Using Data Mining Techniques to Predict Product Quality from Physicochemical Data

A. Nachev[1], **M. Hogan**[1]
[1]Business Information Systems, Cairnes Business School, NUI, Galway, Ireland

Abstract - *Product quality certification is sometimes expensive and time consuming, particularly if it requires assessment made by human experts. This study explores experimentally how data mining techniques can facilitate that process. We use a dataset of physicochemical characteristics of red and white wine samples, available from laboratory tests, in order to build models that predict wine quality. Four data mining techniques are used: multilayer perceptrons, cascade-correlation neural networks, general regression neural networks, and support vector machines. We study how hyper-parameters of the models influence their predictive abilities and how reduction of dimensionality affects their performance. We also compare the models by the metrics prediction accuracy, mean absolute deviation, and area over the regression error characteristics curve.*

Keywords: data mining, neural networks, cascade-correlation neural networks, general regression neural networks, support vector machines.

1 Introduction

Today, with improvement of technologies, industries become more efficient and the production processes become quicker. In many cases, however, the human expertise is still essential for the product quality assurance process. With increase of demand for goods, quality certification becomes an expensive step in the production process.

The aim of this study is to explore the potential of four predictive techniques: neural networks (NN) a.k.a. multilayer perceptrons (MLP), cascade-correlation neural networks (CCNN), general regression neural network (GRNN), and support vector machines (SVM), to facilitate the quality certification of a product, based on available product characteristics. This would allow automating the process and minimizing usage of human expertise.

Here we focus on the wine quality prediction using data from both physicochemical laboratory tests and sensory tests. Wine is usually characterised by density, alcohol or various acids, which can be obtained by lab tests, while sensory tests are done by human experts. Wine classification is not an easy task as the relationships between physicochemical analysis and sensory tests analysis are complex and not well understood [12].

Predicting wine quality by data mining techniques is still in an early stage, but there are some promising results in the domain. Sun et al. [16] used NNs fed with 15 input variables used to predict six geographic wine origins. The data included 170 samples. Vlassides et al. [19] used NNs to classify three sensory attributes (e.g. sweetness) of Californian wine, based on grape maturity levels and chemical analysis. Moreno et al. [13] used probabilistic neural networks (PNN) to discriminate 54 wine samples into two red wine classes. Yu et al. [20] used spectral measurements from 147 bottles of rice wine to predict 3 categories of wine. Fei et al., [8] utilized least squares support vector machines on physicochemical data of red wine samples. These chemometrics were obtained through the use of visible and near infrared (Vis/NIR) transmittance spectroscopy. Beltran et al. [5] utilize SVM in addition to, and in comparison with, radial basis function neural networks (RBFNN) and linear discriminant analysis (LDA), in the classification of Chilean wine. The analyses are carried out on data derived from wine aroma chromatograms of three different Chilean wine varieties. Bapna and Gangopadhyay [4] and Cortez et al. [6] compared several data mining techniques for classification of wine.

In this paper, we estimate performance of NN, CCNN, GRNN, and SVM in predicting red and wine quality based on 11 physicochemical characteristics and explore how model hyper-parameters influence their ability to discriminate between quality classes.

The paper is organized as follows: Section 2 provides an overview of the multilayer perceptrons, cascade-correlation neural networks, general regression neural network, and support vector machines used build a predictive models; Section 3 discusses the dataset used in the study, its features, preprocessing steps, and feature selection; Section 4 presents and discuses the experimental results; and Section 5 gives the conclusions.

2 Data Mining Models

We adopt four predictive techniques: the most common NN type - MLP, cascade-correlation neural networks, general regression neural networks, and support vector machines. This section outlines briefly each of those.

