Proceedings of Machine Learning Research 329:1–20, 2026 Conformal and Probabilistic Prediction with Applications

Distortion and Consistency in Conformal Prediction

Ilia Nouretdinov I.R.NOURETDINOV@RHUL.AC.UK
Centre for Reliable Machine Learning, Royal Holloway University of London, Egham Hill, Egham, Surrey, TW20 OEX, United Kingdom.

Editor: Ernst Ahlberg, Ulf Johansson, Henrik Boström, Alberto Carlevaro, Johan Hallberg Szabadváry and Lars Carlsson

Abstract

Conformal Prediction (CP) is a framework for reliable machine learning. Its aim is to convert a classification algorithm into a calibrated one with guaranteed validity properties. The core element (metaparameter) of a CP algorithm is a Non-Conformity Measure (CM) function that typically links the framework to an underlying algorithm.

Typically, NCM represents an information distance between a data example from an object space and a bag of data examples. In most cases, it has a residual form: the difference between the true value of an example and the label predicted by the underlying algorithm. However, there is also an alternative principle for the NCM construction. Its core is a *consistency* function that is a function of a bag of data examples only and measures, in some sense, regularity in a bag. Consistency functions can be transformed into NCM functions in a standard way.

Therefore, we suggest considering the consistency function as an intermediate chain between the underlying algorithm and the Conformity Measure. In the example of a simple nearest-neighbour approach, we demonstrate that it covers the capabilities of the standard method for defining conformity measures.

Our contributions are as follows. First, we compare the standard and consistency Conformity Measures and show that the consistency link plays a useful regulating role. In particular, it reduces the distortion effect, a loss in accuracy caused by the choice of NCM. Second, we compare several possible ways to define the consistency of a set with respect to an underlying algorithm. Third, we show new possibilities for the consistency language in constructing an NCM function. We demonstrate this in the area of ensemble learning: the consistency function for concrete underlying methods can be summarised, or integrated over a parameter.

Keywords: Conformal prediction, non-conformity measures, evaluation.

1. Introduction

The *Conformal Prediction (CP)* Vovk et al. (2005) framework in machine learning is a way of making individual hedged (confident) predictions that are *valid* under weak (i.i.d. or exchangeability) assumptions about the data distribution. It is a wide framework that can be connected to a standard underlying prediction algorithm via its core detail, the *Non-Conformity Measure* (NCM), thereby converting it into a reliable form.

The NCM function is a measure of the information similarity between an object (data instance) and a bag (multiset) of other data instances. It can also be interpreted as a measure of conformity (non-strangeness) of this object in the assumption that the bag is a sample of the data-generating distribution.

There is no essential limitation on the choice of the NCM function. An underlying machine learning algorithm, such as Nearest Neighbours, SVM, Neural Nets, Random Forests, and others, enables the extraction of the NCM. This measure can then be used to estimate the agreement between the real label of an example and the label predicted by an underlying algorithm. This is usually presented as numerical values, for example, the distances in Nearest Neighbours, or the Lagrangian α coefficients in SVM etc.

Another approach for constructing NCM was presented in Nouretdinov et al. (2011). The idea was to use a **consistency** that measures the overall "regularity" property of the data in a bag. The consistency of a data set was understood as a good separability between the data classes, and the conformity of an example is measured as a change in consistency after the elimination of this example.

The goal of the current work is to generalise this approach to cover machine learning algorithms. Practically, this would mean including an intermediate chain in the CP framework: *underlying method – consistency function – NCM*. So, the question is: does it open up new possibilities for constructing NCM functions?

Another motivating question concerns a **distortion** effect frequently observed in practice: a slight loss of accuracy when the CP framework is added, especially when the data bag size is relatively small. Can the consistent conformity measure decrease this effect?

The plan of the work is as follows. In Sect.2, we discuss in more detail the key motivating ideas mentioned above. In Sect.3, we pay special attention to the distortion challenge and give a proof that the distortion effect may appear even in very clear cases. In Sect.4, we define the core scheme of the suggested approach, how consistency can be defined, and included into the chain of the CP framework. In Sect.5, we start with an illustration of the distortion and recovery effects on artificial data. Then, using k-Nearest-Neighbours as the basic underlying algorithm and a real data set, we illustrate all the main effects: loss of accuracy and its recovery due to a new approach, several consistency functions compared, and elements of ensemble learning. Sect.6 concludes and summarises the work.

2. Background and Context

2.1. Conformal Prediction

Conformal Prediction (CP) framework was presented in the book Vovk et al. (2005). It is a hedged machine learning framework that allows one to provide a prediction with an individual measure of confidence. The core element of CP is the *Non-Conformity Measure (NCM)* defined as follows.

Definition 1 *Let Z be a measurable object space. A* Non-Conformity Measure (NCM) *A is a measurable function of the type $A(z, Z')$ where $z \in Z$ and Z' is a finite subset of Z. The output values of A should belong to a strictly ordered set.*

In this work, we will also use the notion of *consistency* function, which is similar to NCM, but without the first argument.

Definition 2 *Let Z be a measurable object space. A* consistency *function C is a measurable function of type $C(Z')$ where Z' is a finite subset of Z. The output values of C should belong to a strictly ordered set.*

Let $Z = X \times Y$ where X be an *object space* and Y be a *label space*. For example, X may be a finite-dimensional feature vector space and $Y = \{0, 1\}$. Let $x_1, \ldots, x_{n-1} \in X$ be training examples with known labels $y_1, \ldots, y_{n-1} \in Y$, and x_n be a new (or testing) example. Then a p-value is assigned to each hypothesis y about the new example's label y_n:

$$p(y) = \frac{|\{i = 1, \ldots, n : \alpha_i \geq \alpha_n\}|}{n}$$

where $\alpha_i = A\left(z_i, \{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n\}\right)$ for

$$z_1 = (x_1, y_1), \ldots, z_{n-1} = (x_{n-1}, y_{n-1}); z_n = (x, y).$$

Conformal prediction is also known in its *smoothed* version

$$p(y) = \frac{|\{i = 1, \ldots, n : \alpha_i > \alpha_n\}|}{n} + \theta_n \frac{|\{i = 1, \ldots, n : \alpha_i = \alpha_n\}|}{n}$$

where θ_n is a randomising random variable distributed uniformly in [0,1].

