Discriminative Learning for Differing Training and Test Distributions

Steffen Bickel BICKEL@MPI-INF.MPG.DE
Michael Brückner BRUM@MPI-INF.MPG.DE
Tobias Scheffer SCHEFFER@MPI-INF.MPG.DE
Max Planck Institute for Computer Science, Saarbrücken, Germany

Abstract

We address classification problems for which the training instances are governed by a 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. We formulate the general problem of learning under covariate shift as an integrated optimization problem. We derive a kernel logistic regression classifier for differing training and test distributions.

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 (*e.g.,* the TREC spam corpus); image processing systems may be deployed to foreign countries 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 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 arbitrarily 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 heuristic of first estimating the bias of the training sample, and then learning the classifier on a weighted version of the training sample. We study the convexity of the integrated optimization problem. We derive a Newton gradient descent procedure, leading to a kernel logistic regression classifier for covariate shift.

After formalizing the problem setting in Section 2, we review models for differing training and test distributions in Section 3. Section 4 introduces our discriminative model and the joint optimization problem. We derive the logistic regression classifier for differing training and test distributions in Section 5 and report on experimental results in Section 6. Section 7 concludes.

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

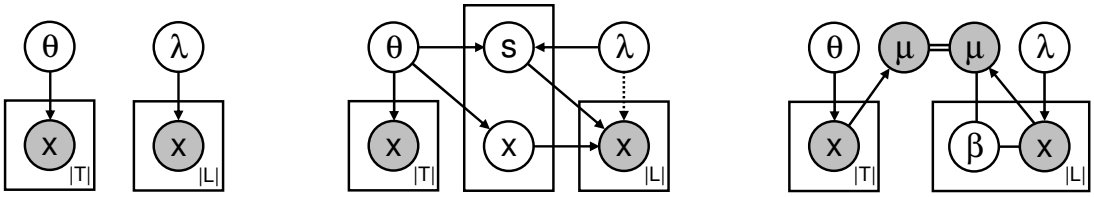

Figure 1. Learning with differing training and test distribution. Left: Learning under covariate shift by estimating training and test densities separately. Center: Generative model of sample selection bias. Right: Kernel mean matching.

2. Problem Setting

In the *covariate shift* problem setting, a labeled training sample $L = \langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\rangle$ is available. This training sample is governed by an unknown distribution $p(\mathbf{x}|\lambda)$, labels are drawn according to an unknown target concept $p(y|\mathbf{x})$. In addition, an unlabeled test set $T = \langle\mathbf{x}_{m+1}, \ldots, \mathbf{x}_{m+n}\rangle$ becomes available. The test set 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 $p(y|\mathbf{x})$.

The goal is to find a classifier $f : \mathbf{x} \mapsto y$ and to predict the missing labels $y_{m+1}, \ldots, y_{m+n}$ for the test instances. From a purely transductive perspective, the classifier can even be seen as an auxiliary step and may be discarded after the labels $y_{m+1}, \ldots, y_{m+n}$ have been conceived. The classifier should in any case perform well on the test data; that is, it should minimize some loss function $E_{(\mathbf{x},y)\sim\theta}[\ell(f(\mathbf{x}), y)]$ that is defined with respect to the unknown test distribution $p(\mathbf{x}|\theta)$.

Note that directly training f on the training data L would minimize the loss with respect to $p(\mathbf{x}|\lambda)$. The minimum of this optimization problem will not generally coincide with the minimal loss on $p(\mathbf{x}|\theta)$.

3. 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. 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* $p(\mathbf{x}, y|\bar{\theta}) = p(\mathbf{x}, y|\lambda)\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ *equals the expected loss with respect to* $p(\mathbf{x}, y|\theta)$*, when the target distribution* $p(y|\mathbf{x})$ *is the same under training and test distribution.*

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.

Both, $p(\mathbf{x}|\theta)$ and $p(\mathbf{x}|\lambda)$ are unknown, but $p(\mathbf{x}|\theta)$ is reflected in T, as is $p(\mathbf{x}|\lambda)$ in L. Shimodaira (2000) and Sugiyama and Müller (2005) propose that estimates $\hat{p}(\mathbf{x}|\theta)$ and $\hat{p}(\mathbf{x}|\lambda)$ be obtained from the test and training data, respectively, using kernel density estimation. In a second step, the estimated density ratio is used to resample the training instances, or to train with weighted examples (Figure 1, left hand side).

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.

This approach can handle gaps in either distribution. When a gap occurs in the test distribution, the weighting factor is simply $\frac{\hat{p}(\mathbf{x}|\theta)}{\hat{p}(\mathbf{x}|\lambda)} = 0$. When a gap occurs in the training distribution, this would lead to a zero denominator—but the ratio is only evaluated for the instances that actually occur in L. Clearly, these instances have a positive $p(\mathbf{x}|\lambda)$.

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). These results do not require the training and test distributions to be estimated. Sample selection bias relies on a model of the data generation process, illustrated in Figure 1, center. 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 s then decides whether $\mathbf{x}$ is moved into the training set ($s = 1$) or is discarded ($s = 0$). The distribution of the selector variable maps the test onto the training distribution:

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

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

Proposition 2 *The expected loss on* $p(\mathbf{x}, y|\bar{\theta}) \propto p(\mathbf{x}, y|\lambda)\frac{1}{p(s=1|\mathbf{x},\theta,\lambda)}$ *equals the expected loss on* $p(\mathbf{x}, y|\theta)$*, when* $p(s|\mathbf{x}, \theta, \lambda)$ *satisfies Equation 1.*

By intuition, the case of a gap in the test distribution ($p(\mathbf{x}|\theta) = 0$) for an instance with positive training density $p(\mathbf{x}|\lambda) > 0$ should not inflict any damage. Training instances that have no likelihood under the test distribution should simply be weighted with a factor of zero. The sample selection bias model, however, cannot handle this case because the training distribution is assumed to be sampled from the test distribution. Hence, $p(s = 1|\mathbf{x}, \theta, \lambda) \propto \frac{p(\mathbf{x}|\lambda)}{p(\mathbf{x}|\theta)} = \frac{p(\mathbf{x}|\lambda)}{0}$. In practice, this problem can be worked-around. But it indicates that the sample selection bias model is inadequate when gaps can occur in the test data.

