Introduction

For the final project I selected the Dry Bean Dataset, found on UCI's machine learning repository. This dataset initially used a computer vision algorithm to find numerical categorizations of images of seven species of dry beans: Seker, Barbunya, Bombay, Cali, Dermosan, Horoz and Sira. From this model 17 attributes were determined, and are contained in this dataset. I decided to focus on the following 8 attributes for the project: Area, Perimeter, MajorAxisLength, MinorAxisLength, AspectRation, Eccentricity, ConvexArea, EquivDiameter, Extent, Solidity, roundness, Compactness. My goal for the project is to use the numerical attributes and machines learning methods to classify the seven species of beans.

Before diving into the project I wanted to get a sense of which of the 8 attributes are most important for determining the species of a bean. So I started by fitting the Logistic Regression model and arbitrarily removing features, selecting fewer features and total number of species. I determined that I could obtain a high accuracy (~98%) rate if I used Area, Perimeter, MajorAxisLength, AspectRation and only used three species, but I wanted to determine for sure what features were pivotal. So I used the Shapley (shap) module in python to fit their model and create plots (Figs. 1 and 2) to highlight which features are of most importance when trying to classify the beans by species. Based on the Shapley model the features MinorAxisLength, Compactness and roundness were determined to be most relevant for classification. Taking the time to do the Shapley test really helped me in the right direction for selecting features for the machine learning models to use for classification. My initial arbitrary selection of features did not even include the most relevant features and according to the plots these three features are far more important than the four I initially picked.

