Refining AI Methods for Medical Diagnostics Management

Tapan P. Bagchi

Abstract

This paper extends the utility of support vector machines—a recent innovation in AI being adopted for cancer diagnostics—by analytically modelling the impact of imperfect class labelling of training data. It uses ROC computations to study the SVM's ability to classify new examples correctly even in the presence of misclassified data in training examples. It uses DOE to reveal that misclassifications present in training data affect training quality, and hence performance, albeit not as strongly as the SVM's primary design parameters. Still, our results give strong support for one's striving to develop the best trained SVM that is intended to be utilized, for instance, for medical diagnostics, for misclassified training data shrink decision boundary distance, and increasing generalization error. Further, this study affirms that to be effective, the SVM design optimization objective should incorporate real life costs or consequences of classifying wrongly.

Keywords: Support Vector Machine, Classification, Soft Margins, ROC, Training Data Quality.
Two-class Classification and the SVM

A missed timely detection of cancer may cost one's life, whereas a false classification of a benign case as cancerous would cost $250,000 in agonizing therapies and its psychological effects—for unjust reason. Clinical diagnostics has always depended on the clinician’s ability to diagnose pathologies based on the observation of symptoms exhibited by the patient and then classifying his/her condition. Correct diagnosis can make a big difference in the form of correct and timely intervention, be it hypertension, diabetes, or the various types of malignancy. Similar situations arise where the precise links between cause and effect is not yet established and its management is predestined to process information as best as possible to draw inferences to guide decisions. For hypertension, for example, attempts have been recently made to probe the situation beyond the measurement of systolic/diastolic blood pressures—one finds such studies attempting to predict the occurrence of hypertension based on observations of age, sex, family history, smoking habits, lipoprotein, triglyceride, uric acid, total cholesterol and body mass index, etc. In many such situations, the treatment given is based on a binary classification—the ailment is present, or it is not (Ture et al. 2005).

Classification of a pathology is challenging not only in respect to acquiring the relevant data through tests about factors known to be associated with the pathology, but also the data analytics adopted to lead to reliable and correct prediction. This paper looks into one such data analysis technique, now about 20 years in use and known as support vector machine or SVM, that helps one to develop classification models based on statistical principles of learning. Like artificial neural networks, an SVM is data driven—it is trained using a dataset of examples with known class (label), and then utilized to predict the class of new examples. How well an SVM works is measured by the accuracy with which it can predict the class of unseen examples (examples not included in training the SVM).

The tone set in the present work is to move beyond the simple notion of “accuracy”—the conventional classifier performance measure—by incorporating analytical modelling of correct/incorrect classification of instances in the training sample. This has not yet been done. We focus specifically on the effect on ROC of imperfect labelling of input data.

Support vector machine is an algorithmic approach proposed by Vapnik and his colleagues (Boser et al. 1992) to the issue of classification of instances (for example, patients who may or may not have diabetes) and it falls in the broader context of supervised learning in artificial intelligence (AI) in computer science. It begins with a set of data consisting of feature vectors of instances \( \{x\} \), and a class tag or label \( \{y\} \) attached to each of those instances. The most common application of SVM aims at training a model that learns from those instances, and estimates the model parameters. Subsequently that model is used to predict the class of an instance for which only feature values are available and one is interested in finding its class \( y \) label, with a high degree of correctness. By an elaborate procedure of optimization, an SVM is designed so as to display minimum classification error for unseen instances, an attribute measured by its “generalization error.” Binary classification is its most common use.

Performance Measures for Classifiers

What do we look for in a classifier? We use here a binary classifier to recall the performance measures...
that are commonly used to indicate the goodness of a classifier. Conventionally, in the AI literature, three measures are used, namely, sensitivity, specificity and accuracy (Liyanage 1995). Assume that a finite population of size N is at hand, in the form of a mixture of negative and positive elements. When an imperfect binary classifier is employed to sort the population, it classifies some positives as positives and the count is designated TP (true positive). The balance positive elements it misclassifies as negative, the count being called FN (false negative). Of the negative elements, it correctly classifies some as negatives, that count being TN (true negative) while the balance of the negative elements is misclassified as positive and counted as FP (false positive). This situation is commonly shown as a “Confusion Matrix” (Table 1). Multiclass confusion matrices are shown in Congalton (1991). Before we go further, we note that the reported accuracy of SVMs in recent studies ranges between 70 and 95%, even if it appears to outdo ANN (Noble 2006; Ban et al. 2010).

Table 1. The Confusion Matrix

<table>
<thead>
<tr>
<th>Elements Classified as positive</th>
<th>Elements Classified as negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truly positive elements</td>
<td>TP, FN</td>
</tr>
<tr>
<td>Truly negative elements</td>
<td>FP, TN</td>
</tr>
</tbody>
</table>

With data available as shown in Table 1, the classifier’s performance is defined as follows:

\[
\text{Sensitivity} = \frac{TP}{TP + FN} = \text{fraction of positives classified correctly}
\]

\[
\text{Specificity} = \frac{TN}{TN + FP} = \text{Fraction of negatives classified correctly}
\]

\[
\text{Overall Accuracy} = \frac{TP + TN}{TP + FN + TN + FP} = \text{Fraction of elements classified correctly}
\]

ROC graphs (Fawcett 2003, 2006) form a compact way of conveying the overall performance of a two-class classifier. However, frequently, behind such measures is the goal of causing minimal loss to society, for instance, by detecting cancer when it is present in a patient, and also minimizing the effect of missing a real case of cancer. Naturally, these losses should have become the conventional performance measures for a classifier. Accordingly, paralleling statistical test of hypothesis theory, one might define two more measures—to indicate the extent of errors or misclassifications that an imperfect binary classifier would deliver, when the positive condition (here “cancer is present”) is clearly less desirable. These are

\[
\text{Estimated Probability (α) of committing a Producer’s Error} = \frac{FP}{TN + FP}
\]

\[
\text{Estimated Probability (β) of committing a Consumer’s Error} = \frac{FN}{TP + FN}
\]

The real life consequence or cost of committing such errors can sometimes be enormous. Visualize, for instance, a cancer test missing a TP element (a malignant case of cancer) and calling it benign (FN). No (or delayed) treatment prescribed in such a case can lead to death for the affected. On the other hand, calling a TN (benign) case positive (FP) would likely entail needless medical treatment and avoidable agony and expenditure—but much less serious in consequence than classifying malignant as benign. However, we shall show later that an apparent “high” (~90%) classification accuracy rate (calling a cancer case “malignant”) may appear to be “quite acceptable.”
But this may not be so! Indeed, using Bayesian principles, it may be easily shown that a good classifier must recognize a malignant case as such with much higher accuracy (low β) just as it must also recognize a benign case as “benign” also with much higher accuracy (low α)—particularly when the occurrence of cancer is only 1-2% or less in the general population. Such performance target should apply to the total testing/classification system employed in attempting a diagnosis—planning data acquisition, data sensing, pre-processing input data, data analytics, and the decision models employed to best help utilize the information thus generated. This is already hinted by Ban et al. (2010).

