Machine Learning approach to discriminate *Saccharomyces cerevisiae* yeast cells using sophisticated image features

Mohamed Tleis[1] and Fons J. Verbeek[1] *

[1]Imaging and Bioinformatics, Leiden Institute of Advanced Computer Science and Mathematics (LIACS), Leiden University, The Netherlands.
http://bio-imaging.liacs.nl/

Summary

In biological research, *Saccharomyces cerevisiae* yeast cells are used to study the behaviour of proteins. This is a time consuming and not completely objective process. Hence, Image analysis platforms are developed to address these problems and to offer analysis per cell as well. The robust segmentation algorithms implemented in such platforms enables us to apply a machine learning approach on the measured cells. Such approach is based on a set of relevant individual cell features extracted from the microscope images of the yeast cells. In this paper, we composed a set of features to represent the intensity and morphology characteristics in a more sophisticated way. These features are based on first and second order histograms and wavelet-based texture measurement. To show the discrimination power of these features, we built a classification model to discriminate between different groups. The building process involved evaluation of a set of classification systems, data sampling techniques, data normalization schemes and attribute selection algorithms. The results show a significant ability to discriminate different cell strains and conditions; subsequently it reveals the benefits of the classification model based on the introduced features. This model is promising in revealing subtle patterns in future high-throughput yeast studies.

1 Introduction

In biological research, Baker's yeast is an excellent standard model organism since many processes that take place in both animal and plant cells occur in a similar way in yeast. Baker's yeast, scientifically known as *Saccharomyces cerevisiae*, can be easily cultured and used to study the behaviour of genes by tagging them with fluorescent proteins.

The study of proteins is time consuming and not completely objective. Hence, automated analysis platforms are developed to address this problem. Such platforms are ideally composed of segmentation, measurement and data analysis modules. The segmentation module segments the cells located in the microscope images, while the measurement module measures various features of the segmented cells. The data analysis part analyzes the measurement and report relevant statistics about the different cell groups [1].

Subtle Patterns are not easy to be extracted from the measurement or basic statistical analysis of the data especially in high throughput screening (HTS) where thousands of images are analysed.

*To whom correspondence should be addressed. Email: m.tleis@liacs.leidenuniv.nl

Moreover, when biologists construct and study different yeast cell strains cultivated in different media, it can be not possible to know if the different cell groups have different characteristics for the same expressed proteins. Thus, the need of an automatic system in order to extract those hidden features.

The idea in this work is to apply a machine learning approach to address the automatic analysis of data. The features (attributes) for training the machine learning system were designed in a way to offer a more sophisticated description of the intensity and morphology characteristics of the cells. Such features includes basic shape descriptors, intensity descriptors, texture measurements, first and second order histogram features, i.e. moment invariants and co-occurrence matrix derived features, in addition to wavelet-based texture measurement.

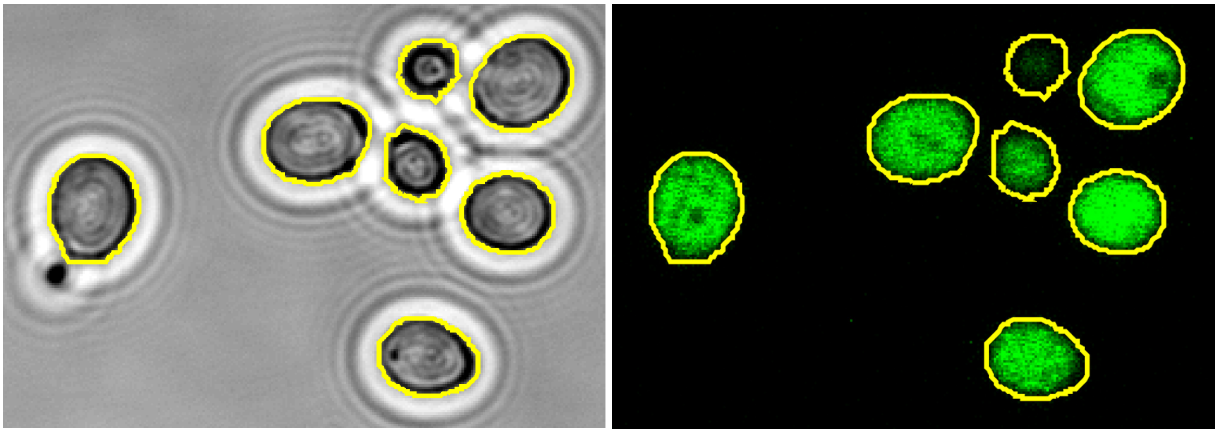
After creating our dataspace with the introduced features. A number of linear and non-linear classifiers were evaluated to select the best model that can classify the instances in the dataspace into cells belonging to a different group, i.e. different strain, or cultured in a different medium. Since our dataspace is imbalanced with a different ratio for different cell groups, we were careful in our evaluation to choose the classifier system that can classify well the majority as well as the minority classes. Therefore, we considered sampling and normalization techniques. Cross validation was considered as well, and feature selection algorithms were evaluated to choose the best algorithm that fits our dataset.

The result shows that many classifiers performed excellently after data preprocessing. Consequently a classification model is built. This Model can significantly discriminate between two different cell groups.

The next section discusses the features used to describe the cells. Subsequently, there is the section about building the classification model, then the results section to illustrate the advantages of machine learning and the designed features in our study. We finalize this paper with a conclusion section and points to consider for future work.

2 Features and Texture Measurement

We have successfully developed a platform to segment *S. cerevisiae* yeast cells and measure a range of features and textures for each individual cell obtained from two channel images acquired by a laser scanning confocal microscope (CLSM). Figure 1 shows a sample two-channels image of *S. cerevisiae* yeast cells. The first channel in Fig. 1a is a bright-field channel depicting yeast cell structures. The second overlaid channel in Fig. 1b is a fluorescent channel of the *BMH1* gene expressed *Bmh1* protein binded with *GFP* protein cultivated in low *NaCl* medium. The yellow contours surrounding the cells in Fig. 1 are the results of our segmentation algorithm [2]. In this study, more sophisticated features and texture measurement were introduced to describe the characteristics of cell morphology and intensity distribution to facilitate the analysis and discrimination of different yeast cells. An image feature is a representation or an attribute of an image describing certain special characteristics of the pattern of interest. While feature extraction is defined as locating those pixels in an image that have some distinctive characteristics [3]. Texture is defined as the visual effect which is produced by spatial distribution of total variations over relatively small areas [4]. Image texture is believed to be a rich source



(a) Bright-field channel depicting structure of *S. cerevisiae* yeast cells

(b) Fluorescent channel depicting *Bmh1-GFP* gene expressed protein

Figure 1: Sample image of segmented *S.cerevisaie* yeast cells in two overlaid channels

of visual information. They are complex visual patterns composed of entities, or sub-patterns, that have characteristic brightness, colour, slope, size, etc. Thus texture can be regarded as a similarity grouping in an image. The most known feature extraction techniques in image analysis are considered in our research. These techniques are classified into histogram based features and the moment invariants derived from them, co-occurrence matrix based features, and multi-scale features [5]. In the following sub-section we start discussing the histogram based features then the moment invariants derived from them. The third sub-section is dedicated to explain the additional features derived from the co-occurrence matrix; and the last sub-section highlights on the multi-scale features and specifically wavelet-based texture features.