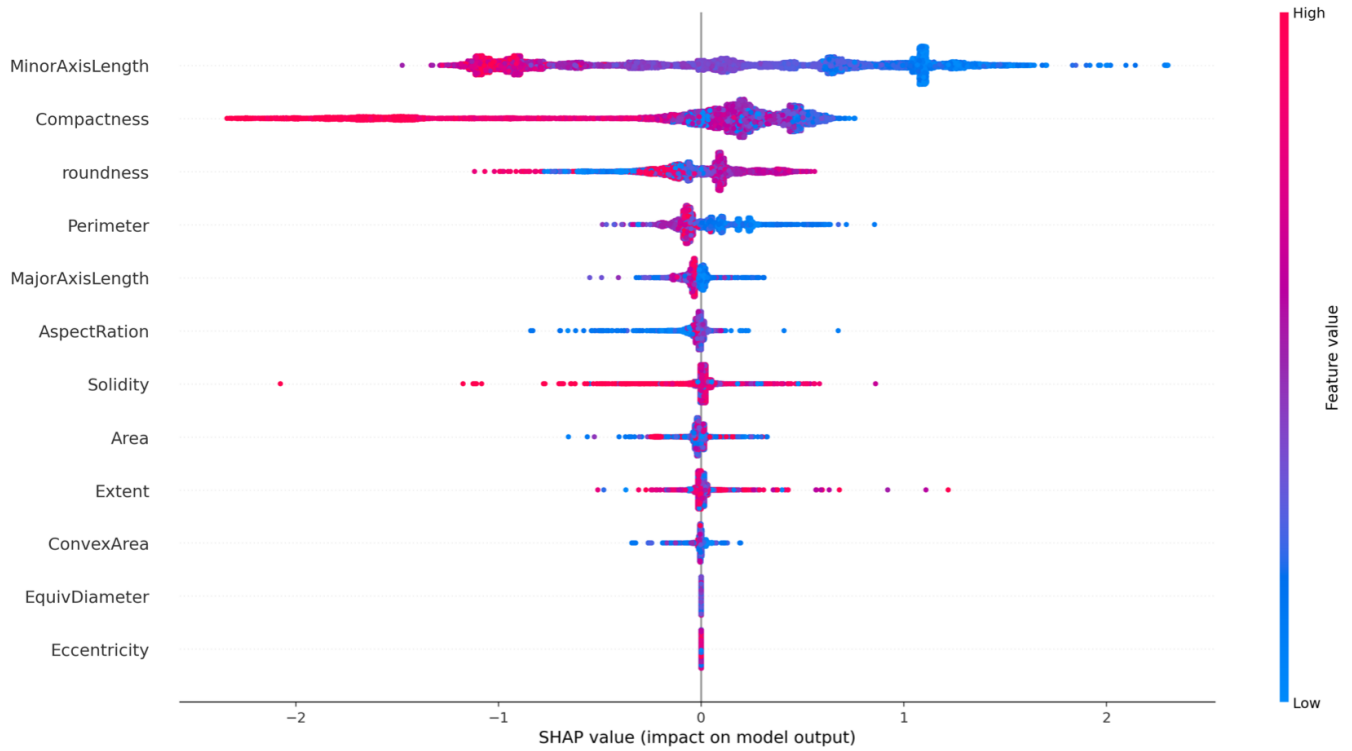

Fig 1. Shapley summary plot

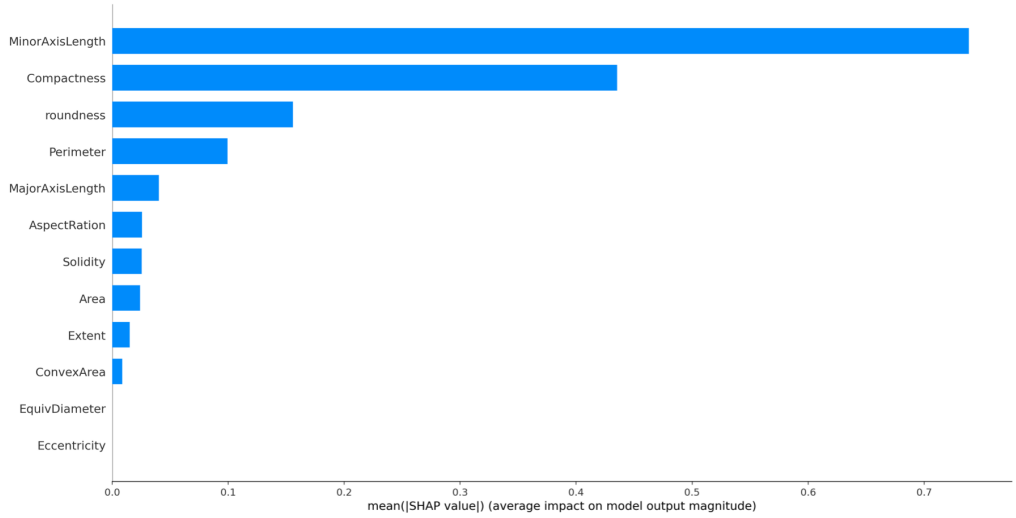

Fig 2. Shapley barplot

Feature Distribution and Statistics

Fig. 1 gives a very good representation of the spread of the data, similar to a violin plot. From this graph we can see that there is a decent spread of the data with a few outliers. However in the three main features there are noticeable clumping of data, which we can interpret to be the different bean species. Though it must be noted that the values from the Shapley plots have been scaled and then applied to the Shapley model to determine a feature importance metric. This is why I computed a pairwise plot of the data, highlighting the three main features.

