CSE422 Lab Project Report: Mushroom Dataset Analysis

Name:

Anit Paul(24141258).

Argha Das(24341220).

Submitted To:

Asif Shahriar

Mashiyat Mahjabin Prapty

1. Introduction

This project aims to analyze a mushroom dataset to classify mushrooms as either edible or poisonous based on their physical characteristics. The motivation behind this project is to build a predictive model that can accurately identify poisonous mushrooms, which can be crucial for safety. The problem it's aiming to solve is the classification of mushrooms, which is a common task in machine learning and can have significant real-world implications for foragers and consumers.

2. Dataset Description

Dataset Description

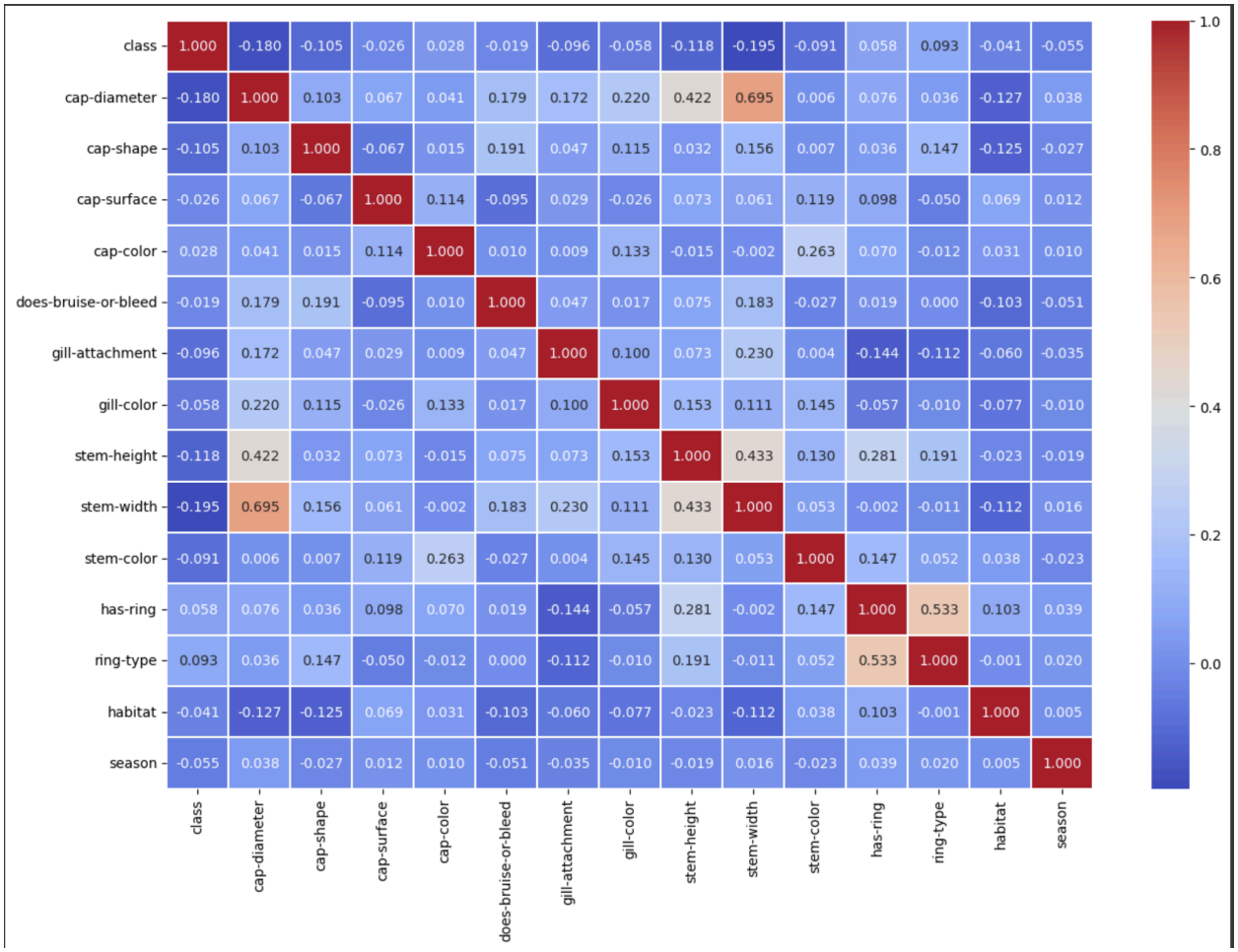
- **Dataset Source**: The dataset used is mushroom_dataset.csv. https://drive.google.com/file/d/19i3IoW38KBgMF96Rcnpu2ZUoO_UBwcdX/view
- **Number of Features**: The dataset initially contains 21 columns (features). After pre-processing (dropping columns with high null values), the number of features used for modeling is 15.
- **Number of Data Points**: The dataset has 61,069 entries (data points) initially. After removing duplicates, 60,923 entries remain.
- **Problem Type**: This is a **classification problem** because the target variable, 'class', aims to categorize mushrooms into distinct classes: edible ('e') or poisonous ('p').
- **Feature Types**:
    - **Quantitative (Numerical)**: cap-diameter, stem-height, stem-width.
    - **Categorical**: The remaining features, including 'class', 'cap-shape', 'cap-surface', 'cap-color', 'does-bruise-or-bleed', 'gill-attachment', 'gill-color', 'stem-color', 'has-ring', 'ring-type', 'habitat', and 'season', are initially objects/categorical. These are later label encoded for model training.

Correlation of Features

The notebook performs label encoding on categorical features to convert them into numerical representations before potentially calculating correlations or training models that require numerical input. A heatmap of feature correlations is generated using seaborn after this encoding and scaling process.

