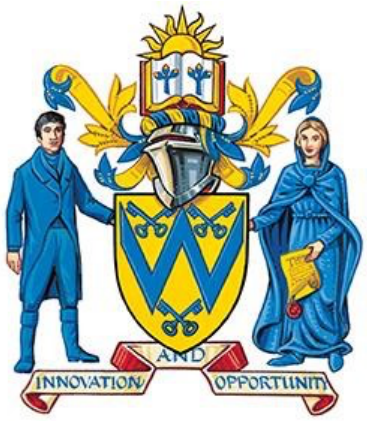
UNIVERSITY OF WOLVERHAMPTON

Application of Machine Learning in Predicting Taiga Goose Migration



Oloyede Festus, Oyebisi.2345336

MSc Data Science.

School of Engineering, Computing & Mathematical Sciences.

Project Supervisor: Dr. Andrew Gascoyne

Acknowledgement

I am forever grateful to Almighty God for granting me the strength, wisdom, and perseverance to complete this project. His guidance has been my constant source of inspiration throughout this journey.

I would also like to thank my family and my friends for their constant support, love motivation and belief in me during every moment I used on this project. You love and motivation kept me going even in the most challenging moments.

My heartfelt appreciation also goes to my supervisor Dr Andrew Gascoyne for his valuable advice, guidance and constructive feedback. His insights played a vital role in shaping the direction and quality of this project

To all who supported me in one way or another – Thank you.

Abstract

This research presents a data-driven approach to understanding and predicting the migratory patterns of the Taiga Goose (Anser fabalis) using machine learning techniques. With the increase in availability of large-scale GPS tracking data and environmental datasets, this project aims to classify migratory behaviours and forecast future migration directions. The study utilizes a cleaned and merged dataset combining geospatial tracking information with meteorological variables such as wind, temperature, and surface pressure.

The methodology includes unsupervised learning using K-Means clustering to categorize migratory behaviour, and predictive modelling through Random Forest, Gradient Boosting, and Long Short-Term Memory (LSTM) networks to estimate future geolocations. Random Forest was found to be the most effective, yielding a low root mean squared error (RMSE) and outperforming the other models. Feature importance analysis identified surface pressure and wind dynamics as key predictors of migration.

Additionally, the study was deployed through a Stream lit web application to allow interactive visualization and prediction of migration patterns. The application supports conservation efforts by offering insights into critical migratory paths and environmental triggers. Overall, this work contributes to ecological forecasting by integrating geospatial analysis, unsupervised clustering, and predictive modelling, supporting both scientific understanding and conservation strategies for migratory bird.

Contents

1 INTRODUCTION 1

1.1 Study Background 2

1.2 Problem Statement 3

1.3 Aim and Objectives 3

1.4 Research Question 4

1.5 Research Justification 4

1.6 Research Scope 5

1.7 Methodology Choice 5

1.8 Key Terms Definition 6

1.9 Dissertation Structure 7

2 LITERATURE REVIEW 8

2.1 Birds Migration: Biological Relevance and Environmental Perspective 8

2.2 Tracking Technologies and Data Acquisition 9

2.2.1 GPS Telemetry and Satellite Tracking 9
2.2.2 Radar and Citizen Science 10

2.3 Applications of Machine Learning in Migration Research 10
2.3.1 Unsupervised Clustering for Pattern 10
2.3.2 Predictive Modelling of Migration Direction 12
2.3.3 Hybrid and Ensemble Methods 12

2.4 Theoretical Review 12
2.4.1 Optimal Migration Theory 13
2.4.2 Flyway Theory 13

2.5 Challenges Facing Research and Gaps 14

3.1 Data Collection 16

3.2 Importing, Loading and Merging the Datasets 16

3.3 Data Preprocessing 18

3.4 Exploratory Data Analysis 20

3.6 Unsupervised Clustering Analysis 25

3.7 Predictive Model Development 26

3.8.1 27

3.8.2 32

4 FINDINGS 34

4.1 Descriptive Analysis 34

4.2 Correlation Result 38

4.3 Model Results 39

4.4 Feature Importance Result 41

5 DISCUSSION 43

5.1 Migration Path of The Taiga Goose 43

5.2 Classification of Migratory Patterns Using K-Means Clustering **44**

5.3 Development of Predictive Models for Migration Direction **45**

5.4 Factors Influencing the Taiga Goose Migration **47**

5.5 Discussion Findings with Theoretical Frameworks **48**

5.6 Conclusion **49**

6 CONCLUSION **50**

6.1 Recommendations **51**

6.2 Contribution to Knowledge **52**

6.3 Limitations **53**

6.4 Implications for Future Research **53**

7 REFERENCES **55**

8 APPENDICES **61**

APPENDIX 1 JUPYTER NOTEBOOK OUTPUT **62**

APPENDIX 2: STREAMIT APPLICATION CODE **78**

Table of Figures

Fig 1.1. Research and objective concept map.

Figure 2.1: A two-dimensional map showing the major migratory flyways and the East Atlantic Flyway used by the Taiga Goose.

Figure 2.2: Conceptual representation of the trade-offs in energy, time, and risk in bird migration (Optimal Migration Theory).

Figure 2.3: Illustration of the Flyway Theory and the routes associated with different migratory paths.

Figure 3.1: Flowchart showing the research design and methodology structure.

Figure 3.2: Overview diagram of the predictive model development stages.

Figure 3.3: Structure of the Random Forest model with feature input and output predictions.

Figure 3.4: Screenshot of the Streamlit application EDA snapshot view.

Figure 3.5: Screenshot of the Streamlit interactive map with toggle features.

Figure 3.6: Screenshot showing prediction per bird feature on the Streamlit dashboard.

Figure 3.7: Screenshot comparing predicted versus actual migration values in the Streamlit application.

Figure 3.8: Histogram showing the distribution of latitude prediction errors.

Figure 3.9: Histogram showing the distribution of longitude prediction errors.

Figure 3.10: Scatter plot of actual vs. predicted latitude values.

Figure 3.11: Migration heatmap showing frequently used geographical zones.

Figure 3.12: Speed variation across clusters and timestamps.

Figure 3.13: Directional movement variations visualized per bird cluster.

Figure 3.14: Altitude distribution among migratory birds.

Figure 3.15: Heading direction patterns captured from sensor data.

Figure 3.16: Clustered radar points based on wind vector components.

Figure 3.17: Combined screenshot of Streamlit mapping outputs and prediction table.

LIST OF ABBREVIATIONS AND ACRONYMS

Abbreviation/Acronym	Full Meaning
AI	Artificial Intelligence
CSV	Comma-Separated Values
ECMWF	European Centre for Medium-Range Weather Forecasts
EDA	Exploratory Data Analysis
ERA5	ECMWF Reanalysis 5th Generation
GPS	Global Positioning System
HDOP	Horizontal Dilution of Precision
IQR	Interquartile Range
KNN	K-Nearest Neighbours
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Squared Error
NetCDF	Network Common Data Form
RF	Random Forest
RMSE	Root Mean Squared Error
SHAP	Shapley Additive explanations
SVM	Support Vector Machine
TOC	Table of Contents

Abbreviation/Acronym	Full Meaning
UI	User Interface
U10	Eastward 10-meter Wind Component
V10	Northward 10-meter Wind Component

LIST OF APPENDICES

APPENDIX 1 JUPYTER NOTEBOOK OUTPUT 62

APPENDIX 2: STREAMIT APPLICATION CODE 78

1 INTRODUCTION

This study examines the use of machine learning methods to predict and understand the migratory patterns of the Taiga Goose (Anser fabalis) with complex, long-distance migrations. The paper considers the increasingly abundant large-scale tracking and environmental information available to harness machine learning to enable a deeper understanding of the spatial and temporal patterns of the species' migration. The analysis starts by mapping and identifying the Taiga Goose's migratory route based on historical GPS tracking information. Then, the analysis utilizes K-Means clustering to group different migratory patterns, yielding a better understanding of the species' annual migratory groupings. Lastly, supervised machine learning models are trained on a variety of features such as environmental factors, flight characteristics, and geographic locations to forecast migratory directions. The models are tested for accuracy and performance in predicting upcoming migratory patterns.

An important part of the study is determining the key factors that drive Taiga Goose migration. By correlation analysis and feature importance, the study shows which environmental or temporal factors most affect migratory behaviour. The integration of clustering, predictive modelling, and factor analysis provides a unified framework by which to understand the processes of avian migration from a fact-based viewpoint.

1.1 Study Background

Migration is a vital biological phenomenon seen among several species of animals, especially birds (Bauer et al., 2019). Taiga Goose (Anser fabalis) is a key subject of study in ecology as well as conservation efforts owing to its long-distance seasonal migrations through a variety of geographical regions (Newton, 2010; Guilford et al., 2011). With changing environmental conditions attributable to climate change as well as anthropogenic changes, the migratory patterns of such species are becoming highly dynamic as well as uncertain (Runge et al., 2015). Conventional methods of tracking as well as forecasting migrations involving marking as well as field observations tend to be energy-intensive, time-consuming, and limited with respect to scalability (Bridge et al., 2011). Recent developments in the field of data science, especially those related to Machine Learning (ML), have provided novel opportunities for the effective analysis of large volumes of space-time data on animal movement (Dodge et al., 2014; Abrahm's et al., 2019).

Machine Learning has increasingly become a valuable resource for identifying patterns in large datasets, acting as a key tool for understanding the multidimensional character of animal migration (Abrahm's et al., 2019; Bauer et al., 2019). The dataset comprises location coordinates, temporal indicators, environmental factors, and physiological parameters obtained through GPS collars, satellite tracking, and remote sensing tools. The use of ML methods to work with such data not only provides a deeper understanding of the behaviour of animal migrations but also facilitates the creation of models that can be used for the management of wildlife, the conservation of biodiversity, and the formulation of policies.

When considering the Taiga Goose, knowledge of migratory routes and behaviours is important because the bird is a species of conservation interest in certain areas. Forecasting migratory directions and factors that affect them could aid in the conservation of habitats, the prevention of human- wildlife conflict, and the avoidance of harm to ecosystems (Runge et al., 2015). The use of cluster analysis methods like K-Means facilitates the grouping of migratory patterns, while predictive modelling can project upcoming directions and changes in migratory patterns. This multi-disciplinary approach that integrates ecology, analytical methods, and artificial intelligence can revolutionize the conventionally adopted ornithological work as well as enhance conservation biology decision-making.

1.2 Problem Statement

Precise prediction and classification of migratory behaviour in bird species are a major challenge considering the intricate, non-linear interplay of biological, environmental, and climatic factors (Runge et al., 2015; Wilcove, & Wikelski, 2008). In the Taiga Goose, literature tends to be poor in high-resolution predictive information that can practically be used for proactive conservation as well as management of habitats. Most of the conventional methods are based on linear models that are incapable of capturing the complex behaviours exhibited by such birds under different environmental stresses.

Furthermore, although information on Taiga Goose migrations is increasingly being made available through GPS telemetry, satellite imagery, and remote sensing, effective tools for analysing and gaining actionable recommendations from the information are not well used. The lack of automated systems that can learn from past patterns and generalize projections to new situations eliminates the potential for data-driven management (Costa, 2019). Unless advanced computational methods such as the use of machine learning are incorporated, chances to detect key migration corridors, unravel movement triggers, and prevent ecological risks could be lost (Runge et al., 2015).

Accordingly, a pressing requirement is for a holistic as well as smart framework exploiting machine learning for classifying, forecasting, and interpreting the migratory patterns of the Taiga Goose. This can provide a means to close the loop from raw data to ecological knowledge so that better-informed decision processes may be made, both in a scientific as well as a policy framework.

1.3 Aim and Objectives

The purpose of the current study is to explore how machine learning methods can be used for predicting the Taiga Goose's migratory patterns. To accomplish the objective, the study has the following specific objectives:

1. To develop an interactive visualisation to predict the Taiga Goose migration pattern and direction using a Predictive Model.
2. To classify Taiga Goose's migratory patterns employing the K-Means clustering algorithm.
3. To develop models for predicting directions of migration based on past and environmental information.

4. To identify the most important factors affecting Taiga Goose migration based on feature importance analysis and correlation analysis.

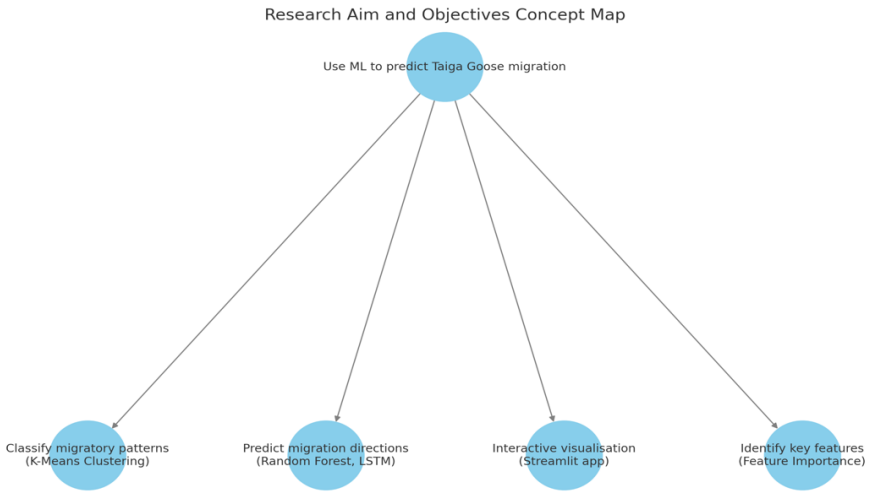


Fig 1.1 research and objective concept map

1.4 Research Question

The following questions are raised to guide the research process:

1. What are the graphical visualisations of typical migration routes taken by the Taiga Goose?
2. How does K-Means clustering classify the Taiga Goose's migratory patterns?
3. How do machine learning models predict the future trajectory of the Taiga Goose migration?
4. What factors impact the Taiga Goose's migratory patterns?

1.5 Research Justification

This research is important in several ways. It is one of several recent studies employing machine learning techniques in the field of wildlife and ecological studies. By presenting a methodological framework for examining migratory patterns through cluster and prediction techniques, the study advances the use of data-based methods for ecological status assessment.

Secondly, the use of unsupervised and supervised learning methods in this investigation illustrates the applied merit of artificial intelligence in addressing actual environment-related issues. This involves helping conservationists as well as ecologists make informed choices on the conservation of migratory bird habitats. Additionally, by determining what variables affect Taiga Goose migration, this research contributes to the understanding of the environmental

stresses such as climate, loss of habitat, and human impacts faced by such birds. The findings can be applied to devise adaptive management strategies as well as early warning systems that minimize the negative impacts of environmental changes on migratory animals. Lastly, the study offers a reproducible analytical workflow for other such avian studies, hence promoting the general field of computational ecology.

1.6 Research Scope

- This study focuses on applying machine learning algorithms to Taiga Goose migration data analysis.
- The analysis is limited to the use of GPS tracking datasets of the Taiga Goose across several migratory seasons.
- Use of K-Means clustering to determine unique migratory behaviour patterns.
- Use of predictive models (for example, Random Forest, Support Vector Machines, or Neural Networks) to predict directions of future migration.
- Study of environmental, climatic, and biological factors influencing migratory behaviour.
- The analysis does not include fieldwork or new migrant data collection; it is based solely on the available, public, or licensed datasets.
- The analysis is also limited to the Taiga Goose species, and generalization to other migratory bird species is outside the framework of this project.

1.7 Methodology Choice

The analysis will employ a data-driven approach utilizing a number of important stages. First, raw migratory data gleaned from GPS tracking units will be accumulated and pre-processed for missing value handling and scale normalization, as well as converting temporal information into a usable format. Geospatial mapping technology will be utilized for visualizing migratory routes as well as identifying seasonal patterns.

Secondly, K-Means clustering, a type of unsupervised machine learning, will be applied to classify migration patterns by spatial and temporal characteristics. This will enable the determination of different phases of movement, including departure, stopovers, and arrival.

Models will then be built in a supervised setting by leveraging factors such as geographic coordinates, elevation, temperature, wind speed, and vegetation index. The models will be trained and tested on historical patterns and variables to forecast the likely migratory course. Lastly, feature importance techniques will be done based on the best model performance values used for measurement.

1.8 Key Terms Definition

Machine Learning (ML): A form of artificial intelligence that empowers systems to improve their performance on their own by learning from data.

Migration: It refers to the periodic movement of animals from one area or environment to another, normally for mating or feeding.

Taiga Goose: A migratory bird species (Anser fabalis) which breeds in boreal habitats and overwinters in temperate regions.

K-Means Algorithm: A machine learning, non-supervised statistical method of grouping data into clusters as a function of feature similarity.

Predictive modelling: The practice of generating models based on past information to forecast upcoming events.

Geospatial Data: Data that contains location information, commonly presented as coordinates like latitude and longitude.

Feature Importance: A method that can be utilized to determine which input features are most responsible for a model's predictions.

1.9 Dissertation Structure

The structure of this study is divided into five chapters:

Chapter One - Introduction: Introduces the background, problem statement, aim, objectives, significance, and scope of the study.

Chapter Two - Literature Review: Summarizes pertinent academic and technical literature regarding avian migration, machine learning methods, and predictive modelling in environmental situations.

Chapter Three - Methodology: Outlines the study design, sources of data, pre-processing steps, machine learning models utilized, and measure of evaluation.

Chapter Four - findings: Summarizes and explains findings derived from clustering analysis and predictive modelling.

Chapter Five - Discussion of Findings: Discuss the findings with pre-existing studies Chapter Six - Conclusion and Recommendations: Summarise main findings, outlines the study's limitations, and provides suggestions for practical application as well as recommendations for further study.

2 LITERATURE REVIEW

This chapter provides a synthesis of the literature related to the prediction of bird migratory behaviours, with particular emphasis on the Taiga Goose. It is divided into *several* sections. First, it provides a general overview of bird migration *to* place the biological and ecological importance of the phenomenon into context. Second, the chapter reviews the development of tracking technologies and methods of collecting data that have improved knowledge of migratory movements. Third, it discusses in depth the use of machine learning (ML) methods in ecological as well as migratory research, including both the use of cluster analysis methods and methods of predictive modelling. The synthesis then contextualizes these information-based methods concerning existing theories, such as Optimal Migration Theory and Flyway Theory, discusses the gaps in research as well as challenges that remain, and lastly provides a framework for the application of ML to predict Taiga Goose migration along with directions for future research.

2.1 Birds Migration: Biological Relevance and Environmental Perspective

Bird migratory behaviour is a sophisticated, adaptive action motivated by environmental signals, food supply, and reproductive needs. Migratory birds migrate long distances from their breeding to their non-breeding grounds to capture seasonal peaks in food supply as well as favourable climate conditions (Berthold, 2001; Newton, 2008). The Taiga Goose (Anser fabalis), which travels extensive tracts of Eurasia, is one of such species whose migratory route is a key indicator of ecosystem condition as well as environmental modification. Not only does their migratory route mirror the climatic adaptive responses of the birds, but regional biodiversity and conservation strategies are also affected.

Increasingly, research has highlighted the interactions between intrinsic factors such as genetic programming and states of physiology and extrinsic factors such as climate patterns and changes to the environment in influencing migratory tactics (Helm et al., 2009). In the example of the Taiga Goose, researchers have accounted for how changes in temperature, rainfall, and

land use can result in changes in migratory timing as well as route selection. These processes highlight the value of predictive models that are capable of merging intricate, multidimensional data to predict patterns of migration under changing environmental processes.

2.2 Tracking Technologies and Data Acquisition

Tracking technologies

The precise tracking of migratory birds is the basis of understanding their movement ecology. Bird ringing (banding) has previously generated preliminary information but was hindered by poor spatial and temporal resolutions. Recent advances in technology have dramatically improved bird migration monitoring through instruments like Global Positioning System (GPS) telemetry, satellite tracking, and radar-based technology.

2.2.1 GPS Telemetry and Satellite Tracking

Telemetry by GPS has become the gold standard for capturing high-resolution spatial-temporal information on the movements of individuals. Birds can be equipped with devices that transmit exact geographic coordinates at short intervals, thus capturing fine-grained movement paths (Bridge et al., 2011). In species like the Taiga Goose, whose travels are thousands of *kilometres*, fine-grained tracking makes it possible to map *flyways* as well as identify key stopover locations. Satellite tracking carries these tools further by offering global coverage with trade-offs in position accuracy as well as expense (Meyburg & Fuller, 2007).

Even though these systems are advantageous, they are limited by factors like the weight of the devices, the life of the batteries, and the number of subjects that can be practically tagged. Therefore, although such techniques provide valuable information for interpreting individual-based migratory dynamics, they are unlikely to capture the population-wide patterns required for accurate *modelling*.

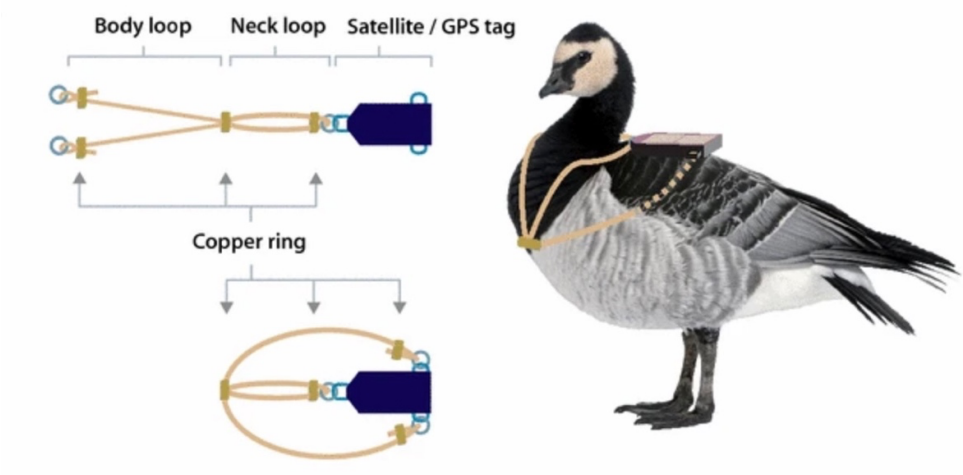


Fig. *2.1*: A schematic design of the *GPS tag* used in tracking a bird

2.2.2 Radar and Citizen Science

Radar systems, both avian radars and meteorological radars, represent a complementary method by tracking the movements of large flocks continuously across extensive regions (Hebblewhite & Haydon, 2010). Radars can pick up flocks in real time and have played a key role in measuring migrant intensity and timing. Radar information tends to be complex to process to eliminate non-biological clutter as well as to disambiguate species-specific returns.

Citizen science projects like eBird have enriched the pool of available information. Millions of observations submitted by birders and ornithologists worldwide give spatial and temporal coverage to complement telemetry and radar observations. Notwithstanding issues of data quality as well as observer bias, citizen science observations support large-sample statistical analysis of migrations as well as being used as input in successful applications of machine learning algorithms to species distribution and abundance prediction (Sullivan et al., 2014).

2.3 Applications of Machine Learning in Migration Research

Machine learning has revolutionized ecological forecasting as it allows for the combination of enormous amounts of heterogeneous data *to* reveal concealed patterns as well as create predictive models. Migration studies are especially suitable for ML due to the ability of the latter to work with non-linear, multi-variable relationships present within biological systems.

2.3.1 Unsupervised Clustering for Pattern

Unsupervised learning algorithms, like K-Means clustering, have been used to categorize patterns of movement into similar movement trajectories or *behavioural* modes without using

pre-*labelled* data. For the Taiga Goose, K-Means clustering can be utilized to determine different migratory routes or *behavioural* phases (e.g., departure, stopover, arrival) based on GPS and radar data. These clusters can indicate natural divisions within the migratory process, thus supporting conservation policy making as well as focused management action.

Clustering algorithms are useful for dimensionality reduction in multivariate data sets as well as for deriving interpretable clusters corresponding to varying migratory *behaviours*. Care should be exercised in choosing the number of clusters as well as the distance measure, as this introduces considerable bias in the outcomes. Experiments using the K-Means algorithm on avian migrations have indicated its efficacy in identifying migratory corridors as well as interannual variability in movement patterns.

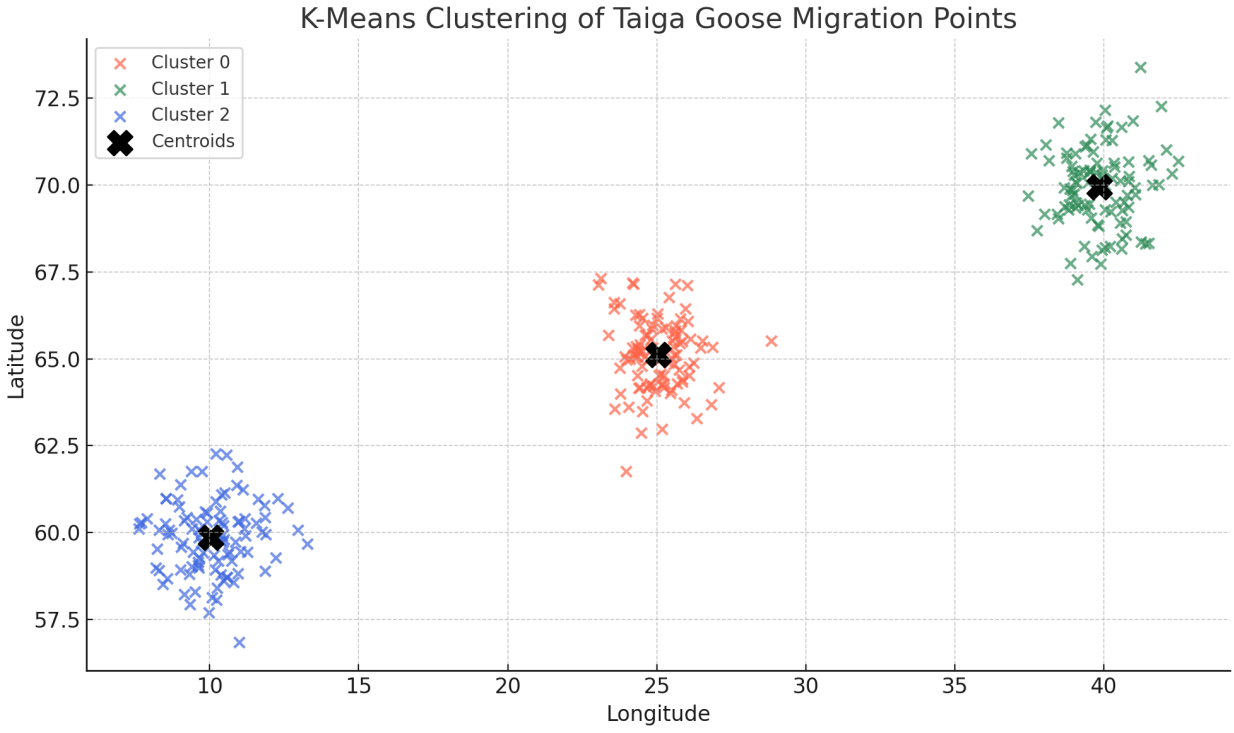


Fig 2.2 Illustration of K-means Clustering applied to simulated GPS migration data of Taiga Goose. The scatter plot shows three behavioural clusters representing different migratory phases. The blue clusters is the arrival phase, the red clusters the stopover phase and green cluster the departure phase.

