MLDM 2021 Coursework Group: *Mars*

Ryan B (**[redacted]**) RB[redacted]@SURREY.AC.UK
Charlie B (**[redacted]**) CB[redacted]@SURREY.AC.UK
Adlan E (**[redacted]**) ME[redacted]@SURREY.AC.UK
Chris E (**[redacted]**) CE[redacted]@SURREY.AC.UK
Chris J (**[redacted]**) C[redacted]@SURREY.AC.UK
Brian N (**[redacted]**) BN[redacted]@SURREY.AC.UK

Abstract - In this report, we present several machine learning model implementations in order to solve two distinct classification challenges: The classification of wine quality based solely on physicochemical characteristics and the classification of the edibility of mushrooms in the Agaricus and Lepiota mushroom families. The models used for the classification tasks were Decision trees, Support-vector machines, neural networks, K-nearest neighbour, and Naive Bayes. K-means clustering and inductive logic programming was also used to further our understanding of the datasets. Model hyper-parameters were optimised using a systematic approach for each task individually. The most effective model found for the wine classification problem was the simple decision tree, with an accuracy and F1-score of 0.877 and 0.872 respectively. Decision trees, Support-vector machines, neural networks, K-nearest neighbour, and inductive logic programming all achieved perfect classification on the mushroom edibility task.

1. Project Definition

This project aims to develop machine learning models to tackle two distinct classification challenges. The first seeks to identify the quality of a red or white wine using only physicochemical input features such as pH or alcohol content. The second seeks to determine whether a particular mushroom is edible or poisonous based on physical characteristics such as cap shape and environmental characteristics such as habitat.

1.1 Wine Quality Dataset

The Wine Quality dataset was sourced from the University of California Irvine Machine Learning Repository and published to Kaggle as a machine learning challenge[1][2]. The dataset is split between the red and white varieties of 'Vinho Verde' wine, each with 11 numeric physicochemical attributes to be used to estimate the quality certification between 0 and 10 for which 0 is the lowest possible certification and 10 the highest. Our primary objectives for this dataset are:

1. Examine how effectively machine learning models can be applied to predict the quality certification of a wine based solely on physicochemical features.
2. Discover the particular physicochemical features which contribute most highly to the quality certification of a wine such that wine makers could use this information to improve their wine making processes.

Attribute	Description	Type	Examples
Class	The type of wine	Categorical	red, white
Fixed acidity	The concentration of tartaric acid in the wine (g/L)	Continuous	5 9, 6 2, 7 2, 8 1
Volatile acidity	The concentration of acetic acid in the wine (g/L)	Continuous	0 270, 0 510, 0 620
Citric acid	The concentration of citric acid in the wine(g/L)	Continuous	0 08, 0 32, 0 47
Residual sugar	The concentration of residual sugar in the wine (g/L)	Continuous	1 6, 6 9, 8 5
Chlorides	The concentration of sodium chloride in the wine (g/L)	Continuous	0 045, 0 068, 0 090
Free sulfur dioxide	The concentration of free sulphur dioxide in the wine (mg/L)	Continuous	14 0, 30 0, 47 0
Total sulfur dioxide	The concentration of total sulphur dioxide in the wine (mg/L)	Continuous	8 0, 97 0, 170 0
Density	The density of the wine (g/mL)	Continuous	99400, 0 99510, 1 00100
pH	The pH of the wine	Continuous	3 00, 3 26, 3 57
Sulphates	he concentration of potassium sulphate in the wine (g/L)	Continuous	0 45, 0 71, 0 82
Alcohol	The alcohol content of the wine (vol %)	Continuous	8 8, 9 5, 11 0
Quality	The human-assessed quality certification of he wine, with 0 being the lowest certification and 10 being the highest	Categorical	0-10

Table 1: Data dictionary for the wine dataset.

1.2 Mushrooms Dataset

The Mushrooms dataset was also originally sourced from the University of California Irvine Machine Learning Repository, published later to Kaggle as a machine learning challenge [3][4]. The dataset is composed of physical descriptions of mushroom samples from the gilled Agaricus and Lepiota mushroom families. Alongside the physical characteristic descriptions, each mushroom sample is labelled as either edible or likely poisonous. The primary objectives for this task are:

1. Examine how effectively machine learning models can be applied to predict the edibility of mushrooms based on physical and environmental characteristics.
2. Examine the correlation of each characteristic on the edibility of a mushroom such that potential rules and heuristics could be derived such that foragers could use this information to spot potentially poisonous mushrooms.

