CSE422 Lab Project Report Fall 25
Mushroom Detection
Sec 17
Group 01

Alma Usha (ID: 23201452)
Tahera Toor (ID: 23201552)

Table of Contents

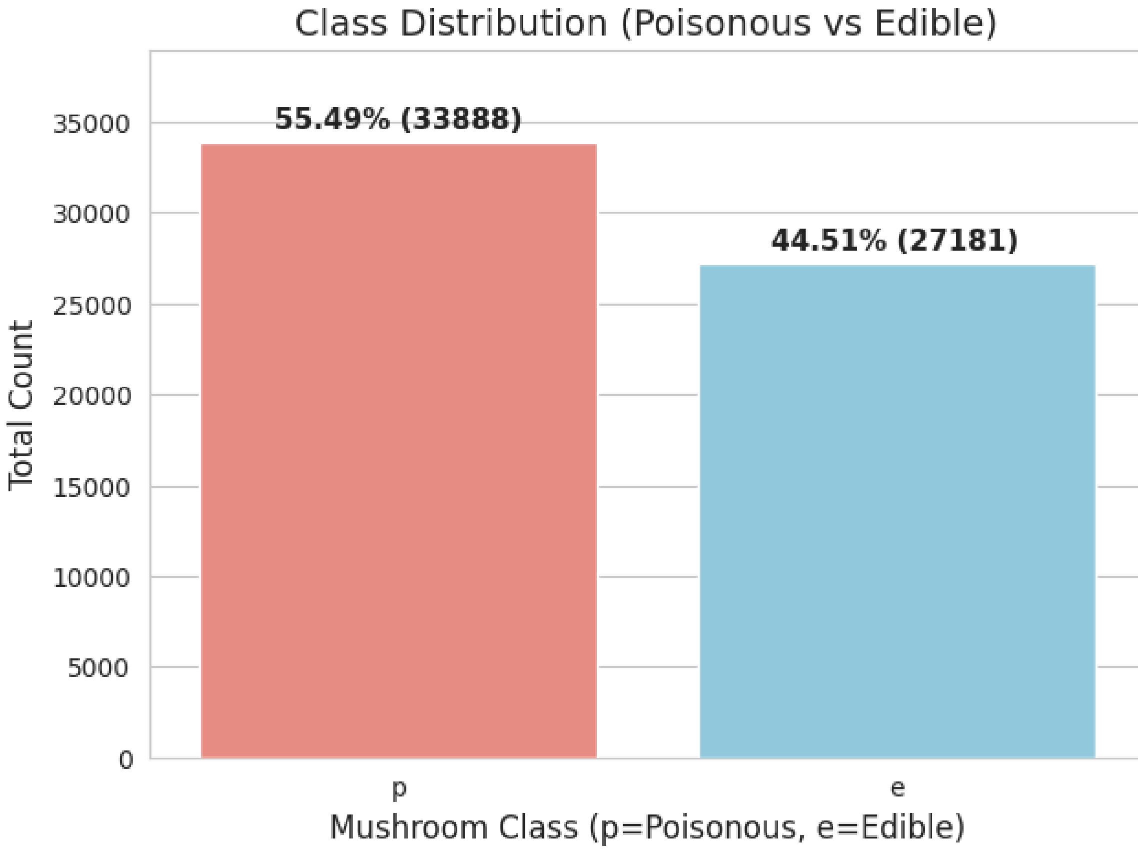
Introduction

The objective of this project is to entail machine learning models to solve the problem of identifying mushrooms and explore the effectiveness of the models learned in the lab in real life. This report presents a detailed analysis of the project classifying mushrooms as poisonous (p) or edible (e). The project follows a systematic approach from data exploration to model deployment, implementing multiple supervised and unsupervised learning techniques to achieve optimal classification performance.

2.1 Dataset Description

- The dataset contains **21 features** including musroom's Cap-diameter, Cap-shape, Cap-surface, Cap-color, does-bruise-or-bleed, Gill-attachment, Gill-spacing, Gill-color, Stem-height, Stem-width, Stem-root, Stem-surface, Stem-color, Veil-type, Veil-color, Has-ring, Ring-type, Spore-print-color, Habitat, Season along with a target feature indicating p(poisonous) or e(edible).
- The problem is a **fundamental classification** problem where the goal is to identify whether the given mushroom is poisonous or edible. As the output can only be one of the two classes, so it's an classification problem.
- There are **61069 data points** in the dataset.
- The dataset contains both **quantitative**(e.g, Cap-diameter, Stem-height, Stem-width) and **categorical** data(e.g, Cap-shape, cap-colour etc)
- Yes, the categorical variables must be encoded to apply machine learning models to it. Machine learning algorithms, especially those in scikit-learn, work with numerical data. We have converted these categorical labels into a numerical format so that the models can understand and process easily.
- From the **correlation heatmap**(presented at pg. 3) we can find that there is a weak correlation between most of the input features. The strongest positive correlation is between cap-diameter and stem-width, suggesting that mushrooms with larger caps tend to have thicker stems.

2.2 Imbalance



We have fairly balanced dataset of mushrooms with the percentage of class 'p' (Poisonous) is 55.49% and the percentage of class 'e' (Edible) is 44.51% .

2.3 Exploratory Data Analysis

Correlation heatmap-

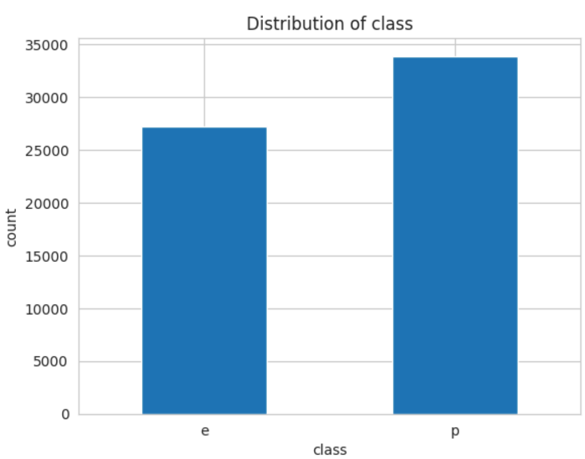
The heatmap reveals a few moderately strong correlations between the variables presented here. There aren't any strong negative correlations present here, but there are weaker correlations as well, in the heatmap.

