Nearest Neighbor Density Ratio Estimation for Large-Scale Applications in Astronomy

Jan Kremer
Department of Computer Science
University of Copenhagen
2200 Copenhagen, Denmark
jan.kremer@di.ku.dk

Fabian Gieseke
Institute for Computing and Information Sciences
Radboud University Nijmegen
6525 EC Nijmegen, Netherlands
f.gieseke@cs.ru.nl

Kim Steenstrup Pedersen
Department of Computer Science
University of Copenhagen
2200 Copenhagen, Denmark
kimstp@di.ku.dk

Christian Igel
Department of Computer Science
University of Copenhagen
2200 Copenhagen, Denmark
igel@di.ku.dk

Abstract

In astronomical applications of machine learning, the distribution of objects used for building a model is often different from the distribution of the objects the model is later applied to. This is known as sample selection bias, which is a major challenge for statistical inference as one can no longer assume that the labeled training data are representative. To address this issue, one can re-weight the labeled training patterns to match the distribution of unlabeled data that are available already in the training phase. There are many examples in practice where this strategy yielded good results, but estimating the weights reliably from a finite sample is challenging. We consider an efficient nearest neighbor density ratio estimator that can exploit large samples to increase the accuracy of the weight estimates. To solve the problem of choosing the right neighborhood size, we propose to use cross-validation on a model selection criterion that is unbiased under covariate shift. The resulting algorithm is our method of choice for density ratio estimation when the feature space dimensionality is small and sample sizes are large. The approach is simple and, because of the model selection, robust. We empirically find that it is on a par with established kernel-based methods on relatively small regression benchmark datasets. However, when applied to large-scale photometric redshift estimation, our approach outperforms the state-of-the-art.

1 Introduction

In many machine learning applications labeled (training) and unlabeled (test) data do not follow the same distribution. One reason can be that the labeled patterns have not been sampled randomly. In astronomy such a sample selection bias arises because objects that are expected to show more interesting properties are preferred when it comes to costly high-quality spectroscopic follow-up observations; other objects whose scientific value may not be that obvious (e.g., seemingly star-like objects) may be overlooked [1]. One way to address this bias is to weight the labeled training sample according to the ratio between the two probability distributions [2]. As this true ratio is usually not available, one has to estimate it from a finite sample. The crucial point is to control the variance of the estimator. Empirically, it seems promising to reduce the variance of the estimator by accepting a slightly higher bias [3]. This gives rise to ratio estimators that, in practice, perform better than the naïve approach of estimating the two densities separately.

In this work, we improve a simple nearest neighbor density ratio estimator [4] by combining it with a principled way of performing model selection [5]. The approach compares well to established kernel-based estimators on a variety of standard, small-sized regression datasets. Furthermore, by selecting proper hyperparameters and by taking huge amounts of patterns into account, we experimentally show that the estimator yields better results compared to the state-of-the-art on a large-scale astronomical dataset.

Let each data point be represented by a feature vector $\mathbf{x}$ from a domain $\mathcal{X}$ with a corresponding label $\mathbf{y}$ from a domain $\mathcal{Y}$. We consider scenarios in which the learner has access to some labeled (source) data S sampled from $p_s(\mathbf{x}, \mathbf{y})$ and a large sample of unlabeled (target) data T sampled from $p_t(\mathbf{x}, \mathbf{y})$. While $p_s(\mathbf{x}, \mathbf{y})$ and $p_t(\mathbf{x}, \mathbf{y})$ may not coincide, we assume that $p_s(\mathbf{y}|\mathbf{x}) = p_t(\mathbf{y}|\mathbf{x})$ for all $\mathbf{x}$ and that the support of p_t is a subset of the support of p_s. This is usually referred to as *covariate shift*, a particular type of *sample selection bias*. In this case the probability density ratio between target and source distribution at a given point reduces to $\beta(\mathbf{x}) = \frac{p_t(\mathbf{x})}{p_s(\mathbf{x})}$.

Different strategies have been proposed to address covariate shift, such as finding a common feature space or re-weighting the source patterns. The latter is conceptually simple, and there are several approaches to estimate appropriate weights via density ratio estimation [2, 4, 5, 6, 7, 8, 9, 10, 11]. These methods are, for example, based on reducing the problem to probabilistic classification between the target and source dataset [6], on using kernel-based methods to match means in an induced Hilbert space [2], or on using nearest neighbor queries to estimate the mismatch between the densities by counting patterns in local regions [4, 8]. It is crucial to control the variance of such an estimator via regularization. Depending on the algorithm at hand, the regularization can take the form of, for example, a kernel bandwidth [2], the rank of a low-rank kernel matrix approximation [10], or a weight norm [11]. The involved parameters are often set by heuristics such as the median of pairwise distances for the kernel bandwidth [12]. As an alternative, Sugiyama et al. [5] suggest a model selection criterion that is unbiased under covariate shift. In the following, we employ this criterion for selecting the neighborhood size of the nearest neighbor estimator via cross-validation. Then, we empirically show that the resulting algorithm can outperform the computationally more expensive state-of-the-art kernel-based estimator due to its ability to consider larger samples in less time.

