ORIGINAL RESEARCH

Decision branch joint venture ex-fold T-z re-validation

Andrew Yatsko*

ITMS, the University of Ballarat, Australia

Received: November 14, 2020 **Accepted:** January 31, 2021 **Online Published:** April 4, 2021
DOI: 10.5430/air.v10n1p12 **URL:** https://doi.org/10.5430/air.v10n1p12

ABSTRACT

Comparing classifier performances may seem a banal affair but makes a side show in machine learning. Usually the paired t-test is used. It requires that two classifiers were run simultaneously or this was simulated. This is not always possible and then entails creating a superstructure only for that purpose. However, the utility of t-test in the given context is altogether doubted. The literature on alternatives is much involved. This does not measure up to the scale of the issue. In this paper the topics in connection with accuracy calculation are surveyed once more, emphasizing the result variation. The known technique of multifold cross-validation is exemplified. A simplified methodology for comparison of classifier performances is proposed. It is based on the accuracy mean and variance and calculating differences between objects defined in these terms. It is being applied to the naive Bayesian and decision tree classifiers implemented on different platforms. The lazy learning approach, applicable to decision trees in discrete domains, is closely followed with an imposition of how it can be improved. Examples are given from the field of health diagnostics.

Key Words: Classifier performance, Cross-validation, Decision trees, Lazy learners, Discrete spaces, WEKA, Diagnostics, NHANES

1. INTRODUCTION

1.1 Resampling using equally-sized folds

Resampling from existing data forms the basis of many methods of classifier accuracy estimation.[1–3] Multi-fold cross-validation (MFCV) is a resampling plan which provides a series of training and test sets to ascertain the accuracy of predictions. In N-fold cross-validation the data-set of size M is partitioned into N segments, or folds, of q or q+1 length by the number of instances, with q lower-approximating M/N. Each time the classifier is trained on an aggregate of N-1 folds and tested on the remaining one. The accuracy is the mean of average successes across all folds.[1,2,4] If the number of folds is large enough, the training sets differ by little from one another and approach the size of the data, so the whole exercise can be viewed as if the classifier was trained and tested on the same data and yet the training and test sets are independent of each other. If N=2 there is just one fold against the other. The number of instances per fold is at its maximum and no data is shared by the training sets. This may appear to be the least biased scenario. Of course, only two measurements of accuracy can be obtained. It is however altogether a different setting, adjacent to the conventional, 'all-but-one' MFCV. If M is exceedingly large, the accuracy can be estimated by the fold rotation.[2] That is, the classifier is trained on one of the folds and then tested on all others in turn. Respectively, $N\cdot(N\text{-}1)$ unique measurements of accuracy can be obtained - a substantial array even with a relatively small N. By contrast, only N besides partial accuracy readings are obtained in the conventional MFCV. The 'two-up' MFCV by rotation seems straightforward but is sketchily described in the literature.

Theoretically, there is just one sample representing the data

*Correspondence: Andrew Yatsko; Email: balunyaan@gmail.com; Address: ITMS, the University of Ballarat, VIC 3353, Australia.

concept.[1,4] Even if the sample is big, or there is more to it - all that is known about the population is contained in the collection of instances emerged so far. It is important to realize that the sample is not an arbitrary one. A sample can truly represent a data concept if only it was randomly drawn from the population. Naturally occurring samples are variably biased from one batch of data to the next but grow random the more is drawn.

Stratification is a known statistical technique based on the assumption that probabilities of different kinds of data occurring in a population are fixed, and so proportions of them present in a sample, randomly drawn from the population, do not change when it is scaled up or down.[1] Stratification is known to lessen variability of results of cross-validation.[2] So, in the case of MFCV, each fold is apportioned about the same relative amount of class data as in the original data-set. This makes accuracy estimation from test sets more reliable. However, when N approaches M implementing stratification becomes less and less feasible or even possible. Nonetheless, in the all-but-one MFCV this impacts training sets in the opposite manner as more and more they resemble the whole data, and so no stratification is required. The leave-one-out cross-validation (LOOCV) is a variant of the all-but-one MFCV where the number of folds equates the number of instances.[1,2] Withholding one instance has a negligible impact on the corresponding training set. However, it is altogether impossible to perform the rotation, as whatever little information is provided in a fold - is for one class only, and so the classifier cannot be trained to recognize other classes. Neither the rotation is feasible of folds that are statistically small.

The fold size and how the fold content is obtained matter not only in connection with stratification. Firstly, subsampling from a representative set, even though in random, is not the same as sampling from the population directly. Secondly, the less is sampled, the less is known about the data. It is not possible to estimate the accuracy reliably from a fold. In the case of LOOCV any estimate of accuracy from a fold is either 0% or 100%, so the result variability is huge. There can be a different take on all this. It makes more sense to understand the accuracy differently - not as measured for individual folds but the one obtained from all data. While the perspective is often given that N measurements of accuracy are obtained in a single pass through the data, from the point of view of all-but-one MFCV there is a merit in simply counting the number of successes across all data and dividing the result by the number of instances M - after all, averaging the accuracy across all folds is the same. This interpretation does not apply to the two-up MFCV where a fold is essentially a data-set of its own, that is to say, there is a series of samples of about the same size rather than one sample so subdivided.

The parity of training and test sets is important. It holds for the two-up MFCV, and virtually for LOOCV, but only to a degree for the all-but-one MFCV in general. Of course, maintaining the parity is economical as it optimizes the data availability for both training and testing. Also, having more of the same is appropriate statistically. Nevertheless, while the accuracy is generally understood as a result of testing of a classifier on a sample of data, the cross-validation actually subjects a classifier to a double-take whereby its ability to analyze and then to infer is tested. The same set could have been used to do both, but the impartiality requires that sets for training and testing were separate. In the absence of parity a correction is required when expressing full effects of the respective sources of influence on the result.

In MFCV instances are randomly assigned to folds. Because this determines the content of training and test sets, the accuracy readings may differ from one assignment to the next. For certainty, the procedure should be repeated. The by-fold accuracy estimation may give the impression that multiple readings are obtained in a single application of the procedure, and so repeating it is unnecessary. This may hold for the two-up MFCV but does not exclude additional passes. However, how many repetitions are possible? Even if to ignore the requirement for proportional representation of data classes in folds - and this is less consequential for the all-but-one than two-up MFCV - there is a limited number of ways the data can be assigned to folds. The master sample is limited - so are the possibilities. This number is certainly less than $M!$ which includes recurring folds. The randomness of data allocation to folds would prevent from folds recurring serially in the repeated MFCV. However, the proportion of recurring content is high. Even if current folds were simply renumbered - accounts for $N!$ subdivisions. This should be avoided or corrected for. Repeating LOOCV does not make sense at all as all folds get reproduced the second time.

Nonetheless, the problem of recurring content only exists because of how in principle the subdivision of data into folds is approached. Segmenting the sample randomly is not the same as circumnavigating all possible unique combinations of instances in folds over a course of repeated subdivisions. This would define the search space for the repeated MFCV.[2] This should be so as the procedure simulates sampling from population. As there are too many instances in a population, it is guaranteed that the content of any sample drawn is unique. Of course, the procedure can only draw instances from a representative sample of the population, not directly from it. For a fold-sized sample its uniqueness at the instance

level will only hold for N draws. In repeated subdivisions only uniqueness at the fold level is feasible. This will still drive the result change by fold, although its range will be reduced, reflecting the limited ability of a single sample to represent population. With the view that it is the average across all folds that counts in the all-but-one MFCV, it is possible to relax the requirement as to the fold content even further by allowing repeating folds, so long each subdivision is unique (up to N-2 folds may repeat). Needless to say, this will further reduce the result variability. Extending the search space like this is neither conventional nor necessary as it is already large enough. Besides, the extension would not be right by the two-up MFCV.

If all possibilities of subdivision into folds are exhausted - this is known as complete enumeration.[2] In this sense LOOCV is complete after a single pass. Generally, if M=q·N, $M!/(M\text{-}q)!/q!$ unique folds of size q can be generated. Assuming they can be assembled to complement each other, this number divided by N gives the number of possible subdivisions. From here, the smaller are the folds, the more likely their content will replicate in random subdivisions, explaining LOOCV. Note, the requirement for proportional distribution of class data into folds further reduces the number of possibilities; the above calculation has to be done by class and the results multiplied, to be precise. If M is large and N moderate, the number of possibilities is huge, nonetheless. It is a futile exercise encompassing them all even if this would prevent from the recurring content. Ideally, there should be measures in place, if not rendering the recurrence altogether impossible, then voiding random subdivision variants with previously encountered fold content, or at least making them a second choice.

1.2 Asymptotic success rate and its variance for large samples

The classification accuracy depends on the training set size. It increases with the amount of training data. Eventually, it reaches its asymptotic rate.[5] The pattern of approach to the asymptote is approximately linear with the inverse of the amount of training data.[5,6] This circumstance can be used to project the accuracy to a larger sample having obtained several readings for different sample sizes from available data.[7] However, the normal course of action in accuracy testing is actually the other way around. There are two sides to classifier performance: the accuracy and the speed. How a classifier performs on all available data is important for taking stock of its abilities. If the accuracy is high enough, the amount of training data can be reduced in order to expedite the performance.

In view of the above, the sample size should be fixed if the accuracy was deemed to be a constant and not a variable in the deterministic sense. Therefore, in order that the ground for classifier comparison was level, the number of folds for MFCV N has to be fixed. It is worth nothing that the overall number of instances M should be the same in comparison too, although usually this is a non-issue as there is only one representative sample of data.

In the conventional MFCV framework, if a classifier makes instance-wise the same predictions regardless of how data is redistributed between folds, the classifier is said to be stable.[2] A stable classifier reduces the accuracy variance to zero, as far as the repeated MFCV is concerned. Classifiers perform more consistently as training sets get bigger in size. However, the side effect of this occurring to a classifier when $N \to M$ is that the training set content variability becomes compromised. The stability of classifiers under LOOCV is altogether artificial because any redistribution of instances is going to produce the same folds. Yet, training sets there get increased to the maximum. While for a stable classifier reporting only the mean accuracy makes sense, for LOOCV the variance is simply unknown. Reaching the stability under the repeated all-but-one MFCV would attest to its propensity for underestimation of the accuracy variance due to the intensity of single sample data recycling. Generally, local methods, like DT-s, that base their decisions on subsets of training data, even though purposefully selected, would be exposed to result variation more than global ones, like NB, making use of all of the training set.

Many classifiers end up with a model of data concept after examining a particular set of instances. If the model is adopted for universal use, such a classifier is stable on any data sourced from the population.[4] The classification can then be viewed as a set of Bernoulli trials with the accuracy a_B being the true success rate, and the error $(1 - a_B)$ being that of failure. The accuracy on a sample represents then a random variable resulting from joint distribution of M i.i.d. (independently and identically distributed) Bernoulli variables.[1,2] The accuracy as a sum of such variables is therefore binomially distributed. Drawing parallels with the M Bernoulli trials, the sum is the number of successes encountered in a sample of that size, but having averaged over all instances, the true mean and variance (as squared quantity) of the sample accuracy a_M are expressed as in Eq.1.[1,2]

$$\begin{aligned} \bar{a}_M &= a_B \\ \tilde{a}^2_M &= a_B \cdot (1 - a_B)/M \end{aligned} \tag{1}$$

From Eq.1 the estimated accuracy is bound to vary less and less as the sample grows, and not only because of the size but also improvement in the predictive ability.

In accuracy variability analysis the rate of correct predictions is often assumed to be distributed normally. This is admissible since at large M the binomial distribution behaves more and more like the normal one.[1] To be precise, all the accuracy values are actually found between 0% and 100% marks. Also, the distribution, while unimodal, is not symmetric, and the higher the mean, the more it is shifted to the right.[8] However, the left tail is thin, and more of the right tail emerges when instances are added to the sample, so indeed it becomes bell-shaped for large enough M with quickly vanishing tails, which is typical of the normal distribution.

