Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica per il Management

Qualità dei dati e accuratezza delle previsioni in sistemi di agricoltura di precisione:

un'indagine sull'International Soil Moisture Network (ISMN)

Relatore:
Chiar.mo Prof.
MARCO DI FELICE

Presentata da:
MATTEO CERONI

Sessione II
Anno Accademico 2024/2025

A mio padre

Introduzione

Nell'epoca in cui viviamo, sensori *in situ* (misure acquisite direttamente sul terreno), satelliti e dispositivi connessi generano enormi volumi di dati. Queste collezioni sono eterogenee e possono racchiudere misure meteorologiche, idrologiche, dati satellitari e anche osservazioni di tipo agricolo (come quelle trattate in questa sede).

La raccolta dei dati, da sola, non basta. L'analisi dei flussi raccolti permette di convertire grandi quantità di osservazioni in informazioni concrete, come ottimizzare l'irrigazione, prevedere lo stress idrico o gestire il rischio agricolo.

L'analisi e l'estrazione della conoscenza dai dati (*data mining*) sono sempre più centrali per enti e imprese. Queste attività consentono di ottenere benefici operativi, efficienze economiche e una migliore gestione dei rischi.

Un utilizzo fondamentale lo troviamo nel *forecasting*: stimare l'evoluzione futura delle variabili misurate nel tempo. In ambito agro-ambientale questo riguarda l'umidità del suolo, le precipitazioni, le temperature e gli indicatori di stato idrico. Previsioni affidabili consentono di pianificare l'uso ottimale degli input (irrigazione e concimazione del terreno, utilizzo di antiparassitari, ecc.) a beneficio di una resa più elevata ed eco-sostenibile.

L'*ISMN* (International Soil Moisture Network) è un archivio globale ad accesso aperto che raccoglie e armonizza le misure dell'umidità e di altri parametri del suolo, ottenute direttamente sul terreno da reti eterogenee. L'archivio mette a disposizione dati e metadati in formato coerente, con controlli

di qualità che facilitano il confronto tra stazioni e network. I dati contenuti in questo archivio costituiscono la base delle analisi effettuate.

Nel presente lavoro è stata sviluppata una *pipeline* in Python che, a partire dai file `.stm` di ISMN, esegue ingestione e normalizzazione delle serie per variabili, verifica del campionamento e pulizia preliminare.

Vengono quindi individuati in modo sistematico i buchi temporali, quantificandone frequenza e durata ed esportando grafici di distribuzione cumulativa della lunghezza dei gap.

Successivamente si utilizzano modelli stagionali interpretabili (SARIMA) per il *forecasting*, mantenendo una pulizia del dato minima e coerente con la frequenza delle osservazioni. L'accuratezza delle previsioni (MAE, RMSE) viene poi letta in funzione dei parametri che descrivono i buchi temporali: percentuale di copertura, durata massima delle lacune e regolarità del campionamento. L'obiettivo della tesi è analizzare i dati del portale ISMN, trasformare le sue misure in previsioni operative e di quantificare quanto le lacune della serie ne condizionino l'affidabilità.

Di seguito si presenta la struttura dell'elaborato, con un quadro generale dei capitoli che la compongono.

Nel primo capitolo verranno introdotti concetti fondamentali ai fini di comprendere al meglio l'elaborato. Verrà spiegato cosa si intende per data mining e machine learning applicati alle serie temporali, verrano introdotti i principi del forecasting e si presenteranno modelli ARIMA e SARIMA come riferimento di lavoro.
Nel secondo si introduce il portale ISMN e viene illustrata una panoramica del progetto svolto.
Nel terzo si approfondiscono gli aspetti operativi del lavoro: dal pre-processing alla previsione delle serie temporali.
Infine, nel quarto capitolo saranno riportati e analizzati i risultati ottenuti.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Stato dell'Arte

1.1 Machine Learning

Il principio alla base dell'apprendimento automatico (*machine learning*) è che le macchine e gli algoritmi possono migliorare le proprie prestazioni grazie all'esperienza. Questo filo conduttore attraversa sia le riflessioni pionieristiche dell'informatica teorica sia le pratiche odierne che mettono in relazione dati, modelli e decisioni da intraprendere.

Tra i riferimenti storici più noti, Alan Turing prefigurò l'idea di macchine capaci di apprendere e propose la nozione di una *child machine* in grado di accrescere le proprie competenze tramite addestramento. [1] Questa intuizione trovò una sistematizzazione molti decenni più tardi nella definizione operativa proposta da Mitchell (1997):

> *"Un programma informatico si dice che apprenda da un'esperienza E, rispetto a una classe di compiti T e a una misura di prestazione P, se la sua prestazione nei compiti in T, misurata secondo P, migliora con l'esperienza E".* [2]

Questa definizione segna il passaggio dalle intuizioni filosofiche a criteri operativi: non basta l'idea che una macchina "impari", è necessario specificare **che cosa** si vuol far apprendere (T), **con quali esempi** (E) e **come misurare** il miglioramento (P).

Questi tre elementi guidano le scelte pratiche — dalla raccolta e qualità dei dati alla selezione dell'architettura del modello e alle procedure di validazione — e rendono misurabile e riproducibile il processo di apprendimento.

1.2 Cenni storici essenziali

Le radici dell'apprendimento automatico risalgono agli albori dell'informatica: già nelle prime riflessioni di Turing [1] si trova l'idea che macchine opportunamente progettate possano migliorare le proprie prestazioni osservando l'ambiente e confrontandosi con compiti concreti. Nei decenni successivi, la ricerca ha alternato momenti di forte entusiasmo a fasi di consolidamento, passando dagli esperimenti pionieristici sul gioco e dai primi modelli neurali lineari alle critiche sui limiti espressivi di alcuni approcci, che in determinati periodi ne hanno rallentato lo sviluppo.

Un nuovo impulso è arrivato con lo sviluppo di algoritmi più efficaci e con la formalizzazione di concetti teorici che hanno reso più chiari i rapporti tra complessità del modello, quantità di dati e capacità di generalizzazione. Sul piano pratico, la riscoperta della *retro-propagazione*[1] e l'affinamento delle tecniche di ottimizzazione hanno reso possibile addestrare reti multistrato [3], aprendo la strada a modelli sempre più profondi.

Negli ultimi quindici-vent'anni, la disponibilità di grandi dataset e di potenza di calcolo ha trasformato questi progressi in risultati applicativi notevoli. Le architetture profonde hanno ottenuto conquiste sostanziali in riconoscimento visivo e in molte altre aree, rafforzando l'archetipo del *deep learning*[2]. [4]

[1] La retro-propagazione è l'algoritmo tipicamente usato per addestrare reti neurali multistrato. Calcola quanto l'errore finale dipende da ciascun peso, propagandolo all'indietro attraverso gli strati per ottenere i gradienti necessari all'aggiornamento dei parametri. [3]

[2] Per "*deep learning*" si intende l'insieme di metodi basati su reti neurali profonde, capaci di apprendere rappresentazioni direttamente dai dati.

Questa evoluzione storica — dai primi concetti teorici agli odierni modelli su larga scala — fornisce il contesto entro cui si inserisce il presente lavoro, che applica metodi di apprendimento automatico a problemi di natura ambientale e agricola.

1.3 Apprendimento automatico e data mining

Per *apprendimento automatico* si intende l'insieme di metodi che costruiscono modelli a partire dai dati e che, attraverso l'esperienza, migliorano la loro capacità di risolvere un compito. I paradigmi principali usati come riferimento sono il *supervisionato* (classificazione, regressione), il *non supervisionato* (clustering, riduzione di dimensionalità), le strategie *semi-* o *auto-supervisionate* e l'utilizzo del *rinforzo*. [5]
In termini generali questi paradigmi si distinguono per il grado di supervisione richiesto. Il *supervisionato* si basa su esempi accompagnati dalla risposta desiderata. Il *non supervisionato* cerca strutture o rappresentazioni direttamente nei dati senza etichette. Gli approcci *semi-* o *auto-supervisionati* colmano il divario combinando piccole quantità di informazione etichettata con grandi insiemi di dati non etichettati o ricavando segnali di apprendimento direttamente dal dato stesso. Infine, con l'utilizzo del *rinforzo* il sistema ottimizza l'apprendimento sulla base di segnali di ricompensa.

Storicamente l'*ML* nasce dall'incrocio tra statistica, riconoscimento di pattern e intelligenza artificiale. Con la diffusione dei sistemi informativi e l'aumento dei volumi di dati sono successivamente nate pratiche organizzative (ad es. *warehouse* e i sistemi OLAP essendo degli archivi e strumenti pensati per conservare dati e facilitarne l'analisi) che hanno favorito l'affermazione del *data mining* come insieme di tecniche applicative. [5]

Gli esempi che seguono illustrano l'ampiezza delle applicazioni del ML in diversi domini. In ambito finanziario si usano modelli per il rilevamento di frodi e la previsione dei prezzi. Nei servizi digitali si progettano motori di raccomandazione e sistemi di analisi del rischio. Nel riconoscimento vocale

e nell'elaborazione del linguaggio i modelli consentono la trascrizione e la comprensione automatica. Sul fronte della regressione l'utilizzo più diffuso è la stima di grandezze continue (p.es. consumo energetico o temperatura interna di un edificio a partire da segnali ambientali e di calendario). In ambito agricolo, esempi pertinenti per questa tesi includono la previsione dell'umidità del suolo, la stima della resa e la definizione di schemi irrigui ottimali, basati su previsioni di evapotraspirazione[3] e precipitazione.

Nel seguito si adotterà dunque una nozione di machine learning (ML) centrata sulla capacità di costruire modelli dai dati con l'obiettivo della *generalizzazione*, ossia ottenere buone prestazioni su dati non osservati in addestramento. Questo richiede un bilanciamento tra complessità del modello, quantità/qualità dei dati e corrette pratiche di validazione.

1.3.1 Data mining e processo di KDD

In termini generali, il *data mining* può essere inteso come il nucleo operativo del più ampio processo di *Knowledge Discovery from Data* (KDD): non è soltanto un insieme di algoritmi, ma la fase in cui si cercano pattern e relazioni utili da dati precedentemente puliti e armonizzati. Il KDD parte dalla raccolta e pulizia delle informazioni (rimozione di errori, duplicati e incongruenze), prosegue con le fasi di integrazione e trasformazione finalizzate a creare strutture di dati omogenee, per arrivare al data mining vero e proprio e concludersi con la valutazione e la presentazione dei risultati.
L'obiettivo pratico è individuare modelli decisionali che siano validi, nuovi, utili e comprensibili. [5]

Sul piano operativo, uno dei riferimenti più diffusi per progettare questi percorsi è lo standard *Cross-Industry Standard Process for Data Mining* (CRISP-DM), che suggerisce di ancorare costantemente il lavoro agli obiettivi di business e di iterare tra comprensione del problema, esplorazione dei

[3]Per "evapotraspirazione" (ET) si intende la perdita complessiva di acqua dovuta all'evaporazione dal suolo e alla traspirazione delle piante. L'ET è un indicatore utile per verificare il fabbisogno irriguo e i bilanci idrici aziendali.

dati, preparazione, modellazione, valutazione e messa in produzione. Questo approccio è utile perché mette in evidenza come l'analisi non sia un'attività lineare, ma un ciclo che spesso richiede di tornare su passaggi precedenti quando emergono nuove esigenze o vincoli. [6]

Praticamente, il *data mining* comprende attività sia descrittive (trovare gruppi omogenei, regole di associazione, anomalie) sia predittive (costruire modelli per classificare o prevedere). In molte di queste attività il machine learning mette a disposizione diversi strumenti: modelli semplici e interpretabili come gli alberi decisionali, approcci robusti basati su *ensemble* che migliorano la stabilità predittiva, metodi a margine come le SVM per dati ad alta dimensione e reti neurali per relazioni non lineari e strutture complesse. Questi strumenti sono supportati da tecniche e architetture (algoritmi distribuiti, streaming, frequent-pattern mining, ecc.) che rendono possibile l'estrazione efficiente di pattern su dataset molto estesi. [5]

1.3.2 Preparazione dei dati (pre-processing)

Per *preparazione dei dati* si intendono le operazioni necessarie a trasformare il materiale grezzo in un dataset effettivamente utilizzabile per modellazione e analisi. In termini pratici, questo comprende l'attività di *pulizia* (rilevazione e correzione di errori, eliminazione di duplicati, gestione di outlier), l'*integrazione e armonizzazione* di fonti eterogenee, la *trasformazione e riduzione della dimensionalità* (standardizzazione, normalizzazione, selezione o estrazione di variabili) e infine la *gestione dei dati mancanti*. Nell'insieme questi passaggi costituiscono il nucleo operativo del KDD. [5]

Dal punto di vista operativo si lavora per lo piu con scelte concrete e ripetibili: si stabilisce innanzitutto la cadenza temporale di lavoro e si riallineano tutte le misure su quella scala per evitare disallineamenti o salti artefatti[4], uniformando formati, fusi orari e unità di misura. Si verificano plausibilità e limiti fisici per individuare errori strumentali o di inserimento.

[4]Per "artefatti" si intendono distorsioni non reali introdotte dal processo di acquisizione o elaborazione dei dati (es. rumore strumentale, disallineamenti temporali, salti fittizi).

Le operazioni di pulizia e trasformazione servono quindi a rendere i dati coerenti e confrontabili, in modo da garantire che la fase di modellazione non sia influenzata da artefatti banali. [5, 8]

La gestione dei dati mancanti si basa più su un principio guida piuttosto che su una regola fissa: non si tratta di "*inventare*" valori ma di scegliere procedure adeguate al tipo e alla durata delle lacune. Per brevi intervalli possono bastare interpolazioni o tecniche locali. Per le serie temporali con struttura si ricorre a modelli tempo-dipendenti (ad es. modelli di spazio-stato o di smoothing), che tengono conto della dipendenza temporale. In contesti statistici più rigorosi si valutano approcci di imputazione multipla per quantificare l'incertezza associata alle ricostruzioni. L'insieme di queste scelte va documentato e valutato rispetto agli obiettivi dell'analisi, perché decisioni diverse sulla preparazione possono influenzare sensibilmente i risultati successivi. [8, 9]

1.3.3 Forecasting su serie temporali

La modellazione predittiva cerca di cogliere relazioni che, apprese sul passato, permettano di prevedere il futuro nelle serie temporali. Questo impone di rispettare l'ordine cronologico: si fissa un orizzonte di previsione, si scelgono solo le informazioni disponibili al momento della previsione e si separano chiaramente i dati "*passati*" da quelli "*futuri*" usati per la valutazione.

