CSE 422: ARTIFICIAL INTELLIGENCE

Fall 2024

Project Report

MUSHROOM CLASS DETECTOR

Student Information:

B M Rauf - 22201782

Mahdi Hasan - 22201760

Section: 16, Group : 16

Date : 06 January 2025

Submitted to:

Hasnat Jamil Bhuiyan

Maazin Munawar

Department of Computer Science and Engineering (CSE)

Table of Contents

Introduction

Mushroom classification is a critical task in the field of mycology and food safety. This project aims to classify mushrooms into categories based on their characteristics using machine learning models. The dataset contains various attributes of mushrooms, such as cap shape, surface, and color, which serve as predictors for the classification task.

The goal is to preprocess the data, scale the features, split the dataset, and train multiple models to identify the most effective one for the classification task. With the growing popularity of mushroom hunting, this model is highly relevant and can help prevent the consumption of toxic mushrooms, which can be life-threatening.

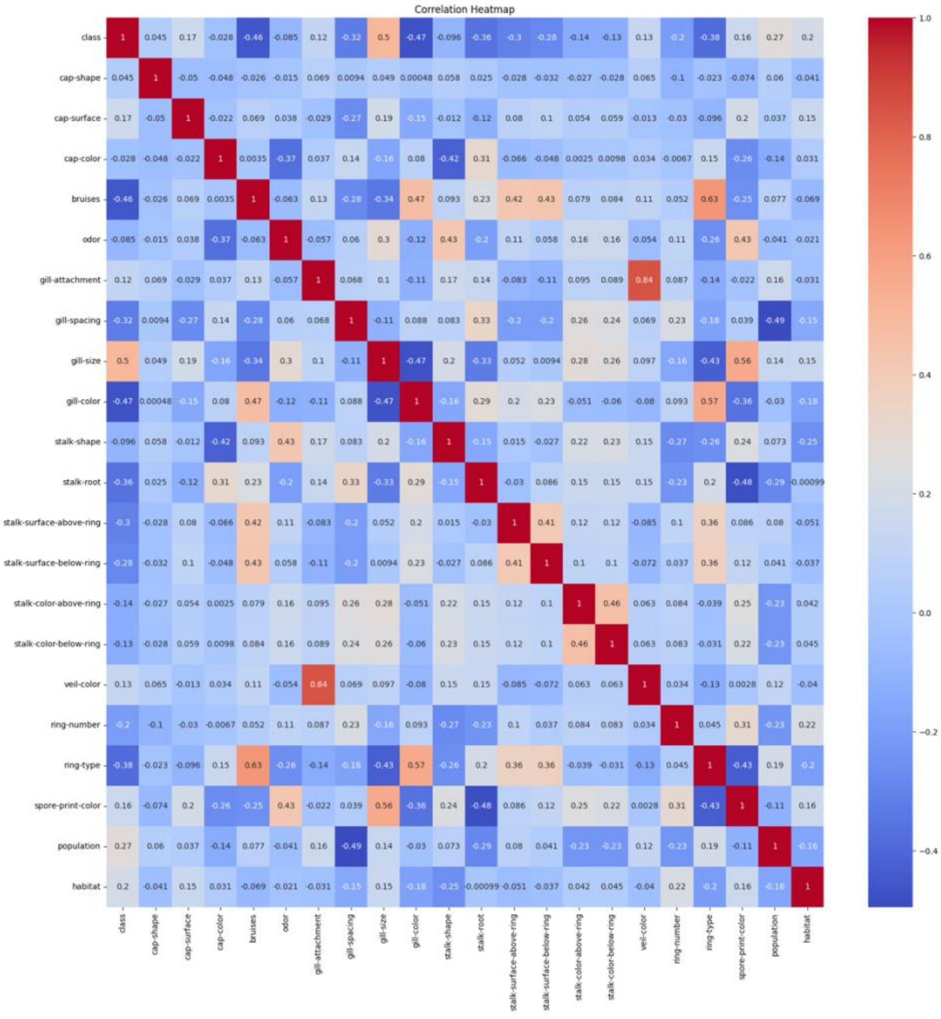
Dataset Description

- Source

  The dataset was originally contributed to the UCI Machine Learning Repository nearly 30 years ago.

