Progetto Bacino Idrico

Marco Braga, Andrea Maver

24 giugno 2022

Abstract

Il seguente progetto si pone l'obiettivo di studiare differenti metodi di previsione per il livello dell'acqua e la portata di uscita di un bacino idrico italiano. L'utilizzo di previsioni accurate è utile per l'ottimizzazione di questa preziosa risorsa, potendo anticipare potenziali periodi di carenze o, al contrario, pericolosi eccessi.

Il report ha la seguente struttura: nella sezione 1 è presente un'introduzione sull'Industria 4.0 e sullo scenario studiato, nella sezione 2 viene brevemente esposto il dataset e le variabili che lo compongono; la sezione 3 verte sugli aspetti metodologici riguardanti il codice implementato; nella sezione 4 vengono esplorati i dati; un richiamo teorico di tutte le tecniche utilizzate è presentato nella sezione 5; la sezione 6 concerne la fase di pre-processing necessaria per manipolare il dataset e renderlo adatto ai modelli; nella sezione 7 viene esposto il training degli algoritmi, con una dettagliata descrizione dei risultati ottenuti. Nella successiva sezione 8 è presente la fase di testing del modello migliore; nella 9 viene spiegata la parte di Deploy del codice; infine verranno discussi i problemi incontrati e possibili sviluppi futuri nella sezione 10. Sono anche presenti la Bibliografia e un Appendice che contiene tabelle e risultati parziali che a causa di spazio non sono stati inseriti nel corpo principale del report, ma che comunque è giusto citare e discutere.

1 Introduzione

Lo scopo del progetto è stimare il Livello dell'Acqua e la Portata di Uscita di un bacino idrico. Tale bacino è artificiale, alimentato da cinque falde acquifere immissari e possiede una sola uscita. Venne creato con lo scopo di limitare i rischi di alluvione, fornire acqua potabile e produrre energia idroelettrica. L'azienda che lo gestisce ha l'obiettivo di utilizzare in modo sostenibile questo prezioso bene, per garantirne qualità e integrità. Poter prevedere il Livello del Bacino e la Portata con una settimana di anticipo permette di non sprecare risorse e ciò porta valore e benefici al gestore che lo ha in carico. Per poter realizzare ciò, è stato costruito un impianto di sensoristica per raccogliere i dati necessari, creando quindi un'azienda interconnessa, condizione essenziale per lo sviluppo nell'Industria 4.0.

Per Industria 4.0 si intende il nuovo modello di produzione basato sulla condivisione del lavoro e l'automazione dei processi produttivi. Lo sviluppo delle tecnologie permette di sfruttare le potenzialità di sistemi intelligenti collegati per trasformare gli ambienti di produzione, fino ad oggi separati, in universi condivisi e polifunzionali. L'Industria 4.0, quindi, si fonda sulla convergenza e l'integrazione delle tecnologie di informazione e automazione. Questo concetto nasce in Germania nel 2011, e viene successivamente declinato per il servizio idrico dalla German Water Partnership, che introduce il termine Water 4.0. Alcuni studi riguardanti l'applicazione dell'Industria 4.0 in questo settore, tra cui [1] e [2], hanno identificato tredici cluster tecnologici utili nell'affrontare le sfide emergenti dell'industria idrica. Tra questi obiettivi si annoverano: il degrado dell'infrastruttura e la manutenzione del patrimonio idrico, la garanzia della qualità della risorsa idrica, il rilevamento delle perdite, la crescente domanda di acqua, la decentralizzazione dei sistemi idrici, le conseguenze

del cambiamento climatico, il miglioramento della soddisfazione del cliente, l'aumento della produttività e dell'efficienza, la sostenibilità della tariffa, nonché la conformità alla normativa e alla regolazione. Le tecnologie abilitanti per un'Industria dell'Acqua che sia 4.0 sono sicuramente IoT, Big Data e Data Analytics, che permettono di trasformare l'industria idrica in un settore in cui le scelte sono guidate dai dati. Inoltre, l'Intelligenza artificiale e l'apprendimento automatico permettono di prevedere e avvisare in tempo reale gli operatori di eventuali problematiche sul campo. Tali tecnologie digitali, a seconda delle loro caratteristiche, possono essere integrate lungo tutta la catena di valore del ciclo dell'acqua: dalla gestione delle infrastrutture fisiche e qualità della risorsa, ai processi aziendali, fino alle relazioni con i propri partner e utenti.

Proprio in questi giorni si discute, sulle principali testate nazionali, come Rai News [4], della grave siccità che sta colpendo il Po e i comuni limitrofi, che si appoggiano al fiume per il fabbisogno di acqua: a ben 125 di essi è stato chiesto il razionamento. Meuccio Berselli, segretario generale dell'Autorità distrettuale del fiume Po ha dichiarato che in decine di comuni di Piemonte e Lombardia "sono già in azione le autobotti per l'approvvigionamento di acqua perché i serbatoi locali afferiscono a sorgenti che non ci sono più". Sempre citando Rai News: "La desertificazione sta mangiando tratti sempre più lunghi e profondi del fiume e questo ha conseguenze gravissime sulle coltivazioni, sulla biodiversità e sul settore idroelettrico, anche nell'immediato." Marco Piccinini, presidente dei frutticoltori di Confagricoltura Emilia, ha sottolineato che se la crisi idrica persisterà le conseguenze saranno anche economiche. Queste criticità potrebbero avere conseguenze minori sulla popolazione se ci fosse una gestione dell'acqua più attenta e guidata dai dati. Questo è lo scopo di una Water 4.0.

2 Dataset

Viene fornito un file *.csv* contente il dataset, composto dalle seguenti nove colonne:

- Data: giorno in cui sono state fatte le misurazioni dei parametri;
- Pioggia Zona i: quantità, in millimetri, di acqua caduta nel giorno nella zona i-esima, con i da 1 a 5 che rappresenta le cinque diverse zone in cui sono poste le falde acquifere immissari del bacino;
- Temperatura Zona 5: temperatura massima, in gradi centigradi, rilevata nel giorno nell'area 5;
- Livello Acqua: livello massimo d'acqua del bacino idrico, rispetto alla massima profondità del bacino, misurato in metri nel giorno. È la prima variabile target che si deve stimare con sette giorni di anticipo;
- Portata Uscita: massimo flusso di acqua, in metri cubi per secondo, in uscita dal bacino idrico rilevato in Data. Rappresenta la seconda variabile target.

3 Aspetti Tecnici

È stato utilizzato Python con le seguenti librerie e versioni: Python versione 3.8.12;

1. pandas versione 1.4.1; Utilizzata per maneggiare dataset tabellari
2. sklearn versione 1.0.2; Utilizzata per apprendimento automatico
3. statsmodels versione 0.13.2; Contiene la definizione dei modelli statistici per serie storiche come ARIMA
4. numpy versione 1.18.5; Fornisce supporto per gestire grandi matrici e array multidimensionali
5. keras versione 2.4.3; Utilizzata per reti neurali
6. tensorflow versione 2.3.0; Insieme a keras, permette di definire modelli di Deep Learning
7. pickle versione 4.0; Consente di salvare dati in formato pickle
8. matplotlib versione 3.5.1; Permette di visualizzare grafici

9. seaborn versione 0.11.2; Altra libreria per visualizzazioni
10. gplearn versione 0.4.2; Contiene algoritmi per la Regressione Simbolica
11. arch versione 5.2.0; Contiene la definizione dei modelli Arch e Garch
12. math; Utilizzata per definire stagionalità sinusoidali
13. os; Permette di navigare nel file system
14. random; Modulo per generare numeri pseudo-casuali

Il progetto, da un punto di vista strutturale di codice, è composto da quattro notebook, rispettivamente Preparation, Training, Testing e Deploy. Inoltre è presente una cartella con all'interno i notebook dei vari tentativi effettuati dei diversi modelli per trovarne le configurazioni ottimali. Nella cartella 'Notebook' è presente anche un file Python 'utils.py' che contiene alcune funzioni, definite da noi, che consentono di visualizzare le serie storiche in diverse modalità. Tali funzione saranno analizzate nella Sezione successiva.

4 Esplorazione del dataset

In questa sezione viene esplorato il dataset da un punto di vista analitico, per poter scoprire eventuali correlazioni o informazioni utili per la definizione dei vari modelli.

Il dataset è composto da 6386 righe e 9 colonne e tutte le variabili sono numeriche di tipo *float*, fatta eccezione per la data che è di tipo *object*. Il primo dato disponibile risale al 6 Gennaio 2003, mentre l'ultimo rilevamento è stato effettuato il 30 giugno 2020. In totale sono disponibili più di diciassette anni di dati e misurazioni. Osservando medie, deviazioni standard e percentili nelle tabelle 11 e 12 di Overview nell'Appendice, si nota come le colonne 'Pioggia Zona *i*' abbiano mediana pari a zero e un settantacinquesimo percentile molto basso rispetto al loro valore massimo, che è sempre superiore a 80 *mm*. Questo indica che la maggior parte dei valori delle precipitazioni è pari a zero. Quindi, definendo i giorni di pioggia le date in cui è caduta più di un 1 *mm* di acqua, come riferito dall'Agenzia Regionale per la Protezione dell'Ambiente (ARPA) della Lombardia [3], i giorni di pioggia sono poco più del 25%. La colonna riguardante la Temperatura nella Zona 5 ha una deviazione standard minore rispetto alle colonne della Pioggia, questo significa minor variabilità nella serie. Inoltre non è così sbilanciata come le precedenti colonne: la mediana e la media non sono eccessivamente distanti. Nessuna delle feature e nessuna delle colonne target sono standardizzate poiché nessuna ha media nulla e varianza unitaria. Inoltre sono tutte in scale e in unità di misura differenti, quindi una successiva normalizzazione nella fase di pre-processing potrebbe rivelarsi utile. I valori del Livello dell'Acqua variano tra 22.53 e 31.67, con una distribuzione spostata verso destra. Inoltre, questo target presenta la minore deviazione standard di tutto il dataset pari a 2.196. Invece, la colonna riguardante la Portata di Uscita dell'Acqua assume valori tra 0.45 e 74.65, con valore mediano di 1.50. Questo significa che, come per le variabili Pioggia, la distribuzione è fortemente asimmetrica verso sinistra e la serie è soggetta a una alta variabilità, infatti la varianza è pari a 4.098. In figura 1 si possono osservare tutte le distribuzioni.

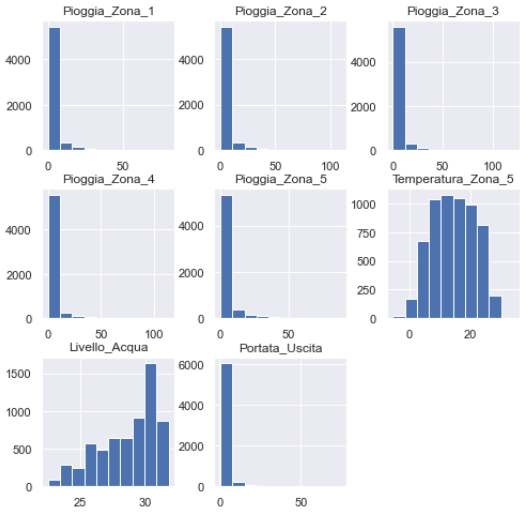

Figura 1: Distribuzioni delle variabili

Come descritto sopra, si possono notare le distribuzioni delle feature 'Pioggia Zona *i*' completamente sbilanciate verso zero, così come la Portata. La distribuzione del-

la 'Temperatura' è abbastanza simmetrica a differenza del Livello dell'Acqua che ha una distribuzione spostata verso destra, cioè verso i valori più alti.

Da questa prima analisi preliminare si nota anche la presenza di 360 valori nulli nelle colonne Pioggia e 361 per la colonna Temperatura, che corrispondono a circa un anno di rilevazioni. Ciò è dovuto all'assenza dei sensori nelle zone delle falde acquifere o a un loro mal funzionamento. Tali valori nulli e i loro corrispondenti valori per le colonne target non saranno presi in considerazione nelle successive analisi.

In figura 2, si può visualizzare la matrice di correlazione calcolata con l'indice di Pearson, definito come segue: date due variabili casuali X e Y, l'indice di correlazione di Pearson è il rapporto tra la covarianza tra X e Y e il prodotto delle deviazioni standard, cioè $\rho = \frac{\sigma_{X,Y}}{\sigma_X \sigma_Y}$. Si può notare una forte correlazione positiva, superiore a 0.85, tra tutte e cinque le colonne Pioggia. Si può ragionevolmente supporre quindi che le falde non siano particolarmente distanti e perciò subiscano, generalmente, le stesse precipitazioni. Questa forte correlazione rappresenta un problema per gli algoritmi: una possibile soluzione potrebbe essere scegliere solamente una colonna, oppure crearne una nuova come somma di tutte e cinque ottenendo così il totale di acqua caduta nelle falde acquifere. La Temperatura e le due variabili target non hanno forti correlazioni con nessuna delle altre feature. In particolare le due variabili target sono correlate positivamente tra loro ma con un coefficiente pari a 0.3.