Un approccio semplice e immediato è il *hold-out* temporale: si sceglie una data di taglio e tutto ciò che precede serve per addestrare, tutto ciò che segue per testare. Per ottenere valutazioni più robuste si usa lo *schema a rolling origin*, che ripete la procedura più volte facendo scorrere in avanti l'istante di previsione e misurando l'errore nel tempo. Questo consente di osservare la stabilità delle prestazioni man mano che arrivano nuovi dati. [7]

Per capire se un modello apprende davvero qualcosa di valevole, è utile confrontarlo con baseline elementari: la baseline *naïve* replica l'ultimo valore osservato, mentre la *seasonal naïve* ripete il valore relativo al ciclo stagionale precedente (ad es. lo stesso giorno della settimana o lo stesso mese dell'anno).

Se un modello non migliora in modo consistente rispetto a queste soglie la sua utilità predittiva è dubbia. [7]

1.4 Data analytics in agricoltura

Per *data analytics* in agricoltura si intende l'insieme delle pratiche e *pipeline*[5] che integrano flussi eterogenei — misure *in situ* (stazioni meteo e sensori di suolo), prodotti da telerilevamento (ad. es. Sentinel, SMAP), dati gestionali aziendali (FMIS) e fonti amministrative — trasformando i dati grezzi in indicatori e previsioni utili per l'azione sul campo. In termini operativi ciò significa che, dopo l'acquisizione, si eseguono controlli di qualità, si armonizzano le informazioni nello spazio e nel tempo, si costruiscono variabili derivate e si applicano modelli predittivi (sia di natura statistica sia basati su ML).
Il ciclo si chiude con la produzione di output utilizzabili per irrigazione, gestione nutritiva, difesa fitosanitaria, pianificazione delle lavorazioni e tracciabilità della filiera. [10, 11]

Le ricerche confermano che, quando questi strumenti sono integrati in processi organizzati e accompagnati da formazione degli operatori, si ottengono benefici misurabili in termini di efficienza d'uso delle risorse, ottimizzazione delle rese e maggiore sostenibilità. Architetture cyber–fisiche e soluzioni IoT forniscono misure ad alta frequenza, mentre modelli predittivi permettono di stimare grandezze rilevanti (p.es. umidità del suolo, rischio di stress idrico, produttività attesa) con impatti concreti su efficienza e resilienza dei sistemi agricoli. [20]

[5]Con "pipeline" si indica la catena di processi che trasforma dati grezzi in output utili: acquisizione e ingestione, controlli di qualità, integrazione e armonizzazione, trasformazioni e costruzione di variabili derivate, modellazione e validazione, e infine distribuzione dei risultati sotto forma di indicatori, mappe, report o servizi. L'uso di pipeline rende ripetibile e tracciabile l'intero flusso analitico.

1.4.1 Gestione dei valori mancanti

I dati mancanti sono una presenza ricorrente nelle reti di misura e nei sistemi agricoli; guasti agli strumenti, interruzioni di comunicazione, operazioni di manutenzione o condizioni ambientali avverse causano spesso lacune temporali. Reti come l'*International Soil Moisture Network* (ISMN) registrano esplicitamente queste assenze e conservano i flag di qualità anziché cancellare le osservazioni, in modo da preservare trasparenza e tracciabilità nelle analisi. [12, 13] Linee guida tecniche sottolineano l'importanza di standardizzare i controlli, annotare i periodi di indisponibilità e rendere tracciabili le correzioni applicate ai dati. [14]

La corretta gestione dei dati mancanti non è un dettaglio operativo, ma una condizione per rendere utili le previsioni: decisioni operative come timing e quantità di irrigazione, allerta per eventi estremi o stime di resa richiedono serie coerenti nel tempo. Per questo motivo si privilegiano stazioni con continuità temporale o si ricostruiscono le serie seguendo criteri documentati, così da migliorare l'affidabilità delle interpolazioni e la stabilità dei modelli previsionali, soprattutto in presenza di eventi meteorologici estremi. [15]

1.4.2 Casi applicativi di ARIMA e SARIMA in agricoltura

I modelli ARIMA e SARIMA sono spesso adottati in ambito agrario perché offrono un compromesso pratico tra semplicità, trasparenza e capacità predittiva quando la struttura temporale è relativamente regolare. Sono particolarmente utili come riferimento operativo e come baseline con cui confrontare modelli più complessi.

Per l'evapotraspirazione e il bilancio idrico, ARIMA è stato impiegato sia su serie *in situ* sia su prodotti satellitari (p.es. ET MODIS), fornendo una base interpretabile per lo scheduling irriguo e il bilancio aziendale. Frequentemente questi approcci vengono ibridati con metodi più flessibili per cogliere eventuali non linearità. [16, 21]

Nell'analisi dell'umidità del suolo superficiale i modelli stagionali SARIMA restituiscono previsioni a breve termine, utili per stimare il rischio di stress idrico e attivare logiche di allerta; rispetto a modelli più complessi, restano vantaggiosi quando serve leggibilità e facilità di validazione. [22]

Su serie a cadenza annuale (produzione, aree coltivate) ARIMA è spesso la scelta naturale se si dispone di una lunga serie di dati storici ma con poche variabili esplicative; in questi casi viene utilizzato per proiezioni di medio termine o come componente interpretabile in approcci ibridi che incorporano indicatori esterni. [17, 18]

In sintesi, ARIMA/SARIMA funzionano bene quando stagionalità e persistenza dominano il segnale: sono modelli di riferimento, facilmente mantenibili e interpretabili, e restano utili ogni volta che si richiedono previsioni spiegabili su serie reali soggette a buchi e cambiamenti strumentali.

Capitolo 2

Progettazione

Questo lavoro si propone di valutare fino a che punto le serie *in situ* raccolte dall'International Soil Moisture Network (ISMN) possano essere impiegate in applicazioni previsionali affidabili. Il progetto combina attività di ingegneria dei dati — ingestione, armonizzazione e controlli di qualità sui file .stm — con analisi quantitative della copertura e sperimentazioni di modellazione e validazione per quantificare l'impatto dei buchi temporali sulle prestazioni previsive.

Il presente capitolo è dedicato alla spiegazione e alla presentazione dei vari passaggi svolti nel progetto.
In particolare, vengono documentate le fase di estrazione e di valutazione della qualità[1] dei dati, con particolare attenzione ai buchi temporali. L'intento è duplice: quantificarne l'estensione e comprenderne l'impatto su eventuali modelli di previsione, fornendo indicazioni operative sulle soglie di copertura e sulla gestione delle lacune.

Per raggiungere tale scopo, si è definita una sequenza di fasi che include: il download e l'organizzazione dei dati, l'analisi dei dati ottenuti, l'identifi-

[1]Per "*qualità*" del dato si intende l'insieme di criteri che ne determinano l'idoneità all'analisi: (i) completezza/copertura delle osservazioni, (ii) accuratezza e plausibilità fisica, (iii) coerenza interna e regolarità del campionamento, (iv) tracciabilità di metadati e sensori.

cazione dei buchi temporali, l'analisi quantitativa della loro distribuzione e la stima sui dati futuri.

Per la redazione di questa tesi, sono stati impiegati strumenti di intelligenza artificiale generativa (ChatGPT-4, versione rilasciata a marzo 2024) a supporto nella creazione di contenuti e materiali di analisi. I dettagli sul suo utilizzo sono descritti in appendice.

2.1 Portale ISMN

L'*International Soil Moisture Network* (ISMN) è un archivio globale e ad accesso aperto che raccoglie misure *in situ* di umidità del suolo provenienti da reti eterogenee, offrendone una visione unificata e coerente. [19]
Il portale è stato realizzato e sviluppato con l'obiettivo di mettere a disposizione della comunità scientifica serie comparabili nello spazio e nel tempo. Inoltre mira ad per armonizzare unità e frequenze di campionamento, accompagnare ogni osservazione con metadati completi (stazione, sensore, profondità, coordinate, quota) e applicare controlli di qualità che generano *quality flags* utili a filtrare i dati in base all'affidabilità. Oltre all'umidità del suolo a diverse profondità sono disponibili variabili ancillari come temperatura dell'aria e del suolo e la precipitazione atmosferica, così da contestualizzare le dinamiche osservate.

L'accesso avviene tramite un'interfaccia cartografica che consente di selezionare reti e stazioni d'interesse, scegliere le variabili e l'intervallo temporale d'interesse, nonché scaricare pacchetti strutturati per rete/stazione/sensore. La filosofia è quella di una *data hosting facility*, che riduce i costi di integrazione per l'utente finale e favorisce confronti tra regioni e periodi diversi, mantenendo la tracciabilità verso le reti contributrici. In questo lavoro ISMN rappresenta la fonte primaria dei dataset su cui vengono condotte l'analisi di copertura, la quantificazione dei buchi temporali e le successive valutazioni previsionali.

2.2 Dati ISMN: acquisizione e struttura

Per garantire completezza e riproducibilità, è stata scaricata un'*istantanea* dell'intero dataset pubblico disponibile sul portale ISMN al momento dell'acquisizione, senza applicare filtri preventivi su reti, stazioni o variabili. L'operazione è stata eseguita tramite uno *script* di automazione che, una volta avviato dall'utente, gestisce in modo non interattivo il download dei pacchetti, ne verifica l'integrità e produce un registro riepilogativo dell'acquisizione.

I pacchetti compressi ottenuti contengono principalmente file `.stm`, che rappresentano le serie temporali per variabile/sensore e sono corredati da metadati. Tali file presentano una struttura a due livelli: un *header* con informazioni descrittive della stazione/sensore e un *corpo* con le osservazioni nel tempo.

Header dei file `.stm`.

- **Network** – Abbreviazione della rete (es. `OZNET`);
- **Stazione** – Nome della stazione (es. `Widgiewa`);
- **Latitudine** – In gradi decimali; valori negativi indicano l'emisfero sud;
- **Longitudine** – In gradi decimali; valori negativi indicano l'emisfero ovest;
- **Elevazione** – Altitudine in metri s.l.m.;
- **Profondità iniziale** – Profondità nel suolo a cui è riferita la misura (limite superiore);
- **Profondità finale** – Profondità nel suolo a cui è riferita la misura (limite inferiore).

Corpo del file (*time series*) Ogni `.stm` contiene le osservazioni come coppie *timestamp–valore*[2], affiancate da *quality flags* utili al filtraggio. I timestamp sono espressi in formato standard (coerente con UTC) e i valori sono in unità proprie della variabile. Le variabili misurate dipendono dalla strumentazione installata nella stazione e sono riconoscibili anche da identificatori presenti nel nome del file (ad es. `sm`, `p`, `ts`, `ta`); in particolare:

- `sm`: *Soil Moisture* (umidità del suolo)
- `p`: Precipitazione (tipicamente in mm).
- `ts`: *Soil Temperature* (temperatura del suolo).
- `ta`: *Air Temperature* (temperatura dell'aria).

I flag di qualità accompagnano ogni record e sono impiegati nel pre-processing per escludere osservazioni dubbie e per documentare la copertura effettiva.

2.3 Analisi dei dati

Una volta completata la fase di acquisizione dei file, si è provveduto ad effettuare un'analisi approfondita volta a valutare la qualità e continuità[3] delle serie temporali presenti nei file `.stm`. L'analisi si è concentrata su parametri come: regolarità temporale delle osservazioni, frequenza di campionamento dei dati (oraria) e livello di completezza delle serie disponibili presenti dei file.

Tali aspetti sono fondamentali per garantire l'affidabilità delle serie temporali che rappresentano un prerequisito per successive elaborazioni statistiche, tra cui ricostruzioni dei dati mancanti e modelli di previsione.

[2]Il termine "*timestamp*" indica un marcatore temporale, ovvero una rappresentazione numerica o testuale della data e dell'ora associata a un evento o a un'osservazione.

[3]Per "*continuità*" si intende la regolarità temporale delle osservazioni e la loro copertura sul periodo analizzato.

2.3.1 Studio della densità delle osservazioni

Motivazione e obiettivi dell'analisi

Ogni file `.stm` rappresenta una serie temporale e il numero di righe presenti è indicativo del grado di completezza informativa della stazione a cui appartiene. L'obiettivo di questa fase era di quantificare la distribuzione della quantità[4] di dati presenti nei file, sia a livello globale (sull'intero archivio o su subset selezionati), sia con riferimento alle specifiche variabili misurate. Questo ha permesso di ottenere un primo filtro utile per selezionare i file effettivamente idonei (p.es. copertura temporale elevata, frequenza regolare, ecc.) ad essere analizzati in dettaglio nelle fasi successive.

Valutazione complessiva della densità

Come primo studio sulla quantità di dati disponibili, è stata condotta un'analisi sul numero di righe presenti nei file `.stm`.
L'analisi può essere effettuata in vari modi: su un singolo network, su un gruppo selezionato oppure sull'intero archivio dei dati.
Il risultato viene rappresentato tramite un grafico cumulativo, che mostra la percentuale di file con un numero di righe inferiore a una certa soglia.

Questo tipo di visualizzazione consente di capire rapidamente quanti file sono completi, parziali o quasi vuoti, fornendo una base utile per decidere quali includere nelle analisi successive.

Analisi per variabile misurata

Per approfondire l'analisi della densità, i file `.stm` sono stati suddivisi a seconda della *variabile misurata*: umidità del suolo (`sm`), temperatura dell'aria (`ta`), temperatura superficiale (`ts`) e precipitazioni (`p`).
Questa comparazione ha reso possibile valutare la distribuzione e la comple-

[4]Per "*quantificare la distribuzione della quantità*" si intende valutare, a livello di file e di variabile, la presenza e la frequenza delle osservazioni mediante indicatori descrittivi (conteggi, mediana, percentili) e l'analisi della distribuzione dei gap temporali.

tezza dei dati, mettendo in evidenza la differenza nella copertura temporale (quantitativa) tra una variabile e l'altra.

