Pattern Recognition 172 (2026) 112515

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Conformal Compressors

Nery Riquelme-Granada [a,*], Khuong An Nguyen [b], Zhiyuan Luo [b]

[a] University of Southampton, Southampton, SO17 1BJ, UK
[b] Royal Holloway University of London, Surrey, TW20 0EX, UK

ARTICLE INFO

Keywords:
Data compression
Coresets
Conformal prediction

ABSTRACT

Standard machine learning predictors become more effective as larger input datasets are made available. While this is desirable for enhancing predictive power, it often implies substantial computational costs. One feasible approach to mitigate this issue is to replace large datasets with smaller, carefully crafted representations that retain the essential properties of the original data. In this paper, we revisit the exploration of this approach through the interaction of coresets - small, provably correct summaries of data - and Conformal Prediction, a robust and general method for calibrating machine learning predictions. Specifically, we build on existing work to introduce Conformal Compressors, a method inspired by coresets that leverages Conformal Prediction for data compression. Initial results indicate that these compressors effectively capture meaningful information from the data while demonstrating significantly better stability and reliability compared to Uniform Random Sampling and state-of-the-art coreset constructions.

1. Introduction

Modern machine learning systems increasingly rely on vast amounts of data to generalise successfully across a wide range of predictive tasks. The "success" of a learner is often assessed through its error bounds [1], which are closely tied to the volume of data utilised during the learning process. Essentially, having more information enhances the likelihood of effective generalisation. However, the algorithmic implications of dealing with large datasets can lead to the depletion of computational resources and storage capacity, rendering the task of uncovering patterns within these extensive datasets quite challenging.

A line of work started in Riquelme-Granada et al. [2] explored the implications of using a surrogate dataset, a coreset [3], to speed-up the instantiation of Conformal Prediction (CP) [4]: a frequentist framework to statistically calibrate the prediction of standard machine learning algorithms such as logistic regression, support vector machines, neural networks, etc. Such calibration process generates *set-valued* predictions that are subject to strong error bounds. Riquelme-Granada et al. [2] demonstrated that an effective data compression methodology can not only save significant computing time during training but also retain important properties of the original dataset, as measured by the efficiency of the resulting conformal predictors. Similar arguments were presented in Riquelme-Granada et al. [5] regarding the probability calibration of Venn-Abers predictors [6], which were instantiated over small coresets for the problem of Support Vector Machines.

In this work, we will further pursue the aforementioned line of research that integrates the concepts of data compression and conformal prediction, but with an important twist: we propose to reverse the arguments presented in Riquelme-Granada et al. [2] and explore the idea of utilising CP to identify a meaningful representation of a dataset, which can then be employed for machine learning tasks. That is, instead of using conformal prediction as a *post hoc* procedure to scrutinise the calibration of learners trained on compressed data, we propose to use CP as a tool for computing the data compression itself. Technically, we draw parallels between the concept of non-conformity scores from CP and that of *sensitivity* from the coreset literature [7], proposing an importance sampling distribution concentrated on these scores for performing data selection. Similar to Riquelme-Granada et al. [2], we assume a *Logistic Regression* (LR) setting. We refer to this new method for reducing data complexity as *Conformal Compressors* (CC), and in the next couple of pages, we will present its motivation and examine its potential.

The contributions of our work can be summarised as follows:

- **Conformal Compressors:** Inspired by the algorithmic framework of coresets, we propose the novel idea of using conformal information generated by CP to define a measure of *importance* for data points in a dataset. This notion is then used to non-uniformly sample *representative* examples from the dataset and collect them into a small summary of data, whose size is just a small fraction, *e.g.* 1 %, of the original dataset size.

* Corresponding author.
E-mail address: N.A.Riquelme-Granada@soton.ac.uk (N. Riquelme-Granada).

https://doi.org/10.1016/j.patcog.2025.112515
Received 31 October 2024; Received in revised form 11 September 2025; Accepted 25 September 2025
Available online 3 October 2025

- **Empirical results:** We present initial empirical results for the performance of CC on public datasets. As is standard in the data reduction literature, we compare our idea against Uniform Random Sampling (URS) and recent coreset constructions for LR. Our results indicate that CC is preferable over both URS and coresets because the predictors trained over the resulting summaries exhibit higher and more stable predictive performance.
- **Extension of the research initiated by** Riquelme-Granada et al. [2]: We extend the foundational work established by Riquelme et al. by introducing innovative methodologies that apply the principles of coreset construction within the context of conformal prediction. Hence, we continue to advance this line of research that explores the interplay between Reliable Machine Learning [8] and Data Compression.

The paper is organised as follows. Section 2 discusses the Logistic Regression learning problem and conformal prediction. Section 3 covers preliminaries for coresets and introduces our main contribution: Conformal Compressors. In Section 4, we present empirical results and discussions on the initial findings on conformal compressors. Section 5 provides an overview of related work. Finally, we summarise our contributions and outline future research directions in Section 6.

2. Binary Logistic Regression and Conformal Prediction

In this section, we provide a comprehensive overview of the problem of Logistic Regression (LR), alongside standard concepts from conformal prediction, addressing both its transductive and inductive versions.

2.1. Binary classification with Logistic Regression

Our focus is on the binary classification problem using logistic regression, a widely recognised form of a generalised linear model. We choose the binary framework as it simplifies the discussion and is easily extendable to multi-class situations through approaches such as "one-against-all" or "one-against-one" [9].

Formally, consider a dataset $D := \{(x_n, y_n)\}_{n=1}^{N}$, where $x_n \in \mathbb{R}^{d+1}$ represents a feature vector *extended* with a 1 to account for the *bias term* and $y_n \in Y$ its corresponding label. Let $Y = \{-1, 1\}$ be a binary label space. For LR, the probability of observing $y_n = 1$, the *positive case*, given parameters $\theta \in \mathbb{R}^{d+1}$ is defined as:

$$p_{logistic}(y_n = 1|x_n; \theta) := \frac{1}{1 + \exp(-x_n \cdot \theta)}. \tag{1}$$

For observing the *negative case*, $y_n = -1$, the logistic probability is:

$$p_{logistic}(y_n = -1|x_n; \theta) := \frac{1}{1 + \exp(x_n \cdot \theta)}. \tag{2}$$

Combining the definitions in Eqs. (1) and (2), we write the likelihood for observing *any label* y_n as:

$$p_{logistic}(y_n|x_n; \theta) := \frac{1}{1 + \exp(-y_n x_n \cdot \theta)}. \tag{3}$$

Using the *per-point likelihood* defined in Eq. (3), and applying the logarithmic function as in Shalev-Shwartz and Ben-David [10], we obtain the *log-likelihood function*, which aggregates over the full dataset of N observations, D, as follows:

$$LL_N(\theta|D) := -\sum_{n=1}^{N} \ln(1 + \exp(-y_n x_n \cdot \theta)). \tag{4}$$

Eq. (4) is the *objective function* for LR, where we seek the optimal parameter $\hat{\theta}$ by maximising the log-likelihood function $LL_N(\theta|D)$. In practice, however, it is more standard to minimise the *negative log-likelihood* over $\theta \in \mathbb{R}^{d+1}$:

$$\mathcal{L}_N(\theta|D) := -LL_N(\theta|D)$$
$$= \sum_{n=1}^{N} \ln(1 + \exp(-y_n x_n \cdot \theta)).$$

Hence, the LR optimisation problem is written as in Eq. (5):

$$\theta^* = arg\,min_{\theta \in \mathbb{R}^{d+1}} \mathcal{L}_N(\theta|D). \tag{5}$$

Our work relies on exploiting *redundancy* in the data, as not all points in the dataset contribute equally to the learning process. Some points are more significant in influencing Eq. 5, which forms the basis of our strategy: to identify and assign weights to points based on their contribution to the LR optimisation problem. The basis for identifying such points lies in using non-conformity scores from CP, which we introduce in the next section, and importance sampling from Coresets, which we detail in Section 3.

2.2. Conformal Prediction framework

We now present the necessary technical details to inscribe the machine learning (ML) setting described in the previous section within the CP framework. First, we define CP in its original, transductive setting; then we extend the definitions to the more standard inductive ML scenario.

Conformal Predictors, as discussed in Vovk et al. [11], are a type of Confidence Predictor that measure conformity to generate prediction *sets* for each test object, containing a *p-value* for each potential label. These p-values provide a robust measure of reliability, reflecting the probability of an object possessing a particular label under the IID assumption.

In this subsection, we adopt a compact notation for representing feature vectors and labels: $z_i = (x_i, y_i)$ denotes the *i-th* example in the finite sequence Z^*. As defined in Section 2.1, $x_i \in \mathbb{R}^{d+1}$ and $y_i \in \{1, -1\}$.

Given any natural number N, a non-conformity measure $\mathcal{A}$ assigns a sequence of real-valued non-conformity scores $(\alpha_1, \alpha_2, \ldots, \alpha_N)$ to each sequence $(z_1, z_2, \ldots, z_N)$, maintaining invariance with permutations [8]. For an unlabelled object x_{N+1}, the Conformal Predictor, determined by $\mathcal{A}$, is defined as in Vovk et al. [12]:

$$\Gamma^{\epsilon}(z_1, z_2, \ldots, z_N, x_{N+1}) := \{y : p^y > \epsilon\}, \tag{6}$$

where $\epsilon \in [0, 1]$ is a user-defined significance level and, for each $y \in Y$, the *p-value* p^y is defined as:

$$p^y = \frac{|\{i = 1, 2, \ldots, N+1 : \alpha_i^y \geq \alpha_{N+1}^y\}|}{N+1}. \tag{7}$$

The non-conformity scores are given by:

$$(\alpha_1^y, \alpha_2^y, \ldots, \alpha_N^y, \alpha_{N+1}^y) := \mathcal{A}(z_1, z_2, \ldots, z_N, (x_{N+1}, y)). \tag{8}$$

In the CP framework, existing machine learning algorithms, termed *single-point predictors*, serve as subroutines. They are integral for defining $\mathcal{A}$, as outlined in Vovk et al. [11].

