Optimal design of antimicrobial peptides using a Genetic Algorithm based on supervised models

Jeremy Guerrero[a], Francesca Rivas[a], Yangzuo Wang[a], Cheng Wen[a]
[a] *Sup'Biotech - Biotechnology Engineering School, Villejuif, France*

***Abstract*—New machine learning techniques using Naïve Bayesian and XGBoost classifiers, and a Genetic Algorithm is presented for optimal design of antimicrobial peptides with activity against cancer cells. The Naïve Bayesian classifier discriminates sequence-relative descriptors, and the XGBoost classifier discriminates physicochemical descriptors well selected using a database of more than two thousands active and inactive anticancer peptides. Using these descriptors, we update the sequence of already active peptides with a driven genetic algorithm. The algorithm is optimizing the global score of each peptide by in-silico replacement of single or paired amino acids that were found the most frequently observed in optimal peptides. The best peptide will be experimentally validated in a laboratory environment on cancer cell lines.**

***Keywords* — Antimicrobial peptides (AMPs), descriptors, Naïve Bayesian, XGBoost, Genetic Algorithm (GA).**

INTRODUCTION

Antimicrobial peptides (AMPs) are amphiphilic polypeptides with a sequence of less than 50 amino acids. These short peptides exist in nature to be used as part of the innate immune system of several multicellular organisms. By identifying foreign substances or antigens, AMPs act to defend the host against some invaders such as bacteria, fungi, parasites and viruses [1,2]. Owing to the increase in antibiotic-resistant microorganisms and the global health issues related to it, AMPs have been seen as potential candidates for the development of new therapeutic methods in medicine, and new ways of protecting crops in agriculture [2]. However, the challenge of finding peptides with specific properties is complex, especially if they must be non-toxic to humans regardless of the application provided. As a result, it would be advisable to develop software tools for the *in-silico* analysis of these peptides by predicting its physicochemical properties and bioactivity in such a way that the best results are then tested in a laboratory environment. Thus, peptide synthesis and testing costs are minimized.

Developing software tools to predict the output of biological experiments (e.g. Peptide bioactivity) is one of the most important objectives in bioinformatics. Most of the *in-silico* prediction tools are based on statistical inferences from known biological data. For example, using molecular descriptors like the sequence of a peptide and its conformation (dipeptides or tripeptides) could be a good approach for finding ways to classify it according to its effectiveness. This is related to the concept of Quantitative Structure-Activity Relationship (QSAR) that is largely used in pharmacology and toxicology [3].

In this paper, we will develop two machine learning algorithms (XGBoost and Naïve Bayesian) that will provide us with a better understanding of which types of descriptors are the most suitable for the classification of peptides, helping us to direct the QSAR by means of a genetic algorithm (GA) for the improvement of peptides using a dataset of AMPs with anticancer activity (ACPs).

MATERIALS AND METHODOLOGY

I. Data preparation

During this work, the Database of Antimicrobial Activity and Peptide Structure (DBAASP) was used as a data source [4]. This database (https://dbaasp.org) is a project done by Giorgi Gogoladze et al. Preference was given to peptide sequences that are short in length (≤ 20 aa). This was done due to a limit in computational capacity and the amount of data processed. In this way, our QSAR can more effectively recognize the most stable and active parts of peptides.

A. Database of Antimicrobial Activity and Structure of Peptides (DBAASP)

A list of peptides with previously identified activity through *in-vitro* experimentation can be found in DBAASP. This database contains information on the structure of AMPs and their specific activity against target organisms gathered from the PubMed website (http://www.ncbi.nlm.nih.gov/pubmed). Thus, it was possible to find essential information about each peptide such as: Target object (Lipid bilayer, proteins, DNA, and others), hemolytic and cytotoxic activity, and physicochemical properties. These descriptors were essential for the classification of peptides using our machine learning algorithms. **Table I** contains the majority of the key features used in our study.

	Index	Key Features
	1	Normalized Hydrophobicity
	2	Net Charge
	3	Isoelectric Point
	4	Penetration Depth
	5	Tilt Angle
	6	Disordered Conformation Propensity
	7	Linear Moment
	8	Propensity to in vitro Aggregation
	9	Angle Subtended by the Hydrophobic Residues
	10	Amphiphilicity Index
	11	Propensity to PPll coil
	12	Length

TABLE I. Peptide features used from DBAASP.

