# Dry Beans Classification Using Machine Learning

Grzegorz Słowiński

*University of Technology and Economics, ul. Jagiellońska 82f, 03-301 Warsaw, Poland*

**Abstract**

A dataset containing over 13k samples of dry beans geometric features is being analysed using machine learning (ML) and deep learning (DL) techniques with the goal to automatically classify the bean species. First the original dataset was reduced to eliminate redundant features (too strongly correlated and echoing others). Then the dataset was visualised and analysed with machine learning techniques: Multinomial Bayes, Support Vector Machines, Decision Tree, Random Forest, Voting Classifier and Artificial Neural Network. The overall accuracies obtained were in range: 88.35 – 93.61%.

**Keywords**

machine learning, deep learning, classification of dry beans.

## 1. Introduction

Classification of dry beans is of some economic importance. Manual classification is labour intensive, etc. Over 13 k samples of dry beans of 7 various species were photographed and their geometry was measured via computer vision techniques in [1]. Then the set was analysed via several machine learning (or data science) and deep learning (or artificial neural network) techniques. The overall accuracy obtained was 87.92-93.13%, depending on technique used.

The dataset used in [1] has been published in the UCI machine learning repository [2]. This work analyses the same dataset using slightly different techniques. Data dimensionality has been reduced. Slightly better accuracies has been achieved. Discussion and comparison to [1] has been carried out.

## 2. Tools

The entire analysis was done using Python and its ML frameworks: numpy, pandas, matplotlib, seaborn, scikit-learn and keras. Google Colab a free cloud version of jupyter notebook was used. The reader can find the Python scripts under link [3].

## 3. Preliminary analysis and visualisation of the dataset

The dataset under study consists of 13611 samples. A sample amounts to 16 geometrical features and a label identifying the species of the bean. The species are as follows: Barbunya, Bombay, Cali, Dermason, Horoz, Seker, and Sira. The features are: Area, Perimeter, MajorAxisLength, MinorAxisLength, AspectRatio, Eccentricity, ConvexArea, EquivDiameter, Extent, Solidity, Roundness, Compactness, ShapeFactor1, ShapeFactor2, ShapeFactor3, and ShapeFactor4. A detailed explanation how the features were calculated is presented in [1].

The geometrical data carry no information about the bean colour. From the practical point of view it is unfortunate, as different dry bean species tend to vary in colour. On the other hand, it makes little difference if we just want to treat the dry beans classification problem as an exercise in building and comparing machine learning models.

[1]29th International Workshop on Concurrency, Specification and Programming (CS&P'21)
EMAIL: grzegorz.slowinski@uth.edu.pl
ORCID: 0000-0001-9770-5063

## 3.1. Correlation analysis and feature reduction

Correlation analysis has shown that several of the features are strongly (positively or negatively) correlated. This is due to the fact that basically all of them are kind of geometric measures. The decision was taken to drop some features to avoid correlations over 0.9 (or negative correlation below -0.9) between them. The benefits of such a decision should be: 1) a significant reduction of the computational complexity 2) a lower risk of overfitting 3) ease of visualisation. The disadvantage is a limited risk of loosing some valuable information and, as a result, a decrease in accuracy.

**Table 1**
Correlation between selected beans features

| | MajorAxis Length | MinorAxis Length | Aspect-Ratio | Extent | Solidity | Roundness | Shape Factor2 | Shape Factor4 |
|---|---|---|---|---|---|---|---|---|
| MajorAxis Length | 1.0000 | 0.8261 | 0.5503 | -0.0781 | -0.2843 | -0.5964 | -0.8592 | -0.4825 |
| MinorAxis Length | 0.8261 | 1.0000 | -0.0092 | 0.1460 | -0.1558 | -0.2103 | -0.4713 | -0.2637 |
| AspectRatio | 0.5503 | -0.0092 | 1.0000 | -0.3702 | -0.2678 | -0.7670 | -0.8378 | -0.4493 |
| Extent | -0.0781 | 0.1460 | -0.3702 | 1.0000 | 0.1914 | 0.3444 | 0.2380 | 0.1485 |
| Solidity | -0.2843 | -0.1558 | -0.2678 | 0.1914 | 1.0000 | 0.6072 | 0.3436 | 0.7022 |
| Roundness | -0.5964 | -0.2103 | -0.7670 | 0.3444 | 0.6072 | 1.0000 | 0.7828 | 0.4721 |
| Shape-Factor2 | -0.8592 | -0.4713 | -0.8378 | 0.2380 | 0.3436 | 0.7828 | 1.0000 | 0.5299 |
| Shape-Factor4 | -0.4825 | -0.2637 | -0.4493 | 0.1485 | 0.7022 | 0.4721 | 0.5299 | 1.0000 |

