# HW3: Qualifying Wine

Jessica Jones
Feb. 25, 2022

**Abstract**

In this task, we are to train and test three different qualifiers on a red wine dataset. We first identified features that may influence wine quality more through correlation matrices. We then compared linear, gaussian RBF, and Laplacian kernel estimators on how well they qualified a new batch of wine. We extrapolated MSE, $R^2$, and performance accuracy for each estimator and tuned hyperparameters to improve performance.

## 1. Introduction and Overview

We are given datasets related to red wine variants of the Portuguese "Vinho Verde" wine. Only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.). We are tasked with classifying and qualifying these wines based on these chemical components. The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones).

We first perform standardization to normalize the data between zero mean and one standard deviation. Then, we created a correlation matrix to help visualize the dataset—this gave us more insight into which features might bias our models. We performed Linear and Kernel Ridge Regression (RBF and LaPlacian) estimation by projecting our data onto the training models and assessed the performance of each classifier by mean squared error (MSE), $R^2$, and accuracy. Finally, we tuned each hyperparameter funneled into each estimator and tested the performance on a new batch of wine, which gave a prediction of quality.

## 2. Theoretical Background

### 2.1. Linear Regression

Linear Regression fits a linear model with coefficients $w = (w_1, \ldots, w_p)$ to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation. Mathematically it solves a problem of the form:

$$\min_w \left|\left| X\,w - y \right|\right|_2^2 \tag{1}$$

If $\hat{y}$ is the predicted value, then:

$$\hat{y}(w, x) = w_0 + w_1 x_1 + \cdots + w_p x_p \tag{2}$$

with $w_0$ being the intercept of the regression.

### 2.2. Kernels and Kernel Ridge Regression (KRR)

A “kernel” is usually used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem. The kernel of a function refers to the set of points that the function sends to 0. The kernel function is what is applied on each data instance to map the original non-linear observations into a higher-dimensional space in which they become separable.

KRR combines ridge regression (linear least squares with l2-norm regularization) with the kernel trick. For non-linear kernels, this corresponds to a non-linear function in the original space.

### 2.3. Linear Kernels

If $(x, y)$ are column vectors, then their linear kernel is:

$$k(x, y) = x^{\mathsf{T}} y \tag{3}$$

### 2.4. Gaussian Radial Basis Function (RBF) Kernel

The Gaussian radial basis function (RBF) kernel is a measure of similarity between two vectors. The induced feature space is the same regardless of the sample of data, so the penalty is a penalty on the function of the model, rather than on its parameterization. It generalizes via the equation:

$$k(x, y) = \exp(-\gamma |x - y|^2) \tag{4}$$

where x and y are the input vectors. If $\gamma = \sigma^{-2}$ , the kernel is known as the Gaussian kernel of variance $\sigma^2$.

### 2.5. Laplacian Kernel

The Laplacian kernel is a variant on the radial basis function kernel defined as:

$$k(x, y) = \exp(-\gamma |x - y|_1) \tag{5}$$

Where x and y are the input vectors and $\|x-y\|_1$ is the Manhattan distance between the input vectors.

### 2.6. Mean Squared Error (MSE)

MSE is a risk metric corresponding to the expected value of the squared (quadratic) error or loss. If $\hat{y}_i$ is the predicted value of the i-th sample, and $y_i$ is the corresponding true value, then the mean squared error (MSE) estimated over $n_{\text{samples}}$ is defined as:

$$\text{MSE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}} - 1} (y_i - \hat{y}_i)^2 \tag{6}$$

### 2.7. Cross Validation (CV)

CV is any of various similar model validation techniques for assessing how the results of a statistical analysis will generalize to an independent data set. CV is a resampling method that uses different portions of the data to test and train a model on different iterations. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

## 3. Algorithm Implementation and Development

### 3.1. Preprocessing

Our first step is to load the wine training and test datasets and outputs. This was done via pandas.read_csv() function and indexed the appropriate training and test datasets. This gave matrices:

$$X_{train} \in \mathbb{R}^{1115\,X\,11},\ Y_{train} \in \mathbb{R}^{1115} \tag{7}$$

to denote the matrix of training features and the vector of training outputs *respectively*. Likewise:

$$X_{test} \in \mathbb{R}^{479\,X\,11}, \qquad Y_{test} \in \mathbb{R}^{479} \tag{8}$$

denote the matrices of test features and the vector of testing outputs *respectively*.
We then standardized features by removing the mean and scaling to unit variance, using

$$z = (x - u)/s \tag{9}$$

Where u is mean 0 and s is the standard deviation of training samples, which is 1. This was done via StandardScalar() function in sklearn.

