Journal of Machine Learning Research 10 (2009) 2137-2155 Submitted 4/08; Revised 7/09; Published 9/09

Discriminative Learning Under Covariate Shift

Steffen Bickel BICKEL@CS.UNI-POTSDAM.DE
Michael Brückner MIBRUECK@CS.UNI-POTSDAM.DE
Tobias Scheffer SCHEFFER@CS.UNI-POTSDAM.DE
University of Potsdam, Department of Computer Science
August-Bebel-Str. 89
14482 Potsdam, Germany

Editor: Bianca Zadrozny

Abstract

We address classification problems for which the training instances are governed by an input distribution that is allowed to differ arbitrarily from the test distribution—problems also referred to as classification under covariate shift. We derive a solution that is purely discriminative: neither training nor test distribution are modeled explicitly. The problem of learning under covariate shift can be written as an integrated optimization problem. Instantiating the general optimization problem leads to a kernel logistic regression and an exponential model classifier for covariate shift. The optimization problem is convex under certain conditions; our findings also clarify the relationship to the known kernel mean matching procedure. We report on experiments on problems of spam filtering, text classification, and landmine detection.

Keywords: covariate shift, discriminative learning, transfer learning

1. Introduction

Most machine learning algorithms are constructed under the assumption that the training data is governed by the exact same distribution which the model will later be exposed to. In practice, control over the data generation process is often less perfect. Training data may be obtained under laboratory conditions that cannot be expected after deployment of a system; spam filters may be used by individuals whose distribution of inbound emails diverges from the distribution reflected in public training corpora; image processing systems may be deployed to foreign geographic regions where vegetation and lighting conditions result in a distinct distribution of input patterns.

The case of distinct training and test distributions in a learning problem has been referred to as *covariate shift* and *sample selection bias*—albeit the term sample selection bias actually refers to a case in which each training instance is originally drawn from the test distribution, but is then selected into the training sample with some probability, or discarded otherwise.

The covariate shift model and the *missing at random* case in the sample selection bias model allow for differences between the training and test distribution of instances; the conditional distribution of the class variable given the instance is constant over training and test set.

In the *covariate shift* problem setting, a training sample is available in matrix $\mathbf{X}_L$ with row vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$. This training sample is governed by an unknown distribution $p(\mathbf{x}|\lambda)$. Vector $\mathbf{y}$ with elements $y_1, \ldots, y_m$ are the labels for training examples and are drawn according to an unknown target concept $p(y|\mathbf{x})$. In addition, unlabeled test data becomes available in matrix $\mathbf{X}_T$ with rows

$\mathbf{x}_{m+1}, \ldots, \mathbf{x}_{m+n}$. The test data is governed by a different unknown distribution, $p(\mathbf{x}|\theta)$. Training and test distribution may differ arbitrarily, but there is only one unknown target conditional class distribution $p(y|\mathbf{x})$.

In discriminative learning tasks such as classification, the classifier's goal is to produce the correct output given the input. It is widely accepted that this is best performed by discriminative learners that directly maximize a quality measure of the produced output. Model-based optimization criteria such as the joint likelihood of input and output, by contrast, additionally assess how well the classifier models the distribution of input values. This amounts to adding a term to the criterion that is irrelevant for the task at hand.

We contribute a discriminative model for learning under different training and test distributions. The model directly characterizes the divergence between training and test distribution, without the intermediate—intrinsically model-based—step of estimating training and test distribution. We formulate the search for all model parameters as an integrated optimization problem. This complements the predominant procedure of first estimating the bias of the training sample, and then learning the classifier on a weighted version of the training sample. We show that the integrated optimization can be convex, depending on the model type; it is convex for the exponential model. We derive a Newton gradient descent procedure, leading to a kernel logistic regression and an exponential model classifier for covariate shift.

After reviewing models for differing training and test distributions in Section 2, we introduce our integrated model in Section 3. We derive primal and kernelized classifiers for differing training and test distributions in Sections 4 and 5. In Section 6, we analyze the convexity of the integrated optimization problem. Section 7 describes an approximation to the joint optimization problem and Section 8 reveals a new interpretation of kernel mean matching and analyzes the relationship to our model. In Section 9 we discuss different tuning procedures for learning under covariate shift. Section 10 provides empirical results and Section 11 concludes.

The discriminative model for the logistic loss is described in a prior conference publication (Bickel et al., 2007). Our original results showed that the resulting optimization problem is not convex. New findings (Section 6) show that the integrated optimization problem can in fact be convex when the loss function is chosen appropriately. Section 7 describes a two-stage approximation that allows to train virtually any type of classifier under covariate shift. The new Section 8 characterizes the relation to kernel mean matching. New experiments include the exponential target model. Section 10 uses an experimental setting that differs from the setting of Bickel et al. (2007) in the parameter tuning process. In some cases, the new setting has improved the performance of baseline methods.

2. Prior Work

If training and test distributions were known, then the loss on the test distribution could be minimized by weighting the loss on the training distribution with an instance-specific factor. Proposition 1 (Shimodaira, 2000) illustrates that the scaling factor has to be $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$.

Proposition 1 *The expected loss with respect to* θ *equals the expected loss with respect to* λ *with weights* $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ *for the loss incurred by each* $\mathbf{x}$*, provided that the support of* $p(\mathbf{x}|\theta)$ *is contained in the*

support of $p(\mathbf{x}|\lambda)$*:*

$$\mathbf{E}_{(\mathbf{x},y)\sim\theta}[\ell(f(\mathbf{x}),y)] \quad = \quad \mathbf{E}_{(\mathbf{x},y)\sim\lambda}\left[\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}\ell(f(\mathbf{x}),y)\right]. \tag{1}$$

After expanding the expected value into its integral $\int \ell(f(\mathbf{x}),y)p(\mathbf{x},y|\theta)d\theta$, the joint distribution $p(\mathbf{x},y|\lambda)$ is decomposed into $p(\mathbf{x}|\lambda)p(y|\mathbf{x},\lambda)$. Since $p(y|\mathbf{x},\lambda) = p(y|\mathbf{x}) = p(y|\mathbf{x},\theta)$ is the global conditional distribution of the class variable given the instance, Proposition 1 follows. All instances $\mathbf{x}$ with positive $p(\mathbf{x}|\theta)$ are integrated over. Hence, Equation 1 holds as long as each $\mathbf{x}$ with positive $p(\mathbf{x}|\theta)$ also has a positive $p(\mathbf{x}|\lambda)$; otherwise, the denominator vanishes. This shows that covariate shift can only be compensated for as long as the training distribution covers the entire support of the test distribution. If a test instance had zero density under the training distribution, the test-to-training density ratio which it would need to be scaled with would incur a zero denominator.

Both, $p(\mathbf{x}|\theta)$ and $p(\mathbf{x}|\lambda)$ are unknown, but $p(\mathbf{x}|\theta)$ is reflected in $\mathbf{X}_T$, as is $p(\mathbf{x}|\lambda)$ in $\mathbf{X}_L$. A straightforward approach to compensating for covariate shift is to first obtain estimates $\hat{p}(\mathbf{x}|\theta)$ and $\hat{p}(\mathbf{x}|\lambda)$ from the test and training data, respectively, using kernel density estimation (Shimodaira, 2000; Sugiyama and Müller, 2005). In a second step, the estimated density ratio is used to re-sample the training instances, or to train with weighted examples.

This method decouples the problem. First, it estimates training and test distributions. This step is intrinsically model-based and only loosely related to the ultimate goal of accurate classification. In a subsequent step, the classifier is derived given fixed weights. Since the parameters of the final classifier and the parameters that control the weights are not independent, this decomposition into two optimization steps cannot generally find the optimal setting of the *joint* parameter vector.

A line of work on learning under sample selection bias has meandered from the statistics and econometrics community into machine learning (Heckman, 1979; Zadrozny, 2004). Sample selection bias relies on a model of the data generation process. Test instances are drawn under $p(\mathbf{x}|\theta)$. Training instances are drawn by first sampling $\mathbf{x}$ from the test distribution $p(\mathbf{x}|\theta)$. A selector variable σ then decides whether $\mathbf{x}$ is moved into the training set ($\sigma = 1$) or moved into the rejected set ($\sigma = -1$). For instances in the training set ($\sigma = 1$) a label is drawn from $p(y|\mathbf{x})$, for the instances in the rejected set the labels are unknown. A typical scenario for sample selection bias is credit scoring. The labeled training sample consists of customers who where given a loan in the past and the rejected sample are customers that asked for but where not given a loan. New customers asking for a loan reflect the test distribution.

In the *missing at random* case, the selector variable is only dependent on $\mathbf{x}$, but not on y; that is, $p(\sigma = 1|\mathbf{x},y,\theta,\lambda) = p(\sigma = 1|\mathbf{x},\theta,\lambda)$. The distribution of the selector variable then maps the test onto the training distribution:

$$p(\mathbf{x}|\lambda) \propto p(\mathbf{x}|\theta)p(\sigma = 1|\mathbf{x},\theta,\lambda).$$

Proposition 2 (Zadrozny, 2004; Bickel and Scheffer, 2007) says that minimizing the loss on instances weighted by $p(\sigma|\mathbf{x},\theta,\lambda)^{-1}$ in fact minimizes the expected loss with respect to θ.

