KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

DRAFT

To remove, add 'final' to class options

The Title of Your PhD Dissertation

Marc Claesen

Supervisor:
Prof. dr. ir. Bart De Moor
Prof. dr. ir. Frank De Smet, co-supervisor

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering Science

November 2015

The Title of Your PhD Dissertation

Marc CLAESEN

Examination committee:
Prof. dr. ir. The Chairman, chair
Prof. dr. ir. Bart De Moor, supervisor
Prof. dr. ir. Frank De Smet, co-supervisor
Prof. dr. ir. The One
Prof. dr. ir. The Other
Prof. dr. External Jurymember
(Far Away)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering Science

November 2015

Preface

{ch:preface}

...

Instructies van de faculteit:

In het voorwoord wordt de algemene doelstelling van het werk samengevat in enkele regels en worden personen, diensten of firma's bedankt voor hun medewerking bij het tot stand komen van het werk.

De naam van firma's en personen uit deze firma's mogen slechts worden vermeld mits hun uitdrukkelijke toelating én na overleg met de supervisor(en)! Steeds wordt de supervisor(en) vermeld, de verantwoordelijke en eventueel de personen die rechtstreeks geholpen hebben bv. door het ter beschikking stelling van meetresultaten, faciliteiten. Ook de instantie die eventueel een doctoraatsbeurs heeft toegekend wordt bedankt (bv. FWO, IWT, ...).

Abstract

{ch:abstract}

...

Instructies van de faculteit:

In een beknopte tekst van maximum 2 pagina's worden de belangrijkste doelstellingen en besluiten geformuleerd, zowel in het Nederlands als in het Engels. Zulke samenvattingen kunnen worden gebruikt in wetenschappelijke verslagen van het departement of de faculteit. Het Engels moet vlekkeloos zijn.

Beknopte samenvatting

. . .

we still have to write the dutch version!

Instructies van de faculteit:

In een beknopte tekst van maximum 2 pagina’s worden de belangrijkste doelstellingen en besluiten geformuleerd, zowel in het Nederlands als in het Engels. Zulke samenvattingen kunnen worden gebruikt in wetenschappelijke verslagen van het departement of de faculteit. Het Engels moet vlekkeloos zijn.

Abbreviations

MD molecular dynamics

List of Symbols

Θ	A nice symbol

Contents

List of Figures

List of Tables

Todo list

we still have to write the dutch version! v

Instructies van de faculteit:

De hoofdstukken: Elk hoofdstuk is ingelast met een bepaald doel voor ogen. Dit doel wordt vermeld in de eerste paragraaf van elk hoofdstuk. Naargelang de aard van de tekst (experiment, uitvoering, theoretische ontwikkeling, ...) volgen de paragrafen elkaar op. Beweringen worden altijd gestaafd, hetzij door eigen experimenten, hetzij door een theoretische afleiding, hetzij door verwijzingen naar de literatuur. Elk hoofdstuk eindigt met een kort samenvattend besluit waarbij nagegaan wordt in hoeverre de doelstelling van het betrokken hoofdstuk verwezenlijkt is. De deelbesluiten moeten de lezer automatisch leiden naar het algemeen besluit aan het einde van het werk.

Chapter 1

This is introduction

{ch:introduction}

Instructies van de faculteit:

De inleiding situeert de problematiek, beschrijft de stand van de huidige kennis terzake, omschrijft de voornaamste doelstelligen van het werk, samen met de beperkende randvoorwaarden en de ter beschikking gestelde middelen en poneert de belangrijkste stellingen.

Illustration of how to include citations. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

And yet another citation.

Introducing some symbol: Θ.

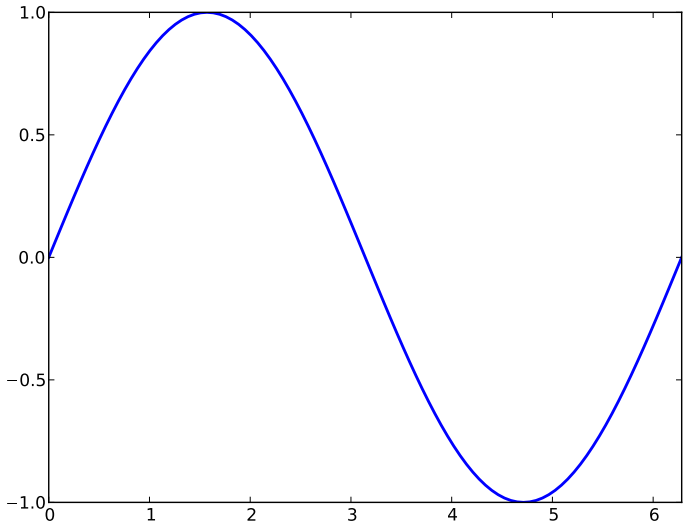


Figure 1.1: Illustration of how to include a figure.

{fig:sine}

Chapter 2

SVM Ensemble Learning from Positive and Unlabeled Data

{ch:resvm}

This chapter has been previously published as:
Claesen, M., De Smet, F., Suykens, J. A., & De Moor, B. (2015). **A robust ensemble approach to learn from positive and unlabeled data using SVM base models**. *Neurocomputing*, 160, 73-84.

Abstract

We present a novel approach to learn binary classifiers when only positive and unlabeled instances are available (PU learning). This problem is routinely cast as a supervised task with label noise in the negative set. We use an ensemble of SVM models trained on bootstrap resamples of the training data for increased robustness against label noise. The approach can be considered in a bagging framework which provides an intuitive explanation for its mechanics in a semi-supervised setting. We compared our method to state-of-the-art approaches in simulations using multiple public benchmark data sets. The included benchmark comprises three settings with increasing label noise: (i) fully supervised, (ii) PU learning and (iii) PU learning with false positives. Our approach shows a marginal improvement over existing methods in the second setting and a significant improvement in the third.

2.1 Introduction

Training binary classifiers on positive and unlabeled data is referred to as PU learning [86]. The absence of known negative training instances warrants appropriate learning methods. Inaccurate label information can be more problematic than attribute noise [138]. Specialised PU learning approaches are recommended when (i) negative labels cannot be acquired, (ii) the training data contains a large amount of false negatives or (iii) the positive set has many outliers.

Practical applications of PU learning typically feature large, imbalanced training sets with a small amount of labeled (positive) and a large amount of unlabeled training instances. The PU learning problem arises in various settings, including web page classification [136], intrusion detection [78] and bioinformatics tasks such as variant prioritization [115], gene prioritization [1, 95] and virtual screening of drug compounds [114].

Though these applications share a common underlying learning problem, the final evaluation criteria may be fundamentally different. For instance, in prioritization one wishes to obtain high precision since highly ranked targets may be subjected to further biological analysis. Intrusion detection, on the other hand, necessitates high recall to ensure that no anomalies go unnoticed.

Following **(author?)** [94], we will use the term *contamination* to refer to the fraction of mislabeled instances in a given set. We will denote the positive and unlabeled training instances by $\mathcal{P}$ and $\mathcal{U}$, respectively. Contamination in $\mathcal{P}$ refers to false positives while contamination in $\mathcal{U}$ refers to the presence of positives in $\mathcal{U}$. Usually $\mathcal{U}$ contains mostly true negative instances (e.g. contamination below 0.5) and $\mathcal{P}$ is assumed to be uncontaminated.

The distributions of the positive and a contaminated unlabeled set overlap even when those of the positive and underlying negative sets do not, which makes classification more difficult compared to a traditional supervised setting. **(author?)** [45] and **(author?)** [17] report statistical approaches to estimate the contamination of the unlabeled set and additionally show that distinguishing positives from unlabeled instances is a valid proxy for distinguishing positives from negatives.

The assumption in PU learning that $\mathcal{P}$ is uncontaminated may be violated in applications due to various reasons [51]. Additionally, outliers in the positive set may have a similar effect on classification performance [102]. We propose a novel PU learning method that is less vulnerable to potential contamination in $\mathcal{P}$ called the robust ensemble of support vector machines (RESVM). RESVM is compared to other methods in a series of simulations based on several public

data sets.

2.2 Related work

PU learning approaches can be split into two main conceptual categories: (i) approaches that account for the contamination of the unlabeled set explicitly by modeling the label noise and (ii) approaches that try to infer an uncontaminated (negative) subset $\hat{\mathcal{N}}$ from $\mathcal{U}$ and then train supervised algorithms to distinguish $\mathcal{P}$ from $\hat{\mathcal{N}}$. When *very* few labeled examples are available, the structure within the data is the main source of information which can be exploited by semi-supervised clustering techniques [3].

Accounting for the contamination of $\mathcal{U}$ in the modeling process This can be done by weighting individual data points, such as in weighted logistic regression [45, 80]. Another approach is by changing the penalties on misclassification during training, as is done in class-weighted SVM [86], bagging SVM [94] and RT-SVM [88].

Inferring an uncontaminated subset from $\mathcal{U}$ Another class of approaches tries to infer a negative set $\hat{\mathcal{N}}$ from $\mathcal{U}$. After the inferential step, binary classifiers are trained to distinguish $\mathcal{P}$ from $\hat{\mathcal{N}}$ in a supervised fashion. Examples of such two-step approaches include S-EM [87], mapping convergence (MC) [135] and ROC-SVM [81].

Class-weighted SVM and related approaches The approach we suggest belongs to the first class of methods and is closely related to class-weighted SVM and bagging SVM (which uses class-weighted SVM internally). We will discuss both of these approaches in more detail before moving on to the proposed method. We evaluated our method compared to both class-weighted SVM and bagging SVM.

2.2.1 Class-weighted SVM

{bsvm}

Class-weighted SVM (CWSVM) is a supervised technique in which the penalty for misclassification differs per class. **(author?)** [86] first applied class-weighted SVM for PU learning by considering the unlabeled set to be negative with noise on its labels. CWSVM is trained to distinguish $\mathcal{P}$ from $\mathcal{U}$. During training,

misclassification of positive instances is penalized more than misclassification of unlabeled instances to emphasize the higher degree of certainty on positive labels. In the context of PU learning, the optimization problem for training CWSVM can be written as:

{eq:bsvm}
$$\min_{\alpha,\xi,b} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \kappa(\mathbf{x}_i,\mathbf{x}_j) + C_{\mathcal{P}}\sum_{i\in\mathcal{P}}\xi_i + C_{\mathcal{U}}\sum_{i\in\mathcal{U}}\xi_i, \tag{2.1}$$

$$\text{s.t. } y_i(\sum_{j=1}^{N}\alpha_j y_j \kappa(\mathbf{x}_i,\mathbf{x}_j) + b) \geq 1-\xi_i, \qquad i = 1,\ldots,N,$$

$$\xi_i \geq 0, \qquad i = 1,\ldots,N,$$

with $\alpha \in \mathbb{R}^N$ the support values, $\mathbf{y} \in \{-1,+1\}^N$ the label vector, $\kappa(\cdot,\cdot)$ the kernel function, b the bias term and $\xi \in \mathbb{R}^N$ the slack variables. The misclassification penalties $C_{\mathcal{P}}$ and $C_{\mathcal{U}}$ require tuning (typically $C_{\mathcal{P}} > C_{\mathcal{U}}$ to emphasize known labels). SVM formulations with unequal penalties across classes have been used previously to tackle imbalanced data sets [98].

2.2.2 Bagging SVM

{baggingsvm}

Mordelet and Vert introduce bagging SVM as a meta-algorithm which consists of aggregating classifiers trained to discriminate $\mathcal{P}$ from a small, random resample of $\mathcal{U}$ [94]. They posit that PU learning problems have a particular structure that leads to instability of classifiers, namely the sensitivity of classifiers to the contamination of the unlabeled set. Bagging is a common technique used to improve the performance of instable classifiers [20].

In bagging SVM, random resamples of $\mathcal{U}$ are drawn and CWSVM classifiers are trained to discriminate $\mathcal{P}$ from each resample. By resampling $\mathcal{U}$, the contamination is varied. This induces variability in the classifiers which the aggregation procedure can then exploit. The size of the bootstrap resample of $\mathcal{U}$ is a tuning parameter in bagging SVM. The ratio $C_{\mathcal{P}}/C_{\mathcal{U}}$ is fixed so that the following holds:

{eq:bagpenalties}
$$|\mathcal{P}| \times C_{\mathcal{P}} = n_{\mathcal{U}} \times C_{\mathcal{U}}, \tag{2.2}$$

with $|\mathcal{P}|$ the size of the positive set and $n_{\mathcal{U}}$ the size of resamples from the unlabeled set. This choice of weights is common in imbalanced settings [24, 36]. All base models in bagging SVM classify the full set of positives against a subset of unlabeled instances and use a high misclassification penalty on the positives similar to CWSVM.

2.3 Robust Ensemble of SVMs

We propose a new technique called the robust ensemble of SVMs (RESVM). RESVM is a bagging method using CWSVM base models as discussed in Section 5.4.3. Base model training sets are constructed by bootstrap resampling both $\mathcal{P}$ and $\mathcal{U}$ separately, both of which may be contaminated.

The key difference between RESVM and bagging SVM is that the former resamples $\mathcal{P}$ in addition to $\mathcal{U}$ to increase variability between base models. RESVM additionally features an extra degree of freedom to control the relative misclassification penalty between positive and unlabeled instances, which is fixed in bagging SVM. **(author?)** [94] report no significant changes when varying the relative penalty in bagging SVM, though our experiments show that it is important in RESVM (see w_{pos} in Table 2.6).

Before elaborating on the details of RESVM, we briefly illustrate the effect of resampling contaminated sets. Subsequently we summarize the mechanisms of bagging and why they are advantageous when learning with label noise in the RESVM approach. Finally, we provide the full RESVM training approach and the way ensemble decision values are computed based on the base model decision values.

2.3.1 Bootstrap resampling contaminated sets

{resampling}

The RESVM approach resamples both $\mathcal{P}$ and $\mathcal{U}$, both of which are potentially contaminated. Resampling contaminated sets with replacement induces variability in contamination across the resampled sets (e.g. resamples of $\mathcal{U}$ and $\mathcal{P}$ that are used for training). The variability in contamination between resamples increases for increasing contamination of the original set. We assume contamination levels below 50%, e.g. less than half the instances in a given set are mislabeled. Due to the law of large numbers the contamination in bootstrap resamples of increasing size converges to the expected contamination, which equals that of the original set that is being resampled. As a result, the variability in contamination decreases for increasing resample size. Figure 2.1 illustrates this property empirically based on $20,000$ repeated measurements for each resample size: the expected value (mean) equals the original contamination, but the variability in resample contamination decreases for increasing resample size.

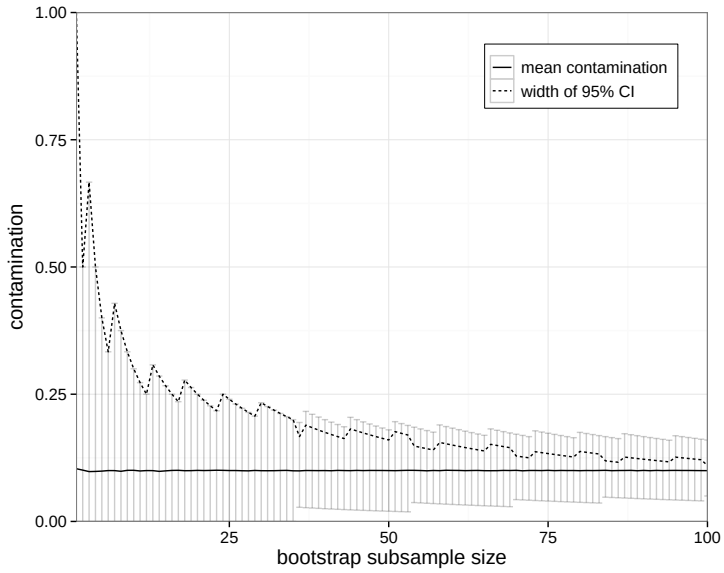


Figure 2.1: Contamination of bootstrap resamples for increasing size of resamples when the original sample has 10% contamination. Errorbars indicate the 95% confidence interval (CI) of contamination in resamples. The contamination varies greatly between small resamples as shown by the CIs. {fig:contamination}

2.3.2 Bagging predictors

(author?) [20] introduced bagging as a technique to construct strong ensembles by combining a set of base models. **(author?)** [21] stated that "the essential problem in combining classifiers is growing a suitably diverse ensemble of base classifiers" which can be done in various ways [23]. In bagging, the ensemble models use majority voting to aggregate decisions of base models which are trained on bootstrap resamples of the training set. From a Bayesian point of view, bagging can be interpreted as a Monte Carlo integration over an approximated posterior distribution [106].

In his landmark paper, **(author?)** [20] noted that base model instability is an important factor in the success of bagging which led to the use of inherently instable methods like decision trees in early bagging approaches [41, 22]. The main mechanism of bagging is often said to be variance reduction [9, 21]. In more recent work, **(author?)** [54] explained that base model instability is not related to the intrinsic variability of a predictor but rather to the presence of influential instances in a data set for a given predictor (so-called *leverage points*).

The effect of bagging is explained as equalizing the influence of all training instances, which is beneficial when highly influential instances are harmful for the predictor's accuracy.

2.3.3 Justification of the RESVM algorithm

We have shown the effect of resampling contaminated sets and provided some basic insight into the mechanics of bagging. We will now link these two elements to justify bagging approaches in the context of contaminated training sets. Its usefulness can be considered by both the variance reduction argument of **(author?)** [9] and equalizing the influence of training points as described by **(author?)** [54].

Variance reduction Resampling a contaminated set yields different levels of contamination in the resamples as explained in Section 2.3.1. Varying the contamination between base model training sets induces variability between base models without increasing bias. This observation enables us to create a diverse set of base models by resampling both $\mathcal{P}$ and $\mathcal{U}$. The variance reduction of bagging is an excellent mechanism to exploit the variability of base models based on resampling [9, 21]. In the context of RESVM, a tradeoff takes place between increased variability (by training on smaller resamples, see Figure 2.1) and base models with increased stability (larger training sets for the SVM models).

Equalizing influence The influence of a training instance on an SVM model can be quantified in terms of its dual weight (the associated α value). Three distinct cases can be distinguished: (i) the training instance is correctly classified and not within the margin ($\alpha = 0$, not a SV), (ii) the training instance lies on the margin and is correctly classified ($\alpha \in [0, C]$, free SV) and (iii) the training instance is incorrectly classified or within the margin ($\alpha = C$, bounded SV), where C is the misclassification penalty associated to the training instance [18]. Instances that are misclassified during training become bounded SVs, which have the maximal α value and can therefore be considered leverage points of the SVM model. When learning with label noise, the mislabeled training instances are likely to end up as bounded SVs. In a best case scenario, the mislabeled training instances are classified in concordance to their true label by the SVM model (which means they must be a bounded SV as the training procedure identifies this as a misclassification). As such, mislabeled training instances act as leverage points for SVM models. Following **(author?)** [54], bagging equalizes the influence of training instances (e.g. lowers the influence of

mislabeled leverage points in comparison to the rest of the data) which yields improved robustness against contamination in the context of RESVM.

