Contents

1 Application to Model Selection for Primal SVM

Skalarprodukt anpassen, Vektoren nicht fett oder neue definition, notation, $\lambda \in \Lambda$ einfugen
Vektoren (in gesamter Arbeit) in runden Klammern

1.1 Introduction

In this part of the thesis the nonconvex inexact bundle algorithm number in thesis is applied to the problem of model selection for *support vector machines* (SVMs) solving classification tasks. It relies on a bilevel formulation proposed by Kunapuli in [5] and Moore et al. in [8].

A natural application for the inexact bundle algorithm is an optimization problem where the objective function value and subgradient can only be computed by numerical approximation. This is for example the case in bilevel optimization.

A general bilevel program can be formulated as in [5, p. 20]

$$\begin{aligned} \min_{x \in X, y \in \mathbb{R}^k} \quad & F(x,y) && \text{upper level} \\ \text{s.t.} \quad & G(x,y) \leq 0 \\ & y \in \left\{ \begin{array}{ll} \arg\max\limits_{y \in Y} & f(x,y) \\ \text{s.t.} & g(x,y) \leq 0 \end{array} \right\}. && \text{lower level} \end{aligned} \tag{1.1}$$

The two objecective functions F and f map from $\mathbb{R}^n \times \mathbb{R}^k$ into $\mathbb{R}$ and the constraint functions G and g map from $\mathbb{R}^n \times \mathbb{R}^k$ into $\mathbb{R}^L$ and $\mathbb{R}^l$ respectively.

The problem consists of an *upper* or *outer level* which is the overall function to be optimized. Contrary to usual constrained optimization problems which are constrained by explicitly given equalities and inequalities a bilevel program is additionally constrained by a second optimization problem, the *lower* or *inner level* problem.

Solving bilevel problems can be divided roughly in two classes: implicit and explicit solution methods. In the explicit methods the lower level problem is usually rewritten by its KKT conditions, these are then added as constraints to the upper level problem. With this solution method the upper and lower level are solved simultaneously. For the setting of model selection for support vector machines as it is used here, this method is

described in detail in [5].

The second approach is the implicit one. Here the lower level problem is solved directly in every iteration of the outer optimization algorithm and the solution is plugged into the upper level objective.

Obviously if the inner level problem is solved numerically, the solution cannot be exact. Additionally the *solution map* $S(x) = \{y \in \mathbb{R}^k \mid y$, that solves the lower level problem, is can be nondifferentiable [9] and since elements of the solution map are plugged into the outer level objective function in the implicit approach, the outer level function then becomes nonsmooth itself.This is why the inexact bundle algorithm seems a natural choice to tackle these bilevel problems.

Moore et al. use the implicit approach in [8] for support vector regression. However they use a gradient decent method which is not guaranteed to stop at an optimal solution. In [7] he also suggests the nonconvex exact bundle algorithm of Fuduli et al. [2] for solving the bilevel regression problem. This allows for nonsmooth inner problems and can theoretically solve some of the issues of the gradient descent method. It ignores however, that the objective function values can only be calculated approximately. A fact which is not addressed in Fuduli's algorithm.

1.2 Notation

As short remark on the notation in this chapter is required.

Due to standard notation in the field of SVM, the variables x and y are used in this chapter in two different contexts. In the setting of bilevel problems as used in (1.1) $x \in X \subset \mathbb{R}^n$ and $y \in \mathbb{R}^k$ are the optimization variables of the bilevel problem.

In the setting of SVMs $x^i \in \mathbb{R}^{feat}$ is an element of the *feature space* that contains the values of one data point. The corresponding variable $y_i \in \{-1, 1\}$ of the *output domain* contains the class in the data point x^i lies. Sometimes the indices of the samples are omitted to express the general dependency on the data (x,y).

During the most part of the chapter, x^i and y^i are used as in the context of SVMs. In the cases where they are used to describe the variables of an optimization problem, this is clearly indicated.

There appear also different derivatives in this chapter. For a continuously differentiable function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ we denote the *gradient* at $(\bar{x}, \bar{y})$ by $\nabla f(\bar{x}, \bar{y}) \in \mathbb{R}^{n+m}$. The partial gradient with respect to x is indicated by $\nabla_y f(\bar{x}, \bar{y}) \in \mathbb{R}^m$. For a continuously

differentiable vector valued function $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^k$ the *Jacobian matrix* is given by $\mathcal{J}F(\bar{x}, \bar{y}) \in \mathbb{R}^{k \times n+m}$. The partial Jacobian with respect to the variable y is denoted $\mathcal{J}_y F(\bar{x}, \bar{y}) \in \mathbb{R}^{k \times m}$.

Finally let $M \in \mathbb{R}^{n \times m}$ be a matrix and $I \subset \{1, ..., n\}$ an index set. Then the expression M_I denotes the matrix that consists of the rows of M corresponding to the indices in the set I.

1.3 Introduction to Support Vector Machines

Support vector machines are linear learning machines that were developed in the 1990's by Vapnik and co-workers. Soon they could outperform several other programs in this area [1] and the subsequent interest in SVMs lead to a very versatile application of these machines [5].

The case that is considered here is binary support vector classification using supervised learning. For a throughout introduction to this subject see also [1]. Here a summary of the most important expressions and results is given.