Date due variabili statistiche, si definisce l'indice di correlazione di Spearman come l'indice di correlazione di Pearson tra le loro variabili rank. Tale indice permette di stabilire quanto bene una relazione tra due variabili possa essere descritta usando una funzione monotona. In questo caso particolare, non si notano grandi differenze tra i due indici di correlazione. Si può però osservare in figura 3 come il coefficiente di correlazione di Spearman tra la colonna Temperatura e le cinque colonne di Pioggia sia più alto in valore assoluto, ma con segno negativo. Intuitivamente più le temperature si alzano, meno

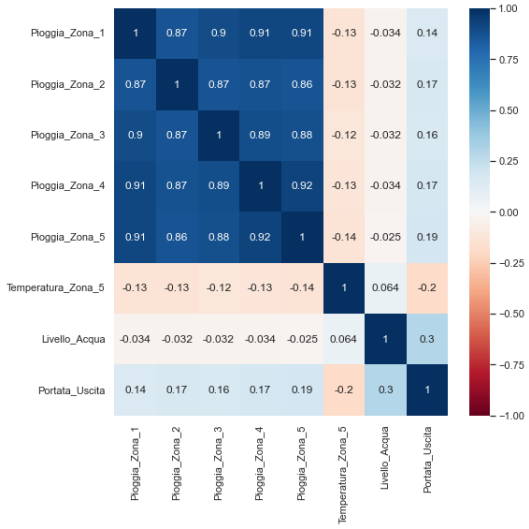

Figura 2: Correlazione di Pearson

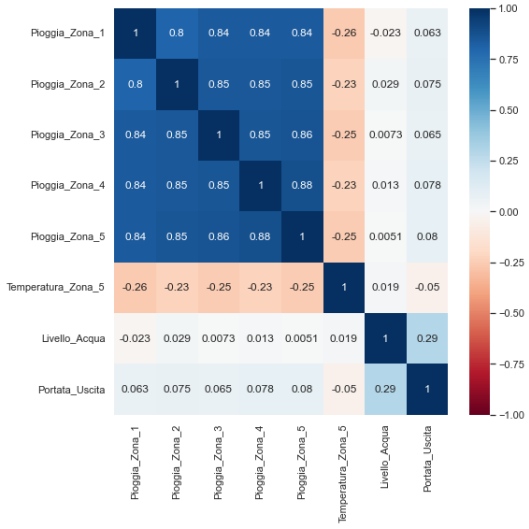

Figura 3: Correlazione di Spearman

sono probabili piogge e temporali. L'indice però rimane, in valore assoluto, più basso di 0.85 e quindi non richiede pre-processing aggiuntivo sulla colonna Temperatura. La correlazione tra le due colonne target rimane positiva con un valore di 0.29 che non aggiunge informazioni rispetto all'indice di Pearson. La correlazione tra le colonne target e le altre colonne feature è, in modulo, inferiore a 0.1, quindi non è presente alcuna correlazione.

Come notato in precedenza, entrambe le correlazioni tra le due feature target sono positive ma basse in valore assoluto. Per studiare più approfonditamente la relazione tra il Livello dell'Acqua e la Portata, si visua-

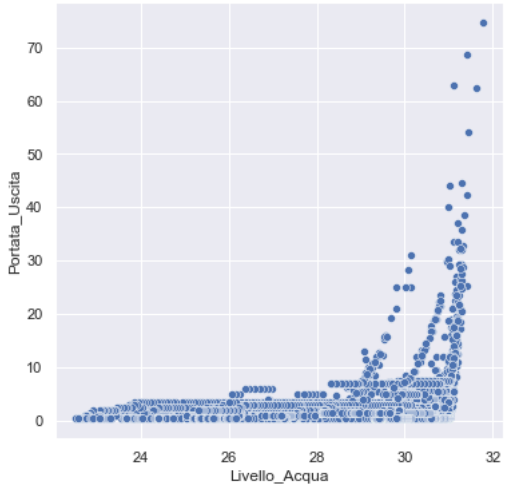

Figura 4: Scatterplot tra Livello e Portata

lizza lo scatterplot in figura 4. Non sembra esserci una chiara relazione tra Livello e Portata. Si può notare però che i valori massimi assunti dalla Portata corrispondono ai valori più alti assunti dal Livello. È ragionevole pensare che più acqua è presente nel bacino, e quindi più aumenta il livello, più acqua dovrà defluire per evitare eccessi, aumentando la Portata. Si potrebbe ipotizzare quasi una dipendenza esponenziale tra i due attributi, in quanto sembra esserci una forma piatta e un successivo aumento molto veloce.

Ci si concentra ora sullo studio delle feature e dei target singolarmente.

Prima di commentare i grafici riguardanti le serie storiche disponibili, si definiscono le funzioni nel file 'utils.py' che permettono di stampare le visualizzazioni. In 'utils.py' è definito un oggetto 'print plot', che richiede come input il dataframe che contiene la colonna che si vuole visualizzare. Si possono richiamare le seguenti funzioni:

- plot series: stampa la serie storica dal primo all'ultimo giorno
- plot years serie: stampa una serie storica per ogni anno. Con anno si intendono 365 giorni successivi partendo dalla prima data disponibile, poi i secondi 365 e così via, non necessariamente gli anni solari.
- boxplot annuali: plot del boxplot per ogni anno solare.
- plot stazionario varianza: plot della relazione tra media e varianza calcolata per ogni anno solare.

4.1 Livello Acqua

La serie che rappresenta il Livello dell'Acqua all'interno del bacino idrico è mostrata nella figura 5. Si nota una periodicità composta da picchi e avvallamenti che, nonostante abbiano incidenza diversa, sembrano piuttosto regolari. Nella figura 6 si visualizza la serie annuale per otto anni successivi, creata tramite la funzione 'plot years serie' definita sopra. Si osserva come durante un singolo anno il Livello assuma valori più alti nei mesi invernali, ai quali segue una discesa costante nella stagione estiva. Tuttavia, i diversi anni presentano un'alta variabilità tra loro, soprattutto nei mesi di novembre e dicembre, durante i quali le risalite del Livello, successive al mese estivo, assumono comportamenti molto differenti.

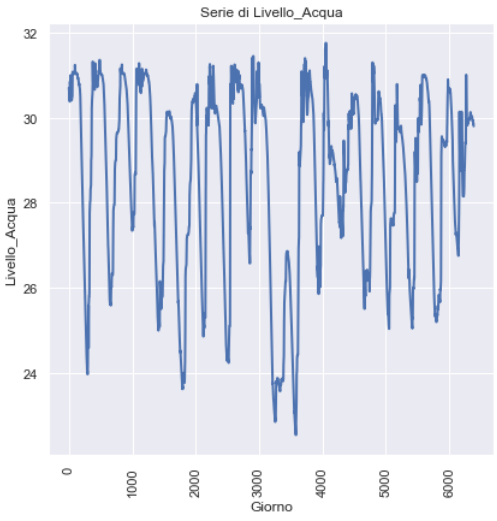

Figura 5: Livello Acqua

I box-plot in figura 7, calcolati grazie alla funzione 'boxplot annuali', avvalorano questa considerazione: si nota come il valore medio oscilli fortemente a seconda dell'anno e come le fasce dei percentili abbiano sempre dimensioni differenti. Questo suggerisce anche una possibile non stazionarietà sia in media che in varianza per la serie: la media non sembra infatti essere costante e inoltre la variabilità per ogni singolo anno varia. Questo rappresenta un problema da risolvere

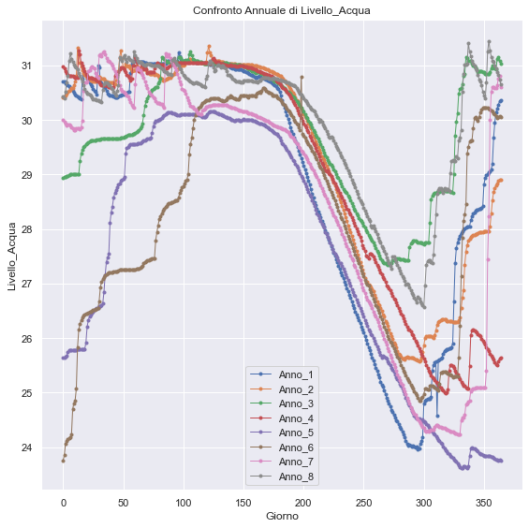

Figura 6: Confronto Livello per 8 anni

prima di poter applicare i modelli della famiglia ARIMA, i quali richiedono una serie stazionaria come punto di partenza.

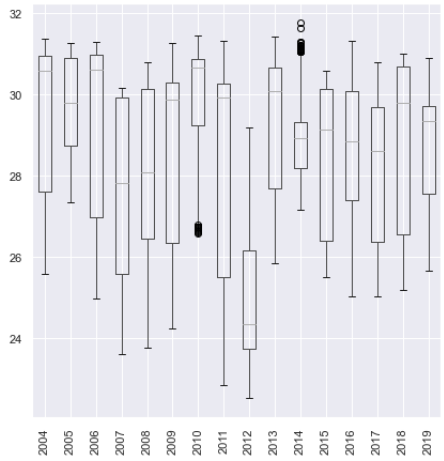

Figura 7: Box-plot annuali Livello

Il grafico in figura 8 mette in relazione la media (asse X) e la varianza (asse Y) per ogni anno di osservazioni e mostra una retta di regressione lineare per i punti rappresentati; è ottenuto tramite la funzione plot stazionario varianza di *utils*. È un test visivo per la stazionarietà: se i punti sono ben modellati dalla regressione lineare, significa che la varianza aumenta o diminuisce al variare della media. In questo caso si può individuare un anno outlier che sbilancia la retta di regressione. Tolto questo punto anomalo, i rimanenti si dispongono lunga una retta. Quindi questo grafico suggerisce, nonostante il punto outlier, una non stazionarietà in varianza.

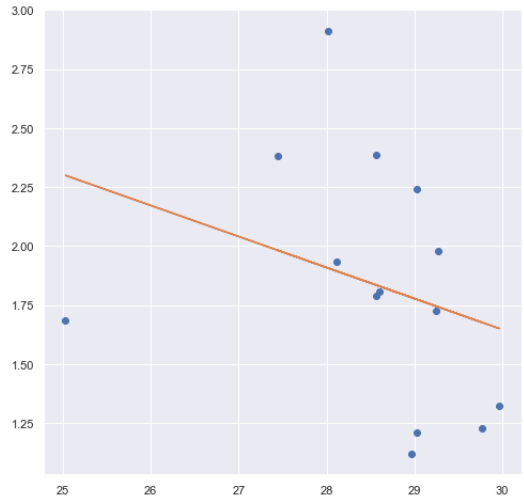

Figura 8: Relazione tra media e varianza per anno

4.2 Portata Uscita

Osservando la serie della Portata Uscita in figura 9 si nota come per tutta la durata delle rilevazioni questa si assesti intorno a valori molto bassi, spesso rasenti lo zero. Sono però presenti dei picchi molto alti e ripidi, che non sembrano seguire alcun tipo di periodicità; ciò potrebbe rivelarsi critico nelle future previsioni. In particolare, solamente sessantasette giorni, cioè 1.8% delle osservazioni totali, hanno presentato una Portata maggiore di 20 metri cubi al secondo. Andando nello specifico per 8 anni successivi, figura 10, si nota come la prima metà dell'anno sia quella che presenta il maggior numero di picchi, i quali però occorrono sempre in periodi diversi. Alcuni anni invece non ne presentano alcuno nei primi mesi dell'anno, assumendo invece valori alti verso la fine.

I box-plot, figura 11, mostrano, come era prevedibile, delle distribuzioni fortemente schiacciate, con outlier molto evidenti per ogni annata. Anche in questo caso la variabilità è molto diversa anno per anno, così come la media. Ciò suggerisce una non stazionarietà in media e in varianza, come per la serie Livello.