2.3.2 Predictive Modelling of Migration Direction

Supervised learning methods have been used heavily in devising models that predict direction as well as the intensity of migration. Regression models, decision trees, random forests, as well as artificial neural networks have been used in attempts at future position prediction based on past movement as well as environmental variables. For example, methods that combine the predictions of multiple models tend to provide better predictive capability as they capture various facets of the movement (Lippert et al., *2024.)*

Key predictors in such models are weather variables like wind speed, temperature, and precipitation, directly affecting flight energetics as well as route choice. Using historical GPS as well as radar data for model calibration, scientists *can* produce predictions for direction as well as intensity of migrations with higher accuracy. For the Taiga Goose, such predictive models can be used for identifying key flyways as well as environmental thresholds for initiating migrations.

2.3.3 Hybrid and Ensemble Methods

Recent advances in hybrid *modelling* have aimed at marrying conventional, theory-based models with data-driven machine learning methodologies. These models take advantage of established ecological theory but employ ML algorithms for learning correction terms or sub-models directly from data (Hooten et al., 2017). For instance, hybrid models have been constructed that are based on principles of fluid dynamics combined with recurrent neural networks for *modelling* migration as an ongoing movement process. These models not only provide enhanced predictive power but also offer increased interpretability due to their embedding of ML components within ecological principles. Ensemble methods, in which multiple individual models' predictions are combined, have been effective as well. By averaging across several models, ensembles can reduce the threat of overfitting and offer more reliable predictions. Such methods are particularly important in ecological applications, where variability within the data as well as in uncertainty are major issues.

2.4 Theoretical Review

A grounding in ecological theory is advantageous in incorporating machine learning into the analysis of migration. Optimal Migration Theory and Flyway Theory are two important theoretical frameworks that guide this research.

2.4.1 Optimal Migration Theory

Optimal Migration Theory suggests that migratory birds take routes where energy costs, minimizing time, and risk are in equilibrium (Alerstam & Lindström, 1990). Optimal Migration Theory gives us reasons why birds may choose certain routes in certain environmental conditions. For example, it is known that Taiga Geese will modify their flight routes based on wind patterns and habitat availability while optimizing their flying efficiency as well as minimizing risk due to predators. By adding these trade-offs in the prediction space of ML models, the researchers can limit the prediction space for ecologically plausible outcomes. In this manner, theoretical predictions act as priors or constraints on the learning process, making the models more reliable.

2.4.2 Flyway Theory

Flyway Theory refers to the substantial, comparatively stable flyways that birds employ for their annual migrations (Berthold, 2001). Flyways are influenced by geographical, climatic, and historic aspects, and act as spatial templates that can be utilized in predictive models. In the case of the Taiga Goose, which is mostly on the East Atlantic Flyway, it is significant to understand the spatial boundaries and environmental aspects of this flyway. ML models that include flyway data can make better predictions as regards the direction of migratory movement by emphasizing the *regions* in which birds are most likely to migrate. This spatial contextualization enhances the interpretability of predictions as well as aligns predictions with established migrant *behaviours*. *This* concept is supported by Optimal Migration Theory, which emphasizes that migratory birds make strategic decisions that balance minimizing energy, time, and risk. As shown in Figure **2.3**, birds may choose different migratory paths based on these trade-offs, with flyways acting as ecological corridors that facilitate such decisions.

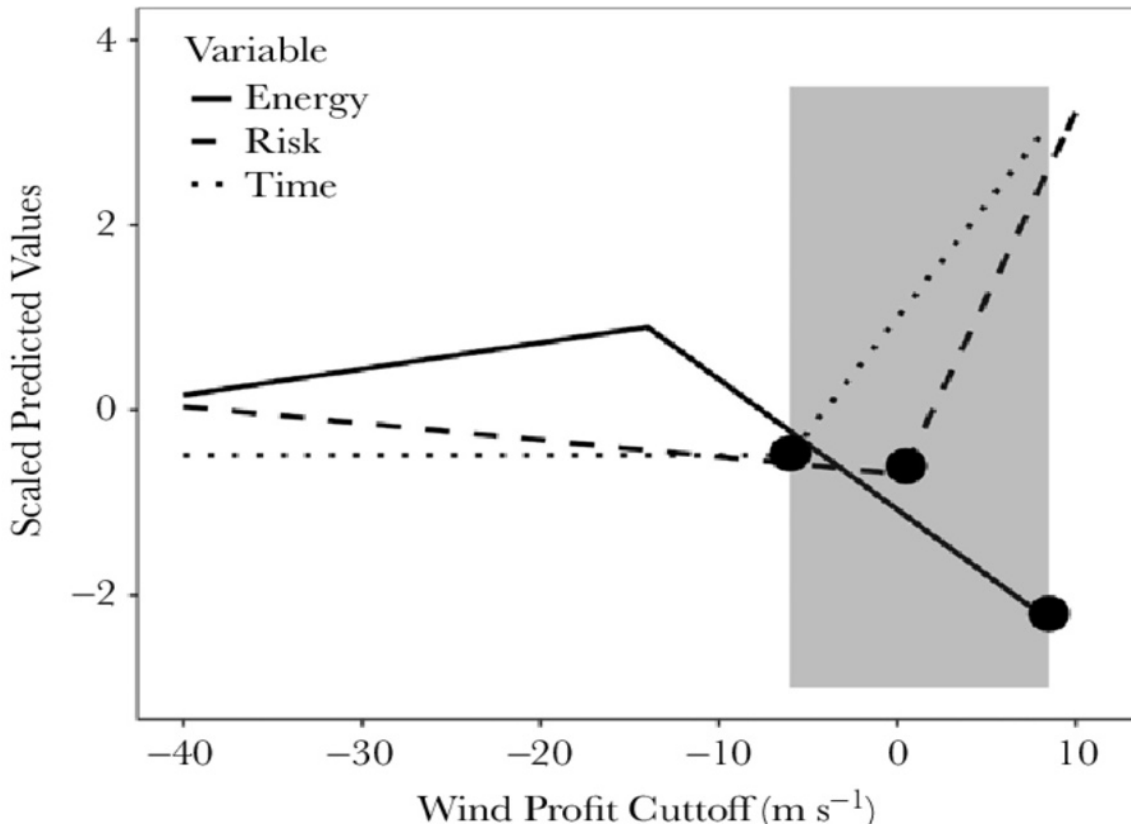


Figure *2.3:* Scaled predicted values for models explaining the effect of wind profit thresholds (m s^{-1}) on three migratory variables. Black dots represent the optimal wind profit threshold for each migratory currency. If optimizing only energy, birds should only fly when wind profits are above 8 m s^{-1}. Time optimizers should fly when wind profits are above −6 m s^{-1}, and to optimize solely for risk, birds should migrate with a wind profit threshold of 0.5 m s^{-1}. The *Gray* rectangle indicates the range of optimum migratory selectivity if multiple strategies are being used. (Source: McCabe,2015).

2.5 Challenges Facing Research and Gaps

Notwithstanding the progress made, there are still challenges in machine learning for predicting bird migration:

2.5.1 Sensor data and Integration: High-resolution GPS device data and radar data tend to be incomplete or sparse. Such data variability in terms of battery life, device weight, or environmental disturbance is a major obstacle for *modelling* as well as validation. Heterogeneous data sources (e.g., telemetry, radar, and *citizen science*) need strong preprocessing as well as normalization methods for their integration.

2.5.2 Spatial-Temporal Resolution: Migratory patterns are dynamic in nature, showing variability in space as well as through time. Models are required to recognize fine-scale temporal variations as well as broad-scale spatial patterns. It is tough to accomplish this balance, and prevailing models often fall short in their ability to be generalizable between different migratory seasons or regions.

2.5.3 Environmental Heterogeneity: Weather is one of the primary drivers for migration but is often non-linear in its impact. Most research has concentrated on *key* meteorological variables, but interactions between multiple environmental variables are not as thoroughly researched. In addition, errors in real-time weather predictions can be transmitted through forecasting paradigms, making the overall predictions less reliable.

2.5.4 Species-Specific Factors: Most models have been conceived for a group of bird species without being specialized in response to *behavioural* or physiological traits. The Taiga Goose, for instance, has distinct migratory patterns that are perhaps not adequately reflected in models based on other waterfowl or passerines.

2.5.5 Model interpretability: While increasingly advanced models are made, their "black box" qualities may conceal the ecological reasoning driving predictions. There is increasing demand for interpretable models that can offer insight into the drivers of changes in movement, as opposed to presenting point predictions. Overcoming these challenges will demand an interdisciplinary solution that incorporates advances in sensor technology, data processing, and machine learning with deep insight into avian ecology.

2.5.6 Computational cost: Training models that are large especially if it’s with geospatial and temporal inputs can be resources intensive. thereby most generally data’s are optimized and sampled when using large ensembles.

3 METHODOLOGY

The strategy used in the provided code is a full-scale geospatial data analysis, data cleansing, unsupervised clustering, and predictive *modelling* with time-series considerations. The subsequent report documents the steps taken and the rationale for the methodological selections, detailing each step from data ingestion and cleansing, via visualization and exploratory data analysis, up to clustering and predictive *modelling*. The methodological choices are justified with citations of current literature and data science best practices.

3.1 Data Collection

Taiga Goose Tracking data, acquired through *Move bank*, were used in this research together with climate data derived from the ERA5 dataset of ECMWF. In this project, the objective revolved around identifying patterns of migration for the Taiga Geese based on GPS coordinates as well as environmental features. Several phases of cleaning and preprocessing were carried out on the data *for* it to be prepared for *modelling* and analysis.

3.2 Importing, Loading and Merging the Datasets

In the research, several Python libraries were imported to facilitate data analysis and visualization. The *panda's* library was used to handle and manipulate the datasets, allowing for the cleaning and preprocessing of both the Taiga Goose tracking data and the climate data from the ERA5 dataset. The *NumPy* library was employed to handle numerical operations and for tasks that involved large arrays. The CSV data is read into a pandas Data Frame with the pd.read_csv() function, and the initial data inspection is performed with the info() function for data types and non-null values. In compliance with the best practices advised by Gorard (2020) and Karrar (2022), the first step is critical for the identification of possible issues like missing values or data types, which would adversely affect the subsequent analysis. To train machine learning models, the sklearn library was imported, specifically for the RandomForestRegressor model and other preprocessing techniques such as scaling. Additionally, seaborn and matplotlib.pyplot were imported to create various visualizations, including scatter plots and line graphs, to display the migration patterns of the geese over time. The folium library was also used for interactive mapping, providing geographical visualizations of migration paths and heatmaps. The time library was imported to simulate processing delays in Streamlit to enhance

the user experience during data visualization. Lastly, xarray was used to work with climate data from NetCDF files, converting them into a *Data Frame* for easier manipulation.

```
import streamlit as st
import pandas as pd
import folium
from folium.plugins import HeatMap, MarkerCluster, TimestampedGeoJson, MiniMap,
from streamlit_folium import st_folium
import joblib
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import streamlit.components.v1 as components
from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np
import pydeck as pdk
import gdown
import os
```

Fig 3.1: python script for importing, loading and preparing cleaned dataset

The main dataset, composed of bird-tracking data, was imported via CSV data. The dataset is composed of migration information for 44 specific Taiga Geese with corresponding columns for longitude, latitude, timestamp, and other pertinent metadata. Climate data derived from ERA5, such as wind speed and temperature, were processed and combined within the dataset as well.

- Bird Tracking Data: The data in this dataset comprised the GPS coordinates (longitude, latitude) of the birds being tracked over time, as well as sensor measurements like external temperature, ground speed, and heading.

- Climate Dataset: Historical climate variables with high spatial-temporal resolution were loaded using the ERA5 dataset in the form of NetCDF files. Those climate variables pertinent for analysis were extracted, such as wind speed (u10 and v10), temperature (t2m), and surface pressure (sp).

Both datasets were combined based on corresponding timestamps as well as geospatial coordinates (latitude and longitude) through an asof merge *to* synchronize bird-tracking data with the closest available climate data points.

```python
# Sort both datasets by timestamp
goose_df = goose_df.sort_values("timestamp")
era5_df = era5_df.sort_values("valid_time")

# Merge using nearest valid_time (ERA5) to each bird timestamp
merged_df = pd.merge_asof(goose_df, era5_df, left_on="timestamp", right_on="valid_time", direction="nearest")

# Drop the duplicate valid_time column
merged_df = merged_df.drop(columns=["valid_time"])

print("\n✅ Merged dataset structure:")
print(merged_df.head())

# Save the merged dataset
merged_file_path = "/Users/admin/Downloads/Merged_Taiga_Goose_ERA5.csv"
merged_df.to_csv(merged_file_path, index=False)

print(f"\n✅ Merged Dataset Saved: {merged_file_path}")
```

Fig 3.2: Python script for merging both dataset

The analysis begins with a large dataset of over 840,000 entries and 28 columns. The dataset contains variables like geospatial coordinates (i.e., location-long, location-lat), meteorological variables (i.e., external-temperature, t2m), GPS-specific meta information (i.e., gps:hdop, gps:satellite-count), and other variables for recording movement and sensor measurements of Taiga geese. Large GPS-telemetry-based datasets like these are now common in migration studies and provide the level of scale needed to capture fine-scale and long-distance patterns of movement (Chandler et al., 2021).

3.3 Data Preprocessing

One of the most critical parts of the strategy is the rigorous cleansing of the dataset, with special attention being paid for outliers. A function, replace outliers (), is specifically designed for the treatment of anomalies in numerical columns. The function employs the Interquartile Range (IQR) method, a common method of exploratory data analysis (Barnett & Lewis, 1994). The procedure involves the computation of the first (Q1) and the third (Q3) quartiles, the computation of the IQR as Q3 – Q1, and the subsequent definition of lower and upper bounds as Q1 – 1.5 × IQR and Q3 + 1.5 × IQR, respectively. Those values beyond these bounds are replaced with the median of the column—a robust measure of centrality least susceptible to outliers (Singh and Kundu, 2022; Hartwig et al., 2020).

The IQR-based method is particularly appropriate for non-normal distributions, e.g., sensor and geospatial data distributions. In contrast with the removal of outliers, which can discard valuable information, the replacement approach preserves the overall structure and statistical properties of the data. The approach follows a previous recommendation in the literature and avoids subsequent analysis, like predictive modelling, from being unduly impacted by outlier observations (Marques et al., 2023).

Also, the approach has a mechanism for the detection of outliers using a threshold. By measuring the distance covered by two consecutive data points and flagging those that exceed a specified threshold (for example, 50 km), the approach detects unrealistic motion that most likely is the result of GPS malfunctioning or data logging error. The outliers are removed from the data, and analysis is conducted on realistic patterns of motion (Karrar, 2022). After the application of the outlier replacement on all the numerical columns (selected with the assistance of select_dtypes(include=['number']), the data is saved for analysis. Saving the data is imperative for the sake of data integrity for exploratory visualization and modelling.

```python
# List of numerical columns to clean
numerical_cols = dff.select_dtypes(include=['number']).columns

def replace_outliers(dff, cols, method="median"):
    cleaned_df = dff.copy()
    for col in cols:
        Q1 = cleaned_df[col].quantile(0.25)
        Q3 = cleaned_df[col].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        if method == "median":
            replacement = cleaned_df[col].median()
        elif method == "mean":
            replacement = cleaned_df[col].mean()
        else:
            continue

        cleaned_df[col] = np.where((cleaned_df[col] < lower_bound) | (cleaned_df[col] > upper_bound), replacement, cleaned_df[col])
    return cleaned_df

df_cleaned = replace_outliers(dff, numerical_cols, method="median")  # Replace outliers with median

# Save the cleaned dataset
df_cleaned.to_csv("new_data.csv", index=False)

print(f"Original dataset size: {dff.shape[0]} rows")
print(f"Cleaned dataset size: {df_cleaned.shape[0]} rows")
```

Fig 3.3 python script for data cleaning

3.4 Exploratory Data Analysis

With the clean data, the next methodological step is geospatial visualization and exploratory analysis. Seaborn's scatterplot function is utilized for the visualization of the migration paths of the Taiga geese, with location-long plotted against location-lat and coloured by a bird identifier. The scatter plots facilitate easier identification of spatial clusters, trends, and potential anomalies of migration behaviour (Li et al., 2025). The visualization indicates obvious migratory corridors and potential stopover sites that play a vital part in the understanding of movement ecology.

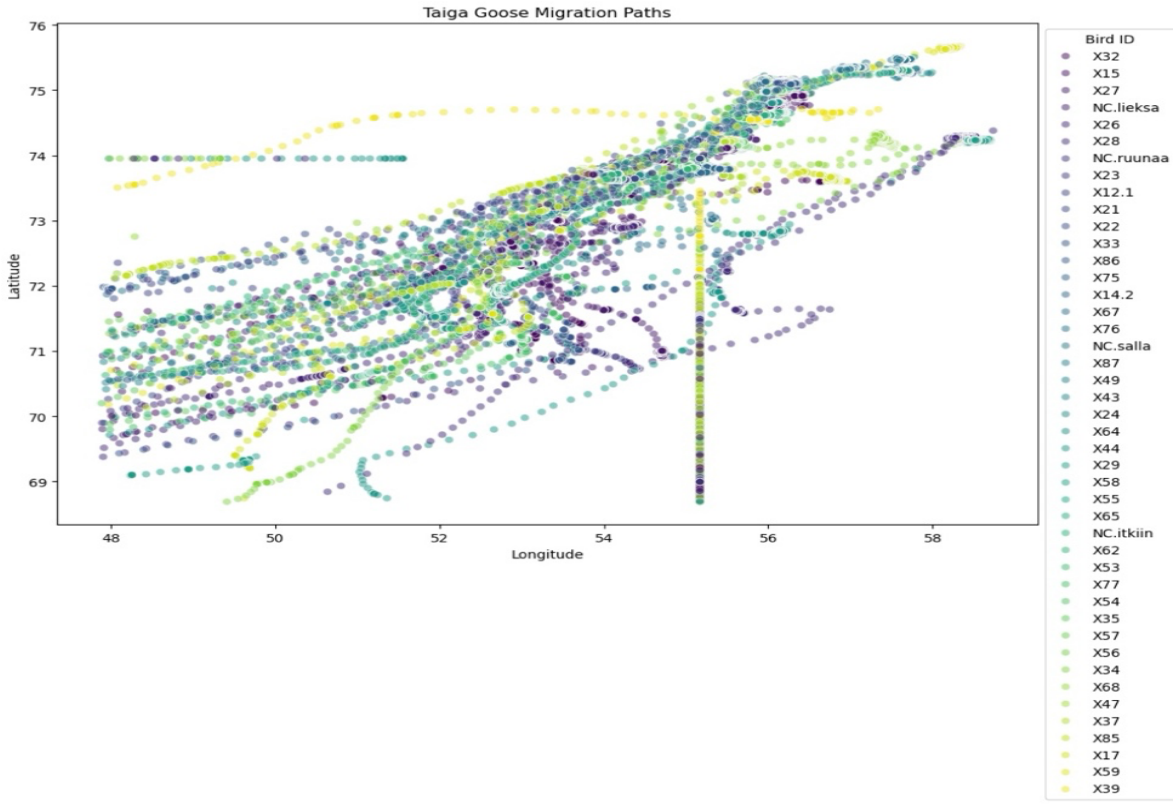


Fig 3.4: Scatterplot of Taiga goose migration paths

Fig 3.5: Scatterplot of Taiga Goose Migration paths with clustering after outliers was removed.

Fig 3.6 was integrated to complement the static visualizations; the method brings together interactive map-based visualizations with Folium. Folium is used for generating maps with marker clustering and heatmaps, allowing the researchers to interact with the geospatial data interactively. Application of marker clusters avoids cluttering the view by clustering neighbouring geospatial points, and heatmaps present a natural visualization of density patterns along migration corridors. In keeping with Walker et al. (2020) and Schöttler et al. (2021), these interactive visualizations not only allow for the interpretability of complex geospatial data but also make it easier to identify high-ecological-significance zones.

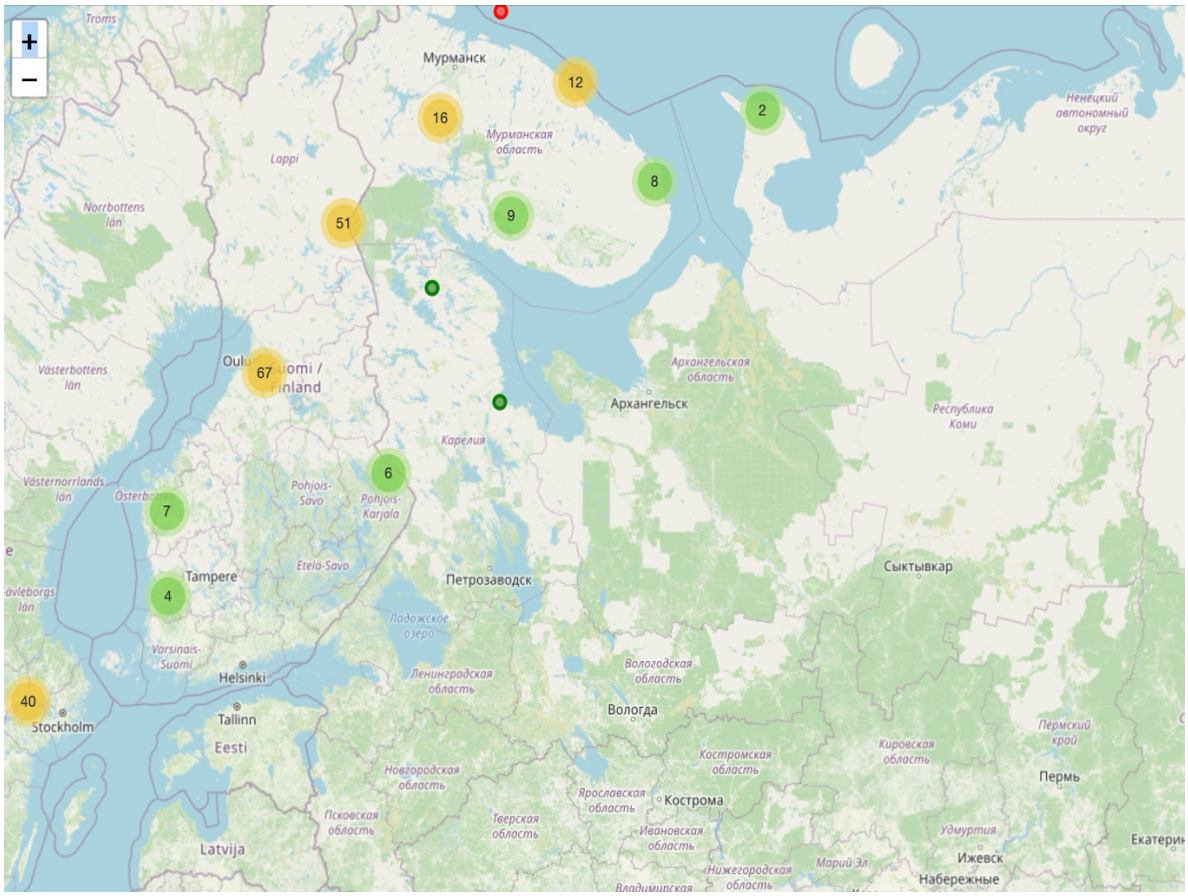


Fig 3.6: Snapshot of Taiga goose Migration marker clustering on map generated through folium

Also, the method employs heat maps for density estimation. Seaborn's kernel density estimation plots and Folium's interactive heatmaps are employed for the visualization of the intensity of the migration occurrences at the spatial scale. These facilitate the identification of the high-density locations most likely representative of critical stopover sites or preferred migration corridors (Huang et al., 2025). Concurrently, the dataset is converted into numeric formats, and the correlation matrix is computed. A corresponding heat map is then created for the assessment of the pair-wise associations of the variables, whose insights inform the ensuing predictive modelling and feature selection.

Also, the use of Seaborn's KDE plots and Folium's heatmaps provides additional representations of spatial density. KDE plots can offer statistical estimates of density, but dynamic exploration of spatial intensity can be achieved with interactive heatmaps. These methods of visualization, as recommended by Wang et al. (2022) and Hu et al. (2024), are of vital importance for uncovering complex spatiotemporal patterns of ecological data.

