CLASSIFYING MOLECULES INTO FAMILIES

ABSTRACT

Machine Learning techniques were used to classify compounds into three different families: Alcohol, Amine, and Hydrocarbon. Supervised and Unsupervised machine learning techniques were employed to investigate a data set. Principal component analysis and k means clustering was employed to assess the dataset. Three different supervised learning models were employed to classify molecules, using a training and testing split of the data. Ultimately using a decision tree classifier was the most effective method, as quantified by performance metrics including f1 scores and confusion matrices.

INTRODUCTION

Machine Learning involves the use of multiple statistical algorithms that can learn from and find patterns in data, with the aim to perform tasks without explicit instructions.[1] The technique finds itself applied to many areas of work, including in robotics,[2] plant breeding of soybean, [3] drug discovery and development[4], and in the case of this work, for classification of molecules into families.

Multiple different approaches may be used to complete such machine learning tasks, and these approaches fall under the umbrella of supervised or unsupervised learning. Supervised learning includes classification and regression algorithms, where pre-labelled training data is used by the learning algorithm through iterative optimisation to generate a function which may predict the 'output', i.e., in the context of classification, the class label.[5] This function may be then used on test data which has not been pre-classified to predict the output labels.[5] This work utilised multiple supervised learning algorithms to classify these molecules. The sci-kit learn[6] decision tree classifier is an algorithm used which makes predictions by learning simple decision rules inferred from the data features. The algorithm uses the Gini index, see equation, where p_i denotes the probability of an element being classified to a distinct class label. The algorithm aims to minimise the index at each branch in the tree so that each rule used maximises the data classified to each label. The index varies between 0 and 1.

$$Gini = 1 - \sum p_i^2 \tag{1}$$

Decision trees are done recursively and are highly sensitive to training data. It is possible to further advance this model by using a 'Random Forest', which averages the decisions of multiple trees into one tree, reducing overfitting issues, and decreasing the variance of the model.[7]

The second supervised learning technique used was binary logistic regression, which models based on the sigmoid function (Equation 3) to output a probability value indicating the likelihood of a binary outcome variable, i.e., 0 or 1 as the integer representation of two outcomes, from the input features.

$$\sigma(z) = \frac{1}{1 - e^{-z}} \tag{2}$$

Hence, this model is not inherently a classifier. By choosing a probability cutoff value, and hence classifying inputs by their probability value in relation to the cutoff, it is possible to classify data.

In the instance of this work, the One vs All method was used, where the multi-class problem is broken by splitting the problem into multiple binary classifier models: k models for k classes.

The final supervised learning technique employed was the scikit-learn k-nearest neighbours' classification. The algorithm does not construct a general internal model, but stores instances of the training data, calculating the distance between the nearest neighbouring samples. The most common distance measure is the previously mentioned Euclidean distance, equation 3.

$$d(x, y) = \sqrt{(y - x)^2} \tag{3}$$

The classification is computed from a majority vote of the nearest neighbours of each point; where the query point is assigned the class which has the highest frequency of samples within the nearest neighbours' of that point.

In contrast, unsupervised learning approaches find patterns from unlabelled and unclassified data. Central applications of machine learning include clustering and dimensionality reduction. This work applied k-means clustering to partition the data. The scikit-learn k-means algorithm separates data into n pre-specified samples groups of equal variances, by minimising the within-cluster sum-of-squares criterion, see equation.

$$\sum_{i=0}^{n} \min(\|x_i - \mu_j\|)^2 \tag{4}$$

This formula is a measure of how coherent the calculated clusters are; however, it is not normalised; just lower values are better. In high-dimensional spaces, Euclidean distances tend to become inflated, and hence in this work, a dimensionality reduction algorithm was applied prior to the k-means clustering, to reduce the inflation of distances. Principal component analysis (PCA) was used for this process. The technique uses principal components (PCs) which are linear combinations of the original input features combined into one variable, the PC. [8] Each PC aims to capture the largest possible portion of the variance in the data, where the first PC captures the most variance and the last PC captures the least. Each PC captures contributions from each descriptor input, quantified by descriptor loadings, see Appendix. Although this technique is not statistically robust,[9] it was useful to discern patterns in the dataset which may not be immediately obvious, and as pre-processor for the k-means algorithm.