This article is structured as follows: in Section 2 we briefly discuss two state-of-the-art kernel-based estimators that serve as a baseline in our experimental evaluation. In Section 3 we present a nearest neighbor-based density ratio estimator and show how it can be extended to perform automatic model selection. In Section 4 we evaluate the proposed nearest neighbor density ratio estimator with integrated model selection in comparison to other methods on a medium-sized regression benchmark and on a large-scale astronomical dataset for photometric redshift estimation. In Section 5 we conclude and give possible directions for future work.

2 Kernel-based Density Ratio Estimation

In density ratio estimation, kernel-based estimators are considered the state-of-the-art [13]. Among these, kernel mean matching (KMM) [2] and the spectral series estimator [10] have shown to perform particularly well.

Given some input space $\mathcal{X}$, a kernel is a positive semi-definite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for which $\forall x, z \in \mathcal{X} : k(x, z) = \langle \Phi(x), \Phi(z) \rangle_{\mathcal{H}}$, where $\Phi : \mathcal{X} \to \mathcal{H}$ maps elements of the input space to a kernel-induced Hilbert space $\mathcal{H}$ [14]. Kernel mean matching aims at matching the means of two distributions in $\mathcal{H}$ by solving the problem

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \ \left\| \frac{1}{N_s} \sum_{i=1}^{N_s} \beta_i \Phi(\mathbf{x}_i^{(s)}) - \frac{1}{N_t} \sum_{i=1}^{N_t} \Phi(\mathbf{x}_i^{(t)}) \right\|_{\mathcal{H}}^2 \tag{1}$$

$$\text{subject to} \ \ \beta_i \in [0, B] \ \text{ and } \ \left| \sum_{i=1}^{N_s} \beta_i - N_s \right| \leq N_s \epsilon \ , \tag{2}$$

where N_s is the number of source domain patterns and N_t is the number of target domain patterns. The parameter B restricts the maximum possible weight and ϵ bounds the deviation of the mean weight from 1. Cortes et al. [7] show that the solution to Eq. (1) converges with high probability

to the true density ratio if the kernel induced by $\Phi(\mathbf{x})$ is universal [15]. The kernel function, which implicitly defines Φ and $\mathcal{H}$, is typically chosen from a parameterized family of functions, and the kernel parameters are parameters of KMM-based approaches.

The spectral series estimator [10], although motivated differently, minimizes an unconstrained version of Eq. (1) for computing training weights. Instead of bounding the weights via B and their mean via ϵ, the solution is regularized by the rank J of a low-rank approximation of the kernel Gram matrix between training points – which results when expanding Eq. (1). Unlike KMM, the spectral series estimator can compute weights not only for the source sample, but also for arbitrary patterns. This allows for selecting the kernel parameters and J via cross-validation, as we shall see later.

Negative theoretical results in the analysis of weighting methods [16, 17] suggest that sample sizes have to be prohibitively large to guarantee reliable weights. However, empirically it has been found that re-weighting often does improve results. Our method is motivated by typical tasks in astronomy, where we deal with large labeled samples and huge unlabeled samples in feature spaces of relatively low dimensionality (e.g., up to $\mathbb{R}^{10}$). For such rather benign scenarios, we aim at estimating weights with high accuracy by taking into account hundreds of thousands of labeled and unlabeled patterns. However, both KMM as well as the spectral series estimator involve $|S| \times |T|$ kernel matrices in their general form. Thus, they are not directly applicable to scenarios with hundreds of thousands of patterns. Special cases might be addressed in a more efficient way. Still, the general cases with non-linear kernel functions involve the computation of such kernel matrices and, depending on the method, quadratic programming, matrix inversion, or eigenvalue decomposition, which exhibit at least a quadratic running time [18, 19, 20]. Therefore, we are considering nearest neighbor-based density ratio estimation, which can be implemented more efficiently.

For the matrix decompositions in the spectral series estimator we used an efficient $\mathcal{O}(n^2)$-algorithm [21]. Both, decomposition as well as the nearest neighbor search, could be sped up by using approximation schemes (e.g., see [22, 23]), but we decided not to introduce such approximations with corresponding hyperparameters in our study.