Proposition 2 *The expected loss with respect to* θ *is proportional to the expected loss with respect to* λ *with weights* $p(\sigma = 1|\mathbf{x},\theta,\lambda)^{-1}$ *for the loss incurred by each* $\mathbf{x}$*, provided that the support of* $p(\mathbf{x}|\theta)$ *is contained in the support of* $p(\mathbf{x}|\lambda)$*:*

$$\mathbf{E}_{(\mathbf{x},y)\sim\theta}[\ell(f(\mathbf{x}),y)] \quad \propto \quad \mathbf{E}_{(\mathbf{x},y)\sim\lambda}\left[\frac{1}{p(\sigma = 1|\mathbf{x},\theta,\lambda)}\ell(f(\mathbf{x}),y)\right].$$

When the model is implemented, $p(\sigma = 1|\mathbf{x},\theta,\lambda)$ is learned by discriminating the training against the rejected examples; in a second step the target model is learned by following Proposition 2 and weighting training examples by $p(\sigma|\mathbf{x},\theta,\lambda)^{-1}$. No test examples drawn directly from $p(\mathbf{x}|\theta)$ are needed to train the model, only labeled selected and unlabeled rejected examples are required. This is in contrast to the covariate shift model that requires samples drawn from the test distribution, but no selection process is assumed and no rejected examples are needed. Covariate shift models can be applied to learning under sample selection bias in the missing at random setting by treating the selected examples as labeled sample and the union of selected (ignoring the labels) and rejected examples as unlabeled sample.

Propensity scores (Rosenbaum and Rubin, 1983; Lunceford and Davidian, 2004) are applied in settings related to sample selection bias; the training data is again assumed to be drawn from the test distribution $p(\mathbf{x}|\theta)$ followed by a selection process. The difference to the setting of sample selection bias is that the selected *and* the rejected examples are labeled. Weighting the selected examples by the inverse of the propensity score $p(\sigma = 1|\mathbf{x},\lambda,\theta)^{-1}$ and weighting the rejected examples by $p(\sigma = -1|\mathbf{x},\lambda,\theta)^{-1}$ results in two unbiased samples with respect to the test distribution.

Propensity scoring can precede a variety of analysis steps. This can be the training of a target model on re-weighted data or just a statistical analysis of the two re-weighted samples. A typical application for propensity scores is the analysis of the success of a medical treatment. Patients are selected to be given the treatment and some other patients are selected into the control group. If the selector variable is not independent of $\mathbf{x}$ (patients may be chosen for an experimental therapy only if they meet specific requirements), the outcome (e.g., ratio of cured patients) of the two groups cannot be compared directly, propensity scores have to be applied.

Maximum entropy density estimation under sample selection bias has been studied by Dudik et al. (2005). Bickel and Scheffer (2007) impose a Dirichlet process prior on several learning problems with related sample selection bias. Elkan (2001) and Japkowicz and Stephen (2002) investigate the case of training data that is only biased with respect to the class ratio, this can be seen as sample selection bias where the selection only depends on y.

Kernel mean matching (Huang et al., 2007) is a two-step method that first finds weights for the training instances such that the first momentum of training and test sets—that is, their mean value—matches in feature space. The subsequent training step uses these weights. Matching the means in feature space is equivalent to matching all moments of the distributions if a universal kernel is used. Huang et al. (2007) derive a quadratic program from Equation 2 that can be solved with standard optimization tools. $\Phi(\cdot)$ is a mapping into a feature space and B is a regularization parameter.

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \left\| \tfrac{1}{m}\sum_{i=1}^{m} \alpha_i \Phi(\mathbf{x}_i) - \tfrac{1}{n}\sum_{i=m+1}^{m+n} \Phi(\mathbf{x}_i) \right\|^2 \\ &\text{subject to } \alpha_i \in [0,B] \text{ and } \left| \tfrac{1}{m}\sum_{i=1}^{m} \alpha_i - 1 \right| \leq \varepsilon \end{aligned} \tag{2}$$

Cortes et al. (2008) theoretically analyze the error that gets introduced by estimating sample selection bias from data. Their analysis covers the kernel mean matching procedure and a cluster-based estimation technique.

KLIEP (Sugiyama et al., 2008) estimates resampling weights for the training examples by minimizing the Kullback-Leibler divergence between the test distribution and the weighted training distribution. Tsuboi et al. (2008) derive an extension to KLIEP for large-scale applications and reveal a close relationship to kernel mean matching.

3. Integrated Model

Our goal is to find model parameters $\mathbf{w}$ for a probabilistic classification model $f(\mathbf{x}) = \text{argmax}_y\, p(y|\mathbf{x};\mathbf{w})$. The model should correctly predict labels of the test data $\mathbf{X}_T$ drawn from $p(\mathbf{x}|\theta)$. A regular *maximum a posteriori* estimation $\mathbf{w}' = \text{argmax}_\mathbf{w}\, p(\mathbf{y}|\mathbf{X}_L;\mathbf{w})p(\mathbf{w})$, would only use the training data $(\mathbf{y},\mathbf{X}_L)$ governed by $p(\mathbf{x}|\lambda)$. By ignoring the test data, this estimate will not generally result in a model that predicts the missing labels of the test data with a minimum error because the training distribution $p(\mathbf{x}|\lambda)$ is different from the test distribution $p(\mathbf{x}|\theta)$.

In the following we devise a probabilistic model that accounts for the difference between training and test distribution. Before we describe the model we define a joint data matrix $\mathbf{X}$ that is a concatenation of the matrices $\mathbf{X}_L$ and $\mathbf{X}_T$. The model is based on a binary selector variable s: Given an instance vector $\mathbf{x}$ from the joint matrix $\mathbf{X}$ of all available instances, selector variable s decides whether $\mathbf{x}$ is drawn into the test data $\mathbf{X}_T$ ($s=-1$) or into the training data $\mathbf{X}_L$ ($s=1$) in which case $\mathbf{y}$ is determined. The variable s is governed by the distribution $p(s|\mathbf{x};\mathbf{v})$. Parameter $\mathbf{v}$ characterizes the discrepancy between the training and test distribution. Based on the model for s we can now describe the generative process underlying our model:

1. Draw parameter vectors $\mathbf{v}$ and $\mathbf{w}$ from prior distributions $p(\mathbf{v})$ and $p(\mathbf{w})$;
2. For each row $\mathbf{x}$ in matrix $\mathbf{X}$ draw binary variable s from distribution $p(s|\mathbf{x};\mathbf{v})$; accordingly, the likelihood of the vector of all selector variables $\mathbf{s}$ is $p(\mathbf{s}|\mathbf{X};\mathbf{v}) = \prod_{i=1}^{m+n} p(s_i|\mathbf{x}_i;\mathbf{v})$;
3. For all selected training examples (all examples $\mathbf{x}_i$ with $s_i = 1$) draw vector $\mathbf{y}$ of all labels from $p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v})$.

This generative process corresponds to the following factorization of the joint probability of the vector of labels $\mathbf{y}$, vector of selector variables $\mathbf{s}$, and parameter vectors $\mathbf{v}$ and $\mathbf{w}$:

$$p(\mathbf{y},\mathbf{s},\mathbf{w},\mathbf{v}|\mathbf{X}) \quad = \quad p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v})p(\mathbf{s}|\mathbf{X};\mathbf{v})p(\mathbf{w})p(\mathbf{v}). \tag{3}$$

3.1 Maximum A Posteriori Parameter Inference

For parameter inference we want to find parameters $\mathbf{w}$ that maximize the posterior probability given all available data (Equation 4). The available data are the data matrix $\mathbf{X}$, the label vector $\mathbf{y}$, and the selection vector $\mathbf{s}$, that splits the data matrix into training and test data. Because the parameter $\mathbf{v}$ is unknown and is not needed for the final classifier the best we can do is to integrate it out (Equation 5).

$$\mathbf{w}^* \quad = \quad \text{argmax}_\mathbf{w}\, p(\mathbf{w}|\mathbf{y},\mathbf{s},\mathbf{X}) \tag{4}$$

$$\quad = \quad \text{argmax}_\mathbf{w} \int p(\mathbf{w},\mathbf{v}|\mathbf{y},\mathbf{s},\mathbf{X})d\mathbf{v}. \tag{5}$$

Integrating over $\mathbf{v}$ is computationally infeasible. In Equation 6, the integral is therefore approximated by the single assignment of values to the parameters which maximizes the posterior—the *maximum a posteriori* (MAP) estimator. In our case, the MAP estimator naturally assigns values to all parameters, $\mathbf{w}$ and $\mathbf{v}$.

$$(\mathbf{w}_{\text{MAP}},\mathbf{v}_{\text{MAP}}) \quad = \quad \text{argmax}_{\mathbf{w},\mathbf{v}}\, p(\mathbf{w},\mathbf{v}|\mathbf{y},\mathbf{s},\mathbf{X}) \tag{6}$$

$$\quad = \quad \text{argmax}_{\mathbf{w},\mathbf{v}}\, p(\mathbf{y},\mathbf{s},\mathbf{w},\mathbf{v}|\mathbf{X}) \tag{7}$$

$$\quad = \quad \text{argmax}_{\mathbf{w},\mathbf{v}}\, p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v})p(\mathbf{s}|\mathbf{X};\mathbf{v})p(\mathbf{w})p(\mathbf{v}). \tag{8}$$

Equation 7 follows from multiplication with a constant $p(\mathbf{y},\mathbf{s}|\mathbf{X})$ and from application of the chain rule. Equation 8 applies the factorization from the generative process of Equation 3.