Evaluation of the machine learning techniques was of importance. Multiple metrics were used, including accuracy scores, f-1 scores, and confusion matrices. The accuracy score presents the proportion of correctly classified samples out of the total samples in the dataset, i.e., how often the model's predictions match the true labels. This metric may not provide a complete picture of the model's performance, given that the cost of misclassification may vary between classes and the overall class distribution may be imbalanced. Consequently, the f-1 score is also commonly used to evaluate classification models. This metric is the harmonic mean of the *precision* and *recall* metrics which quantitate the proportion of true positive predictions out of all positive predictions, and the proportion of true positives out of all true positive instances in the data set.

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

The score ranges from 0 to 1, where 1 indicates a perfect precision and recall, whilst 0 indicates either precision or recall is 0. Finally, the confusion matrix is a valuable metric for classification evaluation, by presenting a summary of the model's predictions compared to the ground truth across different classification labels to define the performance of the classification algorithm.

		Predicted	
		Negative	**Positive**
Actual	**Negative**	TN	FP
	Positive	FP	TP

Each cell in the matrix counts from the predicted and actual values. A 'TN' value indicates a True Negative, i.e, the number of negative instances classified correctly. A 'FN' value indicates a False Negative, for the number of negative instances which were classified incorrectly by the model. The same formula stands for the TP and FP terms, for True Positive and False Positive.

In this work, these machine learning approaches were applied to a dataset of variables related to the properties of 1748 different molecules. The overarching goal was to assess whether this given dataset would be suitable to predict labels from the given parameters for a larger database. The data set was explored with exploratory statistics and unsupervised learning approaches then classified using supervised learning models.

METHODOLOGY

The 'molecules' dataset was analysed using the python programming language, using NumPy, Pandas, sci-kit learn and statsmodels modules for statistical analyses and machine learning approaches. The Matplotlib module was used for data visualisation.

RESULTS AND DISCUSSION

The dataset provided consisted of 11 features and 1748 compounds, and with no missing data, the entire dataset was used for analysis and prediction. The aim of the project was to use the input features to classify each compound into three categories: Alcohol, Amine, and Hydrocarbon. Of the 13 features the dataset provided, three were continuous variables; and the remaining ten were discrete, see Table 1.

Table 1; Features provided in the 'molecules' dataset.

Feature name	Description	Type
BoilingPoint	Measured Boiling Point	Continuous
mw	Molecular Weight	Continuous
polararea	Area of the polar area	Continuous
heavycnt	Number of non-hydrogen atoms	Discrete

hbondacc	Number of hydrogen-bond acceptors	Discrete
C number	Number of C atoms	Discrete
N number	Number of N atoms	Discrete
O number	Number of O atoms	Discrete
Side chain number	Number of side chains	Discrete
Double bond number	Number of double bonds	Discrete
Triple bond number	Number of triple bonds	Discrete

Correlation analysis using the R^2 metric found that *mw*, *heavycnt*, *C number*, and *BoilingPoint*, were highly correlated to each other, with R^2 scores ranging from 0.83 to 0.99, see Figure 1. An R^2 of 1.00 suggests a perfect linear correlation, 0 suggests no linear correlation, and -1 suggests a perfect negative linear correlation. Other variables had a poor correlation to those variables, ranging between 0.26 to – 0.12. See Figure 1. The second area of high correlation consisted of *O number*, *hbondacc*, and *polararea*, with R^2 scores ranging from 0.87 to 0.95. Similarly, any other variable from the data set had a weak relationship with these three variables, ranging from 0.33 to -0.23. Unsurprisingly, the N number was loosely correlated with *hbondacc* and *polararea*, with R^2 of 0.33 and 0.25 respectively. Whilst the oxygen atom is more electronegative than that of nitrogen, with a Pauling electronegativity score of 3.5 versus 3.0, the N atom is still more electronegative than that of carbon or hydrogen, atoms which takes values of 2.5 and 2.1 respectively. The variables *Side chain number, Double bond number,* and *Triple bond number* had no correlations to any other features, all with R^2 scores ranging from -0.19 to 0.07.