It is also to be noted that mere probability estimations may not suffice in real applications of classification. One would find it much more useful when the real costs associated with false negative (FN—say a positive (malignant) misclassified as negative (benign)) outcomes and those with false positives (FP—negatives misclassified as positive) are incorporated into the analysis of the decision at hand. A broad area of decision analysis is now devoted to this goal (Raiffa and Schlaifer 1961).

ROC (Receiver Operating Characteristics) graphs (Fawcett 2003, 2006) form a compact way of conveying the overall performance of a two-class classifier. Our defining concepts of ROC are drawn from Fawcett (2003). Receiver operating Characteristics (ROC) curves were originally used in signal detection theory to show the tradeoff between hit rates and false alarm rates of two-class classifiers. In recent years, the medical decision making community has extensively engaged in using ROC for diagnostic testing. Supervised learning, a class of methods to which SVM belongs, uses training data to develop input/output relationships, to subsequently develop its class predictive capability of yet unseen examples. Such capabilities of a given classifier in turn, are evaluated using performance measures listed earlier. We return to ROC soon.

**Bayes’ Decision Rule and EVPI**

Why diagnose? Why classify pathologies? An auto engine that cranks and cranks but does not start will in most cases cause inconvenience, delays, etc. A medical condition that seems to be not normal needs attention and treatment to avoid undesirable consequences. There are countless other situations where a decision cannot be avoided or postponed about appropriate action to remedy the malaise. Clearly, fixing the problem sometimes requires sophisticated domain knowledge about cause-effect. But before that step is taken, one would need to identify how the situation at hand is distinct from other situations. As for deciding on the course of action, there is much now available under “Decision Theory” to help us pick the “right” action based on the attributes of the situation—alternatives available, data and information, uncertainties prevailing—and preferences of the decision maker, among other factors (Raiffa and Schlaifer 1961; Hammond et al. 2002). In such cases, many decisions may be made without the aid of pencil and paper. In fact, Drucker has stated, “A decision is a judgment...a choice between alternatives. It is rarely a choice between right and wrong. It is often a choice between two courses of action, neither of which is probably more nearly right than the other” (Edersheim 2007). The right decision, it is experienced, generally has a salutary effect, while the wrong one may prove fatal. While some choice making is subjective, others require quantitative and objective attack.
Decision theory or analysis extends us a method for rational decision making particularly when consequences are not fully known or controllable (Sen 2010). A problem involving choice is characterized typically by (a) the different possible strategies (alternatives) that one may employ, (b) the states of nature—future events that one cannot control, and (c) the consequences—payoffs or losses one would incur by pursuing some particular decision alternative. This is the rational approach to decision making. Payoffs are the consequence resulting from a specific combination of a decision alternative and a state of nature. If this reflects a “cost”, it is noted as a loss. It is often possible to structure such combinations in the form of a table—the payoff table—an example being Table 2. Table 2 displays the payoffs prepared by a construction company that wishes to build an apartment building with capacities of 50, 100 or 150 units while the market comprises a demand for 50, 100 and 150 units with unknown probabilities. The estimated payoffs shown for each “apartment-market” combination are in US$. Broadly, payoffs can be profit, cost, time, distance or any other appropriate value or utility. The prevailing possible states of nature may be known with certainty, or one may have no information on their likelihood. The consequences themselves (the entries in the payoff table) may be known a priori with certainty, or be probabilistic. This issue of uncertainty significantly impacts the decision making process itself.

Table 2: Payoff Table for a Construction Company facing Alternative Project Decisions

<table>
<thead>
<tr>
<th>Project Alternatives</th>
<th>State of Nature</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Market Demand for Number of Units</td>
</tr>
<tr>
<td></td>
<td>Low (50 units)</td>
</tr>
<tr>
<td>Build 50 units</td>
<td>400,000</td>
</tr>
<tr>
<td>Build 100 units</td>
<td>100,000</td>
</tr>
<tr>
<td>Build 150 units</td>
<td>(200,000)</td>
</tr>
</tbody>
</table>

When the likelihood of the states of nature is unknown, the common approaches are three: (a) The optimistic approach is used by those with an optimistic outlook—he/she chooses the alternative with the largest possible payoff. For Table 2, the optimistic decision would be to “build 150 units; ignore any downside risk of loss due to unoccupied units”. (b) A conservative decision maker would face uncertainty so as to maximize the minimum payoff from each alternative. If a cost table is provided in place of the payoff table, the maximum cost is determined for each alternative. The alternative with the minimum of these maximum costs would be selected. For Table 2, the conservative decision is to select the project “build 50 units; this will take care of the worst case scenario and assure a protected minimum payoff of 400,000”. Lastly, (c) adopts the minimax regret approach. The choice is made here by calculating the “regrets” (the opportunity to be missed) for each state of nature—the difference between each payoff and the largest possible payoff for that state of nature. One then chooses the alternative with the minimum of these maximum regrets. For Table 2, the minimax regret choice is “build 150 units; this will minimize the regret of doing a project but not earning enough that one could have.” The key supporting these decisions is
that even though the probabilities for the states of nature are unknown, the payoff for each “alternative-state of nature” combination is known in actual $ terms.

Decisions with probabilities (and therefore the risk) for the states of nature known (or estimated) adopt other procedures, the dominant one being the Expected Value (EV) approach. In EV, the expected return (payoff) for each decision is calculated by summing the products of the payoff under each state of nature and the probability of the respective state of nature occurring. One then selects the alternative yielding the best expected return, EV being evaluated as -

$$EV(d_i) = \sum_{j=1}^{N} P(s_j)V_{ij}$$

where $N$ = the number of the states of nature, $P(s_j)$ = the probability of the state of nature $s_j$ and $V_i$ = the payoff corresponding to decision alternative $d_i$ and state of nature $s_j$. For Table 2, the EV approach would require us to provide the probability of encountering each of the three possible states of nature. These are estimated as 0.2 for low market demand, 0.5 for medium market demand and 0.3 for high demand. Given these, the EV decision strategy would select “build 100 units” as the best strategy, for this has the highest expected value ($EV = $100,000×0.2 + $800,000×0.5 + $800,000×0.3 = $660,000). One witnesses here the superior rationale and ease of executing the decision making process when explicit payoff values are utilized.

Yet another related approach—known as EVPI (Expected Value of Perfect Information)—is available to assess if the occurrence of the state of nature were perfectly and exactly known, and how much improvement would be expected in the payoffs. This is especially useful to know if more resources could be marshalled to acquire better estimates of the probabilities used in calculating EV—approaching perfect prediction of these probabilities. EVPI is the increase in the expected payoff that would result if one knew with certainty which state of nature would actually occur. Thus EVPI provides the upper bound on EV of any information seeking survey. The steps to find EVPI are as follows:

Step 1: Find the best possible payoff for each state of nature. For Table 2, these are 400,000 for low demand, 800,000 for medium demand and 1,200,000 for high demand market conditions respectively. This shows the highest payoff you would get from each decision alternative if you knew which state of nature was actually present.

