OPTIMIZING BINARY CLASSIFICATION USING GENETIC ALGORITHMS

ARTIFICIAL INTELLIGENCE OF BIOLOGICAL INSPIRATION

Charbonnier Lucas

Universidad Carlos III de Madrid

December 14, 2025

Contents

1 Introduction **2**

2 Dataset Presentation and Preprocessing **2**

3 Reference Models (Baselines) **3**

4 Genetic Algortihm for feature selection **4**

4.1 Presentation 4

4.2 First results 6

4.3 Integration of Mutation 7

4.4 Tested Variants: Selection, Crossover, and Replacement 8

5 Conclusion et perspectives **9**

1 Introduction

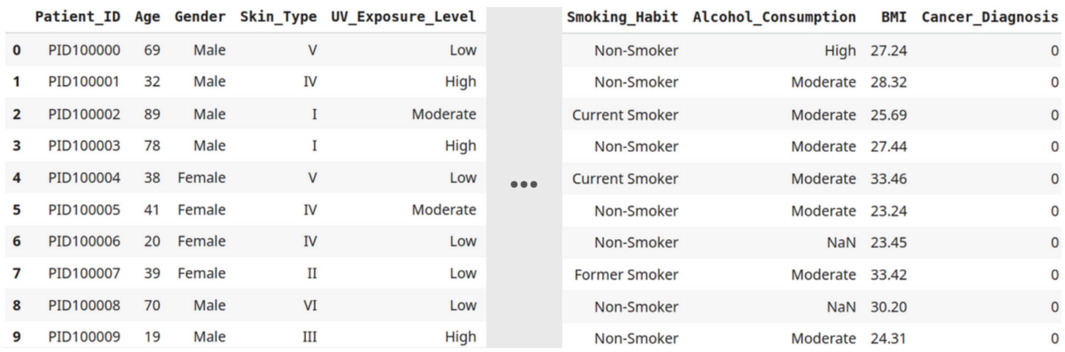
Early detection of skin cancer is a major medical challenge: rapid and reliable diagnoses significantly improve patient prognosis. Modern medical datasets often contain numerous features (clinical measurements, image descriptors, demographic metadata), yet a significant portion of these variables may be redundant or uninformative. Feature selection aims to identify a minimal subset of variables designed to improve predictive performance while reducing model complexity, thereby facilitating interpretability and decreasing computational costs.

In this work, we explore the use of a Genetic Algorithm for feature selection on a skin cancer dataset. Genetic algorithms are optimization methods inspired by the mechanisms of natural evolution: they operate on a population of candidate solutions (chromosomes), evaluate their aptitude via a fitness function, and then apply simulated biological operators to produce increasingly adapted solutions.

The objective of this project is twofold: first, to establish a performance baseline by comparing reference models (Logistic Regression, Random Forest, SVM, KNN); second, to design and test a practical implementation of a genetic algorithm to select a feature subset leading to comparable or better performance while reducing the number of variables.

2 Dataset Presentation and Preprocessing

The project relies on a medical dataset focused on skin cancer detection. This dataset comprises 3,000 observations and 23 features, each representing a clinical measurement, a patient habit, or a descriptor derived from dermatological image analysis. The objective is to predict a binary target variable indicating the presence or absence of malignancy.



	Patient_ID	Age	Gender	Skin_Type	UV_Exposure_Level	...	Smoking_Habit	Alcohol_Consumption	BMI	Cancer_Diagnosis
0	PID100000	69	Male	V	Low		Non-Smoker	High	27.24	0
1	PID100001	32	Male	IV	High		Non-Smoker	Moderate	28.32	0
2	PID100002	89	Male	I	Moderate		Current Smoker	Moderate	25.69	0
3	PID100003	78	Male	I	High		Non-Smoker	Moderate	27.44	0
4	PID100004	38	Female	V	Low	...	Current Smoker	Moderate	33.46	0
5	PID100005	41	Female	IV	Moderate		Non-Smoker	Moderate	23.24	0
6	PID100006	20	Female	IV	Low		Non-Smoker	NaN	23.45	0
7	PID100007	39	Female	II	Low		Former Smoker	Moderate	33.42	0
8	PID100008	70	Male	VI	Low		Non-Smoker	NaN	30.20	0
9	PID100009	19	Male	III	High		Non-Smoker	Moderate	24.31	0

Figure 1: Skin cancer Dataset

Certain dataset variables are not directly suitable for machine learning models and must be converted into a numerical format. For instance, the feature `UV_Exposure_Level` consists of three categories: *Low*, *Moderate*, and *High*. To make this information interpretable by supervised learning models, it was transformed using One-Hot Encoding. Rather than assigning an implicit order to the categories, this method creates a binary column for each level. Post-encoding, the dataset contains a total of 40 features.