Nel grafico in figura 12, viene mostrata una retta di regressione lineare per media e varianza annuali. Corrisponde allo stesso

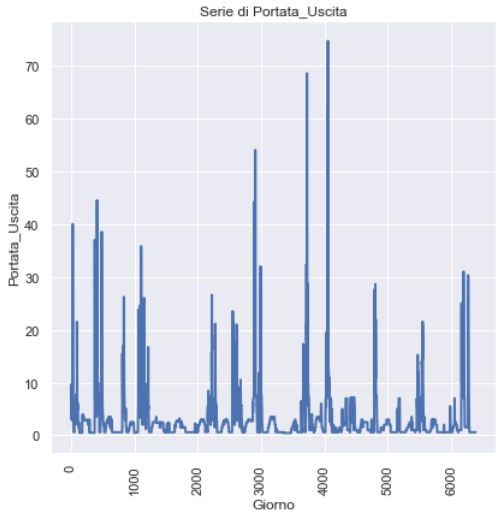

Figura 9: Portata Uscita

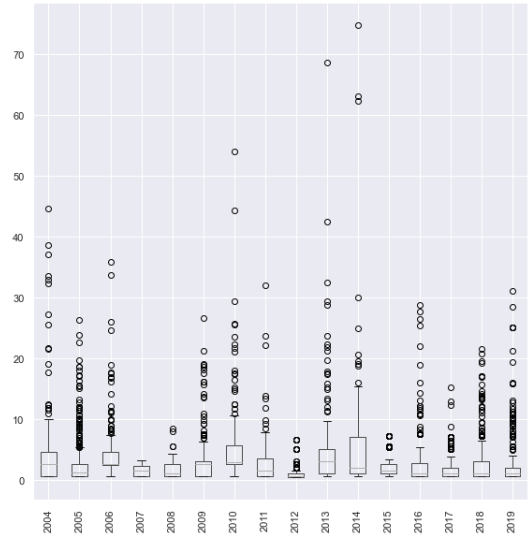

Figura 11: Box-plot annuali Portata

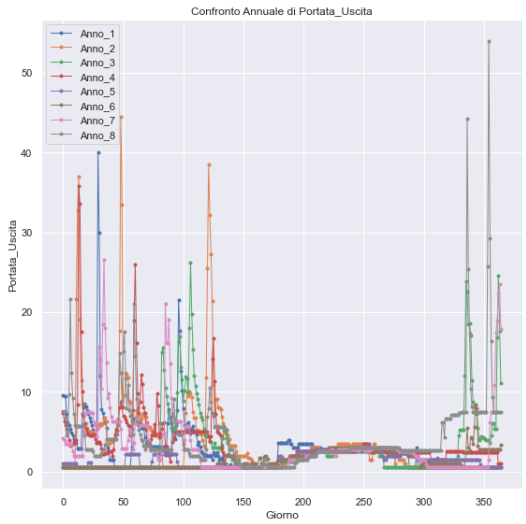

Figura 10: Confronto Portata per 8 anni

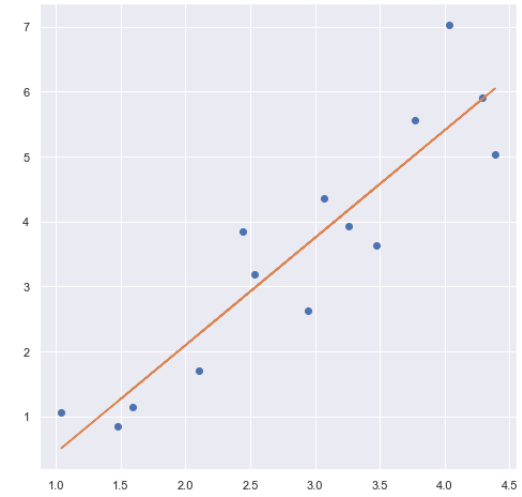

Figura 12: Relazione tra media e varianza per anno

test visivo del Livello Acqua. In questo caso i punti si dispongono lungo una retta, senza outliers. Quindi questo grafico suggerisce, una non stazionarietà in varianza".

Per risolvere la non stazionarietà in varianza è possibile applicare il logaritmo alle colonne target e studiare le nuove serie ottenute. Per risolvere la non stazionarietà in media si può applicare un trend o una differenziazione ai modelli ARIMA. Questi argomenti verranno approfonditi successivamente nella sezione apposita.

4.3 Pioggia Totale

Come affermato nei paragrafi precedenti, la forte correlazione tra le varie colonne riguardanti la Pioggia risulta essere un problema, soprattutto per i metodi, come la Regressione Lineare, che richiedono una non collinearità tra gli attributi. Per questo motivo si è deciso di creare una nuova variabile denominata 'Pioggia Totale' e definita come somma dei valori delle cinque zone. In questo modo si ottiene il valore totale, espresso in millimetri, di acqua caduta nelle cinque falde immissari. La serie riguardante la quantità totale di Pioggia è osservabile in figura 13. Non sembra esserci una chiara periodici-

tà e come per la Portata sono presenti picchi molto alti. Anche osservando anno per anno, in figura 14, si nota come, nella gran parte dei giorni, assuma valore molto vicino a zero e siano presenti picchi non regolari. Si può notare però una diminuzione delle precipitazioni tra i mesi di giugno e luglio, osservabile nella fascia centrale della figura, durante i quali la serie non presenta valori particolarmente alti. Le precipitazioni non hanno un comportamento regolare e sono molto difficili da prevedere, infatti non è presente alcuna periodicità ben definita; è però ragionevole supporre che ci siano più rovesci nei mesi invernali e autunnali rispetto alla stagione estiva. Il totale di Pioggia caduta segue la seguente distribuzione: media pari a 14.28, ma con una deviazione standard elevata pari a 36.62. Il minimo e il primo quartile sono pari a zero, la mediana invece è pari a 0.2, mentre l'ultimo quartile è pari a 7. Infine il massimo valore assunto dalla serie è 425.6. Da questi dati si nota una serie sbilanciata verso sinistra, poiché la mediana è nettamente inferiore rispetto alla media.

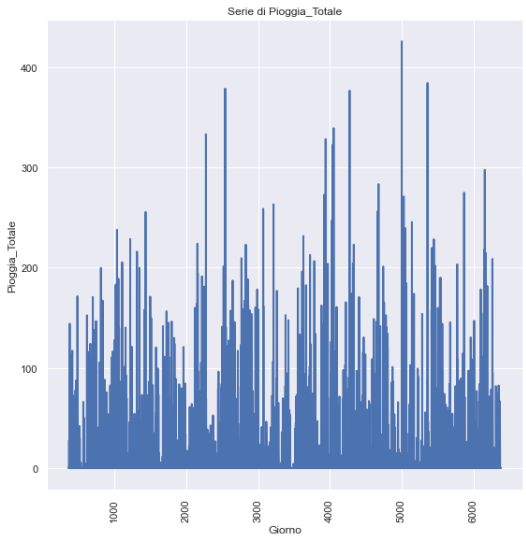

Figura 13: Pioggia Totale

Una nuova matrice di correlazione secondo l'indice di Pearson si può visualizzare in figura 15: al posto delle cinque colonne Pioggia, è presente l'attributo 'Pioggia Totale'. Non è presente una forte correlazione tra questa nuova variabile e le altre colonne del dataset, poiché, in modulo, tutti i valori sono minori di 0.17.

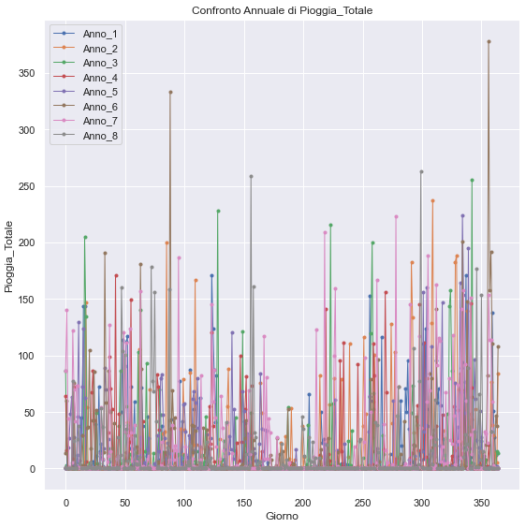

Figura 14: Confronto Pioggia per 8 anni

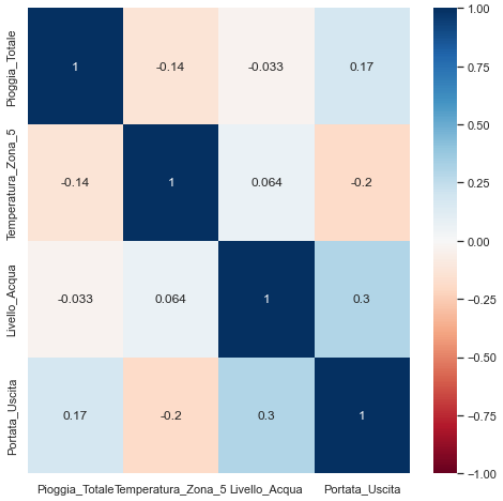

Figura 15: Nuova matrice di correlazione

In Appendice sono anche visibili due scatterplot che mettono in relazione la Pioggia Totale con i due target(figura 38): non si riesce a dedurre una chiara relazione tra questa feature e i due target, i punti sono sparsi nel piano bidimensionale.

4.4 Temperatura

Osservando la serie totale della Temperatura Zona 5 e quella divisa anno per anno in figura 16 e 17, si nota una forte e ben definita periodicità. Picchi e avvallamenti sono tutti all'incirca della stessa intensità, anche se tra anno ed anno ci sono molte oscillazioni a livello giornaliero. Come è ragionevole pensare la temperatura cresce costantemente fino

ai mesi estivi, per poi scendere in inverno. Questa è sicuramente la serie più regolare tra quelle visualizzate; inoltre non sembra esserci un trend evidente. Invece, osservando gli scatterplot inserirti nell'Appendice (figura 39), non si notano particolari relazioni tra questa variabile e i target, a parte il fatto che le precipitazioni più intense siano state registrate in giorni con temperature intorno ai 10 °C.

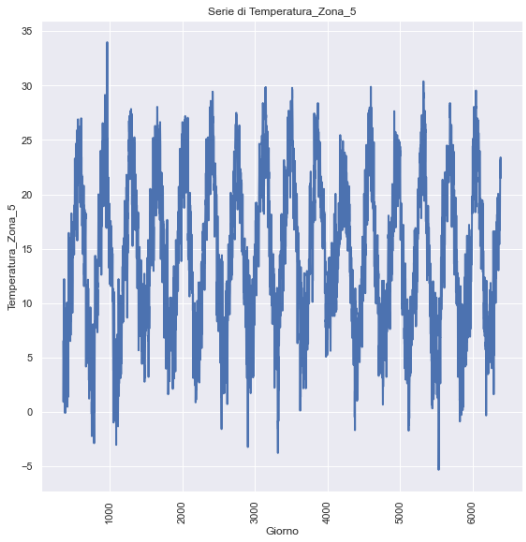

Figura 16: Temperatura Zona 5

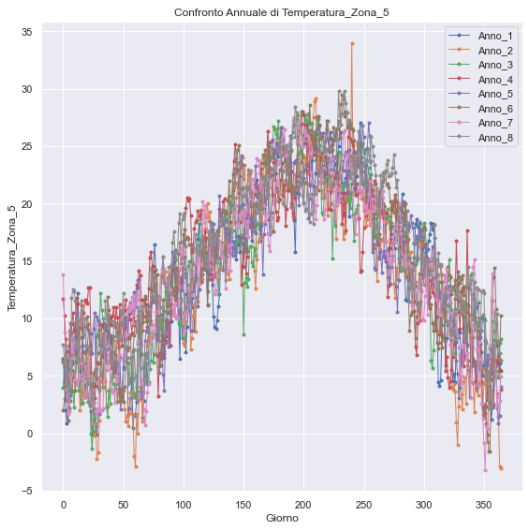

Figura 17: Confronto Temperatura Zona 5 per 8 anni

5 Aspetti metodologici

In questa sezione viene presentata la teoria dietro i modelli utilizzati per le previsioni, in modo che possano essere compresi al meglio, sia a livello di funzionamento che di risultati. I modelli possono essere divisi in tre famiglie: i modelli statistici, basati su fondamenti teorici statistici, i modelli di Machine Learning, cioè algoritmi generici di apprendimento artificiale che vengono adattati per essere applicati a serie temporali e infine le reti neurali.

5.1 Modelli statistici

I Modelli statistici che sono stati utilizzanti sono i seguenti: Regressione Lineare, Sarimax, Arch e Var.

5.1.1 Regressione Lineare

Sia Y una variabile casuale che si vuole prevedere attraverso una funzione $P(X_1, ..., X_n)$ dove $X_1, ..., X_n$ sono variabili casuali che posso osservare e P è una funzione incognita. Sia $\hat{Y} = P(X_1, ..., X_n)$, allora l'errore di previsione si definisce come $E_r = Y - \hat{Y}$. Sia $\mathcal{L}$ una funzione di perdita o di costo, allora l'obiettivo di una regressione statistica è minimizzare, sull'insieme di tutte le possibili funzioni P, $E[\mathcal{L}(E_r)]$ dove con E indica il valore atteso. Spesso come funzione di perdita viene utilizzata $\mathcal{L}(E) = E^2$ perché è una funzione continua, convessa e derivabile, quindi semplice da ottimizzare. Una regressione si definisce lineare quando l'insieme delle funzioni su cui si cerca il minimo del valore atteso è ridotto alle funzioni lineari, cioè $P(X_1, ..X_m) = \beta_0 + \beta_1 X_1 + ... + \beta_n X_n$. Esiste una variante di questo modello detta Regressione Ridge in cui si penalizza il modello aggiungendo la norma due del vettore dei coefficienti da stimare. Questa aggiunta permette di evitare overfitting e altri problemi tipici dell'addestramento autonomo degli algoritmi.

Come si può notare dalla definizione, questo è un modello generico, non è proprio delle serie storiche, anche perché non sono presenti assunzioni sulla dipendenza temporale

delle osservazioni. A causa della sua semplicità è stato utilizzato come modello baseline, di confronto rispetto agli altri modelli.

5.1.2 SARIMAX

Prima di descrivere i Modelli ARMA, e le loro varianti, è necessario definire il concetto di Stazionarietà (debole). Sia $\{X_t\}_t$ una serie storica, cioè una serie indicizzata di variabili casuali. $\{X_t\}_t$ si dice stazionaria in senso debole se:

- $\exists\ \mu = E(X_t)\ \forall t$,
- $\exists\ \mu = Var(X_t) < \infty\ \forall t$,
- $Cov(X_t, X_{t-h}) = \gamma_h$ dove $h \in Z$, cioè la covarianza dipende solo dalla distanza h tra le variabili e non da t.

