Mushroom Classification using Multiple Machine Learning Models

Introduction

The objective of this project is to classify mushrooms as 'edible' or 'poisonous', using different machine learning models, based on various physical features of mushrooms such as their cap shape and color, gill size and color, odor, bruises, population, etc.

To carry out this task in Python some essential libraries that will be used are numpy and pandas for data handling and preprocessing, sklearn (scikit-learn) for machine learning, and matplotlib and seaborn for data visualization.

The dataset used, that was obtained from Kaggle, contains 8124 instances of mushrooms with 22 physical characteristics and a binary target/class label – 'p' for poisonous and 'e' for edible.

```
class
e    4208
p    3916
```

As seen above, out of the 8124 rows we have 4208 instances of edible mushrooms and 3916 instances of poisonous mushrooms. Even though it's not a 50-50 collection, the dataset is not particularly unbalanced. The images below show a glimpse of the dataset.

class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	stalk-shape	stalk-root	stalk-surface-above-ring	s
p	x	s	n	t	p	f	c	n	k	e	e	s	s
e	x	s	y	t	a	f	c	b	k	e	c	s	s
e	b	s	w	t	l	f	c	b	n	e	c	s	s
p	x	y	w	t	p	f	c	n	n	e	e	s	s
e	x	s	g	f	n	f	w	b	k	t	e	s	s

stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	ring-number	ring-type	spore-print-color	population	habitat
s	w	w	p	w	o	p	k	s	u
s	w	w	p	w	o	p	n	n	g
s	w	w	p	w	o	p	n	n	m
s	w	w	p	w	o	p	k	s	u
s	w	w	p	w	o	e	n	a	g

All the feature columns have values of the 'object' datatype, i.e., all are categorical features. Below are all the unique values for a few of the columns in the dataset –

```
class -
Number of Unique values: 2
Unique values: ['p' 'e']

cap-shape -
Number of Unique values: 6
Unique values: ['x' 'b' 's' 'f' 'k' 'c']

cap-surface -
Number of Unique values: 4
Unique values: ['s' 'y' 'f' 'g']
```

```
cap-color -
Number of Unique values: 10
Unique values: ['n' 'y' 'w' 'g' 'e' 'p' 'b' 'u' 'c' 'r']

bruises -
Number of Unique values: 2
Unique values: ['t' 'f']

odor -
Number of Unique values: 9
Unique values: ['p' 'a' 'l' 'n' 'f' 'c' 'y' 's' 'm']
```

```
stalk-root -
Number of Unique values: 5
Unique values: ['e' 'c' 'b' 'r' '?']

stalk-surface-above-ring -
Number of Unique values: 4
Unique values: ['s' 'f' 'k' 'y']

stalk-surface-below-ring -
Number of Unique values: 4
Unique values: ['s' 'f' 'y' 'k']
```

```
stalk-color-above-ring -
Number of Unique values: 9
Unique values: ['w' 'g' 'p' 'n' 'b' 'e' 'o' 'c' 'y']

stalk-color-below-ring -
Number of Unique values: 9
Unique values: ['w' 'p' 'g' 'b' 'n' 'e' 'y' 'o' 'c']

veil-type -
Number of Unique values: 1
Unique values: ['p']
```

Although there were no NAN values in the dataset as such, from observing the unique values for each column, a value for the column stalk-root was seen to be '?' which clearly represents a missing value. Secondly, the column 'veil-type' seems to have just one value in the entire dataset, while all other columns had 2 or more, non-null categorical values.

Data Preprocessing

On checking the number of rows that have '?' as value for 'stalk-root', 2480 rows were flagged. Since that's quite a large number of rows, instead of removing those rows entirely or approximating a value based on other columns (such as by using voting) , we will simply eliminate this column and make do with the remaining features for predicting the class of the mushrooms.

As 'veil-type' has the same value for all instances of mushrooms in the dataset, so, it is a redundant column and will not be helpful in the classification. Thus, this column is also removed.

In order to proceed, these categorical features are first converted to numerical form. For this LabelEncoder() from sklearn.preprocessing module is used, which ensures that the values are numerically encoded based on the lexical order of unique values. So, the target column – class – has its values changed from 'e' and 'p' to 0 and 1.