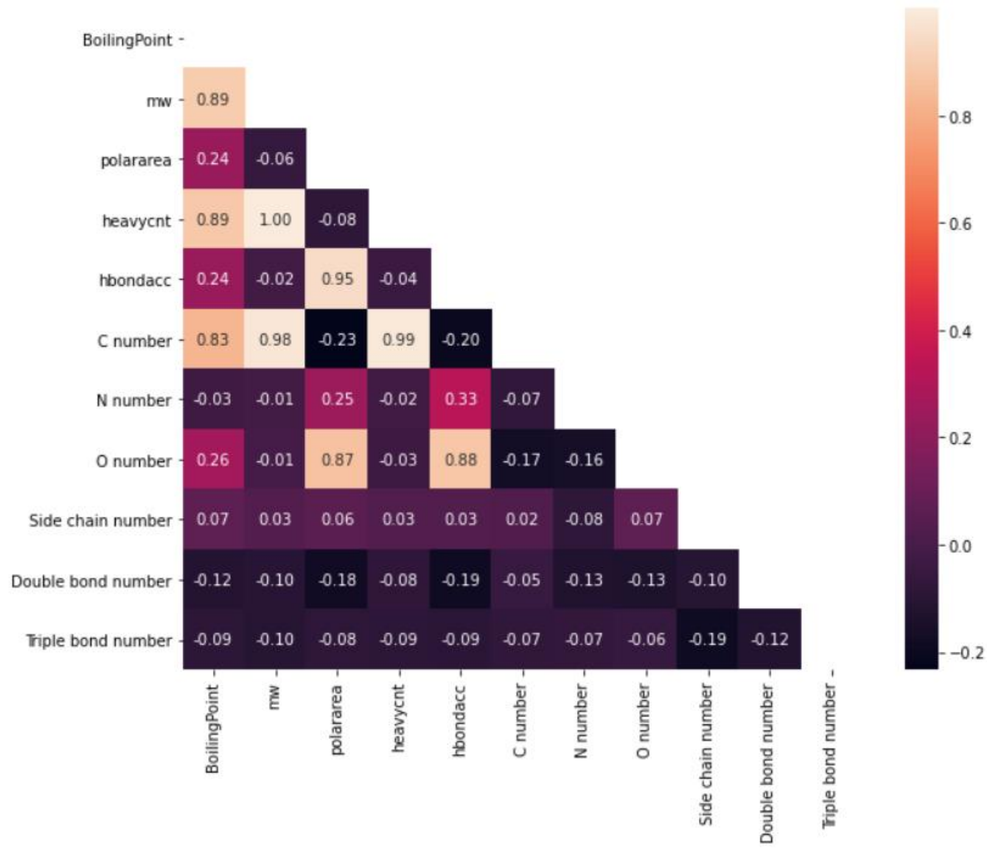


Figure 1; Correlation Matrix for each of the features provided in the 'molecules' dataset.

To further investigate the relationship between each feature and the classification of each compound, exploratory statistics were used to compare the data trends for each class label. For some variables, there was only a subtle difference between classes in terms of their distribution, averages, and outlier values: e.g., *mw*. The important note for this feature was that although each class had a similar distribution, the average value tended to be greater for samples representing the Hydrocarbon class. For the four features *polararea*, *hbondacc*, *O number,* and *N number*, the difference in classes was more obvious, where for each feature the hydrocarbon class held a value of 0, and for *O number*, the amine class also held a value of 0, see Table 2.