Attribute	Description	Type	Examples
Cap shape	The shape of the mushroom cap	Categorical	b – Bell, x – Convex, f - Flat
Cap surface	The texture of the cap surface	Categorical	s – Smooth, g – Grooves, y - Scaly
Cap color	The colour of the cap	Categorical	n - Brown, p - Pink, y - Yellow
Bruises	Whether the mushroom has bruises	Categorical	t – True, f - False
Odor	The odour of the mushroom	Categorical	a – Almond, f – Foul, n - None
Gill attachment	The attachment of the gill	Categorical	a – Attached, d – Descending, f - Free
Gill spacing	The spacing of gills on the mushroom	Categorical	c – Close, w – Crowded, d - Distant
Gill size	The size of the gills	Categorical	b – Broad, n - Narrow
Gill color	The colour of the gills	Categorical	k – Black, n – Brown, o - Orange
Stalk shape	The shape of the mushroom stalk	Categorical	e – Enlarging, t - Tapering
Stalk root	The state of the stalk root	Categorical	b – Bulbous, r – Rooted, ? - Missing
Stalk surface above ring	The surface texture of the stalk above the mushroom ring	Categorical	f – Fibrous, y – Scaly, k - Silky
Stalk color above ring	The surface texture of the stalk below the mushroom ring	Categorical	f – Fibrous, y – Scaly, k - Silky
Stalk color below ring	The colour of the stalk below the mushroom ring	Categorical	n-Brown, p - Pink, y - Yellow
Veil type	The structure of the mushroom veil	Categorical	p – Partial, u - Universal
Veil color	The colour of the veil	Categorical	n – Brown, o – Orange, w - White
Ring number	The number of rings on the mushroom	Categorical	n – None, o – One, t - Two
Ring type	The appearance of the mushroom rings	Categorical	c – Cobwebby, f – Flaring, p – Pendant
Spore print color	The colour of the spore prints	Categorical	n – Brown, o – Orange, w - White
Population	The surrounding population of the mushroom	Categorical	a – Abundant, c – Clustered, y - Solitary
Habitat	The surrounding habitat of the mushroom	Categorical	g – Grasses, l – Leaves, d - Woods
Class	The edibility of the mushroom	Categorical	e – Edible, p - Poisonous

Table 2: Data dictionary for the mushrooms dataset.

1.3 Performance Metrics

The challenge then is split into two classification tasks, one multi-class and one binary-class. For the wine dataset, the primary statistics by which models will be evaluated are the weighted classification F1-score and the classification accuracy. Since a large degree of class imbalance can be observed in the dataset, F1-score serves as a more reliable primary metric to maximise for each model. Using the weighted average for this score allows the evaluation to further take into account this class imbalance, without being skewed too far by the least populous classes. For the Mushrooms dataset, the same two primary statistics are examined. F1-score is used as the primary metric for this evaluation. Whereas accuracy is concerned primarily with true positives and true negatives, F1-score deals better when false negatives and false positives are a concern. Due to the nature of the problem, it is critical that false positives are avoided (i.e a poisonous mushroom is classified as safe to eat), and so F1-score wins out in this case.

1.4 Models

The models that will be applied to solve these tasks in this report are: Decision trees, Support-vector machines, neural networks, K-nearest neighbour classifiers, K-means clustering, Naive Bayes classifiers and inductive logic programming.

2. Data Preparation

2.1 Data Loading

Both the mushroom and wine datasets used in this project were encoded using the comma separated values (csv) data format, and loaded in at run-time using the Pandas library.

2.2 Data Cleaning

Mushroom Dataset

The source mushroom dataset was found to have no null fields or missing values. Duplicate rows and columns were tested for with none found. Since all fields were categorical, it was not necessary to detect and correct outlier values. Given these observations, the mushroom dataset was considered clean

Wine Dataset

34 rows in the sourced wine dataset were found to contain at least one null field. The target class distribution of the records containing null values was examined and each of these records was found to belong to the populous classes 5, 6 and 7. With this information, it was deemed acceptable to simply drop the rows containing null values, as we had sufficient examples for their classes already.

Each field was then tested for outliers, with rows dropped when a row containing an outlying field was detected.

Finally, records and fields were tested for duplicates. With none found, and having removed null values and outliers, the dataset was considered clean.

2.3 Variable Transformation

In order to prepare the categorical data in the mushrooms dataset for modelling, each field was one-hot encoded, taking the total number of fields from 20 to 108.

Functions were created to offer further transformations for use in the modelling stage. Standardisation and normalisation functions were created to fit the wine data, as well as binning functions to create a discretised versions of the dataset. Additionally, a feature selection method using a chi-squared model was implemented to create datasets with reduced dimensionality for both the wine and mushroom datasets by using the most salient features obtained from that model.

2.4 Data Exploration and Visualisation

Key findings:

<u>Wine</u>

- The dataset is unbalanced in terms of the two categorical fields, "type" and "quality" with white wines outnumbering reds by over three-to-one and the majority of wines classified as quality 5, 6 or 7.
- All continuous numerical features have extreme outliers, many standard deviations from the mean.
- Compared with reds, white wines tend to have:
    - Lower volatile acidity, fixed acidity, pH, sulphates, chlorides and density.
    - Higher residual sugar, free sulfur dioxide and total sulfur dioxide.
- There are no obvious relationships between input features and quality, other than that wines rated more highly tended to have a higher alcohol content.

<u>Mushrooms</u>

- The dataset is well balanced with a 48/52 split between poisonous and edible mushrooms.
- Mushrooms with bruises value “f”, odor “f”, gill-size “n” and stalk-surface ring values of “k” tend to be poisonous.

