Business resource for wine industry

- Linking wine quality to wine physicochemical properties

Cenlin He[1], Liuli Chen[2], and Ankush Verma[2]
[1]Dept. of Atmospheric and Oceanic Sciences, UCLA, CA 90095
[2]Dept. of Computer Sciences, UCLA, CA 90095

1. Introduction

Wine is a tasty alcoholic beverage produced from fermented fruits such as grapes. Nowadays, wine has become increasingly popular. United States is the fourth largest country for wine production with nearly 80% of US people consuming wine. California accounts for 90% of total wine production in the US. To support the growth of wine industry, wine companies are making efforts to understand the key factors influencing wine quality and the way to improve wine taste. A number of wine quality evaluations, along with physicochemical laboratory tests for wine properties (e.g., alcohol, pH, and etc.), have been conducted by wine companies as a part of wine certification process. This provides a good opportunity for us to investigate the role of wine physicochemical properties in affecting wine quality (i.e., taste). In this study, we seek to build statistical prediction models to understand the relationship between wine quality and wine physicochemical properties. We also quantify the impacts of wine physicochemical properties on wine quality based on various statistical analyses. This study offers useful implications to wine companies on the improvement of wine taste.

2. Methods

2.1 Data collection

This study used two types of wine, including red and white *vinho verde* wine produced from Portugal. The data were collected from May 2004 to February 2007 using only protected designation of origin samples that were tested by an inter-professional organization named CVRVV, whose goal is to improve wine quality and marketing. The data were recorded by a computerized system, which automatically manages the process of wine testing from producer requests to laboratory and sensory analysis. Each entry represents one analytical/sensory test and the final dataset can be downloaded at the UCI database (https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/).

The dataset contains 1599 red wine samples and 4898 white wine samples. We conducted statistical analyses on the two types of wine separately, because the red and white wine tastes are very different. In this dataset, only the wine properties measured from the most common physicochemical tests were included in order to avoid discarding examples. The wine physicochemical properties used in this study include fixed acidity (g(tartaric acid)/L), volatile acidity (g(acetic acid)/L), citric acid (g/L), chlorides (g(NaCl)/L), free sulfur dioxide (SO_2) (mg/L), total SO_2 (mg/L), sulfate (g(K_2SO_4)/L), density (g/cm^3), pH (a metric to quantify acidity), residual sugar (g/L), and alcohol (vol. %). The wine quality was evaluated and graded by a minimum of three sensory assessors (using blind tastes) in a scale ranging from 0 (very bad) to 10 (excellent). The final sensory score was the median of these evaluations.

2.2 Statistical analysis

In the statistical analysis, the wine quality was treated as output variable (i.e., predicted variable),

while the wine physicochemical properties were treated as input variables. We firstly did an exploratory data analysis on the dataset to investigate the basic statistics and distribution (including kernel density estimation) of each variable and the pairwise correlation among variables. Following this, we examined and removed outliers in the dataset. Then, we applied the multiple linear regression analysis to both red and white wine to build optimal prediction models for wine quality, followed by error analyses and model evaluations (e.g., normal Q-Q plot and regression error characteristic (REC) curves). We also studied the influential points for the multiple linear regression models. Furthermore, we used the generalized additive model analysis with kernel regression and compared the results with the multiple linear regression.

3. Exploratory data analysis

3.1 Basic statistics of variables

Table 1 shows basic statistics of each variable, including mean, median, standard deviation, and the range of the data. We found that the alcohol amount (~10.5%) in red and white wines is similar, with a small variation among different samples. Both red and white wines are acid with pH less than 7, while white wine is slightly more acid than red wine on average. The white wine, containing more than a factor of two higher amount of residual sugar, is sweeter than the red wine. The amount of free and total SO_2 in white wine is a factor of 2-3 higher than that in red wine. However, the averaged amount of fixed and volatile acidity in red wine is higher than that in white wine. The different physicochemical properties of red and white wines probably reflect the different production processes for the two types of wines.

Figure 1 shows the distribution of red and white wine quality. Both the density distribution and boxplot indicate a similar wine quality for the two types of wines. The median wine quality is 6, with the minimum of 3 and maximum of 8 (red wine) or 9 (white wine).

3.2 Pairwise correlation analysis

Table 2 shows the pairwise correlation coefficients and the associated *p*-values among variables for red wine. We found that volatile acidity, citric acid, sulfate, and alcohol are correlated with red wine quality, with correlation coefficients (r) larger than 0.2 and *p*-values less than 0.01. This implies that these four properties play a dominant role in controlling red wine taste. The citric acid is negatively correlated with volatile acidity (i.e., acetic acid) but positively correlated with fixed acidity (i.e., tartaric acid). This probably reflects the characteristics of wine fermentation processes and the fruits and/or additives used in red wine production. The amount of these organic acids contained in red wine directly affects the acidity of wine, which is indicated by the high correlation between pH and these organic acids (see Table 2). The total SO_2 is highly correlated ($r = 0.67$) with free SO_2, because the free SO_2 dominates the total SO_2 amount in red wine. The sulfate has a relatively good correlation with the organic acids and chlorides, which suggests they have similar sources such as the fruits and additives used in wine fermentation. Interestingly, the alcohol amount is highly negatively correlated with red wine density, primarily because alcohol is the major component having a smaller density than water. The alcohol balances the high density caused by residual sugar and organic acids, which also explains that the wine density is very close to pure water density (1 g/L).

Similar to red wine, white wine shows a relatively high pairwise correlation between citric acid and fixed acidity, total SO_2 and free SO_2, pH and organic acids, alcohol and density (see Table 3). However, unlike red wine, the white wine quality is more correlated with alcohol, wine density, volatile acidity, and chlorides, which could contribute to the

different tastes of wine and red wines. The density of white wine is dominated by residual sugar and total SO_2 ($r > 0.5$) but less controlled by fixed acidity and citric acid ($r < 0.3$) (see Table 3), which is different from red wine. This is partly due to the higher sweetness of white wine. It is interesting that the residual sugar (organic species) is correlated with SO_2 (inorganic species), which is likely because of the similar sources of these two chemicals during white wine production. Thus, differences in wine production processes result in different physiochemical properties of white and red wines and the associated relationship between wine quality and its properties, which contributes to the different tastes of white and red wines.

3.3 Kernel density estimation

Based on the pairwise correlation analysis, four important physicochemical properties that show the highest correlation with wine quality were selected for kernel density estimations for both red and white wine cases (see Figure 2). For the red wine, we have selected alcohol, citric acid, volatile acidity and sulfate. The kernel density curves for alcohol, volatile acidity and sulfate show a bell shape similar to the Gaussian distribution but with a positive skewness (i.e., right-tailed), which are centered at 9.5%, 0.5 g/L, and 0.6 g/L, respectively. However, the citric acid shows a relatively uniform distribution over 0-0.5 g/L, followed by a sharp decrease. For white wine, we have selected volatile acidity, alcohol, wine density and chlorides. All the four variables show a right-skewed Gaussian distribution, which are centered at 9.5%, 0.995 g/cm^3, 0.25 g/L, and 0.5 g/L, respectively (see Figure 2). But unlike red wine, the alcohol in white wine has a much larger variation in its distribution, with a stronger right tail (i.e., more samples in values > 11%).

3.4 Outlier removal

Because outliers could significantly affect the statistical analysis, we detected and removed outliers before conducting multiple linear regressions and generalized additive model regressions. Figure 3 shows the outliers based on eye detection for both red and white wine datasets. The outliers include wine samples with extremely high contents of organic acids, SO_2, and residual sugar. Preliminary multiple linear regression results show slight but noticeable improvements after outlier removal. The R^2 increases from 0.3567 to 0.3644 for red wine, while R^2 increases from 0.2806 to 0.2877 for white wine. The improvement is small because the number of outliers is much smaller compared with the total datasets (1599 for red wine and 4989 for white wine). The following regression analyses are based on datasets with outlier removal.

4. Multiple linear regression

4.1 Model results

We conducted multiple linear regressions with both backward and forward stepwise analysis to find the best-fit model. The regression in both directions shows the same set of variables at a 5% significance level (i.e., $p < 0.05$) for red and white wine. Figure 4 shows a summary of regression results with backward stepwise analysis. The qualities of both red and white wines are affected by alcohol, volatile acidity, free SO_2, pH, and sulfate at a 1% significance level (i.e., $p < 0.01$). Besides these five variables, red wine quality is also significantly influenced by chloride and total SO_2, while white wine quality is largely influenced by fixed acidity, density, and residual sugar.

Using only variables at the 1% significance level, the best-fit multiple linear regression model (see Figure 4) for red wine is ***Quality = 4.414 - 0.984 Volatile acidity - 1.758 Chloride + 0.006 Free SO_2 - 0.004 Total SO_2 - 0.504 pH + 0.936 Sulfates + 0.292 Alcohol***, and for white wine is ***Quality = 225.5 + 0.125 Fixed acidity - 1.856 Volatile acidity + 0.105 Residual sugar + 0.004 Free SO_2 - 226.5 Density +***

0.921 pH +0.731 Sulfates + 0.102 Alcohol. The negative regression coefficient of volatile acidity for both red and white wine indicates that the increase of volatile acidity degrades the wine taste, while the positive regression coefficients of alcohol, sulfates, and free SO_2 indicate that increasing the amount of these three chemicals can improve both red and white wine qualities. It is interesting that a higher pH decreases red wine quality but increases white wine quality, which suggests that the red wine with a stronger acidity (i.e., lower pH) tastes better and this is the opposite for white wine. The regression model also shows that an increase of chloride and total SO_2 results in a decrease of red wine quality, while an increase of fixed acidity and residual sugar improves white wine taste.

We further calculated the standardized regression coefficients (*B*) (i.e., partial regression coefficient divided by standard deviation of variable) for each variable in the best-fit model, which represent the relative importance of regressors, with larger *B* corresponding to higher importance. The first four important factors are chlorides (*B* = -38.9), sulfates (*B* = 5.6), volatile acidity (*B* = -5.5), and pH (*B* = -3.3) for red wine, and density (*B* = -78010), volatile acidity (*B* = -18.5), sulfate (*B* = 6.4), and pH (*B* = 6.1) for white wine. For example, an increase of sulfate by one standard deviation can increase the red wine quality by 5.6 and white wine quality by 6.4. It is interesting that alcohol has the highest pairwise correlation with wine quality, but it is less important than other variables in terms of influencing wine quality in the multiple linear regression model.

4.2 Model evaluations

Figure 5 shows the evaluations of the best-fit multiple linear regression models for red and white wines, including the adjusted R^2, root-mean-square-error (RMSE), prediction-versus-observation plot, normal Q-Q plot, and REC curve. The adjusted R^2 of the best-fit multiple linear regression model indicates that 37% of wine quality variation is accounted for by the regression model for red wine and 29% is accounted for by the regression model for white wine. This implies that there should be some missing factors not accounted for in the regression model. The RMSE is 0.64 for red wine model and 0.75 for white wine, reflecting a slightly better performance of the red wine model than white wine model. The prediction-versus-observation plot shows that the regression model underestimates the wine quality in high values and overestimates the wine quality in low values for both red and white wines. This may be because we have more data points in the middle range of wine quality so that the regression model tends to predict the medium wine quality well. The normal Q-Q plot shows that both the observations and model predictions for red and white wines follow the normal distribution, but model predictions exhibit a similar underestimate and overestimate pattern as shown by the prediction-versus-observation plot. The REC curves illustrate the performance of the best-fit regression model compared with the theoretical perfect model. The closer the regression model curve is to the perfect model curve, the better the regression model performs. Our regression model has an encouraging performance for both red and white wines.

