Estimating True And False Positive Rates In Higher Dimensional Problems and its Data Mining Applications

Andrew Foss Osmar R. Zaïane

University of Alberta
Department of Computer Science, Alberta, Canada
{afoss,zaiane}@cs.ualberta.ca

Abstract

If we can estimate the accuracy of our observations then we can estimate the true and false positive rates over a series of samples in high dimensional data mining problems. To date such issues have been largely neglected and previously no algorithm has been provided to facilitate the computations involved.In high dimensional data mining tasks, increasing sparsity leads to decreasing true positive rates. Estimating this effect allows the estimation of the true size of membership of a class or cluster allowing us to identify the top candidates for these false negatives, while tracking the likelihood of false positives. These estimates of true and false positive rates can also help researchers avoid unnecessary costs by collecting only the number of samples that are really needed. We propose an algorithm for these computations designated the Statistical Error Rate Algorithm (SERA) and give an example of its use.

1. Introduction

In every classification projects, one has to take account of false positives (FP) and false negatives (FN). This issue becomes especially acute in very-high dimensional experiments such as those employing microarrays. While, we will discuss this problem with frequent references to microarrays, the implications are entirely general to any problem with N samples and D dimensions with particular relevance where D is large.

In this paper, 'targets' are features or points that we seek. Features will refer to dimensions or attributes which show some pattern, for example, indicating up or down regulated genes. In the microarray scenario, there are usually a small number of samples, which are treated as points and a large number of genes which are treated as dimensions. In other data mining contexts, the targets could be data points each of which has values over a set of attributes. This is illustrated in Figure 1. Thus the semantics depend on the circumstances but the algorithm presented is entirely general.

While a dataset contains a set of T targets, we observe the set O. The true positives are $T \cap O$, the false negatives are $T \backslash (T \cap O)$ and the false positives are $O \backslash (T \cap O)$. False positives are also called type I errors while false negatives are type II errors.

In the data mining community we often assume that our data is free of error. This allows us to trust our results. If we obtain a result such as a cluster containing a set of points P, we typically report P as the target cluster members. However, in any real world scenario, this approach is unsafe. Virtually all data is noisy and this noise leads to both false positives and false negatives. As dimensionality increases, this problem becomes increasingly acute as sparsity increases exponentially with dimensionality and, thus, the likelihood of observing a cluster similarly declines under standard approaches.

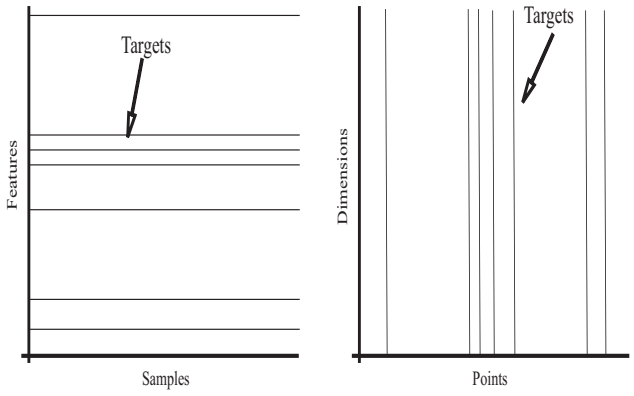

Figure 1. Examples of targets in different data mining scenarios. On the left, the targets are features showing significance across samples. On the right the targets are points clustered or classified based on similarity on a set of dimensions.

For example, let us imagine a survey made of 100 students. Each is asked 50 questions. Suppose 20 of the students do genuinely belong to a group that should give similar or identical answers on a subset of the questions. What chance have we of observing a subspace cluster of all 20? Answers may be misrecorded or misinterpreted. If we find a group of 18, we should ask what is the likely true size of the group? Which of the students not clustered but closely similar should be included? Obviously such questions are important but, surprisingly, have received little attention.

Once one has a series of samples S with dimensionality D such as a number of microarray tests of patients with the same disease and we are looking for a target set of activated genes out of thousands measured, the statistical problem becomes non-trivial. This paper provides an algorithm for computing the answer to this problem and presents how this answer can be applied to improve research results and reduce research costs. Specifically, the problems of both false positives and false negatives are addressed.