3. Model Development

3.1 Clustering

K-means Clustering is an unsupervised model that combines common data points together into clusters. As both of the datasets represent a supervised problem, due to them both being labeled. Due to clustering being an unsupervised method, it cannot use the same metrics and evaluation as the other models, also it cannot be used to classify the models. Hence, K-means clustering is used to get a better insight to the spread and behaviour of the datasets, and to uncover what models would work best with the spread of data. Metrics like homogeneity, completeness, silhouette, and runtime should be considered and should be relatively good for both datasets, even though we are only using the model to further our understanding of the datasets. Cluster plots will be the main source of information from this model, as it provides a visual representation of the decision space, and changes from experiments can easily be recorded.

K-Means Clustering provides high interpretability of the datasets, will work well with the large datasets, while also guaranteeing convergence. However, it is not designed for classification tasks, and clusters have to be chosen manually. Due to this, parameters and features like cluster size, k-means algorithm type, and PCA dimension reduction will have to be experimented with, to get the best clustering model.

3.2 Decision Trees

Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the features of the data. The advantages of decision trees is that it works well with numerical and categorical features, it also requires very little data preprocessing, it also does not make assumptions about the shape of the data as it is a non-parametric model and feature selection happens automatically: unimportant features will not influence the result. Some disadvantages of decision trees would be that they are prone to overfitting and prone to sampling errors. Decision trees were chosen as one of the models due to its proficiency with multiclass and binary data.

The main library used is the latest Sklearn (scikit-learn) library to train and test the models. Sklearn is also open source, meaning that if there are any problems, previous experiences posted online can be used to debug or improve code. Decision trees also have access to the shap library, which allows investigations into the features that contribute to the datasets. xgboost and random forest are also tools available to compare the results achieved in predicting both datasets.

3.3 Support Vector Machine

Support Vector Machines (SVM) are widely used in classification and regression tasks. A SVM motive is to find a hyperplane in N-dimensional space that distinctly separates a set of given data points. The optimal plane is that which has the maximum distance between data points of each of the classes. Unfortunately, SVM lacks interpretability when compared with models such as Decision Trees. This is because the output is the class of the data point instead of the reasoning behind the assignment to a particular class. Furthermore it is not possible to extract classification probabilities from SVM because it is not a probabilistic model. In addition to this, SVM only works when classes can be well separated by a simply drawn plane.

Both datasets are classification tasks with the wine dataset being a multi-class and mushroom dataset being binary. Despite the disadvantages of the model, the SVM has been chosen as SVM models typically perform well on classification tasks. SVM should be able to apply a decision plane even in the multi-class classification task.

3.4 Artificial Neural Networks

An artificial neural network is a massively parallel distributed processor consisting of computational units known as “neurons”, designed to mimic the brain. It is a supervised learning method that can be applied to classification or regression tasks.

In this experiment, artificial neural networks – single perceptron or single/multi-layer perceptrons - were applied to mushroom and wine classification tasks. Given the very distinct nature of data in the two datasets, it was expected that the neural network model variant providing the best classifier would be different in each case.

The initial expectation was that mushrooms would be more readily classified than wines, due to the nature of the targets. Mushrooms are definitively classed as either edible or poisonous whereas wine classifications are subjective having been provided by a human wine reviewer. Even if the same reviewer provided all of the wine quality ratings, it is impossible to know if ratings were applied consistently. It is likely that any model will struggle to predict classifications assigned in this way.

Neural networks can construct functions of great complexity to fit data points during training. If such a function existed to determine wine quality from the continuous numerical physicochemical inputs then a neural network, however complex, is likely to provide only an approximation. If such a function existed for our mushroom dataset, it would be possible to construct a Boolean function to represent this relationship using a neural network with just a single hidden layer. Of the possible node arrangements, single Perceptron models were deemed inappropriate for the wine dataset as they would not be able to derive the required non-linear function from the continuous feature inputs.

A major drawback of neural network models is the time required to train them. The back-propagation method used for calculating weight updates is computationally expensive and becomes more so for larger and more complex networks with more nodes. Broadly, results from neural network models also suffer from a lack of explainability. Through training, a set of weights are arrived at which describe the approximate relationship between data inputs and model predictions. These weights are not easily interpretable so without going into technical details, it is generally difficult to explain to those unfamiliar with neural networks why a model has returned the predictions it has.

Neural networks can be sensitive to outliers in input feature data, with models of sufficient complexity attempting to fit extreme data points, resulting in overfitting and an inability of our model to generalise. Extreme outliers exist in the wine dataset but various techniques exist to reduce the impact of these.

3.5 K Nearest neighbours

K Nearest neighbours (KNN) is a supervised learning model, considered to be part of the foundation of instance-based learning. KNN analysis plots training data points in a multidimensional space and when presented with an unknown point reviews the distance in this space with respect to each point and classifies based on the classes on the “k” nearest neighbours. Predominant advantages are the “zero time” training of the model as test examples are just stored. As a result, no information is lost in the training process.

The disadvantages of the model are linked to this. Continual storage of information results in continuous growth of the storage requirement of the model. Additionally, whilst more information stored should provide more accurate classification, the other side of that advantage is that the processing power requirement constantly increases, or the query time constantly increases. There is potential for the model to suffer from the curse of dimensionality as every point is always considered when predicting..