4.3 Influential point analysis

In order to understand the influence of each data point on the regression coefficient of alcohol in the multiple linear regression model, we repeated the multiple linear regression on the dataset with the removal of one data point each time. Figure 6 shows the difference between the new and original regression coefficient of alcohol as a function of the data point removed for both red and white wines. The dataset has been sorted by an increasing order of alcohol amount. We found that wine samples with

higher alcohol amount exhibit larger influences on the regression coefficient of alcohol for red wine, but there is no such pattern for white wine.

5. Generalized additive model with kernel regression

5.1 Model results

Figures 7 and 8 show the results of generalized additive model with Nadaraya-Watson kernel regression for red and white wines, respectively. We selected six key variables for the generalized additive model fitting, which contribute to the largest reduction of model RMSE. For red wine, the six regressors include alcohol, volatile acid, sulfate, chloride, residual sugar, and pH, while the regressors for white wine include alcohol, volatile acid, free SO_2, chloride, residual sugar, and fixed acidity. We found that the kernel regression curves for alcohol and volatile acidity show good linear trend for both red and white wines, with increasing alcohol leading to better wine taste and increasing volatile acidity leading to degraded wine quality.

Figure 7 shows that the red wine quality increases with the increasing sulfate amount within the range of 0.2-0.8 g/L sulfate, but decreases as the sulfate amount further increases from 0.8 to 1.3 g/L. This provides helpful implications on how to control sulfate amount to improve red wine taste during wine production. The kernel regression curve for pH also shows a slightly negative linear trend, which indicates that a higher acidity improves red wine quality in the pH range of 3.0-4.0. This is consistent with the results from the multiple linear regression. The kernel regression curves of the rest of two variables are rather non-linear.

Figure 8 shows that the white wine quality increases with the increasing free SO_2 in a range of 0-25 mg/L SO_2, but remains the same as free SO_2 increases from 25 to 80 mg/L. Thus, adding free SO_2 into white wine may not be an efficient way to improve its quality when the wine already contains 25 to 80 mg/L free SO_2. The regression curve for fixed acidity exhibits a weak negative linear trend, suggesting that less fixed acidity could slightly improve white wine taste. The regression curves for chloride and residual sugar do not show significant overall non-zero-slope linear trends.

5.2 Model evaluations

Figures 9 and 10 show the evaluations of the generalized additive model for red and white wines, respecitively. The adjusted R^2 indicates that 45% of red wine quality variation is explained by the generalized additive model and 29% is explained for white wine. The RMSE is 0.60 for red wine model and 0.70 for white wine model. Similar patterns of the prediction-versus-observation plot are also observed for the generalized additive models compared with the multiple linear regression models, where the model underestimates the high wine quality and overestimates the low wine quality. This is consistent with the results from the normal Q-Q plot, which additionally shows that the model predictions are close to the normal distribution. The REC curves further illustrate that the performance of the generalized additive model is promising.

5.3 Comparison with multiple linear regression

Compared with the multiple linear regression models, the generalized additive models produce higher R^2 and lower RMSE, showing a slightly better model performance. Figure 11 shows that the REC curve of generalized additive model is slightly closer to that of the perfect model. The key variables in generalized additive model are mostly consistent with those in the best-fit multiple linear regression models for both red and white wines. This implies that these selected variables really play a critical role in regulating wine quality. However, from a practical perspective, the multiple linear regression model is easier to apply to the real wine production process,

because of its simplicity.

6. Conclusions and implications

In this study, we applied various statistical analyses (e.g., correlation analysis, kernel density estimation, outlier removal, and etc.) to the red and white wine datasets to understand the effects of wine physicochemical properties on wine quality. We built multiple linear regression models and generalized additive models for both red and white wines to link wine taste to several key physicochemical properties. Our analyses showed that removing the outliers slightly improved the performance of multiple linear regression models. The regression model indicated that the amount of alcohol, volatile acidity, sulfate, free SO_2, and pH exerted significant influences on red and white wine quality, where alcohol, sulfate, and free SO_2 showed positive correlations and volatile acidity showed negative correlations. Relatively high pH increased the taste of white wine, but decreased the taste of red wine. The sweetness was more important for white wine, which improved white wine quality. Furthermore, the generalized additive model showed consistent results with multiple linear regression model in terms of the most important variables. Both the multiple linear regression model and generalized additive model explained 30-45% of the variations of red and white wine quality, suggesting that there should be some missing variables not accounted for in the models. Overall, our results from the two types of regression model analyses could be very helpful for the wine industry to produce more tasty wine, since the wine physicochemical properties used in our analyses can be controlled during wine production.

7. Limitations and future work

There are some limitations in this study, including missing variables not accounted for in the regression models such as temperature and other chemical components. The outlier removal method is also arbitrary, which calls for a more quantitative mathematical approach. The prediction models are also limited to two types of regression models in our analyses, which may not be the best way to capture the whole features of the datasets.

In the future research, we will try to use other analysis methods such as principle component analysis (PCA), support vector machine (SVM) and cluster analysis. We will also include more wine physicochemical properties in the analysis if the data is available. We also would like to further investigate some key factors (e.g., sugar, pH, alcohol, and etc.) in more details.

Tables

Table 1. Variable statistics

Variables (unit)	**Red wine (1599 samples)**					**White wine (4898 samples)**				
	Mean	**Median**	**Std***	**Min***	**Max***	**Mean**	**Median**	**Std**	**Min**	**Max**
Quality	5.64	6.00	0.81	3.00	8.00	5.88	6.00	0.89	3.00	9.00
Fixed acidity (g(tartaric acid)/L)	8.32	7.90	1.74	4.60	15.90	6.85	6.80	0.84	3.80	14.20
Volatile acidity (g(acetic acid)/L)	0.53	0.52	0.18	0.12	1.58	0.28	0.26	0.10	0.08	1.10
Citric acid (g/L)	0.27	0.26	0.19	0.00	1.00	0.33	0.32	0.12	0.00	1.66
Chlorides (g(NaCl)/L)	0.09	0.08	0.05	0.01	0.61	0.05	0.04	0.02	0.01	0.35
Free sulfur dioxide (mg/L)	15.87	14.0	10.46	1.00	72.00	35.31	34.00	17.01	2.00	289.00
Total sulfur dioxide (mg/L)	46.47	38.00	32.90	6.00	289.00	138.36	134.00	42.50	9.00	440.00
Sulfate (g(K_2SO_4)/L)	0.66	0.62	0.17	0.33	2.00	0.49	0.47	0.11	0.22	1.08
Density (g/cm^3)	1.00	1.00	0.00	0.99	1.00	0.99	0.99	0.00	0.99	1.04
pH	3.31	3.31	0.15	2.74	4.01	3.19	3.18	0.15	2.72	3.82
Residual sugar (g/L)	2.54	2.20	1.41	0.90	15.50	6.39	5.20	5.07	0.60	65.80
Alcohol (vol. %)	10.42	10.20	1.07	8.40	14.90	10.51	10.40	1.23	8.00	14.20

*standard deviation (std), minimum (min), maximum (max).

Table 2. Correlation and *p*-value matrix for red wine

Correlation / ***p*-value**	Quality	Fixed acidity	Volatile acidity	Citric acid	Chloride	Free sulfur dioxide	Total sulfur dioxide	Sulfate	Density	pH	Residual sugar	Alcohol
Quality		0.12	-0.39	0.23	-0.13	-0.05	-0.19	0.25	-0.18	-0.06	0.01	0.48
Fixed acidity	0.00		-0.26	0.67	0.09	-0.15	-0.11	0.18	0.67	-0.68	0.12	-0.06
Volatile acidity	0.00	0.00		-0.55	0.06	-0.01	0.08	-0.26	0.02	0.24	0.002	0.20
Citric acid	0.00	0.00	0.00		0.20	-0.06	0.04	0.31	0.37	-0.54	0.14	0.11
Chlorides	0.00	0.00	0.01	0.00		0.01	0.05	0.37	0.20	-0.27	0.06	-0.22
Free sulfur dioxide	0.04	0.00	0.67	0.01	0.82		0.67	0.05	-0.02	0.07	0.19	-0.07
Total sulfur dioxide	0.00	0.00	0.00	0.16	0.06	0.00		0.04	0.07	-0.07	0.20	-0.21
Sulfate	0.00	0.00	0.00	0.00	0.00	0.04	0.09		0.15	-0.20	0.01	0.09
Density	0.00	0.00	0.38	0.00	0.00	0.38	0.00	0.00		-0.34	0.36	-0.50
pH	0.02	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00		-0.09	0.21
Residual sugar	0.58	0.00	0.94	0.00	0.03	0.00	0.00	0.83	0.00	0.00		0.04
Alcohol	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.09	

Table 3. Correlation and *p*-value matrix for white wine

correlation / *p*-value	Quality	Fixed acidity	Volatile acidity	Citric acid	Chloride	Free sulfur dioxide	Total sulfur dioxide	Sulfate	Density	pH	Residual sugar	Alcohol
Quality		-0.11	-0.19	-0.01	-0.21	0.01	-0.17	0.05	-0.31	0.10	-0.10	0.44
Fixed acidity	0.00		-0.02	0.29	0.02	-0.05	0.09	-0.02	0.27	-0.43	0.09	-0.12
Volatile acidity	0.00	0.11		-0.15	0.07	-0.10	0.09	-0.04	0.03	-0.03	0.06	0.07
Citric acid	0.52	0.00	0.00		0.11	0.09	0.12	0.06	0.15	-0.16	0.09	-0.08
Chlorides	0.00	0.11	0.00	0.00		0.10	0.20	0.02	0.26	-0.09	0.09	-0.36
Free sulfur dioxide	0.57	0.00	0.00	0.00	0.00		0.62	0.06	0.29	0.00	0.30	-0.25
Total sulfur dioxide	0.00	0.00	0.00	0.00	0.00	0.00		0.13	0.53	0.00	0.40	-0.45
Sulfate	0.00	0.23	0.01	0.00	0.24	0.00	0.00		0.07	0.16	-0.03	-0.02
Density	0.00	0.00	0.06	0.00	0.00	0.00	0.00	0.00		-0.09	0.84	-0.78
pH	0.00	0.00	0.03	0.00	0.00	0.97	0.87	0.00	0.00		-0.19	0.12
Residual sugar	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.06	0.00	0.00		-0.45
Alcohol	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.22	0.00	0.00	0.00	

Figures

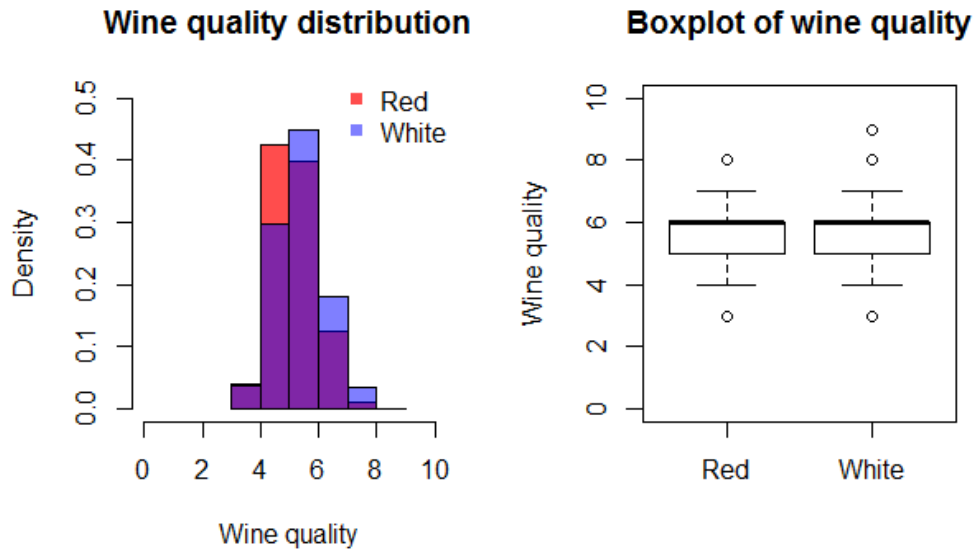

Figure 1. Distributions of red and white wine quality.

