

UNIVERSITÀ DEGLI STUDI DI SALERNO



Fondamenti di Intelligenza Artificiale

CORSO DI LAUREA TRIENNALE IN INFORMATICA



ENIA

Professore:

Fabio Palomba

Studenti:

Gerardo Frino (0512111971)

Benedetto Scala (0512110582)

Maria Lombardi (0512110039)

github repo: <https://github.com/benedettoscala/EnIA-FIA>

ANNO ACCADEMICO 2022/2023

Indice

1	Introduzione	3
1.1	Obiettivo del Sistema	3
1.2	Struttura del documento	3
2	Analisi del problema	4
2.1	CRISP-DM	5
3	Buisness Understanding	6
3.1	Specifica PEAS	6
3.1.1	Performance	6
3.1.2	Environment	6
3.1.3	Actuators	7
3.1.4	Sensors	7
4	Data Understanding	8
4.1	Scelta del dataset	8
4.2	Comprensione del dominio applicativo	8
4.2.1	Evapotraspirazione	8
4.2.2	Fabbisogno idrico delle colture: ETCrop	9
4.2.3	Coefficiente Di Coltura: Kc	10
4.2.4	Esempio	12
4.2.5	Il ruolo della pioggia	12
4.3	Produzione Del Dataset	14
5	Data Preparation	16
5.1	Data Cleaning	16
5.2	Feature Selection: Algoritmo Genetico	17
5.2.1	Inizializzazione	17
5.2.2	Selezione	18
5.2.3	Crossover	18

5.2.4	Mutazione	18
5.2.5	Stopping Condition	18
5.2.6	Funzione Di Fitness	18
5.2.7	Risultati	19
5.3	Feature Scaling	19
5.4	Data Balancing	20
5.4.1	Oversampling: SMOTE	21
5.4.2	Undersampling: Random Under Sampler	21
5.4.3	Risultati ottenuti	22
6	Modelling	23
6.1	Gaussian Naive Bayes	23
6.2	Decision Tree	24
6.3	Random Forest	24
7	Validazione	26
7.0.1	Divisione Test Set e Training Set	26
7.0.2	Metriche per la valutazione	27
7.1	Valutazione dei modelli	28
8	Deployment	30
8.1	Idea Principale: Servizio Reperibile sul WEB	30
8.2	Integrazione in EnIA	31

Capitolo 1

Introduzione

In un mondo in cui le risorse idriche diventano sempre più scarse e il cambiamento climatico rende l'agricoltura sempre più difficile, diventa fondamentale trovare nuovi modi per risparmiare acqua nell'agricoltura.

L'intelligenza artificiale (AI) rappresenta una delle opportunità più promettenti per raggiungere questo obiettivo. Con l'aiuto di algoritmi di apprendimento automatico e tecniche di elaborazione, l'IA può aiutare gli agricoltori a monitorare e ottimizzare l'uso dell'acqua, migliorando la produttività e la sostenibilità a lungo termine.

In questo testo esploreremo come l'IA può essere utilizzata per ridurre l'uso di acqua nell'agricoltura e migliorare la resilienza delle colture alle condizioni climatiche estreme.

1.1 Obiettivo del Sistema

L'obiettivo del progetto è quello di realizzare un agente autonomo in grado di consigliare all'utente se è vantaggioso irrigare o meno, attraverso l'utilizzo di un modello di machine learning supervisionato, il quale genera risultati attraverso una analisi dei dati passati in input.

1.2 Struttura del documento

Nelle sezioni successive si procederà con l'analisi del problema individuato, procedendo con la sua formalizzazione e descrizione dell'ambiente in cui opera. Una sezione sarà dedicata alle modalità scelte per il raccoglimento dei dati e alla creazione e struttura del Dataset risultante. In seguito, si descriverà la soluzione proposta, analizzando modelli, tecnologie e architetture utilizzati per tale scopo.

Capitolo 2

Analisi del problema

In questa sezione viene analizzato il problema individuato, la sua formalizzazione e descrizione mediante il modello PEAS. Oltre ciò, viene descritto l'ambiente in cui l'agente opera, e le sue principali caratteristiche.

La questione in esame sarebbe potuta essere risolta con semplicità attraverso l'implementazione di un algoritmo che, basandosi esclusivamente sulla quantità di pioggia, avrebbe determinato se irrigare o meno l'ambiente agricolo.

Tuttavia, tale soluzione avrebbe presentato numerose limitazioni, tra cui:

- La mancanza di considerazione di altri fattori importanti, come il tipo di coltivazione e il clima.
- l'irrigazione eccessiva o , all'altro estremo, insufficiente che avrebbe causato danni alla coltura e all'ambiente circostante.

Pertanto, è stata necessaria la formulazione di una metodologia più sofisticata che andasse a considerare una serie più grande di fattori per risolvere il quesito che ci siamo posti.

Si è deciso quindi di utilizzare un algoritmo di machine learning in quanto esso è in grado di considerare una vasta gamma di fattori e di apprendere continuamente dai dati storici e dalle condizioni attuali per prendere decisioni più accurate sull'irrigazione. Inoltre, l'utilizzo di un algoritmo di machine learning consente di adattarsi ai cambiamenti delle condizioni ambientali e di migliorare continuamente le decisioni prese.

L'idea alla base è stata quella di creare un sistema di irrigazione che utilizzasse diversi livelli di intensità, al fine di adattarlo alle esigenze specifiche del terreno e della coltura in questione.

I livelli di intensità di irrigazione citati in precedenza, sono stati considerati come particolari "etichette" e ci hanno permesso di interpretare il problema in questione come un caso di problema di classificazione.

2.1 CRISP-DM

Dato che stiamo trattando un modulo di machine learning, il modello di ciclo di vita che è stato utilizzato per arrivare alla soluzione è il CRISP-DM (Cross-Industry Standard Process for Data Mining). Il CRISP-DM è un modello non sequenziale in cui le diverse fasi possono essere eseguite un numero illimitato di volte.

