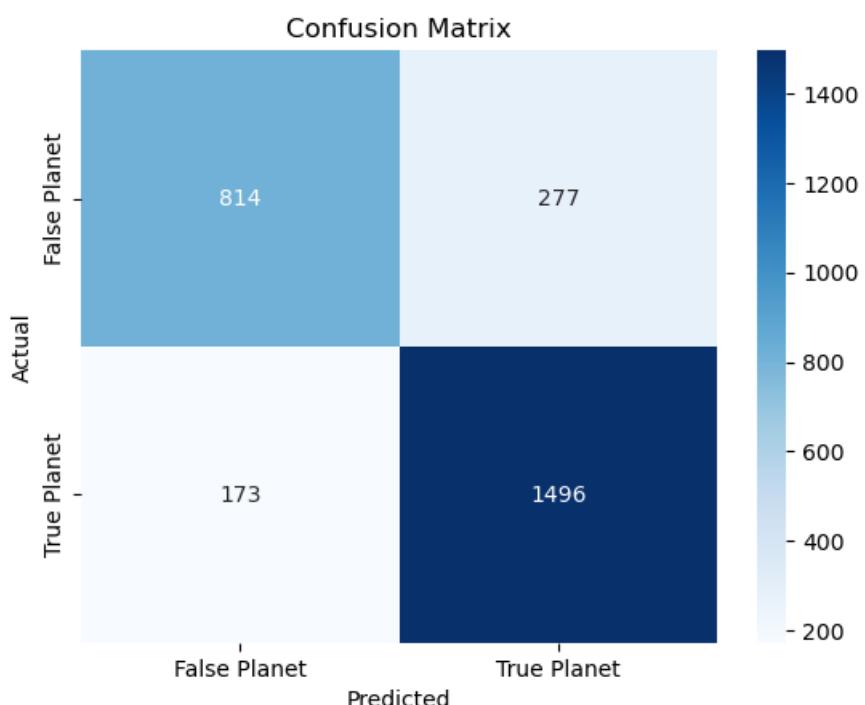


Figura 4.5. Matrice di Confusione della Random Forest sui dati TESS/Kepler.

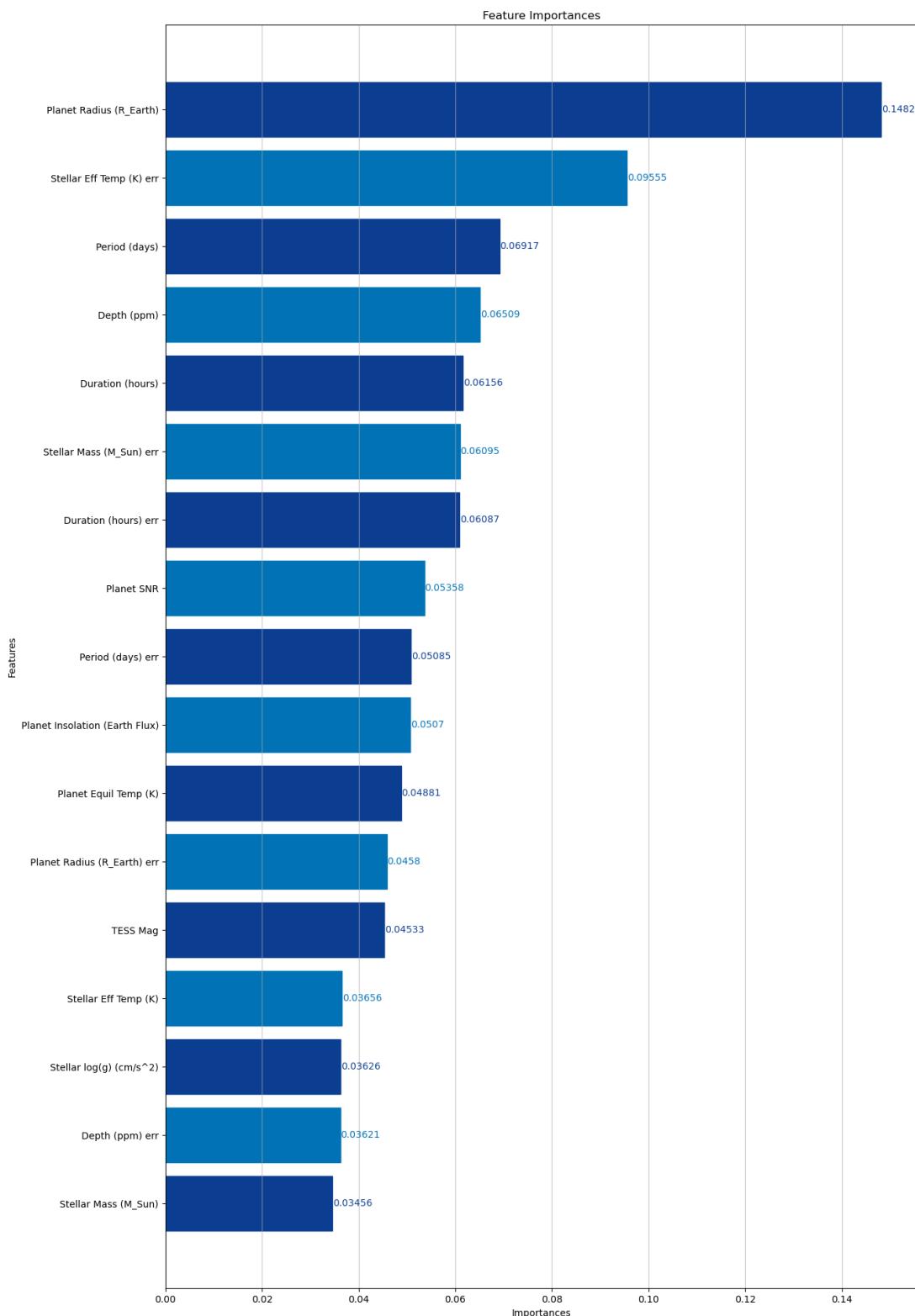


Figura 4.6. Importanza delle caratteristiche restituite dalla Random Forest sui dati TESS/Kepler.

Riassunto Random Forest

Nella Tabella 4.8, viene presentato un riassunto dei risultati ottenuti attraverso l'utilizzo dell'algoritmo Random Forest su diversi set di dati. Si osserva un lieve migliora-

mento dell’accuratezza nei test eseguiti sul *dataset* combinato di TESS/Kepler rispetto a quelli condotti esclusivamente sul *dataset* di TESS, sebbene sia nell’ordine dei decimali. Tuttavia, i risultati relativi a TESS/Kepler appaiono affidabili, come evidenziato anche dalla matrice di confusione illustrata nella Figura 4.5. La situazione cambia quando si esamina la matrice di confusione relativa al solo *dataset* di TESS, mostrata nella Figura 4.1.

	Accuratezza (%)	F1 Score (%)	Precisione (%)	Recall (%)
TESS	83,06	90,00	84,17	96,71
Kepler	90,47	91,07	92,79	89,42
TESS/Kepler	83,69	86,92	84,37	89,63

Tabella 4.8. Riassunto dei risultati ottenuti tramite l’algoritmo del Random Forest su vari set di dati.

4.4 Apprendimento Non Supervisionato

L’apprendimento non supervisionato ha coinvolto l’utilizzo delle Self-Organizing Maps (consultare anche la sezione 3.6.2). In questo contesto, è stato considerato un unico *dataset*, risultante dall’unione dei cataloghi di TESS e Kepler. Nella Tabella 4.4 è presente un breve riepilogo delle caratteristiche del *dataset*. Anche in questo caso, sono stati condotti diversi test, in particolare 4, con l’obiettivo comune di utilizzare la Self-Organizing Map per la classificazione tra *Veri Pianeti* e *Falsi Pianeti*, impiegando il set di dati TESS/Kepler. Nelle prossime sezioni, verrà fornita una spiegazione dettagliata di tutti i test e dei risultati ottenuti tramite l’utilizzo della SOM.

4.4.1 SOM: Test 1

Il primo test mira a utilizzare la Self-Organizing Map per la classificazione tra *Veri Pianeti* e *Falsi Pianeti*, impiegando il set di dati TESS/Kepler senza ricorrere a tecniche di bilanciamento e utilizzando tutte le sue caratteristiche.

Selezione del modello

Anche in questo caso, è stata adottata una griglia di ricerca per ottimizzare i parametri. In particolare, sono stati ottimizzati: il numero di righe e colonne della SOM, le distanze utilizzate per il calcolo del BMU e del vicinato, e il numero di vicini più vicini utilizzati durante la classificazione. Per questo test specifico, è stata impiegata una mappa 30x30 con 100,000 iterazioni durante la fase di addestramento. La distanza euclidea è stata utilizzata per il calcolo dei BMU, mentre la distanza di Manhattan è stata adottata per il raggio del vicinato. Il parametro K è stato fissato a 9 per i K vicini più vicini durante la fase di classificazione. Inoltre, i pesi della SOM sono stati inizializzati prendendo in modo pseudo-casuale campioni dal *dataset*, includendo il 50% della classe *Vero Pianeta* e il 50% della classe *Falso Pianeta*. In genere, i pesi della mappa, si inizializzano con vettori casuali, questo per sottolineare la sua capacità di auto-organizzarsi. Tuttavia nelle applicazioni pratiche ci si aspetta di raggiungere l’ordinamento finale il più rapidamente possibile, quindi la selezione di un buon stato iniziale può accelerare la convergenza [29].

Visualizzazione dei dati

Dopo aver ottenuto i migliori parametri, si procede con la fase di addestramento. Nella Figura 4.9 sono registrati i decadimenti, del tasso di apprendimento e del raggio del vicinato. La visualizzazione della SOM può essere eseguita mediante la U-Matrix, che rappresenta le distanze tra i neuroni dopo la fase di addestramento. Le unità più "vicine" tra di loro sono evidenziate con colori simili. Poiché il calcolo delle distanze tra i neuroni deve considerare anche le adiacenze di ciascuna unità, esistono due tipi di U-Matrix: rettangolare ed esagonale. La U-Matrix rettangolare considera 4 adiacenze, a differenza di quella esagonale che ne considera 6. Nella Figura 4.10 è mostrata la U-Matrix rettangolare. La SOM viene utilizzata anche come strumento di visualizzazione dei dati per analizzare la formazione dei cluster con le singole *features*. Come evidenziato nella Figura 4.11, le *features Stellar Eff Temp (K)*, *Stellar log(g) (cm/s²)*, e *TESS Mag* mostrano alcune anomalie. Per questo motivo, nel test 4.4.3 e 4.4.4, sono state eliminate, insieme a *Stellar Eff Temp (K) err*, poiché è correlata a *Stellar Eff Temp (K)*. Nella Figura 4.12 viene visualizzata la dispersione di ogni osservazione sulla mappa. Per ogni osservazione, è stato calcolato il suo BMU e la sua posizione è stata determinata dalle sue coordinate, con l'aggiunta di un offset calcolato in modo pseudo-casuale in un range tra [-0,5, 0,5]. Nella Figura 4.13 sono rappresentati pallini di colore verde e blu di varie dimensioni. I pallini verdi indicano i Veri Pianeti, mentre quelli blu indicano i Falsi Pianeti. La dimensione del pallino è determinata dal numero di occorrenze della Best Matching Unit (BMU) associata a quel neurone. Si è osservato che se un neurone è la BMU di x osservazioni, tutte le x osservazioni saranno categorizzate come "Veri Pianeti" o "Falsi Pianeti", senza mescolanza. Questa interessante caratteristica è sfruttata nella costruzione della *label map* e quindi nella fase successiva di classificazione, come descritto nella sezione successiva.

Classificazione e Risultati

Attraverso procedure adeguate, le Self-Organizing Maps (SOM) possono essere impiegate per la classificazione di nuovi campioni. Suggerisce [29] che, se i dati in input sono già classificati, è possibile associare i neuroni a una delle classi di partenza. Il metodo proposto prevede due approcci: il primo, adatto quando si dispone di un numero significativo di campioni, consiste nell'osservare la distribuzione di essi sulla mappa e assegnare la classe più frequente all' i -esimo nodo. In caso di parità, si può considerare un intorno più ampio. Il secondo approccio, utilizzabile quando non si ha un gran numero di campioni iniziali, prevede l'uso dell'algoritmo K-Nearest Neighbors. Nel nostro contesto, avendo a disposizione un numero sufficiente di campioni, abbiamo adottato il primo metodo. Inoltre, in assenza di problematiche di parità dovute alla mancanza di mescolanza, come descritto nella sezione precedente della visualizzazione dei dati (Sezione 4.4.1), abbiamo sfruttato le etichette già esistenti per etichettare ogni neurone, ottenendo così una *label map*. Come evidenziato nella Figura 4.14, che rappresenta la mappa delle etichette, alcuni neuroni risultano isolati. La figura mostra in verde i neuroni etichettati come *Vero Pianeta* e in blu come *Falso Pianeta*. Considerando le proprietà della Self-Organizing Map (SOM), la quale è in grado di auto-organizzare i neuroni simili in zone vicine, abbiamo ricostruito questa *label map* iniziale basandoci su un intorno 3x3. Durante questa ricostruzione, abbiamo riassunto le etichette nel momento in cui nell'intorno 3x3 erano presenti più neuroni della classe opposta rispetto all'etichetta del i -esimo neurone. Inoltre, dato che il *dataset* è sbilanciato, le classi sono state pesate diversamente, attribuendo il valore 0,9 alla classe *Vero Pianeta* e 1 alla classe *Falso Pianeta*, calcolati durante la selezione del modello.

