Journal of Science and Development Economics
Nam Can Tho University

Website: jsde.nctu.edu.vn

Predicting edible and toxic mushrooms with multi-layer perceptron method in streaming data

Nguyen Ngoc Pham[1], Phan Thi Xuan Trang[1], Tran Thi Thuy[2], Ngo Ho Anh Khoi[1]
[1]*Faculty of Information Technology, Nam Can Tho University*
[2]*Nam Can Tho University*

Received: November 10, 2023
Revised: November 30, 2023
Accepted: December 5, 2023

Keywords: *advanced machine learning, artificial intelligence, mushroom diagnosis, MLP classifier*

Từ khoá: *chẩn đoán nấm, học máy nâng cao, phân loại MLP, trí tuệ nhân tạo*

ABSTRACT

In today's rapidly advancing technological landscape, a significant portion of current technological developments is centered around the field of Artificial Intelligence (AI). Machine Learning, a subfield of AI, applies statistical and mathematical methods to enhance computer performance. AI has made substantial contributions to solving a wide range of problems over the past decade. Particularly, given the current context where distinguishing between different types of mushrooms is not well understood, especially to prevent the consumption of poisonous mushrooms that can have severe health consequences. The chosen algorithm for this project is the MLP (Multi-layer Perceptron) classifier, which has seen significant development in recent years and has widespread applications in various AI domains. This is why it has been selected as the foundation for the system in this project. The research topic aims to utilize this algorithm to construct a system that rapidly predicts whether a mushroom is edible or poisonous, facilitating practical applications. The application of artificial intelligence in the development of a system related to cognitive differentiation holds the promise of reducing the use of toxic mushrooms in daily life.

TÓM TẮT

Trong bối cảnh công nghệ phát triển nhanh chóng ngày nay, một phần đáng kể của sự phát triển công nghệ hiện nay tập trung vào lĩnh vực Trí tuệ nhân tạo (AI). Machine Learning, một lĩnh vực con của AI, áp dụng các phương pháp thống kê và toán học để nâng cao hiệu suất máy tính. AI đã có những đóng góp đáng kể *trong việc giải quyết nhiều vấn đề trong thập kỷ qua. Đặc biệt, trong bối cảnh hiện nay, việc phân biệt giữa các loại nấm khác nhau chưa được hiểu rõ, đặc biệt là để ngăn chặn việc tiêu thụ nấm độc có thể gây hậu quả nghiêm trọng cho sức khỏe. Thuật toán được chọn cho dự án này là thuật toán phân loại MLP (Multi-layer Perceptron), thuật toán này đã có sự phát triển đáng kể trong những năm gần đây và có ứng dụng rộng rãi trong nhiều lĩnh vực AI khác nhau. Đây là lý do tại sao nó được chọn làm nền tảng cho hệ thống trong dự án này. Đề tài nghiên cứu nhằm mục đích sử dụng thuật toán này để xây dựng một hệ thống dự đoán nhanh chóng nấm ăn được hay nấm độc, tạo điều kiện thuận lợi cho các ứng dụng thực tế. Việc ứng dụng trí tuệ nhân tạo vào việc phát triển hệ thống liên quan đến sự khác biệt về nhận thức hứa hẹn sẽ giảm việc sử dụng nấm độc trong cuộc sống hàng ngày.*