The class-label posterior $p(y|\mathbf{x};\mathbf{w}_{\text{MAP}})$ is conditionally independent of $\mathbf{v}_{\text{MAP}}$ given $\mathbf{w}_{\text{MAP}}$. However, $\mathbf{w}_{\text{MAP}}$ and $\mathbf{v}_{\text{MAP}}$ are dependent. Assigning a single MAP value to $[\mathbf{w},\mathbf{v}]$ instead of integrating over $\mathbf{v}$ is a common approximation. However, sequential maximization of $p(\mathbf{s}|\mathbf{X};\mathbf{v})$ over parameters $\mathbf{v}$ followed by maximization of $p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v})$ over parameters $\mathbf{w}$ with fixed $\mathbf{v}$ would amount to an additional degree of approximation and will not generally coincide with the maximum of the product in Equation 8. Such a sequential maximization corresponds to the predominant two-step procedure for learning under covariate shift.

In the next sections we will discuss the likelihood functions $p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v})$ and $p(\mathbf{s}|\mathbf{X};\mathbf{v})$ and the optimization problem for parameter inference based on maximization of Equation 8.

3.2 Label Likelihood and Discriminative Weighting Factors

In order to define the label likelihood we first derive a discriminative expression for $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ which will no longer include any density on instances. When $p(s=-1)>0$, which is implied by the test set not being empty, the definition of s allows us to rewrite the test distribution as $p(\mathbf{x}|\theta)=p(\mathbf{x}|s=-1,\theta)$. Since test instances are only dependent on parameter θ but not on parameter λ, equation $p(\mathbf{x}|s=-1,\theta)=p(\mathbf{x}|s=-1,\theta,\lambda)$ follows. By an analogous argument, $p(\mathbf{x}|\lambda)=p(\mathbf{x}|s=1,\theta,\lambda)$ when $p(s=1)>0$. This implies Equation 9.

In Equation 10, Bayes' rule is applied twice; the two terms of $p(\mathbf{x}|\theta,\lambda)$ cancel each other out in Equation 11. Since $p(s=-1|\mathbf{x},\theta,\lambda)=1-p(s=1|\mathbf{x},\theta,\lambda)$, Equation 12 follows.

The conditional $p(s=1|\mathbf{x},\theta,\lambda)$ discriminates training ($s=1$) against test instances ($s=-1$).

$$\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)} = p(\mathbf{x}|s=-1,\theta,\lambda)\frac{1}{p(\mathbf{x}|s=1,\theta,\lambda)} \tag{9}$$

$$= \frac{p(s=-1|\mathbf{x},\theta,\lambda)p(\mathbf{x}|\theta,\lambda)}{p(s=-1|\theta,\lambda)}\frac{p(s=1|\theta,\lambda)}{p(s=1|\mathbf{x},\theta,\lambda)p(\mathbf{x}|\theta,\lambda)} \tag{10}$$

$$= \frac{p(s=1|\theta,\lambda)}{p(s=-1|\theta,\lambda)}\frac{p(s=-1|\mathbf{x},\theta,\lambda)}{p(s=1|\mathbf{x},\theta,\lambda)} \tag{11}$$

$$= \frac{p(s=1|\theta,\lambda)}{p(s=-1|\theta,\lambda)}\left(\frac{1}{p(s=1|\mathbf{x},\theta,\lambda)}-1\right). \tag{12}$$

The significance of Equation 12 is that it shows how the optimal example weights, the test-to-training ratio $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$, can be determined without knowledge of either training or test density. The right hand side of Equation 12 can be evaluated based on a model that discriminates training against test examples and outputs how much more likely an instance is to occur in the test data than it is to occur in the training data. Instead of potentially high-dimensional densities $p(\mathbf{x}|\theta)$ and $p(\mathbf{x}|\lambda)$, a conditional distribution of the single binary variable s needs to be modeled.

The expression $p(s|\mathbf{x},\theta,\lambda)$ in Equation 12 corresponds to the parametric model $p(s|\mathbf{x};\mathbf{v})$ of Equation 3. With this model we can predict test-to-training density ratios for the training data in $\mathbf{X}_L$ according to Equation 12.

Since our goal is discriminative training, the likelihood function $p(\mathbf{y}|\mathbf{X}_L;\mathbf{w})$ (not taking training-test difference $\mathbf{v}$ into account) would be $\prod_i p(y_i|\mathbf{x}_i;\mathbf{w})$. By using this likelihood $p(\mathbf{y}|\mathbf{X}_L;\mathbf{w})$ instead of $p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v})$, one would wrongly assume that the training data $\mathbf{X}_L$ was governed by the test

distribution. Intuitively, $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ dictates how many times, on average, $\mathbf{x}$ should occur in $\mathbf{X}_L$ if $\mathbf{X}_L$ was governed by the test distribution θ. When the individual conditional likelihood of $\mathbf{x}$ is $p(y|\mathbf{x};\mathbf{w})$, then the likelihood of $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ occurrences of $\mathbf{x}$ is $p(y|\mathbf{x};\mathbf{w})^{\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}}$. Using a parametric model $p(s|\mathbf{x};\mathbf{v})$, according to Equation 12 the test-to-training ratio $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ can be expressed as [1]

$$\frac{p(s=1)}{p(s=-1)}\left(\frac{1}{p(s=1|\mathbf{x};\mathbf{v})}-1\right).$$

Therefore, we define the likelihood function as [2]

$$p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v}) = \prod_{i=1}^{m} p(y_i|\mathbf{x}_i;\mathbf{w})^{\frac{p(s=1)}{p(s=-1)}\left(\frac{1}{p(s_i=1|\mathbf{x}_i;\mathbf{v})}-1\right)}. \tag{13}$$

As an immediate corollary of Manski and Lerman (1977), the likelihood function of Equation 13 has the property that when the true value $\mathbf{v}^*$ is given, its maximizer over $\mathbf{w}$ is a consistent estimator of the true parameter $\mathbf{w}^*$ that has produced labels for the test data under the test distribution θ. That is, as the sample grows, the maximizer of Equation 13 converges in probability to the true value $\mathbf{w}^*$ of parameter $\mathbf{w}$.

For the statistical analysis of case-control studies, Prentice and Pyke (1979) estimate the ratio of two odds ratios with a discriminative model using a formula similar to Equation 12. This double odds ratio is a statistical measure of the relative risk of an incidence (e.g., lung cancer) given a specific exposure (e.g., cigarette smoking) based on data from a retrospective study.

3.3 Optimization Problem for Integrated Model

The likelihood function $p(\mathbf{s}|\mathbf{X};\mathbf{v})$ resolves to $p(s_i=1|\mathbf{x}_i;\mathbf{v})$ for all training instances and $p(s_i=-1|\mathbf{x}_i;\mathbf{v})$ for all test instances:

$$p(\mathbf{s}|\mathbf{X};\mathbf{v}) = \prod_{i=1}^{m} p(s_i=1|\mathbf{x}_i;\mathbf{v}) \prod_{i=m+1}^{m+n} p(s_i=-1|\mathbf{x}_i;\mathbf{v}). \tag{14}$$

Equation 15 summarizes Equations 6 to 8 and Equation 16 inserts the likelihood models (Equations 13 and 14).

$$\begin{aligned} p(\mathbf{w},\mathbf{v}|\mathbf{y},\mathbf{s},\mathbf{X}) &\propto p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v})p(\mathbf{s}|\mathbf{X};\mathbf{v})p(\mathbf{w})p(\mathbf{v}) && (15)\\ &= \left(\prod_{i=1}^{m} p(y_i|\mathbf{x}_i;\mathbf{w})^{\frac{p(s=1)}{p(s=-1)}\left(\frac{1}{p(s_i=1|\mathbf{x}_i;\mathbf{v})}-1\right)}\right) && (16)\\ &\quad \left(\prod_{i=1}^{m} p(s_i=1|\mathbf{x}_i;\mathbf{v}) \prod_{i=m+1}^{m+n} p(s_i=-1|\mathbf{x}_i;\mathbf{v})\right) p(\mathbf{w})p(\mathbf{v}). \end{aligned}$$

Using a logistic model for $p(s=1|\mathbf{x};\mathbf{v})$, we notice that Equation 12 can be simplified as in Equation 17.

$$\frac{p(s=1)}{p(s=-1)}\left(\frac{1}{1/(1+\exp(-\mathbf{v}^\mathsf{T}\mathbf{x}))}-1\right) = \frac{p(s=1)}{p(s=-1)}\exp(-\mathbf{v}^\mathsf{T}\mathbf{x}). \tag{17}$$

1. For a simplified presentation we drop the conditioning in the prior ratio, that is, $p(s|\theta,\lambda)=p(s)$.
2. The variable s in the prior ratio $\frac{p(s=1)}{p(s=-1)}$ does not need an index i because at this point it is not conditioned on $\mathbf{x}_i$.

Optimization Problem 1 is derived from Equation 16 in logarithmic form, using linear models $\mathbf{v}^\mathsf{T}\mathbf{x}_i$ and $\mathbf{w}^\mathsf{T}\mathbf{x}_i$ and a logistic model for $p(s=1|\mathbf{x};\mathbf{v})$. Negative log-likelihoods are abbreviated $\ell_\mathbf{w}(y_i\mathbf{w}^\mathsf{T}\mathbf{x}_i) = -\log p(y_i|\mathbf{x}_i;\mathbf{w})$ and $\ell_\mathbf{v}(s_i\mathbf{v}^\mathsf{T}\mathbf{x}_i) = -\log p(s_i|\mathbf{x}_i;\mathbf{v})$, respectively; this notation emphasizes the duality between likelihoods and empirical loss functions. The regularization terms correspond to Gaussian priors on $\mathbf{v}$ and $\mathbf{w}$ with variances $\sigma_\mathbf{v}^2$ and $\sigma_\mathbf{w}^2$.

