Wine Quality Prediction Models

Introduction to Machine Learning and Knowledge Extraction

Master's in Data Science and Engineering

Rojan Aslani (up202204382)

Farzam Salimi (up201007922)

December 2022

1 Introduction

Knowing the quality of wine can be of interest in many fields including, wine producers, distributers, sellers, and the consumers. This important factor can define the price of the wine and its target market, and therefore being of high interest. Traditionally, to determine the quality of a wine sample, wine tasters were hired to taste the wine and assess it. To this day this method continues to be used, because no good enough substitution has been proposed yet.

Wine tasting and quality evaluation is often a very subjective matter. There can be many cases that wine tasters evaluate the same wine differently based on their palette. Hence, Automating the qualification of wine can be of great interest to many industries, because of decrease in costs, and faster and more consistent results. On the other hand, convincing the public about the robustness of the results of this qualification can be a hard task. Wine tasting at its core is a very traditional activity and public might not be prepared to accept that machines can do the assessment individually.

The availability of a model able to assign labels (in this case, the quality of wine) to new unlabeled objects, given the values of its predictive attributes (physiochemical tests) can offer several advantages for the shareholders, such as:

- reduce costs
- increase profits
- improve product and service quality
- improve customer satisfaction

1.1 State of the art

In the previous years several works have focused on building models that can predict wine quality. One of the most robust works that captured the authors' attention is [1], which uses the same dataset used in this work. After a carefully analysis of their work, following observations were made:

- even though the data is imbalanced, this detail was not addressed in any way
- duplicated instances were not removed, and the designed model is possibly overfit and biased – probable inaccuracy in the estimation of accuracy of the model.

In other works published in [2] many authors used a test set that is possibly biased. Test sets were not isolated from the training set and in the pre-processing stage, undergoing several operations that can alter them in comparison to real data. The lack of full separation between the test set and the train set can, as well, lead to a biased result and inaccurate estimation of the accuracy. In this project the mentioned issues were addressed with attention to avoid biased accuracy results.

1.1.1 Data mining success criteria

The wine business, as we mentioned before is a very subjective area, and defining a business success criteria related to it consequently will be a subjective one. The goal will be to achieve the highest accuracy related to the available data. This accuracy can be achieved by testing different machine learning techniques and strategies. The main concentration in this project will be on the deep imbalanced analysis and try to balance the data for training and estimate and compare the results with other studies.

The main question addressed in the project is *how can different machine learning techniques predict the wine quality*? Also, based on the other studies on the same dataset, we want to see the *effects of balancing the data* and figuring out the *main reasons for the possible differences*.

1.2 Project plan

As mentioned earlier, we aim to build a predictive model that can predict the quality of wine with high accuracy, possibly replacing the traditional qualification method. Several models were implemented to find the one with the highest accuracy, ranging from distance based method to ensemble methods.

Throughout this work the cross-industry process for data mining (CRISP-DM) methodology was used to structure the analysis. CRISP-DM is a robust and well-proven methodology that provides a structured approach to planning a data mining project. This model is an idealized sequence of events, which are ordered in the following manner:

1. Business Understanding (Section 1)
2. Data Understanding (Section 2.1)
3. Data Preparation (Section 2.2)
4. Modeling (Section 2.3)
5. Evaluation (Section 3)
6. Deployment (Not applicable to this work)

2 Materials and Methods

This study will consider vinho verde (in this project will be referred to as wine), a unique product from the Minho (northwest) region of Portugal. The data was acquired from UCI[1] website [3]. The data represents qualification of wine samples that were collected by [1] from May 2004 to February 2007 from origin samples that were tested at the official certification entity. The dataset includes 2 files, one for white and one for red one. In this work, we used the red wine in the data exploratory section to avoid the repetitive analysis of both datasets. In addition, we used the both datasets to present different models created by python (red wine) and Rapid Miner (white wine). The quality of wine was graded by a the median value of at least three sensory assessors and is classified on a scale of 0 (very bad) to 10 (excellent).

The data analysis and development of models can be done using several resources (software and programming languages tools). In this study python programming language and RapidMiner were used.

2.1 Red Wine Dataset Exploratory Data Analysis

As justified earlier, only the CSV file containing data about red wine was used in this part to present the data exploratory section. This file contains 1599 records in a tabular format. Each record represents the information about a distinct wine sample. There is a total of 12 attributes, 11 of which contain predictive values regarding physicochemical tests of different wine samples, followed by a "Quality" attribute, considered the target attribute. The target is a non-binary, qualitative, ordinal attribute, with 11 classes ranging from 0 to 10 (multi-label). This attribute is ordered but imbalanced (there are many more normal values than excellent and poor ones).

