2nd International Conference of Communication Science Research (Paper Template)
24-25 July 2018; Surabaya, Indonesia http://conference-fikom.unitomo.ac.id

Classifying Dry Bean Types Using Decision Trees

Patrick Krol
CS 422: Data Mining, Hood College
Frederick, MD
pdk6@hood.edu

***Abstract*—In this paper, I take a look at a database found on the UCI Database Repository cataloging various bean types and their measurements. The original authors of this database found that, using decision trees, they could get an accuracy of 92.5 percent [1]. So following in their footsteps, I first create a decision tree with all attributes present in the process. To do this, I used python and the SciKitLearn package and more specifically, the DecisionTreeClassifier method. Upon randomly splitting up the data set into 80 percent for training and 20 percent for testing, I found that this tree had an accuracy of around 89.5 percent. Digging deeper, I realized that some of the attributes were holding back the accuracy. So, a second decision tree was made and this yielded an accuracy of around 90.3 percent. This was better, but not ideal. I then individually classify each bean type represented in this data set creating seven different trees. I found that these decision trees have an average accuracy of about 97 percent. This sounds great, but some beans get classified into several types of beans or no types of beans making the result ultimately useless.**

I. Introduction

We have all heard, "Beans, beans, the magical fruit, the more you eat the more you toot, the more you toot the better you feel, so let's have beans for every meal," but what if you do not know what bean you are eating and you happen to have the equipment handy to measure the bean at hand? This is the important question I will attempt to answer. But really, why beans? "Seed quality is definitely influential in crop production. Therefore, seed classification is essential for both marketing and production to provide the principles of sustainable agricultural systems"[1]. This is the goal the original authors of the database had in mind when they decided to classify bean types. My goals are a little different as I did not really have a specific problem in mind when looking at this database. For me, this database looked like a good entry point into the world of classification. So my goal was to use one of the common classification techniques such as clustering, KNN, or decision trees to classify the beans and get a result similar to those found in the article that this database was tested in.

II. Literature Review

The paper titled *Multiclass classification of dry beans using computer vision and machine learning techniques* that was originally published along with this dataset used 4 different classification methods. These were Multilayer Perceptron, Support Vector Machine, k-Nearest Neighbors, and Decision Tree. Overall correctness percentages published for each technique were 91.73%, 93.13%, 87.92%, and 92.52% respectively [1]. This means that the decision tree had the second highest accuracy out of the four methods, only falling behind the Support Vector Machine. As mentioned in the introduction, dry beans are "the most important and the most produced pulse over the world"[1]. A pulse in food terms is "the edible seed from a legume plant."[2] It is common knowledge that using lower quality seeds will result in lower crop yields regardless of the conditions in which these seeds are grown. It follows that high quality seeds will produce more seed. The authors thus designed this database to root out the beans of adequate quality.

I. Methods

A. Data

The Dry Beans database can be located on the UCI Machine Learning Repository. This is a good database to work with for several reasons, but first, what does the data look like? Here is a snapshot of the Dry Bean Database:

Table 1: Data Sample

Area	erimeter	rAxisLe	rAxisLe	pectRati	centric	nvexAr	ivDiam	Extent	Solidity	oundne	mpactn	apeFact	apeFact	apeFact	apeFact	Class
28395	610.29	208.18	173.89	1.1972	0.5498	28715	190.14	0.7639	0.9889	0.958	0.9134	0.0073	0.0031	0.8342	0.9987	SEKER
28734	638.02	200.52	182.73	1.0974	0.4118	29172	191.27	0.784	0.985	0.887	0.9539	0.007	0.0036	0.9099	0.9984	SEKER
29380	624.11	212.83	175.93	1.2097	0.5627	29690	193.41	0.7781	0.9896	0.9478	0.9088	0.0072	0.003	0.8259	0.9991	SEKER
30008	645.88	210.56	182.52	1.1536	0.4986	30724	195.47	0.7827	0.9767	0.9039	0.9283	0.007	0.0032	0.8618	0.9942	SEKER
30140	620.13	201.85	190.28	1.0608	0.3337	30417	195.9	0.7731	0.9909	0.9849	0.9705	0.0067	0.0037	0.9419	0.9992	SEKER

As one can see, this data has quite a few attributes. It includes 16 total attributes and one "Class" column that denotes which bean type that specific bean or row belongs to. Among the attributes, there are 12 denoting dimensionality of the bean and 4 shape attributes as follows:

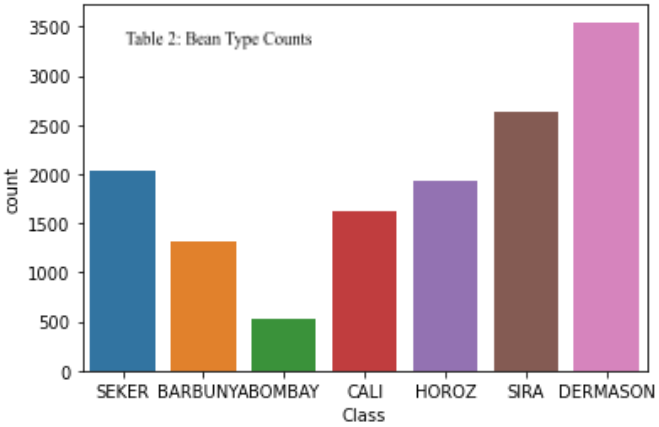
1. Area (A): The area of a bean zone and the number of pixels within its boundaries.
2. Perimeter (P): Bean circumference is defined as the length of its border.
3. Major axis length (L): The distance between the ends of the longest line that can be drawn from a bean.

4. Minor axis length (l): The longest line that can be drawn from the bean while standing perpendicular to the main axis.
5. Aspect ratio (K): Defines the relationship between L and l.
6. Eccentricity (Ec): Eccentricity of the ellipse having the same moments as the region.
7. Convex area (C): Number of pixels in the smallest convex polygon that can contain the area of a bean seed.
8. Equivalent diameter (Ed): The diameter of a circle having the same area as a bean seed area.
9. Extent (Ex): The ratio of the pixels in the bounding box to the bean area.
10. Solidity (S): Also known as convexity. The ratio of the pixels in the convex shell to those found in beans.
11. Roundness (R): Calculated with the following formula: (4piA)/(P^2)
12. Compactness (CO): Measures the roundness of an object: Ed/L
13. ShapeFactor1 (SF1)
14. ShapeFactor2 (SF2)
15. ShapeFactor3 (SF3)
16. ShapeFactor4 (SF4)

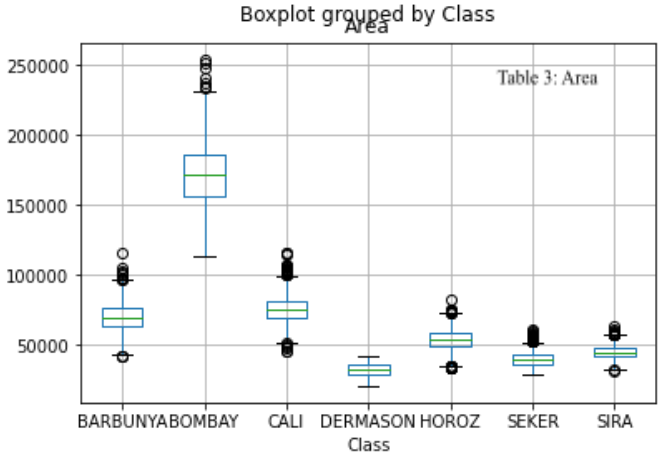
Then the final column, has the following dry bean types:

1. Seker
2. Barbunya
3. Bombay
4. Cali
5. Dermosan
6. Horoz
7. Sira

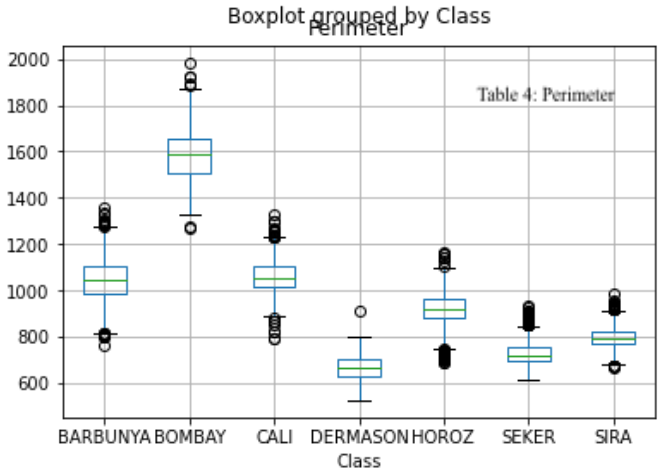
There are a total of 13611 entries, however the distribution of bean types across entries is not equal as can be seen in table 2.



