Determining Wine Quality with Machine Learning

Author: Misha Ward
[1] University of Washington Bothell, Bothell WA 98043, USA

Abstract: Have you ever wondered if a certain wine is good? There are many people who drink wine but cannot figure out the answer to that question. Determining wine quality can be incredibly difficult without years of experience. Although becoming an expert in wine tasting can take years, recent advancements in artificial intelligence, specifically machine learning, have been developed that can possibly help with this issue. In this paper, multiple machine learning methods and models are used to see if a machine can predict whether a wine is "good" or "bad."

1. Introduction

1.1. *Background*

Wine has been around for millennia and is drank in many cultures around the world. Although there are many different types of wine, the most common types in North America are red and white wines. Although these two types of wines are similar, this paper will focus on red wine for machine learning classification. Historically, wine has been classified on a scale of 1-10, where 1 is the worst quality and 10 is the best quality. Quality has also been left to experts called connoisseurs who gather years of experience by tasting different types of wine. However, with advances in machine learning, using historical data can possibly be used to help predict wine classification between good and bad based on scientific measurements of the wine. The topic of this paper is to use machine learning models and determine if red wines can be classified and which models are the most accurate if any.

As a disclosure, the author of this paper does not drink or know much about wine and has a very limited background in the field.

1.2. *Problem statement*

The problem is to use machine learning to replicate the classification work that wine experts do. Specifically, machine learning should be used to review some or all of the features of the wine dataset and be able to classify the wine. The red wine dataset is publically available and has 12 features with 1599 data objects. An example of the first few rows of the dataset can be seen in Table 1 below.

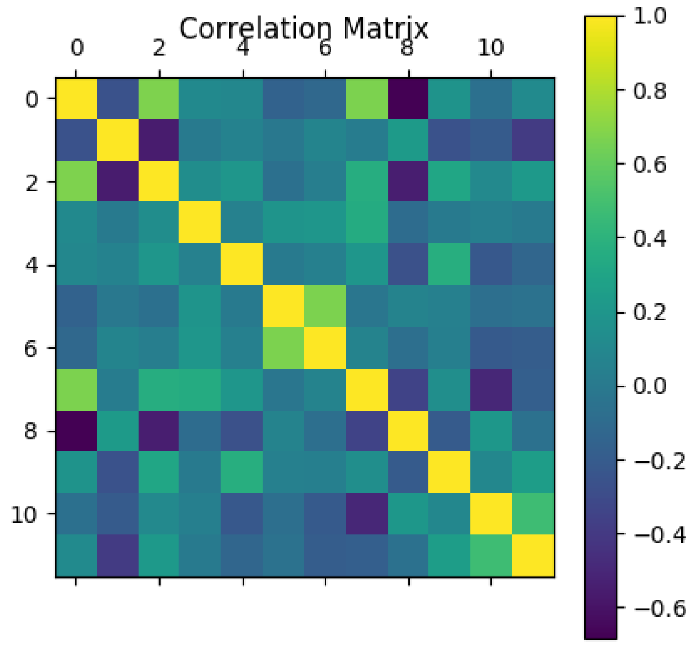
Table 1: Red Wine Dataset Example

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	0
7.8	0.88	0	2.6	0.098	25	67	0.9968	3.2	0.68	9.8	0
7.8	0.76	0.04	2.3	0.092	15	54	0.997	3.26	0.65	9.8	0

From Table 1, the data at first glance has many similar values for each features (column) as all values look to be continuous numbers. However, the most important column, Quality, is the label that will be predicted and is the only feature subjectively created by human input. Originally, quality was ranked from 1, meaning the worst quality, to 10, being the best quality with an interval of one. For the purpose of this research, the rankings were modified to 0 being "bad" and 1 being "good" to simplify the problem and make it easier to classify the results.

