UNIVERSITA' DEGLI STUDI DI
NAPOLI FEDERICO II

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

Elaborato Esame - Google Colab
Information Systems and Business Intelligence

Prof. Flora Amato
Anno Accademico 2023/2024

Francesco Panariello M63/1433

Riccardo Romano M63/1489

Giovanni Riccardi M63/1480

Sommario

Capitolo 1: Introduzione ed Analisi

1.1 Scopo del Progetto

Il presente progetto si propone di analizzare e interpretare in modo significativo un dataset che contiene informazioni sulla popolazione batterica, la temperatura e l'umidità. L'obiettivo principale è formulare previsioni sulle tendenze emergenti attraverso l'estrazione di dati rilevanti. In particolare, si mira a predire il numero di maschi adulti all'interno di una popolazione batterica, basandosi su fattori ambientali cruciali come temperatura e umidità. Per raggiungere questo obiettivo, sarà necessario adottare un approccio metodico e impiegare tecniche avanzate di analisi dei dati.

#Estratto visivo dei dataset sheet1 e sheet3

time	temperature_mean	relativehumidity_mean		Date	no. of Adult males	temperature_mean	relativehumidity_mean
2022-01-01	11,22	77		15-giu	1	24,62	45
2022-01-02	9,87	86		16-giu	1	26,79	46
2022-01-03	9,33	79		17-giu	0	26,02	53
2022-01-04	11,05	72		18-giu	1	25,04	48
2022-01-05	10,17	73		19-giu	0	25,09	43
2022-01-06	5,13	84		20-giu	0	27,95	39
2022-01-07	3,89	77		21-giu	0	29,08	30
2022-01-08	1,87	71		22-giu	0	28,91	36
2022-01-09	1,4	84		23-giu	0	28,88	41
2022-01-10	3,44	81		24-giu	0	27,44	41
2022-01-11	4,19	66		25-giu	0	26,03	40
2022-01-12	2,95	64		26-giu	0	26,69	39
2022-01-13	1,33	71		27-giu	0	29,77	36
2022-01-14	6,1	43		28-giu	0	27,89	47
2022-01-15	7,76	58		29-giu	0	27,13	53
2022-01-16	6,14	85		30-giu	0	27,29	43
2022-01-17	6,29	73		01-lug	0	28,34	39
2022-01-18	6,59	65		02-lug	0	26,33	43
2022-01-19	6,02	76		03-lug	0	28,44	43
2022-01-20	6,66	79		04-lug	0	29,82	34
2022-01-21	4,07	66		05-lug	0	29,17	35
2022-01-22	3,65	57		06-lug	0	26,61	51
2022-01-23	4,75	66		07-lug	0	25,86	57
2022-01-24	4,38	74		08-lug	0	23,11	58
2022-01-25	3,67	71		09-lug	0	22,62	47
2022-01-26	5,88	65		10-lug	0	24,81	35
2022-01-27	6,64	72		11-lug	0	23,35	53
2022-01-28	5,71	67		12-lug	0	23,85	50
2022-01-29	5,82	52		13-lug	3	25,2	41

1.2 Caricamento e Analisi Esplorativa dei Dati

Il primo passo pratico del progetto consiste nel caricare i dati dai fogli "Sheet1" e "Sheet3" del dataset fornito. L'analisi esplorativa iniziale rappresenta un tassello fondamentale, poiché consente di ottenere una visione d'insieme dei dati a disposizione, comprendendone la struttura e individuando eventuali anomalie o punti di particolare interesse.

```
import pandas as pd

# Carica i dati
file_path = 'temp_humid_data.xlsx'
s1 = pd.read_excel(file_path, sheet_name='Sheet1')
s3 = pd.read_excel(file_path, sheet_name='Sheet3')
```

Per eseguire questa fase, sarà utilizzato il modulo pandas. L'importazione dei dati mediante il metodo read_excel consentirà di acquisire direttamente le informazioni dal file Excel, fornendo così una base solida per l'effettuazione di analisi efficienti e accessibili.

1.3 Gestione e Preparazione dei Dati

La fase di pulizia e preparazione dei dati riveste un ruolo cruciale per assicurare la validità delle analisi condotte. In questa fase, ci occupiamo di eliminare le righe che contengono valori mancanti, anche se è importante notare che nel dataset fornito non sono presenti tali valori (questa procedura viene comunque eseguita a fini didattici).

#Valutazione presenza di Missing Value

```
print(f'Sheet 1 - I missing value per ogni attributo sono : \n{s1.isna().sum()}
')
print(f'Start time: ' + sheet1_data.Date.min().strftime('%Y-%m-%d'))
print(f'End time: ' + sheet1_data.Date.max().strftime('%Y-%m-%d') )
print(f'\n\nSheet 3 - I missing value per ogni attributo sono :
\n{s3.isna().sum()} ')
print(f'Start time: ' + sheet3_data.Date.min().strftime('%Y-%m-%d'))
print(f'End time: ' + sheet3_data.Date.max().strftime('%Y-%m-%d') )
```

```
Sheet 1 - I missing value per ogni attributo sono :
temperature_mean         0
relativehumidity_mean    0
dtype: int64
Start time: 2022-01-01
End time: 2022-12-31


Sheet 3 - I missing value per ogni attributo sono :
no. of Adult males       0
temperature_mean         0
relativehumidity_mean    0
dtype: int64
Start time: 2023-06-15
End time: 2023-09-28
```

Gestione dei dati mancanti

```
# Pulizia dei dati (gestione dei valori mancanti, ecc.)
s1 = s1.dropna() # Rimuove righe con valori mancanti
s3 = s3.dropna() # Rimuove righe con valori mancanti
```

Attraverso l'utilizzo del metodo dropna, procediamo con l'eliminazione delle righe che presentano valori mancanti. Questo passo assume un'importanza fondamentale al fine di prevenire distorsioni o interpretazioni erronee nelle fasi successive dell'analisi.

1.4 Analisi Descrittiva

L'analisi descrittiva rappresenta un passo significativo per ottenere una comprensione di base delle caratteristiche intrinseche dei dati. Durante questa fase, vengono calcolate statistiche descrittive quali media, mediana, valore minimo, valore massimo, deviazione standard, asimmetria (skewness) e l'appiattimento rispetto alla distribuzione normale (Kurtosis).

Inoltre, si è utilizzata una funzione "compare_datasets_statistics" cosi da avere anche metriche di comparazione dei due dataset.

```
def compare_datasets_statistics(df1, df2):

    def get_statistics(df):
        stats = df.describe().T
        stats['Skewness'] = df.skew()
        stats['Kurtosis'] = df.kurtosis()
        return stats

    def compare_columns(df1, df2, common_columns):
        comparison_results = {}
        for column in common_columns:
            if    df1[column].dtype    in    [np.float64,    np.int64]    and
df2[column].dtype in [np.float64, np.int64]:
                stat,        p        =        ks_2samp(df1[column].dropna(),
df2[column].dropna())
                comparison_results[column]  =  {'KS  Statistic':  stat,  'P-
Value': p}
        return pd.DataFrame(comparison_results).T

    stats_df1 = get_statistics(df1)
    stats_df2 = get_statistics(df2)

    common_columns = set(df1.columns).intersection(set(df2.columns))
    comparison_results_df = compare_columns(df1, df2, common_columns)

    return stats_df1 , stats_df2 , comparison_results_df
```

Le statistiche descrittive forniscono una panoramica immediata delle caratteristiche principali dei dati, quali la distribuzione delle temperature e dell'umidità, nonché il numero di maschi adulti nel corso del tempo. Questa analisi assume un ruolo fondamentale nell'identificare tendenze, individuare dati anomali e guidare le fasi successive del progetto.