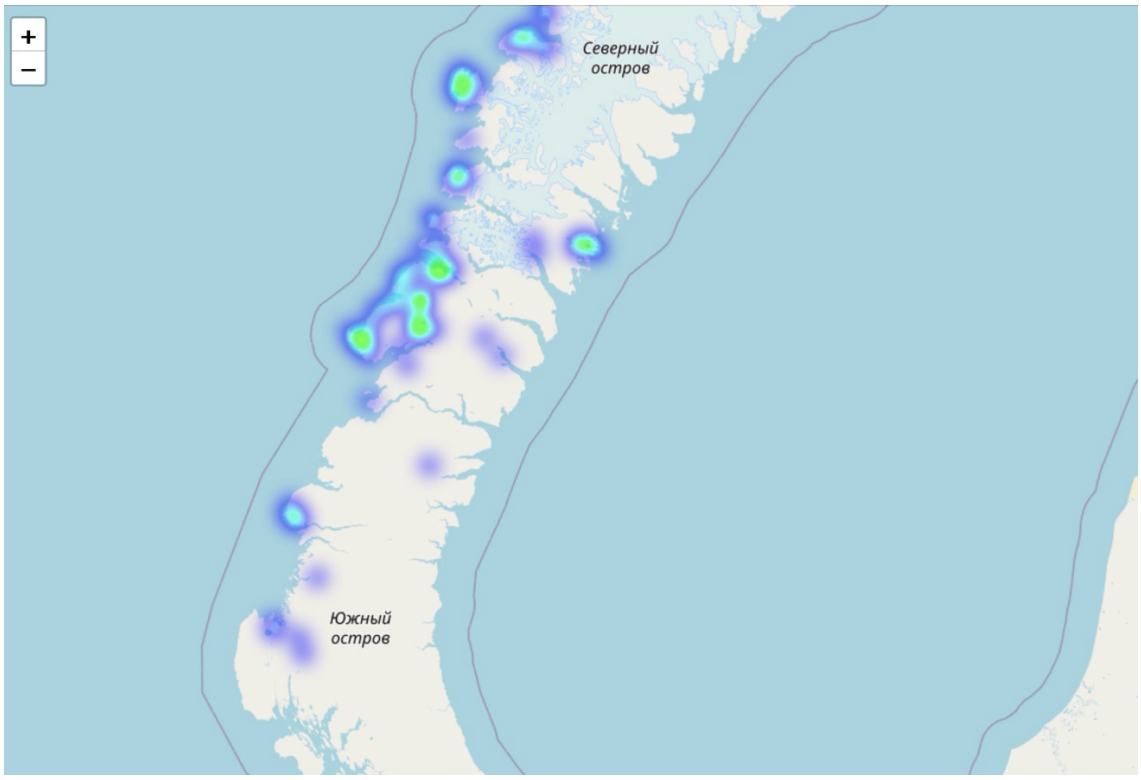


Fig 3.7: Snapshot of Taiga goose migration path heat map generated through folium.

3.5 Statistical Analysis and Visualization

The strategy also encompasses a series of statistical tests for the analysis of the distributional aspects of the data. With Seaborn, we generate several plots, histograms, boxplots, and line plots to analyse the centre tendency, spread, and time-based trends of the data. Histograms and boxplots provide information on distributions of variables such as flight altitude and direction, while line plots are used for observing the variation of speed and direction with time, typically by timestamp and cluster.

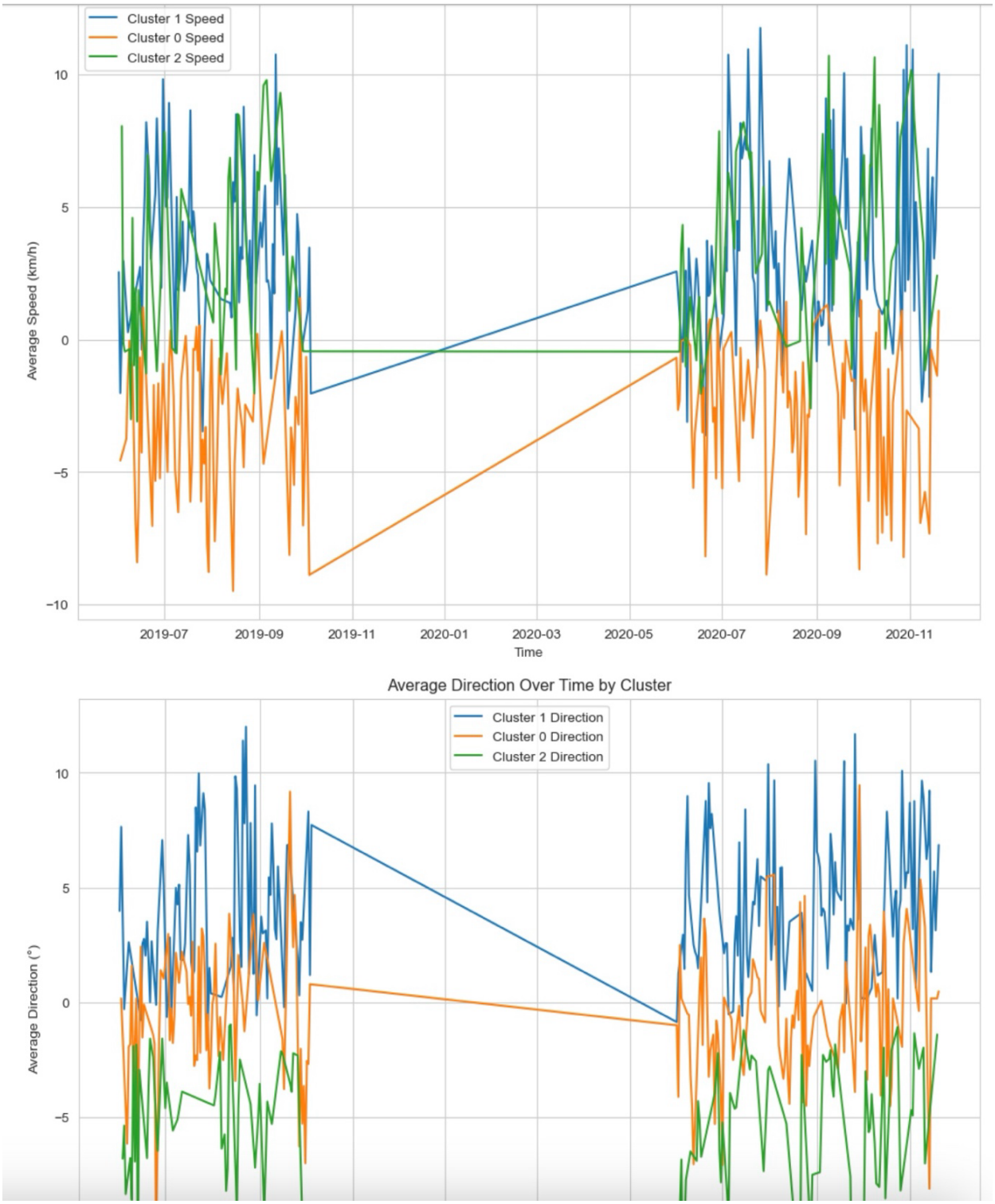


Fig 3.8: Time series line plot showing average speed over time by cluster (Top plot) and Average direction over time by cluster (bottom plot)

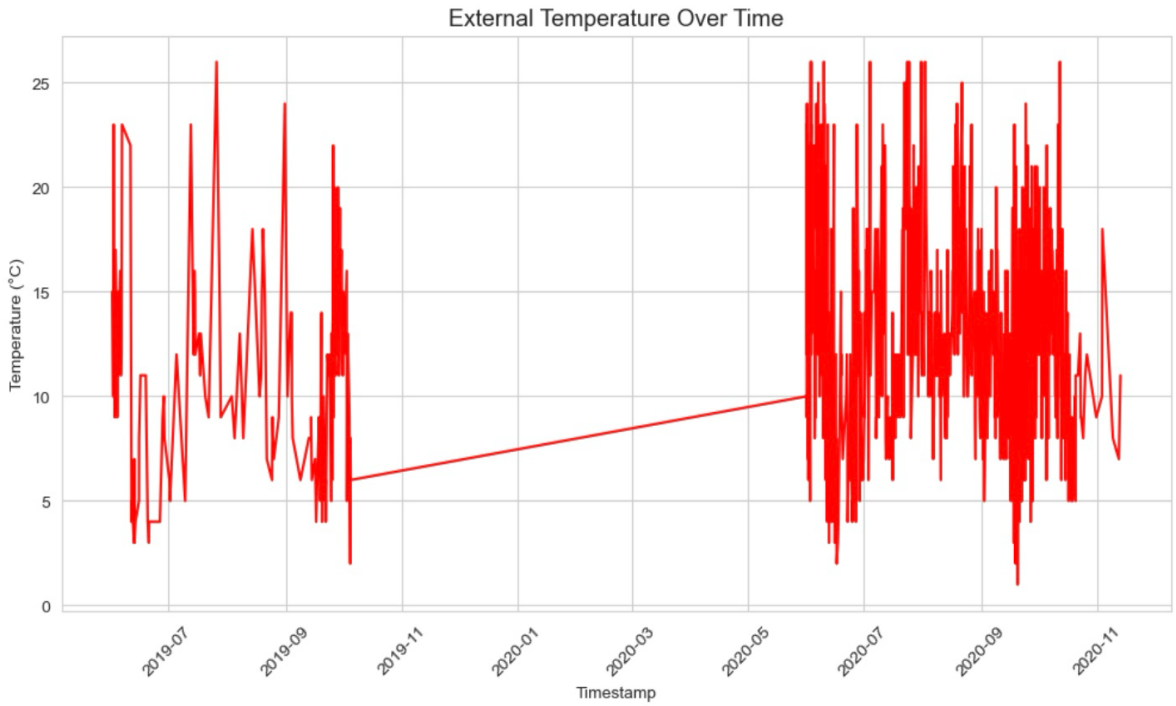


Fig 3.9. Time series line plot showing external temperature over time

3.6 Unsupervised Clustering Analysis

To better understand the migration pattern of the Taiga Goose, unsupervised clustering is employed. In this context, KMeans clustering is specifically employed for data partitioning based on the u10 and v10 components of the velocities, which represent horizontal and vertical velocities of the wind. KMeans clustering is a strong data clustering algorithm for clustering data points by minimizing the sum of squares of the data points for the clusters (Huang et al., 2021). In this work, the algorithm is initialized with three clusters and a fixed state for reproducibility purposes. The geospatial points resulting from the clustering process are then coloured based on their corresponding cluster label.

Aside from the standard KMeans, the computational needs of this large dataset are addressed with the application of MiniBatchKMeans. Unlike the standard KMeans, this variation processes data in batches, thus reducing the computational load but still approximating the results of the standard KMeans. In keeping with the work of Pei and Ye (2022), MiniBatchKMeans is particularly ideal for large data, making the clustering analysis scalable and effective.

The clustering analysis is employed for the purpose of identifying singular patterns of migration or behaviour among the geese. By clustering data based on the conditions of the wind, the method uncovers underlying environmental factors influencing migration, hence proving theoretical frameworks such as the Optimal Migration Theory (Hunter and Simon, 2022).

```
from sklearn.cluster import MiniBatchKMeans
import folium
from folium.plugins import MarkerCluster

clustering = df_clean[['u10', 'v10']]
kmeans = MiniBatchKMeans(n_clusters=3, random_state=42, batch_size=100
df_clean['cluster'] = kmeans.fit_predict(clustering_data)

df_sampled = df_clean.sample(n=1000, random_state=42)
```

Fig 3.11: Python code for implementation of Kmeans clustering and use of MiniBatchKMeans in the Taiga Goose dataset.

3.7 Predictive Model Development

One of the most important components of the strategy is the building of predictive models that project the geospatial position (latitude and longitude) of the Taiga Goose. Two of the primary regression models used are Random Forest and Gradient Boosting, with a secondary time-series predictive model based on the use of Long Short-Term Memory (LSTM) networks.

The Random Forest model is implemented using sci-kit-learn's Random Forest Regressor. The dataset is split into training and test sets, and the model is trained on a subset of features chosen using extensive feature engineering. Random Forest is an ensemble learning technique that employs multiple decision trees grouped together to avoid overfitting and identify complicated, nonlinear relationships between features (Ganaie et al., 2022). The model's performance is evaluated using Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), two measures commonly used in predictive modeling literature.

On the other hand, the Gradient Boosting model is used with the MultiOutputRegressor wrapper with the base estimator being GradientBoostingRegressor since the target variable is multidimensional. Gradient Boosting continuously reduces the error of predictions by adding

weak learners, a method of immense utility for finding subtle nonlinear relationships (Ilhan et al., 2023). In spite of the robust theoretical foundations of ensemble methods such as Random Forest and Gradient Boosting, their performance is highly dependent on the appropriate tuning of the hyperparameters and feature selection.

Aside from ensemble methods, the strategy also entails a time-series forecasting model with the usage of LSTM neural networks. The LSTMs, built with the Keras API of TensorFlow, can learn long-term dependencies of time with the assistance of specialized gate controlling units for the vanishing gradient problem (Kashif, 2023). The model is structured with the first LSTM layer of 64 units, then the dropout layer for avoiding the problem of overfitting, the subsequent LSTM layer of 32 units, the subsequent dropout layer, and the final dense layer with two neurons for the latitude and longitude predictions. A helper function creates sequential samples from the data with the usage of the sliding window technique with a fixed sequence length (i.e., seven-time steps). Training of the LSTM model is conducted with the Adam optimizer and the usage of the mean squared error as the loss function.

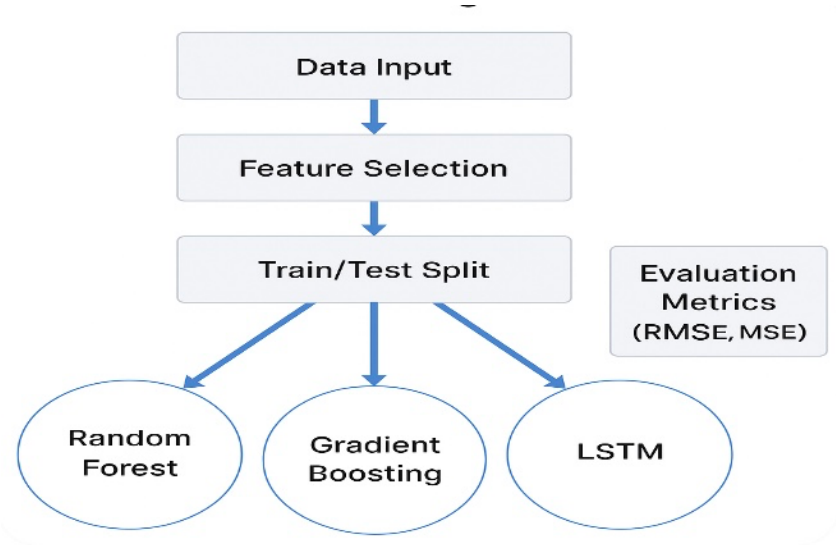


Fig 3.12: Predictive Modelling workflow

3.8.1 Streamlit Application for Model Deployment and Visualization

To improve the accessibility and interactivity of the migration analysis, a user-friendly web application was created using the Streamlit which is a python framework. This App served as a practical interface for displaying predictions, geospatial data, and clustering insights derived from the study. The application allowed users to interact with the model and explore the distribution of Taiga goose across the globe.

The dashboard is divided into five Main sections:

- EDA Snapshots: This part presented pre-generated visual summaries of exploratory data analysis, including movement patterns, altitude trends, correlation heatmaps, and behaviour-based clustering results.

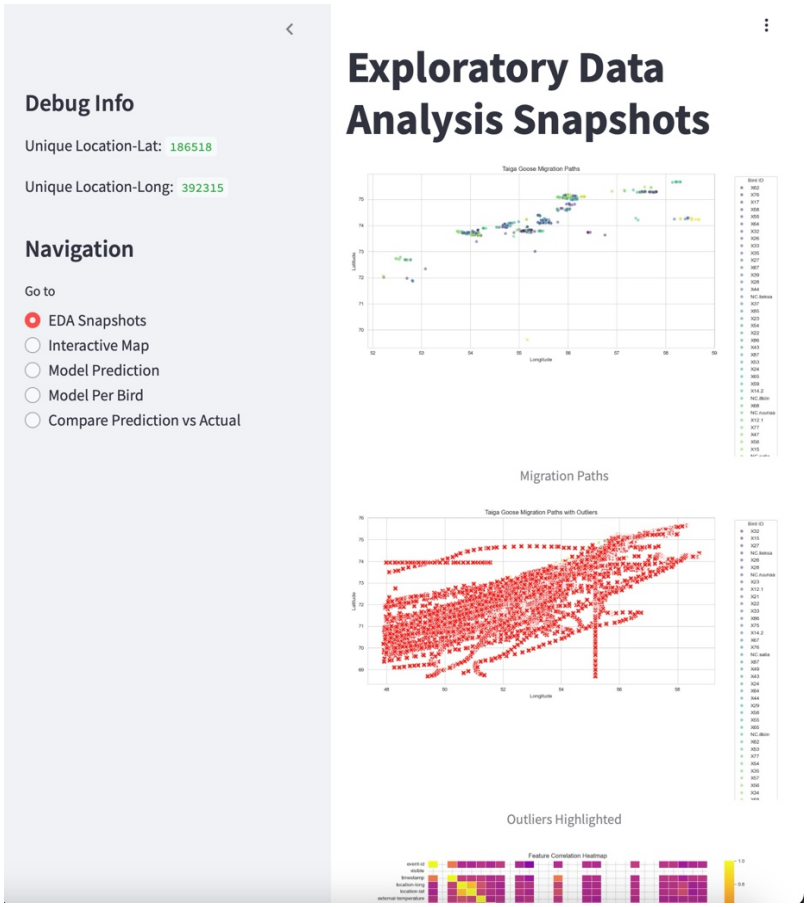


Figure 3.13: Snapshot of exploratory data analysis on stream lit dashboard

- Interactive Map: Leveraging Folium and integrated into Streamlit via streamlit_folium, this section featured a dynamic map with clustered data points, heatmaps, and time-based animations of migration. Users could view bird locations over time and explore spatial movement intuitively. The map incorporated several dynamic layers to improve user engagement and ecological interpretation. The base map was rendered using OpenStreetMap tiles, and migration data points were displayed using a Marker Cluster, allowing users to inspect individual bird details such as speed, direction, and timestamp. A heatmap overlay was included to highlight areas with high migratory density, offering insights into stopover hotspots. The TimestampedGeoJson layer animated movement over time, enabling users to explore temporal trends in migration. A Mini Map was embedded for navigational reference, while a Mouse Position tracker displayed precise coordinates as the user hovered across the map. A layer control panel was also provided, giving users the ability to toggle map features on and off to suit their analytical needs.

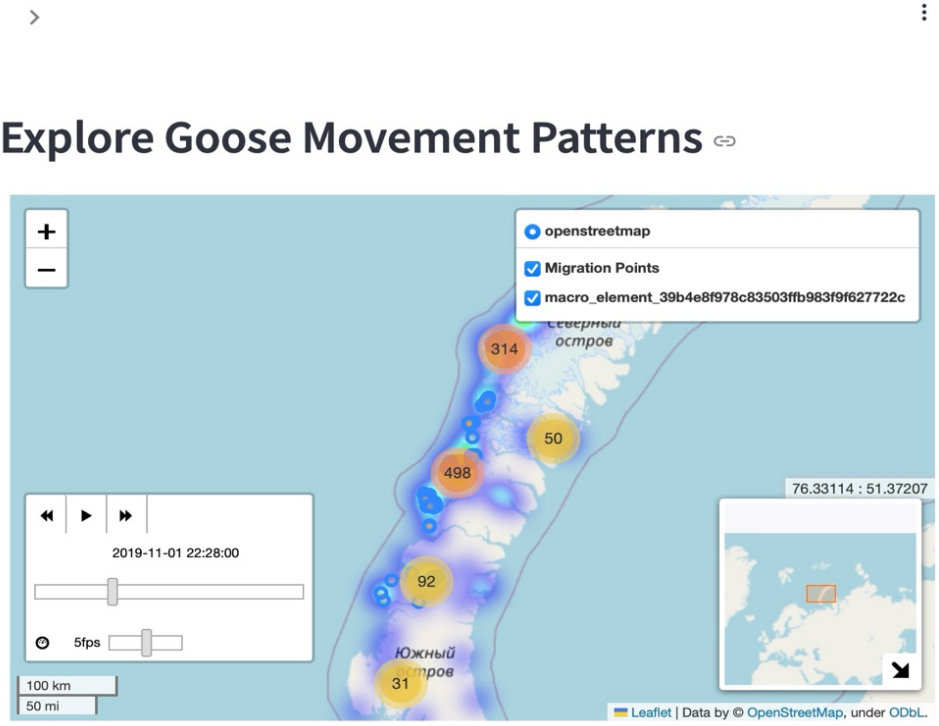


Figure 3.14: Interactive Heatmap Display of Taiga Goose Migration Patterns on stream lit dashboard.

- Model Prediction: Users could manually input values for selected features used during model training. The application returned latitude and longitude predictions from a pre-trained Random Forest model, making real-time forecasting possible without retraining.

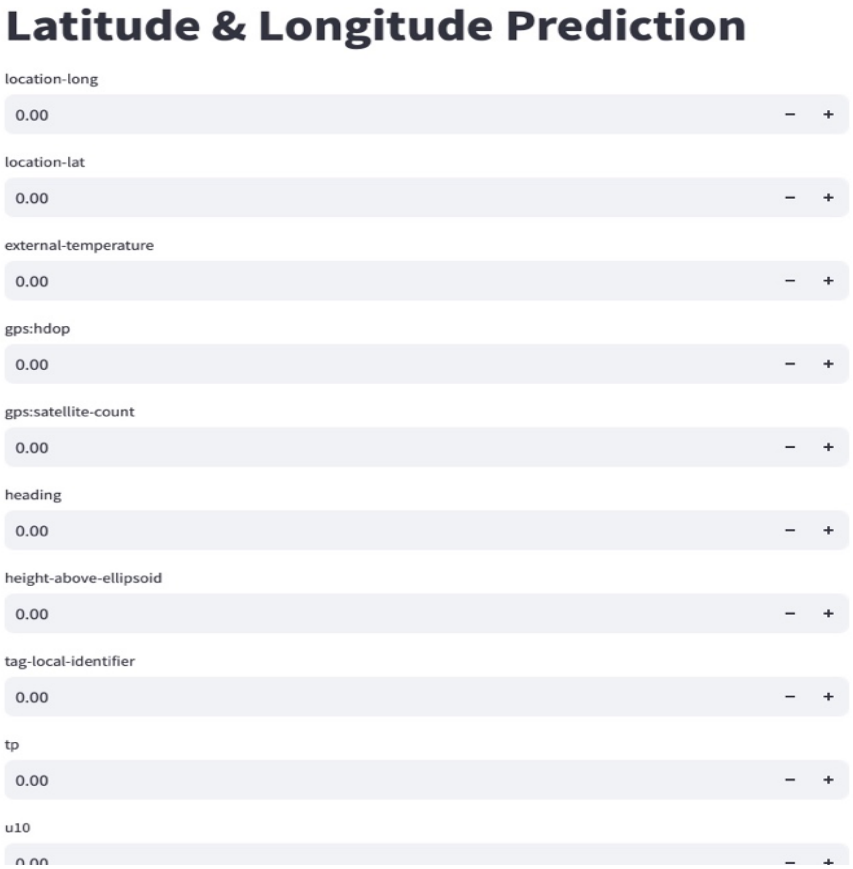


Fig 3.15: Model prediction using longitude and latitude with other features on stream lit Dashboard

- Model Per Bird: This section enabled users to select an individual bird and visualize all its predicted locations on a map. It also offered the option to download the full prediction dataset for the selected bird.

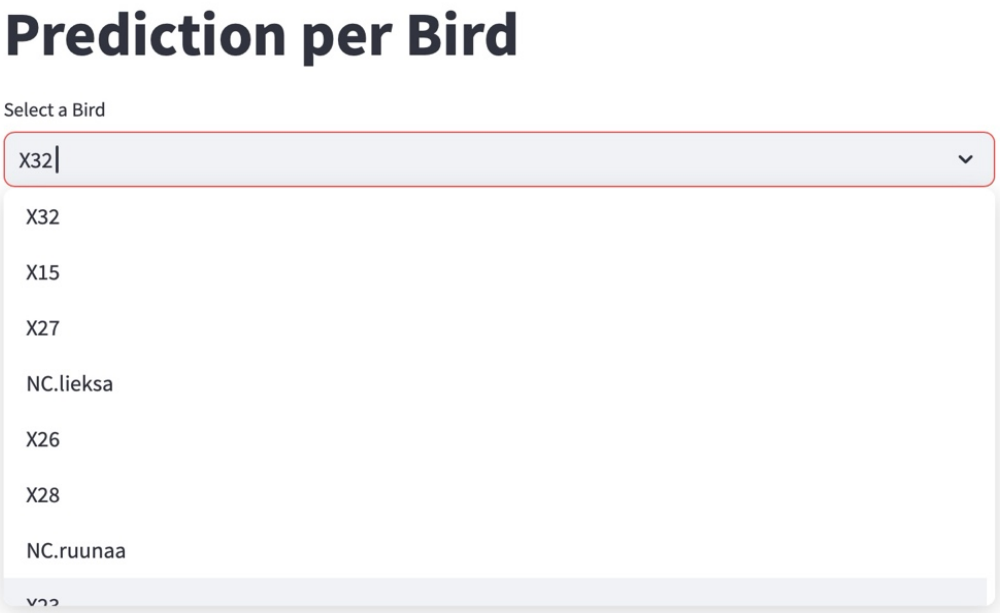


Figure 3.16: Model Per bird prediction section on stream lit application

- Prediction vs Actual Comparison: This part provided a performance assessment of the predictive model. Users could filter predictions by date range, compare predicted coordinates with true values, and view error metrics such as RMSE and MAE. Histograms and scatter plots illustrated the accuracy and distribution of the model's errors.