However, the class distribution remains imbalanced, with 2,568 negative cases and only 432 positive cases. Consequently, this led us to use the F1-score as the evaluation metric rather than simple accuracy.

To objectively evaluate model performance, the data was split into a training set (70%) and a test set (30%). This separation ensures that the final evaluation is performed on unseen data, thereby mitigating the risk of overfitting. For certain experiments, particularly during extensive testing with the genetic algorithm, we used a reduced version of the dataset (100 samples) to decrease computational time.

3 Reference Models (Baselines)

Before introducing the genetic algorithm, it is essential to establish baseline models to benchmark their performance against the results achieved using the genetic algorithm. To this end, several classification models from the `scikit-learn` library were trained on the entire set of available features : Logistic Regression, KNeighbors, Random Forest, and SVM.

To obtain reliable baselines, each classifier was configured with a set of simple yet consistent hyperparameters, without attempting exhaustive optimization. For Logistic Regression, the regularization parameter $C = 2$ was selected following a few empirical trials. For KNN, various values of k ranging from 1 to 20 were tested, and the value yielding the highest accuracy on the test set was retained. The Random Forest was set to 100 trees, a classic trade off between performance and computational cost. Finally, for the SVM, the RBF kernel was retained as it often represents a robust choice for non-linear data.

These models were evaluated using accuracy and the F1-score, a metric particularly relevant in our case given the dataset's significant class imbalance.

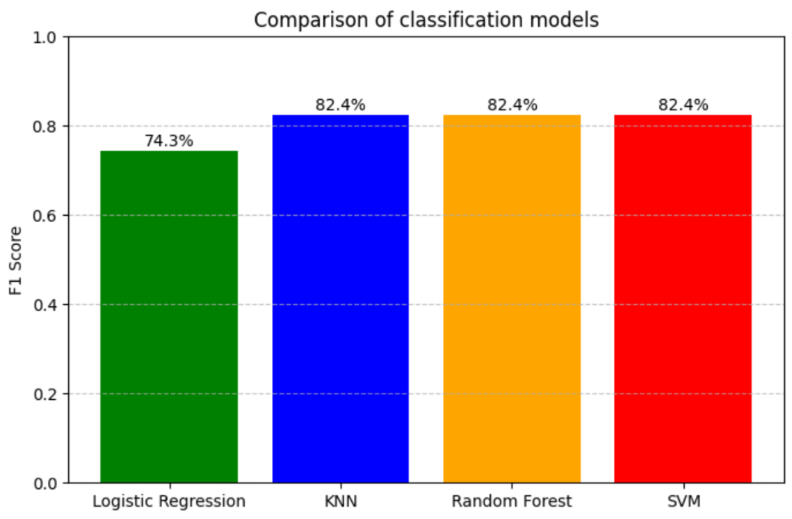


Figure 2: Baseline results

The KNN, Random Forest, and SVM models give the same results. From here on, let's use, for example, Random Forest as a baseline and try to improve the score.

4 Genetic Algortihm for feature selection

4.1 Presentation

To enhance the performance of classical models, we employ a genetic algorithm designed based on biological evolutionary principles. The primary objective of this project is to use a genetic algorithm to automatically select an optimal subset of variables capable of improving classification performance.

General Principle

According to natural selection, the fittest individuals are the most likely to reproduce. Their characteristics are transmitted to their offspring through crossover and mutation mechanisms, thereby increasing their chances of survival.

Initial Population Generation

We initialize a starting population of arbitrary size, composed of chromosomes. Each chromosome is a binary list containing as many elements as there are features in the dataset. It indicates which dataset features will be retained during training with the classical model.

For example, the chromosome `[1, 0, 1, 1, 0, ... , 0]` indicates that only features 1, 3, and 4 are used for model training.

The initial population consists of two types of individuals:

- An elitist portion composed of full chromosomes (all set to 1) to ensure the complete model is part of the evolutionary process.
- The remaining individuals are generated randomly, with a number of active variables ranging between 30% and 70% of the total variables.

Evaluation

We must define a fitness function that measures the quality of each chromosome. For each individual :

- Selected variables (genes set to 1) are retained in the data.
- A classifier (Random Forest, selected as the best performing model) is trained on the reduced training set.
- Fitness corresponds to the F1-score obtained on the test set.

