**Analysis of Predictive Models with the Psychochemical Properties of Wine Versus Wine Quality**

**ADTA 5550 Project**

**Azhan Saleem**

**ABSTRACT**

The wine industry, a significant contributor to the economy, continually seeks advancements in quality assessment. Leveraging data analytics, this project explores the predictive capabilities of machine learning models based on the physicochemical properties of wines. The dataset, sourced from the UCI Machine Learning Repository, captures detailed information on the chemical composition of red and white wines. The primary questions addressed are whether wine quality can be accurately predicted, and which variables exert the most influence on quality. Three models, a multilayer neural network, linear regression, and random forest, were constructed and evaluated.

The exploratory data analysis (EDA) phase unveiled nuances in the dataset's physicochemical features. The subsequent modeling involved meticulous preparation of data, including the extraction and scaling of numeric features. The neural network model, with early stopping and regularization techniques, showcased a detailed architecture, achieving an accuracy of approximately 56%. Linear regression and random forest models, assessed using R-squared, yielded values around 28% and 46%, respectively. Despite challenges in predicting extreme quality values, the neural network emerged as the most effective model.

Further discussions highlight the necessity for a more extensive dataset and equal representation of red and white wines for comprehensive analyses. The report concludes by emphasizing the industry's potential benefit from utilizing the neural network model for analyzing qualities 5 to 7 while underscoring the importance of additional research to enhance predictive capabilities across a broader spectrum of wine qualities.

**Table of Contents**

**1. Introduction 1**

Why This Data is Important1

Analytics Questions 1

Can Wine Quality be Predicted?1

What Variables Impact Wine Quality the Most?1

How Can Predictive Models Benefit the Wine Industry?1

Plan of Analysis 2

**2. EDA/ Data Understanding & Data Preparation3**

Univariate Analysis5

Bivariate Analysis7

Multivariate Analysis 10

Variable Significance12

**3. Methods / Modeling** **14**

Multilayer Neural Network Model15

Preparation for Neural Network Model Construction15

Data Splitting for Neural Network Model16

Feature and Target Extraction for Neural Network Model16

Neural Network Model Building17

Training History Visualization for Neural Network Model20

Results and Interpretations for Neural Network Model21

Linear Regression Model22

Preparation for Linear Regression Model Construction22

Data Splitting for Linear Regression Model22

Feature and Target Extraction and Model Building for Linear Regression Model23

Results and Interpretations for Linear Regression Model25

Random Forest Model26

Preparation for Random Forest Model Construction26

Data Splitting for Random Forest Model26

Random Forest Model Building27

Results and Interpretations for Random Forest Model28

**4. Evaluation / Discussion** **29**

Training Evaluation30

Model Evaluation30

Conclusion33

**5. Works Cited** **34**

**1. Introduction:**

"Wine is the most delightful and most civilized thing in the world." - Ernest Hemingway

The wine industry would agree with what Ernest Heminway said as the wine industry has held a significant economic impact by generating over $200 billion annually within the United States (Balter, 2017) and it has continuously demanded improvements in quality assessment and refinement. Considering this, our project focused on leveraging data analytics to predict wine quality based on key physicochemical properties.

**1.1 Why This Data is Important:**

Our dataset, sourced from the IC Irvine Machine Learning Repository, captured detailed information on the chemical composition of red and white wines, which provides a comprehensive overview of their physicochemical properties. This data would be crucial for winemakers, as it would serve as a bridge between the scientific understanding of wine and its characteristics and the subjective enjoyment from consumers. By analyzing this dataset, we aimed to discover patterns and correlations that would give wine industry insiders actionable insights to enhance overall wine quality.

**1.2** **Analytics Questions:**

*Can Wine Quality be Predicted and Which Model Works Best?*

The central question driving our analysis was whether we could predict the quality of wine accurately. By exploring the intricate relationships between features like acidity, alcohol content, residual sugar, etc., we developed three different machine learning models to predict the overall quality of a wine sample and wanted to see which produced the most accurate predictions. This exploration was grounded in the hypothesis that the chemical composition of wine significantly influences its perceived quality.

*What Variables Impact Wine Quality the Most?*

Beyond predicting wine quality, our analysis was to seek and identify which physicochemical properties hold the most influence on wine quality. Through correlation and model variable importance analyses, we sought to determine which factors, such as acidity, sulfur content, alcohol levels, etc., affected the quality of a wine the most.

*How Can Predictive Models Benefit the Wine Industry?*

Beyond academic curiosity, our project held a practical objective: to provide actionable insights for winemakers. By developing predictive models capable of assessing and predicting wine quality, we aspired to equip winemakers with a valuable tool for quality control and continuous improvement. The results of our models could potentially improve decision-making processes in vineyards and wineries.

**1.3** **Plan of Analysis:**

Our approach consisted of a comprehensive exploratory data analysis (EDA) to understand the dataset's nuances, cleaning, and preprocessing steps to ensure data quality, and the development of predictive models using machine learning techniques. By conducting in-depth analyses and employing neural network models, we sought out the goal of contributing to the improvement of wine quality assessment practices.

In conclusion**,** this project not only addressed the scientific curiosity regarding wine quality prediction but also addressed the practical needs of the wine industry. The subsequent sections will delve into the specifics of our data, present our findings, and discuss the implications and potential applications of our predictive models.

**2. Data understanding/EDA:**

As mentioned above, our objective was to analyze and understand the relationships between wines’ psychochemical features and the perceived quality of these wines. To begin our analysis, we conducted exploratory data analysis (EDA) to help us truly understand the structure and scope of the data, to set the stage for additional in-depth analyses, and to help us to determine the design and structure our predictive models. We used python-based programs such as Jupyter and Google Colab for our EDA and continued to use these programs for the remainder of the project.

Looking at the summary of our dataset, “combined\_winequality”, we found that our dataset contained 6,497 observations and 13 variables; all that relate to the wines’ psychochemical properties, the type of wine, red or white, and the wines’ qualities. These variables consisted of:

**Fixed Acidity:** Fixed acidity represents the total concentration of acids (usually tartaric acid) in the wine. It contributes to the wine's tartness and stability.