The data quality of the data was assessed using statistics and visualization techniques. Some statistics of the attributes are presented in Table 1. A quick analysis of the data in Table 1 shows that:

- This is a supervised classification task
  - all the transactions are labeled
  - target attribute is an ordinal value
- the data has **no missing values**
- the attributes have different ranges and almost all of them have extreme values

Table 1. Statistics and information about red wine dataset.

Attribute	*Scale type*	*Data type*	*Mean ± stdev*	*Min, Max*	*N/A*
Fixed acidity (g(tartaric acid)/dm³)	Quantitative	Float	8.32 ± 1.74	(4.60, 15.90)	0
Volatile acidity (g(acetic acid)/dm³)	Quantitative	Float	0.53 ± 0.18	(0.12, 1.58)	0
Citric acid (g/dm3)	Quantitative	Float	0.27 ± 0.19	(0.00, 1.00)	0
Residual sugar (g/dm3)	Quantitative	Float	2.54 ± 1.4	(0.90, 15.50)	0
Chlorides (g(sodium chloride)/dm3)	Quantitative	Float	0.09 ± 0.05	(0.01, 0.61)	0
Free sulfur dioxide (mg/dm3)	Quantitative	Float	15.87 ± 10.46	(1.00, 72.00)	0
Total sulfur dioxide (mg/dm3)	Quantitative	Float	46.47 ± 32.90	(6.00, 289.00)	0
Density (g/cm3)	Quantitative	Float	1.00 ± 0.002	(0.99, 1.00)	0
pH	Quantitative	Float	3.31 ± 0.15	(2.74, 4.01)	0
Sulphates (g(potassium sulphate)/dm3)	Quantitative	Float	0.66 ± 0.17	(0.33, 2.00)	0
Alcohol (vol.%)	Quantitative	Float	10.42 ± 1.06	(8.40, 14.9)	0
Quality	Ordinal	Integer	5.64 ± 0.81	(3.00, 8.00)	0

[1] https://archive.ics.uci.edu

2.2 Data Exploration and Pre-processing

In this section we concentrate on the exploring the data to evaluate whether changes are needed to adapt the data to the aim of the project (developing a prediction model). This includes a process of cleaning, transforming, and organizing the data. The important point is when these data are used by machine learning algorithms the analysis problem can look more complex than it really is if there is no data pre-processing. This increases the time required for the induction of assumptions or models and resulting in models that do not capture the true patterns present in the data set [4].

Regarding the quality of the data, the data seems to have been prepared carefully. It is consistent, interpretable, timeliness, and seems to be accurate. Regarding the completeness, knowing that the scale of quality was set between 0 and 10 (11 classes), there is a lack of data in the limits of the range, having no wine samples qualified as 0, 1, 2, 9, or 10, in the red wine dataset. Unfortunately, this information cannot be accurately generated by prediction methods. Hence, we were forced to consider the range of quality from 3 to 8 (6 classes). Moreover, the red wine dataset has 240 **duplicate** values, which were removed to avoid overfitting of the model (the same process has been done for the white wine as well).

Univariate Analysis

Using visualization and measures of skewness, it was determined that some of the attributes are highly skewed. To transform these variables to normal distribution, log function was used. The before and after results were visually assessed and are present in the annexed file (*wine_prediction.ipynb*).

The Quality attribute follows a normal distribution, having different number of objects in each class. The extremity classes 3 and 8, have 10 and 17 instances each, respectively. On the other hand, the frequent classes, 5 and 6, have 577 and 535 instances, respectively. This implies a clearly imbalanced distribution of the target attribute, as demonstrated in Figure 1 (a).