2.1 Multilayer Perceptrons

An MLP is a feedforward NN model that maps sets of input data onto a set of appropriate output, either values or class labels. It uses three layers of neurons, called nodes (see Figure 1), with nonlinear activation functions that can

distinguish non-linearly separable data, or separable by a hyperplane. Nodes of two adjacent layers are fully connected by weighted links represented by matrices *IW*, *LW*, and bias vectors *b*. The two activation functions are both sigmoids:

$$f_H(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \quad f_L(x) = \frac{1}{1 + e^{-\beta x}} \ , \tag{1}$$

where f_H is a hyperbolic tangent which ranges from -1 to 1; f_L is a log-sigmoid function, equivalent in shape, but ranges from 0 to 1. Here x is the weighted sum of the inputs. Finding an optimal size of the hidden layer is a general problem with all MLP. We used the heuristic:

$$n_h = \frac{n_s}{\alpha(n_i + n_o)} \tag{2}$$

where n_h is the size of the hidden layer; n_s is the number of training samples; n_i and n_o are the size of the input and output layers respectively; $\alpha \in [5, 10]$ is a scaling factor, smaller of noisy data and larger for relatively less noisy data. The network was trained by Levenberg-Marquardt (LM) backpropagation (BP) algorithm [9]. LM is a second-order nonlinear optimization technique that uses an approximation to the Hessian matrix. It was chosen from the various BP training algorithms as it trains a moderate size NN 10 to 100 times faster than the usual gradient descent backpropagation method and produces better results.

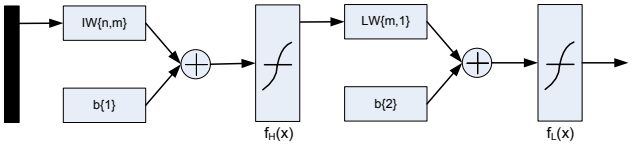

Figure 1. Architecture of an MLP neural network with one hidden layer.

2.2 Cascade-Correlation Neural Networks

CCNN are self-growing neural networks similar to MLP, but they don't have fixed size or topology [2]. The CCNN have three layers: input, hidden, and output, similarly to MLP. The output layer consists of a single node if the network is used for regression problems, or contains several nodes for classification problems, one per class label. In contrast to MPL, the CCNN start training without hidden layer - input nodes are fully connected to the output nodes with adjustable weights. During the training, the network adds new hidden nodes. It creates a multi-layer structure called a 'cascade', because the output from all input and hidden nodes existing already in the network, feed new nodes. In the beginning,every input is connected to every output neuron by a connection with an adjustable weight. The network adds new hidden nodes one by one (Figure 2) until the residual error gets acceptably small or the user interrupts this process.

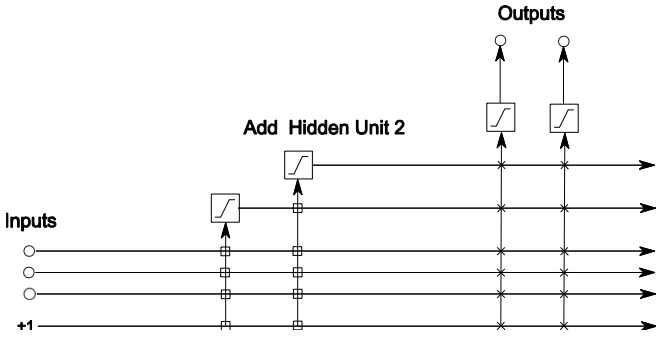

Figure 2. Cascade architecture after adding two hidden nodes (adapted from [2]). The vertical lines sum all incoming activation. Boxed connections are frozen, 'x' connections are trained repeatedly.

To install a new hidden neuron, instead of a single candidate, the system uses a pool of trainable candidate nodes (usually four to eight), each with a different set of randomly selected initial weights. All candidates receive the same input signals and see the same residual error for each training pattern, but because they are not installed yet and do not interact with one another or affect the active neural network during training, all of these candidate units are trained in parallel; when no further progress is being made in training, the network installs the candidate whose score is the best (minimises the residual error). The use of this pool of candidates is beneficial in two ways: it greatly reduces the chance that a useless unit will be permanently installed, and it speeds up the training because many parts of weight-space can be explored simultaneously.

While the candidate weights are being trained, none of the weights in the active network are changed. Once a new hidden node has been added to the network, its input-side weights (boxed connections in Figure 2) are frozen; the output-side connections ('x' connections) continue to be adjustable. The learning algorithm modifies the weights attempting to minimize the residual error of the network. Each new neuron becomes a permanent feature-detector in the network, available for producing outputs or for creating other, more complex feature detectors.