The CP output can be presented as a *prediction set* for a *significance level* ε,

$$R_\varepsilon = \{y : p(y) > \varepsilon\},$$

or as a *forced prediction*

$$\hat{y}_n = \arg\max_y p(y)$$

supplied with individual *confidence*

$$1 - \max_{y \neq \hat{y}_n} \{p(y)\}.$$

The *validity* properties of the CP guarantee under the i.i.d. assumption that the prediction set R_ε covers the true value of y_n with probability at least $1 - \varepsilon$.

In the case of classification (finite Y), a prediction is called *certain* at level ε if the size of the prediction set is at most 1. Individual confidence is the complement to 1 of the smallest significance level for which the prediction is certain.

To measure the performance of a different version of CP, different criteria are discussed and compared in Vovk et al. (2016). To compare a CP version with a machine learning method outside the CP framework, success can be measured by *accuracy*, that is, the proportion of correct predictions. By the effect of *distortion*, we mean that the accuracy of the forced conformal prediction is lower than the accuracy of the underlying method applied outside the CP framework.

2.2. NCM construction

NCM in its original setting can be obtained from an underlying machine learning algorithm such as SVM, Nearest Neigbours, etc. The most straightforward way to define the NCM score α_i on the basis of its prediction method is to measure the difference between the real label y_i and the label predicted for x_i after training on the other examples.

An alternative to the scheme considered above was used in Nouretdinov et al. (2011), where the CP framework was applied to high-dimensional data (MRI scan) with a small

number of training examples (38 patients). The distinction between the two classes (with depression and the control group) had an additional goal of detecting the *Region of Interest* (ROI) in the 3-dimensional voxel space. For each voxel, oxygenation level was measured, which yielded a feature vector. ROI was defined as the set of voxels for which the difference between the two classes was statistically significant. A larger ROI volume indicates better separability between classes.

In the CP approach, the NCM was not based directly on an underlying algorithm. Instead, an individual contribution of an example to the picture of the whole data set was measured. The ROI volume was interpreted as the level of *consistency* within the data set. The non-conformity of an example was measured by the gain of consistency if this example is excluded from the data.

Another motivating example is the well-known Support Vector Machine (SVM). Its principal approach is to separate two classes of data using a band as wide as possible. This allows interpreting the bandwidth as another form of 'consistency' related to the entire data. In a typical separable case, a support vector is a data point whose elimination increases the width of the band. Previously, when SVM was used within the conformal framework, the non-conformity score for an example was defined either as the λ coefficient from the dual optimisation problem or as the distance of the example from the separating hyperplane. Consistency may be an alternative to these approaches, combining the useful properties of both.

Our intention is to extend this scheme, moving from a very concrete setting to a more general understanding. There are several ways to define consistency for unlabelled data, such as compactness, compressibility, or other forms of regularity. As this work is focused on classification with two (or potentially more) classes, the natural understanding of consistency is *separability*, i.e., a clear difference between the classes. It has two faces: *statistically significant difference*, or *machine learning classification accuracy*. The first of them was illustrated in Nouretdinov et al. (2011), the second is the focus of the current work.

2.3. Ensemble learning within CP framework

The problems of combining different algorithms within the CP framework were studied. For the current state of the art, we can refer to the work Toccaceli and Gammerman (2017) for a review and recent developments. It appears that effective combining of even 2 or 3 components is a tricky question, and this is not easy to extend it to larger (e.g. parameterised) families. We wish to discuss the potential application of consistency to the ensemble learning problem, because the consistency link is another stage at which algorithms can be combined, as an alternative to combining NCM values or p-values, and it may be more amenable to combining parameterised families of underlying methods.

3. Distortion

3.1. Forced prediction and distortion challenge

As we mentioned above in the CP review, its output can take the form of a 'forced' point prediction (by selecting the label with the highest p-value) with individual confidence. As CP is based on the underlying method, its role can be seen as a kind of calibration: a definite

prediction without any measure of confidence is transformed into a possibly uncertain one while retaining validity properties.

In the majority of cases, the algorithm's underlying prediction matches the CP's point prediction. However, if one examines the exceptions where they do not match, in a typical case, the prediction without CP is more likely to be correct. Practical challenges related to forced prediction and its accuracy are also discussed in Paisios et al. (2019).

By *distortion* effect, we mean the loss of accuracy of the forced prediction of CP, compared to the prediction of the underlying method. This understanding of the term is closer to *distortion phenomenon* discovered in Gammerman et al. (2013) rather than to distortion in such works as Zecchin et al. (2026); Ganesan et al. (2025) where it is not related to forced prediction.

This distortion effect will be further illustrated in the experiments. The cause of this effect is the influence of a new example on the conformity scores of old examples. This effect becomes less pronounced as the number of examples in the training set grows, but remember that in Nouretdinov et al. (2011) it was relatively small, as was the case in some other practical applications with expensive data collection.

We try to detect two causes of the distortion effect: the *unavoidable* 'hard' distortion caused by the way the **CP framework** is constructed, regardless of which NCM is used; and the 'soft' distortion caused by the **choice of NCM**. In this section, we will study this effect in some extreme cases to demonstrate that it includes an unavoidable component, but can be decreased by the appropriate choice of NCM.

