4
Machine Learning to Identify Subtle Patterns and Improve Object Recognition

In this chapter, we choose some sophisticated features and use them in a machine learning approach to improve the recognition of objects and to identify subtle patterns between different objects. As a case study, we applied this machine learning approach on S. cerevisiae yeast cells. In the first experiment, we identify different characteristics between two different cell groups cultivated under different stress levels. In the second experiment, we improve the recognition of S. cerevisiae yeast cells by classifying the intact cells from artefacts, i.e. debris and dead cells existing in the culture medium. Since the dataset generated in both experiments are imbalanced, we were careful in the evaluation of classifiers built by the machine learning process. We considered sampling of data, scaling, feature selection, cross-validation and evaluation metrics.

This chapter is based on the following publications:

- Mohamed S. Tleis and Fons J. Verbeek. "Machine Learning approach to discriminate *Saccharomyces cerevisiae* yeast cells using sophisticated image features." Journal of Integrative Bioinformatics, 12(3):276, 2015.
- Mohamed S. Tleis and Fons J. Verbeek. "Machine Learning approach to segment *Saccharomyces cerevisiae* yeast cells." In third International Conference on Advances in Biomedical Engineering (ICABME). pp. 278-281. IEEE, 2015.

4.1 Introduction

Subtle Patterns such as different characteristics that are hard to notice from the basic measurment and statistical analysis of data, are not easily extracted from these measurement or basic statistical analysis. These subtle patterns are especially intrinsic in high throughput screening (HTS) where thousands of images are analysed. Moreover, when biologists study an organism in two different conditions, it can be not possible to know if different groups of that organism have different characteristics. In addition, recognition of objects in images prior to their analysis is quintessential in order to generate meaningful conclusions. Thus, the need of an automatic system to extract those hidden features and to recognize objects.

As an application, we take the *S. cerevisiae* as a case study. The segmentation module of the automated analysis platform, i.e. *YeastAnalysis*, discussed in Chapter 2, provides a segmentation of the cells obtained from 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. 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 for an automatic system to identify any charactersitic difference.

Our novel segmentation algorithm in *YeastAnalysis* segments intact cells as well as artefacts. These artefacts can be dead cells or debris existing in the culture medium. Artefacts have negative effects on the analysis result of the experiments. These artefacts can be interactively excluded after the segmentation using *YeastAnalysis* platform; however, in high throughput screening (HTS) where thousands of images are analysed, manual checking is not feasible anymore. Thus, the need of an automatic system to exclude such artefacts prior to analysis.

The research question in this work is what machine learning approach can be used to improve object recognition and identify subtle patterns in a dataset of sophisticated features? and the sub-question is whether the combination of various feature sets can improve the performance of this machine learning approach? The features (attributes) for training the machine learning system were selected in a way to offer a more sophisticated description of the intensity and morphology characteristics of the objects. The machine learning workflow is depicted in Fig. 4.1.

Applying the extraction techniques that we will use is mentioned in the first section. We created two different dataset on the basis of sophisticated features. Since our dataset 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 prior to feature selection algorithms. A number of linear and non-linear classifiers were evaluated to select the best model that can classify the instances in the first dataspace into cells belonging to a different group, i.e. different strain, or cultured in a different medium, and the best model that can classify the instances in the second dataset into intact cells or artefacts.

The result shows that many classifiers have an excellent performance after data preprocessing. Consequently, two classification models are built. As a result, the first model allows to identify the different characteristics of objects, while the second model has raised the performance level of the measurement and consequently the pattern recognition system.

In the next section, we will discuss the sophisticated feature sets we presented then we discuss the classification process for our dataset.

4.2 Sophisticated Features

In Chapter 2 we presented our developed platform to perform image analysis on *S. cerevisiae* yeast cells. This platform can segment the 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 4.2 shows a sample two-channels image of *S. cerevisiae* yeast cells. The first channel in Fig. 4.2(a) is a bright-field channel depicting yeast cell structures. The second overlaid channel in Fig. 4.2(b) is a fluorescent channel of the *BMH1* gene expressed Bmh1 protein binding with GFP protein cultivated in low NaCl medium. The yellow contours surrounding the cells in Fig. 4.2 are the results of our segmentation algorithm [Tle14]. Such segmentation of individual cells enables us to introduce sophisticated features and texture measurement to describe the characteristics of cell morphology and intensity distribution of individual cells. These features and texture measurement (cf. Section 1.3.2) facilitate the analysis and discrimination of different yeast cells.

Next, we define the concept of extraction techniques. Subsequently, we select and discuss well-known extraction techniques that are useful in analyzing biological images. The benefits of combining these extraction techniques will be revealed in the results. Before that, we discuss building a machine learning system using the feature set extracted using these techniques explained in this section.

