Persistent homology classification algorithm

Mark Lexter D. De Lara [Corresp. 1, 2]

[1] Institute of Mathematical Sciences and Physics, College of Arts and Sciences, University of the Philippines Los Baños, College, Los Baños, Laguna, Philippines

[2] Institute of Mathematics, University of the Philippines Diliman, Quezon City, Metro Manila, Philippines

Corresponding Author: Mark Lexter D. De Lara
Email address: mddelara@up.edu.ph

Data classification is an important aspect of machine learning, as it is utilized to solve issues in a wide variety of contexts. There are numerous classifiers, but there is no single best-performing classifier for all types of data, as the no free lunch theorem implies. Topological data analysis is an emerging topic concerned with the shape of data. One of the key tools in this field for analyzing the shape or topological properties of a dataset is persistent homology, an algebraic topology-based method for estimating the topological features of a space of points that persists across several resolutions.

This study proposes a supervised learning classification algorithm that makes use of persistent homology between training data classes in the form of persistence diagrams to predict the output category of new observations. Validation of the developed algorithm was performed on real-world and synthetic datasets. The performance of the proposed classification algorithm on these datasets was compared to that of the most widely used classifiers. Validation runs demonstrated that the proposed persistent homology classification algorithm performed at par if not better than the majority of classifiers considered.

Persistent Homology Classification Algorithm

Mark Lexter D. De Lara[1]

[1]**University of the Philippines Los Baños**

Corresponding author:
Mark Lexter D. De Lara[1]

Email address: mddelara@up.edu.ph

ABSTRACT

Data classification is an important aspect of machine learning, as it is utilized to solve issues in a wide variety of contexts. There are numerous classifiers, but there is no single best-performing classifier for all types of data, as the no-free lunch theorem implies. Topological data analysis is an emerging topic concerned with the shape of data. One of the key tools in this field for analyzing the shape or topological properties of a dataset is persistent homology, an algebraic topology-based method for estimating the topological features of a space of points that persists across several resolutions.

This study proposes a supervised learning classification algorithm that makes use of persistent homology between training data classes in the form of persistence diagrams to predict the output category of new observations. Validation of the developed algorithm was performed on real-world and synthetic datasets. The performance of the proposed classification algorithm on these datasets was compared to that of the most widely used classifiers. Validation runs demonstrated that the proposed persistent homology classification algorithm performed at par if not better than the majority of classifiers considered.

INTRODUCTION

Machine learning is an important branch of artificial intelligence that deals with the study of computer systems and algorithms that can learn and develop on their own without being explicitly programmed to do so. It is focused on the development of computer programs capable of performing data processing and predictive analysis. The three major categories of machine learning techniques are supervised, unsupervised, and reinforcement learning. Supervised learning is a process in which a system learns from a readily available training dataset of accurately classified observations. One of the major tasks addressed by supervised learning is classification.

Classification Algorithms

Classification is the process of identifying, recognizing, grouping, and understanding new observations into categories or classes (Alpaydin, 2014). A training dataset is composed of $(n+1)$-dimensional data points which are split into an n-dimensional input vector often called attributes and into a one-dimensional output category or class. A dataset can be univariate, bivariate, or multivariate in nature, whereas the attributes are quantifiable properties that can be categorical, ordinal, integer-valued, or real-valued. A classification algorithm or classifier is a procedure that performs classification tasks. The term classifier may also refer to the mathematical function that maps input attributes to an output category.

Classification algorithms have found a wide range of applications in domains such as computer vision, speech recognition, biometric identification, biological classification, pattern recognition, document classification, and credit scoring (Abiodun et al., 2018). Classification problems can be categorized as binary or multi-class. Binary classification entails allocating an observation to one of two distinct categories, whereas multi-class classification involves assigning an observation to one of more than two distinct classes.

Numerous classication algorithms have been developed since the rise of artificial intelligence. While several of these classifiers are specifically developed to solve binary classification problems, some

algorithms can be used to solve binary and multi-class classification problems. Many of these multi-class classifiers are extensions or combinations of one or more binary classifiers.

The no-free lunch theorem proved by David Wolpert and William Macready implies that no learning or optimization algorithm works best on all given problems (Wolpert, 1996; Wolpert and Macready, 1997). A classifier can be chosen depending on the type of data at hand. Since then, many state-of-the-art classifiers have been developed. Some of the most commonly used classifiers are logistic regression, Naive Bayes classifier, linear discriminant analysis, support vector machines, k-nearest neighbor, decision trees, and neural networks.

A classification algorithm is a linear classifier if it uses a linear function or linear predictor that assigns a score to each category k based on the dot product of a weight vector and a feature vector. The linear predictor is given by the score function, $Score(X_i, k) = \beta_k X_i$, where X_i is the feature vector for the observation i, and β_k is the weight vector corresponding to category k. Observation i is mapped by the linear predictor to the category k with the highest score function $\beta_k X_i$. Examples of linear classifiers include logistic regression, support vector machines, and linear discriminant analysis (Yuan et al., 2012).

Topological Data Analysis (TDA)

Data scientists employ techniques and theories drawn from various fields of mathematics, particularly algebraic topology, statistics, information science, and computer science. In mathematics, there is a growing field called topological data analysis, which is an approach that uses tools and techniques from topology to analyze datasets (Chazal and Michel, 2021). In the past two decades, TDA has been applied in various areas of science, engineering, medicine, astronomy, image processing, and biophysics.

Analyzing the shape of data is one of the motivations for using TDA. One of the key tools used in TDA is persistent homology (PH), a method based on algebraic topology that can be used for determining invariant features or topological properties of a space of points that persist across multiple resolutions (Edelsbrunner and Harer, 2008; Carlsson, 2009; Edelsbrunner and Harer, 2010). These derived invariant features are sensitive to small changes in the input parameters which make PH attractive to researchers studying qualitative features of data. PH involves representing a point cloud by a filtered sequence of nested complexes, which are turned into novel representations like barcodes and then interpreted statistically and qualitatively based on persistent topological features obtained (Otter et al., 2017). The following section gives a brief description of how to compute PH and a more detailed discussion of pertinent information about the homology of simplicial complexes and the complete process of computing PH of a point cloud can be found in the Appendix.

Computing Persistent Homology

Given a finite metric space S, such as a point cloud or a collection of experimental data, it is assumed that S represents a sample from an underlying topological space, say space T. Describing the topology of T based only on a finite sample S is not a straightforward task. However, PH can be utilized to determine the general shape and topological properties of T while maintaining the robustness of the data.

A point cloud can undergo a filtration process that turns it into a sequence of nested simplicial complexes. This is accomplished by considering a finite number of increasing parameters and recording the sublevel sets that track changes in the topological features of the complexes. More specifically, gradually thickening the points give birth to new topological features, like simplices and n-dimensional holes (Zomorodian and Carlsson, 2005). Figure 1 shows an illustration of the filtration process. A more formal and extensive discussion of how to compute the persistent homology of a point cloud is presented in the Appendix.

The appearance (birth) and disappearance (death) of intrinsic topological features, like homology groups and Betti numbers, can be recorded and visualized in many ways. The most frequently used are persistence barcodes or persistence diagrams. The duration or life span of the persisting topological features is essential in analyzing the qualitative and topological properties of a dataset.

Definition 1. *Let K be a finite simplicial complex and $K_1 \subseteq K_2 \subseteq \ldots \subseteq K_r = K$ be a finite sequence of nested subcomplexes of K. K is called a filtered simplicial complex and the sequence $\{K_1, K_2, \ldots\}$ is called a filtration of K.*

The homology of each of the subcomplexes can be computed. For each p, the inclusion maps $K_i \rightarrow K_j$ induce $\mathbb{F}_2$-linear maps $\partial_i^j : H_p(K_i) \rightarrow H_p(K_j)$ for all $i, j \in \{1, 2, \ldots, r\}$ with $i \leq j$. It follows from functoriality that $\partial_k^j \circ \partial_i^k = \partial_i^j$ for all $i \leq k \leq j$.
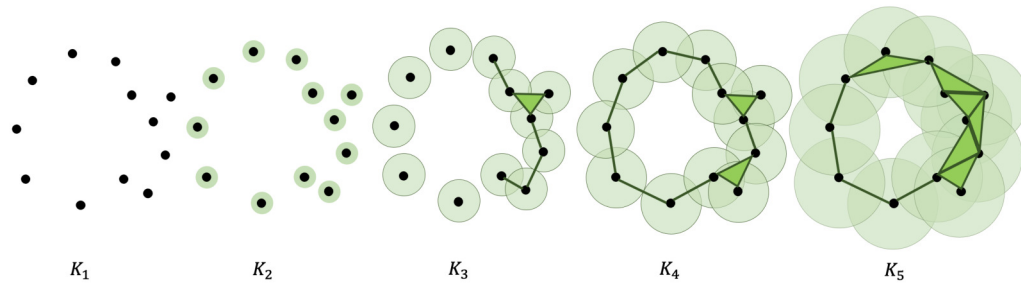
Figure 1. Filtration of a point cloud into a nested sequence of simplicial complexes, with $K_1 \subseteq K_2 \subseteq ... \subseteq K_5$.

Definition 2. *Let K_s be a subcomplex in the filtration of the simplicial complex K, or K_s be the filtered complex at time s, and $Z_k^s = Ker\partial_k^s$ and $B_k^s = Im\partial_{k+1}^s$ be the k-th cycle group and boundary group of K_s, respectively. The k-th homology group of K_s is $H_k^s = Z_k^s/B_k^s = Ker(\partial_k^s)/Im(\partial_{k+1}^s)$.*

Definition 3. *For $p \in \{0,1,2,...\}$, the p-persistent k-th homology group of K given a subcomplex K_s is*

$$H_k^{s,p}(K,K_s) = H_k^{s,p}(K) = Z_k^s/(B_k^{s+p}\bigcap Z_k^s) = \frac{Ker(\partial_k^s)}{Im(\partial_{k+1}^{s+p})\bigcup Ker(\partial_k^s)}.$$

The p-th persistent k-th Betti number $\beta_k^{s,p}$ of K_s is the rank of $H_k^{s,p}(K)$.

Persistent Vector Spaces and Persistence Diagrams

Definition 4. *A **persistent vector space** Carlsson (2014) is defined to be a family of vector spaces with linear maps between them, denoted by $\left\{V_\delta \xrightarrow{v_{\delta,\delta'}} V_{\delta'}\right\}_{\delta \leq \delta' \in \mathbb{R}}$, such that*

(1) $dim(V_\delta) < \infty$ for each $\delta \in \mathbb{R}$,

(2) there exist $\delta_I, \delta_F \in \mathbb{R}$ such that all maps $v_{\delta,\delta'}$ are linear isomorphisms for $\delta, \delta' \geq \delta_F$ and for $\delta, \delta' \leq \delta_I$, and

(3) there are only finitely many values of $\delta \in \mathbb{R}$ such that $v_{\delta-\varepsilon} \not\cong v_\delta$

(4) for any $\delta < \delta' < \delta'' \in \mathbb{R}$, we have $v_{\delta',\delta''} \circ v_{\delta,\delta'} = v_{\delta,\delta''}$.

Note that there are only finitely many values of $\delta \in \mathbb{R}$ for which the linear maps are not isomorphisms. This implies that the indexing set $\mathbb{R}$ can be reduced to $\mathbb{N}$, and we may equivalently define a persistent vector space to be an $\mathbb{N}$-induced family $\left\{V_{\delta_i} \xrightarrow{v_{i,i+1}} V_{i+1}\right\}_{i \in \mathbb{N}}$ such that

(1) $dim(V_{\delta_i}) < \infty$ for each $i \in \mathbb{N}$ and

(2) there exists $F \in \mathbb{N}$ such that all the $v_{i,i+1}$ are isomorphisms for $i \geq F$.

A persistent vector space can be obtained from a filtered simplicial complex and for each persistent vector space, one may associate a multiset of intervals called a persistence barcode or persistence diagram. A barcode represents each persistent generator with a horizontal line beginning at the first filtration level where it appears and ending at the filtration level where it disappears. Short bars correspond to noise or artifacts in the data, while lengthy bars correspond to relevant topological features. The standard treatment of persistence barcodes and diagrams appears in Edelsbrunner et al. (2002) and Zomorodian and Carlsson (2005).

Presented here is a modern presentation of the definition of persistence diagrams and persistence barcodes Chowdhury and Mémoli (2018) and Edelsbrunner et al. (2015).