3 Nearest Neighbor Density Ratio Estimation Revisited

We consider the algorithm proposed by Lima et al. [4] to estimate appropriate ratios via nearest neighbor queries, see Algorithm 1. The efficiency of the approach is ensured via the use of k-d trees. For the sake of completeness, we briefly sketch how these spatial data structures can be used to speed up nearest neighbor search before outlining the details of the density ratio estimator.

3.1 Nearest Neighbor Search in Low Dimensions

A classical *k-d tree* [24] is a binary tree constructed from a d-dimensional point set $S \subset \mathbb{R}^d$. The inner nodes correspond to hyperplanes splitting the data in $\mathbb{R}^d$ and the leaf nodes define a partitioning of S. The tree can be built recursively in $\mathcal{O}(|S| \log |S|)$ time. Starting from the root node numbered by 0 and $S_0 = S$, each inner node v with children u and w partitions the data S_v into two almost equal-sized subsets S_u and S_w. If S_v contains only a single point (or a predefined number of points), v becomes a leaf node. At tree level j, the datasets are split according to the median in dimension $j \bmod d + 1$.

To efficiently search for the nearest neighbor of a given query point $\mathbf{q} \in \mathbb{R}^d$, one can make use of the hierarchical subdivision induced by a k-d tree: The tree is traversed in two phases. During the first phase, the tree is processed from top to bottom to find the d-dimensional leaf (box) that contains the query point (the search is guided by the median values). The query point is then compared with all points that are stored in the corresponding leaf, which yields the first nearest neighbor candidate. Afterwards, in the second phase, the tree is processed from bottom to top and on the way back to the root, neighboring boxes are checked for points that are potentially closer to $\mathbf{q}$ than the current candidate. In case the distance of $\mathbf{q}$ to the splitting hyperplane is larger than the distance between $\mathbf{q}$ and its current nearest neighbor candidate, one can safely ignore the whole subtree that has not yet been visited. These distance checks can be performed efficiently by resorting to the associated median values. The generalization to $k > 1$ neighbors is straightforward (see, e.g., [24], or [25], for details).

Algorithm 1 NEAREASTNEIGHBORRATIOESTIMATOR

Require: A set $S = \{\mathbf{x}_1^{(s)}, \ldots, \mathbf{x}_{N_s}^{(s)}\} \subset \mathbb{R}^d$ and a set $T = \{\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_{N_t}^{(t)}\} \subset \mathbb{R}^d$ of patterns from the source and target domain, respectively, a query set $Q = \{\mathbf{x}_1^{(q)}, \ldots, \mathbf{x}_{N_q}^{(q)}\} \subset \mathbb{R}^d$, and a number $K > 1$.

Ensure: Weights $\widehat{\beta}(\mathbf{x}_1^{(q)}), \ldots, \widehat{\beta}(\mathbf{x}_{N_q}^{(q)}) \in \mathbb{R}$ for the patterns in Q.

1: Construct k-d trees $\mathcal{T}_s$ and $\mathcal{T}_t$ for S and T, respectively.
2: $W = \{\}$
3: **for** $j = 1, \ldots, N_q$ **do**
4: Compute the K nearest neighbors for $\mathbf{x}_j^{(q)}$ in S (via k-d tree $\mathcal{T}_s$).
5: Let r_j be the Euclidean distance between $\mathbf{x}_j^{(q)}$ and its K-th nearest neighbor.
6: Compute number l_j of nearest neighbors in T with distance less than r_j to $\mathbf{x}_j^{(q)}$ (via k-d tree $\mathcal{T}_t$).
7: $W = W \cup \left\{ l_j \cdot \frac{N_s}{K \cdot N_t} \right\}$
8: **end for**
9: **return** W

In the best case, all nearest neighbors are contained in the leaf that stems from the first phase and no further subtrees need to be processed on the way back to the root. For such queries, the runtime is logarithmic in the number $|S|$ of points. This also holds for the expected case as shown by [26] (given constant d). In the worst case, however, the complete k-d tree needs to be processed, which leads to a linear instead of a logarithmic runtime per query. From a practical perspective, the running time depends on the dimensionality of the feature space: for moderate dimensions (e.g., up to $d = 30$), a logarithmic running time behavior can be expected, while for larger d the performance often decreases significantly due to the *curse of dimensionality* [27].