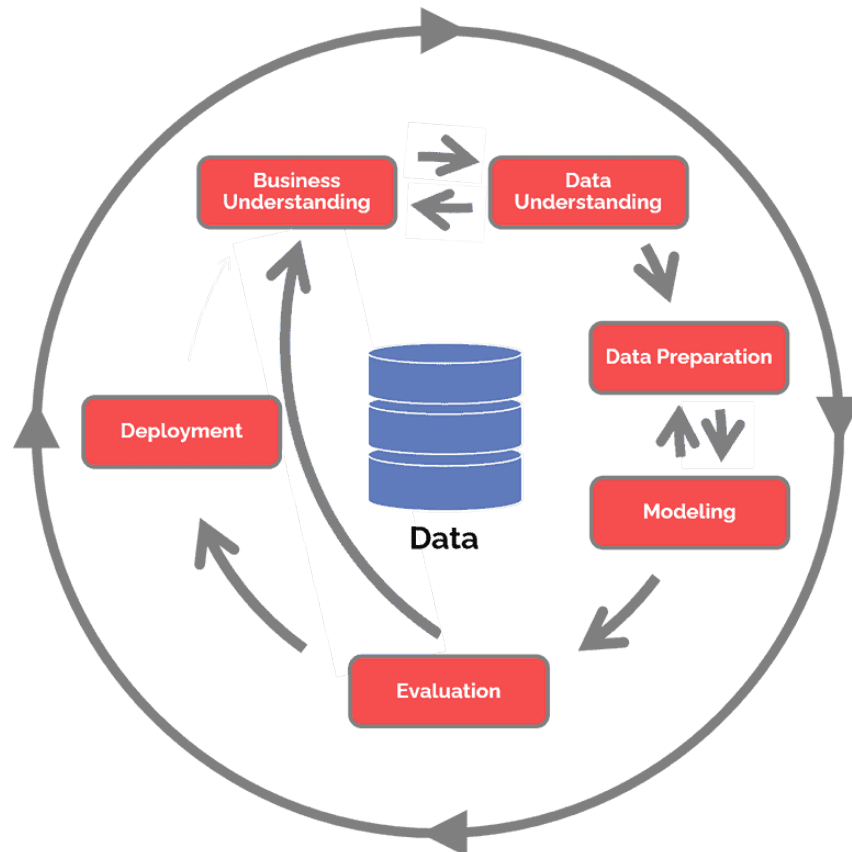


Figura 2.1: CRISP-DM

Capitolo 3

Buisness Understanding

3.1 Specifica PEAS

3.1.1 Performance

Le misure di valutazione dell'efficacia dell'agente sono basate sulle sue capacità di predire con precisione il livello di irrigazione che un determinato terreno dovrà ricevere in un giorno specifico. Tali metriche sono fondamentali per garantire che il terreno riceva la giusta quantità di acqua per garantire una crescita ottimale delle colture e una gestione efficiente delle risorse idriche.

3.1.2 Environment

L'ambiente operativo in cui agisce l'agente incaricato è costituito dal terreno destinato all'irrigazione, su cui è coltivata una specifica pianta, che presenta caratteristiche e necessità uniche in relazione alle sue esigenze di crescita e sviluppo. Il terreno in questione è caratterizzato, anche, dalle condizioni meteorologiche del luogo in cui si trova, che possono avere un'ulteriore influenza sulle caratteristiche del terreno stesso, come ad esempio la disponibilità di acqua e l'intensità dei fattori climatici. L'agente, pertanto, deve essere in grado di valutare e tenere conto di queste variabili per garantire un'irrigazione ottimale. Le caratteristiche che si possono osservare dell'ambiente sono quindi:

- Statico: In quanto le caratteristiche dell'ambiente non cambiano durante le deliberazioni dell'agente.
- Episodico: In quanto la decisione di procedere o meno con l'irrigazione è determinata esclusivamente dalle esigenze specifiche della coltura e dalle condizioni meteorologiche del luogo e non esistono stati precedenti o successivi alla decisione finale presa dall'agente

- Discreto: In quanto, l'insieme di percezioni che l'agente dovrà considerare sono limitate e chiaramente distinte, come per esempio: la quantità di pioggia in mm in un metro quadro o il coefficiente di crescita di una coltura.
- Completamente osservabile: Poiché la decisione di procedere con l'irrigazione del terreno è basata sulle previsioni meteorologiche o sulle caratteristiche di una coltura che sono informazione accessibili all'agente in ogni momento
- Non deterministico: In quanto l'ambiente è suscettibile a mutamenti indipendentemente dalle azioni intraprese dall'agente.
- A singolo agente: In quanto l'unico agente che opera in questo ambiente è quello di cui stiamo parlando

3.1.3 Actuators

Gli attuatori dell'agente consistono nel sistema informativo fornito dallo schermo di un qualsiasi computer, il quale fornisce all'utente consigli sulla gestione del proprio impianto idrico, consentendo di attivare o disattivare l'irrigazione e di scegliere l'intensità più adeguata nel caso in cui essa venga attivata..

3.1.4 Sensors

I sensori dell'agente includono sia le stazioni meteorologiche che i sensori nel terreno. Essi forniscono informazioni sulle condizioni meteorologiche del giorno corrente, tenendo conto dell'ambiente agricolo in esame, nonché sulle caratteristiche specifiche della coltura coltivata.

Capitolo 4

Data Understanding

4.1 Scelta del dataset

La ricerca di un dataset adeguato è stata lunga e poco conclusiva. Nonostante un'esauriente ricerca su piattaforme come Github, Reddit, Kaggle, non siamo riusciti a trovare un dataset che soddisfacesse le nostre esigenze. Pertanto, abbiamo deciso di optare per una soluzione diversa e cioè produrre manualmente il dataset richiesto.

4.2 Comprensione del dominio applicativo

Prima di procedere con l'elaborazione del dataset, risulta indispensabile stilare un elenco esaustivo dei concetti fondamentali relativi al mondo agricolo. Tale iniziale operazione consentirà non solo di comprendere appieno il contesto applicativo in cui ci siamo immersi, ma anche di introdurre euristiche che rivestiranno un ruolo essenziale per la determinazione dei dati richiesti.

4.2.1 Evapotraspirazione

Introduciamo il concetto fondamentale di evapotraspirazione, che rappresenta il processo mediante il quale le piante assorbono acqua dal suolo mediante le radici, trasmettendola in forma liquida agli apparati fogliari. A partire dal mesofillo fogliare, l'acqua si trasforma in vapore e viene rilasciata nell'atmosfera attraverso le aperture stomatiche delle foglie. Tale fenomeno viene definito traspirazione (T). Parallelamente, il suolo perde acqua a causa dell'evaporazione diretta (E). L'evapotraspirazione (ET) è la somma della quantità d'acqua persa dal suolo per evaporazione e dalle piante per traspirazione.

Il concetto di evapotraspirazione è strettamente correlato al fabbisogno idrico delle colture (ETCrop).

4.2.2 Fabbisogno idrico delle colture: ETCrop

Il fabbisogno idrico delle colture ET Crop è la quantità di acqua necessaria per soddisfare la perdita di acqua attraverso l'evapotraspirazione delle piante. In altre parole, rappresenta la quantità di acqua richiesta dalle colture per crescere in modo ottimale.

Il fabbisogno idrico delle colture dipende principalmente da:

- dal clima: in un clima soleggiato e caldo le colture hanno bisogno di più acqua al giorno rispetto a un clima nuvoloso e fresco
- dal tipo di coltura: colture come il mais o la canna da zucchero hanno bisogno di più acqua di colture come il miglio o il sorgo
- lo stadio di crescita della coltura: le colture completamente cresciute hanno bisogno di più acqua di quelle appena piantate

Il fabbisogno idrico delle colture può essere calcolato mediante la seguente formula:

$$ET_{crop} = ET_0 * Kc$$

ET0 è la quantità d'acqua (mm) evapotraspirata, in un determinato intervallo di tempo, da una superficie interamente coperta da una coltura ideale (festuca arundinacea: sostanzialmente un prato verde) con caratteristiche standard: fitta, bassa, uniforme, in piena attività vegetativa, posta in condizioni di rifornimento idrico del terreno ottimali.

Esistono vari metodi per calcolare ET0, tuttavia in questo testo non verranno considerati, poiché OpenMeteo, uno strumento per l'acquisizione di informazioni sulle condizioni meteorologiche che utilizzeremo per produrre il dataset, lo calcolerà automaticamente.”

4.2.3 Coefficiente Di Coltura: K_c

Il coefficiente di coltura (K_c), noto anche come fattore di coltura, che appare nella formula appena citata, rappresenta una costante che indica l'influenza del tipo di coltura sulla quantità di acqua richiesta per garantire la crescita ottimale delle piante.

K_c dipende principalmente da:

- il tipo di coltura
- lo stadio di crescita della coltura
- il clima

Partendo dal presupposto che la festuca arundinacea, la coltura di riferimento, ha un k_c uguale a 1, possiamo fare dei ragionamenti su altri tipi di colture, per esempio:

- Il mais completamente sviluppato, con la sua ampia superficie fogliare, sarà in grado di traspirare, e quindi utilizzare, più acqua rispetto a quella di riferimento. Il suo k_c sarà infatti uguale a 1,15
- Il cetriolo, anch'esso completamente sviluppato, utilizzerà meno acqua rispetto a quella di riferimento. Il suo k_c sarà uguale a 0,85

Il K_c è, quindi, un fattore moltiplicativo di correzione delle differenze che si riscontrano nell'evapotraspirazione di una certa coltura rispetto a quella di riferimento.