Analisi descrittiva di base

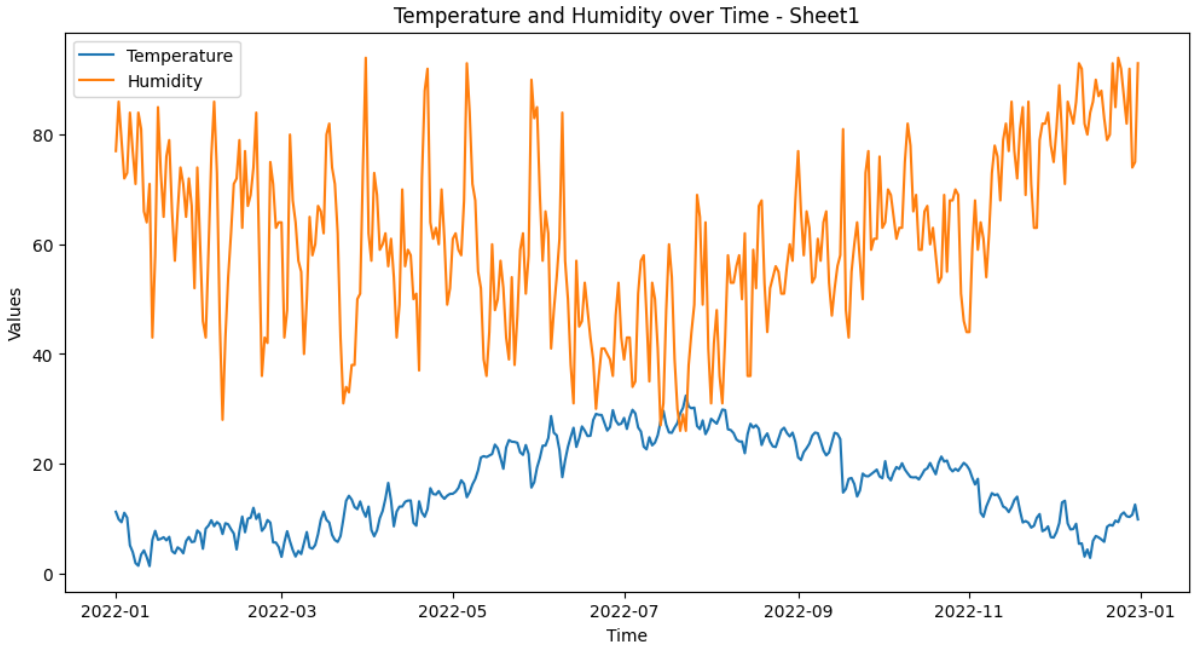
```
 statistical analysis of the first dataset 1
                       count       mean        std    min    25%    50%  \
temperature_mean       365.0  16.038740   7.965726   1.33   9.15  15.41
relativehumidity_mean  365.0  61.249315  15.660750  26.00  50.00  61.00

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

 statistical analysis of the second 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

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

I VALORI DI SKEWNESS E KURTOSIS NEI DATI INDICANO CHE LA DISTRIBUZIONE DELLA TEMPERATURA MEDIA È SIMMETRICA E APPIATTITA NEL PRIMO DATASET E LEGGERMENTE ASIMMETRICA E APPIATTITA NEL SECONDO, MENTRE LA DISTRIBUZIONE DELL'UMIDITÀ RELATIVA È APPROSSIMATIVAMENTE SIMMETRICA E ANCH'ESSA LEGGERMENTE APPIATTITA IN ENTRAMBI I DATASET.

I risultati del test di Kolmogorov-Smirnov indicano che le distribuzioni di temperature_mean e relativehumidity_mean tra i due dataset sono statisticamente diverse. In altre parole, ci sono differenze significative nelle distribuzioni di temperatura media e umidità relativa tra i due insiemi di dati.
In realtà, come vedremo tramite powerBI, i dati dello sheet3 non son altro che un estratto di quelli dello sheet1

1.5 Visualizzazione dei dati sheet 1

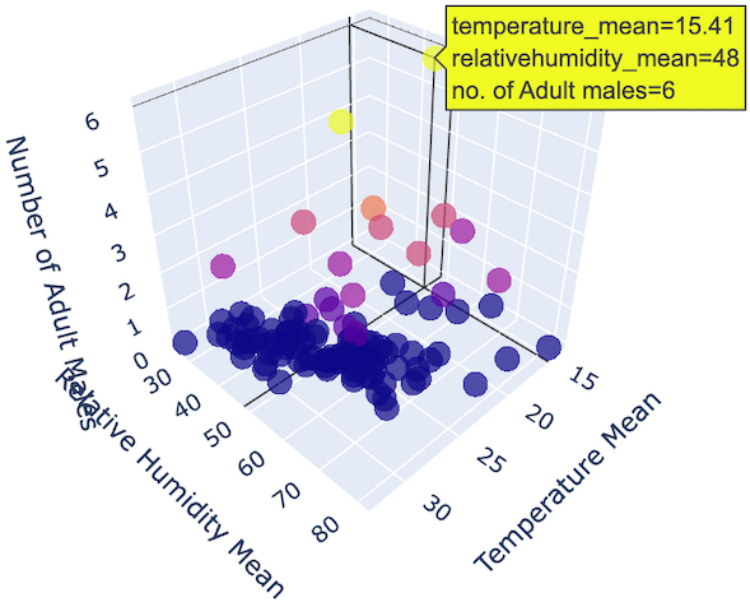
Utilizziamo grafici per esplorare e interpretare le tendenze e le relazioni nei dati. Queste rappresentazioni visive ci aiutano a ottenere una comprensione più approfondita delle informazioni e a individuare pattern visivi.
Il codice fornisce un esempio di come creare tali grafici utilizzando la libreria Matplotlib.



La visualizzazione dei dati di temperatura e umidità nel tempo ci consente di individuare possibili tendenze stagionali o anomalie. Questa rappresentazione visiva costituisce uno strumento potente per l'analisi esplorativa, offrendo insights immediati che potrebbero non emergere chiaramente dai dati in forma tabellare.

1.6 Visualizzazione dei dati sheet 3

Per esplorare e comprendere l'influenza combinata della temperatura e dell'umidità sul numero di maschi adulti, ho creato una visualizzazione tridimensionale interattiva, che può essere orientata per una visione più dettagliata. Questa rappresentazione ci consente di osservare in modo intuitivo le interazioni tra le tre variabili, offrendo una visione approfondita della loro relazione

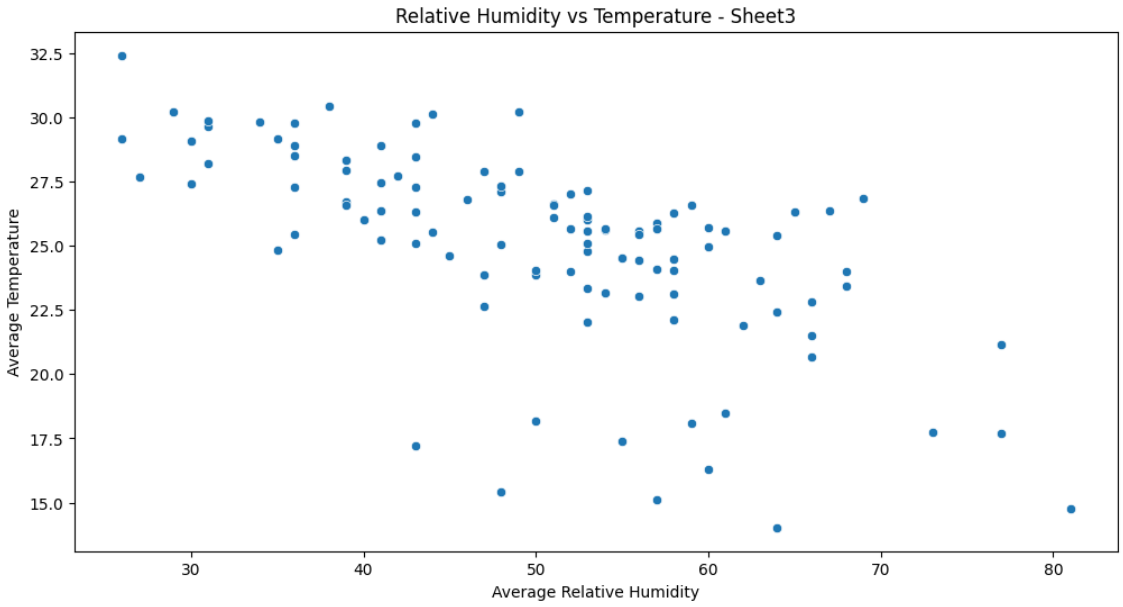


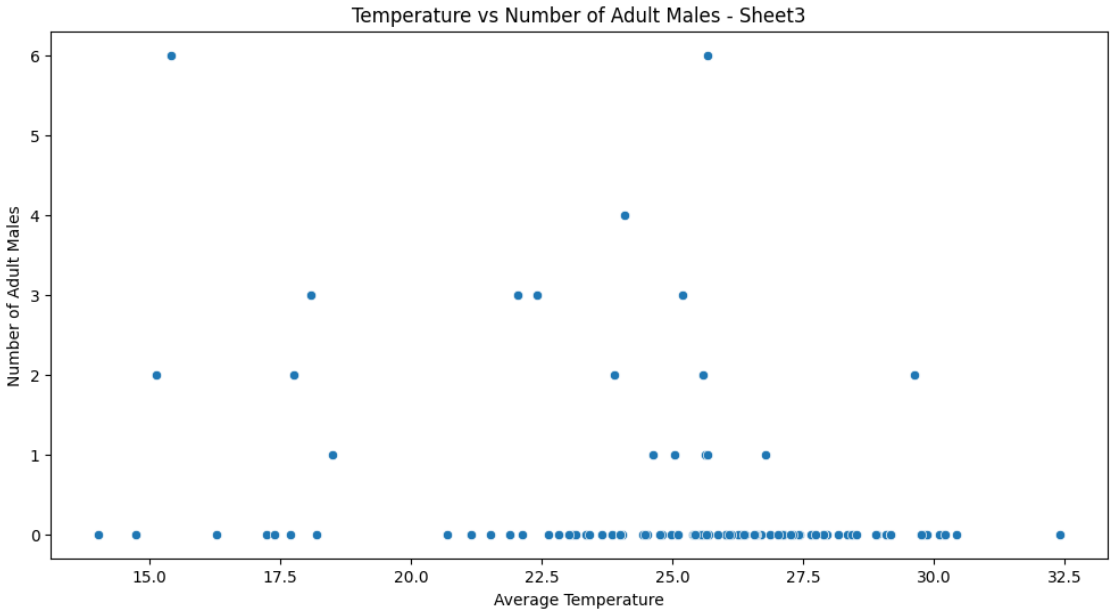
La visualizzazione tridimensionale utilizza uno scatter plot per mostrare chiaramente come le variazioni di temperatura e umidità influenzino il numero di maschi adulti. Questo tipo di grafico offre un'immagine chiara delle relazioni tra le variabili ambientali e la risposta osservata nel numero di maschi adulti.

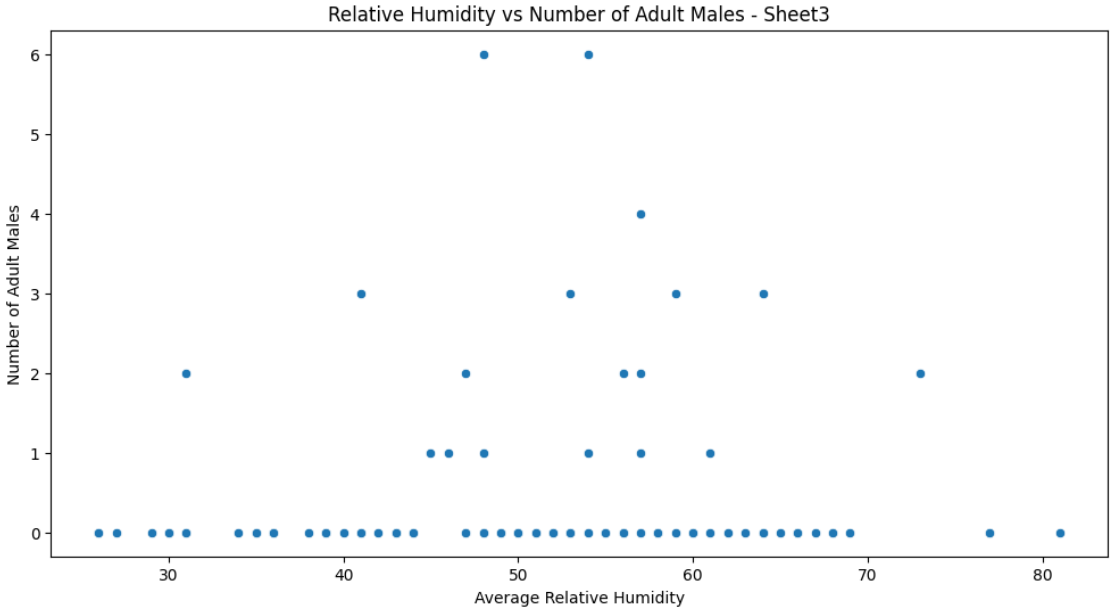
Per esplorare ulteriormente le relazioni tra gli attributi, sono stati impiegati anche grafici bidimensionali. Questi grafici forniscono una prospettiva più dettagliata sulle interazioni tra coppie specifiche di variabili, offrendo un'analisi più approfondita dei legami tra temperatura, umidità e il numero di maschi adulti. Sebbene la visualizzazione tridimensionale offra una visione complessiva delle relazioni, i grafici bidimensionali sono stati utilizzati per approfondire l'esame di connessioni specifiche tra le coppie di attributi, ampliando così la nostra comprensione complessiva del dataset.

#Codice esplicativo di uno dei grafici

