Curtin University

STAT2004

Analytics for Observational Data

Semester 2, 2025

Final Project Report

An Analysis of Red Wine Quality Using Multivariate Statistical Techniques

Ryan Sheik, 20972312, Bachelor of Science (Data Science), Bachelor of Commerce (Finance)

Peter Misiun, 20886060, Bachelor of Science (Data Science)

Declaration

The work presented in this report is my/our own work and all references are duly acknowledged.

This work has not been submitted, in whole or in part, in respect of any academic award at Curtin University or elsewhere.

SHEIKR

Ryan Sheik

19/10/2025

MISIUNP

Peter Misun

19/10/2025

Contents

Introduction

Our project aims to gain deeper insight into the scope of multivariate data analysis by implanting statistical techniques most relevant to exploring statistically significant insights from our sourced Red Wine Quality dataset. Our comprehensive goal is examining the applicability of observation corresponding to a practical scenario through utilising the 'rrr' package within the R program, explicitly chosen for its capabilities within multivariate modelling.

Previous research into factors affecting wine quality were observed through sensory evaluation through properties such as aroma, feel and visual appearance. The primary issue in this approach is subjective and time-consuming, relying on specialised expertise and hence is not the most effective approach to determining wine quality. The ability to predict wine quality through objective measurements of chemical properties provides invaluable insight to achieve optimal wine-quality whilst factoring important chemical factors that yield quality-wine.

Our 'Red Wine Quality Dataset', retrieved through the UCI Machine Learning Repository (Cortez et al. 2009) reveals physicochemical test results for 1,599 samples of Portuguese red wine known as 'Vinho Verde' supplemented with quality-testing results assigned by wine experts.

Our core research questions guiding our multivariate analysis are: Are we able to model and therefore predict wine quality based on chemical properties, using multivariate statistical modelling and what insights can these models convey in respect to the relationship between chemical properties and wine quality?

These principal goals of our analysis to model and further examine a relationship between the response variables (wine quality) and the predictor variables (physicochemical properties) can be answered through the implementation of principal component analysis (PCA), linear discriminant analysis (LDA) and reduced rank regression (RRR). The 'rrr' package within R enables us to create (using the 'rrr()' function) and visualise (using the 'rank_trace()' and 'pairwise_plot()' functions) such models. In doing so, insights regarding the most significant chemical properties impacting wine quality as well as predictions of wine quality based on these properties can be deduced.

Methods

The dataset structure retrieved from the UCI Machine Learning Repository (Cortez et al. 2009), affirmed 11 predictor variables indicative of the chemical properties and 1 response variable (a quality score ranging from 3-8). The predictor variables include physical properties (density), salt content (chlorides), sugar content (residual sugar), preservatives (sulphates), acidity content (fixed acidity, volatile acidity, citric acid and pH), sulphur dioxide levels (free and total) and alcohol content. Our dataset chosen meets the project requirements regarding our analysis: it contains more than 4 numerical variables, contains no missing values, has more than 7 columns (p=12), has more than 50 rows (n=1599) as well as most importantly, satisfying the condition n>>p, making it apt for multivariate analysis.

Key Features of the 'rrr' package

The 'rrr' package provides a unified framework for our employed multivariate techniques like PCA, LDA and RRR, allowing for consistent implementation and comparison across methods. This package includes automatic rank selection procedures based on information criteria. Visualisation functions commonly used within R, such as 'rank_trace()' and 'pairwise_plot()' aid in visually facilitating the interpretation of our results.

The core function of 'rrr()' is to perform reduced-rank regression. By specifying different type arguments, the 'rrr()' function can perform PCA (type = 'PCA'), LDA (type = 'LDA') and standard reduced-rank regression (type = 'rrr') in which the rank parameter controls the dimensionality of our reduced-rank output.

In our analysis, two of the core functions used are:

```
# PCA using rrr package
pca_rrr <- rrr(predictors_scaled_df, predictors_scaled_df,
            type = "pca", rank = "full")

# Reduced-rank regression
rrr_result <- rrr(x = predictors_matrix, y = response_matrix, rank = 1)
```

Multivariate Techniques Employed

Through carrying out reduced-rank regression, we model our relationship using the formula Y = XC + E, in which Y represents the response matrix (n x q). Here X is a (n x p) matrix of predictor variables, C represents a (p x q) matrix of regression coefficients whilst E represent the error terms in our regression model.

The principal goal of RRR is that coefficient matrix (C) does not use all the possible dimensions. Rather, its ranks are restricted to r < min (p, q). Therefore, C can be reduced into C=AB^T, where A represents p x r while B represents q x r. The columns within the smaller matrices define hidden factors (r) that accurately capture the relationship between predictors and responses.

In simpler terms, RRR glances over minor patterns that convey how variable X is related to Y. For the univariate (q=1), applicable to our Wine Quality analysis, RRR simplifies to identifying a single direction in the predictor space that can best predict the outcome.

Before applying multivariate techniques to our retrieved dataset, it is integral to recognise and handle outliers to ensure our analysis is not skewed. Outliers were detected through applying the Interquartile Range method (IQR), highlighting any data points that fall well below/above our typical range. Rather than removing our outliers from our dataset, this would result in potentially valuable information being lost, we have applied a method known as winsorization. Winsorization works to replace extreme values (outliers in this instance) with less extreme values falling under specific percentiles (5% & 95% percentiles in our analysis), preserving the sample, yet reducing the impact of outliers and not skewing our analysis in the process. This process is crucial for the LDA model, as this model requires the data used to be as close to normal as possible.

Standardisation is an essential process for multivariate analysis so predictor variables with larger numerical ranges do not provide inaccurate analysis and subsequently make predictor variables sensitive to variable scales. All predictor variables in our dataset were standardised (scaled to have a mean of 0 and standard deviation of 1), through the integration of the 'scale()' function.

PCA is performed to reduce the number of variables whilst identifying the main sources of variation in our dataset. We conducted our PCA analysis using two methods: the standard prcomp() function along with the 'rrr()' function setting the type to 'pca' to ensure consistent results. In deciding the number of components to retain were determined based on two criteria: cumulative variance (retaining enough components to at least explain 75% total variation in data), eigenvalue criterion (only keeping components with an eigenvalue > 1, more simply known as Kaiser's criterion.

LDA is applied to categorise wines into 3 quality groups (low, medium or high) based on their chemical properties. Low wine quality scores ranged from 3-5, medium quality was rated at 6 and high-quality wine scores ranged from 7-8. This categorisation was implemented using the following code:

```
redwine_data$quality_class <- cut(redwine_data$quality,
                                  breaks = c(0, 5, 6, 10),
                                  labels = c("Low", "Medium", "High"))
```

After categorisation, LDA was then performed via the ‘lda()’ function retrieved from the MASS package (Venables and Ripley 2002) on the scaled predictor variables. The decision to not use the ‘rrr’ package for LDA was made as the results produced were difficult to interpret and use effectively.

The performance of the LDA model was evaluated using R-squared (proportion of variance explained) and mean squared error (MSE). Coefficient magnitudes were examined to identify which chemical properties have the strongest influence on wine quality predictions.

Results

Exploratory Data Analysis (EDA) Results:

EDA highlighted key features of the wine quality dataset. Summary statistics from our analysis displayed vast variation in scales among predictor variables, from pH values averaging 3.311 (ranging from 2.74 – 4.01) to sulphur dioxide concentration averaging 46 (ranging from 6 – 289). Fixed acidity yielded an average of 8.32 g/L (tartaric acid), to volatile acidity averaging 0.528 g/L (acetic acid) and alcohol content averaged 10.42% by volume. These variations among chemical properties highlight diversity in wine composition, partly attributed to differing grape varieties, growing conditions and winemaking processes.

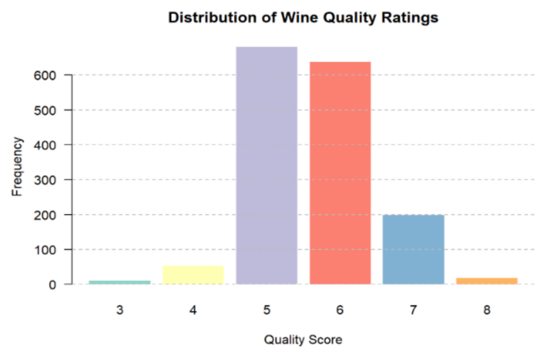


Figure 1, Distribution of Wine Quality Ratings

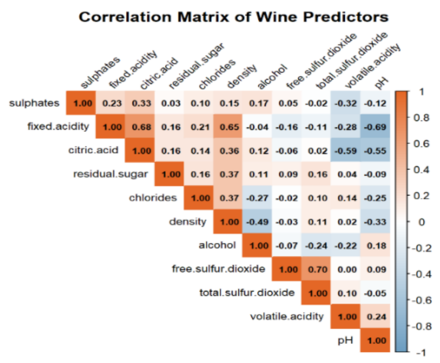


Figure 2, Correlation Matrix of Wine Predictors

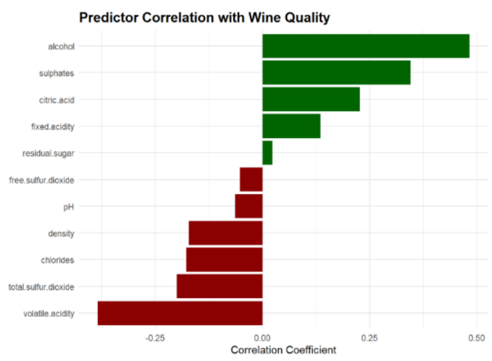


Figure 3, Predictor Correlation with Wine Quality

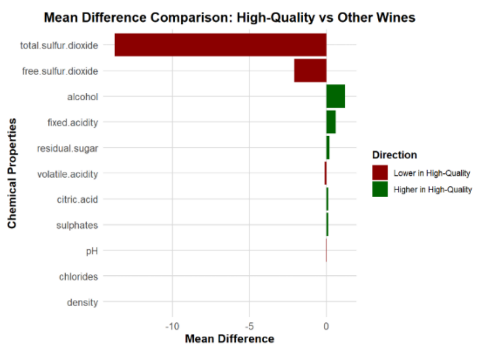


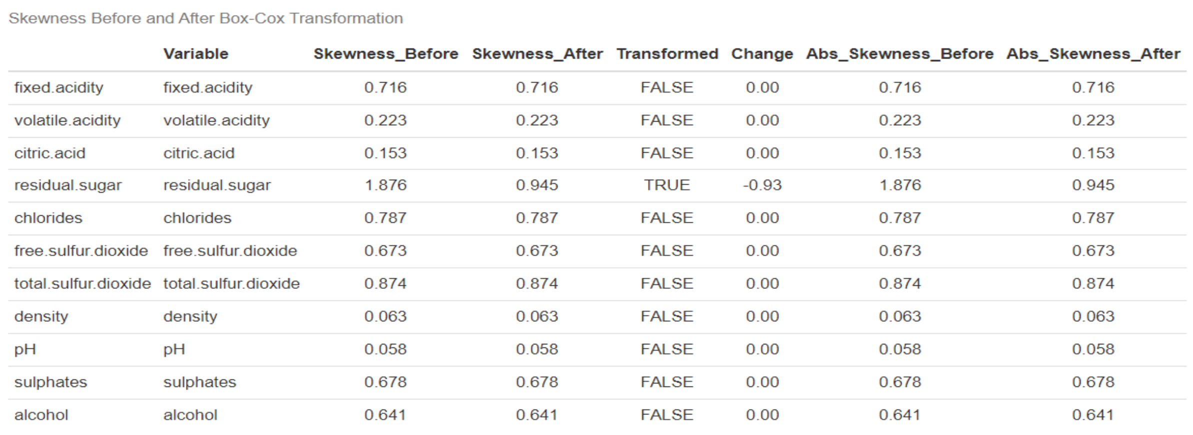
Figure 4, Mean Difference Comparison: High-Quality vs Other Wines

Figure 1 displays the distribution of wine quality ratings within our red wine dataset. This sees wines with a quality rating of 5 having the highest proportion, with wines with a quality rating of 6 falling not far behind. This displays the high number of wines tested with a ‘medium’ quality within our dataset, leading to questions on what properties within the wine caused this.

