Prediction of Dry Beans Category using Support Vector Machine,Naïve Bayes and Decision Tree Algorithms

Anu Joseph

*Abstract*— This paper is about predicting the category or class label of dry beans on the basis of their features by using different machine learning classification algorithms such as Support vector machine, Naïve Baye’s and Decision tree. Some data preparation techniques like null value check, balancing of data, low variance data removing, data standardization and features and dimensionality reduction methods are implemented to make the data suitable for algorithms. Python codes are used to analyze and visualize the results of algorithms

Keywords— machine learning algorithms; python code; class imbalance; data preparation; support vector machine; naïve bayes; decision tree; dry beans class labeling

# Introduction

# For bean growers and the market, choosing the best seed species is one of the top priorities. Due to the fact that various genotypes are grown all over the world, it is crucial to distinguish the best seed variety from the mixed dry bean population; otherwise, the market price of these mixed species of beans could drastically decline. Here, machine learning techniques can be employed to obtain results more quickly than through human analysis. Numerous elements contributed to the seed species being difficult for humans to see and predict. A machine learning technique uses labelled input data to train a function that, when presented with fresh, unlabelled data, provides the desired result.

# the Data set

The data set is taken from UCI machine learning repository. It contains 13,611 grain samples from 7 different registered dry bean varieties that were captured using a high-resolution camera. The grains yielded a total of 16 characteristics, including 12 dimensions and 4 shape types.

TABLE 1 describes the features included in the data set.

Table 1: Data set features

|  |  |  |  |
| --- | --- | --- | --- |
| No | Description | Type | Categorical value range |
| 1 | Area | Numerical |  |
| 2 | Perimeter | Numerical |  |
| 3 | Major Axis Length | Numerical |  |
| 4 | Minor Axis Length | Numerical |  |
| 5 | Aspect Ration | Numerical |  |
| 6 | Eccentricity | Numerical |  |
| 7 | Convex Area | Numerical |  |
| 8 | Equivalent Diameter | Numerical |  |
| 9 | Extent | Numerical |  |
| 10 | Solidity | Numerical |  |
| 11 | Roundness | Numerical |  |
| 12 | Compactness | Numerical |  |
| 13 | Shape factor 1 | Numerical |  |
| 14 | Shape factor 2 | Numerical |  |
| 15 | Shape factor 3 | Numerical |  |
| 16 | Shape factor 4 | Numerical |  |
| 17 | Class | Categorical | SEKER,BARBUNYA,  BOMBAY,CALI,  HOROZ,SIRA,  DERMASON |

# data set preparation

This section covers the data preparation techniques. Measures the balance of the specified data in the data collection using data imbalancing procedures. Through the use of missing value checks, null values and missing values are identified, allowing for the accurate prediction of data results. Low variance analysis identifies instances that can be avoided. Character data is transformed into binary representation through categorical feature encoding. The best features from the data collection are determined via dimensionality and feature selection methods.

## Class Imbalance

There is a class imbalance when observation in one class is higher than observation in other classes. It is common in machine learning especially classification problems. It will produce high accuracy than actual and mislead us to wrong class predictions. In our data set there are 7 classes of dry bean species each of having different frequencies. The imbalanced state is figured out by the graph in Figure 1.

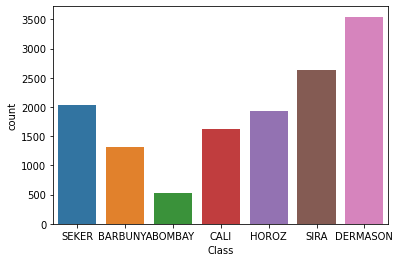


Figure 1: class imbalance

To overcome this, three techniques are used.

1. Random undersampling

Undersampling is the practice of omitting a portion of observations from the majority class. This process is repeated until the majority and minority classes are equal. In random Undersampling the omission is done randomly. One of its shortcomings is the potential for this random omission to delete important data. As seen in Figure 1, there are 7 class labels in this project, each with a distinct number of members. BOBMBAY (522) is the class with the fewest members, and DERMASON has the most (3546). All class counts after random undersampling will be equal to the class that is least counted. ie, 522. A new data set with 3654 instances will be the result of random undersampling.

1. Random oversampling

Increasing the number of copies of the minority class is the definition of oversampling. This is the antithesis of undersampling. After oversampling, the data set with 13,611 cases increases to 24,822 instances. The class label with the highest count (3546) was evenly distributed throughout the other classes as well.

1. SMOTE Oversampling

For the minority class, this strategy produces fake data. By selecting a point at random from the minority class and calculating its k-nearest neighbours, SMOTE (Synthetic Minority Oversampling Technique) operates. Between the specified point and its neighbours, the artificial points are inserted. The technique is represented in Figure 2.