Feature Selection

The initial 22 features are now reduced 20 features, due to redundancy and missing values.

Feature Importances

For gaining a preliminary insight regarding the importance of each of those 20 features when classifying the mushrooms, 80:20 splitting is done on the dataset (for training and testing, respectively) and a Random Forest Classifier is fitted. Hyperparameter tuning is also done using GridSearchCV to ensure that the results are not tainted due to poor choice of hyperparameters. Using rf.feature_importances_ (where rf is the fitted Random Forest Classifier), we get the importances of all the features with respect to classification. The table below shows these feature importances in decreasing order of importance –

	feature	importance
4	odor	0.299705
7	gill-size	0.154159
8	gill-color	0.140350
17	spore-print-color	0.088949
18	population	0.049912
10	stalk-surface-above-ring	0.045746
3	bruises	0.043066
11	stalk-surface-below-ring	0.036257
19	habitat	0.028595
6	gill-spacing	0.028435
9	stalk-shape	0.019359
2	cap-color	0.013530
13	stalk-color-below-ring	0.010412
15	ring-number	0.009832
16	ring-type	0.007502
12	stalk-color-above-ring	0.006503
0	cap-shape	0.005997
5	gill-attachment	0.005105
1	cap-surface	0.004635
14	veil-color	0.001950

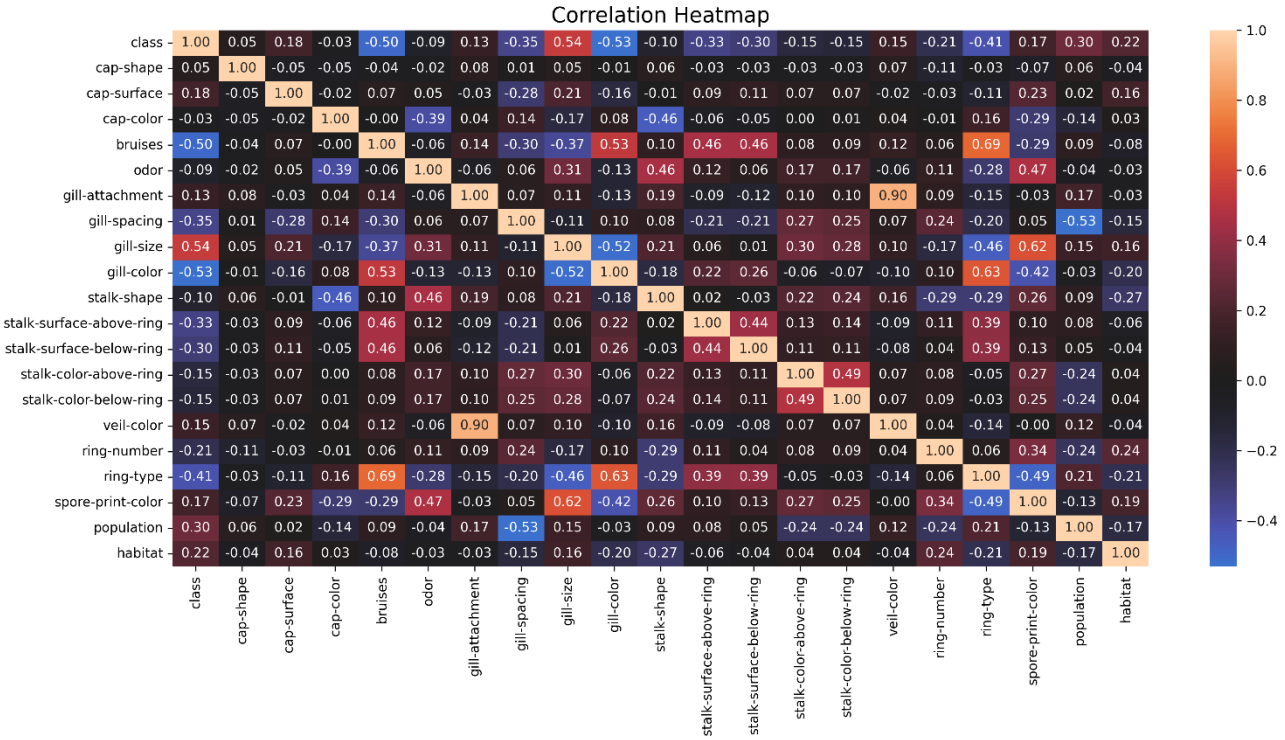
Based on these scores, ‘odor’ is the most important feature when it comes to predicting the class of mushrooms, followed by ‘gill-size’ and ‘gill-color’, while ‘gill-attachment’, ‘cap-surface’, and ‘veil-color’ have minimum relevance. In order to determine the minimum number of features needed to accurately and precisely classify the mushrooms, we will use brute-force approach and train new Random Forest Classifiers on subsets of the features (which are sorted in descending order of their importances). We will start with one feature - ‘odor’ – train and evaluate the model, and then iteratively add one more feature at a time and train and evaluate another model. This will be repeated until all 20 features have been added and we have the performance metrics of all 20 models. The results of this are shown below –

```
Accuracy of Random Forest with top 1 features: 0.9846153846153847
Accuracy of Random Forest with top 2 features: 0.9846153846153847
Accuracy of Random Forest with top 3 features: 0.9883076923076923
Accuracy of Random Forest with top 4 features: 0.9932307692307693
Accuracy of Random Forest with top 5 features: 0.9944615384615385
Accuracy of Random Forest with top 6 features: 1.0
Accuracy of Random Forest with top 7 features: 1.0
Accuracy of Random Forest with top 8 features: 1.0
Accuracy of Random Forest with top 9 features: 1.0
Accuracy of Random Forest with top 10 features: 1.0
Accuracy of Random Forest with top 11 features: 1.0
Accuracy of Random Forest with top 12 features: 1.0
Accuracy of Random Forest with top 13 features: 1.0
Accuracy of Random Forest with top 14 features: 1.0
Accuracy of Random Forest with top 15 features: 1.0
Accuracy of Random Forest with top 16 features: 1.0
Accuracy of Random Forest with top 17 features: 1.0
Accuracy of Random Forest with top 18 features: 1.0
Accuracy of Random Forest with top 19 features: 1.0
Accuracy of Random Forest with top 20 features: 1.0
```