1.1 Related Work

Storey and Tibshirani [19] discuss extensively the problem faced by researchers using microarrays. The essence is that there are typically thousands of genes being tested and a confidence level has to be established for significance for each gene. Traditional p-value cutoffs of 0.01 or 0.05, widely employed in science by convention, lead to a potential

$$FP = pD \tag{1}$$

false positives. With dimensionality D large, FP becomes unacceptably large.

The initial means of handling this was to tighten the criteria. The Šidák correction for multiple comparisons [22] tightens the criteria for the probability of a false positive at the feature level (α_c) to yield a desired probability at the experimental level (α_e). Assuming all the features are independent, the correction is

$$\alpha_c = 1 - \sqrt[D]{1 - \alpha_e} \tag{2}$$

The Bonferroni correction [1, 8] noted that, for α_c small, using approximations a simpler form of Equation 2 can be derived, that is

$$\alpha_c = \frac{\alpha_e}{D} \tag{3}$$

The usual application of these corrections is to choose α_c such that $\alpha_e \leq 0.05$ or $FP < 1$ (with reference to Equation 1). If D is large, this approach is extremely conservative. That is, α_c is so small to likely cause many false negatives [19]. While false positives are expensive to handle, false negatives can mean the whole work fails. To reduce this, some authors have applied this correction less stringently (e.g. [14, 11, 9]). Lee et al. [18] reported their results at several threshold levels but this makes it difficult to interpret.

Holms proposed a step-down approach that is less conservative [15]. The features are first ranked by their probability of being targets. Then the threshold probability is also varied according to the ranking such that for feature rank k, the threshold probability p is

$$p = \frac{\alpha_e}{D - k + 1} \tag{4}$$

This also assumes the features are independent. A similar approach that has been shown to allow for some feature dependencies is the False Discovery Rate (FDR) correction [6, 7].

To improve on this, the q-statistic was proposed [20, 19]. The q value is similar to the p value except that it measures significance in terms of the false discovery rate (FDR) rather than the false positive rate (FPR). The FPR is the rate at which non-target features, sometimes referred to as truly null, are designated significant, i.e. observed as targets. The FDR differs in that it is the rate that features designated significant are not targets. With certain provisos the FDR can be written as the probability $Pr(\text{feature } i \text{ is non-target} \mid \text{feature } i \text{ is significant})$ and the FPR as $Pr(\text{feature } i \text{ is significant} \mid \text{feature } i \text{ is non-target})$. This is a more sophisticated way of tightening the criteria for acceptance of a feature as significant where the number of significant features is much less than the number of non-target ones.

In addition, Westfall and Young proposed a bootstrapping method [24] that does not involve any assumption of independence. The disadvantage to this method is that it is typically expensive to compute and strictly empirical [10].

Elsewhere [12] we have proposed using standard p-value levels but dividing the samples in two or more groups. This reduces the likelihood of false positives by the power of the number of groups. Working with labelled genome data, this method yielded classification results better than or equal to the best achieved so far.

These approaches per se do not explicitly model the estimated impact of the noise or errors in the data, if known, or, if not known, apply the conventional assumed error rate (5%) to estimate the impact on the statistical model, given the experimental parameters such as the number of samples and the number of features (points and dimensions). The approach proposed here is to estimate, given known or estimated error rates,

1 the expected number of false positives, and

2 the true number of targets given the observed number.

In the following, we present an algorithm to make this estimate and then propose and illustrate how this output can

be leveraged to overcome the difficulties of false positives and negatives.

The contributions of this paper are

1 A new approach to statistical analysis of true and false positive rates in multi-sample, multi-dimensional problems.

2 An algorithm for applying this in data mining applications.

3 A new method for identifying target genes in microarray experiments with diagnostic labelling, which could be applied in any similar problems.

In the next section the statistical basis of this approach is explained and the algorithm presented. Then sample results using the algorithm are given and an example of its application in a real world situation. Finally, we conclude and present a proposal for reducing research costs especially for microarray and similar applications.