In classification data from a possibly high dimensional vector space $\tilde{X} \subset \mathbb{R}^{feat}$, the feature or *input space* is divided into two classes. These lie in the output domain $\tilde{Y} = \{-1, 1\}$. Elements from the feature space will mostly be called *data points* here. They get *labels* from the feature space. Labeled data points are called *examples*. The functional relation between the features and the class of an example is given by the usually unknown *response* or *target function* $f(x)$. Supervised learning is a kind of machine learning task where the machine is given examples of input data with associated labels, the so called *training data* (X, Y). Mathematically this can be modeled by assuming that the examples are drawn identically and independently distributed (iid) from the fixed joint distribution $P(x, y)$. This usually unknown distribution states the probability that a data point x has the label y [12, p. 988]. The overall goal is then to optimize the generalization ability, meaning the ability to predict the output for unseen data correctly [1, chapter 1.2].

1.3.1 Risk minimization

The concept of SVM's was originally inspired by the statistical learning theory developed by Vapnik. A detailed examination of the subject is given in [11]. In [13] the subject is approached from a more explaining point of view.

The idea of *risk minimization* is to find from a fixed set or class of functions the one that

is the best approximation to the response function. This is done by minimizing a loss function that compares the given labels of the examples to the response of the learning machine.

As the response function is not known only the expected value of the loss can be calculated. It is given by the *risk functional*

$$R(\lambda) = \int \mathcal{L}(y, f_\lambda(x)) \mathrm{d}P(x, y). \tag{1.2}$$

Here $\mathcal{L} : \mathbb{R}^2 \to \mathbb{R}$ is the loss function, $f_\lambda : \mathbb{R}^n \cap \mathcal{F} \to \mathbb{R},\ \lambda \in \Lambda$ the approximate response function found by the learning machine and $P(x, y)$ the joint distribution the training data is drawn from. The goal is now to find a function $f_{\bar{\lambda}}(x)$ in the chosen function space $\mathcal{F}$ that minimizes this risk functional [12, 989].

As the only given information is provided by the training set inductive principles are used to work with the *empirical risk*, rather than with the risk functional. The empirical risk only depends on the finite training set and is given by

$$R_{\mathrm{emp}}(\lambda) = \frac{1}{l} \sum_{i=1}^{l} \mathcal{L}(y_i, f_\lambda(x^i)), \tag{1.3}$$

where l is the number of data points. The law of large numbers ensures that the empirical risk converges to the risk functional as the number of data points grows to infinity. This however does not guarantee that the function $f_{\lambda,\mathrm{emp}}$ that minimizes the empirical risk also converges towards the function $f_{\bar{\lambda}}$ that minimizes the risk functional. The theory of consistency provides necessary and sufficient conditions that solve this issue [12, p. 989].

Vapnik therefore introduced the structural risk minimization (SRM) induction principle . It ensures that the used set of functions has a structure that makes it strongly consistent [12]. Additionally it takes the complexity of the function that is used to approximate the target function into account. "The SRM principle actually suggests a tradeoff between the quality of the approximation and the complexity of the approximating function" [12, p. 994]. This reduces the risk of *overfitting*, meaning to overly fit the function to the training data with the result of poor generalization [1, chapter 1.3].

Support vector machines fulfill all conditions of the SRM principle. Due to the kernel trick that allows for nonlinear classification tasks it is also very powerful. For more detailed information on this see [5] and references therein.

1.3.2 Support Vector Machines

In the case of linear binary classification one searches for a an affine hyperplane $w \in \mathbb{R}^f eat$ shifted by $b \in \mathbb{R}$ to separate the given data. The vector w is called weight vector and b is the bias.

Let the data be linearly separable. The function deciding how the data is classified can then be written as

$$f(x) = \text{sign}(\langle w, x \rangle - b).$$

Support vector machines aim at finding such a hyperplane that separates also unseen data optimally.

???Picture of hyperplane

One problem of this intuitive approach is that the representation of a hyperplane is not unique. If the plane described by (w, b) separates the data, there exist infinitely many hyperplanes (tw, b), $t > 0$, that separate the data in the same way. To have a unique description of a separating hyperplane the *canonical hyperplane for given data* $x \in X$ is defined by

$$f(x) = \langle w, x \rangle - b \quad \text{s.t.} \quad \min_i |\left\langle w, x^i \right\rangle - b| = 1.$$

This is always possible in the case where the data is linearly separable and means that the inverse of the norm of the weight vector is equal to the distance of the closest point $x \in X$ to the hyperplane [5, p. 10].

This gives rise to the following definition: The *margin* is the minimal Euclidean distance between a training example x^i and the separating hyperplane. A bigger margin means a lower complexity of the function [1].

A *maximal margin hyperplane* is the hyperplane that realizes the maximal possible margin for a given data set.

Proposition 1.1 ([1, Proposition 6.1]) Given a linearly separable training sample $\Omega = \{(x^i, y_i), ..., (x^l, y_l)\}$ the hyperplane (w, b) that solves the optimization problem

$$\|w\|^2 \quad \text{s.t.} \quad y_i(\langle w, x \rangle - b) \geq 1, \quad i = 1, ..., l,$$

realizes a maximal margin hyperplane.

The proof is given in [1, chapter 6.1].

Generally one cannot assume the data to be linearly separable. This is why in most applications a so called *soft margin classifier* is used. It introduces the slack variables ξ_i that measure the distance of the misclassified points to the hyperplane:

Fix $\gamma > 0$. A *margin slack variable of the example* (x^i, y_i) with respect to the hyperplane (w, b) and target margin γ is

$$\xi_i = \max(0, \gamma - y_i(\langle w, x \rangle + b))$$

If $\xi_i > \gamma$ the point is considered misclassified. One can also say that $\|\xi\|$ measures the amount by which training set "fails to have a margin of γ" [1, section 2.1.1].

For support vector machines the target margin is set to $\gamma = 1$.