Figure 2 above reveals correlation among the 11 different predictor variables. Strong positive correlations are highlighted in dark orange whilst strong negative correlations are highlighted through darker hues of blue. Notable relationships between predictor variables include strong positive correlation between fixed acidity and citric acid (yielding a correlation coefficient of 0.67) and the strong negative correlation between fixed acidity and pH (r = -0.69).

Figure 3 identifies alcohol content as the strongest positive predictor of wine quality ($r \approx 0.48$), followed by sulphates ($r \approx 0.25$) and citric acid ($r \approx 0.23$), while volatile acidity exhibits the strongest negative correlation ($r \approx -0.39$), consistent with its association with undesirable vinegar-like characteristics. Total sulphur dioxide, chlorides, and density show moderate negative correlations, whereas residual sugar, free sulphur dioxide, pH, and fixed acidity demonstrate negligible relationships with quality. These correlation patterns confirm that wine quality is determined by multiple chemical properties rather than any single factor, with the moderate correlation strengths (all $|r| < 0.5$) justifying the application of multivariate techniques (PCA) for dimensionality reduction, LDA for quality classification, and RRR for predictive modelling—to effectively capture these multidimensional relationships and address the research objectives.

Figure 4 displays the mean difference of the predictor variables in wines with a quality score above and below 7, conveying high and low/medium quality wines respectively. This provides a comprehensive view of which specific variables differ the most in specific wine quality categories. This sees variables relating to the sulphur dioxide content showing the highest negative difference, presenting low quality wines as typically having a higher sulphur dioxide content compared to high quality wines. These higher sulphur dioxide levels are associated with unpleasant aromas which typically take away from the wine's natural flavours, explaining the lower quality scores.



Skewness Before and After Box-Cox Transformation

	Variable	Skewness_Before	Skewness_After	Transformed	Change	Abs_Skewness_Before	Abs_Skewness_After
fixed.acidity	fixed.acidity	0.716	0.716	FALSE	0.00	0.716	0.716
volatile.acidity	volatile.acidity	0.223	0.223	FALSE	0.00	0.223	0.223
citric.acid	citric.acid	0.153	0.153	FALSE	0.00	0.153	0.153
residual.sugar	residual.sugar	1.876	0.945	TRUE	-0.93	1.876	0.945
chlorides	chlorides	0.787	0.787	FALSE	0.00	0.787	0.787
free.sulfur.dioxide	free.sulfur.dioxide	0.673	0.673	FALSE	0.00	0.673	0.673
total.sulfur.dioxide	total.sulfur.dioxide	0.874	0.874	FALSE	0.00	0.874	0.874
density	density	0.063	0.063	FALSE	0.00	0.063	0.063
pH	pH	0.058	0.058	FALSE	0.00	0.058	0.058
sulphates	sulphates	0.678	0.678	FALSE	0.00	0.678	0.678
alcohol	alcohol	0.641	0.641	FALSE	0.00	0.641	0.641

Figure 5, Normality Checks on Predictor Variables After Outlier Removal

Figure 5 displays the use of a Box-cox transformation, testing whether the predictor variables require more steps post outlier removal to make them more normal. This method uncovers which variables display an absolute skewness above 1 and takes the steps necessary to make this variable have an absolute skewness below 1. We see that all predictor variables except residual sugar pass this test. This sees residual sugar undergo a box-cox transformation, going from having an absolute skewness of 1.9 to 0.9. This ensures that our data expresses adequate normality for the models benefiting from using normal data.

Principal Component Analysis (PCA) Results:

PCA was conducted using traditional methods alongside the 'rrr' package to capture the main sources of variation within the wine quality dataset. Based on the criterion mentioned of retaining components that convey at least 75% of cumulative variance, 5 principal components were chosen for in-depth analysis.

	PC	Eigenvalue	Variance	Cumulative
PC1	1	3.136	28.512	28.512
PC2	2	2.046	18.603	47.115
PC3	3	1.600	14.543	61.658
PC4	4	1.081	9.823	71.481
PC5	5	0.897	8.157	79.638
PC6	6	0.711	6.461	86.100
PC7	7	0.561	5.103	91.203
PC8	8	0.414	3.764	94.966
PC9	9	0.303	2.754	97.720
PC10	10	0.187	1.699	99.420
PC11	11	0.064	0.580	100.000

Figure 6, PCA Summary Table

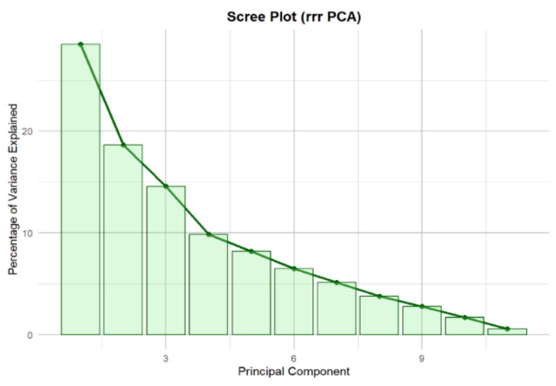


Figure 7, Scree Plot for PCA

The first 5 components demonstrate 79.64% of the total variance, surpassing the 75% variance commonly used in PCA applications. This indicates these 5 components capture a large portion of the systematic variation within the dataset yet substantially reducing the dimensionality (from 11 predictor variables to 5 variables). The first 4 components all have eigenvalues > 1, adhering to Kaiser's criterion, and principal component 5 yields an eigenvalue of 0.897. Although <1, PC5 warrants inclusion to our PCA as it still explains 8.157% variance, justifying its inclusion.

This in addition to the scree plot produced using data obtained from the 'rrr' package makes it easier to interpret the variance explained by each principal component. This shows that as the index of the principal component increases, the percentage of variance explained decreases.

Variable	PC1	PC2	PC3	PC4	PC5
fixed.acidity	0.5015	-0.0606	-0.1014	0.0341	0.1726
volatile.acidity	-0.2473	0.3657	-0.3485	-0.2027	-0.0304
citric.acid	0.4640	-0.1843	0.1981	0.0589	0.1093
residual.sugar	0.1663	0.1831	0.0942	-0.7960	0.2638
chlorides	0.2178	0.2956	-0.2026	-0.1232	-0.5012
free.sulfur.dioxide	-0.0599	0.3649	0.5882	0.0977	0.0226
total.sulfur.dioxide	0.0068	0.4810	0.4793	0.1086	0.0876
density	0.3978	0.3092	-0.2054	-0.1355	-0.0291
pH	-0.4242	-0.0142	0.0534	-0.2620	-0.2436
sulphates	0.2195	-0.1718	0.2785	-0.1067	-0.7506
alcohol	-0.0832	-0.4668	0.2842	-0.4310	0.0690

Figure 8, Component Loadings for First 5 Principal Components

Figure 8 displays the loadings of the first 5 principal components, disclosing the underlying structure of wine-quality dataset. These 5 principal components can be interpreted as the following:

PC1 – General Acidity and Density Component (28.512% variance): PC1 displayed strong positive loadings for fixed acidity (0.5015), citric acid (0.4640) and density (0.3978), implying wines with high PC1 scores have higher fixed and citric acid content, greater density and lower pH.

PC2 – Sulphur Dioxide and Alcohol Component (18.603% variance): PC2 is strongly linked to total sulphur dioxide (0.4810), free sulphur dioxide (0.3649), and volatile acidity (0.3657), while alcohol has a strong negative loading (-0.4668). This component distinguishes wines that have higher sulphur dioxide concentration from wines with higher alcohol content, reflecting a preservation vs alcohol balance in respect to winemaking.

PC3 – Freshness component (14.543% variance): PC3 shows strong positive loadings for free sulphur dioxide concentration (0.5882) and total sulphur dioxide concentration contrasted by a strong negative loading for volatile acidity (-0.3485). It differentiates fresher, well-preserved wines from those with higher volatile acidity.

PC4 – Sweetness Component (9.823% Variance): PC4 is dominated by a strong negative loading for residual sugar content (-0.7960), with moderate negative loadings for alcohol (-0.4310) and pH (-0.2620). This component primarily reflects the sweetness of wine.

PC5 – Mineral Composition component (8.157% variance): PC5 displayed strong negative loadings for sulphates (-0.7506) and chloride (-0.5012), capturing the variation in the mineral content of the wines.

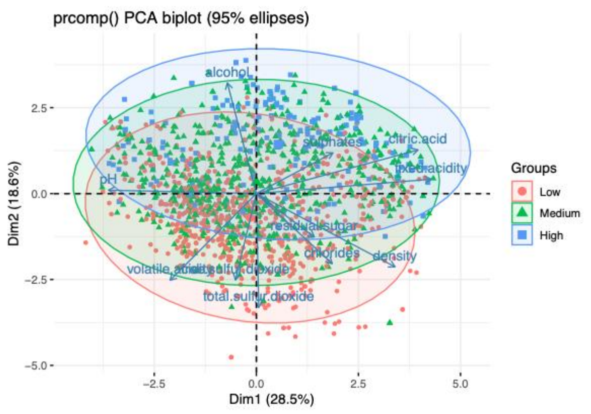


Figure 9, PCA biplot using prcomp() results

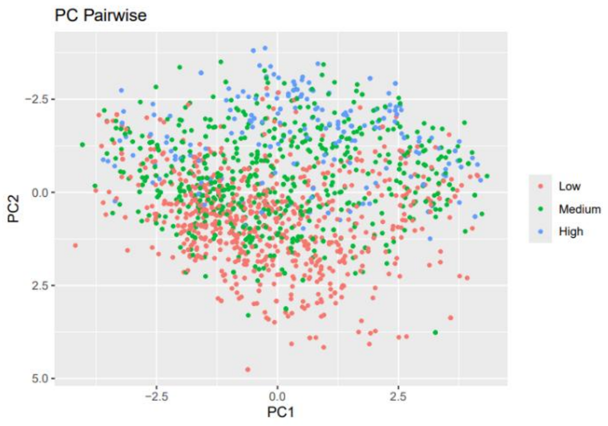


Figure 10, PCA biplot using 'pairwise_plot()' from 'rrr'

Figure 9 illustrates the wine samples (coloured data points) in the space of the first 2 principal components, along with loading vectors (arrows) showing how the original variables impact these components. The plot reveals separation among quality classes, with high-quality wines (blue) clustering towards the right side of PC1 and lower portion of PC2. The loading vectors reveal that alcohol points towards the high-quality cluster, while volatile acidity, density and predictors relating to sulphur dioxide content point in the opposite direction. This pattern suggests that higher alcohol levels, coupled with lower acidity, density and sulphur dioxide content, are key distinguishing factors between higher-quality wine and lower-quality wine respectively.

