Machine Learning

Course Project Report (Classification)

(Draft - 03, Team No: 06)

Title of the project: Mushroom Classification Dataset

Student 1 : Abbinav Sankar Kailasam, abbinav.k-25@scds.saiuniversity.edu.in
Student 2 : Sounak Saha, sounak.s-25@scds.saiuniversity.edu.in

ML Category: Classification

---

1. Introduction

Mushrooms are available in a great variety among nature, some definitely edible, some definitely poisonous while others are of unknown edibility. The Mushroom Classification dataset provides 20 different features that would help predict whether a mushroom is poisonous or edible. The dataset includes 61,069 mushrooms with caps based on 173 species. Each mushroom is classified between definitely poisonous (poisonous and unknown edibility combined) and definitely edible. The objective is to build different machine learning models that will help classify a new instance of mushroom.

- Problem statement: Predict the classification of mushrooms depending on its features.

2. Dataset and Features

Two datasets given are primary data and secondary data. The primary dataset consists of names of various species of mushrooms with their respective characteristics and class (poisonous or edible), while the secondary dataset consists of several instances of mushrooms with their characteristics and class. As the number of instances are greater in the secondary dataset, this data will be utilised to train all machine learning models.

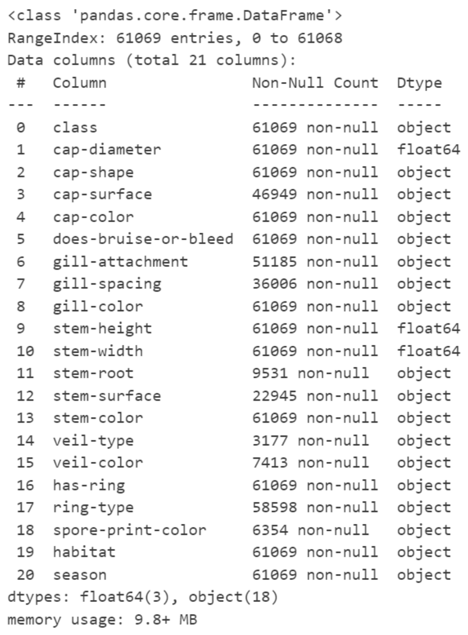
The dataset has 61,069 samples and 21 features. The various input features consist of many characteristics of mushrooms like cap shape, cap diameter, gill attachment, gill spacing, stem height, stem width, veil type, veil colour, etc., while the target feature is the identification between poisonous and edible.

2.1 Exploratory Data Analysis of the dataset (EDA)

The Exploratory data analysis of the dataset will help us make some preliminary conclusions about the dataset and will help in visualising the data better. The dataset is then split into 2 dataframes - X : Input data, and its corresponding y : target data. Both X and y are further split into 80% training and 20% testing data which are stratified along y (X_train, X_test, y_train, y_test). EDA is conducted to both the training and testing sets to check the similarities between the datasets while at the same time making initial references.

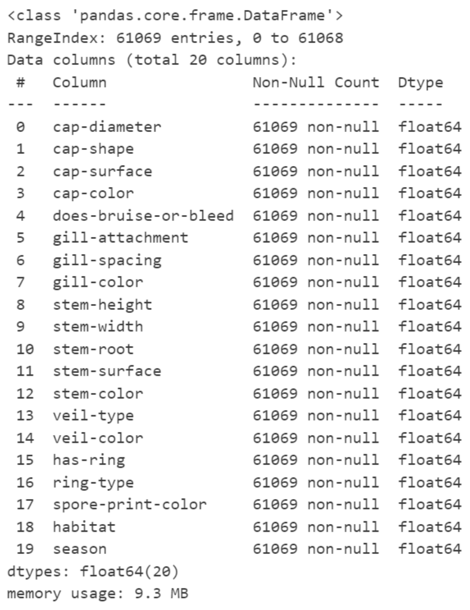
Data Cleaning: (of the whole dataset)

Info if the dataset provided :



```
<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
```