Sia ora $\{Y_t\}_t$ una serie storica, siano X_t i valori degli attributi al tempo t, allora il modello $ARIMAX(p, d, q)$ si definisce come segue: $\phi_p(B)\Delta^d(Y_t - \beta^T X_t) = \theta(B)\epsilon_t$ dove ϵ_t rappresenta un white noise con varianza costante σ^2, B definisce l'operatore di backward cioè $B(Y_t) = Y_{t-1}$ e $B^k(Y_t) = Y_{t-k}$, $\phi_p(B) = 1 - \phi_1 B - ... - \phi_p B$ è un polinomio in B dove ϕ_i rappresentano i parametri da stimare per la parte AutoRegressiva (AR) del modello, $\theta_q(B) = 1 + \theta_1 B - ... - \theta_q B^q$ è il polinomio in B per la parte a Media Mobile (MA) del modello dove i θ_i rappresentano la seconda parte dei parametri da stimare, $\Delta^d = (1 - B)^d$ rappresenta l'operatore di differenziazione e la parte di Integrazione (I) del modello, β^T è il vettore trasposto dei coefficienti per i regressori esterni X_t.

Nel modello descritto precedentemente si può introdurre una componete stagionale data, per esempio, da alcuni comportamenti periodici della serie. Allora il modello diventa (senza introduzione di regressori esterni) un $SARIMA(p, d, q)(P, D, Q)_s$ definito come $\Delta^d \Delta_s^D Y_t = \frac{\theta_q(B)\Theta_Q(B)}{\phi_p(B)\Phi_P(B)}\epsilon_t + \mu$ dove $\Delta^d, \phi_p(B)$ e $\theta_q(B)$ sono gli stessi definiti in precedenza, mentre $\Delta_s^D = (1 - B^s)^D$ rappresenta la differenziazione stagionale dove s è la stagionalità, $\Phi_P(B) = 1 - \Phi_1 B^s - ... - \Phi_p B^{p*s}$ rappresenta la componente stagionale Autoregressiva, $\Theta_Q(B) = 1 + \Theta_1 B^s - ... - \Theta_q B^{q*s}$ rappresenta la parte stagionale a Media Mobile e infine μ assume il ruolo di media o drift. I coefficienti da stimare sono ϕ_i, θ_i definiti in precedenza a cui si aggiungono Φ_i e Θ_i.

Questi metodi nascono per esprimere le varie dipendenze passate presenti in una serie storica. Infatti, il valore al tempo t dipende dai valori della serie nei passi precedenti. L'indice di questi passi dipenderà dalla stagionalità e dalla serie stessa. Il Modello SARIMAX è il principale e più semplice algoritmo applicato alle serie temporali, permette di inserire una componente trend o dei regressori esterni per migliorare le previsioni.

5.1.3 Arch e Garch

Un modello Autoregressivo a Eteroschedasticità Condizionata (Autoregressive Conditionally Heteroscedastic o Arch) è un modello il cui obiettivo è stimare la varianza di una serie storica, in particolare nel caso in cui questa varianza sia mutevole. I modelli ARCH nascono nel contesto di problemi econometrici e finanziari che trattano, tra le altre, le serie dei prezzi di alcuni prodotti, che per natura sono caratterizzate da un'alta volatilità, cioè variabilità. Come notato nella sezione di esplorazione, la serie della Portata è caratterizzata, in alcuni giorni, da alti valori, mentre per il resto delle osservazioni assume valori molto bassi. Ciò implica un'alta variabilità della serie e di conseguenza anche un alto valore per la varianza. Nonostante non sia una serie economica, ma proprio per la sua struttura, si è deciso di applicare i modelli Arch e Garch alla serie della Portata, pensando che possano essere efficaci anche in questo particolare caso. Un modello $Arch(m)$ è definito come segue: sia $\{Y_t\}_t$ una serie storica, allora $Var(Y_t|Y_{t-1}, ...Y_{t-m}) = \sigma_t^2 = \alpha_0 + \alpha_1 Y_{t-1}^2 + ... + \alpha_m Y_{t-m}^2$. Mentre il modello $Garch(m, n)$ si definisce come $Var(Y_t|Y_{t-1}, ...Y_{t-m}) = \sigma_t^2 = \alpha_0 + \alpha_1 Y_{t-1}^2 + ... + \alpha_m Y_{t-m}^2 + \beta_1 \sigma_{t-1}^2 + ... + \beta_m \sigma_{t-m}^2$.

5.1.4 VAR

Il modello Autoregressivo Vettoriale (Vector AutoRegression o VAR) rappresenta un'e-

stensione multi-variata del precedente modello ARIMA. Si definisce come segue: $\mathbf{Y}_t = \Phi(B)\mathbf{Y}_{t-1} + \epsilon_t = \Phi_1\mathbf{Y}_{t-1} + ... + \Phi_p\mathbf{Y}_{t-p} + \epsilon_t$, dove, per un $VAR(p)$, $\Phi(B) = \sum_{i=0}^{p-1} \Phi_i B^i$ è un polinomio matriciale di ordine p nell'operatore ritardo B; Y_t è un vettore di variabili nella forma $Y_t = \begin{bmatrix} y_{1t} \\ ... \\ y_{nt} \end{bmatrix}$ e ε_t è un vettore conforme di disturbi stocastici tale che $E(\varepsilon_t) = 0$ e $E(\varepsilon_{it}^2) = \sigma_i^2, i = 1, ..., n$. Questo modello è stato scelto perché permette di prevedere allo stesso passo entrambe le colonne target, andando a scoprire delle potenziali relazioni non individuate in precedenza. Un'altra caratteristica è che gli attributi vengono considerati delle variabili da prevedere. Quindi non è necessario traslare indietro le colonne target, in quanto ad ogni passo t il modello prevede gli attributi e le colonne target allo stesso passo t, senza bisogno di variabili esterne. Prevedere il valore degli attributi a sette giorni non rientra tra gli obiettivi del progetto, ma il modello sfrutta le dipendenza tra le previsioni dei vari dati e questo aiuta per gli scopi del report.

5.2 Modelli machine learning

I modelli di machine learning scelti per il forecast sono la Symbolic Regression e il K-Nearest Neighbour.

5.2.1 Symbolic Regression

La Symbolic Regression rappresenta una generalizzazione del concetto di regressione lineare. Questo algoritmo cerca la funzione matematica che meglio approssima la variabile di output all'interno dello spazio di tutte le formule matematiche possibili, partendo da semplici operazioni come somma, differenza, funzioni trigonometriche e esponenziali. Nonostante sia difficile produrre dei buoni modelli, questi hanno una grande qualità che risiede nella loro spiegabilità. Inoltre, problemi di overfitting sono spesso evitati, a causa del fatto che il modello cerca di mantenere le formule più corte possibili, non necessariamente quelle con accuratezza più alta. Lo svantaggio di questo metodo, come sarà spiegato più dettagliatamente nella Sezione riservata al Training, è il costo computazionale, infatti la ricerca della funzione ottimale richiede diverse iterazioni, che possono rivelarsi molto lunghe e dispendiose.

5.2.2 K-Nearest Neighbour

Il K-Nearest Neighbours (KNN) è una delle tecniche più conosciute e utilizzate per task di classificazione. Una osservazione, definita come insieme di attributi e un'etichetta, viene classificata in base alle label dei K dati i cui attributi sono più vicini, nello spazio delle feature, a quelli dell'osservazione da classificare. Il concetto di vicinanza può essere espresso attraverso diverse misure, come le distanze Euclidea o L^p. Questa tecnica può essere adattata per le serie storiche: scelta una sotto-sequenza della serie, l'algoritmo cerca le K sotto-sequenze più simili ad essa; osservando i valori successivi a queste sequenze, il modello produce una previsione facendo la media o la mediana di questi valori. La lunghezza della sotto-sequenza, il numero di sotto-sequenze simili da individuare e la quantità di dati da considerare per la previsione sono tutti iper-parametri decisi dal ricercatore.

Questa tecnica ha una particolarità: non è presente un modello sottostante, come nel caso di una Rete Neurale, di una generica SVM o dei modelli statistici. Lo scopo di questo modello è individuare nel passato pattern simili a quelli che si ripresentano nel presente e che rappresentano la base della previsione futura. Nonostante la sua semplicità, si rivela, in molti casi, un'ottima metodologia per le serie storiche.

5.3 Reti Neurali

Le reti neurali sono un tecnica di Machine Learning usata sia per la regressione che per la classificazione e rappresentano l'elemento centrale degli algoritmi di Deep Learning. Il loro nome e la loro struttura sono ispirati al cervello umano, imitando il modo in cui i neuroni biologici si inviano segnali. Le reti neurali artificiali sono composte da diversi *layer* di neuroni. Ciascuno di essi si connette ad un altro neurone di un layer successivo

e ha associati un peso e una soglia. Esistono diverse tipologie di reti neurali: feedforward, convoluzionali e ricorrenti. Queste ultime sono la tipologia più adatta per trattare serie storiche o dati sequenziali come i documenti testuali. Si distinguono dalle reti feedforward e convoluzionali per l'introduzione di una componente mnemonica, cioè le reti ricorrenti prendono informazioni dagli input precedenti per influenzare l'input e l'output correnti. Le reti neurali tradizionali presuppongono che input e output siano indipendenti l'uno dall'altro, mentre l'output delle ricorrenti dipende dagli elementi precedenti all'interno della sequenza. Per costruire la rete ricorrente appena descritta sono state utilizzate due sue varianti: le reti Long Short-Term Memory (o LSTM) e quelle Gated Reccurent Units (o GRU). Le reti LSTM sono state introdotte come soluzione al problema del Vanishing Gradient, che ricorre spesso nelle reti ricorrenti. Affrontano il problema delle dipendenze a lungo termine: se lo stato precedente che sta influenzando la previsione attuale non è nel passato recente, il modello potrebbe non essere in grado di prevedere con precisione lo stato attuale. Per rimediare a questo, i neuroni delle reti LSTM hanno tre gate: uno di ingresso, uno di uscita e un forget gate. Queste porte controllano il flusso di informazioni necessarie per prevedere l'output nella rete. Le reti GRU sono simili alle LSTM ma possiedono solamente due gate: reset and update gate, che controllano quante e quali informazioni conservare dal passato.

6 Pre-processing

Questa sezione riguarda tutti i passaggi e modifiche apportate al dataset per poterlo rendere utilizzabile al meglio dagli algoritmi.

Il primo step consiste nel trasformare il formato della colonna Data da *object* a *date*, prestando attenzione alla differenza tra la notazione anglosassone e quella italiana riguardo l'ordine di giorni e mesi. Successivamente vengono create due nuove colonne, 'Year' e 'Month', utili per la creazione delle visualizzazioni mostrate nella precedente sezione di esplorazione. Dopo l'eliminazione delle righe contenenti valori nulli, il dataset è composto da 6025 righe, e si scelgono solo le seguenti colonne per proseguire con le analisi: Data, Year, Pioggia Totale, Temperatura Zona 5, Livello Acqua, Portata Uscita. Prima della divisione in train e test, viene eseguita un'importante operazione: per la successiva implementazione dei modelli ARIMA è utile utilizzare delle sinusoidi per descrivere la stagionalità annuale deterministica, cioè priva di componenti casuali. Si è deciso di introdurle all'interno dei dataset con un processo a ritroso, cioè dopo aver trovato il numero ottimale nella fase di training, si è modificato il dataset alla radice durante il pre-processing; le sinusoidi scelte dopo vari test sono quattro, cioè due seni e due coseni, entrambi di frequenza $\frac{2\pi}{365.25}$. Quel 0.25 serve a modellare la presenza degli anni bisestili, considerati aggiungendo ogni volta un quarto di anno.

A questo punto il dataset può essere diviso in un dataset di training e uno di test. La suddivisione viene effettuata mantenendo una percentuale di 80% del dataset originale per il train, che corrisponde a 4820 righe, e 20% per il test, cioè le rimanenti 1205 osservazioni. I dati non vengono mescolati e ne viene mantenuto l'ordine; infatti, per lavorare con le serie storiche, si impara dal passato per prevedere il futuro, perciò la sequenzialità dei dati è fondamentale. A questo punto vengono creati dei dataset nuovi utilizzando la funzione *collect windows* definita all'interno del file *utils*: questa permette di laggare in avanti di sette giorni i valori della colonna target scelta, eliminando le righe con valori nulli che vengono generate da questa operazione. Il risultato sono quattro nuovi dataset, denominati df train portata, df test portata, df train livello e df test livello. I dataset Livello hanno, in ogni riga, la Pioggia Totale, la Temperatura Zona 5 e la Portata al tempo t, mentre il Livello è al tempo $t+7$. Mentre per i dataset della Portata, gli attributi sono la Pioggia Totale, la Temperatura Zona 5 e il Livello al tempo t, mentre la Portata è al tempo $t+7$. Naturalmente tutti i dataset elencati contengono anche le sinusoidi create in precedenza. Successivamente viene creata un'ulteriore versione specifica per le

reti neurali, che contiene entrambi i target traslati sette giorni in avanti. Quindi, ogni riga di questo dataset è composta come segue: la Pioggia Totale, la Temperatura Zona 5 al tempo t, mentre la Portata e il Livello al tempo $t + 7$. Successivamente questo dataset viene suddiviso in train e test utilizzando le percentuali sopra definite. Utilizzando i quattro dataset precedenti, ne vengono creati dei corrispettivi con valori degli attributi standardizzati, lasciando intatti i dati target. Viene utilizzato uno *StandardScaler* che viene addestrato sui dataset di train e successivamente applicato quelli di test. Nascono così i dataset df train prep portata, df test prep portata, df train prep livello e df test prep livello. Come in precedenza viene creata anche la versione normalizzata dei dataset specifici per le reti neurali. Vengono anche aggiunte delle colonne contenenti il logaritmo del target all'interno di tutti i dataset appena creati. Lavorare con il logaritmo di una serie permette di riportare ad una stazionarietà in varianza, che è uno dei requisiti per applicare i modelli Arima.

