Applying Data Science Principles to classify plant species

Tomide Victor Afolabi
Afolabi-t@ulster.ac.uk
School of Computing, Engineering and Intelligent Systems
Ulster University
Londonderry, United Kingdom

Abstract*— This study aimed to utilise data science principles to classify plant species based on ecological and morphological features. The plant dataset included 340 instances with heterogeneous variables, and machine learning algorithms such as k-Nearest Neighbours (kNN), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), and Naïve Bayes (NB) were employed to establish a classification model. The study also analysed the significant features crucial to plant classification, which could benefit various fields such as plant taxonomy and biodiversity. The performance of each model was evaluated and compared using various metrics such as confusion matrix, prediction accuracy, Recall, F-score, and so on, between two different types of plant species datasets. The study found that RF was the most accurate algorithm for classification problems on the two types of datasets, with 0.846 (ACC), 0.991 (AUROC), and 0.846 (F-score). Moreover, the majority of algorithms, except for KNN, showed improvement in the dataset containing all 15 variables.

Keywords: Data science, Machine learning, multi-class classification, confusion matrix, performance evaluation metrics, prediction accuracy,

I. INTRODUCTION

The widespread adoption of information technology and easy access to data has led to the generation of vast amounts of data, commonly known as big data. However, managing and storing such large amounts of data has presented significant challenges to companies. This data is often characterised by its raw, dirty, incorrect, incomplete, and imbalanced nature. Fortunately, the emergence of data science has been instrumental in proffering solutions to these challenges by extracting useful insights to make informed decisions (1).

The extraction of information from data requires several stages, including data collection, data preprocessing, exploratory data analysis, feature engineering, model selection and training, model evaluation, and deployment. However, data collected from various sources may contain errors, inconsistencies, and missing values, which can negatively affect the performance of a model. Therefore, it is crucial to apply data science principles to handle the inherent challenges presented by dirty data. Failure to preprocess data can result in a poorly developed model, leading to suboptimal decisions. This highlights the importance of applying data science principles in data analysis to ensure accurate and reliable results.
Machine learning (ML) is a field within the broader discipline of data science that involves creating algorithms and statistical models that improve a computer system's performance on a specific task with minimal programming instructions. These algorithms are designed to recognize patterns and relationships in data, and they learn to make predictions or decisions on new data based on this knowledge. There are several types of machine learning algorithms, including supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

In the context of this study, the problem at hand pertains to multi-class classification, with the target variable comprising 36 potential answers to the classification problem.

Shangzhou_Wang, et al (2022), programmatically made use of a published article on a multi-class classification problem. Evaluation and analysis were done on the performances of five machine learning (ML) algorithms namely: Logistic regression (LR), *k*-nearest neighbours (KNN), Naïve Bayes (NB), Random forest (RF) and eXtreme Gradient Boosting (XGBoost) and two deep learning models (Multilayer perceptrons (MLP) and Convolutional neural networks (CNN)) using techniques like feature importance, features correlation, confusion matrix, variable clustering and kernel density estimation. After all ML algorithms and deep learning models used were compared, eXtreme Gradient Boosting (XGBoost) was the best model (2).

F. Lazzeri and S. Penchikala (2019) established 6 principles for developing healthy data-driven organisations. Architect the End-to-End Solution of Principle 3 explained the essence of DS principles. the importance of DS principles to researchers in data analysis (3).

Lei Shi et al, (2022), presented new algorithms namely: the feature ranking method and instance filter, to make a multi-class classification prediction on an agricultural dataset, which was introduced to enhance the capability of Random forest, an ML algorithm. Feature selection was also employed. The new algorithm was evaluated and tested. The outcome indicated that the new approach exhibited a satisfactory performance. (4).

This research contributes to the identification of previously unknown plant species, as well as those with potential dangers or benefits to human health, which can aid in the early detection of such plants and support the development of medicines or other useful applications. The objective of the study is to utilise principles of data science to solve the classification problem of plant species. Several machine learning algorithms such as K-Nearest Neighbors (kNN), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), and Naïve Bayes (NB) were implemented to categorise the plant dataset, and their performance was analysed and compared.