When the model is implemented, $p(s = 1|\mathbf{x}, \theta, \lambda)$ is learned from the training and test data. The training data serve as examples that have been selected into the training sample ($s = 1$). But no discarded instances ($s = 0$) are available. The test instances have been drawn directly from $p(\mathbf{x}|\theta)$, selector s has not been instantiated. In practice (Bickel & Scheffer, 2007), an estimate of $p(s = 1|\mathbf{x}, \theta, \lambda)$ is determined by discriminating the training data ($s = 1$) from the test data (unknown s, treated as $s = 0$).

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); Japkowicz and Stephen (2002) investigate the case of training data that is only biased with respect to the class ratio.

Huang et al. (2007) devise the nonparametric kernel mean matching method for learning from differing training and test distributions. It finds weights for the training instances such that the first momentum of training and test sets – *i.e.,* their mean value – matches. $\Phi(\cdot)$ is a mapping into a feature space and B is a regularization parameter. Huang et al. derive a quadratic program from Equation 2 that can be solved with standard optimization tools.

$$\min_{\beta} \left|\left| \frac{1}{m}\sum_{i=1}^{m} \beta_i \Phi(\mathbf{x}_i) - \frac{1}{n}\sum_{i=m+1}^{m+n} \Phi(\mathbf{x}_i) \right|\right|^2 \quad (2)$$
$$\text{subject to } \beta_i \in [0, B] \text{ and } \left|\frac{1}{m}\sum_{i=1}^{m} \beta_i - 1\right| \leq \epsilon$$

Figure 1 summarizes different existing approaches for learning with differing training and test distributions.

4. Discriminative Learning for Differing Distributions

In this section, we derive a purely discriminative model that directly estimates weights for the training instances. No distributions over instances are modeled explicitly. Figure 2 illustrates the model. For each element $\mathbf{x}$ of the training set, selector variable $\sigma = 1$ indicates that it has been drawn into L. For each $\mathbf{x}$ in the test data, $\sigma = 0$ indicates that it has been drawn into the test set. The probability $p(\sigma = 1|\mathbf{x}, \theta, \lambda)$ has the following intuitive meaning: Given that an instance $\mathbf{x}$ has been drawn at random from the bag $L \cup T$ of training and test set; the probability that $\mathbf{x}$ originates from L is $p(\sigma = 1|\mathbf{x}, \theta, \lambda)$. Hence, the value of σ is observable for all training ($\sigma = 1$) and test ($\sigma = 0$) instances. The dependency between the instances and σ is undirected; neither training nor test set are assumed to be generated from the other sample.

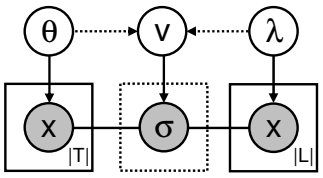

Figure 2. Discriminative model for learning with differing training and test distributions.

In the following equations we will derive an expression for $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ using the selector variable σ. In Equation 3 two prior probabilities are introduced that cancel each other out. The prior probabilities reflect the ratio of the number of labeled to unlabeled examples. Equations 4 to 6 follow from arithmetics. In order to understand the equality of 6 and 7, observe that Bayes' rule, applied to the term $p(\sigma = 1|\mathbf{x}, \theta, \lambda)$ found in 7, yields $p(\sigma = 1|x, \theta, \lambda) \propto p(\sigma = 1|\theta, \lambda)p(x|\sigma = 1, \theta, \lambda)$.

Training instances ($\sigma = 1$) are governed by λ; they are conditionally independent of θ given λ. Hence, $p(x|\sigma = 1, \theta, \lambda) = p(\mathbf{x}|\lambda)$. For test instances ($\sigma = 0$), equality $p(x|\sigma = 0, \theta, \lambda) = p(x|\theta)$ holds accordingly. The conditional $p(\sigma = 1|\mathbf{x}, \theta, \lambda)$ discriminates training ($\sigma = 1$) against test instances ($\sigma = 0$).

$$\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)} = \frac{p(\sigma = 1|\theta, \lambda)}{p(\sigma = 0|\theta, \lambda)} \frac{p(\sigma = 0|\theta, \lambda)}{p(\sigma = 1|\theta, \lambda)} \frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)} \quad (3)$$
$$= \frac{p(\sigma = 1|\theta, \lambda)}{p(\sigma = 0|\theta, \lambda)} \left(1 + \frac{p(\sigma = 0|\theta, \lambda)p(\mathbf{x}|\theta)}{p(\sigma = 1|\theta, \lambda)p(\mathbf{x}|\lambda)} - 1\right) \quad (4)$$
$$= \frac{p(\sigma = 1|\theta, \lambda)}{p(\sigma = 0|\theta, \lambda)} \left(\frac{1}{\frac{1}{1+\frac{p(\sigma=0|\theta,\lambda)p(\mathbf{x}|\theta)}{p(\sigma=1|\theta,\lambda)p(\mathbf{x}|\lambda)}}} - 1\right) \quad (5)$$
$$= \frac{p(\sigma = 1|\theta, \lambda)}{p(\sigma = 0|\theta, \lambda)} \left(\frac{1}{\frac{p(\sigma=1|\theta,\lambda)p(\mathbf{x}|\lambda)}{p(\sigma=1|\theta,\lambda)p(\mathbf{x}|\lambda)+p(\sigma=0|\theta,\lambda)p(\mathbf{x}|\theta)}} - 1\right) \quad (6)$$
$$= \frac{p(\sigma = 1|\theta, \lambda)}{p(\sigma = 0|\theta, \lambda)} \left(\frac{1}{p(\sigma = 1|\mathbf{x}, \theta, \lambda)} - 1\right) \quad (7)$$