Thus, in this work it was decided to limit the set of features list to these 8 members: MajorAxisLength, MinorAxisLength, AspectRatio, Extent, Solidity, Roundness, ShapeFactor2, ShapeFactor4, and to exclude: Area, Perimeter, Eccentricity, ConvexArea, EquivDiameter, Compactness, ShapeFactor1, ShapeFactor3. The issue of high correlations among some features was not addressed in [1]. The visualisation of the data was done by pair-plot, and is presented in figure 1.

It shows that the Bombay species is trivial to classify as its beans are significantly bigger than others. the classification of other species seems to be much more difficult, and we can expect more errors. The correlations between pairs of the selected features are listed in Table 1.

## 4. Machine Learning techniques used and results

In this work the following techniques were used: Multinomial Gaussian Classifier, Support Vector Classifier, Decision Tree, Random Forest, Voting Classifier, Artificial Neural Network (Multilayer Perceptron or MLP).

The full dataset was divided into the training and test subsets. 80% of samples were used for training and 20% for testing. Division of all available samples into the training and test subsets is crucial for a correct methodology. The aim of all ML or DL methods is to achieve a ”generalization” ability. Thus it is important to check the accuracy of classifying new samples, ones that have not been used during training. Otherwise, there is a very serious risk that the model will suffer form overfitting. Overfitted models perform very well on the training data but much worse on new data. Overfitting (as one of the most important issues in ML) is widely discussed in ML handbooks [4-6].

## 4.1. Multinomial Naive Bayes classifier

Naive Bayes models are based on Bayes's theorem. They are extremely fast and simple, but on the other hand, their performance is usually limited. They can be used as a baseline for classification problems (see [4], p. 382).

The overall accuracy obtained with Multinomial Bayes Classifier was 64,30 %. The problem was to classify into 7 different classes. Thus the blind (random) classification should result in about 1/7 =

14,29% accuracy. As one can see, classification is more difficult if there are more classes. Random classification should give accuracy equal to about 1/(number of classes). Thus we can see that even this simple model perform about 50 percent points better than the random approach.

