Swarm Behaviour Analysis
Statistical Methods for High Dimensional Data
Final Project

Horatiu Palaghiu[†], Leonardo Schiavo[‡]

***Abstract*—Recent studies on various animal species have highlighted the development of swarm intelligence, emphasizing the distinction between random and organized actions. The focus of this study is to identify the characteristics of organized swarm behavior.**

***Index Terms*—Swarm Behaviour, Lasso, Generalized Lasso, Feature Extraction, Group Lasso.**

I. Introduction

Swarm behavior refers to the collective and coordinated actions exhibited by groups of individuals, typically animals or even artificial agents, that interact with each other to achieve common goals. This phenomenon is characterized by decentralized, self-organized systems where individuals within the group respond to local interactions with their neighbors rather than following central coordination. Swarm behavior is observed in various natural contexts, such as flocks of birds, schools of fish, and colonies of insects, as well as in artificial systems like drone swarms and robotic networks.

The key principles underlying swarm behavior include emergence, where complex group behavior arises from simple interactions at the individual level, and robustness, as the collective can adapt to changing environments or disturbances. Studying swarm behavior provides insights into the mechanisms of decentralized coordination, communication, and decision-making that contribute to the success of the group as a whole. Researchers draw inspiration from natural swarms to develop algorithms and technologies that emulate these principles, contributing to fields such as robotics, optimization, and distributed computing. Understanding swarm behavior not only enhances our comprehension of the natural world but also offers innovative solutions for addressing complex problems in various domains.

In the scope of this research, we delved deeper into the study of swarm behavior, focusing on the utilization of feature extraction algorithms. The primary objective was to discern which distinctive traits within the data generated by simulated swarms are particularly significant for understanding their collective behavior.

Feature extraction algorithms proved to be crucial tools in our approach, enabling us to identify and isolate salient features that influence swarm dynamics. These algorithms facilitated the reduction of data complexity, honing in on key aspects such as position, velocity, direction, and other relevant metrics. This process of feature extraction is fundamental for obtaining a more concise and interpretable representation of the data, making the analysis of collective behavior within the swarm more accessible.

Through this methodology, we were able to pinpoint recurring patterns and relationships among the extracted features, providing a solid foundation for interpreting the behavior of simulated swarms. The use of feature extraction algorithms emerged as a crucial tool in unveiling underlying dynamics, contributing to a deeper understanding of interactions and strategies adopted by individuals within a group. This approach not only advances our theoretical comprehension of swarms but also offers new perspectives for the development of predictive models and practical applications based on these principles.

II. Dataset

The "Swarm Behaviour" dataset [1], [2] is a synthetic dataset. In this experiment, flocks of boids (bird-android objects) are simulated by the Reynolds' Model. This model, introduced in 1987, is based on three fundamental principles that guide the collective behavior of individual boids within a group.

- **Separation**: Boids avoid crowding together and maintain a minimum separation from their neighbors, creating space and preventing collisions.
- **Alignment**: Boids align their velocities with the average velocity of their nearby counterparts, fostering cohesion and synchronicity within the group.
- **Cohesion**: Boids move towards the average position of their neighbors, promoting a sense of cohesion and group unity.

By incorporating these simple yet powerful rules, the Reynolds' Model captures emergent complex behaviors observed in natural flocks, such as flocking, alignment, and grouping. The synthetic flocks generated through this model serve as the basis for the experimental videos in the "Swarm Behaviour" dataset, providing a simulated but realistic representation of collective animal behavior.

A short videos of these synthetic flocks are presented to individuals who then assign a score from 0 to 100 based on how much the boids in the video appear to exhibit flocking, alignment, or grouping behavior. The experiment data is

[†]Department of Mathematics, University of Padova, email: horatiu.palaghiu@studenti.unipd.it

[‡]Department of Mathematics, University of Padua, email: leonardo.schiavo.1@studenti.unipd.it

consolidated into a single dataset using a $50/100$ threshold, proposing a binary classification between flocks exhibiting flocking, alignment, and grouping behaviors, or none of the three, discarding mixed data.