2. Methodology

As has been often mentioned in the literature, if we have a 5% chance of a false positive (FP) and a large number of features then we will have many false positives. 10,000 genes will be expected to give 500 FPs. Lee [17], in the context of microarray studies, argues that this is a reason for replicating the tests on each chip. This doubles the cost but greatly reduces (without eliminating) the FP rate. Another way of eliminating FPs is developed here based on a single round of microarray testing.

The question we address is how many targets (true positives) or non-targets (false positives) will be seen consistently if we test many samples? That is, if we have s samples, how many will appear with a measured value considered non-null on every sample? From this we can answer the questions, 'how many samples do we really need?' and 'how can we eliminate false positives?'.

The underlying statistical question is, essentially, 'given n coins with a weighted probability of coming up heads of p and d tosses of each, what is the probability of seeing at least the same k coins coming up heads in every round of tosses?' If there is no error or noise then $p = 1$.

In a typical high dimensional problem such as a micro array (MA) study where there are both target and non-target features, we can ask two questions both of the form 'given n genes with a probability of testing positive p and d samples, what is the chance that k unique genes are positive in every sample (that is, consistently up or down regulated)?' There are two questions because, typically, we have a small number of target genes which are being sought and which we can expect to observe at some high likelihood p_t, even though we do not initially know which they are, and then a very large number of other genes from which we may get false positives due to error or noise with a probability of $p_f = 1 - p_t$, making the reasonable assumption that both will be subject to the same error rate or noise level.

The error rate can sometimes be known. For example, microarray manufacturers provide estimates and, in general, controlled experiments can be made on known quantities. However, this is often difficult or expensive so, in most situations, the error rate or degree of stochasticity is unknown. While this might appear to be a major hurdle, given the importance of the issue to science, the research community has long since adopted a convention of assuming an error rate of 5%, which means that if a measure is made with better than 95% certainty, it is accepted. In terms of probability one seeks $p < 0.05$ or, sometimes, $p < 0.01$. Other p-values may be used when there is good reason to do so. In the algorithm provided in this paper, any confidence level can be input.

Our approach assumes that the stochastic processes resulting in errors due to the samples or the measuring technology are independent between features. We are not assuming anything about the correlations between features, only that their variances are independent. The algorithm given assumes that the error rates are the same for all features or points, but it could readily be modified if a distribution of these was known.

Since this problem has been framed in terms of coin tosses, we can derive a statistical formulation to evaluate the probability. The problem is related to the binomial distribution. The binomial distribution deals with a sequence of n tosses of a weighted coin. The probability of getting exactly k successes in n trials is given by the Probability Mass Function:

$$Pr(k) = \binom{n}{k} p^k (1-p)^{(n-k)} \tag{5}$$

In the problem addressed in this paper, we are tossing n labelled weighted coins a sequence of d times and we want to know how often the same set of at least k coins come down 'heads' every time.

This is best expressed as an algorithm. Suppose we want to compute how many ways we have of getting at least k heads on every throw of n weighted coins. On the first throw of all the coins, as per Equation 5, we have $\binom{n}{k}$ ways of getting k heads with a probability $p^k(1-p)^{(n-k)}$. Since we are interested in 'at least' k heads, we have to compute the probability of $k+1$ heads, $k+2$ heads, etc. up to all n coins being heads. This is expressed as the Cumulative Distribution Function:

$$Pr(X \geq k) = \sum_{j=k}^{n} \binom{n}{j} p^j (1-p)^{(n-j)} \tag{6}$$

Algorithm 1 SERA – Statistical Error Rate Algorithm

Input: Samples s, number of features n, number of significant features observed k, probability p of seeing a significant feature in a sample.
Output: Probability of seeing k significant features
Note: $choose(a, b)$ computes $\binom{a}{b}$

/*Main Function*/
$ComputeProbability(n, s, k, p)$ {
$\{Mx, Px\} \leftarrow BuildMatrix(p)$ /*Algorithm 2*/
return $Compute(n, s, k)$
}

$Compute(n, s, k)$ {
$val \leftarrow 0$
for $w \leftarrow 0$ to $n - k$ incrementing by 1 **do**
$val \leftarrow val + choose(n, n - w) \times Px(n, w) \times CompLine(n, w, s - 1)$
end for
return val
}