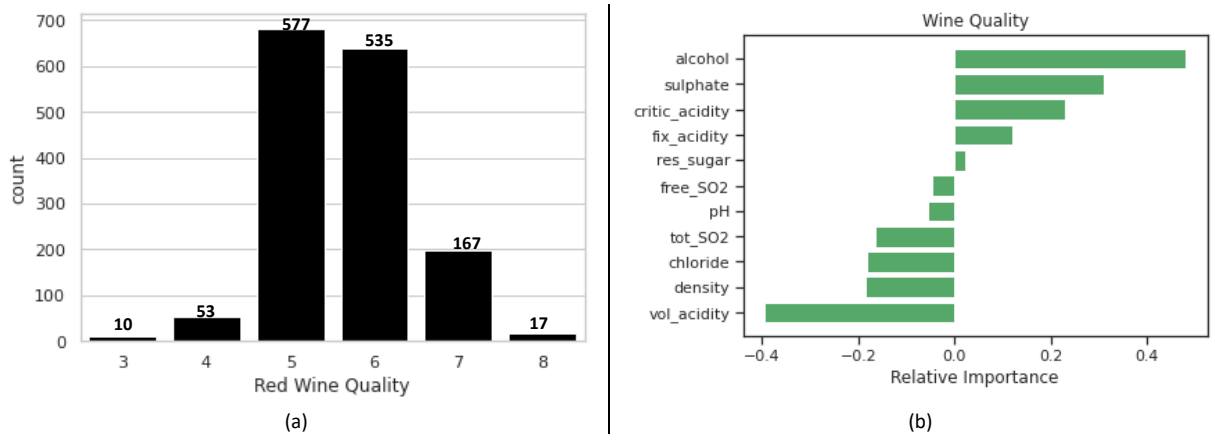

Figure 1. Data visualization of quality attribute of red wine dataset. (a) Histogram of frequencies, denoting a typical normal shape distribution. (b) Correlation values of all attributes with wine quality.

Normalization

Moreover, the attributes were normalized using z-score method to make training less sensitive to the scale of the features. (subtracting the data by the average value and dividing it by the standard deviation). In the normalized data, the more negative a value is, the lower than the average it is. The more positive a value is, the higher than the average it is.

Correlations – multivariate Analysis

Relationships between pairs of attributes was assessed. It was concluded that there is a low **correlation** between all attributes and the Quality attribute, the highest correlation being +0.48 (Figure 1 (b)). Also, various attempts of dimension reduction by feature selection (filter method) was done, all of which decreased the accuracy of the model drastically. Hence, it was decided to move forward with using all the attributes in the prediction model.

Feature Selection

The predictive performance of classification algorithms is mainly affected by the predictive attributes in a data set. Each predictive attribute describes a specific characteristic of a data set. Usually, the more predictive attributes we have for a data set, the better is our description of its main aspects. In this project, the 11 predictive attributes all seem to be relevant and consistent, and no pair of predictive attributes have higher than 0.75 correlation, meaning there is no redundancy (correlations evaluated using filter method). Hence, no measures regarding **feature selection** were taken in this step.

Meanwhile, it is desirable to reduce the number of input variables to both reduce the computational cost of modeling and, in some cases, to improve the performance of the model. Hence, in the modeling phase, some of the methods remove the correlated attributes using **backward wrapper**.

No alteration of data type was necessary, and no columns (predictive attributes) were dropped at this phase. Also, no derived nor aggregated attributes were produced. All rows and columns were used, besides the rows that were duplicated data. The process of feature selection for the white wine dataset is quite different and has been done in the software (Rapid Miner), and it explained in its section.

Outliers

The **outliers** in this dataset are considered of high value because they represent the extreme classes (very good or very bad) wine. These values are legitimate but extreme, bringing high importance to the prediction model, hence they are not going to be removed.

Imbalanced Data

The biggest constrain of this project is that the data is very **imbalanced**, having few really bad wines, and few excellent wines. These extreme classifications can be considered outliers but cannot be removed/ignored. The two most common strategies to deal with imbalanced data are under sampling and oversampling. Both strategies have the goal of approximating the number of objects in all classes. Using

under sampling would imply a big loss of important information from the majority classes, hence, **oversampling** was done to increase the number of objects in the minority classes by duplicating some objects or by creating new objects according to the existing ones. To do so, the synthetic minority oversampling technique (**SMOTE**) was used. This technique selects *k* neighbors in relation to each minority object. Then, for a given percentage *p* of oversampling chosen, around *p/100* from the *k* nearest neighbors are chosen. An object is synthetically generated by choosing randomly for each attribute a value between the attribute value of the object under consideration from the minority class and the attribute value of its neighbor [4]. As we will mention later on, one of reasons that red wine dataset may give a better result over the white wine dataset is the fact that samples that are qualified in extremities had so little objects.

Label Encoding

In addition to that, **label encoding** was applied to the quality attribute to convert its values from natural numerical system to one-hot encoded. This was done because some of the machine learning methods require an array input for the target attribute.

Subsets

Lastly, the data was **split** into **training** (80% of data) and **test** (20% of data) datasets for the red wine and 60% to 40% for the white wine. The training subset is used to fit the model and it went through all the mentioned preprocessing steps. The test subset, which will be used to compute estimate accuracy of the model, was carefully handled to make sure that the data is the most similar to read data as possible. The test dataset did not undergo oversampling.

To see details about the functions and packages used to execute each of the steps in python and rapid miner, see Appendix I.

2.3 Modeling

As mentioned earlier, this project consists of a supervised classification prediction task of a multivariate dataset. Classification task is a predictive task where the label to be assigned to a new, unlabeled, object, given the value of its predictive attributes, is a qualitative value representing a class or category [4]. Several Machine Learning algorithms were developed both in Python and RapidMiner to achieve the objectives. The details of these two approaches are explored in this section.