Figure 10 shows the biplot produced using the 'pairwise_plot()' function from 'rrr'. This plot is essentially the same (after y-axis reversed) as figure 9 (without ellipses and loading vectors). This shows the implementation of the 'rrr' package can be adequate in producing accurate PCA results, however, the interpretation at times can differ, requiring slight adjustments.

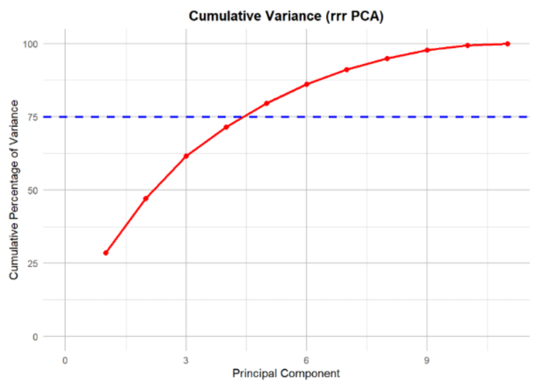


Figure 11, Cumulative Variance Plot

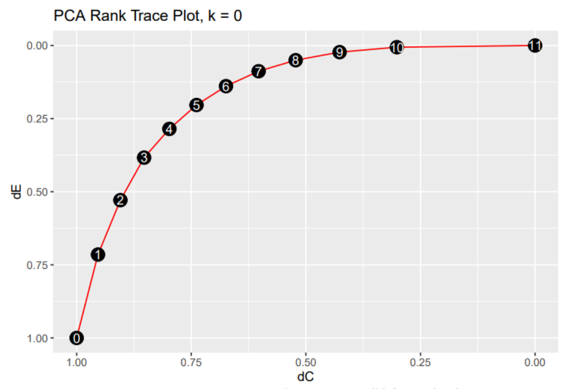


Figure 12, Plot produced using 'rank_trace()' from 'rrr'

Figures 11 and 12 display a cumulative variance plot and a plot created using the 'rank_trace()' function with the 'rrr' package respectively. The rank trace plot required the x and y axis to be reversed, but in doing so, a plot conveying the same information as figure 11 was produced. This is due to the plots differing axis meanings, in which the rank trace plot from 'rrr' starts at PC11 and goes to PC1, showing that after PC1, 100% of the total variance is explained. This differs from the cumulative variance plot, as it starts from PC1, and goes until PC11. This gives us a visual representation of how when the principal component index increases, the variance explained by each subsequent principal component decreases but the cumulative variance increases.

```
##           Component  ANOVA_p KW_p Pearson_r  Corr_p Significance
## cor...1         PC1 0.00000    0     0.130 0.00000          ***
## cor...2         PC2 0.00000    0     0.483 0.00000          ***
## cor...3         PC3 0.00000    0     0.270 0.00000          ***
## cor...4         PC4 0.00000    0     0.127 0.00000          ***
## cor...5         PC5 0.00546    0     0.073 0.00352           **
```

Figure 13, PCA Inference Summary Table

Figure 13 displays a summary table after completing inference tests, testing for significance in specific areas. As all ANOVA and Kruskal-Wallis p-values are below 0.05 for each principal component, this means that there is a statistically significant difference present between wine quality categories. In addition to this, PC2 was the only principal component to achieve a Pearsons correlation coefficient above 0.3 (achieved 0.48), conveying PC2 as being the most informative principal component for explaining differences in wine quality. As PC2 captures sulphur dioxide and alcohol content within the wines, this displays sulphur dioxide and alcohol content as being the most informative properties in predicting overall wine quality.

Linear Discriminant Analysis (LDA) Results:

LDA was performed to categorise wines into differing wine-quality scores, resulting in 744 Low quality wines (46.5%), 638 Medium quality (39.9%) and 217 high quality (13.6%). The coefficients of the linear discriminates from our analysis presented the importance of different chemical properties for quality classification:

```
## Coefficients of linear discriminants:
##                              LD1        LD2
## fixed.acidity         0.27176593  0.9344151
## volatile.acidity     -0.40187943  0.4208420
## citric.acid          -0.18547239  0.6086221
## residual.sugar        0.10980553  0.8044242
## chlorides            -0.13751895 -0.1498870
## free.sulfur.dioxide   0.09411355 -0.6522766
## total.sulfur.dioxide -0.30001473  0.7046993
## density              -0.17945919 -1.5453851
## pH                   -0.07794741  0.3389733
## sulphates             0.46997869  0.1898560
## alcohol               0.73211122 -0.7473873
##
## Proportion of trace:
##    LD1    LD2
## 0.9602 0.0398
```

Figure 14, Linear Discriminant Table Summary

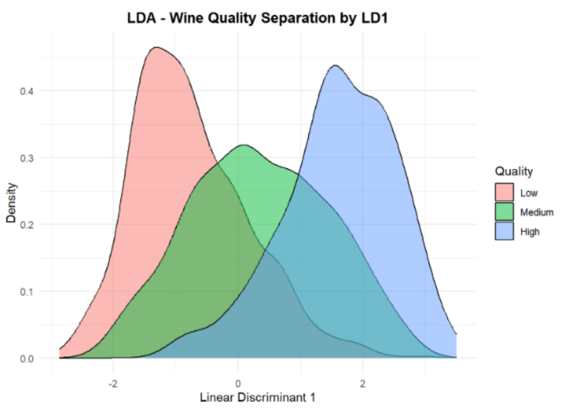


Figure 15, Wine Quality Separation by LD1

Figure 14 shows LD1 (the first linear discriminant) disclosed the highest coefficients for alcohol (0.732), sulphates (0.470) and fixed acidity (0.272), with negative coefficients for volatile acidity (-0.402), total sulphur dioxide (-0.300) and density (-0.719). This implies LD1 primarily separates wines based on alcohol content, sulphate levels and acidity characteristics. The variance LD1 explained **96.02%** of the between-group variance, while LD2 explained only 3.98%, indicates that wine quality classes are primarily separated along a single dimension related to the chemical properties captured by LD1. Due to this result, in later steps we will be using the results from LD1 as the overall LDA result.

In addition to this, figure 15 shows LD1 being able to distinguish between low (red), medium (green) and high (blue) quality wines, represented by 3 distinct regions on the plot. This shows that typically, low-quality wines tend to have a lower LD1 score, whereas higher quality wines tend to have a higher

LD1 score. This indicates that chemical properties used in the analysis have meaningful discriminatory power and can be used to accurately predict wine quality based on the specific chemical properties. However, the limitations of this model can be identified as incorrect classification can occur in some cases. This is due to the medium quality wines having a widespread region on the density plot, deeply encroaching on the regions designated for the low and high-quality wines. This is likely due to the gradual quality transition between wine qualities, instead of distinct boundaries and differences.

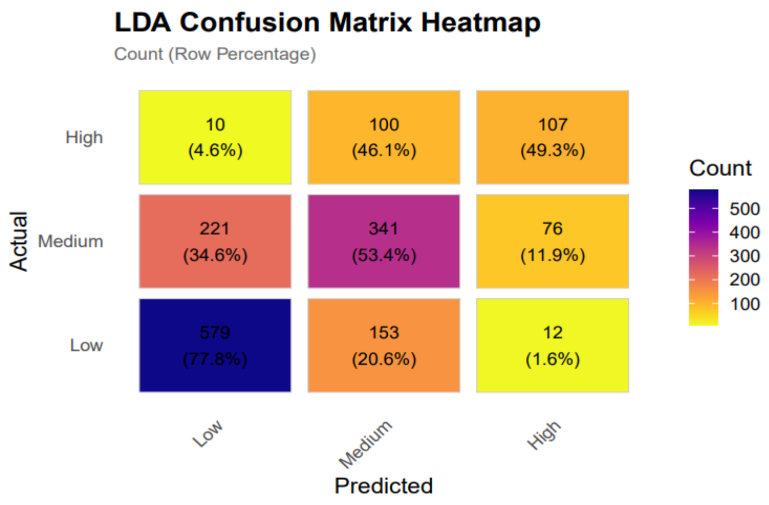


Figure 16, LDA Predictions Confusion Matrix

Figure 16 displays the predictive power of the LDA model. This displays how classifying different wine qualities into low, medium and high categories produced an overall classification accuracy of 64.23%. This result suggests chemical properties do contain essential information for predicting wine quality, though the relationship is not deterministic. The model was most effective at identifying low-quality wines (78% accuracy) and had more difficulty distinguishing between medium (53% accuracy) and high-quality (49% accuracy) wines, which is not surprising as the distinction between these categories may depend more on subtle sensory characteristics not fully captured by the measured chemical properties.