```
import seaborn as sns
# Grafico della temperatura vs numero di maschi adulti
plt.figure(figsize=(12, 6))
sns.scatterplot(x='temperature_mean', y='no. of Adult males', data=s3)
plt.title('Temperature vs Number of Adult Males - Sheet3')
plt.xlabel('Average Temperature')
plt.ylabel('Number of Adult Males')
plt.show()
```

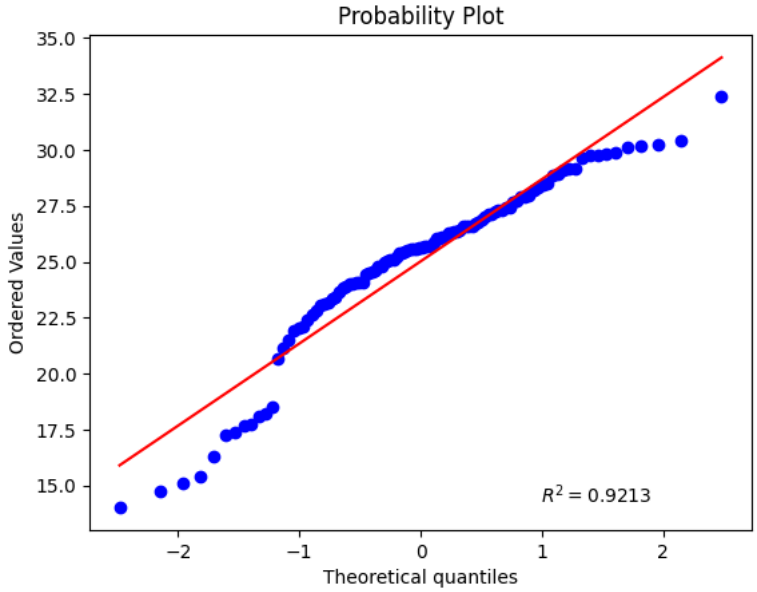

Relative Humidity vs Temperature - Sheet3
Average Temperature
Average Relative Humidity
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
30
40
50
60
70
80


Temperature vs Number of Adult Males - Sheet3
Number of Adult Males
Average Temperature
6
5
4
3
2
1
0
15.0
17.5
20.0
22.5
25.0
27.5
30.0
32.5

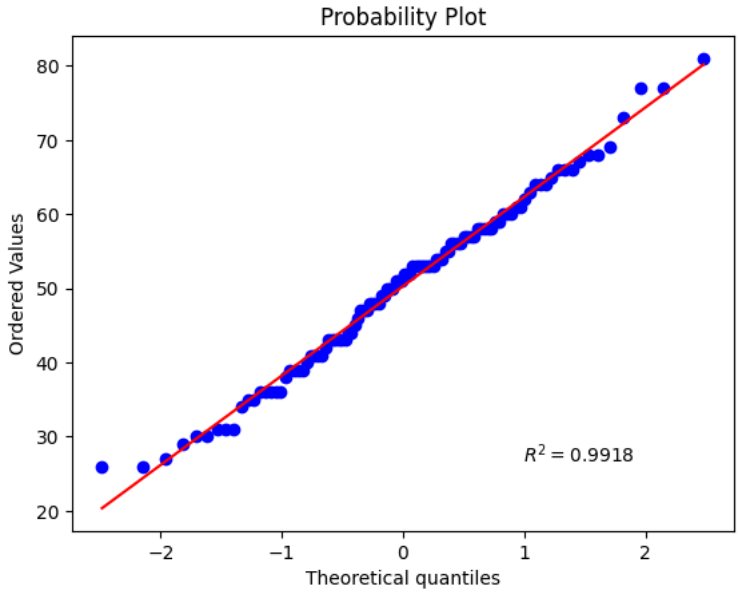

Relative Humidity vs Number of Adult Males - Sheet3
Number of Adult Males
Average Relative Humidity
6
5
4
3
2
1
0
30
40
50
60
70
80

Il probability plot viene utilizzato per valutare quanto bene i dati seguono una distribuzione teorica (la distribuzione normale). Se i punti nel plot seguono approssimativamente una retta, ciò suggerisce che i dati seguono la distribuzione specificata. Nel nostro caso sembra che i dati abbiamo una distribuzione normale.

#Temperature

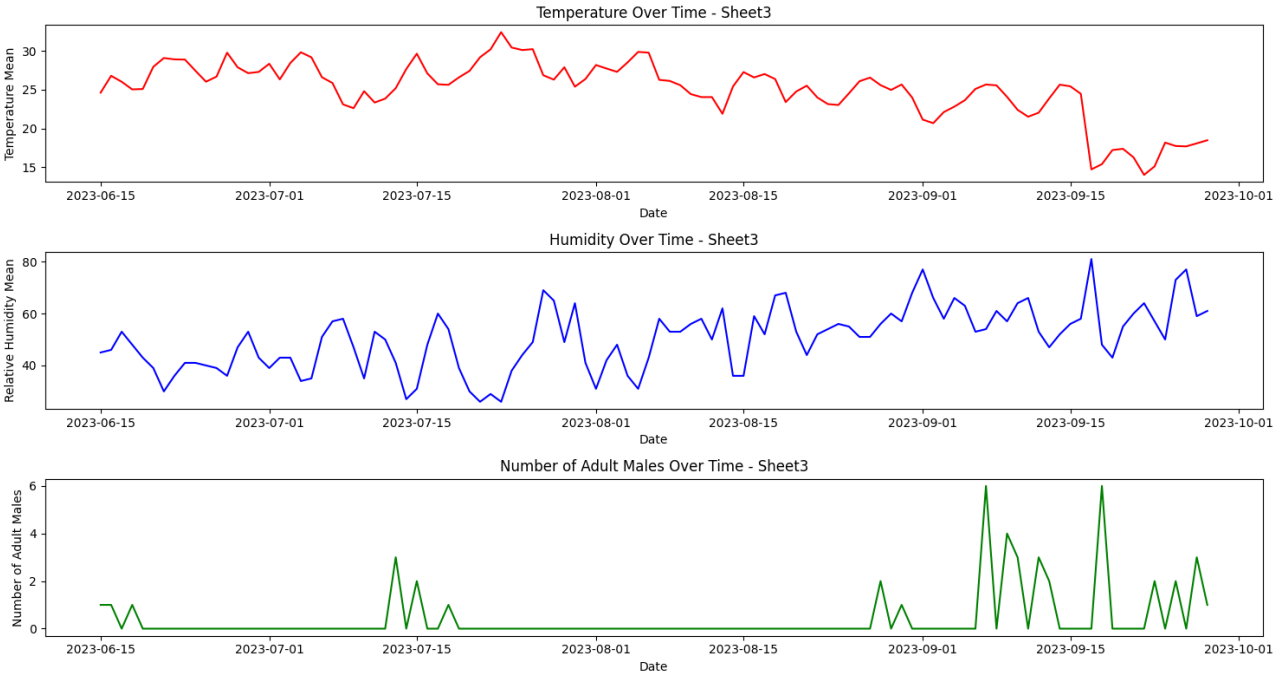


#Humidity

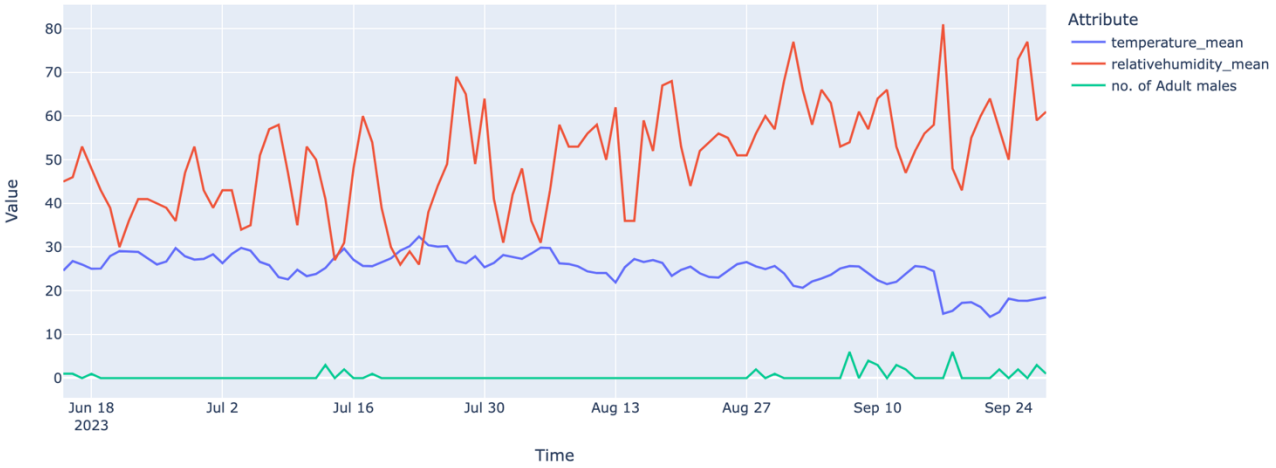


1.7 Analisi delle Serie Storiche

Nell'analisi, sono stati utilizzati grafici per visualizzare le variazioni temporali di temperatura, umidità e il numero di maschi adulti nel dataset Sheet3. La visualizzazione è stata suddivisa in tre grafici distinti, ognuno dedicato a una delle variabili in esame.



L'analisi delle serie temporali è essenziale per individuare tendenze, modelli stagionali o anomalie nei dati. Nel contesto specifico di questo studio, l'obiettivo è stato comprendere meglio come le variazioni temporali di temperatura e umidità possano influenzare la popolazione di maschi adulti, fornendo così informazioni utili sulla dinamica ambientale e la sua possibile influenza sulla popolazione in esame.



.

1.8 Analisi di Stazionarietà

L'analisi della stazionarietà è stata condotta utilizzando il test di Dickey-Fuller. La stazionarietà è un concetto chiave nelle serie temporali, indicando la costanza delle proprietà statistiche nel tempo. La funzione `test_stationarity` applica il test di Dickey-Fuller a diverse serie temporali, in particolare al numero di maschi adulti, alla temperatura media e all'umidità relativa media del dataset `s3`.