3.2. Influence of CP framework

3.2.1. Unavoidable distortion effects

Theorem 3 *Assume that the data sequence*

$$((x_1, y_1), \ldots, (x_n, y_n))$$

is stochastically generated by an exchangeable distribution on $(X \times Y)^n$*, where* X *is measurable, and* $|Y| = k$*; a Conformal Predictor is trained on the set* $(x_1, y_1), \ldots, (x_{n-1}, y_{n-1})$ *applied to the new (testing) example* x_n*, and assigns p-values* p_0 *and* p_1 *to the hypotheses* $y_n = 0$ *and* $y_n = 1$*, respectively; the prediction is made by the largest p-value as*

$$\hat{y}_n = \arg\max_y p_y$$

and the tie cases are resolved by a fair random choice.

Let us assume the diversity constraint that the alpha values $\alpha_1, \ldots, \alpha_n$ *are all different with probability 1.*[1] *Then the probability of making a mistake is at least* $\frac{k-1}{kn}$*.*

Proof:

1. This poses restrictions on both the generating distribution and the choice of NCM. First, the distribution should be continuous so that the same example cannot appear twice with a positive probability. Second, all the alpha values produced by NCM for a concrete data sequence must be different. If the first requirement is satisfied, the second can be forced by resolving ties using an auxiliary function mapping X to a strictly ordered set.

In (non-smoothed) CP, the p-value (even for the wrong hypothesis) is always at least $\frac{1}{n}$. Event A "the p-value assigned to the true hypothesis is $\frac{1}{n}$" has the probability of $\frac{1}{n}$. Consequently, event B "the p-value assigned to one of the wrong hypotheses beats the p-value assigned to the true hypothesis" has probability at least $\frac{k-1}{kn}$. Indeed, in the best case, when all the p-values assigned to the wrong hypotheses are always at the lowest level of $\frac{1}{n}$, the true hypothesis wins only in 1 of k cases within event A, so the probability of the mistake (event B) is exactly $\frac{k-1}{kn}$, and increases in any less favorable case.

3.2.2. Role of the diversity constraint

Assume that NCM works in an 'ideal' way: $\alpha_i = 0$ if the classification (x_i, y_i) is correct and $\alpha_i = 1$ if the classification is wrong. This is achievable, e.g., if $y_i = f(x_i)$ for a deterministic function f. In this case, the p-value will be 1 for the true hypothesis and $\frac{1}{n}$ for a wrong hypothesis, so there will be no errors by the CP, and no distortion effect.

So, this is actually achieved by making the p-value too conservative in the case of studying the correct hypothesis. This is possible in the non-smoothed version of CP.

3.2.3. Smoothed settings

In order to analyse the distortion in the smoothed setting (as described in Sect. 2.1), we need to resolve a potential ambiguity. Is it assumed that θ_n is the same for all the hypotheses, or is it randomly re-generated for each of them?

Literally, the first understanding is true: it is assumed that the sequence $\theta_1, \dots, \theta_n, \dots$ is generated as another i.i.d. sequence that complements the data stream. So, this would not help much in resolving the ties: where the non-smoothed p-values for the true hypothesis and for one (or more) wrong hypotheses are meeting each other at level $\frac{1}{n}$, there they will all be just replaced with $\frac{\theta_n}{n}$ and the tie still has to be resolved by another randomiser.

The second understanding is more interesting. First, it naturally allows us to set the tie probability to 0: the p-values for the different hypotheses will differ almost surely. Second, *it makes the diversity constraint unnecessary* to observe the distortion effect. Indeed, due to the validity properties of CP, the p-value of the true hypothesis is uniformly distributed in $[0, 1]$. So, the probability that CP makes a mistake becomes $\frac{k-1}{kn}$ as in Theorem 3. Indeed, the p-value for the true hypothesis is uniformly distributed in $[0, 1]$. The p-value for any concrete wrong hypothesis is uniformly distributed in an interval of the form $[\frac{m_0}{n}, \frac{m_1}{n}]$ for some integer $0 \le m_0 < m_1 < n$. If $m_0 = 0$ and $m_1 = 1$, this gives the wrong hypothesis the smallest chance to win over the true hypothesis. However, even in this case, the true hypothesis p-value falls into $[0, \frac{1}{n}]$ with probability $\frac{1}{n}$, and competes with $k-1$ other values of the same kind.

3.3. Influence of NCM construction

3.3.1. Two sources of distortion

As we have seen above, the main source of the distortion is a high p-value assigned to the wrong hypothesis. In the non-smoothed setting, it is limited from below by $\frac{1}{n}$. On the other hand, the distribution of the p-value assigned to the true hypothesis is not improvable because it has a fixed distribution, unless alpha values are allowed to be equal. Therefore,

we can switch our focus to observing **the p-value assigned to the wrong hypothesis**, as the core of the distortion challenge. We may consider $\frac{1}{n}$ as an ideal baseline and try to approach it.

As we will see below, another source of distortion lies in the way the underlying methods are converted into non-conformity scores. This can cause even higher p-values than $\frac{1}{n}$; however, this kind of distortion may be recovered by modifying the way of NCM calculation.

3.3.2. Model task: catching one exception

Consider the following setting, related to the task of assessing the p-value assigned to a wrong hypothesis. Assume that $x_1, \ldots, x_n$ is a random sample generated with a continuous distribution (such as normal or uniform on [0,1] – we need it for diversity), with independently generated labels $y_1, \ldots, y_n$ where each y_i is 1 with probability $1 - \delta$ and 0 otherwise. Let us assume that $\delta \ll \frac{1}{n}$: we model the case where the alternative label is possible but abnormal.

In other words, we assume always surely that either all the labels are 1 (if we consider the true hypothesis), or (if we consider its wrong alternative) only one label is different from the others:

$$y_1 = \cdots = y_{n-1} = 1; y_n = 0;$$

and the probability of the rest is negligibly small. In the ideal case, the wrong hypothesis is rejected at the level $\frac{1}{n}$, so that $p(0) \leq \frac{1}{n}$. In standard (non-smoothed) CP, this is equivalent to: n-th example is the strangest in a strict sense ($\alpha_n > \alpha_i$ for any $i < n$). In smoothed CP, this is still the most desirable condition, although the hypothesis has a chance to be rejected in borderline cases ($\alpha_n \geq \alpha_i$ with equality for some $i < n$). So, can it happen that α_n is not the highest if we consider different NCMs?