Possiamo, inoltre, suddividere lo stadio di crescita della coltura in 4 stadi:

- Stadio iniziale: questo è il periodo che intercorre tra la semina o il trapianto fino a quando la coltura copre circa il 10% del terreno
- Stadio di sviluppo: questo periodo inizia al termine dello stadio iniziale e dura fino al raggiungimento della piena copertura del suolo (copertura del suolo 70-80%).
- Stadio di mezza-stagione: questo periodo inizia alla fine della fase di sviluppo della coltura e dura fino alla maturità; include la fioritura e l'allegagione
- Stadio finale: il suo periodo inizia alla fine della fase di mezza stagione e si protrae fino all'ultimo giorno della raccolta.

Il clima influenza la durata del periodo di crescita totale e le varie fasi di crescita. In un clima fresco un certo raccolto crescerà più lentamente che in un clima caldo

Detto ciò, possiamo assegnare quattro valori diversi del k_c (uno per ogni stadio di crescita) alle colture:

Crop	Initial stage	Crop dev. stage	Mid-season stage	Late season stage
Barley/Oats/Wheat	0.35	0.75	1.15	0.45
Bean, green	0.35	0.70	1.10	0.90
Bean, dry	0.35	0.70	1.10	0.30
Cabbage/Carrot	0.45	0.75	1.05	0.90
Cotton/Flax	0.45	0.75	1.15	0.75
Cucumber/Squash	0.45	0.70	0.90	0.75
Eggplant/Tomato	0.45	0.75	1.15	0.80
Grain/small	0.35	0.75	1.10	0.65
Lentil/Pulses	0.45	0.75	1.10	0.50
Lettuce/Spinach	0.45	0.60	1.00	0.90
Maize, sweet	0.40	0.80	1.15	1.00
Maize, grain	0.40	0.80	1.15	0.70
Melon	0.45	0.75	1.00	0.75
Millet	0.35	0.70	1.10	0.65
Onion, green	0.50	0.70	1.00	1.00
Onion, dry	0.50	0.75	1.05	0.85
Peanut/Groundnut	0.45	0.75	1.05	0.70
Pea, fresh	0.45	0.80	1.15	1.05
Pepper, fresh	0.35	0.70	1.05	0.90
Potato	0.45	0.75	1.15	0.85
Radish	0.45	0.60	0.90	0.90
Sorghum	0.35	0.75	1.10	0.65
Soybean	0.35	0.75	1.10	0.60
Sugarbeet	0.45	0.80	1.15	0.80
Sunflower	0.35	0.75	1.15	0.55
Tobacco	0.35	0.75	1.10	0.90

La tabella precedente mostra i valori medi di K_c per le varie colture e fasi di crescita. In realtà, il K_c , come detto in precedenza, dipende anche dal clima e, in particolare, dall'umidità relativa e dalla velocità del vento. I valori sopra indicati devono essere ridotti di 0,05 se l'umidità relativa è alta ($UR > 80\%$) e la velocità del vento è bassa ($u < 2m/sec$), ad esempio se $K_c = 1,15$ diventa $K_c = 1,10$. I valori devono essere aumentati di 0,05 se l'umidità relativa è bassa ($UR < 50\%$) e la velocità del vento è elevata ($u > 5m/sec$), ad esempio $K_c = 1,05$ diventa $K_c = 1,10$.

4.2.4 Esempio

Una volta calcolati sia ET_0 , sia K_c , possiamo calcolare i valori richiesti di acqua in mm di una specifica coltura, attraverso la formula definita in precedenza:

$$ET_{crop} = ET_0 * K_c$$

Per esempio, se dovessimo calcolare l'ETCrop di una carota, mentre è nel suo stadio iniziale, tale che l' ET_0 sia 5.0 e con umidità relativa pari al 85% e velocità del vento pari a 1 m/sec, allora avremmo:

$$K_c = 0.45 - 0.05 = 0.40$$

e quindi:

$$ET_{Crop} = 0.40 * 5.0 = 2.25mm/giorno$$

La pianta di carota con queste condizioni richiederà 2,25 mm di acqua al giorno.

Tuttavia, è importante notare che l'utilizzo di questa formula non tiene conto dell'effetto della pioggia sul successo del raccolto. La pioggia infatti è una componente fondamentale per il buon sviluppo delle colture e la sua assenza può influire negativamente sulla resa finale. Pertanto, è importante considerare anche questo fattore quando si valuta l'efficacia di questa formula per il calcolo del rendimento del raccolto.

4.2.5 Il ruolo della pioggia

Oltre all'irrigazione, le precipitazioni sono un'altra fonte di acqua per le colture. Quando la pioggia fornisce tutta l'acqua necessaria per la crescita ottimale delle piante, non è necessaria l'irrigazione e il fabbisogno idrico per l'irrigazione (alias IN, e cioè la quantità d'acqua che bisogna somministrare alle colture attraverso l'irrigazione) è pari a zero: $IN = 0$.

Tuttavia, se durante la stagione di crescita non ci sono precipitazioni, l'acqua deve essere fornita interamente dall'irrigazione e il fabbisogno idrico per l'irrigazione (IN) corrisponde al fabbisogno idrico della coltura (ET crop): $IN = ET_{crop}$.

Nella maggior parte dei casi, però, le colture ricevono solo una parte dell'acqua di cui hanno bisogno dalle precipitazioni, mentre il resto viene fornito dall'irrigazione. In questi casi, il fabbisogno idrico per l'irrigazione (IN) è la differenza tra il fabbisogno idrico totale della coltura (ET crop) e l'acqua piovana effettivamente utilizzata dalle piante (Pe). In formula: $IN = ET_{crop} - Pe$.

Per calcolare Pe, dobbiamo fare alcune considerazioni:

Quando la pioggia cade sulla superficie del suolo, una parte di essa si infiltra nel terreno, una

parte ristagna sulla superficie, mentre una parte scorre sulla superficie.

Quando le precipitazioni cessano, parte dell'acqua che ristagna in superficie evapora nell'atmosfera, mentre il resto si infiltra lentamente nel terreno. Di tutta l'acqua che si infiltra nel suolo, una parte verrà immagazzinata dalle radici, mentre il resto verrà perso.

Le formule che verranno utilizzato per calcolare Pe saranno:

$$Pe = 0.8P - 0.33, \text{ if } P > 2.5 \text{ mm/giorno}$$

$$Pe = 0.6P - 0.83, \text{ if } P < 2.5 \text{ mm/giorno}$$

Quindi, prendendo l'esempio precedente sulla carota, supponendo che quel giorno siano precipitati 1.5 mm di acqua, avremmo che:

$$Pe = 0.6 * 1.5 - 0.83 = 0.07$$

quindi:

$$IN = ET_{Crop} - Pe = 2.25 - 0.07 = 2.18 \text{ mm/giorno}$$

4.3 Produzione Del Dataset

Dopo aver analizzato e compreso tutte le variabili relative al problema in esame, siamo stati in grado di produrre un dataset solido e completo, che ha soddisfatto pienamente le richieste. Abbiamo utilizzato l'API di Open meteo per reperire i dati meteo degli ultimi dieci anni relativi al comune di Fisciano. Le feature per ogni entry del dataset scaricato erano:

- surface_pressure (hPa): la pressione atmosferica
- soil_moisture_0_to_7cm (m^3/m^3): l'umidità del terreno dai 0 ai 7 cm di profondità
- et0_fao_evapotranspiration (mm): la reference crop evapotranspiration definita in precedenza
- rain (mm): mm di pioggia
- windspeed_10m (km/h): velocità del vento a 10 m di altezza dal terreno
- relativehumidity_2m (%): umidità relativa a 2 m di altezza dal terreno
- temperature_2m ($^{\circ}\text{C}$): temperatura media a 2 m di altezza dal terreno
- soil_temperature_0_to_7cm ($^{\circ}\text{C}$): la temperatura media del terreno dai 0 a 7 cm di profondità
- cloudcover (%): la copertura delle nuvole espressa in percentuale
- shortwave_radiation (W/m^2): radiazione ad onde corte espresso in Watt per metri quadri

Il dataset risultante scaricato era di questo tipo:

	0	1	2	3	4
time	2010-01-01	2010-01-02	2010-01-03	2010-01-04	2010-01-05
surface_pressure (hPa)	983.0	990.18	1001.91	1000.48	988.98
soil_moisture_0_to_7cm (m^3/m^3)	0.43	0.41	0.39	0.4	0.42
et0_fao_evapotranspiration (mm)	0.98	1.54	1.48	0.59	0.65
rain (mm)	37.9	5.1	0.0	11.3	7.7
windspeed_10m (km/h)	25.43	21.95	11.36	9.02	8.98
relativehumidity_2m (%)	83.54	76.42	59.25	77.58	87.79
temperature_2m ($^{\circ}\text{C}$)	11.3	10.77	7.72	7.63	12.6
soil_temperature_0_to_7cm ($^{\circ}\text{C}$)	12.21	11.5	10.01	8.84	11.65
cloudcover (%)	76.67	50.12	39.58	97.54	91.21
shortwave_radiation (W/m^2)	1144.0	1792.0	1931.0	739.0	1105.0

Abbiamo poi assegnato ad ogni entry del dataset, mediante uno script python:

- in maniera casuale, una delle colture definite in precedenza (crop) e l'attributo stadio di crescita (stageOfGrowth).
- l'attributo Kc (fattore della coltura), basato sul tipo di coltura e sullo stadio di crescita, nonché sulla velocità del vento e sull'umidità relativa
- l'attributo irrigation, calcolato mediante le euristiche che abbiamo descritto precedentemente

L'attributo "irrigation" è stato poi ulteriormente suddiviso in quattro categorie:

- Nessuno (0): per i valori di IN uguali a 0 mm/giorno
- Basso (1): per i valori di IN che vanno da 0 mm/giorno a 1,5 mm/giorno
- Medio (2): per i valori di IN che vanno da 1,5 mm/giorno a 3,0 mm/giorno
- Elevato (3): per i valori di IN che sono superiori a 3,0 mm/giorno

Il dataset finale con "irrigation" come variabile dipendente è il seguente in figura:

	0	1	2	3
time	2010-01-01	2010-01-02	2010-01-03	2010-01-04
surface_pressure (hPa)	983.0	990.18	1001.91	1000.48
soil_moisture_0_to_7cm (m ³ /m ³)	0.43	0.41	0.39	0.4
et0_fao_evapotranspiration (mm)	0.98	1.54	1.48	0.59
rain (mm)	37.9	5.1	0.0	11.3
windspeed_10m (km/h)	25.43	21.95	11.36	9.02
relativehumidity_2m (%)	83.54	76.42	59.25	77.58
temperature_2m (°C)	11.3	10.77	7.72	7.63
soil_temperature_0_to_7cm (°C)	12.21	11.5	10.01	8.84
cloudcover (%)	76.67	50.12	39.58	97.54
shortwave_radiation (W/m ²)	1144.0	1792.0	1931.0	739.0
crop	Bean	Cabbage	Carrot	Eggplant
stageOfGrowth	CropDevStage	InitialStage	MidSeasonStage	LateSeasonStage
Kc	0.7	0.45	1.05	0.8
irrigation	0	0	2	0

Capitolo 5

Data Preparation

La data preparation è una delle fasi fondamentali nella creazione di un modello di machine learning. Questa fase consiste nel raccogliere, pulire, trasformare e organizzare i dati in modo da renderli utilizzabili dal modello.

La qualità dei dati utilizzati nel processo di machine learning è essenziale per il successo del modello. Se i dati non sono accurati, incompleti o non rappresentativi, il modello produrrà previsioni inaffidabili e poco utili.

In questa sezione verranno analizzati i vari passaggi necessari per preparare i dati per il modello. Ci si concentrerà sulla raccolta dei dati, la pulizia dei dati, l'analisi dei dati e la loro trasformazione in un formato adatto al modello di machine learning.

Verranno inoltre descritti gli strumenti e le tecniche, come ad esempio l'utilizzo di librerie specifiche per la pulizia dei dati, la normalizzazione dei dati e la gestione dei dati mancanti.

5.1 Data Cleaning

Il data cleaning, o pulizia dei dati, è un processo fondamentale nella preparazione dei dati per l'analisi e il machine learning. Consiste nel valutare e correggere eventuali errori, incongruenze e incoerenze nei dati, al fine di ottenere un set di dati accurato e affidabile.

Durante la raccolta dei dati, è comune che si verifichino errori o incongruenze. Ad esempio, possono essere presenti valori mancanti, dati duplicati, valori errati o formati di dati diversi. Tali problemi possono influenzare negativamente l'accuratezza dei risultati dell'analisi o del modello di machine learning.

Dato che il nostro, è un dataset artificiale prodotto da noi stessi, non abbiamo avuto istanze di valori nulli, ne tantomeno duplicate.

5.2 Feature Selection: Algoritmo Genetico

La fase di feature selection è una fase importante dell'addestramento di un modello di machine learning. In questa fase, si selezionano le variabili che hanno un impatto significativo sulla variabile dipendente.

È stata scelta una soluzione basata su algoritmi genetici, che sono un tipo di algoritmo euristico utilizzato per risolvere problemi di ottimizzazione

Gli algoritmi genetici sono ispirati al principio della selezione naturale ed evoluzione biologica teorizzato nel 1859 da Charles Darwin.

Il GA ci ha permesso di scegliere le feature che rendessero più performanti i modelli di ML.

5.2.1 Inizializzazione

Si è scelto di utilizzare un array di interi per codificare il problema. L'array di interi presenta 13 posizioni, 1 per ogni attributo presente nel dataset, esclusa la variabile dipendente irrigazione.

Ogni posizione è di tipo booleano, e può quindi assumere soltanto due valori: 0 o 1

Quando l'*i*-esimo elemento dell'array è 0, significa che la corrispondente feature non verrà considerata, mentre se è 1, la feature sarà considerata.

Quindi dato l'array di feature del tipo:

```
['surface_pressure(hPa)', 'soil_moisture_0_to_7cm(m/m)', 'et0_fao_evapotranspiration(mm)',  
  'rain (mm)', 'windspeed_10m (km/h)', 'relativehumidity_2m (%)', 'temperature_2m  
  (°C)', 'soil_temperature_0_to_7cm (°C)', 'cloudcover (%)', 'shortwave_radiation (W/m²)',  
  'crop', 'stageOfGrowth', 'Kc']
```

con questo array di codifica:

```
[0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1]
```

considererà queste feature:

```
['et0_fao_evapotranspiration(mm)', 'rain(mm)', 'cloudcover(%)', 'crop', 'stageOfGrowth', 'Kc']
```

La size della popolazione per decision tree e per gaussian naive bayes è stata posta a 25, mentre per l'algoritmo random forest, data la sua complessità temporale molto più costosa rispetto agli altri 2, è stata ridotta a 10. Grazie a questo cambiamento si è passati da un tempo di esecuzione di quasi 15 minuti ad uno di 4 minuti.