1. INTRODUCTION

In recent years, the cultivation of edible mushrooms has experienced robust growth in various regions across my country, becoming a relatively significant sector within agricultural production. Due to its high-profit potential, mushroom cultivation has emerged as a rich source of nutrition for humans, particularly in the case of certain medicinal mushroom varieties. Furthermore, the cultivation of straw mushrooms accelerates the process of agricultural and forestry waste decomposition, contributing to environmental cleanliness. Moreover, the practice of growing edible mushrooms enhances the value per unit of cultivated land area, generating employment opportunities and playing a role in adjusting the structure of the agricultural and rural economy. It also provides valuable export products, augmenting income for farmers and improving their livelihoods. Additionally, it aids in rural development by contributing to the overall economic transformation and providing sustainable growth. The cultivation of mushrooms not only offers economic benefits but also plays a pivotal role in waste management and ecological balance. It signifies a harmonious interplay between agriculture and the environment, showcasing how modern agricultural practices can positively impact both human well-being and the ecosystem. This shift towards mushroom cultivation exemplifies the dynamic evolution of agricultural practices and the recognition of mushrooms as a valuable resource, fostering socio-economic development while promoting a sustainable and ecologically conscious approach to farming. In the present global context, an exhaustive examination encompassing 2,786 mushroom species from 99 countries was conducted. This comprehensive review drew upon 9,783 case reports sourced from over 1,100 references. The analysis yielded valuable insights, identifying 2,189 mushroom species as edible, with a safe consumption status confirmed for 2,006 of them. Furthermore, 183 species were identified as requiring some form of pre-treatment before safe consumption, or they were associated with allergic reactions in certain individuals. Additionally, there were 471 species for which edibility remained uncertain due to incomplete or missing consumption data, and 76 species whose edibility and toxicity status remained unconfirmed, with varying opinions on their safety. These findings underscore the critical importance of exercising caution when foraging for mushrooms and highlight the necessity for robust policies and guidelines to ensure the safe utilization of mushrooms on a global scale. It's worth noting that many regions across the world have implemented management and control measures to address mushroom utilization in light of these findings (Li, 2021) [1]. The situation in Vietnam is characteristic of a country with a rich diversity of mushroom species. However, approximately 200 mushroom species in Vietnam are considered toxic and potentially hazardous to human health. Some common poisonous mushroom types in Vietnam include Thanh Ba toxic mushroom, water caltrop toxic mushroom, and Amanita toxic mushroom. Toxic mushrooms in Vietnam are typically found during the rainy season, when the weather is humid. In many cases, Vietnamese individuals have experienced poisoning after consuming mushrooms of uncertain origin or improperly prepared ones (Nguyen Thi Thu Ha, 2016) [2]. In the research of Tutuncu updated in 2022 (Tutuncu, 2022) [3], based on 22 features within the Mushroom dataset and four different machine learning algorithms, the outcomes of applying these algorithms to the mushroom dataset are compared. Models were constructed to classify mushrooms into edible and poisonous categories. The success rates of these models were obtained from Naive Bayes, MLP classifier, Support Vector Machine, and AdaBoost algorithms, with success rates of 90.99%, 98.82%, 99.98%, and 100% respectively. Upon further examination of these results, considering external mushroom characteristics, it was determined whether a mushroom is edible or poisonous with 100% accuracy using the AdaBoost model. This study highlights the significant capabilities of machine learning algorithms in accurately classifying mushrooms based on their features, contributing to the vital task of distinguishing between edible and toxic varieties for human safety.

Based on the aforementioned article, a study was conducted to explore the application of modern Artificial Intelligence algorithms in solving a mushroom diagnosis dataset, with a specific focus on the MLP classifier (Multi-layer Perceptron). Hence, the main objective of the research was to utilize the classic MLP classifier algorithm combined with “sliding window” techniques capable of dynamically altering internal functions and updating over time. This approach aimed to address the mushroom classification problem, alongside additional user-friendly website offering comprehensive information to users about the content and operational procedures of the mushroom type prediction methodology.

In essence, this study integrated advanced Artificial Intelligence techniques, particularly the MLP classifier, to create a comprehensive approach for classifying mushroom types. The research journey encompassed theoretical groundwork, algorithmic exploration, and practical application development, with the ultimate aim of providing a user-friendly tool for predicting mushroom types.

2. MATERIALS AND METHODS

From the article "Edible and Poisonous Mushrooms Classification by Machine Learning Algorithms" the researchers utilized the Mushroom dataset available in the UC Irvine machine learning repository. This dataset comprises 22 features and was chosen for the study. In the process of sourcing data for the project, numerous datasets were considered. Among them, the "Mushroom Classification" dataset (UCI Machine Learning, 2016) [4] was selected due to its comprehensive specifications and high availability, featuring 23 columns and approximately 8000 records last updated six years ago. Other datasets included the "Mushroom Edibility Classification" dataset by Devzohaib (2022) [5] with 21 attributes and 61000 entries, last updated five months ago. Additionally, the "Secondary Mushroom Dataset Data Set" by Saxena (Shruti, 2022) [6] provided hierarchical data and was updated nine months ago. The "Mushroom Attributes" dataset (Pedersen, 2023) [7] was the most recent, updated on February 10, 2023. Furthermore, the "Mushroom Classification dataset" of Jha (2020) [8], featuring a dataset from two years prior, was also considered. All of these datasets were sourced from the Kangle repository. These diverse datasets served as the foundation for the research, enabling the researchers to comprehensively analyze and classify edible and poisonous mushrooms using various machine-learning algorithms, ultimately contributing to a deeper understanding of mushroom classification and safety.