Optimization Problem 1 *Over all* $\mathbf{w}$ *and* $\mathbf{v}$*, minimize*

$$\sum_{i=1}^{m} \frac{p(s=1)}{p(s=-1)} \exp(-\mathbf{v}^\mathsf{T}\mathbf{x}_i)\ell_\mathbf{w}(y_i\mathbf{w}^\mathsf{T}\mathbf{x}_i) + \sum_{i=1}^{m+n} \ell_\mathbf{v}(s_i\mathbf{v}^\mathsf{T}\mathbf{x}_i) + \frac{1}{2\sigma_\mathbf{w}^2}\mathbf{w}^\mathsf{T}\mathbf{w} + \frac{1}{2\sigma_\mathbf{v}^2}\mathbf{v}^\mathsf{T}\mathbf{v}.$$

4. Primal Learning Algorithm

We derive a Newton gradient descent method that directly minimizes Optimization Problem 1 in the attribute space. To this end, we need to derive the gradient and the Hessian of the objective function. The update rule assumes the form of a set of linear equations that have to be solved for the update vector $[\Delta_\mathbf{v}, \Delta_\mathbf{w}]^\mathsf{T}$. It depends on the current parameters $[\mathbf{v},\mathbf{w}]^\mathsf{T}$, all combinations of training and test data, and resulting coefficients. In order to express the update rule as a single equation in matrix form, we define

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_L & \mathbf{X}_T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{X}_L \end{bmatrix},$$

where $\mathbf{X}_L$ and $\mathbf{X}_T$ are the matrices of training vectors and test vectors, respectively.

Theorem 3 *The update step for the Newton gradient descent minimization of Optimization Problem 1 is* $[\mathbf{v}',\mathbf{w}']^\mathsf{T} \leftarrow [\mathbf{v},\mathbf{w}]^\mathsf{T} + [\Delta_\mathbf{v},\Delta_\mathbf{w}]^\mathsf{T}$ *with*

$$(\mathbf{X}\mathbf{\Lambda}\mathbf{X}^\mathsf{T} + \mathbf{S}) \begin{bmatrix} \Delta_\mathbf{v} \\ \Delta_\mathbf{w} \end{bmatrix} = -\mathbf{X}\mathbf{g} - \mathbf{S} \begin{bmatrix} \mathbf{v} \\ \mathbf{w} \end{bmatrix}. \tag{18}$$

The definitions of coefficients $\mathbf{\Lambda}$, $\mathbf{S}$, and $\mathbf{g}$—and the proof of the theorem—can be found in Appendix A.

Given the parameter $\mathbf{w}$, a test instance $\mathbf{x}$ is classified as $f(\mathbf{x};\mathbf{w}) = \text{sign}(\mathbf{w}^\mathsf{T}\mathbf{x})$.

5. Kernelized Learning Algorithm

We derive a kernelized version of the integrated classifier for differing training and test distributions. A transformation Φ maps instances into a target space in which a kernel function $k(\mathbf{x}_i,\mathbf{x}_j)$ calculates the inner product $\Phi(\mathbf{x}_i)^\mathsf{T}\Phi(\mathbf{x}_j)$. The update rule (Equation 18) thus becomes

$$(\Phi(\mathbf{X})\mathbf{\Lambda}\Phi(\mathbf{X})^\mathsf{T} + \mathbf{S}) \begin{bmatrix} \Delta_\mathbf{v} \\ \Delta_\mathbf{w} \end{bmatrix} = -\Phi(\mathbf{X})\mathbf{g} - \mathbf{S} \begin{bmatrix} \mathbf{v} \\ \mathbf{w} \end{bmatrix}. \tag{19}$$

$\Phi(\mathbf{X})$ is defined by

$$\Phi(\mathbf{X}) = \begin{bmatrix} \Phi(\mathbf{X}_L) & \Phi(\mathbf{X}_T) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \Phi(\mathbf{X}_L) \end{bmatrix}.$$

According to the Representer Theorem, the optimal separator is a linear combination of examples. Parameter vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ in the dual space weight the influence of all examples:

$$\begin{bmatrix}\mathbf{v}\\ \mathbf{w}\end{bmatrix} = \Phi(\mathbf{X})\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}.$$

Equation 19 can therefore be rewritten as Equation 20. We now multiply $\Phi(\mathbf{X})^\mathsf{T}$ from the left to both sides and obtain Equation 21. We replace all resulting occurrences of $\Phi(\mathbf{X})^\mathsf{T}\Phi(\mathbf{X})$ by the kernel matrix $\mathbf{K}$ and arrive at Equation 22; $\mathbf{S}$ is replaced by $\mathbf{S}'$ such that $\Phi(\mathbf{X})^\mathsf{T}\mathbf{S}\Phi(\mathbf{X}) = \Phi(\mathbf{X})^\mathsf{T}\Phi(\mathbf{X})\mathbf{S}'$, that is, $S'_{i,i} = \sigma_\mathbf{v}^{-2}$ for $i = 1,\ldots,m+n$ and $S'_{m+n+i,m+n+i} = \sigma_\mathbf{w}^{-2}$ for $i = 1,\ldots,m$. Equation 22 is satisfied when Equation 23 is satisfied. Equation 23 is the update rule for the dual Newton gradient descent.

$$(\Phi(\mathbf{X})\boldsymbol{\Lambda}\Phi(\mathbf{X})^\mathsf{T} + \mathbf{S})\Phi(\mathbf{X})\begin{bmatrix}\Delta_{\boldsymbol{\alpha}}\\ \Delta_{\boldsymbol{\beta}}\end{bmatrix} = -\Phi(\mathbf{X})\mathbf{g} - \mathbf{S}\Phi(\mathbf{X})\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}, \tag{20}$$

$$\Phi(\mathbf{X})^\mathsf{T}(\Phi(\mathbf{X})\boldsymbol{\Lambda}\Phi(\mathbf{X})^\mathsf{T} + \mathbf{S})\Phi(\mathbf{X})\begin{bmatrix}\Delta_{\boldsymbol{\alpha}}\\ \Delta_{\boldsymbol{\beta}}\end{bmatrix} = -\Phi(\mathbf{X})^\mathsf{T}\Phi(\mathbf{X})\mathbf{g} - \Phi(\mathbf{X})^\mathsf{T}\mathbf{S}\Phi(\mathbf{X})\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}, \tag{21}$$

$$(\mathbf{K}\boldsymbol{\Lambda}\mathbf{K} + \mathbf{K}\mathbf{S}')\begin{bmatrix}\Delta_{\boldsymbol{\alpha}}\\ \Delta_{\boldsymbol{\beta}}\end{bmatrix} = -\mathbf{K}\mathbf{g} - \mathbf{K}\mathbf{S}'\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}, \tag{22}$$

$$(\boldsymbol{\Lambda}\mathbf{K} + \mathbf{S}')\begin{bmatrix}\Delta_{\boldsymbol{\alpha}}\\ \Delta_{\boldsymbol{\beta}}\end{bmatrix} = -\mathbf{g} - \mathbf{S}'\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}. \tag{23}$$

Given the parameters, test instance $\mathbf{x}$ is classified by $f(\mathbf{x};\boldsymbol{\beta}) = \operatorname{sign}(\sum_{i=1}^m \beta_i k(\mathbf{x},\mathbf{x}_i))$.

6. Convexity Analysis and Solving the Optimization Problems

The following theorem specifies sufficient conditions for convexity of Optimization Problem 1. With this theorem we can easily check whether the integrated classifier for covariate shift is convex for specific models of the negative log-likelihood functions. The negative log-likelihood function $\ell_\mathbf{w}$ itself and its first and second derivatives are needed. Equations 31 and 32 in Appendix A define shorthand notation which we will use in the following.

Theorem 4 *Optimization Problem 1 is convex if the loss function $\ell_\mathbf{v}$ is convex and $\ell_\mathbf{w}$ is log-convex and non-negative. The log-convexity condition is equivalent to*

$$\ell_\mathbf{w}\ell''_\mathbf{w} - \ell'^2_\mathbf{w} \geq 0. \tag{24}$$

Proof Looking at Optimization Criterion 1 we immediately see that the regularizers are convex. If $\ell_\mathbf{v}$ is convex, the second term is convex as well. We therefore only need to analyze the convexity of the term

$$\sum_{i=1}^m \frac{p(s=1)}{p(s=-1)}\exp(-\mathbf{v}^\mathsf{T}\mathbf{x}_i)\ell_\mathbf{w}(y_i\mathbf{w}^\mathsf{T}\mathbf{x}_i).$$

A sum is convex if the single summands are convex. And a sufficient condition for convexity of a function is that it is non-negative and log-convex. The above expression is non-negative as $\ell_\mathbf{w}$ is non-negative. This means we only need to check whether

$$\log\frac{p(s=1)}{p(s=-1)} - \mathbf{v}^\mathsf{T}\mathbf{x}_i + \log\ell_{\mathbf{w},i}$$

is convex. The prior ratio is assumed to be constant. The second term is linear and therefore convex and the third term is the log-convexity condition of $\ell_{\mathbf{w}}$. The second derivative of $\log \ell_{\mathbf{w}}$ is

$$\ell_{\mathbf{w}}^{-1}\ell_{\mathbf{w}}'' + \ell_{\mathbf{w}}^{-2}\ell_{\mathbf{w}}'^2,$$

thus $\log \ell_{\mathbf{w}}$ is convex if $\ell_{\mathbf{w}}\ell_{\mathbf{w}}'' - \ell_{\mathbf{w}}'^2$ is non-negative. ■

In order to check Optimization Criterion 1 for convexity we need to choose models of the negative log-likelihood $\ell_{\mathbf{v}}$ and $\ell_{\mathbf{w}}$ and derive their first and second derivatives. These derivations are also needed to actually minimize Optimization Criterion 1 with the Newton update steps derived in the last section.

