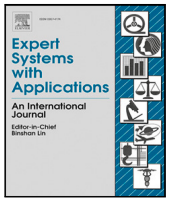
Expert Systems With Applications 245 (2024) 122998

Contents lists available at ScienceDirect

Expert Systems With Applications

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



Support vector machines within a bivariate mixed-integer linear programming framework

John Alasdair Warwicker *, Steffen Rebennack

Institute of Operations Research (IOR), Karlsruhe Institute of Technology, 76185 Karlsruhe, Baden-Württemberg, Germany

ARTICLE INFO

Keywords:
Support vector machine
Optimisation
Mixed-integer linear programming
Outlier detection
Feature selection

ABSTRACT

Support vector machines (SVMs) are a powerful machine learning paradigm, performing supervised learning for classification and regression analysis. A number of SVM models in the literature have made use of advances in mixed-integer linear programming (MILP) techniques in order to perform this task efficiently. In this work, we present three new models for SVMs that make use of piecewise linear (PWL) functions. This allows effective separation of data points where a simple linear SVM model may not be sufficient. The models we present make use of binary variables to assign data points to SVM segments, and hence fit within a recently presented framework for machine learning MILP models. Alongside presenting an inbuilt feature selection operator, we show that the models can benefit from robust inbuilt outlier detection. Experimental results show when each of the presented models is effective, and we present guidelines on which of the models are preferable in different scenarios.

1. Introduction

Machine learning (ML) is a branch of artificial intelligence which utilises methods from statistics and computer science to perform complex data analysis tasks (Marsland, 2011). As opposed to ML methods which make predictions, mixed-integer programming (MIP) approaches aim to make decisions. Although there is a vast amount of research into these growing fields, the recent advances in the computing power of MIP solution software have rarely been utilised in tandem with ML approaches (Bertsimas & Dunn, 2019). While MIP models tend to be slow, an extended understanding of their solution methods means tailored approaches can be applied to increase their efficiency and aid in the solution of ML problems.

Supervised learning is a subfield of ML which aims to train a model on labelled data. Examples of supervised learning methods which have been approached from a mathematical programming perspective include regression and classification problems (Park et al., 2017; Rebennack & Krasko, 2020; Sudermann-Merx & Rebennack, 2021). Recently, Warwicker and Rebennack (2023b) presented a framework of mixed-integer linear programming (MILP) models for regression and clustering problems, whereby the knowledge of the special structure of the models allows for improvements in efficiency and robustness.

Another well-known supervised ML approach is support vector machines (SVMs), which are a robust prediction method based on statistic learning theory (Boser et al., 1992; Cortes & Vapnik, 1995; Vapnik, 1998). The aim of SVMs is to classify labelled data, allowing for predictions based on new inputs. Given a set of training examples, each labelled as belonging to one of two categories, SVMs find a separating hyperplane such that the distance to the nearest data points is maximised (i.e., maximising homogeneity), while misclassification errors are minimised. Applications of SVMs are seen in many fields, including finance (Fieberg et al., 2023; Huang et al., 2005), medicine (Guyon et al., 2002) and computational biology (Noble, 2006), alongside problem decomposition for combinatorial optimisation problems (Sun et al., 2021, 2019). For a general overview of SVM approaches, see e.g., Burges (1998) and Christmann and Steinwart (2008).

While finding an SVM has typically been approach from a machine learning perspective, a number of mathematical programming approaches have been presented in the literature. Early approaches for classifying separable data were presented by Mangasarian (1965), which assumes the convex hulls of each set of labelled data do not intersect, and Mangasarian (1968), where the assumption on the convex hulls is bypassed. These approaches used simple linear programming (LP) methods to find the separating hyperplanes. In particular, when the convex hulls intersect, the latter method presented a separating plane consisting of connected non-linear manifolds — we refer to this as a piecewise linear (PWL) SVM in the remainder of this paper.

Among the first mathematical programming approaches, Vapnik (1998) presented a non-linear mathematical programming model for

* Corresponding author.
E-mail addresses: john.warwicker@kit.edu (J.A. Warwicker), steffen.rebennack@kit.edu (S. Rebennack).

https://doi.org/10.1016/j.eswa.2023.122998
Received 24 August 2023; Received in revised form 5 December 2023; Accepted 18 December 2023
Available online 9 January 2024

SVM based on statistical learning theory. Their approach aims to minimise the *structural risk*, which is calculated as the width of the margin of the hyperplane (i.e., how close it is to the nearest data points). Furthermore, they introduce a penalty term for misclassification errors, which is known as minimising the *empirical risk*. Bradley and Mangasarian (1998) presented the first LP formulations for SVM using a linear distance metric, without any assumptions on the data. Zhou et al. (2002) presented a linear programming SVM model which directly controls the maximisation of the margin by introducing another variable, allowing for increased efficiency (at a small cost to the overall accuracy). Note that these models are unsuitable for non-separable data.

The models described above utilise a parameter to control the tradeoff between structural and empirical risk. Another consideration to make in SVM models is which features of the data to select. Typically, the number of features of a data set is much larger than the number of data points, and handling all of the features leads to very inefficient models with results that are difficult to interpret. Appropriately selecting features reduces these risks and minimises overfitting risks. Regarding feature selection in MILP models for SVM, Maldonado et al. (2014) proposed an extension of the previously discussed formulations which implement feature selection within the model, by introducing a budget constraint to limit the number of features that are used. Furthermore, the cost of acquiring each feature is considered. Benítez-Peña et al. (2019) presented an integer linear programming approach for SVM incorporating cost-sensitive feature selection, where the objective function explicitly minimises misclassifications while limiting the number of features. Labbé et al. (2019) presented a MILP model for feature selection in SVM which works by bounding the weights of the variables, as well as the use of a budget constraint. The value of the bounds of the weights is crucial to the performance of their formulation; hence, a number of methods are discussed to find tight bounds. A similar problem is considered in Baldomero-Naranjo et al. (2020), where heuristic methods are proposed.

Baldomero-Naranjo et al. (2021) recently proposed a MILP formulation for SVM which implements simultaneous feature selection and outlier detection; their model also utilises heuristics to model the big-M values and to find solutions efficiently. Aside from robust approaches, recent advances in mathematical programming formulations for SVM models have considered noisy data (see e.g., Blanco et al., 2022) and semi-supervised approaches (see e.g., Burgard et al., 2023).

In general, the approaches above are limited by either an assumption on the separability of the data, or by the shape requirements on the separating hyperplane that is being built (i.e., its linearity and connectivity). In this paper, we present extensions of the current models for SVM that allow for more accurate classification. The SVMs model we present are analytical in nature, and can be used for training models on the given data. In particular, we present relaxations that allow the separating hyperplanes to be constructed as continuous PWL functions. The models we present are inspired by existing MILP models for PWL regression and clustering (Park et al., 2017; Rebennack & Krasko, 2020; Warwicker & Rebennack, 2023b). Due to the special symmetrical structure of these models, their reliance on binary variables to assign data points to segments of the hyperplanes, and the logical implications modelled by big-$\mathcal{M}$ constraints, they allow improvements to be made in terms of efficiency and robustness.

In particular, this paper has the following contributions:

- We firstly present three new MILP models for SVM. The first allows the data to be modelled by multiple, non-connected linear hyperplanes; the second fits a single, continuous PWL hyperplane to classify the data; the third allows the data to be modelled by a number of continuous PWL hyperplanes.
- We discuss how the three presented models fit within the recently presented framework for similar regression- and classification-based machine learning problems presented by Warwicker and Rebennack (2023b). This allows the implementation of inbuilt outlier detection.
- We present an extension of the framework to allow for feature selection, which is based on SVM models in the literature.
- We present experimental results on a series of ad-hoc data sets designed to showcase where each model is most effective, alongside a series of real-world data sets exhibiting similar characteristics. We further compare the presented models with a state-of-the-art SVM model with implicit outlier detection and explicit feature selection.

The rest of this paper is structured as follows. In Section 2, we discuss existing quadratically constrained and mixed-integer linear models for SVM from the literature. We introduce three new MILP models for SVM in Section 3, and discuss how they fit within the existing MILP framework for machine learning models in Section 4. We present an experimental analysis in Section 5, and we conclude with Section 6.

2. Mathematical programming models for support vector machines

2.1. Preliminaries

Throughout this paper we use the following notation: $[n]$ to denote the set $\{1,\dots,n\}$; $\|\cdot\|_k$ denotes the ℓ_k norm (for $k \in \{1,2,\infty\}$).

Consider a set of data Ω, which has been partitioned into two classes. Each object $i \in [I]$ in Ω is represented by a pair $(x_i, z_i) \in \mathbb{R}^n \times \{-1,1\}$, where n is the number of features of the data. x_i contains the (continuous) values of such features, and z_i provides the fixed labels, either 1 or -1, associated with the two classes in Ω.

For a given set of labelled objects, the support vector machine (SVM) determines a separating hyperplane $f(x) := w^\intercal \cdot x + d = 0$ that optimally separates the two classes (where w is the normal vector to the hyperplane, with weights w_n for each feature $n \in [n]$). If the data is linearly separable, two parallel hyperplanes that separate the data can be found such that the distance between them is maximised (with the maximum-margin hyperplane $f(x)$ lying between them). Data labelled with $z_i = 1$ should lay on or above the hyperplane $w^\intercal \cdot x + d = 1$, while data labelled with $z_i = -1$ should lay on or below the hyperplane $w^\intercal \cdot x + d = -1$. The distance between the hyperplanes is given by $2/\|w\|$, so to maximise the distance between the planes, we wish to minimise some function of $\|w\|$. To prevent data falling into the margin, we require

$$\begin{aligned} w^\intercal \cdot x_i + d &\geq 1 \quad \text{if } z_i = 1, \\ w^\intercal \cdot x_i + d &\leq -1 \quad \text{if } z_i = -1. \end{aligned}$$

That is,

$$z_i(w^\intercal \cdot x_i + d) \geq 1 \qquad \forall i \in [I] \qquad (\star).$$

If the data is not linearly separable, the hyperplane should also seek to minimise classification errors (i.e., to penalise those data points not adhering to equation $(\star)$). The classical SVM model objective function is a compromise between structural risk (given by a function of the inverse of the margin, $\|w\|$) and the empirical risk (given by a function of the total deviation of misclassified objects) (Vapnik, 1998; Zhou et al., 2002).

2.2. Linear programming models for SVM

The classical formulation for SVM, which was introduced by Bradley and Mangasarian (1998), introduces a slack variable ($\xi_i \in \mathbb{R}^+$) for each data point to measure the deviation if that data point is misclassified. A penalty parameter $C > 0$ is also introduced to balance the tradeoff between structural and empirical risk.

$$(\ell_2 - \text{SVM}) \quad \min \quad \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{I}\xi_i \tag{1a}$$

$$\text{s.t.} \quad z_i(w^\intercal x_i + d) \geq 1 - \xi_i \quad \forall i \in [I] \tag{1b}$$

$$\xi_i \geq 0 \qquad \forall i \in [I] \tag{1c}$$

The objective function (1a) aims to minimise the structural risk (using the ℓ_2 norm) and empirical risk. If a data point i is misclassified (i.e., it has label $z_i = 1$ and lies below the line, or has label $z_i = -1$ and lies above the line), then constraint (1b) assigns a non-negative value to ξ_i to measure how far it has deviated. Constraint (1c) gives the domain of the continuous variables ξ_i.

Bradley and Mangasarian (1998) also presented a linear formulation using the ℓ_1 norm, where the objective function is replaced by:

$$(\ell_1 - \text{SVM}) \quad \min \quad \|w\|_1 + C \sum_{i=1}^{I} \xi_i \tag{2}$$

An equivalent linear formulation is presented in formulation (3). In order to guarantee the linearity, the absolute values of the weights of the hyperplane coefficients are calculated.

$$(\ell_1 - \text{SVM}) \quad \min \quad \sum_{j=1}^{n} W_j + C \sum_{i=1}^{I} \xi_i \tag{3a}$$

$$\text{s.t.} \quad (1b)\text{–}(1c) \tag{3b}$$

$$w_j \leq W_j \qquad \forall j \in [n] \tag{3c}$$

$$w_j \geq -W_j \qquad \forall j \in [n] \tag{3d}$$

$$W_j \geq 0 \qquad \forall j \in [n] \tag{3e}$$

The objective function (3a) again seeks to minimise the structural and empirical risk. However, the structural risk is calculated somewhat differently. Constraints (3c)–(3e) calculate the absolute values of each of the weights to guarantee a linear formulation.

Zhou et al. (2002) presented another linear programming formulation for SVM. Their formulation, which explicitly controls the maximisation of the margin through the introduction of a continuous margin variable r, uses ideas from statistical learning theory (Vapnik, 1998). By seeking explicitly to maximise the margin, as well as minimising the empirical risk in the case of non-separable data, the overall risk is minimised, while some generalisation is lost in the simplicity of the formulation. We present their model in formulation (4).

$$(\text{LP} - \text{SVM}) \quad \min \quad -r + C \sum_{i=1}^{I} \xi_i \tag{4a}$$

$$\text{s.t.} \quad z_i(w^\intercal x_i + d) \geq r - \xi_i \qquad \forall i \in [I] \tag{4b}$$

$$-1 \leq w_j \leq 1 \qquad \forall j \in [n] \tag{4c}$$

$$\xi \geq 0 \qquad \forall i \in [I] \tag{4d}$$

$$r \geq 0 \tag{4e}$$

In formulation (4), the weights of the hyperplane are constrained (in constraint (4c)) to lie between -1 and 1. Instead, the margin variable r, which is calculated in constraint (4b) and (4e), is minimised in the objective function (4e). The loosening of the margin (through the variable r) leads to some loss of generalisation; however, this can lead to vast speedups (Zhou et al., 2002).