2.3.1 Machine Learning with Python

After pre-processing the data, assuming the data is now normally distributed, standardized, and balanced, different machine learning techniques were implemented to generate different models. The tested algorithms are explained in detail below.

Binary Classification

This project was mostly focused on multi-class modeling, however, initially the performance of models was assessed with binary classification. The quality class was divided into two classes, one from 0 to 5 and another from 6 to 10. In the binary approach having several predictive attributes, a hyperplane separates the data into two categories (good and bad quality) – more complex decision borders were created and more complex classification model was induced. When the decision border becomes more complex, it also becomes more difficult to induce a model that can find the border using simple classification techniques. Algorithms with clever heuristics are necessary to induce functions that will find complex borders. However, such algorithms typically tend to overfit, taking into account unnecessary details present in the data set.

Multilabel Classification

As mentioned earlier several models were generated by distance based, probability based, and search based algorithms using the wine dataset, namely:

- **K nearest neighbors (kNN)**: discriminative algorithm - the number of neighbors (k) was tuned manually. The Euclidean distance was the distance measure.
- **Logistic Regression**
- **Naïve Bayes**: Generative algorithm
- **Artificial Neural Networks (ANN)**: hyper parameter tuning was done using an algorithm that tests several pre-defined hyperparameter values to create models, and finally returns the model with the highest accuracy (see *wine_prediction.ipynb* for more information). The hyperparameters taken into consideration are number of nodes, dropout probability, learning rate, batch size, and the number of epochs. ANN model was developed with various activation functions (relu, selu, elu, tanh – For more information visit Appendix II).
- **Support Vector Machines (SVM)**: SVM model was made with both one vs one (OvO) and one vs all (OvA) classifier and accuracy was evaluated in both models. OvO and OvA are decomposition strategies that are commonly used for multiclass data.
- **Adaptive boosting (Adaboost):** Hyper parameters of Adaboost were manually tuned.

Regarding feature selection, it was confirmed with RapidMiner that Feature selection (backward wrapper) does not change the accuracy of the model. Moreover, in many cases in RapidMiner in feature selection step no features were removed.

A decomposition strategy decomposes the original multi-class classification task into several binary classification tasks, for which any binary classification algorithm can be used. The binary classification outputs are then combined to obtain a multi-class classification. The two main decomposition strategies are OvO and OvA. As mentioned earlier, both OvO and OvA strategies were implemented in SMV models.

2.3.2 Machine Learning with RapidMiner

One of the software that were used in this project is the Rapid Miner. The benefit of this software to the python is that it is more user friendly than the python, and it provides all the necessary tools to build a complete predictive model with different methods fast. For this project, we built several different models (more than 100) with different methods and different hyper-parameters to figure out what can be the better combination of the hyper-parameters in related to the selected method and our dataset.

One of the abilities of the Rapid miner is in the auto-model, which automatically build different types of models based on the dataset. However, these models are created without any human supervision. Therefore, sometimes they can be unreliable. To use those types of models in this project we separated each of those automatically created processes separately, and it gave us the opportunity to dive deep into the created model and verify all the details of the process and hyper-parameters selections.

So, one of the activities that we did related to the Rapid Miner was to analyze each of the created models separately and create our own models based on them. For example, as mentioned earlier in the report, one of the important details of the wine quality dataset was the fact that it is extremely imbalanced and oversampling is necessary. Hence, SMOTE method was applied to understand how the model generated used imbalanced data is different from one generated using balanced data, in terms of accuracy. We added this feature and as well changed the hyper-parameters related to the different models and estimated the results separately. The time-frame of running these models as well was different from one to another. Our different runs could take from few minutes to even one hour for each method.

Modeling was done on the white wine dataset with different techniques that are listed below:

- Decision Tree
- Deep learning
- Fast Large Margin
- Generalized linear model
- Gradient boosted tree (GBT)
- Logistic regression
- Naïve Bayes
- Random Forest
- SVM

For each of these models both imbalanced and balanced (by SMOTE) models were generated and evaluated, after hyper-parameter tuning.

For the feature selection in the rapid miner we used the feature engineering. In this method, first we performed a primary parameter optimization to make sure that the model is proper to some extent before the feature engineering. Then we validate the model in the feature set which has been done by the feature engineering of the rapid miner.

3 Results and Discussion

The main concern of most data analytic applications is the classification model's predictive performance, which is associated with how frequently the labels are predicted correctly. Several measures can be used to assess the predictive performance of a classification model.

Being a supervised classification task, the evaluation of the performance of the models, validation is commonly used. Validation can estimate the generalization capability of a model, using test set. As explained earlier, the training set is used to train the models and test set is used to validate the accuracy of the model. The trained model will be facing the test data for the first time, and the overall accuracy (average of accuracy of prediction for all classes) is then estimated according to the prediction capability of the model. One way to visually evaluate this comparison, is to use confusion matrix, which allows the visual verification of accuracy for each of the classes.