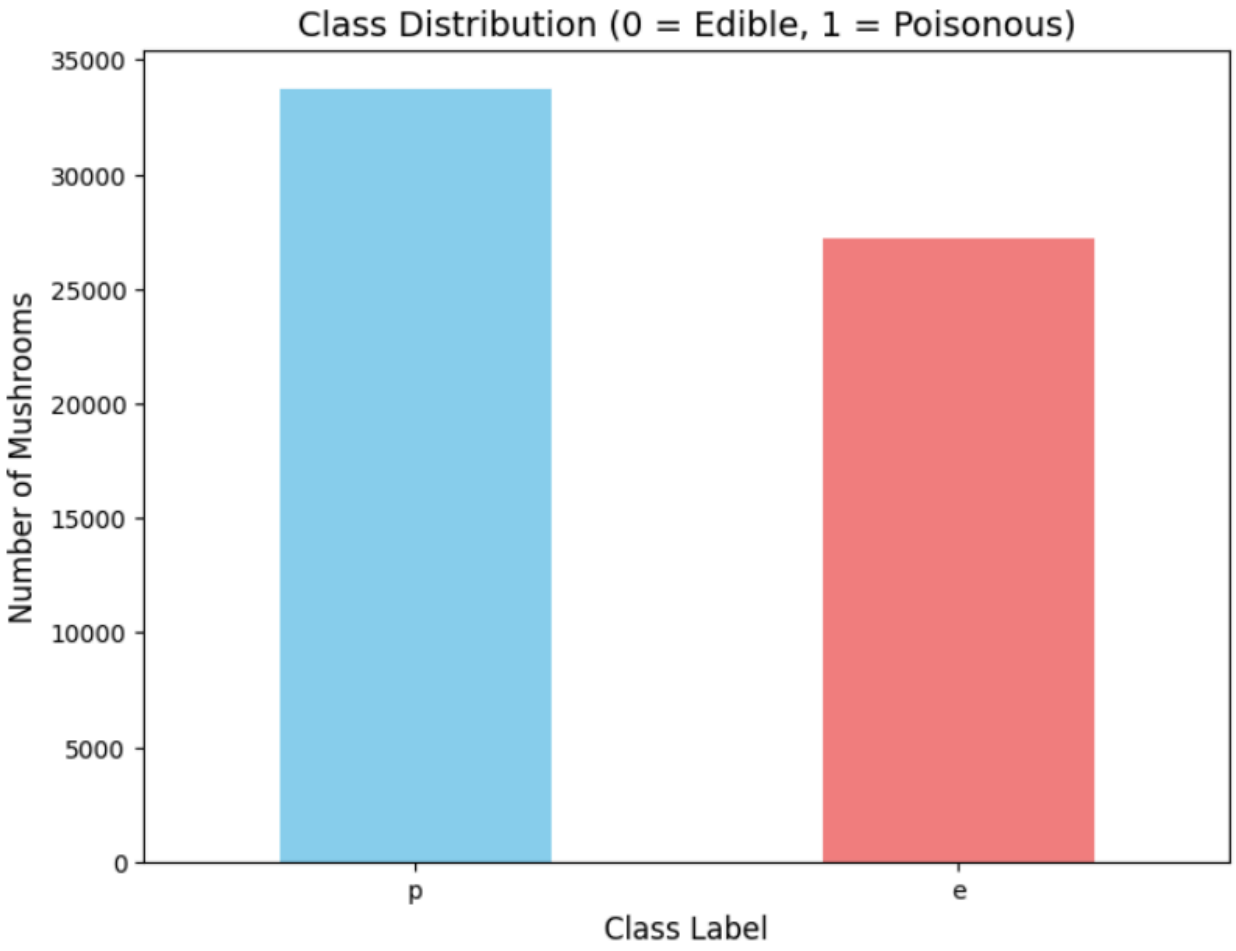
- **Heatmap of Feature Correlations**
    - This heatmap visually represents the correlation coefficients between all pairs of features in the pre-processed dataset. The color intensity indicates the strength and direction of the correlation.
    - **Understanding from Correlation Test**: By examining the heatmap, one can understand which features have strong positive or negative correlations with each other and with the target variable ('class'). For example, features highly correlated with the 'class' variable are likely to be important predictors for classifying mushrooms.

	class	cap-diameter	cap-shape	cap-surface	cap-color	does-bruise-or-bleed	gill-attachment	gill-color	stem-height	stem-width	stem-color	has-ring	ring-type	habitat	season
class	1.000000	-0.180027	-0.105437	-0.026357	0.028259	-0.018929	-0.095828	-0.058439	-0.118079	-0.195303	-0.091048	0.058480	0.093088	-0.040917	-0.055143
cap-diameter	-0.180027	1.000000	0.103308	0.067314	0.040796	0.178872	0.172210	0.220467	0.421718	0.695047	0.005678	0.076451	0.035590	-0.126680	0.038099
cap-shape	-0.105437	0.103308	1.000000	-0.067492	0.015317	0.191394	0.047063	0.114667	0.032410	0.156399	0.006814	0.035628	0.146625	-0.125380	-0.026740
cap-surface	-0.026357	0.067314	-0.067492	1.000000	0.113804	-0.095043	0.028756	-0.025839	0.073466	0.060818	0.119182	0.098034	-0.050464	0.069077	0.011666
cap-color	0.028259	0.040796	0.015317	0.113804	1.000000	0.009770	0.009104	0.133122	-0.014732	-0.001833	0.263227	0.069970	-0.012218	0.030795	0.009617
oes-bruise-or-bleed	-0.018929	0.178872	0.191394	-0.095043	0.009770	1.000000	0.046568	0.017397	0.074825	0.182553	-0.027221	0.018790	0.000356	-0.103344	-0.050738
gill-attachment	-0.095828	0.172210	0.047063	0.028756	0.009104	0.046568	1.000000	0.099768	0.072960	0.230488	0.003589	-0.143989	-0.112033	-0.059993	-0.034916
gill-color	-0.058439	0.220467	0.114667	-0.025839	0.133122	0.017397	0.099768	1.000000	0.153005	0.111064	0.145376	-0.056671	-0.009918	-0.077377	-0.009767
stem-height	-0.118079	0.421718	0.032410	0.073466	-0.014732	0.074825	0.072960	0.153005	1.000000	0.433214	0.130427	0.280723	0.191361	-0.023065	-0.018665
stem-width	-0.195303	0.695047	0.156399	0.060818	-0.001833	0.182553	0.230488	0.111064	0.433214	1.000000	0.052755	-0.002101	-0.010869	-0.111615	0.015511
stem-color	-0.091048	0.005678	0.006814	0.119182	0.263227	-0.027221	0.003589	0.145376	0.130427	0.052755	1.000000	0.146929	0.052484	0.037773	-0.023495
has-ring	0.058480	0.076451	0.035628	0.098034	0.069970	0.018790	-0.143989	-0.056671	0.280723	-0.002101	0.146929	1.000000	0.533020	0.102714	0.039280
ring-type	0.093088	0.035590	0.146625	-0.050464	-0.012218	0.000356	-0.112033	-0.009918	0.191361	-0.010869	0.052484	0.533020	1.000000	-0.001424	0.020347
habitat	-0.040917	-0.126680	-0.125380	0.069077	0.030795	-0.103344	-0.059993	-0.077377	-0.023065	-0.111615	0.037773	0.102714	-0.001424	1.000000	0.004980
season	-0.055143	0.038099	-0.026740	0.011666	0.009617	-0.050738	-0.034916	-0.009767	-0.018665	0.015511	-0.023495	0.039280	0.020347	0.004980	1.000000