To illustrate the application of SVM, we apply the linear formulation (3) to a synthetic data set. We set the value of the penalty term $C = 10$ in order to focus on minimising the empirical risk. Fig. 1a shows an application to a separable data set, in which the margin is minimised. In this case, each data point is correctly classified and lies outside (or on) the margin, meaning $\xi_i = 0$ for each $i \in [I]$. The objective function value in this case is given by the structural risk, and is 1.94.

For Fig. 1b, we exchanged the labels of two data points, resulting in a non-separable data set. In this case, the margin is increased, meaning eight data points now fall within the margin. The misclassification error of the two exchanged data points leads to a large objective function value. In this case, outlier detection might prove useful, whereby the resulting model will better fit the non-outlier data points. In this case, the structural risk is calculated as 1.17, whereas the empirical risk is calculated as 9.05; this leads to an objective function value of 91.67 (i.e., $1.17 + 9.05C$). If we decrease the value of C, we see a larger sum of the slack values (contributing to the empirical risk), with a smaller value of the structural risk. This highlights the importance of setting the parameter C. This effect is discussed further in Section A of the appendix.

For a comprehensive overview of LP formulations for SVM, see e.g., (Rivas-Perea et al., 2012).

2.3. Mixed-integer linear models: Implementing feature selection

In most cases, data sets often contain data with a large number of features, many of which are noisy, irrelevant or redundant. Hence, feature selection is necessary to identify the most relevant features, as well as reducing computational costs. Recent approaches for SVM have embedded feature selection within the models.

Feature selection can be classified in three ways (Guyon et al., 2008). *Wrapper* methods aim to measure the usefulness of the features based on the performance of the classifier (i.e., the SVM), while *filter* methods aim to measure the relevance of the feature via statistics. While wrapper methods are more useful since they are attempting to optimise the performance of the classifier, they are more computationally expensive. *Embedded* methods aim to simultaneously perform feature selection and classifier construction, and can be thought of as a compromise between the two former methods.

In practical applications for SVM, these methods aim to select the best subset of features based on the misclassification cost, and the cost of acquiring each feature (Freitas et al., 2007). From an optimisation perspective, bio-inspired heuristics for implementing feature selection within SVM models have recently been considered (see e.g., Aladeemy et al., 2017; Alcaraz et al., 2022; Huang & Wang, 2006), alongside mathematical programming approaches. The goal of the following mixed-integer linear programming (MILP) models is to implement embedded feature selection within the model. This is typically done using a binary variable for each feature, which is set to 1 if that feature is chosen.

Maldonado et al. (2014) presented an extension of formulation (3) (first presented by Bradley and Mangasarian (1998)) to include feature selection. For each feature $j \in [n]$, the relative weight is bounded above and below, and the cost of acquiring each feature is given by $c_j \geq 0$ (if this information is unavailable, this can be fixed to 1 for all features). The budget $B > 0$ limits the amount of features to be selected.

$$(l_1 - \text{SVM}_\text{F}) \quad \min \quad \sum_{i=1}^{I} \xi_i \tag{5a}$$

$$\text{s.t.} \quad z_i(w^\intercal x_i + d) \geq 1 - \xi_i \qquad \forall i \in [I] \tag{5b}$$

$$L_j v_j \leq w_j \leq U_j v_j \qquad \forall j \in [n] \tag{5c}$$

$$\sum_{j=1}^{n} c_j v_j \leq B \tag{5d}$$

$$v_j \in \{0, 1\} \qquad \forall j \in [n] \tag{5e}$$

$$\xi_i \geq 0 \qquad \forall i \in [I] \tag{5f}$$

The objective function (5a) minimises the sum of errors (i.e., the empirical risk), where the errors are calculated in constraint (5b). If the binary variable v_j (for feature $j \in [n]$) is set to 1, then the weight is bounded within the limits $[L_j, U_j]$; otherwise, it is set to 0. Constraint (5d) ensures the optimal selection of features such that the budget is not exceeded, while constraints (5e)–(5f) give the variable domains.

Formulation (5) does not explicitly aim to minimise the structural risk in the objective function (5a). However, features are selected (in constraints (5b)–(5e)) using a budget constraint, while forcing the

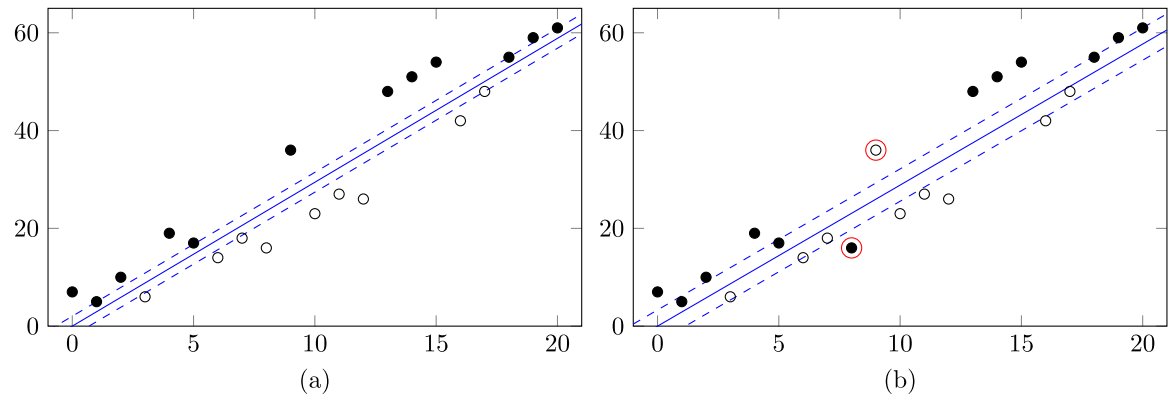


Fig. 1. An example of ℓ_1-SVM applied to a synthetic data set using (a) Separable data; (b) Non-separable data.

weights for each feature to fall within a given interval $[L_j, U_j]$ (if selected, i.e., if $v_j = 1$). Hence, the binary variables and the weight vector w are linked, which implicitly minimises the structural risk while avoiding the requirement for a trade-off parameter C.

Maldonado et al. (2014) also presented a similar extension to formulation (4) (first presented by Zhou et al. (2002)), in which the maximisation of the margin is explicitly controlled. In both models, for a given budget $B > 0$, it is possible that there are a number of different solutions with objective value 0, which is typical for data sets with many more features than objects. Hence, Labbé et al. (2019) proposed the following extension of formulation (5), which, as well as implementing feature selection using a budget constraint, also takes the structural and empirical risk into account. The weights are decomposed into a positive and negative part, of which at most one is non-zero. For ease of presentation, the costs of acquiring each feature are set to 1, and the budget B states how many features can be selected.

$$(l_1 - \text{SVM}'_{\text{F}}) \quad \min \quad \sum_{j=1}^{n}(w_j^+ + w_j^-) + C\sum_{i=1}^{I}\xi_i \tag{6a}$$

$$\text{s.t.} \quad z_i\left(\sum_{j=1}^{n}(w_j^+ - w_j^-)x_i j + d\right)$$

$$\geq 1 - \xi_i \qquad \forall i \in [I] \tag{6b}$$

$$\sum_{j=1}^{n} v_j \leq B \tag{6c}$$

$$w_j^+ \leq u_j v_j \qquad \forall j \in [n] \tag{6d}$$

$$w_j^- \leq -l_j v_j \qquad \forall j \in [n] \tag{6e}$$

$$w_j^+ \geq 0, w_j^- \geq 0 \qquad \forall j \in [n] \tag{6f}$$

$$\text{(5e)–(5f)} \tag{6g}$$

Formulation (6) explicitly calculates the structural and empirical risk to disambiguate between solutions that would attain the same objective value in formulation (5). In particular, the objective function (6a) calculates the structural risk as the sum of the positive and negative aspects of the weight, while the empirical risk is calculated again as the sum of errors. Constraint (6b) acts similarly to the previous models (e.g., to constraint (5b)), and constraint (6c) imposes a budget on the features (assuming uniform costs). Then, constraints (6d)–(6e) calculate the positive and negative parts of the weights for the features that are selected, while constraint (6f) ensures their non-negativity.

Lee et al. (2020) also allowed for group feature selection within the model, as well as selecting individual features. In their proposed model, savings can be made by selecting from a group of features compared to selecting them individually. Furthermore, they presented a robust model that allows for uncertainty in the feature costs. Baldomero-Naranjo et al. (2021) presented an advancement of the robust SVM model first presented by Brooks (2011) which implements feature selection alongside outlier detection. The feature selection is implemented similarly to Labbé et al. (2019) using a *budget* variable, and the outlier detection works similarly.

The models described within this subsection require tight upper and lower bounds for the weights, which act as big-$\mathcal{M}$ constructs. Belotti et al. (2016) presented a non-convex, non-linear model (based on the quadratically-constrained mixed-integer model for SVM presented by Brooks (2011)), which reformulates the initial model to avoid relying on complicating big-$\mathcal{M}$ constraints. Somewhat surprisingly, they found the reformulated model was faster; however, recent advancements in MILP solution software (such as CPLEX) have since implemented many of the modelling aspects that led to the increased efficiency of the reformulated model, such as aggressive bound tightening. Labbé et al. (2019) considered two approaches to find tight values for the big-$\mathcal{M}$ constants, including an exact approach in which a simplified linear relaxation of the MILP is iteratively solved until tight bounds have been computed. Baldomero-Naranjo et al. (2021) also presented a number of approaches for tightening the big-$\mathcal{M}$ constraints based on approaches introduced by Baldomero-Naranjo et al. (2020).

3. A framework for piecewise linear support vector machines

The models presented in the previous section produce a separating hyperplane with a maximised margin to classify labelled data. This allows the predicted classification of future data points based on their relative position to the hyperplane. In this section, we present a framework consisting of three new models for finding SVM in $\mathbb{R}^2$. The basis of this framework is that instead of presenting a singular, linear hyperplane, a more accurate classification of data can be provided using a connected, piecewise linear (PWL) hyperplane. PWL hyperplanes for SVM have been presented by, e.g., Mangasarian (1968), which allow for data sets that are non-separable by traditional SVM methods to become separable. Furthermore, for data which does not adhere to a strict, linear relationship, a PWL separating hyperplane will allow for better models, and retain the linearity which makes non-linear hyperplanes difficult to calculate.

PWL functions consist of connected linear segments which intersect at breakpoints. The models we present in this section use binary variables to assign data points to their respective segments. Hence, it is important that the number of segments is chosen well. If the number of segments in the PWL function is too large, this can lead to redundant segments and present possible occurrences of overfitting the given data. Further problems can occur when all data points associated with a given segment share the same label; in such cases, the model may suggest a hyperplane that is far away from the data, which provides little information. We further use big-$\mathcal{M}$ constraints within the presented formulations to model logical implications, implying the efficiency of the models depends on the tightness of the big-$\mathcal{M}$ values. We present a discussion on how to overcome such problems after we present the models.

3.1. Model 1: Fitting multiple hyperplanes

As a first preliminary model, we present a simple extension to formulation (3) which allows the data to be modelled by multiple, i.e., $B \geq 1$, disconnected (linear) hyperplanes. Each data point $i \in [I]$ will be *assigned* to one hyperplane $b \in [B]$, using a binary variable and a big-$\mathcal{M}$ constraint. We require the data points to be ordered by some feature (for example, if a number of different observations are recorded at different time points, the variable representing time will be ordered). We note that although the choice of feature can affect the quality of the resulting SVM model, this choice can be typically inferred from the data set and the goal of the classification task. To achieve the modelling, we use the ordering constraints for PWL regression functions that were introduced by Rebennack and Krasko (2020). This will allow us to ensure that the model does not assign one hyperplane to all data points with a given label. The resulting hyperplanes will be of the form $w_b^\intercal x_i + d_b = 0$, for $b \in [B]$. We present Model 1 (denoted as PWL-SVM$_1$) in formulation (7).