### 3.2. Correlation Matrix

I wanted to identify which physical properties are the ones that affect quality the most. I used the correlation matrix, specifically the corr() function, on the training dataset to visualize relationships. This outputted features that positively impacted wine quality and displayed negative correlations as well seen in Figure 1. Some physical properties, like **volatile acidity**, affected wine quality negatively, while **alcohol** positively affect the wine quality.

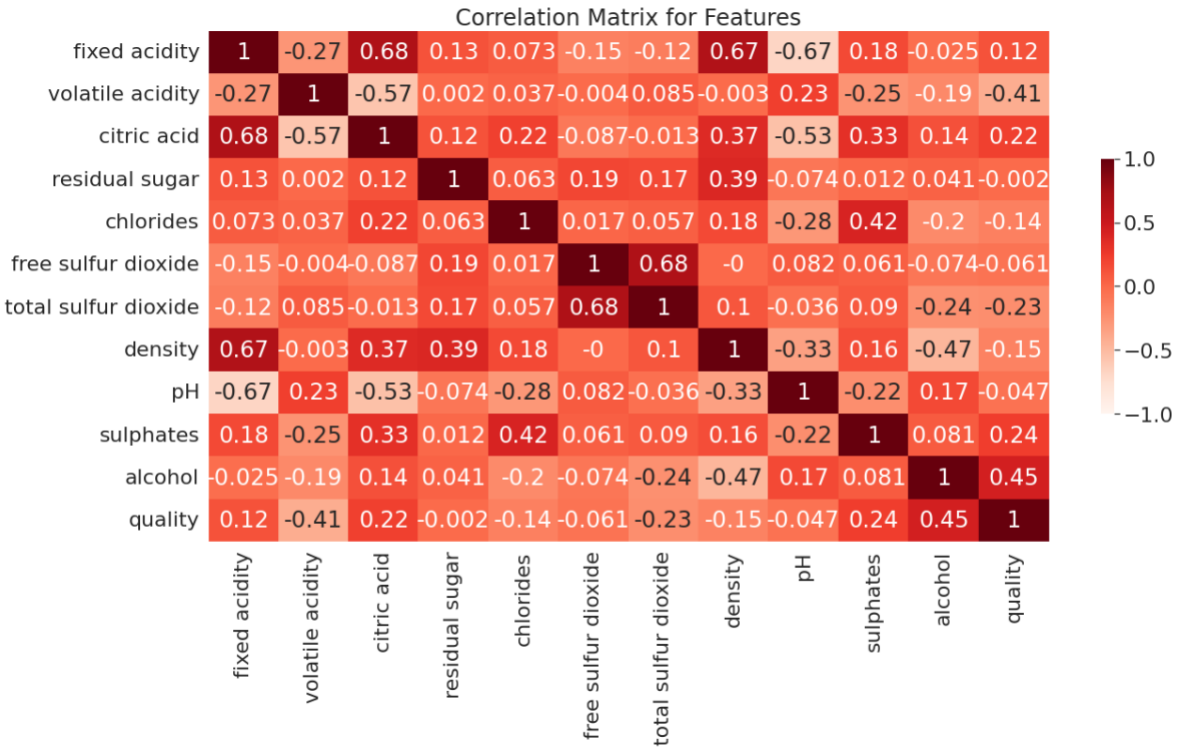


Figure 1: Correlation Matrix of Wine Features

### 3.3. Model Fitting

Curve fitting happens where one has a parametrized model function meant to explain some phenomena and wants to adjust the numerical values for the model to match some data most closely. The classifier estimator instance is first fitted to the model; that is, it must learn from the model. This is done by passing our training set to the .fit() method in sklearn.

### 3.4. 10-Fold Cross Validation

For 10-fold CV, the training set is split into 10 smaller sets. The following procedure is followed for each of the k "folds", which is 10:

- A model is trained using k−1 of the folds as training data
- the resulting model is validated on the remaining part of the data (i.e., it is used as a test set to compute a performance measure such as accuracy).

The performance measure reported by k-fold cross-validation is then the average of the values computed in the loop.

### 3.5. Parameter Tuning

Hyper-parameters are parameters that are not directly learnt within estimators. In scikit-learn they are passed as arguments to the constructor of the estimator classes. We were recommended to search the hyper-parameter space for the best cross validation score. For the Linear Regression estimator, specifying the value of the cv attribute will trigger the use of cross-validation with GridSearchCV() from sklearn, for example cv=10 for 10-fold cross-validation. In order to optimize our remaining models, RBF and LaPlacian, we conducted a Grid Search using GridSearchCV(). This function exhaustively considers all parameter combinations, and consists of:

- an estimator (regressor or classifier);
- a parameter space;
- a method for searching or sampling candidates.
- a cross-validation scheme; and
- a score function.

