Red Wine Quality Prediction

Project Report

Name	-	Matrikelnummer
Paul Wecker	-	943726
Robert Wienröder	-	948215
Vipin Singh	-	945801

Machine Learning II - Prof. Dr. Timothy Downie
Master Data Science
Berliner Hochschule für Technik

Abstract

This report presents a comparative analysis of two non-linear machine learning techniques, Projection Pursuit Regression (PPR) and Spline Smoothing. Both methods are non-linear methods and capable of modelling complex relationship. They were applied to a dataset of physiochemical properties of different types of red wine, with the objective of predicting wine quality. The project addresses the challenge of data imbalance through various preprocessing methods like stratified sampling, oversampling, undersampling and inverse frequency weighting. Additionally, we used the mean MSE over all quality classes (classes being the discrete values of the quality scale). After hyperparameter optimization on number of terms for PPR and on degrees of freedom for Spline Smoothing, the preprocessing method(s) yielding the model with the best predictive performance was chosen. Since Spline Smoothing only works with one predictor variable, additionally the variable with the best model had to be selected. PPR's effectiveness in handling complex, multidimensional relationships leads to its superior prediction performance over Spline Smoothing for red wine quality (PPR with 3 terms, mean MSE 0.93 vs. Spline Smoothing with 4 degrees of freedom on predictor variable "volatile acidity", mean MSE 1.46; both trained on over- and undersampled data). All in all, the importance of preprocessing as well as the distinct characteristics of each method in predictive modeling are shown.

Contents

Chapter 1

Introduction

1.1 Description of our code-structure

Before we start with the main report we would like to describe our projects code structure and the included files. This is the link to our GitHub repository in which we have all of our source code, data, rendered plots and additional files related to this project: https://github.com/pwckr/mltwo-project

We work with the R-command *source()* to call other scripts within a script. Therefore we could split up the files and prevent redundancy and over long files. To execute the R-code it is enough to just run the "final_comparison.R"-file, since this will call everything else. You just need to make sure that your working directory is the main-folder of the project (so one layer above *src/* and *data/*).

This is a brief description of our code:

File name	Description
setup.R	Setups the project with data partitions, seeds and some variables
helper_functions.R	Defines functions used throughout the project, such as for sampling, visualing
imbalance_plot.R	Creates the histogram of the quality values
pursuit_projection.R	PPR model optimizing, fitting, selection
spline_smoothing_pca.R	Spline Smoothing model optimizing, fitting, selection for variable *PC1*
spline_smoothing_alcohol.R	Spline Smoothing model optimizing, fitting, selection for variable *alcohol*
spline_smoothing_density.R	Spline Smoothing model optimizing, fitting, selection for variable *density*
spline_smoothing_pH.R	Spline Smoothing model optimizing, fitting, selection for variable *pH*
spline_smoothing_residualsugar.R	Spline Smoothing model optimizing, fitting, selection for variable *residual.sugar*
spline_smoothing_volacidity.R	Spline Smoothing model optimizing, fitting, selection for variable *volatile.acidity*
final_spline.R	Model selection among the spline models (only on validation set)
final_comparison.R	Model selection among best spline and ppr model (on test set) - **This file will call all the others**

Table 1.1: Description of files in the project

1.2 Project description

This report documents the results of a group project in the "Machine Learning II" course taught by Timothy Downie in the winter term of 2023/24 as part of the Data Science Master's program at Berliner Hochschule für Technik. The project compares two supervised machine learning methods, Projection Pursuit Regression and Spline Smoothing. Both methods were used to create models on a medium-sized dataset, predicting the quality of red wine. The main goal was to evaluate and compare the predictive performance of these methods.

1.2.1 Data Selection and Preprocessing

The "Red Wine Quality" dataset containing the physiochemical attributes of various red wines was chosen from the website of Kaggle, an online community of data scientists. This dataset meets the criteria of containing 1000 to 2000 instances and at least five predictor variables plus one target variable (wine quality in this case). The data was provided as a data frame in a standard .csv format. It was split into subsets for training (60%), validation (20%), and testing (20%). Beforehand, duplicate rows were removed and regarded as erroneous as the high number of continuous predictor variables made it seem very unlikely to encounter wines with the same properties. The validation subset was used for Hyperparameter Optimization (HPO) and model selection, while the test subset was exclusively reserved for final method evaluation and comparison.

1.2.2 Machine Learning Methods

Regression and classification methods would have been both possible since the target variable is numeric but discrete. The decision to use regression techniques was made primarily due to their more nuanced predictions. The methods selected for this project were *Projection Pursuit Regression (PPR)* and *Spline Smoothing*. PPR was chosen mainly because of its capability to identify combined non-linear effects of predictor variables in multidimensional data. The reason for selecting Spline Smoothing was its ability to fit smooth curves to non-linear complex relationships between a predictor and a target variable. This selection spawned our interest in seeing how far a non-linear method utilizing only one variable would work for modeling the quality level.

1.2.3 Model Fitting and Evaluation

The first data exploration showed a significant imbalance (see section 1.4) with a strong over-representation of middle-range wine qualities. Different techniques like stratified sampling, combined over- and undersampling, and inverse frequency weighting were used to deal with this imbalance. Hyperparameter optimization was conducted for the number of terms (PPR) and degrees of freedom (Spline Smoothing). To account for the imbalance in validation, the primary metric for model evaluation was the mean MSE over all classes (classes being the unique discrete values of wine quality). Using the techniques dealing with imbalance, along with the HPO process, we fitted numerous models, out of which we opted for the one with the lowest mean MSE over all classes (on the validation set). Finally, we compare the best PPR model with the best spline smoothing model using the test set created earlier.

1.3 Data description

- **Name of the dataset:** Red Wine Quality
- **Number of observations:** 1599
- **Number of predictor variables:** 11
- **Number of target variables:** 1

- **Number of duplicate rows:** 240
- **Number of unique observations:** 1359

Synopsis

The dataset can be found on Kaggle or in our references [1]. It contains physiochemical properties of different types of red wine and a quality score (target variable). The dataset scores are not distributed equally, since there are a lot more "normal" quality wines than "poor" or "excellent" quality wines.

Variables

Feature	Type	Description
Fixed Acidity	Continuous	Concentration of non-evaporating acids in wine
Volatile Acidity	Continuous	Amount of acetic acid (high amounts lead to vinegar-like taste)
Citric Acid	Continuous	Adds freshness and flavor; present in small amounts.
Residual Sugar	Continuous	Sugar remaining after fermentation, affecting sweetness
Chlorides	Continuous	Amount of salt in wine
Free Sulfur Dioxide	Continuous	Prevents microbial growth and oxidation
Total Sulfur Dioxide	Continuous	High amounts lead to different smell and taste
Density	Continuous	Mass per volume unit, influenced by alcohol and sugar content
pH	Continuous	Indicates acidity and basicity level
Sulphates	Continuous	Antimicrobial and antioxidant, contribute to sulfur dioxide level
Alcohol	Continuous	Alcohol level by volume
Quality	Discrete	Target variable

Table 1.2: Red Wine Quality Dataset Features

1.4 Data imbalance

As previously mentioned in Section 1.2, the first data exploration showed a strong imbalance in the data. Of the 1599 observations, only 10 or 0.006% were assigned a quality rating of 3, whereas 82% of the observations had ratings of 5 or 6. The following histogram (Figure 1.1) depicts the distribution of quality ratings for the red wines:

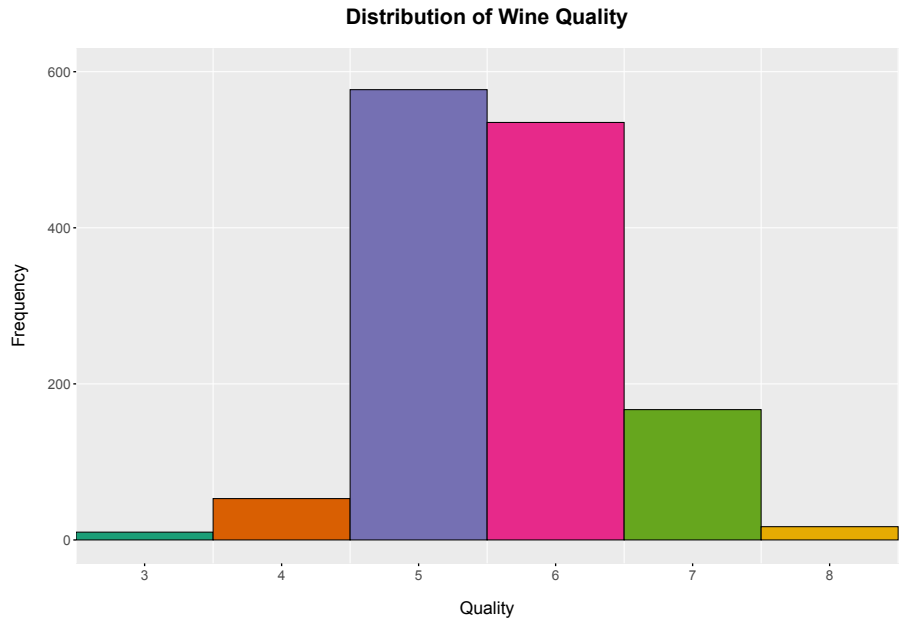


Figure 1.1: Distribution of Quality Ratings in the Red Wine Dataset

Despite the application of PPR and Spline Smoothing for continuous outcomes, the quality scores were discrete, as mentioned. Considering the target variable's imbalance, conventional regression metrics like the overall MSE would have had to result in misleading conclusions by under-representing prediction errors for the less frequent quality scores. Because of that, the integer values were regarded as distinct "classes" to enable us to use class-specific weighting and the mean of the Mean Squared Error (MSE) over all classes.

Methodologies for Addressing Imbalance

- **Stratified Sampling:** Random sampling resulted in under-representation of observations with quality ratings of 3, 4, 8 and in some cases also 7 in the subsets. To ensure the presence of each class in the training, validation, and testing set, stratified sampling was used.
- **Oversampling:** Synthetic Minority Over-sampling Technique (SMOTE) from the smotefamily package [2] for oversampling the minority classes was applied. SMOTE, a technique to over-sample, which is well acknowledged since it was published in 2002 by Chawla et. al [3], generates synthetic samples for classes, increasing them to balance the class distribution.
- **Undersampling:** A custom undersampling function was developed. Classes exceeding the size of the smallest class were randomly reduced to match the size, again resulting in a more balanced class distribution. Experimenting with undersampling showed that it generated very small training data sets. Because of that, in the end both, under- and oversampling, were used simultaneously.
- **Hybrid Sampling:** The issues of over- and under-sampling were tackled by using both methods in one custom function and call it "hybrid sampling". This way of sampling resulted in a fairly balanced training data set which will be referred to hereafter as simply "balanced data".

- **Inverse Frequency Weighting:** In cases in which hybrid sampling failed to create a dataset with the correct target sizes, presumably due to rounding reasons, an additional tool to make sure each class was weighted equally was required. Another custom function was created, with the weights being calculated as the total number of observations divided by the frequency of each class. This resulted in weights being assigned inversely proportional to class frequencies.
- **Mean MSE over classes:** To counter the bias towards frequently occurring classes, the mean MSE across all classes was calculated for validation. Like this, prediction errors were accounted for also in less frequent classes.

Chapter 2

Projection Pursuit Regression

The first applied machine learning method is Projection Pursuit Regression (PPR). This chapter introduces PPR mathematically, then describes the model fitting process including hyperparameter optimization and lastly assesses the performance of the different fitted models.