(PWL-SVM$_1$)

$$\min \quad \sum_{b=1}^{B}\sum_{j=1}^{n} W_{b,j} + C\sum_{i=1}^{I}\xi_i \tag{7a}$$

$$\text{s.t.} \quad z_i(w_b^\intercal x_i + d_b) \geq 1 - \xi_i - M_i^1(1-\delta_{i,b}) \qquad \forall i \in [I];\ b \in [B] \tag{7b}$$

$$w_{b,j} \leq W_{b,j} \qquad \forall b \in [B];\ j \in [n] \tag{7c}$$

$$w_{b,j} \geq -W_{b,j} \qquad \forall b \in [B];\ j \in [n] \tag{7d}$$

$$\sum_{b=1}^{B}\delta_{i,b} = 1 \qquad \forall i \in [I] \tag{7e}$$

$$\delta_{i+1,b+1} \leq \delta_{i,b} + \delta_{i,b+1} \qquad \forall i \in [I-1];\ b \in [B-1] \tag{7f}$$

$$\delta_{i+1,1} \leq \delta_{i,1} \qquad \forall i \in [I-1] \tag{7g}$$

$$\delta_{i,B} \leq \delta_{i+1,B} \qquad \forall i \in [I-1] \tag{7h}$$

$$\xi_i \in [0, M_i^1] \qquad \forall i \in [I] \tag{7i}$$

$$\delta_{i,b} \in \{0,1\} \qquad \forall i \in [I];\ b \in [B] \tag{7j}$$

$$W_{b,j} \geq 0;\ w_{b,j} \in \mathbb{R} \qquad \forall b \in [B];\ j \in [n] \tag{7k}$$

The objective function (7a) seeks to minimise the total structural risk across the B hyperplanes, as well as the total empirical risk. Such an objective function mirrors that of formulation (3) and we utilise this objective function in order to ensure the linearity of the model. In the objective, the sum of the absolute values of the weights is used to give the structural risk, and the sum of the error terms provides the empirical risk. The weights and error terms are calculated in constraint (7b), while the absolute values of the weights are calculated in constraints (7c)–(7d).

Constraint (7e) ensures that each data point is only associated with one hyperplane. That is, for data point $i \in [I]$, the binary variable $\delta_{i,b}$ is set to 1 if this data point is associated with hyperplane $b \in [B]$. In this case, the values of the weights $w_{b,j}$ (for feature $j \in [n]$) and the error term ξ_i are calculated as in constraint (7b). If $\delta_{i,b} = 0$, then constraint (7b) can essentially be ignored as the calculation would be dominated by the big-$\mathcal{M}$ term, not limiting the setting of the weights and errors.

Constraints (7f)–(7h) ensure the ordering of the data points amongst hyperplanes (as presented by Rebennack and Krasko (2020)). If these constraints were omitted, the formulation could then associate all data points of one class to one hyperplane, and all the other data points to another one. Constraints (7i)–(7k) present the domains of the variables.

Regarding the big-$\mathcal{M}$ terms in constraint (7b) (and constraint (7i)), they should be large enough such that in the case where $\delta_{i,b} = 0$, then ξ_i gets set to 0. That is, we require

$$M_i^1 \geq \max_{b\in[B]} z_i(w_b^\intercal x_i + d_b)$$

for all $i \in [I]$. We can calculate this term for each data point by considering the line connecting that data point to another as the normal vector to a hyperplane. Then, the maximum value of the normal vector will give a bound on the value of M_i^1 for that data point. In particular, suppose the line connecting two points is given by $y = cx + d$; this is equivalent to a hyperplane of $cx_1 - x_2 + d = 0$. Hence, we seek to find the maximum possible value of $z_i(cx_1 - x_2 + d)$ for each data point $i \in [I]$ given by $(x_{i,1}, x_{i,2}, z_i)$.

Let $[\underline{C}, \overline{C}]$ and $[\underline{D}, \overline{D}]$ be extreme values for the gradient and intercept respectively, calculated by interpolating between all possible pairs of data points (see e.g., Rebennack & Krasko, 2020; Warwicker & Rebennack, 2023a for exact derivations of these terms). Then, for $i \in [I]$, we can set:

$$M_i^1 := \max\{|\underline{C}x_{i,1} - x_{i,2} + \underline{D}|, |\underline{C}x_{i,1} - x_{i,2} + \overline{D}|, |\overline{C}x_{i,1} - x_{i,2} + \underline{D}|, |\overline{C}x_{i,1} - x_{i,2} + \overline{D}|\}.$$

Fig. 2 presents an application of Model 1 to the non-separable synthetic data set used in Fig. 1. We are able to see that while the first misclassified data point is still misclassified, the second now lies on the correct side of the hyperplane. From the analysis of the parameter C in Section A of the appendix, we can observe that the empirical risk significantly decreases for each value of C, in comparison with formulation (3). Furthermore, if the objective function is weighted in favour of minimising structural risk (i.e., $C \leq 0.4$), then Model 1 also presents improvements in the structural risk.

We present a further example application of Model 1 to a higher dimension data set (i.e., in three dimensions) in Section B of the appendix. In general, Model 1 is applicable to any dimension (i.e., $n \geq 1$); however, the impact of the non-continuity may be more pronounced in higher dimensions. As a further limitation, we remark that if $B > 2$, there is a risk that a hyperplane will consist only of data points of one class, which will lead to possible instances of overfitting. This can be overcome by including a requirement that each hyperplane must contain at least one data point of each label, in order to ensure a separating hyperplane. Such an implementation is presented in Section C of the appendix, alongside a discussion.

3.2. Model 2: Continuous piecewise linear hyperplanes

As a second contribution towards the presented framework, we present a model for SVM using PWL hyperplanes. While Model 1 is applicable in higher dimensions, the remaining models we present require bivariate data sets (i.e., $n = 2$), due to the complexity in modelling continuities in the hyperplane. We believe that efficient formulations for modelling PWL functions in higher dimensions could be applied within this SVM framework. However, to date, such formulations are heuristic in nature or make limiting assumptions on the shape of the segments.

In order to ensure the (two-dimensional) PWL representation of a hyperplane, we require the model to enforce continuity between adjacent linear segments of the PWL function. This will ensure the data will be (ideally) separated by a single, continuous PWL function. For this purpose, we present an extension of Model 1 which ensures the continuity of the hyperplane. We note again that there is an order requirement on the chosen variable (e.g., time).

To ensure continuity of adjacent segments, we use the modelling formulation that was introduced by Rebennack and Krasko (2020), who presented a MILP model for PWL regression. We note that Kong and Maravelias (2020) presented a similar formulation for PWL regression; however, an analysis showed the former approach is preferable across a number of data sets (Warwicker & Rebennack, 2022). Consider two adjacent hyperplanes (b and b+1), represented by $w_{b,1}x_1 + w_{b,2}x_2 + d_b = 0$ and $w_{b+1,1}x_1 + w_{b+1,2}x_2 + d_{b+1} = 0$ respectively. These two hyperplanes

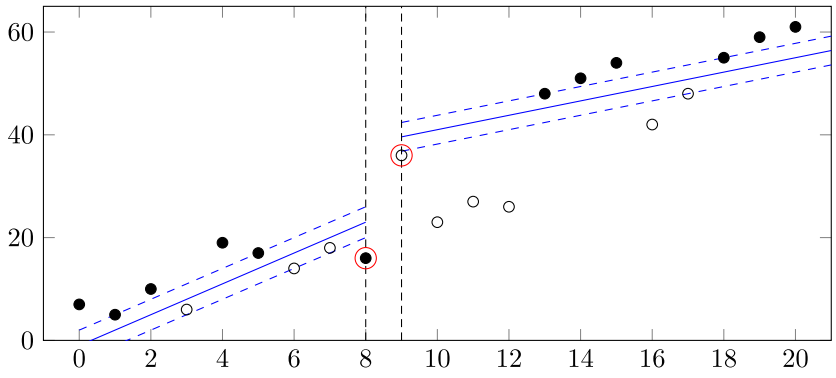


Fig. 2. An example of Model 1 with $B = 2$ applied to a synthetic data set using non-separable data.

intersect at the breakpoint $x_1^\star$, which is calculated as:

$$x_2 = \frac{-w_{b,1}x_1^\star - d_b}{w_{b,2}} = \frac{-w_{b+1,1}x_1^\star - d_{b+1}}{w_{b+1,2}}.$$

$$\implies x_1^\star = \frac{\frac{d_b}{w_{b,2}} - \frac{d_{b+1}}{w_{b+1,2}}}{\frac{w_{b+1,1}}{w_{b+1,2}} - \frac{w_{b,1}}{w_{b,2}}}.$$

Clearly, we require $w_{b,2} \neq 0$, $w_{b+1,2} \neq 0$. The equation above can be interpreted as follows. The numerator represents the change in intercept between adjacent hyperplanes, while the denominator represents the change in gradient. Let us denote the change in gradient as Δ_g. For the two data points either side of the breakpoint $x_1^\star$ (i.e., $x_{i,1}$ and $x_{i+1,1}$ for some $i \in [I]$, where $x_{i,1} \leq x_1^\star \leq x_{i+1,1}$), depending on the sign of Δ_g, we can multiply through by the denominator to attain the following inequalities:

$$\Delta_g > 0 \implies \Delta_g x_{i,1} \leq \frac{d_b}{w_{b,2}} - \frac{d_{b+1}}{w_{b+1,2}} \leq \Delta_g x_{i+1,1};$$

$$\Delta_g < 0 \implies \Delta_g x_{i,1} \geq \frac{d_b}{w_{b,2}} - \frac{d_{b+1}}{w_{b+1,2}} \geq \Delta_g x_{i+1,1}.$$

For the presented model, we assume that $w_{b,2} = w_{b+1,2} > 0$; that is, we replace the variables $w_{b,2}$ with a single, strictly positive variable w_2. Although this comes at some minor loss of generality, it simplifies the modelling required for continuity between adjacent segments. In particular, we can divide the above equations through by w_2, and the breakpoint location between adjacent hyperplanes is given by

$$x_1^\star = \frac{d_b - d_{b+1}}{w_{b+1,1} - w_{b,1}},$$

and hence

$$\Delta_g > 0 \implies (w_{b+1,1} - w_{b,1})x_{i,1} \leq d_b - d_{b+1} \leq (w_{b+1,1} - w_{b,1})x_{i+1,1};$$
$$\Delta_g < 0 \implies (w_{b+1,1} - w_{b,1})x_{i,1} \geq d_b - d_{b+1} \geq (w_{b+1,1} - w_{b,1})x_{i+1,1}.$$

In the former case ($\Delta_g > 0$), the gradient increases between adjacent segments (*i.e.*, there is a *convex* turn); otherwise, the gradient decreases between segments (we assume w.l.o.g. that the gradients do not coincide). Following from Rebennack and Krasko (2020), we introduce a binary variable γ_b ($\forall b \in [B-1]$, where $B \geq 2$ is the number of *breakpoints* of the continuous PWL hyperplane) which indicates the change in gradient between adjacent segments. The binary variable is *activated* when adjacent data points are associated with adjacent segments (*i.e.*, when $\delta_{i,b} = \delta_{i+1,b+1} = 1$.)

We present Model 2 for the fitting of a separating PWL hyperplane (denoted as PWL-SVM$_2$) in formulation (8) , where we remove the subscript on the weight variables for clarity of presentation. Note that we consider a PWL hyperplane with $B-1$ connected segments (i.e., B total breakpoints).

(PWL-SVM$_2$)

$$\min \quad \left(\sum_{b=1}^{B} W_b\right) + w_2 + C\sum_{i=1}^{I}\xi_i \tag{8a}$$

$$\text{s.t.} \quad z_i(w_b x_1 + w_2 x_2 + d_b) \geq 1$$
$$- \xi_i - M_i^1(1-\delta_{i,b}) \qquad \forall i \in [I];\ b \in [B-1] \tag{8b}$$
$$w_b \leq W_b \qquad \forall b \in [B-1] \tag{8c}$$
$$w_b \geq -W_b \qquad \forall b \in [B-1] \tag{8d}$$
$$\text{(7e)–(7h)} \tag{8e}$$
$$\delta_{i,b} + \delta_{i+1,b+1} + \gamma_b - 2 \leq \delta_{i,b}^+ \qquad \forall i \in [I-1];\ b \in [B-2] \tag{8f}$$
$$\delta_{i,b} + \delta_{i+1,b+1} + (1-\gamma_b) - 2 \leq \delta_{i,b}^- \qquad \forall i \in [I-1];\ b \in [B-2] \tag{8g}$$
$$d_{b+1} - d_b \geq x_i(w_b - w_{b+1}) - M_i^2(1-\delta_{i,b}^+) \qquad \forall i \in [I-1];\ b \in [B-2] \tag{8h}$$
$$d_{b+1} - d_b \leq x_{i+1}(w_b - w_{b+1}) + M_{i+1}^2(1-\delta_{i,b}^+) \qquad \forall i \in [I-1];\ b \in [B-2] \tag{8i}$$
$$d_{b+1} - d_b \leq x_i(w_b - w_{b+1}) + M_i^2(1-\delta_{i,b}^-) \qquad \forall i \in [I-1];\ b \in [B-2] \tag{8j}$$
$$d_{b+1} - d_b \geq x_{i+1}(w_b - w_{b+1}) - M_{i+1}^2(1-\delta_{i,b}^-) \qquad \forall i \in [I-1];\ b \in [B-2] \tag{8k}$$
$$\text{(7i)–(7j)} \tag{8l}$$
$$W_b \geq 0 \qquad \forall b \in [B-1] \tag{8m}$$
$$w_b \in \mathbb{R};\ w_2 \in \mathbb{R}_{\neq 0} \qquad \forall b \in [B-1] \tag{8n}$$
$$\gamma_b \in \{0,1\} \qquad \forall b \in [B-2] \tag{8o}$$
$$\delta_{i,b}^+, \delta_{i,b}^- \in [0,1] \qquad \forall i \in [I-1];\ b \in [B-1] \tag{8p}$$

The objective function (8a) again sums the structural and empirical risk. For the calculation of the structural risk, the second margin variable w_2 (which is fixed amongst all segments) only appears once. This is to avoid the inclusion of extra segments contributing excess value to the objective function, despite improving the accuracy of the SVM. This places an increased importance on the first margin variables. Including a penalty term in front of the second margin variable w_2 would provide some balance; however, we prefer to avoid including another penalty term. Constraints (8b)–(8d) act similarly to constraints (7b)–(7d); however, we are now limiting the models to two dimensions.