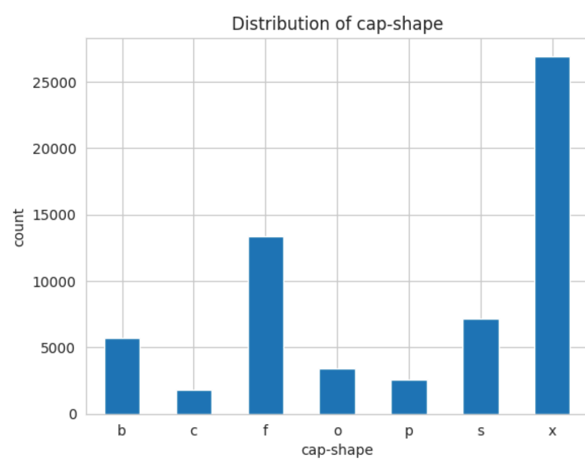
Data Overview

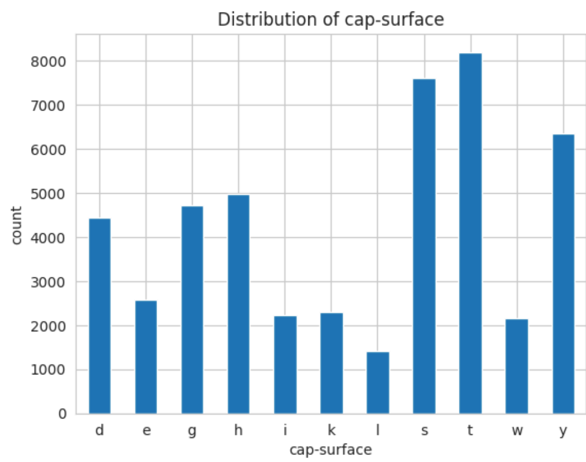
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61069 entries, 0 to 61068
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   class                 61069 non-null  object
 1   cap-diameter          61069 non-null  float64
 2   cap-shape             61069 non-null  object
 3   cap-surface           46949 non-null  object
 4   cap-color             61069 non-null  object
 5   does-bruise-or-bleed  61069 non-null  object
 6   gill-attachment       51185 non-null  object
 7   gill-spacing          36006 non-null  object
 8   gill-color            61069 non-null  object
 9   stem-height           61069 non-null  float64
 10  stem-width            61069 non-null  float64
 11  stem-root             9531 non-null   object
 12  stem-surface          22945 non-null  object
 13  stem-color            61069 non-null  object
 14  veil-type             3177 non-null   object
 15  veil-color            7413 non-null   object
 16  has-ring              61069 non-null  object
 17  ring-type             58598 non-null  object
 18  spore-print-color     6354 non-null   object
 19  habitat               61069 non-null  object
 20  season                61069 non-null  object
dtypes: float64(3), object(18)
memory usage: 9.8+ MB
```

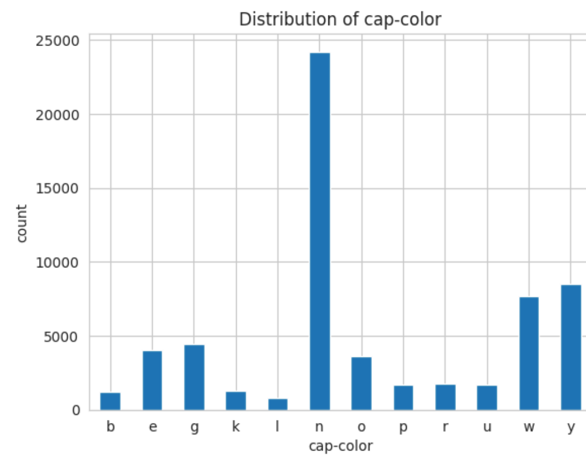
This is a mushroom dataset containing 61,000 entries of mushroom's different characteristics. It includes various data types, such as floats for cap-diameter, stem-height, stem-height, and categorical objects for all other characteristics of the mushroom dataset.

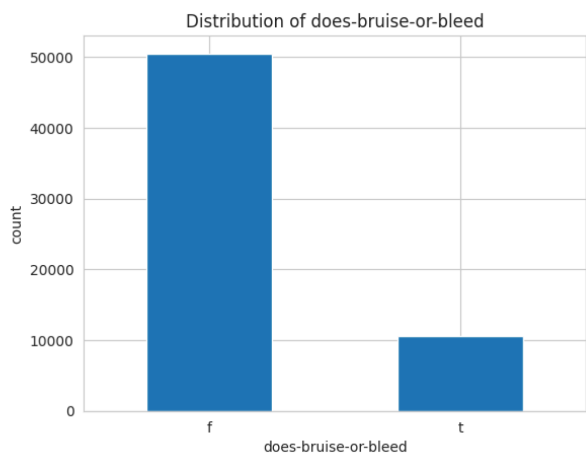
Distribution of categorical features

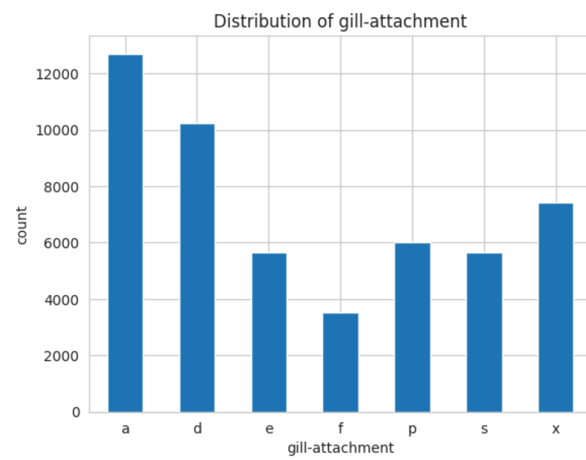


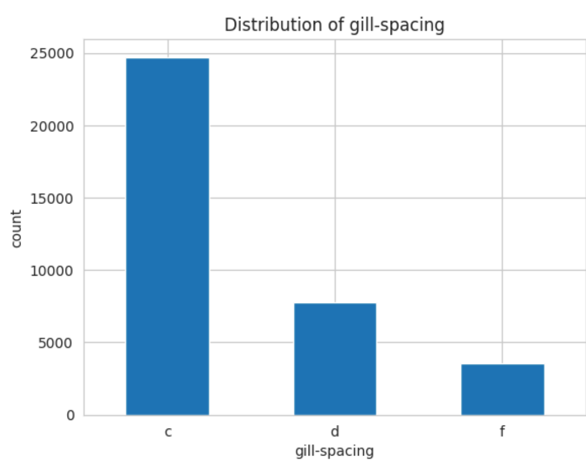


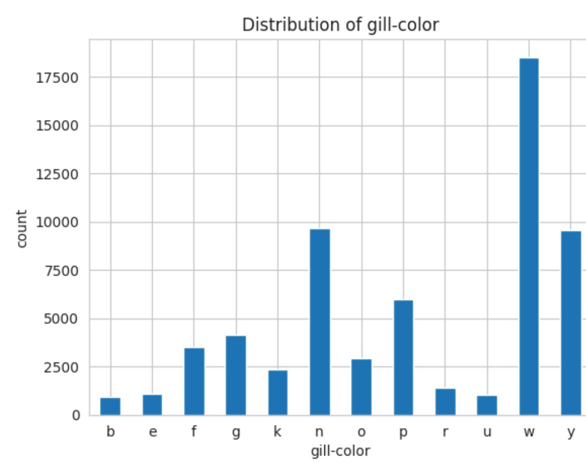

Distribution of cap-surface
count