II. DATA DESCRIPTION

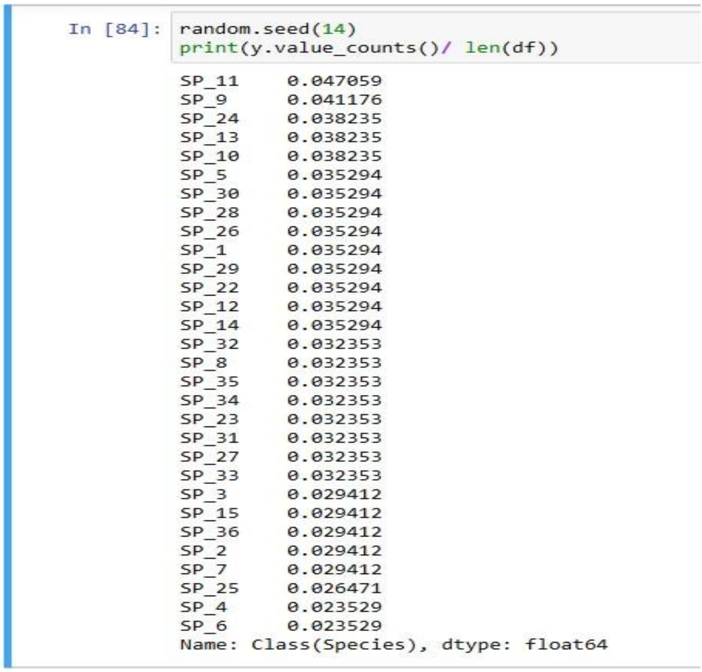
The Leaf dataset used in this study was obtained from the UC Irvine machine learning repository and was originally created by Pedro F. B. Silva and André R. S. Marçal, based on leaf specimens collected by Rubim Almeida da Silva at the University of Porto, Portugal. The initial dataset comprised only 340 instances and was in poor condition. However, after undergoing preprocessing, the data was transformed to include 15 attributes and 480 instances, which were used in the analysis.

The target variable (Species) was categorised into 36 segments representing plant species from SL_1 to SL_36.

Variable type	Name	Type	Completeness
Explanatory	1)Specimen_Numbe	Numeric	Yes
	2) Eccentricity	Numeric	Yes
	3) Asp_Ratio	Numeric	Yes
	4) Elongation	Numeric	Yes
	5) Solidity	Numeric	Yes
	6) Stoc_Convexity	Numeric	Yes
	7) Isop_Factor	Numeric	Yes
	8) Max_Ind_Depth	Numeric	Yes
	9) Lobedness	Numeric	Yes
	10) Ave_Intensity	Numeric	Yes
	11) Ave_Contrast	Numeric	Yes
	12) Smoothness	Numeric	Yes
	13) Thir_moment	Numeric	Yes
	14) Uniformity	Numeric	Yes
	15) Entropy	Numeric	Yes
Target	16) Class(Species)	Nominal	Yes

Table 1, Data description.

Fig 1 shows that the initial analysis of the dataset indicated that the output class ratio (OCR) was imbalanced. However, this issue was addressed by using SMOTE (Synthetic Minority Oversampling Technique) to balance the data. As a result of applying SMOTE, the OCR became equal for all possible values of the target variable.



```
In [84]: random.seed(14)
         print(y.value_counts()/ len(df))

SP_11    0.047059
SP_9     0.041176
SP_24    0.038235
SP_13    0.038235
SP_10    0.038235
SP_5     0.035294
SP_30    0.035294
SP_28    0.035294
SP_26    0.035294
SP_1     0.035294
SP_29    0.035294
SP_22    0.035294
SP_12    0.035294
SP_14    0.035294
SP_32    0.032353
SP_8     0.032353
SP_35    0.032353
SP_34    0.032353
SP_23    0.032353
SP_31    0.032353
SP_27    0.032353
SP_33    0.032353
SP_3     0.029412
SP_15    0.029412
SP_36    0.029412
SP_2     0.029412
SP_7     0.029412
SP_25    0.026471
SP_4     0.023529
SP_6     0.023529
Name: Class(Species), dtype: float64
```