The continuity of the hyperplanes is implicitly modelled through constraints (8e)–(8k). Constraint (8e) brings in the same constraints (7e)–(7h) from the previous model. However, the constraint (7e) now

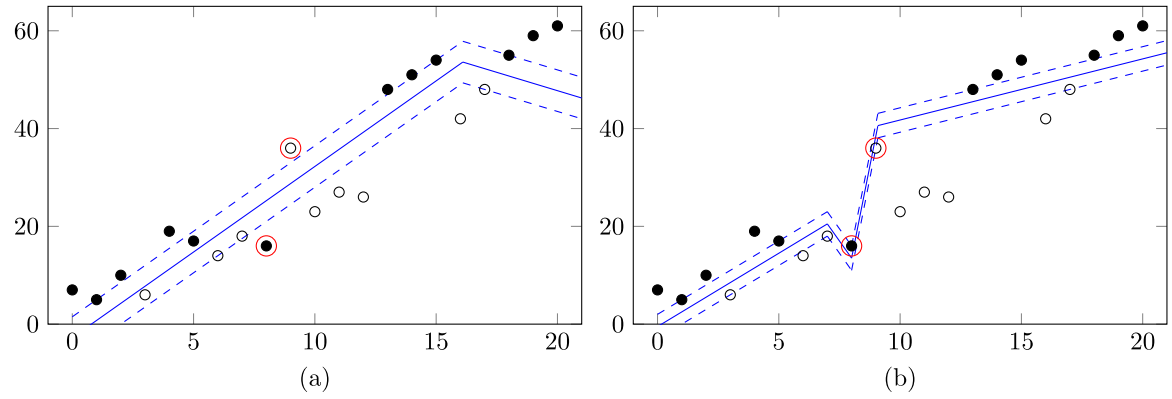


Fig. 3. An example of Model 2 with (a) $B = 2$ and (b) $B = 4$, applied to a synthetic data set.

ensures that each data point is only associated with one segment of the PWL hyperplane, while the constraints (7f)–(7h) act similarly to ensure the correct ordering. Constraints (8f)–(8g) set the value of the binary variable γ_b to 0 or 1 between adjacent segments. This fixes the value of one of the continuous variables $\delta^+_{i,b}$ or $\delta^-_{i,b}$ to 1; note these variables are continuous, and are defined over $[0, 1]$. Depending on which of the variables is set to 1, either constraints (8h)–(8i) or (8j)–(8k) are *activated*, ensuring the continuity as described above. Finally, constraints (8l)–(8p) present the domains of the variables.

Regarding the big-M terms M^2_i, it is required that

$$M^2_i \geq \overline{D} - \underline{D} - \max x_i(w_b - w_{b+1}) = \overline{D} - \underline{D} - x_i(U_1 - L_1),$$

where U_1 and L_1 are respectively the upper and lower bounds for the term w_b (for the given feature). We refer to e.g., Baldomero-Naranjo et al. (2020, 2021) for more information on how to accurately set these bounds.

Fig. 3 shows how Model 2 models the synthetic data set from previous illustrations. We see that while the data is inseparable for linear hyperplanes, we can find more accurate classification using a PWL hyperplane (see Fig. 3a). In Fig. 3b, the PWL hyperplane has entirely eliminated the misclassification of the data points (i.e., $\xi_i = 0, \forall i \in [I]$), allowing the data to be separated. Since these two data points are likely outliers, we note that the presented model is vulnerable to overfitting the data if the number of segments is set too large. We discuss implementing inbuilt outlier detection into the models in Section 4 in order to minimise this risk.

3.3. Model 3: A general model for piecewise linear SVM

For a more general SVM model, we combine the ideas from the previous two formulations. That is, we are looking to fit a given number of continuous, PWL hyperplanes to separate the data. This will overcome the issues of overfitting within the previous models, and allow for more accurate classification. We will separate the data with $K \geq 1$ hyperplanes, over which $B \geq 2K$ breakpoints will be distributed. Model 3, which we present in formulation (9) (and denote as PWL-SVM$_3$), uses similar modelling techniques to the model for clusterwise PWL regression (which was presented by Warwicker and Rebennack (2023b)), in order to model the discontinuities between adjacent hyperplanes.

(PWL-SVM$_3$)

$$\min \quad \left(\sum_{b=1}^{B} W_b\right) + w_2 + C\sum_{i=1}^{I}\xi_i \tag{9a}$$

$$\text{s.t.} \quad \text{(8b)–(8d)} \tag{9b}$$

$$\text{(7e)–(7h)} \tag{9c}$$

$$\sum_{b=1}^{B-K-1} Z_b = K - 1 \tag{9d}$$

$$\delta_{i,b} + \delta_{i+1,b+1} + \gamma_b - 2 \leq \delta^+_{i,b} + Z_b \qquad \forall i \in [I-1]; b \in [B-K-1] \tag{9e}$$

$$\delta_{i,b} + \delta_{i+1,b+1} + (1 - \gamma_b) - 2 \leq \delta^-_{i,b} + Z_b \qquad \forall i \in [I-1]; b \in [B-K-1] \tag{9f}$$

$$\text{(8h)–(8k)} \tag{9g}$$

$$\text{(7i)–(7j)} \tag{9h}$$

$$\text{(8m)–(8p)} \tag{9i}$$

$$Z_b \in \{0, 1\} \qquad \forall b \in [B-K-1] \tag{9j}$$

The objective function, the calculation of weights and errors, and the assignment and ordering of the binary variables act similarly to the previous models, and are covered by the terms (9a)–(9c).

Constraints (9d)–(9f) model the discontinuity between adjacent hyperplanes. Firstly, the binary variable Z_b is set to 1 if the breakpoint b is the last breakpoint in its hyperplane; constraint (9d) ensures this only occurs for the given number of hyperplanes. Then, if $Z_b = 1$, constraints (9e) and (9f) can be ignored; otherwise, the are enacted and the continuity is enforced between the current and adjacent segment (via constraints (8h)–(8k), included by constraint (9g)).

Finally, the remaining constraints (9h)–(9j) provide the variable domains.

We note that once the model has been run and the PWL segments have been assigned, it is possible to re-run Model 2 on the newly assigned separated data sets to re-fit the connected PWL hyperplanes. This will then allow the user to assign different margins to each hyperplane, further optimising for structural risk. However, we note that this may come at a small loss of generality and optimality.

In Fig. 4, we present a comparison between Model 2 and Model 3 on a new synthetic data set. Both models use three segments, yet Model 3 distributes the three segments between two hyperplanes, allowing for a somewhat more natural separation. Data sets from time-series analysis often exhibit such discontinuities, in which connected PWL hyperplanes may not be sufficient to optimally separate and classify the data. We again note the possible vulnerability of the presented formulation for overfitting the data.

3.4. Comparison of formulations

Fig. 5 presents a flowchart representation of the three presented models, detailing the relationships between subsequent formulations.

Further, in Table 1, we compare the three formulations we have presented in this section. For the sake of comparison, we consider all formulations for bivariate data (i.e., we set $n = 2$ for formulation (7)). Note

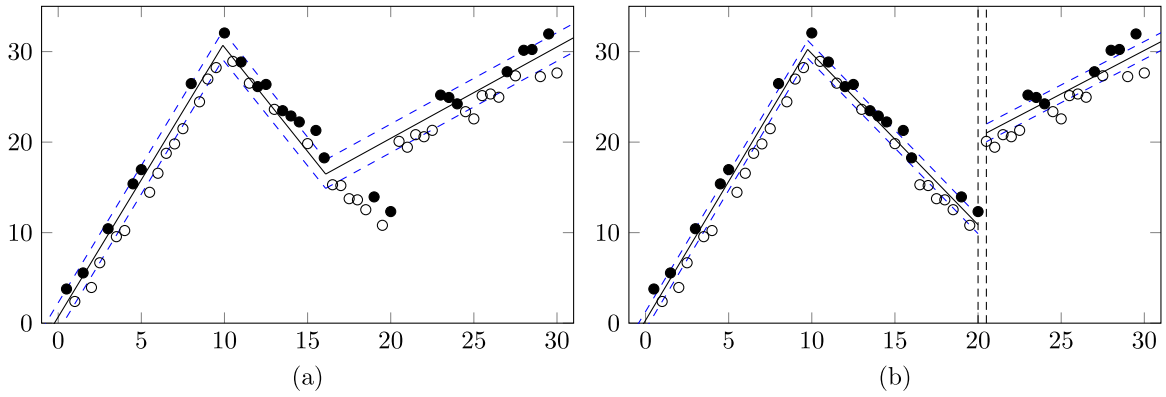


Fig. 4. An example of (a) Model 2 and (b) Model 3 applied to a synthetic data set.

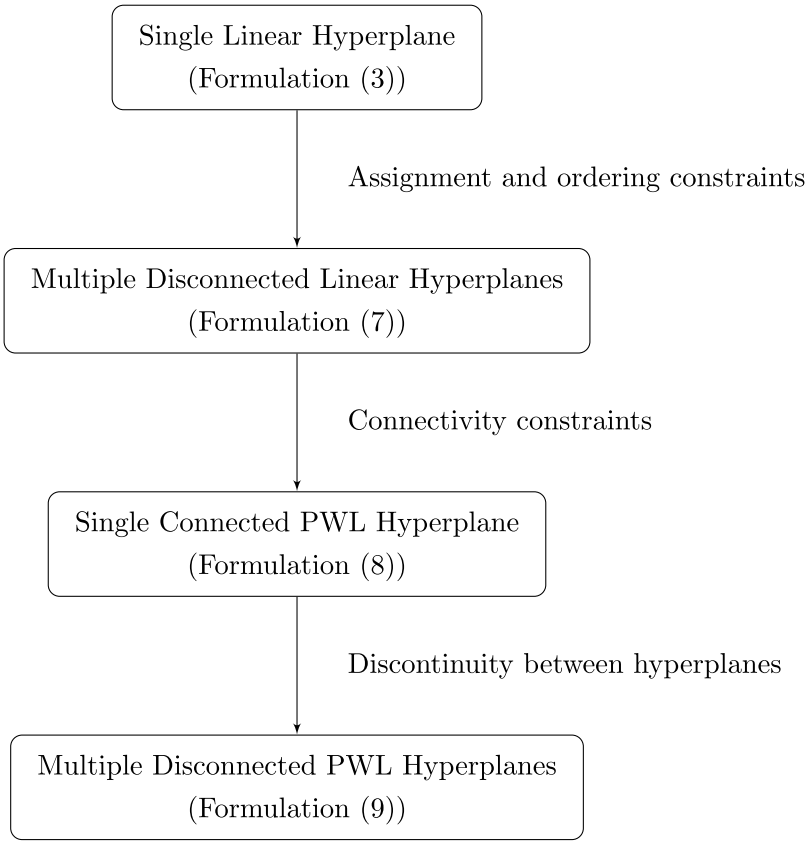


Fig. 5. Flowchart describing the relationship between the presented models.

Table 1
A comparison of Models 1–3.

Formulation	Model 1	Model 2	Model 3
Constraints	(7a)–(7k)	(8a)–(8p)	(9a)–(9j)
Nature of hyperplanes	Multiple linear	Piecewise linear	Multiple piecewise linear
Nature of continuity	Discontinuous	Continuous	Mixed
# Binary variables	$I \cdot B$	$(I+1) \cdot B - I - 2$	$(I+2) \cdot B - I - K - 3$
# Continuous variables	$4 \cdot B + I$	$(2I+1) \cdot B - I - 1$	$(2I+1) \cdot B - I - 1$
# Constraints	$(2I+3) \cdot B + 2 \cdot I - 1$	$(8I-5) \cdot B - 11 \cdot I + 9$	$(8I-5) \cdot B - 11 \cdot I + 10$

that while B represents the number of segments in formulation (7), it represents the number of breakpoints in formulations (8)–(9).

Regarding the number of binary variables, Model 2 introduces $B-2$ binary variables to model the change in gradient between adjacent segments. Model 3 further introduces $K-1$ binary variables to model the discontinuity; however, the removal of the continuity aspect between adjacent segments makes the formulation easier to solve.

Regarding the application of the three models, they can each be preferable in different situations. Models 1 and 3 contain discontinuities, which means for future data points lying in these areas, the models may have to be re-solved in order to provide more accurate

hyperplanes. This also holds for data points lying outside the domain of the pre-existing data set.

In general, we advise practitioners to perform a preliminary analysis of the data set before selecting which of these models to apply. For example, if there is a known discontinuity in the data which is to be modelled, we advise implementing either Model 1 or Model 3, where the latter may provide more information in more complex data sets. Otherwise, we advise the use of Model 2, which can provide more information than a standard model for linear SVM (e.g., compared to formulation (3)).