Clearly, with the top 6 features alone we are getting 100% accuracy, which means these features represent the rest of the feature-space well enough. Therefore, using all the features for classifying the mushrooms is redundant. Those 6 most important features are – ‘odor’, ‘gill-size’, ‘gill-color’, ‘spore-print-color’, ‘population’, and ‘stalk-surface-above-ring’.

Correlation

Furthermore, to get a better idea about the relevance of each feature with respect to the class of the mushroom, a heatmap is plotted to visualize their correlation with the target.



From the ‘class’ row/column in the heatmap we can see that the features ‘gill-size’, ‘gill-color’, and ‘bruises’ are most correlated with the target, followed by ‘ring-type’, ‘gill-spacing’, ‘stalk-surface-above-ring’, ‘stalk-surface-below-ring’, and ‘population’. Here, all except ‘gill-size’ and ‘population’ are negatively correlated with ‘class’. Since, correlation can take both positive and negative values, the absolute values will be considered for the feature selection (magnitudes closer to 1 mean highly correlated while closer to 0 mean less correlated, so, the signs don’t matter). The table below shows the absolute correlations of all 20 features with ‘class’, in descending order of their absolute correlation values –

	feature	absolute correlation
7	gill-size	0.540024
8	gill-color	0.530566
3	bruises	0.501530
16	ring-type	0.411771
6	gill-spacing	0.348387
10	stalk-surface-above-ring	0.334593
11	stalk-surface-below-ring	0.298801
18	population	0.298686
19	habitat	0.217179
15	ring-number	0.214366
1	cap-surface	0.178446
17	spore-print-color	0.171961
12	stalk-color-above-ring	0.154003
13	stalk-color-below-ring	0.146730
14	veil-color	0.145142
5	gill-attachment	0.129200
9	stalk-shape	0.102019
4	odor	0.093552
0	cap-shape	0.052951
2	cap-color	0.031384

‘odor’ which was considered most important feature based on the fitted Random Forest Classifier, is one among the least correlated with ‘class’. ‘gill-size’ and ‘gill-color’ are the top highly correlated features, which also aligns with their high importance scores.