2.3.4 RESVM training

RESVM uses CWSVM base models trained on resamples from the original training set, where both $\mathcal{P}$ and $\mathcal{U}$ are being resampled. The technique involves 5 hyperparameters: 3 to define the resampling strategy and 2 for the base models. Additional hyperparameters may be involved, for example γ for the RBF kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$.

The number of base models to include in the ensemble, $n_{nmodels}$, is the first hyperparameter. Using more base models improves the stability of the ensemble (up to a certain plateau) at a linear increase in computational cost for training and prediction. n_{models} is not a traditional hyperparameter in the sense that a good value can be determined during training, for example based on out-of-bag error estimates [8].[1]

By resampling $\mathcal{P}$, RESVM takes potential contamination of the labeled instances into account by design. Since the contamination between $\mathcal{P}$ and $\mathcal{U}$ can vary, the ability to vary the size of resamples from $\mathcal{P}$ and $\mathcal{U}$ separately is required. This results in two tuning parameters: n_{pos} and n_{unl}. In general, using small base model training sets results in increased base model variability which then necessitates using more base models in the ensemble to obtain a given level of stability. In our experiments, we have tuned n_{pos} and n_{unl} but it is also possible to obtain good values using out-of-bag techniques [90].[1]

RESVM additionally inherits at least 2 hyperparameters from its SVM base models, namely misclassification penalties for both classes and, if applicable, hyperparameters related to the kernel function. We define the CWSVM penalties in see Eq. (5.5) based on 2 hyperparameters $C_{\mathcal{U}}$ and w_{pos}:

{eq:resvmpenalties}

$$C_{\mathcal{P}} = C_{\mathcal{U}} \times w_{pos} \times \frac{n_{unl}}{n_{pos}}. \tag{2.3}$$

w_{pos} enables reweighting labeled and unlabeled instances after equalizing class imbalance. In bagging SVM, w_{pos} is always fixed to 1.

{labelednote}

[1] Note that the error estimates in out-of-bag techniques must account for potential contamination. See our discussion of hyperparameter tuning for a possible score function.

The RESVM training approach has been summarised in Algorithm 1. The algorithm uses 5 hyperparameters plus additional kernel parameters.

Algorithm 1: Training procedure for RESVM. {RESVM}

Data: $\mathcal{P}$: the set of positive instances.
$\mathcal{U}$: the set of unlabeled instances.
Input: n_{models}: number of base models to include in the ensemble.
n_{unl}: size of bootstrap resamples of $\mathcal{U}$.
n_{pos}: size of bootstrap resamples of $\mathcal{P}$.
$C_{\mathcal{U}}$: misclassification penalty for $\mathcal{U}$ in class-weighted SVM.
w_{pos}: relative positive misclassification penalty coefficient.
$\kappa(\cdot,\cdot)$: kernel function to be used by base models.
Output: Ω: RESVM with n_{models} base models.
begin
  $\Omega \leftarrow \emptyset$;
  $C_{\mathcal{P}} \leftarrow C_{\mathcal{U}} \times w_{pos} \times \frac{n_{unl}}{n_{pos}}$;
  for $i \leftarrow 1$ **to** n_{models} **do**
    // create base model training set from $\mathcal{P}$ and $\mathcal{U}$.
    $\mathcal{P}^{(i)} \leftarrow$ sample n_{pos} instances from $\mathcal{P}$ with replacement;
    $\mathcal{U}^{(i)} \leftarrow$ sample n_{unl} instances from $\mathcal{U}$ with replacement;
    // train CWSVM base model $\psi^{(i)}$ and add to ensemble Ω.
    $\psi^{(i)} \leftarrow$ train CWSVM for $\mathcal{P}^{(i)}$ vs. $\mathcal{U}^{(i)}$ (parameters $C_{\mathcal{P}}$, $C_{\mathcal{U}}$, κ);
    $\Omega \leftarrow \{\Omega, \psi^{(i)}\}$;

{RESVM}

2.3.5 RESVM prediction

{ensembledecvals}

RESVM uses majority voting to aggregate base model predictions. By default, the returned label is the one predicted by most base models. The fraction of positive votes for a test instance $\mathbf{x}$ can be written as:

$$v(\mathbf{x}) = \frac{n_{models} + \sum_{i=1}^{n_{models}} \mathtt{sgn}(\psi^{(i)}(\mathbf{x}))}{2n_{models}}, \tag{2.4}$$ {eq:majorityvote}

where $\mathtt{sgn}(\cdot)$ is the sign function and $\psi^{(i)}$ denotes the decision function of SVM base model i with codomain $\mathbb{R}$. $v(\cdot)$ has the interval $[0, 1]$ as codomain.

The RESVM decision value for a test instance $\mathbf{x}$ is defined as the fraction of votes in favor of the positive class $v(\mathbf{x})$ unless the result is unanimous. In the case of a unanimous vote, the ensemble decision value is based on the decision values of its base models to increase the model's ability to differentiate. In case

of a unanimous negative vote, the sum of the decision values of the base models is taken (each SVM base model decision value is negative in this case). In case of a unanimous positive vote, the sum of the decision values of the base models (all positive) plus one is taken. The decision value $d(\cdot)$ has codomain $\mathbb{R}$ and is computed as follows:

{eq:resvmdecval}
$$d(\mathbf{x}) = \begin{cases} v(\mathbf{x}) & \text{if } 0 < v(\mathbf{x}) < 1, \\ \sum_{i=1}^{n_{models}} \psi^{(i)}(\mathbf{x}) & \text{if } v(\mathbf{x}) = 0, \\ 1 + \sum_{i=1}^{n_{models}} \psi^{(i)}(\mathbf{x}) & \text{if } v(\mathbf{x}) = 1. \end{cases} \tag{2.5}$$

The resulting label for a given decision threshold T can be written as follows:

{eq:resvmlabel}
$$l(\mathbf{x}) = \mathtt{sgn}\big(d(\mathbf{x}) - T\big). \tag{2.6}$$

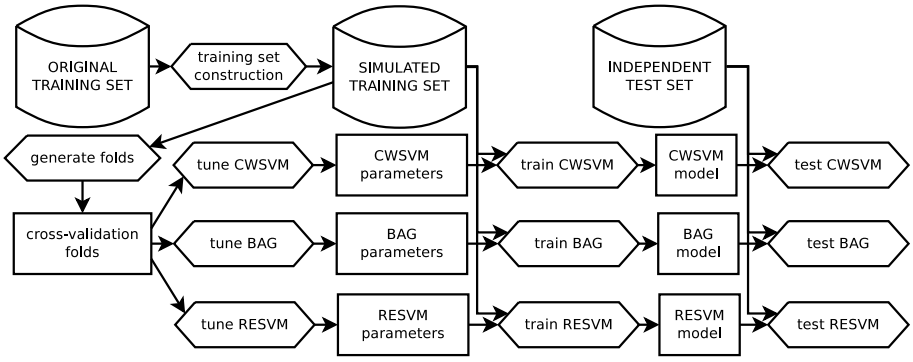
The default decision value threshold for positive classification is $T = 0.5$ (this is majority voting, e.g. positive iff more than half of all base models predict positive). Using the modified decision values $d(\mathbf{x})$ instead of the votes $v(\mathbf{x})$ does not affect the predicted labels for typical choices of the threshold T (e.g. $T \in (0, 1)$). It does, however, affect performance measures that use the entire range of decision values such as area under the PR curve. Using $d(\mathbf{x})$ enables us to rank different instances that received all positive or all negative votes by base models (e.g. $v(\mathbf{x}) = 1$ and $v(\mathbf{x}) = 0$, respectively).

2.4 Experimental setup

RESVM has been compared to class-weighted SVM (CWSVM) and bagging SVM (BAG) in a number of simulations to assess the merits of our modifications compared to conceptually comparable algorithms. In this Section we will summarize the experimental setup (training set construction, model selection and performance evaluation) and the data sets we used.

2.4.1 Simulation setup

Our experiments consist of repeated simulations on a variety of data sets under different settings. Briefly, in each iteration hyperparameters were optimized per approach based on cross-validation on the training set (using identical folds for all approaches). Subsequently, a model with the optimal parameters is trained on the full training set and used to predict an independent test set. An overview of the experiments is shown in Figure 2.2. Every experiment consists of 20 repetitions.



{fig:benchmark}

Figure 2.2: Overview of a single benchmark iteration.

To assess what situations are favorable per approach, we have investigated three different settings with distinct label noise configurations. For every data set, we performed 10 iterations per simulation in the following settings:

1. **supervised**: no contamination in $\mathcal{P}$ or $\mathcal{U}$ ($\mathcal{U}$ is the negative class).
2. **PU learning**: contamination in $\mathcal{U}$ but not in $\mathcal{P}$.
3. **semi-supervised**: contamination in both $\mathcal{P}$ and $\mathcal{U}$. The contamination levels in $\mathcal{P}$ and $\mathcal{U}$ were always chosen equal.

The contamination levels we used were chosen per data set based on when differences between the three approaches become visible. A summary is available in Table 2.1 in Section 5.4.2. When applicable, contamination was introduced by flipping class labels (e.g. true positives in $\mathcal{U}$ and true negatives in $\mathcal{P}$). This effectively changes the empirical densities of the classes in the training set (illustrated in Figure 2.3 in the next Section).

Every binary learning task was repeated 20 times to get reliable assessments of all methods. Each repetition involves redoing all steps shown in Figure 2.2, including resampling of training sets based on the known true positives and true negatives. Contamination was introduced at random where applicable by flipping class labels.

Hyperparameter selection In every iteration, hyperparameters were tuned per setting using 10-fold cross-validation over a grid of parameter tuples. To ensure a fair comparison, one set of folds is generated in each iteration and used by all methods. We ensured that the optimal values that were found during

tuning in any setting were never on the edge of the search grid. The search resolution in comparable parameters between methods was always defined to be identical (for example γ in the case of an RBF kernel).

The same search grids were used in all three settings for a given data set to illustrate that a method can work well in a supervised setting with a given search grid but degrade when label noise is added. Since negative labels are unavailable in PU learning, we used the following score function in all learning settings which only requires positive labels for hyperparameter selection [80]:

{puscore}
$$\texttt{pu\_score} = \frac{\texttt{precision} \times \texttt{recall}}{Pr(y=1)} = \frac{\texttt{recall}^2}{Pr(\hat{y}=1)}, \tag{2.7}$$

where $Pr(y=1)$ is the fraction of known positive labels in the predicted set and $Pr(\hat{y}=1)$ is the fraction of positive predictions made by the classifier. Note that this score function is not ideal when $\mathcal{P}$ is contaminated, though we obtained good results even in that setting.

The following parameters were tuned per method: (CWSVM) $C_{\mathcal{P}}$ and $C_{\mathcal{U}}$, (BAG) $C_{\mathcal{U}}$ and $n_{\mathcal{U}}$ and (RESVM) $C_{\mathcal{U}}$, w_{pos}, n_{pos} and n_{unl}. In both ensemble approaches we consistently used 50 base models.

Performance assessment Models are trained with the optimal hyperparameters on the full training set and subsequently tested on the independent test set. We use the known test labels to compute the area under the Precision-Recall curve (AUC) for each model. We opted to use PR curves because they capture the performance of interest of models over their entire operating range and work well for imbalanced data [37].

We used statistical analyses to determine whether one approach trumps another while accounting for the variability between simulations. The nonparametric Wilcoxon signed-rank test is recommended for pairwise comparisons between learning algorithms [39]. In every setting per data set we performed a paired one-tailed Wilcoxon signed-rank test comparing the area under the PR curve of bagging SVM and RESVM with alternative hypothesis h_1 : $AUC^{RESVM} > AUC^{BAG}$ (pairs being iterations). Low p-values indicate a statistically significant improvement.

Implementation details We used the class-weighted SVM implementation available in LIBLINEAR [48] and LIBSVM [25] for models using the linear and RBF kernel, respectively. Bagging SVM and RESVM were implemented using the EnsembleSVM library [30].[2] The decision values of bagging SVM

[2]Python code for RESVM is available at `https://github.com/claesenm/resvm`.

used to compute PR curves were defined in the same way as for RESVM (see Section 2.3.5).

2.4.2 Data sets

{data}

We used a synthetic data set and 5 publicly available data sets:[3]

- `synthetic`: a 2-D binary data set. Positive instances are sampled from a standard normal distribution. Negative instances are sampled from a circle centered at the origin with radius 4 with 2-D noise superimposed from a standard normal distribution. Training and testing data was generated in every iteration. Figure 2.3 shows densities for all settings.
- `cancer`: the Wisconsin breast cancer data set related to breast cancer diagnosis. It consists of 10 features and 683 instances without an explicit train/test partitioning so we partitioned it at random in every iteration.
- `ijcnn1`: used for the IJCNN 2001 neural network competition [104], comprising 2 classes, 22 features and $49,990/91,701$ training/testing instances.
- `covtype`: a common classification benchmark about predicting forest cover types based on cartographic information [16]. We used a subsample of $100,000/40,000$ training/testing instances.
- `mnist`: a digit recognition task [79]. This data set contains 10 classes (one for each digit), 780 features, $60,000$ training instances and $10,000$ test instances with an almost uniform class distribution. We performed one-versus-all classification for each digit.
- `sensit`: SensIT Vehicle (combined), vehicle classification [42]. This data set contains 3 classes with an uneven distribution. We performed one-versus-all classification for each class. This data set has 100 features, $78,823$ training instances and $19,705$ testing instances.

Most data sets have a prespecified test set, except for `synthetic` and `cancer`. We used the prespecified test sets when available. We used the RBF kernel for all data sets except `mnist` (linear kernel). Note that both RESVM and bagging SVM models are always implicitly nonlinear due to their majority voting scheme, even when using linear base models.

[3]Public data at: `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

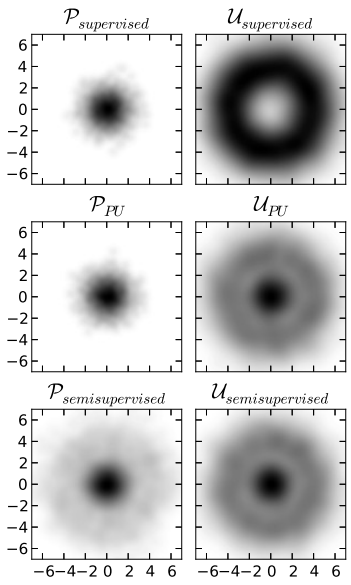


Figure 2.3: Empirical densities of the `synthetic` data used for training per problem setting (visualized in input space). The supervised densities (top row) are based on samples of the underlying positive and negative classes. The use of high contamination (30%) induces similar empirical densities for $\mathcal{P}$ and $\mathcal{U}$ in the semi-supervised setting (bottom row). {fig:densities}

In every setting each original data set was resampled without replacement to construct training sets to use in the simulations. The resampled training sets are typically significantly smaller than what is available in the original data sets to show that some methods can obtain good models even with few training instances. An overview of the actual training sets we constructed is presented in Table 2.1.

data set	d	contamination in percent	training set $\lvert\mathcal{P}\rvert$	training set $\lvert\mathcal{U}\rvert$	test set $\lvert\mathcal{P}\rvert$	test set $\lvert\mathcal{N}\rvert$
`synthetic`	2	30	100	200	$5,000$	$5,000$
`cancer`	10	30	50	200	100	100
`ijcnn1`	22	10	100	$10,000$	$8,712$	$82,989$
`covtype`	54	30	100	1,000	20,000	20,000
`mnist`	780	10	50	$2,000$	$\approx 1,000$	$\approx 9,000$
`sensit 1`	100	30	100	$1,000$	$4,575$	$15,130$
`sensit 2`	100	30	100	$1,000$	$5,520$	$14,455$
`sensit 3`	100	30	100	$1,000$	$9,880$	$9,825$

Table 2.1: Overview of the data sets used in simulations: number of features, contamination (when applicable), training set size as used in the experiments and test set size. The `mnist` data set consists of 10 classes and the test set is almost uniformly distributed. The `sensit` data set has 3 classes with uneven class distribution in the test set, so we treat it separately here.

{table:datasets}

2.5 Results and discussion

We will summarize all results of our simulation experiments comparing class-weighted SVM (CWSVM), bagging SVM (BAG) and the robust ensemble of SVMs (RESVM). First we will show the results of each setting separately. Subsequently we present an overview of the number of wins per setting for each method across all data sets. Section 2.5.6 shows the results of an experiment to assess the effect of contamination in $\mathcal{P}$ and $\mathcal{U}$ on all methods. Finally, we include an interesting observation regarding the optimal hyperparameters of RESVM that were found using cross-validation on the `mnist` data set per setting in Table 2.6.

2.5.1 Results for supervised classification

Table 2.2 summarizes our results in a fully supervised setting. In these experiments both $\mathcal{P}$ and $\mathcal{U}$ are uncontaminated. Based on the number of wins per simulation and the confidence intervals, we can conclude that all methods are competitive in this setting.

The confidence intervals show that all methods obtain comparable results for all simulations except `mnist` digit 8, where CWSVM performs poorly compared to the others. This performance difference could be caused by the fact we used linear class-weighted SVM while both ensemble methods implicitly yield

nonlinear decision boundaries. A linear model may be too simple to properly distinguish this digit from the others.

The overall good results in the supervised setting confirm that the score function in Equation (2.7) is a good choice for tuning. In these supervised experiments we could have used a traditional score like accuracy, area under the ROC curve or F-measure, but these would no longer be useful in the other settings. The performance in these supervised experiments can be considered an objective baseline for comparison in the PU learning and semi-supervised setting since only levels of contamination are varied.

data	area under PR curve			p	number of wins		
	CWSVM	BAG	RESVM		CWSVM	BAG	RESVM
synthetic	98.1–98.7	98.7–98.8	98.7–98.8		2	12	6
cancer	98.4–98.8	98.4–98.7	98.3–98.7		8	12	0
ijcnn1	85.3–87.4	79.1–81.6	82.3–86.2	• • •	16	0	4
covtype	77.1–78.3	76.8–78.5	76.8–78.7		8	6	6
mnist (positive = x)							
0	96.9–97.5	96.9–97.4	96.9–97.4		7	8	5
1	98.1–98.3	98.3–98.5	98.2–98.5		0	8	12
2	87.3–89.1	88.5–89.8	89.6–90.5	•	2	6	12
3	83.7–85.9	86.9–88.7	88.8–90.1	• • •	0	5	15
4	88.8–90.2	89.8–91.1	90.8–92.2	• • •	1	3	16
5	78.7–80.9	79.2–81.0	81.4–83.2	• •	3	3	14
6	92.4–93.4	93.9–94.7	94.3–94.9		0	8	12
7	92.2–92.9	92.6–93.2	93.1–93.7	• • •	1	3	16
8	56.5–58.9	74.3–76.1	79.6–80.5	• • •	0	0	20
9	72.5–75.6	77.8–80.3	81.5–82.6	• • •	0	2	18
sensit (positive = x)							
1	80.5–81.4	79.8–80.7	80.5–81.3	•	10	2	8
2	65.7–75.4	72.6–74.0	73.5–74.9	• • •	15	0	5
3	35.5–56.1	92.3–92.7	91.7–92.3		0	15	5

Table 2.2: 95% CIs for mean test set performance in a fully supervised setup, the results of a paired one-tailed Wilcoxon signed-rank test comparing the AUC of BAG and RESVM with alternative hypothesis $h_1 : AUC^{RESVM} > AUC^{BAG}$ and the number of times each approach had best test set performance. Test result encoding: • $p < 0.05$, • • $p < 0.01$ and • • • $p < 0.001$.