After reviewing the basic dataset, it's clear that there will be certain preprocessing that will be needed to ensure that the data is standardized and consistent. Although the data is not missing values or contains strings to parse, the numbers are not standardized. Preprocessing the data allows for all the values in the table to now be scaled in a range from 0 to 1. Once preprocessing has been completed and all features (except the quality label) have been scaled, the correlation matrix was created which can be seen below in Figure 1. The correlation matrix shows how closely the attributes compare to each other statistically. From the graphic below, one can see that many of the attributes do not share much correlation with each other. In fact, the lack of correlation will most likely signal issues with model accuracy as much of the data cannot relate to each other.

Figure 1: Correlation Matrix of the Red Wine Dataset



2. Methodology

2.1. Experiment Methodology

To complete this experiment, many different methods were employed including data visualization, use of various machine learning models, and several measurements of results. Each method is detailed below.

2.1.1. Data Visualization

Data visualization was critical for this project to fully explain complicated data trends and concepts. During this project, graphs like the correlation matrix, 2D and 3D graphics, ROC curve, tables, and other graphics were utilized. By using Python and public Python libraries (Pandas, Numpy, and Sci-Kit Learn), the graphics were able to be produced quickly and consistently. An example of the 2D and 3D charts can be seen below in Figure 2 and Figure 3. Although the original data is in 11 dimensions, by using principle components of the data (PCA function within Sci-kit Learn), the dimensionality of the data can be reduced making it possible to visualize. These two graphs show how the data looks like in lower dimensionality that are easier to understand. Although not perfect, these charts can start to show how clustered the data is together that otherwise would be impossible to understand at 11 dimensions. Additionally, these figures show how the two classes of good and bad start to diverge allowing for machine learning models to predict the classification.

Figure 2: 2D Chart of Red Wine Dataset

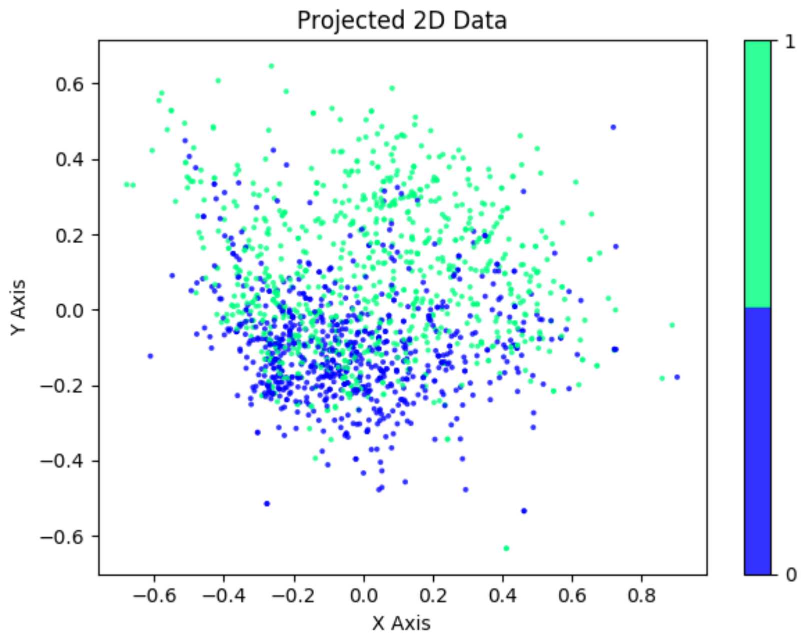
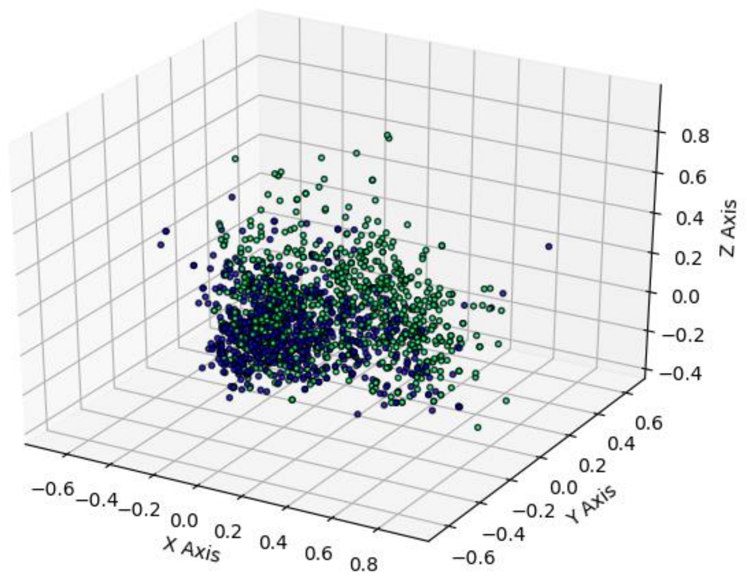