Imbalanced Dataset

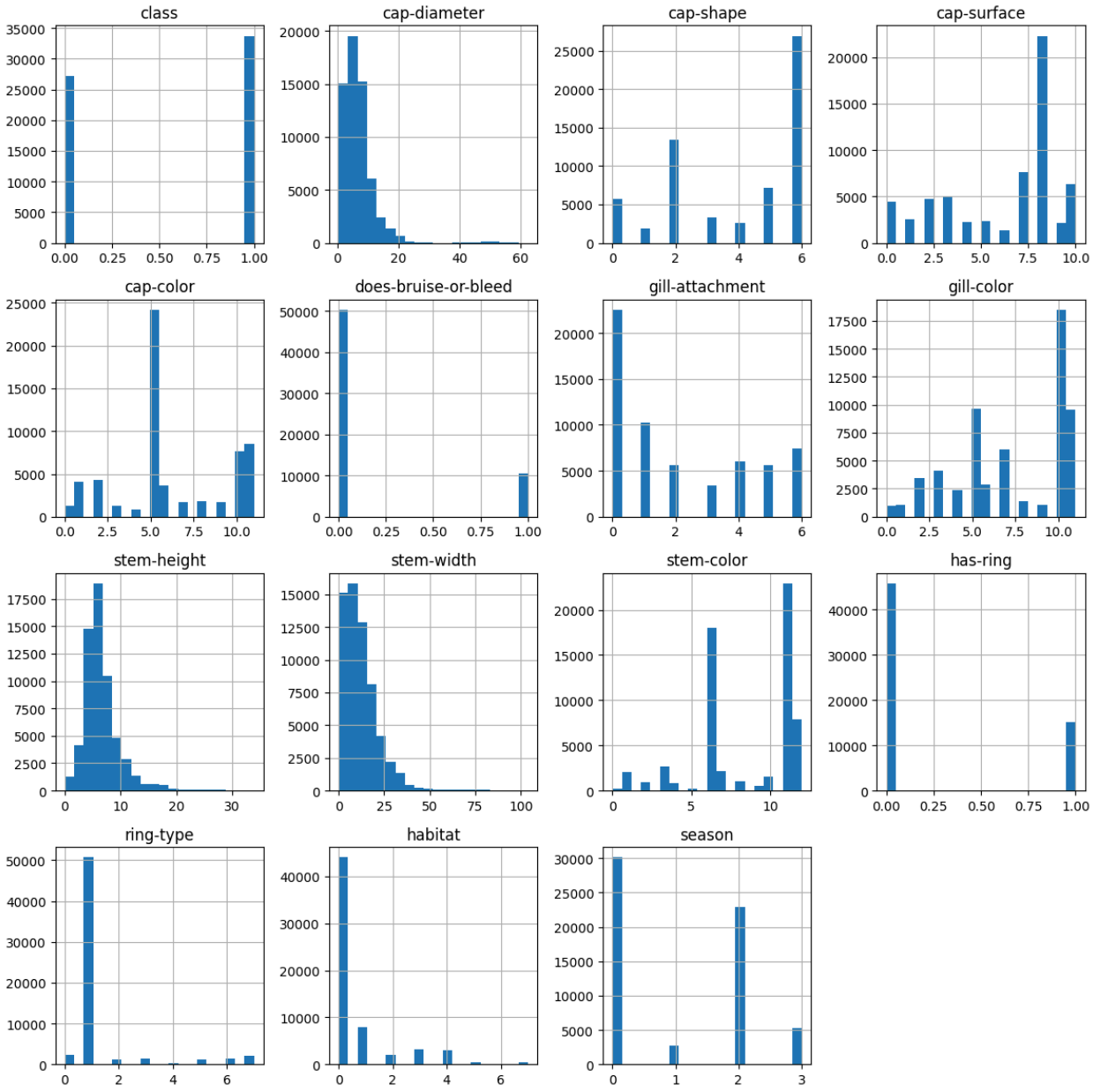
- **Output Feature Class Distribution**: The notebook checks the unique values and counts for the 'class' feature. The classes are 'p' (poisonous) and 'e' (edible).
    - Poisonous ('p'): 33,742 instances
    - Edible ('e'): 27,181 instances The classes are not perfectly equal, but reasonably balanced.
- **Bar Chart Representation**
    - A bar chart is plotted to show the distribution of edible versus poisonous mushrooms. This visual representation confirms the counts mentioned above.



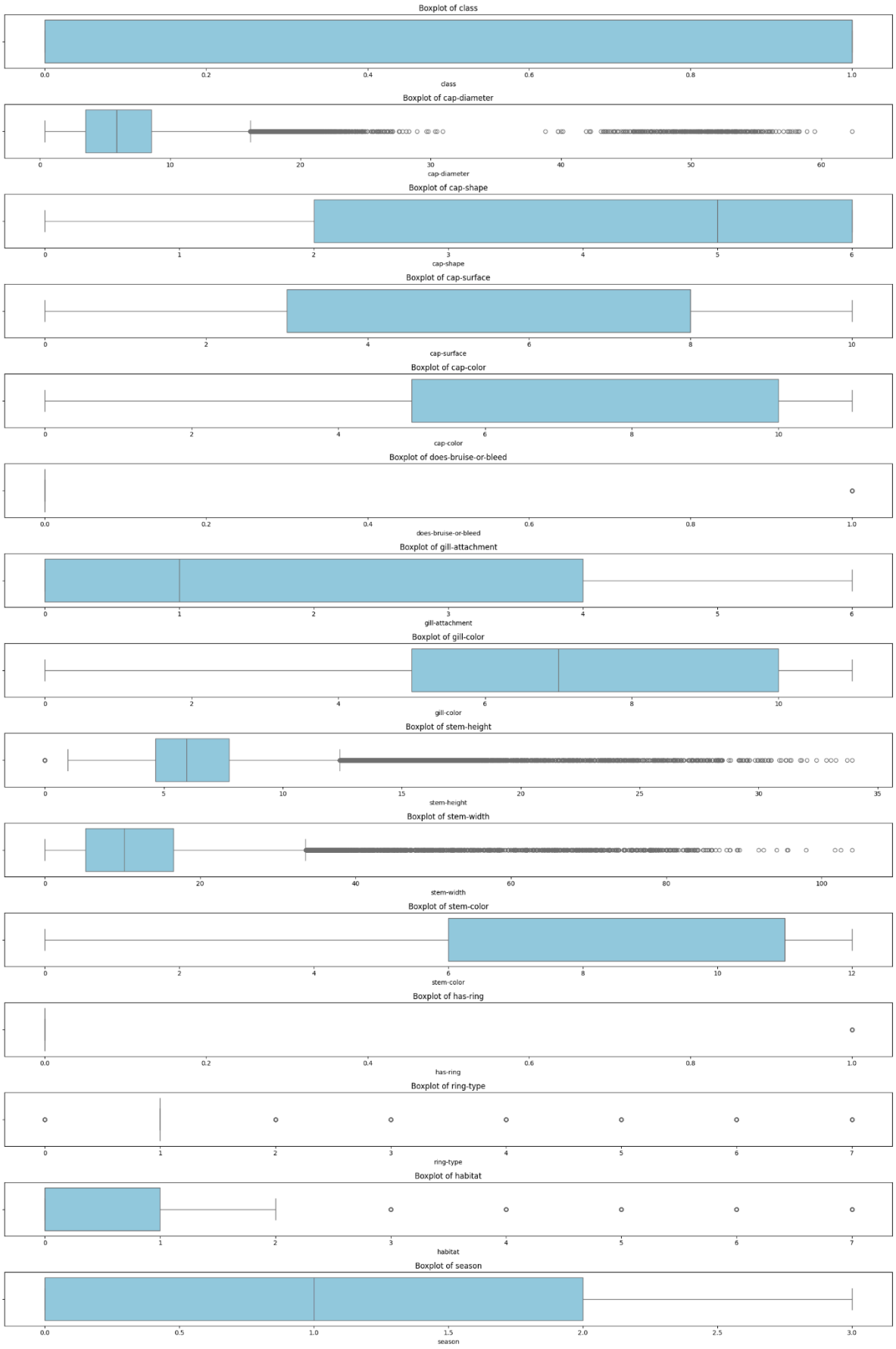
Exploratory Data Analysis (EDA)