2.1 Mathematical Overview

Projection Pursuit Regression (PPR) is an advanced regression technique that extends Generalised Additive Models (GAM) [4]. GAM are functions that consist of different kinds of regression functions ("sub-functions") being added together. PPR extends this concept by allowing the different sub-functions to be any linear combination of the predictor variables. The emerging functions are called ridge functions. They are especially useful with non-linear, complex relationships between variables in datasets. PPR detects and uses these complex structures by projecting the predictor variables into a lower-dimensional space.
The projections of the predictor variables X in directions defined by the vectors ω_m are represented by V_m:

$$V_m = \omega_m^\top X \tag{2.1}$$

The ω_m vectors are chosen to capture the most "interesting" or informative aspects of the data with respect to the response variable Y. Each V_m is essentially a linear combination of the predictor variables, and these combinations are used in the ridge functions g_m to model the response. The idea is that these projections can reveal underlying structures in the data that might not be apparent in the original predictor space. The model can be fitted like this:

$$Y \approx f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m g_m(V_m) \tag{2.2}$$

where:

- Y is the response variable
- $X = (X_1, X_2, \ldots, X_p)^\top$ is the vector of predictor variables
- β_0 is the constant parameter
- β_m are the coefficients for each ridge function
- M is the number of ridge functions
- g_m are the ridge functions
- $V_m = \omega_m^\top X$ are the projections of X with direction vectors ω_m

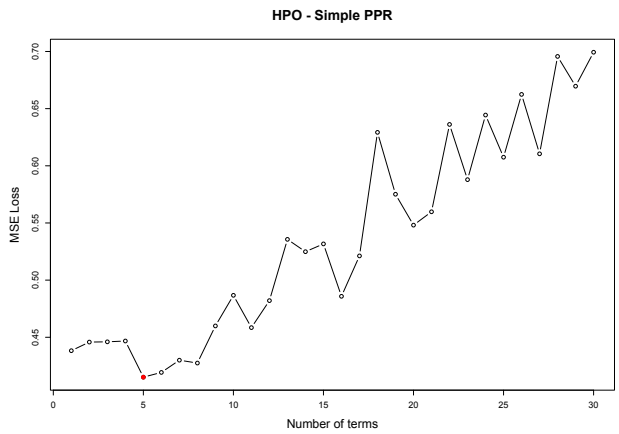
PPR's strength lies in its ability to model complex, nonlinear relationships by combining these projections and ridge functions in a flexible way. Each ridge function acts on a different aspect or feature of the data, as identified by the projections, allowing the model to adapt to various patterns in the data.

2.2 Fitting Process

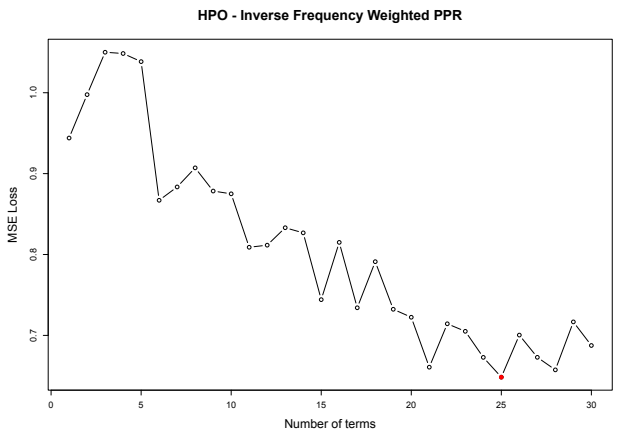
After the preprocessing described in Section 1.4, the PPR model was fitted iteratively on different values of the critical hyperparameter. This process was repeated four times on the same dataset, preprocessed in four different ways: First without any preprocessing done, second with inverse frequency weighting, third with over- and undersampling called "hybrid sampling" and fourth with hybrid sampling and weighting.

Hyperparameter Optimization

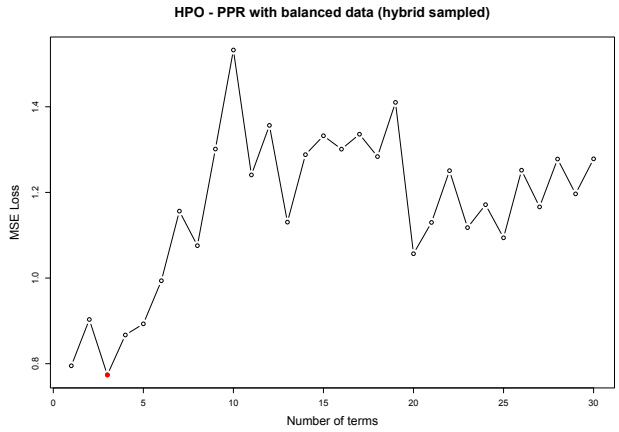
In the Hyperparameter Optimization section for the Projection Pursuit Regression (PPR) model, the primary focus was on the number of ridge functions g_m, representing different directions in the predictor space. Other hyperparameters were not extensively explored to simplify computation and minimize overfitting risks, but also because of limited knowledge. The initial range for the number of terms was set between 1 and 30. The models were iteratively fitted with various counts of ridge functions, and their performance was evaluated on a validation set using Mean Squared Error (MSE) as a metric. A plot was created to visualize the MSE against the number of terms, assisting in identifying the optimal term count for minimizing MSE. This approach was aligned with the criteria used in spline smoothing, ensuring consistency in the model fitting process. The optimal number of terms for the simplest PPR model was determined and is 5 (see Figure 2.1), based on the lowest MSE across all four preprocessing scenarios. Evaluation of the performance during HPO was based on the MSE metric, which was chosen here over the mean MSE over all classes to avoid giving the models too much freedom (like the freedom to fit every class perfectly), potentially resulting in overfitting.



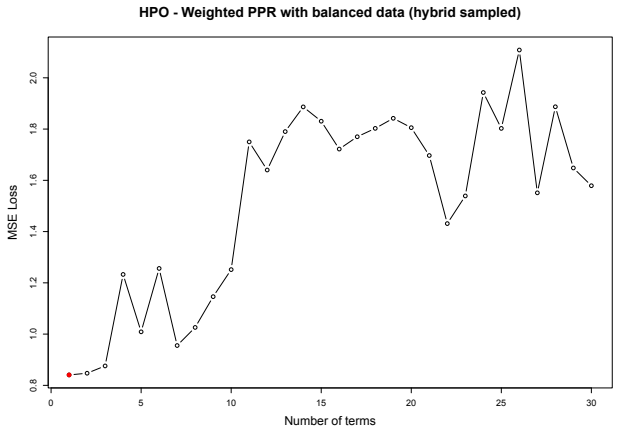
Optimal number of terms: 5



Optimal number of terms: 25



Optimal number of terms: 3



Optimal number of terms: 1

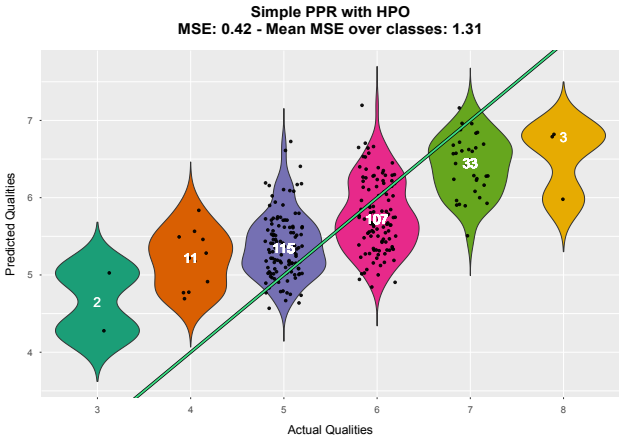
Figure 2.1: Hyperparameter Optimization: MSE Loss for Varying Term Counts in Pursuit Projection Regression with four different preprocessings

Final Model

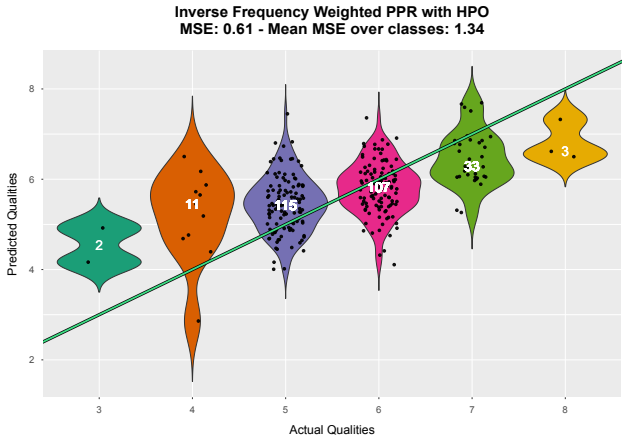
Once the optimal number of terms was identified, the PPR model was re-fitted to the training data using this hyperparameter value. The fitted model was then evaluated on the validation dataset. As evaluation metric the mean MSE over all classes was used (see Section 1.4)

2.3 Results

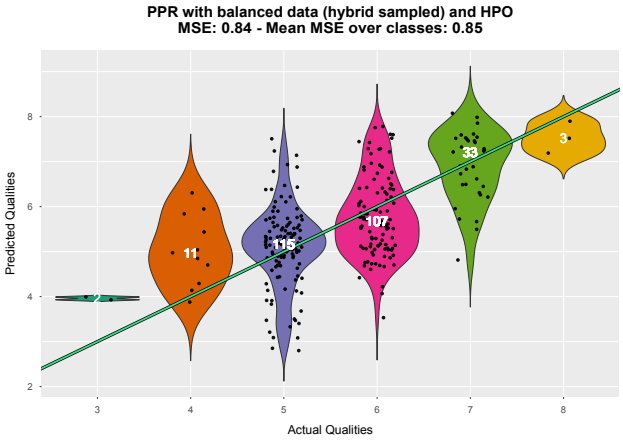
For each of the four preprocessing scenarios, we optimized the number of terms, to train the final models. For each of the four models the performance was evaluated on the validation set. The results are plotted below in Figure 2.2, with models in the second row using hybrid sampled data and those in the second column using weighted data. The actual quality ratings are contrasted with the predicted ones, for each class there is one violin plot. To make all datapoints visible, some jittering is applied. Each subplot has a different scale, which complicates comparative assessment, but makes the plot easier to look at. The line of the identity function $f(x) = x$ serves as a visual reference for the locations of "perfect predictions", making identical scaling unnecessary.



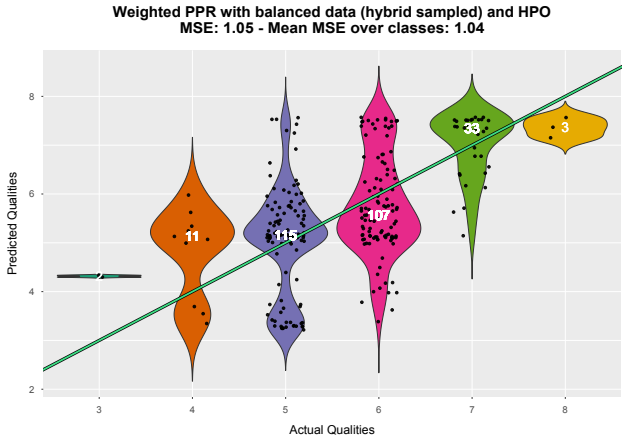
Number of terms: 5



Number of terms: 25



Number of terms: 3



Number of terms: 1

Figure 2.2: Prediction Analysis on Validation Set: Violin Plot of Predicted vs. Actual Values in Pursuit Projection Regression, with Line of Equality and Jittered Scatter Plot with four different preprocessings

The table of MSE values 2.1 shows that the lowest mean MSE across all classes is achieved by the PPR model using balanced data (a combination of over- and undersampling). As indicated in Figure 2.2, the optimal number of terms for this model configuration is 3.