Fig1. Output class ratio

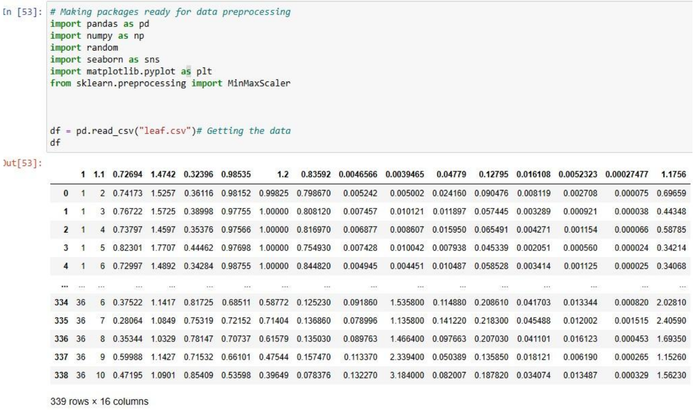
III. METHODOLOGY

A. Jupyter Notebook

It was formerly known as the Ipython Notebook. It is a software browser-based application for computing, visualising, analysing and so on. It contains input and output cells that further contain codes, texts, plots and so on, [5].

B. Python

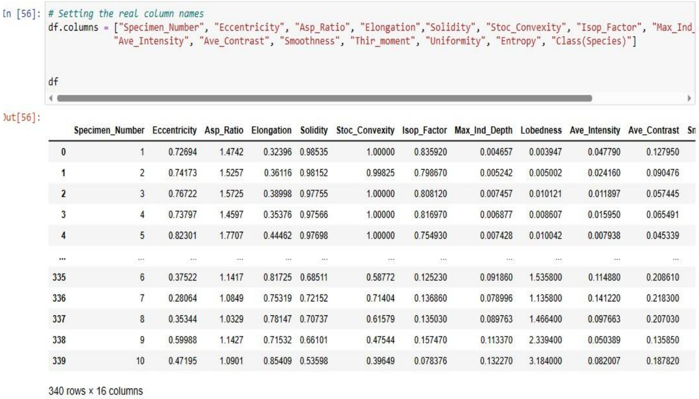
Python is a programming language that allows for the automation of tasks and data analysis [6]. It includes numerous packages and libraries that are useful for real-world data analysis. In this study, data preprocessing (DP) was performed using Python within the Jupyter Notebook environment. Some of the Packages used are pandas, numpy, and min-max scaler.



```
In [53]: # Making packages ready for data preprocessing
         import pandas as pd
         import numpy as np
         import random
         import seaborn as sns
         import matplotlib.pyplot as plt
         from sklearn.preprocessing import MinMaxScaler



         df = pd.read_csv("leaf.csv")# Getting the data
         df
```

```
Out[53]:
        1  1.1  0.72694  1.4742  0.32396  0.98535      1.2   0.83592  0.0046566  0.0039465   0.04779   0.12795  0.016108  0.0052323  0.00027477   1.1756
  0     1    2  0.74173  1.5257  0.36116  0.98152  0.99825  0.798670   0.005242   0.005002  0.024160  0.090476  0.008119   0.002708    0.000075  0.69659
  1     1    3  0.76722  1.5725  0.38998  0.97755  1.00000  0.808120   0.007457   0.010121  0.011897  0.057445  0.003289   0.000921    0.000038  0.44348
  2     1    4  0.73797  1.4597  0.35376  0.97566  1.00000  0.816970   0.006877   0.008607  0.015950  0.065491  0.004271   0.001154    0.000066  0.58785
  3     1    5  0.82301  1.7707  0.44462  0.97698  1.00000  0.754930   0.007428   0.010042  0.007938  0.045339  0.002051   0.000560    0.000024  0.34214
  4     1    6  0.72997  1.4892  0.34284  0.98755  1.00000  0.844820   0.004945   0.004451  0.010487  0.058528  0.003414   0.001125    0.000025  0.34068
...   ...  ...      ...     ...      ...      ...      ...       ...        ...        ...       ...       ...       ...        ...         ...      ...
334    36    6  0.37522  1.1417  0.81725  0.68511  0.58772  0.125230   0.091860   1.535800  0.114880  0.208610  0.041703   0.013344    0.000820  2.02810
335    36    7  0.28064  1.0849  0.75319  0.72152  0.71404  0.136860   0.078996   1.135800  0.141220  0.218300  0.045488   0.012002    0.001515  2.40590
336    36    8  0.35344  1.0329  0.78147  0.70737  0.61579  0.135030   0.089763   1.466400  0.097663  0.207030  0.041101   0.016123    0.000453  1.69350
337    36    9  0.59988  1.1427  0.71532  0.66101  0.47544  0.157470   0.113370   2.339400  0.050389  0.135850  0.018121   0.006190    0.000265  1.15260
338    36   10  0.47195  1.0901  0.85409  0.53598  0.39649  0.078376   0.132270   3.184000  0.082007  0.187820  0.034074   0.013487    0.000329  1.56230

339 rows × 16 columns
```

