Group 3 – Wine Quality

By Tristan Leiter, Giulio Iepure

Application context.

Suppose you own a vineyard in Portugal.

We produce several different wines with our amazing grapes.

- ➢ But how do we assess the quality of our wine to adjust for the correct pricing?

Personal samples always require a lot of time and effort.

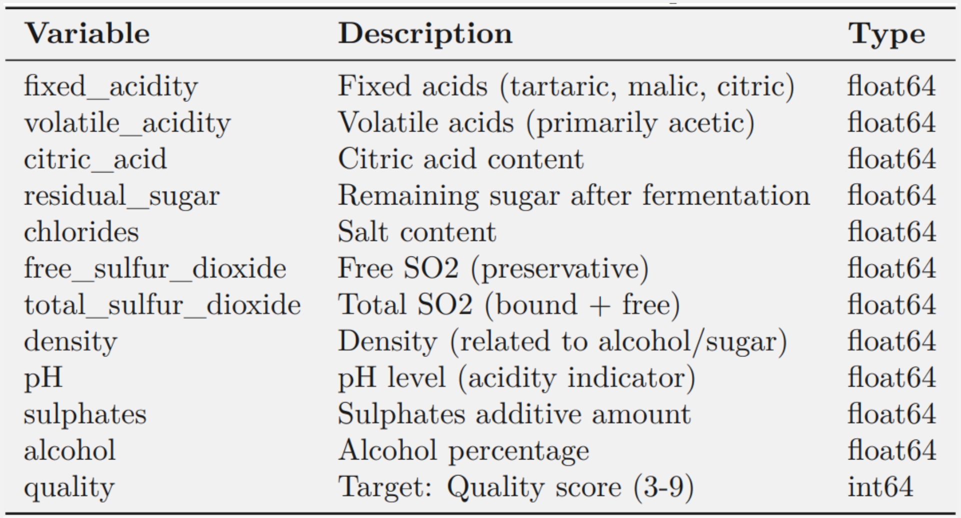
Specifically, too much personal sampling in the short-term can contribute to **distortions** in one's **personal perception** and the **reliability** of our **estimates**.

Problem setup.

Multi-class classification problem to predict the wine quality.

Data source:

- Data is obtained from Kaggle.
- It includes 21 000 observations, including 1 dependent variable, wine quality, and 11 features.
- The dependent variable measures the **wine quality** on an **ordinal scale** from 3 (lowest) to 9 (highest).
  - ➢ This is why, we apply a **multi-class classification** approach as it treats the scores as distinct, ordered categories rather than continuous values.
- In addition, it can be assumed that the **dependency** between at least some features and the dependent variable is **complex** (for example chemical features) with **non-normality** considerations.
  - ➢ This is why, algorithms, like **decision trees** in general, but, for example, **random forests** specifically are well suited to capturing the dependency structure in the data without imposing strong assumptions about the underlying data generating process.



Variable	Description	Type
fixed_acidity	Fixed acids (tartaric, malic, citric)	float64
volatile_acidity	Volatile acids (primarily acetic)	float64
citric_acid	Citric acid content	float64
residual_sugar	Remaining sugar after fermentation	float64
chlorides	Salt content	float64
free_sulfur_dioxide	Free SO2 (preservative)	float64
total_sulfur_dioxide	Total SO2 (bound + free)	float64
density	Density (related to alcohol/sugar)	float64
pH	pH level (acidity indicator)	float64
sulphates	Sulphates additive amount	float64
alcohol	Alcohol percentage	float64
quality	Target: Quality score (3-9)	int64

Table 1: Descriptions of the features and the target variable.

Data exploration.

The dependent variable shows a balanced distribution.

Dependent Variable: Wine Quality

- The dependent variable measures the wine quality on an ordinal scale from 3 (lowest) to 9 (highest).
- Wine quality is balanced across all quality scores; each consists of 3 000 observations in the dataset.
- The dependent variable is set up as a factor to retain the ordinality of the scale.

Quality Score	Obs.
3	3 000
4	3 000
5	3 000
6	3 000
7	3 000
8	3 000
9	3 000

Features

Multicollinearity

- In terms of multicollinearity, we encounter high correlations between most of our features. Amongst others, residual sugar and density show a correlation of 0.79 whilst residual sugar and free sulfur dioxide are also highly correlated (0.78).

Informational Value

- We discovered that density and total and free sulfur are features with a high predictive power, followed by residual sugar and chlorides.

Standardization

- The variances and scale of our features differ by several orders of magnitude, thus, and to prepare the data properly for regularization we standardize the features before modeling.

Bimodality

- Most features show a bimodal distribution (e.g. density, citric acid, and others). This is likely due to the inclusion of both red and white wines in the dataset.

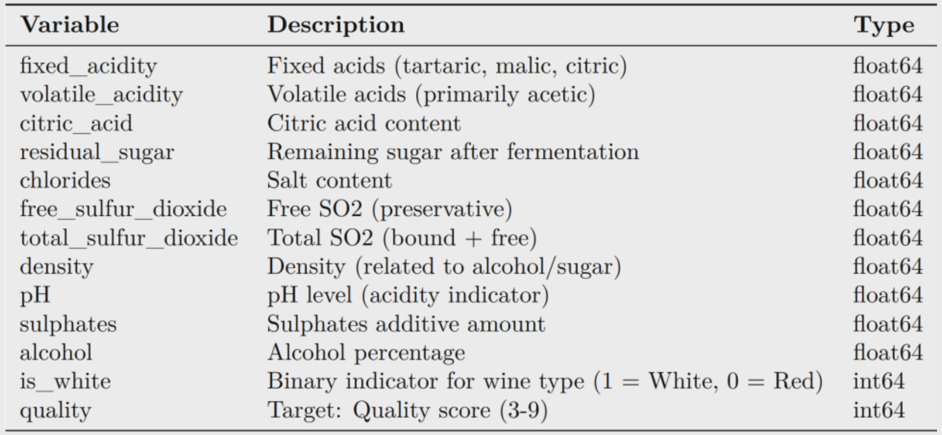
Feature selection and engineering

Selection

- Even though we encounter high correlations, none are higher than 0.9 whilst all show at least low predictive power (in terms of informational value).

Feature engineering

- We create a dummy variable, which captures whether the wine type is rather a white wine (=1) or a red wine (=0) based on partitioning the dataset on the most differentiating features (total_sulfur_dioxide, chlorides and volatile_acidity).



Variable	Description	Type
fixed_acidity	Fixed acids (tartaric, malic, citric)	float64
volatile_acidity	Volatile acids (primarily acetic)	float64
citric_acid	Citric acid content	float64
residual_sugar	Remaining sugar after fermentation	float64
chlorides	Salt content	float64
free_sulfur_dioxide	Free SO2 (preservative)	float64
total_sulfur_dioxide	Total SO2 (bound + free)	float64
density	Density (related to alcohol/sugar)	float64
pH	pH level (acidity indicator)	float64
sulphates	Sulphates additive amount	float64
alcohol	Alcohol percentage	float64
is_white	Binary indicator for wine type (1 = White, 0 = Red)	int64
quality	Target: Quality score (3-9)	int64

Table 2: Descriptions of the features and the target variable after feature engineering.

Feature characteristics.

We observe several crucial characteristics like multicollinearity and bimodality.

Dependent Variable: Wine Quality

- The dependent variable measures the wine quality on an ordinal scale from 3 (lowest) to 9 (highest).
- Wine quality is balanced across all quality scores; each consist of 3 000 observations in the dataset.
- The dependent variable is set up as a factor to retain the ordinality of the scale.