Among advantages of CCNN can be mentioned self-organizing architecture, quick learning, applicable to large datasets, obtaining good results with little or no adjustment parameters and less chance to get trapped in local minima, compared to the MLP. They have, however, a significant potential for overfitting the training data, which results in a very good accuracy on the training dataset but not always good accuracy on new, unseen during the training data.

2.3 General Regression Neural Networks

The GRNN are a kind of radial basis function (RBF) NN proposed by Specht [15]. They are a powerful regression tool, which features simple structure and implementation and fast training. A GRNN consists of four layers: input, hidden, summation, and output (Figure 3). The function of the input layer is to pass the input values x_i to the hidden layer.

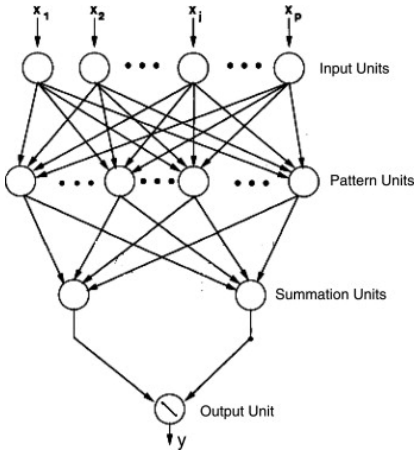

Figure 3. GRNN architecture: feed-forward NN with input, hidden, summation, and output layers.

The hidden layer consists of all training patterns X_i. When an unknown pattern X is presented to the network, the squared distance $D_i^2 = (X - X_i)^T (X - X_i)$ between the X and each X_i is calculated and passed to the kernel function. The summation layer has two nodes (units), A and B, where A computes the summation function, which is numerator of (3), and B computes the denominator.

$$Y(X) = \frac{\sum_{i=1}^{n} Y_i \exp(\frac{-D_i^2}{2\sigma^2})}{\sum_{i=1}^{n} \exp(\frac{-D_i^2}{2\sigma^2})}, \quad (3)$$

where σ is the width of the kernel. The output node computes A/B, which is Y.

2.4 Support Vector Machines

SVM, originally introduced by Vapnik in 1990s [17], provide a new approach to the problem of pattern recognition with clear connections to the underlying statistical learning theory. They differ radically from comparable approaches such as NN because SVM training always finds a global minimum in contrast to NN [18].

SVMs are supervised learning methods used for classification and regression. Training data is a set of points of the form

$$D = \{(x_i, c_i) \mid x_i \in \Re^p,\ c_i \in \{-1,1\}\}_{i=1}^{n} \quad (4)$$

where the c_i is either 1 or −1, indicating the class to which the point x_i belongs. Each data point x_i is a p-dimensional real vector. During training a linear SVM constructs a $p-1$-dimensional hyperplane that separates the points into two classes (Figure 4). Any hyperplane can be represented by $w \cdot x - b = 0$ where w is a normal vector and $\cdot$ denotes dot product. Among all possible hyperplanes that might classify the data, SVM selects one with maximal distance (margin) to the nearest data points (support vectors).

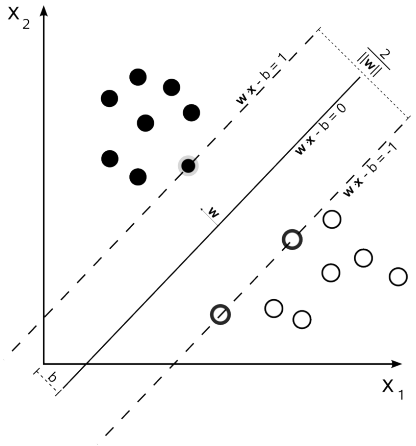

Figure 4. Maximum–margin hyperplane for a SVM trained with samples from two classes. Samples on the margin are support vectors.