Let $\mathcal{V} = \left\{ V_{\delta_i} \xrightarrow{v_{i,i+1}} V_{i+1} \right\}_{i \in \mathbb{N}}$ be a persistent vector space. Compatible bases $(B_i)_{i \in \mathbb{N}}$ for each V_{δ_i}, $i \in \mathbb{N}$ can be chosen such that $v_{i,i+1}|_{B_i}$ is injective for each $i \in \mathbb{N}$, and $rank(v_{i,i+1}) = card\big(im((v_{i,i+1}|_{B_i}) \cap B_{i+1})\big)$, for each $i \in \mathbb{N}$ Edelsbrunner et al. (2015). Note that $v_{i,i+1}|_{B_i}$ denotes the restriction of $v_{i,i+1}$ to the set B_i. Fix such a collection $(B_i)_{i \in \mathbb{N}}$ of bases. Then, define

$$L := \big\{ (b,i)|b \in B_i, \ b \notin Im(v_{i-1,i}), \ i \in \{2,3,\ldots\} \big\} \cup \big\{ (b,1)|b \in B_1 \big\}.$$

For each $(b,i) \in L$, the integer i is interpreted as the birth index of the basis element b or the first index at which a certain algebraic signal appears in the persistent vector space. Now, define a map $l : L \to \mathbb{N}$ as follows

$$\ell(b,i) := max\big\{ k \in \mathbb{N}|v_{k-1,k} \circ \cdots \circ v_{i+1,i+2} \circ v_{i,i+1}(b) \in B_k \big\}.$$

For each $(b,i) \in L$, the integer $\ell(b,i)$ is called the death index or the index at which the signal b disappears from the persistent vector space.

Definition 5. *The **persistence barcode** of $\mathcal{V}$ is defined to be the following multisets of intervals:*

$$\textbf{Pers}(\mathcal{V}) := \big[[\delta_i, \delta_{j+1})| \text{ there exists } (b,i) \in L \text{ such that } \ell(b,i) = j \big],$$

where the bracket notation denotes taking the multiset and the multiplicity of $[\delta_i, \delta_{j+1}]$ is the number of elements $(b,i) \in L$ such that $\ell(b,i) = j$.

The elements of $\textbf{Pers}(\mathcal{V})$ are called persistence intervals. They can be represented as a set of line segments over a single axis. Equivalently, $\textbf{Pers}(\mathcal{V})$ can be visualized as a multiset of points lying on or above the diagonal in $\overline{\mathbb{R}}^2$ counted with multiplicity, where $\overline{\mathbb{R}}$ denotes $[-\infty, +\infty]$.

Definition 6. *The **persistence diagram** of $\mathcal{V}$ is given by*

$$Dgm(\mathcal{V}) := \big[(\delta_i, \delta_{j+1}) \in \overline{\mathbb{R}}^2 | [\delta_i, \delta_{j+1}) \in \textbf{Pers}(\mathcal{V}) \big],$$

where multiplicity of $(\delta_i, \delta_{j+1}) \in \overline{\mathbb{R}}^2$ is given by the multiplicity of $[\delta_i, \delta_{j+1}) \in \textbf{Pers}(\mathcal{V})$.

Discussion of the robustness and stability of the persistence diagram requires the notion of distance. Given two persistence diagrams, say X and Y, the definition of distance between X and Y is given as follows.

Definition 7. *Let $p \in [1, \infty]$. The p-th Wasserstein distance between X and Y is defined as*

$$W_p[d](X,Y) := \inf_{\phi:X \to Y} \left[\sum_{x \in X} d[x, \phi(x)]^p \right]^{1/p}$$

for $p \in [1, \infty)$ and as

$$W_\infty[d](X,Y) := \inf_{\phi:X \to Y} \sup_{x \in X} d[x, \phi(x)]$$

for $p = \infty$, where d is a metric on $\bar{\mathbb{R}}^2$ and ϕ ranges over all bijections from X to Y.

Normally, d is taken to be L_q where $q \in [1, \infty]$ and the most commonly used distance function is the Bottleneck distance $W_\infty[L_\infty]$. Equivalently, the bottle distance between persistence diagrams can be defined as follows.

Definition 8. *The **bottleneck distance** between persistent diagrams and more generally, between multisets C, D of points in $\overline{\mathbb{R}}^2$ is given by*

$$d_B(C,D) := \inf \big\{ \sup_{a \in C} \|a - \phi(a)\| | \phi : C \cup \Delta^\infty \to D \cup \Delta^\infty \text{ is a bijection.} \big\},$$

where $\|(p,q) - (p',q')\|_\infty := max(|p - p'|, |q - q'|)$ for each p, q, p', and $q' \in \mathbb{R}$, and Δ^∞ is the multiset consisting of each point on the diagonal, taken with infinite multiplicity.

Basis for the classification algorithm

Let I be an index set with K elements, where $K \in \mathbb{N}$. Suppose each $k \in I$ corresponds to a class of points in $\mathbb{R}^n$. Let X_k be the set of n-dimensional points in the Euclidean space belonging to class k. Now, suppose there is an n-dimensional point $\alpha^p = (\alpha_1^p, \alpha_2^p, \ldots, \alpha_n^p) \in \mathbb{R}^n$. The problem of determining which of the K classes of points α^p belongs to is a classification problem. Note that the collection of points belonging to the same class has its own topological properties. It is a natural intuition that the inclusion of an additional point α^p in a point cloud where it belongs should result in a very small change in its topological properties. What follows is a way to analyze a point cloud and its properties.

Given a point cloud, we can compute its corresponding topological properties in the form of a persistence barcode or a persistence diagram. To do this, we follow the construction presented in Section . From here on, this is referred to as the **long construction**.

For each point cloud X_k, $k \in I$, applying the **long construction** is the same as defining the following;

(i) the Vietoris Rips complex filtration given by $X_1^k \subseteq X_2^k \subseteq \cdots \subseteq X_{m+1}^k$ corresponding to the X_k with fixed $m \in \mathbb{N}$, $\varepsilon > 0$, and the parameter values ε_j's with $\varepsilon_1 < \varepsilon_2 < \cdots < \varepsilon_{m+1}$ and $\varepsilon_j = \frac{(j-1)\varepsilon}{m}$, for each $j \in \{1, 2, \ldots, m+1\}$,

(ii) an $\mathbb{N}$-indexed persistent vector space $\mathscr{X}^k = \left\{ X_i^k \xrightarrow{x_{i,i+1}^k} X_{i+1}^k \right\}_{i \in \mathbb{N}}$ such that

    (1) $dim(X_i^k) < \infty$ for each $i \in \mathbb{N}$ and

    (2) there exists $m \in \mathbb{N}$ such that all the $x_{i,i+1}$ are isomorphisms for $i \geq m$.

(iii) chosen compatible bases $(B_i^k)_{i \in \mathbb{N}}$ for each respective class k or X_i^k, $k \in I$ such that $x_{i,i+1}^k|_{B_i^k}$ denote the restriction of $x_{i,i+1}^k$ to the set B_i^k and $rank(x_{i,i+1}^k) = card\left(im((x_{i,i+1}|_{B_i^k}) \cap B_{i+1})\right)$,

(iv) the set given by $L_k := \left\{ (b,i) | b \in B_i^k, \ b \notin im(x_{i-1,i}^k), \ i \in \{2, 3, \ldots\} \right\} \cup \left\{ (b,1) | b \in B_1^k \right\}$,

(v) the map defined by $\ell_k(b,i) := max\left\{ r \in \mathbb{N} | x_{r-1,r} \circ \cdots \circ x_{i+1,i+2} \circ x_{i,i+1}(b) \in B_r^k \right\}$,

(vi) the persistence barcode given by $\mathbf{Pers}(\mathscr{X}^k) := \left[[\delta_i, \delta_{j+1}) | \text{ there exists } (b,i) \in L_k \text{ such that } \ell_k(b,i) = j \right]$, where the bracket notation denotes taking the multiset and the multiplicity of $[\delta_i, \delta_{j+1}]$ is the number of elements $(b,i) \in L_k$ such that $\ell_k(b,i) = j$, and

(vii) the persistence diagram given by $\mathbf{Dgm}(\mathscr{X}^k) := \left[(\delta_i, \delta_{j+1}) \in \overline{\mathbb{R}}^2 | [\delta_i, \delta_{j+1}) \in \mathbf{Pers}(\mathscr{V}) \right]$, where multiplicity of $(\delta_i, \delta_{j+1}) \in \overline{\mathbb{R}}^2$ is given by the multiplicity of $[\delta_i, \delta_{j+1}) \in \mathbf{Pers}(\mathscr{X}^k)$.

Now, we consider the scenario of adding the n-dimensional point $\alpha^p = (\alpha_1^p, \alpha_2^p, \ldots, \alpha_n^p) \in \mathbb{R}^n$ to the point clouds. Define $Y_k = X_k \cup \{\alpha^p\}$ for each $k \in I$. Apply the **long construction** for each of these point clouds, that is define the following;

(i) the Vietoris Rips complex filtration given by $Y_1^k \subseteq Y_2^k \subseteq \cdots \subseteq Y_{m+1}^k$ corresponding to the Y_k with fixed $m \in \mathbb{N}$, $\varepsilon > 0$, and the parameter values ε_j's where $\varepsilon_1 < \varepsilon_2 < \cdots < \varepsilon_{m+1}$ and $\varepsilon_j = \frac{(j-1)\varepsilon}{m}$ for each $j \in \{1, 2, \ldots, m+1\}$,

(ii) the $\mathbb{N}$-indexed persistent vector space $\mathscr{Y}^k = \left\{ Y_i^k \xrightarrow{y_{i,i+1}^k} Y_{i+1}^k \right\}_{i \in \mathbb{N}}$ similar to that of $\mathscr{X}^k$,

(iii) compatible bases $(\widetilde{B}_i^{\,k})_{i \in \mathbb{N}}$ similar to that of $(B_i^{\,k})_{i \in \mathbb{N}}$,

(iv) the set $\widetilde{L}_k$ similar to that of L_k,

(v) the map $\widetilde{\ell}_k$ similar to that of ℓ_k

(vi) the persistence barcode or multiset given by $\mathbf{Pers}(\mathscr{Y}^k)$, and

(vii) the persistence diagram given by $Dgm(\mathscr{Y}^k)$.

192 Finally, classification is done by choosing the class k for which the change in the topological properties
193 between X_k and Y_k is minimum. This is inspired by the following observations.

Theorem 1. *Let I be a finite index set. Suppose that for each $k \in I$, X_k is the set of points in $\mathbb{R}^n$ belonging to class k, and that $X_i \cap X_j = \emptyset$ for each $i \neq j$. Assuming $\alpha^p = (\alpha_1^p, \alpha_2^p, \ldots, \alpha_n^p) \in \mathbb{R}^n$ is a point identical to one of the points in X_p, for an element $p \in I$. Then, for each $k \in I$,*

$$d_B\big(Dgm(\mathscr{X}^p), Dgm(\mathscr{Y}^p)\big) \leq d_B\big(Dgm(\mathscr{X}^k), Dgm(\mathscr{Y}^k)\big),$$

194 *where $\mathscr{X}^k$, and $\mathscr{Y}^k$ are the $\mathbb{N}$-indexed persistent vector spaces corresponding X_k and $X_k \cup \{\alpha^p\}$, respec-*
195 *tively, for each $k \in I$.*

196 *Proof.* Let $\alpha^p \in X_p$ be an n-dimensional point where $p \in I$. Choose any element $k \in I$ such that $k \neq p$.
197 Then, $X_k \cap X_p = \emptyset$. Apply the **long construction** for X_k, $Y_k = X_k \cup \{\alpha^p\}$, X_p, and $Y_p = X_p \cup \{\alpha^p\}$. From
198 the long constructions leading to the computation of persistence diagrams, we can define the following

199 (i) the Vietoris Rips complex filtrations associated to X_k, Y_k, X_p and Y_p, with respect to some $m \in \mathbb{N}$,
200     $\varepsilon > 0$, and the parameter values ε_j's with $\varepsilon_1 < \varepsilon_2 < \cdots < \varepsilon_{m+1}$ and $\varepsilon_j = \frac{(j-1)\varepsilon}{m}$ for each $j \in$
201     $\{1, 2, \ldots, m+1\}$,

202 (ii) the $\mathbb{N}$-indexed persistent vector spaces $\mathscr{X}^k$, $\mathscr{Y}^k$, $\mathscr{X}^p$, and $\mathscr{X}^p$,

203 (iii) respective compatible bases $(B_i{}^k)_{i \in \mathbb{N}}$, $(B_i{}^p)_{i \in \mathbb{N}}$, $(\widetilde{B}_i{}^k)_{i \in \mathbb{N}}$, and $(\widetilde{B}_i{}^p)_{i \in \mathbb{N}}$,

204 (iv) the sets L_k, L_p, $\widetilde{L}_k$, and $\widetilde{L}_p$,

205 (v) the maps ℓ_k, ℓ_p, $\widetilde{\ell}_k$, and $\widetilde{\ell}_p$,

206 (vi) the persistence barcodes or multisets given by **Pers**$(\mathscr{X}^k)$, **Pers**$(\mathscr{Y}^k)$, **Pers**$(\mathscr{X}^p)$, and **Pers**$(\mathscr{Y}^p)$,
207   and

208 (vii) the persistence diagrams given by $Dgm(\mathscr{X}^k)$, $Dgm(\mathscr{Y}^k)$, $Dgm(\mathscr{X}^p)$, and $Dgm(\mathscr{Y}^p)$.