2.1 First order histogram based features

Assuming that our microscope image is a function $f(x, y)$ of two space variables x and y, $x = 0, 1, .N-1$ and $y = 0, 1...M-1$. The function $f(x, y)$ can take discrete values $i = 0, 1...L-1$, where L is the total number of intensity levels in the image. The intensity-level histogram is a function showing the number of pixels for each intensity level in the whole image. This function is depicted in Eq. 1, where $\delta(j, i)$ is the Kronecker delta function, depicted in Eq. 2.

$$h(i) = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} \delta(f(x, y), i), \tag{1}$$

$$\delta(j, i) = \begin{cases} 1, j = i \\ 0, j \neq i \end{cases} \tag{2}$$

The histogram of intensity levels is obviously a concise and simple summary of the statistical information contained in the image. Calculation of the grey-level histogram involves single pixel. Thus the histogram contain the first-order statistical information about the image, i.e. the region of interest (RoI) of cell objects. Different useful image features are worked out from

the histogram to quantitatively describe the first-order statistical properties of the cells. In this paper, we considered many basic shape descriptors based on the first order histogram, and the most relevant texture features among those originally proposed by Haralick et al. [6, 7]. A list of those features are listed in Table 1.

Another way to characterize the texture is by deriving moment invariants from the first order statistical information in the histogram [8]. The following sub-section discusses these moment invariant features.

2.2 Moment invariant features

Recognition of visual patterns independent of position, size, and orientation in the visual field has been a goal of much recent research. To achieve maximum utility and flexibility, it would be useful if the extraction technique is insensitive to variations in shape and provide improved performance with repeated trials. This property is known in moment invariant techniques. An image moment is defined as a certain particular weighted average, i.e. moment, of the pixel intensities in an image. Traditionally, moment invariants are computed based on the information provided by both the shape boundary and its interior region. Image moments are useful after segmentation to describe cell characteristics that uniquely describe the shape of that cell. Low order moments are used to derive simple properties including area, total intensity, centroid, skewness, kurtosis and information about the cell's orientation. Moment invariant values are invariant to translation, scale and rotation of the cell [12]. Although wavelet transform (discussed in section 2.4) is scale invariant, it is not in all cases translation or rotation invariant [13], this is an additional advantage of moment invariants in our measurements. Moment Invariants have been frequently used as features for image processing, remote sensing, shape recognition and classification. They showed to be fairly reliable at distinguishing certain classes of topographic objects [12] as well as in many other applications [5]. In yeast studies, the first and second moment invariants were the top predictors to classify virulent from non virulent cells [14].

The set of seven moment invariants proposed by Hu are widely known, and hence we adopted them in our study [15, 7]. We define Hu's set as in Eq. 9, where Φ_1 and Φ_2 are invariants based on second order moments, while Φ_3 ... Φ_7 are invariants based on third order moment.

$$hu = \{\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5, \Phi_6, \Phi_7\}. \tag{9}$$

The effectiveness of moment invariants will increase when fused with the results of other techniques [12]. In this research we fuse them with second order statistical texture measurements obtained from the co-occurrence matrix and wavelet-based texture measurements to get the best from these approaches in the classification step required to discriminate between various cell conditions. The co-occurrence matrix based features and the wavelet-based texture features are discussed in the following sub-sections.

Table 1: Features based on first order histogram

Features	Description
Size	The number of pixels occupied by the cell.
Total Intensity	Sum of the intensity values of the pixels occupied by the cell.
Intensity Standard Deviation	The standard deviation from the mean (intensity/pixel) of the intensity values at each pixel.
Perimeter	The perimeter representation method used to estimate the perimeter of the cell is that of Vossepoel and Smeulders [9].
Circularity	The circularity of detected shapes [10]. $$Circularity = \frac{4\pi Size}{perimeter^2}. \quad (3)$$
Vacuole Size	If the fluorescent protein is expressed in the cytoplasm and nucleus, but not in the vacuole, the size of the central vacuole can be estimated. This is done by using a vacuole filter algorithm that looks in the fluorescent images for a region, inside the RoI (region of Interest) representing every cell, that forms the largest connected region with the lowest intensity values.
Membrane Features	Different features can be measured in the membrane pixels surrounding the cell border. Such features include size, total Intensity, Intensity standard deviation.
Variance	The variance (μ_2 or σ^2) is a measure of intensity contrast and can be computed from the second statistical moment. $$\mu_2(z) = \sum_{i=0}^{L-1}(z_i - m)^2.P(z_i) \quad (4)$$ where z_i is the intensity value of the histogram at location *i*, *m* the mean intensity value, *L* the total number of intensity levels (histogram range), and *P(z_i)* is the corresponding histogram with i between 0 and L-1. [11])
Relative Smoothness	The variance (σ^2) is used to establish the descriptor of relative smoothness (R): $$R(z) = 1 - \frac{1}{1 + \sigma^2(z)} \quad (5)$$ This measure is zero for areas of constant intensities where the variance is zero there, and it approaches 1 for large values of the variance [11].
Skewness	The skewness (μ_3) of the intensity histogram which is the third statistical moment. $$\mu_3(z) = \sum_{i=0}^{L-1}(z_i - m)^3.P(z_i) \quad (6)$$ A negative skewness means that most of the pixel values are high and thus concentrated at the right side of the histogram. A positive skewness means that most of the pixel values are low and thus concentrated at the left side of the histogram. [11])
Uniformity	The uniformity (U) has a maximum value for a cell image in which all intensity levels are equal [11]. $$U(z) = \sum_{i=0}^{L-1} P^2(z_i) \quad (7)$$
Entropy	The Entropy (e), which is a measure of variability, is zero for constant images [11]. $$e(z) = -\sum_{i=0}^{L-1} P(z_i).log_2 P(z_i) \quad (8)$$

2.3 Co-occurrence matrix based features

The simplicity of texture attributes can not completely characterize texture of the cells. Studies state that similar textures agree in their second-order statistics [5] and hence textures can be discriminated if they differ in their second-order statistics. Therefore one of the major statistical methods used in texture analysis is the one based on the definition of the joint probability distribution of pairs of pixels. Methods based on second-order statistics, i.e. statistics given by pairs of pixels, have been shown to achieve good discrimination rates in texture classification [16], and considered to be important in automated image analysis [17]. The second-order statistical features for texture analysis are derived from the co-occurrence matrix [6]. They were demonstrated to feature a potential for effective texture discrimination in biomedical images as well [18]. The second-order histogram is defined as the co-occurrence matrix $h_{d\theta}(i,j)$. When divided by the total number of neighbouring pixels $R(d,\theta)$ in the image, this matrix becomes the estimate of the joint probability $p_{d\theta}(i,j)$ of two pixels, a distance d apart along a given direction θ having particular (co-occurring) values i and j [5]. Formally, for image $f(x,y)$ with a set of L discrete intensity levels, the matrix $h_{d\theta}(i,j)$ is defined such that its $(i,j)^{th}$ entry is equal to the number of times that: $f(x1,y1) = i$ and $f(x2,y2) = j$, where $(x2,y2) = (x1,y1) + (dcos\theta, dsin\theta)$. This yields a square matrix of dimension equal to the number of intensity levels in the image, for each distance d and orientation θ. Most relevant co-occurrence matrix derived features used for the purpose of texture discrimination are the angular second moment, correlation, inertia, absolute value, entropy and maximum probability [6, 19]. Table 2 lists descriptions for these features. In this research we calculated the measures at distance $d = 1$ for horizontal, vertical and diagonal orientations at $\theta = 0°, 90°, 45°$ and $135°$.