The above model leaves us with the problem of learning a model $p(\sigma = 1|\mathbf{x}, \mathbf{v})$ of $p(\sigma = 1|\mathbf{x}, \theta, \lambda)$; according to Equation 7, this will then provide us with a weight

for each example. Finally, a classifier has to be learned from the weighted data. An obvious approach is to exercise these steps sequentially; all existing work on learning from biased samples has followed this idea of solving two optimization problems sequentially.

It should be noted that solving two subsequent optimization problems is an ad-hoc approach to the actual problem at hand. The integrated learning problem is to find parameters $\mathbf{v}$ for a model $p(\sigma = 1|\mathbf{x}, \mathbf{v})$ of $p(\sigma = 1|\mathbf{x}, \theta, \lambda)$ *and* parameters for $f(\mathbf{x}; \mathbf{w})$, the final classifier. We will therefore solve the problem of finding the entire vector of MAP parameters: $[\mathbf{w}, \mathbf{v}]^\mathsf{T}_{MAP} = \text{argmax}_{\mathbf{w},\mathbf{v}}\ p(\mathbf{w}, \mathbf{v}|L, T)$. Equation 8 applies the chain rule, 9 exploits that the classifier $\mathbf{w}$ is independent of the test data T, given L and the covariate shift $\mathbf{v}$. Equation 10 applies Bayes rule twice and exploits that $\mathbf{w}$ is independent of $\mathbf{v}$.

Equation 10 shows that the posterior can be factorized into the likelihood of the training data given the model parameters $P(L|\mathbf{w}, \mathbf{v})$, the likelihood of the observed selection variables σ – written $P(L, T|\mathbf{v})$ –, and the priors on the model parameters. Since the goal is discriminative training, the likelihood of the training data $P(L|\mathbf{w}, \mathbf{v})$ is resolved as just the conditional likelihood of the class values y_i over the training instances in Equation 11—this is standard in logistic regression and SVMs. We now use the weighted likelihood expression of Manski and Lerman (1977). Intuitively, $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ dictates how many times, on average, $\mathbf{x}$ should occur in L if 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 $p(y|\mathbf{x}, \mathbf{w})^{\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}}$.

Resolving $\frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\lambda)}$ according to Equation 7 leads to the first factor of Equation 11. In Equation 12, constants $p(\sigma=1|\mathbf{v})$ and $p(\sigma=0|\mathbf{v})$ are drawn out of the product.

$$p(\mathbf{w}, \mathbf{v}|L, T) = p(\mathbf{w}|\mathbf{v}, L, T)p(\mathbf{v}|L, T) \tag{8}$$

$$= p(\mathbf{w}|\mathbf{v}, L)p(\mathbf{v}|L, T) \tag{9}$$

$$\propto P(L|\mathbf{w}, \mathbf{v})P(L, T|\mathbf{v})p(\mathbf{w})p(\mathbf{v}) \tag{10}$$

$$= \left(\prod_{i=1}^{m} P(y_i|\mathbf{x}_i; \mathbf{w})^{\frac{p(\sigma=1|\mathbf{v})}{p(\sigma=0|\mathbf{v})}\left(\frac{1}{p(\sigma_i=1|\mathbf{x}_i;\mathbf{v})}-1\right)}\right) \left(\prod_{i=1}^{m+n} P(\sigma_i|\mathbf{x}_i; \mathbf{v})\right) p(\mathbf{w})p(\mathbf{v}) \tag{11}$$

$$= \left(\prod_{i=1}^{m} P(y|\mathbf{x}_i; \mathbf{w})^{\frac{1}{p(\sigma_i=1|\mathbf{x}_i;\mathbf{v})}-1}\right)^{\frac{p(\sigma=1|\mathbf{v})}{p(\sigma=0|\mathbf{v})}} \left(\prod_{i=1}^{m} P(\sigma_i = 1|\mathbf{x}_i; \mathbf{v}) \prod_{i=m+1}^{m+n} P(\sigma_i = 0|\mathbf{x}_i; \mathbf{v})\right) p(\mathbf{w})p(\mathbf{v}) \tag{12}$$

Term $P(L, T|\mathbf{v})$ is the likelihood of the selector variables σ_i, given parameter $\mathbf{v}$ of the bias model. It is a product over all training and test instances (Equation 11) and can be spelled out as in Equation 12. Equation 12 gives the joint optimization criterion that has to be maximized in order to find MAP parameters $[\mathbf{v}, \mathbf{w}]^\mathsf{T}$. The first factor corresponds to the likelihood of the classifier's parameters $\mathbf{w}$, the second term to the model of the discrepancy between training and test distribution. The factors interact by means of $\mathbf{v}$ which occurs in both terms. First modeling the discrepancy between training and test distribution (*i.e.,* choosing $\mathbf{v}$ to maximize the second factor) and then training the classifier (choosing $\mathbf{w}$ to maximize the first factor for fixed $\mathbf{v}$) is a plausible heuristic, but it is not generally optimal.

