A Comparison of Wine Cultivar Classification with Support Vector Machines and K-Nearest Neighbours

Charles Chudley
School of Computer Science
University of Nottingham
Nottingham, UK
14301432

Joel Dolman
School of Computer Science
University of Nottingham
Nottingham, UK
20356919

Abstract—**The UCI Wine dataset has seen widespread usage in the development of new machine learning techniques. However, it has only seen limited usage in the analysis of it's original target, wine cultivar (grape variety) classification. This paper trains two machine learning models, a support vector machine and a k-nearest neighbor model as well as conducts a preliminary data exploration to gain insight into what defines the wine cultivars in this dataset. We find that the kNN algorithm substantially outperforms the SVM in both accuracy and training time, and that the proline levels and hue of the wine best explain a wine's cultivar.**

Keywords—Support Vector Machine, k-Nearest Neighbors, Cultivar, Data Modelling and Analysis

I. Introduction

Since the invention of Wine 8000 years ago in what is modern day Georgia, humans have been manipulating wine's ingredients and fine tuning their manufacturing process to adjust taste, mouthfeel, and alcohol content. Until recently, this development has been supported by anecdotal studies, where farmers favour breeding grapes that produce the 'best' wine and where winemakers adjust their processes to optimise the desired output characteristics. Increasingly, however, this process is becoming more scientific, with molecule level analysis being used to link characteristics with the chemicals and compounds that cause them.

It is in this wider context that this paper presents itself, by utilising modern classification techniques like k-Nearest Neighbours (kNN) and support vector machines (SVM), this paper will attempt to the classify 3 distinct Cultivars (Grape varieties) using a set of 13 separate features provided by the UCI Machine Learning Wine Dataset. The aim of this paper is to gain insight into: the features that best define the classification boundaries, the optimal training hyperparameters for each model and which model overall best classifies the data points. To this end, there will firstly be an analysis of the data's summary statistics, such as arithmetic mean, standard deviation and correlation from which initial hypotheses can be drawn. Following this, several classification models of the SVM and kNN varieties will be trained on the data set. The models will be trained and analysed with different methods. The SVM approach will focus on various dimensionalities of data including using the full dataset, principal component analysis and a qualitative approach. Whereas kNN will focus on the proficiencies of different *slices* of data, i.e what is the smallest percentage of data we can use before accuracy begins to deteriorate? The SVM approach has been chosen due to its demonstrated proficiency in multi-class classification in high dimensional feature space. k-Nearest Neighbours has been chosen because of its accuracy, and effective and unbiased nature.

Finally, there will be a discussion pertaining to the proficiency of each model as well as any insights garnered concerning the relationship between different cultivars and their constituent features. The discussion will also attempt to put the results into the wider context of the literature, highlighting possible contradictions in addition to presenting possible future works.

II. Literature Review

A. *Dataset*

The Wine Dataset [1] used in this paper was one of the first high dimensional, complete, and large datasets collected with the express intention of developing classification models, with 13 features, 178 observations and three possible cultivar classes. It's status as an industry standard has attracted researchers who are focused on developing new models rather than investigating the properties of wine. This bias in the literature presents an opportunity for novel work by utilizing this dataset for wine analysis and not the development of new ML techniques.

One of the first papers written partially using the wine dataset was that of Aeberhard, Coomans and de Vel [2] which tested early implementations of linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA). In this study, they demonstrate that due to LDA's assumption that all features have identical covariances, QDA tends to perform better on real-world datasets. Even in 1994, they achieved a 99.4% and 98.9% classification accuracy with QDA and LDA respectively. This level of accuracy demonstrates one of the key reasons the wine dataset has assumed such a prominent place in the development of machine learning (ML) techniques; it is high dimensional, has a good number of datapoints and above all it is easily separable, providing researchers with strong accuracies even with their new models.

B. *Support Vector Machines*

A support vector machine defines the boundaries between classes in feature space through a series of 'support vectors', which can be thought of as datapoints on the outer limits of a cluster. As clusters are not always easily separable, SVMs employ kernels, which are functions that can project input

2022 IEEE

features into a higher dimensional feature space. For example, if a dataset is made up of variable x, then by using a polynomial kernel the dataset would become x, x^2. Projecting features into a new space can enable the SVM to find hyperplanes that separate features in this new space.

They have been proven to be highly accurate when applied to the correct problem, even achieving a 99.44% accuracy with 537 support vectors on the same wine classification dataset this paper is using [4]. This paper, like those cited above, was concerned with developing new SVM technologies, namely by applying linear and quadratic programming techniques to a multiclass SVM, removing the need for a one versus all classification and thus removing the need for k-1 number of trained SVMs where k is the number of possible classifications.

This paper has partially chosen to use SVMs in the analysis due to the high performance demonstrated in the above papers, however SVMs also provide opportunity for interpretation. If the separating hyperplane is defined by 1, 2 or 3 dimensions, it can be represented graphically, providing a rare (in ML) visualisation. Their resultant metrics also provide strong opportunity for inference, for example a high number of support vectors required to classify the clusters indicates heavily mixed classification clusters.

