Determining Quality of Red Wines

Katherine Kuenster

Department of Computer Science and Mathematics, Santa Clara University
500 El Camino Real, Santa Clara, CA 95053, USA
kkuenster@scu.edu

Abstract— **As a lifetime lover of red wine, I chose to investigate what features of a wine make it good, or bad quality, and to explore various machine learning algorithms in order to predict the quality of red wines, based on these features. While personal preference may play a role in finding a good wine for oneself, it will be interesting to explore how a machine will classify particular wines, and it would be even more interesting to try the wines and see if the machine is correct. While this goal could be approached using various algorithms, supervised and unsupervised, I will be exploring the performance of K-Nearest Neighbors, Logistic Regression, and Gaussian Naive Bayes for this particular dataset.**

I. Introduction

In this project, I explore red wines and their quality, hoping to create a model that can categorize a particular wine as good or bad quality based on its features. I will utilize and compare KNN, logistic regression, and Naive Bayes, and their abilities to correctly classify wines by their quality. The following is a documentation and comparison of the mentioned algorithms, along with an analysis of the wine features that correlate the strongest to its quality.

II. Data

This data[1] was obtained from the University of Portugal and focuses on Portuguese wine and their attributes along with a score indicating the score it received from the taster (1-10). The set of data consists of 1,599 red wines, each attached with the details of the wine, and the given true score. To simplify the decision making, I transformed the target variable, quality, to be binary, such that wines with a score of 6 or higher are good quality (1), and wines with a score of 5 or less are to be considered bad quality (0). The features that will be used to predict our target variable are the following:

1. Fixed Acidity
2. Volatile Acidity
3. Citric Acid
4. Residual Sugar
5. Chlorides
6. Free Sulfur Dioxide
7. Total Sulfur Dioxide
8. Density
9. pH
10. Sulfates
11.Alcohol

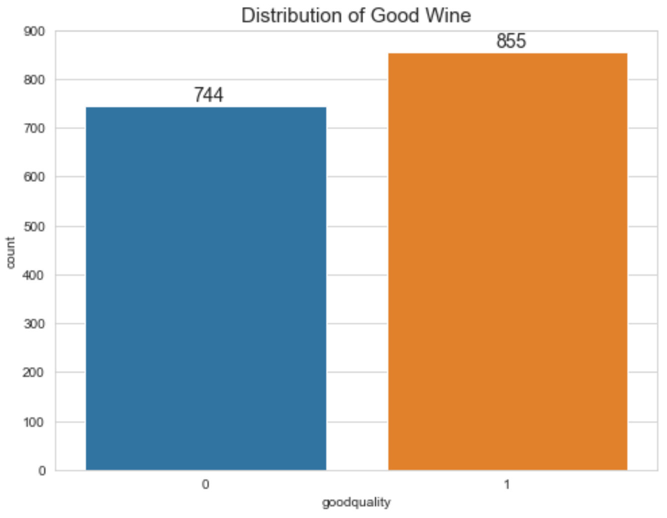


Fig. 1 A plot of the original dataset with the value count for the target variable, where 0 is bad quality red wine (wines scored 5 and below), and 1 is good quality red wine (wines scored 6 or higher).

III. Correlation Matrix

To explore the relationship between the wine features and the quality of the wine, a correlation matrix can be used. A correlation matrix is a table that displays the relationship between variable coefficients. Fig. 2[2] demonstrates the methodology behind acquiring the coefficients used in a coefficient matrix:

$$\mathbf{R} = \begin{pmatrix} 1 & r_{12} & r_{13} & \cdots & r_{1p} \\ r_{21} & 1 & r_{23} & \cdots & r_{2p} \\ r_{31} & r_{32} & 1 & \cdots & r_{3p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{p1} & r_{p2} & r_{p3} & \cdots & 1 \end{pmatrix}$$

where

$$r_{jk} = \frac{s_{jk}}{s_j s_k} = \frac{\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)}{\sqrt{\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}\sqrt{\sum_{i=1}^{n}(x_{ik} - \bar{x}_k)^2}}$$

is the Pearson correlation coefficient between variables $\mathbf{x}_j$ and $\mathbf{x}_k$.

Fig. 2 A mathematical explanation behind the generation of a correlation matrix.

The correlation matrix generated showed that the features most aligned with the quality of the wine were alcohol, sulfates, and citric acid levels, with coefficients of 0.476, 0.251 and 0.226 respectively.

IV. Testing and Training Groups

In order to train and test the algorithms, separating the data into training and testing sets, with a split of 75:25, where 75% of the data (1199 events) to be used for training, and the remaining 25% of the data (400 events) will be used to gage how well the training went, by evaluating the accuracy of the models on the test set.

V. Logistic Regression

Logistic regression is a supervised machine learning algorithm that is intended for datasets containing numerical input types, and a categorical target variable that contains two classes such that the relationship between the classes is binary. Logistic regression creates a linear boundary between classes in order to place future values into their correct classification.