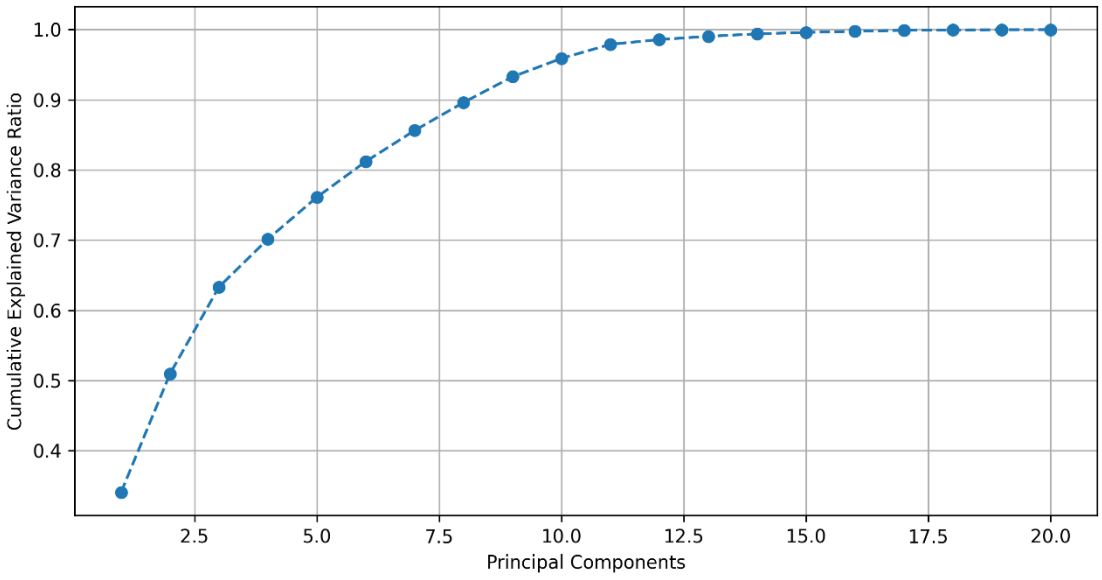
In order to determine the minimum number of highly correlated features required to accurately and precisely classify the mushrooms, we will use the same brute-force approach that was used earlier with the feature importance scores. The results of the iterative model training and evaluation are shown below –

```
Accuracy of Random Forest with top 1 features: 0.7409230769230769
Accuracy of Random Forest with top 2 features: 0.8326153846153846
Accuracy of Random Forest with top 3 features: 0.8652307692307692
Accuracy of Random Forest with top 4 features: 0.9372307692307692
Accuracy of Random Forest with top 5 features: 0.9581538461538461
Accuracy of Random Forest with top 6 features: 0.9716923076923077
Accuracy of Random Forest with top 7 features: 0.976
Accuracy of Random Forest with top 8 features: 0.9864615384615385
Accuracy of Random Forest with top 9 features: 0.9993846153846154
Accuracy of Random Forest with top 10 features: 0.9993846153846154
Accuracy of Random Forest with top 11 features: 0.9993846153846154
Accuracy of Random Forest with top 12 features: 1.0
Accuracy of Random Forest with top 13 features: 1.0
Accuracy of Random Forest with top 14 features: 1.0
Accuracy of Random Forest with top 15 features: 1.0
Accuracy of Random Forest with top 16 features: 1.0
Accuracy of Random Forest with top 17 features: 1.0
Accuracy of Random Forest with top 18 features: 1.0
Accuracy of Random Forest with top 19 features: 1.0
Accuracy of Random Forest with top 20 features: 1.0
```

We can see that at least 12 of the highly correlated features are required together, to facilitate 100% accurate classification of mushrooms. This implies that the fitted Random Forest Classifier gives us a more well-rounded assessment of feature importance, in regard to classifying the mushrooms, than the correlation values. Therefore, we are able to get a much better combination of most relevant features from the former.

PCA

Lastly, we also explore Principal Component Analysis for feature selection via dimensionality reduction. The plot below shows the cumulative explained variance by the principal components.