When the classes are not linearly separable (there is no hyperplane that can split the two classes), a variant of SVM, called soft-margin SVM, chooses a hyperplane that splits the points as cleanly as possible, while still maximizing the distance to the nearest cleanly split examples. The method introduces slack variables, ξ_i, which measure the degree of misclassification of the datum x_i. Soft-margin SVM penalizes misclassification errors and employs a parameter (the soft-margin constant C) to control the cost of misclassification. Training a linear SVM classier solves the constrained optimization problem (5).

$$\min_{w,b,\xi_k} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \quad (5)$$
$$s.t. \quad w \cdot x_i + b \geq 1 - \xi_i$$

In dual form the optimization problem can be represented by:

$$\min_{\alpha_i} \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i \cdot x_j - \sum_{i=1}^{n} \alpha_i \quad (6)$$
$$s.t. \quad 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{n} \alpha_i c_i = 0$$

The resulting decision function $f(x) = w \cdot x + b$ has weight vector $w = \sum_{k=1}^{n} \alpha_k y_k x_k$. Data points x_i for which $\alpha_i > 0$ are called support vectors, since they uniquely define the maximum margin hyperplane. Maximizing the margin allows one to minimize bounds on generalization error.

If every dot product is replaced by a non-linear kernel function, it transforms the feature space into higher-dimensional, thus though the classifier is a hyperplane in the high-dimensional feature space it may be non-linear in the original input space. The resulting classifier fits the

maximum-margin hyperplane in the transformed feature space. Some common kernels include:

- Polynomial kernel $K(\vec{x}_i, \vec{x}_j) = (\gamma \vec{x}_i^T \vec{x}_j + r)^d$
- RBF kernel $K(\vec{x}_i, \vec{x}_j) = \exp(\gamma \left\| \vec{x}_i - \vec{x}_j \right\|^2)$
- Sigmoid kernel $K(\vec{x}_i, \vec{x}_j) = \tanh(\gamma \vec{x}_i^T \vec{x}_j + r)$

A non-linear SVM is largely characterized by the choice of its kernel, and SVMs thus link the problems they are designed for with a large body of existing work on kernel based methods. Once the kernel is fixed, SVM classifiers have few user-chosen parameters. The best choice of kernel for a given problem is still a research issue. Because the size of the margin does not depend on the data dimension, SVM are robust with respect to data with high input dimension. However, SVM are sensitive to the presence of outliers, due to the regularization term for penalizing misclassification (which depends on the choice of C). The SVM algorithm requires $O(n^2)$ storage and $O(n^3)$ to learn.

3 Dataset and Preprocessing

The data used in this study represent wine sample collection of Vinho Verde wines, white and red (CVRVV, 2008), which consists of two distinct sets made up of 4898 white and 1599 red samples. Each instance consists of 12 physiochemical variables: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, and a quality rating. The quality rating is based on a sensory taste test carried out by at least three sommeliers and scaled in 11 quality classes from 0 - very bad to 10 - very excellent. A summary the datasets is presented in Table 1.

Table 1: The physicochemical data statistics per wine type

Attributes	*Red Wine*			*White wine*		
	Min	*Max*	*Mean*	*Min*	*Max*	*Mean*
Fixed acidity	4.6	15.9	8.3	3.8	14.2	6.9
Volatile acidity	0.1	1.6	0.5	0.1	1.1	0.3
Citric acid	0.0	1.0	0.3	0.0	1.7	0.3
Residual sugar	0.9	15.5	2.5	0.6	65.8	6.4
Chlorides	0.01	0.61	0.08	0.01	0.35	0.05
Free sulpher dioxide	1	72	14	2	289	35
Total sulfur dioxide	6	289	46	9	440	138
Density	0.990	1.004	0.996	0.987	1.039	0.994
pH	2.7	4.0	3.3	2.7	3.8	3.1
Sulphates	0.3	2.0	0.7	0.2	1.1	0.5
Alcohol	8.4	14.9	10.4	8.0	14.2	10.4

Using the data in their original format for building models is inappropriate due to some deficiencies. A specific problem is the large amplitude of the variable values due to the different nature and different units of measurements of those values, e.g. sulfur dioxide (1 – 72) vs. sulfates (0.3 – 2). Such an inconsistency could affect the predictive abilities of the models by making some variables more 'influential' than others. Moreover, some models require inputs within the unit hypercube, i.e. between 0 and 1. A natural approach of meeting that requirement could be a linear transformation that divides all input values by the dataset maximum, however mostly of the input values will fall very close to zero, and the model would perform poorly. A better approach is to process each data variable (data column) separately. We did so by using the transformation

$$x_i^{new} = \frac{x_i^{old} - \min_i}{\max_i - \min_i}, \quad (7)$$

which scales down the variables within the unit hypercube.