3.2 Nearest Neighbor Density Ratio Estimator

We are now ready to outline the details of Algorithm 1: In Step 1, k-d trees for the source and target patterns are built. In Steps 2 to 8, all query patterns $\mathbf{x}_j^{(q)}$ are processed. For each query pattern, the K nearest neighbors w.r.t. the source patterns in S are computed. This is followed by the computation of the number l_j of nearest neighbors in T whose distance to $\mathbf{x}_j^{(q)}$ is less than or equal to the previously computed K-th nearest neighbor. Finally, this result is re-weighted according to the number N_s of source patterns, the number N_t of target patterns and the number K of nearest neighbors. Hence, the true density ratio $\beta(\mathbf{x}_j^{(q)})$ of target and source distribution at a point $\mathbf{x}_j^{(q)}$ in the feature space $\mathbb{R}^d$ is approximated via

$$\widehat{\beta}(\mathbf{x}_j^{(q)}) = l_j \cdot \frac{N_s}{K \cdot N_t} \ . \tag{3}$$

As k-d trees speed up nearest neighbor computation for low-dimensional feature spaces, we get good running time results in this scenario: the construction of the trees for the source and target patterns takes $\mathcal{O}(N_s \log N_s)$ and $\mathcal{O}(N_t \log N_t)$ time, respectively. For each query pattern, nearest neighbors are computed via these trees. The number K of neighbors is crucial for the accuracy of the algorithm, and the question of how to choose it is not discussed in [4]. We propose to select K via cross-validation by minimizing the model selection criterion proposed in [5]. It seeks to minimize the least-squares error between true and estimated density ratio, as in regression. However, we usually do not have access to the true density ratio. Therefore, we use a substitution to estimate the minimizer of the least-squares error up to a constant. The expected least-squares loss between true and estimated density ratio over the source probability density $p_s(\mathbf{x})$ is given by

$$\begin{aligned} L(\beta, \widehat{\beta}) &= \int (\beta(\mathbf{x}) - \widehat{\beta}(\mathbf{x}))^2 \, p_s(\mathbf{x}) \mathrm{d}\mathbf{x} \\ &= \int \widehat{\beta}(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} - 2 \int \widehat{\beta}(\mathbf{x}) \beta(\mathbf{x}) p_s(\mathbf{x}) \mathrm{d}\mathbf{x} + \int \beta(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} \\ &= \int \widehat{\beta}(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} - 2 \int \widehat{\beta}(\mathbf{x}) p_t(\mathbf{x}) \mathrm{d}\mathbf{x} + \int \beta(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} \ , \end{aligned} \tag{4}$$

where we substituted the true density ratio $\beta(\mathbf{x}) = \frac{p_t(\mathbf{x})}{p_s(\mathbf{x})}$. As the third term does not depend on the estimated ratio $\widehat{\beta}(\mathbf{x})$, we can estimate $L(\beta, \widehat{\beta})$ up to a constant by

$$\widehat{L}(\beta, \widehat{\beta}) = \frac{1}{|S|} \sum_{\mathbf{x} \in S} \widehat{\beta}(\mathbf{x})^2 - \frac{2}{|T|} \sum_{\mathbf{x} \in T} \widehat{\beta}(\mathbf{x}) \ . \tag{5}$$

Here, we have replaced the expectations in the first two terms by their empirical estimates. As long as $p_s(\mathbf{x})$ and $p_t(\mathbf{x})$ do not change, the constant term in Eq. (5) will not change and thus, we can safely ignore it when comparing different weight estimates $\widehat{\beta}$.

It is important to note that we evaluate Eq. (5) *post hoc* on trained density ratio estimators. Direct unconstrained minimization of Eq. (5) with respect to $\widehat{\beta}(\mathbf{x})$ for $\mathbf{x} \in S$ (i.e., the values needed for re-weighted training) would lead to the trivial solution $\widehat{\beta}(\mathbf{x}) = 0$ for $\mathbf{x} \in S$. An open-source Python package of the nearest neighbor estimator with integrated model selection is available on Github.[1]

4 Experiments

We consider two experiments: re-weighted regression on standard domain adaptation benchmarks and weight computation for photometric redshift estimation.