The process for computing the prediction set in Eq. (6) may vary depending on the specific problem context. For a feature vector x_{N+1}, we compute Eq. (7) per pair $(x_{N+1}, y)\,\forall y \in Y$ and filter the $|Y|$ p-values via Eq. (6). This procedure is repeated for each subsequent example.

To build more intuition on these fundamental formulations in CP, each α_i in Eq. (8) is really a score that measures the *strangeness* of example z_i with respect to the rest of the examples $(z_1, z_2, \ldots, z_{i-1}, z_{i+1}, \ldots, z_N, (x_{N+1}, y))$. The greater the score, the more unusual the example is, according to $\mathcal{A}$. Since the goal of measuring non-conformity is to make a statement about whether the pairing (x_{N+1}, y) is a reasonable one, Eq. (7) compares the relative strangeness of this pair, *i.e.*, with respect to the strangeness of all other examples. If a large fraction of α_i are larger than α_{N+1} (the strangeness score of object x_{N+1} with a postulated label y [1]), then it is likely that y is a good label for x_{N+1}, since this event is not very strange. If the opposite happens and most of the α_i are smaller than α_{N+1}, then this indicates that y is indeed a very

[1] This is why we write y and not y_{N+1}. y represents only *the hypothetical case* where it is the true label for the new object x_{N+1}, while y_{N+1} *is the true label* for the object.

strange label for x_{N+1}. Eq. (7) conveys this information in the form of a p-value for each $y \in Y$, where high values indicate stronger evidence for y being a good label candidate for x_{N+1}, and low values indicate a lack of evidence supporting that belief. Finally, after obtaining one p-value for each $y \in Y$, Eq. (6) produces the final prediction set that includes all labels for which Eq. (7) assigned p-values greater than the significance level ϵ. CP guarantees that the true label y_{N+1} will be included in the prediction set from Eq. (6) with probability at least $1 - \epsilon$ [11].

2.2.1. Inductive Conformal Predictors

Examining the definitions of Conformal Predictors reveals their original design for an online setting, benefitting from improved prediction accuracy as the training sequence lengthens. Nonetheless, they require recomputation from scratch for every new example. Consequently, these *transductive* algorithms quickly become computationally expensive for large data sets, limiting their applicability.

Inductive Conformal Predictors (ICPs) [13] offer an *inductive* alternative, enhancing the usability of their transductive counterparts by mitigating computational demands and relaxing the online constraint. The trade-off involves reduced validity and efficiency, resulting in less precise and larger prediction sets. For an in-depth discussion on *transductive versus inductive* learning, refer to Vapnik [14].

To utilise ICPs, the training set $(z_1, z_2, \dots, z_N)$ is divided into:

- The proper training set $(z_1, z_2, \dots, z_m)$ of size $m < N$;
- The calibration set $(z_{m+1}, z_{m+2}, \dots, z_N)$ of size $N - m$.

Typically, the training set is larger than the calibration set. The definitions from Section 2.2 adapt to this setting: the non-conformity measure $\mathcal{A} : Z^m \times Z \to \mathbb{R}$ requires:

$$p^y := \frac{|\{i = m+1, m+2, \dots, N+1 : \alpha_i^y \geq \alpha_{N+1}^y\}|}{N - m + 1}, \tag{9}$$

with non-conformity scores:

$$\alpha_i = \mathcal{A}(z_1, z_2, \dots, z_m, z_i),\ i = m+1, m+2, \dots, N; \tag{10}$$

$$\alpha_{N+1}^y = \mathcal{A}(z_1, z_2, \dots, z_m, (x_{N+1}, y)). \tag{11}$$

Notice that Eqs. (9) and (10) are re-definitions of Eqs. (7) and (8), respectively. The modifications are subtle: they depict the same computations with the important difference that they are performed over the calibration set, which is of fixed size. Hence, ICPs evaluate the conformity of a new example in relation to the training set and calibration set. They significantly reduce computation time by calculating the non-conformity measure for each calibration example only once. A single non-conformity score is computed for each new example to compare against existing scores, making ICPs advantageous for large-scale machine learning compared to CPs. Riquelme-Granada et al. [2] reveals how a coreset-based strategy can further enhance the instantiation of ICP in terms of computing time.

Finally, to complete our exposition of ICP, we show how to use the LR model θ^* (Eq. (5)) to define a non-conformity measure, thereby making the connection between single-point predictors and conformal predictors explicit. The non-conformity scores α_i (Eq. (10)) can then be written as follows:

$$\alpha_i = \mathcal{A}(z_1, z_2, \dots, z_m, z_i) = \Delta(y_i, f_{\theta^*}(x_i)), \tag{12}$$

where $f : \mathbb{R}^{d+1} \to \mathbb{R}$ is a function that, parameterised by the LR model θ^* trained on the proper training set $(z_1, z_2, \dots, z_m)$, returns the logit for the positive class, and $\Delta : Y \times \mathbb{R} \to \mathbb{R}$ is a discrepancy function between a label and a prediction [2] [13]. To simplify the notation, let the logit for the positive class for object x_i be $o_i = f_{\theta^*}(x_i)$. For LR, a natural choice of discrepancy function is proposed by Vovk et al. [11]:

$$\alpha_i = \Delta(y_i, o_i) = \begin{cases} 1 + \exp(-o_i), & \text{if } y_i = 1; \\ 1 + \exp(o_i), & \text{if } y_i = -1. \end{cases} \tag{13}$$

[2] Notice that α_{N+1} can also be written as Eq. (12)

Therefore, the resulting non-conformity scores will be large when the LR model does not consider y_i a suitable label for x_i, and small otherwise.

In the next section, we present the framework of coresets and how they define a non-uniform importance distribution over a dataset to select critical examples from it. With that notion in mind, we then show that non-conformity scores, in their own right, can be used to implement a similar mechanism for detecting redundancy in data.

3. Coresets & Conformal Compression

Having discussed the background concepts of transductive and inductive CP, this section introduces the concept of data summarisation, presenting coresets as a principled method for achieving approximately correct summaries of data. Building on coreset approaches, we then present our contribution: Conformal Compressor (CC), an approach that leverages the non-conformal information generated by CP to craft informative data summaries.

3.1. Summarising data

As previously mentioned, having more data means access to more information from which, hopefully, we will get a better understanding of an underlying truth. Learning becomes harder as datasets grow because many machine learning algorithms use computationally expensive numerical solvers to optimise an underlying objective function. Given that not all data points in a dataset are equally relevant to this optimisation problem, substantial research effort has been put in identifying the portion of data that is more important from the optimisation point of view. The set constructed with the more important portion of data can be interpreted as a summary of the original dataset since it contains sufficient information to provide a (provably) *good* solution to the optimisation problem. It is common to refer to such a summary as a "compression" of the original data [15], emphasising the fact that the portion of important points is small compared to the original full data.

3.2. Coresets

Coresets are a powerful algorithmic framework that has been used to analyse big and complex data. A coreset is a small *weighted* set of data *i.e.* a summary, such that the solution found in the summary is provably correct with respect to the solution found in the full data. Ideally, a coreset should be significantly smaller than the original dataset. Furthermore, state-of-the-art coreset results involve coresets whose sizes are *independent* of the original data size [16].

Most machine learning problems involve defining an optimisation problem to estimate model parameters or to describe other aspects of the data, as shown in the previous section for LR. Thus, for a given learning problem, a coreset can be constructed to reduce the volume of data. Learning can then proceed using the coreset alone, discarding the rest of the data, while still guaranteeing *approximately* the same result [17].

3.2.1. Coreset design

Any algorithm that constructs a coreset for some dataset can only provide provably correct guarantees for one specific learning problem. Hence, the design of a coreset construction algorithm entirely depends on the problem of interest. Nevertheless, we can still define coresets in the following problem-agnostic fashion.

Definition 1. (Δ-coreset): Let $f : \mathbb{R}^d \to \mathbb{R}$ be the objective function of some learning problem and $D \subseteq \mathbb{R}^d$ be the input data. We call C a Δ-coreset for D if the following inequality holds:

$$|f(D) - f(C)| \leq \Delta |f(D)|. \tag{14}$$

What makes coresets highly desirable is that they are not merely a heuristic approach for performing data reduction; they come equipped with theoretical guarantees, as generally stated in Eq. (14). We say that if such a guarantee can be obtained by approximating the original dataset D with a weighted set C with respect to function f, then C is called a Δ-coreset for D.

Notice that Δ establishes a bound for the solution quality found in the coreset. That is, Δ defines *how far* a solution found in the coreset can be from a solution found in the original dataset.

There are three general quantities that need to be bounded when designing a coreset algorithm for an optimisation problem: 1) the size M of the coreset; 2) the error tolerance Δ, *i.e.* maximum discrepancy between solutions found in the full data and in the coreset; and 3) the running time of the algorithm, *i.e.* if constructing the coreset is computationally comparable to solving the problem using the full data, then it becomes difficult to justify computing a coreset in the first place.

Designing a coreset algorithm for a learning problem is a challenging task. In fact, as stated in Munteanu et al. [18], not every problem allows for a practical coreset construction. For the problem of LR, however, a good number of coreset constructions have been proposed in the specialised literature; see Huggins et al. [19], Tolochinksy et al. [20], Munteanu et al. [21], and Munteanu et al. [18].

3.2.2. Coresets via importance sampling

A naive randomised approach to construct coresets is by doing uniform sampling. We can simply assign probability $1/N$ to each point in D and then sample according to this distribution. The problem with this simple approach is that we have to sample a very large number of points to ensure relatively low error [22]. This happens because different points make different contributions to the objective function at hand and hence we can easily leave important points out of the coreset if we do not sample sufficiently large coresets. Hence, a more sophisticated sampling scheme is needed.

The most effective randomised approach for constructing coresets is by doing non-uniform sampling, namely *importance sampling* [23]. Under this approach, rather than just assigning equal probability to all our input data points, we first compute an importance score that tells us "how redundant" a data point is for our learning problem. This score is called the *sensitivity* of the point, and is the central quantity for constructing coresets non-uniformly. Once we computed the sensitivity for each input point, we sample M points according to these sensitivity scores. The final step is to compute the *weights* for each sampled point, which are generally inverse-proportional to the sensitivity scores, and return the M weighted points.