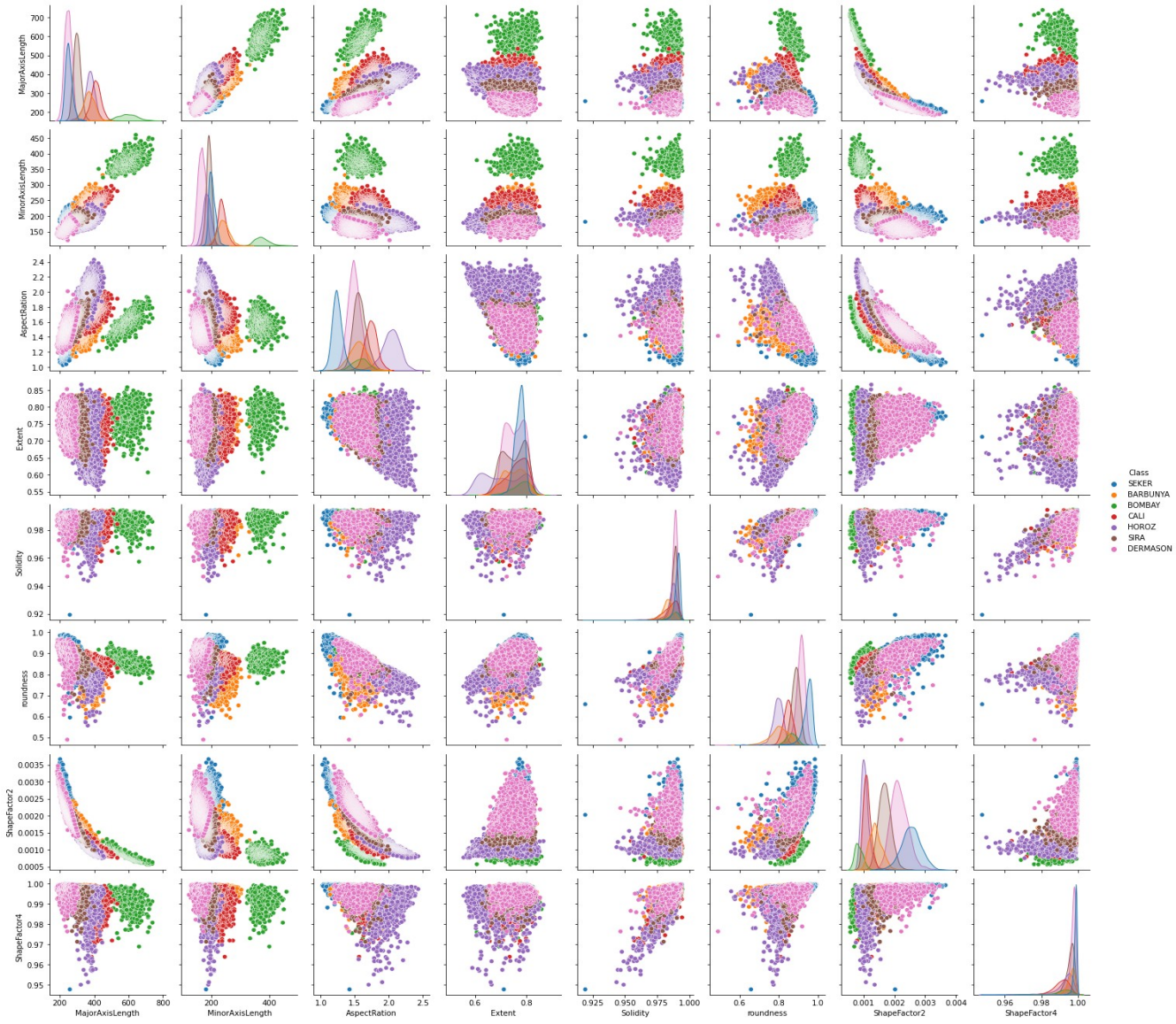


**Figure 1:** The selected features of dry beans (pairplot)

## 4.2. Support Vector Classifier

Support vector machines (SVM), which can be used as regressors or classifiers, are considered a very powerful and flexible algorithms. On the other hand. they may need a lot of computing power (see [4] p.405). The SVM principle is to partition the classes by ”drawing a line” (or plane) in a way that maximises the margin between classes. As straight lines (or planes) do not usually produce the best solution, SVC can apply different kernels (polynomial, radial and others). SVC is wider explained in [4,5]. SVCs with different kernels were tried. Table 2 presents the parameters used and the accuracy obtained.

The results are quite similar for all kernels. The accuracy can be further improved to some extent (tenths of %, maybe 1%) by increasing C, but this will also significantly increase the training time.

**Table 2**
The parameters for SVM (other parameters have default values).

| Kernel type | C parameter | Approx. computing time* | Overall accuracy |
|---|---|---|---|
| Linear function | $10^5$ | 41 s | 91.55% |
| Polynomial, degree=3 | $10^5$ | 21 s | 91.26% |
| Radial basis | $10^7$ | 34 s | 92.18% |

*- computation was done on colab: Intel(R) Xeon(R) CPU @ 2.20GHz and 12,69GB RAM

## 4.3. Decision Tree

A decision tree (DT) belongs to the class of so called non-parametric algorithms. The term non-parametric can be misleading. In fact, a decision tree has parameters, but their number is not constant.

During the learning phase, a decision tree tries to find the best questions partitioning the dataset in order to reduce information impurity (the measure is the Gini index or information entropy). The great advantage of decision trees is that they are extremely intuitive. On the other hand, a decision tree has no limited degrees of freedom, so it is easy to overfit (if the user is not aware of that). The splits made by a decision tree are always orthogonal (made on one feature at a time), so the decision tree is very sensitive to data rotation (see [5], p.188).

In [1] the authors created a decision tree with the depth of 4 (4 questions max) and 9 leaves. We decided to limit the depth of our decision tree to 5 and to 16 leaves max in order to get a decision tree that has size similar to DT obtained in [1].

Figure 2 shows the decision tree obtained under the above limits. The overall accuracy is 88.35%. Preliminary tests showed that a better accuracy of about 92,3% could be obtained with a bigger decision tree; however, the bigger the decision tree, the less intuitive it becomes, and the more difficult it is to visualise.

