Mushroom Classification Project

LEENA GODBOLE

This is the submission for the graduate student project for Leena Godbole in CSC 410.

ACM Reference Format:
Leena Godbole. 2024. Mushroom Classification Project. *ACM Trans. Graph.* 37, 4, Article 111 (August 2024), 3 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The mushroom dataset, found on the UC Irvine Machine Learning Repository, has 8124 observations, and labels edible vs poisonous mushrooms based on the feature space shown in table 1. In the following paper various machine learning techniques will be applied to this dataset in order to classify the edible versus poisonous label of the mushroom.

2 DATA PREPARATION

The data was downloaded directly from the UC Irvine Machine Learning repository using the ucimlrepo. Early exploration into the data includes checking for a balance in the labels (edible v poisonous) and for missing datas. In figure 2, it can be seen that the poisonous versus edible mushrooms do have a small discrepancy, but not large enough to merit any bias towards one class or another. No downsampling of the majority class was deemed necessary.

The 'stalk-root' feature did have some missing data values, 30.5 percent of the data was NANs. This lead to the decision to drop the 'stalk-root' column entirely from the feature space, rather than imputing.

Note, all the features in this data set are categorical data, some even just binary categories. Additionally all of the data types described in figure 1 are nominal, rather than ordinal. This resulted in one hot encoding, rather than regular encoding to be used.

One hot encoding was used on the label set with size [:1]. Poisonous became class [1] and edible became class [0]. One hot encoding was also done on the feature space X, which changed the data shape from [8124, 22] to [8124, 112].

The data was then split into a train and test sets, stratifying on the label with an 80/20 split.

3 MODELING

3.1 Support Vector Machines on Full Feature Space

An SVM model was created with a linear kernel function in order to classify the mushroom dataset. The model was trained with all of the feature data. The confusion matrix for the test data can be seen in figure 3.

Variable Name	Role	Type	Description
poisonous	Target	Binary	poisonous = p, edible = e
cap-shape	Feature	Categorical	bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s
cap-surface	Feature	Categorical	fibrous=f,grooves=g,scaly=y,smooth=s
cap-color	Feature	Categorical	brown=n,buff=b,cinnamon=c,gray=g,green=r, pink=p,purple=u,red=e,white=w,yellow=y
bruises	Feature	Categorical	bruises=t,no=f
odor	Feature	Categorical	almond=a,anise=l,creosote=c,fishy=y,foul=f, musty=m,none=n,pungent=p,spicy=s
gill-attachment	Feature	Categorical	attached=a,descending=d,free=f,notched=n
gill-spacing	Feature	Categorical	close=c,crowded=w,distant=d
gill-size	Feature	Categorical	broad=b,narrow=n
gill-color	Feature	Categorical	black=k,brown=n,buff=b,chocolate=h,gray=g, green=r,orange=o,pink=p,purple=u,red=e, white=w,yellow=y
stalk-shape	Feature	Categorical	enlarging=e,tapering=t
stalk-root	Feature	Categorical	bulbous=b,club=c,cup=u,equal=e, rhizomorphs=z,rooted=r,missing=?
stalk-surface-above-ring	Feature	Categorical	fibrous=f,scaly=y,silky=k,smooth=s
stalk-surface-below-ring	Feature	Categorical	fibrous=f,scaly=y,silky=k,smooth=s
stalk-color-above-ring	Feature	Categorical	brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y
stalk-color-below-ring	Feature	Categorical	brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y
veil-type	Feature	Binary	partial=p,universal=u
veil-color	Feature	Categorical	brown=n,orange=o,white=w,yellow=y
ring-number	Feature	Categorical	none=n,one=o,two=t
ring-type	Feature	Categorical	cobwebby=c,evanescent=e,flaring=f,large=l, none=n,pendant=p,sheathing=s,zone=z
spore-print-color	Feature	Categorical	black=k,brown=n,buff=b,chocolate=h,green=r, orange=o,purple=u,white=w,yellow=y
population	Feature	Categorical	abundant=a,clustered=c,numerous=n, scattered=s,several=v,solitary=y
habitat	Feature	Categorical	grasses=g,leaves=l,meadows=m,paths=p, urban=u,waste=w,woods=d

Fig. 1. Table 1: Feature Space Summary

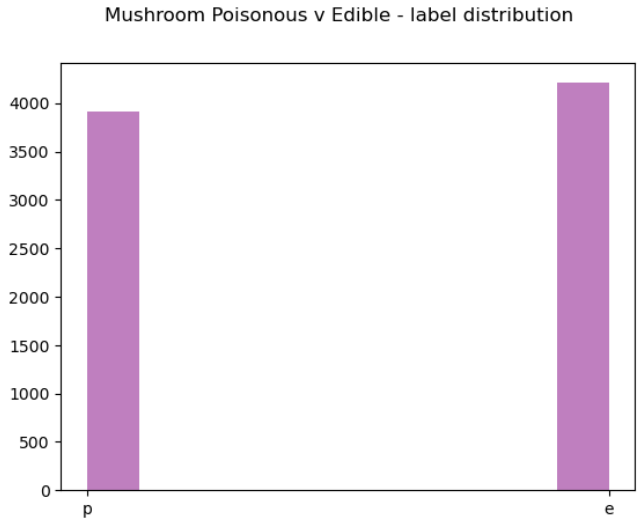

Fig. 2. Label Distribution - Poisonous v Edible

3.2 Random Forests on Full Feature Space

Random Forest Classifiers were done on the full feature space to predict the class labels of edible [0] vs poisonous [1]. It was run with the following number of trees in each forest: 50, 25, 10, and 5. All of the full random forest models had an accuracy and precision of 100