When constructing coresets using importance sampling, defining and computing the sensitivity is very challenging because (i) computing the exact sensitivity is not computationally tractable; that is, we need to define lower and upper bounds for it and prove that using these bounds yields small coresets, and (ii) the bounds should be efficiently computable. On this front, it is worth mentioning that coreset algorithms need to inspect each input point at least once. Hence, by "efficiently computable" we mean near-linear time.

Fig. 1 shows an example of how a coreset approach, proposed by Huggins *et al.* [19], uses importance sampling over synthetic input data to obtain a weighted coreset. As explained in Feldman [7], one of the key theoretical challenges when working with importance-sampling based coreset constructions is the bounding of the sensitivity measure. For the algorithm depicted in the figure, the authors used a k-clustering subroutine to upper-bound the total sensitivity. For the technical details, we refer the reader to Huggins et al. [19], Lemma 3.1, p. 4.

To ground the discussion in intuition, notice that the k-clustering bound brings an intuitive interpretation of sensitivity: points that are bunched together are redundant, whereas points that are far from other points exert more influence over the LR objective function Reddi et al. [24], Huggins et al. [19]. Hence, points far from the cluster centres are assigned high sensitivity scores while nearby points will get low scores. After computing the sensitivities, the algorithm defines the non-uniform distribution based on these values.

As detailed in Section 2.2, valid prediction sets are generated by comparing the *strangeness* of examples. That is, through its internal workings, CP provides strong non-conformity measurements that can be interpreted as a form of sensitivity regarding the model's knowledge. When considering ICP, calibration points that are not well supported by the proper training set are assigned high non-conformity scores. Hence, such nonconforming points constitute important sources of uncertainty for the final prediction set. Somewhat analogous to the sensitivity scores used in coresets, then, non-conformity scores could be understood as quantifiers of the importance of individual data points based on the amount of uncertainty *they induce* into the whole prediction system. This interesting parallel between CP and sensitivity-based coreset algorithms is the bedrock of the idea we are introducing in the next section.

3.3. Introducing Conformal Compressors

We now turn to the main idea introduced in this work: Conformal Compressors (CC), which proposes using conformal prediction to inspect large datasets and ultimately select good representative samples from them. CC reverses the existing interactions between CP and coresets Riquelme-Granada et al. [2], Riquelme-Granada et al. [5], where a coreset is built as a proxy to the original dataset, and CP functions as a distribution-free uncertainty quantification tool that operates over the coreset to assess *how much uncertainty is retained* in the compressed data compared to the original dataset. We remind the reader that the main idea in data compression is to effectively *approximate* data; therefore, if the original examples do not provide a good representation of the underlying data-generating distribution, a *good approximation* to them will also suffer from the same shortcomings. We emphasise this to clarify that data approximation is not concerned with *addressing* the limitations and particularities of datasets, but rather with faithfully representing them.

Returning to our discussions in Section 3.2.2, we stated that coreset methods define the concept of *sensitivity* and that a non-uniform distribution is concentrated on this measure to probabilistically choose influential points from a set of points. The assumption here is, as noted by Reddi et al. [25], that points that are far from others, or points that are members of very small clusters (i.e., sensitive points), make much more significant contributions to the objective functions of learning problems, e.g., the LR objective defined in Eq. (5). However, coreset algorithms classically present three important particularities; they are:

1. **problem dependent:** This refers to the price we must pay for having strong approximation guarantees, as stated in Eq. (14). This means that when using a coreset construction, we can only benefit from its performance guarantees as long as we are compressing data for a particular target problem, e.g., LR. Using it for a different classification algorithm may reduce the method to a heuristic with intractable behaviour, which could perform worse than Uniform Random Sampling.
2. **complex to design:** This is closely related to the previous point. Recovering strong approximation guarantees for a new ML problem of interest requires instantiating a theoretical framework, as presented in Bachem et al. [22], Reddi et al. [25], Feldman and Langberg [23], or Munteanu et al. [18]. This could be a challenging endeavour, not only because it requires deep theoretical insights into the ML problem and its underlying mechanisms but also because some problems do not allow for coreset constructions (see Phillips [26] and Munteanu et al. [18]).
3. **oblivious to labelling information:** This arises from coresets being used in ML originally for unsupervised learning problems (see Section 5 for details). In supervised settings, however, we do possess ground truth labels at training time, which we believe could be useful information when selecting meaningful examples from the input data.

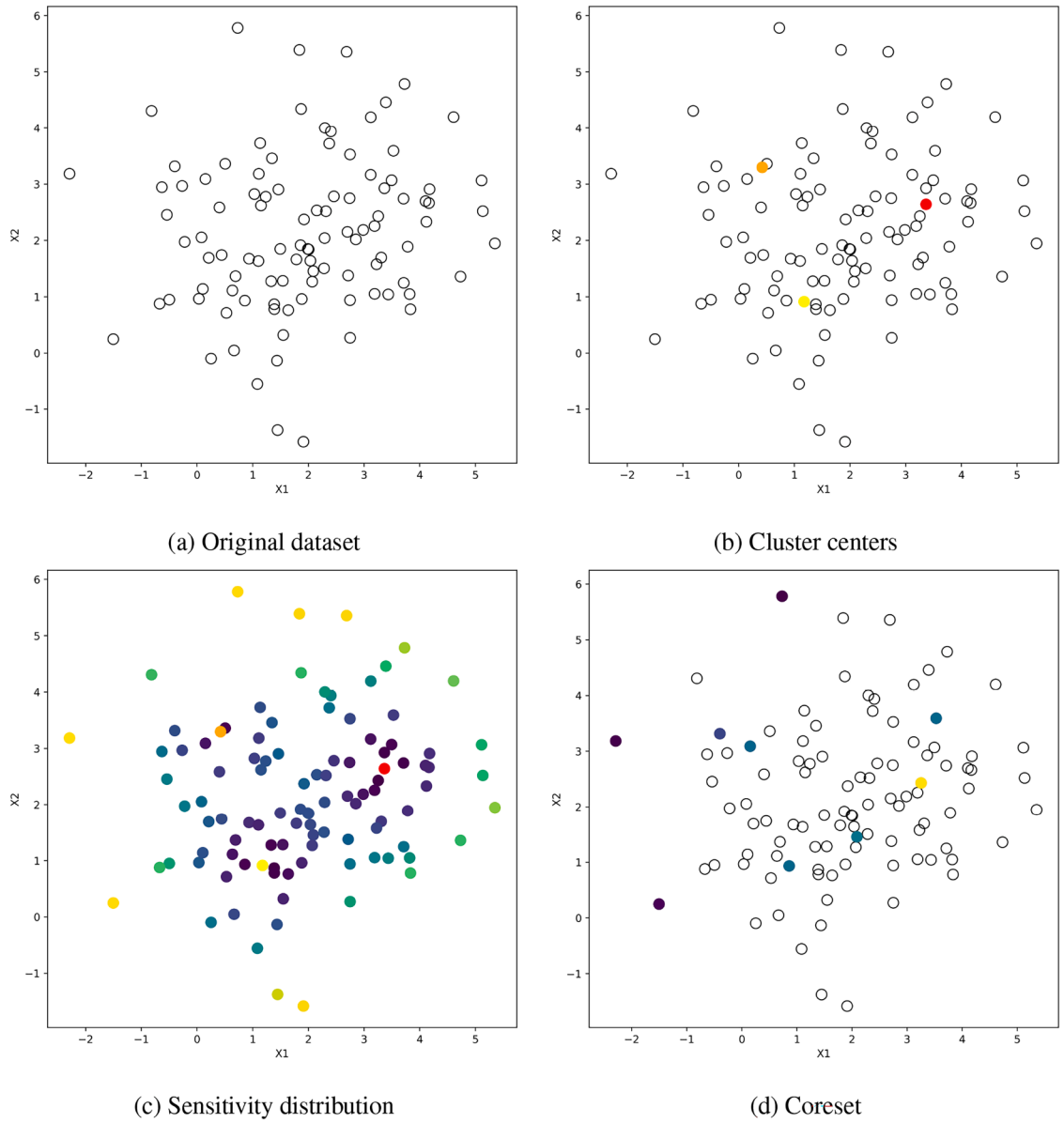


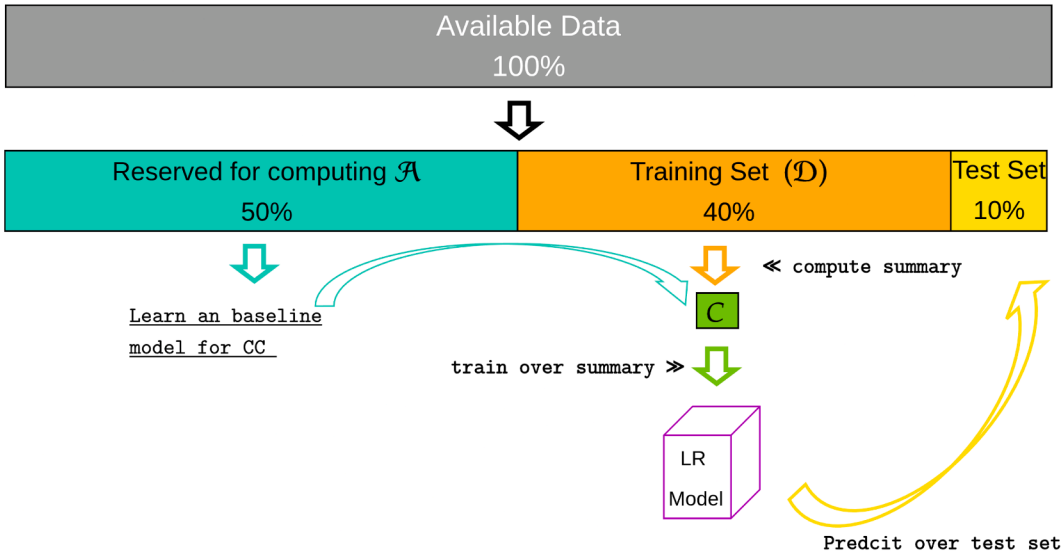
Fig. 1. The coreset construction from Huggins et al. [19] over a simple 2-dimensional synthetic dataset; (a) shows an illustrative synthetic 2-dimensional data for which a logistic regression coreset will be computed; the data has 100 points; (b) displays a k-clustering of the data to be used by the coreset algorithm to compute the sensitivities, with $k = 3$; (c) shows the sensitivity distribution for the dataset; the brighter the colour, the more sensitive the point is; (d) shows the obtained coreset, with the coreset size being 9 points; the colours here indicate the weights of points: the brighter the colour, the "heavier" the point is.