0
1000
2000
3000
4000
5000
6000
7000
8000
d e g h i k l s t w y
cap-surface

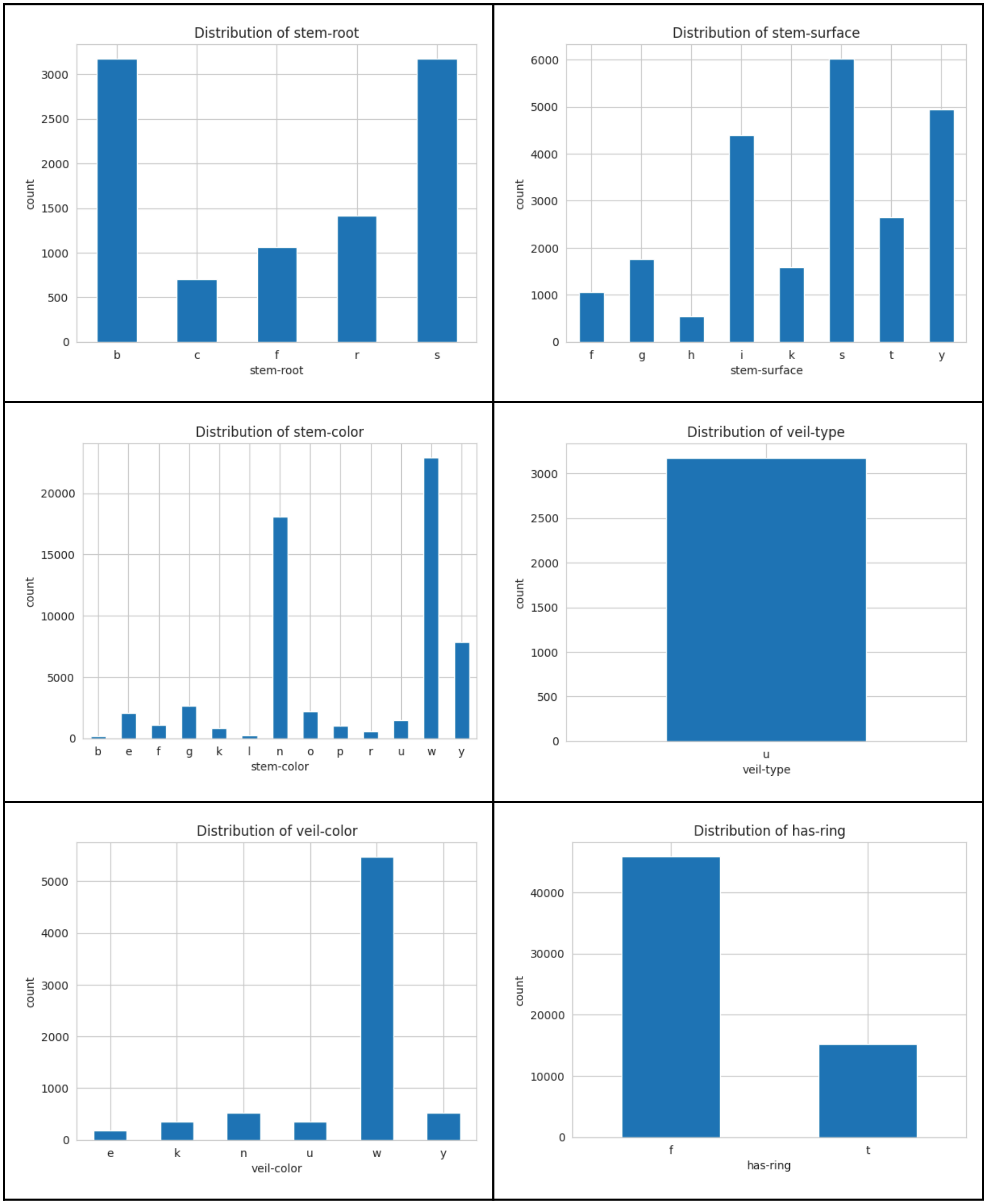

Distribution of cap-color
count
0
5000
10000
15000
20000
25000
b e g k l n o p r u w y
cap-color

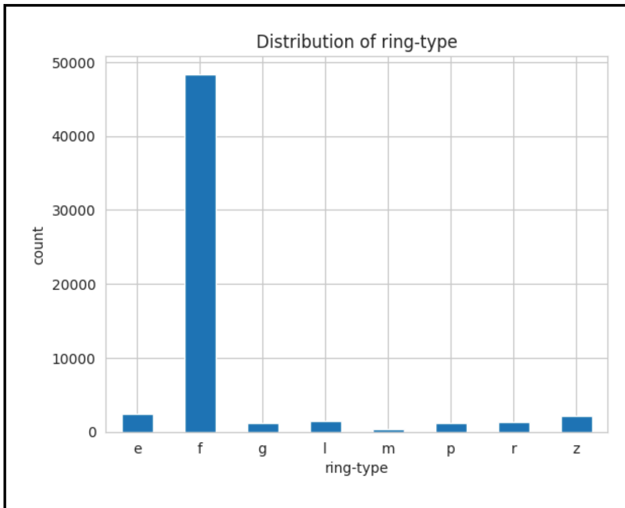

Distribution of does-bruise-or-bleed
count
0
10000
20000
30000
40000
50000
f t
does-bruise-or-bleed

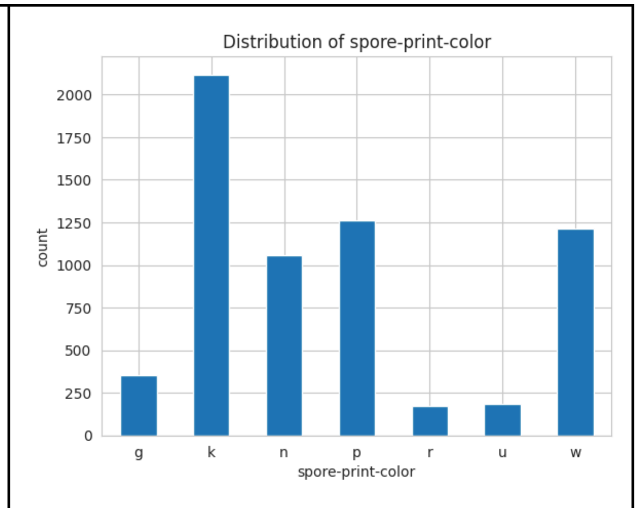

Distribution of gill-attachment
count
0
2000
4000
6000
8000
10000
12000
a d e f p s x
gill-attachment

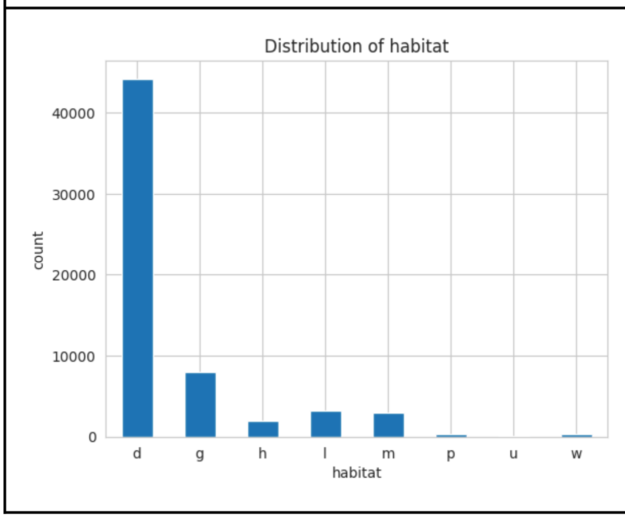

Distribution of gill-spacing
count
0
5000
10000
15000
20000
25000
c d f
gill-spacing

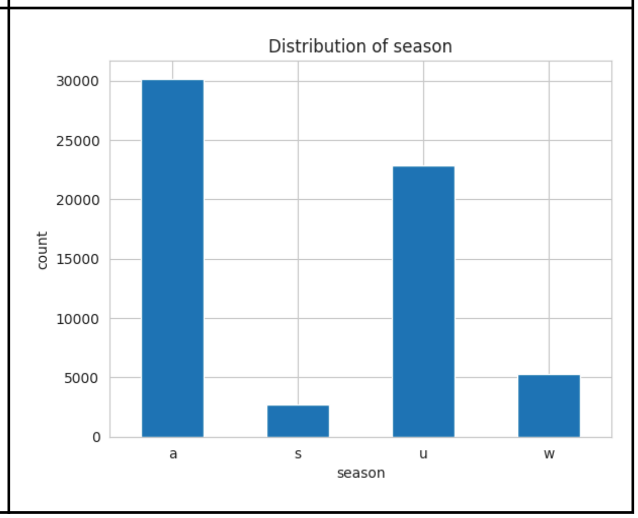

Distribution of gill-color
count
0
2500
5000
7500
10000
12500
15000
17500
b e f g k n o p r u w y
gill-color


Distribution of stem-root
count
3000
2500
2000
1500
1000
500
0
b
c
f
r
s
stem-root
Distribution of stem-surface
count
6000
5000
4000
3000
2000
1000
0
f
g
h
i
k
s
t
y
stem-surface
Distribution of stem-color
count
20000
15000
10000
5000
0
b
e
f
g
k
l
n
o
p
r
u
w
y
stem-color
Distribution of veil-type
count
3000
2500
2000
1500
1000
500
0
u
veil-type
Distribution of veil-color
count
5000
4000
3000
2000
1000
0
e
k
n
u
w
y
veil-color
Distribution of has-ring
count
40000
30000
20000
10000
0
f
t
has-ring

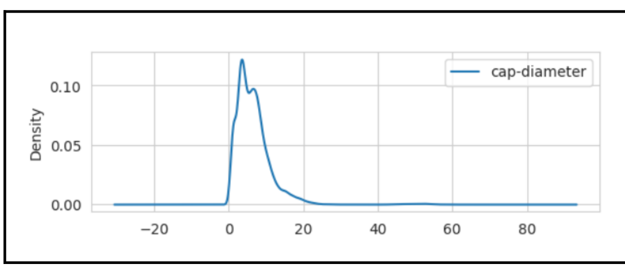


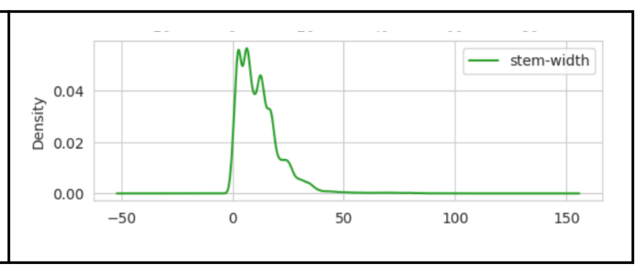


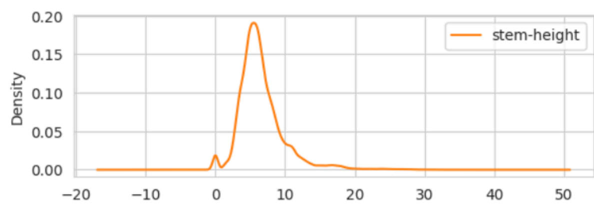




Density Plots







Density plots show the distribution of a continuous variable. The features are not normally distributed. All three features are concentrated in relatively narrow, positive ranges, skewed with right tails because of extreme outliers.

Dataset Preproccessing

Faults Handling-

1. **Null / Missing values**: Several categorical features, such as veil-type, veil-color, stem-root, stem-surface, spore-print-color contains high amounts of missing values (84 - 95%)