```
from statsmodels.tsa.stattools import adfuller

def test_stationarity(timeseries, significance_level=0.005):
    # Test di Dickey-Fuller
    print('Risultati del test di Dickey-Fuller:')
    df_test = adfuller(timeseries, autolag='AIC')
    df_output = pd.Series(df_test[0:4], index=['Test Statistic', 'p-value',
'#Lags Used', 'Number of Observations Used'])

    for key, value in df_test[4].items():
        df_output['Critical Value (%s)' % key] = value

    print(df_output)

    # Stampa se la serie è stazionaria o non stazionaria
    if df_output['p-value'] <= significance_level:
        print("\nLa serie temporale è stazionaria.\n")
    else:
        print("\nLa serie temporale non è stazionaria.\n")
```

Questo test restituisce una "Test Statistic" e un "p-value". Quando il p-value è inferiore al livello di significatività predefinito (generalmente 0.005), l'analisi suggerisce che la serie temporale è stazionaria, mentre un p-value superiore indica non stazionarietà. In base ai risultati ottenuti, è possibile concludere che le serie considerate sono non stazionarie.

```
# Testare la stazionarietà per la temperatura media e l'umidità relativa media
con soglia a 0.005
print("Analisi della stazionarietà per il numero di maschi adulti:")
test_stationarity(s3['no. of Adult males'])
```

```
Analisi della stazionarietà per il numero di maschi adulti:
Risultati del test di Dickey-Fuller:
Test Statistic                -1.748632
p-value                        0.406246
#Lags Used                     8.000000
Number of Observations Used   97.000000
Critical Value (1%)           -3.499637
Critical Value (5%)           -2.891831
Critical Value (10%)          -2.582928
dtype: float64

La serie temporale non è stazionaria.
```

```
print("Analisi della stazionarietà per la temperatura media:")
```

```
test_stationarity(s3['temperature_mean'])
```

```
Analisi della stazionarietà per la temperatura media:
Risultati del test di Dickey-Fuller:
Test Statistic                  -1.798042
p-value                          0.381434
#Lags Used                       2.000000
Number of Observations Used    103.000000
Critical Value (1%)             -3.495493
Critical Value (5%)             -2.890037
Critical Value (10%)            -2.581971
dtype: float64

La serie temporale non è stazionaria.
```

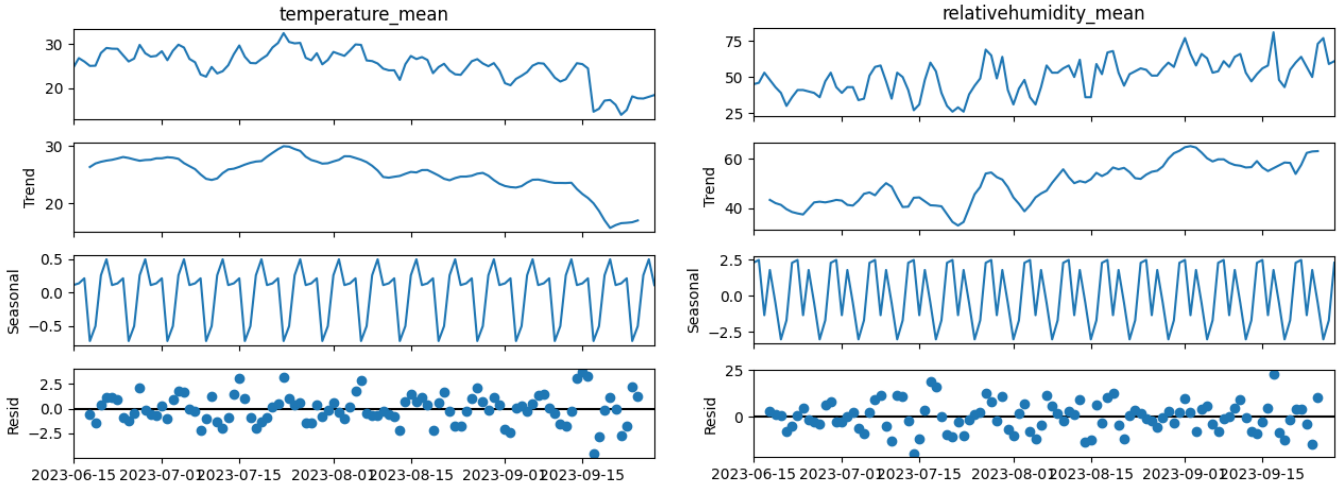
```
print("\nAnalisi della stazionarietà per l'umidità relativa media:")
test_stationarity(s3['relativehumidity_mean'])
```

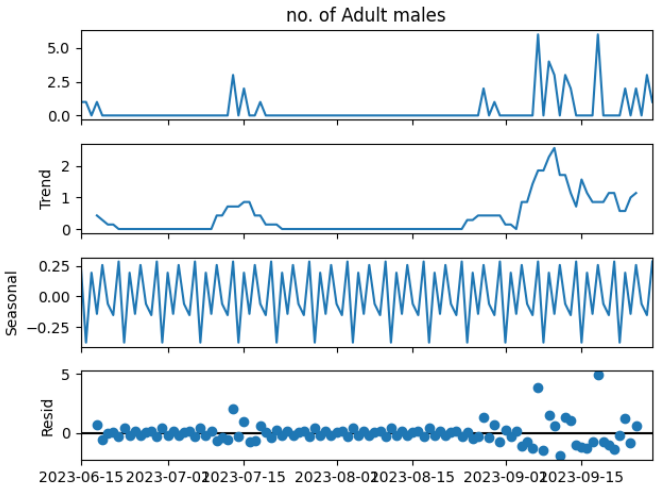
```
Analisi della stazionarietà per l'umidità relativa media:
Risultati del test di Dickey-Fuller:
Test Statistic                  -1.288532
p-value                          0.634402
#Lags Used                       8.000000
Number of Observations Used     97.000000
Critical Value (1%)             -3.499637
Critical Value (5%)             -2.891831
Critical Value (10%)            -2.582928
dtype: float64