Step 2: Compute the expected value of these respective best possible payoffs (achieved by selecting the most profitable alternative) for each state of nature utilizing the probabilities (here 0.2, 0.5 and 0.3 respectively) for the three possible states of nature. For our example, it is ($Expected payoff from best decisions = $400,000×0.2 + $800,000×0.5 + $1,200,000×0.3 =) $840,000.

Step 3: Subtract EV (= $660,000) found earlier from the Expected payoff from best decisions (= $840,000) calculated in Step 2 above. This is EVPI (= $840,000 - $660,000 = $180,000) for this problem.

Again, we note that we have considerable more insight into the consequence of our decisions when we are dealing with payoffs rather than only probabilities. Think of this for the situation when an imperfect classification of a patient transmits the statement,
“our test is ~90% correct”. Being told of only this, one is generally unable to intuit his/her chances of being free from ailment.

How good or useful really is a classification or a pathological test that is “90% correct?” Many textbooks work this out when describing Bayesian methods of estimating conditional probabilities. We recall here the salient points to highlight the core of this enigma.

**The Cancer Test Enigma**

A medical test is used to check cancer. This test has a known reliability:

\[
P(\text{Test +ive /person has cancer}) = 0.92 \\
P(\text{Test +ive /person is healthy}) = 0.04
\]

We know further that cancer is rare and in the general population \(P(\text{cancer}) = 0.001 = (0.1\%)\). If a person is randomly selected and his test is +ive, what is the chance that he has cancer?

The answer to this question may be found as follows.

We use here a simple result from conditional probability. If \(A\) and \(B\) are events with \(Pr(A) > 0\), the *conditional probability of \(B\) given \(A\)* is:

\[
Pr(B|A) = \frac{Pr(A \cap B)}{Pr(A)}
\]

From the data provided, \(Pr(\text{person has cancer}) = 0.001\), hence \(Pr(\text{person is healthy}) = 0.999\)

Using the data \(Pr(\text{Test +ive / person has cancer}) = 0.92\), and \(Pr(\text{Test +ive / person is healthy}) = 0.04\) we get:

\[
Pr(\text{person has cancer | Test +ive}) = \frac{Pr(\text{Test +ive / person has cancer}) \times Pr(\text{person has cancer})}{[Pr(\text{Test +ive}) \times Pr(\text{person has cancer}) + Pr(\text{Test +ive / person is healthy}) \times Pr(\text{person is healthy})]}
\]

\[
= \frac{0.92 \times 0.001}{0.92 \times 0.001 + 0.04 \times 0.999}
\]

It may be verified that \(Pr(\text{person has cancer | Test +ive}) = 0.022505\)

Clearly, this indicates that the chances are only 2% that a person who is tested +ive by this medical test actually has cancer! Quite a frightening conclusion! The test being “90% correct” would even be worse.

What this is implying is that a classifier or a test that declares a person to be suffering from cancer on average 90% of the time has only 2% chance of being correct. This is because in real life, only a small fraction (1 in a thousand or 0.1%) of the population suffers from cancer (Liyanage 1995, Patrick 2013).

**The SVM Classification Methodology**

Since 1980, as the power of computing began to grow, automated learning aimed at modelling and...
understanding relationships among a set of variables derived from objects drew much interest (Soman et al. 2011). The goal became that of using supervised learning to model the relationship between some selected inputs and outputs. Artificial Neural Nets (ANN) and Support Vector Machines (SVM) are two such devices created in that period and these continue even today as state-of-the-art classification methods. Of late, in the last twenty or so years, SVM has been extensively used to target problems of classification where an input-output training dataset is presented to the algorithm, which in turn, when its learning is complete, becomes capable of classifying yet new input data. The first such procedure was designed and presented by Vapnik and his colleagues (Boser et al. 1992). Most work on SVM and its applications have focused on the two-class pattern classification problem (El-Naqa 2012).

Briefly, the two-class SVM classifier may be described as follows, though comprehensive references on it are already extensive (Vapnik 1998; Han and Kamber 2006; Ng 2013; Ben-Hur and Weston 2010). This summary is based on Ben-Hur and Weston’s “User’s Guide.”

Let vector x of inputs be a pattern that we need to classify and let y (a scalar) denote its assigned class label, ±1. Let (x, y), i = 1, 2, ... l be the training examples based on l patterns classified earlier by examining each example and tagging or labelling it as “+1” or “-1” earlier. The SVM’s learning task then becomes constructing the classifier or a decision function f(x) that would be able to correctly classify a new input pattern x not included in the training set. Such classifiers may be linear, or nonlinear.

If the training dataset is linearly separable, there will exist a linear function or hyperplane of the form -
\[ f(x) = w^T x - \gamma \]  (1)
such that for each training example x, the function yields f(x) = 0 whenever y = +1, and f(x) < 0 when y = -1. Thus the training data is separated by a function f(x) = w^T x = 0, the equation representing the hyperplane in the x space. While there may be many such hyperplanes existing that can achieve such separation of x, SVM aims at locating the hyperplane that maximizes the separation between the two classes of x it creates. Mathematically, this is achieved by finding unit vector w that minimizes a cost function
\[ \frac{1}{2} w^T w = \frac{1}{2} \| w \|^2 \]
subject to the separability constraints -
\[ y_i (w^T x_i - \gamma) \geq 1; i = 1, 2, 3, ..., l. \]  (2)

Sometimes the training data is not completely separable by a hyperplane. In such situations a slack variable \( \xi \) is added to relax the strict separability constraints in (2) as follows:
\[ y_i (w^T x_i - \gamma) \geq 1 - \xi_i; \xi_i \geq 0; i = 1, 2, ..., l \]  (3)

The new cost function that now must be minimized becomes-
\[ J(w, \xi) = \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{l} \xi_i \]  (4)

Vapnik called C a user-specified, positive “regularization” parameter. In the general sense, not all situations comprising training examples \( \{(x_i, y_i), i = 1, 2, ... l\} \) can be effectively modelled by the linear relationship (1), for the relationship may be nonlinear. To handle these, SVM utilizes kernels—functions that can easily compute dot products of two vectors, a key requirement to achieve computational efficiency (Ng 2013).
In (1) \( w \) is a weight vector and \( b \) is the bias. The hyperplane \( (x: f(x) = w^T x - y) \) divides the input space of \( x \) into two and the sign of \( f(x) \), the discriminant function of the classifier, denotes the side of the hyperplane a point \( x \) is on. The decision boundary is the demarcation between the two regions classified as positive and negative. When the boundary is a linear function of the inputs, it is called a linear classifier. In general, this boundary can be nonlinear. (We do not elaborate this here, but kernel functions may be utilized when the input space cannot be separated by a hyperplane due to nonlinear relationships between input and output present. See Boser et al. 1992.)