C. *K-Nearest Neighbours*

The dataset has also been used in the development of kNN methods. A 2007 paper made use of it to develop faster variations of kNN. They did this by (i) adding a synthesis patterns system and (ii) by adding a “greedy method” in various combinations. They found that their upgraded kNN was able to classify the wine data with accuracy as high as 98.7% using the “greedy classifier”, denoted by GDC-kNNC. It did this using a divide and conquer strategy to find kNNs of the test pattern and a space and time requirement that simple kNN methods don’t use [11]. Another paper used the data set to propose an extension to kNN on feature projections (kNNFP) [12]. They added a weight learning element to the algorithm based on the assumption that it would be proportional to the accuracy. For most values of k used, the weighted variation improved the algorithms classification accuracy of the wine dataset by as much as 3%. The fact that it is a well-known dataset that other kNN papers have made use of shows that it’s well suited to this dataset. Both papers cited here utilise the dataset for purposes of ML method development, and to the knowledge of this paper, there have been no research papers citing this dataset with the express purpose of providing insight into wine chemistry.

In a K-Nearest Neighbour algorithm, data is classified based on a similarity measure. It does this by assessing the similarity between points, i.e through Euclidean Distance. When the algorithm has determined which training points (the nearest neighbours) are the most similar to the test point and their classification, it determines what classification the test point comes under based on the classification of k nearest neighbours (where k is an integer). In other words, it looks at the classification of the k most similar points to the test point. The test point is then added to the classification that is most numerous among the k neighbours. In order to determine whether the algorithm has predicted correctly, the data is initially “partitioned” into a training set and a test set. In this paper, the proportion of the data which has been selected for training will be denoted by P. Both sets are then passed to the algorithm, which checks its answers (or “guesses”) based on the test set. This allows an accuracy value to be obtained.

k-Nearest Neighbours has been chosen for this paper because of its accuracy, and effective and unbiased nature (its simplicity lends to its transparency). It has also been chosen because there are many examples in research that show that provides results which have a high accuracy of results. It also has been shown to be hugely customisable which is a feature that will be useful for the methodology.

III. Methodology

Before any methodology can be attempted, the data must first undergo pre-processing. Inspection of the data shows that it is well formatted, with no missing datapoints but also without correct labelling. Therefore, when the dataset is first read into R, the data must be formatted into a dataframe with the correct labels. Across the 13 attributes; there is huge variation in the mean value and standard deviations. As such, it is also necessary to standardise the data so that all features are standardized, this paper uses z-normalization for this purpose, which will enable comparison between features by putting them firmly on the same scale, for example, feature 13, “Proline” has an unstandardised mean of 746.9 and a standard deviation (sd) of 314.9, which would cause the significance of feature 1, Alcohol to be lost in comparison as it only has a mean of 13 and sd of 0.811.

After this, a ggpairs scatter plot was created. This graphing function shows the scatter plot of all datapoints as defined by their pairwise interactions with other independent variables (This can be seen in figure 15 in the appendix). This initial visualisation was useful to make preliminary judgements on which features present easily separable clusters. Due to the qualitative nature of this analysis, it was here where the two methodologies diverge, with each researcher choosing their own feature subsets to analyse.

A. *SVM*

In our SVM classification approach, three different sets of features will be used for training, the full set of 13 features, a principal component analysis (PCA) selected features and a correlation/qualitatively selected feature set. These have been chosen for two reasons; firstly, it will allow for comparison between the use of a full feature set versus a reduced one, i.e is the increased training time warranted for a small increment in accuracy? Secondly, when dimensionality reduction is performed with a feature *engineering* method such as PCA, rather than a feature *selection* process, the context of the original features is lost (i.e Color Intensity is more interpretable than Principal Component number 4), and thus assessing both PCA and correlation feature selection will enable a discussion weighing if the benefits of PCA are worth losing the model’s grounding in reality. The PCA will be conducted using the prcomp function from the pracma R library, which will return the same number of principal components as original features, that is, 13.

This paper will utilise the R package e1071 to train the SVM. Firstly each dataset is passed to a hyper-parameter tuning function, this compares all different combinations of kernels, gammas and “C”s, saving the best (by classification accuracy) combination for each different Kernel. The best performing hyperparameters are then sent to a 10-fold cross validation function from which an average accuracy rate can be deduced without fear of overfitting. SVM kernels are mathematical functions that allow the SVM to project its data into a higher dimensionality feature space, this study tests four common kernels: linear, polynomial, radial and sigmoid. Gamma defines how far the influence of a single training example reaches, for example in there is a high gamma of 100, only the datapoints very close to the decision boundaries will be considered as support vectors, which could lead to overfitting in non-linearly separable data. The C parameter is the SVM's reward function, defining how bad the misclassification of a datapoint is. A high C tells the SVM that any misclassification is heinous and can therefore lead to overfitting. In the hyperparameter tuning, both gamma and C are substituted with the values: [1e-03, 1e-02, 1e-01, 1e+00, 1e+01, 1e+02].

Following the training of the models, the results of the hyper-parameter tuning will be presented in a table. There will also be three diagrams created using the ggplot R package. The first will denote the results of the data analysis preceding the SVM training, which will help visualise the dataset as well as demonstrate important correlation distributions. Secondly the PCA scree plot will be plotted, a scree plot demonstrates how much of the variation in the dataset can be explained by each consecutive principal component and is important as the 'elbow' of the graph denotes the optimum number if principal components. As stated above, a benefit of using a SVM model is its potential for visualisation when the number of features is less than 4. Therefore, the final graph shown will denote the one of the trained SVM's decision boundaries.

B. *K-Nearest Neighbours*

The initial foray into kNN classification immediately presented a complication. How many nearest neighbours (k) should be used for classification? On the one hand, choosing a small value for k can be quite noisy and will have a high level of influence on the result. On the other hand, a very large value of k will have decision boundaries which are smoother (resulting in a lower variance), but it's more computationally expensive. Although there isn't a rule about how k should be chosen, there exists a general guideline that k should be equal to the square root of the number of observations.