We use a logistic model $\ell_{\mathbf{v}}(s_i\mathbf{v}^\mathsf{T}\mathbf{x}) = \log(1+\exp(-s_i\mathbf{v}^\mathsf{T}\mathbf{x}))$; the abbreviations of Appendix A can now be expanded:

$$\ell'_{\mathbf{v},i}s_ix_{ij} = -\frac{\exp(-s_i\mathbf{v}^\mathsf{T}\mathbf{x}_i)}{1+\exp(-s_i\mathbf{v}^\mathsf{T}\mathbf{x}_i)}s_ix_{ij}; \quad \ell''_{\mathbf{v},i}x_{ij}x_{ik} = \frac{\exp(-s_i\mathbf{v}^\mathsf{T}\mathbf{x}_i)}{(1+\exp(-s_i\mathbf{v}^\mathsf{T}\mathbf{x}_i))^2}x_{ij}x_{ik}.$$

For the target classifier, we detail the derivations for logistic and for exponential models of $\ell_{\mathbf{w}}$. For the logistic model the derivatives of $\ell_{\mathbf{w}}$ are the same as for $\ell_{\mathbf{v}}$, only $\mathbf{v}$ needs to be replaced by $\mathbf{w}$ and s_i by y_i. For an exponential model with $\ell_{\mathbf{w}}(y_i\mathbf{w}^\mathsf{T}\mathbf{x}) = \exp(-y_i\mathbf{w}^\mathsf{T}\mathbf{x})$ the abbreviations are expanded as follows:

$$\ell'_{\mathbf{w},i}y_ix_{ij} = -\exp(-y_i\mathbf{w}^\mathsf{T}\mathbf{x}_i)y_ix_{ij}; \quad \ell''_{\mathbf{w},i}x_{ij}x_{ik} = \exp(-y_i\mathbf{w}^\mathsf{T}\mathbf{x}_i)x_{ij}x_{ik}.$$

Using Theorem 4 we can now easily check the convexity of the integrated classifier with logistic model and with exponential model for $\ell_{\mathbf{w}}$.

Corollary 5 *With a logistic model for $\ell_{\mathbf{w}}$, the condition of Equation 24 is violated and therefore Optimization Problem 1 with logistic model for $\ell_{\mathbf{w}}$ may not convex in general.*

Proof Inserting the logistic function into Equation 24 we get the following solution.

$$\ell_{\mathbf{w},i}\ell''_{\mathbf{w},i} - \ell'^2_{\mathbf{w},i} = \frac{\exp(-y_i\mathbf{w}^\mathsf{T}\mathbf{x}_i)}{(1+\exp(-y_i\mathbf{w}^\mathsf{T}\mathbf{x}))^2}\left(\log(1+\exp(-y_i\mathbf{w}^\mathsf{T}\mathbf{x})) - \exp(-y_i\mathbf{w}^\mathsf{T}\mathbf{x}_i)\right). \tag{25}$$

The fraction in Equation 25 is always positive, the difference term is always negative which violates the condition of Equation 24. ■

Empirically, we find that it is a good choice to select the parameters of a regular, *iid* logistic regression classifier as starting point for the Newton gradient search.

One can easily show that the condition of Equation 24 is violated when $\ell_{\mathbf{w}}$ is chosen as hinge loss or quadratic loss.

Corollary 6 *Optimization Problem 1 with exponential model for $\ell_{\mathbf{w}}$ is convex.*

Proof The exponential loss is non-negative and its logarithm is linear and therefore convex. ■

This means the global optimum of Optimization Problem 1 with exponential model for $\ell_{\mathbf{w}}$ can easily be found by Newton gradient descent.

7. Two-Stage Approximation to Integrated Model

The previous sections describe a complete solution to the learning problem under covariate shift. Optimization Problem 1 is convex for the exponential model; solving it using the efficient procedures derived in Sections 4 and 5 produces a globally optimal solution.

For the logistic model, unfortunately, convexity cannot be guaranteed. Furthermore, the regularized regression classifier is deeply embedded in Optimization Problem 1. It would not be easy to replace it by a different type of classifier such as, for instance, a decision tree. We will now discuss an approximation to Optimization Problem 1 which solves two consecutive optimization problems. The first optimization problem produces example-specific weights; the second step generates a classifier from the weighted examples. Both optimization problems are convex for exponential, logistic, and hinge loss as well as for many other loss functions. But most significantly, the two-stage approximation is conceptually simple: the second optimization step can be carried out by any learning procedure that is able to scale the loss incurred by each example using prescribed weight factors. Example-specific weights can easily be incorporated into virtually any learning method. Furthermore, as a result of the decomposition into two optimization problems parameter tuning becomes much easier because cross-validation can be used (cf. Section 9).

The derivation in Section 3.1 approximates the integral over $\mathbf{v}$ by simultaneously selecting a pair of values which maximize the posterior. This leads to the joint MAP hypothesis over $\mathbf{v}$ and $\mathbf{w}$. In the resulting optimization problem, $\mathbf{v}$ and $\mathbf{w}$ are free parameters. At a higher degree of approximation, one may factorize the posterior (Equation 26) and at first approximate the integral over $\mathbf{v}$ by the maximum of $p(\mathbf{v}|\mathbf{y},\mathbf{s},\mathbf{X})$ (Equations 29 and 30). Subsequently, the posterior over $\mathbf{w}$ is maximized given fixed parameters $\mathbf{v}_{\text{MAP}'}$ (Equations 27 and 28).

$$\begin{aligned}
\mathbf{w}^* &= \operatorname{argmax}_{\mathbf{w}} \int p(\mathbf{w},\mathbf{v}|\mathbf{y},\mathbf{s},\mathbf{X}) d\mathbf{v} \\
&= \operatorname{argmax}_{\mathbf{w}} \int p(\mathbf{w}|\mathbf{y},\mathbf{s},\mathbf{X};\mathbf{v}) p(\mathbf{v}|\mathbf{y},\mathbf{s},\mathbf{X}) d\mathbf{v} && (26)\\
&\approx \operatorname{argmax}_{\mathbf{w}} p(\mathbf{w}|\mathbf{y},\mathbf{s},\mathbf{X};\mathbf{v}_{\text{MAP}'}) && (27)\\
&= \operatorname{argmax}_{\mathbf{w}} p(\mathbf{y}|\mathbf{s},\mathbf{X};\mathbf{w},\mathbf{v}_{\text{MAP}'}) p(\mathbf{w}) && (28)\\
\text{with}\quad \mathbf{v}_{\text{MAP}'} &= \operatorname{argmax}_{\mathbf{v}} p(\mathbf{v}|\mathbf{y},\mathbf{s},\mathbf{X}) && (29)\\
&= \operatorname{argmax}_{\mathbf{v}} p(\mathbf{s}|\mathbf{y},\mathbf{X};\mathbf{v}) p(\mathbf{v}). && (30)
\end{aligned}$$

This results in two optimization problems. Only parameter $\mathbf{v}$ is free in the first stage (Optimization Problem 2). The test-to-training ratio (Equation 17) can be derived from the resulting value of $\mathbf{v}$.

Optimization Problem 2 *Over* $\mathbf{v}$*, minimize*

$$\sum_{i=1}^{m+n} \ell_{\mathbf{v}}(s_i \mathbf{v}^\mathsf{T} \mathbf{x}_i) + \frac{1}{2\sigma_{\mathbf{v}}^2} \mathbf{v}^\mathsf{T} \mathbf{v}.$$

In the second stage (Optimization Problem 3), the target model parameters $\mathbf{w}$ are optimized with constant parameters $\mathbf{v}$ and constant example weights. The parameters $\mathbf{v}$ are the result of Optimization Problem 2.

Optimization Problem 3 *Over* $\mathbf{w}$ *(*$\mathbf{v}$ *is constant), minimize*

$$\sum_{i=1}^{m} \frac{p(s=1)}{p(s=-1)} \exp(-\mathbf{v}^\mathsf{T} \mathbf{x}_i) \ell_{\mathbf{w}}(y_i \mathbf{w}^\mathsf{T} \mathbf{x}_i) + \frac{1}{2\sigma_{\mathbf{w}}^2} \mathbf{w}^\mathsf{T} \mathbf{w}.$$

The criterion of Optimization Problem 3 weights the loss $\ell_{\mathbf{w}}(y_i \mathbf{w}^\top \mathbf{x}_i)$ that each example incurs such that the sample is matched to the test distribution. The last term $\frac{1}{2\sigma_{\mathbf{w}}^2}\mathbf{w}^\top\mathbf{w}$ is the regularizer of the regression. Optimization Problem 3 can easily be adapted to virtually any type of classification mechanism by inserting the appropriate loss function $\ell_{\mathbf{w}}(y_i \mathbf{w}^\top \mathbf{x}_i)$ and regularizer. Operationally, an arbitrary classification procedure is applied to a sample that is either resampled from the training data according to sampling distribution $\frac{p(s=1)}{p(s=-1)}\exp(-\mathbf{v}^\top\mathbf{x}_i)$, or the classifier is applied to the training data with the example-specific loss scaled according to $\frac{p(s=1)}{p(s=-1)}\exp(-\mathbf{v}^\top\mathbf{x}_i)$.