Table 2; Mean values for each label classification for polararea, hbondacc, N number, O number.

Label	polararea	hbondacc	N number	O number
Hydrocarbon	0.00	0.00	0.00	0.00
Alcohol	27.93	1.49	0.00	1.49
Amine	18.48	1.32	1.32	0.00

For *BoilingPoint,* see Figure 2, all three classes had similar distributions. Despite the disparity in the size of each class, all three distributions are similar, with a slightly positive skew.

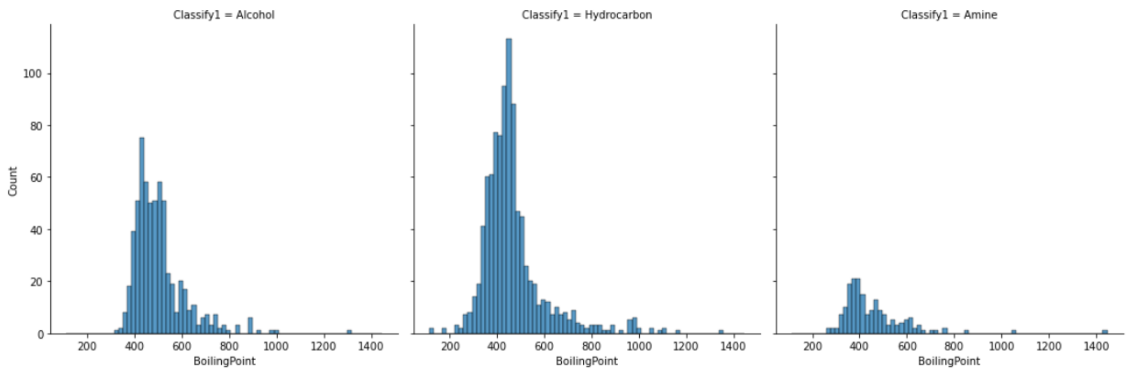


Figure 2; Histograms depicting the distribution of data for the BoilingPoint for each class label.

Alcohols had a higher mean value, at 500.4, whilst amine and alcohols were similar with respective means of 452.1 and 465.6. The alcohol class had a standard deviation of 107.2, whilst amines and hydrocarbons had respective standard deviations of 135.1 and 134.3, see Appendix. These values set alcohols apart from the other two classes, useful given their similarity to amines noticeable for other features, such as *polararea*.

Alcohols and Amines were most likely to have one side chain, whilst hydrocarbons were most likely to have zero or two, where in this case using the mean value for each class does not provide much information, especially given the categorical nature of this feature, see Figure 3.

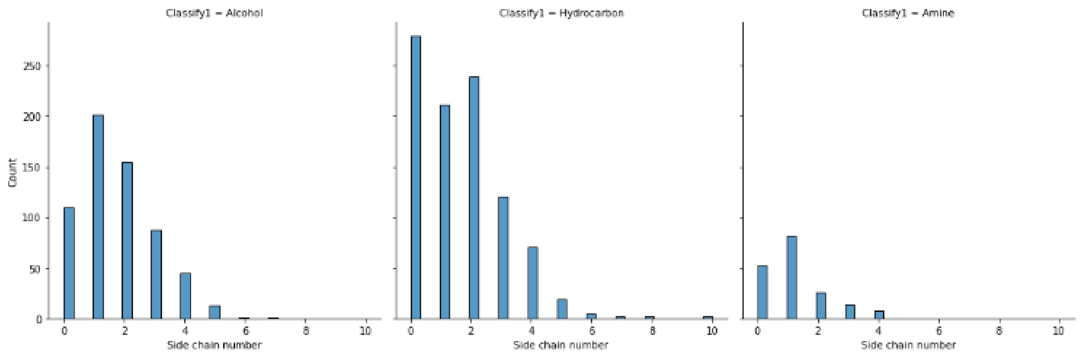


Figure 3; Histograms depicting the distribution of data for the Side chain number for each class label in the data set.