Consequently, individuals producing better scores are more likely to be selected.

Selection

Selection aims to choose the individuals that will be allowed to reproduce. While several selection modes exist, we initially adopt the tournament method :

- A subset of the population (5 individuals) is selected at random.
- The best of the 5 is chosen as a parent.
- The process is repeated until the desired number of parents is obtained.

This method favors high performing individuals while retaining an element of chance.

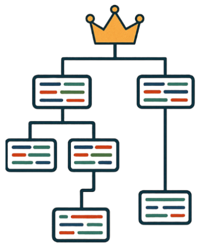


Figure 3: Tournament selection method

Reproduction

Reproduction aims to construct child chromosomes derived from two parents. We initially use ”One-point” crossover :

- A crossover point is chosen randomly.
- The segments of the two parents are swapped to produce two children.

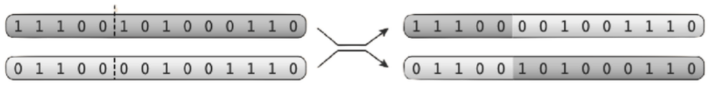


Figure 4: One-point crossover reproduction method

Replacement

Replacement consists of constructing the new generation by choosing among children and parents. We initially use elitist replacement :

- We keep the 20% best parents for the N+1 generation.
- The rest of the N+1 generation is filled with the children.

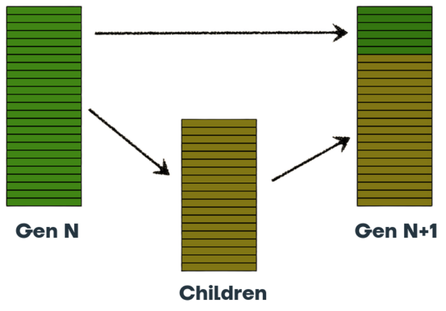


Figure 5: Elitist replacement method

Global Gentic Algorithm

The integration of these distinct stages culminates in the complete algorithmic workflow illustrated in the figure below.

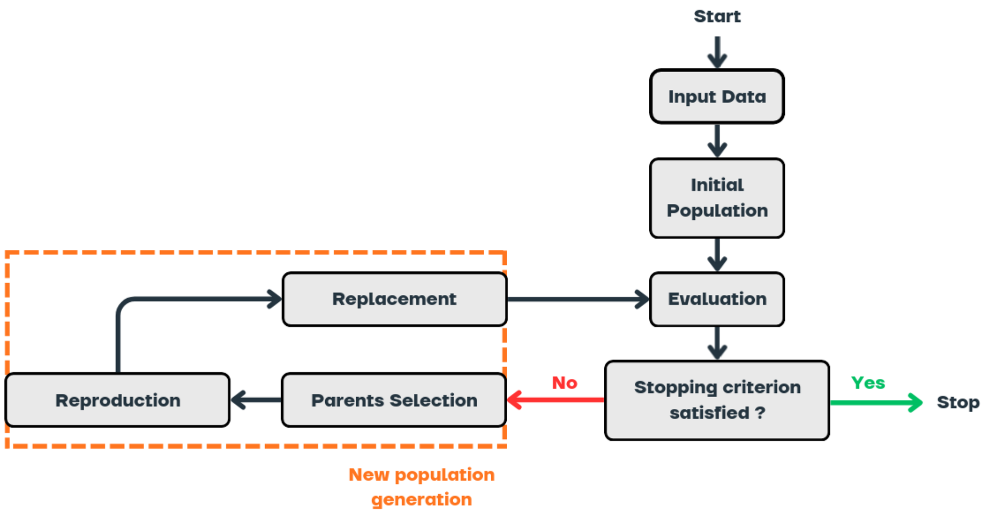


Figure 6: Global Overview of the Implemented Algorithm

4.2 First results

The following results illustrate the performance achieved after the implementation of the first version of the genetic algorithm.