Upon close inspection of the above three points, and keeping in mind the idea of valuing points that are *sensitive* to the dataset, we realised that conformal prediction itself can be an effective means of addressing these limitations. Specifically, CP is concerned with capturing the *conformity* of the calibration set with a training set through the lens of a trained ML model. Therefore, these values could be interpreted as the *particularity* of points. Furthermore, CP is a general framework with mild assumptions on data, which can be deployed with virtually any ML model, thereby addressing the concerns raised in point 1 above; it provides a sound theoretical basis that guarantees error bounds (point 2), and finally, CP was designed for the supervised setting and therefore takes full advantage of labelling information (point 3). These are strong motivations for instantiating CC as an alternative to coresets for reducing the complexity of modern large-scale data.

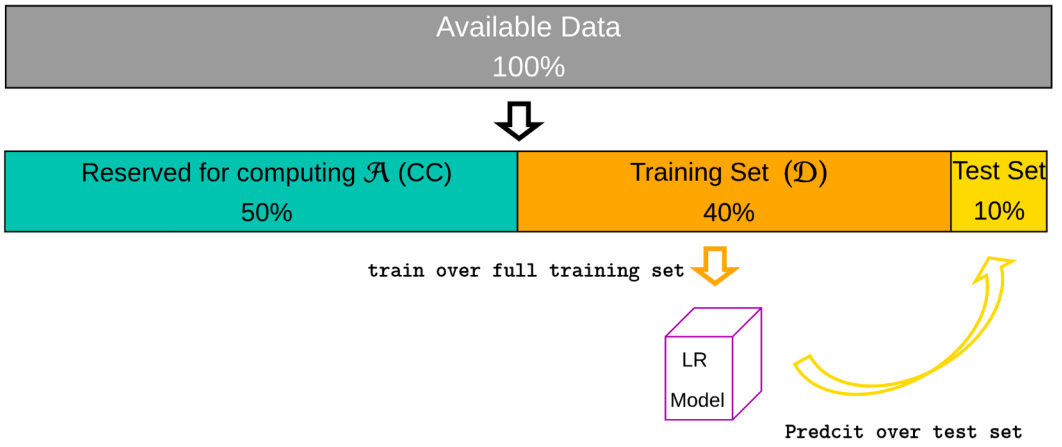
3.3.1. Non-conformity and sensitivity

Inspired by the arguments in coreset construction methods, we therefore propose utilising the non-conformity measure, as written in Eq. (12), as a mechanism for performing importance sampling and for finding a representative subset of a large dataset that is critical for learning a LR classifier. In essence, we link the two concepts of sensitivity and non-conformity to establish the first steps towards a systematic approach to summarising data focusing on the importance of data points through the *knowledge* of a trained model and the *uncertainty* it induces. Data points that yield higher non-conformity scores α_i are considered less conforming, indicating that they could be outliers or points about which the model is less certain. Consequently, these points can be prioritised for inclusion in the final compressed representation of the data, as they can be seen as *peculiar* examples which, if we were to train a fresh LR classifier on the dataset, would have a high impact on the LR optimisation problem. Importantly, since our method concentrates the sampling distribution on the α_i values, it naturally takes into account the labelling information in the dataset (see Eq. (13).

In Section 4, CC will be evaluated against standard sampling methods for summarising data. We will occasionally refer to the output of a conformal compressor as a *conformal summary*.



(a) Experiment Pipeline for the sampling-based data reduction methods: URS, CC, LogCore and SigCore. Only CC uses the auxiliary LR model learned from the Turquoise data block.



(b) Experiment Pipeline for the Full method, where no data reduction is performed.

Fig. 2. A diagram showing one iteration of our experiment pipeline, differentiating the cases where we train a LR classifier using data reduction methods (2a) and the case where we do not apply any data reduction (2b). Since CC is a data-driven reduction method, a portion of the available data that does not overlap with the training set is used to train a baseline model to compute non-conformity scores.

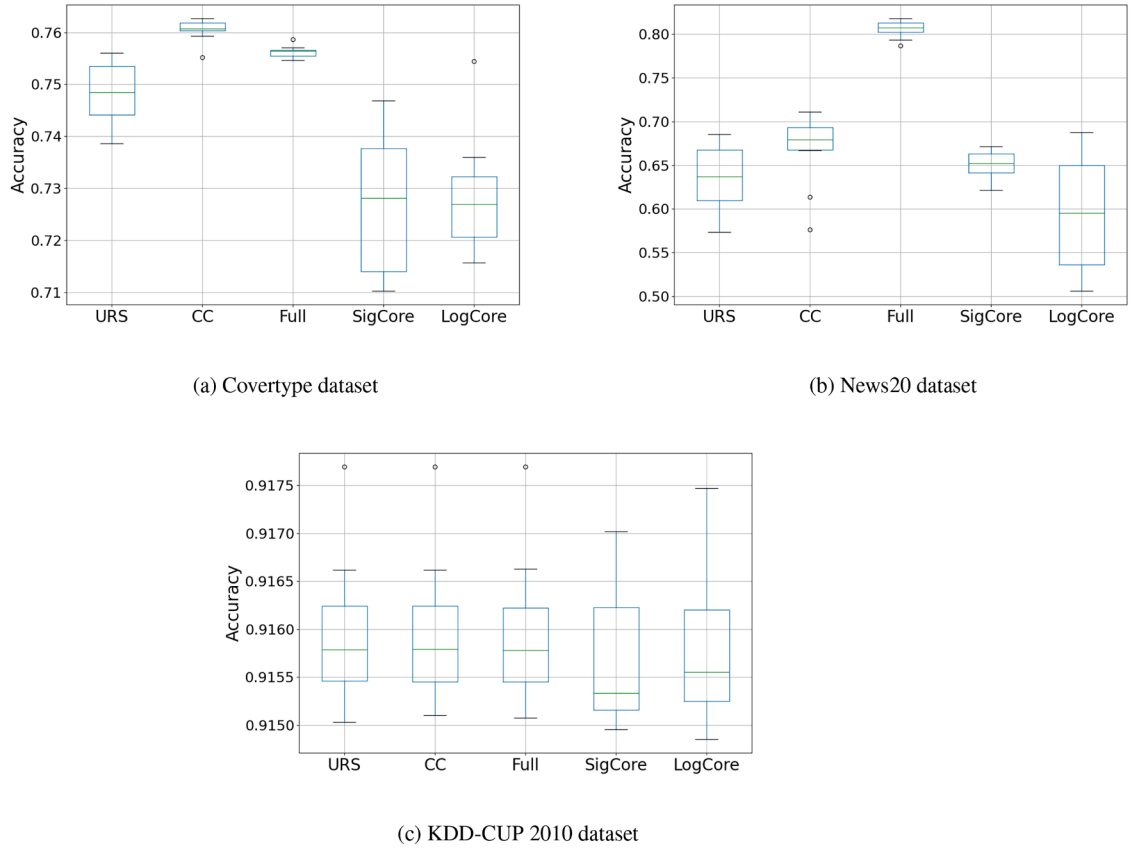
3.3.2. Computing a conformal summary

Algorithm 1 describes the computations that a conformal compressor performs to summarise a dataset D. The calculations can be broken down as follows:

- Calculate non-conformity scores for all examples in the dataset.
  - assuming a trained ML model is available, *i.e.* in our case, we assume a fully trained LR classifier, we inscribe it within the CP framework in the inductive setting, as detailed in Section 2.2.1. Crucially, this trained LR model has not seen the input dataset D as part of its training. D, then, can be considered as a calibration set for the ICP.
  - Compute the non-conformity score α_i for each example $z_i \in D$ using the non-conformity measure $\mathcal{A}$, as defined in Eq. (13) (line 2).
- Use the non-conformity scores to define an importance sampling distribution over the examples in D (lines 3–5). An example with high non-conformity score has better chances of getting chosen as representative than a point with a low non-conformity[3].
- Sample M points without replacement (lines 7–13). The resulting set C is the conformal summary for the given input dataset D.

Through this methodology, we aim to create a compressed dataset that not only preserves the predictive qualities of the original dataset but also retains essential information for making valid inferences. A conformal compressor, hence, serves as a *data-driven* alternative to coresets for reducing dataset size. As previously mentioned, CC is the first approach that utilises the machinery of conformal prediction to tackle the limitations of coresets discussed earlier in this section.

[3] It is not uncommon to instantiate CP using *conformity scores*, rather than non-conformity ones. If that is the case, for the non-conformity measure in Eq. (13), we suggest concentrating the sampling distribution on the $s_i = -\alpha_i$ conformity scores.



(a) Covertype dataset

(b) News20 dataset

(c) KDD-CUP 2010 dataset

Fig. 3. Comparison of the classification accuracy, over 10 runs, of URS, CC, Full, SigCore and LogCore across the three datasets. The central line in each box indicates the median. The bottom and top of the box indicate the 25th and 75th percentiles respectively. The small circles denote the outliers. The sample size M used by all sampling methods is 1 % of the training set.

Algorithm 1: Conformal Compressors (CC).