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
Simple PPR	2.87	1.56	0.32	0.28	0.49	2.31	0.42	1.31
Inverse Frequency Weighted PPR	2.52	2.38	0.58	0.38	0.63	1.54	0.61	1.34
PPR with Balanced Data	0.92	1.63	0.85	0.83	0.60	0.30	0.84	0.85
Weighted PPR with Balanced Data	1.73	1.40	1.16	1.10	0.42	0.44	1.05	1.04

Table 2.1

Chapter 3

Spline Smoothing

As the second machine learning method Spline Smoothing was chosen. In this chapter, Spline Smoothing is introduced from a mathematical point of view without going into algorithmic details. After that, the selection of a model for each of the six selected predictor variables is shown (one of which is the first principal component).

3.1 Mathematical Overview

Spline Smoothing describes piece-wise defined regression functions constrained by properties that account for the resulting function's continuity and smoothness [5]. Smoothing Splines are related to B-Splines insofar as they are also piece-wise defined using similar basis functions for each piece. The basis functions should be differentiable at least twice (see cost function 3.2). First, the basis functions are restricted to compose a continuous function with a continuous first derivative:

$$\begin{aligned} f(c_k^-) &= f(c_k^+), \\ f'(c_k^-) &= f'(c_k^+) \end{aligned} \tag{3.1}$$

Second, the smoothness characteristic of the function is handled by an addition to the error term, conceptually similar to the addition in ridge regression with the difference that this addition punishes models with strong "wiggliness"(being the opposite of smoothness):

$$J(f, \lambda) = \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_{x_1}^{x_n} \left(f''(x)\right)^2 \, dx \tag{3.2}$$

Adding the integral of the squared second derivative of the spline function over the span of the single predictor variable X punishes functions with a "wiggly" curve as their slope changes quickly and thus they have more extreme values in the second derivative, as opposed to functions whose slope does not change as much. The factor λ controls the extent of the smoothness and thus allows for experimentation with the smoothness property.
Setting λ to ∞ leads to a second derivative of 0 and consequently to a linear function as a linear function's slope doesn't change. On the other hand, setting λ to 0 yields a function that interpolates the data and overfits it.
Therefore, finding a good smoothing spline model involves finding an appropriate λ that balances smoothness and goodness of fit. Besides λ, the order of the piece-wise polynomial (usually cubic) influences the smoothness of the model. As a higher order leads to more turning points of the curve, higher orders impose stronger wiggliness.

3.2 Fitting Process

Using smoothing splines to predict wine quality posed the task of choosing a predictor variable as one model can only account for one predictor variable. Out of the eleven predictor variables in the data set, five were picked: Alcohol, density and pH value for their graspability because most people know these terms without requiring domain knowledge; residual sugar and volatile acidity because their initial data exploration promised to have the most non-linear effect on the quality level. Additionally principal component analysis (PCA) was used to fit one model on the first principal component (PC1) of the training data. Using the first principal component, we tried to circumvent the limitation of spline smoothing only being able to utilize one predictor variable. This way, we hoped to concentrate as much information as possible in only one predictor variable, namely the first principal component. Summarizing, the chosen variables are:

- Alcohol
- Density
- pH value
- Residual Sugar
- Volatile Acidity
- PC1

Hyperparameter Optimization

Finding good models required controlling the smoothness of the model. In the chosen implementation, the smoothness is manipulated by the key hyperparameter degrees of freedom (df) (high value for df yields a model with strong wiggliness). This way, manipulating λ, which is usually on a less intuitive scale, is circumvented. Evaluation of the performance during HPO was based on the MSE metric, which again was chosen over the mean MSE over all classes to avoid giving the models too much freedom (like the freedom to fit every single class perfectly), potentially resulting in overfitting. For each model, the validation MSE loss for varying df was computed. Figure 3.1 showcases this investigation for the variable alcohol without preprocessing (simple case).

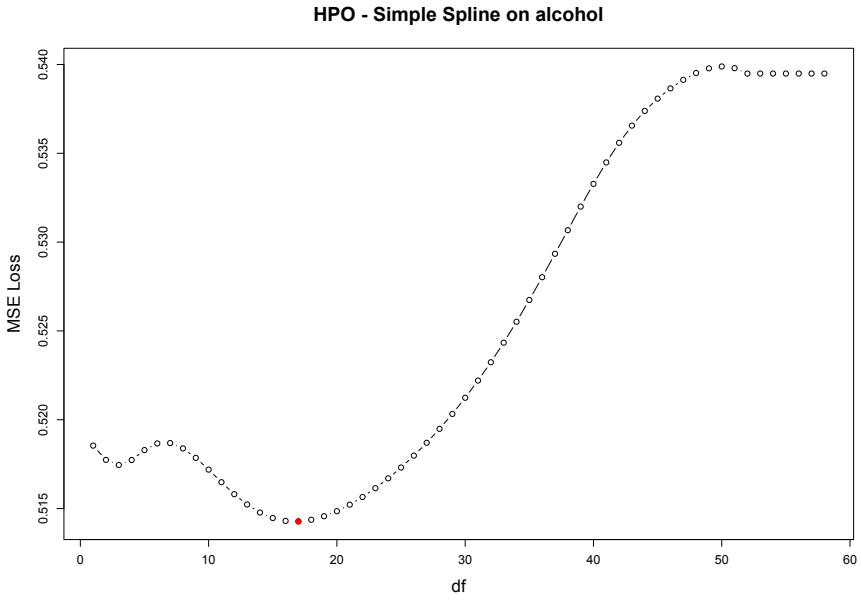


Figure 3.1: Optimization of Hyperparameters: MSE Loss for varying DF in Spline Smoothing Model

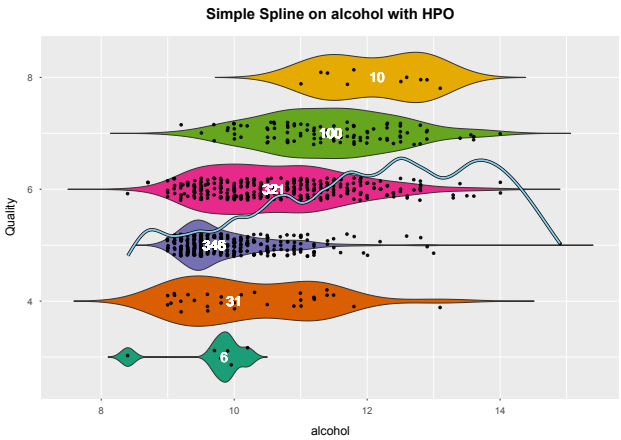
The figure illustrates well that there is an optimal choice for df in terms of the mean squared error (here 18).

To reduce the amount of showed plots for this report we decided to hide further MSE plots for the Hyperparameter optimizations and just write the optimal number found out by the optimization in the caption of the single splines.

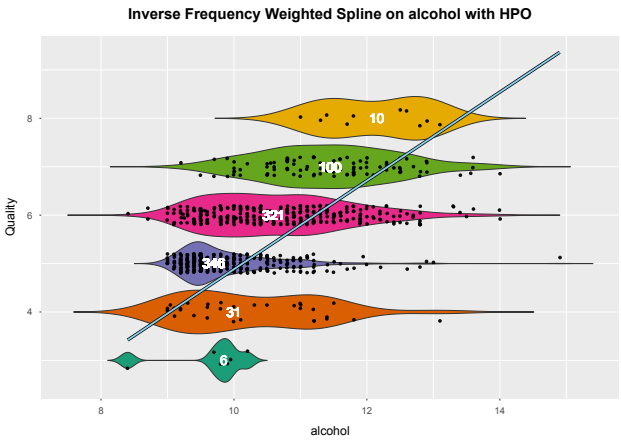
3.2.1 Alcohol vs Quality

The fitted spline of the four models below are plotted as thick blue lines, along with the training data, represented as "jittered" scatter plots, complemented with a violin plot of the training data to show the distibution for each quality level. Like for PPR, the HPO was conducted and the model was fitted four times on the same dataset, preprocessed in four different ways: First without any preprocessing done (except for the initial removal of duplicates and stratified sampling; top left), second with inverse frequency weighting (top right), third with over- and undersampling called "hybrid sampling" (bottom left) and fourth with hybrid sampling and weighting (bottom right).

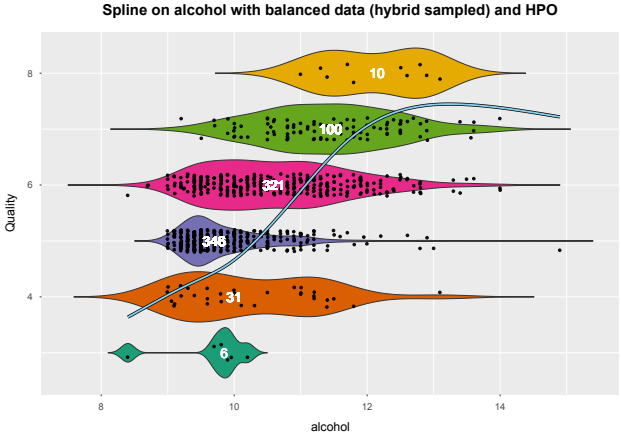
For alcohol, it is clear that no balancing or weighting results in a very wiggly model with a high DF; paired with the lowest validation MSE (top left). In the top right corner, simple weighting without hybrid sampling reduces the value of df to two, resulting in a curve resembling simple linear regression.



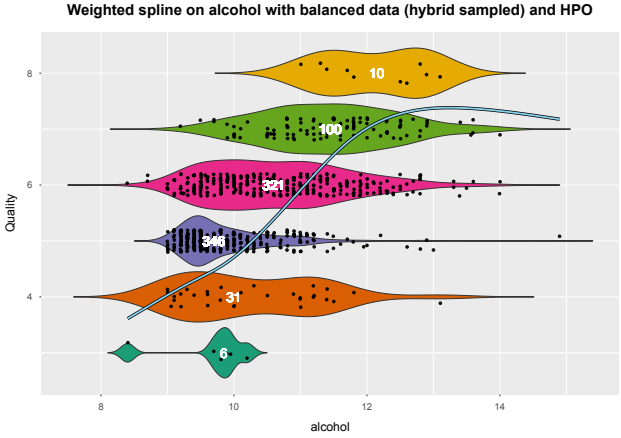
df = 18, val.- MSE = 0.51



df = 2, val.- MSE = 0.97



df = 4, val.- MSE = 1.09

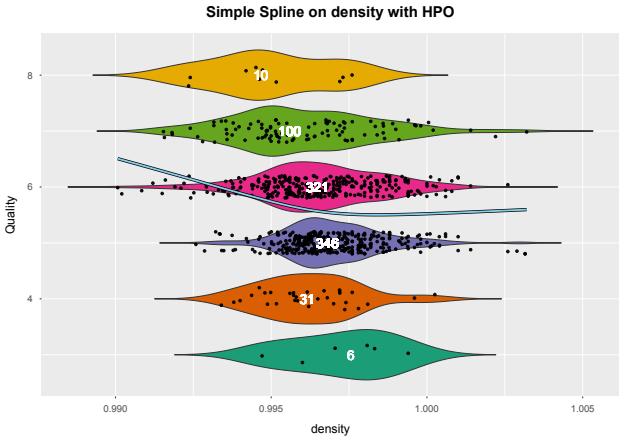


df = 4, val.- MSE = 1.05

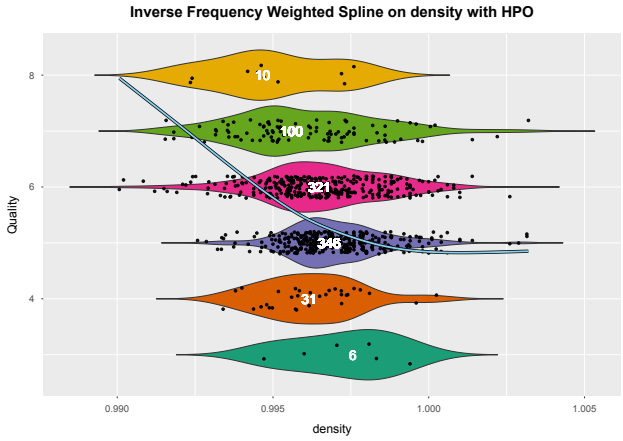
Figure 3.2: Fitted models (blue line) on variable "alcohol" and the training data

3.2.2 Density vs Quality

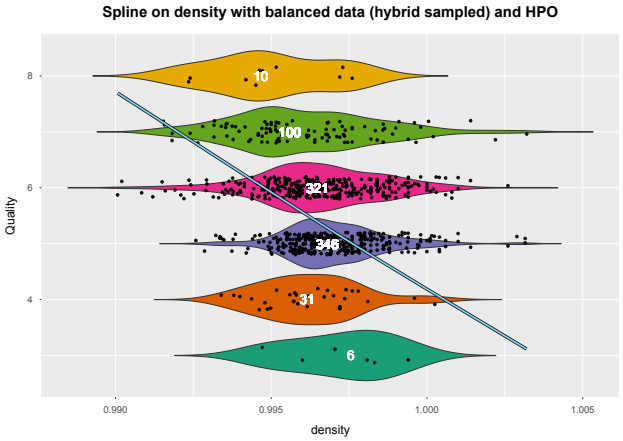
For density, the models using hybrid sampling only use a simple linear model, while the best validation MSE is achieved by the simple spline model.