Out of curiosity, let us briefly consider the extreme case of disjoint training and test distributions; *i.e.,* $p(\mathbf{x}|\theta)p(\mathbf{x}|\lambda) = 0$ for all $\mathbf{x}$. In this case, the second factor is maximized by a $\mathbf{v}$ that assigns $p(\sigma = 1|\mathbf{x}; \mathbf{v}) = 1$ for all elements of L (subject to a possible regularization imposed by $p(\mathbf{v})$). Hence, the likelihood of the training data $p(y|\mathbf{x}, \mathbf{w})^{\frac{1}{1}-1}$ equals 1 for all possible classifiers $\mathbf{w}$. The choice of the classifier $\mathbf{w}$ is thus determined solely by the inductive bias $p(\mathbf{w})$. This result makes perfect sense because the training sample contains no information about the test distribution.

5. Kernel Logistic Regression Classifier

We will now use the logistic function as a model of both the likelihood of y given $\mathbf{x}$, and of the value of the selector variable σ given $\mathbf{x}$:

$$p(y = 1|\mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{w}^\mathsf{T}\mathbf{x})} \tag{13}$$

$$p(\sigma = 1|\mathbf{x}; \mathbf{v}) = \frac{1}{1 + \exp(-\mathbf{v}^\mathsf{T}\mathbf{x})}. \tag{14}$$

In addition, we impose the usual exponential prior on the parameters:

$$p(\mathbf{w}) \propto \exp\left(\frac{-\mathbf{w}^\mathsf{T}\mathbf{w}}{2s^2_\mathbf{w}}\right); \quad p(\mathbf{v}) \propto \exp\left(\frac{-\mathbf{v}^\mathsf{T}\mathbf{v}}{2s^2_\mathbf{v}}\right). \tag{15}$$

By maximizing the joint posterior (Equation 12) over the parameter vector $[\mathbf{w}, \mathbf{v}]^\mathsf{T}$, we will arrive at a logistic regression classifier for covariate shift.

Optimization Problem 1 *Over all* $\mathbf{w}$*, and* $\mathbf{v}$*, maximize* $p(\mathbf{w}, \mathbf{v}|L, T)$ *(as given in Equation 12), where* $p(y = 1|\mathbf{x}, \mathbf{w})$ *and* $p(\sigma = 1|\mathbf{x}, \mathbf{v})$ *are specified in Equations 13 and 14 and* $p(\mathbf{w})$ *and* $p(\mathbf{v})$ *in Equation 15.*

5.1. Primal Logistic Regression

We derive a Newton gradient method that directly maximizes Optimization Problem 1 in the attribute space. To this end, we need to derive the gradient $\frac{\partial p(\mathbf{w},\mathbf{v}|L,T)}{\partial \mathbf{w}}$ and $\frac{\partial p(\mathbf{w},\mathbf{v}|L,T)}{\partial \mathbf{v}}$ and the Hessian — the second derivatives of $p(\mathbf{w},\mathbf{v}|L,T)$. It consists of $\frac{\partial^2 p(\mathbf{w},\mathbf{v}|L,T)}{\partial^2 \mathbf{w}}$, $\frac{\partial^2 p(\mathbf{w},\mathbf{v}|L,T)}{\partial \mathbf{w}\partial \mathbf{v}}$, and $\frac{\partial^2 p(\mathbf{w},\mathbf{v}|L,T)}{\partial^2 \mathbf{v}}$.

The update rule assumes the form of a set of linear equations that have to be solved for the update vector $[\Delta_\mathbf{w}, \Delta_\mathbf{v}]^\mathsf{T}$. It depends on the current parameters $[\mathbf{w},\mathbf{v}]^\mathsf{T}$, 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

$$\mathbf{X} = \begin{bmatrix} \mathbf{L}^\mathsf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}^\mathsf{T} \\ \mathbf{0} & \mathbf{T}^\mathsf{T} \end{bmatrix},$$

where $\mathbf{L}$ and $\mathbf{T}$ are the matrices of training vectors, and test vectors respectively.

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

$$(\mathbf{X}^\mathsf{T}\Lambda\mathbf{X} + \mathbf{S}) \begin{bmatrix} \Delta_\mathbf{w} \\ \Delta_\mathbf{v} \end{bmatrix} = \mathbf{X}^\mathsf{T}\mathbf{e} - \mathbf{S} \begin{bmatrix} \mathbf{w} \\ \mathbf{v} \end{bmatrix}. \tag{16}$$

The definitions of coefficients Λ, $\mathbf{e}$, *and* $\mathbf{S}$ *– 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 $f(\mathbf{x};\mathbf{w}) = \text{argmax}_y p(y|\mathbf{x};\mathbf{w})$ (Equation 13).