The notebook includes several EDA steps:

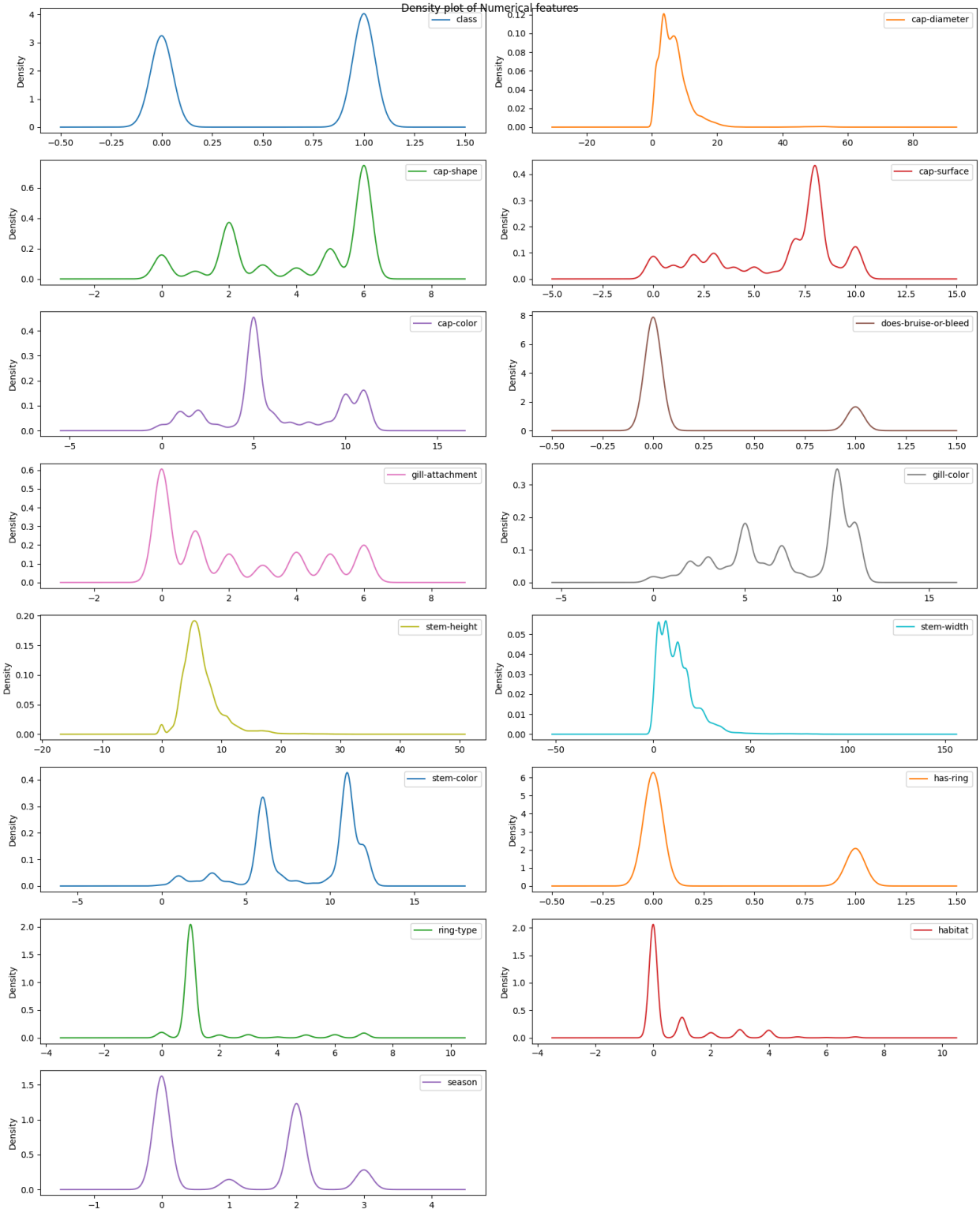
- **Histograms for Numerical Features (after encoding)** (This output displays histograms for all 15 features (including the encoded categorical ones) to show their distributions. This helps in understanding the spread and central tendency of each feature.



Boxplots for Numerical Features (after encoding)


Boxplot of class
class
Boxplot of cap-diameter
cap-diameter
Boxplot of cap-shape
cap-shape
Boxplot of cap-surface
cap-surface
Boxplot of cap-color
cap-color
Boxplot of does-bruise-or-bleed
does-bruise-or-bleed
Boxplot of gill-attachment
gill-attachment
Boxplot of gill-color
gill-color
Boxplot of stem-height
stem-height
Boxplot of stem-width
stem-width
Boxplot of stem-color
stem-color
Boxplot of has-ring
has-ring
Boxplot of ring-type
ring-type
Boxplot of habitat
habitat
Boxplot of season
season

    - Boxplots are generated for each of the 15 numerical features to identify their statistical distribution, including medians, quartiles, and potential outliers.
- **Density Plots for Numerical Features (after encoding)** Density plots are provided for all 15 features, offering a smoothed version of the histograms to visualize the probability density of each feature.


