**Voting Classifier with SVC, LR, DT, and RF**

**Using a Mushroom Classification dataset**

Bobby Gabriel, [bgabriel@bellarmine.edu](mailto:bgabriel@bellarmine.edu)

Link to dataset - <https://www.kaggle.com/datasets/uciml/mushroom-classification>

**ABSTRACT**

This project is to explore various classification models and determine the best model to select with a given data set. Classification is useful when working with a large dataset and when there is a need to predict the likelihood that a certain piece of data will fall into one of the predetermined categories. Classification, in short, is just pattern recognition; so, it has a wide range of uses across all different fields in math, economics, science, etc. For this specific project, I will be using a dataset from Kaggle about mushroom classifications and features. My classification models will try to predict if a given mushroom is edible or poisonous based on physical features.

1. **INTRODUCTION**

In this project, I have experimented with various regression and classifier machine learning models to further learn about a given dataset. In my project, I have utilized sklearn’s extensive resources and built-in ML models like SVC, Logistic Regression, Random Forest Classifier, Decision Tree Classifier, and Voting Classifier to compare which model best fit my data. I have determined that the Random Forest Classifier was the best fit to my dataset. The dataset I am working with is about mushrooms and provides data on different physical features of a mushroom.

1. **BACKGROUND** 
   1. *Data Set Description*

The Mushroom Classifier dataset I am using is from Kaggle and can be found using the link at the top of the paper. This dataset contains 23 columns that describe features of various mushrooms. Specific mushrooms are not listed in this dataset, just information spanning across 23 species of mushrooms on whether it is edible/poisonous, the color of the mushroom, various attributes for size and shape, and other important defining features of mushrooms. I chose to work with this because I knew that it would be a relatively simple dataset to use the 4 classification techniques (SVC, DT, LR, RF) due to the categorical data. This dataset was supplied over 30 years ago by the UC Irvine Machine Learning Repository. It was recently added to Kaggle due to the popularity in mushroom hunting in the recent years.

* 1. *Machine Learning Model*

The models I chose to use for my data set were Logistic Regression, Support Vector Classifier, Decision Tree Classifier, Random Forest Classifier, and Voting Classifier. Logistic Regression is most used for classification problems because it predicts a binary outcome based on a given independent variable. Going on to Support Vector Classification, this is a supervised learning algorithm that can be used for classification problems. The algorithm finds a hyperplane in an N-dimensional space and classifies the data points. The next model used is the Decision Tree Classifier. This model, again, can be used for classification and regression. The key difference for DT classifier is that it does not make any assumptions about the characteristics of the sample. The DT model implements a step-by-step decision process that evaluates 2 features at a time on a given node, selects one, then moves on to the next node. The next classifier I used is the Voting Classifier. The Voting Classifier is an ensemble learning method that uses numerous models (in this case SVC, LR, RFC, and DTC) and predicts an output based on their highest probability of chosen class as the output. Instead of training itself, it trains on the models supplied to it. The final classifier used and the one I selected for my final model is the Random Forest Classifier. This classifier uses an ensemble learning approach where multiple individual models are combined to form one optimal model. For RF classifier, multiple decision tree classifiers are combined to form the best result

1. **EXPLORATORY ANALYSIS (done)**

This dataset contains 23 columns with 8124 samples with all object data types. The data was collected with 23 species of gilled mushrooms in the Agaricus, and Lepiota Family Mushroom drawn from The Audubon Society Field Guide to North American Mushrooms.