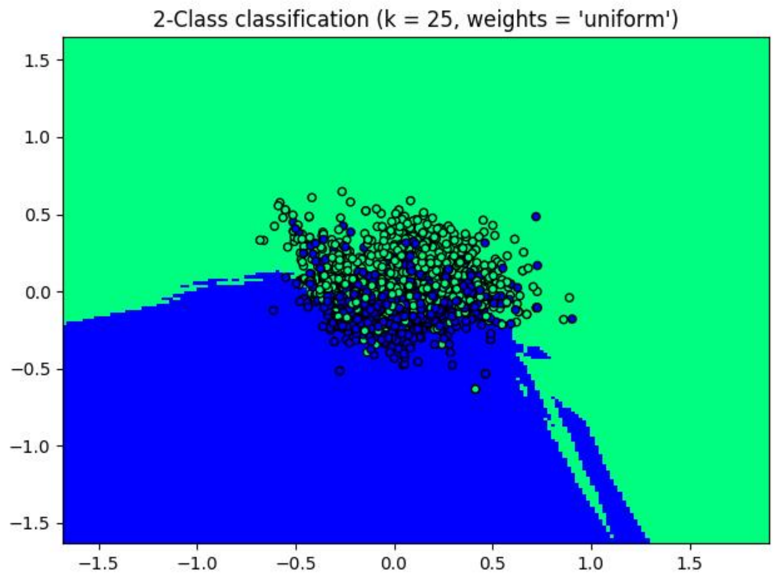
Figure 3: 3D Chart of Red Wine Dataset



2.1.2. Machine Learning Models

Along with extensive use of visual aids, the project used many different types of models to predict the wine quality. The main model that yielded the most interesting results was the K-Nearest Neighbor as it was one of the two most accurate models. Below is a graphic that shows the result of the decision boundaries of the K-Nearest Neighbor in Figure 4. In Figure 4, one can see how the decision boundaries of the model looked like and how the test data plotted on it. As one can see, the decision boundaries look fairly reasonable as much of the green data points ("good") lie above in the green shaded zone while the blue ("bad") are mostly located below.

Figure 4: K-Nearest Neighbor Graphic Showcasing Decision Boundaries



Although K-Nearest Neighbor showed interesting results, the project also utilized 9 other models listed below:

- Linear SVM
- RBF SVM
- Decision Tree
- Random Forest
- Neural Network
- AdaBoost
- Naïve Bayes
- QDA
- Logistic Regression

Even though the project utilized many different models, the additional models did not help provide substantial increases to accuracy or other important result metrics. With the average accuracy of 68.5% of the ten models, the models help solve the problem of qualifying wines based on scientific measurements but not completely. Despite this, by running many different models, one can see how complex this data set is and the difficulty to predict red wine quality.

2.1.3. Measurement of Results

Measurements of results were captured in many different metrics that are standard in statistics and machine learning communities. These measurements include model accuracy, precision, and recall, F1 scores. Each of these result measurements are described below along with the formula:

Accuracy: A measure of how close the average value is to the true value.

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_p + F_n}$$

Precision: A measure of how close estimates are to each other.

$$Precision = \frac{T_p}{T_p + F_p}$$

Recall: The true positive rate, or how many relevant items were selected.

$$Recall = \frac{T_p}{T_p + T_n}$$

F1 Score: Harmonic mean, used for high precision average of the ratios above.