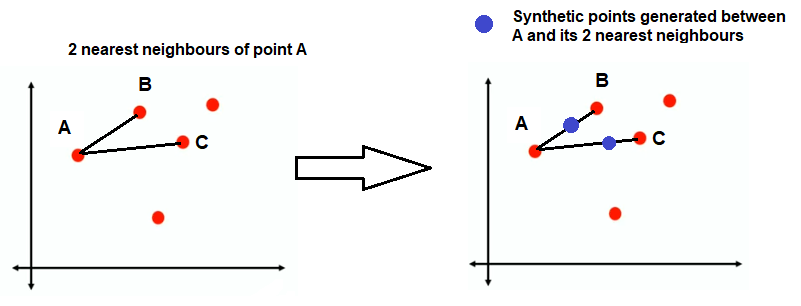


Figure 2: SMOTE technique

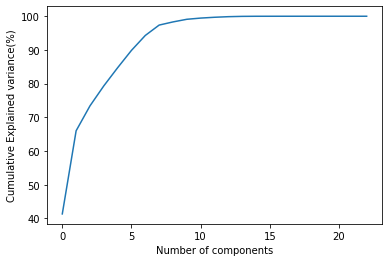
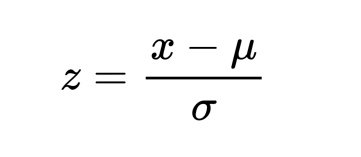
Here, for the grain categories except DERMASON, synthetic points are calculated using k-nearest neighbour concept and the data set resulted contains 24,822 instances. Python coding with necessary libraries like imblearn is used to implement the code.

## Categorical Feature Encoding

As we are aware, the majority of data in real life are categorical string values, and the majority of machine learning models only operate with integer values. However, certain models can also work with other values that are intelligible to the model. In their most basic form, all models carry out mathematical operations that can be carried out using a variety of instruments and methods. However, it is an unfortunate fact that mathematics is entirely dependent on numbers. So, in summary, we can conclude that most models demand numbers as the data, not words or anything else, and these numbers can be float or integer. Data set considering here involves a categorical data which is termed as Class, includes the grain categories of dry beans. This categorical data is converted to binary using OneHotEncoder in python library scikit-learn.

## Data Standardization

Standardization, a scaling approach, transforms the statistical distribution of the data into the following form, rendering the data scale-free:

* mean - 0 (zero)
* standard deviation – 1

By doing this the data set will scale between unit variance and zero mean. Python sklearn library offers a method StandardScaler from preprocessing module is used to code this.

## Dimensionality reduction and features selection

Dimensionality reduction is the process of minimizing the number of attributes in a dataset while retaining as much variance as is feasible in the original dataset. Prior to training the model, we do dimensionality reduction as part of the data pre-treatment procedure. If the number of dimensions are less, training data time can reduced and better result will produced through machine learning algorithms. So it’s necessary to perform dimensionality reduction techniques. In this project 3 techniques are tried to plot the results.

1. LDA (Linear Discriminant Analysis)

It is commonly used for classification of multi-class data sets. LDA is a linear technique to reduce dimensions. Python scikit-learn library with module discriminant analysis is used to implement the code. The disadvantage of plotting class labels is that they cannot be separated after plotting. Data is categorised by LDA in a way that miniggmises its separability. So it’s not the best method to use in dry beans class prediction problem.

1. PCA (Principal Component Analysis)

It’s a remedial method to LDA. In a dataset, PCA looks for a group of uncorrelated components with the highest variance. The resultant plots of dry bean data set for PCA in are Figure 3 and Figure 4

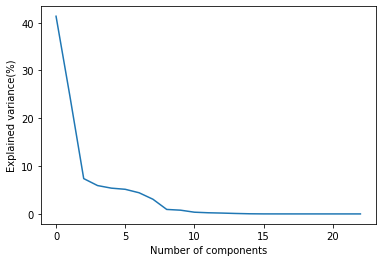


Figure 3: PCA of Explained Variance

Figure 4: PCA of Cumulative Explained Variance

1. Kernel PCA

PCA converts dimensions linearly and is widely used for large applications. In machine learning we may need to transform data which is non- linear, kernel PCA helps to perform non-linear applications. Resultant plots for dry bean data set by Kernel PCA are in Figure 5 and Figure 6.