→ **Columns** with a very high missing percentage were removed, as keeping them would have reduced model performance.

2. **Remaining missing values**:

→ Filling the remaining missing values with **Mode imputation** as it prevents loss of data. After these imputation steps, all missing values in the identified columns have been handled.

3. **Categorical values**: As identified earlier, most features are categorical and need to be converted to a numerical format.

→ Used **One-Hot Encoding** to convert the categorical features into numerical representations.

4. **Data leakage**: The initial preprocessing created a target_bin column which was accidentally included in the feature set, creating perfect correlation with target.

→ **Explicitly dropped** target_bin from feature set and re-processed data without leakage columns.

Additionally- **scaled** the **numerical features** to a mean of 0 and a standard deviation of 1 using StandardScaler, to make the gradient descent to converge more efficiently

Dataset Splitting

The pre-processed dataset was split into training and testing sets to evaluate the models' performance on unseen data. We used a standard train_test_split with a test size of 20%, resulting in:

Train set: 80% of the data, used to train the models.
Test set: 20% of the data, used to test the models' performance.
The splitting process was also stratified, ensuring that the ratio of the two income classes in the training set is the same as in the testing set. This is important for imbalanced datasets.

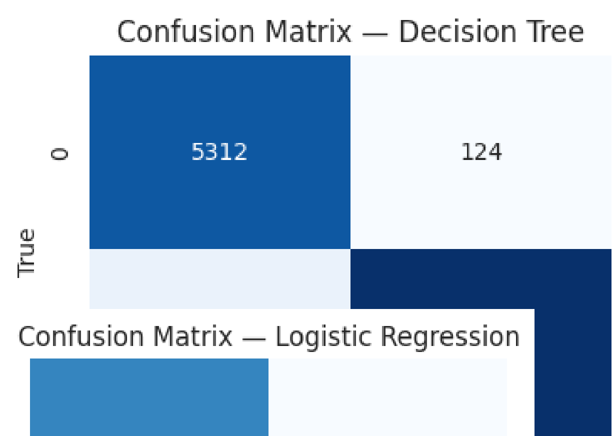
Model Training &Testing

We used Decision Tree, LogisticRegression, and Neural Network with Multiple Perceptrons to train the dataset for the supervised part. We also used K-Means clustering for the unsupervised part. However, to identify the best cluster value, we used the elbow method and silhouette point. Here, we used the silhouette point, which was used in the clustering.