4. SVM models within the unified framework for machine learning problems

The MILP models for SVM we have presented so far fit within the same framework, in that they use binary variables to assign data points to segments of hyperplanes, and model logical implications through big-$\mathcal{M}$ constraints. In particular, they share a common makeup with the existing framework of MILP models for machine learning problems, which has recently been presented by Warwicker and Rebennack (2023b). This framework has incorporated models for supervised and unsupervised learning problems, such as regression and clustering (Park et al., 2017; Rebennack & Krasko, 2020). The goal of these models is to fit PWL-based regression functions to predict trends and groupings based on unlabelled data. We now show how the SVM models presented so far in this work can also fit within this framework, showcasing its generality even for supervised learning approaches and for fitting hyperplanes.

The idea of the unifying framework is that tailored methods can be employed (based on the specific knowledge of the structure of the models) to make them more robust, and to improve their efficiency. In this section, we discuss how the structure of the presented SVM models can be utilised in order to implement outlier detection. Additionally, we present the use of feature selection for the presented SVM models, which can also be implemented as an improvement within the framework.

For models within this framework, the combinatorial Benders decomposition introduced by Codato and Fischetti (2006) can typically be used to find speedups in the solution process. However, this is dependent on the choice of the objective function. For the application to SVM, typical objective functions render this approach ineffective. In particular, any objective functions seeking to minimise the maximum error, which typically benefit from CBD approaches, are ineffective at measuring the empirical risk.

4.1. Outlier detection

The models we have presented in Models 1–3 are able to optimally separate data that cannot be classified accurately within a linear SVM model framework. However, in some cases, the MILP models can lead to hyperplanes that are susceptible to overfitting the data. In order to ensure the robustness of the models, we can implement inbuilt outlier detection such that a fixed number of data points are excluded from the calculation of the optimal hyperplane. Such outliers may be points which have been mislabelled, or correctly labelled points which have been subject to error in their measurements. If the outliers are not taken into account, the models will provide hyperplanes that better fit the non-outlier data points.

In terms of implementing outlier detection within MILP models, Sudermann-Merx and Rebennack (2021) suggested the use of statistical models to identify sets of possible outlier points (based on their relative position to the remaining data points), leading to efficient approximations. Since the distance metric they considered is immune to y-outliers, the outliers were likely located in extreme positions in the x-axis. Hence, the idea of their approach is to reduce the number of binary variables used in the MILP formulation.

Recently, Warwicker and Rebennack (2023a) considered all data points as possible outliers, allowing the model to exclude a given number of data points from the calculation of the objective function. This ensures an optimal removal of outlier points. A similar approach has been implemented within SVM models with ramp loss by Brooks (2011).

We present the following extension to Model 3, noting that a similar extension can be implemented within the other models within the presented framework, as well as the existing models from the literature (i.e., formulations (1)–(6)). The binary variable ρ_i states that the data point $i \in [I]$ is included in the model (i.e., it is not an outlier point), and contributes towards the objective function. Let $Q \geq 0$ be the number of outlier points to be excluded by the model (this is set by the user). Naturally, Q should not be set too large, which can lead to loss of information. We recommend a trial-and-error based approach if the number of outliers in the data is not known. Alternatively, implementing a (bounded) loss function within the objective can be useful, although this may come at the cost of non-linearity (and non-convexity) (Collobert et al., 2006).

(PWL-SVM$_{\text{O}}$)

$$\min \quad \left(\sum_{b=1}^{B} W_b\right) + w_2 + C\sum_{i=1}^{I}\xi_i \tag{10a}$$

$$\text{s.t.} \quad z_i(w_b x_1 + w_2 x_2 + d_b) \geq 1$$

$$-\xi_i - M_i^1(2 - \delta_{i,b} - \rho_i) \qquad \forall i \in [I];\ b \in [B] \tag{10b}$$

$$w_b \leq W_b \qquad \forall b \in [B] \tag{10c}$$

$$w_b \geq -W_b \qquad \forall b \in [B] \tag{10d}$$

$$\sum_{i=1}^{I}\rho_i = I - Q \tag{10e}$$

$$\rho_i \in \{0,1\} \qquad \forall i \in [I] \tag{10f}$$

(9c)–(9j) (10g)

The objective function (10a) calculates the structural and empirical risk as with the previous formulations. However, constraint (10b) is now only *activated* if the given data point is assigned to the segment, and is not an outlier point (i.e., $\delta_{i,b} = \rho_i = 1$ for the given data point $i \in [I]$ and segment $b \in [B]$). The absolute values of the weight are calculated in constraint (10c)–(10d), while constraint (10e) ensures only Q data points are considered as outliers. Constraint (10f) gives the domain of the newly introduced binary variables, while the remaining constraints match that of the model for which they are applied (e.g., formulation (9) in this instance).

Formulation (10) adds I extra binary variables within the model, which increases its complexity. We can note that if we were able to rule out some data points as outliers (possibly by some heuristic method or statistical analysis), then their value for ρ_i could be fixed to 1, reducing the complexity of the model. However, this may sacrifice some accuracy and the resultant solution may not be optimal (with regards to the given objective function).

In Fig. 6, we present an example of inbuilt outlier detection embedded within Model 2, for the same synthetic data set presented in Fig. 3. As the number of outliers increases, so does the quality of the model. We see that if $Q = 2$, both outlier points are correctly identified (and excluded), ensuring an optimal separation (i.e., the empirical risk is 0) and the maximisation of the margin.

4.2. Feature selection

We now discuss a simplified implementation of feature selection that is applicable for the three models we have presented. The implementation we present can also be generalised to be applicable

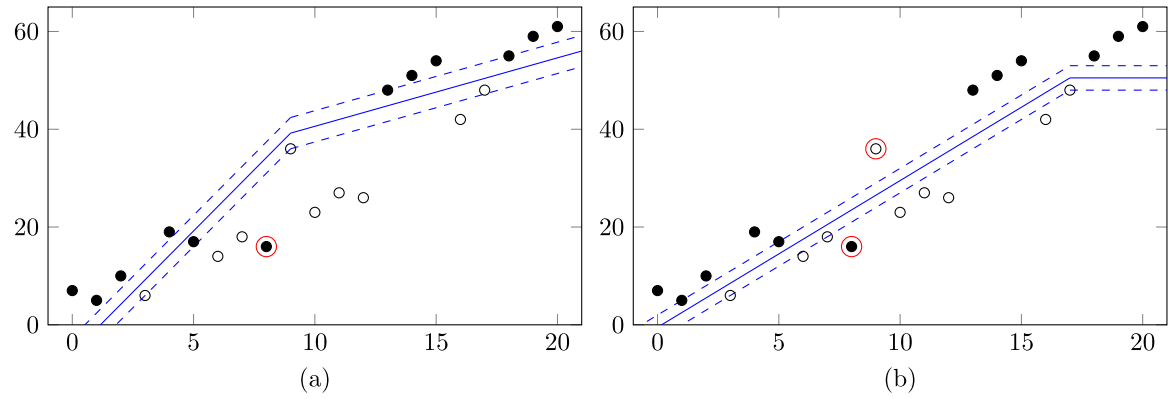


Fig. 6. An example of Model 2 applied to a synthetic data set with $B = 3$ and (a) $Q = 1$, (b) $Q = 2$. The circled data points have been excluded from the calculation of the objective function.

to the previous models that have been presented within the framework (i.e., clustering and regression problems Warwicker & Rebennack, 2023b).

Firstly, we note that the models we have presented (in particular, Models 2 and 3) seek to separate labelled data in $\mathbb{R}^2$. Therefore, any feature selection is limited to selecting only two features. Secondly, there is a requirement (to preserve the continuity in the modelled hyperplanes) that one of the features must be ordered. We suggest that this feature can be fixed by the user, and is usually the most important feature. For example, this can correspond to the *time* in the case of time-series data.

The model we present is an extension of Model 3, and is based on formulation (5) which was first presented by Maldonado et al. (2014). We fix the first feature, and for each subsequent feature $j \in [J]$, the binary variable v_j states whether or not that feature is selected by the model. As opposed to formulation (5), we are assuming that the cost of acquiring each feature is equal, and that we are *selecting* at most one. This replaces the *budget function*. As with formulation (5), we also are not aiming to explicitly minimise the structural risk, since we require the weights to fall within a given interval $[L_j, U_j]$ for each feature $j \in [J]$; this implicitly keeps the margins small.

(PWL-SVM$_{\text{FS}}$)

$$\min \quad \sum_{i=1}^{I} \xi_i \tag{11a}$$

$$\text{s.t.} \quad z_i \left(w_b x_1 + \left(\sum_{j=1}^{J} w_j x_2 \right) + d_b \right) \geq 1 - \xi_i - M_i^1 (1 - \delta_{i,b}) \qquad \forall i \in [I]; b \in [B] \tag{11b}$$

$$L_j v_j \leq w_j \leq U_j v_j \qquad \forall j \in [J] \tag{11c}$$

$$\sum_{j=1}^{J} v_j = 1 \tag{11d}$$

$$v_j \in \{0, 1\} \qquad \forall j \in [J] \tag{11e}$$

(9c)–(9j) (11f)

The objective function (11a) minimises the empirical risk, calculated as the sum of the errors. Constraint (11b) again sets the values of the weights and calculates the errors in the case that the given data point $i \in [I]$ is assigned to the segment $b \in [B]$. Constraint (11c) only allows the weights to be set as a non-zero value if the binary variable $v_j = 1$ for the feature $j \in [J]$, in which case they are limited to lie within the range $[L_j, U_j]$. Constraint (11d) limits only one feature to be selected for the given model, while constraint (11e) gives the domain of the new binary variables. The remaining constraints again match those of the model to which they are applied (e.g., formulation (9)).

We are further assuming that the weights for the features than can be selected are non-negative. That is, we can set $L_j = 0$ (or to a small constant $\epsilon > 0$) for all $j \in [J]$, while setting U_j to be arbitrarily large. Of course, setting the value too large can lead to some loss of performance in branch-and-bound settings. We again refer to Baldomero-Naranjo et al. (2020, 2021) for more information on how to accurately set these bounds.

Formulation (11) *selects* the feature that leads to a SVM with the best *fit* for the given model (measured by the value of the given objective function). That is, the model seeks to find the most relevant feature that leads to an easily distinguishable split between the two sets of labelled data, and hence the best accuracy for future predictions. By fixing the value of v_j to 1 for each feature $j \in [J]$, the model can also present a ranked list of how *valuable* each feature is with regard to SVM predictions.

Finally, we note that as with the models presented by Baldomero-Naranjo et al. (2021), feature selection and outlier detection can be implemented simultaneously within the models we present, as well as those presented in Warwicker and Rebennack (2023b).

5. Experimental analysis

We now present a series of experimental comparisons of the formulations presented so far, to highlight when each one is advantageous.

5.1. Bivariate ad-hoc data sets

We firstly present a series of comparisons on ad-hoc data sets, which have been designed to showcase exactly when each of the formulations performs the best, in terms of solution quality and efficiency. In each case, a (possibly interrupted, piecewise) linear function is chosen, and Gaussian noise is introduced. Aside from the fourth data set, the labels are assigned based on proximity to the value of the function without any noise. For each data set, 50 data points are taken.

1. The first data set allows a linear SVM to be fitted;
2. The second data set allows a PWL SVM to be fitted, but is non-separable for a linear SVM;
3. The third data set allows multiple PWL SVMs to be fitted, but is prohibitive for a single linear or PWL SVM;
4. The fourth data set shows the same data as the second, but with the labels randomly assigned. This is designed to see how the models perform on random (and mislabelled) data.

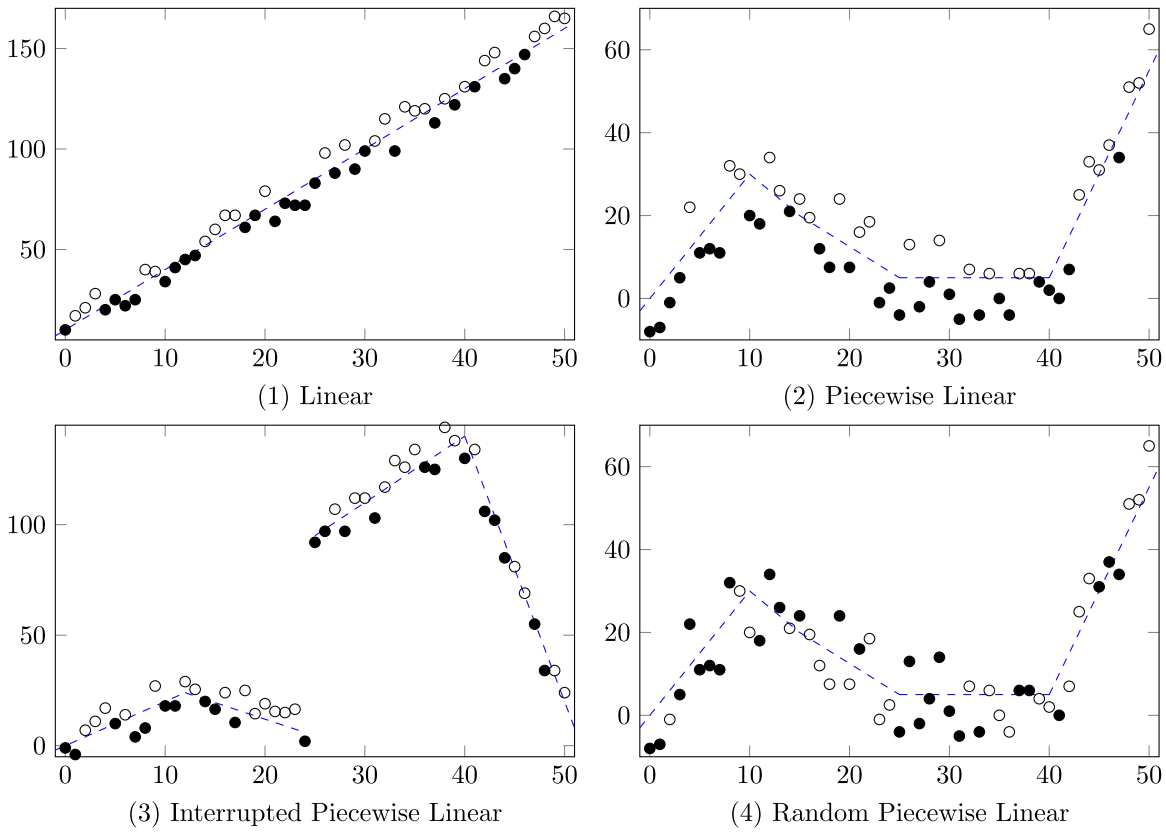


Fig. 7. Four ad-hoc data sets designed to showcase the advantages of the presented formulations. The blue dashed line indicates the function used as a basis for the given data labelling.