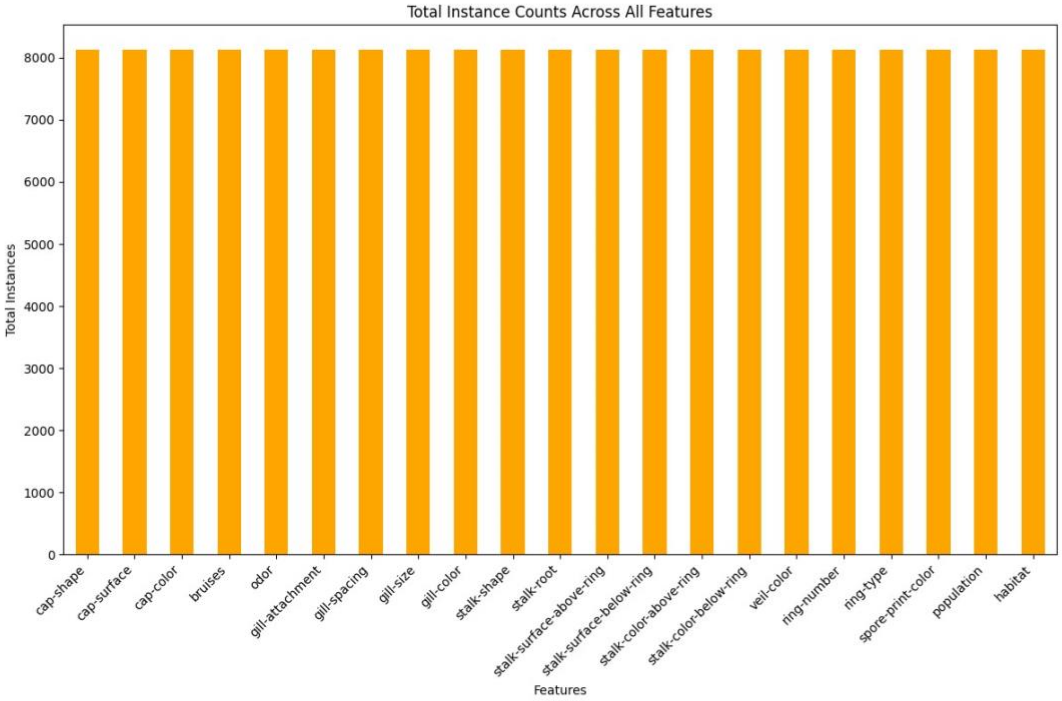
  - Link: https://www.kaggle.com/datasets/uciml/mushroom-classification
  - Reference : The dataset is based on descriptions from "The Audubon Society Field Guide to North American Mushrooms" (1981).

- Dataset Description
    - Number of Features: 22 features, including cap shape, odor, and habitat.
    - Problem Type: Classification problem, as the target is categorical (edible vs. poisonous).
    - Number of Data Points: 8,124 samples corresponding to 23 species of gilled mushrooms.
    - Feature Types: All features are categorical and were encoded into numerical values using LabelEncoder.
    - Correlation Analysis: A heatmap was generated to visualize correlations, highlighting highly correlated features that might impact the model.



- Balanced Dataset

For the output feature, all the unique classes have an equal number of instances.



Dataset Preprocessing

- Faults
  - NULL Values: In the actual dataset there was no null values, so we randomly inserted 5 – 10% null values.
  - Categorical Values: All features were categorical.
- Solutions
  - Dropped Columns: The 'veil-type' column was removed due to irrelevance, as it contained only one unique value. Handled Null values and fill categorical null values with mode . Still if a row contains any null value, that row will be dropped.