$CompLine(n, w, s)$ {
/*if we've seen this parameter combination before*/
/*return the stored value, otherwise compute and store*/
if $AlreadyComputed(n, w, s)$ **then**
return $GetStored(n, w, s)$
end if
$val \leftarrow 0$
for $v \leftarrow 0$ to $n - k$ incrementing by 1 **do**
$val \leftarrow val + Mx(v, w) \times$
$CompLine(n, Max(w, v), s - 1)$
end for
$SetStored(val, n, w, s)$
$AlreadyComputed(n, w, s) \leftarrow true$
return val
}

Algorithm 2 Build Look up Value Matrix

Input: Number of features n, number of significant features observed k, probability p of seeing a significant feature in a sample.
Output: Mx, Px which store the precalculated values
Note: $choose(a, b)$ computes $\binom{a}{b}$

$BuildMatrix(p, n, k)$ {
/*Reset temporary value vectors*/
$\forall n, k\ Px \leftarrow p^k(1-p)^{(n-k)}$
for $w \leftarrow n$ to k decrementing by 1 **do**
$fac \leftarrow choose(n, w)$
$ptemp \leftarrow probabilityPx(n, w)$
for $v \leftarrow n$ to k decrementing by 1 **do**
if $(w + v \geq n + k)$ **then**
$Mx(w, v) \leftarrow fac$
else if $(w \leq v)$ **then**
$fac \leftarrow choose(v, w)$
$Condition1(w, v, fac)$
else
$Condition2(w, v)$
end if
end for
end for
return Mx, Px
}

$Condition1(w, v, val)$ {
$tot \leftarrow val$
$j \leftarrow w - 1$
$i \leftarrow 1$
while $(j >= k)$ **do**
$tot \leftarrow tot + choose(v, j) \times choose(n - v, i)$
$j \leftarrow j - 1$
$i \leftarrow i + 1$
end while
$Mx(w, v) \leftarrow (tot \times ptemp))$
}

$Condition2(w, v)$ {
$tot = choose(n - v, w - v)$
$j \leftarrow v - 1$
$i \leftarrow w - v + 1$
while $((j \geq k))$ **do**
$tot \leftarrow tot + choose(v, j) \times choose(n - v, i)$
$j \leftarrow j - 1$
$i \leftarrow i + 1$
end while
$Mx(w, v) \leftarrow (tot \times ptemp))$
}

On the second round, we have to check, for each of the first possible sets of throws, the chance of a match with all possible k or more heads combinations in the second throw, i.e. where k or more coins have heads in both rounds of throws. This has to be continued for all of the d rounds. It is immediately clear that there is a tree of computations with a fixed branching factor. This suggests an intractable computation for large values of the parameters as the number of computations increases exponentially.

Fortunately, we can greatly accelerate the computation. Firstly, it can be seen that the comparison between each pair of rounds involves the same computations, so a matrix of values can be prebuilt and used for lookup (Algorithm 2). At the same time, as the calculation proceeds, we can build a vector of probability values for different values of n and k.

Further, we can formulate the problem using a recursive algorithm and the return value from each call depends only on the parameters passed. Thus we can keep a log of return values and calling parameters and look up the result in this vector whenever we find a parameter match (Algorithm 1). Since the redundancy is high, the problem of exponential blowup is avoided. The principal cost in the calculation turns out to be the computations of $\binom{n}{k}$ for the lookup table where $n > 1000$ primarily because high precision math libraries are required that take a great deal more computing power on a 32bit machine. If a table of these values was precomputed on a very fast computer or computing cloud, then the remaining computations for any given problem would return very quickly.

We have not found in the literature any previous algorithm that provides this computation so we are unable to offer a comparison.

3. Results

As discussed above, there are two separate applications of this algorithm. One is to estimate the false positive rate. That is, given n points or features that are non-target and therefore have a low probability of being false positives, how many will be seen consistently? For example, in a microarray experiment with n features, assuming they are (almost) all non-target, how many will show as significantly regulated on every sample. In a clustering application, out of n points with d dimensions being considered, how many false positives can we expect to see? Here we assume that, in order for a false positive to appear in a d dimensional cluster, it has to be falsely given significance on all d dimensions. This would apply explicitly to applications like projection clustering [2, 16] and, arguably, implicitly to other algorithms.