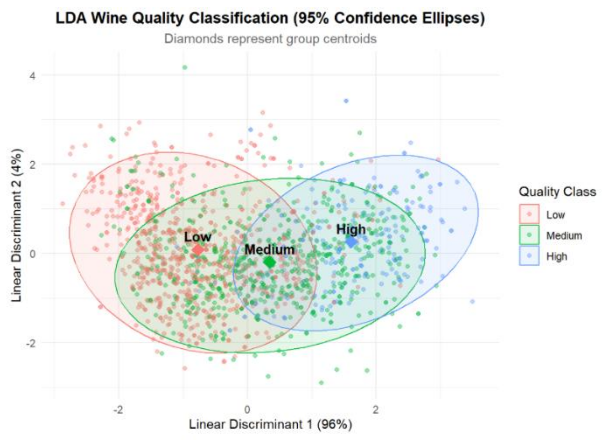


Figure 17, LD1 vs LD2 Plot Created Using 'pairwise_plot()' from 'rrr'

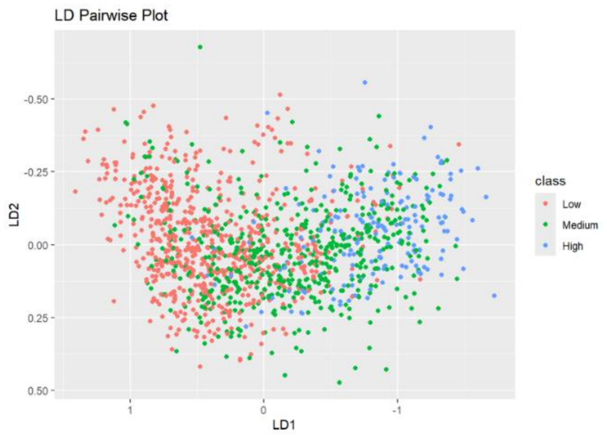


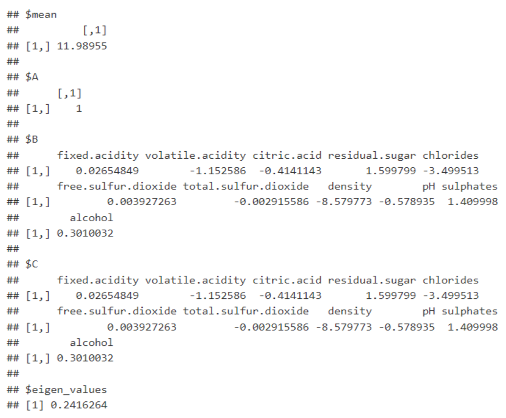
Figure 18, LD1 vs LD2 Plot Created Using 'pairwise_plot()' from 'rrr'

Figure 17 shows the separation of the three category classes in the space of the 2 linear discriminants. The 95% confidence ellipses (coloured ovals) show the expected distribution of each respective class. There is clear separation along LD1 (horizontal axis), with low-quality wines on the left, medium-quality wines in the centre, and high-quality wines on the right. The overlap between classes indicates that perfect separation is not possible based on chemical properties alone, reflecting the inherent complexity and subjectivity of wine quality assessment. High-quality wines have higher LD1 scores, characterised by higher alcohol content, higher sulphates concentration, lower volatile acidity (avoiding vinegar-like characteristics), and lower total sulphur dioxide.

Figure 18 displays the same information as figure 17 but was created using the ‘pairwise_plot()’ function from the ‘rrr’ package and having its x and y axes reversed. This further solidifies how the ‘rrr’ package can be implemented in order to produce meaningful and interpretable results, but might require transformations in certain areas to generate the same results as other methods.

Reduced-Rank Regression (RRR) Results:

Reduced-rank regression was implemented through the ‘rrr’ package to model the relationship between chemical properties and wine quality. A rank-1 model was fitted, as the response variable(quality) is univariate. The RRR model output from our analysis included:



```
## $mean
##          [,1]
## [1,] 11.98955
##
## $A
##      [,1]
## [1,]    1
##
## $B
##      fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## [1,]    0.02654849        -1.152586  -0.4141143       1.599799 -3.499513
##      free.sulfur.dioxide total.sulfur.dioxide   density        pH sulphates
## [1,]         0.003927263         -0.002915586 -8.579773 -0.578935  1.409998
##        alcohol
## [1,] 0.3010032
##
## $C
##      fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## [1,]    0.02654849        -1.152586  -0.4141143       1.599799 -3.499513
##      free.sulfur.dioxide total.sulfur.dioxide   density        pH sulphates
## [1,]         0.003927263         -0.002915586 -8.579773 -0.578935  1.409998
##        alcohol
## [1,] 0.3010032
##
## $eigen_values
## [1] 0.2416264
```

Figure 19, RRR Result Summary

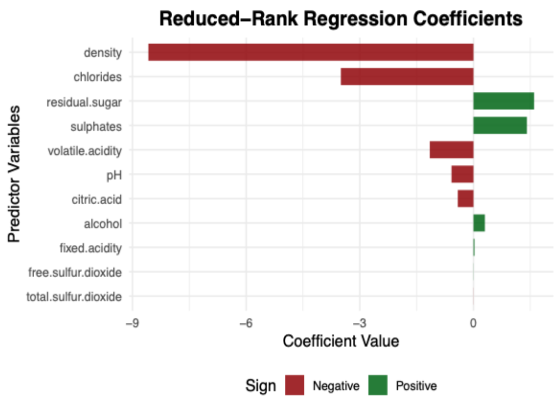


Figure 20, RRR Coefficients Plot

The coefficient matrix (displayed in figures 19 and 20) shows how each chemical property plays a factor into determining wine-quality and indicates that variables with the largest absolute coefficients have the most substantial influence in deciding the wine quality.

Density shows a very strong negative effect (-8.58), indicating higher density is associated within lower-quality wine. However, this relationship must be interpreted carefully. Density is inversely related to alcohol content because ethanol is less dense than water, so the negative density coefficient indirectly reflects the positive relationship between alcohol and quality.

On the other hand, residual sugar and sulphates are recognised to have a strong positive influence, obtaining a coefficient of 1.60 and 1.41 respectively, suggesting that greater sugar and sulphate concentration is linked to higher quality wine. This is due to sugar’s naturally enjoyable sweet taste and sulphates (primarily potassium sulphate), act as antioxidants protecting against oxidation, enhance microbial stability, and contribute to aroma complexity.

Chlorides (-3.50) and volatile acidity (-1.15) both exhibit moderate negative effects. Chlorides at elevated levels can impart a salty taste to wine, which is generally undesirable. The negative coefficient indicates that wines with lower chloride content tend to receive higher quality ratings. Volatile acidity, primarily acetic acid produced by bacteria, can give wine a vinegar-like character at high levels. The negative coefficient aligns with wine quality principles: excessive volatile acidity is a wine fault that detracts from quality.

Together, these patterns show that wines with lower density, lower chloride and acidity levels combined with higher sulphate and residual sugar concentrations tend to achieve higher quality wine-tasting ratings.

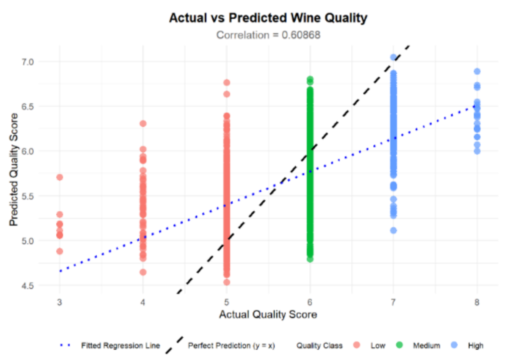


Figure 21, Actual vs Predicted Wine Quality Results

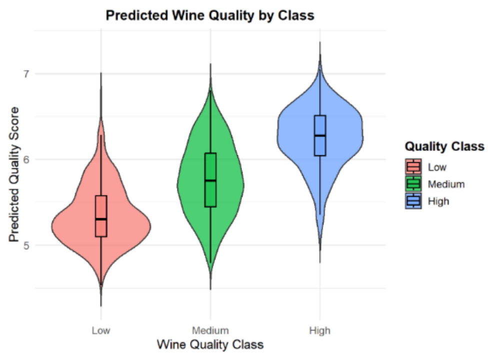


Figure 22, Predicted Wine Quality Results (Violin Plot)

The overall model performance was evaluated using R^2 and mean squared error. From our analysis, the RRR model produced an R^2 value of 0.3705, indicating that the RRR model explains approximately 37% of the variance in wine quality scores.

The mean squared error (MSE) of 0.4103 indicates that, on average, the squared difference between predicted and actual wine-quality scores is about 0.41. Taking the square root gives a root mean squared error (RMSE) of approximately 0.64, implying predictions typically deviate from quality achieved by about 0.64 points on the quality scale. The correlation between actual and predicted quality scores was approximately 0.609, consistent with the R^2 value ($0.609^2 \approx 0.370$). This moderate positive correlation indicates that the model captures a meaningful portion of the quality variation and can predict wine quality based on properties with relative accuracy, though substantial unexplained variance remains. This information is visually displayed in figure 21.

Figure 22 displays the predictive capabilities of the RRR model, showing how low, medium and high-quality wines can be distinguished amongst each other. This is shown as low quality wines (set to a quality score of 5 and under), have a median predicted score of about 5.3, medium quality wines (set to a quality score of 6), had a median predicted score of about 5.8 and high quality wines (set to a quality score of over 6), had a median predicted score of about 6.3. This shows how although on average, the model was able to distinguish low, medium and high-quality wines, low quality wines were often predicted as having a higher quality and higher quality wines were often predicted as having a slightly lower quality. This is likely due to the bias present within the data, in which the dataset presents a significant number of wines with a quality score of 5-6.

Overall Results and Interpretation:

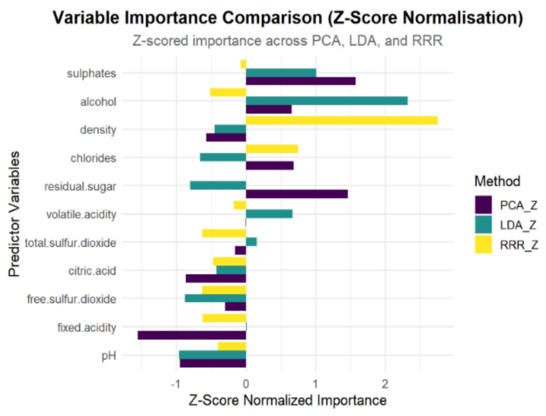


Figure 23, Variable Importance Comparison (Z-Score Normalisation)

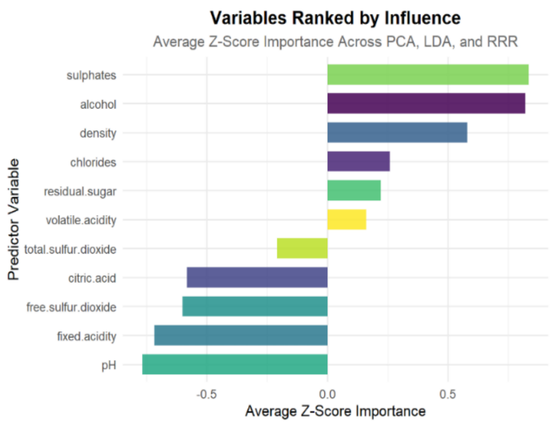


Figure 24, Variable Ranked by Influence

The variable importance comparison across PCA, LDA, and RRR (Figures 23 and 24) implements Z-score normalisation to compare results among the models created and discover which variables are the most influential overall in affecting wine quality.

The plots reveal that alcohol, sulphates, and density consistently emerge as the most influential chemical properties for determining wine quality. Sulphates demonstrate the highest overall importance with an average Z-score of approximately 0.8. Alcohol is also demonstrated as having high overall importance with an average Z-score of approximately 0.8, showing particularly strong predictive power in LDA (Z-score ≈ 2.3). Density follows not far behind sulphates and alcohol, presenting an average Z-score ≈ 0.6. Chlorides, residual sugar, and volatile acidity display moderate importance with Z-scores ranging between 0.15 and 0.3, while pH, fixed acidity, free sulphur dioxide, citric acid, and total sulphur dioxide show negative normalised importance when compared to the other predictor variables. The consistency of these findings across three different analytical approaches strengthens confidence that alcohol content, sulphate levels, and density are the primary chemical determinants of wine quality, directly addressing our research objective of identifying which physicochemical properties have the most significant impact on quality ratings.