**Volatile Acidity:** Volatile acidity is the concentration of volatile acids (e.g., acetic acid) in the wine. Higher levels of volatile acidity can give wine undesirable off-flavors, often described as vinegar-like.

**Citric Acid:** Citric acid is a weak organic acid found naturally in citrus fruits. It can be added to wine to increase acidity or impart a citrusy flavor.

**Residual Sugar:** Residual sugar is the amount of sugar that remains unfermented in the wine after the fermentation process. It affects the wine's sweetness.

**Chlorides:** Chloride ions can be present in wine as a result of winemaking practices. High chloride levels can contribute to a salty or briny taste.

**Free Sulfur Dioxide:** Free sulfur dioxide (SO2) is used as a preservative in winemaking to prevent oxidation and microbial spoilage.

**Total Sulfur Dioxide:** Total sulfur dioxide includes both free and bound forms of sulfur dioxide. It is another measure of the wine's sulfite content and its ability to prevent spoilage.

**Density:** Density is a measure of the wine's mass per unit volume. It can be used to calculate the alcohol content of the wine.

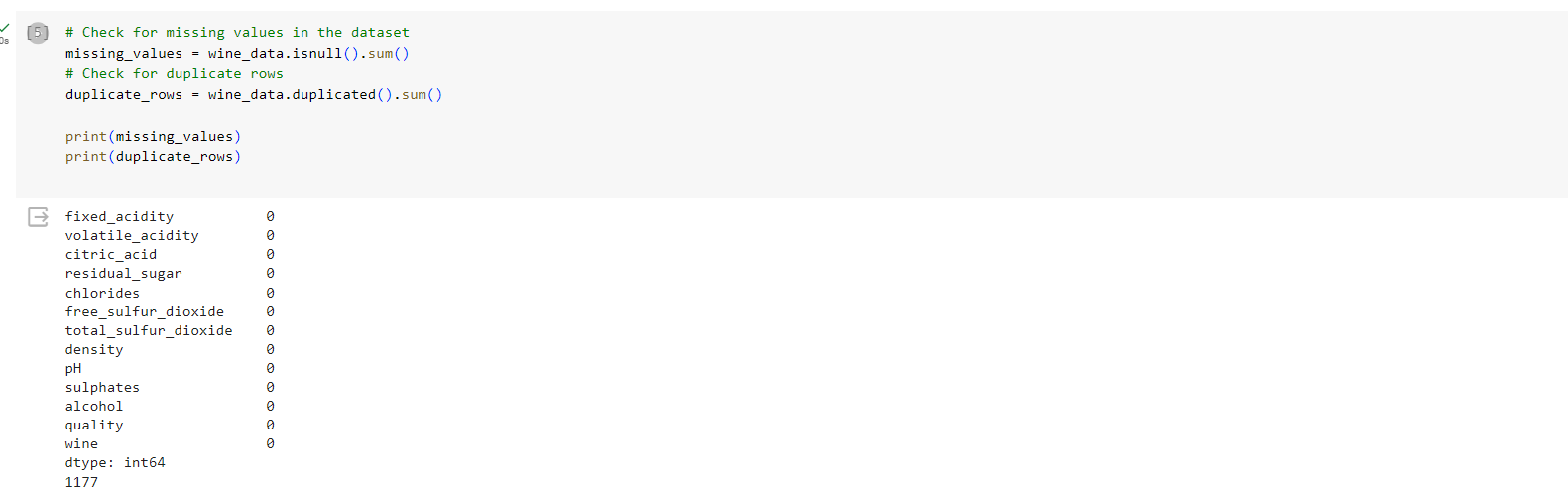
**pH:** pH measures the acidity or alkalinity of the wine. It affects the wine's chemical stability and can influence its taste.

**Sulphates:** Sulphates, such as potassium sulphate, are sometimes added to wine as a nutrient for yeast during fermentation. They can also affect the wine's aroma and flavor.

**Alcohol:** Alcohol content is the percentage of ethanol in the wine. It significantly influences the wine's body, mouthfeel, and overall sensory characteristics.

**Quality (Score Between 0 and 10):** Quality is often assessed using a numerical score, with higher scores indicating better quality. It's based on various sensory evaluations and may consider factors like taste, aroma, appearance, and overall impression. Our target variable.

**Wine:** The type of wine, red or white.

For our project, “quality” was used as our target variable while the other 12 variables would be our predictor variables. Data cleaning is a vital part of EDA so as part of this process we ensured that there were no missing records in our dataset and looked to see if there were any duplicates amongst the data.

A collage of graphs

Description automatically generatedAs seen in the figure above, there were no missing values in the dataset; however, there were 1,177 duplicate records. Before proceeding with how to handle the duplicate records, we performed additional analysis of the dataset by looking at the descriptive statistics.

**2.1 Univariate Analysis:**

A collage of graphs

Description automatically generated

Based on the univariate analysis above, we determined the following insights. For Fixed Acidity, most wines had a fixed acidity level between 6 and 8. For Volatile Acidity, most wines had volatile acidity levels below 0.6. For Citric Acid, the distribution appeared multi-modal, with peaks around 0, 0.3, and 0.5. For Residual Sugar, most wines showed to have had residual sugar levels below 10; however, there was a long tail which indicated that some wines have very high residual sugar. For Chlorides, most wines registered chloride levels below 0.1. For Free Sulfur Dioxide, the distribution was right-skewed with most wines having values below 60. For Total Sulfur Dioxide, most wines recorded total sulfur dioxide levels below 200. For Density, most wines clustered around 0.995. For pH, the pH values of wines were mostly between 3.0 and 3.5, meaning that wine is quite acidic. For Sulphates, most wines had sulphate levels below 0.8. As for Alcohol, the alcohol content of most wines were observed to be between 9% and 12%. For Quality, our target variable, the wines’ qualities ranged from 3 to 9 with most wines’ quality ratings scoring either 5 or 6. Additionally, of the 6,497 wines that were sampled, 1,599 were red wines compared to 4,898 that were white wines. The distribution of wine qualities and wine types can be seen in the figure below.