If we assume that the input data space spanned by \( x \) is linearly separable, a linear decision boundary (a hyperplane) exists in it. Indeed, many such hyperplanes may exist. The goal of SVM learning is to use the input data to design an optimum hyperplane \( f(x) \) that will maximize the geometric distance (the “margin”) between the examples in the two classes. This is achieved as stated earlier by finding unit vector \( w \) that minimizes the cost function \( \frac{1}{2} w^T w = \frac{1}{2} \| w \|^2 \) subject to the separability constraints -

\[
y_i (w^T x_i - y) \geq 1 \quad i = 1, 2, 3, ..., l.
\]

These constraints here ensure that the classifier \( f(x) \) classifies each example \( x \) correctly. Under the just stated assumption of linear separability being possible, the hard margin SVM (Figure 1) can be constructed to help classify unseen examples. Note that \( y \) computed once (4) has been minimized (Soman et al. 2011). Mathematically this problem is one of optimization:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \| w \|^2 \\
\text{using} & \quad w, y \\
\text{subject to} & \quad y_i (w^T x_i - y) \geq 1 \quad i = 1, 2, 3, ..., n
\end{align*}
\]

\[
(5)
\]

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{The Hard Margin SVM in the \((X_1, X_2)\) feature space}
\end{figure}

**Soft Margin SVM**

The hard margin classifier gives us the narrowest margin—having minimized \( \frac{1}{2} \| w \|^2 \).

It works well and is able to classify new examples with good accuracy when the data is linearly separable. Sometimes the examples are not linearly separable—due to some nonlinearity present or the data being noisy. The hard margin SVM often exhibits low accuracy in predicting the class of new examples in such situations. In that case, a classifier with wider margin may be designed to improve prediction, if we allow the classifier function \( f(x) \) to misclassify (by design) a few training examples (Ben-Hur and Weston 2010).
Important observations

Note that the margins in the hard margin case are given by (2), written as:

$$y_i(w^T x_i - \gamma) \geq 1; \ i = 1, 2, 3, \ldots, l.$$  

With the introduction of the slack variables (\(\xi_i \geq 0\)), these constraints change to (6). Thus, the margin imposed constraints in this case become:

$$y_i(w^T x_i - \gamma) \geq 1 - \xi_i; \ \xi_i \geq 0; \ i = 1, 2, 3, \ldots, l.$$  

When the value of the slack variable (also known as the margin error) \(\xi_i\) for any example \(i\) appears in the range \(0 \leq \xi_i \leq 1\), the example \(x_i\) falls inside the margin bordered by the decision boundaries but has a margin error \(\xi_i\) and a margin error cost \(C \xi_i\) associated with it (see (7)). All \(x_i\) examples with \(0 \leq \xi_i \leq 1\) form support vectors of the soft margin SVM. The decision boundaries continue to be:

$$f(x_i) = w^T x - \gamma = 1 \text{ and } f(x_i) = w^T x - \gamma = -1$$

If \(x_i\) is such that for it

$$\xi_i > 1,$$

the example \(x_i\) is misclassified

and for it the corresponding cost (penalty paid due to misclassification) is \(C \xi_i\) (see (7)).

Cortes and Vapnik (1995) provided the following interpretation and use of the penalty cost or soft margin constant \(C\). When \(C\) is very large, the soft margin SVM behaves like a hard margin SVM, not allowing any points inside the margin for a linearly separable set of examples. When \(C\) is decreased, the margin expands in width, allowing some points with \(0 \leq \xi_i \leq 1\) to get inside the soft margin, i.e., to fall inside the margin (with a corresponding penalty). If the optimum solution resulting from the minimization of \(J(w, \xi)\) results in some \(\xi_i > 1\), those examples are misclassified (see Figure 3).
Cortes and Vapnik suggested that when soft margins are used, several different values of $C$ should be tried to help design the best soft margin SVM—known as CSVM—one that maximizes accuracy while classifying new examples (not included in training) by allowing some misclassifications at a penalty.

A significant variation of the single soft margin constant ($C$) model (CSVM) was subsequently suggested (Davenport 2005; Bach et al. 2007; Lavindrasana et al. 2007; Brandon et al. 2010; Hua et al. 2011; Cohen et al. 2003) to tackle situations when the examples dataset is unbalanced, in which one class contains a lot more examples than the other. In this case, the standard notion of overall accuracy (see Section 2) is not a good way to measure the success of a classifier (Ben-Hur and Weston 2010).

To correct this, one needs to assign different costs for misclassification to each class. The margin error cost and the misclassification cost component in $J(w, \xi)$ in (7), i.e., $\sum_{i=1}^{l} \xi_{i}$ is replaced by two terms, one for each class for which the margin is penetrated or the example is misclassified, as follows.

$$C \sum_{i=1}^{l} \xi_{i} \rightarrow C_{+} \sum_{i=1}^{l_{+}} \xi_{i} + C_{-} \sum_{i=1}^{l_{-}} \xi_{i},$$

when $C_{+}$ and $C_{-}$ are soft margin constants respectively for the examples labelled as positive and negative respectively and $l_{+}$ and $l_{-}$ are the example set’s positive and negative examples.

At this point, an important assumption is imposed by common practice, rationalizing “imbalance” (Davenport 2005; Cohen et al. 2003). To set the relative values for $C_{+}$ and $C_{-}$ it is assumed that the number of misclassified examples in each class is proportional to the number of examples in each class. Then $C$, and $C_{n}$ are chosen such that $C_{+} n_{-} = C_{-} n_{+}$.

Figure 3: One example allowed inside the Soft Margin and one Misclassified

Hence,

$$\frac{C_{+}}{C_{-}} = \frac{n_{-}}{n_{+}},$$

when $n_{+}$ and $n_{-}$ are the number of positive and negative examples in each class in the training data. Thus, in the minimization of the objective function -

$$j(w, \xi) = \frac{1}{2} \|w\|^2 + C_{+} \sum_{i=1}^{l_{+}} \xi_{i} + C_{-} \sum_{i=1}^{l_{-}} \xi_{i} \quad (8)$$

and the subsequent search for the best value of the soft margin constants $C_{+}$ and $C_{-}$ that would maximize overall accuracy of the SVM, we need to adjust only one parameter ($C_{+}$ or $C_{-}$), the other being found using the ratio of $n_{-}$ and $n_{+}$.