2.4 Multi-scale features and Wavelet-based texture measurement

Various methods adopted for calculating multi-scale features. The most commonly used are the Wigner distributions, Gabor function and wavelet transforms. Wigner distribution are found to possess interference terms between different components of a signal. These interference terms lead to wrong signal interpretation. Gabor filters are criticized for their non-orthogonality that result in redundant features at different scales or channel. On the other hand, the wavelet transform, being a linear operation, does not produce interference terms nor redundant features. For this reason, our interest is in the application of the wavelet transform to texture analysis. Discrete wavelet transform (DWT) derived features appear to be a suitable tool to be used for digital image texture analysis, because they allow analysis of images at various levels of resolution. The DWT provides powerful insight into an image's spatial and frequency characteristics [20]. Moreover, it has shown to be an efficient descriptor for phenotyping [21]. In general, wavelet analysis is highly capable of revealing aspects of data such as trends, breakdown points, discontinuities in higher derivatives and self similarity [22]. Approximations and details are the most important terms in wavelet analysis. The approximations are the high-scale, low-frequency components of the image signal, while the decomposition process in the wavelet transform generates the coefficient matrices for the level-one approximation and horizontal, vertical and diagonal details. In this study, we include a bi-orthogonal wavelet in which texture details are derived from the three different directions on the same scale as the original image [21].

Table 2: Co-occurrence-matrix based features

Features	Description
Angular second moment (ASM)	Also known as uniformity and it is a measure of cell homogeneity. The maximum value is achieved when all the elements in the co-occurrence matrix are equal. $$ASM = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}[p(i,j)]^2 \quad (10)$$
Correlation	The correlation measures the dependencies between the yeast cell image pixels. μ_x, μ_y and σ_x, σ_y denote the mean and standard deviations of the row and column sums of the co-occurrence matrix respectively. $$Correlation = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\frac{ijp(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y} \quad (11)$$
Inertia	Inertia is also known as contrast. It is calculated by squaring the subtraction of the examined pixel values. Thus, the minimum value is when the pixels have the same grey-level value, and the maximum is achieved when squaring the subtraction of L and 1, i.e. L^2. $$Inertia = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}(i-j)^2p(i,j) \quad (12)$$
Absolute value	It calculates the absolute value of the subtraction of the examined pixel values. Hence it ranges between 0 and L. $$\textit{Absolute value} = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\lvert i-j\rvert p(i,j) \quad (13)$$
Inverse difference	It has relatively high value when the high value in the co-occurrence matrix are near the main diagnoal, where the difference $(i-j)$ is smaller there. $$\textit{Inverse difference} = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\frac{p(i,j)}{1+(i-j)^2}. \quad (14)$$
Entropy	The entropy measures the complexity of the texture. It is a measure of randomness, achieving its highest value when the elements in the matrix are maximally random. $$entropy = -\sum_{i=0}^{L-1}\sum_{j=0}^{L-1}p(i,j)log_2[p(i,j)]. \quad (15)$$
Maximum probability	Gives an indication of the strongest response in the co-occurrence matrix. $$\textit{Maximum probability} = max_{i,j}p(i,j) \quad (16)$$

3 Building a Classification Model

Our dataset consists of 1440 yeast cell instances belonging to two major classes, one representing cells expressing 14-3-3 proteins attached to a green fluorescent protein (GFP) in a low (0mM) $NaCl$ medium, and the other representing same cell strains in a high (50mM) $NaCl$ medium. All these cells are measured for the features mentioned in the previous section after segmenting them. Each instance I in this dataset is mapped to one element of the set (p, n) of positive and negative class labels representing the two different cell classes of low and high $NaCl$ medium respectively. We need to build a classification model to map from instances to predicted classes. Given a classifier and a test set of instances, a two-by-two confusion matrix, also known as contingency table is constructed to represent the dispositions of the set of instances. This matrix forms the basis for many metrics we used to evaluate the classifiers [23].

To find our best classifier system, we prepared an experiment in the *Weka* machine learning workbench [24] with 23 different linear and non-linear machine learning algorithms including the popular predictors such as decision trees, naive Bayes, least-square linear predictors, and support vector machines. We applied a supervised classification on our dataset of 1440 instances (cells) with 10 fold cross validation.

In the evaluation step, our primary focus was on the Area Under *ROC* (*AUC*), but since our dataset is imbalanced with a ratio of cells in a low $NaCl$ to that in a high $NaCl$ medium being $2.7 : 1$, we carefully considered several other metrics of the minority class as well, especially the *AUC* for that class (referred to as A_{min}). We tested various sampling techniques and normalization schemes and feature selection algorithms and evaluated their effect on the final accuracy of the classifiers. In the coming sub-section, the sampling techniques and imbalanced dataset are discussed. The second sub-section address the cross validation technique used on the dataset. Then normalization schemes are discussed. Subsequently, the adopted feature selection algorithms are highlighted. After that, a little detail about the used evaluation metrics. The last subsection is about the classifiers considered in the comparison.

3.1 Imbalanced Dataset and Sampling Techniques

Our dataset S is considered imbalanced since it exhibits an unequal distribution between its positive (yeast cells in low $NaCl$ medium) and negative (yeast cells in a high $NaCl$ medium) classes. Hence, in this domain, we require a classifier that will provide high accuracy for the negative minority class without severely jeopardizing the accuracy of the positive majority class. Conventional evaluation practice of using singular assessment criteria, such as the overall accuracy or error rate, does not provide adequate information in the case of imbalanced learning because most standard algorithms assume or expect balanced class distributions or equal misclassification costs. The problem of learning from imbalanced data is a relatively new challenge that has attracted growing attention from both academia and industry. The induction rules that describe the minority concepts are often fewer and weaker than those of majority concepts, since the minority class is often both outnumbered and under-represented. Successive partitioning of the dataspace results in fewer and fewer observations of minority class examples resulting in fewer leaves describing minority concepts and successively weaker confidence

estimates. In addition, concepts that have dependencies on different feature space conjunctions can go unlearned by the sparseness introduced through partitioning. The application of sampling techniques has shown to improve classifier accuracy. Therefore, we considered three different sampling techniques, namely under-sampling, over-sampling and Synthetic Minority Oversampling technique (*SMOTE*).