Quality Score	Obs.
3	3 000
4	3 000
5	3 000
6	3 000
7	3 000
8	3 000
9	3 000

Features

Multicollinearity

- In terms of multicollinearity, we encounter high correlations between most of our features. Amongst others, residual sugar and density show a correlation of 0.79 whilst residual sugar and free sulfur dioxide are also highly correlated (0.78).

Informational Value

- We discovered that density and total and free sulfur are features with a high predictive power, followed by residual sugar and chlorides.

Standardization

- The variances and scale of our features differ by several orders of magnitude, thus, and to prepare the data properly for regularization we standardize the features before modeling.

Bimodality

- Most features show a bimodal distribution (e.g. density, citric acid, and others). This is likely due to the inclusion of both red and white wines in the dataset.

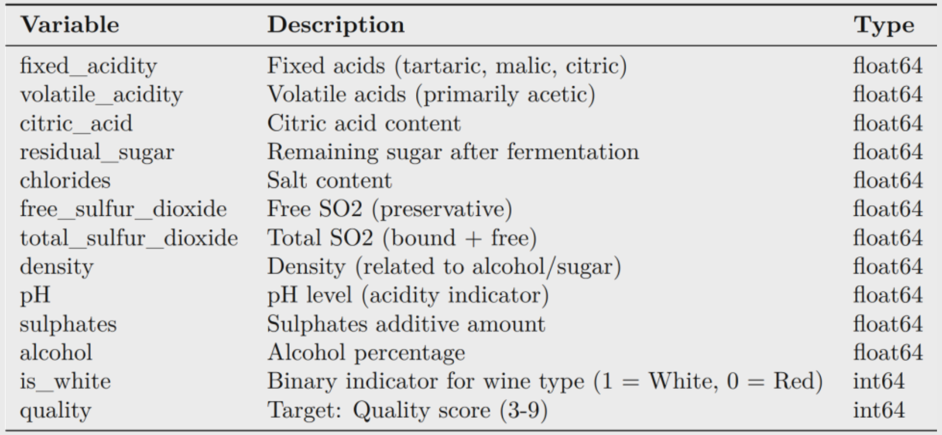
Feature selection and engineering

Selection

- Even though we encounter high correlations, none are higher than 0.9 whilst all show at least low predictive power (in terms of informational value).

Feature engineering

- We create a dummy variable, which captures whether the wine type is rather a white wine (=1) or a red wine (=0) based on partitioning the dataset on the most differentiating features (total_sulfur_dioxide, chlorides and volatile_acidity).



Variable	Description	Type
fixed_acidity	Fixed acids (tartaric, malic, citric)	float64
volatile_acidity	Volatile acids (primarily acetic)	float64
citric_acid	Citric acid content	float64
residual_sugar	Remaining sugar after fermentation	float64
chlorides	Salt content	float64
free_sulfur_dioxide	Free SO2 (preservative)	float64
total_sulfur_dioxide	Total SO2 (bound + free)	float64
density	Density (related to alcohol/sugar)	float64
pH	pH level (acidity indicator)	float64
sulphates	Sulphates additive amount	float64
alcohol	Alcohol percentage	float64
is_white	Binary indicator for wine type (1 = White, 0 = Red)	int64
quality	Target: Quality score (3-9)	int64

Table 2: Descriptions of the features and the target variable after feature engineering.

Feature selection and engineering

We create a new dummy variable to resemble either white or red wine.

Dependent Variable: Wine Quality

- The dependent variable measures the wine quality on an ordinal scale from 3 (lowest) to 9 (highest).
- Wine quality is balanced across all quality scores; each consist of 3 000 observations in the dataset.
- The dependent variable is set up as a factor to retain the ordinality of the scale.

Quality Score	Obs.
3	3 000
4	3 000
5	3 000
6	3 000
7	3 000
8	3 000
9	3 000

Features

Multicollinearity

- In terms of multicollinearity, we encounter high correlations between most of our features. Amongst others, residual sugar and density show a correlation of 0.79 whilst residual sugar and free sulfur dioxide are also highly correlated (0.78).

Informational Value

- We discovered that density and total and free sulfur are features with a high predictive power, followed by residual sugar and chlorides.

Standardization

- The variances and scale of our features differ by several orders of magnitude, thus, and to prepare the data properly for regularization we standardize the features before modeling.

Bimodality

- Most features show a bimodal distribution (e.g. density, citric acid, and others). This is likely due to the inclusion of both red and white wines in the dataset.

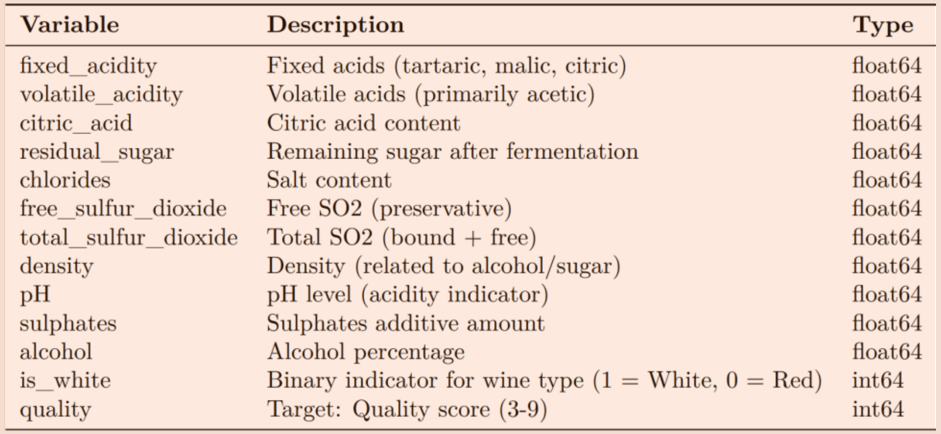
Feature selection and engineering

Selection

- Even though we encounter high correlations, none are higher than 0.9 whilst all show at least low predictive power (in terms of informational value).

Feature engineering

- We create a dummy variable, which captures whether the wine type is rather a white wine (=1) or a red wine (=0) based on partitioning the dataset on the most differentiating features (total_sulfur_dioxide, chlorides and volatile_acidity).



Variable	Description	Type
fixed_acidity	Fixed acids (tartaric, malic, citric)	float64
volatile_acidity	Volatile acids (primarily acetic)	float64
citric_acid	Citric acid content	float64
residual_sugar	Remaining sugar after fermentation	float64
chlorides	Salt content	float64
free_sulfur_dioxide	Free SO2 (preservative)	float64
total_sulfur_dioxide	Total SO2 (bound + free)	float64
density	Density (related to alcohol/sugar)	float64
pH	pH level (acidity indicator)	float64
sulphates	Sulphates additive amount	float64
alcohol	Alcohol percentage	float64
is_white	Binary indicator for wine type (1 = White, 0 = Red)	int64
quality	Target: Quality score (3-9)	int64

Table 2: Descriptions of the features and the target variable after feature engineering.

Loss-measure.

Macro F1-score is used to compare the performance of all our implemented models.

Macro F1-score:

$$F1 = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

$$Macro\ F1 = \frac{(F1_{class1} + F1_{class2} + \ \dots . + F1_{classN})}{N}$$

,where N is the total number of classes (6).

Explanations:

- The quality scores are balanced in our dataset; thus, accuracy would also work.
- Macro F1 focuses on a balanced performance: both precision and recall are incorporated in our loss-measure.

Data splitting:

- 70% of the data is allocated to the training set.
- The remaining 30% is reserved for the test set.
- We use 5-fold cross validation for all models in the training set.

Additional formulas:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$Precision = \frac{(TP)}{(TP + FP)}$$

$$Recall/Sensitivity = \frac{(TP)}{(TP + FN)}$$

- TP: Actual quality is 7. Model predicts 7.
- TN: Actual quality is 7. Model predicts something else.
- FP: Actual quality is NOT 7. Model predicts 7.
- TN: Actual quality is NOT 7. Model does NOT predict a 7.