Notice that in all of this, we have consistently aimed at accuracy—a fraction or quantity that indicates how accurately the SVM will be able to predict the classes of new examples presented to it. There has been no attention given so far to the consequence of misclassifications—the impact of the fraction ($= 1 – $ accuracy) of examples that get misclassified.
A more realistic version of CSVM

The approach this present work proposes is to change the manner in which the relative values for $C_+$ and $C_-$ are set before (8) is executed. Simply stated, we impose penalties for misclassification or margin errors based on the *losses* (negative payoffs) each type of misclassification or margin error would cause (see Figure 3). For instance, if the onset of cancer is misclassified, the eventuality is either delayed treatment, or death, the value of life for a living person in the USA being held today being about 5 million dollars. The cost of needless treatment when there is no cancer present but SVM tags it a malignancy is about $250,000 per case. Hence in this case, $C_+ = \frac{5,000,000}{250,000} = 20$

Accordingly, the objective to be minimized, i.e.,

$$J(w, \xi) = \frac{1}{2} \|w\|^2 + C_+ \sum_{i=1}^{l^+} \xi_i + C_- \sum_{i=1}^{l^-} \xi_i$$

becomes

$$J(w, \xi) = \frac{1}{2} \|w\|^2 + 20 \sum_{i=1}^{l^+} \xi_i + C_- \sum_{i=1}^{l^-} \xi_i$$

The single model parameter $C_+$ is now to be adjusted so as to produce maximum accuracy. Note that here we have now let a process based on decision theoretic notions (payoffs) help set the relative values of $C_+$ and $C_-$, rather than the speculated “equal likelihood” of the occurrence of misclassifications in each class as done in CSVM. A lot of “noise” in the input data enters through imperfect tests. In testing for cancer, for instance, methods now vary from manual lump detection to mammography to biopsy to genetic mutation screening (Stanford Medicine 2013). Therefore, in learned view, using payoffs or losses is a more realistic basis for setting the asymmetric boundaries of the soft margin SVM (Lachiche and Flach 2003).

### Table 3: Input data for a Classification problem to be learned by a Hard Margin SVM

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>-1</td>
</tr>
<tr>
<td>4.5</td>
<td>4.5</td>
<td>-1</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>-1</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>-1</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>-1</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>-1</td>
</tr>
<tr>
<td>1.5</td>
<td>1.5</td>
<td>-1</td>
</tr>
</tbody>
</table>

**Analytical Results with Graphical Displays**

We now work out examples of classification by using a simple hard margin classifier originally coded in Excel® by Soman et al. (2011). Subsequently we extend Soman et al.’s code to handle the *asymmetric* soft margin SVM. Notations used by Soman et al. are closely followed to permit ready cross-referencing.

The classification problem tackled here is orated to be linearly separable and kept close to the examples in Chapter 4 of Soman et al. (2011).

The first problem is setting up a hard margin SVM using the data in Table 3 that we are told is linearly separable. Note that this problem has two input variables $x_1$ and $x_2$ for each data instance, and a class indicated by $y$.

The hard margin SVM’s decision function (1) is obtained by stepping through computations, leading to the optimization problem (5). Cost $C$ here was taken to be 100 and kept fixed. Minimization of $\frac{1}{2} \|w\|^2$ was done by Solver® built in Excel® —by manipulating parameters $w_1$ and $y$. Table 4 displays the calculations and the optimized $w_2$ and $y$ (gamma), the classes obtained by SVM, along with the performance of the hard margin SVM. Figure 4 shows misclassified instances.
This hard margin SVM was trained using the data in Table 3. Clearly, the hard margin forced certain input examples to be misclassified—accuracy here was 0.5 or 50%. Note further that no true positive examples were misclassified as negative—FN and therefore Type II error of this SVM is 0. However, all truly negative examples were misclassified as positive—these are the False Positive (FP). Part of this problem could be nonlinearity in that the relationship between \((x_i, x_j)\) and class tag \(y\) in Table 3 is nonnegative.

This could also be caused by noise in measuring or preprocessing \(x_i, x_j\) and the label \(y\) in preparing Table 3. Part of this problem, it is suggested, can be tackled by introducing kernel functions that transform the input data into their transformed values when the relationship between \(x\) and \(y\) is nonlinear (Boser et al. 1992). It is also probable, as Vapnik (2008) suggested, that we have not used here the “best” value for \(C\), the “regularization parameter”, to minimize (4), the hard margin SVM. Here we noted that varying \(C\) from \(10^{-12}\) to \(10^{12}\) made no difference in the SVM’s performance.

### Table 4  Hard Margin Classification of the input data in Table 3

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(y)</th>
<th>Maximum Margin Hyperplane</th>
<th>Upper Bounding Plane</th>
<th>Lower Bounding Plane</th>
<th>(w_1)</th>
<th>(w_2)</th>
<th>(y)</th>
<th>Data Classification by SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Misclassified</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>-1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Misclassified</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>-1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Misclassified</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>-1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Misclassified</td>
</tr>
<tr>
<td>4.5</td>
<td>4.5</td>
<td>-1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Misclassified</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Correct</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Correct</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Correct</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Correct</td>
</tr>
<tr>
<td>1.5</td>
<td>1.5</td>
<td>1</td>
<td>-8.41597E+14</td>
<td>2.4293356</td>
<td>-1.683E+15</td>
<td>0</td>
<td>1.18822E-15</td>
<td>-1</td>
<td>Correct</td>
</tr>
</tbody>
</table>

Figure 4  Data classification by the Hard SVM

Table 5  Performance of the Hard Margin Classifier of the data in Table 3

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>(\alpha)</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>(\beta)</th>
<th>(P(\text{Type I error}))</th>
<th>(P(\text{Type II error}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>((TP+TN)/(TP+FN+TN+FP))</td>
<td>(TP/(TP+FN))</td>
<td>(TN/(TN+FP))</td>
<td>(TP/(TP+FP))</td>
<td>(FN/(TP+FN))</td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

\(TP\) = True Positive, \(TN\) = True Negative, \(FP\) = False Positive, \(FN\) = False Negative.
The Excel code was subsequently extended to tackle the asymmetric soft margin SVM design problem. The classification problem continued to be the same (Table 3)—train an SVM (here with asymmetric soft margin) to help classify examples in the $x_1$-$x_2$ domain. For optimization, (7) was replaced by (8) with sets $l^+$ and $l^-$ identified in the training data set (Table 3), the two sets of slack variables ($\xi$) belonging to $l^+$ and $l^-$ introduced and penalties $C_+$ and $C_-$ specified.

Table 6 displays the intermediate calculations and the optimized $w_1$, $w_2$, and $\gamma$ (gamma), the classes obtained by the ASVM, along with the performance of the asymmetric margin SVM. The particular situation displayed used the following parameters: $C^+ = 50$, $C^- = 50$. Figures 5 and 6 graphically display the examples, as well as the two misclassified instances, in the $x_1$-$x_2$ space.