209 Since $\alpha^p \in X_p$, then $\alpha^p \notin X_k$, $X_p = Y_p$, and Y_k has one more element than X_k. Also, $\alpha^p \in \widetilde{B}_1{}^k$ and
210 $\alpha^p \notin B_1{}^k$. Note that there is at least one $(\alpha^p, i) \in \widetilde{L}_k$ with i at least the first index 1. Then $\widetilde{\ell}_k(\alpha^p, i) \geq 1$,
211 by the definition of $\widetilde{\ell}_k : \widetilde{L}_k \to \mathbb{N}$. That is, the death index corresponding (α^p, i) must be greater than 1.
212 This implies that **Pers**$(\mathscr{Y}^k)$ will have one more element than **Pers**$(\mathscr{X}^k)$. This means that **Pers**$(\mathscr{Y}^k)$
213 has one unmatched point with respect to the pairing with **Pers**$(\mathscr{X}^k)$. Then, the computation of
214 $d_B\big(Dgm(\mathscr{Y}^k), Dgm(\mathscr{X}^k)\big)$ boils down to the optimal distance between the matched points or the distance
215 between the unmatched point to the diagonal. In either case, this is greater than zero.
216 Now, since $X_p = Y_p$, then $d_B\big(Dgm(\mathscr{Y}^p), Dgm(\mathscr{X}^p)\big) = 0$.
217 Hence, for each $k \in I$, $d_B\big(Dgm(\mathscr{X}^k), Dgm(\mathscr{Y}^k)\big) \geq d_B\big(Dgm(\mathscr{X}^p), Dgm(\mathscr{Y}^p)\big) = 0$.
218     $\square$

Theorem 2. *Let I be a finite index set. Suppose that for each $k \in I$, X_k is the set of points in $\mathbb{R}^n$ belonging to class k, and that $X_i \cap X_j = \emptyset$ for each $i \neq j$. Assume $\alpha^p \in \mathbb{R}^n$ is the new data point and $\alpha^p \notin X_k$ for each $k \in I$. Let α^{q^*} be a point in X_{q^*}, where $q^* \in I$, such that α^{q^*} is the closest point to α^p among all points in $\bigcup_{k=1}^{K} X_k$. If the **long construction** is applied to X_{q^*} and $Y_{q^*} = X_{q^*} \cup \{\alpha^p\}$, and to X_k and $Y_k = X_k \cup \{\alpha^p\}$, for each $k \in I$, $k \neq q^*$, and the sets L_k, L_{q^*}, $\widetilde{L}_k$, and $\widetilde{L}_{q^*}$ are limited to elements with birth index of 1, then for each $k \in I$,*

$$d_B\big(Dgm(\mathscr{X}^{q^*}), Dgm(\mathscr{Y}^{q^*})\big) \leq d_B\big(Dgm(\mathscr{X}^k), Dgm(\mathscr{Y}^k)\big),$$

219 *where $\mathscr{X}^k$ and $\mathscr{Y}^k$ are the $\mathbb{N}$-indexed persistent vector spaces corresponding to X_k and $X_k \cup \{\alpha^p\}$,*
220 *respectively, for each $k \in I$.*

221 *Proof.* Let I be a finite index set. Suppose that for each $k \in I$, X_k is the set of points in $\mathbb{R}^n$ belonging to
222 class k, and that $X_i \cap X_j = \emptyset$ for each $i \neq j$. Assume $\alpha^p \in \mathbb{R}^n$ is the new point and $\alpha^p \notin X_k$ for each $k \in I$.
223 For each $k \in I$, let α_k^q be the element in X_k which is closest to α^p. Suppose that α^{q^*} is the closest element
224 to α^p among any α_k^q, where $k \in I$.

225   Without loss of generality, choose any element $k \in I$ such that $k \neq q^*$. Then, $X_k \cap X_{q^*} = \emptyset$. Apply
226 the **long construction** for X_k, $Y_k = X_k \cup \{\alpha^p\}$, X_{q^*} and $Y_{q^*} = X_{q^*} \cup \{\alpha^p\}$. From the long constructions
227 leading to the computation of persistence diagrams, we define the following;

(i) the Vietoris Rips complex filtrations associated to X_k, Y_k, X_{q^*} and Y_{q^*}, with respect to some $m \in \mathbb{N}$,
228
229      $\varepsilon > 0$, and the parameter values ε_j's with $\varepsilon_1 < \varepsilon_2 < \cdots < \varepsilon_{m+1}$ and $\varepsilon_j = \frac{(j-1)\varepsilon}{m}$, for each $j \in$
230      $\{1, 2, \ldots, m+1\}$,

231 (ii) the $\mathbb{N}$-indexed persistent vector spaces $\mathscr{X}^k$, $\mathscr{Y}^k$, $\mathscr{X}^{q^*}$, and $\mathscr{X}^{q^*}$,

232 (iii) respective compatible bases $(B_i^k)_{i \in \mathbb{N}}$, $(B_i^{q^*})_{i \in \mathbb{N}}$, $(\widetilde{B}_i^{\,k})_{i \in \mathbb{N}}$, and $(\widetilde{B}_i^{\,q^*})_{i \in \mathbb{N}}$,

233 (iv) the sets L_k, L_{q^*}, $\widetilde{L}_k$, and $\widetilde{L}_{q^*}$ whose elements are restricted to those with birth index of 1,

234 (v) the maps ℓ_k, ℓ_{q^*}, $\widetilde{\ell}_k$, and $\widetilde{\ell}_{q^*}$,

235 (vi) the persistence barcodes or multisets given by $\mathbf{Pers}(\mathscr{X}^k)$, $\mathbf{Pers}(\mathscr{Y}^k)$, $\mathbf{Pers}(\mathscr{X}^{q^*})$, and $\mathbf{Pers}(\mathscr{Y}^{q^*})$,
236 and

237 (vii) the persistence diagrams given by $Dgm(\mathscr{X}^k)$, $Dgm(\mathscr{Y}^k)$, $Dgm(\mathscr{X}^{q^*})$, and $Dgm(\mathscr{Y}^{q^*})$.

238   For each $k \in I$, define $\alpha_k^q \in \mathbb{R}^n$ to be the point in X_k which is closest to α^p. Note that limiting the
239 basis elements to those with birth index of 1 means that only the 0-dimensional topological properties
240 or the connectedness of the simplices are considered. Also, for each $k \in I$, X_k and Y_k differ only by a
241 single point, the point α^p. By these, the 0-dimensional hole corresponding to α^p in the filtration of
242 Y_k disappears faster if α_k^q is closer to α^p. Moreover the computation of $d_B\big(Dgm(\mathscr{X}^k), Dgm(\mathscr{Y}^k)\big)$ is
243 reduced to the computation of $\widetilde{\ell}_k(\alpha^p, 1) = \frac{1}{2}\|\alpha^P - \alpha_k^q\|$ for each $k \in I$. Since α^{q^*} is closest to α^p among
244 all the α_k^q's, then this implies that $\widetilde{\ell}_{q^*}(\alpha^{q^*}, 1) \leq \widetilde{\ell}_k(\alpha_k^q, 1)$, for any $k \in I$, where $(\alpha^{q^*}, 1) \in (\widetilde{B}_i^{\,q^*})_{i \in \mathbb{N}}$ and
245 $(\alpha_k^q, 1) \in (\widetilde{B}_i^{\,k})_{i \in \mathbb{N}}$. This completes the proof. $\square$
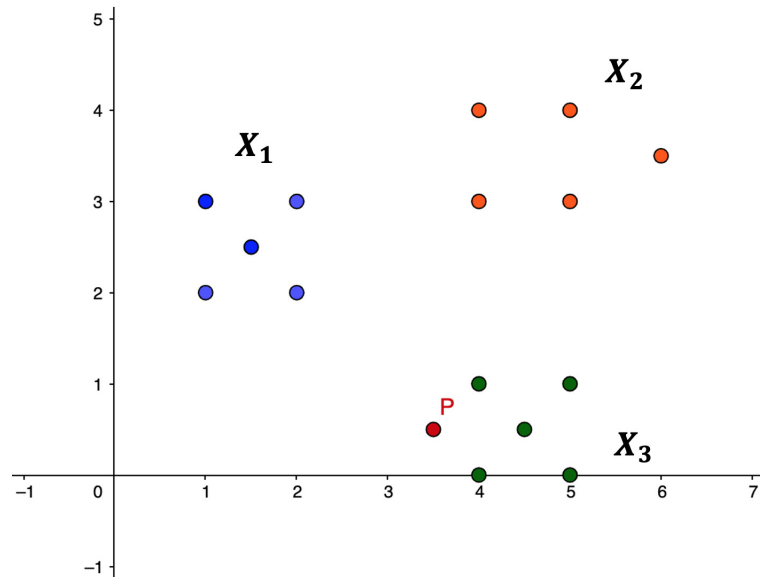
Figure 2. Point clouds X_1, X_2, and X_3, and point P.

246 **Illustration 1.** *Consider the point clouds labelled as X_1, X_2, and X_3, and the point P in the 2-dimensional*
247 *space as presented in Fig 2. Suppose that the point P has to be categorized into one of the point clouds.*
248   *For each $i \in \{1, 2, 3\}$, define $Y_i = X_i \cup \{P\}$ and compute the persistent homologies of X_i and Y_i.*
249 *Presented in Fig 3 are the respective persistence barcodes corresponding to the 0-dimensional holes in*
250 *the filtration of the X_i's and Y_i's. Note that each pair of X_i and Y_i differ only by a single point, the point*
251 *P. Since only the 0-dimensional holes are considered here, then the persistence barcode for Y_i has one*

more bar than the persistence barcode for X_i, for each $i \in \{1,2,3\}$. These differences, represented by the green bars in Fig 3, are indicative of the shortest distance of P from the point clouds. The life span of the 0-dimensional hole corresponding to point P in the filtration of Y_i is the shortest when X_i is the closest point cloud to P. For each $k \in \{1,2,3\}$,

$$d_B\big(Dgm(\mathscr{X}^3), Dgm(\mathscr{Y}^3)\big) \leq \widetilde{\ell}_k(P,1) = \frac{1}{2}\|P - \alpha_k^q\|,$$

where α_k^q is the point in X_k that is closest to P.
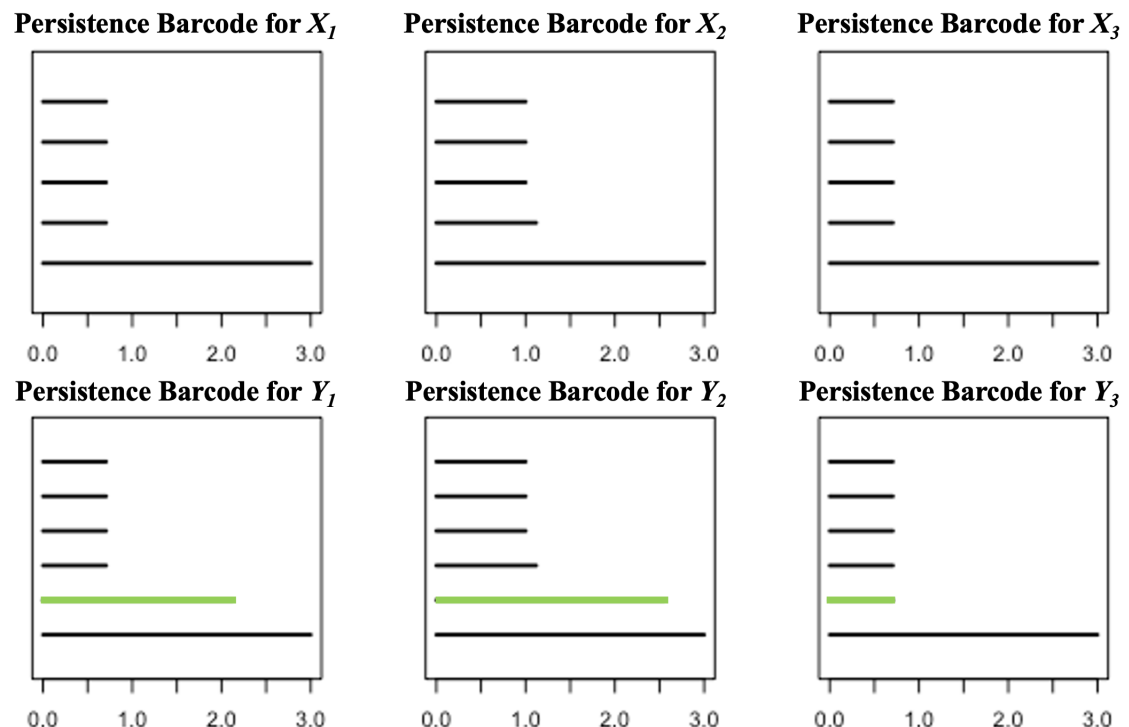
Figure 3