5.2.2 Selezione

Un algoritmo di selezione indica come avviene la selezione degli individui da poter portare nella prossima generazione. È stato utilizzato la tournament selection: Il processo di tournament selection prevede la selezione di un numero prefissato di individui casuali dal pool di candidati (nel nostro caso si è scelto di sceglierne 3). Questi individui vengono quindi confrontati tra loro e viene scelto il migliore in base alla sua idoneità (fitness). Il vincitore viene quindi selezionato come genitore per il crossover, e il processo viene ripetuto fino a quando non sono stati selezionati abbastanza genitori per la generazione successiva.

5.2.3 Crossover

È stato deciso di utilizzare l'algoritmo one-point in cui si va a selezionare un punto del patrimonio genetico dei genitori e si procede alla generazione di due figli tramite scambio di cromosomi.

5.2.4 Mutazione

Una mutazione è un'operazione che modifica casualmente uno o più geni di un individuo nella popolazione. La mutazione simula il processo biologico di mutazione genetica che può verificarsi nella natura, dove il materiale genetico di un organismo subisce modifiche casuali. Abbiamo deciso di utilizzare la mutazione bit flip che consiste nella modifica di un singolo gene binario.

5.2.5 Stopping Condition

L'algoritmo termina dopo 100 iterazioni, o dopo 40 iterazioni senza miglioramenti

5.2.6 Funzione Di Fitness

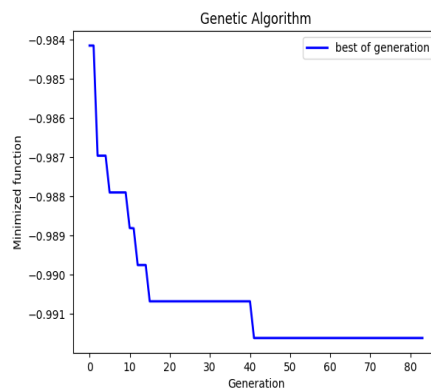
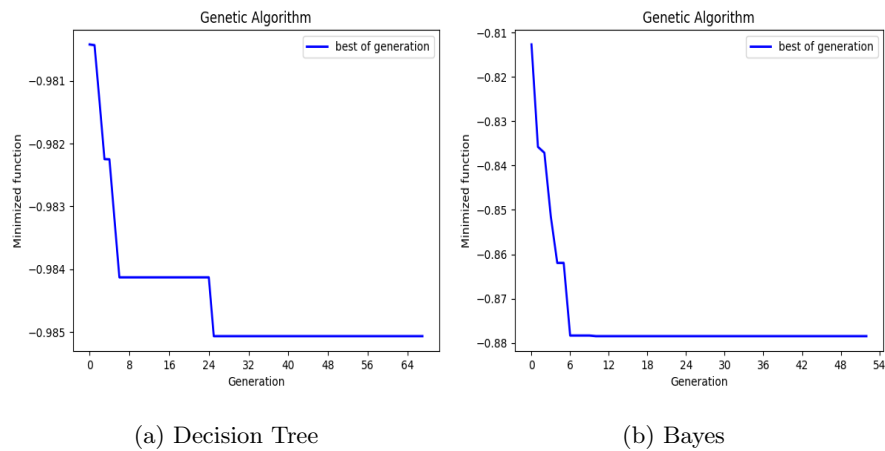
La funzione di fitness misura il grado di conformità di una soluzione rispetto al problema. Nel nostro caso la funzione di fitness restituisce il valore dell'f-score (media ponderata tra accuracy e recall) che il modello assume quando viene inizializzato con i parametri dell'algoritmo genetico.

5.2.7 Risultati

Dato che ogni algoritmo di addestramento funziona diversamente rispetto ad altri, si è deciso di utilizzare la ricerca genetica per trovare per ogni machine learner le feature più adatte. Per quanto riguarda il decision tree abbiamo riscontrato miglioramenti con le seguenti feature 'et0_fao_evapotranspiration (mm)', 'rain (mm)', 'temperature_2m (°C)', 'Kc'.

Parlando del modello Gaussian Naive Bayes, abbiamo riscontrato miglioramenti con le seguenti feature 'et0_fao_evapotranspiration (mm)', 'rain (mm)', 'stageOfGrowth', 'Kc'.

Infine, l'algoritmo random forest ha ricevuto miglioramenti con le medesime feature del modello gaussian naive bayes, e cioè 'et0_fao_evapotranspiration (mm)', 'rain (mm)', 'stageOfGrowth', 'Kc'.



(c) Random Forest

5.3 Feature Scaling

Il feature scaling è una tecnica di pre-elaborazione dei dati che consiste nel modificare il range di variazione delle caratteristiche (feature) di un dataset. Questo può essere utile per addestrare un modello di machine learning perché alcuni algoritmi sono sensibili alla scala delle feature e possono avere prestazioni migliori se le feature sono normalizzate.

Il tipo di normalizzazione che verrà utilizzato nel contesto di questo progetto è lo scaling min-max.

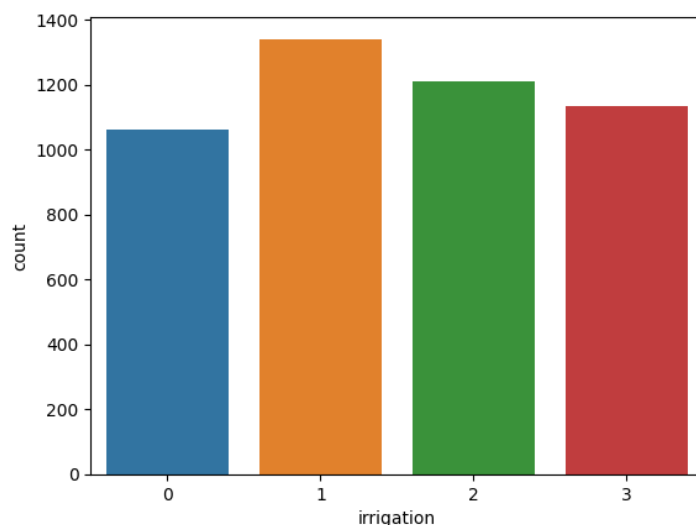
Lo scaling min-max è un tipo di feature scaling che trasforma le feature in modo che abbiano valori compresi tra 0 e 1, secondo la formula:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Qui, X_{max} e X_{min} rappresentano rispettivamente il valore massimo e minimo della variabile.

5.4 Data Balancing

Il data balancing nella preparazione dei dati di un modello di machine learning è una tecnica utilizzata per bilanciare il numero di istanze di ogni classe della variabile dipendente. Questo è utile quando si lavora con dati sbilanciati, ovvero quando una classe di dati ha un numero di esempi molto maggiore rispetto alle altre classi. Nel nostro caso, come si può vedere dal grafico, il dataset presenta un certo grado di sbilanciamento:



Ci sono tre possibili soluzioni che avremmo potuto adottare.

- lasciare il dataset invariato
- utilizzare una tecnica di oversampling
- utilizzare una tecnica di undersampling

Abbiamo deciso di sperimentare attraverso il metodo empirico quale fosse la tecnica di bilanciamento migliore.

5.4.1 Oversampling: SMOTE

SMOTE (Synthetic Minority Over-sampling Technique) è un algoritmo di over-sampling utilizzato per risolvere il problema di sbilanciamento delle classi in un set di dati. L'obiettivo di SMOTE è generare nuovi campioni sintetici della classe di minoranza utilizzando i campioni già esistenti.

