Edible Mushrooms vs Poisonous Mushroom

Anthony Huynh and Nguyen Hoang

***Abstract*—Background: Mushrooms species can classified as edible, poisonous, or unknown and not recommended. The latter class was combined with poisonous one in our data set. Our data provided us with 22 mushroom attributes, a multivariate data set characteristic, 8124 instances, with a categorical attribute characteristics. Method:Like the different types of classifications for mushrooms, there are many ways to do use it's data for machine learning. In this project we utilized three supervised learning techniques for classification: logistic regression, k-nearest neighbor, and support vector machine. We split the training and testing sets of data, created models, and compared the results. Results: The SVM Model gave us an accuracy of 1. The non-NCA, KNN Model gave us an accuracy of 0.99889 and an accuracy of 0.99409 for the NCA model. The Logistic Regression model gave us an accuracy of 0.94645 while the Multinomial Logistic Regression with L1 penalty gave us an accuracy of 0.93426. Conclusion: The SVM model gave us the best results.**

I. Introduction

There are many types of mushrooms and not all are edible. There are many that are not edible since they are poisonous. Some have unknown edibility and some are not recommended. We took mushroom data from UCI's Machine Learning Repository. This data contains descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Gills also called "lamellae" in Mycology are many platelike or bladelike structures attached to the underside of the cap in some mushrooms. Both Agaricus and Lepiota are a part of the Agaricaeae family and are in the order of Agaricales. In the Agaricaea family the caps are scurfy to smooth, and range from roughly flat to umbonate. Despite these characteristics there is no simple rule for determining whether or not a mushroom is poisonous or edible. There are only a few indications that can lead you to make an assumption whether or not a mushroom is poisonous or not and one of them are the white gills. Mushrooms with a red color on the cap are also a good indication.

II. Task Description

For our project we decided to classify mushrooms in the Agaricus and Lepiota family on whether they are edible or not. The mushrooms are described in terms of their physical characteristics and our data is taken from UCI's Machine Learning Repository. They received their data from Audubon Society Field guide.

We wanted to use three models for this binary classification task; logistic regression, K - Nearest Neighbor (KNN), and Support Vector Machine (SVM).

III. Major Challenges and solutions

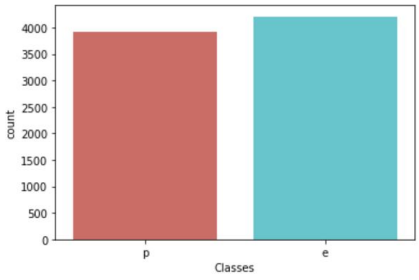
The first major challenge was to decide what models we would like to create. We had to choose supervised machine learning since it was a classification problem. We then decided on those methods stated above. The next challenge was to import the data and change some of the data characteristics to fit what we needed to be done. For example, in the classes field within our data set, edible was defined as "e" while poisonous was defined as "p." We could not use it in our binary classification model since it is a string and not a numeric data type. We had to change them to 0 and 1, respectively. We also needed to change the other attributes to numeric data types from the string values we were given. Another challenge was presented when trying to use the categorical data. We needed to clean the data for it to be functional and to prevent overlapping between each other. To achieve this we needed to define the distance in between points.

IV. Experiments

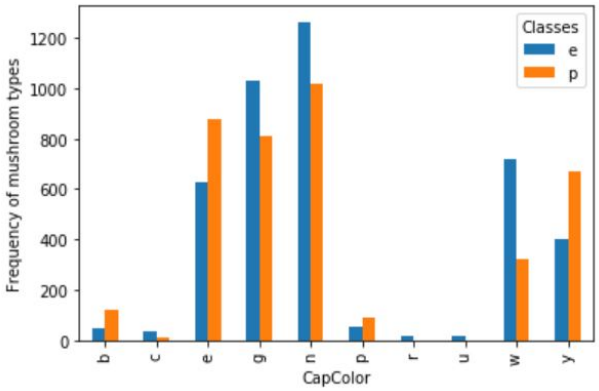
A. Data Set Description

Our data set consisted of the following attributes: cap(shape, surface, color), bruises, odor, gill(attachment, spacing, size, color), stalk(shape, root, surface, color), veil(type, color), ring(number, type), spore color, population and habitat. In the data retrieval stage of our project, we counted the total amount of poisonous and edible mushrooms. We came up with an exact value and displayed the two classes on a graph. Once again, E means edible and P means poisonous. The results are shown below.