8. Relationship to Kernel Mean Matching

Huang et al. (2007) motivate the kernel mean matching algorithm as a procedure that minimizes the distance between the means of unlabeled and weighted labeled data in feature space. If the kernel is universal this is equivalent to minimizing the difference of the distributions. We derive a new interpretation for kernel mean matching that shows its relation to Optimization Problem 2 and the above two-stage approximation.

Using a hinge loss for $\ell_{\mathbf{v}}(s_i(\mathbf{v}^\top\mathbf{x}_i + b))$ in Optimization Problem 2 and an explicit offset parameter b we obtain a regular support vector machine. The kernel matrix of this SVM is $\begin{bmatrix} \mathbf{K}_{(LL)} & \mathbf{K}_{(LT)} \\ \mathbf{K}_{(LT)}^\top & \mathbf{K}_{(TT)} \end{bmatrix}$ and the target variables are $s_i \in \{-1, 1\}$. An SVM can heuristically be simplified by setting the dual parameters α_i for the unlabeled examples to a fixed value $\frac{m}{n}$. This can be interpreted as a mixture between an SVM and a Rocchio classifier. The α_i corresponding to the labeled examples ($s_i = 1$) are trained with an SVM; setting α_i of all unlabeled examples ($s_i = -1$) to $\frac{m}{n}$ approximates the negative class (the unlabeled examples) by their centroid in feature space in accordance with the Rocchio classifier (Joachims, 1997).

The SVM optimization criterion with fixed $\alpha_i = \frac{m}{n}$ for examples with $s_i = -1$ is

$$\begin{aligned} \min_{\alpha_L} \quad & \tfrac{1}{2}\alpha_L^\top \mathbf{K}_{(LL)}\alpha_L - \tfrac{m}{n}\alpha_L^\top \mathbf{K}_{(LT)}\mathbf{1} + \tfrac{1}{2}\tfrac{m^2}{n^2}\mathbf{1}\mathbf{K}_{(TT)}\mathbf{1} - \alpha_L^\top\mathbf{1} - n\tfrac{m}{n} \\ & \text{subject to } \alpha_i \in \left[0, \sigma_{\mathbf{v}}^2\right] \text{ and } \textstyle\sum_{i=1}^m \alpha_i = \sum_{i=1}^n \frac{m}{n} = m; \end{aligned}$$

vector α_L denotes all elements α_i with $i = 1, \ldots, m$. We can drop the constant terms ($\alpha_L^\top\mathbf{1}$ is constant because of the second constraint) and arrive at Optimization Problem 4.

Optimization Problem 4

$$\min_{\alpha_L} \quad \tfrac{1}{2}\alpha_L^\top \mathbf{K}_{(LL)}\alpha_L - \tfrac{m}{n}\alpha_L^\top \mathbf{K}_{(LT)}\mathbf{1} \quad \text{subject to } \alpha_i \in \left[0, \sigma_{\mathbf{v}}^2\right] \text{ and } \alpha_L^\top\mathbf{1} = m.$$

This is the dual objective of kernel mean matching. The only difference is that Huang et al. (2007) relax the second constraint up to a small constant ε, their constraint is $m(1-\varepsilon) \leq \alpha_L^\top\mathbf{1} \leq m(1+\varepsilon)$. Empirically we find that setting ε to zero has no impact on the performance. The parameter $\sigma_{\mathbf{v}}^2$ corresponds to parameter B in Equation 2.

In order to solve the second stage (Optimization Problem 3), kernel mean matching does not use re-weighting factors $\frac{p(s=1)}{p(s=-1)}\exp(-\mathbf{v}^\top\mathbf{x}_i - b)$ but directly uses the dual α_i parameters as weights.

To sum up, kernel mean matching can be interpreted as a variant of Optimization Problem 2. It discriminates training against test examples using a partially Rocchio-style approximation to the SVM optimization criterion.

9. Parameter Tuning

Optimization Problem 1 relies on hyper-parameters $\sigma_{\mathbf{v}}^2$ and $\sigma_{\mathbf{w}}^2$ that need to be tuned. For the two-stage approximation of Section 7 and reference methods like kernel mean matching two similar parameters need to be specified. In addition to the regularization parameters kernel parameters need to be tuned for non-linear kernels. Parameter tuning for covariate shift models is much more difficult than for regular prediction models because in the covariate shift setting there is no labeled data available drawn from the test distribution. Parameter tuning by regular cross-validation on the labeled training data is inappropriate because the labeled training data is not governed by the test distribution.

In the following paragraphs we describe different tuning procedures; two procedures require prior knowledge and one does not require prior knowledge on the hyper-parameters. The tuning procedures with prior knowledge can be used for all described models. The one without prior knowledge cannot be used for kernel mean matching and the one-stage model of Optimization Problem 1.

A typical setting with prior knowledge on the hyper-parameters is when the difference between training and test data is introduced by a covariate shift over time and the input distribution shifts constantly over time. The most recent data is the unlabeled test data and the older data has been labeled and is the training data. In this setting the parameters can be tuned by splitting the labeled training data into two consecutive parts. The tuning models are learned on the part with earlier timestamps and the hyper-parameters $\sigma_{\mathbf{v}}^2$ and $\sigma_{\mathbf{w}}^2$ and kernel parameters are optimized on the part with later timestamps.

Another setting with prior knowledge is when in addition to the pair of training and test set an additional pair of training and fully labeled test set from a different domain with a similar magnitude of covariate shift is available. This additional set can be used to tune the parameters. Due to the similar magnitude of the covariate shift the optimal parameters for the additional domain are assumed to be a good choice for the parameters of the target domain.

For some two-stage models for covariate shift there is no prior knowledge necessary to tune hyper-parameters. Sugiyama et al. (2008) propose to tune the regularizer of the KLIEP model with cross-validation. In this manner the first stage parameter $\sigma_{\mathbf{v}}^2$ (and kernel parameters) of the two-stage model of Section 7 can be tuned as follows. The training and the test data are both split into training and tuning folds and the hold-out likelihood of the tuning folds is optimized with grid search on $\sigma_{\mathbf{v}}^2$ (and kernel parameters). The hold-out likelihood measures the predictive performance of the model $p(s|\mathbf{x};\mathbf{v})$ with respect to predicting the selector variable s of the hold out examples. Once the regularizer of the first stage is tuned, the second stage parameter $\sigma_{\mathbf{w}}^2$ (and kernel parameters) can be tuned with cross-validation on weighted training data (Sugiyama and Müller, 2005). The data of training folds as well as the data of tuning folds are weighted with the estimated training-to-test ratio.

Kernel mean matching does not provide out-of-sample predictions and it is therefore difficult to tune the regularization parameter B with cross-validation. The one-stage model of Optimization Problem 1 is also difficult to tune with cross-validation because there is a bidirectional influence between the parameters $\sigma_{\mathbf{v}}^2$ and $\sigma_{\mathbf{w}}^2$.

In order to compare the one-stage model and kernel mean matching to the other two-stage models we use tuning procedures based on prior knowledge in the empirical studies in the next section.

10. Empirical Results

We study the benefit of two versions of the integrated classifier for covariate shift and other reference methods on spam filtering, text classification, and landmine detection problems. The first integrated classifier uses a logistic model for $\ell_{\mathbf{w}}$ ("integrated log model"), the second an exponential model for $\ell_{\mathbf{w}}$ ("integrated exp model"); $\ell_{\mathbf{v}}$ is a logistic model in both cases.

The first baseline is a classifier trained under *iid* assumption with logistic $\ell_{\mathbf{w}}$. All other reference methods consist of a two-stage procedure: first, the difference between training and test distribution is estimated, the classifier is trained on weighted data in a second step. The second method is kernel mean matching (Huang et al., 2007); we set $\varepsilon = \sqrt{m-1}/\sqrt{m}$ as proposed by the authors. In the third method, separate density estimates for $p(\mathbf{x}|\lambda)$ and $p(\mathbf{x}|\theta)$ are obtained using kernel density estimation (Shimodaira, 2000), the bandwidth of the kernel is chosen according to the rule-of-thumb of Silverman (1986).

The last two reference methods rely on the two-stage approximation of Optimization Problems 2 and 3 with a logistic regression ("two-stage LR") and an exponential model classifier ("two-stage exp model") as their second stages. The example weights are computed according to Equation 17 using a logistic model in the first stage, $p(s=1|\mathbf{x};\mathbf{v})$ is estimated by training a logistic regression that discriminates training from test examples.

The baselines differ in the first stage, the second stage is based on a logistic regression classifier with weighted examples in all cases but the two-stage exponential model baseline. We use a maximum likelihood estimate of $\frac{m}{n}$ for $\frac{p(s=1)}{p(s=-1)}$. We use tuning procedures that rely on prior knowledge (cf. Section 9). Short descriptions of the respective tuning data can be found below. For all experiments we tune the regularization parameters of all methods (and the variance parameter of the RBF kernels for the landmine experiments) by maximizing AUC on the tuning set.