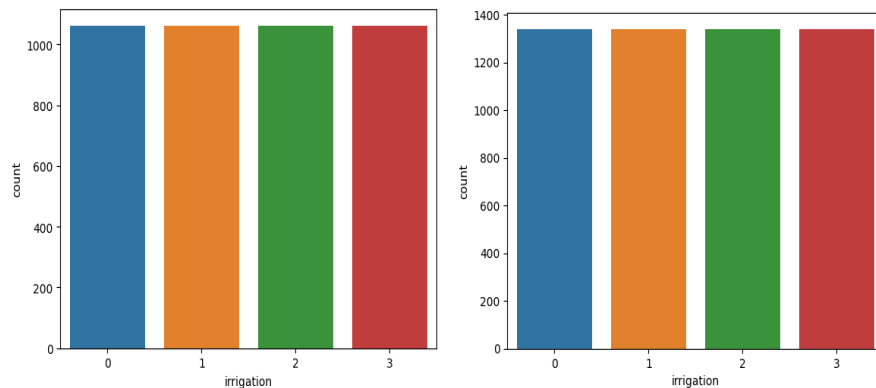
Il processo di SMOTE può essere descritto come segue:

1. Per ogni campione della classe di minoranza, si selezionano k vicini più prossimi all'interno della stessa classe.
2. Per ogni campione della classe di minoranza, si seleziona uno dei k vicini casualmente e si calcola la differenza tra il campione e il vicino selezionato.
3. Si moltiplica la differenza per un numero casuale tra 0 e 1 e si somma il risultato al campione originale.
4. Si ripetono i passaggi 2 e 3 fino a quando non si raggiunge il numero desiderato di campioni sintetici.

In questo modo, SMOTE genera campioni sintetici che si trovano lungo le linee che collegano i campioni della classe di minoranza esistenti. Questi nuovi campioni sono poi utilizzati per aumentare il numero di campioni nella classe di minoranza, migliorando così la performance del modello di classificazione

5.4.2 Undersampling: Random Under Sampler

Random Under Sampler(RUS) è una tecnica di undersampling che consiste nel selezionare casualmente esempi dalla classe di maggioranza e eliminarli dal set di dati di addestramento. In questo modo, il numero di esempi della classe di maggioranza viene ridotto e il set di dati diventa bilanciato



(a) SMOTE

(b) RUS

5.4.3 Risultati ottenuti

L'applicazione delle tecniche di bilanciamento menzionate ha portato a miglioramenti modesti, come si può vedere di seguito. Anche se tali miglioramenti non sono eccezionali, sono comunque significativi. UNTOUCHED sta per il dataset non bilanciato, RUS per il dataset bilanciato con random under sampler e SMOTE per il dataset bilanciato con SMOTE.

<pre>-----UNTOUCHED SCORES----- average precision: 0.9786891543916452 average recall: 0.9787051951867145 average f1-score: 0.978651855263497 average accuracy score: 0.9783097997892518 -----RUS SCORES----- average precision: 0.9773147480826753 average recall: 0.9773798351110964 average f1-score: 0.9772850413666634 average accuracy: 0.9774028961407885 -----SMOTE SCORES----- average precision: 0.9812145019941158 average recall: 0.9810789971344164 average f1-score: 0.9811146993904728 average accuracy: 0.9811716696109387</pre>	<pre>-----UNTOUCHED SCORES----- average precision: 0.865035832213595 average recall: 0.8621734281378867 average f1-score: 0.8624212782162843 average accuracy score: 0.8576551494648106 -----RUS SCORES----- average precision: 0.8691451159112725 average recall: 0.8688523984922991 average f1-score: 0.8680261687582256 average accuracy: 0.8681770941592184 -----SMOTE SCORES----- average precision: 0.8722516100126054 average recall: 0.8718043826822193 average f1-score: 0.8715284949383575 average accuracy: 0.8715531151326313</pre>
--	--

(a) Decision Tree

(b) Bayes

<pre>-----UNTOUCHED SCORES----- average precision: 0.981924341444999 average recall: 0.9822530886899099 average f1-score: 0.9820456391740295 average accuracy score: 0.9816795518828684 -----RUS SCORES----- average precision: 0.9770805979516448 average recall: 0.977132438255676 average f1-score: 0.9770509956275483 average accuracy: 0.9771681563084597 -----SMOTE SCORES----- average precision: 0.9855323943442231 average recall: 0.9854387365577608 average f1-score: 0.9854657231514075 average accuracy: 0.9854578459056071</pre>

(c) Random Forest

Si può notare un leggero miglioramento in tutti i casi utilizzando SMOTE, pertanto verrà adottata questa tecnica.

Capitolo 6

Modelling

In questa sezione ci concentreremo sull'addestramento di diversi modelli di machine learning al fine di esplorare le loro potenzialità. Esploreremo diverse tecniche di addestramento e valutazione dei modelli, approfondendo le loro caratteristiche e peculiarità. Verranno utilizzati gli algoritmi messi a disposizione da scikit learn.

6.1 Gaussian Naive Bayes

L'algoritmo di machine learning Naive Bayes è un algoritmo di classificazione probabilistica che si basa sul teorema di Bayes. L'obiettivo dell'algoritmo è quello di classificare istanze di dati in classi predefinite, utilizzando un set di caratteristiche (feature) che descrivono ogni istanza.

In sostanza, l'algoritmo di Naive Bayes calcola la probabilità di ogni classe data una istanza di dati, utilizzando la regola di Bayes. Il classificatore assume che tutte le caratteristiche siano indipendenti tra loro, anche se questo può non essere del tutto vero nella realtà (da qui il termine "naive" o "ingenuo").

Per calcolare la probabilità di una classe, l'algoritmo utilizza una funzione di verosimiglianza per ogni caratteristica, che indica quanto è probabile che una data caratteristica abbia un determinato valore per una classe specifica. Successivamente, viene utilizzata la regola di Bayes per combinare le probabilità delle caratteristiche e calcolare la probabilità finale di una classe.

L'algoritmo di Naive Bayes è particolarmente utile quando si hanno dati di grandi dimensioni, in cui la complessità computazionale di altri algoritmi di machine learning potrebbe essere proibitiva. Inoltre, il classificatore può funzionare bene anche quando le assunzioni di indipendenza tra le caratteristiche non sono del tutto rispettate.

Il termine "Gaussian" nel Gaussian Naive Bayes si riferisce all'assunzione che le caratteristiche di input siano distribuite secondo una distribuzione normale o gaussiana. Questa assunzione semplifica il calcolo delle probabilità condizionate, poiché la distribuzione gaussiana è completamente caratterizzata dalla media e dalla varianza.

6.2 Decision Tree

L'algoritmo di apprendimento automatico "Decision Tree" è un algoritmo di classificazione e regressione utilizzato per costruire un modello predittivo che prende decisioni gerarchiche utilizzando un albero. L'albero è composto da nodi interni che rappresentano le decisioni basate sul valore di una determinata caratteristica di input, e da foglie che rappresentano le classi o i valori di output.

L'obiettivo dell'algoritmo Decision Tree è quello di suddividere il dataset in gruppi omogenei in base alle caratteristiche di input, in modo che ogni gruppo abbia una maggiore similarità all'interno e una maggiore diversità tra i gruppi. L'algoritmo seleziona iterativamente la caratteristica di input che fornisce la suddivisione migliore del dataset, in modo da massimizzare l'omogeneità all'interno di ciascun gruppo e la diversità tra i gruppi. La selezione della caratteristica di input viene fatta utilizzando una metrica di impurità, come ad esempio l'entropia o l'indice di Gini.

Una volta che la migliore caratteristica di input è stata selezionata, il dataset viene suddiviso in due sottoinsiemi in base al valore della caratteristica selezionata. Questa operazione viene ripetuta ricorsivamente sui due sottoinsiemi fino a quando tutti i dati in ciascun sottoinsieme appartengono alla stessa classe o hanno lo stesso valore di output. In tal caso, la foglia corrispondente viene etichettata con la classe o il valore di output.