The confusion matrix is often used for classification analysis, where a C×C matrix (C is the number of classes) is created by matching the predicted values in columns, with the desired (previously known) classes in rows.

3.1 Results of Models Created in Python

In each step of pre-processing the accuracy of the models was tested. Many steps in the pre-processing did indeed decrease the accuracy of the model. Initially, before removing duplicates, standardizing, and balancing the data, the accuracy levels were at about 80%. The decrease in accuracy in the case of this project does not mean that the quality of the models are less. Indeed, the quality of the models are higher because the are unbiased and can better predict the minority classes.

The estimated average accuracy of the designed models are presented in Table 2. The confusion matrices for these models are presented in Figure 2. These results are the final results after hyperparameter tuning and pre-processing of the data.

Table 2. Accuracy of different models of machine learning using red wine dataset and different methods (model was trained with 80% of the data and 20% was used for validation and calculation of accuracy) – using python.

Modeling Technique	**Accuracy**
K-nn (n =2)	43%
Multiclass Logistic Regression	42%
Naïve Bayes	35%
Neural network – 4 layers of relu	**49%**
Neural network – 4 layers of tanh	43%
Neural network – 4 layers of selu	**50%**
Neural network – 4 layers of elu	47%
Neural network – 2 layers of relu and 2 layers of selu	44%
SVM	46%
Adaboost	**49%**

Even though the highest value of accuracy in the model is about 50%, this result might be considered a good result considering the extremely imbalanced data we are dealing with. Earlier in the project models were created with less pre-preparation steps and they seemed to show a higher level of accuracy. But those values were biased and unreal. For example, if we do not apply SMOTE method and use imbalanced data to train the model, we obtained much higher accuracy (about 80%) due to overfitting and giving a high weight to majority classes. This is a biased model because in many cases, introducing a transaction that is for example graded 9, if the model guesses it belongs to level 6, the accuracy will not be lowered because there are very few instances of quality graded as 9. Hence, the accuracy stays high, but we will miss all the excellent or very bad wines.

As demonstrated in Table 2, the accuracy of the ANN model that is built using 4 layers of *Selu* is highest between all the tested techniques. The confusion matrix of this model is presented in Figure 2 (d). The best result would be the one that the highest values are present in the main diagonal, meaning the label was classified correctly. As we can see in the Figure 2 (a) to (d), none of the models was able to correctly predict the class of wines that were in the minority classes. On the other hand, the values are normally close to the real one, having zeros in the top right and low left corners, which is a good sign. Even though the performances are quite similar to eachother, ANN with 4 layers of relu, ANN with 4 layers of selu, and Adaboost have the most promising results.

