Machine Learning
https://doi.org/10.1007/s10994-024-06561-8

Permutation-invariant linear classifiers

Ludwig Lausser[1,2] · **Robin Szekely**[1] · **Hans A. Kestler**[1]

Received: 30 November 2022 / Revised: 31 January 2024 / Accepted: 26 April 2024

Abstract
Invariant concept classes form the backbone of classification algorithms immune to specific data transformations, ensuring consistent predictions regardless of these alterations. However, this robustness can come at the cost of limited access to the original sample information, potentially impacting generalization performance. This study introduces an addition to these classes—the permutation-invariant linear classifiers. Distinguished by their structural characteristics, permutation-invariant linear classifiers are unaffected by permutations on feature vectors, a property not guaranteed by other non-constant linear classifiers. The study characterizes this new concept class, highlighting its constant capacity, independent of input dimensionality. In practical assessments using linear support vector machines, the permutation-invariant classifiers exhibit superior performance in permutation experiments on artificial datasets and real mutation profiles. Interestingly, they outperform general linear classifiers not only in permutation experiments but also in permutation-free settings, surpassing unconstrained counterparts. Additionally, findings from real mutation profiles support the significance of tumor mutational burden as a biomarker.

Keywords Permutation invariance · Linear classifiers · Concept classes · Tumor mutational burden

Editors: Michelangelo Ceci, João Gama, Jose Lozano, André de Carvalho, Paula Brito.

Ludwig Lausser and Robin Szekely contributed equally to this work.

✉ Hans A. Kestler
hans.kestler@uni-ulm.de

Ludwig Lausser
ludwig.lausser@thi.de

Robin Szekely
robinsz@web.de

[1] Institute of Medical Systems Biology, Ulm University, Albert-Einstein-Allee 11, 89081 Ulm, Germany

[2] Faculty of Computer Science, Technische Hochschule Ingolstadt, Esplanade 10, 85049 Ingolstadt, Germany

1 Introduction

Supervised machine learning techniques are a key ingredient in analyses and diagnoses based on high-dimensional data (Kraus et al., 2018). Their underlying algorithms allow for screening through molecular marker profiles of tens of thousands of measurements that by far exceed the capability of a manual inspection. The revealed multivariate patterns have therefore a high potential of extending the existing toolbox of known diagnostic models.

Nevertheless, these new classification tasks come also with new challenges (L'Heureux et al., 2017). A prominent example might be the joint analysis of heterogeneous data sources that were recorded under varying conditions. Here, subgroups or single samples of a data collection might be affected by individual intended or unintended modifications. Depending on the number of affected samples, they will be called group-wise or sample-wise data transformations in the following. Both data transformations can affect a classifier in two different ways. The corresponding altered samples can distract a training algorithm from its original classification task or directly corrupt the predictions of the trained classifier. In both cases, classifiers can suffer from a degenerated generalization performance.

Various countermeasures against these effects of heterogeneous data sources were proposed (Valente & Rocha, 2015). They try to reduce or even neglect the influence of the corresponding data transformations. If a countermeasure is guaranteed to neglect the effect of a specific data transformation it is said to induce an invariance (Haasdonk & Burkhardt, 2007). These methods will be the major focus of this article.

Group-wise transformations (e.g. common biases) affect subsets of samples identically and therefore might be detected and identified by comparisons of multiple samples. Their influence might be reduced by group-wise normalization techniques that estimate the common parameters of a transformation from a set of affected training or test samples (Singh & Singh, 2020). Other classifiers directly induce invariances against global group-wise transformations. An example might be distance-based classifiers, such as the k-NN, that are invariant against a common global rescaling of all training and test samples (Fix & Hodges, 1951).

Sample-wise transformations affect individual samples by different data transformations from a family of data transformations. They can be considered as a generalization of the group-wise ones. In this setting, no common parameters exist and can therefore not be estimated. As a consequence, most of the techniques described above do not protect from this type of influence. We will concentrate on this type of data transformation in the following.

Sample-wise data transformations of this type can in general be detected during the training phase of a classifier and a corresponding invariance might be induced by the underlying training algorithm (Haasdonk & Burkhardt, 2007). In this case, the predictions of a so-called invariant classifier will not be influenced by a specific data transformation or a family of data transformations. However, inducing a sample-wise invariance via adaptation can increase the complexity of the original learning task, as the corresponding models often require exact parameter settings. It is therefore necessary to increase the required amount of training samples, which might not be available. This especially holds for the high-dimensional setting ($n \gg m$), in which the number of features n by far exceeds the number of available samples m.

Alternatively, an invariance can be induced during the design phase of a classifier (Lausser et al., 2018a). Here, an invariant model or concept class might be chosen in order to counteract a known or assumed type of sample-wise data transformation (Lausser et al., 2020). In contrast to the training approach, this design choice is not data-driven and does not require additional training samples. However, it requires an a priori assumption of the type of data transformation and is likely to be designed for a more general family. As an invariant model generally cannot distinguish between the original sample and the overlaying noise effects a certain amount of information is excluded from the classification process in any case. The choice of an invariant concept class should therefore be as specific as possible.

Invariant concept classes can be generated in two different ways. The first option is to couple a non-invariant concept class to a preprocessing that generates an invariant data representation. In this way, the new concept class will again neglect the influence of the corresponding data transformation. However, the new concept class is only loosely related to the original one and loses its interpretation. For example, a linear classifier operating on rank-transformed data is not linear in the original feature space (Lausser et al., 2018a). It is therefore unclear how to switch from these invariant concept classes to another. The second option is to determine a subclass of all classifiers of a concept class that fulfills an invariance property. The invariant subclass keeps the interpretation of the original one and can easily be extended or restricted in order to adapt the invariance property. They can generate a fine granular grid of related invariant concept classes.

We previously outlined the landscape of invariant subclasses of the concept class of linear classifiers (Schmid et al., 2014; Lausser & Kestler, 2014; Lausser et al., 2020). In this work, we extend this roadmap by the concept class of permutation-invariant linear classifiers. This type of linear classifier neglects the effects of disordered feature representations. We characterize this concept class by its capacity, which is constant regardless of the dimensionality of its underlying feature representation. We utilize the concept class for the design of a permutation-invariant version of the (linear) support vector machine (SVM) (Vapnik, 1998), which interestingly seems to be beneficial in specific permutation-free scenarios.

Our theoretical work indicates that for a linear classifier, the gain of a permutation invariance coincides with the restriction to the analysis of unweighted sums or rates. Those features were found to be valuable features in the context of tumor classification (Stenzinger et al., 2019). Here, the tumor mutational burden, the number of mutations within a sample, was found to allow various classifications in this field of application (Chalmers et al., 2017). Our own experiments with the permutation-invariant SVM on the COSMIC Cell Line Gene Mutation Profiles dataset (Forbes et al., 2010) fall in line with these observations.

The remaining article is organized as follows. Section 2 provides the utilized notation and the underlying notion of concept classes and their invariance properties. The concept class of permutation-invariant linear classifiers is introduced and analyzed in Sect. 3. It especially introduces a permutation-invariant version of the linear support vector machine, which is later evaluated and compared against its unconstrained counterpart in experiments on artificial and real-life datasets (Sect. 4). The corresponding results are reported in Sect. 5. The theoretical findings and empirical results are finally summarised and discussed in Sect. 6.

2 Methods

Our work is mainly based on a standard notation for classification experiments. We will focus on crisp classification functions $c \in \mathcal{C}$ that predict a categorical class label $y \in \mathcal{Y}$ of an object according to a set of measurements $\mathbf{x} \in \mathcal{X}$

$$c : \mathcal{X} \rightarrow \mathcal{Y}. \tag{1}$$

The feature space is assumed to be embedded in an n-dimensional Euclidean space $\mathcal{X} \subseteq \mathbb{R}^n$. The label space is assumed to comprise a finite set of discrete class labels ($|\mathcal{Y}| < \infty$). We concentrate on binary classification problems (e.g. $\mathcal{Y} = \{0, 1\}$) in the following.

A sample $\mathbf{x} \in \mathcal{X}$ is given by a n-dimensional vector $\mathbf{x} = (x^{(1)}, \dots, x^{(n)})^T$. We make use of index vectors $\mathbf{i} = (i_1, \dots, i_k)^T$, $k \leq n$ to indicate specific subsets or permutations of a sample $\mathbf{x}$. In this case, $\mathbf{x}^{(\mathbf{i})} = (x^{(i_1)}, \dots, x^{(i_k)})^T$. As we focus on permutations, we allow $\mathbf{i} \in \mathcal{I}$ with $\mathcal{I} = \bigcup_{k=1}^{n} \mathcal{I}_k$, $\mathcal{I}_k \subset \mathbb{N}^k$ and

$$\mathcal{I}_k = \{(i_1, \dots, i_k)^T \mid \forall j : 1 \leq i_j \leq n, \forall j' \neq j : i_{j'} \neq i_j\}. \tag{2}$$

The classifier itself is chosen from a predefined concept or function class $c \in \mathcal{C}$, which characterizes the structural properties and the layout of a classifier (Anthony & Biggs, 1997). Typically, the optimal structure $c \in \mathcal{C}$ is unknown a priori and is learned in a data-driven training phase for a specific classification task

$$l : \mathcal{C} \times \mathcal{T} \mapsto c_{\mathcal{T}} \in \mathcal{C}. \tag{3}$$

Here, l is the chosen learning algorithm and $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$ denotes a set of labeled training examples. If the training set is known from the context, the subscript $_{\mathcal{T}}$ will be omitted. The generalization performance of a trained classifier is estimated on an independent validation set $\mathcal{V} = \{(\mathbf{x}'_i, y'_i)\}_{i=1}^{m'}$. We will use empirical accuracy as a quality measure

$$Acc(c, \mathcal{V}) = \frac{1}{m'} \sum_{(\mathbf{x},y) \in \mathcal{V}} \mathbb{I}_{[c(\mathbf{x})=y]}, \tag{4}$$

where $\mathbb{I}_{[\cdot]}$ denotes the indicator function.

2.1 Invariant concept classes

Invariance properties guarantee a classifier to be unaffected by a certain family of data transformations (Haasdonk & Burkhardt, 2007; Lausser et al., 2018a). That is, the classifier will predict the same class label in the presence or absence of a transformation. It might be used to neglect certain types of noise or confounders. Although invariances can in general be the result of the training process l, they are more likely to be induced by some kind of design principle in order to counteract assumed interference.

We previously found that invariances can be traced back to structural properties of classifiers and can therefore be formulated as invariant concept (sub-) classes (Lausser et al., 2020). Those can be chosen a priori according to external domain knowledge and linked to standard training algorithms. Invariances can therefore be incorporated during the design phase of a classification algorithm.

We will use the following definition of invariant classifiers and concept classes (Schmid et al., 2014):

Definition 1 A classifier $c : \mathcal{X} \rightarrow \mathcal{Y}$ is *invariant* against a parameterised class of data transformations $f_\theta : \mathcal{X} \rightarrow \mathcal{X}$ if

$$\forall \theta \in \Theta, \forall \mathbf{x} \in \mathcal{X} : \quad c(f_\theta(\mathbf{x})) = c(\mathbf{x}). \tag{5}$$

A concept class $\mathcal{C}$ is *invariant* against f_θ if each $c \in \mathcal{C}$ is invariant against f_θ.