Because there is no clear way to determine what k should be, the model will be run for a range of values of k. In this case the model will be run for values of k from 1 to 15. It was also run for a range of training partitions (P), in this case 10% to 90% in 10% steps. For each value of P, the algorithm will run for the full k range which creates a large dataset of results. It is hypothesised that this in turn will increase the likelihood of finding the best value for accuracy.

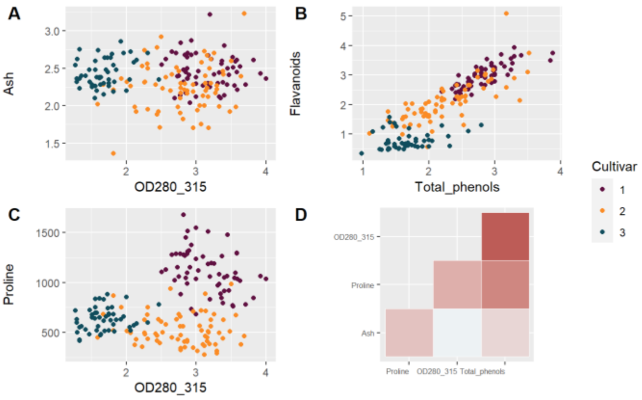
This paper utilizes the R packet Class to train the KNN model. Firstly, the dataset is passed into the kNN function. The function also takes a value for the maximum number of nearest neighbours to be used, given by Kmax. It also takes a value for the maximum percentage of data that is to be used in training, given by Pmax. The function then divides the dataset into a training set and a testing set, starting with 10% for training and increasing by 10% up to *Pmax*. For the purposes of this experiment, *Pmax* was defined to be 90%. The remainder is left for testing purposes. For each training value, hereafter referred to as the *P value*, the function is trained on the training set and then is tested on the test set. When testing it uses a *K value*, which defines the number of nearest neighbours the function should use when making its "predictions". It repeats this process, starting with the *K value* being 1 and increasing by 1 on each iteration until it reaches *Kmax*. In this case, *Kmax* is set to be 15.

There is also the question of cross validation. How do we ensure that the results of the algorithm are reliable? To solve this, a 10-fold cross validation method was employed. A 10-fold validation method divides the dataset into ten "folds". Nine of these folds make up the training set and the remaining one is the test set. The algorithm is run ten times, with each folds taking it in turns to be the test set. This allows ten values for accuracy to be collected, and a mean taken. This is the value that will then associated with the inputs (primarily the values for K and P). This repetition ensures that results are reliable.

IV. Results

A. *SVM*

Figure 1 denotes a subset of scatterplots between some independent variables in the dataset, with the colour of each point illustrating which cultivar class it belongs to, these were chosen after an initial analysis of the aforementioned 'ggpairs' plot. In figure 1A there is plotted the two variables with the lowest correlation at 0.004, ash and OD280-31. Two features that are lowly correlated are unlikely to capture the same information and by extension should bring more unique information to the model, however this graph shows that the three classes will not be easily separable using only these two features. Figure 1B denotes the highest correlation between features at 0.865 between total phenols and flavonoids. This correlation can be seen in positive linear shape on the graph, it also demonstrates the presence of collinearity, meaning that there is a strong linear relationship between these two features. Collinearity can pose an issue with modelling as it often means that the two features are capturing some other fundamental variable, therefore inclusion of both these variables may impact our models by over-weighting the importance of this underlying variable. Having conducted a qualitive analysis of all bi-feature scatter plots, it is clear that the most easily human separable representation of the classes is found between OD280-315 and Proline (figure 1C). In the final plot is the correlation matrix for our subset of the data, with darker shades denoting stronger correlation. We can see that OD280-315 and Proline sit between the two extremes, with a correlation of 0.313. This analysis showed that Proline and OD280_315 should be used in the feature selected feature subset rather than using the lowest correlation features.



a.

Fig. 1 Four plots demonstrating interesting interactions between a subset of independent variables

Having conducted PCA to obtain the second subset of data, the Scree graph has been plotted in figure 2, also denoting an elbow at 4 principal components. From this analysis we conclude that the PCA subset will contain principal components 1 through 4.

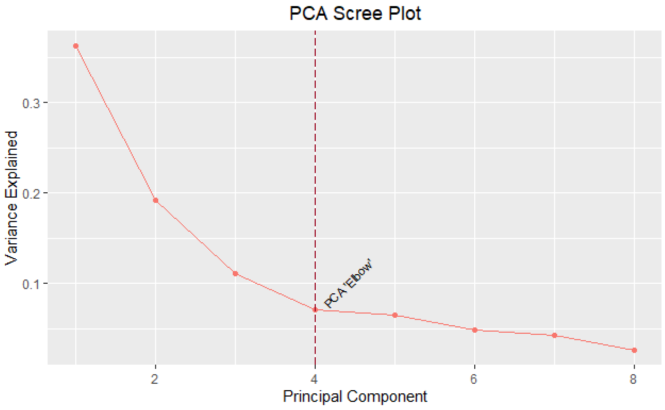


Fig. 2 PCA Scree plot demonstrating the first 4 principal components explain the majority of the data variance.

Having now concluded the exact three data subsets to be used, the results of the SVM hyperparameter tuning for each subset of data is denoted in figure 3, note that the gamma parameter is not needed in a linear kernel.