Another problem with utilizing the original data without preprocessing is that using all features of a dataset does not always lead to best or even satisfactory results. This is due to the fact that too much information used for both tratning and testing can lead to overfitting or overtraining. We explored how presence or absence of variables presented to the model for training and testing affects the performance.

Variable selection, or reduction of dimensionality, is a technique commonly used in machine learning for building robust learning models. Removing most irrelevant and redundant features from the data usually helps to alleviate the effect of the curse of dimensionality and to enhance the generalization capability of the model, yet to speed up the learning process and to improve the model interpretability. The variable selection also helps to acquire better understanding about data and how they are related with each other. Dimensionality reduction is considered as an application-specific problem, which is not backed by a universal theory. The exhaustive search approach that considers all possible subsets is the best strategy applicable for datasets with small cardinality, but impractical for large number of features, as our case is.

There are two distinct groupings of variable selection algorithms, specifically wrapper methods and filter methods. The wrapper methods employ the feature subset selection algorithm in unison with an induction algorithm. The selection algorithm proceeds to unearth a favorable subset of data whilst using this induction algorithm to evaluate proposed subsets. The filter methods use a preprocessing step and autonomously select variables independent of the induction algorithm. There are a number of algorithms that fall under the umbrella of the filter approach, such as the relief algorithm, which assigns a weighting of relevance to each feature, that is, the relevance of the selected variable to the target output; and the decision tree algorithm, which is used to select feature subsets for the nearest neighbor algorithm [11].

Rueda et al. [14] highlight a particular strength possessed by wrapper algorithms. The authors state that if variables are highly correlated with the response, the filter algorithm would typically include them, even if they diminished the overall algorithm performance. While in the wrapper approach, the induction algorithm may discover these diminishing effects, and exclude them [3].

4 Empirical Results

Using the data described above, we built and tested a number of predictive models, based on the four techniques - SVM, CCNN, GRNN, and NN.

In order to estimate the models performance we used the following metrics:

- *Prediction accuracy* ACC_t at certain error tolerance values t = 0.25, 0.5, 1, and 2.
- *Mean Absolute Deviation* (MAD), which is a robust performance measure of the model variability [1]

$$MAD = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y}_i\right|, \quad (8)$$

where y_i and $\hat{y}_i$ are the class label and predicted value, respectively.

- *Area over the regression error characteristic curve*. The regression error characteristic (REC) curve plots the error tolerances along the horizontal axis versus the prediction accuracy on the vertical. The area over the REC curve (AOC) is a scalar value that estimates the overall model performance regardless of the error tolerance values applied to each model instance. The lower the AOC is, the better it performs.

We tested the models with a number of hyper-parameters in order to find their optimal values and ensure maximal performance. Avoiding bias in training and testing, we applied 5-fold cross-validation. The dataset was divided on five subsets, each of which is 20%. The overall performance estimation metrics were calculated using each of those 20% for testing after training the model on the remaining 80% of data.

In order to estimate the influence of reduction of data dimensionality in the model performance, we applied wrapper and filter attribute evaluator methods outlined above. These methods combined differing search techniques, which resulted in combinations of proposed variable subsets. Models were tested with different combinations of variables and the results were compared by the aforementioned metrics. The results obtained were different for the red wine and the white wine datasets.

In the red wine case, the best results were obtained by the chi-squared attribute evaluation technique [10], which calculates chi-squared worth of each attribute with respect to the class. Results are summarized in Table 2 The experiments showed that chi-squared worth cut-off point between 169.86 and 145.40 performs best, which resulted in four red wine attributes used in training and testing the models, namely alcohol, volatile acidity, sulphates, and citric acid.

We found that the optimal SVM parameters used to produce a minimal mean squared error (MSE) of the model are: c=1.398; ε =0.746; kernel=polynomial; d=1; γ =0.572; and r=0.530.

Results obtained from the error tolerance study of the models are compared by REC curves in Figure 5. It is the model with the least area over the curve (AOC) that is most accurate, with the point closest to the 100% accurate and zero threshold intersection, indicating the best threshold level of the model. Figure 5 is quantitatively summarised in the Table 3.