The second application is regarding the target features that we expect to observe with high probability. If we observe m, how many really exist? Answering that allows us to estimate the true positive and false negative rates.

Table 1. The number of target features designated significant in every sample by an application and the number that can be inferred to exist at a 95% confidence level for different numbers of samples (e.g. 1 sample, 5 samples, etc.).

Designated Significant	Inferred Targets for No. of Samples				
	1	5	10	20	30
5	6	8	9	10	10
10	12	14	16	16	17
15	17	20	22	23	23
20	23	27	28	29	30
25	28	32	34	35	36
30	34	38	40	41	42
50	55	62	64	65	66
100	109	120	122	124	125

3.1. False Negatives and the True Positive Rate

Some sample computations are shown in Table 1. The first column is the number of points or features observed and the other columns indicate the estimated true number of targets for various numbers of dimensions or samples, respectively. An error rate of 5% is assumed. For example, for 10 samples, if we see 10 features testing positive on all the samples, then, at the 95% confidence level, we can expect there are 16 target features, so we observe about 62%. If we observe a total of 30 features, then the real number is about 40 so 75% of the likely target features have been observed.

Once the number of false negatives has been estimated, from which the false positive rate can be estimated, features which were not significant (or unclustered points, etc.) but which had a high similarity to those selected as targets can be added. For example, suppose 75 points are detected in a 20 dimensional subspace cluster but we estimate there should have been 100. All of the 75 points are clustered because they have similar or identical values for the 20 dimensions. Other points may have such similarity but on fewer dimensions and could be ranked on the number of dimensions on which they were similar. The top 25 could then be incorporated into the cluster. Since measurement error/noise has likely caused some points to fall below the chosen similarity threshold, these are the points that most likely represent the missed cluster members. It is important to note that whether one is looking for sets of features or

sets of points, the same statistical approach can be applied.

Naturally, the number of points clustered in the above example is a function of the heuristics, such as thresholds, employed. The assumption is that a threshold which seems 'reasonable' to a researcher will likely correspond to the conventional 5% error rate. This assumption is the foundation of the 5% convention throughout the research community.

Researchers can use our algorithm on their own set of parameters so we just give examples of usage here to illustrate the utility.

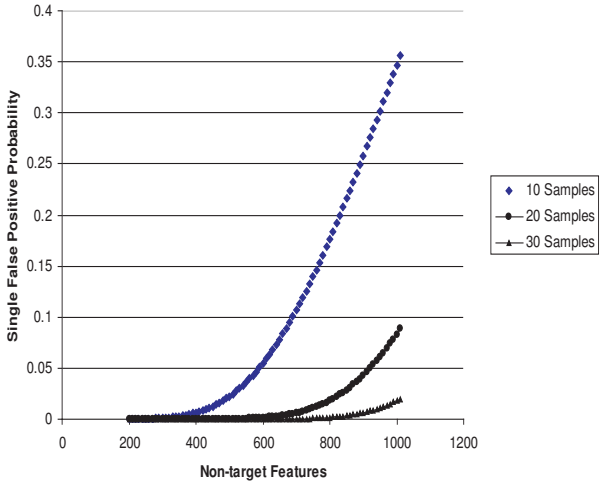

Figure 2. False Positive likelihood for a single feature to test positive on all samples with increasing number of non-target features (e.g. genes) tested (0.4 = 40%). Probability of a FP for a single feature on a single sample = 5% (0.05).

3.2. False Positives

If we have a large number m of non-target features in our test arrays, then we can expect some FPs. The question is how many and at what size of m can any be expected?

For example, if there are 20 samples and we are looking to see if a single non-target feature tests positive in every sample then up to $m = 200$, the likelihood is vanishingly small (Figure 1). At $m = 1000$, the probability is about 8% and rising fairly quickly. Thus if we have a microarray with more than 1000 genes on it some FPs are quite possible. For 10 samples the likelihood is over 35% while for 30, at $m = 1000$, the likelihood is under 1%. This is illustrated in Figure 2.