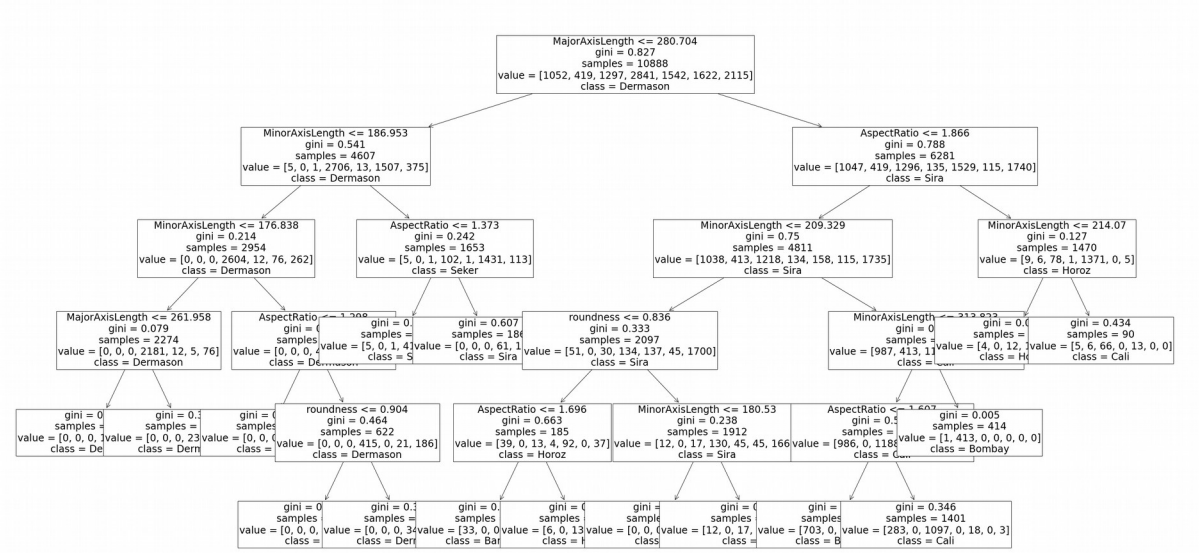


**Figure 2:** Visualisation of the obtained decision tree produced with the plot_tree method from sci-kit learn.

In another experiment with a big decision tree we set max depth =10 and max leaf nodes =30. The accuracy improved and reached 91.59% (Table 3).

**Table 3**
Decision trees parameters and performance

| Decision tree | Max depth | Max leaf nodes | Overall accuracy |
|---|---|---|---|
| small | 5 | 16 | 88.35% |
| big | 10 | 30 | 91.59% |

It can be seen that to obtain an accuracy similar to that reported in [1] with a decision tree, the tree would have to be much bigger (losing the main advantage of decision trees, i.e., the intuitive interpretation). One should keep in mind that in this work the amount of features has been reduced from 16 to 8. The excluded features were highly correlated to the retained features (being other geometrical measures of the same beans), so they accounted for little additional information. However, they present this information in a slightly different manner (”rotated”), making the task easier for the decision tree.

To see this better, assume that in some dataset we have two parameters A and B, and the classification is obvious, but it depends on the A/B ratio that is not explicitly present in the dataset. This can be hard for a decision tree to solve. The addition of an extra column, A/B, will add no new information to the dataset, but it will help the decision tree quite a bit. It can be supposed that in the dry bean case, the 8 removed categories contained little extra information, but they presented essentially the same information in a way more appropriate for the decision tree.

## 4.4. Random Forest

The random forest idea is as follows: take many decision trees (employing some randomness, so the trees differ) and let them vote. So the classification decision taken by a random forest is a decision taken by the most numerous group of decision trees in a random set of trees.

Usually a random forest performs better than a single decision tree. However, a random forest is considered a ”black-box” model being very hard to interpret.

A random forest of 150 decision trees was created. No restrictions on trees were applied. The accuracy obtained was 93.61%, the best so-far, better than the best accuracy reported in [1]. In addition, the training process was fast and took about 2 s, which 10-20 times faster than for SVC.

## 4.5. Voting Classifier

The idea of ”voting”, which by default is used in random forests, can be applied to any classifiers. There are 2 main ways of voting: ”hard” (straightforward, direct voting) and ”soft” (the votes are weighted depending on how confident the classifier is with its choice). Like in the case of a random forest, there is a good chance that the voting result will be more accurate than for any particular classifier.