L'analisi si è rivelata utile per individuare quali variabili detengono una maggiore continuità osservativa, rendendole più idonee per utilizzarle in elaborazioni successive, come la ricostruzione dei dati mancanti o l'applicazione di modelli previsionali.

Distribuzione dei valori osservati

In aggiunta alla valutazione della completezza dei file, è stata condotta anche un'analisi sulla distribuzione dei valori osservati per ciascuna variabile. Per ogni dataset i valori sono stati suddivisi in intervalli (*bin*) e ne è stata calcolata la frequenza relativa, così da ottenere una stima della distribuzione di probabilità[5]. Questa analisi consente di evidenziare la variabilità intrinseca dei dati e di individuare eventuali valori anomali o estremi.

2.3.2 Distribuzione percentuale delle osservazioni

Una seconda analisi è stata condotta su come si distribuisce la quantità delle osservazioni all'interno dell'archivio dei dati, considerando il numero totale di righe per ciascun dataset. Lo scopo era identificare eventuali squilibri nella quantità di dati disponibili, valutando se esistano gruppi di file scarsamente popolati o, al contrario, particolarmente ricchi di osservazioni.

L'analisi è stata eseguita sia sull'intero insieme dei file disponibili, sia separando i risultati in base alle diverse variabili visualizzate.

[5]Per "*distribuzione di probabilità*" si intende la funzione empirica che associa a ciascun intervallo (bin) la frequenza relativa delle osservazioni; può essere stimata tramite istogramma.

2.3.3 Presenza e distribuzione dei buchi temporali

Nell'ambito dell'analisi di serie temporali, la presenza di interruzioni nei dati rappresenta un aspetto critico, in quanto può influenzare significativamente la qualità delle elaborazioni e compromettere l'affidabilità di eventuali modelli previsionali. Per valutare l'estensione e la distribuzione di tali discontinuità, è stata condotta un'analisi mirata all'identificazione e quantificazione delle assenze di dati nei file `.stm`. Si è applicato un approccio multilivello che consente di caratterizzare in modo più dettagliato sia la frequenza che l'ampiezza delle lacune, tenendo conto del tipo di analisi da svolgere successivamente.

Frequenza dei buchi nei dataset

La prima analisi si è concentrata sulla quantificazione del numero di buchi presenti nei file. Utilizzando diverse scale temporali - *oraria*, *giornaliera* e *mensile* - è stato possibile ottenere una visione d'insieme della distribuzione delle interruzioni all'interno dell'intero dataset.

Durata delle interruzioni

L'indagine successiva si è focalizzata sulla lunghezza dei buchi, ossia sulla durata temporale di ciascuna assenza. Anche in questo caso sono state considerate le tre scale temporali, analizzandone la rispettiva copertura.

2.4 Analisi e previsione delle serie temporali

Nel contesto dell'analisi di dati ambientali strutturati in serie temporali, una delle principali questioni affrontate è stata la seguente: *quali operazioni analitiche sono effettivamente applicabili a queste serie, date le loro caratteristiche?* In particolare, ci si è interrogati sulla possibilità di utilizzare tali dati non solo a fini descrittivi, ma anche previsionali, valutando l'andamento futuro delle variabili considerate.

L'elevata presenza di buchi temporali osservata durante le fasi di analisi preliminare ha reso necessario approfondire in quale misura tali discontinuità potessero influenzare l'applicabilità di modelli di previsione. La qualità, la densità informativa e la regolarità della serie temporale costituiscono infatti condizioni essenziali per la costruzione di modelli affidabili.

Alla luce di queste considerazioni è emersa l'opportunità di valutare la possibilità di effettuare una previsione sui dati disponibili, con l'obiettivo di stimare l'evoluzione futura delle variabili ambientali analizzate. Tale approccio si è rivelato particolarmente utile sia per colmare le lacune presenti nelle serie temporali, sia per ottenere una raffigurazione più completa e continua del fenomeno osservato.

2.4.1 Modellazione della serie temporale: dal modello ARIMA a SARIMA

Il modello ARIMA: struttura e funzionamento

Il modello ARIMA (AutoRegressive Integrated Moving Average) rappresenta una delle tecniche statistiche più adottate per la modellazione e l'analisi futura di serie temporali. Il modello è adatto per serie temporali non stazionarie che non presentano stagionalità e viene adottato in ambiti dove si desidera stimare l'andamento futuro di un fenomeno sulla base dei dati osservati in passato.
ARIMA è composta da tre elementi fondamentali:

- **AR** (AutoRegressive): componente autoregressiva, che rappresenta la dipendenza del valore attuale dai valori passati della stessa serie;
- **I** (Integrated): differenziazione della serie per renderla stazionaria, ovvero per eliminare trend o variazioni di lungo periodo;
- **MA** (Moving Average): componente di media mobile, che tiene conto degli errori passati nella previsione.

Il modello è indicato con la notazione ARIMA (p, d, q), dove:

- p rappresenta l'ordine dell'autoregressione;
- d il grado di differenziazione necessario per ottenere la stazionarietà della serie;
- q l'ordine della media mobile.

La forma di un modello ARIMA può essere descritta come:

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

dove ϕ_i sono i coefficienti auto-regressivi, θ_j i coefficienti della media mobile, ed ε_t rappresenta il termine di errore (*white noise*[6]).

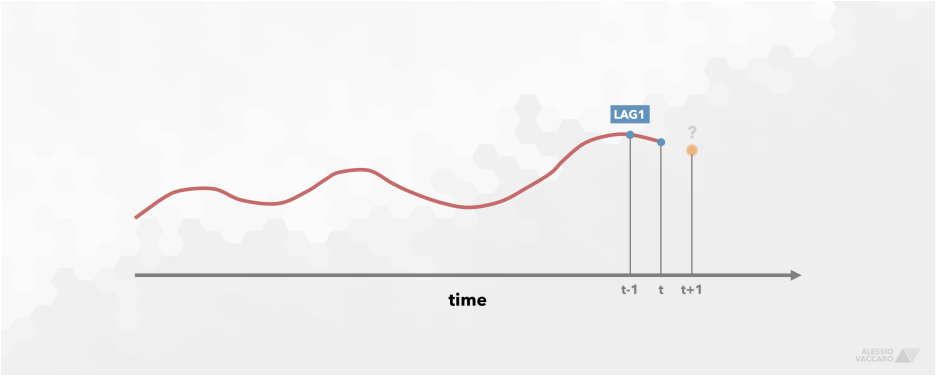

Figura 2.1: Rappresentazione grafica del concetto di previsione basata su lag temporali. Il modello utilizza i valori passati per stimare il valore futuro della serie.

Nel progetto è stato utilizzato ARIMA come punto di partenza per valutare la predicibilità delle serie temporali ambientali disponibili.

[6]Per "*white noise" si intende idealmente la parte imprevedibile di una serie* temporale: una sequenza di errori centrata (media nulla) e con varianza costante, che non mostra dipendenze nel tempo (cioè i termini non sono correlati tra loro).

Validazione del modello: approccio train-test split

Al fine di validare le prestazioni del modello, è stato adottato un approccio *train-test split*, che prevede la suddivisione della serie in due porzioni. La prima, denominata *training set*, è utilizzata per stimare i parametri del modello, ossia per addestrarlo. La seconda parte, detta *test set*, è invece impiegata per verificare la capacità predittiva del modello, confrontando le previsioni generate con i valori effettivamente osservati.

Tale tecnica consente di simulare una situazione realistica in cui si desidera prevedere il comportamento futuro di un fenomeno basandosi esclusivamente su dati storici, valutando al contempo l'accuratezza del modello.

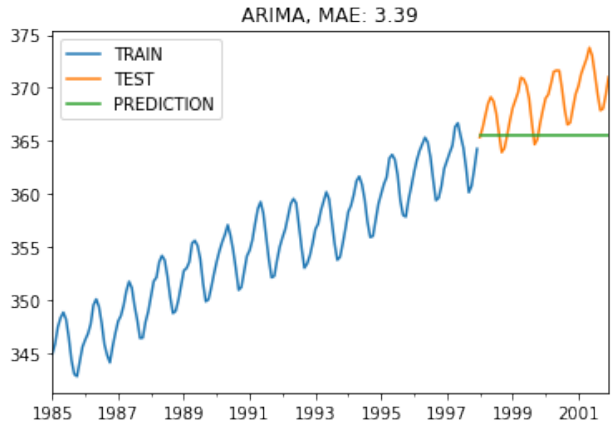

Figura 2.2: Esempio di suddivisione train-test su una serie temporale.

Dal modello ARIMA al modello SARIMA.

Sebbene il modello ARIMA offra una struttura flessibile per l'analisi e la previsione di serie temporali non stazionarie, esso non è in grado di gestire in modo esplicito la stagionalità. Questa limitazione diventa pesante in presenza di fenomeni ciclici regolari, come quelli tipicamente osservati nei dati ambientali, dove l'andamento delle variabili può variare secondo pattern orari, giornalieri, mensili o annuali. Diversamente, l'applicazione della media settimanale (Figura 4.12) genera una previsione sostanzialmente piatta, che

non riesce a seguire le variazioni osservate nei dati. Nonostante il dataset sia quasi privo di buchi, questa impostazione di imputazione compromette la capacità del modello di catturare i pattern sottostanti.
La notazione del modello diventa SARIMA$(p, d, q)(P, D, Q)_s$, dove:

- p, d, q sono gli ordini della componente non stagionale (autoregressiva, integrazione e media mobile);

- P, D, Q rappresentano gli ordini delle corrispondenti componenti stagionali;

- s indica il periodo della stagionalità che è quindi legato alla frequenza di campionamento della serie (ad es. $s = 24$ per stagionalità giornaliera in dati orari, oppure $s = 12$ per stagionalità annuale in dati mensili).

L'espressione generale di un modello SARIMA può essere scritta come:

$$\Phi_P(B^s)\phi_p(B)(1-B)^d(1-B^s)^D y_t = \Theta_Q(B^s)\theta_q(B)\varepsilon_t$$

dove: - B è l'operatore di ritardo (lag); - $\phi_p(B)$ e $\theta_q(B)$ rappresentano le componenti non stagionali AR e MA; - $\Phi_P(B^s)$ e $\Theta_Q(B^s)$ sono le componenti stagionali AR e MA; - $(1-B)^d$ e $(1-B^s)^D$ rappresentano le differenziazioni non stagionali e stagionali.

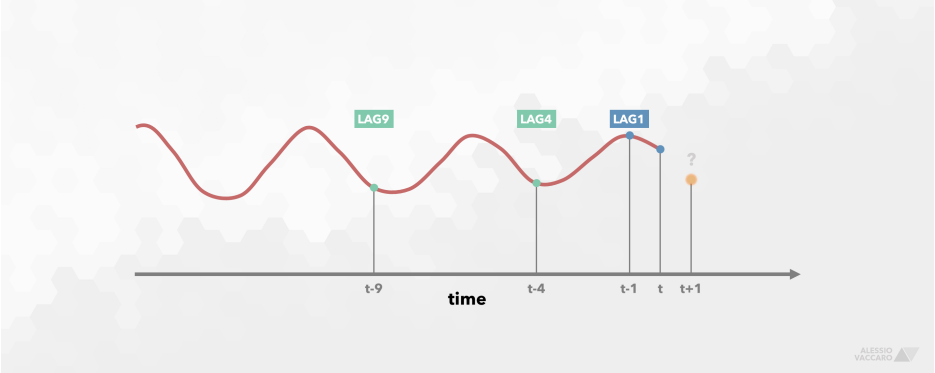

Figura 2.3: Rappresentazione del concetto di stagionalità nel modello SARIMA. A differenza della Figura 2.3 il valore futuro è stimato non solo in base ai dati recenti, ma anche a quelli ciclicamente ricorrenti nel tempo.

Capitolo 3

Implementazione

Di seguito viene illustrata l'implementazione dell'impostazione percentuale derivante dallo studio/analisi dei dati, presentato nel capitolo precedente.

Mostrando gli script sviluppati e i passaggi adottati, l'obiettivo è documentare in modo chiaro e sistematico il percorso seguito per trasformare la progettazione concettuale in codice eseguibile e funzionale.

3.1 Pre-processing

Dopo aver descritto il portale ISMN e la natura dei dati forniti (sezione 2.1), qui si illustra il processo di pre-processing necessario per rendere i dati pronti all'analisi.

Login È stato sviluppato uno script Python che simula il login al sito ISMN utilizzando la libreria `requests`[1].

```
session = requests.Session()
response = session.get(login_url)
soup = BeautifulSoup(response.text, 'html.parser')
csrf_token = soup.find('input', {'name':
```

[1]Per motivi di sicurezza il token/parametro CSRF è stato sostituito con il placeholder `token_name` nel codice riportato.

```
    'token_name'}).get('value')

payload = {
    'login': username,
    'password': password,
    'token_name': csrf_token,
}
session.post(login_url, data=payload)
```

Il codice crea una sessione HTTP[2] persistente tramite `requests.Session()` e invia una richiesta GET[3] alla pagina di login. All'interno del contenuto HTML[4] della risposta, viene cercato il token CSRF[5] necessario per l'autenticazione. Questo token viene estratto tramite la libreria `BeautifulSoup`. Infine, viene costruito un dizionario contenente le credenziali dell'utente e il token, che viene inviato tramite POST[6] per completare il login.

Acquisizione dei dati Dopo aver effettuato con successo l'accesso al portale è stato implementato uno script per il download automatico dei dataset.

```
def download(session):
    zip_filename = 'download_archive.zip'
    extract_dir = 'download_archive_extracted'
```

[2]HTTP (HyperText Transfer Protocol) è il protocollo usato per la comunicazione tra client e server nel web.

[3]GET è un metodo HTTP usato per richiedere dati da una risorsa specifica, come una pagina web.

[4](HyperText Markup Language) è il linguaggio di markup utilizzato per strutturare e visualizzare le pagine web; il contenuto HTML rappresenta quindi il codice sorgente restituito dal server in risposta alla richiesta.

[5]Il token CSRF (Cross-Site Request Forgery) è un valore univoco generato dal server per prevenire attacchi in cui richieste non autorizzate vengono inviate da un utente autenticato.