This results in the following optimization problem for finding an optimal separating hyperplane (w, b):

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|w\|_2^2 + \frac{C}{2}\sum_{i=1}^{nd} \xi_i^2 \\ \text{s.t.} \quad & y_i \left(\left\langle w, x^i \right\rangle - b \right) \geq 1 - \xi_i \\ & \forall i = 1, \ldots, nd. \end{aligned} \tag{1.4}$$

The first part of the objective function is the regularization, the second part the actual loss function. The parameter $C > 0$ gives a trade-off between the richness of the chosen set of functions f_λ to reduce the error on the training data and the danger of overfitting to have good generalization. It has to be chosen a priori [5].

Instead of the Euclidean norm it is also possible to use the 1-norm in the loss function. Then the resulting optimization problem reads

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{nd}\xi_i \\ \text{s.t.} \quad & y_i\left(\left\langle w, x^i\right\rangle - b\right) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & \forall i = 1, \ldots, nd. \end{aligned}$$

In this form, however, the problem does not fit the theory described in [9], which is used later to solve the constructed bilevel problem. refer to appendix were counterexample to strong regularity is given?
It is shown in appendix ... that there exist data sets for which the strong regularity condition stated in ... is violated. Thus the subgradient of the upper level objective function cannot be calculated as proposed in [9].

In order to fulfill all conditions assumed in [9] it is necessary to reformulate problem (1.4). The new formulation makes use of the *implicit bias*, meaning that the bias is not calculated separately but as part of the variable w. By adding an additional one to the end of each feature vector we can use the vectors $\tilde{x}_i := (x_i, 1)^\top$ and $\tilde{w} := (w, w_b)$ to state the following optimization problem:

$$\begin{aligned} \min_{\tilde{w},\xi} \quad & \frac{1}{2}\|\tilde{w}\|^2 + \frac{C}{2}\sum_{i=1}^{nd}\xi_i^2 \\ \text{s.t.} \quad & y_i\left\langle \tilde{w}, \tilde{x}^i\right\rangle = y_i\left(\left\langle w, x^i\right\rangle - w_b\right) \geq 1 - \xi_i \\ & \forall i = 1, \ldots, nd. \end{aligned} \tag{1.5}$$

Not treating the bias separately is a strategy used for example to achieve a more efficient implementation [3, section 3.2, p. 22]. In the course of this section the gain of this particular formulation is the fact that it has a unique solution compare for... where it is shown. It is however shown in [3, section 3.2, p. 22] that the solutions that are found with explicit and implicit bias are different. This results from the fact that in case of implicit bias the variable b also enters the regularization term.

From now on we work with the implicit bias. To see the derivation of the bilevel problem with the explicit bias compare for [5, section 2.2].

1.3.3 Multiple Hyper-parameters

To examine the performance of the bilevel approach in the more dimensional case a model suggested by Moore et al. in [8] and called *multi-group* support vector classification (multiSVC) is used. There different hyper-parameters are allowed for different subgroups of the trainings data. In section 4.3 of [8] the model is described for a regression function. In this thesis the same technique is used for classification.

The motivation behind the approach is that different groups of samples from the training set can have slightly different properties and should therefore have their own weighting parameters C_g. On the one hand this can improve the generalization results. On the other hand properties of the different data groups can be identified by their respective hyper-parameters. Moore et al. explain in [8, section 4.3, p. 9] that for example a large C_g signifies reliable data in the respective group whereas a smaller C_g suggests a poorer quality.

To perform multiSVC divide the trainings data into G (pairwise disjoint) groups. Define the vector of hyper-parameters $C := [C_1, ..., C_G]^\top$. For the sake simplicity in this thesis all the groups are of equal size but all derivations are also possible for differently sized groups.

The multi-group classification problem in constrained form can be stated as

$$\begin{aligned} \min_{\tilde{w},\xi} \quad & \frac{1}{2}\|\tilde{w}\|^2 + \sum_{g=1}^{G}\left(\frac{C_g}{2}\sum_{i\in n_g}\xi_i^2\right) \\ \text{s.t.} \quad & y_i\left\langle \tilde{w}, \tilde{x}^i\right\rangle \geq 1-\xi_i \\ & \forall i \in \bigcup_{g=1}^{G} n_g = \{1, ..., nd\}. \end{aligned} \tag{1.6}$$

Here n_g is the index set that contains the indices of the data of the g'th group.

1.4 Formulation of the Bilevel Problem

The hyper-parameters C_g in the objective function of the classification problem have to be set beforehand. This step is part of the model selection process. To set the parameters optimally different methods can be used. A very intuitive and widely used approach is doing *cross validation* (CV) with a grid search implementation [5, p. 30].

To prevent overfitting and get a good parameter selection, especially in case of little data, commonly T-fold cross validation is used [5, p. 30]. For this technique the training data is randomly partitioned into T subsets of equal size. One of these subsets is then left out of the training set and instead used afterwards to get an estimate of the generalization error. To use CV for model selection it has to be embedded into an optimization algorithm over the hyper-parameter space. Commonly this is done by discretizing the parameter space and for T-fold CV training T models at each grid point. The resulting models are then compared to find the best parameters in the grid. Obviously for a growing number of hyper-parameters this is very costly. An additional drawback is that the parameters are only chosen from a finite set [5, p. 30].

A more recent approach is the formulation as a bilevel problem used in [5] and [8]. This makes it possible to optimize the hyper-parameters continuously.