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

The hypothesis function, h(x), returns the predicted class that data point X belongs to. By comparing the predicted class to the actual class, we can find the accuracy, precision, and recall:

```
               Pred Negative  Pred Positive
True Negative            134             51
True Positive             48            167

 Accuracy = 0.752

 Precision = 0.766

 Recall = 0.7767

 F1 Score = 0.771
```

Fig. 3 The confusion matrix for the dataset using logistic regression to predict the test set

By producing a confusion matrix for logistic regression, the model shows ~75% accuracy, which represents correct predictions over all predictions. The precision, correctly classified good wines in comparison to all wines scored positively, is ~77%, and similarly, the recall of the model, representing correctly classified good wines vs. all good wines, is ~78%.

Logistic regression shows that it was able to mostly classify wines into their correct category of good quality or bad quality, but did struggle to correctly classify around 1/8th of the test set for both negativity and positivity.

VI. K Nearest neighbors

K Nearest Neighbors (KNN) is another popular supervised machine learning algorithm that is known for grouping data that is similar, and separating data that is not similar. Specifically, KNN relies on the parameter, K, which determines how many neighbors, or groups, will be available for data to be sorted into. KNN uses and takes advantage of the distance between data points in order to assign them to the closest group/neighbor in order to classify new data points.

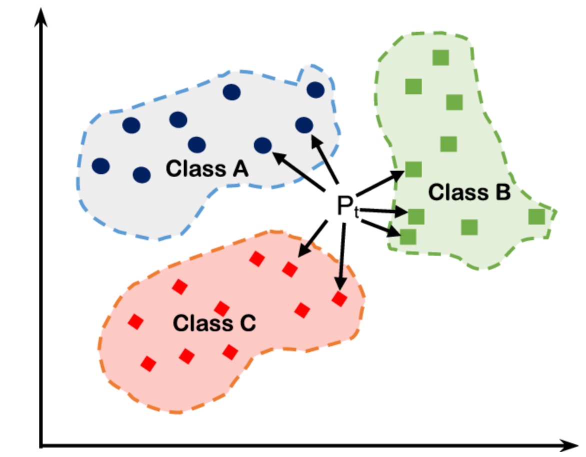


Fig. 4 A visual of KNN and how it classifies new data points

As Fig. 4 demonstrates with k = 3, KNN involves choosing K clusters, such that all points in a cluster are similar, and all points outside of a cluster are as different as possible.

For my KNN model, I decided to use cross-validation to find the best value to use for K, that would give the best testing accuracy. This was important because with KNN, the training accuracy will often be highest with K=1, because every data point is being grouped to a single section, but with a lower K, the testing accuracy will often be lower, because the model has overfit to the training data.

On my training set, I looped through K values of 1 to 100, and to no surprise, the model performed with 100% accuracy for K = 1. To combat this, and to find a proper K value that will help my testing accuracy, I plotted the testing accuracy over K values 1 to 100, and found that a K value of 75 will give the best testing accuracy.

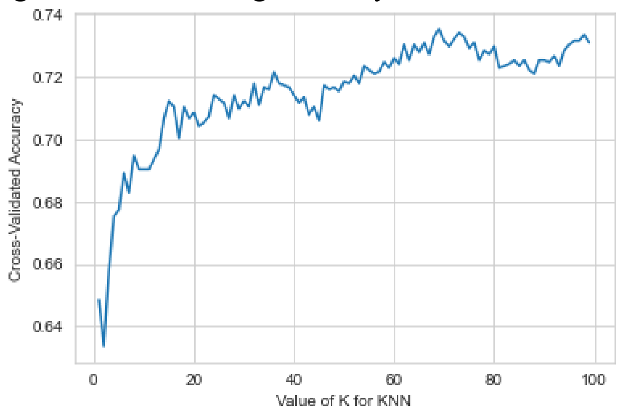


Fig. 5 A visual of KNN and how it classifies new data points

Fig. 5 demonstrates the graph of test accuracy vs. number of K neighbors used, and indicates the highest accuracy around K=75, which I will use for evaluating the performance of my model via a confusion matrix.

As predicted, a K-value of 75 gives the best testing accuracy as follows:

```
                Pred Negative  Pred Positive
True Negative             122             63
True Positive              41            174

 Accuracy = 0.74

 Precision = 0.734

 Recall = 0.8093

 F1 Score = 0.77
```

Fig. 6 The confusion matrix for the dataset using KNN to predict the test set

Fig. 6 demonstrates the performance of KNN using a K-value of 75 on the test set, performing with an overall accuracy of 74%, precision at ~73%, recall ~81%, and f1 score, 77%. Overall, the model performed similarly to logistic regression, doing a decent job at correct classifications, but missing a portion of the data. KNN appears to have done better with classifying good wines as good, over classifying bad wines as bad, which could be a result of there being more good wine examples than bad wine examples.