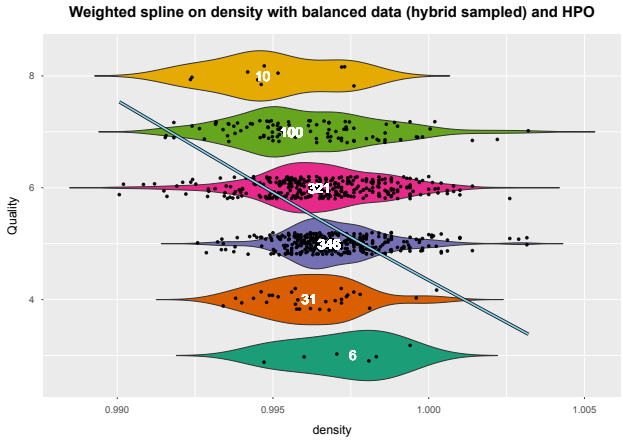
df = 3, val.- MSE = 0.63



df = 3, val.- MSE = 0.77



df = 2, val.- MSE = 0.94

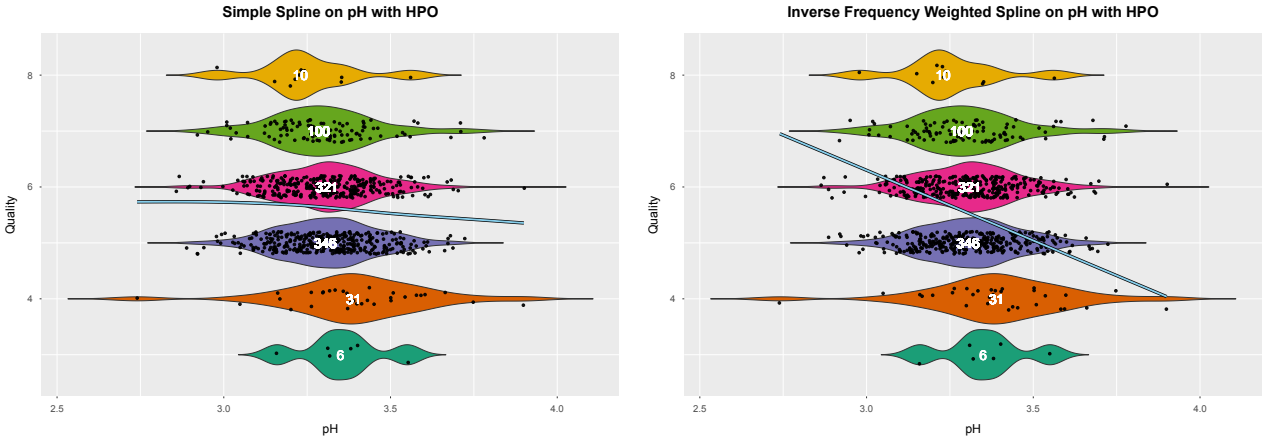


df = 2, val.- MSE = 0.86

Figure 3.3: Fitted models (blue line) on variable "density" and the training data

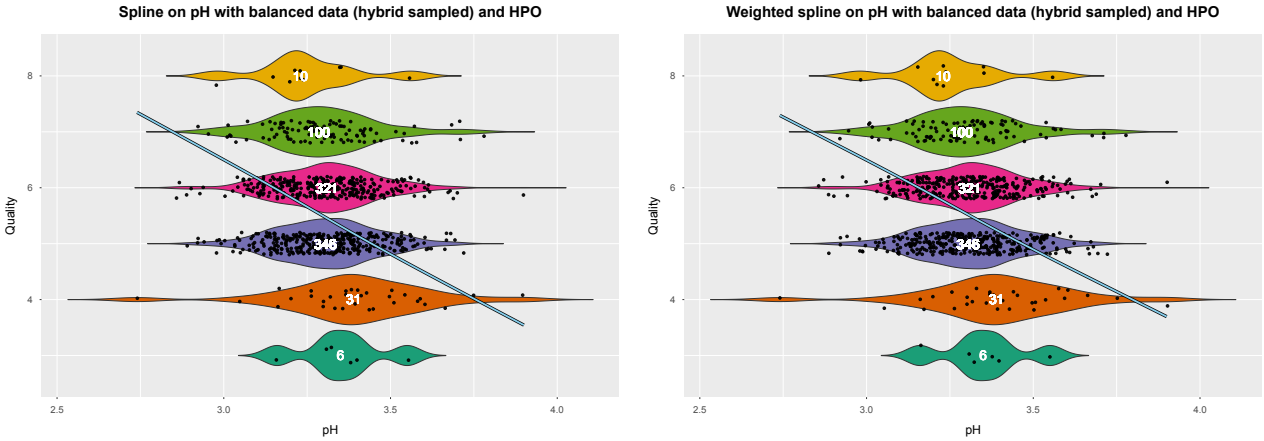
3.2.3 pH vs Quality

The models of the variable pH depict a similar effect as above: only the simple model allows some curvature while having the best validation MSE.



df = 3, val.- MSE = 0.67

df = 2, val.- MSE = 0.78



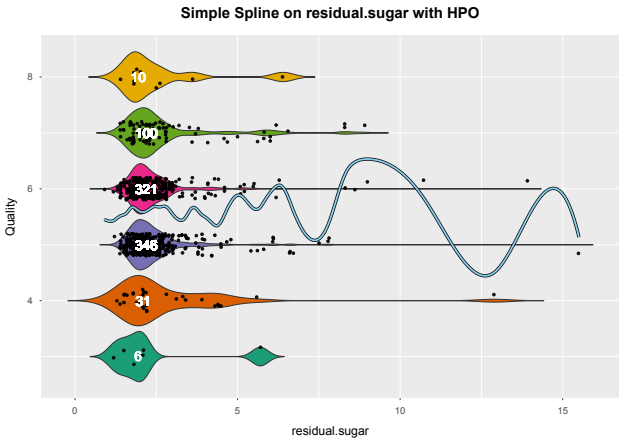
df = 2, val.- MSE = 0.89

df = 2, val.- MSE = 0.85

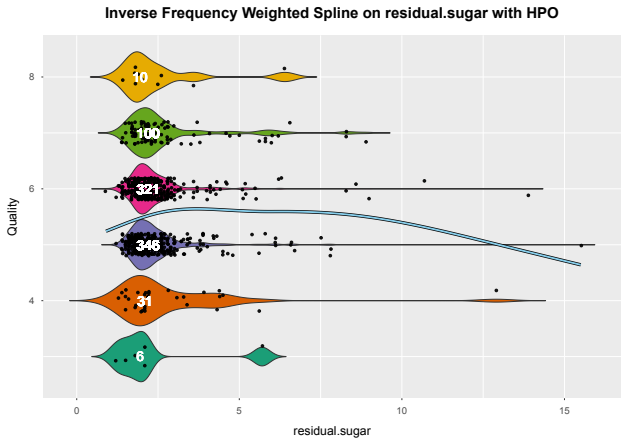
Figure 3.4: Fitted models (blue line) on variable ph and the training data

3.2.4 Residual.Sugar vs Quality

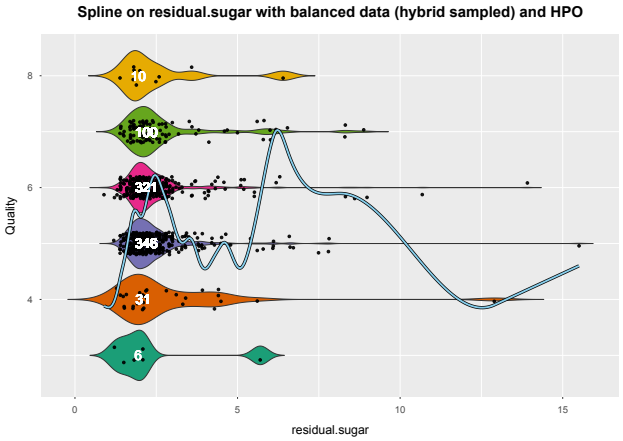
For the variable "residual sugar", only the weighted model displays a curve that looks reasonable.



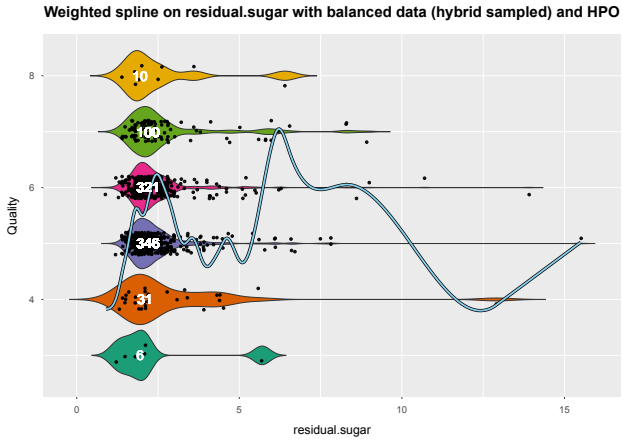
df = 19, val.- MSE = 0.67



df = 3, val.- MSE = 0.68



df = 16, val.- MSE = 0.83

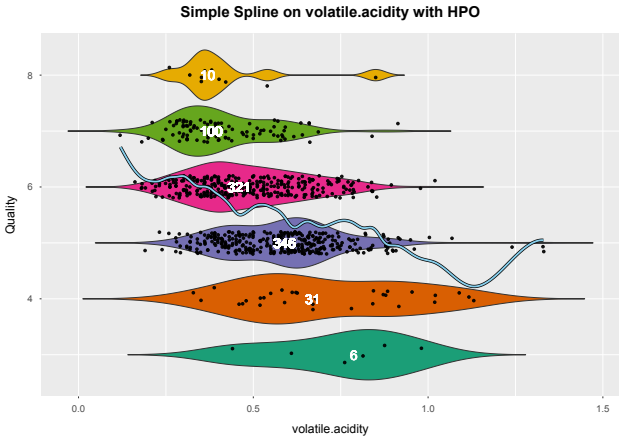


df = 16, val.- MSE = 0.82

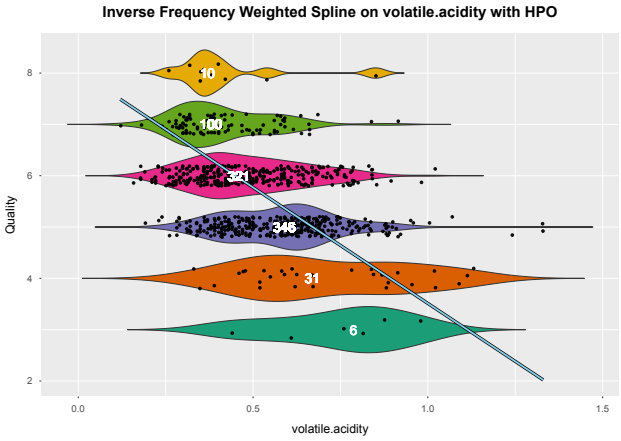
Figure 3.5: Fitted models (blue line) on variable residual sugar and the training data