Each row in the dataset corresponds to an image from a short video and is characterized by 2400 predictive features and the binary target variable $\{0, 1\}$ for classification. Specifically, each image contains 200 boids, and for each boid, a vector of 12 features is recorded $(12 * 200 = 2400)$. As the simulation is in a two-dimensional plane, each vectorial feature has two components. Specifically, the position, velocity, and alignment, cohesion, and separation vectors each have two components. Additionally, for each boid, two more features, n_S and n_{AC}, are calculated, representing the number of boids in the neighbourhood of the separation vector or the alignment/cohesion vector, respectively.

We will now go into the details of the meaning and specifications of each feature.

A. Position

The position of the 200 boids is encoded as a two-coordinate vector (x, y), where $x \in (-1500, 1500)$ and $y \in (-1000, 1000)$.

B. Velocity

The instantaneous velocity of each boid is encoded as (V_x, V_y), capturing only the current velocity. Since the temporal trajectory and specific movement law of each boid are unknown, and the simulation software details are unavailable, reconstructing the entire trajectory of each boid was not feasible.

C. Alignment, Cohesion, and Separation Vectors

The three vectors alignment, (A_x, A_y), cohesion (C_x, C_y), and separation (S_x, S_y), result from a specific process. For each boid, a neighborhood is computed based on a chosen topology (in the reference dataset, Euclidean topology is employed, but [2] explores alternatives like Voronoi tessellation or neighborhoods considering the potential obscuration of visibility by nearby boids). Subsequently, neighboring boids are identified, and the vectors are calculated as follows:

$$A = \frac{1}{|N|} \sum_{\text{neighbor } j \in N} \mathbf{v}_j$$

$$C = \frac{1}{|N|} \sum_{\text{neighbor } j \in N} (\text{position}_j - \text{position})$$

$$S = \sum_{\text{neighbor } j \in N} \frac{\text{position} - \text{position}_j}{d}$$

Here, N represents the set of neighbors, and d denotes the distance between the boid and its neighboring counterparts. These vectors capture essential aspects of the boids' interactions, encompassing alignment with neighbors, cohesion towards the group, and separation to avoid collisions. These vectors are then typically scaled by certain weights and added together to get the final velocity vector for the boid. Adjusting the weights allows you to control the strength of each behavior (alignment, cohesion, and separation) in influencing the boid's motion.

D. n_S and n_{AC}

The integer values n_S and n_{AC} correspond to the number of boids in the neighbourhood of the fixed boid, specifically within the neighborhoods created by the separation and alignment/cohesion vectors, respectively.

III. Algorithms for feature selection and results

As with any dataset with a very large number of features, we believe that we might have many redundant variables that actually reduce the generalization capability of our models, and might also reduce the overall accuracy of our classifier. Thus, feature selection methods are applied to our dataset, in order not only to generate algorithms that better describe the relationship between our data and the response variable, but also reduce the complexity of said algorithms.

Variables are selected by applying a penalization term to our coefficients that multiply each predictor, which results in shrinking some of the less important coefficients.

A. Baseline models: Linear Regression and Logistic Regression

In order to evaluate any improvements made by our embedded feature selection methods, we first try some baseline models on the full dataset:

Linear Regression: The most common regression algorithm, assumes a linear relationship between the response variable and the data, optimizing on the loss function:

$$\text{Loss}_{\text{Linear}} = \sum_{i=1}^{N} (y_i - (\beta_0 + \beta_1 x_i))^2$$

Logistic Regression: Used for binary classification problems, it models probability of something belonging to a certain class using the sigmoid function. The loss function is:

$$\text{Loss}_{\text{Logistic}} = -\frac{1}{N} \sum_{i=1}^{N} \left(y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \right)$$

B. Ridge

Ridge regression prevents overfitting by adding an L_2 penalty to the coefficients, shrinking some of them towards zero as a result. We are looking to optimize on the function:

$$\min_{\beta_0, \beta} \left\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - \beta_0 - x_i^T \beta)^2 + \lambda \|\beta\|_2^2 \right\}$$