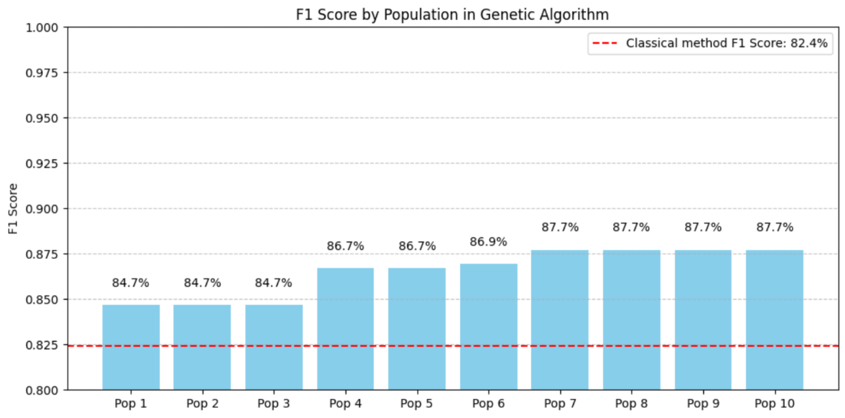


Figure 7: Results obtained with the first version of the algorithm

After just one generation with a population of 10 individuals, the algorithm yielded a 5.3% performance gain compared to the baseline. These results represent a solid starting point. However, to prevent premature convergence and reach higher score, the next iteration will incorporate a mutation operator

4.3 Integration of Mutation

To more closely mimic the natural evolutionary process and further optimize performance, we introduce a mutation stage :

- For each gene, a mutation occurs with a probability equal to the `mutation_rate`.
- If a mutation occurs, the gene is bit-flipped ($0 \rightarrow 1$, $1 \rightarrow 0$).

The final global workflow is therefore updated as follows :

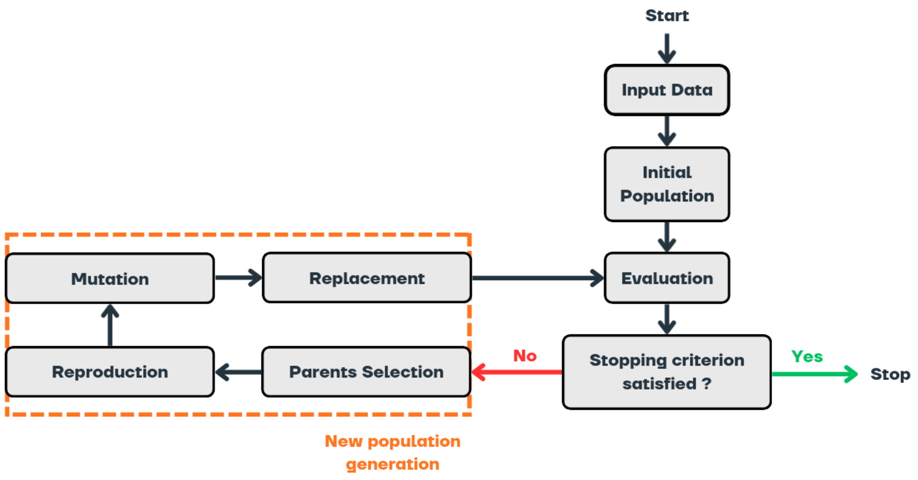


Figure 8: Final Algorithm Workflow including Mutation

After running the algorithm for 10 generations with a mutation rate of 10%, here are the results obtained :

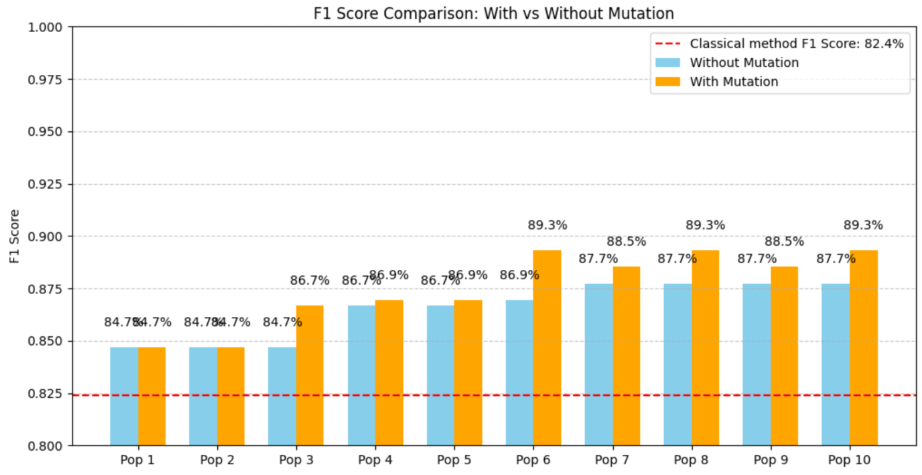


Figure 9: Results obtained with the mutation step

We observe that with the mutation step, the obtained score is generally better for each population. Moreover, we manage to reach a final score of 89.3%, which corresponds to an increase of 6.9% compared to the initial score.

This result is very good and shows that the algorithm works efficiently, preventing premature convergence.