Advances in the fusion of persistent homology and machine learning

Computation of persistent homology (PH) has been applied to a variety of fields, including image analysis, pattern comparison and recognition, network analysis, computer vision, computational biology, oncology, and chemical structures. Some examples can be found in the works of Charytanowicz et al. (2010), Goldenberg et al. (2010), Nicolau et al. (2011), Xia and Wei (2014), Giansiracusa et al. (2019), and Ignacio and Darcy (2019). Furthermore, advances in the different aspects of computing PH have been increasing at a very rapid rate. Various software have also been developed for computing persistent homology. These software packages include JavaPlex, Perseus, Dipha, Dionysus, jHoles, GUDHI (Geometry Understanding in Higher Dimensions), Rivet, Ripser, PHAT (Persistent Homology Algorithm Toolkit), and R-TDA (R package for Topological Data Analysis) (Otter et al., 2017; Pun et al., 2018). On the other hand, machine learning has been in development as early as 1950s, but it was not until the 1990s that a shift from a knowledge-driven approach to a data-driven approach in studying machine learning took place. It was also during this time when support vector machines and neural networks became very popular. Persistent homology, which has only been around for a decade, has also received attention in recent years.

A direct exposition of the use of machine learning and persistence barcodes was used by Giansiracusa et al. (2019) in solving a fingerprint classification problem. They also showed that better accuracy rates can be achieved when applying topological data analysis to 3-dimensional point clouds of oriented minutiae points. Yu-Min Chung and Lawson (2020) used persistence curves, rather than barcodes or

diagrams, to analyze time series classification problems. Ismail et al. (2020) used PH-based machine learning algorithms to predict next day direction of stock price movement. Hofer et al. (2017) incorporated topological signatures to deep neural networks and performed classification experiments on 2D object shapes and social network graphs. Gonzalez-Diaz et al. (2020) used PH to define neural networks by incorporating the concept of representative datasets. In their study, they used persistence diagrams in choosing points that would be included in the representative dataset. Chen et al. (2019) leveraged topological information to regularize the topological complexity of kernel classifiers by incorporating a topological penalty, while Pokorny et al. (2014) used topological approaches to improve trajectory classification in robotics. Edwards et al. (2021) introduced TDAExplore, a machine learning image analysis pipeline based on topological data analysis. TDAExplore can be used to classify high-resolution images and characterize which image regions contribute to classification. PH has also been found useful in unsupervised learning and clustering tasks. Islambekov and Gel (2019) presented an unsupervised PH-based clustering algorithm for space-time data. They evaluated the performance of their proposed algorithm on synthetic data and compared it to other well-known clustering algorithms such as K-means, hierarchical clustering, and DBSCAN (Density-based spatial clustering of applications with noise).

Pun et al. (2018) published a survey of PH-based machine learning algorithms and their applications. They presented ways how to use persistent homologies to improve machine learning algorithms such as support vector machines, tree-based methods, and artificial neural networks. The strategies require extracting topological features from individual data points and adding them as new attributes to the data points. The classification algorithms are then modified in such a way that they can utilize the additional attributes to classify the data points more accurately. These strategies of modifying learning algorithms by incorporating topological signatures as additional features to the data points are very effective when the data under scrutiny contains high-dimensional points and when points belonging to the same class share common topological properties. This means that the improved algorithm can be used to group together data points with similar shapes or topological properties. On the other hand, these strategies may not work when the dataset under consideration contains data points in a Euclidean space, when each of the points in the dataset has very low dimensions, or if data points from the same class do not share the same topological properties.

On the contrary, the objective of this study was to develop a supervised classification algorithm for dealing with classification problems involving points in a Euclidean space, datasets that can be separated into classes using hyperplanes, or when points from the same class have a certain level of proximity with each other. The proposed classifier in this study takes substantial use of persistent homologies and topological signatures associated with the different classes of points in the dataset instead of the individual data points' persistent homologies and topological features.

PERSISTENT HOMOLOGY CLASSIFICATION ALGORITHM (PHCA)

A supervised classification algorithm or classifier is a technique that assigns a new object or observation to a class based on the training set and the new observation's attribute values. The classifier proposed in this work, termed persistent homology classification algorithm (PHCA), is largely reliant on the topological properties of each class of points in the available training set. The topological features derived from the persistence diagrams, generated by computing the persistent homology of each class in the training set, will be used to construct a linear classifier or a score function for classifying new observations. The section titled Computing Persistent Homology contains a discussion of the basis of this algorithm, while this section shows the outline of the proposed algorithm's implementation.

Interpretation of a point cloud's persistent homology

Let Z be a point cloud. Computing the persistent homology of Z means letting Z undergo filtration and recording the topological properties of Z that persist across multiple resolutions. The result of computing the persistent homology of Z or the summary of the topological properties of the space underlying Z can be visualized using persistence barcodes and persistence diagrams as shown in Definitions 5 and 6. See the Appendix for an illustration. Moreover, it can also be represented by an $n_i \times 3$ matrix and it will be denoted by $\mathscr{P}(Z)$ in this section. The number of rows of $\mathscr{P}(Z)$, n_i, is the number of topological features or n-dimensional holes detected in the filtration of Z. The 0-dimensional holes are the connected components, 1-dimensional holes refer to loops, 2-dimensional holes represent voids, and so on. One can limit the maximum dimension that can be detected in the filtration of Z. The first column entry of the i-th

329 row of $\mathscr{P}(Z)$ indicates the dimension of i-th topological feature in the filtration of Z. The second and
330 third column entries in the i-th row of $\mathscr{P}(Z)$ give the birth and death time of the i-th topological feature,
331 respectively. The birth and death times refer to the filtration steps when the topological feature appears
332 and disappears, respectively.
333 Let $\mathscr{P}(Z)$ be the persistent homology of a point cloud Z. Although it can be visualized as a persistence
334 diagram, it denotes an $n_i \times 3$ matrix, where n_i is the number of holes, of various dimensions, which are
335 detected in the complex filtration.

Setting of maximum parameter value

337 The maximum scales, denoted by *maxsc*, must be fixed. The scale here refers to the size of the parameter
338 values that will be used in the filtration and the computation of the persistent homology of a point cloud.
339 The suitable value for *maxsc* depends on the data at hand and it can be chosen to be equal to half of the
340 maximum distance between any two points in the point cloud.

Training and Classification

342 Assume that X is a training dataset consisting of $(n+1)$-dimensional data points categorized into k
343 classes. Each data point's initial n entries contain its attributes, while the $(n+1)$-th entry gives its label
344 or classification. Suppose that $X = X_1 \cup X_2 \cup \ldots \cup X_k$, where $X_1, \ldots, X_k$ are the k classes of data points,
345 and m_i is the number of elements in X_i. Each training point cloud X_i can be viewed as an $(m_i) \times (n+1)$
346 matrix with each row representing a point belonging in X_i.
347 Suppose that a new n-dimensional point, denoted by α, needs to be classified. To begin, analyze the
348 different classes in the training set or describe the topological properties of each class in X by computing
349 $\mathscr{P}(X_i)$ for each $i \in \{1, \ldots, k\}$. Then, measure the effect of including p in each class in X. Define
350 $Y_i = X_i \cup \{\alpha\}$ and compute $\mathscr{P}(Y_i)$ for each $i \in \{1, \ldots, k\}$. Record the changes in the persistent homology
351 between X_i and Y_i for each $i \in \{1, \ldots, k\}$. Specifically, record the change from $\mathscr{P}(X_i)$ to $\mathscr{P}(Y_i)$ for each
352 $i \in \{1, \ldots, k\}$. Lastly, identify the class that is least impacted by the inclusion of α. This procedure
353 follows the long construction defined in Section .

Score Function for PHCA

355 The use of persistent homology in the development of the score function that will act as the linear predictor
356 for PHCA is described here. Consider a training set categorized into two classes, A and B. Suppose α is a
357 point that needs to be classified and that α belongs to either A or B only, as shown in Fig. 4. The proposed
358 PHCA works by computing the persistent homology of A, B, $A \cup \{\alpha\}$, and $B \cup \{\alpha\}$. Afterward, measure
359 the changes in the topological features from A to $A \cup \{\alpha\}$ and from B to $B \cup \{\alpha\}$. Suppose that point α is
360 closer to point cloud A than point cloud B, then the change in the persistent homology of A to $A \cup \{\alpha\}$
361 will be lower than the change in the persistent homology from B to $B \cup \{\alpha\}$. This is because the closer a
362 point α to a point cloud, say Z, the birth, and death of some topological features of $Z \cup \{p\}$, particularly
363 the 0-dimensional holes, will come earlier. That is, the early appearance or disappearance of topological
364 features translates to a shorter life span of topological features in the filtration of the complexes. This
365 procedure which is based on Theorems 2 and 3 works particularly well when the point clouds for different
366 classes are disjoint from one another or when a point under consideration is close to a particular point
367 cloud. In light of these, the score function used for PHCA is computed in the following manner.
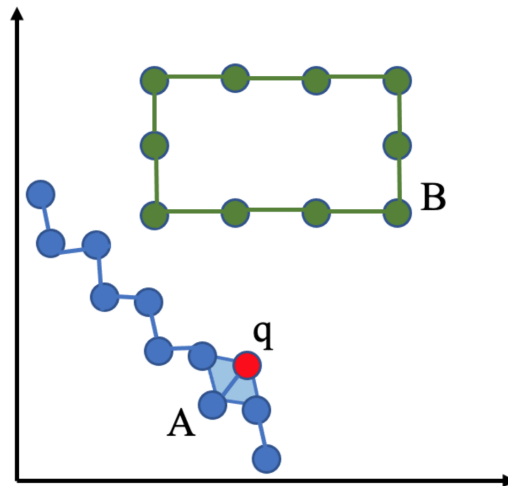
Figure 4. Training set of points categorized into class *A*(blue points) and class *B*(green points), and a new point *q*.

Recall that we have defined $Y_i = X_i \cup \{\alpha\}$ for each $i \in \{1,\ldots,k\}$, and that $\mathscr{P}(Y_i)$ is the persistent homology of Y_i for each $i \in \{1,\ldots,k\}$. The score function $Score(Y_i)$ is computed as the difference between the sum total of the lifespan of all the topological features in $\mathscr{P}(Y_i)$ and the sum total of the lifespan of all the topological features in $\mathscr{P}(X_i)$. This is equivalent to the difference between the sum of the entries of the third column of $\mathscr{P}(X_i)$ and the sum of the entries of the third column of $\mathscr{P}(Y_i)$. More explicitly, the score function value is computed as

$$Score(X_i) = \left| \sum_{a \in Y_i} \ell(a,1) - \sum_{b \in X_i} \ell(b,1) \right|, \tag{1}$$

where ℓ is the map that sends a basis element with birth index of 1 to its death index.

Finally, the new data point α is classified under class j if $Score(X_j) \le Score(X_i)$ for all $i \in \{1,\ldots,k\}$. Algorithm 1 gives the pseudocode for the persistent homology classification algorithm (PHCA).

Algorithm 1 Persistent Homology Classification Algorithm

Require: $X_1, X_2, \ldots X_k$, *maxd*, *maxsc*, and α
Ensure: *Class*(α)
  procedure
    for $i = 1$ to k **do**
      $\mathscr{P}(X_i) \leftarrow n_i \times 3$ *matrix, a result of computing PH of X_i*
      $\mathscr{P}(Y_i = X_i \cup \{\alpha\}) \leftarrow n_i \times 3$ *matrix, a result of computing PH of Y_i*
      Compute $Score(X_i)$
    end for
    $Class(\alpha) \leftarrow \arg\min_{\forall i}\{Score(X_i)\}$
  end procedure

Otter et al. Otter et al. (2017) established that computing the persistent homology of a point cloud using the standard algorithm has cubic complexity in terms of the number of simplices in the worst-case scenario. This, combined with the fact that PHCA is a linear classifier, means that the proposed classifier can be used to perform classification tasks in polynomial time.