{table:supervised}

2.5.2 Results for PU learning

The results of our experiments in a PU learning setting are shown in Table 2.3. In the pure PU learning setting, $\mathcal{P}$ is uncontaminated but $\mathcal{U}$ is contaminated. Class-weighted SVM tends to suffer from the largest loss in performance between supervised learning and pure PU learning based on area under PR curves. Class-weighted SVM obtains less wins than it did in the supervised simulations (21 wins in PU learning compared to 73 in the supervised setting), except on the `cancer` data set. Bagging SVM and RESVM maintain strong performance. Bagging SVM obtains a comparable number of wins and RESVM gains many compared to the supervised setting.

On the `mnist` data, RESVM consistently exhibits the best performance (based on the Wilcoxon signed-rank test), though the effective improvement over bagging SVM is marginal. On `sensit` with classes 2 or 3 as positive, bagging SVM obtains the majority of wins though the confidence intervals of its area under the PR curve overlap completely with those of RESVM. On the other data sets, no worthwile differences were obtained between both ensemble methods.

2.5.3 Results of semi-supervised classification

In the semi-supervised setting we deliberately violated the assumption of an uncontaminated positive training set by contaminating $\mathcal{P}$ and $\mathcal{U}$. The results listed in Table 2.4 confirm that both class-weighted and bagging SVM are vulnerable to contamination in $\mathcal{P}$ and experience very large performance losses. We believe this is induced by using high misclassification penalties for training instances in $\mathcal{P}$ without any resampling to account for potential false positives. In bagging SVM this leads to a systematic bias in all base models. The resampling strategy of RESVM prevents systematic bias over all base models.

The results clearly show that RESVM is more robust to false positives, evidenced by a much lower drop in predictive performance for almost all data sets. The performance difference between bagging SVM and RESVM is statistically significant for all data sets except `covtype` and `sensit`. Surprisingly, CWSVM obtains 8 wins on `sensit` with class 2 as positive. RESVM shows the best and most consistent performance overall.

On the `mnist` data, RESVM not only achieved consistently higher area under the PR curve, but visual inspection showed that its PR curves almost always dominated the others over the entire range. This means that in this experiment, RESVM models are always better than the others regardless of design priorities (high precision versus high recall). As an illustration, Figure 2.4 shows the PR

data	area under PR curve			p	number of wins		
	CWSVM	BAG	RESVM		CWSVM	BAG	RESVM
synthetic	96.9–98.4	97.9–98.6	98.2–98.5		6	8	6
cancer	98.2–98.5	87.5–98.4	96.1–98.1		10	7	3
ijcnn1	71.2–76.5	73.4–78.2	72.6–80.7	•	1	5	14
covtype	65.2–67.9	70.2–72.2	71.4–73.0		0	6	14
mnist (positive = x)							
0	74.1–77.8	90.5–93.3	94.6–95.5	• • •	0	5	15
1	89.1–91.2	95.2–96.7	96.4–97.3	• •	0	5	15
2	55.2–60.1	75.5–80.0	84.2–86.1	• • •	0	0	20
3	54.6–60.2	74.5–80.3	83.6–86.2	• • •	0	2	18
4	57.8–62.5	73.9–80.3	83.9–85.9	• • •	0	2	18
5	53.3–56.7	63.8–70.3	69.1–72.6	•	0	7	13
6	66.9–71.0	85.9–89.7	90.6–92.5	• •	0	4	16
7	71.4–74.8	84.0–88.0	90.0–91.4	• • •	0	1	19
8	34.8–38.8	63.5–69.1	72.2–74.8	• • •	0	4	16
9	50.5–54.8	66.2–71.0	74.2–76.4	• • •	0	1	19
sensit (positive = x)							
1	61.6–73.0	70.6–75.3	72.5–76.2	•	2	7	11
2	58.6–68.1	68.5–70.5	67.8–70.0		2	10	8
3	33.2–50.2	90.2–91.8	89.7–91.1		0	14	6

Table 2.3: 95% CIs for mean test set performance in a PU learning setup, the results of a paired one-tailed Wilcoxon signed-rank test comparing the AUC of BAG and RESVM with alternative hypothesis $h_1 : AUC^{RESVM} > AUC^{BAG}$ and the number of times each approach had best test set performance. Test result encoding: • $p < 0.05$, • • $p < 0.01$ and • • • $p < 0.001$. {table:pulearning}

and ROC curves of a representative simulation with digit 7 as positive. Since the PR curve of RESVM completely dominates the others we know that its ROC curve does too [37].

Finally, it is worth noting that the confidence intervals of RESVM tend to be narrower than those of both other approaches. Even though RESVM base models have more variability compared to bagging SVM base models, the overall performance of RESVM is more reliable. This constitutes an important practical advantage since assessing different models is not trivial outside of simulation studies (e.g. when no negative labels are available).

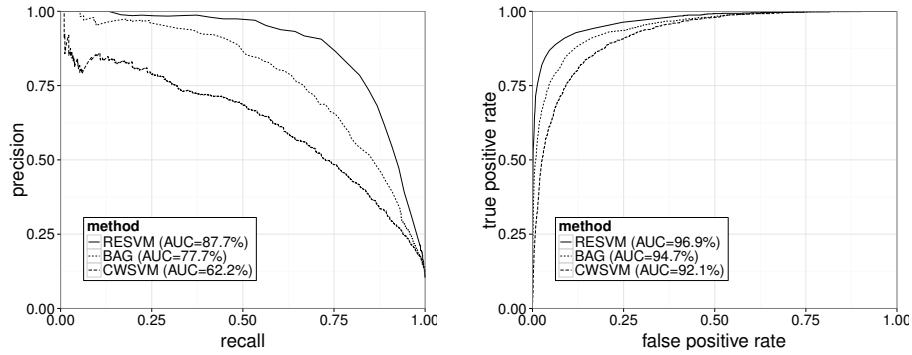
data	area under PR curve CWSVM	BAG	RESVM	p	number of wins CWSVM	BAG	RESVM
synthetic	83.6–90.0	91.9–94.9	96.4–97.4	• • •	3	2	15
cancer	62.5–80.2	91.1–96.7	96.2–97.6	•	1	8	11
ijcnn1	69.8–73.4	67.4–70.4	72.0–75.2	• • •	5	2	13
covtype	58.1–61.8	61.2–64.2	60.4–65.7		4	4	12
mnist (positive = x)							
0	59.9–64.1	72.8–81.1	91.4–93.4	• • •	0	0	20
1	80.3–82.7	90.6–93.4	96.1–97.4	• • •	0	0	20
2	42.3–48.0	55.1–63.7	79.8–83.0	• • •	0	0	20
3	43.8–47.6	59.9–66.0	78.1–81.1	• • •	0	0	20
4	52.4–56.2	66.4–72.8	79.7–83.4	• • •	0	0	20
5	40.5–45.2	56.0–61.1	65.8–69.4	• • •	0	2	18
6	52.4–57.3	72.9–79.3	87.9–90.9	• • •	0	0	20
7	58.7–61.6	69.9–77.3	87.9–90.2	• • •	0	1	19
8	29.7–33.9	48.3–55.3	68.0–71.0	• • •	0	0	20
9	42.1–44.9	52.5–59.0	68.7–72.7	• • •	0	0	20
sensit (positive = x)							
1	34.5–49.4	59.6–69.0	60.6–66.4		3	12	5
2	44.9–53.7	46.4–53.4	50.1–56.7	•	8	4	8
3	44.5–61.1	75.4–83.5	80.5–84.9	•	1	7	12

Table 2.4: 95% CIs for mean test set performance in a semi-supervised setup, the results of a paired one-tailed Wilcoxon signed-rank test comparing the AUC of BAG and RESVM with alternative hypothesis $h_1 : AUC^{RESVM} > AUC^{BAG}$ and the number of times each approach had best test set performance. Test result encoding: • $p < 0.05$, • • $p < 0.01$ and • • • $p < 0.001$. {table:semisupervised}

2.5.4 A note on the number of repetitions per experiment

The tightness of the confidence intervals of generalization performance allow us to conclude that the number of repetitions (20) is sufficient to demonstrate the merits of RESVM (see Tables 2.2–2.4). Increasing the number of repetitions further would yield even narrower confidence intervals and increase the amount of statistically significant results in the Wilcoxon signed-rank test comparing bagging SVM and RESVM (due to increased power). All key conclusions remain valid if the number of repetitions would be increased.

Additional statistically significant results may only be obtained in experiments where the improvement offered by RESVM is too small to be of practical significance (as large improvements already yield significant test results). Failure



{fig:pr} (a) Precision-Recall curves. {fig:roc} (b) ROC curves.

{fig:curves} Figure 2.4: Performance in semi-supervised setting on `mnist`, digit 7 as positive.

to reject the null hypothesis ($h_0 : AUC^{BAG} \geq AUC^{RESVM}$) in our current results indicates that (i) bagging SVM is effectively better than RESVM, (ii) they are comparable or (iii) the performance improvement of RESVM is too small to yield a significant test result given the current sample size (number of repetitions). Increasing the number of repetitions can only lead to additional statistically significant results in the latter situation.

To illustrate our claims, we performed 100 repetitions for `covtype` in the semi-supervised setting. This yielded the following CIs and win counts: CWSVM 59.0–60.5% (8 wins), bagging SVM 62.3–63.5% (21 wins), RESVM 63.8–65.8% (71 wins). The p-value of the Wilcoxon signed-rank test becomes 2×10^{-5}, while the p-value was insignificant with 20 repetitions (Table 2.4).

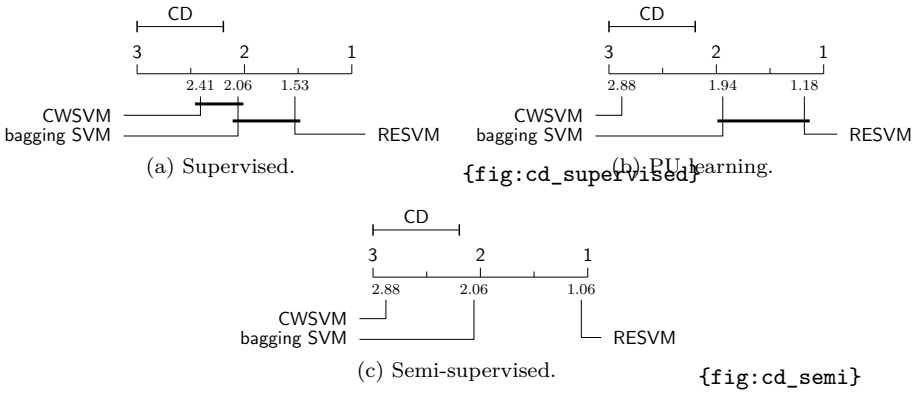
2.5.5 Trend across data sets

In the previous tables we have shown the results per data set for each setting. In this section we summarize the results across all data sets, using critical difference diagrams [39] in Section 2.5.5 and an overview of win counts in Section 2.5.5.

Critical difference diagrams

{cds}

In every setting, we compared the performance of the three learning approaches across all data sets using non-parametric statistical tests. For each data set, approaches were ranked based on their mean area under the PR curve across all



(a) Supervised.

(b) PU learning.

{fig:cd_supervised} {fig:cd_pu}

(c) Semi-supervised.

{fig:cd_semi}

Figure 2.5: Critical difference diagrams for each setting. Groups of algorithms that are not significantly different at the 5% significance level are connected. {fig:cds}

iterations. Multiclass data sets count once per class. Friedman tests per setting yielded significant evidence of differences between the three learning approaches at the $\alpha = 0.05$ level, though this was marginal in the supervised setting ($p = 0.034$). The Nemenyi post-hoc test [97] was used after each omnibus test to assess differences between all approaches. The critical difference diagrams in Figure 2.5 visualize the results.

Critical difference diagrams were introduced by **(author?)** [39] to visualize a comparison of multiple learning approaches over multiple data sets. These diagrams depict the average rank of each approach (lower is better) along with the critical difference (CD). The critical difference is the minimum difference in average ranks that yields a significant result in the Nemenyi post-hoc test. It depends on the significance level ($\alpha = 0.05$), the number of learning approaches (3) and the number of data sets (17).

From Figure 2.5 we can conclude that bagging SVM and RESVM are comparable in the PU learning setting (both significantly better than CWSVM). In the semi-supervised setting, bagging SVM is statistically significantly better than CWSVM and RESVM is significantly better than both other approaches across all data sets.

Win counts

{wins}

The number of wins per method across all data sets are summarized in Table 2.5. The top half shows the total number of wins across all data sets, which weights

`mnist` and `sensit` heavier than the other data sets since we performed several one-vs-all experiments. Because RESVM consistently performed very strong on `mnist`, the top half is an overly optimistic representation.

The bottom half of Table 2.5 contains normalized results, where every data set contributes equally. Based on these numbers we can conclude that there is little difference between the three methods in a supervised setting. In the PU learning setting, ensemble methods become favorable over CWSVM (bagging SVM and RESVM being competitive). Finally, in the semi-supervised setting RESVM pulls far ahead of both other methods and obtains 65% of the normalized wins, which is over three times more than bagging SVM and over five times more than class-weighted SVM.

	CWSVM		bagging SVM		RESVM	
setting	count	win %	count	win %	count	win %
supervised	73	21	93	27	**174**	**51**
PU learning	21	6	88	26	**231**	**68**
semi-supervised	25	7	42	12	**273**	**80**
supervised	**44.8**	**37.3**	40.3	33.6	36.0	30.0
PU learning	18.3	15.3	39.4	32.8	**62.2**	**51.8**
semi-supervised	17.0	14.2	24.0	20.0	**79.0**	**65.8**

Table 2.5: Number of wins in simulations for each method per setting. The bottom half shows normalized number of wins, where wins in multiclass data sets (`mnist` and `sensit`) are divided by the number of classes.

{table:wins}

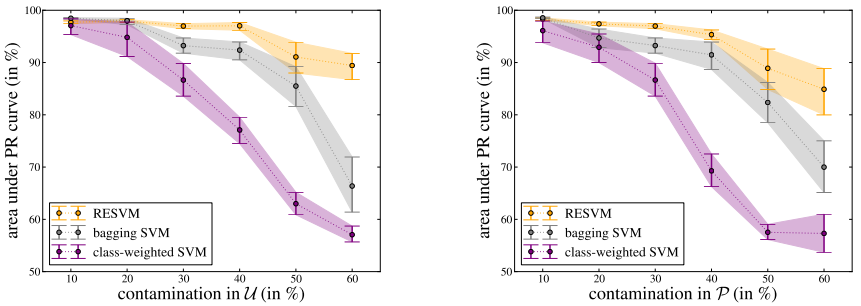
2.5.6 Effect of contamination

{varycontamination}

In this Section we show the effect of different levels of contamination in $\mathcal{P}$ and $\mathcal{U}$ on the `synthetic` data set. In these simulations, we fixed the contamination level in one part of the training set ($\mathcal{P}$ or $\mathcal{U}$) and the contamination of other was varied. The fixed contamination was set to 30%. Twenty simulations were run per contamination setting.

In these experiments, we used random search to tune hyperparameters of each method [10] using the Optunity package.[4] Briefly, hyperparameters were searched by random sampling 100 tuples uniformly within a given box and subsequently the best tuple was selected as before. We ensured that the optimal hyperparameters were never too close to the edge of the feasible region (if so,

[4]Optunity is available at: `http://www.optunity.net`.



(a) Effect of contamination in $\mathcal{U}$.

{fig:contamination0}

(b) Effect of contamination in $\mathcal{P}$.

{fig:contaminationP}

Figure 2.6: Effect of different levels of contamination in $\mathcal{U}$ and $\mathcal{P}$ on generalization performance. The plots show point estimates of the mean area under the PR curve across experiments and the associated 95% confidence intervals.

{fig:contaminationeffect}

the box was expanded). Note that this approach of testing a fixed number of tuples favors methods with less hyperparameters. Even though RESVM has more hyperparameters than the other methods, good models can be obtained at the same search cost.

The results are shown in Figure 2.6. In general, contamination in $\mathcal{P}$ causes larger performance losses than the same level of contamination in $\mathcal{U}$ for all algorithms. As expected, the difference in sensitivity to contamination in $\mathcal{P}$ and $\mathcal{U}$ is smallest for RESVM in which $\mathcal{P}$ and $\mathcal{U}$ are resampled similarly. At high contamination levels, RESVM is the only method that still works well (even at 60%).

Figure 2.6a illustrates that RESVM and bagging SVM behave in a similar fashion at contamination levels of $\mathcal{U}$ up to 50% and both outperform class-weighted SVM. RESVM outperforms bagging SVM for contamination levels of 30–50% but the consistency (width of CI) and performance losses of both methods are comparable. Figure 2.6b shows the increased robustness of RESVM to contamination in $\mathcal{P}$ resulting in reduced loss of generalization performance for increasing contamination.

2.5.7 RESVM optimal parameters

As an illustration of the implicit mechanism of RESVM we show some of the optimal tuning parameters for every setting in Table 2.6. These parameters were obtained by performing 10-fold cross-validation on the training set.

	0	1	2	3	4	5	6	7	8	9	mean
$\mathbf{n_{pos}}$											
supervised	20	20	20	20	20	20	10	20	20	10	18
PU learn	10	10	10	10	10	15	10	10	10	10	10.5
semi-sup.	10	5	10	10	10	10	10	10	10	10	9.5
$\mathbf{n_{unl}/n_{pos}}$											
supervised	10	10	10	10	10	10	10	10	10	10	10
PU learn	5	5	5	5	5	5	5	5	5	5	5
semi-sup.	5	5	5	8	5	5	5	5	5	5	5.25
$\mathbf{w_{pos}}$											
supervised	1.6	1.6	1.6	3.2	3.2	3.2	3.2	1.6	3.2	2.4	2.48
PU learn	4.8	6.4	3.2	6.4	4.8	6.4	4.8	4.8	6.4	6.4	5.44
semi-sup.	12.8	6.4	4.8	2.1	4.8	6.4	4.8	3.2	3.2	3.2	5.17

Table 2.6: Medians of optimal hyperparameters per digit obtained via cross-validation and mean of all medians per setting. The normalized relative weight on positives versus unlabeled instances (w_{pos}) is associated with the relative size and contamination of the positive and unlabeled training sets.