Let $\Omega = \{(x^1, y_1), ..., (x^{nd}, y_{nd})\} \subset \mathbb{R}^{feat+1}$ be a given data set of size $nd = |\Omega|$. The associated index set is denoted by $\mathcal{N}$. For classification the labels y_i are ± 1. For T-fold cross validation let $\bar{\Omega}_t$ and Ω_t be the training set and the validation set respectively within the t'th fold and $\bar{\mathcal{N}}_t$ and $\mathcal{N}_t$ the respective index sets. For multi-group SVC the index set $\bar{\mathcal{N}}_t$ is again divided into the G sets $\bar{\mathcal{N}}_t^g$ to account for the different groups associated with the different hyper-parameters. Furthermore let $f^t : \mathbb{R}^{feat+1} \cap \mathcal{F} \to \mathbb{R}$ be the response function trained on the t'th fold and $\lambda \in \Lambda$ the hyper-parameter to be optimized. For a general machine learning problem with upper and lower loss function $\mathcal{L}_{upp}$ and $\mathcal{L}_{low}$ respectively the bilevel problem reads

$$
\begin{aligned}
\min_{\lambda, f^t} \quad & \mathcal{L}_{upp}\left(\lambda, f^1|_{\Omega_1}, ..., f^T|_{\Omega_T}\right) && \text{upper level} \\
\text{s.t.} \quad & \lambda \in \Lambda \\
& \text{for } t = 1, ..., T: \\
& f^t \in \left\{ \begin{array}{ll} \arg\min\limits_{f \in \mathcal{F}} & \mathcal{L}_{low}(\lambda, f, (x^i, y_i)_{i=1}^l \in \bar{\Omega}_t) \\ \text{s.t.} & g_{low}(\lambda, f) \leq 0 \end{array} \right\}. && \text{lower level}
\end{aligned}
$$

In the case of multigroup support vector classification the T inner problems have the SVM formulation (1.6). This problem can also be rewritten into an unconstrained form. It is helpful when using the inexact bundle algorithm for solving the bilevel problem. For the t'th fold the resulting hyperplane is identified with the variable $\tilde{w}^t \in \mathbb{R}^{feat+1}$. The inner level problem for the t'th fold can therefore be stated as

$$\tilde{w}^t \in \arg\min_{\tilde{w}} \left\{ \frac{1}{2}\|\tilde{w}\|^2 + \sum_{g=1}^{G} \left(\frac{C_g}{2} \sum_{i \in \tilde{\mathcal{N}}_t^g} \max\left\{1 - y_i \left\langle \tilde{w}, \tilde{x}^i \right\rangle, 0\right\}^2 \right) \right\}. \tag{1.7}$$

For the upper level objective function different choices are possible. All that are presented here use the implicit bias. They all can also be used with the explicit bias term.

Simply put the outer level objective should compare the different inner level solutions and pick the best one. An intuitive choice is therefore to pick the misclassification loss, that counts how many data points of the respective validation set Ω_t are misclassified when taking function f^t.

The misclassification loss can be written as

$$\mathcal{L}_{mis}(\tilde{w}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{|\mathcal{N}_t|} \sum_{i \in \mathcal{N}_t} \left(-y_i \left\langle \tilde{w}^t, \tilde{x}^i \right\rangle \right)_\star, \tag{1.8}$$

where the step function $(\cdot)_\star$ is defined component wise for a vector as

$$(r_\star)_i = \begin{cases} 1, & \text{if } r_i > 0, \\ 0, & \text{if } r_i \leq 0 \end{cases}. \tag{1.9}$$

The drawback of this simple loss function is that it is not continuous and thus not suitable for subgradient based optimization. Therefore another loss function is used for the upper level problem, the *hinge loss* or *L1-loss*. It is an upper bound on the misclassification loss and reads

$$\mathcal{L}_{hinge}(\tilde{w}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{|\mathcal{N}_t|} \sum_{i \in \mathcal{N}_t} \max\left\{1 - y_i \left\langle \tilde{w}^t, \tilde{x}^i \right\rangle, 0\right\}. \tag{1.10}$$

It is also possible to square the max term. This results in the *L2-loss* function

$$\mathcal{L}_{hingequad}(\tilde{w}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{|\mathcal{N}_t|} \sum_{i \in \mathcal{N}_t} \max\left\{1 - y_i \left\langle \tilde{w}^t, \tilde{x}^i \right\rangle, 0\right\}^2. \tag{1.11}$$

We refer to this second loss function also as *hingequad loss*.

For the bilevel problem discussed in this thesis the hingequad function is chosen as objective of the upper level problem. Hence the final resulting bilevel formulation for model selection in multigroup support vector classification is

$$
\begin{aligned}
\min_{C} \quad & \mathcal{L}_{hingequad}(\tilde{w}) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|\mathcal{N}_t|}\sum_{i\in\mathcal{N}_t}\max\left\{1-y_i\left\langle\tilde{w}^t,\tilde{x}^i\right\rangle,0\right\}^2 \\
\text{s.t.} \quad & C := (C_1,...,C_G) > 0 \\
& \text{for } t = 1,...,T \\
& \tilde{w}^t \in \arg\min_{\tilde{w}}\left\{\frac{1}{2}\|\tilde{w}\|^2 + \sum_{g=1}^{G}\left(\frac{C_g}{2}\sum_{i\in\mathcal{N}_t^g}\max\left\{1-y_i\left\langle\tilde{w},\tilde{x}^i\right\rangle,0\right\}^2\right)\right\}.
\end{aligned}
\tag{1.12}
$$

Obviously this bilevel problem contains the usual support vector classification with only one data group as a special case.

1.5 Solution with the Inexact Bundle Algorithm

more introduction?

The bilevel problem (1.12) derived above is to be solved with the bundle algorithm which one? more?. This requires having in every iteration an approximate value of the upper level objective function and an approximate subgradient of the upper level objective.