The hard voting classifier was implemented using 3 classifiers described above: the radial kernel SVC, the ”big” decision tree, and the random forest.

The obtained accuracy was 92.80%. Thus in this case it is worse than for the random forest. This gives us a clue that voting should be used carefully and preferably with models exhibiting similar performance; otherwise ”stupid” models can outvote ”smart” models. It seems that this flaw of democracy does not only apply to human societies, but is more universal in nature.

## 4.6. Artificial Neural Network

Besides the (shallow) machine learning/data science methods presented above, a deep learning technique, the so-called dense artificial neural network, has also been tried.

For an artificial neural network the data needs additional treatment. First, the names of bean species were labelled with numbers and then these numbers 0-6 were codded as so called ”one-hot”.

The reason of using ”one-hot” encoding is well explained for example in [4] p. 376 or [6] pp. 190-194. The other operation is scaling, a standardisation or normalisation of the training data. The data (each feature) is centred around zero (by subtracting the average) and normalised (by dividing by the standard deviation). Standardisation is said to ease the training process and tends to bring in improvement in performance [5] p. 72.

Two architectures of ANN were tried. The first one is similar to the one described in [1], except that the input layer size in our case is 8 not 16. The network has 3 hidden layers with 17, 12, 3, neurons, respectively. Rectified Linear Unit was used as an activation function in hidden layers. The output layer has 7 neurons, one for each bean species. Sigmoid is the activation function for the output layer. The optimiser used was: RMSprop, and the loss function was the categorical cross entropy. The validation set was 20% of training set. The network was trained for 24 epochs. The architecture of this network and the training process are presented in figure 3.

