UNIVERSITA' DEGLI STUDI DI
NAPOLI FEDERICO II

Scuola Politecnica e delle Scienze di Base
Corso di Laurea Magistrale in Ingegneria Informatica

Elaborato del corso di **Corso di Information System & Business Intelligent**
Prof. F. Amato - A.A. 2023 – 24

STUDENTI:

DAIANA	CIPOLLARO	M63001490	da.cipollaro@studenti.unina.it
FRANCESCO	DI SERIO	M63001442	f.diserio@studenti.unina.it

Indice

Progetto Finale

Esercizio 1 – Analisi del Dataset in Py

Partendo dal Dataset fornito, è stata fatta un'analisi preliminare, che ci ha permesso di produrre il workflow (riportato di seguito):

1. Definizione degli obiettivi dell'analisi

Il dataset è composto da due file csv, il primo composto da 365 istanze e il secondo di 106, dotato di una colonna "*no. of Adult males*", di cui il primo risulta privo. Alla luce di ciò è stato deciso (e consigliato) di modellare i dati come una serie storica e affrontare il progetto con l'obiettivo di addestrare dei modelli predittivi in grado di apprendere dai dati del secondo file e successivamente fare predizioni sulla classe mancante del primo.

Data l'esigua mole di campioni a disposizione, il processo di apprendimento è risultato essere abbastanza limitato. Per tale ragione è stato deciso di dividere il training set in train, validation e test (come di consuetudine) e una volta addestrato il modello e ottenuto i parametri migliori, riaccorpare lo split di train e validation in un unico 'train' e usare questi dati per un addestramento più performante.

2. Analisi dei Dati

Si è proseguito con uno studio dettagliato dei dati, per assicurarsi che fossero adatti al progetto che si intendeva affrontare e che fossero completi, accurati e in che formato fossero state fornite le varie colonne.

Le colonne messe e disposizione hanno un campo dedicato alla locazione temporale della prelazione dei dati, ma il file scelto per il train ha tale colonna, 'Date', in formato stringa per cui è stato necessario un'operazione di preprocessing per adattare tale stringa al formato desiderato 'data'. Inoltre, mancava di un attributo: l'anno, che è stato assunto essere il '2022' in accordo ad osservazioni fatte durante le attività didattiche e poiché notato che vi fosse una corrispondenza tra le colonne presenti nel primo file con quelle del secondo (con le relative corrispondenze temporali).

La colonna di interesse invece, si presentava come un intero; quindi, il problema è stato classificato come problema di *regressione* e come tale è stato affrontato con modelli affini.

Si è inoltre notato che il dataset fosse privo di dati mancanti per cui non sono state necessarie particolari operazioni di preprocessing di missing value. Siamo quindi passati allo studio di distribuzione dei dati del training set per vedere come si distribuissero i campioni delle relative colonne durante i vari giorni; e la distribuzione relativa ai dati di train rispetto a quelli di test, per capire se, e quanto, i primi fossero rappresentativi dei secondi.

Studio Statistico

Si è proseguito con uno studio dei valori statistici per comprendere la rappresentatività dei dati di train rispetto a quelli di test e quindi farci un'idea dei risultati che avremmo potuto avere a valle dell'apprendimento.

Dai valori ottenuti, è stato possibile ottenere diverse informazioni sulle caratteristiche delle due serie di dati:

Primo dataset:

	count	mean	std	min	25%	50%	\
temperature_mean	365.0	16.038740	7.965726	1.33	9.15	15.41	
relativehumidity_mean	365.0	61.249315	15.660750	26.00	50.00	61.00	

	75%	max	Skewness	Kurtosis
temperature_mean	23.41	32.41	0.096121	-1.266360
relativehumidity_mean	72.00	94.00	-0.011114	-0.638466

- Entrambe le distribuzioni sembrano abbastanza simmetriche, dato che la skewness è vicina a zero.
- La kurtosis per entrambe le distribuzioni è leggermente più bassa del valore tipico per una distribuzione normale. Questo significa che:
    - Non ci sono particolari problemi con outlier.
    - La distribuzione ha code abbastanza leggere.
    - La distribuzione è "piatta" e meno pronunciata alle code.

  Questo è stato utile per comprendere quanto i dati fossero concentrati intorno alla media.

Secondo dataset:

	count	mean	std	min	25%	50%	\
no. of Adult males	106.0	0.415094	1.120101	0.00	0.00	0.00	
temperature_mean	106.0	25.015566	3.768792	14.03	23.70	25.64	
relativehumidity_mean	106.0	50.283019	11.928162	26.00	41.25	51.50	

	75%	max	Skewness	Kurtosis
no. of Adult males	0.0000	6.00	3.304269	11.802826
temperature_mean	27.2975	32.41	-1.023589	0.944560
relativehumidity_mean	58.0000	81.00	0.045763	-0.327317

- La serie **no. of Adult males** mostra una skewness significativa, indicando una distribuzione più *asimmetrica*. La *kurtosis è anche abbastanza elevata*, suggerendo code più pesanti rispetto a una distribuzione normale.
- La skewness di **temperature_mean** è negativa, indicando una leggera asimmetria a sinistra. La kurtosis è vicina a 1, suggerendo una leggera coda più pesante rispetto a una distribuzione normale.
- La skewness di **relativehumidity_mean** è vicina a zero, indicando simmetria. La kurtosis è leggermente più bassa del valore tipico per una distribuzione normale.

Test di Kolmogorov-Smirnov

Abbiamo poi effettuato un **test di Kolmogorov-Smirnov** per verificare la significativa somiglianza tra i due dataset, quello con e quello senza colonna target per capire quanto affidabili siano le previsioni effettuate dato che il nostro dataset è formato da pochi campioni e risulta difficile fare uno split ottimale per addestramento e valutazione dell'accuracy.

Il test KS confronta le distribuzioni cumulative empiriche delle due serie (**relativehumidity_mean** e **temperature_mean**). Il valore di KS Statistic ottenuto è abbastanza elevato e siccome indica la massima differenza tra le due distribuzioni cumulative, allora possiamo dire che le due distribuzioni sono significativamente differenti. Inoltre, il P-Value associato è molto basso in entrambi i casi, indicando che si può respingere l'ipotesi nulla che le distribuzioni siano uguali. Quindi, le distribuzioni sono statisticamente diverse.

```
 the Kolmogorov - Smirnov test between common column is
                          KS Statistic       P-Value
temperature_mean             0.554381  6.457601e-24
relativehumidity_mean        0.343448  4.000434e-09
```

Alla luce di quanto detto possiamo affermare che la rappresentatività dei dati di train è bassa rispetto a quelli di test e quindi ci aspettiamo anche risultati non ottimi.

Test di Normalità

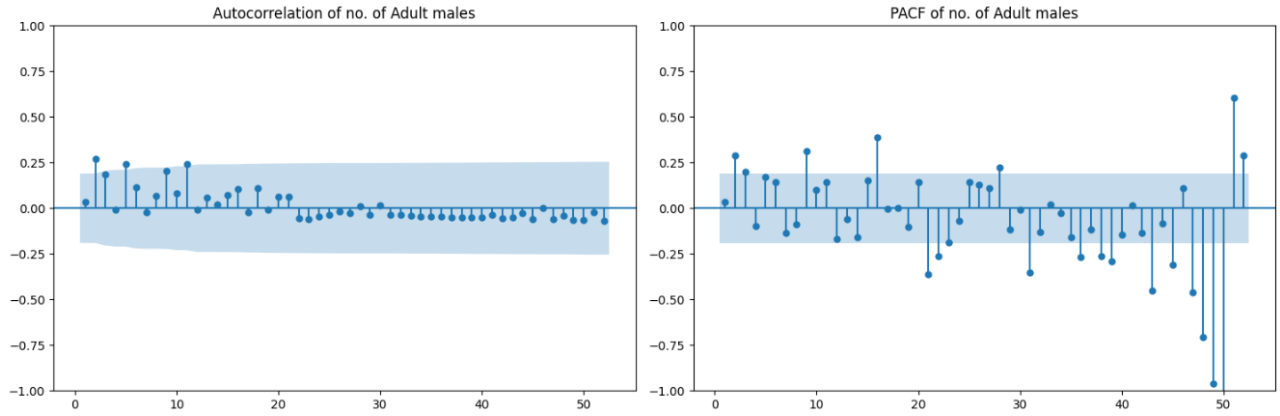
In seguito, è stato eseguito un **Test di Normalità** per prepararci al calcolo degli intervalli di confidenza o eseguire test di ipotesi sulla media, in quanto questi metodi presuppongono una distribuzione normale dei dati.

Nella sua definizione il test di normalità verifica se un campione di dati proviene da una popolazione con una distribuzione normale (o gaussiana).

Autocorrelazione

Dopodiché è stato fatto uno studio dell'**autocorrelazione** (ACF) dei dati della serie temporale per studiare le correlazioni tra le osservazioni riportate nelle istanze, valutando quanto le osservazioni attuali dipendano da quelle passate a intervalli di tempo specifici, mostrando come le osservazioni si correlino con i loro ritardi.

È stata studiata anche la funzione di **Autocorrelazione Parziale** (PACF) per misurare la correlazione tra le osservazioni attuali e quelle di ritardi intermedi, eliminando l'effetto delle correlazioni a ritardi più lunghi. In questo modo si mostra la correlazione diretta tra due osservazioni, isolando l'effetto delle osservazioni intermedie, molto utile per individuare un possibile ordine di un modello AR (AutoRegressive).