Density plot of Numerical features
class
cap-diameter
cap-shape
cap-surface
cap-color
does-bruise-or-bleed
gill-attachment
gill-color
stem-height
stem-width
stem-color
has-ring
ring-type
habitat
season
Density

3. Dataset Pre-processing

Null / Missing Values

- **Fault**: The dataset contains null/missing values in several columns. The initial check shows columns like cap-surface, gill-attachment, gill-spacing, stem-root, stem-surface, veil-type, veil-color, ring-type, and spore-print-color have missing values
    1. gill-spacing has 41.04% missing values.
    2. stem-root has 84.39% missing values.
    3. stem-surface has 62.43% missing values.
    4. veil-type has 94.80% missing values.
    5. veil-color has 87.86% missing values.
    6. spore-print-color has 89.60% missing values.
    7. Cap-surface has 23.12% missing values
- **Solution**:
    1. Columns with more than a 40% threshold of missing values are dropped. These include gill-spacing, stem-root, stem-surface, veil-type, veil-color, spore-print-color, Cap-surface.
    2. For the remaining columns with missing values (cap-surface, gill-attachment, ring-type), imputation is performed using the SimpleImputer with the 'most_frequent' strategy, as these are categorical features (or treated as such for imputation if their dtype was object before encoding).

Duplicate Values

- **Fault**: The dataset might contain duplicate rows.
- **Solution**: Duplicate rows are removed using df.drop_duplicates().

Categorical Values

- **Fault**: Machine learning models often require numerical input. The dataset contains categorical features (object dtype).

- **Solution**: Label Encoding is applied to all categorical features to convert them into numerical representations. The features encoded are: 'class', 'cap-shape', 'cap-surface', 'cap-color', 'does-bruise-or-bleed', 'gill-attachment', 'gill-color', 'stem-color', 'has-ring', 'ring-type', 'habitat', 'season'.

Feature Scaling

- **Fault**: Numerical features might have different scales, which can affect the performance of some algorithms (e.g., KNN, Neural Networks).
- **Solution**: `StandardScaler` is used to standardize the features by removing the mean and scaling to unit variance. This is applied to the feature set `X` (all columns except 'class').

---

4. Dataset Splitting

- The dataset is split into training and testing sets.
- **Test Set**: 30% of the data.
- **Train Set**: 70% of the data.
- **Stratification**: Stratified splitting is performed based on the target variable 'class' (`stratify=y`) to ensure that both training and testing sets have a similar proportion of edible and poisonous mushrooms.
- A `random_state` is used for reproducibility.

---

5. Model Training & Testing

The following models are trained and tested on the dataset:

Logistic Regression

- A Logistic Regression model is trained and its accuracy is evaluated.
- The accuracy score is printed.
- A classification report and confusion matrix are generated and displayed

Random Forest Classifier

- A Random Forest Classifier is trained and its accuracy is evaluated.
- The accuracy score is printed.
- A classification report and confusion matrix are generated and displayed

K-Nearest Neighbors (KNN)

- A KNN classifier is trained (default n_neighbors=5) and its accuracy is evaluated.
- The accuracy score is printed.
- A classification report and confusion matrix are generated and displayed

Neural Network

- A Sequential Neural Network model is built using Keras (TensorFlow backend).
    - **Architecture**:
        - nn_model.add(Dense(64, input_dim=X_train.shape[1], activation='relu')) # Layer 1
        - nn_model.add(Dense(128, activation='relu')) # Layer 2
        - nn_model.add(Dense(128, activation='relu')) # Layer 3
        - nn_model.add(Dense(64, activation='relu')) # Layer 4
        - nn_model.add(Dense(64, activation='relu')) # Layer 5
        - nn_model.add(Dense(32, activation='relu')) # Layer 6
        - nn_model.add(Dense(32, activation='relu')) # Layer 7
        - nn_model.add(Dense(1, activation='sigmoid')) # Output Layer
    - **Training**: Trained for 25 epochs with a batch size of 64, using a validation split of 20%.
    - **Evaluation**:

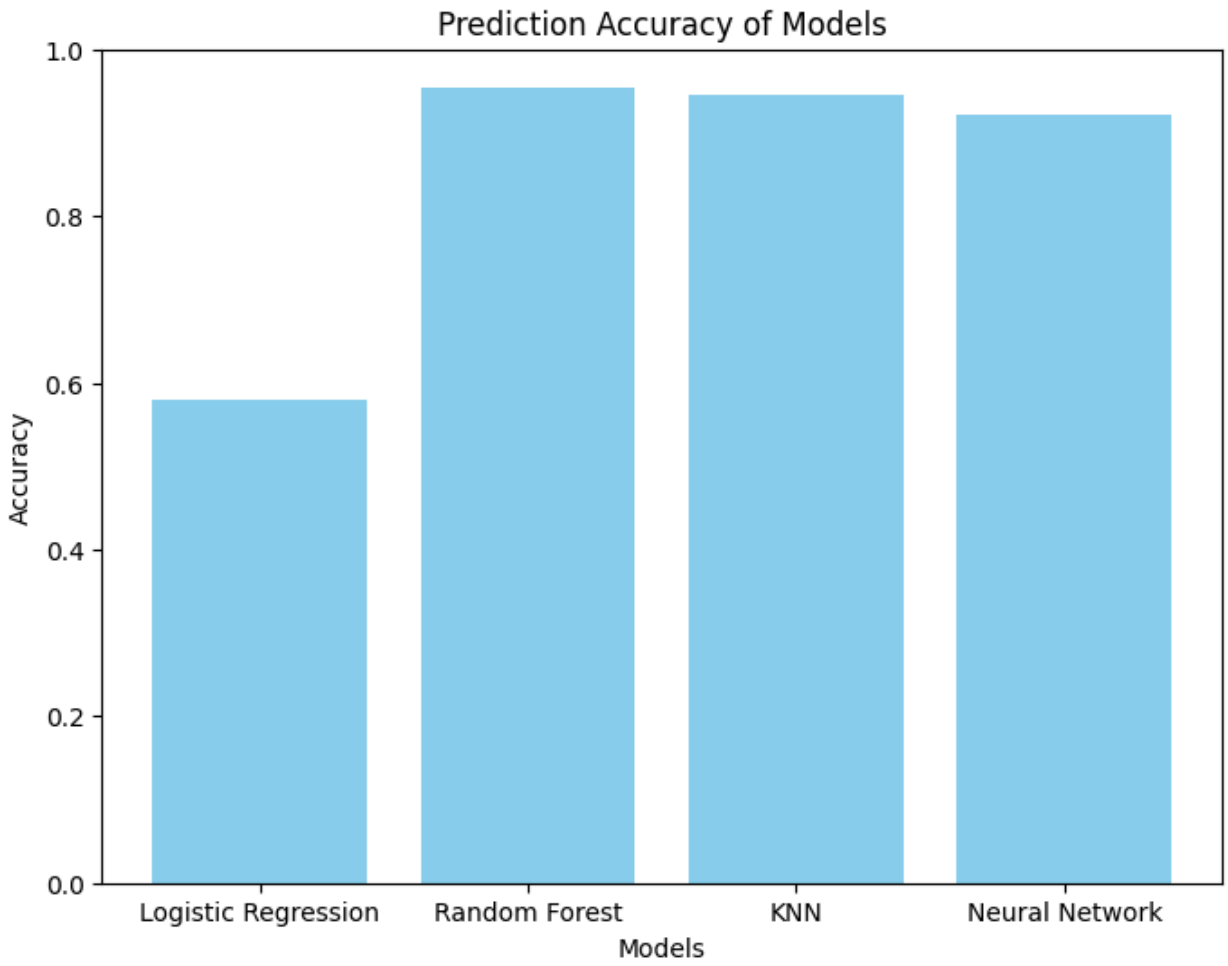
- Test loss and test accuracy are printed.
- A classification report and confusion matrix are generated and displayed