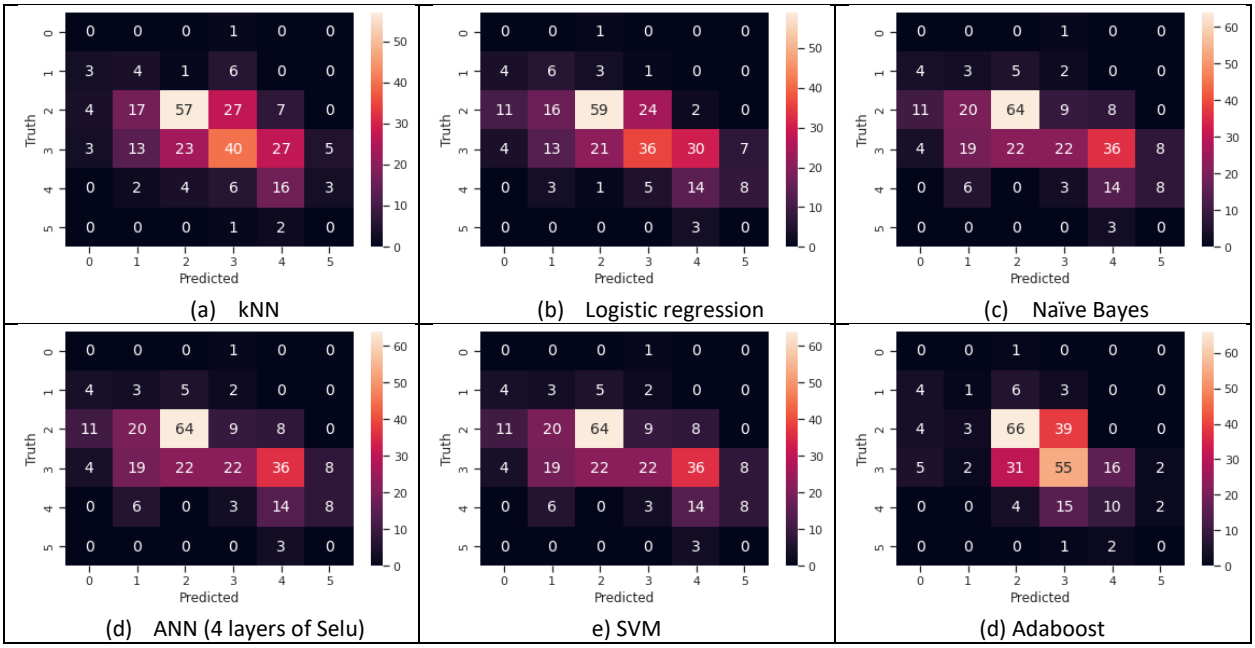

Figure 2. Confusion matrices of accuracy of models developed in python. The axis values represent classes of quality of wine, where 0 represents 3, 1 represents 4, 2 represents 5, and so on.

Comparing the 3 best models, we can conclude that AdaBoost is advantageous because it has no hyperparameter tuning, which is a huge advantage in comparison to ANN, which has many hyperparameters. Both ANN and AdaBoost are computationally expensive and hard to interpret.

3.2 Results of Models Created in RapidMiner

The result of the applying 9 different methods on the dataset with and without balancing the data is shown in Table 3. This table shows the best output based on the accuracy of each method by testing different hyper-parameters, as well as the features automatically selected by each method, using backward wrapper.

Looking at the selected features by each algorithm, it is possible to make some general conclusions. For instance, alcohol has been chosen in most of the methods (15 out of 18), so clearly is an important factor in wine quality. In contrary, chloride has been chosen in only 5 models.

Another interesting finding from comparing these methods is related to balanced and imbalanced data results. In most of the cases the models with the balanced data used more features than the imbalanced ones. For instance, in the deep learning model, the imbalanced model just used 4 features to reach to the best accuracy, but the model with the balanced data uses 7 features.

Table 3. Results of applying 9 different methods on the white wine dataset in RapidMiner, showing the feature selection and model accuracy. Accuracy values of the data before and after balancing using SMOTE is also previewed.

	Selected Features											
	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	**Model accuracy**
Decision Tree			✓									44.9%
Decision Tree (balanced)											✓	43.11%
Deep Learning		✓			✓			✓			✓	50.5%
Deep Learning (balanced)	✓	✓	✓			✓		✓	✓		✓	50.75%
Fast Large Margin		✓	✓					✓			✓	49.9%
Fast Large Margin (balanced)		✓	✓	✓			✓			✓	✓	43.33%
Generalized Linear Model		✓	✓	✓			✓			✓	✓	46.3%
Generalized Linear Model (balanced)	✓	✓		✓	✓	✓		✓	✓	✓	✓	45.84%
Gradient Boosted Trees	✓	✓		✓		✓	✓			✓	✓	53.0%
Gradient Boosted Trees (balanced)	✓			✓			✓	✓	✓		✓	52.38%
Logistic Regression		✓	✓	✓						✓	✓	50.8%
Logistic Regression (balanced)	✓	✓			✓				✓		✓	47.61%
Naive Bayes		✓									✓	50.4%
Naive Bayes (balanced)											✓	46.95%
Random Forest		✓		✓	✓						✓	43.8%
Random Forest (balanced)	✓							✓	✓			43.19%
SVM	✓	✓	✓	✓	✓		✓		✓	✓		45.5%
SVM (balanced)	✓			✓							✓	48.68%

Also, regarding the model overall accuracy we cannot find a trend to ensure the balanced models are working better in this table. In different models with different algorithm, the effects of balancing the data on overall accuracy is different. For example, in the Fast Large Margin method the accuracy decreases dramatically by applying the SMOTE for balancing the data, however, for a method like SVM this effect is reverse.

Overall, with RapidMiner the best overall accuracy is achieved by the Gradient Boosted Trees (GBT) with 53% accuracy. The selected features for this model are alcohol, sulphates, fixed acidity, volatile acidity, residual suger, free sulfur dioxide, and total sulfur dioxide. However, the main goal of these comparisons was to have the better understanding of each model and its hyper-parameters and effects of balancing the data.

Table 4 is the confusion matrix of the classes prediction accuracy. This table can give a good understanding of how the prediction of each class is performed by each model, and specifically how did each model worked on the predicting the classes with the limited number of instances.

Table 4. Class prediction accuracy of each model for each class with the models created in RapidMiner.