EVALUATION METHODOLOGY

The validity of PHCA was determined by solving four classification tasks using three well-studied datasets and a synthetic dataset made of points from three separate geometric figures. The number of classes per dataset ranges from 2 to 10, while the number of attributes per dataset ranges from 2 to 2025. The performance of the proposed PHCA was quantified and then compared to the performance of five main classification algorithms on four validation datasets. Linear discriminant analysis (LDA), Classification and Regression Trees (CART), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Random Forest (RF) were the five classifiers used as benchmarks in this study.

All validation runs implemented in this study employed five-fold cross-validation. That is, the validation process involved dividing each class of the datasets into five sections. Each validation run requires a training set and a testing set. At any particular run, the testing set is composed of a partition from each class and the training set is composed of all the remaining data points. This guarantees that each class will have a representation in the testing. This scheme also assured that every single data point served as a testing point in one of the validation runs. To evaluate the methods, the confusion matrix was generated to determine the following metrics: precision, recall(sensitivity), specificity, accuracy, and F1 score. The confusion matrix gives the number of data points per class that is correctly predicted or incorrectly predicted.

For instance, consider a particular class, say C_i, among k classes. Then, we can define the following for each $i \in \{1, 2, \ldots, k\}$.

TP_i is the number of true positives in class C_i, or the number of instances in C_i which are predicted to belong in C_i.

TN_i is the number of true negatives in class C_i, or the number of instances outside C_i which are predicted to not belong in C_i.

FP_i is the number of false positives in class C_i, or the number of instances outside C_i which are predicted to belong in C_i.

FN_i is the number of false negatives in class C_i, or the number of instances in C_i which are predicted to not belong in C_i.

The five metrics per class C_i are computed as follows

$$\text{Precision for Class } C_i : Prec(C_i) = \frac{TP_i}{TP_i + FP_i} \tag{2}$$

$$\text{Recall for Class } C_i : Rec(C_i) = \frac{TP_i}{TP_i + FN_i} \tag{3}$$

$$\text{Specificity for Class } C_i : Spec(C_i) = \frac{TN_i}{TN_i + FP_i} \tag{4}$$

$$\text{Accuracy for Class } C_i : Acc(C_i) = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \tag{5}$$

$$\text{F1 score for Class } C_i : F1(C_i) = \frac{2 \times Prec(C_i) \times Rec(C_i)}{Prec(C_i) + Rec(C_i)} \tag{6}$$

A high sensitivity prediction in Class C_i implies that the reliability of predicting that an instance does not belong to C_i is high. However, predicting that an instance belong to C_i with high sensitivity is inconclusive. On the other hand, the high specificity of prediction in Class C_i implies that the reliability of predicting that an instance belongs to C_i is high. And, predicting that an instance does not belong to C_i

413 with high specificity is inconclusive. Moreover, precision for Class C_i gives the proportion of data points
414 predicted to really belong in C_i. Recall gives the proportion of data points in Class C_i which are correctly
415 classified. Lastly, F1 score for Class C_i is a performance measure that combines recall and precision. F1
416 score is computed as the harmonic mean of precision and recall. This is a particular instance of the F_β
417 score which allows for more weight towards one of precision or recall over the other, needed for particular
418 problems. An F-score may have a maximum value of 1.0, indicating perfect precision and recall, or a
419 minimum value of 0, indicating that either precision or recall is zero.

The performance of PHCA and the five benchmark classification algorithms were compared using the
Nemenyi post-hoc test. Pairwise comparisons of the methods were measured using mean rank differences.
A $p - value$ was computed for each pair of methods using the formula

$$p = \frac{\bar{R}_i - \bar{R}_j}{\sqrt{\frac{k(n+1)}{12}}},$$

420 where R_i and R_j are mean ranks of two methods, k is the number of methods and n is the number of
421 performance metric means per method. A $p - value$ of less than $\alpha = 0.05$ implies that the two methods
422 are significantly different.

423 The validation of PHCA and the benchmark classifiers were done in R using the TDA, Caret, and
424 DescTools packages. R-TDA package, containing the Dionysius library, was used for the computation of
425 the persistent homology of the point clouds. The Caret package was used for the implementation of the
426 benchmark classifiers, with the same caret parameters, which are metric and control. While DescTools
427 was used for the implementation of the Nemenyi Test.

RESULTS AND DISCUSSION

429 The performances of the proposed PHCA and the five benchmark classification algorithms (i.e. LDA,
430 CART, KNN, SVM, and RF) for the four classification problems are presented here. Validation results
431 for each classification task are also discussed in the following sections. Program codes, written in R, to
432 implement the classification algorithms can be found on https://github.com/mlddelara/PHCA. Recall that
433 PHCA works in a way that a data point in the testing set will be classified under a class if the inclusion of
434 the point in the particular class' training set results in the least change in the persistence diagram of the
435 training set with the additional data point and least change in the engineered features corresponding the
436 data point to be classified.

Iris plant dataset

438 The iris plant dataset composed of 150 observations created by Fisher (1936) was obtained from the
439 UCI Machine Learning Repository (Dua and Graff, 2017). This is one of the commonly used datasets in
440 pattern recognition. The dataset is divided into 3 categories: iris setosa, iris versicolour, and iris virginica.
441 Each category is comprised of 50 data points, with four attributes namely sepal length, sepal width, petal
442 length, and petal width (expressed in centimeters).

Table 1. Precision and recall per class of each of the six methods when classifying the iris dataset.

Classifier	Precision (%)			Recall (%)		
	Class 1	Class 2	Class 3	Class 1	Class 2	Class 3
LDA	100.00	90.91	100.00	100.00	100.00	90.00
CART	100.00	83.33	100.00	100.00	100.00	80.00
KNN	100.00	90.91	100.00	100.00	100.00	90.00
SVM	100.00	83.33	100.00	100.00	100.00	80.00
RF	100.00	83.33	100.00	100.00	100.00	80.00
PHCA	100.00	94.00	96.00	100.00	95.92	94.11

Table 2. Accuracy, specificity per class, and F1 score per class of each of the six methods when classifying the iris dataset.

Classifier	Accuracy (%)	Specificity (%)			F1 score (%)		
		Class 1	Class 2	Class 3	Class 1	Class 2	Class 3
LDA	96.67	100.00	95.00	100.00	100.00	95.24	94.74
CART	93.33	100.00	90.00	100.00	100.00	90.91	88.89
KNN	96.67	100.00	95.00	100.00	100.00	95.24	94.74
SVM	93.33	100.00	90.00	100.00	100.00	90.91	88.89
RF	93.33	100.00	90.00	100.00	100.00	90.91	88.89
PHCA	96.67	100.00	97.03	97.98	100.00	94.95	95.05

Number of data points:	150		Number of classes:	3	
Training set size:	120		Number of attributes:	4	
Testing set size:	30				
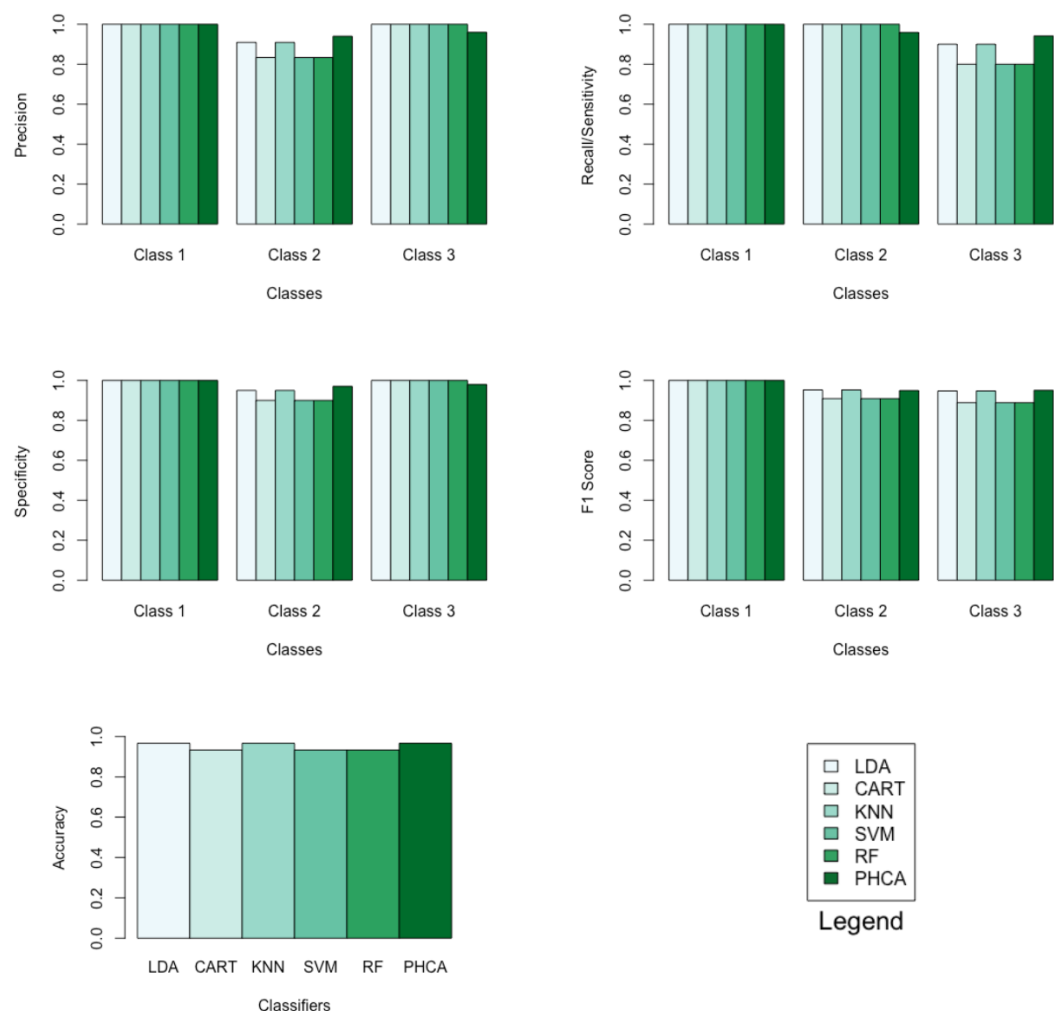
Figure 5. Barplots of performance metrics of PHCA and the five other classifiers for the iris dataset.

Table 3. Nemenyi's test of pairwise comparison of the different classification algorithms for the iris dataset.

	Mean rank difference	p-value
CART-LDA	-6.576923	0.9770
KNN-LDA	0.000000	1.0000
SVM-LDA	-6.576923	0.9770
RF-LDA	-6.576923	0.9770
PHCA-LDA	-3.346154	0.9990
KNN-CART	6.576923	0.9770
SVM-CART	0.000000	1.0000
RF-CART	0.000000	1.0000
PHCA-CART	3.230769	0.9992
SVM-KNN	-6.576923	0.9770
RF-KNN	-6.576923	0.9770
PHCA-KNN	-3.346154	0.9990
RF-SVM	0.000000	1.0000
PHCA-SVM	3.230769	0.9992
PHCA-RF	3.230769	0.9992

Table 1 shows the performance of PHCA and the five major classification algorithms in terms of precision and recall per class. Moreover, Table 2 shows the mean performance of the six methods in terms of accuracy, specificity per class, and F1 score per class. Barplots of these results are presented in Figure 5. LDA, KNN, and PHCA got an accuracy of 96.67%, while CART, SVM and RF obtained an accuracy of 93.33%. Table 3 shows a summary of the Nemenyi test results and the $p-values$ implies that there is no significant difference in the performance of PHCA and the other methods in terms of the mean performance metrics.

Wheat seeds dataset

The wheat seeds dataset was created by Charytanowicz et al. (2010) at the Institute of Agrophysics of the Polish Academy of Sciences in Lublin. It is available at the UCI Machine Learning Repository (Dua and Graff, 2017). The dataset is composed of 210 observations, which is divided equally into 3 categories: kama, rosa, and canadian wheat variety. Each observation is characterized by seven attributes, namely, area, perimeter, compactness, length of kernel, width of kernel, asymmetry coefficient, and length of kernel groove. All of these attributes are real-valued and continuous.

Table 4. Precision and recall per class of each of the six methods when classifying the wheat seeds dataset.