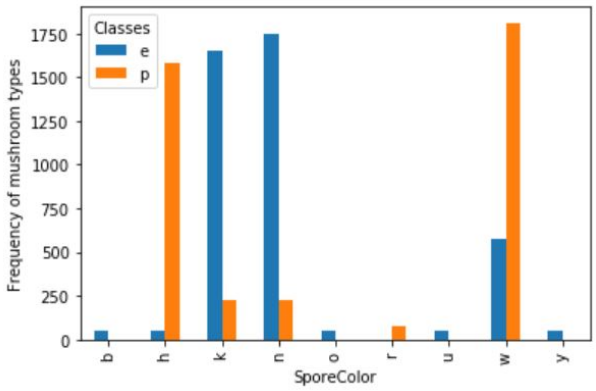
```
In [5]: mushroom_df["Classes"].value_counts()
Out[5]: e    4208
        p    3916
        Name: Classes, dtype: int64

In [6]: sns.countplot(x = "Classes", data = mushroom_df, palette = "hls")
Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1abdced0>
```



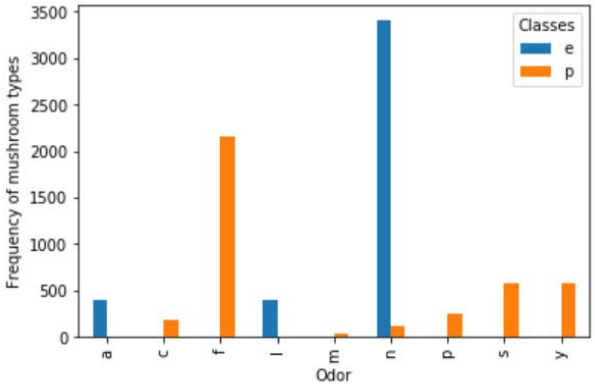
Another step we took to better understand our given data was to try to find relationships between and edible mushroom and a poisonous mushroom. Like stated earlier, there is no simple rule for determining this, but we wanted to make assumptions based on their attributes. In the first graph we noticed that our assumption about mushrooms with red caps are likely to be poisonous holds true. Red cap is defined by "e" in the cart. Other cap colors that are likely to predict a mushroom being poisonous is defined by "y" which is the color yellow, "b" for buff, and "p" for pink caps.



The second chart we looked at the data for spore colors. Chocolate color defined by "h", green defined by "r", and "w" defined by white are spore colors that are more likely to be poisonous compared to edible. While "k" for black and "n" for brown are more likely to be edible.



The third graph we made about our data set is for odor. From this graph we noticed that if the mushroom gives off creosote, foul, musty, pungent, spicy, or fishy smell then they are likely to be poisonous. The letter representing each are "c", "f", "m", "p", "s", and "y" respectively.



B. Evaluation Metrics

We then split this data into thirds. One third for the training data, one third for validation, and another third as a testing set.

C. Results and analysis

For Logistic Regression we printed out the Confusion Matrix, Average Accuracy, Per - Class Precision and Per - Class Recall. The results are as followed:

```
Confusion Matrix:
[[1255   47]
 [  98 1308]]
Average Accuracy: 0.9464549483013294
Per-Class Precision: [0.92756837 0.96531365]
Per-Class Recall: [0.96390169 0.93029872]
```

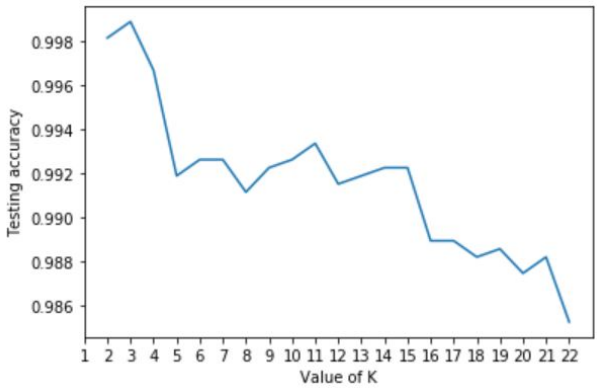
For the Confusion Matrix we followed the numeric values of whether or not the mushroom is edible or poisonous. Poisonous mushrooms were evaluated as a 0 and edible mushrooms were evaluated as a 1. The matrix used the testing data set. As a result we came up with "1308" in the bottom right corner which is our True Positive value. This value told us that 1308 mushrooms were edible and we were able to correctly predict them. On the other hand, "1255" of the mushrooms in our data were correctly predicted as poisonous. This can be seen on the top left corner of our matrix - which is our True Negative value. Our model only concluded that "47" of the mushrooms were poisonous when they are actually edible - this is our False Negative. On the other hand the results show our False Positive as "98" meaning that we predicted a mushroom to be edible even though it is poisonous.