[6]POST è un metodo HTTP utilizzato per inviare dati al server, come nel caso del login.

```
headers = {'User-Agent': 'Mozilla/5.0',
           'Referer': dataviewer_url}

if not os.path.exists(extract_dir) or not
    os.listdir(extract_dir):
    download_response = session.get(download_url,
        headers=headers)
    if download_response.status_code == 200:
        with open(zip_filename, 'wb') as f:
            f.write(download_response.content)
        with zipfile.ZipFile(zip_filename, 'r') as zip_ref:
            zip_ref.extractall(extract_dir)
```

Il codice effettua il dowload dell'archivio ed estrae automaticamente il contenuto nella cartella `download_archive_extracted`, evitando le ridondanze.

Estrazione e struttura dati Una volta scaricato l'archivio, questo viene decompresso nella cartella `download_archive_extracted`, che mantiene la suddivisione per network e stazioni.
All'interno della directory principale `download_archive_extracted` sono presenti le sottocartelle relative ai diversi *Network*. Ciascuna di esse contiene a sua volta le cartelle delle singole *Stazioni*, al cui interno si trovano i file con estensione `.stm`.

I file `.stm`, come introdotto nella sezione 2.2, rappresentano le serie temporali acquisite e contengono osservazioni orarie corredate da eventuali flag e metadati:

```
# Header e metadati
2020/01/01 00:00  0.245  0  0
2020/01/01 01:00  0.248  0  0
2020/01/01 03:00  0.250  0  0
...
```

Filtraggio per variabili In alcune analisi, i file vengono classificati in base al loro nome, che include la variabile misurata (sezione 2.2).

`AACES_AACES_Farm01_ts_0.000000_0.050000_6507A_195001_202503.stm`

In questo caso, il suffisso `_ts_` indica che il file contiene dati di *soil temperature*.

Pacchetti adottati Per la realizzazione degli script e delle analisi si è fatto ampio uso di librerie Python. In particolare `pandas` per la gestione e manipolazione delle serie temporali, `statsmodels` e `pmdarima` per la stima dei modelli ARIMA/SARIMA, e `scikit-learn` per il calcolo delle metriche di valutazione.
Per completare l'ambiente di esecuzione si impiegano inoltre moduli di sistema: `os` per la gestione delle variabili d'ambiente e dei percorsi di file, e `psutil` per il monitoraggio delle risorse (memoria/CPU) durante l'esecuzione degli script.

```
# import principali utilizzati negli script
import os
import psutil
from statsmodels.tsa.statespace.sarimax import SARIMAX
import pandas as pd
from pmdarima import auto_arima
from sklearn.metrics import mean_squared_error,
    mean_absolute_error
import numpy as np
import requests
from bs4 import BeautifulSoup
```

3.2 Prime analisi implementate

Le metodologie presentate nella sezione 2.3 sono state implementate attraverso una serie di script in linguaggio Python, sviluppati con l'obiettivo di automatizzare le diverse fasi di analisi. Ciascuno script è stato progettato per svolgere un compito specifico, dall'elaborazione dei dati grezzi fino alla produzione di risultati sintetici sotto forma di tabelle e grafici. Nelle sezioni seguenti vengono descritte le principali procedure implementative.

3.2.1 Analisi della densità dei dati

Per quantificare la distribuzione del numero di osservazioni nei file `.stm` è stata implementata una procedura che, a partire dalle directory dei network, itera ricorsivamente[7] sulle stazioni e sui relativi file. In ciascun file vengono ignorate le righe di intestazione e i commenti (identificabili dal carattere `#`), in modo da considerare esclusivamente le osservazioni effettive.

Un estratto semplificato del codice è riportato di seguito:

```
if os.path.isdir(station_path):
    for file in os.listdir(station_path):
        if file.endswith('.stm'):
            file_path = os.path.join(station_path, file)
            with open(file_path, 'r') as f:
                lines = f.readlines()
                data_lines = [line for line in lines
                if not line.startswith('#') and
                    line.strip() != '']
                dataset_row_counts.append(len(data_lines))
```

[7] La "*ricorsione*" è una tecnica in cui una funzione richiama sé stessa per affrontare un problema suddividendolo in sotto-problemi analoghi. In questo caso viene utilizzata per esplorare automaticamente la struttura ad albero delle cartelle, scendendo progressivamente dai network alle stazioni e infine ai singoli file.

Il risultato è una collezione di valori, uno per ciascun dataset, che rappresenta il numero di osservazioni disponibili.

3.2.2 Analisi per variabili e distribuzione dei valori

Per ottenere i conteggi delle osservazioni suddivisi per variabile è stato utilizzato lo stesso processo di ricorsione introdotto nell'analisi della densità (sezione 3.2.1), classificandolo in base alla variabile di riferimento.

```
with open(file_path, 'r') as f:
    lines = f.readlines()
    all_counts.append(count)
    if "_p_" in file:
         p_counts.append(count)
    elif "_sm_" in file:
        sm_counts.append(count)
    elif "_ts_" in file:
        ts_counts.append(count)
    elif "_ta_" in file:
        ta_counts.append(count)
```

A questo punto, il file viene classificato in base alla variabile contenuta, riconosciuta dal suffisso nel nome del file, come spiegato nella sezione 3.1. In questo modo si ottengono quattro serie distinte, una per ciascuna variabile, oltre a una collezione generale che aggrega tutti i dataset.

Per lo studio della distribuzione statistica dei valori osservati, i valori numerici estratti dai file vengono convertiti in una struttura `pandas.Series` e suddivisi in intervalli `bin` di ampiezza prefissata. Per ciascun intervallo si calcola la frequenza relativa, ottenendo così una stima della distribuzione di probabilità.

```
data = pd.Series(valori)
counts, bins = np.histogram(data, bins=20, density=True)
```

Questo output viene poi rappresentato graficamente, permettendo di valutare la forma della distribuzione e di evidenziare eventuali anomalie o outliner.

3.3 Analisi dei buchi temporali

L'identificazione delle interruzioni dei dati (buchi temporali) è stata implementata tramite uno script dedicato, che elabora i file `.stm` estraendo i timestamp delle osservazioni e confrontandoli in sequenza. Per ogni coppia di righe consecutive viene calcolata la differenza temporale (`delta`), successivamente interpretata in base alla granularità di analisi richiesta.
La procedura è stata progettata per operare su tre scale temporali distinte:

- **oraria**, in cui il gap viene espresso in ore rispetto all'intervallo di campionamento previsto (1h);
- **giornaliera**, in cui la differenza viene convertita in giorni interi;
- **mensile**, in cui la distanza tra osservazioni è normalizzata in mesi approssimati a 30 giorni.

Il calcolo della differenza temporale e la logica per distinguere le diverse modalità è stato implementato come segue:

```
curr_time = datetime.strptime(parts[0] + " " + parts[1],
    "%Y/%m/%d %H:%M")
delta = curr_time - prev_time
if mode == "day":
    gap_giorni = delta.total_seconds() / (24*3600)
    if gap_giorni >= 1:
        ...
elif mode == "month":
    gap_mesi = delta.days // 30
    if gap_mesi >= 1:
        ...
```

```
else:  # modalità oraria
    gap_ore = (delta - timedelta(hours=1)).total_seconds()
        / 3600
    if gap_ore > 0:
        ...
```

3.3.1 Quantificazione dei buchi temporali

Oltre alla rilevazione puntuale delle interruzioni è stata introdotta un'analisi di sintesi con l'obiettivo di valutare l'incidenza complessiva dei buchi nei dataset. Questa fase non costituisce l'analisi principale ma permette di osservare come i gap si distribuiscano a livello aggregato, fornendo una misura della loro frequenza e diffusione.

A tal fine è stato sviluppato uno script che, a partire dai risultati dell'individuazione dei buchi (Sezione 3.3), considera ogni file come un dataset indipendente ed estrae il numero totale di interruzioni in esso contenute. I valori ottenuti vengono quindi raccolti per variabile (`sm`, `p`, `ts`, `ta`) e successivamente rappresentati mediante una curva cumulativa, che descrive la percentuale di stazioni caratterizzate da un numero massimo di buchi inferiore o uguale a una determinata soglia.

```
for ciascun file_gap in lista_csv_gap:
    num_buchi = len(file_gap)
    contatori["Tutti"].append(num_buchi)

    if "_sm_" in file_name:
        contatori["sm"].append(num_buchi)
    elif "_p_" in file_name:
        contatori["p"].append(num_buchi)
    ...
```

```
# ordinamento e costruzione del grafico cumulativo
for variabile, serie in contatori.items():
    serie.sort()
    cumulative = np.linspace(0, 100, len(serie))
    plt.step(serie, cumulative, where="post", label=variabile)
```

In questo modo si ottiene una visione d'insieme della portata delle interruzioni, evidenziando non solo il numero medio di gap ma anche le differenze strutturali tra le variabili considerate.

3.3.2 Lunghezza buchi temporali

Per valutare l'estensione delle lacune riscontrate nelle serie è stato condotto uno studio sulla durata dei gap. Questa analisi, complementare a quella sulla loro frequenza, non si concentra sul numero complessivo di interruzioni per dataset, ma sull'ampiezza dei singoli eventi mancanti. A tal fine è stato realizzato uno script che legge i file prodotti dall'identificazione dei buchi (Sezione 3.3) ed entra riga per riga nei CSV, estraendo la durata di ciascun intervallo vuoto secondo la scala temporale di riferimento (oraria, giornaliera o mensile).

In questo caso i buchi non vengono trattati come insiemi aggregati per stazione, ma come eventi elementari analizzati individualmente. I valori ottenuti vengono poi suddivisi per variabile (`sm`, `p`, `ts`, `ta`) e rappresentati tramite una distribuzione cumulativa delle lunghezze osservate, visualizzata con un grafico a gradini.

```
for ciascun file_gap in lista_csv_gap:
    for ciascuna riga in file_gap:
        durata = riga['Gap (h/gg/mm)']
        contatori["Tutti"].append(durata)
```

```
        if "_sm_" in file_name:
            contatori["sm"].append(durata)
        elif "_p_" in file_name:
            contatori["p"].append(durata)
        ...

# costruzione del grafico cumulativo delle lunghezze
for variabile, serie in contatori.items():
    serie.sort()
    cumulative = [i / len(serie) * 100 for i in range(1,
        len(serie)+1)]
    plt.step(serie, cumulative, where="post", label=variabile)
```

Tale approccio consente di evidenziare non solo la presenza o meno di interruzioni, ma anche la loro effettiva gravità: se si manifestano come fenomeni sporadici e di breve durata oppure come lacune estese in grado di compromettere porzioni consistenti della serie temporale.

3.4 Analisi e previsione delle serie temporali

In questa sezione si è passati dall'analisi descrittiva delle serie temporali alla loro modellazione previsionale. A tale proposito è stato adottato il modello SARIMA, applicato alle serie adeguatamente pre-elaborate per gestire la presenza di dati mancanti.
L'implementazione permette di applicare e mettere a confronto diverse strategie di elaborazione di tali assenze, valutando per ciascuna i risultati ottenuti attraverso indicatori quantitativi e rappresentazioni grafiche.

3.4.1 Preparazione dei dati

Per l'addestramento del modello è stato adottato un approccio di tipo *train–test split*, come già introdotto nella sezione 2.4.1. In particolare, le serie sono state suddivise in due insiemi distinti: uno utilizzato per stimare i parametri del modello (training set) e l'altro per la verifica delle prestazioni (test set). Sono state considerate due configurazioni di partizione, con l'80% oppure il 90% delle osservazioni destinate all'addestramento.

Poiché l'analisi è stata condotta unicamente sulle serie orarie, il numero di osservazioni da includere nell'operazione non è trascurabile e può, di conseguenza, diventare onerosa in termini di risorse computazionali. Per questo motivo è stato introdotto un meccanismo di controllo della **RAM** che adatta automaticamente il numero massimo di dati processabili in funzione della memoria disponibile (tramite una stima), evitando interruzioni dovute a limiti di calcolo.

Qui sotto riportato il codice che calcola la dimensione disponibile:

```
def calcola_max_dati_da_ram(ram_disponibile_gb,
    dati_per_14gb=52000):
return int((dati_per_14gb / 14) * ram_disponibile_gb)
```

3.4.2 Strategie di trattamento dei dati mancanti

Per verificare se le interruzioni temporali incidano sulla fase di test del modello, sono state adottate diverse strategie di imputazione dei valori mancanti. Di seguito vengono elencate, riportando come esempio di codice soltanto alcune di esse (quelle più rappresentative).

- **Rimozione dei dati mancanti**: le osservazioni assenti vengono semplicemente eliminate dalla serie.

- **Media ultimo giorno**: i valori mancanti vengono sostituiti con la media delle osservazioni disponibili nelle ultime 24 ore.

```
ultimo_giorno = df.dropna().last("1D")
media = ultimo_giorno["Value"].mean()
df["Value"] = df["Value"].fillna(media)
```

- **Media ultima settimana**: imputazione tramite la media delle osservazioni disponibili negli ultimi 7 giorni.

- **Media mobile settimanale**: media mobile calcolata su una finestra temporale di 7 giorni.

- **Media ultime 6 ore**: sostituzione con la media dei valori disponibili nelle sei ore precedenti.

```
ultime_6_ore = df.dropna().last("6H")
media = ultime_6_ore["Value"].mean()
df["Value"] = df["Value"].fillna(media)
```

- **Media mobile**: imputazione tramite media mobile su una finestra temporale definita (es. 12 ore).

```
    ultime_12_ore = df.dropna().last("12H")
    media = ultime_12_ore["Value"].mean()
    df["Value"] = df["Value"].fillna(media)
```

- **Interpolazione lineare**: stima i valori mancanti tracciando una retta tra i punti noti immediatamente precedenti e successivi, garantendo la continuità della serie. È efficace per buchi brevi e regolari, ma può risultare poco realistica in presenza di vuoti lunghi o variazioni irregolari.

```
df["Value"] = df["Value"].interpolate(
    method='time')
```

Il confronto tra queste soluzioni ha permesso di valutare come la scelta del metodo di imputazione influisca sulla stabilità del modello e sull'accuratezza delle previsioni.

3.5 Addestramento e valutazione del modello

La procedura di addestramento e valutazione del modello è stata implementata come una funzione unica che, a partire da una serie oraria preelaborata (Sez. 3.4), esegue le fasi di selezione del modello, stima, previsione e misurazione delle prestazioni.