The libraries used for this model were pandas and numpy, with items from sklearn including model_selection, neighbours, and metrics. The parameter to change for the KneighborsClassifier is n_neighbours, which changes how many points closest to the point being estimated are being considered in the classification.

Plotting exact chemical qualities and using KNN may take into account features in a non-optimal fashion. Data exploration revealed features to not be of equal relevance, especially for different wine types. Therefore ,since KNN uses Euclidean distance to calculate the similarity of data

points, the KNN model may struggle to model the wine data. With respect to mushrooms, the data is binary encoded making it a lot more defined when mapping class boundaries and considering nearest neighbours. KNN performance is likely to be fast despite the mention of the curse of dimensionality as the preprocessed datasets are relatively small (1724 and 4344 KB, compared to potential GBs of data).

3.6 Naive Bayes

Naïve Bayes classifiers are a class of machine learning model that utilise Bayes Theorem to derive conditional probabilities that map inputs to class labels. Naïve Bayes models are among the most efficient and simple to implement offering comparable classification performance to more complex methods, and as such are often chosen to create initial baseline solutions. Whilst Naïve Bayes models rely on the assumption that input features have conditional independence, they often perform relatively well even when this assumption is violated. Naive Bayes models do however require a large amount of training examples to calculate probability distributions over classes. For the mushroom dataset, this does not pose a significant problem as there are thousands of examples of both edible and poisonous mushrooms present to train on. With the wine dataset, there are only a handful of examples at the extremes and as such the classifiers will struggle to derive accurate posterior distributions on those classes.

Due to the nature of the training algorithm, Naive Bayes classifiers are unable to predict classes that are not present in the training dataset. This poses an issue for the wine dataset, wherein wines can be certified with a rating from 0-10. Since the dataset contains examples only from 3-9, the classifier will always predict 0 probability that an input belongs to class 0, 1, 2 or 10.

For both Wine and Mushrooms datasets, correlations between attributes are low and as such the chosen Naïve Bayes classifiers are expected to produce reasonable benchmark F1 and accuracy scores. Three Naïve Bayes classifiers are chosen to model each task: Multinomial Naïve Bayes, Gaussian Naïve Bayes, and Bernoulli Naïve Bayes, all of which make different assumptions on the distribution of the input data and as such will require bespoke pre-processing for optimal performance. .

3.7 ILP and Relational Learning

ILP relational learning is a method used to extract logical rules from a relational dataset of background knowledge.
Relational learning was chosen to be used on the mushroom dataset to provide a highly human-interpretable set of rules that can describe the edibility of mushrooms in the dataset. This is very beneficial for the mushrooms dataset, as if this was in a real world scenario, being able to tell users exactly what to look for if they do not have time to use the other models. Even though the model itself is relatively quick to train and implement, a database must first be converted into the relational dataset form of background knowledge, which can be a very long and painstaking process, even when using a script.
Although the model can be used to predict if a mushroom is poisonous in its final state, the main aim of this model is to induce logic rules and the quality of the rules. Therefore, the only metric that will be used is to check the accuracy of each rule with the entire dataset, to prove the rules are logically sound. Due to the perfect metric scores of other models, the logic rules are expected to be 100% accurate, and if they are accurate, it proves that the dataset is logically sound and the models are working correctly, despite the suspiciously perfect scores.

4. Model evaluation / Experiments

4.1 Clustering

By looking at the clustering results of the wine quality dataset in fig 1, two groupings representing red and white wine clusters can be observed, implying that red and white wine have characteristically distinct chemical makeups. Data points in the white wine cluster are more densely packed as compared to the red wine cluster, suggesting a higher degree of similarity between individual variants of white wine which could lead to less separable classes when trying to derive quality with classification algorithms. Red wine sees a more spread distribution and as such is potentially more separable. From this then it can be hypothesised that the models will perform better on red wine samples than white.

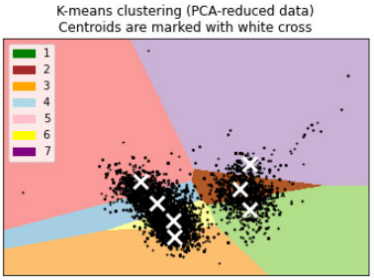

Figure 1: Clustering results for the wine dataset, showing white wine on the left most clusters, and red wine on the right most clusters.

When performing the same clustering only on wine with quality 5 and 6 (figure 2), the shape of the clusters follow a strikingly similar distribution when compared with figure 1. From this it can be hypothesised that the quality classifications are not cleanly separable, at least in this dimensional space, and rather sit on top of eachother. This observation may suggest that the classification algorithms will struggle to model this data, as algorithms such as perceptron-based models and SVMs rely heavily on the separability of classes. Further, this result may suggest that additional features outside of the physicochemical are required to properly separate these classes. This was a predicted shortcoming of the dataset, which does not include factors that would have gone into the wine certification such as tasting notes, winery/region prestige,

grape type. Relying only on the chemical makeup may then be too simplistic.