B. BioPython: Helicity and Stability

The helicity percentage tells us about the fraction of amino acids which tend to be in the helix in the protein structure. Ho Yeon Nam et al. [5] showed that by including monomeric mutations to selected peptides in order to increase their helicity index, it is achievable to improve their activity and selectivity against antimicrobial membranes. This characteristic is also related to the hemolytic percentage and cytotoxicity of a given peptide. The correlation between both properties was extensively tested by McLaughlin and coworkers where they reported that the degree of helicity of the KLA peptides was proportional to the cytotoxicity [6].
The stability of a peptide tells us about the half-life that it can have when being in the environment or within a multicellular organism. This quality is especially important because it indicates the travel time capacity of the peptide. In other words, the more stability the peptide has, the more likely it will conserve its structure and hit its specific target. In the research by Kunchur Guruprasad et al [7], 400 possible dipeptide structures were described and it was concluded that 80 were directly related to a development of instability. By containing any of these dipeptides, it made a protein more susceptible to spontaneous degradation. Several hypotheses have been developed to explain this occurrence, one of them expresses that stability is a consequence of the differential occurrence of certain amino acids. The presence of amino acids such as Met (M), Gln (Q), Pro (P), Glu (E) and Ser (S) are presented with a relatively high frequency in unstable proteins. A second approach mentions that the instability or stability characteristics are governed by an arrangement of certain amino acids in a specific order. Therefore, the appearance of one amino acid in juxtaposition with the other could be a significant factor in determining the stability of the protein. **Table II** contains the final features added into our database used in this project.

	Index	Key Features
	13	Molecular Weight
	14	Helicity Percentage
	15	Instability Index

TABLE II. Peptide features extracted from BioPython.

C. Emergent features: Single peptide occurrence, Dipeptide and 4 Gap Peptide as descriptors

The choice of suitable descriptors is crucial for the quality of statistical models to predict the activity of a peptide. During the study we realized that certain dipeptide structures and 4-gap peptides continued to appear constantly. For example, the K-K pair is continually seen in stable peptides, as is the combination of K at location *j* and K at location *j* + 4 (e.g. K ---- K). That is why as a novel approach, we have generated a library of single peptides, dipeptides and 4-gap peptides to take them as new descriptors and label them depending if each one appears in a stable or unstable peptide. This will also be considered for the development of our Naïve Bayesian Classifier and our GA.

II. Data classification

For this paper, the stability is a characteristic related to both the secondary structure of the peptide and its functionality. That is why we have chosen it as a classification feature. The stability index was obtained through the Biopython package and the threshold for this was selected taking into consideration the research of Dilani G. Gamage et al. where they explain that an index less than 40 characterizes a stable peptide [8]. Furthermore, the helicity percentage was used as another classifier to optimize our ACPs. It has been considered that peptides should have a percentage of helicity determined by a threshold ($0.5 \leq x$), in this way their structure will have a greater specificity and capacity to disrupt the lipid membrane of cancer cells. This is because an alpha-helix amphipathic structure particularly has activity towards membranes containing phosphatidylcholine, a lipid commonly found in mammalian plasma membranes [9]. Finally, a threshold on the right ($x < 0.8$) was selected so that these peptides do not have a toxic or hemolytic activity against human cells [6].

III. Machine learning techniques used for ACP classification

By using the known properties of ACPs, the challenge of predicting stable peptides is significantly reduced since it requires fewer resources and less processing time. This is possible with machine learning algorithms where classes of peptides are identified by means of observations or descriptors included in a training dataset [10]. We take 2 approaches to recognize the importance of ACPs descriptors and thus be able to classify and predict their stability. On the one hand, the XGBoost algorithm was used to classify peptides as Stable and Unstable based on their physicochemical features. On the other hand, we employed a Naïve Bayesian algorithm to use the peptide structures found in DBAASP and predict the stability score of each peptide through the frequency of appearance of its parts in the libraries created by us that store stable and unstable structural parts.

A. XGBoost algorithm

The Extreme Gradient Boosting (XGBoost) developed by Tianqi Chen, is a scalable algorithm where a strong predictive model is built through assembling many weaker models which result in a Decision Tree bigger in size than other machine learning models [11]. Gradient Boosting happens by optimizing the loss function. In linear regression, there is an attempt to fit a linear equation for modeling the relationship between two variables. Using the least squares method, it is expected to find the best fit by minimizing the errors between the data points and the line to be drawn. This mathematical loss optimization is given by the sum of the n

squared errors that should be near to 0. The same concept is applicable for our boosting algorithm [11].