{table:hyperparameters}

An interesting observation is that the size of the training sets that are being used decreases for increasing contamination. Increasing label noise induces RESVM to favor smaller base model training sets for which the variability in contamination is larger (see Figure 2.1). Though this may appear counterintuitive, bagging approaches are known to exhibit a bias-variance tradeoff [9] for which using weaker base models with increased variability may yield better ensembles [73].

The optimal value of the misclassification penalty for positive training instances relative to unlabeled instances, w_{pos}, changes between learning settings (see Equation (2.3)). It exhibits expected behaviour: the maximum value is obtained when the certainty on $\mathcal{P}$ relative to $\mathcal{U}$ is largest (e.g. the pure PU learning setting). This parameter implicitly balances empirical certainty on $\mathcal{P}$ and $\mathcal{U}$ and is an important degree of freedom in RESVM. In bagging SVM, this parameter is implicitly fixed to 1 via Equation (5.6) [94]. Note that w_{pos} need not be larger than 1 (which would place extra emphasisis on the known labels after accounting for class imbalance). In highly imbalanced settings where $n_{unl} \gg n_{pos}$, the optimal value of w_{pos} may well be less than 1.

2.6 Conclusion

We have introduced a new approach for learning from positive and unlabeled data, called the robust ensemble of SVMs (RESVM). RESVM constructs an ensemble model using a bagging strategy in which the positive and unlabeled sets are resampled to obtain base model training sets. By resampling both $\mathcal{P}$ and $\mathcal{U}$, our approach is more robust against false positives than others.

The robustness of our approach to potential contamination in both $\mathcal{P}$ and $\mathcal{U}$ can be attributed to the synergy between our resampling scheme and voting aggregation. The resampling itself strongly resembles a typical bootstrap approach. RESVM uses class-weighted SVM base models though the resampling scheme is likely to work well with other types of base models.

RESVM was compared with class-weighted SVM and bagging SVM on several data sets under different label noise conditions. The trends across data sets show that bagging SVM and RESVM outperform class-weighted SVM in PU learning. In a pure PU learning setting the average improvement over existing methods is modest though RESVM classifiers exhibit lower variance in performance making it more reliable.

In the semi-supervised setting, label noise was introduced in $\mathcal{P}$ to highlight the improved robustness of RESVM compared to the other methods. Our experimental results show that RESVM remains very strong in the semi-supervised setting while both other approaches degrade dramatically. Statistical analysis showed that RESVM is significantly better than both other approaches across all data sets.

Visual inspection of the PR curves shows that in the majority of experiments the curve for RESVM not only has higher AUC but completely dominates the other curves. As such RESVM models are a good approach regardless of design priorities (high recall versus high precision).

A weakness of RESVM is its amount of hyperparameters (5 plus potential kernel parameters), though RESVM models are less sensitive to accurate tuning of these parameters than standard SVM. Our experiments indicated that although RESVM has more hyperparameters, good models can be obtained at the same search effort than the other approaches (e.g. testing the same number of hyperparameter tuples). An interesting question is whether prior knowledge regarding contamination of $\mathcal{P}$ and $\mathcal{U}$ can help in limiting the search scope for some of the hyperparameters (n_{pos}, n_{unl} and w_{pos} specifically).

Chapter 3

EnsembleSVM: A Library for Ensemble Learning Using Support Vector Machines

{ch:ensemblesvm}

This chapter has been previously published as:
Claesen, M., De Smet, F., Suykens, J. A. K., & De Moor, B. (2014). **EnsembleSVM: A library for ensemble learning using support vector machines**. *Journal of Machine Learning Research*, 15(1), 141–145.

Abstract

`EnsembleSVM` is a free software package containing efficient routines to perform ensemble learning with support vector machine (SVM) base models. It currently offers ensemble methods based on binary SVM models. Our implementation avoids duplicate storage and evaluation of support vectors which are shared between constituent models. Experimental results show that using ensemble approaches can drastically reduce training complexity while maintaining high predictive accuracy. The `EnsembleSVM` software package is freely available online at `http://esat.kuleuven.be/sista/ensemblesvm`.

3.1 Introduction

Data sets are becoming increasingly large. Machine learning practitioners are confronted with problems where the main computational constraint is the amount of time available. Problems become particularly challenging when the training sets no longer fit into memory. Accurately solving the dual problem for SVM training with nonlinear kernels requires a run time which is at least quadratic in the size of the training set n, thus training complexity is $\Omega(n^2)$ [19, 85].

`EnsembleSVM` employs a divide-and-conquer strategy by aggregating many SVM models, trained on small subsamples of the training set. Through subdivision, total training time decreases significantly, even though more models need to be trained. For example, training p classifiers on subsamples of size n/p, results in an approximate complexity of $\Omega(n^2/p)$. This reduction in complexity helps in dealing with large data sets and nonlinear kernels.

Ensembles of SVM models have been used in various applications [129, 84, 96]. Collobert et al. [33] use ensembles for large scale learning and employ a neural network to aggregate base models. Valentini and Dietterich [123] provide an implementation which allows base models to use different kernels. For efficiency reasons, we require base models to share a single kernel function.

While other implementations mainly focus on improving predictive performance, our framework primarily aims to (i) make nonlinear large-scale learning feasible through complexity reductions and (ii) enable fast prototyping of novel ensemble algorithms.

3.2 Software Description

The `EnsembleSVM` software is freely available online under a LGPL license. `EnsembleSVM` provides ensembles of instance-weighted SVMs, as defined in Equation (3.1). The default approach we offer is bagging, which is commonly used to improve the performance of unstable classifiers [20]. In bagging, base models are trained on bootstrap subsamples of the training set and their predictions are aggregated through majority voting.

Base model flexibility is maximized by using instance-weighted binary support vector machine classifiers, as defined in Equation (3.1). This formulation lets users define misclassification penalties per training instance $C_i,\ i = 1, \ldots, n$ and encompasses popular approaches such as C-SVC and class-weighted SVM

[34, 98].

{weightedsvm}

$$\min_{\mathbf{w},\xi,\rho} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{i=1}^{n} C_i\xi_i, \tag{3.1}$$

$$\text{subject to } y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + \rho) \geq 1 - \xi_i, \qquad i = 1, \ldots, n$$

$$\xi_i \geq 0, \qquad i = 1, \ldots, n$$

When aggregating SVM models, the base models often share support vectors (SVs). The `EnsembleSVM` software intelligently caches distinct SVs to ensure that they are only stored and used for kernel evaluations once. As a result, `EnsembleSVM` models are smaller and faster in prediction than ensemble implementations based on wrappers.

3.2.1 Implementation

`EnsembleSVM` has been implemented in `C++` and makes heavy use of the standard library. The main implementation focus is training speed. We use facilities provided by the `C++11` standard and thus require a moderately recent compiler, such as `gcc` ≥ 4.7 or `clang` ≥ 3.2. A portable Makefile system based on GNU autotools is used to build `EnsembleSVM`.

`EnsembleSVM` interfaces with `LIBSVM` to train base models [25]. Our code must be linked to `LIBSVM` but does not depend on a specific version. This allows users to choose the desired version of the `LIBSVM` software in the back-end.

The `EnsembleSVM` programming framework is designed to facilitate prototyping of ensemble algorithms using SVM base models. We particularly provide extensive support to define novel aggregation schemes, should the available options be insufficient. Key components are extensively documented in the code and on a wiki, which serves as a high-level guideline.[1] Intuitive APIs are provided for convenient features such as thread pools, command line interface and deserialization to enable users to develop new tools efficiently.

[1] The `EnsembleSVM` development wiki is available at `https://github.com/claesenm/EnsembleSVM/wiki`.

The EnsembleSVM library was built with extensibility and user contributions in mind. Major API functions are well documented to lower the threshold for external development. The executable tools provided with EnsembleSVM are essentially wrappers for the library itself. The tools can be considered as use cases of the main API functions to help developers.

3.2.2 Tools

The main tools in this package are esvm-train and esvm-predict, used to train and predict with ensemble models. Both of these are pthread-parallelized. Additionally, the merge-models tool can be used to merge standard LIBSVM models into ensembles. Finally, esvm-edit provides facilities to modify the aggregation scheme used by an ensemble.

EnsembleSVM includes a variety of extra tools to facilitate basic operations such as stratified bootstrap sampling, cross-validation, replacing categorical features by dummy variables, splitting data sets and sparsifying standard data sets. We recommend retaining the original ratio of positives and negatives in the training set when subsampling.

3.3 Benchmark Results

{bench}

To illustrate the potential of our software, EnsembleSVM 2.0 has been benchmarked with respect to LIBSVM 3.17. To keep the experiments simple, we use majority voting to aggregate predictions, even though more sophisticated methods are offered. For reference, we also list the best obtained accuracy with a linear model, trained using LIBLINEAR [49]. Linear methods are common in large-scale learning due to their speed, but may result in significantly decreased accuracy. This is why scalable nonlinear methods are desirable.

We used two binary classification problems, namely the covtype and ijcnn1 data sets.[2] Both data sets are balanced. Features were always scaled to $[0, 1]$. We have used C-SVC as SVM and base models ($\forall\ i : C_i = C$). Reported numbers are averages of 5 test runs to ensure reproducibility. We used the RBF kernel, defined by the kernel function $\kappa(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma||\mathbf{x}_i - \mathbf{x}_j||^2}$. Optimal parameter selection was done through cross-validation.

The covtype data set is a common classification benchmark featuring 54 dimensions [16]. We randomly sampled balanced training and test sets of

[2] Available at: http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html and UCI.

100,000 and 40,000 instances respectively and classified class 2 versus all others. The `ijcnn1` data set was used in a machine learning challenge during IJCNN 2001 [104]. It contains 35,000 training instances in 22 dimensions.

data set	**test set accuracy**			**no. of SVs**		**time (s)**	
	LIBSVM	LIBLINEAR	ESVM	LIBSVM	ESVM	LIBSVM	ESVM
covtype	0.92	0.76	0.89	26516	50590	728	35
ijcnn1	0.98	0.92	0.98	3564	7026	9.5	0.3

Table 3.1: Summary of benchmark results per data set: test set accuracy, number of support vectors and training time. Accuracies are listed for a single LIBSVM model, LIBLINEAR model and an ensemble model. {resultstable}

Results in Table 3.1 show several interesting trends. Training EnsembleSVM models is orders of magnitude faster, because training SVMs on small subsets significantly reduces complexity. Subsampling induces smaller kernels per base model resulting in lower overall memory use.

Ensembles can end up with more support vectors than a single SVM. Due to our parallelized implementation, prediction with ensemble models was faster than with LIBSVM models in both experiments even though the ensembles comprise twice as many SVs.

The ensembles in these experiments are competitive with a traditional SVM even though we used simple majority voting. For covtype, ensemble accuracy is 3% lower than a single SVM and for ijcnn1 the ensemble is marginally better (0.2%). Linear SVM falls far short in terms of accuracy for both experiments, but is trained much faster (< 2 seconds).

We obtained good results with very basic aggregation. **(author?)** [33] illustrated that more sophisticated aggregation methods can improve the predictive performance of ensembles. Others have reported performance improvements over standard SVM for ensembles using majority voting [123, 129].

3.4 Conclusions

EnsembleSVM provides users with efficient tools to experiment with ensembles of SVMs. Experimental results show that training ensemble models is significantly faster than training standard LIBSVM models while maintaining competitive predictive accuracy.

Linear methods are frequently applied in large-scale learning, mainly due to their low training complexity. Linear methods are known to have competitive accuracy for high dimensional problems. As our benchmarks showed, the difference in accuracy may be large for low dimensional problems. As such, fast nonlinear methods remain desirable in large-scale learning, particularly for low dimensional tasks with many training instances. Our benchmarks illustrate the potential of the ensemble approaches offered by `EnsembleSVM`.

Ensemble performance may be improved by using more complex aggregation schemes. `EnsembleSVM` currently offers various aggregation schemes, both linear and nonlinear. Additionally, it facilitates fast prototyping of novel methods through its `Pipeline` framework.[3]

`EnsembleSVM` strives to provide high-quality, user-friendly tools and an intuitive programming framework for ensemble learning with SVM base models. The software will be kept up to date by incorporating promising new methods and ideas when they are presented in the literature. User requests and suggestions are welcome and appreciated.

[3] `https://github.com/claesenm/EnsembleSVM/wiki/Pipeline`

Chapter 4

Hyperparameter Search in Machine Learning

{ch:mic2015}

This chapter has been previously published as:
Claesen, M., & De Moor, B. (2015). **Hyperparameter Search in Machine Learning**. In *Proceedings of the 11th Metaheuristics International Conference (MIC)*, Agadir, Morocco.
Manuscript available at `http://arxiv.org/abs/1502.02127`.

Abstract

We describe the hyperparameter search problem in the field of machine learning and discuss its main challenges from an optimization perspective. Machine learning methods attempt to build models that capture some element of interest based on given data. Most common learning algorithms feature a set of hyperparameters that must be determined before training commences. The choice of hyperparameters can significantly affect the resulting model's performance, but determining good values can be complex; hence a disciplined, theoretically sound search strategy is essential.

4.1 Introduction

Machine learning research focuses on the development of methods that are capable of capturing some element of interest from a given data set. Such elements include but are not limited to coherent structures within data (clustering) or the ability to predict certain target values based on given characteristics, which may be discrete (classification) or continuous (regression).

A large variety of learning methods exist, ranging from biologically inspired neural networks [15] over kernel methods [109] to ensemble models [22, 31]. A common trait in these methods is that they are parameterized by a set of hyperparameters λ, which must be set appropriately by the user to maximize the usefulness of the learning approach. Hyperparameters are used to configure various aspects of the learning algorithm and can have wildly varying effects on the resulting model and its performance.

Hyperparameter search is commonly performed manually, via rules-of-thumb [65, 64] or by testing sets of hyperparameters on a predefined grid [103]. These approaches leave much to be desired in terms of reproducibility and are impractical when the number of hyperparameters is large [28]. Due to these flaws, the idea of automating hyperparameter search is receiving increasing amounts of attention in machine learning, for instance via benchmarking suites [44] and various initiatives.[1] Automated approaches have already been shown to outperform manual search by experts on several problems [13, 10].

We briefly introduce some key challenges inherent to hyperparameter search in Section 4.2. The combination of all these hurdles make hyperparameter search a formidable optimization task. In Section 4.3 we give a succinct overview of the current state-of-the-art in terms of algorithms and available software.

4.1.1 Example: controlling model complexity

A key balancing act in machine learning is choosing an appropriate level of model complexity: if the model is too complex, it will fit the data used to construct the model very well but generalize poorly to unseen data (overfitting); if the complexity is too low the model won't capture all the information in the data (underfitting). This is often referred to as the bias-variance trade-off [52, 35], since a complex model exhibits large variance while an overly simple one is strongly biased. Most general-purpose methods feature hyperparameters to control this trade-off; for instance via regularization as in support vector machines and regularization networks [47, 62].

[1] Such as `http://www.automl.org/` and `https://www.codalab.org/competitions/2321`.

4.1.2 Formalizing hyperparameter search

The goal of many machine learning tasks can be summarized as training a model $\mathcal{M}$ which minimizes some predefined loss function $\mathcal{L}(\mathbf{X}^{(te)};\ \mathcal{M})$ on given test data $\mathbf{X}^{(te)}$. Common loss functions include mean squared error and error rate. The model $\mathcal{M}$ is constructed by a learning algorithm $\mathcal{A}$ using a training set $\mathbf{X}^{(tr)}$; typically involving solving some (convex) optimization problem. The learning algorithm $\mathcal{A}$ may itself be parameterized by a set of hyperparameters λ, e.g. $\mathcal{M} = \mathcal{A}(\mathbf{X}^{(tr)};\ \lambda)$. An example model $\mathcal{M}$ is a support vector machine classifier with Gaussian kernel [109], for which the training problem $\mathcal{A}$ is parameterized by the regularization constant C and kernel bandwidth σ, i.e. $\lambda = [C, \sigma]$.

The goal of hyperparameter search is to find a set of hyperparameters $\lambda^\star$ that yield an optimal model $\mathcal{M}^\star$ which minimizes $\mathcal{L}(\mathbf{X}^{(te)};\ \mathcal{M})$. This can be formalized as follows [28]:

$$\lambda^\star = \arg\min_{\lambda} \mathcal{L}\big(\mathbf{X}^{(te)};\ \mathcal{A}(\mathbf{X}^{(tr)};\ \lambda)\big) = \arg\min_{\lambda} \mathcal{F}(\lambda;\ \mathcal{A},\ \mathbf{X}^{(tr)}, \mathbf{X}^{(te)},\ \mathcal{L}). \quad (4.1)$$ {equation}

The objective function $\mathcal{F}$ takes a tuple of hyperparameters λ and returns the associated loss. The data sets $\mathbf{X}^{(tr)}$ and $\mathbf{X}^{(te)}$ are given and the learning algorithm $\mathcal{A}$ and loss function $\mathcal{L}$ are chosen . Depending on the learning task, $\mathbf{X}^{(tr)}$ and $\mathbf{X}^{(te)}$ may be labeled and/or equal to each other. In supervised learning, a data set is often split into $\mathbf{X}^{(tr)}$ and $\mathbf{X}^{(te)}$ using hold-out or cross-validation methods [43, 75].

4.2 Challenges in hyperparameter search

{challenges}

The characteristics of the search problem depend on the learning algorithm $\mathcal{A}$, the chosen loss function $\mathcal{L}$ and the data set $\mathbf{X}^{(tr)}$, $\mathbf{X}^{(te)}$, as shown in Equation (4.1). Hyperparameter search is typically approached as a non-differentiable, single-objective optimization problem over a mixed-type, constrained domain. In this section we will discuss the origins and consequences of challenges in hyperparameter search.

4.2.1 Costly objective function evaluations

{time}

Each objective function evaluation requires evaluating the performance of a model trained with hyperparameters λ. Depending on the available computational resources, the nature of the learning algortihm $\mathcal{A}$ and size of the problem ($\mathbf{X}^{(tr)}$, $\mathbf{X}^{(te)}$) each evaluation may take considerable time.

Training times in the order of minutes are considered fast, since days and even weeks are not unheard of [77, 38, 119]. Evaluation time is exacerbated when procedures that train multiple models are employed; for instance to reliably estimate generalization performance [43, 75]. This leads to an increasing need for efficient methods to optimize hyperparameters that require a minimal amount of objective function evaluations.

Additionally, the time required to train and test models can be contingent upon the choice of hyperparameters. Some hyperparameters have an obvious influence on train and/or test time, e.g. the architecture of neural networks [15] and size of ensembles [22, 31]. The influence of hyperparameters can also be subtle, for instance regularization and kernel complexity can significantly affect training time for support vector machines [18].

4.2.2 Randomness

{randomness}

The objective function often exhibits a stochastic component, which can be induced by various components of the machine learning pipeline, for example due to inherent randomness of the learning algorithm (initialization of a neural network, resampling in ensemble approaches, ...) or due to finite sample effects in estimating generalization performance. This stochasticity can sometimes be addressed via machine learning techniques; but unfortunately such solutions typically dramatically increase the time required per objective function evaluation, limiting their usefulness in some settings.