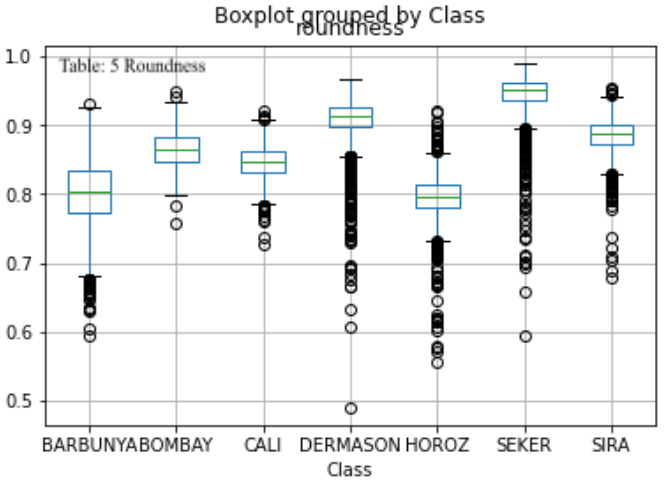
As we can see, the Dermason dry bean type has the highest representation in this database, then the Sira bean, Seker, Horoz, Cali, Barbunya, and finally with the least amount of representation, the Bombay bean. Counts range from 3500 at the greatest to just over 500 at the least. Then to get an idea of how these beans are different from one another I looked into the distribution across bean types of some of the attributes of the beans. First is area in Table 3.



This boxplot shows that Bombay beans are significantly larger than the rest of the dry beans selected for this database. Horoz, Seker, and Sira all are fairly close in area. The same could be said for Barbunya and Cali. Then the Dermason bean is clearly smaller than the rest.



Now taking a look at Table 4, the perimeter of the beans, we see the same pattern emerge for the most part. The only significant difference is that the Horoz bean sticks out a little more than the others it was paired with in the area plot. These are both measurements of the size though. To get a better idea of the shape, I also plotted the roundness of the beans as seen in Table 5.



This plot shows that the roundness of the bean in general does rely quite a bit on what type of bean being dealt with, but there is also a lot of randomness to the roundness seen in the large number of outliers present in the graph. This is especially evident in the Dermason, Horoz, and Seker beans.

A different aspect of this database that appealed to me is the ease of use. Generally, the first step when doing any sort of analysis on a data set is to make sure the data is clean and uniform. This could mean checking for incorrect entries or filling in and or deleting rows with null values. However, this dry bean database was captured using a computer vision system as the title of the original paper suggests. Each bean was visually captured with a high resolution camera, then the pictures were fed into the computer vision system which then takes very precise and scientific measurements [1]. This means that the database already had very uniform data that was in a ready-to-go state.

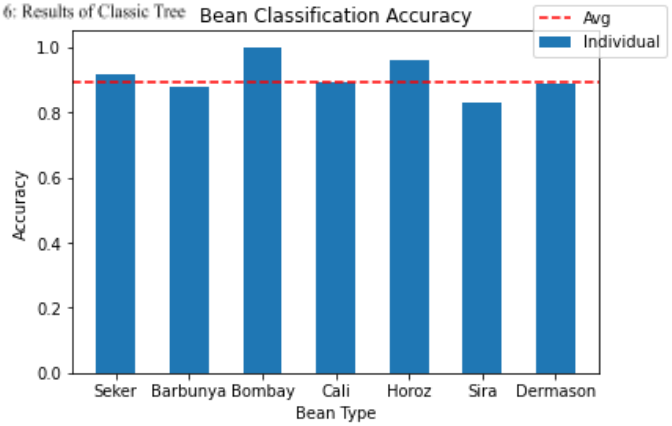
B. Methods

To classify the seven different bean types I decided to use a decision tree as mentioned previously. There is no real reason behind this decision other than that it is a common and well documented technique that seemed like a good introduction into the world of classification. I decided to move forward with python, using the SciKitLearn package, and more specifically centered around the DecisionTreeClassifier method which generates the actual tree. It is relatively easy to use. The only preprocessing needed in order to input the data set into the DecisionTreeClassifier method was to randomly split off 80% of the data for the creation of the tree and then split off the "Class" column from the 16 attributes. These two dataframes then become the two inputs for the decision tree method. This method also works well for this project as it supports multi-class classification, or it can classify into multiple outputs rather than just a binary result. However, I was not satisfied with just this one tree. I decided to make a new decision tree, but this time removing some of the more ambiguous attributes to try and increase the overall accuracy of the tree. The attributes I decided to remove were eccentricity, majoraxislength, and minoraxislength. Then after this, I decided to test out how the results would differ if instead of creating one multi-class decision tree, I made seven binary decision trees. They would classify each bean type separately. To do this, I simply separated out the bean I wished to look at, and made its class column all ones. Then I took the rest of the database and turned its class column into zeros. This allowed the creation of binary decision trees.