3.3.3. Examples of NCM

Let us compare several NCMs based on the 1NN method. In the following, we use d_i^+ as the distance between x_i and its nearest neighbour of the same class, and d_i^- for the neighbour of the other-class and $d_i = \min\{d_i^+, d_i^-\}$ as the distance from any neighbour. If x_j is the nearest neighbour of x_i, we call x_i a **co-neighbour** of x_j.

- **Str** (straightforward): $\alpha_i = 0$ if the prediction of 1NN is correct, i.e. the nearest neighbour of x_i has the same label (equivalently, $d_i^+ < d_i^-$), $\alpha_i = 1$ otherwise. This is the most literal approach: the strangeness of an example is the mismatch between its predicted and real label.
- **Simp** (simple): $\alpha_i = \frac{\pm 1}{d_i}$, which is negative if the 1-NN prediction is correct ($d_i^+ < d_i^-$).
- **CA** (consistency based on accuracy): α_i is the leave-one-out accuracy of the 1-NN method in the rest of the examples. We take it as a simple example of *consistency* approach to NCM construction.

If the distribution is continuous, then all x_i are distinct with probability 1. In cases where the diversity assumption is formally incorrect, it should be enforced by using an a priori strangeness order on unlabeled objects x that is used to resolve strangeness bonds.

3.3.4. Comparison on the model task

With **Str** NCM, the example is strange if and only if it is misclassified with the 1-NN method after training on the rest of the examples. This means that the last example x_n (if assigned the wrong label) is the strangest one, but not strictly: it shares this property with any of its co-neighbours. On average, there will be exactly 1 such example; therefore, the probability of its existence will always be positive.

With **Simp** NCM, the probability of misclassifying the last label may be smaller than with Straightforward NCM (due to a rare but possible **imparity effect** where the nearest neighbour of the new example is *not* one of its co-neighbours). However, the distortion probability is still positive. Indeed, for any concrete configuration of n points, there is a pair with the smallest distance. If the last label is one of them, it is not strictly the strangest one, but shares the highest α_i with the second element of this pair. Within an exchangeable mode, the fall of x_n in this pair has a chance of $\frac{2}{n}$. If strangeness ties are resolved in the way mentioned at the end of Sec. 3.3.3, the p-value will have equal chances of being $\frac{1}{n}$ and $\frac{2}{n}$; so it will strictly exceed $\frac{1}{n}$ with probability $\frac{1}{n}$.

This effect disappears with **CA** NCM, where the abnormally labelled example becomes strictly the strangest. Its elimination will make the remaining data sample 'pure' (with 1NN accuracy 100%), unlike the elimination of any other example. So, the wrong hypothesis p-value is $\frac{1}{n}$ exactly.

Thus, in this simple stochastic example, we observe both the distortion effect and the potential usage of the consistency for the **recovery effect**. In the remainder of the work, we will explore and study it in more detail.

3.3.5. Analysis of the recovery effect

In the example above, the **CA** NCM had shown a smaller distortion than NCMs directly based on the underlying 1NN method. The special property of **CA** NCM was its independence from one of two arguments.

$$\alpha_i = A\left(z_i, \{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n\}\right) = C\{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n\}$$

was actually a function of its second argument only! We will say that C is a **consistency function**, and this NCM has the property of **elimination consistency**.

What played a major role in the recovery effect is that the consistency of a set increased drastically once all its elements (pairs (x_i, y_i) for $i = 1, \ldots, n$) follow the property $y_i = 1$, while injecting one exception ($y_n = 0$) made the set relatively inconsistent.

For consistent NCM, injection of the exception $y_n = 0$ did not change the strangeness of the n-th example, but it decreased the background strangeness of any other (i-th for $i < n$) example, because

$$C\{(x_1, 1), \ldots, (x_{i-1}, 1), (x_{i+1}, 1), \ldots, (x_{n-1}, 1), (x_n, 1)\}$$

$$> C\{(x_1, 1), \ldots, (x_{i-1}, 1), (x_{i+1}, 1), \ldots, (x_{n-1}, 1), (x_n, 0)\}$$

The latter property does not follow from the definition of the consistency function. But it is typically true if consistency is defined on the basis of a machine learning method in relation to how y depends on x. From a general machine learning perspective, having uniformly

labelled data (all $y_i = 1$) is an ideal case of pure consistency, and this is not specific to the underlying method of 1NN.

What happens if an exception is injected into non-uniformly labelled data? Assume that the data already includes exactly one example with $y_j = 0$ for some $j < n$. We cannot say that having 2 unusual labels is always less consistent than having 1. In certain cases, the opposite can even hold for the 1NN method. The labelling, with one exception, is not purely consistent, but with two exceptions, it may be. Indeed, one exception will always be misclassified in leave-one-out mode. But two exceptions may be close enough to each other and far enough from all the rest, and the accuracy is perfect in this case. If the data sequence is generated as described in the model task above, this may occur only occasionally. But if the data-generating mechanism is another, this is not necessarily a mistake.

4. Consistency

In order to conduct further studies, we define two steps in a more general form: from an underlying method to consistency and from consistency to NCM.

4.1. From an underlying method to a consistency function

Although the notion of consistency may be wide enough, in this work, we focus on its linking to an underlying prediction algorithm, as needed in the context of NCM construction. Let us discuss how the consistency function is defined for a set of labelled data examples, with respect to an underlying classification method. A straightforward may be just to take the accuracy (proportion of correct predictions) of the underlying method on the data set. This approach is universally applicable and simple enough to start with, although not a unique one.

Let us continue with the example of the kNN method for $k = 1$. Assume that there is a bag Z of labelled examples $z_1 = (x_1, y_1), \ldots, z_n = (x_n, y_n)$ where x_i belong to the object space X, and y_i belongs to the label space Y, and the task is to define the **consistency function** $C(Z)$.