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$\gamma$</th>
<th>MM Hyperplane</th>
<th>Upper Bounding Plane</th>
<th>Lower Bounding Plane</th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$\gamma$ (gamma)</th>
<th>$\xi$ Values</th>
<th>ASVM classification Check: $\xi_i (\xi) &lt; 1?$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>4.999999599</td>
<td>7.9999993</td>
<td>1.99999958</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>0</td>
<td>Correct</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>-1</td>
<td>3.999999792</td>
<td>6.9999995</td>
<td>1.00000007</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>0</td>
<td>Correct</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>-1</td>
<td>4.999999599</td>
<td>7.9999993</td>
<td>1.99999958</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>7.1696E-08</td>
<td>Correct</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>-1</td>
<td>3.999999792</td>
<td>6.9999995</td>
<td>1.00000007</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>0.333333341</td>
<td>Correct</td>
</tr>
<tr>
<td>4.5</td>
<td>4.5</td>
<td>-1</td>
<td>1.500000277</td>
<td>4.5</td>
<td>-1.4999994</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>0</td>
<td>Correct</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>1</td>
<td>2.00000018</td>
<td>4.9999999</td>
<td>-0.9999995</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>0.333333332</td>
<td>Correct</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>1</td>
<td>2.00000018</td>
<td>4.9999999</td>
<td>-0.9999995</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>0</td>
<td>Correct</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>1</td>
<td>1.000000374</td>
<td>4.0000001</td>
<td>-1.9999993</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>3.21121E-08</td>
<td>Correct</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1</td>
<td>1.000000374</td>
<td>4.0000001</td>
<td>-1.9999993</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>0</td>
<td>Correct</td>
</tr>
<tr>
<td>1.5</td>
<td>1.5</td>
<td>1</td>
<td>4.499999696</td>
<td>7.4999994</td>
<td>1.49999997</td>
<td>0.3333333</td>
<td>0.333333364</td>
<td>1.999999987</td>
<td>1.999999991</td>
<td>Misclassified</td>
</tr>
</tbody>
</table>

Figure 5  Data instances correctly classified and those misclassified by the Hard SVM
Table 7  Performance of the Asymmetric Soft Margin Classifier on the data in Table 3; $C^+ = 90$, $C^- = 10$

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>$0.5 = \frac{(TP+TN)}{(TP+FN+TN+FP)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>$1 = \frac{TP}{(TP+FN)}$</td>
</tr>
<tr>
<td>Specificity</td>
<td>$0 = \frac{TN}{(TN+FP)}$</td>
</tr>
<tr>
<td>Precision</td>
<td>$0.5 = \frac{TP}{(TP+FP)}$</td>
</tr>
<tr>
<td>$P(\text{Type II error}) = \beta$</td>
<td>$0 = \frac{FN}{(TP+FN)}$</td>
</tr>
</tbody>
</table>

Hyperparametric Analysis

Table 7 shows the performance of the ASVM that used $C^+ = 90$ and $C^- = 10$. It has been shown that an SVM’s performance would depend on the model parameters selected to design the ASVM. For Table 7, the penalty ($C^+$) for misclassifying a positive example as negative (causing an FN) is nine (9) times more than the penalty ($C^-$) for misclassifying a negative example as positive (FP). Hence, not surprisingly, the classifier produced no FN, and all negatives are classified as positive (FP = 5).

Once the model was set up, it was exercised to check if the ASVM’s performance changed in directions as expected. Table 8 displays some sensitivities. It is apparent that as penalty $C^+$ for the SVM’s misclassifying a true positive example as "negative" (FN) goes up relative to $C^-$ (the cost of the SVM’s misclassifying a negative example as positive), Type II error ($\beta$)–the likelihood of classifying a positive as negative–goes down.

In this case, the chance ($\alpha$) of misclassifying a negative case as positive went up. This again impels one to pursue the decision theoretic premise–rather than merely find chance probabilities. It is certainly more rational to ask, in the domain in which the classifier’s verdict is to be used, what would be the consequent costs/payoffs?

Figure 6  Asymmetric SVM Classification of data instances in Table 3 with $C^+ = C^- = 50$

Clearly, knowing mere chances here would not suffice and one might have to even backtrack to the decision maker’s orientation toward risk and consider optimistic, minimax loss, minimax regret, or still other strategies, including the utility of the decision outcomes (Hammond et al. 2002). An SVM’s misclassifying a malignant (positive) case benign (negative) can never be so serious.
A perspective of classifier performance

Can we estimate the expected value or payoff (EVPC) of a perfect classifier? As designed, a perfect classifier produces no misclassifications; thus, it is able to prevent/avoid all costs consequent to using the classification the SVM delivers. The outcome of the ASVM as displayed by Table 8 can indeed be quite sensitive to the costs used in the objective function (for instance the penalties in (8)). It is also to be noted that the geometric component of (8), namely $\frac{1}{2} \| w \|^2$ reduces in significance. In fact, the role of $\frac{1}{2} \| w \|^2$ is to help minimize generalization error— the ability of the SVM to correctly classify a new case presented to it (Han and Kamber 2006). When costs of relatively high magnitude enter (8), the optimization objective gets dominated by the cost or penalty terms in it. Therefore, it is to be easily appreciated that EVPC equals all costs that are precluded by the perfect classifier.

As the state of the art goes, every SVM owns two significant characteristics. Like many other classification methods, SVM is data driven. Furthermore, it is non-parametric—it does not require one to make any assumptions about the distribution of the data in the population from which it is collected. We note that nonparametric methods are also the mainstay in classical applications of statistical analysis where for various reasons it is not advisable or tenable to assume that the data has come from a Gaussian distribution. But by giving up the parametric scaffold, one is often forced to adopt techniques; several nonparametric tests have less statistical power ($= 1 – \beta$), i.e., a lower ability to reject a hypothesis when it is not tenable (Feller 1968).

Still another issue remains an intrigue with the SVM, the method being data driven (Nguyen et al. 2009; Bi and Zhang 2010). How is SVM’s performance affected by the quality of the input data used to build SVM models? We present below a quantitative approach to determine these sensitivities.

SVM is a state-of-art method that aims to learn relatively complex input-output relationships existing in a dataset. The key objective of applying SVM is classification, though the other methods including regression, cluster analysis, etc. have also benefited by clever use of kernel functions. The special features of SVM include a sound base of statistical learning to extract information that is more insightful than what is possible by other classification methods. However, SVM cannot uncover more information than what is available in the data itself based on which training or prediction is done. Thus, SVM helps reveal relationships between input and output, but it cannot add anything additional to that externally. Importantly, SVM working with noisy data is no substitute; for instance, a perfect or highly reliable

---

**Table 8** Sensitivity of the Asymmetric Soft Margin SVM’s Performance on Costs $C^+$ and $C^-$

<table>
<thead>
<tr>
<th>$C^+$</th>
<th>$C^-$</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>$P[\text{Type I error}] = \alpha$</th>
<th>$P[\text{Type II error}] = \beta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>75</td>
<td>25</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>25</td>
<td>75</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>99</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>undefined</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.2</td>
<td>0.2</td>
</tr>
</tbody>
</table>
clinical test. Input data for SVM generally requires preprocessing—this is where the class to which each input example belongs is assigned. This typically requires high calibre expertise. However, even well-regarded medical professionals and technologies may produce only ~50 to 90% correct diagnostics (Bruns et al. 2000; Pewser 2007; SIDM 2013; Sweilam et al. 2013). Congalton (1991) also notes this issue in the classification of remotely sensed data. This begs the question—how sensitive is the performance of SVM to errors in the input data, in particular, errors in class labelling of input data during its pre-processing? We attempt here to set up an analytical procedure to address this, and some sensitivities, under stated assumptions. For it, we consider a two-class situation and examine its resulting ROC Curves.