3.5.1 Flusso operativo

La funzione opera secondo i seguenti passi:

1. **Selezione della stagionalità**: vengono considerate due ipotesi di periodicità oraria ($m = 12$ e $m = 24$), coerenti con cicli infragiornalieri[8]. La stagionalità m coincide con il periodo s introdotto nella notazione teorica del modello SARIMA (Sez. 2.4.1). La scelta tra le due configurazioni avviene confrontando l'AIC[9] dei modelli stimati e selezionando la più parsimoniosa e con miglior adattamento ai dati.

2. **Controllo della dimensione del campione**: per prevenire errori di memoria, la lunghezza massima della serie utilizzata è adattata in base alla RAM disponibile. Qualora l'adattamento non fosse sufficiente si riduce progressivamente il numero di osservazioni.

3. **Suddivisione in train e test**: la serie è separata in due blocchi contigui secondo una quota prefissata (80% o 90% per il training).

[8] Il termine "*infragiornaliero*" indica fenomeni che si ripetono all'interno della giornata, come i cicli semigiornalieri (12 ore) e giornalieri (24 ore).

[9] L'*Akaike Information Criterion* (AIC) è un indicatore che bilancia la bontà di adattamento del modello con la sua complessità, penalizzando quelli con troppi parametri. Valori più bassi corrispondono a modelli più parsimoniosi ed efficaci.

4. **Selezione automatica dei parametri**: per ciascun valore di m viene avviata una procedura di ricerca automatica (funzione `auto_arima`), che esplora combinazioni di ordini non stagionali (p, d, q) e stagionali (P, D, Q) entro limiti prefissati ($0 \leq p, q, P, Q \leq 2$). L'algoritmo incrementa i valori dei parametri in modo stepwise[10], stimando ogni volta il modello e calcolandone l'AIC. La configurazione con AIC minimo è scelta come migliore, in quanto rappresenta un buon compromesso tra complessità e capacità di adattamento.

5. **Stima e previsione**: il modello selezionato è adattato sui dati di training e utilizzato per prevedere l'orizzonte corrispondente al blocco di test.

6. **Valutazione**: la qualità delle previsioni è misurata tramite *RMSE*, *MAE* e *MSE*[11] sul test set.

3.5.2 Dettagli di implementazione

Selezione del modello Per ogni ipotesi di stagionalità m la funzione esplora uno spazio limitato ma rappresentativo di configurazioni $(p, d, q)(P, D, Q)_m$, stimando l'AIC per ciascuna combinazione. La ricerca è guidata dalla funzione `auto_arima`, che utilizza un approccio euristico (stepwise) per ridurre il numero di modelli stimati, garantendo comunque una buona copertura dello spazio dei parametri. La scelta finale ricade sul modello con AIC più basso, così da privilegiare parsimonia e capacità di previsione.

[10] Il termine "*stepwise*" indica una ricerca iterativa che aggiunge o rimuove parametri a piccoli passi, valutando di volta in volta la bontà del modello e arrestando l'esplorazione quando ulteriori complessità non migliorano significativamente il criterio scelto (in questo caso l'AIC).

[11] L'*RMSE* (Root Mean Squared Error) e l'*MSE* (Mean Squared Error) misurano la deviazione quadratica media tra osservati e previsti, mentre l'*MAE* (Mean Absolute Error) si basa sulla deviazione assoluta media. In generale, valori più bassi indicano previsioni più accurate.

Gestione delle risorse Poiché si lavora su serie *orarie*, la numerosità può essere elevata. La funzione integra un controllo preventivo della memoria disponibile e, al bisogno, riduce in modo controllato il numero di osservazioni elaborate, mantenendo comunque una finestra sufficiente per la stima e la validazione.

Schema di validazione La suddivisione *train–test* è temporale (blocchi contigui), così da preservare la struttura di dipendenza seriale. Sono previste due impostazioni (80/20 e 90/10, con uno scarto del 1%) per verificare la sensibilità dei risultati alla dimensione del training set.

Metriche Le prestazioni[12] sono riportate tramite:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_t (y_t - \hat{y}_t)^2} \quad \mathrm{MAE} = \frac{1}{n}\sum_t |y_t - \hat{y}_t| \quad \mathrm{MSE} = \frac{1}{n}\sum_t (y_t - \hat{y}_t)^2$$

Queste misure consentono di confrontare in modo omogeneo le diverse strategie di imputazione e le ipotesi di stagionalità.

[12] In queste formule y_t indica il valore osservato, $\hat{y}_t$ la previsione e n il numero di osservazioni di test.

Capitolo 4

Risultati sperimentali

Questo capitolo riporta i risultati ottenuti a seguito dall'analisi e implementazioni descritte nel Capitolo 3 e fornisce una panoramica chiara sull'effettiva qualità dei dati, sulla presenza di discontinuità temporali e sulle prestazioni dei modelli di previsione applicati.

4.1 Quantificazione delle osservazioni nei dataset

La prima analisi ha riguardato la distribuzione del numero di osservazioni nei file `.stm`. Le variabili ricavate sono: precipitazione, umidità del suolo, temperatura superficiale e temperatura dell'aria.
Il risultato è riportato in Figura 4.2, dove sono mostrate le curve cumulative relative a ciascuna variabile insieme alla distribuzione totale.

Dalla Figura 4.2 emerge come l'umidità del suolo (sm) e la temperatura superficiale (ts) siano le variabili più rappresentate in termini di numero di file, mentre la precipitazione (p) e la temperatura dell'aria (ta) sono presenti in quantità più limitata. Tuttavia i file relativi alla temperatura dell'aria tendono a contenere un numero di osservazioni mediamente più elevato rispetto agli altri, come confermato dalle statistiche descrittive (Tab. 4.1).

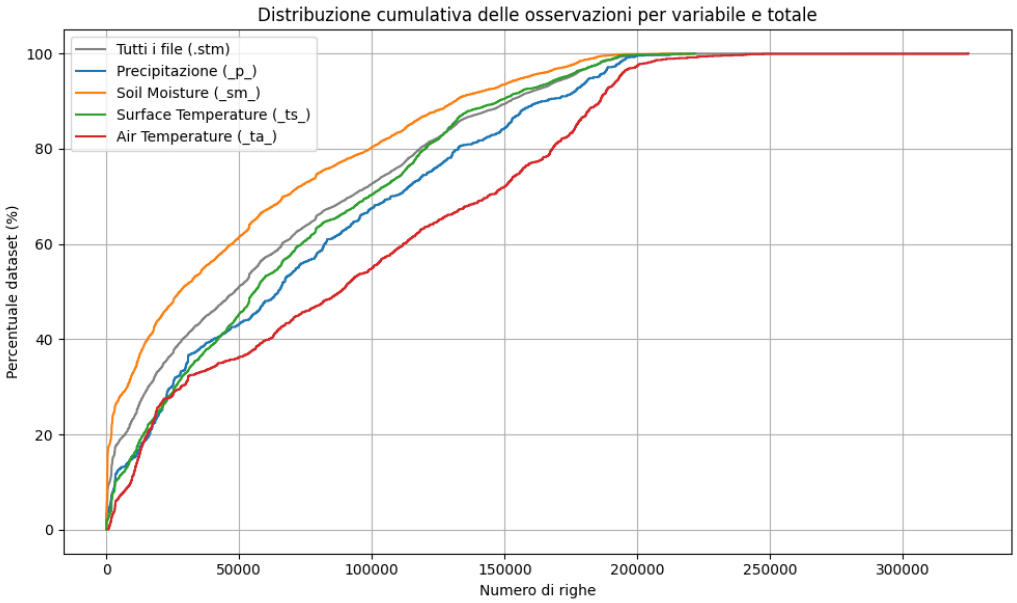

Figura 4.1: Distribuzione cumulativa delle osservazioni per variabile e totale.

Nel grafico, una curva più vicina all'alto indica la presenza di molti dataset con poche osservazioni, mentre una curva più bassa riflette dataset mediamente più estesi e completi.

Variabile	Media	Mediana	Min	Max
Precipitazione (p)	73 836	64 770	2	216 290
Soil Moisture (sm)	48 683	27 702	2	220 043
Surface Temperature (ts)	67 691	55 996	2	221 618
Air Temperature (ta)	91 098	88 091	663	324 658

Tabella 4.1: Statistiche descrittive del numero di osservazioni per variabile.

Nel complesso, la distribuzione cumulativa evidenzia un'elevata eterogeneità: accanto a dataset molto estesi (oltre 300 000 osservazioni) si osservano anche numerosi file con poche centinaia di righe. Questa disomogeneità costituisce un primo indicatore della qualità non uniforme dei dati, che verrà

ulteriormente approfondita con l'analisi dei buchi temporali nella sezione successiva.

Analisi per intervalli (bin)

Oltre al conteggio complessivo delle osservazioni nei file `.stm` è stata condotta un'analisi per intervalli (*bin*) con ampiezza pari a 20 000 osservazioni. In questo modo si è ottenuta una distribuzione statistica più dettagliata.
Il grafico in Figura 4.2 mostra la percentuale di dataset che ricadono in ciascun intervallo, distinguendo le diverse variabili considerate.

Dall'istogramma emerge come la maggior parte dei file sia concentrata nei primi bin (0–20 000 osservazioni); in particolare per l'umidità del suolo (sm), che presenta una forte prevalenza di dataset brevi. Al contrario, la temperatura dell'aria (ta) mostra una distribuzione più ampia, con una quota significativa di file che superano le 100 000 osservazioni e casi che arrivano oltre le 300 000 righe.

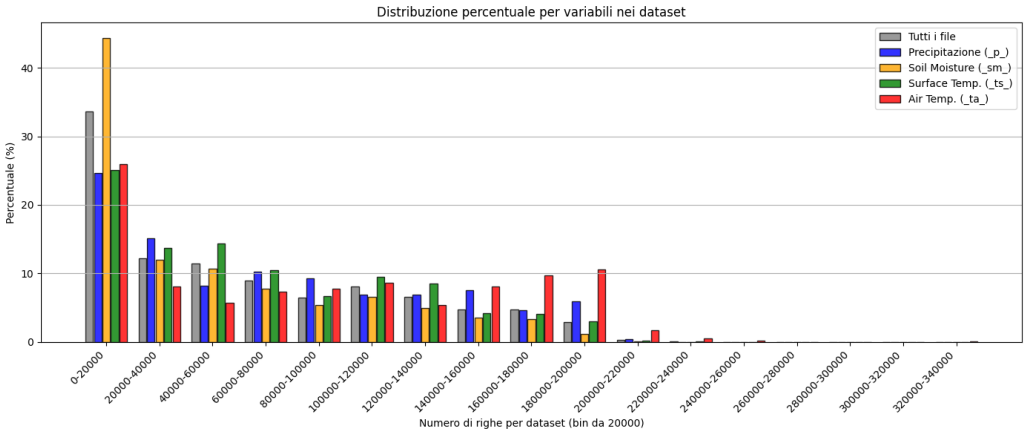

Figura 4.2: Distribuzione percentuale delle osservazioni per intervalli di ampiezza 20 000, suddivisa per variabile.

Nel complesso, la distribuzione risulta fortemente asimmetrica: pochi dataset molto lunghi convivono con un numero elevatissimo di file ridotti. Questo conferma l'eterogeneità già osservata e suggerisce che non tutte le variabili o le reti siano ugualmente adatte a supportare analisi di lungo periodo o modelli previsionali.

4.2 Distribuzioni cumulative

Un aspetto rilevante riguarda la presenza di discontinuità nelle serie temporali, che costituiscono una caratteristica ricorrente del dataset ISMN. Per valutarne l'estensione, sono state prodotte distribuzioni cumulative del numero di buchi rilevati nei file `.stm`, considerando la loro presenza (quantità) e dimensione (lunghezza).

4.2.1 Distribuzione comulativa sulla quantità dei buchi temporali

I grafici in Figura 4.3–4.4–4.5 mostrano la distribuzione cumulativa del numero di buchi osservati per ciascuna variabile alle tre scale temporali considerate. Poiché rappresentano la totalità dei dataset, le curve cumulano fino al 100%. L'andamento della crescita riflette la distribuzione dei buchi: una salita rapida indica interruzioni contenute nella maggior parte dei file, mentre una crescita più lenta segnala la presenza di dataset con molte lacune. In conseguenza, un andamento più graduale segnala la presenza di serie con numerosi buchi e una distribuzione più eterogenea.

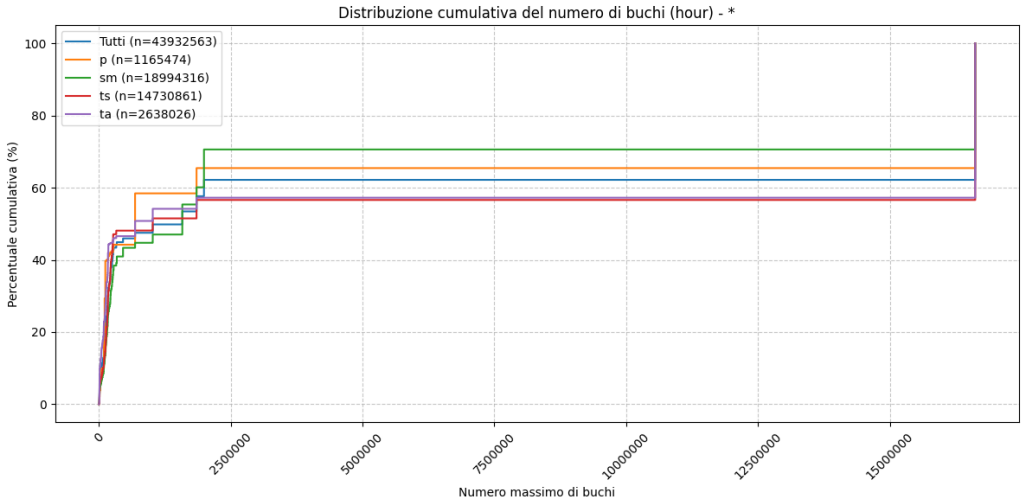

Figura 4.3: Distribuzione cumulativa del numero di buchi su scala oraria.