Let D be a distance function (metric) on X. As before, for an example (x_i, y_i), we denote the distance from the nearest neighbour of the same class as

$$d_i^+ = \min_{j:j\neq i, y_j = y_i} D(x_i, x_j)$$

and the distance to the nearest neighbour of the other class as

$$d_i^- = \min_{j:j\neq i, y_j \neq y_i} D(x_i, x_j).$$

We may consider various versions of the consistency function.

- **CA** (accuracy): Proportion of i s.t. $d_i^- / d_i^+ > 1$.
- **CS** (sum): Sum of $\left(d_i^- / d_i^+\right)^2$ by i.
- **CS2** (sum, version 2): Sum of d_i^- / d_i^+.

- **CSL** (sum in logarithmic scale): Sum of $\log(d_i^-/d_i^+)$.
- **CT** (the other class only): Sum of just $\left(d_i^-\right)^2$.
- **CT2** (the other class, version 2): Sum of d_i^-.
- **CTL** (the other class in logarithmic scale): Sum of $\log\left(d_i^-\right)$.

The version CA is the 'basic' one, it is equal to leave-one-out cross-validation accuracy of the one-nearest-neighbour approach. Versions CS, CS2, CSL try to extract more numeric information from the nearest neighbour measurements. This poses the question of how they should be combined; this is why we take three different formulas to compare. Versions CT, CT2, CTL are simplified from CS, CS2, CSL by eliminating the same class distance and using only the other class distance. Although such an approach ignores some available information, it might be more stable against occasional repetitions of identical or similar items during the data collection. All these formulas can be easily extended to the case $k > 1$ by modifying d_i^+ and d_i^- by replacing the minimum distance with the average of k smallest distances.

We will also include some initial results of ensemble learning. When we say that k is in the range 1-10, this means that the value consistency function is calculated for each k, and then averaged.

4.2. From consistency to NCM

In Nouretdinov et al. (2011)) the connection between NCM A and consistency C can be understood in the following way:

$$A\left(z, \{z_1, \ldots, z_{n-1}\}\right) = C\left(\{z_1, \ldots, z_{n-1}\}\right) - C\left(\{z_1, \ldots, z_{n-1}, z\}\right)$$

where the last term can be dropped because, at the p-value calculation stage, it becomes a constant. In words, it can be formulated: the strangeness of an example is the gain in consistency after its elimination. This is called **elimination consistency**.

The alternative to elimination consistency is **replacement** or **conjugation consistency**. As far as we focus on the binary classification case $Y = \{0, 1\}$, let us use the following modification for real data sets:

$$A\left(z, \{z_1, \ldots, z_n\}\right) = C\left(\{z_1, \ldots, z_n, \overline{z}\}\right)$$

where $\overline{z} = \overline{(x, y)} = (x, 1 - y)$. In other words, the strangeness score of an example within a set of examples is measured as the consistency gain after changing the label of this example to the opposite.

Although elimination consistency is more widely applicable, replacement consistency is more convenient for initial studies due to its fixed set size. This allows us to avoid a 'skewed' comparison between NCM and the consistency defined for different set sizes.

5. Experiments

In this work, we focus on the essence of distortion and recovery effects, keeping the models as simple as possible to demonstrate them. Therefore, in addition to using replacement consistency, we prefer a simple (nearest-neighbour) underlying method, leaving the alternative (Support Vector Machine) to the Appendix. Furthermore, we leave the comparison and optimisation of computational complexity out of the scope, working with the consistency approach in its pure (not an approximate) form.

5.1. Artificial data

5.1.1. Data generation

As a toy data example, we generate N examples with one-dimensional objects x and binary labels y. They are generated as follows: y_i is a fair Bernoulli sequence (equal chances of 0 and 1), x_i is distributed as $N(dy_i, 1)$ where d is a parameter.

5.1.2. Settings

For quick comparison, we selected only the following settings.

- **NoCP**: just leave-one-out accuracy of 1NN on the data set.
- **Simp**: as defined in Sect. 3.3.3.
- **Stand**: standard NCM $\frac{d_i^+}{d_i^-}$. This measure is included due to its wide usage, although this is not a pure analog of the *one* nearest neighbour method: in some sense, it considers two neighbours to quantify the strangeness. However, it is interesting to check this shift as an alternative to consistency.
- **CA**: as defined in Sect. 4.1
- **CTL**: as defined in Sect. 4.1

In the last two settings, we mean the elimination consistency from Sect. 4.2.

5.1.3. Results and observations

The results are presented in Tab. 1. Each experiment is repeated 100 times. The data is processed in leave-one-out mode.

The following observation mostly refers to the middle level ($d = 3$), which is the most interesting case due to realistic accuracy. In two other cases, they confirm the same tendencies, but in a weaker and more noisy way.

- The distortion effect is observed (and significant[2]) in both the Simp and Stand settings.

2. Here, we mean sign test significance. For reference: $binom.test(61, 100)\$p.value = 0.0352$ and $binom.test(65, 100)\$p.value = 0.0035$. So, the percentage below 35 is a significant decrease and the percentage above 65 is a significant increase.

- In Stand, the distortion effect is smaller than in Simp, but more information is used to achieve this (two numerical distance values instead of one).
- CA is significantly good for recovery against Simp but weak against Stand. However, the comparison A vs. Simp is fair, as they use the same information.
- CTL is a way to over-perform both Simp and Stand. The improvement is stable in all settings and significant for $N = 50$.
- For $N = 50$, CTL shows a small improvement even against the NoCP baseline!