Questo studio è stato fondamentale per poter definire il n.ro di lag significativi nella serie. In particolare, si è scelto di contare i lag significativi soprattutto considerando lo studio grafico della PACF per diversi motivi:

- Quelli della PACF erano in n.ro maggiore di quelli dell'ACF → per la scelta del max n.ro di lag i secondi sarebbero già stati compresi
- In letteratura si consiglia di usare una PACF nel caso in cui la f. di autocorrelazione abbia picchi bruschi in 0 e valori troppo oscillanti ad esso e nel nostro caso la PACF era più omogenea dell'ACF.

Tipo di Andamento

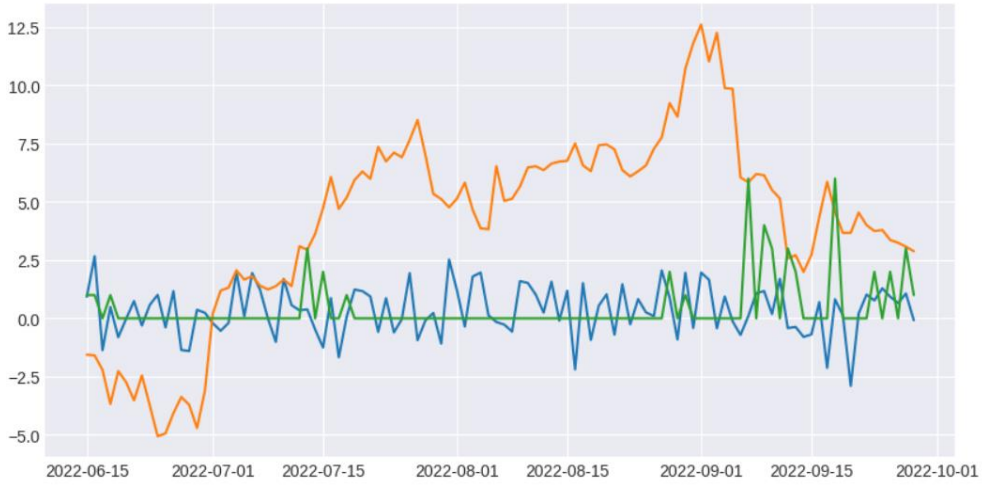
Prima di scegliere quali tipi di modelli predittivi usare è stato studiato il segnale generato dalla distribuzione della classe target nel tempo e quindi confrontato con un random walk e un white noise (generati in py). È stato condotto uno studio grafico da cui è stato dedotto:

1. **Media e Varianza Costante**: media costante tendente allo zero. Ogni osservazione è, in media, zero, e non c'è alcun modello o tendenza evidente. Inoltre, non ci sono variazioni sistematiche nella dispersione dei dati.
2. **Assenza di Correlazione Temporale**: Le osservazioni risultano scorrelate tra loro. Ciò significa che non c'è alcuna dipendenza temporale tra un'osservazione e la successiva.
3. **Casualità**: Le osservazioni sono casuali e indipendenti tra loro. Questo lo rende un processo stocastico molto semplice.

Stazionarietà

Per valutare la stazionarietà è stato fatto il test **Augmented Dickey-Fuller** (ADF) che ha prodotto un $p_{value} = 0.406 > 0.05$, quindi, rigettando l'hp. alternativa e appurando che la serie sotto esame è una serie temporale non stazionaria.

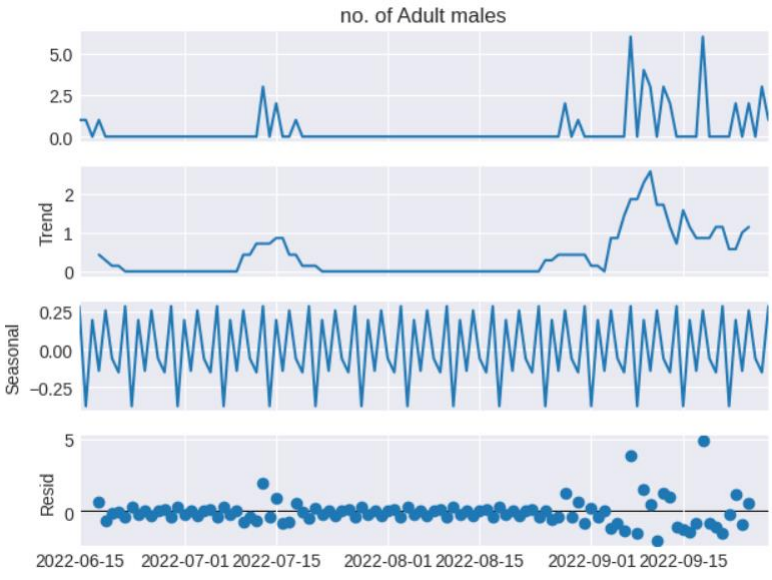
Alla luce di questi studi, anche se in mancanza di caratteristiche quali la stazionarietà, il nostro modello è molto più vicino a un white noise che non a un random walk.



Stagionalità

Lo studio è proseguito con l'analisi della **stagionalità** tramite tecniche di decomposizione additiva (la moltiplicativa non è stata possibile in quanto la classe target era composta principalmente da valori nulli).

Da questa non è emerso nessun trend caratteristico, inoltre la stagionalità risultava concentrata vicino allo zero [-0.25; 0.25], indice del fatto che la presenza di molti valori pari a zero avrebbe portato a predizioni praticamente sempre nulle e che non esisteva una vera e propria stagionalità del modello (cosa riscontrata anche successivamente con i modelli statistici).



Inoltre, i residui, sempre concentrati vicino allo 0, riportano una forte coda sulle ultime acquisizioni, indicando la criticità dell'andamento dei dati a disposizione.

3. Modelli Predittivi

Lo studio è proseguito col training sui dati. Alla luce di quanto detto sono stati analizzati i seguenti modelli:

Modelli statistici

Per poter applicare i modelli statistici è risultato necessario prima di tutto rendere la serie stazionaria, per cui è stata applicata una differenziazione del dataset considerando la differenza tra il valore attuale e il precedente, ottenendo, per ogni colonna del dataset, un $p_{value} < 0.05$.

Dopodiché è stato effettuato lo split manuale dei dati in train-val-test; inoltre, per poter conservare la sequenzialità dei dati e non perdere la dipendenza temporale non sono state adottate tecniche di random stratify.

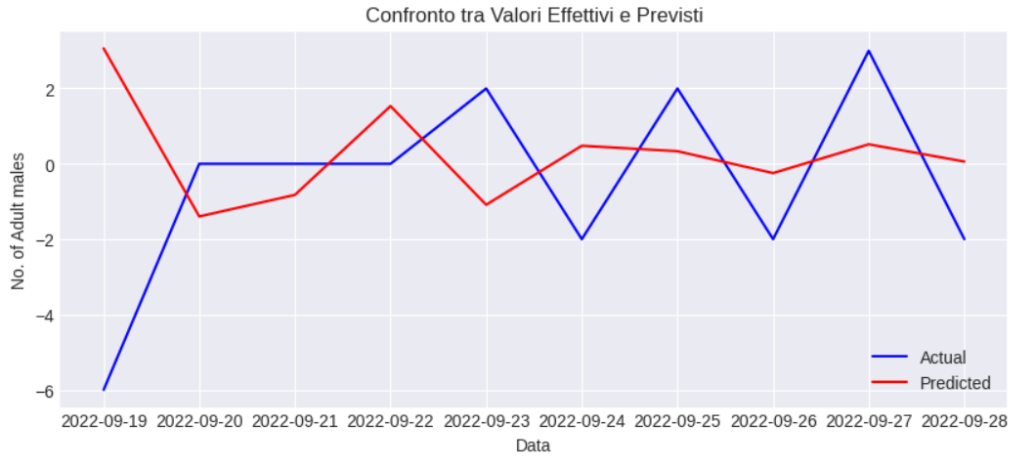
VAR (Vector Autoregression)

Il primo modello preso in esame è stato **VAR**: ottimo per l'analisi di serie multivariate in cui ogni colonna rappresenta un fattore del modello pesato per un opportuno coefficiente; da cui abbiamo ottenuto come miglior modello:

- `Lag Order 6`
- `RMSE: 2.231493323466737`

Il tutto è stato poi valutato anche sui dati di test, ottenendo:

- `RMSE: 3.450927852200181`
- `MAE: 2.6347223064268612`



ARIMAX (AutoRegressive Integrated Moving Average with eXogenous variables)

Data la natura dei dati, si è proseguito con **ARIMAX:** un'estensione di ARIMA che permette l'inclusione di variabili esogene, nel nostro caso 'relativehumidity_mean' e 'temperature_mean', che influenzano l'andamento della colonna target 'no. of Adult males'.

La prima parte del modello indica appunto la componente auto-regressiva (AR), la seconda è la componente di integrazione (I), segue la componente di media mobile (MA), e infine abbiamo quella per le variabili eXogene (X).