**Table 1: Data Types**

|  |  |
| --- | --- |
| *Variable Name* | *Data Type* |
| V1 **class** | **Object** |
| V2 **cap-shape** | **Object** |
| V3 **cap-surface** | **Object** |
| V4 **cap-color** | **Object** |
| V5 **bruises** | **Object** |
| V6 **odor** | **Object** |
| V7 **gill-attachment** | **Object** |
| V8 **gill-spacing** | **Object** |
| V9 **gill-size** | **Object** |
| V10 **gill-color** | **Object** |
| V11 **stalk-shape** | **Object** |
| V12 **stalk-root** | **Object** |
| V13 **stalk-surface-above-ring** | **Object** |
| V14 **stalk-surface-below-ring** | **Object** |
| V15 **stalk-color-above-ring** | **Object** |
| V16 **stalk-color-below-ring** | **Object** |
| V17 **veil-type** | **Object** |
| V18 **veil-color** | **Object** |
| V19 **ring-number** | **Object** |
| V20 **ring-type** | **Object** |
| V21 **spore-print-color** | **Object** |
| V22 **population** | **Object** |
| V23 **habitat** | **Object** |

In this data set, there were no missing variables for any of the columns. I checked all variables with a distribution plot and there does not seem to be any skew or any unusual columns in the data.

1. **METHODS**
   1. *Data Preparation*

For the data preparation and preprocessing, I didn’t have to do too much work. For one, there were no missing values on any of the columns, so I did not need to impute any columns using the mean or median. Out of the 23 variables, I deemed only 2 to be unnecessary for the goal of the model (to determine whether a mushroom is poisonous). The 2 variables I dropped were bruises and population. I dropped ‘bruises’ because any mushroom can be damaged or bruised, it does not influence whether it is poisonous/edible. I dropped population because the number of mushrooms is irrelevant to determining whether it is poisonous. If you were to look at a single mushroom, the important factors to determine if its poisonous are its physical features.

After dropping the 2 variables, the next preprocessing I did was Ordinal encoding of all the variables. Since every variable was an object data type, I had to convert it into something the model could recognize. I chose to use ordinal encoding for variables because there were a range of values for many of the columns (example – gill-attachment had a – attached, d – free, or n – notched). One-hot encoding would not work for this as it is only useful on nominal data types.

The last preprocessing I performed was using train/test/split and Standard Scaler by sklearn. This helped with the non-tree models of my data like SVC and Logistic Regression. Standard Scaler centers the values in each column around the mean with a unit standard deviation.

* 1. *Experimental Design*

You will run your model several times with different parameters to see what different results you get. In a table, describe your experimental parameters. Three or four experiments are sufficient. This is where you will describe how you divided your data into train, validate and test data sets. For example:

Table X: Experiment Parameters

|  |  |
| --- | --- |
| **Experiment Number** | **Parameters** |
| 1 | Standardized features with 80/10/10 split for train, validate, and test with  **Logistic Regression:**  Solver: lbfgs  Multi\_class: auto  Max\_iter: 10000  **Random Forest Classifier:**  Criterion: gini  N\_estimators: 1000  **SVC:**  Gamma: auto  Kernel: linear  **Decision Tree Classifier:**  Criterion: gini  Max\_depth: 3  Random\_state: 0 |
| 2 | Standardized features with 80/10/10 split for train, validate, and test with  **Logistic Regression:**  Solver: liblinear  Multi\_class: auto  Max\_iter: 10000  **Random Forest Classifier:**  Criterion: entropy  N\_estimators: 1000  **SVC: score 1.0**  Gamma: auto  Kernel: poly  **Decision Tree Classifier:**  Criterion: entropy  Max\_depth: 3  Random\_state: 0 |
| 3 | Standardized features with 80/10/10 split for train, validate, and test with  **Logistic Regression:**  Solver: saga  Multi\_class: auto  Max\_iter: 10000  **Random Forest Classifier:**  Criterion: gini  N\_estimators: 10000  **SVC: score 1.0**  Gamma: auto  Kernel: rbf  **Decision Tree Classifier:**  Criterion: gini  Max\_depth: 10  Random\_state: 0 |
| 4 | Standardized features with 80/10/10 split for train, validate, and test with  **Logistic Regression:**  Solver: sag  Multi\_class: multinomial  Max\_iter: 10000  **Random Forest Classifier:**  Criterion: entropy  N\_estimators: 10000  **SVC:**  Gamma: auto  Kernel: sigmoid  **Decision Tree Classifier:**  Criterion: entropy  Max\_depth: 10  Random\_state: 0 |

* 1. *Tools Used*

The following tools were used for this analysis: Google drive with Google Colab was used solely for this project. For the notebook, Python 3.6.9 was used. In addition to base Python, the following libraries were also used: Pandas 1.3.5, Seaborn 0.11.2, Sklearn 0.18.1. Pandas was used to read and visualize the dataset and .csv file. It was also used for all data preprocessing. Seaborn was used to visualize each column of my dataset and see if data was skewed at all. Sklearn was used to import and use all my training models.