Su scala oraria quasi tutti i file mostrano discontinuità. Per una parte significativa i buchi restano entro soglie contenute, mentre esistono anche casi estremi con milioni di interruzioni;

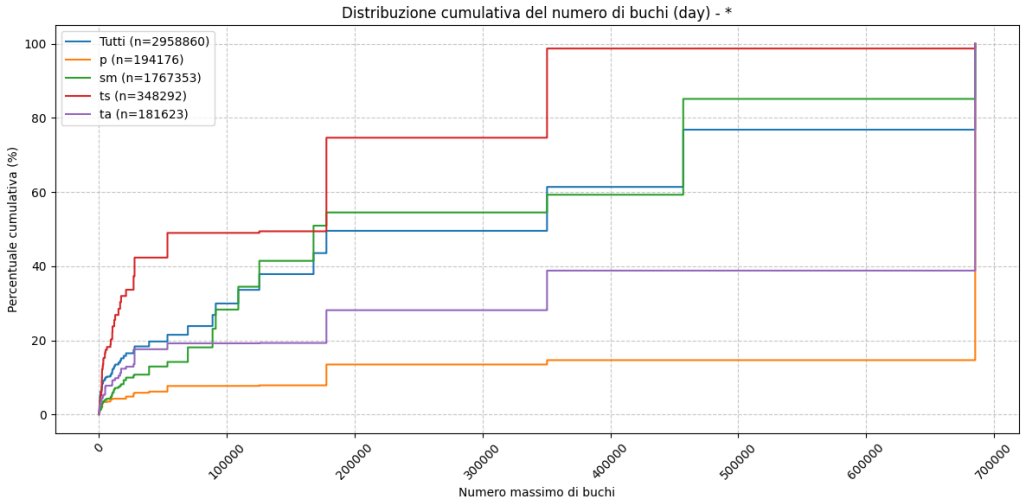

Figura 4.4: Distribuzione cumulativa del numero di buchi su scala giornaliera.

Su scala giornaliera molte serie hanno centinaia o migliaia di giorni mancanti, con differenze marcate tra variabili (ad es. la *sm* risulta più spezzata).

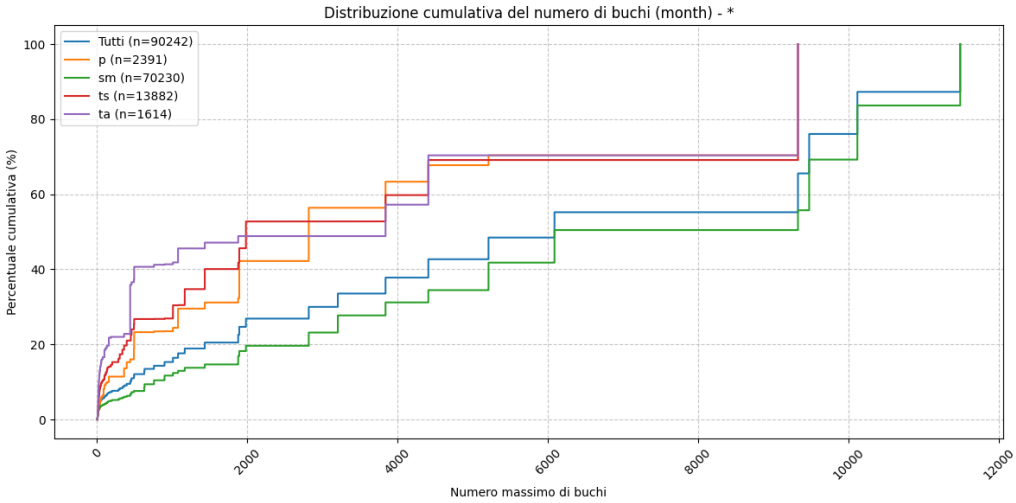

Figura 4.5: Distribuzione cumulativa del numero di buchi su scala mensile.

Su scala mensile si osservano lacune distribuite anche su lunghi periodi, confermando l'eterogeneità della qualità dei dati.

4.2.2 Distribuzione cumulativa sulla lunghezza dei buchi temporali

Le curve cumulative riportate nei grafici in Figura 4.6–4.7–4.8 mostrano la percentuale di buchi che non supera una determinata soglia temporale. Nel complesso, emerge che la grande maggioranza delle interruzioni ha una **lunghezza contenuta**, limitata a poche unità temporali: la curva cumulativa cresce molto rapidamente nelle prime posizioni, indicando che oltre la metà dei buchi si risolve entro intervalli molto brevi.

Le differenze tra variabili restano significative. In particolare, i dataset relativi all'umidità del suolo (sm) e alla temperatura superficiale (ts) presentano con maggiore frequenza buchi estesi, che si protraggono su più giorni o addirittura mesi. Al contrario, le serie di precipitazione (p) e soprattutto di temperatura dell'aria (ta) mostrano discontinuità generalmente più brevi, con curve cumulative che tendono a saturarsi più velocemente.

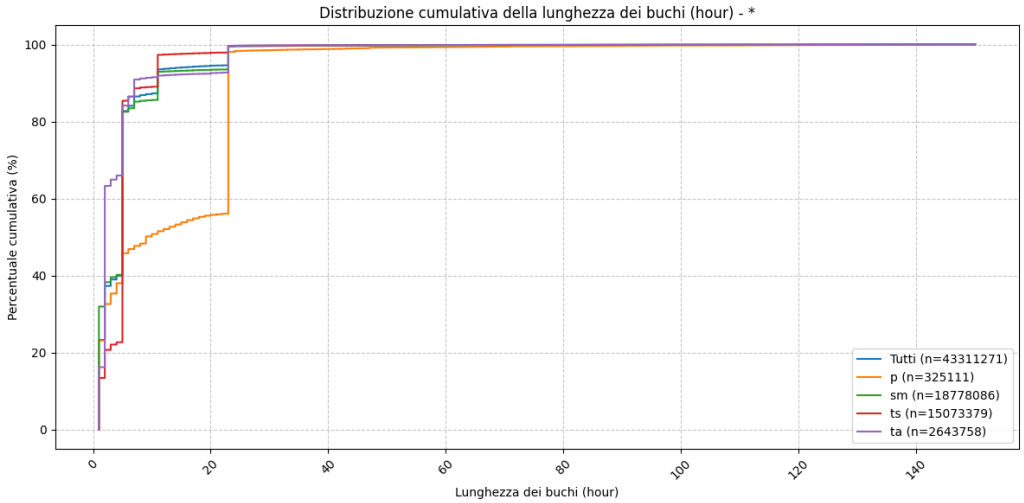

Figura 4.6: Distribuzione cumulativa della lunghezza dei buchi su scala oraria.

Nel complesso, emerge che la grande maggioranza delle interruzioni ha una **lunghezza contenuta**, limitata a poche unità temporali: la curva cumulativa cresce molto rapidamente nelle prime posizioni, indicando che oltre la metà dei buchi si risolve entro intervalli molto brevi.

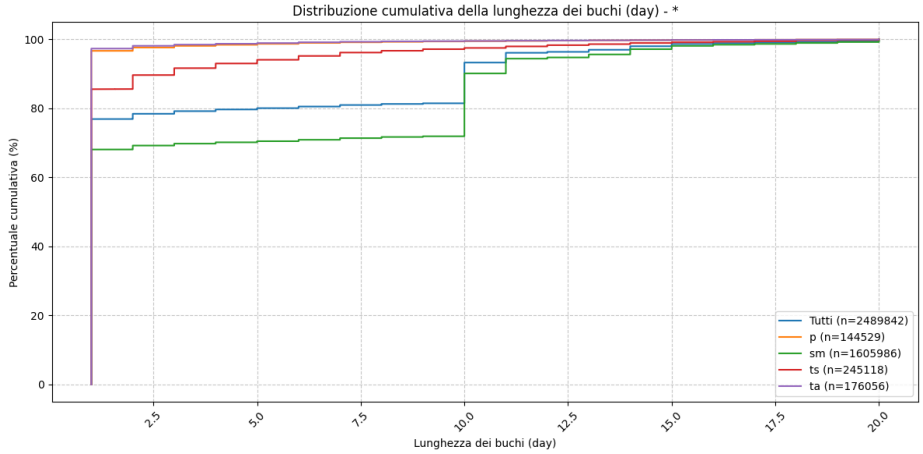

Figura 4.7: Distribuzione cumulativa della lunghezza dei buchi su scala giornaliera

Le differenze tra variabili restano significative. In particolare, i dataset relativi all'umidità del suolo (sm) e alla temperatura superficiale (ts) presentano con maggiore frequenza buchi estesi, che si protraggono su più giorni o addirittura mesi. Al contrario, le serie di precipitazione (p) e soprattutto di temperatura dell'aria (ta) mostrano discontinuità generalmente più brevi, con curve cumulative che tendono a saturarsi più velocemente.

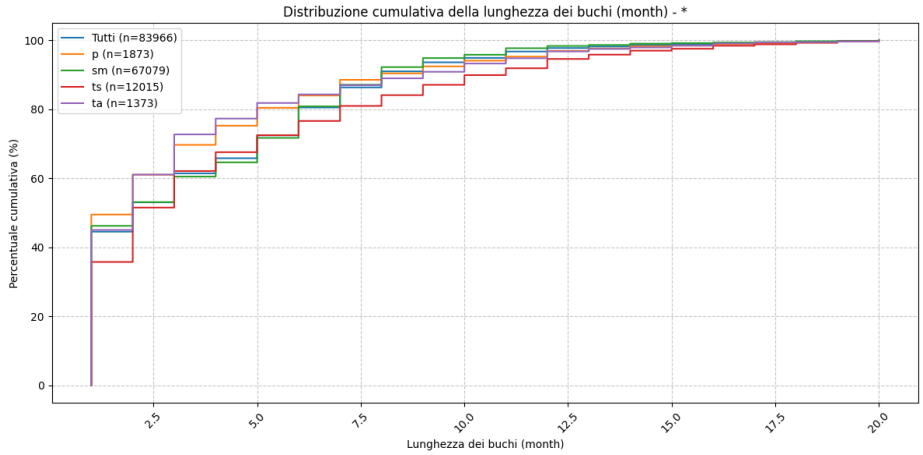

Figura 4.8: Distribuzione cumulativa della lunghezza dei buchi su scala mensile.

Il confronto tra le tre scale temporali conferma questa tendenza:

- su scala **oraria**, la quasi totalità dei buchi non supera poche decine di ore;
- su scala **giornaliera**, la distribuzione si sposta leggermente verso lacune più lunghe, ma rimane concentrata nei primi valori;
- su scala **mensile**, emergono con maggiore chiarezza quei casi sporadici di interruzioni estese, che pur rappresentando una frazione ridotta incidono in modo rilevante sulla continuità delle serie.

Questi risultati suggeriscono che, pur essendo numerose, le interruzioni sono spesso di breve durata; tuttavia, in alcune variabili e contesti specifici occorre considerare la presenza di lacune estese, che possono compromettere analisi a lungo termine o applicazioni previsionali.

4.3 Modellazione mediante SARIMA

Dopo aver analizzato la presenza e la distribuzione dei buchi temporali, in questa sezione ci concentriamo sulla fase di modellazione delle serie. L'obiettivo è verificare l'applicabilità di modelli autoregressivi stagionali (SARIMA) ai dati disponibili, valutando in particolare la capacità di descrivere e prevedere l'andamento delle osservazioni nonostante la presenza di lacune nei dataset.

Inoltre, la selezione dei *network* oggetto di analisi si basa sulla disponibilità di dati *in situ* con buona copertura spaziale e temporale, utili a testare la robustezza dei modelli. Inoltre presentano lacune e problemi di qualità tipici dei contesti operativi, quindi sono casi significativi per valutare strategie di imputazione e la performance di SARIMA come *baseline*.

4.3.1 Scelte sui dati e sull'ambito di analisi

Variabile considerata

Per l'applicazione dei modelli SARIMA si è scelto di concentrare l'attenzione sulla variabile *soil moisture* (sm) con frequenza oraria. Questa decisione deriva da considerazioni di carattere sia tecnico sia metodologico.
Dal punto di vista operativo, infatti, l'archivio ISMN nasce proprio per la misura dell'umidità del suolo, che risulta pertanto la variabile più diffusa e regolare tra quelle disponibili; mentre la temperatura dell'aria, temperatura superficiale e precipitazione compaiono solo in sottoinsiemi più limitati delle serie. Inoltre le analisi svolte nelle sezioni precedenti hanno evidenziato come le misure di *sm* siano non soltanto più numerose, ma anche quelle su cui è stato possibile quantificare in maniera più sistematica la presenza di interruzioni.

Un aspetto da sottolineare riguarda proprio l'incidenza dei buchi temporali: pur essendo la variabile più popolata, sm presenta infatti una frequenza non trascurabile di lacune, che possono influenzare la stima dei parametri e la qualità delle previsioni. La scelta di concentrarsi su questa variabile permette perciò di affrontare in modo esplicito il problema dei dati mancanti, valutando l'efficacia delle diverse strategie di trattamento prima della modellazione.

Scala temporale

Un ulteriore aspetto riguarda la scala temporale di riferimento.
È stata scelta quella oraria, poiché le serie in questa forma si sono dimostrate più complete e consistenti rispetto alle aggregazioni giornaliere o mensili.
Inoltre la regolarità oraria consente di sfruttare appieno le potenzialità del modello SARIMA nel cogliere pattern stagionali infra-giornalieri, in particolare i cicli diurni (periodo $m = 24$) e semidiurni ($m = 12$), mentre le

aggregazioni a tempi più lunghi ridurrebbero sensibilmente la capacità di identificare e stimare tali componenti.

Ambito di analisi

Infine, l'analisi è stata condotta a livello di singolo network e non sull'insieme complessivo dei dati. Una modellazione basata su un'aggregazione indiscriminata rischierebbe di nascondere le peculiarità proprie di ciascuna rete, legate alle condizioni climatiche, agli strumenti impiegati e alle procedure di misura adottate. Per questa ragione la valutazione è stata impostata considerando separatamente i diversi network, così da preservarne le specificità.

Inoltre, il confronto tra reti è stato reso più significativo selezionando sottogruppi di dimensioni comparabili, ossia insiemi di stazioni con un grado analogo di popolamento delle serie e una simile copertura temporale.
In questo modo è possibile effettuare analisi parallele e confronti diretti tra network differenti, senza che squilibri nella numerosità delle osservazioni compromettano la coerenza dei risultati.