Una volta che l'albero è stato costruito, viene utilizzato per classificare o predire il valore di output di nuovi dati. L'algoritmo attraversa l'albero partendo dalla radice e seguendo le decisioni basate sul valore delle caratteristiche di input. Alla fine, l'algoritmo arriva a una foglia che fornisce la classificazione o la predizione del valore di output.

6.3 Random Forest

L'algoritmo di apprendimento automatico "Random Forest" è un algoritmo di apprendimento supervisionato che si basa sulla creazione di un insieme di alberi decisionali. L'idea alla base dell'algoritmo è di costruire molteplici alberi decisionali su sottoinsiemi differenti dei dati di addestramento, e quindi combinare le previsioni di questi alberi per produrre una previsione finale.

In particolare, l'algoritmo Random Forest seleziona casualmente una sottoinsieme di caratteristiche dal dataset di addestramento per ogni albero decisionale da costruire. Questo processo è chiamato "bagging" (bootstrap aggregating). Inoltre, l'algoritmo utilizza anche il "random subspace method", ovvero la selezione casuale di un sottoinsieme di esempi di addestramento per ciascuna caratteristica.

Per ogni albero decisionale, l'algoritmo Random Forest costruisce l'albero utilizzando una suddivisione ricorsiva dei dati in base alle caratteristiche selezionate e utilizzando un criterio

di impurità, come l'indice di Gini o l'entropia. La creazione di ogni albero avviene in modo indipendente dagli altri alberi.

Una volta che tutti gli alberi sono stati costruiti, l'algoritmo Random Forest combina le previsioni di ciascun albero per ottenere una previsione finale. La combinazione delle previsioni può essere effettuata attraverso la votazione maggioritaria, ovvero selezionando la classe o il valore di output che è stato predetto dalla maggior parte degli alberi. In alternativa, la combinazione delle previsioni può essere effettuata attraverso il calcolo della media delle previsioni.

Capitolo 7

Validazione

In questa sezione del documento andremo ad elencare le metodologie e le metriche che sono state utilizzate per validare le performance degli algoritmi utilizzati.

7.0.1 Divisione Test Set e Training Set

Uno degli step fondamentali nella validazione di un algoritmo di machine learning è quello della divisione tra i dati di addestramento e quelli di test. La divisione permette di testare l'algoritmo su dati che non sono stati utilizzati per l'addestramento e di valutarne le performance in modo obiettivo. In genere, si utilizza una proporzione di 70-80% per i dati di addestramento e il restante 20-30% per quelli di test. Abbiamo deciso di utilizzare per la nostra validazione la k-fold cross validation. In questa tecnica, il set di dati viene diviso casualmente in k subset (folding) uguali. Successivamente, si utilizzano k-1 subset per addestrare il modello e il subset rimanente per testarlo. Questa procedura viene ripetuta k volte, utilizzando ogni volta un subset differente come set di test.



Figura 7.1: Un esempio di K-Fold Cross Validation

Infine, si ottiene una media delle misure di performance, ottenute in ogni iterazione, per avere

un'idea generale della performance del modello. In questo modo, si cerca di evitare il rischio di overfitting, ovvero di addestrare il modello su dati specifici e non generalizzabili.

7.0.2 Metriche per la valutazione

Nella valutazione di un modello di machine learning, è importante disporre di misure che consentano di valutare l'accuratezza delle predizioni del sistema. Nel caso di un classificatore, le principali metriche utilizzate per valutare le prestazioni del modello sono la recall, la precisione, l'accuracy e l'f1-score.

- Accuratezza (Accuracy): è la frazione dei campioni correttamente classificati rispetto al totale dei campioni. Questa metrica funziona bene quando le classi sono bilanciate, ma può essere ingannevole quando ci sono classi sbilanciate.

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

- Precisione (Precision): è la proporzione di campioni classificati correttamente come positivi rispetto al totale dei campioni classificati come positivi. Ciò indica quanto spesso il classificatore ha ragione quando prevede una classe specifica.

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

- Recall (Recall): è la proporzione di campioni correttamente classificati come positivi rispetto al totale dei campioni appartenenti a quella classe. Questo indica quanto spesso il classificatore è in grado di trovare tutti i campioni appartenenti a una determinata classe.

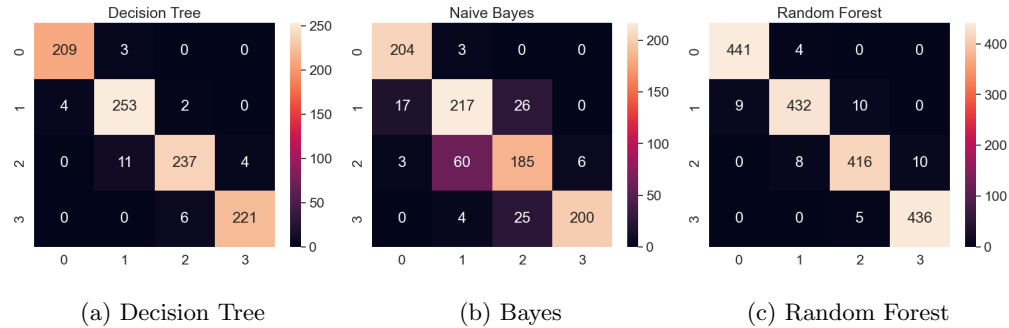
$$Recall = \frac{TP}{TP+FN}$$

- F1-score: è una media armonica tra precisione e recall.

$$F1\text{-score} = 2 * \frac{precision * recall}{precision + recall}$$

7.1 Valutazione dei modelli

Per comprendere al meglio le performance abbiamo deciso di generare le matrici di confusione di tutti e 3 i modelli.



Si osserva una significativa differenza di prestazioni tra il modello probabilistico e i due modelli basati su alberi. In particolare, Bayes mostra un rendimento decisamente inferiore rispetto ai suoi concorrenti a causa della sua ipotesi di indipendenza tra le diverse feature considerate. Sono state, poi, calcolate le metriche per tutti e tre i modelli di machine learning. Successivamente, è stato selezionato il modello migliore in base alle performance misurate mediante il metodo empirico e la tecnica k-fold, valutando gli errori commessi da ciascun algoritmo.

```

-----UNTOUCHED SCORES-----
average precision: 0.9786891543916452
average recall: 0.9787051951867145
average f1-score: 0.978651855263497
average accuracy score: 0.9783097997892518
-----RUS SCORES-----
average precision: 0.9773147480826753
average recall: 0.9773798351110964
average f1-score: 0.9772850413666634
average accuracy: 0.9774028961407885
-----SMOTE SCORES-----
average precision: 0.9812145019941158
average recall: 0.9810789971344164
average f1-score: 0.9811146993904728
average accuracy: 0.9811716696109387
-----UNTOUCHED SCORES-----
average precision: 0.865035832213595
average recall: 0.8621734281378867
average f1-score: 0.8624212782162843
average accuracy score: 0.8576551494648106
-----RUS SCORES-----
average precision: 0.8691451159112725
average recall: 0.8688523984922991
average f1-score: 0.8680261687582256
average accuracy: 0.8681770941592184
-----SMOTE SCORES-----
average precision: 0.8722516100126054
average recall: 0.8718043826822193
average f1-score: 0.8715284949383575
average accuracy: 0.8715531151326313

```

(a) Decision Tree

(b) Bayes

```

-----UNTOUCHED SCORES-----
average precision: 0.981924341444999
average recall: 0.9822530886899099
average f1-score: 0.9820456391740295
average accuracy score: 0.9816795518828684
-----RUS SCORES-----
average precision: 0.9770805979516448
average recall: 0.977132438255676
average f1-score: 0.9770509956275483
average accuracy: 0.9771681563084597
-----SMOTE SCORES-----
average precision: 0.9855323943442231
average recall: 0.9854387365577608
average f1-score: 0.9854657231514075
average accuracy: 0.9854578459056071

```

(c) Random Forest

Dalle metriche di valutazione emerge che il random forest rappresenta l'algoritmo con il miglior punteggio. Tuttavia, l'incremento marginale di prestazioni non sembra giustificare l'utilizzo di tale algoritmo, considerando il considerevole costo in termini di risorse richiesto. Pertanto, nella fase di deployment, si è optato per l'utilizzo del decision tree, che, seppur meno performante, risulta comunque una valida opzione.