(same for all other classes).

Loss-measure.

Macro F1-score is used to compare the performance of all our implemented models.

Macro F1-score:

$$F1 = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

$$Macro\ F1 = \frac{(F1_{class1} + F1_{class2} + \ \ldots . + F1_{classN})}{N}$$

,where N is the total number of classes (6).

Explanations:

- The quality scores are balanced in our dataset; thus, accuracy would also work.
- Macro F1 focuses on a balanced performance: both precision and recall are incorporated in our loss-measure.

Data splitting:

- 70% of the data is allocated to the training set.
- The remaining 30% is reserved for the test set.
- We use 5-fold cross validation for all models in the training set.

Additional formulas:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$Precision = \frac{(TP)}{(TP + FP)}$$

$$Recall/Sensitivity = \frac{(TP)}{(TP + FN)}$$

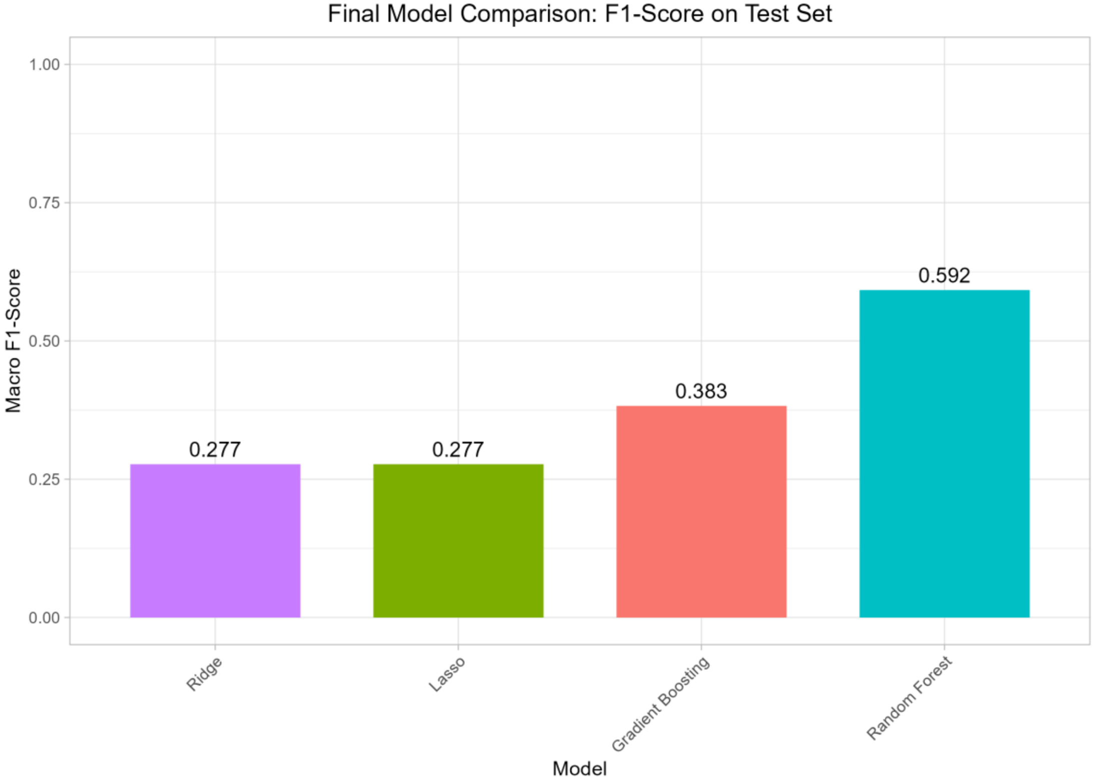
- TP: Actual quality is 7. Model predicts 7.
- TN: Actual quality is 7. Model predicts something else.
- FP: Actual quality is NOT 7. Model predicts 7.
- TN: Actual quality is NOT 7. Model does NOT predict a 7.

(same for all other classes).

Model performance comparison.

Model results for lasso, ridge, random forest and gradient boosting.

Performance Comparison:



Key Insights:

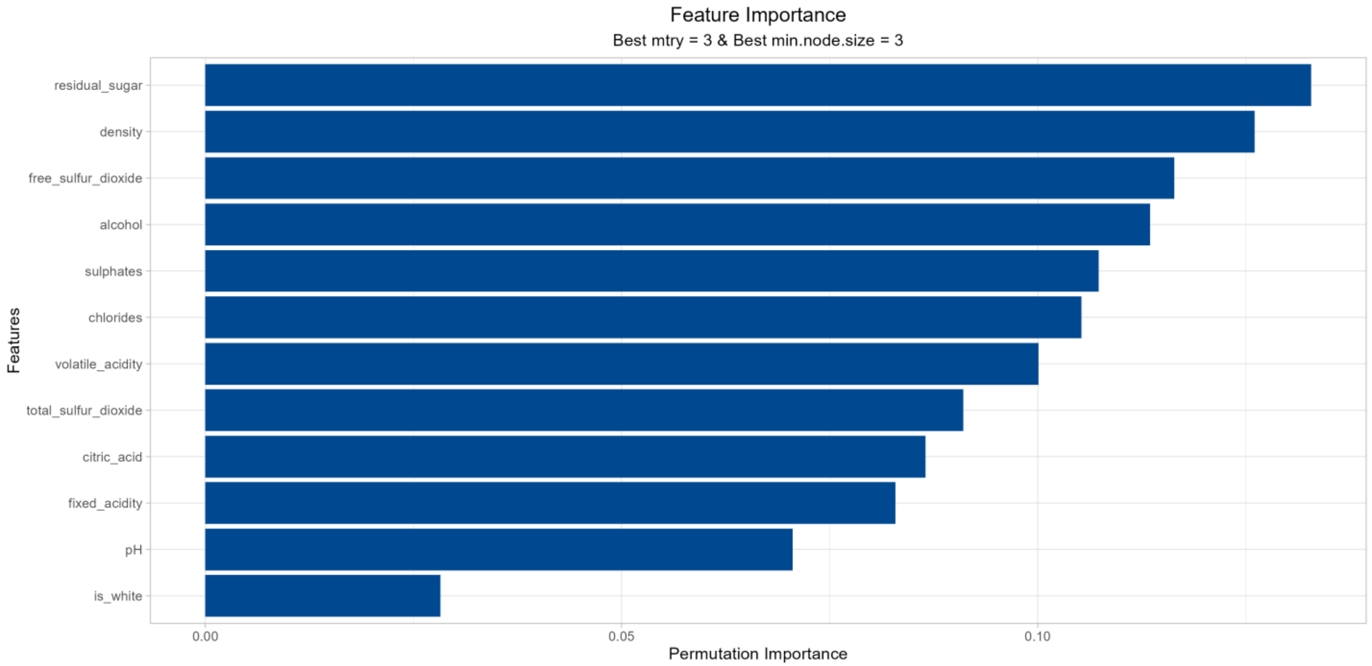
- **Random Forest is the winner**
  - With a Macro F1-score of 0.592, the Random Forest model shows a moderate predictive power.
  - It performs much better than random guessing but is not yet highly accurate.
- **Gradient Boosting performs worse than RF**
  - GBM is more sensitive to the hyperparameters (compared to RF), we performed a k-fold cross validation grid search on the tree-depth (4,6 and 8), the learning rate (0.01 to 0.05) and the number of trees (initially with 2’000, the model selected 463). The best hyperparameters could be outside these ranges.

Random Forest is selected to be the final model.

Random Forest performance.

No dominant feature is identified, but all features seem to show at least some importance.