This inherent stochasticity directly implies that the empirical best hyperparameter tuple, obtained after a given set of evaluations, is not necessarily the true optimum of interest $\lambda^\star$. Fortunately, many search methods are designed to probe many tuples close to the empirical best. If the search region surrounding the empirical optimum is densely sampled, we can determine whether the empirical best was an outlier or not in a post-processing phase, for instance by assuming Lipschitz continuity or smoothness.

4.2.3 Complex search spaces

The number of hyperparameters is usually small (≤ 5), but it can range up to hundreds for complex learning algorithms [12] or when preprocessing steps are also subjected to optimization [68]. It has been demonstrated empirically that in many cases only a handful of hyperparameters significantly impact performance, though identifying the relevant ones in advance is difficult [10].

Hyperparameters are usually of continuous or integer type, leading to mixed-type optimization problems. Continuous hyperparameters are commonly related to regularization. Common integer hyperparameters are related to network architecture for neural networks [15], size of ensembles [22, 31] or the parameterization of kernels in kernel methods [109].

Some tasks feature highly complex search spaces, in which the very existence of certain hyperparameters are conditional upon the value of others [68, 13, 11]. A simple example is optimizing the architecture of neural networks [15], where the number of hidden layers is one hyperparameter and the size of each layer induces a set of additional hyperparameters, conditional upon the number of layers.

4.3 Current approaches

{state-of-the-art}

A wide variety of optimization methods have been used for hyperparameter search, including particle swarm optimization [92, 82], genetic algorithms [120], coupled simulated annealing [134] and racing algorithms [14]. Surprisingly, randomly sampling the search space was only established recently as a baseline for comparison of optimization methods [10]. Bayesian and related sequential model based optimization techniques using variants of the expected improvement criterion [70] are receiving a lot of attention currently [13, 67, 116, 6, 44], owing to their efficiency in terms of objective function evaluations.

Software packages are being released which implement various dedicated optimization methods for hyperparameter search. Such packages are usually intended to be used in synergy with machine learning libraries that provide learning algorithms [103]. Most of these packages focus on Bayesian methods [68, 116, 11], though metaheuristic optimization approaches are also offered [28]. The increased development of such packages testifies towards the growing interest in automated hyperparameter search.

4.4 Conclusion

A fully automated, self-configuring learning strategy can be considered the holy grail of machine learning. Though the current state-of-the-art still has a long way to go before this goal can be reached, it is evident that hyperparameter search is a crucial element in its pursuit. Automated hyperparameter search is a hot topic within the machine learning community which we believe can benefit greatly from the techniques and lessons learnt in metaheuristic optimization.

Chapter 5

Building Classifiers to Predict the Start of Glucose-Lowering Pharmacotherapy Using Belgian Health Expenditure Data

{ch:diabetesjmlr}

This chapter has been submitted to:
Claesen, M., De Smet, F., Gillard, P., Mathieu, C.& De Moor, B. (2015). **Building Classifiers to Predict the Start of Glucose-Lowering Pharmacotherapy Using Belgian Health Expenditure Data**. *Journal of Machine Learning Research: special issue on Learning from Electronic Health Data*.

Abstract

Early diagnosis is important for type 2 diabetes (T2D) to improve patient prognosis, prevent complications and reduce long-term treatment costs. We present a novel risk profiling approach based exclusively on health expenditure data that is available to Belgian mutual health insurers. We used expenditure data related to drug purchases and medical provisions to construct models that predict whether a patient will start glucose-lowering pharmacotherapy in the coming years, based on that patient's recent medical expenditure history. The design and implementation of the modeling strategy are discussed in detail and several learning methods are benchmarked for our application. Our best performing model obtains between 74.9% and 76.8% area under the ROC curve, which is comparable to state-of-the-art risk prediction approaches for T2D based on questionnaires. In contrast to other methods, our approach can be implemented on a population-wide scale at virtually no extra operational cost. Possibly, our approach can be further improved by additional information about some risk factors of T2D that is unavailable in health expenditure data.

5.1 Introduction

Type 2 diabetes mellitus (T2D) is a chronic metabolic disorder characterized by hyperglycemia and is considered one of the main threats to human health [139]. In developed countries, T2D makes up about 85% of diabetes mellitus patients and occurs when either insufficient insulin is produced, the body becomes resistant to insulin or both [132]. Prediabetes and less severe cases of T2D are initially managed by lifestyle changes, specifically increasing physical exercise, dietary change and smoking cessation [121, 40, 4]. If this yields insufficient glycemic control, pharmacotherapy with glucose-lowering agents (GLAs) like metformin or insulin is started [122, 4].

Several studies have indicated that one third to one half of T2D patients are undiagnosed [59, 74, 108]. Additionally, patients often remain undiagnosed for extended periods of time, with average diagnose-free intervals ranging from 4 to 7 years [61]. The prognosis of untreated patients can deteriorate rapidly as prolonged hyperglycemia can cause serious damage to many of the body's systems. Timely diagnosis of T2D proves challenging in contemporary medicine, as many patients already present signs of complications of the disease at the time of clinical diagnosis of T2D [60, 105, 76, 7, 58, 66].

Earlier diagnosis and subsequent treatment is believed to prevent or delay complications and improve prognosis [101, 46]. When impaired glucose tolerance is diagnosed early, initial treatment can often be limited to lifestyle changes [99, 121, 40]. Compared to pharmacotherapy, lifestyle changes are simple, fully manageable by the patient and far less likely to cause serious treatment-induced complications like hypoglycemia [112, 137]. Complementary to health benefits, early diagnosis of T2D poses a health economical advantage, as patients that do not require acute or intensive long-term treatment are far less demanding on the health care system.

Universal screening for T2D is cost-prohibitive [130, 46], but many organizations advise opportunistic screening of high-risk subgroups [132, 2, 46, 4]. Several risk profiling strategies have been developed to aid in the timely diagnosis of T2D [5, 118, 83, 91, 26, 63, 111]. Risk profiling is typically done by assessing some of the key risk factors for T2D, which include obesity [93], genetic predisposal [113, 69], lifestyle [107] and various clinical parameters. Existing risk profiling approaches are implemented via questionnaires, potentially augmented with clinical information that is available to the patient's general practitionner [55, 117, 83, 53, 110, 63]. Commonly required information includes BMI, family history, exercise and smoking habits and various clinical parameters.

In this work, we present an alternative approach for risk profiling which only requires data that is already available to Belgian mutual health insurers.

This work was done in collaboration with the National Alliance of Christian Mutualities (NACM). NACM is the largest Belgian mutual health insurer with over four million members. Our approach does not require any questionnaires or additional clinical information and predicts whether a patient will start taking GLAs in the next few years. Interestingly, our approach works well despite the fact that Belgian health insurer data contains little direct information regarding key risk factors of T2D, that is weight, lifestyle and family history are all unavailable.

5.2 Existing Type 2 Diabetes Risk Profiling Approaches

{sec:stateoftheart}

The Cambridge Risk Score (CRS) was developed to assess the probability of undiagnosed T2D based on data that is routinely available in primary care records, including age, sex, medication use, family history of diabetes, BMI and smoking status [55], The CRS has been shown to be useful on multiple occasions [55, 100, 117], though its AUC seems to depend heavily on the population in which it is used, ranging between 67% [117] and 80% [55]. The information used in the CRS is comparable to another approach which obtained AUCs ranging between 70% and 78% [5].

The FINDRISC score is based on a 10-year follow-up using age, BMI, waist circumference, history of antihypertensive drugs and high blood glucose, physical activity and diet with reported AUCs of 85% and 87% in predicting drug-treated diabetes [83]. The strongest reported predictors in this study were BMI, waist circumference, history of high blood glucose and physical activity. **(author?)** [53] developed a risk score based on age, sex, BMI, known hypertension, physical activity and family history of diabetes with AUC ranging from 72% to 87.6%. The German diabetes risk score reached AUCs ranging from 75% to 83% on validation data and is based on age, waist circumference, height, history of hypertension, physical activity, smoking, and diet [110].

(author?) [63] developed a decision tree for risk prediction achieving 82% AUC in a cross-validation setting, based on weight, age, family history and various clinical parameters. Various other approaches based on routine clinical information have demonstrated similarly accurate predictions of type 2 diabetes [118, 91].

5.3 Health Expenditure Data

The Belgian health care insurance is a broad solidarity-based form of social insurance. Mutual health insurers such as NACM are the legally-appointed bodies for managing and providing the Belgian compulsory health care and disability insurance, among other things. To implement their operations, Belgian mutual health insurers dispose of large databases containing health expenditure records of all their respective members.

These expenditure records hold all financial reimbursements of drugs, procedures and contacts with health care professionals. Each record comprises a timestamp, financial details and a description of the claim. The financial aspect is irrelevant from a medical point of view, but the type of resource-use as indicated by the description can contain medical information about the patient. These types belong to one of two main categories:

1. **Drug purchases** are recorded per package. The coding of packages contains information about the active substances in the drug along with the volume of the package.
2. **Medical provisions** are identified by a national encoding along with an identifier of the associated medical caregiver. Each provision has a distinct code number.

In addition to resource-use data, some biographical information is available about each patient including age, gender, place of residence and social parameters. In the remainder of this Section we will elaborate on expenditure records related to drugs and provisions as used in our models. Subsequently we will briefly summarize the main strengths and limitations of using health expenditure data for predictive modeling.

5.3.1 Records Related to Drug Purchases

Expenditure records concerning drug purchases contain information about the active substances in the drug and the purchased volume. We mapped all active substances onto the anatomical therapeutic chemical (ATC) classification system maintained by the **(author?)** [131]. The ATC classification system divides active substances into different groups based on the organ or system on which they act and their therapeutic, pharmacological and chemical properties. Each drug is classified in groups at 5 levels in the ATC hierarchy: fourteen main groups (1st level), pharmacological/therapeutic subgroups (2nd level), chemical subgroups (3rd and 4th level) and the chemical substance (5th level).

After mapping records onto the ATC classification system, a patient's medication history consists of specific ATC codes (5th level) along with the associated number of defined daily doses (DDD). In the period of interest, purchases of 4,580 distinct active substances were recorded in the NACM database. Table 5.1 shows an example of the classification of active substance on all levels in the ATC system.

level	ATC code	description
1	A	alimentary tract and metabolism
2	A10	drugs used in diabetes
3	A10B	blood glucose lowering drugs, excluding insulins
4	A10BA	biguanides
5	A10BA02	metformin

Table 5.1: Example of the ATC classification system: classification of metformin per level.

{table:atc-example}

5.3.2 Records Related to Medical Provisions

Expenditure records concerning medical provisions can be considered tuples containing time-stamped identifiers of the patient, physician and medical provision. A single patient-physician interaction may yield multiple such records, one for each specific provision that occurred.

In the Belgian health care system, medical provisions are encoded via the Belgian nomenclature of medical provisions [126], which is maintained by the National Institute for Health and Disability Insurance (NIHDI).[1] This nomenclature is an unstructured list of unique codes (numbers) for each provision that is being refunded. Nomenclature numbers are added when new provisions are defined or when revisions are made. A single provision may correspond to multiple numbers for various reasons.

5.3.3 Advantages of Health Expenditure Data

The key benefit of expenditure databases is that they centralize structured medical information across all medical stakeholders to yield a comprehensive, longitudinal overview of each patient's medical history. Other health data sources are commonly fragmented, e.g. medical records maintained by the

[1] The website of NIHDI is available at `http://www.riziv.fgov.be`.

patient's general practitioner or hospital often contain only a subset of the patient's medical history. This fragmentation hampers the identification of patterns that may indicate elevated risk for diseases like type 2 diabetes. The NACM database comprises claims records of over four million Belgians, which enables complex modeling. Additionally, claims data have few omissions due to the financial incentive for patients and medical stakeholders (e.g., hospitals) to claim refunds. While other health data sources may contain more detailed information, the strength of NACM's data is in its volume, both in terms of number of patients and the amount of information that is recorded per individual. Finally, as most people tend to stay affiliated with the same mutual health insurer, their expenditure records provide long-term information.

5.3.4 Limitations of Health Expenditure Data

Belgian health expenditure data is strictly limited to what is required for mutual health insurers to implement their operations, which are mainly administrative in nature. Detailed health information such as diagnoses and test results are not directly available. In some other countries, health insurers dispose of more detailed information, such as ICD-10 codes which include diagnoses and symptoms [133]. Including such information is out of scope of this work as we focus exclusively on data that is already available to Belgian mutual health insurers. Biographical information about patients does not contain direct information about some important risk factors such as lifestyle, family history and BMI, though this may be partially embedded indirectly in medical resource-use.

5.4 Methods

In this Section we define the prediction task and describe all its aspects: the overall setup (Section 5.4.1), the data and its representation (Section 5.4.2) and the learning algorithms (Section 5.4.3). Briefly, our aim is to predict which patients will start glucose-lowering pharmacotherapy within the next 4 years, based on expenditure records of the previous 4 years.

Our key hypothesis is that patients with increased risk for T2D or those that are already afflicted but not diagnosed have a different medical expenditure history than patients without impaired glycemic control. We essentially use the start of GLA therapy as a proxy for diagnosis of (advanced) type 2 diabetes. This is reasonable since most patients that start GLA therapy above 40 years old have T2D [132].

We posed this task as a binary classification problem. Our classifiers produce a numeric level of confidence that a given patient will start glucose-lowering pharmacotherapy. When predicting a population, the outputs can be used to rank patients according to decreasing confidence that the patients will start glucose-lowering therapy. Highly ranked patients represent a high-risk subgroup which can be targetted for clinical screening. Briefly, we used nested cross-validation to obtain unbiased estimates of the predictive performance of each vectorization and learning approach. Predictive performance of all models was quantified via (area under) receiver operating characteristic (ROC) curves.

Data Our work is based on a subset of the expenditure records of NACM. All data extractions and analyses were performed at the Medical Management Department of the NACM under supervision of the Chief Medical Officer. The other research partners received no personally identifiable information (including small cells) from NACM. The patient selection protocol and vector representations are described in detail in Section 5.4.2.

Class definitions The positive class was defined as patients that require GLAs for long-term glycemic control.[2] The negative class is then defined as patients that do not need GLAs. Expenditure records related to GLAs were used to identify a set of known positives. However, the absence of such records in a patient's resource use history is not proof that this patient has no need for GLAs. This subtle difference is crucial, because it is well known that patients with impaired glycemic control or T2D often remain undiagnosed and hence untreated for a very long time [59, 74, 4]. As we cannot identify negatives, we had to build models from positive and unlabeled data.

PU learning Learning binary classifiers from positive and unlabeled data (PU learning) is a well-studied branch of semi-supervised learning [80, 45, 94, 32]. PU learning is more challenging than fully supervised binary classification, since it requires special learning approaches and quality metrics for hyperparameter optimization that account for the lack of known negatives. We benchmarked three PU learning methods, which are discussed in more detail in Section 5.4.3.

Software The entire data analysis pipeline was implemented using open-source software. For general data transformations and preprocessing we used *SciPy*

[2] GLAs are defined as any drug in ATC category `A10`, which includes metformin, sulfonylurea and insulin.

and *NumPy* [71, 127]. The learning algorithms we used are available in *scikit-learn* and *EnsembleSVM* [103, 31] . Finally, we used *Optunity* for automated hyperparameter optimization [29].

5.4.1 Experimental Setup

{setup}

We gathered all expenditure records during the 4-year interval of 2008 up to 2012. The selection protocol and representations of patients' medical resource-use are discussed in detail in Section 5.4.2. All vector representations of patients include age (in years), an indicator variable for gender and positive entries related to the patient's medical resource-use. A patient vector $\mathbf{p}$ can be written in the following general form, where d_{meds} and d_{provs} denote the number of features in the vectorization of medication and provision use, respectively:

$$\mathbf{p} \in \mathbb{R}_+^{2+d_{\text{meds}}+d_{\text{provs}}} = \begin{bmatrix} age & gender & medication & provisions \\ \mathbb{R}_+ & \{0,1\} & \mathbb{R}_+^{d_{\text{meds}}} & \mathbb{R}_+^{d_{\text{provs}}} \end{bmatrix}. \tag{5.1}$$

In Sections 5.4.2 and 5.4.2 we explain how records related to medication purchases and provisions were represented in vector form. All entries in the vector representations were consistently normalized to the interval $[0, 1]$ by dividing feature-wise by the 99^{th} percentile and subsequently clipping where necessary. These normalized vector representations are used as inputs for the learning algorithms described in Section 5.4.3.

Figure 5.1 summarizes the full machine learning pipeline, which starts from expenditure records and ends with models to predict whether a patient will start glucose-lowering pharmacotherapy along with an estimate of their generalization performance. We used nested cross-validation to estimate generalization performance of different learning configurations [128]. The outer 3-fold cross-validation is used to estimate generalization performance of the full learning approach. Internally, twice iterated 10-fold cross-validation was used to find optimal hyperparameters for every learning method.

Model evaluation Models are compared based on area under the ROC curve. ROC curves visualize a classifier's performance spectrum by depicting its true positive rate (TPR)[3] as a function of its false positive rate (FPR)[4] while varying the decision threshold to decide on positives. Area under the ROC curve (AUROC) is a useful summary statistic of a classifier's performance. AUROC

[3]TPR measures the fraction of true positives that are correctly identified by the classifier.
[4]FPR measures the fraction of true negatives that are incorrectly identified by the classifier.

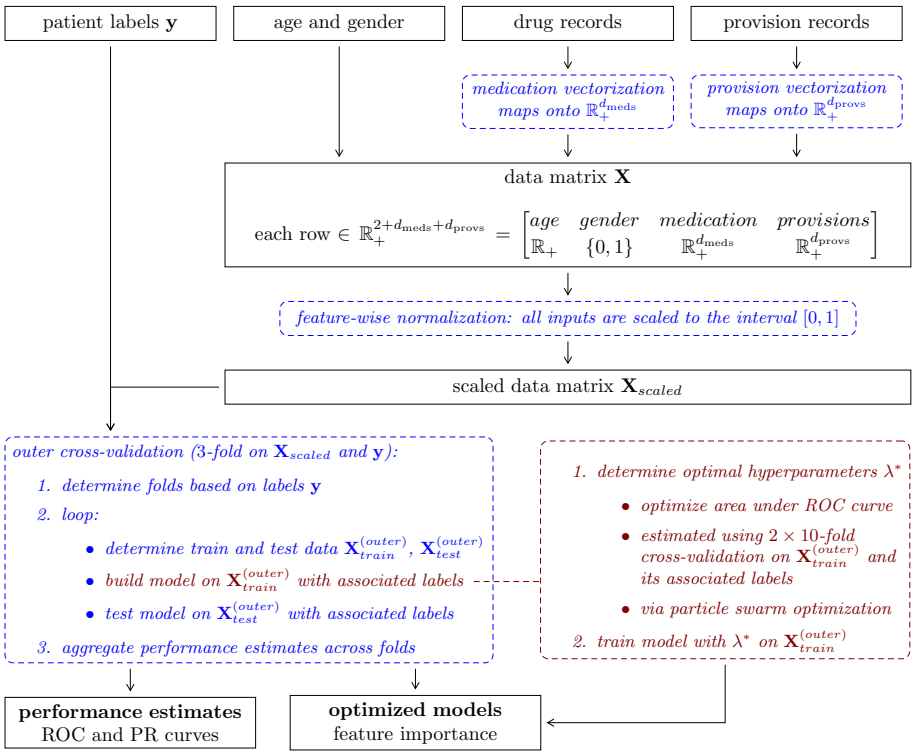


Figure 5.1: Overview of the full learning approach: data set vectorization, normalization and the nested cross-validation setup. Per iteration, hyperparameter optimization and model training is done based exclusively on $\mathbf{X}_{train}^{(outer)}$.