These importance patterns reveal the key chemical differences between high-quality and lower-quality wines. High-quality wines are characterized by elevated alcohol content, which reflects optimal grape ripeness at harvest and contributes to desirable body and mouthfeel, alongside higher sulphate concentrations that enhance stability and aroma complexity. Lower density in high-quality wines serves as an indirect indicator of higher alcohol content, since ethanol is less dense than water. Additionally, high-quality wines exhibit lower chloride levels, avoiding undesirable salty characteristics, and reduced volatile acidity, preventing the vinegar-like faults associated with bacterial contamination. In contrast, variables with near-zero importance scores—such as pH, fixed acidity, and free sulphur dioxide—show little differentiation between quality classes, suggesting that wine quality is primarily driven by a focused set of chemical properties related to fermentation completeness, preservation quality, and absence of faults, rather than the full spectrum of measured characteristics.

Discussion

Synthesis of Findings and Connection to Objectives

Our comprehensive multivariate analysis of wine quality has successfully addressed the project objectives outlined in the Introduction. The three multivariate techniques; PCA, LDA and RRR provided consistent results pertaining to the determinants of wine quality.

Addressing Objective 1 (Multivariate Data Analysis): The PCA results displayed five principal components explained nearly 80% of the total variance in the chemical properties linked to wine quality. These components capture important sources of variation associated with predictor variables: general acidity and density (PC1, 28.5%), sulphur dioxide and alcohol (PC2, 18.6%), freshness and preservation (PC3, 14.5%), sweetness (PC4, 9.8%), and mineral composition (PC5, 8.2%), providing an interpretable breakdown of wine chemistry. Overall, the analysis effectively reduced the dataset from 11 original variables to just 5 core predictor variables, retaining most of the relevant information while simplifying the structure of the data. Using inference techniques, it was discovered that PC2 (sulphur dioxide and alcohol) was the most important in capturing information regarding different wine quality properties.

Addressing Objective 2 (R Package Demonstration): The 'rrr' package proved to be an effective and versatile tool for multivariate statistical modelling, allowing us to perform PCA and RRR. The visualisation functions captured useful visual summaries of results. The implementation was user friendly and provided clear output, including coefficient matrices and goodness-of-fit measures.

Addressing Objective 3 (Identifying Key Chemical Properties): The analysis consistently identified alcohol content, sulphates, and density as the most important chemical properties influencing wine quality across all three multivariate techniques. In the LDA, alcohol had the highest coefficient (0.732) in LD1, which explained 96% of between-group variance. In the RRR, density had the largest coefficient (-8.58), followed by chlorides (-3.50), residual sugar (1.60) and sulphates (1.41). More in-depth results for this are displayed within figures 23 and 24, displaying how results of variable importance amongst different models are compared. This consistency across methods strengthens confidence in the findings and suggests that these chemical properties play a central role in determining wine quality.

Research Question

Can we model and predict wine quality based on physicochemical properties using multivariate statistical methods, and what insights can reduced-rank regression provide about the underlying structure of this relationship?

The LDA successfully classified wines into quality categories with 64.23% accuracy. The RRR model explained 37% of quality variance supplemented by a correlation of 0.609 between actual and predicted values. These results demonstrate that objective chemical measurements contain meaningful information about wine quality and can be used to make reasonably accurate predictions. The reduced-rank regression provided insights into the underlying structure of the relationship. By constraining the coefficient matrix to rank 1, RRR identified a single latent dimension that captures the relationship between chemical properties and quality. This dimension is characterised by negative associations with density and volatile acidity, and positive associations with sulphates and residual sugar. The simplicity of the rank-1 model makes it interpretable while still capturing a meaningful portion of the quality variation. The consistency of results across the three analytical techniques; alcohol content, sulphates, and volatile acidity emerged as important variables in all three analyses and suggests that these chemical properties play a central role in determining wine quality. The convergence of results from PCA, classification (LDA) and regression (RRR) methods indicates that the identified relationships are robust. The negative relationship between volatile acidity and quality is well-established in winemaking. The negative coefficients in both LDA (-0.402) and RRR (-1.15) confirm that lower volatile acidity is associated with higher quality.

Several issues arose during the analysis that warrant discussion. First, the presence of outliers in all predictor variables required careful treatment. Residual sugar showed the highest percentage of outliers followed by chlorides. The decision to use winzorisation rather than deletion of outliers reflects a balance between data retention and analytical robustness. Second, the imbalanced distribution of quality scores (figure 1), with most wines rated 5 or 6, limited the analysis of extreme quality levels suggested that the model performed best at identifying low-quality wines but had more difficulty distinguishing between medium and high-quality wines. Hence, distinction between medium and high quality may depend more on subtle sensory characteristics. Including additional chemical measurements or sensory evaluation data could improve model performance. Third, the strong correlation between some predictor variables (figure 2) raised concerns about multicollinearity. For example, fixed acidity and citric acid showed a strong positive correlation ($r \approx 0.67$), while fixed acidity and pH showed a strong negative correlation ($r \approx -0.68$). However, the use of dimension reduction techniques (PCA) and regularized methods (RRR with rank constraint) helped mitigate these concerns.

The analyses conducted in this project are satisfactory for addressing the stated objectives, with some important caveats. The combination of PCA, LDA, and RRR provided a comprehensive understanding of the wine quality data from multiple analytical perspectives. The consistency of findings across methods strengthens confidence in the results. The use of the 'rrr' package successfully demonstrated its capabilities for multivariate modelling. However, several aspects could be improved in future work. Firstly, the analysis assumed linear relationships between predictors and quality, which may not fully capture the complexity of wine chemistry and wine quality may be influenced by non-linear effects or interactions between chemical properties. Exploring non-linear models or interaction terms could potentially improve predictive accuracy. Secondly, the dataset is limited to red wines from a single region. Expanding the analysis to include wines from multiple regions or comparing red and white wines could provide broader insights.

Limitations of the Project

Several limitations of this project should be acknowledged. The dataset is limited to red wines from a single region, which may limit generalisability.

The analysis assumed linearity between predictor variables and wine-quality, which may not fully capture the complexity of wine chemistry. The moderate predictive accuracy ($R^2 = 0.37$) produced by the RRR model indicates that chemical properties alone cannot fully explain wine quality, as quality is influenced by many factors beyond chemistry, including sensory attributes, vintage conditions, winemaking techniques, and aging. Despite these limitations, the project provided meaningful insights into the determinants of wine quality.

Conclusion

This project successfully demonstrated the application of multivariate data analysis techniques to the Wine Quality dataset, providing comprehensive insights into the relationships between physicochemical properties and expert quality ratings. The use of three complementary techniques; PCA, LDA, and RRR provided a thorough understanding of the data from multiple analytical perspectives, with consistent findings across methods strengthening confidence in the results.

The analysis revealed that wine quality is systematically related to physicochemical properties, with alcohol content, volatile acidity, sulphates, density, and chlorides emerging as the most influential factors. The PCA identified five principal components capturing nearly 80% of chemical variation (79.638%), with PC1 alone explaining 28.512% of variance. The LDA successfully classified wines into quality categories with 64.23% accuracy, with a single discriminant function (LD1) explaining 96.02% of between-group variance. The RRR model proved to be an effective tool for multivariate modelling and explained 37% of quality variance with a correlation of 0.609 between actual and predicted values. The ‘rrr’ package assisted in the production and analysis of results, showing the creation of the PCA and RRR models aswell as plots necessary for visually interpreting key results.

The findings have practical implications for wine production and quality assessment. The identification of key chemical properties associated with quality provides winemakers with clear targets for monitoring and intervention. However, the moderate predictive accuracy of chemical-based models indicates that chemical properties alone cannot fully predict quality and the importance of sensory evaluation by trained experts cannot be undervalued.

References

Chen, L., & Huang, J. Z. (2012). Sparse reduced-rank regression for simultaneous dimension reduction and variable selection. Journal of the American Statistical Association, 107(500), 1533-1545.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems, 47(4), 547-553.

Dua, D., & Graff, C. (2019). UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences. http://archive.ics.uci.edu/ml

Izenman, A. J. (2008). Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning. Springer Science & Business Media.

Johnson, R. A., & Wichern, D. W. (2007). Applied Multivariate Statistical Analysis (6th ed.). Pearson Prentice Hall.

Venables, W. N., & Ripley, B. D. (2002). Modern Applied Statistics with S (4th ed.). Springer.

Wickham, H. (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York.

Appendices

Appendix A: Complete R Code

The complete R code for all analyses is provided below, organized by analysis step.

~~~
---
title: "An Analysis of Red Wine Quality Using Multivariate Statistical Techniques (R Code)"
author: "Peter Misiun, Ryan Sheik"
date: "2025-10-10"
output:
  html_document: default
  pdf_document: default
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
~~~

```{r, warning=FALSE, message=FALSE}
# load neccessary libraries
library(rrr)
library(ggplot2)
library(dplyr)
library(corrplot)
library(MASS)
library(car)
library(knitr)
library(gridExtra)
library(RColorBrewer)
library(viridis)
library(reshape2)
library(e1071)
library(factoextra)
library(scales)
library(tidyr)
library(ggcorrplot)
library(mvnormtest)

```

## **Step 1: Data Preparation**

1.1: Load and Prepare Data
```
```

~~~
```{r}
# load in red wine dataset
redwine_data <- read.csv("winequality-red.csv", sep = ";")

# get predictors and response variables
predictors <- redwine_data[, -which(names(redwine_data) == "quality")]
response <- redwine_data$quality
```
~~~

1.2: Data Structure and Basic Information

~~~
```{r}
# print structure + stats of datasets
str(redwine_data)

# print summary stats of data
summary(redwine_data)
```
~~~

1.3: Outlier Detection and Treatment

~~~
```{r}
# iqr outlier detection
outlier_detection <- function(x) {
  Q1 <- quantile(x, 0.25)
  Q3 <- quantile(x, 0.75)
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  return(x < lower_bound | x > upper_bound)
}

# detect outliers for each pred var
outlier_counts <- sapply(predictors, function(x) sum(outlier_detection(x)))
outlier_summary <- data.frame(
  Variable = names(outlier_counts),
  Outlier_Count = outlier_counts,
  Percentage = round(outlier_counts / nrow(predictors) * 100, 2)
)

# print table
kable(outlier_summary, caption = "Outlier Detection Summary (IQR Method)")

# winsorisation for outliers
winsorize <- function(x, cut = 0.05) {
  cut_point_top <- quantile(x, 1 - cut, na.rm = TRUE)
  cut_point_bottom <- quantile(x, cut, na.rm = TRUE)
~~~