II. Results

After constructing the original decision tree, I then ran the rest of the 20% test data through the tree to be classified, collecting all the correct entries. Results are found in table 6.

Table 6: Results of Classic Tree

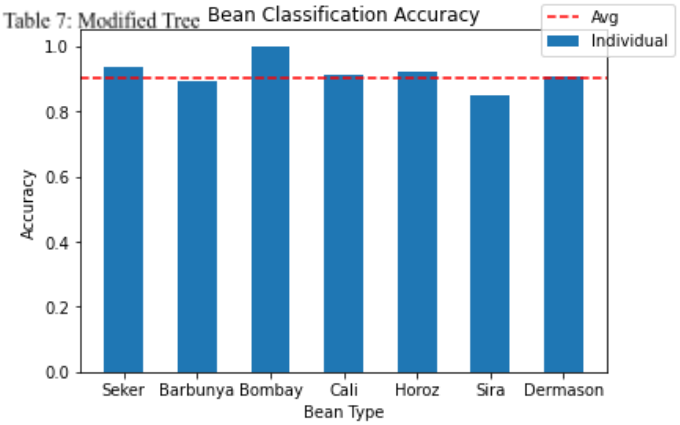


The accuracy of this first decision tree with no modifications is as follows:

- Overall Accuracy: 89.5%
- Seker: 91.5%
- Barbunya: 87.7%
- Bombay: 100%
- Cali: 89.1%
- Horoz: 95.9%
- Sira: 83.0%
- Dermason: 88.8%

Next, I tested the modified decision tree with a few unnecessary features removed. Results are found in Table 7.

Table 7: Modified Tree



- Overall Accuracy: 90.3%
- Seker: 93.4%
- Barbunya: 89.1%
- Bombay: 100%
- Cali: 91.1%
- Horoz: 92.3%
- Sira: 84.7%
- Dermason: 90.5%

Next, I tested the results of the separated decision trees.

- Overall Accuracy: 97.1%
- Seker: 98.1%
- Barbunya: 98.1%
- Bombay: 100%
- Cali: 97.6%
- Horoz: 98.0%
- Sira: 93.5%
- Dermason: 94.7%

Now these seem like fantastic results! There is a problem though.

III. Conclusion

The original decision tree fell about 3% short of the 92.5% goal. The second modified tree that was created did slightly better than the first as it found an overall accuracy of 90.3% versus the 89.5% of the first tree. I am unsure what the authors did differently as far as modification of the basic decision tree to increase accuracy. Now the results of the split decision trees may seem very good, however, this is where the nuances of statistics come into play. These split trees are good for telling one what a bean is not, but they cannot be used for classifying what a bean is. This is because if you take a single bean and run it through all the trees, you could be left with a bean that gets classified into several types of bean or into no types of beans depending on the bean. So, this means that the modified tree had the highest overall accuracy of the trees tested. Also, Some beans did better than others in terms of classification. Beans such as the Bombay bean had a perfect classification record, while others such as Sira and Dermason had less than stellar percentages. This is due to Bombay having very unique features while the other beans have similar features.

References

[1] Murat Koklu and Ilker Ali Ozkan. “Multiclass classification of dry beans using computer vision and machine learning techniques”. In: Computers and Electronics in Agriculture 174 (2020), p. 105507. issn: 0168-1699. doi: https :/ / doi . org / 10 . 1016 / j . compag . 2020 . 105507. url: https : / / www .sciencedirect.com/science/article/pii/S0168169919311573.

[2] “Legumes and Pulses.” *The Nutrition Source*, 2 Mar. 2022, https://www.hsph.harvard.edu/nutritionsource/legumes-pulses/#:~:text=A%20pulse%20is%20the%20edible,the%20pod%20is%20the%20pulse.

[3] Murat Koklu and Ilker Ali Ozkan, C. (2020), UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], rvine, CA: University of California, School of Information and Computer Science.