1. **RESULTS**

Experiment 1

Timeline

Description automatically generated

Experiment 2

Timeline

Description automatically generated with medium confidence

Experiment 3

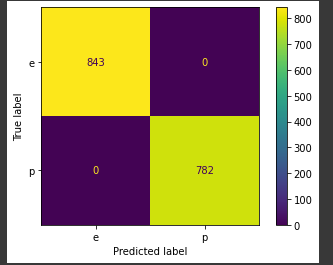
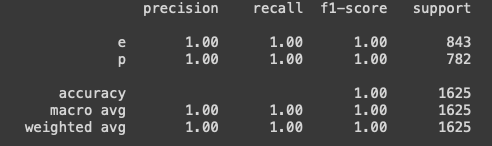
Timeline

Description automatically generated with medium confidence

Experiment 4

Timeline

Description automatically generated

Confusion Matrix and Classification report of the best model – Random Forest (e means edible, p means poisonous)

* 1. *Discussion of Results*

The model that provided the best and most consistent result was the Random Forest Classifier at 100% score. Although all other models except Logistic Regression achieved this rating, the Random Forest Classifier held this score throughout 4 different experiments where I changed parameters. I believe Random Forest was the best due to the ensemble learning method and the fact that it took multiple different decision trees and combines their input for the best value. This model uses a wide range of inputs and is very accurate. My worst performing model was Logistic Regression. Although I say worst, it still achieved an accuracy of 95%. I believe this was the lowest performing because it is not as in depth and complex of an algorithm as the others.

* 1. *Problems Encountered*

One of the problems I had was finding a good dataset to use my models on. I knew I wanted a dataset that uses ordinal values and makes a prediction on a target variable, but it was difficult finding a dataset that wasn’t overcomplicated and wasn’t too small. Another problem I ran into was when I was running my experiments. Using sklearn’s documentation, I tried to give a unique set of parameters for each experiment, but some parameters didn’t work with each other. An example would be for Logistic Regression, the solver ‘liblinear’ did not want to work with multiclass ‘multinomial`. So, each time I ran into that problem, I would have to switch the parameters to make them compatible with each other.

* 1. *Limitations of Implementation*

The limitation of my model is that it can be slow the more trees you add and slow to make predictions. So, if I ever wanted to scale my dataset, it might become too much of a problem. A solution to this would be a simpler classifier model such as Logistic Regression. It will not be as accurate, but it is faster than a tree-based model like decision tree or random forest.

* 1. *Improvements/Future Work*

The improvement I would make to my model in future work would be to keep the same classifiers because they performed well and expand on the data. This dataset was a good start, but it only covered 23 species of mushrooms so there is no way of knowing that if more species of mushrooms were entered, that the model would correctly guess if they were poisonous/edible based on their physical features. I would also keep the same variables; I thought it was a nice spread of all the physical features a mushroom could have to help determine if edible or not.

1. **CONCLUSION**

For this project I compared 5 different models of supervised learning algorithms – SVC, Decision Tree Classifier, Random Forest Classifier, Logistic Regression, and Voting Classifier which trains using the previous 4. To compare these models, I used a dataset that classifies mushrooms and their various attributes. This dataset was all ordinal data which made it perfect to use these classifiers. My goal and target for these models was to determine whether or not a mushroom was edible or poisonous. So after preparing the data and fitting the models I determined that the most consistent and accurate model was the Random Forest Classifier at 100% accuracy. Along with Random Forest, all other models except for Logistic Regression achieved the same score of 100% at various times but only Random Forest managed to score 100% through all 4 experiments. Overall, I believe I had a good model.

**REFERENCES**

<https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html>