Classifier	Precision (%)			Recall (%)		
	Class 1	Class 2	Class 3	Class 1	Class 2	Class 3
LDA	100.00	100.00	82.35	78.57	100.00	100.00
CART	100.00	100.00	77.78	71.43	100.00	100.00
KNN	100.00	93.33	73.68	57.14	100.00	100.00
SVM	100.00	100.00	82.35	78.57	100.00	100.00
RF	100.00	100.00	77.78	71.43	100.00	100.00
PHCA	85.71	92.86	94.28	86.96	94.21	91.67

Table 5. Accuracy, specificity per class, and F1 score per class of each of the six methods when classifying the wheat seeds dataset.

Classifier	Accuracy (%)	Specificity (%)			F1 score (%)		
		Class 1	Class 2	Class 3	Class 1	Class 2	Class 3
LDA	92.85	100.00	100.00	89.3	88.00	100.00	90.30
CART	90.48	100.00	100.00	85.70	83.30	100.00	87.50
KNN	85.71	100.00	96.40	82.10	72.70	96.60	84.90
SVM	92.86	100.00	100.00	89.30	88.00	100.00	90.30
RF	90.48	100.00	100.00	85.70	83.30	100.00	87.50
PHCA	90.95	92.90	96.50	97.10	86.30	93.50	93.00

Number of data points:	210	Number of classes:	3
Training set size:	168	Number of attributes:	7
Testing set size:	42		
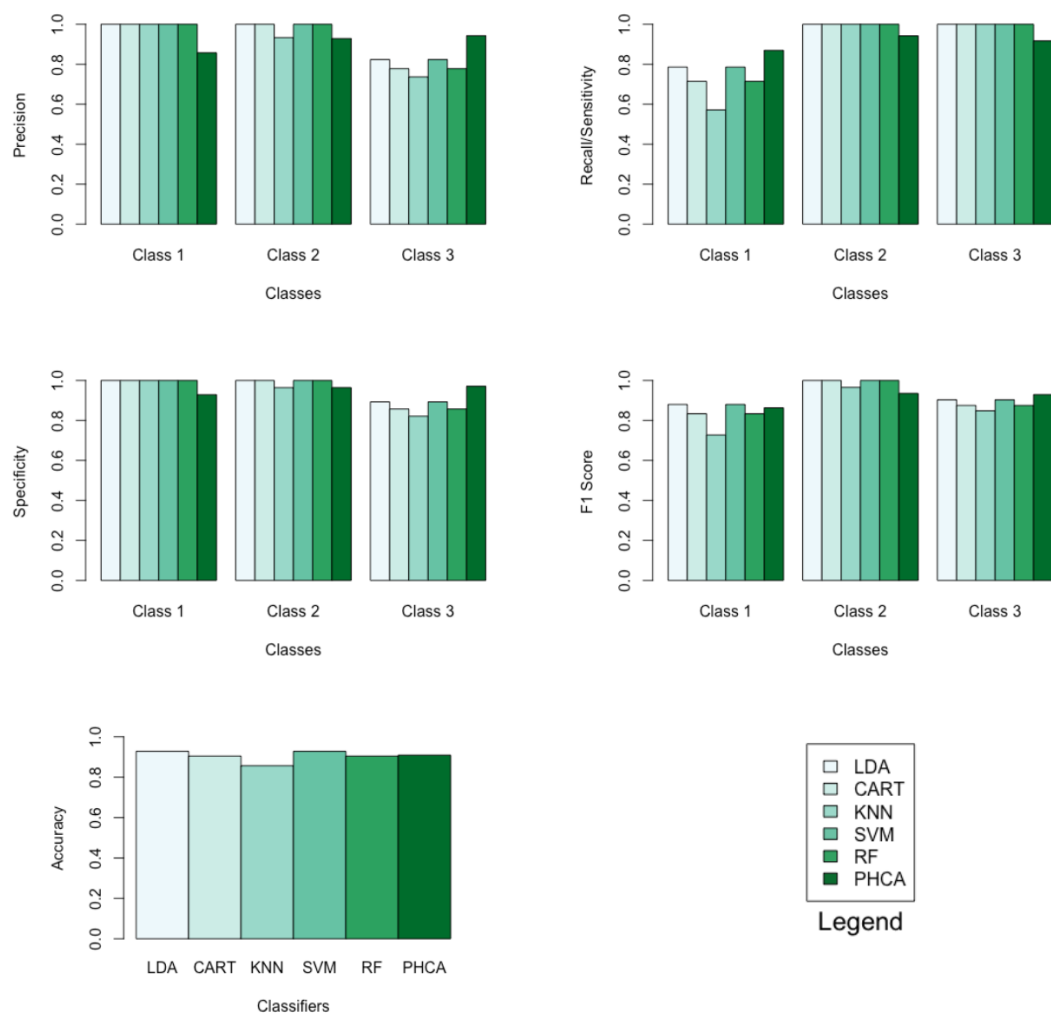
Figure 6. Barplots of performance metrics of PHCA and the five other classifiers for the wheat seeds dataset.

Table 6. Nemenyi's test of pairwise comparison of the different classification algorithms for the wheat seeds dataset.

	Mean rank difference	p-value
CART-LDA	-3.192308	0.9992
KNN-LDA	-10.961538	0.8206
SVM-LDA	0.000000	1.0000
RF-LDA	-3.192308	0.9992
PHCA-LDA	-9.653846	0.8871
KNN-CART	-7.769231	0.9527
SVM-CART	3.192308	0.9992
RF-CART	0.000000	1.0000
PHCA-CART	-6.461538	0.9787
SVM-KNN	10.961538	0.8206
RF-KNN	7.769231	0.9527
PHCA-KNN	1.307692	1.0000
RF-SVM	-3.192308	0.9992
PHCA-SVM	-9.653846	0.8871
PHCA-RF	-6.461538	0.9787

457   The performance of PHCA and the five major classification algorithms in terms of precision and
458 recall per class is shown in table 4. Furthermore, Table 5 compares the six methods in terms of accuracy,
459 specificity per class, and F1 score per class. Barplots of these results are presented in Figure 6. For
460 the wheat seeds dataset, PHCA got the third highest accuracy, which is at 90.95%. Notice that PHCA
461 got the least performance in some metrics per class, but these were offset when PHCA got the highest
462 performance in terms of precision for class 3, recall for class 1, specificity for class 3, and F1 score for
463 class 3. Moreover, the results of the Nemenyi test, which can be found in Table 6, show that there is no
464 significant difference between the performance of the six methods in terms of the mean performance
465 metrics.

466 **Social network ads dataset**
467 The social network ads dataset was created by Raushan (2017) and was retrieved from the Kaggle
468 repository. The dataset is composed of 400 data points and is comprised of uneven number of observations
469 per class. There are 143 data points for Class 1 and 257 data points for Class 2. Each data point has two
470 attributes, age and estimated salary, and a class label, whether a customer purchased a product or not.

Table 7. Precision, recall, specificity, F1 score, and accuracy (in percentage) of each of the six methods when classifying the social network ads dataset.

Classifier	Precision	Recall	Specificity	F1 Score	Accuracy
LDA	80.77	82.35	64.28	81.55	75.95
CART	95.35	80.39	92.85	87.23	84.81
KNN	76.36	82.35	53.57	79.24	72.15
SVM	95.56	84.31	92.85	89.58	87.34
RF	90.91	84.31	85.71	84.21	81.01
PHCA	90.27	86.89	81.20	88.55	85.00
Number of data points:		400	Number of classes:		2
Training set size:		181	Number of attributes:		2
Testing set size:		119			
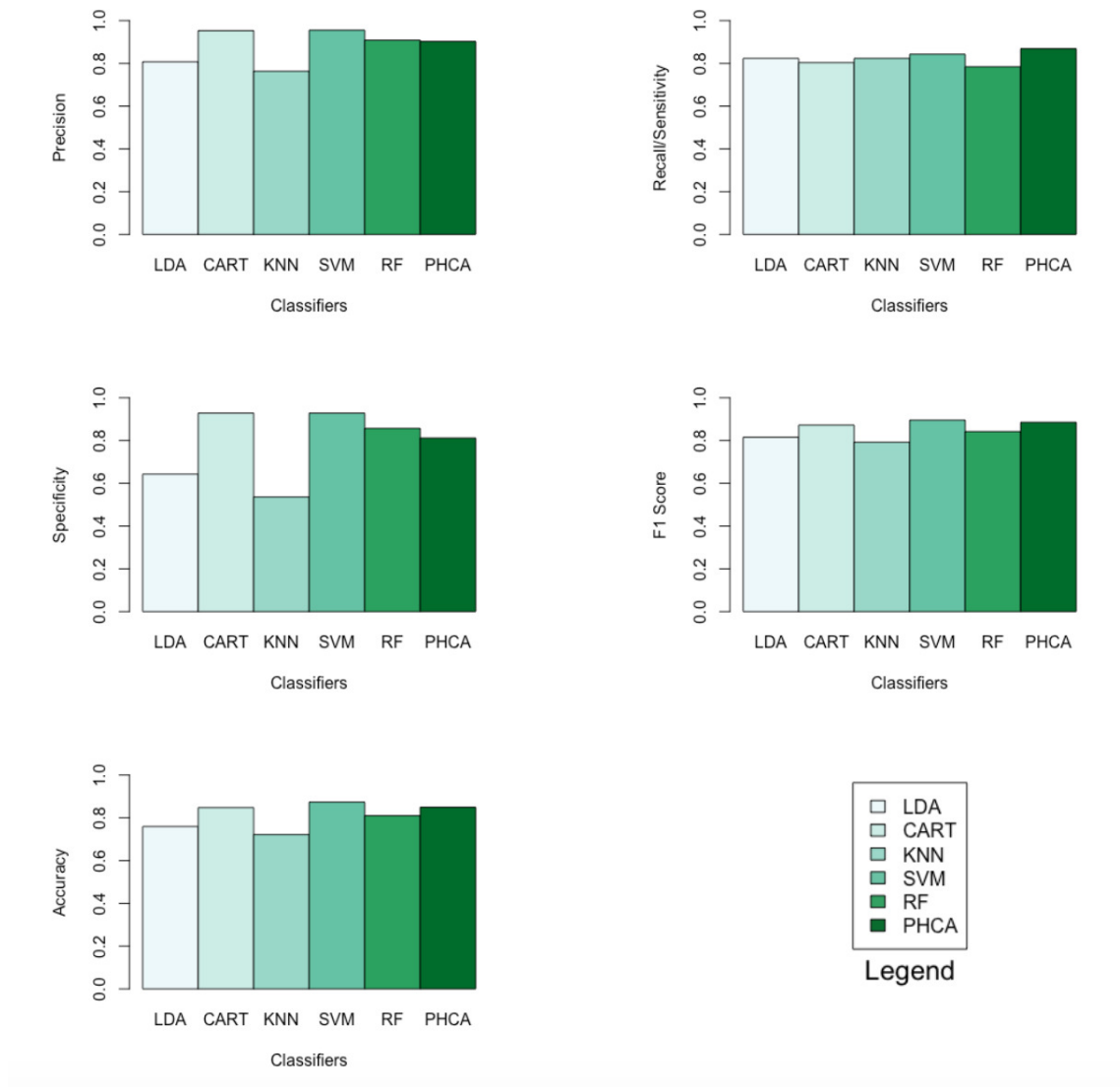
Figure 7. Barplots of performance metrics of PHCA and the five other classifiers for the social network ads dataset.

Table 7 shows the performance of PHCA and the five major classification algorithms in terms of precision, recall, specificity, F1 score, and accuracy. Barplots of these results are presented in Figure 7. For this binary classification problem, PHCA got the highest recall at 86.89%. PHCA also got the second-highest F1 score and accuracy, which are at 88.55% and 85%, respectively. The results of the Nemenyi test, which is summarized in Table 8, show that there is no significant difference between the performance of PHCA and the other classification algorithms in terms the mean performance metrics. Moreover, the mean rank difference of 18 and the $p-value$ of 0.0155 between KNN and SVM shows that SVM performed significantly better than KNN.

PHCA ranked 4th on all metrics with precision of 90.27%, recall of 86.89%, specificity of 81.2%, F1 score of 88.55%, and accuracy of 85%. PHCA bested LDA and KNN in terms of precision, specificity, F1 score, and accuracy, and it bested CART and RF in terms of recall, F1 score, and accuracy.

Table 8. Nemenyi's test of pairwise comparison of the different classification algorithms for the social ads dataset.