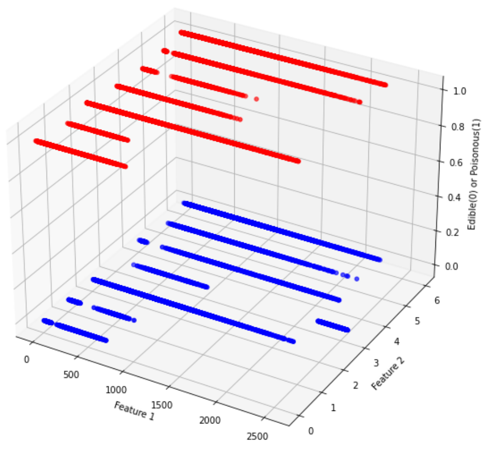
Most of the columns of the data are of object data type, and many columns have null values. Thus, the data has to be cleaned to work with it. All the columns of the input data are converted from object to numeric data type using ordinal encoder while label encoder is used for the target data. The null values present in each column of the input dataset are filled with the median of its respective columns. The dataset is then transformed into a numeric (16,069 x 21) dataset with no null values :



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61069 entries, 0 to 61068
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   cap-diameter          61069 non-null  float64
 1   cap-shape             61069 non-null  float64
 2   cap-surface           61069 non-null  float64
 3   cap-color             61069 non-null  float64
 4   does-bruise-or-bleed  61069 non-null  float64
 5   gill-attachment       61069 non-null  float64
 6   gill-spacing          61069 non-null  float64
 7   gill-color            61069 non-null  float64
 8   stem-height           61069 non-null  float64
 9   stem-width            61069 non-null  float64
 10  stem-root             61069 non-null  float64
 11  stem-surface          61069 non-null  float64
 12  stem-color            61069 non-null  float64
 13  veil-type             61069 non-null  float64
 14  veil-color            61069 non-null  float64
 15  has-ring              61069 non-null  float64
 16  ring-type             61069 non-null  float64
 17  spore-print-color     61069 non-null  float64
 18  habitat               61069 non-null  float64
 19  season                61069 non-null  float64
dtypes: float64(20)
memory usage: 9.3 MB
```

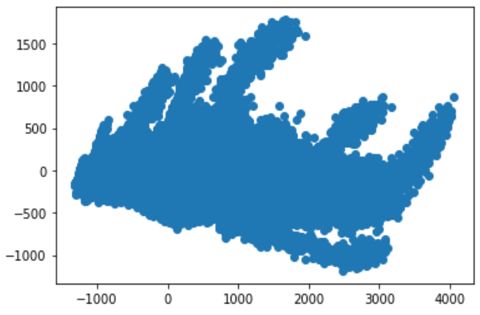
Sample EDA of the dataset:

[1] Visualising the relation between 2 inputs with respect to the target: (Manual Selection)

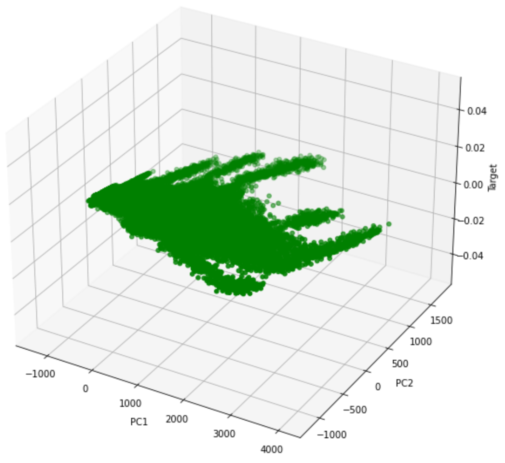


After Principal Component Analysis (PCA), the input set is converted to a 2 dimensional dataset (from a 20 dimensional data) enabling to visualise better :

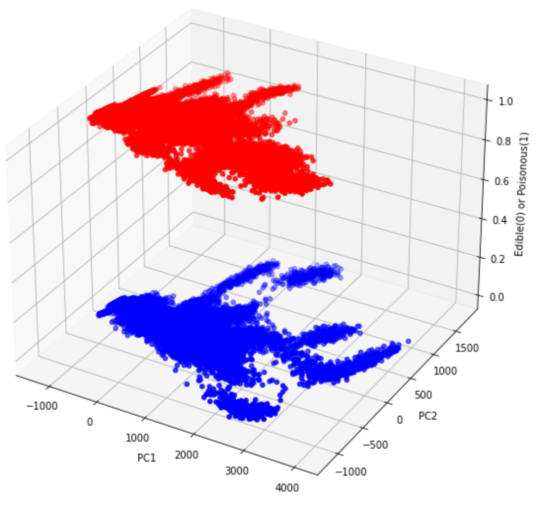
[2] PCA: Relation b/w features (2D)



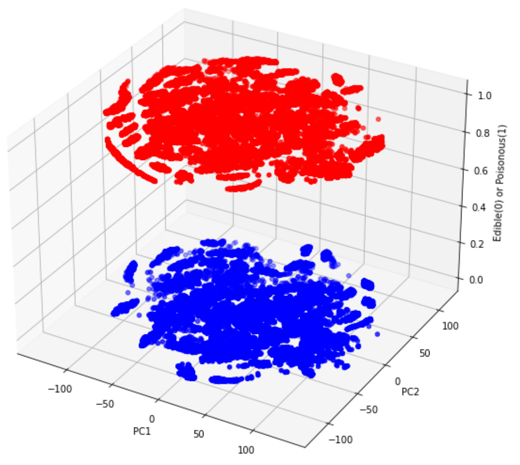
[3] PCA: Relation b/w features (3D)



[4] PCA: Relation b/w features with respect to target



[5] TSNE: Relation b/w features with respect to target



From [2], [3], [4], and [5], it is noticed that the relationship between the input variables and target is not a linearly separable classification.

3. Methods

Following are the various methods used in this project.

3.1 Baseline - Logistic Regression

- Brief description of the method :