We define subsets $S_{min} \subset S$ and $S_{maj} \subset S$ where S_{min} is the set of minority class instances in S, and S_{maj} is the set of majority class instances in S, so that $S_{min} \cap S_{maj} = \{\phi\}$ and $S_{min} \cup S_{maj} = \{S\}$. Random under-sampling removes data from the original data set. In particular, we randomly select a set of majority class instances in S_{maj} and remove these instances from S so that $|S| = |S_{min}|+|S_{maj}|-|E|$, where E represents the set removed by the sampling procedure. Under-sampling readily gives us a simple method for adjusting the balance of the original data set S; however, removing instances from the majority class may cause the classifier to miss important concepts pertaining to the majority class.

In oversampling, multiple instances of certain examples become "tied" since it simply appends replicated data to the original dataset, leading to overfitting. In particular, overfitting in oversampling occurs when classifiers produce multiple clauses in a rule for multiple copies of the sample example which causes the rule to become too specific; although the training accuracy will be high in this scenario, the classification performance on the unseen testing data is generally far worse.

The synthetic minority oversampling technique (*SMOTE*), on the other hand, is a powerful method that has shown a great deal of success in various applications. It creates artificial data based on the feature space similarities between existing minority instances. Specifically, for subset S_{min}, consider the K-nearest neighbours for each instance $x_i \in S_{min}$ for some specified integer K; the K-nearest neighbours are defined as the K elements of S_{min} whose euclidean distance between itself and x_i under consideration exhibits the smallest magnitude along the n-dimensions of feature space *X*. To create a synthetic sample, we randomly select one of the K-nearest neighbours, then multiply the corresponding feature vector difference with a random number $\in [0, 1]$, and finally add this vector to the minority instance $x_i \in S_{min}$ as depicted in Eq. 17, where $\hat{x}_i \in S_{min}$ is one of the K-nearest neighbours for x_i, and $\delta \in [0, 1]$ is a random number. Therefore, the resulting synthetic instance according to Eq. 17 is a point along the line segment joining x_i under consideration and the randomly selected K-nearest neighbour $\hat{x}_i$ [25].

$$x_{new} = x_i + (\hat{x}_i - x_i) \times \delta, \tag{17}$$

3.2 Training, Testing, and Cross Validation

In our prediction problem to predict the cells cultured in high $NaCl$ medium vs. those in low $NaCl$ medium, the classification models are given a training dataset of known ground-truth data, and a testing dataset of unknown first-seen data against which the models are tested. In order to limit problems like over-fitting and give an insight on how the model will generalize to an independent dataset, the widely used 10-fold cross validation is considered [26]. Overfitting occurs when the classification model does not fit this validation data as well as it fits the training data. Cross validation is important in protecting against testing hypotheses suggested

by the data, known as Type III errors [27]. Its advantage is that all observations are used for both training and validation, and each observation is used for validation exactly once.

3.3 Feature Scaling or Normalization

Feature scaling is a method used to standardize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step [28]. Since the range of values of the raw data in our dataset varies, some machine learning algorithms will not work properly without normalization. For example in distance classifiers, the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance. In our work we considered two well-known normalization techniques which are unit-length normalization (UL) and zero-mean and unit-variance normalization (MV) [29]. Feature normalization techniques represent a vital part in building the classification model. They aim at normalizing the individual components of the extracted feature vectors in such a way that the resulting vectors are better suited for classification.

Unit length normalization (UL) scales all of the components x_i (i = 1, 2,.,d) of vector x of all instances in our dataset S in accordance with the expression in Eq. 18 to produce the normalized feature vector x^*, where $||.||$ denotes the norm operator, and ${x_i}^*$ stands for the i^{th} component of the normalized vector x^*.

$$ {x_i}^* = \frac{x_i}{||x||}, i = 1, 2, ...d, \tag{18} $$

The zero-mean and unit-variance normalization is defined in Eq. 19, where μ denotes the mean value of the feature vector x and σ represents its standard deviation. The MV technique transforms the feature vector x to a random variable with a mean value of zero and variance of one. It is assumed the individual components of the feature vector are normally distributed.

$$ {x_i}^* = \frac{(x_i - \mu)}{\sigma}, i = 1, 2, ., d, \tag{19} $$

3.4 Attribute Selection

Attribute selection also known as variable or feature selection, can be defined as a process that chooses a minimum subset of M features from the original set of N features, so that the feature space is optimally reduced according to a certain evaluation criterion. The reduced feature space are the most important parameters which help in predicting the outcome. Finding the best feature subset is usually intractable and many problems related to feature selection have been shown to NP-hard. The objective of feature or variable selection is three-fold:

- improving the prediction performance of the classifiers.
- providing faster and more cost effective classifiers.

- providing a better understanding of the underlying process that generated the data.

Selecting the most relevant variables is suboptimal for building our predictor, particularly if the variable are redundant or irrelevant. Redundant features are those which provide no more information than the currently selected features, and irrelevant features provide no useful information in any context. For our data, we considered three well-known feature selection algorithms which are the Information Gain (IG) method, Correlation Feature Selection (CFS) and Principal Component analysis (PCA).

3.5 Evaluation metrics, *ROC* and *AUC*

A receiver operating characteristic (*ROC*) curve is a two dimensional graphical plot that illustrates the performance of a binary classifier system, which predicts a two-class problem in which the outcomes are labelled either as positive (p) or negative (n) [30]. The curve is created by plotting the true positive rate (TPR) on the Y axis against the false positive rate (FPR) on the X-axis at various threshold settings. TPR is also known as sensitivity in biomedical informatics, or recall in machine learning [31]. TPR defines how many correct positive results occur among all positive samples available during the test. On the other hand, FPR also known in biomedical informatics as (1-Specificity) defines how many incorrect positive results occur among all negative samples available during the test [32]. Each prediction result or instance of a confusion matrix represents one point in the *ROC* space [33].

The *ROC* graphs are useful for evaluating our classifiers and visualizing their performance. *ROC* graphs have been successfully used in medical decision making, and in recent years, they are gaining popularity in machine learning and data mining research, due to the realization that scalar measures such as simple classification accuracy, error rate or error cost are often poor metrics for measuring performance. *ROC* graphs have properties that make them especially useful for domains with skewed class distribution and unequal classification error costs. These characteristics have become increasingly important as research continues into the areas of cost-sensitive learning and learning in the presence of unbalanced classes [23]. *ROC* analysis provides tools to select possibly optimal models [34]. Classifiers appearing on the left-hand side of an *ROC* graph, near the X axis, may be thought of as conservative: they make positive classifications only with strong evidence so they make few false positive errors, but they often have low true positive rates as well. Classifiers on the upper right-hand side of an *ROC* graph may be thought of as liberal: they make positive classification with weak evidence so they classify nearly all positives correctly, but they often have high false positive rates. Many real world domains are dominated by large numbers of negative instances, so performance in the far left-hand side of the *ROC* graph comes more interesting. We have used the area under the *ROC* curve (AUC), also known as c-statistic [35], which is usually interpreted according to the following ratings: $x = 1$, perfect; $1 > x \geq 0.9$, excellent; $0.9 > x \geq 0.8$, good; $0.8 > x \geq 0.7$, fair; $0.7 > x \geq 0.6$, poor; $0.6 > x \geq 0.5$, fail (random guessing for *AUC*); $x < 0.5$, unacceptable [36]. *AUC* is a common statistic most often used for model comparison in the machine learning community [37]. Despite its popularity, some machine learning researches show that the *AUC* is quite noisy as a classification measure [38], and has some other significant problems in model comparison [39, 40]. Therefore, we also considered the standard

accuracy metric, which is widely used to get an additional insight into the results. More importantly, we considered the *AUC* for the minority class (A_{min}) as well. A_{min} reveals the dangers of blindly looking into the *AUC* alone in the raw dataset classifiers evaluation. However, the *AUC* becomes a safe metric when the data is preprocessed with a sampling technique as the result will show in the following section.