3.2.5 Volatile.Acidity vs Quality

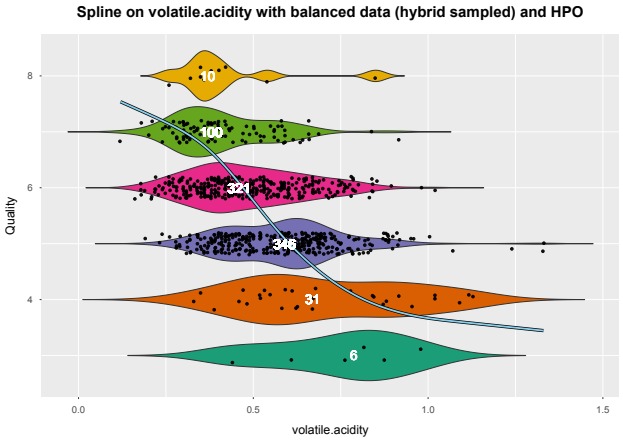
For "volatile acidity", the two hybrid sampled models seem to be very similar, while the simple model with a high degree of freedom looks quite "wiggly". The weighted model is approximately linear.



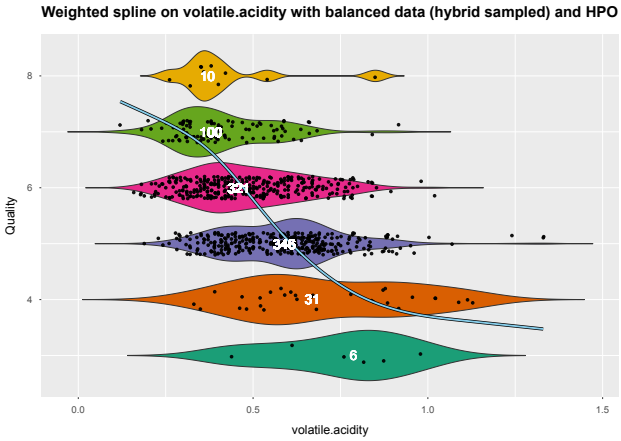
df = 21, val.- MSE = 0.54



df = 2, val.- MSE = 0.79



df = 4, val.- MSE = 1.02

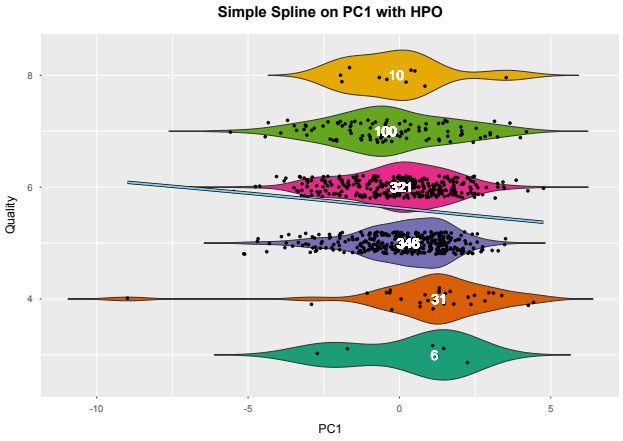


df = 4, val.- MSE = 1

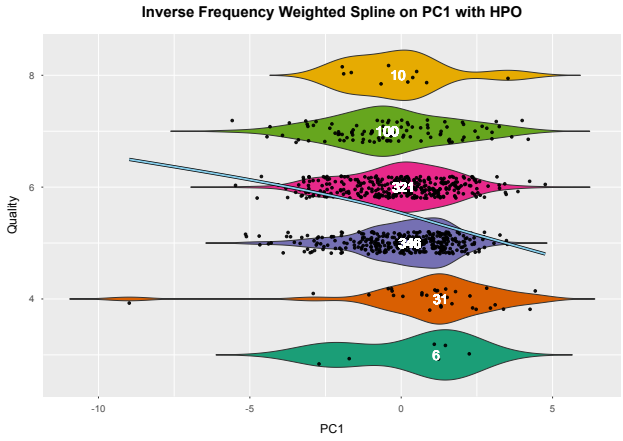
Figure 3.6: Fitted models (blue line) on volatile acidity and the training data

3.2.6 PC1 vs Quality

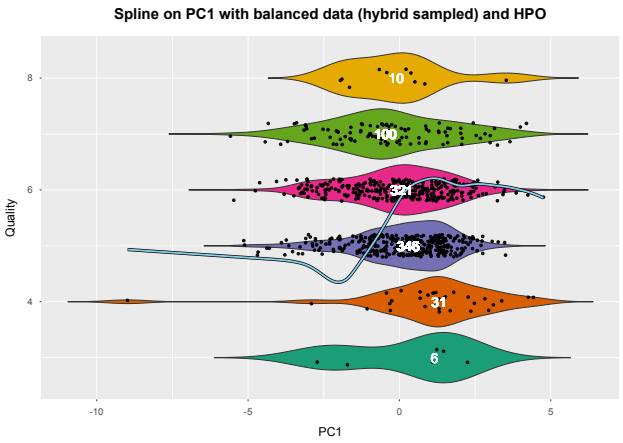
Models based on the first principal component look quite linear apart from the model trained on hybrid sampled data. Even though the weighted models have very low degrees of freedom, they still have some curvature, unlike the simple model.



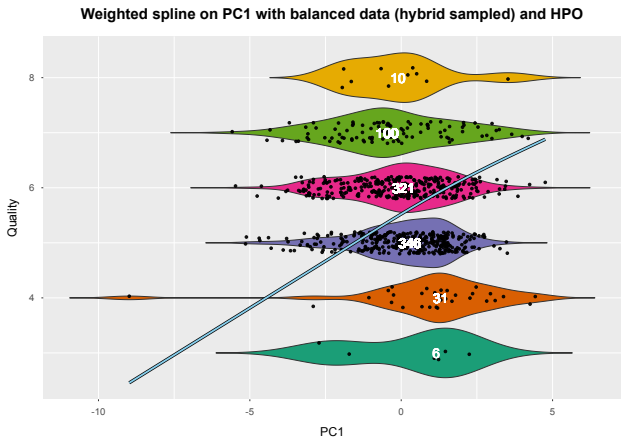
df = 2, val.- MSE = 0.67



df = 2, val.- MSE = 0.70



df = 8, val.- MSE = 0.99



df = 4, val.- MSE = 0.83

Figure 3.7: Fitted models (blue line) on the first PC and the training data

3.3 Results

After the models have been displayed on the different variables with their shape, degree of freedom and performance on the validation set, one model for each variable needs to be chosen. In order to do so, we investigate the models' performances on the validation dataset. The plots shown below in Figures 3.8, 3.9, 3.10, 3.11, 3.12 and 3.13 are grouped like for PPR in Figure 2.2: the models in the second row are created using hybrid sampling and the models in the right column are created using weights.

For each model the overall mean squared error as well as the mean MSE over all classes are displayed. The latter metric accounts more heavily for lower and higher quality wines than the former, which is why it was selected as the evaluation metric to choose the final model. The actual quality ratings are contrasted with the predicted ones, for each class there is one violin plot. To make all datapoints visible, some jittering is applied. Each subplot has a different scale, which complicates comparative assessment, but makes the plot easier to look at. The line of the identity function $f(x) = x$ serves as a visual reference for the locations of "perfect predictions", making identical scaling unnecessary. The exact performance metrics are found in the tables below the plots.

3.3.1 Alcohol vs Quality

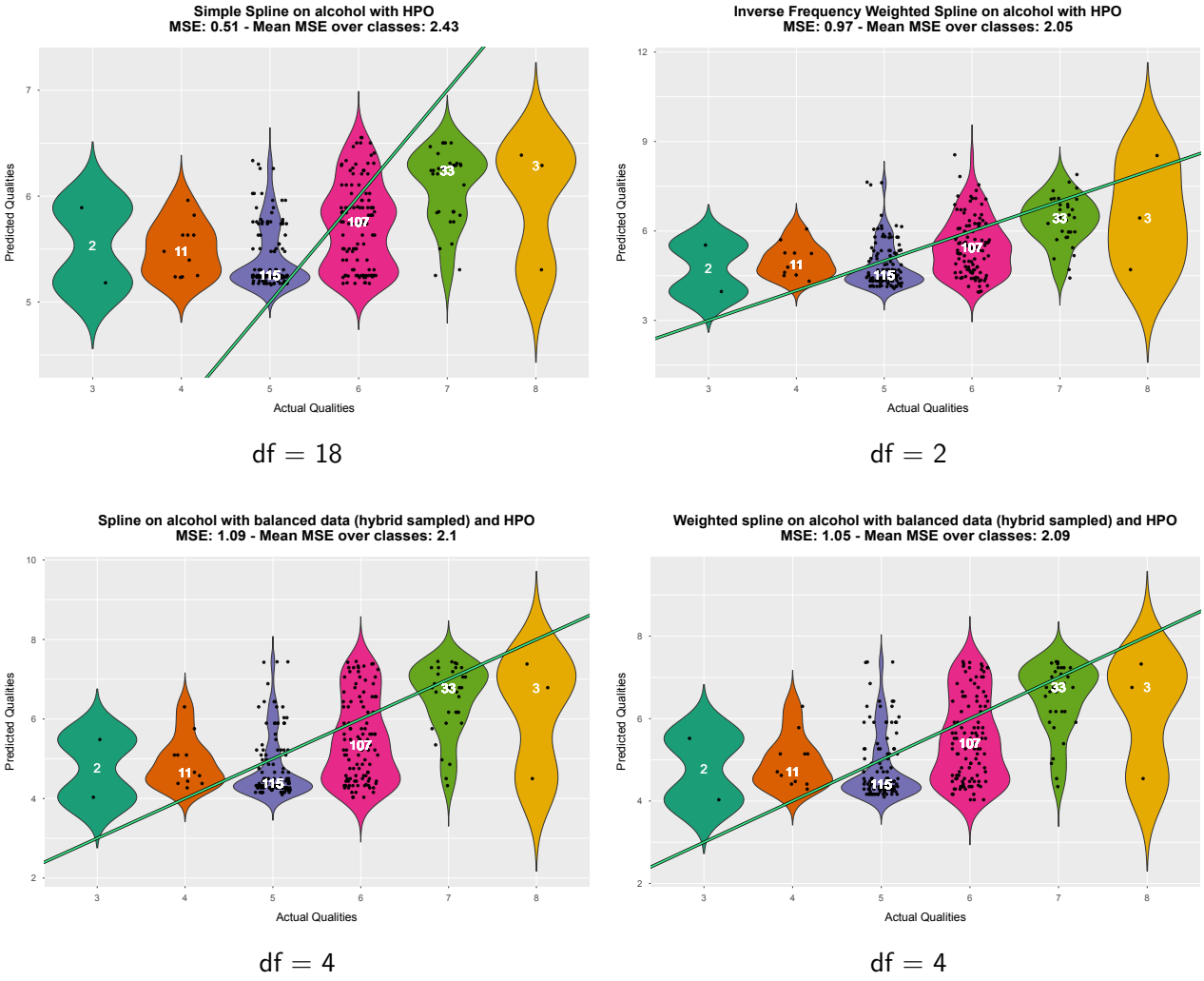


Figure 3.8: Model performances on the validation set for "alcohol-models"

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
Simple Spline	6.56	2.31	0.27	0.24	0.95	4.26	0.51	2.43
Inverse Frequency Weighted Spline	3.65	1.30	0.59	1.18	1.06	4.54	0.97	2.05
Hybrid Sampling	3.61	1.21	0.69	1.38	1.01	4.70	1.09	2.10
Hybrid Sampling and Weighting	3.71	1.26	0.66	1.31	0.97	4.65	1.05	2.09

Table 3.1: Summary of spline models on variable "Alcohol". Evaluated on validation set.