Obviously therefore, one strategy for eliminating false positives is to have a large number of samples. However, for microarrays, each sample can cost thousands of dollars to collect and process. Another alternative for the genome researcher is proposed in this paper taking advantage of the fact that we can estimate the number of FPs to be expected. This is outlined in the Proposal below (Section 4).

For the data mining researcher, the utility is in helping correct for false negatives while tracking false positives since both impact effectiveness. The algorithm can be incorporated into clustering and other algorithms as a post-processor. For example, after producing a clustering result, the algorithm can be used to estimate the number of false negatives for each cluster or class. Suppose the estimated number is m. Then the m points nearest to the cluster could be incorporated. Another approach is to use it to estimate the overall error rate. An example of how these can be done is given in Section 3.3.2.

The following section gives a real world example in the high noise, very high dimensional application of microarray analysis. This example is of particular interest because it appears that no other approach to this data has yielded a small set of target genes. Where genes have been positively associated with cancers, the number of genes involved is very small. The preprocessing is particular to microarray data but the subsequent computations have very general application.

3.3. Colon Tumour Dataset

3.3.1 Direct Analysis

This dataset [4] has 62 Affymetrix oligonucleotide arrays from biopsies of colon tumours, 40 are labelled positive and 22 are labelled negative. Our interest is in determining which genes are associated with the malignant condition. Values for 2000 genes are provided for each sample.

For this analysis, the 40 positive samples are assessed for being consistently different than the negative cases. That is, the negative samples provide the standard levels against which the positive samples are to be tested. For each gene a count is made of how often it is up or down regulated over the samples. The data was normalised in a standard way by taking the log_2 of the values divided by their mean value over all attributes for each sample [10]. Minus and plus superscripts are used to identify the groups. The mean $\bar{x}_i^-$ and standard deviation σ_i^- of the negative samples for feature i were computed for each gene and the values for the positive genes x_i^+ were converted such that

$$x_i^{+'} = \frac{(x_i^+ - \bar{x}_i^-)}{\sigma_i^-} \tag{7}$$

This gives normalized deviations of the positive group from the negative group means. In order to accumulate the

Table 2. The top genes from the Colon Tumour database.

Rank	No.	Score	Label
1	376	40	Hsa.3056 X59871 gene 1 Human TCF-1 mRNA for T cell factor 1 (splice form C)
2	764	38	Hsa.3067 X05276 gene 1 Human mRNA for fibroblast tropomyosin TM30 (pl)
3	492	37	Hsa.2311 T97199 3' UTR 1 120285 INTEGRIN BETA-4 SUBUNIT PRECURSOR (HUMAN)
4	580	37	Hsa.1207 T51571 3' UTR 1 72250 P24480 CALGIZZARIN
5	1891	37	Hsa.36161 H29546 3' UTR 2a 52669 NEUROTENSIN RECEPTOR (Homo sapiens)

set of genes in the positive samples that are consistently up or down regulated, genes were checked for being greater or less than the mean of the other group by some small amount, here at least 0.05σ. For the experimental group, only one gene (40 samples) was consistently different in the same direction than the control group. Applying the algorithm shows that 4 other genes are likely to be target genes, rounding to the nearest integer. Thus we can predict that the top 5 genes are strong candidates for being associated with this disease condition. These genes are given in Table 2. The highest scoring gene appears in a cluster of genes that may be implicated in Leukemia [13]. The likelihood of a false positive is small, as can be inferred from Figure 2.

Alon et al. applied clustering to this dataset [3] based on a deterministic-annealing algorithm and showed that this clustering could separate the two classes but was not effective in identifying a few genes that likely play a fundamental role in this disease condition. After clustering, Alon et al. removed the 500 most significant genes and found that their algorithm still gave a successful clustering of the two classes. The various approaches to multiple test corrections can regulate the number of targets identified by adjusting thresholds but cannot identify the correct number.

Suppose no gene had scored 40 over 40 samples. This would indicate that there may be less than 5 target genes. To further investigate, we could take the highest scoring gene. Suppose that gene had a score of 20. Then if we apply the algorithm on the basis of one positive on 20 samples then we get a result, after rounding, of 3 anticipated false negatives.