Table 2
Reference values obtain by formulation (3) (Bradley & Mangasarian, 1998) for the presented ad-hoc data sets.

Data set	1	2	3	4
OF value	5.49	62.18	138.00	127.82
Structural risk	5.49	0.18	0	0.06
Empirical risk	0	20.67	46.00	42.58
Runtime	0.01	0.01	0.01	0.01
% Misclassified	0	37.25	45.10	80.39

The four data sets are presented in Fig. 7.

We use $C = 3$ throughout this subsection, as (from preliminary experiments) this value ensures that when the data set is separable (with regards to the chosen SVM model), the empirical risk is set to 0 (in the majority of cases), and the structural risk is minimised. In particular, we firstly wish for the models to find an exact separation (where this is possible) by minimising the empirical risk. Once the data is able to be separated, we then wish the models to find the best fit by minimising the structural risk. Hence, a large value of C (i.e., greater than 1) fits this. However, for real-world data, we expect the data to be non-separable, so we recommend a value of $C \leq 1$ in this case.

Firstly, as a means of a benchmark comparison, we present the runtime and risk values for the standard SVM model (formulation (3), presented by Bradley and Mangasarian (1998)) in Table 2. This formulation acts equivalently to Models 1–3 if the number of segments (and clusters) are set to 1.

We see that while this formulation is able to find an optimal separation and minimise the empirical risk for data set (1), the empirical risk for the remaining data sets is very high, leading to large objective function values. Further, many data points are misclassified. This suggests that linear SVMs may not provide the necessary information for data sets that exhibit the patterns shown in data sets (2) and (3). In particular, we note that the objective function when formulation (3) is applied to data set (4) is lower than when applied to data set (3), highlighting further the need for more accurate SVM models when the data set exhibits a non-linear separation.

In Tables 3–4, we present the result for models (1)–(3) across the four data sets. When solving for model (3), we are assuming two clusters.

Regarding data set (1), we see that the three presented models are able to find SVMs that separate the data. Model 1 is able to find small improvements as the number of segments increases, while the runtime also increases. Hence, there is a tradeoff between desired performance and runtime. For a larger number of segments, there is a risk of overfitting the available data, meaning potential loss of accuracy for classifying future data points. Models 2 and 3, however, cannot find improvements with an increased number of segments due to the requirements on continuity, so there is no advantage to SVM models with more segments in these cases.

For data set (2), we see that all models are able to separate the data when at least four segments are used (implying an advantage over single-segment models, such as formulation (3)). As the number of segments increases beyond 4, there is a similar pattern as for data set (1); that is, Model 1 can find small improvements in the structural risk, while Models 2 and 3 are constrained by continuity. We further see that Model 2 requires at least four segments to separate the data, while Models 1 and 3 require only 3. Although the value of the objective function is higher for Model 2, we note that the information provided by the breakpoint locations is also valuable, as we can identify points where the relationship between the given variables may change.

For data set (3), the advantage of the presented models over formulation (3) is again evident. When Models 1 and 3 consist of at least four segments, the empirical risk is small (a larger value for C would result

Table 3
Results of Models (1)–(3) applied to data sets (1) and (2).

		Data set (1)						Data set (2)					
		Segments						Segments					
		2	3	4	5	6	7	2	3	4	5	6	7
Model 1	OF value	5.34	5.12	5.12	4.19	3.90	3.31	18.21	3.40	1.51	1.40	1.29	1.10
	Structural risk	5.34	5.12	5.12	4.19	3.90	3.31	2.78	3.40	1.51	1.40	1.29	1.10
	Empirical risk	0	0	0	0	0	0	5.14	0	0	0	0	0
	Runtime	0.03	0.05	0.13	0.28	0.61	1.86	0.03	0.06	0.11	0.14	0.25	0.48
	% Misclassified	0	0	0	0	0	0	7.84	0	0	0	0	0
Model 2	OF value	5.49	5.49	5.49	5.49	5.49	5.49	17.96	8.47	3.82	3.82	3.82	3.82
	Structural risk	5.49	5.49	5.49	5.49	5.49	5.49	2.46	2.66	3.82	3.82	3.82	3.82
	Empirical risk	0	0	0	0	0	0	5.17	1.94	0	0	0	0
	Runtime	0.02	0.11	0.36	1.42	7.30	29.22	0.03	0.13	0.31	1.31	10.75	119.31
	% Misclassified	0	0	0	0	0	0	5.88	3.92	0	0	0	0
Model 3	OF value	5.12	5.12	5.12	5.12	5.12	5.12	17.96	2.58	2.58	2.58	2.58	2.58
	Structural risk	5.12	5.12	5.12	5.12	5.12	5.12	2.46	2.58	2.58	2.58	2.58	2.58
	Empirical risk	0	0	0	0	0	0	5.17	0	0	0	0	0
	Runtime	0.02	0.11	0.25	0.80	8.25	34.58	0.03	0.13	0.41	2.75	20.66	138.31
	% Misclassified	0	0	0	0	0	0	7.84	0	0	0	0	0

Table 4
Results of Models (1)–(3) applied to data sets (3) and (4).

		Data set (3)						Data set (4)					
		Segments						Segments					
		2	3	4	5	6	7	2	3	4	5	6	7
Model 1	OF value	86.62	58.07	16.79	11.09	4.27	3.44	96.09	74.36	56.41	44.08	32.68	21.68
	Structural risk	2.62	11.81	10.79	6.29	4.27	3.44	0.42	0.59	2.46	2.77	3.37	4.37
	Empirical risk	28.00	15.42	2.00	1.60	0	0	31.89	24.59	17.99	13.77	9.77	5.77
	Runtime	0.02	0.05	0.09	0.17	0.42	1.28	0.05	0.06	0.17	0.38	0.56	2.75
	% Misclassified	27.45	31.37	1.96	1.96	0	0	56.86	43.14	33.33	17.64	9.80	5.88
Model 2	OF value	128.00	121.15	76.41	39.90	37.75	34.93	121.05	105.18	94.13	86.73	80.13	76.50
	Structural risk	2.00	1.94	12.31	28.42	20.55	19.90	0.11	2.23	4.13	6.33	8.13	8.25
	Empirical risk	42.00	40.07	21.37	3.83	5.73	5.00	40.31	34.32	30.00	26.80	24.00	22.75
	Runtime	0.05	0.36	0.83	5.14	38.05	778.19	0.05	0.38	3.89	16.89	121.28	1498.47
	% Misclassified	41.18	72.55	43.14	13.73	15.69	15.69	56.86	58.82	45.10	45.10	41.18	37.25
Model 3	OF value	119.05	67.16	16.38	16.38	16.38	16.30	103.18	92.13	84.73	78.13	74.50	68.00
	Structural risk	0.18	3.74	15.18	15.18	15.18	14.30	0.23	2.13	4.33	6.13	6.25	8.00
	Empirical risk	39.63	21.14	0.40	0.40	0.40	0.67	34.32	30.00	26.80	24.00	22.75	20.00
	Runtime	0.05	0.14	0.52	1.70	19.00	170.00	0.05	0.19	1.38	11.64	113.00	487.95
	% Misclassified	27.45	41.18	3.92	1.96	1.96	1.96	56.86	49.02	43.13	31.37	23.53	21.57

in a value of 0), and we see that further segments can lead to small improvements in both cases. In this case, Model 2 is prohibited by the continuity requirements, and cannot effectively model the discontinuity present in the data set. In such cases, the generality of Model 3 is desirable, since it can effectively model the discontinuity, while also providing breakpoint information for the two separated clusters of data.

We see for data set (4) that all models are ineffective, and lead to large objective function values when there is no apparent pattern in the data. Although the runtime required by Models 2 and 3 increases quickly as the number of segments increases, the large objective function value is a desirable property of the models, since we can identify that the given features lead to *worse* SVMs. Hence, if feature selection was implemented, the formulations are able to identify which features lead to poor SVM models such as those present in this data set, and which may lead to better SVM models (i.e., any features with a that lead to a resemblance with data sets (1)–(3)).

In general, we see that as the objective function improves, the percentage of misclassified points decreases. On the few occasions where this does not hold, we note that the models are not performing well due to the limited number of linear segments used. Further, in these cases we see the empirical risk decreases, meaning the magnitude of the error of the misclassified points is decreasing overall.

5.2. Real world data sets

We now present a comparison of the three models on four real-world data sets taken from the UCI Machine Learning Repository (Dua & Graff, 2017), in order to assess their applicability, and show when each can be useful from an analytical perspective. The data sets are typically used in SVM applications for prediction, and exhibit a number of the properties highlighted in the previous section. In this section (and the analysis of outlier detection in Section 5.3), we firstly select, for each data set, two features for the application of the three models — we later analyse the effectiveness of the inbuilt feature selection in Section 5.3. We present the full information about the data sets in Table 5, and plot example applications of Model 2 with 4 breakpoints in Fig. 8. Throughout this section, we use $C = 1$.

Table 5
Real world data sets used for experimental comparison of SVM models. We consider only non-categorical features.

Data set	Test	Fixed variable	# Features	Training set size
Adult	Earnings (≤50k?)	Age	3	73
Credit	Credit rating	Unknown	4	138
Heart	Heart disease?	Age	12	134
WBC	Breast cancer?	Unknown	25	122

Firstly, Table 6 shows the reference values obtained from the application of formulation (3) to the four data sets. In each case, the empirical risk is very large as the data is not necessarily linearly separated. However, this formulation is very fast due to the lack of binary variables. By implementing feature selection within formulation (3), the model will seek to find features that portray the best linear separation. If such a linear separation is unavailable, a PWL separation may be preferable.

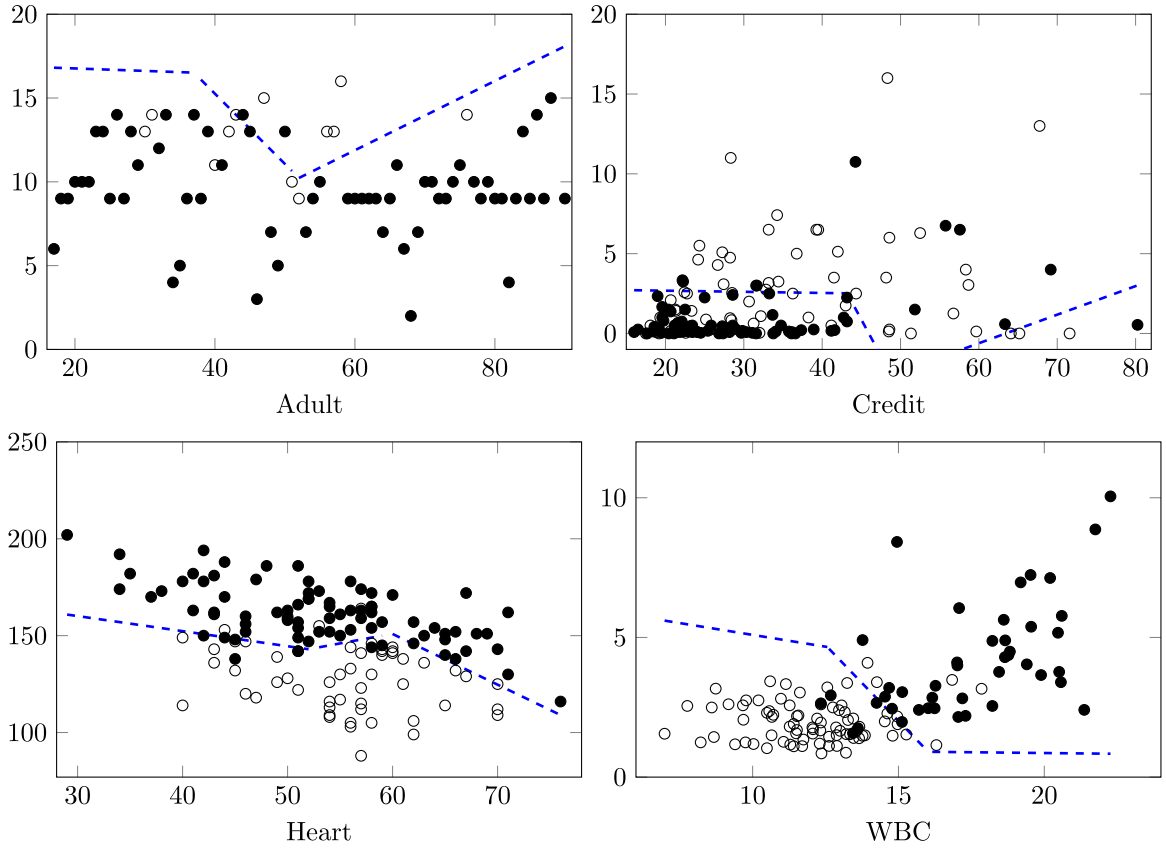


Fig. 8. Four real-world data sets designed to showcase the advantages of the presented formulations. Each model is presented alongside a PWL SVM (from Model 2) with 4 breakpoints.

