Introduction

Mushroom classification is a classic problem in machine learning, The goal is to predict whether a mushroom is edible or poisonous based on its physical characteristics.

The dataset used in this project is sourced from the UCI Machine Learning Repository and is publicly available on (Kaggle).

Dataset overview

This dataset is widely used in educational field due to its clean and well-organized structure, it contains 8,124 instances and 23 columns in total, 22 predictive features and 1 target column named "class", which indicates whether a mushroom is edible (e) or poisonous (p).
All the features are categorical, describing various observable characteristics of mushrooms, such as cap shape, odor, gill color, stalk surface, and more. Importantly, the dataset is complete with no missing values, making it ideal for practicing classification techniques without the need for extensive data cleaning.

Data preprocessing

The first step after loading the dataset was checking for any missing values. Once we confirmed the data was clean, we moved on to encoding the categorical features. Since all columns in the dataset are categorical, we used LabelEncoder to convert each column into numeric values suitable for machine learning models.

Next, we separated the features from the target column and split the data into training and testing sets, using a 70/30 ratio to ensure we could properly evaluate the model's performance on unseen data.

Given the large number of features, and recognizing that not all of them contribute equally to prediction accuracy, we performed a feature importance analysis using a Random Forest classifier, this allowed us to identify the most impactful features. Finally, we selected the top features based on this analysis to reduce dimensionality, which helps improve model generalization and efficiency.

Models training

After completing the preprocessing steps, we moved on to the model training phase. At this point, the dataset was clean, encoded, and reduced to the most informative features, making it well-prepared for training machine learning models.
We trained a variety of classification models to compare their performance on the mushroom dataset. The models used include:

- Decision Tree Classifier
- Random Forest Classifier
- K-Nearest Neighbors (KNN)
- Support Vector Machine (SVM)
- Naive Bayes
- Artificial Neural Network (ANN)
- Logistic Regression
  Each model was trained on the 70% training portion of the data, and evaluated on the 30% test portion. We measured performance using key classification metrics such as accuracy, precision, recall, and F1-score.

Here is a table representing the models' performance:

Models training and accuracy measurements

Model	Accuracy	Macro Precision	Macro Recall	Macro F1-Score
Decision Tree	100%	100%	100%	100%
Random Forest	100%	100%	100%	100%
KNN	100%	100%	100%	100%
SVM	98%	98%	98%	98%
Naive Bayes	85%	85%	84%	84%
Logistic Regression	90%	90%	90%	90%
ANN	100%	100%	100%	100%

Looking at the table, it's clear that random forest, decision tree, KNN, and ANN delivered outstanding results all of them achieved 100% accuracy, along with perfect scores in precision, recall, and F1-score, these models were able to perfectly learn the patterns in the dataset, thanks to how clean and well-structured the data is, and how strong some features like "odor" are in separating the classes.

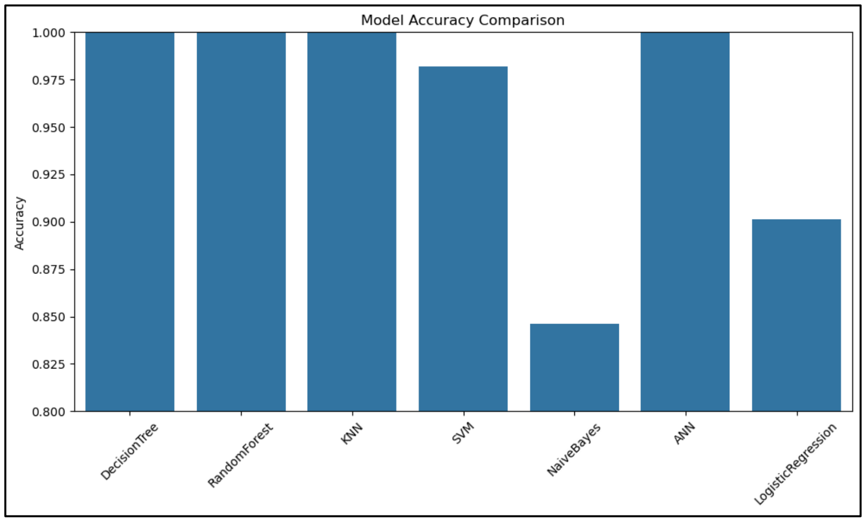
The SVM also did a great job with 98% accuracy, showing it can handle this type of data very well, even though it didn't hit 100%. Logistic Regression performed decently too, reaching 90% accuracy, which is pretty good for a linear model. It handled the classification reasonably well but may have struggled a bit with the more complex patterns in the data.

Naive Bayes, on the other hand, had the lowest performance, with 85% accuracy, that's not terrible, but it's noticeably behind the others this drop makes sense since Naive Bayes assumes that all features are independent of each other and in a dataset like this, that assumption doesn't really hold up.

Overall, the results show that the mushroom dataset is quite easy to classify with the right models, especially when we focus on the most important features.

Data visualization

Bar Plot (Bar Chart): is a type of chart that represents data using rectangular bars. The length of each bar is proportional to the value it represents.



In this visualization, the bar plot is showing the accuracy of different classification models. Each bar corresponds to one model, and its height indicates how well that model performed on the test data.

Confusion matrix: is a table used to evaluate the performance of a classification model. It shows the number of correct and incorrect predictions made by the model compared to the actual labels.

Conclusion

I picked the mushroom dataset because it’s a well-known benchmark used for teaching classification problems. It is clean, complete, and contains only categorical variables, which made it an excellent case for practicing encoding and model training. In real life, identifying poisonous mushrooms is a critical task, especially in food safety and foraging scenarios. Having a reliable model that can classify mushrooms based on physical characteristics can help prevent poisoning and even save lives. The dataset is perfectly structured with strong distinguishing features, particularly the "odor" attribute, which makes prediction highly effective. After preprocessing the data using label encoding and feature selection, we trained several classification models and evaluated them using accuracy, precision, recall, and F1-score. The Random Forest, Decision Tree, ANN, and KNN models all achieved perfect 100% performance across all metrics. This shows how powerful machine learning can be when data quality is high and features are informative. Among these, Random Forest stood out due to its ensemble nature and ability to avoid overfitting while maintaining excellent generalization. The worst-performing model was Naive Bayes, which scored 85%, mainly due to its unrealistic assumption of feature independence. Logistic Regression performed reasonably with 90%, but it lacked the complexity to handle non-linear feature interactions. Visualizations such as bar plots and confusion matrices helped illustrate model performance and highlighted the ease with which this dataset can be separated into two classes. One key takeaway is that not all features contribute equally feature importance analysis helped reduce dimensionality without sacrificing accuracy. Overall, this project showed that with the right models and preprocessing, even basic features can lead to high-impact classification outcomes.