Training vs Validation Accuracy Plot Training vs Validation Loss Plot

6. Model Selection/Comparison Analysis

The notebook evaluates the models based on several metrics:

- **Prediction Accuracy Bar Chart** (Generated in the cell with id: G845y4P4SiDp)
  - A bar chart compares the accuracy scores of all trained models: Logistic Regression, Random Forest, KNN, and Neural Network.



- **Precision, Recall Comparison**
    - Classification reports are generated for each model, which include precision, recall, and F1-score for each class (edible/poisonous). These values are printed below each model's accuracy score.

```
Logistic Regression:
              precision    recall  f1-score   support

           0       0.55      0.35      0.42      8150
           1       0.59      0.77      0.67     10127

    accuracy                           0.58     18277
   macro avg       0.57      0.56      0.55     18277
weighted avg       0.57      0.58      0.56     18277

Random Forest:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95      8150
           1       0.95      0.97      0.96     10127

    accuracy                           0.95     18277
   macro avg       0.96      0.95      0.95     18277
weighted avg       0.95      0.95      0.95     18277

KNN:
              precision    recall  f1-score   support

           0       0.94      0.93      0.94      8150
           1       0.95      0.96      0.95     10127

    accuracy                           0.95     18277
   macro avg       0.95      0.95      0.95     18277
weighted avg       0.95      0.95      0.95     18277

Nural Network:
              precision    recall  f1-score   support

           0       0.92      0.90      0.91      8150
           1       0.92      0.94      0.93     10127

    accuracy                           0.92     18277
   macro avg       0.92      0.92      0.92     18277
weighted avg       0.92      0.92      0.92     18277
```