{fig:flowchart}

is equal to the probability that the classifier ranks a random positive higher than a random negative and is known to be equivalent to the Wilcoxon test of ranks [57].

Hyperparameter search We used Optunity's particle swarm optimizer to identify suitable hyperparameters for each approach based on the given training set as defined by the outer cross-validation procedure [29]. Every tuple of hyperparameters was evaluated using twice iterated 10-fold cross-validation on the training set. Per technique, the hyperparameters that maximized cross-validated area under the ROC curve were selected and used to train a model on the full training set.

Computing ROC curves Full label knowledge is required to compute ROC curves. In previous work, we introduced a method to compute bounds on ROC curves based on positive and unlabeled data [27]. Briefly, it is based on the positions of known positives in a ranking produced by a given classifier and requires two things:

- The rank distributions of labeled and latent positives must be comparable. This holds when known and latent positives follow the same distribution in input space (ie. the vector representation of patients). This is a fair assumption in our application, since we specifically ignore records after the start of glucose-lowering pharmacotherapy while identifying the set of positives (see Section 5.4.2), so the medication regimen of known positives has not yet diverged from the regimen of untreated patients.
- An estimate $\hat{\beta}$ of the fraction of latent positives in the unlabeled set is needed, that is the fraction of members that have never used GLAs but are likely to start glucose-lowering pharmacotherapy. In the period 2010–2014 roughly 8% of members of NACM aged 40 or higher started using GLAs. Underestimating $\hat{\beta}$ results in an underestimated ROC curve and vice versa [27]. We opted to be conservative and used $\hat{\beta}_{lo} = 5\%$ to estimate lower bounds and $\hat{\beta}_{up} = 10\%$ for upper bounds.

We consistently used the *lower* bounds for hyperparameter search. All our performance reports contain lower and upper bounds, based on $\hat{\beta}_{lo}$ and $\hat{\beta}_{up}$, respectively.

Diagnosing overfitting In addition to measuring performance, we diagnosed overfitting via the concept of rank distributions as defined by **(author?)** [27]. The rank distribution of a subset of test instances is defined as the distribution of the positions of these test instances in a ranking of the full test set based on a model's predicted decision values. We diagnose overfitting based on the rank distributions of known positive training instances ($\mathcal{P}_{train}$) and known positives in the independent test fold ($\mathcal{P}_{test}$) after predicting the full data set. If the model overfits, the rank distribution of $\mathcal{P}_{train}$ is inconsistent with the rank distribution of $\mathcal{P}_{test}$. Specifically, ranks in $\mathcal{P}_{test}$ are worse than those in $\mathcal{P}_{train}$ when the model overfits. This can be quantified via the Mann-Whitney U test [89] based on ranks of $\mathcal{P}_{train}$ and $\mathcal{P}_{test}$ after predicting the full data set (that is all outer folds). The Mann-Whitney U test is expected to yield a non-significant result when the rank distributions of $\mathcal{P}_{train}$ and $\mathcal{P}_{test}$ are comparable. We report the average p-values of the test across outer cross-validation folds for each model (low p-values indicate overfitting).

5.4.2 Data Set Construction

{data}

We constructed a data set containing records of patients born before 1973 (e.g. 40 or more years old in 2012). Patients with records of glucose-lowering agents (GLAs) during less than 30 days were discarded. Patients with records of glucose-lowering therapy prior to 2012 were discarded. Patients that joined NACM after 2005 were also discarded, as we cannot determine whether these patients used GLAs in the recent past.

All patients that started glucose-lowering pharmacotherapy in 2012 or later are included as known positives ($n = 31,066$), along with unlabeled patients that were sampled at random from the remaining NACM members ($n = 79,243$). Known positives have a minimum of 30 days between the first and last purchase of GLAs to avoid contaminating the data set with false positives, for instance due to insulin use in surgical and medical ICUs [125, 124]. It must be noted that some false positives remain, that is patients that use GLAs but not for glycemic control.

In Sections 5.4.2 and 5.4.2 we describe the vector representations of records regarding medication and medical provisions, respectively.

Representation of Medication Records

{ATC-vectorization}

The simplest way to represent medication purchases during a time interval is by having one input dimension per active substance (level 5 ATC codes) and counting the purchased volume in terms of DDDs. This representation is easy to construct but fails to capture any similarity between active substances, such as the system or organ on which they act.

Imposing structure We can directly use the hierarchical structure of the ATC system to define a measure of similarity between drugs. To impose structure between drugs we included input dimensions related to more generic levels of the ATC hierarchy (levels 1 to 4). On more generic levels we summed all DDD counts of active substances per category (level 5). This redundancy allowed us to express similarity between different active substances with a standard inner product. By normalizing every feature to the unit interval, we obtained the desired effect that patients with comparable drug use on ATC level 5 are more similar than patients that only share coefficients on more generic levels. Figure 5.2 illustrates this vector representation of trees and the effect of normalization.

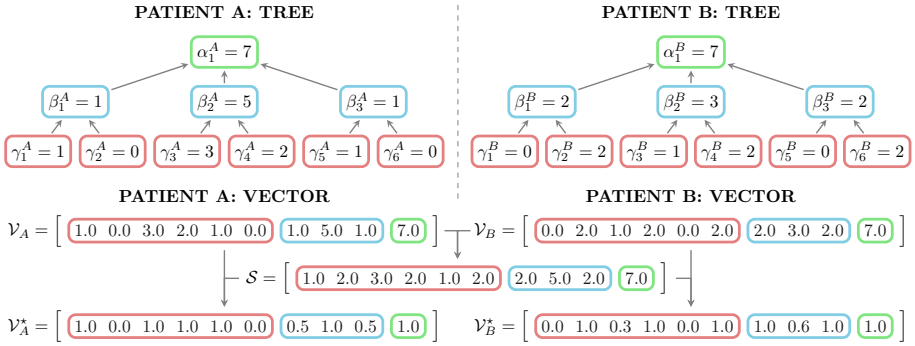


Figure 5.2: Visualization and vectorization of trees. In the tree representation, the value of internal nodes is the sum of the values of its children. The unnormalized vector representations $\mathcal{V}_A$ and $\mathcal{V}_B$ contain the values per node in the tree representation in some fixed order. Inner products between unnormalized representations $\mathcal{V}_A$ and $\mathcal{V}_B$ are mainly influenced by the top level nodes, since those have the largest value by construction. This undesirable effect can be fixed through feature-wise scaling. The scaling vector $\mathcal{S}$ was constructed using node-wise maxima. The normalized vector representations $\mathcal{V}_A^\star$ and $\mathcal{V}_B^\star$ are obtained by dividing the vector representations ($\mathcal{V}_A$, $\mathcal{V}_B$) element-wise by entries in the scaling vector $\mathcal{S}$. $\mathcal{V}_A^\star$ and $\mathcal{V}_B^\star$ are used as input to classifiers in the remainder of this work. As desired, the inner product of normalized vector representations is increasingly influenced by similarities at higher depths in the tree representations.

{medication-vectorization}

Summary All vectorizations related to drug purchases are described in Table 5.2.

vectorization	description	d_{meds}
ATC 5	counts of DDDs per medication class in ATC level 5	4,580
ATC 1–4	counts of DDDs per medication class in ATC levels 1–4	1,257
ATC 1–5	counts of DDDs per medication class in ATC levels 1–5	5,837

{table:meds-vects}

Table 5.2: Summary of vectorization schemes used for records of drug purchases.

Representation of Provision Records

{nomenclature}

When considering a specific time period, we can describe records by a (sparse) three-dimensional tensor containing frequency counts as illustrated in Figure 5.3. We filtered all provisions with a description containing *diabetes*, *insulin* and

glucose and provisions not recorded with a physician identifier. After filtering, 5,799 distinct provision codes remain (denoted by *#provisions*).

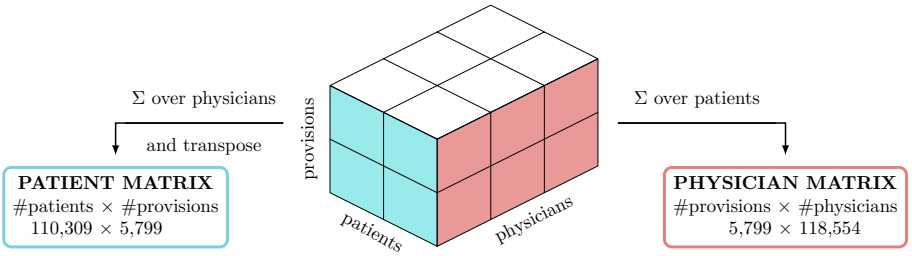


Figure 5.3: Tensor formulation of medical provisions with three components: patients, physicians and provisions. Each entry in the tensor is the frequency of the given tuple. This provision tensor is very sparse. The patient matrix is obtained by summing counts over all physicians (transposed). The physician matrix is obtained by summing counts over all patients. These matrices capture complementary information.

{fig:provisions-tensor}

Each patient is modelled by a histogram of their provisions in the period of interest. This essentially means we compute the sum over the *physician*-component of the tensor representation to obtain a matrix, in which rows and columns represent patients and provisions, respectively. Unfortunately, the encoding of provisions has no medically relevant structure in contrast to the ATC hierarchy for drugs as discussed in Section 5.4.2.

Imposing structure In order to define a reasonable similarity measure between patients, we first had to impose a structure onto the nomenclature that captures similarity between provisions. To structure provisions, we should not use information originating from the patient matrix, as this may cause information leaks (since the patient matrix is used directly in our models for prediction). Instead, we used the complementary physician matrix as a basis to define similarity between provisions, which essentially serves as a proxy for the medical specializations to which each provision belongs.

First, we computed cosine similarity between provisions based on the physician matrix. We used cosine similarity because it is known to work well for text mining with bag-of-words representations, which is comparable to our use case as it also features sparse, high dimensional input spaces. The cosine similarity κ_{cos} between two row vectors $\mathbf{u}$ and $\mathbf{v}$ is defined as:

{eq:cosine}

$$\kappa_{cos}(\mathbf{u}, \mathbf{v}) = \frac{\langle \mathbf{u}, \mathbf{v} \rangle}{\|\mathbf{u}\| \cdot \|\mathbf{v}\|} = \frac{\mathbf{u}\mathbf{v}^T}{\|\mathbf{u}\| \cdot \|\mathbf{v}\|}. \tag{5.2}$$

Using cosine similarity we can construct a pair-wise similarity matrix $\mathbf{S}_{prov}$ between provisions based on the rows of the physician matrix $\mathbf{x}_i, i = 1..\#provisions$:

$$\mathbf{S}_{prov} = \left(\kappa_{cos}(\mathbf{x}_i, \mathbf{x}_j)\right)_{ij} \in \mathbb{R}^{\#provisions \times \#provisions}. \tag{5.3}$$ {eq:cosine-kernel}

$\mathbf{S}_{prov}$ expresses similarity between provision codes based on the physicians that provide them and can be regarded as a proxy for the medical subdomain each provision frequently occurs in. In our context, its entries range from 0 (completely orthogonal) to +1 (exact similarity). To impose sparsity we set all entries of $\mathbf{S}_{prov}$ below 0.05 to 0. Its structure is visualized in Figure 5.4, which clearly indicates that our approach successfully identifies some coherent groups of provisions.

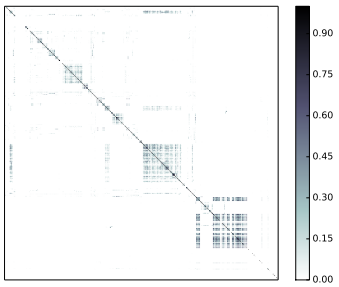


Figure 5.4: Structure of the provision similarity matrix $\mathbf{S}_{prov}$ based on providing physicians. {fig:Sprovs}

Finally, the structured representation of provisions $\mathbf{P}_{struct}$ is defined as the matrix product between the patient matrix $\mathbf{P}_{flat}$ and the provision similarity matrix $\mathbf{S}_{prov}$:

$$\mathbf{P}_{struct} = \mathbf{P}_{flat} \times \mathbf{S}_{prov} \in \mathbb{R}^{\#patients \times \#provisions}. \tag{5.4}$$

$\mathbf{P}_{struct}$ approximately captures which provisions occur in a patient's history with redundancy based on medical specializations.

Summary All vectorizations related to medical provisions are described in Table 5.3.

vectorization	symbol	description	d_{provs}
PROVS FLAT	$\mathbf{P}_{flat}$	entries taken from the patient matrix	5,799
PROVS STRUCT	$\mathbf{P}_{struct}$	captures similarity between provisions	5,799
PROVS BOTH	$\mathbf{P}_{flat} \mid \mathbf{P}_{struct}$	concatenation of flat & structured	11,598

Table 5.3: Summary of vectorization schemes used for records of medical provisions. {table:prov-vects}

5.4.3 Learning Methods {learning-methods}

Having only positive and unlabeled data (PU learning) presents additional challenges for learning algorithms. Two broad classes of approaches exist to tackle these problems: (i) two-phase methods that first attempt to identify likely negatives from the unlabeled set and then train a supervised model on the positives and inferred negatives [87, 135] and (ii) approaches that treat the unlabeled set as negatives with label noise [45, 80, 94, 32].

We have tested three approaches from the latter category in this work, namely class-weighted SVM [86], bagging SVM [94] and the robust ensemble of SVM models [32]. All of these approaches are based on support vector machines. We used the linear kernel on vector representations of patients as described in Section 5.4.2.[5] We will briefly introduce each method in the following subsections.

Class-weighted SVM {bsvm}

Class-weighted SVM (CWSVM) uses a misclassification penalty per class. CWSVM was first applied in a PU learning context by **(author?)** [86], by considering the unlabeled set to be negative with noise on its labels. A CWSVM is trained to distinguish positives ($\mathcal{P}$) from unlabeled instances ($\mathcal{U}$), leading to

[5] Though it must be noted that the ensemble methods are always implicitly nonlinear.

the following optimization problem:

{eq:bsvm}
$$\min_{\alpha,\xi,b} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \kappa(\mathbf{x}_i,\mathbf{x}_j) + C_{\mathcal{P}}\sum_{i\in\mathcal{P}}\xi_i + C_{\mathcal{U}}\sum_{i\in\mathcal{U}}\xi_i, \tag{5.5}$$

$$\text{s.t. } y_i(\sum_{j=1}^{N}\alpha_j y_j \kappa(\mathbf{x}_i,\mathbf{x}_j) + b) \geq 1 - \xi_i, \qquad i = 1,\ldots,N,$$

$$\xi_i \geq 0, \qquad i = 1,\ldots,N,$$

where $\alpha \in \mathbb{R}^N$ are the support values, $\mathbf{y} \in \{-1,+1\}^N$ is the label vector, $\kappa(\cdot,\cdot)$ is the kernel function, b is the bias term and $\xi \in \mathbb{R}^N$ are the slack variables for soft-margin classification. The misclassification penalties $C_{\mathcal{P}}$ and $C_{\mathcal{U}}$ require tuning. We used the implementation available in scikit-learn [103] based on LIBSVM [25].

Bagging SVM

{baggingsvm}

In bagging SVM, random resamples are drawn from the unlabeled set and CWSVM classifiers are trained to discriminate all positives from each resample [94]. Resampling the unlabeled set induces variability in the base models which is exploited via bagging. Base model predictions are aggregated via majority voting.

Bagging SVM with linear base models has two hyperparameters, namely the size of resamples of the unlabeled set $n_{\mathcal{U}}$ and the misclassification penalty on unlabeled instances $C_{\mathcal{U}}$. The misclassification penalty on positives $C_{\mathcal{P}}$ is fixed via the following rule:

{eq:bagpenalties}
$$C_{\mathcal{P}} = \frac{n_{\mathcal{U}} \times C_{\mathcal{U}}}{|\mathcal{P}|}, \tag{5.6}$$

where $|\mathcal{P}|$ denotes the number of known positives. The heuristic rule in Equation 5.6 is common in imbalanced settings [24, 36]. We implemented bagging SVM using the EnsembleSVM library [31].

Robust Ensemble of SVM models

The robust ensemble of SVM models (RESVM) is a modified version of bagging SVM in which both the positive and unlabeled sets are resampled when constructing base model training sets [32]. The extra resampling induces additional variability between base models which improves performance

when combined with a majority vote aggregation scheme. **(author?)** [32] demonstrated that resampling the positive set provides robustness against false positives, which makes RESVM appealing for our application since our data set is known to contain a small fraction of false positives (as explained in Section 5.4.2).

When using linear base models, the RESVM approach has four hyperparameters that must be tuned, namely resample sizes and misclassification penalties per class. This approach was implemented based on EnsembleSVM [31].

5.5 Results and Discussion

Section 5.5.1 shows the predictive performance per learning configuration and compares these performances to the current state-of-the-art in large-scale risk assessment for T2D. Section 5.5.2 shows performance curves of the best configuration, which enable us to determine suitable cutoffs to identify target groups in practice. Finally, Section 5.5.3 describes a simple approach to assess which features contribute most to risk according to our best models.

5.5.1 Benchmark of learning methods

{benchmark}

Table 5.4 summarizes the performance of each learning configuration. The AGE,GENDER feature set provides a baseline for comparison, all other feature sets include these as well. As shown in the results, this two-dimensional representation already carries some information.

Based on Table 5.4 we can conclude that a patient's medication history is highly informative to predict the start of GLA therapy. Using features based on ATC level 5, the RESVM model obtained an AUC between 72.55% and 74.43%. By adding redundancy as described in Section 5.4.2 the performance based on medication history alone was further increased to between 74.34% and 76.27% for the best learning approach (RESVM).