TABLE I. SVM RESULTS

Data	Hyper Parameters			Classification Accuracy (sd)	Time to Run (Seconds)
	Kernel	*Gamma*	*C*		
Full	Polynomial	0.1	10	93.17% (8.38)	0.861
PCA	Linear	N/A	1	93.24% (6.99)	0.54
FS	Linear	N/A	0.1	85.95% (10.15)	9.945

Fig. 3 Table denoting SVM training Results

The best performing SVM was trained, somewhat surprisingly, using the PCA dataset, achieving an accuracy of 93.24% with the smallest spread of cross-validation results at a 6.99 standard deviation. So not only is it more accurate, but it is more reliable in all folds of cross-validation. PCA outperforming the whole dataset may indicate that some variables are detracting from the accuracy of the model.

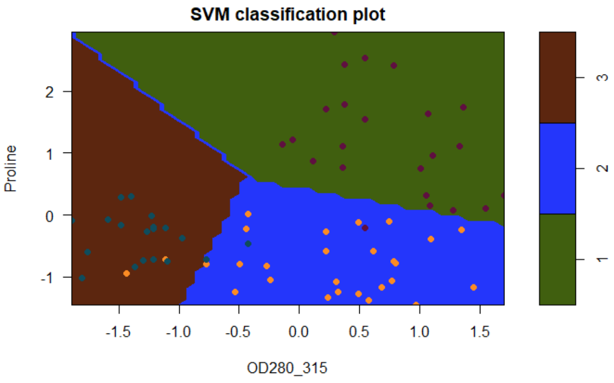


Fig. 4 Visualisation of SVM based on manual feature selection

One of the advantages of having a dual feature model is that these SVMs can be depicted graphically, and whilst it is the worst performing SVM out of the batch, it is still useful to visualise.

Figure 4 denotes cultivar class '3' points in blue, with their cluster region (defined by the SVM) in brown, as we can see here there are two mis-classified yellow points within this region. The yellow points belong to cultivar class 2 which is defined by a blue coloured classification region. This region contains two misclassified class 3 points and one misclassified class 1 point. The cultivar class 1 region, denoted by a green background is the only region that has a 100% classification rate, containing no misclassified points.

B. K-Nearest Neighbours

As has already been mentioned, there is an uneven split of classes in the dataset (59, 71 and 48 datapoints for class 1, 2 and 3 respectively). The dataset also has a huge range of standard deviations (sd) and mean values. Once the data had been standardised, it was much easier to work with. The ggpairs plots suggests that Proline/OD280_315 look the most easily separable, but Proline/Hue & Proline/Color_intensity were also possible. As a result, figures 5-7 were made to confirm these theories.

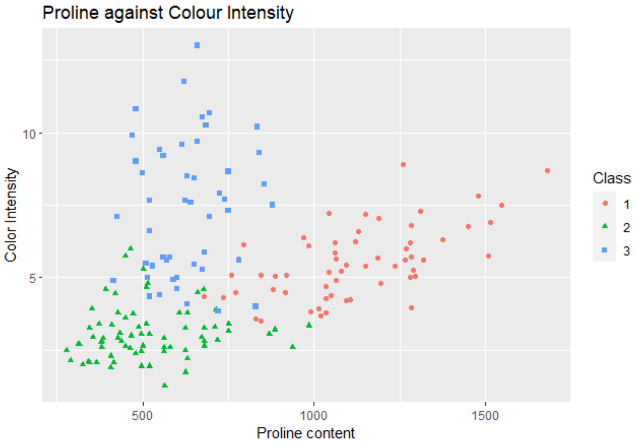


Fig. 5 Proline against Colour Intensity for all data points. Grouped by class

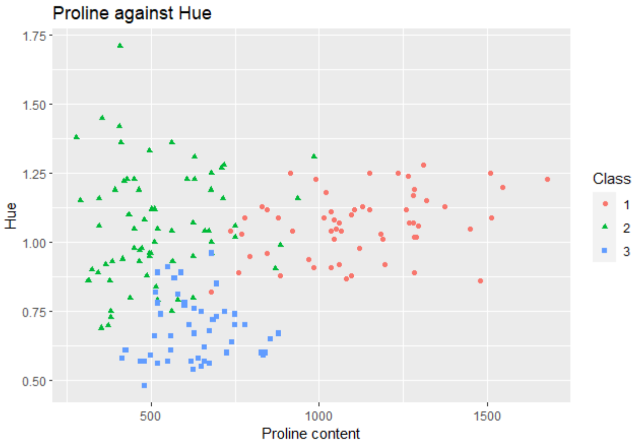


Fig. 6 Proline against Hue for all data points. Grouped by class

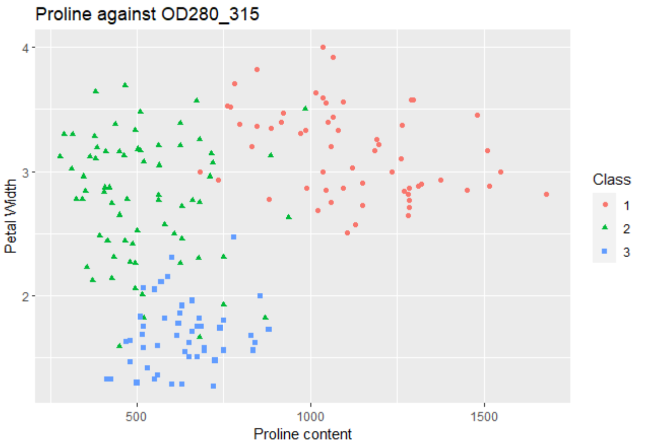


Fig. 7 Proline against OD280/315 for all data points. Grouped by class

Analysis of the results using all attributes revealed that the partition using P = 50% gave the highest mean accuracy, with a Mean Classification accuracy across the partition of 98%. This can be seen in table 2. Looking at figure 8, it can also be seen that there is a great deal of fluctuation in the accuracy as K varies. Figure 9 also shows this but also shows that the results are much more consistent at higher P values than at lower ones such as 10% and 20%. They also suggest that the consistency drops off somewhat as P is increased past 70%.