3.3.2 Density vs Quality

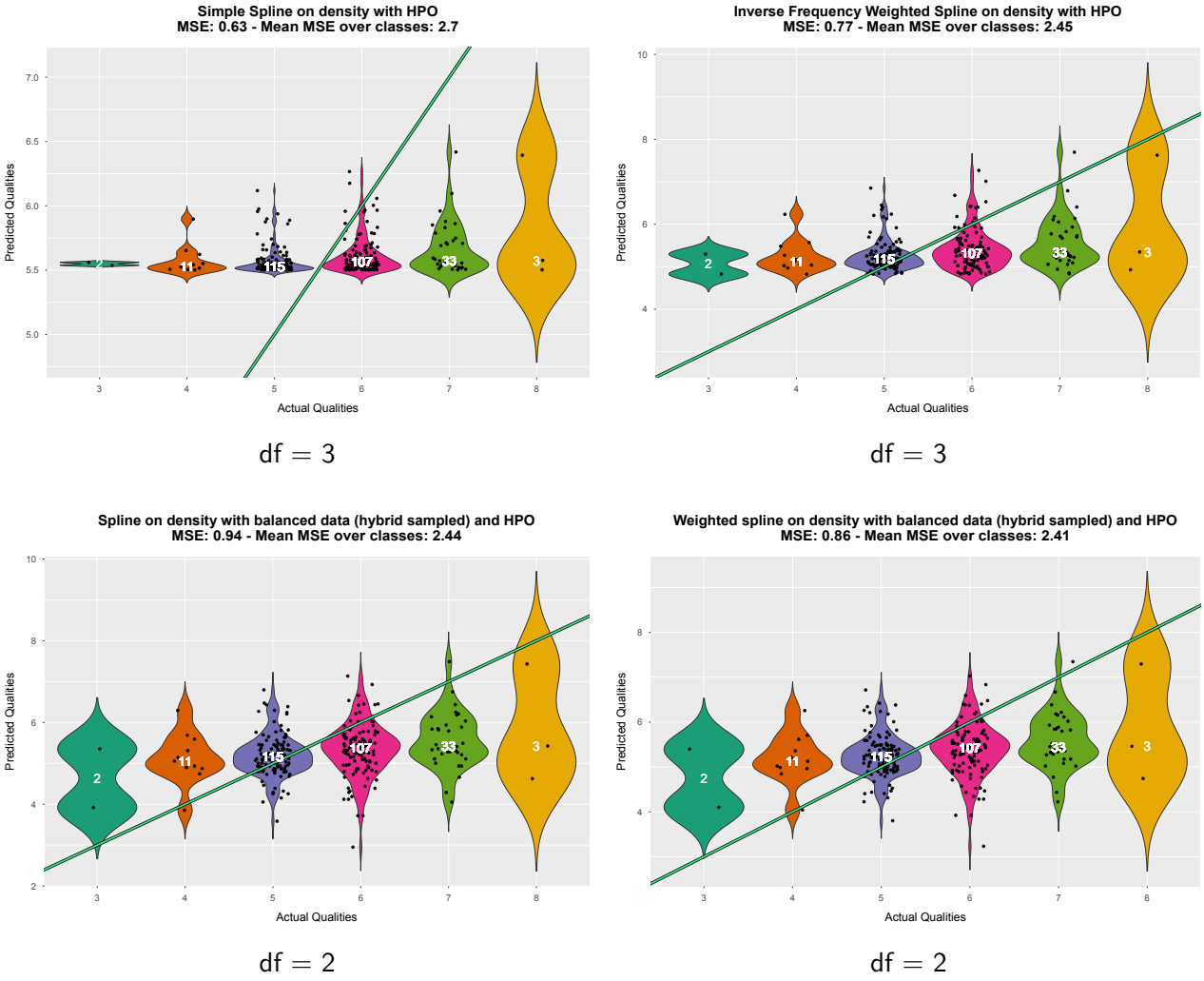


Figure 3.9: Model performances on the validation set for "density-models"

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
Simple Spline	6.50	2.49	0.34	0.17	1.80	4.90	0.63	2.70
Inverse Frequency Weighted Spline	4.31	1.71	0.21	0.58	2.38	5.54	0.77	2.45
Hybrid Sampling	3.19	1.62	0.30	0.88	2.54	6.11	0.94	2.44
Hybrid Sampling and Weighting	3.48	1.71	0.28	0.74	2.38	5.85	0.86	2.41

Table 3.2: Summary of spline models on variable "Density". Evaluated on validation set.

3.3.3 pH vs Quality

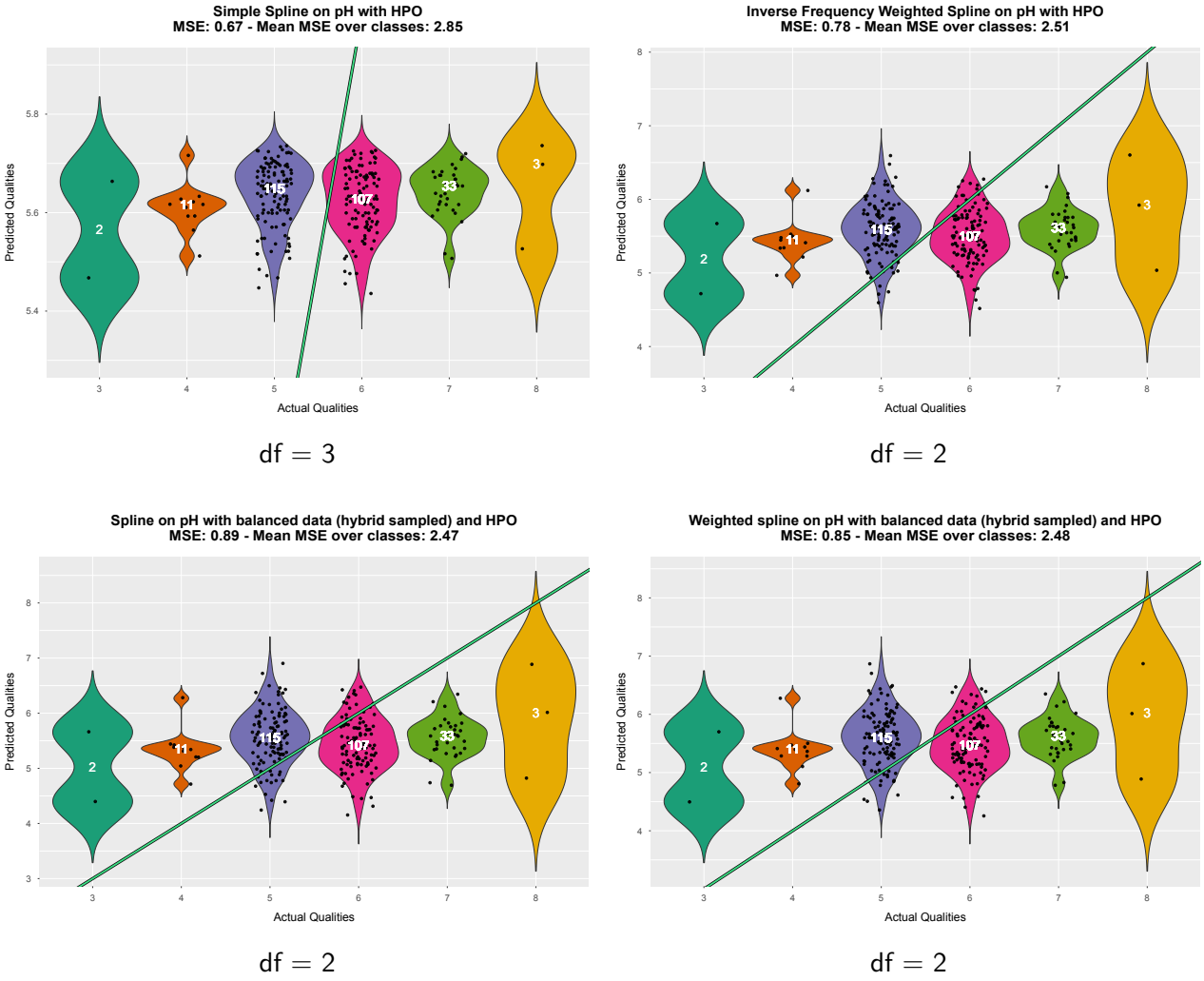


Figure 3.10: Model performances on the validation set for "pH-models"

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
Simple Spline	6.59	2.59	0.41	0.15	1.84	5.51	0.67	2.85
Inverse Frequency Weighting	5.04	2.13	0.49	0.37	2.03	5.01	0.78	2.51
Hybrid Sampling	4.51	1.92	0.55	0.55	2.19	5.12	0.89	2.47
Hybrid Sampling and Weighting	4.77	2.06	0.58	0.47	2.06	4.94	0.85	2.48

Table 3.3: Summary of spline models on variable "pH". Evaluated on validation set.

3.3.4 Residual.Sugar vs Quality

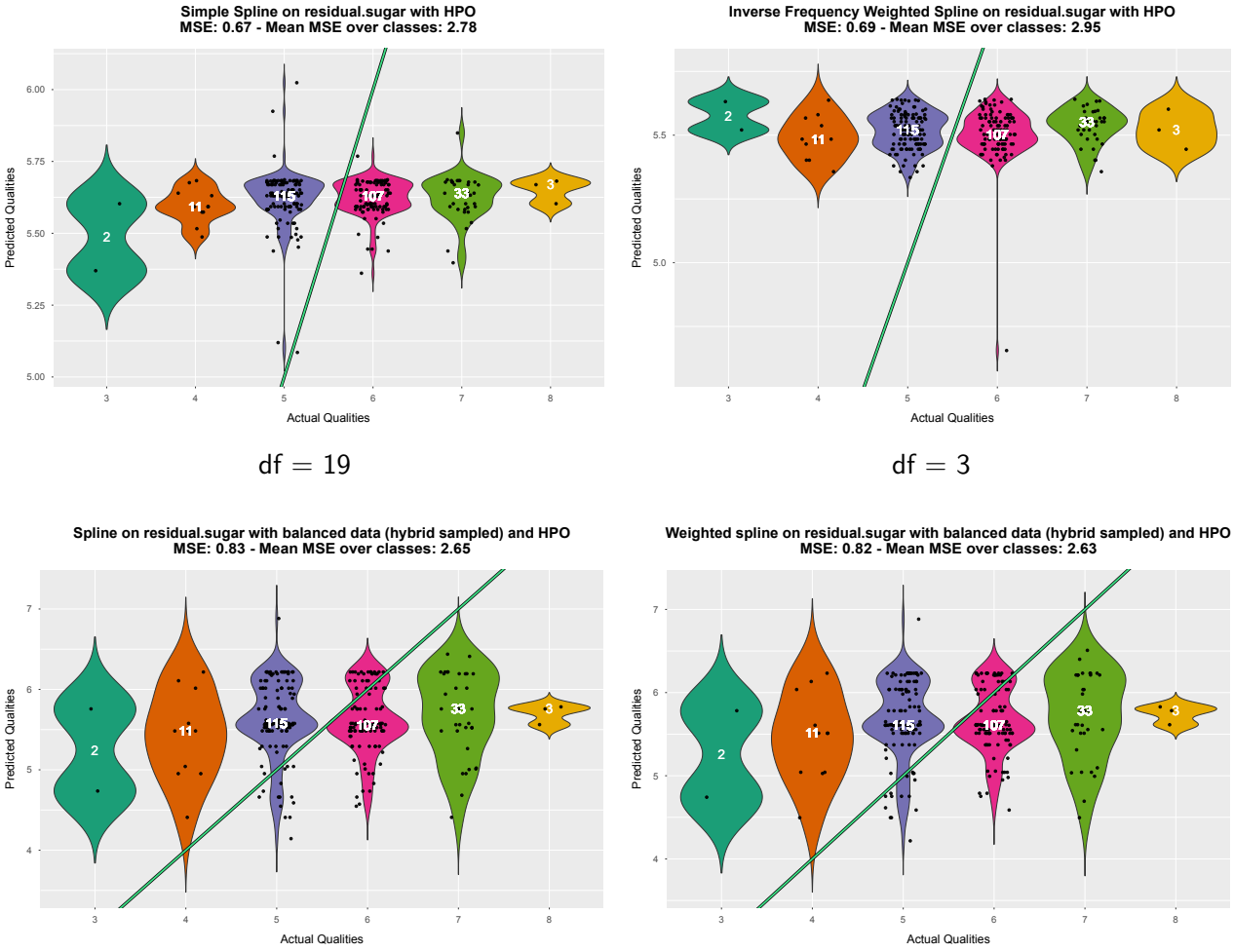


Figure 3.11: Model performances on the validation set for "residual-suger-models"