However, further modifications can be made, notably by changing the selection / reproduction / replacement modes, in order to increase even more the final score.

4.4 Tested Variants: Selection, Crossover, and Replacement

Beyond the baseline version of the Genetic Algorithm, several optimizations were explored to enhance the quality of the resulting solutions.

First, different selection modes were tested : *Roulette Wheel Selection*, where the probability of being chosen is proportional to performance, favoring promising solutions while maintaining diversity. Also, *Sorting based Selection*, which consists of retaining only the best individuals at each generation.

In parallel, another reproduction strategy was experimented with. In addition to simple one-point crossover, a *Multi-point Crossover* was used to blend the information contained in parental chromosomes more finely.

Finally, more aggressive replacement strategies, including *Pure Elitism*, were evaluated: only the best solutions of each generation are preserved, guaranteeing a monotonic progression of fitness but at the risk of reducing genetic diversity.

After several experiments with various combinations, the optimal configuration yielded superior results. This configuration specifically utilized *Sorting based Selection* to prioritize high-fitness individuals, *Multi-point Crossover* to promote diversity through complex chromosomal recombination, and an *Elitist Replacement* scheme for generating the subsequent population.

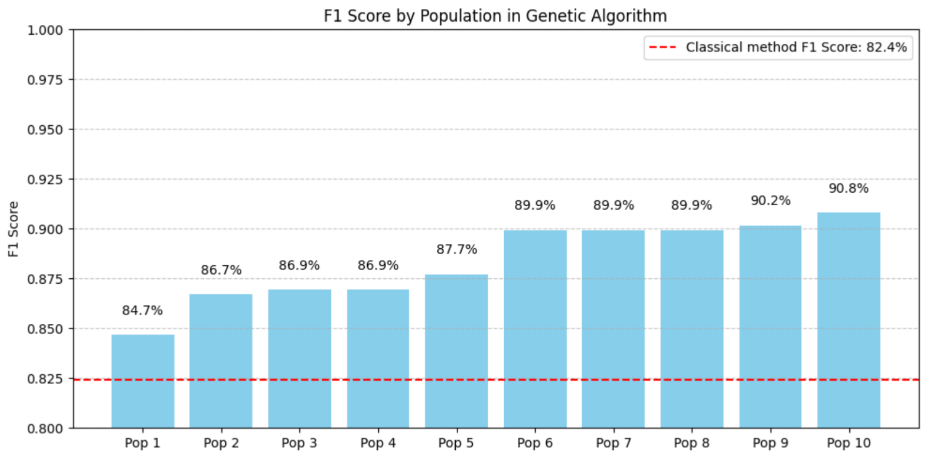


Figure 10: Results obtained with the best combination

Indeed, we obtain the best score with this version, 90.8%. This represents an increase of 8.4% compared to the initial score.

This combination is the most effective because it balances convergence and diversity. *Sorting based Selection* quickly promotes the best individuals, *Multi-point Crossover* enhances genetic diversity and exploration, and *Classical Replacement* preserves the top 20% of parents while allowing new offspring to improve the population. This balance prevents premature convergence and leads to better overall performance.

5 Conclusion et perspectives

In this project, we successfully demonstrated the effectiveness of a genetic algorithm applied to feature selection for the diagnosis of skin cancer. Starting with a clinical dataset characterized by high dimensionality and class imbalance, our objective was to identify a subset of variables capable of maximizing the predictive performance of a Random Forest classifier.

Our experiments highlighted the importance of the evolutionary strategy. While a naive implementation of the genetic algorithm already provided a performance gain of 5.3% over the baseline, the integration of a mutation operator proved crucial in preventing premature convergence, raising the improvement to 6.9%. Furthermore, by fine-tuning the selection, reproduction and replacement mechanisms, we achieved a final score of 90.8%. By the way, this optimal performance was obtained by retaining **only 18 features out of the total 40.** This represents a total improvement of **8.4%** compared to the reference model trained on all features.

We can also have a look to accuracy even if in our case it's not the most relevant metric, but often considered a more intuitive. This optimal model reached 0.92 of accuracy compared to the baseline of 0.88, representing a clear **4%** improvement.

These results confirm that reducing the number of features does not merely reduce computational costs, because it also effectively eliminates noise and redundancy, allowing the classifier to focus on the most discriminative patterns for malignancy detection.

In conclusion, this study illustrates that bio-inspired artificial intelligence, specifically Genetic Algorithms, offers a powerful and flexible framework for solving complex optimization problems in medical diagnostics.