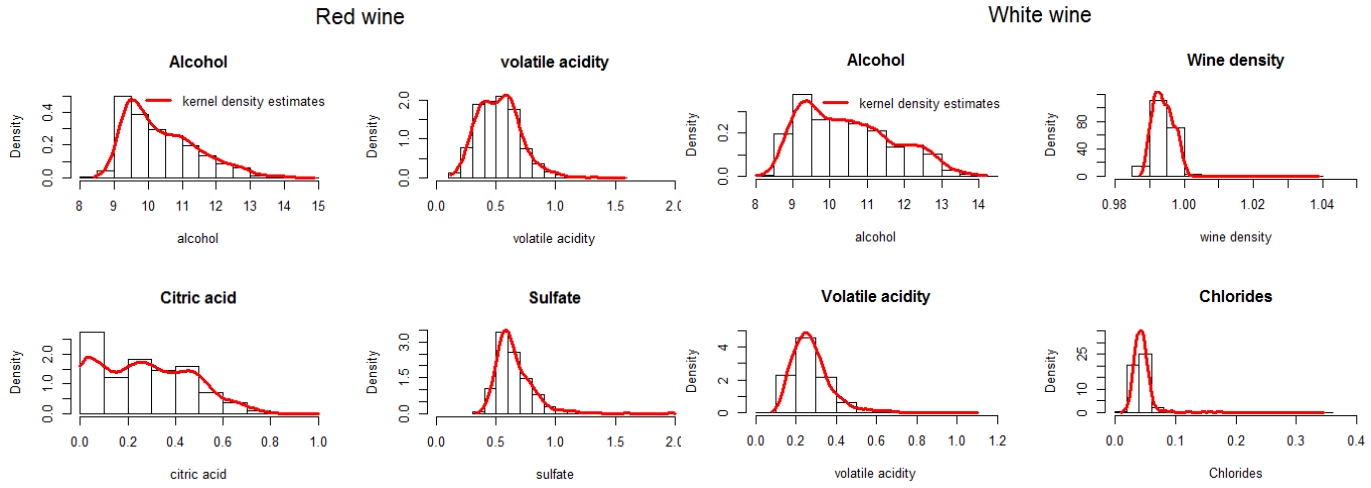

Figure 2. Kernel density estimations for 4 key physicochemical properties of red and white wines.

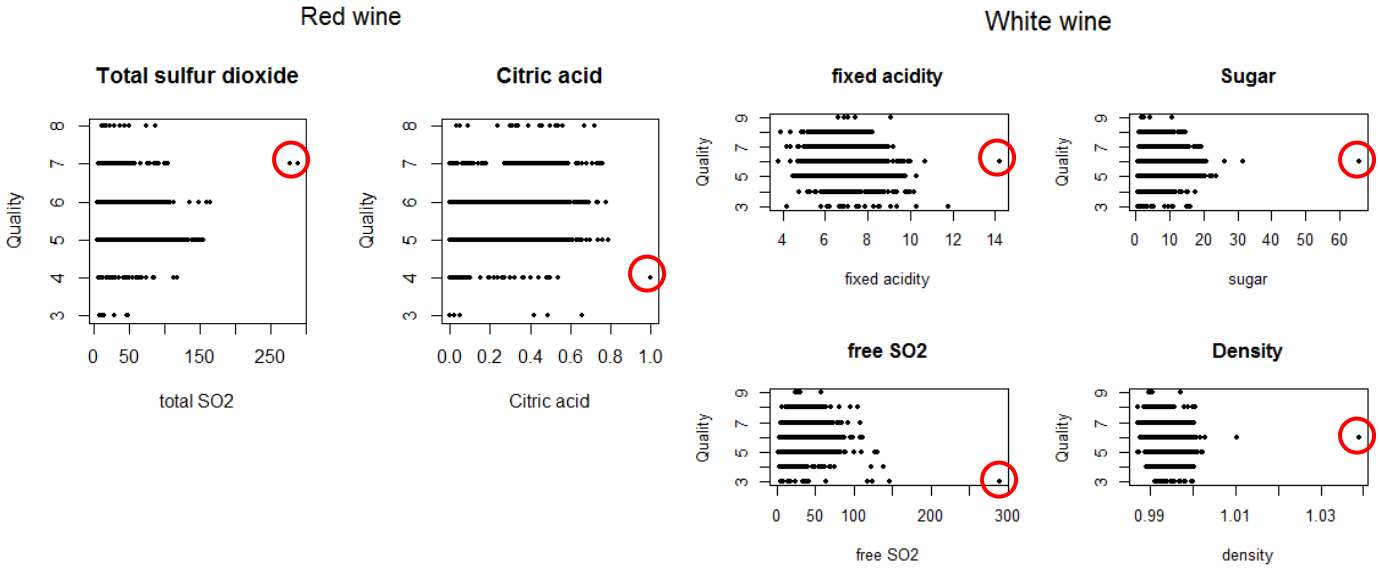

Figure 3. Outliers (marked by red circles) for red and white wine datasets.

Red wine (backward)

```
Call:
lm(formula = ry ~ rva + rch + rfs + rts + rph + rso + ral)

Residuals:
     Min       1Q   Median       3Q      Max
-2.73071 -0.36790 -0.04832  0.45481  2.04996

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.4142394  0.3998620  11.039  < 2e-16 ***
rva         -0.9836996  0.1002278  -9.815  < 2e-16 ***
rch         -1.7579506  0.4033101  -4.359 1.39e-05 ***
rfs          0.0062382  0.0021320   2.926  0.00348 **
rts         -0.0040696  0.0007144  -5.697 1.45e-08 ***
rph         -0.5044674  0.1171733  -4.305 1.77e-05 ***
rso          0.9356792  0.1098654   8.517  < 2e-16 ***
ral          0.2917024  0.0169564  17.203  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6426 on 1587 degrees of freedom
Multiple R-squared:  0.3672, Adjusted R-squared:  0.3644
F-statistic: 131.5 on 7 and 1587 DF,  p-value: < 2.2e-16
```

White wine (backward)

```
Call:
lm(formula = wy ~ wfa + wva + wsu + wfs + wde + wph + wso + wal)

Residuals:
    Min      1Q  Median      3Q     Max
-3.6017 -0.4981 -0.0418  0.4651  3.1120

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.255e+02  2.214e+01  10.185  < 2e-16 ***
wfa          1.254e-01  2.289e-02   5.476 4.56e-08 ***
wva         -1.856e+00  1.090e-01 -17.020  < 2e-16 ***
wsu          1.051e-01  8.370e-03  12.551  < 2e-16 ***
wfs          4.412e-03  6.923e-04   6.373 2.02e-10 ***
wde         -2.265e+02  2.242e+01 -10.103  < 2e-16 ***
wph          9.211e-01  1.106e-01   8.326  < 2e-16 ***
wso          7.307e-01  1.011e-01   7.230 5.57e-13 ***
wal          1.024e-01  2.914e-02   3.515 0.000444 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.747 on 4884 degrees of freedom
Multiple R-squared:  0.2889, Adjusted R-squared:  0.2877
F-statistic:   248 on 8 and 4884 DF,  p-value: < 2.2e-16
```

Figure 4. Multiple linear regression results determined by backward stepwise analysis for red and white wines.

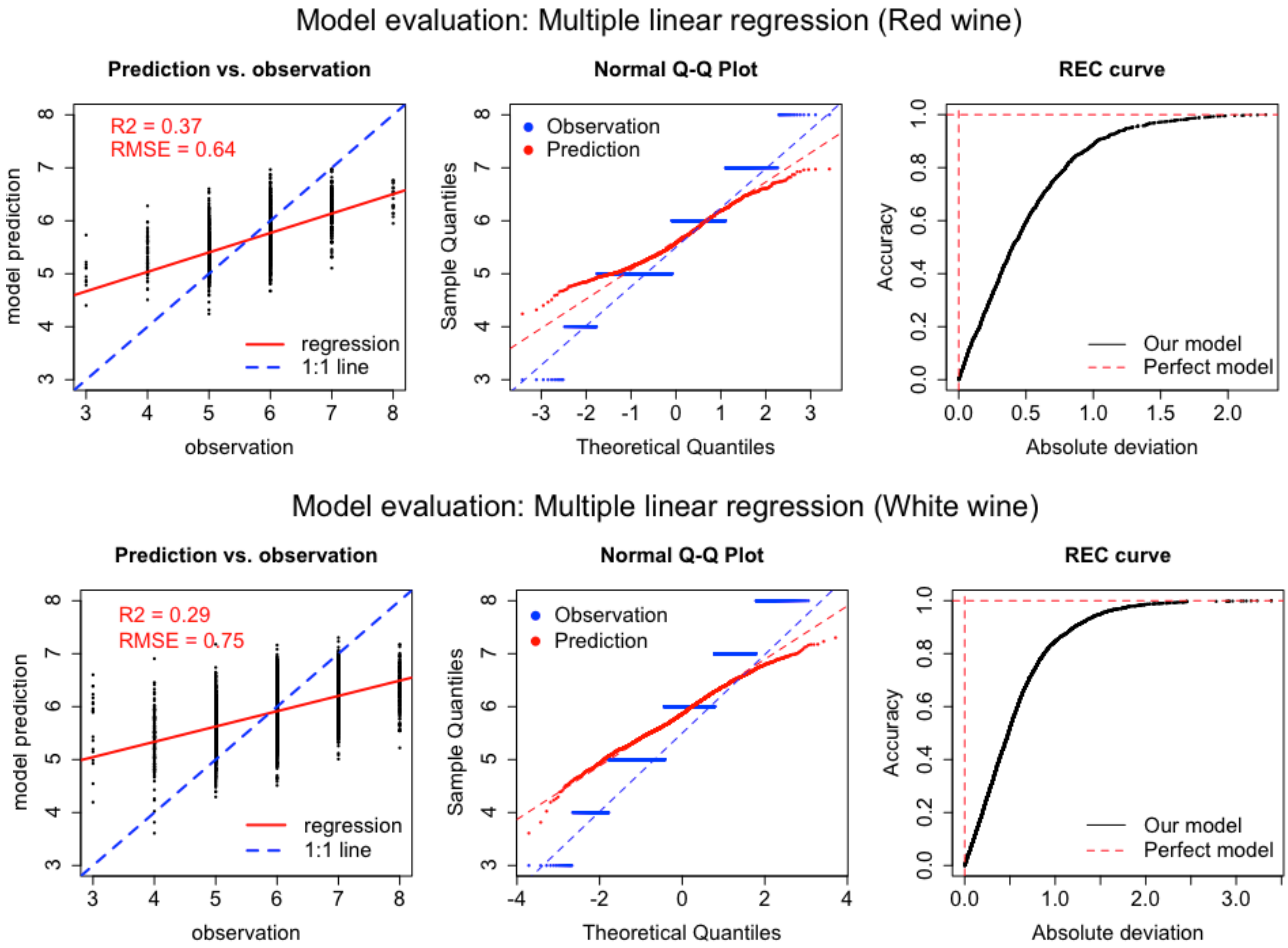

Figure 5. Evaluations of multiple linear regression models for red and white wines.