Feature importance:



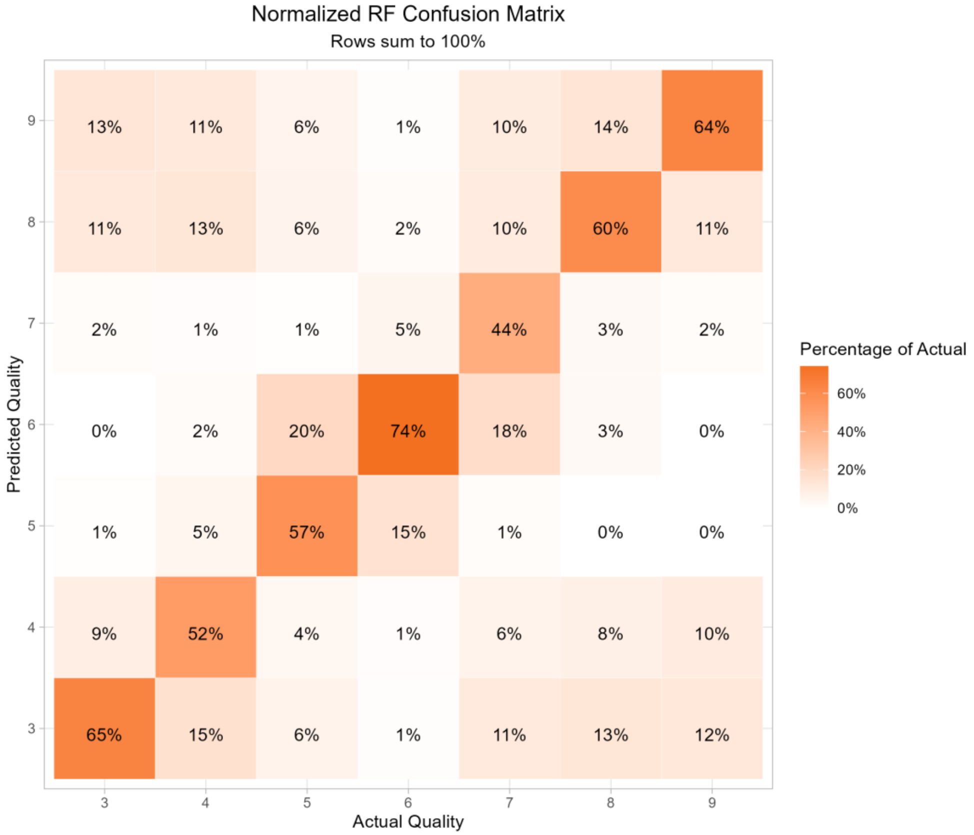
Key Insights:

- **Residual_sugar**, **density** and **free_sulfur_dioxide** are very important in our RF model.
  - Removing these 3 features would decrease the model precision by 0.375.
- The created dummy variable **is_white** has little predictive power, yet its strength is derived from **interaction effects**.
- No **dominant feature**:
  - No single feature is overly dominant. Wine quality is complex, which results from subtle interactions between many chemical properties and processes.

Random Forest performance.

The model does reasonably well, but the performance in the tails is worrying.

Confusion Matrix:

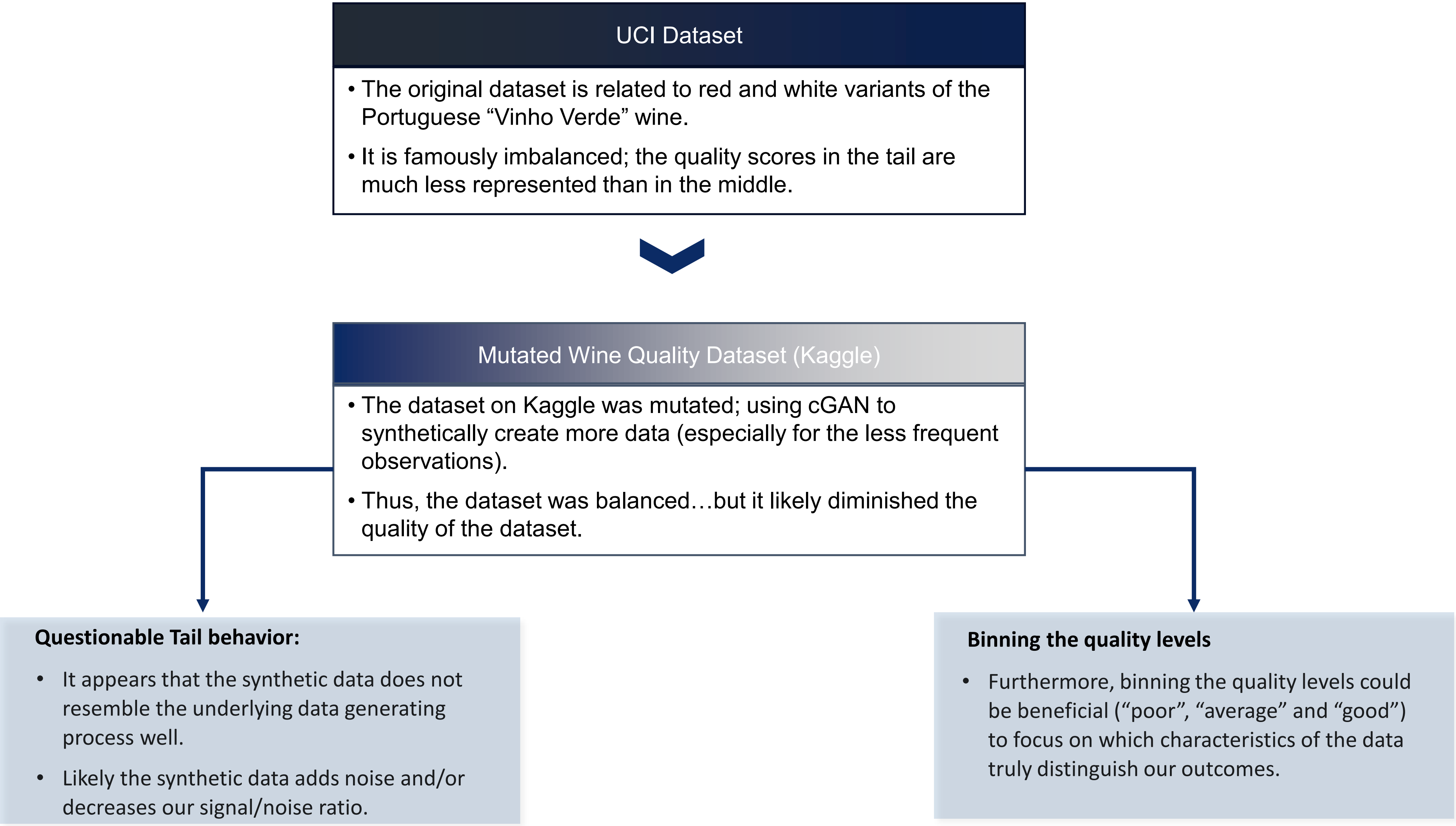


Key Insights:

- The model has a **poor learning capability** on the **tails**
  - Fairly low confidence in predicting the classes 3,4 and 8 and 9.
  - Tends to misclassify them, in some cases, on the other extreme.
- The RF model is **reasonably good** at identifying the **classes** in the **middle**.
- **Struggle's** to **accurately distinguish** between 5,6 and 7 but performs well if the true quality is 6.

Modeling performance.

Final considerations on the poor modeling of the tail behavior.



UCI Dataset

- The original dataset is related to red and white variants of the Portuguese “Vinho Verde” wine.
- It is famously imbalanced; the quality scores in the tail are much less represented than in the middle.

Mutated Wine Quality Dataset (Kaggle)

- The dataset on Kaggle was mutated; using cGAN to synthetically create more data (especially for the less frequent observations).
- Thus, the dataset was balanced…but it likely diminished the quality of the dataset.