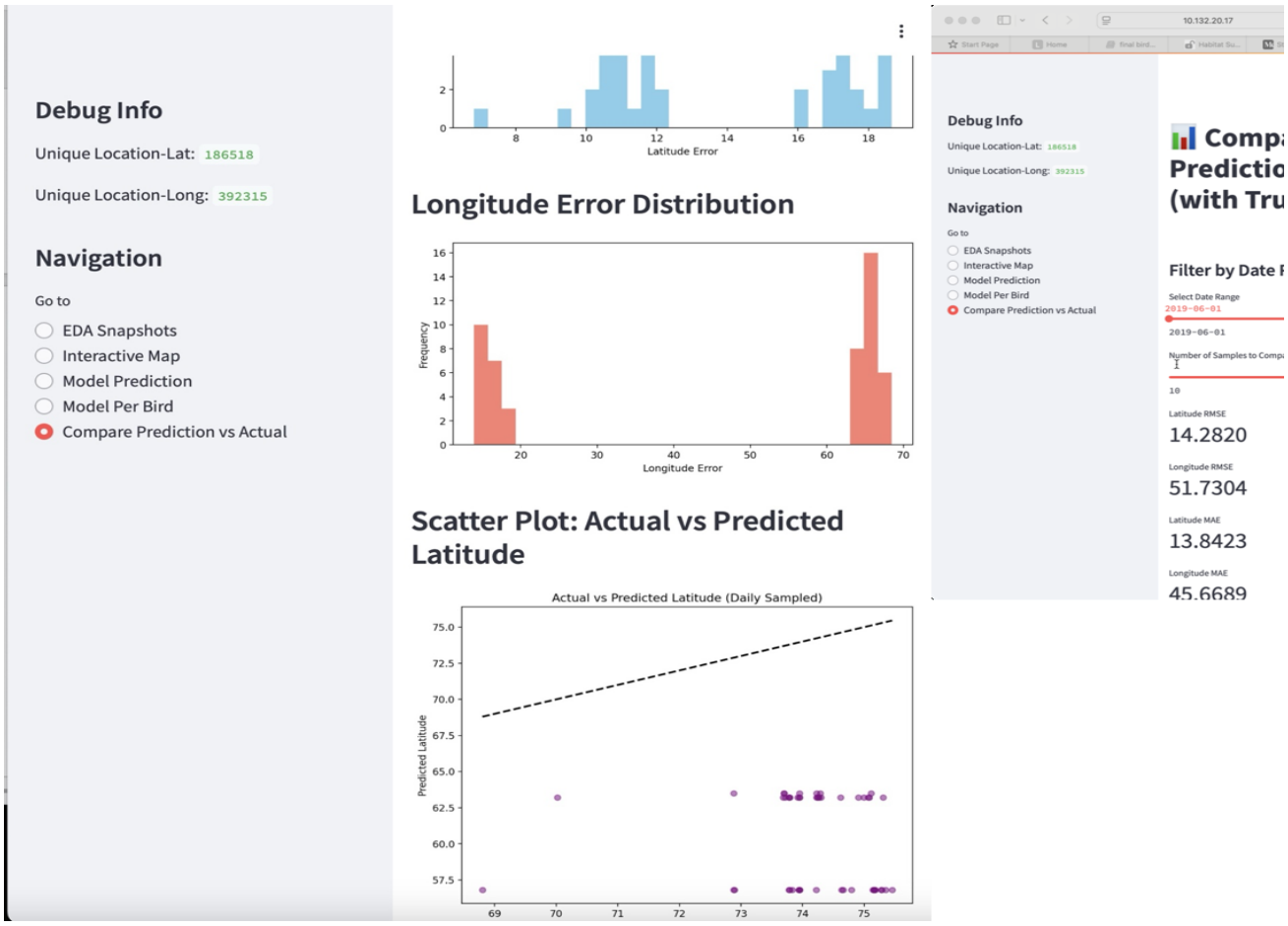


Figure 3.17: Snapshot of Streamlit Application – Comparison of Predicted vs Actual Migration Coordinates

This figure shows two views of the Streamlit app interface used to evaluate prediction accuracy. The left panel displays error distributions and scatter plots comparing predicted and actual latitudes. The right panel shows the user interface for selecting date ranges and sample size, along with RMSE and MAE metrics for model performance evaluation.

Overall, the Streamlit application enhanced the transparency and usability of the project's findings. It provided a practical platform for exploring migration behaviour and model accuracy in a visual and interactive format. Similar tools have proven effective in communicating data science findings in ecological research. The streamlit application can be explored further with this link. **http://10.246.17.82:8501/**

3.8.2 Chapter Summary

The method proposed here is a strong and integrated approach for the analysis of the migration of Taiga geese. The integration of robust data cleansing, exploratory data analysis, unsupervised clustering, and advanced predictive modelling is a cohesive framework that is not only

appropriate for the purposes of the research but also useful for the scientific community of movement ecology. By connecting the most up-to-date data science methodologies with well-established theoretical frameworks, this approach offers a solid foundation for future studies on the understanding and prediction of bird migration in the context of rapid environmental change.

4 FINDINGS

This chapter offers an extensive analysis of a dataset of geospatial and meteorological measurements across more than 840,000 events. It opens with a descriptive analysis of salient statistics and possible geolocation anomalies, as well as external temperature, GPS quality metrics, flight movement, altitude, and meteorological variables. This chapter presents descriptions of the use of K-means clustering analyses for the determination of discrete wind regimes (see Appendix 1), of cross-correlation analyses for the determination of the relationships between variables, and of evaluation of predictive models Random Forest, Gradient Boosting, and Long Short-Term Memory (LSTM) networks for geospatial coordinates forecasting. It closes with a determination of feature importance, in which atmospheric variables, especially surface pressure and wind components, have a substantial impact on performance in the models.

4.1 Descriptive Analysis

- Geolocation: The average longitude is approximately 49.21, with a standard deviation of 12.40. The average latitude is approximately 72.10, with a standard deviation of 4.28. These figures, however, which under ordinary circumstances would be expected to be within a smaller range (i.e., latitude between -90 and 90, and longitude between -180 and 180), show that either the data represent a transformed coordinate system, or the values could be misinterpreted unless the study context is well understood. This requires careful domain-specific interpretation.
- External Temperature: The ambient temperature varies moderately with the outside temperature being 13.12 °C with a standard deviation of 5.99 °C. The minimum and maximum values of the outside temperature range from -5.0 °C to 53.0 °C. Any such fluctuation is typical of environmental monitoring and could be due to seasonal or regional climatic variation.
- GPS Quality Metrics: The “gps:hdop” (horizontal dilution of precision) has a mean value of about 1.08 with a relatively low standard deviation (0.35), suggesting that the GPS data are of consistently good quality. While the “gps:satellite-count” is

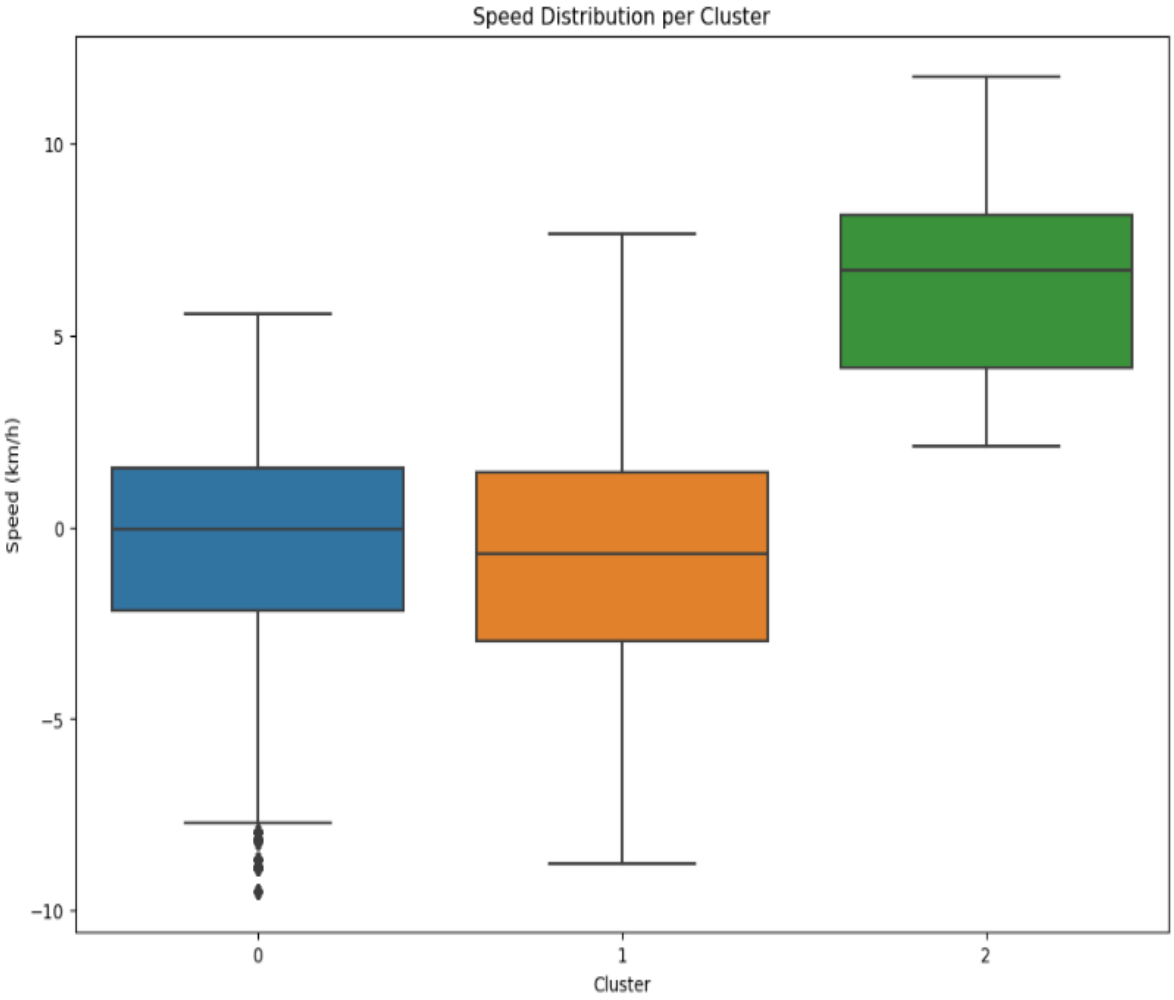
approximately 7.53, typical of consumer GPS receivers, indicating that a medium number of satellites were used for positioning, increasing the robustness of the position estimates.

- Flight and Movement: The “groundspeed” variable has a considerable standard deviation (20.02) compared with its mean (0.49). In this case, the majority of the records will be zero (as the 25th, 50th, and 75th percentiles all being zero suggest), with occasional extremely high values (as much as 18,133.35), suggesting that the majority of the events will be stationary or low speed, but with the occasional extremely high-speed motion. Additionally, the "heading" variable, which denotes the direction of orientation, exhibits a large spread, ranging from 0 to over 500 degrees. The spread is beyond the usual 0–360 degree spread, suggesting some of the values may be the result of cumulative rotations or contain sensor noise/outliers.
- Altitude: “Height-above-ellipsoid” has a mean of approximately 61 m, but a standard deviation of 156.70 m. Of particular significance is the minimum of -2000 m, most likely a mistake or placeholder for bad data, and the maximum of approximately 10,000 m. These values further reinforce the importance of handling outliers robustly for further analysis.

- Meteorological Variables: The "u10" and "v10" variables represent horizontal speeds of the wind. Their means for "u10" is 1.28, and for "v10" is 0.22, with their respective standard deviations of approximately 4.50 and 4.96. Their distributions suggest most of the values will be middling, although with potential for strong winds, or maybe even error of measurements. In addition, the reading of "t2m", the 2-meter surface temperature, is averaged at 285.08 (presumed to be in the usual meteorological unit, Kelvin), with a negligible standard deviation of 2.91. The conversion would be around 12 °C on the mean, which is also in agreement with the outside temperature measurements. "sp", most likely surface pressure, has a mean of approximately 100,735 with a standard deviation of 1,100, with little variation from the mean a characteristic common for pressure data measured on units of Pascals or some other scale.

The dataset analysis, after intensive outlier correction, indicates a vast range of geospatial and meteorological measurements obtained from over 840,000 events. In treating potential outliers, the method employed robust statistical procedures, namely, imputing values beyond the 1.5

IQR range with the median. As a case, outlier values of the ground speed with the 25th, 50th, and 75th percentiles being zero against a maximum of over 18,133 m/s were corrected efficiently such that further analysis was not adversely affected by such extreme values. Additionally, implausible GPS points—identified using the Haversine distance with a 50-km cutoff (with outlier indices such as 4118, 4119, 4123, etc.) were excluded for data integrity improvement.



In Figures 4.1and 4. 2, the cluster analysis of the u10 (horizontal wind speed) and v10 (vertical wind component) further split the data into three groups. Cluster 0, with a mean u10 of around -0.52 (std ≈ 2.85) and a mean v10 of 3.45 (std ≈ 3.10), is characteristic of the regime with a weakly negative horizontal wind component and a moderately positive vertical component. Cluster 1, with a mean u10 of -0.82 (std ≈ 3.26) and a mean v10 of -4.92 (std ≈ 3.04), is characteristic of the reverse wind regime with a more negative horizontal component and a strongly negative vertical component.

Figure 4.1: Cluster Analysis on Speed

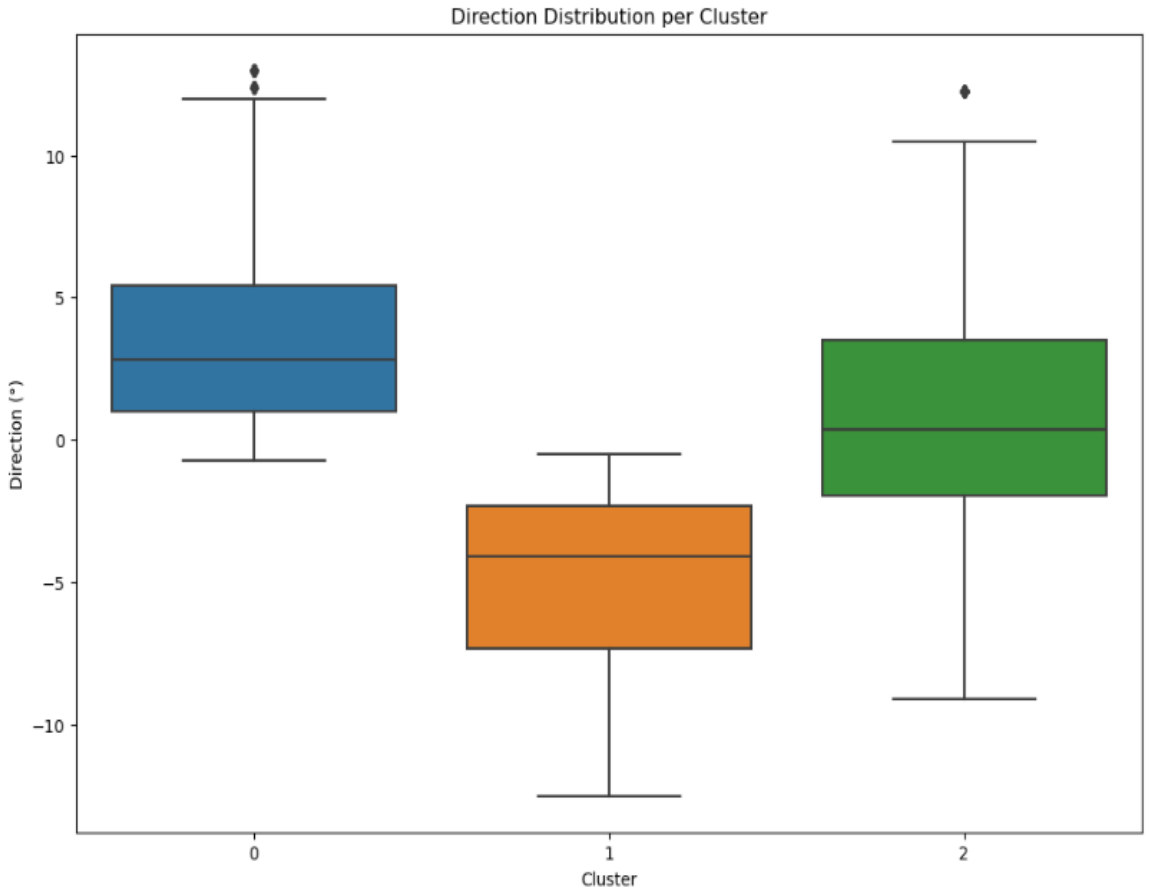


Figure 4.2: Cluster Analysis on direction

Cluster 2, however, is characterized by a strong horizontal component with a mean u10 of 6.36 (std ≈ 2.49) and a near-neutral vertical component (mean v10 ≈ 0.84, std ≈ 4.12). The relative homogeneity of each cluster suggests that the data reflect unique meteorological regimes that can potentially affect migration behaviour, for example, by allowing energy-saving travel during the tailwind regime of Cluster 2.

4.2 Correlation Result

Correlation Matrix

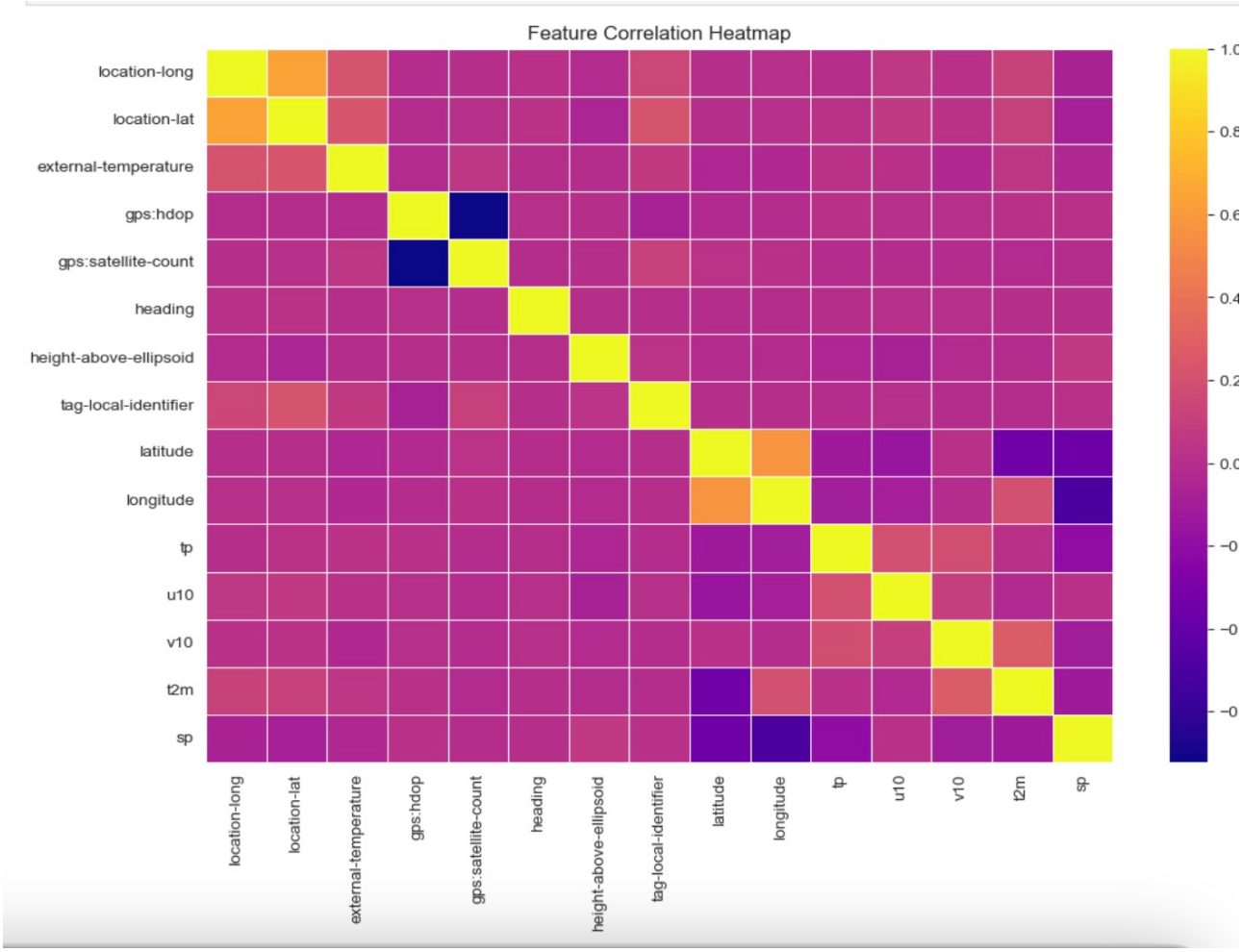


Fig 4.3: Correlation matrix

A correlation analysis of the cleaned numeric variables in Figure 4.3 offers further insights into the relationships of the dataset. A positive correlation of 0.63 for location-long and location-lat indicates a spatial gradient across the study site. External temperature is weakly positively correlated with longitude (≈ 0.215) and latitude (≈ 0.232), indicating geographical gradients that may reflect climatic gradients. The strong negative correlation of -0.722 for gps:hdop and gps:satellite-count supports the predicted inverse relationship whereby higher numbers of satellites improve the position. The meteorological variables, u10 and v10, show weak relationships with other variables; u10 is moderately positively correlated with the variable tp (≈ 0.193), but v10, weakly correlated with tp (≈ 0.182), shows only weak relationships with pressure and temperature. The height-above-ellipsoid variable, however, shows little

relationship with geospatial coordinates, indicating that the variation of altitudes is likely due to factors other than horizontal positioning, such as sensor error or atmospheric processes. Furthermore, the moderate relationships observed for tag-local-identifier with the two geospatial coordinates indicate possible spatial clumping of individual birds, possibly due to natural aggregations or the practical difficulties of sensor deployment.

4.3 Model Results

The predictive model results in Figure 4.5 indicates a remarkable difference in the performance of the methods used for predicting geospatial coordinates from the selected features. The Random Forest model, being an ensemble method with the capacity of handling nonlinear relationships, had a mean squared error (MSE) of 0.0075 and the respective root mean squared error (RMSE) of 0.0868. These values indicate the accuracy of the predictions of the target variables, with the error values being remarkably low. The result shows that the Random Forest model was able to capture the underlying patterns of the data, resulting in predictions that closely resemble the observed values.

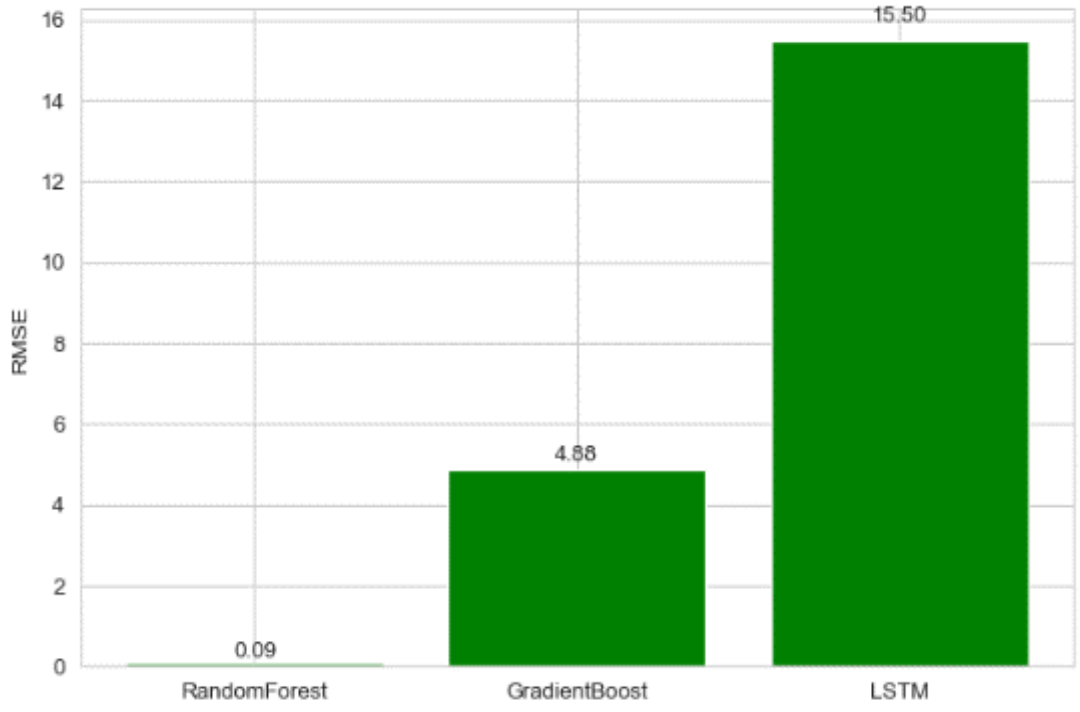


Fig 4.5: Model Comparison

On the other hand, the Gradient Boosting model, with a MultiOutputRegressor wrapper for handling the multi-dimensional nature of the target values, performed considerably worse. The Gradient Boosting model had an MSE of 23.8076 and an RMSE of 4.8793. The high error values indicate that the model did not generalize as well on the test data, perhaps due to the complexity of the relationships among the features and the geospatial targets, or because the boosting parameters were not properly optimized. The wide gap in the performance of the Random Forest and Gradient Boosting models suggests the sensitivity of gradient boosting procedures to the peculiarities of the dataset and the potential for further optimization, or a different methodological strategy, for this family of models.

The analysis was further extended into a time-series forecasting framework using a Long Short-Term Memory (LSTM) neural network. The sequence generation step transformed the input data into sequences of seven-time steps, resulting in input arrays of shapes (840205, 7, 13) for the features and (840205, 2) for the targets. The sliding window technique was important for the inclusion of the temporal dependencies inherent in the migration data. The LSTM model was constructed with two layers—a first LSTM with 64 units (configured with the capability of returning sequences), followed by a dropout for avoiding overfitting, and a second LSTM with 32 units, also with dropout. The final dense layer, with two neurons, produced the latitude and longitude predictions. The model was trained for 10 epochs with a batch size of 32 and a learning rate of 0.001, optimized with the Adam optimizer with the mean squared error as the loss function.

Even with the carefully designed network structure and rigorous training schedule, the performance of the LSTM model was worse compared with that of the Random Forest model. When tested on the test set, the LSTM model produced an MSE of 240.2746 and an RMSE of 15.5008. These values mirror the fact that the LSTM could not learn the complex temporal relationships and the spatial dependencies present in the dataset and, hence made much greater prediction mistakes. The fact that the training loss drops progressively with the number of epochs, as can be observed from the training logs, shows that the model was learning with increasing time; however, the uniformly high validation loss suggests potential model generalization issues or the need for further tuning of the hyperparameters.