First, having a dataset of independent characteristics in Table I and II (input), we seek to predict the stability of the ACPs (output). The algorithm intends to create a base model and to calculate the errors (or residuals). The residuals will be helpful to generate another fit model (or residual model) and to predict new outputs. This occurs until the loss function has been minimized [12].

$$Final\ prediction = Base\ value + \sum_{i=1}^{n} (LR \cdot RP_i\ by\ RM_i) \quad (1)$$

Where LR is the Learning Rate, the speed at which the predicted value is expected to change; and $RP_i\ by\ RM_i$ is the i^{nd} Residual Prediction by the Residual Model i. Additionally, in XGBoost there are 3 more parameters to consider: λ for regularization, γ for controlling the overfitting, and *eta* that is related to how fast the algorithm converges to the next value. From these, the XGBoost algorithm gives rise to a Decision Tree where each node of the tree needs a Similarity Score (*SS*) based on the Sum of the Residuals (*SR*) and their number (*#R*).

$$SS = \frac{SR^2}{\#R + \lambda} \quad (2)$$

The value for λ has a great impact on the XGBoost algorithm, but it will not be taken into account to a great extent within this paper.

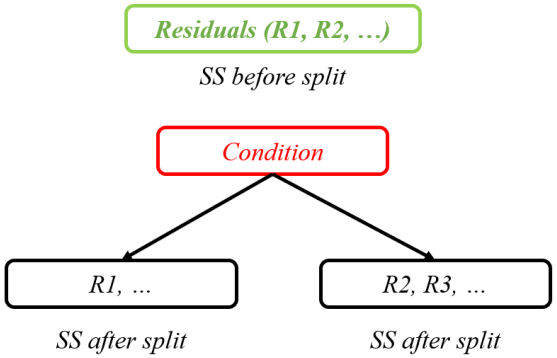

Figure 1. Example of a XGBoost tree.

In the case of γ, if it is less than the gain value ($G = SS_{after\ split} - SS_{before\ split}$) there is more probability in one branch or split to happen than another one. This is where the decisions within the tree are determined [13].

For our study, λ, γ and *eta* were established as 1, 0, and 0.3 respectively [14]. This is because it can influence our algorithm to be more conservative (less changes in the data structure that the algorithm manages). Also, with Python we can use other specific parameters for the *XGBoostClassifier* command of the *Scikit-learn* library depending on the learning of our model. We have selected: a *'binary: logistic'* classification, *early_stopping_rounds* as 10 (if in 10 rounds the model does not improve the loss function, then the training is stopped), and *eval_metric* (or loss function) as *'logloss'*. From the dataset of ACPs, we trained our algorithm using 14 descriptors from **Table I** and **II** without including Instability. As mentioned before, this last feature was relevant for classifying the ACPs as Stable or Unstable.

B. Naïve Bayesian algorithm

The Naïve Bayes methods are a set of supervised learning algorithms that apply the Bayes' theorem assuming that there is no relationship between the features.

$$P(Y|X) = \frac{P(X|Y)\,P(Y)}{P(X)} \quad (3)$$

Where $X = (X_1, X_2, ..., X_n)$ are the key features, $Y \in \{"Stable", "Unstable"\}$, and $P(Y|X)$is the probability of Y given X. Even if the features depend on each other, these independently contribute to the probability of a specific data point being in a certain class. This is the reason why it is called Naïve [15].

For our model, the peptide stability was predicted by recognizing the structural (or emergent) descriptors in each sequence and in the whole dataset according to their probability. In other words, if there is a peptide with the following sequence KRIVQRIKDFLR, its composition (single peptides, dipeptides, and 4-gap peptides) and its probability given Y will be stored in libraries. For example:

$d\_single = CreateDictionarySingle(sequences)$

Where:

d_single= {K:[2, 1.34], R:[3, 0.5], I:[2, 0.98], (...)}

The frequency of each descriptor in the Stable and Unstable set of peptides is a measure of how this descriptor discriminates between these 2 classes of peptides. In terms of probability, this ratio can be calculated as:

$$score(X_i) = \frac{P(X_i|Stable)}{P(X_i|Unstable)} \quad (4)$$

And simplified as following:

$$score(X_i) = \frac{Counts\ X_i\ in\ Stable}{Counts\ X_i\ in\ Unstable} \quad (5)$$