Questionable Tail behavior:

- It appears that the synthetic data does not resemble the underlying data generating process well.
- Likely the synthetic data adds noise and/or decreases our signal/noise ratio.

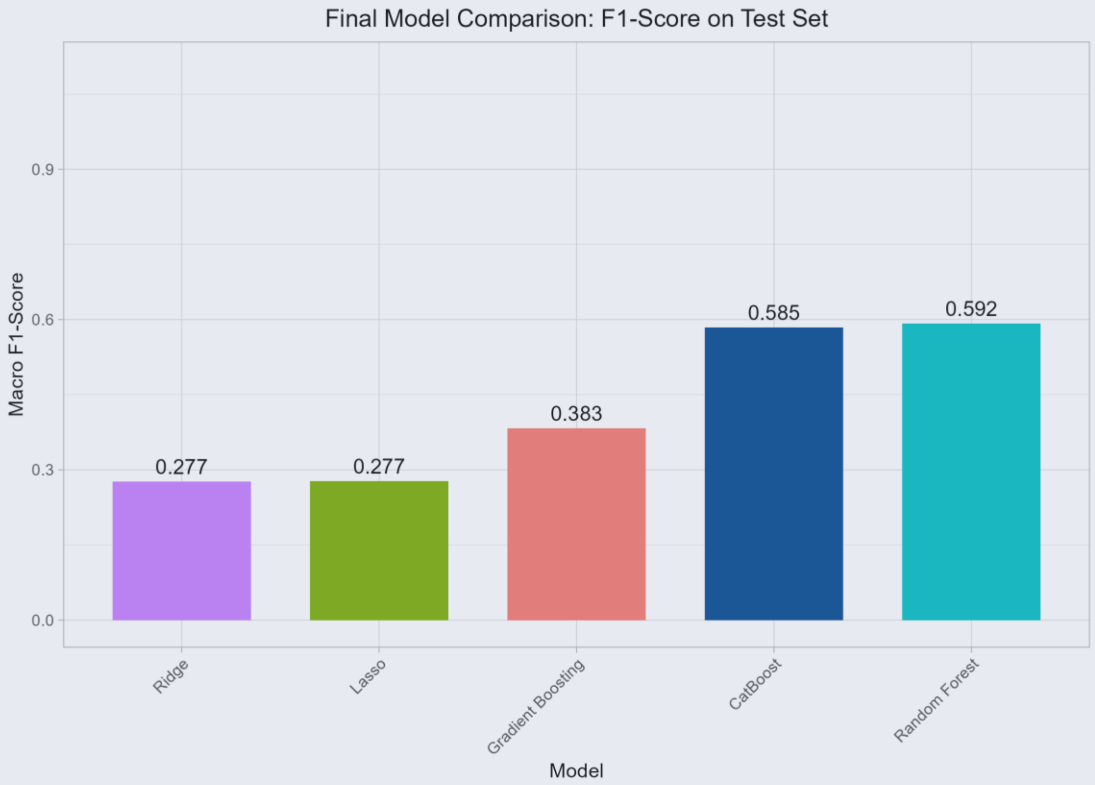
Binning the quality levels

- Furthermore, binning the quality levels could be beneficial (“poor”, “average” and “good”) to focus on which characteristics of the data truly distinguish our outcomes.

Extension with CatBoost.

Model performance including CatBoost.

Performance Comparison:

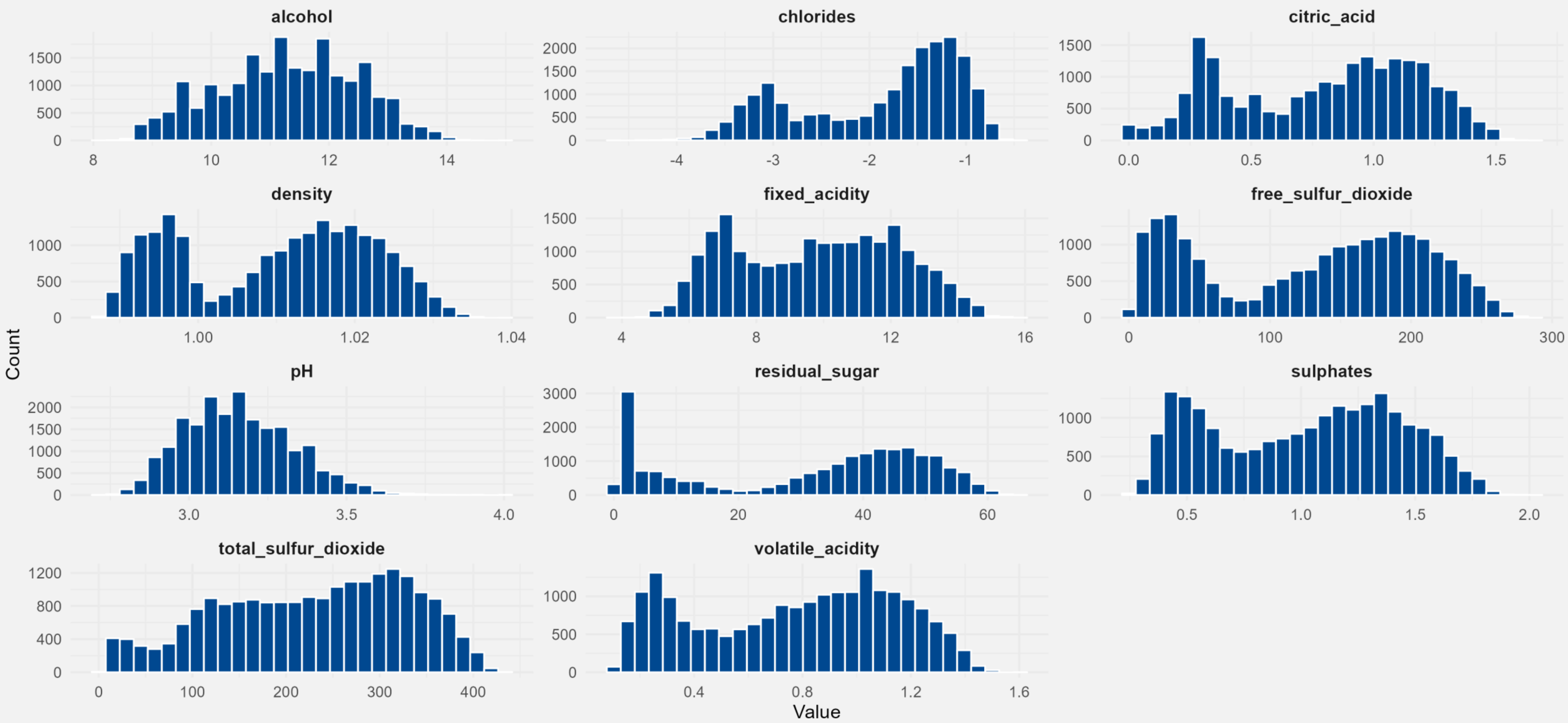


Key Insights:

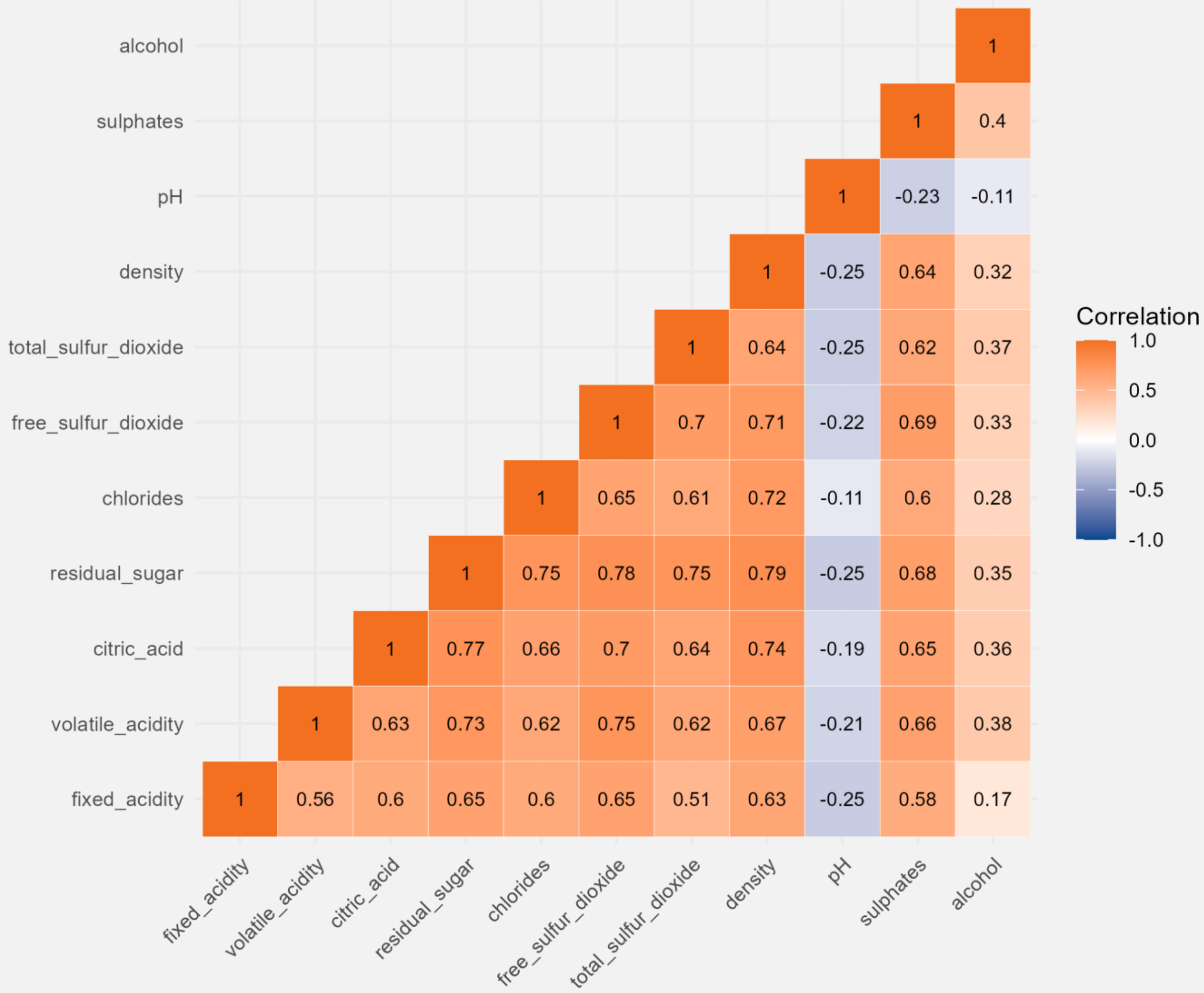
- **Random Forest is still the winner**
  - With a Macro F1-score of 0.592, the Random Forest model shows a moderate predictive power.
  - It performs much better than random guessing but is not yet highly accurate.
- **CatBoost gets fairly close!**
  - The test-score of 0.585 is very close to the one of the RF-model (0.592).
  - It's conceptually similar to GBM, but has, amongst others, a built in Ordered-Boosting feature, which acts as a regularization tool to prevent our model from learning from the weird synthetic data (in the edge-cases).

Even CatBoost cannot enhance our predictive power.
But we are close to the best result on Kaggle (which uses accuracy)!

Feature distribution.

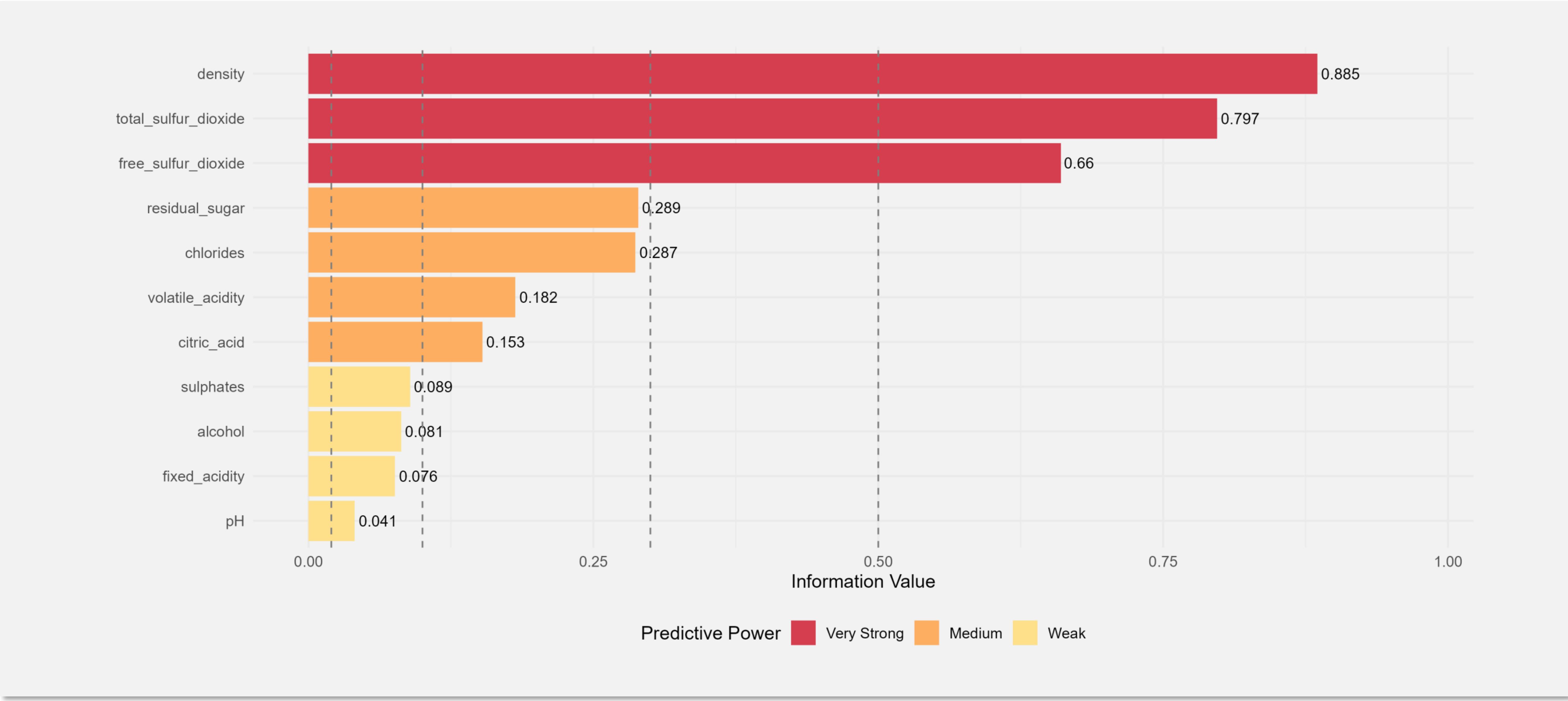


Multicollinearity.