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

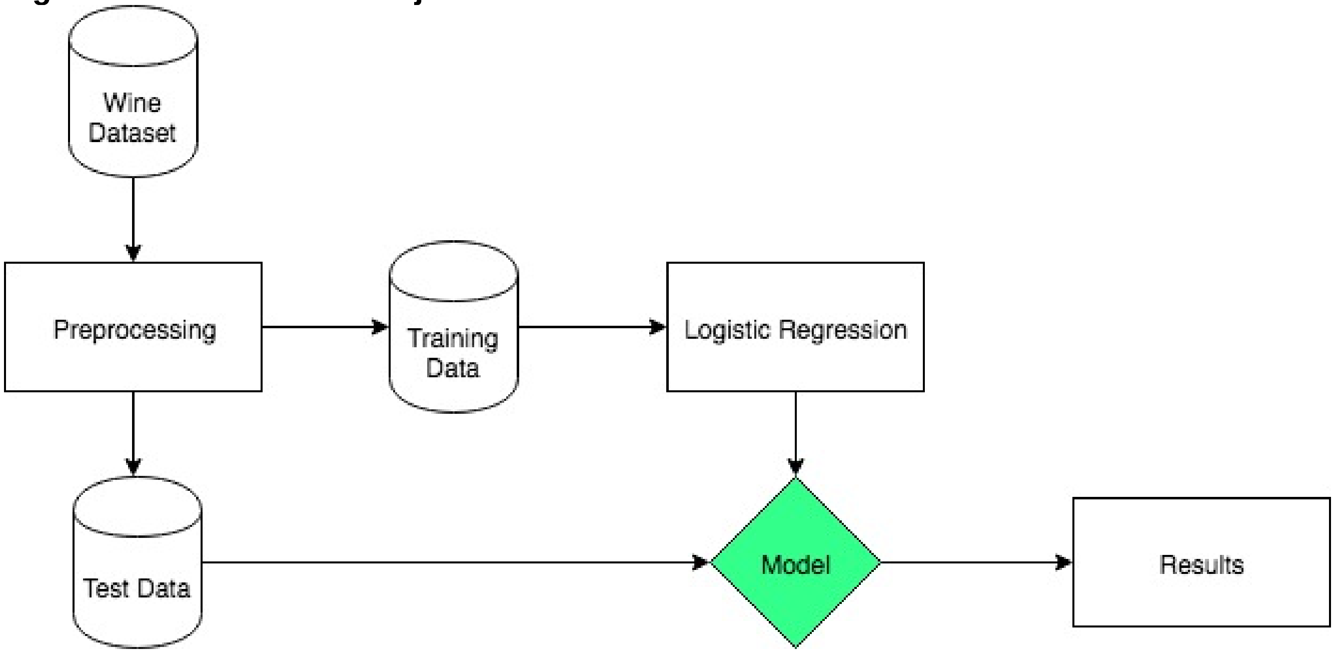
Above is a summary of the calculations for the results. For the purposes of this project, public libraries (Sci-kit Learn) calculated these in the final software package by utilizing these formulas. Final results can be found in Section 3.3 of this paper.

2.2. *Flow Chart*

Many complicated projects usually need to be broken into steps in order for them to be successful and repeatable. Due to the complexity of the project and machine learning in general, this project was also broken into steps. These steps can be seen in the below flow chart in Figure

5. Figure 5 shows how the original wine dataset is preprocessed and split into two datasets, one for training, and another for testing. From this point, the training data would be passed into the model to train while the test data would then be passed into the fully trained model and compared to the actuals resulting in metrics like accuracy.

Figure 5: Flow Chart of Project Data



2.3. *Pseudo Code*

After breaking up the project into smaller pieces and creating a flowchart to help organize the work statement, pseudo code was created to help organize how the software package would be developed. Originally, the main model that the project would focus on was the logistic model due to its simplistic nature and widespread use in two-class classification problems. It was believed that using the logistic regression on the training would result with a very accurate model. Below is example pseudo code for this project. Based on this pseudo code, most of the code was developed and framed around 6 steps including 1) Load Data, 2) Preprocessing, 3) Transformation, 4) Models, 5) Cross Validation/ Evaluation, and finally 6) Results.