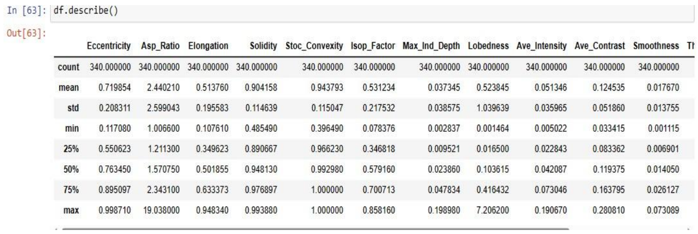
Fig 2. Reading the data.



```
In [56]: # Setting the real column names
         df.columns = ["Specimen_Number", "Eccentricity", "Asp_Ratio", "Elongation","Solidity", "Stoc_Convexity", "Isop_Factor", "Max_Ind_
                       "Ave_Intensity", "Ave_Contrast", "Smoothness", "Thir_moment", "Uniformity", "Entropy", "Class(Species)"]


         df
```

```
Out[56]:
     Specimen_Number  Eccentricity  Asp_Ratio  Elongation  Solidity  Stoc_Convexity  Isop_Factor  Max_Ind_Depth  Lobedness  Ave_Intensity  Ave_Contrast  Sn
  0                1       0.72694     1.4742     0.32396   0.98535         1.00000     0.835920       0.004657   0.003947       0.047790      0.127950
  1                2       0.74173     1.5257     0.36116   0.98152         0.99825     0.798670       0.005242   0.005002       0.024160      0.090476
  2                3       0.76722     1.5725     0.38998   0.97755         1.00000     0.808120       0.007457   0.010121       0.011897      0.057445
  3                4       0.73797     1.4597     0.35376   0.97566         1.00000     0.816970       0.006877   0.008607       0.015950      0.065491
  4                5       0.82301     1.7707     0.44462   0.97698         1.00000     0.754930       0.007428   0.010042       0.007938      0.045339
...              ...           ...        ...         ...       ...             ...          ...            ...        ...            ...           ...
335                6       0.37522     1.1417     0.81725   0.68511         0.58772     0.125230       0.091860   1.535800       0.114880      0.208610
336                7       0.28064     1.0849     0.75319   0.72152         0.71404     0.136860       0.078996   1.135800       0.141220      0.218300
337                8       0.35344     1.0329     0.78147   0.70737         0.61579     0.135030       0.089763   1.466400       0.097663      0.207030
338                9       0.59988     1.1427     0.71532   0.66101         0.47544     0.157470       0.113370   2.339400       0.050389      0.135850
339               10       0.47195     1.0901     0.85409   0.53598         0.39649     0.078376       0.132270   3.184000       0.082007      0.187820

340 rows × 16 columns
```