Looking at the data trends through the histograms in Fig. 3, again all three classes have a slight positive skew. Additionally, the features Double bond number, and Triple bond number were not useful for classification of molecules. These categorical variables took integer values between 1-10, 1-13, and 1-3, with similar distribution of data between classes. Most molecules did not have a double or triple bond, and for each class, the distribution of data was very similar.

Principal Component Analysis followed by K-means clustering was applied to the entire dataset, using all features. The K-means clustering algorithm appeared to produce clusters of each class label, however given that the algorithm was applied to the principal components rather than the true dataset, this is not a method that can be used for robust classification. However, given that the first three principal components capture 73% of the explained cumulative variance, the descriptor loadings may be probed to investigate underlying trends in the data. For PC1, the most important feature was *mw*, for PC2, it was *hbondacc*, and for PC3 it was *N number;* see Appendix.

The distribution of classes between these compounds was poor, suggesting the need for resampling algorithms to be considered. The original data held only 10.47% of compounds with the Amine class, whilst 54.41% of compounds had the Hydrocarbon class, leaving 35.13% to the Alcohols. Imbalanced data may be difficult to classify and hence SMOTE, Synthetic Minority Oversampling Technique, was employed, and used for the training data for each supervised learning model, which was then tested on the original test data to avoid data leakage.[10] See Fig. 4.

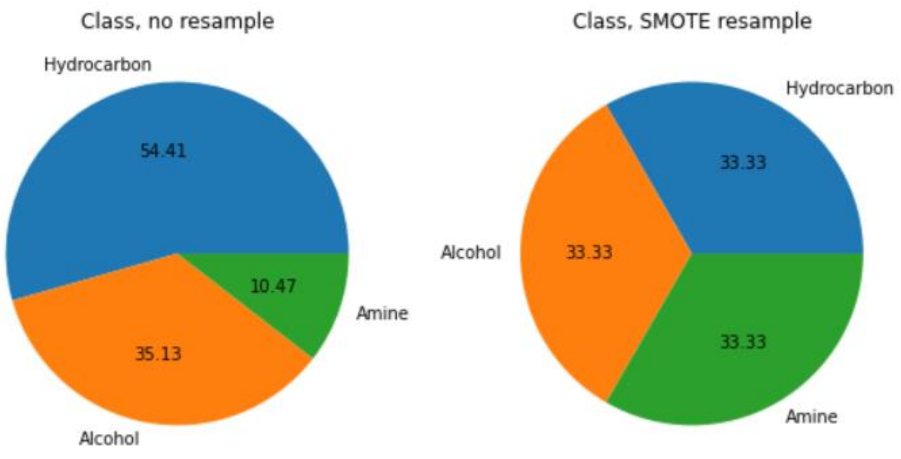


Figure 4; True data versus SMOTE resampled data

Both the original data and SMOTE data were used as training data for each of the three models investigate to provide a comparison, and to enable critical analysis of the use of SMOTE resampling.

When the models were trained using every feature provided by the dataset, for the knn model, the SMOTE data performed slightly better, see table 3. Given that accuracy scores may be misleading to model performance, and the f1 score provides a quantitative balanced representation of precision and recall, f1 score was the most heavily used performance metric. See the Appendix for confusion matrix.

Table 3; KNN performance metrics for the first model investigated, employing every feature from the dataset.

KNN Class, model 1	**F1 score – true data**	**F1 score – SMOTE data**
Alcohol	0.93	0.91
Amine	0.40	0.55
Hydrocarbon	0.97	0.97

Evidently, even when the model is trained on every feature available, the balance between precision and recall as quantified by the F1 score was poor for the Amine class. Use of SMOTE resampling helped somewhat to alleviate this, but not a lot, increasing the score 1.375x from 0.4 to 0.55. The decision tree, however, classified each molecule perfectly, regardless of whether the data had been SMOTE resampled or not. Both trees were simple, both with 1 root node, 1 decision node, and 3 leaf nodes. For the logistic regression model, when all features were used to train the model, the model failed to converge due to perfect separation of the outcome variables, for both the true data and the SMOTE data.