A graph of a normal distribution

Description automatically generated with medium confidenceA graph of different colored squares

Description automatically generated with medium confidence

**2.2 Bivariate Analysis:**

After completing our univariate analysis, we started our analysis on the relationships and correlations amongst the variables. To find the correlations we created a correlation matrix; however, in our python-based applications, correlation would only be calculated for numerical variables. All of our variables except “wine” were numerical; “wine” was a categorical variable, so in order to calculate its correlation with the other variables, we had to convert it to a numerical variable. This process involved creating a dummy variable where we set “white” equal to 0 and “red” equal to 1. After doing this step and having every variable be numerical, we were able to create the above correlation matrix. For our correlation matrix, color served as an indication for the type of relationship between two variables: the redder a box was, then the stronger and more positive the relationship was; the greyer a box was, then the weaker the relationship was or the A screenshot of a graph

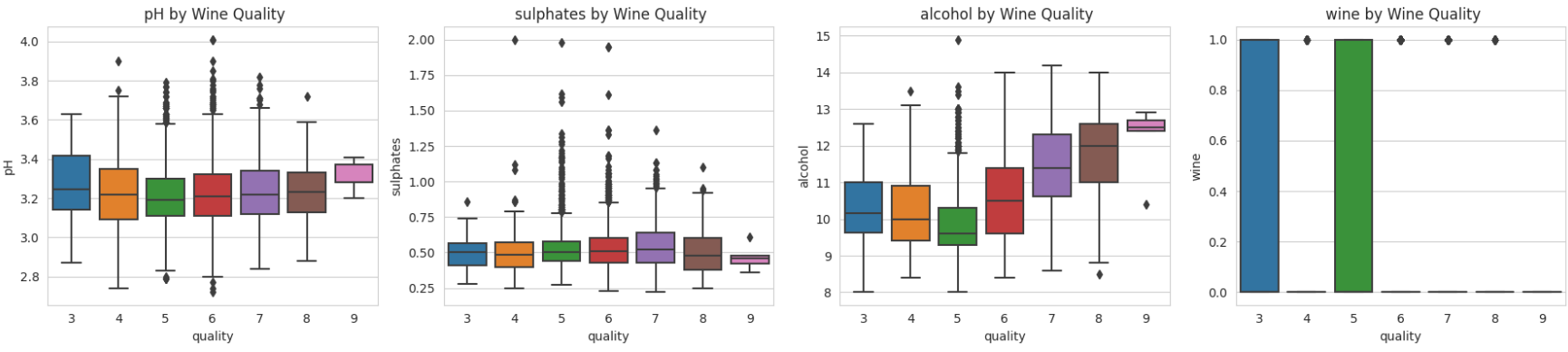
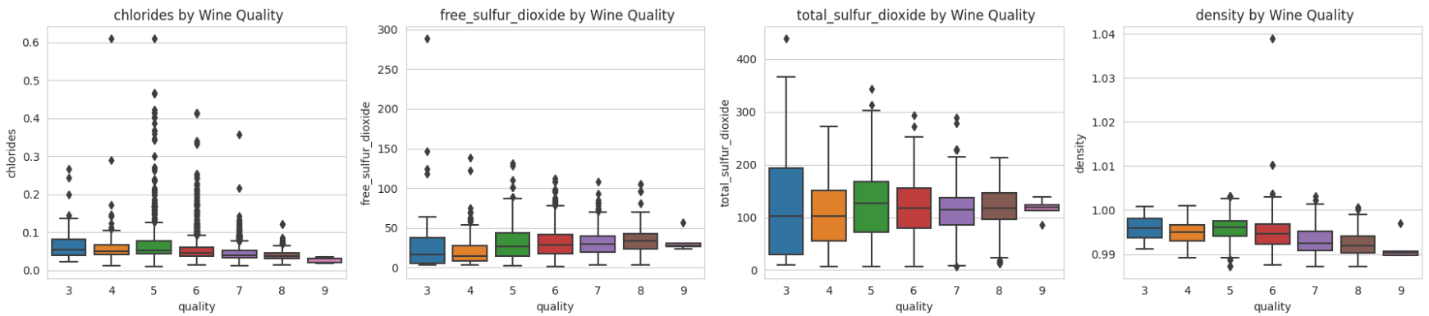
Description automatically generatedrelationship was non-existent; and the bluer a box was, then the stronger and more negative the relationship was. Predictor variables with the highest absolute correlation to our target variable, “quality”, were: “alcohol” at 0.44, “density” at -0.31, and “volatile\_acidity” at -0.27. Thus, we made the following conclusions. “Alcohol” had a moderate positive correlation with “quality”, which suggested that wines with higher alcohol content generally had a higher quality rating. “Density” had a moderate and negative correlation with “quality”; additionally, “density” had strong correlations with several other features like “residual\_sugar”, “fixed\_acidity”, and “alcohol”. This was expected as “density” could be influenced by the concentration of these substances in the wine; this also prompted us to analyze the potential of multicollinearity within our dataset. “Volatile\_acidity” had a moderate negative correlation with “quality”, which meant that wines with higher volatile acidity usually had a lower quality rating.

Continuing with our bivariate analysis, we looked more into the relationships between our predictor variables and our target variable “quality”. To do so we created boxplots for each variable and “quality”. Based on these graphs below, we determined the following conclusions.

A graph of different colored boxes

Description automatically generated

Between “fixed\_acidity” and “quality”, there was no clear trend between them. Between “volatile\_acidity” and “quality”, higher quality wines tended to have lower volatile acidity. Between “citric\_acid” and “quality”, wines with higher quality ratings appear to have slightly more citric acid on average. Between “residual\_sugar” and “quality”, there was no trend between them, but there were outliers present especially with lower quality wines. Between “chlorides” and “quality”, higher quality wines tended to have lower chloride levels. Between “free\_sulfur\_dioxide” and “quality”, there were no distinct trends. Between “total\_sulfur\_dioxide" and "quality”, higher quality wines generally had lower total sulfur dioxide, especially in the case of white wines. Between “density” and “quality”, wines with slightly lower densities were observed to be of higher quality. Between “pH” and “quality”, there was no clear trend. Between “sulphates” and “quality”, wines with slightly higher sulphate levels were observed to be of higher quality. Lastly between “alcohol” and “quality”, wines with higher alcohol content tended to be associated with higher wine quality.