This scree plot doesn't have any clear 'elbow' point, but we can notice a slight decline in slope after the 3rd PC, a much slower rate after 8th PC, and a near stagnation after the 14th PC. The first three principal components explain more than 60% of the variance in the data, while the first eight

explain nearly 90% of the variance, and it takes at least 14 PCs to retain 100% of the variance. The same brute-force approach as earlier is used to determine the minimum number of components required to accurately classify the mushrooms. The obtained results mirrored the interpretation of the scree plot, with at least 14 PCs being needed to classify the mushrooms with 100% accuracy. The detailed results of the model training and evaluations are shown below –

```
Accuracy of Random Forest after PCA - 1D: 0.6553846153846153
Accuracy of Random Forest after PCA - 2D: 0.9673846153846154
Accuracy of Random Forest after PCA - 3D: 0.9913846153846154
Accuracy of Random Forest after PCA - 4D: 0.9883076923076923
Accuracy of Random Forest after PCA - 5D: 0.9950769230769231
Accuracy of Random Forest after PCA - 6D: 0.9950769230769231
Accuracy of Random Forest after PCA - 7D: 0.9975384615384615
Accuracy of Random Forest after PCA - 8D: 0.9975384615384615
Accuracy of Random Forest after PCA - 9D: 0.9969230769230769
Accuracy of Random Forest after PCA - 10D: 0.9981538461538462
Accuracy of Random Forest after PCA - 11D: 0.9963076923076923
Accuracy of Random Forest after PCA - 12D: 0.9981538461538462
Accuracy of Random Forest after PCA - 13D: 0.9981538461538462
Accuracy of Random Forest after PCA - 14D: 1.0
Accuracy of Random Forest after PCA - 15D: 1.0
Accuracy of Random Forest after PCA - 16D: 1.0
Accuracy of Random Forest after PCA - 17D: 1.0
Accuracy of Random Forest after PCA - 18D: 1.0
Accuracy of Random Forest after PCA - 19D: 1.0
Accuracy of Random Forest after PCA - 20D: 1.0
```

Among all variations, we can see that the best Random Forest classifiers were those that were fitted on all the features, the top 6 features based on importances given by the Random Forest classifier (after hyperparameter tuning), the 12 most correlated features, or the 14 components. Since PCA is not helping us get a significant dimensionality reduction whilst maintaining high accuracy and other metrics, we will not use it any further. Similarly, since we have a possibility of getting 100% accurate and precise classification with just 6 features, we will not proceed with 12 highly correlated features either. So, to see how well other classifiers work, we will use 2 datasets - one with all 20 features and another with only those top 6 features.

Models Used

The below models were trained and tested on the 2 pre-processed datasets (with 20 and 6 features).

1. Logistic Regression
2. Softmax Regression
3. Decision Tree
4. Bagging Classifier
5. Pasting Classifier
6. Random Forest Classifier
7. AdaBoost Classifier
8. Gradient Classifier

9. Hard Voting Classifier
10. Soft Voting Classifier
11. Stacking Classifier
12. Linear Kernel SVC
13. Polynomial Kernel SVC
14. RBF Kernel SVC

Results & Observations

In summary, all the models that involved decision trees in some manner, such as decision trees themselves or ensemble learning models using decision trees as base estimators, were the models that performed best and gave 100% accurate and precise classification, for both datasets. Other than the tree-based models, support-vector classifier using polynomial kernel (degree 5) also gave 100% accuracy and precision in classifying the mushrooms, for both datasets, On the other hand, SVC with RBF kernel gave 100% accuracy and precision while classifying using the smaller dataset (6 features), while giving 100% precision but only 98% accuracy with the of bigger dataset (20 features). Logistic Regression, Softmax Regression, and SVC with Linear Kernels, couldn't meet the performance levels of the other models even after hyperparameter tuning, and their performances declined even more when using the dataset of smaller feature-space.

Conclusion

Since, tree-based models gave the best results, Decision Tree Classifier could be the best option among all others as it's the least computationally intensive algorithm, least complex and also highly explainable. Although, support-vector classifier with the polynomial kernel also performed equally well, when thinking about applying these models in a real-life scenario, where, it is likely for non-ML-experts to also be involved, a simpler and easily interpretable model that is able to perform with high accuracy and precision (and other performance metrics as well) should be preferred.