In particolare, abbiamo usato la versione *auto-ARIMAX* che permette di iterare il modello per trovare i parametri migliori:

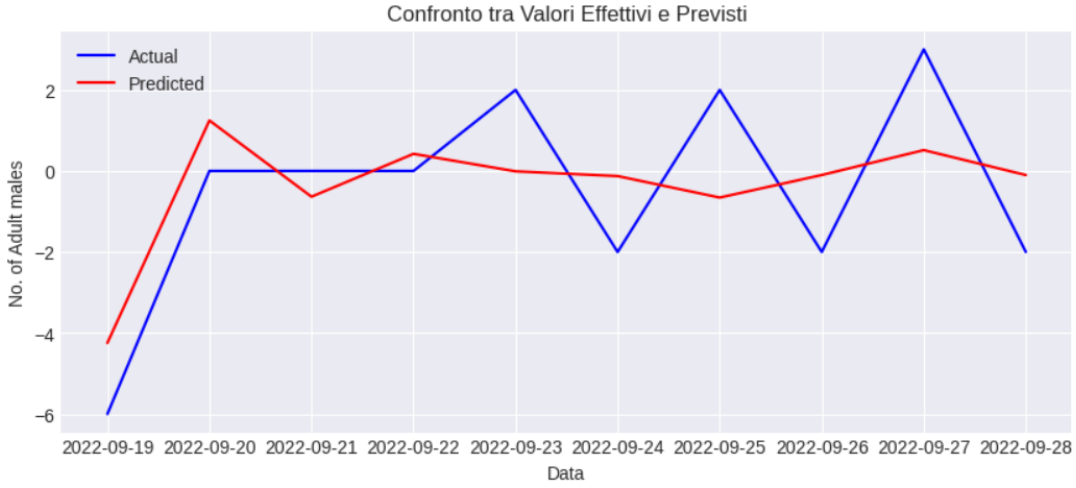
- *p*: ordine degli auto-regressori (il numero di ritardi passati da considerare)
- *d*: ordine di integrazione (il numero di differenziazioni necessarie per rendere la serie stazionaria)

- *q*: ordine dei medi mobili (il numero di errori passati da considerare)

Ottenendo come miglior risultato: `ARIMA(1,0,1)(0,0,0)[0]`; confermando l'assenza di stagionalità nella colonna di interesse.

Le metriche calcolate dalle predizioni sui dati di test risultano essere le seguenti:

- `RMSE: 1.8225002327472086`
- `MAE: 1.688201953490547`



Data la ridotta mole di dati, in letteratura, si è soliti considerare i modelli di ML quasi equivalenti a quelli di DL per prestazioni, perché questi ultimi sembrano non funzionare bene con un quantitativo di dati molto ridotto. Alla luce di ciò si è riportato sia un modello di ML che uno molto semplice di DL.

Machine Learning

Per poter usare questo tipo di modelli è stato necessario assicurarci che si preservasse la dipendenza temporale dei dati. Per tale ragione, la prima operazione che è stata effettuata è stato un preprocessing, facendo **data augumentation**: si sono aggiunte delle colonne contenenti il valore del campione al tempo t-1, creando una dipendenza dal valore precedente e quindi preservando il fattore di storicità dell'acquisizione.

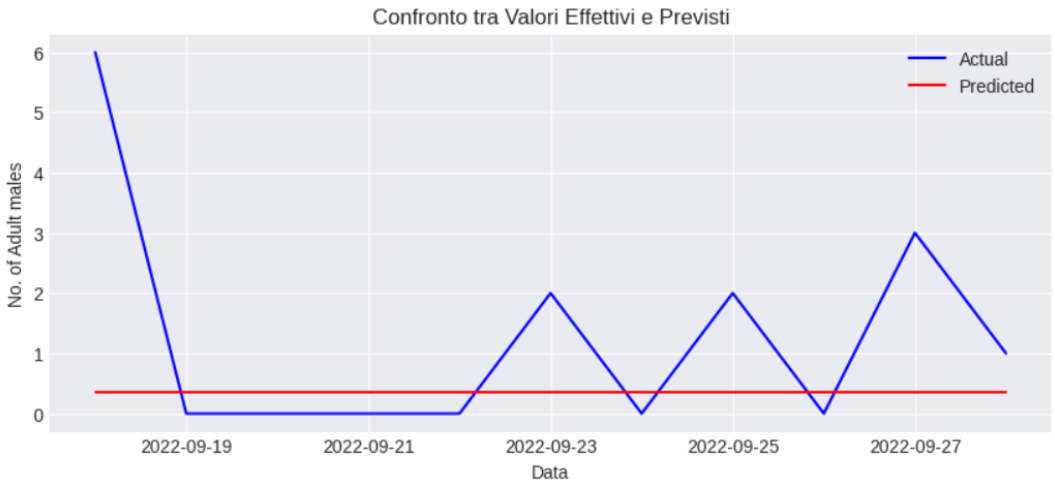
Regression Tree

Il modello scelto per il ML è stato il regression tree, un modello abbastanza semplice, ma che si adattasse bene anche ai casi di dataset ridotti. È stato effettuato un addestramento con diversi parametri come *max_depth, criterion, min_samples_split, min_samples_leaf e min_weight_fraction_leaf*; iterando combinazioni di questi parametri si è trovato come miglior modello:

```
Best Parameters: {'min_weight_fraction_leaf': 0.2,
                  'min_samples_split': 5,
                  'min_samples_leaf': 2,
                  'max_depth': 7,
                  'criterion': 'poisson'}
```

Ottenendo poi sulle predizioni:

- `Mean Squared Error: 4.115702479338843`

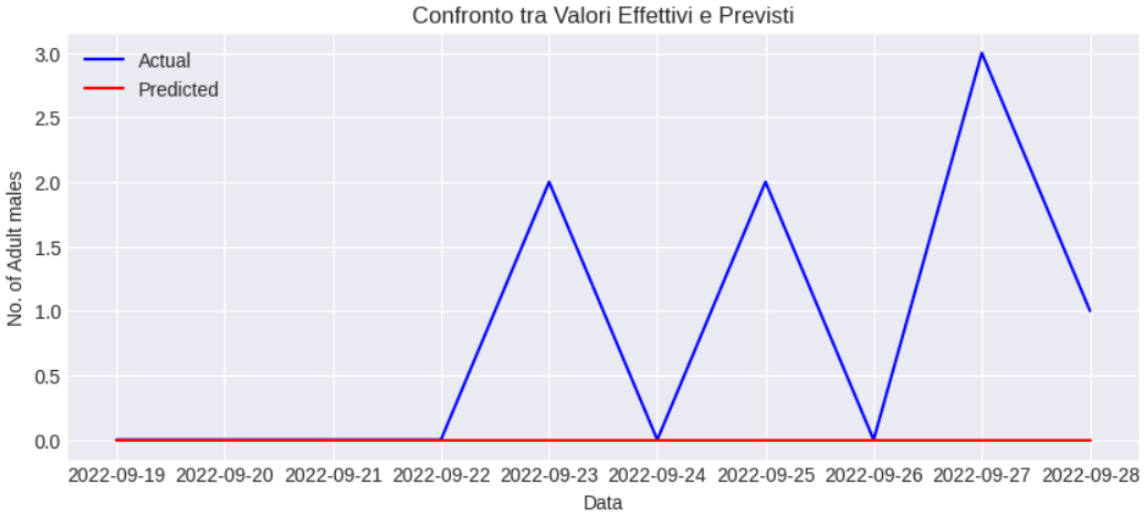


Deep Learning

Per il modello di DL è stato scelto di implementare, manualmente, una rete neurale molto semplice di tre livelli Fully-Connected e con funzione di attivazione 'ReLu'. L'addestramento è stato effettuato per un numero di epoche pari a 100, con 32 elementi per batch: si è ottenuto un gap molto elevato tra la curva di train e quella di validation. Quindi il modello performa male, come ci si aspettava.

Sui dati di test ha prodotto:

- `Mean Squared Error: 1.8`



Esercizio 2 – Dashboard Streamlit

1. Definizione della struttura della Dashboard

Per consentire un'analisi interattiva dei dataset a disposizione e permettere all'utente di addestrare il modello di previsione dei dati che ritenesse più opportuno, si è deciso di realizzare una dashboard mediante l'utilizzo della libreria Python **streamlit**. Tale libreria consente di implementare una pagina web personalizzabile con la visualizzazione di grafici e tabelle, l'inserimento di widget ed elementi di interfaccia grafica che permettono all'utente di interagire e sentirsi protagonista del processo di analisi dei dati.

La dashboard implementata è stata strutturata in due parti principali: un'analisi interattiva dei dati che permettesse di confrontare i dataset, e un'ispezione maggiormente approfondita del dataset dotato della colonna target, con la possibilità di effettuare addestramenti di modelli sulla base delle scelte optate nell'esercizio precedente.

Il passaggio preliminare è consistito nel definire la configurazione della pagina web, specificando opzioni come il nome della dashboard, l'icona visualizzata in alto a sinistra e il titolo della pagina stessa.