**2.3 Multivariate Analysis:**

A screenshot of a graph

Description automatically generated After looking at the bivariate analysis, we continued our EDA process with multivariate analysis. To do so, we created several pairplots, which can be seen below, between the key variables of “volalite\_acidity”, “citric\_acid”, “chlorides”, “sulphates”, and “alcohol”. For the plots the darker blue the dots were, then the lower quality the wine was, while the darker red the dots were, then the higher quality the wine was. Key observations noted were that higher quality wines tended to have higher levels of citric acid and alcohol content and tended to have lower levels of chlorides. For the “volatile acidity” and “sulphates” pairplots, wines of all quality levels were spread across so no additional trends were noticed.

A screenshot of a graph

Description automatically generatedUsing the same variables we used in our multivariate analysis, we looked at the distribution of the data amongst the wine type, “red” and “white”. The graphs created for this analysis can be seen below. For the plots, red wines are indicated as “blue” and white wines are indicated as “orange”. Based on these graphs below, we determined the following conclusions.

For “alcohol”, both red and white wines had a wide distribution, but white wines tended to have a slightly higher mode. For “volatile\_acidity”, red wines generally recorded higher volatile acidity levels compared to those of white wines. For “chlorides”, red wines had a higher chloride concentration compared to those of white wines. For “citric\_acid”, the citric acid content in red wines was more spread out while the distribution of white wines’ citric acid contents peaked around 0.3. For “density”, both wine types had similar density distributions, with white wines having a slightly more pronounced peak.

**2.4 Variable Significance:**

Additionally, we conducted an ANOVA test on the dataset. For our ANOVA test, it evaluated the hypothesis that all groups, in our case, different wine quality levels, have the same population mean for a given feature. So, if we were presented with a small p-value, then the results suggested to us that at least two group means were different. Our results of the ANOVA A close-up of a computer screen

Description automatically generatedtest can be seen below.

All variables had a p-value significantly less than 0.05, indicating that there were statistically significant differences in the means of these features across different quality levels. Variables “alcohol”, “density”, and “chlorides” had especially low p-values, suggesting that their distributions vary considerably across quality levels.

Based on the findings in our EDA, we concluded that multicollinearity could be present upon our variables. In order to confirm this, we calculated Variance Inflation Factor (VIF) for all the variables. The results of our initial VIF test can be seen in the figure below.

A screenshot of a computer program

Description automatically generated

These results indicated that multicollinearity was indeed a factor in our dataset as variables with VIF scores over 10 indicate multicollinearity. As shown above, “density” and “ph” have significantly high VIF scores. We continued to remove variables until all remaining variables produced VIF scores of 10 or less. Our results from final VIF test can be seen below.

A screen shot of a computer

Description automatically generated

Thus, the variables that were deemed significant after the removal of variables causing multicollinearity were “volatile\_acidity”, “citric\_acid”, “residual\_sugar”, “chlorides”, “free\_sulfur\_dioxide”, and “wine”.

To ensure data quality and the robustness of any subsequent models, we looked to identify outliers that were present in the dataset. We identified 1473 records, approximately 22.67% of the dataset, that contained outlier values in one or more features.

**3.0 Methods / Modeling:**

For this project, we designed a multilayer neural network, a linear regression model, and a random forest model to predict the quality of wine. The Python libraries TensorFlow and Keras were used to build, train, and validate the neural network capable of predicting wine quality. The computational infrastructure for deep learning was established utilizing the NVIDIA GPU within the Google Colab. The execution of the deep learning experiment was facilitated by applying the Jupyter Notebook in conjunction with Google Colab, which was instrumental in creating the deep learning environment. (Chollet, 2021)

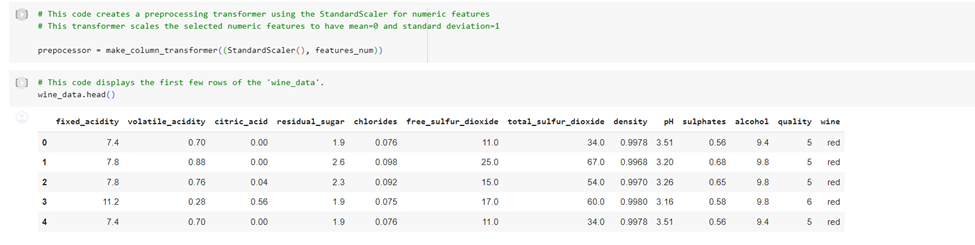
Several Python libraries were imported to add functionality to Jupyter Notebooks and Colab. The basic libraries and the script to load them were "import tensorflow as tf," "import pandas as pd," and "import numpy as np." To establish a deep learning environment, we loaded supplementary libraries. The libraries we used were: "from sklearn.preprocessing import StandardScaler," and "from sklearn.compose import make\_column\_transformer." We also loaded: "from tensorflow import keras," and "from tensorflow.keras import layers, callbacks." (Chollet, 2021)

**3.1 Multilayer Neural Network Model**

**3.1.1 Preparation for Neural Network Model Construction:**

Before feeding the data into the neural network, we had to prepare the data to be able to pass through the network. First, we identified and extracted the numeric features from the wine dataset and assigned them to the variable “features\_num” using the script illustrated below.