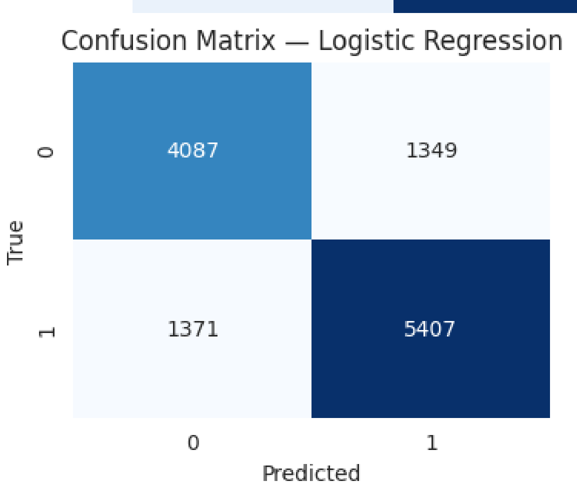
Model Comparison Analysis

We trained and tested **3 supervised** models on the dataset-

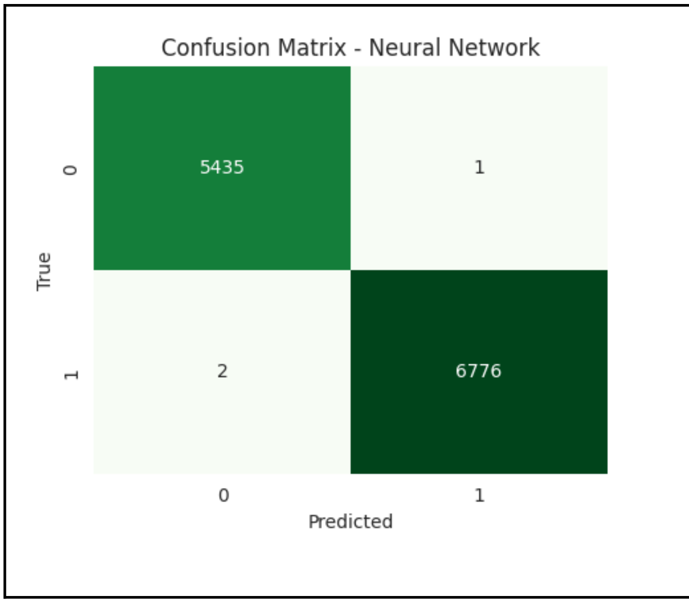
Decision Tree:
- Accuracy: 94.64%
- Precision: 98.05%
- Recall : 92.17%
- F1-score : 95.02%
- AUC : 0.9837

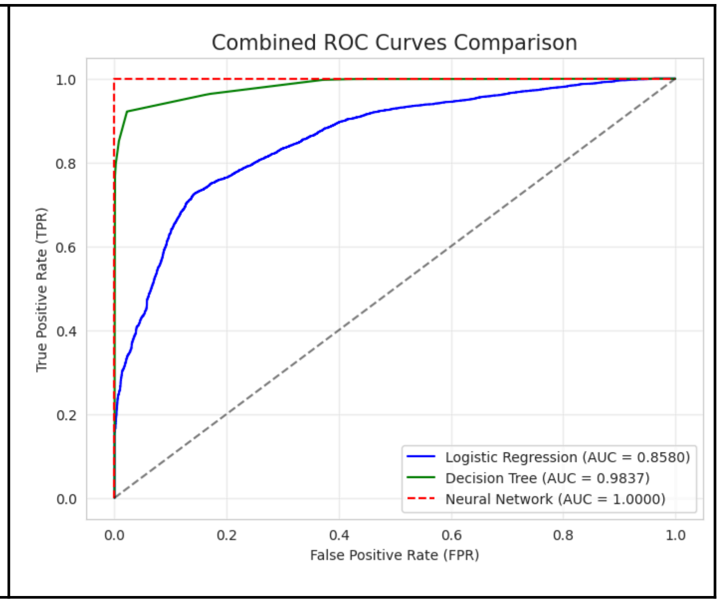


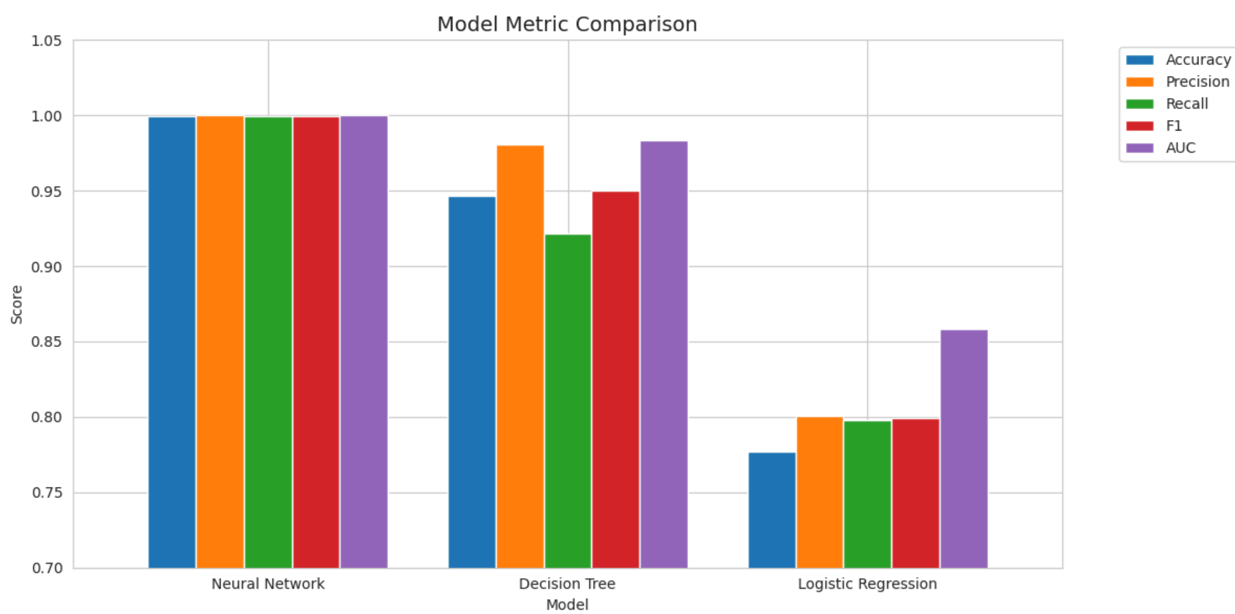
Logistic Regression:
- Accuracy: 77.73%
- Precision: 80.03%
- Recall : 79.77 %
- F1-score : 79.90%
- AUC : 0.8580



Neural Network (MLP Classifier):
- Accuracy: 99.97 %
- Precision: 99.98 %
- Recall : 99.97 %
- F1-score: 99.97 %
- AUC : 0.99







Unsupervised Learning: **K-Means Clustering**

In addition to supervised learning, we performed an unsupervised clustering analysis using K-Means. We chose an optimal number of clusters and visualized them using a heatmap. The heatmap showed that the clusters had distinct characteristics based on the mean feature values. This analysis provided valuable insights into the inherent structure of the data without using the class labels. But the clustering did not succeed as much because the result found was-

Poisonous mushrooms- 60%, Edible mushrooms- 40%

Whereas the real classes were- Poisonous: 55.2%, Edible: 44.8%

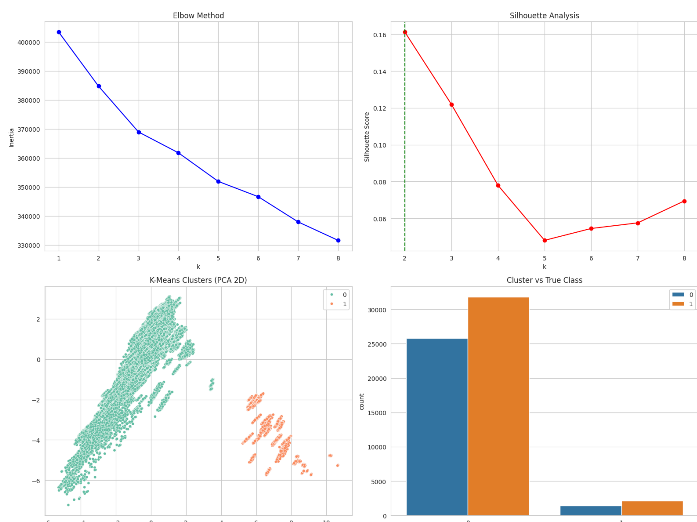
So, the clustering could not do a correct analysis of the dataset.

However, to identify the best cluster value, we used the elbow method. Here, 2 gives the elbow point, which was used in the clustering.

- Number of clusters (K) was set to 2.
- Clusters were visualized after dimensionality reduction.

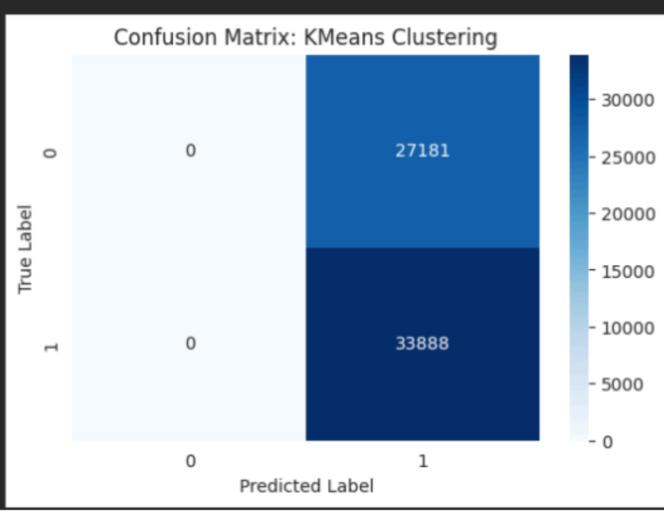
Observation:

- The clusters roughly correspond to edible and poisonous classes.
- Some overlap exists, indicating the need for supervised labels for higher accuracy.