Original Project Pseudo Code

```
# 1. Load red wine data.
dataset_link = '/Users/MishaWard/Desktop/winequality-red.csv'
data = pandas.read_csv(dataset_url)

#2. Preprocessing
min_max_scaler = preprocessing.MinMaxScaler()
X_train_minmax = min_max_scaler.fit_transform(data)

#3. Transformation
#TBD

# 4. Logistic Model, example model
clf = linear_model.logistic_regression_path(C=e5)
clf.fit(test2[0], test2[1])
plt.clf()

# 5. Evaluation
# Compare models

# Results
# print(results)
# plt.show()
```

3. Experiment and Result

3.1. Technical Details of Experiments

This project contained many different technical challenges, especially in regards to machine learning. For example, with preprocessing, it was necessary to learn about the dataset and understand if there were issues with the data. This included finding out if there were any missing values or incorrect numbers. Fortunately, the most complicated aspect for preprocessing for this project was scaling the values in order to reduce the weight of one feature over another.

Once preprocessing was completed, finding and utilizing models on the dataset was complicated and took time to understand and program. Documentation sites like http://scikit-learn.org/stable/ helped immensely in this task with determining which models would be useful and how to program them. It was very important during this stage of the project to use proven ways of increasing the accuracy. These methods included reducing dimensionality (feature engineering), multiple models, model tuning, and cross validation.

While some of these techniques did help, including multiple models, some did not seem to impact the accuracy, such as reducing dimensions.

In addition to models, showcasing easily understandable results was also difficult to do. Plotting results and metrics took time and practice to ensure that the graphics were correct and easy to read. Much research went into how to use mathematic and plotting libraries to code the graphics in simplistic terms

It should be noted that multiple experiments were also conducted on the number of categories that would be classified. As previously mentioned, the original dataset was ranked from 1 to 10 and reduced to two categories to simplify the problem. Although two did simplify things, other ranges of categories were tried, including 3 ranks of “bad”, “medium” and “good”. When this ranking system was used, the model preformed almost perfectly but after investigating, this was due to most of the data, roughly 85% fell within the “medium” category and 17% fell within the “good” category. Due to the imbalance in the data, I felt that the two-class classification was more applicable as the two classes were distributed evenly.

3.2. *Related Work*

After extensive research, there were other projects that looked into wine quality classification but the one that was most prominent was a study on “Selection of important features and predicting wine quality using machine learning techniques” in the Procedia Computer Science Journal. This article utilized the same dataset and used the linear regression, neural network, and support vector machine models to classify the wine. In the article, it describes how neural network and SVM was more precise but that the SVM was better at making predictions (Gupta, 312). Unfortunately, the paper fails to mention accuracy explicitly so a direct comparison is impossible to make.

3.3. *Results*

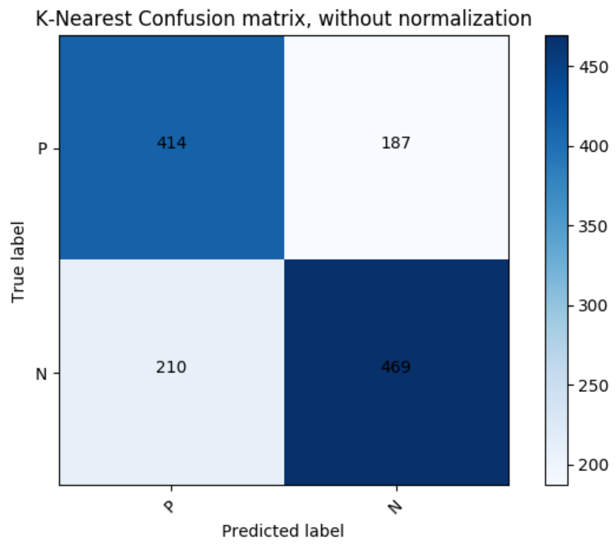
The results of this project were positive as the accuracy reached as high as 70%, yet this could be better. After running ten different machine learning algorithms on the test data, the average model accuracy was 68.5% while the minimum accuracy was 64.9% (Decision Tree) and the maximum accuracy was 70.4% (Logistic Regression).