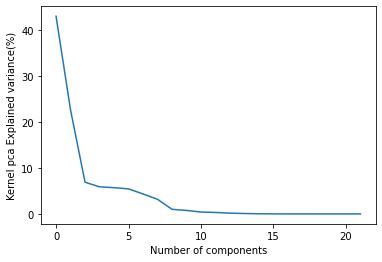


Figure 5: Kernel PCA of explained ratio

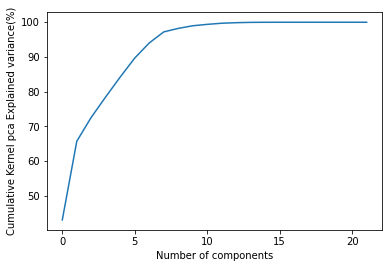
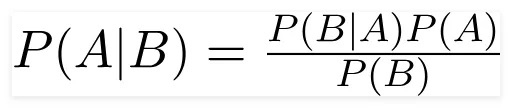


Figure 6: Kernel PCA of cumulative explained ratio

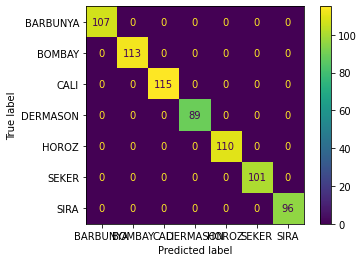
Analysing the results of both PCA and Kernel PCA it’s clear that both are same. There are 17 dimensions in the data set, by both of this methods data shrinked into 8 attribute contributing 97% towards the cumulative explained variance. In this project we focus on PCA than Kernel PCA due to less execution time. Both codes for PCA and Kernel PCA are implemented through scikit-learn library and corresponding methods.

# machine learning classification techniques

## Naïve Bayes Algorithm

Using probabilistic machine learning, a Naive Bayes classifier performs classification tasks. The Bayes theorem serves as the classifier's foundation. The theorem is:

## Decision Tree Algorithm

A non-parametric supervised learning approach called a decision tree is used for both classification and regression applications. A root node, branches, internal nodes, and leaf nodes make up its hierarchical tree structure. It works under different criteria like entropy and gini to find the root node and branches efficiently.

## Support Vector Machine

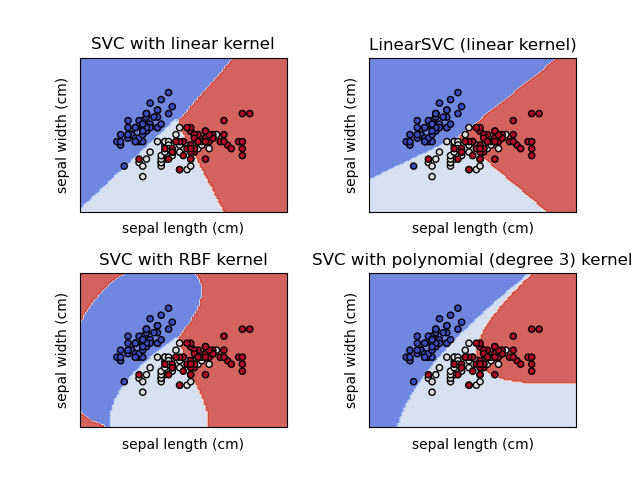
It is a supervised learning method used for classification, regression and outlier detection. SVM works well for data sets with high dimensions and lower ones. Kernels like rbf, linear are used to perform the decision function. It uses the train set for decision function so that memory usage can be reduced. Figure 7 shows different kernel implementations in SVM.

Figure 7: SVM with different kernels

Figure is just to understand the kernel in SVM. It’s not the representation of dry bean data set.

# application of techniques and results

Machine learning techniques and results are performed through HP LAPTOP-H4FUMO8M with Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz processor specifications and 8GB RAM.

Techniques are trained through parameters from the data set. Confusion matrix and accuracy score has been resulted for each classifier model, which further taken for analysis. All models were trained using 16 parameters. For each model 3 types of evaluations are performed corresponding to random over sampling, random under sampling and SMOTE.

## Naïve Bayes

This classifier resulted confusion matrix for all the three class imbalancing techniques are in Figure 8, Figure 9 and Figure 10. This result contributed by the 8 parameters in the data set. The accuracy score is different for each technique.

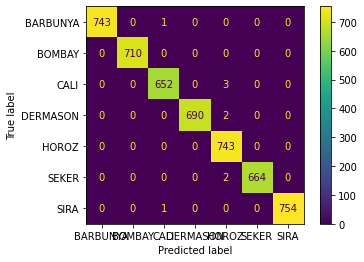
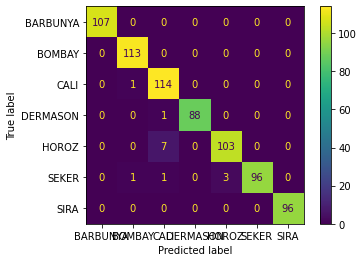
Figure 8: Under Sample Confusion matrix for Naïve Bayes classifier

Figure 9: Over Sample Confusion Matrix for Naïve Bayes classifier

Figure 10: SMOTE Confusion Matrix for Naïve Bayes classifier

## Decision Tree

The confusion matrix of decision tree classifier for all the three class imbalancing techniques are in Figure 11, Figure 12 and Figure 13. This result contributed by 8 parameters in the data set. We can use either entropy or gini as criteria to perform the decision function. In this project the accuracy score for entropy is greater than that of gini, so the confusion matrix of entropy is considered.