Table 2 Chi-squared attribute evaluation for red wine.

Attribute	*Chi-squared worth*	*Percentage importance*
Alcohol	497.7464	29.61
Volatile acidity	354.4793	21.09
Sulphates	252.0535	15.00
Citric acid	169.8607	10.11
Total sulphur dioxide	145.3958	8.65
Density	130.73	7.78
Chlorides	82.6207	4.92
Fixed acidity	48.0288	2.86
pH	0	0.00
Residual sugar	0	0.00
Free sulphur dioxide	0	0.00

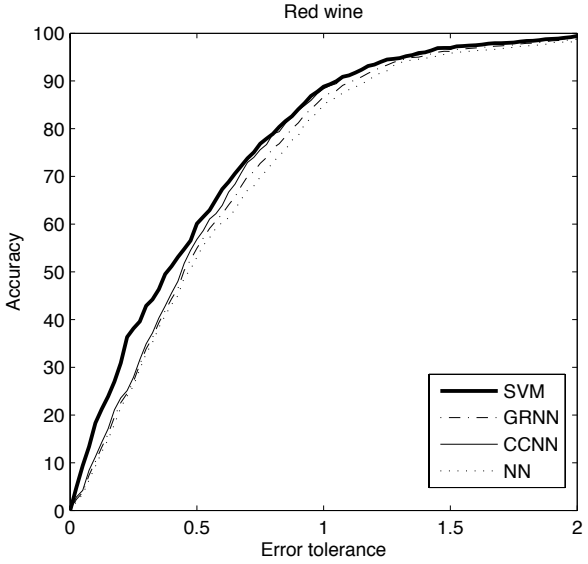

Figure 5. REC curves of red wine test set. SVM - tick solid line; CCNN - thin solid line; GRNN - dash-dot line; NN - dot line.

Table 3 Performance of red wine quality prediction models. Estimation metrics include: accuracy at certain error tolerances (ACCt), mean absolute deviation (MAD), and area above the REC curve (AOC).

	$ACC_{0.25}$	$ACC_{0.5}$	ACC_1	$ACC_{1.5}$	ACC_2	*MAD*	*AOC*
ANN	0.261	0.531	0.850	0.958	0.981	0.592	0.662
CCNN	0.279	0.568	0.884	0.966	0.983	0.548	0.630
GRNN	0.27	0.549	0.867	0.962	0.979	0.577	0.644
SVM	0.381	0.601	0.888	0.969	0.994	0.496	0.506

It should be noted that according to the metrics MAD and AOC, SVM outperform all other models. They show clear advantage in the low error tolerance ranges where direct hits in predictions is important, or one-away hits, where error tolerance less than 0.5 is acceptable. When error tolerance increases and requirement for correct classifications relaxes, CCNN networks become equally good to SVM. Last in performance is the classic feed-forward NN and the second last is GRNN, which is between NN and CCNN.

Similarly, we explored dimensionality reduction in the white wine case. Results showed that best technique for ranking attributes is symmetrical uncertainty ranking, which is one of the most effective entropy-based feature selection approaches. Experimentally we found that alcohol content in white wine bears most importance (26.47%); density ranks second in importance (19.19%); with chlorides following next (14.35%). Total sulfur dioxide, citric acid, free sulfur dioxide and volatile acidity complete the model, all registering close importance percentage between 9.9% and 10.4%. Results are summarized in Table 4.

Table 4 Symmetrical uncertainty attribute evaluation for white wine.