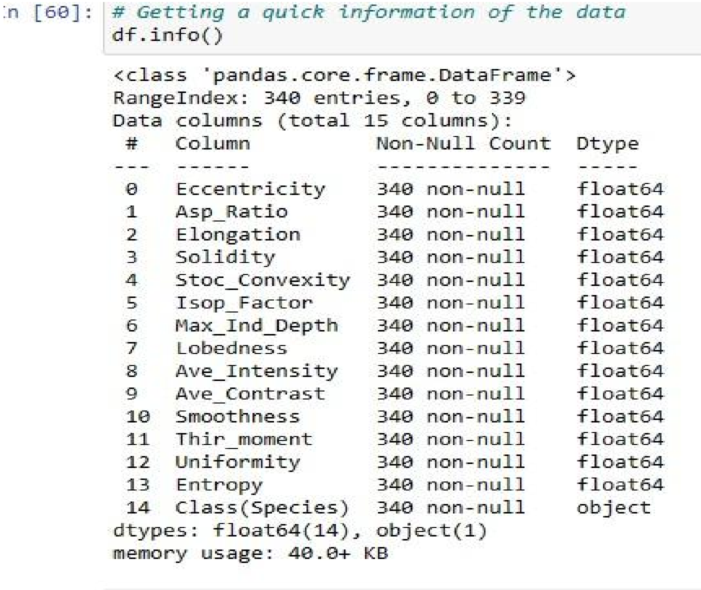
Fig 3. Getting column names



```
In [63]: df.describe()
```

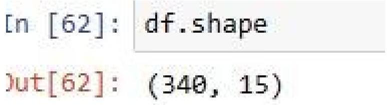
```
Out[63]:
        Eccentricity   Asp_Ratio  Elongation    Solidity  Stoc_Convexity  Isop_Factor  Max_Ind_Depth   Lobedness  Ave_Intensity  Ave_Contrast  Smoothness  Th
count     340.000000  340.000000  340.000000  340.000000      340.000000   340.000000     340.000000  340.000000     340.000000    340.000000  340.000000
mean        0.719854    2.440210    0.513760    0.904158        0.943793     0.531234       0.037345    0.523845       0.051346      0.124535    0.017670
std         0.208311    2.599043    0.195583    0.114639        0.115047     0.217532       0.038575    1.039639       0.035965      0.051860    0.013755
min         0.117080    1.006600    0.107610    0.485490        0.396490     0.078376       0.002837    0.001464       0.005022      0.033415    0.001115
25%         0.550623    1.211300    0.349623    0.890667        0.966230     0.346818       0.009521    0.016500       0.022843      0.083362    0.006901
50%         0.763450    1.570750    0.501855    0.948130        0.992980     0.579160       0.023860    0.103615       0.042087      0.119375    0.014050
75%         0.895097    2.343100    0.633373    0.976897        1.000000     0.700713       0.047834    0.416432       0.073046      0.163795    0.026127
max         0.998710   19.038000    0.948340    0.993880        1.000000     0.858160       0.198980    7.206200       0.190670      0.280810    0.073089
```

Fig 4. Getting statistical information using describe method



```
In [60]: # Getting a quick information of the data
         df.info()

         <class 'pandas.core.frame.DataFrame'>
         RangeIndex: 340 entries, 0 to 339
         Data columns (total 15 columns):
          #   Column           Non-Null Count  Dtype
         ---  ------           --------------  -----
          0   Eccentricity     340 non-null    float64
          1   Asp_Ratio        340 non-null    float64
          2   Elongation       340 non-null    float64
          3   Solidity         340 non-null    float64
          4   Stoc_Convexity   340 non-null    float64
          5   Isop_Factor      340 non-null    float64
          6   Max_Ind_Depth    340 non-null    float64
          7   Lobedness        340 non-null    float64
          8   Ave_Intensity    340 non-null    float64
          9   Ave_Contrast     340 non-null    float64
          10  Smoothness       340 non-null    float64
          11  Thir_moment      340 non-null    float64
          12  Uniformity       340 non-null    float64
          13  Entropy          340 non-null    float64
          14  Class(Species)   340 non-null    object
         dtypes: float64(14), object(1)
         memory usage: 40.0+ KB
```