Given that the categorical variables *heavycnt* and *C number* are highly correlated to the continuous features *mw* and *BoilingPoint,* these categorical input variables were removed for use in the classification models, partly to avoid overfitting, but also to avoid categorical predictions. Following the same reasoning, *hbondacc* and *O number* were also not used as input features given their high correlation to the continuous *polararea*. Additionally, given that the features *Double bond number* and *Triple bond number* have little to no relationship to the classification labels, nor a particular impact on the variance or trends in the overall data set (no descriptor loadings for PC1 or PC2, which cover 62% of the dataset's cumulative explained variance) these features were also removed. Unsurprisingly, these removals from the feature selection had little impact on the model performance. For the knn classifier, f1 scores remained the same, with a decrease of 0.01 in f1 score for Alcohol and Amine prediction using the SMOTE data.

Table 4; Performance metrics for the KNN model 2, employing the features BoilingPoint, mw, polararea, N number, Side chain number.

KNN Class, model 2	**F1 score – true data**	**F1 score – SMOTE data**
Alcohol	0.93	0.90
Amine	0.40	0.54
Hydrocarbon	0.97	0.97

The decision tree classifier remained a perfect classifier, again producing a simple tree with 1 decision node, 1 root node and 3 leaf nodes for both the true data and the SMOTE data.

Again, the one vs all binary logistic regressions failed to converge due to complete separation. This error is a result of one or more predictor variables perfectly predicting the output variables, which leads to infinite parameter estimates. Ultimately the logistic regression model continued to have issues until only one predictor variable was used – but at that point the model was inaccurate and provided low f1 scores, for example, if *BoilingPoint* is used as the predictor variable, although all three regression models converge, the f1 scores of the model are insufficient, see table 5. When the model is fitted to the true data especially, the scores are lower.

Table 5; Performance metrics for One vs All binary linear regression models trained on the BoilingPoint input variable.

BLR Class	F1 score – true data	F1 score – SMOTE data
Alcohol	0.015	0.500
Amine	0.000	0.192
Hydrocarbon	0.702	0.660

In comparison to the performance of both the k-nearest neighbours' model and the decision tree classifier, the binary logistic regression model does not keep up.

The third model investigated used only two input variables; *N number* and *polararea*, and out of the k-nearest neighbours' models, performed the best, see Table 6. This time, the model was able to predict samples of the amine class much more effectively, and it is evident that, like the binary linear regression, a simple model in this case is the most effective.

Table 6; Performance metrics for the KNN model 3, using only N number and polararea as input features.

KNN Class, model 3	F1 score – true data	F1 score – SMOTE data
Alcohol	0.99	0.99
Amine	0.97	0.97
Hydrocarbon	1.00	1.00

The decision tree again performed excellently, with perfect f1 scores of 1.0, and the minimal 1 root node, 1 decision node, and 3 leaf nodes.

Overall, the decision tree consistently performed excellently. Should the tree have become convoluted, with many decision nodes and poor class separations, it may have been prudent to investigate using a random forest model or using the k nearest neighbours' model.