3.6 Classifiers Evaluated

In this work, 23 different linear and non-linear famous classification systems were evaluated on our dataset. A list of these models along with short descriptions are shown in Table 3. These classifiers were evaluated using the *Weka* machine learning workbench [24], and for *ROC* analysis, we use *pROC* package [41] of the *R* programming language and environment for statistical computing [42] connected with Weka through the *RWeka* package [43]. The next section shows how these classifiers perform on our dataset.

4 Results

Using our dataset with the presented sophisticated features, we evaluated the classifiers by considering the area under *ROC* curve (*AUC*) as our main metric in addition to the minority class A_{min} and *ACC* (accuracy) of each classification system. We listed in Table 4 the *AUC*, A_{min} and *ACC* scores for each classifier under the best sampling, normalization and feature selection algorithms. Most classifiers have optimal results with the *SMOTE* sampling technique and the *MV* normalization scheme. The Feature Selection with the best results was the *IG* method. It is clear from the results that a number of classifiers were able to excellently discriminate between the two cell groups with *AUC*, A_{min} and *ACC* metrics above 0.9. The following subsections reveal the power of sampling, the effect of normalization and feature selection, in addition to the power of the discriminators based on our designed feature space.

4.1 Power of Sampling

The first obvious fact from the results in Table 4 is the power of data sampling on the classifiers performance. *SMOTE* has not failed to improve the overall classification within all the algorithms tested, unlike the under-sampling and over-sampling methods. This makes *SMOTE* an excellent choice for our data. Moreover, *SMOTE* addresses the information loss of the under-sampling and the over-representation issue of the over-sampling methods. After applying *SMOTE*, the *AUC* was improved in average by $.025$, and the average accuracy increased slightly by 0.007. However, the strongly significant different is shown when considering the *AUC* for the minority class (A_{min}) where the average improvement was increased by $.139$ per classifier from an average of $.708$ to $.847$. This also reveals the reason why accuracy is not a sufficient measure in our imbalanced dataset. Generally, the power of sampling was more conspicuous in *PART, C4.5, JRIP, SMO* and *VFDT* classifiers.

Table 3: Classification Algorithms evaluated in this study

Classifier	Description
C4.5	A decision tree classifier. At each node of the tree, C4.5 chooses the attribute that most effectively splits its set of samples into subsets enriched in one class or the other [44].
Adaptive Boosting (AdaB)	A machine learning meta-algorithm used in conjunction with a weak learner (Decision Tree) to improve its performance [45].
PART	Builds a partial C4.5 decision tree in each iteration and makes the ”best” leaf into a rule [46].
Decision Table Majority (DTM)	A decision table with a default rule mapping to the majority class. It has a set of features (schema) and a set of labelled instances (body) [47].
Decision Stump (DSmp)	A model consisting of a one-level decision tree. i.e, it is a decision tree with one internal node (the root) which is immediately connected to the terminal nodes (its leaves) [48].
One Rule (OneR)	OneR generates one rule for each predictor in the data, then selects the rule with the smallest total error as its ”one rule” [49].
JRip	JRip implements a propositional rule learner, Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [50].
Bayes Network (BNet)	Bayes Network learning represents the dataset variables via a directed acyclic graph (DAG) based on probability theory [51].
K-Nearest Neighbour (IBK)	Predicts the class of the single nearest training instance for each test instance [52].
Locally Weighted Learning (LWL)	Uses an instance-based algorithm (Decision Stump) to assign instance weights [53].
LogitBoost (ALR)	Performs additive logistic regression on the base learner (Decision Stump) [54].
Random Committee (RCom)	Builds an ensemble of randomizable base classifiers (Random Tree). The final prediction is a straight average of the predictions generated by the individual base classifiers..
Random Subspace (RSub)	Constructs a decision tree based classifier (REPTree) with multiple trees constructed in randomly chosen subspaces [55].
Hoeffding Tree (VFDT)	An incremental decision tree induction algorithm capable of learning from massive data. It assumes that the distribution of variables does not change over time [56].
Logistic Model Tree (LMT)	A logistic model tree basically consists of a standard decision tree structure with logistic regression functions at the leaves [57].
REPTree	Fast decision tree learner. Builds a decision/regression tree using information gain/-variance and prunes it using reduced-error pruning.
Random Forest (RFor)	Constructs a forest of random decision trees at training time and outputting the mode class (classification) or mean prediction (regression) of the individual trees [58].
Random Tree (RTre)	Constructs a tree with randomly chosen attributes at each node.
Logistic (Log)	Building and using a multinomial logistic regression model with a ridge estimator [59].
Stochastic Gradient Descent (SGD)	Implements stochastic gradient descent for learning various linear models (binary SVM, binary logistic regression, squared loss, Huber loss and epsilon-insensitive loss).
Sequential Minimal Optimization (SMO)	Sequential minimal optimization algorithm for training a support vector classifier [60].
SimpleLogistic (SLog)	Classifier for building linear logistic regression models. LogitBoost with simple regression functions as base learners is used for fitting the logistic models [57].
Voted Perceptron (VPer)	Based on a linear predictor function combining a set of weights with the feature vector, and a transformation of online learning, in that it processes elements one at a time [61].

Table 4: *AUC*, A_{min} and *ACC* of classification algorithms using raw dataset, and after sampling, normalization and feature selection.