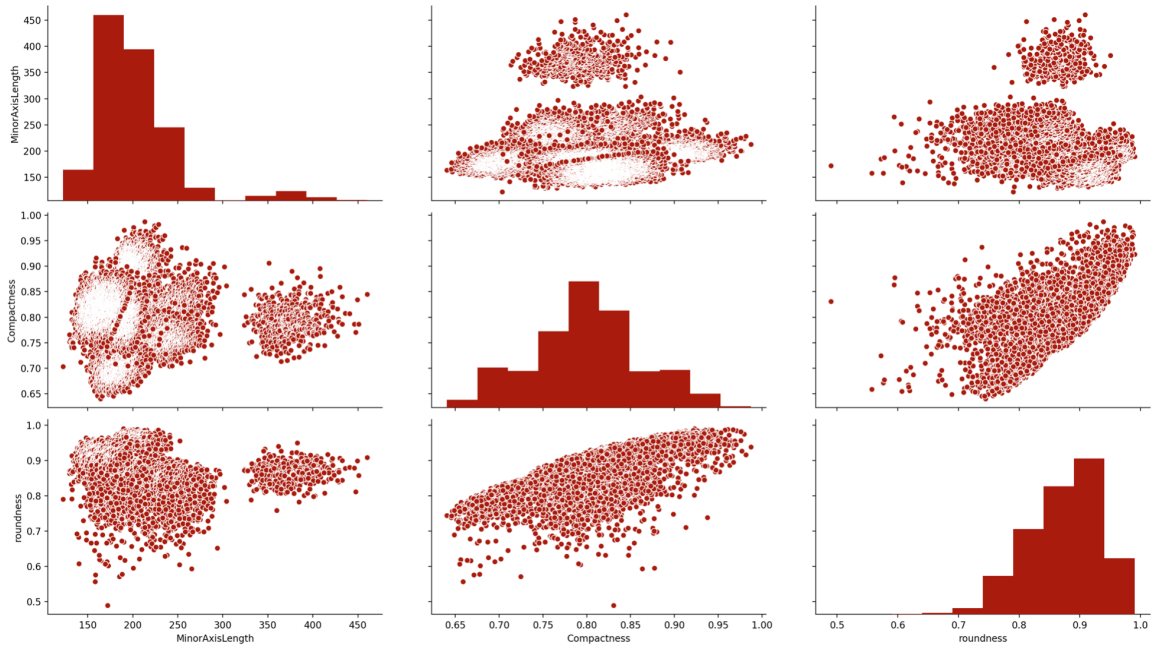

Fig 3. Pairwise plot of three main features (MinorAxisLength, Compactness, roundness)

From the pairwise plot (Fig. 3) we can see that there is no strong correlation between any of the features, because the data does not have linearity in either the positive or negative direction. There is some clumping in the data e.g. between MinorAxisLength and Compactness, roundness and MinorAxisLength, though they are not in distinctly separate quadrants of the graph. Looking at the histograms they are unimodal and follow a general Gaussian distribution. There is a right skew and a distinct clump in the MinorAxisLength graph, which may be a separation of species. There is also a slight left skew in the roundness histogram. Since there is no strong separation of data I decided to report mean and standard deviation of the full dataset instead of grouped by species. The **means** of the main features are: **MinorAxisLength = 202.27**, **Compactness = 0.8**, **roundness = 0.87**. The **standard deviations** of the main features are: **MinorAxisLength = 44.97**, **Compactness = 0.06**, **roundness = 0.06**.

Because there is no strong linearity in the data I won't be using linear regression as a classifier. Additionally since there aren't strong groupings of data I won't be using k-nearest Neighbours or k-means clustering. Therefore I've decided to use the **following models** to classify the species data and to compare performance measures: **Logistic Regression**, **Decision trees** and **Linear Support Vector Machines** (SVM). I tested Gaussian and polynomial models of SVM. Gaussian produced the same accuracy of linear and polynomial performed worse than both Linear and Gaussian (~87%), so I went with the Linear model for the project. I was surprised as I thought a degree 7 polynomial would produce the best result since there are 7 species to be classified.

Model Performance

The model accuracy for the logistic regression, decision tree and SVM classification models were 0.91, 0.88 and 0.92 respectively.