```
Original shape: (61069, 80)
PCA explained variance ratio: [0.05706485 0.0444065 ]
Total variance explained: 0.10147134944050482
Optimal k: 2

--- Clustering Evaluation ---
Silhouette Score: 0.174
Adjusted Rand Index (ARI): -0.0019
Normalized Mutual Information (NMI): 0.0006

Cluster Distribution:
Cluster 0: 57539 samples (94.2%)
Cluster 1: 3530 samples (5.8%)

Cluster 0 (57539 samples):
  Class 0: 25769 (44.8%)
  Class 1: 31770 (55.2%)

Cluster 1 (3530 samples):
  Class 0: 1412 (40.0%)
  Class 1: 2118 (60.0%)
```

```
==============================
K-MEANS AS CLASSIFIER METRICS
==============================
Accuracy  : 0.5549
Precision : 0.5549
Recall    : 1.0000
F1-score  : 0.7138

Detailed Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00     27181
           1       0.55      1.00      0.71     33888

    accuracy                           0.55     61069
   macro avg       0.28      0.50      0.36     61069
weighted avg       0.31      0.55      0.40     61069
```



Model Comparison & Evaluation

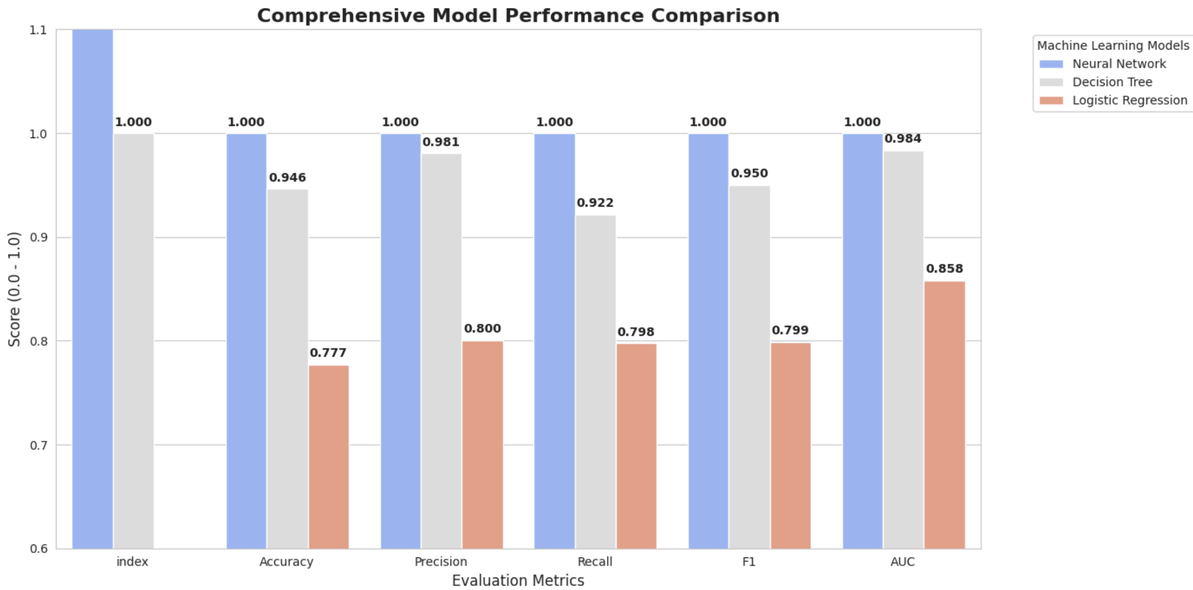
Evaluation Metrics (Classification)

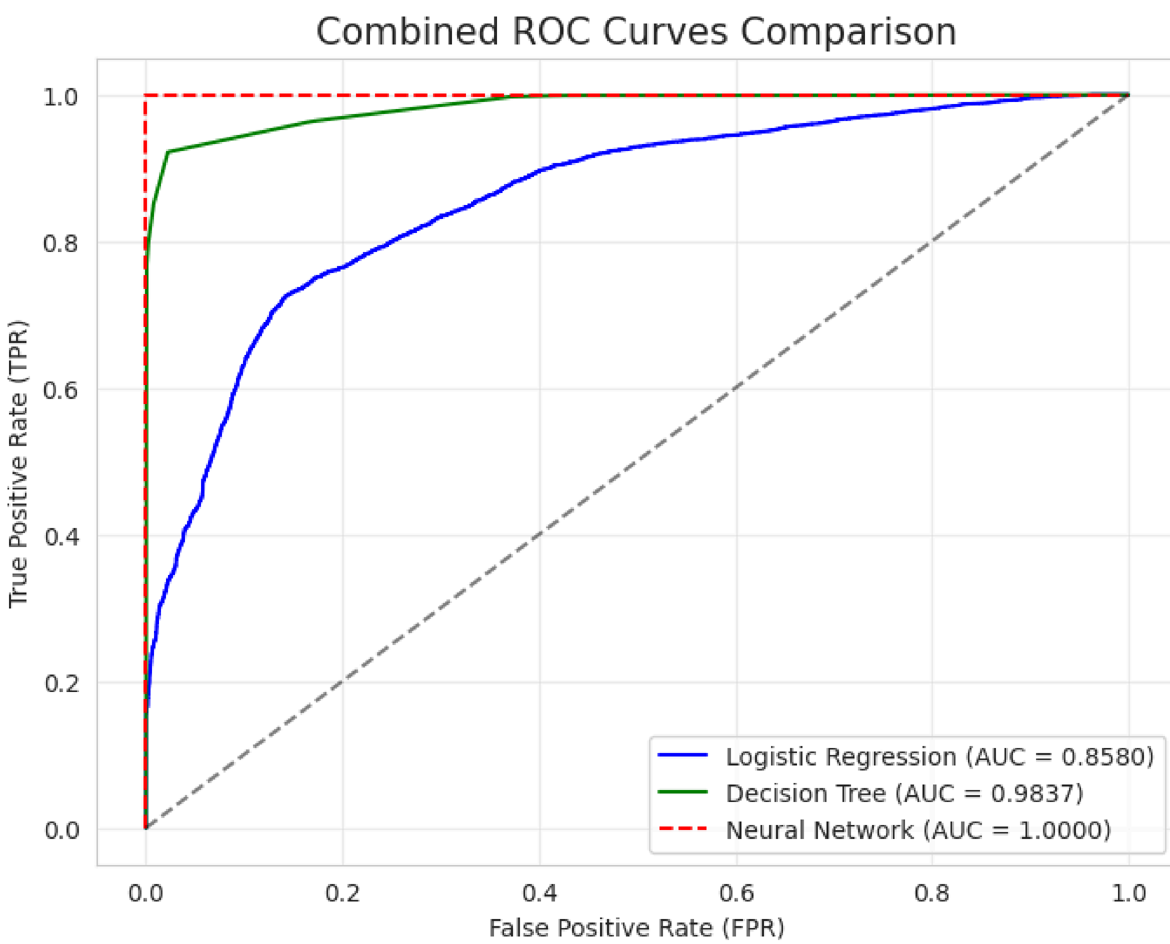
- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix
- ROC Curve and AUC Score

A bar chart was used to compare the accuracy of all models.

Comparing all supervised model

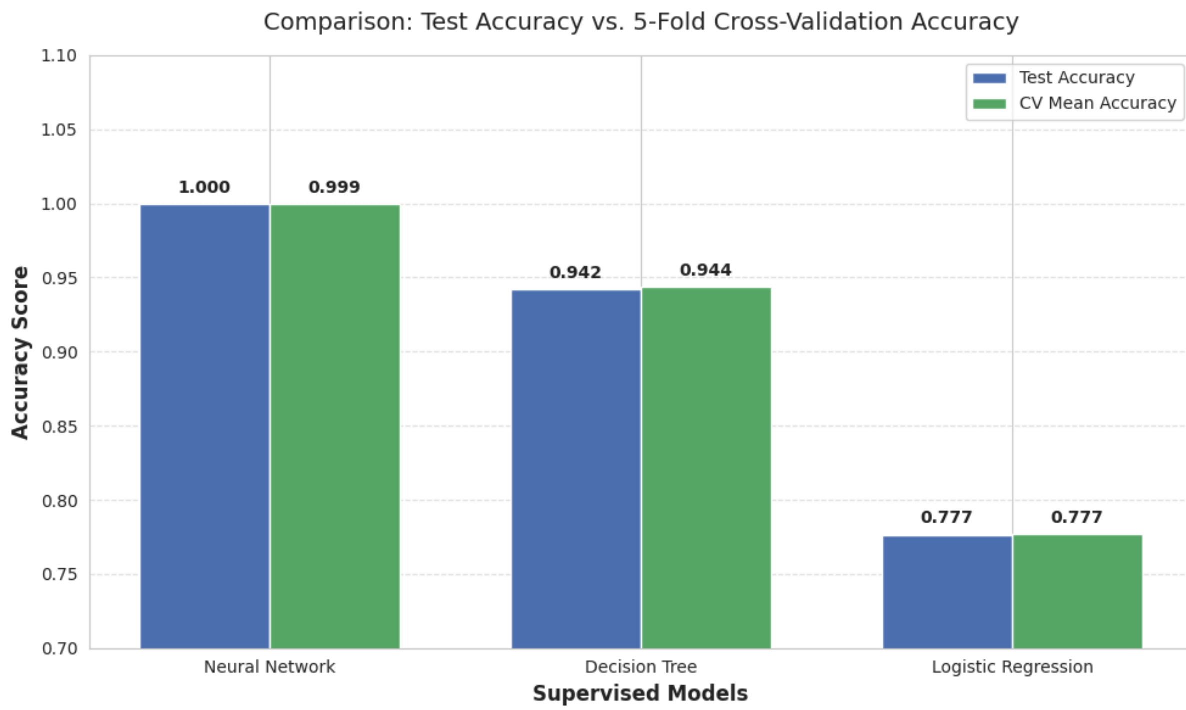
We can say neural network is the best for this dataset.


Comprehensive Model Performance Comparison
Score (0.0 - 1.0)
1.1
1.0
0.9
0.8
0.7
0.6
1.000
1.000
0.946
0.777
1.000
0.981
0.800
1.000
0.922
0.798
1.000
0.950
0.799
1.000
0.984
0.858
index
Accuracy
Precision
Recall
F1
AUC
Evaluation Metrics
Machine Learning Models
Neural Network
Decision Tree
Logistic Regression

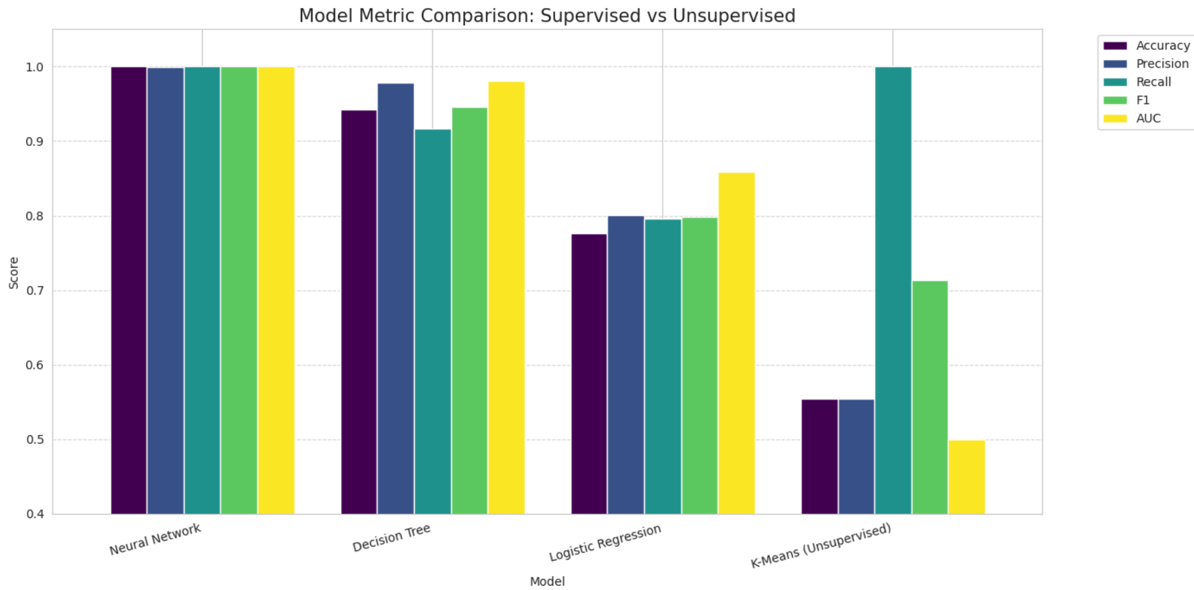

Combined ROC Curves Comparison
True Positive Rate (TPR)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate (FPR)
Logistic Regression (AUC = 0.8580)
Decision Tree (AUC = 0.9837)
Neural Network (AUC = 1.0000)

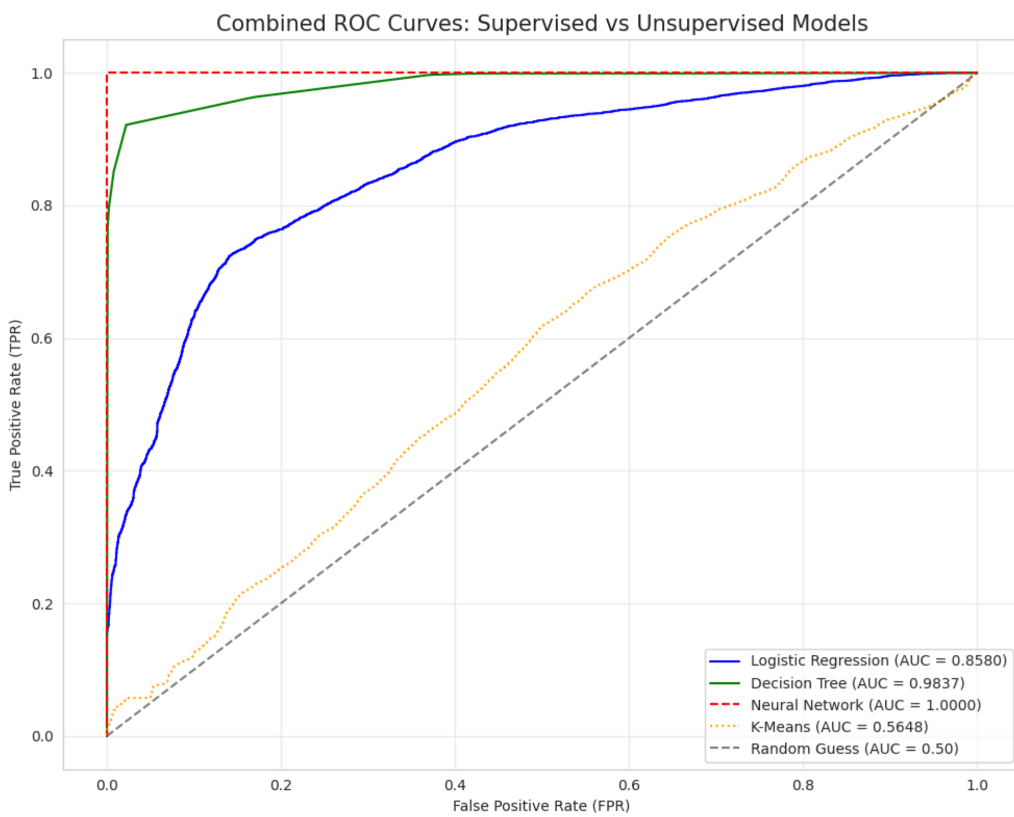
K Fold Cross Validation for supervised Models:

```
--- Starting 5-Fold Cross Validation ---
Processing Fold 1...
Processing Fold 2...
Processing Fold 3...
Processing Fold 4...
Processing Fold 5...

--- K-Fold Validation Complete (Supervised Models Only) ---
              Model  Accuracy  CV_Mean_Accuracy
     Neural Network  0.999918          0.999325
      Decision Tree  0.942197          0.943875
Logistic Regression  0.776568          0.776748
```



Comparing supervised and unsupervised both models:


Model Metric Comparison: Supervised vs Unsupervised
Accuracy
Precision
Recall
F1
AUC
Score
1.0
0.9
0.8
0.7
0.6
0.5
0.4
Neural Network
Decision Tree
Logistic Regression
K-Means (Unsupervised)
Model


Combined ROC Curves: Supervised vs Unsupervised Models
True Positive Rate (TPR)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate (FPR)
Logistic Regression (AUC = 0.8580)
Decision Tree (AUC = 0.9837)
Neural Network (AUC = 1.0000)
K-Means (AUC = 0.5648)
Random Guess (AUC = 0.50)

Results Summary:

Model	Accuracy	AUC Score	Performance Ranking
Neural Network	**~1.000**	**1.0000**	**Best (1st)** — Perfect separation of classes.
Decision Tree	**~0.94**	**0.9837**	**Excellent (2nd)** — Very high precision and AUC.
Logistic Regression	**~0.78**	**0.8580**	**Good (3rd)** — Solid baseline, but less accurate than others.
K-Means	**0.5549**	**0.5648**	**Poor (4th)** — Failed to distinguish classes effectively.

Conclusion

From the results, it's clear that the **Neural Network** (MLP) provided the best performance for this classification task. Its accuracy, precision, and recall scores were consistently higher than those of Decision Tree and Logistic Regression, which is reflected in its superior ROC AUC score (nearing 1). Although the dataset was fairly simple, the results suggest that the non-linear capabilities of the Neural Network did better to capture the complex relationships within the data.

Here, higher percentage of precision means almost no false positives, so when the model would predict a mushroom to be edible, it would be correct almost always (99.97% for neural network). On the other hand, higher percentage of recall means almost no false negatives, which means it would not miss any poisonous mushrooms.

The main challenge was the null values of the dataset. We had to implement 2 specific techniques to handle this. Further improvements could be made to make the clustering more accurate by using techniques such as- not discarding any original features and training with DBSCAN or hierarchical clustering to better train the model. Despite the challenges, the project successfully demonstrates the entire machine learning pipeline, from data cleaning and pre-processing to model training and performance evaluation.