La Figura 4.15 illustra il risultato di questa operazione, evidenziando una *label map* più omogenea. Nel momento in cui bisogna classificare una nuova osservazione, ne si calcola il suo Best Matching Unit (BMU), il quale sarà già etichettato. Se si decidesse di assegnare la medesima etichetta alla predizione, ciò non sarebbe sufficiente. Si tiene conto anche di un vicinato, utilizzando il metodo K Nearest Neighbors (vedi anche sezione 3.5). Basandoci sui K vicini, si effettua una statistica: se nei K vicini ci sono più neuroni etichettati come *Vero Pianeta*, allora la classe predetta sarà *Vero Pianeta*; altrimenti, sarà *Falso Pianeta*. Questo approccio si basa sui pesi calcolati dall'algoritmo della SOM e sui K vicini più vicini per la classificazione. Attraverso questa procedura sono stati ottenuti i seguenti risultati: Accuratezza 77,57%, F1 Score 82,52%, Precisione 78,01%, e Recall 87,59%. La Figura 4.16 mostra la matrice di confusione.

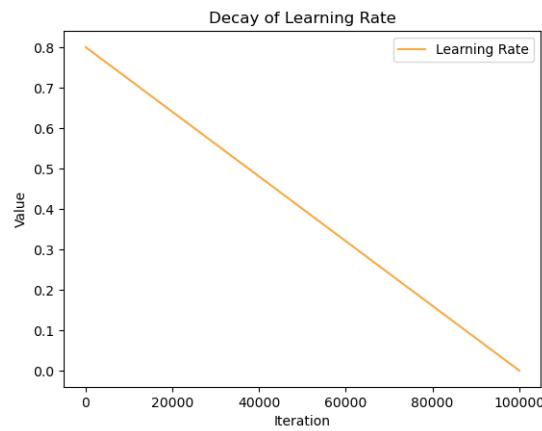


Figura 4.7. Decadimento del tasso di apprendimento (Test 1 SOM).

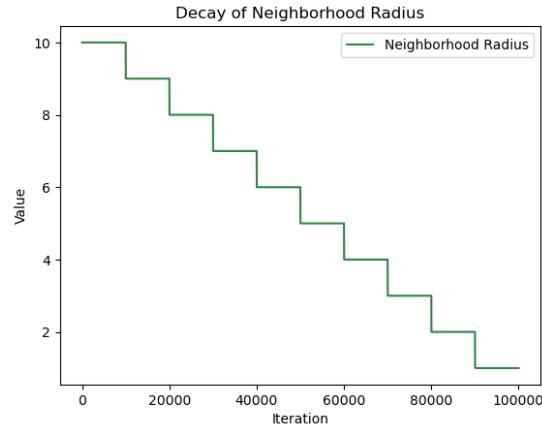


Figura 4.8. Decadimento del raggio del vicinato (Test 1 SOM).

Figura 4.9. Decadimento del tasso di apprendimento e del raggio del vicinato (Test 1 SOM).

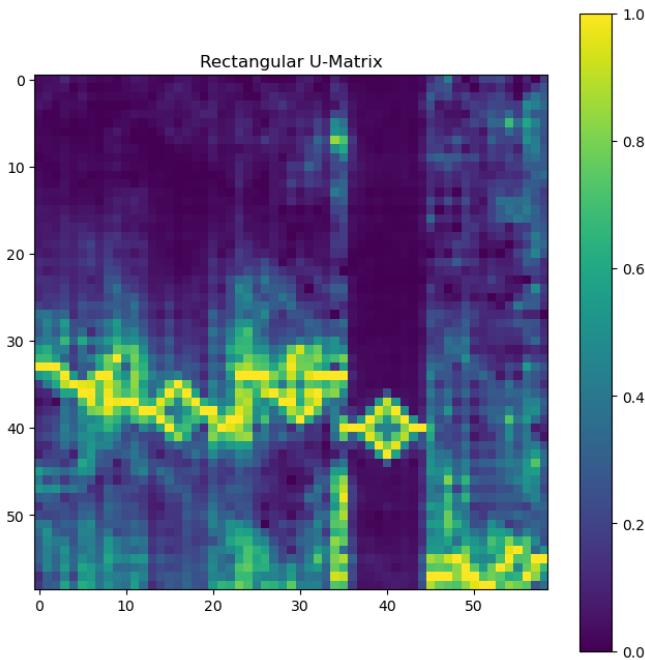


Figura 4.10. U-Matrix Rettangolare (Test 1 SOM).

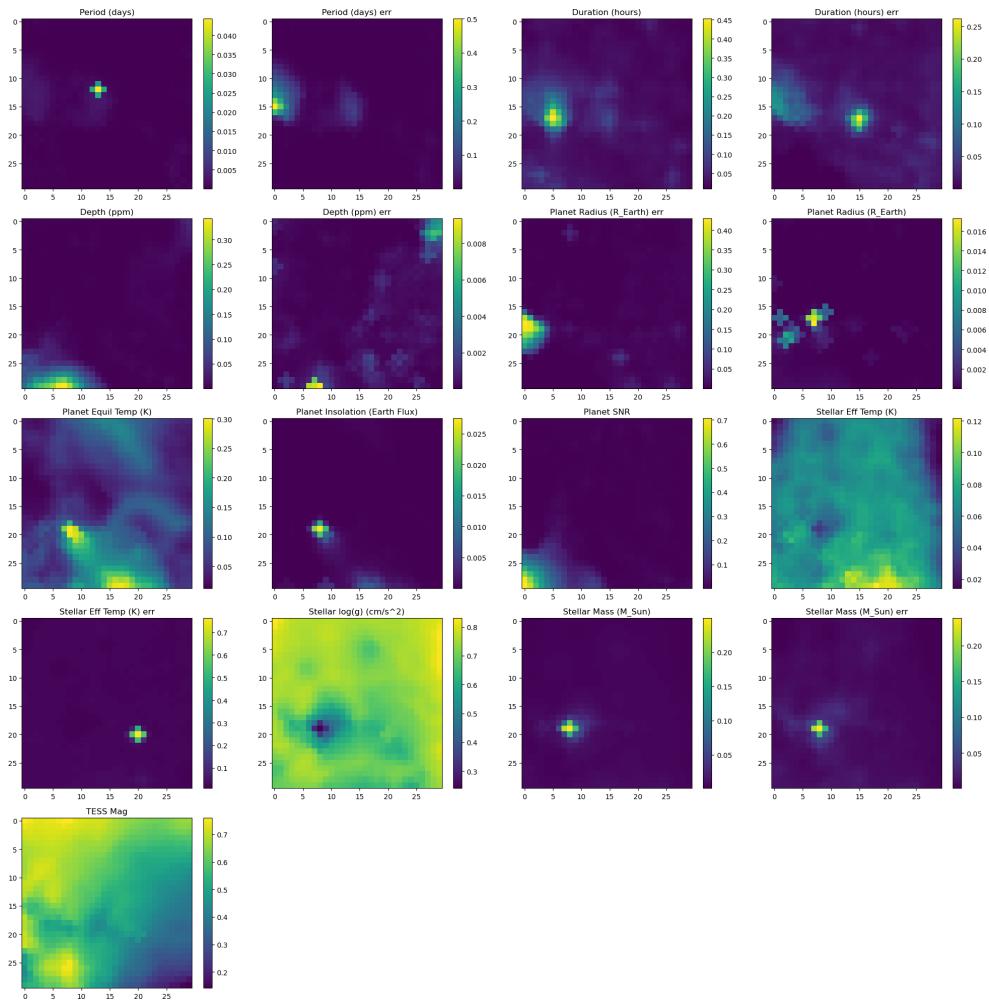


Figura 4.11. Cluster delle singole *features* nello spazio della SOM (Test 1 SOM).

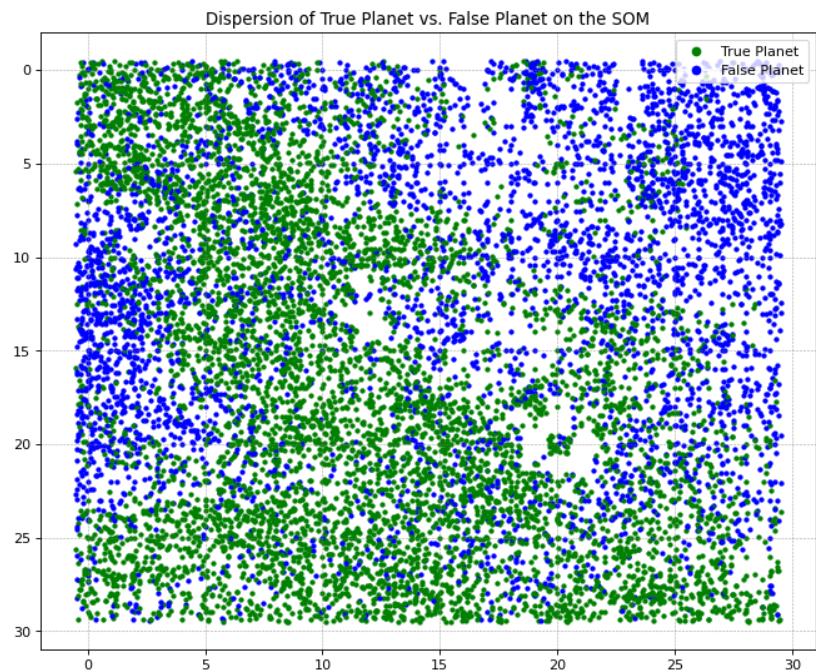


Figura 4.12. Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 1 SOM).

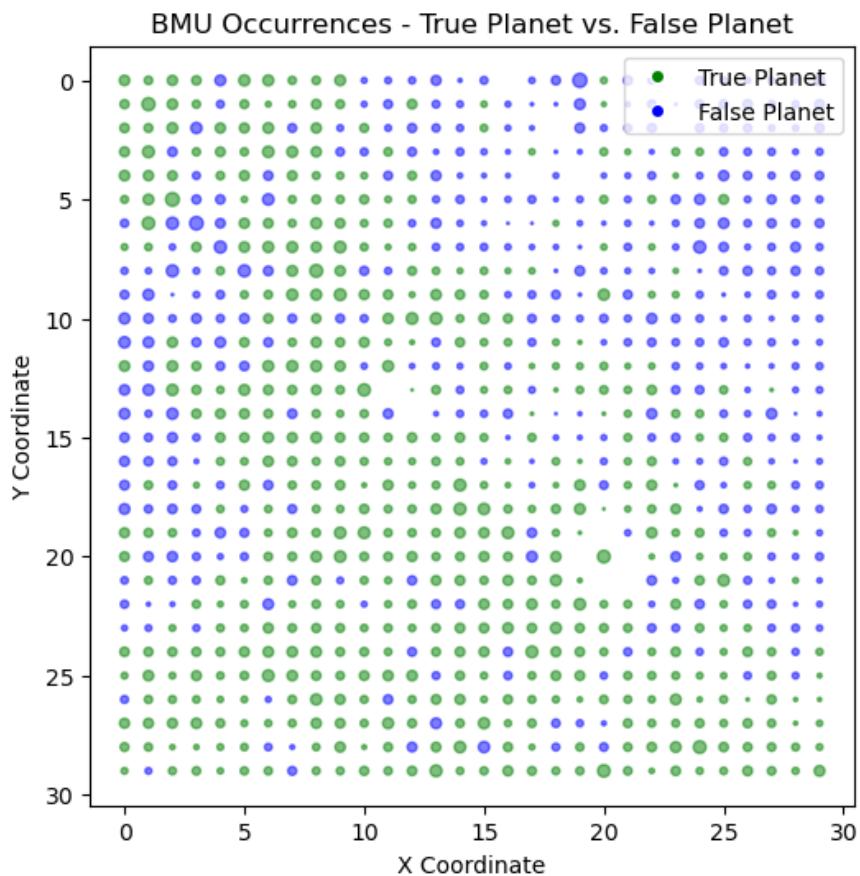


Figura 4.13. Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 1 SOM).