Fig 5. Getting statistical information using info method

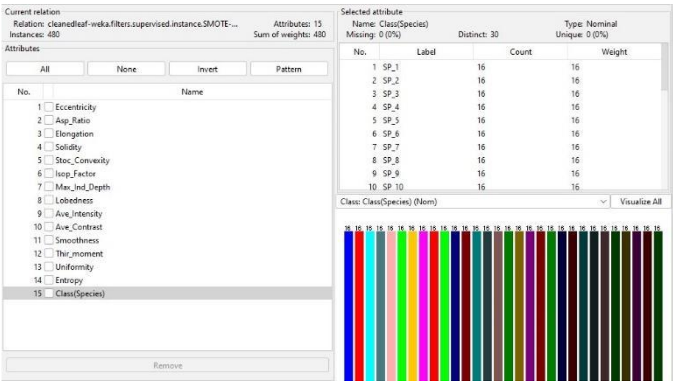


```
In [62]: df.shape

Out[62]: (340, 15)
```

Fig 6. Getting statistical information using shape method

C. Weka (Waikato Environment for Knowledge Analysis)

The University of Waikato in New Zealand created Weka, a Java-based application containing open-source ML algorithms. The Weka tool, utilised in this research work, was beneficial for modelling and visualising data, and also for balancing data through SMOTE technique.



Current relation
Relation: cleanedleaf-weka.filters.supervised.instance.SMOTE-... Attributes: 15
Instances: 480 Sum of weights: 480

Attributes

No.	Name
1	□ Eccentricity
2	□ Asp_Ratio
3	□ Elongation
4	□ Solidity
5	□ Stoc_Convexity
6	□ Isop_Factor
7	□ Max_Ind_Depth
8	□ Lobedness
9	□ Ave_Intensity
10	□ Ave_Contrast
11	□ Smoothness
12	□ Thir_moment
13	□ Uniformity
14	□ Entropy
15	□ Class(Species)

Selected attribute
Name: Class(Species) Type: Nominal
Missing: 0 (0%) Distinct: 30 Unique: 0 (0%)

No.	Label	Count	Weight
1	SP_1	16	16
2	SP_2	16	16
3	SP_3	16	16
4	SP_4	16	16
5	SP_5	16	16
6	SP_6	16	16
7	SP_7	16	16
8	SP_8	16	16
9	SP_9	16	16
10	SP_10	16	16

Class: Class(Species) (Nom)

Fig 7. Balancing the data using SMOTE

D. Data preprocessing (DP)

Data preprocessing is a crucial component of data science that involves various tasks or operations performed on raw data to make it more suitable for use (7). In this study, some preprocessing steps, such as data wrangling, normalisation, and feature extraction, were employed.

1. Data wrangling refers to the process of cleaning and transforming raw data into a format that is usable for downstream purposes such as analysis (8). In this study, Python was used to perform data wrangling. Sixteen attributes were added to the dataset as the data was incomplete. The variable "specimen number" was removed, and the target variable was moved to the last index in the dataset. The values of the variables were also edited to be meaningful nominal values instead of just integers. For instance, the value 2 was changed to SP_2 (species number 2).

2. Normalisation is a data preprocessing technique that involves transforming data values to a common scale. In this study, Min-Max normalisation was used for all explanatory variables using Python. To detect outliers in the dataset, boxplots were plotted and the outliers were replaced with the mean values of the columns to which they belonged.

3. Feature extraction refers to techniques used to reduce the number of explanatory variables by selecting relevant features for the target variable [10]. There are different methods for selecting features, including wrapper methods, filter methods, and

embedded methods. In this study, two filter methods were employed: correlationAttributeEval (Ranker method), which suggested that no explanatory feature should be removed, and cfsSubsetEval (GreedyStepwise method), which suggested that 8 explanatory variables should be retained, namely: Eccentricity, Asp_Ratio, Elongation, Solidity, Isop_Factor, Max_Ind_Depth, Ave_Contrast, and Entropy

.