Definition 1 allows for following three major implications:

(a) *Sample-wise invariance* The invariant classifier does not assume a common data transformation for all test samples in $\mathcal{V}$. In principle, each test sample can be affected by an individual one (individual choice of $\theta_i \in \Theta$)

$$\forall(\mathbf{x}_i, y_i) \in \mathcal{V} : c(f_{\theta_i}(\mathbf{x}_i)) = c(\mathbf{x}_i). \tag{6}$$

(b) *Invariance against cascaded data transformations* Invariant classifiers can absorb the influence of cascaded applications of data transformations

$$\forall \theta_1, \dots, \theta_k \in \Theta : c(f_{\theta_1}(f_{\theta_2}(\dots (f_{\theta_k}(\mathbf{x}))))) \tag{7}$$

$$= c(f_{\theta_2}(\dots (f_{\theta_k}(\mathbf{x})))) \tag{8}$$

$$= c(f_{\theta_k}(\mathbf{x})) \tag{9}$$

$$= c(\mathbf{x}). \tag{10}$$

(c) *Inheritance of invariance* Ensembles classifiers $h(\mathbf{x}) \in \mathcal{H}(\mathcal{C})$ and their underlying fusion architectures u based on classifiers $c(\mathbf{x}) \in \mathcal{E} \subseteq \mathcal{C}$ inherit the invariance properties of $\mathcal{C}$

$$h(f_\theta(\mathbf{x})) = u\Big(c_1(f_\theta(\mathbf{x})), \dots, c_{|\mathcal{E}|}(f_\theta(\mathbf{x}))\Big) = u(c_1(\mathbf{x}), \dots, c_{|\mathcal{E}|}(\mathbf{x})) = h(\mathbf{x}), \tag{11}$$

where $h : \mathcal{X} \longrightarrow \mathcal{Y}$ and $u : \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_{|\mathcal{E}|} \longrightarrow \mathcal{Y}$.

Note that constant classifiers (e.g., $\forall \mathbf{x} : c(\mathbf{x}) = 1$) are invariant against all data transformations. However, they are also uninformative and are therefore neglected in the following.

2.2 Concept classes and their capacity

The choice of a concept class $\mathcal{C}$ does not only determine the structural properties of the finally trained classifier $c_\mathcal{T}$ but also the training process l itself. It affects the risk of overfitting and a declined generalization performance. That is a complex concept class that can be adapted to any dataset $\mathcal{T}$ of size m is likely to result in a classifier $c_\mathcal{T}$ that misclassifies unseen samples $\mathcal{V}$. More rigid concept classes (of a lower complexity) have a lower chance of adapting to arbitrary structures and tend to have more accurate predictions.

Different notions of complexity exist. In the following, we will utilize the Vapnik-Chervonenkis dimension (VCdim), which is a combinatorial measure for the capacity of a concept class (Vapnik, 1998). It is based on the identification of a maximal set of m data points $\mathbf{X}_m = \{\mathbf{x}_i\}_{i=1}^m$, $\mathbf{x}_i \in \mathcal{X}$ that can receive all 2^m binary labelings $\mathbf{Y}_m = \{y_i\}_{i=1}^m$, $y_i \in \mathcal{Y}$ by a classifier $c \in \mathcal{C}$

$$\mathrm{VCdim}(\mathcal{C}) = \arg\max_{m} \quad \exists \mathbf{X}_m \forall \mathbf{Y}_m \forall i \exists c \in \mathcal{C} : c(\mathbf{x}_i) = y_i. \tag{12}$$

The influence of the VCdim on the generalization ability of a classification model is for example described in the probably approximately correct (PAC) learning framework, where it is applied to obtain upper bounds of the generalization performance of a classifier (Valiant, 1984). If two classifiers with equal empirical performance are available, the one with the lower VCdim should be preferred (Burges, 1998). This especially holds for nested hierarchies of concept classes ($\mathcal{C} \supseteq \mathcal{C}_1 \supseteq \cdots \supseteq \mathcal{C}_k$). The smallest reasonable subclass of $\mathcal{C}$ should be chosen.

2.3 The concept class of linear classifiers ($\mathcal{C}_{lin}$)

One of the oldest formalized classification principles is the idea of linear classification models (Fisher, 1936). The underlying concept class will be denoted by $\mathcal{C}_{lin}$ in the following. Classifiers of this concept class separate the feature space into two classes via linear hyperplanes (Fig. 1). The theoretical capabilities of $\mathcal{C}_{lin}$ were analyzed in different ways. It was especially shown that some rather simple patterns are not linearly separable (XOR problem) (Minsky & Papert, 1988). However, it was proven by Cover that the probability for linearly separable patterns increases with the dimensionality n of the data if the samples lie in general position (Cover, 1965). Nowadays it is known that the VCdim of $\mathcal{C}_{lin}$ depends on the dimensionality n of the underlying feature space $\mathcal{X}$ (Burges, 1998). It corresponds to $\mathrm{VCdim}(\mathcal{C}_{lin}) = n + 1$.

Many training algorithms in the literature explicitly or implicitly utilize $\mathcal{C}_{lin}$ as underlying concept class (Lausser & Kestler, 2010). Some well-known examples are the perceptron (Rosenblatt, 1958), the linear discriminant analysis (Fisher, 1936), or the support vector machine (Vapnik, 1998). Linear classifiers are also basic building blocks for various ensemble techniques (Freund & Schapire, 1997; Kestler et al., 2011) and especially multi-class classifier systems (Abe, 2010; Lausser et al., 2018b, 2019).

We utilize the following definition for the concept class of linear classifiers $\mathcal{C}_{lin}$.

Definition 2 ($\mathcal{C}_{lin}$) The concept class of linear classifiers $\mathcal{C}_{lin}$ is defined as

$$\mathcal{C}_{lin} = \left\{ \mathbb{I}_{[\langle \mathbf{w}, \mathbf{x} \rangle \geq t]} \mid \mathbf{w} \in \mathbb{R}^n, t \in \mathbb{R} \right\}. \tag{13}$$

The two parameters of a linear classifier are $\mathbf{w} \in \mathbb{R}^n$, determining the orientation of its hyperplane, and $t \in \mathbb{R}$, providing the distance of the hyperplane to the origin. An overview of the influence of the two parameters is given in Fig. 1.

For a one-dimensional input space ($n = 1$) it can be simplified as follows

$$\mathcal{C}_{lin} = \left\{ \mathbb{I}_{[w \cdot x \geq t]} \mid w \in \mathbb{R}, t \in \mathbb{R} \right\} \tag{14}$$

$$= \left\{ \mathbb{I}_{[\mathrm{sgn}(w)|w| \cdot x \geq t]} \mid w \in \mathbb{R}, t \in \mathbb{R} \right\} \tag{15}$$

$$= \left\{ \mathbb{I}_{[a \cdot x \geq t]} \mid a \in \{-1, 0, +1\}, t \in \mathbb{R} \right\}, \tag{16}$$

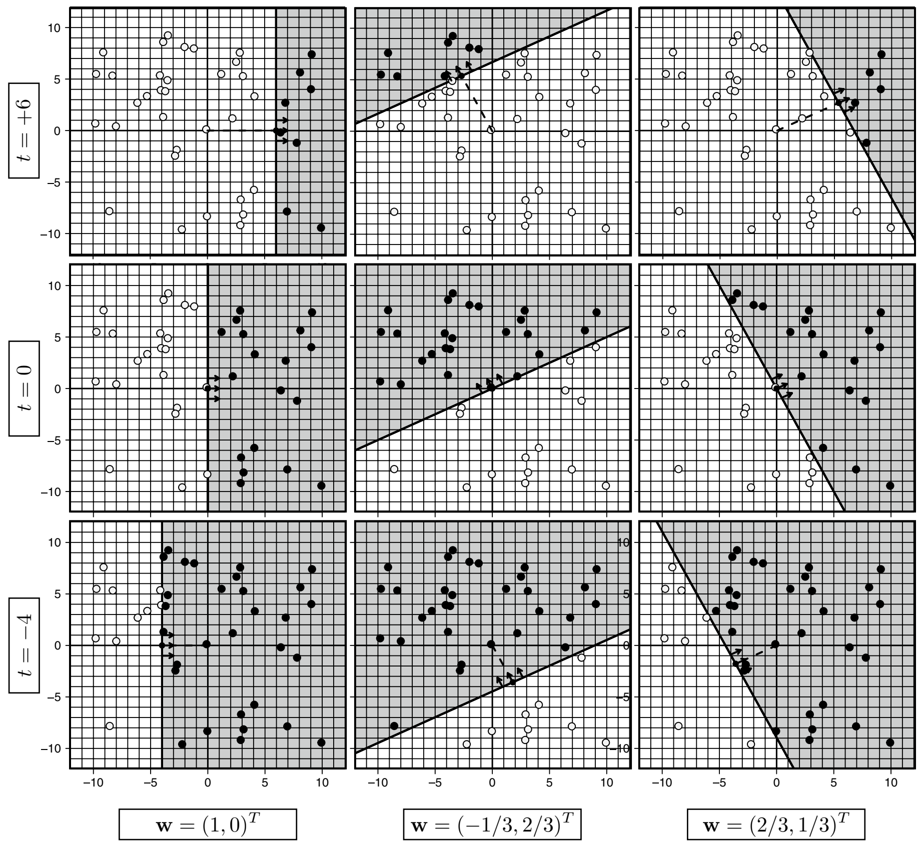

Fig. 1 Concept class $\mathcal{C}_{lin}$: General linear classifiers. The figure shows examples of general linear classifiers for three different orientations ($\mathbf{w} \in \mathbb{R}^2$) and three different distances to the origin ($t \in \mathbb{R}$). The classifiers shown in the first column are additionally members of the concept class of single threshold classifiers ($\mathcal{C}_{stc}$)

where function sgn(w) denotes the signum function fulfilling $w = \text{sgn}(w)|w|$. It is therefore solely determined by its distance to the origin and its polarisation.

2.4 The concept class of fixed linear projections ($\mathcal{C}_{\mathbf{w}^*} \subseteq \mathcal{C}_{lin}$)

The concept class of linear classifiers comprises various subclasses with individual structural properties inducing distinct characteristics in terms of complexity, invariances, and interpretation. We focus on the complexity-reducing concept class of fixed linear projections $\mathcal{C}_{\mathbf{w}^*} \subseteq \mathcal{C}_{lin}$ in the following (Fig. 2A).

Definition 3 **($\mathcal{C}_{\mathbf{w}^*}$)** The concept class of fixed linear projections $\mathcal{C}_{\mathbf{w}^*} \subseteq \mathcal{C}_{lin}$ is defined as

$$\mathcal{C}_{\mathbf{w}^*} = \left\{ \mathbb{I}_{[\langle \mathbf{w},\mathbf{x}\rangle \geq t]} \mid \mathbf{w} = a \cdot \mathbf{w}^*, a \in \mathbb{R}, t \in \mathbb{R} \right\}, \tag{17}$$

where $\mathbf{w}^*$ is an arbitrary but fixed vector $\mathbf{w}^* \in \mathbb{R}^n$.