5.2. Kernel Logistic Regression

We derive a kernelized version of the logistic regression classifier for differing training and test distributions. A transformation Φ maps instances into a target space in which 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 16) thus becomes

$$(\Phi(\mathbf{X})^\mathsf{T}\Lambda\Phi(\mathbf{X}) + \mathbf{S}) \begin{bmatrix} \Delta_\mathbf{w} \\ \Delta_\mathbf{v} \end{bmatrix} = \Phi(\mathbf{X})^\mathsf{T}\mathbf{e} - \mathbf{S} \begin{bmatrix} \mathbf{w} \\ \mathbf{v} \end{bmatrix} \tag{17}$$

According to the Representer Theorem, the optimal separator is a linear combination of examples. Parameter vectors ω and ν in the dual space weight the influence of all examples:

$$\begin{bmatrix} \mathbf{w} \\ \mathbf{v} \end{bmatrix} = \Phi(\mathbf{X})^\mathsf{T} \begin{bmatrix} \omega \\ \nu \end{bmatrix}.$$

Equation 17 can therefore be rewritten as Equation 18. We now multiply $\Phi(\mathbf{X})$ from the left to both sides in Equation 19. We replace all resulting occurrences of $\Phi(\mathbf{X})\Phi(\mathbf{X})^\mathsf{T}$ by the kernel matrix $\mathbf{K}$ and arrive at Equation 20; the details of the coefficients $\mathbf{S}'$ can be found in Appendix A. Equation 20 is satisfied when Equation 21 is satisfied. Equation 21 is the update rule for the dual Newton gradient descent.

$$\begin{aligned} &(\Phi(\mathbf{X})^\mathsf{T}\Lambda\Phi(\mathbf{X}) + \mathbf{S})\Phi(\mathbf{X})^\mathsf{T} \begin{bmatrix} \Delta_\omega \\ \Delta_\nu \end{bmatrix} \\ &= \Phi(\mathbf{X})^\mathsf{T}\mathbf{e} - \mathbf{S}\Phi(\mathbf{X})^\mathsf{T} \begin{bmatrix} \omega \\ \nu \end{bmatrix} \end{aligned} \tag{18}$$

$$\begin{aligned} &\Phi(\mathbf{X})(\Phi(\mathbf{X})^\mathsf{T}\Lambda\Phi(\mathbf{X}) + \mathbf{S})\Phi(\mathbf{X})^\mathsf{T} \begin{bmatrix} \Delta_\omega \\ \Delta_\nu \end{bmatrix} \\ &= \Phi(\mathbf{X})\Phi(\mathbf{X})^\mathsf{T}\mathbf{e} - \Phi(\mathbf{X})\mathbf{S}\Phi(\mathbf{X})^\mathsf{T} \begin{bmatrix} \omega \\ \nu \end{bmatrix} \end{aligned} \tag{19}$$

$$(\mathbf{K}\Lambda\mathbf{K} + \mathbf{K}\mathbf{S}') \begin{bmatrix} \Delta_\omega \\ \Delta_\nu \end{bmatrix} = \mathbf{K}\mathbf{e} - \mathbf{K}\mathbf{S}' \begin{bmatrix} \omega \\ \nu \end{bmatrix} \tag{20}$$

$$(\Lambda\mathbf{K} + \mathbf{S}') \begin{bmatrix} \Delta_\omega \\ \Delta_\nu \end{bmatrix} = \mathbf{e} - \mathbf{S}' \begin{bmatrix} \omega \\ \nu \end{bmatrix} \tag{21}$$

Given the parameters, test instance $\mathbf{x}$ is classified by $f(\mathbf{x};\omega) = \text{argmax}_y p(y|\mathbf{x};\omega)$:

$$p\left(y=1|\mathbf{x};\omega\right) = \frac{1}{1+\exp\left(-\sum_{i=1}^m \omega_i k(\mathbf{x},\mathbf{x}_i)\right)}. \tag{22}$$

5.3. Solving the Optimization Problems

The optimization problems for the regular and kernel logistic regression classifiers are convex and therefore the Newton gradient descent algorithm is not only efficient, but will also find the global maximum with certainty. The question arises whether the same is true for the logistic regression with differing training and test distributions. Alas, this is not generally the case.

Theorem 2 *Optimization Problem 1 is not generally convex. It contains local* pockets of convexity*.*

It is generally a good choice to select the parameters of a regular, *iid* logistic regression classifier as starting point for the Newton gradient search. Thereby, the search converges to a local optimum that is at least as good as the *iid* classifier. Since *iid* logistic regression has a convex optimization criterion, this starting point is easily found.

Corollary 1 *The set of all vectors* $\mathbf{w}$ *that separate the training instances is a convex set. Within this set, Optimization Problem 1 is also convex. That is, if the global maximum of Optimization Problem 1 separates the training data, then it can be found from any starting point that also separates the training data.*

The proofs of Theorem 2 and Corollary 1 can be found in Appendix B.

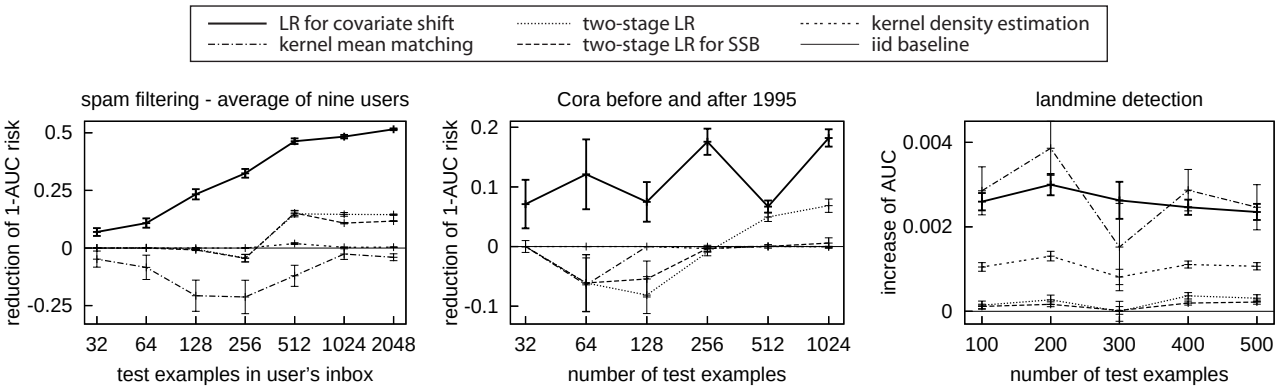

Figure 3. Average reduction of 1-AUC risk over nine users for spam filtering (left) and Cora *Machine Learning/Networking* classification before and after 1995 (center) and average increase of AUC for landmine detection over 812 pairs of mine fields (right) depending on the number of test examples.