4.4 Feature Importance Result

The feature importance analysis in Figure 5 shows that the most influential predictor is surface pressure (sp), with a contribution of 0.382594, indicating that it has the most effect on the predictions of the model. The wind variables come next, with the vertical component of the wind (v10), at 0.280054 and the horizontal component of the wind (u10), at 0.150717, indicating the significance of the dynamics of the wind on the predictive model.

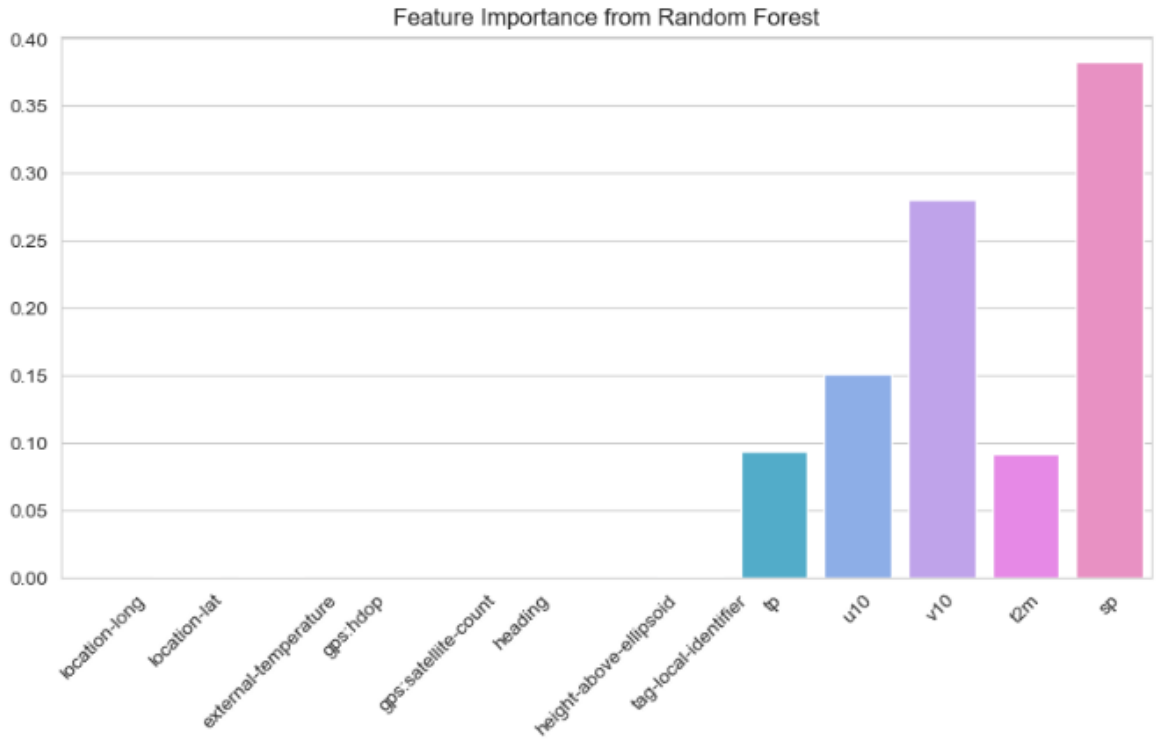


Fig 4.6 Feature importance from the random forest model.

4.5 Chapter summary

The variable tp (0.093422) and the temperature at 2 meters (t2m, 0.09147) also make a notable contribution, indicating the significance of the meteorological conditions. In strong contrast, geospatial variables such as location-long (0.000073), location-lat (0.000307), and other sensor-derived variables such as external temperature, GPS measurements, and tag-local-identifier play no notable roles. This suggests that even though the dataset contains spatial context and sensor data, the atmosphere and the dynamics of the wind primarily affect the performance of the model.

While the Random Forest model yielded extremely low prediction errors, making it incredibly effective for the current task of regression, the Gradient Boosting and the LSTM models had severe limitations under the current conditions. The Random Forest's success can be attributed to its inherent ensemble learning mechanism, which prevents variance and overfitting, whereas the larger values of the error of the Gradient Boosting model indicate that it could require further optimization or other approaches for optimum performance. Similarly, the LSTM model, although theoretically suitable for forecasting time-series, is less effective at comprehending the nuances of the migration data as of now, as indicated by its larger values of MSE and RMSE. These findings reiterate the importance of model selection and adjustment of the model's hyperparameters for predictive analysis, particularly when dealing with complex, high-dimensional, and temporally correlated data.

The relatively small variation observed in the predicted migration coordinates can be attributed to a combination of ecological and algorithmic factors. Firstly, since the dataset exclusively includes individuals from the same species the Taiga Goose migration behaviour is inherently similar. These birds are known to migrate in flocks and follow consistent, species-wide routes, which naturally limits variability in movement patterns. Secondly, the Random Forest model used for prediction tends to average outcomes across multiple decision trees. As a result, predictions are biased toward the most common migration paths present in the training data, especially when the majority of historical data points cluster around specific seasonal stopovers. Thirdly, although environmental variables such as wind speed, temperature, and speed were included in the model, their range and variance may not have been sufficient to induce significant shifts in predicted locations. Consequently, while the model does show sensitivity to changes—most notably in wind speed (v10)—the overall predictions remain within a narrow geographic margin. This suggests that the model has learned a stable representation of the dominant migration patterns in the dataset but may lack exposure to edge cases or atypical movements that could have produced more diverse outputs.

5 DISCUSSION

The Geospatial and meteorological data from over 840,000 events. It examines geolocation, temperature, GPS quality, flight dynamics, altitude, and wind speed variables. Robust outlier correction, clustering, and correlation analysis detect singular meteorological regimes and spatial patterns. Predictive models like Random Forest, Gradient Boosting, and LSTM are validated, with feature importance assigned to surface pressure and wind dynamics. The findings establish the significance of precise digital measurements and model selection on intricate environmental interactions, with insights into migration behaviour and predictive analysis of high-dimensional data. The findings inform improved forecasting.

5.1 Migration Path of The Taiga Goose

The migration path after the data is removed from the outliers is a well-defined and obvious corridor that the geese predominantly travel. In this refined visualization, the geese's distribution is denser, indicating that the geese migrate preferentially along specific routes instead of exhibiting a random dispersal across the landscape. One of the most intriguing observations of the cleaned graph is that most of the migration points cluster within a rather narrow latitude-longitude corridor. This suggests the occurrence of a preferred flyway of migration, possibly determined by local topographical features such as wetlands, river valleys, or fields that serve as reliable stopover sites. The clustering of the points along such corridors is also corroborated by earlier studies that have highlighted the importance of fixed migration corridors in birds, particularly those making use of known stopover sites for the replenishment of energy reserves (Hebblewhite & Haydon, 2010; Newton, 2008).

Also, the sanitized graph shows a lower level of noise in the dataset, allowing geodesic distances to be calculated with higher precision. Higher precision enables the identification of the most critical migratory segments and potential bottlenecks along the route. As a point of illustration, denser appearing locations could be the sites of geese clustering for foraging or resting, a situation recorded extensively in migration ecology research (Fox et al., 2014). Also, the coherence of the migration track on the processed graph indicates the effectiveness of the data-cleansing process in removing spurious points that might otherwise have exaggerated

travel distances or modified the true flight paths. In keeping with the research of Bridge et al. (2011), this kind of methodological accuracy is important for the development of reliable migratory behaviour models since it ensures the subsequent analysis is based on representative data.

The cleaned migration paths graph confirms that the Taiga Geese adhere to distinct, consistent migratory corridors. The visualization not only facilitates the accurate computation of migration distances but also provides valuable insights into the spatial ecology of these birds—insights that are critical for understanding their movement ecology and for informing conservation strategies aimed at preserving critical migratory habitats. The rigorous handling of outliers in the dataset further bolsters the credibility of the movement assessments. Outlier values in variables such as ground speed and altitude were carefully treated using robust statistical methods—values beyond the 1.5 IQR range were replaced by the median, and implausible GPS points (exceeding a 50 km geodesic jump) were removed. This preprocessing step is critical given that unfiltered outliers can distort estimates of central tendency and variability, as noted in previous migration studies (Dingle, 2014). By addressing these outliers, the study ensured that the resulting geodesic distance computations and visualizations reliably reflected the birds' true movement patterns.

5.2 Classification of Migratory Patterns Using K-Means Clustering

The second objective of the investigation was to group the migration patterns of the Taiga Goose based on the characteristics of the movement using K-Means clustering. The unsupervised learning method was employed for the wind parameters, i.e., the horizontal (u10) and vertical (v10) components of the wind known to influence migration. The justification for the use of K-Means is well documented in the literature; for example, since MacQueen's (1967) seminal work, K-Means clustering is effective at partitioning large data sets into homogeneous groups when the number of groups is known.

The clustering analysis yielded three groups for the wind parameters. Cluster 0 had a weakly negative horizontal (mean u10 ≈ -0.52) and a moderately positive vertical (mean v10 ≈ 3.45) component. Cluster 1 had a more negative horizontal (mean u10 ≈ -0.82) and a strongly negative vertical (mean v10 ≈ -4.92) component. Cluster 2 was distinctly different, with a strong positive horizontal (mean u10 ≈ 6.36) and near-neutral vertical (mean v10 ≈ 0.84) component. These

differences are important because they suggest that the Taiga Goose is exposed to, and can take advantage of, disparate regimes of wind during migration.

Supporting Alerstam and Lindström's (1990) Optimal Migration Theory, these characteristic wind conditions could influence the birds' routing and timing of flight. As a for-instance, Cluster 2's positive horizontal wind could be representative of beneficial tailwind conditions that decrease energy cost and facilitate long-distance travel. Conversely, the negative wind conditions of Cluster 1 could be associated with the challenge of headwind, potentially requiring birds to alter flight routes or consume extra energy. However, even though K-Means clustering presents a useful framework for the identification of such regimes, assumptions underlying the methods, such as the assumption of spherical clusters with homogeneous variance, may not fully capture the complexity of biological movement patterns (Shamoun-Baranes et al., 2016). In addition, the sensitivity of the technique to initial centroids and the number of clusters selected a priori suggest that other clustering techniques (e.g., DBSCAN) could be explored further to validate these results.

The clustering results also capture the most important characteristic of the ecology of migration: the dynamic responsiveness of the birds to varying environmental conditions. Not only do the clusters demarcate discrete conditions of the wind, but they also indicate that the birds could be following varying migration strategies. In accordance with research by Newton (2006) and Berthold (2001), such flexibility of behavior is vital for the success of migration, allowing birds to maximize energy efficiency, minimize risks, and adapt to varying environmental conditions. The fact that such behaviors can be categorized with machine learning techniques highlights the strength of data-driven approaches for providing detailed insights into migration strategies, bridging the gap between empirical observations and theoretical frameworks.

5.3 Development of Predictive Models for Migration Direction

The third objective was the formulation of predictive models for the prediction of the migration or direction of the Taiga Goose. In this research segment, a number of modelling techniques were employed, including Random Forests, Gradient Boosting, and Long Short-Term Memory (LSTM) networks. The Random Forest model was a robust predictor with a very low mean squared error (MSE) of 0.0075 and root mean squared error (RMSE) of 0.0868. In accordance with Breiman's (2001) assertion, the ensemble approach of Random Forests circumvents overfitting because predictions from several decision trees are averaged, hence retaining

complex non-linear relationships with high accuracy. The excellent performance metrics indicate that the Random Forest model was extremely effective at modeling the association of the input features with the geospatial coordinates. In accordance with previous research that has utilized Random Forests with success for application on ecological data, this result indicates the potential of Random Forests for predictive modeling (Gómez et al., 2019).

Conversely, the Gradient Boosting model, with a MultiOutputRegressor wrapper used for the multi-dimensional target of latitude and longitude, produced much higher error statistics (MSE: 23.8076, RMSE: 4.8793). In keeping with the findings of Friedman (2001), if Gradient Boosting is a powerful technique for making predictions progressively better with iterative refinement, it is also incredibly sensitive to the specification of the hyperparameters, as well as data noise. The relatively poor performance of the Gradient Boosting model here may be because the algorithm was not best optimized for the specific structure of the dataset or because the non-linear relationships present in the data on migration were better addressed by the Random Forest approach. In keeping with this interpretation, several studies have noted that boosting procedures can be less effective than other approaches for ecological forecasting when the data contain high levels of variation and when the key features are dominated by noise (Kays et al., 2015).

Aside from ensemble approaches, the investigation also explored the use of deep learning, specifically LSTM networks, for the detection of temporal structure in migration data. The LSTM model was constructed with two layers: a first with 64 units with a dropout layer preceding it, and a second with 32 units with a dropout layer preceding it, leading up to a dense output layer for predictions of latitude and longitude. The use of a sequence window of 7-time steps was intended to capture short-term structure. However, with a theoretically strong design and training regimen of 10 epochs with batch size 32, the LSTM model had a much higher RMSE of 15.5008 and MSE of 240.2746. In keeping with Hochreiter and Schmidhuber (1997), the LSTMs would be well suited for modeling sequence data; however, the large prediction error here suggests that the complex temporality of the migrant behaviors, which are prone to abrupt environmental changes and individual variations, may not have been well represented within a window of 7 days. Contrary to work by Gómez et al. (2019), showing the potential of the use of LSTMs for trajectory predictions under experimental settings, the finding here suggests that longer sequence windows, additional feature engineering, or hybrid model approaches may be required for the improvement of LSTM performance here.

Typically, the outcomes of predictive modeling here reflect the trade-offs of model complexity and predictive accuracy. The superior performance of the Random Forest model highlights the strength of ensemble methods for handling high-dimensional, noisy data characteristic of migration studies. In contrast, the comparatively low performance of the Gradient Boosting and LSTM models serves as a reminder that complex modeling strategies, though theoretically attractive, must be carefully optimized to the available data. These findings align with existing work highlighting the importance of model selection and tuning of hyperparameters for ecological forecasting (Chakraborty et al., 2020).

The feature importance analysis conducted in this study reveals that meteorological factors play a dominant role in shaping the migratory patterns of the Taiga Goose. In particular, surface pressure (sp) emerged as the most influential predictor, followed by the wind components u10 (horizontal wind speed) and v10 (vertical wind component). These findings underscore that the environmental conditions encountered during migration have a profound impact on flight dynamics and route selection. In line with Newton's (2008) work on migration ecology, the prominence of surface pressure and wind factors in our analysis suggests that these geese may rely heavily on favorable atmospheric conditions to optimize their energy expenditure and minimize travel time during migration. This finding supports earlier studies by Shamoun-Baranes et al. (2016) that highlighted the impact of weather conditions on migratory decisions, particularly the role of pressure systems in shaping migratory corridors.

5.4 Factors Influencing the Taiga Goose Migration

The importance of the horizontal components of the wind, as indicated by u10 and v10, further emphasizes the significance of meteorological dynamics. A strong horizontal wind component (u10) can act as a tailwind, reducing the energetic expense of migration by boosting the speed of flight. Similarly, the vertical wind component (v10) can influence the birds' efficiency of gaining or losing altitude, a consideration critical for the navigation of complex terrain and the harnessing of thermal updrafts on the migration route. In alignment with Alerstam and Lindström's (1990) Optimal Migration Theory, these aspects of the wind play a critical role because they directly influence the energy balance of the migrant birds, hence influencing their route and the timing of their take-off.

On the other hand, geospatial variables such as latitude and longitude and sensor-measured variables such as external temperature and GPS-specific values did not show much significance

in our model. This suggests that even though these variables play a critical role in observing the birds' migration path and routes, they play less of a direct role in the dynamics of the birds' migration than the meteorological variables. This aligns with the literature review, which believes that migration is primarily triggered by external environmental factors—i.e., weather, food, and photoperiod—rather than by fixed locations (Berthold, 2001; Alerstam, 2011).

Also, the relative insignificance of ambient temperature in the feature importance analysis may be surprising, given the review's emphasis on seasonal fluctuation and food availability. However, this result may be a function of the fact that across the conditions under which the Taiga Goose is migrating, temperature differences may be less significant or may be being picked up by other correlated meteorological variables, such as pressure on the surface and winds. In keeping with the conclusions of Hebblewhite and Haydon (2010), it is possible that even if temperature is a factor in migration, its effect may be secondary to the direct effects of pressure and wind on the birds' ability to fly.

5.5 Discussion Findings with Theoretical Frameworks

The findings of the research can be best understood with the general theoretical frameworks of bird migration. Optimal Migration Theory (Alerstam & Lindström, 1990) predicts that birds take the most energy-maximizing and time-maximizing routes, and the observed clustering based on the wind factors is corroborated by this concept. The individual clusters of this research most likely reflect the varied environmental regimes affecting the efficiency of migration. As illustrations, birds with beneficial tailwind conditions (as indicated by Cluster 2) can possibly migrate with greater efficiency, while birds with headwind conditions (Cluster 1) can be forced to change flight plans. These processes are consistent with the work of Newton (2006) and Berthold (2001), which highlighted the role of weather and wind on migration routes.

The predictive models developed here further extend the understanding of migration by attempting to foretell future locations and migration paths. In keeping with the Flyway Theory (Berthold, 2001), whereby birds follow fairly fixed migration routes, the high accuracy of the Random Forest model indicates that, at least with current environmental circumstances, migration paths can be predicted fairly accurately. The less impressive results of the Gradient Boosting and LSTM models, however, indicate that migration prediction remains a challenging task, particularly when unexpected environmental changes or complex behavioral responses

occur. In keeping with Kays et al. (2015), the combination of multi-source data and the careful selection of modeling methods is the key for overcoming such challenges.

Also, the emphasis of the research on robust data preprocessing—i.e., the detection and removal of outliers aligns with the migration literature recommendation. In accordance with Dingle (2014), the precision of migration analysis is heavily dependent on the track data quality. The methodological robustness of this research, especially with the treatment of outliers (i.e., the substitution of extreme values of ground speed and the removal of spurious GPS points), ensures that the subsequent analysis is based on solid ground. This is vital for descriptive analysis as for predictive analysis since noisy data can severely compromise the performance of sophisticated models (Robinson et al., 2010).

The findings of the feature importance analysis also correspond with general theoretical frameworks for migration. Optimal Migration Theory, for instance, suggests that birds take routes of minimum energy expenditure and travel time (Alerstam & Lindström, 1990). The high importance of pressure and wind for the models is substantiated by this theory because these variables directly influence the energetic cost of flight. Furthermore, the Flyway Theory, which emphasizes the significance of geographical corridors for migration (Berthold, 2001), is complemented by the current results: even though the routes exhibit spatial patterns, the environmental conditions along the routes' key predictors are the that determining migration success.

5.6 Conclusion

The present work meets the objectives of assessing the patterns of migration, classifying the migratory habits, and developing predictive models for the Taiga Goose. By carrying out stringent geodesic distance calculations and accurate spatial representations, the work confirms that the Taiga Goose uses well-defined migration corridors under the influence of environmental factors. The application of K-Means clustering is effective in segregating the changing wind regimes most likely to be responsible for migration, supporting theoretical frameworks such as Optimal Migration Theory and Flyway Theory. Predictive model results indicate that even though ensemble methods such as Random Forests can be incredibly accurate, other advanced methods such as Gradient Boosting and LSTM still require improvement for handling the complexities of migratory behaviour.

6 CONCLUSION

This dissertation was focused on analysing the migratory behaviour of the Taiga Goose with a holistic data-driven approach incorporating geospatial analysis, unsupervised clustering, and predictive modelling. In alignment with the objectives, the research first analysed the patterns of movement and migration paths using precise geodesic distance calculations and rich visualization, thus determining well-defined migration corridors and key stopover places. The rigorous data cleansing process of median-based outlier replacement and the removal of implausible GPS data ensured that the resulting trajectories correctly reflected the motion of the birds. These findings confirm earlier work (Bridge et al., 2011; Hebblewhite & Haydon, 2010) and prove that high-resolution telemetry data can uncover complex migration patterns.

The second objective was achieved with the use of KMeans clustering, which divided the movement data based on the components of the wind (v10 and u10). The analysis identified three distinct clusters corresponding with distinct wind regimes. Following Optimal Migration Theory (Alerstam & Lindström, 1990), the clusters suggest that the Taiga Geese can change their courses based on the variation of the winds—maximizing their energy expenditure and efficiency of flight. The results of the unsupervised clustering not only provided strong evidence of underlying behavioral modes but also emphasized the impact of environmental factors on migration choice.

The third task was the building of predictive models for predicting migration direction and geospatial location. Ensemble methods, namely the Random Forest model, had the highest predictive accuracy with extremely low error values (MSE: 0.0075, RMSE: 0.0868). The Gradient Boosting and LSTM models, with their theoretical potential, had higher error values, demonstrating the challenge of modelling complex, non-linear migratory patterns. Feature importance analysis revealed that meteorological variables, especially surface pressure (sp) and the wind components (u10 and v10), were the most important factors for migration. This is consistent with the significance of environmental factors on the patterns of movement and is consistent with previous work (Breiman, 2001; Friedman, 2001).

6.1 Recommendations

Several implications for future research and practice of conservation can be inferred from the findings. First, greater emphasis on the inclusion of temporal dynamics in the analysis is warranted. Whereas the LSTM model provided initial results on time-series forecasting, future research should explore longer sequence windows or hybrid strategies that combine traditional time-series analysis with deep learning for improved detection of abrupt environmental change and seasonal cycles.

Second, additional feature selection and tuning of the parameters is recommended for advanced predictive models. Even if Random Forest performed well, the suboptimal performance of Gradient Boosting and the LSTM models suggests that a more advanced approach—potentially incorporating the use of ensemble hybrid models or more robust hyperparameter optimization methods—would be required for improved predictive accuracy.

Third, the research must be extended spatially to encompass other environmental variables, such as the metrics of habitat quality and fine-scale weather forecasts. These data could provide a better understanding of the drivers of migration at the scale of ecology and allow for the development of improved conservation strategies.

Future studies should combine data from several sources to complement GPS telemetry. Coupling high-resolution sensor data with remote sensing imagery and environmental data (including land cover, vegetation indices, and climatic variables) could offer a richer understanding of migration ecology. Integrative frameworks that couple Geographic Information Systems (GIS) with machine learning can potentially offer higher predictive power (Kays et al., 2015). The multidimensional approach would allow for the detection of environmental drivers with higher precision, making migration forecasting models more accurate.

It is recommended that longitudinal studies, which follow continuously across years and seasons, be conducted. Adaptive sampling designs, which change the frequency of following based on critical migration seasons or environmental signals, would be better equipped to capture finer-scale temporal variation of the migration patterns. Inter-annual variation and long-term trends could be examined with longitudinal data, which would be important for establishing the impacts of climate change and habitat alteration on migration corridors (Newton, 2008).

Interdisciplinary collaboration among ecologists, data scientists, and remote sensing specialists can enhance the methodological strength and real-world relevance of migration studies. Synthesizing insights from diverse disciplines can encourage the development of new analytical frameworks that can capture the richness of the data better. For instance, collaborations can enable the use of hybrid modelling frameworks that combine mechanistic and machine learning, gaining a better understanding of migration processes and enabling improved conservation decision-making (Gómez et al., 2019).

These additional recommendations aim to widen the scope of future research and maximize the precision and relevance of research on migratory behaviour, ultimately resulting in better ecological forecasting and effective conservation management.

6.2 Contribution to Knowledge

This project is making several significant contributions to the fields of movement ecology and the analysis of avian migration. First, it demonstrates the strength of a robust, end-to-end data science workflow that brings together geospatial analysis, unsupervised clustering, and predictive modelling. By breaking down each step, from data ingestion and cleansing to model assessment, this project provides a reproducible framework for the analysis of migration patterns for other taxa.

Second, the demarcation of distinct wind regimes by KMeans clustering offers fresh insights into the behavioral ecology of the Taiga Goose, revealing that the birds adapt their migration tactics based on varying environmental conditions. The finding corroborates and enhances the existing hypotheses, such as the Optimal Migration Theory and Flyway Theory, by relating specific meteorological factors with the observed migration.

Third, the feature importance analysis highlights the dominant influence of surface pressure and wind components on migration. Not only does this sophisticated understanding of environmental drivers advance the scientific discourse, but it also offers practical guidance for the development of targeted conservation strategies.

Finally, by comparing a number of predictive modelling strategies, the research emphasizes the relative strengths and weaknesses of ensemble methods relative to deep learning for the application of ecological forecasting. Comparative evaluation of this nature is of special usefulness for scholars looking to utilize advanced analytics for the analysis of animal migration, especially for those with high data volatility and non-linear, complex relationships.

6.3 Limitations

This study, although it contributes, has some limitations. First, the dataset, though large, is limited by the inherent limitations of GPS telemetry, for instance, the short lifespan of the batteries and potential signals being lost in remote areas. This can cause gaps or biases in the migration routes recorded. Second, the application of the KMeans clustering assumes spherical clusters with homogeneous variances, which may not be representative of the complex, non-linear nature of migration routes. Third, though the Random Forest model was of high accuracy, the predictive power of the Gradient Boosting and the LSTM models was less than ideal, indicating the need for further improvement of model structure and the adjustment of the hyperparameters. Lastly, the work focused primarily on the spatial patterns and did not fully integrate the dynamics of time, which is critical for the understanding of migration according to seasonal and environmental variation. Future work should address these gaps for the improvement of model robustness and generalizability. The full dataset wasn't also explored to a desirable extent as most of the processes and dataset are optimized.

6.4 Implications for Future Research

The results of this study have important implications for migration studies and conservation. The ability of accurately assess the patterns of migration and classify distinctive migration behaviours provides valuable information on the navigation of complex habitats by Taiga Geese. The information is critical for the development of conservation strategies, particularly with the increasing degradation of habitats and climatic change. In consonance with Madsen et al. (2019), understanding migration routes and the selection of stopover sites can be utilized for the demarcation of protected areas and the implementation of strategies for the mitigation of human-induced disturbances.

Also, the strong predictive power of the Random Forest model suggests that ensemble methods would be especially well adapted for operational migration forecasting. This potential is especially useful for dynamic contexts, where real-time predictions would be useful for the management of conflicts between conservation objectives and human activities, such as agriculture or urban development. However, the challenges with Gradient Boosting and LSTM methods also suggest the necessity for further research into hybrid and adaptive modelling strategies, which can better cope with the nonlinear, multidimensional nature of migratory data.