L'ultimo step consiste nella creazione di nuove colonne personalizzate per i dataset normalizzati di Livello e Portata. Per il Livello viene aggiunto il tempo, che va da uno fino al numero di righe del dataset, il tempo al quadrato e l'intercetta che assume tutti valori 1. Le prime due rappresentano un trend deterministico quadratico, scelto dopo aver tentato diverse configurazioni, e perché è evidente che non sia presente una relazione lineare. La terza è necessaria per per introdurre l'intercetta nei modelli OLS. Per la Portata, invece, viene creata la colonna intercetta, identica alla precedente, il tempo, questa volta definito come il seno del logaritmo del tempo standard, per simulare una funzione con picchi e quindi non lineare, e infine il Livello al quadrato, per cercare di ricostruire il comportamento con il target osservato nella sezione di esplorazione. Come anticipato per le sinusoidi, la scelta di introdurre queste nuove variabili è stata presa nella fase di training per migliorarne i risultati, ma è stato deciso di inserire questi cambiamenti già nella fase di pre-processing per una questione di ordine e coerenza con il resto del codice. Prima dell'addestramento di ogni modello si sceglieranno accuratamente solo le colonne necessarie a quel tipo di algoritmo.

Per concludere questa fase, tutti i dataset creati vengono salvati in formato *plk.zip* per occupare meno spazio sul disco. Ricapitolando, i quattordici dataset salvati sono i seguenti:

- df train portata e df test portata: colonna Portata traslata sette gionri in avanti e attributi non manipolati;
- df train prep portata e df test prep portata: colonna Portata traslata a sette giorni in avanti e attributi normalizzati;
- df train livello e df test livello: colonna Livello traslata e attributi non manipolati;
- df train prep livello e df test prep livello: colonna Livello traslata e attributi normalizzati;
- df train multivariato e df test multivariato: dati originali senza nessuna alterazione, viene utilizzato per il modello Autoregressivo Multivariato;
- df train reti e df test reti: sia Portata che Livello traslati e attributi non manipolati;
- df train prep reti e df test prep reti: sia Portata che Livello traslati e attributi normalizzati.

Vengono salvati anche gli *StandardScaler* utilizzati per la normalizzazione. Uno *StandardScaler* permette di standardizzare gli attributi del dataset in cui il target è il Livello, l'altro quando il target è la Portata. È necessario crearne due distinti perché gli attributi sono diversi per ognuno dei due target.

7 Training

In questa sezione viene spiegata in modo dettagliato la fase di training di tutti i modelli, facendo particolare attenzione alle scelte che sono state prese e ai diversi tentativi

fatti. Vengono forniti anche tutti i risultati degli addestramenti, in termini di MSE e i loro tempi, che sono però una misura indicativa in quanto dipendono dalla potenza dell'architettura che li esegue.

7.1 Modelli statistici

7.1.1 Regressione Lineare

Per la regressione lineare vengono utilizzati i dataset df train prep Livello Acqua e df train prep Portata Uscita, contenenti gli attributi normalizzati e il target traslato sette giorni in avanti.

Si comincia prevedendo il Livello Acqua. Come attributi si selezionano la Pioggia Totale, la Temperatura Zona 5 e la Portata Uscita. Si applica la cross validation per serie storiche definita in sklearn: vengono effettuati 60 split lungo la serie dove il test set ha lunghezza 50 osservazioni. Dopo aver definito una regressione di tipo *Ridge*, viene effettuato un fit sui valori di addestramento e successivamente si predice su quelli di validazione. I risultati delle diverse iterazioni vengono salvati in un array, dal quale poi si ricava la media per ottenere il valore di errore medio finale. Successivamente il modello viene allenato su tutti i dati di training e salvato come 'Ridge Livello Acqua'. Per la Portata il procedimento è identico ma il Livello diventa un attributo e la Regressione è priva della norma tipica del Ridge. Dopo la Cross validation, il modello è allenato su tutti i dati di train e denominato.

I risultati di questo modello baseline molto semplice verranno utilizzati come metro di paragone per i successivi, e sono mostrati in tabella 1.

	Time (ms)	MSE
Portata	206	17.292
Livello	270	5.172

Tabella 1: Risultati Regressione Lineare

Prima di giungere ai risultati finali descritti sopra, sono state testate diverse configurazioni, i cui risultati sono esposti nell'Appendice. Per ogni target, è stata testata sia la regressione lineare sia la Ridge applicandole ad entrambi i dataset, cioè sia quello con gli attributi normalizzati sia quello con le feature non normalizzate.

7.1.2 SARIMAX

Per i modelli SARIMAX vengono utilizzati i dataset df train prep Livello Acqua e df train prep Portata Uscita. Dopo aver separato le feature dal target si applica una regressione lineare (minimi quadrati) al modello per capire quanto gli attributi siano significativi. Dopo aver deciso quali attributi mantenere e quali eliminare, si osservano i residui della regressione lineare. Infine, si testano i vari modelli Sarimax e si valutano osservando i residui e calcolando indici come AIC e BIC. Il modello finale deve restituire errori white noise, quindi i grafici di autocorrelazione (acf) e autocorrelazione parziale (pacf) devono essere all'interno delle bande di confidenza centrate in zero per ogni indice diverso da zero. Gli indici AIC e BIC, invece, essendo proporzionali all'opposto della funzione di massima verosimiglianza, devono essere minimizzati. Una volta individuato il miglior modello sui dati di train, si applica una cross validation definita appositamente per questo problema: ad ogni passo si considera un dataset di train di lunghezza variabile, che inizia con il primo dato disponibile, e un dataset di validation che corrisponde a sette osservazioni successive. Ad ogni passo si allena il modello sui dati di train, si effettua la previsione a sette giorni e infine si calcola l'errore tra l'ultima osservazione del validation e l'ultimo dato predetto. Questo perché l'obiettivo del progetto consiste nel prevedere le colonne target a sette giorni di distanza e solo su questo devono basarsi le valutazioni dei modelli.

Livello Acqua

Dal dataset df train prep Livello Acqua vengono isolati gli attributi Pioggia Totale, Temperatura, Portata Uscita, Time e Time square, Intercetta, le quattro sinusoidi e il logaritmo del target, scelto per eliminare la stazionarietà in varianza della serie. Una volta calcolate le previsioni, per ottenere il valore nell'ordine di grandezza iniziale, si applica la funzione inversa, e quindi l'esponenziale. Dal punto di vista statistico questa

operazione viene legittimata come segue: in teoria, bisognerebbe moltiplicare l'esponenziale della previsione con l'esponenziale della varianza del modello. La varianza però non è nota, quindi andrebbe stimata. Aggiungere alla previsione un altro elemento di incertezza non migliora i risultati del modello; si considera quindi solo l'esponenziale della previsione, mostrando che questo è il miglior stimatore, non rispetto all'errore quadratico medio (perché quello richiederebbe la varianza), ma rispetto all'errore medio in valore assoluto. I risultati di questo primo tentativo sono mostrati in figura 18: si notano dei picchi nella previsione in corrispondenza dei valori più alti della Portata. Valutando anche il valore del coefficiente per la Portata, pari a 0.02, è stato deciso di eliminarla dagli attributi per evitare un eccesso di errore.

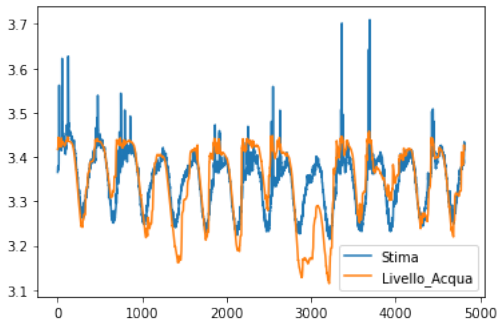

Figura 18: Stima Livello Acqua con Portata

Eliminandola e ripetendo il procedimento descritto sopra, si giunge ai risultati in figura 19, nettamente migliori. Tutti gli attributi risultano significativi poiché assumono un p-value pari a zero, anche se le due componenti del trend hanno un coefficiente pari dell'ordine di grandezza inferiore a 10^{-5}, a significare il loro minimo contributo; per questo motivo è stato deciso di non introdurre un trend nel modello.

Osservando la serie dei residui in figura 20, così come le loro Acf e Pacf (21 e 22) si nota come il valore dell'autocorrelazione decresca con velocità geometrica verso zero. Inoltre solo i primi tre valori dell'autocorrelazione parziale sono significativamente diversi da zero. Si decide di applicare anche una differenziazione semplice per risolvere il problema della non stazionarietà in media definito nelle sezioni precedenti. Quindi

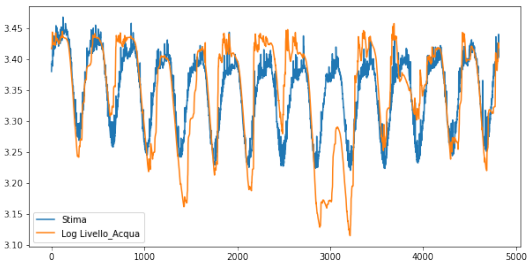

Figura 19: Serie reale e previsione Livello

la componente ARIMA del modello risulta essere $(3, 1, 1)$.

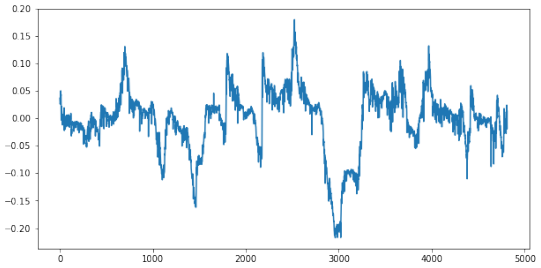

Figura 20: Residui Arima Livello

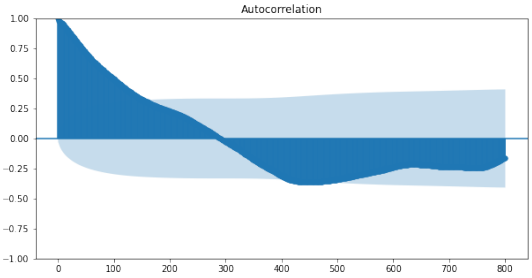

Figura 21: Acf Livello OLS

Il modello Sarimax è creato scegliendo come attributi Pioggia, Temperatura e le quattro sinusoidi. Lo si addestra scegliendo il metodo Nelder-Mead, indicato *nm*, e si pone un limite di 3000 iterazioni. Viene scelto questo metodo del simplesso, euristico per funzioni non lineari, perché l'opzione di default, che sarebbe *bfgs*, non porta a convergenza i risultati; questa scelta potrebbe essere problematica in alcuni casi in quanto non è assicurato che il minimo individuato sia quello globale. Il numero di iterazioni massimo di default viene superato e quindi è stato posto un limite più alto, che non viene mai raggiunto.

Osservando i grafici in figura 23 e 24 che mostrano Acf e Pacf dei residui del modello

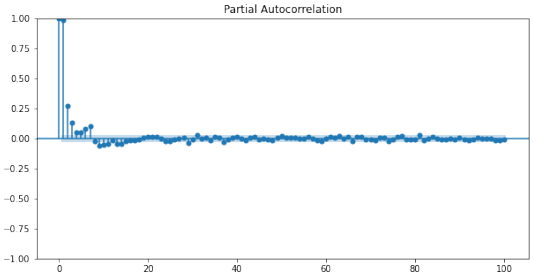

Figura 22: Pacf Livello OLS

SARIMAX, si nota come seguano un modello white noise, in quanto solo il valore al passo zero ha un'autocorrelazione non nulla.

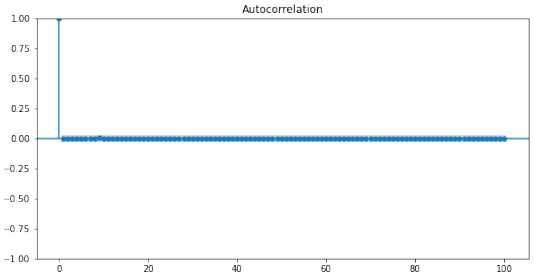

Figura 23: Acf Livello Sarimx

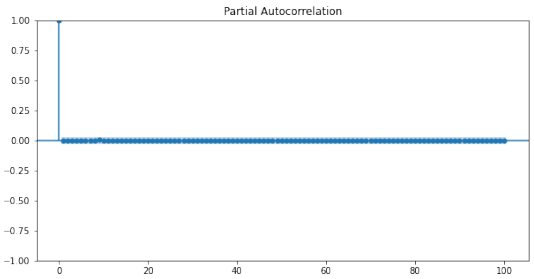

Figura 24: Pacf Livello Sarima

I risultati della cross validation applicata a questo modello sono mostrati in tabella 2. Il valore di MSE è molto basso, mostrando quindi l'ottimo comportamento dei Modelli Sarimax. Valori così bassi di AIC e BIC, che vanno idealmente minimizzati, indicano anche loro un buon adattamento.

Per giungere a questa configurazione sono stati provati diversi tentativi.