Capitolo 8

Deployment

In questo capitolo spiegheremo come abbiamo implementato il nostro modello, fornendo una descrizione dettagliata dei tool e dei moduli Python utilizzati. Inoltre, illustreremo come abbiamo utilizzato il modello sulla piattaforma EnIA (Progetto Ingegneria Del Software)

8.1 Idea Principale: Servizio Reperibile sul WEB

Per implementare il deployment del modello di Machine Learning è stato necessario seguire alcuni passaggi fondamentali.

Innanzitutto si è fatto il dump del classificatore utilizzato su un file .pkl tramite il modulo python pickle.

Dopodichè, è stato scelto di utilizzare un framework web per la costruzione di un sito web accessibile all'utente mediante chiamate HTTP GET. In particolare, è stato scelto Flask, un framework leggero per applicazioni web in Python.

Successivamente, il dump del classificatore è stato caricato sul server, e poi è stato creato un endpoint per ricevere le richieste GET. L'utente può accedere all'endpoint passando alcuni parametri di input, tra cui la latitudine, la longitudine, il tipo di coltura e lo stadio di crescita della stessa

Una volta ricevuti i parametri, il server reperisce le previsioni dei prossimi 7 giorni basandosi sulla latitudine e la longitudine inseriti dall'utente utilizzando l'API di Open Meteo, dopodichè li dà in pasto al modello di Machine Learning per elaborare i dati e generare una previsione dell'irrigazione. La previsione viene quindi formattata in un file JSON, che viene inviato come risposta al client che ha fatto la richiesta GET.

Il file JSON contiene la quantità di acqua consigliata per l'irrigazione. In questo modo, l'utente può avere una panoramica completa sulle previsioni per i prossimi giorni e agire di conseguenza per l'irrigazione delle piante.

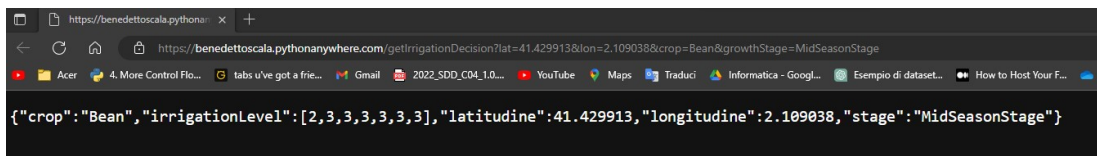
Inoltre, per il deployment del sito web è stato utilizzato PythonAnywhere, un servizio di hosting

web per applicazioni Python. PythonAnywhere ha permesso di eseguire l'applicazione web in un ambiente di produzione stabile e sicuro, garantendo la disponibilità del sito web 24 ore su 24 e 7 giorni su 7.

Per richiedere i consigli del modello sull'irrigazione per una coltura di fagioli durante la mezza stagione, in una determinata area geografica rappresentata da una latitudine di 41.42 e una longitudine di 2.10, è possibile utilizzare l'URL:

`https://benedettoscala.pythonanywhere.com/getIrrigationDecision?lat=41.429913&lon=2.109038&crop=Bean&growthStage=MidSeasonStage`

Il quale restituirà per la data del 21/03/2023 questo risultato:



```
{ "crop": "Bean", "irrigationLevel": [2, 3, 3, 3, 3, 3, 3], "latitudine": 41.429913, "longitudine": 2.109038, "stage": "MidSeasonStage" }
```

Figura 8.1: Risposta json alla richiesta HTTP al server

8.2 Integrazione in EnIA

È importante precisare che il progetto di ingegneria del software EnIA (Environmental Intelligence For Agriculture) utilizza il modello di Machine Learning descritto tramite chiamate HTTP. Per accedere a tale modello, è stato sviluppato un adapter che invia le richieste HTTP al server Flask e elabora le risposte JSON ricevute.

In questo modo, l'utente può facilmente e intuitivamente accedere ai consigli sull'irrigazione tramite un'interfaccia grafica utente.

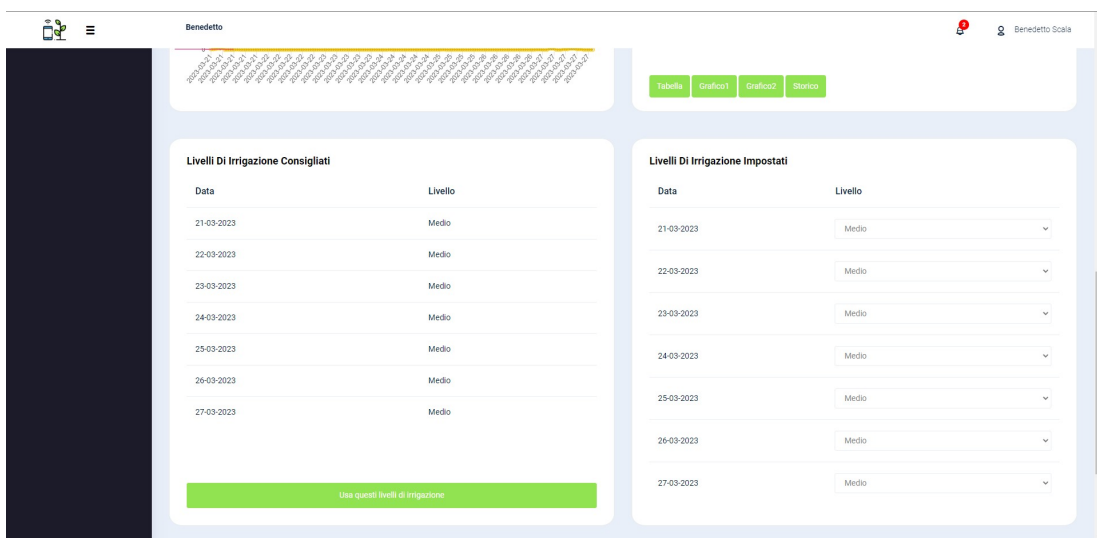


Figura 8.2: La pagina dedicata ai consigli sull'irrigazione su EnIA

Bibliografia

- [1] C. Brouwer, M. Heibloem *Irrigation Water Management: Irrigation Water Needs*, ©FAO 1986, <https://www.fao.org/3/s2022e/s2022e05.htm#chapter%201%20introduction>.
- [2] Carmelo Santonoceto, *Evapotraspirazione* Università degli studi di Reggio Calabria, Corso di Laurea Magistrale in Scienze e Tecnologie Agraria (LM 69), Corso Gestione agronomica delle risorse idriche. https://www.unirc.it/documentazione/materiale_didattico/1462_2016_412_27105.pdf
- [3] Lawrence O. Hall W. Philip Kegelmeyer Nitesh V. Chawla, Kevin W. Bowyer *Smote: Synthetic minority over-sampling technique*, Journal of Artificial Intelligence Research 16 (2002)