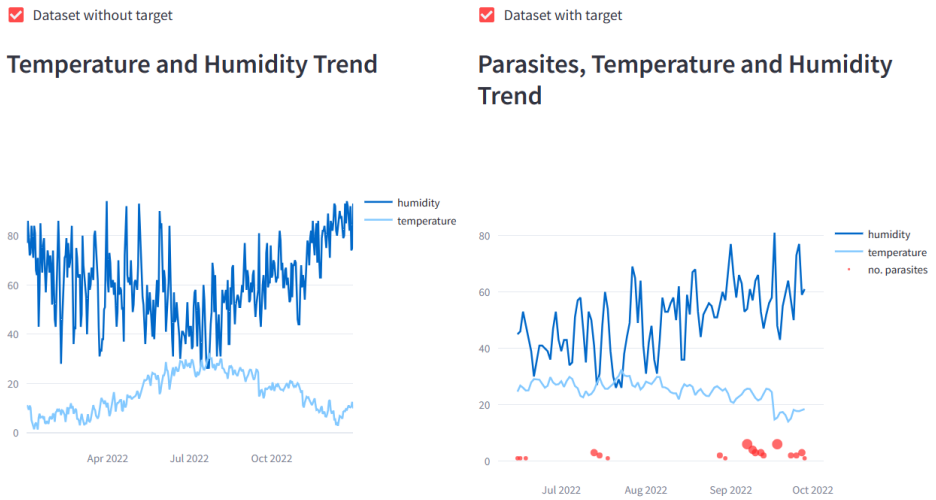
2. Confronto interattivo dei dataset

Per effettuare l'analisi dei dataset mediante streamlit, è risultato necessario scaricare i file csv relativi ai due dataset, che sono stati opportunamente caricati all'interno della cartella GitHub del progetto. La lettura dei dati è stata implementata mediante una funzione cachata, in modo da rendere più rapido il caricamento della pagina web ad ogni aggiornamento. È seguita quindi la trasformazione della colonna relativa alla data in entrambi i dataset, per rimuovere l'orario e aggiungere l'anno dove non era presente.

Per questa prima parte della dashboard, la pagina è stata divisa in due colonne: a sinistra sono riportate le informazioni riguardanti il dataset senza la colonna target, a destra quelle relative al dataset completo.

Grafici e tabelle

Il primo tipo di confronto realizzato è quello mediante osservazione dell'andamento temporale dei fattori presenti all'interno dei due dataset. Con l'utilizzo di due **checkbox**, una per ogni dataset, l'utente può decidere se visualizzare o meno il grafico corrispondente alla selezione effettuata. Per la realizzazione dei plot è stata utilizzata la libreria **plotly**: in particolare, per le colonne relative alla temperatura e all'umidità medie è stato utilizzato un grafico a linee, mentre per la colonna riguardante il numero di parassiti è stata scelta una rappresentazione mediante markers circolari, optando per la visualizzazione dei soli valori non nulli in modo da alleggerire l'impatto visivo.



Al di sotto dei plot è stata invece consentita una visualizzazione tabellare dei valori, in modo da fornire un diverso punto di vista sull'analisi dei dati. In particolare, per ogni tabella a scorrimento, è stata aggiunta una colonna *selected* di valori booleani (inizializzata a **False** nel momento in cui si effettua il caricamento dei dataset) che risulta essere l'unica colonna modificabile: spuntando la casella relativa alla riga desiderata, sarà evidenziato sul grafico corrispondente, se visualizzato, il campione corrispondente a tale riga. L'utente potrà quindi usufruire di questo tool per effettuare in maniera più rapida analisi comparative sia tra le due tabelle che all'interno della stessa, come ad esempio confrontare il valore del numero di parassiti il primo giorno di ogni mese.

Al fine di implementare questa utility, sono state definite due variabili di stato di streamlit che permettessero di mantenere la persistenza della selezione effettuata tra un aggiornamento e l'altro della pagina: in particolare, le variabili vengono aggiornate all'interno della callback *update*, chiamata subito dopo la modifica di una delle due tabelle.

Dataframe: Dataset without target

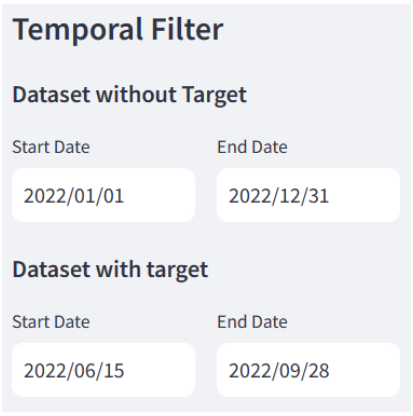
selected	time	temperature_mean	relativehumidity_mean
☐	2022-01-01	11.22	77
☐	2022-01-02	9.87	86
☐	2022-01-03	9.33	79
☐	2022-01-04	11.05	72
☐	2022-01-05	10.17	73
☐	2022-01-06	5.13	84
☐	2022-01-07	3.89	77
☐	2022-01-08	1.87	71
☐	2022-01-09	1.4	84
☐	2022-01-10	3.44	81

Dataframe: Dataset with target

selected	Date	no. of Adult males	temperature_mean	relativeh
☐	2022-06-15	1	24.62	
☐	2022-06-16	1	26.79	
☐	2022-06-17	0	26.02	
☐	2022-06-18	1	25.04	
☐	2022-06-19	0	25.09	
☐	2022-06-20	0	27.95	
☐	2022-06-21	0	29.08	
☐	2022-06-22	0	28.91	
☐	2022-06-23	0	28.88	
☐	2022-06-24	0	27.44	

Filtri temporali

All'interno della sidebar della dashboard sono stati implementati dei filtri temporali che consentono di estendere o ridurre la finestra temporale di osservazione dei grafici appena descritti. Tramite un form di input, l'utente potrà selezionare le date di inizio e fine periodo desiderate, in maniera separata per i due dataset. Mediante questa funzionalità è stato possibile ridurre la finestra di osservazione del grafico di sinistra al fine di focalizzare l'attenzione sull'andamento dei fattori di temperatura media e umidità relativa media nel solo arco temporale coperto dal secondo dataset, ottenendo un'ulteriore conferma della perfetta sovrapposizione delle due tabelle in questo periodo.



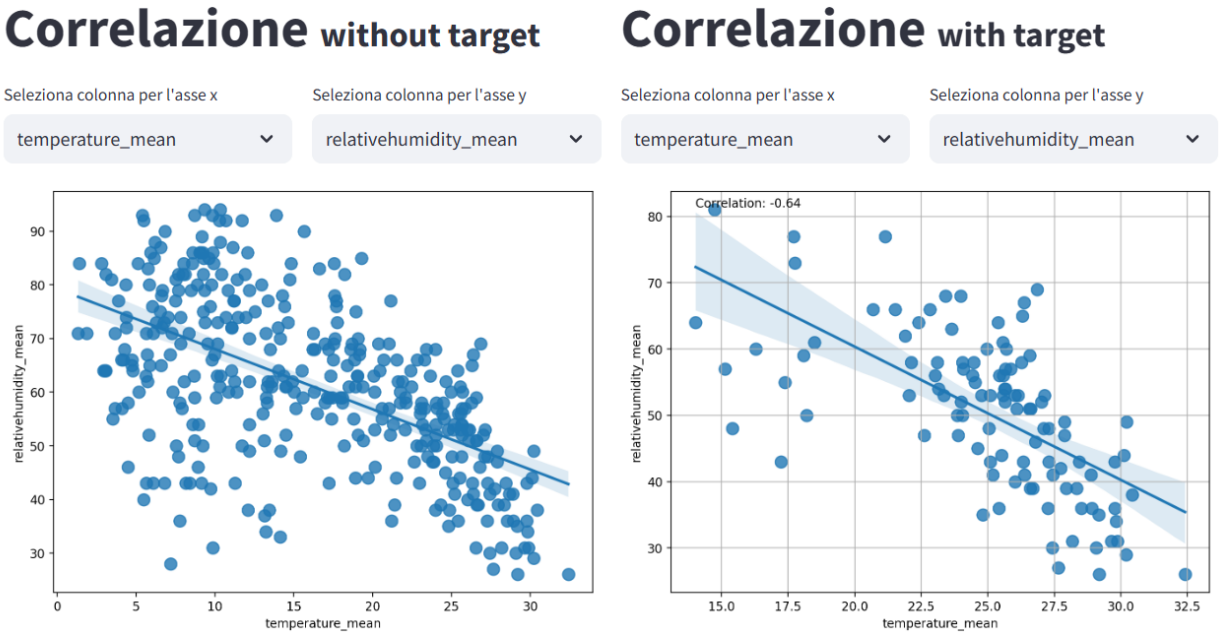
Come valori di default dei form, sono state impostate le date che permettessero di coprire il maggior arco temporale possibile, in modo da fornire una visualizzazione globale dei dati al momento del primo avvio della pagina.

Correlazione

Con lo scopo di fornire un'ulteriore punto di vista sull'analisi dei dati a disposizione, è stata inserita all'interno della dashboard la visualizzazione della correlazione tra i valori presenti all'interno dei dataset. Mantenendo

la divisione in due colonne della pagina, a sinistra è possibile osservare la correlazione tra i dati provenienti dalla tabella senza target, mentre a destra vi è il grafico relativo all'altro dataset.

In questo caso l'interattività è ottenuta mediante due **selectbox** per ciascun dataset: l'utente può selezionare a proprio piacimento la colonna da visualizzare sull'asse delle ascisse e quella da porre invece sull'asse delle ordinate. Il grafico a dispersione si adatterà alle scelte effettuate e mostrerà la correlazione tra i valori delle colonne selezionate anche mediante la visualizzazione del coefficiente di correlazione.