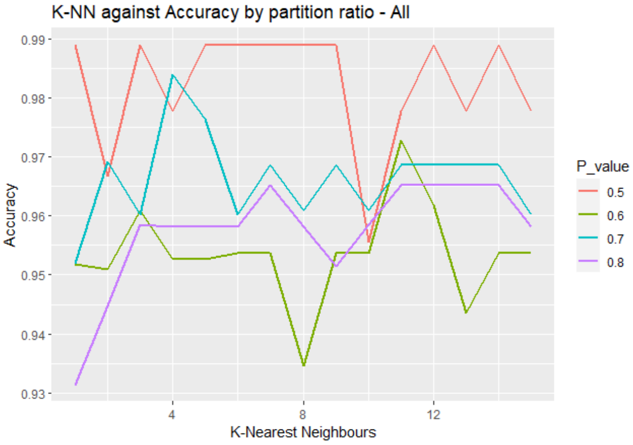


Fig. 8 KNN against Accuracy for partition training ratios 50% to 80%

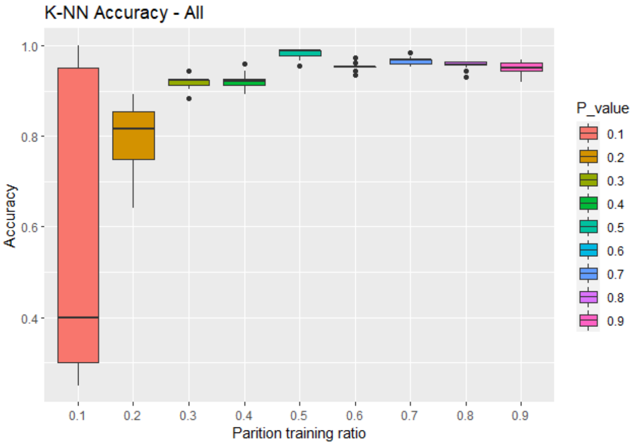


Fig. 9 Accuracy against Partition training ratio for all attributes

Next Proline and OD280/OD315 were the two attributes used in the KNN function. Here, the 90% partition is shown to have the highest Mean Classification Accuracy at 99% (table 2). However, partitions at 80% and 60% are less than 1% less accurate. This is well demonstrated by figure 11 which also shows that they have the lowest standard divisions of all the partitions (3%).

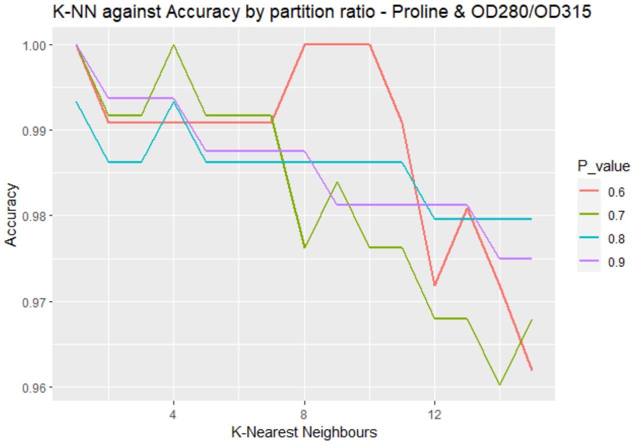


Fig. 10 KNN against Accuracy for partition training ratios 60% to 90%

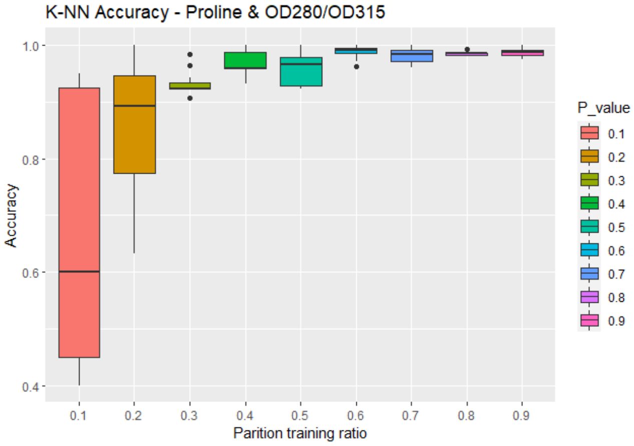


Fig. 11 Accuracy against Partition training ratio for Proline & OD280/315

Finally, Proline and Hue were the two attributes used in the KNN function. In this case, the 70%-90% partitions were the most accurate on average. Unlike the other two KNN functions, Proline and Hue seems to show a clearer trend in the data. Figure 12 shows that each partition decreases in accuracy as the number of nearest neighbours increases by about 1-2%. This is different to the previous two methods as they don't have this trend, instead, their decline in accuracy seems to be much more random (figures 8 & 10). Figure 12 also shows that up until around k = 8, the function classifies the test set with near 100% accuracy, which is notably better than the other methods. Another noteworthy result can be seen in figure 13. Here we see that the standard deviation, particularly at the higher partition ratios, is much lower than that of the previous two results. Just 1-2% compared to 3% or 5-6%.