CONCLUSION

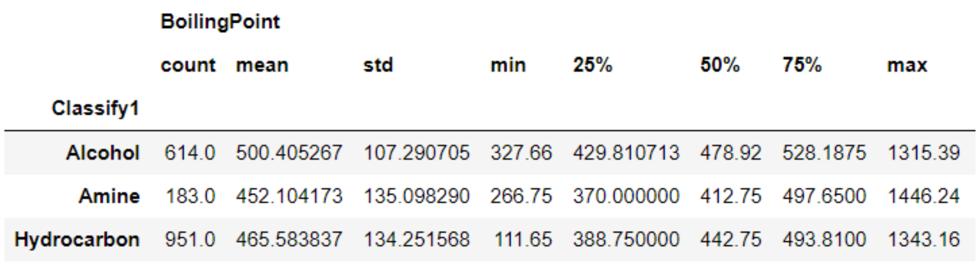
The binary linear regression model was the least effective at classification, followed by the k-nearest neighbours' model, which although was ineffective when trained on the whole dataset, when only two variables were used to train the model – *N number* and *polararea* the model worked well, with no difference in f1 score between the true data and the SMOTE data. Ultimately, the decision tree was the most effective classifier in this context. SMOTE resampling improved predictions only slightly, and hence the overall dataset would be suitable to predict labels for a larger database, without resampling if the appropriate classification method is chosen, such as decision tree classification.

REFERENCES

1. J. G. Carbonell, R. S. Michalski and T. M. Mitchell, in *Machine Learning*, eds. R. S. Michalski, J. G. Carbonell and T. M. Mitchell, Morgan Kaufmann, San Francisco (CA), 1983, pp. 3-23.
2. J. Hu, H. Niu, J. Carrasco, B. Lennox and F. Arvin, *IEEE Transactions on Vehicular Technology*, 2020, **69**, 14413-14423.
3. M. Yoosefzadeh-Najafabadi, H. J. Earl, D. Tulpan, J. Sulik and M. Eskandari, *Front Plant Sci*, 2020, **11**, 624273.
4. J. Vamathevan, D. Clark, P. Czodrowski, I. Dunham, E. Ferran, G. Lee, B. Li, A. Madabhushi, P. Shah, M. Spitzer and S. Zhao, *Nature Reviews Drug Discovery*, 2019, **18**, 463-477.
5. P. Cunningham, M. Cord and S. J. Delany, in *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval*, eds. M. Cord and P. Cunningham, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 21-49.
6. F. Pedregosa , G. Varoquaux , A. Gramfort , V. Michel , B. Thirion , O. Grisel , M. Blondel , P. Prettenhofer , R. Weiss , V. Dubourg , J. Vanderplas , A. Passos , D. Cournapeau , M. Brucher , M. Perrot and E. Duchesnay *Journal of Machine Learning Research*, 2011, **12**, 2825-2830.
7. scikitLearn, 1.11.2. Random forests and other randomized tree ensembles¶, https://scikit-learn.org/stable/modules/ensemble.html#forest).
8. N. Fey, A. Koumi, A. V. Malkov, J. D. Moseley, B. N. Nguyen, S. N. G. Tyler and C. E. Willans, *Dalton Transactions*, 2020, **49**, 8169-8178.
9. A. Fayomi, Y. Pantazis, M. Tsagris and A. T. A. Wood, *Statistics and Computing*, 2023, **34**, 26.
10. A. N. Tarekegn, M. Giacobini and K. Michalak, *Pattern Recognition*, 2021, **118**, 107965.

APPENDIX

DIST OF BP ACROSS CLASSES



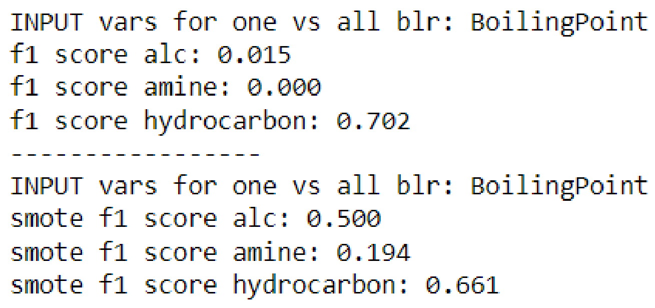
	BoilingPoint							
	count	mean	std	min	25%	50%	75%	max
Classify1								
Alcohol	614.0	500.405267	107.290705	327.66	429.810713	478.92	528.1875	1315.39
Amine	183.0	452.104173	135.098290	266.75	370.000000	412.75	497.6500	1446.24
Hydrocarbon	951.0	465.583837	134.251568	111.65	388.750000	442.75	493.8100	1343.16

PCA LOADINGS

Table 7; Feature Loadings for the first three PCs where underlined loadings show the most important feature for that PC. Loadings below 0.3 were not included.