Author's address: Leena Godbole, lngodbole@uncg.edu.

ACM 0730-0301/2024/8-ART111
https://doi.org/XXXXXXX.XXXXXXX

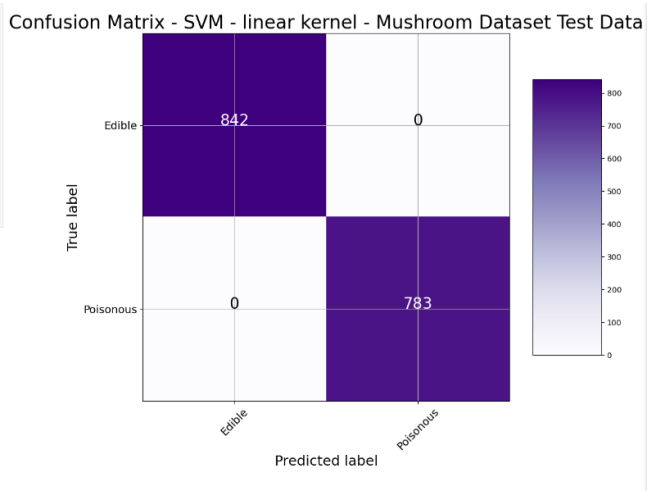

Fig. 3. Confustion Matrix - SVM Linear Kernel

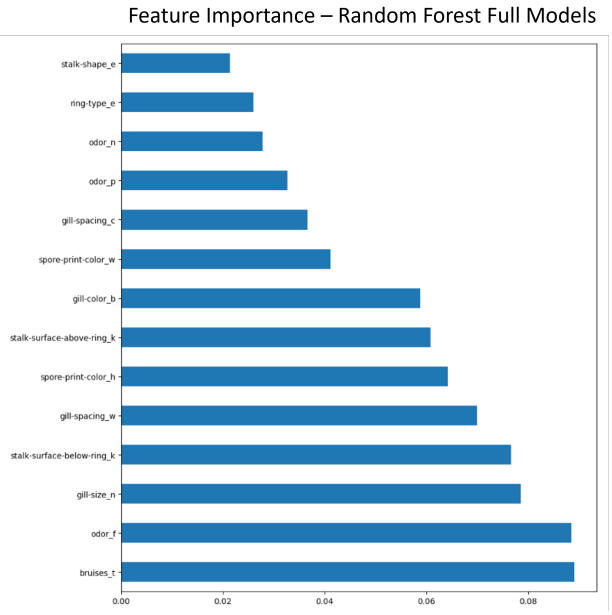

Fig. 4. Feature Importance from Full RF

percent on the test datasets. Since the model could predict the test class labels without mislabeling any data, the confusion matrices will not be show. In figure 4, you can see the feature importance from these models. Bruises present (True), foul odor, narrow gill size, silky stalk surface below ring, crowded gill spacing were the strongest features for classification. However, together they only contribute to a fraction of the variation in the model, many more features are needed than the top 5 to classify well.

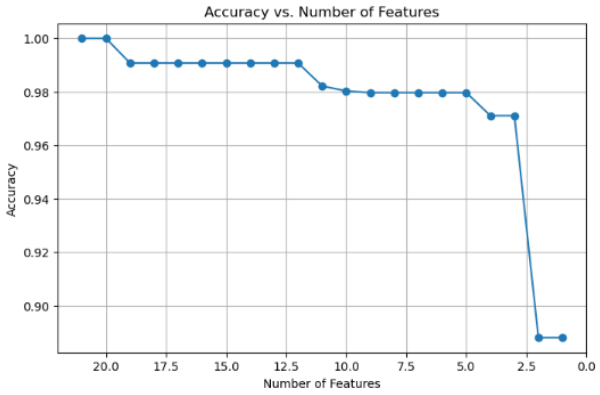

Fig. 5. Accuracy v Number of Features - RF

3.3 Random Forests - Feature Selection

The random forest classifier was repeated, leaving out low performing features. Features were left out according to the feature importance output from the full random forest models. A plot of accuracy versus the number of features can be seen in figure 5. All models were run with the default setting of 10 trees per forest.

The accuracy of the random forest models starts to drop of with the number of features less than 20. Keep in mind, this is on the encoded dataset, with 112 features total, therefor only 18 percent of the features are needed to create a fully accurate model.