4.1 Regression Benchmarks

We compared our approach to kernel mean matching (KMM) [2] and the spectral series estimator [10] following the protocol of the experiments in [7]. For each of the eight regression datasets, which are rather small (the largest having 16 512 labeled and 9511 unlabeled patterns), we created a biased subset S of the original dataset T. As defined in [7], each point is moved from T to S with probability

$$p(s = 1|\mathbf{x}) = \frac{e^v}{1 + e^v} \ , \tag{6}$$

where v is defined as

$$v = \frac{4\mathbf{w} \cdot (\mathbf{x} - \overline{\mathbf{x}})}{\sqrt{\mathrm{Var}(\mathbf{w} \cdot (\mathbf{x} - \overline{\mathbf{x}}))}} \ , \tag{7}$$

for a pattern $\mathbf{x} \in \mathbb{R}^d$, and $\mathbf{w} \in \mathbb{R}^d$ chosen uniformly at random from $[-1, 1]^d$. Thus, the bias is only determined by the covariate $\mathbf{x}$. The ideal method, which we consider as a baseline, weights the points in S with $\frac{1}{p(s=1|\mathbf{x})}$. For each dataset, we selected the $\mathbf{w}$ that maximized the difference in regression loss between ideal and unweighted method among 10 trials.

After having estimated the weights, we use them to re-weight the loss function of a linear regularized least-squares estimator. Here, we select the regularization parameter $\lambda \in \{2^n : n \in \{-3, \ldots, 4\}\}$ via leave-one-out cross-validation. Since KMM has no mechanism for automatically choosing its hyperparameter, we chose the bandwidth $\sigma = \sqrt{d/2}$ for $\mathbf{x} \in \mathbb{R}^d$ [7]. For the spectral series estimator and the nearest neighbor method, we chose their parameters by performing 5-fold cross-validation using Eq. (5). We selected the bandwidth parameter ϵ of the spectral series estimator from $\{\epsilon_0^{-1}, \epsilon_0^0, \epsilon_0^1, \epsilon_0^2\}$, with $\epsilon_0 = \mathrm{median}(\{\|\mathbf{x} - \mathbf{y}\|_2^2 : \mathbf{x}, \mathbf{y} \in S\})/8$ [12], and the rank J from $\{1, \ldots, \lfloor N_s \times 4/5 \rfloor\}$. For the nearest neighbor estimator we selected a $K \in \{2, 3, 4, 5, 8, 16, 32\}$ and also considered a variant with fixed $K = 2$ to demonstrate the influence of model selection (the value $K = 2$ corresponds to the most frequent choice for K in the model selection algorithm).

Figure 1 shows the relative normalized mean squared error (NMSE) and the standard deviation over 10 different trials for each method on a test set not used for training or estimation of weights. For each dataset we scaled the results linearly so that the unweighted NMSE yielded 1.0. The weighted methods almost always perform better than unweighted regression. Although the sample sizes are small, the nearest neighbor estimator performs on a par with the kernel methods among which the spectral series estimator performs best. Because of the sample sizes, our method cannot tap its full potential and using a fixed $K = 2$ seems to be a viable approach. However, the picture changes when moving to our real-world large-scale application.

[1] `https://github.com/kremerj/nnratio`.

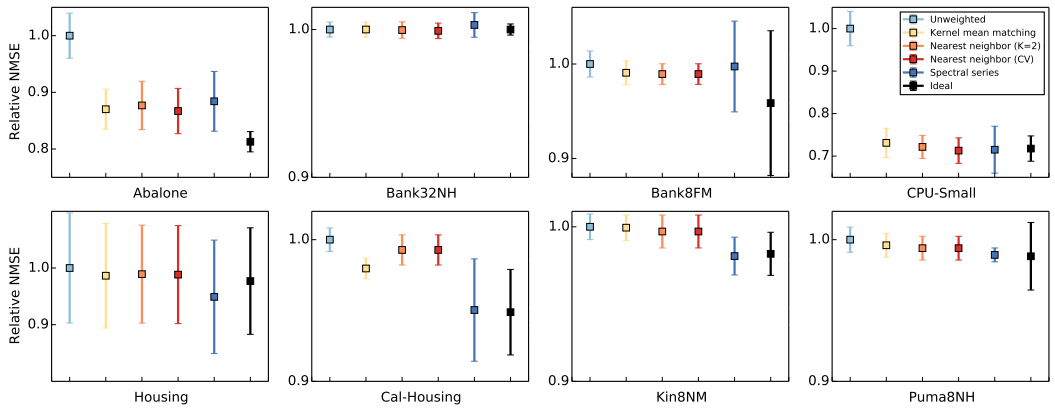
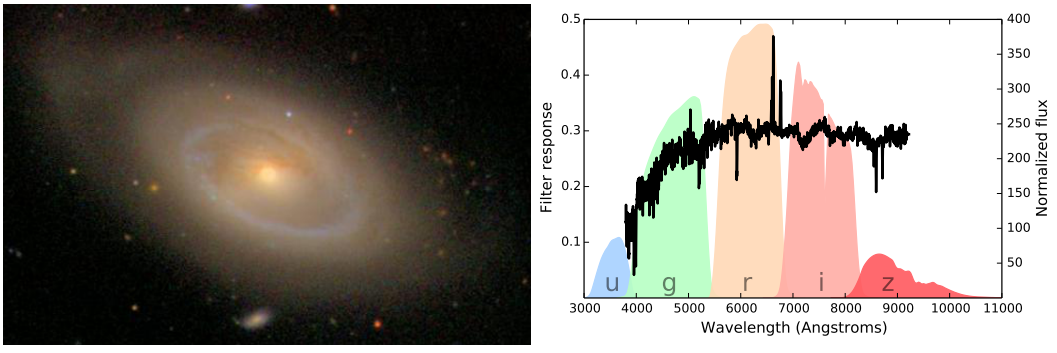