Clearly the method adopted for any particular application contributes to the error rate. For example, different clustering methods will yield different sized clusters. In the example above, the number of genes detected is influenced by the choices made in selecting them. Thus we do not know exactly what our overall error rate is without labelled data to test against, in this case genome level labelling. However, we can certainly improve on the raw result by making the conventional assumption of a 5% error rate as we have above, and using that to derive the number of false negatives and positives.

3.3.2 Application to Classification and Clustering

A number of different types of algorithms have been used to classify or cluster this dataset [12, 3, 5, 21, 23]. Accuracy rates compared against the diagnostic labels range up to 92%. That means, for the group of 40 positive samples, these algorithms classified up to 36 correctly. Our algorithm FASTGENE [12] achieved the highest score but the differences were small. Naturally we considered why we had not achieved 100% accuracy. A number of the genes appeared to be misclassifying the remaining samples. However, it is interesting to ask how many out of 40 should we expect to see, given that there must be a non-zero error rate inherent in the data as well as consequential to the heuristics of the classifier?

In order to apply SERA, we need an estimate of the number of dimensions relevant to the detection of the samples. If we knew this then we can find the error rate. Alternatively, if we assume an error rate then we can get an estimate of the number of relevant dimensions. First, let us take the result of the last section, that is that there are 5 relevant dimensions. Then, SERA shows that finding 36 out of 40 indicates only about a 1% error rate so is actually an excellent experimental result. If the error rate is, in fact, higher, then the number of relevant dimensions is less.

Of course, SERA is computing based on 36 out of 40 samples having proved positive on all 5 dimensions every time. The actual algorithms will not necessarily require this or even investigate in this way. However, implicit in any algorithm is both the concept of a selected feature (or point) being consistent in its significance, that is significant frequently if not every time, and then a post processing stage, as we discussed above, where the best candidates are selected according to some threshold. This secondary stage accommodates somewhat for the errors or noise. The judicious choice of such thresholds actually achieves the low error rate indicated by SERA as it is a statistical assessment of the likelihood of such a result.

4. Optimising Microarray Results and Minimising Costs: A Proposal

We have shown that the number of positives - e.g. target genes - seen in every sample is typically 50% - 75% of the underlying number present. This helps us decide how many of the genes which test positive in less than all of the samples are to be accepted. We have also shown that false positives can occur but the likelihood drops sharply with increasing number of samples. While computing the likelihood for n non-target genes for $n > 1000$ is beyond standard floating point routines on a 32-bit computer, higher precision routines or more powerful computing resources can be employed if needed.

From the above we can propose the following way of eliminating false positives. Even though this involves a second round of testing of the samples taken from patients, the second round is much cheaper due to the small number of genes tested for and the number of samples required can be much reduced so lowering the overall cost.

Since we have an estimate of the number of possible false positives fp, if fp is above some small value that indicates a significant risk of assigning importance falsely to a gene (e.g. $fp > 0.5$) and, given that we have identified a set of potential target genes G, new tests can be performed on only the top $|G'|$ ranking genes such that $|G'| \geq |G|, G \subseteq G'$.

More formally, let the genes testing repeatedly positive across all samples be G^*. Let the estimate of the number misclassified using the algorithm described here be $Est(p)$ based on the error rate p determined by the experimenters from experiments or as sufficiently conservative to account for all likely measurement errors. Then of the ranked set of genes G the top g candidates that were not in G^*, where $g \geq Est(p)$ along with G^* are retested.

That is, retest those genes identified and those that came close to being identified as implicated in the disease. The probability of a false positive appearing twice declines exponentially with the number of retesting rounds and thus, in most cases, are well below any meaningful threshold after a second round.

This approach greatly reduces the retesting costs by minimizing the likelihood that any false positives will occur across all the samples without the high cost of replication tests using microarray chips. Such replication would only be important if the number of samples was small.

Variations on this approach can be utilised in many applications.