To understand the issues arising when computing especially the subgradient of a bilevel objective consider again the general bilevel problem (1.1).

At the current iterate x^k (this is now no data point but the optimization variable of the upper level problem) the function value of the upper level objective function can be computed by solving the lower level problem given x^k. The resulting solution y^k is then inserted into the upper level objective and its value can be calculated.

For computing a subgradient, it is not that simple. Additionally to the variation of the upper level function F with respect to the variable x also the variation of the solution y^k with respect to x has to be considered. To do this we follow the strategy described in [9]. Refer there also for a more throughout analysis on this subject.

1.5.1 Assumptions

Consider the general bilevel problem formulation (1.1) without the explicit constraint $G(x,y) \leq 0$. Let the feasible set X be a nonempty compact set and $\tilde{A}$ and open set containing X. To use the implicit programming approach suggested in [9], the following

assumptions have to hold true:

(A1) The upper level objective F is continuously differentiable on $\tilde{A} \times \mathbb{R}^k$.

(A2) The lower level program possesses a unique solution y_x for every $x \in \tilde{A}$.

(A3) The generalized equation coming from the lower level program is strongly regular at all points (x, y_x), where $x \in \tilde{A}$ and y_x is the corresponding solution of the lower level problem.

A *(perturbed) generalized equation* (GE) is a relation of the form

$$0 \in H(x, y) + N_U(y),$$

where $H : \mathbb{R}^n \times \mathbb{R}^k \to \mathbb{R}^k$ and $N_U(y)$ denotes the *normal cone* to the convex set $U \subset \mathbb{R}^k$ at the point $y \in \mathbb{R}^k$.

Definition 1.2 ([9, Definition 2.6, p. 18]) Let U be a convex subset of $\mathbb{R}^k$ and y in the closure of U. Then

$$N_U(y) := \{\xi \in \mathbb{R}^k \mid \langle \xi, z - y \rangle \leq 0 \quad \forall z \in U\}$$

is called normal cone to U at y.

The constraint induced by the inner level problem can be rewritten as a generalized equation via its optimality condition. Let U correspond to the feasible set of the inner level problem defined by the intersection of the set Y with the set defined by $g(\bar{x}, y) \leq 0$ for a fixed variable $\bar{x} \in X$. Then the lower level problem can be expressed in an unconstrained way by using the indicator function defined in (**??**). If the constraint set is convex, the indicator function is a convex function and for every element of its subdifferential the subgradient inequality (**??**) holds.

This yields that for all elements ξ of the subdifferential $\partial \mathtt{i}_U(y)$ it holds

$$\begin{aligned} &\mathtt{i}_U(z) - \mathtt{i}_U(y) \geq \langle \xi, z - y \rangle \quad \forall z \in U \\ \Leftrightarrow\, &\langle \xi, z - y \rangle \leq 0 \quad \forall z \in U, \end{aligned}$$

because for $y, z \in U$ the indicator function is zero. This means that the subdifferential of the indicator function of the set U at the point y coincides with the normal cone to

the set U at the point y and so for a constrained minimizer of the function $f(x,\cdot)$ on the set U the following optimality condition holds:

$$0 \in \partial f(x,y) + N_U(y).$$

If f is continuously differentiable, the function $\partial f = \nabla f : \mathbb{R}^n \times \mathbb{R}^k \to \mathbb{R}^k$ is single valued and continuous and the relation above is a generalized equation.

Let us now verify the above assumptions (A1) – (A3) for the hyper-parameter finding bilevel problem (1.12). First of all, the feasible set X has to be a convex compact set. In order to assure that, the constraint of the hyper-parameter C has to be adapted. Therefore choose two constants $l_C, u_C > 0$ with $l_C < u_C$ constrain the vector C by the component wise meant inequalities

$$l_C \leq C \leq u_C.$$

Assumption (A1) is fulfilled as $\mathcal{L}_{hingequad}$ is a continuously differentiable function because of the squared max-term.

Next assumptions (A2) and (A3) are verified. In order to do this consider the lower level problem in its unconstrained formulation. It can be rewritten in matrix form using the following definitions:

$$H(C) := \begin{pmatrix} \mathbb{I} & \\ & \tilde{C} \end{pmatrix} \in \mathbb{R}^{n_1 \times n_1}$$

$$A := \begin{pmatrix} -y_1(x^1)^\top & -1 & & \\ \vdots & & \ddots & \\ -y_N(x^N)^\top & & & -1 \end{pmatrix} \in \mathbb{R}^{nd \times n_1}, \quad b := \begin{pmatrix} -1 \\ \vdots \\ -1 \end{pmatrix} \in \mathbb{R}^{nd}.$$

Here $\tilde{C}$ is a diagonal matrix with the vector $(C_1, ..., C_1, ..., C_G, ..., C_G)^\top$ on the diagonal such that the hyper-parameters correspond to the ξ_i of the different groups. The number $n_1 = feat + 1 + nd$ where $feat$ is the dimension of the feature space and nd the number of data points in the sample.

Then the lower level problem can be written as

$$\begin{aligned} \min_{\tilde{w},\xi} \quad & \frac{1}{2}(\tilde{w}^\top, \xi^\top) H(C) \begin{pmatrix} \tilde{w} \\ \xi \end{pmatrix} \\ \text{s.t.} \quad & A \begin{pmatrix} \tilde{w} \\ \xi \end{pmatrix} \leq b. \end{aligned} \tag{1.13}$$

This formulation of the lower level problem corresponds to the one in Appendix A.1.3 in [9]. Only the objective function depends on the hyper-parameter C, not the constraints.