The listed datasets are all relevant to predicting mushroom types; however, not all of them meet the requirements of the current research. Some datasets have inherent issues that render them unsuitable for this project. To be eligible for use in this study, a dataset must be numeric, have specific classifications, contain multiple fields for objective results, and be the most up-to-date. Among the datasets mentioned, only the "Mushroom Attributes" dataset fulfills these criteria. This dataset is a potential candidate due to its numeric nature, well-defined classifications, comprehensive attributes for objective analysis, and recent updates.

It's important to note that this dataset might be temporary in nature, as the mushroom landscape evolves and diversifies over time. This could lead to potential changes in the classification of the dataset, indicating an environment of instability. Consequently, attempting to augment the dataset by adding more classes may not be feasible using conventional machine learning algorithms, as traditional methods do not accommodate such dynamic changes. Hence, advanced algorithms capable of handling data in dynamic environments are required for this purpose. After transforming the Mushroom Attributes dataset from its raw form into a standardized format for system utilization, adhering to data input rules is essential when using the prediction system. According to the system's regulations, the predictive labels must precede the parameters, and the parameters must be placed afterwards. Following the data normalization process, a standardized dataset was obtained, comprising a total of 8124 records. This dataset includes parameters with the following format:

1:< cap-shape>2:<cap-surface> 3:<cap-color> 4:<bruises%3F> 5:< odor> 6:<gill-

attachment> 7:< gill-spacing> 8:< gill-size> 9:< gill-color> 10:< stalk-shape> 11:< stalk-root> 12:< stalk-surface-above-ring> 13:< stalk-surface-below-ring> 14:< stalk-color-above-ring> 15:<stalk-color-below-ring> 16:<veil-type> 17:<veil-color> 18:<ring-number> 19:<ring-type> 20:<spore-print-color> 21:<population> 22:< habitat> 23:<class>

This format ensures that the input data conforms to the system's requirements, facilitating accurate and efficient predictions while maintaining consistency and adherence to the specified input structure. These indices have been validated through experts in the field. Each index will have its own parameters within the database.

- Cap shape: It refers to the shape of the mushroom cap. (1: convex, 3: flat, 4: knobbed, 5: bell, 13: conical, 14: sunken)

- Cap surface: It denotes the surface texture of the mushroom cap. (3: fibrous, 8: grooves, 2: scaly, 1: smooth)

- Cap color: This represents the color of the mushroom cap. (24: brown, 5: buff, 13: cinnamon, 8: gray, 20: green, 10: pink, 16: purple, 25: red, 7: white, 2: yellow)

- Bruises? (0: no, 1: yes)

- Odor: It indicates the odor of the mushroom. (11: almond, 12: anise, 13: creosote, 2: fishy, 3: foul, 17: musty, 24: none, 10: pungent, 1: spicy)

- Gill attachment: It refers to the attachment of the gills. (11: attached, 18: descending, 3: free, 24: notched)

- Gill spacing: It represents the spacing between gills. (13: close, 7: crowded, 18: distant)

- Gill size: It describes the size of the gills. (5: broad, 24: narrow)

- Gill color: This specifies the color of the gills. (14: black, 24: brown, 5: buff, 21: chocolate, 8: gray, 20: green, 15: orange, 10: pink, 16: purple, 25: red, 7: white, 2: yellow)

- Stalk shape: It denotes the shape of the stalk. (25: enlarging, 9: tapering)

- Stalk root: This indicates the root of the stalk. (5: bulbous, 13: club, 16: cup, 25: equal, 20: rhizomorphs, 22: rooted)

- Stalk surface above ring: The surface of the stalk above the ring. (3: fibrous, 14: silky, 1: smooth, 2: scaly)

- Stalk surface below ring: The surface of the stalk below the ring. (3: fibrous, 14: silky, 1: smooth, 2: scaly)