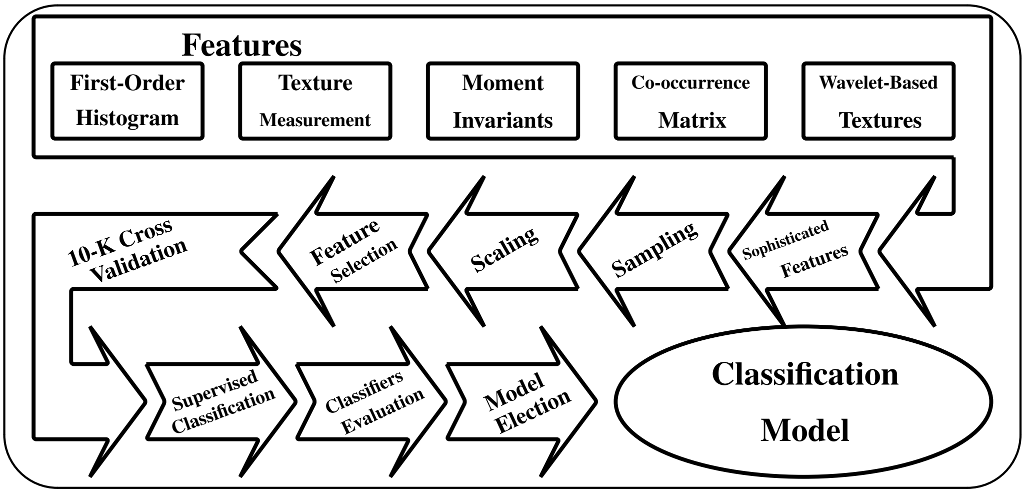

Figure 4.1: *Workflow for building the classification models.*

4.2.1 Feature Extraction Techniques

Feature extraction is one very important area of application in image processing, in which algorithms are used to detect and isolate various desired portions or shapes (features) of a digitized image. Feature extraction is defined as locating those pixels in an image that have some distinctive characteristics [Gub09]. In our research, we considered the most known feature extraction techniques in image analysis. These techniques are classified into histogram based features and the moment invariants derived from them, co-occurrence matrix based features, and multi-scale features [Mat98]. Hereafter, we start discussing the histogram based features then the moment invariants derived from them. Subsequently, we explain the additional features derived from the co-occurrence matrix. We complete this sub-section by highlighting on the multi-scale features and specifically wavelet-based texture features.

First order statistics based features

The histogram of intensity levels is 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 contains 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 study, we considered many basic shape descriptors based on the first order statistics, and the most relevant texture features among those originally proposed by Haralick et al. [Har79, Gon08]. These features were already discussed in Section 2.4 and a list of those features are listed in Table 2.1 and Table 2.2.

Another way to characterize the texture is by deriving moment invariants from the first order statistical information in the histogram [Pap65]. The following sub-section discusses these moment invariant features.

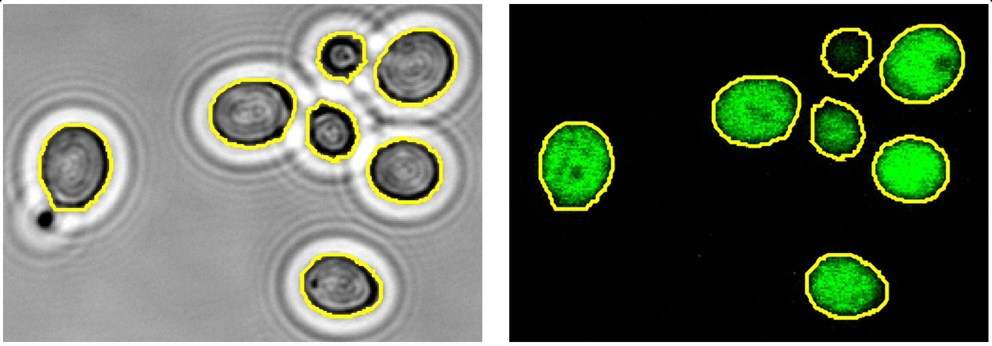
(a) Bright-field channel depicting structure of cells. (b) Fluorescent channel depicting *Bmh1-GFP* gene expressed protein.

Figure 4.2: *S. cerevisiae Yeast Cell Images.*

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 a certain particular weighted average, i.e. moment, of the pixel intensities in an image or a function of such moments chosen to have some attractive property or interpretation. 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 [Key01]. For example, the first rotation invariant moment is known to be analogous to the moment of inertia around the image's centroid, where the pixel intensities are analogous to physical density. The seventh one, is skew invariant, which enables it to distinguish mirror images of otherwise identical images [Hu62]. Although wavelet transform (discussed in section 4.2.1) is scale invariant, it is not in all cases translation or rotation invariant [Jaf05], 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 [Key01] as well as in many other applications [Mat98]. In yeast studies, the first and second moment invariants were the top predictors to classify virulent from non virulent cells [van07].

Hu [Hu62] defines seven shape descriptor values derived from central moments through order three that are independent to object translation, scale and orientation. Translation invariance is achieved by computing moments that are normalized with respect to the center of gravity so that the center of mass of the distribution is at the origin (central moments). Size invariant moments are derived from algebraic invariants but these can be shown to be the result of a simple size normalization. From the second and third order values of the normalized central moments a set of seven invariant moments can be computed which are independent of rotation. The set of seven moment invariants proposed by Hu are widely known [Gon08], and hence we adopted them in our study. We define Hu's set as in Eq. 4.1, 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.

$$hu = \{\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5, \Phi_6, \Phi_7\} \tag{4.1}$$

The effectiveness of moment invariants will increase when fused with the results of other techniques [Key01]. In this research we fuse them with 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.

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 [Mat98] 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 [Wes76], and considered to be important in automated image analysis [Nie81]. The second-order statistical features for texture analysis are derived from the co-occurrence matrix [Har79]. They were demonstrated to feature a potential for effective texture discrimination in biomedical images as well [Ler93].

The second-order histogram is defined as the co-occurrence matrix $C_{d\theta}(i,j)$. 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 [Mat98]. Formally, for image $f(x,y)$ with a set of L discrete intensity levels, the matrix $C_{d\theta}(i,j)$ 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 $(x2,y2) = (x1,y1) + (d\cos\theta, d\sin\theta)$. 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 [Har79, Lah09]. Table 4.1 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 $135°$ to achieve a degree of rotational invariance.

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 functions and wavelet transforms. These transform methods of texture analysis represent an image in a space whose coordinate system has an interpretation that is closely related to the characteristics of a texture, i.e. frequency or size. 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 [Koc01]. Moreover, it has shown to be an efficient descriptor for phenotyping [Cao14]. In general, wavelet analysis is highly capable of revealing aspects of data such as trends, breakdown points, discontinuities in higher derivatives and self similarity [Shr13].