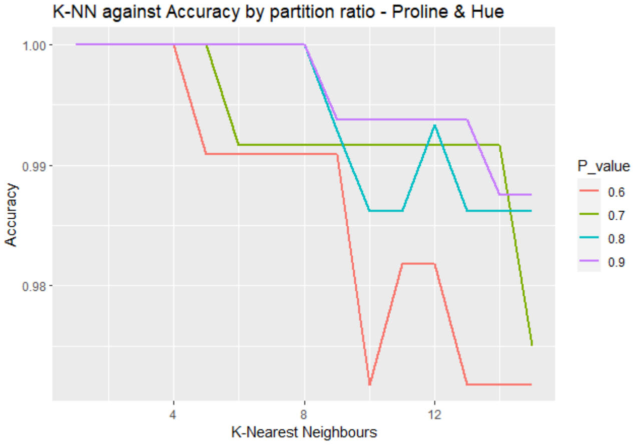


Fig. 12 KNN against Accuracy for partition training ratios 60% to 90%

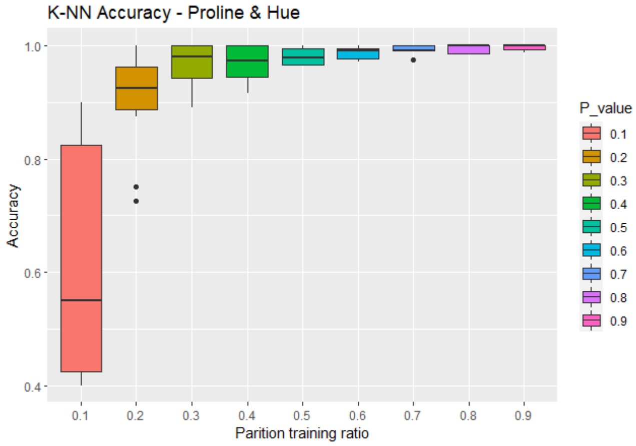


Fig. 13 Accuracy against Partition training ratio for Proline & Hue

As a sidenote, a low k value is reasonably accurate, but this drops off quickly as the k value is increased. This is particularly visible for low *P values* can be seen in figure 14. This is the case for all attribute groups used. However, the standard deviation of these accuracy results is extremely low (figures 9, 11 & 13).

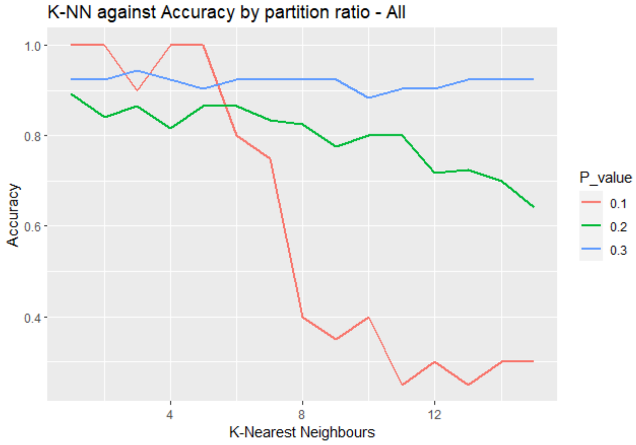


Fig. 14 KNN against Accuracy for partition training ratios 10% to 30%

TABLE II. K-NN RESULTS

Data	Best result for each variation			
	Partition used for training	***Mean Classification Accuracy (sd)***	***K-NN Peak Accuracy***	***Time to run (seconds)***
Full	50%	98% (5%)	1,3,5-9,12,14	0.674
Proline & OD280/OD315	60%	99% (3%)	1 & 8-10	0.295
Proline & Hue	80%	99% (1%)	1-8	0.230

Fig. 15 Table denoting K-NN training Results

V. Discussion

The aim of this paper was to gain insight into what features best define each of these cultivars and which method best describes the wine dataset as a whole. Before discussing the different models however, it is worth considering the insights gained from the preliminary data analysis. We saw in this analysis that Class 1 is characterized by high OD280-150, high proline, high phenols and low colour intensity. Class 2 is characterized by high OD280-350, low Proline, middling phenols and low colour intensity. Class 3 is characterized by low OD280-350, low Proline, low phenols, and high colour intensity. These insights, while crude, paint a much more vivid picture of how these features define each cultivar than what can be gather from the ML techniques.

A. kNN

Comparing the results of the kNN methods, it seems clear that using fewer attributes is indeed better for classification, as was expected. It also showed that the standard deviation of the results is noticeably lower for a decreased number of attributes. Interestingly, despite the fact that Proline & OD280 seemed to have better separation than Proline & Hue in pre-processing, the later has a lower standard deviation than the former, which is contrary to what would be expected. A comparison of figures 10 and 12 also shows a marginally higher accuracy than Proline & OD280 (table 2). In general, then, it would seem that data classification in kNN works best when a partition of 60%-90% is used for training as this produces the most accurate results with the smallest standard deviation. It also confirms the belief that 2 attributes are much more precise than many in kNN.

With regards to the number of nearest neighbours that should be used (*K value*), the results from the Proline & Hue and the Proline & OD280 models suggests that a lower value of K is significantly more accurate than a higher one. This leads us to suggest that in general, no more than 4 nearest neighbours should be used on kNN.

When compared to other attempts at wine cultivar classification, this model competes with the best. As stated in the literature review, the GDC-kNNC approach taken in 2007 [11] returned a classification accuracy of 98.7%, which is 0.3 percentage points lower than the accuracy presented in this paper. Whilst that paper was developing new classification techniques, this paper merely implements the model as a means to further insight into the world of wine.

B. SVM

The principal objective of the SVM methodology was to determine the best subset of features for optimal SVM classification and by extension create an evaluation of the value of PCA when it obfuscates the context of the features.