	Class Precision						
	3	4	5	6	7	8	9
Decision Tree	0.00%	0.00%	0.00%	45.16%	0.00%	0.00%	0.00%
Decision Tree (balanced)	0.00%	0.00%	0.00%	45.19%	46.38%	0.00%	0.00%
Deep Learning	0.00%	24.14%	**57.72%**	49.87%	45.05%	0.00%	0.00%
Deep Learning (balanced)	0.00%	33.33%	**55.32**	**51.25%**	41.18%	0.00%	0.00%
Fast Large Margin	0.00%	**50.00%**	**52.76%**	**50.47%**	38.89%	0.00%	0.00%
Fast Large Margin (balanced)	0.00%	**50.00%**	50.88%	49.52%	45.45%	0.00%	0.66%
Generalized Linear Model	0.00%	10.00%	47.97%	49.25%	31.46%	20.00%	0.00%
Generalized Linear Model (balanced)	0.00%	40.00%	43.12%	**51.83%**	38.24%	0.00%	7.69%
Gradient Boosted Trees	0.00%	**100.0%**	**55.19%**	**51.49%**	**53.97%**	0.00%	0.00%
Gradient Boosted Trees (balanced)	0.00%	100.0%	**55.89%**	**50.78%**	**62.07%**	0.00%	0.00%
Logistic Regression	0.00%	40.00%	**55.46%**	**50.38%**	41.32%	0.00%	0.00%
Logistic Regression (balanced)	0.00%	**50.00%**	**52.80%**	**50.75%**	35.96%	0.00%	1.64%
Naive Bayes	0.00%	30.43%	**56.46%**	**50.91%**	35.00%	0.00%	0.00%
Naive Bayes (balanced)	0.00%	0.00%	**53.21%**	49.09%	50.00%	0.00%	0.00%
Random Forest	0.00%	40.00%	42.11%	46.03%	31.50%	0.00%	0.00%
Random Forest (balanced)	0.00%	0.00%	0.00%	45.31%	0.00%	0.00%	1.92%
SVM	0.00%	0.00%	44.90%	45.56%	0.00%	0.00%	0.00%
SVM (balanced)	0.00%	0.00%	**53.15%**	49.73%	47.83%	0.00%	2.86%

Several interesting observations can be made from Table 4. The first obvious one is that none of the models was able to predict instances of quality class 3, and very few were able to predict class 8, similar to the results obtained in python models. This might be due to the fact that class 3 and 8 have very few instances (both in red and white wine datasets). The low number of instances limits the training of the model and applying SMOTE to increase the number of instances possibly overfits the model.

Another reason might be can be due to the fact that it is our lowest class, it means that there could be many cases that the wine tasters were tasting two separate wines with similar physiochemical tests and they were doubtful that if this is a wine of class 3 or 4 for example, consequently affecting the prediction of the models overall.

Also, class 6 is the only class which has a good amount of correct predictions between all the models. This class has the most amount of instances in our dataset and this fact can be a good reason for it. It seems that the number of the instances in the real dataset has the great impact on the prediction of the classes. This make sense, since the classes with the greater number of instances had a bigger share in the test set and consequently they had the more chance to be predicted correctly. There are many other interesting observation can be made from Table 4 such as the fact that random forest has a much higher accuracy with the imbalanced dataset than the balanced one, which is possibly a biased result.

In comparison with other studies, the accuracy of the designed models in this project are lower than the ones in other studies, since we totally isolated the test set from training set. However, comparing to the regression based studies the result of the accuracies are near each other [2] [1].

4 Conclusions and Future works

The objective of this project was to develop a model that can predict, with high accuracy, the classification of a wine sample, according to its physiochemical tests. To do so, a deep analysis of machine learning models was carried on to find the optimal prediction model.

Some of the important tasks carried out in the preprocessing phase of the project include normalization, balancing using SMOTE, standardization, and in the machine learning part, applying feature selection, OvO and OvA decomposition strategies, tuning algorithms, and others were used.

The models were developed using two datasets of white and red wine. Red wine models were developed in python and white wine models were developed in RapidMiner. The highest accuracy in the red wine is the neural network with 4 layers of *selu*, predicting with an accuracy of 50%. This model can predict the majority class with an accuracy of 64%, but the minority classes have a much lower accuracy.

The models created using white wine dataset, were summarized and reported in two tables. In this dataset the gradient boosted tree had the best accuracy of about 53%.

To understand the effect of imbalanced data on prediction models, models with balanced and imbalanced dataset were developed and compared. In some methods the overall accuracy improved and in some cases it decreased with the addition of a SMOTE method. This is not a shocking fact, as the high imbalanced data makes it hard to achieve high values of accuracy, because of lack of data in the extreme classes (excellent wine or very poor wine). To overcome this issue, the use of a bigger dataset with higher

number of instances in these classes can improve the performance of the models. The current model was trained using an average of only 15 instances in the extreme labels and thousand instances in the median classes. Even though SMOTE was used to balance the data, the newly created data is almost duplicated and does not add much value to the prediction model.