3. Ispezione e addestramento sul dataset con il target

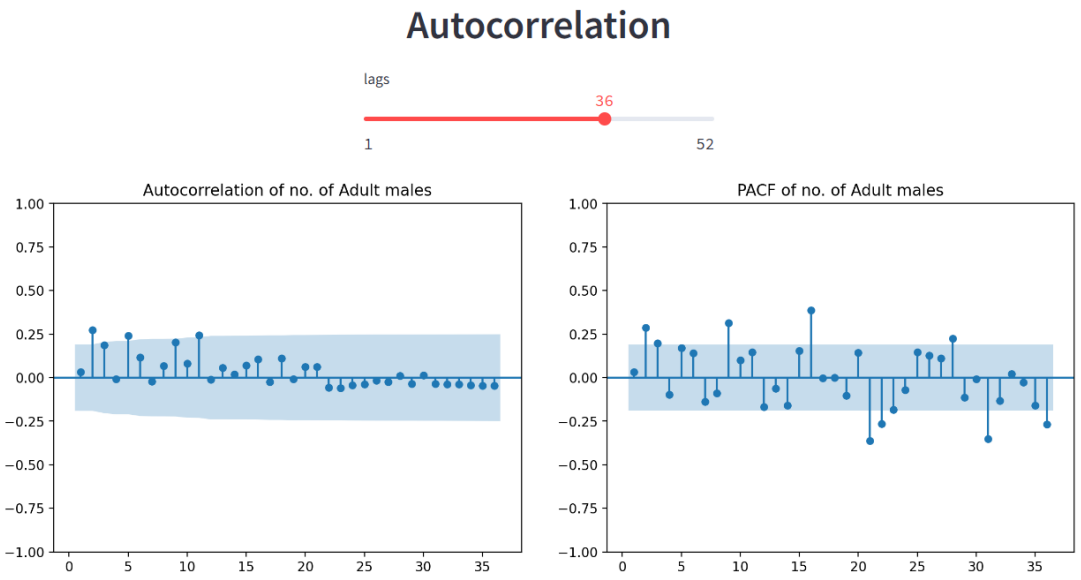
Nella seconda parte della dashboard si è deciso di spostare l'attenzione sul dataset contenente il numero di parassiti, ed in particolare proprio su questa colonna. Sono state effettuate analisi statistiche e trasformazioni al fine di preparare il dataset all'addestramento di modelli di regressione per la predizione dell'andamento temporale del target, considerato come una serie storica. Il tutto è stato realizzato al fine di garantire ancora una volta l'interattività con l'utente, per consentire confronti tra modelli diversi addestrati con parametri a sua scelta.

Autocorrelazione

L'ispezione del dataset è iniziata con l'analisi dell'autocorrelazione della colonna target.

Tramite uno slider, l'utente ha la possibilità di selezionare il numero di lags da utilizzare per il calcolo dell'autocorrelazione, a partire da un minimo di 1 fino ad arrivare a un massimo di 52, ossia il massimo consentito per il dataset a disposizione. Il valore di default è stato impostato a 20, che corrisponde al valore ottimale trovato durante l'analisi effettuata nel primo esercizio.

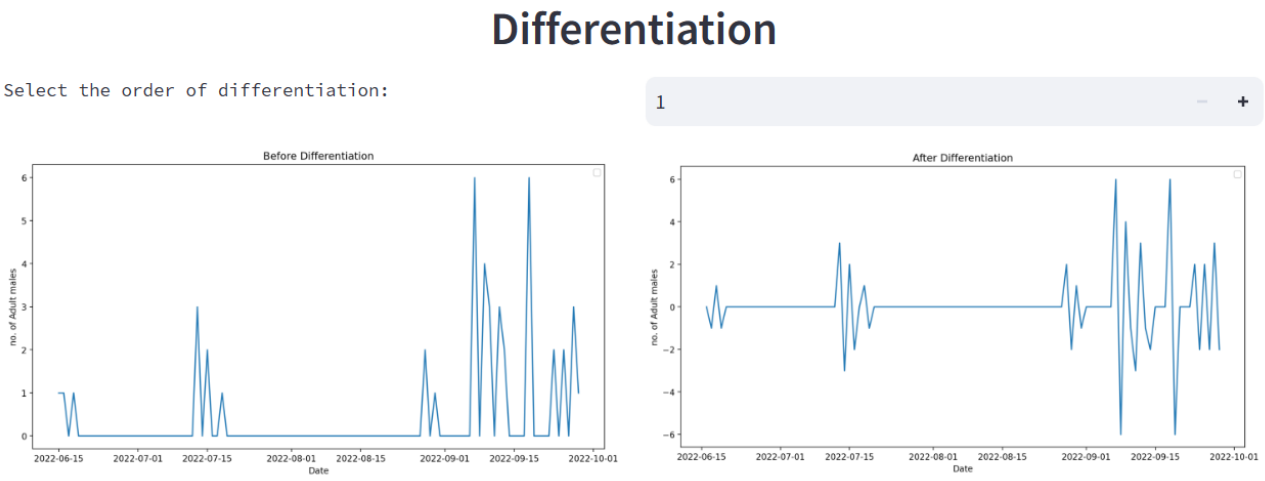
Sulla base del numero di lags indicato, la dashboard mostrerà due funzioni di auto-correlazione: a sinistra l'**ACF** e a destra la **PACF**, entrambe calcolate con i valori della colonna target utilizzando una funzione cachata. In questo modo risulterà possibile effettuare dinamicamente diversi tentativi per determinare il numero di lags ottimale per l'addestramento successivo dei modelli di regressione.



Differenziazione

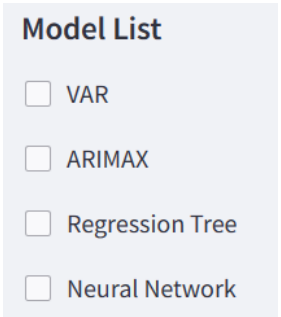
Dall'analisi di stazionarietà effettuata e documentata al punto precedente, è risultato che nessuna delle tre colonne del dataset fosse stazionaria e si è quindi deciso di effettuare una differenziazione del primo ordine. Anche all'interno della dashboard è stata inserita questa operazione, rendendola però interattiva: sarà quindi l'utente a decidere l'ordine di differenziazione da applicare alle tre serie storiche.

Tramite un widget di input numerico, si può infatti digitare un numero compreso tra 1 e 10 (di default 1) e una differenziazione dell'ordine corrispondente verrà applicata al dataset. Sulla dashboard verranno visualizzati due grafici affiancati, tramite i quali sarà possibile confrontare l'andamento temporale della serie relativa al numero di parassiti prima e dopo la differenziazione selezionata.



Addestramento dei modelli

Una volta scelto il numero di lags e l'ordine di differenziazione, il dataset è pronto per essere usato per l'addestramento di modelli regressivi che cerchino di predire l'andamento futuro della colonna target. Tramite una serie di **checkbox** implementate all'interno della sidebar della pagina, è possibile scegliere di visualizzare, e quindi addestrare, fino a quattro modelli diversi, che corrispondono ai modelli utilizzati durante il primo esercizio: un modello VAR, un modello ARIMAX, un Regression Tree e una rete neurale (multi-layer perceptron).



Quando l'utente spunta la casella relativa ad uno di questi modelli nella barra laterale, la pagina si aggiorna mostrando diverse opzioni di scelta degli iper-parametri di addestramento di quel modello, nonché i risultati delle predizioni effettuate sul test set, sia in forma grafica che testuale. In particolare, per ogni modello è stato deciso di rappresentare il confronto tra l'andamento reale e quello predetto dei valori del test set, mediante un'unica funzione di visualizzazione, realizzata con la tecnica del caching per velocizzare la ricarica della pagina.

Modelli statistici- VAR & ARIMAX

Per l'addestramento dei modelli statistici è stato preliminarmente effettuato uno splittaggio manuale del dataset differenziato in training set (90%) e test set (10%). Successivamente, è stata data all'utente la possibilità di selezionare i parametri di addestramento a proprio piacimento. Per il modello VAR, va definito il numero di lags su cui effettuare il fitting, da un minimo di 1 a un massimo di 52; per quanto riguarda ARIMAX, invece, l'utente può selezionare i 3 ordini del modello (AR,I,MA) in un range da 0 a 10. Le variabili esogene di questo modello sono la temperatura e l'umidità medie del dataset differenziato. In entrambi i casi dopo l'addestramento vengono effettuate e graficate le predizioni sul test set, calcolando e poi mostrando sulla colonna di sinistra della pagina l'**RMSE** (Root Mean Squared error) e il **MAE** (Mean Absolute Error).

Machine Learning - Regression Tree

Al fine di addestrare il Regression Tree, è stata operata una trasformazione del dataset che aggiungesse informazioni temporali: tramite una funzione di ottimizzazione si determina il numero di lags ottimale per ogni colonna, per poi aggiungere per ogni fattore (tranne il target) un numero di colonne pari a questo numero ottimale: tali colonne contengono i valori assunti dal fattore corrispondente nei lags precedenti. L'utente ha quindi la possibilità di selezionare gli iperparametri da utilizzare per l'addestramento dell'albero, quali la massima profondità dell'albero (da 1 a 10), il numero minimo di split (da 2 a 10) e il numero minimo di campioni per ogni foglia (da 1 a 10). Il dataset viene quindi splittato in training e test set secondo le stesse proporzioni di prima, viene addestrato il modello e si effettuano le predizioni sul test set, calcolando e visualizzando in particolare l'**MSE** (Mean Squared Error).