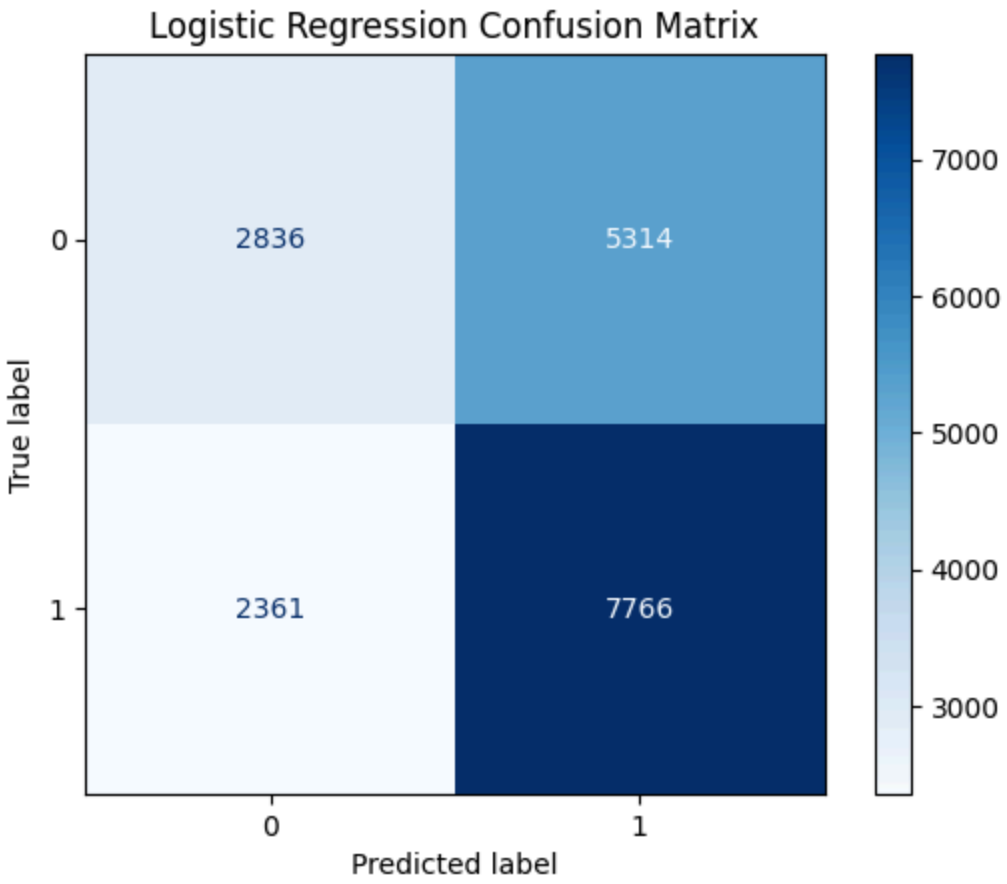
Model Selection/Comparison Analysis

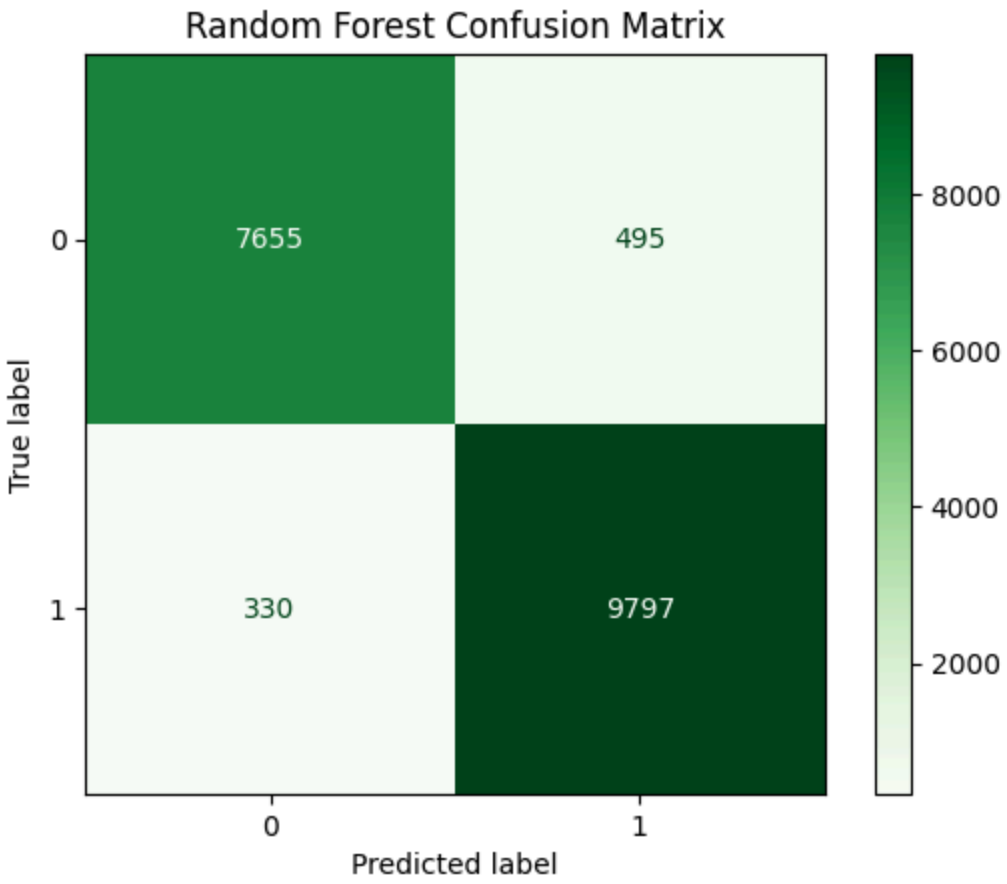
The following models were trained and evaluated:

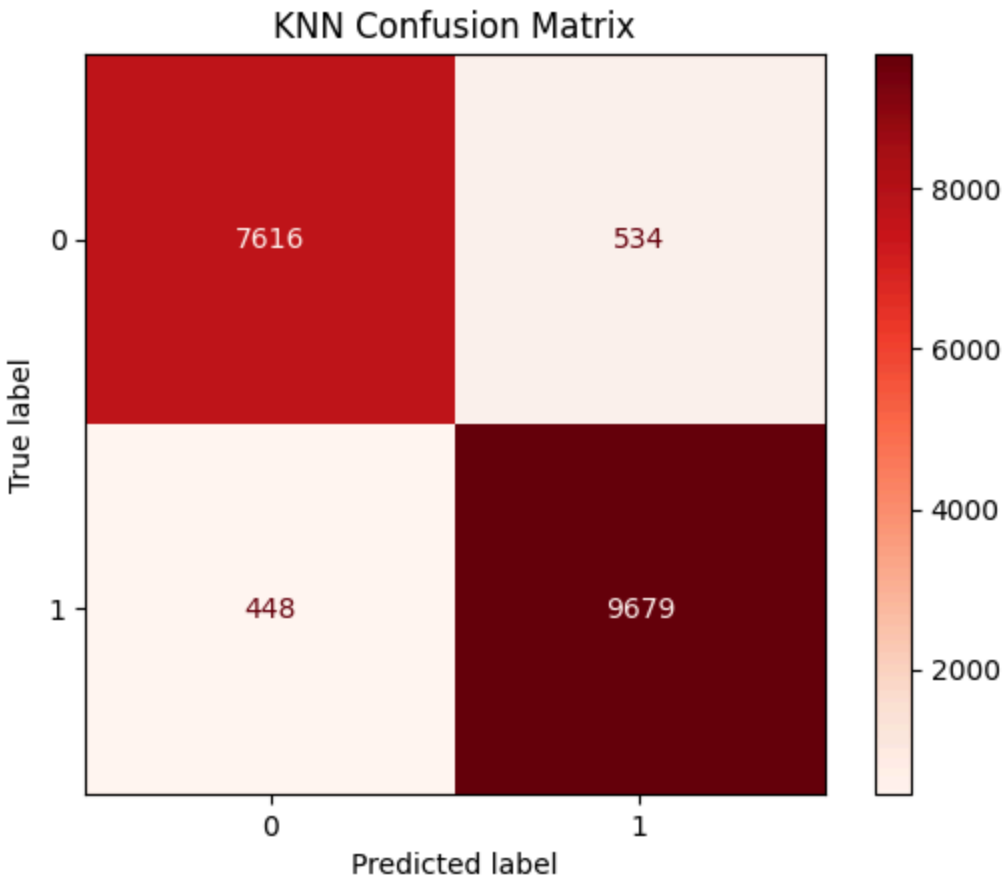
- **Logistic Regression**: Accuracy = 58%
- **Random Forest**: Accuracy = 95%
- **K-Nearest Neighbors (KNN)**: Accuracy = 95%
- **Neural Network**: Accuracy = 92%