  Logistic Regression is a supervised machine learning algorithm that predicts the probability of a certain instance belonging to a particular class. Logistic regression predicts the probability of a new instance of mushroom being definitely poisonous or definitely edible.

3.2 Support Vector Machine

- Description for Linear SVM:
  Linear SVM finds a decision boundary between two classes with the help of the support vectors. The support vectors decide the extent of margin between 2 classes and, the decision boundary which lies in between this margin decides the misclassification. Linear SVM should maximise margin and minimise misclassification. A decision boundary is drawn between classes - definitely edible and definitely poisonous, with the help of support vectors.

- Description for Kernel SVM classifier:
  Kernel SVM is used when the point cloud is not linearly separable. As SVM is natively linear, the point cloud should be converted such that it can be separated using a hyperplane.

  Linear Kernel:
  Linear Kernel SVM converts the 20 dimensional input data to higher (20+i) dimension space using a linear function. SVM is operated in this (20+i) dimensional plane and finds a hyperplane that separates the 2 classes with maximised margins and minimised misclassification using support vectors.

  Polynomial Kernel:
  Polynomial Kernel SVM converts the 20 dimensional input data to higher (20+i) dimensional space using a polynomial function (n-degree). SVM is operated in this (20+i) dimensional plane and finds a hyperplane that separates the 2 classes with maximised margins and minimised misclassification using support vectors.

  RBF (Radial Basis Function) Kernel:
  RBF Kernel SVM converts a 20 dimensional data to a higher (20+i) dimensional space using a gaussian function (-(||x(i) - x(j)||)/2(sigma)^2). SVM is operated in this (20+i) dimensional plane and finds a hyperplane that separates the 2 classes with maximised margins and minimised misclassification using support vectors.

3.3 Decision Trees

- Description for Decision Tree Classifier :
  Decision Tree is a non-parametric supervised machine learning algorithm which breaks down a complex dataset into smaller subsets, forming a binary tree with each branch being a possible outcome. Given a user-specified depth, the decision tree is formed such that a pure node arrives as soon as possible (Gini = 0). The decision tree algorithm stops when all the leaf nodes of the binary tree are pure. (Gini : Measure of impurity in a node)

3.4 Ensemble Learning

- Description for Random Forest Classifier :
  Random forest classifier is an ensemble learning algorithm that contains many decision trees on random subsets of the original data. The mode of all outputs is considered to improve the predictive accuracy of the dataset. This algorithm provides for a much stronger prediction than a single decision tree for the whole dataset.