Predictive performance based on provisions alone turned out fairly poor, showing only a mild improvement compared to models based exclusively on age and gender for all learning algorithms. Interestingly, the best approach for representations based on provisions was class-weighted SVM, with RESVM being worst of all three learning methods. It appears that for these representations, large training sets are more important than base model variability: class-weighted SVM uses the full training set, bagging SVM uses all positives and a

features	RESVM AUROC (%)	p	bagging SVM AUROC (%)	p	class-weighted SVM AUROC (%)	p
AGE, GENDER	55.74–56.64	$*$	58.61–59.67	$*$	**60.96–62.21**	0.04
ATC 5	**72.55–74.43**	0.17	70.83–72.62	0.09	71.89–73.74	0.01
ATC 1–4	**73.12–75.07**	0.07	69.57–71.24	$*$	73.05–74.91	0.04
ATC 1–5	**74.34–76.27**	0.13	71.50–73.27	0.05	72.13–73.94	$*$
PROVS FLAT	58.45–59.51	$*$	60.74–61.92	$*$	**63.01–64.31**	$*$
PROVS STRUCT	57.40–58.39	0.02	59.53–60.58	0.01	**62.53–63.81**	0.01
PROVS BOTH	58.89–59.75	$*$	61.72–62.87	$*$	**63.45–64.75**	$*$
ATC | PROVS	**74.89–76.82**	0.04	69.72–71.40	$*$	73.77–75.64	$*$

Table 5.4: Average bounds on area under the ROC curve and p-value of the Mann-Whitney U test over all folds for different feature sets per learning approach in a long-term prediction setup. The lower and upper bounds on AUC were computed with $\hat{\beta}_{lo} = 0.05$ and $\hat{\beta}_{up} = 0.10$, respectively. The ATC | PROVS feature set is the concatenation of the best performing sets per aspect, namely ATC 1–5 and PROVS BOTH. Stars ($*$) denote p-values below 0.005.

{table:results}

subset of unlabeled instances per base model and RESVM uses (small) subsets of both positives and unlabeled instances per base model.

The best representation included age, gender, and structured information about the drugs and provision history of each patient. The best learning method on this representation was RESVM, achieving an AUC between 74.89% and 76.82%. In Section 5.5.1 we compare the performance of our approach to competing screening methods.

Finally, RESVM appears most resistant to overfitting in the hyperparameter optimization stage as it consistently exhibits the highest average p-values in our diagnostic test (higher is better, see Section 5.4.1). We believe this to be attributable to the use of small resamples of both positives and unlabeled instances when training base models in RESVM, since this makes it unlikely to obtain a structural overfit of the ensemble model on the full training set. In contrast, bagging SVM is far more prone to overfitting because every base model is trained on all positives.

Comparison to State-of-the-art

{stateoftheart}

Our best approach obtained cross-validated AUC between 74.89% and 76.82% (exact numbers are unknown due to the lack of known negatives). This is comparable to many competing approaches, based on questionnaires and some

clinical information such as the Cambridge Risk Score (AUC 67%–80%, [117, 55]), the Danish risk score (AUC 72%–87.6%, [53]), the German diabetes risk score (AUC 75%–83%, [110]) and a Dutch approach (AUC 74%, [5]). Approaches using detailed clinical information generally perform better, but are more expensive to maintain [118, 91, 83, 63]. The key advantage of our approach is the fact it is easy to implement on a population wide scale at virtually no operational cost.

The target class we used in this work is stricter than in the risk prediction methods mentioned in Section 5.2, namely patients that require GLAs for glycemic control versus patients with impaired glycemic control, respectively (except for **(author?)** [83], which also predicted drug-treated T2D). It is reasonable to assume that our models generally rank patients with impaired glycemic control but without a need for GLAs higher than patients without impaired glycemic control. In our performance assessment both of these patient groups are essentially treated as negatives, in contrast to the screening programmes mentioned previously which treat patients with impaired glycemic control as positives. Hence, we believe the performance of our models would appear higher when evaluated against a target class comprising all patients with impaired glycemic control, as is done in the evaluation of other screening approaches. Unfortunately, we are unable to accurately identify patients with impaired glycemic control but without need for GLAs.

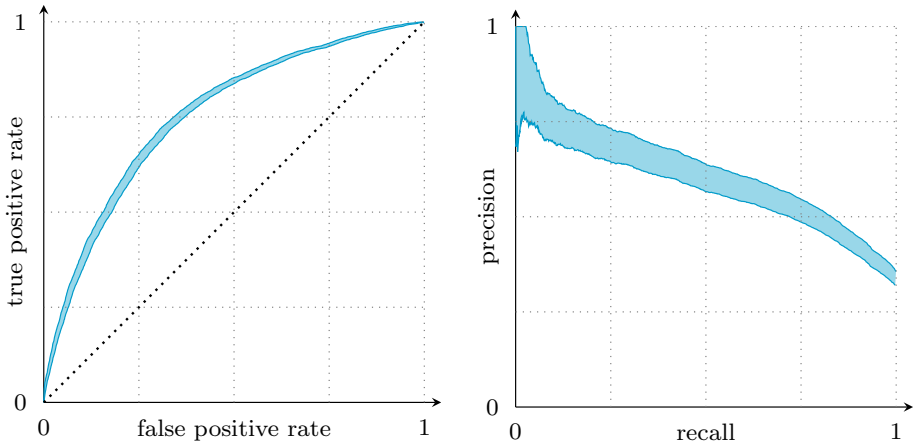
All competing methods use either clinical information or direct knowledge of risk factors that is unavailable to us. Furthermore, some of the characteristics that are lacking in our data have been reported to be the most informative to assess risk for T2D [83, 118, 91]. We obtained generalization performances that are comparable to existing approaches, despite these missing predictors. Finally, our approach is the only one that is based exclusively on existing data that is always available, without requiring additional patient contacts or clinical tests.

{resvmroc}

5.5.2 Performance Curves for RESVM

The RESVM model based on ATC | PROVS vectorization had the best overall performance. Figure 5.5 shows bounds on the ROC and PR curves for this model. These bounds were computed using the technique described by **(author?)** [27]. The true curve is unknown because we do not dispose of negative labels.

ROC curves enable us to determine a cutoff to use in practice, based on a suitable balance between true and false positive rate (sensitivity and 1-specificity, respectively). Determining a suitable balance requires a tradeoff between the relative importance of identifying undiagnosed patients (true positives) vis-à-vis increased amounts of screening tests on patients that are in fact healthy (false positives).



{fig:roc-diabetes} (a) Receiver Operating Characteristic curve. {fig:pr-diabetes} (b) Precision-Recall curve.

Figure 5.5: Performance curves for the best model: RESVM classifier based on ATC | PROVS vectorization. The lower and upper bounds are estimated using $\hat{\beta}_{lo} = 5\%$ and $\hat{\beta}_{up} = 10\%$, respectively.

betes-performance-curves}

It should be noted that precision depends on class balance, and therefore the PR curve shown in Figure 5.5b is not representative for screening an overall population, since the overall population has a higher fraction of negatives than our custom data set (i.e. precision would be lower in practice). In contrast, the bounds in ROC space are representative because ROC curves are insensitive to changes in class distribution [50].

5.5.3 Feature Importance Analysis for the RESVM Model

{features}

The RESVM model is implicitly nonlinear due to its majority voting rule to aggregate base model decisions, which poses problems in assessing the importance of each predictor. However, our use of linear base models enables a simple approximation. The decision value for base model $i \in \{1, \ldots, n_{\text{models}}\}$ for a test instance $\mathbf{z}$ can be written as follows:

$$f_i(\mathbf{z}) = \langle \mathbf{w}_i, \mathbf{z} \rangle + \rho_i = \mathbf{w}_i^T \mathbf{z} + \rho_i,$$

where $\mathbf{w}_i$ is the separating hyperplane and ρ_i is a bias term. A simple linear approximation of such ensemble models can be computed as the average of all

base model hyperplanes:

$$\bar{\mathbf{w}} = \sum_{i=1}^{n_{\text{models}}} \mathbf{w}_i / n_{\text{models}}.$$

Feature importance can then be determined based on the coefficients in $\bar{\mathbf{w}}$. The range of every feature is comparable, since we normalized all features to the unit interval $[0, 1]$. This allows us to conclude that the features with largest (positive) coefficients in $\bar{\mathbf{w}}$ contribute most to risk according to our model.

Via this approach, the risk associated to use of cardiovascular medication (ATC main category C) far outweights all other ATC main categories. This is not surprising, as diabetes is known to be strongly related to cardiovascular problems [72, 56, 66]. The relative importance of features and associated medical implications will be discussed in detail in a forthcoming medical paper.

5.6 Conclusion

In this work we have demonstrated the ability to predict clinical outcomes based solely on readily available health expenditure data. We successfully built proof-of-concept classifiers to predict the start of glucose-lowering pharmacotherapy in patients above 40. Our experiments show that accurate predictions can be made based on historical medication purchases. These predictions can be further improved by incorporating information about medical provisions and the use of appropriate vectorization schemes.

Since adult patients starting glucose-lowering pharmacotherapy are mainly afflicted with type 2 diabetes (T2D), our models can be used for T2D risk assessment. Our approach presents a novel method for case finding which can be easily incorporated in modern healthcare, since all required data is already available. The associated operational costs are very low as the entire workflow can be fully automated without any need for patient contacts or medical tests. As such, our work provides an efficient and cost-effective method to identify a high risk subgroup, which can then be screened using decisive clinical tests.

Interestingly, our approach works well even though health expenditure data contains very limited direct information on some important known risk factors. In that sense, our approach is fundamentally different from the current state-of-the-art which mainly focuses on quantifying known risk factors directly, either by asking the patient or through clinical tests. The performance of our approach is expected to improve further when additional information about these risk factors can be obtained, e.g. family history and lifestyle.

Chapter 6

This is conclusion

{ch:conclusion}

...

Instructies van de faculteit:

Algemene besluiten: Verwijzend naar de inleiding en naar de besluiten van de afzonderlijke hoofdstukken worden op het einde van het proefschrift de voornaamste besluiten gebundeld. Hier wordt de nadruk gelegd op de eigen inbreng, de verworven resultaten, de 'stellingen' van het proefschrift en de originele bijdragen tot het onderzoeksdomein. De onopgeloste problemen worden aangestipt en suggesties voor eventueel verder onderzoek worden gemaakt.

Appendix A

This is myappendix

{ch:myappendix}

...

Instructies van de faculteit:

De appendices: ze omvatten alle gedeelten uit de tekst die weliswaar essentieel zijn voor het proefschrift, maar waarvan de inlassing in de tekst de leesbaarheid ervan nadelig zouden beïnvloeden bv. omwille van hun lengte. Zo kunnen bv. de brute meetresultaten of een computerprogramma met zijn bron, commentaar en voorbeelden beter thuishoren in een appendix dan in de tekst zelf. De appendices kunnen desgevallend worden gebundeld in een apart boekdeel.

Bibliography

[1] Stein Aerts, Diether Lambrechts, Sunit Maity, Peter Van Loo, Bert Coessens, Frederik De Smet, Leon-Charles Tranchevent, Bart De Moor, Peter Marynen, Bassem Hassan, Peter Carmeliet, and Yves Moreau. Gene prioritization through genomic data fusion. *Nature Biotechnology*, 24(5):537–544, May 2006. pages 6

[2] KGMM Alberti, Mayer B Davidson, Ralph A DeFronzo, Allan Drash, Saul Genuth, Maureen I Harris, Richard Kahn, Harry Keen, William C Knowler, Harold Lebovitz, et al. Report of the expert committee on the diagnosis and classification of diabetes mellitus. *Diabetes Care*, 21:S5, 1998. pages 45

[3] Carlos Alzate and Johan A. K. Suykens. A semi-supervised formulation to binary kernel spectral clustering. In *2012 IEEE World Congress on Computational Intelligence (IEEE WCCI/IJCNN 2012)*, Brisbane, Australia, June 2012. pages 7

[4] American Diabetes Association et al. Standards of medical care in diabetes–2014. *Diabetes care*, 37(Supplement 1):S14–S80, 2014. pages 45, 50

[5] Caroline A Baan, Johannes B Ruige, Ronald P Stolk, JC Witteman, Jacqueline M Dekker, Robert J Heine, and EJ Feskens. Performance of a predictive model to identify undiagnosed diabetes in a health care setting. *Diabetes Care*, 22(2):213–219, 1999. pages 45, 46, 62

[6] François Bachoc. Cross validation and maximum likelihood estimations of hyper-parameters of gaussian processes with model misspecification. *Computational Statistics & Data Analysis*, 66:55–69, 2013. pages 41

[7] David J Ballard, Linda L Humphrey, L Joseph Melton, Peter P Frohnert, Chu-Pin Chu, W Michael O'Fallon, and Pasquale J Palumbo. Epidemiology of persistent proteinuria in type II diabetes mellitus: population-based study in Rochester, Minnesota. *Diabetes*, 37(4):405–412, 1988. pages 45

[8] Robert E Banfield, Lawrence O Hall, Kevin W Bowyer, and W Philip Kegelmeyer. A comparison of decision tree ensemble creation techniques. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(1):173–180, 2007. pages 12

[9] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1-2):105–139, 1999. pages 10, 11, 28

[10] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(1):281–305, 2012. pages 26, 38, 40, 41

[11] James Bergstra, Dan Yamins, and David D Cox. Hyperopt: A Python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in Science Conference*, pages 13–20. SciPy, 2013. pages 41

[12] James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *Proceedings of The 30th International Conference on Machine Learning*, pages 115–123, 2013. pages 40

[13] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011. pages 38, 41

[14] Mauro Birattari, Zhi Yuan, Prasanna Balaprakash, and Thomas Stützle. F-race and iterated f-race: An overview. In *Experimental methods for the analysis of optimization algorithms*, pages 311–336. Springer, 2010. pages 41

[15] Christopher M Bishop et al. Neural networks for pattern recognition. 1995. pages 38, 40, 41

[16] Jock A. Blackard and Denis J. Dean. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Computers and Electronics in Agriculture*, 24(3):131–151, December 1999. pages 17, 34

[17] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Semi-supervised novelty detection. *Journal of Machine Learning Research*, 11:2973–3009, 2010. pages 6

[18] Léon Bottou and Chih-Jen Lin. Support vector machine solvers. *Large scale kernel machines*, pages 301–320, 2007. pages 11, 40

[19] Léon Bottou and Chih-Jen Lin. Support Vector Machine Solvers. In Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston, editors, *Large Scale Kernel Machines*, pages 301–320, Cambridge, MA, USA, 2007. MIT Press. pages 32

[20] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, August 1996. pages 8, 10, 32

[21] Leo Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3):229–242, 2000. pages 10, 11

[22] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. pages 10, 38, 40, 41

[23] Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005. pages 10

[24] Gavin C Cawley. Leave-one-out cross-validation based model selection criteria for weighted LS-SVMs. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 1661–1668. IEEE, 2006. pages 8, 59

[25] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. pages 16, 33, 59

[26] G Charlone, L Torsten, C Bendix, et al. A Danish diabetes risk score for targeted screening. *Diabetes Care*, 27:727–733, 2004. pages 45

[27] Marc Claesen, Jesse Davis, Frank De Smet, and Bart De Moor. Assessing binary classifiers using only positive and unlabeled data. *arXiv preprint arXiv:1504.06837*, 2015. pages 53, 62

[28] Marc Claesen, Jaak Simm, Dusan Popovic, Yves Moreau, and Bart De Moor. Easy hyperparameter search using Optunity. *CoRR*, abs/1412.1114, 2014. `http://www.optunity.net`. pages 38, 39, 41

[29] Marc Claesen, Jaak Simm, Dusan Popovic, Yves Moreau, and Bart De Moor. Easy hyperparameter search using Optunity. *arXiv preprint arXiv:1412.1114*, 2014. pages 51, 52

[30] Marc Claesen, Frank De Smet, Johan A.K. Suykens, and Bart De Moor. EnsembleSVM: A library for ensemble learning using support vector machines. *Journal of Machine Learning Research*, 15:141–145, 2014. pages 16

[31] Marc Claesen, Frank De Smet, Johan A.K. Suykens, and Bart De Moor. EnsembleSVM: A library for ensemble learning using support vector machines. *Journal of Machine Learning Research*, 15:141–145, 2014. pages 38, 40, 41, 51, 59, 60

[32] Marc Claesen, Frank De Smet, Johan A.K. Suykens, and Bart De Moor. A robust ensemble approach to learn from positive and unlabeled data using SVM base models. *Neurocomputing*, 160(0):73 – 84, 2015. pages 50, 58, 59, 60

[33] Ronan Collobert, Samy Bengio, and Yoshua Bengio. A parallel mixture of SVMs for very large scale problems. *Neural Computation*, 14(5):1105–1114, 2002. pages 32, 35

[34] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995. pages 33

[35] Felipe Cucker and Steve Smale. Best choices for regularization parameters in learning theory: on the bias-variance problem. *Foundations of Computational Mathematics*, 2(4):413–428, 2002. pages 38

[36] Anneleen Daemen, Olivier Gevaert, Fabian Ojeda, Annelies Debucquoy, Johan A.K. Suykens, Christine Sempoux, Jean-Pascal Machiels, Karin Haustermans, and Bart De Moor. A kernel-based integration of genome-wide data for clinical decision support. *Genome Medicine*, 1(4):39, 2009. pages 8, 59

[37] Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 233–240, New York, NY, USA, 2006. ACM. pages 16, 22

[38] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, pages 1223–1231, 2012. pages 40

[39] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006. pages 16, 24, 25

[40] Diabetes Prevention Program Research Group et al. Reduction in the incidence of type 2 diabetes with lifestyle intervention or metformin. *The New England journal of medicine*, 346(6):393, 2002. pages 45

[41] Thomas G Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning*, 40(2):139–157, 2000. pages 10

[42] Marco F Duarte and Yu Hen Hu. Vehicle classification in distributed sensor networks. *Journal of Parallel and Distributed Computing*, 64(7):826–838, 2004. pages 17

[43] Bradley Efron and Gail Gong. A leisurely look at the bootstrap, the jackknife, and cross-validation. *The American Statistician*, 37(1):36–48, 1983. pages 39, 40

[44] Katharina Eggensperger, Matthias Feurer, Frank Hutter, James Bergstra, Jasper Snoek, Holger Hoos, and Kevin Leyton-Brown. Towards an empirical foundation for assessing Bayesian optimization of hyperparameters. In *NIPS workshop on Bayesian Optimization in Theory and Practice*, 2013. pages 38, 41

[45] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 213–220, New York, NY, USA, 2008. ACM. pages 6, 7, 50, 58

[46] Michael M Engelgau, KM Narayan, and William H Herman. Screening for type 2 diabetes. *Diabetes Care*, 23(10):1563–1580, 2000. pages 45

[47] Theodoros Evgeniou, Massimiliano Pontil, and Tomaso Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000. pages 38

[48] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. pages 16

[49] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, June 2008. pages 34

[50] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, June 2006. pages 63

[51] B. Frenay and M. Verleysen. Classification in the presence of label noise: A survey. *Neural Networks and Learning Systems, IEEE Transactions on*, 25(5):845–869, May 2014. pages 6

[52] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992. pages 38

[53] Charlotte Glümer, Bendix Carstensen, Annelli Sandbæk, Torsten Lauritzen, Torben Jørgensen, and Knut Borch-Johnsen. A Danish diabetes risk score for targeted screening – the Inter99 study. *Diabetes Care*, 27(3):727–733, 2004. pages 45, 46, 62