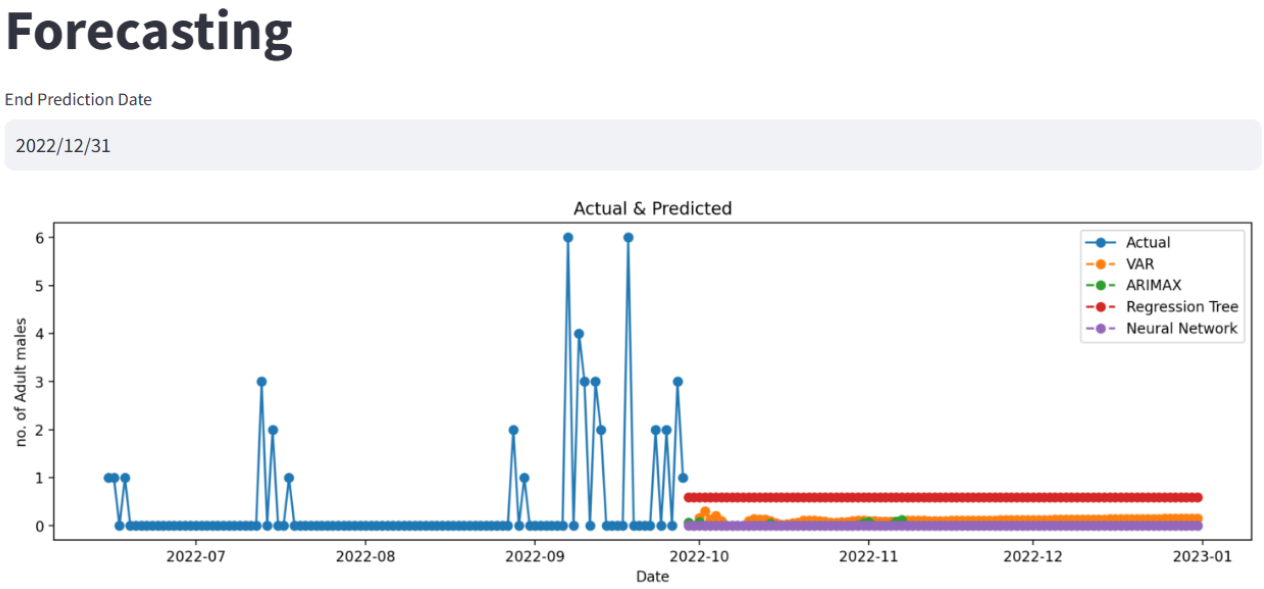
Deep Learning - Neural Network

Il quarto modello di regressione lineare consiste in una rete neurale composta da tre strati fully-connected, di 1000, 500 e 1 neurone rispettivamente, tutti e tre aventi la ReLU come funzione di attivazione. Il dataset utilizzato per il fitting del percettrone è lo stesso del regression tree, ma prima dell'addestramento si effettuano alcune operazioni di pre-processing: splittaggio in training (80%), validation (10%) e test (10%) e normalizzazione di tipo min-max tra 0 e 1 di tutte le colonne tranne il target. La funzione di loss utilizzata è l'MSE, mentre l'ottimizzatore scelto è Adam con un valore di learning rate che viene selezionato dall'utente da un minimo di 10^{-6} a un massimo di 1. Gli altri due parametri settabili sono il numero di epoche, da 1 a 100, e la dimensione dei batch, da 1 a 85 (dimensione del dataset di training). Oltre alla visualizzazione dell'MSE e dell'andamento delle predizioni sul test set, dopo il fitting viene rappresentato sulla dashboard anche un plot che confronta l'andamento della loss sul training set rispetto a quella sul validation all'avanzare delle epoche di addestramento.

Forecasting

Si è infine deciso di implementare il forecasting della serie storica relativa al numero di parassiti per giorni futuri rispetto al dataset ristretto, utilizzando i dati contenuti nel dataset sprovvisto della colonna target. Mediante un form di inserimento della data, l'utente potrà decidere fino a quale giorno (massimo 31/12/2022, l'ultima data presente nel dataset) calcolare e quindi visualizzare le predizioni, che partiranno dal 29/09/2022, ossia il giorno dopo l'ultimo presente nel dataset utilizzato come base per il fitting dei modelli.

A seconda dei modelli che sono stati selezionati nella sidebar, verrà effettuato il forecasting sul periodo selezionato utilizzando quegli stessi modelli addestrati in precedenza. Il risultato verrà visualizzato su un unico grafico finale che mostra l'andamento del target dal 15 Giugno al 29 Settembre (valori reali) e le predizioni ottenute nell'arco temporale successivo, sia per permettere l'osservazione del trend temporale in maniera continuativa, sia per confrontare contemporaneamente le predizioni realizzate dai diversi modelli.



Esercizio 3 – Power BI

Nell'esercizio documentato di seguito è stato realizzato un report per l'Exploratory Data Analysis (Analisi Esplorativa dei Dati), con l'obiettivo di comprendere la loro struttura, identificare modelli e individuare eventuali relazioni tra le variabili che sarebbero potute tornare utili in fasi predittive.

Attraverso l'utilizzo di Power BI, piattaforma di business intelligence, sono state esplorate le dinamiche interne di questi dati per estrarne valori significativi e comprendere meglio il contesto sottostante, grazie alla realizzazione di grafici e diagrammi che consentono di studiare l'andamento delle classi a disposizione.

Inner & Outer Join

Subito dopo aver caricato i dati all'interno di Power BI, è stata realizzata una doppia operazione di merge dei dataset con l'obiettivo di mettere in risalto le somiglianze e le differenze tra le due tabelle, preparandole in questo modo all'elaborazione grafica successiva.

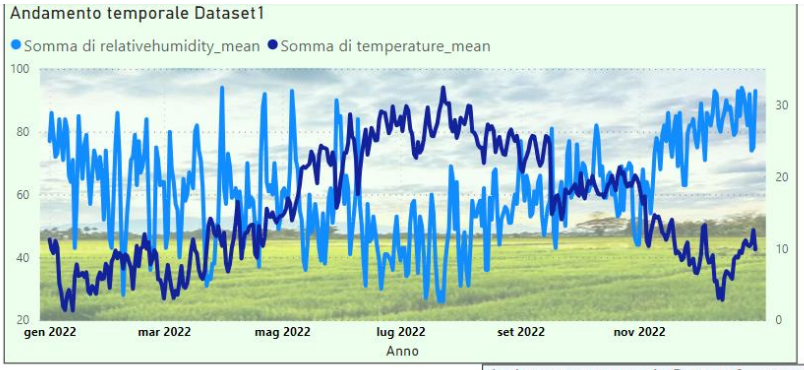
È stato dapprima realizzato un **inner join** selezionando come colonne corrispondenti la colonna 'time' del dataset senza target e la colonna 'Date' dell'altro dataset, ottenendo come risultato una tabella che comprendesse i valori di umidità e temperatura provenienti da entrambe le tabelle, ma solo rispetto all'arco temporale comune.

In seguito, è stato implementato un **full outer join** che permettesse di coprire tutto l'anno 2022, lasciando valore nullo in corrispondenza dei giorni non presenti all'interno del dataset con il target. La tabella risultante è servita per analizzare le differenze tra le due tabelle, come per la distribuzione complessiva dei dati.

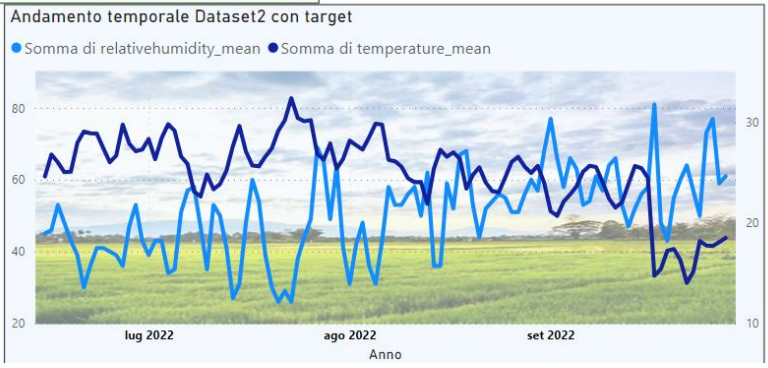
Temporal Analysis

In una prima istanza è stata realizzata una rappresentazione visiva dei due dataset messi a disposizione in modo da studiarne relative somiglianze e andamenti. Da qui, come era già stato possibile osservare sotto forma tabellare, il secondo dataset risultava essere una sottoparte del primo, a cui è stata aggiunta una colonna: 'no. of Adults males'.

Per la realizzazione dei grafici è stato selezionato il grafico a linee, posizionando sull'asse delle x la colonna relativa alla data, e su quello delle y la temperatura media e l'umidità relativa media.