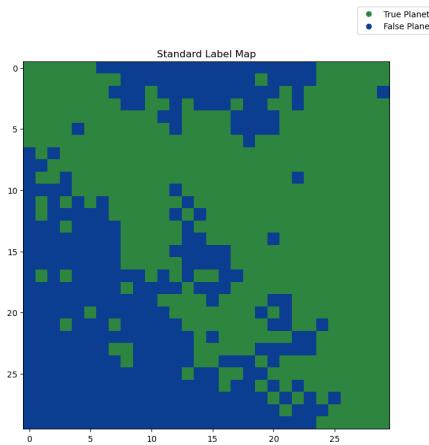


Figura 4.14. *Label Map* (Test 1 SOM).

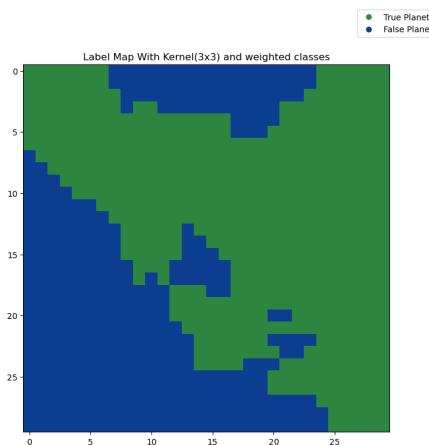


Figura 4.15. *Label Map* ricostruita sulla base di un'intorno 3x3 (Test 1 SOM).

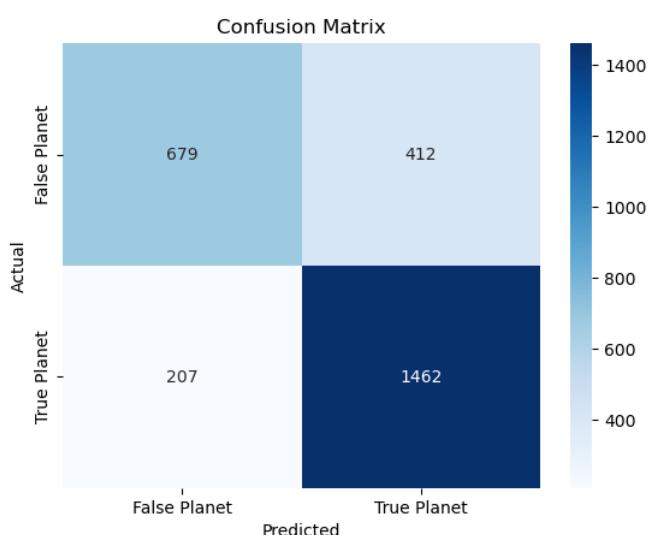


Figura 4.16. Matrice di confusione (Test 1 SOM).

4.4.2 SOM: Test 2

Come nel primo test, anche in questo caso vengono considerate tutte le caratteristiche del *dataset*. Tuttavia, a differenza del primo, qui vengono adottate tecniche di bilanciamento dei dati. In particolare, viene utilizzata la tecnica del sotto-campionamento (vedi sezione 3.7.4), ottenendo così un *dataset* bilanciato composto da 5,402 campioni per classe.

Selezione del modello

Per questo specifico test, vengono ottimizzati gli stessi parametri e i pesi iniziali della SOM descritti nella sezione 4.4.1. In particolare, è stata impiegata una mappa 30x30 con 100,000 iterazioni durante la fase di addestramento. La distanza euclidea è stata utilizzata per il calcolo dei Best Matching Unit (BMU) e per il raggio del vicinato. Il parametro K è stato fissato a 7 per i K vicini più vicini durante la fase di classificazione.

Visualizzazione dei dati

La Figura 4.19 illustra i decadimenti del tasso di apprendimento e del raggio del vicinato. Nella Figura 4.20 è mostrata la U-Matrix rettangolare. La formazione dei cluster nelle singole caratteristiche è evidenziata dalla Figura 4.21. Nella Figura 4.22 viene visualizzata la dispersione di ogni osservazione sulla mappa. La distribuzione delle classi e le occorrenze dei Best Matching Unit (BMU) sono mostrate nella Figura 4.23 (una spiegazione approfondita dei grafici viene fatta nella sezione 4.4.1).

Classificazione e Risultati

La Figura 4.24 mostra la *label map* ricostruita su di un intorno 3x3. A differenza del test 1, i pesi delle due classi ora sono unitari (il valore 1 viene assegnato ad entrambi le classi), poiché si tratta di un *dataset* bilanciato. Attraverso lo stesso approccio del test 1 per la classificazione, sono stati ottenuti i seguenti risultati: Accuratezza 77,04%, F1 Score 78,03%, Precisione 73,72%, e *Recall* 82,87%. La Figura 4.25 mostra la matrice di confusione.

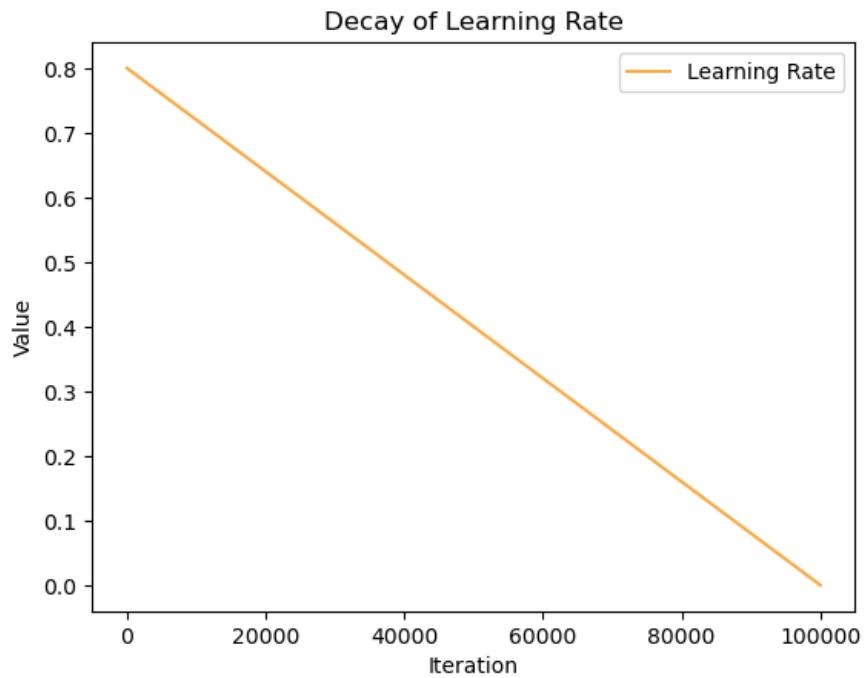


Figura 4.17. Decadimento del tasso di apprendimento (Test 2 SOM).

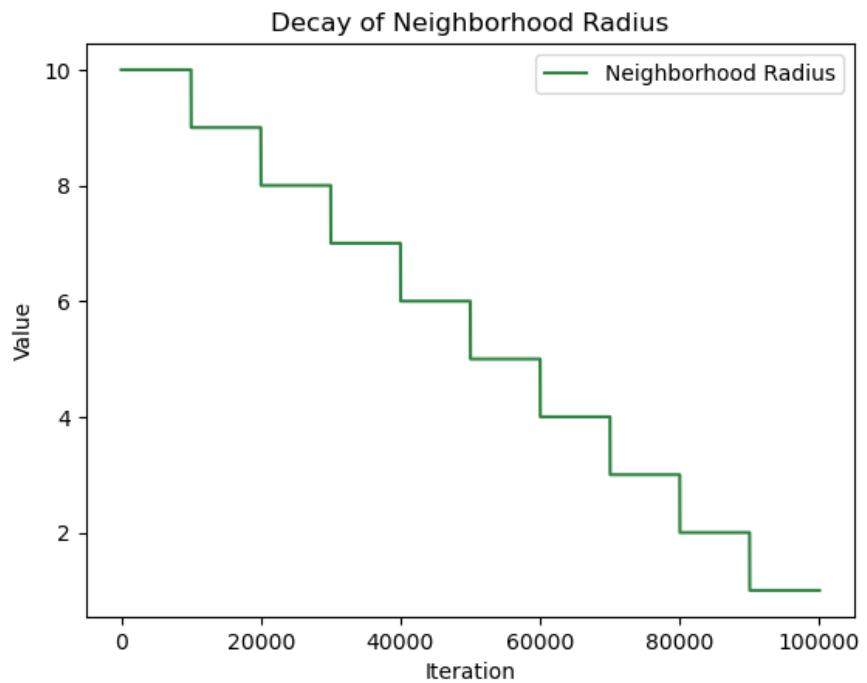


Figura 4.18. Decadimento del raggio del vicinato (Test 2 SOM).

Figura 4.19. Decadimento del tasso di apprendimento e del raggio del vicinato (Test 2 SOM).

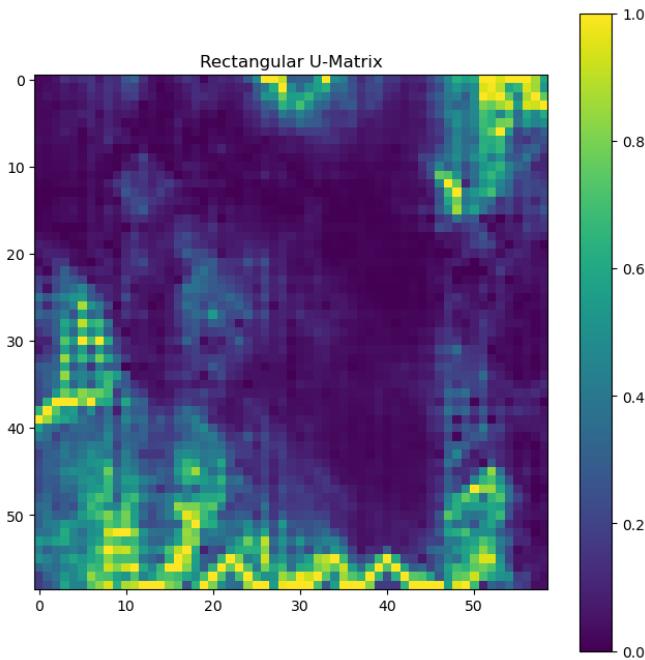


Figura 4.20. U-Matrix Rettangolare (Test 2 SOM).

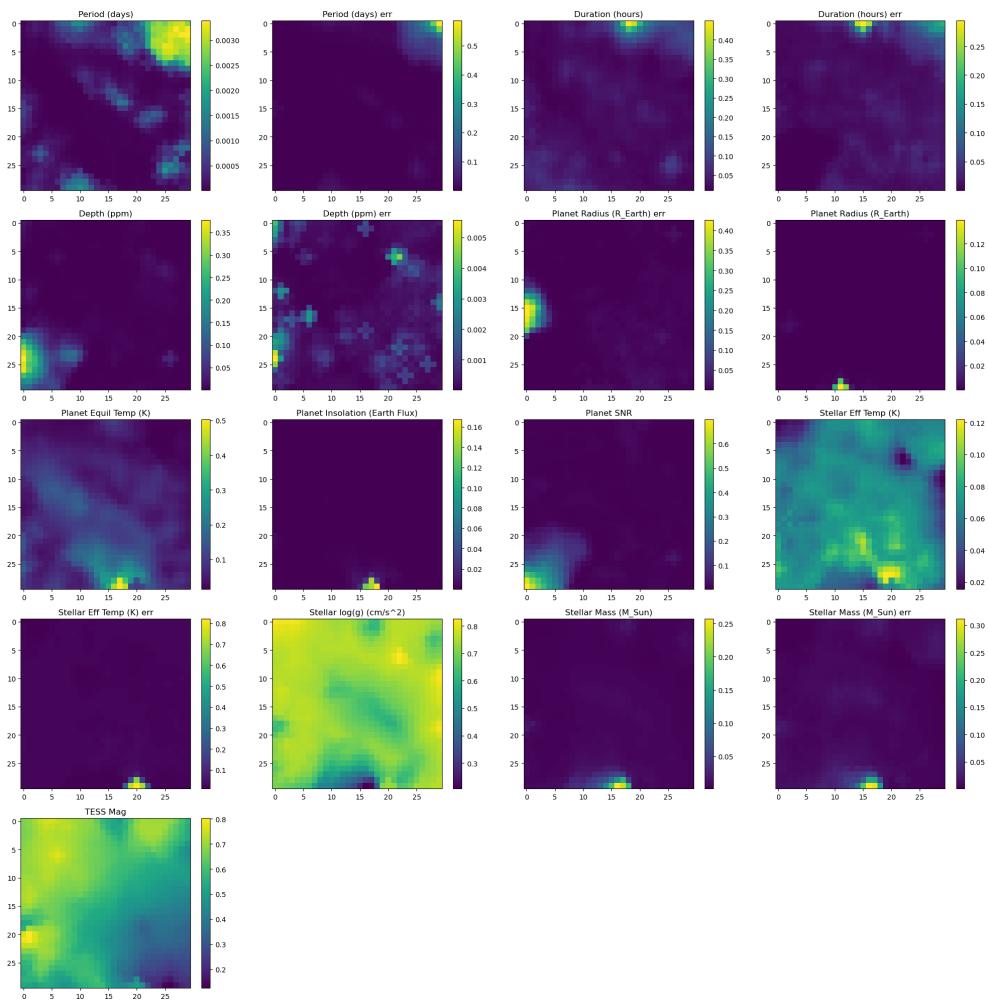


Figura 4.21. Cluster delle singole *features* nello spazio della SOM (Test 2 SOM).

