NOVA
IMS
Information Management School

NEURAL AND EVOLUTIONARY LEARNING - PROJECT

PREDICTIONS OF LACTOSE PERCENTAGE BASED ON DATA FROM AUTOMATED MILKING SYSTEMS

Andriani Kakoulli, Susana Pires
20230484 20230540

Contents

1. Brief Introduction

This project report is developed in the context of the curricular unit of Neural and Evolutionary Learning (NEL), where several models were studied. Its goal is to compare and discuss the purpose and performance on the models predictions of lactose contents, based only on data from milking robots. This data originates from Automated Milking Systems (AMS) employed at a cow milk production facility, where the cows chose when they go to the AMS to get milked.

As for the group approach on this project, it consisted of doing a context and surface analysis on the data since it was almost ready to use (rtu), then developing and evaluating the models, discussing the results and leaving suggestions on what could have been done better and future work.

2. Exploration and Data Analysis (EDA)

The initial step before the manipulation of the data was, of course, to explore through it, both visually and statistically, while understand its characteristics (notebook *1_EDA_Project*). Overall, it's possible to argue that the data, both the variables and the target, were almost cleaned and structured. The highlight was the small amount of data available to work with, which after further inspection on the missing values, would decrease even more.

The aforementioned missing values were only spotted in the column of *dry_days*, specifically on 147 rows matching with the records for the first lactation period. According to the metadata, the variable of *dry_days* is indicating the difference in days between the end date of the previous lactation cycle and the start date of the corresponding one. Thus, the rows with missing values matching with all the data of the first lactation was a reasonable relationship, since the cow, in its first lactation period, would not have any dry days recorded (there was not a preceding lactation). The chosen approach was to only use data from the second lactation onwards, so the next step was to remove the rows without any value for *dry_days* as dropping the same records on the target using the same indexes, as imputing them with zero would not make sense, because having zero days as *dry_days* has a different meaning of not having the chance to go through *dry_days*.

The visual exploration included histograms and bar plots of the variables, while the chosen statistical method for identifying the outliers was to define the lower and upper bound of each column's data using the IQR method. An imbalance in the lactation variable was observed, however the challenge was accepted due to the small amount of data available. Regarding the outliers, a first approach was to delete them, but that would leave even less data, thus, after some research on the topic[1, 2], and given a bigger sample, those values probably would not be classified as outliers. The second approach was to transform the data to be more normally distributed, by applying the logarithmic and the square root transformations respectively in each column.

Finally, the new cleaned dataset was concatenated with the target (added on the last column) and exported as a *csv* file *dt_cleaned* for future use on the four algorithms.

3. Models

3.1. Genetic Programming (GP)

Based on the functions of *gpolnel* library, the first steps were to load the data, split them to train-test and subsequently the train to train-validation, and to transform them to tensor datasets. Then, after checking the *gpol* implementation to define the search space, the problem instance and lastly, the genetic algorithm. Next, it was used the *solve* function to iterate through the genetic algorithm and get the best solution on each population. The

[1] Antanaitis R, Džermeikaitė K, Krištolaitytė J, Girdauskaitė A, Arlauskaitė S, Tolkačiovaitė K, et al. The Relation between Milk Lactose Concentration and the Rumination, Feeding, and Locomotion Behavior of Early-Lactation Dairy Cows. Animals. 2024 Jan;14(6):836.

[2] Piwczyński D, Siatka K, Sitkowska B, Kolenda M, Özkaya S, Gondek J. Comparison of selected parameters of automated milking in dairy cattle barns equipped with a concentrate feeding system. animal. 2023 Dec 1;17(12):101011.