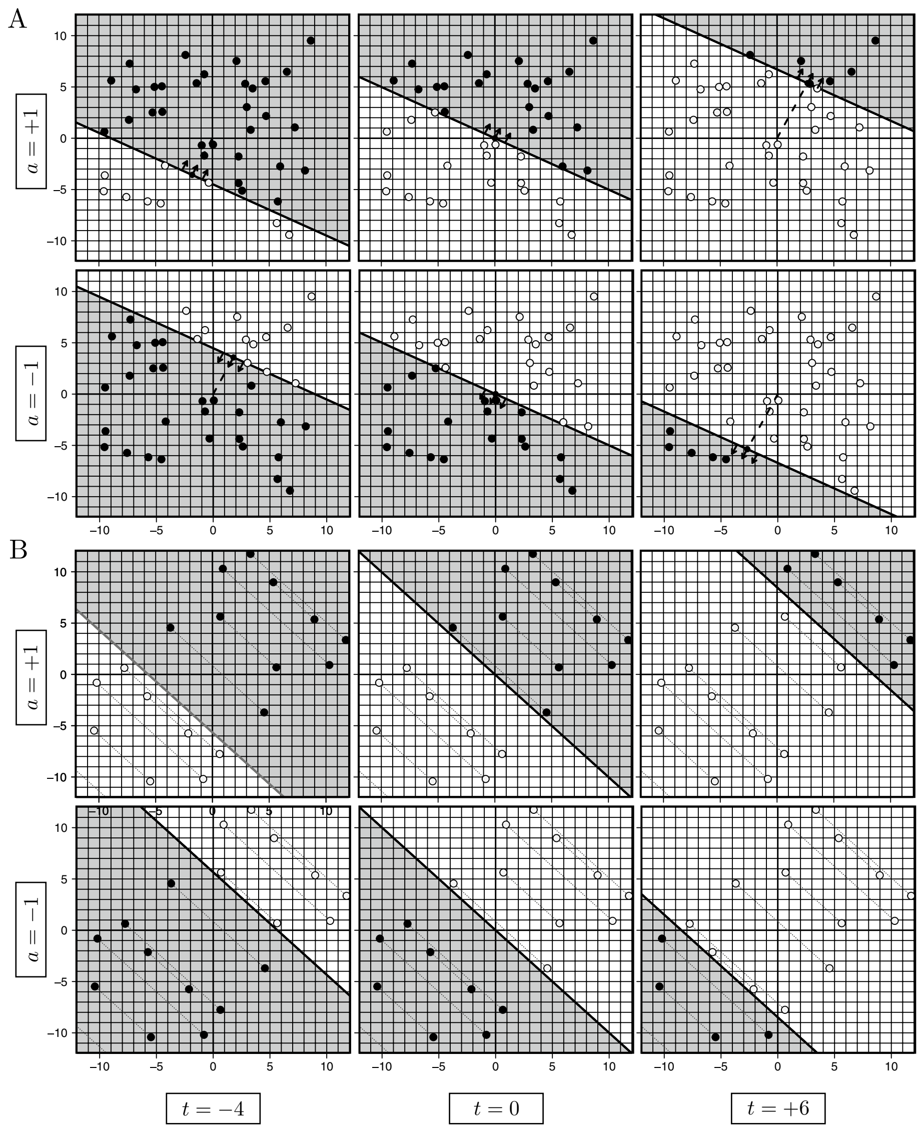

Fig. 2 Concept class $\mathcal{C}_{\mathbf{w}^*}$: Fixed linear projections. The figure shows linear classifiers for two fixed linear projection $\mathbf{w}^*$. They vary in their distances to the origing ($t \in \mathbb{R}$) and polarisations $a \in \{+1, -1\}$. **A** Classifiers for the fixed projection $\mathbf{w}^* = (1, 2)^T$. **B** Illustrates $\mathbf{w}^* = (1, 1)^T$, the permutation-invariant concept class $\mathcal{C}_{sum}$

The concept class $\mathcal{C}_{\mathbf{w}^*}$ comprises linear classifiers that operate on a priori selected weight vector that is selected during the design phase of a classification algorithm. This vector was neither trained nor adapted on the training samples $\mathcal{T}$ or the validation samples $\mathcal{V}$. This especially excludes data-driven projection methods such as principle components (Jolliffe, 1986) or independent components (Stone, 2004).

The restriction to an a priori fixed (untrainable) weight vector $\mathbf{w}^* \in \mathbb{R}^n$ clearly reduces the flexibility of $\mathcal{C}_{lin}$ and also affects its complexity as

$$\mathcal{C}_{\mathbf{w}^*} = \left\{ \mathbb{I}_{[\langle \mathbf{w}, \mathbf{x} \rangle \geq t]} \mid \mathbf{w} = a \cdot \mathbf{w}^*, a \in \mathbb{R}, t \in \mathbb{R} \right\} \tag{18}$$

$$= \left\{ \mathbb{I}_{[a \cdot \langle \mathbf{w}^*, \mathbf{x} \rangle \geq t]} \mid a \in \mathbb{R}, t \in \mathbb{R} \right\} \tag{19}$$

$$= \left\{ \mathbb{I}_{[a \cdot x' \geq t]} \mid a \in \mathbb{R}, t \in \mathbb{R} \right\}, \tag{20}$$

where $x' = \langle \mathbf{w}^*, \mathbf{x} \rangle \in \mathbb{R}$ is an univariate projection. Linear classifiers $\mathcal{C}_{\mathbf{w}^*} \subseteq \mathcal{C}_{lin}$ originally designed for a multivariate input space $\mathcal{X} \subseteq \mathbb{R}^n$ are therefore equivalent to linear classifiers for a (projected) univariate input space $\mathcal{X}' \subseteq \mathbb{R}$. As a consequence $\mathrm{VCdim}(\mathcal{C}_{\mathbf{w}^*}) = 2$ and $\mathcal{C}_{\mathbf{w}^*} = \mathcal{C}_{lin}$ if $n = 1$.

Note that $\mathcal{C}_{\mathbf{w}^*}$ absorbs size and polarisation of $\mathbf{w}^*$. It is therefore shared by different projections $\mathbf{w} \in \mathbb{R}^n$ with $\mathbf{w}^* = a \cdot \mathbf{w}$, $a \in \mathbb{R}$.

2.4.1 Single threshold classifiers ($\mathcal{C}_{stc}$)

The concept class $\mathcal{C}_{\mathbf{w}^*}$ is a basic building block for constructing other subclasses of $\mathcal{C}_{lin}$ by allowing multiple weight vectors. A prominent example is the concept class of single threshold classifiers $\mathcal{C}_{stc} \subseteq \mathcal{C}_{lin}$, which utilizes the union of basis vector $\mathbf{e}_i$, $1 \leq i \leq n$, with $e_i^{(j)} = 1$ if $i = j$ and $e_i^{(j)} = 0$ else. It can summarized as

$$\mathcal{C}_{stc} = \{ \mathbb{I}_{[a \cdot x^{(i)} \geq t]} \mid a = \pm 1, t \in \mathbb{R}, i \in \{1, \ldots, n\} \}. \tag{21}$$

It can again be observed that $\mathcal{C}_{stc} = \mathcal{C}_{lin}$ if $n = 1$. $\mathcal{C}_{stc}$ is especially suitable for analyzing single features independently. They are often used as base learners in classifier ensembles (Freund & Schapire, 1997; Kestler et al., 2011).

As $\mathcal{C}_{\mathbf{w}^*}$ can provide at most $2m$ labelings for m fixed data points, $\mathcal{C}_{stc}$ can provide at most $2mn$ labelings in an n dimensional setting. As a consequence and because of $\mathcal{C}_{stc} \subset \mathcal{C}_{lin}$, $\mathrm{VCdim}(\mathcal{C}_{stc}) \leq \min(m^*, n+1)$, where

$$m^* = \operatorname*{arg\,max}_{m} \{2mn \geq 2^m\}. \tag{22}$$

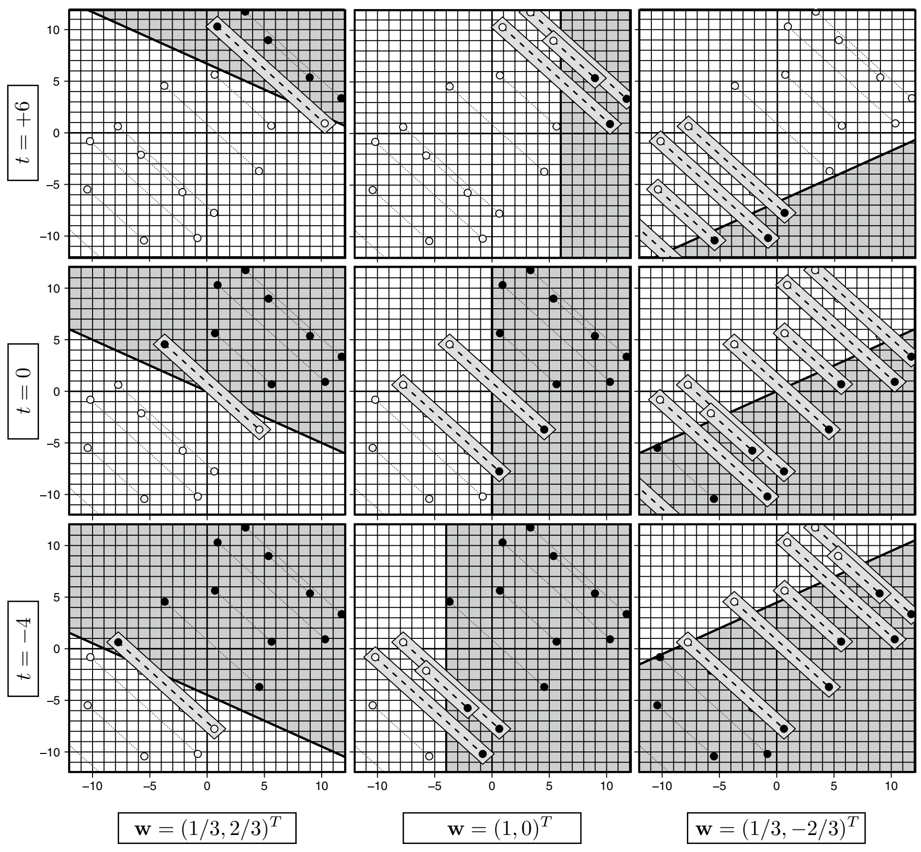

Fig. 3 Influence of permuted feature representations. The influence of permuted feature representations on the classification via general linear classifiers $c \in \mathcal{C}_{lin} \backslash \mathcal{C}_{sum}$ is shown. Each sample $\mathbf{x}^{((1,2)^T)}$ is connected to its permuted counterpart $\mathbf{x}^{((2,1)^T)}$. Those pairs that receive two different class labels are marked by a gray box

3 Permutation-invariant linear classifiers ($\mathcal{C}_{sum} \subseteq \mathcal{C}_{lin}$)

Various subclasses of linear classifiers fulfill invariance properties (Lausser et al., 2020; Schmid et al., 2014; Lausser & Kestler, 2014). In the following, we add the invariant subclass of linear classifiers $\mathcal{C}_{sum} \subseteq \mathcal{C}_{lin}$ to this list. It is characterized by equivalent weights for all feature dimensions. An illustration of $\mathcal{C}_{sum}$ can be found in Fig. 2B. The influence of permuted feature representations is given in Fig. 3.