6. Empirical Results

We study the benefit of logistic regression for covariate shift ("LR for covariate shift") and other reference methods on spam filtering, text classification and landmine detection problems. The first baseline is a classifier trained under *iid* assumption. 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 baselines differ in the first stage, the second stage is based on a logistic regression classifier with weighted examples in any case.

The first reference method is two-stage logistic regression ("two-stage LR"). The example weights are computed according to Equation 7, $p(\sigma = 1|\mathbf{x}, \mathbf{v})$ is estimated by training a logistic regression that discriminates training from test examples. The second method is the two-stage procedure for learning under sample selection bias (Zadrozny, 2004; Bickel & Scheffer, 2007) ("two-stage LR for SSB"). It estimates $p(s = 1|\mathbf{x}, \mathbf{v})$ like the previous method, the training examples are reweighted by $p(s = 1|\mathbf{x}, \mathbf{v})^{-1}$ and the weights are normalized. The third method is kernel mean matching (Huang et al., 2007). In the fourth 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). We tune the regularization parameters of the logistic regression methods, the B parameter of kernel mean matching, and the variance parameter of the RBF kernels on a separate tuning set.

For the spam filtering experiments we use the preprocessed data sets of Bickel and Scheffer (2007). There are nine different inboxes with test emails (5270-10964 emails, depending on inbox) and one set of training emails from different sources. We use a fixed set of 1000 emails as training data. We randomly select 32-2048 emails from one of the original inboxes. We repeat this process ten times. The performance measure is the rate by which the 1-AUC risk is reduced (Bickel & Scheffer, 2007) over the *iid* baseline; it is computed as $1 - \frac{1-AUC}{1-AUC_{iid}}$. We use linear kernels for all methods. Figure 3 (left) shows the result for various numbers of test examples. The results for a specific number of test examples are averaged over 10 random test samples and averaged over all nine inboxes. Averaged over all users and inbox sizes the absolute AUC of the *iid* classifier is 0.992. Error bars indicate standard errors of the 1-AUC risk.

The integrated logistic regression classifier for covariate shift outperforms all reference methods, the differences are highly significant. For 2048 examples the 1-AUC risk is even reduced by an average of 50%! For this problem, kernel mean matching fails to beat the baseline on average. We believe that kernel mean matching does not harmonize well with linear kernels.

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 1219 papers written before 1995 from both classes as training examples and 1820 papers written after 1995 as test examples. Title and abstract are transformed into term frequency vectors, the number of distinct words is 40,000. We again use linear kernels and average the results over 10 random test samples for different sizes of test sets. The resulting 1-AUC risk is shown in Figure 3 (center). The average absolute AUC of the *iid* classifier is 0.979. The logistic regression for covariate shift outperforms all other methods, for 1024 examples it reduces the 1-AUC risk by 18%.

In a third set of experiments we study landmine detection 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 pairs of mine fields, using one field as training, and the other as test data; results are increases over the *iid* baseline, averaged over all 29×28 combinations. We use RBF kernels for all methods. The results are displayed in Figure 3 (right). The average absolute AUC of the *iid* baseline is 0.64 with a standard deviation of 0.07. For this problem, logistic regression for covariate shift and kernel mean matching outperform all other methods on average. There is no significant difference between kernel mean matching and the integrated logistic regression classifier, but all other reference methods perform worse.

7. Conclusion

Equation 12 states the criterion to be maximized for learning a MAP classifier under covariate shift. Procedures that deviate from directly maximizing Equation 12, for instance by first only tuning parameters $\mathbf{v}$ of the example-specific weights and then learning the classifier $\mathbf{w}$ for fixed weights, only approximate the MAP classifier. Optimization Problem 1 is the special case of Equation 12 for the kernel logistic regression classifier. We derived a Newton gradient descent procedure. For spam and classification of scientific papers with linear kernels, the logistic regression classifier for covariate shift outperforms all references; for landmine detection with RBF kernels, both kernel mean matching and the logistic regression classifier perform well.

Acknowledgment

We gratefully acknowledge support from the German Science Foundation DFG and from STRATO AG. Thanks to Jiayuan Huang et al. who provided their implementation of the kernel mean matching algorithm.

References

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

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

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

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

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

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

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

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

Silverman, B. W. (1986). *Density estimation for statistics and data analysis.* Chapman & Hall, London.

Sugiyama, M., & Müller, K.-R. (2005). Model selection under covariate shift. *Proceedings of the International Conference on Artificial Neural Networks.*

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

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

A. Newton Gradient Descent

In this Appendix, we derive the Newton gradient descent method for the logistic regression classifier for differing training and test distributions.