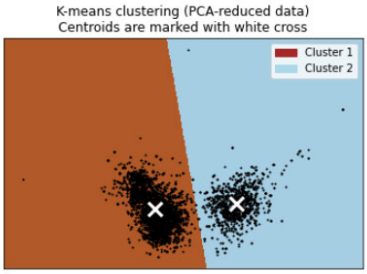

Figure 2: Clustering results of the wine dataset, with only the data points with quality 5 and 6.

In contrast, fig 3 displays clustering results for the mushroom dataset. Well defined separable clusters for specific species and subspecies of mushrooms can be observed. In addition, there is a tangible separation between poisonous and edible mushrooms. This high level separability lends itself well to the classification challenges, and as such it can be hypothesised that the selected machine learning models will produce excellent results on this dataset.

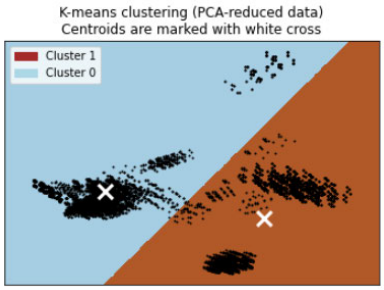

Figure 3: Clustering results for the mushroom dataset, showing clear clusters and separation between labels

While experimenting with the models, the PCA r size could only be set to 2 for both datasets, as both of them could not be reduced by anything larger than 2, due to the small initial number of dimensions. Other parameters like algorithm type and number of clusters (features) did not change the overall spread of data, but did change the metrics. Using the elkan algorithm instead of auto, did reduce the runtime from around 0.24s to 0.04s although with an increase in RAM usage. Also keeping the cluster size to the same number of possible labels (mush=2, wine=7) for either dataset gave better metrics for completeness, compared to different cluster sizes. However, increasing the cluster size above the label number, improved the homogeneity and silhouette metrics. E.g. Mushroom feature = 2, gave H= 0.527, C= 0.527, S= 0.58. While feature = 7, gave H=0.569, C=0.232, S=0.619.

4.2 Decision Trees

Wine

The wine data was tested using 10-fold cross validation to give consistency throughout the models. An initial baseline model was then trained and tested on the test set, yielding 0.870 accuracy, 0.864 precision, 0.869 recall and 0.865 F1-score for the test set.

As the results of the test set fell 10% below the train set, the decision tree was checked for overfitting based on the alphas received from pruning. A graph was created to check for alpha vs train & test to check that the model did not have massively varying train and test results at all levels of alpha, and therefore is not overfitted, which decision trees are usually prone to.

Using entropy as the criterion on the dataset, the variance between the training and test set was reduced and achieved 0.877 accuracy and 0.872 weighted F1-score. However, it may not be worth using the extra processing power to test the model using entropy as it achieved minimal difference in the prediction metrics.

Mushrooms

An initial baseline decision tree model was validated and trained on the train set, achieving 0.999 accuracy, 0.999 precision, 0.999 recall and 0.999 F1-score when tested using the test set.

To verify such a strong result, the nodes were pruned to validate the consistency of the model. Using a ccp_alpha value of 0.005, by removing 4 nodes the train and test results fell by 0.01. At a ccp_alpha value of 0.01, removing 10 nodes, the train and test results fell by 0.02. This suggests that the model is not overfitted and consistent.

Results

For the wine dataset, using the shap library, the resulting tree was examined and the key features picked out by the model were pH, chlorides, citric-acid and sulphates as the main explanatory features for quality. Splitting the dataset by the type of wine, the experiments found that density, fixed-acidity, free-sulfur-dioxide and pH are the main contributors to explaining red wine quality. As for white wine, the results regard pH, chlorides, citric-acid and free-sulfur-dioxide as the main explanatory variables.

As for the mushroom dataset, stalk-surface-above-ring_k (silky), gill-size_n (narrow) and spore-print-color_h (chocolate) were the main explanatory features found by the tree to determine whether the mushroom is poisonous or edible.

The results achieved using xgboost and random forest model were similar when compared to the simple decision tree for both datasets, thus it may not be worth using these more complicated models which come at the cost of significantly increased processing cost and lower explainability and interpretability. Nonetheless, decision trees proved strong at working with multiclass and binary

data, and yielded promising results for both metrics outlined in Section 1.3.

4.3 SVM

The experiment with SVM was implemented using scikit-learn version 0.24.2. One of the parameters was a choice between four kernels: Linear Function, Polynomial Function, Radial Basis Function (RBF) and Sigmoid Function. Kernels are used to convert input data space into higher-dimensional space. All four kernels were used in order to find the optimal choice. The kernel for the RBF function has been set as 'auto' so that it uses a coefficient equal to one over the number of features. All four variations of the SVM using the four different kernel hyperparameters were trained on the train data using the 10 fold cross validation. The accuracy and f score were extracted from these two methods.