Definition 4 ($\mathcal{C}_{sum}$) The concept class of permutation-invariant linear classifiers $\mathcal{C}_{sum} \subseteq \mathcal{C}_{lin}$ is defined as

$$\mathcal{C}_{sum} = \left\{ \mathbb{I}_{[\langle \mathbf{w}, \mathbf{x} \rangle \geq t]} \mid \mathbf{w} = a \cdot \mathbf{1}_n, a \in \mathbb{R}, t \in \mathbb{R} \right\}, \tag{23}$$

where $\mathbf{1}_n = (1, \dots, 1)^T$ denotes an n-dimensional one.

It can be seen that $\mathcal{C}_{sum} = \mathcal{C}_{\mathbf{w}^*}$ for $\mathbf{w}^* = \mathbf{1}_n$. As a consequence the complexity of $\mathcal{C}_{sum}$ is $\mathrm{VCdim}(\mathcal{C}_{sum}) = 2$, which is constant and independent from the dimensionality of the input space. Fixing all weights $w^{(i)} = a$ results in a ridged linear classifier, which no longer can alter the direction of its hyperplane. Only its distance to its origin and its polarization can be adapted. Let $c(\mathbf{x}) \in \mathcal{C}_{sum}$ be a randomly chosen classifier. Then

$$c(\mathbf{x}) = \mathbb{I}_{[\langle \mathbf{w},\mathbf{x}\rangle \geq t]} = \mathbb{I}_{[a\sum_{i=1}^{n} x^{(i)} \geq t]} = \mathbb{I}_{[an\bar{x} \geq t]} = \mathbb{I}_{[ax' \geq t]}, \tag{24}$$

where $\bar{x} \in \mathbb{R}$ denotes the sample-wise average of $\mathbf{x} \in \mathbb{R}^n$ and $x' = n\bar{x}$.

$\mathcal{C}_{sum}$ also induces an invariance against permutations of the features of $\mathbf{x}$, which cannot be achieved by any other non-constant linear classifier $\mathcal{C}_{lin} \backslash \mathcal{C}_{sum}$.

Theorem 1 *A non constant linear classifier $c \in \mathcal{C}_{lin}$ is invariant against permutations of feature entries*

$$f_\pi : \mathbf{x} \mapsto \mathbf{x}^{(\pi)} \tag{25}$$

with $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ and $\pi \in \mathcal{I}_n$ if and only if $c \in \mathcal{C}_{sum}$.

Proof of Theorem 1 In order to prove the invariance of a linear classifier to a certain type of data transformation f_θ, we have to prove that

$$\forall \mathbf{x} \forall \theta : \langle \mathbf{w}, f_\theta(\mathbf{x})\rangle \geq t \Longleftrightarrow \langle \mathbf{w}, \mathbf{x}\rangle \geq t. \tag{26}$$

$\Longrightarrow$: For $c \in \mathcal{C}_{sum}$ ($\mathbf{w} = a \cdot \mathbf{1}_n$) all permutations lead to the same projection

$$\forall \mathbf{x} \forall \pi : \langle \mathbf{w}, f_\pi(\mathbf{x})\rangle = a \cdot \sum_{i=1}^{n} x^{(\pi(i))} = an\bar{x}. \tag{27}$$

A classifier $c \in \mathcal{C}_{sum}$ is therefore invariant against feature permutations.

$\Longleftarrow$: The opposite direction is proven by providing a counterexample for each $c(\mathbf{x}) \in \mathcal{C}_{lin} \backslash \mathcal{C}_{sum}$ that fulfills

$$\forall c \exists \pi \exists \mathbf{x} : \langle \mathbf{w}, f_\pi(\mathbf{x})\rangle < t \Longleftrightarrow \langle \mathbf{w}, \mathbf{x}\rangle \geq t. \tag{28}$$

As $\mathcal{C}_{lin} = \mathcal{C}_{sum}$ for univariate input spaces ($n = 1$) a minimal example requires at least $n = 2$ dimensions. Additionally, there must be at least two weights $w^{(i)} \neq w^{(j)}$ (e.g., $w^{(i)} > w^{(j)}$). Let π_{ij} be an arbitrary permutation swapping features i and j. We construct $\mathbf{x} = (0, \dots, 0, x^{(i)}, 0, \dots, 0)^T$ with $x^{(i)} = x' \neq 0$ fulfilling

$$w^{(i)}x' \geq t \quad \text{and} \quad w^{(j)}x' < t. \tag{29}$$

As a consequence $c(\mathbf{x}) = 1$ and $c(\pi_{ij}(\mathbf{x})) = 0$. $\square$

3.1 Permutation-invariant SVMs (SVM$_{sum}$)

In this article, we will focus on the support vector machine (SVM) as a training algorithm for linear classifiers (Vapnik, 1998). It is designed to maximize the margin between the training samples and the linear hyperplane separating two classes.

The original SVM maximizes the margin by regularisation of $||\mathbf{w}||_2$. This can be summarised by the following constrained optimization problem

$$\min_{\mathbf{w},t,\xi} \quad \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{m}\xi_i \tag{30}$$

$$\text{s.t.} \quad \forall i : y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle - t) \geq 1 - \xi_i \tag{31}$$

$$\forall i : \xi_i \geq 0. \tag{32}$$

The class labels are assumed to be $\mathcal{Y} = \{-1, +1\}$. $C \in \mathbb{R}^+$ denotes the cost parameter that represents the trade-off between margin maximization and error minimization. The parameters ξ_i denote the slack variables, enabling SVMs to operate in the non-separable case by measuring the deviation from the ideal condition. An SVM operating on a general linear classifier $\mathcal{C}_{lin}$ will be denoted as SVM$_{lin}$.

The SVM can be adapted for different subclasses of $\mathcal{C}_{lin}$ (Lausser et al., 2020). For a fixed linear projection $\mathbf{w}^*$ the SVM can be modified for $\mathcal{C}_{\mathbf{w}^*}$ by projecting each data point $\mathbf{x}_i$ to $x_i' = \langle \mathbf{w}^*, \mathbf{x}_i\rangle \in \mathbb{R}$ before starting its training (or its validation). This simplifies the optimization as only a univariate $w \in \mathbb{R}$ is required. For $\mathcal{C}_{sum}$ this projection is given by $x_i' = n\bar{x}_i$. The corresponding SVM will be denoted as SVM$_{sum}$.

4 Experiments

We compared SVM$_{sum}$ and SVM$_{lin}$ in experiments on artificial datasets and real datasets. All experiments were performed in `R` using the `TunePareto` package (Müssel et al., 2012). For the training of all support vector machines, the `e1071` package was utilised (Meyer et al., 2020). A fixed cost parameter of $C = 1$ was chosen. All experiments were additionally repeated for two types of reference classifiers. Random forest classifiers (RF) with 100, 250, 500 trees (Breiman, 2001), and k-Nearest Neighbor classifiers (k-NN) with 1, 3, 5 neighbors, were chosen (Fix & Hodges, 1951), the corresponding figures can be found in the Appendix. The classifiers were compared in the absence and presence of permutations.

The accuracy difference of classifiers SVM$_{sum}$ and SVM$_{lin}$ is reported as

$$A_{diff}(\mathcal{V}) = Acc(c_{sum}, \mathcal{V}) - Acc(c_{lin}, \mathcal{V}), \tag{33}$$

where $c_{sum} \in \mathcal{C}_{sum}$ and $c_{lin} \in \mathcal{C}_{lin}$. A positive value denotes a higher accuracy of SVM$_{sum}$, and a negative one a higher accuracy of SVM$_{lin}$.

4.1 Permutation experiments

For experiments with permutations, we replaced the original validation set $\mathcal{V} = \{(\mathbf{x}_i', y_i')\}_{i=1}^{m'}$ by

$$\mathcal{V}' = \{(f_{i,p}(\mathbf{x}_i'), y_i')\}_{i=1}^{m'}, \tag{34}$$

where each test sample was individually affected by a permutation

$$f_{i,p}(\mathbf{x}) = \mathbf{x}^{(\pi_i)} \quad \text{with} \quad \pi_i \in \mathcal{I}_n \quad \text{and} \quad p \leq \frac{1}{n} \sum_{j=1}^{n} \mathbb{I}_{\left[j \neq \pi_i^{(j)}\right]}. \tag{35}$$

That is the original feature vector $\mathbf{x}$ is replaced by a permuted version $\mathbf{x}^{(\pi_i)}$. The influence of the permutation is regulated by the parameter $p \in [0, 1]$, which will be called permutation rate in the following. At most np randomly selected feature values are permuted. The corresponding percentage is chosen in $p \in \{0\%, 20\%, \dots, 100\%\}$. Note that $p = 0\%$ denotes a permutation-free experiment.

4.2 Experiments on artificial datasets

The classifiers were evaluated in various settings with and without permutations. Those settings are based on distinct artificial datasets of varying dimensionality $n \in \{10, 100, 500, 1000\}$. Each experiment was repeated 1000 times. In a single experiment, a classifier was adapted on a training set $\mathcal{T}$ and evaluated on a validation set $\mathcal{V}$ of 2×50 samples each. That is each datasets comprises 50 samples of class $y = 1$ and 50 samples of class $y = 0$. The samples themselves were drawn according to setting-specific distributions. See Table 1 for an overview. For a specific type of artificial data, results are summarised by the median value and the interquartile range (IQR).

4.2.1 Basic experiments

We conducted experiments on three types of permutation-free settings (Table 1), which we believe to be suitable for SVM$_{sum}$ as they resemble unweighted sums. The optimal decision boundaries of these settings correspond to linear classifiers. They, therefore, also lie in the range of the more general SVM$_{lin}$. All settings are parameterized by a free parameter d that can be used to adjust the overlap of the classes ($d > 0$). Here, a smaller value of d indicates a higher overlap of the classes. For larger values of d, the settings become linear separable thereby allowing for optimal accuracy of linear classifiers.

In the first one, the classifiers were applied to discriminate binary feature vectors $\mathbf{x} \in \mathbb{B}^n$. The individual entries of a sample were drawn (independent from their position) according

Table 1 Artificial datasets for basic experiments: The table provides the distributions of analyzed samples $\mathbf{x} \in \mathcal{X}$ in dependency on their class label $y \in \{+, -\}$ and their dimensionality n

Distribution	Sampling of $(\mathbf{x}, y)$ $\mathbf{x} = (x^{(1)}, \dots, x^{(n)})^T$	Input space $\mathcal{X}$	Parameter ranges (class-specific $y \in \{+, -\}$)
Bernoulli	$x^{(i)} \sim Be(p_y)$	$\mathcal{X} \subseteq \mathbb{B}^n$	$p_y \in \{0.05, 0.10, \dots, 0.95\}$
Gamma	$x^{(i)} \sim \Gamma(\alpha_y, \beta)$	$\mathcal{X} \subseteq \mathbb{R}^n$	$\alpha_y \in \{0.05, 0.10, \dots, 0.95\}$, $\beta = 1$
Gaussian	$x^{(i)} \sim \mathcal{N}(\mathbf{c}_y, \mathbf{I})$	$\mathcal{X} \subseteq \mathbb{R}^n$	$\mathbf{c}_+ = (c_+^{(1)}, \dots, c_+^{(n)})^T$
			$\mathbf{c}_- = \mathbf{c}_+ + d\mathbf{w}/\|\mathbf{w}\|_2$
			$c_+^{(i)} \sim \mathcal{U}(0, 10)$
			$w^{(i)} \sim \mathcal{U}(0, 1)$
			$d \in \{1, 1.1, \dots, 10\}$

Each training dataset and each test dataset comprises 2×50 samples to *Bernoulli* distributions with a class-specific probability of success p_y. The overlap can be regulated by $d = |p_+ - p_-|$. All features have the same influence on the class label.