	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	free_sulfur_dioxide	total_sulfur_dioxide	density	pH	sulphates	alcohol
alcohol											1
sulphates										1	0.4
pH									1	-0.23	-0.11
density								1	-0.25	0.64	0.32
total_sulfur_dioxide							1	0.64	-0.25	0.62	0.37
free_sulfur_dioxide						1	0.7	0.71	-0.22	0.69	0.33
chlorides					1	0.65	0.61	0.72	-0.11	0.6	0.28
residual_sugar				1	0.75	0.78	0.75	0.79	-0.25	0.68	0.35
citric_acid			1	0.77	0.66	0.7	0.64	0.74	-0.19	0.65	0.36
volatile_acidity		1	0.63	0.73	0.62	0.75	0.62	0.67	-0.21	0.66	0.38
fixed_acidity	1	0.56	0.6	0.65	0.6	0.65	0.51	0.63	-0.25	0.58	0.17

Informational Value.



Dummy variable wine type.

1. Clustering

- We partition the data into two distinct groups via K-means clustering (k=2).

2. Feature selection

- The most differentiating features (total_sulfur_dioxide, chlorides and volatile_acidity) are selected for clustering and separation. Prior to that, the data was scaled to prevent an overarching influence from one variable with greater ranges than the others.

3. Cluster identification

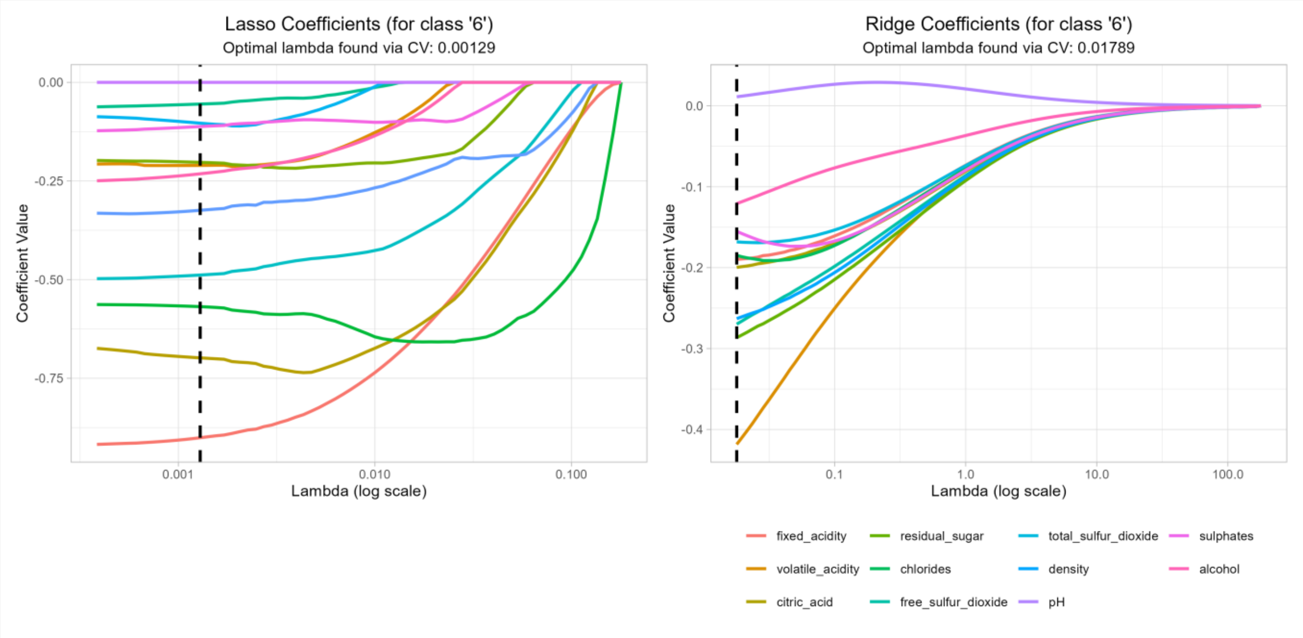
- The cluster with a high mean of total_sulfur_dioxide and low mean chlorides was identified as a white wine and vice versa as a red wine.

4. Variable engineering

- A new binary variable is created, indicating white wine (=1) or red wine (=0).

Performance and feature selection for Lasso and Ridge regression.

- Acccuracy for cross-validation across a range of lambda values.
- *Selecting the right model (comparing Lasso vs RF) with Macro F1-score.*
- *Macro F1-score and Accuracy are highly correlated with a balanced dataset.*



Lasso Coefficients (for class '6')

Optimal lambda found via CV: 0.00129

Ridge Coefficients (for class '6')

Optimal lambda found via CV: 0.01789

Performance and feature selection for RF.

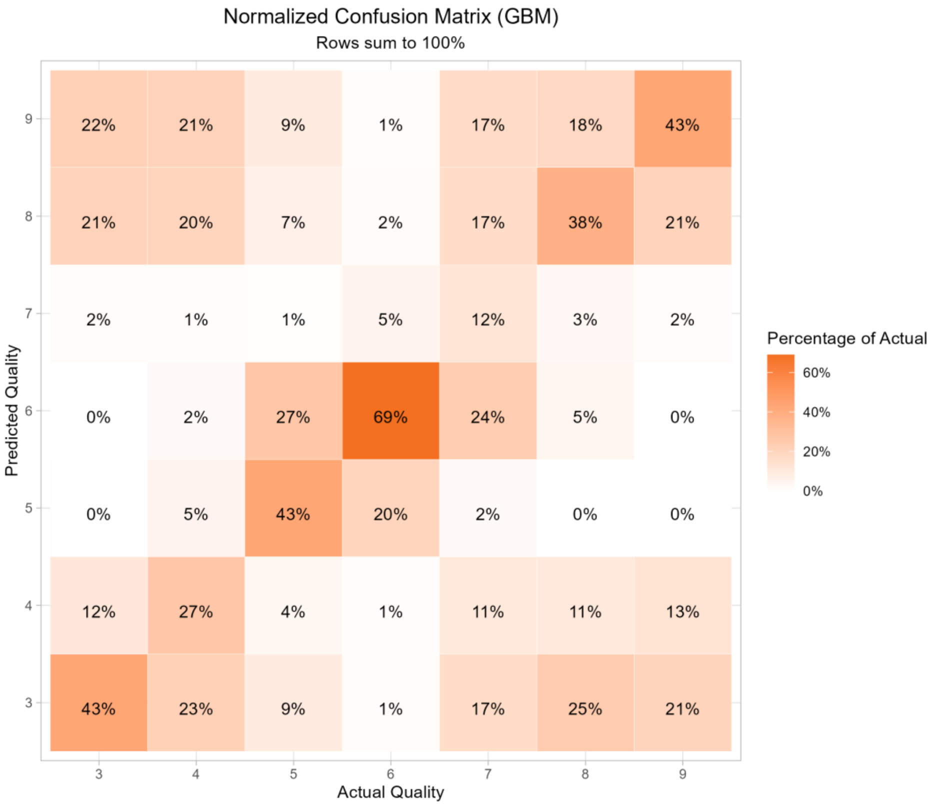
- Internal loss measure for optimization is done on gini impurity on each split, default for classification in the ranger package.
- Optimization to minimize multinomial deviance.
- Cross-validation: Hyperparameter grid-search tunes to maximize the mean macro F1-score across the folds.
- Final evaluation is done on macro F1-score.

- Tested hyperparameters:
- Number of variables to sample at each split: 2,3,5, and 6
- Min. size of a terminal node: 1,3,5 and 7.