Table 6
Reference values obtain by formulation (3) (Bradley & Mangasarian, 1998) for the presented real-world data sets.

Data set	Adult	Credit	Heart	WBC
OF value	30.95	112.50	46.07	30.88
Structural risk	0.06	0.45	0.18	1.19
Empirical risk	30.88	112.04	45.89	29.69
Runtime	0.01	0.01	0.01	0.01
% Misclassified	16.44	69.78	33.58	23.58

Tables 7–8 show the results of Models 1–3 applied to the four real-world data sets. Model 1 is the fastest model overall, and produces the lowest objective function values. However, the non-connectivity exhibited can be disadvantageous, since it will not be clear where the trends change.

Regarding the performance of Model 2 on these data sets, the runtimes quickly increase due to the complication of finding a continuous PWL function. In particular, for the Heart and WBC data sets, the runtime exceeds the time limit (86,400 s = 1 day) with only 6 segments. Further, for these data sets, the decrease in objective function value is small for increased number of segments. For the Adult data set, the inclusion of more linear segments can lead to significant increases in quality (e.g., comparing the model applied with 3 and 4 segments). We note that after such an improvement, there is only minor improvement seen from further segments; hence, we suggest that 4 segments is the best in this case to find the right balance between solution quality and potential overfitting.

Model 3 exhibits similar properties to Model 2 on the real-world data sets with a slight increase in solution quality (measured by the objective function value). Model 3 can be seen as midway between Models 1 and 2, where there is a tradeoff between solution quality and efficiency, and the information provided through continuity. Naturally, enforcing continuity in all places can lead to overfitted models, so requiring a small number of disconnected (PWL) hyperplanes in the SVM model can be advantageous.

The models themselves require the number of segments and hyperplanes to be given as input. We have seen that this selection can be crucial with regard to the quality of the fit (measured by the objective function value). While testing multiple settings for these parameters can be prohibitive for large-scale data sets, we note that a preliminary analysis of the data set may provide some indication as to the most natural parameter settings. For example, heuristic approaches may indicate when an extra segment or hyperplane leads to a large increase in the quality of the solution.

5.3. Enhancements within the framework

We have seen from Sections 5.1–5.2 where the three models we have presented can be applied most effectively, and the relative advantages and disadvantages of each one. Nevertheless, the models can be improved by taking advantage of their structure (Warwicker & Rebennack, 2023b); see Section 4 for more details. In particular, we will show how outlier detection can be used to improve the quality of the model.

Table 7
Results of Models 1–3 applied to the real-world data sets.

		Adult						Credit					
		Segments						Segments					
		2	3	4	5	6	7	2	3	4	5	6	7
Model 1	OF value	16.69	14.19	10.64	8.64	7.14	5.74	80.78	73.38	69.37	66.17	63.37	60.17
	Structural risk	1.54	3.04	3.69	4.58	3.08	3.07	0.68	0.82	0.85	1.66	0.85	1.66
	Empirical risk	15.15	11.15	10.64	4.06	4.06	2.67	80.10	72.56	68.52	64.52	62.52	58.52
	Runtime	0.03	0.11	0.25	0.53	0.58	1.20	0.06	0.28	4.41	21.64	119.06	685.84
	% Misclassified	12.33	9.59	6.85	4.11	4.11	2.74	51.80	38.85	36.69	35.25	34.53	33.09
Model 2	OF value	20.50	20.02	17.71	17.28	15.35	14.56	97.97	94.01	93.14	90.96	90.49	⋆
	Structural risk	0.58	0.61	2.78	3.68	4.43	5.58	0.60	0.85	1.03	2.81	2.92	⋆
	Empirical risk	19.92	19.41	14.94	13.60	10.92	8.98	97.37	93.15	92.11	88.15	87.57	⋆
	Runtime	0.08	0.52	4.52	18.52	92.52	431.78	0.17	2.20	117.17	1233.69	85,203.90	⋆
	% Misclassified	24.66	24.66	19.18	17.81	14.70	12.33	69.06	65.47	64.75	61.87	57.97	
Model 3	OF value	17.91	15.99	14.89	12.83	11.43	10.29	92.24	90.02	88.83	86.89	86.42	⋆
	Structural risk	1.18	1.85	2.09	3.83	4.72	5.21	0.48	3.11	1.48	5.35	5.47	⋆
	Empirical risk	16.73	14.13	12.81	9.00	6.71	5.08	91.76	86.91	87.35	81.55	80.96	⋆
	Runtime	0.05	0.27	1.38	8.63	41.80	130.86	0.06	1.42	23.11	589.28	49,942.73	⋆
	% Misclassified	13.70	10.96	9.59	6.85	5.48	5.48	53.62	51.45	52.17	47.10	45.65	⋆

Table 8
Results of Models 1–3 applied to the real-world data sets (cont.).

		Heart						WBC					
		Segments						Segments					
		2	3	4	5	6	7	2	3	4	5	6	7
Model 1	OF value	32.06	28.28	25.57	22.01	19.30	16.27	26.67	22.87	20.87	20.05	18.32	17.51
	Structural risk	1.75	3.21	3.84	4.81	5.44	3.47	0.42	2.22	2.22	1.32	0.32	2.87
	Empirical risk	30.31	25.07	21.74	17.20	13.87	12.8	26.25	20.64	18.64	18.74	18.00	14.64
	Runtime	0.08	0.13	0.75	4.47	12.56	14.47	0.05	0.14	0.45	2.97	7.86	12.78
	% Misclassified	21.64	16.42	14.18	11.19	8.96	8.21	17.07	11.38	10.57	10.57	7.32	8.94
Model 2	OF value	36.86	35.21	34.95	34.74	⋆	⋆	28.94	28.93	28.92	28.92	28.91	⋆
	Structural risk	0.27	0.48	0.70	1.16	⋆	⋆	0.06	0.17	0.19	0.19	0.18	⋆
	Empirical risk	36.59	34.73	34.25	33.57	⋆	⋆	28.88	28.76	28.73	28.73	28.73	⋆
	Runtime	0.09	2.42	111.67	10,894.61	⋆	⋆	0.11	1.11	18.89	570.67	20,959.16	⋆
	% Misclassified	25.37	23.88	23.13	23.13	⋆	⋆	22.76	21.95	21.95	21.95	21.95	⋆
Model 3	OF value	34.40	33.97	33.20	32.77	⋆	⋆	26.54	26.46	26.46	26.46	26.44	⋆
	Structural risk	0.41	0.49	1.22	1.30	⋆	⋆	0.12	0.16	0.16	0.16	0.15	⋆
	Empirical risk	34.00	33.48	31.98	31.47	⋆	⋆	26.42	26.31	26.30	26.30	26.29	⋆
	Runtime	0.08	1.53	30.94	1504.59	⋆	⋆	0.06	0.58	10.20	231.01	7301.59	⋆
	% Misclassified	24.63	22.39	20.90	20.90	⋆	⋆	19.51	19.51	19.51	19.51	19.51	⋆

Further, we show how feature selection can be implemented within the models, leading to results generalisable to higher dimensional data sets.

Outlier detection

Section 4.1 highlighted how inbuilt outlier detection can be used to improve the quality of the presented models. We show that although the inclusion of outlier detection leads to slower models, the increase in quality can be significant. Hence, there is a tradeoff between improved models and efficiency; further, the inclusion of too many outlier points can lead to models that overfit the given data. In Table 9, we implement outlier detection within Model 1 to highlight this.

We see from Table 9 that the inclusion of inbuilt outlier detection can have a significant impact on the quality of solutions. In particular, there are cases where the removal of an outlier has a larger effect on the objective function value than the inclusion of an extra linear segment (e.g., Adult data set, 4 segment and 1 outlier vs. 5 segments and 0 outliers). Of course, implementing inbuilt outlier detection leads to much longer solve times, yet is particularly important in SVM applications due to potential misclassifications (possibly due to human error). In general, the removal of outliers leads to much more robust models.

Feature selection

We have seen from Section 4.2 how feature selection can be implemented within the models we have presented, using constructs presented by e.g., Maldonado et al. (2014). The goal of feature selection is to identify features that affect the model the most in order to improve the quality of the model. In Table 10, we analyse how feature selection affects the efficiency of Model 1. Note that in each case, one feature is *fixed*, and we are assessing the effectiveness of choosing the second feature.

We see from Table 10 that inbuilt feature selection can lead to longer solve times, due to the inclusion of further binary variables and more complex constraints. However, the effectiveness of the approach is clear, as the model essentially *chooses* the most relevant features which lead to more informative hyperplanes. Any irrelevant features can be ignored. This implementation can be generalised for higher dimensions (especially for Model 1), where more features can be selected.

With regards to simultaneous outlier detection and feature selection, we saw in experimental results implementing both methods that the same features are selected, while the outlier detection leads to improved objective function values (compared to those presented in Table 10). That is, each of the two enhancements work to the benefit of the model without disrupting each other. We only present their applications individually in this current work to better highlight their respective benefits.

5.4. Experimental comparison with the state-of-the-art

In order to benchmark the performance of the presented models, we present an experimental comparison with the RL-FS-M model for SVM

Table 9
Application of inbuilt outlier detection to Model 1.

Segments	2			3			4			5		
Outliers	0	1	2	0	1	2	0	1	2	0	1	2
	Adult											
OF value	16.69	14.50	12.50	14.19	12.00	9.14	10.64	8.08	6.08	8.64	6.58	4.58
Structural risk	1.54	2.25	2.25	3.04	3.75	2.19	3.69	4.02	4.69	4.58	2.52	3.19
Empirical risk	15.15	12.25	10.25	11.15	8.25	6.95	10.64	4.06	1.39	4.06	4.06	1.39
Runtime	0.03	0.17	0.31	0.11	0.41	2.27	0.25	3.06	6.14	0.53	4.03	11.34
% Misclassified	12.33	9.59	8.22	9.59	6.85	6.85	6.85	4.11	1.37	4.11	4.11	1.37
	Credit											
OF value	80.78	78.61	76.41	73.38	71.11	68.91	69.37	66.97	63.89	66.17	63.68	⋆
Structural risk	0.68	0.68	0.69	0.82	0.86	0.86	0.85	0.86	2.32	1.66	2.48	⋆
Empirical risk	80.10	77.92	75.72	72.56	70.25	68.05	68.52	66.11	61.57	64.52	61.21	⋆
Runtime	0.06	2.23	25.67	0.28	20.08	911.83	4.41	1022.63	25,868.98	21.64	11,944.20	⋆
% Misclassified	51.80	50.36	49.64	38.85	36.69	35.97	36.69	35.97	35.25	35.25	34.53	⋆
	Heart											
OF value	32.06	28.79	25.14	28.28	25.10	21.88	25.57	22.70	18.73	22.01	18.55	14.53
Structural risk	1.75	1.78	1.85	3.21	3.85	1.28	3.84	3.85	3.54	4.81	2.96	4.86
Empirical risk	30.31	27.01	23.29	25.07	21.24	20.60	21.74	18.84	15.19	17.20	15.59	9.67
Runtime	0.08	0.41	2.22	0.13	3.39	35.41	0.75	37.70	344.63	4.47	381.02	836.03
% Misclassified	21.64	19.40	14.93	16.42	13.43	14.18	14.18	11.94	9.70	11.19	11.19	5.22
	WBC											
OF value	26.67	22.90	18.87	22.87	20.87	18.81	20.87	18.87	16.87	20.05	18.05	15.52
Structural risk	0.42	1.93	2.27	2.22	2.22	2.25	2.22	2.22	2.22	1.32	1.32	2.90
Empirical risk	26.25	20.97	16.62	20.64	18.64	16.56	18.64	16.64	14.64	18.74	16.74	12.62
Runtime	0.05	0.28	1.91	0.14	2.36	21.08	0.45	11.20	241.47	2.97	156.45	935.64
% Misclassified	17.07	13.01	9.76	11.38	10.57	12.20	10.57	9.76	8.94	10.57	9.76	8.13

Table 10
Application of inbuilt feature selection to Model 1.