5. Conclusions

This paper presents an algorithm which could prove useful for both data mining researchers and those planning genome and similar high-dimensional studies. The problem of false positives and negatives is a major issue for researchers. As we have shown, the presented algorithm can give both an estimate of the number of false positives, which gives the true positive rate, and the likelihood of false positives. This algorithm can be incorporated into data mining applications to adtake advantage of these results to improve effectiveness.

We have also shown that this approach can lead to an estimate of the number of target points or features in situations where other approaches can only give a general ranking. Finally, medical researchers can use the estimates to reduce costs and procedures by minimising the number of samples that have to be taken.

References

[1] H. Abdi. *N.J. Salkind (ed.): Encyclopedia of Measurement and Statistics*, chapter Bonferroni and Sidak corrections for multiple comparisons. Sage, 2007.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 94–105, 1998.

[3] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. USA*, 96(12):67456750, 1999.

[4] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. http://microarray.princeton.edu/oncology/affydata/index.html, 2007.

[5] A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini. Tissue classification with gene expression profiles. *Journal of Computational Biology*, 7(3-4):559–583, 2000.

[6] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society B*, 57:289–300, 1995.

[7] Y. Benjamini and D. Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals. of Statistics*, 29(4):1165–1188, 2001.

[8] C. Bonferroni. *Il calcolo delle assicurazioni su gruppi di teste*. pages 13-60. Rome, 1935.

[9] R. B. Brem, G. Yvert, R. Clinton, and L. Kruglyak. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 296:752–755, 2002.

[10] S. Drăghici. *Data analysis for DNA microarrays*. Chapman and Hall, 2003.

[11] W. G. Fairbrother, S. Yeh, R. F., P. A., and C. B. Burge. Predictive identification of exonic splicing enhancers in human genes. *Science*, 297:1007–1013, 2002.

[12] A. Foss and O. R. Zaïane. A hybrid classification and clustering approach for medical diagnostics and other high dimensional data. Technical Report TR08-15, University of Alberta, 2008.

[13] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. L. G. et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[14] I. Hedenfalk, D. Duggan, Y. D. Chen, M. Radmacher, M. Bittner, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, M. Raffeld, Z. Yakhini, A. Ben-Dor, E. Dougherty, J. Kononen, L. Bubendorf, W. Fehrle, S. Pittaluga, S. Gruvberger, N. Loman, O. Johannsson, H. Olsson, B. Wilfond, G. Sauter, O. P. Kallioniemi, A. Borg, and J. Trent. Gene-expression profiles in hereditary breast cancer. *N. Engl. J.Med.*, 344:539–548, 2001.

[15] Y. Hochberg and A. Tamhane. *Multiple Comparison Procedures*. John Wiley and Sons, 1987.

[16] C. hung Cheng, A. W.-C. Fu, and Y. Zhang. Entropy-based subspace clustering for mining numerical data. In *Proc. Of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 84–93, 1999.

[17] J. K. Lee. Analysis issues for gene expression array data. *Clinical Chemistry*, 47:1350–1352, 2001.

[18] T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, N. M. Hannett, C. T. Harbison, C. M. Thompson, I. Simon, J. Zeitlinger, E. Jennings, H. Murray, D. Gordon, B. Ren, J. Wyrick, J. Tagne, T. Volkert, E. Fraenkel, D. Gifford, and R. Young. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 298:799–804, 2002.

[19] J. Storey and R. Tibshirani. Statistical significance for genome-wide studies. *PNAS*, 100:9440–9445, 2003.

[20] J. D. Storey. A direct approach to false discovery rates. *J. R. Stat. Soc. B*, 64:479–498, 2002.

[21] F. T.S., C. N., D. N., B. D.W., S. M., and H. D. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16:906–914, 2000.

[22] Z. Šidák. Rectangular confidence regions for the means of multivariate normal distributions. *Journal of the American Statistical Association*, 62:626–633, 1967.

[23] J. Wang, T. Bo, I. Jonassen, O. Myklebost, and E. Hovig. Tumor classification and marker gene prediction by feature selection and fuzzy c-means clustering using microarray data. *BMC Bioinformatics*, 4(1):60, 2003.

[24] P. Westfall and S. Young. *Resampling based multiple testing: Examples and methods for p-value adjustment*. John Wiley and Sons, 1993.