Figure 1: The relative normalized mean squared errors (NMSE) for kernel- and nearest neighbor-based ratio estimators on different regression datasets. The error bars indicate the standard deviations over 10 different samplings using the selection probability $p(s = 1|\mathbf{x})$ for a fixed $\mathbf{w}$.

4.2 Redshift Estimation

We evaluated our method on a large-scale astronomical dataset [10]. The problem we consider is photometric redshift estimation of galaxies. The redshift phenomenon is caused by the Doppler effect which shifts the spectrum of an object towards longer wavelengths if it is moving away from the observer. Because the universe is expanding uniformly, we can infer a galaxy's velocity by its redshift and, thus, its distance to Earth. Hence, redshift estimation is a useful tool for determining the geometry of the universe. A photometric observation contains the intensities of an object (in our case, galaxies) in 5 different bands (*u,g,r,i,z*), ranging from ultraviolet to infrared. Spectroscopy, in contrast, measures the photon count at certain wavelengths. The resulting spectrum allows for identifying the chemical components of the observed object and thus, enables determining many interesting properties, including the redshift. Spectroscopy, however, is much more time-consuming than photometric observation and therefore, costs could be greatly reduced if we could predict suitable candidates for follow-up spectroscopy from low-quality low-cost photometry. Figure 2 shows examples of corresponding photometric and spectroscopic observations.

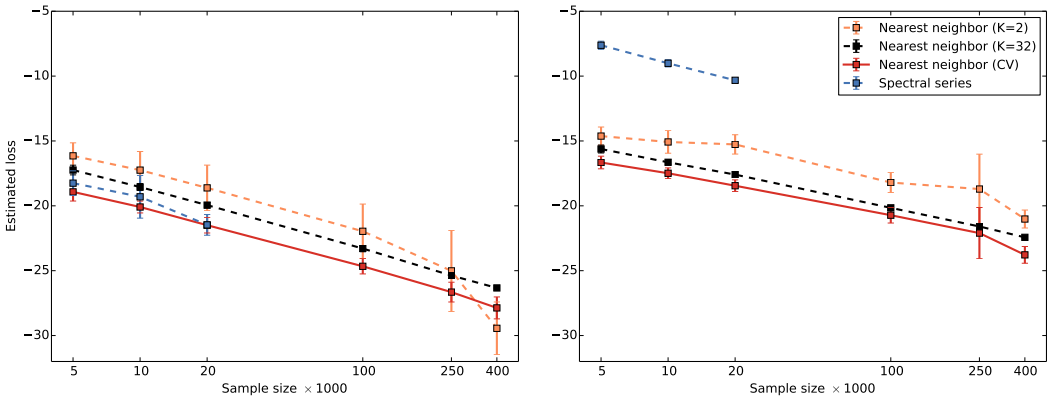
For each of the 5 bands a point spread function (*model*) and a composite model (*cmodel*) are fit to the photometric observation. We take the 4 magnitude differences between adjacent bands and the magnitude in the red band for *model* and *cmodel*. Thus, we arrive at $2 \times (4 + 1) = 10$ covariates for each galaxy.



(a) (b)

Figure 2: An example from the *Sloan Digital Sky Survey* (SDSS) [28]. (a) An image of the spiral galaxy NGC 5750. (b) Its associated spectrum overlapping the five photometric intensity band filters *u,g,r,i,z*.