- Stalk color above ring: The color of the stalk above the ring. (24: brown, 5: buff, 13: cinnamon, 8: gray, 15: orange, 10: pink, 25: red, 7: white, 2: yellow)

- Stalk color below ring: The color of the stalk below the ring. (24: brown, 5: buff, 13: cinnamon, 8: gray, 15: orange, 10: pink, 25: red, 7: white, 2: yellow)

- Veil type: It indicates the type of veil covering the mushroom. (10: partial, 16: universal)

- Veil color: The color of the veil covering the mushroom. (24: brown, 15: orange, 7: white, 2: yellow)

- Ring number: The number of rings on the mushroom. (24: none, 15: one, 9: two)

- Ring type: The type of ring on the mushroom. (13: cobwebby, 25: evanescent, 3: flaring, 12: large, 24: none, 10: pendant, 1: sheathing)

- Spore print color: The color of the mushroom's spore print. (14: black, 24: brown, 5: buff, 21: chocolate, 20: green, 15: orange, 16: purple, 7: white, 2: yellow)

- Population: The abundance of the mushroom. (11: abundant, 13: clustered, 24: numerous, 1: scattered, 19: several, 2: solitary)

- Habitat: The habitat where the mushroom is found. (8: grasses, 12: leaves, 10: meadows, 16: paths, 7: urban, 18: woods)

- Class: The classification of the mushroom. (10: edible, 25: poisonous)

The dataset used in this experiment consists of two parts: the training data and the testing data. The training data includes 5686 samples (which accounts for 70% of the original data), and the testing data contains 2438 samples (which accounts for 30% of the original data). The positions of these data samples will be shuffled in each experiment. Random shuffling will be performed both before training and after training. The experiment will be conducted using a batch learning approach with a batch size of 5686. This means that the system will execute 5686 batches, where each batch contains approximately 129 data samples.

All the achieved results are based on Balanced Accuracy, a metric that can be used to evaluate the performance of a binary classifier. It is particularly useful when classes are imbalanced, meaning one of the two classes appears much more frequently than the other. Using Balanced Accuracy is much more intricate than traditional accuracy. Traditional accuracy is a straightforward calculation of the percentage of a data group based on the total available data. It initially works well, but when data is heavily skewed (e.g., 1 data point in class A, 999 data points in class B), the accuracy's correctness is compromised. To address this issue, a computation method has been devised that calculates the percentage based on true negatives, true positives, false negatives, and false positives. With these parameters in place, the formula for Balanced Accuracy can be applied to calculate the most accurate and optimal percentage. Balanced Accuracy takes into account these factors to provide a more comprehensive and reliable evaluation of a classifier's performance, especially when dealing with imbalanced datasets where traditional accuracy may not be a suitable measure.

The Balanced Accuracy formula used in these experiments is:

$$\text{Balanced accuracy} = (\text{Sensitivity}+\text{Specificity})/2$$

The experimental dataset consists of two parts: one for training and the other for testing. The training dataset contains 5686 rows of data, and the testing dataset contains 2438 rows of data. This dataset will vary in each experiment, but the data itself is shuffled and preserved without any changes. The model used in the experiment follows a batch data approach. This means that the model utilizes the dataset in batches. The original dataset is divided into smaller groups of 50 steps each, where each batch contains 129 data points. This batch size is chosen to strike a balance between being manageable and informative for the experimentation process.

3. RESULTS AND DISCUSSION

Currently, foundational databases are facing challenges due to their lack of adaptability over time. This issue arises because these databases are trained using classical algorithms, which involve a one-time training process and require relearning from scratch whenever new data arrives. For instance, if data set 1 is trained to create a model, when new data set 2 arrives, data set 1 needs to be relearned from the beginning alongside data set 2 to generate a new model. However, in the modern context, real-world environments experience continuous changes in data over time. Training must be carried out continuously in real-time, necessitating the constant updating of predictive models. This underscores the importance of learning from data in dynamically evolving environments. Therefore, the experimental approach must involve continuous learning methods within non-stable environments to effectively adapt to the changing data landscape. Several methods have been applied to transform classical algorithms into Continuous Learning methods, replacing them with the sliding window approach to advance traditional machine learning techniques. The description of the Sliding Windows approach is as follows:

Considering the recent development of concepts in an ever-changing training data environment, this approach involves a temporal window of fixed size (based on time intervals or the number of data points). This method can either reclassify the "group" type (based on data selected within the temporary window) or update models if online learning allows for it. In this case, the process of "forgetting" (as mentioned earlier) is automatically managed through this learning method. Typically, this method comprises three steps: Detect concept drift by using statistical tests on different windows, If an observed change occurs, select representative and recent data to adjust the models, Update the models. The window size is predetermined by the user. The key point of these methods lies in determining the window size. Most methods use a fixed-size window that is configured for each real-world problem. This allows classical algorithms to be used in dynamic environments but lacks the characteristics of incremental learning (not reusing stored data, only using the model to improve). Therefore, the historical section of the following algorithms focuses solely on presenting Incremental Learning algorithms, which have been researched and developed in recent years. By applying artificial intelligence methods, specifically algorithms like MLP classifier combined with the Sliding Window approach, a comparative analysis of these three algorithms' results can be achieved. Utilizing a chart to compare the average outcomes of these algorithms, this method ensures the most fairness in assessing the data's credibility when comparing results across different algorithms. The experimental average results of the algorithms are depicted in the chart below (Figure 2).

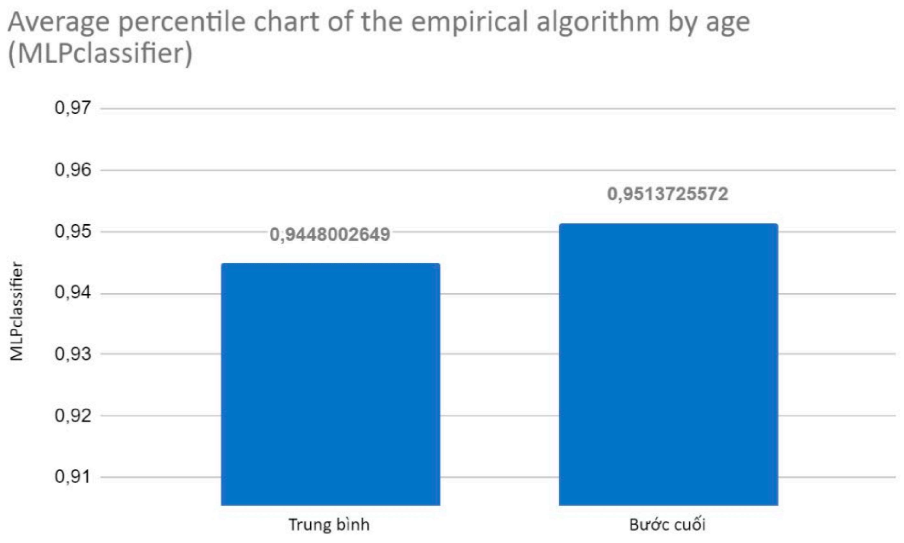

Figure 1. Average percentile chart of the empirical algorithm by age (MLP classifier)

Based on the data on the chart, we can analyze the average ratio and the final step's relatively good ratio of MLP classifier as follows:

The performance of MLP classifier is quite stable, consistently achieving above 91%: The data indicates that MLP classifier maintains stable and noteworthy performance with an average accuracy of over 94% (precisely 94.480%), and the accuracy in the final stage is also above 95% (precisely 95.137%). This is a significant advantage, demonstrating that MLP classifier has the capability to provide accurate predictions in most cases, making it suitable for addressing this stress prediction problem. In addition to calculating the algorithm's average results, another approach, such as analyzing the experimental model results by age, provides a more comprehensive and detailed view. This helps us assess the results visually and arrive at the most accurate conclusions. The results of the experimental model by age are represented in the chart below (Figure 3). This additional analysis based on age allows us to gain insights into how the model's performance varies across different age groups. It provides a more nuanced understanding of the model's strengths and weaknesses and helps in making informed decisions about its applicability and performance in different scenarios.