We use the spam filtering data of Bickel et al. (2007); the collection contains nine different inboxes with test emails (5270 to 10964 emails, depending on inbox) and one set of training emails compiled from various different sources. We use a fixed set of 1000 emails as training data. We randomly select between 32 and 2048 emails from one of the original inboxes. We repeat this process 10 times for 2048 test emails and 20 to 640 times for 1024 to 32 test emails. As tuning data we use the labeled emails from an additional inbox different from the test inboxes.

The performance measure is the rate by which the $1-\text{AUC}$ risk is reduced over the *iid* baseline (Bickel and Scheffer, 2007); it is computed as $1-\frac{1-\text{AUC}}{1-\text{AUC}_{iid}}$. We use linear kernels for all methods. We analyze the rank of the kernel matrix and find that it fulfills the universal kernel requirement of kernel mean matching; this is due to the high-dimensionality of the data.

Figure 1 (top row) shows the results for various numbers of unlabeled examples. The left column of Figure 1 compares the integrated classifiers for covariate shift to the kernel mean matching and kernel density estimation baselines. The right column compares the integrated classifiers (Optimization Problem 1) with the two-stage approximations (Optimization Problems 2 and 3). The results for a specific number of unlabeled examples are averaged over 10 to 640 random test samples and averaged over all nine inboxes. Averaged over all users and inbox sizes the absolute AUC of the *iid* classifier is 0.994. Error bars indicate standard errors of the $1-\text{AUC}$ risk.

The integrated and two-step logistic regression and exponential models and kernel mean matching perform similarly well. The differences to the *iid* baseline are highly significant. For 1048 examples the $1-\text{AUC}$ risk is even reduced by an average of 30% with the integrated exponential model classifier! The kernel density estimation procedure is not able to beat the *iid* baseline. The

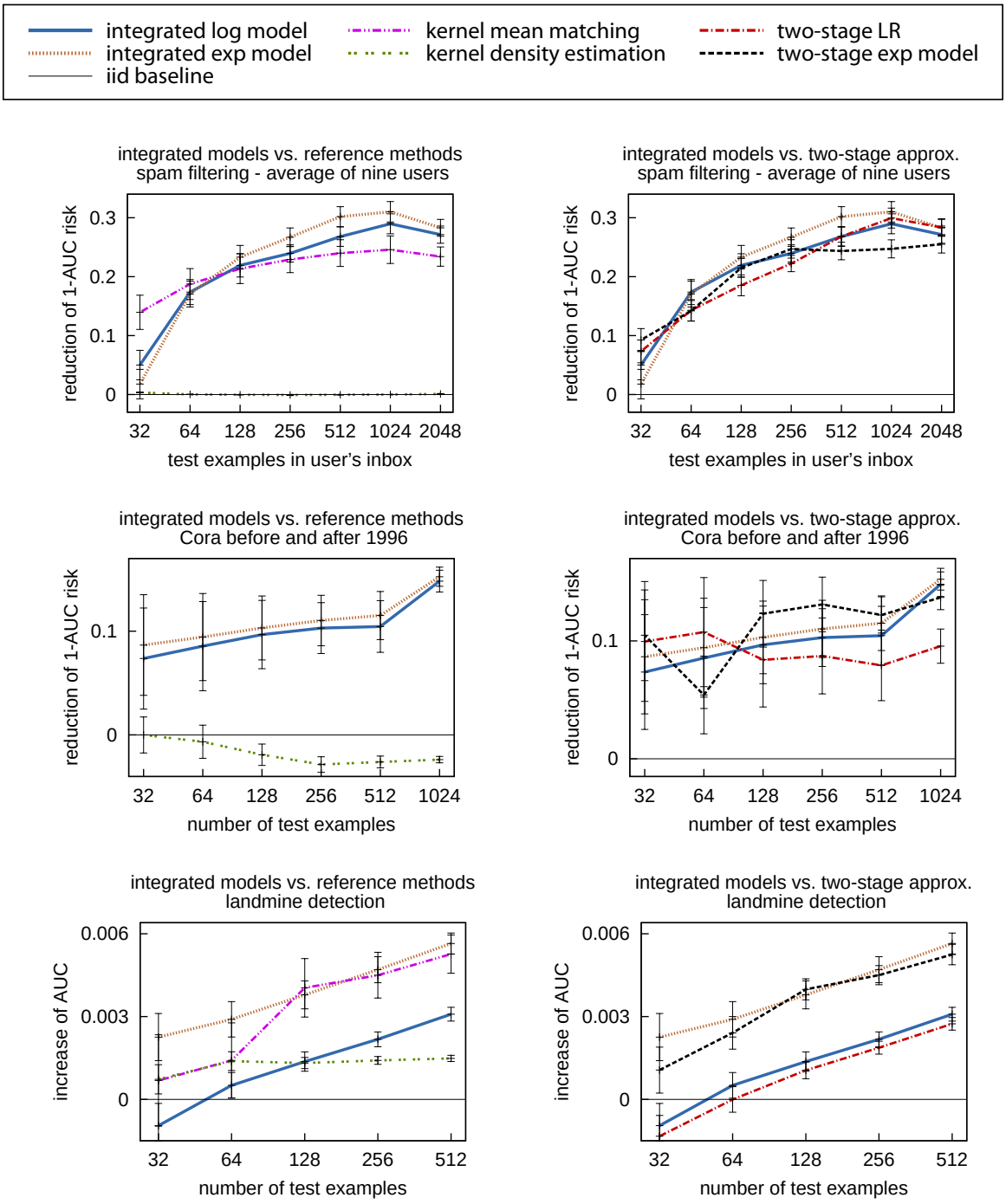

Figure 1: Average reduction of 1 – AUC risk over nine users for spam filtering (top row) and Cora *Machine Learning/Networking* classification before and after 1996 (second row) and average increase of AUC for landmine detection over 812 pairs of mine fields (bottom row) depending on the number of unlabeled test examples.

convex integrated exponential model performs slightly better than its two-stage approximation; for larger number of test examples (512 to 2048) this difference is statistically significant according to a paired t-test with significance level of 5%. For the logistic model, the two-stage optimization performs similarly well as the integrated version.

We now study text classification using computer science papers from the Cora data set. The task is to discriminate Machine Learning from Networking papers. We select 812 papers written before 1996 from both classes as training examples and 1285 papers written after 1996 as test examples. For parameter tuning we apply an additional time split on the training data; we train on the papers written before 1995 and tune on papers written 1995 (cf. Section 9).

Title and abstract are transformed into *tfidf* vectors, the number of distinct words is about 40,000. We again use linear kernels (rank analysis verifies the universal kernel property) and average the results over 20 to 640 random test samples for different sizes (1024 for 20 samples to 32 for 640 samples) of test sets. The resulting $1-\text{AUC}$ risk is shown in Figure 1 (second row). The average absolute AUC of the *iid* classifier is 0.998. The methods based on discriminative density estimates significantly outperform all other methods. Kernel mean matching is not displayed because its average performance lies far below the *iid* baseline. The integrated models reduce the $1-\text{AUC}$ risk by 15% for 1024 test examples.

In a third set of experiments we study the problem of detecting landmines using the data set of Xue et al. (2007). The collection contains data of 29 mine fields in different regions. Binary labels (landmine or safe ground) and nine dimensional feature vectors extracted from radar images are provided. There are about 500 examples for each mine field. Each of the fields has a distinct distribution of input patterns, varying from highly foliated to desert areas.

We enumerate all 29×28 pairs of mine fields, using one field as training and the other as test data. For tuning we hold out 4 of the 812 pairs. Results are increases over the *iid* baseline, averaged over all $29 \times 28 - 4$ combinations. We use RBF kernels with kernel width 0.3 for all methods. The results are displayed in Figure 1 (bottom row). The average absolute AUC of the *iid* baseline is 0.64 with a standard deviation of 0.07; note, that the error bars are much smaller than the absolute standard deviation because they indicate the standard error of the *differences* to the *iid* baseline.

For this problem, the exponential model classifiers and kernel mean matching significantly outperform all other methods on average. Considering only methods with logistic target model, kernel mean matching is better than all other methods. Integrated logistic regression and two-stage logistic regression are still significantly better than the *iid* baseline except for 32 and 64 test examples. The integrated classifiers are slightly better than the two-stage variants.

11. Conclusion

We derived a discriminative model for learning under differing training and test distributions. The contribution of each training instance to the optimization problem ideally needs to be weighted with its test-to-training density ratio. We show that this ratio can be expressed—without modeling either training or test density—by a discriminative model that characterizes how much more likely an instance is to occur in the test sample than it is to occur in the training sample.

We described a generative model whose parameters can be estimated with a joint MAP hypothesis of both the parameters of the test-to-training model and the final classifier. Optimizing these dependent parameters sequentially incurs an additional approximation compared to solving the joint optimization problem. We derived a primal and a kernelized Newton gradient descent procedure for

the joint optimization problem. Theorem 4 specifies the condition for the convexity of Optimization Problem 1. Checking the condition using popular loss functions as models of the negative log-likelihoods reveals that Optimization Problem 1 is convex with exponential loss.

We gave a new interpretation for kernel mean matching and show that it is also based on a discriminative model similar to Optimization Problem 2.

Empirically, we found that the integrated and the two-stage models as well as kernel mean matching outperform the *iid* baseline and the kernel density estimation model in almost all cases. In some cases, the integrated models perform slightly better than their two-stage counterparts. The performance of kernel mean matching depends on the problem; for one out of three problems it did not beat the *iid* baseline, for the others it yielded comparable results to the integrated models.