IV. MODELLING

A. K-Nearest Neighbour (KNN)

KNN is a machine learning technique that falls under the category of supervised learning. This approach involves modelling data based on the proximity of the data points. It can be applied to solve both classification and regression problems. Essentially, it computes the distances between all data points in relation to new or unseen data and selects the data points with the shortest distance to the new data point. [11].

B. Support Vector Machine (SVM)

SVM, also known as support vector networks, was first introduced by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. It is a type of supervised learning algorithm that is commonly used for classification problems. SVM uses hyperplanes and support vectors to categorise the data. In cases where data points are not linearly separable, SVM transforms the data into a higher dimension to enable separation. This technique is useful for solving complex classification problems such as the one tackled in this research work.

C. Decision Tree (DT)

DT is a machine learning model that constructs regression or classification models in the form of a tree structure [12]. It is a supervised learning algorithm that uses discrete values of the target variable for classification. The model consists of internal nodes that are labelled as input features. Each node is represented as a square shape and shows how decisions are made. Typically, a decision tree begins with a single node that branches out into possible outcomes.

D. Random Forest (RF)

RF is a type of supervised learning algorithm that is also known as a random decision tree. It employs bagging to model datasets and makes use of ensemble learning to generate and combine multiple models or classifiers to solve a problem. Unlike a single decision tree, a random forest consists of multiple decision trees

E. Naïve Bayes (NB)

NB is a supervised machine learning technique that is based on Bayes' theorem and assumes independence between the features. Bayes' theorem is a rule that states the probability of an event depends on the conditions related to the event. In notation, we have two events, A and B, which may be conditionally related to each other. P(A|B) represents the conditional probability, which is the probability of A given that B has occurred. P(A∩B) denotes how often A and B are observed to occur together, while P(A) denotes how often A is observed to occur.

Bayes theorem can be represented by equation 1:

$$P\ (A|B) = \frac{P(B|A)P(A)}{P(B)} \qquad (1)$$

F. K Fold Cross-Validation (CV)

CV is a technique used to compare and evaluate machine learning models (13). It involves dividing the dataset into two parts, one for training and one for validation. One type of CV is known as a k-fold CV, which involves automatically repeating the CV process by a parameter called k. The value of k determines the number of iterations that the CV process will go through.

G. Synthetic Minority Oversampling Technique (SMOTE)

SMOTE is a data augmentation technique that addresses the class imbalance problem by generating synthetic samples for the minority class (14). This method involves creating new samples by interpolating between minority class instances. In this study, SMOTE was used to balance the leaf data.

H. Performance Evaluation

1. Confusion Matrix (CM) is a tool used to evaluate the outcomes of a classification problem, also known as a matching matrix in unsupervised learning. It is a table that presents the different prediction outcomes and results of a classification problem, helping to visualise its performance (15). Each row of the matrix represents the number of instances in a target variable class. There are several measures used in the CM, including True Positive (TP), False Positive (FP), False Negative (FN), False Positive (FP), Condition Positive (CP), Condition Negative (CN), Recall, and Precision.

2. Prediction Accuracy (PA) or Accuracy (ACC)

Accuracy refers to the proportion of correct predictions among all observations made for the target variable 'Diagnosis', expressed as a percentage. It is calculated by equation 2.

ACC = TR + TN/ (Total examples). (2)

3. F-score or F- measure

It is a machine learning evaluation metric that measures a model's accuracy. It combines the precision and recall scores of a model (16). It is calculated by equation 3

(2 * precision * Recall) / (Precision + Recall). (3)

V. RESULTS

After performing feature extraction using Weka with cfsSubsetEval (GreedyStepwise method) and correlationAttributeEval (Ranker method), an additional dataset was obtained. This enabled the evaluation and comparison of machine learning algorithms on two different types of leaf datasets.