3.4 Random Forest with PCA

Since the models run thus far have been fully accurate, finding the leanest version of the data to get good classification is now considered. PCA is dimensionality reduction technique that was used to create a smaller dataset for the random forests to train on.

PCA was done with 25 and 35 omponents. These numbers were chosen due to the previous section where it was seen that 20 features could accurately classify, even on a small RF model with 10 trees.

To perform the PCA, the train and test X sets of data were scaled with a standard scalar, and then the PCA was performed with both 25 and 35 features. The results can be seen in figure 6.

The models with PCA did not perform as well as the full models or the feature selection models. The combination of 25 components and 25 trees in the random forest model resulted in the best classification accuracy.

Note, explain-ability is lost when PCA is applied due to the dimensional reduction of the original feature space. Therefore, especially with the nature of this classification, where the features of the mushrooms are obviously important to know, this set of models in not recommended to make decisions with.

3.5 Deep Learning - MLP

Tensorflow and keras were used to create the multilayer perceptron neural network. A sequential model was initiated and the following layers were added:

- model.add(Dense(10, activation='relu', kernel-initializer='he-normal', input-shape=(n-features,)))

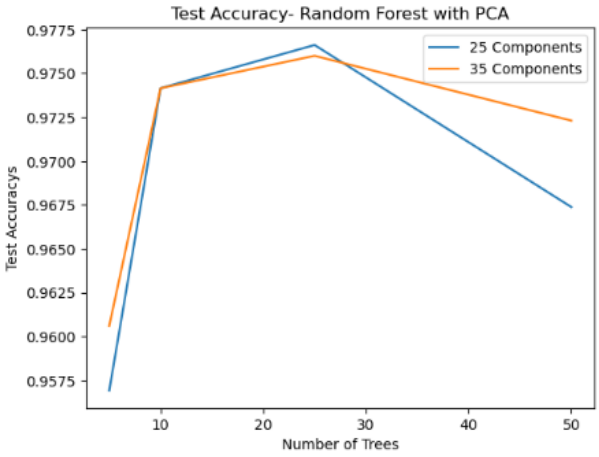

Fig. 6. Random Forest with PCA Model Comparison

Dataset	Model	Train Accuracy	Train F1 Score	Test Accuracy	Test F1 Score
Mushroom Dataset, PCA 25 Features	RF: nTrees = 5	0.999846	0.999846	0.956923	0.956912
Mushroom Dataset, PCA 35 Features	RF: nTrees = 5	0.999846	0.999846	0.960615	0.960600
Mushroom Dataset, PCA 25 Features	RF: nTrees = 50	1.000000	1.000000	0.967385	0.967319
Mushroom Dataset, PCA 35 Features	RF: nTrees = 10	1.000000	1.000000	0.972308	0.972267
Mushroom Dataset, PCA 25 Features	RF: nTrees = 10	1.000000	1.000000	0.974154	0.974120
Mushroom Dataset, PCA 35 Features	RF: nTrees = 25	1.000000	1.000000	0.974154	0.974120
Mushroom Dataset, PCA 35 Features	RF: nTrees = 50	1.000000	1.000000	0.976000	0.975968
Mushroom Dataset, PCA 25 Features	RF: nTrees = 25	1.000000	1.000000	0.976615	0.976587
Mushroom Dataset	SVM - linear kernel	1.000000	1.000000	1.000000	1.000000
Mushroom Dataset, Full Model	RF: nTrees = 50	1.000000	1.000000	1.000000	1.000000
Mushroom Dataset, Full Model	RF: nTrees = 25	1.000000	1.000000	1.000000	1.000000
Mushroom Dataset, Full Model	RF: nTrees = 10	1.000000	1.000000	1.000000	1.000000
Mushroom Dataset, Full Model	RF: nTrees = 5	1.000000	1.000000	1.000000	1.000000
Mushroom Dataset, Full Model	RF: nTrees = 5	1.000000	1.000000	1.000000	1.000000
Mushroom Dataset	MLP, 20 epochs	1.000000	1.000000	1.000000	1.000000

Fig. 7. Model Metric Comparison

- model.add(Dense(8, activation='relu', kernel-initializer='he-normal'))
- model.add(Dense(1, activation='sigmoid'))

The model was then compiled and run, which resulted in a test accuracy and F1 score of 100

4 MODEL COMPARISON

All the models run performed very well; the models run with the full feature space all were able to classify with an accuracy and F1 score of 100 percent. The summary of the models can be seen in figure 7. Note, the Random Forest feature selection models are not included in this table.

The models chosen are all relatively complex, and choosing which one to move forward with may depend on the application of the model. The support vector machine and highly accurate and runs fast compared to the other models. The full model and feature selection random forests can output the tree structure used to classify, which could be very useful for a field guide of some sort. Since PCA was not performed on these models, the explainability is very straight forward. The deep learning was accurate, but perhaps overkill for the needs of this modeling project.