Classifier	Raw Dataset			Sampled			Normalized			Features Selected		
	AUC	A_{min}	*ACC*	*AUC*	A_{min}	*ACC*	*AUC*	A_{min}	*ACC*	*AUC*	A_{min}	*ACC*
$RCom$	.972	.937	.940	.984	.980	.943	.983	.979	.939	.982	.977	.939
$RSub$	.959	.923	.911	.978	.978	.928	.976	.976	.921	.975	.974	.927
$RFor$	.965	.917	.920	.981	.974	.937	.977	.972	.922	.978	.970	.930
$SLog$	.926	.841	.867	.926	.911	.854	.926	.911	.853	.926	.911	.854
Log	.945	.839	.898	.916	.925	.872	.916	.919	.872	.916	.920	.872
$LogB$	.930	.837	.878	.933	.918	.874	.932	.918	.868	.932	.918	.868
$DTab$	.922	.833	.876	.942	.933	.872	.942	.936	.873	.943	.936	.872
LMT	.927	.808	.901	.937	.898	.908	.936	.896	.905	.937	.895	.904
$REPT$	.891	.789	.887	.932	.912	.892	.923	.903	.887	.923	.904	.887
$AdaB$	.908	.767	.855	.919	.899	.850	.917	.895	.849	.917	.895	.849
$PART$	.884	.764	.901	.916	.892	.904	.932	.906	.910	.925	.901	.909
$C4.5$	.853	.747	.893	.915	.885	.914	.900	.864	.906	.898	.861	.906
$JRip$	.853	.747	.893	.916	.887	.905	.918	.891	.894	.911	.877	.894
$BNet$	.881	.703	.808	.888	.853	.830	.889	.862	.824	.889	.862	.824
SGD	.950	.683	.888	.869	.815	.869	.866	.810	.864	.866	.810	.864
$RTre$	.853	.669	.883	.877	.830	.877	.879	.835	.879	.875	.826	.874
IBK	.851	.668	.881	.881	.839	.881	.881	.839	.881	.881	.839	.881
LWL	.870	.644	.736	.874	.819	.773	.874	.819	.773	.874	.819	.773
SMO	.803	.628	.867	.866	.815	.866	.866	.814	.865	.866	.814	.865
$OneR$	.767	.553	.834	.795	.733	.794	.795	.733	.794	.795	.733	.794
$DSmp$	.769	.450	.733	.774	.692	.774	.774	.692	.774	.774	.692	.774
$VPer$	.543	.289	.638	.526	.515	.527	.802	.735	.798	.802	.736	.798
$VFDT$	.500	.267	.733	.638	.582	.606	.736	.668	.661	.736	.668	.661

4.2 Effect of Normalization and Feature Selection

Normalization showed an extra average improvement of $.015$ and $.013$ for the *AUC* and A_{min} respectively, and improved accuracy by 0.012 over the sampled dataset. The most significant difference of normalization was shown in *VPer* and *VFDT* classifiers. However, the best feature selection algorithm (*IG*) showed no change in the averages of *AUC* and A_{min} and only a little accuracy increase for the *RFor* and *RSub* models. The negligible effect of feature selection could be addressed to our choice of well composed relevant features.

4.3 The Powerful discriminators

Wavelet-based texture measurement has shown their superiority to discriminate our instances in the top classifiers. Specifically, the *Smoothness* and *Uniformity* textures in the horizontal, diagonal and vertical wavelet detail images were used as root nodes in most base trees in the *RCom*, *RFor*, *RSub* and *C4.5* Decision tree classifiers. In addition, the fused invariant moments with wavelet details in many dimensions obtained high weights in the functions of *SLog* and *LMT* classifiers revealing their discriminative power. They also showed up multiple times within the

decision trees built by various models. The second order histogram features extracted from the co-occurrence matrix had also played a major role in the discrimination of yeast cells, they showed up in almost every classifier, though at lower level in the decision trees or with a smaller weight in the regression functions.

Most of the top classifiers built complex models that are not easy to interpret. *RCom* has built ten random trees of sizes between 267 and 297. *RSub* built a "relatively" less complex model of ten random *REP* trees of sizes between 47 and 87. *RFor* built ten random trees each using seven random attribute values in its construction. On the other hand, the *Logistic* and *Simple Logistic* algorithms built decent regression functions. *Simple Logistic* built its function with a few attributes (22 of the total 90 considered). This makes *Logistic* regression a much more appealing classification model for our domain. The *Logistic regression* classifier has two output terms, coefficients and odds ratios. High coefficient values when predicting the negative class were noticed in many moment invariant features in the wavelet detail images. High odds ratios were noticed in texture measurements and co-occurrence matrix features as well. The *Simple Logistic* has used in its function, eight features from the co-occurrence matrix, five features from the wavelet texture measurement, two features from the combination of wavelet and moment invariants, four features from moment invariants, one texture measure and two basic shape descriptors.

Support Vector Machines (*SVMs*) are famous classifiers. The *SMO*, which is an implementation of the *SVMs*, did not rank as an excellent classifier when using the default parameters. However, this changes when optimizing its parameters by increasing the complexity constant to 5, disabling any normalization within the classifier itself, fit logistic models to *SVM* outputs and use a normalized Polynomial Kernel. This optimization pushed the *SMO* into the top five classifiers with an *AUC* of 0.92.

Next we study the considered feature sets for their contribution to the power of discrimination.

4.4 Feature spaces performance

To investigate whether our composed feature sets has any added value to the discrimination power, we started by comparing the classification performance using different set of features including basic shape descriptors, invariant moments, wavelet texture measurement, invariant moments on wavelet detail images, co-occurrence matrix derived features, basic texture measurement and a full feature space combining all the feature sets. The difference is shown in Fig. 2 and 3. This test was performed on both $Logistic$ and $C4.5$ classifiers. Logistic regression was chosen as it is the top "non-random based" classifier and C4.5 as a non-linear approach for comparison. In both classifiers, the benefits of using the full set is obvious. In the Logistic classifier, the performance of individual feature sets were not sufficient, except for the basic texture measurement which shows a very good discrimination with an *AUC* of 0.82. However, using the full features set shifted the performance of the Logistic classifier into the excellent category with an *AUC* of 0.92. In the non-linear $C4.5$ decision tree classifier, all the individual feature sets except the basic set have good discrimination. However, none is ranked as excellent. Only when the feature sets are fused together the classifier has an excellent discrimination rate with an *AUC* of 0.91.

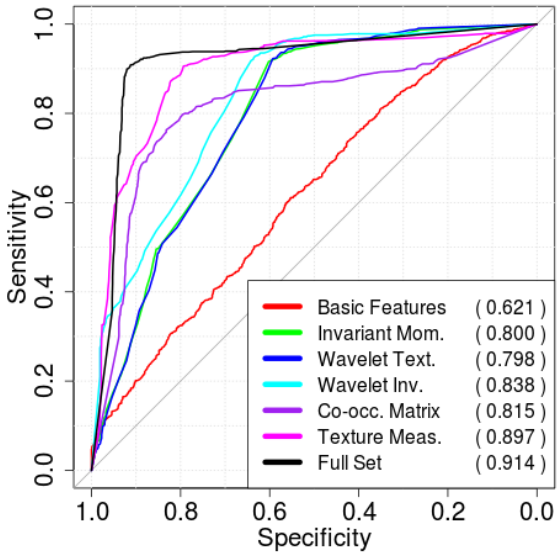


Figure 2: *ROC* analysis and *AUC* value of C4.5 classifier using various feature sets

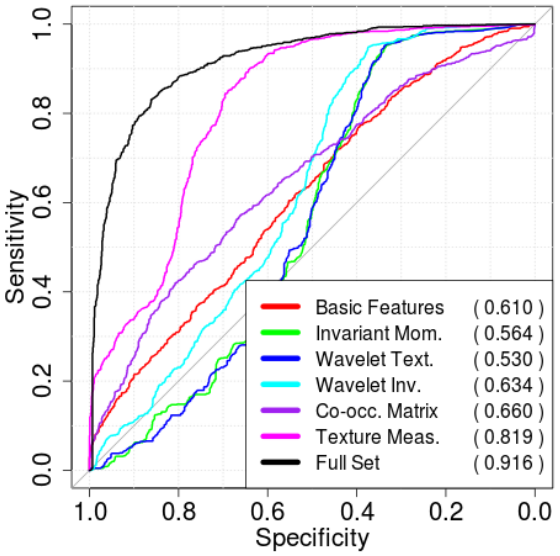


Figure 3: *ROC* analysis and *AUC* value of Logistic classifier using various feature sets

Since 3^{rd} order moment invariants might be more noise-prone than their 2^{nd} order counterpart, we study whether it really adds any discrimination value by creating only two small feature spaces. The first having only the second moment invariants while the second feature space contains the whole set of seven invariant moments. Figure 4 and 5 show the *ROC* graph of the performance of the $Logistic$ and $C4.5$ classifiers on both feature spaces. The third order moments show to have an additional discriminative value in both classifiers for this dataset.