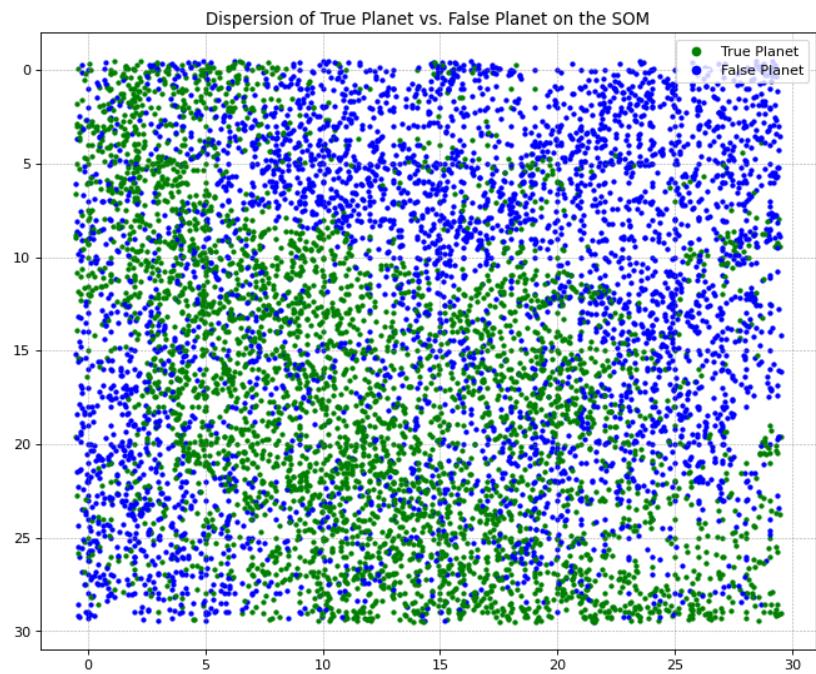


Figura 4.22. Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 2 SOM).

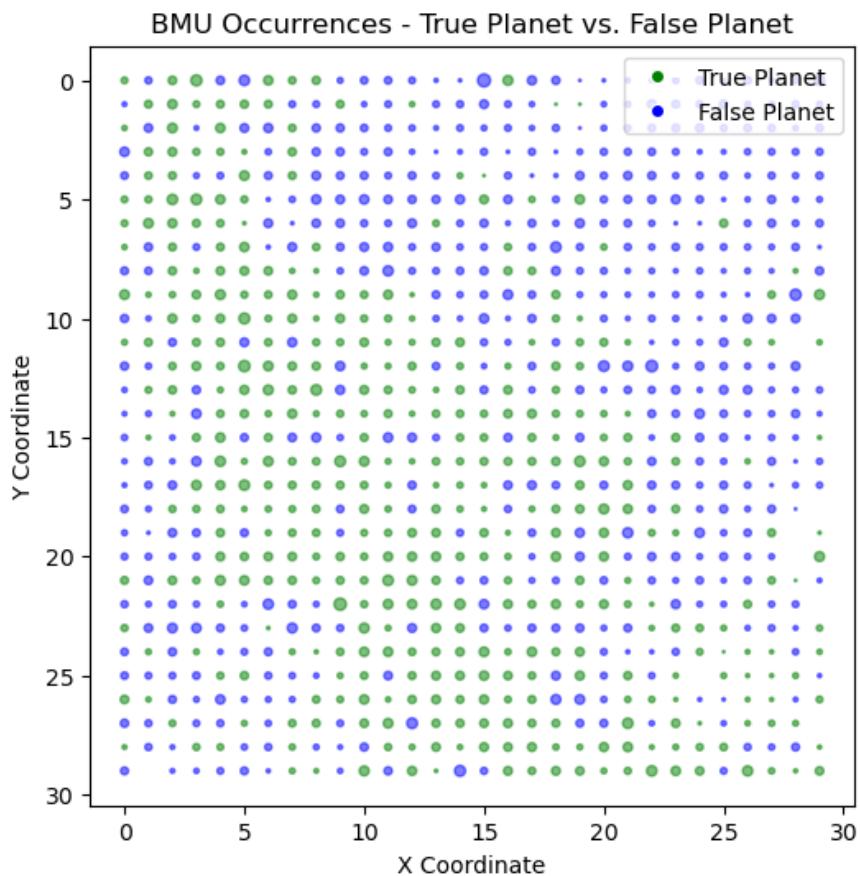


Figura 4.23. Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 2 SOM).

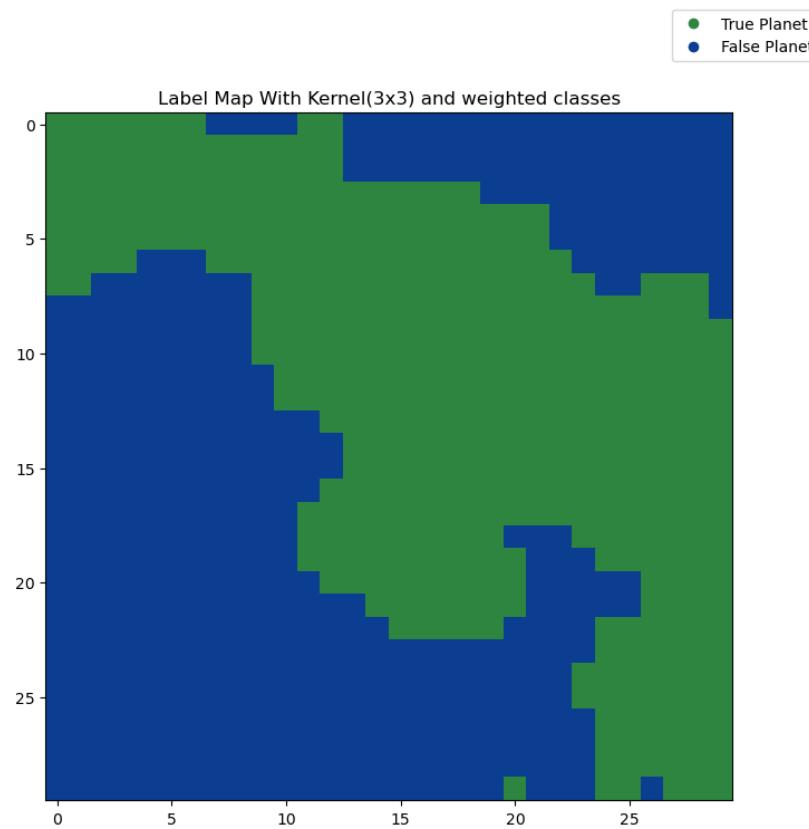


Figura 4.24. *Label Map* ricostruita sulla base di un'intorno 3x3 (Test 2 SOM).

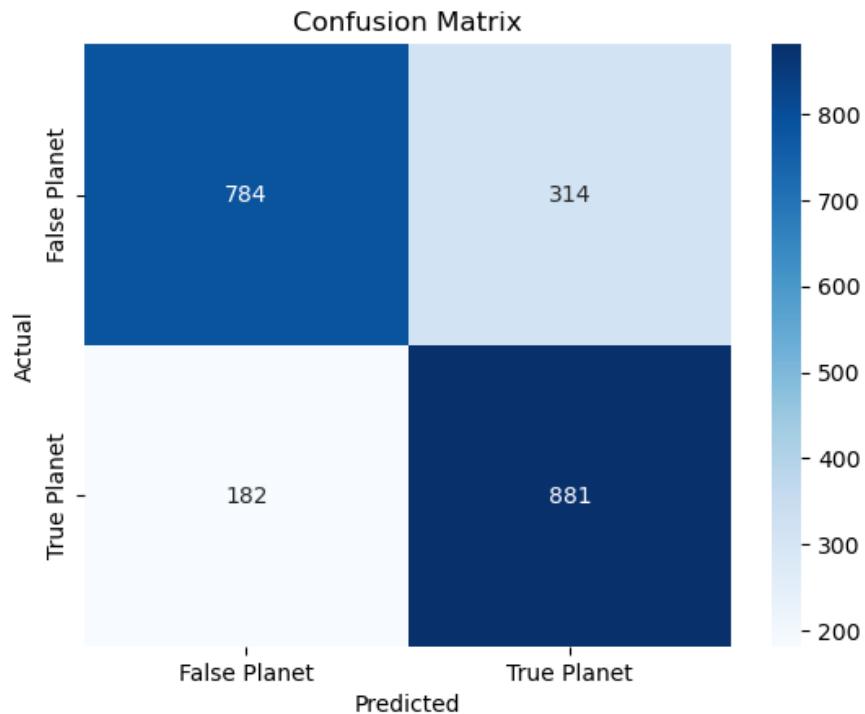


Figura 4.25. Matrice di confusione (Test 2 SOM).

4.4.3 SOM: Test 3

Come nel primo test, anche qui non vengono adottate tecniche di bilanciamento del set di dati. Tuttavia, a differenza sia del primo che del secondo, vengono eliminate alcune *features* anomale, come descritto nella sezione 4.4.1.

Selezione del modello

Per questo specifico test, vengono ottimizzati gli stessi parametri e i pesi iniziali della SOM descritti nella sezione 4.4.1. In particolare, è stata impiegata una mappa 30x30 con 100,000 iterazioni durante la fase di addestramento. La distanza di Manhattan è stata utilizzata per il calcolo dei Best Matching Unit (BMU) e per il raggio del vicinato. Il parametro K è stato fissato a 7 per i K vicini più vicini durante la fase di classificazione.

Visualizzazione dei dati

La Figura 4.28 illustra i decadimenti del tasso di apprendimento e del raggio del vicinato. Nella Figura 4.29 è mostrata la U-Matrix rettangolare. La formazione dei cluster nelle singole caratteristiche è evidenziata dalla Figura 4.30. Nella Figura 4.31 viene visualizzata la dispersione di ogni osservazione sulla mappa. La distribuzione delle classi e le occorrenze dei Best Matching Unit (BMU) sono mostrate nella Figura 4.32 (una spiegazione approfondita dei grafici viene fatta nella sezione 4.4.1).

Classificazione e Risultati

La Figura 4.33 mostra la *label map* ricostruita su di un intorno 3x3. Inoltre, dato che il *dataset* è sbilanciato, in fase di ricostruzione della *label map*, le classi sono state pesate diversamente, attribuendo il valore 0,9 alla classe *Vero Pianeta* e 1 alla classe *Falso Pianeta* calcolati durante la selezione del modello. Attraverso lo stesso approccio del test 1 per la classificazione, sono stati ottenuti i seguenti risultati: Accuratezza 76,15%, F1 Score 81,11%, Precisione 77,85%, e Recall 84,66%. La Figura 4.34 mostra la matrice di confusione.

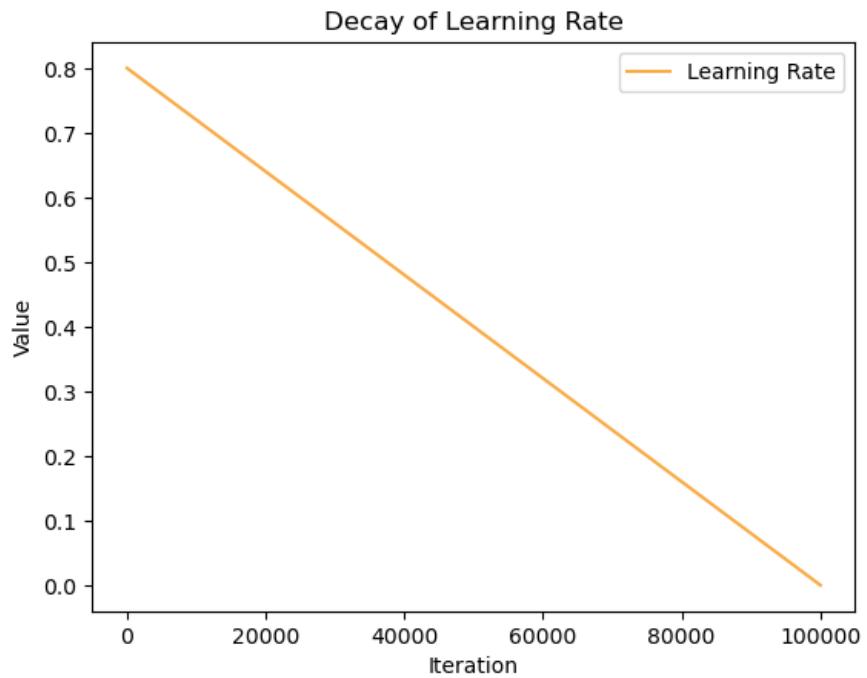


Figura 4.26. Decadimento del tasso di apprendimento (Test 3 SOM).