In the second experiment, a comparable setting based on real-valued samples was analyzed $\mathbf{x} \in \mathbb{R}^n$. Here, the samples were drawn according to *Gamma* distributions with class-specific shape parameters α_y ($d = |\alpha_+ - \alpha_-|$).

The third experiment is based on *Gaussian* distributions with class-specific centroids $\mathbf{c}_y$. Those centroids are chosen randomly but have to fulfill a predefined distance d. As a consequence, the individual features have an individual influence on the optimal linear classifier.

4.2.2 Permutation experiments

For the permutation experiments, we generated artificial datasets comprising $0.25 \cdot n$ informative and $0.75 \cdot n$ uninformative features. In the case of an uninformative feature, the values were drawn according to a common distribution for both classes. The corresponding distributions and parameter settings are listed in Table 2. All experiments were repeated in the absence ($p = 0\%$) and presence ($p > 0\%$) of permutations.

4.3 Experiments on real datasets

We additionally performed experiments on the COSMIC Cell Line Gene Mutation Profiles dataset (Forbes et al., 2010). Its main characteristics are provided in Table 3.

COSMIC, the Catalogue Of Somatic Mutations In Cancer, provides comprehensive information on somatic mutations in human cancers. The cell line gene mutation profiles dataset summarises gene mutations in cancer cell lines from low-throughput or high-throughput studies. Classes with less than 9 samples were removed. Overall, the dataset comprises $n = 17{,}972$ binary variables (gene mutated yes/no) for $m = 728$ samples from $|\mathcal{Y}| = 20$ different tissues.

We compared the performance of SVM$_{sum}$ and SVM$_{lin}$ systematically in binary classification experiments for each pair of classes of the COSMIC Cell Line Gene Mutation Profiles dataset (190 experiments). Those experiments were designed as 10×10 cross-validation experiments (Japkowicz & Shah, 2011).

Table 2 Artificial datasets for permutation experiments: The table provides the parameter settings for generating informative and uninformative features for the permutation experiment

Distribution	Parameter ranges	
	Class specific $y \in \{+,-\}$	Uninformative $y = \circ$
Bernoulli	$p_+ = 0.05, p_- = 0.8$	$p_\circ = 0.5$
Gamma	$\alpha_+ = 0.05, \alpha_- = 0.8$	$\alpha_\circ = 0.5$
Gaussian	$\mathbf{c}_+ = (c_+^{(1)}, \dots, c_+^{(n)})^T, c_+^{(i)} \sim \mathcal{U}(0, 10)$	$\mathbf{c}_\circ = (0, \dots, 0)^T$
	$\mathbf{c}_- = \mathbf{c}_+ + d\mathbf{w}/\|\mathbf{w}\|_2, w^{(i)} \sim \mathcal{U}(0, 1)$	

The informative features use distinct distributions for each class label $y \in \{+,-\}$. The uninformative ones use a common distribution for both classes. Each training dataset and each test dataset comprises 2×50 samples

Table 3 Summary of the COSMIC Cell Line Gene Mutation Profiles dataset (Forbes et al., 2010)

Class	Description	Samples	Mutational burden (%)
y_1	Autonomic ganglia	33	2.54
y_2	Bone	35	2.03
y_3	Breast	43	3.33
y_4	Central nervous system	53	2.24
y_5	Cervix	11	4.28
y_6	Endometrium	9	10.20
y_7	Haematopoietic and lymphoid tissue	123	4.89
y_8	Kidney	27	2.83
y_9	Large intestine	41	9.74
y_{10}	Liver	9	2.83
y_{11}	Lung	136	3.71
y_{12}	Oesophagus	23	4.11
y_{13}	Ovary	33	3.76
y_{14}	Pancreas	17	1.89
y_{15}	Skin	46	4.70
y_{16}	Soft tissue	18	4.24
y_{17}	Stomach	20	4.95
y_{18}	Thyroid	11	2.64
y_{19}	Upper aerodigestive tract	22	2.57
y_{20}	Urinary tract	18	3.08

The table gives the class labels y_i, a short description, the number of samples per class, and the mean mutational burden per class. All samples are given by mutation profiles of $n = 17{,}972$ binary variables (mutation yes/no)

4.3.1 Permutation experiments

Similar to the artificial datasets, the real datasets were analyzed in the absence ($p = 0\%$) and presence of permutations ($p > 0\%$). The permutations were introduced as defined in Eqs. 34 and 35 and again applied to the validation sets of the 10×10 cross-validation experiments.

5 Results

The results of the experiments described in Sect. 4 are summarized here. The evaluation of the artificial experiments is given in Sect. 5.1. The results for the real dataset can be found in Sect. 5.2.

5.1 Results on artificial datasets

The following section provides the results achieved on the artificial datasets.

Table 4 Table summary on basic experiments (SVM$_{sum}$ vs SVM$_{lin}$): The table provides the achieved median accuracy differences and the corresponding interquartile range for each data type and each dimensionality n

Distribution	Dimensionality	Differences in accuracy		Top differences	
		Median (%)	IQR $[q_{0.25}, q_{0.75}]$	Top d	Median acc (%)
Bernoulli	n = 10	1.0	[0.0%, 4.0%]	0.10	4.0
	n = 100	0.0	[0.0%, 6.0%]	0.05	13.0
	n = 500	0.0	[0.0%, 0.0%]	0.05	17.0
	n = 1000	0.0	[0.0%, 0.0%]	0.05	15.0
Gamma	n = 10	2.0	[0.0%, 5.0%]	0.10	3.0
	n = 100	2.0	[0.0%, 9.0%]	0.10	13.0
	n = 500	0.0	[0.0%, 4.0%]	0.05	18.0
	n = 1000	0.0	[0.0%, 1.0%]	0.05	21.0
Gaussian	n = 10	0.0	[−1.0%, 1.0%]	0.5	3.0
	n = 100	0.0	[0.0%, 5.0%]	1.3	11.0
	n = 500	2.0	[0.0%, 10.0%]	1.6	16.0
	n = 1000	3.0	[0.0%, 13.0%]	1.8	19.0

Additionally, the parameter difference d with the highest median accuracy differences is reported

5.1.1 Results for basic experiments

The results gained for the comparison of SVM$_{sum}$ and SVM$_{lin}$ on the basic permutation-free settings are presented in Fig. 4 and summarized in Table 4. The parameter difference d with the highest median accuracy difference is also reported. The comparisons to the other reference classifiers can be found in Supplementary Fig. 7.

For the Bernoulli distributed data (Fig. 4A), SVM$_{sum}$ achieved non-negative median accuracy differences for all parameter differences. For higher dimensionalities, positive accuracy differences are concentrated on smaller parameter differences d. The highest median accuracy difference was observed for the smallest parameter difference $d = 0.05$. For $n = 100$ it is equal to 13.0%. It increased to 17.0% for $n = 500$. A median value of 15.0% was observed for $n = 1000$. Similar patterns were observed for the reference classifiers RF and k-NN, where in general the k-NNs were more affected by changes in d.

For the Gamma distributed data (Fig. 4B), the results were comparable. Increasing performance differences again occurred for higher dimensionality. The highest median differences were observed for smaller parameter differences. For $n = 100$, the highest median difference (13.0%) was achieved for $d = 0.10$. For $n = 500$ a value of 18.0% was achieved for $d = 0.10$. For $n = 1000$, a median difference of 21.0% was observed for $d = 0.05$. In this setting, RFs and k-NNs show different behaviour. While the accuracy differences against RFs are negative for some experiments of $n = 10$, the k-NNs are outperformed in almost all settings.

Equivalent patterns were identified for the Gaussian distributed data (Fig. 4C). For $n = 100$ the highest median difference of 11.0% occurred for $d = 1.3$. For $n = 500$ it increased to 16.0% occurred for $d = 1.6$ and for $n = 1000$ it finally achieved 19.0% for $d = 1.8$. For all reference classifiers, no negative median accuracy difference was observed.

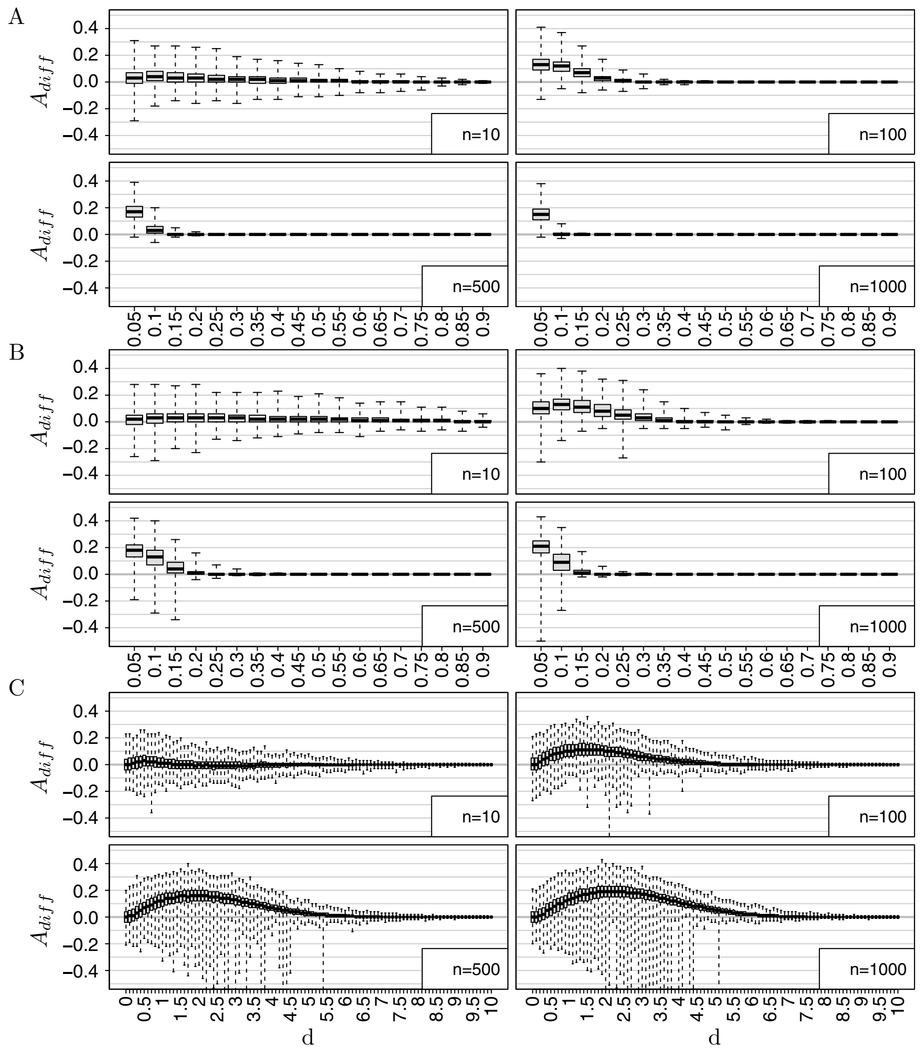

Fig. 4 Results of basic experiments. The figure shows boxplots for the accuracy differences of SVM$_{sum}$ and SVM$_{lin}$ on the Bernoulli, Gamma, and Gaussian distributed datasets (**A**–**C**). Positive values indicate a better performance of SVM$_{sum}$. Results for different dimensionalities $n \in \{10, 100, 500, 1000\}$ are shown. Each boxplot summarizes the results of 1000 repeated experiments for a specific parameter difference d

Overall experiments, the use of SVM$_{sum}$ led to a decline of more than 5% accuracy only in a small number of experiments (Bernoulli: 0.6%, Gamma: 0.7%, Gaussian: 1.7%). For the RFs and k-NNs these numbers do not exceed 1.6%. An exception is the Gamma setting for RFs. Due to the effects in $n = 10$, the corresponding numbers are between 8.9 and 9.8%.