The above analysis requires that the classifier was stable, and applicable to MFCV this spells zero variance. Yet, at large M this result does not contradict Eq.1 and is expected since repeating the procedure can only underestimate the variance. However, this is controversial: a classifier rather not to be stable as only then the accuracy variance can be measured by the repeated MFCV (with specifics in the two-up case). Also, instability would mean larger variance. Will the assumption of normality of accuracy distribution hold? It still may because limiting a population to a single sample instead reduces the variance. The consolation is found in using larger training sets – they stabilize classifiers. That would be in agreement with the requirement for M to be large so that Eq.1 could apply. Of course, any training set is only a subset of the master sample. Therefore, for the all-but-one MFCV larger N are preferable, whereas for the two-up MFCV smaller N are preferable. To resolve the controversy properly, though, one should look for the innate connection between classifier training and testing on the same data that intuitively exists. Then, treating the training and test set pair as a unit can give a rough idea of the sample size in terms of Eq.1. Note that in the repeated all-but-one MFCV the test set is fragmented but amounts to all of the data, and at large N the difference in size between the training set and the all data can be neglected.

To compare performances of two classifiers on the same data, usually the paired t-test is used.[1,9] This involves obtaining the accuracy for exactly the same fold by both methods and taking the difference. The resulting statistic, aggregating the differences across all folds in all passes through the data, is then Student t-distributed. The statistic generally has the form $t = (m - \mu)/(s/k^{1/2})$ where m is the sample of differences mean, s is the sample standard deviation, and k is the sample size. A qualification is required as to the data the t-test actually handles: it is not the differences as such but their mean. Therefore, μ is the population mean of these means. The population variance is $1/k$ of the variance of underlying differences, approximated in the expression for t-statistic with s^2/k. For that reason it is unimportant how the differences are actually distributed – their mean and so t will tend to be distributed normally with $k \to \infty$ according to the central limit theorem.[8] The sample size determines the degrees of freedom - a parameter in the Student's probability density function (equal to k-1 the divisor in calculating s^2). The distribution approaches the normal one (with zero mean and unit variance) with more degrees of freedom in store.

In the given context, the true mean μ (of the accuracy difference size k sample means) is hypothesized to be zero and then, based on the value of the statistic, this is either accepted or rejected. The t-test result, or p-value, has to be small enough for the 'null' hypothesis to be rejected, as the divergence is then said to be statistically significant.

While using the t-test and its variations is widespread, there is a growing opinion that it is not ideal for the classifier performance comparison.[9] The main point is that it does not directly measure the probability of the null hypothesis or of its alternative. Indeed, the probability it estimates is that of occurrence of the aforementioned statistic absolute values as found or higher, conditional on the null hypothesis being true. Also, the test result is difficult to interpret as it does not translate to how the underlying accuracies of two methods are different.

An added convenience of the t-test is that it obviates the need to know the true variance, but this may be its weak point when k is small. In this connection, the fact that the accuracies are distributed approximately normally is important because the mean of their differences (given classifier independence of one another) is then also approximately normal, voiding the requirement for k to be large. Yet, the 'hypothetical' mean μ in the t-test is assumed to be known. It is set to zero in the paired t-test applicable to classifier comparison; however, this ignores that this value may much differ from the sample mean m. Can the test be relied on at high absolute values of the statistic?

On the implementation side, the pairing for the t-test requires exactly the same fold configuration in two algorithms being compared. Random folds cannot be reproduced, so a special set-up is in order, but this may be too much ado if the algorithms cannot be run outside their platforms.

LOOCV results are usually reported without any mention of variance. This is understandable since the fold content cannot be refreshed. Some alteration is required to test variability of the result under this approach. For example, instances can be withheld in turn with the procedure run on the rest of the data. Where classifiers are compared, with the accuracy obtained by MFCV, usually only the p-value is reported, or simply a

statement is made about the statistical significance of the difference in results. Sometimes, however, classification results include the variance alongside their mean. Knowing both for accuracy or error is useful for comparison with other published results or testing of new algorithms. Comparing two objects instead of two numbers has its challenges, though. How to approximately compare two probability distributions is described in the section on methods.

1.3 Lazy learners

In terms of computations some classification methods would be advantaged and others disadvantaged from using MFCV for their accuracy estimation. Some methods are 'eager' and others are 'lazy'.[1] The eager classifiers train before testing, the lazy ones test immediately. Both types make use of the training data, though. Yet, while the former try to 'learn' from it, the latter just 'mine' it. Albeit, these are the polarities and there can be approaches treading the middle ground. The decision tree (DT) variety generally belongs to eager methods; the nearest neighbor - to lazy methods.[1] Also, there are flavors: a DT can be run in lazy mode by working out only the branch that leads to the test instance; and a nearest neighbor in eager mode by reducing the training data to a set of critical instances before testing. The Naive Bayes (NB) classifier,[1] due to its simplicity, can be implemented as either eager or lazy method. Essentially, the choice between 'eager' and 'lazy' is driven by the application of either inductive or deductive reasoning. The inductive, or eager, methods produce a model of data - a compact description of the data concept. Therefore, with these methods MFCV can be very fast – all instances in a set get tested against the same model. The deductive, or lazy, methods have to mine for answers time and again, from one test instance to the next, although this is presumed being easily done and with added flexibility as to how. Yet, eager methods may come to a grinding halt on LOOCV due to effectively being forced into the lazy mode (while not being simple nor flexible).

In this work a DT is compared with a lazy variant thereof.[10,11] Also this is done across different platforms. The eager DT is J48 from WEKA[1,12] (implementing C4.5[10]) and the lazy DT is one proposed here and to be described. Additionally, the NB algorithm from WEKA is compared with another NB developed by own means. WEKA provides for the all-but-one MFCV but does not have an option to apply it repeatedly; however, this can be managed by reordering data instances anew, for which a utility exists ('Randomize' among unsupervised filters), before invoking the procedure. The partial results then need to be grossed up.

1.4 Lazy decision trees

DT-s can be run in lazy mode.[11] The design upholds the general principles of C4.5. However, since not all of the tree is required, the reckoning can be much simpler. Firstly, only the path leading to the instance in question has to be laid. In all-discrete domains the path is determined by certain attribute values and these are known from the instance, so only the attributes have to be selected. Secondly, no pruning is required since creating the model (whole tree) is not on the agenda, so no need to hone it. Thirdly, treating any missing values is easy - the attributes with missing values in the test instance are simply ignored. The last consideration brings to the fore that there can be better ways than relying on the same model (or a part of it) all the time. Creators of Lazy-DT[11] note that the Information Gain (IG) criterion for branch propagation, one attribute a time, in C4.5-like trees uses the averaged information pertaining to data in the current node, whereas a particular value of an attribute, otherwise nondescript, can be highly predictive in context.

Shorter trees are generally known to be more robust than more developed ones.[13] The strong attributes are usually taken up at the beginning, and the rest are not necessarily strong in subsets. Also, information dissipates down the branches (using the inverted tree / river system construct). The main limitation of the conventional DT-s is that they are univariate, that is, the decisions of how to dissect the data, although step-wise optimal, are based on a single selected feature.[14] This limitation manifests in a number of ways. Particularly, the loss of information is largely due to the "hill climbing" - a search strategy whereby the best path immediately in view is taken without the ability to backtrack to explore other options, also known as 'greedy' search.[1,11] As the sub-optimal decisions previously taken compound any subsequent ones, the information clatter intensifies. Contributors of dissipating information in DT-s are also the phenomena of fragmentation and replication.[11,14] The fragmentation brings about smaller subsets of data with reduced choices for decision making. Particularly, the ability to calculate IG precisely wanes as the branches become 'thinner'. It is a direct consequence of tree propagation. The replication is secondary to fragmentation in eager DT-s. It is often caused by the occurrence of independently strong attributes.[10,11] Since only one can be chosen at any particular time, a previously undervalued attribute becomes split-on in a number of nodes across the tree structure. Such a tree is wasteful in the form, and so according to the minimum description length principle[1] is not optimal.

The perspective of added flexibility is a major reason to opt for the lazy mode. This should not be missed where possible. For example, the Decision Branch (DB) algorithm exercised

by us previously[15] simply utilizes IG. The Lazy-DT, though, replaces the IG criterion with an entropy reduction one depending on attribute value in the test instance.[11] So much so, the dissipation of information remains a big problem with any DT. It has been observed that a DT is better off with data that has many interrelated features, none of them especially strong, than with data made of independently strong features.[10,11] Dissipation of information in continuous domains is offset by dissecting ranges in context, not once and for all, and also dichotomizing them instead of multiply dividing.[10] Prediscretizing data exacerbates the problem as a feature once selected cannot be split on the second time, thus limiting choices for narrowing the selection down.[10,15] Previously, parallels were drawn between DB and the Info-Neighbor - a nearest neighbor algorithm that weights features by IG.[15] Since nearest neighbor algorithms draw their conclusions from observing multiple features, this prompts a compatible hybridization of DB which calculations are otherwise based on a limited number although selected features.

2. METHODS

2.1 Distribution of data into folds

More formally, data-set D is subdivided into folds F_n ($n = 1 \cdots N$). Classifier C assigns instance x_m ($m = 1 \cdots M$) to one of the applicable data classes, which can be either successful or not, with the target class being y_m. If instance x_m is in F_n classifier C is trained on $D \setminus F_n$. The procedure is run R times with different subdivisions of data into folds. The accuracy estimate in run r is given by Eq.2.

$$a_r = \frac{1}{M} \sum_{m=1}^{M} I[C_r(x_m) = y_m] \tag{2}$$

In Eq.2 $I[\cdot]$ is the identifier function (also indicator function, also Kronecker delta) taking values 1 when its argument is true and 0 when it is false. The accuracy mean, second moment, and variance over R runs are defined as in Eq.3.

$$\begin{aligned} m_R a &= \frac{1}{R} \cdot \sum_{r=1}^{R} a_r \\ s_R a &= \frac{1}{R} \cdot \sum_{r=1}^{R} {a_r}^2 \\ v_R a &= \frac{1}{R-1} \cdot \sum_{r=1}^{R} (a_r - m_R a)^2 \end{aligned} \tag{3}$$

Similar to a running total, the quantities from Eq.3 can be obtained by updating from one run to the next as specified in Eq.4.

$$\begin{aligned} m_{(r+1)} &= \frac{1}{r+1} \cdot \left[r \cdot m_r + a_{(r+1)} \right] \\ s_{(r+1)} &= \frac{1}{r+1} \cdot \left[r \cdot s_r + {a_{(r+1)}}^2 \right] \\ v_{(r+1)} &= \frac{r+1}{r} \cdot \left[s_{(r+1)} - {m_{(r+1)}}^2 \right] \end{aligned} \tag{4}$$

In Eq.4 the values of mean, second moment, and variance for run $r+1$ are updated from the values thereof for run r. The reference to accuracy a is omitted for brevity. Of course $m_1 = a_1$, $s_1 = a_1^2$, and $v_1 = 0$ (inconsequentially for $r > 1$).

Instead of the class-indiscriminate accuracy in Eq.2 other objectives can be pursued such as individual class accuracies and their mean. The expression in Eq.2 has to be reformed appropriately. In the case of class-specific accuracies the summation is done over instances of a particular class and the result is divided by their number instead of M.