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
Simple Spline	6.20	2.55	0.40	0.15	1.90	5.52	0.67	2.78
Inverse Frequency Weighting	6.64	2.23	0.27	0.25	2.15	6.14	0.78	2.95
Hybrid Sampling	5.31	2.32	0.63	0.31	2.02	5.30	0.83	2.65
Hybrid Sampling and Weighting	5.39	2.42	0.66	0.27	1.93	5.11	0.82	2.63

Table 3.4: Summary of spline models on variable "Residual Sugar". Evaluated on validation set.

3.3.5 Volatile.Acidity vs Quality

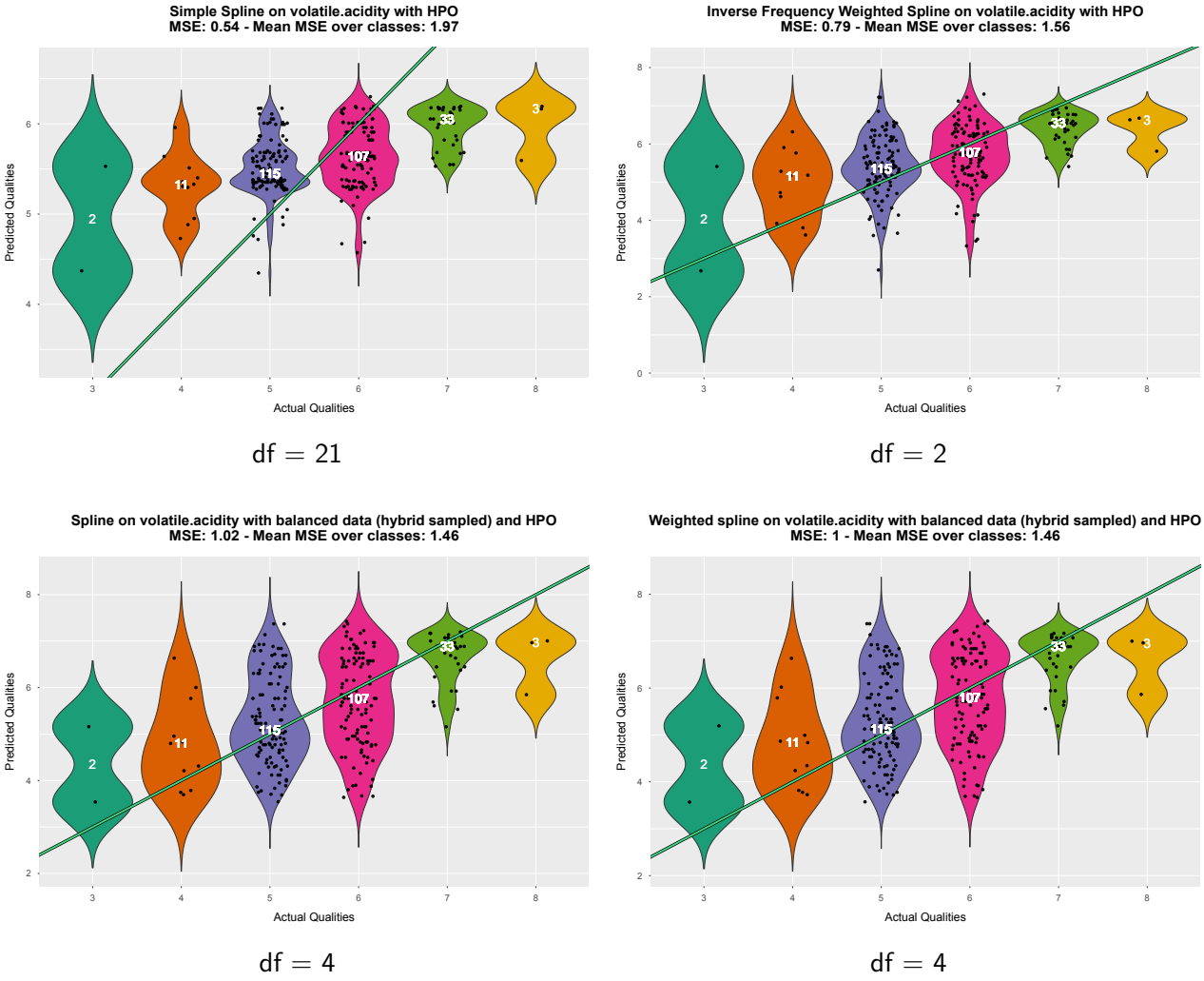


Figure 3.12: Model performances on the validation set for "volatile-acidity-models"

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
Simple Spline	4.14	1.82	0.37	0.25	1.11	4.14	0.54	1.97
Inverse Frequency Weighting	2.95	1.60	0.76	0.72	0.53	2.80	0.79	1.56
Hybrid Sampling	2.47	1.51	1.00	1.10	0.42	2.23	1.02	1.46
Hybrid Sampling and Weighting	2.57	1.54	0.99	1.06	0.41	2.21	1.00	1.46

Table 3.5: Summary of spline models on variable "Volatile Acidity". Evaluated on validation set.

3.3.6 PC1 vs Quality

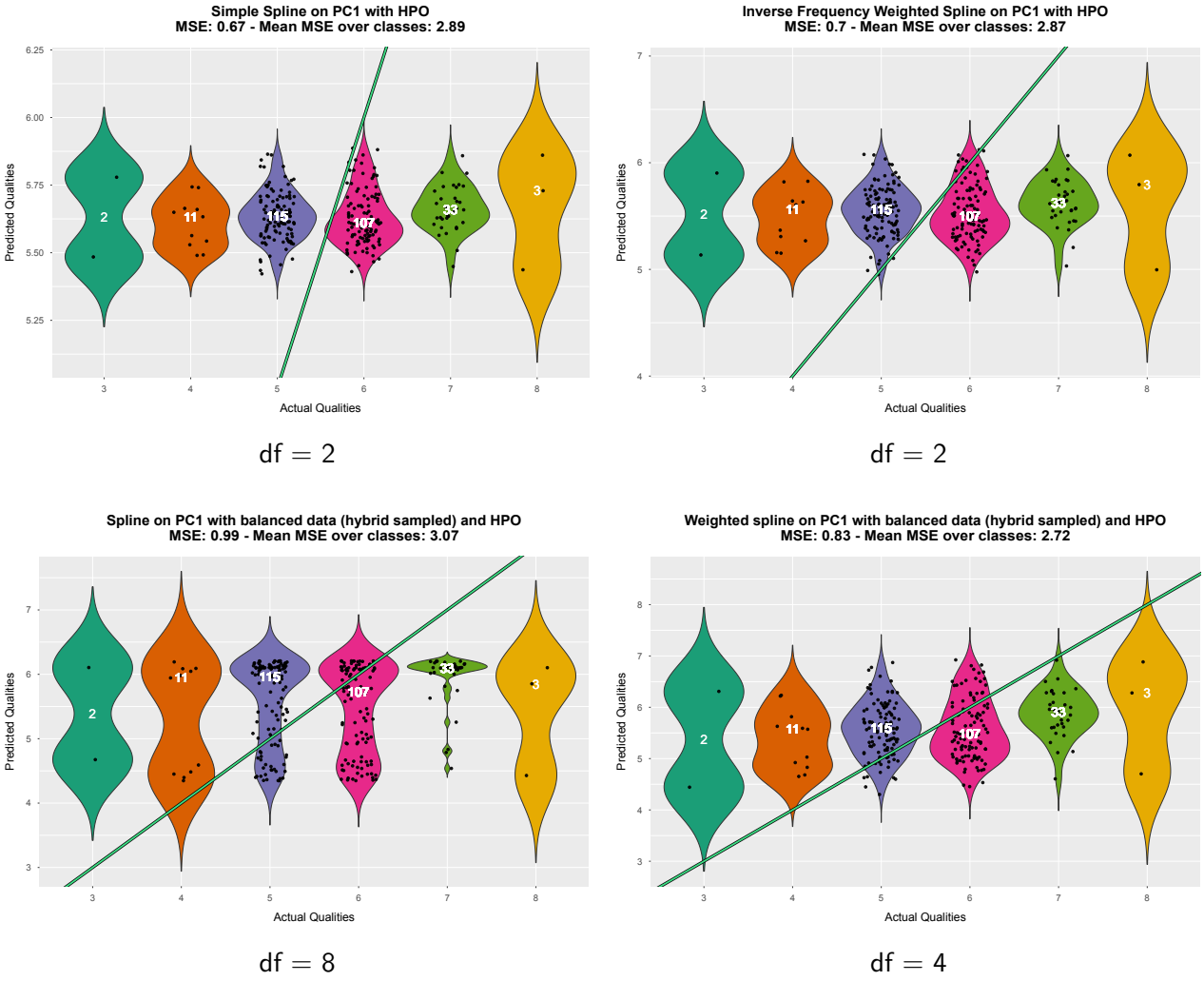


Figure 3.13: Model performances on the validation set for "PC1-models"

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
Simple Spline	6.95	2.60	0.41	0.15	1.79	5.43	0.67	2.89
Inverse Frequency Weighted Spline	6.49	2.26	0.36	0.29	1.93	5.87	0.70	2.87
Spline with Balanced Data	6.22	2.41	0.75	0.75	1.32	6.99	0.99	3.07
Weighted Spline with Balanced Data	6.52	2.21	0.58	0.53	1.46	5.03	0.83	2.72

Table 3.6: Summary of spline models on variable "PC1". Evaluated on validation set.

3.3.7 Final (Best) Spline Models

The table of MSE values 3.7 shows the models' performances. Based on the mean MSE over all classes, the variable volatile acidity with a preprocessing of over- and undersampling (denoted as "balanced data") is the best Spline Smoothing model. As indicated in Figure 3.12, the optimal number of degrees of freedom for this model configuration is 4.