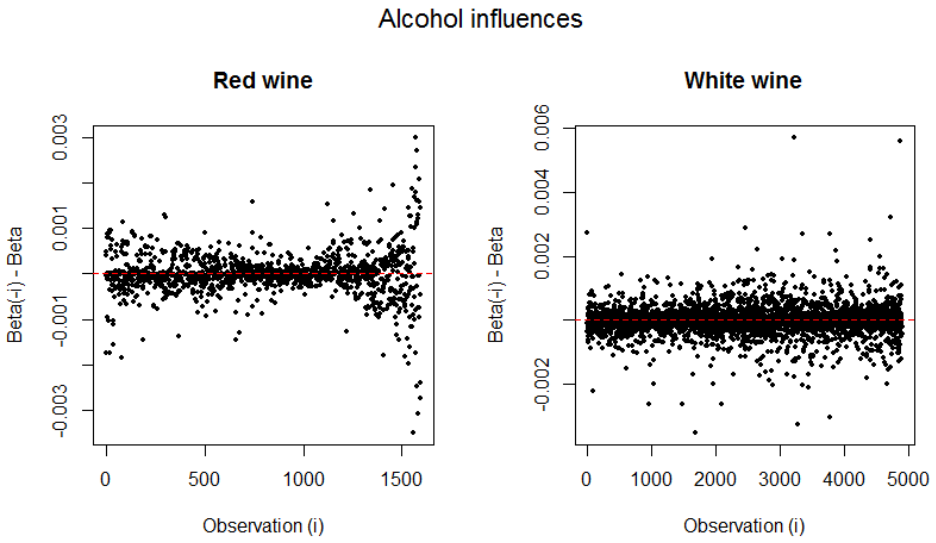

Figure 6. The influences of observational points on multiple linear regression coefficient of alcohol. The observations have been sorted by an increasing order of alcohol amount.

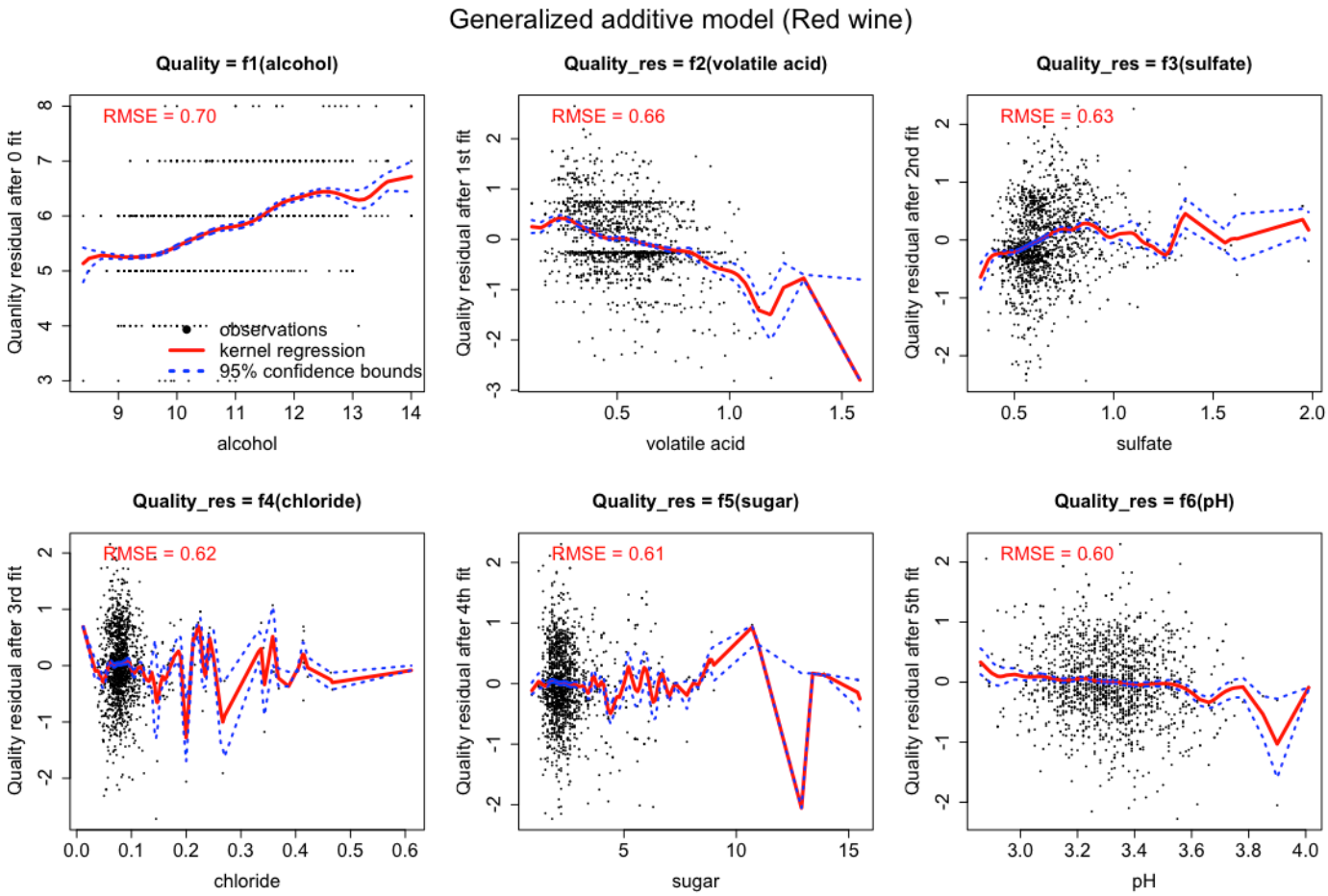

Figure 7. Generalized additive model with kernel regression for red wine.

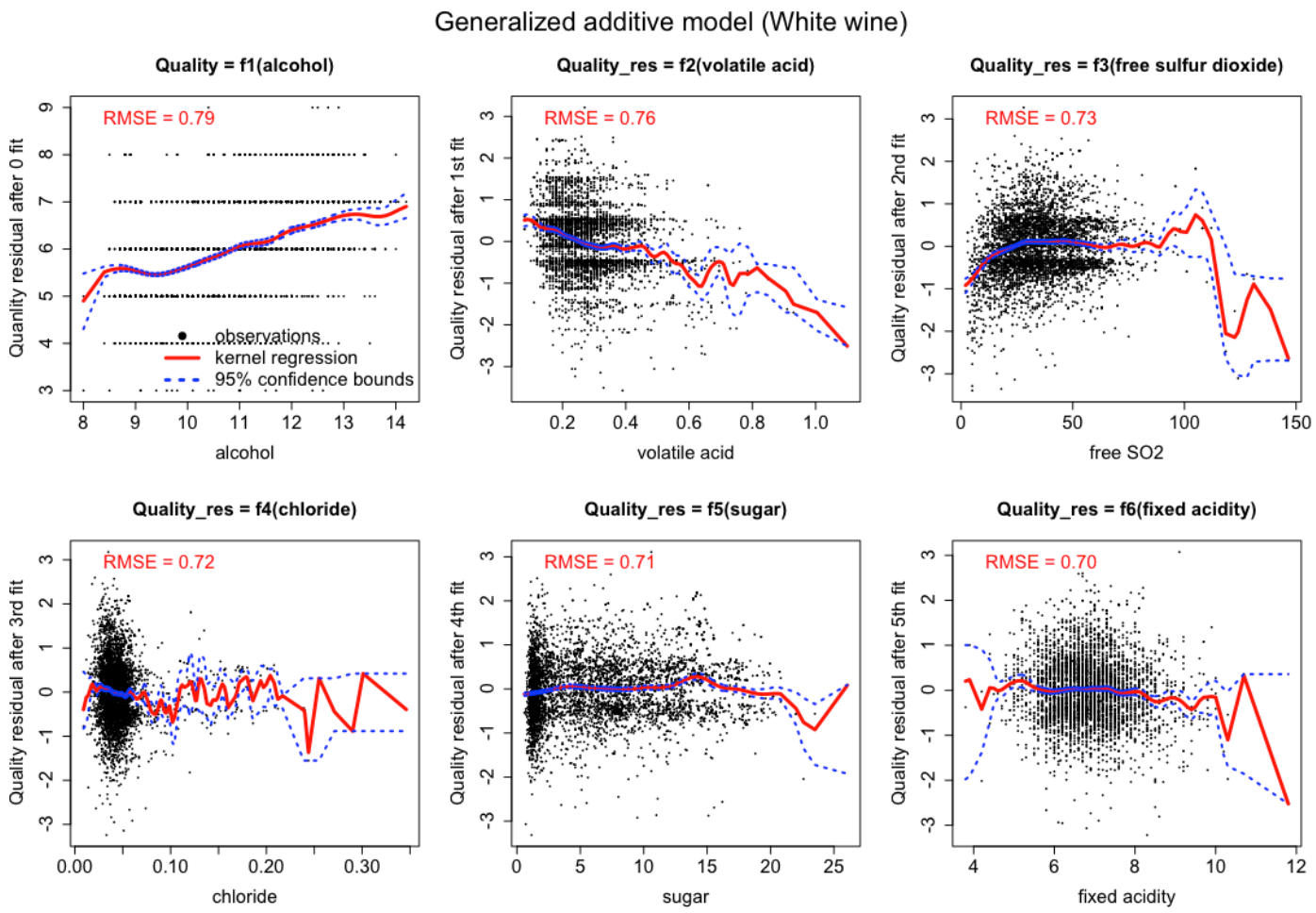

Figure 8. Generalized additive model with kernel regression for white wine.

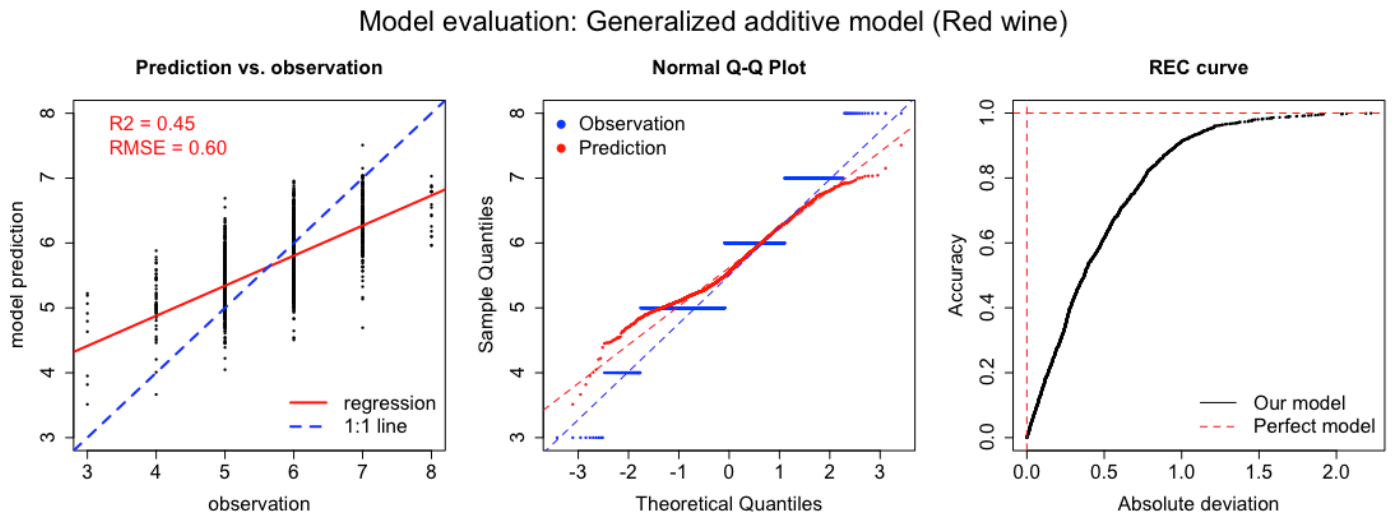

Figure 9. Evaluations of generalized additive models for red wines.