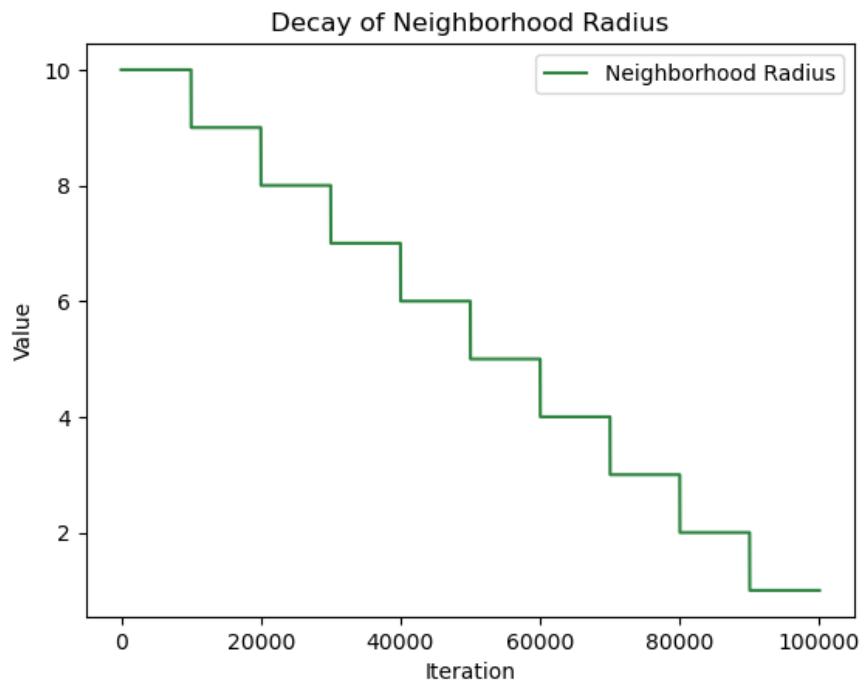


Figura 4.27. Decadimento del raggio del vicinato (Test 3 SOM).

Figura 4.28. Decadimento del tasso di apprendimento e del raggio del vicinato (Test 3 SOM).

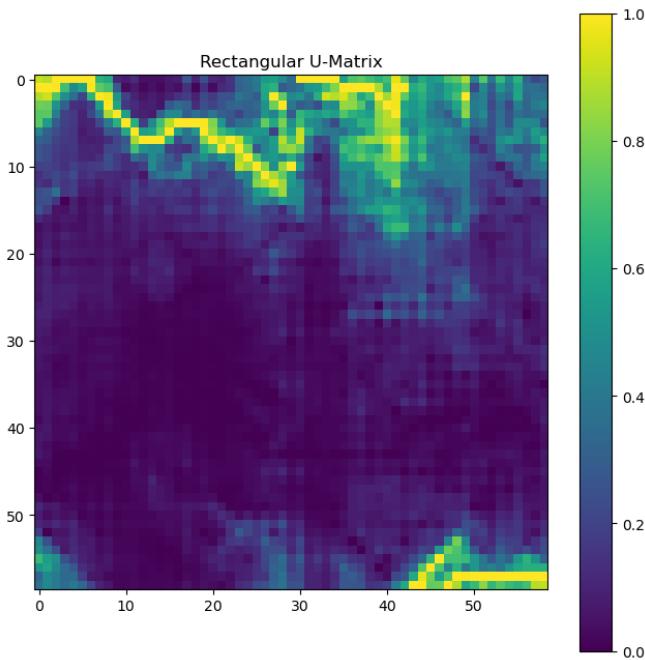


Figura 4.29. U-Matrix Rettangolare (Test 3 SOM).

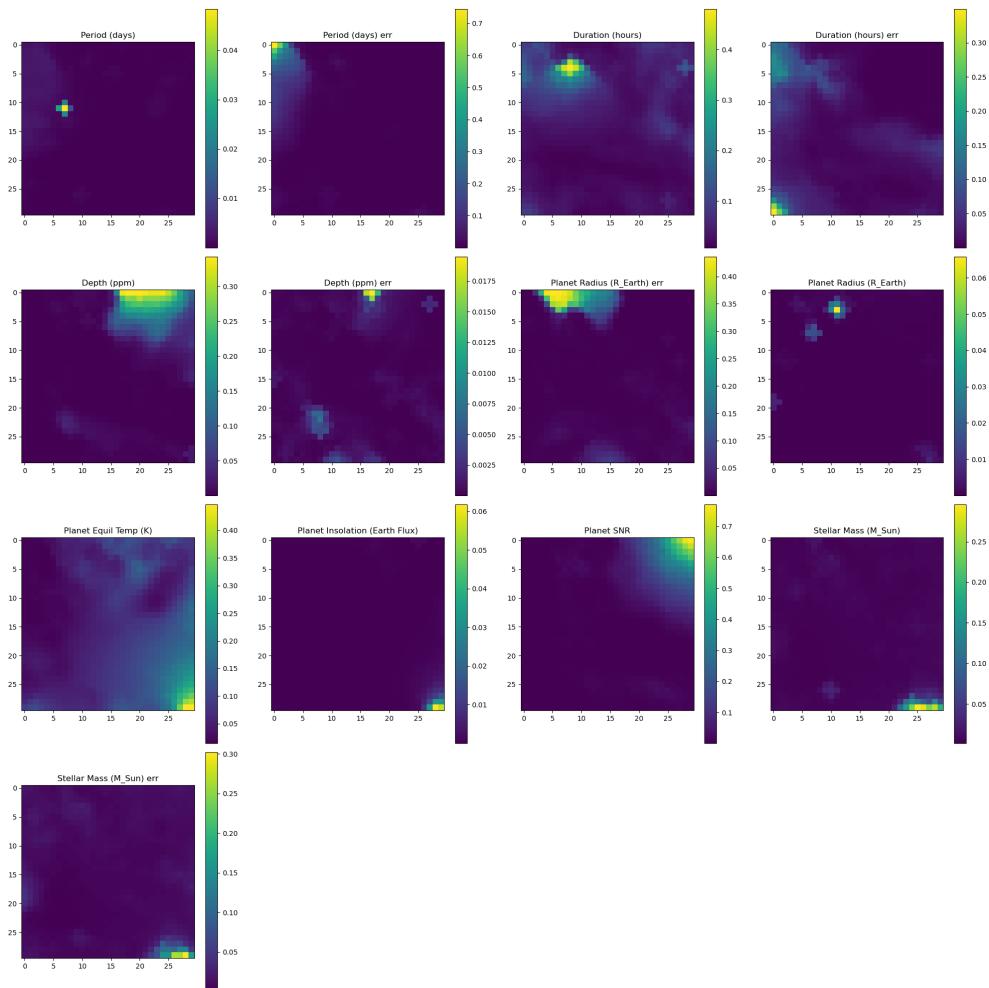


Figura 4.30. Cluster delle singole *features* nello spazio della SOM (Test 3 SOM).

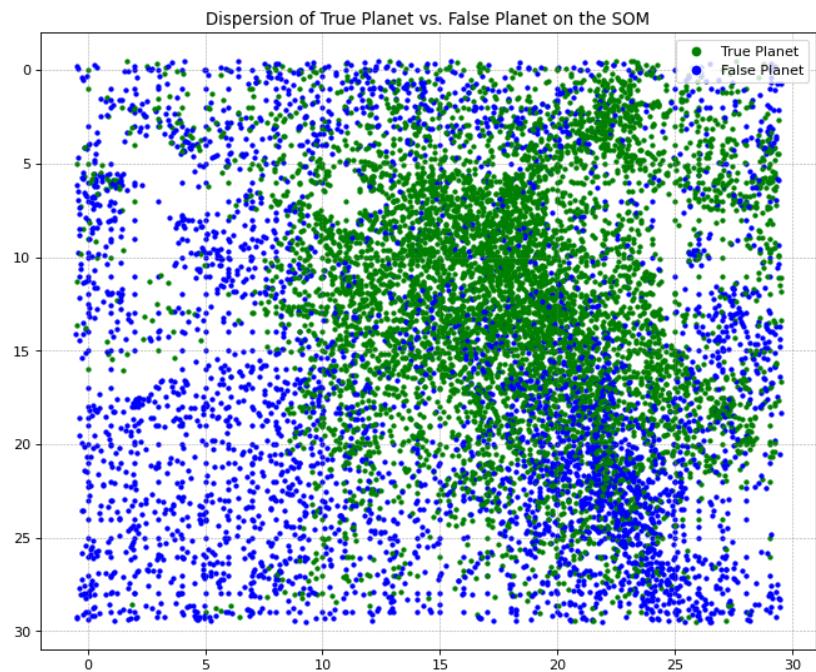


Figura 4.31. Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 3 SOM).

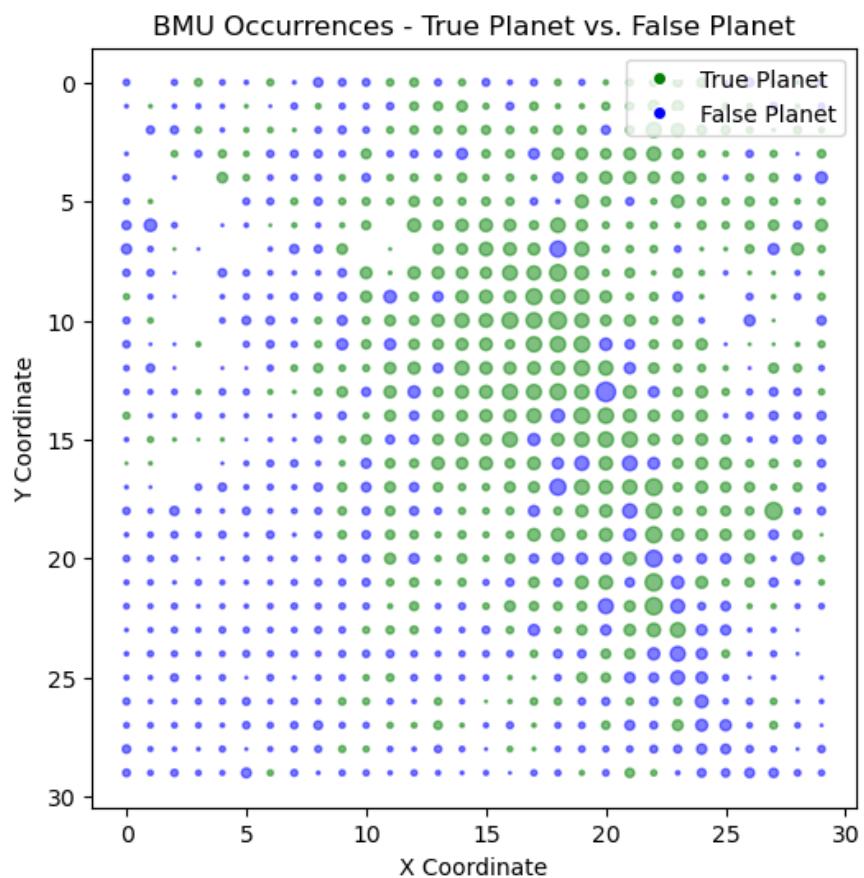


Figura 4.32. Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 3 SOM).

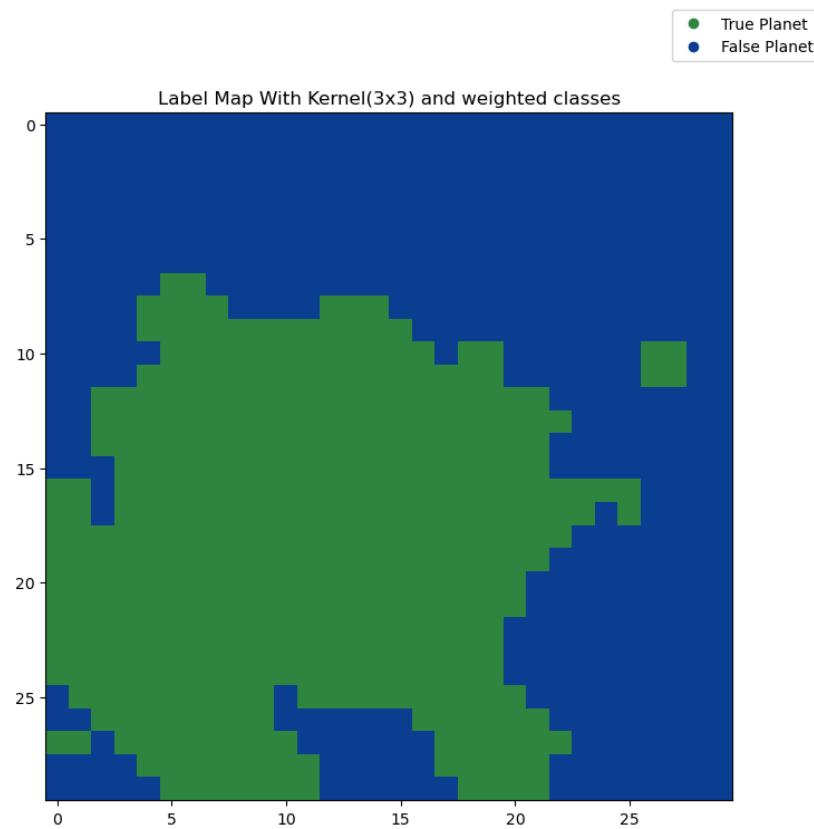


Figura 4.33. *Label Map* ricostruita sulla base di un'intorno 3x3 (Test 3 SOM).