Input: Dataset to be compressed $\mathcal{D} = \{z_i = (x_i, y_i)\}_{i=1}^{N}$, trained LR model available via function f_{θ^*}, sample size $M \leq N$

Output: Conformal Summary $C \subseteq D$, $|C| = M$

for $i = 1$ **to** N **do**
  $\alpha_i = \Delta(y_i, f_{\theta^*}(x_i))$ // From Eq. (13)
$\bar{m} \leftarrow \frac{1}{N} \sum_{i=1}^{N} \alpha_i$ // aggregate non-conformity scores for normalisation
for $i = 1$ **to** N **do**
  $p_i = \frac{\alpha_i}{\bar{m}}$ // define importance sampling distribution
$C \leftarrow \emptyset$; $R \leftarrow \{1, \ldots, N\}$
for $t = 1$ **to** M **do**
  // sample M points without replacement
  $Z \leftarrow \sum_{j \in R} p_j$
  foreach $j \in R$ **do**
    $q_j \leftarrow p_j / Z$
  $i \sim \text{Categorical}(\{q_j\}_{j \in R})$
  $C \leftarrow C \cup \{(x_i, y_i)\}$
  $R \leftarrow R \setminus \{i\}$
return C

In the following section, we will detail our experimental results to illustrate the first empirical results on conformal compressors over public datasets, and we will benchmark it against strong baselines for sampling methods.

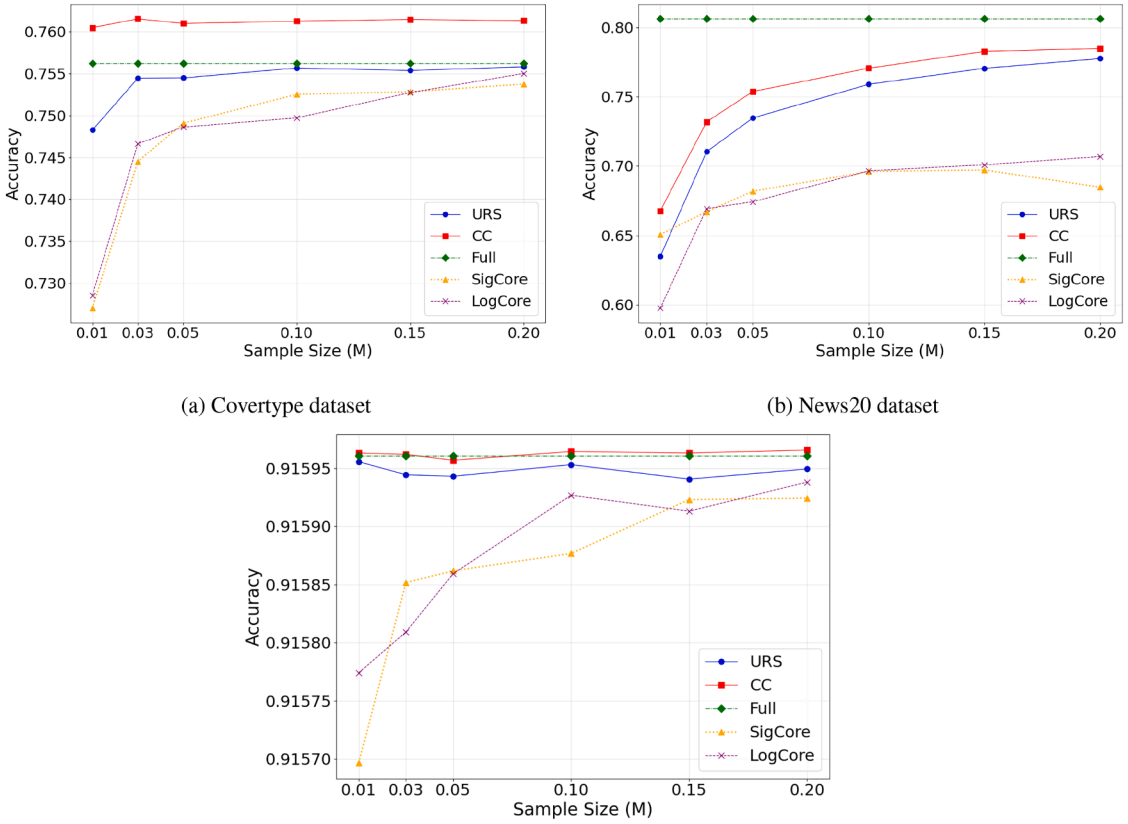
Table 1
The three datasets used in our experiments are Covertype, News20 and KDD-CUP 2010.

Dataset	Examples	Features
News20	19,996	1,355,191
Covertype	581,012	54
KDD-CUP 2010	8,918,054	20,216,830

4. Empirical evaluations

Having outlined our approach, this section presents our experimental results. Conformal compressors will be benchmarked against Uniform Random Sampling, which is the most standard test for data approximation techniques, *i.e.* see Huggins et al. [19], Riquelme-Granada et al. [27], and Riquelme-Granada et al. [5]. Additionally, we include two recent coreset constructions for Logistic Regression proposed by Tolochinksy *et al.* in Tolochinksy et al. [20]: the *logistic coreset* and the *Sigmoid coreset*, both of which are strong coreset construction methods that use importance sampling.

We evaluated the methods using the datasets detailed in Table 1. All the datasets are public and can be downloaded from https://www.csie.ntu.edu.tw/~cjlin/libsvm/. The Covertype dataset ([28]) contains cartographic features and the labels correspond to different forest cover types; KDD-CUP 2010 ([29]) is a massive dataset generated for an educational data mining competition; finally, based on 20,000 messages taken from 20 newsgroups, News-20 was generated in Keerthi and DeCoste [30] for experiments that needed both high data size and dimensionality.



(a) Covertype dataset

(b) News20 dataset

(c) KDD-CUP 2010 dataset

Fig. 4. Comparison of the accuracy scores across various sample sizes for URS, CC, Full, SigCore and LogCore across the three datasets. Remarkably, for Covertype dataset, CC outperforms even the Full method. CC is also much more predictable and performant than the rest of the sampling methods.

Our experiments focus on evaluating how effectively conformal compressors generate small proxy datasets by leveraging CP principles and comparing their performance against strong sampling alternatives.

We will shed light on the following key research questions:

- **Do Conformal Compressors provide improvements in learning performance compared to Uniform Random Sampling and Coresets?** This question will help establish whether small conformal summaries can maintain or enhance predictive performance.
- **Are Conformal Compressors computationally feasible?** This question will allow us to analyse whether the method is too demanding in terms of computing time.

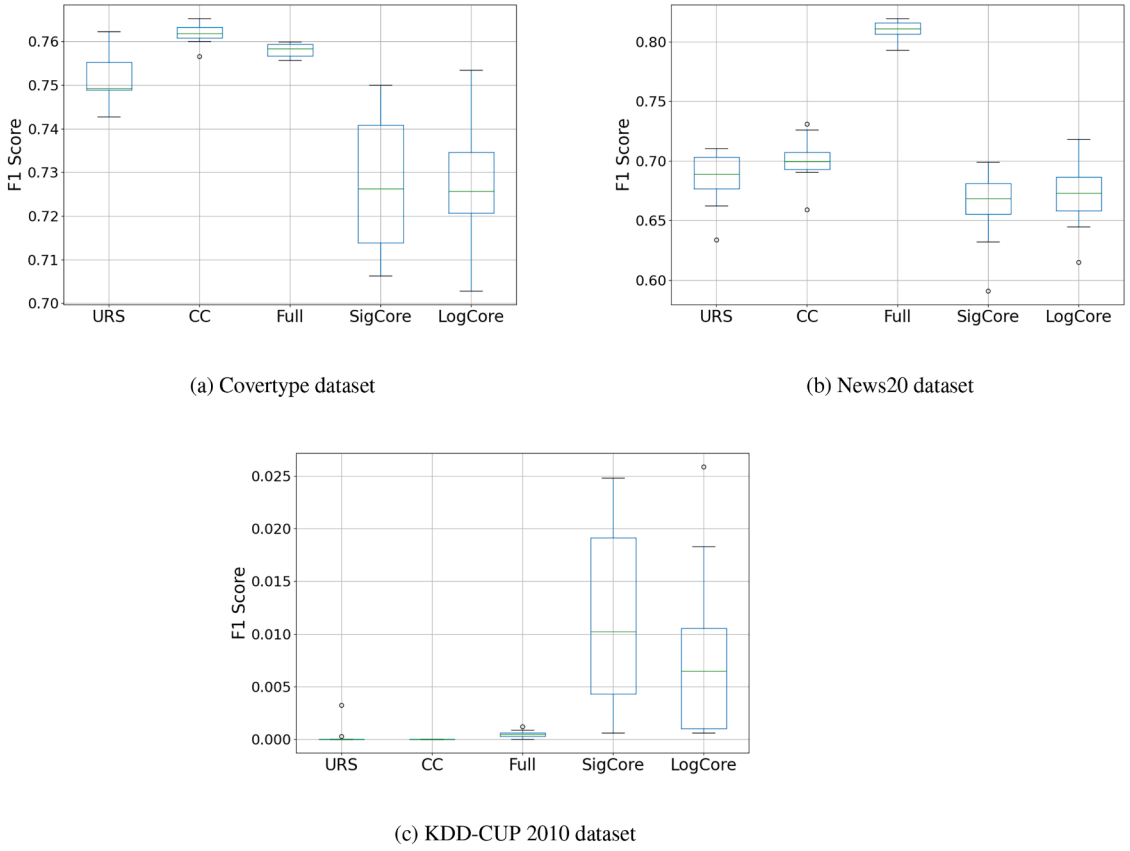
4.1. Evaluation procedure

We define five methods for our empirical evaluations: *Uniform Random Sampling* (URS), *Conformal Compressors* (CC), *Logistic Coreset* (LogCore), *Sigmoid Coreset* (SigCore) and *Full*, which involves not applying any sampling mechanism, i.e. using the entire input data for training a LR classifier. We remind the reader that URS acts as a form of coreset and serves as a baseline to gauge the effectiveness of more sophisticated data approximation techniques. Also, we shall refer to the set {URS, CC, LogCore, SigCore} as *sampling methods* when describing steps common to all of them. The main goal of the experiments is to assess the extent to which the predictive power of an LR classifier is retained when it is trained on a summary of the training dataset instead of the full dataset. In particular, we measure how the conformal summaries we compute compare with other well-known data reduction techniques with respect to standard ML metrics, each of which we will define later in Section 4. All sampling methods require a sample size, M, for producing a summary of the data. We consider values of 1 %, 3 %, 5 %, 10 %, 15 %, and 20 %; i.e., M is a percentage of the training dataset $\mathcal{D}$. For the Full method, we do not apply any data reduction. This method is useful for effectively measuring the loss in predictive performance incurred by sampling methods.