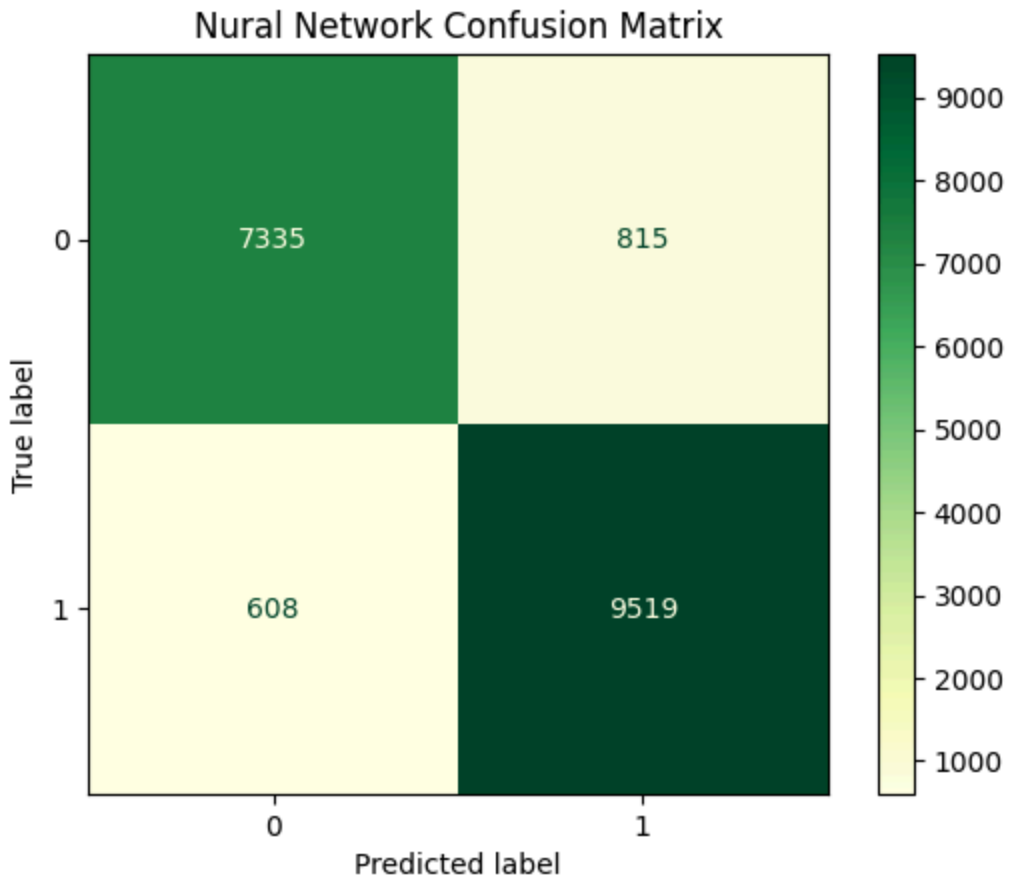
Random Forest and KNN outperformed all other models, followed closely by the Neural Network. Logistic Regression, while simple, was not effective due to its linear nature and limited capacity to capture complex patterns.

- **Confusion Matrix**
    - Confusion matrices are plotted for all models in a single figure with multiple subplots (Generated in the cell with id: nT0b06F9SiBv). Each matrix shows the true positives, true negatives, false positives, and false negatives for the respective model.

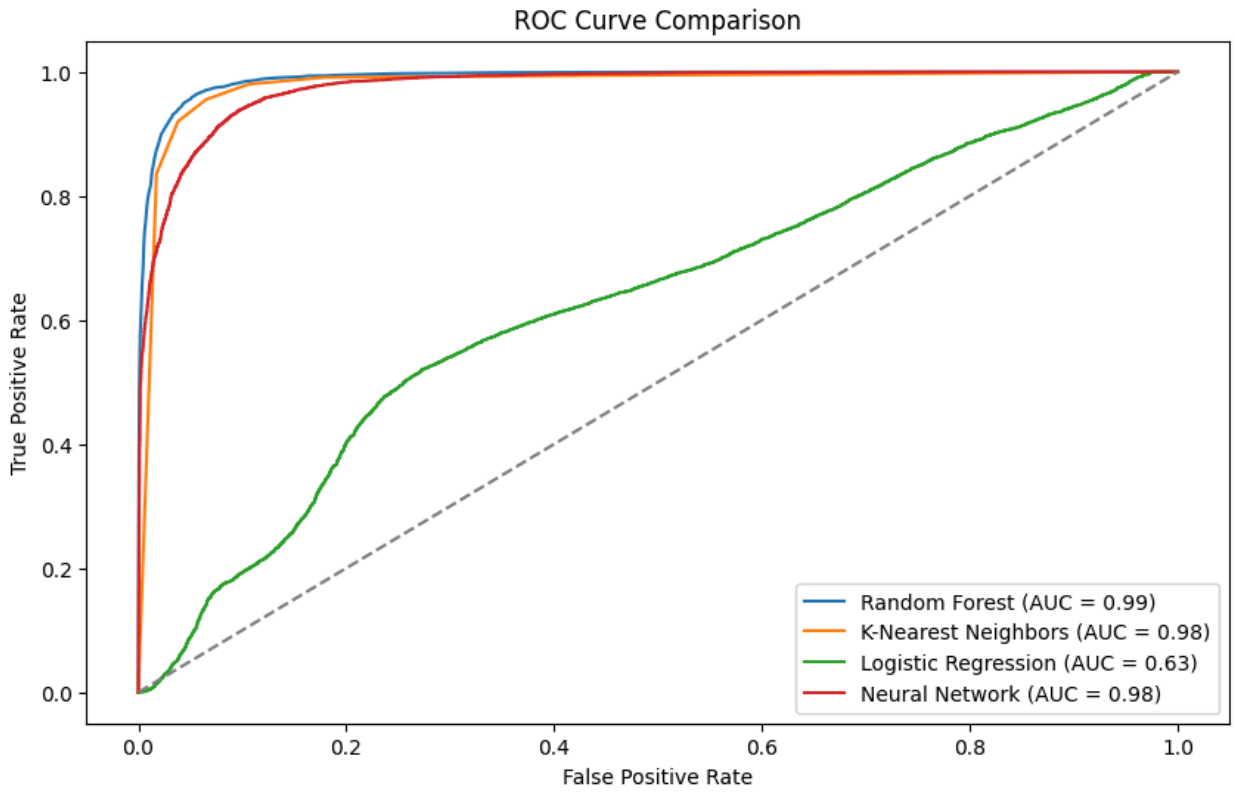

Logistic Regression Confusion Matrix
2836
5314
2361
7766
0
1
True label
0
1
Predicted label
7000
6000
5000
4000
3000


Random Forest Confusion Matrix
7655
495
330
9797
0
1
True label
0
1
Predicted label
8000
6000
4000
2000


KNN Confusion Matrix
7616
534
448
9679
0
1
True label
0
1
Predicted label
8000
6000
4000
2000



- **AUC Score, ROC Curve**
  - ROC curves are plotted for all models on the same graph for comparison (Generated in the cell with id: bY4l0C13SiGP).
  - The AUC (Area Under the ROC Curve) scores for each model are also calculated and displayed in the legend of the ROC curve plot.



7. Conclusion

The mushroom classification project demonstrates the strong capability of machine learning models in distinguishing between edible and poisonous mushrooms with high accuracy. Among the models tested, Random Forest and K-Nearest Neighbors (KNN) achieved the best performance, both reaching an accuracy of 95% with balanced precision, recall, and F1-scores, indicating their effectiveness in handling the dataset's structure. The Neural Network also performed well with a 92% accuracy, showcasing its strength in capturing complex data patterns. In contrast, Logistic Regression served as a baseline but struggled with an accuracy of only 58%, highlighting its limitations with non-linear relationships. These results underscore the importance of selecting models suited to the data characteristics. The success across models was also heavily influenced by comprehensive pre-processing steps, including handling missing values, encoding

categorical features, and scaling, which prepared the data for optimal learning. Overall, this project illustrates that with proper data preparation and appropriate model selection, machine learning can effectively and accurately address classification challenges involving categorical data.