4.3.2 Casi studio con copertura elevata (reti quasi complete)

L'analisi è stata condotta sull'insieme dei network disponibili, valutando in ciascun caso le prestazioni del modello SARIMA al variare delle strategie di imputazione e dei parametri stagionali. Al fine di illustrare in maniera più chiara i comportamenti riscontrati, vengono qui riportati alcuni esempi rappresentativi di reti caratterizzate da una copertura pressoché completa, nei quali la presenza di dati mancanti non costituisce un ostacolo rilevante. In particolare, sono stati selezionati i network `IIT_KANPUR` (**0.06**% di dati mancanti su **44.044** dati disponibili) e `VDS` (**0.01**% di dati mancanti su **346.236** dati disponibili), che ben evidenziano le differenze derivanti dalle scelte di imputazione e dal valore attribuito al periodo stagionale m.

IIT_KANPUR

La Figura 4.9 mostra i risultati ottenuti utilizzando l'interpolazione lineare e una stagionalità pari a $m = 12$. In questo caso il modello è in grado di riprodurre in maniera coerente l'andamento della serie, con errori contenuti e una buona aderenza tra valori osservati e previsti.

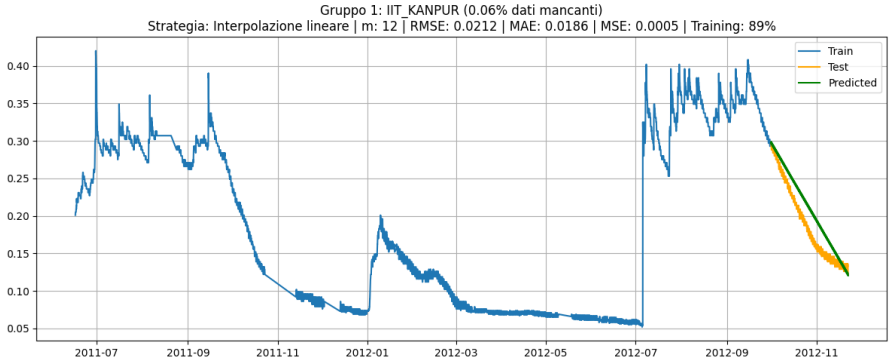

Figura 4.9: Risultati per `IIT_KANPUR` con interpolazione lineare e stagionalità $m = 12$. La previsione risulta plausibile e gli errori sono bassi.

Al contrario, la Figura 4.10 evidenzia gli effetti negativi della scelta di un valore stagionale non adeguato ($m = 24$) o di strategie di imputazione eccessivamente semplificate (ad es., media mobile). In questo caso la previsione risulta appiattita o divergente, con errori decisamente più elevati.

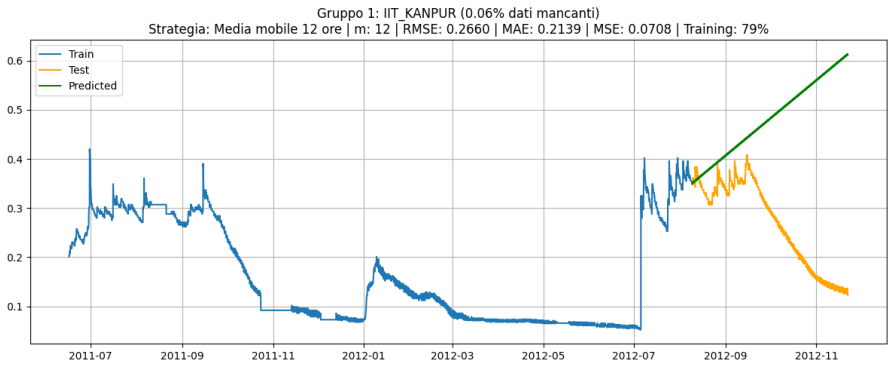

Figura 4.10: Risultati per `IIT_KANPUR` con imputazione tramite media mobile: la previsione diverge rispetto ai valori osservati.

Questi confronti dimostrano che, anche in presenza di una copertura quasi completa, la qualità della previsione dipende in maniera sostanziale sia dalla scelta del parametro stagionale m, sia dalla strategia di imputazione adottata.

VDS

Il secondo caso, riportato in Figura 4.11, riguarda il network `VDS`, anch'esso caratterizzato da un numero trascurabile di valori mancanti. L'utilizzo della media mobile a 6 ore consente di ottenere risultati coerenti con la dinamica osservata, con valori previsti plausibili e una precisione complessivamente accettabile.

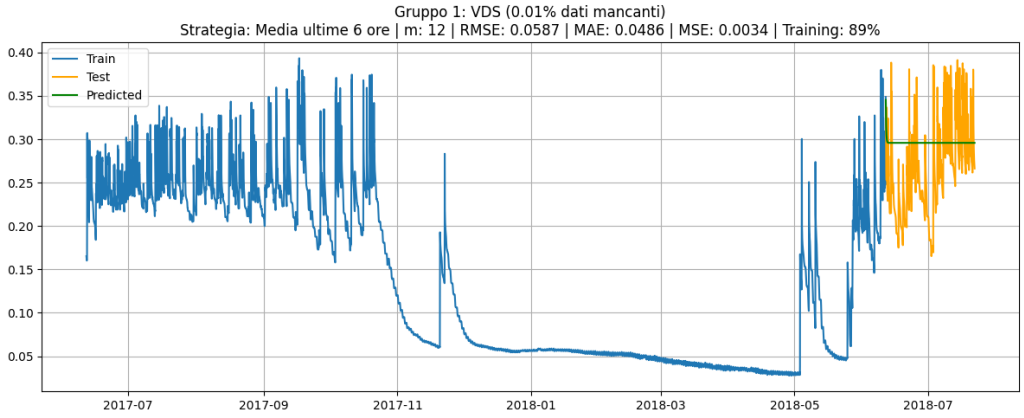

Figura 4.11: Risultati per `VDS` con imputazione tramite media mobile a 6 ore. La previsione risulta accettabile.

Diversamente, l'applicazione della media settimanale (Figura 4.12) genera una previsione sostanzialmente piatta, che non riesce a seguire le variazioni osservate nei dati. Nonostante il dataset sia quasi privo di buchi, questa impostazione di imputazione compromette la capacità del modello di catturare i pattern sottostanti.

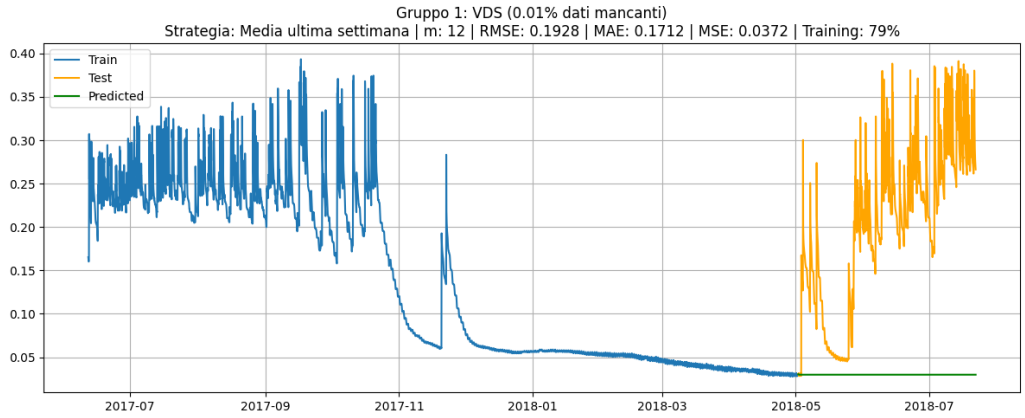

Figura 4.12: Risultati per `VDS` con imputazione tramite media settimanale. La previsione risulta piatta e poco informativa.

Questi esempi mostrano come anche in reti con copertura quasi totale la qualità della previsione possa variare sensibilmente a seconda della strategia di imputazione adottata, confermando la necessità di valutare criticamente ogni approccio prima dell'applicazione su larga scala.

4.3.3 Casi studio con copertura bassa(reti problematiche)

Un analisi in aggiunta alle reti quasi complete, è stata estesa anche a network caratterizzati da un'elevata incidenza di dati mancanti, prossima al 99%. Questi casi rappresentano scenari critici in cui la modellazione diventa particolarmente instabile, fornendo indicazioni utili sui limiti effettivi dell'approccio SARIMA. Tra le reti più significative sono state selezionate `RUSWET-GRASS` (con **39.683** dati disponibili) e `MONGOLIA` (con **109.880** disponibili), che ben esemplificano due situazioni differenti: la prima segnata da lunghi vuoti irregolari, la seconda da un campionamento molto rado ma più regolare.

RUSWET-GRASS

Il network `RUSWET-GRASS` presenta una copertura estremamente ridotta (oltre il 98% di valori mancanti), con lacune molto estese e irregolari che compromettono la possibilità di individuare una stagionalità stabile.
L'ispezione dei file di buchi mostra che in alcuni tratti le osservazioni seguono un passo pressoché regolare di circa dieci giorni, ma nella maggior parte dei casi gli intervalli sono irregolari e di lunghezza variabile. Da questa alternanza tra segmenti pseudo-periodici e vuoti prolungatisici si può ricavare uno studio utile ai fini della ricerca.

La Figura 4.13 mostra l'esito dell'interpolazione lineare con stagionalità $m = 12$. La previsione risulta del tutto implausibile, con una curva predetta che si discosta nettamente dall'andamento osservato. In presenza di vuoti così estesi, l'interpolazione introduce informazioni spurie che il modello amplifica in fase di previsione.

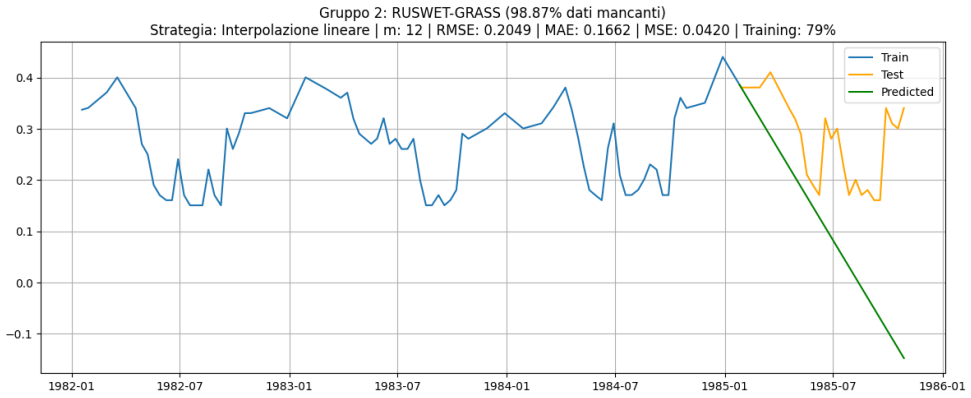

Figura 4.13: `RUSWET-GRASS` – Interpolazione lineare, $m = 12$: previsione implausibile.

Nella Figura 4.14 i dati mancanti sono stati semplicemente rimossi ($m = 12$). L'andamento predetto risulta leggermente più coerente, ma la stagionalità rimane frammentata e la qualità della previsione è comunque scarsa.

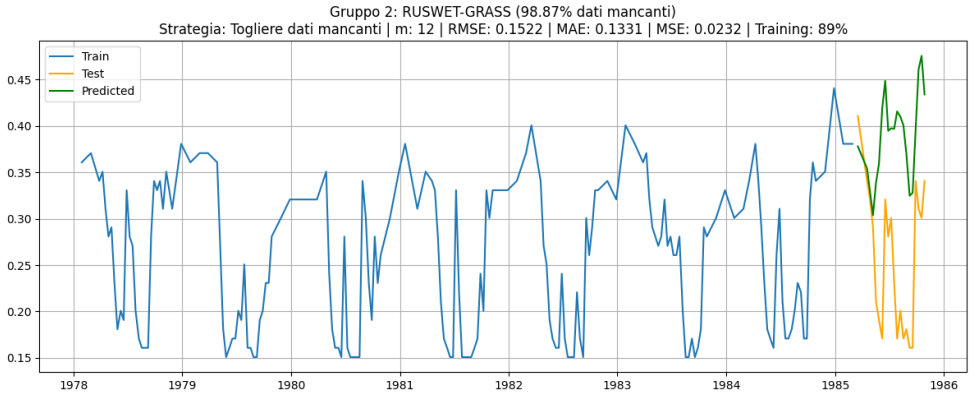

Figura 4.14: `RUSWET-GRASS` – Rimozione dei mancanti, $m = 12$: risultati leggermente migliori ma ancora inconsistenti.

Un caso particolarmente critico è rappresentato dalla media dell'ultimo giorno (Figura 4.15). La previsione appare piatta, con metriche di errore molto basse. Tuttavia, tale risultato è fuorviante: il modello restituisce un andamento costante che non riflette alcuna dinamica reale, rendendo la previsione priva di valore informativo.

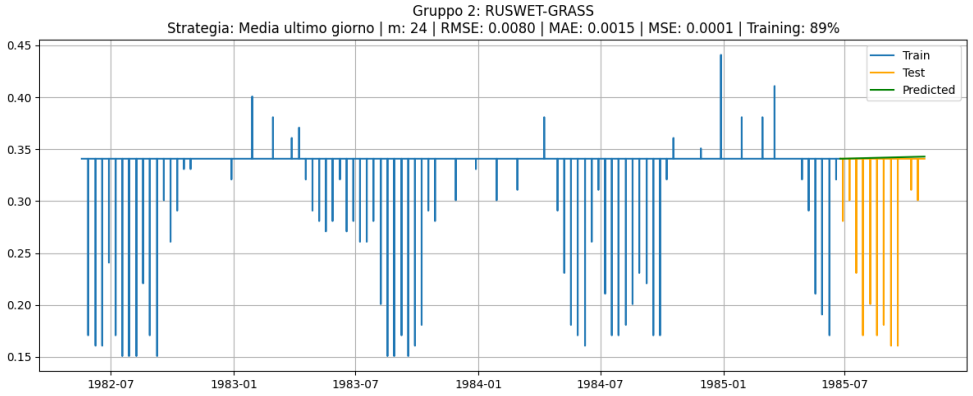

Figura 4.15: `RUSWET-GRASS` – Media ultimo giorno, $m = 24$: previsione appiattita e metriche fuorvianti.