$d=1$	NoCP	Simp	Stand	CA	CTL	1	NoCP	Simp	Stand	CA	CTL	1	NoCP	Simp	Stand	CA	CTL
N=10	57.0	55.1	56.6	56.3	57.4	20	57.7	56.6	57.7	57.5	58.8	50	60.1	59.2	60.0	60.0	60.9
>No		33.5	48.5	39.5	57.5			30.0	50.0	38.0	57.5			16.0	49.0	39.0	60.0
>Simp			55.5	63.0	62.0				64.5	70.0	64.5				66.0	71.0	69.0
>Stand				42.5	58.0					47.0	59.0					45.0	62.0
>A					60.5						63.0						62.0
$d=3$	NoCP	Simp	Stand	CA	CTL	3	NoCP	Simp	Stand	CA	CTL	3	NoCP	Simp	Stand	CA	CTL
N=10	89.4	84.0	86.1	86.3	87.1	20	89.3	87.0	88.5	88.2	89.4	50	90.7	89.5	90.5	90.4	91.3
>No		10.0	27.0	10.0	36.0			11.5	35.0	11.0	45.5			01.0	40.5	24.0	60.5
>Simp			61.5	72.5	67.0				65.0	72.0	67.0				74.5	87.0	82.5
>Stand				50.5	59.0					45.0	59.0					51.5	66.0
>A					59.5						62.5						69.5
$d=5$	NoCP	Simp	Stand	CA	CTL	5	NoCP	Simp	Stand	CA	CTL	5	NoCP	Simp	Stand	CA	CTL
N=10	99.1	92.9	93.8	94.2	93.8	20	99.1	96.5	97.0	96.8	97.1	50	99.2	98.2	98.6	98.3	98.7
>No		04.0	06.5	00.0	07.0			06.5	09.0	00.0	11.0			04.0	14.5	01.0	19.5
>Simp			58.5	62.0	58.5				59.0	52.5	58.0				62.0	56.0	63.0
>Stand				47.5	50.0					43.0	50.0					39.0	54.5
>A					47.5						57.0						63.0

Table 1: Results on the artificial data set

d	N	NoCP	Simp	Stand	CA	CTL
1	10	57.00	57.00	56.70	57.00	57.00
1	20	57.75	57.75	57.60	57.75	59.10
1	50	60.12	60.12	60.06	60.12	61.04
1	100	60.66	60.66	60.66	60.66	61.67
1	200	59.345	59.345	59.37	59.345	60.295
3	10	89.40	89.40	88.40	89.40	88.50
3	20	89.35	89.35	89.25	89.35	89.80
3	50	90.72	90.72	90.78	90.72	91.28
3	100	90.25	90.25	90.22	90.25	91.07
3	200	89.90	89.90	89.925	89.90	90.665
5	10	99.10	99.10	98.30	99.10	97.60
5	20	99.15	99.15	99.10	99.15	98.95
5	50	99.26	99.26	99.36	99.26	99.34
5	100	99.24	99.24	99.27	99.24	99.31
5	200	99.035	99.035	99.04	99.035	99.155

Table 2: What happens if the ties are resolved with the underlying method

5.1.4. Tie breaking alternative

Table 2 models an alternative way of dealing with ties: when the p-value assigned to different hypotheses are equal, the final solution is made just by the underlying method as is. This is done to observe whether the distortion effect is mostly produced through such ties or by strictly wrong solutions. According to this series of results, the answer is the first. In practice, p-values, in some sense, decrease the 'resolution' of the prediction, and this is the main mechanism of distortion.

However, a surprising effect is that the recovery in its CTL version still gives a positive impact on accuracy in this setting! In most cases, it produces a higher accuracy than the original setting. This observation is in favour of the consistency framework.

5.2. Real data and ensembling

5.2.1. Data Sets

We apply the algorithms to the following real data examples from the UCI repository Irvine, with a binary label classification task.

1. Raisin Data Set Irvine (2019b) presented in Cınar et al. (2020). The examples of the data set are 900 raisin grains of two equally presented classes, Kecimen and Besni, 450 items from each of them. Each feature vector has 7 numerical dimensions (area, perimeter, major axis length, minor axis length, eccentricity, convex area, extent).

2. Wisconsin Diagnostic Breast Cancer Data Irvine (2019c). The examples are from two classes: Benign and Malignant, feature vectors have 30 numerical dimensions.

3. Haberman's Survival Data Set Irvine (2019a). Two classes: positive/negative survival in 5 years, three dimensions (age, year of operation, number of nodes).

All dimensions were standalised (scaled so that the standard deviation is 1) before applying the Euclidean distance to the vectors.

5.2.2. Experimental settings

Table 3 includes the comparison results. The prior goal of this section was to compare different ways of defining the consistency function with each other and to demonstrate the possibility of ensembling. Therefore, we use a slightly different set of NCM to compare, including all the variations from Sect. 4.2.

We follow the methodology described in Sect. 4.2, using the kNN underlying algorithm with variable k, Euclidean distance, and replacement consistency. In addition, we include a line on ensemble learning where the consistency function is mixed over the values $k = 1, \ldots, 10$ of the kNN parameter (number of neighbours).

The training and test sets are randomly selected from the data set. We set the testing set size to 100. As for the number of training set examples, we try 100 and also 20, because the distortion effect is stronger when the size is small. The results in each case are averaged over 100 such random splits.

The column names refer to: (NoCP) – underlying method itself; (Stand) – within the CP framework but without consistency; (A,S,...,TL) – versions of the consistency function.

5.2.3. Observations

The observed effects are as follows.