IV. Genetic algorithm used for ACP improvement

A Genetic Algorithm (GA) is a type of optimization algorithm, usually employed to find the maximum or minimum of a function. GA imitates the natural process of biology where certain organisms develop random mutations in different protein structures, staying with those that present the best functionality. However, instead of a totally random combination, it is possible to set the level of randomization and the level of control. Thus, it is more powerful and more efficient than random search and exhaustive search algorithms [16]. In this project, 50 stable sequences and 50 unstable sequences were chosen as the initial parents; some of them have the possibility to reproduce descendants by CrossOver. To simulate the real natural condition, the possibility to have kids is linked to their performance (Bayesian score). We designed a Roulette-like selection model [17]. First converting the Bayesian score of each sequence to integer from 10 to 70, this number will decide how many numbers from 0 to 140 a sequence will randomly

create. Then, we have created 4 random numbers from 0 to 140. If any 4 of them are included by the list of random numbers each sequence created, this sequence will be picked out to do CrossOver with others chosen sequences. This is the same as the Roulette game: the higher the score of a sequence is, this sequence will take more places on the Roulette, then we turn the roulette 4 times, if in any round the roulette stops at the number the sequence took, this sequence will be chosen as winner.

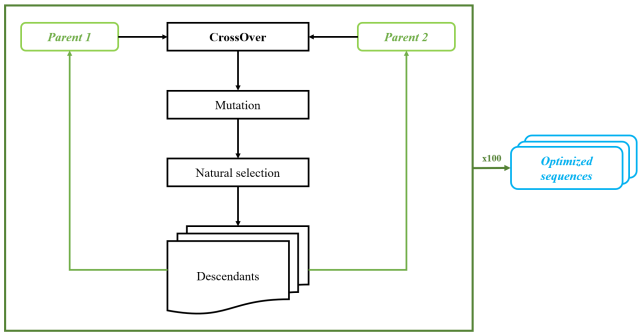

Figure 2. GA based on natural selection theory.

The second step is the CrossOver process. In this case, the sequences of peptides are considered as chromosomes, one chromosome of a parent will split into two parts of random lengths, each part will combine to another part from another splitted sequence. By doing so, each pair of parents will have 4 children in every crossover process.

The mutation step follows the CrossOver. The sequences representing the children have a chance to transform some of their amino acids to another type. To realize that we set a threshold 1/1000, then give each amino acid in a sequence a random probability (0 to 1). If this number is less than 1/1000, this amino acid will transform to other types of amino acid, for example from Cys to Arg. This step simulates the natural process of evolution and gives the children larger diversity, which helps us to find good hidden combinations.

After the mutation, the second generation of sequences will be chosen by natural selection, the most important property is the Bayesian Score, which decides the possibility of a sequence having children or not. Then to prevent the sequence from growing too long (long sequences tend to have higher score, but they are less stable), we set the length threshold as 25 amino acids, the sequences beyond this length will be cut off.

After natural selection, the second generation sequences will become parents and give birth to third generation sequences. This cycle will run several times, and the last generation will have higher average scores, these newly formed sequences will be the optimized anticancer peptides.

RESULTS

For the XGBoost model, first, the dataset with 2594 ACP sequences was imported. However, it was not possible to use all the samples because a lot of them were repeated or lacked values within the physicochemical characteristics. The data frame was cleaned of all those samples that could not be used. In the end, a total of 1053 peptides remained. Consequently, we split the data into 2 groups randomly: 75% to test the model and 25% to train it.

The algorithm started with a loss function of 0.56397 and after 25 iterations it was minimized until 0.26902. Following that, the model iterated 10 more times without changing this value, which tells us that this is the best loss function value that we could obtain with our training set. Briefly, after iterating our model, it obtained a final 73.098% probability of adequately predicting the stability of each peptide.

The *Feature importance* chart gives us the F score of each physicochemical descriptor to understand the exact importance of each feature. With these, it is possible to determine the probability of the peptides to be stable or not. In **Figure 3**, the Amphiphilicity Index is the most important descriptor with a precision score of 73, followed by Tilt Angle with 61. Likewise, the scores allowed the algorithm to construct a Decision Tree. In this structure, each of the nodes tests a different attribute of the peptide and assigns a path in the tree until finally predicting if it is Stable or Unstable. The order and frequency in which the tree asks each attribute of the peptide are determined by the score of each feature. In other words, the more an attribute is used to make key decisions in the tree, the greater its relative importance (or F score).