For the wine dataset, the SVM variations performed generally well. The Linear SVM achieved a test accuracy of 0.55 and an f1 score of 0.55. The Polynomial SVM achieved a test accuracy of 0.56 and an f1 score of 0.56. The RBF SVM achieved a test accuracy of 0.58 and an f1 score of 0.58. Lastly, the Sigmoid SVM achieved a test accuracy of 0.42 and an f1 score of 0.42. All the variants performed similarly but the best performing model was the RBF by a slight margin.

Significantly better metrics were observed in the mushroom dataset. Both the Linear and Polynomial SVM achieved an accuracy of 1 and an f1 score of 1. The RBF SVM achieved an accuracy of 1 and an f1 score of 0.99. Lastly, the Sigmoid SVM achieved an accuracy of 0.89 and an f1 score of 0.89. These are incredible results however they must be treated with caution as it is rare to construct a perfect classifier. Upon inspection of literature, other literature models also achieve perfect accuracy. This suggests that the dataset lends itself to building optimal classifiers.

In addition to these metrics, ROC curves and confusion matrices were implemented for the models. These can be found in detail in the corresponding notebook.

In summary, the SVM model performed better on the binary classification task. An interesting observation is that the sigmoid kernel has consistently performed worse than other candidate options. Future use cases must avoid the use of the sigmoid kernel in an SVM implementation.

4.4 Artificial Neural Networks

Experiments followed the same methodology for both datasets with each alteration to the model design or input data evaluated using k-fold cross validation before a final candidate model was tested on previously unseen test data held out at the beginning.

Wine

A number of experiments were conducted aimed at reducing the impact of outliers, such as standardisation, binning and the removal of outlying rows from the dataset. Of these, standardisation yielded a marked improvement in the model's predictive ability but no improvement was observed with the other methods.

Hyper-parameters were tuned using a series of grid searches. Due to lengthy model training times, it was infeasible to test all possible variants but marked improvements in model performance were observed by altering the activation function from rectilinear to hyperbolic tangent shape, by increasing regularisation, the number of hidden layers and the number of epochs and by replacing the stochastic gradient descent optimiser with a quasi-Newton method. F1 score was measured at 0.624 versus the 0.547 achieved by the benchmark model with standardised data.

Wine qualities are ordered and categorical so can be treated as either a regression or classification problem. An experiment was conducted where an MLP regressor was used to predict wine qualities in the continuous range 3-9 and the output rounded to the nearest integer. This however did not improve model performance compared with the MLP classifier.

Mushrooms

It transpired that very little experimentation was required to determine an optimal model for the mushroom dataset. A benchmark MLP model with a single layer of nodes was initially tried and found to give perfect predictions on both validation and test data. In order to verify this result and to prove that it did not result from an anomaly in input data or a simple coding error, data quality was first verified and the model was then deliberately handicapped by reducing the number of epochs. With weights insufficiently converged, the model made more classification errors, suggesting that perfect results from the fully trained model could be believed. A single perceptron model was then trained and also found to give perfect predictions, even on the unseen test data, suggesting that mushroom data are in fact linearly separable.

Results

Whilst neural network models performed better on the mushroom dataset, it should be remembered that the target is binary whereas the wine dataset is multi-class, meaning greater scope for classification error. In this context, f1 and accuracy scores of ~0.61 for the final wine model evaluated on test data are respectable. Model training times were significantly higher for the more complex, deeper models required for wine quality prediction with weight convergence occurring in seconds rather than hundredths of a second observed for the

simple, single-perceptron model required for the mushroom dataset. Whilst, results from NN models broadly suffer from a lack of explainability, in the specific case for the mushroom dataset, a relatively simple linear formula for classification was arrived at, making it somewhat more explainable than the wine model, the results from which cannot be expressed in terms that a layman could understand.

4.5 Experiment K Nearest Neighbours

The KNN model was trained using a classic modelling style. The data was split into a 75% train, and 25% test dataset. For each experiment, 10 fold cross validation was used on the train set. Following the estimation, measures of accuracy and f1 score were taken. Hyperparameter tuning was conducted to find the best f1-weighted score at k = 5 for each dataset.

The KNN model was then tested against the mushroom test dataset, yielding perfect f1 and accuracy scores of 1.000, it would appear then that there are very defining factors when determining the mushroom class as anticipated. For KNN the binary input following the one hot encoding from preprocessing is likely to have made the class boundaries very defined.

With respect to the wine dataset, the recorded accuracy was 0.767 and f1 score 0.593 when tested against the test set. Extreme labels of 3 and 9 were very well predicted, with accuracy becoming less and less for the more populous classes. It is observed that for classes 5, 6, and 7, more predictions are incorrect than correct in total, however, the majority of incorrect predictions are close to the correct class so some development with feature selection to inspect the information gain of each attribute could help tune the model and obtain a higher percentage of correct predictions.

4.6 Naive Bayes

Wine Dataset

An initial baseline test using the raw wine data to train each Naive Bayes model produced F1-scores of 0.338, 0.418, and 0.364, and classification scores of 0.381, 0.402, and 0.364 for Multinomial, Gaussian and Bernoulli models respectively using 10-fold cross validation. The first experiment to improve model scores pre-processed the dataset using standardisation, normalisation, and binning.