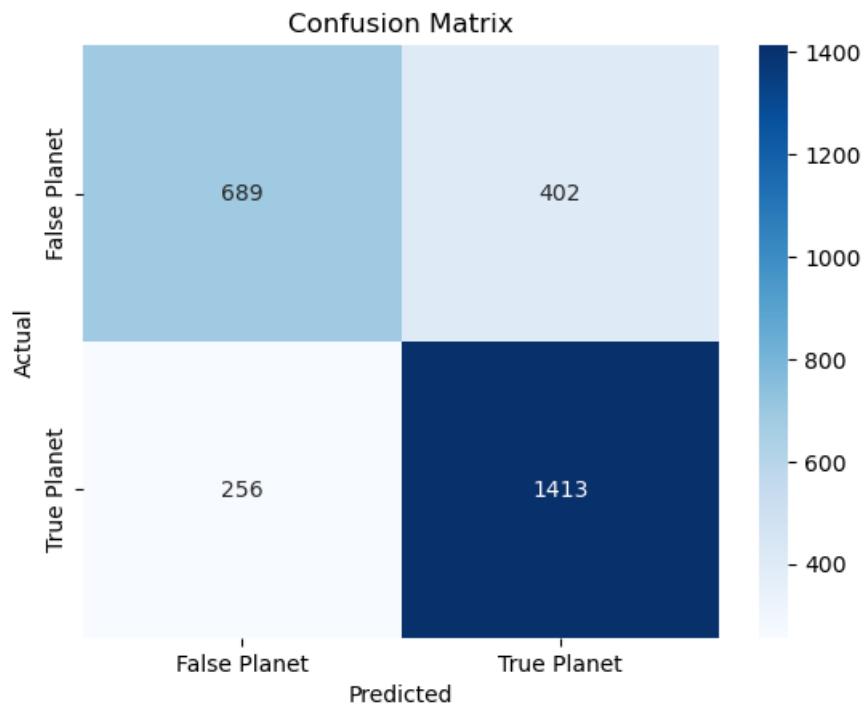


Figura 4.34. Matrice di confusione (Test 3 SOM).

4.4.4 SOM: Test 4

Come nel secondo test, anche qui vengono adottate tecniche di bilanciamento del set di dati. In particolare, viene utilizzata la tecnica del sotto-campionamento (vedi sezione 3.7.4), ottenendo così un *dataset* bilanciato composto da 5,402 campioni per classe. Inoltre, come nel test 3, vengono eliminate alcune *features* anomale (approfondite nella sezione 4.4.1).

Selezione del modello

Per questo specifico test, vengono ottimizzati gli stessi parametri e i pesi iniziali della SOM descritti nella sezione 4.4.1. In particolare, è stata impiegata una mappa 30x30 con 100,000 iterazioni durante la fase di addestramento. La distanza euclidea è stata utilizzata per il calcolo dei BMU, mentre la distanza di Manhattan è stata adottata per il raggio del vicinato. Il parametro K è stato fissato a 11 per i K vicini più vicini durante la fase di classificazione.

Visualizzazione dei dati

La Figura 4.37 illustra i decadimenti del tasso di apprendimento e del raggio del vicinato. Nella Figura 4.38 è mostrata la U-Matrix rettangolare. La formazione dei cluster nelle singole caratteristiche è evidenziata dalla Figura 4.39. Nella Figura 4.40 viene visualizzata la dispersione di ogni osservazione sulla mappa. La distribuzione delle classi e le occorrenze dei Best Matching Unit (BMU) sono mostrate nella Figura 4.41 (una spiegazione approfondita dei grafici viene fatta nella sezione 4.4.1).

Classificazione e Risultati

La Figura 4.42 mostra la *label map* ricostruita su di un intorno 3x3, i pesi delle due classi ora sono unitari (il valore 1 viene assegnato ad entrambi le classi), poiché si tratta di un *dataset* bilanciato. Attraverso lo stesso approccio del test 1 per la classificazione, sono stati ottenuti i seguenti risultati: Accuratezza 75,47%, F1 Score 75,75%, Precisione 73,73%, e Recall 77,89%. La Figura 4.43 mostra la matrice di confusione.

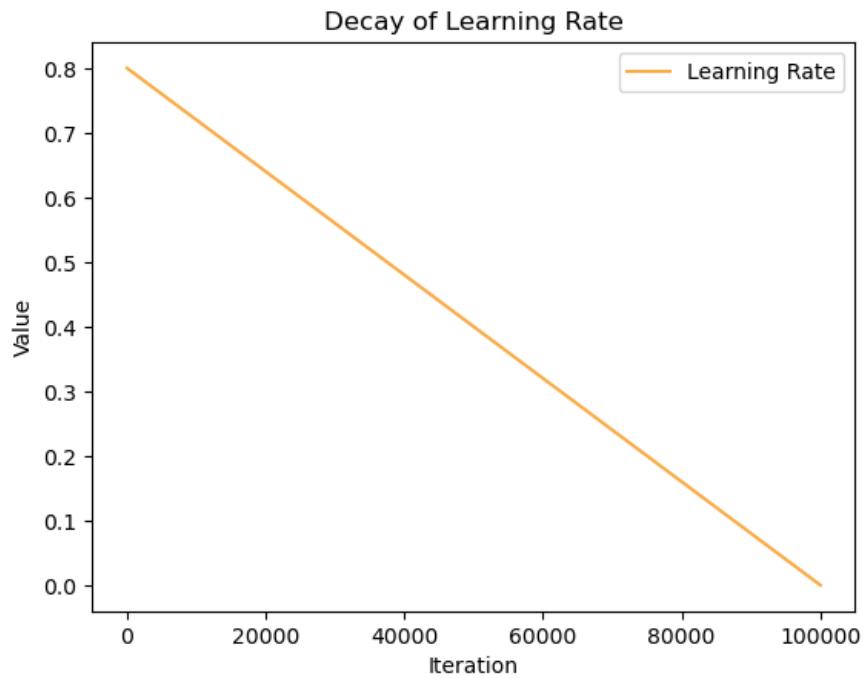


Figura 4.35. Decadimento del tasso di apprendimento (Test 4 SOM).

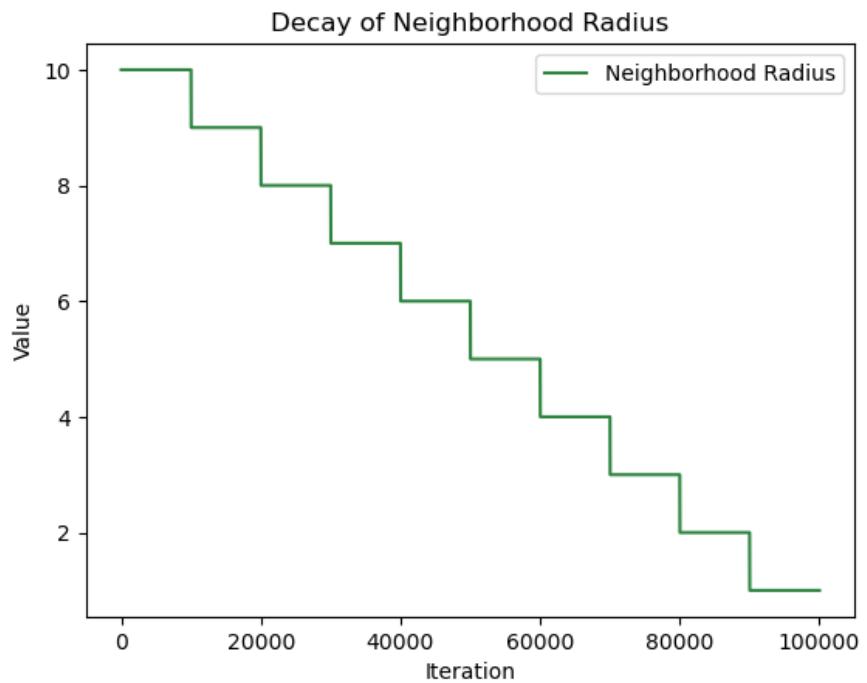


Figura 4.36. Decadimento del raggio del vicinato (Test 4 SOM).

Figura 4.37. Decadimento del tasso di apprendimento e del raggio del vicinato (Test 4 SOM).

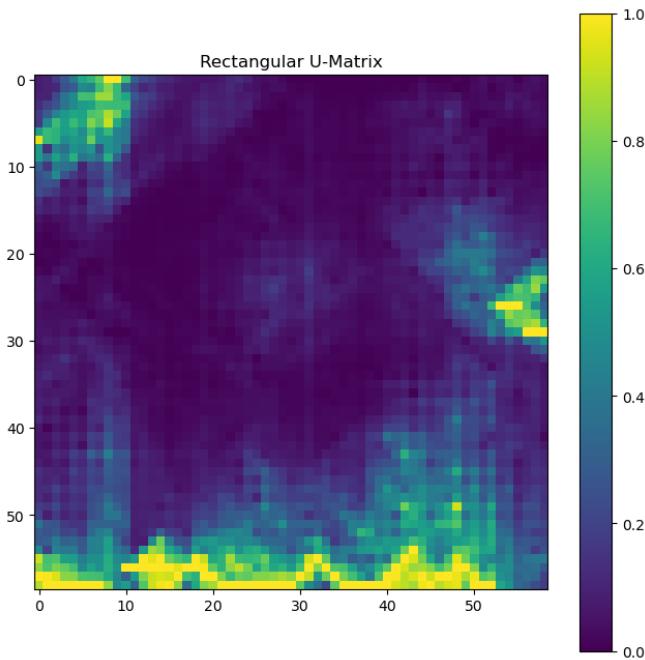


Figura 4.38. U-Matrix Rettangolare (Test 4 SOM).

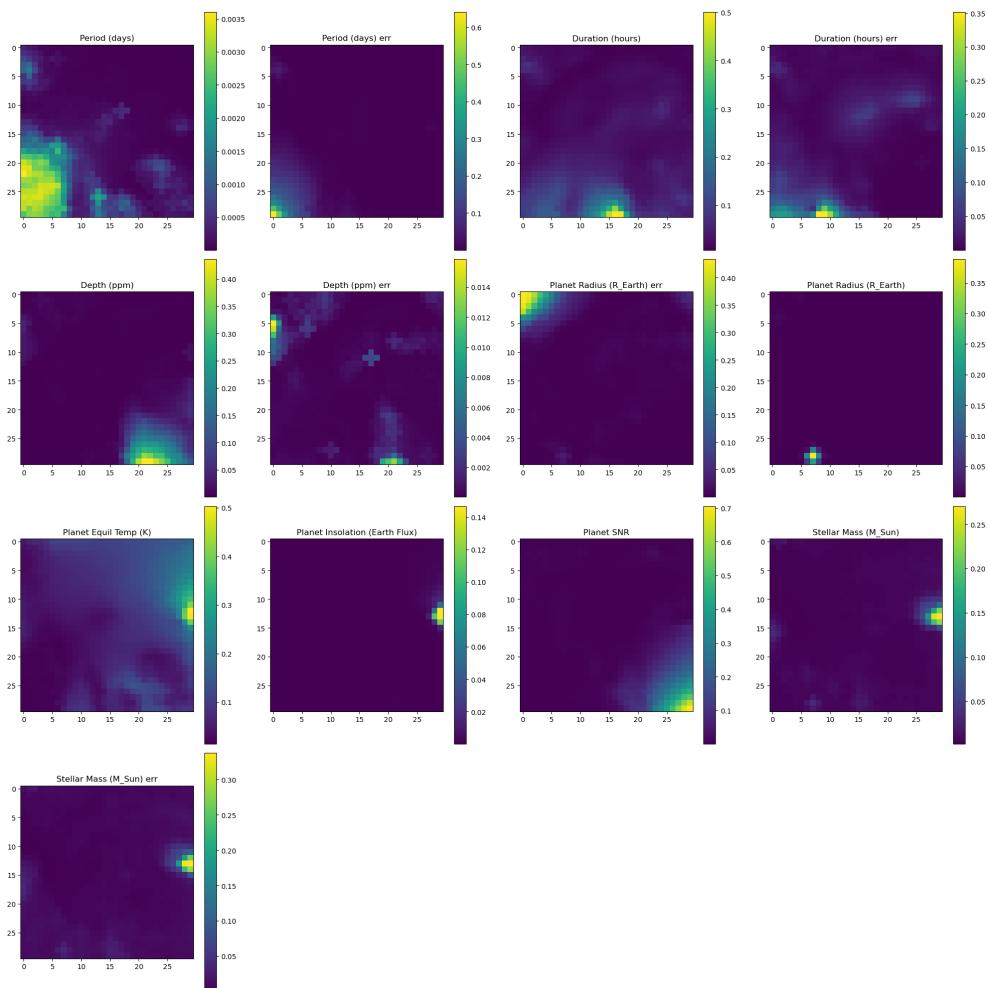


Figura 4.39. Cluster delle singole *features* nello spazio della SOM (Test 4 SOM).