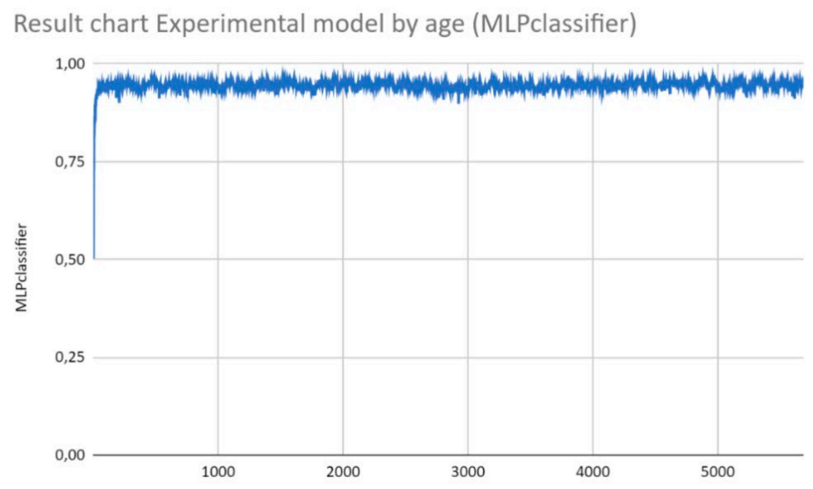

Figure 2. Result chart Experimental model by age (MLP classifier)

Considering the chart, we observe that the MLP classifier algorithm starts with a relatively low point of around 50.196% and gradually increases in the first 5 to 10 steps before stabilizing. Analyzing the chart, it's evident that MLP classifier exhibits relative stability. However, the highest accuracy achieved by MLP classifier can reach up to 96.614%, which is quite high and can be compared favorably with many other algorithms. Specifically, in the range of steps from 443 to 525, as shown in the chart below (Figure 3).

This specific analysis within the range of steps from 443 to 525 highlights the algorithm's remarkable performance during this phase. The fact that MLP classifier achieves such a high accuracy rate is an encouraging result and underscores its effectiveness in handling the given problem. It demonstrates the potential of MLP classifier as a powerful tool for accurate predictions, particularly in the specified range of steps.

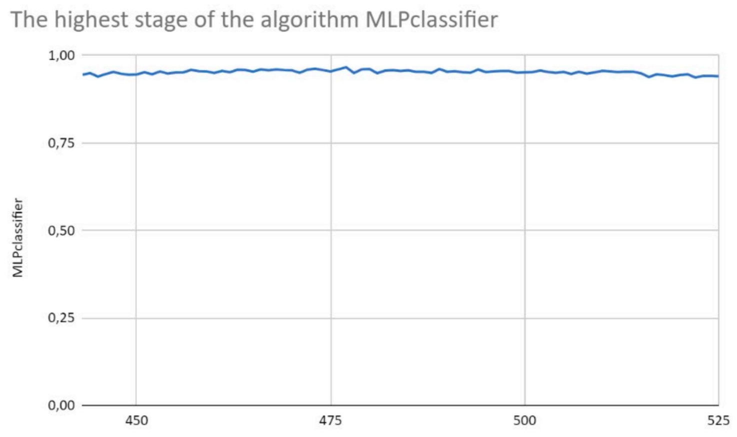

Figure 3. The highest stage of the algorithm MLP classifier (Batch 443-525)

From step 443 to step 525, the MLP classifier model demonstrated consistent and gradual performance improvement, achieving a notable accuracy rate consistently above 96.614%. This highlights MLP classifier's capability to handle non-continuous and missing value data effectively. This feature reduces the need for extensive data preprocessing, enabling the algorithm to operate efficiently across various types of data. MLP classifier consistently proves itself as a robust algorithm, even during periods where the results might not be optimal. It quickly stabilizes and maintains its performance, as depicted in the chart below (Figure 4, Figure 5). This behavior further reinforces the adaptability and reliability of MLP classifier, making it a valuable tool for various scenarios where data might exhibit variations or missing values. Its ability to maintain stability and achieve high accuracy rates, as showcased in the chart, is a strong testament to MLP classifier's effectiveness and potential for real-world applications.