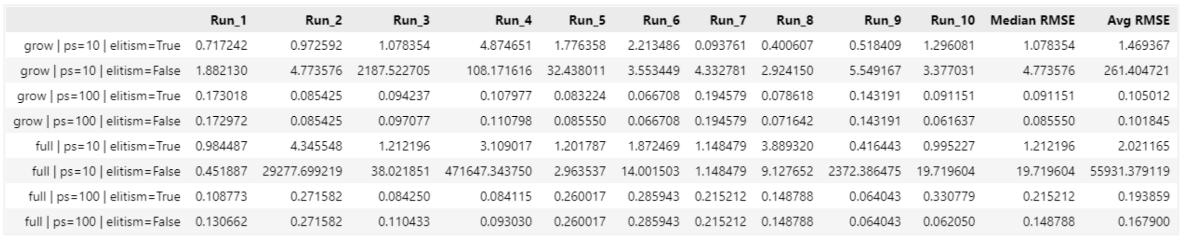
parameters of the final procedure were selected using a manual approach similar to *gridsearch*, which will be explained further in the following paragraph. Finally, for the evaluation of the GP model on the unseen test data, the median of the fitness was computed (which was the RMSE since its applied to a regression problem) based on the 10 runs of the *solve* function with the chosen parameters and the best individual of each run.

It is important to note that plenty iterations had been run to identify the best parameters. A simple comparison among the three selectors was done based on the results of the fitness. This comparison proved that tournament selection gave the best results in a reasonable amount of time, while rank selection and roulette wheel selectors resulted in bigger fitness scores. For the parameters of initializer, population size and elitism, the conducted *gridsearch* was testing the values of 10 and 100 for population size, the initializers '*grow*' and '*full*', and the binary values (*True*, *False*) for elitism. The value of each combination and each run was saved in a *DataFrame*, from which was calculated the median and performed comparisons. The "winner" of this *gridsearch* was the combination of the '*full*' selector, a population size of 100, and the choice of *True* in elitism, which gave the lowest median of RMSE with a value of 0.512142. This value will be used for evaluation of the four models, along with the 10-dimensional vector of the RMSEs for the selected combination. All of the description is available on file *2.1_GP*.

3.2. Efficient Geometric Semantic Genetic Programming (Eff_GSGP)

Concerning the implementation of the Efficient GSGP algorithm, which is available on notebook *2_2_Eff_GSGP*., the approach was done using the library *gpol* as follows. First, the data was loaded through a newly defined function *load_ams*, then it was splatted by indices into train, validation and test sets. Due to the small amount of available data, the split was of 30% to test, 20% to validation and 50% to training set. Following by the definition of the functional set, search space and problem instance. As for the next step of instantiation, after some details on saving the *log_files*, the algorithm was defined.

Considering the previous results on GP, the *selector* was set to *prm_tournament* with a selection *pressure* of 0.07 as default. For the experiments on improving this algorithm, they are described on **Figure 1**.



	Run_1	Run_2	Run_3	Run_4	Run_5	Run_6	Run_7	Run_8	Run_9	Run_10	Median RMSE	Avg RMSE
grow | ps=10 | elitism=True	0.717242	0.972592	1.078354	4.874651	1.776358	2.213486	0.093761	0.400607	0.518409	1.296081	1.078354	1.469367
grow | ps=10 | elitism=False	1.882130	4.773576	2187.522705	108.171616	32.438011	3.553449	4.332781	2.924150	5.549167	3.377031	4.773576	261.404721
grow | ps=100 | elitism=True	0.173018	0.085425	0.094237	0.107977	0.083224	0.066708	0.194579	0.078618	0.143191	0.091151	0.091151	0.105012
grow | ps=100 | elitism=False	0.172972	0.085425	0.097077	0.110798	0.085550	0.066708	0.194579	0.071642	0.143191	0.061637	0.085550	0.101845
full | ps=10 | elitism=True	0.984487	4.345548	1.212196	3.109017	1.201787	1.872469	1.148479	3.889320	0.416443	0.995227	1.212196	2.021165
full | ps=10 | elitism=False	0.451887	29277.699219	38.021851	471647.343750	2.963537	14.001503	1.148479	9.127652	2372.386475	19.719604	19.719604	55931.379119
full | ps=100 | elitism=True	0.108773	0.271582	0.084250	0.084115	0.260017	0.285943	0.215212	0.148788	0.064043	0.330779	0.215212	0.193859
full | ps=100 | elitism=False	0.130662	0.271582	0.110433	0.093030	0.260017	0.285943	0.215212	0.148788	0.064043	0.062050	0.148788	0.167900

Figure 1 - *RMSE values of GSGP.*

The solve procedure was cemented with the following parameters: *n_iter*=100, *tol*=0.001, *test_elite*=True, *verbose*=3, *log*=2. Since there was some code troubleshooting that went beyond the available experience, a different approach was determined to go around the difficulties of implementing a cross-validation method on the algorithm. As so, for each of the runs that got the best RMSE, the kernel was restarted and a new different seed was declared (to keep the reproducibility, the seed list is available in the code). The parameter combination that achieved better median fitness (RMSE) of 0.085550 was *initializer=grow*, *ps*=100, *elitism=False*.