Standardisation and normalisation were tested as Gaussian and Bernoulli Naïve Bayes models make assumptions on the distribution of the input data, and as such improvement was expected by fitting the data to those distributions. Binning discretises the data which can help models such as Multinomial Naïve Bayes which expect discrete frequencies as data inputs. The best F1-scores for each model obtained by testing each pre-processing method were 0.456, 0.418, and 0.364 with frequency binning for Multinomial and Gaussian Naïve Bayes and raw data for Bernoulli.

For the Multinomial and Gaussian models, the bin size for binning was then optimised systematically by trialling bin sizes between 0 and 20 and obtaining the highest F1 score. This optimisation yielded improved F1-scores of 0.484 and 0.429 using bin sizes of 6 and 17 for the Multinomial and Gaussian models, respectively.

Dimensionality reduction was then performed using feature selection by fitting a chi-squared model to the dataset to systematically determine features in order of importance, and then iterating over every possible input dimension size to find the optimal number of features to use. This final experiment yielded improved f1 scores of 0.512, 0.492 and 0.441, with classification accuracies of 0.487, 0.470 and 0.364 for Multinomial, Gaussian and Bernoulli Naïve Bayes models using 4, 6 and 6 features, respectively.

Mushrooms Dataset

An initial baseline test using the raw mushroom data to train each model produced F1-scores of 0.891, 0.830, and 0.885, with accuracy scores of 0.892, 0.834, and 0.886 for Multinomial, Gaussian and Bernoulli Naïve Bayes models respectively using 10-fold cross validation. Since the mushroom attribute fields are categorical, experimentation with binning and normalisation wasn't necessary. The main test then in this case was the same feature selection method used previously. This experiment yielded improved f1 scores of 0.974, 0.975 and 0.974, with classification accuracies of 0.974, 0.975 and 0.974 for Multinomial, Gaussian and Bernoulli Naïve Bayes models using 6, 46 and 7 features, respectively.

Conclusions

For the wine dataset, the best Naive Bayes model was determined to be the Multinomial Naive Bayes classifier, with highest obtained F1 and accuracy scores. This result can be explained by the fact that the input data, when frequency binned, most closely follows the distributions expected by the Multinomial model which typically requires features as frequency counts. Nonetheless, the overall results for each Naive Bayes classifier were comparatively weak on this dataset when contrasted to more complex models implemented in neural networks and decision trees. This was an expected result, as Naive Bayes models typically struggle with continuous numeric features.

The mushroom dataset saw far better performance from the Naive Bayes models, with Gaussian Naive Bayes yielding the best classifier by F1 and accuracy score. Naive Bayes models typically perform best on categorical data, and as such the result comparative to the wine dataset was to be expected. The classifiers likely fell short

of perfect scores due to the fact that there was some small dependence between features.

4.7 ILP and Relational Learning

In order to extract the rules the original dataset was first converted into a relational dataset using a bespoke python script [see, Datasets-Convert2Relational.ipynb]. This script converts the original dataset into positive examples (poisonous), negative examples (edible), and background knowledge for each row of the original dataset, along with the requisite module loader, modeh, modeb, and determination clauses. When converted into this format, the mushroom dataset now spans over 270,000 lines of clauses from the original 8000 records. This significantly increases the runtime of the ILP model which should normally be very quick, which can even take far longer to run if we had used slightly larger datasets.
After converting the dataset, the induction code in lab 7 which uses ALEPH, was used to extract rules as seen in Table 4. Each rule was tested and validated to be 100% true, for every row in the dataset (gilled Agaricus and Lepiota mushroom families).

Conditions			
Poisonous if	Smooth Cap Surface	Close Gill Spacing	Narrow Gill Size
Poisonous if	No Bruises	Smooth Stalk Surface Below The Ring	Woodland Habitat
Poisonous if	Has Bruises	Urban Habitat	
Poisonous if	Several Population Size	Grassy Habitat	
Poisonous if	Narrow Gill Size	Scattered Population Size	
Poisonous if	Close Gill Spacing	Silky Stalk Surface Above The Ring	
Poisonous if	Narrow Gill Size	Clustered Population Size	
Poisonous if	Foul Odor		
Poisonous if	Green Spore Print Color		
Poisonous if	Buff Gill Color		

Table 4: Table of rules extracted from dataset, using logic induction

Because the rules are 100% true for the entire mushroom dataset, it explains why most of the other models performed so well on this dataset. It proves that there are clear well defined rules that make a mushroom poisonous, which most of the models have picked up on and fitted to with 100% accuracy. Also, due the large number of rules that each cover a large number of poisonous data points, it is safe to assume that the rules cover most, if not all, poisonous mushrooms within the confinements of the dataset. Due to variety of different rules that define all of the poisonous mushrooms in the dataset, there is a high change that these rules will apply to most unseen mushrooms of the gilled Agaricus and Lepiota families, and therefore most of the models will be able to accurately predict unseen data as well.

Due to time constraints, the wine dataset was not adapted for relational learning. Future work to model this dataset would follow a similar procedure as the mushroom dataset, but with additional bucketing of numerical data as Prolog struggles with numerical data. However, due to possible subjective wine quality labels and the loss of specific data through binning, it is more likely that the ILP model will struggle to induce many logical rules, with some even being un-accurate.