The objective of this project was to predict the quality of a wine sample from 0 (poor quality) to 10 (excellent quality), according to its physiochemical test. While 0-10 is a common scale, other scales of lower interval can be used to measure the quality of wine. It can be suggested that this scale narrows down for the future works, having fewer classes to classify will increase the accuracy of the data while decreasing the imbalanceness.

It was interesting to be able to sometimes do a quick simulation of a model in RapidMiner, and develop and analyze it in more detail in Python.

In future works it would be interesting to explore the optimal techniques with bigger datasets that contain higher number of instances in the minority classes, as well as the missing classes. Also, for different datasets with bigger amount of instances it would be necessary to revise the hyper-parameters, since these parameters have been set for the current dataset.

It would be interesting to apply this machine learning method in the industry to be able to effectively, accurately and in a fast manner, classify the quality of wine samples. The integration of such solution as a decision support tool can be beneficial to speed the process of entering the market, as well as placing the produced wine in the correct level of wines. The deployment phase would involve generation of a model with higher number of instances and possibly increasing the accuracy of the prediction model.

5 Bibliography

[1] A. C. F. A. T. M. a. J. R. P. Cortez, "Modeling wine preferences by data mining from physiochemical properties.," *Desicion Support Systems,* vol. 47, no. 3, pp. 547-553, 2009.

[2] V. Kumar, "Prediction of quality of Wine," 2017. [Online]. Available: https://www.kaggle.com/code/vishalyo990/prediction-of-quality-of-wine.

[3] P. Cortez, "Wine Quality Data Set," 2009. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Wine+Quality. [Accessed 11 11 2022].

[4] J. Moreira, A. Carvalho and T. Horvath, A General Introduction to Data Analytics, WILEY, 2018.

6 Appendix I

Table 5. Most relevant functions and packages of python used in each step of the project.

Task		Function (Class)	Package
Pre-processing	Duplicate removal	`drop_duplicates()`	`pandas`
	Normalization	`StandardScaler()`	`sklearn.preprocessing`
	Oversampling	`SMOTE()`	`imblearn.over_sampling`
	Encoding	`LabelEncoder()`	`sklearn.preprocessing`
Modeling	K nearest neighbor	`KNeighborsClassifier(n=3)`	`sklearn.neighbors`
	Logistic Regression	`LogisticRegression()`	`sklearn.linear_model`
	Naïve Bayes	`GaussianNB()`	`sklearn.naive_bayes`
	SVM	`SVC()`	`sklearn.svm`
	Neural Network	`Sequential()`	`keras.models`
	Adaboost	`AdaBoostClassifier()`	`sklearn.ensemble`
Evaluation	Report	`classification_report()`	`sklearn.metrics`
	Confusion matrix	`confusion_matrix()`	`sklearn.metrics`

7 Appendix II

ReLU, SELU, ELU, and Tanh are all activation functions used in artificial neural networks. ReLU stands for Rectified Linear Unit, SELU for Scaled Exponential Linear Unit, ELU for Exponential Linear Unit, and Tanh for hyperbolic tangent. These function alongside with some others are presented in Figure 3. Each type of activation function has its own properties that influence how a neural network will learn.

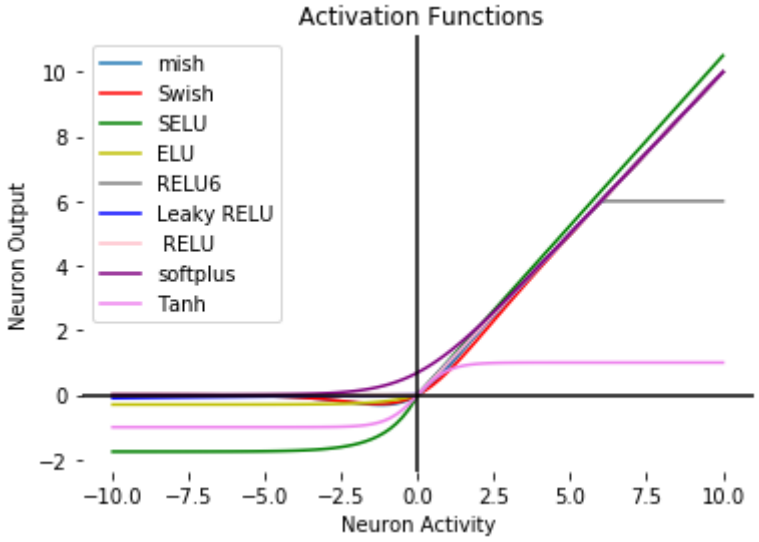

Figure 3. Different activation functions for Deep Neural Networks [5].