The above describes cross-validation schematics applicable to the all-but-one MFCV which this paper is mainly about. Changes that are required to fit in the two-up MFCV follow from the fact that each fold in this type is a representative sample of data in its own right and gets tested from one other fold, not the rest of the data. Formally, if instance x_m is in fold F_{n1} then classifier C is trained on fold $F_{n2} (n_2 \neq n_1)$. The summation in Eq.2 is therefore performed over instances in F_{n1} and the result is normalized with their number instead of M. Since testing of each fold $n_1 = 1 \cdots N$ is conducted against all other folds $n_2 = 1 \cdots N$ in turn, the number of accuracy readings is $N \cdot (N-1) \cdot R$ not R and so Eq.3 has to be changed to that effect, although Eq.4 can be used as is.

Subdividing data-set into N folds R times can be arranged for as follows.

Repeated random allocation of class instances to evenly sized folds

Firstly, the instances have to be distributed into folds randomly in each subdivision, while fulfilling the requirement for proportional class representation in each fold. Towards this aim, each instance is supplied with N counters of placement into respective folds. Also, each fold is provided with a counter of all instances distributed into it so far, irrespective of class. Then, separately by class, each instance is randomly distributed into one of the folds with the same minimum number of placements altogether, and within that range into a fold for which the minimum number of placements holds pertaining to the instance.

Secondly, to identify folds and guarantee uniqueness of their

content in subsequent subdivisions a reservation can be made as follows. For each data class randomly select k+1 instances, with k upper-approximating R/N, and assign them to fold 1. Repeat for folds 2 through to N. These controlling instances in each fold and class within fold are numbered from 0 to k. Instances with such index 0 never leave their fold. Instances with index k are rotated (unrelated to elsewhere in the text) from one subdivision to the next by shifting into the next fold, with instances in fold N sent to fold 1. Up to the index $k-1$ the instances remain static. Once instances k are going to return to their initial locations, they are released (become ordinary) and instances indexed $k-1$ commence rotating.

The placement of instances in the first procedure promotes even distribution of data between folds with the proportional or so representation of classes. It is so set up that the novel content could appear early in consecutive runs. However, this is similar to stratification and simply guarantees what the random distribution is set to achieve in the long run. The manipulation involving the small number of reserved instances in the second procedure prevents from the same content reappearing in the same or a different fold, as previously noted. Although, for large M and moderate N which is often the case, steps in the second procedure are only precautionary.

The first procedure overcommits instances if $N = 2$ resulting in idle content swapping between the folds one and two. To remedy this, the algorithm can be run with $N = 4$ and pairing folds one way or another before evaluating the accuracy. This may seem an artefact due to the constraint limiting affiliation of instances with same folds. However, while the previously encountered content is the least probable when $N = 2$, it is inevitable when $N = 1$ same as when $N = M$ - the transition accounts for another half of the data-set.

Neither the main procedure nor the preventive measures against recurring fold content are implementable when N approaches M. Particularly, it may be impossible to stratify the data by fold. Not that small folds are acceptable in the two-up MFCV. For the all-but-one MFCV, while it is feasible in the second procedure to draw instances irrespective of class, perhaps it is better to draw folds in random from the space of all possible combinations of q instances so as to complement the instances already drawn until all are accounted for. This would provide for one subdivision of data into folds. Needless to say, once selected, a fold becomes unavailable, particularly in subsequent cycles. When $N = M$ the space of folds is that of instances, which is exactly the basis for LOOCV.

2.2 Classifier performance comparison

2.2.1 *Probability densities*

The two, assumed normal, accuracy distributions arising from two different classifiers from the same data will overlap, as illustrated in Figure 1.

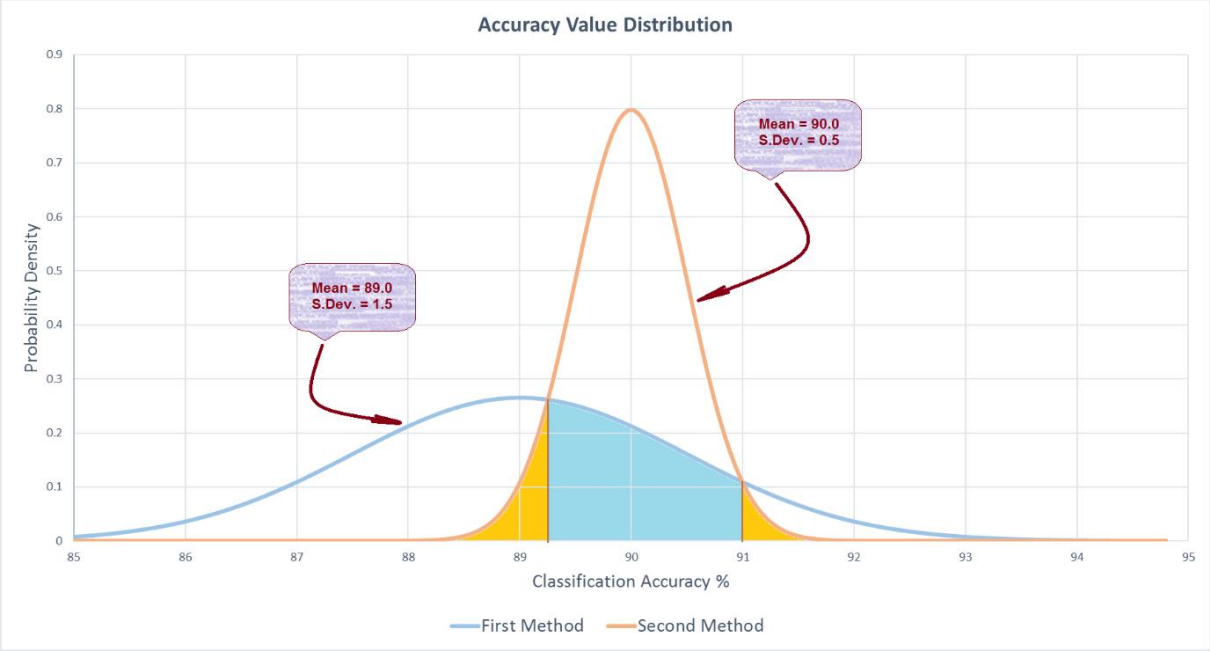

Figure 1. Two classification accuracy (%) normal probability densities (means ± standard deviations: 89.0±1.5 and 90.0±0.5), their intersection with individual contributions

In this connection, the primary goal is to establish where the probability density functional graphs intersect. The normally distributed random variable X taking values x has generally the probability density function $p(x)$ given by Eq.5.[8]

$$p(x) = \frac{e^{-\frac{1}{2}\cdot\left(\frac{x-\mu}{\sigma}\right)^2}}{\sigma\cdot\sqrt{2\cdot\pi}} \tag{5}$$

It is positive, bell-shaped, centered at μ - the mean value of X; with the symmetrical, vanishing at infinity, spread to either side controlled by σ^2 - the variance of X(see Eq.5 and Figure 1). The statement of normality of data distribution is often written as $X \sim N(\mu, \sigma^2)$ in the aforementioned terms. The property of any probability density function is that over the universe of discourse of values of its argument it integrates to 1 (total probability) which is the area under its graph. It is implicit that $\sigma > 0$ although the limit when $\sigma \to 0$ can be considered. By expansion, setting $\sigma = 0$ turns $p(x)$ into a delta-function: it is zero everywhere except at μ where it is infinite, and yet it integrates to 1 over all x.

The profiles of two normal probability density functions are bound to intersect a number of times from the above depiction of their shapes. Suppose $X_0 \sim N(\mu_0, \sigma_0^2)$ with density p_0 and $X_1 \sim N(\mu_1, \sigma_1^2)$ with density p_1 then it is required to solve $p_0(x) = p_1(x)$ for x. Noting that quantities on both sides of this equation are positive and taking the natural logarithm allows to reduce the full expression in terms of Eq.5 to the form in Eq.6.

$$\begin{aligned} &{\sigma_0}^2\cdot(x-\mu_1)^2 - {\sigma_1}^2\cdot(x-\mu_0)^2 \\ &= 2\cdot\ln(\sigma_0/\sigma_1)\cdot{\sigma_0}^2\cdot{\sigma_1}^2 \end{aligned} \tag{6}$$

Clearly, Eq.6 represents a quadratic equation with up to two real roots. In the standard form the equation and its solution are written as in Eq.7.

$$\begin{aligned} &a\cdot x^2 + b\cdot x + c_1 = c_0 \\ &\underline{a = 0} \\ &\quad x = (\mu_0 + \mu_1)/2 \\ &\underline{a \neq 0} \\ &\quad x = (-b \pm \sqrt{d})/(2\cdot a) \\ &\quad d = b^2 - 4\cdot a\cdot(c_1 - c_0) \end{aligned} \tag{7}$$

The solution of Eq.7 depends on whether $a = 0$ or not (unrelated to 'accuracy'). When $a = 0$ the equation is not quadratic but linear. The symbols in Eq.7 have denominations as per Eq.8.

$$\begin{aligned} a &= {\sigma_0}^2 - {\sigma_1}^2 \\ b &= -2\cdot({\sigma_0}^2\cdot\mu_1 - {\sigma_1}^2\cdot\mu_0) \\ c_1 &= {\sigma_0}^2\cdot{\mu_1}^2 - {\sigma_1}^2\cdot{\mu_0}^2 \\ c_0 &= 2\cdot\ln\left(\frac{\sigma_0}{\sigma_1}\right)\cdot{\sigma_0}^2\cdot{\sigma_1}^2 \end{aligned} \tag{8}$$

For the quadratic equation, the solution discriminant d in Eq.7 can be reduced to the form in Eq.9.

$$\begin{aligned} d = 4\cdot{\sigma_0}^2\cdot{\sigma_1}^2\cdot[(\mu_0-\mu_1)^2 \\ +2\cdot({\sigma_0}^2 - {\sigma_1}^2)\cdot\ln(\sigma_0/\sigma_1)] \end{aligned} \tag{9}$$

One can verify that $d \geq 0$ due to both multipliers in the last term of Eq.9 having the same sign. The discriminant turns into zero if $\sigma_0 = 0$ or $\sigma_1 = 0$ or both $\mu_0 = \mu_1$ and $\sigma_0 = \sigma_1$. Nothing of the three is attainable, which is down to $\sigma > 0$ generally, and $\sigma_0 \neq \sigma_1$ due to $a = 0$ representing the special case (see Eq.8). So conditioned, the two graphs then intersect precisely in two points. Because $\sigma\cdot ln(\sigma) \to 0$ when $\sigma \to 0$ appropriate limits in Eq.7 can be considered. When one $\sigma = 0$ but not the other the solution is unique, it is μ with the matching index. When both $\sigma = 0$ the solution is undetermined. In this case the two delta-functions intersect everywhere but the two points determined by μ values that have to be different. Note that small non-zero σ value handling, when the two are different, is precision demanding - the probability density function slope steepness makes area calculations in the next section sensitive to small errors in x, especially when μ values are close (see Figure 1).

2.2.2 *Likeness in results*

Surely any two numbers are either different or not. This corresponds to the case when both $\sigma = 0$. Although this case is exotic, the fact that two classification methods perform differently or exactly the same can be established directly from comparing their (mean) accuracies. Sometimes, the variance is simply omitted, and then comparing the means is the only option left, but the question should be asked whether the variances are same or different.

Generally, when comparing two waveforms, the degree to which their profiles overlap would give the answer to how they are different (see Figure 1). With this approach it is immediately clear that if the accuracy means as well as variances match then both classification methods perform equally well, in other words, they are 100% or totally similar. By contrast, the similarity would diminish and be 0% at the infinity when the accuracy mean values become more distant. Nonetheless, in the special case of one $\sigma = 0$ but not the

other, the two methods are 0% similar (totally dissimilar) regardless of μ values. To visualise the second, the two intersection points in Figure 1 become closer as the higher waveform grows thinner ($\sigma \to 0$) and taller vertically, while squatter horizontally, so the shaded area where the two profiles intersect vanishes.