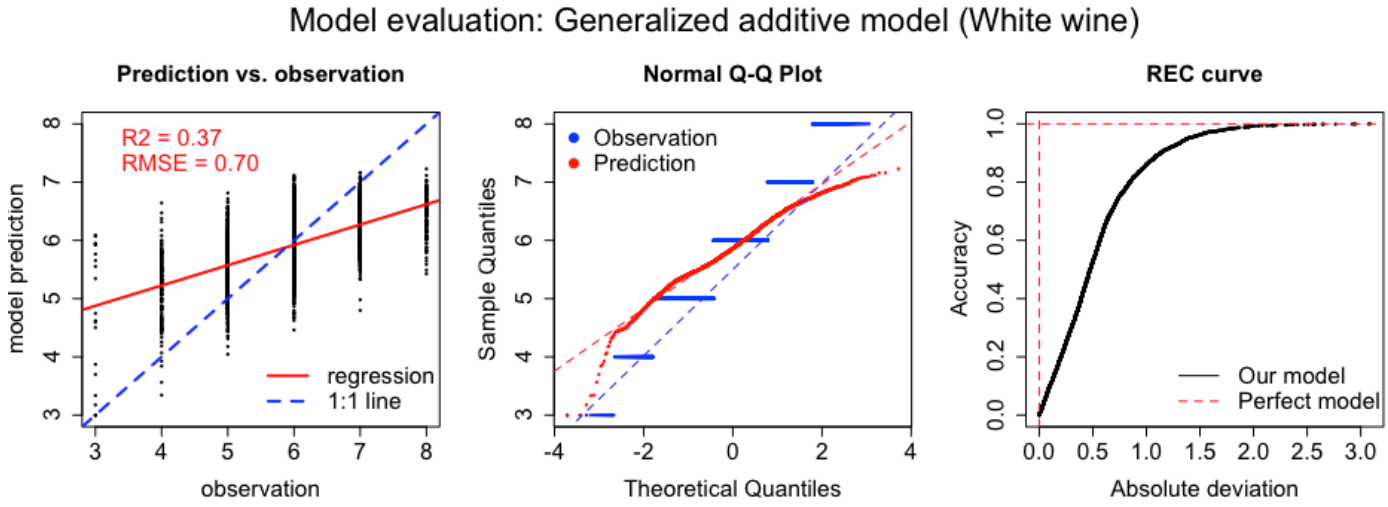

Figure 10. Evaluations of generalized additive models for red and white wines.

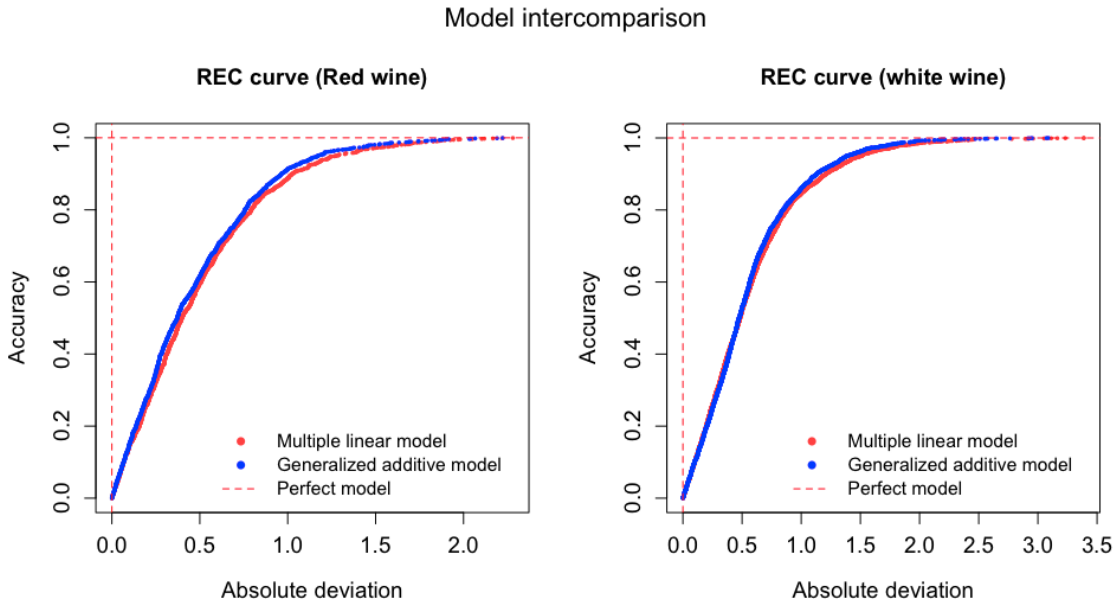

Figure 11. REC curves for comparison between generalized additive model and multiple linear regression model.

C codes

```
#include <R.h>
#include <Rmath.h>

double gauss(double x, double sd) {
        return 1.0/(sqrt(2 * acos(-1.0)) * sd) * exp(-(x*x)/(2*sd*sd));
}

/* x and y will be the data vectors each of length n,
b is the bandwidth,
g2 will be the vector of m gridpoints where we want to calculate the
kernel regression estimates, est2, which is thus also of length m.
*/
void kernreg2 (double *x, double *y, int *n, double *b,
                    double *g2, int *m, double *est2)
{
       int i,j;
       double a1,a2,c;
       for(i = 0; i < *m; i++){
        a1 = 0.0;
                a2 = 0.0;
                for(j=0; j < *n; j++){
                //Guassian kernal is used.
                c = gauss(x[j]-g2[i], *b);
                        a1 += y[j] * c;
                        a2 += c;
        }
                if(a2 > 0.0) est2[i] = a1/a2;
                else est2[i] = 0.0;
       }
}
```