Portata Uscita

L'implementazione del codice della Portata risulta molto simile a quello precedentemente descritto per il Livello, con però alcune differenze. Dal dataset df train prep

T (s)	MSE (cv)	AIC	BIC
1.56	0.173	-41806.40	-41741.6

Tabella 2: Risultati Sarimax Livello

Portata Uscita vengono isolati gli attributi Pioggia Totale, Temperatura, Livello Acqua e Livello Acqua al quadrato, Time, Intercetta, le quattro sinusoidi e la variabile target, cioè il logaritmo della Portata. A differenza della parte precedente, in questo caso uno dei due target, il Livello, viene usato per prevedere l'altro, entrando a far parte degli attributi. I risultati di questo primo modello OLS sono presenti in figura 25. Le feature risultano tutte significative con valori del p-value pari a zero.

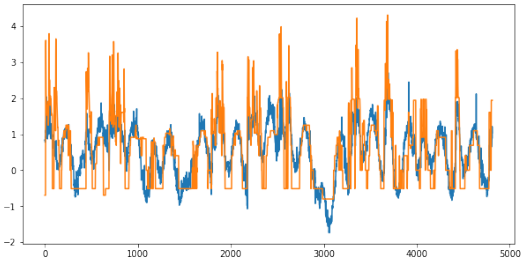

Figura 25: Serie reale e previsione Portata

Vengono mostrati i residui e i grafici di Acf e Pacf (figure 26, 27, e 28). Osservando l'Acf si nota come questo decresca geometricamente fino a zero e dal Pacf si rileva un solo valore significativamente diverso da zero.

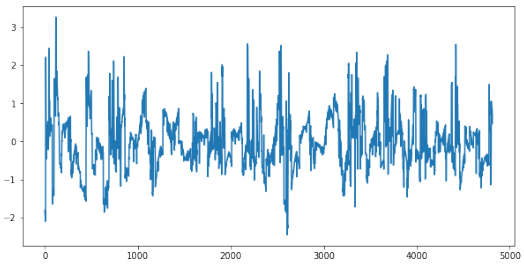

Figura 26: Residui Arima Portata

Il modello Sarimax creato è di ordine $(1, 1, 1)$ e tutti gli attributi sono quelli evidenziati in precedenza tranne l'Intercetta, che viene rimossa. Anche in questo caso si utilizza il metodo *nm* che porta a convergenza dei risultati, con numero massimo di

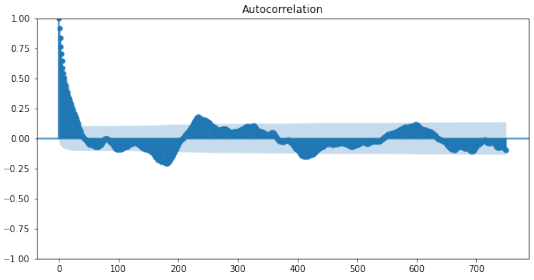

Figura 27: Acf Portata OLS

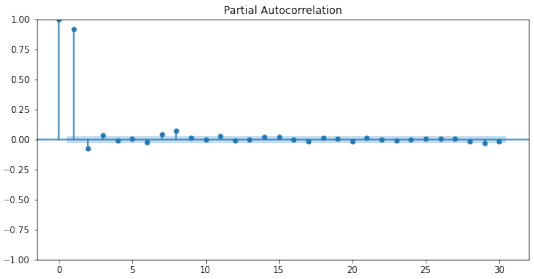

Figura 28: Pacf Portata OLS

iterazioni di 11000, che fortunatamente non viene raggiunto.

Come per il Livello si nota dalle figure 29 e 30 che le autocorrelazioni, anche a molti lag, non si discostano significativamente dal valore zero, a indicare una quasi perfetta normalità dei residui.

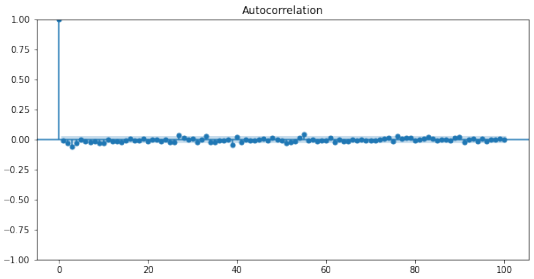

Figura 29: Acf Portata Sarimax

Anche dai risultati numerici dell'addestramento mostrati nella tabella 3 si nota un MSE basso e degli indici AIC e BIC di molto inferiori allo zero. Questo significa che l'addestramento ha avuto un esito molto buono, anche se, quasi naturalmente, non quanto quello del Livello.

Per svolgere i vari tentativi e trovare la configurazione migliore è stata utilizzata una funzione chiamata 'tentativi regressione', che esegue degli ARIMA, differenziati per i target, che hanno come input il numero di sinusoidi richieste e il numero di potenze. L'utilizzo della funzione evita l'inutile ripetizione di linee di codice sostanzialmente identiche tra loro. Per il Livello vengono testate due, quattro e otto sinusoidi avendo come attributo anche la Portata. Viene scelta la configurazione a due sinusoidi per fare ulteriori tentativi con diversi indici, cioè $(3, 1, 0)$, anche con i dati non normalizzati. Per la Portata invece vengono testati modelli con dati normalizzati e non e diversi indici.

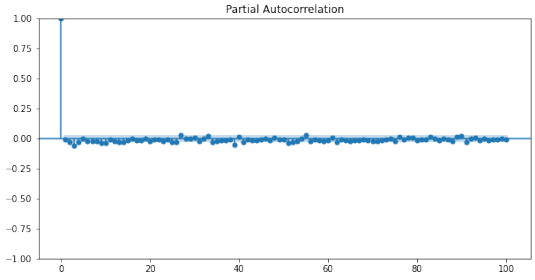

Figura 30: Pacf Portata Sarimax

T (s)	MSE (cv)	AIC	BIC
28.6	2.67	-173.55	-95.81

Tabella 3: Risultati Sarimax Portata

7.1.3 Arch

Per iniziare l'implementazione dei modelli Arch è utile porre l'attenzione sui residui del modello SARIMAX appena sviluppato per la previsione della Portata. Osservando la figura 31 si nota come siano presenti dei picchi in corrispondenza di quelli originali della Portata; questo significa che il modello, in quei casi, non riesce a prevedere in modo accurato. È importante notare che a differenza di tutti gli altri, questo modello è stato sviluppato solamente per la Portata, in quanto, proprio a causa della sua natura, è utile per serie con alta varianza, che il Livello non presenta. Inoltre, l'output dell'algoritmo non sarà un valore di MSE ma delle bande di intervalli di confidenza all'interno delle quali ci si aspetta che il valore effettivo cadrà.

Dato che non verrà valutato l'errore del modello come nei casi precedenti, non è necessario applicare una cross validation; il modello viene quindi diviso solamente tra train e test. Consapevoli del fatto che que-

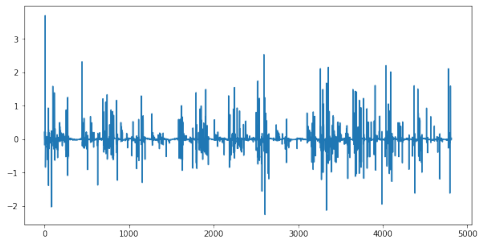

Figura 31: Residui modello arma Portata

sto modello non sarà utilizzato nel Testing, si decide di chiamare il dataset su cui viene provato test e non validation. Si considera la sola serie Portata Uscita, che quindi va isolata dal precedente dataset, in cui però è presente il suo logaritmo; il primo e fondamentale passo è applicare l'esponenziale per tornare ai valori iniziali. Questo è necessario in quanto l'interesse risiede proprio nella presenza di picchi e l'utilizzo del logaritmo, che appiattisce tutti i valori, sarebbe inadeguato. Viene addestrato un Sarimax $(1, 1, 1)$, con metodo *nm* e 1000 iterazioni massime. Ne si calcolano le previsioni che poi si inseriscono, insieme agli intervalli di confidenza al 95%, all'interno di un nuovo dataset arma predictions df. Viene creato un ulteriore dataset, questa volta formato da due colonne, cioè l'esponenziale ricreato prima e i residui tra valori reali e predetti. A questo punto è possibile definire il modello GARCH, che necessita dei residui appena creati come input, oltre a due parametri, p e q, che assumono entrambi valore 10; questi indicano rispettivamente il numero di lag della Y e il numero di lag della varianza. Viene effettuato il fit del modello scegliendo 5 come frequenza di aggiornamento. Grazie ai risultati del forecast è possibile calcolare i nuovi intervalli di confidenza, necessari per il seguente plot e per il risultato stesso dell'algoritmo.

In figura 32 si osserva la forma della banda di confidenza della Portata: nonostante i valori vadano molto oltre quelli reali, è interessante notare come i picchi delle due corrispondano, a significare che effettivamente questo modello, nato per l'utilizzo in contesti economici di alta variabilità, riesce a identificare le sezioni con cambiamenti repentini. È necessario precisare che è presente solo la banda superiore perché la Portata non può assumere valori minori di zero, quindi quella inferiore sarebbe superflua. Questo algoritmo potrebbe rivelarsi utile all'azienda che, anche se con poca precisione a livello di quantità, potrebbe permettere di capire quando aspettarsi dei giorni con Portata più alta del normale.

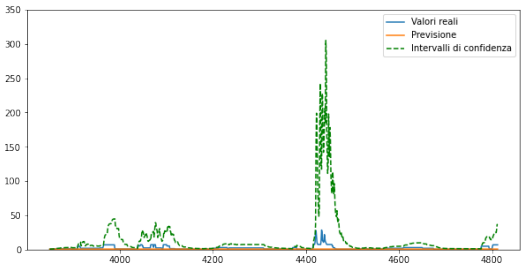

Figura 32: Intervallo di confidenza Arch

7.1.4 VAR

Per questo algoritmo viene utilizzato il dataset df train multivariato, che contiene attributi e feature non manipolate; vengono eliminate le colonne Data e Year perché non necessarie. Si definisce una funzione per testare la casualità di Granger: questa permette di ottenere una matrice in cui i valori sono p-value, che se fossero più bassi della soglia di 0.05, implicherebbero che l'ipotesi nulla, cioè che X (riga) non causa Y (colonna), può essere rigettata. Questo è utile in quanto permette di scoprire se il passato di una serie influenzi il futuro di un'altra. Alla funzione vanno forniti dataset e le colonne, e l'output è la matrice in tabella 4

	Piog_x	Temp_x	Liv_x	Port_x
Piog_y	1.000	0.0	0.026	0.025
Temp_y	0.0005	1.0	0.000	0.002
Liv_y	0.000	0.0	1.000	0.000
Port_y	0.000	0.0	0.000	1.000

Tabella 4: Granger Casuality

Tutti i valori in tabella, eccetto gli elementi sulla diagonale in cui ogni valore influenza se stesso, sono sotto la soglia, quindi, come detto precedentemente, l'ipotesi nulla può essere rigettata. Per procedere è necessario scegliere il valore dell'ordine tramite la funzione VAR; dalla tabella 5 si nota che il valore migliore è 7 perché, nonostante non

	AIC	BIC	FPE	HQIC
0	15.39	15.39	4.813e+06	15.39
1	4.872	4.899	130.6	4.882
2	4.691	4.740	109.0	4.708
3	4.599	4.669*	99.41	4.624
4	4.579	4.670	97.39	4.611
5	4.569	4.683	96.49	4.609*
6	4.563	4.697	95.83	4.610
7	4.558	4.715	95.43	4.613
8	4.552	4.730	94.84	4.615
9	4.551	4.750	94.69	4.621
10	4.538*	4.759	93.54*	4.616

Tabella 5: Selezione ordine VAR

assuma il valore minimo per nessun indice (contrassegnati con l'asterisco), è quello generalmente più basso per tutti.

A questo punto si può iniziare con il vero e proprio addestramento del modello applicando la cross-validation, come effettuato per i modelli ARIMA, per 3000 ripetizioni. Vengono eseguito il fit dei modelli e salvati i risultati della previsione a sette giorni.

	T (s)	MSE
Portata	35.6	10.602
Livello	35.6	0.126

Tabella 6: Risultati VAR

I risultati, sia per Livello che per Portata, sono mostrati in tabella 6. Questo algoritmo si dimostra molto performante per la previsione del Livello, che assume il valore minimo di MSE tra tutti i modelli tentati; per quanto riguarda la Portata, invece, la previsione non è così accurata: la presenza di picchi altissimi in una serie che assume quasi costantemente valori vicini a zero si è rivelata una grossa problematica da risolvere, come sarà anche per i modelli che verranno esposti in seguito.

7.2 Modelli machine learning

7.2.1 Symbolic Regression

Per questo tipo di algoritmo vengono utilizzati i dataset con feature normalizzate e target laggati di sette giorni, uno per il Livello e uno per la Portata. Come in altri casi il dataset viene ulteriormente splittato in due, per ottenere train e validation, con proporzione 80/20, e senza mescolamento dei dati.

A questo punto è necessario definire tutti gli iper-parametri che andranno a caratterizzare l'addestramento del modello. Molto importanti sono le funzioni matematiche che verranno utilizzate nella ricerca della formula che meglio approssima la serie; in questo caso sono state scelte somma, differenza, prodotto, quoziente, seno e logaritmo per la previsione del Livello e somma, differenza, prodotto, quoziente, seno e una funzione personalizzata per la Portata. La funzione in questione opera come base per l'addestramento del modello e se scelta con cura può migliorarne significativamente le performance; in questo caso è stato scelto di partire con la funzione $y = x + x^2$.