**Modelling Imperfect Classification at Training Data acquisition Stage**

We represent here the total process of training the SVM as a two-stage process, the first being input data acquisition and its preprocessing including class tagging. The output of this stage is the class-labelled training dataset \(\{(x_i, y_i)\}\). The second stage is the development of the SVM model itself, which comprises training. Subsequently, the trained SVM is applied to predict the class of a yet unseen feature vector \(x\).

Stage 1 is the preprocessing stage that through a human expert’s judgment (common in medicine) or other means assigns a class label \(y\) to each input example \(x_i\). We assume here that this assignment itself is imperfect or erroneous: For a truly positive example \(x\) belonging to class \(R^+\) that comprises \(P\) positive members, the preprocessing expert assigns a positive label \(T^+\) with probability \(\mu\). The rest of \(P\) are wrongly assigned the negative label \(T^-\) with \(Pr[T^-/R^+] = (1 - \mu)\). Similarly, truly negative examples comprise class \(R^-\) in which there are \(N\) negative members. Preprocessing assigns positive and negative labels or tags to these truly negative \((R^-)\) members with probabilities \(Pr[T^-/R^-] = \nu\), \(1 \geq \nu \geq 0\) and \(P[T^+/R^-] = (1 - \nu)\) respectively. As the result of these label assignments done at the exit of Stage 1, the expected True Positive count (TP) becomes \(\mu P\); true negative count (TN) equals \(\nu N\); False Positive (FP) count equals \((1 - \nu)N\) and False Negative count equals \((1 - \mu)P\).

Stage 2 is the action zone for the SVM which receives its input from Stage 1, some examples \((x)\) being mislabelled in Stage 1 itself due to imperfect preprocessing/diagnosis. By assumption, Stage 1 does not affect \(x\), but probably only \(y\). Additionally, we are already aware that the SVM also is not a perfect classifier, its accuracy being \(\leq 1\) (Section II above). This might greatly affect the overall accuracy of an SVM application—so the class counts resulting after SVM’s acting on the pre-processed data change. True Positives now become \(TP'\) in count, for instance. So do change the other counts of classes assigned.

The SVM’s sensitivity (accuracy in classification in Stage II based on \(x\)) is denoted by the Bernoulli rate \(\theta\), hence \(Pr[\text{SVM label }+/T^+] = \theta\), \(1 \geq \theta \geq 0\) and \(Pr[\text{SVM label }-/T^+] = (1 - \theta)\). Similarly, the SVM’s performance for classifying negative labelled examples as negative is \(Pr[\text{SVM label }-/T^-] = \phi\), \(1 \geq \phi \geq 0\), and lastly \(Pr[\text{SVM label }+/T^-] = (1 - \phi)\). These changes lead to new error-affected counts of true positive (\(TP'\)), true negative (\(TN'\)), false positive (\(FP'\)) and false negative...
(FN') cases as shown in Table 9. In Table 9, we reflect these changes by utilizing the original counts of the truly positive (P) and truly negative (N) examples in the original training dataset.

Table 9 Derived Error-affected Confusion Matrix Class Counts

<table>
<thead>
<tr>
<th></th>
<th>0μP + θ(1−μ)P</th>
<th>ϕvN + ϕ(1−v)N</th>
<th>(1−θ)μP + (1−θ)(1−μ)P</th>
<th>(1−ϕ)(1−v)N + (1−ϕ)vN</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TN'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FP'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 10 SVM Performance Measures affected by Imperfect Class Labelling of Input Data

<table>
<thead>
<tr>
<th>Overall Sensitivity = TP'/(TP' + FN')</th>
<th>θμP + θ(1−μ)P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Specificity = TN'/(TN' + FP')</td>
<td>ϕvN + ϕ(1−v)N</td>
</tr>
<tr>
<td>α = Probability[mis classifying true positive as negative] = FP'/(TN' + FP')</td>
<td>1 - Specificity</td>
</tr>
<tr>
<td>β = Probability[mis classifying true negative as positive] = FN'/(TP' + FN')</td>
<td>1 - Sensitivity</td>
</tr>
</tbody>
</table>

Using performance measure definitions given in Section II for sensitivity, specificity, Pr[Type I classification error] = α and Pr[Type II classification error] = β, and on substituting TP', TN', FN' and FP' for TP, TN, FN and FP in the respective expressions in Section II we obtain Table 10.

To check the correctness of the expressions in Table 10 we can take the perfect (error-free) preprocessing condition of μ = 1 (no misclassification of positive examples in Stage I) and υ = 1 (i.e., no misclassification of negative examples in Stage I). Substituting these yields overall sensitivity = θ, overall specificity = ϕ, Pr(Type I error) = α = 1 − ϕ and Pr(Type II error) = β = 1 − θ, all anomalies occurring only at the SVM stage. To the best of our knowledge, such sensitivities in classification in the context of the SVM have not been explicitly stated in the literature yet (Bi and Zhang 2010).

It is not difficult from Table 10 to determine the sensitivities of these performance measures to input data quality (here any misclassification at the preprocessing stage) before the data is fed to SVM. For illustration, the sensitivity of α to θ and μ has been worked out. Figure 7 shows it. It is noted that as the pre-processor’s and the SVM’s sensitivities (μ and θ) rise from 0.0 to higher values, this causes improvement in a SVM’s performance. This is expected. Measured by α, the overall probability of misclassification of a positive example, it falls.

Figure 7 Dependency of Overall Type I Error α to Pre-proc (μ=υ) and SVM (θ=ϕ) Accuracies
We now turn to determining the ROC Curve for the input mislabelling-inflicted two-class SVM. Our defining concepts of ROC are drawn from Fawcett (2003, 2006). Receiver operating Characteristics (ROC) curves were originally used in signal detection theory to show the tradeoff between hit rates and false alarm rates of two-class classifiers. In recent years, the medical decision making community has extensively engaged in using ROC for diagnostic testing. Supervised learning, a class of methods to which SVM belongs, uses training data to develop input/output relationships, to subsequently develop its class predictive capability of yet unseen examples. Such capabilities of a given classifier in turn are evaluated using performance measures listed in Section II.

Developing the ROC requires two additional measures—False Positive Rate (FPR = 1 – Specificity) and True Positive Rate (TPR = Sensitivity). Given a two-class classifier, its ROC curve itself is plotted as FPR (on X axis) vs. TPR (on Y axis), as shown in Figure 8. As Fawcett (2006, Figure 2) shows, each classifier occupies a point in the ROC space, with good classifiers appearing in the north west corner, whereas virtually useless classifiers that perform worse than randomly assigning examples to positive/negative classes locating in the south east corner, in that space (see Figure 8). Our mission presently is to determine the effect and extent of input data mislabelling (which affects training), on the ROC of the SVM.