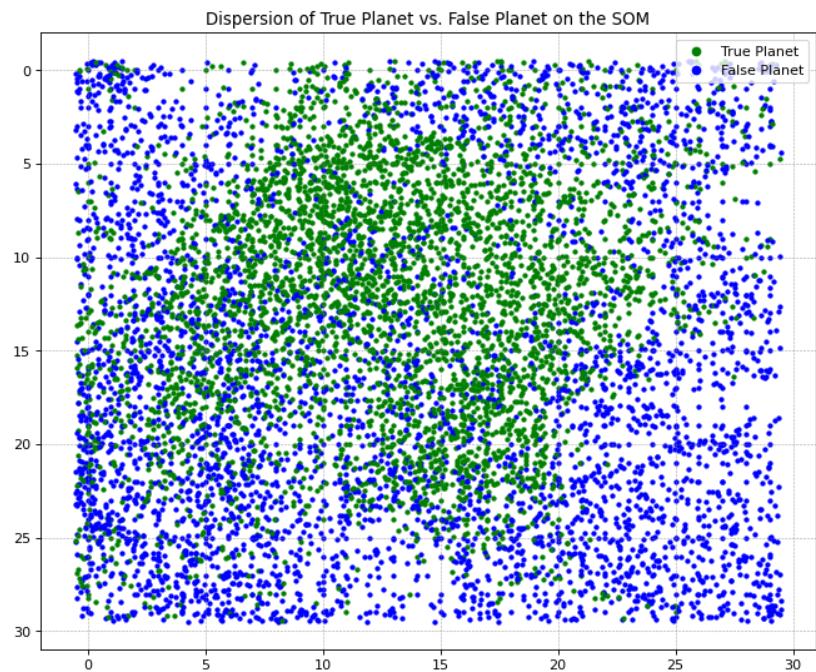


Figura 4.40. Dispersione di Veri Pianeti e Falsi Pianeti in corrispondenza del BMU (Test 4 SOM).

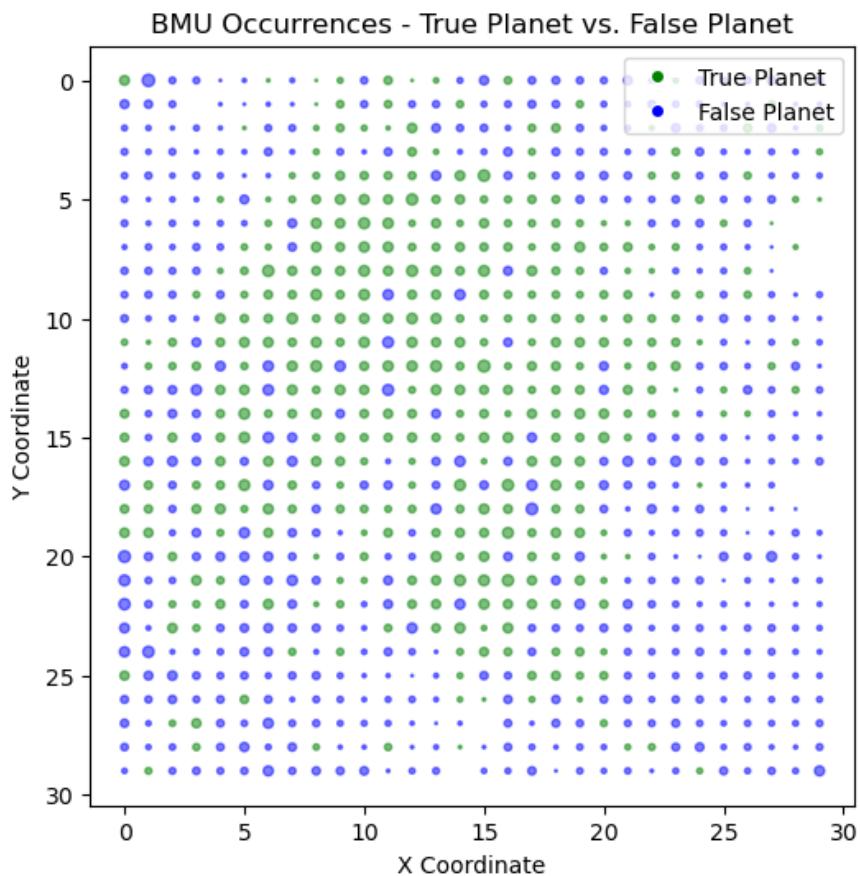


Figura 4.41. Distribuzione di Veri Pianeti e Falsi Pianeti in corrispondenza della Best Matching Unit (BMU), con evidenza delle relative occorrenze (Test 4 SOM).

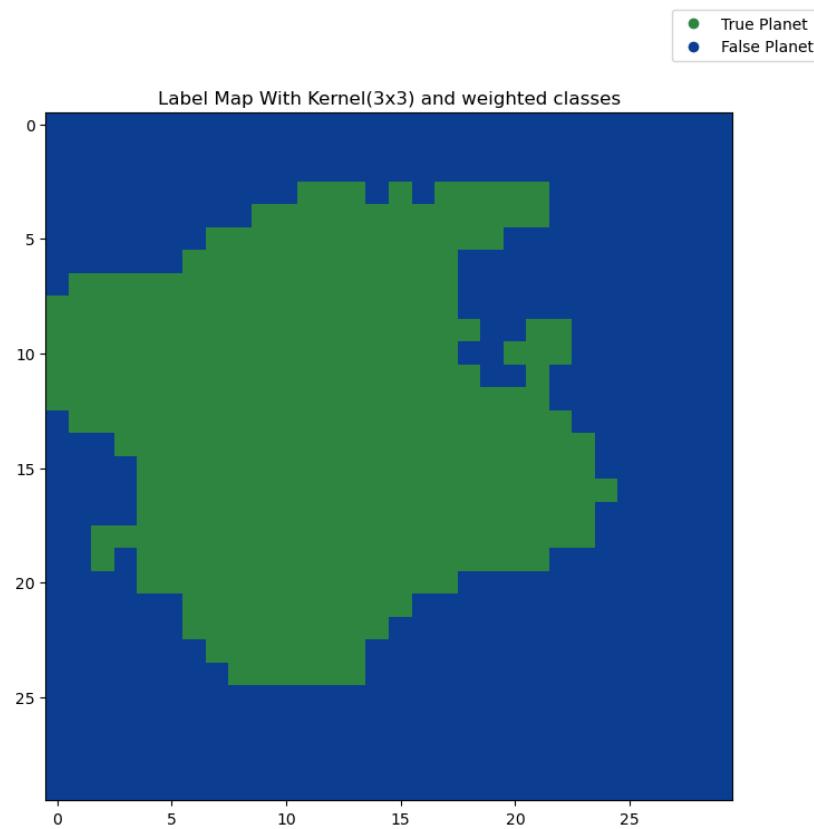


Figura 4.42. *Label Map* ricostruita sulla base di un'intorno 3x3 (Test 4 SOM).

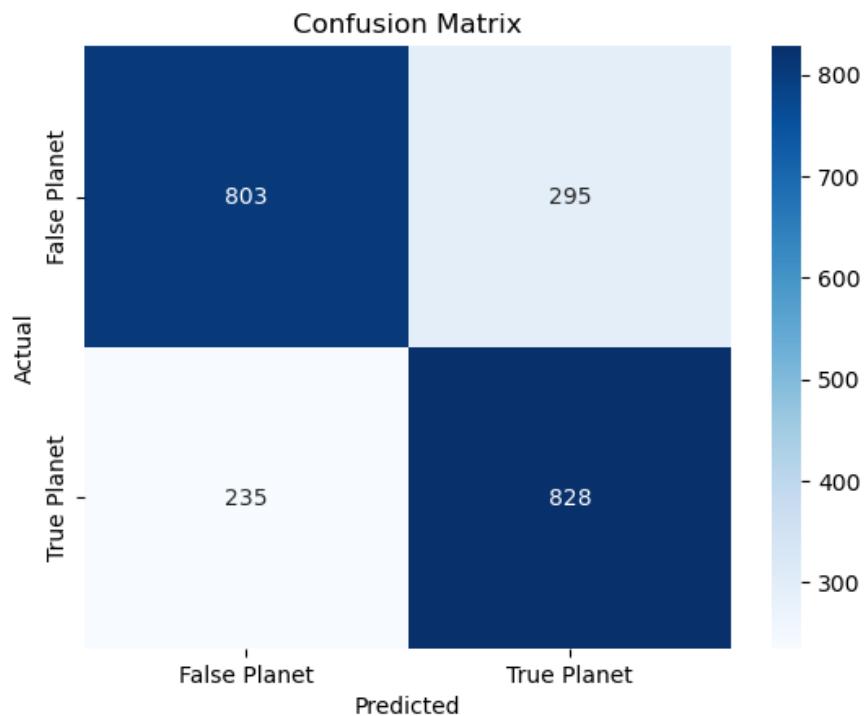


Figura 4.43. Matrice di confusione (Test 4 SOM).

Risultati Self-Organizing Maps

Nella Tabella 4.9, viene presentato un riassunto dei risultati ottenuti attraverso l'utilizzo della Self-Organizing Map. Si evidenzia che il primo test, che ha considerato l'intero dataset, è risultato, seppur nell'ordine dei decimali, il migliore. Tuttavia va considerata la natura sbilanciata del set di dati, a differenza dei Test 2 e 4.

	Accuratezza (%)	F1 Score (%)	Precisione (%)	Recall (%)
Test 1	77,57	82,52	78,01	87,59
Test 2	77,04	78,03	73,72	82,87
Test 3	76,15	81,11	77,85	84,66
Test 4	75,47	75,75	73,73	77,89

Tabella 4.9. Risultati ottenuti dai vari test tramite Self-Organizing Map.

4.5 Risultati

In questa sezione sono presentati i risultati ottenuti mediante approcci sia supervisionati che non, sul *dataset* combinato di TESS e Kepler denominato TESS/Kepler (dettagliato nella Tabella 4.4). La relativa tabella è la 4.10. Emergono i risultati superiori del Random Forest su tutte le metriche di valutazione.

	Accuratezza (%)	F1 Score (%)	Precisione (%)	Recall (%)
Random Forest	83,69	86,92	84,37	89,63
Gradient Boosting	82,97	86,40	83,51	89,51
Albero Decisionale	79,63	83,35	82,42	84,30
K Nearest Neighbors	79,49	84,09	79,19	89,63
SOM: Test 1	77,57	82,52	78,01	87,59
SOM: Test 2	77,04	78,03	73,72	82,87
SOM: Test 3	76,15	81,11	77,85	84,66
SOM: Test 4	75,47	75,75	73,73	77,89

Tabella 4.10. Risultati ottenuti mediante approcci sia supervisionati che non, sul *dataset* combinato di TESS e Kepler denominato TESS/Kepler.

Capitolo 5

Conclusioni

5.1 Osservazioni

In conclusione, le potenzialità delle tecniche di Machine Learning sono evidenti, anche se accompagnate da sfide significative come la necessità di un ampio *dataset* di alta qualità. Durante questo lavoro, abbiamo esplorato l'applicazione di diversi approcci di Machine Learning, tra cui: Alberi Decisionali, Random Forest, K-Nearest Neighbors, Gradient Boosting e Self-Organizing Maps (SOM), nel contesto della classificazione di pianeti e falsi pianeti utilizzando dati provenienti dai telescopi Kepler e TESS. Il *dataset* di TESS, considerato singolarmente, richiede ulteriori elaborazioni poiché è fortemente sbilanciato, con un numero di campioni della classe minoritaria insufficiente (consulta anche la Tabella 4.4). Tuttavia, l'aggiunta del catalogo di Kepler ha permesso di ottenere un *dataset* unito più generico, con un bilanciamento migliore tra le classi. Nonostante ciò, il *dataset* rimane leggermente sbilanciato. Il Random Forest ha dimostrato di essere un modello robusto, raggiungendo un'accuratezza del 83,69% (consulta anche la Tabella 4.10). Questo risultato è promettente e sottolinea l'efficacia di tale approccio nella classificazione di esopianeti. Un ulteriore vantaggio della Random Forest è la sua interpretabilità, che consente di comprendere il processo decisionale attraverso il concetto di "*gloss box*". Le Self-Organizing Maps (SOM) hanno contribuito in modo significativo alla comprensione della struttura dei dati. La visualizzazione dello spazio bidimensionale generato dalle SOM ha fornito una rappresentazione intuitiva di come le osservazioni si distribuiscono. In conclusione, l'impiego di Random Forest e SOM ha mostrato il potenziale di questi metodi nell'esplorare e comprendere i dati. Tuttavia, vi è ancora spazio per miglioramenti e nuove sfide, come la gestione di *dataset* più estesi, la ricerca di nuovi approcci e l'approfondimento delle caratteristiche, temi affrontati nella sezione 5.2. Il codice sorgente è disponibile nella repository GitHub ExoNet [51].