Ulteriori parametri sono i seguenti; Livello: *population size=500, tournament size=50, generations=75, stopping criteria=0.1, metric='mse', function set=function set, p crossover=0.65, p subtree mutation=0.15, p hoist mutation=0.05, p point mutation=0.1, verbose=0, random state=RANDOM STATE, n jobs=-1*; Portata: *population size=1000, tournament size=50, generations=200, stopping criteria=0.1, metric='mse', function set=function set, p crossover=0.65, p subtree mutation=0.15, p hoist mutation=0.05, p point mutation=0.1, verbose=0, random state=RANDOM STATE, n jobs=-1*.

Viene eseguito il classico fit-predict dei modelli e vengono plottati i grafici che rappresentano la serie vera e quella predetta. Un ulteriore test è svolto applicando una cross-validation per valutare la robustezza degli algoritmi e dei risultati, che sono mostrati in tabella 7. Per effettuare questa validazione sono stati leggermente diminuiti dei parametri, in particolare quelli di *population size* e *generations* in modo tale da non rendere la procedura troppo computazionalmente onerosa.

	T (min)	MSE	MSE (CV)
Portata	14,51	5.566	5.505
Livello	10,54	2.916	1.454

Tabella 7: Risultati Symbolic Regression

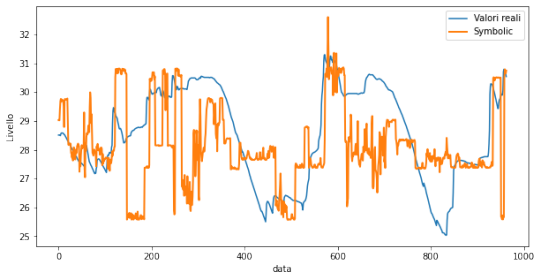

Figura 33: Livello reale e predetto da Symbolic

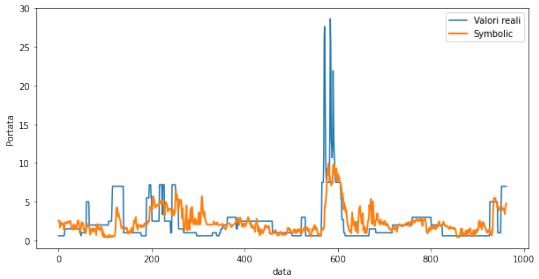

Figura 34: Portata reale e predetta da Symbolic

Osservando invece le figure 33 e 34 che rappresentano la parte di serie di validazione e quella predetta si nota che per il livello, nonostante il buon valore di MSE, la serie predetta non sia particolarmente simile a quella reale e presenti delle sorte di gradini orizzontali seguiti da picchi o avvallamenti verticali. Per quanto riguarda la Portata invece si osserva che i picchi vengono sostanzialmente tutti individuati, anche se l'incidenza non è molto precisa; il fatto che l'algoritmo riesca a individuare queste zone molto poco frequenti e molto diverse dal resto della serie indica che come metodo potrebbe avere dei buoni sviluppi. Anche il valore di MSE è tra i più bassi riscontrati tra i vari modelli per la Portata.

I vari tentativi effettuati riguardano tutti diverse configurazioni dei parametri sopra elencati: si è provato aggiungendo e togliendo le operazioni matematiche e anche fornendone di nuove personalizzate. I principali iper-parametri su cui si è lavorato sono stati quelli riguardo le generazioni e la dimensione delle popolazioni. Anche aumentando questi parametri non si è notato però un miglioramento significativo, mentre invece aumentava molto il tempo di addestramento. Il trade-off non era affatto conveniente.

7.2.2 K-nearest neighbors

Per sviluppare l'algoritmo KNN è necessario l'utilizzo del dataset originale, una volta isolando Data e Portata e una Data e Livello. Questo perché nella ricerca delle sotto-sequenze più simili è fondamentale l'uso della serie nella sua totalità. Prima di poter procedere con la ricerca vanno definiti alcuni iper-parametri: p indica la lunghezza della sotto-sequenza simile da cercare, k è il numero di sequenze da individuare, h determina di quanto avanti si vorrà fare la previsione, in questo caso sette giorni, $N\ rip$ è il numero di ripetizioni per cross-validation, *step* indica di quanto andare avanti per lo slittamento del train e *pesi* può essere media o mediana per la stima futura. Per la previsione del Livello gli iper-parametri scelti sono i seguenti: *p=365*3, k=5, h=7, N rip=2000, step=1, pesi='media'*; mentre per la Portata sono *p=365*2, k=5, h=7, N rip=1000, step=1, pesi='media'*.

A questo punto si può procedere con l'addestramento del modello: scelta la sotto-sequenza target, inizia la ricerca delle k più simili e se ne salvano i valori futuri, per poi farne la media o la mediana, producendo così la previsione.

I risultati dell'addestramento sono mostrati in tabella 8. Da questi valori si nota come la serie Portata abbia ancora causato più difficoltà nell'apprendimento: probabilmente nel caso in cui fosse stata scelta come sotto-sequenza query una sezione contenente uno dei picchi, ci sarebbero potuti essere diversi problemi nella ricerca di quelle simili. Per il Livello, invece, il buon valore di MSE è verosimilmente causa del fatto che la serie è molto periodica e regolare, perciò la ricerca è stata sicuramente meno problematica. Il diverso tempo è dato dal differente numero di generazioni per cui sono stati eseguiti gli algoritmi; anche aumentando quelle per la Portata non si sono riscontrati miglioramenti.

Per raggiungere la configurazione di iper-parametri descritta si sono svolti diversi tentativi valutando diverse quantità di sotto-sequenze da cercare, diverse lunghezze e di-

	T (min)	MSE
Portata	0,422	9.785
Livello	1,37	1.102

Tabella 8: Risultati KNN

versi step. In generale un maggior numero di ripetizioni ha portato a risultati migliori, ma il fattore più importante è risultato essere il numero di sotto-sequenze: troppe poche e la previsione mancherà di robustezza; troppe e ne verranno considerate anche di non particolarmente simili, sporcando così la previsione.

7.3 Reti neurali

Per quanto riguarda il training della rete neurale, la prima importante fase è l'impostazione di alcuni *random seed*, in modo tale da permettere di ottenere ripetibilità nei risultati. Questi riguardano la generazione di numeri random di *python*, *numpy* e *tensorflow*; inoltre viene istanziata una sessione di *tensorflow* che permetta di ottenere valori riproducibili. Per l'addestramento della rete vengono utilizzati i dataset df train reti e df train prep reti, creati appositamente nella fase di pre-processing. Vengono utilizzati i dati originali per predire la Portata, mentre quelli normalizzati per il Livello, in quanto nello sviluppo di vari tentativi questa configurazione ha portato i migliori risultati. I dataset vengono ulteriormente divisi in train e validation con una proporzione 80/20, in modo da poter valutare il risultato dell'addestramento; da entrambi vengono isolati i dati riguardanti le feature e quelli riguardanti il target. Ai dataset delle feature viene aggiunta una dimensione in modo tale che possano essere adatti all'utilizzo nelle reti.

Come detto precedentemente, si è deciso di creare due reti: una per prevedere il valore di Portata e una per il Livello. Entrambe le reti hanno la stessa struttura, che perciò verrà esposta solamente una volta; quello che differisce nella loro implementazione sono sostanzialmente solo le parti riguardanti la scelta della colonna target e delle feature. La rete è strutturata in questo modo: dopo tre *layer* LSTM è presente un GRU, seguito da un Dense per poter ottenere il risultato della regressione. È anche stato fatto un tentativo con una rete a doppia uscita, con struttura simile alla precedente, ma che presenta uno split dopo l'ultimo strato LSTM, che permette di ottenere contemporaneamente la previsione per entrambi i target. Creando due reti separate è possibile utilizzare una colonna target come attributo per prevedere l'altro, perciò nel calcolo del livello le feature saranno Pioggia Totale, Temperatura Zona 5 e Portata, mentre per la Portata saranno Pioggia Totale, Temperatura Zona 5 e Livello.

Tutti gli strati richiedono degli iperparametri particolari per poter funzionare, in particolare *activation="tanh", recurrent activation="sigmoid", recurrent dropout=0, unroll=False, use bias=True.* A questi vengono aggiunti anche degli iper-parametri per l'inizializzazione dei kernel, delle ricorrenze e dei bias, importanti per la riproducibilità. I *layer* hanno numero di unità decrescente, cioè 100 per il primo LSTM, 50 per il secondo, 30 per il terzo e 10 per il GRU. Lo strato Dense utilizza un tipo di attivazione *relu*. Il modello adopera un ottimizzatore *adam* e come metrica di valutazione della perdita il MSE. Viene anche creata una funzione di *early-stopping*: questa monitora la progressione delle epoche di addestramento, in modo tale che queste presentino sempre una diminuzione nella funzione di perdita; nel caso si dovesse notare un aumento, questo *call-back* ferma l'addestramento prima che abbia raggiunto il numero totale di epoche scelto. Il numero di parametri addestrabili della rete finale si aggira intorno agli 80000, con delle piccole differenze nel caso di quelle a uscita singola e quella a doppia uscita. La rete viene addestrata per un massimo di 50 epoche, con eventuale stop prima del raggiungimento, e viene testata sul dataset di validazione; importante anche in questo caso che non si rimescolino i dati nel set di training. Per concludere vengono mostrati gli errori per le variabili target e dei grafici che permettono di osservare il cambiamento del valore di loss per training e validation con il passare delle epoche.

In seguito (tabella 9) vengono mostrati i risultati delle due reti create e nelle figure 35

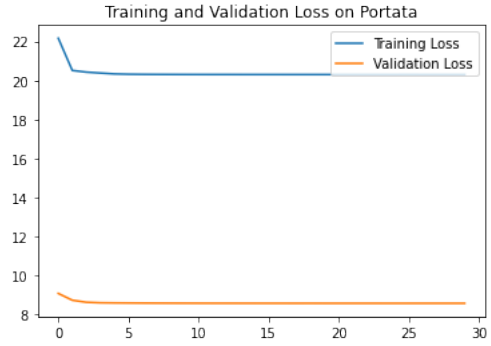

Figura 35: Loss per Portata

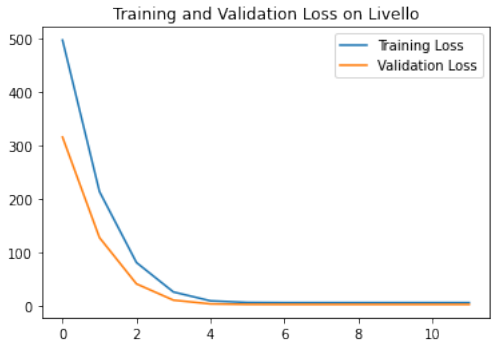

Figura 36: Loss per Livello

e 36 l'andamento della perdita con il proseguire delle epoche per Portata e Livello. Si osserva come la loss per la Portata rimanga sostanzialmente invariata per tutte le epoche di addestramento, a dimostrazione del fatto che il modello non sia riuscito ad adattarsi bene; al contrario, la serie Livello sembra essere stata appresa con miglior successo.

	Epoche	T (sec)	MSE
Portata	18	42,2	8.568
Livello	12	30,8	2.765

Tabella 9: Risultati reti neurali

Come spiegato in precedenza, è stata tentata anche una rete a doppia uscita, che però non permetteva di utilizzare il Livello come attributo per prevedere la Portata, in quanto anch'esso era un target. Si è notato che questo peggiorava leggermente i risultati. Inoltre, la scelta di creare due reti separate è sembrata più coerente, perché nel caso fossero state scelte come migliori per la parte di Testing, non si avrebbe avuto il vincolo di prevedere inutilmente anche il target non scelto. Per quanto riguarda le reti sono stati testate diverse configurazioni di *layer*, cambiando il numero di unità e aggiungendone di nuovi, sopratutto per la Portata che dava risultati peggiori. Questi test hanno portato i tempi di addestramento ad allungarsi drasticamente, ma non a miglioramenti degli errori. Per questo motivo sono state create due reti identiche e piuttosto semplici.

È corretto far notare che, nonostante l'impostazione di tutti i parametri, purtroppo non si è riusciti a ottenere una configurazione che garantisse risultati sempre perfettamente riproducibili: la terza cifra decimale dei valori di MSE spesso cambia a ogni addestramento, così come il numero di epoche di addestramento della Portata; il Livello sembra più stabile, sia a livello di errore che epoche. Detto questo, i risultati restano comunque indicativamente sempre gli stessi, e variano per centesimi o millesimi di errore.

8 Testing

Questa fase viene sviluppata per misurare le performance dei migliori modelli testandoli su dati inediti, sui quali non si sono addestrati. Vengono scelti i due algoritmi meglio performanti dalla fase di addestramento, uno per target; i modelli identificati sono i SARIMAX, sia per il forecast della Portata che per quello del Livello. La scelta è stata presa, oltre che per i bassi valori di errore, anche in ragione del fatto che questi algoritmi siano sviluppati appositamente per serie storiche, e sono sembrati essere i più adatti a questo caso particolare.