~~~~
  x[x > cut_point_top] <- cut_point_top
  x[x < cut_point_bottom] <- cut_point_bottom
  return(x)
}

predictors_winsorized <- as.data.frame(lapply(predictors, winsorize))
colnames(predictors_winsorized) <- colnames(predictors)

```

**1.4: Scale Predictor Variables**

```{r}
predictors <- predictors_winsorized

# scale predictors
predictors_scaled <- scale(predictors)
predictors_scaled_df <- as.data.frame(predictors_scaled)
```


## **Step 2: Exploratory Data Analysis**

**2.1: Response Variable Analysis**

```{r}
# display quality distribution
par(mar = c(5, 4, 4, 2))
quality_table <- table(response)
barplot(quality_table,
        main = "Distribution of Wine Quality Ratings",
        xlab = "Quality Score",
        ylab = "Frequency",
        col = brewer.pal(length(quality_table), "Set3"),
        border = NA,
        las = 1)
grid(nx = NA, ny = NULL, lty = 2, col = "gray")
```

**2.2: Predictor Variable Distributions**

```{r}
# boxplots for each predictor
par(mfrow = c(2, 6), mar = c(3, 3, 1, 1), mgp = c(1.5, 0.5, 0))
for(i in 1:ncol(predictors)) {
  boxplot(predictors[, i],
          main = names(predictors)[i],
~~~~

```
          col = "lightblue",
          border = "darkblue",
          pch = 19,
          cex = 0.6,
          ylab = "",
          cex.main = 0.8)
  grid(nx = NA, ny = NULL, lty = 2, col = "gray")
}
par(mfrow = c(1, 1))


# density plots
par(mfrow = c(4, 3), mar = c(3, 3, 1, 1), mgp = c(1.5, 0.5, 0))
for(i in 1:ncol(predictors)) {
  var_data <- predictors[, i]
  dens <- density(var_data)
  # get density values
  hist_info <- hist(var_data, plot = FALSE, breaks = 20)
  y_max <- max(hist_info$density, dens$y,
               dnorm(mean(var_data), mean(var_data), sd(var_data))) * 1.05
  hist(var_data,
       probability = TRUE,
       main = names(predictors)[i],
       xlab = "",
       ylab = "Density",
       col = "lightgreen",
       border = "white",
       breaks = 20,
       ylim = c(0, y_max))
  lines(dens, col = "darkgreen", lwd = 2)
  # include normal curve
  x_norm <- seq(min(var_data), max(var_data), length = 100)
  y_norm <- dnorm(x_norm, mean = mean(var_data), sd = sd(var_data))
  lines(x_norm, y_norm, col = "red", lty = 2, lwd = 1.5)


  legend("topright",
         legend = c("Actual", "Normal"),
         col = c("darkgreen", "red"),
         lty = c(1, 2),
         lwd = c(2, 1.5),
         cex = 0.7,
         bty = "n")
}
par(mfrow = c(1, 1))
```

~~~
**2.3: Relationships Between Variables**

```{r}
# correlation matrix
cor_matrix <- cor(predictors)
par(mar = c(0, 0, 3, 0))
corrplot(cor_matrix,
        method = "color",
        type = "upper",
        order = "hclust",
        tl.cex = 0.8,
        tl.srt = 45,
        tl.col = "black",
        col = colorRampPalette(c("#6D9EC1", "white", "#E46726"))(100),
        main = "Correlation Matrix of Wine Predictors",
        mar = c(0, 0, 2, 0),
        addCoef.col = "black",
        number.cex = 0.7)
```

**2.4 Descriptive Statistics**

```{r}
# print mean vectors
print(round(colMeans(predictors), 3))

# predictor correlation with quality
quality_cor <- cor(predictors, response)
quality_cor_df <- data.frame(
  Variable = rownames(quality_cor),
  Correlation = quality_cor[,1]
)
quality_cor_df <- quality_cor_df[order(abs(quality_cor_df$Correlation), decreasing = TRUE), ]

# display correlation of predictors with wine quality
ggplot(quality_cor_df, aes(x = reorder(Variable, Correlation), y = Correlation, fill = Correlation > 0)) +
  geom_col(show.legend = FALSE) +
  coord_flip() +
  scale_fill_manual(values = c("darkred", "darkgreen")) +
  theme_minimal() +
  labs(
    title = "Predictor Correlation with Wine Quality",
    x = NULL,
~~~

```
    y = "Correlation Coefficient"
  ) +
  theme(
    plot.title = element_text(face = "bold", size = 14),
    panel.grid.major.y = element_line(color = "gray90")
  )
```

## **Step 3: Data Transformation**

3.1: Apply Box-Cox transformation to skewed variables

From previous plots, we can see that data in some classes is significantly skewed. These classes can be transformed into more normal data using the Box-Cox transformation.

```{r}
# get variables needing to be transformed
predictors_transformed <- predictors
transformed_vars <- c()

for(i in 1:ncol(predictors)) {
  if(abs(skewness(predictors[,i])) > 1) {
    transformed_vars <- c(transformed_vars, names(predictors)[i])

    # apply box cox transformation
    bc <- boxcox(predictors[,i] ~ 1, plotit = FALSE)
    lambda <- bc$x[which.max(bc$y)]
    if(lambda == 0) {
      predictors_transformed[,i] <- log(predictors[,i] + abs(min(predictors[,i])) + 1)
    } else {
      predictors_transformed[,i] <- ((predictors[,i] + abs(min(predictors[,i])) + 1)^lambda - 1) / lambda
    }
  }
}

# calc skewness before and after
skewness_comparison <- data.frame(
  Variable = names(predictors),
  Skewness_Before = sapply(predictors, skewness),
  Skewness_After = sapply(as.data.frame(predictors_transformed), skewness),
  Transformed = names(predictors) %in% transformed_vars
)

skewness_comparison$Change <- skewness_comparison$Skewness_After - skewness_comparison$Skewness_Before
skewness_comparison$Abs_Skewness_Before <- abs(skewness_comparison$Skewness_Before)
skewness_comparison$Abs_Skewness_After <- abs(skewness_comparison$Skewness_After)
```

~~~~
skewness_comparison <- skewness_comparison %>%
  mutate(across(where(is.numeric), ~round(., 3)))

# print table
kable(skewness_comparison,
      caption = "Skewness Before and After Box-Cox Transformation",
      align = "lcccccc")
```

**3.2: Compare Results from Shapiro-Wilk Normality Test**

```{r}
set.seed(123)
sample_idx <- sample(1:nrow(predictors), min(5000, nrow(predictors)))

# after transformation
predictors_transformed_sample <- predictors_transformed[sample_idx, ]
predictors_transformed_t <- t(predictors_transformed_sample)
mshapiro_result_after <- mshapiro.test(predictors_transformed_t)

# display after transformation box cox
print(mshapiro_result_after)
```

## **Step 4: Principal Component Analysis (PCA)**

**4.1: PCA Implementation - Method Comparison**

4.1.1: Traditional PCA using prcomp()

```{r}
# use prcomp to perform pca
pca_standard <- prcomp(predictors, scale. = TRUE)

# Generate scree plot
# display scree plot with line at y=1 for eigenvalues
screeplot(pca_standard, type = "barplot", main = "Scree Plot using prcomp()")
abline(h = 1, lty = 2, col = "red", lwd = 2)

# get eigenvalues
eigenvalues_std <- pca_standard$sdev^2
```

4.1.2: PCA using RRR Package
~~~~

~~~
```{r, warning=FALSE, warning=FALSE}

# use rrr to perform pca on scaled data

pca_rrr <- rrr(predictors_scaled_df, predictors_scaled_df, type = "pca", rank = "full")


# calc var explained metrics

gof <- pca_rrr$goodness_of_fit

cumulative_explained <- 1 - gof

incremental_explained <- c(cumulative_explained[1], diff(cumulative_explained))


# convert to percent

variance_explained_rrr <- incremental_explained * 100

cumulative_variance_rrr <- cumulative_explained * 100


# scree data frame

scree_data_rrr <- data.frame(

  PC = 1:length(variance_explained_rrr),

  Eigenvalue = eigenvalues_std,

  Variance = variance_explained_rrr,

  Cumulative = cumulative_variance_rrr

)


# rounding numbers

scree_data_rrr <- scree_data_rrr %>%

  mutate(

    Eigenvalue = round(Eigenvalue, 3),

    Variance = round(Variance, 3),

    Cumulative = round(Cumulative, 3)

  )


# print table

kable(

  scree_data_rrr,

  caption = "Eigenvalues and Variance Explained (rrr PCA)",

  align = "c"

)

```
~~~

4.2: Component Selection and Justification

~~~
```{r}

# determine least amount of pcs for a cumulative var of 75% or more

n_components_75_rrr <- which(cumulative_variance_rrr >= 75)[1]


#print info

cat("Number of PCs needed to explain 75% variance:", n_components_75_rrr, "\n")


n_components_75 <- n_components_75_rrr
~~~

```
cat("\nSelected components for analysis:", n_components_75, "PCs (explaining",
    round(cumulative_variance_rrr[n_components_75], 1), "% variance)\n")
```
```

4.3: PCA Loadings Interpretation

```
```{r}
# display loadings for first 5 pc
loadings_rrr <- pca_rrr$PC[, 1:5]
loadings_rrr_subset <- loadings_rrr[1:11, 1:5]

loadings_rrr_df <- data.frame(
  Variable = colnames(redwine_data)[1:11],
  loadings_rrr_subset,
  row.names = NULL
)

# round loadings
loadings_rrr_df <- loadings_rrr_df %>%
  mutate(across(-Variable, ~round(., 4)))

# print loadings table
kable(
  loadings_rrr_df,
  caption = "PCA Loadings for First 5 Principal Components (rrr)",
  align = "lcccccc"
)
```
```

4.4: PCA Visualisation

4.4.1: Variance Visualization

```
```{r}
# display scree plot for rrr pca
ggplot(scree_data_rrr, aes(x = PC, y = Variance)) +
  geom_line(color = "darkgreen", size = 1) +
  geom_point(color = "darkgreen", size = 2) +
  geom_bar(stat = "identity", alpha = 0.3, fill = "lightgreen", color = "darkgreen", size = 0.3) +
  labs(title = "Scree Plot (rrr PCA)",
       x = "Principal Component",
       y = "Percentage of Variance Explained") +
  theme_minimal() +
  theme(plot.title = element_text(face = "bold", hjust = 0.5),
        panel.grid.major = element_line(color = "gray", linewidth = 0.5),
        panel.grid.minor = element_line(color = "lightgray", linewidth = 0.25))
```

```
# display cumulative variance plot
ggplot(scree_data_rrr, aes(x = PC, y = Cumulative)) +
  geom_line(color = "red", size = 1) +
  geom_point(color = "red", size = 2) +
  geom_hline(yintercept = 75, linetype = "dashed", color = "blue", size = 1) +
  labs(title = "Cumulative Variance (rrr PCA)",
       x = "Principal Component",
       y = "Cumulative Percentage of Variance") +
  theme_minimal() +
  scale_x_continuous(limits = c(0, NA)) +
  scale_y_continuous(limits = c(0, 100)) +
  theme(plot.title = element_text(face = "bold", hjust = 0.5),
        panel.grid.major = element_line(color = "gray", linewidth = 0.5),
        panel.grid.minor = element_line(color = "lightgray", linewidth = 0.25))
```