Nel complesso, la forte discontinuità della serie `RUSWET-GRASS`, con lunghi intervalli irregolari, non consente al modello SARIMA di fornire risultati attendibili, indipendentemente dalla strategia di imputazione adottata.

MONGOLIA

Diversa è la situazione del network `MONGOLIA`, dove la copertura è anch'essa prossima al 99% ma le osservazioni disponibili seguono un campionamento sostanzialmente regolare, con valori arrivano con passo fisso $\approx$ 10 giorni. Questo rende la serie assimilabile a un dataset a bassa frequenza piuttosto che a una serie oraria frammentata.

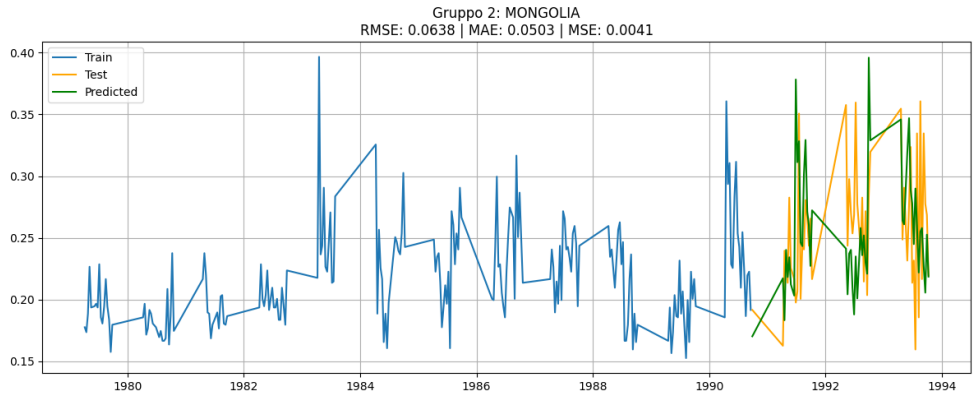

Figura 4.16: Risultati per `MONGOLIA` ($\sim$99% di dati mancanti, campionamento regolare a $\sim$10 giorni). Nonostante la bassa copertura, la regolarità temporale rende il modello più stabile rispetto ad altri casi con buchi irregolari.

Il confronto tra `RUSWET-GRASS` e `MONGOLIA` evidenzia dunque che, a parità di copertura, la regolarità degli intervalli di misura rappresenta un elemento decisivo: una serie rada ma regolare conserva un'informazione modellabile, mentre una serie densa con buchi irregolari risulta pressoché inutilizzabile.

Conclusioni e sviluppi futuri

Il punto di partenza di questo elaborato è stato un interrogativo semplice, ma cruciale: *è possibile formulare previsioni attendibili sull'umidità del suolo quando i dataset a disposizione presentano buchi temporali e in che misura queste lacune incidono sui risultati?*

L'analisi condotta mostra come la copertura dei dati sia l'elemento più determinante per l'attendibilità delle previsioni. In presenza di serie quasi complete, con una quota di osservazioni superiore al 90–95%, che seguono un campionamento sostanzialmente regolare e interruzioni di breve durata, il modello SARIMA è in grado di fornire previsioni coerenti con l'andamento reale della variabile.
Al contrario, quando i vuoti diventano estesi o ricorrenti, la stagionalità si spezza e le stime perdono consistenza, fino a risultare ingannevoli.

In questo contesto, la gestione dei dati mancanti assume un peso non trascurabile. Tecniche come l'interpolazione o l'uso di medie mobili possono risultare efficaci nel colmare intervalli limitati, ma applicate a buchi prolungati finiscono per introdurre distorsioni più gravi del problema originario.
In altre parole, non sempre "riempire" significa migliorare la serie: in certi casi il dataset rimane di fatto inutilizzabile ai fini previsionali.

Un aspetto altrettanto delicato riguarda la scelta del parametro stagionale m. Gli esempi analizzati hanno evidenziato come un valore non adatto possa compromettere le prestazioni anche su dataset quasi privi di lacune, mentre una scelta coerente con la frequenza osservativa consente di cogliere correttamente i cicli infragiornalieri.

Il confronto tra reti con caratteristiche simili ha permesso inoltre di delineare con maggiore chiarezza le condizioni in cui il modello mantiene una reale capacità predittiva.
Non va sottovalutato il valore informativo dei cosiddetti fallimenti: grafici piatti, metriche apparentemente buone ma prive di significato, previsioni inconsistenti. Proprio questi casi hanno contribuito a definire i limiti entro cui SARIMA può essere applicato con un minimo di affidabilità.
Un ulteriore elemento riguarda la quota di dati destinata al training: nei test condotti non sono emerse differenze cruciale tra uno split all'80% e uno al 90%, se non una lieve riduzione della variabilità degli errori nel secondo caso.

Da ultimo, occorre ricordare che le sole metriche numeriche non bastano. Valori di errore bassi non garantiscono di per sé previsioni sensate: una valutazione visiva e qualitativa dell'andamento stimato rimane indispensabile per giudicare l'affidabilità del modello.

Nel complesso, quanto emerso suggerisce che SARIMA può essere utilizzato con successo nella previsione dell'umidità del suolo, ma solo entro confini ben precisi: serie con elevata copertura, buchi di durata ridotta, imputazioni limitate e un parametro stagionale scelto con consapevolezza. Al di fuori di tali condizioni, la capacità predittiva risulta compromessa e le stime non possono essere considerate affidabili.

Lo studio condotto non si limita a verificare la possibilità di prevedere l'umidità del suolo in presenza di buchi temporali, ma solleva un interrogativo più ampio: *come tradurre questi risultati in strumenti utili per il settore agricolo e per una gestione più sostenibile delle risorse?*

Gli sviluppi futuri potrebbero riguardare l'estensione a modelli multivariati, l'integrazione con dati eterogenei (satelliti, sensori IoT, rilievi da drone) e l'impiego di tecniche avanzate di machine learning. Un ulteriore passo potrebbe essere la validazione in casi applicativi concreti, come la gestione irrigua o la pianificazione colturale, dove previsioni affidabili dell'umidità del suolo avrebbero un impatto diretto in termini di sostenibilità ed efficienza.

Bibliografia

[1] A. M. Turing, "Computing Machinery and Intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[2] T. M. Mitchell, *Machine Learning*. New York: McGraw–Hill, 1997.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[5] J. Han, M. Kamber, J. Pei. *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann, 2011.

[6] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth. *CRISP-DM 1.0: Step-by-step data mining guide*, 2000. URL `https://the-modeling-agency.com/crisp-dm.pdf`.(ultimo accesso: 05/10/2025).

[7] R. J. Hyndman, G. Athanasopoulos. *Forecasting: Principles and Practice*, 3rd ed. OTexts, 2021. URL: `https://otexts.com/fpp3/`. (ultimo accesso: 05/10/2025).

[8] J. Durbin, S. J. Koopman. *Time Series Analysis by State Space Methods*, 2nd ed. Oxford University Press, 2012.

[9] R. J. A. Little, D. B. Rubin. *Statistical Analysis with Missing Data*, 2nd ed. Wiley-Interscience, 2002.

[10] D. Weraikat, K. Šimunović, M. Žagar, M. Sokač. Data Analytics in Agriculture: Enhancing Decision-Making for Crop Yield Optimization and Sustainable Practices. *Sustainability* **16**(17):7331, 2024. DOI: `https://doi.org/10.3390/su16177331`.

[11] M. Zaborowicz, J. Frankowski. Big Data Analytics and Machine Learning for Smart Agriculture. *Agriculture* **15**(7):757, 2025. DOI: `https://doi.org/10.3390/agriculture15070757`.

[12] W. A. Dorigo *et al.* The International Soil Moisture Network: serving Earth system science for over a decade. *Hydrology and Earth System Sciences* **25**:5749–5804, 2021. DOI: `https://doi.org/10.5194/hess-25-5749-2021`.

[13] International Soil Moisture Network (ISMN). ISMN quality flags (documentazione web), 2024–. URL: `https://ismn.earth/data/ismn-quality-flags/` (ultimo accesso: 05/10/2025).

[14] N. Gaur, M. R. Levi, P. Knox. *Soil Moisture Data Quality Guidance*. NOAA/NIDIS – National Coordinated Soil Moisture Monitoring Network (NCSMMN), 9 dicembre 2024. DOI: `https://www.drought.gov/documents/soil-moisture-data-quality-guidance`.

[15] G. Tang, M. P. Clark, S. M. Papalexiou. The Use of Serially Complete Station Data to Improve the Temporal Continuity of Gridded Precipitation and Temperature Estimates. *Journal of Hydrometeorology* **22**(6), 2021. DOI: `https://journals.ametsoc.org/view/journals/hydr/22/6/JHM-D-20-0313.1.xml`.

Forecasting long-term monthly precipitation using SARIMA models in Kerala.

[16] G. Landeras, A. Ortiz-Barredo, J. J. López. Forecasting Weekly Evapotranspiration with ARIMA and Artificial Neural Network Models. *Journal of Irrigation and Drainage Engineering* **135**(3):323–334, 2009. DOI: https://doi.org/10.1061/(ASCE)IR.1943-4774.0000008.

[17] N. Annamalai, J. Johnson. Analysis and Forecasting of Area Under Cultivation of Rice in India: Univariate Time Series Approach. *SN Computer Science* **4**, 2023. DOI: https://doi.org/10.1007/s42979-022-01604-0.

[18] H. Zulfiqar, R. Ahmad, U. Shahzad. Hybrid ARIMA-IIS Approach for Wheat Yield Forecasting: An Integrated Approach. *iRASD Journal of Economics* **6**(1):109–127, 2024. DOI: https://doi.org/10.52131/joe.2024.0601.0197.

[19] W. A. Dorigo, W. Wagner, R. Hohensinn, S. Hahn, C. Paulik, A. Xaver, A. Gruber, M. Drusch, S. Mecklenburg, P. van Oevelen, A. Robock, and T. Jackson, "The International Soil Moisture Network: a data hosting facility for global in situ soil moisture measurements," *Hydrology and Earth System Sciences*, vol. 15, no. 5, pp. 1675–1698, 2011. DOI: https://doi.org/10.5194/hess-15-1675-2011.

[20] A. Montalvo, O. Camacho, D. Chavez, *Cyber-Physical Systems for Smart Farming: A Systematic Review*, *Sustainability*, 17(14), 2025, Art. 6393.

[21] M. Phesa, N. Mbatha, A. Ikudayisi, *MODIS Evapotranspiration Forecasting Using ARIMA and ANN Approach at a Water-Stressed Irrigation Scheme in South Africa*, *Hydrology*, 11(10), 2024, Art. 176. DOI: https://doi.org/10.3390/hydrology11100176.

[22] M. T. Kumar, M. C. Rao, *Studies on predicting soil moisture levels at Andhra Loyola College, India, using SARIMA and LSTM models*, *Environmental Monitoring and Assessment*, 195(12), 2023, Art. 1426. DOI: https://doi.org/10.1007/s10661-023-12080-1.

Dichiarazione sull'uso di strumenti di GenAI

Nel corso della redazione di questa tesi sono stati utilizzati strumenti di intelligenza artificiale generativa, in particolare ChatGPT-4 (versione rilasciata a marzo 2024). Tali strumenti sono stati impiegati principalmente per due finalità distinte: tra maggio e luglio 2025 come supporto nello sviluppo e nell'adattamento di porzioni di codice Python finalizzate all'acquisizione e alla pre-elaborazione dei dati impiegati nelle analisi SARIMA. Nel mese di settembre 2025 si è fatto invece ricorso a ChatGPT-4 per suggerimenti relativi alla formattazione LaTeX del documento (es. strutture di frontespizio e ambienti) e per chiarimenti interpretativi su alcuni risultati delle analisi.

L'uso di questi strumenti è stato limitato a funzioni di supporto: le proposte di codice e i testi suggeriti sono stati verificati, adattati e integrati manualmente dall'autore prima di ogni utilizzo; i risultati sperimentali, le scelte metodologiche e le conclusioni presentate nella tesi sono frutto dell'autore e ne costituiscono la responsabilità esclusiva. Le parti eventualmente prodotte con l'ausilio della GenAI sono state revisionate e, ove necessario, corrette dall'autore.

Ringraziamenti

Desidero innanzitutto ringraziare il Prof. Marco Di Felice, relatore di questa tesi di Laurea, che in questi mesi di lavoro ha saputo guidarmi, con suggerimenti pratici, nelle ricerche e nella stesura dell'elaborato.

Ringrazio infinitamente mio padre e mia madre, perché senza il loro supporto e pazienza non avrei mai potuto intraprendere questo percorso di studi. Vi voglio bene.

Un ringraziamento speciale va a mia sorella Anna Rita, che con pazienza ed affetto mi è stata vicina nei momenti più belli e più brutti. Grazie per aver creduto in me.

Ad Alice, che è stata sempre il mio porto sicuro: grazie per avermi riportato coi piedi per terra, per la tua presenza costante e per aver ascoltato tutti i miei pensieri, anche i più stupidi. Non ce l'avrei mai fatta senza di te. Ti voglio bene bubu.

Grazie a Michela e Lisa per aver reso il periodo universitario un percorso fatto di sostegno reciproco, incoraggiamento e complicità.

Grazie a Marco, Marco e Mirko per le nottate passate a giocare a Magic e, oltre al gioco, per il nostro tempo assieme. Tra partite, risate e chiacchiere avete reso questo percorso più leggero.

Grazie al mio gruppo di amici di Age: con voi sto bene, ci si diverte e ho trovato supporto e compagnia nei momenti più disperati.

Vorrei ringraziare i miei maestri di tennis Marco, Enrico ed Edoardo, che con dedizione e pazienza mi hanno aiutato nel mio percorso di crescita non solo sportiva, ma anche personale.

Grazie ad Arturo e al suo negozio: grazie per avermi dato un posto dove sentirmi a casa, libero di essere me stesso e divertirmi.

Ad Alberto, grazie per il supporto negli ultimi esami e per aver condiviso con me i progetti finali: la tua presenza è stata fondamentale.

A me, per aver avuto il coraggio di iniziare e la forza di arrivare fino in fondo.

Ed infine, all'unico vero fratello, Nemo, che mi è stato accanto in tutto: "WOOF".