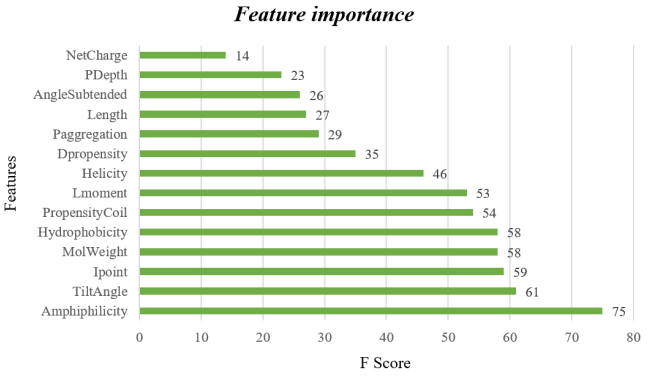

Figure 3. F score of physicochemical features (XGBoost).

Using the test set, we can calculate the performance of the model through the accuracy score. In **Figure 4**, from 171 Stable peptides, 157 (or more than 91%) were classified as Stable. And, for 93 Unstable peptides, 80 (or more than 86%) were classified in the correct label. As a result, the accuracy score for our test set is 89.77% meaning that our model is not overfitted to the train set Thereby, it is good enough to predict the stability of the peptides using physicochemical descriptors.

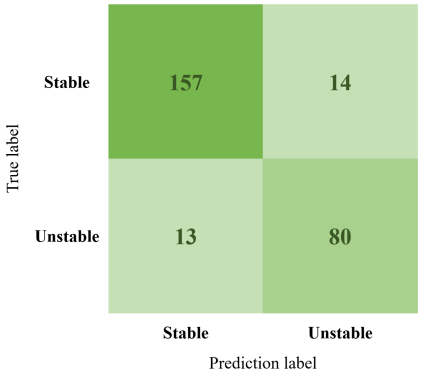

Figure 4. Confusion matrix (XGBoost).

For the Naïve Bayesian algorithm, it is possible to calculate the stability for any given sequence depending on its amino acids conformation. We established 2 data frames. The first for those peptides with an Instability index less than or equal to 40. The second for those greater than 40. From each one, 3 different libraries were created that contained the amount of amino acid residues, dipeptides, and 4-gap peptides. Finally, we had 9 libraries including 3 more from the general database (undivided). In such a manner, we identified the emergent descriptors for the different ACP classes. Nevertheless, not all descriptors are good features for classification. We used **Eq. 4** to calculate a normalized score for each descriptor in the stable and unstable libraries. The more positive the score of a feature, the more likely it is to appear in a Stable peptide and the more negative it's score, the likelihood to appear in an Unstable peptide increases. Additionally, to be able to see the distribution of normalized scores of the emergent features and its frequency of occurrence we plotted an histogram in **Figure 5**. From this, it was possible to obtain those emergent features that best describe stable peptides **(Table III)** and unstable peptides **(Table IV)**.

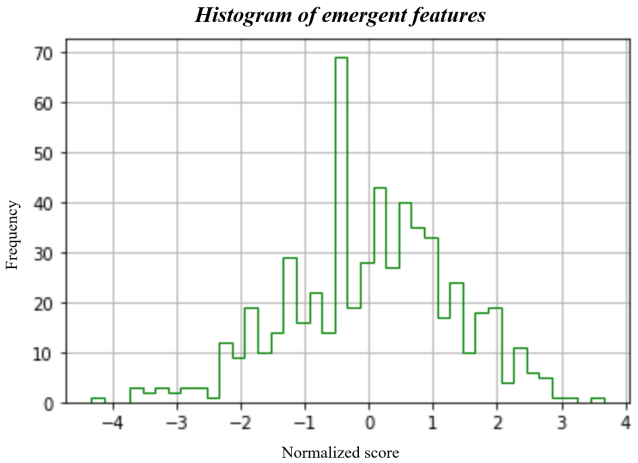

Figure 5. Frequencies of normalized scores for emergent features (Naïve Bayes).

	Emergent descriptor	Normalized score
	F----Q	3.657
	AI	3.185
	T----K	2.869
	VL	1.711
	(...)	(...)

TABLE III. Emergent features with high normalized scores.

	Emergent descriptor	Normalized score
	R----P	-4.339
	RW	-3.649
	CG	-2.989
	P----Y	-0.127
	(...)	(...)

TABLE IV. Emergent features with low normalized scores.

For the Naïve Bayesian scoring function with the emergent descriptors, as an example, we can analyze the sequence of one peptide: AAGKAKK. We can see in **Table V** some of the possible descriptors of this sequence and its probability of occurrence in the stable library. The amino acid Alanine (A) has a 33% probability of appearing in a stable sequence, the Lysine (K) has 75% chance and the dipeptide Lysine-Lysine (KK) has a 100% chance of appearing.