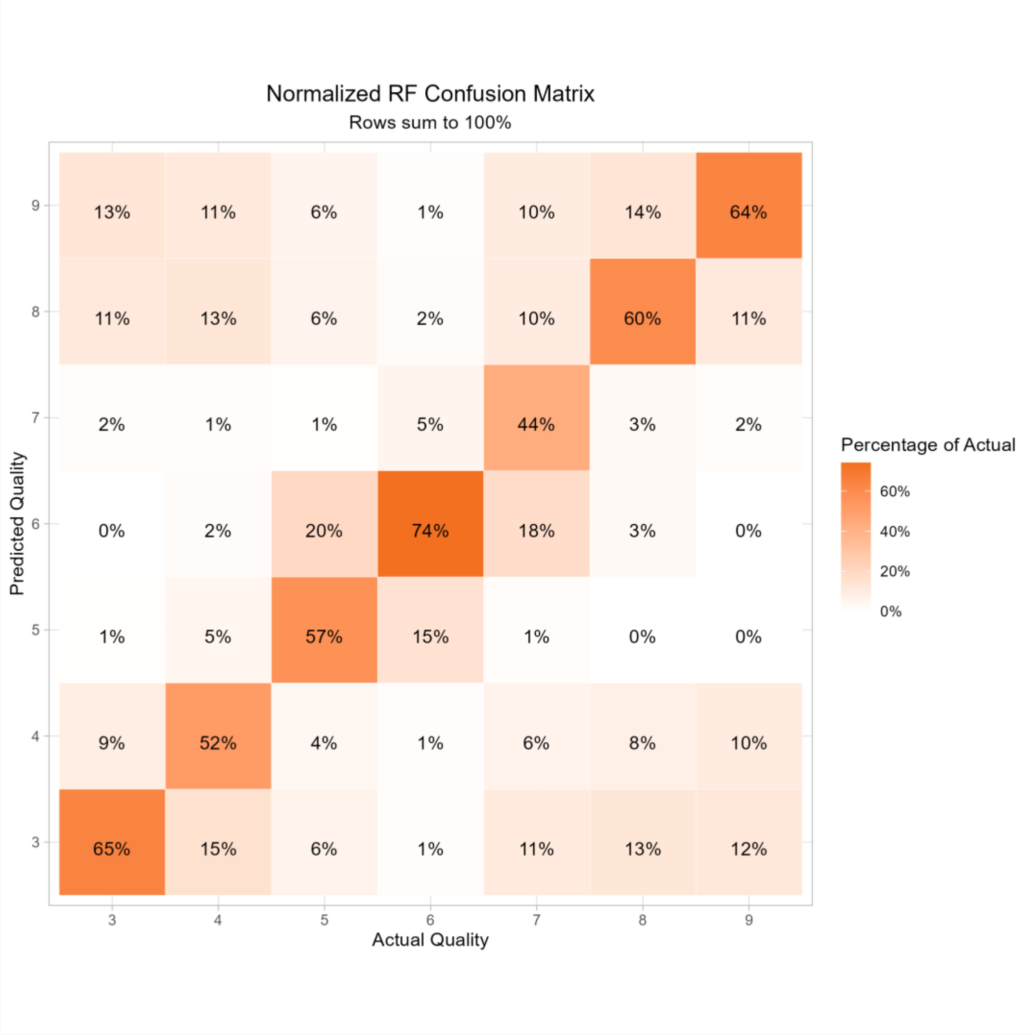
Performance and feature selection for GBM.

- Optimization to minimize multinomial deviance.
- Cross-validation: Hyperparameter grid-search to maximize the mean macro F1-score across the folds.
- Final evaluation is done on macro F1-score.

- Tested hyperparameters:
- Tree depth: 4,6 and 8.
- Learning Rate: 0.01 and 0.05
- Minimum observations in a node: 10, 15
- Number of trees: 2’000 and selecting the optimal number based on the out-of-bag error.



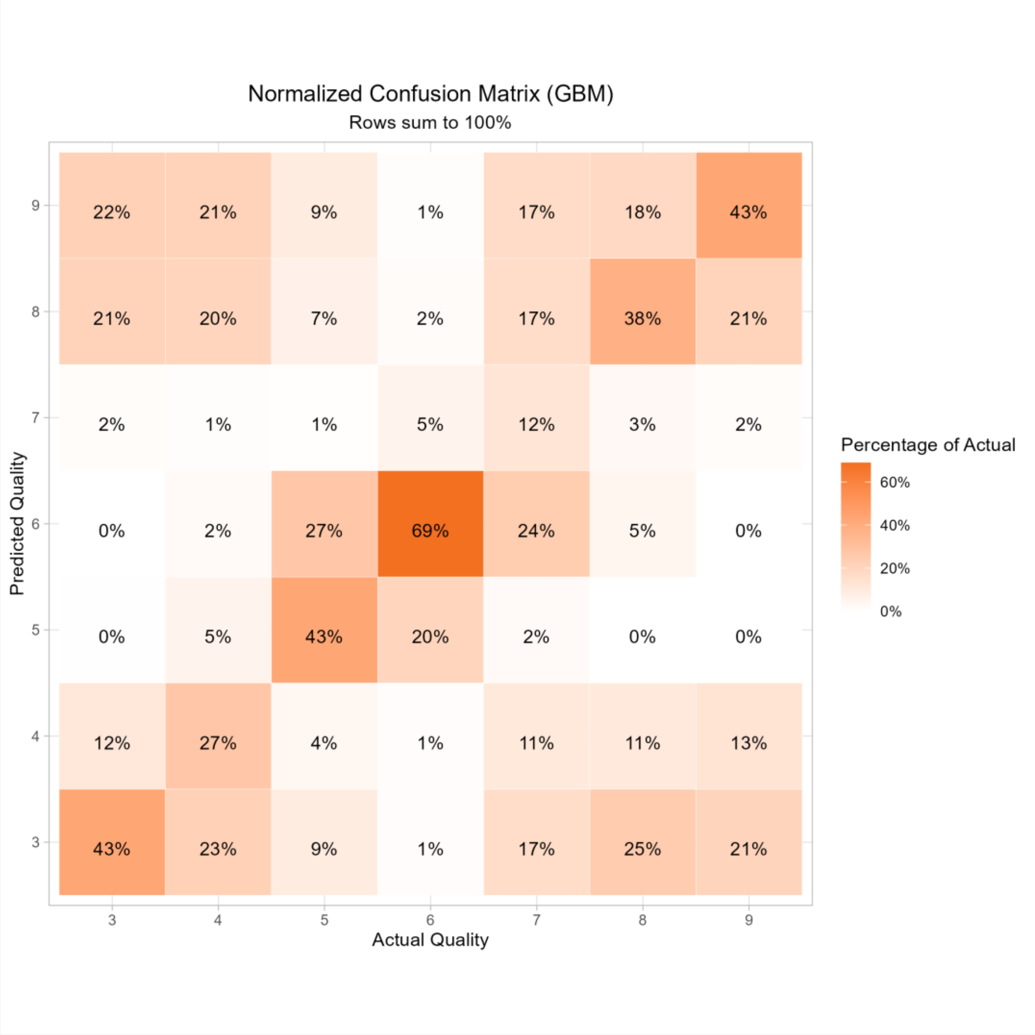
RF vs GBM



Normalized RF Confusion Matrix

Rows sum to 100%

Predicted Quality \ Actual Quality	3	4	5	6	7	8	9
9	13%	11%	6%	1%	10%	14%	64%
8	11%	13%	6%	2%	10%	60%	11%
7	2%	1%	1%	5%	44%	3%	2%
6	0%	2%	20%	74%	18%	3%	0%
5	1%	5%	57%	15%	1%	0%	0%
4	9%	52%	4%	1%	6%	8%	10%
3	65%	15%	6%	1%	11%	13%	12%



Normalized Confusion Matrix (GBM)

Rows sum to 100%

Predicted Quality \ Actual Quality	3	4	5	6	7	8	9
9	22%	21%	9%	1%	17%	18%	43%
8	21%	20%	7%	2%	17%	38%	21%
7	2%	1%	1%	5%	12%	3%	2%
6	0%	2%	27%	69%	24%	5%	0%
5	0%	5%	43%	20%	2%	0%	0%
4	12%	27%	4%	1%	11%	11%	13%
3	43%	23%	9%	1%	17%	25%	21%