- Description for AdaBoost Classifier :
  AdaBoost classifier is an ensemble learning algorithm that combines several weak learners to obtain a strong learner. AdaBoost is a sequential learning algorithm where each new predictor corrects its predecessor's misclassification instances. Thus, each new learner focuses on harder instances than its predecessor.

- Description for Gradient Boost Classifier :
  Gradient Boost Classifier is an ensemble learning algorithm that combines several weak learners to obtain a strong learner. Gradient Boost is a sequential learning algorithm where each new predictor corrects its predecessor's uncertainty measure. Thus, each new learner minimises their uncertainty compared to their predecessor.

4. Experiments & Results

4.1 Protocol

- Details about splitting into training and testing datasets:
  The dataset was split into 80% training and 20% testing sets to implement the respective machine learning models.

- Preprocessing of the dataset:
  - Feature Scaling:
    Standardisation : Experiments were conducted with a standardised input data, where the features were transformed to have mean=0 and std=1.

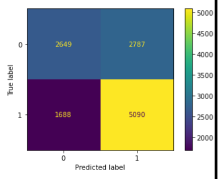
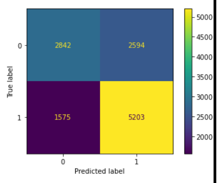
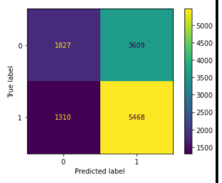
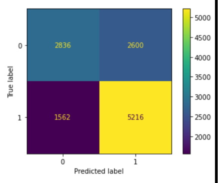
    $$X' = (X - mean)/ std$$

  - Feature Selection:
    Select Percentile : The input dataset retained the user specified highest scoring percentage of features (80%). The score is determined by the correlation between the feature and output data.

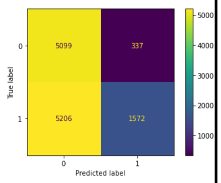
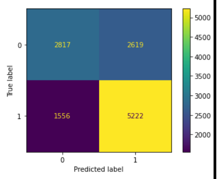
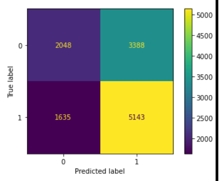
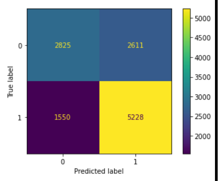
  - Feature Reduction:
    Principal Component Analysis (PCA) : The 20 dimensional input data was converted to 2 (or 10) dimensional data using PCA. PCA reduces the dimensions of the input data while maintaining the spread of the data. Experiments were performed with this dataset.

4.2 Results

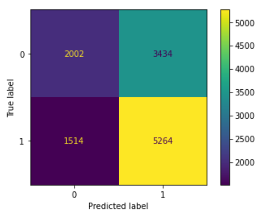
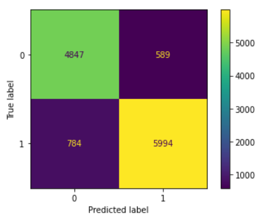
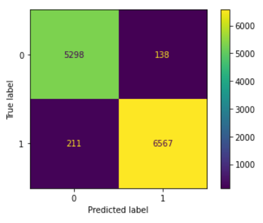
- Baseline results using Logistic regression:

Logistic Regression	Raw Data	Standardised Data	PCA data (2 comp.)	Select Percentile Data (80%)
Accuracy Score	0.63	0.66	0.60	0.66
Precision	0.65	0.67	0.60	0.67
Recall	0.75	0.77	0.81	0.77
F1 Score	0.69	0.71	0.69	0.71
Cross Validation : F1	0.69 +/- 0.0023	0.71 +/- 0.0035	0.69 +/- 0.0047	0.71 +/- 0.0034
Confusion Matrix				

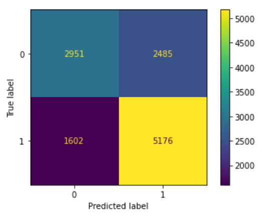
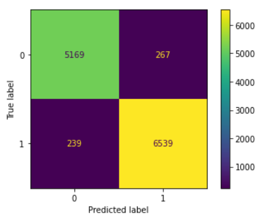
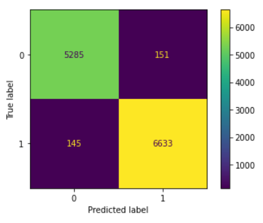
- Results based on various LinearSVM classification:

Linear SVC	Raw Data	Standardised Data	PCA Data (2 comp)	Select Percentile Data (80%)
Accuracy Score	0.55	0.66	0.59	0.66
Precision	0.82	0.67	0.60	0.67
Recall	0.23	0.77	0.76	0.77
F1 score	0.36	0.71	0.67	0.72
Cross Validation : F1	0.42 +/- 0.2190	0.71 +/- 0.0034	0.67 +/- 0.0046	0.71 +/- 0.0033
Confusion Matrix				

- Results based on PCA Kernel SVM (10 comp.) :

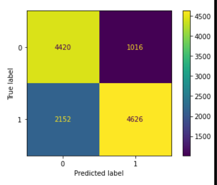
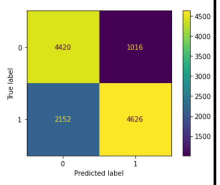
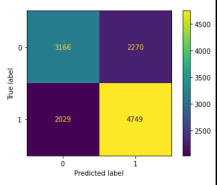
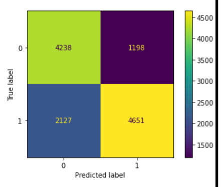
Kernel SVM	Linear K SVM	Polynomial K SVM	RBF K SVM
Accuracy Score	0.59	0.89	0.97
Precision	0.61	0.91	0.98
Recall	0.78	0.88	0.97
F1 Score	0.68	0.90	0.97
Cross Validation : F1	0.68 +/- 0.0039	0.89 +/- 0.0030	0.97 +/- 0.0016
Confusion Matrix			

- Results based on Select Percentile Kernel SVM (80%) :

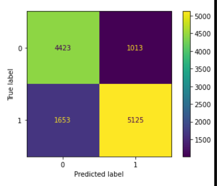
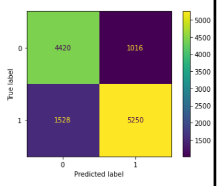
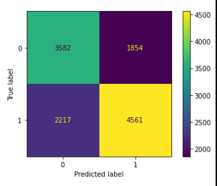
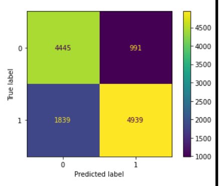
Kernel SVM	Linear K SVM	Polynomial K SVM	RBF K SVM
Accuracy Score	0.67	0.96	0.98
Precision	0.68	0.96	0.98
Recall	0.76	0.96	0.98
F1 Score	0.72	0.96	0.98
Cross Validation : F1	0.72 +/- 0.0038	0.96 +/- 0.0012	0.97 +/- 0.0011
Confusion Matrix			

- Results based on Decision Tree Classifier :

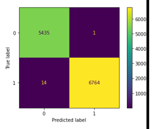
Decision Tree Classifier	Raw Data	Standardised Data	PCA data (2 comp.)	Select Percentile Data (80%)
Accuracy Score	0.74	0.74	0.65	0.73

Decision Tree Classifier	Raw Data	Standardised Data	PCA data (2 comp.)	Select Percentile Data (80%)
Precision	0.82	0.82	0.68	0.80
Recall	0.68	0.68	0.70	0.69
F1 Score	0.74	0.74	0.69	0.74
Cross Validation : F1	0.74 +/- 0.0136	0.74 +/- 0.0136	0.69 +/- 0.0080	0.74 +/- 0.0047
Confusion Matrix				

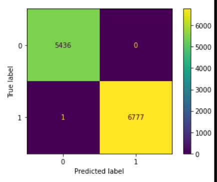
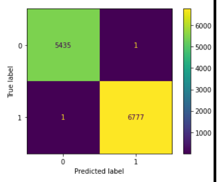
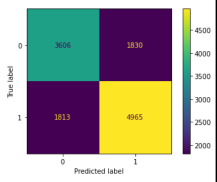
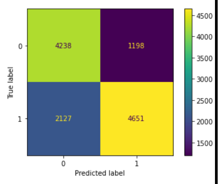
- Results based on Random Forest Classifier :

Random Forest Classifier	Raw Data	Standardised Data	PCA data (2 comp.)	Select Percentile Data (80%)
Accuracy Score	0.78	0.79	0.67	0.77
Precision	0.83	0.84	0.71	0.83
Recall	0.76	0.77	0.67	0.73
F1 Score	0.79	0.80	0.69	0.78
Cross Validation : F1	0.81 +/- 0.0112	0.81 +/- 0.0136	0.69 +/- 0.0066	0.77 +/- 0.0072
Confusion Matrix				

- Results based on Hyperparameter tuning applied on Random Forest standardised data : {n_estimators = 500, max_depth = 14}

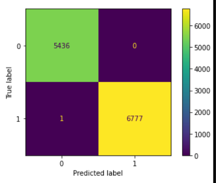
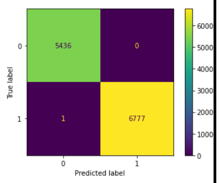
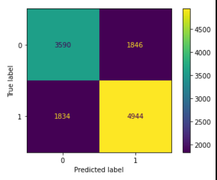
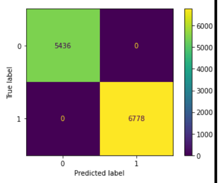
RF Grid Search CV	Accuracy	Precision	Recall	F1 Score	CV : F1	Confusion Matrix
Metrics	0.99	1.00	1.00	1.00	0.99 +/- 0.0003	

- Results based on AdaBoost Classifier :