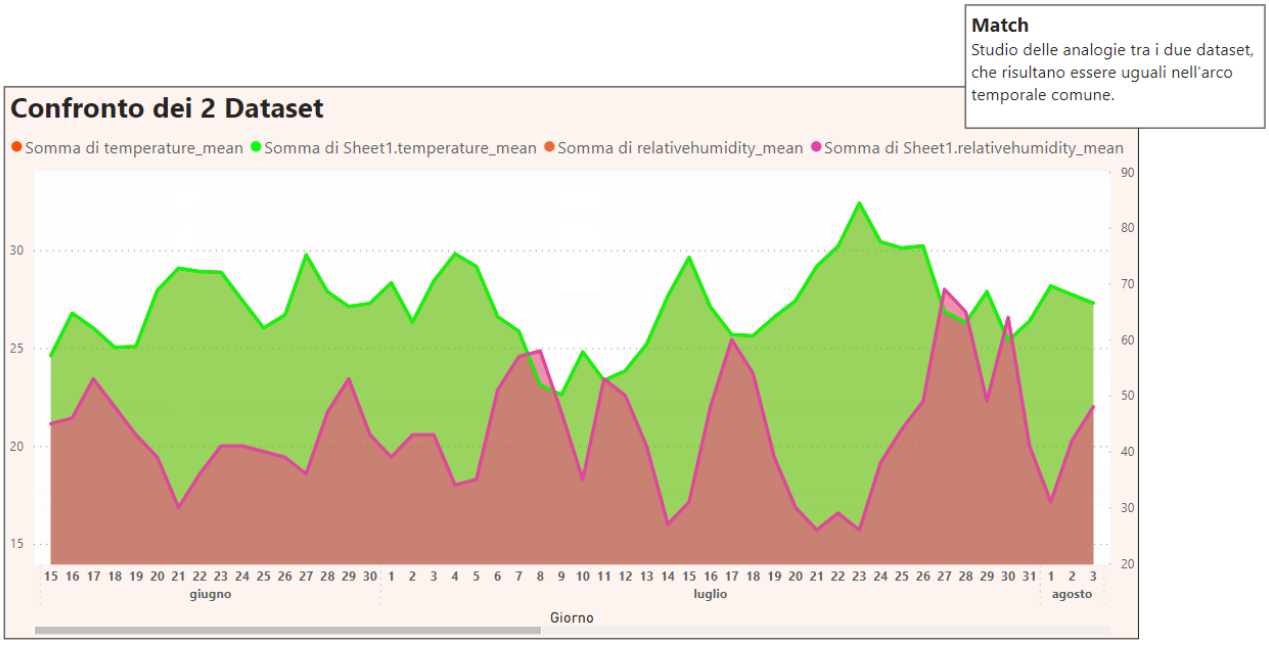


Studio preliminare del dataset
Analisi dell'andamento temporale dei due dataset

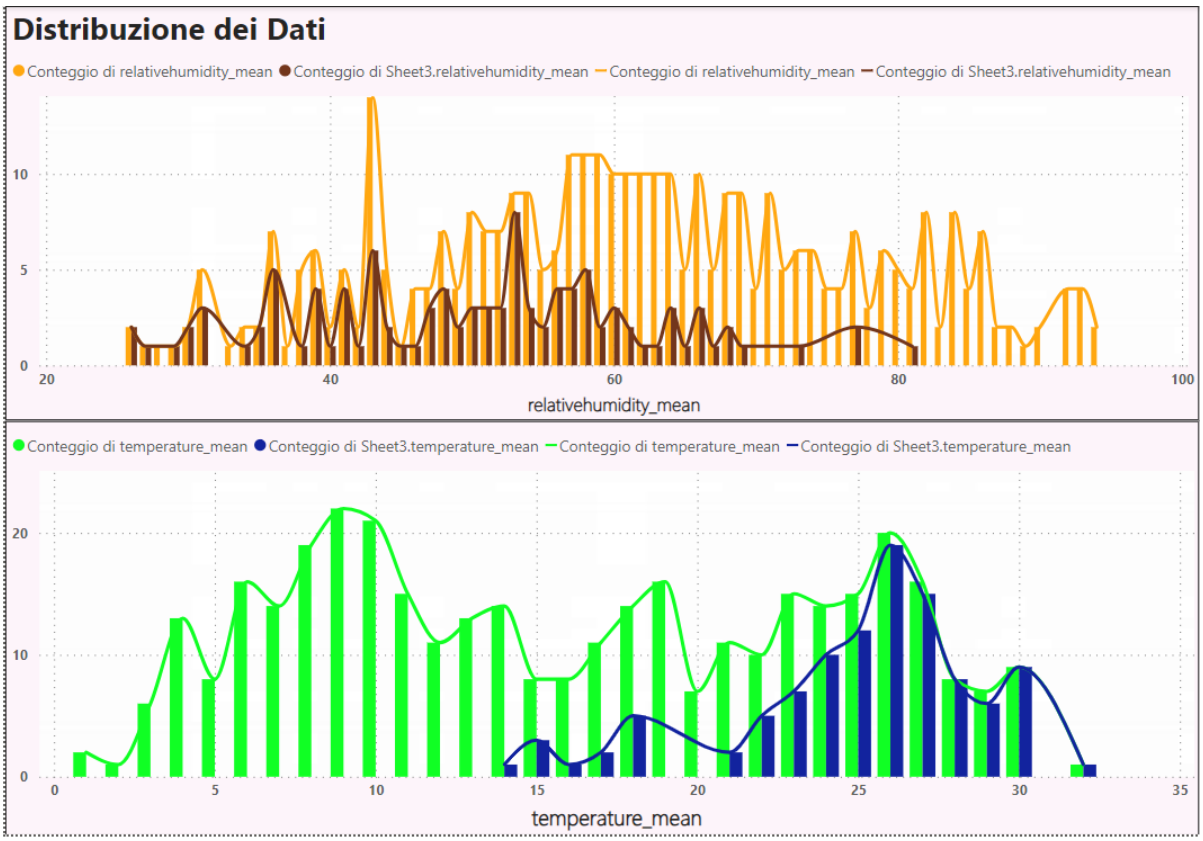


Dataset Match

Per quanto detto, si è proseguito con un vero e proprio confronto tra i due dataset, appurando con un grafico ad aree come i valori di media, varianza, deviazione standard, somma, conteggio e mediana coincidessero tra loro per entrambe le classi messe a disposizione tra i due dataset per il periodo comune.



Per poter comprendere anche come questi dati si comportassero nel tempo in relazione alla specifica colonna sotto esame abbiamo proseguito con l'analisi delle distribuzioni dei dati delle relative colonne 'relativehumidity_mean' e 'temperature_mean'.

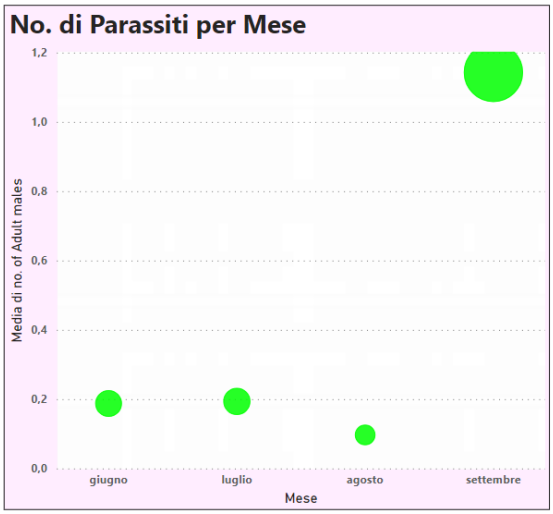


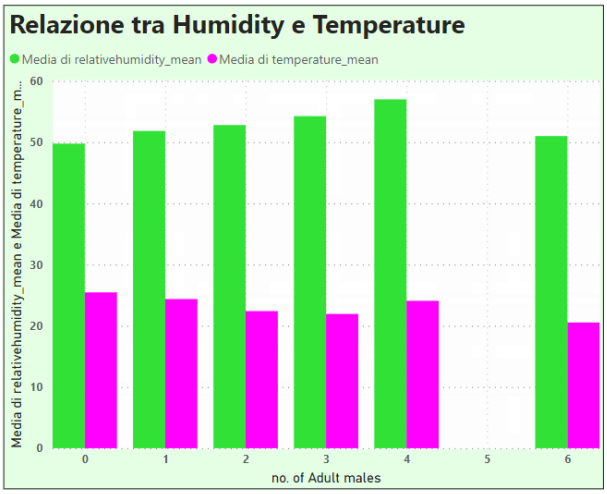
Elaborato di Information System & Business Intelligent
Studente: DAIANA CIPOLLARO FRANCESCO DI SERIO

Target Analysis

Una volta appurata la natura dei dati e la relazione che sussisteva tra i due dataset siamo passati allo studio del secondo dataset, e in particolare alla classe 'no. of Adult males', che abbiamo assunto come classe target. Ne abbiamo studiato la distribuzione della media al variare dei mesi, notando la sua irregolarità.

Dopodiché sono state condotte delle analisi di tal classe anche in relazione alle altre due colonne, per comprendere come i valori di interesse variassero in relazione a quelli noti messi a disposizione. Ed è stata notata l'assenza di ogni tipo di relazione. I valori ottenuti risultano essere pressappoco costanti, ad indicare che non ci sia un vero e proprio pattern o relazione di discriminazione delle due classi per quella di interesse e implicando così anche delle osservazioni riguardanti il fatto che le predizioni che quindi si sarebbero ottenute in seguito non sarebbero state accurate.