One of the main models I used to analyze this problem was K-Nearest Neighbors in which ran other result statistics which are seen below:

Measurement	**Result**
Accuracy	68.59%
Precision	71.81%
Recall	67.16%
F1 Score	69.81%

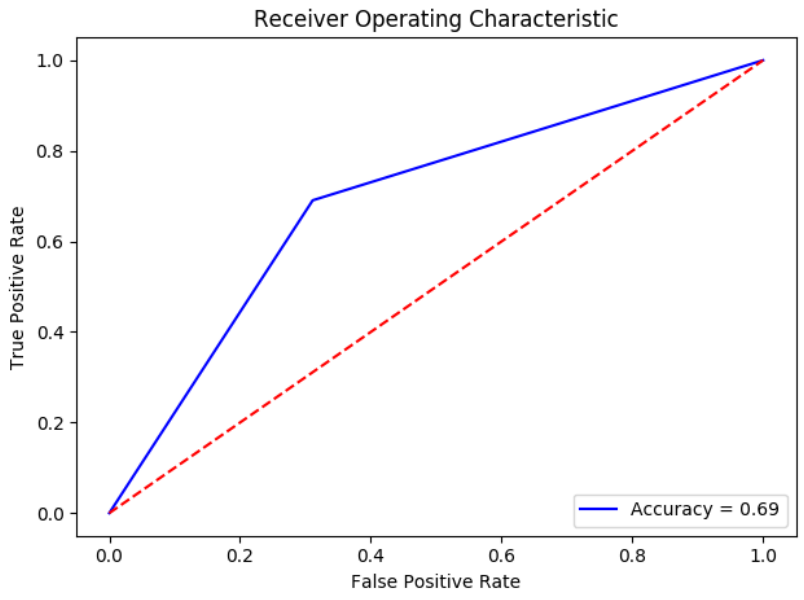
Besides basic result measurements, a confusion matrix, Figure 6, was also created for the K-Nearest Neighbor model to see what kind of error prediction occurred.

Figure 6: Confusion Matrix of Wine Dataset



Along with the confusion matrix, a Receiver Operating Characteristic (ROC) curve was created to help show how accurate the model was. In the case of the K-Nearest Neighbor model, the ROC curve shows that the model is not perfect but does provide some useful insight.

Figure 7: ROC Curve of K-Nearest Neighbor



4. Conclusion

While the project might have not yielded the results that were initially expected, the original goal of using and learning about machine learning techniques and methods was still accomplished. Additionally, further research can be conducted into certain aspects of the project to increase the accuracy. Potential reasons for the inaccurate results might be due to 1) outliers in the data, 2) noise in the data, 3) the experts who qualified the wine were subjective and thus, created errors and inconsistencies, 4) and not enough data, such as price or red wine brand/vineyard. By looking into these various reasons and obtaining the data for each of the red wines in the dataset, one could determine if the accuracy of the models could be increased. Finally, the conclusion of this project should not mean the conclusion of learning more about machine learning. With many other datasets out there, more practice on this data would help with understanding what could be done with this dataset.

References

Developers, s.-l. (2017, 08 6). scikit-learn.org. Retrieved from scikit-learn.org: http://scikit-learn.org/stable/auto_examples/linear_model/plot_logistic.html#sphx-glr-auto-examples-linear-model-plot-logistic-py

Ray, Sunil, and Business Analytics and Intelligence. “8 Proven Ways for Boosting the ‘Accuracy’ of a Machine Learning Model.” Analytics Vidhya, 2 May 2017, www.analyticsvidhya.com/blog/2015/12/improve-machine-learning-results/.

“Scikit-Learn.” *1.4. Support Vector Machines - Scikit-Learn 0.19.1 Documentation*, 2018, scikit-learn.org/stable/.

VanderPlas, Jake. “In Depth: Principal Component Analysis.” Introducing Scikit-Learn | Python Data Science Handbook, jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html.

Yogesh Gupta, Selection of important features and predicting wine quality using machine learning techniques, Procedia Computer Science, Volume 125, 2018, Pages 305-312, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2017.12.041.