R codes

```
######## STATS202A Final Project ########
### Group: Cenlin He, Liuli Chen, Verma Ankush ###
### Date: Nov. 27th, 2014 ###

#### Project Title: Wine quality and its controlling factors ####
```

```
############## Section One: Data introduction ####################
# Data filename: "winequality-red.csv" & "winequality-white.csv"
# Data source: https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/
# Data: red wine & white wine
# Number of samples: red wine (1599), white wine (4898)
# Input variables (based on physicochemical tests): fixed acidity, volatile acidity,
#           citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide,
#           density, pH, sulphates, alcohol
# Output variable (based on sensory data): wine quality (score between 0 (bad) and 10 (good))
##################    END Section One    ######################

################ Section Two: Data exploration ####################
## Read in data
data_red     = read.table("winequality-red.csv", header=T, sep=";")
data_white = read.table("winequality-white.csv", header=T, sep=";")
## remove observations with NaN
wine_red = na.omit(data_red)
rm(data_red)       # release RAM space
wine_white = na.omit(data_white)
rm(data_white)    # release RAM space

## assign values to variables
# Red wine case
red_quality     = as.numeric(wine_red$quality)
red_facid        = wine_red$fixed.acidity
red_vacid        = wine_red$volatile.acidity
red_cacid        = wine_red$citric.acid
red_sugar        = wine_red$residual.sugar
red_chlor        = wine_red$chlorides
red_fso2         = wine_red$free.sulfur.dioxide
red_tso2         = wine_red$total.sulfur.dioxide
red_dens         = wine_red$density
red_ph           = wine_red$pH
red_so4          = wine_red$sulphates
red_alco         = wine_red$alcohol
red_totset       = cbind(red_quality,red_facid,red_vacid,red_cacid,red_chlor,red_fso2,
                         red_tso2,red_so4,red_dens,red_ph,red_sugar,red_alco)
rm(wine_red)

# White wine case
white_quality     = as.numeric(wine_white$quality)
```

```
white_facid        = wine_white$fixed.acidity
white_vacid        = wine_white$volatile.acidity
white_cacid        = wine_white$citric.acid
white_sugar        = wine_white$residual.sugar
white_chlor        = wine_white$chlorides
white_fso2         = wine_white$free.sulfur.dioxide
white_tso2         = wine_white$total.sulfur.dioxide
white_dens         = wine_white$density
white_ph           = wine_white$pH
white_so4          = wine_white$sulphates
white_alco         = wine_white$alcohol
white_totset       = cbind(white_quality,white_facid,white_vacid,white_cacid,white_chlor,white_fso2,
                           white_tso2,white_so4,white_dens,white_ph,white_sugar,white_alco)
rm(wine_white)

## Exploratory data analysis
# 1. Statistical summary of variables
# please first install package "psych"
# install.packages("psych")
library(psych)
describe(red_totset)
describe(white_totset)

# 2. Distribution of wine quality
par(mfrow=c(1,2))
hist(red_quality,nclass=5,freq=F,main="Wine quality distribution",
       xlab="Wine quality",ylim=c(0,0.5),xlim=c(0,10),col=rgb(1,0,0,0.7))
hist(white_quality,nclass=6,freq=F,add=T,col=rgb(0,0,1,0.5))
legend(x=6.5,y=0.55,legend=c("Red","White"),bty="n",
          col=c(rgb(1,0,0,0.7),rgb(0,0,1,0.5)),pch=c(15,15))
boxplot(list(Red=red_quality,White=white_quality),main="Boxplot of wine quality",
            ylim=c(0,10),ylab="Wine quality")

# 3.Correlation analysis
corr_red = corr.test(red_totset)     # corr.test() is in package "psych"
round(corr_red$r,digit=2)     # pairwise correlation coefficient
round(corr_red$p,digit=2)     # pairwise p-value for correlation
corr_white = corr.test(white_totset)
round(corr_white$r,digit=2)
round(corr_white$p,digit=2)
```

```
# 4. kernel density estimate for key variables
## Red wine
# var 1: alcohol
bandw = bw.nrd(red_alco)       # bandwidth, Scott 1992 formula
rf_alco = density(red_alco,bw=bandw,n=100,from=min(red_alco),to=max(red_alco))    # density function
# var 2: volatile acidity
bandw = bw.nrd(red_vacid)
rf_vacid = density(red_vacid,bw=bandw,n=100,from=min(red_vacid),to=max(red_vacid))
# var 3: citric acid
bandw = bw.nrd(red_cacid)
rf_cacid = density(red_cacid,bw=bandw,n=100,from=min(red_cacid),to=max(red_cacid))
# var 4: sulfate
bandw = bw.nrd(red_so4)
rf_so4 = density(red_so4,bw=bandw,n=100,from=min(red_so4),to=max(red_so4))
# plot
par(mfrow=c(2,2),oma=c(0,0,2,0))
hist(red_alco,freq=F,main="Alcohol", xlab="alcohol",ylab="Density")
lines(rf_alco,col="Red",lwd=3)    # plot the kernel density curve
legend(x=9.6,y=0.55,legend=c("kernel density estimates"),bty="n",lty=1,lwd=3,col="Red")
hist(red_vacid,freq=F,main="volatile acidity", xlab="volatile acidity",ylab="Density",xlim=c(0,2))
lines(rf_vacid,col="Red",lwd=3)
hist(red_cacid,freq=F,main="Citric acid", xlab="citric acid",ylab="Density",xlim=c(0,1))
lines(rf_cacid,col="Red",lwd=3)
hist(red_so4,freq=F,main="Sulfate", xlab="sulfate",ylab="Density",xlim=c(0,2))
lines(rf_so4,col="Red",lwd=3)
mtext("Red wine",outer=T,cex=1.4)

## White wine
# var 1: alcohol
bandw = bw.nrd(white_alco)       # bandwidth, Scott 1992 formula
wf_alco = density(white_alco,bw=bandw,n=100,from=min(white_alco),to=max(white_alco))    # density function
# var 2: density
bandw = bw.nrd(white_dens)
wf_dens = density(white_dens,bw=bandw,n=100,from=min(white_dens),to=max(white_dens))
# var 3: volatile acidity
bandw = bw.nrd(white_vacid)
wf_vacid = density(white_vacid,bw=bandw,n=100,from=min(white_vacid),to=max(white_vacid))
# var 4: chloride
bandw = bw.nrd(white_chlor)
wf_chlor = density(white_chlor,bw=bandw,n=100,from=min(white_chlor),to=max(white_chlor))
# plot
```

```
par(mfrow=c(2,2),oma=c(0,0,2,0))
hist(white_alco,freq=F,main="Alcohol", xlab="alcohol",ylab="Density")
lines(wf_alco,col="Red",lwd=3)    # plot the kernel density curve
legend(x=9.5,y=0.4,legend=c("kernel density estimates"),bty="n",lty=1,lwd=3,col="Red")
hist(white_dens,freq=F,main="Wine density", xlab="wine density",ylab="Density",xlim=c(0.98,1.05),ylim=c(0,120))
lines(wf_dens,col="Red",lwd=3)
hist(white_vacid,freq=F,main="Volatile acidity", xlab="volatile acidity",ylab="Density",xlim=c(0,1.2),ylim=c(0,5))
lines(wf_vacid,col="Red",lwd=3)
hist(white_chlor,freq=F,main="Chlorides", xlab="Chlorides",ylab="Density",xlim=c(0,0.4),ylim=c(0,35))
lines(wf_chlor,col="Red",lwd=3)
mtext("White wine",outer=T,cex=1.4)

# 5. pairwise scatterplot for correlation analysis
## Red wine
par(mfrow=c(2,2),oma=c(0,0,2,0))
# var 1: alcohol
rmod_alco = lm(red_quality ~ red_alco)
plot(red_alco,red_quality,main="Quality vs. Alcohol",xlab="Alcohol",ylab="Wine quality",pch=16,cex=0.5)
abline(rmod_alco,lty=1,lwd=2,col="Red")
legend("bottomright",legend="r = 0.48",text.col="Red",bty="n")
# var 2: volatile acidity
rmod_vacid = lm(red_quality ~ red_vacid)
plot(red_vacid,red_quality,main="Quality vs. Volatile acid",xlab="Volatile acid",ylab="Wine quality",pch=16,cex=0.5)
abline(rmod_vacid,lty=1,lwd=2,col="Red")
legend("topright",legend="r = -0.39",text.col="Red",bty="n")
# var 3: citric acid
rmod_cacid = lm(red_quality ~ red_cacid)
plot(red_cacid,red_quality,main="Quality vs. Citric acid",xlab="Citric acid",ylab="Wine quality",pch=16,cex=0.5)
abline(rmod_cacid,lty=1,lwd=2,col="Red")
legend("topright",legend="r = 0.23",text.col="Red",bty="n")
# var 4: sulfate
rmod_so4 = lm(red_quality ~ red_so4)
plot(red_so4,red_quality,main="Quality vs. Sulfate",xlab="sulfate",ylab="Wine quality",pch=16,cex=0.5)
abline(rmod_so4,lty=1,lwd=2,col="Red")
legend("bottomright",legend="r = 0.25",text.col="Red",bty="n")
mtext("Red wine",outer=T,cex=1.4)

## White wine
par(mfrow=c(2,2),oma=c(0,0,2,0))
# var 1: alcohol
wmod_alco = lm(white_quality ~ white_alco)
```

```
plot(white_alco,white_quality,main="Quality vs. Alcohol",xlab="Alcohol",ylab="Wine quality",pch=16,cex=0.5)
abline(wmod_alco,lty=1,lwd=2,col="Red")
legend("bottomright",legend="r = 0.44",text.col="Red",bty="n")
# var 2: Density
wmod_dens = lm(white_quality ~ white_dens)
plot(white_dens,white_quality,main="Quality vs. Density",xlab="Density",ylab="Wine quality",pch=16,cex=0.5)
abline(wmod_dens,lty=1,lwd=2,col="Red")
legend("topright",legend="r = -0.31",text.col="Red",bty="n")
# var 3: volatile acid
wmod_vacid = lm(white_quality ~ white_vacid)
plot(white_vacid,white_quality,main="Quality vs. Volatile acid",xlab="Volatile acid",ylab="Wine quality",pch=16,cex=0.5)
abline(wmod_vacid,lty=1,lwd=2,col="Red")
legend("topright",legend="r = -0.19",text.col="Red",bty="n")
# var 4: chloride
wmod_chlor = lm(white_quality ~ white_chlor)
plot(white_chlor,white_quality,main="Quality vs. Chloride",xlab="chloride",ylab="Wine quality",pch=16,cex=0.5)
abline(wmod_chlor,lty=1,lwd=2,col="Red")
legend("topright",legend="r = -0.21",text.col="Red",bty="n")
mtext("White wine",outer=T,cex=1.4)

# 6. outlier detection and removal
### red wine
par(mfrow=c(1,2),oma=c(0,0,2,0))
#ox1 = red_alco
ox2 = red_tso2
ox3 = red_cacid
oy = red_quality
#plot(ox1,oy,main="Alcohol",xlab="alcohol",ylab="Quality",pch=16,cex=0.8)
plot(ox2,oy,main="Total sulfur dioxide",xlab="total SO2",ylab="Quality",pch=16,cex=0.5)
plot(ox3,oy,main="Citric acid",xlab="Citric acid",ylab="Quality",pch=16,cex=0.5)
mtext("Red wine",outer=T, cex=1.4)

## white wine
par(mfrow=c(2,2),oma=c(0,0,2,0))
ox1 = white_facid
ox2 = white_sugar
ox3 = white_fso2
ox4 = white_dens
oy = white_quality
plot(ox1,oy,main="fixed acidity",xlab="fixed acidity",ylab="Quality",pch=16,cex=0.6)
```

```
plot(ox2,oy,main="Sugar",xlab="sugar",ylab="Quality",pch=16,cex=0.6)
plot(ox3,oy,main="free SO2",xlab="free SO2",ylab="Quality",pch=16,cex=0.6)
plot(ox4,oy,main="Density",xlab="density",ylab="Quality",pch=16,cex=0.6)
mtext("White wine",outer=T, cex=1.4)

## remove outliers
# red wine
ry = red_quality[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rfa = red_facid[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rva = red_vacid[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rca = red_cacid[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rsu = red_sugar[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rch = red_chlor[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rfs = red_fso2[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rts = red_tso2[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rde = red_dens[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rph = red_ph[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
rso = red_so4[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
ral = red_alco[red_cacid < 0.9 & red_tso2 < 250 & red_alco < 14.5]
# white wine
wy = white_quality[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wfa = white_facid[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wva = white_vacid[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wca = white_cacid[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wsu = white_sugar[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wch = white_chlor[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wfs = white_fso2[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wts = white_tso2[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wde = white_dens[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wph = white_ph[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wso = white_so4[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
wal = white_alco[white_facid < 13 & white_sugar < 40 & white_fso2 < 200 & white_dens < 1.01]
##############################    END Section Two    ##############################

############### Section Three: Building prediction model ######################