    - Encoding: Categorical variables were encoded using LabelEncoder to convert categorical values into numeric representations.

Feature Scaling

Feature scaling is essential for ensuring that the machine learning algorithms treat all features equally. The dataset was scaled using the MinMaxScaler from scikit-learn, which normalizes the values between 0 and 1. This step ensures the dataset is suitable for algorithms sensitive to magnitude differences in features.

Dataset Splitting

The dataset was split into training and testing subsets to evaluate model performance:

- Splitting Technique: `Stratified Random Sampling` was used to ensure that the distribution of the target variable in the training and testing subsets matches the original dataset.
- Training Set: 70% of the data.
- Testing Set: 30% of the data.

The splitting ensured an unbiased evaluation of the model's performance on unseen data, while maintaining the representativeness of the target variable distribution. The `random_state` parameter was used to ensure reproducibility of the splits.

Model Training & Testing

Multiple machine learning models were trained to classify mushrooms, including:

1. ## Decision Tree

   - Description:

     A decision tree is a supervised learning algorithm used for classification and regression tasks. It works by splitting the dataset into subsets based on the feature that provides the maximum information gain or minimizes impurity (e.g., Gini index or entropy). Each internal node represents a decision based on a feature, branches represent the outcomes, and leaf nodes represent the final prediction.

   - How It Works:
     - The algorithm starts with the root node and recursively splits the dataset based on features.
     - It continues splitting until a stopping criterion (e.g., maximum depth, minimum samples per leaf) is met.
     - The final tree structure is used to classify new data by traversing the nodes based on feature values.

2. ## Random Forest Classifier

   - Description:

     Random Forest is an ensemble learning method that builds multiple decision trees during training and aggregates their predictions (e.g., majority voting for classification) to improve accuracy and reduce overfitting. It is robust to noise and works well on large datasets.

- How It Works:
    - Multiple decision trees are created using random subsets of the data and features (bagging).
    - Each tree provides a classification result, and the most frequent class is chosen as the final prediction.
    - The randomness in data and feature selection ensures diversity among trees, reducing overfitting.

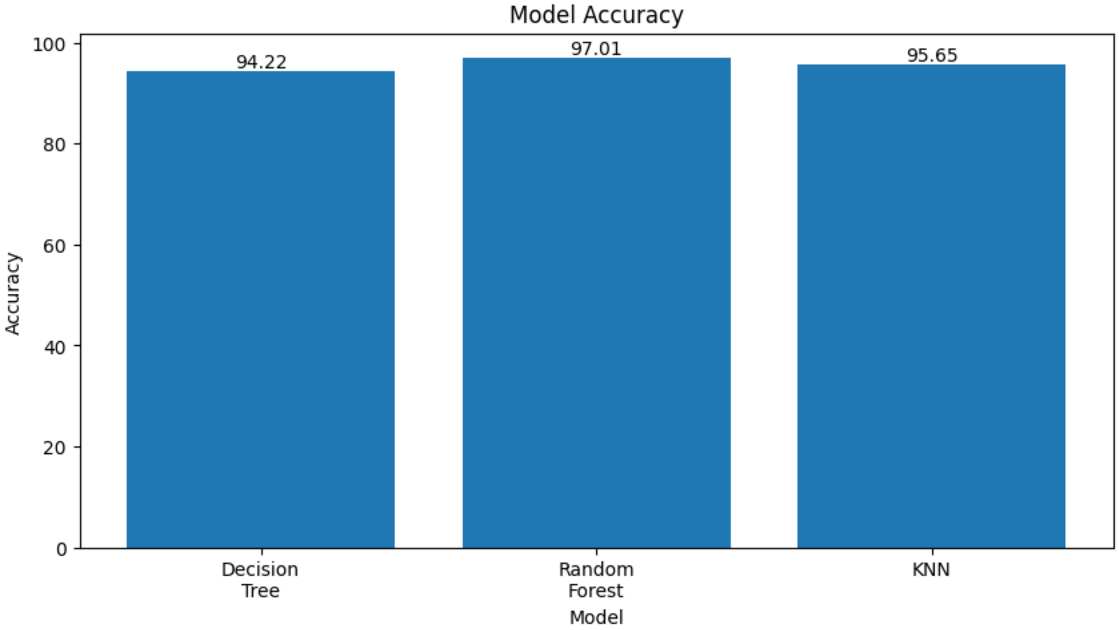
3. K-Nearest Neighbors (KNN) Classifier

- Description:

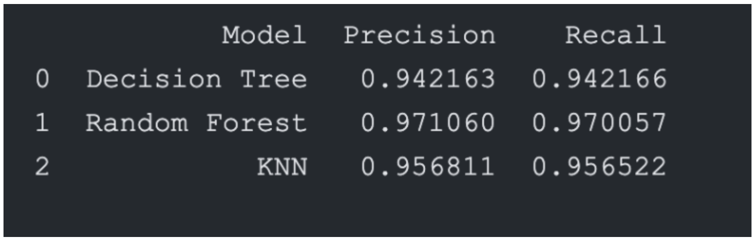
  KNN is a simple, non-parametric algorithm used for classification and regression. It makes predictions by finding the k-th nearest data points in the training set to a given test point, based on a distance metric (e.g., Euclidean distance).
- How It Works:
    - The algorithm calculates the distance between the test point and all training points.
    - It selects the k closest points and assigns the test point to the class most common among these neighbors.
    - The choice of k and the distance metric significantly affect the algorithm's performance.

Model Selection/Comparison Analysis

- Bar Chart: Showcased the prediction accuracy of all models.

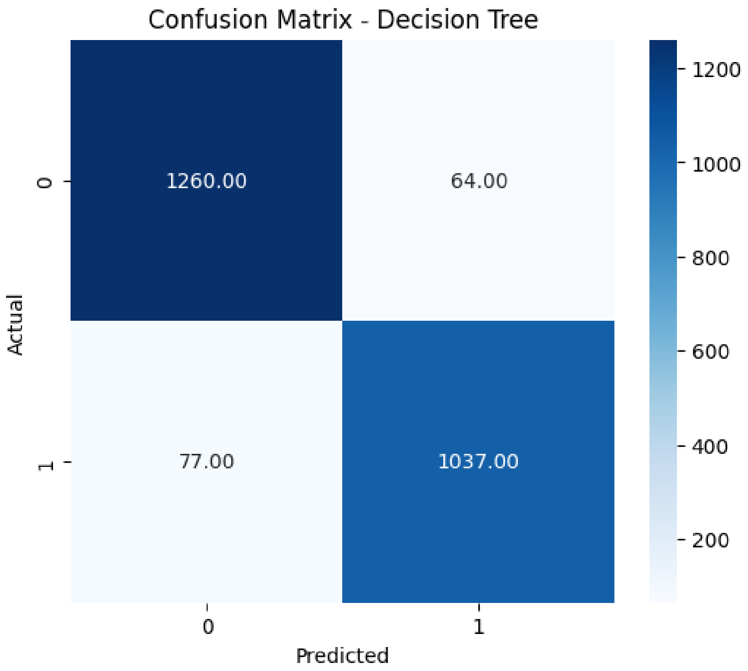


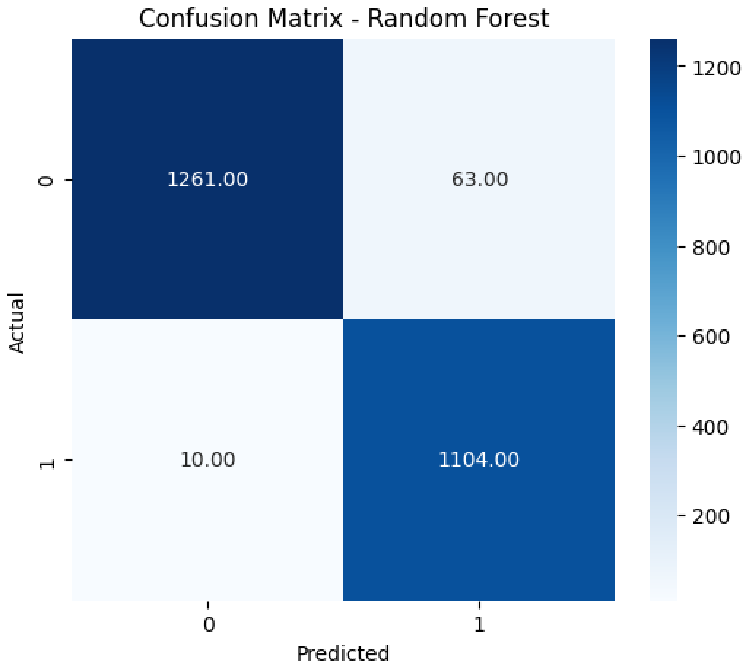
- Precision and Recall:
  - Precision: The proportion of correctly predicted positive instances out of all predicted positives. It measures accuracy in positive predictions.