5 Conclusion

In this paper, we addressed our principal research question on how a machine learning approach can discriminate *S. cerevisiae* yeast cells cultivated in high or low $NaCl$ medium? and the sub-questions on what extracted object features are the best to predict whether a measured cell belongs to high or low sodium chloride ($NaCl$) medium? and what machine learning algorithm can best work on our dataset of individual cell measurements? Moreover, what are the best sampling, normalization and feature selection algorithms to be used on such dataset?

From our experiment, we show that a machine learning approach is efficient in our classification problem. The Wavelet-based texture measurements, co-occurrence matrix derived features, moment invariant features and texture measurement derived from the image histogram, forms a set of powerful discriminators in the top classification models, from which the Logistic model was chosen as the most efficient for classifying our cells. Furthermore, optimization of an *SvM* classifier might be possible. Sampling of our dataset with the *SMOTE* method showed to have a significant effect on building the classification model system. In addition, the *MV* normalization scheme showed an extra improvement. However, the best feature selection algorithm tested showed a little non-significant improvement probably due to the fact that we selected the right feature sets. With this machine learning process and the chosen feature sets, it becomes possible

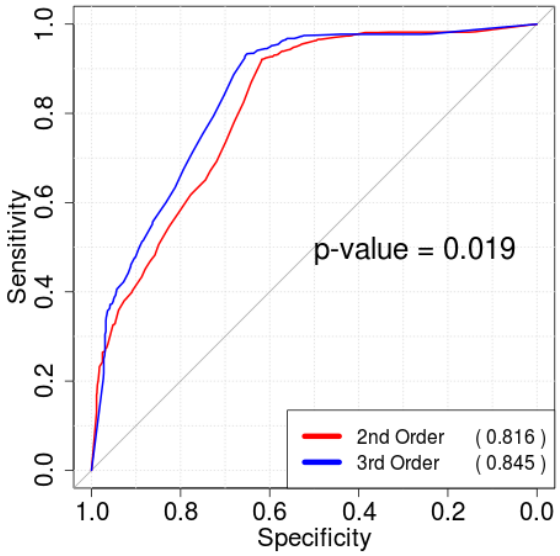


Figure 4: Performance of C4.5 classifier using second and up-to third order moment invariant features

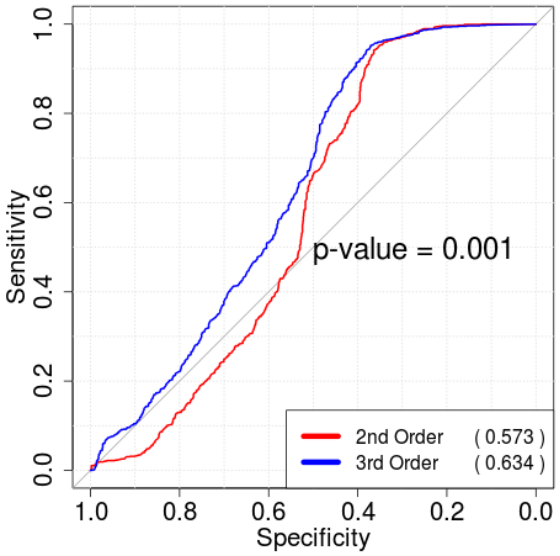


Figure 5: Performance of Logistic classifier using second and up-to third order moment invariant features

as future work, to classify different cell strains and conditions in a high-volume high-throughput studies.

6 Acknowledgement

We would like to thank Paul van Heusden who offered us the yeast cell images. Moreover, this work is partly supported by the Erasmus Mundus JOYSLEEN project, the *Landelijke Stichting voor Blinden en Slechtziendenand* (LSBS) and the Raymond-Sackler organizations.

References

[1] M. Tleis, G. Anemaet, P. van Heusden and F. Verbeek. Image analysis platform for yeast biologists. In *Advances in Biomedical Engineering (ICABME), 2013 2nd International Conference on*, pages 105–108. IEEE, 2013.

[2] M. Tleis and F. J. Verbeek. Extracting contours of oval-shaped objects by hough transform and minimal path algorithms. In *Sixth International Conference on Digital Image Processing*, pages 915903–915903. International Society for Optics and Photonics, 2014.

[3] J. Gubbi, S. Marusic and M. Palaniswami. Smoke detection in video using wavelets and support vector machines. *Fire Safety Journal*, 44(8):1110–1115, 2009.

[4] A. Baraldi and F. Parmiggiani. An investigation of the textural characteristics associated with gray level cooccurrence matrix statistical parameters. *Geoscience and Remote Sensing, IEEE Transactions on*, 33(2):293–304, 1995.

[5] A. Materka, M. Strzelecki et al. Texture analysis methods–a review. *Technical university of lodz, institute of electronics, COST B11 report, Brussels*, pages 9–11, 1998.

[6] R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.

[7] R. Gonzalez and R. Woods. *Digital Image Processing*. Pearson Education, third edition, 2008.

[8] A. Papolulis. *Random Variables and Stochastic Processes*. McGraw Hill Book Company, 1965.

[9] A. M. Vossepoel and A. W. M. Smeulders. Vector code probability and metrication error in the representation of straight lines of finite length. *Computer Graphics and Image Processing*, 20:347–364, 1982.

[10] W. Rasband. Imagej, 2013. U. S. National Institutes of Health, Bethesda, Maryland, USA, http://imagej.nih.gov/ij/ 1997-2013.

[11] R. Gonzalez and R. Woods. *Digital Image Processing*. Pearson Education, third edition, 2008.

[12] L. Keyes and A. Winstanley. Using moment invariants for classifying shapes on large-scale maps. *Computers, Environment and Urban Systems*, 25(1):119–130, 2001.

[13] K. Jafari-Khouzani and H. Soltanian-Zadeh. Rotation-invariant multiresolution texture analysis using radon and wavelet transforms. *Image Processing, IEEE Transactions on*, 14(6):783–795, 2005.

[14] P. van der Putten, L. Bertens, J. Liu, F. Hagen, T. Boekhout and F. J. Verbeek. Classification of yeast cells from image features to evaluate pathogen conditions. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 6506 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*. 2007.