### 1. multiple linear regression model
# red wine
rfl_back = step(lm(ry ~ rfa + rva + rca + rsu + rch + rfs + rts + rde + rph + rso + ral),
                direction = "backward")
summary(rfl_back)
```

```
rfl_forw = step(lm(ry ~ rfa + rva + rca + rsu + rch + rfs + rts + rde + rph + rso + ral),
                direction = "forward")
summary(rfl_forw)
rfl_both = step(lm(ry ~ rfa + rva + rca + rsu + rch + rfs + rts + rde + rph + rso + ral),
                direction = "both")
summary(rfl_both)
# relative importance (standard regression coefficient)
rcoef_st = rfl_back$coefficients[2:8] / c(sd(rva),sd(rch),sd(rfs),sd(rts),sd(rph),sd(rso),sd(ral))
round(rcoef_st,digits=3)

# white wine
wfl_back = step(lm(wy ~ wfa + wva + wca + wsu + wch + wfs + wts + wde + wph + wso + wal),
                direction = "backward")
summary(wfl_back)
wfl_forw = step(lm(wy ~ wfa + wva + wca + wsu + wch + wfs + wts + wde + wph + wso + wal),
                direction = "forward")
summary(wfl_forw)
wfl_both = step(lm(wy ~ wfa + wva + wca + wsu + wch + wfs + wts + wde + wph + wso + wal),
                direction = "both")
summary(wfl_both)
# relative importance (standard regression coefficient)
wcoef_st = wfl_back$coefficients[2:9] / c(sd(wfa),sd(wva),sd(wsu),sd(wfs),sd(wde),sd(wph),sd(wso),sd(wal))
round(wcoef_st,digits=3)

### kernel regression for four key variables
system("R CMD SHLIB gaussian.c")
dyn.load("gaussian.so")

## red wine
var_name1 = c("Chlorides","Volatile acidity","pH","Sulfates")
xv = cbind(rch,rva,rph,rso)
len = length(ry)
par(mfrow=c(2,2),oma=c(0,0,2,0))
for (i in 1:4) {
        y = ry
        x = xv[,i]
        bw = bw.nrd(x)
        m = 100
        ma = max(x)
        mi = min(x)
        le = mi-3*bw
```

```
        ri = ma+3*bw
        g2= seq(le, ri, length=m)

        a = .C("kernreg2", as.double(x), as.double(y), as.integer(len),
                as.double(bw), as.double(g2), as.integer(m), est2=double(m))

        plot(c(le, ri), c(min(y), max(y)), type="n", xlab=var_name1[i] , ylab="quality",main=var_name1[i])
        points(x, y, pch=16, cex=0.5)
        lines(g2, a$est2, col="red",lwd=3)

        rec = matrix(, nrow = 0, ncol = 5000)
        for (step in 1:200) {
                b = sample(1:len, 5000, rep=T)
                x2 = double(5000)
                y2 = double(5000)
                for (i in 1:5000) {
                        x2[i] = x[b[i]]
                        y2[i] = y[b[i]]
                }
                a2 = .C("kernreg2", as.double(x2), as.double(y2), as.integer(5000),
                        as.double(bw), as.double(g2), as.integer(m), est2=double(m))
                rec = rbind(rec, a2$est2)
        }

        r5 = c()
        r195 = c()
        for (i in 1:m) {
                v = as.vector(rec[,i])
                v = sort(v)
                r5[i] = v[5]
                r195[i] = v[195]
        }
        lines(g2, r5, lty=3, col="blue",lwd=3)
        lines(g2, r195, lty=3, col="green",lwd=3)
}
mtext("Red wine",outer=T, cex=1.4)

## white wine
var_name1 = c("Density","Volatile acidity","pH","Sulfates")
xv = cbind(wde,wva,wph,wso)
len = length(wy)
```

```
par(mfrow=c(2,2),oma=c(0,0,2,0))
for (i in 1:4) {
	y = wy
	x = xv[,i]
	bw = bw.nrd(x)
	m = 100
	ma = max(x)
	mi = min(x)
	le = mi-3*bw
	ri = ma+3*bw
	g2= seq(le, ri, length=m)

	a = .C("kernreg2", as.double(x), as.double(y), as.integer(len),
		as.double(bw), as.double(g2), as.integer(m), est2=double(m))

	plot(c(le, ri), c(min(y), max(y)), type="n", xlab=var_name1[i] , ylab="quality",main=var_name1[i])
	points(x, y, pch=16, cex=0.5)
	lines(g2, a$est2, col="red",lwd=3)

	rec = matrix(, nrow = 0, ncol = 5000)
	for (step in 1:200) {
		b = sample(1:len, 5000, rep=T)
		x2 = double(5000)
		y2 = double(5000)
		for (i in 1:5000) {
			x2[i] = x[b[i]]
			y2[i] = y[b[i]]
		}
		a2 = .C("kernreg2", as.double(x2), as.double(y2), as.integer(5000),
			as.double(bw), as.double(g2), as.integer(m), est2=double(m))
		rec = rbind(rec, a2$est2)
	}

	r5 = c()
	r195 = c()
	for (i in 1:m) {
		v = as.vector(rec[,i])
		v = sort(v)
		r5[i] = v[5]
		r195[i] = v[195]
	}
```

```
        lines(g2, r5, lty=3, col="blue",lwd=3)
        lines(g2, r195, lty=3, col="green",lwd=3)
}
mtext("White wine",outer=T, cex=1.4)

### Error analysis
# red wine
pred_r = rep(0,length(ry))    # prediction
xr = cbind(rep(1,length(ry)),rva,rch,rfs,rts,rph,rso,ral)
for (i in 1:8) {
        pred_r = pred_r + rfl_back$coefficients[i] * xr[,i]
}
res_r = pred_r - ry
RMSE_r = sqrt(mean((pred_r - ry)^2))
par(mfrow=c(1,3),oma=c(0,0,2,0))
# Residual plot
#plot(pred_r,res_r,pch=16,cex=0.5,xlab="model prediction",ylab="residuals",main="Residuals",xlim=c(3,8))
#abline(h=0,lty=2,col="red")
#legend(x=2.5,y=3,legend="RMSE = 0.64",text.col="Red",bty="n")
# Prediction vs. observation plot
plot(c(3,8),c(3,8),type="n",ylab="model prediction",xlab="observation",main="Prediction vs.
observation",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
points(ry,pred_r,pch=16,cex=0.5)
modr = lm(pred_r ~ ry)
abline(modr,lty=1,lwd=2,col="Red")
abline(a=0,b=1,lty=2,lwd=2,col="blue")
legend("topleft",legend=c("R2 = 0.37","RMSE = 0.64"),text.col="Red",bty="n",cex=1.5)
legend("bottomright",legend=c("regression","1:1 line"),lty=c(1,2),lwd=2,col=c("Red","blue"),bty="n",cex=1.5)
#qq plot
qqnorm(ry,ylim=c(3,8),pch=16,cex=0.5,col="blue",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
q2 = qqnorm(pred_r,plot.it=F)
points(q2,pch=16,cex=0.5,col="red")
qqline(ry,lty=2,lwd=1,col="blue")
qqline(pred_r,lty=2,lwd=1,col="red")
legend("topleft",legend=c("Observation","Prediction"),pch=16,col=c("blue","red"),bty="n",cex=1.5)
# REC plot
RECcal <- function(predict, actual)   {
        err_prev = 0.0
        correct = 0.0
        n = length(actual)
        rec_x = rep(NA,n)
```

```
        rec_y = rep(NA,n)
        err_orig = abs(predict - actual)
        err = sort(err_orig,decreasing = F)
        for (i in 1:n) {
                if (err[i] > err_prev) {
                        rec_x[i] = err_prev
                        rec_y[i] = correct / n
                        err_prev = err[i]
                }
                correct = correct + 1.0
        }
        return(cbind(rec_x,rec_y))
}
rREC = RECcal(pred_r,ry)
plot(rREC[,1],rREC[,2],main="REC curve",xlab="Absolute deviation",ylab="Accuracy",
      pch=16,cex=0.5,cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
abline(h=1.0,lty=2,col="red")
abline(v=0.0,lty=2,col="red")
legend("bottomright",legend=c("Our model","Perfect model"),lty=c(1,2),col=c("black","red"),bty="n",cex=1.5)
mtext("Model evaluation: Multiple linear regression (Red wine)",outer=T, cex=1.4)

## white wine
pred_w = rep(0,length(wy))    # prediction
xw = cbind(rep(1,length(wy)),wfa,wva,wsu,wfs,wde,wph,wso,wal)
for (i in 1:9) {
        pred_w = pred_w + wfl_back$coefficients[i] * xw[,i]
}
res_w = pred_w - wy
RMSE_w = sqrt(mean((pred_w - wy)^2))
par(mfrow=c(1,3),oma=c(0,0,2,0))
# Residual plot
#plot(pred_w,res_w,pch=16,cex=0.5,xlab="model prediction",ylab="residuals",main="Residuals",xlim=c(3,8))
#abline(h=0,lty=2,col="red")
#legend(x=2.5,,y=4,legend="RMSE = 0.75",text.col="Red",bty="n")
# Prediction vs. observation plot
plot(c(3,8),c(3,8),type="n",ylab="model                    prediction",xlab="observation",main="Prediction                    vs.
observation",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
points(wy,pred_w,pch=16,cex=0.5)
modw = lm(pred_w ~ wy)
abline(modw,lty=1,lwd=2,col="Red")
abline(a=0,b=1,lty=2,lwd=2,col="blue")
```

```
legend("topleft",legend=c("R2 = 0.29","RMSE = 0.75"),text.col="Red",bty="n",cex=1.5)
legend("bottomright",legend=c("regression","1:1 line"),lty=c(1,2),lwd=2,col=c("Red","blue"),bty="n",cex=1.5)
#qq plot
qqnorm(wy,ylim=c(3,8),pch=16,cex=0.5,col="blue",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
q2 = qqnorm(pred_w,plot.it=F)
points(q2,pch=16,cex=0.5,col="red")
qqline(wy,lty=2,lwd=1,col="blue")
qqline(pred_w,lty=2,lwd=1,col="red")
legend("topleft",legend=c("Observation","Prediction"),pch=16,col=c("blue","red"),bty="n",cex=1.5)
# REC plot
wREC = RECcal(pred_w,wy)
plot(wREC[,1],wREC[,2],main="REC curve",xlab="Absolute deviation",ylab="Accuracy",
      pch=16,cex=0.5,cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
abline(h=1.0,lty=2,col="red")
abline(v=0.0,lty=2,col="red")
legend("bottomright",legend=c("Our model","Perfect model"),lty=c(1,2),col=c("black","red"),bty="n",cex=1.5)
mtext("Model evaluation: Multiple linear regression (White wine)",outer=T, cex=1.4)

### influential points for alcohol
## red wine
raln = sort(ral)
ryn = ry[order(ral)]
rvan = rva[order(ral)]
rchn = rch[order(ral)]
rfsn = rfs[order(ral)]
rtsn = rts[order(ral)]
rphn = rph[order(ral)]
rson = rso[order(ral)]
beta1 = matrix(0,length(ryn))
for (i in 1:length(ryn)) {
       model2 = lm(ryn[-i] ~ rvan[-i] + rchn[-i] + rfsn[-i] + rtsn[-i] + rphn[-i] + rson[-i] + raln[-i])
       beta1[i] = model2$coefficients[8] - rfl_back$coefficients[8]
}
par(mfrow=c(1,2),oma=c(0,0,2,0))
plot(c(0,length(ryn)),c(min(beta1),max(beta1)),type="n",xlab="Observation (i)",
      ylab="Beta(-i) - Beta",main="Red wine")
points(1:length(ryn),beta1,pch=16,cex=0.5)
abline(h=0,lty=2,col="red")

## white wine
waln = sort(wal)
```

```
wyn = wy[order(wal)]
wfan = wfa[order(wal)]
wvan = wva[order(wal)]
wsun = wsu[order(wal)]
wfsn = wfs[order(wal)]
wden = wde[order(wal)]
wphn = wph[order(wal)]
wson = wso[order(wal)]
beta1 = matrix(0,length(wyn))
for (i in 1:length(wyn)) {
        model2 = lm(wyn[-i] ~ wfan[-i] + wvan[-i] + wsun[-i] + wfsn[-i] + wden[-i] + wphn[-i] + wson[-i] + waln[-i])
        beta1[i] = model2$coefficients[9] - wfl_back$coefficients[9]
}
plot(c(0,length(wyn)),c(min(beta1),max(beta1)),type="n",xlab="Observation (i)",
      ylab="Beta(-i) - Beta",main="White wine")
points(1:length(wyn),beta1,pch=16,cex=0.5)
abline(h=0,lty=2,col="red")
mtext("Alcohol influences",outer=T,cex=1.4)


############ Generalized additive models with N-W kernel regression ##########
### Red wines
y1 = ry
n = length(ry)
par(mfrow=c(2,3),oma=c(0,0,2,0))
# var1: alcohol
xreg = ral
yy = y1
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
          as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality = f1(alcohol)",xlab="alcohol",ylab="Quanlity residual after 0
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)   # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)   # sample the index of observations
        sam_x = xreg[sam_ind]   # sample X
        sam_y = yy[sam_ind] # sample Y
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
```

```
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
legend("bottomright", legend=c("observations","kernel regression","95% confidence bounds"),
col=c("black","red","blue"), pch=c(16,NA,NA),lty=c(0,1,3), lwd=c(0,3,3),bty="n",cex=1.5)
fit1 = r_kreg$est2
rfit1 = fit1
y2 = yy - fit1   # residual
#plot(fit1,y2)   # check if residual is randomly distributed vs. predicted quality
RMSE = c(sqrt(mean(y2^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.70",text.col="red",bty="n",cex=1.5)

# var2: volatile acidity
xreg = rva
yy=y2
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
          as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f2(volatile acid)",xlab="volatile acid",ylab="Quality residual after 1st
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)   # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
```

```
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit2 = r_kreg$est2
rfit2 = fit2
y3 = yy - fit2
#plot(fit2,y3)
RMSE = c(RMSE,sqrt(mean(y3^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.66",text.col="red",bty="n",cex=1.5)

# var3: sulfate
xreg = rso
yy=y3