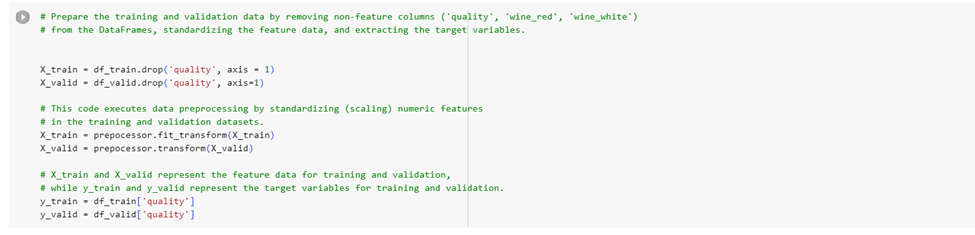
Given the inherent efficiency of neural networks when exclusively operating on numerical data, this procedural step assumed significance in the neural network construction process to numeric features. It was imperative to note that in instances where categorical data was employed, a prerequisite transformation became essential, often in the form of one-hot encoding and the creation of dummy variables. Additionally, this phase involved the segregation of predictor variables, excluding the incorporation of the target variable. (Chollet, 2021). The numeric features were then scaled to normalize them, ensuring a mean of 0 and standard deviation of 1 for all features. As depicted in the figure below, the normalization procedure employed the StandardScaler, wherein the application of this scaler facilitated the transformation of the numeric features to conform to the specified mean and standard deviation criteria; thus, enhancing the homogeneity and comparability of the dataset.

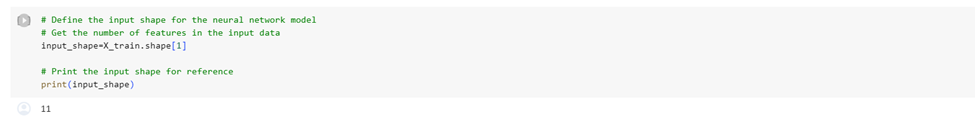


**3.1.2 Data Splitting for Neural Network Model:**

In training a neural network, a pivotal step involved partitioning the dataset into distinct training and validation subsets at a minimum. We split our data into subsets equivalent to 70% training and 30% validation. Partitioning into training and validation was paramount in neural network training methodologies to mitigate overfitting and ensure the data had been trained and had not been exposed to the same data during both the training and validation phases. This approach allowed for assessing error metrics on the training and validation sets comprehensively evaluating the model's performance. Modern neural network construction practices have strongly advocated dividing data when the dataset size permits. Refer to the figure below for a visual representation of this partitioning.

**3.1.3 Feature and Target Extraction for Neural Network Model:**

After sub-setting into training and validation, we created new variables X\_train and X\_valid, which only included the numeric variables. Concurrently, the predictor and target variables, "quality," were excluded from these subsets. The ensuing step involved the standardization of these newly derived subsets. The procedural implementation of these operations was delineated in the below figure, providing a succinct representation of the coding employed to execute these preparatory steps.

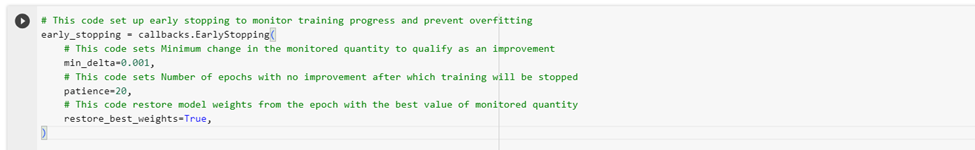
The shape of our data, as input into the neural network model, now comprises 11 dimensions. The shape can be depicted in the figure below, which presents the corresponding script and the resultant output, thereby encapsulating this pivotal phase in the preprocessing procedure.

**3.1.4 Neural Network Model Building:**

We built an early stopping into the training loop of our neural network. Early stopping is a regularization technique used to prevent overfitting by discontinuing the training loop when the network is not improving during the training cycle. This minimizes a model's tendency to learn the training data too well, including its noise and details that do not generalize to new, unseen data. (Chollet, 2021)

We used several components within the early stopping script. Here is an explanation of the parameters used in the **EarlyStopping** callback:

* **min\_delta:** This parameter sets the minimum change in the monitored quantity (e.g., validation loss) to qualify as an improvement. If the change is less than this value, it is considered insignificant, and the training continues.
* **Patience:** This parameter determines the number of epochs with no improvement, after which the training will be stopped. In other words, training is halted if the monitored quantity does not improve for the specified number of epochs.
* **restore\_best\_weights:** This parameter restores the model weights from the epoch with the best value of the monitored quantity. This is useful because even though the training might continue for additional epochs without improvement, we want to keep the model parameters that performed best on the validation set. See the figure below.

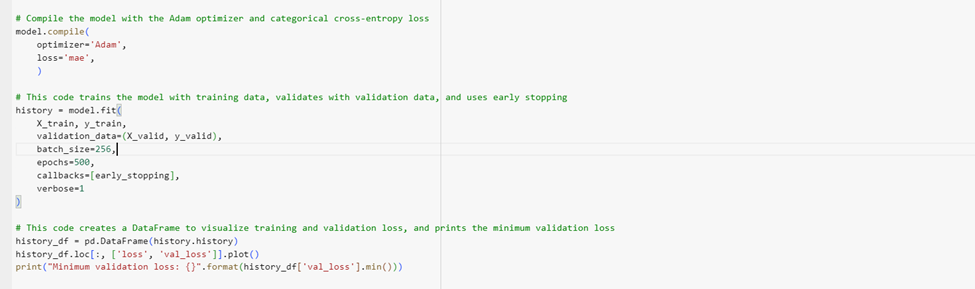


The variable "num\_classes" was assigned the value 3, denoting the number of classes in the context of the neural network. This part of the code served the purpose of constructing the neural network model. The architecture was delineated within the keras. Sequential framework encompassed 3 fully connected Dense layers. Each dense layer was configured with “32” units and ‘relu’ activation functions, contributing to the network's overall structure.