For arbitrary μ values and non-zero σ values following steps can be taken to calculate the similarity:

Probabilistic similarity between normally distributed populations via T-z approximation

1. Obtain the one or two points of intersection of $p_0(x)$ with $p_1(x)$ using Eq.7 and order them by value if applicable.

2. Calculate the probability in tails of each probability density function on left and right of each intersection point.

3. Obtain the probability between intersection points for each density function by subtracting the values for appropriate two left or right tails. It is zero if there is only one intersection point.

4. Add up minimums of probabilities between the two distributions on the left of the first / single intersection point, on the right of the second / single intersection point, and between them if this applies (use Figure 1 for guidance). This is the value of the criterion, small as 0, high as 1; or 0% and 100% respectively.

In fact, the similarity criterion represents a distance between populations - like measure[16] and is expressed as follows (Eq.10).

$$P_S = \int_{-\infty}^{\infty} \min(p_0(x), p_1(x)) \cdot \mathrm{d}x \tag{10}$$

The convenience of Eq.10 is that the result P_S is probability-related with essentially the same meaning as the p-value in the t-test and others.[1,8,9] The above procedure stipulates how the integration can be done, but there is one difficulty.

The normal probability density function is not integrable in a closed form. Fortunately, for the probability in the tail of $p(x)$ there exists an old analytic approximation shown in Eq.11.[8]

$$T(z) = \frac{\sqrt{\pi/2} \cdot \mathrm{e}^{-z^2/2}}{(\pi - 1) \cdot z + \sqrt{z^2 + 2 \cdot \pi}}$$
$$z = (x - \mu)/\sigma \geq 0 \tag{11}$$

Eq.11 approximates the tail fairly well throughout, with the result at both ends $z = 0$ and $z = \infty$ exactly matching the true values: $T(0) = 0.5$ and $T(\infty) = 0$. Some logistics is involved in applying the expression when $z < 0$ and the tail is right or when $z > 0$ and the tail is left. One should take into account that in terms of z what is on the left of zero is the mirror image of what is on the right thereof, and the total area under the curve is 1.

If random variables X_0 and X_1 are normally distributed then, if they are independent of each other, their linear combination is also normally distributed with the mean and variance as in Eq.12.[8]

$$\begin{gathered} X_0 \sim N(\mu_0, \sigma_0^2) \\ X_1 \sim N(\mu_1, \sigma_1^2) \\ X_2 = a \cdot X_0 + b \cdot X_1 \\ X_0 \perp X_1 \Rightarrow X_2 \sim N(\mu_2, \sigma_2^2) \\ \mu_2 = a \cdot \mu_0 + b \cdot \mu_1 \\ \sigma_2^2 = a^2 \cdot \sigma_0^2 + b^2 \cdot \sigma_1^2 \end{gathered} \tag{12}$$

This classic result can be used to calculate the probability of one method being more accurate than the other, or how 'dominant' it is. Suppose probability density $p_2(x)$ results from subtracting X_1 from X_0. Then by Eq.12 $\mu_2 = \mu_1 - \mu_0$ and $\sigma_2^2 = \sigma_0^2 + \sigma_1^2$. All that is required to evaluate the probability of $X_1 \geq X_0$ is to integrate $p_2(x)$ for $x \geq 0$. The Eq.11 can be used again. This will work even with one $\sigma = 0$ of the two.

The requirement for independence needs consideration but is not so limiting. Generally, unless one of the two classifiers in comparison builds on results of the other, the requirement would be satisfied. The "all other conditions equal" clause concerning the applicability of both formulated methods should not escape scrutiny either. It was previously stated that the data as well as the number of folds for MFCV need to be the same. However, implicitly, applicable to the classifier comparison, the problem has to be one and only too. By the same token, it is feasible to compare results by a particular classifier but for different problems. If there are several representative samples of the same data, though, at least they have to be of the same size. If this holds then again, it is feasible to compare quality of data between them from the perspective of particular classifier as applicable to a particular problem. The same would apply to MFCV specification differences. All these aspects are relevant to the current work, although only the comparison of classifiers is pursued.

The Appendix instructs on using MS Excel to program either of the comparison methods for evaluation purposes.

2.3 Proposed lazy decision tree for discrete domains

Suppose the distance d in the space of nominal / discrete features is defined by Eq.13.

$$d(x_1, x_2) = \frac{1}{G} \cdot \sum_{g=1}^{G} I\left[x_{1,g} \neq x_{2,g}\right] \tag{13}$$

The expression in Eq.13 is known as overlap metric. As before $I[\cdot]$ is the identifier function; other notation has the following meaning: x_1 and x_2 are two vectors of features; g is the feature index and G is the number of features / 'genes'. The algorithm for DT in the lazy mode is formulated as follows.

Decision Branch Joint Venture

In: an instance to classify; Out: class of the instance

Parameters: min. size of the leaf; min. purity of the leaf

1. Consider narrowing the search by calculating the average distance to the test instance, using Eq.13, for subsets of data defined by that instance values of each attribute in turn. Select the subset (node) where the distance is the shortest.

2. If the selected subset is smaller than parametrically defined or selection is idle then proceed to Step 3. Else, accept the selection and exclude the corresponding feature from subsequent consideration. Repeat from Step 1 applicable to the current node. The initial node consists of all data in the training set.

3. Once the selection cannot be narrowed down more, so the leaf node has been reached, do one of the following:

- if purity of the leaf in respect of the biggest class in it is high enough, set parametrically, assign that class to the instance in question;

- if the purity is deficient, calculate the average distance from instances of each class in the leaf to the instance in question, using Eq.13. Choose the class closest overall to the instance to determine its affiliation.

3. Evaluation

The simulation conducted in this work uses data from the USA National Health and Nutrition Examination Surveys (NHANES)[17] for years 2011-14, consisting of 6860 records. Missing values were filled and continuous variables discretised. The population is older adults. Classification is performed in respect of Type 2 Diabetes Mellitus (DM), Cardiovascular Disease (CVD), or Hypertension (HT) statuses. The prevalence of DM, CVD, and HT in the population is roughly 20%, 40%, and 45%, respectively, based on the data. The features were preselected so they were by no means strong, neither were they weak features. All problems rely on their own, although vastly intersecting feature-sets. Fifty features are used to classify DM, fifty - CVD, and forty - HT. It is relevant to the current discourse that feature-sets in all problems are favorable for NB. This is the same data-set as previously reported.[18]

The surveys are conducted two-yearly. Each survey is an independent snap-shot of the population. Therefore, the data is comprised of two natural subsamples, referred in the text as first and second. The second data-set is slightly (under 10%) larger than the first. It makes sense examining them separately as this removes the artificial element introduced by subsampling from a sample rather than population. This is advantageous computationally but, above all, gives the opportunity to perform a reality check.

The exercise consisted of running the conventional 10-fold cross-validation 100 times. The repeated (all-but-one) MFCV was performed according to how it is described in the methods section, applicable to the proposed modified DB algorithm or own implementation of NB. The repeated MFCV was simulated for algorithms tied to the WEKA platform by repeatedly launching them after randomly reordering data, all-in-all 100 times, with 10-fold cross-validation option. The classifiers on WEKA are J48 and NB.[1, 12] The predictive ability of algorithms via resampling is assessed for each diagnostic problem – dataset combination out of the three and two aforementioned, respectively. During each run of MFCV a number of accuracy aspects are evaluated. Particularly of interest are the sensitivity, specificity and the balanced accuracy. The first two are accuracies in respect of either the cohort diagnosed with a chronic condition that applies, or the healthy controls, respectively. The balanced accuracy is their simple arithmetic mean. The accuracy reporting and the comparison between classification methods is based on the last values out of the repeated MFCV. Background dynamics of the value change through the exercise is analyzed.

4. Results

4.1 Comparison of imprecise

The comparison of classifier performances on different platforms is presented in Tables 1&2 for NB and DT classifiers, respectively, on the basis of balanced accuracy mean and standard deviation (square root of variance). The columns in the tables have the following designations titled accordingly: 'Same' uses the criterion expressed in Eq.10; 'More' applies Eq.12 to calculate probability of $X_1 - X_0 \geq 0$ where X_0 and X_1 correspond to the results under 'Weka' and 'Own', respectively; 'Less' is complementary to 'More', evaluating probability of the opposite. Unlike for DT, the results for

NB look much more synchronized, whether in respect of similarity or dominance as the criterion, which is plausible since, notwithstanding the environmental differences, the algorithm should be the same, give or take.

NB exploits the conditional independence of features, given a class, and loses traction where this does not hold. Yet, it is regarded as a 'parametric' method because it makes an assumption about involved probabilities (parameterizes a model distribution, as it were). As the result, it effectively harnesses more data than, for example, DT which is 'non-parametric'. So, if the assumption holds, NB may perform better than DT. On the other hand, NB is known for its ability to handle class imbalance.[15] This is because it gleans the information from all over the instance space. The higher balanced accuracy seen in Table 1 for DM than for CVD or HT can be a confluence of favorable circumstances for NB on this problem.

Table 1. Balanced accuracy (%) standard deviation bounds and p-value for own NB being in the same range as, or jointly more/less than the Weka equivalent

Problem/ Sample		Weka	Own	Same	More	Less
DM	1	84.0±0.2	84.3±0.2	0.448	0.857	0.143
DM	2	86.0±0.2	86.1±0.2	0.795	0.642	0.358
CVD	1	77.5±0.1	77.5±0.1	1.000	0.500	0.500
CVD	2	78.3±0.1	78.2±0.1	0.610	0.237	0.763
HT	1	76.6±0.1	76.6±0.1	1.000	0.500	0.500
HT	2	78.6±0.1	78.6±0.1	1.000	0.500	0.500

Table 2. Balanced accuracy (%) standard deviation bounds and p-value for own DT being in the same range as, or jointly more/less than the Weka counterpart

Problem/ Sample		Weka	Own	Same	More	Less
DM	1	70.0±0.7	81.3±0.6	0.000	1.000	0.000
DM	2	75.7±0.9	87.4±0.5	0.000	1.000	0.000
CVD	1	81.1±0.5	77.5±0.6	0.001	0.000	1.000
CVD	2	81.9±0.5	82.5±0.5	0.542	0.804	0.196
HT	1	73.9±1.0	75.8±0.5	0.189	0.956	0.044
HT	2	75.1±0.7	84.3±0.4	0.000	1.000	0.000

Although inconsistently, J48 from WEKA is lagging behind DB-JV across the problem spectrum and the data samples. The proposed lazy DT differs in design from J48 well beyond its basic transformation from 'eager' to 'lazy'. Evidently, J48 lacks capabilities to deal with the class imbalance.[18] So, at least for DM, the balanced accuracy results in Table 2 are much different.

For a closer look into the balanced accuracy differences in the case of DM, the sensitivity results, including confidence intervals, are presented in Figure 2 for DT classifiers on different platforms.

As seen from Figure 2, there is an appreciable contrast in sensitivity between the proposed DB-JV and J48 DT from WEKA. This is verifiable even without the apparatus of p-values. Indeed, the upper bound of the 99.7% confidence interval (three standard deviations) of the assumed normally distributed sensitivity for J48 is clearly less than the lower bound of the analogous interval for DB-JV in both samples (consult Figure 1). DM is much less prevalent than CVD or HT. The proposed lazy DT is better equipped than the standard one to deal with a possible class imbalance by absorbing the contained in leaves secondary information passed on to them by progenitor nodes. However, the Table 2 results for DB-JV, while acceptable, do not compare favorably with previously reported ones by other methods for the same data,[18] particularly the earlier mentioned Info-Neighbor which uses IG weighted overlap metric from Eq.13.

In DB-JV the leaf size and its purity were set to be no less than 5 instances and 90% or more, respectively. When running, the level of detail at leaf ('length' of the branch) was between 1 and 5, but 3 on average, with zero corresponding to all training data. The NB of own making does not require a parameter entry. The algorithms on WEKA were run with their default settings, although neither J48 nor NB seemed to attract any parameters of critical importance.