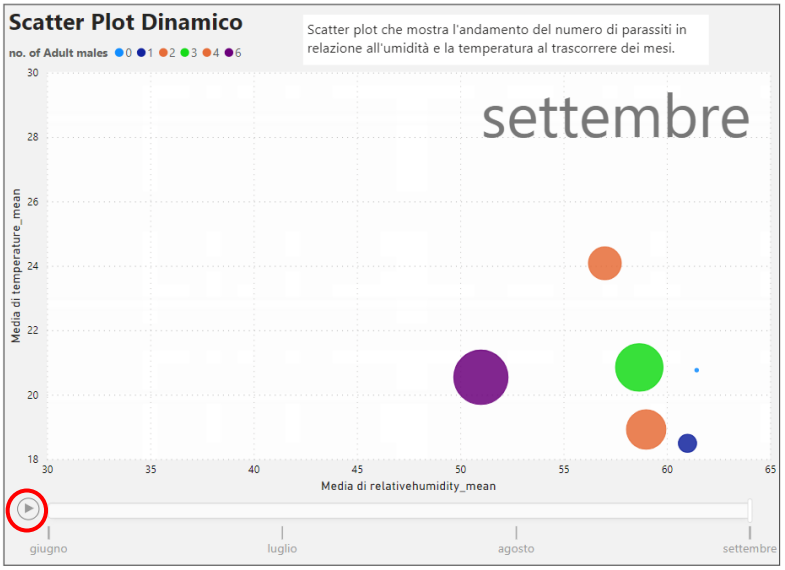


Non c'è una relazione evidente tra la colonna target e le altre, ma si nota un leggero trend all'aumentare il n.ro di parassiti tra [0,4], per quanto riguarda l'umidità media.

Scatter Plot

In ultima istanza è stato realizzato anche uno scatter plot dinamico per rendere un po' più interattivo lo studio condotto finora, evidenziando come questi valori target non fossero in qualche evidente modo legati a quelli delle colonne 'relativehumidity_mean' e 'temperature_mean'.

Lo scatter plot riporta sull'asse delle x l'umidità e su quello delle y la media della temperatura, permettendo di visualizzare l'andamento del n.ro di parassiti al trascorrere dei mesi da giugno a settembre.



Esercizio 4 – Bonita

Il progetto realizzato è stato incentrato su un caso d'uso che illustra il flusso di lavoro per il processo di immagazzinamento e spedizione di un'azienda di consegne di pacchi (come GLS, poste italiane, Bartolini, …).

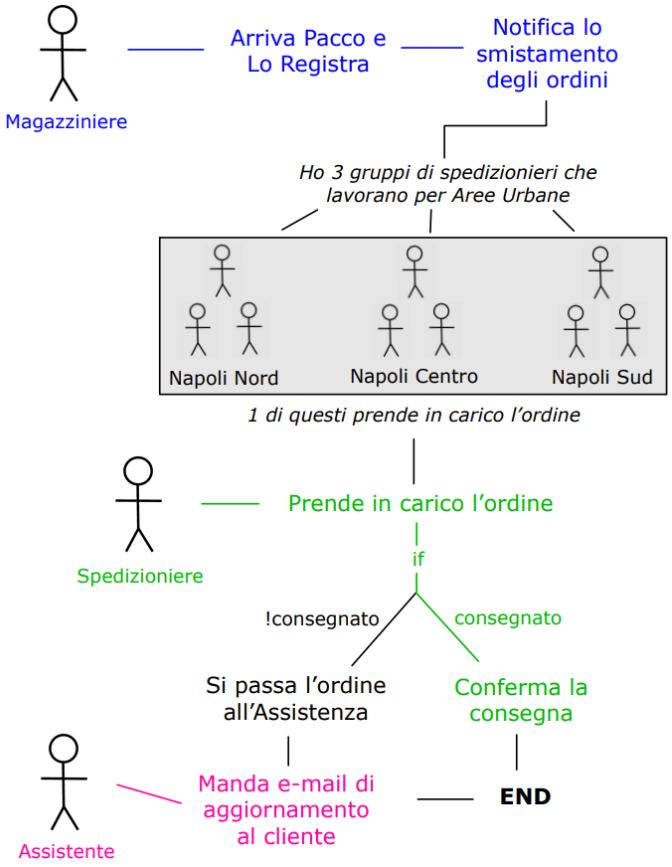
Precondizioni: un cliente ha effettuato un ordine, fornendo le informazioni di spedizione.

Una volta che il pacco giunge in magazzino, l'attore Operatore del Magazzino ne fa una rassegna inserendo nel sistema tutte le informazioni relative all'ordine, tra le quali la locazione del pacco nel plesso. Una volta terminata la rassegna delle spedizioni etichettate per aree, i vari corrieri potranno prendere in carico l'ordine relativo alla loro zona di competenza ed effettuare la consegna.

Il flusso del progetto si basa sullo schema seguente (a cui sono state apportati miglioramenti e aggiustamenti in fase di elaborazione del modello):

I ruoli fondamentali individuati sono:

- Magazziniere
- Corriere/Spedizioniere
- Assistente



I colori identificano i vari task che ci si aspetta siano svolti dai rispettivi attori.

Il flusso ha inizio con l'arrivo del pacco al magazzino, dove il magazziniere prende in carico l'ordine e lo registra come tale nel DB specificandone i vari attributi.

Una volta terminata questa fase preliminare, i corrieri possono prendere in consegna i pacchi a seconda della loro zona di competenza e spedirli. Se la consegna va a buon fine lo notificano al sistema e compilano una ricevuta con i dettagli del relativo ordine.

In caso contrario, l'ordine dovrà ritornare in giacenza e quindi il magazziniere dovrà segnalarne il ripristino e riposizionarlo in magazzino, tenendo, così, sempre aggiornato lo stato del prodotto e la sua localizzazione. Contemporaneamente l'assistenza clienti si metterà in contatto con il cliente per informarlo della mancata consegna dell'ordine.

Workflow

1. Start

Il flusso di esecuzione parte con l'assunzione che un cliente abbia richiesto all'azienda di spedire un ordine. Questi arriva al magazzino e deve essere gestito: sarà un operatore a far partire l'intera istanza di processo. Mediante il filtro "Actor filter iniziator", il magazziniere in questione sarà designato come l'iniziatore del processo durante tutta la sua esecuzione.

2. Register Order

L'addetto al magazzino prende in carico l'ordine e ne effettua la registrazione sulla base dati, inserendo:

- Codice identificativo
- Data di arrivo del pacco
- Locazione
- Tipologia
- Indirizzo di destinazione
- CAP

3. *Gateway – Which zone?*

   Una volta segnalato il CAP è necessario che quest'ordine sia visibile solo al gruppo di corrieri che tratta una determinata zona: Napoli Nord, Napoli Centro, Napoli Sud. E per rendere disponibile questo pacco solo al gruppo interessato è stato usato un gateway di tipo esclusivo con 3 output.

4. Deliver Order

   Quando il corriere fa il proprio accesso nel sistema, gli compariranno tutti gli ordini, che per zona di competenza, può prendere in carico. Se questi decide di proseguire con la spedizione aggiorna lo stato del pacco come 'in consegna'.

   È stata definita anche una variabile di processo per consentire di tracciare l'informazione del corriere che sta trattando l'articolo in maniera esclusiva con l'utilizzo di un particolare tipo di filtro per attori: 'Same task user'. In questo modo, se l'ordine viene preso in carico da un determinato corriere non potrà essere visualizzato dagli altri.

5. *Gateway – Delivered?*

   Se la consegna va a buon fine l'impiegato setta la variabile 'Delivered' come *true*, altrimenti a *false*, ed è stato implementato un altro gateway per permettere lo split del flusso.

6. Generate Receipt

   Se '*Delivered == true*' segue un task che riguarda la compilazione della ricevuta d'acquisto, in particolare con il campo 'forwarder', e l'aggiornamento della 'end date' nello shipment del pacco del magazzino.

7. Update Receipt

   Si conclude questo primo flusso di esecuzione con un task di Sistema, che a differenza dei precedenti (che erano tutti task umani) coinvolge la presenza diretta del sistema quale attore principale dell'azione. Infatti, questi aggiorna lo 'stato' dell'ordine. E setta le variabili della ricevuta: 'Data di Arrivo dell'ordine' e il 'codice identificativo'.

8. *Gateway – Restore & Mail*

   Qualora la consegna non fosse andata a buon fine si avviano due azioni in parallelo svolte da due attori differenti ed è per tale ragione che è stato fatto uso di un gateway parallelo.

9. Restore

   Una volta che il corriere riporta il pacco in magazzino, il magazziniere che ha iniziato il flusso di processo riprende in carico l'ordine e ne aggiorna lo 'stato' come in giacenza e lo riloca all'interno del magazzino aggiornando eventualmente la 'locazione'.

10. Send Mail

    Se la consegna non è stata possibile risulta necessario informare il cliente dello stato di avanzamento del prodotto e che questo sarebbe passato in stato di giacenza.

La comunicazione è stata realizzata tramite l'uso di un provider fittizio con il quale si effettua un collegamento grazie al connettore istanziato con l'estensione 'Email' che permette di mandare messaggi di posta elettronica.

In questo ultimo task, un adetto all'assistenza si fa carico della comunicazione con l'utente stilando una mail che includa un 'subject' e un 'body' da compilare.