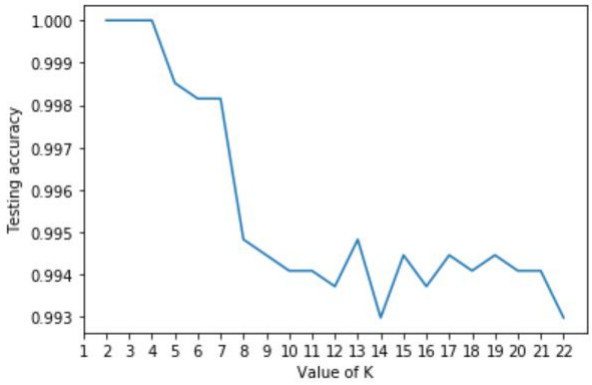
With the confusion matrix we were able to find the accuracy of our model by performing (TP + TN)/ Total. In our case it was 2563/2708 giving us an accuracy of 0.9465. We were also able to find the precision and recall. The precision for poisonous mushrooms was 0.92756837 while the precision for edible mushrooms was 0.96531365. The recall for poisonous mushrooms was 0.96390169 and 0.93029872 for edible mushrooms.

```
Sparsity with L1 penalty: 4.55%
Test score with L1 penalty: 0.9343
Confusion Matrix:
[[1211   91]
 [  87 1319]]
Average Accuracy: 0.9342688330871491
Per-Class Precision: [0.93297381 0.93546099]
Per-Class Recall: [0.93010753 0.93812233]
```

L1 regularized logistic regression is widely used for many classification problems, particularly ones with many features as it zeroes out the features' weights. A regularization is required in order to avoid over-fitting and L1 regularization has been shown to have good generalization performance in the presence of many irrelevant features.

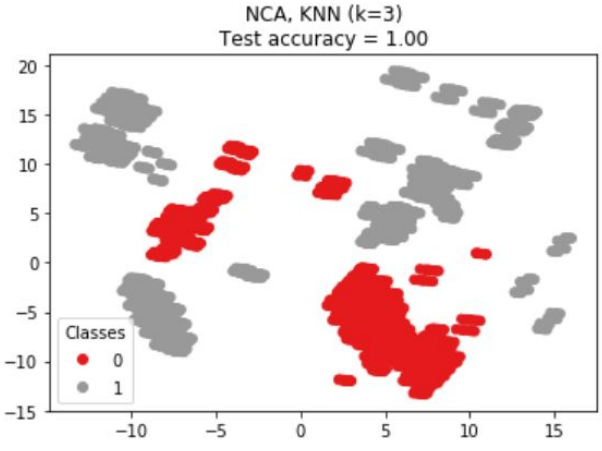
Our second method for this classification problem was using KNN. For KNN we ran two graphs: one to run our k value from 1 to 22 to see the testing accuracy and other to test accuracy for the Neighborhood Component Analysis or NCA. The graphs are as shown below.





Instead of finding the optimal k value by squaring the total number of features and rounding, we just showed how the k value effected the testing accuracy. Generally speaking, the more k values we used the less accurate our models were. The left graph or the one without using NCA depicts the best accuracy is when k is equal to 3. After using NCA on the right, the k values from 2 to 4 work the best as in they give the best accuracy. The confuson matrix, accuracy, precision, and recall for both will be presented soon.

Furthermore, we also produced a clustering graph for NCA when k is equals to 3.



Neighborhood Component Analysis is an algorithm that uses a technique similar to KNN to find a space in which neighborhoods of points sharing the same labels are tighter than points with different labels. NCA clusters data based on the results of dimensionality reduction on the matrix. NCA is attractive for classification because it can naturally handle multi - class problems without any increase in the model size, and does not introduce additional parameters that require fine - tuning by the user. NCA classification has been shown to work well in practice for data sets of varying size and difficulty. The nearest neighbor classification can naturally produce highly irregular decision boundaries. In our case, as a result of NCA, our knn.score value is equal to 1 since if you reduce the dimension, you also reduce the complexity of the total number of features. This will lead to a better overall accuracy.