Table 4.1: 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. $$ASM = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}[p(i,j)]^2 \quad (4.2)$$
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. $$Correlation = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\frac{ijp(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y} \quad (4.3)$$
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. $$Inertia = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}(i-j)^2p(i,j) \quad (4.4)$$
Absolute value	It calculates the absolute value of the subtraction of the examined pixel values. Hence it ranges between 0 and L. $$Absolute\ value = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\vert i-j\vert p(i,j) \quad (4.5)$$
Inverse difference	It has relatively high value when the high value in the co-occurrence matrix are near the main diagonal, where the difference $(i-j)$ is smaller there. $$Inverse\ difference = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\frac{p(i,j)}{1+(i-j)^2}. \quad (4.6)$$
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. $$entropy = -\sum_{i=0}^{L-1}\sum_{j=0}^{L-1}p(i,j)\log_2[p(i,j)]. \quad (4.7)$$
Maximum probability	Gives an indication of the strongest response in the co-occurrence matrix. $$Maximum\ probability = max_{i,j}\,p(i,j) \quad (4.8)$$

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 [Cao14]. The derived wavelet-based textures are the same as those derived for the original cells and listed in Table 2.2.

In this section we highlight on the powerful set of extraction techniques that we haven chosen in our research to measure individual yeast cells. The extracted sophisticated features and textures are quintessential for pattern recognition and identification of subtle patterns residing within our measured data. In the following section, we will discuss the application of machine learning using these sophisticated features. Then we show the results that reveal the advantages of these features.

4.3 Constructing the Classification Model

In this section, we will address the creation of our datasets and discuss the sampling techniques applied on these datasets to study whether such techniques have any impact on the classification results. Similarly we discuss the various normalization schemes used. Subsequently we highlight the used feature selection algorithms. Thereafter we provide detail about the evaluation metrics. The last part is about the classifiers considered in the comparison. The results are discussed later in the next section.

4.3.1 Imbalanced Dataset and Sampling Techniques

In order to generate our dataset for the purpose of object pattern discrimination and object recognition, we extracted features to describe the object characteristics and morophology in a more sophisticated way.

In our two experiments performed on *S. cerevisiae* yeast cells, we generate two datasets S_1 and S_2. S_1 consisting of 1440 yeast cell instances belonging to two major classes, one representing cells showing genes tagged with (GFP) reporter and expressing 14-3-3 proteins. The first class of cells is cultivated under low osmotic stress level, i.e. *non-NaCl* medium, and the other representing same cell strains under a high stress level, i.e. *0.5M-NaCl* medium. The second dataset, i.e. S_2, consists of 1380 segmented objects belonging to two major classes, one representing intact yeast cells and the other representing artefacts in the microscope images. After segmentation, all cells are measured for the features mentioned previously in Section 4.2. Each instance I in these two datasets is mapped to one element of the set $(\mathtt{p}, \mathtt{n})$ of positive and negative class labels representing the two different cell classes of low vs. high stress levels in S_1 and intact vs. artefacts in S_2 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 [Faw06].

Our dataset S_1 and S_2 are considered imbalanced since they exhibit an unequal distribution between their positive class, i.e. yeast cells under low osmotic stress in S_1 or intact cells in S_2, and their negative class, i.e. yeast cells under high stress in S_1 or artefacts in S_2 classes. The ratio of positive to negative classes being around $2.7 : 1$ in S_1 and $8 : 1$ in S_2. 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. In conventional evaluation practice, singular assessment criteria, e.g. the overall accuracy or error rate are used. These criteria do 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 and has become apparent in all kinds of datasets.

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 rules 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 well-known sampling techniques, namely under-sampling, over-sampling and Synthetic Minority Oversampling technique (*SMOTE*).

We define subsets $S_{min} \subset S_d$ and $S_{maj} \subset S_d$ where S_d refers to either dataset S_1 or S_2, S_{min} is the set of minority class instances in S_d, and S_{maj} is the set of majority class instances in S_d, so that $S_{min} \cap S_{maj} = \{\phi\}$ and $S_{min} \cup S_{maj} = \{S_d\}$.

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_d so that $|S_d| = |S_{min}| + |S_{maj}| - |E|$, 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_d; 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. 4.9, 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. 4.9 is a point along the line segment joining x_i under consideration and the randomly selected K-nearest neighbour $\hat{x_i}$ [He09]. Applying *SMOTE* to balance both of our datasets increased the accuracy of the classification as we will see in the next section. Moreover, the classification accuracy obtained after applying *SMOTE* significantly outperformed both the under-sampling and over-sampling methods.

$$x_{new} = x_i + (\hat{x_i} - x_i) \times \delta, \quad \text{where } \delta \in [0,1] \tag{4.9}$$

4.3.2 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 [Ver14]. 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) [Štr09]. 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_d in accordance with the expression in Eq. 4.10 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^*.

$$x_i{}^* = \frac{x_i}{||x||}, i = 1,2,...d, \tag{4.10}$$

The zero-mean and unit-variance normalization is defined in Eq. 4.11, 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.

$$x_i{}^* = \frac{(x_i - \mu)}{\sigma}, i = 1, 2, \ldots, d, \tag{4.11}$$

Applying both normalization schemes to scale our dataset improved the accuracy of the classification. More details are illustrated in the next section. The zero-mean and unit-variance normalization outperforms the unit length normalization in both experiment dataset, hence, we adopted the zero-mean unit variance as a normalization scheme for our datasets.

4.3.3 Feature Selection

In machine learning and statistics, dimensionality reduction or dimension reduction is the process of reducing the number of random variables under consideration, and can be divided into feature selection and feature extraction. It is used to assist in the data analysis process.

Feature selection also known as variable or attribute 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 on the testing dataset.
- providing faster and more cost effective classifiers.
- providing a better understanding of the underlying process that generated the data.