Problem (1.13) has a strictly convex function and therefore a unique global minimizer. Denoting the vector of all slack variables ξ_i with ξ this minimizer is given by $(\tilde{w}^*, \xi^*)^\top$. It depends on the vector of hyper-parameters C.

Due to convexity for this problem the KKT conditions are necessary and sufficient. This means that at the minimum $(\tilde{w}^*, \xi^*)$ holds

$$\begin{aligned} & \text{there is a } \mu^* \in \mathbb{R}^{nd} \text{ such that} \\ & \nabla_{(\tilde{w},\xi)} L(C, \tilde{w}^*, \xi^*, \mu^*) = 0 \\ & \mu^* \geq 0, \quad A \begin{pmatrix} \tilde{w}^* \\ \xi^* \end{pmatrix} \leq b, \quad \left\langle \mu^*, \left(A \begin{pmatrix} \tilde{w}^* \\ \xi^* \end{pmatrix} - b \right) \right\rangle = 0, \end{aligned}$$

for a given C, where $L(C, \tilde{w}, \xi, \mu) = \frac{1}{2}(\tilde{w}^\top, \xi^\top) H(C) (\tilde{w}^\top, \xi^\top)^\top + \left\langle \mu, \left(A(\tilde{w}^\top, \xi^\top)^\top - b \right) \right\rangle$ is the Lagrangian of the problem and μ^* the multiplier corresponding to the inequality constraints.

Using the KKT conditions a special GE can be derived for problem (1.13):

$$0 \in \begin{bmatrix} \nabla_{(\tilde{w},\xi)} L(C, \tilde{w}, \xi, \mu) \\ -A \left((\tilde{w})^\top, (\xi)^\top \right)^\top + b \end{bmatrix} + N_{\mathbb{R}^{feat+1} \times \mathbb{R}^{nd}_+}. \tag{1.14}$$

The derivation of this particular form is given in [9, chapter 4, p. 71–72 and chapter 5, p. 92].

This new GE is very convenient due to the simple structure of the cone but it depends now also on the Lagrange multiplier μ. Therefore this multiplier also has to be unique at the solution $\tilde{w}^*, \xi^*$. To check this introduce the following index sets:

$$I(\tilde{w},\xi) = \{i \in \{1,...,nd\} \mid A(\tilde{w},\xi)^\top - b = 0\}$$
$$I_+(\tilde{w},\xi) = \{i \in I(\tilde{w},\xi) \mid \mu_i > 0\}$$

,

which denote the set of *active* and *stronly acitve* inequality constraints respectively. The set I_+ also depends on the multiplier μ. In cases where μ is not unique, this has to be indicated by writing $I_+(\tilde{w},\xi,\mu)$. As it is shown in the following that for the presented problem the multiplier μ is always unique, it is however omitted.

For the optimal Lagrange multiplier μ^* to be unique the *linear independence constraint qualification* (LICQ) has to hold in the minimum [9, Theorem 4.8, p. 82], [14, Theorem 1, p. 3].

In the present case, where there are only inequality constraints, the LICQ holds at a point $(\tilde{w},\xi)$ if the rows of the matrix $\mathcal{J}_{(\tilde{w},\xi)}\left(A(\tilde{w},\xi)^\top - b\right) = A$ that correspond to the indices in $I(\tilde{w},\xi)$ are linearly independent [9, p. 82, 96].

For problem (1.13) the LICQ holds at every feasible point $(\tilde{w},\xi)$ because all rows of the matrix A are linearly independent. This means that linear independence particularly holds for all rows corresponding to the active constraints at any minimum $(\tilde{w}^*,\xi^*)$. Hence the lower level problem possesses a unique solution vector $(\tilde{w}^*,\xi^*)$ and a unique corresponding Lagrange multiplier μ^* For every hyper-parameter (vector) C. Thus assumption (A2) is fulfilled.

Finally to show that assumption (A3) holds, strong regularity has to be shown for the generalized equation (1.14). This is done by using Theorem 5.8 in [9, p. 96].

For stating the theorem the following definition is needed:

Definition 1.3 ([9, Definition 4.4, p. 81]) Let $M \in \mathbb{R}^{n\times n}$ be a matrix and $K \subset \mathbb{R}^n$ a cone. The matrix M is strictly copositive with respect to K if

$$\langle d, Md\rangle > 0 \quad \text{for all} \quad d \in K, d \neq 0.$$

Theorem 1.4 (c.f. [9, Theorem 5.8, p. 96]) *Suppose that LICQ holds at $(\tilde{w},\xi)$ and that the matrix $\mathcal{J}_{(\tilde{w},\xi)}\left(\nabla_{(\tilde{w},\xi)}L(C,\tilde{w},\xi,\mu)\right)$ is strictly copositive with respect to the kernel of the matrix A_{I_+}.*
Then the GE (1.14) is strongly regular at $(C,\tilde{w},\xi,\mu)$.

To assure that the Hessian of the Lagrangian is well defined, the lower level objective function has to be two times continuously differentiable. This is given for the quadratic objective function of (1.13) and it holds that

$$\mathcal{J}_{(\tilde{w},\xi)}\left(\nabla_{(\tilde{w},\xi)}L(C,\tilde{w},\xi,\mu)\right) = \begin{pmatrix} \mathbb{I} & \\ & \tilde{C} \end{pmatrix}.$$

Because all components of the vector $(C_1, ..., C_1, ..., C_G, ..., C_G)^\top$ are greater than zero, this matrix is positive definite and as such strictly copositive on the whole $\mathbb{R}^{feat+1+nd}$. This means that all necessary conditions are fulfilled in order to compute a subgradient with respect to C of the upper level objective function $\mathcal{L}_{upp}(\tilde{C}, \xi(C))$ in the way presented in [9].