Equations 13 and 14 detail the conditional distributions of the class and selector variables. We abbreviate

$$p_i = p\left(y_i = 1 | \mathbf{x}_i; \mathbf{w}\right); \quad q_i = p\left(\sigma_i = 1 | \mathbf{x}_i; \mathbf{v}\right)$$

and thus rephrase Optimization Problem 1 (Equation 12) with logistic conditionals, exponential prior, and binary labels $y_i \in \{0, 1\}$ as

$$\begin{aligned} & p(\mathbf{w}, \mathbf{v} | L, T) \\ \propto & \left(\prod_{i=1}^{m} \left(p_i^{y_i} (1-p_i)^{1-y_i}\right)^{\frac{1}{q_i}-1} \right)^{\frac{p(\sigma=1|\mathbf{v})}{p(\sigma=0|\mathbf{v})}} \\ & \left(\prod_{i=1}^{m+n} \left(q_i^{\sigma_i} (1-q_i)^{1-\sigma_i}\right) \right) \exp\left(\frac{-\mathbf{w}^\top \mathbf{w}}{2 s_{\mathbf{w}}^2}\right) \exp\left(\frac{-\mathbf{v}^\top \mathbf{v}}{2 s_{\mathbf{v}}^2}\right) \end{aligned} \tag{23}$$

The maximum likelihood estimate of $p(\sigma = 1|\mathbf{v})/p(\sigma = 0|\mathbf{v})$ is $\frac{m}{n}$; a symmetric beta prior with parameter α may be contained in $p(\mathbf{v})$ which leads to a MAP estimate of $\frac{m+\alpha}{n+\alpha}$. We omit α for notational brevity. The log-posterior is

$$\begin{aligned}&\log p(\mathbf{w}, \mathbf{v}|L, T)\\ &= \frac{m}{n}\sum_{i=1}^{m}\left(\frac{1}{q_i}-1\right)(y_i \log p_i + (1-y_i)\log(1-p_i)) \qquad (24)\\ &\quad + \sum_{i=1}^{m+n}(\sigma_i \log q_i + (1-\sigma_i)\log(1-q_i)) - \frac{\mathbf{w}^\mathsf{T}\mathbf{w}}{2s_\mathbf{w}^2} - \frac{\mathbf{v}^\mathsf{T}\mathbf{v}}{2s_\mathbf{v}^2} + C\end{aligned}$$

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

$$\begin{aligned}&\frac{\partial \log p(\mathbf{w}, \mathbf{v}|L, T)}{\partial w_j} \qquad (25)\\ &= \frac{m}{n}\sum_{i=1}^{m}\left(\frac{1}{q_i}-1\right)(y_i - p_i)\, x_{ij} - \frac{1}{s_\mathbf{w}^2} w_j \qquad (26)\\ &\frac{\partial \log p(\mathbf{w}, \mathbf{v}|L, T)}{\partial v_j}\\ &= \frac{m}{n}\sum_{i=1}^{m}\left(\frac{1}{q_i}-1\right)(-y_i \log p_i - (1-y_i)\log(1-p_i))x_{ij}\\ &\quad + \sum_{i=1}^{m+n}(\sigma_i - q_i)\, x_{ij} - \frac{1}{s_\mathbf{v}^2} v_j \qquad (27)\end{aligned}$$

The Hessian is the matrix of second derivatives.

$$\begin{aligned}&\frac{\partial^2 \log p(\mathbf{w}, \mathbf{v}|L, T)}{\partial w_j \partial w_k}\\ &= -\frac{m}{n}\sum_{i=1}^{m}\left(\frac{1}{q_i}-1\right) p_i(1-p_i)\, x_{ij}x_{ik} - \frac{1}{s_w^2}\delta_{jk} \qquad (28)\\ &\frac{\partial^2 \log p(\mathbf{w}, \mathbf{v}|L, T)}{\partial v_j \partial w_k} \qquad (29)\\ &= -\frac{m}{n}\sum_{i=1}^{m}\left(\frac{1}{q_i}-1\right)(y_i - p_i)x_{ij}x_{ik} \qquad (30)\\ &\frac{\partial^2 \log p(\mathbf{w}, \mathbf{v}|L, T)}{\partial v_j \partial v_k}\\ &= -\frac{m}{n}\sum_{i=1}^{m}\left(\frac{1}{q_i}-1\right)(-y_i \log p_i - (1-y_i)\log(1-p_i))x_{ij}x_{ik}\\ &\quad - \sum_{i=1}^{m+n} q_i(1-q_i)\, x_{ij}x_{ik} - \frac{1}{s_\mathbf{v}^2}\delta_{jk} \qquad (31)\end{aligned}$$

We define the following abbreviations.

$$\begin{aligned}a_i &= \frac{m}{n}\left(\frac{1}{q_i}-1\right)(y_i - p_i) \qquad (32)\\ b_i &= \frac{m}{n}\left(\frac{1}{q_i}-1\right)(-y_i \log p_i - (1-y_i)\log(1-p_i)) \qquad (33)\\ c_i &= \frac{m}{n}\left(\frac{1}{q_i}-1\right) p_i(1-p_i) \qquad (34)\end{aligned}$$

We can rewrite gradient and Hessian.

$$\begin{aligned}\frac{\partial \log L(\mathbf{v}, \mathbf{w})}{\partial \mathbf{w}} &= \mathbf{L}\mathbf{a} - s_\mathbf{w}^{-2}\mathbf{w} \qquad (35)\\ \frac{\partial \log L(\mathbf{v}, \mathbf{w})}{\partial \mathbf{v}} &= \mathbf{L}\operatorname*{diag}_{i=1..m}(b_i + 1 - q_i)\mathbf{1} +\\ &\quad \mathbf{T}\operatorname*{diag}_{i=1..n}(-q_{m+i})\mathbf{1} - s_\mathbf{v}^{-2}\mathbf{v} \qquad (36)\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 \log L(\mathbf{v}, \mathbf{w})}{\partial \mathbf{w}\partial \mathbf{w}} &= -\mathbf{L}diag(\mathbf{c})\mathbf{L}^\mathsf{T} - s_\mathbf{w}^{-2}\mathbf{I} \qquad (37)\\ \frac{\partial^2 \log L(\mathbf{v}, \mathbf{w})}{\partial \mathbf{w}\partial \mathbf{v}} &= -\mathbf{L}diag(\mathbf{a})\mathbf{L}^\mathsf{T} \qquad (38)\\ \frac{\partial^2 \log L(\mathbf{v}, \mathbf{w})}{\partial \mathbf{v}\partial \mathbf{v}} &= -\mathbf{L}\operatorname*{diag}_{i=1..m}(b_i + q_i(1-q_i))\,\mathbf{L}^\mathsf{T} \qquad (39)\\ &\quad -\mathbf{T}\operatorname*{diag}_{i=1..n}(q_{m+i}(1-q_{m+i}))\,\mathbf{T}^\mathsf{T} - s_\mathbf{v}^{-2}\mathbf{I}\end{aligned}$$