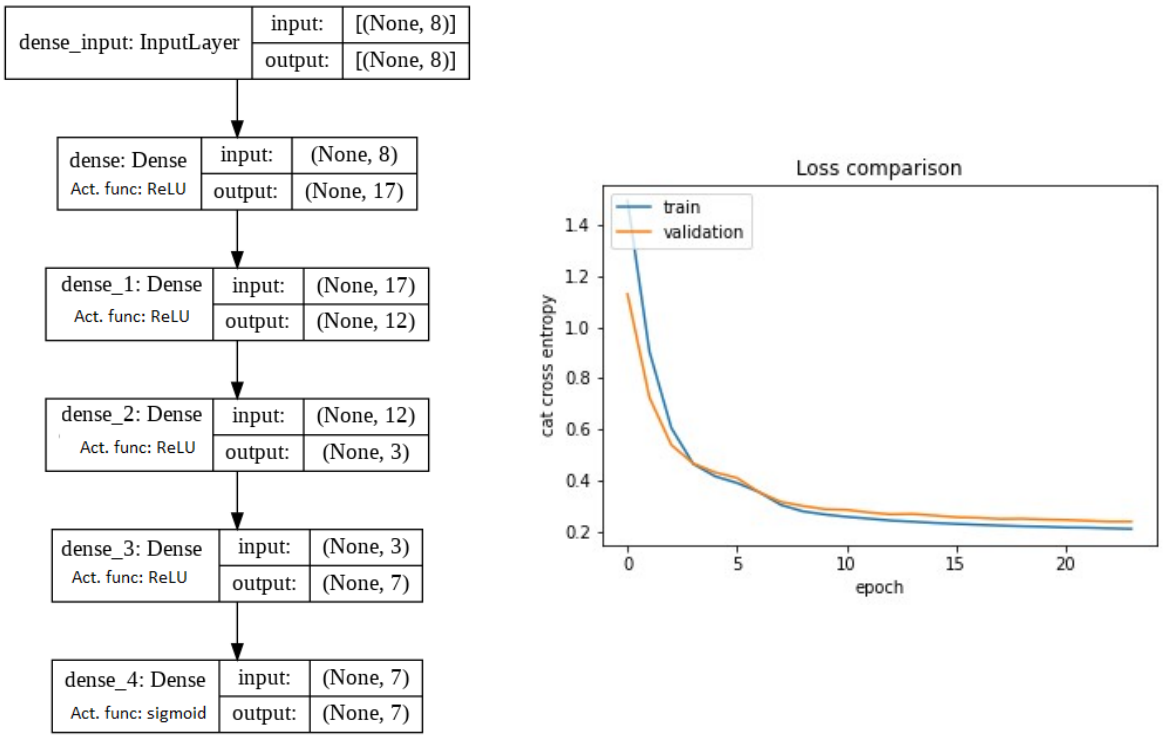


**Figure 3.** The architecture (left) and the training process (right) of the first ANN.

The overall accuracy was 92.58%. In an attempt to improve it, another ”bigger” ANN was tried. Besides the normal layers, a dropout layer was added. A dropout layer only works during the training and randomly ”cuts off” (sets to zero) some inputs. It is expected that dropout layers reduce the risk of overfitting [6] p.109. The architecture of the network and its training are shown in figure 4. The same optimiser and loss function were used: RMSprop and the categorical cross entropy, respectively. The network was also trained for 24 epochs. The overall accuracy was 92.77%, thus the improvement was not much.

## 5. Results and Conclusions

The dry beans dataset has been analysed by different machine learning and deep learning techniques. Table 4 shows the summary results.

It can be seen that in general the task of beans classification is a relatively simple task in terms of the necessary computing power. All training times were shorter than 1 minute using a free google-colab computer.

The accuracy of Naive Bayes is much worse than for the other methods. This is not surprising, as this method is known to be fast but not very accurate, and is suggested as a preliminary method to check if there is „something" in the data, rather than to do a full analysis. The other methods give accuracy in a relatively close range of 88.3-93.6%. This is comparable to [1] where the accuracy was in range 87.9-93.13%.

The authors in [1] trained their models with all 16 features. Here, some strongly correlated features were eliminated. The results show that this elimination has not decreased the accuracy.

The only technique, that suffered from this elimination to some extend seems to be decision tree. This is due to the fact that decision tree operates on one feature at a time and cannot "combine" features. See also the discussion in p. 4.3 devoted to the decision tree section.