- Some loss of accuracy (distortion) is stable for data sets 1,2 and partially observable for data set 3 when the CP based on standard NCM (Stand) is compared to the underlying algorithm itself (NoCP).
- Whenever $k = 1$, the effect of recovery is strong. Any formula (CA,CS,CS2,CSL) performs better than (Stand). The only exception is (CA) for Data Set 3, but there was no distortion to correct.
- Comparing (CA, CS, CS2, CSL) for the large training size (100): the best results are for $k = 5$ (CS2) (Data Set 1), $k = 1$ (CS2) (Data Set 2), and $k = 10$ (CS2) (Data Set 3); therefore, all are achieved with (CS2). For Data Sets 1 and 2, they are even better than the best (NoCP) results. For Data Set 3, the best (NoCP), (Stand), and (CS2) results are close to each other.
- For the small training size (20), we have the best results for $k = 1$ (CS) (Data Set 1), $k = 1$ (CA) (Data Set 2), and $k = 5$ (CS) (Data Set 3). Again, they perform better than (NoCP), except for Data Set 3, but the relative preference is not stable.
- Observing the group of columns (CT,CT2,CTL), we may see further improvement of accuracy as far as the best achievements are compared. However, in this group, the consistency formulas are less directly related to the original method.
- Observing the last line, we can compare the aggregated results (the consistency function is averaged over the parameter), which are quite reasonable, especially in Data

train	test	k	NoCP	Stand	CA	CS	CS2	CSL	CT	CT2	CTL
20	100	1	76.42	75.54	76.57	79.21	78.68	78.04	79.66	79.45	79.85
20	100	5	78.99	78.18	79.00	75.16	76.85	79.01	75.69	75.78	75.44
100	100	1	79.99	79.92	80.00	83.33	82.64	81.38	84.38	83.75	82.86
100	100	5	85.01	84.93	84.77	85.73	85.78	85.63	86.04	86.07	85.64
100	100	10	85.09	84.83	84.64	84.88	85.16	86.65	85.51	85.33	84.96
ensemble											
100	100	1-10	–	–	85.10	85.58	85.48	85.54	85.83	85.88	85.47

Table 3: Accuracy comparison chart: kNN on Raisins.

train	test	k	NoCP	Stand	CA	CS	CS2	CSL	CT	CT2	CTL
20	100	1	90.09	88.52	90.52	89.43	89.74	90.13	89.98	90.20	90.06
20	100	5	86.63	83.64	84.72	78.43	81.45	81.44	78.81	78.73	18.97
100	100	1	93.91	93.73	94.04	94.91	95.13	95.11	95.00	95.15	95.08
100	100	5	94.64	94.33	94.23	93.39	93.74	94.07	93.97	93.84	93.65
100	100	10	93.94	93.69	92.94	90.96	91.65	92.62	92.06	91.69	91.36
ensemble											
100	100	1-10	–	–	94.72	93.42	93.73	94.30	93.52	93.47	93.72

Table 4: Accuracy comparison chart: kNN on WDBC.

train	test	k	NoCP	Stand	CA	CS	CS2	CSL	CT	CT2	CTL
20	100	1	65.25	65.74	65.37	68.81	68.35	66.56	68.96	68.73	68.27
20	100	5	72.82	72.75	72.37	73.18	73.14	62.65	73.30	73.39	73.62
100	100	1	66.62	66.78	66.92	73.01	72.95	69.38	69.38	68.92	70.81
100	100	5	73.43	73.35	72.38	74.21	73.95	73.38	74.01	74.12	74.10
100	100	10	74.69	74.57	74.18	74.46	74.56	74.45	74.52	74.37	74.42
ensemble											
100	100	1-10	–	–	73.38	73.82	73.86	72.65	74.46	74.40	73.66

Table 5: Accuracy comparison chart: kNN on Haberman's Survival.

Set 1: for (CA), they are even better than the results for concrete values of k; in the other columns, they are slightly worse but close to the best of them. About the same can be said about Data Set 2: the aggregated results perform better than three for (CA), and better than two of three for most of the others. The results for Data Set 3 are less interesting, as the difference between $k = 5$ and $k = 10$ is small.

6. Conclusion and Future Work

The main achievement of this work is the generalisation of experience from Nouretdinov et al. (2011), showing that the scheme is applicable in a general context and offers some advantages and applications. A typical approach to defining conformity scores can be translated into the language of consistency, that is, as a function of a simple type (of the only argument, a finite set) rather than the conformity measure itself.

As a limitation of the suggested approach, we note that in extremely noisy cases, consistency may even degrade accuracy. For example, if the labels are assigned randomly. If we mechanically average the results over all possible combinations of labels, the accuracy will be about 1/2, whether or not consistency is used; therefore, there are cases where it is worse with consistency. The suggested approach makes sense for relatively clean data and is motivated by the distortion that is observed in such cases.

Comparing the results of Nouretdinov et al. (2011) and the current study, we can say that they show complementary approaches to the engineering of the consistency function: from a statistical analysis of the feature space, and from a pre-selected underlying algorithm, or their family in the case of ensembling. An interesting future goal may be its further reduction, e.g., by adding an 'interaction' step where the core function is a function of a two-element set rather than any set.

Other directions for future work may include: the development of more computationally efficient ('inductive') algorithms; the study of other underlying methods; and a deeper study of ensembling.

Acknowledgments

The author is grateful to Alexander Gammerman, Volodymir Vovk, Alexander Balinsky for useful discussion.

References

Ilkay Cınar, Murat Koklu, and Sakir Tasdemir. Classification of raisin grains using machine vision and artificial intelligence methods. 2020. URL https://api.semanticscholar.org/CorpusID:248342352.

Alex Gammerman, Volodya Vovk, and Vladimir Vapnik. Learning by transduction, 2013. URL https://arxiv.org/abs/1301.7375.

Unnikrishnan Kunnath Ganesan, Giuseppe Durisi, Matteo Zecchin, Petar Popovski, and Osvaldo Simeone. Online conformal compression for zero-delay communication with distortion guarantees. In *2025 IEEE International Symposium on Information Theory (ISIT)*, pages 1–6, 2025. doi: 10.1109/ISIT63088.2025.11195225.

UC Irvine. Machine learning repository. URL https://archive.ics.uci.edu/ml/index.php.

UC Irvine. Haberman's survival data set, 2019a. URL https://archive.ics.uci.edu/dataset/43/haberman+s+survival.

UC Irvine. Raisin data set, 2019b. URL https://archive.ics.uci.edu/dataset/850/raisin.

UC Irvine. Wisconsin diagnostic breast cancer data, 2019c. URL https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic.