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
          as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f3(sulfate)",xlab="sulfate",ylab="Quality residual after 2nd
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)    # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
```

```
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit3 = r_kreg$est2
rfit3 = fit3
y4 = yy - fit3
#plot(fit3,y4)
RMSE = c(RMSE,sqrt(mean(y4^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.63",text.col="red",bty="n",cex=1.5)

# var4: chlorides
xreg = rch
yy=y4
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
         as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f4(chloride)",xlab="chloride",ylab="Quality residual after 3rd
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)    # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
```

```
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit4 = r_kreg$est2
rfit4 = fit4
y5 = yy - fit4
#plot(fit4,y5)
RMSE = c(RMSE,sqrt(mean(y5^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.62",text.col="red",bty="n",cex=1.5)

# var5: sugar
xreg = rsu
yy=y5
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
          as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f5(sugar)",xlab="sugar",ylab="Quality residual after 4th
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)    # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit5 = r_kreg$est2
```

```
rfit5 = fit5
y6 = yy - fit5
#plot(fit5,y6)
RMSE = c(RMSE,sqrt(mean(y6^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.61",text.col="red",bty="n",cex=1.5)

# var6: pH
xreg = rph
yy=y6
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f6(pH)",xlab="pH",ylab="Quality residual after 5th
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)    # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit6 = r_kreg$est2
rfit6 = fit6
y7 = yy - fit6
RMSE = c(RMSE,sqrt(mean(y7^2)))
print(RMSE)
```

```
legend("topleft",legend="RMSE = 0.60",text.col="red",bty="n",cex=1.5)
mtext("Generalized additive model (Red wine)",outer=T,cex=1.4)

### Error analysis red wine
# red wine
pred_r = rfit1 + rfit2 + rfit3 + rfit4 + rfit5 + rfit6
res_r = pred_r - ry
RMSE_r = sqrt(mean((pred_r - ry)^2))
par(mfrow=c(1,3),oma=c(0,0,2,0))
# Prediction vs. observation plot
plot(c(3,8),c(3,8),type="n",ylab="model prediction",xlab="observation",main="Prediction vs.
observation",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
points(ry,pred_r,pch=16,cex=0.5)
modr = lm(pred_r ~ ry)
abline(modr,lty=1,lwd=2,col="Red")
abline(a=0,b=1,lty=2,lwd=2,col="blue")
legend("topleft",legend=c("R2 = 0.45","RMSE = 0.60"),text.col="Red",bty="n",cex=1.5)
legend("bottomright",legend=c("regression","1:1 line"),lty=c(1,2),lwd=2,col=c("Red","blue"),bty="n",cex=1.5)
#qq plot
qqnorm(ry,ylim=c(3,8),pch=16,cex=0.5,col="blue",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
q2 = qqnorm(pred_r,plot.it=F)
points(q2,pch=16,cex=0.5,col="red")
qqline(ry,lty=2,lwd=1,col="blue")
qqline(pred_r,lty=2,lwd=1,col="red")
legend("topleft",legend=c("Observation","Prediction"),pch=16,col=c("blue","red"),bty="n",cex=1.5)
# REC plot
rREC = RECcal(pred_r,ry)
plot(rREC[,1],rREC[,2],main="REC curve",xlab="Absolute deviation",ylab="Accuracy",
     pch=16,cex=0.5,cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
abline(h=1.0,lty=2,col="red")
abline(v=0.0,lty=2,col="red")
legend("bottomright",legend=c("Our model","Perfect model"),lty=c(1,2),col=c("black","red"),bty="n",cex=1.5)
mtext("Model evaluation: Generalized additive model (Red wine)",outer=T, cex=1.4)

### white wines:alcohol,vacid,fso2
y1 = wy
n = length(wy)
par(mfrow=c(2,3),oma=c(0,0,2,0))
# var1: alcohol
xreg = wal
yy = y1
```

```
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
                as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality = f1(alcohol)",xlab="alcohol",ylab="Quanlity residual after 0 fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)   # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)   # sample the index of observations
        sam_x = xreg[sam_ind]   # sample X
        sam_y = yy[sam_ind] # sample Y
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
legend("bottomright", legend=c("observations","kernel regression","95% confidence bounds"), col=c("black","red","blue"), pch=c(16,NA,NA),lty=c(0,1,3), lwd=c(0,3,3),bty="n",cex=1.5)
fit1 = r_kreg$est2
wfit1 = fit1
y2 = yy - fit1   # residual
#plot(fit1,y2)   # check if residual is randomly distributed vs. predicted quality
RMSE = c(sqrt(mean(y2^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.79",text.col="red",bty="n",cex=1.5)

# var2: volatile acid
xreg = wva
yy=y2
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
```

```
                as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f2(volatile acid)",xlab="volatile acid",ylab="Quality residual after 1st fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)   # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit2 = r_kreg$est2
wfit2 = fit2
y3 = yy - fit2
#plot(fit2,y3)
RMSE = c(RMSE,sqrt(mean(y3^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.76",text.col="red",bty="n",cex=1.5)

# var3: free so2
xreg = wfs
yy=y3
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
                as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f3(free sulfur dioxide)",xlab="free SO2",ylab="Quality residual after 2nd fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
```

```
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)    # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit3 = r_kreg$est2
wfit3 = fit3
y4 = yy - fit3
#plot(fit3,y4)
RMSE = c(RMSE,sqrt(mean(y4^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.73",text.col="red",bty="n",cex=1.5)

# var4: chloride
xreg = wch
yy=y4
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
                as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f4(chloride)",xlab="chloride",ylab="Quality residual after 3rd
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)    # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
```

```
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit4 = r_kreg$est2
wfit4 = fit4
y5 = yy - fit4
#plot(fit4,y5)
RMSE = c(RMSE,sqrt(mean(y5^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.72",text.col="red",bty="n",cex=1.5)

# var5: sugar
xreg = wsu
yy=y5
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
                as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f5(sugar)",xlab="sugar",ylab="Quality residual after 4th
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)    # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
```

```
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit5 = r_kreg$est2
wfit5 = fit5
y6 = yy - fit5
#plot(fit5,y6)
RMSE = c(RMSE,sqrt(mean(y6^2)))
print(RMSE)
legend("topleft",legend="RMSE = 0.71",text.col="red",bty="n",cex=1.5)

# var6: fixed acidity
xreg = wfa
yy=y6
bw = bw.nrd(xreg)
r_kreg = .C("kernreg2", as.double(xreg), as.double(yy), as.integer(n),
                as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
plot(xreg,yy,main="Quality_res = f6(fixed acidity)",xlab="fixed acidity",ylab="Quality residual after 5th
fit",pch=16,cex=0.4,cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
lines(sort(xreg),r_kreg$est2[order(xreg)],lty=1,lwd=3,col="Red")
# repeat 200 times for resample 100 pairs of observations
sam_fm = matrix(0.0, 200,n)   # store the reults of kernel estimate
for (i in 1:200) {
        sam_ind = sample(1:n, size=5000, replace=T)
        sam_x = xreg[sam_ind]
        sam_y = yy[sam_ind]
        sam_res = .C("kernreg2", as.double(sam_x), as.double(sam_y), as.integer(5000),
                        as.double(bw), as.double(xreg), as.integer(n), est2=double(n))
        sam_fm[i,] = sam_res$est2
}
# compute the 2.5th & 97.5th percentile
fm_p25 = matrix(0,n)
```

```
fm_p975 = matrix(0,n)
for (i in 1:n) {
        fm_p25[i] = quantile(sam_fm[,i],0.025)
        fm_p975[i] = quantile(sam_fm[,i],0.975)
}
# plot 95% unconfidence bpimds
lines(sort(xreg),fm_p25[order(xreg)],col="blue",lwd=2,lty=3)
lines(sort(xreg),fm_p975[order(xreg)],col="blue",lwd=2,lty=3)
fit6 = r_kreg$est2
wfit6 = fit6
y7 = yy - fit6
RMSE = sqrt(mean(y7^2))
print(RMSE)
legend("topleft",legend="RMSE = 0.70",text.col="red",bty="n",cex=1.5)
mtext("Generalized additive model (White wine)",outer=T,cex=1.4)

### Error analysis red wine
pred_w = wfit1 + wfit2 + wfit3 + wfit4 + wfit5 + wfit6
res_w = pred_w - wy
RMSE_w = sqrt(mean((pred_w - wy)^2))
par(mfrow=c(1,3),oma=c(0,0,2,0))
# Prediction vs. observation plot
plot(c(3,8),c(3,8),type="n",ylab="model prediction",xlab="observation",main="Prediction vs.
observation",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
points(wy,pred_w,pch=16,cex=0.5)
modw = lm(pred_w ~ wy)
abline(modw,lty=1,lwd=2,col="Red")
abline(a=0,b=1,lty=2,lwd=2,col="blue")
legend("topleft",legend=c("R2 = 0.37","RMSE = 0.70"),text.col="Red",bty="n",cex=1.5)
legend("bottomright",legend=c("regression","1:1 line"),lty=c(1,2),lwd=2,col=c("Red","blue"),bty="n",cex=1.5)
#qq plot
qqnorm(wy,ylim=c(3,8),pch=16,cex=0.5,col="blue",cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
q2 = qqnorm(pred_w,plot.it=F)
points(q2,pch=16,cex=0.5,col="red")
qqline(wy,lty=2,lwd=1,col="blue")
qqline(pred_w,lty=2,lwd=1,col="red")
legend("topleft",legend=c("Observation","Prediction"),pch=16,col=c("blue","red"),bty="n",cex=1.5)
# REC plot
wREC = RECcal(pred_w,wy)
plot(wREC[,1],wREC[,2],main="REC curve",xlab="Absolute deviation",ylab="Accuracy",
      pch=16,cex=0.5,cex.main=1.5,cex.lab=1.5,cex.axis=1.5)
```

```
abline(h=1.0,lty=2,col="red")
abline(v=0.0,lty=2,col="red")
legend("bottomright",legend=c("Our model","Perfect model"),lty=c(1,2),col=c("black","red"),bty="n",cex=1.5)
mtext("Model evaluation: Generalized additive model (White wine)",outer=T, cex=1.4)

### model intercomparison: MLR vs GAM
# MLR
# red wine
rp_mlr = rep(0,length(ry))    # prediction
xr = cbind(rep(1,length(ry)),rva,rch,rfs,rts,rph,rso,ral)
for (i in 1:8) {
        rp_mlr = rp_mlr + rfl_back$coefficients[i] * xr[,i]
}
rREC_mlr = RECcal(rp_mlr,ry)
# white wine
wp_mlr = rep(0,length(wy))    # prediction
xw = cbind(rep(1,length(wy)),wfa,wva,wsu,wfs,wde,wph,wso,wal)
for (i in 1:9) {
        wp_mlr = wp_mlr + wfl_back$coefficients[i] * xw[,i]
}
wREC_mlr = RECcal(wp_mlr,wy)
# GAM
# red wine
rp_gam = rfit1 + rfit2 + rfit3 + rfit4 + rfit5 + rfit6
rREC_gam = RECcal(rp_gam,ry)
# white wine
wp_gam = wfit1 + wfit2 + wfit3 + wfit4 + wfit5 + wfit6
wREC_gam = RECcal(wp_gam,wy)

par(mfrow=c(1,2),oma=c(0,0,2,0))
plot(rREC_mlr[,1],rREC_mlr[,2],main="REC curve (Red wine)",xlab="Absolute deviation",ylab="Accuracy",
pch=16,cex=0.5,cex.main=1.2,cex.lab=1.2,cex.axis=1.2,col="brown1")
points(rREC_gam[,1],rREC_gam[,2],pch=16,cex=0.5,col="blue")
abline(h=1.0,lty=2,col="red")
abline(v=0.0,lty=2,col="red")
legend("bottomright",legend=c("Multiple linear model","Generalized additive model","Perfect
model"),lty=c(0,0,2),pch=c(16,16,-1),col=c("brown1","blue","red"),bty="n",cex=1)
plot(wREC_mlr[,1],wREC_mlr[,2],main="REC curve (white wine)",xlab="Absolute deviation",ylab="Accuracy",
      pch=16,cex=0.5,cex.main=1.2,cex.lab=1.2,cex.axis=1.2,col="brown1")
points(wREC_gam[,1],wREC_gam[,2],pch=16,cex=0.5,col="blue")
abline(h=1.0,lty=2,col="red")
```

```
abline(v=0.0,lty=2,col="red")
legend("bottomright",legend=c("Multiple linear model","Generalized additive model","Perfect model"),lty=c(0,0,2),pch=c(16,16,-1),col=c("brown1","blue","red"),bty="n",cex=1)
mtext("Model intercomparison",outer=T, cex=1.4)

########################## END Section Three ############################
```