1.5.2 The Adjoint Problem

In order to calculate the needed subgradient of bilevel problem the technique of *adjoint equations* is used. This technique is well known from optimal control (c.f. [9, p. 126] and references therein). For the derivation of the adjoint problem and the relults used below, refer to chapters 6 and 7 of [9].

The adjoint problem to the given bilevel problem (1.12) can be stated as the following constrained quadratic problem using the expressions defined in (1.5.1):

$$\begin{aligned} \min_{p} \quad & \frac{1}{2}\langle p, H(C)p\rangle - \left\langle \nabla_{(\tilde{w},\xi)}\mathcal{L}_{upp}(\tilde{w},\xi), p\right\rangle \\ \text{s.t.} \quad & A_{I_+M}p = 0. \end{aligned} \tag{1.15}$$

Here the index set $I_+M = I_+(\tilde{w},\xi) \cup M_i(\tilde{w},\xi)$. The set $M_i(\tilde{w},\xi) \subset I(\tilde{w},\xi) \setminus I_+(\tilde{w},\xi)$ is a subset of the inequality constraints, that are active but not strictly active at the point $(\tilde{w},\xi)$ and has to be chosen suitably. What this means is explained next.

First introduce the notation $I_0(\tilde{w},\xi) = I(\tilde{w},\xi) \setminus I_+(\tilde{w},\xi)$. Constraints whose indices are in this set are also calles *weakly active*. Let $\mathcal{P}(I_0(\tilde{w},\xi))$ be the power set of this index set and $\mathbb{K}(\tilde{w},\xi)$ an index set that contains indices for all elements of the set $\mathcal{P}(I_0(\tilde{w},\xi))$. The index set $M_i(\tilde{w},\xi)$ is then the element of $\mathcal{P}(I_0(\tilde{w},\xi))$ corresponding to the index $i \in \mathbb{K}(\tilde{w},\xi)$. To ensure a proper choice of the index set $M_i(\tilde{w},\xi)$ the book offers a theorem and a corollary that are applicable in the given context.

For better readability the brackets behind the index sets are omitted. All index sets refer to the same point.

Theorem 1.5 (c.f. [9, Theorem 7.10, p. 137]) *Consider the GE (1.14) at the point* $(C, \tilde{w}, \xi, \mu)$. *Suppose that the assumptions (A2) and (A3) hold and that for a given* $i \in \mathbb{K}(\tilde{w}, \xi)$ *the following system in* (z_1, z_2, z_3, z_4) *is inconsistent:*

$$\begin{aligned} -(A_{I_0 \setminus M_i})^\top z_1 + (H(C))^\top z_3 - (A_{I_+ \cup M})^\top z_4 &= 0 \\ z_2 + A_{M_i} z_3 &= 0 \\ \left(\mathcal{J}_C \left(H(C)(\tilde{w}, \xi)^\top\right)\right)^\top z_3 &= 0 \\ (z_1, z_2) \geq 0 \quad (z_1, z_2) &\neq 0 \\ z_3 \in \ker(A_{I_+}). \end{aligned}$$

Then subgradient can be calculated via the method above.

Corrolary 1.6 (c.f. [9, Corollary 7.12, p. 139]) *Let the assumptions of Theorem 1.5 be fulfilled. Suppose that the constraints do not depend on the variable* C *and that*

$$\ker\left(\mathcal{J}_C \left(H(C)(\tilde{w}, \xi)^\top\right)^\top\right) \cap \ker\left(A_{I_+}\right) = \{0\}.$$

Then the subgradient can be calculated via the method above for each $i \in \mathbb{K}(\tilde{w}, \xi)$.

Neither the assumptions of the corollary nor those of the theorem can be assured in general. They have therefore to be checked for the specific situation in every iteration.

how checked? to be added when programmed

1.6 On Error Bounds and Regularity

The algorithms ??.1 and ??.1 – only one? put some assumptions on the objective function and the nature of the inexactness. In this section we give a short comment on how far theses assumptions can be met.

1.6.1 Error Bounds

It is assumed in (**??**) that the error on the function value and subgradient are bounded. This bound does not have to be known, but it has to exist and allows an estimation of the possible accuracy to which the algorithms??? can solve the given problem (c.f. Lemma **??**).

Due to the complicated structure of bilevel problems no tight error bounds can be given for problem (1.12) but the existence of general error bounds can be shown.

The error on the function value in bilevel problems originates from the fact, that in the implicit approach the numerically calculated minimal point of the lower level is inserted into the upper level function. Of course, this optimal point can only be approximated to a given tolerance. As the lower level problem of (1.12) is a strictly convex quadratic optimization problem, a very common stopping criterion is to check for approximate first order optimality.

For constrained problems first order optimality means that the KKT conditions (which are necessary and sufficient in the convex case) are fulfilled. Approximate optimality then means, that the conditions hold true within the bounds of a chosen stopping tolerance $\texttt{tol} > 0$.

The solver that is used for the numerical solution of the problem in this thesis is the `quadprog` solver from MATLAB. This solver uses the above stopping condition scaled by the input values [6].

This however cannot give an estimate for the minimizing argument of the lower level objective. This can be seen when taking a strictly convex whose slope is so small, that the stopping condition is fulfilled for every point on a compact subset of the feasible set U_C. In this case also the approximate minimizing point found by any algorithm can be any point in U_C.