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

Next to feature selection is feature extraction, which transforms the data in the high-dimensional space to a space of fewer dimensions. We considered the main linear technique for feature extraction, i.e. the principal component analysis (PCA), which performs a linear mapping of the data to a lower-dimensional space in such a way that the variance of the data in the low-dimensional representation is maximized.

4.3.4 Building a Classifier

The prediction problem for our case study is to predict whether the cells are cultivated under high osmotic stress vs. those under low osmotic stress in dataset S_1 or those intact cells vs. artefacts in dataset S_2. 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 [Kim11]. Over-fitting 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, also known as Type III errors [Don13a]. Its advantage is that all observations are used for both training and validation, and each observation is used for validation exactly once.

4.3.5 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) [Lak11]. 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 [Li14]. 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 incorrectly labeled positive results occur among all negative samples available during the test [Sen13]. Each prediction result or instance of a confusion matrix represents one point in the *ROC* space [Ras13].

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 popular 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 [Faw06]. *ROC* analysis provides tools to select possibly optimal models [Bes10]. 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 [Her11], 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 [Tsa12]. *AUC* is a common statistic most often used for model comparison in the machine learning community [Chi13]. Despite its popularity, some machine learning researches show that the *AUC* is quite noisy as a classification measure [Han10], and has some other significant problems in model comparison [Lob08, Han09]. 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.

4.3.6 Classifiers Evaluated

In this work, 23 different linear and non-linear classification systems were evaluated on our dataset. These systems including the popular predictors such as decision trees, naive Bayes, least-square linear predictors, and support vector mahcines. A complete list of these models along with short descriptions are shown in Table 4.2. We applied a supervised classification on our two datasets S_1 and S_2, with a 10 fold cross validation.

4.4 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. For dataset S_1, we listed in Table 4.3 the *AUC*, A_{min} and *ACC* scores for each classifier under the best sampling, normalization and feature selection algorithms. For dataset S_2, we only listed the top ten classifiers in Table 4.4. Figures 4.3 and 4.4 shows the noticeable difference between *AUC* and A_{min}. This is the rational behind evaluating the *AUC* for the minority class as well as that of the complete dataset.

For our case study, 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 in both datasets S_1 and S_2 with an *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 feature space.

Table 4.2: 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 [Qui93].
Adaptive Boosting (AdaB)	A machine learning meta-algorithm used in conjunction with a weak learner (Decision Tree) to improve its performance [Fre96].
PART	Builds a partial C4.5 decision tree in each iteration and makes the "best" leaf into a rule [Fra98].
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) [Koh95].
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) [Iba92].
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" [Hol93].
JRip	JRip implements a propositional rule learner, Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [Coh95].
Bayes Network (BNet)	Bayes Network learning represents the dataset variables via a directed acyclic graph (DAG) based on probability theory [Fri97].
K-Nearest Neighbour (IBK)	Predicts the class of the single nearest training instance for each test instance [Aha91].
Locally Weighted Learning (LWL)	Uses an instance-based algorithm (Decision Stump) to assign instance weights [Atk96].
LogitBoost (ALR)	Performs additive logistic regression on the base learner (Decision Stump) [Fri98].
Random Committee (RCom)	Builds an ensemble of randomization 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 [Ho98].
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 [Hul01].
Logistic Model Tree (LMT)	A logistic model tree basically consists of a standard decision tree structure with logistic regression functions at the leaves [Lan05].
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 [Bre01].
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 [Ces92].
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 [Pla98].
SimpleLogistic (SLog)	Classifier for building linear logistic regression models. LogitBoost with simple regression functions as base learners is used for fitting the logistic models [Lan05].
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 [Fre98].

4.4.1 Power of Sampling

The first obvious fact from the result in Table 4.3 and 4.4 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* on dataset S_1, 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 demonstrates the reason why accuracy is not a sufficient measure in our imbalanced dataset. Generally, power of sampling is more conspicuous in *PART, C4.5, JRIP, SMO* and *VFDT* classifiers (cf. Fig. 4.5).

On dataset S_2, the power of sampling is more obvious since S_2 exhibits a higher ratio between majority and minority class instances. After applying *SMOTE* on dataset S_2, the *AUC* was improved in average by .127, but the average accuracy decreased slightly by 1.1%.