Method	MSE							Mean MSE
	3	4	5	6	7	8	Total	over classes
PC1 (Weights, Balanced data)	6.52	2.21	0.58	0.53	1.46	5.03	0.83	2.72
Alcohol (Weights)	3.65	1.30	0.59	1.18	1.06	4.54	0.97	2.05
Density (Weights, Balanced data)	3.48	1.71	0.28	0.74	2.38	5.85	0.86	2.41
pH (Balanced data)	4.51	1.92	0.55	0.55	2.19	5.12	0.89	2.47
Residual sugar (Weights, Balanced data)	5.39	2.42	0.66	0.27	1.93	5.11	0.82	2.63
Volatile acidity (Balanced data)	2.47	1.51	1.00	1.10	0.42	2.23	1.02	1.46

Table 3.7: Summary of MSE results of the best spline models for each chosen variable. Evaluated on the validation set.

Chapter 4

Method Comparison

The best models for Projection Pursuit Regression and Spline Smoothing were determined.

For PPR, the best model is the one that was trained on hybrid sampled data, meaning over- and under-sampling was employed. The HPO resulted in the optimal number of terms being 3. The mean MSE over all classes is 0.93 for this model.

For Spline Smoothing, the best results were obtained with volatile.acidity as the predictor variable, and again, for this best model, hybrid sampled data was used. HPO showed that the optimal number of degrees of freedom is 4. The mean MSE over all classes is 1.46 for this model.

One could argue against the choice of the mean MSE over all classes as the evaluation criterion. Choosing models this way would lead to a lack of robustness due to the deficient number of observations in the classes of extreme values. However, the "normal" MSE is also lower for PPR than for Spline Smoothing, and this counts for the best models picked ($0.76 < 1.14$) as well as when comparing the lowest total MSE values obtained when considering all trained models of both methods ($0.42 < 0.51$). As PPR can use all eleven predictor variables, its superiority over Smoothing Splines, which only uses one predictor variable per model, is not too surprising. On the other hand, the simplicity due to the single predictor variable in Smoothing Splines leads to better interpretability. The spline plots were shown in Section 3.2. Such visualizations can deliver valuable insights beyond numbers (as famously shown by Anscombe's quartet). The spline plot for the best model is shown again below in Figure 4.1.

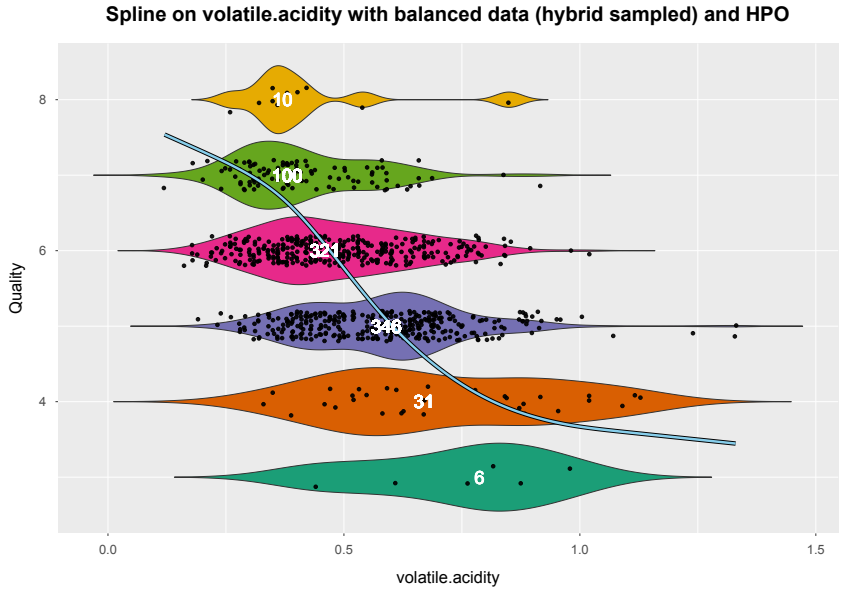


Figure 4.1: Spline plot of the best performing spline smoothing model (df = 4) with the training data

Again, the validation results for both best models were plotted below, in Figures 4.2 and 4.3, with overall MSE and class-wise mean MSE in the plot title. The violin plots juxtapose actual versus predicted quality ratings with jittering for better visibility of all data points. The identity function $f(x) = x$ again serves as a visual reference for the locations of "perfect predictions".

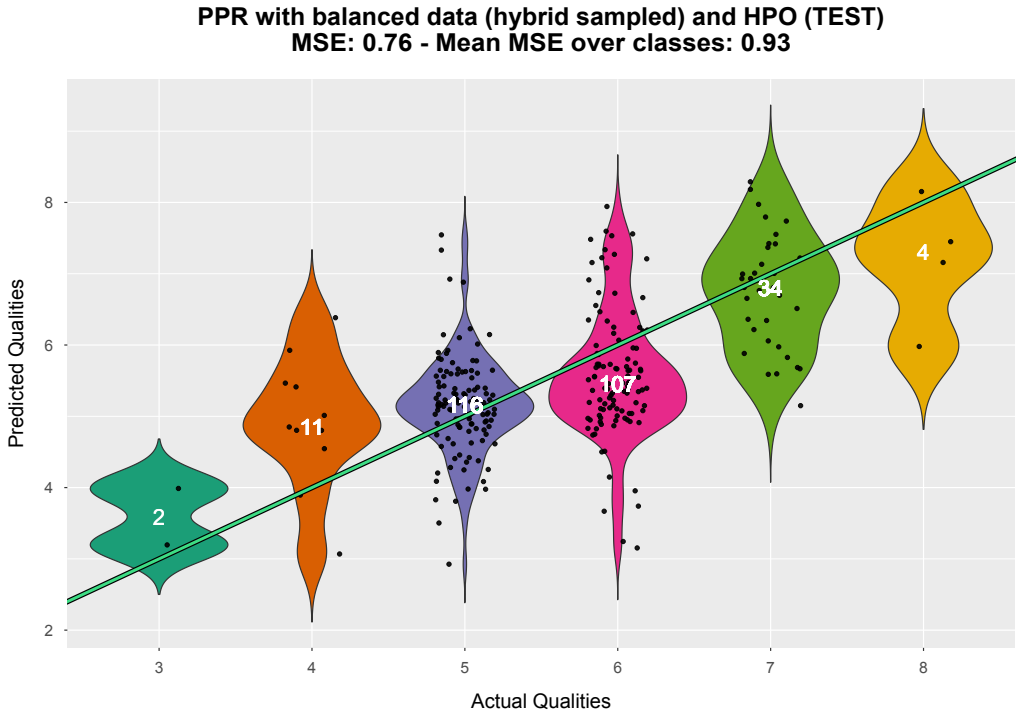


Figure 4.2: Violin Plot of Predictions on Test Set vs. Actual Values for best PPR model

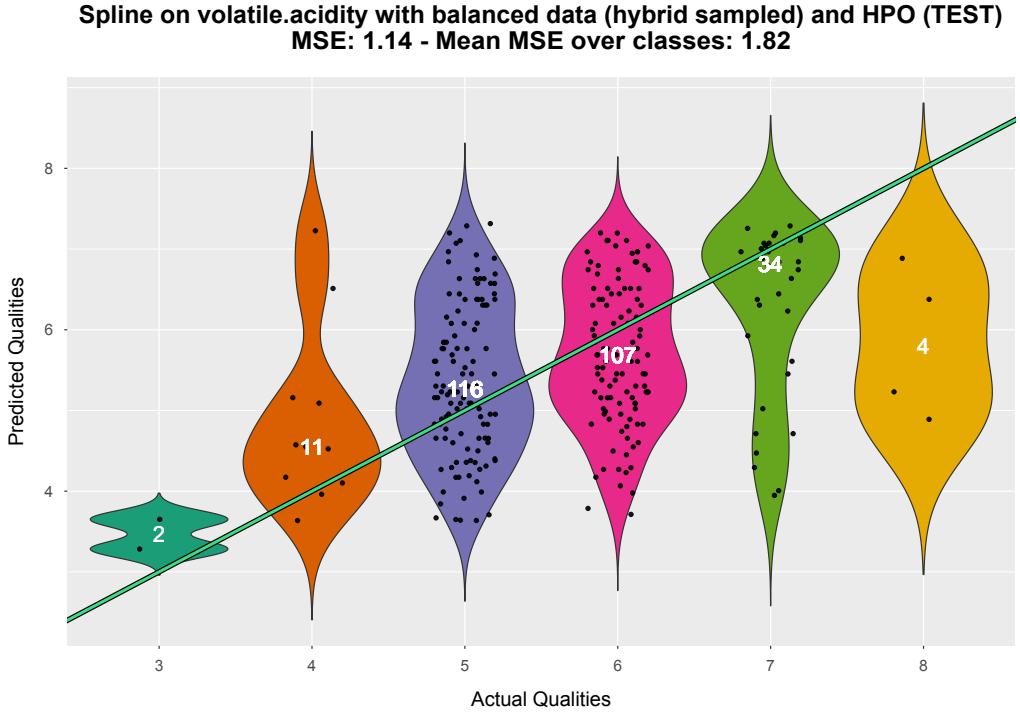


Figure 4.3: Violin Plot of Predictions on Test Set vs. Actual Values for best Spline model

Chapter 5

Conclusion

In this project, an extensive comparison of two advanced machine learning techniques, Projection Pursuit Regression (PPR) and Spline Smoothing, was performed. For this, a dataset of the physiochemical properties of red wines was used. The primary goal was to assess and compare their effectiveness in predicting wine quality (target variable, numeric discrete). Applying PPR and Spline Smoothing to the red wine dataset has delivered valuable insights.

PPR arose as the superior method between the two. Since it can make use of all predictor variables at once and identify complex, multidimensional directions in the predictor space, this is comprehensible. However, this flexibility can also lead to poorer model interpretability.

Spline Smoothing is less effective for multidimensional directions in the predictor space. Its biggest strength lies in capturing smooth, continuous patterns, making it an excellent choice for datasets where understanding the relationship between variables is at least as crucial as prediction performance.

The project also highlighted the importance of preprocessing in machine learning. Techniques like stratified sampling, oversampling, undersampling, and inverse frequency weighting were crucial in addressing the imbalance in the dataset. These strategies ensured that models were not biased towards the more frequently occurring classes and that the rarer quality ratings were adequately represented.

Overall, this examination provided valuable insights into the efficacy of PPR and Spline Smoothing in predicting wine quality but also showed the significance of thoughtful preprocessing and model evaluation strategies, especially in the presence of class imbalance. The results from this project could advise future research and applications in similar domains, particularly in scenarios where complex, non-linear relationships are present in the data.

References

[1] *Red Wine Quality*. 2024.

[2] *SMOTE: Synthetic Minority Oversampling TEchnique*. 2024. URL: https://www.rdocumentation.org/packages/smotefamily/versions/1.3.1/topics/SMOTE.

[3] N. Chawla et. al. "SMOTE: Synthetic Minority Over-sampling Technique". In: *arXiv:1106.1813* (2002).

[4] Downie. *Lecture slides week 8, Machine Learning 2, Winterterm 2023/2024*.

[5] Downie. *Lecture slides week 2, Machine Learning 2, Winterterm 2023/2024*.

Appendix: Use of AI Tools

Throughout our project, we made use of generative AI. Being aware of the issues that come with relying too heavily on generative AI, we utilized these technologies in a fashion that allowed for immediate validation and testing. In the context of programming, for example, we would instead ask for small but detailed operations, free from the context of our project, instead of whole blocks of code. We used ChatGPT 3.5/4 from OpenAI and "Github Copilot" for programming with R. Additionally, we had the image-generating AI "Dall-E" (also from OpenAI) create the cover image for our project.

These were the specific tasks in which we used AI tools:

- Image generation
- Code debugging
- Grammar correction
- Creation of report structures in LaTeX such as tables
- Recommendation of R packages
- Optimizing and finetuning visualizations

Throughout the use, we validated the information generated by the AI tools.