3.3. Neuroevolution (NEvol)

The structure of this regression model was the implementation of a Neural Network with backpropagation. Before modelling, the first decision was to evaluate the four optimizers: *Gradient Descent* (GD), *Stochastic GD*, *MiniSGD* and *RMSProp*. This decision was made by training four models using one of the optimizers in each one, and with the use of the *k-fold cross validation* method to get more robust results for the best performing optimizer. As for the procedure, after loading the cleaned data and defining the parameters, the 5-fold CV was conducted splitting the data randomly in train and validation. In each of the five iterations, the NN instance was defined as well as its architecture, same for all four models. The following steps were to compile the models, set the batch sizes and move on to the training of the models, in parallel with storing the results of their evaluation metric. Lastly, having five values for the evaluation metric for each model, it was calculated the corresponding average mean absolute error (MAE) and compared the computed values using the ANOVA statistical test. The returned table lead us to the conclusion that only *RMSProp* was significantly different from the other optimizers, and due to the similar MAE values of *GD*, *SGD* and *MiniSGD* models, we decided to proceed with *Stochastic Gradient Descent*.

The construction of the NN model with backpropagation was implemented in 500 epochs, using the sequential model with two dense layers, the first with activation 'relu' and the second without an activation since it is performing on a regression problem with one output, since the predicted value is just the lactose percentage. The model was then compiled with the *SGD* optimizer, and after training, the evaluation on the test data was made by the root mean square error (RMSE). The final value of RMSE was calculated by the median of the ten runs' errors of this model, with the exact value of 0.057923, as available on notebook *2_3_Neuroevolution*.

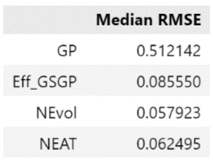
3.4. NeuroEvolution of Augmenting Topologies (NEAT)

Regarding the implementation of NEAT algorithm, which is available on notebook *2_4_NEAT*, the approach was done using the library *neat-python* imported from GitHub (https://github.com/CodeReclaimers/neat-python.git). First, the data was loaded through two csv files (*dt_cleaned.csv* for the features and *y_lact_cleaned.csv* for the target), then it was splatted into train, validation and test sets. Due to the small amount of available data, the split got 99, 33 and 45 instances for each of the corresponding sets. Followed by the configuration of NEAT, the *eval_genomes* function was defined for each of the sets. On a separate *Python script*, the configuration file was defined (on *config-feedforward-neat.py*), since it's on this file that's possible to adjust some of the parameters for this algorithm. Then it is run for 50 generations, until it outputs a *winner*, from which it's possible to get the *.format* (*genome*).

When predicting the results with NEAT, and to perform a better (but not perfect) evaluation, a similar approach to GSGP was taken to go around some challenges. Considering the defined parameters on the configuration file, this algorithm was run for 10 iterations, with a different seed on each. By saving the fitness values (RMSE) for each run, it's possible to calculate the median RMSE on test set, that achieved a score of 0.062495.

4. Discussion of the Results

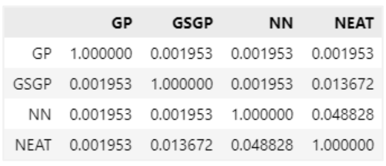
In respect to the results achieved throughout this project, **Figure 2** represents the median best fitness achieved on each algorithm. The comparison among the models was done using precise statistical testing methods. Those two methods were the *Wilcoxon* test and *Tukey's Honestly Significant Difference* (HSD) test.