We now describe the evaluation protocol, which is depicted in Fig. 2.

- We split all available data into three parts: (1) a portion to be used for simulating a pre-trained model that conformal compressors need, (2) a training set, and (3) a test set. Since a trained model is necessary to instantiate CP, part (1) is fundamental for providing function f_{θ^*} to CC as input (see Algorithm 1). Part (2) is the training set, which we refer to in our notation as $\mathcal{D}$. This is the data that we wish to effectively summarise for the efficient learning of an LR classifier. We use part (3) to assess the predictive performance of the LR classifiers trained either on all of $\mathcal{D}$ (Full method) or on a summary of it (sampling methods).
- For CC, we train an LR model[4] on the data portion (1) to simulate a pre-existing trained model that we use in Algorithm 1. In reality, assuming the existence of such a model can be regarded as a reasonable assumption, given how integral methods such as transfer learning [31] or domain adoption [32] are to modern ML pipelines.
- For all sampling methods, we compute summaries of size M% of the training set following their respective procedures (Fig. 2a). The Full method utilises the complete training set (Fig. 2b).
- We train an LR classifier *on each* data summary (sampling methods) and on $\mathcal{D}$ (Full method), performing 20-fold cross-validation.
- Each resulting LR model makes predictions over the test set.

[4] Note that this model should not be confused with the one trained on the data summaries or the full dataset in Fig. 2.



(a) Covertype dataset

(b) News20 dataset

(c) KDD-CUP 2010 dataset

Fig. 5. Comparison of the F1 Score, over 10 runs, of URS, CC, Full, SigCore and LogCore across the three datasets. The central line in each box indicates the median. The bottom and top of the box denote the 25th and 75th percentiles, respectively, while the small circles represent the outliers. The sample size M used by all sampling methods is 1 % of the training set.

We repeat the above steps 10 times for each sample size $M \in \{1, 3, 5, 10, 15, 20\}\%$ of D for the sampling methods.

4.2. Metrics

We outline the evaluation criteria used to assess the performance of the five methodologies described:

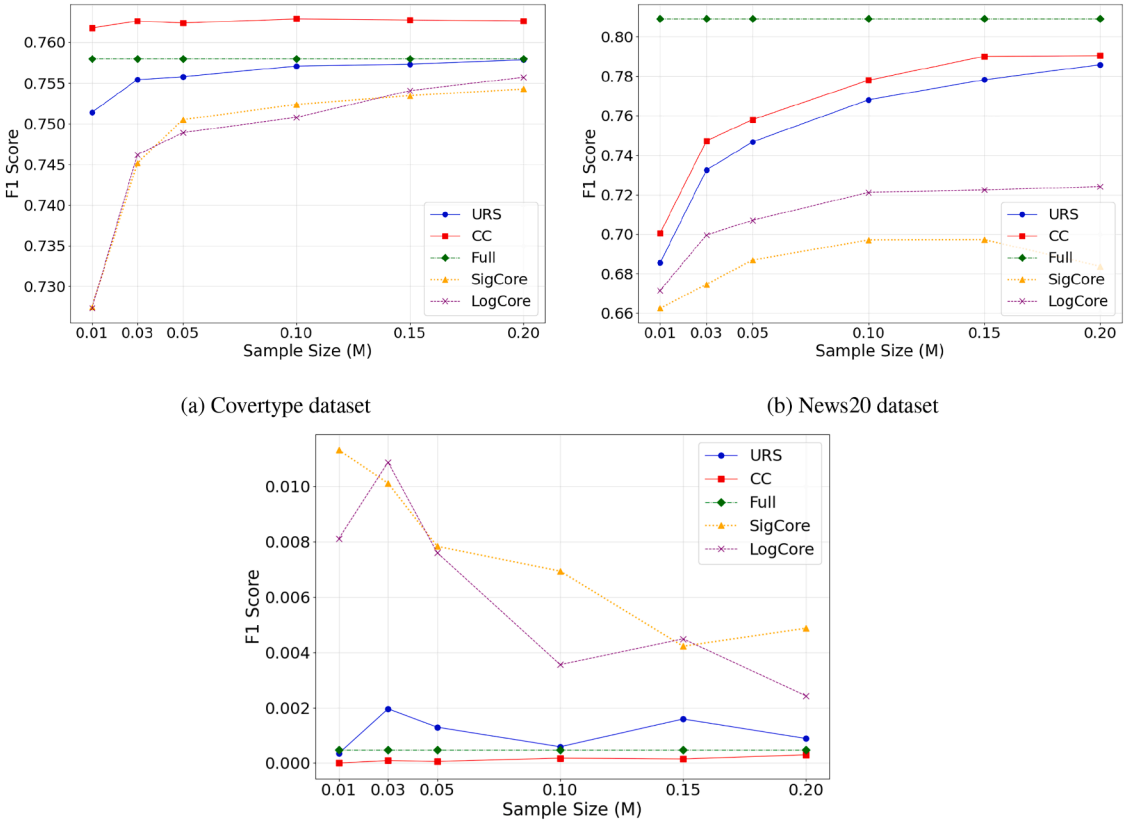
- **Classification Accuracy:** This measure is defined as the percentage of correctly classified test examples and is commonly used as a baseline metric for measuring performance in a supervised-learning setting.
- **F1 Score:** This score represents the harmonic average of precision and recall [33], computed as $F_1 := 2\frac{P*R}{R+P}$, where P is precision and R is recall [33]. A higher F1 score indicates better classification performance.
- **Compression Time**: This metric refers to the total time, in seconds, taken to generate the compressed dataset using each sampling method. It is of particular importance for *CC*, *LogCore*, and *SigCore*, as these are more intricate compression techniques. It is much less relevant for *URS*, since it is, by definition, the quickest approach for summarising data. It is completely irrelevant for the *Full* method, as it does not perform any data compression. For the sake of fairness, even though it is reasonable to assume that a pre-trained model can be used to instantiate CC, we consider as part of CC's compression time the training of a LR model using 5-fold cross-validation.

4.3. Accuracy results

The results displayed in Fig. 3 highlight the performance trends observed for the five methods across the three public datasets. The box plots illustrate the distribution of classification accuracy scores for each method, with the central line indicating the median value, while the edges of the boxes represent the 25th and 75th percentiles, respectively. Outliers are denoted by small circles.

From the figure, we observe that CC generates summaries that not only perform better than URS and the two coreset methods in most cases but also demonstrate greater reliability in terms of the variance it induces. Remarkably, CC achieves even higher accuracy than the *Full* method, despite only using 1 % of the full dataset. One possible explanation for this phenomenon is that by concentrating the sampling mechanism on the non-conformity scores, conformal compressors effectively remove examples that could lead the optimiser to become *stuck* in misleading local minima. This claim is based on intuition and has yet to be verified. However, it is not unusual to see some coresets improving the outcomes of machine learning algorithms (*cf.* Huggins et al. [19], Feldman and Langberg [23], Munteanu et al. [18]), which makes it interesting considering that conformal compressors are not coresets. Further, this observation can lead us to ask the research question whether the theory of CP can be used with theoretical arguments for coreset constructions to dote CC with strong compression guarantees.

Fig. 4 presents the mean accuracy scores across increasing sample sizes. This representation provides insight into how the accuracy of the sampling methods evolves as the size of the data representation changes, confirming the reliability and robustness of the proposed CC methodology, as it is in most cases lower bounded by URS. This behaviour is characteristic of sampling mechanisms that effectively detect meaningful patterns in the input data. The two coreset methods, quite surprisingly, underperform against CC and URS. They do improve with larger sample sizes; however, their performance, as measured by our standard machine learning metrics, is significantly inferior to that of conformal compressors.



(a) Covertype dataset

(b) News20 dataset

(c) KDD-CUP 2010 dataset

Fig. 6. Comparison of the F1 scores across various sample sizes for URS, CC, Full, SigCore and LogCore across the three datasets. Notably, for the Covertype dataset, CC outperforms even the Full method and reliably approximates its performance while being lower-bounded by URS, and even by the coreset algorithms, in most cases.

Additionally, the CC lines exhibit much more tractability than those of the other sampling methods, which can occasionally appear *spiky*.

4.4. F1 Score results

The results for the F1 Score across the three datasets are illustrated in Fig. 5. This metric provides a comprehensive view of the performance, balancing the contributions of precision and recall.

In general, the Full data approach demonstrates greater stability for the F1 Score, followed closely by CC, which also outperforms Full for the Covertype dataset. In contrast, the sampling methods show a significant decline in performance for the KDD dataset, potentially due to the imbalances present in this competition-grade dataset. Similar to the accuracy results, as shown in Fig. 6, CC displays a much more stable line when compared to URS and the coresets, which shows promising generalisation for different data. Notice that coresets here behave quite unpredictably, showing higher F1 scores for smaller sample sizes and then decreasing dramatically. It is clear that the KDD data is particularly challenging for these methods. Conformal compressors, on the other hand, behave quite closely to Full, virtually matching its predictive performance when the sample size is 20 %.