4.4.2: Pairwise Plots with Quality Groups

~~~
```{r, warning=FALSE, message=FALSE}
# create quality classes
redwine_data$quality_class <- cut(redwine_data$quality,
                                  breaks = c(0, 5, 6, 10),
                                  labels = c("Low", "Medium", "High"))

# set variables as needed
pca_df <- as.data.frame(pca_standard$x[, 1:2])
colnames(pca_df) <- c("PC1", "PC2")
pca_df$Quality <- as.factor(redwine_data$quality_class)

fviz_pca_biplot(pca_standard, label = "var", habillage = as.factor(redwine_data$quality_class),
                addEllipses = TRUE, ellipse.level = 0.95,
                title = "prcomp() PCA biplot (95% ellipses)")

# create pairwise plots using rrr package, with 1 flipped on y axis
pairwise_plot(predictors_scaled_df, redwine_data$quality_class, type = "pca", rank = "full", k = 0, interactive = FALSE) + scale_y_reverse()
```
~~~

4.4.3: Rank Trace Analysis

~~~
```{r, warning=FALSE, message=FALSE}
# rank trace from rrr and reverse axes
rank_trace(predictors_scaled_df, predictors_scaled_df, type = "pca") +
  scale_x_reverse() +
  scale_y_reverse()
```
~~~

4.5: PCA Inference

~~~
```{r}
# prep pca data as needed
pca_df <- as.data.frame(pca_standard$x[, 1:5])
colnames(pca_df) <- paste0("PC", 1:5)
pca_df$Quality <- as.factor(response)
pca_df$Quality_Score <- as.numeric(response)

# anova, kruskal and cor combined
combined_summary <- lapply(1:5, function(i) {
  pc <- paste0("PC", i)

    aov_p <- summary(aov(as.formula(paste0(pc, " ~ Quality")), data = pca_df))[[1]][["Pr(>F)"]][1]

  kw <- kruskal.test(as.formula(paste0(pc, " ~ Quality")), data = pca_df)

  cor_test <- cor.test(pca_df[[pc]], pca_df$Quality_Score)

  data.frame(
    Component = pc,
    ANOVA_p = round(aov_p, 5),
    KW_p = round(kw$p.value, 5),
    Pearson_r = round(cor_test$estimate, 3),
    Corr_p = round(cor_test$p.value, 5)
  )
}) %>%
  bind_rows() %>%
  mutate(
    Significance = case_when(
      Corr_p < 0.001 ~ "***",
      Corr_p < 0.01 ~ "**",
      Corr_p < 0.05 ~ "*",
      TRUE ~ ""
    )
  )

# display summary of results
combined_summary

```
~~~

## **Step 5: Linear Discriminant Analysis (LDA)**

5.1: LDA Implementation and Model Fitting

5.1.1: Data Preparation for LDA

~~~
```{r}
# check distribution of classes
quality_table <- table(redwine_data$quality_class)
print(quality_table)

# scale transformed pred for lda
predictors_transformed_scaled <- scale(predictors_transformed)
predictors_transformed_scaled_df <- as.data.frame(predictors_transformed_scaled)

# create combined dataset for lda
lda_data <- cbind(predictors_transformed_scaled_df, quality_class = redwine_data$quality_class)
```
~~~

5.1.2: LDA Model Fitting

~~~
```{r}
# perform lda
lda_result <- lda(quality_class ~ ., data = lda_data)

# print results
print(lda_result)

# lda preds
lda_predict <- predict(lda_result, lda_data)

# prep data
lda_df <- data.frame(
  LD1 = lda_predict$x[, 1],
  LD2 = lda_predict$x[, 2],
  Quality = redwine_data$quality_class
)

# variance explained by linear discriminants
ld_proportion <- lda_result$svd^2 / sum(lda_result$svd^2)
ld_df <- data.frame(LD = paste0("LD", 1:length(ld_proportion)), Proportion = ld_proportion)

# display the variance explained by linear discriminants
ggplot(ld_df, aes(x = LD, y = Proportion)) +
  geom_bar(stat = "identity", fill = "skyblue", width = 0.7) +
  labs(title = "Variance Explained by Linear Discriminants",
       y = "Proportion of Variance", x = "Linear Discriminants") +
~~~

~~~~
  theme_minimal(base_size = 14) +
  theme(plot.title = element_text(face = "bold")) +
  coord_cartesian(xlim = c(0.5, length(ld_proportion) + 0.5))

```

**5.2: Model Performance Evaluation**

5.2.1: Classification Accuracy

```{r}
# get confusion matrix
confusion_matrix <- table(Actual = redwine_data$quality_class,
                         Predicted = lda_predict$class)

# calc row percents
cm_df <- as.data.frame(confusion_matrix)
colnames(cm_df) <- c("Actual", "Predicted", "Count")

cm_df <- cm_df %>%
  group_by(Actual) %>%
  mutate(Percentage = Count / sum(Count) * 100,
         Label = paste0(Count, "\n(", round(Percentage, 1), "%)")) %>%
  ungroup()

ggplot(cm_df, aes(x = Predicted, y = Actual, fill = Count)) +
  geom_tile(color = "grey80", width = 0.9, height = 0.9) +
  geom_text(aes(label = Label), size = 4, color = "black") +
  scale_fill_viridis(option = "C", direction = -1) +
  labs(title = "LDA Confusion Matrix Heatmap",
       subtitle = "Count (Row Percentage)") +
  theme_minimal(base_size = 14) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        panel.grid = element_blank(),
        plot.title = element_text(face = "bold"),
        plot.subtitle = element_text(size = 11, color = "gray40"))

# calc accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("\nLDA Classification Accuracy:", round(accuracy * 100, 2), "%\n")

```

5.2.2: Variable Importance

```{r}
~~~~

```
# lda loadings
lda_loadings <- lda_result$scaling

# use only LD1
variable_importance <- abs(lda_loadings[, 1])
sorted_importance <- sort(variable_importance, decreasing = TRUE)

# get variable important of all variables
var_importance_df <- data.frame(
  Variable = names(sorted_importance),
  Importance = sorted_importance
)

# plot variable importance
ggplot(var_importance_df, aes(x = reorder(Variable, Importance), y = Importance)) +
  geom_bar(stat = "identity", fill = "tomato") +
  coord_flip() +
  labs(title = "Variable Importance in LDA (LD1 Only)",
       x = "", y = "Importance") +
  theme_minimal(base_size = 14) +
  theme(plot.title = element_text(face = "bold"))
```

5.3: LDA Visualization and Interpretation

5.3.1: Density Plots by Discriminant

```{r}
# display ld1 density separation
ggplot(lda_df, aes(x = LD1, fill = Quality)) +
  geom_density(alpha = 0.5) +
  labs(title = "LDA - Wine Quality Separation by LD1",
       x = "Linear Discriminant 1", y = "Density") +
  theme_minimal() +
   theme(plot.title = element_text(face = "bold", hjust = 0.5))

# display ld2 density separation
ggplot(lda_df, aes(x = LD2, fill = Quality)) +
  geom_density(alpha = 0.5) +
  labs(title = "LDA - Wine Quality Separation by LD2",
       x = "Linear Discriminant 2", y = "Density") +
  theme_minimal() +
   theme(plot.title = element_text(face = "bold", hjust = 0.5))
```

5.3.2: Scatter Plot Visualization

```{r}
# calc centroids
```

~~~~
centroids <- lda_df %>%
  group_by(Quality) %>%
  summarise(LD1 = mean(LD1), LD2 = mean(LD2))

# print lda scatter plot
enhanced_scatter <- ggplot(lda_df, aes(x = LD1, y = LD2, color = Quality)) +
  geom_point(alpha = 0.4, size = 1.5) +
  stat_ellipse(aes(fill = Quality), geom = "polygon", alpha = 0.1, level = 0.95) +
  geom_point(data = centroids, aes(x = LD1, y = LD2, color = Quality),
             size = 5, shape = 18, show.legend = FALSE) +
  geom_text(data = centroids, aes(x = LD1, y = LD2, label = Quality),
            color = "black", size = 4, fontface = "bold", nudge_y = 0.3) +
  labs(title = "LDA Wine Quality Classification (95% Confidence Ellipses)",
       subtitle = "Diamonds represent group centroids",
       x = paste0("Linear Discriminant 1 (", round(ld_proportion[1] * 100, 1), "%)"),
       y = paste0("Linear Discriminant 2 (", round(ld_proportion[2] * 100, 1), "%)"),
       color = "Quality Class",
       fill = "Quality Class") +
  theme_minimal() +
  theme(plot.title = element_text(face = "bold", hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5, color = "gray40"))

print(enhanced_scatter)

# use pairwise_plot function from rrr flipped on x and y axis
pairwise_plot(predictors_transformed_scaled_df, redwine_data$quality_class, type = "lda") + scale_x_reverse() + scale_y_reverse()
```



## **Step 6: Reduced-Rank Regression (RRR)**

**6.1: RRR Model Implementation**

6.1.1: Data Preparation and Model Fitting

```{r}
# prep data
predictors_transformed_matrix <- as.matrix(predictors_transformed)
response_matrix <- as.matrix(redwine_data$quality)



# perform rrr using rrr package, rank = 1 and response is univariate
rrr_result <- rrr(x = predictors_transformed_matrix, y = response_matrix, rank = 1)

print(rrr_result)
```
~~~~

6.1.2: Model Coefficients and Predictions

~~~
```{r}
# get coefficients + fitted vals
rrr_coefficients <- rrr_result$C
rrr_mean <- rrr_result$mean

# convert rrr_mean to numeric and calculate fitted values
rrr_fitted <- as.numeric(rrr_mean) + predictors_transformed_matrix %*% t(rrr_coefficients)
```
~~~

6.2: Model Performance Evaluation

~~~
```{r}
# model performance metrics
ss_total <- sum((response_matrix - mean(response_matrix))^2)
ss_residual <- sum((response_matrix - rrr_fitted)^2)
rrr_r_squared <- 1 - (ss_residual / ss_total)
rrr_mse <- mean((response_matrix - rrr_fitted)^2)

# print info
cat("R-squared:", round(rrr_r_squared, 4), "\n")
cat("MSE:", round(rrr_mse, 4), "\n")
```
~~~

6.3: RRR Results Visualization

6.3.1: Coefficient Analysis

~~~
```{r}
# prep coefficient data
rrr_coef_df <- data.frame(
  Variable = colnames(predictors_transformed_matrix),
  Coefficient = as.numeric(rrr_coefficients),
  Abs_Coefficient = abs(as.numeric(rrr_coefficients))
) %>%
  arrange(desc(Abs_Coefficient))