Ilia Nouretdinov, Sergi G. Costafreda, Alexander Gammerman, Alexey Chervonenkis, Vladimir Vovk, Vladimir Vapnik, and Cynthia H.Y. Fu. Machine learning classification with confidence: Application of transductive conformal predictors to mri-based diagnostic and prognostic markers in depression. *NeuroImage*, 56(2):809–813, 2011. ISSN 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2010.05.023. URL https://www.sciencedirect.com/science/article/pii/S105381191000755X. Multivariate Decoding and Brain Reading.

Andreas Paisios, Ladislav Lenc, Jiří Martínek, Pavel Král, and Harris Papadopoulos. A deep neural network conformal predictor for multi-label text classification. In Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, and Evgueni Smirnov, editors, *Proceedings of the Eighth Symposium on Conformal and Probabilistic Prediction and Applications*, volume 105 of *Proceedings of Machine Learning Research*, pages 228–245. PMLR, 09–11 Sep 2019. URL https://proceedings.mlr.press/v105/paisios19a.html.

Paolo Toccaceli and Alexander Gammerman. Combination of conformal predictors for classification. In Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, and Harris Papadopoulos, editors, *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*, volume 60 of *Proceedings of Machine Learning Research*, pages 39–61. PMLR, 13–16 Jun 2017. URL https://proceedings.mlr.press/v60/toccaceli17a.html.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer Science & Business Media, 2005.

Vladimir Vovk, Valentina Fedorova, Ilia Nouretdinov, and Alexander Gammerman. Criteria of efficiency for conformal prediction. In *Proceedings of the 5th International Symposium on Conformal and Probabilistic Prediction with Applications - Volume 9653*, COPA 2016, page 23–39, Berlin, Heidelberg, 2016. Springer-Verlag. ISBN 9783319333946. doi: 10.1007/978-3-319-33395-3_2. URL https://doi.org/10.1007/978-3-319-33395-3_2.

Matteo Zecchin, Unnikrishnan Kunnath Ganesan, Giuseppe Durisi, Petar Popovski, and Osvaldo Simeone. Prediction-powered communication with distortion guarantees. *IEEE Journal on Selected Areas in Information Theory*, 7:33–45, 2026. doi: 10.1109/JSAIT.2026.3668947.

Data Set	train	test	degree	C	SVM CP	Cons-SVM CP
Raisins	20	100	1	1000	74.91	74.93
Raisins	20	100	1	1	77.25	**78.41**
Raisins	20	100	5	1000	68.66	69.61
Raisins	20	100	5	1	67.79	69.73
Raisins	100	100	1	1000	82.60	84.84
Raisins	100	100	1	1	82.90	**85.00**
Raisins	100	100	5	1000	74.40	73.49
Raisins	100	100	5	1	75.78	71.54
WDBC	20	100	1	1000	83.85	82.21
WDBC	20	100	1	1	**85.27**	84.06
WDBC	20	100	5	1000	76.52	70.75
WDBC	20	100	5	1	76.19	75.83
WDBC	100	100	1	1000	91.73	88.49
WDBC	100	100	1	1	**95.06**	93.63
WDBC	100	100	5	1000	88.46	88.83
WDBC	100	100	5	1	88.45	89.65
Haberman's	20	100	1	1000	67.60	67.42
Haberman's	20	100	1	1	**59.73**	68.93
Haberman's	20	100	5	1000	62.86	61.00
Haberman's	20	100	5	1	63.37	62.31
Haberman's	100	100	1	1000	71.45	**74.18**
Haberman's	100	100	1	1	73.43	73.97
Haberman's	100	100	5	1000	64.06	59.31
Haberman's	100	100	5	1	66.84	64.07

Table 6: SVM accuracy: comparison chart.

Data Set	train	test	degree	C	Cons-SVM CP
Raisins	20	100	1	ensemble	**68.53**
Raisins	20	100	5	ensemble	**70.12**
Raisins	100	100	1	ensemble	**85.66**
Raisins	100	100	5	ensemble	**73.88**
WDBC	20	100	1	ensemble	**85.66**
WDBC	20	100	5	ensemble	**76.26**
WDBC	100	100	1	ensemble	**94.14**
WDBC	100	100	5	ensemble	89.50
Haberman's	20	100	1	ensemble	68.80
Haberman's	20	100	5	ensemble	62.06
Haberman's	100	100	1	ensemble	73.99
Haberman's	100	100	5	ensemble	**66.13**

Table 7: SVM accuracy, with ensemble (0.1,1,10,100,1000) for C.

Appendix A: results with SVM

Table 6 shows the results with the Support Vector Machine (SVM) as the underlying algorithm, on the same data sets. In its standard form, the non-conformity scores are Lagrange multipliers, which are 0 for non-support vectors and positive for support vectors, especially for misclassified examples lying within the margin or on the wrong side. The sign of the coefficients is changed by the conformity sense.

As a consistency measure, we use the **number of non-support vectors**. Note that the misclassified examples are also support vectors, so this is very close to measuring the accuracy. In the standard scheme of non-separable SVM, the penalty for these examples is proportional to their distance to the correct side of the margin because this task is easier for optimisation. For our purposes, we can simplify it to only the overall number. The degree (of the polynomial kernel) and C are standard SVM parameters. We leave some of them as choices to compare the best results achieved.

For a training set size of 100, the results show an advantage for the consistency approach in two databases, but a disadvantage for WDBC, where the best accuracy is achieved without consistency.

However, in all cases, the difference is slight. This happens because the idea of SVM itself is already consistent. The optimised function in the simplest SVM is the width of the separating margin, which is a consistency measure across the entire data set.

However, to fully convert it to a consistent language, we need to cover the non-separable case. This is why we switched it to the number of non-support vectors.

We also tested the ensemble approach for merging methods with different values of the parameter C. The results are presented in Tab. 7, and the majority show advantages over $C = 1000$ or $C = 1$; in the remaining cases, the accuracy is close to the best, though slightly worse.