The results demonstrate that the optimal feature set for SVM classification was that of 4 principal components. This feature set outperforming the whole feature set indicates the inclusion of some data that detracts from the model. This is likely to be the inclusion of the colinear Flavanoinds/Total_phenols features. Inclusion of highly correlated features may cause the SVM to give a higher weighting to the underlying feature connecting the two features than what would otherwise be optimal.

The PCA trained SVM outperformed the manually selected features both in accuracy, accuracy spread and training time. The high training time of the latter (9.945 seconds) is due to the SVM having to work harder to separate the two classes given only two features, this means the SVM will have to project to a much higher dimensionality feature subspace and use more points in this space as support vectors than with the other two models. Considering these points, the loss of context when using PCA is worth it in this case. However, in comparison to the full feature set, the PCA approach was only 0.32 seconds faster, and only 0.06 percentage points more accurate, therefore leading to the conclusion that using the full feature set would prove better than that of PCA overall. It is also important to note that small discrepancies such as this may become exacerbated as the number of data points in a dataset increase, so PCA may become worthwhile when the number of datapoints is greater than 178.

In the wider context of the literature, one can see that the specific SVM approach taken by this paper is sub-par in comparison. For example, this paper achieved an accuracy of 93.24%, which is far below that of the 99.4% achieved by Bredensteiner and Bennett in 1999 [4]. The fact that their paper was developing new SVMs and that this paper was merely employing an established R library only goes to highlight this paper's poor SVM performance.

C. *Comparison*

In a comparison of the Time to Run between the kNN and SVM models, one can observe that kNNs with high values for K are considered computationally expensive. However, the models were still able to run significantly faster in most cases than the SVM datasets. The best result for SVM can be seen in row 2 of table 1 (fig. 3) with a training time of 0.54s vs the much shorter training time of kNN of 0.23s which can be seen in table 2 (fig.15). Comparing the accuracy levels of the best results from each method, we can also see that kNN was able to achieve a much higher accuracy than SVM was. (99% compared to 93.24%). It was also able to achieve a much lower standard deviation of cross validation accuracies than the SVM (1% compared to 6.99%). In addition to this, the running time was also lower, despite running for 15 different K values (0.203s compared to 0.54s). This is important because it tells us that although SVM is more accurate for high dimension data, if that data can be simplified down into 2 dimensions, then KNN becomes vastly superior.

kNN also outperforms SVM in a contextual way. The best performing kNN was trained on the features proline and hue, which achieved a higher accuracy rate than the best SVM. This shows that the decisions between cultivars are easily explained with these two features. This conclusion has real world implications for the wine industry, for example if classifying new wines into these cultivars, only hue and proline content data would need be collected before classification. Another real-world application could be 'new world' wine makers trying to imitate French or Italian wines can now focus on creating the correct hue and proline content over other methods. The improved classification can also be used in fraud detection or the development of new wines. The kNN therefore is less of a black box than the SVM approach, more accurate, faster to train and outperforms some of the relevant literature. This in conjunction with the insights taken from the preliminary research creates a solid base for understanding and classifying wine cultivars.

VI. Conclusion

This paper set out with a clear goal. To utilise the popular UCI ML Library Wine Dataset to gain insight into what defines different wine cultivars. To this end, a preliminary statistical investigation was conducted, which, amongst other things, found that there were certain features that presented more easily separable representations of the classes, that certain features presented co-linearity and that the spread of values in the dataset would warrant z-standardization. Following this, both a support vector machine and a k-nearest-neighbour's model were created. The SVM was trained on the full dataset, a selected feature set and a PCA feature set, the latter of which performed best. The kNN method found a much higher accuracy using only two features from the original 13, proline and hue.

Future research could explore switching methodologies to enable a more in-depth comparison of the two classification methods. For example, reproducing the kNN algorithm but training it with PCA features, or reproducing the SVM but running it on varying quantities of the data rather than the 90-10 train-test split it employed. There should also be further investigation regarding the lack of SVM proficiency seen in this paper. The literature has seen much higher classification accuracies which suggests one of two issues. Either the SVM was deficient due to its coded implementation or the R package e1071 itself is somehow deficient, either possibility warrants further investigation. These ideas for future research would go some way to answering the question as to why kNN outperformed the SVM so dramatically.

In summary, this paper has gathered some insights into the determinants of cultivar classification, as well as a strong overall view of the data. Beyond this, issues and comparisons have been discussed concerning support vector machines and kNNs. Having found this implementation of kNN outperforms the SVM in accuracy, training time, interpretability, and simplicity. This paper has also considered the real-world importance of this type of research. Deeper insights into what makes a wine what "class" can contribute to fraud detection, help new world wines break into the market or help in the development of new cultivars. And with wine playing as much of an important role in our modern world as it did in the ancient, it is certain that a better understanding of this bacchian delight could never be a bad thing.

Contributions

Joel Dolman was responsible for all SVM related work. Including the SVM work in the literature review as well as the SVM methodology, results, and discussion whereas Charles Chudley was responsible for all kNN related work in these same areas. Shared parts of the paper such as abstract, introduction, preliminary data analysis, comparison and conclusion were done in an iterative manner, with one partner writing, the other reviewing. This process was then repeated until both members were happy with the content, this approach was taken such that both members had equal input in these areas.

REFERENCES

[1] Forina, M., Leardi, R., Armanino, C., Lanteri, S., Conti, P. and Princi, P., 1988. PARVUS: An extendable package of programs for data exploration, classification and correlation. Journal of Chemometrics, 4(2), pp.191-193.