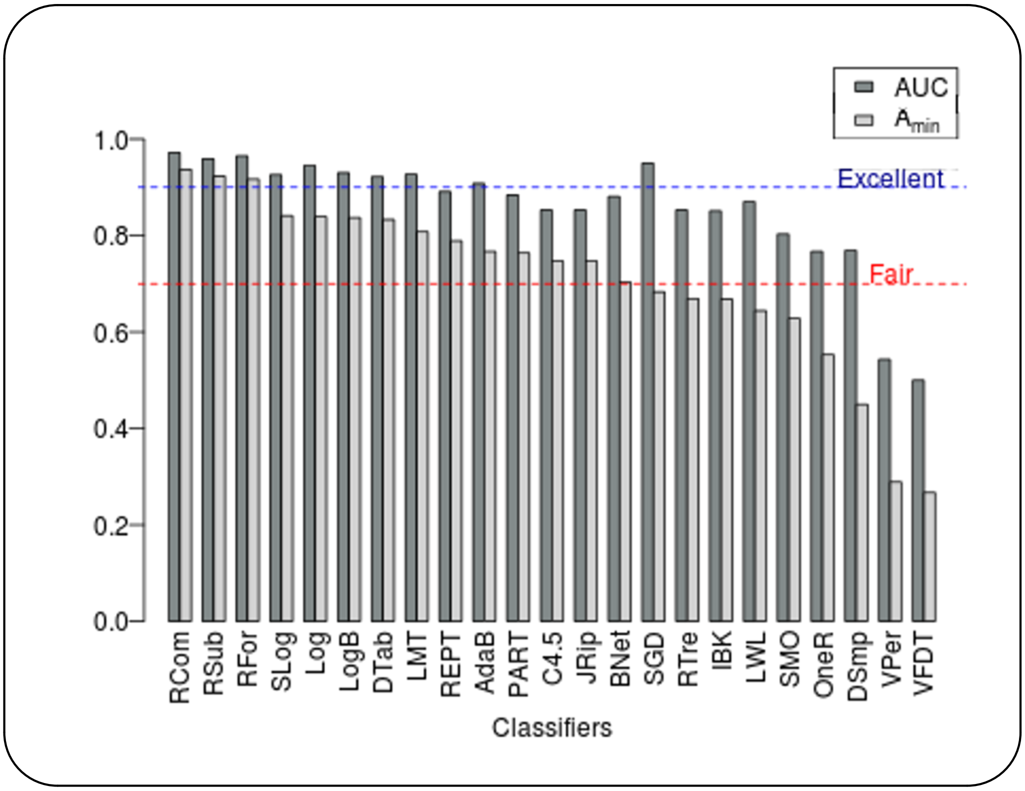

Figure 4.3: *AUC and* A_{min} *of classifiers for raw dataset* S_1.

However, the strongly significant different is shown when considering the *AUC* for the minority class ($\mathtt{A}_{\mathtt{min}}$) where the average improvement was increased by .369 per classifier from an average of .565 (poor classification) to .935 (excellent classification) (cf. Fig. 4.6). This also reveals the reason why accuracy is not a sufficient measure in our imbalanced dataset.

4.4.2 The effect of Normalization and Feature Selection

On dataset S_1, Normalization showed an extra average improvement of .015 and .013 for the *AUC* and $\mathtt{A}_{\mathtt{min}}$ respectively, and improved accuracy by 1.2% over the original dataset. As can be noticed from Fig. 4.7 and 4.8, the most significant difference of normalization was shown in *VPer* and *VFDT* classifiers. Similarly for dataset S_2, Normalization showed an extra average improvement of .007 and .009 for the *AUC* and $\mathtt{A}_{\mathtt{min}}$ respectively, and improved accuracy by 5.8% over the sampled dataset.

The best feature selection algorithm, i.e. *IG*, showed no significant change in the averages of *AUC* and $\mathtt{A}_{\mathtt{min}}$ for both dataset. On dataset S_2, the *AUC* was .949 and $\mathtt{A}_{\mathtt{min}}$ .944. The accuracy metric has a slight extra increase of .03% up to a final average accuracy of 92.2%.

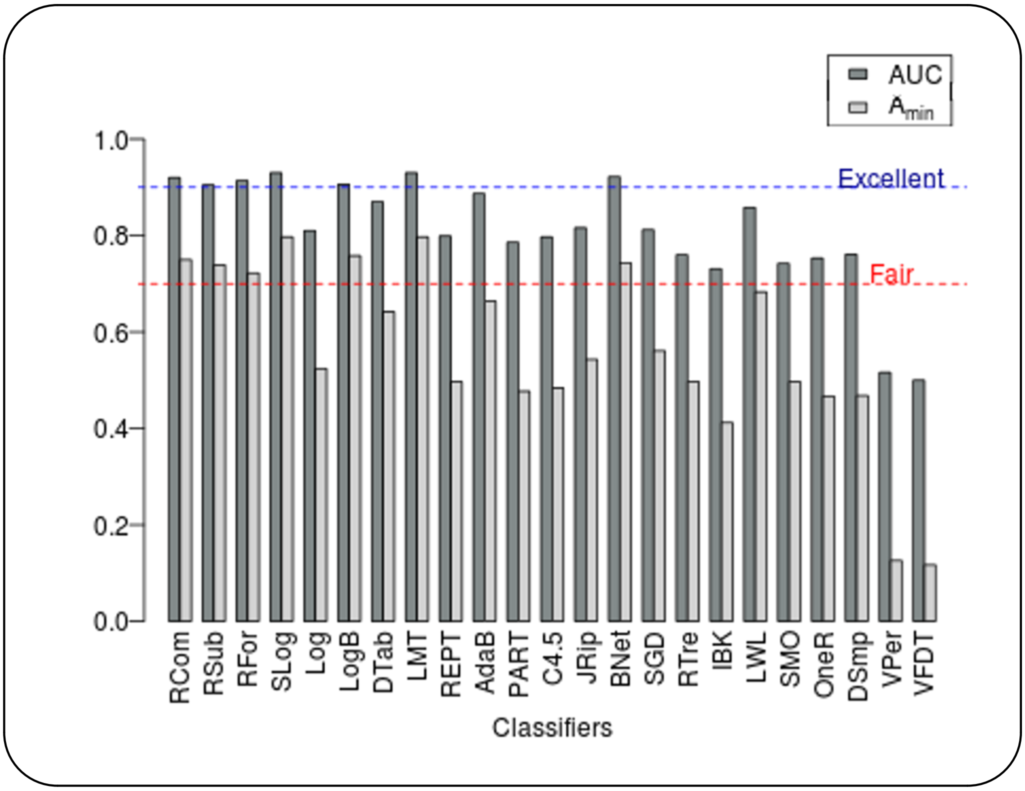

Figure 4.4: *AUC and* $\mathtt{A}_{\mathtt{min}}$ *of classifiers for raw dataset* S_2.

Table 4.3: *AUC*, A_{min} and *ACC* of classification algorithms using raw dataset S_1, 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

Although the effect of feature selection is negligible on the prediction performance of the classifiers, it has two advantages; the first advantage is the provision of faster and more cost effective calssifiers, and the second is the provision of a better understanding of the underlying process that generated the data.