La serie temporale non è stazionaria.
```

1.9 Analisi di Stagionalità

L'analisi della stagionalità è stata condotta utilizzando la libreria `statsmodels.tsa.seasonal`. Questa libreria include la decomposizione stagionale degli effetti. La funzione `seasonal_decompose` è stata applicata separatamente alle serie temporali corrispondenti all'umidità relativa media, alla temperatura media e al numero di maschi adulti nel dataset `s3`, utilizzando il modello additivo.

```
from statsmodels.tsa.seasonal import seasonal_decompose

humid_s_dec_additive = seasonal_decompose(s3.relativehumidity_mean, model = "additive")
temp_s_dec_additive = seasonal_decompose(s3.temperature_mean, model = "additive")
s_dec_additive = seasonal_decompose(s3['no. of Adult males'], model = "additive")
```





I grafici risultanti mostrano le componenti della decomposizione stagionale: trend, stagionalità e residui.
Il modello additivo rappresenta la stagionalità come una somma di effetti, consentendo di identificare eventuali modelli ciclici o periodici nei dati.
L'analisi della stagionalità ci offre dunque una visione dettagliata delle variazioni cicliche nei dati, aiutando a comprendere meglio i pattern ricorrenti nel tempo.

1.9 Analisi dell'Autocorrelazione totale e parziale

1.9.1 Analisi dell'Autocorrelazione totale

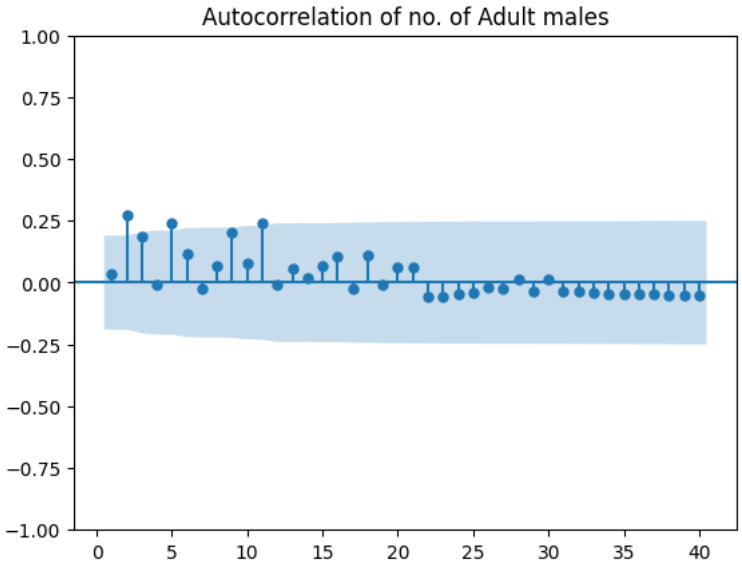
L'autocorrelazione è una misura statistica che valuta quanto i valori di una serie temporale siano simili a quelli ritardati. Essa fornisce informazioni sulla relazione tra un'osservazione in un determinato momento e le osservazioni in momenti precedenti, rivelando possibili modelli temporali o dipendenze nei dati.

I picchi nel grafico di autocorrelazione indicano la forza dell'autocorrelazione a un dato ritardo. Picchi più alti rappresentano una correlazione più forte tra i valori ritardati e quelli attuali. Una decrescita più lenta può indicare la presenza di una componente stagionale nella serie temporale.

Il codice utilizza la libreria `statsmodels.graphics.tsaplots` per plottare la funzione di autocorrelazione (ACF) per la serie temporale del numero di maschi adulti (``'no. of Adult males'`) nel dataset `s3`.

```
import statsmodels.graphics.tsaplots as sgt

sgt.plot_acf(s3['no. of Adult males'] , lags= 40 , zero =False , title =
"Autocorrelation of no. of Adult males") #do not include the actual value
plt.show()
```

Questo grafico non evidenzia particolari pattern da tenere in considerazione.



1.9.2 Analisi dell'Autocorrelazione parziale

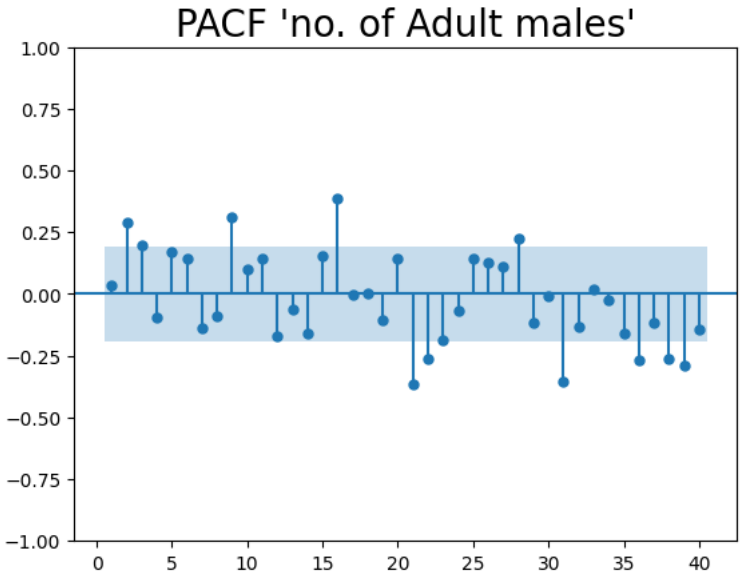
La Funzione di Autocorrelazione Parziale (PACF) misura la correlazione tra una serie temporale e le sue versioni ritardate, eliminando le variazioni già spiegate dai ritardi intermedi. Ad esempio, la PACF a un ritardo di 2 misurerebbe la correlazione tra la serie e la sua versione ritardata di due punti temporali, eliminando gli effetti dei ritardi 1.

Il grafico della PACF è utile per individuare eventuali correlazioni dirette tra una osservazione e le sue osservazioni ritardate, considerando solo gli effetti diretti e rimuovendo gli effetti intermedi.

Il codice utilizza la libreria `statsmodels.graphics.tsaplots` per plottare la Funzione di Autocorrelazione Parziale (PACF) per la serie temporale del numero di maschi adulti (``no. of Adult males'') nel dataset `s3`.