C. Lasso

Lasso imposes an L_1 penalty on the coefficients, promoting sparsity by encouraging some coefficients to become exactly zero. It minimizes the formula:

$$\min_{\beta_0,\beta} \left\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - \beta_0 - x_i^T \beta)^2 + \lambda \|\beta\|_1 \right\}$$

D. Elastic Net

This regularization technique combines both the L_1 and L_2 penalties of Lasso and Ridge regression, respectively. This enables us to handle both feature selection and multicolinearity. Therefore, we need to optimize on the function:

$$\text{Loss} = \frac{1}{2N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \lambda \left(\frac{1}{2}(1-\alpha)||\beta||_2^2 + \alpha||\beta||_1 \right)$$

This time, optimization is done on both α and λ hyperparameters. We opted for a classic grid search with α variables between 0 and 1, and λ between 0 and 100.

E. Logistic Group Lasso

Group lasso, as the name itself suggests, is an extension of classic L_1 regularization, where sparsity is imposed not just on individual features, but on predefined groups of features. We are looking at minimizing the loss function:

$$-\sum_{i=1}^{N} \left(y_i \log(p_i) + (1-y_i) \log(1-p_i) \right) + \lambda \sum_{g=1}^{G} \sqrt{p_g} \, \|\beta_g\|_2$$

where G is the number of groups, and $p_g, g \in \{1, G\}$ represents the sizes of each of these groups.

In the case of our data, we noticed that the features naturally form groups in two distinct ways: by boid and boid measure but at the end both gave the same result.

F. Generalized Lasso

The most versatile extension of classic L_1 regularization techniques that we know of is Generalized Lasso - it encompasses a flexible framework, where the penalty term encourages sparsity in the model coefficients, which makes it perfect for our high-dimensional data. The structure of the regularization is imposed through a penalty matrix D:

$$\text{Loss} = \frac{1}{2N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} ||D \cdot \beta_j||_q$$

However, when using the *genlasso* package in R, we noticed that, unlike the *glmnet* package, there is no intercept term added by default to the X matrix, so we had to do it manually. Moreover, we added a column of zeroes to the penalty matrix, so that the intercept term is not penalized by genlasso.

IV. Results

In order to establish a baseline we performed two simple regression: linear and logistic. Now we can compare the baseline with the models we tried and just introduced in the previous section. The results are in Table 1.

Model	Accuracy	Precision	Recall
Linear	90.45 %	88.76 %	92.63 %
Logistic	90.74 %	89.08 %	92.88 %
LC	91.00 %	92.55 %	89.76 %
RC	88.53 %	89.68 %	87.68 %
EN	91.11 %	93.00 %	89.62 %
GL	91.50 %	88.21 %	95.82 %
sSVM	86.06 %	88.32 %	84.51 %
GenL	70.50 %	95.06 %	64.17 %

TABLE 1: Models Results: in order they are Linear Regression, Logistic Regression, Lasso Classifier, Ridge Classifier, Elastic Net Classifier, Logistic Group Lasso Classifier (grouped by boid and by same measure), Sparse SVM Classifier, Generalized Lasso Classifier.

V. Concluding Remarks

We have observed that utilizing the Lasso classifier, i.e., a linear model with L1 regularization, along with the Elastic Net classifier and Logistic Group Lasso, yielded modest yet improved accuracy compared to the baseline. By employing L2 regularization on both individual variables and groups, we have discerned that the features most crucial for predicting Grouped, Flocking, and Aligned behaviors against none of the three are velocity, cohesion vector, and the number of boids within the separation radius. This underscores the significance of regularization techniques in identifying key predictors in our predictive modeling framework.

References

[1] K. K. Shadi Abpeikar, "Motion behaviour recognition dataset collected from human perception of collective motion behaviour," 2023.

[2] M. B. Md Mohiuddin Khan, Kathryn Kasmarik, "Autonomous detection of collective behaviours in swarms," 2020.