4.4.3 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 fusion of 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 have also played a major role in the discrimination of yeast cells, they showed up in almost every classifier, though at lower level in decision trees or

Table 4.4: Area under *ROC* and Accuracy of the top 10 classification algorithms using raw dataset S_2, and after sampling, normalization and feature selection.

Classifier	Raw Dataset			Sampled		Normalized		Features Selected	
	AUC	AC1	ACC	AC1	ACC	AC1	ACC	AC1	ACC
SLG	.930	.797	.944	.987	.948	.987	.948	.986	.948
LMT	.930	.797	.944	.981	.953	.981	.956	.979	.956
BYN	.922	.743	.918	.974	.930	.971	.927	.971	.927
RNC	.920	.750	.937	.991	.973	.992	.972	.991	.968
RNF	.914	.722	.938	.988	.966	.991	.964	.990	.962
LBS	.906	.758	.942	.975	.920	.975	.919	.975	.919
RSS	.905	.739	.938	.990	.955	.989	.958	.988	.955
BGG	.903	.763	.938	.989	.952	.990	.950	.990	.951
ABM	.887	.664	.930	.966	.910	.967	.903	.967	.903
DTB	.870	.642	.930	.953	.888	.961	.895	.964	.898

with a smaller weight in regression functions.

Considering dataset S_1, 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. Moreover, high odds ratios were noticed in texture measurements and co-occurrence matrix features. 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.