VII. Gaussian Naive Bayes

Gaussian Naive Bayes is a classifier that makes an assumption that all features in a dataset conform to a Gaussian distribution. The reason Naive Bayes is naive is because it assumes that all features in the dataset are independent of one another, which is typically not the case. Naive Bayes uses Bayes' Theorem to calculate the probability that an event belongs to a particular class, based on the given conditions of that event.

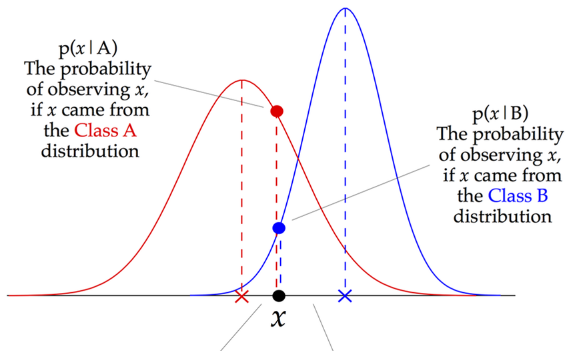
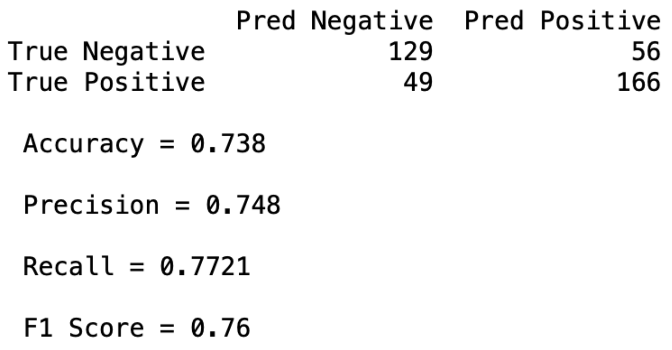

p(x | A)
The probability of observing x, if x came from the Class A distribution
p(x | B)
The probability of observing x, if x came from the Class B distribution
x

Fig. 7 The logistics behind Gaussian Naive Bayes classification, and predicting classes for new data

Using Gaussian Naive Bayes with my dataset, and creating a confusion matrix containing metrics to evaluate performance, we can compare the performance of all three models given the dataset.



```
               Pred Negative  Pred Positive
True Negative            129             56
True Positive             49            166

 Accuracy = 0.738

 Precision = 0.748

 Recall = 0.7721

 F1 Score = 0.76
```

Fig. 8 The confusion matrix produced from using Gaussian Naive Bayes classification to predict the test set

Evaluating the confusion matrix for Gaussian Naive Bayes, it performed similarly to logistic regression and KNN, with an accuracy of 73.8%, a precision score of 74.8%, recall of 77.21%, and an f1 score of 76%. The model was able to correctly classify a majority, but not all events. Gaussian Naive Bayes performed the worst in terms of overall accuracy, which is most likely due to the assumption of feature independence that naive bayes suggests.

VIII. Conclusions

TABLE 1
Accuracy Report for Classification Algorithms

Method	Accuracy	Precision	Recall	F1
LR	0.752	0.766	0.777	0.771
KNN	0.740	0.734	0.810	0.770
GNB	0.738	0.748	0.772	0.760

Based on the results in Table 1, it appears that logistic regression and KNN performed very similarly in terms of metrics, with logistic regression having the highest percentage in all categories except for recall. As mentioned, Gaussian Naive Bayes has the worst overall accuracy, but performed better than KNN did for overall model precision.

From these numbers, I infer that the training was not as successful as it could have been, giving relatively low test scores across all models used. If we were to decide to combat this and go back to receive better training, this could entail removing features from the dataset that show a low correlation to the target variable, quality. Another potential reason behind the low scores could be that the dataset contained a few outliers that it was unable to learn given the majority of the set falling into similar categories.

Data preprocessing can play an essential role in the performance of a model, which can be difficult to combat if you are not as familiar with the set of data you are working with.

I would most likely add more data, if possible, and if the results were still not satisfactory, explore the relationship between features further, and clean out the data to remove outliers and create a more digestible set for the models to learn.

Acknowledgment

I would like to acknowledge the work of P. Cortez, A. Cerdeira, F. Almeida, T. Matos, J. Reis and their research that provided this dataset, as well as the University of California Irvine for making the set more accessible. I owe another thank you to Dr. Smita Ghosh at Santa Clara university for providing me the opportunity to further explore machine learning and the freedom to choose a topic I am passionate about.

References

[1] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, J. Reis. Universidade de Minho, Guimarães, Portugal. Viticulture Commision of the Vinho Verde Region, Porto, Portugal

[2] N. Helwig. *Data, Covariance, and Correlation.* University of Minnesota. 16 Jan. 2017. http://users.stat.umn.edu/~helwig/notes/datamat-Notes.pdf