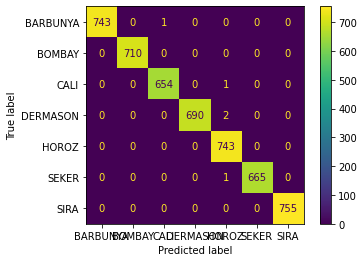
Figure 11: Under Sampling confusion matrix for Decision tree classifier

Figure 12: Over Sampling confusion matrix for Decision tree classifier

Figure 12: Over Sampling confusion matrix for Decision tree classifier

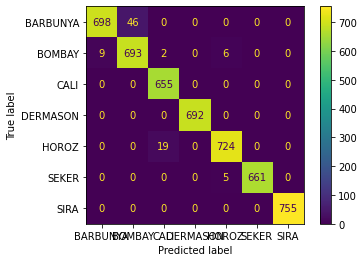
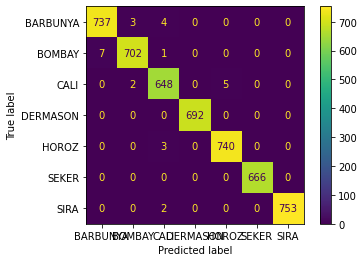
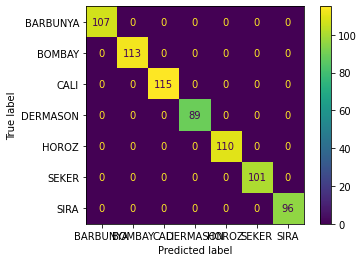


Figure 13: SMOTE Confusion matrix for Decision tree classifier

## Support Vector Machine

SVM with kernel rbf ,gamma value 0.01 and c=1000 is used for prediction in this data set consideration. Kernel can be either rbf or linear, here both results the same accuracy. The confusion matrix for kernel rbf is plotted in Figure 14, 15 and 16 respectively.



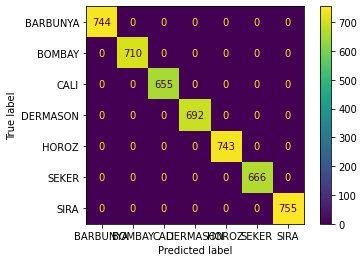
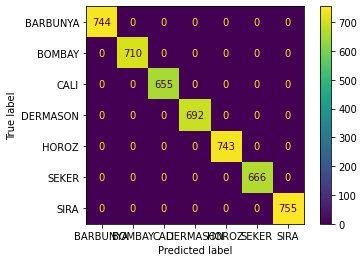
Figure 14: Under Sampling confusion matrix for SVM classifier

Figure 15: Over Sampling confusion matrix for SVM classifier

Figure 16: SMOTE confusion matrix for SVM classifier

# Conclusion

The accuracy score for each imbalancing technique of classifier models are listed in the Table 2.

|  |  |  |  |
| --- | --- | --- | --- |
| No | Imbalancing technique | Classifier model | Accuracy score |
| 1 | Random under sampling | Naïve Bayes | 99.3% |
| Decision Tree | 98.1% |
| SVM | 100% |
| 2 | Random Over Sampling | Naïve Bayes | 99.8% |
| Decision Tree | 99.5% |
| SVM | 100% |
| 3 | SMOTE | Naïve Bayes | 99.2% |
| Decision Tree | 98.2% |
| SVM | 100% |

By analysing the result it’s clear that on all the imbalancing techniques over sampling produces best accuracy score compared to other two. The classifier SVM produces the best result among the three. Even though the results for all the three classifiers are nearly same 100% accuracy score is produced by SVM. In this project SVM is identified as the best classifier for the data set’s class prediction.

# Reference

<https://rukshanpramoditha.medium.com/principal-component-analysis-pca-with-scikit-learn-1e84a0c731b0>

<https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/>

<https://www.digitalocean.com/community/tutorials/standardscaler-function-in-python>

<https://pbpython.com/categorical-encoding.html>

<https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c#:~:text=A%20Naive%20Bayes%20classifier%20is,based%20on%20the%20Bayes%20theorem>.

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

[Support Vector Machines (SVM) in Python with Sklearn • datagy](https://datagy.io/python-support-vector-machines/)

[Python | Decision tree implementation - GeeksforGeeks](https://www.geeksforgeeks.org/decision-tree-implementation-python/)

[Naive Bayes Classifier with Python - AskPython](https://www.askpython.com/python/examples/naive-bayes-classifier)

https://towardsdatascience.com/11-dimensionality-reduction-techniques-you-should-know-in-2021-dcb9500d388b