Feature	PC1	PC2	PC3
Cumulative variance	**34.7**	**62.1**	**73.0**
BoilingPoint	0.471	-	-
mw	0.509	-	-
Polararea	-	0.562	-
Heavycnt	0.509	-	-
Hbondacc	-	0.567	-
C number	0.500	-	-
N number	-	-	0.539
O number	-	0.526	-
Side chain number	-	-	0.544
Double bond number	-	-	0.339
Triple bond number	-	-	0.500

BLR MODEL PERFORMANCE



```
INPUT vars for one vs all blr: BoilingPoint
f1 score alc: 0.015
f1 score amine: 0.000
f1 score hydrocarbon: 0.702
-----------------
INPUT vars for one vs all blr: BoilingPoint
smote f1 score alc: 0.500
smote f1 score amine: 0.194
smote f1 score hydrocarbon: 0.661
```

PERFORMANCE ASSESSMENT FOR THE DECISION TREE MODEL ON TRUE DATA, USING *N number* AND *polararea* AS INPUT FEATURES

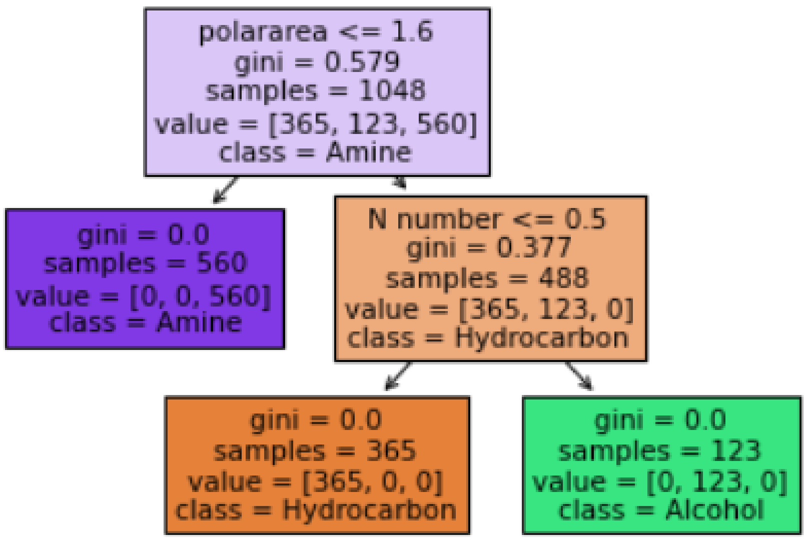
```
Classification report:
              precision    recall  f1-score   support

     Alcohol       1.00      1.00      1.00       249
       Amine       1.00      1.00      1.00        60
 Hydrocarbon       1.00      1.00      1.00       391

    accuracy                           1.00       700
   macro avg       1.00      1.00      1.00       700
weighted avg       1.00      1.00      1.00       700

-------------------------
Accuracy Score: 1.0
-------------------------
Confusion Matrix:

array([[249,   0,   0],
       [  0,  60,   0],
       [  0,   0, 391]], dtype=int64)
```


polararea <= 1.6
gini = 0.579
samples = 1048
value = [365, 123, 560]
class = Amine
gini = 0.0
samples = 560
value = [0, 0, 560]
class = Amine
N number <= 0.5
gini = 0.377
samples = 488
value = [365, 123, 0]
class = Hydrocarbon
gini = 0.0
samples = 365
value = [365, 0, 0]
class = Hydrocarbon
gini = 0.0
samples = 123
value = [0, 123, 0]
class = Alcohol