AdaBoost Classifier	Raw Data	Standardised Data	PCA data (2 comp.)	Select Percentile Data (80%)
Accuracy Score	0.99	0.99	0.70	0.99
Precision	1.00	1.00	0.73	1.00
Recall	1.00	1.00	0.73	1.00
F1 Score	1.00	1.00	0.73	1.00
Cross Validation : F1	0.99 +/- 7.3773	0.99 +/- 7.3773	0.73 +/- 0.0025	0.99 +/- 0.0001
Confusion Matrix				

- Results based on Gradient Boost Classifier :

Gradient Boost Classifier	Raw Data	Standardised Data	PCA data (2 comp.)	Select Percentile Data (80%)
Accuracy Score	0.99	0.99	0.70	1.00
Precision	1.00	1.00	0.73	1.00
Recall	1.00	1.00	0.73	1.00
F1 Score	1.00	1.00	0.73	1.00
Cross Validation : F1	0.99 +/- 0.0001	0.99 +/- 0.0001	0.73 +/- 0.0028	0.99 +/- 0.0001

Gradient Boost Classifier	Raw Data	Standardised Data	PCA data (2 comp.)	Select Percentile Data (80%)
Confusion Matrix				

5. Discussion

- The data has been trained using various machine learning models like : Logistic Regression, Linear SVM, Kernel SVM, Decision Trees, Random Forest, AdaBoost and Gradient Boost.
- The Classification of the point cloud was not linearly separable.
- Experiments were conducted with Raw, standardised, PCA and select percentile data. Feature selection conducted by select percentile (80% of features) performed the best overall.
- Hyperparameter tuning using grid search was applied to random forest classification with standardised data. The accuracy and f1 score increased significantly after this experiment, as shown above.
- The gradient boosting and adaboost algorithm performed the best classification with an accuracy of 1.00 on select percentile data and standardised data respectively.
- Random forest with hyperparameter tuning using grid search cv performed second best with an accuracy of 0.998 on standardised data.
- Experiments with kernel SVM RBF and poly performed third best with an accuracy of 0.98 and 0.96 respectively with select percentile data.
- Experiments with PCA (2 comp) performed poorly in all machine learning models.

6. Conclusion

- The goal of the secondary mushrooms classification dataset was to classify whether a mushroom was poisonous or edible based on their characteristics. A dataset containing 61,069 instances was trained using various machine learning algorithms to obtain minimum misclassification in the testing set. Gradient boost and AdaBoost performed the best across all techniques with 0 misclassification and accuracy of 1.0. All linear classification algorithms produced poor results as the point cloud was not linearly separable. Thus, this project proved that a mushroom can be classified based on their characteristics into poisonous and edible.

7. References

[1] UCI Machine Learning Repository: Secondary Mushroom Dataset Data Set

[2] Mushroom data creation, curation, and simulation to support classification tasks - PMC