[2] Aeberhard, S., Coomans, D. and De Vel, O., 1994. Comparative analysis of statistical pattern recognition methods in high dimensional settings. Pattern Recognition, 27(8), pp.1065-1077.

[3] Tan, P.J. and Dowe, D.L., 2004, December. MML inference of oblique decision trees. In Australasian Joint Conference on Artificial Intelligence (pp. 1082-1088). Springer, Berlin, Heidelberg.

[4] Bredensteiner, E.J. and Bennett, K.P., 1999. Multicategory classification by support vector machines. In Computational Optimization (pp. 53-79). Springer, Boston, MA

[5] Ash, C.S., 1915. Contributions of the Chemist to the Wine Industry. Industrial & Engineering Chemistry, 7(4), pp.273-274.

[6] Ebeler, S.E. and Thorngate, J.H., 2009. Wine chemistry and flavor: looking into the crystal glass. Journal of agricultural and food chemistry, 57(18), pp.8098-8108.

[7] Guth, H., 1997. Identification of character impact odorants of different white wine varieties. Journal of Agricultural and Food Chemistry, 45(8), pp.3022-3026.

[8] Villano, C., Lisanti, M.T., Gambuti, A., Vecchio, R., Moio, L., Frusciante, L., Aversano, R. and Carputo, D., 2017. Wine varietal authentication based on phenolics, volatiles and DNA markers: State of the art, perspectives and drawbacks. Food Control, 80, pp.1-10.

[9] Riul Jr, A., de Sousa, H.C., Malmegrim, R.R., dos Santos Jr, D.S., Carvalho, A.C., Fonseca, F.J., Oliveira Jr, O.N. and Mattoso, L.H., 2004. Wine classification by taste sensors made from ultra-thin films and using neural networks. Sensors and Actuators B: Chemical, 98(1), pp.77-82.

[10] Zhong, P. and Fukushima, M., 2007. Regularized nonsmooth Newton method for multi-class support vector machines. Optimisation Methods and Software, 22(1), pp.225-236. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[11] P. Viswanath, M. Narasimha Murty, Shalabh Bhatnagar, Partition based pattern synthesis technique with efficient algorithms for nearest neighbor classification, Pattern Recognition Letters,Volume 27, Issue 14, 2006, Pages 1714-1724, ISSN 0167-8655

[12] Guvenir, H.A. and Akkus, A., 1997. Weighted k nearest neighbor classification on feature projections. In Proceedings of the 12-th International Symposium on Computer and Information Sciences.

APPENDIX

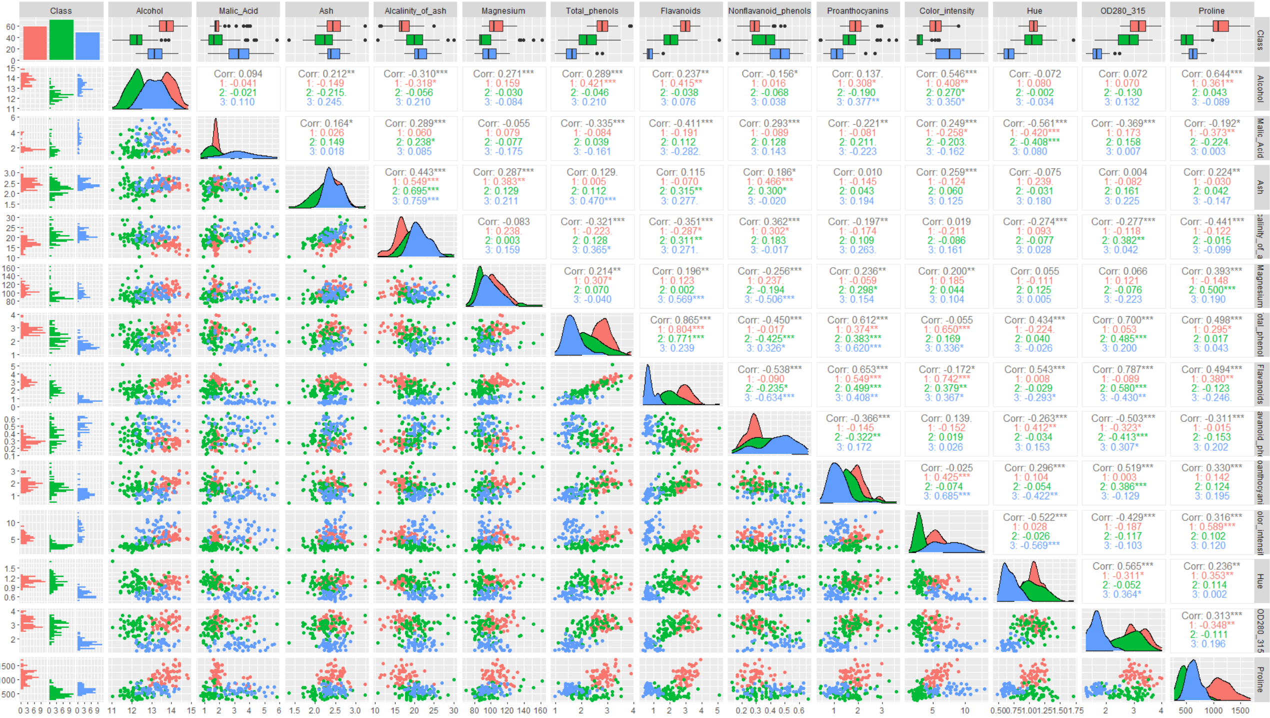


Fig.15 ggpairs plot used in initial analysis