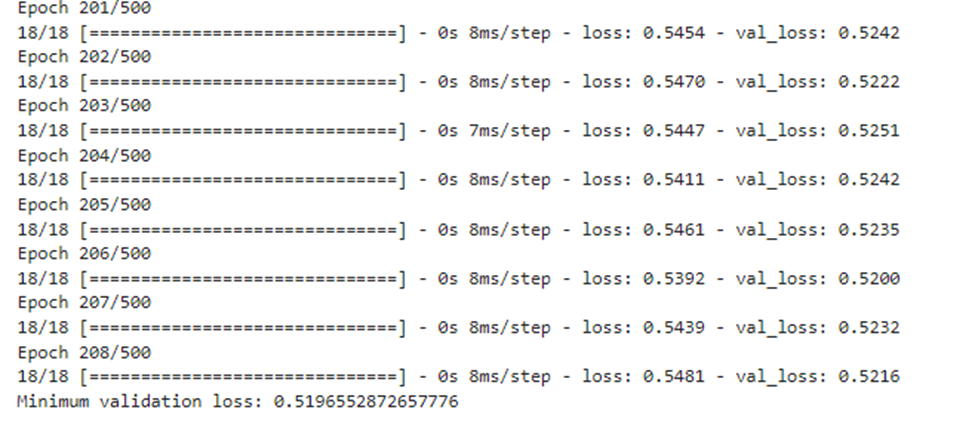
We added dropout layers with ‘0.3’ and batch normalization layers to enhance our model's performance between these fully connected dense layers. The model's final layer, “the output .layer" had a single unit. This representation of the neural network architecture was integral to the objective of this project. See the following figure. (Chollet, 2021)



Next, the neural network was compiled, utilizing an ‘Adam’ optimizer and mean absolute error (MAE) as the loss function. The compile process was instrumental in configuring the model for the subsequent training loop. The training procedure next executed, training the model over the training data while validating against a separate validation dataset. The training was conducted over 500 epochs with a batch size of 256. This implementation integrated the early stopping mechanism, as indicated by including the 'early\_stopping' callback, to monitor the training progress and minimize overfitting. A data frame, 'history\_df,' was also generated to represent the training and validation loss graphically. The minimum validation loss was then printed, providing a quantitative insight into the model's performance during training. This comprehensive approach facilitated a thorough understanding of the neural network's behavior and convergence patterns. This can be seen in the figure below.

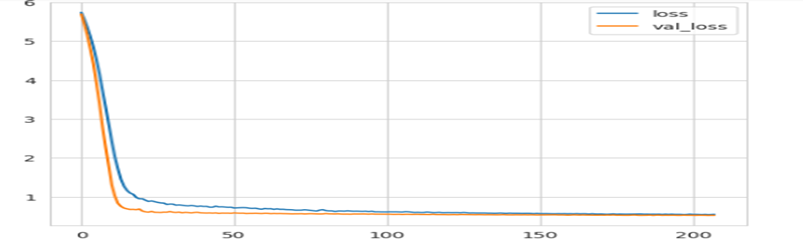


The incorporated early stopping mechanism in the training loop concluded at epoch 208, revealing a training loss of 0.5481 and a corresponding validation loss of 0.5216. Notably, the minimum validation loss was recorded at 0.5196552872657776. A graphical representation of these metrics was recorded in the figure below.

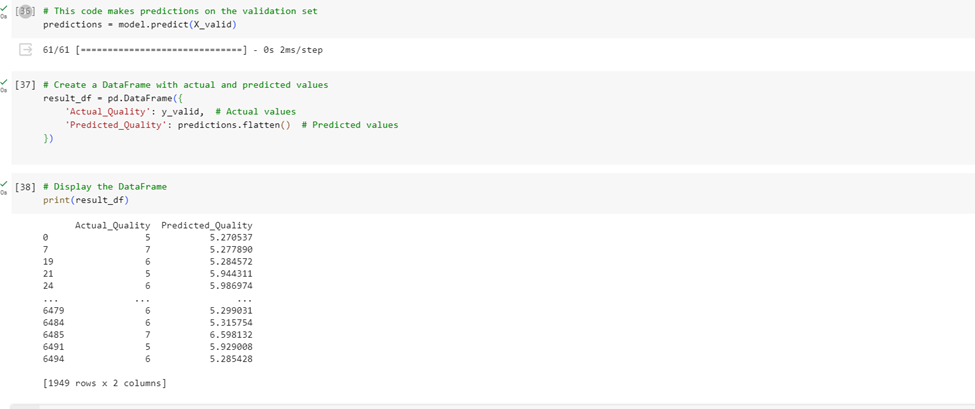


**3.1.5 Training History Visualization for Neural Network Model:**

A visual representation of the training and validation process was recorded and depicted in the following figure.



Next, we used the trained model to make predictions on the validation subset and display the actual values alongside them. This process showed how well our model predicts the wine quality.



**3.1.6 Results and Interpretations for Neural Network Model:**

The table in the preceding figure comprised two columns, "Actual\_Quality" and "Predicted\_Quality," representing individual instances from the validation set. Each row corresponded to a specific instance, providing the actual quality of the wine ("Actual\_Quality") and the corresponding quality predicted by the neural network ("Predicted\_Quality"). A comprehensive table analysis revealed that the model's predictive performance falls short of expectations. A considerable number of predictions clustered around a variation of 5, with only a solitary instance observed at row 6485, demonstrating a deviation beyond 5. This discrepancy suggested that the model struggles to make accurate predictions, and instances where "Actual\_Quality" and "Predicted\_Quality" closely align were relatively infrequent. The model's effectiveness in predicting wine quality was suboptimal, as indicated by the notable disparities between predicted and actual values.

**3.2 Linear Regression Model**

**3.2.1 Preparation for Linear Regression Model Construction:**

Similar to our neural network model, in the creation of our linear regression model we once again had to prepare the data before processing it through this model. Exclusively having numerical variables was necessary for the creation of our linear regression model. For a linear regression model, we had to assume there was a linear relationship between our independent variables and the target variable, “quality”. With the data preparation performed for our neural network model and created a dummy variable/one-hot encoding on “wine” to make it a numerical variable, we confirmed that our dataset contained only numerical variables and proceeded with the creation of a linear regression model.

**3.2.2 Data Splitting for Linear Regression Model:**

A screen shot of a computer program

Description automatically generatedFor the creation of our linear regression model, splitting the dataset into separate training and validation subsets was needed. To remain consistent with previous modeling procedures, we split our initial dataset into 70% for training data and 30% for validation data. Data splitting was done to reduce the effects of overfitting and to ensure the data was exposed to the same data during the training and validation phases. Refer back to 3.1.2, for additional information regarding the benefits of data splitting for model creation and testing. The code used to split the dataset for the linear regression model can be seen in the figure below.