	Emergent descriptor	Probability to appear in Stable Peptide
	A	0.33
	K	0.75
	G	0.25
	KK	1

TABLE V. Example of a structural analysis for an ACP with a Naïve Bayesian approach.

The scoring equation created for this peptide can be calculated as the following formula:

$$score(AAGKAKK) = 3{\cdot}score('A') + 1{\cdot}score('G') + 3{\cdot}score('K') + 1{\cdot}score('KK') = 4.5 \quad (6)$$

We consider the peptide with a score higher than 0 should be stable. A scoring equation is created and calculated for each of the peptides in the training set of our algorithm. When all of the stability scores are calculated for our training set, then it is compared with the real Instability index obtained previously from our database. This makes it possible to estimate the quality of our model. Over 200 samples were used to test the model and it was achievable to obtain a confusion matrix (**Figure 6**) with an accuracy score of 67%.

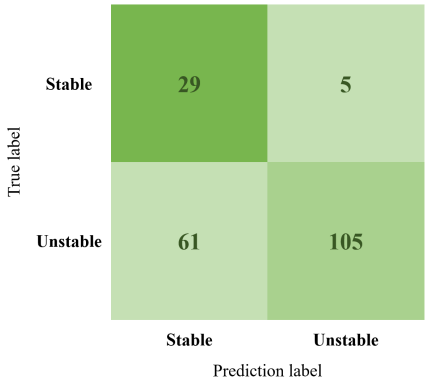

Figure 6. Confusion matrix (Naïve Bayes).

For the GA, after running it several rounds, the stability of newly born sequences has been improved remarkably. **Figure 7** represents a graph of the stability predicted vs. the Naïve Bayesian score found before (parent peptides in blue) and after (descendant peptides in red) the GA treatment. As we can see, the more optimization cycles with the GA, the smaller is the Instability Index of the descendants meaning that they are obtaining an improvement in their predicted stability. Furthermore, their Naïve Bayesian score is improving, going from a media of 7.2 in the first 10 cycles to

a media of 8.1 after 30 cycles. This means that the *de novo* peptides (or descendant peptides) have more parts found in the stability library than before. The 3 optimized sequences with the highest scores are shown in **Table VI**.