```
import statsmodels.graphics.tsaplots as sgt

sgt.plot_pacf(s3['no. of Adult males'] , lags= 40 , zero =False ,  method =
('ols') ) #do not include the actual value
plt.title("PACF 'no. of Adult males'" , size = 20)
plt.show()
```

Questo grafico ancora una volta non rileva pattern particolari a cui prestare attenzione.



Capitolo 2: Scelta del modello

La scelta del modello giusto è cruciale per garantire una rappresentazione accurata e predittiva dei dati. Basandoci sulle analisi precedenti, emerge l'importanza di selezionare il modello più appropriato in base alle caratteristiche specifiche delle serie temporali esaminate. La valutazione della stazionarietà, stagionalità, autocorrelazione e altre analisi descrittive forniscono preziose informazioni sulla struttura dei dati.

L'efficienza del modello è essenziale, e ciò implica che i coefficienti stimati devono essere significativamente diversi da zero in modo statistico. In altre parole, il modello più complesso deve dimostrare un miglioramento significativo rispetto a modelli più semplici, altrimenti la complessità aggiunta potrebbe non essere giustificata.

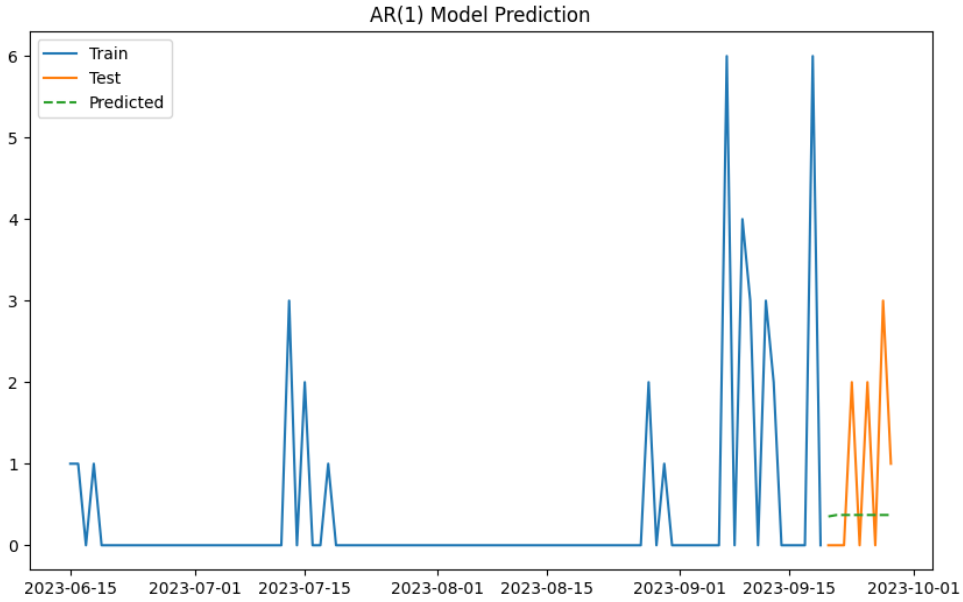
Allo stesso tempo, la parcimonia è altrettanto rilevante. Se un modello più semplice fornisce previsioni ragionevoli e i coefficienti sono statisticamente significativi, è preferibile mantenere la semplicità del modello. Questo principio sottolinea l'importanza di evitare eccessiva complessità quando modelli più parsimoniosi possono ottenere risultati simili.

Considerando queste considerazioni, abbiamo proceduto con la selezione dei seguenti modelli, cercando di bilanciare efficienza e parcimonia per garantire una rappresentazione accurata dei dati e ottenere previsioni attendibili.

2.1 AR

Il modello AutoRegressivo (AR) utilizzato è un AR(1), che considera solo il valore precedente della serie temporale per fare previsioni. Nel codice, il modello AR(1) viene addestrato sulla serie temporale "no. of Adult males". L'ordine (1, 0, 0) indica un termine AR di ordine 1 senza differenziazione.

Il modello utilizza il 90% dei dati per l'addestramento e il restante 10% per il forecasting. Non è possibile aumentare la complessità del modello, poiché il p-value associato al coefficiente AR è elevato, indicando che non è statisticamente significativo. Aumentare la complessità del modello non porterebbe a miglioramenti significativi.

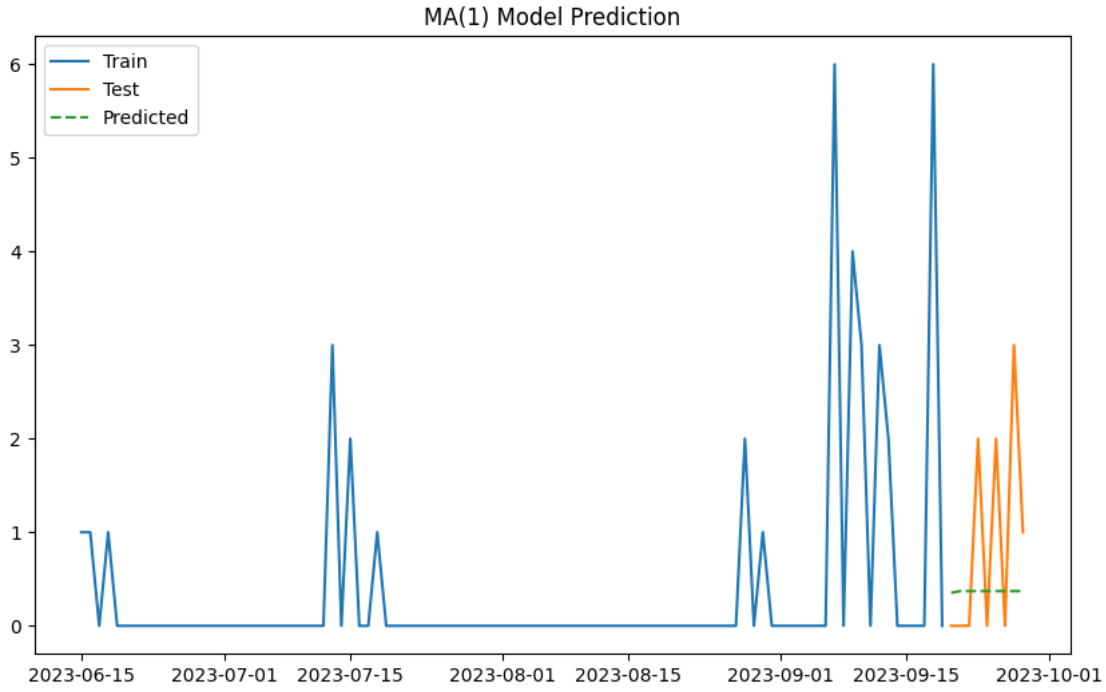


2.2 MA

```
model_ma_1 = ARIMA(s3['no. of Adult males'][1:],order = (0,0,1))

results_ma_1 = model_ma_1.fit()
results_ma_1.summary()
```

Per quanto riguarda il modello di media mobile (MA), specificamente il modello MA(1) implementato con order=(0,0,1), esso considera solo il termine di media mobile di ordine 1 nella previsione. In altre parole, il valore previsto al tempo t è influenzato solo dal termine di errore al tempo t-1. Anche in questo caso, il modello utilizza il 90% dei dati per l'addestramento e il 10% rimanente per il forecasting.

Tuttavia, i p-value associati ai coefficienti non superano il livello di significatività del 95%, indicando che i coefficienti non sono statisticamente significativi. Ciò implica che l'aggiunta di tali coefficienti non migliorerebbe in modo significativo le prestazioni del modello



2.3 ARMA

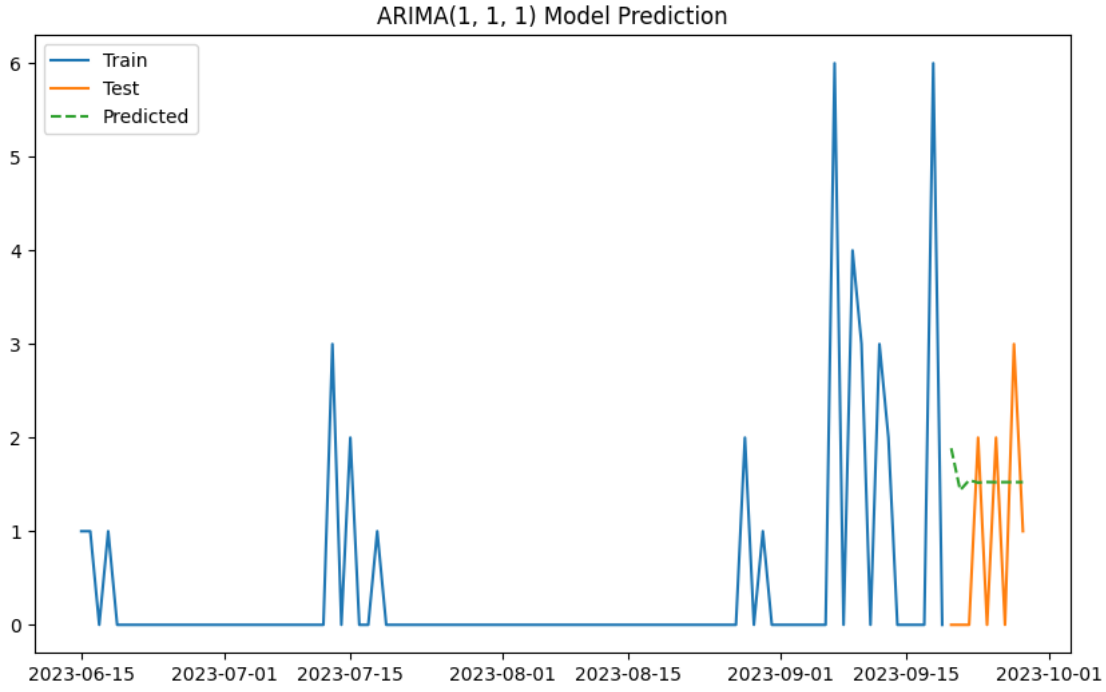
La mancata stazionarietà dei dati non ci ha permesso di prendere in esame questo tipo di modello.

2.3 ARIMA