Between the samples, the accuracy by the same method is notably different (see Tables 1&2, Figure 2). The accuracy is always higher for the second sample, regardless of the method and problem it is applied to. Although completely foreign to each other, the samples are about the same size. Size can impact accuracy, but the second sample is only slightly bigger, and it is likely the accuracy is close to asymptotic. This can only mean the data is not homogenous. The two samples as they appear in the surveys[17] are different in attributes they list. Where a misalignment occurred, the missing values had to be filled. As a statement, features behind the attributes native to the first sample are more relevant to the represented problems. Because of the imputation these features may have become elevated in relevance in the second sample, which is a known issue as the surrogates usually seek to minimize the classification error.[15]

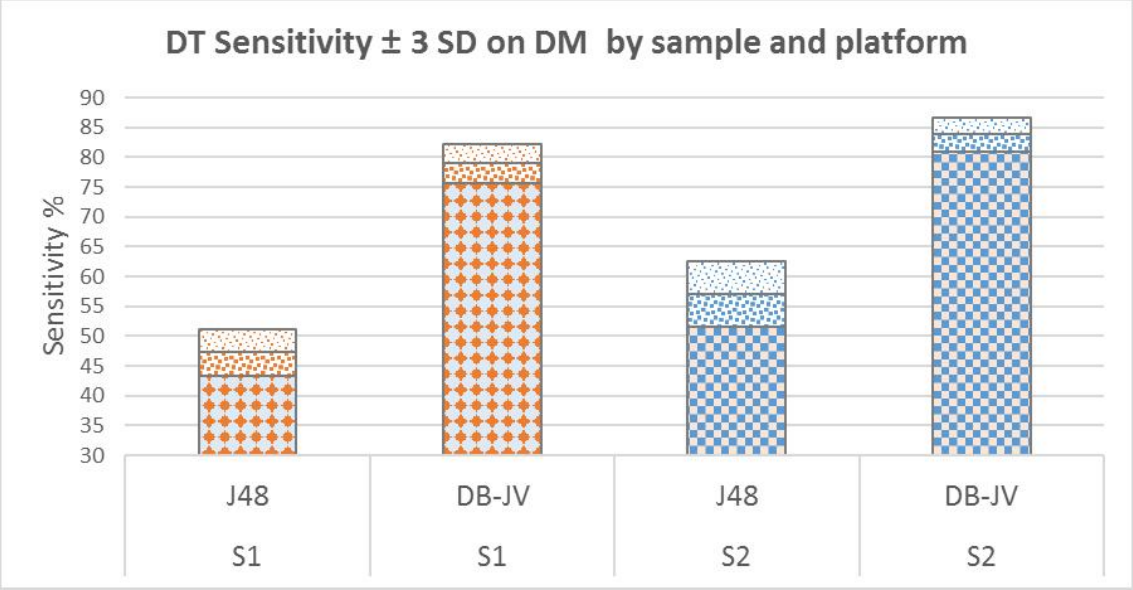

Figure 2. Diagnostic sensitivity of platform dependent DT classifiers on DM by sample

The results in Tables 1&2 are consistent with the theoretical accuracy variance being smaller for larger samples, according to Eq.1. It is not possible to do calculations exactly without adapting Eq.1 for the balanced accuracy, but this metric approximates the situation when samples are class-balanced,[18] that is, there are equal numbers of any class. By seeing Eq.2 in this light it is easy to verify that Eq.1 can then be used as is. By the way of some underestimation of the variance, suppose the sample size is 6400 (about twice the size of the data-set for compatibility with the equation) and the accuracy is between 90% and 50% then by Eq.1 the standard deviation is between 0.4% (3/8) and 0.6% (5/8), respectively. If the sample size is 3600 then the theoretical standard deviation range is from 0.5% (3/6) to 0.8% (5/6), which is an overestimation but the interval change is minor. From Table 1 the standard deviation for NB on any platform is 0.1%-0.2%. From Table 2 the standard deviation for DB-JV is 0.4%-0.6% and for J48 it is 0.5%-1.0%. At least, the theoretical and experimental results are comparable. Although, the repeated all-but-one MFCV is expected to underestimate the actual variance. This certainly holds for NB and to some extent for DB-JV in respect of the theoretical one. NB results vary less than in any of the DT-s. This can be explained by NB extracting information from all data (enabled by its assumptions), not locally as the DT classifiers.

4.2 Background dynamics

References to ten-fold cross-validation for accuracy evaluation are found time and again in the literature. Usually, it is also ten times repeated.[1] However, the latter is an open question.[3,4] Nether the current demonstration offers a stopping criterion for iterations. However, it is admissible that the fold content undergoing dramatic changes from one iteration to the next is able to offset the result-so-far markedly. The accuracy mean (and so the variance) solidifies the more iterations are performed. Therefore, if large content changes are pursued from the very beginning, while the mean is still poorly established, it can be expected the accuracy will stabilize sooner than otherwise. Clearly, the MFCV capability in WEKA provides only for a single pass through data, and so leaves no clues for subsequent runs. By contrast, the proposed algorithm for repeated subdivision into folds promotes fast content changes by redirecting instances to folds, they have least visited, after each run. It also guarantees that content of each fold is unique throughout the exercise. With this preamble, the following exposition is intended for better understanding of a potential stopping criterion for the repeated MFCV.

The balanced accuracy coefficient of variation (standard deviation to mean ratio) change through time is shown in Figures 3&4 for DB-JV and J48 classifiers, respectively, with the own NB and one from WEKA trending similarly to either of the DT-s. The quantity mean between steps 10 and 100 is also shown to provide a reference level.

All graphics in Figures 3&4 show clear signs of plateauing over time. This is expected as the values of both the accuracy mean and standard deviation become ‘heavier’. How the coefficient changes from iteration to iteration is purely circumstantial, driven by random subdivision of data into folds. The pattern of change can be oscillating, increasing, decreasing - however, it is largely meaningless. It can be different

next time around. Compare the curves for different samples but the same problem - often they are much unlike the other. The convergence to the true ratio is stochastic. Note, the line depicting the coefficient mean is not the asymptotic value of the parameter.

Figure 3. DB-JV balanced accuracy coefficient of variation MFCV time series and its 10-100 mean for different problem-sample combinations

The convergence to a solution is manifested in the magnitude of value leaps and the time they occur. While rapid, high amplitude changes mostly cease within 40 iterations, less intense flare-ups in some cases can be observed even after 70 epochs of repeated MFCV. In this sense 100 rounds of the procedure does not seem too many. How fast the convergence is - would depend on the classifier, and from Figures 3&4 one may conclude that J48 does better, but there is a scale difference with charts for DB-JV. However, if smooth transition were to signify convergence, this could be because J48 is a blunter instrument than DB-JV. At the same time, the perceived smoothness could be simply a result of the MFCV version tied to J48 being less radical. More properly comparing the convergence depending on the repeated MFCV actual procedure can be done for NB since, apart from computational preferences, the classifier is hypothetically the same in both environments. From Table 1 the difference in results is not statistically significant across the problem spectrum, in the p-value parley, to say the least. If this means that both results may be close to their asymptotic level after

100 runs then one may look closer to the outset for a trend (these charts are not shown). However, the proposed is hardly useful without stopovers to measure the differences pairwise.

Across the board, the coefficient of variation is small, actually less than 1% for all classifiers but J48 where it is no more than 2%, in the time frame pertinent to Figures 3&4. Evidently, the fluctuation decreases over time, becoming even less in magnitude than the perceived value of the coefficient of variation. Therefore, it is almost immaterial for one decimal place results as in Tables 1&2. There is a connection between the coefficient and how it changes. Provided the accuracy mean and standard deviation has stabilized, one can roughly estimate whether another repetition is able to impact the result significantly. Suppose the mean accuracy is 50.0% and the next measurement is as radical as 0.0% or 100.0%. From Eq.4 after 100 iterations this is able to bring 0.5% change about to the mean, and so the last digit will be affected. Likewise, after 1000 iterations only 0.05% change is possible - the dependence is inverse. So, in real terms, the result with one decimal place is not going to be affected as the changes have to be assumed going both ways. If the number of iterations is small, though, one should not count on opposing changes occurring intermittently, or rather that leaning one way can be sustained for long. However, the results in Tables 1&2 can be stable enough even after 100 iterations. The standard deviations are small relative to corresponding means. Even if a change of three standard deviations is considered, still it will not affect the last digit. This is easily verifiable by multiplying the standard deviations by 3/101. The biggest standard deviation in Tables 1&2 is 1.0% and yet, if the next measurement of accuracy is as rare as described, its contribution will be only 0.03%.

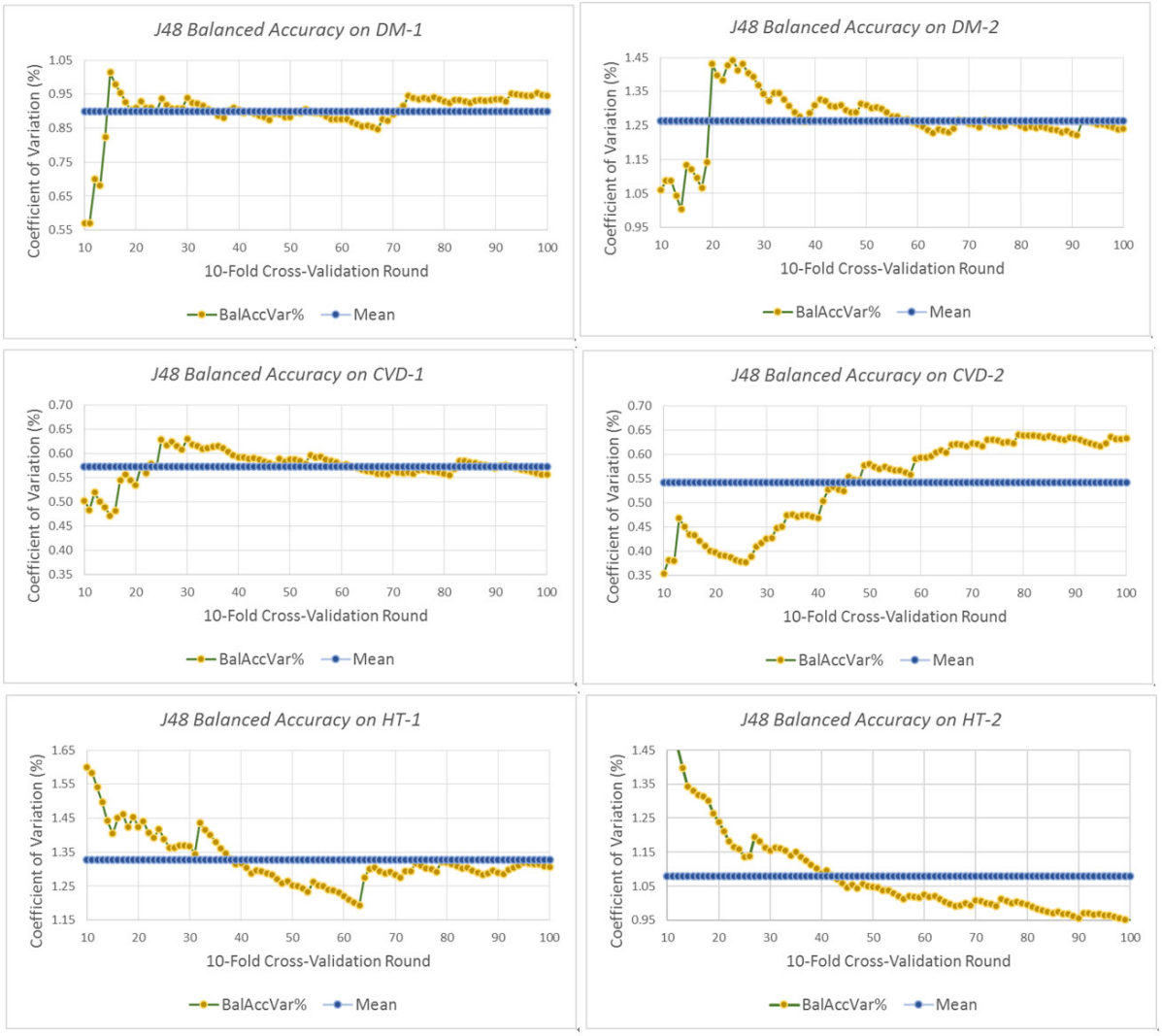

Figure 4. J48 balanced accuracy coefficient of variation MFCV time series and its 10-100 mean for different problem-sample combinations

5. DISCUSSION

5.1 Performance computation-wise

The NB on WEKA executes much quicker than the algorithm to NB specifications of own making. It is likely the version on WEKA is implemented in the eager mode, not the lazy mode as its vis-a-vis. This impression stays notwithstanding the interaction with environment to perpetuate MFCV on WEKA. The same is true in respect of J48, but it outputs a tree structure, so it ought to be an eager method. Also, WEKA has a dedicated entry for lazy methods in its methods directory, but NB and J48 are found elsewhere. Eager methods are generally faster than lazy ones in the course of MFCV, as previously noted. Of course, the lazy DB-JV differs much in design from J48 to conduct the speed comparison with confidence.