I dati utilizzati in questa fase sono contenuti all'interno dei file df test prep Portata Uscita e df test prep Livello Acqua creati alla fine del pre-processing con l'idea di poter fornire agli algoritmi delle nuove osservazioni. Il fatto che i dati siano comunque etichettati permette di valutare gli errori di previsione e notare se ci siano differenze più o meno evidenti tra i risultati in fase di training e i nuovi. Una sostanziale differenza tra i due potrebbe significare che il modello ha avuto un problema di over-fitting, caso in cui l'algoritmo si abitua troppo ai dati su cui si addestra, e non riesce a generalizzare le previsioni su dati nuovi.

Dopo avere scelto le colonne necessarie vengono caricati i modelli creati durante il training, cioè arima Livello Acqua e arima Portata Uscita. Per eseguire il primo test vengono isolati i primi sette valori del dataset escludendo la colonna target; su questi viene eseguita la previsione, della quale va considerata solo il settimo valore, corrispondente al valore previsto una settimana dopo. Per avere dei valori effettivi è anche necessario applicare l'esponenziale ai risultati, perché i modelli sono stati creati prevedendo il logaritmo dei target. Ripetendo queste operazioni e scegliendo valori slot di valori, si possono ottenere diverse previsioni per poi eventualmente calcolare un errore medio di previsione. Il procedimento appena esposto viene eseguito in maniera analoga sia per il Livello che per la Portata.

I risultati di questa fase sono mostrati in tabella 10

	T (min)	MSE
Portata	0,422	9.785
Livello	1,37	1.102

Tabella 10: Risultati testing

9 Deploy

Questa sezione finale viene creata per poter simulare una situazione reale, in cui vengono registrati i dati provenienti dai vari sensori e inseriti in un dataset; tramite queste misurazioni si devono prevedere i valori a sette giorni di Portata e Livello. In un problema di classificazione basterebbe un nuovo dataset di una riga contenente tutti gli attributi utilizzati nella creazione dei modelli; tuttavia, la previsione delle serie storiche richiede una configurazione di nuovi dati diversa. Dato che i modelli scelti sono stati entrambi SARIMAX, che hanno bisogno del passato per prevedere il futuro, è necessario fornire qualche dato storico per completare al meglio questa fase.

In particolare, viene fornito un nuovo dataset detto 'deploy finto' contenente nove nuove righe di dati inventati creati da noi, seguendo come ispirazione gli ultimi valori effettivi del dataset originale. Questi servono esclusivamente per poter eseguire correttamente il codice e, in una situazione reale, sarebbero sostituiti dalla vera serie misurata giorno per giorno; per fare ciò andrebbe anche cambiato il nome e naturalmente anche il percorso del file.

È importante stare attenti a eseguire tutte le operazioni effettuate sui dataset di train e test, perché in caso contrario il modello non potrebbe eseguire la previsione. Proprio per questo prima di tutto è necessario creare la colonna Pioggia Totale come somma delle cinque singole zone. Inoltre, i valori degli attributi vanno normalizzati utilizzando lo *Standard Scaler* apposito salvato alla fine della fase di pre-processing; successivamente si creano le quattro sinusoidi utilizzate nel training, la colonna dei logaritmi dei target e altri eventuali attributi.

Prendendo solo i primi sette valori del dataset e applicando uno shift indietro di una settimana si ottiene una tabella nella quale il settimo valore non presenta alcun valore per la variabile target ma contiene i valori degli attributi, a simulare l'osservazione odierna con la quale si dovrà prevedere a sette giorni. Si isola la prima riga e si applica il modello. Una volta fatto ciò, utilizzando la parte di tabella rimanente che corrisponde al futuro, viene eseguito il fit e ricavato il valore predetto a sette giorni. I risultati di Portata e Livello sono salvati all'interno di due nuovi file *.csv* chiamati forecast livello e forecast portata.

10 Conclusioni e sviluppi futuri

Dopo avere sviluppato il progetto si possono trarre delle conclusioni finali: la previsione della serie Livello è stata, per tutti i tipi di modelli testati, migliore di quella della Portata; questo è evidentemente causa della sua maggiore periodicità e ripetitività, caratteristica molto favorevole per gli algoritmi. Dal lato opposto, prevedere una serie che assume nella sua quasi totalità valori prossimi allo zero, per poi essere caratterizzata da picchi altissimi che appaiono senza un'evidente causa, è molto complicato. Proprio a causa di questi picchi i modelli hanno avu-

to maggiori difficoltà, constatabili dai valori MSE delle previsioni della Portata, generalmente molto maggiori di quelli calcolati per il Livello. Un'altra considerazione va fatta riguardo il costo computazionale di alcuni algoritmi: per parte dei modelli si è dovuto cercare un equilibrio tra l'impostazione di iper-parametri come numero di ripetizioni, unità o generazioni, che assumendo valori alti avrebbero potuto portare a risultati migliori, ma che avrebbero anche reso l'addestramento eccessivamente lungo e costoso. È proprio su questo trade-off tra bontà dei risultati e onerosità computazionale che si potrebbero concentrare dei potenziali sviluppi futuri. Con una potenza di calcolo maggiore si potrebbero tranquillamente gestire addestramenti che, nel nostro caso, sarebbero risultati limitanti per uno sviluppo vario e di diverse tecniche. Per espandere ancora di più la ricerca si potrebbero testare altri modelli di previsione, che siano essi statistici, come gli UCM, o degli altri algoritmi di machine learning.

Riferimenti bibliografici

[1] Sarni W. et al. *Harnessing the Fourth Industrial*, 2018

[2] Alabi M. e Arnesh T. *Industry 4.0 and water industry: a south african perspective and readiness*, 2020

[3] https://www.arpalombardia.it/Pages/Indicatori/2018/Idrometeorologia/Numero-giorni-pioggia-lungo-periodo-2018.aspx

[4] https://www.rainews.it/articoli/2022/06/allarme-siccit-del-po-chiesto-razionamento -dellacqua-in-125-comuni-0af916a5-b7cb-4a49-ae5b-45a27bfbf924.html

Appendix

	Pioggia_Zona_1	Pioggia_Zona_2	Pioggia_Zona_3	Pioggia_Zona_4
count	6026	6026	6026	6026
mean	2.471225	3.340657	2.670063	2.674743
std	6.650012	8.903134	7.166643	7.655847
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000
75%	0.800000	1.400000	1.200000	0.600000
max	80.600000	110.000000	120.200000	113.600000

Tabella 11: Overview prime colonne

	Pioggia_Zona_5	Temperatura_Zona_5	Livello_Acqua	Portata_Uscita
count	6026	6025	6386	6386
mean	3.129871	14.530141	28.571961	2.784619
std	8.025121	6.944029	2.196148	4.098431
min	0.000000	-5.350000	22.530000	0.450000
25%	0.000000	9.000000	26.930000	0.600000
50%	0.000000	14.500000	29.265000	1.500000
75%	1.200000	20.100000	30.430000	3.000000
max	88.400000	34.000000	31.760000	74.650000

Tabella 12: Overview colonne rimanenti

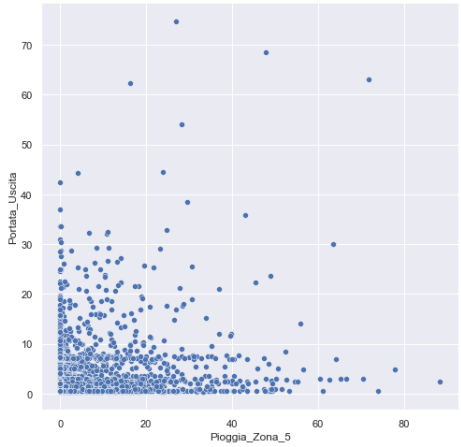

Figura 37: Pioggia zona 5 e Portata

Tentativi Regressione Lineare
Portata:

- dati non normalizzati, Lineare e Ridge: MSE: 17.292, 17.292
- dati normalizzati, Lineare: MSE: 17.292

Livello:

- dati non normalizzati, Lineare e Ridge: MSE: 5.172, 5.172
- dati normalizzati, Lineare: MSE: 5.172

Tentativi Arima

Livello, dati non normalizzati: AIC = -41806.298, BIC -41741.510

Portata, dati non normalizzati: AIC -173.558, BIC -95.811

Tentativi Symbolic Regression

Livello:

- function set = ['add', 'sub', 'mul', 'div', 'sin', 'log'] population size=1000, tournament size=50, generations=200, stopping criteria=0.1, metric='mse', function set=function set, p crossover=0.65, p subtree mutation=0.15, p hoist mutation=0.05, p point mutation=0.1, verbose=0, random state=RANDOM STATE, n jobs=-1 MSE: 2.916

Portata:

- a = np.sin(np.log(np.abs(x)+1)); portata log trend = gpl.functions.make function(function = portata log trend, name='trend', arity=1) function set = ['add', 'sub', 'mul', 'div', portata log trend] population size=1000, tournament size=50, generations=20, stopping criteria=0.1, metric='mse', function set=function set, p crossover=0.65, p subtree mutation=0.15, p hoist mutation=0.05, p point mutation=0.1, verbose=0, random state=RANDOM STATE, n jobs=-1 MSE: 5.772

- function set = ['add', 'sub', 'mul', 'div', 'sin'] population size=1000, tournament size=50, generations=100, stopping criteria=0.1, metric='mse', function set=function

set, p crossover=0.65, p subtree mutation=0.15, p hoist mutation=0.05, p point mutation=0.1, verbose=0, random state=RANDOM STATE, n jobs=-1 MSE: 5.807

Tentativi KNN
Livello:

- p=365*2, k=5, h=7, N rip=1000, step=2, pesi='media'
  MSE: 2.522
- p=365*3, k=5, h=7, N rip=2000, step=2, pesi='media'
  MSE: 1.154
- p=365*3, k=5, h=7, N rip=2000, step=2, pesi='mediana'
  MSE: 1.219

Portata:

- p=365*3, k=5, h=7, N rip=1000, step=2, pesi='media'
  MSE: 13.527
- p=365*2, k=5, h=7, N rip=2000, step=2, pesi='media'
  MSE: 17.739
- p=365*3, k=5, h=7, N rip=2000, step=2, pesi='media'
  MSE: 18.181
- p=365*3, k=5, h=7, N rip=1000, step=2, pesi='mediana'
  MSE: 13.594

Tentativi Reti Neurali
Rete doppia uscita (stessa rete ma con split al livello dei layer GRU): MSE Portata: 9.633; MSE Livello: 2.71; addestramento in 12 epoche
Rete Portata con dati normalizzati (stessa rete): MSE: 8.566
Rete Livello con dati non normalizzati (stessa rete): MSE: 2.756

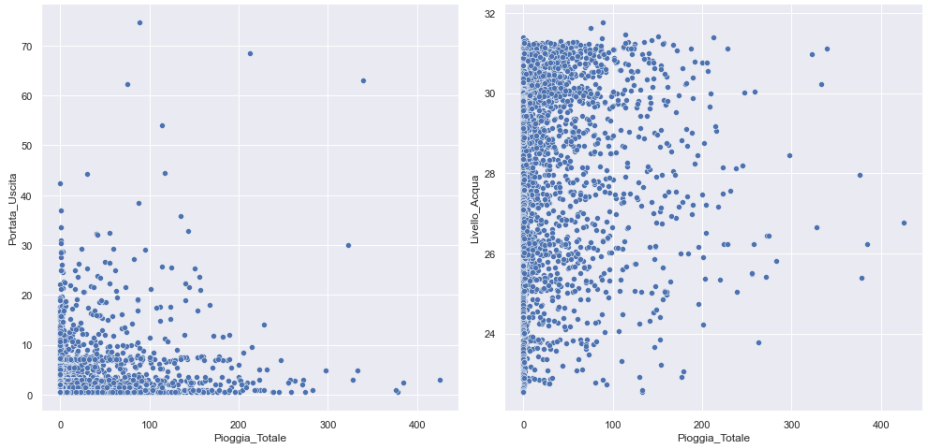

Figura 38: Pioggia Totale-Portata e Pioggia Totale-Livello

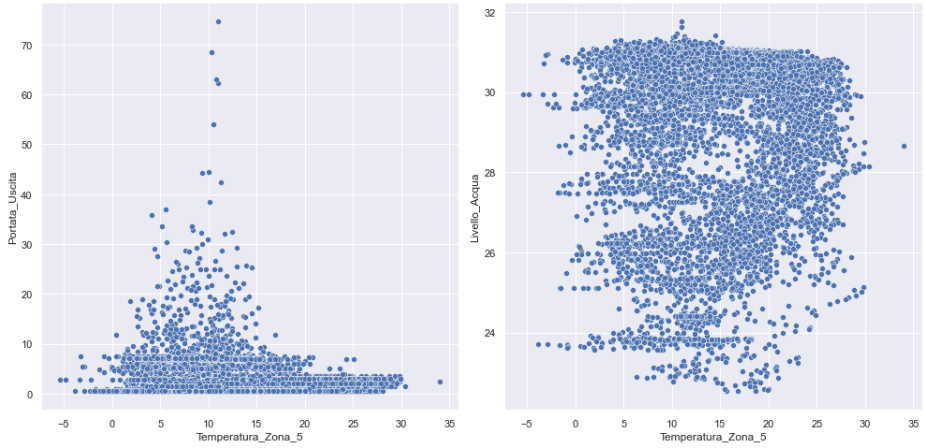

Figura 39: Temperatura zona 5-Portata e Temperatura zona 5-Livello