It still can be assured, that the minimizer $\tilde{w}$ found by an algorithm is not infinitely far away from the exact minimizer $\tilde{w}^*$. This follows immediately if the constraint set is compact. If it is unbounded in any direction this follows from strict convexity. Without loss of generality instead of the KKT conditions one can consider the first order optimality condition in the unconstrained case, namely that the norm of the gradient at a minimizer is zero. (Due to unboundedness of the constraint set is the examined directions the problem can be treated as an unconstrained one for them. In the direction where the constraint set is bounded, possible choices for $\tilde{w}$ are also bounded.) For strictly convex functions the gradient is strictly monotone, hence the area where its norm is smaller or

equal to any chosen tolerance is compact. This means that in this case all possible choices of $\tilde{w}$ are bounded.

Using local Lipschitz continuity of the upper level objective function on the admissible set we can conclude that

$$\|\mathcal{L}_{upp}(\tilde{w}) - \mathcal{L}_{upp}(\tilde{w}^*)\| \leq L_{upp}\|\tilde{w} - \tilde{w}^*\| \leq L_{upp} D_{\tilde{w}}(\texttt{tol})$$

where L_{upp} is the Lipschitz constant of the upper level objective on the admissible set. The number $D_{\tilde{w}}(\texttt{tol})$ is the maximal distance of points $\tilde{w}_1$ and $\tilde{w}_2$ in the compact set originating from the intersection of the constraint set and the set where the lower level objective has a gradient smaller than the chosen tolerance `tol`.

An equally loose bound can be provided for the approximate subgradient via local Lipschitz continuity of the upper level objective and Lipschitz continuity of the solution map [9, chapter 5]. The inexactness of the subgradient has two reasons. On the one hand the subgradient is not calculated at the exact minimizer $\tilde{w}^*$ but at an approximated point $\tilde{w}$. As the subdifferential is bounded by the Lipschitz constant on U_{ad} the bound for that part of the subgradient error that originates from taking the subgradient only at an approximate point is given by $2L_{upp}L_w$. Here $L_{\tilde{w}}$ is the Lipschitz constant of the solution function $\tilde{w}(C)$.

On the other hand for computation of the subgradient the adjoint problem (1.15) has to be solved numerically and therefore only gives an approximate solution. As the adjoint problem is also a strictly convex constrained optimization problem its minimizing argument p can be estimated the same way as above.

Hence the distance between the exact subgradient and the approximate one in the sense of p is

$$\begin{aligned}\langle \mathcal{J}_C H(C)\tilde{w}^*, (p^* - p)\rangle &\leq \|\mathcal{J}_C H(C)\|_2 \|\tilde{w}^*\| \|p^* - p\| \\ &\leq \|\max\{1, C_{max}\}\|\tilde{w}_{max} D_p(\texttt{tol}).\end{aligned}$$

Here C_{max} denotes the maximum component of the vector of hyper-parameters and $\tilde{w}_{max}$ is the maximum of the function $\tilde{w}(C)$ on U_{ad}. The sum of those bounds is the overall bound of the approximate subgradient error.

1.6.2 Regularity

For the proof of Lemma **??** regularity of the upper level objective function with the solution map inserted is required.

We have however to remark here, that from Proposition 7.4 in [9] only follows weak semismoothness of this function. In [10, p. 82] an example is given that semismooth, quasidifferentiable function does not have to be subdifferentially regular. As semismooth functions are a subset of weakly semismooth functions, we cannot conclude that the objective function of (1.12) with the solution of the lower level plugged in is regular.

Proving regularity for the particular instance (1.12) of a bilevel problem may be possible, taking into account the structure provided by the strictly convex objective functions of both levels and the fact that the class of semismooth and regular functions is identical to the class of lower-$\mathcal{C}^1$ functions [4, equation (3), p. 5]. It lies however beyond the scope of this thesis.

1.7 Numerical Experiments

Rechtschreibfehler, Namen, Stil überprüfen

References

[1] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[2] A. Fuduli, M. Gaudioso, and G. Giallombardo. Minimizing nonconvex nonsmooth functions via cutting planes and proximity control. *SIAM J. Optim*, 14(3):743–756, 2004.

[3] S. R. Gunn. Support vector machines for classification and regression. Technical report, Faculty of Enigneering, Science and Mathematics, School of Electronics and Computer Science, University of Southampton, 1998.

[4] W. Hare, C. Sagastizàbal, and M. Solodov. A proximal bundle method for nonsmooth nonconvex functions with inexact information. *Computational Optimization and Applications*, 63:1–28, 2016.

[5] G. Kunapuli. *A bilevel optimization approach to machine learning*. PhD thesis, Rensselaer Polytechnic Institute Troy, New York, 2008.

[6] The MathWorks, Inc. *Documentation: Optimization Toolbox (quadprog)*, 2017.

[7] G. Moore, C. Bergeron, and K. P. Bennett. Gradient-type methods for primal SVM model selection. Technical report, Rensselaer Polytechnic Institute, 2010.

[8] G. Moore, C. Bergeron, and K. P. Bennett. Model selection for primal SVM. *Machine Learning*, 85(1):175–208, 2011.

[9] J. Outrata, M. Kočvara, and J. Zowe. *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints*. Springer US, 1998.

[10] J. E. Spingarn. Submonotone subdifferentials of Lipschitz functions. *Trans. Amer. Math. Soc.*, 264:77 – 89, 1981.

[11] V. N. Vapnik. *Statistical Learning Theory*. JOHN WILEY & SONS INC, 1998.

[12] V. N. Vapnik. An overview of statistical learning theory. *IEEE Trans. Neural Networks*, 10(5):988–999, 1999.

[13] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer New York, 2013.

[14] G. Wachsmuth. On licq and the uniqueness of lagrange multipliers. *Operations Research Letters*, 41(1):78 – 80, 2013.