```
model_ar_1_i_1_ma_1 = ARIMA(s3['no. of Adult males'], order=(1,1,1))
results_ar_1_i_1_ma_1 = model_ar_1_i_1_ma_1.fit()
results_ar_1_i_1_ma_1.summary()
```

Anche nel caso del modello ARIMA (1,1,1), che include una componente auto-regressiva di ordine 1, una differenziazione di ordine 1 e una componente di media mobile di ordine 1, il test del p-value è stato eseguito. Tuttavia, il risultato mostra che nessuno dei coefficienti del modello ha superato l'ipotesi del 95% di significatività statistica.

Questo significa che, secondo il test del p-value, nessuno dei parametri stimati nel modello ARIMA (1,1,1) è considerato statisticamente significativo. Di conseguenza, l'aggiunta di tali componenti non porterebbe a un miglioramento significativo nelle prestazioni del modello. In termini pratici, ciò suggerisce che un modello più semplice potrebbe essere preferibile, in quanto evita di introdurre complessità senza un beneficio statisticamente significativo.



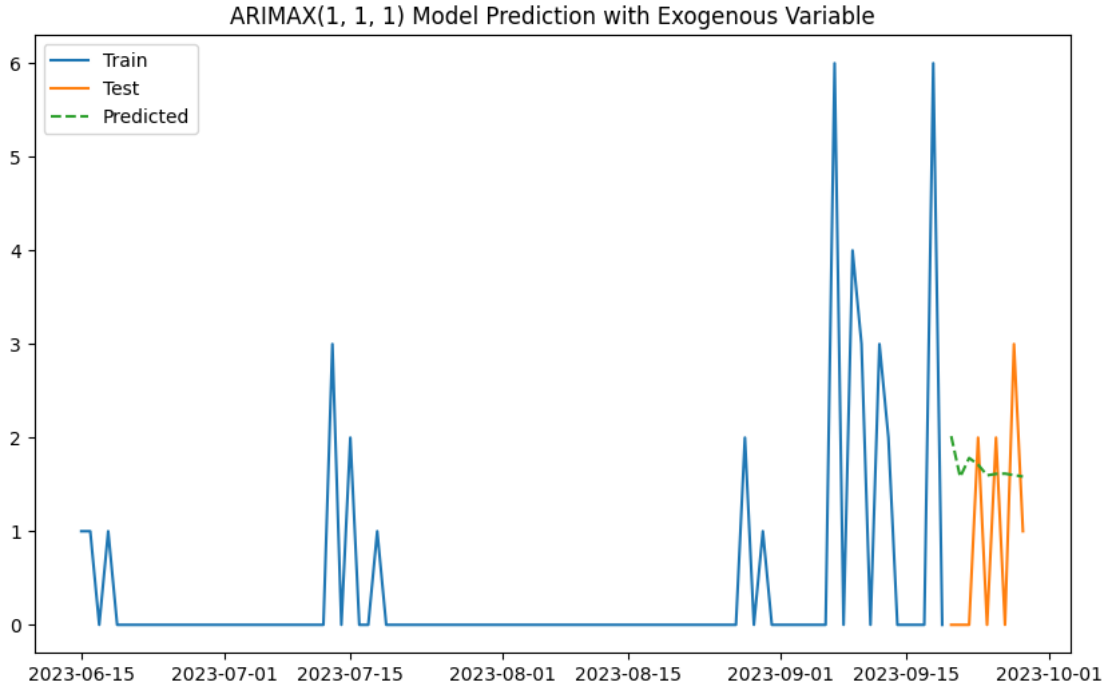
2.4 ARIMAX

```
model_ar_1_i_1_ma_1_Xspx = ARIMA(s3['no. of Adult males'], exog =
s3.temperature_mean, order=(1,1,1))
results_ar_1_i_1_ma_1_Xspx = model_ar_1_i_1_ma_1_Xspx.fit()
results_ar_1_i_1_ma_1_Xspx.summary()
```

Anche nel caso del modello ARIMAX (1,1,1) con una variabile esogena ('temperature_mean') è stata effettuata un'analisi del p-value [è stato effettuato anche con la variabile esogena 'relativehumidity_mean']. Tuttavia, i risultati indicano che nessuno dei coefficienti del modello è considerato statisticamente significativo al livello del 95%.

Questo suggerisce che, secondo il test del p-value, l'aggiunta della variabile esogena non contribuisce in modo significativo alla previsione del numero di maschi adulti. La mancanza di significatività dei coefficienti implica che il modello ARIMAX con nessuna delle due variabili esogene potrebbe essere la scelta ottimale in base ai criteri di significatività statistica.

In considerazione di ciò, ancora una volta potrebbe essere utile esplorare ulteriormente l'opzione di modelli più semplici.

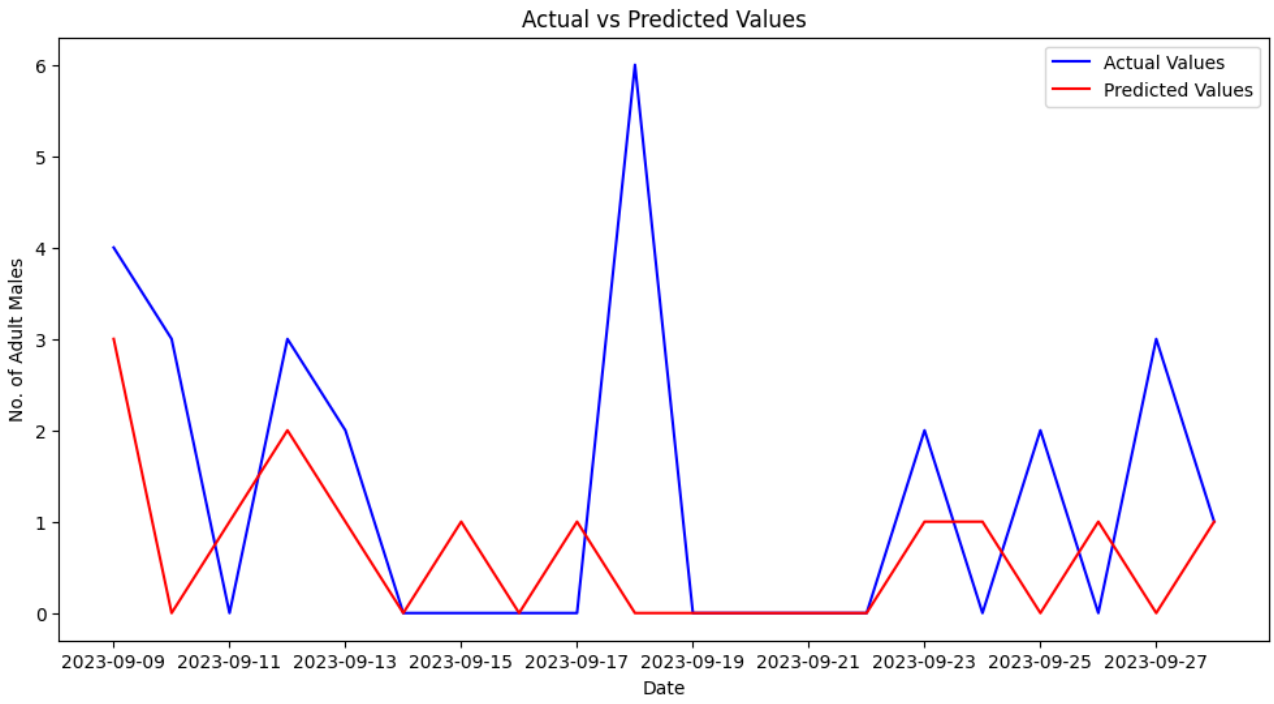


2.5 MLP

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error
df = s3

# Preprocessing and Creating Lagged Features
def create_lagged_features(df, n_lags):
    lagged_df = df.copy()
    for i in range(1, n_lags + 1):
        lagged_df[f'lag_{i}'] = lagged_df['no. of Adult males'].shift(i)
    lagged_df = lagged_df.dropna()
    return lagged_df

n_lags = 9  # Number of lags (can be tuned)
lagged_df = create_lagged_features(df, n_lags)

# Splitting Data
X = lagged_df[[f'lag_{i}' for i in range(1, n_lags + 1)]]
y = lagged_df['no. of Adult males']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
shuffle=False)

# Scaling Features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Definire il modello MLP con la regolarizzazione L2 (alpha) e relu
nell'ultimo strato
mlp = MLPRegressor(
      hidden_layer_sizes=(5, 30),
      activation='relu',
      alpha=0.001,
      max_iter=5000,
      random_state=42  # Per la riproducibilità dei risultati
    )

# Addestrare il modello
mlp.fit(X_train_scaled, y_train)

# Valutare il modello
y_pred = mlp.predict(X_test_scaled)

# Impostare tutti i valori negativi a 0
y_pred = np.maximum(0, y_pred)
```

```
# Calcolare il Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')

# Plotting the predicted values against the actual values
plt.figure(figsize=(12, 6))
plt.plot(y_test.index, y_test, label='Actual Values', color='blue')
plt.plot(y_test.index, y_pred.round(), label='Predicted Values',
color='red')
plt.title('Actual vs Predicted Values')
plt.xlabel('Date')
plt.ylabel('No. of Adult Males')
plt.legend()
plt.show()
```

Il modello utilizzato è un Multi-Layer Perceptron (MLP) implementato con la libreria scikit-learn. Mentre gli MLP sono noti per la loro flessibilità nell'affrontare complessità nei dati, i risultati ottenuti, purtroppo, non sono stati ottimali. Questo suggerisce che la relazione tra il numero di maschi adulti e le variabili di temperatura e umidità potrebbe essere più intricata di quanto possa essere catturata da un modello lineare.