4.5. Compression time results

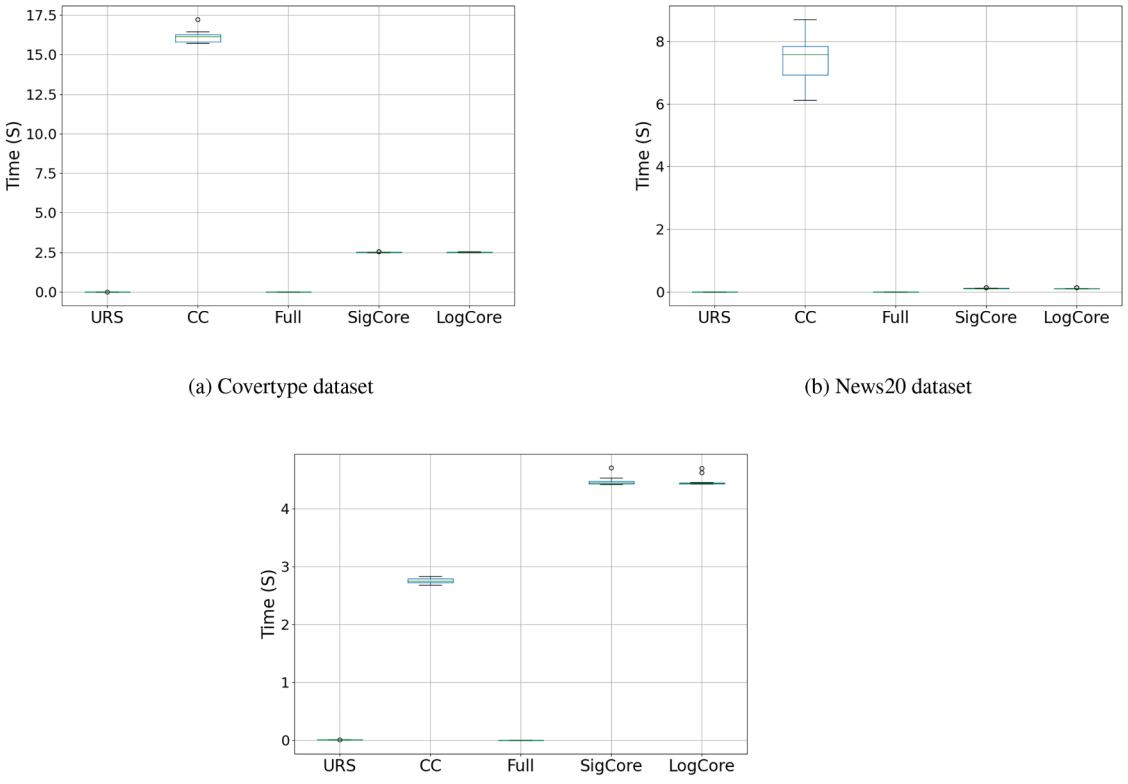
The compression time results, displayed in Fig. 7, highlight the computational efficiency of the various sampling methods in producing compressed datasets. URS, given its simplicity, demonstrates the shortest compression time, while the more sophisticated methods, CC, LogCore, and SigCore, show increased computational demands. For Covertype, CC costs 7x the time of the two coreset methods. A similar observation can be made about News20. For KDD, however, both coreset methods become more expensive than instantiating a conformal compressor.

It is important to consider that the plots in Fig. 7 assume a fixed sample size of 1 % for the sampling methods. If we look at Fig. 8, we can see the dependency between the methods' computational efficiency and the sample size parameter M. CC, just like URS and Full, is invariant with respect to M. SigCore and LogCore, on the other hand, exhibit increased computational times as the sample size M increases. This trend highlights the additional complexity involved in constructing coresets (see Tolochinksy et al. [20], Algorithm 1) compared to CC. Overall, while CC incurs a higher computational cost for smaller data, it scales much better than the coresets for the more challenging datasets, demonstrating its potential and viability for large-scale learning tasks.

5. Coresets in the literature

The technique of coresets was born in the field of computational geometry as a systematic approach for approximating the optimal solution for *shape-fitting problems* such as the *Minimum Enclosing Ball* (MEB) Agarwal and Sharathkumar [34], Badoiu and Clarkson [35], Bădoiu and Clarkson [36], Munteanu et al. [37] and *Directional Width* (DW) Agarwal et al. [38], Chan [39]. In machine learning, coresets have mainly been studied in the context of unsupervised learning Feldman et al. [15], Har-Peled and Mazumdar [40], Bachem et al. [22], Zhang et al. [41], Ackermann et al. [42] to devise fast approximation and streaming algorithms for computationally intractable problems, e.g., clustering.

Incursions in the supervised learning area are much more recent. Reddi et al. [25] explored coresets for the problem of *Empirical Risk Minimisation*, central to statistical learning theory. Specifically, for the Logis-

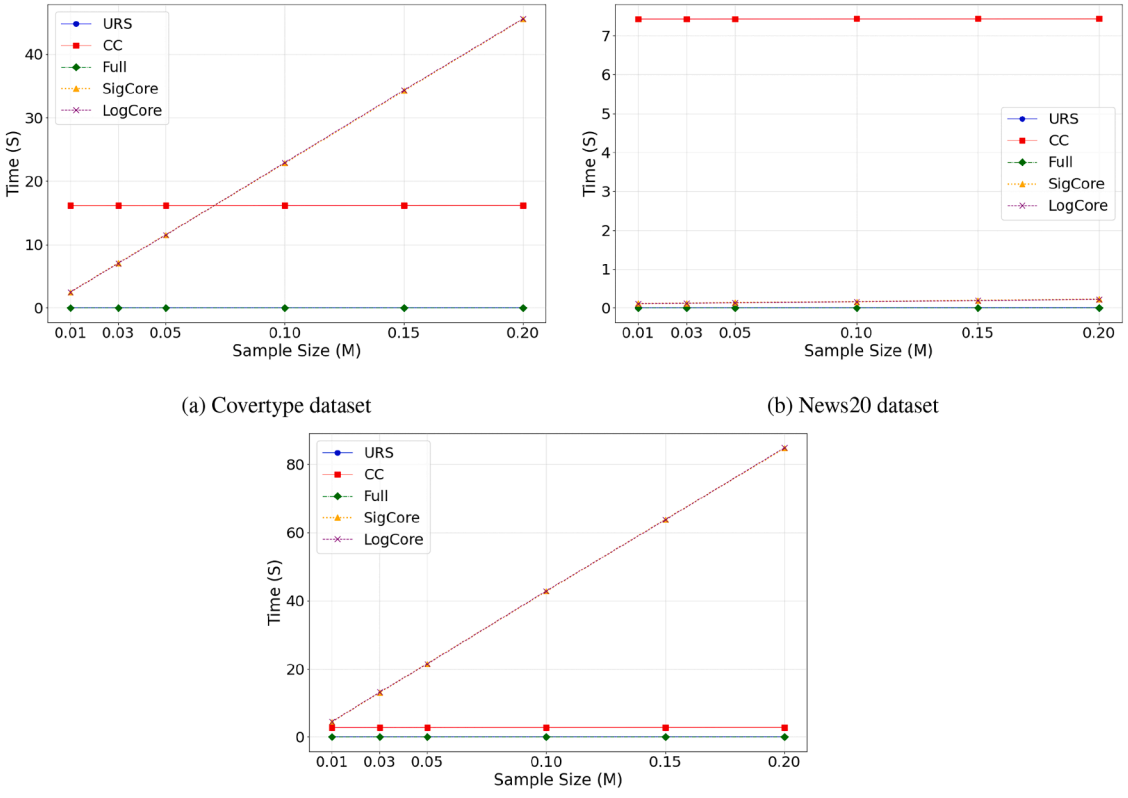


(a) Covertype dataset

(b) News20 dataset

(c) KDD-CUP 2010 dataset

Fig. 7. Comparison of compression times, over 10 runs, for URS, CC, Full, SigCore, and LogCore across the three datasets. The central line indicates the median, with the bottom and top indicating the 25th and 75th percentiles. For $M = 1\,\%$ of the training set, compression time for CC is more costly than that of the coresets, except for the KDD dataset.



(a) Covertype dataset

(b) News20 dataset

(c) KDD-CUP 2010 dataset

Fig. 8. Comparison of compression times across various sample sizes for URS, CC, Full, SigCore, and LogCore across the three datasets. CC's compression time is virtually independent from the sample sizes used. The two coreset methods quickly become more costly than CC for the more challenging datasets.

tic and Hinge loss functions, they proposed an iterative gradient-based coreset algorithm by exploiting the following observation: at each iteration of the optimisation process, only a small set of points are important, i.e., those that contribute the most to the objective function. Thus, they showed that the problem can be solved over this compact set of points, obtaining a provably good solution. One interesting observation made by Reddi *et al.* is that the process of computing the well-known gradient descent algorithm can be seen as an instantiation of their framework, where the coreset consists of the gradients computed at each iteration.

Huggins et al. [19] proposed a coreset algorithm assuming very similar supervised learning settings but from a Bayesian perspective, i.e., Bayesian Logistic Regression. They proved that, by approximating (up to a multiplicative factor) the log-likelihood of the observations using a weighted subset of the data, one can obtain a coreset with high probability. More recently, Campbell [43] has developed general lower and upper bounds for coresets in the Bayesian setting. Munteanu et al. [18] also focused on logistic regression, demonstrating that the impossibility of designing a sub-linear streaming algorithm for logistic regression implies that logistic regression admits no sub-linear coreset construction. Aligned with these two works, Braverman et al. [44] proposed a novel method that uses the sensitivity approach in a streaming setting to compress unbounded data streams into positively-weighted coresets. Their approximation guarantees, however, only hold for the problem of Least-Mean-Squares (LMS). Other problems that have more recently been studied through the lens of coresets include decision trees [45], time series [46], and a proliferation of works on deep neural networks [47–50].

Finally, the intersection between coresets and conformal prediction is an emerging area of research, with considerable potential for new discoveries. Gao et al. [51] propose a coreset to approximate Euclidean balls based on confident sets. They demonstrate how their theoretical results can be applied to CP in high-dimensional prediction sets. Zhang et al. [52] consider the problem of Label Distribution Learning and propose a learning model capable of deriving closed-form expressions for the label distribution mean, variance, and covariance conditional on a given sample. They show how their method can effectively quantify label distribution uncertainty and further illustrate that this information can be calibrated using conformal prediction. Additionally, they study the application of their method in a coreset-based active learning setting. However, none of these works can be compared to our work as they do not use CP to derive an importance sampling distribution, a classical approach in coresets, leveraging the non-conformity information of the data sample to be compressed.

6. Conclusion and future work

In this work, we introduced the method of Conformal Compressors, which, to the best of our knowledge, is the first data compression approach that employs conformal information to approximate data.

Our results demonstrate that conformal compressors consistently outperform uniform sampling and two competitive coreset constructions across multiple public datasets, showcasing enhanced predictive performance while maintaining a high degree of reliability. Notably, conformal compressors achieve results comparable to the Full method while utilising only a fraction of the dataset, illustrating their effectiveness in generating small proxy datasets. We are particularly interested in further studying this phenomenon. Additionally, our findings indicate that conformal compressors provide greater stability in performance across varying sample sizes, which could be promising for addressing the generalisation issues that coresets exhibit.