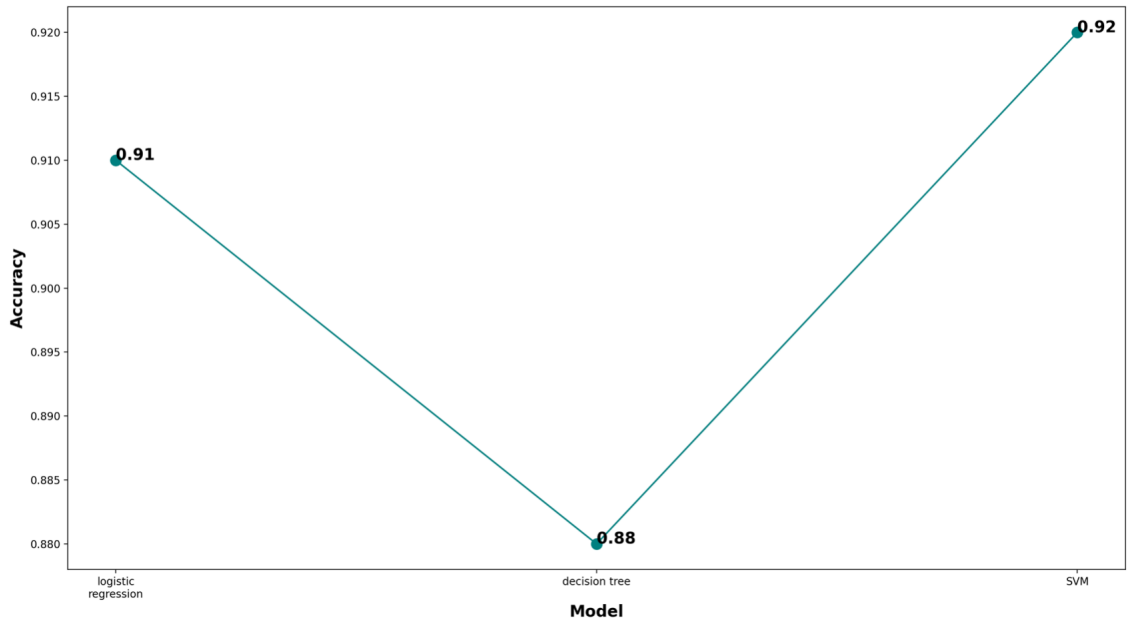

Fig 4. Model accuracy (logistic regression, decision tree, SVM)

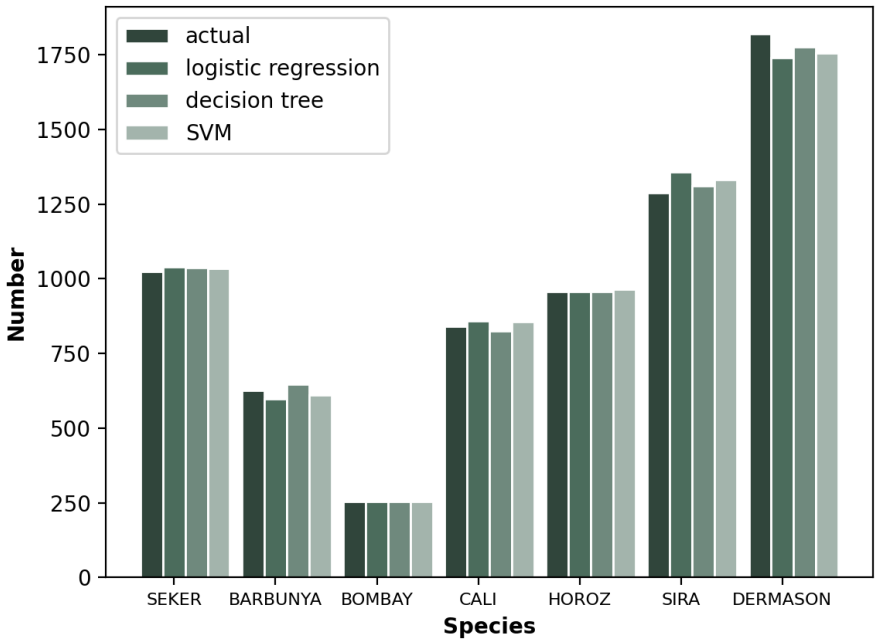

Fig 5. Model classification rate

At first I was a bit disappointed in the accuracy rates I obtained, since these values are not much different than the initial test accuracy I did on the logistic regression including all features. I felt like I went to the trouble of determining Shapley models for ideal features and there wasn't any improvement in accuracy. However after I plotted Fig. 5 which shows the numbers of the species the models classified the beans into, I realised there really isn't a huge difference in the classification rates regardless of the accuracy being on the lower end.

Program Instructions

I mostly used standard packages from numpy, pandas, seaborn and matplotlib for data handling and visualization. The dataset I used did not require any additional cleaning as it did not have any NaN values. The only preprocessing I did was to rename the species in the Class attribute to numbers so the models could process them more efficiently. The only other processing I did was using the Shapley (shap) module to determine which features were the most important. I had some trouble setting up the shap module in the traditional way using pip (I already have my PATH and PYTHONPATH set up). After struggling with it for a while the only way I could download it properly was by using conda and setting up a virtual environment. The instructions to setup and run this environment are on lines 32 - 35 of the program kanestor_project.py. To run the program you just run the environment commands in the terminal, then you run the program as normal.

e.g. (shap-env) kimberlynestor@Kimberlys-MacBook-Pro Project % python kanestor_project.py