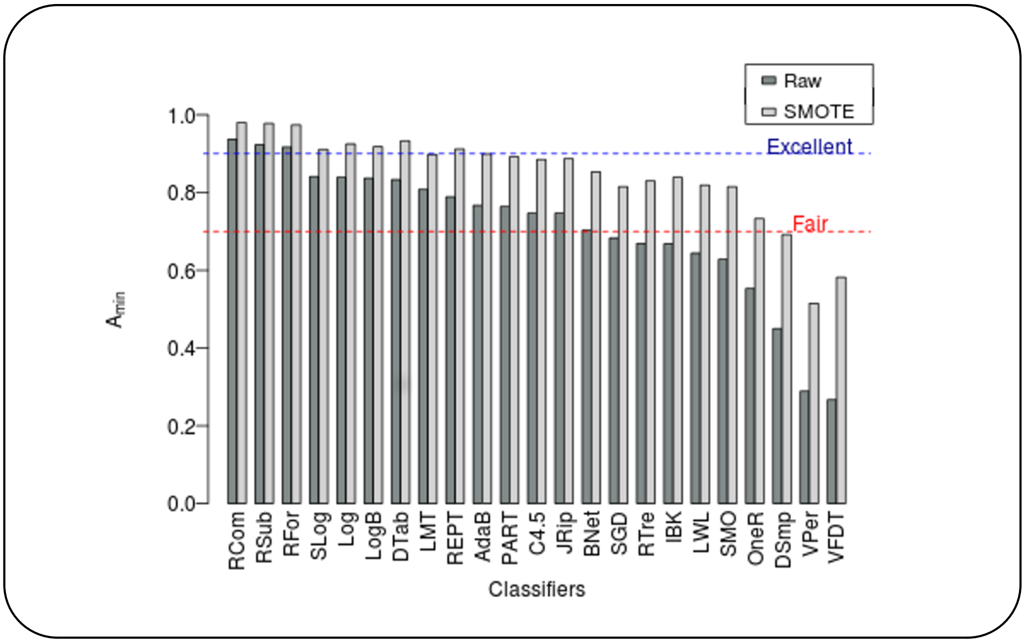

Figure 4.5: *A_{min} of classifiers for raw and sampled dataset S_1.*

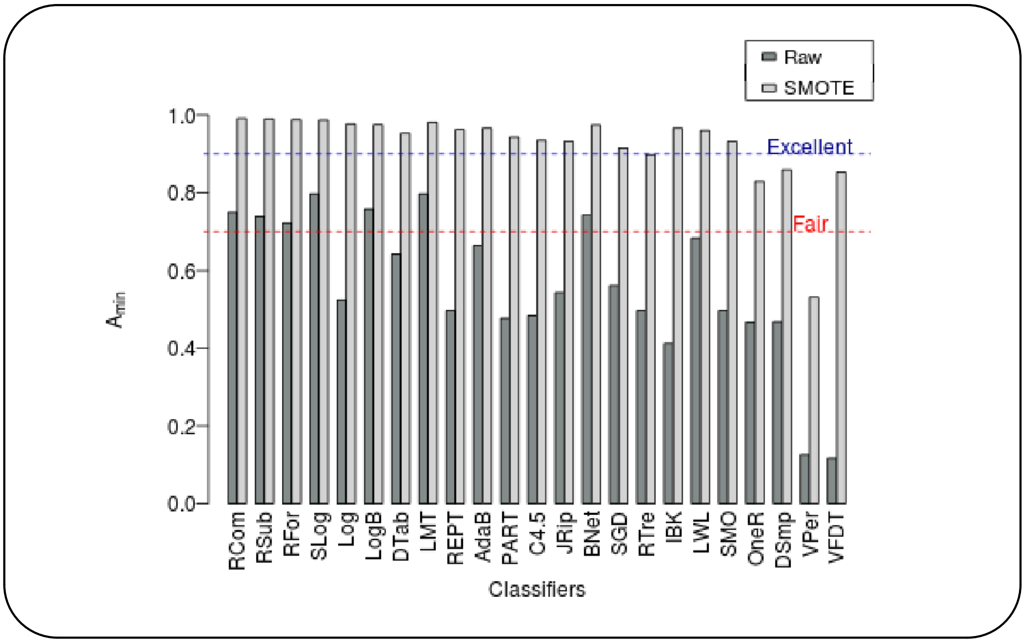

Figure 4.6: *A_{min} of classifiers for raw and sampled dataset S_2.*

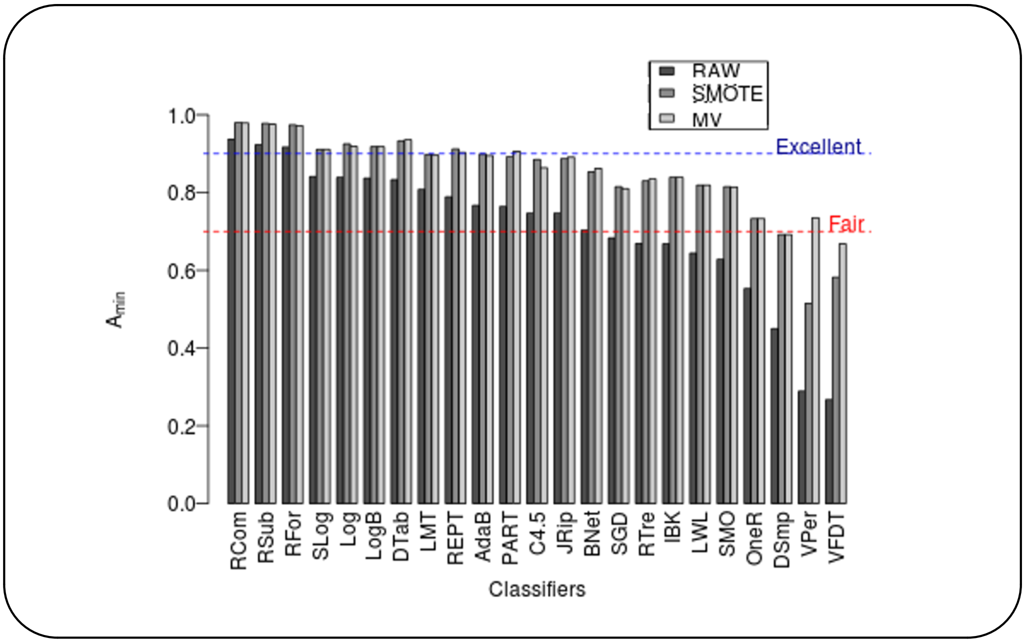

Figure 4.7: A_{min} *of classifiers for raw, sampled and scaled dataset* S_1.

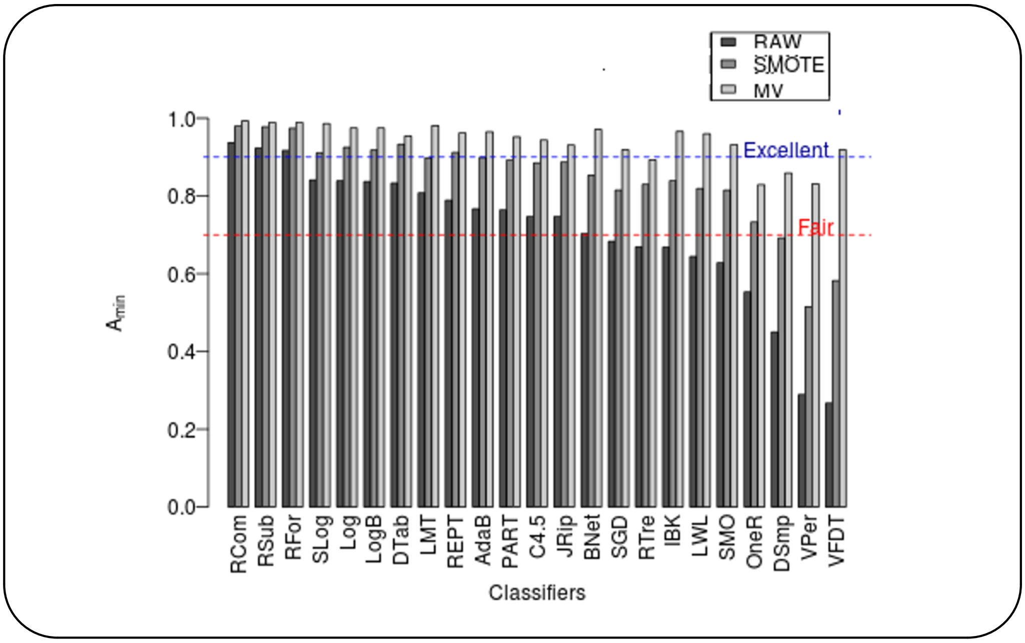

Figure 4.8: A_{min} *of classifiers for raw, sampled and scaled dataset* S_2.

Support Vector Machines (*SVMs*) are popular classifiers. The *SMO*, which is an implementation of the *SVMs*, did not rank as an excellent classifier in dataset S_1 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.

Figure 4.9 shows initially that the *SVM* classifier evaluated as a poor classifier with an $\mathtt{A_{min}}$ value of .628. This value is improved into "good" level after sampling with the *SMOTE* algorithm, i.e. .815. However, optimization pushed the *SMO* into the top five classifiers with an $\mathtt{A_{min}}$ of 0.92.

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

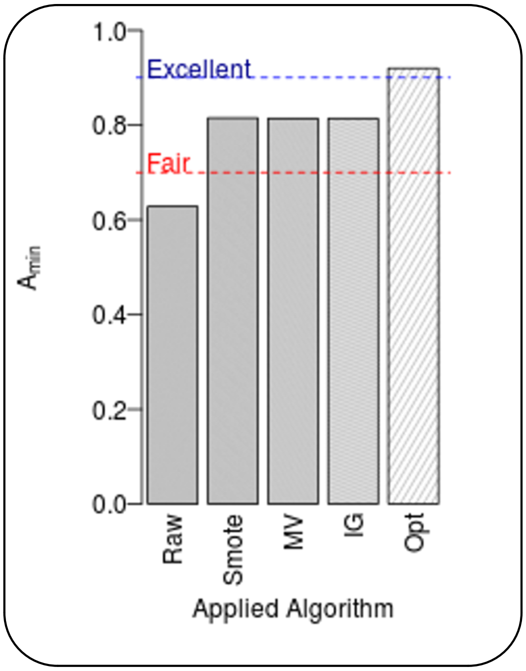

Figure 4.9: $\mathtt{A_{min}}$ *of SMO classifier for raw dataset* S_1*, and after the application of SMOTE, MV and IG algorithms for sampling, scaling and feature selection respectively.*

4.4.4 Feature sets performance

In order to investigate whether our composed feature sets has any added value to the discrimination power, we started by comparing the classification performance on dataset S_1 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. 4.10. 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 0.916. 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 0.914.