The predictive modelling discussion here also highlights the importance of the integration of both spatial and temporal analysis. While geodesic distance calculations and maps of the spatial domain present a snapshot of migration routes, the incorporation of time-series analysis—as experimented with the application of LSTM networks represents a critical, if challenging, horizon for migration research. Future work would be well advised to expand the time window and integrate other contextual variables (e.g., weather forecasts, health of the habitat indices) for the improvement of deep learning model performance. Following the approach of Chakraborty et al. (2020), such integrated approaches would be likely to make improved, actionable predictions, ultimately informing better management and conservation.

7 REFERENCES

Abrahms, B., Hazen, E. L., Bograd, S. J., Brashares, J. S., Robinson, P. W., Scales, K. L., ... & Costa, D. P. (2019). Dynamic ensemble models to predict distributions and anthropogenic risk exposure for highly mobile species. Diversity and Distributions, 25(8), 1182-1193.

Alerstam, T. (2011). Bird migration. Cambridge University Press.

Alerstam, T., & Lindström, Å. (1990). Optimal bird migration: The relative importance of time, energy, and safety. Ornis Scandinavica, 21(3), 167–177.

Bauer, S., Barta, Z., Ens, B. J., Hays, G. C., McNamara, J. M., & Klaassen, M. (2019). Animal migration: Linking models and data beyond taxonomic limits. Biological Reviews, 94(3), 1028-1044.

Berthold, P. (2001). Bird migration: A general survey. Oxford University Press.

Berthold, P. (2001). Bird migration: A general survey. Oxford University Press.

Boyle, W. A. (2010). Altitudinal bird migration in North America. The Auk, 127(2), 296–306.

Bridge, E. S., Kelly, J. F., Contina, A., Gabrielson, R. M., MacCurdy, R. B., & Winkler, D. W. (2011). Advances in tracking small migratory birds: A technical review of light-level geolocation. Journal of Field Ornithology, 82(3), 239–248.

Bridge, E. S., Kelly, J. F., Contina, A., Gabrielson, R. M., MacCurdy, R. B., & Winkler, D. W. (2011). Technology on the move: Recent and forthcoming innovations for tracking migratory birds. BioScience, 61(9), 689-698. https://doi.org/10.1525/bio.2011.61.9.7

Chakraborty, T., Joshi, A., & Verma, S. (2020). Time series analysis and forecasting of bird migration patterns using ARIMA models. Ecological Modelling, 421, 108967.

Chandler, R.B., Crawford, D.A., Garrison, E.P., Miller, K.V. and Cherry, M.J. (2021). Modeling abundance, distribution, movement, and space use with camera and telemetry data. Ecology, 103(10). https://doi.org/10.1002/ecy.3583.

Chapman, B. B., Brönmark, C., Nilsson, J. Å., & Hansson, L. A. (2011). Partial migration: An introduction. Philosophical Transactions of the Royal Society B, 365(1553), 2015–2022.

Cochran, W. W. (1980). Wildlife telemetry. Wildlife Management Techniques Manual, 4, 507–520.

Dingle, H. (2014). Migration: The biology of life on the move. Oxford University Press.

Dodge, S., Bohrer, G., Weinzierl, R., Davidson, S. C., Kays, R., Douglas, D. C., & Wikelski, M. (2014). The environmental-data automated track annotation (Env-DATA) system: Linking animal tracks with environmental data. Movement Ecology, 2(1), 3.

Eichhorn, G., Drent, R. H., Stahl, J., Leito, A., & Alerstam, T. (2017). Spring stopover routines in geese: A question of condition? Journal of Avian Biology, 48(5), 657–665.

Fox, A. D., Eide, N. E., Bergersen, E., & Madsen, J. (2014). Predicting impacts of climate change on goose migration routes.

Fox, A. D., Elmberg, J., Tombre, I. M., & Hessel, R. (2014). Agriculture and herbivorous waterfowl: A review of the scientific basis for improved management. Biological Reviews, 89(1), 153-166.

Ganaie, M.A., Tanveer, M., Ponnuthurai Nagaratnam Suganthan and Václav Snášel (2022). Oblique and rotation double random forest. Neural Networks, 153(0), pp.496–517. https://doi.org/10.1016/j.neunet.2022.06.012.

Gienapp, P., Leimu, R., & Merilä, J. (2005). Responses to climate change in avian migration timing: Meta-analysis of 50 years of data. Journal of Animal Ecology, 74(2), 271–281.

Gill, F. B. (2007). Ornithology (3rd ed.). W. H. Freeman.

Gill, R. E., Tibbitts, T. L., Douglas, D. C., Handel, C. M., Mulcahy, D. M., Gottschalck, J. C., ... & Piersma, T. (2009). Extreme endurance flights by landbirds crossing oceanic barriers. Science, 323(5916), 925–928.

Global Change Biology, 20(5), 1745–1754.

Gómez, C., Tenan, S., Jiguet, F., & Bouten, W. (2019). Predicting bird migration patterns using deep learning models. Ecological Informatics, 53, 100973.

Gorard, S. (2020). Handling missing data in numeric analyses. International Journal of Social Research Methodology, 23(6), pp.1–10. https://doi.org/10.1080/13645579.2020.1729974.

Guilford, T., Meade, J., Freeman, R., Biro, D., Evans, T., Bonadonna, F., & Boyle, D. (2011). A dispersive migration of the lesser black-backed gull Larus fuscus: Satellite tracking and repeatability. Journal of Avian Biology, 42(6), 479-486.

Gwinner, E. (1996). Circannual rhythms in bird migration: Control of timing and routes. Journal of Experimental Biology, 199(1), 39–48.

Hartwig, F.P., Davey Smith, G., Schmidt, A.F., Sterne, J.A.C., Higgins, J.P.T. and Bowden, J. (2020). The median and the mode as robust meta-analysis estimators in the presence of small-study effects and outliers. Research Synthesis Methods, 11(3), pp.397–412. https://doi.org/10.1002/jrsm.1402.

Hays, G. C., Christensen, A., Fossette, S., Schofield, G., Talbot, J., & Mariani, P. (2016). Route optimisation and solving Zermelo's navigation problem during long distance migration in cross flows. Ecology Letters, 19(6), 646-654.

Hebblewhite, M., & Haydon, D. T. (2010). Distinguishing technology from biology: A critical review of the use of GPS telemetry data in ecology. Philosophical Transactions of the Royal Society B, 365(1550), 2303–2312.

Hu, J., Liang, Y., Fan, Z., Liu, L., Yin, Y. and Zimmermann, R. (2024). Decoupling Long-and Short-Term Patterns in Spatiotemporal Inference. IEEE Transactions on Neural Networks and Learning Systems, 0(0), pp.1–13. https://doi.org/10.1109/tnnls.2023.3293814.

Huang, G., Hu, W., Du, J., Jia, Y., Zhou, Z., Lei, G., Saintilan, N., Wen, L. and Wang, Y. (2025). Identification and scenario-based optimization of ecological corridor networks for waterbirds in typical coastal wetlands. Ecological Indicators, [online] 171(0), p.113147. https://doi.org/10.1016/j.ecolind.2025.113147.

Huang, W., Peng, Y., Ge, Y. and Kong, W. (2021). A new Kmeans clustering model and its generalization are achieved by joint spectral embedding and rotation. PeerJ Computer Science, 7(0), pp.e450–e450. https://doi.org/10.7717/peerj-cs.450.

Hunter, L.M. and Simon, D.H. (2022). Time to Mainstream the Environment into Migration Theory? International Migration Review, 57(1), p.019791832210743. https://doi.org/10.1177/01979183221074343.

Ilhan, E., Turali, M.Y. and Kozat, S.S. (2023). Gradient Boosting With Moving-Average Terms for Nonlinear Sequential Regression. IEEE Signal Processing Letters, 30(0), pp.1182–1186. https://doi.org/10.1109/lsp.2023.3309577.

Jensen, R. A., Madsen, J., Johnson, F. A., & Tamstorf, M. P. (2016). Snowmelt and climate change drive changes in goose migration timing. Global Change Biology, 22(10), 3963–3971.

Johnston, A., Fink, D., Reynolds, M. D., Hochachka, W. M., Sullivan, B. L., & Pardieck, K. L. (2019). Comparing citizen science and professional data sources to model avian distribution. Diversity and Distributions, 25(1), 167–179.

Karrar, A.E. (2022). The Effect of Using Data Pre-Processing by Imputations in Handling Missing Values. Indonesian Journal of Electrical Engineering and Informatics (IJEEI), [online] 10(2). https://doi.org/10.52549/ijeei.v10i2.3730.

Kashif, M. (2023). Classifying Tweets with Keras and TensorFlow using RNN (Bi-LSTM). Lahore Garrison University Research Journal of Computer Science and Information Technology, [online] 7(02), pp.12–16. https://doi.org/10.54692/lgurjcsit.2023.0702455.

Kays, R., Crofoot, M. C., Jetz, W., & Wikelski, M. (2015). Terrestrial animal tracking as an eye on life and planet. Science, 348(6240), aaa2478.

Kenward, R. E. (2001). A manual for wildlife radio tagging. Academic Press.

Li, D., Hu, X., Rollo, J., Luther, M., Lu, M. and Liu, C. (2025). Spatial Cluster Characteristics of Land Surface Temperatures. Sustainability, [online] 17(6), pp.2653–2653. https://doi.org/10.3390/su17062653.

Marques, H.O., Swersky, L., Sander, J., Ricardo and Zimek, A. (2023). On the evaluation of outlier detection and one-class classification: a comparative study of algorithms, model selection, and ensembles. Data Mining and Knowledge Discovery, 37(4), pp.1473–1517. https://doi.org/10.1007/s10618-023-00931-x.

Mehlman, D. W., Mabey, S. E., Ewert, D. N., Duncan, C., Able, K. P., Cimprich, D. A., ... & Woodrey, M. S. (2005). Conserving stopover sites for forest-dwelling migratory landbirds. The Auk, 122(4), 1281-1290.

McCabe, J.D., 2015. *Explaining migratory behaviors using optimal migration theory*. PhD thesis. University of Maine. Available at: https://digitalcommons.library.umaine.edu/etd/2360 [Accessed 18 Apr. 2025].

Nathan, R., Getz, W. M., Revilla, E., Holyoak, M., Kadmon, R., Saltz, D., & Smouse, P. E. (2008). A movement ecology paradigm for unifying organismal movement research. Proceedings of the National Academy of Sciences, 105(49), 19052-19059. https://doi.org/10.1073/pnas.0800375105

Newton, I. (2008). The migration ecology of birds. Academic Press.

Newton, I. (2010). The migration ecology of birds. Academic Press.

Pei, Y. and Ye, L. (2022). Cluster analysis of MNIST data set. Journal of Physics: Conference Series, 2181(1), p.012035. https://doi.org/10.1088/1742-6596/2181/1/012035.

Runge, C. A., Martin, T. G., Possingham, H. P., Willis, S. G., & Fuller, R. A. (2015). Conserving mobile species. Frontiers in Ecology and the Environment, 12(7), 395-402.

Schöttler, S., Yang, Y., Pfister, H. and Bach, B. (2021). Visualizing and Interacting with Geospatial Networks: A Survey and Design Space. Computer Graphics Forum, 0(0). https://doi.org/10.1111/cgf.14198.

Shamoun-Baranes, J., van Loon, E., Alon, D., Åkesson, S., Soerensen, H., & Sapir, N. (2016). Bird migration and environmental conditions: An advanced modeling approach for predicting movement patterns. Ecological Modelling, 320, 339-348.

Singh, G. and Kundu, S. (2022). Outlier and Trend Detection Using Approximate Median and Median Absolute Deviation. International Conference on Computational Intelligence and Networks (CINE), 2019-Mar(0), pp.01–06. https://doi.org/10.1109/cine56307.2022.10037489.

Sullivan, B. L., Wood, C. L., Iliff, M. J., Bonney, R. E., Fink, D., & Kelling, S. (2014). eBird: A citizen-based bird observation network in the biological sciences. Biological Conservation, 169, 31-40.

Walker, J.D., Letcher, B.H., Rodgers, K.D., Muhlfeld, C.C. and D'Angelo, V.S. (2020). An Interactive Data Visualization Framework for Exploring Geospatial Environmental Datasets and Model Predictions. Water, 12(10), p.2928. https://doi.org/10.3390/w12102928.

Wang, Y., Wu, H., Zhang, J., Gao, Z., Wang, J., Yu, P. and Long, M. (2022). PredRNN: A Recurrent Neural Network for Spatiotemporal Predictive Learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, 0(0), pp.1–1. https://doi.org/10.1109/tpami.2022.3165153.

Wilcove, D. S., & Wikelski, M. (2008). Going, going, gone: Is animal migration disappearing? PLoS Biology, 6(7), e188. https://doi.org/10.1371/journal.pbio.0060188

8 APPENDICES

APPENDIX HEADING 1 **ERROR! BOOKMARK NOT DEFINED.**

APPENDIX 1 EXAMPLE TITLE **ERROR! BOOKMARK NOT DEFINED.**

APPENDIX 2 **ERROR! BOOKMARK NOT DEFINED.**

APPENDIX 1 JUPYTER NOTEBOOK OUTPUT

[7]:
```
import pandas as pd

# Loading my first dataset (taiga geese)
goose_df = pd.read_csv("/Users/admin/Downloads/geese.csv")
######## Exploring the dataset(EDA)##########

# Displaying first few rows

display(goose_df.head())

# Printing column names for reference
print("✅ Column Names in Bird Tracking Dataset:", goose_df.columns)
```

	event-id	visible	timestamp	location-long	location-lat	external-temperature	gps:hdop	gps:satellite-count	ground-speed	h
0	17714159009	True	2019-06-01 00:01:00.000	28.611073	66.863953	7.0	1.3	5.0	1.666668	
1	17714159015	True	2019-06-01 00:11:00.000	28.611586	66.863785	7.0	1.2	5.0	0.000000	
2	17714159020	True	2019-06-01 00:21:00.000	28.611605	66.863754	8.0	1.2	5.0	0.277778	
3	17714159025	True	2019-06-01 00:31:00.000	28.611860	66.863785	7.0	1.1	5.0	0.000000	
4	17714159031	True	2019-06-01 00:41:00.000	28.611790	66.863716	14.0	1.1	5.0	0.000000	

```
✅ Column Names in Bird Tracking Dataset: Index(['event-id', 'visible', 'timestamp', 'location-long',
'location-lat',
       'external-temperature', 'gps:hdop', 'gps:satellite-count',
       'ground-speed', 'heading', 'height-above-ellipsoid', 'sensor-type',
       'individual-taxon-canonical-name', 'tag-local-identifier',
       'individual-local-identifier', 'study-name'],
      dtype='object')
```

[9]:
```
# Checking for missing values in each column
print("\n=== Missing Values ===")
print(goose_df.isnull().sum())
```

```
=== Missing Values ===
event-id                      0
visible                       0
timestamp                     0
location-long              6082
location-lat               6082
external-temperature      36397
gps:hdop                  36397
gps:satellite-count       36397
ground-speed              36397
heading                   36397
height-above-ellipsoid    36397
```

```
missing_percent = goose_df.isnull().mean() * 100
print("\n=== Percentage of Missing Values ===")
print(missing_percent)
```

```
=== Percentage of Missing Values ===
event-id                            0.000000
visible                             0.000000
timestamp                           0.000000
location-long                       0.718663
location-lat                        0.718663
external-temperature                4.300751
gps:hdop                            4.300751
gps:satellite-count                 4.300751
ground-speed                        4.300751
heading                             4.300751
height-above-ellipsoid              4.300751
sensor-type                         0.000000
individual-taxon-canonical-name     0.000000
tag-local-identifier                0.000000
individual-local-identifier         0.000000
study-name                          0.000000
dtype: float64
```

[4]:
```
# Dropping rows with missing essential location data
goose_df_clean = goose_df.dropna(subset=["location-long", "location-lat"])
print("\n=== After Dropping Rows with Missing GPS Coordinates ===")
print(goose_df_clean.isnull().sum())
```

```
=== After Dropping Rows with Missing GPS Coordinates ===
event-id                                0
visible                                 0
timestamp                               0
location-long                           0
location-lat                            0
external-temperature                36272
gps:hdop                            36272
gps:satellite-count                 36272
ground-speed                        36272
heading                             36272
height-above-ellipsoid              36272
sensor-type                             0
individual-taxon-canonical-name         0
tag-local-identifier                    0
individual-local-identifier             0
study-name                              0
dtype: int64
```

[5]:
```
#  Imputing missing values in sensor columns with the median value.
# Defining the list of columns to impute.
sensor_columns = ["external-temperature", "gps:hdop", "gps:satellite-count", "ground-speed", "heading"
```

[6]:
```
#  filling missing values with the column's median.
for col in sensor_columns:
    if goose_df_clean[col].isnull().sum() > 0:
        median_val = goose_df_clean[col].median()
        goose_df_clean.loc[:, col] = goose_df_clean[col].fillna(median_val)
```

[7]:
```
# Verifying that missing values have been imputed in the sensor columns
print("\n=== Missing Values in Sensor Columns After Imputation ===")
print(goose_df_clean[sensor_columns].isnull().sum())
```

```
=== Missing Values in Sensor Columns After Imputation ===
external-temperature      0
gps:hdop                  0
gps:satellite-count       0
ground-speed              0
heading                   0
height-above-ellipsoid    0
dtype: int64
```

[8]:
```
# resaving the cleaned dataset
clean_file_path = "/Users/admin/Downloads/goose_cleaned.csv"
goose_df_clean.to_csv(clean_file_path, index=False)
print(f"\n✅ Cleaned goose dataset saved as: {clean_file_path}")
```

```
✅ Cleaned goose dataset saved as: /Users/admin/Downloads/goose_cleaned.csv
```

[9]:
```
pip install xarray netCDF4 h5netcdf pandas numpy
```

```
Requirement already satisfied: xarray in /opt/anaconda3/lib/python3.12/site-packages (2025.1.2)
```

[10…]
```python
import xarray as xr

# Define file paths (update if needed)
file1_path = "/Users/admin/Downloads/data_stream-oper_stepType-accum.nc"  # Precipitation
file2_path = "/Users/admin/Downloads/data_stream-oper_stepType-instant.nc"  # Wind & Temperature

# Load NetCDF files
ds1 = xr.open_dataset(file1_path)  # Dataset 1: Precipitation
ds2 = xr.open_dataset(file2_path)  # Dataset 2: Wind & Temperature

# Print dataset details
print("✅ ERA5 Dataset 1 Structure (Precipitation):")
print(ds1)

print("\n✅ ERA5 Dataset 2 Structure (Wind & Temperature):")
print(ds2)
```

```
✅ ERA5 Dataset 1 Structure (Precipitation):
<xarray.Dataset> Size: 143MB
Dimensions:     (valid_time: 2193, latitude: 81, longitude: 201)
Coordinates:
    number      int64 8B ...
  * valid_time  (valid_time) datetime64[ns] 18kB 2019-01-01T03:00:00 ... 2020...
  * latitude    (latitude) float64 648B 70.0 69.75 69.5 ... 50.5 50.25 50.0
  * longitude   (longitude) float64 2kB -10.0 -9.75 -9.5 ... 39.5 39.75 40.0
    expver      (valid_time) <U4 35kB ...
Data variables:
    tp          (valid_time, latitude, longitude) float32 143MB ...
Attributes:
    GRIB_centre:             ecmf
    GRIB_centreDescription:  European Centre for Medium-Range Weather Forecasts
    GRIB_subCentre:          0
    Conventions:             CF-1.7
    institution:             European Centre for Medium-Range Weather Forecasts
    history:                 2025-02-24T00:03 GRIB to CDM+CF via cfgrib-0.9.1...

✅ ERA5 Dataset 2 Structure (Wind & Temperature):
<xarray.Dataset> Size: 571MB
Dimensions:     (valid_time: 2193, latitude: 81, longitude: 201)
Coordinates:
    number      int64 8B ...
  * valid_time  (valid_time) datetime64[ns] 18kB 2019-01-01T03:00:00 ... 2020...
  * latitude    (latitude) float64 648B 70.0 69.75 69.5 ... 50.5 50.25 50.0
  * longitude   (longitude) float64 2kB -10.0 -9.75 -9.5 ... 39.5 39.75 40.0
    expver      (valid_time) <U4 35kB ...
Data variables:
    u10         (valid_time, latitude, longitude) float32 143MB ...
    v10         (valid_time, latitude, longitude) float32 143MB ...
    t2m         (valid_time, latitude, longitude) float32 143MB ...
    sp          (valid_time, latitude, longitude) float32 143MB ...
Attributes:
    GRIB_centre:             ecmf
    GRIB_centreDescription:  European Centre for Medium-Range Weather Forecasts
    GRIB_subCentre:          0
    Conventions:             CF-1.7
    institution:             European Centre for Medium-Range Weather Forecasts
    history:                 2025-02-24T00:03 GRIB to CDM+CF via cfgrib-0.9.1...
```

[11…]
```python
import pandas as pd

# Convert NetCDF to Pandas DataFrame
df1 = ds1.to_dataframe().reset_index()  # Convert precipitation dataset
df2 = ds2.to_dataframe().reset_index()  # Convert wind & temperature dataset

# Display first few rows
print("\n=== Sample of Precipitation Data ===")
print(df1.head())

print("\n=== Sample of Wind & Temperature Data ===")
print(df2.head())
```

```
=== Sample of Precipitation Data ===
           valid_time  latitude  longitude  number expver        tp
0 2019-01-01 03:00:00      70.0     -10.00       0   0001  0.000006
1 2019-01-01 03:00:00      70.0      -9.75       0   0001  0.000007
2 2019-01-01 03:00:00      70.0      -9.50       0   0001  0.000008
3 2019-01-01 03:00:00      70.0      -9.25       0   0001  0.000009
4 2019-01-01 03:00:00      70.0      -9.00       0   0001  0.000009

=== Sample of Wind & Temperature Data ===
           valid_time  latitude  longitude  number expver       u10  \
0 2019-01-01 03:00:00      70.0     -10.00       0   0001  6.236557
1 2019-01-01 03:00:00      70.0      -9.75       0   0001  6.176987
```

[12…]
```
# Convert time column to datetime format
df1["valid_time"] = pd.to_datetime(df1["valid_time"])
df2["valid_time"] = pd.to_datetime(df2["valid_time"])

# Filter for 2019-2020 period
df1 = df1[(df1["valid_time"] >= "2019-01-01") & (df1["valid_time"] <= "2020-12-31")]
df2 = df2[(df2["valid_time"] >= "2019-01-01") & (df2["valid_time"] <= "2020-12-31")]

print("\n✅ Data filtered for 2019-2020 period.")
```

```
✅ Data filtered for 2019-2020 period.
```

[13…]
```
# Merge on timestamp (valid_time), latitude, and longitude
era5_merged = pd.merge(df1, df2, on=["valid_time", "latitude", "longitude"], how="inner")

print("\n✅ Merged ERA5 dataset structure:")
print(era5_merged.head())

# Save the merged dataset with a new name to avoid overwriting
era5_merged.to_csv("/Users/admin/Downloads/ERA5_Merged_New.csv", index=False)
print("\n✅ ERA5 Merged Dataset Saved: /Users/admin/Downloads/ERA5_Merged_New.csv")
```

```
✅ Merged ERA5 dataset structure:
           valid_time  latitude  longitude  number_x expver_x        tp  \
0 2019-01-01 03:00:00      70.0     -10.00         0     0001  0.000006
1 2019-01-01 03:00:00      70.0      -9.75         0     0001  0.000007
2 2019-01-01 03:00:00      70.0      -9.50         0     0001  0.000008
3 2019-01-01 03:00:00      70.0      -9.25         0     0001  0.000009
4 2019-01-01 03:00:00      70.0      -9.00         0     0001  0.000009

   number_y expver_y       u10        v10         t2m          sp
0         0     0001  6.236557  -9.269714  269.363037  102611.875
1         0     0001  6.176987  -9.587097  269.529053  102584.875
2         0     0001  6.036362  -9.864441  269.661865  102567.875
3         0     0001  5.930893 -10.132019  269.777100  102537.875
4         0     0001  5.837143 -10.395691  269.882568  102501.875

✅ ERA5 Merged Dataset Saved: /Users/admin/Downloads/ERA5_Merged_New.csv
```

[14…]
```
import pandas as pd

# Load the merged ERA5 dataset
era5_file_path = "/Users/admin/Downloads/ERA5_Merged_New.csv"
era5_df = pd.read_csv(era5_file_path)

# Check for missing values
print("\n✅ Missing values per column:")
print(era5_df.isnull().sum())
```

```
✅ Missing values per column:
valid_time    0
latitude      0
longitude     0
number_x      0
expver_x      0
tp            0
number_y      0
expver_y      0
u10           0
v10           0
t2m           0
sp            0
dtype: int64
```

[15…]
```
# Drop unnecessary columns
era5_df = era5_df.drop(columns=["number", "expver"], errors="ignore")

print("\n✅ Unnecessary columns removed.")
```

```
✅ Unnecessary columns removed.
```

[16…]
```
# Convert valid_time to datetime format
era5_df["valid_time"] = pd.to_datetime(era5_df["valid_time"])

print("\n✅ Timestamp format verified.")
```

```
✅ Timestamp format verified.
```

[17…]
```
# Save the cleaned dataset
era5_df.to_csv("/Users/admin/Downloads/ERA5_Cleaned_Final.csv", index=False)
```

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
warnings.filterwarnings("ignore", category=UserWarning)

dff=pd.read_csv("C:/Users/USER/Downloads/Merged_Taiga_Goose_ERA5_Cleaned (1).csv")
```

[2]:
```
dff.info()
```

```
 12  individual-taxon-canonical-name  840212 non-null  object
 13  tag-local-identifier             840212 non-null  int64
 14  individual-local-identifier      840212 non-null  object
 15  study-name                       840212 non-null  object
 16  latitude                         840212 non-null  float64
 17  longitude                        840212 non-null  float64
 18  era5_nearest_time                840212 non-null  object
 19  number_x                         840212 non-null  float64
 20  expver_x                         840212 non-null  float64
 21  tp                               840212 non-null  float64
 22  number_y                         840212 non-null  float64
 23  expver_y                         840212 non-null  float64
 24  u10                              840212 non-null  float64
 25  v10                              840212 non-null  float64
 26  t2m                              840212 non-null  float64
 27  sp                               840212 non-null  float64
dtypes: bool(1), float64(19), int64(2), object(6)
memory usage: 173.9+ MB
```

[3]:
```
print(dff.describe())
```