5.2 Sviluppi Futuri

Numerose sono le possibilità di miglioramento e sviluppo futuri per consolidare e ampliare gli obiettivi raggiunti durante questo elaborato di tesi. Alcune direzioni promettenti potrebbero includere:

- Utilizzo di Mappe Non Lineari: L'applicazione di mappe non lineari, come Isomap, potrebbe rappresentare un passo avanti significativo. Queste tecniche conservano le distanze geodetiche e possono gestire efficacemente il rumore gaussiano presente nei dati. L'utilizzo di mappe non lineari potrebbe migliorare ulteriormente la capacità del modello di individuare e classificare Veri Pianeti e Falsi Pianeti.
- Tecniche di *Data Augmentation*: Il *dataset* attuale, in particolare quello di TESS, è ancora limitato con 5,199 Veri Pianeti e circa 1,355 Falsi Pianeti, come descritto nella Tabella 4.1. Per affrontare questa limitazione, l'implementazione di tecniche di data augmentation potrebbe essere utile. L'aumento del *dataset* attraverso tecniche come Generative Adversarial Network (GAN) o modelli di diffusione potrebbe fornire un insieme più ampio e robusto di dati per l'addestramento del modello, migliorando la sua capacità di generalizzazione e classificazione.
- Approfondimento delle Caratteristiche: Ulteriori ricerche potrebbero concentrarsi sull'identificazione di caratteristiche astronomiche più informative e rilevanti. I valori nulli che si sono presentati, potrebbero essere sostituiti con dati reali, prelevandoli dal catalogo di Gaia.

Ringraziamenti

Ringrazio la mia famiglia per il costante sostegno, sia nei momenti felici che in quelli difficili. A mio padre, Pasquale, a mia madre, Emilia, e alle mie sorelle, Anna e Ludovica, va il mio più sentito ringraziamento per aver reso possibile il mio percorso accademico.

Un pensiero speciale va ai miei nonni, Anna, Pasquale, e alla nonna Betta, le vostre preziose presenze nella mia vita sono una fonte di ispirazione e saggezza. A tutti i miei familiari per il vostro affetto e il sostegno continuo.

Desidero esprimere il mio sincero ringraziamento alla Prof.ssa Laura Inno, sono particolarmente grato per il supporto ricevuto, la disponibilità dimostrata durante questo percorso e tutte le opportunità che mi sono state offerte. Inoltre, desidero ringraziare il Dott. Stefano Fiscale, che mi ha affiancato e guidato soprattutto nel percorso di tesi, mostrando grande disponibilità ed enfasi.

Desidero esprimere il mio sincero ringraziamento al Prof. Angelo Ciaramella e al Prof. Raffaele Montella. Grazie per la vostra preziosa guida, disponibilità, e per le numerose opportunità che ho avuto in questi tre anni. Avete reso il mio percorso di studio ricco di conoscenze ed esperienze, e sono profondamente grato per il vostro costante supporto.

A Sasy e Feffa, grazie per essere stati lì, anche quando forse non ne eravate consapevoli, e per avermi aiutato a superare sfide e a godere dei momenti di gioia. Siete stati una parte importante di questo capitolo della mia vita, e sono grato per ogni singolo momento condiviso, grazie di cuore. A Marco, Giuseppe, Franzese, Stefano, Giovanni, Massimo, Alessandra e a tutti i miei amici che hanno condiviso con me momenti di studio e relax. Grazie per aver reso indimenticabile il mio percorso universitario. Apprezzo davvero la vostra compagnia e i momenti spensierati che abbiamo trascorso insieme.

Un ringraziamento speciale va al gruppo di colleghi più straordinario, che nel corso del tempo è diventato una vera e propria famiglia. In ordine alfabetico Camilla, Carmine, Gaia, Giancarlo, Giuseppe, Lorenzo, Mario, Noemi, Raffaele, Vincenzo, Ylenia e a tutti coloro che hanno vissuto con me questa esperienza, grazie per il vostro supporto che è stato fondamentale e ha reso il mio percorso accademico un'esperienza unica e indimenticabile. Grazie Mario, per tutte le incredibili esperienze vissute tra progetti, eventi e premi i quali sono stati il cuore di questo percorso straordinario. È da qui che tutto è partito, e senza, tutto questo non sarebbe mai nato. Un ringraziamento particolare va a Core & Cafè per la gioia quotidiana che mi offrite con il vostro caffè.

Bibliografia

- [1] Christian Janiesch, Patrick Zschech, and Kai Heinrich. Machine learning and deep learning. *Electronic Markets*, 31(3):685–695, 2021.
- [2] About NASA. <https://www.nasa.gov/about/>.
- [3] What is an Exoplanet? In depth. <https://exoplanets.nasa.gov/what-is-an-exoplanet/in-depth/>.
- [4] What is an Exoplanet? <https://exoplanets.nasa.gov/faq/3/what-is-an-exoplanet/>.
- [5] Cos’è un transito? <https://exoplanets.nasa.gov/faq/31/whats-a-transit/#:~:text=Transits%20reveal%20an%20exoplanet%20not,over%20a%20period%20of%20time>.
- [6] Shawn Seader, Jon M Jenkins, Peter Tenenbaum, Joseph D Twicken, Jeffrey C Smith, Rob Morris, Joseph Catanzarite, Bruce D Clarke, Jie Li, Miles T Cote, et al. Detection of potential transit signals in 17 quarters of kepler mission data. *The Astrophysical Journal Supplement Series*, 217(1):18, 2015.
- [7] Gaia. <https://www.asi.it/esplorazione/cosmologia/gaia/>.
- [8] William J Borucki. Kepler mission: development and overview. *Reports on Progress in Physics*, 79(3):036901, 2016.
- [9] Transiting Exoplanet Survey Satellite. <https://exoplanets.nasa.gov/tess/>.
- [10] TESS Segment. <https://tess.mit.edu/science/observations/>.
- [11] Eetu Halsio. Extrasolar planets: detection methods and observational campaigns. B.S. thesis, E. Halsio, 2020.
- [12] Science area. <https://tess.mit.edu/science-area/>.
- [13] Follow-up observations. <https://tess.mit.edu/followup/>.
- [14] Nils J Nilsson. *Intelligenza artificiale*. Apogeo Editore, 2002.
- [15] Stuart J Russell and Peter Norvig. *Intelligenza artificiale. Un approccio moderno*, volume 1. Pearson Italia Spa, 2005.
- [16] Issam El Naqa and Martin J Murphy. *What is machine learning?* Springer, 2015.

- [17] Dingfu Zhou, Zhihang Liao, Rong Chen, et al. Deep learning enabled diagnosis of children's adhd based on the big data of video screen long-range eeg. *Journal of Healthcare Engineering*, 2022, 2022.
- [18] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- [19] Defu Zhang, Xiyue Zhou, Stephen CH Leung, and Jiemin Zheng. Vertical bagging decision trees model for credit scoring. *Expert Systems with Applications*, 37(12):7838–7843, 2010.
- [20] Sotiris B Kotsiantis. Decision trees: a recent overview. *Artificial Intelligence Review*, 39:261–283, 2013.
- [21] Steven J Rigatti. Random forest. *Journal of Insurance Medicine*, 47(1):31–39, 2017.
- [22] Candice Bentéjac, Anna Csörgő, and Gonzalo Martínez-Muñoz. A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, 54:1937–1967, 2021.
- [23] Shiliang Sun and Rongqing Huang. An adaptive k-nearest neighbor algorithm. In *2010 seventh international conference on fuzzy systems and knowledge discovery*, volume 1, pages 91–94. IEEE, 2010.
- [24] Oliver Kramer and Oliver Kramer. K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors*, pages 13–23, 2013.
- [25] Zhongheng Zhang. Introduction to machine learning: k-nearest neighbors. *Annals of translational medicine*, 4(11), 2016.
- [26] Jianglin Huang, Jacky Wai Keung, Federica Sarro, Yan-Fu Li, Yuen-Tak Yu, WK Chan, and Hongyi Sun. Cross-validation based k nearest neighbor imputation for software quality datasets: an empirical study. *Journal of Systems and Software*, 132:226–252, 2017.
- [27] Yanru Zhang and Ali Haghani. A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies*, 58:308–324, 2015.
- [28] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.
- [29] Teuvo Kohonen. Essentials of the self-organizing map. *Neural networks*, 37:52–65, 2013.
- [30] Salvador García, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez, and Francisco Herrera. Big data preprocessing: methods and prospects. *Big Data Analytics*, 1(1):1–22, 2016.
- [31] Taghi M Khoshgoftaar and Jason Van Hulse. Imputation techniques for multivariate missingness in software measurement data. *Software Quality Journal*, 16:563–600, 2008.

- [32] Pilar Rey-del Castillo and Jesús Cardeñosa. Fuzzy min–max neural networks for categorical data: application to missing data imputation. *Neural Computing and Applications*, 21(6):1349–1362, 2012.
- [33] Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.
- [34] Rupam Deb and Alan Wee-Chung Liew. Missing value imputation for the analysis of incomplete traffic accident data. *Information sciences*, 339:274–289, 2016.
- [35] Xiao-Yuan Jing, Fumin Qi, Fei Wu, and Baowen Xu. Missing data imputation based on low-rank recovery and semi-supervised regression for software effort estimation. In *Proceedings of the 38th International Conference on Software Engineering*, pages 607–618, 2016.
- [36] Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.
- [37] Markus Maier, Matthias Hein, and Ulrike Von Luxburg. Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters. *Theoretical Computer Science*, 410(19):1749–1764, 2009.
- [38] Carolyn Mair, Gada Kadoda, Martin Lefley, Keith Phalp, Chris Schofield, Martin Shepperd, and Steve Webster. An investigation of machine learning based prediction systems. *Journal of systems and software*, 53(1):23–29, 2000.
- [39] Ekrem Kocaguneli, Tim Menzies, and Jacky W Keung. On the value of ensemble effort estimation. *IEEE Transactions on Software Engineering*, 38(6):1403–1416, 2011.
- [40] Emilia Mendes, Ian Watson, Chris Triggs, Nile Mosley, and Steve Counsell. A comparative study of cost estimation models for web hypermedia applications. *Empirical Software Engineering*, 8:163–196, 2003.
- [41] Zheng Zhao, Fred Morstatter, Shashvata Sharma, Salem Alelyani, Aneeth Anand, and Huan Liu. Advancing feature selection research. *ASU feature selection repository*, pages 1–28, 2010.
- [42] David J Armstrong, Jevgenij Gamper, and Theodoros Damoulas. Exoplanet validation with machine learning: 50 new validated kepler planets. *Monthly Notices of the Royal Astronomical Society*, 504(4):5327–5344, 2021.
- [43] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.
- [44] Verayuth Lertnattee and Thanaruk Theeramunkong. Analysis of inverse class frequency in centroid-based text classification. In *IEEE International Symposium on Communications and Information Technology, 2004. ISCIT 2004.*, volume 2, pages 1171–1176. IEEE, 2004.
- [45] ExoFOP TESS Data. https://exofop.ipac.caltech.edu/tess/view_toi.php.

- [46] Kepler cumulative KOI. <https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-TblView?app=ExoTbls&config=cumulative>.
- [47] TESS NASA Documentation. https://exoplanetarchive.ipac.caltech.edu/docs/API_TOI_columns.html.
- [48] Kepler NASA Documentantion. https://exoplanetarchive.ipac.caltech.edu/docs/API_kepcandidate_columns.html.
- [49] Muhammad Murtadha Ramadhan, Imas Sukaesih Sitanggang, Fahrendi Rizky Nasution, and Abdullah Ghifari. Parameter tuning in random forest based on grid search method for gender classification based on voice frequency. *DEStech transactions on computer science and engineering*, 10(2017), 2017.
- [50] David J Armstrong, Maximilian N Günther, James McCormac, Alexis MS Smith, Daniel Bayliss, François Bouchy, Matthew R Burleigh, Sarah Casewell, Philipp Eigmüller, Edward Gillen, et al. Automatic vetting of planet candidates from ground-based surveys: machine learning with ngt. *Monthly Notices of the Royal Astronomical Society*, 478(3):4225–4237, 2018.
- [51] Attilio Di Vicino. ExoNet. <https://github.com/Attilio-Di-Vicino/ExoNet>, 2023.