	Median RMSE
GP	0.512142
Eff_GSGP	0.085550
NEvol	0.057923
NEAT	0.062495

Figure 2 - *Median RMSE per algorithm.*

The statistical analysis of the algorithms indicated whether there was a statistically significant difference between each pair of algorithms by metrics such as the *p-value* and the confidence interval. The pairwise **Wilcoxon test** for multiple comparisons gave an output of p-values to compare the paired finesses of the algorithms and thus, identify which of them differ significantly (**Figure 3**).



	GP	GSGP	NN	NEAT
GP	1.000000	0.001953	0.001953	0.001953
GSGP	0.001953	1.000000	0.001953	0.013672
NN	0.001953	0.001953	1.000000	0.048828
NEAT	0.001953	0.013672	0.048828	1.000000

Figure 3 - *p-values of the pairwise Wilcoxon test.*

Specifically, if the p-value was less than the chosen significance level, the conclusion was that the difference between the two corresponding algorithms was statistically significant. By displaying the heatmap of the results, we got a visual inspection of the p-values. The diagonal of the returned **Figure 4** indicated the comparison of each group with itself and thus we ignore its values.

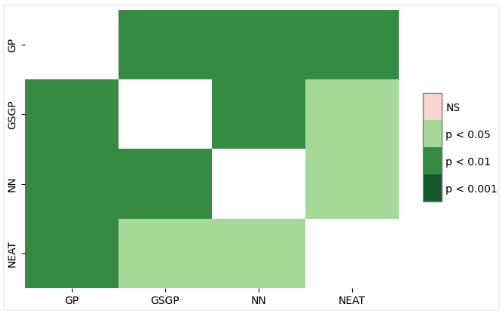
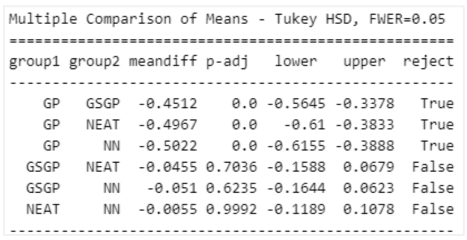


Figure 4 - *Heatmap of the pairwise Wilcoxon test.*

With the **Tukey's HSD test**, the deeper look into the statistical metrics provided a paired confidence intervals and a binary value of "True" or "False" (**Figure 5**) indicating whether we should reject the null hypothesis of no difference between the algorithms.



```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
group1 group2 meandiff p-adj   lower   upper  reject
-----------------------------------------------------
    GP   GSGP  -0.4512    0.0 -0.5645 -0.3378   True
    GP   NEAT  -0.4967    0.0   -0.61 -0.3833   True
    GP     NN  -0.5022    0.0 -0.6155 -0.3888   True
  GSGP   NEAT  -0.0455 0.7036 -0.1588  0.0679  False
  GSGP     NN   -0.051 0.6235 -0.1644  0.0623  False
  NEAT     NN  -0.0055 0.9992 -0.1189  0.1078  False
-----------------------------------------------------
```

Figure 5 - *Output result of the pairwise Tukey's HSD test.*

The tables and visualizations seen in the appendix and cross-referenced before were the exact results of the two tests. The interpretation of them led us to the conclusion that all pairs of GP were statistically different from each other at a significance level of 0.01. It is also observed that in Wilcoxon test, the pair of GSGP and NN had a p-value of 0.001953 which was less than the significance level of 0.01, but the Tukey's HSD did not reject the hypothesis and thus NN and GSGP algorithms were not significantly different.

5. Conclusion & Future Work

In conclusion, this project highlighted the strengths and weaknesses of different evolutionary and neural learning approaches in predicting lactose content from AMS data. The detailed comparison and statistical analysis provided insights into the most effective methods for this specific problem, as remembered by the *No Free Lunch Theorem* that guarantees there is no " 'super' algorithm that performs better than all the others on all possible existing optimization problems"[3].

As for future work, there are a few things to improve, such as the implementation of the algorithms with cross validation, improve code readability on the *gpol* library, and implement several other combinations of parameters as experiments to see if a better result can be achieved with one of the tested algorithms.

[3] Vanneschi L, Silva S. Lectures on Intelligent Systems [Internet]. Cham: Springer International Publishing; 2023 [cited 2024 May 31]. (Natural Computing Series). Available from: https://link.springer.com/10.1007/978-3-031-17922-8