MoNeT PYF Machine Learning

Ana Lucía Dueñas Chávez and Juan José Olivera Loyola *

May 4, 2021

Abstract

In this report we explain the project objectives and steps performed to implement a lightweight ML tool that approximates the results of a Mosquito population simulator.

1 Introduction

Mosquitos often carry diseases such as Dengue, Malaria, etc. That's why effort and economic investment has been placed in researching management of species' populations via introduction of gentically modified organisms who can surpress disease propagation. However, such actions require extensive knowledge and obtaining it through real test and error experiments is both infeasable and dangerous.

MoNeT (Mosquito Networks Taskforce) was created to develop computer tools that are capable of simulating mosquito population dynamics, movement and genetics aiding biologists to solve for the right variables. One of it's key reaseach lines is eliminating mosquitos in a population network of mosquitos in the French Polynesia Island. MoNeT's simulator is capable of producing population dynamics outcomes. However, it's computationally expensive, and during interdisciplinary meetings, it's common to speculate on many possible configuration out of the air. A more lightweight tool is needed to give immediate likely estimations for hypothesized configurations.

Machine Learning techniques have proven to be a general powerful tool for estimating functions and statistical distrbution spaces given large amounts of sample data and training time. Besides being universal approximators, ML models take short computational time for making arbitrary estimations and take very few memory space in comparison to the whole dataset they are approximating. Thus, ML techniques are a great technology to build a quick estimation tool for experiment outcomes from prior data.

2 Objective

2.0.1 Main Objective

To produce a lightweight ML tool that can estimate elimination probability and time window results of release experiments of genetically modified mosquitos.

*assisted by PhD. Benjamín Valdez (ITESM CQ) in collaboration with PhD. Héctor M. Sanchez (UCB)

2.0.2 Subobjectives

- Interpretability: Understanding why a classification was made by the model is expected. Also, some confidence on the prediction is desireable.
- Framework Compliant: Prefereably built around scikit-learn. Has to accept numpy arrays and save and load models with joblib.
- Flexible Resolution: Model can be readjusted to a different dataset; same experiment but different zone of interest.

3 Data Exploration

3.1 Features

We begin by describing our dataset. Each experiment run is configured by 5 parameters:

- (*pop*) Population size per node
- (*ren*) Number of weekly releases
- (*res*) Release size
- (*mad*) Adult lifespan reduction
- (*mat*) Male mating reduction

So, we have a feature describing the environment where the mosquitos are to be released, 2 features describing a modification on the mosquito organisms and finally 2 features to describe releasing modes.

3.2 Target Variables

MoNeT simulator outcomes can be analyzed with several metrics for each experiment configuration. From these we try to predict two:

- (*POE*) Probability of Elimination
- ($WOP_{threshold}$) Window of time that wild gene population is below a specific threshold.

Figure 1. shows several outcome instances by the simulator for the same configuration. In the vertical axis we have the proportion of the population of mosquitos with the wild gene we are trying to surpress. On the horizontal axis we have time in years. Each line represents a different outcome in the long run. We can observer for this configuration, that all outcomes at first reduce significantly the target population the first year , however, the population reestablishes in the following years for most of them.

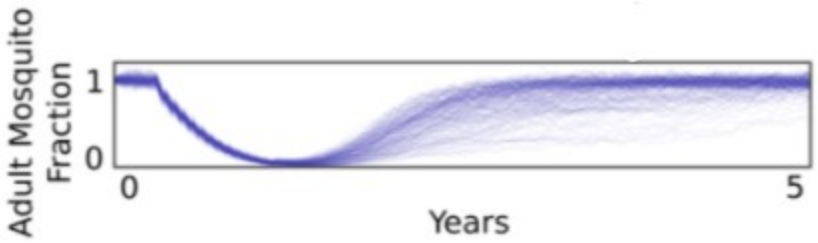


Figure 1: Different possible outcomes for a single arbitrary experiment configuration.

3.3 Dataset Organization

The dataset used for this project contains the results of 16,000 runs. However, each experiment run has several random factors so it commonly generates mulitple outcomes as shown in Figure 1. To account for this, simulation proportions are grouped by a worst case bound result and saved under a file with the specific percentile associated (50%, 75%, 90%). For example, let's say we read a Probability of Elimination Outcome (POE_i) of 0.83 for features x_i=($pop = 20, ren = 8, res = 10, mat = 30, mad = 20$) for the 50% file. This means that 50% of the simulations under this specific configuration resulted at least in a 70% probability of elimination. Then, if we were to see a POE value of 0.71 for the same configuration on the 90%, this would mean that configuration x_i, most probable outcome bad, as the metric got lower when taking into account the majority of the simulation results. Also, as more simulations are taking into consideration, the variance is expected to be lowest for percentil 50 and highest for percentil 90.