[15] M.-K. Hu. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187, 1962.

[16] J. S. Weszka, C. R. Dyer and A. Rosenfield. A comparative study of texture measures for terrain classification. *IEEE Trans. Systems Maintenance Cybernet*, pages 269–285, 1976.

[17] H. Niemann. *Pattern analysis*. Springer-Verlag Berlin, 1981.

[18] R. Lerski, K. Straughan, L. Schad, D. Boyce, S. Blüml and I. Zuna. Viii. mr image texture analysisan approach to tissue characterization. *Magnetic resonance imaging*, 11(6):873–887, 1993.

[19] M. Lahtinen, K. Holli, L. Harrison, P. Dastidar, S. Soimakallio and H. Eskola. Software phantoms for texture analysis. In *World Congress on Medical Physics and Biomedical Engineering, September 7-12, 2009, Munich, Germany*, pages 308–311. Springer, 2009.

[20] M. Kociołek, A. Materka, M. Strzelecki and P. Szczypiński. Discrete wavelet transform–derived features for digital image texture analysis. In *Proc. of Interational Conference on Signals and Electronic Systems*, pages 163–168. 2001.

[21] L. Cao. *Biological model representation and analysis*. Ph.D. thesis, Section imaging and bioinformatics, Liacs, Leiden University, 2014.

[22] G. Shruthi and R. K. AN. Image reconstruction using discrete wavelet transform. *IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)*, 2(4):14–20, 2013.

[23] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[24] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[25] H. He and E. A. Garcia. Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9):1263–1284, 2009.

[26] S. Kim, H. Zhang, R. Wu and L. Gong. Dealing with noise in defect prediction. In *Software Engineering (ICSE), 2011 33rd International Conference on*, pages 481–490. IEEE, 2011.

[27] J. P. Donate, P. Cortez, G. G. Sanchez and A. S. De Miguel. Time series forecasting using a weighted cross-validation evolutionary artificial neural network ensemble. *Neurocomputing*, 109:27–32, 2013.

[28] N. K. Verma, L. Rao and S. K. Sharma. Motor imagery eeg signal classification on dwt and crosscorrelated signal features. In *Industrial and Information Systems (ICIIS), 2014 9th International Conference on*, pages 1–6. IEEE, 2014.

[29] V. Štruc and N. Pavešic. A comparison of feature normalization techniques for pca-based palmprint recognition. In *Proc. Internat. Conf. MATHMOD*, pages 2450–2453. 2009.

[30] V. Lakafosis, A. Traille, H. Lee, E. Gebara, M. M. Tentzeris, G. DeJean and D. Kirovski. Rfid-coa: The rfid tags as certificates of authenticity. In *RFID (RFID), 2011 IEEE International Conference on*, pages 207–214. IEEE, 2011.

[31] S. Li, Y. Zhang, J. Xu, L. Li, Q. Zeng, L. Lin, Z. Guo, Z. Liu, H. Xiong and S. Liu. Non-invasive prostate cancer screening based on serum surface-enhanced raman spectroscopy and support vector machine. *Applied Physics Letters*, 105(9):091104, 2014.

[32] J. Senthilnath, V. Das, S. Omkar and V. Mani. Clustering using levy flight cuckoo search. In *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012)*, pages 65–75. Springer, 2013.

[33] G. Rastrelli, G. Corona, L. Vignozzi, E. Maseroli, A. Silverii, M. Monami, E. Mannucci, G. Forti and M. Maggi. Serum psa as a predictor of testosterone deficiency. *The journal of sexual medicine*, 10(10):2518–2528, 2013.

[34] T. Beshah and S. Hill. Mining road traffic accident data to improve safety: Role of road-related factors on accident severity in ethiopia. In *AAAI Spring Symposium: Artificial Intelligence for Development*. Citeseer, 2010.

[35] C. Herder, J. Baumert, A. Zierer et al. Immunological and cardiometabolic risk factors in the prediction of type 2 diabetes and coronary events: Monica/kora augsburg case-cohort study. *PLoS One*, 6(6):e19852, 2011.

[36] P. Tsai, M. Torabinejad, D. Rice and B. Azevedo. Accuracy of cone-beam computed tomography and periapical radiography in detecting small periapical lesions. *Journal of endodontics*, 38(7):965–970, 2012.

[37] B. F. Chimieski and R. D. R. Fagundes. Association and classification data mining algorithms comparison over medical datasets. *Journal of Health Informatics*, 5(2), 2013.

[38] B. Hanczar, J. Hua, C. Sima, J. Weinstein, M. Bittner and E. R. Dougherty. Small-sample precision of roc-related estimates. *Bioinformatics*, 26(6):822–830, 2010.

[39] J. M. Lobo, A. Jiménez-Valverde and R. Real. Auc: a misleading measure of the performance of predictive distribution models. *Global ecology and Biogeography*, 17(2):145–151, 2008.

[40] D. J. Hand. Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine learning*, 77(1):103–123, 2009.

[41] X. Robin, N. Turck, A. Hainard, N. Tiberti, F. Lisacek, J.-C. Sanchez and M. Mller. proc: an open-source package for r and s+ to analyze and compare roc curves. *BMC Bioinformatics*, 12:77, 2011.

[42] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015. URL `http://www.R-project.org/`.

[43] K. Hornik, C. Buchta and A. Zeileis. Open-source machine learning: R meets weka. *Computational Statistics*, 24(2):225–232, 2009.

[44] R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[45] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Thirteenth International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, San Francisco, 1996.

[46] E. Frank and I. H. Witten. Generating accurate rule sets without global optimization. In J. Shavlik (editor), *Fifteenth International Conference on Machine Learning*, pages 144–151. Morgan Kaufmann, 1998.

[47] R. Kohavi. The power of decision tables. In *Machine Learning: ECML-95*, pages 174–189. Springer, 1995.

[48] W. Iba and P. Langley. Induction of one-level decision trees. In *Proceedings of the ninth international conference on machine learning*, pages 233–240. 1992.

[49] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90, 1993.

[50] W. W. Cohen. Fast effective rule induction. In *Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.

[51] N. Friedman, D. Geiger and M. Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[52] D. Aha and D. Kibler. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[53] C. Atkeson, A. Moore and S. Schaal. Locally weighted learning. *AI Review*, 1996.

[54] J. Friedman, T. Hastie and R. Tibshirani. Additive logistic regression: a statistical view of boosting. Technical report, Standford University, 1998.

[55] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998. URL `http://citeseer.ist.psu.edu/ho98random.html`.

[56] G. Hulten, L. Spencer and P. Domingos. Mining time-changing data streams. In *ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pages 97–106. ACM Press, 2001.

[57] N. Landwehr, M. Hall and E. Frank. Logistic model trees. *Machine Learning*, 59(1-2):161–205, 2005.

[58] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[59] S. le Cessie and J. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41(1):191–201, 1992.

[60] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schoelkopf, C. Burges and A. Smola (editors), *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998. URL `http://research.microsoft.com/~jplatt/smo.html`.

[61] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. In *11th Annual Conference on Computational Learning Theory*, pages 209–217. ACM Press, New York, NY, 1998.