This work marks a step forward in developing the line of research initiated in Riquelme-Granada et al. [2], demonstrating the potential of integrating conformal prediction with data compression techniques. Looking ahead, future work will explore the generalisation of conformal compressors across various machine learning models. Additionally, we aim to investigate the use of the p-values from conformal prediction as informative elements in the compression process, which could refine the ideas presented thus far. Lastly, we plan to define appropriate weights for data points in the conformal summaries, akin to coresets.

CRediT authorship contribution statement

Nery Riquelme-Granada: Writing – review & editing, Writing – original draft, Project administration, Methodology, Investigation, Formal analysis, Conceptualization; **Khuong An Nguyen:** Writing – review & editing; **Zhiyuan Luo:** Project administration, Conceptualization, Writing – review & editing.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] V. Vapnik, The Nature of Statistical Learning Theory, Springer Science & Business Media, 2013.
[2] N. Riquelme-Granada, K. Nguyen, Z. Luo, Coreset-based Conformal Prediction for large-scale learning, in: Conformal and Probabilistic Prediction and Applications, 2019, pp. 142–162.
[3] E. Tolochinsky, D. Feldman, Coresets for monotonic functions with applications to deep learning, CoRR, abs/1802.07382 (2018).
[4] A. Gammerman, V. Vovk, Hedging predictions in machine learning: the second computer journal lecture, Comput. J. 50 (2) (2007) 151–163.
[5] N. Riquelme-Granada, K.A. Nguyen, Z. Luo, Fast probabilistic prediction for kernel SVM via enclosing balls, in: Conformal and Probabilistic Prediction and Applications, PMLR, 2020, pp. 189–208.
[6] V. Vovk, I. Petej, Venn-abers predictors, (2012). arXiv preprint arXiv:1211.0025
[7] D. Feldman, Core-sets: updated survey, In: Sampling Techniques for Supervised or Unsupervised Tasks, F. Ros, S. Guillaume, Springer, Cham, (2020) 23–44.
[8] V. Balasubramanian, S.-S. Ho, V. Vovk, Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications, Morgan Kaufmann, 2014.
[9] J. Milgram, M. Cheriet, R. Sabourin, "One against one" or "one against all": which one is better for handwriting recognition with SVMs?, in: Tenth International Workshop on Frontiers in Handwriting Recognition, Suvisoft, 2006.
[10] S. Shalev-Shwartz, S. Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, 2014.
[11] V. Vovk, A. Gammerman, G. Shafer, Algorithmic Learning in a Random World, Springer Science & Business Media, 2005.
[12] V. Vovk, V. Fedorova, I. Nouretdinov, A. Gammerman, Criteria of efficiency for conformal prediction, in: Symposium on Conformal and Probabilistic Prediction with Applications, Springer, 2016, pp. 23–39.
[13] V. Vovk, Conditional validity of inductive conformal predictors, in: Asian Conference on Machine Learning, 2012, pp. 475–490.
[14] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[15] D. Feldman, M. Schmidt, C. Sohler, Turning big data into tiny data: constant-size coresets for k-means, PCA and projective clustering, in: Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2013, pp. 1434–1453.
[16] V. Braverman, D. Feldman, H. Lang, New frameworks for offline and streaming coreset constructions, CoRR abs/1612.00889 (2016). http://arxiv.org/abs/1612.00889
[17] I. Jubran, A. Maalouf, D. Feldman, Introduction to coresets: accurate coresets, (2019). arXiv preprint arXiv:1910.08707
[18] A. Munteanu, C. Schwiegelshohn, C. Sohler, D. Woodruff, On coresets for logistic regression, Adv. Neural Inf. Process. Syst. 31 (2018).
[19] J. Huggins, T. Campbell, T. Broderick, Coresets for scalable Bayesian logistic regression, in: Adv. Neural Inf. Process. Syst., 2016, pp. 4080–4088.
[20] E. Tolochinksy, I. Jubran, D. Feldman, Generic coreset for scalable learning of monotonic kernels: logistic regression, sigmoid and more, in: International Conference on Machine Learning, PMLR, 2022, pp. 21520–21547.
[21] A. Munteanu, S. Omlor, D. Woodruff, Oblivious sketching for logistic regression, in: International Conference on Machine Learning, PMLR, 2021, pp. 7861–7871.
[22] O. Bachem, M. Lucic, A. Krause, Practical coreset constructions for machine learning, (2017). arXiv preprint arXiv:1703.06476
[23] D. Feldman, M. Langberg, A unified framework for approximating and clustering data, in: Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing, ACM, 2011, pp. 569–578.

[24] S.J. Reddi, J. Konečnỳ, P. Richtárik, B. Póczós, A. Smola, AIDE: fast and communication efficient distributed optimization, (2016). arXiv preprint arXiv:1608.06879
[25] S.J. Reddi, B. Póczos, A.J. Smola, Communication efficient coresets for empirical loss minimization, in: UAI, 2015, pp. 752–761.
[26] J.M. Phillips, Coresets and sketches, (2016). arXiv preprint arXiv:1601.00617
[27] N. Riquelme-Granada, K.A. Nguyen, Z. Luo, Coreset-based data compression for logistic regression, in: International Conference on Data Management Technologies and Applications, Springer, 2020, pp. 195–222.
[28] J.A. Blackard, D.J. Dean, Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables, Comput. Electron. Agric. 24 (1999) 131–151.
[29] J. Stamper, A. Niculescu-Mizil, S. Ritter, G.J. Gordon, K.R. Koedinger, Algebra I 2008–2009. Challenge data set from KDD Cup 2010 Educational Data Mining Challenge, Carnegie Mellon University, Pittsburgh, PA, (2010), Retrieved April 25, 2015. http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp
[30] S.S. Keerthi, D. DeCoste, A modified finite Newton method for fast solution of large scale linear SVMs, J. Mach. Learn. Res. 6 (Mar) (2005) 341–361.
[31] K. Weiss, T.M. Khoshgoftaar, D. Wang, A survey of transfer learning, J. Big Data 3 (1) (2016) 9.
[32] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, Analysis of representations for domain adaptation, Adv. Neural Inf. Process. Syst. 19 (2006).
[33] C. Goutte, E. Gaussier, A probabilistic interpretation of precision, recall and F-score, with implication for evaluation, in: European Conference on Information Retrieval, Springer, 2005, pp. 345–359.
[34] P.K. Agarwal, R. Sharathkumar, Streaming algorithms for extent problems in high dimensions, in: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2010, pp. 1481–1489.
[35] M. Badoiu, K.L. Clarkson, Smaller core-sets for balls, in: Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2003, pp. 801–802.
[36] M. Bădoiu, K.L. Clarkson, Optimal core-sets for balls, Comput. Geomet. 40 (1) (2008) 14–22.
[37] A. Munteanu, C. Sohler, D. Feldman, Smallest enclosing ball for probabilistic data, in: Proceedings of the Thirtieth Annual Symposium on Computational Geometry, ACM, 2014, p. 214.
[38] P.K. Agarwal, S. Har-Peled, K.R. Varadarajan, Approximating extent measures of points, J. ACM (JACM) 51 (4) (2004) 606–635.
[39] T.M. Chan, Faster core-set constructions and data-stream algorithms in fixed dimensions, Comput. Geomet. 35 (1–2) (2006) 20–35.
[40] S. Har-Peled, S. Mazumdar, On coresets for k-means and k-medSian clustering, in: Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, ACM, 2004, pp. 291–300.
[41] Y. Zhang, K. Tangwongsan, S. Tirthapura, Streaming k-means clustering with fast queries, in: Data Engineering (ICDE), 2017 IEEE 33Rd International Conference on, IEEE, 2017, pp. 449–460.
[42] M.R. Ackermann, M. Märtens, C. Raupach, K. Swierkot, C. Lammersen, C. Sohler, StreamKM++: a clustering algorithm for data streams, J. Exper. Algorith. (JEA) 17 (2012) 2–4.
[43] T. Campbell, General bounds on the quality of Bayesian coresets, Adv. Neural Inf. Process. Syst. 37 (2024) 126946–126976.
[44] V. Braverman, D. Feldman, H. Lang, D. Rus, A. Statman, Least-Mean-Squares coresets for infinite streams, IEEE Trans. Knowl. Data Eng. 35 (9) (2023) 8699–8712.
[45] I. Jubran, E.E. Sanches Shayda, I.I. Newman, D. Feldman, Coresets for decision trees of signals, Adv. Neural Inf. Process. Syst. 34 (2021) 30352–30364.
[46] L. Huang, K. Sudhir, N. Vishnoi, Coresets for time series clustering, Adv. Neural Inf. Process. Syst. 34 (2021) 22849–22862.
[47] C. Baykal, L. Liebenwein, I. Gilitschenski, D. Feldman, D. Rus, Sensitivity-informed provable pruning of neural networks, SIAM J. Math. Data Sci. 4 (1) (2022) 26–45.
[48] X. Xia, J. Liu, S. Zhang, Q. Wu, H. Wei, T. Liu, Refined coreset selection: Towards minimal coreset size under model performance constraints, (2023). arXiv preprint arXiv:2311.08675
[49] X. Xia, J. Liu, J. Yu, X. Shen, B. Han, T. Liu, Moderate coreset: a universal method of data selection for real-world data-efficient deep learning, in: The Eleventh International Conference on Learning Representations, 2022.
[50] S. Yang, Z. Xie, H. Peng, M. Xu, M. Sun, P. Li, Dataset pruning: Reducing training data by examining generalization influence, (2022). arXiv preprint arXiv:2205.09329
[51] C. Gao, L. Shan, V. Srinivas, A. Vijayaraghavan, Computing High-dimensional Confidence Sets for Arbitrary Distributions, (2025). arXiv preprint arXiv:2504.02723
[52] D. Zhang, R. Tsuchida, D. Sejdinovic, Label Distribution Learning using the Squared Neural Family on the Probability Simplex, (2024). arXiv preprint arXiv:2412.07324