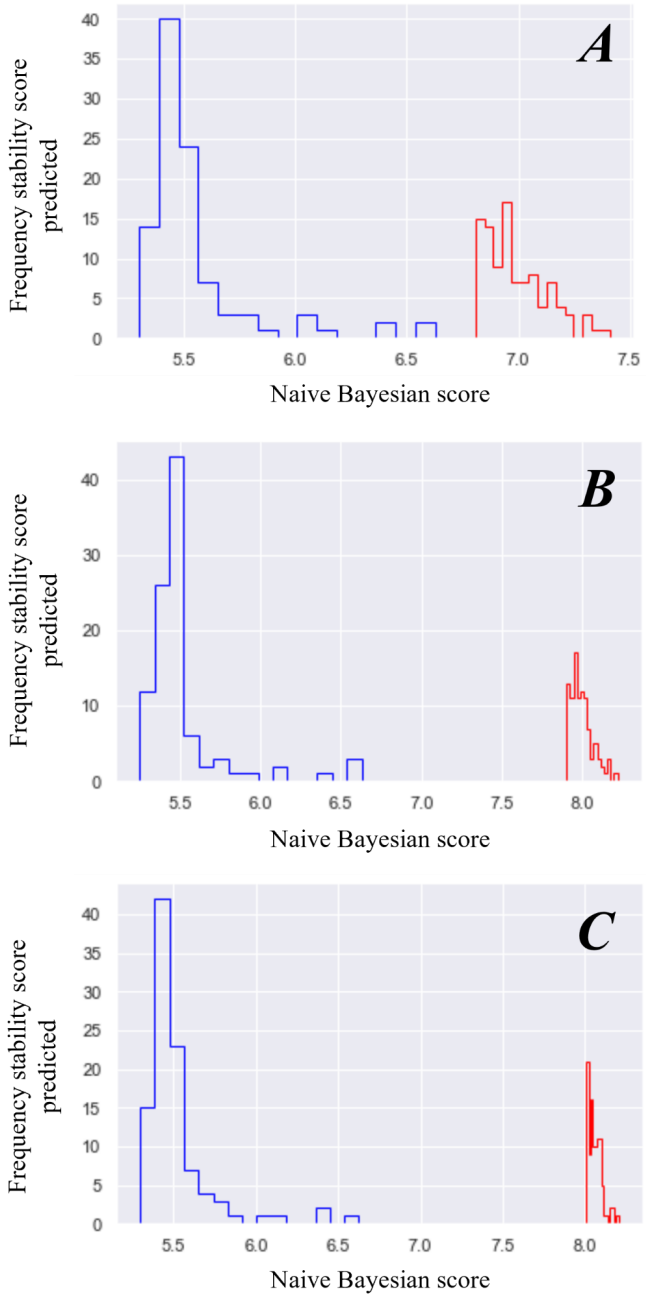

Figure 7. Graph of predicted stability score predicted and Naive Bayessian score of the first (blue) and last (red) population of best 100 de novo sequences generated after 10, 20 and 30 cycles of optimization procedure with GA. (**A**) 10 optimization cycles with GA (**B**) 20 optimization cycles with GA (**C**) 30 optimization cycles with GA.

	Optimized peptide sequences
	RSRQCVNVDGFLKNVGGFAGEISKIEKKIEEYIFEY
	DGWEEWEKKIQKIEELLKKAEEQLEKKGAIAVAKL
	DGWEEWEKKIQKIEELLKKAAEEQFKKLKQKK

TABLE VI. Optimized peptides after 5 cycles of GA.

DISCUSSION AND PERSPECTIVES

Taking the physicochemical characteristics as descriptors, a better supervised classification model is achieved. By making use of DBAASP and the BioPython package, features of great importance can be obtained to determine whether a peptide (ACP or other) will be Stable or Unstable. This, incorporated in the XGBoost algorithm, generates a better accuracy score (89.77%) compared to the algorithm based on Bayes' theorem. Physicochemical properties are values produced from the general conformation of the peptide. In other words, not only the probability that an amino acid or dipeptide is in a specific class is taken into account, but also its position and interaction with the other amino acid residues around it. This given that the properties have been obtained from various formulas and models previously validated both *in-silico* and *in-vitro* [18, 19].

After being dealt with by the Naïve Bayesian method, several descriptors for each peptide were obtained. Instead of screening the descriptors and selecting those that contribute a lot to the stable peptide, all of them were contained in our model, which is quite different from the QSAR process. As it is well known, the more descriptors a QSAR has, the more R^2 it will get, which means the model is probably overestimated. However, in our project, we use the log to calculate the score for each peptide. Thus, in our model, there is a penalty system and when the descriptor is not good enough, it won't make any contribution to the activity, and will decrease the probability of being stable. As a consequence, it will be even harder to get a negative or high score for peptides. Besides, even if we consider the bad descriptors as noise, as long as we have enough training tests, the average score of the noise will be around 0. Then, it won't have any influence on the final result. Nonetheless, the final Naïve Bayesian model does not have a good accuracy score (67%), but it is indispensable for developing emergent descriptor libraries. Which later serves for the execution of the GA. This approach is similar to that carried out by Guruprasad et al. [7], with the difference that our model considers the count of dipeptides and 4-gap peptides. Owing to the GA, we increased the average stability score of the peptides and got optimized peptides. However, there are some problems remaining to be solved. The first one is that the length of the *de novo* peptides has the tendency to be longer after each cycle, which means a criteria for the length is probably necessary in this method. Additionally, the code should be optimized to decrease the running time.

For future work, it is recommended to comprehend in greater depth the function of dipeptides and 4-gap peptides in protein structures. Thus, we can iterate the Naïve Bayesian classifier developed in the paper and make the GA more precise. Some laboratory experiments would help us to verify if the optimized peptides truly have higher activity. As a result, the experimental data can improve our model and make it a powerful tool to discover new ACPs.

CONCLUSION

This paper presents two supervised machine learning models to classify peptides with anticancer activity according to the Instability index. These two approaches are based on a Gradient Boosting model with a dataset that includes physicochemical properties (**Table I** and **II**), and on a Naïve Bayesian model with the creation of libraries with emergent descriptors. The XGBoost algorithm was chosen as the best prediction model due to the creation of a Decision True that considers diverse numerical features related to the behaviour of the peptide structure. The Naïve Bayesian algorithm focuses more on the sequence of peptides rather than their properties, which provides another way to predict the stability of peptides. Still, it requires more precision in the selection of its emergent descriptors to be an enhanced prediction model. However, for the optimization of the ACPs, a new Genetic Algorithm was developed based on a Bayesian score. With this, we simulate *in-silico* the natural mutation of organisms until obtaining the sequence with the best score.

ACKNOWLEDGEMENTS

We would like to thank our tutors, especially Dr. Jean-Yves TROSSET and Dr. Patrick GONZALES for their numerous advice during the whole process.