One way or another, from the user point of view a quick analysis is a virtue. Both NB and J48 on WEKA are set up with some parameters that regulate their performance but are not specific to the methods. So, irrespective of the above, a question may arise whether WEKA does a proper job or runs a demo. If it is the second, then how can this be enabled? One answer is: by sampling from the training set. If the data is as abundant as in the current work examples, this may represent a viable option. The treat is universal, and therefore applies to eager and lazy methods alike.

5.2 Other approaches to resampling

Other known resampling plans to measure classifier accuracy are as follows. In resubstitution the classifier is trained and subsequently tested on all available data.[1,2] Only one measurement of accuracy can be obtained. It is regarded as an optimistic method of assessment because the testing is performed on data for which the classifier is optimized. Also, more data is used for training than in any other resampling plan. Despite this advantage, there is a danger of overfitting which is a phenomenon of classifier failure on any new data. Classifiers tune to data, but this has to be on the whole, and with a single sample this aspect is largely neglected. Additionally, the more parameters classifier has, the better it can flex to accommodate data singularities, but they may represent rare exceptions or a spurious content. These singularities would appear at a reduced rate or not at all in a new data. On the other hand, due to a trade-off between the parameters when tuning, even the data bulk in a different sample may appear foreign to the classifier.

The resampling plan where a fixed portion of data is randomly reserved for testing and the rest is used for training is known as 'hold-out'.[1,2] Usually the procedure is repeated a number of times. Often the test data withholding rate is 1/3 but it can be any, which offers a control more flexible than in the two-up MFCV to test the classifier ability to learn. This procedure is statistically sound, and more so if data is stratified, but it does not guarantee that all instances in the representative sample get methodically tested. It compares to estimating accuracy from a fold rather than all data in the all-but-one MFCV. Therefore, the result can be expected to vary more.

Bootstrap methods[1,2] are based on sampling with replacement. Using the notation in the text M instances are randomly drawn from the representative sample of the same size to form a training set. The classifier gets then tested on instances that were not drawn. This is repeated R times. On average, there are approximately M/e test instances. This translates to 36.8% effective withholding rate in respect of test instances (although, this common interpretation of the original result may be an overstatement[7]). The utility of this technique is often noted in relation to sparse data.[1] In small data-sets adding more of the same (explaining the name choice for the technique) can help to classify correctly test instances that are lacking the support. Even in larger data-sets, where the correct instances would have a sufficient support, withholding too much for testing would undermine foundations of the data. So, the replication may be beneficial even then. Nonetheless, the accuracy estimate so obtained is deemed pessimistic, nothing like the estimate by resubstitution which is optimistic. Also, not all training instances are necessarily genuine, some may be unexplainable by applicable feature-sets. The negative impact of replication in this regard is similar to the overfitting in resubstitution. Ways are sought to balance the accuracy estimator by combining it with the resubstitution success rate while being mindful of overfitting risks.[7,19] Also, a bootstrapping scheme where data is stratified by class does not seem impossible.

The bootstrap is particularly different from the hold-out in that the test data withholding rate is variable. The repeated hold-out is sometimes referred to as Monte-Carlo cross-validation.[3] However, the term equally applies to the bootstrap as well as to MFCV.[2] In classification the purpose of resampling is to produce a series of test and training set pairs to enable cross-validation (not necessarily reciprocal). The classification accuracy is found by validating a set, that is, by comparing its instances actual class affiliations with the projected ones. Since a computer simulation using pseudo-random numbers is involved - this is a Monte-Carlo method.[20] Essentially, the random resampling for classification accuracy testing is a Monte-Carlo simulation whatever are the specifics. 'Resampling' implies 'repeating'; although, applicable to MFCV, there is more to it.

A number of comparison studies find the conventional MFCV,

which may include repetitions, more appropriate for accuracy estimation than any of the resampling techniques cited above.[2,3]

Comparison of classifiers has generally much more to offer than a pair of classifiers on a single data-set can hope for. Ultimately, how one knows what algorithm to choose? The generality of this question implies experiences involving various domains of data. The experiences form a basis for meta-data that can be analyzed. There are ranking methods to compare performances of multiple classifiers on multiple data-sets.[21] A performance generalization of individual methods in various applications is achievable through devices of the Bayesian analysis.[9] However, comparing two classifiers on the same data paves the way for a comparison on the grand scale.

5.3 Rotation vs conventional

Figure 5 gives the idea of the two-up MFCV type. The MFCV by rotation was run once with 10 folds, thus producing 90 serial measurements of accuracy. WEKA does not offer this capability.

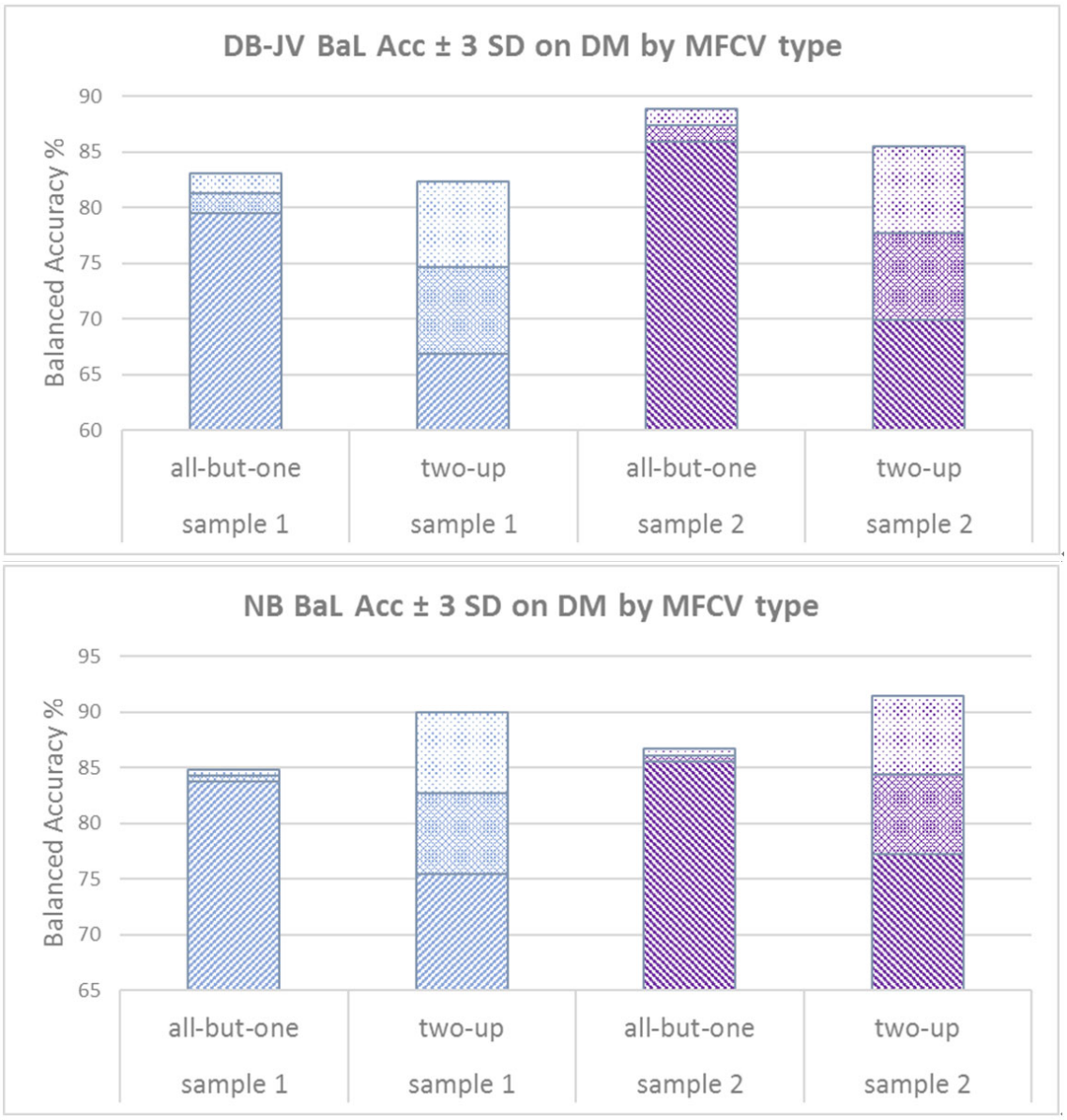

Figure 5. MFCV type dependent balanced accuracy of DB-JV and NB classifiers in DM diagnostic

The balanced accuracy so obtained differs in two respects from the one by the conventional 100 × 10 MFCV as in Tables 1&2. Firstly, the mean is less than otherwise because the training sets are much smaller. Secondly, the variation of results is much higher as both training and test sets are segment-wise independent in their midst in addition to being 9 and 10 times smaller, respectively, than in the repeated all-but-one MFCV. These observations are no different for DM, CVD, or HT; therefore, only the results for DM are shown.

Interestingly, despite being so reduced, the training sets exhibit a great deal of resilience as the mean accuracy seen in Figure 5 is still meaningful. Also, the wide confidence interval for better chart legibility belies the effective 'amplitude' of changes; but even two standard deviations off the mean already account for 95.5% of normally distributed data (see Figure 1). The accuracy degrades more for DB-JV than for NB, which has to be on the part of NB being a parametric method, as previously noted, and so effectively accessing more data than DB-JV. In this connection, some of DB-JV traction may have been lost as the subset hierarchy it was yielding was 1 level less, both on average and binding, than in the all-but-one MFCV, while with the same parameter settings.