	Mean rank difference	p-value
CART-LDA	12.4	0.2252
KNN-LDA	-2.2	0.9988
SVM-LDA	15.8	0.0517
RF-LDA	7.1	0.7987
PHCA-LDA	11.3	0.3254
KNN-CART	-14.6	0.0919
SVM-CART	3.4	0.9903
RF-CART	-5.3	0.9328
PHCA-CART	-1.1	1.0000
SVM-KNN	18.0	0.0155 *
RF-KNN	9.3	0.5515
PHCA-KNN	13.5	0.1477
RF-SVM	-8.7	0.6234
PHCA-SVM	-4.5	0.9661
PHCA-RF	4.2	0.9749

Synthetic dataset

This dataset was created by uniformly sampling 200 points from each of the following figures, the circle defined by $x^2 + y^2 = 25$, the sphere defined by $x^2 + y^2 + (z-1)^2 = 1$, and the torus defined by $\left(3 - \sqrt{x^2 + y^2}\right)^2 + (z+1)^2 = 1$. The x, y, and z coordinates of the 600 points served as the attributes, and the category was assigned according to which figure the points belong to.

Table 9. Precision and recall per class of each of the six methods when classifying the synthetic dataset.

Classifier	Precision (%)			Recall (%)		
	Class 1	Class 2	Class 3	Class 1	Class 2	Class 3
LDA	76.92	96.55	100.00	100.00	70.00	97.5
CART	100.00	100.00	100.00	100.00	100.00	100.00
KNN	100.00	100.00	100.00	100.00	100.00	100.00
SVM	100.00	100.00	100.00	100.00	100.00	100.00
RF	100.00	100.00	100.00	100.00	100.00	100.00
PHCA	100.00	100.00	100.00	100.00	100.00	100.00

Table 10. Accuracy, specificity per class, and F1 score per class of each of the six methods when classifying the synthetic dataset.

Classifier	Accuracy (%)	Specificity (%)			F1 score (%)		
		Class 1	Class 2	Class 3	Class 1	Class 2	Class 3
LDA	89.16	85.00	98.75	100.00	86.95	81.15	98.73
CART	100.00	100.00	100.00	100.00	100.00	100.00	100.00
KNN	100.00	100.00	100.00	100.00	100.00	100.00	100.00
SVM	100.00	100.00	100.00	100.00	100.00	100.00	100.00
RF	100.00	100.00	100.00	100.00	100.00	100.00	100.00
PHCA	100.00	100.00	100.00	100.00	100.00	100.00	100.00

Number of data points:	600	Number of classes:	3
Training set size:	480	Number of attributes:	3
Testing set size:	120		
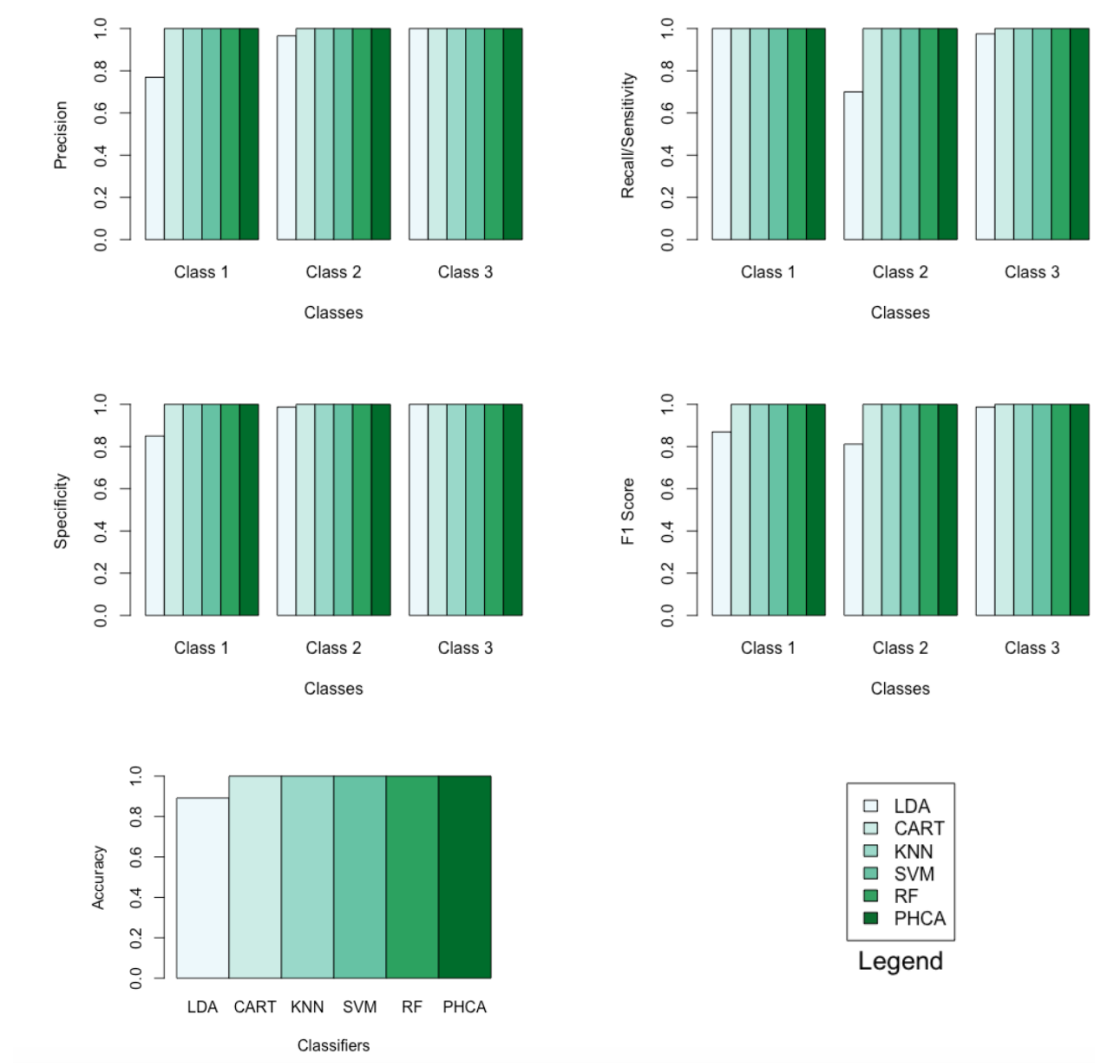
Figure 8. Barplots of performance metrics of PHCA and the five other classifiers for the synthetic dataset.

Table 11. Nemenyi's test of pairwise comparison of the different classification algorithms for the synthetic dataset.

	Mean rank difference	p-value
CART-LDA	30	0.0096 *
KNN-LDA	30	0.0096 *
SVM-LDA	30	0.0096 *
RF-LDA	30	0.0096 *
PHCA-LDA	30	0.0096 *
KNN-CART	0	1.0000
SVM-CART	0	1.0000
RF-CART	0	1.0000
PHCA-CART	0	1.0000
SVM-KNN	0	1.0000
RF-KNN	0	1.0000
PHCA-KNN	0	1.0000
RF-SVM	0	1.0000
PHCA-SVM	0	1.0000
PHCA-RF	0	1.0000

Table 9 shows the performance of PHCA and the five major classification algorithms in terms of precision and recall per class. Moreover, Table 10 shows the performance of the six methods in terms of accuracy, specificity per class, and F1 score per class. Barplots of these results are presented in Figure 8. For this synthetic dataset, PHCA and all benchmark classifiers, except LDA, got 100% for all performance metrics considered in this study. This is consistent with the results of the Nemenyi test which are summarized in Table 11. The $p-values$ show that the performance of LDA is significantly different than the performance of the other algorithms, including PHCA.

MNIST database of handwritten digits

The MNIST database (Modified National Institute of Standards and Technology database (Lecun et al., 1998)) is a massive library of handwritten digits that is often used for training and testing image processing techniques. It was made by "re-mixing" samples from the original datasets from NIST. The MNIST database of handwritten digits contains 60,000 training examples and 10,000 test instances. It is a subset of a bigger set accessible from the National Institute of Standards and Technology (NIST). The digits have been centered and size-normalized in a fixed-size picture. The original NIST black and white photos were resized to fit within a 20x20 pixel frame while maintaining their aspect ratio. The generated photos feature grey levels as a result of the normalization algorithm's anti-aliasing method. The images were centered in a 28x28 image by computing the pixel's center of mass and translating the image to place this point in the 28x28 field's center (Lecun et al., 1998). From the large MNIST database, 500 samples were chosen randomly. At each of the five-fold cross-validation runs, 400 data points served as training points, and 100 data points served as testing points. Note that for this dataset, each data point which is represented by a 28×28 matrix was transformed into a 1×2025 vector using histogram of oriented gradients, a feature descriptor used in computer vision and image processing.

Table 12. Precision per class of each of the six methods when classifying the MNIST dataset.

Classifier	Precision per class (%)									
	1	2	3	4	5	6	7	8	9	10
LDA	90.9	88.9	95.7	92.2	94.1	93.5	97.9	93.2	86.8	88.7
CART	77.8	50.0	NA	23.8	NA	NA	57.1	26.7	NA	NA
KNN	90.9	81.8	90.9	90.9	100.0	100.0	90.9	100.0	100.0	83.3
SVM	100.0	90.0	76.9	100.0	90.0	100.0	100.0	87.5	100.0	88.9
RF	90.9	90.0	90.9	100.0	100.0	100.0	100.0	100.0	100.0	83.3
PHCA	100.0	92.0	92.0	96.0	94.0	86.0	96.0	86.0	86.0	92.0

Table 13. Recall per class of each of the six methods when classifying the MNIST dataset.

Classifier	Recall per class (%)									
LDA	100.0	96.0	88.0	94.0	96.0	86.0	92.0	82.0	92.0	94.0
CART	70.0	60.0	0.0	100.0	0.0	0.0	40.0	80.0	0.0	0.0
KNN	100.0	90.0	100.0	100.0	80.0	90.0	100.0	70.0	90.0	100.0
SVM	100.0	90.0	100.0	100.0	90.0	100.0	100.0	70.0	100.0	80.0
RF	100.0	90.0	100.0	100.0	80.0	100.0	100.0	80.0	100.0	100.0
PHCA	92.6	88.5	95.8	84.1	100.0	87.8	92.3	97.7	95.6	79.3

Table 14. Specificity per class of each of the six methods when classifying the MNIST dataset.

Classifier	Specificity per class (%)									
	1	2	3	4	5	6	7	8	9	10
LDA	100.0	99.6	98.7	99.3	99.6	98.5	99.1	98.0	99.1	99.3
CART	97.8	93.3	100.0	64.4	100.0	100.0	96.7	75.6	100.0	100.0
KNN	98.9	97.8	98.9	98.9	100.0	100.0	98.9	100.0	100.0	97.8
SVM	100.0	98.9	96.7	100.0	98.9	100.0	100.0	98.9	100.0	98.9
RF	98.9	98.9	98.9	100.0	100.0	100.0	100.0	100.0	100.0	97.8
PHCA	100.0	99.1	99.1	99.6	99.3	98.5	99.6	98.5	98.5	99.1

Table 15. F1 score per class and accuracy of each of the six methods when classifying the MNIST dataset.