# display coefficient importance plot
ggplot(rrr_coef_df, aes(x = reorder(Variable, Abs_Coefficient),
                        y = Coefficient,
                        fill = Coefficient > 0)) +
  geom_bar(stat = "identity", alpha = 0.85, width = 0.7) +
  scale_fill_manual(values = c("darkred", "darkgreen"), labels = c("Negative", "Positive")) +
  coord_flip() +
  labs(title = "Reduced-Rank Regression Coefficients",
~~~

```
      x = "Predictor Variables",
      y = "Coefficient Value",
      fill = "Sign") +
  theme_minimal(base_size = 13) +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5, color = "gray40"),
        legend.position = "bottom")

# use coefficient to get variable importance
rrr_importance <- data.frame(
  Variable = rrr_coef_df$Variable,
  Importance = rrr_coef_df$Abs_Coefficient
) %>%
  arrange(desc(Importance))

# display variable importance
ggplot(head(rrr_importance, 11), aes(x = reorder(Variable, Importance), y = Importance)) +
  geom_bar(stat = "identity", fill = "orange", alpha = 0.85, width = 0.7) +
  coord_flip() +
  labs(title = "Variables by RRR Coefficient Magnitude",
      subtitle = "Higher absolute coefficient = stronger influence on wine quality",
      x = "Predictor Variables",
      y = "Absolute Coefficient Value") +
  theme_minimal(base_size = 13) +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5, color = "gray40"))
```

6.3.2: Prediction Accuracy Visualisation

~~~
```{r, warning=FALSE, message=FALSE}
# prep prediction data
prediction_df <- data.frame(
  Actual = response_matrix[, 1],
  Predicted = rrr_fitted[, 1],
  Quality_Class = redwine_data$quality_class
)

# actual vs pred plot
ggplot(prediction_df, aes(x = Actual, y = Predicted, color = Quality_Class)) +
  geom_point(alpha = 0.7, size = 2.8) +
  geom_abline(aes(intercept = 0, slope = 1, linetype = "Perfect Prediction (y = x)"),
              color = "black", linewidth = 1) +
  geom_smooth(aes(linetype = "Fitted Regression Line"),
              method = "lm", se = FALSE, color = "blue", linewidth = 1) +
  scale_linetype_manual(values = c("Perfect Prediction (y = x)" = "dashed",
~~~

```
                              "Fitted Regression Line" = "dotted")) +
  labs(title = "Actual vs Predicted Wine Quality",
       subtitle = paste("Correlation =", round(cor(prediction_df$Actual, prediction_df$Predicted), 5)),
       x = "Actual Quality Score",
       y = "Predicted Quality Score",
       color = "Quality Class",
       linetype = "") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5, color = "gray40"),
        legend.position = "bottom",
        legend.text = element_text(size = 8),
        legend.title = element_text(size = 8),
        legend.key.size = unit(1.2, "lines"))
```

6.3.3: Predicted Quality Comparison (Violin Plot)

```{r}
ggplot(prediction_df, aes(x = Quality_Class, y = Predicted, fill = Quality_Class)) +
  geom_violin(trim = FALSE, alpha = 0.7) +
  geom_boxplot(width = 0.1, outlier.shape = NA, alpha = 0.5, color = "black") +
  theme_minimal(base_size = 13) +
  labs(
    title = "Predicted Wine Quality by Class",
    x = "Wine Quality Class",
    y = "Predicted Quality Score",
    fill = "Quality Class"
  ) +
  theme(
    plot.title = element_text(face = "bold", hjust = 0.5, size = 14),
    legend.title = element_text(face = "bold")
  )
```

6.4: Residual Analysis

```{r}
# calc residuals
prediction_df$Residuals <- prediction_df$Actual - prediction_df$Predicted

# residuals vs predicted plot
p1 <- ggplot(prediction_df, aes(x = Predicted, y = Residuals, color = Quality_Class)) +
  geom_point(alpha = 0.7, size = 2.5) +
  geom_hline(yintercept = 0, color = "black", linetype = "dashed", linewidth = 1) +
```

~~~~
  labs(title = "Residuals vs Predicted Values",
       x = "Predicted Quality",
       y = "Residuals",
       color = "Quality Class") +
  theme_minimal(base_size = 13) +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        legend.position = "bottom",
        legend.text = element_text(size = 9),
        legend.title = element_text(size = 10),
        legend.key.size = unit(0.6, "lines"))

# residuals distribution plot
p2 <- ggplot(prediction_df, aes(x = Residuals, fill = Quality_Class)) +
  geom_density(alpha = 0.6) +
  geom_vline(xintercept = 0, color = "black", linetype = "dashed", linewidth = 1) +
  labs(title = "Distribution of Residuals",
       x = "Residuals",
       y = "Density",
       fill = "Quality Class") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        legend.position = "bottom",
        legend.text = element_text(size = 9),
        legend.title = element_text(size = 10),
        legend.key.size = unit(1, "lines"))

# display residual plots side by side
grid.arrange(p1, p2, ncol = 2)
```



## **Step 7: Variable Importance Comparison**



**7.1: Comparison (PCA vs LDA vs RRR)**


```{r}
# create dataframes to store required values for each technique
pca_importance <- data.frame(
  Variable = colnames(predictors),
  PCA_Importance = rowSums(abs(loadings_rrr[, 1:5])),
  Method = "PCA"
)

lda_importance <- data.frame(
  Variable = rownames(lda_result$scaling),
  LDA_Importance = abs(lda_result$scaling[, 1]),
~~~~

```
  Method = "LDA"
)

rrr_importance <- data.frame(
  Variable = colnames(predictors_transformed_matrix),
  RRR_Importance = abs(as.numeric(rrr_coefficients)),
  Method = "RRR"
)

# normalise importance scores
comparison_df <- data.frame(
  Variable = pca_importance$Variable,
  PCA = pca_importance$PCA_Importance / max(pca_importance$PCA_Importance),
  LDA = lda_importance$LDA_Importance / max(lda_importance$LDA_Importance),
  RRR = rrr_importance$RRR_Importance / max(rrr_importance$RRR_Importance)
)

# get avg importance for each var across techniques
comparison_df$Average_Importance <- rowMeans(comparison_df[, c("PCA", "LDA", "RRR")])
comparison_df <- comparison_df[order(comparison_df$Average_Importance, decreasing = TRUE), ]



comparison_z <- comparison_df %>%
  mutate(
    PCA_Z = as.numeric(scale(PCA)),
    LDA_Z = as.numeric(scale(LDA)),
    RRR_Z = as.numeric(scale(RRR))
  )

# calc avg z score
comparison_z$Average_Z_Importance <- rowMeans(comparison_z[, c("PCA_Z", "LDA_Z", "RRR_Z")])
comparison_z <- comparison_z[order(comparison_z$Average_Z_Importance, decreasing = TRUE), ]

# combine scores
comparison_z_melt <- melt(
  comparison_z,
  id.vars = "Variable",
  measure.vars = c("PCA_Z", "LDA_Z", "RRR_Z", "Average_Z_Importance"),
  variable.name = "Method",
  value.name = "Z_Importance"
)

# display importance after z score normalisation
ggplot(comparison_z_melt[comparison_z_melt$Method != "Average_Z_Importance", ],
       aes(x = reorder(Variable, Z_Importance), y = Z_Importance, fill = Method)) +
  geom_bar(stat = "identity", position = "dodge") +
  coord_flip() +
  labs(title = "Variable Importance Comparison (Z-Score Normalisation)",
       subtitle = "Z-scored importance across PCA, LDA, and RRR",
       x = "Predictor Variables",
       y = "Z-Score Normalized Importance",
       fill = "Method") +
  theme_minimal(base_size = 13) +
  scale_fill_viridis(discrete = TRUE) +
  theme(plot.title = element_text(face = "bold", hjust = 0.5),
        plot.subtitle = element_text(color = "gray40", hjust = 0.5))
```

~~~
**7.2: Comparison (Overall Importance)**


```{r}
# display overall importance of variables
ggplot(comparison_z, aes(x = reorder(Variable, Average_Z_Importance), y = Average_Z_Importance, fill = Variable)) +
  geom_bar(stat = "identity", alpha = 0.85, width = 0.7) +
  coord_flip() +
  labs(title = "Variables Ranked by Influence",
       subtitle = "Average Z-Score Importance Across PCA, LDA, and RRR",
       x = "Predictor Variable",
       y = "Average Z-Score Importance") +
  scale_fill_viridis(discrete = TRUE) +
  theme_minimal(base_size = 13) +
  theme(plot.title = element_text(face = "bold", hjust = 0.5),
        plot.subtitle = element_text(color = "gray40", hjust = 0.5),
        legend.position = "none")
~~~

## **Extra Plot for Report**

```{r}
# get info for predictors in high qual wines using winsorized data
high_quality_data <- redwine_data[redwine_data$quality >= 7, ]
medium_low_quality_data <- redwine_data[redwine_data$quality < 7, ]

# Use the winsorized predictors that match the original dataset rows
high_quality_winsorized <- predictors_winsorized[redwine_data$quality >= 7, ]
medium_low_quality_winsorized <- predictors_winsorized[redwine_data$quality < 7, ]

# calc mean dif using winsorized data
variable_differences <- data.frame(
  Variable = colnames(predictors_winsorized),
  High_Quality_Mean = colMeans(high_quality_winsorized),
  Other_Mean = colMeans(medium_low_quality_winsorized)
)
```

```
variable_differences$Difference <- variable_differences$High_Quality_Mean - variable_differences$Other_Mean

variable_differences$Abs_Difference <- abs(variable_differences$Difference)

variable_differences <- variable_differences[order(variable_differences$Abs_Difference, decreasing = TRUE), ]


# display info of differences

ggplot(head(variable_differences, 11),

       aes(x = reorder(Variable, Abs_Difference), y = Difference,

           fill = Difference > 0)) +

  geom_col(show.legend = TRUE) +

  coord_flip() +

  scale_fill_manual(

    values = c("darkred", "darkgreen"),

    labels = c("Lower in High-Quality", "Higher in High-Quality"),

    name = "Direction"

  ) +

  labs(

    title = "Mean Difference Comparison: High-Quality vs Other Wines",

    x = "Chemical Properties",

    y = "Mean Difference"

  ) +

  theme_minimal() +

  theme(

    plot.title = element_text(face = "bold", hjust = 0.5, size = 14),

    axis.title.x = element_text(size = 12, face = "bold"),

    axis.title.y = element_text(size = 12, face = "bold"),

    axis.text = element_text(size = 10),

    panel.grid.major = element_line(color = "gray85", linewidth = 0.5),

    panel.grid.minor = element_blank(),

    legend.title = element_text(face = "bold")

  )
```
```
```