Now we define $\mathbf{S}_{i,i} = s_\mathbf{w}^{-2}$ for $i = 1..\dim(\mathbf{L})$ and $\mathbf{S}_{\dim(\mathbf{L})+j,\dim(\mathbf{L})+j} = s_\mathbf{v}^{-2}$ for $j = 1..\dim(\mathbf{T})$ and set

$$\mathbf{X} = \begin{bmatrix}\mathbf{L}^\mathsf{T} & \mathbf{0}\\ \mathbf{0} & \mathbf{L}^\mathsf{T}\\ \mathbf{0} & \mathbf{T}^\mathsf{T}\end{bmatrix},\quad \mathbf{e} = \begin{bmatrix}\mathbf{a}\\ \operatorname*{diag}_{i=1..m}(b_i + 1 - q_i)\,\mathbf{1}\\ \operatorname*{diag}_{i=1..n}(-q_{m+i})\,\mathbf{1}\end{bmatrix},$$

$$\mathbf{\Lambda} = \begin{bmatrix}\operatorname{diag}(\mathbf{c}) & \operatorname{diag}(\mathbf{a}) & \mathbf{0}\\ \operatorname{diag}(\mathbf{a}) & \operatorname*{diag}_{i=1..m}(b_i + q_i(1-q_i)) & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \operatorname*{diag}_{i=1..n}(q_{m+i}(1-q_{m+i}))\end{bmatrix}.$$

The Newton update rule for $\mathbf{w}$ and $\mathbf{v}$ can now be expressed as

$$\begin{bmatrix}\mathbf{w}'\\ \mathbf{v}'\end{bmatrix} = \begin{bmatrix}\mathbf{w}\\ \mathbf{v}\end{bmatrix} + \left(\mathbf{X}^\mathsf{T}\mathbf{\Lambda}\mathbf{X} + \mathbf{S}\right)^{-1}\left(\mathbf{X}^\mathsf{T}\mathbf{e} - \mathbf{S}\begin{bmatrix}\mathbf{w}\\ \mathbf{v}\end{bmatrix}\right).$$

For the kernelized update rule $\mathbf{S}$ is replaced by $\mathbf{S}'$ such that $\Phi(\mathbf{X})\mathbf{S}\Phi(\mathbf{X})^\mathsf{T} = \Phi(\mathbf{X})\Phi(\mathbf{X})^\mathsf{T}\mathbf{S}'$, i.e., $\mathbf{S}'_{i,i} = s_\mathbf{w}^{-2}$ for $i = 1..m$ and $\mathbf{S}'_{m+j,m+j} = s_\mathbf{v}^{-2}$ for $j = 1..m+n$, and $\Phi(\mathbf{X})$ is defined by

$$\Phi(\mathbf{X}) = \begin{bmatrix}\Phi(\mathbf{L}^\mathsf{T}) & \mathbf{0}\\ \mathbf{0} & \Phi(\mathbf{L}^\mathsf{T})\\ \mathbf{0} & \Phi(\mathbf{T}^\mathsf{T})\end{bmatrix}.$$

B. Proof of Theorem 2 and Corollary 1

The optimization criterion is convex if the (negative) Hessian is positive semi-definite, which means Λ is positive semi-definite. This, in turn, is the case if and only if the Cholesky decomposition of Λ exists; *i.e.*, if there is a $\mathbf{G}$ such that $\mathbf{G}^\mathsf{T}\mathbf{G} = \Lambda$. According to the Cholesky-Crout algorithm the diagonal elements of the second block of $\mathbf{G}$ are $\sqrt{b_i + q_i(1-q_i) - \frac{a_i^2}{c_i}}$. Hence, the Cholesky decomposition exists if and only if for all training instances $\mathbf{x}_i$: $b_i + q_i(1-q_i) - \frac{a_i^2}{c_i} \geq 0$. Resolving a_i, b_i, and c_i, this is equivalent to

$$\begin{aligned}&\forall i, y_i = 0 : \tfrac{n}{m}q_i^2 \geq \tfrac{p_i}{1-p_i} - \ln\tfrac{1}{1-p_i}\\ &\forall i, y_i = 1 : \tfrac{n}{m}q_i^2 \geq \tfrac{1-p_i}{p_i} - \ln\tfrac{1}{p_i}\end{aligned} \quad . \qquad (40)$$

Equation 40 characterizes the local pockets of convexity of the optimization criterion. For all linear classifiers, the version space is well known to be a convex set; the corner points are those separators $\mathbf{w}$ that touch one of the training examples. Within this set, $\mathbf{w}$ can be chosen with a sufficiently large norm to satisfy Equation 40. Hence, Optimization Problem 1 is convex within this set which proves Corollary 1.

Equation 40 also indicates that the larger the number of test examples n, the more likely the condition holds and the global optimum can be found by the algorithm.