  - Recall: The proportion of correctly predicted positive instances out of all actual positives. It measures the ability to find all positive instances.

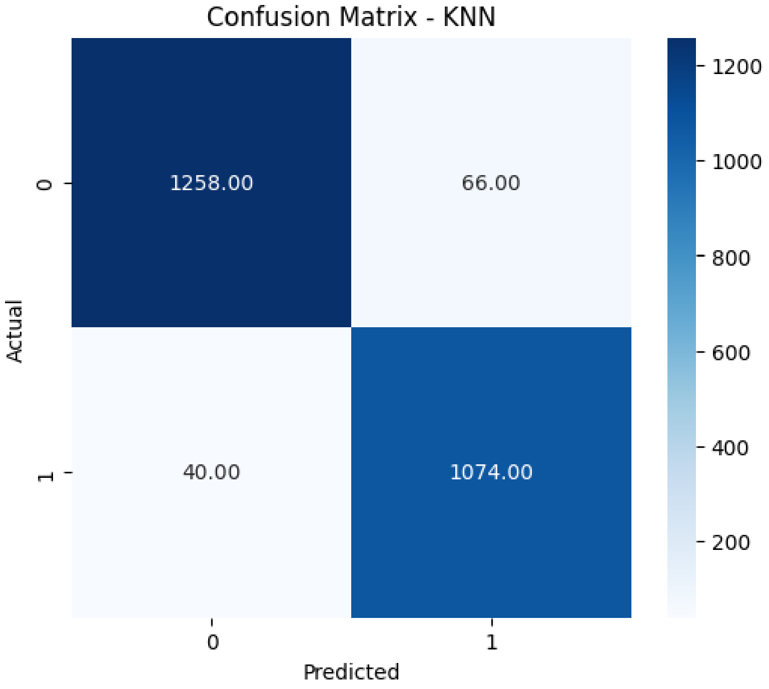


```
           Model  Precision    Recall
0  Decision Tree   0.942163  0.942166
1  Random Forest   0.971060  0.970057
2            KNN   0.956811  0.956522
```

- Confusion Matrix: Analyzed for each model to evaluate classification performance in detail.


Confusion Matrix - Decision Tree


Confusion Matrix - Random Forest



Conclusion

The Mushroom Class Detector project successfully used machine learning to classify mushrooms as edible or poisonous. Through proper preprocessing, scaling, and splitting of the dataset, the models—Decision Tree, Random Forest, and K-Nearest Neighbors—achieved 94.22%, 97.01%, and 95.65% accuracy respectively. Metrics like precision, recall, and confusion matrix confirmed their effectiveness. This project shows how machine learning can be a reliable tool to improve food safety and prevent the consumption of harmful mushrooms.