Since 3^{rd} order moment invariants are more complex than there 2^{nd} order counterpart, and they might be more noise-prone [Goo96], 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 4.11 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.

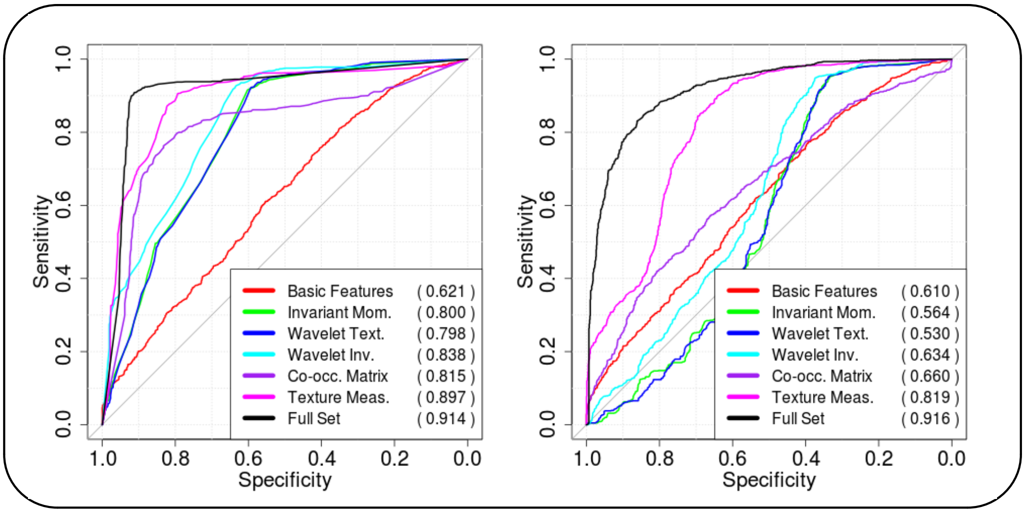

Figure 4.10: *ROC analysis and AUC value of C4.5 (left) and Logistic (right) classifiers using various feature sets*

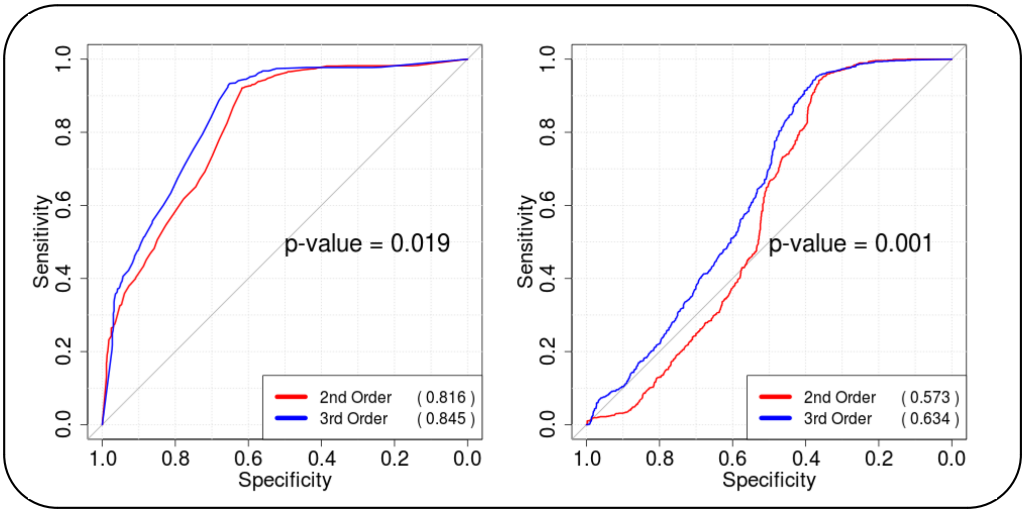

Figure 4.11: *Performance of C4.5 (left) and Logistic (right) classifiers using second and up-to third order moment invariant features*

4.5 Discussion

In this chapter, we addressed our principal research question and showed that a machine learning approach can discriminate *S. cerevisiae* yeast cells to identify essential characteristic differences between two groups treated under different conditions, and a similar approach can improve the object recognition based on its measured sophisticated features. In addition we address the sub question and showed that a combination of various feature extraction methods have a significant role in improving the accuracy of the built classification models.

In the case study of yeast cells, we were able to find a significant difference in the expression of 14-3-3 proteins for cells cultivated under different stress levels. This difference in their characteristics is a subtle pattern that is hard to be noticed using standard methods, such as investigation the measurement data and examining the basic statistical information. On the other hand, the object recognition allow us in our experiment to exclude artefacts prior to data analysis, which consequently improves the identification of subtle patterns.

In this work, we introduced a machine learning workflow to be followed in building a classification model. We showed that the extracted object features explained in Section 4.2 are advantageous to predict the group that an object belongs to. 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. 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. The best feature selection algorithm tested showed a little non-significant improvement; however, it provides a more simple, faster and cost effective classification model. With this machine learning process and the chosen feature sets, it becomes possible as future work, to classify different cell strains and conditions in a high-volume high-throughput studies.