5. Discussion of the results, interpretation and critical assessment

Algorithm	Wine		Mushrooms	
Model	**Accuracy**	**F1**	**Accuracy**	**F1**
Decision Tree	0.877	0.872	1.000	1.000
SVM	0.580	0.580	1.000	1.000
Neural Networks	0.612	0.609	1.000	1.000
K-Nearest Neighbour	0.767	0.593	1.000	1.000
Multinomial Naive Bayes	0.531	0.513	0.972	0.972
Gaussian Naive Bayes	0.514	0.497	0.977	0.977
Bernoulli Naive Bayes	0.442	0.350	0.948	0.948

Table 5: Table of results, for all classification models

From Table 5 above it can be observed that the best performing machine learning model for the wine classification task was the decision tree, obtaining the highest F1 and accuracy score on the test set of 0.872 and 0.877 respectively. One explanation for this result is that of the algorithms tested in this report, the decision tree makes the fewest assumptions on the shape of the data and importance of each feature. Models with strict requirements on these inputs such as the Bayesian models fall short on this dataset for this same reason. The second best model according to F1-score, the neural network

model, likely suffered due to a lack of training examples within many of the classes. Neural networks typically require far more training examples than algorithms such as decision trees, and as such suffered in this case. Since our dataset lacked completely examples of wines with certifications of 0, 1, 2, and 10, and only had few examples of wine with 8 and 9 classifications, the evaluation of model performance here is likely to be somewhat unreliable due to the lack of test examples.

For the Mushrooms dataset, all but the Bayesian models were able to obtain perfect accuracy and F1-scores, suggesting that for these particular families of mushrooms, strong relationships between physical characteristics and edibility exist. This is further elucidated by the results of the relational model implementing ILP, which provide shockingly simple rules that can perfectly describe the edibility of the mushrooms in the dataset. An example of this is that 'if a mushroom has a green spore print colour, it is poisonous'. Such results could be easily translated into pamphlets for potential foragers to serve as a quick field guide to these particular mushroom families. This of course hinges on the assumption that the mushroom samples in the dataset are exhaustive for these particular families.

6. Conclusions

This paper examined the effectiveness of different machine learning models when applied to the tasks of the binary-class mushroom edibility task and multi-class wine quality classification task.

Simple decision trees were found to be the most effective machine learning model for solving the wine classification task, with a final test accuracy and F1-score of 0.877 and 0.872 respectively. Examining the decision rules for the tree, the most salient predictors of a wine's quality certification using only physicochemical features were found to be pH, chlorides, citric-acid and sulphate.

The mushroom challenge was solved perfectly by all but the Naive Bayes models, suggesting that strong well-defined relationships between physical mushroom characteristics and their edibility exist. This is illustrated perfectly by the simple rules as output by the inductive learning program which can perfectly classify mushroom edibility using three or fewer physical characteristics.

Due to time constraints, the individual experiments carried out by each team member were limited by necessity. More time spent on the modelling tasks testing a greater number of hyper-parameter settings and model families would offer potentially great improvements on our best model scores. Reinforcement learning for example was a methodology that was cut from this research paper due to limited resources and time, and would be potentially beneficial to implement at a later time.

Future work may also entail ensembling, a method which may have improved wine classification performance. The challenge here would be in constructing a voting system for a multi-class problem.

Additionally, limitations of training examples for sparse classes in the wine dataset hampered our final models, and as such collecting more examples of these poorly represented classes would likely provide the best increase in performance for this task.

Contributions

Everyone - Dataset acquisition, project plan, proofreading and editing, streamlining code, and writeup for discussions of results and conclusion. Further individual contribution covered below.

Ryan B[redacted] ([redacted]) - Contributed to Pre-Processing, clustering model and report, relational dataset conversion, and logic/relational model and report.

Charlie B[redacted] ([redacted]) - Contributed to data pre-processing, KNN modelling, abstract and data preparation writeup.

Adlan E[redacted] ([redacted]) - Contributed to data exploration and visualisations, Decision Tree modelling, evaluation and report.

Chris E[redacted] ([redacted]) - Data pre-processing, Naive Bayes models, relational dataset optimisations, feature selection/dimensionality reduction, report writing.

Chris J[redacted] ([redacted]) - Data exploration and visualisations, pre-processing, shared binning functions, neural network modelling, evaluation and report.

Brian N[redacted] [redacted])- Contributed to data-preprocessing, data visualisation, SVM modelling & evaluation and report writing.

References

[1] Kaggle.com. 2021. Wine Quality. [online] Available at: <https://www kaggle.com/rajyellow46/wine-quality>

[2] P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis.

Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009

[3] Kaggle.com. 2021. Mushroom Classification. [online] Available at: <https://www.kaggle.com/uciml/mushroom-classification >

[4] J. Schlimmer (1987). UCI Machine Learning Repository [https://archive.ics.uci.edu/ml/datasets/mushroom]. Irvine, CA: University of California, School of Information and Computer Science.