To determine the effect of mislabelling examples in Stage 1 we make an approximation—\((1 - \theta)(1 - \mu) \approx 0\) and \((1 - \phi)(1 - \nu) \approx 0\) as the most interesting SVMs would lay in the N-W corner of Figure 8. With this FPR and TPR become

\[
FPR = \frac{(1-\phi)\nu}{\phi + (1-\phi)\nu} \quad (9)
\]

and

\[
TPR = \frac{\theta}{\theta + (1-\theta)\mu} \quad (10)
\]

On substituting different numerical values for the SVM parameters \(\theta, \phi, \mu\) and \(\nu\), \((FPR, TPR)\) pairs as \((X, Y)\) points for each SVM with misclassifications occurring in Stage 1 can be generated. Table 11 and Figure 9 show typical SVMs with the displayed parameter values.
It is possible to sense the sensitivity of the SVM to Stage 1 misclassification error rates \((1 - \mu)\) (misclassifying a positive example as negative, i.e., as FN) and \((1 - \nu)\) (misclassifying a negative example as positive, i.e. as FP). Two cases are shown in Figures 10 and 11. Both figures show several SVMs that are as bad as or even worse than random classification. However, such displays do not precisely disclose the relative impact of the inaccuracies active at the pre-processing (Stage I) and SVM (Stage II) steps. To do this, we utilize the relationships (9) and (10) and invoke orthogonal array experimentation (Montgomery 2005), as follows. First we select the two working levels (settings) for each SVM parameter (“Experimental Factor”) \(\theta, \phi, \mu\) and \(\nu\), the “high” setting being 1.0 while the “low” will be 0.5. The combinations used in the experiments were guided by the \(L_8\) array (Table 12), column assignments carefully done so as to avoid confounding of effects (Montgomery 2005).

Each “Experiment” was the simulation of the operation of an SVM using (9) and (10) parameterized...
by the rows listed against “Computational Experiment #” in Table 12. Factor effect calculations are straightforward. The effects themselves are shown graphically in Figure 12. Interpretations of these results are given below.

Table 12  L₈ Orthogonal Array Computations to find Factor Effects on FPR and TPR

<table>
<thead>
<tr>
<th>Computational Experiment #</th>
<th>SVM's Sensitivity θ</th>
<th>SVM's Specificity ϕ</th>
<th>Pre-processing Sensitivity μ</th>
<th>Pre-processing Specificity ν</th>
<th>Classifier's performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.00 1.00</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>0.00 1.00</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>0.5</td>
<td>0.33 1.00</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>0.50 1.00</td>
</tr>
<tr>
<td>5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>0.00 0.50</td>
</tr>
<tr>
<td>6</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>0.5</td>
<td>0.67 0.67</td>
</tr>
<tr>
<td>7</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>0.50 0.50</td>
</tr>
<tr>
<td>8</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.33 0.67</td>
</tr>
</tbody>
</table>

Figure 12  SVM Factor (Parameter) Effects estimated using an L₈ Orthogonal experimental array with working ranges high settings θ = ϕ = μ = ν = 1.0 and low settings θ = ϕ = μ = ν = 0.5

Figure 13 Simulated performance of eight different Two-class SVMs using parameters θ, ϕ, μ and ν as shown in Table 12
The simulations conducted with results shown in Table 12, Figure 12 and Figure 13 are quite informative. The first thing one notices is that the impact of having a high accuracy SVM characterized by sensitivity $\theta$ and specificity $\phi$ appears to be stronger than having high accuracy pre-processing (near perfect $\mu$ and $\nu \sim 1.0$) of input data aimed to keep mislabelled examples in the training set itself low. For the same working range variation of $\theta$ and $\phi$ (0.5 to 1.0), the effect of varying $\mu$ and $\nu$ also from 0.5 to 1.0 seems relatively low. Hence it is observed from Figure 12 that $\theta$ and $\phi$ respectively affect TPR and FPR strongly. On the other hand, low $\mu$ raises TPR mildly, while high $\nu$ raises FPR—also mildly.

Thus, the relative impact of the SVM parameters ($\theta$ and $\phi$) in respect to pre-processing parameters ($\mu$ and $\nu$) gets inferred as more significant. However, as the location of good SVMs on Figures 8 and 13 indicate, it would be our learned advise to keep accuracies in both stages—pre-processing of input data, as well as classification by SVM high, i.e., keep all four parameters $\theta$, $\phi$, $\mu$ and $\nu$ as close to 1.0 as possible. A harm that a high rate of misclassification at Stage I would do is also to shrink the width of the soft margin, raising generalization errors of the classifier (see (8)).

**Conclusions**

To study birds one doesn’t have to start with the ostrich. A sparrow has all the parts that a typical bird has and it will often do just fine. This study used small classification problems with the same intention—to quickly capture and study performance features of the SVM, a powerful tool being increasingly employed in today’s management of medical diagnostics (El-Naqa et al. 2002).

Our study used time-honoured SVM performance metrics including ROC, the soft margin model permitting less than perfect classification, as well as imperfections in training (input) data. The goal was to shed light on the relationships among SVM design parameters, classification performance metrics, as well as defects in the form of misclassifications present in the training data. It used the orthogonal array experimental framework to simulate different conditions that were hypothesized to affect both the sensitivity and also specificity of two-class classifiers in general and the SVM in particular—hitherto not done in the literature.

We first established that there is a strong relationship between the SVM’s performance and its ability to classify examples correctly, even in the presence of misclassified training examples. We next found that misclassified inputs also affect the quality of training, and hence performance, though not as strongly. Still, our results provide strong support for one’s striving to develop the best trained SVM that is intended to be utilized, for instance, for medical diagnostics. We illustrated the criticality of this requirement by invoking a well-known enigma in the common perception of test quality. Such effort would require high level of diligence in ensuring that the input data has minimal misclassifications in it. Misclassifications present in training data cause narrowing of the distance between boundary planes, thus increase generalization error. Furthermore, real life consequences (costs) of making wrong decisions need to be incorporated in the design optimization objective. Such costs emanate, for instance, from calling benign pathology malignant, or vice versa.

Thus, however enabling it is in many complex and multi-dimensional situations, by itself, SVM is no magic bullet—its performance depends almost totally on the quality of the SVM model developed as well as the training examples showed to it and the penalties set when the soft margin model is adopted. In view of this one may doubt if computer science alone would be able to substitute for more accurate and sophisticated clinical tests, for instance, evolving genetic investigations for cancer diagnosis (El-Naqa 2002; Jolie 2013).
References

news(update/2013-06-25/breast-cancer-awareness-raised-by-angelina-jolie/

- Ng, Andrew (2013). Machine Learning, Lecture 229, Coursera, Stanford.

Tapan Bagchi holds a B Tech (IIT Kanpur), MASc and Ph D (University of Toronto) and a D Sc (IIT Kharagpur). A Registered Professional Engineer (Ontario) and Fellow (Institution of Engineers, India), he served EXXON-Mobil Corporation on four continents for 16 years before returning to academics. Author of 60 journal papers, over 100 presentations and six texts on Operations Management, IE/OR, Quality Engineering, Computer Science and Engineering Economics, currently he is the Director, SVKM’s NMIMS University at Shirpur. Earlier, he served IIT Kanpur, NITIE Mumbai, IIT Bombay, SPIIMR Mumbai, NDS Infoserv and IIT Kharagpur variously as Professor, Dean and Director. He can be contacted at tapan.bagchi@nmims.edu