Table 5 Summary of permutation experiments (SVM$_{sum}$ vs SVM$_{lin}$): The table provides the achieved median accuracy differences and the corresponding interquartile range

Distribution	Dimensionality	Differences in accuracy		Top differences	
		Median (%)	IQR $[q_{0.25}, q_{0.75}]$	Top d	Median acc (%)
Bernoulli	n = 10	−7.0	[−18.0%, 4.0%]	$p = 1.0$	11.0
	n = 100	0.0	[−2.0%, 8.0%]	$p = 1.0$	21.0
	n = 500	0.0	[0.0%, 0.0%]	$p = 1.0$	11.0
	n = 1000	0.0	[0.0%, 0.0%]	$p = 1.0$	6.0
Gamma	n = 10	−11.0	[−20.0%, −1.0%]	$p = 1.0$	6.0
	n = 100	−2.0	[−7.0%, 11.0%]	$p = 1.0$	22.0
	n = 500	0.0	[0.0%, 2.0%]	$p = 1.0$	14.0
	n = 1000	0.0	[0.0%, 0.0%]	$p = 1.0$	8.0
Gaussian	n = 10	7.0%	[−10.0%, 20.0%]	$p = 1.0$	26.0
	n = 100	27.0	[16.8%, 31.0%]	$p = 0.6$	30.0
	n = 500	29.0	[24.0%, 32.0%]	$p = 0.4$	30.0
	n = 1000	30.0	[25.0%, 33.0%]	$p = 0.4$	31.0

Additionally, the parameter difference with the highest median accuracy differences is reported

5.1.2 Results for permutation experiments

The results for the comparison of SVM$_{sum}$ and SVM$_{lin}$ on the permutation experiments are given in Fig. 5. A summary can be found in Table 5. The results for the reference classifiers can be found in Supplementary Fig. 8. The underlying datasets of these experiments comprise informative and uninformative features. Both will equivalently influence SVM$_{sum}$ as its structural properties do not allow the selection of individual features. This is of course a disadvantage in the absence of permutations ($p = 0$). However, the invariance property of SVM$_{sum}$ becomes an advantage for increasing permutation rates ($p > 0$). The highest median accuracy differences were in general achieved for the highest permutation levels ($p = 1$).

For the Bernoulli distributed data (Fig. 5A) the highest median accuracy differences start with 11.0% at $n = 10$ reach their overall maximum of 21.0% at $n = 100$ and afterward decline to 11.0% and 6.0% for higher dimensionalities ($n = 500$ and $n = 1000$). Negative median accuracy differences become rare with increasing dimensionality.

Comparable results were found for the Gamma distributed data (Fig. 5B). Here the highest median accuracy differences start with 6.0% for $n = 10$, increase to 22.0% at $n = 100$, and afterward decline to 14.0% for $n = 500$ and 8.0% for $n = 1000$. Negative median accuracy differences were observed up to a dimensionality of $n = 100$.

For the Gaussian distributed data (Fig. 5C) positive effects of the permutation invariance were observed for smaller values of p than for the former two settings. For all $n > 10$ positive median accuracy differences were observed for all $p > 0$. In this case, negative median accuracy differences only occurred for the permutation-free experiments ($p = 0$). The highest median accuracy differences were observed for $p = 1.0$ ($n = 10$), $p = 0.6$ ($n = 100$), $p = 0.4$ ($n = 500$ and $n = 1000$). For $n = 10$ a level of 26.0% was achieved. The remaining dimensionalities form a plateau at 30.0%.

In general, for lower dimensionalities, the SVM$_{lin}$ can counteract the effects of permutations for smaller values of p. Here better results for the general classifier can be found.

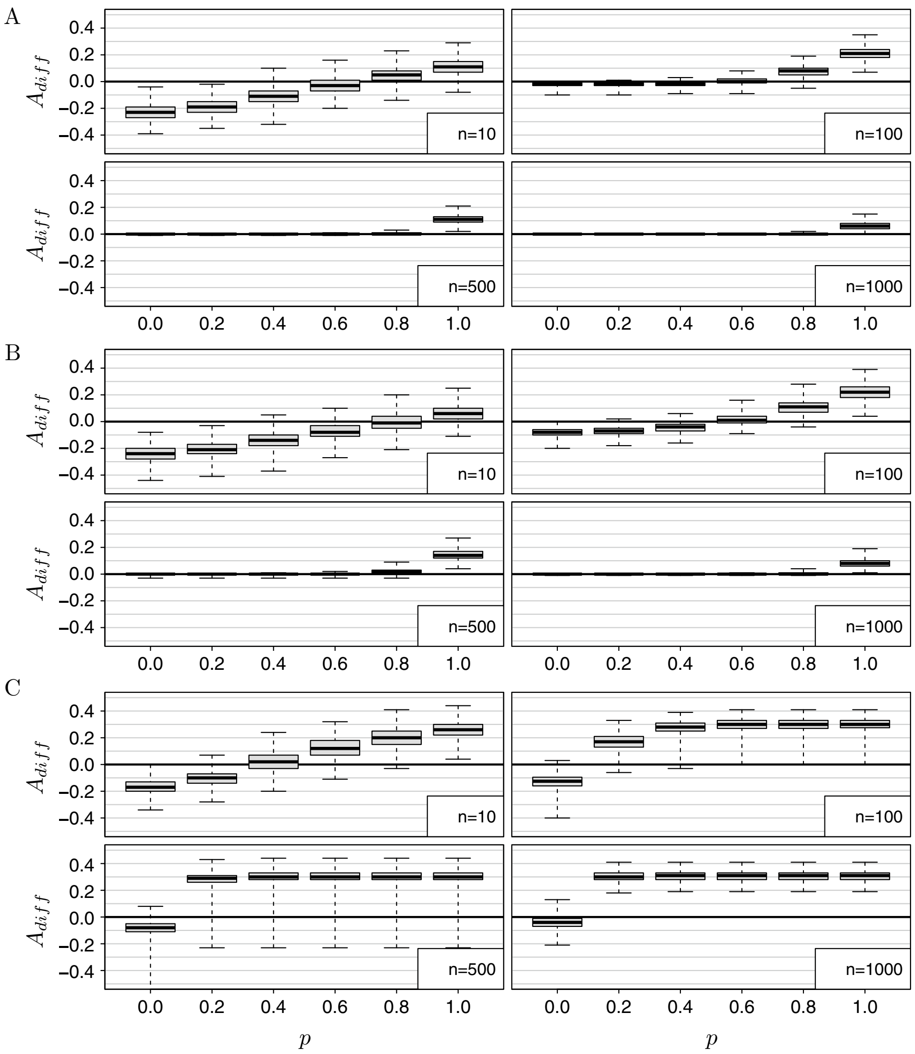

Fig. 5 Results of permutation experiments. The figure shows boxplots for the accuracy differences of SVM$_{sum}$ and SVM$_{lin}$ on the Bernoulli, Gamma, and Gaussian distributed datasets (**A**–**C**). Positive values indicate a better performance of SVM$_{sum}$. Results for different dimensionalities $n \in \{10, 100, 500, 1000\}$ are shown. Each boxplot summarises the results of 1000 repeated experiments for a specific permutation rate $p \in [0, 1]$. Note that $p = 0$ corresponds to a permutation-free experiment

Comparable patterns can be observed for the RFs and the k-NNs, where the effect disappears in earlier stages for k-NNs than for SVM$_{lin}$ and RFs. A reason might be the training of the classifiers on permutation-free training samples $\mathcal{T}$.

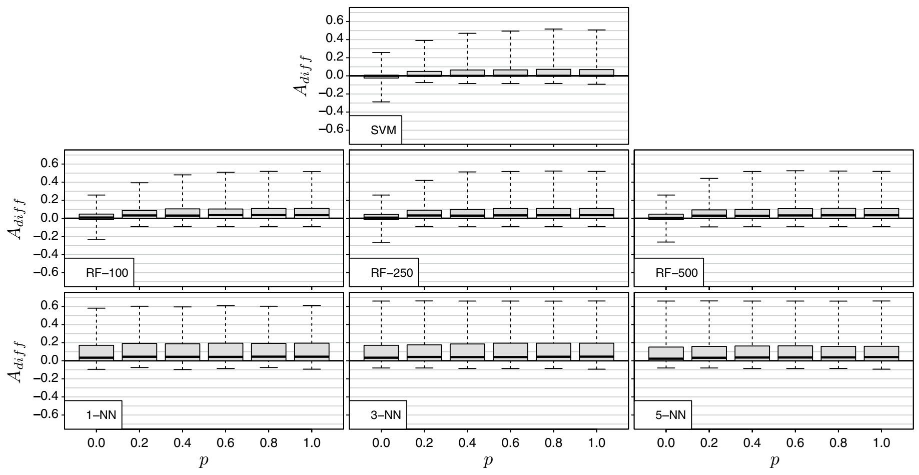

Fig. 6 Results for permutation experiments (reference classifiers). The figure summarises all experiments conducted for all pairs of classes $\{y_i, y_j\} \in \mathcal{Y}$ of the COSMIC dataset ($|\mathcal{Y}| = 20$). It shows boxplots for the accuracy differences of the reference classifiers and SVM$_{sum}$. SVM$_{lin}$, RF, and k-NN classifiers are shown. Positive values indicate a better performance of SVM$_{sum}$. Results for different dimensionalities $n \in \{10, 100, 500, 1000\}$ are shown. Each boxplot summarises the results of 1000 repeated experiments for a specific permutation rate $p \in [0, 1]$. Note that $p = 0$ corresponds to a permutation-free experiment

5.2 Results on real datasets

The results of the 10×10 CV experiments with the COSMIC Cell Line Gene Mutation Profiles dataset (Forbes et al., 2010) are given in Fig. 6. It comprises the accuracy differences for the experiments with all 190 pairs of classes and all permutation rates.