REFERENCES

[1] Y. Huan, Q. Kong, H. Mou and H. Yi, "Antimicrobial Peptides: Classification, Design, Application and Research Progress in Multiple Fields", Frontiers in Microbiology, vol. 11, 2020. Available: 10.3389/fmicb.2020.582779 [Accessed 23 December 2021].

[2] R. Kundu, “Cationic amphiphilic peptide: A nature-inspired synthetic antimicrobial peptide”, ChemMedChem. Available: 10.1002/cmdc.202000301 [Accessed: 22 December 2021]

[3] S. Gad, "QSAR", Encyclopedia of Toxicology, pp. 1-9, 2014. Available: 10.1016/b978-0-12-386454-3.00971-4 [Accessed 23 December 2021].

[4] G. Gogoladze et al., "dbaasp: database of antimicrobial activity and structure of peptides", FEMS Microbiology Letters, vol. 357, no. 1, pp. 63-68, 2014. Available: 10.1111/1574-6968.12489 [Accessed 23 December 2021].

[5] H. Nam et al., "Helicity Modulation Improves the Selectivity of Antimicrobial Peptoids", ACS Infectious Diseases, vol. 6, no. 10, pp. 2732-2744, 2020. Available: 10.1021/acsinfecdis.0c00356 [Accessed 23 December 2021].

[6] J. Cross, "MEDLINE, PubMed, PubMed Central, and the NLM", Editors' Bulletin, vol. 2, no. 1, pp. 1-5, 2006. Available: 10.1080/17521740701702115.

[7] K. Guruprasad, B. Reddy and M. Pandit, "Correlation between stability of a protein and its dipeptide composition: a novel approach for predicting in vivo stability of a protein from its primary sequence", "Protein Engineering, Design and Selection", vol. 4, no. 2, pp. 155-161, 1990. Available: 10.1093/protein/4.2.155 [Accessed 23 December 2021].

[8] D. Gamage, A. Gunaratne, G. Periyannan and T. Russell, "Applicability of Instability Index for In vitro Protein Stability Prediction", Protein & Peptide Letters, vol. 26, no. 5, pp. 339-347, 2019. Available: 10.2174/0929866526666190228144219 [Accessed 23 December 2021].

[9] Y. Jin et al., "Antimicrobial Activities and Structures of Two Linear Cationic Peptide Families with Various Amphipathic β-Sheet and α-Helical Potentials", Antimicrobial Agents and Chemotherapy, vol. 49, no. 12, pp. 4957-4964, 2005. Available: 10.1128/aac.49.12.4957-4964.2005 [Accessed 23 December 2021].

[10] "Classification Algorithm in Machine Learning - Javatpoint", www.javatpoint.com, 2021. [Online]. Available: https://www.javatpoint.com/classification-algorithm-in-machine-learning. [Accessed: 23- Dec- 2021].

[11] T. Chen and C. Guestrin, "XGBoost", Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. Available: 10.1145/2939672.2939785 [Accessed 23 December 2021].

[12] Gradient Boost Machine Learning|How Gradient boost work in Machine Learning. India: Unfold Data Science, 2020.

[13] Maths behind XGBoost|XGBoost algorithm explained with Data Step by Step. India: Unfold Data Science, 2020.

[14] "XGBoost Parameters — xgboost 1.5.1 documentation", Xgboost.readthedocs.io, 2021. [Online]. Available: https://xgboost.readthedocs.io/en/stable/parameter.html. [Accessed: 23- Dec- 2021].

[15] "1.9. Naive Bayes", scikit-learn, 2021. [Online]. Available: https://scikit-learn.org/stable/modules/naive_bayes.html. [Accessed: 23- Dec- 2021].

[16] M. Albadr, S. Tiun, M. Ayob and F. AL-Dhief, "Genetic Algorithm Based on Natural Selection Theory for Optimization Problems", Symmetry, vol. 12, no. 11, p. 1758, 2020. Available: 10.3390/sym12111758 [Accessed 23 December 2021].

[17] 高扬, 卫谆, 尹会生, 白话大数据与机器学习, 2016, 机器工业出版社, 329 pp..

[18] D. Eisenberg, R. Weiss and T. Terwilliger, "The helical hydrophobic moment: a measure of the amphiphilicity of a helix", Nature, vol. 299, no. 5881, pp. 371-374, 1982. Available: 10.1038/299371a0 [Accessed 24 December 2021].

[19] "Peptide calculator", Pepcalc.com, 2021. [Online]. Available: http://pepcalc.com/notes.php?all. [Accessed: 24 December 2021].