Classifier	Acc	F1 score per class (%)								
	(%)	1	2	3	4	5	6	7	8	9	10
LDA	92.0	95.2	92.3	91.7	93.1	95.0	89.6	94.8	87.2	89.3	91.2
CART	35.0	73.7	54.5	NA	38.5	NA	NA	47.1	40.0	NA	NA
KNN	92.0	95.2	85.7	95.2	95.2	88.9	94.7	95.2	82.4	94.7	90.9
SVM	93.0	100.0	90.0	87.0	100.0	90.0	100.0	100.0	77.8	100.0	84.2
RF	95.0	95.2	90.0	95.2	100.0	88.9	100.0	100.0	88.9	100.0	90.9
PHCA	92.0	96.2	90.2	93.9	95.0	96.9	86.9	94.1	91.5	90.5	85.2

Number of data points:	500	Number of classes: 10
Training set size:	400	Number of attributes: 2025
Testing set size:	100	
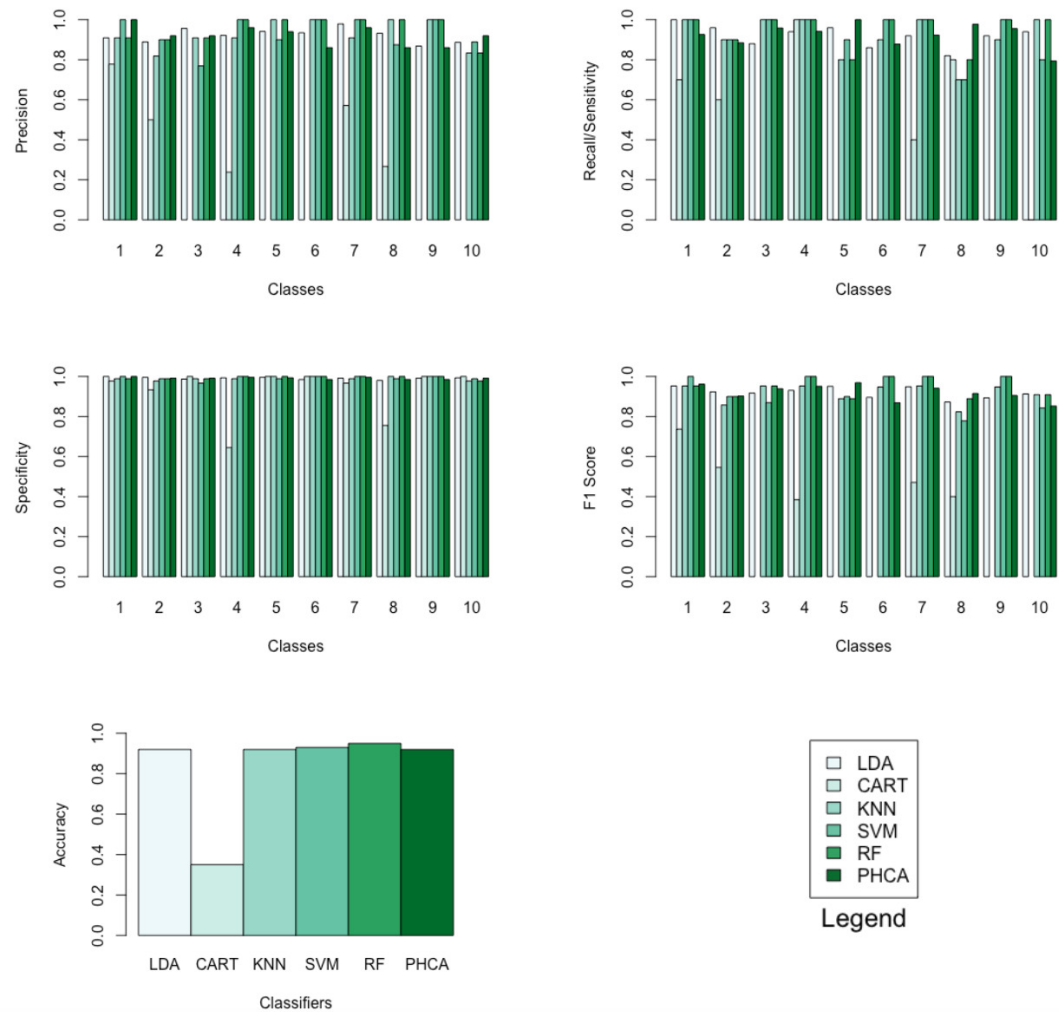
Figure 9. Barplots of performance metrics of PHCA and the five other classifiers for the MNIST dataset.

Table 16. Nemenyi's test of pairwise comparison of the different classification algorithms for the MNIST dataset.

	Mean rank difference	p-value
CART-LDA	-48.59166	0.03322 *
KNN-LDA	16.54878	0.88258
SVM-LDA	32.07317	0.27317
RF-LDA	43.57317	0.04462 *
PHCA-LDA	2.87805	0.99997
KNN-CART	65.14044	0.00086 *
SVM-CART	80.66483	1.0e-05 *
RF-CART	92.16483	2.1e-07 *
PHCA-CART	51.46971	0.01919 *
SVM-KNN	15.52439	0.90814
RF-KNN	27.02439	0.47074
PHCA-KNN	-13.67073	0.94496
RF-SVM	11.50000	0.97367
PHCA-SVM	-29.19512	0.37988
PHCA-RF	-40.69512	0.07541

Tables 12, 13, 14, and 15 show the performance of PHCA and the five major classification algorithms in terms of precision, recall, specificity, and F1-score per class, and accuracy. Barplots of these results are presented in Figure 9. PHCA got an accuracy of 92%. It was outperformed by RF and SVM with accuracy of 95 and 93, respectively. Moreover, Table 16, which is a summary of the Nemenyi post-hoc test results, show that the performance of PHCA is not significantly different from the performance of the other classification algorithms, except from the performance of CART. The mean rank difference of 51.469709 between CART and PHCA, and the $p-value$ of 0.01919 show that PHCA outperformed CART significantly for the sampled MNIST data points, in terms of the mean performance metrics.

The five validation experiments exhibit that the proposed classifier, PHCA, can be used to solve binary or multi-class classification problems, classification problems involving points from the n-dimensional euclidean space or image data, classical or synthetic datasets, and datasets whose number of features ranges from 2 to the thousands. Moreover, the performance of PHCA has no significant difference form the performance of the benchmark classifiers, except for a few instances where a benchmark classification algorithm performed significantly poorer than the performance of PHCA.

These validation results do not imply that PHCA is better than any of the other major classification algorithms. However, these results illustrate the no-free lunch theorem, which implies that no learning algorithm works best on all given problems. Moreover, these results suggest that PHCA can be at par or even better than some classifiers in solving some particular classification problems.

What sets PHCA apart from the well-known machine learning classifiers is that it is non-parametric, but at the same time a linear classifier. It is a non-parametric algorithm in the sense that it does not restrict the data to follow a particular distribution, nor fix the number of datasets' parameters for the algorithm to work.

CONCLUSIONS

The main contribution of this study was the development of PHCA, a non-parametric but linear classifier which utilizes persistent homology, a major and very powerful TDA tool. PHCA was applied in solving four different classification problems with varying sizes, number of classes, and number of attributes. For the four classification problems, the performance of PHCA was measured and compared to the performance of LDA, CART, KNN, SVM, and RF, in terms of precision, recall, specificity, accuracy, and F1 score. A five-fold cross-validation was used in all validation runs. PHCA performed impressively in each of the four classification problems. PHCA ranked either second or third in the first three datasets in almost all metrics, while although it ranked 5th in the synthetic dataset, it got an accuracy of 99.16%. Additionally, all the classifiers, except for PHCA, ranked last in terms of accuracy and F1 scores in at least one of the four classification problems. In conclusion, the validation results show that PHCA can perform

well, or even better, than some of the widely used machine learning classifiers in solving classification problems. Moreover, this study does not imply that PHCA works better than other machine learning algorithms, but this shows that PHCA can work in solving some classification problems.

The validation of PHCA in this study was limited to relatively small problems which are restricted by the computers used in this study. PHCA can be further validated by considering larger problems and by using more powerful computers which can solve problems involving datasets with higher dimensions. Furthermore, some improvements that can be imposed on the proposed classification algorithm in this study is by considering other topological signatures or by considering PH representations other than persistence diagrams, such as persistence landscapes. Recent advancements and modifications on the computation of persistent homology may also be implemented to possibly improve the performance of PHCA. Validation of the proposed algorithm were implemented in *R* using TDA package and GUDHI library in solving persistent homology of data. It should be noted that there are other platforms and solvers which can be used, like JavaPlex, Perseus, Dipha, Dionysus, jHoles, Rivet, Ripser, and PHAT, which offer some variations in the way PH can be computed. Indeed, this study has opened a lot of research opportunities which can be explored by mathematicians, data scientists, and computer programmers.

ACKNOWLEDGMENTS

This work was created with the guidance of Dr. Clarisson Rizzie P. Canlubo and Dr. Rachelle R. Sambayan. The conduct of this research was made possible with the support of the Institute of Mathematical Sciences, College of Arts and Sciences, University of the Philippines Los Baños, and by the DOST Accelerated Science and Technology Human Resource Development Program. These institutions had no role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript.

REFERENCES

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., and Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938.

Alpaydin, E. (2014). *Introduction to Machine Learning, third edition*. Adaptive Computation and Machine Learning series. MIT Press.

Carlsson, G. (2009). Topology and data. *Bull Am Math Soc*, 46.

Carlsson, G. (2014). Topological pattern recognition for point cloud data. *Acta Numerica*, 23:289–368.

Charytanowicz, M., Niewczas, J., Kulczycki, P., Kowalski, P., Lukasik, S., and Zak, S. (2010). A complete gradient clustering algorithm for features analysis of x-ray images.

Chazal, F. and Michel, B. (2021). An introduction to topological data analysis: Fundamental and practical aspects for data scientists. *Frontiers in Artificial Intelligence*, 4, 667963.

Chen, C., Ni, X., Bai, Q., and Wang, Y. (2019). A topological regularizer for classifiers via persistent homology. In Chaudhuri, K. and Sugiyama, M., editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 2573–2582. PMLR.

Chowdhury, S. and Mémoli, F. (2018). Persistent path homology of directed networks. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, page 1152–1169, USA. Society for Industrial and Applied Mathematics.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Edelsbrunner, Letscher, and Zomorodian (2002). Topological persistence and simplification. *Discrete & Computational Geometry*, 28(4):511–533.

Edelsbrunner, H. and Harer, J. (2008). Persistent homology — a survey.

Edelsbrunner, H. and Harer, J. (2010). *Computational topology: an introduction*. Am. Math. Soc., Providence.

Edelsbrunner, H., Jablonski, G., and Mrozek, M. (2015). The persistent homology of a self-map. *Foundations of Computational Mathematics*, 15:1213–1244.

Edwards, P., Skruber, K., Milićević, N., Heidings, J. B., Read, T.-A., Bubenik, P., and Vitriol, E. A. (2021). Tdaexplore: Quantitative analysis of fluorescence microscopy images through topology-based machine learning. *Patterns*, 2(11):100367.

Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Annual Eugenics*, 7(Part II):179–188.

593 Giansiracusa, N., Giansiracusa, R., and Moon, C. (2019). Persistent homology machine learning for
594   fingerprint classification. In *2019 18th IEEE International Conference On Machine Learning And*
595   *Applications (ICMLA)*, pages 1219–1226.

596 Goldenberg, A., Zheng, A. X., Fienberg, S. E., and Airoldi, E. M. (2010). A survey of statistical network
597   models. *Found Trends Mach Learn*, 2.

598 Gonzalez-Diaz, R., Gutiérrez-Naranjo, M. A., and Paluzo-Hidalgo, E. (2020). Representative datasets:
599   The perceptron case.

600 Hofer, C., Kwitt, R., Niethammer, M., and Uhl, A. (2017). Deep learning with topological signatures. In
601   *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17,
602   page 1633–1643, Red Hook, NY, USA. Curran Associates Inc.

603 Ignacio, P. S. P. and Darcy, I. K. (2019). Tracing patterns and shapes in remittance and migration networks
604   via persistent homology. *EPJ Data Science*, 8(1):1.

605 Islambekov, U. and Gel, Y. R. (2019). Unsupervised space–time clustering using persistent homology.
606   *Environmetrics*, 30(4):e2539.

607 Ismail, M. S., Md Noorani, M. S., Ismail, M., Abdul Razak, F., and Alias, M. A. (2020). Predicting next
608   day direction of stock price movement using machine learning methods with persistent homology:
609   Evidence from kuala lumpur stock exchange. *Applied Soft Computing*, 93:106422.

610 Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document
611   recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

612 Nicolau, M., Levine, A. J., and Carlsson, G. (2011). Topology based data analysis identifies a subgroup
613   of breast cancers with a unique mutational profile and excellent survival. *Proc Natl Acad Sci USA*, 108.

614 Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., and Harrington, H. A. (2017). A roadmap for the
615   computation of persistent homology. *EPJ Data Science*, 6(1):17.

616 Pokorny, F. T., Hawasly, M., and Ramamoorthy, S. (2014). Multiscale topological trajectory classification
617   with persistent homology. In *Robotics: Science and Systems X*, pages 1219–1226.

618 Pun, C. S., Xia, K., and Lee, S. X. (2018). Persistent-homology-based machine learning and its applica-
619   tions – a survey.

620 Raushan, R. (2017). Social network ads, version 1.

621 Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural Computation*,
622   page 1341–1390.

623 Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions*
624   *on Evolutionary Computation*, 1(1):67–82.

625 Xia, K. and Wei, G. . W. (2014). Persistent homology analysis of protein structure, flexibility, and folding.
626   *Int J Numer Methods Biomed Eng*, 30.

627 Yu-Min Chung, W. C. and Lawson, A. (2020). A persistent homology approach to time series classification.

628 Yuan, G., Ho, C., and Lin, C. (2012). Recent advances of large-scale linear classification. *Proceedings of*
629   *the IEEE*, 100(9):2584–2603.

630 Zomorodian, A. and Carlsson, G. (2005). Computing persistent homology. *Discrete & Computational*
631   *Geometry*, 33(2):249–274.