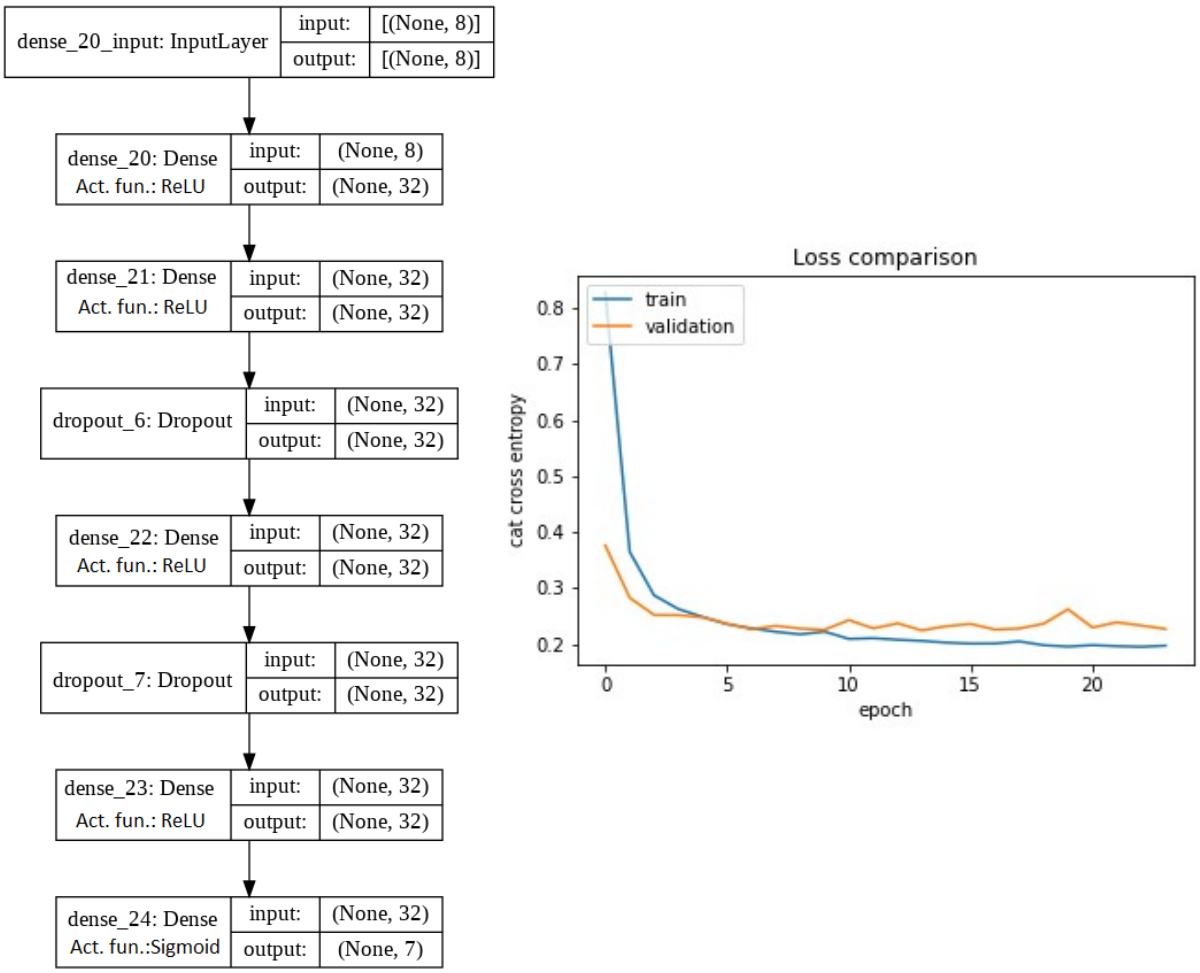


**Figure 4.** The architecture (left) and the training process (right) of the second ANN.

One can see that the models vary strongly in the terms of the computing time. SVC and ANN (and also Voting Classifier, because it includes SVC) are the slowest learners. Random forest seems to be the best method in this case, as it perform best and its training time is also reasonable.

Confusion matrices provide a comfortable way to visualise results in more details and compare actual values with predicted ones. The confusion matrix for the random forest classifier (the best performer) will be discussed further. It is presented in figure 5. The most frequent mistakes were between Dermason and Sira (38 + 44). On the other hand, Bombay was classified perfectly which is not surprising. It is easy to notice that Bombay beans are significantly bigger than other species.

The dry beans dataset appeared to be an interesting dataset to demonstrate and compare ML techniques. Two ideas for further research:

- Deeper insight how and if the elimination of correlated features influences the ML training process. This study shows that there is little, if any, performance decrease. One may try to investigate if the elimination reduces the training time and how much.
- Despite ”manual” feature reduction, as done in this work, on may try to use PCA (the primary component analysis) to reduce the dimensionality of data and also analyse its influence on model performance (accuracy and training time)

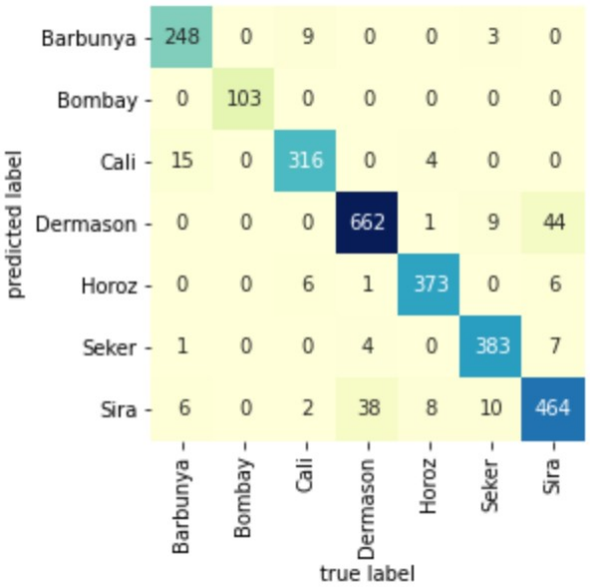


**Figure 5**. Test subset confusion matrix for the random forest classifier.

## 6. References


[1] Murat Koklu, Ilker Ali Ozkan, Multiclass classification of dry beans using computer vision and machine learning techniques, Computers and Electronics in Agriculture 174 (2020) 105507

[2] Dry beans dataset at UCI repository: https://archive.ics.uci.edu/ml/datasets/Dry+Bean+Dataset, access 23.06.2021

[3] Colab notebook containing computation scripts for this work: https://colab.research.google.com/drive/11X6VevSMybenGkRqK1Xj_1EJmU3vomFB?usp=sharing

[4] Jake VanderPlass, Python Data Science Handbook, O’Reilly, 2017

[5] Aurelien Geron, Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow, O’Reilly, 2019

[6] Francois Chollet, Deep Learning with Python, Manning Publications, 2018