5.2.1 Results for permutation-free experiments ($p = 0$)

In the permutation-free experiments ($p = 0$), the SVM$_{lin}$ was not able to outperform SVM$_{sum}$ in 60.00% of all cases. However, both classifiers showed an equal median performance, which is indicated by a median accuracy difference of 0%. The IQR ranges from $[-2.4\%, 0.6\%]$. Against the RFs, the median accuracy difference varied from 0.5 to 0.8%. The first quartile ranged from -1.4 to -1.5% and the third quartile ranged from 4.5 to 4.6%. Compared to k-NNs median accuracy differences in the range of 2.3% and 3.4% were achieved. All first quartiles lie at 0% while the third quartiles lie between 15.1 and 17.1%.

5.2.2 Results for permutation experiments ($p > 0$)

With the introduction of permutations ($p > 0$), SVM$_{lin}$ was able to outperform SVM$_{sum}$ less frequently. With an increasing value of p, the accuracy difference also increased in favor of SVM$_{sum}$. For $p = \{0.2, 0.4, 0.6, 0.8\}$, SVM$_{sum}$ performed worse in 23.7%, 21.1%, 20.0%, and 19.0% of all experiments, respectively. When all features are permuted, SVM$_{sum}$ outperforms SVM$_{lin}$ in 50.0% and achieves equal accuracy in 31.0% of all cases. The RFs outperform the SVM$_{sum}$ only in 19.0% up to 24.2% of all cases. For the k-NNs these numbers range from 12.1 to 16.3%

6 Discussion and conclusion

In this work, we extended the canon of invariant concept classes by the concept class of permutation-invariant linear classifiers. The structural properties of these classifiers guarantee predictions that are immune against disordered feature representations and therefore might be preferable in the presence of this type of noise. However, they might also be restrictive as their invariance property is directly coupled to a fixed linear projection towards the sample-wise centroids.

Our experiments identified scenarios in which the permutation-invariant SVM was able to outperform its non-invariant counterpart. Interestingly, not all of these scenarios were affected by permutations. In this context, the permutation-invariant SVM can be seen as a randomly chosen fixed projection SVM. Applied to the correct context it can achieve better results than the unguided large margin criterion. Coupled with a smaller (and constant) VC-dimension the identification of a fixed projection seems to be worthwhile. However, this projection has to be selected a priori based on external domain knowledge or independent experiments. It has to be chosen during the design phase of a classifier before starting its training. Otherwise, the data-driven adaptation (on the training set) would lead to the adaptation of a general linear classifier. Although it might be unrealistic for de novo experiments, it might be realistic for larger research contexts with multiple (related) classes (Pan & Yang, 2010; Lausser et al., 2018b).

In the basic permutation-free scenario, performance differences were especially detected in settings with highly overlapping class-wise distributions. They were in general in favor of the permutation-invariant SVM. This result is a direct consequence of the declined performance of the standard version of the SVM. It might be partially explained by an overadaptation of this more flexible model. Capable of adapting to potentially more informative features the standard SVM might be misled by the relatively small number of available training samples. In contrast, the permutation-invariant version has to rely on the overall sample-wise mean values of its training samples, which can only provide a rough overview of the characteristics of the classes. Although in general suboptimal for the tested classification problems, this simpler and more ridged learning task seems to be more suitable for the limited amount of training samples and leads to a higher generalization performance.

Other settings with well-separable class-wise distributions allowed high classification performance for both types of classifiers. While the standard SVM is designed to take advantage of a large margin between two classes the good performance of the permutation-invariant SVM might not be obvious. However, if the projections of both classifiers are not orthogonal to each other a larger margin for the standard SVM directly implies a larger margin for the permutation-invariant version. In other words, a larger margin for the standard SVM increases the differences in the sample-wise means of the two classes. Otherwise, if the projections of both classifiers are orthogonal to each other the large margin of the standard SVM does not influence the permutation-invariant version at all. In this case, the permutation-invariant SVM is likely to lead to severely declined performance.

As expected, the permutation-invariant SVM was not affected by permuting the feature representation of individual samples. As both its training algorithm and its trained classifier are invariant against this type of data transformation its classification performance is constant within each permutation experiment. It also allows for high classification performance in these settings. In contrast, the standard SVM (based on general linear classifiers) can severely suffer from affecting training and test samples. While affected training samples distract the standard SVM's focus from potentially informative features, affected test samples no longer fulfill patterns identified for the original data. Interestingly, the reference classifiers were able to counteract lower permutation rates, which was most prominent in low-dimensional settings. A reason might be the design of our study. While we applied permutations to the validation samples, the training samples were not modified. This allows feature-selecting classifiers to deselect uninformative features and to concentrate on a small set of informative ones. If the random permutation swaps two uninformative features it will not be recognized by these classifiers.

We observed similar behavior for our experiments with the mutation profiles of the COSMIC dataset, where again the performance of the standard SVM decreased with increasing permutation rates. However, these effects can be more decisive than in the well-controlled artificial experiment on binary profiles. A possible reason might be the more complex structure of the mutation dataset. Besides their high dimensionality ($n \gg m$), the underlying mutation profiles can also be seen as sparse data as they contain in mean between 1.9 and 10.2% mutations per class. The dataset therefore draws per se the attention of feature-sensitive classifiers to a smaller subset of features (whether they are informative or not). Those preselected features are likely to be depleted by permutations.

In contrast to the corresponding artificial datasets, we can not assume the cancer types in the COSMIC dataset to be simply characterized by their mutational burden. Specific patterns of cancer development must be assumed. However, these patterns might be hard to detect due to the dimensionality of mutation profiles. Based on an analysis of the mutational burden the permutation-invariant SVM achieved at least the same accuracy in 114 of 190 permutation-free settings of our experiments. This also strongly supports the usage of tumor mutational burden (Stenzinger et al., 2019), a recently identified biomarker, as part of a diagnosis. Differences in the mutational burden of tumor entities were also observed in large-scale studies (Chalmers et al., 2017) and were shown to predict survival after immunotherapy across multiple cancer types (Samstein et al., 2019).

This said we believe the concept class of permutation-invariant linear classifiers to be a concept class with unique structural properties and its dedicated field of application. However, it is also limited in its possibility of taking different types of information into account. In the presence of large training sets and in the absence of permutations other classification models might be preferable. Nevertheless, the permutation-invariant SVM showed comparable performance to the standard linear counterpart in most of our test scenarios. It is therefore a strong reference classifier for other more complex approaches.

As we are mainly interested in an outline of the concept class of permutation-invariant linear classifiers, we needed to decide on a training algorithm for an empirical evaluation. The SVM was chosen as a state-of-the-art training algorithm for the concept class of linear classifiers. However, other training algorithms for the concept class of linear classifiers exist and they might be adapted for the permutation-invariant subclass (Bishop, 2006; Abe, 2010; Hastie et al., 2001). The performance of these classifiers must be evaluated in different experiments.

The baseline performance achieved by any permutation-invariant linear classifier is also of interest for theoretical purposes. In contrast to other classifiers, the permutation-invariant ones react identically in the absence or presence of permutation rates of up to 100%. They are totally unaware of the assignment of features to values. Their baseline performance could be used to estimate the benefit of this assignment in the form of performance differences.

From a structural point of view, the concept class of permutation-invariant linear classifiers evaluates an unweighted sum or (in case of binary features) an unweighted majority vote (Kuncheva et al., 2003). It might be misguided by a large number of uninformative features. It is therefore more suitable for the application on a semantically preselected panel of measurements (Taudien et al., 2016). In future work, we will address this point by coupling permutation-invariant linear classifiers to feature selection processes and feature-selecting multi-classifier systems in order to identify a suitable tradeoff between permutation invariance and generalization ability.

Appendix: Results on artificial datasets (reference classifiers)

The following appendix provides the results achieved by the reference classifiers on the artificial datasets. The accuracy difference towards SVM_{sum} is shown.

- Figure 7: Results for basic experiments (reference classifiers).
- Figure 8: Results for permutation experiments (reference classifiers).

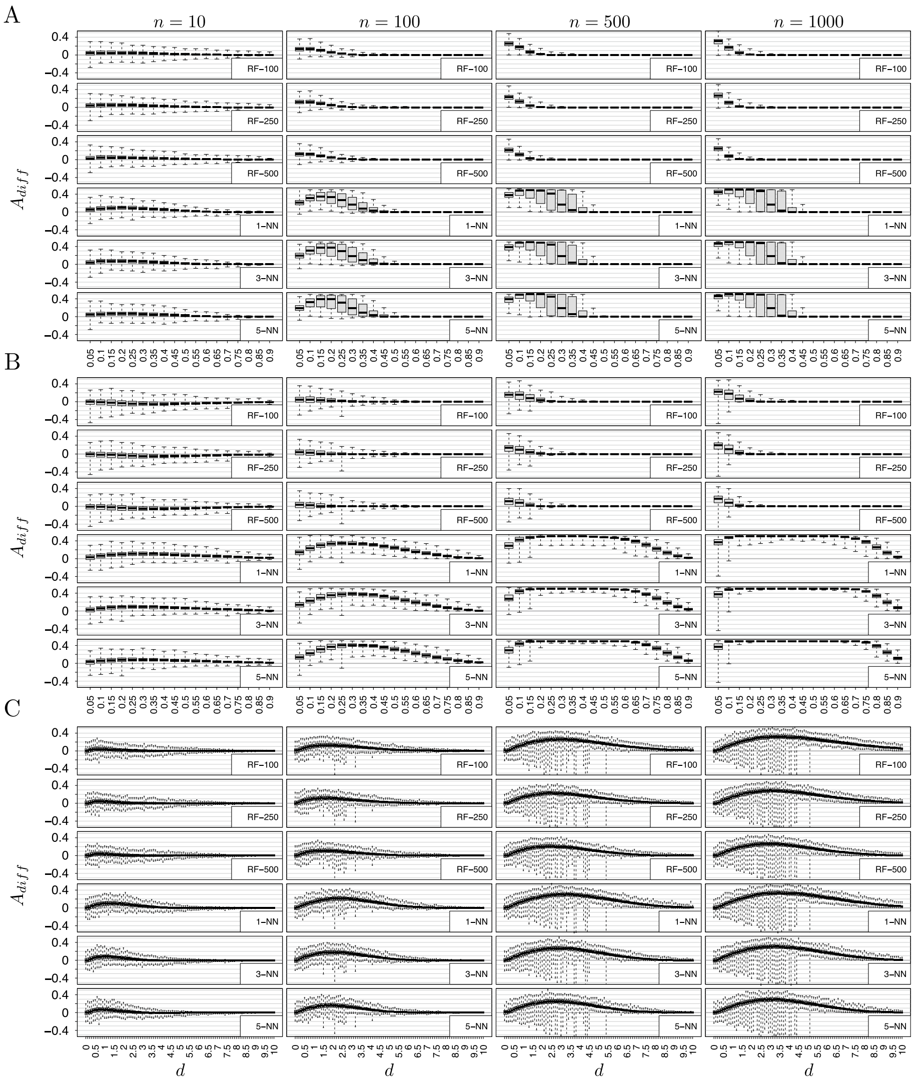

Fig. 7 Results for basic experiments (reference classifiers). The figure shows boxplots for the accuracy differences of SVM$_{sum}$, and the reference classifiers on the Bernoulli, Gamma, and Gaussian distributed datasets (**A**–**C**). RF and k-NN classifiers are shown. Positive values indicate a better performance of SVM$_{sum}$. Results for different dimensionalities $n \in \{10, 100, 500, 1000\}$ are shown. Each boxplot summarizes the results of 1000 repeated experiments for a specific parameter difference d