Algorithms	ACC	AUROC	Recall	Precision	F-score
KNN	0.821	0.907	0.821	0.824	0.819
SVM	0.727	0.978	0.727	0.740	0.721
DT	0.729	0.892	0.729	0.735	0.727
RF	0.846	0.991	0.846	0.850	0.846
NB	0.794	0.981	0.794	0.811	0.797

Fig 8. Performance results using full features

Algorithms	ACC	AUROC	Recall	Precision	F-score
KNN	0.825	0.909	0.825	0.828	0.824
SVM	0.663	0.975	0.663	0.656	0.651
DT	0.735	0.889	0.735	0.740	0.734
RF	0.842	0.987	0.842	0.845	0.841
NB	0.779	0.988	0.779	0.788	0.779

Fig 9. Performance results using selected features.

VI. DISCUSSION

Five ML algorithms were on two kinds of leaf dataset namely: KNN, SVM, DT, RF and NB. Performance comparison was studied between ML techniques and between two datasets.

In Fig. 8, RF happened to perform better than the rest having the highest values of performance evaluation measures. 0.846 (ACC), 0.991 (AUROC) and 0.846 (F-score), followed by KNN, NB, DT and SVM. It is also revealed that KNN performed better on a dataset with fewer variables in Fig. 9. While the rest except RF did fairly better on the dataset with complete variables in Fig 8. RF performance was higher on a dataset with complete variables.

VII. CONCLUSION

The study utilised five machine learning (ML) algorithms based on data science principles to tackle a classification problem using a leaf dataset. The methodology involved data collection and preprocessing using Python, as well as model building and performance evaluation utilising Weka.

The findings suggest that among the algorithms tested, Random Forest (RF) performed the best in solving multi-class classification problems using the attribute selection dataset recommended by the correlationAttributeEval (Ranker method).

VIII. RECOMMENDATION

It is recommended to conduct a study that uses alternative ML algorithms to challenge the superiority of RF observed in this study. Additionally, conducting a study that includes all 16 variables in the leaf dataset could be an interesting area for further investigation.

IX. REFERENCE

[1] https://www.geeksforgeeks.org/introduction-to-data-science/

[2] S. Wang et al 2022, A machine learning software tool for multiclass classification. [online]. https://www.sciencedirect.com/science/article/pii/S2665963822000847

[3] F. Lazzeri and S. Penchikala (2019), The Data Science Mindset: Six Principles to Build Healthy Data-Driven Organisations. [online]. The Data Science Mindset: Six Principles to Build Healthy Data-Driven Organizations (infoq.com)

[4] L. Shi et al 2022, Multi-Class Classification of Agricultural Data Based on Random Forest and Feature Selection [online]. Multi-Class Classification of Agricultural Data Based on Random Forest and Feature Selection: Computer Science & IT Journal Article | IGI Global (igi-global.com)

[5] https://en.wikipedia.org/wiki/Project_Jupyter

[6] What Is Python Used For? A Beginner's Guide | Coursera

[7] Data Preprocessing: Definition, Key Steps and Concepts (techtarget.com)

[8] Data wrangling - Wikipedia

[9] Normalization (c3iot.ai)

[10] How Does Feature Selection Benefit Machine Learning Tasks? (h2o.ai)

[11] Shivam Sharma 2021, K-Nearest Neighbour: The Distance-Based Machine Learning Algorithm. [online]. https://www.analyticsvidhya.com/blog/2021/05/knn-the-distance-based-machine-learning-algorithm/#:~:text=The%20abbreviation%20KNN%20stands%20for,by%20the%20symbol%20'K'.

[12] https://en.wikipedia.org/wiki/Decision_tree_learning

[13]https://link.springer.com/referenceworkentry/10.1007/978-0-387-39940-9_565#:~:text=Definition,used%20to%20validate%20the%20model.

[14]https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

[15] What is a Confusion Matrix in Machine Learning? (simplilearn.com)

[16]https://www.v7labs.com/blog/f1-score-guide#:~:text=F1%20score%20is%20a%20machine%20learning%20evaluation%20metric%20that%20measures,prediction%20across%20the%20entire%20dataset