	Segments						Segments					
	2	3	4	5	6	7	2	3	4	5	6	7
	Adult						Credit					
OF value	14.84	10.33	5.18	2.33	0.00	0.00	80.10	72.54	67.80	63.54	58.08	53.64
Runtime	0.03	0.11	0.11	0.19	0.16	0.30	0.08	0.38	3.06	12.69	43.64	158.58
% Misclassified	10.96	10.96	1.37	0	0	0	50.36	36.69	36.69	34.53	25.18	22.30
	Heart						WBC					
OF value	30.31	24.48	19.73	15.28	11.70	6.75	19.32	15.70	10.24	6.00	4.00	0.74
Runtime	0.09	0.53	3.92	12.58	35.22	32.31	0.45	6.22	19.78	23.48	50.02	107.42
% Misclassified	21.64	13.43	11.94	10.45	5.22	2.24	12.20	8.94	4.88	2.44	1.63	1.63

which was recently presented by Baldomero-Naranjo et al. (2021). RL-FS-M implements feature selection (via a budget constraint) into the SVM model with ramp loss first presented by Brooks (2011), which differentiates between data points falling into the margin and those that are explicitly misclassified. The number of misclassified data points is explicitly minimised in the objective function.

The models presented in this paper (i.e., Models 1–3) could also be implemented within the RL-FS-M framework; this section shows that even within a simpler model, the presented PWL enhancements can lead to improvements in the efficiency and accuracy of the SVM classifier.

Firstly, in Table 11 we apply RL-FS-M (using their suggested bound tightening procedure, and setting the number of features to two) to the ad-hoc data sets presented in Fig. 7. While the values for the objective function, structural risk and empirical risk cannot be explicitly compared (with those from Tables 3–4), they provide a guide for the relative performance of Models 1–3. The results suggest that while RL-FS-M can perform well on data sets with obvious separability (e.g., data set (1)) and on random data sets with no obvious patterns (e.g., data set (4)), it is not as effective on non-linearly separable data sets (e.g., data sets (2) and (3)). On these, Models 1–3 perform much better, especially with a sufficient number of PWL segments (this is partly due to the design of the data sets). The ability to capture non-linear separations effectively highlights the potential for the presented models.

Table 11
Reference values obtain by RL-FS-M (Baldomero-Naranjo et al., 2021) for the presented ad-hoc data sets.

Data set	1	2	3	4
OF value	4.77	13.91	35.00	35.69
Structural risk	3.71	1.13	1.50	1.69
Empirical risk	1.06	12.78	33.50	34.00
Runtime	0.02	0.09	0.97	2.47
% Misclassified	0	7.84	31.37	33.33

Table 12 shows the performance of RL-FS-M to the real-world data sets (where the number of features is again limited to 2). In this case, we see the performance is dependent on the data set (as with the presented models in Tables 7–8). For the Adult data set, Models 1–3 outperform RL-FS-M; we can see from Fig. 8 that this data set exhibits a general non-linear separation, and is relatively sparse in one class. The remaining data sets present a more linear separation, leading to a slight advantage for RL-FS-M. However, with increasing number of PWL segments, there is competitive performance from the presented models. With the introduction of outlier detection (see Table 9), the performance of Model 1 can significantly improve, leading to large improvements over RL-FS-M. We expect similar large improvements for Models 2–3. The presented models can be tuned (in the number of segments, clusters and outliers) and also provide information as to any change in the relationship between the variables (given by the breakpoints); such information cannot be inferred from RL-FS-M.

Table 12
Reference values obtain by RL-FS-M (Baldomero-Naranjo et al., 2021) for the presented real-world data sets.

Data set	Adult	Credit	Heart	WBC
OF value	22.47	80.57	30.06	25.74
Structural risk	1.32	1.37	1.88	3.34
Empirical risk	21.15	79.20	28.19	22.40
Runtime	0.20	25.80	0.38	0.09
% Misclassified	12.33	23.02	8.96	6.50

Table 13
Reference values obtain by RL-FS-M (Baldomero-Naranjo et al., 2021) for the presented real-world data sets (with all features available).

Data set	Adult	Credit	Heart	WBC
OF value	18.97	80.57	30.06	20.55
Runtime	0.28	985.60	39.06	8.61
% Misclassified	12.32	23.02	8.96	7.32

Finally, Table 13 presents results for RL-FS-M on the real-world data sets where there is a free choice of features. However, for a fair comparison, we fix the first feature and allow the model to choose a second (i.e., we set a budget of 2). In general, when compared to Model 1 (presented in Table 10), the performance is dependent on the data set. However, given enough segments (i.e, increased model complexity), Model 1 is able to outperform RL-FS-M in each case. Again, the inclusion of outlier detection in Model 1 leads to significant increases in comparative performance. This further highlights that a hybrid approach would lead to an improved model, and that applying the improvements of outlier detection and feature selection into the presented models, the performance significantly increases.

6. Conclusion

Support vector machines (SVMs) are a powerful tool to classify labelled data. We have presented a framework consisting of three mixed-integer linear programming (MILP) models for finding appropriate SVMs for classified data in two dimensions, including a generalised model and a model which implements feature selection. Due to the use of binary variables and logical implications modelled by big-M constraints, these models fit well into a recently presented MILP framework for machine learning problems. As such, we can use tailored solution methods to increase their robustness and improve their efficiency.

We have seen from an experimental analysis where each of the presented models performs well, and the quality of information provided by each. The application of inbuilt outlier detection and feature selection leads to more robust models which provide higher levels of information, as well as showcasing when these models outperform the state-of-the-art.

For future work, we wish to implement the presented models and framework within modern SVM approaches, such as those presented by Baldomero-Naranjo et al. (2021). We further aim to generalise the models further into higher dimensions. This will involve the presentation of a MILP model for fitting triangulations to data, which will be designed with the existing MILP framework in mind. We also wish to present MILP models for SVM where the data can receive more labels; a straightforward extension is to classify (say $M > 0$) different labels using M SVMs, where each classifies based on a given label (i.e., does it have this label or does it not?). Additionally, we aim to extend the presented models for applications within semi-supervised learning, where kernel-free methods typically perform well.

CRediT authorship contribution statement

John Alasdair Warwicker: Conceptualization, Methodology, Software, Validation, Writing – original draft. **Steffen Rebennack:** Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Steffen Rebennack reports financial support was provided by the German Research Foundation.

Data availability

Data will be made available on request.

Acknowledgments

This work was funded by the Deutsche Forschungsgemeinschaft, Germany (DFG, German Research Foundation - 445857709).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eswa.2023.122998.

References

Aladeemy, M., Tutun, S., & Khasawneh, M. T. (2017). A new hybrid approach for feature selection and support vector machine model selection based on self-adaptive cohort intelligence. *Expert Systems with Applications*, *88*, 118–131.
Alcaraz, J., Labbé, M., & Landete, M. (2022). Support Vector Machine with feature selection: A multiobjective approach. *Expert Systems with Applications*, *204*, Article 117485.
Baldomero-Naranjo, M., Martínez-Merino, L. I., & Rodríguez-Chía, A. M. (2020). Tightening big Ms in integer programming formulations for support vector machines with ramp loss. *European Journal of Operational Research*, *286*(1), 84–100.
Baldomero-Naranjo, M., Martinez-Merino, L. I., & Rodriguez-Chia, A. M. (2021). A robust SVM-based approach with feature selection and outliers detection for classification problems. *Expert Systems with Applications*, *178*, Article 115017.
Belotti, P., Bonami, P., Fischetti, M., Lodi, A., Monaci, M., Nogales-Gómez, A., & Salvagnin, D. (2016). On handling indicator constraints in mixed integer programming. *Computational Optimization and Applications*, *65*(3), 545–566.
Benítez-Peña, S., Blanquero, R., Carrizosa, E., & Ramírez-Cobo, P. (2019). Cost-sensitive feature selection for support vector machines. *Computers & Operations Research*, *106*, 169–178.
Bertsimas, D., & Dunn, J. (2019). *Machine learning under a modern optimization lens*. Dynamic Ideas LLC.
Blanco, V., Japón, A., & Puerto, J. (2022). A mathematical programming approach to SVM-based classification with label noise. *Computers & Industrial Engineering*, *172*, Article 108611.
Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory* (pp. 144–152).
Bradley, P. S., & Mangasarian, O. L. (1998). Feature selection via concave minimization and support vector machines. In *Proceedings of the fifteenth international conference on machine learning* ICML '98, (pp. 82–90). Morgan Kaufmann Publishers Inc..
Brooks, J. P. (2011). Support vector machines with the ramp loss and the hard margin loss. *Operations Research*, *59*(2), 467–479.
Burgard, J. P., Pinheiro, M. E., & Schmidt, M. (2023). Mixed-integer quadratic optimization and iterative clustering techniques for semi-supervised support vector machines.
Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*(2), 121–167.
Christmann, A., & Steinwart, I. (2008). *Support vector machines*. Springer.
Codato, G., & Fischetti, M. (2006). Combinatorial Benders' cuts for mixed-integer linear programming. *Operations Research*, *54*, 756–766.
Collobert, R., Sinz, F., Weston, J., & Bottou, L. (2006). Trading convexity for scalability. In *Proceedings of the twenty-third international conference on machine learning* ICML '06, (pp. 201–208). Morgan Kaufmann Publishers Inc..
Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.
Dua, D., & Graff, C. (2017). UCI machine learning repository.
Fieberg, C., Metko, D., Poddig, T., & Loy, T. (2023). Machine learning techniques for cross-sectional equity returns' prediction. *OR Spectrum*, *45*(1), 289–323.
Freitas, A., Costa-Pereira, A., & Brazdil, P. (2007). Cost-sensitive decision trees applied to medical data. In I. Y. Song, J. Eder, & T. M. Nguyen (Eds.), *Data warehousing and knowledge discovery* (pp. 303–312). Springer.
Guyon, I., Gunn, S., Nikravesh, M., & Zadeh, L. A. (2008). *Feature extraction: foundations and applications, Vol. 207*. Springer.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, *46*(1), 389–422.

Huang, W., Nakamori, Y., & Wang, S.-Y. (2005). Forecasting stock market movement direction with support vector machine. *Computers & Operations Research*, *32*(10), 2513–2522.

Huang, C.-L., & Wang, C.-J. (2006). A GA-based feature selection and parameters optimizationfor support vector machines. *Expert Systems with Applications*, *31*(2), 231–240.

Kong, L., & Maravelias, C. T. (2020). On the derivation of continuous piecewise linear approximating functions. *INFORMS Journal on Computing*, *32*(3), 531–546.

Labbé, M., Martínez-Merino, L. I., & Rodríguez-Chía, A. M. (2019). Mixed integer linear programming for feature selection in support vector machine. *Discrete Applied Mathematics*, *261*, 276–304.

Lee, I. G., Zhang, Q., Yoon, S. W., & Won, D. (2020). A mixed integer linear programming support vector machine for cost-effective feature selection. *Knowledge-Based Systems*, *203*, Article 106145.

Maldonado, S., Pérez, J., Weber, R., & Labbé, M. (2014). Feature selection for support vector machines via mixed integer linear programming. *Information Sciences*, *279*, 163–175.

Mangasarian, O. L. (1965). Linear and nonlinear separation of patterns by linear programming. *Operations Research*, *13*(3), 444–452.

Mangasarian, O. (1968). Multisurface method of pattern separation. *IEEE Transactions on Information Theory*, *14*(6), 801–807.

Marsland, S. (2011). *Machine learning: an algorithmic perspective*. Chapman and Hall/CRC.

Noble, W. S. (2006). What is a support vector machine? *Nature biotechnology*, *24*(12), 1565–1567.

Park, Y. W., Jiang, Y., Klabjan, D., & Williams, L. (2017). Algorithms for generalized clusterwise linear regression. *INFORMS Journal on Computing*, *29*(2), 301–317.

Rebennack, S., & Krasko, V. (2020). Piecewise linear function fitting via mixed-integer linear programming. *INFORMS Journal on Computing*, *32*(2), 507–530.

Rivas-Perea, P., Cota-Ruiz, J., Chaparro, D. G., Venzor, J. A. P., Carreón, A. Q., & Rosiles, J. G. (2012). Support vector machines for regression: a succinct review of large-scale and linear programming formulations. *International Journal of Intelligence Science*, *3*(1), 5–14.

Sudermann-Merx, N., & Rebennack, S. (2021). Leveraged least trimmed absolute deviations. *OR Spectrum*, 1–26.

Sun, Y., Ernst, A., Li, X., & Weiner, J. (2021). Generalization of machine learning for problem reduction: a case study on travelling salesman problems. *OR Spectrum*, *43*, 607–633.

Sun, Y., Li, X., & Ernst, A. (2019). Using statistical measures and machine learning for graph reduction to solve maximum weight clique problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(5), 1746–1760.

Vapnik, V. N. (1998). *Statistical learning theory*. Wiley-Interscience.

Warwicker, J. A., & Rebennack, S. (2022). A comparison of two mixed-integer linear programs for piecewise linear function fitting. *INFORMS Journal on Computing*, *34*(2), 1042–1047.

Warwicker, J. A., & Rebennack, S. (2023a). Generating optimal robust continuous piecewise linear regression with outliers through combinatorial Benders decomposition. *IISE Transactions*, *55*(8), 755–767.

Warwicker, J. A., & Rebennack, S. (2023b). A unified framework for clustering and regression problems via mixed-integer linear programming. *Discrete Applied Mathematics*, *336*, 15–36.

Zhou, W., Zhang, L., & Jiao, L. (2002). Linear programming support vector machines. *Pattern Recognition*, *35*(12), 2927–2936, Pattern Recognition in Information Systems.