3.4 Histograms

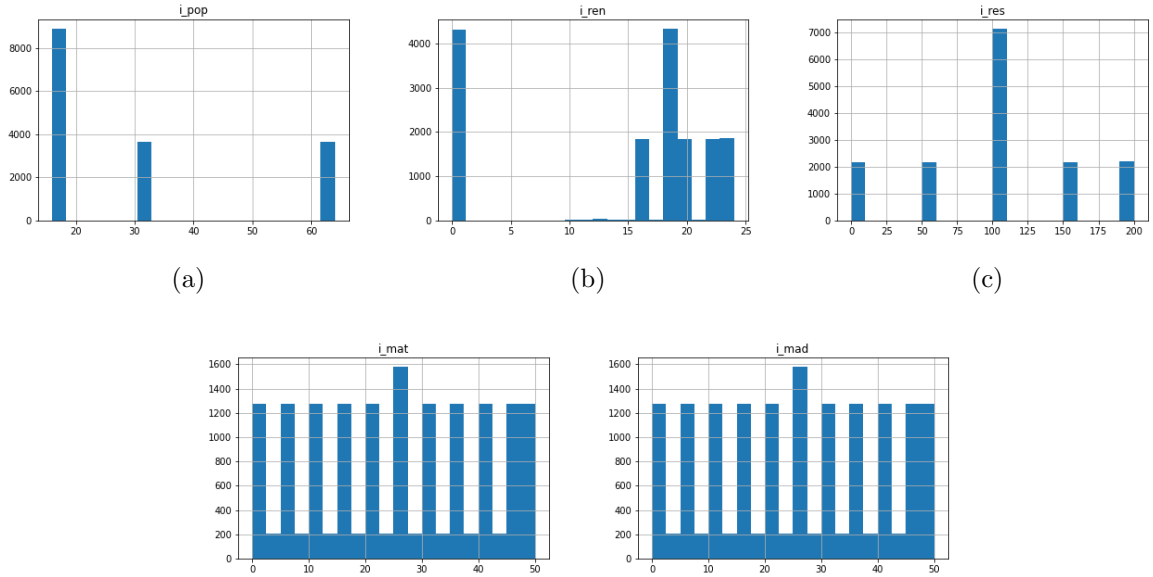
3.4.1 Features

After plotting each feature variable on it's own histogram in Fig. 2 we can see that all values are integer and some values have more presence overall. After couting the frequency of each unique value for each feature we confirm that to produce the dataset, certain values were given more weight by repeating it more times in simulation runs. For *pop*, *ren* and *res*, just a few values had majority weight. However, both *mad* and *mat* shared same importance across multiples of 5 from $[0 - 50]$.

3.4.2 Targets

Our targets are computed from aggregating multiple runs per configuration and are divided into 3 quantiles: 50%, 75%, 90%. We then will plot the each target variable distribution for each quantile.

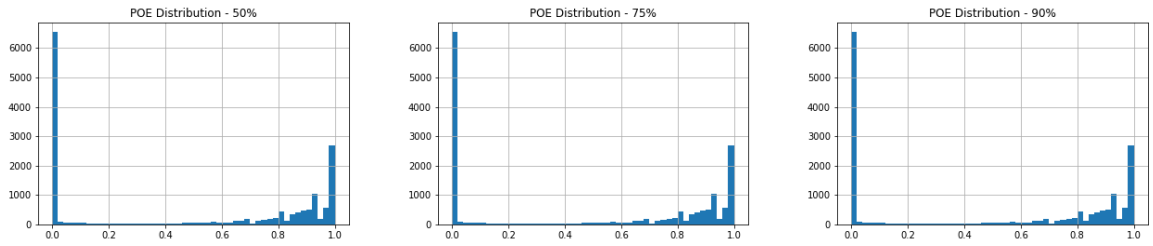
We find from Fig. 3 that probability of elimination is the same regardless the quantile matrics we are looking at.



(a) (b) (c)

(d) (e)

Figure 2: (a) pop (b) ren (c) res (d) mad (e) mat



(a) (b) (c)

Figure 3: (a) POE 50% (b) POE 75% (c) POE 90%

3.5 Correlations

3.6 Discretazing Target Variables

The results of this model will be used during brainstorm meetings and it are meant to loosely evaluate if some hypothesis are reasonable to explore. For that, we are expected to deliver categorical results. Instead of a real interval of values, we ought to give ['LOW', 'MID', 'HIGH'] labels.