The dataset contains a sample of 467 710 galaxies whose redshift has been confirmed by spectroscopy and an unconfirmed sample of 540 237 galaxies. The task is to estimate the redshift of the unconfirmed (target) sample by training on the spectroscopically confirmed (source) sample. As we do not have ground-truth labels for the target sample, we simply recorded the estimated loss given by Eq. (5) as in [10], see Figure 3. Interestingly, the absolute estimates are more accurate when we consider the dataset as-is. In Figure 3(b) we consider a preprocessed dataset where we standardized the covariates to have zero mean and unit variance, as is common for methods that rely on pattern distances. Here, we see that the nearest neighbor estimator with model selection outperforms the other methods even clearer, although the absolute estimated loss is higher than the one for the original data, see Figure 3(a). If the task can benefit from re-weighting, then the performance is likely to improve with more accurate weights.



(a) (b)

Figure 3: The estimated loss for the nearest neighbor and spectral series estimator on an astronomical dataset typically used in the context of photometric redshift estimation. (a) The original dataset. (b) Results with covariates transformed to have zero mean and unit variance.

In our experiment, we trained the ratio estimators with increasing sample sizes from 5000 to 400 000 patterns (each from source and target sample) and estimated the weights on hold-out test samples of size 50 000 (source and target) not used for training. As KMM cannot produce out-of-sample weights, we only compared the nearest neighbor estimator (with K either being fixed or chosen by model selection) and the spectral series estimator using the same parameters as in the first experiment. As running times become prohibitively large for the spectral series estimator, we only recorded it up to sample sizes of 20 000 patterns. Figure 4 shows the running time per sample size on an AMD Opteron 6380. The time measured includes the time used for cross-validation on a single-core machine. It should be noted that the cross-validation procedure is parallelizable to the point that its additional costs for the gained accuracy are minimal. For comparable running times the nearest neighbor estimator is able to use more samples than the spectral series estimator and thus, estimate weights more accurately. Furthermore, selecting the parameter K via cross-validation performs better than our default choice $K = 2$ (which was the most frequently selected value in the model selection experiments on the benchmark datasets).

5 Conclusion

Sample selection bias is a common problem in astronomy [29], where datasets are typically large and the feature space dimensionality is often low. For this scenario, we suggest to use a nearest neighbor density ratio estimator combined with a model selection criterion, which is unbiased under covariate shift, for choosing the neighborhood size. The resulting algorithm is simple, robust due to the systematic hyperparameter choice, and—as we experimentally demonstrate—highly efficient and accurate. Future work will consider the theoretical properties of the estimator and an implementation on GPUs [25] for handling datasets with billions of patterns efficiently and at low cost.

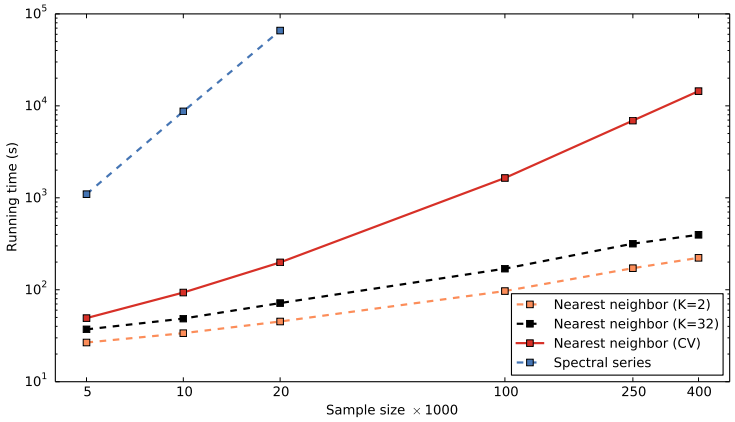


Figure 4: The running times per sample size for the different estimators, including the time used for cross-validation. The nearest neighbor estimators can utilize considerably larger samples than the spectral series estimator given the same time constraints.

Acknowledgments

The authors gratefully acknowledge support from The Danish Council for Independent Research through the project *SkyML* (FNU 12-125149). The authors also would like to thank R. Izbicki for providing the astronomical dataset. Funding for SDSS-III has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, and the U.S. Department of Energy Office of Science. The SDSS-III web site is http://www.sdss3.org/.

SDSS-III is managed by the Astrophysical Research Consortium for the Participating Institutions of the SDSS-III Collaboration including the University of Arizona, the Brazilian Participation Group, Brookhaven National Laboratory, Carnegie Mellon University, University of Florida, the French Participation Group, the German Participation Group, Harvard University, the Instituto de Astrofisica de Canarias, the Michigan State/Notre Dame/JINA Participation Group, Johns Hopkins University, Lawrence Berkeley National Laboratory, Max Planck Institute for Astrophysics, Max Planck Institute for Extraterrestrial Physics, New Mexico State University, New York University, Ohio State University, Pennsylvania State University, University of Portsmouth, Princeton University, the Spanish Participation Group, University of Tokyo, University of Utah, Vanderbilt University, University of Virginia, University of Washington, and Yale University.