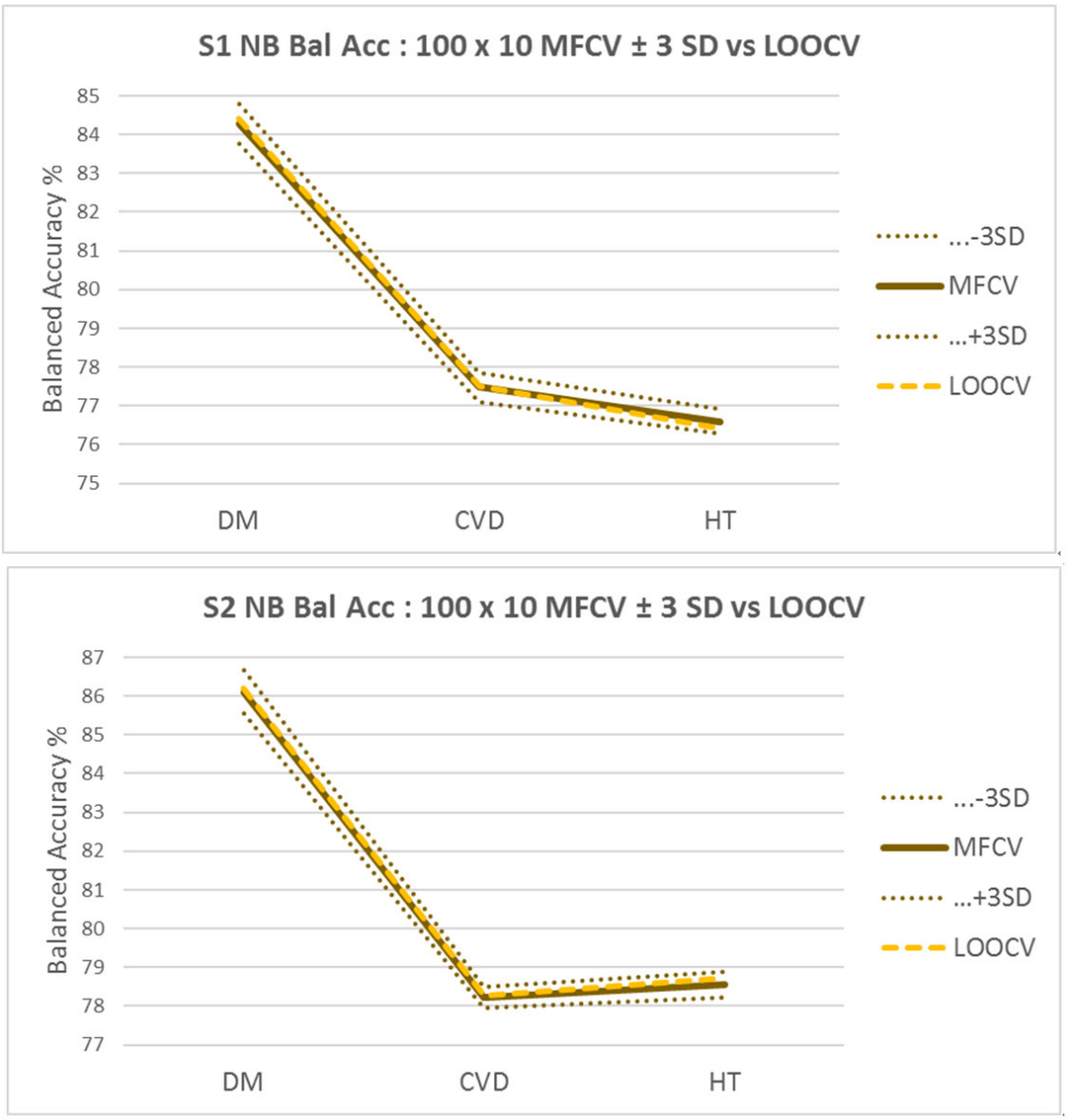

Figure 6. NB balanced accuracy and confidence interval by repeated 10-fold or leave-one-out cross-validation by problem for samples 1&2

The two-up confidence intervals are not only wide but also about the same for DB-JV and NB, as evident from Figure 5. The first is explainable by the training and test set pair variability and their size. Of course, a fold-sized sample is no match to all data, but it is not the instance count that matters, instead the second may well have to do with the lost perspective when sampling from the representative sample of, rather than population. This can be a source of artificial noise that may have caused the marginalization of classifier ability for error correction, particularly in NB.

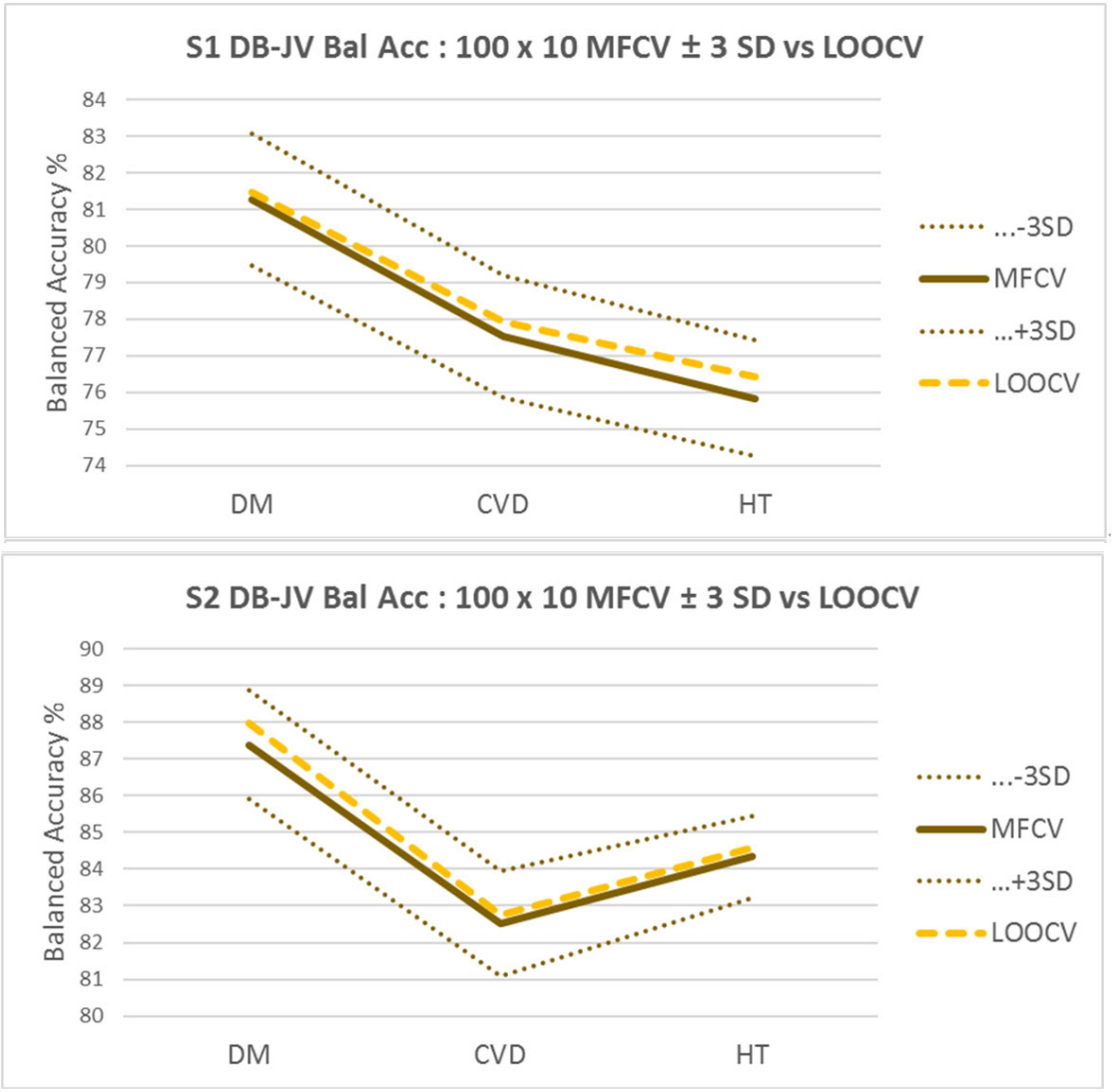

Figure 7. DB-JV balanced accuracy and confidence interval by repeated 10-fold or leave-one-out cross-validation by problem for samples 1&2

However, the two-up result variation, as seen in Figure 5, is in principal agreement with Eq.1. For example, for a training and test sample pair of 900 instances the mean accuracy of 90% translates to the standard deviation of 1.0%, and of 50% to 1.7% (5/3). The observed standard deviation is in the vicinity of 2.5% while the sample is somewhat smaller. For 400 instances the theoretical range so defined is from 1.5% to 2.5%, indicating high sensitivity of the standard deviation to the size change when sample is small. Generally, since the two-up MFCV engages multiple samples, experimental result variance for it should be in better agreement with the theoretical one than for the all-but-one MFCV that is limited to a single sample. However, this may be difficult to judge due to data distortion that occurs at the fold level, as was pointed out.

5.4 Leaving one instance vs one fold out

Comparison of results from Tables 1&2 with LOOCV is graphically represented in Figures 6&7 for NB and DB-JV, respectively. While this variant of the all-but-one MFCV can be enabled on WEKA by entering the number of instances for the number of folds, the current comparison does not involve the platform.

LOOCV closely follows the mean result of 100 times applied conventional 10-fold cross-validation, with the balanced accuracy by LOOCV typically higher but within one standard deviation, or so, from the aforementioned mean. In this connection, DB-JV with LOOCV was reaching the leaf level in about the same number of steps as with 100 × 10 MFCV, parameterized no differently. The likeness in results is more discernible in Figure 7 than in Figure 6 because the three standard deviations confidence interval, as applicable to 100 × 10 MFCV, is much wider for DB-JV than for NB. The NB result stability is extraordinary, drawing attention once again.

LOOCV uses about 10% more data for training than 10-fold cross-validation, which can explain the observed in Figures 6&7 difference in results. For lazy methods on large data LOOCV offers a much better deal since computationally it compares to a single MFCV run.

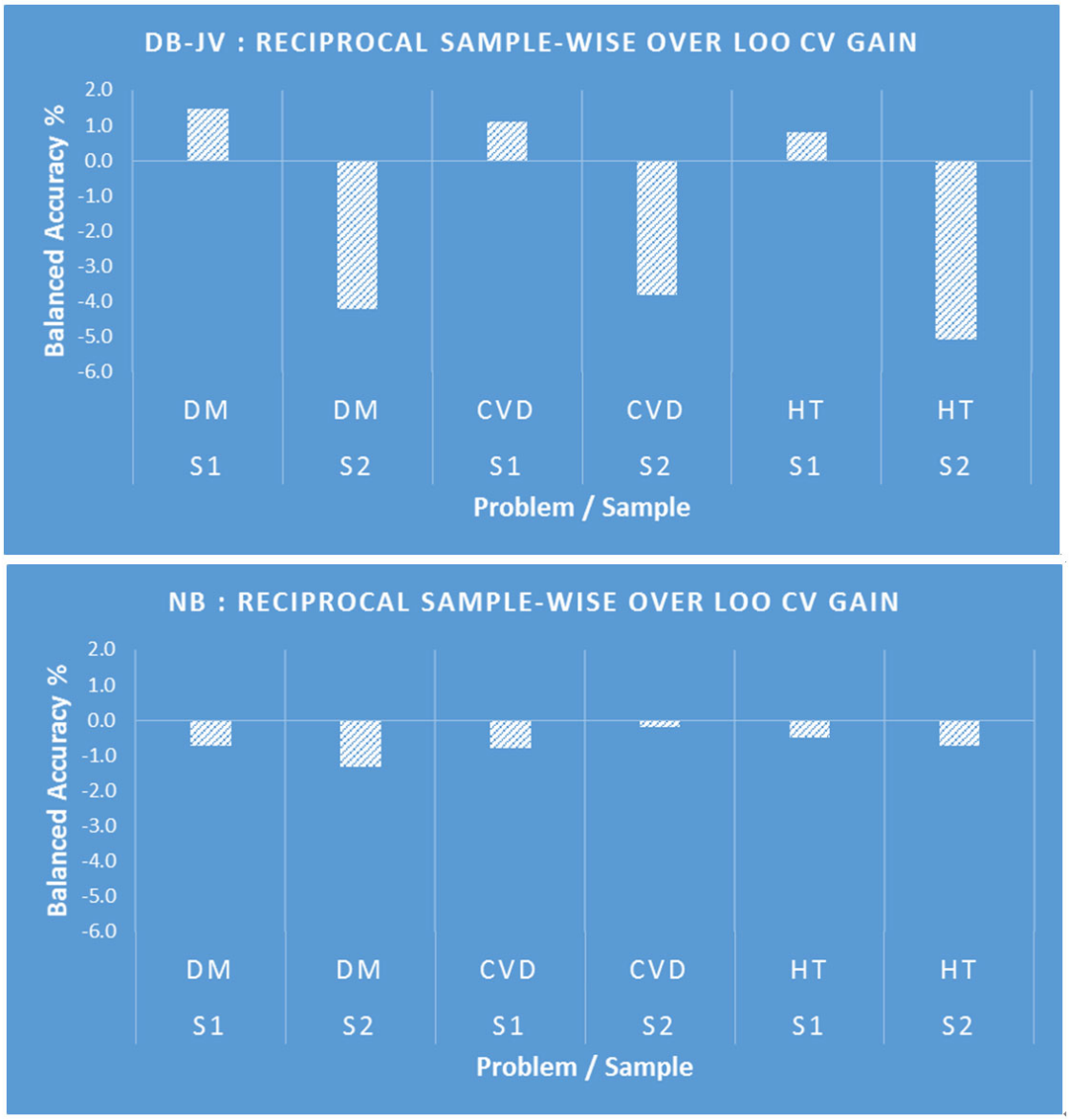

Figure 8. Opposing-sample-validated over same-sample LOOCV balanced accuracy gain/loss for DB-JV and NB classifiers