[54] Yves Grandvalet. Bagging equalizes influence. *Machine Learning*, 55(3):251–270, 2004. pages 10, 11

[55] SJ Griffin, PS Little, CN Hales, AL Kinmonth, and NJ Wareham. Diabetes risk score: towards earlier detection of type 2 diabetes in general practice. *Diabetes/metabolism research and reviews*, 16(3):164–171, 2000. pages 45, 46, 62

[56] Scott M Grundy, Ivor J Benjamin, Gregory L Burke, Alan Chait, Robert H Eckel, Barbara V Howard, William Mitch, Sidney C Smith, and James R Sowers. Diabetes and cardiovascular disease a statement for healthcare professionals from the American Heart Association. *Circulation*, 100(10):1134–1146, 1999. pages 64

[57] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, 1982. pages 52

[58] Maureen I Harris and Richard C Eastman. Early detection of undiagnosed diabetes mellitus: a US perspective. *Diabetes/metabolism research and reviews*, 16(4):230–236, 2000. pages 45

[59] Maureen I Harris, Katherine M Flegal, Catherine C Cowie, Mark S Eberhardt, David E Goldstein, Randie R Little, Hsiao-Mei Wiedmeyer, and Danita D Byrd-Holt. Prevalence of diabetes, impaired fasting glucose, and impaired glucose tolerance in US adults: the Third National Health and Nutrition Examination Survey, 1988–1994. *Diabetes Care*, 21(4):518–524, 1998. pages 45, 50

[60] Maureen I Harris, Ronald Klein, Catherine C Cowie, Michael Rowland, and Danita D Byrd-Holt. Is the risk of diabetic retinopathy greater in non-hispanic blacks and mexican americans than in non-hispanic whites with type 2 diabetes?: A US population study. *Diabetes Care*, 21(8):1230–1235, 1998. pages 45

[61] Maureen I Harris, Ronald Klein, Tim A Welborn, and Matthew W Knuiman. Onset of NIDDM occurs at least 4–7 yr before clinical diagnosis. *Diabetes Care*, 15(7):815–819, 1992. pages 45

[62] Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The entire regularization path for the support vector machine. In *Journal of Machine Learning Research*, pages 1391–1415, 2004. pages 38

[63] Kenneth E Heikes, David M Eddy, Bhakti Arondekar, and Leonard Schlessinger. Diabetes risk calculator a simple tool for detecting undiagnosed diabetes and pre-diabetes. *Diabetes Care*, 31(5):1040–1045, 2008. pages 45, 46, 62

[64] Geoffrey E Hinton. A practical guide to training restricted boltzmann machines. In *Neural Networks: Tricks of the Trade*, pages 599–619. Springer, 2012. pages 38

[65] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification, 2003. pages 38

[66] Frank B Hu, Meir J Stampfer, Steven M Haffner, Caren G Solomon, Walter C Willett, and JoAnn E Manson. Elevated risk of cardiovascular disease prior to clinical diagnosis of type 2 diabetes. *Diabetes Care*, 25(7):1129–1134, 2002. pages 45, 64

[67] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, pages 507–523. Springer, 2011. pages 41

[68] Frank Hutter, Holger H Hoos, Kevin Leyton-Brown, and Thomas Stützle. ParamILS: an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36(1):267–306, 2009. pages 40, 41

[69] InterAct Consortium et al. The link between family history and risk of type 2 diabetes is not explained by anthropometric, lifestyle or genetic risk factors: the EPIC-InterAct study. *Diabetologia*, 56(1):60–69, 2013. pages 45

[70] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998. pages 41

[71] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001. [Online; accessed 2015-04-16]. pages 51

[72] WB Kannel and DL McGee. Diabetes and cardiovascular disease: The Framingham study. *JAMA*, 241(19):2035–2038, 1979. pages 64

[73] Maarten Keijzer and Vladan Babovic. Genetic programming, ensemble methods and the bias/variance tradeoff – introductory investigations. In

Riccardo Poli, Wolfgang Banzhaf, William B. Langdon, Julian Miller, Peter Nordin, and Terence C. Fogarty, editors, *Genetic Programming*, volume 1802 of *Lecture Notes in Computer Science*, pages 76–90. Springer Berlin Heidelberg, 2000. pages 28

[74] Hilary King, Ronald E Aubert, and William H Herman. Global burden of diabetes, 1995–2025: prevalence, numerical estimates, and projections. *Diabetes Care*, 21(9):1414–1431, 1998. pages 45, 50

[75] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence*, volume 14, pages 1137–1145, 1995. pages 39, 40

[76] Eva M Kohner, Stephen J Aldington, Irene M Stratton, Susan E Manley, Rury R Holman, David R Matthews, and Robert C Turner. United Kingdom Prospective Diabetes Study, 30: diabetic retinopathy at diagnosis of non–insulin-dependent diabetes mellitus and associated risk factors. *Archives of Ophthalmology*, 116(3):297–303, 1998. pages 45

[77] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. pages 40

[78] Ar Lazarevic, Aysel Ozgur, Levent Ertoz, Jaideep Srivastava, and Vipin Kumar. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the Third SIAM International Conference on Data Mining*, 2003. pages 6

[79] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. pages 17

[80] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pages 448–455, 2003. pages 7, 16, 50, 58

[81] Xiaoli Li and Bing Liu. Learning to classify texts using positive and unlabeled data. In *IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence*, pages 587–592, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc. pages 7

[82] Shih-Wei Lin, Kuo-Ching Ying, Shih-Chieh Chen, and Zne-Jung Lee. Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert systems with applications*, 35(4):1817–1824, 2008. pages 41

[83] Jaana Lindström and Jaakko Tuomilehto. The diabetes risk score a practical tool to predict type 2 diabetes risk. *Diabetes Care*, 26(3):725–731, 2003. pages 45, 46, 62

[84] Bin Linghu and Bing-Yu Sun. Constructing effective SVM ensembles for image classification. In *Knowledge Acquisition and Modeling (KAM), 2010 3rd International Symposium on*, pages 80–83, 2010. pages 32

[85] Nikolas List and Hans Ulrich Simon. SVM-optimization and steepest-descent line search. In *Proceedings of the 22nd Annual Conference on Computational Learning Theory*, 2009. pages 32

[86] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. Building text classifiers using positive and unlabeled examples. In *Proceedings of the Third IEEE International Conference on Data Mining*, ICDM '03, pages 179–186, Washington, DC, USA, 2003. IEEE Computer Society. pages 6, 7, 58

[87] Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. Partially supervised classification of text documents. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 387–394, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. pages 7, 58

[88] Zhigang Liu, Wenzhong Shi, Deren Li, and Qianqing Qin. Partially supervised classification – based on weighted unlabeled samples support vector machine. In *Proceedings of the First international conference on Advanced Data Mining and Applications*, ADMA'05, pages 118–129, Berlin, Heidelberg, 2005. Springer-Verlag. pages 7

[89] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947. pages 53

[90] Gonzalo Martínez-Muñoz and Alberto Suárez. Out-of-bag estimation of the optimal sample size in bagging. *Pattern Recognition*, 43(1):143–152, 2010. pages 12

[91] Marguerite J McNeely, Edward J Boyko, Donna L Leonetti, Steven E Kahn, and Wilfred Y Fujimoto. Comparison of a clinical model, the oral glucose tolerance test, and fasting glucose for prediction of type 2 diabetes risk in Japanese Americans. *Diabetes Care*, 26(3):758–763, 2003. pages 45, 46, 62

[92] Michael Meissner, Michael Schmuker, and Gisbert Schneider. Optimized particle swarm optimization (OPSO) and its application to artificial neural network training. *BMC bioinformatics*, 7(1):125, 2006. pages 41

[93] Ali H Mokdad, Earl S Ford, Barbara A Bowman, William H Dietz, Frank Vinicor, Virginia S Bales, and James S Marks. Prevalence of obesity, diabetes, and obesity-related health risk factors, 2001. *JAMA*, 289(1):76–79, 2003. pages 45

[94] Fantine Mordelet and Jean-Philippe Vert. A bagging SVM to learn from positive and unlabeled examples. *arXiv preprint arXiv:1010.0772*, 2010. pages 6, 7, 8, 9, 28, 50, 58, 59

[95] Fantine Mordelet and Jean-Philippe Vert. ProDiGe: Prioritization of disease genes with multitask machine learning from positive and unlabeled examples. *BMC bioinformatics*, 12(1):389, 2011. pages 6

[96] Fantine Mordelet and Jean-Philippe P. Vert. ProDiGe: Prioritization Of Disease Genes with multitask machine learning from positive and unlabeled examples. *BMC bioinformatics*, 12(1):389+, 2011. pages 32

[97] Peter Nemenyi. Distribution-free multiple comparisons. In *BIOMETRICS*, volume 18, page 263. INTERNATIONAL BIOMETRIC SOC 1441 I ST, NW, SUITE 700, WASHINGTON, DC 20005-2210, 1962. pages 25

[98] Edgar Osuna, Robert Freund, and Federico Girosi. Support Vector Machines: Training and Applications. Technical Report AIM-1602, 1997. pages 8, 33

[99] Xiao-Ren Pan, Guang-wei Li, Ying-Hua Hu, Ji-Xing Wang, Wen-Ying Yang, Zuo-Xin An, Ze-Xi Hu, Jina-Zhong Xiao, Hui-Bi Cao, Ping-An Liu, et al. Effects of diet and exercise in preventing NIDDM in people with impaired glucose tolerance: the Da Qing IGT and Diabetes Study. *Diabetes Care*, 20(4):537–544, 1997. pages 45

[100] PJ Park, SJ Griffin, L Sargeant, and NJ Wareham. The performance of a risk score in predicting undiagnosed hyperglycemia. *Diabetes Care*, 25(6):984–988, 2002. pages 46

[101] Stephen G Pauker. Deciding about screening. *Annals of internal medicine*, 118(11):901–902, 1993. pages 45

[102] Mykola Pechenizkiy, Alexey Tsymbal, Seppo Puuronen, and Oleksandr Pechenizkiy. Class noise and supervised learning in medical domains: The effect of feature extraction. In *Computer-Based Medical Systems, 2006. CBMS 2006. 19th IEEE International Symposium on*, pages 708–713. IEEE, 2006. pages 6

[103] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer,

Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. pages 38, 41, 51, 59

[104] Danil Prokhorov. IJCNN 2001 neural network competition. *Slide presentation in IJCNN*, 2001. pages 17, 35

[105] Ulla Rajala, Mauri Laakso, Qing Qiao, and Sirkka Keinänen-Kiukaanniemi. Prevalence of retinopathy in people with diabetes, impaired glucose tolerance, and normal glucose tolerance. *Diabetes Care*, 21(10):1664–1669, 1998. pages 45

[106] J Sunil Rao and Robert Tibshirani. The out-of-bootstrap method for model averaging and selection. *University of Toronto*, 1997. pages 10

[107] Jared P Reis, Catherine M Loria, Paul D Sorlie, Yikyung Park, Albert Hollenbeck, and Arthur Schatzkin. Lifestyle factors and risk for new-onset diabetes: a population-based cohort study. *Annals of internal medicine*, 155(5):292–299, 2011. pages 45

[108] Robert J Rubin, William M Altman, and Daniel N Mendelson. Health care expenditures for people with diabetes mellitus, 1992. *The Journal of Clinical Endocrinology & Metabolism*, 78(4):809A–809F, 1994. pages 45

[109] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002. pages 38, 39, 41

[110] Matthias B Schulze, Kurt Hoffmann, Heiner Boeing, Jakob Linseisen, Sabine Rohrmann, Matthias Möhlig, Andreas FH Pfeiffer, Joachim Spranger, Claus Thamer, Hans-Ulrich Häring, et al. An accurate risk score based on anthropometric, dietary, and lifestyle factors to predict the development of type 2 diabetes. *Diabetes Care*, 30(3):510–515, 2007. pages 45, 46, 62

[111] Peter EH Schwarz, Jiang Li, Manja Reimann, Alta E Schutte, Antje Bergmann, Markolf Hanefeld, Stefan R Bornstein, Jan Schulze, Jaakko Tuomilehto, and Jaana Lindstrom. The Finnish Diabetes Risk Score is associated with insulin resistance and progression towards type 2 diabetes. *The Journal of Clinical Endocrinology & Metabolism*, 94(3):920–926, 2009. pages 45

[112] Holbrooke S Seltzer. Drug-induced hypoglycemia. a review of 1418 cases. *Endocrinology and metabolism clinics of North America*, 18(1):163–183, 1989. pages 45

[113] Iris Shai, Rui Jiang, JoAnn E Manson, Meir J Stampfer, Walter C Willett, Graham A Colditz, and Frank B Hu. Ethnicity, obesity, and risk of type 2 diabetes in women a 20-year follow-up study. *Diabetes care*, 29(7):1585–1590, 2006. pages 45

[114] Brian K. Shoichet. Virtual screening of chemical libraries. *Nature*, 432(7019):862–865, December 2004. pages 6

[115] Alejandro Sifrim, Dusan Popovic, Léon-Charles Tranchevent, Amin Arderschirdavani, Ryo Sakai, Peter Konings, Joris Vermeesch, Jan Aerts, Bart De Moor, and Yves Moreau. eXtasy: Variant prioritization by genomic data fusion. *Nature Methods*, 10:1083–1084, 2013. pages 6

[116] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pages 2951–2959, 2012. pages 41

[117] Annemieke MW Spijkerman, Matthew F Yuyun, Simon J Griffin, Jacqueline M Dekker, Giel Nijpels, and Nicholas J Wareham. The performance of a risk score as a screening test for undiagnosed hyperglycemia in ethnic minority groups data from the 1999 health survey for England. *Diabetes Care*, 27(1):116–122, 2004. pages 45, 46, 62

[118] Michael P Stern, Ken Williams, and Steven M Haffner. Identification of persons at high risk for type 2 diabetes mellitus: do we need the oral glucose tolerance test? *Annals of Internal Medicine*, 136(8):575–581, 2002. pages 45, 46, 62

[119] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014. pages 40

[120] Jinn-Tsong Tsai, Jyh-Horng Chou, and Tung-Kuan Liu. Tuning the structure and parameters of a neural network by using hybrid taguchi-genetic algorithm. *Neural Networks, IEEE Transactions on*, 17(1):69–80, 2006. pages 41

[121] Jaakko Tuomilehto, Jaana Lindström, Johan G Eriksson, Timo T Valle, Helena Hämäläinen, Pirjo Ilanne-Parikka, Sirkka Keinänen-Kiukaanniemi, Mauri Laakso, Anne Louheranta, Merja Rastas, et al. Prevention of type 2 diabetes mellitus by changes in lifestyle among subjects with impaired glucose tolerance. *New England Journal of Medicine*, 344(18):1343–1350, 2001. pages 45

[122] Robert C Turner, Carole A Cull, Valeria Frighi, Rury R Holman, UK Prospective Diabetes Study (UKPDS) Group, et al. Glycemic control

with diet, sulfonylurea, metformin, or insulin in patients with type 2 diabetes mellitus: progressive requirement for multiple therapies (UKPDS 49). *JAMA*, 281(21):2005–2012, 1999. pages 45

[123] Giorgio Valentini and Thomas G Dietterich. Low bias bagged support vector machines. In *ICML*, pages 752–759, 2003. pages 32, 35

[124] Greet Van den Berghe, Alexander Wilmer, Greet Hermans, Wouter Meersseman, Pieter J Wouters, Ilse Milants, Eric Van Wijngaerden, Herman Bobbaers, and Roger Bouillon. Intensive insulin therapy in the medical ICU. *New England Journal of Medicine*, 354(5):449, 2006. pages 54

[125] Greet Van den Berghe, Pieter Wouters, Frank Weekers, Charles Verwaest, Frans Bruyninckx, Miet Schetz, Dirk Vlasselaers, Patrick Ferdinande, Peter Lauwers, and Roger Bouillon. Intensive insulin therapy in critically ill patients. *New England journal of medicine*, 345(19):1359–1367, 2001. pages 54

[126] R Van den Oever and C Volckaert. Financing health care in Belgium. the nomenclature: from fee-for-service to budget-financing. *Acta chirurgica Belgica*, 108(2):157, 2008. pages 48

[127] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011. pages 51

[128] Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, 7(1):91, 2006. pages 51

[129] Shi-jin Wang, Avin Mathew, Yan Chen, Li-feng Xi, Lin Ma, and Jay Lee. Empirical analysis of support vector machine ensemble classifiers. *Expert Systems with Applications*, 36(3, Part 2):6466 – 6476, 2009. pages 32, 35

[130] Nicholas J Wareham and Simon J Griffin. Should we screen for type 2 diabetes? evaluation against national screening committee criteria. *BMJ: British Medical Journal*, 322(7292):986, 2001. pages 45

[131] WHO Collaborating Centre for Drug Statistics Methodology. *Guidelines for ATC classification and DDD assignment*. World Health Organization, 2015. pages 47

[132] World Health Organization et al. Prevention of diabetes mellitus: report of a WHO study group [meeting held in geneva from 16 to 20 november 1992]. (WHO technical report number 844), 1994. pages 45, 49

[133] World Health Organization et al. International classification of diseases (ICD). 2012. pages 49

[134] Samuel Xavier-de Souza, Johan AK Suykens, Joos Vandewalle, and Désiré Bollé. Coupled simulated annealing. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 40(2):320–335, 2010. pages 41

[135] Hwanjo Yu. Single-class classification with mapping convergence. *Machine Learning*, 61(1-3):49–69, November 2005. pages 7, 58

[136] Hwanjo Yu, Jiawei Han, and Kevin C. Chang. PEBL: positive example based learning for web page classification using SVM. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–248, New York, NY, USA, 2002. ACM Press. pages 6

[137] Nicola N Zammitt and Brian M Frier. Hypoglycemia in type 2 diabetes pathophysiology, frequency, and effects of different treatment modalities. *Diabetes Care*, 28(12):2948–2961, 2005. pages 45

[138] Xingquan Zhu and Xindong Wu. Class noise vs. attribute noise: A quantitative study. *Artificial Intelligence Review*, 22(3):177–210, 2004. pages 6

[139] Paul Zimmet, KGMM Alberti, and Jonathan Shaw. Global and societal implications of the diabetes epidemic. *Nature*, 414(6865):782–787, 2001. pages 45

This is curriculum

{ch:curriculum}

...

Instructies van de faculteit:

Beknopt CV van de doctorandus.

List of publications

Input file chapters/publications/publications.tex does not exist. Make sure its starts with “\chapter{List of publications}”. To not include this chapter in the table of contents, use the starred version of the \chapter command...

Instructies van de faculteit:

Lijst van de publicaties door de doctorandus/a (auteur of co-auteur).

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
IMINDS-STADIUS
Kasteelpark Arenberg 10, bus 2446
B-3001 Leuven
marc.claesen@esat.kuleuven.be
http://esat.kuleuven.be/stadius