```
           event-id  location-long   location-lat  external-temperature  \
count  8.402120e+05  840212.000000  840212.000000         840212.000000
mean   1.772610e+10      49.211483      72.101289             13.124546
std    5.728257e+06      12.395345       4.276370              5.988180
min    1.770443e+10      13.525927      55.470840             -5.000000
25%    1.772661e+10      52.764491      72.351357              9.000000
50%    1.772682e+10      55.167532      73.950688             12.000000
75%    1.772731e+10      56.019547      74.794647             16.000000
max    1.773164e+10      58.737148      75.674942             53.000000

            gps:hdop  gps:satellite-count   ground-speed       heading  \
count  840212.000000        840212.000000  840212.000000  840212.00000
mean        1.079257             7.527806       0.494217     178.75023
std         0.346547             2.571375      20.015578     102.34053
min         0.500000             3.000000       0.000000       0.00000
25%         0.800000             6.000000       0.000000      92.00000
50%         1.000000             7.000000       0.000000     178.00000
75%         1.300000             9.000000       0.000000     265.00000
max        15.900000            22.000000   18133.347840     503.00000

       height-above-ellipsoid  tag-local-identifier  ...      longitude  \
count           840212.000000         840212.000000  ...  840212.000000
mean                60.980211         196691.265128  ...       2.984792
std                156.703511           4695.026790  ...      21.666961
min              -2000.000000         191166.000000  ...     -10.000000
25%                  5.000000         191181.000000  ...      -9.800000
50%                 36.000000         200679.000000  ...      -9.500000
75%                 95.000000         200694.000000  ...      39.800000
max               9984.000000         200712.000000  ...      40.000000
```

```
       number_x  expver_x             tp  number_y  expver_y            u10  \
count  840212.0  840212.0  840212.000000  840212.0  840212.0  840212.000000
mean        0.0       1.0       0.000126       0.0       1.0       1.276835
std         0.0       0.0       0.000269       0.0       0.0       4.496161
min         0.0       1.0       0.000000       0.0       1.0     -12.618123
25%         0.0       1.0       0.000007       0.0       1.0      -1.507991
50%         0.0       1.0       0.000034       0.0       1.0       1.083399
75%         0.0       1.0       0.000131       0.0       1.0       3.828882
max         0.0       1.0       0.002760       0.0       1.0      18.274409

                v10           t2m             sp
count  840212.000000  840212.000000  840212.000000
mean        0.220732     285.084562  100734.901449
std         4.962832       2.907339    1100.224470
min       -17.047119     270.717283   97215.770000
25%        -2.929265     283.469403   99965.500000
50%         0.179840     285.154133  100780.103333
75%         3.516968     286.705327  101651.396667
max        14.988022     296.515540  103462.353333

[8 rows x 21 columns]
```

[4]:
```python
# List of numerical columns to clean
numerical_cols = dff.select_dtypes(include=['number']).columns

def replace_outliers(dff, cols, method="median"):
    cleaned_df = dff.copy()
    for col in cols:
        Q1 = cleaned_df[col].quantile(0.25)
        Q3 = cleaned_df[col].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        if method == "median":
            replacement = cleaned_df[col].median()
        elif method == "mean":
            replacement = cleaned_df[col].mean()
        else:
            continue

        cleaned_df[col] = np.where((cleaned_df[col] < lower_bound) | (cleaned_df[col] > upper_bound), replacement, cleaned_df[col])
    return cleaned_df

df_cleaned = replace_outliers(dff, numerical_cols, method="median")  # Replace outliers with median

# Save the cleaned dataset
df_cleaned.to_csv("nw_data.csv", index=False)

print(f"Original dataset size: {dff.shape[0]} rows")
print(f"Cleaned dataset size: {df_cleaned.shape[0]} rows")
```

```
Original dataset size: 840212 rows
Cleaned dataset size: 840212 rows
```

[5]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 8))
sns.scatterplot(
    data=df_cleaned,
    x="location-long",
    y="location-lat",
    hue="individual-local-identifier",
    alpha=0.5,
    palette="viridis"
```

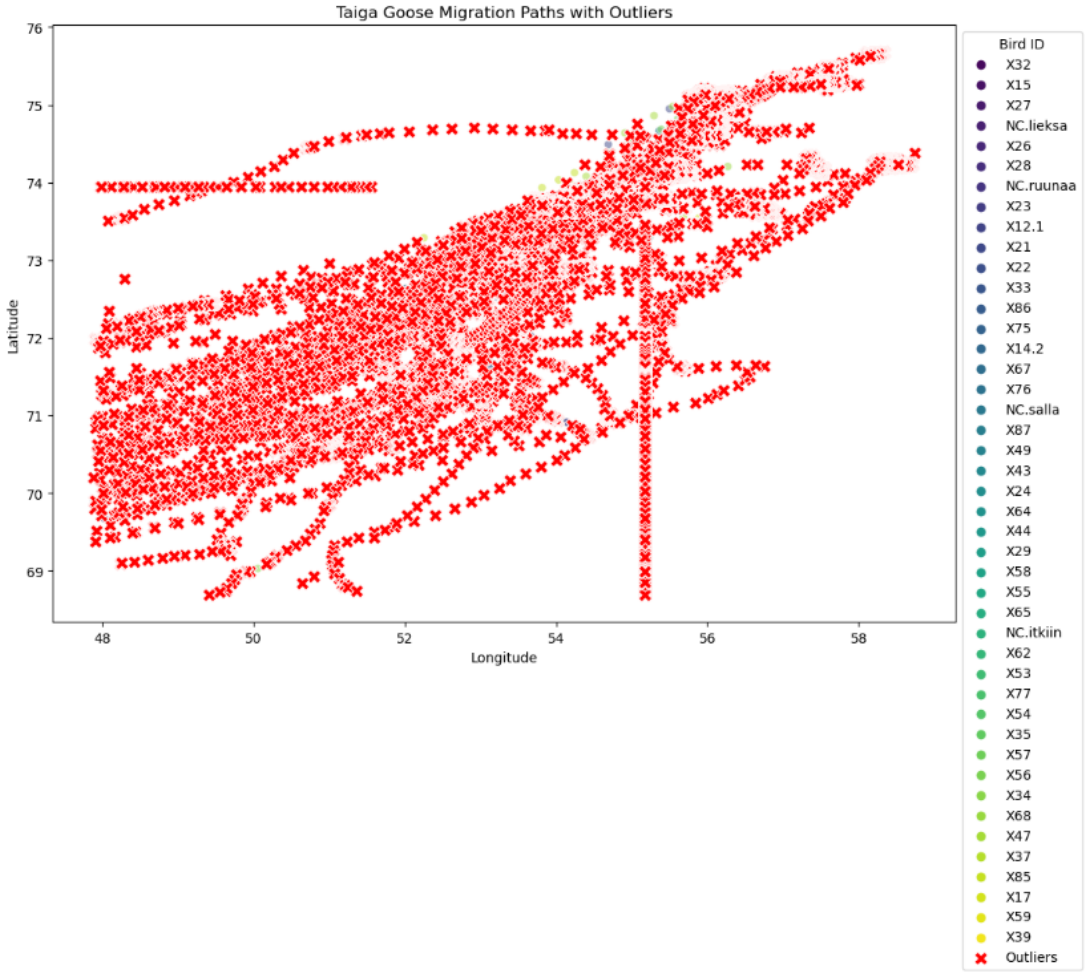
```
import geopy.distance

def haversine(lat1, lon1, lat2, lon2):
    coords_1 = (lat1, lon1)
    coords_2 = (lat2, lon2)
    return geopy.distance.geodesic(coords_1, coords_2).km

# Calculate the distance between each point and the next
distances = [haversine(df_cleaned['location-lat'][i], df_cleaned['location-long'][i], df_cleaned['location-lat'][i+1], df_cleaned['location-long'][i+1])

# Set a threshold for reasonable movement, for example, 50 km (you can adjust this threshold)
threshold = 50  # in km
outliers = [i for i, dist in enumerate(distances) if dist > threshold]

# Print only the first 10 outliers
print("Outlier indices:", outliers[:10])
```

```
Outlier indices: [4118, 4119, 4123, 4124, 4129, 4130, 4134, 4135, 4139, 4140]
```

```
# Check for missing data in the latitude and longitude columns
missing_lat = df_cleaned['location-lat'].isnull().sum()
missing_long = df_cleaned['location-long'].isnull().sum()

print(f"Missing latitude data: {missing_lat}")
print(f"Missing longitude data: {missing_long}")
```

```
Missing latitude data: 0
Missing longitude data: 0
```

```
import seaborn as sns
import matplotlib.pyplot as plt

# Filter the data to show the outliers
outlier_data = df_cleaned.iloc[outliers]

# Plot migration paths including outliers
plt.figure(figsize=(12, 8))
sns.scatterplot(
    data=df_cleaned,
    x="location-long",
    y="location-lat",
    hue="individual-local-identifier",
    alpha=0.5,
    palette="viridis"
)
sns.scatterplot(
    data=outlier_data,
    x="location-long",
    y="location-lat",
    color="red",
    label="Outliers",
    marker="X",
    s=100
)
plt.title("Taiga Goose Migration Paths with Outliers")
plt.xlabel("Longitude")
plt.ylabel("Latitude")
plt.legend(title="Bird ID", bbox_to_anchor=(1, 1))
plt.show()
```

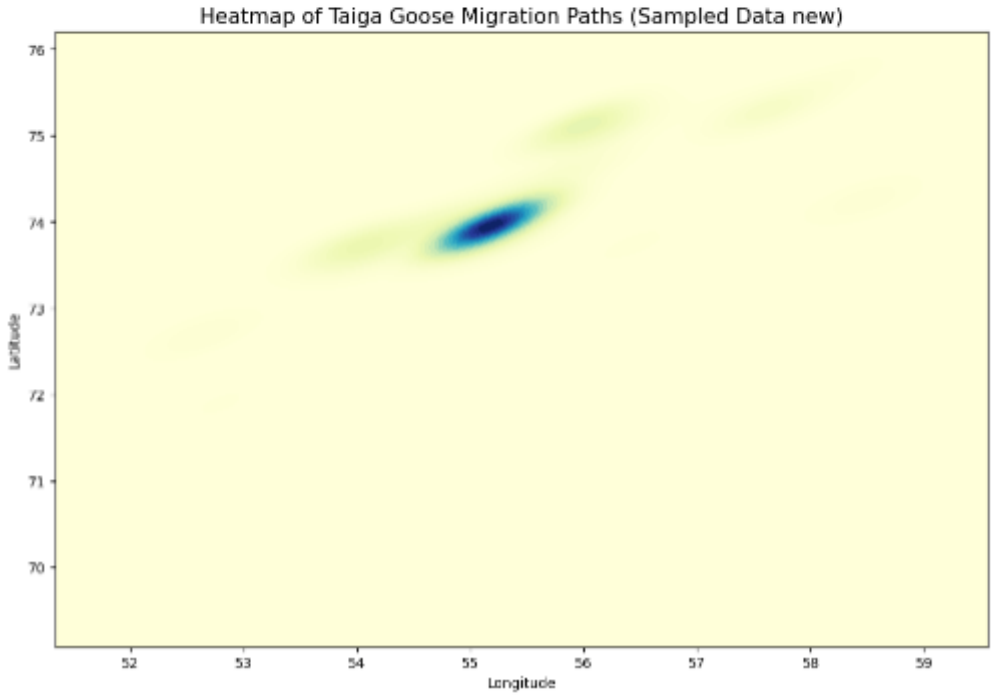



```
# Remove the rows corresponding to the outliers
df_clean = df_cleaned.drop(df_cleaned.index[outliers])

# Confirm the number of rows after cleaning
print(f"Shape of cleaned data: {df_clean.shape}")
```

```python
from sklearn.cluster import MiniBatchKMeans
import folium
from folium.plugins import MarkerCluster

clustering = df_clean[['u10', 'v10']]
kmeans = MiniBatchKMeans(n_clusters=3, random_state=42, batch_size=1000)
df_clean['cluster'] = kmeans.fit_predict(clustering_data)

df_sampled = df_clean.sample(n=1000, random_state=42)

map_center = [df_sampled['location-lat'].mean(), df_sampled['location-long'].mean()]
my_map = folium.Map(location=map_center, zoom_start=6)

marker_cluster = MarkerCluster().add_to(my_map)

for idx, row in df_sampled.iterrows():
    folium.CircleMarker(
        location=[row['location-lat'], row['location-long']],
        radius=5,
        color='blue' if row['cluster'] == 0 else ('red' if row['cluster'] == 1 else 'green'),
        fill=True,
        fill_opacity=0.6,
        popup=f"Cluster {row['cluster']}"
    ).add_to(marker_cluster)
("C:/Users/USER/Downloads/Merged_Taiga_Goose_ERA5_Cleaned (1).csv")
output_path = "C:/Users/USER/Downloads/Taiga_Goose_Clusters_Map_Optimize.html"
my_map.save(output_path)

print(f"Map has been saved to: {output_path}")
```

```
Map has been saved to: C:/Users/USER/Downloads/Taiga_Goose_Clusters_Map_Optimize.html
```

```python
import seaborn as sns
import matplotlib.pyplot as plt

# Sampling
df_sample = df_clean.sample(n=1000, random_state=42)  # You can adjust the number (1000) as per your system capacity

# Create a heatmap of the migration paths based on Longitude and Latitude
plt.figure(figsize=(12, 8))
sns.kdeplot(
    data=df_sample,
    x='location-long',
    y='location-lat',
    cmap="YlGnBu",
    fill=True,
    thresh=0,
    levels=30  # Lower number of levels for smoother heatmap
)

# Customize plot
plt.title('Heatmap of Taiga Goose Migration Paths (Sampled Data new)', fontsize=15)
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```



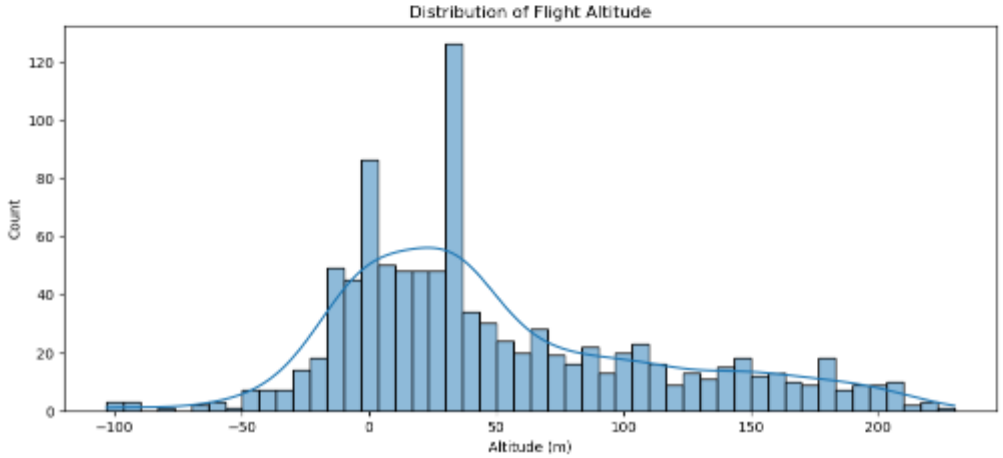
```
import folium
from folium.plugins import HeatMap

# Sampling a smaller portion of the data
df_sample = df_clean.sample(n=1000, random_state=42)  # Adjust the number as needed

# Create a map centered around the average latitude and longitude
map_center = [df_sample['location-lat'].mean(), df_sample['location-long'].mean()]
my_map = folium.Map(location=map_center, zoom_start=6)

# Prepare the data for heatmap (list of latitudes and longitudes)
heat_data = [[row['location-lat'], row['location-long']] for index, row in df_sample.iterrows()]

# Add the heatmap layer
HeatMap(heat_data, radius=10, blur=15).add_to(my_map)

# Save the map to an HTML file
output_path = "C:/Users/USER/Downloads/Taiga_Goose_Heatmap_Sample.html"
my_map.save(output_path)

print(f"Heatmap saved to: {output_path}")
```

```
Heatmap saved to: C:/Users/USER/Downloads/Taiga_Goose_Heatmap_Sample.html
```

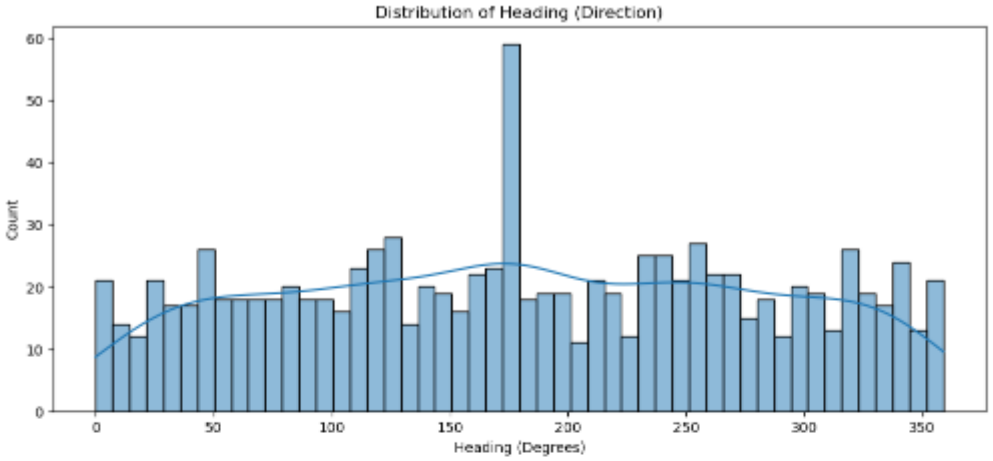
```
import folium
from folium.plugins import HeatMap, MarkerCluster

# Sampling a smaller portion of the data
df_sample = df_clean.sample(n=1000, random_state=42)  # You can adjust the number as needed

# Create a map centered around the average latitude and longitude
map_center = [df_sample['location-lat'].mean(), df_sample['location-long'].mean()]
my_map = folium.Map(location=map_center, zoom_start=6)

# Create MarkerCluster to group close locations
marker_cluster = MarkerCluster().add_to(my_map)

# Add markers for each bird's migration point, colored by the cluster
for idx, row in df_sampled.iterrows():
    folium.CircleMarker(
        location=[row['location-lat'], row['location-long']],
        radius=5,
        color='blue' if row['cluster'] == 0 else ('red' if row['cluster'] == 1 else 'green'),
        fill=True,
        fill_opacity=0.6,
        popup=f"Cluster {row['cluster']}, Bird ID: {row['individual-local-identifier']}, Speed: {row['u10']}, Direction: {row['v10']}"
    ).add_to(marker_cluster)

# Prepare the data for the heatmap (list of latitudes and longitudes)
heat_data = [[row['location-lat'], row['location-long']] for index, row in df_sample.iterrows()]

# Add the heatmap layer
HeatMap(heat_data, radius=10, blur=15).add_to(my_map)

# Save the map to an HTML file
output_path = "C:/Users/USER/Downloads/Taiga_Goose_Heatmap_Clusters.html"
my_map.save(output_path)

print(f"Heatmap with Clusters saved to: {output_path}")
```

```
Heatmap with Clusters saved to: C:/Users/USER/Downloads/Taiga_Goose_Heatmap_Clusters.html
```

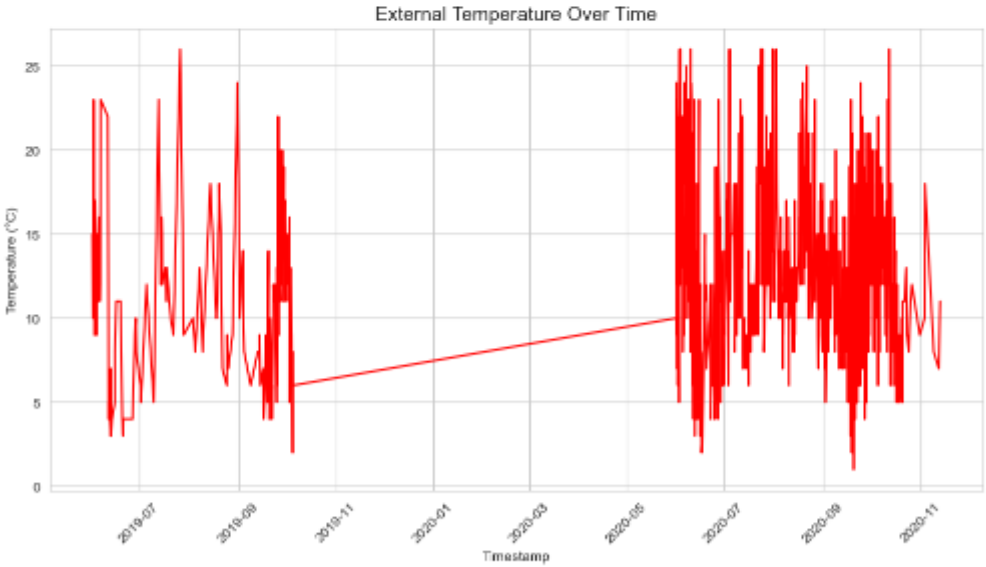
```
plt.figure(figsize=(12, 5))
sns.histplot(df_sample["height-above-ellipsoid"], bins=50, kde=True)
plt.title("Distribution of Flight Altitude")
plt.xlabel("Altitude (m)")
plt.show()
```



```
plt.figure(figsize=(12, 5))
sns.histplot(df_sample["heading"], bins=50, kde=True)
plt.title("Distribution of Heading (Direction)")
plt.xlabel("Heading (Degrees)")
plt.show()
```

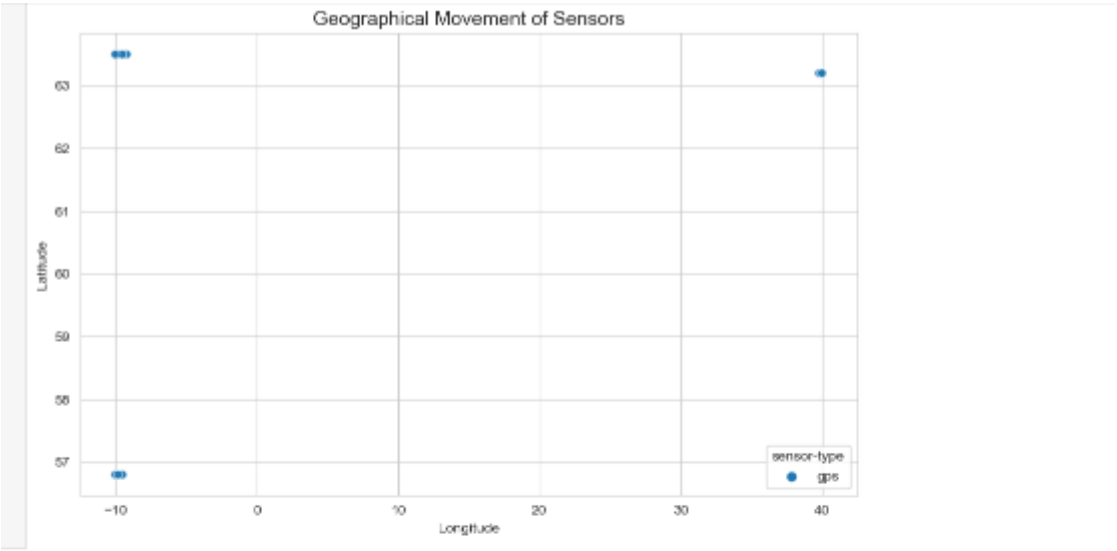



```
# Set a colorful style
sns.set_style("whitegrid")
plt.rcParams["axes.prop_cycle"] = plt.cycler(color=sns.color_palette("tab10"))

# Temperature Over Time
plt.figure(figsize=(12, 6))
sns.lineplot(data=df_sample, x="timestamp", y="external-temperature", color="red", linewidth=1.5)
plt.title("External Temperature Over Time", fontsize=14)
plt.xlabel("Timestamp")
plt.ylabel("Temperature (°C)")
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