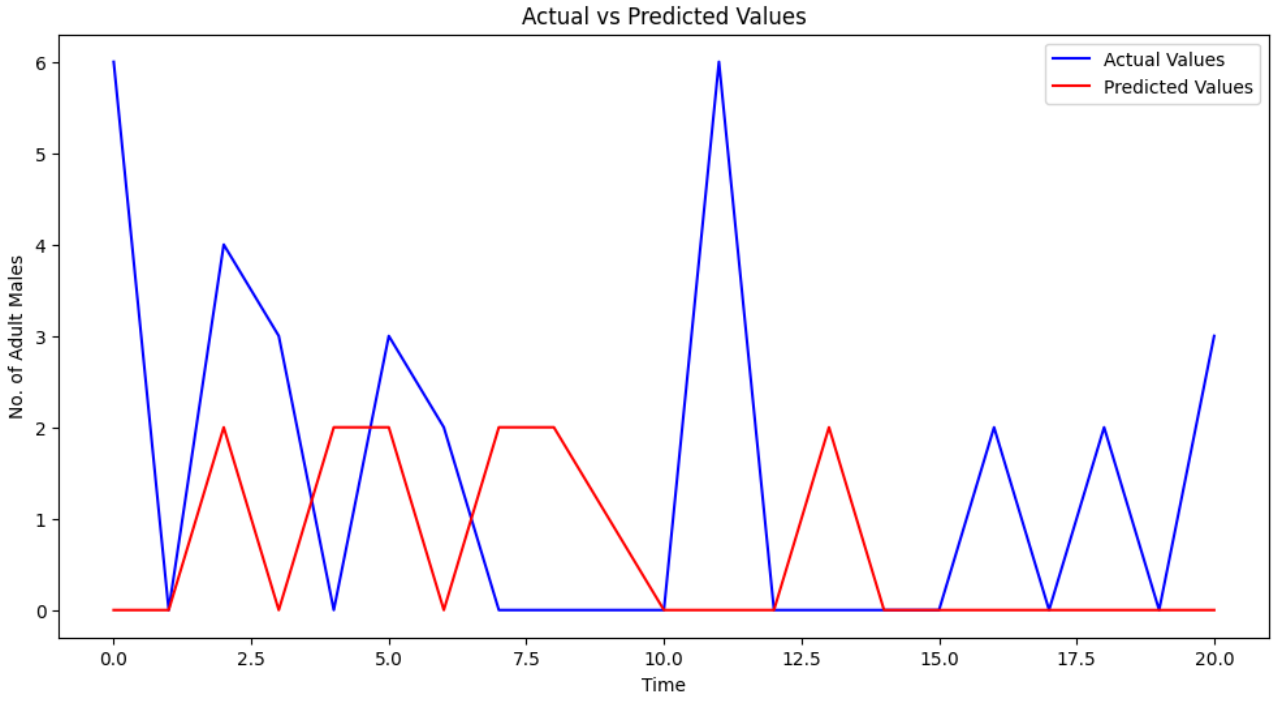
2.6 LSTM

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import LSTM, Dense
import matplotlib.pyplot as plt

# Load your data
# df = pd.read_excel('your_dataset.xlsx')

# Assuming 'no. of Adult males' is the target variable
# Assuming df is already loaded with your data

# Normalize features
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(df[['no. of Adult males']].values)

# Create lagged dataset
def create_dataset(data, look_back=1):
    X, Y = [], []
    for i in range(len(data)-look_back-1):
        a = data[i:(i+look_back), 0]
        X.append(a)
        Y.append(data[i + look_back, 0])
    return np.array(X), np.array(Y)

look_back = 3  # Number of lagged features
X, y = create_dataset(scaled_data, look_back)
X = np.reshape(X, (X.shape[0], X.shape[1], 1))  # Reshape for LSTM

# Split into train and test sets
train_size = int(len(X) * 0.8)
test_size = len(X) - train_size
X_train, X_test = X[0:train_size], X[train_size:len(X)]
y_train, y_test = y[0:train_size], y[train_size:len(y)]

# Define LSTM model
model = Sequential()
model.add(LSTM(100, return_sequences= True, input_shape=(look_back, 1)))
model.add(LSTM(150))
model.add(Dense(25))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam')

# Train the model
model.fit(X_train, y_train, epochs=2000, batch_size=16, verbose=1)

# Make predictions
```

```
y_pred = model.predict(X_test)

# Invert predictions
y_pred = scaler.inverse_transform(y_pred).round()

y_test = scaler.inverse_transform([y_test])



# Plotting the results
plt.figure(figsize=(12, 6))
plt.plot(y_test[0], label='Actual Values', color='blue')
plt.plot(y_pred[:,0], label='Predicted Values', color='red')
plt.title('Actual vs Predicted Values')
plt.xlabel('Time')
plt.ylabel('No. of Adult Males')
plt.legend()
plt.show()
```

Il modello utilizzato è una rete neurale ricorrente LSTM (Long Short-Term Memory), implementata attraverso la libreria Keras. Gli LSTM sono particolarmente adatti per modellare serie temporali complesse grazie alla loro capacità di catturare relazioni a lungo termine.

Tuttavia, nonostante le caratteristiche avanzate del modello, i risultati ottenuti sono migliori ma di certo non soddisfacenti.



2.7 Regressione Lineare

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import numpy as np

# Variabili indipendenti e dipendente
X = s3[['temperature_mean', 'relativehumidity_mean']]  # Variabili
indipendenti
y = s3['no. of Adult males']  # Variabile dipendente

# Divisione dei dati in set di addestramento e test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=0)

# Addestramento del modello di regressione lineare
model = LinearRegression()
model.fit(X_train, y_train)

# Previsione dei valori sul set di test e calcolo dell'errore quadratico
medio (RMSE)
y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred.round())
rmse = np.sqrt(mse)
rmse

# Preparazione dei dati di Sheet1 per la previsione (primi 20 record)
X_sheet1_for_prediction = s1[['temperature_mean',
'relativehumidity_mean']].head(20)

# Effettuare le previsioni utilizzando il modello addestrato
predicted_adult_males_sheet1 = model.predict(X_sheet1_for_prediction)

# Creare un DataFrame per visualizzare i risultati
predictions_sheet1_df = X_sheet1_for_prediction.copy()
predictions_sheet1_df['Predicted no. of Adult males'] =
predicted_adult_males_sheet1

# Visualizzazione delle prime 20 previsioni
predictions_sheet1_df['Predicted no. of Adult males'] =
predictions_sheet1_df['Predicted no. of Adult males'].round()

predictions_sheet1_df
```

Il modello utilizzato è una regressione lineare implementata attraverso la libreria scikit-learn. Mentre la regressione lineare è generalmente meno adatta per le serie storiche rispetto ai modelli specifici per il tempo, abbiamo esplorato questa opzione data la mancanza di successo con i modelli precedenti. Tuttavia, i risultati non sono stati ottimali, confermando la sfida di modellare il numero di maschi adulti in base alle sole variabili di temperatura e umidità.

Le previsioni sono state effettuate sui dati di Sheet1, ma purtroppo non hanno fornito risultati soddisfacenti. Ciò suggerisce che il modello lineare potrebbe non catturare adeguatamente le complessità dei dati e delle relazioni temporali sottostanti.

Date	temperature_mean	relativehumidity_mean	Predicted no. of Adult males
2022-01-01	11.22	77	2.0
2022-01-02	9.87	86	2.0
2022-01-03	9.33	79	2.0
2022-01-04	11.05	72	2.0
2022-01-05	10.17	73	2.0
2022-01-06	5.13	84	3.0
2022-01-07	3.89	77	3.0
2022-01-08	1.87	71	4.0
2022-01-09	1.40	84	4.0
2022-01-10	3.44	81	3.0
2022-01-11	4.19	66	4.0
2022-01-12	2.95	64	4.0
2022-01-13	1.33	71	4.0
2022-01-14	6.10	43	4.0
2022-01-15	7.76	58	3.0

Capitolo 3: Conclusione

In conclusione, nonostante un'approfondita analisi esplorativa dei dati (EDA) che ha evidenziato la stazionarietà, la stagionalità e altre caratteristiche rilevanti, l'applicazione di vari modelli di previsione non ha portato a un modello predittivo soddisfacente. Questo risultato deludente potrebbe essere attribuito alla scarsità di dati disponibili o alla mancanza di una forte correlazione tra gli attributi considerati e il target, nel nostro caso il numero di maschi adulti.

La complessità delle dinamiche sottostanti potrebbe non essere stata adeguatamente catturata dai modelli selezionati, che potrebbero essere stati limitati dalla dimensione ridotta del dataset o da altri fattori non considerati. Questa situazione evidenzia la necessità di una valutazione approfondita delle relazioni tra le variabili e l'importanza di considerare ulteriori fattori che potrebbero influenzare il fenomeno in esame.

In futuro, potrebbe essere utile esplorare modelli più avanzati, acquisire dati supplementari o includere nuove variabili per migliorare la comprensione e la previsione del numero di maschi adulti. L'analisi e la riflessione continue sulle sfide incontrate durante il processo di modellizzazione sono essenziali per informare future iterazioni e ricerche mirate al miglioramento delle previsioni.