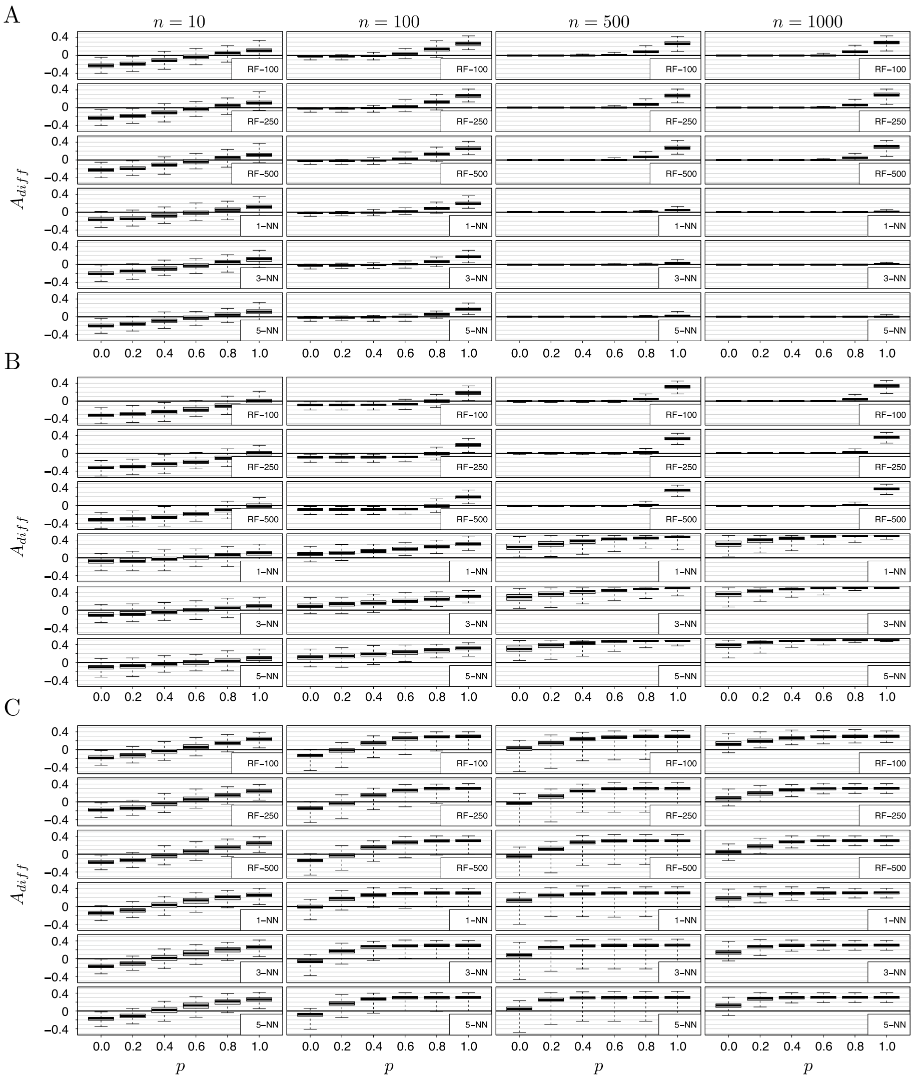

Fig. 8 Results for permutation experiments (reference classifiers). The figure shows boxplots for the accuracy differences of SVM$_{sum}$ and the reference classifiers on the Bernoulli, Gamma, and Gaussian distributed datasets (**A**–**C**). RF and k-NN classifiers are shown. Positive values indicate a better performance of SVM$_{sum}$. Results for different dimensionalities $n \in \{10, 100, 500, 1000\}$ are shown. Each boxplot summarises the results of 1000 repeated experiments for a specific permutation rate $p \in [0, 1]$. Note that $p = 0$ corresponds to a permutation-free experiment

Author contributions L.L. conceived the idea and performed theoretical analysis, L.L. and R.S. conceived the experiments, R.S. performed data acquisition, L.L. and R.S. analyzed the results, L.L. implemented the algorithms, L.L. drafted the manuscript, H.A.K. supervised and guided the study. L.L., R.S., and H.A.K. wrote the manuscript. All authors reviewed the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. The research leading to these results has received funding from the German Research Foundation (DFG, SFB 1074 project Z1; SFB 1506 project B6), and TRANSCAN VI - PMTR-pNET, ID 01KT1901B) all to HAK.

Data availability The data used in this work is all public.

Code availability Code will be made available after publication.

Declarations

Conflict of interest The authors declare not conflict of interest.

Ethical approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Abe, S. (2010). *Support vector machines for pattern classification*. Berlin: Springer.

Anthony, M. H. G., & Biggs, N. (1997). *Computational learning theory* (Vol. 30). Cambridge: Cambridge University Press.

Bishop, C. M. (2006). *Pattern recognition and machine learning (information science and statistics)*. Berlin: Springer.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery, 2*(2), 121–167.

Chalmers, Z. R., Connelly, C. F., Fabrizio, D., et al. (2017). Analysis of 100,000 human cancer genomes reveals the landscape of tumor mutational burden. *Genome Medicine, 9*(1), 34.

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers, 3*, 326–334.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics, 7*(2), 179–188.

Fix, E., & Hodges, J. L. (1951). Discriminatory analysis: Nonparametric discrimination: Consistency properties. Tech. Rep. Project 21-49-004, Report Number 4, USAF School of Aviation Medicine, Randolf Field, Texas.

Forbes, S. A., Bindal, N., & Bamford, S., et al. (2010). Cosmic: mining complete cancer genomes in the catalogue of somatic mutations in cancer. *Nucleic Acids Research 39(suppl_1):D945–D950.*

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139.

Haasdonk, B., & Burkhardt, H. (2007). Invariant kernel functions for pattern analysis and machine learning. *Machine Learning, 68*(1), 35–61.

Hastie, T., Tibshirani, R., & Friedman, J. H. (2001). *The elements of statistical learning*. New York: Springer.

Japkowicz, N., & Shah, M. (2011). *Evaluating learning algorithms: A classification perspective*. New York: Cambridge University Press.

Jolliffe, I. (1986). *Principal component analysis*. New York: Springer.

Kestler, H. A., Lausser, L., Lindner, W., et al. (2011). On the fusion of threshold classifiers for categorization and dimensionality reduction. *Computational Statistics, 26*(2), 321–340.

Kraus, J. M., Lausser, L., Kuhn, P., et al. (2018). Big data and precision medicine: Challenges and strategies with healthcare data. *International Journal of Data Science and Analytics, 6*(3), 241–249.

Kuncheva, L. I., Whitaker, C. J., Shipp, C. A., et al. (2003). Limits on the majority vote accuracy in classifier fusion. *Pattern Analysis & Applications, 6*(1), 22–31.

Lausser, L., & Kestler, H. A. (2010). Robustness analysis of eleven linear classifiers in extremely high-dimensional feature spaces. In: Schwenker, F., Gayar, N. E. (Eds.), *Artificial neural networks in pattern recognition* (Vol LNAI 5998, pp. 72–83).

Lausser, L., & Kestler, H. A. (2014). Fold change classifiers for the analysis for the analysis of gene expression profiles. In: Gaul, W., Geyer-Schulz, A., Baba, Y., et al. (Eds.), *proceedings volume of the German/Japanese workshops in 2010 (Karlsruhe) and 2012 (Kyoto), studies in classification, data analysis, and knowledge organization* (pp. 193–202).

Lausser, L., Schäfer, L. M., Schirra, L. R., et al. (2019). Assessing phenotype order in molecular data. *Scientific Reports, 9*(1), 11746.

Lausser, L., Schmid, F., Schirra, L. R., et al. (2018). Rank-based classifiers for extremely high-dimensional gene expression data. *Advances in Data Analysis and Classification, 12*(4), 917–936.

Lausser, L., Szekely, R., Klimmek, A., et al. (2020). Constraining classifiers in molecular analysis: Invariance and robustness. *Journal of the Royal Society Interface, 17*(163), 20190612.

Lausser, L., Szekely, R., Schirra, L. R., et al. (2018). The influence of multi-class feature selection on the prediction of diagnostic phenotypes. *Neural Processing Letters, 48*(2), 863–880.

L'Heureux, A., Grolinger, K., Elyamany, H. F., et al. (2017). Machine learning with big data: Challenges and approaches. *IEEE Access, 5*, 7776–7797.

Meyer, D., Dimitriadou, E., & Hornik, K., et al. (2020). e1071: Misc functions of the department of statistics, probability theory group (Formerly: E1071), TU Wien. https://CRAN.R-project.org/package=e1071, r package version 1.7-4.

Minsky, M., & Papert, S. A. (1988). *Perceptrons: An introduction to computational geometry*. Cambridge: MIT Press.

Müssel, C., Lausser, L., Maucher, M., et al. (2012). Multi-objective parameter selection for classifiers. *Journal of Statistical Software, 46*(5), 1–27.

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering, 22*(10), 1345–1359.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386.

Samstein, R. M., Lee, C. H., Shoushtari, A. N., et al. (2019). Tumor mutational load predicts survival after immunotherapy across multiple cancer types. *Nature Genetics, 51*(2), 202–206.

Schmid, F., Lausser, L., & Kestler, H. A. (2014). Linear contrast classifiers in high-dimensional spaces. In El Gayar, N., Schwenker, F., Suen, C. (Eds.), *IAPR workshop on artificial neural networks in pattern recognition* (pp. 141–152).

Singh, D., & Singh, B. (2020). Investigating the impact of data normalization on classification performance. *Applied Soft Computing, 97*(105), 524.

Stenzinger, A., Allen, J. D., Maas, J., et al. (2019). Tumor mutational burden standardization initiatives: Recommendations for consistent tumor mutational burden assessment in clinical samples to guide immunotherapy treatment decisions. *Genes, Chromosomes and Cancer, 58*(8), 578–588.

Stone, J. (2004). *Independent component analysis: A tutorial introduction*. Cambridge: MIT Press.

Taudien, S., Lausser, L., Giamarellos-Bourboulis, E. J., et al. (2016). Genetic factors of the disease course after sepsis: Rare deleterious variants are predictive. *EBioMedicine, 12*, 227–238.

Valente, E., & Rocha, M. (2015). Integrating data from heterogeneous DNA microarray platforms. *Journal of Integrative Bioinformatics, 12*, 281.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM, 27*(11), 1134–1142.

Vapnik, V. (1998). *Statistical learning theory*. New York: Wiley.