| **Results from 10-Fold Cross Validations Pre- & Post-Tuning by Model** | | | | | | |
|---|---|---|---|---|---|---|
| | **Linear** | **Linear_tuned** | **RBF** | **RBF_tuned** | **Laplace** | **Laplace_tuned** |
| **Efficiency (ms)** | **n/a** | **153** | **n/a** | **86** | **n/a** | **87** |
| **score** | **n/a** | **0.34** | **n/a** | **0.43** | **n/a** | **0.36** |
| $\alpha$ | **1** | **10** | **1** | **0.1** | **1** | **0.1** |
| $\gamma$ | **n/a** | **n/a** | **n/a** | **0.06** | **n/a** | **0.06** |
| **MSE** | **0.4126** | **0.4124** | **0.7011** | **0.4351** | **0.4013** | **0.36** |
| $R^2$ | **0.34** | **0.3405** | **0.128** | **0.3027** | **0.3626** | **0.4277** |
| **accuracy** | **91.2345** | **91.2345** | **89.4487** | **91.3774** | **91.4914** | **92.0868** |

Table 1: Results from 10-fold Cross Validations

| Quality prediction outputs by model | | | |
|---|---|---|---|
| **Wine** | **Linear** | **RBF** | **Laplacian** |
| **1** | **6.01** | **6.14** | **6.21** |
| **2** | **5.30** | **4.15** | **5.22** |
| **3** | **6.56** | **5.17** | **5.56** |
| **4** | **6.08** | **5.28** | **5.69** |
| **5** | **5.96** | **5.5** | **5.65** |

Table 2: Quality Predictions by Each New Model

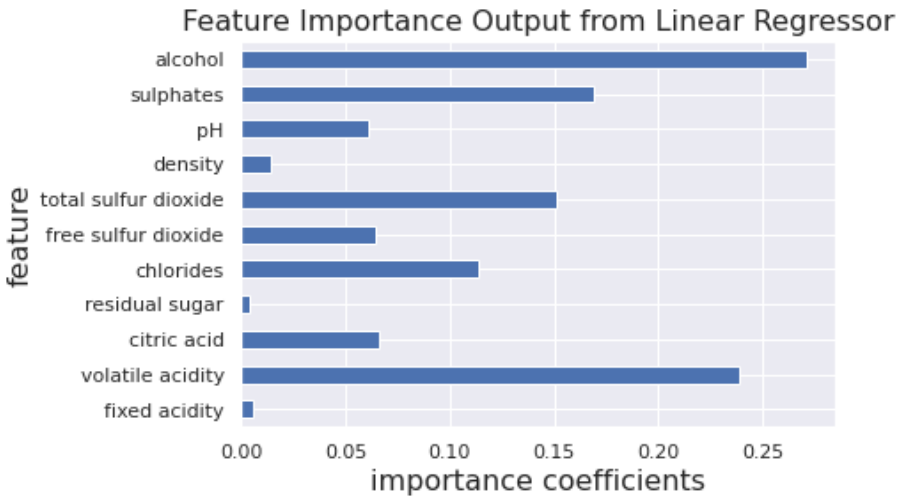


Figure 2: Features Important for Linear Regressor

The results pre- and post- parameter tuning are given in Table 1. **Alpha (α)** is the parameter which balances the amount of emphasis given to minimizing RSS vs minimizing sum of square of coefficients. α can take various values: α = 0: The objective becomes same as simple linear regression. The **gamma (*γ*)** parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close', thus giving a decision region. This approach can be computationally expensive, so I ran GridSearchCV() multiple times to narrow down the ranges of alpha and gamma values to use during tuning.

# 4. Computational Results

## 4.1. Prediction on a New Batch of Wine

When running each new model on a new batch of wine, the results are given in Table 2. The original data's output scores range from 3 to 8 in quality, with alcohol levels and volatile acidity heavily influencing quality. Given this, each model predicted quality to fit within that range, from 4 being low and 6 being high quality. To assess feature importance, I pulled feature coefficients from the new linear model ran on the new batch of wine. When done, the importance coefficients related back to our original correlation matrix on the original data, with alcohol content and volatile acidity biasing model prediction, seen in Figure 2.

# 5. Summary and Conclusions

Given the resulting accuracies from each model, linear and nonlinear models perform similarly when qualifying batches of wine. Each model is biased based on the features imported into each classifier, and persisted after tunning.

# 6. Acknowledgements

**References**

Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553. ISSN: 0167-9236.

Pedregosa et al., Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.

Zhang, J. and Marszalek, M. and Lazebnik, S. and Schmid, C. Local features and kernels for classification of texture and object categories: A comprehensive study International Journal of Computer Vision 2007 https://research.microsoft.com/en-us/um/people/manik/projects/trade-off/papers/ZhangIJCV06.pdf