**3.2.3 Feature and Target Extraction and Model Building for Linear Regression Model:**

A computer screen shot of a program

Description automatically generatedAfter the data split, we began the process to fit the data to the linear regression model. For the initial linear regression model, we referred to our VIF test done as part of our EDA. We used the all predictor variables and the results can be seen in the figure below. Additionally, we set the random state to 42 for reproducibility amongst our testings.

To potentially improve the linear regression model, we utilized feature engineering; so, we decided to only include variables that were deemed significant/relevant which were indicated by VIF scores of 10 or less; these variables consisted of “volatile\_acidity”, “citric\_acid”, “residual\_sugar”, “chlorides”, “free\_sulfur\_dioxide”, and “wine”. This linear regression model A screen shot of a computer program

Description automatically generatedand its result can be seen below.

A screen shot of a computer

Description automatically generatedOur results indicated a worse performing model, so we added back some variables that were just excluded in the model above. The best performing linear regression model only excluded the “pH” and “density” variables and can be seen in the figure below.

A screen shot of a graph

Description automatically generatedA screen shot of a computer

Description automatically generated

**3.2.4 Results and Interpretations for Linear Regression Model:**

To evaluate the performance of the linear regression models, we used mean squared error (MSE) and R-squared. MSE measured the average squared difference between predicted values and the actual values in the dataset; so, a better performing model would have a lower MSE value. R-squared measured the proportion of the variance in the target variable, “quality”, that was explained by the independent/predictor variables; R-squared values range from 0 to 1 with 0 indicating that the model does not explain the variance and 1 indicates a perfect fitted model. So, a better performing model would have a higher R-squared value. Among the linear regression models, we created the first model with all predictor variables performed the best with a MSE of 0.5276 and R-squared value of 0.2769. Regardless of the changes we made, our best performing model yielded a R-squared value of 0.2679 which indicated that the linear regression model did not properly fit the dataset for our situation.

**3.3 Random Forest Model**

**3.3.1 Preparation for Linear Regression Model Construction:**

Attempting to create a better predictive model to predict wine quality, we subsequently created a random forest model. Unlike the neural network and linear regression models, we did not have to ensure the data was numerical before for the creation of the random forest model as random forest model are able to handle both numerical and categorical datasets. However, since we had already cleaned and transformed our “wine” variable into a numerical variable, we did not conduct any additional data preparation steps and proceeded to split the data.

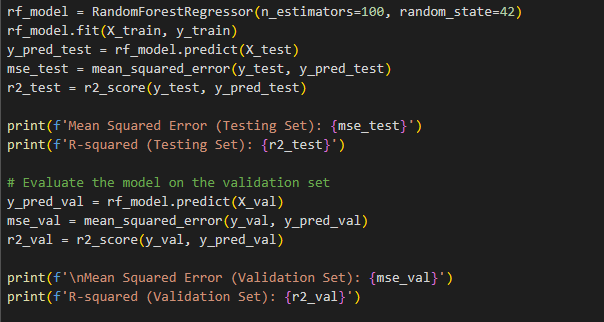
**3.3.2 Data Splitting for Random Forest Model:**

For the creation of our random forest model, we once again split the dataset into separate training and validation subsets. For this data split, we allocated 60% for training data, 20% for testing data, and 20% for validation data. Here, we incorporated the validation dataset to further assist with model performance. The benefits of data splitting are detailed more in the previous section 3.1.2. The code used to split the dataset for the linear regression model can be seen in the figure below.

A computer screen shot of a program code

Description automatically generated

**3.3.3 Random Forest Model Building:**

 We built a model that trains the wine dataset to a random forest regression model that uses MSE and R-squared as the performance metrics and that visualizes the predictions against the actual values in the validation dataset. For the random forest, we instantiated the model with 100 trees and set the random state to 42 for reproducibility. After this we again referenced the VIF test conducted in our EDA and we decided to drop variables “density” and “pH” due to their potential causation to multicollinearity and their potential to decrease the performance of the model. We then proceeded with the creation of the random forest model which can be seen in the following figure.

**3.3.4 Results and Interpretations for Random Forest Model:**

A screenshot of a computer screen

Description automatically generated

The figure above visualized the results of our random forest model; it plotted the predicted quality of the wine from our validation dataset compared to their actual quality values in the same validation dataset. Scoring the performance of the model was done with MSE and R-squared. Ideally, we were looking for a model that would have a sufficiently low MSE and a R-squared value close to 1; however, based on our results for the random forest model, our MSE was low, but the R-squared value was only 0.46 on our validation dataset. This indicated that only 46% of our variance in the validation dataset could be explained by the model; meaning that this model showed reasonable performance but did not meet our expectations for an accurate predictive model for “quality”.

**4.0 Evaluation / Discussion**:

We created three unique models, neural network, linear regression, and random forest, to predict the quality of wine given the wine’s psychochemical properties. We then took our three predictive models and compared their performance metrics to determine the best model evaluate and discuss further. Our neural network model roughly had an accuracy rate around 56%. Our best performing linear regression and random forest models used R-squared to score performance and our linear regression model had a R-squared value around 28% and our random forest model had a R-squared value around 46%. Thus, we determined that our neural network model was the best predictive model to evaluate the wine quality in the dataset.

The wine classification dataset was divided into two sets: 70% for training and 30% for testing. The objective was to see which model performed best at classifying the wine quality, which ranged from 1 to 9. To ensure the accuracy of the model, we applied some preprocessing steps. The variables were normalized using StandardScaler, which helped us to maintain consistency and avoid biases.

We designed a traditional multi-layer neural network consisting of three hidden layers and one output layer. In order to avoid overfitting, we employed some techniques that included early stopping, a dropout layer with a rate of 0.3, and batch normalization. These techniques helped us to normalize the output of the previous layers, reduce internal covariate shift, and speed up the training process. For the output layer, we used a dense layer with one unit which produced a highly accurate and reliable output.