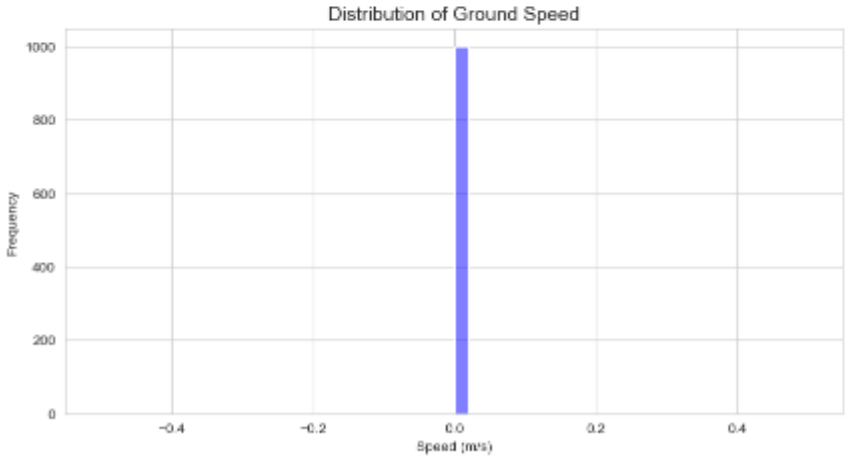



```
# Longitude vs. Latitude
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df_sample, x="longitude", y="latitude", hue="sensor-type", alpha=0.7)
plt.title("Geographical Movement of Sensors", fontsize=14)
plt.xlabel("Longitude")
plt.ylabel("Latitude")
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```

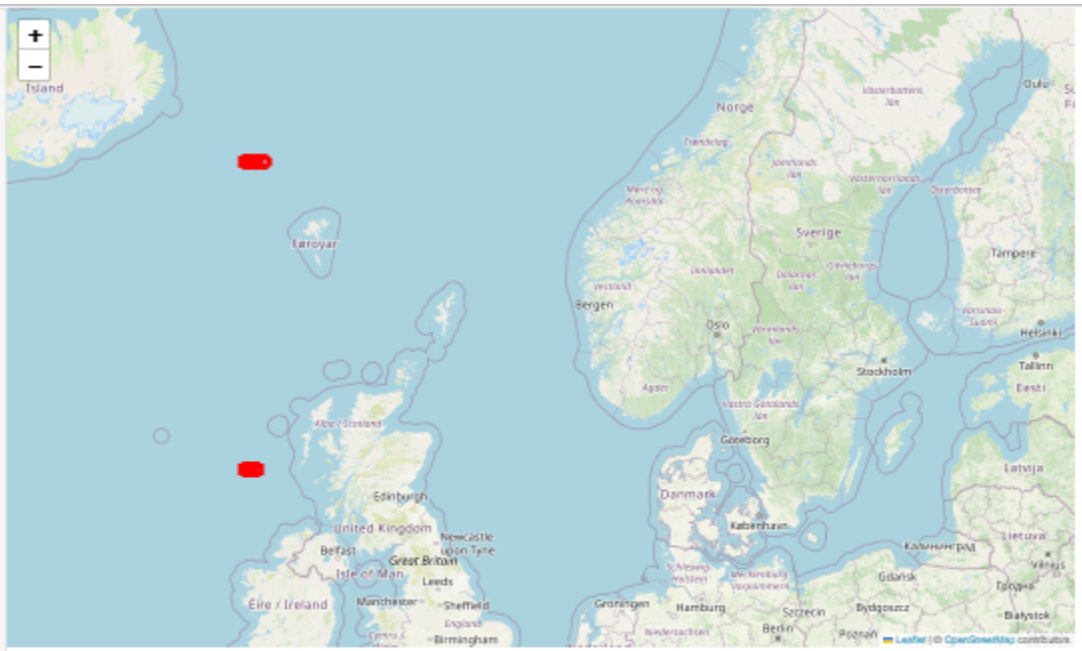



```
# Distribution of Ground Speed
plt.figure(figsize=(10, 5))
sns.histplot(df_sample["ground-speed"], bins=50, kde=True, color="blue")
plt.title("Distribution of Ground Speed", fontsize=14)
plt.xlabel("Speed (m/s)")
plt.ylabel("Frequency")
plt.show()
```




```
# Visualizing movement trajectory
mapvisual = folium.Map(location=[df_cleaned["latitude"].mean(), df_cleaned["longitude"].mean()], zoom_start=5)
for _, row in df_cleaned.sample(15000).iterrows():
    folium.CircleMarker([row["latitude"], row["longitude"]], radius=5, color="red").add_to(mapvisual)
mapvisual.save("goose_trajectory_map.html")
mapvisual
```

[23]:



[24]:
```
df_cleane = df_cleaned.drop(['event-id', 'visible', 'sensor-type', 'individual-taxon-canonical-name', 'individual-local-identifier', 'study-name', 'ere
df_clean= df_cleane.copy()
df_clean = df_clean.drop(['timestamp'], axis=1)
```

[25]:
```
df_clns = df_clean.apply(pd.to_numeric, errors='coerce')  # Convert where possible
correlation_numeric = df_clns.corr()

print(correlation_numeric)
```

```
                        location-long  location-lat  external-temperature  \
location-long                1.000000      0.630118              0.215024
location-lat                 0.630118      1.000000              0.232307
external-temperature         0.215024      0.232307              1.000000
gps:hdop                    -0.009236     -0.011289             -0.019944
gps:satellite-count         -0.000590      0.011147              0.040031
heading                      0.012773      0.022226             -0.001206
height-above-ellipsoid      -0.017175     -0.057783             -0.005487
tag-local-identifier         0.142812      0.214028              0.066321
latitude                    -0.000119     -0.000426             -0.047719
longitude                    0.008544      0.008306             -0.038911
tp                           0.001663      0.020474              0.019989
u10                          0.056646      0.061097              0.017062
v10                          0.011466      0.024533             -0.038815
t2m                          0.116250      0.101142              0.039169
sp                          -0.072071     -0.083687             -0.037264

                        gps:hdop  gps:satellite-count   heading  \
location-long          -0.009236            -0.000590  0.012773
location-lat           -0.011289             0.011147  0.022226
external-temperature   -0.019944             0.040031 -0.001206
gps:hdop                1.000000            -0.721981  0.010039
gps:satellite-count    -0.721981             1.000000 -0.004954
heading                 0.010039            -0.004954  1.000000
height-above-ellipsoid -0.001320            -0.001107  0.003827
tag-local-identifier   -0.076824             0.104822  0.000971
latitude               -0.027881             0.027456 -0.005383
longitude              -0.013943             0.014089 -0.004720
tp                      0.011487            -0.010557  0.000770
u10                     0.004548            -0.011185  0.007006
v10                     0.006409            -0.012104  0.005382
t2m                     0.012840            -0.017058  0.002190
sp                      0.012928            -0.008062 -0.000870

                        height-above-ellipsoid  tag-local-identifier  \
location-long                        -0.017175              0.142812
location-lat                         -0.057783              0.214028
external-temperature                 -0.005487              0.066321
gps:hdop                             -0.001320             -0.076824
gps:satellite-count                  -0.001107              0.104822
heading                               0.003827              0.000971
height-above-ellipsoid                1.000000              0.027322
tag-local-identifier                  0.027322              1.000000
latitude                             -0.018419             -0.001482
longitude                            -0.019311              0.001746
tp                                   -0.047704             -0.015219
u10                                  -0.079161              0.006651
v10                                  -0.019317             -0.007598
t2m                                  -0.015403             -0.002951
sp                                    0.062777              0.016633
```

```python
df = df_clean.copy()

#'timestamp' is present, and kept for ARIMA
drop_cols = ["sensor-type", "event-id", "visible", "individual-taxon-canonical-name",
             "individual-local-identifier", "study-name", "era5_nearest_time",
             "number_x", "eqvar_x", "number_y", "eqvar_y", "ground-speed"]
df.drop(columns=[c for c in drop_cols if c in df.columns], errors="ignore", inplace=True)

print("Columns in final df:", df.columns)
```

```
Columns in final df: Index(['location-long', 'location-lat', 'external-temperature', 'gps:hdop',
       'gps:satellite-count', 'heading', 'height-above-ellipsoid',
       'tag-local-identifier', 'latitude', 'longitude', 'tp', 'u10', 'v10',
       't2m', 'sp'],
      dtype='object')
```

```python
# For multi-output models, we define:
feature_cols = [c for c in df.columns if c not in ["latitude", "longitude", "timestamp"]]
X = df[feature_cols].dropna()  # numeric features
y = df[["latitude", "longitude"]].dropna()
```

```python
# Align X and Y
X = X.loc[y.index]
y= y.loc[X.index]
```

```python
print("X shape:", X.shape)
print("Y shape:", y.shape)
```

```
X shape: (848212, 13)
Y shape: (848212, 2)
```

```python
from sklearn.model_selection import train_test_split

# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

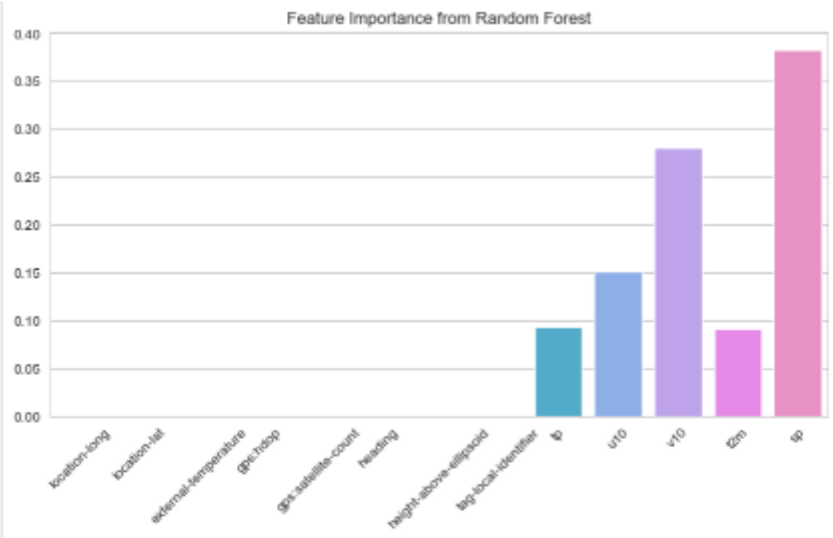
```python
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.metrics import mean_squared_error, accuracy_score

# Random Forest
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
```

```
▾        RandomForestRegressor

RandomForestRegressor(random_state=42)
```

```python
y_pred_rf = rf.predict(X_test)
mse_rf = mean_squared_error(y_test, y_pred_rf)
rmse_rf = np.sqrt(mse_rf)
print(f"Random Forest MSE:  {mse_rf:.4f}")
print(f"Random Forest RMSE: {rmse_rf:.4f}")
```

```
Random Forest MSE:  0.0075
Random Forest RMSE: 0.0868
```

```python
# Get feature importance
feature_importance = rf.feature_importances_

# Plot feature importance
plt.figure(figsize=(10, 5))
sns.barplot(x=X.columns, y=feature_importance)
plt.xticks(rotation=45)
plt.title("Feature Importance from Random Forest")
plt.show()
```




```
# Gradient Boosting
# Since GradientBoostingRegressor is not natively multi-output, MultiOutputRegressor is used to handle 2D target
from sklearn.multioutput import MultiOutputRegressor

gbr = GradientBoostingRegressor(n_estimators=100, random_state=42)
gbr_multi = MultiOutputRegressor(gbr)
gbr_multi.fit(X_train, y_train)
```

```
MultiOutputRegressor
estimator: GradientBoostingRegressor
    GradientBoostingRegressor
```

```
y_pred_gbr = gbr_multi.predict(X_test)
mse_gbr = mean_squared_error(y_test, y_pred_gbr)
rmse_gbr = np.sqrt(mse_gbr)
print(f"Gradient Boosting MSE:  {mse_gbr:.4f}")
print(f"Gradient Boosting RMSE: {rmse_gbr:.4f}")
```

```
Gradient Boosting MSE:  23.8876
Gradient Boosting RMSE: 4.8793
```

```
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.optimizers import Adam

def create_sequences(X, y, seq_length=7):
    X_df = pd.DataFrame(X, index=Y.index) if not isinstance(X, pd.DataFrame) else X
    Y_df = pd.DataFrame(y) if not isinstance(y, pd.DataFrame) else y
    Xs, Ys = [], []
    for i in range(len(X_df) - seq_length):
        X_seq = X_df.iloc[i : i + seq_length].values
        Y_seq = Y_df.iloc[i + seq_length].values
        Xs.append(X_seq)
        Ys.append(Y_seq)
    return np.array(Xs), np.array(Ys)

# Recreate X, Y for LSTM
X_lstm = X.copy()
Y_lstm = y.copy()
```

```
seq_length = 7
X_seq, Y_seq = create_sequences(X_lstm, Y_lstm, seq_length=seq_length)
print("LSTM X_seq shape:", X_seq.shape)
print("LSTM Y_seq shape:", Y_seq.shape)
X_train_lstm, X_test_lstm, y_train_lstm, y_test_lstm = train_test_split(
    X_seq, Y_seq, test_size=0.2, random_state=42
)
```

```
LSTM X_seq shape: (848205, 7, 13)
LSTM Y_seq shape: (848205, 2)
```

```
# LSTM
model = Sequential()
model.add(LSTM(64, input_shape=(seq_length, X.shape[1]), return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(32))
model.add(Dropout(0.2))
model.add(Dense(2))  # Latitude, Longitude prediction
```

```
model.compile(optimizer=Adam(learning_rate=0.001), loss="mse")
model.summary()
history = model.fit(
    X_train_lstm, y_train_lstm,
    epochs=10,
    batch_size=32,
    validation_split=0.1,
    shuffle=False
)
```

```
Model: "sequential"
```

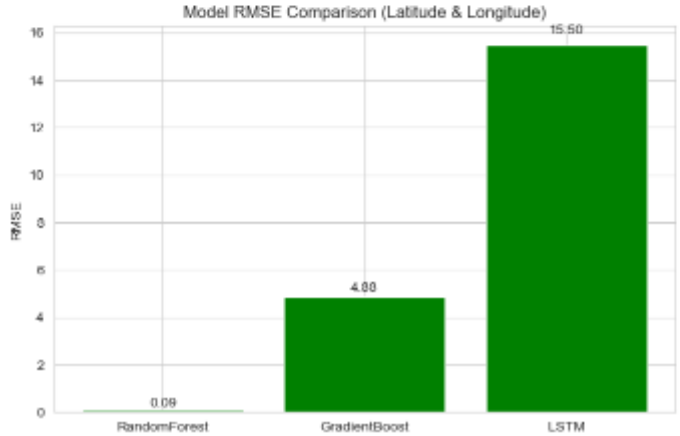
Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 7, 64)	19,968
dropout (Dropout)	(None, 7, 64)	0
lstm_1 (LSTM)	(None, 32)	12,416
dropout_1 (Dropout)	(None, 32)	0
dense (Dense)	(None, 2)	66

```
Total params: 32,450 (126.76 KB)
Trainable params: 32,450 (126.76 KB)
Non-trainable params: 0 (0.00 B)
Epoch 1/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 149s 8ms/step - loss: 441.9167 - val_loss: 239.1776
Epoch 2/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 114s 6ms/step - loss: 250.9487 - val_loss: 239.2487
Epoch 3/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 115s 6ms/step - loss: 248.4114 - val_loss: 239.1669
Epoch 4/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 127s 7ms/step - loss: 246.2727 - val_loss: 239.1381
Epoch 5/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 145s 8ms/step - loss: 244.5822 - val_loss: 239.1526
Epoch 6/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 133s 7ms/step - loss: 243.2541 - val_loss: 239.1252
Epoch 7/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 139s 7ms/step - loss: 242.3134 - val_loss: 239.1245
Epoch 8/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 154s 8ms/step - loss: 241.6145 - val_loss: 239.1243
Epoch 9/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 120s 6ms/step - loss: 241.1350 - val_loss: 239.1223
Epoch 10/10
18905/18905 ━━━━━━━━━━━━━━━━━━━━ 128s 7ms/step - loss: 240.8264 - val_loss: 239.1161
```

```
# Predict and Evaluate
y_pred_lstm = model.predict(X_test_lstm)
mse_lstm = mean_squared_error(y_test_lstm, y_pred_lstm)
rmse_lstm = np.sqrt(mse_lstm)
print(f"LSTM MSE:  {mse_lstm:.4f}")
print(f"LSTM RMSE: {rmse_lstm:.4f}")
```

```
5252/5252 ━━━━━━━━━━━━━━━━━━━━ 14s 3ms/step
LSTM MSE:  240.2746
LSTM RMSE: 15.5008
```

```
model_names = ["RandomForest", "GradientBoost", "LSTM"]
model_rmse = [rmse_rf, rmse_gbr, rmse_lstm]  # from the steps above

plt.figure(figsize=(8,5))
bars = plt.bar(model_names, model_rmse, color='green')
plt.title("Model RMSE Comparison (Latitude & Longitude)")
plt.ylabel("RMSE")
for bar in bars:
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2., 1.02*height,
             f"{height:.2f}", ha='center', va='bottom')
plt.show()
```



APPENDIX 2: STREAMIT APPLICATION CODE

```
import streamlit as st
import pandas as pd
import folium
from folium.plugins import HeatMap, MarkerCluster, TimestampedGeoJson, MiniMap, MousePosition
from streamlit_folium import st_folium
import joblib
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import streamlit.components.v1 as components
from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np

# --- Constants ---
DATA_PATH = "/Users/admin/Downloads/new_data.csv"  # Use a copy to prevent overwriting
MODEL_PATH = "rf_model.pkl"
FEATURES_PATH = "rf_features.pkl"

# --- Load Data & Model ---
df = pd.read_csv(DATA_PATH).copy()  # Always copy after loading
model = joblib.load(MODEL_PATH)
feature_names = joblib.load(FEATURES_PATH)
df['timestamp'] = pd.to_datetime(df['timestamp'])
›
# --- Debug info ---
st.sidebar.title("Debug Info")
st.sidebar.write("Unique Location-Lat:", df['location-lat'].nunique())
st.sidebar.write("Unique Location-Long:", df['location-long'].nunique())

# --- Sidebar Navigation ---
st.sidebar.title("Navigation")
section = st.sidebar.radio("Go to", [
    "EDA Snapshots",
    "Interactive Map",
    "Model Prediction",
    "Model Per Bird",
    "Compare Prediction vs Actual"
])



# --- Section 1: EDA Snapshots ---
if section == "EDA Snapshots":
    st.title("Exploratory Data Analysis Snapshots")
    st.image("/Users/admin/Downloads/eda_migration_paths.png", caption="Migration Paths")
    st.image("/Users/admin/Downloads/eda_migration_outliers.png", caption="Outliers Highlighted")
    st.image("/Users/admin/Downloads/eda_correlation_heatmap.png", caption="Correlation Heatmap")
    st.image("/Users/admin/Downloads/eda_altitude_dist.png", caption="Altitude Distribution")
    st.image("/Users/admin/Downloads/eda_heading_dist.png", caption="Heading Distribution")
    st.image("/Users/admin/Downloads/eda_temperature.png", caption="Temperature Over Time")
    st.image("/Users/admin/Downloads/eda_cluster_speed.png", caption="Speed per Cluster")
    st.image("/Users/admin/Downloads/eda_heatmap.png", caption="Migration Heatmap")
    st.image("/Users/admin/Downloads/eda_cluster_speed_over_time.png", caption="Speed Over Time")
    st.image("/Users/admin/Downloads/eda_cluster_direction_over_time.png", caption="Direction Over Time")
    st.image("/Users/admin/Downloads/eda_sensor_geolocation.png", caption="Sensor Movement")

# --- Section 2: Interactive Map ---
elif section == "Interactive Map":
    st.header("Explore Goose Movement Patterns")
    df_sample = df.sample(n=1000, random_state=42)

    if 'cluster' not in df_sample.columns:
        df_sample = df_sample.dropna(subset=['u10', 'v10'])
        kmeans = KMeans(n_clusters=3, random_state=42)
        df_sample['cluster'] = kmeans.fit_predict(df_sample[['u10', 'v10']])

    map_center = [df_sample['location-lat'].mean(), df_sample['location-long'].mean()]
    my_map = folium.Map(location=map_center, zoom_start=5, control_scale=True)
    marker_cluster = MarkerCluster(name="Migration Points").add_to(my_map)

    for _, row in df_sample.iterrows():
        color = ['blue', 'red', 'green'][int(row['cluster']) % 3]
        popup = f"""
        <b>Bird ID:</b> {row['individual-local-identifier']}<br>
        <b>Speed:</b> {row['u10']:.2f} m/s<br>
        <b>Direction:</b> {row['v10']:.2f}&deg;<br>
        <b>Timestamp:</b> {row['timestamp']}
        """
        folium.CircleMarker(
            location=[row['location-lat'], row['location-long']],
```

```
            location=[row['location-lat'], row['location-long']],
            radius=5,
            color=color,
            fill=True,
            fill_opacity=0.6,
            popup=popup
        ).add_to(marker_cluster)

    heat_data = [[row['location-lat'], row['location-long']] for _, row in df_sample.iterrows()]
    HeatMap(heat_data, radius=10, blur=15).add_to(my_map)

    features = [
        {
            'type': 'Feature',
            'geometry': {'type': 'Point', 'coordinates': [row['location-long'], row['location-lat']]},
            'properties': {
                'time': row['timestamp'].isoformat(),
                'popup': f"{row['individual-local-identifier']} - {row['u10']:.2f} m/s",
                'icon': 'circle',
                'iconstyle': {'fillColor': 'orange', 'radius': 4}
            }
        } for _, row in df_sample.iterrows()
    ]

    TimestampedGeoJson({"type": "FeatureCollection", "features": features},
                       period='PT1H', add_last_point=True, loop=False,
                       auto_play=False, time_slider_drag_update=True).add_to(my_map)

    MiniMap(toggle_display=True).add_to(my_map)
    MousePosition(position='bottomright').add_to(my_map)
    folium.LayerControl(collapsed=False).add_to(my_map)
    components.html(my_map._repr_html_(), height=800)

# --- Section 3: Model Prediction ---
elif section == "Model Prediction":
    st.title("Latitude & Longitude Prediction")

    input_data = {}
    for feature in feature_names:
        input_data[feature] = st.number_input(f"{feature}", value=0.0, key=f"input_{feature}")

    if st.button("Predict Location"):
        input_df = pd.DataFrame([input_data])
        pred = model.predict(input_df)[0]
        st.success(f"Predicted Latitude: {pred[0]:.4f}, Longitude: {pred[1]:.4f}")

        result_df = input_df.copy()
        result_df['pred_latitude'] = pred[0]
        result_df['pred_longitude'] = pred[1]
        csv = result_df.to_csv(index=False).encode()
        st.download_button("Download Prediction", csv, "prediction.csv", "text/csv")

# --- Section 4: Model Per Bird ---
elif section == "Model Per Bird":
    st.title("Prediction per Bird")
    bird_ids = df['individual-local-identifier'].unique()
    selected_bird = st.selectbox("Select a Bird", bird_ids)
    bird_df = df[df['individual-local-identifier'] == selected_bird]

    st.write(f"Showing {len(bird_df)} records for bird: {selected_bird}")

    if st.button("Predict All for This Bird"):
        bird_input = bird_df[feature_names].dropna()
        bird_preds = model.predict(bird_input)
        bird_df_result = bird_df.copy()
        bird_df_result['pred_latitude'] = bird_preds[:, 0]
        bird_df_result['pred_longitude'] = bird_preds[:, 1]
        st.map(bird_df_result[['pred_latitude', 'pred_longitude']].rename(columns={
            'pred_latitude': 'latitude', 'pred_longitude': 'longitude'
        }))

        csv = bird_df_result.to_csv(index=False).encode()
        st.download_button("Download Bird Predictions", csv, "bird_predictions.csv", "text/csv")

# --- Section 5: Comparing Prediction vs Actual ---
elif section == "Compare Prediction vs Actual":
    st.title("\U0001F4CA Compare Model Predictions vs Actual (with True Evaluation)")

    if 'location-lat' in df.columns and 'location-long' in df.columns:
        actual = df[['timestamp', 'location-lat', 'location-long']].copy()
        actual = actual.dropna()
```

```
        actual = df[['timestamp', 'location-lat', 'location-long']].copy()
        actual = actual.dropna()

        # Convert to date for stratified sampling
        actual['date'] = actual['timestamp'].dt.date

        # --- Date filtering UI ---
        st.subheader("Filter by Date Range")
        min_date = df['timestamp'].min().date()
        max_date = df['timestamp'].max().date()

        start_date, end_date = st.slider(
            "Select Date Range",
            min_value=min_date,
            max_value=max_date,
            value=(min_date, max_date),
            format="YYYY-MM-DD"
        )

        # --- Apply date filter ---
        filtered_actual = actual[
            (actual['timestamp'].dt.date >= start_date) &
            (actual['timestamp'].dt.date <= end_date)
        ]

        # --- Random sampling from filtered results ---
        sample_size = st.slider("Number of Samples to Compare", 10, min(100, len(filtered_actual)), 50)
        sampled_actual = filtered_actual.sample(n=sample_size, random_state=42)

        # Align with input features
        model_input = df[feature_names].loc[sampled_actual.index].dropna()
        sampled_actual = sampled_actual.loc[model_input.index]

        predictions = model.predict(model_input)

        sampled_actual['pred_latitude'] = predictions[:, 0]
        sampled_actual['pred_longitude'] = predictions[:, 1]
        sampled_actual['lat_error'] = sampled_actual['location-lat'] - sampled_actual['pred_latitude']
        sampled_actual['lon_error'] = sampled_actual['location-long'] - sampled_actual['pred_longitude']

        rmse_lat = np.sqrt(mean_squared_error(sampled_actual['location-lat'], sampled_actual['pred_latitude
        rmse_lon = np.sqrt(mean_squared_error(sampled_actual['location-long'], sampled_actual['pred_longitu
        mae_lat = mean_absolute_error(sampled_actual['location-lat'], sampled_actual['pred_latitude'])
        mae_lon = mean_absolute_error(sampled_actual['location-long'], sampled_actual['pred_longitude'])

        st.metric("Latitude RMSE", f"{rmse_lat:.4f}")
        st.metric("Longitude RMSE", f"{rmse_lon:.4f}")
        st.metric("Latitude MAE", f"{mae_lat:.4f}")
        st.metric("Longitude MAE", f"{mae_lon:.4f}")

        st.write("### Diverse Daily Sample - Prediction vs Actual Table")
        st.dataframe(sampled_actual[['timestamp', 'location-lat', 'pred_latitude', 'lat_error',
                                     'location-long', 'pred_longitude', 'lon_error']].head(20))

        st.write("### Latitude Error Distribution")
        plt.figure(figsize=(8, 4))
        plt.hist(sampled_actual['lat_error'], bins=30, color='skyblue')
        plt.xlabel("Latitude Error")
        plt.ylabel("Frequency")
        st.pyplot(plt.gcf())

        st.write("### Longitude Error Distribution")
        plt.figure(figsize=(8, 4))
        plt.hist(sampled_actual['lon_error'], bins=30, color='salmon')
        plt.xlabel("Longitude Error")
        plt.ylabel("Frequency")
        st.pyplot(plt.gcf())

        st.write("### Scatter Plot: Actual vs Predicted Latitude")
        plt.figure(figsize=(8, 6))
        plt.scatter(sampled_actual['location-lat'], sampled_actual['pred_latitude'], alpha=0.5, c='purple')
        plt.plot([sampled_actual['location-lat'].min(), sampled_actual['location-lat'].max()],
                 [sampled_actual['location-lat'].min(), sampled_actual['location-lat'].max()], 'k--', lw=2)
        plt.xlabel("Actual Latitude")
        plt.ylabel("Predicted Latitude")
        plt.title("Actual vs Predicted Latitude (Daily Sampled)")
        st.pyplot(plt.gcf())

    else:
        st.warning("location-lat and location-long columns not found in dataset.")
```