Attribute	*Symmetrical Uncertainty*	*Percentage importance*
alcohol	0.08998	26.46
density	0.06524	19.18
chlorides	0.04878	14.34
total sulphur dioxide	0.03513	10.33
citric acid	0.03468	10.20
free sulphur dioxide	0.03376	9.92
volatile acidity	0.03241	9.53

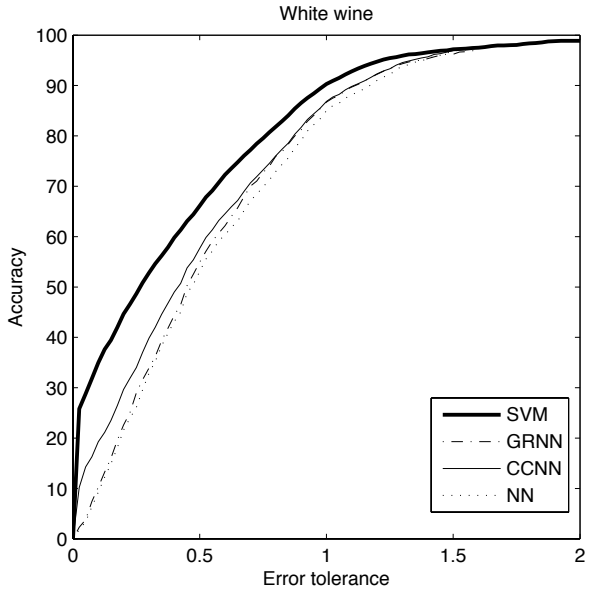

Figure 6. REC curves of red wine test set. SVM - tick solid line; CCNN - thin solid line; GRNN - dash-dot line; NN - dot line.

Findings also show that with white wine, SVM performs best with c=2.438; ε=0.684; kernel=polynomial; d=1; γ =1.266; and r=1.522.

Figure 6 graphically compares the models performance in the terms of REC and Table 5 summarizes the estimation metrics. Similarly to the red wine case, the white wine results show that SVM outperforms the three neural networks, with even higher accuracy in the low error tolerance values, but it also outperforms the other models in higher error tolerance values (between 0.5 and 1.5).

The three neural network models show similar performance with little advantage of CCNN over GRNN and the classic NN. In a relatively large error tolerance (above 1), CCNN and GRNN perform similarly and slightly better than NN.

Table 5 Performance of white wine quality prediction models. Estimation metrics include: accuracy at certain error tolerances (ACCt), mean absolute deviation (MAD), and area above the REC curve (AOC).

	$ACC_{0.25}$	$ACC_{0.5}$	ACC_1	$ACC_{1.5}$	ACC_2	*MAD*	*AOC*
ANN	0.261	0.531	0.850	0.968	0.988	0.594	0.658
CCNN	0.339	0.576	0.868	9.969	0.988	0.514	0.581
GRNN	0.290	0.549	0.867	0.962	0.988	0.589	0.630
SVM	0.486	0.661	0.902	0.971	0.988	0.477	0.566

Finally, it can be summarized that SVM could be a better alternative of prediction models based on neural networks for application areas, like the one explored here. At the same time, certain neural network types, such as CCNN and GRNN can be considered as good candidates for predicting models, both outperforming the classic neural network.

5 Conclusions

Recently, wine industry expands its marketplace, which encourages adoption of advanced technologies in the production process. The quality certification is an important step in that. Traditionally, it is based on sensory tests carried out by human experts. This, however, is not as efficient as needed, because the procedure is time consuming and expensive. Data mining may help in the quality certification by processing physicochemical laboratory test data and building models that predict product quality classes. Various modeling techniques can be applied to solve the task and each of them shows specific performance characteristics.

The goal of this study is to explore how model hyper-parameters of the classic backpropagation neural network, cascade-correlation neural network, general regression neural network, and support vector machine, affect their predictive abilities in solving that task. We used an existing data set of 1599 red wine samples, and 4898 white wine samples, each of which consisting of 11 physicochemical characteristics. In order to quantify the model performance, we used metrics, such as prediction accuracy, mean absolute deviation, and area over the regression error characteristics curve. Our findings show that support vector machine with polynomial kernel outperforms the three neural network

models in all the metrics. The SVM advantage can clearly be seen with small values of error tolerance, that is where predicted quality is required to be very close to the real one. From another hand, the CCNN and GRNN show similar performance with little advantage of the CCNN over GRNN. Last in ranking is the classic NN, which despite its popularity as classification and regression tool, is not the best choice in this application domain. We also tested how various techniques for reduction of dimensionality influence the models performance. Empirically we found that best variable set selection techniques are chi-squared attribute evaluation and symmetrical uncertainty ranking for the red and white wine, respectively.