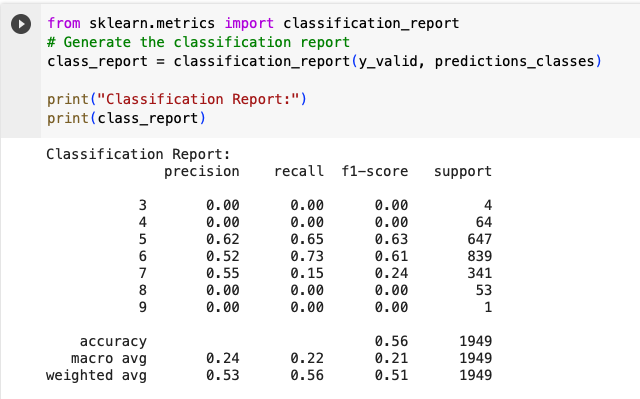
**4.1 Training procedure:**

We used several parameters during the training procedure for our wine dataset, which helped in optimizing our neural network model.

* Optimizer: We chose Adam for the optimization algorithm to train our neural network model. Which calculates adaptive learning rates for each parameter based on the first and second moments of the gradients. The first moment, which is mean, and the second, which is uncentered variance.
* Loss Function: We set to minimize the "mae” loss, which means Mean Absolute Error which quantifies the model prediction error. MAE metric measure is the difference between the predicted values and the true values.
* Early stopping: We applied the early stopping to prevent overfitting. The training process stopped after 208 epochs, before any degradation in model performance on validation data.
* Batch size/Epochs: We chose a batch size of 256, meaning 256 samples per update. The model iterated 500 times through the training set.

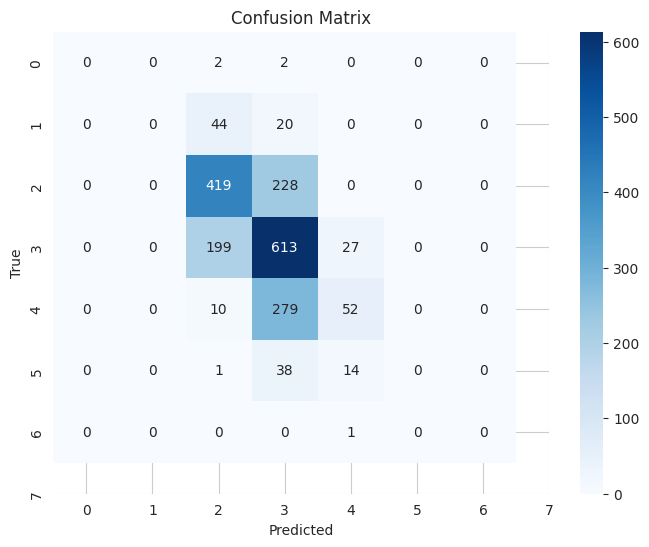
**4.2 Model evaluation:**

We evaluated our wine classification neural network model on the testing dataset and achieved an accuracy rate of about 56%. Additionally, the model had a minimum validation loss of 0.5061. This indicates that our wine classification model correctly predicted wine classes for approximately 56% of previously unseen new data. We also generated the classification reports for each class. Reports are defined below:



In the classification report, precision showed how accurate the positive predictions were for each class. Recall (Sensitivity) showed the proportion of correctly identified positive observations to all actual positives for each class. F-1 score was the harmonic mean of precision and recall, which gave a balanced measure of the classifier's performance. Support indicated the number of occurrences of each class in the testing dataset.

After analyzing the classification report and confusion matrix, we found that our model could accurately classify wine quality 5 with an accuracy of 62% positive predictions and an overall accuracy of 63%. This meant that for every ten wines of quality 5, our model would correctly classify six of them. The model could correctly classify wine quality 6 with an accuracy of 52% positive predictions and an overall accuracy of 61%. This meant that for every ten wines of quality 6, our model will correctly classify 6 of them. The model could correctly classify wine quality 7 with an accuracy of 55% positive predictions and an overall accuracy of 24% This means that for every ten wines of quality 7, our model would correctly classify 2 of them. However, we found that the model had a precision of 0% positive predictions and 0% overall accuracy for wine qualities 3, 4, 8, and 9. In other words, the model could not accurately classify these wine qualities, and its positive predictions for these qualities were incorrect.



When it comes to classifying data, there were two parameters that were commonly used in the classification report. The first one was called the macro average, which indicated the average F1-score across classes. It was used to get an idea of the model's performance in a more general sense. Our model had an average F-1 score of 21% across classes. The second parameter was a weighted average, which showed the weighted averaged precision, recall, and F1-score was calculated by considering class frequencies. This helped to better evaluate the performance of the model across all classes. So, our model had 51% weighted averaged.

**4.3 Conclusion:**

Based on the performance of our best model, the neural network model, we determined that this model could be used as a building block for additional analysis into wine quality. Wine industry experts could use our model to analyze the variables that go into producing a wine of quality 5, 6, and 7. However, they would need to conduct additional research into what would produce wines with qualities on the much lower and much higher end of the spectrum. Looking at the dataset presented to us, we recorded the dataset to contain less than 6,500 records. We believe a study for such a big and influential industry such as the wine industry to have more records. We would recommend obtaining additional samples to be studied for better predictive models and for more conclusive results. In our dataset, the number of white wines outnumbered the number of red wines by a ratio of 3 to 1; the two types of wine could be uniquely affected by the different psychochemical property. So, we would recommend that for the next study on wine quality to have an equal number of samples split between red and white wines and analyze whether the two types have different variables affect quality. Ultimately, the wine quality is affected by several variables and our predictive models were moderately accurate for wines with qualities ranging from 5 to 7 but struggled with predicting the quality from 0 to 4 and 8 to 9.

**Works Cited**

Chollet, Francois. (2022). Deep learning with python, second edition. Manning Publications.

Cortez,Paulo, Cerdeira,A., Almeida,F., Matos,T., and Reis,J.. (2009). Wine Quality. UCI

Machine Learning Repository. https://doi.org/10.24432/C56S3T.

“How Much Is America’s Wine Industry Worth? | Wine Spectator.” Wine Spectator, 2017,

[www.winespectator.com/articles/how-much-is-americas-wine-industry-worth](http://www.winespectator.com/articles/how-much-is-americas-wine-industry-worth).