From looking at both evaluations, we can conclude that the KNN results are better than NCA results. Normally, NCA should produce a better outcome compared to just KNN, but our model shows otherwise. This is most likely due to the slight variations in numerical errors that computers have when performing on a high-dimensional data. As we can see that the difference in accuracy of both methods is small, only up to the order of 0.001.

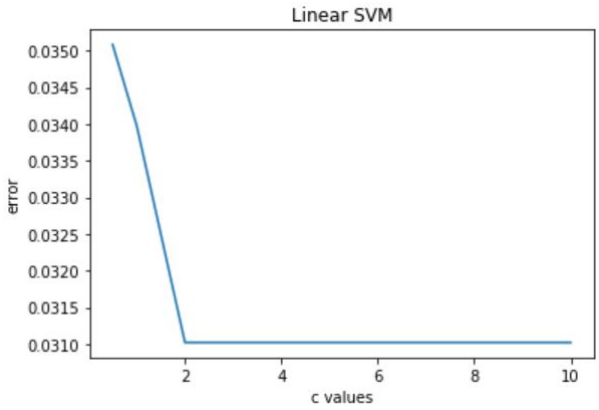
KNN Confusion Matrix

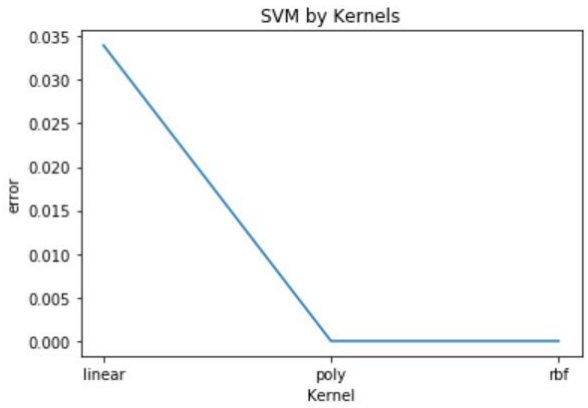
```
Confusion Matrix:
[[1302    0]
 [   3 1403]]
Average Accuracy: 0.9988921713441654
Per-Class Precision: [0.99770115 1.        ]
Per-Class Recall: [1.         0.99786629]
```

NCA Confusion Matrix

```
Confusion Matrix:
[[1298    4]
 [  12 1394]]
Average Accuracy: 0.9940915805022157
Per-Class Precision: [0.99083969 0.99713877]
Per-Class Recall: [0.9969278  0.99146515]
```

In our third model we used Support Vector Machine also known as SVM. This was similar to what we did for homework 5. In this project we noticed that for a Linear SVM, using C values 2 or greater would provide us with the least amount of error. However, when we ran SVM by kernels, the Poly Kernel and RBF worked better compared to a Linear SVM. The SVM Confusion Matrix turned out a lot better with 1's for accuracy, per - class precision, and per - class recall.






```
Confusion Matrix:
[[1302    0]
 [   0 1406]]
Average Accuracy: 1.0
Per-Class Precision: [1. 1.]
Per-Class Recall: [1. 1.]
```

D. Conclusion and Future Works

In conclusion, we can conclude that SVM performs the best out of the three models we presented. The list for best accuracy to worst accuracy is SVM, KNN, and then Logistic Regression. We had to overcome the challenges of cleaning a data set up so that we are able to split it up into three different groups. Changing the attributes into a numerical value took a bit of time.

This research encourages the use of machine learning to help scientists and other researchers in their field of study. Since there are many mushrooms and new ones are being found each year, scientists can use machine learning to easily predict whether a newly found mushroom is poisonous or edible.

REFERENCES

[1] Sarkar, D. (2019, March 27). Categorical Data. Retrieved December 18, 2019, from https://towardsdatascience.com/understanding-feature-engineering-part-2-categorical-data-f54324193e63.

[2] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[3] Lee, Su-In Lee, Honglak Abbeel, Pieter Ng, Andrew. (2006). Efficient L1 Regularized Logistic Regression. Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06). 21.