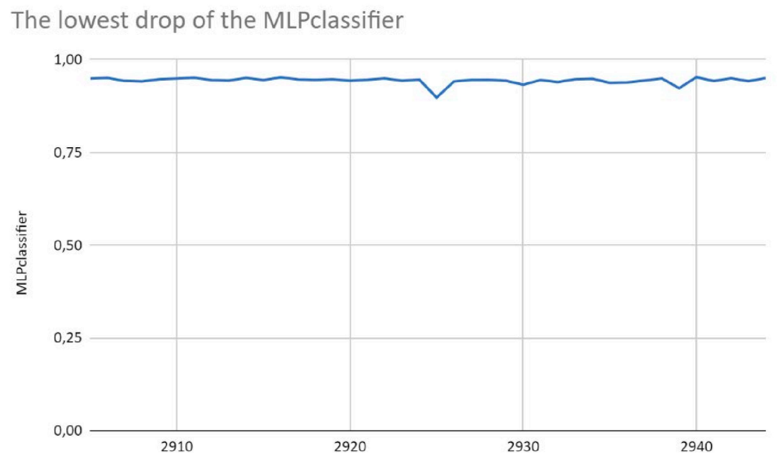

Figure 4. The lowest drop of the MLP classifier (Batch 2905 -2944)

During the MLP classifier's training phase, there was a decline in performance from step 2905 to step 2944, reaching a lowest point of 89.751% accuracy at step 2925. This could be attributed to the model struggling to learn from the data or encountering undesired variations. However, the model swiftly recovered afterward and began to improve its performance. From step 2906 to step 2920, the MLP classifier exhibited a relatively stable period following the performance dip. The accuracy rates during this phase didn't differ significantly, ranging from 94.44% to 95.34%. This indicates that the model maintained a relatively stable state.

In summary, the MLP classifier algorithm offers several advantages and maintains consistent performance with accuracy above 95%. This demonstrates that the MLP classifier can be a promising and valuable choice for various prediction and classification tasks. However, it's important to note that each algorithm has its own strengths and weaknesses, and the choice of algorithm depends on the specific requirements of the task and the characteristics of the data.

Installation:

Based on the final results presented in the previous section, the chosen algorithm to address the problem is the MLP classifier. The project will encompass various functional nodes, including prediction functionality, running classical algorithms, a list of processed models, system configuration, and user authentication. This project will be implemented within a website environment, divided into two main user roles: the Algorithm Installer (administrator or developer) and the Diagnostician (end user). These roles are illustrated in the use case diagram below:

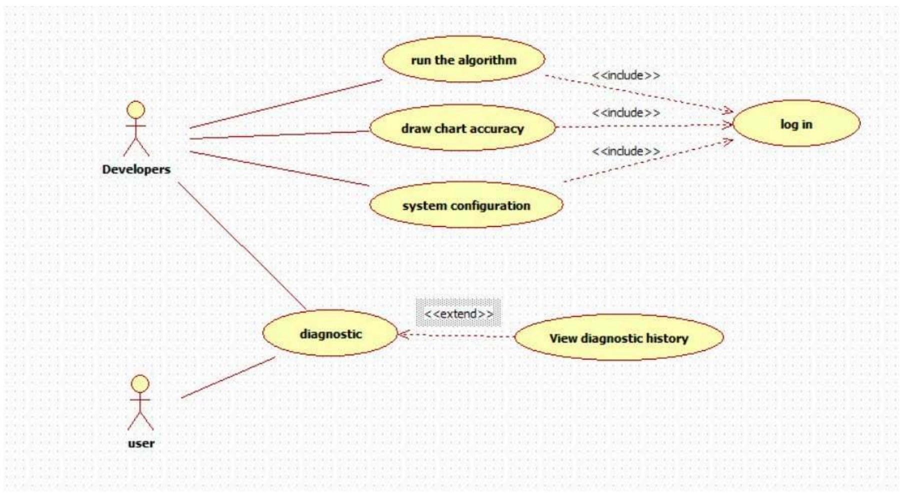

Figure 5. System use-case diagram

In order to run the software, the user's machine needs to have some Python libraries and Python version 3.9.9 installed. After extracting the contents, you will find a folder named "_SETUP_". Inside this folder, there will be a file named "python-3.9.9-amd64.exe" used to install Python 3.9.9, and a file named "inLib.bat" to install the necessary libraries required to run the software. Once the environment setup is completed, there will be a file named "Remove.bat" which is used to delete unnecessary files, including test files. This should only be used in two cases: right after extraction and installation, or to delete all previous test data. Finally, to run the program, you need to use the "Runserver.bat" file to launch the application. This file is configured to execute the command "py manage.py runserver", and the program will run on the default port "http://127.0.0.1:8000/". It's important to note that the user's computer must be connected to the internet at all times, and the minimum system requirements include Windows 10, 2GB RAM, and 10GB or more of available hard drive space to ensure smooth and stable performance. To use the software after successful installation, you can access the portal at "http://127.0.0.1:8000" to enter the main page of the system. On the main interface page, you will see a set of input fields used for making predictions. Below are the forms that have been built within the "Mushroom Prediction" system:

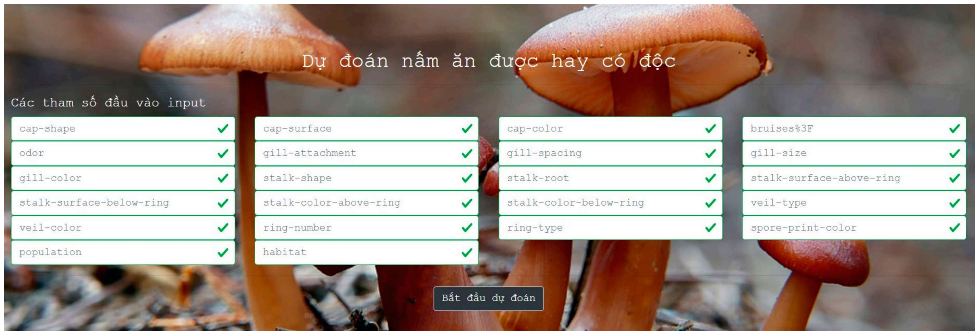

Figure 6. The main interface of the Mushroom Prediction system

4. CONCLUSION

Developing algorithms with multiple training options provides flexibility and high customization for the system. This allows users to experiment and choose the most suitable algorithm for specific tasks, leading to higher performance and reliability in prediction and classification. Developing a web interface in this direction makes the system user-friendly and convenient. The combination of machine learning and a website interface enables non-experts to interact easily with the system, input data, and receive classification results in a visual and clear manner. Addressing the issue of data variability by developing training algorithms in a model-based approach is a significant step in handling data fluctuations. The ability to update the model with new data helps the system maintain high and reliable performance over time. The potential for expansion and development of the topic is outlined up to the research phase, but important future directions have been proposed. This demonstrates the potential for further development and real-world application of the system. The suggested extensions include automating raw data processing, optimizing the model training process, improving the user interface, and deploying the application in practical settings. Using the MLP classifier algorithm based on the requirements for dynamic data is highly reasonable. These algorithms effectively handle changing data and help the system achieve optimal performance in various scenarios.

REFERENCES

[1] Li, H. (2021). Reviewing the world's edible mushroom species: A new evidence-based classification system, *Comprehensive Reviews in Food Science and Food Safety.* https://ift.onlinelibrary.wiley.com/doi/10.1111/1541-4337.12708.

[2] Nguyễn Thị Thu Hà (2016). *Giới thiệu về nấm.* Công ty TNHH Xuất Nhập Khẩu 2 Lúa. https://www.2lua.vn/article/gioi-thieu-ve-nam-y-nghia-va-vai-tro-cua-nam-5844d8b6e4951903508b4568.html

[3] Tutuncu, K. (2022). Edible and Poisonous Mushrooms Classification by Machine Learning Algorithms. *Mediterranean Conference on Embedded Computing (MECO),* https://ieeexplore.ieee.org/abstract/document/9797212

[4] UCI Machine Learning (2016). *Mushroom Classification.* www.kaggle.com/datasets/uciml/mushroom-classification

[5] Devzohaib (2022). *Mushroom Edibility Classification*. www.kaggle.com/datasets/devzohaib/mushroom-edibility-classification

[6] Shruti, S. (2022). *Secondary Mushroom Dataset Data Set*. www.kaggle.com/datasets/shrutisaxena/secondary-mushroom-dataset-data-set

[7] Pedersen, U.T. (2023). *Mushroom Attributes*, www.kaggle.com/datasets/ulrikthygepedersen/mushroom-attributes

[8] Jha, A.K. (2020). *Mushroom Classification Dataset.* www.kaggle.com/datasets/alokkumarjha/mushroom-classification-dataset