To make easier our exploration analysis and training we discretized every target variable. We did so by dividing our target variable' ranges into 3 intervals all the points that lay within the same interval get assigned the same label. However, we did not uniformly divide the ranges to obtain the intervals, instead, we decided the boundaries from what we observed in the target histograms. For the data exploration we decided on the following intervals:

3.7 Understanding POE outcomes through decision trees

Our feature space is composed of discrete integer data and a low amount of unique values per features. Nonetheless, our relationships are far from linear, making it hard to analyze manually. At this point we tried to use a Decision Tree and analyze its' decision conditions to understand better the data.

3.8 Understanding WOP outcomes throught 3D plotting

4 Model Selection

4.1 Evaluating multiple configurations

4.2 Tree Results

4.3 KNN Results

4.4 NN Results

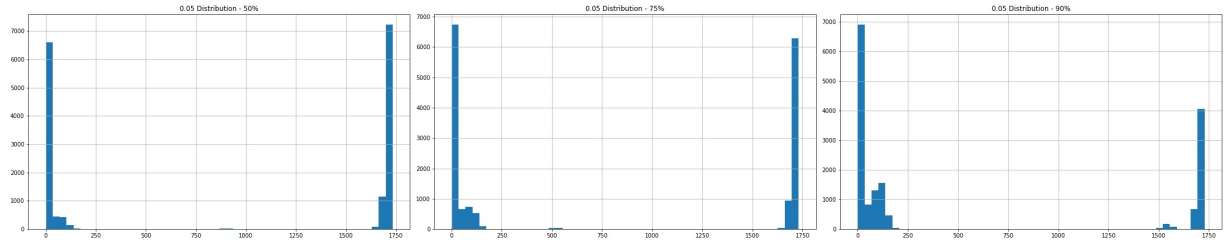
5 Pipeline Integration

6 Discussion

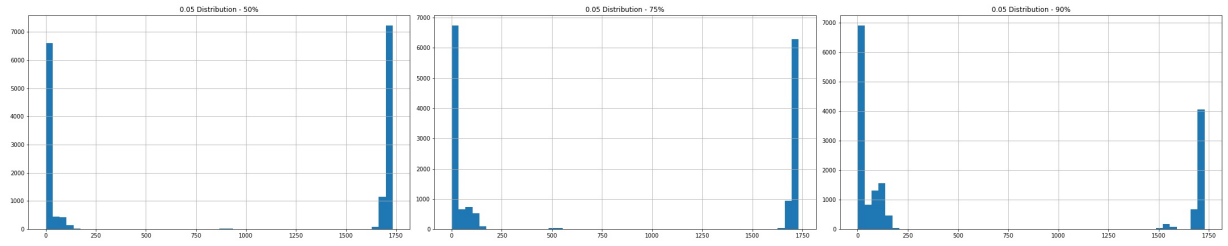
- Weight of specifics values for each feature might heavily influnce model training. Would be nice to verify results in interpolated points.