LOOCV is regarded as an optimistic estimator. Indeed, the results suggest this may be the case. Firstly, the LOOCV training sets are the biggest in the MFCV range. Secondly, LOOCV does not seem to be much different from the resubstitution which is optimistic. For example, with NB, modifying the training set by removing one instance and adding another will not cause a substantive difference to the involved feature value probabilities by class in a large data.[18] Some apprehension about LOOCV exists, though, and this may be because of its proximity to resubstitution.[2,3] Theoretically, LOOCV may fail on noisy data, due to overfitting, if the classifier is unstable. Figures 6&7 suggest that NB is more stable than DB-JV, although both have the ability to handle the class imbalance, characteristic of DM, and this is a condition similar to noise.[15]

5.5 Wild-card testing

The ultimate purpose of accuracy testing is to build confidence in practical utility of a diagnostic tool. Therefore, it is sometimes proposed as a challenge to test a model on a benchmark set reserved solely for that purpose. Applicable to the examples in this work, there are two sets of data obtained independently of each other by sampling directly from the population - the surveys are run serially and as new campaigns.

The results of testing of one sample using the other for training are represented graphically in Figure 8 as a gain/loss over LOOCV in terms of balanced accuracy differences.

The comparison with LOOCV is appropriate as the amounts of test data are exactly same, and the training sets differ in size by less than 10%. From Figure 8 the following is evident across the problem spectrum. For DB-JV there are balanced accuracy 1%-2% gains at best when the second sample is used for training. As per previous, it is the sample which is more self-consistent and also slightly bigger than the first one. The fortunes are other way around when the first sample is used for training, amounting to 5% - 6% loss at worst. In the case of NB the accuracy is lost throughout but by little, 1% - 2% at worst. As per previous, NB is less sensitive to data changes than DB-JV. These results are encouraging notwithstanding having had the surveys aligned. More so that LOOCV is an optimistic estimator.

6. Conclusion

Nowadays much more is available in terms of data access and computational power than just few decades ago. Understanding of requirements as to the classification accuracy / error estimation is changing. Previously the emphasis was on small data-sets. The size was not so limiting as to undermine the predictive ability of a classifier. The real challenge was to get a peek of what is beyond, should more data become available. The concern was that the confidence in results from the same model would diminish. This energized the quest for accuracy scatter estimation, and in this connection finding ways of imprecise result comparison. However, even theoretically, the more data - the less variation. The results in this work actually confirm this finding. Be it as it may, but does this make the accuracy interval estimation methods redundant? While a simulation can be performed on a large sample, realistically the training set has to be small as only then the classifier is fast. How to correct for this belongs to the online learning and is beyond the scope of this paper, but reducing the heat around the issue of accuracy estimation seems justifiable, and so the paper contributes two simple complementary techniques to compare results by different classifiers from the same data. Although, how many iterations of multi-fold cross-validation are exactly required to establish classification accuracy mean and standard deviation with confidence – may not have a short answer. It is not a fixed number, neither it is small, what seems to be accepted in the literature. If anything, it will certainly depend on the precision required. Clearly, it depends also on the amount of data available. However, classification methods may be more reliant or less reliant, which can impact on the convergence greatly. Although, more accurate methods should be also more reliant.

In the part of evaluation of the methodology for classifier comparison this work features a lazy decision tree algorithm. Decision trees in discrete domains lack flexibility and the lazy approach can remedy that. At the same time, dissipation of information in any tree accounts for much of their underperformance. A hybrid with the nearest neighbor method is proposed to counter the loss. The scheme shows some success over a state of the art decision tree on a publically available server. A proper performance comparison across computational platforms is hardly possible without enlisting the back office capabilities, yet the proposed methodology makes it easier.

7. Appendix

Attached are four text files that will open in MS Excel and can be made into a workbook. The worksheets contain formulae to calculate the p-value of two classification methods being close, or the probability of one performing better or worse than the other. Files ‘Same1.csv’ and ‘Same2.csv’ utilize Eq.7&11 to calculate the similarity between two methods based on their performance. The first file applies to the case when both methods have the same standard deviation of their accuracy. The second file accommodates standard deviations that differ. The file ‘MoreLess.csv’ makes use of Eq.11&12

to ascertain the probability of one method performing better than the other.

These three files share some essential notation. The columns 'M1', 'S1' and 'M2', 'S2' set the means and standard deviations of accuracy of the first and second classification methods, respectively. The only place where any input occurs is immediately below these headings. The entries correspond to μ_0, σ_0 (first method) and μ_1, σ_1 (second method) found in the text. The column 'Sensible' runs the check whether the standard deviations are the same ('Same1.csv') or different ('Same2.csv') - if 'TRUE' this is the right choice of formulae. Much of other column headings in 'Same2.csv' can be traced to Eq.8. The result in 'Same1.csv' and 'Same2.csv' is found under 'P-Value'. In 'MoreLess.csv' the result is under 'P-More' and 'P-Less' addressing how often the second method is better and worse than the first, respectively. The two probabilities are complementary to each other. The second method is dominant if the result for 'P-More' is greater than for 'P-Less'; conversely, the first method is dominant; or else they are both equally dominant. 'M3' and 'S3' in this file are column titles for the mean and standard deviation, respectively, of the difference between the second ('M2' and 'S2') and the first ('M1' and 'S1') random variables in terms of Eq.12. 'M3' and 'S3' entries correspond to μ_2 and σ_2 in the text.

Other columns in above files perform conversions and run calculations for different scenarios because Eq.11 does not provide ready answers. For example 'Same2.csv' handles two x values: one with 'plus' and another with 'minus' in terms of Eq.7. This explains the headings 'XP' and 'XM'. It is convenient then to sort the values in the ascending order, so headings 'XF' and 'XS' appear. Then it is required to obtain z using Eq.11 which depends on means and standard deviations that apply, those under 'M1', 'S1' or 'M2', 'S2'. So, there are the headings 'Z1F', 'Z1S', 'Z2F', 'Z2S'. Similar notation is used in 'Same1.csv' and 'MoreLess.csv' files but only one x value there requires handling. A column may include additional results underneath the primary one to facilitate calculations that follow.

The table in the file 'Approximation.csv' verifies the expression in Eq.11 against what Excel has in its repertoire (presumably achieving its result by numeric integration).

REFERENCES

[1] Witten IH, Frank E, Hall M. Data mining: Practical machine learning tools and techniques (3rd ed.). 2011. Morgan Kaufmannn.

[2] Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection. Proceedings of the 14-th International Joint Conference on Artificial Intelligence (IJCAI-95) 1995; 1137-1143. Morgan Kaufmann.

[3] Molinaro AM, Simon R, Pfeiffer RM. Prediction error estimation: A comparison of resampling methods. Bioinformatics. 2005; 21(15): 3301-7. PMid:15905277. https://doi.org/10.1093/bioinformatics/bti499

[4] Vanwinckelen G, Blockeel H. Look before you leap: Some insights into learner evaluation with cross-validation. ECML/PKDD Workshop on Statistically Sound Data Mining 2014; JMLR: Workshop and Conference Proceedings 1:3-19.

[5] Fukunaga K. Introduction to statistical pattern recognition (2-nd edition). Academic Press. 1990. PMid:2169829. https://doi.org/10.1016/B978-0-08-047865-4.50007-7

[6] Fukunaga K, Hayes RR. Effects of sample size on classifier design. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1989; 11:873-85. https://doi.org/10.1109/34.31448

[7] Sahiner B, Chan HP, Hadjiiski L. Classifier performance estimation under the constraint of a finite sample size: Resampling schemes applied to neural network classifiers. Neural Networks. 2008; 21: 476-83. PMid:18234468. https://doi.org/10.1016/j.neunet.2007.12.012

[8] Leon-Garcia A. Probability, statistics, and random processes for electrical engineering (3rd ed.). 2008. Pearson Education.

[9] Benavoli A, Corani G, Demsar J, et al. Time for a change: A tutorial for comparing multiple classifiers through Bayesian analysis. Journal of Machine Learning Research. 2017; 18:1-36.

[10] Quinlan R. C4.5: Programs for machine learning. 1993. Morgan Kaufmann.

[11] Friedman JH, Kohavi R, Yun Y. Lazy decision trees. Proceedings of the 13-th National Conference on Artificial Intelligence. 1996; 717-24. AAAI.

[12] Waikato Environment of Knowledge Analysis (WEKA) "http://www.cs.waikato.ac.nz/ml/weka/". The University of Waikato, Hamilton, New Zealand.

[13] Holte RC. Very simple classification rules perform well on most commonly used datasets. Machine Learning. 1993; 11:63-90. https://doi.org/10.1023/A:1022631118932

[14] Liu H, Setiono R. Feature transformation and multivariate decision tree induction. In Arikawa S, Motoda H (editors): DS'98, LNAI. 1998; 1532:279-91. Springer-Verlag. https://doi.org/10.1007/3-540-49292-5_25

[15] Jelinek HF, Yatsko A, Stranieri A, et al. Diagnostic with incomplete nominal/discrete data. Artificial Intelligence Research 2015; 4(1):22-35. https://doi.org/10.5430/air.v4n1p22

[16] Assenza A, Valle M, Verleysen M. A comparative study of various probability density estimation methods for data analysis. International Journal of Computational Intelligence Systems. 2008; 1(2): 188-201. https://doi.org/10.1080/18756891.2008.9727616

[17] National health and nutrition examination surveys (NHANES). http://cdc.gov/nchs/nhanes/

[18] Yatsko A. Weighting features by the value displacement rebound. Artificial Intelligence Research. 2020; 9(1): 27-35. https://doi.org/10.5430/air.v9n1p27

[19] Efron B. The estimation of prediction error: covariance penalties and cross-validation. Journal of American Statistical Association. 2004; 99(467):619-32. https://doi.org/10.1198/016214504000000692

[20] Tasinaffo PM, Gonsalves GS, da-Cunha AM, et al. Using Monte Carlo method to estimate the behavior of neural training between balanced and unbalanced data in classification of patterns. Artificial Intelligence Research. 2018; 7(2): 1-25. https://doi.org/10.5430/air.v7n2p1

[21] Demsar J. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research. 2006; 7: 1-30.