6 References

[1] J. Bi and K. P. Bennett. Regression error characteristic curves. In Proceedings of the 20th International Conference on Machine Learning, 2003.

[2] Fahlman, S. and Lebiere C. "The Cascade-Correlation Learning Architecture" in D. S. Touretzky (ed.), Advances in Neural Information Processing Systems 2, Morgan Kaufmann, 1990.

[3] Guetlein, M., Frank, E., Hall, M., Karwath, A. "Large Scale Attribute Selection Using Wrappers"; In Proc. IEEE Symposium on CIDM, pp.332-339, 2009.

[4] Bapna, S. and Gangopadhyay, A. A Wavelet-Based Approach to Preserve Privacy for Classification Mining. Decision Sciences, 37, 623-642, 2006.

[5] Beltran, N. H., Duarte-Mermoud, M. A., Soto Vicencio, V. A., Salah, S. A. & Bustos, M. A. Chilean Wine Classification Using Volatile Organic Compounds Data Obtained With a Fast GC Analyzer. IEEE Transactions on Instrumentation and Measurement, 57, 2421-2436, 2008.

[6] Cortez, P., Cerdeira, A., Almeida, F., Matos, T. & Reis, J. Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems, 47, 547-553, 2009.

[7] CVRVV. Portuguese Wine - Vinho Verde. Comissa˜o de Viticultura da Regia˜o dos Vinhos Verdes (CVRVV), http://www.vinhoverde.pt, July 2008.

[8] Fei, L., Li, W. & Yong, H. Application of least squares support vector machines for discrimination of red wine using visible and near infrared spectroscopy. Intelligent System and Knowledge Engineering, ISKE' 08, 2008.

[9] Hagan, M. T., and Menhaj, M. B. Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks, 5, 989-993, 1994.

[10] Hall, M., Frank, E., Holmes, G., Fahringer, B., Reuteman, P. & Witten, I. H. The WEKA Data Mining Software: An Update. SIGKDD *Explorations,* 11, 2009.

[11] Kohavi, R. & John, G. H. 1997. Wrappers for feature subset selection. Artificial Intelligence, 97, 273-324, 1997.

[12] Legin, A., Rudnitskaya, A., Luvova, L., Vlasov, Y.,Natale, C., and D'Amico, A. Evaluation of Italian wine by the electronic tongue: recognition, quantitative analysis, and correlation with human sensory perception. Analytica Chimica Acta, 33-34, 2003.

[13] Moreno, I., D. Gonzalez-Weller, V. Gutierrez, M. Marino, A. Camean, A. Gonzalez, and A. Hardisson. Differentiation of two Canary DO red wines according to their metal content from inductively coupled plasma optical emission spectrometry and graphite furnace atomic absorption spectrometry by using Probabilistic Neural Networks. Talanta, 72:263–268, 2007.

[14] Rueda, I. E. A., Arciniegas, F. A. & Embrechts, M. J. SVM sensitivity analysis: an application to currency crises aftermaths. IEEE Transactions on Systems, Man and Cybernetics, 34, 387-398, 2004.

[15] Specht, D. Enhancement to probabilistic neural networks. In: Proceedings of the International Joint Conference on Neural Networks, vol.1, pp. 761-768, 1991.

[16] Sun, L., K. Danzer, and G. Thiel. Classification of wine samples by means of artificial neural networks and discrimination analytical methods. Fresenius' Journal of Analytical Chemistry, 359:143–149, 1997.

[17] Vapnik, V., The Nature of Statistical Learning Theory. Springer, New York, 1995.

[18] Vapnik, V., Kotz, S., Estimation of Dependences Based on Empirical Data, Springer, New York, 2006.

[19] Vlassides, S., J. Ferrier, and D. Block. Using Historical Data for Bioprocess Optimization: Modeling Wine Characteristics Using Artificial Neural Networks and Archived Process Information. Biotechnology and Bioengineering, 73(1), 2001.

[20] Yu, H. Lin, H. Xu, Y. Ying, B. Li, and X. Pan. Prediction of Enological Parameters and Discrimination of Rice Wine Age Using Least-Squares Support Vector Machines and Near Infrared Spectroscopy. Agricultural and Food Chemistry, 56:307–313, 2008.