7 Conclusion

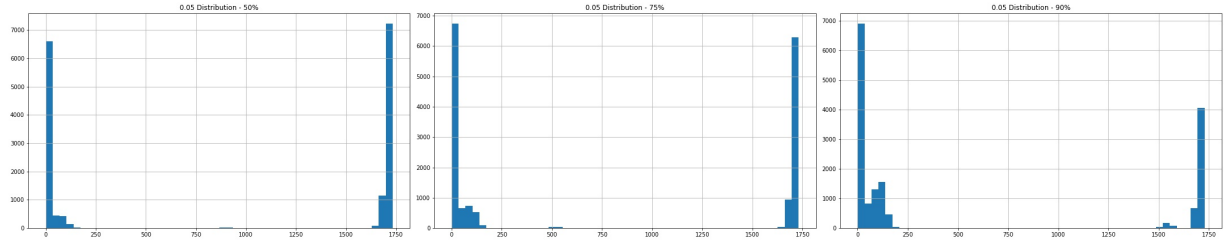
8 References



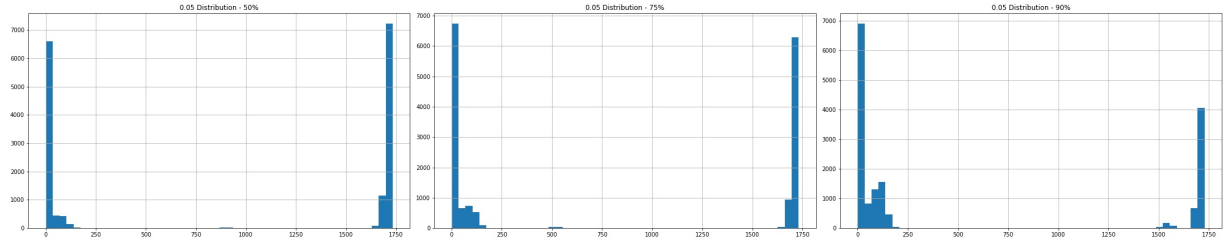
(a)



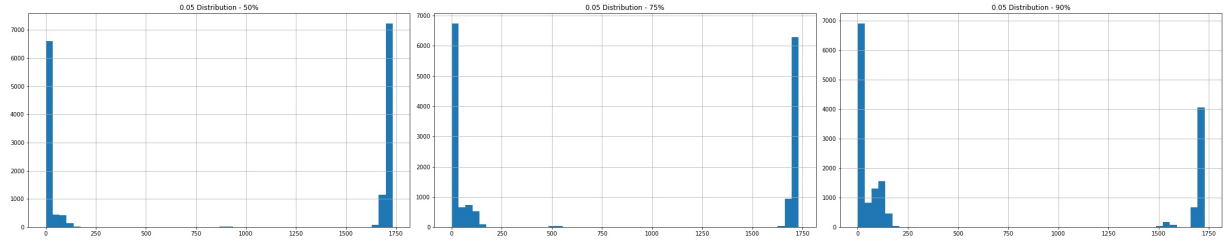
(b)



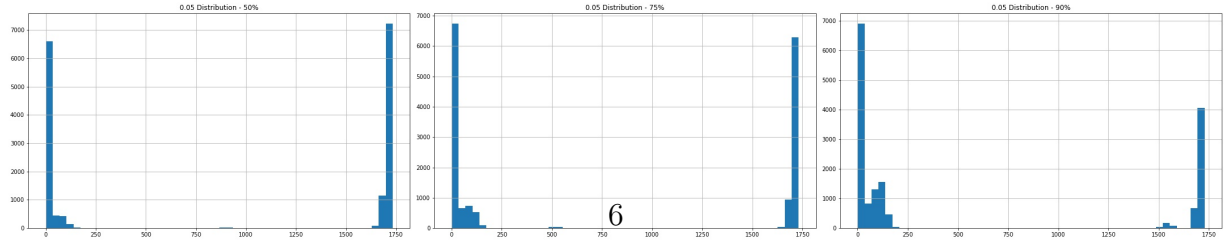
(c)



(d)

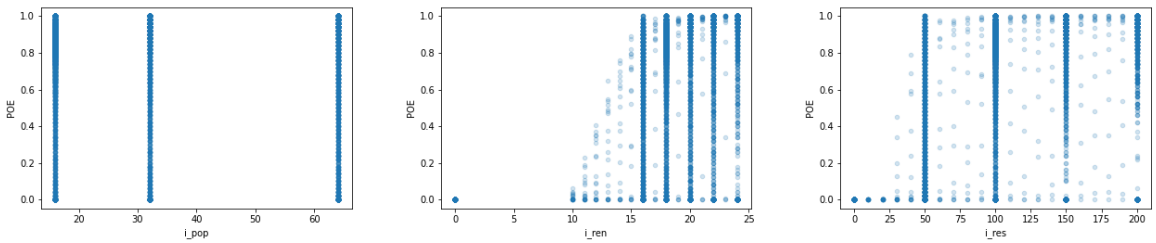


(e)



(f)

Figure 4: (a) $WQP_{0.05}$, (b) $WQP_{0.10}$, (c) $WQP_{0.50}$, (d) $WQP_{0.75}$, (e) $WQP_{0.90}$, (f) $WQP_{0.95}$



(a) (b) (c)

(d) (e)

Figure 5: (a) $pop - POE$ (b) $ren - POE$ (c) $res - POE$ (d) $mad - POE$ (e) $mat - POE$

Target	Low	Mid	High
POE	$[0 - 0.6)$	$[0.6 - 0.85)$	$[0.85 - 1)$
WOP_i	$[0 - 250)$	$[250 - 1500)$	$[1500 - 2000)$

Figure 6: Discretization intervals for data exploration