References

[1] D. J. Mortlock, S. J. Warren, B. P. Venemans, M. Patel, P. C. Hewett, R. G. McMahon, C. Simpson, T. Theuns, E. A. Gonzales-Solares, A. Adamson, S. Dye, N. C. Hambly, P. Hirst, M. J. Irwin, E. Kuiper, A. Lawrence, and H. J. A. Rottgering. A luminous quasar at a redshift of z = 7.085. *Nature*, 474(7353):616–619, 2011.

[2] J. Huang, A. J. Smola, A. Gretton, K. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems (NIPS)*, volume 19, pages 601–608. MIT Press, 2007.

[3] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Bünau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, pages 1433–1440. MIT Press, 2008.

[4] M. Lima, C. E. Cunha, H. Oyaizu, J. Frieman, H. Lin, and E. S. Sheldon. Estimating the redshift distribution of photometric galaxy samples. *Monthly Notices of the Royal Astronomical Society (MNRAS)*, 390(1):118–130, 2008.

[5] M. Sugiyama and K-R. Müller. Model selection under covariate shift. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 235–240. Springer, 2005.

[6] S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 81–88. ACM, 2007.

[7] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh. Sample selection bias correction theory. In *Algorithmic Learning Theory*, pages 38–53. Springer, 2008.

[8] M. Loog. Nearest neighbor-based importance weighting. In *Proceedings of the International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE Press, 2012.

[9] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence. *Dataset Shift in Machine Learning*. MIT Press, 2009.

[10] R. Izbicki, A. B. Lee, and C. M. Schafer. High-dimensional density ratio estimation with extensions to approximate likelihood computation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS). JMLR W&CP*, volume 33, pages 420–429, 2014.

[11] T. Kanamori, S. Hido, and M. Sugiyama. A least-squares approach to direct importance estimation. *Journal of Machine Learning Research (JMLR)*, 10:1391–1445, 2009.

[12] B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[13] M. Sugiyama, T. Suzuki, and T. Kanamori. Density ratio estimation: A comprehensive review. In *Statistical Experiment and Its Related Topics*, number 1703, pages 10–31, 2010.

[14] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

[15] I. Steinwart and A. Christmann. *Support Vector Machines*. Information Science and Statistics. Springer-Verlag, 2008.

[16] S. Ben-David, T. Lu, T. Luu, and D. Pál. Impossibility theorems for domain adaptation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS). JMLR W&CP*, volume 9, pages 129–136, 2010.

[17] S. Ben-David and R. Urner. On the hardness of domain adaptation and the utility of unlabeled target samples. In *Algorithmic Learning Theory*, pages 139–153. Springer, 2012.

[18] M. W. Bern and D. Eppstein. Optimization over zonotopes and training support vector machines. In *Proceedings of the Workshop on Algorithms and Data Structures*, number 2125, pages 111–121. Springer, 2001.

[19] G. H. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, second edition, 1989.

[20] M. Kojima, S. Mizuno, and A. Yoshise. A polynomial-time algorithm for a class of linear complementarity problems. *Mathematical Programming*, 44:1–26, 1989.

[21] I. S. Dhillon. *A New $O(n^2)$ Algorithm for the Symmetric Tridiagonal Eigenvalue Eigenvector Problem*. PhD thesis, University of California at Berkeley, Berkeley, CA, USA, 1998. UMI Order No. GAX98-03176.

[22] S. Arya, D. M. Mount, N. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1994.

[23] N. Halko, P.-G. Martinsson, and J. A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

[24] J. L Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[25] F. Gieseke, J. Heinermann, C. Oancea, and C. Igel. Buffer k-d trees: Processing massive nearest neighbor queries on GPUs. In *Proceedings of the International Conference on Machine Learning (ICML). JMLR W&CP*, volume 32, pages 172–180, 2014.

[26] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.

[27] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[28] H. Aihara, C. A. Prieto, D. An, S. F. Anderson, E. Aubourg, E. Balbinot, T. C. Beers, A. A. Berlind, S. J. Bickerton, and D. et al. Bizyaev. The eighth data release of the sloan digital sky survey: first data from sdss-iii. *The Astrophysical Journal Supplement Series*, 193(2):29, 2011.

[29] J. W. Richards, D. L. Starr, H. Brink, A. A. Miller, J. S. Bloom, N. R. Butler, J. B. James, J. P. Long, and J. Rice. Active learning to overcome sample selection bias: Application to photometric variable star classification. *The Astrophysical Journal*, 744(2), 2012.