The two-stage model is conceptually simpler than the integrated model, and may in some cases have the greatest practical utility. The main advantage compared to the integrated model is that regularization parameters can be tuned without prior knowledge by cross-validation. Another advantage of the two-stage model is that in the second stage, after the example-specific weights have been derived, virtually any learning mechanism can be employed to produce the final classifier from the weighted training sample. This comes at the cost of only a marginal loss of performance compared to the integrated model.

Acknowledgments

We gratefully acknowledge support from the German Science Foundation DFG and from STRATO AG. We thank Google for supporting us with a Google Research Award. We also thank the anonymous reviewers for their helpful comments. We wish to thank Jiayuan Huang, Alex Smola, Arthur Gretton, Karsten Borgward, and Bernhard Schölkopf who provided their implementation of the kernel mean matching algorithm.

Appendix A. Newton Gradient Descent—Proof of Theorem 3

In this Appendix, we derive Newton gradient descent updates for Optimization Problem 1 and thereby prove Theorem 3. We abbreviate

$$\ell_{\mathbf{v},i} = \ell_{\mathbf{v}}(s_i \mathbf{v}^\mathsf{T} \mathbf{x}_i); \;\; \ell'_{\mathbf{v},i} s_i x_{ij} = \frac{\partial \ell_{\mathbf{v}}(s_i \mathbf{v}^\mathsf{T} \mathbf{x}_i)}{\partial v_j}; \;\; \ell''_{\mathbf{v},i} x_{ij} x_{ik} = \frac{\partial^2 \ell_{\mathbf{v}}(s_i \mathbf{v}^\mathsf{T} \mathbf{x}_i)}{\partial v_j v_k}; \tag{31}$$

$$\ell_{\mathbf{w},i} = \ell_{\mathbf{w}}(y_i \mathbf{w}^\mathsf{T} \mathbf{x}_i); \;\; \ell'_{\mathbf{w},i} y_i x_{ij} = \frac{\partial \ell_{\mathbf{w}}(y_i \mathbf{w}^\mathsf{T} \mathbf{x}_i)}{\partial w_j}; \;\; \ell''_{\mathbf{w},i} x_{ij} x_{ik} = \frac{\partial^2 \ell_{\mathbf{w}}(y_i \mathbf{w}^\mathsf{T} \mathbf{x}_i)}{\partial w_j w_k}; \tag{32}$$

$$\omega_i = \frac{p(s=1)}{p(s=-1)} \exp(-\mathbf{v}^\mathsf{T} \mathbf{x}_i)$$

and denote the objective function of Optimization Problem 1 by

$$F(\mathbf{v}, \mathbf{w}) \;\; = \;\; \sum_{i=1}^{m} \omega_i \ell_{\mathbf{w},i} + \sum_{i=1}^{m+n} \ell_{\mathbf{v},i} + \frac{1}{2\sigma_{\mathbf{w}}^2} \mathbf{w}^\mathsf{T} \mathbf{w} + \frac{1}{2\sigma_{\mathbf{v}}^2} \mathbf{v}^\mathsf{T} \mathbf{v}.$$

We compute the gradient with respect to $\mathbf{v}$ and $\mathbf{w}$.

$$\begin{aligned}\frac{\partial F(\mathbf{v},\mathbf{w})}{\partial v_j} &= -\sum_{i=1}^{m}\omega_i\ell_{\mathbf{w},i}x_{ij}+\sum_{i=1}^{m+n}\ell'_{\mathbf{v},i}s_ix_{ij}+\frac{1}{\sigma_{\mathbf{v}}^2}v_j\\ \frac{\partial F(\mathbf{v},\mathbf{w})}{\partial w_j} &= \sum_{i=1}^{m}\omega_i\ell'_{\mathbf{w},i}y_ix_{ij}+\frac{1}{\sigma_{\mathbf{w}}^2}w_j.\end{aligned}$$

The Hessian is the matrix of second derivatives.

$$\begin{aligned}\frac{\partial^2 F(\mathbf{v},\mathbf{w})}{\partial v_j\partial v_k} &= \sum_{i=1}^{m}\omega_i\ell_{\mathbf{w},i}x_{ij}x_{ik}+\sum_{i=1}^{m+n}\ell''_{\mathbf{v},i}x_{ij}x_{ik}+\frac{1}{\sigma_{\mathbf{v}}^2}\delta_{jk}\\ \frac{\partial^2 F(\mathbf{v},\mathbf{w})}{\partial v_j\partial w_k} &= -\sum_{i=1}^{m}\omega_i\ell'_{\mathbf{w},i}y_ix_{ij}x_{ik}\\ \frac{\partial^2 F(\mathbf{v},\mathbf{w})}{\partial w_j\partial w_k} &= \sum_{i=1}^{m}\omega_i\ell''_{\mathbf{w},i}x_{ij}x_{ik}+\frac{1}{\sigma_{\mathbf{w}}^2}\delta_{jk}.\end{aligned}$$

We can rewrite the gradient as $\mathbf{X}\mathbf{g}+\mathbf{S}\begin{bmatrix}\mathbf{v}\\ \mathbf{w}\end{bmatrix}$ and the Hessian as $\mathbf{X}\mathbf{\Lambda}\mathbf{X}^\mathsf{T}+\mathbf{S}$ using the following definitions, where d is the dimensionality of $\mathbf{X}_T$ and $\mathbf{X}_L$.

$$\begin{aligned} g_i &= -\omega_i\ell_{\mathbf{w},i}+\ell'_{\mathbf{v},i} && \text{for} \quad i=1,\ldots,m;\\ g_{m+i} &= -\ell'_{\mathbf{v},m+i} && \text{for} \quad i=1,\ldots,n;\\ g_{m+n+i} &= \omega_i\ell'_{\mathbf{w},i}y_i && \text{for} \quad i=1,\ldots,m;\\ S_{i,i} &= \sigma_{\mathbf{v}}^{-2} && \text{for} \quad i=1,\ldots,d;\\ S_{d+i,d+i} &= \sigma_{\mathbf{w}}^{-2} && \text{for} \quad i=1,\ldots,d;\end{aligned}$$

$$\mathbf{\Lambda}=\begin{bmatrix}\operatorname*{diag}_{i=1,\ldots,m}\left(\omega_i\ell_{\mathbf{w},i}+\ell''_{\mathbf{v},i}\right) & \mathbf{0} & -\operatorname*{diag}_{i=1,\ldots,m}\left(\omega_i\ell'_{\mathbf{w},i}y_i\right)\\ \mathbf{0} & \operatorname*{diag}_{i=1,\ldots,n}\left(\ell''_{\mathbf{v},m+i}\right) & \mathbf{0}\\ -\operatorname*{diag}_{i=1,\ldots,m}\left(\omega_i\ell'_{\mathbf{w},i}y_i\right) & \mathbf{0} & \operatorname*{diag}_{i=1,\ldots,m}\left(\omega_i\ell''_{\mathbf{w},i}\right)\end{bmatrix}.$$

The update step for the Newton gradient descent minimization of Optimization Problem 1 is $[\mathbf{v}',\mathbf{w}']^\mathsf{T}\leftarrow[\mathbf{v},\mathbf{w}]^\mathsf{T}+[\Delta_\mathbf{v},\Delta_\mathbf{w}]^\mathsf{T}$ with

$$(\mathbf{X}\mathbf{\Lambda}\mathbf{X}^\mathsf{T}+\mathbf{S})\begin{bmatrix}\Delta_\mathbf{v}\\ \Delta_\mathbf{w}\end{bmatrix}=-\mathbf{X}\mathbf{g}-\mathbf{S}\begin{bmatrix}\mathbf{v}\\ \mathbf{w}\end{bmatrix}.$$

References

S. Bickel and T. Scheffer. Dirichlet-enhanced spam filtering based on biased samples. In *Advances in Neural Information Processing Systems*, 2007.

S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the International Conference on Machine Learning*, 2007.

C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh. Sample selection bias correction theory. In *Proceedings of the International Conference on Algorithmic Learning Theory*, 2008.

M. Dudik, R. Schapire, and S. Phillips. Correcting sample selection bias in maximum entropy density estimation. In *Advances in Neural Information Processing Systems*, 2005.

C. Elkan. The foundations of cost-sensitive learning. In *Proceedings of the International Joint Conference on Artificial Intellligence*, 2001.

J. Heckman. Sample selection bias as a specification error. *Econometrica*, 47:153–161, 1979.

J. Huang, A. Smola, A. Gretton, K. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, 2007.

N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6:429–449, 2002.

T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, 1997.

J. Lunceford and M. Davidian. Stratification and weighting via the propensity score in estimation of causal treatment effects: a comparative study. *Statistics in Medicine*, 23(19):2937–2960, 2004.

C. Manski and S. Lerman. The estimation of choice probabilities from choice based samples. *Econometrica*, 45(8):1977–1988, 1977.

R. Prentice and R. Pyke. Logistic disease incidence models and case-control studies. *Biometrika*, 66(3):403–411, 1979.

P. Rosenbaum and D. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, 2000.

B. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.

M. Sugiyama and K.-R. Müller. Input-dependent estimation of generalization error under covariate shift. *Statistics and Decision*, 23(4):249–279, 2005.

M. Sugiyama, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems*, 2008.

J. Tsuboi, H. Kashima, S. Hido, S. Bickel, and M. Sugiyama. Direct density ratio estimation for large-scale covariate shift adaptation. In *Proceedings of the SIAM International Conference on Data Mining*, 2008.

Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with Dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the International Conference on Machine Learning*, 2004.